Penda DIEYE
Laëtitia MULENDA
Mehdi BAKLOUTI

Projet 5DOOP

Contexte :

Nous sommes recruté afin de mettre en place une plateforme d'intelligence économique et de suivre les flux d'étudiants, de personnels administratifs et d'encadrement.
Il s'agit de proposer une plateforme "Big Data Engineering", Data Scientist & Big Data Engineer pour :
Participer à la conception, l'implémentation et le déploiement d'une architecture pour :

- collecter
- nettoyage
- stocker
- analyser les données des étudiants :
    - leurs régions d'origine
    - leurs institutions d'origine
    - leur mobilité
- suivre leur cursus :
    - présence
    - stages et entreprises
    - Organisation du Contrat Professionnel

I. Spécifications générales et fonctionnelles

SUPINFO, fondée en 1965, fut parmi les premières institutions à former des professionnels aux univers alors naissants de l'informatique puis à ses différents développements. C'est en France l'un des établissements d'enseignement supérieur qui forme en informatique le plus grand nombre d'étudiants.

C'est près de 130.000 inscrits, 12 campus en France et à l'international et plus de 15000 anciens.

Et face à ce très grand volume de données, notre objectif sera de transformer cette data en valeur ajoutée afin de faciliter le suivi du personnel administratif et des flux d'étudiants suivant les différents campus.

Dans ce contexte, notre plateforme doit alors permettre de répondre aux questions suivantes :

- Qui sont les étudiants les plus performants, selon la région / l'institution d'origine, etc.
- Qui sont les étudiants qui arrêtent leurs études et pourquoi ?
- Pourquoi il y a plus d'étudiants dans une région et peu dans une autre ?
- Comment revitaliser les campus ?
- Quel est l'impact d'un salon étudiant sur les recrutements ?
- Quelle est la durée moyenne de l'embauche des diplômés ?
- Quelles entreprises recrutent le plus d'étudiants sur Supinfo ?
- Qui sont les concurrents de SUPINFO ?
- Quelles régions ont plus de contrats Pro et pourquoi ?
- Quelles sont les prévisions de croissance de l'école ?
- Comment attirer plus d'étudiants ?

Cette liste reste non exhaustive car de nouvelles fonctionnalités pourront être développées et mises en place afin d'optimiser davantage les processus.

II. Architecture de la plateforme

1. Choix de notre base NoSQL

Les données à représenter étant hétérogènes, nous proposons d'implémenter MongoDB comme base de données NoSQL.

L'un des avantages de MongoDB est qu'il distribue les données sur plusieurs machines : c'est la méthode du sharding pour la répartition sur un certain nombre de nœuds. Le partitionnement est ainsi utilisé pour prendre en charge les déploiements avec de très grands ensembles de données et des opérations à haut débit.

Nous avons aussi de la haute disponibilité avec le mécanisme de réplication ou Replicat Set. Il s'agit d'une réplication maître esclave entre un nœud principal, et en général, deux nœuds secondaires. Celui-ci permet de mettre en place un mécanisme de fail over automatique et de load balancing (sur l'écriture) et donc réduit le temps de bascule et d'indisponibilité et améliore les performances.

Nous pouvons grâce à MongoDB stocker des documents JSON, donc, des documents bien structurés, et puis les manipuler très souplement avec un langage très riche. Cela représente une bonne souplesse et une bonne richesse fonctionnelle pour travailler avec notre base de données.

Enfin, l'utilisation de MongoDB peut nous garantir une sécurité et une performance durable sur nos données.

2. Types de traitements à effectuer

Il nous a été demandé dans le sujet de répondre à la question suivante :

« Expliquer comment son analyse peut aider à expliquer pourquoi certaines régions ont moins d'élèves et comment y remédier ? »

Pour ce projet, nous allons effectuer un traitement de données à temps réel qui permettra d'analyser au temps les flux d'étudiants à travers les campus.

3. Automatisation des workflows de traitement SPARK

Parlons à présent du traitement de données : notre choix d'infrastructure s'est porté sur Apache SPARK.

Spark est un moteur d'analyse unifié pour le traitement de données à grande échelle, rapide, dédié au Big Data par le biais de machines en clusters.

Pourquoi Spark ?

- Permet d'effectuer différentes tâches avec les mêmes données grâce au partage de données in-memory à travers les DAGs
- Simplicité d'usage
- Sa vitesse de traitement, vu le volume conséquent de nos données (permet de lancer des programmes 100 fois plus rapidement que Hadoop MapReduce in-memory, et 10 fois plus vite sur le disque)
- Fournit un ensemble d'outils de haut niveau : Spark SQL pour le traitement de données structurées, Mlib pour l'apprentissage machine, GraphX pour le traitement des graphes et Structured Streaming pour le calcul incrémental et le traitement des flux
- Fournit différentes APIs et bibliothèques (Python, Java Etc...)

