Université Abdelmalek Essaâdi
Ecole Nationale des Sciences Appliquées
Al Hoceima

RAPPORT DE STAGE

Streaming Intelligence : Apprentissage Automatique en Temps Réel sur les Flux de Données

Réalisé par:
NACHOUR Ilham

Encadré par:
M. Thierry BERTIN

Année universitaire 2022/2023

Remerciement

Nous tenons à exprimer notre profonde gratitude envers toutes les personnes et les entités qui ont contribué de manière significative à la réussite de notre stage. Leur soutien, leurs conseils et leur assistance ont été précieux tout au long de cette expérience enrichissante.

Tout d'abord, nous souhaitons remercier **Thierry BERTIN** pour nous avoir accueillis au sein de **3D SMART FACTORY**. Votre mentorat et vos précieux enseignements nous ont permis d'acquérir des compétences essentielles et de nous développer professionnellement. Votre patience et votre volonté de partager votre expertise ont grandement contribué à notre apprentissage.

Nous tenons également à remercier l'ensemble de l'équipe de **3D SMART FACTORY** pour leur collaboration et leur générosité. Nous avons eu la chance de travailler avec des collègues exceptionnels qui nous ont fait sentir comme des membres à part entière de l'équipe.

Nous n'oublions pas de remercier nos familles et nos amis pour leur soutien indéfectible tout au long de cette période de stage. Vos encouragements et votre compréhension ont été essentiels pour maintenir notre motivation et notre détermination.

Résumé

Notre projet reflète la mise en place d'une infrastructure robuste destinée à la gestion de données en temps réel issues de sources boursières, mettant en exergue Alpha Vantage en tant que source de données prépondérante.

Nous avons orchestré un pipeline complet, englobant la collecte initiale des données jusqu'à la prédiction de la clôture boursière, en utilisant le modèle ARIMA.

Le socle de notre solution repose sur Apache Kafka, procurant une gestion fluide et fiable des flux de données en temps réel, et garantissant une disponibilité élevée ainsi qu'une scalabilité horizontale.

L'automatisation a été la pierre angulaire de notre approche, embrassant l'intégralité du projet, de la phase initiale de collecte des données à la gestion des pipelines. Cette automatisation a été soigneusement conçue pour enclencher le processus de prédiction dès la détection de nouvelles données sur le site, supprimant ainsi la nécessité d'une intervention humaine continue.

En outre, l'intégration fluide de la plateforme Confluent Cloud a renforcé la fiabilité et la capacité de notre solution.

Dans l'ensemble, ce projet a établi des bases solides pour la prédiction en temps réel des données financières, offrant des opportunités significatives pour des prises de décision plus éclairées dans le domaine boursier.

Nous avons opté pour la réalisation de notre projet une gestion agile avec la méthode Scrum ce qui nous a permis d'atteindre un taux de réussite pour notre projet plus élevé.

Mot clés: ARIMA, Apache Kafka , Automatisation ,Confluent Cloud , Prédiction en temps réel...

Abstract

Our project reflects the implementation of a robust infrastructure for managing real-time data from stock market sources, highlighting Alpha Vantage as the dominant data source.

We orchestrated a complete pipeline, encompassing initial data collection through to stock market close prediction, using the ARIMA model.

The foundation of our solution is based on Apache Kafka, providing fluid and reliable management of real-time data flows, and guaranteeing high availability and horizontal scalability.

Automation was the cornerstone of our approach, embracing the entire project, from the initial data collection phase through to pipeline management. This automation was carefully designed to trigger the prediction process as soon as new data was detected on site, removing the need for ongoing human intervention.

In addition, the seamless integration of the Confluent Cloud platform enhanced the reliability and capacity of our solution.

Overall, this project has laid a solid foundation for real-time prediction of financial data, offering significant opportunities for more informed decision-making in the stock market.

We opted for agile management with the Scrum method for the realization of our project, which enabled us to achieve a higher success rate for our project.

Key words: ARIMA, Apache Kafka , Automation ,Confluent Cloud , Real-time prediction…

Table de Matières

Introduction générale

Au sein de l'Industrie 4.0, une ère caractérisée par la numérisation et l'intégration de bout en bout, notre projet occupe une place stratégique en se concentrant sur l'application du Machine Learning en temps réel sur les flux de données financières. Cette initiative vise à optimiser la prise de décision et l'efficacité opérationnelle dans un domaine où chaque seconde compte.

Notre objectif est de présenter une solution complète de Machine Learning en temps réel, dans laquelle l'automatisation joue un rôle essentiel, permettant une gestion fluide et réactive des données tout au long du processus. Cette automatisation nous permet de capturer, de traiter et de prédire les données en temps réel de manière efficace.

Le présent rapport est structuré en 3 chapitres couvrant l'état d'avancement de notre travail.

- **Chapitre 1** : description de l'organisme d'accueil. Ensuite description du cadre général du projet.

- **Chapitre 2** : description de l'ensemble des notions clés liées au projet et les différentes approches du Machine Learning.

- **Chapitre 3** : description des besoins puis la méthode de conduite adoptée. Suivie d'une architecture technique de notre pipeline.

Chapitre 1 : Contexte Général

1. Introduction :

Le "contexte général de projet" se réfère à une phase initiale et fondamentale dans la gestion d'un projet. Il s'agit de l'étape où l'on recueille, analyse et organise les informations, les besoins, les contraintes et les objectifs qui entourent le projet. Cette démarche vise à obtenir une compréhension complète de l'environnement dans lequel le projet évoluera. Elle permet de définir les contours du projet, d'identifier les parties prenantes, de délimiter les attentes et de poser les bases pour la planification et la mise en œuvre ultérieures du projet. En somme, le contexte général de projet offre une vue d'ensemble qui guide toutes les étapes suivantes du processus de gestion de projet.

Dans ce chapitre, nous allons présenter l'organisme d'accueil **3D SMART FACTORY** . Nous aborderons également la problématique et les objectifs qui encadrent ce projet. Enfin, nous détaillerons la méthodologie de travail qui sera mise en place.

2. Présentation de l'entreprise :

3D SMART FACTORY est une entreprise marocaine récemment établie à Mohammedia. Fondée en 2018, elle est dirigée par **M. Khalil LBBARKI**.

Cette entreprise novatrice se consacre à soutenir les jeunes entrepreneurs dans la promotion de leurs projets et à récompenser leurs idées. Sa principale mission consiste à accompagner les entrepreneurs depuis la phase d'étude jusqu'à la production, en cherchant à harmoniser les aspects économiques et sociaux du développement.

3D SMART FACTORY crée un environnement propice à l'essor des start-ups, en mettant à leur disposition diverses ressources de soutien et de formation dès les premières étapes de leur réflexion ou de leur création. Parmi les services proposés figurent :

- *Encouragement* : La motivation est un élément crucial pour *3D SMART FACTORY*, visant à stimuler l'engagement des entrepreneurs. Cela se traduit par :
  - Une attractivité accrue du travail,
  - Des opportunités de croissance,
  - Un développement personnel renforcé.

- *Encadrement* : Une équipe d'experts assure un accompagnement complet, de la conception initiale au succès du projet. Les domaines couverts comprennent :
  - La gestion de projet,
  - Le coaching,
  - L'établissement d'un plan de travail solide.

- *Financement* : *3D Smart Factory* soutient les porteurs de projets en les aidant à concrétiser leurs idées grâce à une analyse approfondie de leurs besoins, notamment en ce qui concerne les aspects financiers, techniques et réglementaires.

Raison sociale	3D SMART FACTORY
Forme juridique	Société à responsabilité limitée

Date de création	2018
Capital	100000 dhs
Activité	Marchand de dispositifs médicaux
site web	3D Smart Factory (csit.ma)
Siège sociale	75 Lotissement La Gare Etg 01 Mohammedia
Email	3dsmartfactory@gmail.com

Table 1 La fiche d'identité de 3D Smart Factory.

3. Cadre de projet

3. 1. Problématique :

La problématique au cœur de ce projet revêt une importance cruciale dans le domaine de la gestion des données en temps réel, spécifiquement en ce qui concerne les sources boursières, avec Alpha Vantage en tant que protagoniste majeur. Tout d'abord, le défi majeur réside dans l'intégration fluide et précise des flux de données en temps réel provenant de multiples sources boursières. Cette intégration doit garantir la cohérence, l'exactitude et la rapidité pour alimenter un modèle de prédiction, en l'occurrence le modèle ARIMA, qui constitue un autre défi majeur. Ce modèle doit être capable de fournir des prédictions précises de la clôture boursière en temps réel, malgré la volatilité et les fluctuations incessantes des marchés financiers.

En parallèle, le projet doit ériger une infrastructure robuste capable de gérer ces flux de données de manière fiable et continue, en assurant une disponibilité élevée et en pouvant s'adapter à des besoins évolutifs. L'automatisation, qui est au cœur de cette entreprise, doit être soigneusement conçue pour englober l'ensemble du processus, de la collecte initiale des données à la gestion des pipelines de prédiction. L'objectif ultime est d'éliminer la dépendance à l'intervention humaine continue, garantissant une réactivité maximale aux nouvelles données du marché.

L'intégration harmonieuse de la plateforme Confluent Cloud vient renforcer la fiabilité, la sécurité, et la capacité de la solution, tout en simplifiant sa gestion opérationnelle. Enfin, la problématique se résume à la nécessité de fournir une base solide pour la prédiction en temps réel des données financières, ce qui ouvre la porte à des prises de décision plus éclairées dans un environnement boursier caractérisé par sa dynamique et sa compétitivité. La gestion agile du projet, grâce à la méthode Scrum, contribue également à son succès en favorisant la collaboration, la flexibilité, et la capacité à s'adapter aux besoins changeants du projet. En somme, cette problématique englobe un éventail de défis techniques et organisationnels visant à créer une solution performante pour le domaine financier en temps réel.

3. 2. Objectif de projet :

Les objectifs du projet définissent clairement la direction à suivre pour relever les défis inhérents à la gestion des données en temps réel dans le contexte des marchés boursiers. Tout d'abord, l'objectif primordial est de mettre en place une infrastructure robuste qui puisse faire face à la complexité des flux de données en temps réel provenant de diverses sources, en mettant spécialement en avant *Alpha Vantage* comme une source de données cruciale. Cette infrastructure devra assurer une disponibilité élevée, une capacité de montée en charge sans faille, et une gestion fluide des données, éléments essentiels pour la fiabilité de la solution.

Un autre objectif essentiel est de créer un pipeline complet pour la prédiction en temps réel de la clôture boursière en utilisant le modèle *ARIMA*. Cette tâche implique la collecte initiale des données, le traitement, et la création de prévisions précises, malgré la volatilité inhérente aux marchés financiers.

L'automatisation est au cœur de la démarche du projet, avec pour objectif de réduire au maximum l'intervention humaine tout au long du processus, de la collecte initiale des données à la gestion des pipelines de prédiction. Cette automatisation permettra d'assurer une réactivité maximale aux nouvelles données du marché, contribuant ainsi à l'efficacité globale du système.

L'intégration en douceur de la plate-forme Confluent Cloud est également un objectif central, visant à renforcer la fiabilité, la sécurité et la capacité de la solution, tout en simplifiant sa gestion opérationnelle.

Enfin, le projet vise à créer une base solide pour la prédiction en temps réel des données financières, offrant ainsi des opportunités significatives pour des prises de décision plus éclairées dans le domaine boursier. La gestion agile du projet, adoptée grâce à la méthodologie Scrum, sert de cadre pour une collaboration efficace, une flexibilité et une capacité d'adaptation aux besoins changeants du projet, contribuant ainsi à un taux de réussite plus élevé. Dans l'ensemble, ces objectifs convergent vers la résolution de la problématique centrale du projet et l'aboutissement d'une solution performante pour la gestion des données financières en temps réel.

4. Méthodologie de travail :

4. 1. Définition

Scrum est une méthode de développement agile orientée projet informatique dont les ressources sont régulièrement actualisées.

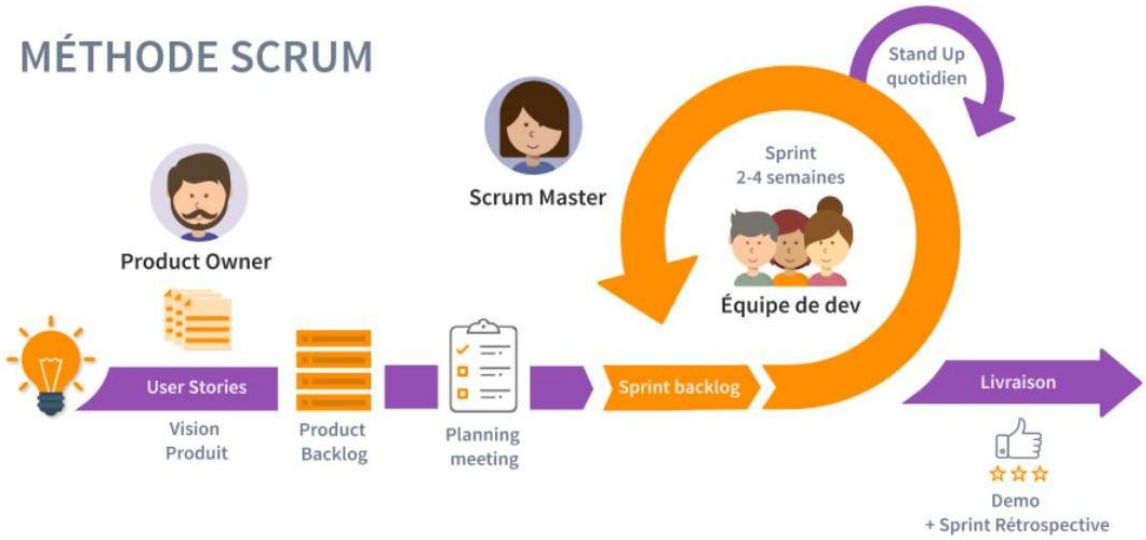


Figure 1 Méthode SCRUM

4. 2. Principes de la méthode Scrum

La méthodologie Scrum a été notre choix pour gérer ce projet de manière organisée et efficace, et elle a joué un rôle central dans la planification, l'exécution et la gestion du travail collectif.

- **Planification Initiale :** Dès le début du stage, nous avons organisé une réunion de démarrage pour définir nos objectifs de projet, les attentes de l'entreprise et les rôles de chaque membre de l'équipe.

- **Backlog du Projet** : Ensemble, nous avons élaboré un backlog complet du projet, répertoriant de manière détaillée les fonctionnalités, les tâches et les livrables à accomplir tout au long du stage. Les éléments du backlog ont été priorisés collectivement en fonction de leur pertinence pour le projet.

- **Planification des Itérations :** Le projet a été divisé en itérations, que nous avons nommées 'sprints', chacune d'une durée de deux semaines. Lors de chaque réunion de planification de l'itération, nous avons travaillé de concert pour sélectionner les tâches à accomplir en fonction des priorités établies. Nous avons également estimé la durée de chaque tâche de manière collaborative pour une planification plus précise.

- **Exécution des Itérations :** Notre équipe a maintenu un esprit de collaboration étroite en organisant des réunions de stand-up quotidiennes, ce qui nous a permis de partager nos progrès, de résoudre collectivement les problèmes et de fournir de l'aide en cas de besoin. Chaque membre de l'équipe a travaillé collectivement sur les tâches assignées pour chaque itération, en suivant les priorités établies ensemble.

- **Revues et Rétrospectives de l'itération :** À la fin de chaque sprint, nous avons tenu des réunions de revue pour évaluer de manière collaborative les tâches terminées et les résultats obtenus. Les réunions de rétrospective nous ont permis de discuter des processus et de formuler des améliorations possibles pour les itérations à venir.

- **Itérations Successives :** Nous avons répété ce cycle de planification, d'exécution, de revue et de rétrospective pour chaque itération du projet, adaptant ensemble le backlog en fonction des nouvelles informations et des besoins changeants.

L'utilisation de la méthodologie Scrum a facilité notre travail en équipe, favorisant une collaboration transparente et efficace pour mener à bien ce projet

ambitieux. Cette approche a également permis la participation active de chaque membre de notre équipe, contribuant ainsi au succès collectif du projet.

5. Conclusion :

En conclusion de ce premier chapitre, nous avons établi le cadre contextuel de notre stage, présenté notre organisme d'accueil, identifié la problématique à laquelle nous nous attaquons, fixé nos objectifs, et exposé notre méthodologie de travail. Ces éléments constituent le socle sur lequel repose notre rapport de stage. Ils nous ont fourni une base solide pour la suite de notre mission, en nous permettant de mieux comprendre le contexte, les enjeux, et les moyens à mettre en œuvre pour atteindre nos objectifs.

Chapitre 2 : Fondements Théoriques

1. Introduction

Après avoir introduit le contexte général dans le premier chapitre, ce chapitre sera dédié à la présentation du cadre théorique du projet. Au cours de cette section, nous allons définir et explorer en détail les divers concepts et notions abordés au sein de ce stage, notamment :

- L'apprentissage automatique (Machine Learning)
- Le temps réel
- Les flux de données
- L'automatisation
- Les modèles *ARIMA*
- Les modèles *XGBoost*
- Les modèles *Random Forest*

2. Machine learning

2.1. Définition

Le Machine Learning, pareillement connu sous le nom d'apprentissage automatique, est une discipline scientifique qui fait partie de l'intelligence artificielle. Son objectif principal est de permettre à des algorithmes de détecter des schémas récurrents, tels que des motifs, au sein de divers ensembles de données, que ces données soient constituées de nombres, de mots, d'images, de statistiques, ou d'autres types d'informations.

2.2. Le fonctionnement de Machine Learning

Le développement d'un modèle de Machine Learning comporte quatre étapes essentielles, généralement supervisées par un Data Scientiste.

La première étape implique la sélection et la préparation d'un ensemble de données d'entraînement. Ces données serviront à enseigner au modèle de Machine Learning comment résoudre le problème spécifique pour lequel il est conçu. Les données peuvent être étiquetées pour indiquer au modèle les caractéristiques à identifier, ou non étiquetées, auquel cas le modèle doit extraire les caractéristiques par lui-même. Une préparation minutieuse, organisation et nettoyage des données sont essentiels pour éviter tout biais dans l'entraînement du modèle, ce qui aurait un impact direct sur ses futures prédictions.

La deuxième étape consiste à choisir l'algorithme à appliquer sur l'ensemble de données d'entraînement. Le choix de l'algorithme dépend du type et du volume des données d'entraînement ainsi que de la nature du problème à résoudre.

La troisième étape est le processus itératif d'entraînement de l'algorithme. Les variables sont soumises à l'algorithme, et les résultats sont comparés à ceux attendus. Les poids et les biais de l'algorithme sont ajustés pour améliorer la précision des résultats. Ce processus se répète jusqu'à ce que l'algorithme produise généralement des résultats corrects. Le modèle de Machine Learning ainsi entraîné est alors prêt à être utilisé.

La quatrième et dernière étape concerne l'utilisation et l'amélioration du modèle. Le modèle est appliqué à de nouvelles données en fonction du problème à résoudre. Par exemple, un modèle de détection de spams est utilisé pour évaluer des e-mails entrants. Pour un aspirateur robot, son modèle de Machine Learning s'enrichit de données provenant de son interaction avec le monde réel, telles que le déplacement des meubles ou l'ajout d'objets dans la pièce. L'efficacité et la précision du modèle peuvent également s'améliorer au fil du temps grâce à cette utilisation continue et à la rétroaction.

2. 3. les principaux algorithmes de Machine Learning

Il existe une grande diversité d'algorithmes de Machine Learning, bien que certains soient plus couramment employés que d'autres. Tout d'abord, différents algorithmes sont adaptés aux données étiquetées.

Les algorithmes de régression, tels que la régression linéaire ou logistique, sont utilisés pour comprendre les relations entre les données. La régression linéaire, par exemple, est employée pour prédire la valeur d'une variable dépendante en fonction de la valeur d'une variable indépendante, comme prédire les ventes annuelles d'un commercial en se basant sur son niveau d'éducation ou son expérience.

La régression logistique, quant à elle, est utile lorsque les variables dépendantes sont binaires. Un autre type d'algorithme de régression appelé machine à vecteur de support est pertinent lorsque la classification des variables dépendantes est plus complexe.

Un autre algorithme de Machine Learning populaire est l'arbre de décision. Cet algorithme établit des recommandations en se basant sur un ensemble de règles de décision déduites de données classifiées. Par exemple, il peut recommander sur quelle équipe de football parier en se basant sur des données telles que l'âge des joueurs ou le taux de victoire de l'équipe.

Pour les données non étiquetées, les algorithmes de "clustering" sont souvent utilisés. Cette méthode vise à identifier des groupes présentant des enregistrements similaires et à les regrouper en fonction de leurs caractéristiques.

Parmi les algorithmes de clustering, on peut citer les K-moyennes, le TwoStep, ou encore le Kohonen.

Les algorithmes d'association, quant à eux, permettent de découvrir des modèles et des relations dans les données, identifiant des règles de type "si / alors", similaires à celles utilisées dans le Data Mining.

Enfin, les réseaux de neurones se présentent sous la forme de structures à plusieurs couches. La première couche prend en charge l'entrée des données, une ou plusieurs couches cachées tirent des conclusions à partir de ces données, tandis que la dernière couche attribue des probabilités à chaque conclusion.

Un réseau de neurones "profond" est composé de multiples couches cachées, chacune affinant les résultats de la précédente, et il est utilisé dans le domaine du Deep Learning.

3. Temps réel :

Le temps réel est un concept informatique fondamental qui se réfère à la capacité d'un système ou d'une application à traiter et à fournir des données ou des résultats instantanément, avec un délai négligeable. En d'autres termes, dans un environnement temps réel, les informations sont traitées et mises à jour au fur et à mesure qu'elles sont générées, sans aucun retard perceptible. Cela signifie que les systèmes en temps réel doivent fonctionner de manière extrêmement efficace pour répondre aux besoins immédiats des utilisateurs ou des processus. Cette approche est cruciale dans de nombreuses applications, telles que le contrôle industriel, la surveillance en temps réel, la finance, les jeux vidéo en ligne, et bien d'autres, où la réactivité et la rapidité de traitement sont essentielles pour assurer des performances optimales et une expérience utilisateur fluide.

4. Flux de données :

Un flux de données, également appelé flux de données en continu ou flux de données en temps réel, fait référence à un flux continu de données numériques qui est généré, transmis et traité en temps réel. Ces flux de données peuvent provenir de diverses sources, telles que capteurs, appareils IoT (Internet des objets), applications en ligne, réseaux sociaux, bases de données, systèmes de surveillance, et bien d'autres.

Voici quelques caractéristiques importantes des flux de données :

- **Temps réel** : Les données sont générées et transmises en continu, souvent sans interruption. Elles sont disponibles instantanément ou presque, ce qui permet de réagir rapidement aux changements ou aux événements en cours.
- **Volume élevé** : Les flux de données peuvent être massifs, générant de grandes quantités d'informations en peu de temps. Le traitement de ces données nécessite souvent des infrastructures et des technologies spéciales.
- **Variété** : Les flux de données peuvent inclure différents types de données, tels que des données structurées, semi-structurées et non structurées. Cela peut inclure des textes, des images, des vidéos, des données géospatiales, des séries temporelles, etc.
- **Vitesse** : Le traitement des flux de données nécessite souvent une grande vitesse, car les données doivent être traitées et analysées en temps réel ou presque en temps réel.
- **Analyse en continu** : Les flux de données sont souvent soumis à des analyses en continu pour extraire des informations utiles, détecter des modèles, effectuer des prédictions, ou prendre des décisions instantanées.
- **Sécurité** : La sécurité des flux de données est cruciale, car les données en temps réel peuvent contenir des informations sensibles. Des mécanismes de sécurité doivent être mis en place pour protéger ces données.

Les entreprises utilisent souvent les flux de données pour prendre des décisions en temps réel, pour la surveillance de systèmes critiques, pour la détection d'anomalies, pour le suivi de la performance, pour la personnalisation des services, et bien d'autres applications. Pour gérer et analyser efficacement les flux de données, elles utilisent des technologies telles que les plates-formes de traitement en continu, les moteurs de

règles, les systèmes de gestion de flux de données, les bases de données en mémoire, et les outils d'analyse de données en temps réel.

5. Automatisation :

5.1. Introduction :

Une révolution dans le domaine de la programmation est l'automatisation de l'exécution de code. Elle permet aux développeurs d'explorer de nouvelles opportunités de développement logiciel et de gagner du temps. Nous examinerons les fondements, les avantages et les applications de l'automatisation de l'exécution de code dans ce rapport .

5.2. Avantages de l'automatisation de l'exécution de code :

Les développeurs, les équipes de développement et les organisations bénéficient de nombreux avantages de l'automatisation de l'exécution du code:

- **Gain de temps :** Les tâches répétées sont automatiques, ce qui permet de se concentrer sur des aspects plus créatifs du développement.
- **Fiabilité accrue :** Les processus automatisés sont moins susceptibles de causer des erreurs humaines, ce qui améliore la qualité et la stabilité du logiciel.
- **Déploiement rapide :** les mises à jour et le déploiement peuvent être effectués plus rapidement, ce qui permet une réactivité accrue aux besoins des utilisateurs.
- **Réduction des coûts :** Les tâches manuelles nécessitent moins de ressources humaines, ce qui peut réduire les coûts opérationnels.
- **La scalabilité :** permet aux systèmes automatisés de s'adapter à des charges de travail variées sans qu'une intervention continue soit nécessaire.

5.3. Applications de l'automatisation d'exécution de code:

Des applications pour l'automatisation de l'exécution de code existent dans divers domaines :

- **DevOps :** Il est essentiel à la mise en œuvre des pratiques DevOps afin d'améliorer la collaboration entre les équipes d'exploitation et de développement.
- **Le cloud computing :** Il facilite le déploiement et la gestion des ressources grâce à l'automatisation.
- **La technologie Internet des objets (IoT) :** Elle facilite la gestion efficace de milliers de dispositifs connectés.
- **Analyse de données :** Elle peut être utilisée pour automatiser le traitement, la visualisation et la génération de rapports de grands ensembles de données.

6. Modèle ARIMA :

Arima représente un outil statistique employé pour examiner les données de séries temporelles ou chronologiques dans le but de réaliser des prévisions.

Une série temporelle est une collection finie de données numériques associées à un indicateur temporel, qui peut être une année, une journée ou même une minute. Habituellement, les séries temporelles sont représentées sous forme de graphiques pour faciliter la visualisation de leur évolution et de leur contexte.

ARIMA, acronyme de "*moyenne mobile autorégressive intégrée*", est un modèle statistique qui permet de déterminer les valeurs intégrées dans les séries temporelles en se basant sur les valeurs précédemment observées. En utilisant ce modèle autorégressif, il est possible d'appliquer des fonctions prédictives à des cas individuels, ce qui s'avère particulièrement utile pour surveiller et anticiper l'évolution d'un phénomène. Ce concept mathématique est couramment utilisé en économétrie et en analyse statistique.