4. Sécurité des données

Nous utiliserons les pratiques recommandées de l'industrie pour protéger et sécuriser les données que nous aurons à traiter. Nous respecterons les procédures et cadres réglementaires en application du Règlement Général sur la Protection des Données (RGPD).

5. Choix de nos outils de développement

Nous allons travailler sous Spark avec Python, en utilisant son API PySpark. Python dispose d'une multiplicité de librairies pour nos différents besoins. Étant donné que le Big Data nécessite beaucoup d'analyse de données et de calculs scientifiques, Python et le Big Data constituent une combinaison parfaite. Les bibliothèques Python sont composées de paquets tels que le calcul numérique, l'analyse de données, l'analyse statistique, la visualisation des données ou bien l'apprentissage automatique.

Par exemple Matplotlib, Skitlearn, Pandas et le module Streamlit servent à mettre en œuvre diverses opérations de Big Data au quotidien ainsi que la mise en place du chargement des données.

6. Architecture proposée

L'architecture que nous proposons est composée de 5 couches qui sont :

❖ *La couche de collecte*

C'est la couche où les données sont collectées, directement ou via des fournisseurs de données, en temps réel ou en mode batch. Les données peuvent provenir d'une source principale ou d'une source secondaire. Les défis de cette couche comprennent la diversité, la confidentialité et la tolérance aux pannes.

En pratique, les données collectées se présentent sous différents formats (texte, image, document etc.). Cette couche est requise pour extraire les informations issues de données non structurées et intégrer les sources de données diversifiées.

La confidentialité des données est nécessaire pour protéger les informations des utilisateurs, en particulier lorsqu'il s'agit de données sensibles telles que la localisation des étudiants, les transactions financières concernant les paiements de mensualité et autres.

La tolérance aux pannes est requise dans la couche de collecte pour minimiser les cas de valeurs manquantes ou d'erreurs causées par le système, la mise en réseau, etc.

❖ *La couche de stockage*

Cette couche est chargée d'acquérir les données à partir des sources de données et si nécessaire, de les convertir dans un format qui convient à la manière dont les données doivent être analysées. Par exemple, une image peut devoir être convertie pour pouvoir être stockée dans un magasin HDFS (Hadoop Distributed File System), un entrepôt Relational Database Management System (RDBMS) ou dans notre base de données MongoDB pour un traitement ultérieur. Les réglementations de conformité et les politiques de gouvernance dictent le stockage approprié pour différents types de données.

Les défis dans la couche de stockage incluent la localité, l'évolutivité, la migration des données, l'intelligence et l'indépendance. Des algorithmes sont nécessaires pour améliorer le calcul élevé et être sensible à la corrélation pour les calculs. Des mécanismes de correction des erreurs sont nécessaires pour la récupération et la gestion de sources hétérogènes de données.

La migration des données est requise lorsque les requêtes arrivent, et les données en continu, au lieu d'être supprimées, doivent être sélectionnées, groupées et ajoutées au système de stockage existant.

La couche de stockage doit être relativement indépendante et capable d'intégrer une infrastructure au niveau de l'entreprise. L'intelligence est requise pour l'optimisation des requêtes avec des Systèmes de stockage distribués.

- *La couche de traitement*

La couche de traitement des données effectue en parallèle le nettoyage, l'intégration, la fusion, l'indexation et la virtualisation des données.

La couche de traitement est confrontée à des défis tels que l'évolutivité, des exigences de faible latence et la flexibilité.

- *La couche d'analyse*

Cette couche lit les données digérées par la couche de massage et de stockage des données. Dans certains cas, la couche d'analyse accède aux données directement à partir de la source de données.

La conception de la couche d'analyse nécessite une réflexion et une planification minutieuses. Des décisions doivent être prises quant à la manière de gérer les tâches pour : Produire les analyses souhaitées ; Dériver des informations à partir des données ; Trouver les entités requises ; Localiser les sources de données pouvant fournir des données pour ces entités et Comprendre quels algorithmes et outils sont nécessaires pour effectuer les analyses.

- *La couche d'application*

Elle agit comme la couche la plus élevée. Les résultats de ces applications aident les utilisateurs à prendre des décisions rapides.