L'algorithme de moyenne mobile intégrée autorégressive est également largement employé dans les domaines liés à l'intelligence artificielle (IA) pour évaluer des transformations ou des changements potentiels. Les modèles d'apprentissage automatique qui exploitent ARIMA incluent des algorithmes spécifiques conçus pour effectuer des prédictions en se basant sur les données temporelles antérieures, l'échelle temporelle en question et ses variations.

L'idéologie derrière ARIMA s'articule autour de trois composants fondamentaux :

- **AutoRegressive (AR)** : La méthode autorégressive utilise un cadre de régression pour prédire une variable en fonction de ses valeurs passées. Un modèle AR d'ordre p peut s'écrire :

  $$= + _ + _ + \ldots + _ +$$

  représente la variable à prédire au temps . est le nombre de décalages temporels utilisés. Les coefficients $_1$, ..., détermine l'influence de chaque variable retardée sur la valeur actuelle de .La valeur de est déterminée par une régression linéaire multiple de p variables indépendantes $_{-1}$, ..., $_{-}$.

- **Moving Average (MA)** : La méthode de la moyenne mobile (MA) vise à capturer la relation entre une observation et les erreurs résiduelles des prédictions précédentes. Dans un modèle MA d'ordre q, il peut être exprimé comme suit :

  $$= + + _ + _ + \ldots + _$$

- Le terme représente l'erreur ou le résidu au temps , qui est la différence entre la valeur réelle de et la valeur prévue basée sur les prédictions précédentes. La valeur est calculée sur la base des erreurs commises par le modèle précédent. Ainsi, chaque terme successif regarde un peu plus loin dans le passé pour incorporer les erreurs commises par ce modèle dans le calcul actuel.

- **Integrated (I)**: est une technique pour rendre les données de séries chronologiques stationnaires, ce qui est une exigence cruciale pour modéliser des données de séries chronologiques. La stationnarité est importante dans l'analyse des séries chronologiques car elle garantit que les propriétés statistiques des données, telles que la moyenne et la variance, restent constantes dans le temps. Cette hypothèse est nécessaire car si le modèle évolue dans le temps, il devient difficile d'estimer les paramètres avec précision. Pour atteindre la stationnarité, une approche courante est la différenciation, qui consiste à calculer les différences entre des observations

consécutives dans la série chronologique. La différenciation aide à stabiliser la moyenne de la série chronologique en éliminant les changements de son niveau, en supprimant efficacement les tendances et la saisonnalité. Dans le modèle ARIMA, le paramètre d représente le nombre de fois où l'opération de différenciation est appliquée à la série chronologique. Chaque opération de différenciation soustrait la valeur de l'observation courante par la valeur décalée _.

En combinant ces trois éléments, un modèle ARIMA est spécifié en utilisant trois paramètres principaux :

- *(ordre AR)* : Le nombre de retards utilisés pour la régression autoregressive.
- *(ordre d'intégration)* : Le nombre de différenciations nécessaires pourrendre la série temporelle stationnaire.
- *(ordre MA)* : Le nombre de retards utilisés pour la moyenne mobile.

Le processus de création d'un modèle ARIMA implique généralement les étapes suivantes :

- **Vérifier la stationnarité des séries chronologiques:** soit en utilisant diagramme de série chronologique ou Graphique ACF et graphique PACF ou bien test Augmented Dickey-Fuller (ADF)
- **rendre la série stationnaire** : soit par la différenciation ou bien élimination de la tendance etc .
- **Identification des ordres** : Utiliser des outils comme les graphiques ACF (fonction d'autocorrélation) et PACF (fonction d'autocorrélation partielle) ou bien Modèle Auto ARIMA.
- **Construction du modèle** : Basé sur les ordres identifiés, construire le modèle ARIMA.
- **Ajustement et évaluation** : Ajuster le modèle aux données historiques et évaluer ses performances en utilisant des métriques telles que l'erreur quadratique moyenne (RMSE) ou l'erreur absolue moyenne (MAE).
- **Prévisions** : Utiliser le modèle ajusté pour faire des prévisions sur les valeurs futures de la série temporelle.

6. Modèle XG Boost

XGBoost, qui est une abréviation de "eXtreme Gradient Boosting", est un modèle très prisé par les experts en science des données. Il a démontré sa supériorité en matière de performances et de vitesse, ce qui lui a permis récemment de dominer les compétitions de machine learning et les hackathons, notamment ceux organisés par Kaggle pour les données structurées ou tabulaires. Il est largement employé pour résoudre une variété de problèmes, que ce soit en classification, régression ou dans les défis courants auxquels font face les entreprises, tout en requérant un minimum de ressources.

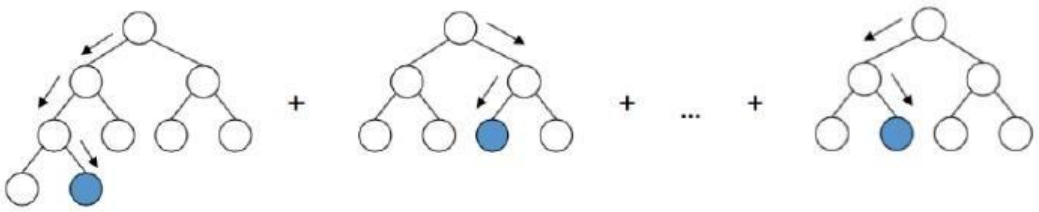


Fig. 2 RN

L'apprentissage d'ensemble, également connu sous le nom d'Ensemble Learning, est un concept du domaine du Machine Learning où l'idée fondamentale consiste à former plusieurs modèles en utilisant le même algorithme d'apprentissage. L'utilisation du terme "ensemble" renvoie à la combinaison de ces modèles individuels pour créer un modèle global plus robuste et plus puissant. Il s'agit essentiellement de réunir des centaines voire des milliers d'apprenants ayant un objectif commun afin de résoudre un problème spécifique. Cette approche vise à améliorer les performances et la stabilité du modèle, à réduire la variance, et à atteindre un niveau de précision bien supérieur à celui qui pourrait être obtenu en utilisant chacun de ces modèles séparément.

L'apprentissage d'ensemble repose sur deux principales méthodes ensemblistes:

a. **Bagging :** *une méthode ensembliste parallèle*

Le bagging, également connu sous le nom de bootstrap aggregating, fonctionne selon les principes suivants :

- Il commence par générer de multiples sous-échantillons aléatoires à partir de l'ensemble de données, avec la possibilité de sélectionner la même valeur

plusieurs fois. Chacun de ces sous-échantillons constitue une base de données distincte pour chaque modèle.

- Ensuite, chaque modèle est entraîné sur l'un de ces sous-échantillons aléatoires afin de créer une prédiction individuelle.
- Le résultat final du modèle est obtenu en sélectionnant la valeur la plus fréquemment prédite, c'est-à-dire celle qui reçoit le plus grand nombre de votes. Dans le cas d'une tâche de régression, la moyenne des prédictions de tous les modèles est utilisée pour obtenir le résultat final.

b. **Boosting :** *une méthode ensembliste séquentielle*

Le boosting génère des modèles qui sont fortement interdépendants, ce qui contraste avec le concept de bagging. En effet, le processus de boosting se déroule de manière itérative comme suit :

- Dans la première étape, un premier modèle de base est créé en utilisant un algorithme préalablement sélectionné, puis il est entraîné sur les données. Initialement, toutes les observations se voient attribuer des poids égaux. À partir des résultats obtenus par ce modèle initial, les poids des observations sont ajustés de telle sorte que si une observation est mal classée, son poids augmente.
- Par la suite, un deuxième modèle est élaboré dans le but de rectifier les erreurs observées dans le premier modèle. Ce deuxième modèle est entraîné en utilisant les données dont les poids ont été ajustés lors de la première étape. Ce processus se répète, avec l'ajout successif de modèles, jusqu'à ce que l'ensemble complet des données d'entraînement soit correctement prédit ou que le nombre maximal de modèles soit atteint.
- Les prédictions finales sont obtenues en combinant les prédictions des modèles précédents, en leur attribuant des poids, avec le dernier modèle ajouté.

Il existe plusieurs modèles basés sur le principe de boosting qui utilisent différentes approches pour calculer les poids. Parmi ces modèles, on peut mentionner : AdaBoost, LPBoost, XGBoost, GradientBoost, et BrownBoost.

Explorons en détail le fonctionnement de l'algorithme sous-jacent à ce modèle. Prenons comme point de départ un classifieur initial faible, noté $_0$. Après avoir optimisé ce classifieur, la méthode de boosting s'efforce de créer un nouveau classifieur faible, noté $_1$, à partir de $_0$ en introduisant un terme de résidu h :

$$() = () + ()$$

Afin que le classifier $_1$ soit plus performant que $_0$. En répétant cette opération un certain nombre de fois, disons p, nous construisons un classifieur final complexe F qui est une combinaison linéaire des classifieurs , où chaque est associé à un poids :

$$() = \sum_{=} ()$$

Le Gradient Boosting est une technique particulièrement puissante lorsque la fonction de perte (qui mesure la différence entre les valeurs théoriques et les valeurs prédites) est différentiable, comme c'est le cas avec une fonction de perte quadratique. Le principe de fonctionnement est le suivant :

- Le processus démarre en initialisant le modèle avec une valeur constante, $_0()$.
- Ensuite, à chaque itération ($1 \leq m \leq M$), les pseudo-résidus sont calculés comme suit :

$$= - \frac{L(, _{m-}())}{_{m-}()}$$

  Ces pseudo-résidus sont essentiellement des erreurs résiduelles permettant de s'approcher davantage de la solution optimale. L'objectif du nouveau modèle est de capturer les observations qui n'ont pas été correctement prédites par les classifieurs précédents.
- Un classifieur faible, h, est ajusté en utilisant les données (,), et un poids associé est déterminé comme suit :

$$_m = mn \sum_{=} L((, _{m-}()) + _m())$$

- Ensuite, le modèle est mis à jour de la manière suivante :

$$_m() = _{m-}() + _m _m()$$

  Où η est le taux d'apprentissage, qui permet de prévenir le sur-apprentissage en régularisant le modèle.

Cet algorithme, lorsqu'il est appliqué aux arbres de régression et de classification, est connu sous le nom de *Gradient Tree Boosting*.

7. Modèle Random Forest:

Random Forest est un algorithme d'apprentissage automatique supervisé utilisé pour résoudre des problèmes de classification et de régression. Il appartient à la famille des méthodes d'ensemble, qui combinent les prédictions de plusieurs modèles individuels pour améliorer la précision et la stabilité.

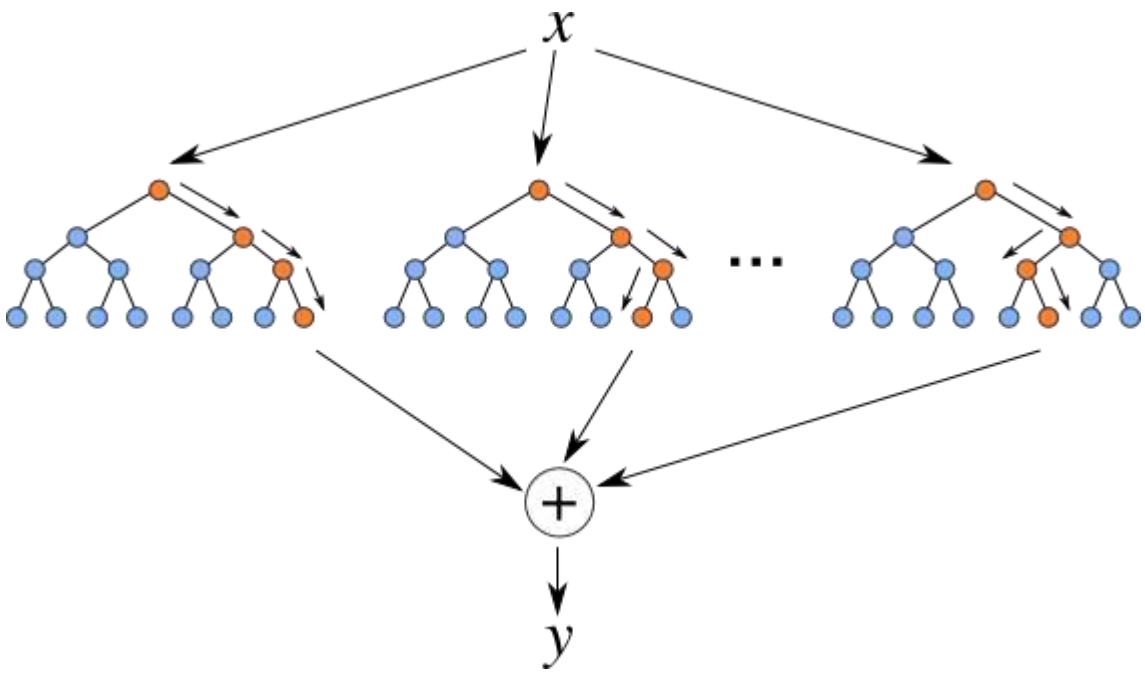


Fig. 3 fig

Comment ça fonctionne ?

a. **Création d'arbres de décision :** Le Random Forest crée un ensemble d'arbres de décision, chacun étant formé sur un sous-ensemble aléatoire de données et de caractéristiques. Cela aide à réduire le surajustement (overfitting) et à améliorer la généralisation.
b. **Bootstrap agrégé (Bagging) :** Le Random Forest utilise une technique appelée "bagging" pour former chaque arbre. Il sélectionne aléatoirement des échantillons avec remplacement à partir de l'ensemble de données pour chaque arbre, ce qui crée une diversité dans les ensembles d'entraînement.
c. **Sous-ensemble de caractéristiques :** Pour chaque arbre, un sous-ensemble aléatoire de caractéristiques est sélectionné à partir de

l'ensemble complet. Cela encourage la diversité entre les arbres et réduit la corrélation entre les prédictions.

d. **Vote majoritaire :** Lors de la prédiction, les résultats de tous les arbres sont agrégés. En classification, le résultat est déterminé par un vote majoritaire. En régression, les prédictions sont en moyenne pour obtenir une valeur finale.

Pourquoi RandomForest ?

- **Performance améliorée** : Les prix des actions sont influencés par de nombreuses variables complexes. En utilisant Random Forest, vous pouvez tirer parti de la puissance de plusieurs arbres de décision pour capturer ces relations complexes et obtenir de meilleures prédictions.
- **Gestion des données** : Random Forest peut gérer les valeurs aberrantes et les données manquantes plus efficacement que les arbres de décision simples, ce qui est important compte tenu de la nature imprévisible des données financières.
- **Réduction du surajustement** : Étant donné que les prix des actions peuvent être volatils et influencés par des facteurs divers, Random Forest peut aider à réduire le risque de surajustement en combinant les prédictions de plusieurs arbres.
- **Analyse des caractéristiques :** Random Forest peut également fournir des informations sur l'importance relative des différentes caractéristiques (variables) dans la prédiction des prix des actions, ce qui pourrait aider à mieux comprendre les tendances du marché.

8. Conclusion :

En conclusion, ce chapitre nous a permis de plonger dans les fondements théoriques essentiels du machine learning appliqué aux problématiques en temps réel avec des flux de données. Nous avons exploré les concepts clés tels que l'automatisation, qui joue un rôle crucial dans la gestion de données en temps réel, ainsi que les algorithmes ARIMA, XGBoost, et Random Forest, qui sont des outils puissants pour la modélisation et la prédiction.

Il est clair que le machine learning offre des opportunités significatives pour améliorer la prise de décision en temps réel dans divers domaines, de la finance à la logistique en passant par la santé. Cependant, il est essentiel de comprendre les bases théoriques pour exploiter pleinement le potentiel de ces méthodes. Les algorithmes ARIMA, XGBoost et Random Forest ont été présentés comme des outils prometteurs, chacun avec ses propres avantages et limitations.

En fin de compte, l'application réussie du machine learning en temps réel repose sur une compréhension approfondie des données, des modèles, et des méthodes appropriées. Dans les chapitres à venir, nous explorerons plus en détail comment mettre en œuvre ces concepts théoriques dans des scénarios pratiques et analyserons les résultats obtenus.

Chapitre 3 : Réalisation et Implémentation

1. Introduction

Ce chapitre constitue le cœur de notre rapport de stage, où nous plongeons au cœur de la mise en œuvre de notre projet. La réalisation de notre mission s'articule autour de six aspects cruciaux.Dans cette section, nous explorerons les outils que nous avons utilisés pour accomplir notre tâche, les détails de la collecte et de la préparation des données, notre processus de sélection de modèle, l'implémentation de ce dernier, le flux de données en temps réel, et enfin, l'automatisation. Chacune de ces étapes est un maillon essentiel de la chaîne qui nous a permis de transformer notre concept initial en une réalité tangible. Dans les sections suivantes, nous détaillerons chacune de ces composantes pour vous offrir un aperçu complet de notre parcours au cours de cette phase de réalisation.

2. Environnement et outils de travail

Pendant notre stage passionnant en équipe, nous avons eu l'opportunité d'explorer un ensemble diversifié d'outils essentiels qui ont grandement contribué à la réussite de nos projets. Parmi ces outils figurent des technologies de pointe telles que Python, Kafka et Jenkins, qui ont joué un rôle central dans notre environnement de travail. Dans cette section, nous allons vous présenter brièvement chacun de ces outils et vous montrer comment nous les avons utilisés pour atteindre nos objectifs de manière efficace et efficience.

2. 1. Langage de programmation

Python est un langage de script de haut niveau, structuré et open source. Il est multi-paradigme et multi-usage. Développé à l'origine par Guido van Rossum en 1989, il est, comme la plupart des applications et outils open source, maintenu par une équipe de développeurs un peu partout dans le monde.Conçu pour être orienté objet, il n'en dispose pas moins d'outils permettant de se livrer à la programmation fonctionnelle ou impérative; c'est d'ailleurs une des raisons qui lui vaut son appellation de « langage agile».

2. 2. Environnement : Google colab

Colab est essentiellement un environnement gratuit de Jupyter, fonctionnant entièrement dans le cloud. Le plus important, c'est que Colab ne nécessite pas de

configuration, et les notebooks que nous créerons peuvent être édités simultanément par les membres d'une équipe de travail. Le plus grand avantage de Colab est qu'il prend en charge les bibliothèques d'apprentissage automatique les plus courantes. Colab, nous permet de :

- Écrire et exécuter du code en Python ;
- Créer, télécharger et partager des carnets de notes ;
- Importer et enregistrer des notebooks depuis et vers Google Drive ;
- Importer et publier des notebooks depuis GitHub ;
- Importer des bases de données externes ;
- Intégration de PyTorch, TensorFlow, Keras, OpenCV ;
- Offrir un service cloud gratuit avec GPU gratuit ;

2. 3. Bibliothèques exploitées

Dans ce paragraphe, nous allons vous présenter brièvement ces bibliothèques Python et vous montrer comment nous les avons utilisées pour atteindre nos objectifs de manière efficace et efficience.

- ***Numpy***

est une bibliothèque Python essentielle pour les calculs numériques, la manipulation de tableaux et de matrices. Elle offre une large gamme de fonctions mathématiques courantes, tout en optimisant les calculs pour une exécution rapide en parallèle. Bien qu'elle excelle dans les calculs, Numpy ne convient pas à l'étiquetage de données structurées, comme les fichiers CSV. De plus, elle permet l'intégration de code provenant de langages tels que C, C++, ou Fortran.

- ***Pandas***

une extension de Numpy, simplifie grandement la manipulation de données structurées. Avec Pandas, vous pouvez aisément accomplir les tâches suivantes :

- ❖ Ajouter de nouvelles colonnes à vos données.
- ❖ Gérer efficacement les valeurs manquantes.
- ❖ Effectuer des opérations de filtrage pour extraire des données spécifiques.
- ❖ Agréger des informations en fonction des colonnes pertinentes.
- ❖ Calculer diverses métriques, notamment la moyenne, la médiane et les sommes.

- ***Matplotlib***

En machine learning et en gestion des données, la représentation graphique des données est essentielle pour filtrer et interpréter les informations. Matplotlib, une bibliothèque Python, offre des capacités de visualisation similaires à Matlab, mais sa nature open source attire de nombreux data scientists vers Python.

- ***Seaborn***

un autre outil de visualisation de données,s'intègre harmonieusement avec Pandas et est souvent utilisé en conjonction avec Matplotlib. En outre, Seaborn est apprécié pour sa fonctionnalité accrue, offrant des visualisations par défaut plus esthétiques. Un autre atout majeur réside dans sa capacité à créer des tableaux de bord, une tâche souvent complexe avec Matplotlib.

- **Scikit-learn**

est une bibliothèque essentielle pour la création de modèles de machine learning. C'est un incontournable pour quiconque se lance dans l'apprentissage machine. Cette bibliothèque offre une vaste gamme de modèles avec des paramètres standard ainsi que de nombreuses variantes, offrant ainsi une grande flexibilité. De plus, Scikit-learn propose une multitude de métriques pour évaluer la performance de vos modèles. Elle prend en charge l'ensemble du processus de développement d'un algorithme de machine learning, de la mise en forme des modèles à la découpe des jeux de données, en passant par la configuration des ensembles de données, l'entraînement des modèles, les tests et l'évaluation de leur performance.

- **Requests**

La bibliothèque requests facilite l'envoi de requêtes HTTP en Python. Elle offre une simplicité d'utilisation tout en proposant de nombreuses fonctionnalités, permettant ainsi de gérer divers aspects tels que l'envoi de

paramètres dans les URL, la personnalisation des en-têtes et la vérification SSL.

2.4. Kafka

Apache Kafka représente une plateforme distribuée spécialisée dans la diffusion continue de données en temps réel. Cette plateforme a la capacité de publier, stocker, traiter et permettre la souscription à des flux d'enregistrements en temps réel. Son objectif est de gérer efficacement des flux de données provenant de multiples sources et de les rendre disponibles à de multiples utilisateurs. En résumé, Apache Kafka ne se limite pas simplement à déplacer d'énormes volumes de données d'un point A à un point B, mais elle peut également le faire entre n'importe quel ensemble de points en fonction de vos besoins, et ce, de manière simultanée si nécessaire.

2.5. Jenkins

Jenkins est un outil d'automatisation bien connu dans le domaine de l'intégration continue et du déploiement continu, comme vous pouvez l'imaginer. Il s'agit d'une application open source écrite en Java qui trouve ses origines dans un projet appelé Hudson, initialement créé par Oracle, puis repris par la fondation Eclipse avant d'être finalement abandonné.

Sa principale utilisation réside dans la création de pipelines CI/CD, permettant aux utilisateurs de se connecter à des systèmes de gestion de code source tels que GitHub. De plus, Jenkins est compatible avec des technologies de conteneurisation telles que Docker et Kubernetes. Il offre également la possibilité d'effectuer diverses autres opérations grâce à une multitude de plug-ins disponibles, couvrant un large éventail d'outils.

La création d'un pipeline Jenkins repose sur un fichier appelé Jenkinsfile, un document texte contenant toutes les étapes nécessaires pour construire le pipeline. Ce fichier Jenkinsfile comprend généralement une section déclarative ainsi qu'une section de script, toutes deux contenant les étapes requises pour rendre le pipeline CI/CD opérationnel.

3. Collecte, prétraitement des données et ingénierie des caractéristiques:

3. 1. Collecte des données

Alpha Vantage est une source de données financières en temps réel qui fournit une gamme complète d'informations sur les marchés financiers, y compris les cotations en temps réel, les données historiques, les rapports financiers et bien d'autres informations utiles pour les investisseurs, les traders et les analystes. En utilisant les services d'Alpha Vantage, les utilisateurs peuvent accéder à des données en temps réel provenant de diverses bourses et marchés, ce qui leur permet de suivre de près les fluctuations des prix, de prendre des décisions d'investissement éclairées et de développer des stratégies financières basées sur des informations actualisées en temps réel. Alpha Vantage est largement utilisé dans le domaine de la finance et de l'analyse des marchés pour ses données fiables et sa facilité d'intégration dans diverses applications et plates-formes financières.

Dans cette section, nous expliquons la manière dont nous avons recueilli les

données du marché boursier d'**IBM** en utilisant l'**API** Alpha Vantage. Cette étape revêt une importance cruciale pour obtenir les données historiques requises en vue de l'analyse et des prédictions de notre projet.

Dans un premier temps, l'URL de la requête API est construite en utilisant le symbole (IBM dans notre cas) et la clé d'API nécessaires à l'authentification. Ensuite, une requête *GET* est envoyée à cette URL grâce à la bibliothèque *requests*, permettant ainsi de récupérer les données financières. Une fois que la réponse de l'API est reçue, elle est traitée et convertie en un objet *JSON* pour une manipulation plus aisée.

```
# Construct the API request URL
url = f'https://www.alphavantage.co/query?function=TIME_SERIES_DAILY&symbol={SYMBOL}&outputsize=full&apikey={API_KEY}'

# Send a GET request to the API
r = requests.get(url)

# Parse the response as JSON
data = r.json()

# Explore the data structure
data.keys()
```

```
dict_keys(['Meta Data', 'Time Series (Daily)'])
```

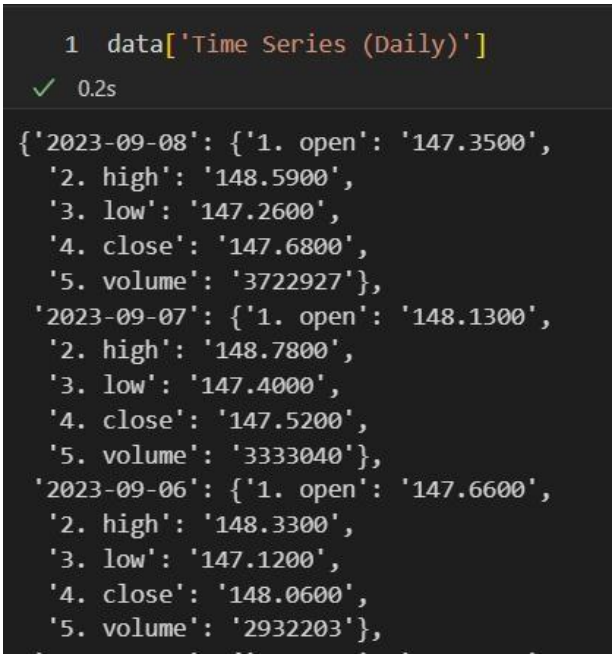
data.keys() donnera les clés de premier niveau du dictionnaire data que nous avons obtenu à partir de la réponse JSON de l'API Alpha Vantage qui sont *Meta Data* et *Time Series (Daily)*.

a. **Meta Data** contiennent des informations sur les données récupérées:
   - Information: Elle décrit la nature des données, qui incluent les prix quotidiens (ouverture, plus haut, plus bas, clôture) et les volumes.
   - Symbol: Elle précise le symbole de l'action, dans ce cas, 'IBM'.
   - Last Refreshed: Elle indique la date à laquelle les données ont été mises à jour pour la dernière fois.
   - Output Size: Elle indique la taille de la sortie, qui est 'Taille complète' dans ce cas.
   - Time Zone: Elle spécifie le fuseau horaire des données, qui est 'US/Eastern'.

```
data['Meta Data']
```

```
{'1. Information': 'Daily Prices (open, high, low, close) and Volumes',
 '2. Symbol': 'IBM',
 '3. Last Refreshed': '2023-09-07',
 '4. Output Size': 'Full size',
 '5. Time Zone': 'US/Eastern'}
```

b. Les données de **Time Series (Daily)** obtenues à partir de l'API Alpha Vantage fournissent un enregistrement détaillé des informations quotidiennes sur le marché boursier de l'action IBM.