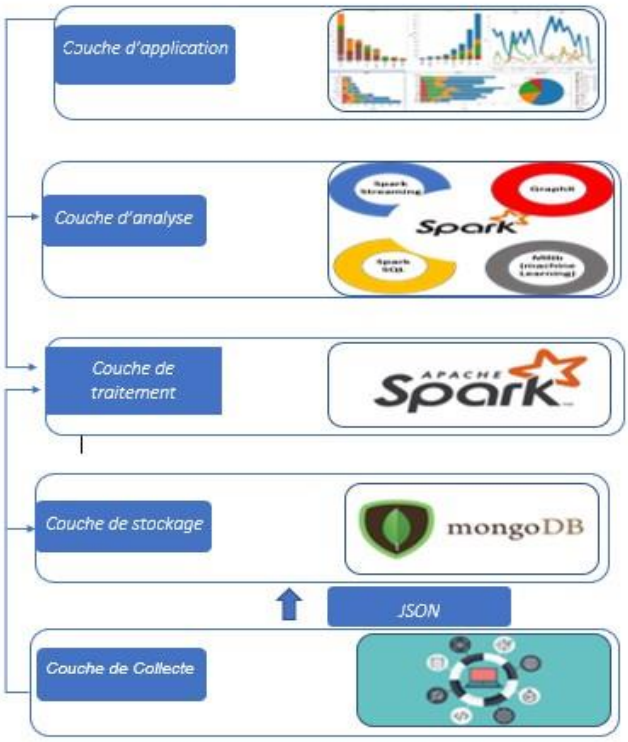
Les défis pour la couche application résident dans la sélection globale du modèle et de la technologie, et incluent les problèmes d'adaptabilité, de valeur et de généralisation.

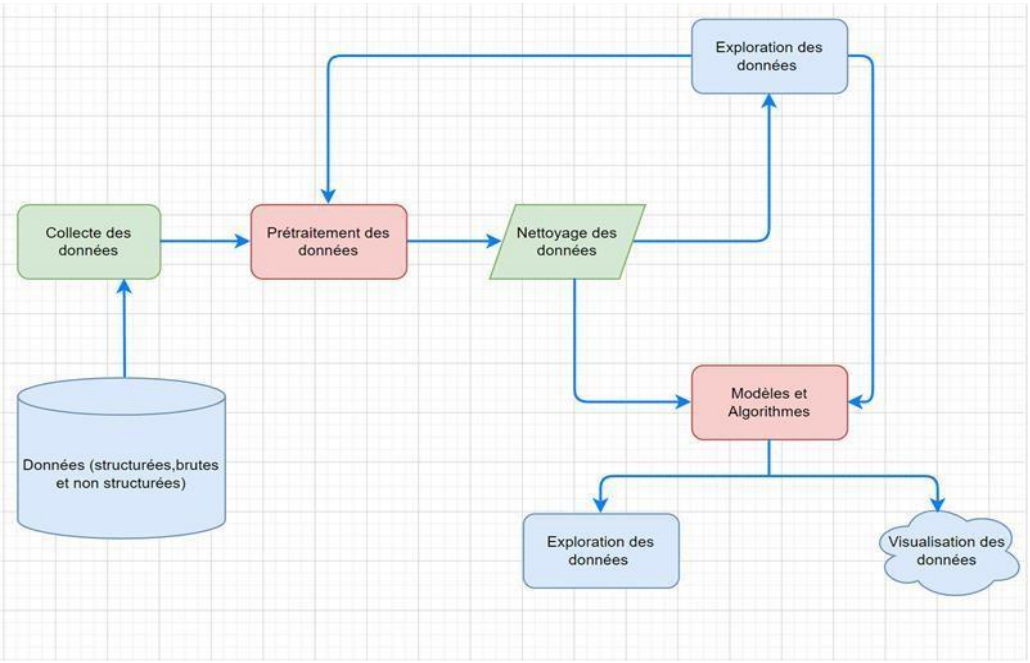
La couche application est censée décider de l'acceptation ou du rejet des différents modèles et algorithmes d'analyse, ainsi que des différentes stratégies d'allocation des ressources pour la couche de stockage et la couche de traitement.

Cette couche doit être capable d'effectuer de manière adaptative l'ajustement mentionné ci-dessus.

Une extraction de « valeur » basée sur la logique métier est attendu. L'ensemble de la solution doit être universellement applicable à de multiples domaines et industries.

Ci-après quelques schémas explicatifs :





7. <u>Implémentation de l'architecture :</u>

Voir le document PDF intitulé « Supinfo Big Data platform » pour regarder les réponses aux différentes questions demandées et avoir un aperçu des graphes et de la solution que notre équipe a implémenté.

Vous pourrez également prendre connaissance de l'annexe contenant le code.