Dans le contexte du marché boursier, les termes ***open***, ***high***, ***low***, ***close*** et ***volume*** sont utilisés pour décrire différentes caractéristiques du prix et de l'activité de négociation d'une action ou d'un instrument financier donné au cours d'une journée de trading:

- **Open** : *le prix d'ouverture*, est le prix auquel une action ou un instrument financier a commencé à être négocié au début d'une séance de trading (par exemple, une journée). Il s'agit du premier prix auquel les acheteurs et les vendeurs ont convenu de réaliser une transaction. Le prix d'ouverture peut être un point de référence important pour comprendre le sentiment initial des traders au début de la journée de trading.
- **High**: *le prix le plus haut,* représente le prix le plus élevé auquel une action a été négociée au cours de la séance de trading. Il indique la valeur maximale atteinte par l'action au cours de la journée. Les traders portent souvent une attention particulière au prix le plus élevé car il montre la limite supérieure du mouvement des prix au cours de cette séance.
- **Low**: l*e prix le plus bas,* est le prix le plus bas auquel une action a été négociée au cours de la séance de trading. Il reflète la valeur minimale

atteinte par l'action au cours de la journée. Les traders surveillent le prix le plus bas pour comprendre la limite inférieure du mouvement des prix au cours de cette séance.

- **Close** : *le prix de clôture,* est le dernier prix auquel une action a été négociée avant la fin de la séance de trading. Il symbolise la dernière transaction de la journée. Le prix de clôture est souvent considéré comme significatif car il est utilisé comme point de référence pour calculer les gains ou les pertes, et il est également le prix utilisé pour déterminer la variation quotidienne de la valeur.
- **Volume** : *le volume,* représente le nombre total d'actions ou de contrats négociés au cours de la séance de trading. Il quantifie le niveau d'activité et d'intérêt pour une action particulière. Un volume élevé peut indiquer une forte participation et un intérêt marqué sur le marché, tandis qu'un volume faible pourrait suggérer un intérêt moindre ou un manque de conviction parmi les traders.

3. 2. Prétraitement des données

Dans cette section, nous décrivons les étapes de prétraitement appliquées aux données brutes obtenues à partir de l'API Alpha Vantage afin de les préparer pour l'analyse et la modélisation:

- **Création d'un DataFrame :** Les données brutes sont initialement stockées sous forme de dictionnaire. Nous créons un DataFrame Pandas à partir des données de la série temporelle (quotidienne) pour faciliter la manipulation des données.

```
1  df = pd.DataFrame.from_dict(data['Time Series (Daily)']).T
2  df
```

✓ 0.2s

	1. open	2. high	3. low	4. close	5. volume
2023-09-08	147.3500	148.5900	147.2600	147.6800	3722927
2023-09-07	148.1300	148.7800	147.4000	147.5200	3333040
2023-09-06	147.6600	148.3300	147.1200	148.0600	2932203
2023-09-05	147.9100	149.0000	147.5719	148.1300	3731281
2023-09-01	147.2600	148.1000	146.9200	147.9400	2727796
...	...	...	...	...	...
1999-11-05	92.7500	92.9400	90.1900	90.2500	13737600
1999-11-04	94.4400	94.4400	90.0000	91.5600	16697600
1999-11-03	95.8700	95.9400	93.5000	94.3700	10369100
1999-11-02	96.7500	96.8100	93.6900	94.8100	11105400
1999-11-01	98.5000	98.8100	96.3700	96.7500	9551800

6002 rows × 5 columns

- **Réorganisation du DataFrame :** Pour nous assurer que les données sont dans l'ordre chronologique, nous inversons l'ordre des lignes dans le DataFrame.
- **Nettoyage des libellés de colonnes :** Nous nettoyons les libellés de colonnes en supprimant les préfixes numériques ("1. open", "2. high", etc.) pour améliorer la lisibilité.
- **Conversion des Types de Données :** Les colonnes numériques sont converties en types de données numériques appropriés, et la colonne "date" est convertie en type de données datetime pour faciliter l'analyse basée sur les dates.

```python
# Reordering the DataFrame
df = df.reindex(index=df.index[::-1])

# Column Label Cleanup
df.columns = [col.split('. ')[1] for col in df.columns]
df.reset_index(inplace=True)
df.rename(columns={'index':'date'}, inplace=True)

# Data Type Conversion
df['date'] = pd.to_datetime(df['date'])
df['open'] = pd.to_numeric(df['open'])
df['high'] = pd.to_numeric(df['high'])
df['low'] = pd.to_numeric(df['low'])
df['close'] = pd.to_numeric(df['close'])
df['volume'] = pd.to_numeric(df['volume'])

display(df)
```

✓ 0.1s

	date	open	high	low	close	volume
0	1999-11-01	98.50	98.81	96.3700	96.75	9551800
1	1999-11-02	96.75	96.81	93.6900	94.81	11105400
2	1999-11-03	95.87	95.94	93.5000	94.37	10369100
3	1999-11-04	94.44	94.44	90.0000	91.56	16697600
4	1999-11-05	92.75	92.94	90.1900	90.25	13737600
...	...	...	...	...	...	...
5997	2023-09-01	147.26	148.10	146.9200	147.94	2727796
5998	2023-09-05	147.91	149.00	147.5719	148.13	3731281
5999	2023-09-06	147.66	148.33	147.1200	148.06	2932203
6000	2023-09-07	148.13	148.78	147.4000	147.52	3333040
6001	2023-09-08	147.35	148.59	147.2600	147.68	3722927

6002 rows × 6 columns

- **Gestion des Valeurs Manquantes :** Nous vérifions la présence de valeurs manquantes dans le DataFrame et confirmons qu'il n'y a pas de données manquantes.

```
1  df.isnull().sum()
✓ 0.0s

date      0
open      0
high      0
low       0
close     0
volume    0
dtype: int64
```

3. 3. Ingénierie des caractéristiques

C’est le processus de sélection, de manipulation et de transformation des données brutes en caractéristiques pertinentes qui peuvent être utilisées dans les modèles d'apprentissage automatique. L'objectif est d'améliorer la performance et l'efficacité des modèles prédictifs en fournissant des informations utiles pour la tâche d'apprentissage supervisé.

Dans cette section, nous réalisons de l'ingénierie des caractéristiques sur notre jeu de données pour créer des fonctionnalités pertinentes et le préparer à la modélisation d'apprentissage automatique.

- **Fonctionnalités retardées** (*3 derniers jours*) **:** Ces caractéristiques reflètent le comportement historique des prix et des volumes au cours des trois derniers jours. Elles peuvent être utilisées pour modéliser comment les variations récentes des prix sont liées aux prix des jours précédents.

```
1  for lag in range(1, 4):
2      for col in ['open', 'high', 'low', 'close', 'volume']:
3          df[f'{col}_lag_{lag}'] = df[col].shift(lag)
✓ 0.0s
```

- **Indicateurs techniques** (*moyennes mobiles, RSI...*) : les indicateurs techniques fournissent un aperçu des tendances du marché et des conditions de surachat/survente. Nous pouvons utiliser ces indicateurs comme signaux pour entrer ou sortir de transactions. Par exemple:

- ***Moyennes mobiles*** : les croisements entre les moyennes mobiles à court et à long terme peuvent être utilisés comme indicateurs de changement de tendance.
- ***RSI*** : des valeurs extrêmes (supérieures à 70 ou inférieures à 30) peuvent indiquer des points d'inversion potentiels.

```python
def calculate_rsi(close_prices, window=14):
    diff = close_prices.diff()
    gain = diff.where(diff > 0, 0)
    loss = -diff.where(diff < 0, 0)

    avg_gain = gain.rolling(window=window, min_periods=1).mean()
    avg_loss = loss.rolling(window=window, min_periods=1).mean()

    rs = avg_gain / avg_loss
    rsi = 100 - (100 / (1 + rs))
    return rsi

df['sma_14'] = df['close'].rolling(window=14).mean()
df['rsi'] = calculate_rsi(df['close'], window=14)
```

- **la variation en pourcentage:** nous calculons la variation en pourcentage du prix de clôture et du volume de transactions, ce qui offre des informations sur les mouvements quotidiens des prix et l'activité de négociation.

```python
df['close_pct_change'] = df['close'].pct_change()  # Percentage change in closing price
df['volume_change'] = df['volume'].pct_change()  # Percentage change in volume
```

- **Adding Date-related Features:** Nous pouvons extraire des informations supplémentaires liées à la date de la colonne date, telles que le jour de la semaine, le mois et l'année, ce qui peut fournir des informations précieuses. Certains jours ou mois peuvent présenter des schémas différents.

```python
df['day_of_week'] = df['date'].dt.dayofweek  # Monday: 0, Sunday: 6
df['month'] = df['date'].dt.month
df['year'] = df['date'].dt.year
```

- **Rapports de prix :** Les ratios de prix offrent un aperçu des relations entre les prix. Par exemple:
  - ***Ratio haut-bas*** : un ratio élevé peut indiquer une volatilité accrue, tandis qu'un ratio faible peut suggérer des mouvements de prix stables.

- ***Ratio clôture-ouverture*** : un ratio supérieur à 1 peut indiquer une tendance haussière (fermeture > ouverture), tandis qu'un ratio inférieur à 1 peut suggérer une tendance baissière.

```
df['high_low_ratio'] = df['high'] / df['low']
df['close_open_ratio'] = df['close'] / df['open']
```

4. Implémentation des modèles

Cette étape décrit comment nous élaborons, formons et évaluons des modèles de prédiction des prix des actions. Nous explorerons trois modèles différents : ARIMA, Random Forest et XGBoost.

4. 1. Random Forest

A. Sélection des fonctionnalités :

- **Analyse de corrélation:** Pour bien assurer une performance maximale de notre model il s'avère qu'il est fondamental de tester la corrélation et la dépendance de chaque feature avec notre cible et entre eux au même temps.

	close
close	1.000000
high	0.999544
low	0.999535
open	0.999068
close_lag_1	0.998444
low_lag_1	0.998082
high_lag_1	0.998074
open_lag_1	0.997601
close_lag_2	0.996959
low_lag_2	0.996626
high_lag_2	0.996600
open_lag_2	0.996155
close_lag_3	0.995475
low_lag_3	0.995162
high_lag_3	0.995122
open_lag_3	0.994687
sma_14	0.993938
year	0.544929
volume_lag_1	0.385927
volume	0.385754
volume_lag_2	0.385118
volume_lag_3	0.384339
high_low_ratio	0.321073
rsi	0.094316
month	0.027835
close_pct_change	0.022102
volume_change	0.016035
close_open_ratio	0.013243
day_of_week	0.000574

Les caractéristiques ayant une corrélation absolue élevée (proche de 1) avec la variable cible ("**close**") sont celles qui sont le plus linéairement liées à la cible. Dans notre cas, les prix "high", "low", "open", SMA et décalés montrent des corrélations extrêmement élevées.

- **Importance des Caractéristiques:** Des caractéristiques telles que "low" et "high" ont une importance plus élevée, ce qui suggère qu'elles jouent un rôle important dans la prédiction de notre cible. les autres fonctionnalités peuvent avoir un pouvoir prédictif limité.

```
Feature: low, Importance: 0.6091970878395111
Feature: high, Importance: 0.387659426831245
Feature: open, Importance: 0.0026723087709098923
Feature: close_open_ratio, Importance: 0.00022936117570642326
Feature: close_pct_change, Importance: 7.476564799021737e-05
Feature: high_low_ratio, Importance: 1.936446411198369e-05
Feature: close_lag_1, Importance: 1.2482502618730036e-05
Feature: low_lag_2, Importance: 1.096420040447212e-05
Feature: rsi, Importance: 9.801998117353814e-06
Feature: volume_lag_2, Importance: 8.55002395360274e-06
Feature: volume, Importance: 8.004298255404898e-06
Feature: low_lag_1, Importance: 7.919866845924e-06
Feature: volume_lag_1, Importance: 7.748588940144348e-06
Feature: volume_lag_3, Importance: 7.5918526984298635e-06
Feature: high_lag_1, Importance: 7.4852053353608304e-06
Feature: sma_14, Importance: 7.194773345843924e-06
Feature: volume_change, Importance: 7.146120423123204e-06
Feature: close_lag_2, Importance: 6.7501654493185095e-06
Feature: open_lag_1, Importance: 6.06996290458518e-06
Feature: open_lag_3, Importance: 5.2816168390654004e-06
Feature: low_lag_3, Importance: 4.991261460627499e-06
Feature: month, Importance: 4.979518916456889e-06
Feature: close_lag_3, Importance: 4.80318548055288e-06
Feature: high_lag_2, Importance: 4.46274503904151e-06
Feature: open_lag_2, Importance: 4.2720912624176245e-06
Feature: high_lag_3, Importance: 4.175825563347415e-06
Feature: year, Importance: 3.6140987372539674e-06
Feature: day_of_week, Importance: 3.395367934417803e-06
```

- **Élimination Récursive des Caractéristiques (RFE):** La méthode RFE supprime de manière récursive les variables les moins importantes lors d'entraînement du modèle et classe les fonctionnalités en fonction de leur importance.

  Nous avons appliqué cette méthode avec une *Régression Linéaire* pour sélectionner les caractéristiques les plus importantes:

```
Selected Features: Index(['open', 'high', 'low', 'high_lag_3', 'low_lag_3', 'close_pct_change',
       'high_low_ratio', 'close_open_ratio'],
      dtype='object')
```

  Les caractéristiques sélectionnées comprennent des éléments liés aux prix ("open", "high", "low") ainsi que des caractéristiques retardées ("high_lag_3", "low_lag_3"). Ces caractéristiques sont considérées comme cruciales par le modèle pour la prédiction.

- **la multicolinearité:** Avant d'entrer dans la phase de modélisation, nous devons aborder le problème de la multicolinéarité. Nous avons observé des valeurs de corrélation élevées, ce qui peut indiquer une forte multicolinéarité, suggérant ainsi que certaines caractéristiques pourraient contenir des informations redondantes.

  Pour atténuer les effets de la multicolinéarité, notre approche consiste à éliminer la caractéristique la moins significative parmi celles qui présentent les corrélations les plus élevées. Après une analyse minutieuse de la corrélation et de l'importance des caractéristiques, nous avons choisi de supprimer la variable *open*. Les variables prédictives essentielles restantes sont : *low, high, close_lag_1, low_lag_3, high_lag_3, close_pct_change, high_low_ratio et close_open_ratio*.

B. Entraînement de modèle:

Tout d'abord, nous sélectionnons les caractéristiques pertinentes à partir d'ensemble de données pour construire la matrice d'entraînement (X) et définit le prix de clôture comme la variable cible (y). Ensuite, nous divisons les données en ensembles d'entraînement et de test pour évaluer la performance du modèle. Le modèle de régression Random Forest est ensuite créé et ajusté aux données d'entraînement avec des hyperparamètres spécifiés. Après l'entraînement, le modèle est utilisé pour effectuer des prédictions sur l'ensemble de test.

Select the desired features for X:

```
X = df_rf[['low', 'high', 'low_lag_3',
           'high_lag_3', 'close_lag_1',
           'close_pct_change', 'high_low_ratio',
           'close_open_ratio']]
```

Define the target variable (y):

```
y = df_rf['close']
```

Split the data into training and testing sets:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)
```

Create and fit the Random Forest Regressor model:

```
rfr_model = RandomForestRegressor(n_estimators=100, max_depth=20,
                                  min_samples_split=10, min_samples_leaf=1, random_state=1)

rfr_model.fit(X_train, y_train)
```

Make predictions:

```
y_pred = rfr_model.predict(X_test)
```

Enfin, diverses métriques de performance telles que *le MSE, le MAE, le RMSE, le MAPE et le R-squared* sont calculées et affichées pour évaluer la précision du modèle. Ces métriques permettent de déterminer à quel point le modèle peut prédire avec succès les valeurs de clôture des données financières, ce qui est essentiel pour la prise de décision en matière d'investissement ou d'analyse de marché.

Report performance metrics:

```
mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
rmse = math.sqrt(mse)
mape = np.mean(np.abs(y_pred - y_test) / np.abs(y_test))
r2 = r2_score(y_test, y_pred)

print("MSE: {:.3f}".format(mse))
print("MAE: {:.3f}".format(mae))
print("RMSE: {:.3f}".format(rmse))
print("MAPE: {:.3f}%".format(mape * 100))
print("R-squared: {}".format(r2))
```

```
MSE: 0.351
MAE: 0.407
RMSE: 0.593
MAPE: 0.341%
R-squared: 0.9997150955889714
```

4. 2. XGBoost

Nous avons suivi des étapes similaires pour développer le modèle XGBoost, en préparant les données, en choisissant les caractéristiques pertinentes, en divisant les données en ensembles d'entraînement et de test, et en ajustant les hyperparamètres pour obtenir les meilleures performances. Après avoir entraîné le modèle XGBoost, nous avons évalué sa performance en utilisant des métriques d'évaluation clés. Les résultats de notre modèle XGBoost ont montré une performance prometteuse comparables à ceux de notre modèle Random Forest.

Select the desired features for X:

```
X=df_xgb[['low','high','open_lag_1',
      'low_lag_1','sma_14','close_open_ratio',
      'open','high_lag_1','close_pct_change']]
```

Define the target variable (y):

```
y=df_xgb['close']
```

Split the data into training and testing sets:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Create and fit the XGBoost model:

```
xgb_model = xgb.XGBRegressor(n_estimators=500,
                             max_depth=3,
                             learning_rate=0.5,
                             objective='reg:squarederror',
                             random_state=42)

xgb_model.fit(X_train, y_train)
```

Make predictions:

```
y_pred = xgb_model.predict(X_test)
```

Report performance metrics:

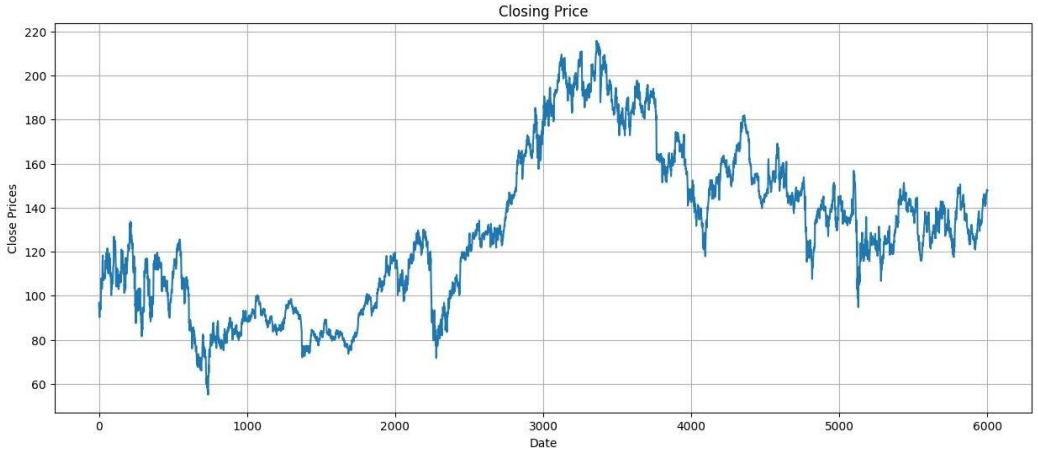
```
mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
rmse = math.sqrt(mse)
mape = np.mean(np.abs(y_pred - y_test) / np.abs(y_test))
r2 = r2_score(y_test, y_pred)

print("MSE: {:.3f}".format(mse))
print("MAE: {:.3f}".format(mae))
print("RMSE: {:.3f}".format(rmse))
print("MAPE: {:.3f}%".format(mape * 100))
print("R-squared: {}".format(r2))
```

```
MSE: 0.278
MAE: 0.369
RMSE: 0.528
MAPE: 0.309%
R-squared: 0.9997833834156734
```

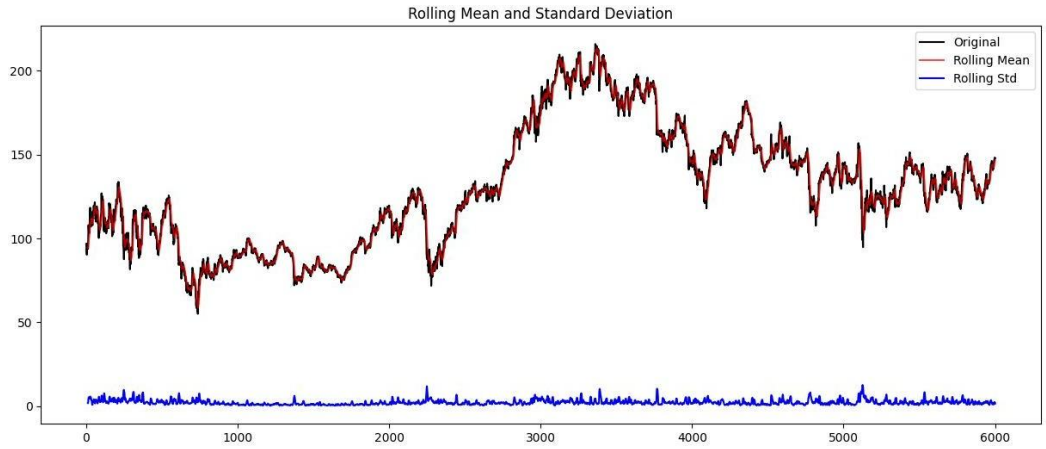
4. 3. ARIMA

Pour **implémenter** ARIMA nous avons passer par les étapes suivantes :

- **Visualisation des données** : Les données de prix de clôture sont tracées dans un graphique pour visualiser la tendance



- **Vérification de la Stationnarité** : La stationnarité des données est vérifiée à l'aide du test Augmented Dickey-Fuller (ADF).



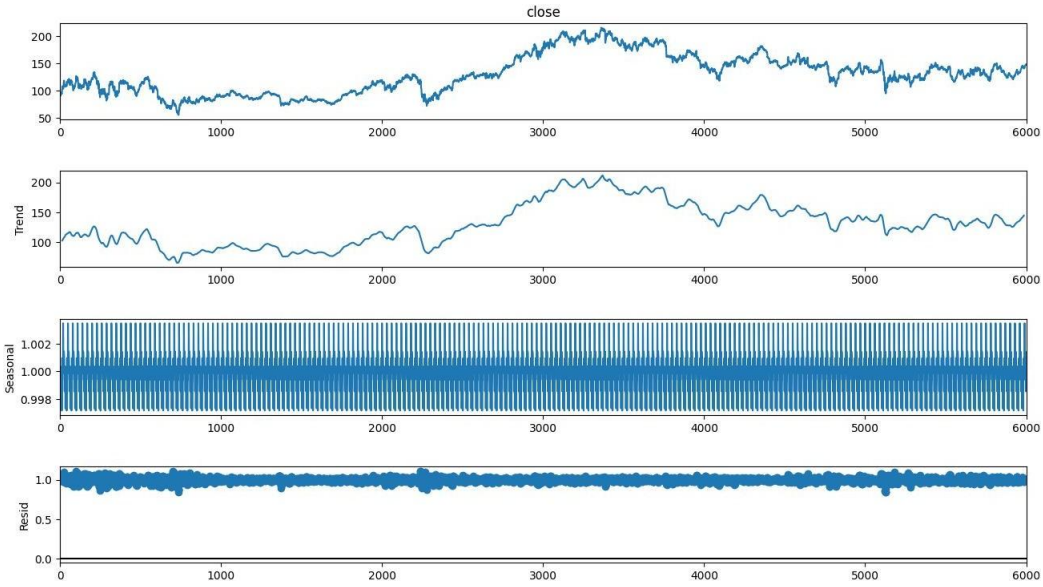
- **Test de Dickey-Fuller:**

```
Results of Dickey-Fuller test
Test Statistics                -1.759231
p-value                         0.400882
No. of lags used               25.000000
Number of observations used  5976.000000
Critical Value (1%)            -3.431445
Critical Value (5%)            -2.862024
Critical Value (10%)           -2.567028
dtype: float64
```

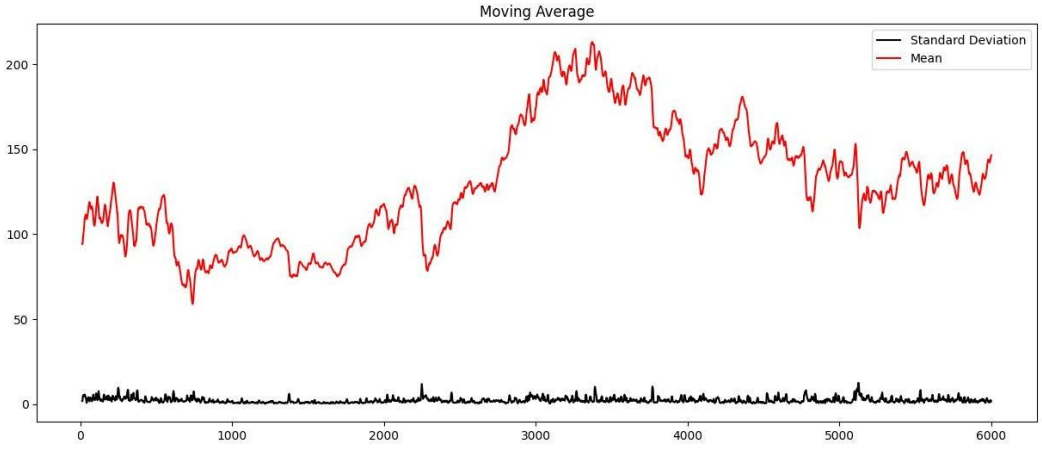
- ***p-valeur*** est un élément essentiel du test de Dickey-Fuller. Elle mesure la probabilité que les résultats observés de la statistique du test se produisent par hasard. Si la valeur p est inférieure à un certain seuil (généralement 0.05), on peut rejeter l'hypothèse nulle selon laquelle la série n'est pas stationnaire. Dans ce cas, la valeur p est de 0.40. Cela suggère que la série temporelle n'est pas stationnaire.
- ***No. of lags used:*** Cela fait référence au nombre de retards (lags) utilisés dans le modèle ARIMA pour corriger les tendances temporelles. Dans ce cas, 25 retards ont été utilisés.
- ***Number of observations used:*** Il s'agit du nombre d'observations de la série temporelle utilisées pour effectuer le test de Dickey-Fuller. Dans ce cas, 5974 observations ont été prises en compte.
- ***Critical Values (1%, 5% et 10%):*** Ce sont les valeurs critiques qui servent de seuils pour évaluer si la série temporelle est stationnaire. Si la statistique du test est inférieure à ces valeurs critiques, cela suggère que la série est stationnaire au niveau de signification correspondant (1%, 5%, 10%). Dans

ce cas, la statistique du test est supérieure aux valeurs critiques .Donc la série n'est pas stationnaire .

-**Décomposition saisonnière** : Les données sont décomposées en composantes de niveau, de tendance, de saisonnalité et de bruit pour mieux comprendre leur structure.



- **Élimination de la tendance** : Une moyenne mobile est calculée pour éliminer la tendance et rendre les données stationnaires.



- **Division des données** : Les données sont divisées en ensembles d'entraînement et de test pour la modélisation

```
train_size = int(len(df_arima) * 0.9)
train_data, test_data = df_arima[:train_size], df_arima[train_size:]
```

- **Identification des ordres** : La fonction ***auto_arima*** est utilisée pour déterminer automatiquement les paramètres optimaux du modèle ARIMA en effectuant des tests de différenciation et en ajustant les ordres p, d et q.

```
model_autoARIMA = auto_arima(train_data['close'],
                             start_p=0, start_q=0,
                             test='adf',
                             max_p=3, max_q=3,
                             m=1,
                             d=None,
                             seasonal=False,
                             start_P=0,
                             D=0,
                             trace=True,
                             error_action='ignore',
                             suppress_warnings=True,
                             stepwise=True)
```

✓ 4.2s

```
Performing stepwise search to minimize aic
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=22792.050, Time=0.20 sec
 ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=22788.489, Time=0.42 sec
 ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=22788.520, Time=0.44 sec
 ARIMA(0,1,0)(0,0,0)[0]             : AIC=22790.129, Time=0.11 sec
 ARIMA(2,1,0)(0,0,0)[0] intercept   : AIC=22790.450, Time=0.54 sec
 ARIMA(1,1,1)(0,0,0)[0] intercept   : AIC=22790.411, Time=0.46 sec
 ARIMA(2,1,1)(0,0,0)[0] intercept   : AIC=22792.455, Time=0.91 sec
 ARIMA(1,1,0)(0,0,0)[0]             : AIC=22786.573, Time=0.08 sec
 ARIMA(2,1,0)(0,0,0)[0]             : AIC=22788.534, Time=0.25 sec
 ARIMA(1,1,1)(0,0,0)[0]             : AIC=22788.495, Time=0.21 sec
 ARIMA(0,1,1)(0,0,0)[0]             : AIC=22786.604, Time=0.17 sec
 ARIMA(2,1,1)(0,0,0)[0]             : AIC=22790.540, Time=0.34 sec

Best model:  ARIMA(1,1,0)(0,0,0)[0]
Total fit time: 4.209 seconds
```

Les paramètres optimaux du modèle ARIMA sont: p,d,q = 1,1,0

- **Prédictions du modèle** : Les prédictions du modèle sont générées en utilisant la méthode ARIMA avec les paramètres optimaux trouvés par auto_arima

```
arima_model = ARIMA(history, order=order)
arima_model_fit = arima_model.fit()
```

0.1s

```
for time_point in range(N_test_observations):
    # Get the true test value
    true_test_value = test_data[time_point]

    # Update the history with the true test value
    history.append(true_test_value)

    # Re-fit the ARIMA model with the updated history
    arima_model = ARIMA(history, order=order)
    arima_model_fit = arima_model.fit()

    # Forecast the next value
    yhat = arima_model_fit.forecast(steps=1)[0]

    # Append the prediction to the list
    model_predictions.append(yhat)
```

- **Évaluation du modèle** : Les performances du modèle sont évaluées en calculant *le MSE, le MAE, le RMSE, le MAPE et le R-squared*

```
mse = mean_squared_error(test_data, model_predictions)
mae = mean_absolute_error(test_data, model_predictions)
rmse = math.sqrt(mse)
mape = np.mean(np.abs(model_predictions - test_data) / np.abs(test_data))
r2 = r2_score(test_data, model_predictions)

print('MSE: {:.3f}'.format(mse))
print('MAE: {:.3f}'.format(mae))
print('RMSE: {:.3f}'.format(rmse))
print('MAPE: {:.3f}%'.format(mape * 100))
print("R-squared: {}".format(r2))
```

✓ 0.0s

```
MSE: 0.003
MAE: 0.036
RMSE: 0.052
MAPE: 0.027%
R-squared: 0.9999579660699509
```

5. Choix du modèle

Dans le cadre de notre stage, le choix du modèle de prévision a été une étape cruciale pour atteindre des résultats de haute qualité. Après une analyse approfondie des différentes approches disponibles, *nous avons décidé d'opter pour le modèle ARIMA*. Cette décision repose sur des données empiriques solides, notamment une erreur quadratique moyenne (**MSE**) exceptionnellement faible, mesurée à seulement 0.03. En comparaison avec des alternatives telles que XGBoost et Random Forest, ARIMA s'est avéré être la solution la plus adaptée à notre problème de prévision. Cette petite MSE témoigne de la capacité d'ARIMA à capturer les tendances et les motifs sous-jacents dans nos données, ce qui est essentiel pour obtenir des prévisions précises et fiables. Nous sommes confiants que ce choix stratégique contribuera de manière significative à la réussite de notre projet et à l'atteinte de nos objectifs de prévision.

En outre, le choix d'ARIMA s'aligne également avec la nature de nos données et les caractéristiques de notre problème. ARIMA est particulièrement bien adapté aux séries temporelles, ce qui est souvent le cas dans des domaines tels que la finance, l'économie et la météo, où les observations dépendent du temps. En prenant en compte les valeurs passées pour prédire les valeurs futures, ARIMA exploite la dépendance temporelle inhérente à nos données, ce qui le distingue des méthodes d'apprentissage automatique telles que XGBoost et Random Forest.

De plus, ARIMA offre l'avantage de la simplicité et de l'interprétabilité. Contrairement à des modèles plus complexes comme XGBoost et Random Forest, ARIMA ne nécessite pas de paramétrage complexe ni de traitement intensif des caractéristiques, ce qui facilite grandement son utilisation et son interprétation. Cette simplicité permet également une meilleure compréhension des mécanismes sous-jacents qui influencent nos prévisions.

En résumé, le choix d'ARIMA comme modèle de prédiction repose sur des preuves solides, telles qu'une MSE exceptionnellement faible, une adéquation aux caractéristiques de nos données et une simplicité d'utilisation. Nous sommes convaincus que ce choix nous permettra d'obtenir des résultats précis et utiles pour notre projet de stage, tout en offrant une base solide pour nos futures analyses et décisions.

6. La partie streaming du projet

6. 1. Présentation de la plate-forme Kafka utilisé

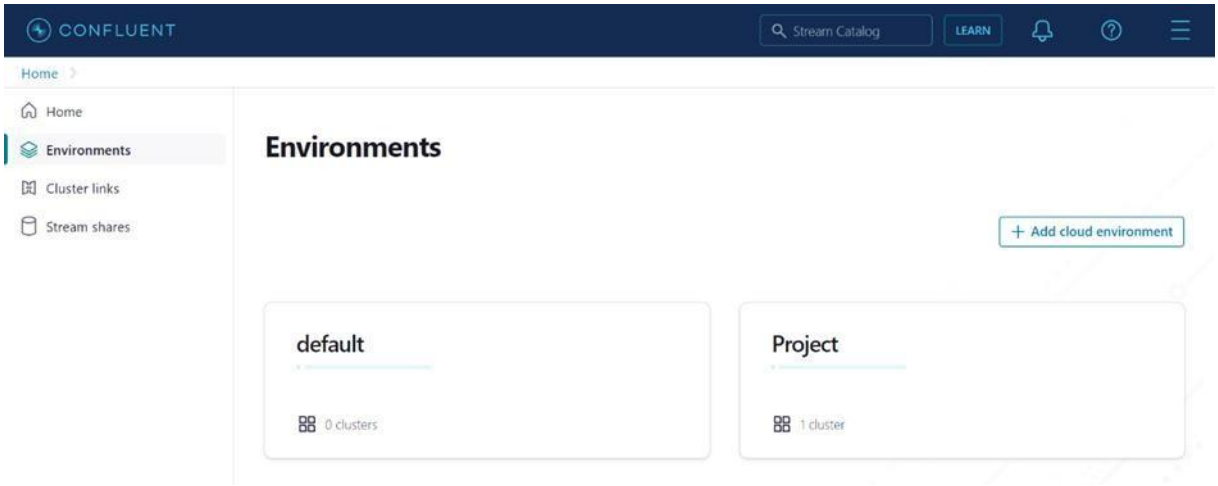
La plate-forme que nous avons décidé d'utiliser est ***Confluent Cloud*** qui est une plate-forme gérée proposée par *Confluent*, la société derrière *Apache Kafka*, qui permet de créer et de gérer facilement un environnement Kafka dans le cloud.

Le platform que la société offre présente une option sans charge qui est bénéfique pour notre projet et pour sa continuité.

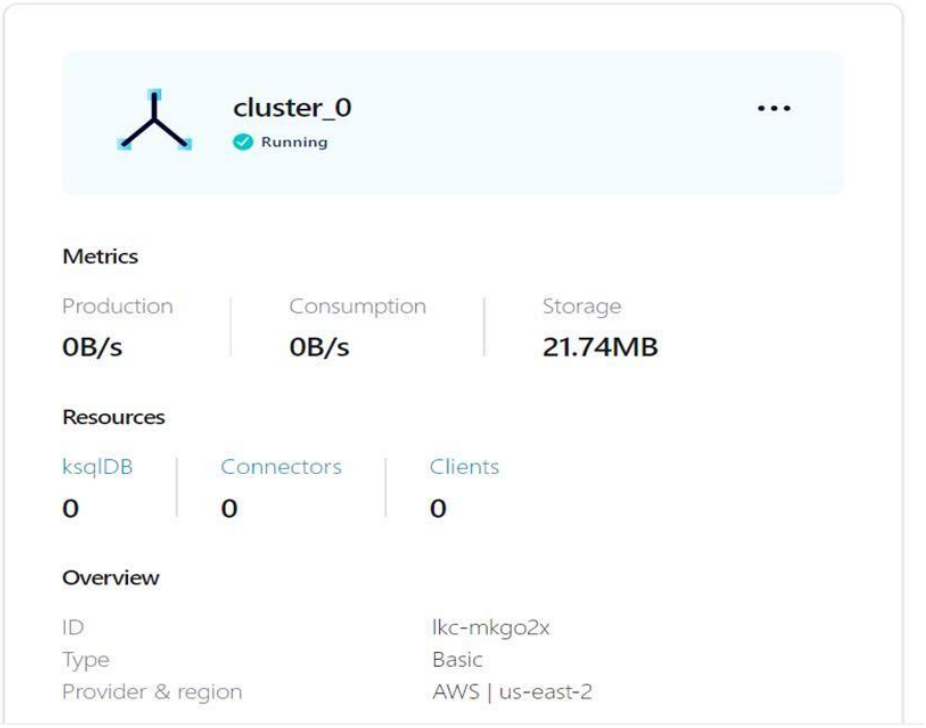
Le platform présente un nombre d'avantage qui englobe une bonne gestion des Kafka clusters ainsi que leur zookeepers, il gère aussi les partitions des topics, facile a utilisé, très sécurisé et il présente une bonne visualisation dès l'entrée de la data jusqu'à l'étape du traitement.

Pour bien comprendre le fonctionnement de notre cluster employé dans le platform Kafka on décrira la démarche étape par étape :

- **Création de l'environnement :** L'environnement créer et qu'on a nommé projet englobe les serveurs de notre espace Kafka ainsi que les partitions ainsi que chaque volume de data qu'on a envoyé.



- **Kafka cluster :** L'interface du cluster présente des informations concernant l'état des serveurs, cela englobe la taille des données envoyées et consommées, le stockage du data, ainsi que des informations spécifiques du cluster créer (ID, Type, Provider…).



- **Topics et partitions :**

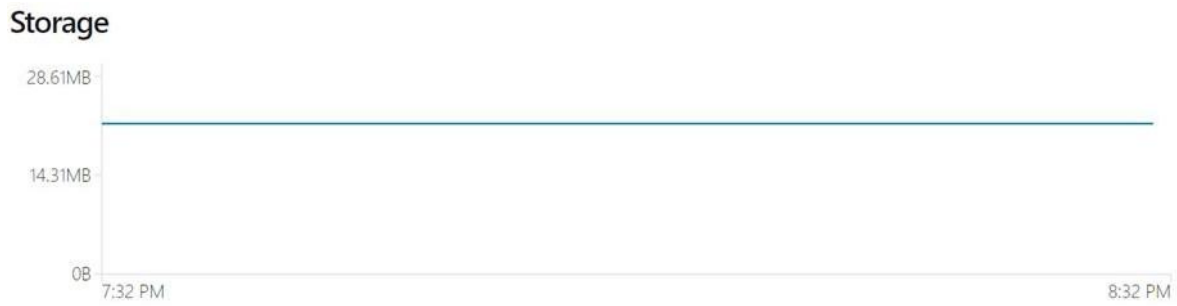
Un topic (ou sujet) est une catégorie ou un canal de données dans Kafka. Il représente un flux de messages ou d'événements spécifiques. Par exemple, vous pourriez avoir un topic pour les données de suivi des utilisateurs, un autre pour les commandes passées, etc.

Une partition est une unité de base de parallélisme et de distribution dans Kafka. Chaque topic est divisé en un certain nombre de partitions, qui sont les unités élémentaires de stockage et de traitement des messages.

Dans notre cas on a utilisé un seul topic avec trois partitions vu ce qui est disponible dans la catégorie sans charge ainsi que la taille de notre data qui n'est pas très massive.

- **Tableau de bord :**

- **Stockage :** Ce graphe présente tous simplement la quantité du data stocker dans nos partitions au cours du temps.



- **Producer :** Dans cette partie le graphe donne une visualisation de la data envoyer à partir du code Producer a notre serveur dans un moment temporelle.

- **Consumer :** Le graphe présente donne en termes de volume en fonction du temps les données consommer à partir du serveur.

6. 2. Code Producer et Consumer

➢ ***Code Producer:***

Tous d'abord, nous avons commencé par définir la fonction *cleaned_data* pour le nettoyage des données suivant les exigences du model machine Learning qu'on a choisi.

```python
def cleaned_data(url):
    r = requests.get(url)
    data = r.json()

    df = pd.DataFrame.from_dict(data['Time Series (Daily)']).T

    df = df.reindex(index=df.index[::-1])
    df.columns = [col.split('. ')[1] for col in df.columns]
    df.reset_index(inplace=True)
    df.rename(columns={'index':'date'}, inplace=True)

    df['date'] = pd.to_datetime(df['date'])
    df['open'] = pd.to_numeric(df['open'])
    df['high'] = pd.to_numeric(df['high'])
    df['low'] = pd.to_numeric(df['low'])
    df['close'] = pd.to_numeric(df['close'])
    df['volume'] = pd.to_numeric(df['volume'])

    df = df.drop(["open","high", "low","volume",'date'], axis = 1)
    df = df.reset_index(drop=True)

    return df
```

Nous avons crée aussi une fonction *delivery_feedback* qui renvoie le statut du message envoyer:

```python
def delivery_report(err, msg):
    if err is not None:
        print('Message delivery failed: {}'.format(err))
    else:
        print('Message delivered to {} [{}]'.format(msg.topic(), msg.partition()))
```

Dans cette partie, nous avons importé nos données en temps réel du site ALPHA Vantage en utilisant l'API key donné et les traiter par la suite suivant les fonctions déjà créer. Nous envoyons à la fin un message d'après notre Kafka Producer instance:

```
# Create a Kafka producer instance
producer = Producer(producer_conf)

# Produce a message to the topic
df = cleaned_data(url)
df_json = df.to_json(date_format='iso', orient='split')

producer.produce(topic, df_json.encode('utf-8'), callback=delivery_report)

# Wait for any outstanding messages to be delivered and delivery reports to be received
producer.flush()

print("Message produced successfully.")
```

```
Message delivered to 3D_topic [0]
Message produced successfully.
```

- ***Code Consumer:***

Nous avons crée une fonction *prediction_results* qui entraîne le modèle choisi **ARIMA** pour prédire le prix de clôture.

```
def prediction_results(data):
    train_data, test_data = data[:int(len(data)*0.9)], data[int(len(data)*0.9):]

    training_data = train_data['close'].values
    test_data = test_data['close'].values

    history = [x for x in training_data]
    model_predictions = []
    N_test_observations = len(test_data)

    model = ARIMA(history, order=(1, 1, 0))
    model_fit = model.fit()

    for time_point in range(N_test_observations):
        true_test_value = test_data[time_point]

        history.append(true_test_value)

        model = ARIMA(history, order=(1, 1, 0))
        model_fit = model.fit()

        yhat = model_fit.forecast(steps=1)[0]

        model_predictions.append(yhat)

    next_day_prediction = model_fit.forecast(steps=1)[0]

    return next_day_prediction
```

En se connectant a notre Kafka cluster pour utiliser les données envoyées a partir de notre instance Producer

```
# Create a Kafka consumer instance
consumer = Consumer(consumer_conf)
# Subscribe to the topic
consumer.subscribe([topic])
```

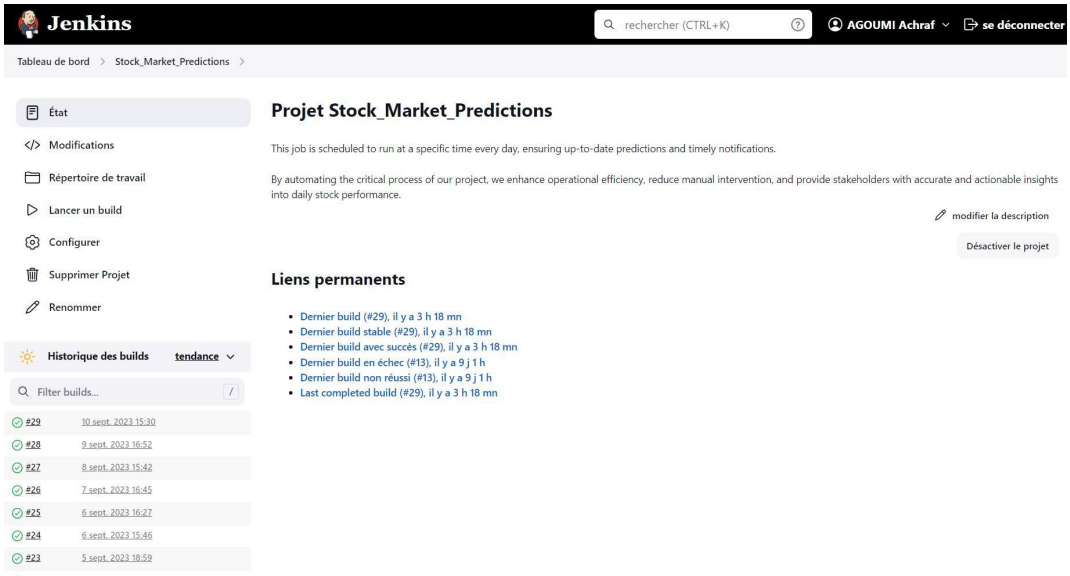
Puis nous faisons appel a notre fonction de prediction pour générer les predictions:

```
while True:
    message = consumer.poll(1.0)
    if message is None:
        continue

    if message.error():
        print("Error:", message.error())
        continue

    try:
        value = message.value().decode('utf-8')
        print("Received message")

        dfs = pd.read_json(value, orient='split')
        print("\nDataFrame created:\n", dfs)

        predictions = prediction_results(dfs)
        print("\nPredicted Price for the Next Day:", predictions)
        break

    except Exception as e:
        print("Error:", e)

consumer.close()
```

```
Received message

DataFrame created:
        close
0      96.75
1      94.81
2      94.37
3      91.56
4      90.25
...      ...
5996  146.83
5997  147.94
5998  148.13
5999  148.06
6000  147.53

[6001 rows x 1 columns]

Predicted Price for the Next Day: 147.54249442494148
```

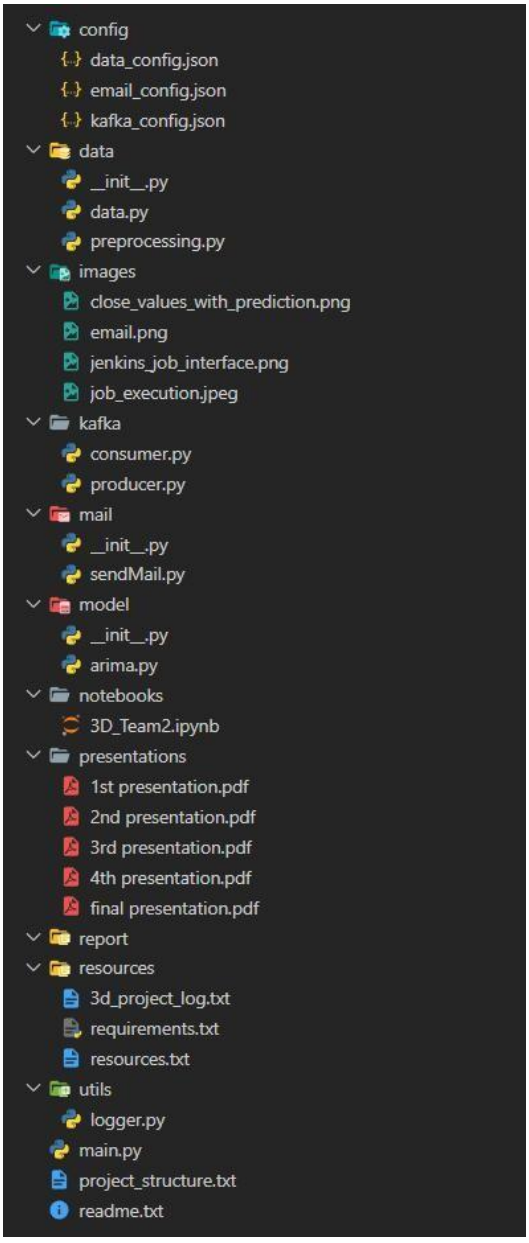
7. Automatisation

Dans notre projet, nous avons mis en œuvre un cadre d'automatisation étendu et finement réglé qui sert d'épine dorsale à nos processus de développement et de déploiement. L'automatisation, dans notre contexte, n'est pas seulement une commodité mais un impératif stratégique. Pour y parvenir, nous avons choisi **Jenkins** comme plate-forme d'automatisation, principalement en raison de ses puissantes capacités d'orchestration de flux de travail complexes et de son solide support pour les pratiques d'intégration et de livraison continues (CI/CD). Notre configuration Jenkins est conçue autour d'un ***travail libre/freestyle job***, méticuleusement configuré pour répondre aux besoins uniques de notre projet. Ce travail est programmé pour exécuter notre script `main.py` à une heure précise chaque jour, garantissant que nos processus sont exécutés. de manière cohérente et efficace.



L'une de nos stratégies clés pour maintenir une base de code rationalisée et maintenable consiste à diviser l'ensemble du projet en packages et scripts. Cette décision offre de nombreux avantages, tels qu'une organisation améliorée du code, une lisibilité améliorée et une maintenance simplifiée. Chaque package encapsule des fonctionnalités spécifiques, garantissant que notre projet reste modulaire et extensible. Cela favorise également la collaboration en équipe, car nous pouvons travailler sur des packages individuels de manière indépendante, réduisant ainsi les conflits et garantissant des cycles de développement plus fluides.

Voici la structure du projet pour référence :



De plus, nous avons intégré une fonction **sendMail** dans notre workflow d'automatisation. Cette fonction automatise le processus d'envoi d'une notification par e-mail à l'utilisateur final, lui fournissant des informations et des prédictions précieuses. Une fois l'exécution du code terminée, un e-mail est automatiquement envoyé, offrant à l'utilisateur la valeur prédite et toute autre information pertinente. Cela permet non seulement de tenir nos parties prenantes bien informées, mais d'améliorer également l'expérience utilisateur en fournissant des mises à jour en temps réel sur l'avancement et les résultats du projet.

Notre engagement en faveur de l'automatisation est motivé par le désir d'atteindre l'excellence opérationnelle. En exploitant la puissance de Jenkins et en divisant méticuleusement notre code en packages, nous avons optimisé notre pipeline de développement, réduit les interventions manuelles et minimisé le risque d'erreur humaine. Cette approche nous permet de concentrer nos efforts sur l'innovation et la résolution créative de problèmes, ce qui se traduit par un processus de développement de projet plus agile, efficace et fiable.

8. Conclusion

Ce chapitre a exploré six facettes essentielles de notre travail : les outils, la collecte et la préparation des données, le choix du modèle, l'implémentation du modèle, le streaming en temps réel, et l'automatisation. Chacun de ces aspects a été examiné en détail

Conclusion générale

Notre voyage à travers ce projet a été une véritable aventure, marquant le début d'une nouvelle ère dans la gestion et la prédiction des données financières en temps réel. Nous avons tracé notre chemin depuis la création de l'infrastructure robuste jusqu'à la prédiction finale des clôtures boursières avec le modèle ARIMA. Ce projet a été une occasion exceptionnelle de mettre en pratique nos compétences en matière de gestion de données, d'automatisation et de modélisation prédictive.

Pour l'avenir, nous envisageons de développer davantage notre modèle de prédiction en utilisant des techniques avancées d'apprentissage automatique, ainsi que de continuer à améliorer notre infrastructure de gestion des données. Nous espérons avoir jeté les bases d'une gestion plus intelligente des données financières en temps réel, offrant ainsi des opportunités significatives pour les investisseurs et les professionnels de la finance.

Ce projet a été une expérience exceptionnelle qui a renforcé nos compétences et notre compréhension du domaine financier en constante évolution. Nous sommes impatients de relever de nouveaux défis passionnants à l'avenir.

Références

- Scrum, la méthode agile: https://www.piloter.org/projet/methode/scrum.htm#def
- Machine Learning : Définition, fonctionnement, utilisations: https://datascientest.com/machine-learning-tout-savoir
- XGBoost : Tout savoir sur le Boosting: https://blent.ai/blog/a/xgboost-tout-comprendre
- Arima, modèle de prévision des séries temporelles: https://www.jedha.co/formation-ia/algorithme-arima
- Random Forest - Site officiel: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
- Apache Kafka, qu'est-ce que c'est ? https://www.redhat.com/fr/topics/integration/what-is-apache-kafka
- DevOps : Construire un pipeline CI/CD avec Jenkins: https://www.data-transitionnumerique.com/devops-jenkins/