Projet traitement de données
Bilan personnel de Khalid Jerrari

Introduction

Le projet de traitement de données que nous avons mené dans le cadre de notre formation à l'Ensai a été une expérience très formatrice, tant sur le plan technique que collaboratif. Travailler à quatre sur une problématique commune m'a permis d'expérimenter les différentes facettes du travail en équipe : répartition des tâches, coordination, fusion des contributions, mais aussi gestion des désaccords et des lacunes d'organisation. Ce projet m'a également confronté à des défis personnels : approfondissement des connaissances, et réflexion sur les meilleures pratiques en développement informatique.

I. Points positifs

Implication personnelle

Face à la structure de l'évaluation, reposant sur une série de questions indépendantes, j'ai fait le choix de ne répondre qu'aux deux premières, celles qui étaient imposées. Il ne m'a pas semblé pertinent de traiter une troisième question : j'ai préféré me concentrer sur ces deux-là, en les explorant à la fois avec Python natif et la bibliothèque pandas. Bien que j'aurais pu répondre à plusieurs autres questions en mobilisant les fonctions que nous avions développées, je n'en percevais que peu d'intérêt réel.

Ce choix m'a permis de me concentrer pleinement sur l'autre volet du projet, que je considérais comme bien plus essentiel. Je me suis ainsi pleinement investi dans l'analyse de la problématique générale définie en équipe. Cette démarche m'a donné l'opportunité d'explorer des concepts de base mais fondamentaux liés à l'apprentissage non supervisé, tout en consolidant mes compétences en matière de techniques de classification.

Apprentissage par la pratique : recodage de K-means et du critère du coude

Dans une optique d'apprentissage approfondi, j'ai choisi de réimplémenter moi-même l'algorithme de K-means ainsi que le critère du coude, comme le suggéraient les consignes, plutôt que d'utiliser directement les versions proposées par des bibliothèques comme scikit-learn. Cette démarche m'a permis de mieux comprendre les mécanismes internes de l'algorithme, d'en identifier les limites et de consolider mes connaissances en vue de l'examen de classification.

Pour valider mon implémentation, j'ai comparé les résultats obtenus à ceux de l'algorithme de scikit-learn sur un échantillon de plus de 100 individus, et les deux approches ont montré une parfaite concordance. En revanche, pour des échantillons de petite taille, des divergences notables apparaissent. Ces écarts s'expliquent principalement par la sensibilité de K-means à l'initialisation aléatoire des barycentre, qui a un impact d'autant plus marqué lorsque le volume de données est faible. Cette observation m'a permis de mieux comprendre l'importance des techniques d'initialisation (comme K-means++) et la manière dont la taille de l'échantillon peut influencer la stabilité des résultats.

Compétences techniques renforcées

Sur le plan technique, ce projet m'a permis de me confronter à des problématiques concrètes et formatrices, parmi lesquelles :

- La gestion du multithreading, nécessaire pour faire fonctionner en parallèle une interface Tkinter et l'affichage en console.
- La résolution de problèmes d'affichage intempestif des graphiques, que j'ai pu corriger en encapsulant les appels aux fonctions graphiques dans le bloc if __name__ == "__main__".
- La comparaison des performances entre du code Python standard et l'utilisation de la bibliothèque pandas, qui s'est révélée non seulement plus rapide dans mes tests, mais aussi plus concise et expressive, avec une syntaxe proche de celle du langage R.
- Les limites du traitement de données en Python natif m'ont clairement montré l'intérêt d'utiliser des bibliothèques spécialisées comme pandas, bien mieux conçues pour manipuler efficacement des ensembles de données.

J'ai également été amené à approfondir l'organisation des modules en Python, ainsi que les différentes méthodes d'importation dans un programme principal. Ce travail m'a confronté à plusieurs difficultés concrètes, notamment :

- Les erreurs d'importation dues à une mauvaise structuration des dossiers ou à l'absence de fichier __init__.py dans les packages, empêchant Python de reconnaître certains dossiers comme des modules valides.
- Les chemins relatifs et absolus, qui peuvent rapidement devenir source de confusion, surtout lorsque le projet comporte plusieurs niveaux de sous-dossiers. Cela peut entraîner des erreurs comme ModuleNotFoundError, même si le fichier existe.
- Les effets de bord liés aux imports croisés (imports circulaires), où deux modules s'appellent mutuellement, ce qui peut provoquer des comportements inattendus ou des erreurs d'exécution.
- Le manque de clarté dans la structure du projet, qui rend difficile la réutilisation ou la compréhension du code, surtout lorsqu'il s'agrandit. Une organisation modulaire bien pensée devient alors essentielle pour la maintenabilité.

J'ai pris conscience que nos connaissances de base en Python sont suffisamment solides pour nous permettre de nous débrouiller efficacement, que ce soit pour concevoir des algorithmes, comprendre les APIs des bibliothèques ou structurer notre code de manière appropriée.

II. Difficultés rencontrées

Cloisonnement dû à un projet individualisé

Le principal obstacle à une collaboration efficace a été, paradoxalement, la nature individualisée du projet de traitement de données. Chaque membre du groupe s'est retrouvé contraint de se concentrer uniquement sur ses propres questions, ce qui a conduit à un travail en silo, avec peu de partage de code ou d'idées. Certaines fonctions redondantes auraient pu être développées de manière collaborative et utilisées dans plusieurs modules, mais cette approche n'a pas été adoptée. Il serait peut-être plus pertinent d'imposer un nombre de fonctions réutilisables à coder, plutôt que de fixer un nombre de questions à traiter. Cela encouragerait un travail plus collaboratif et permettrait à chacun de progresser dans un cadre collectif plus enrichissant.

Problèmes d'intégration du travail

Le regroupement final du travail a été particulièrement laborieux. En raison de l'absence d'une convention de nommage, d'une organisation claire des répertoires et d'une structure de modules partagée définie en amont, nous avons dû faire face à une grande hétérogénéité dans les fichiers, les chemins et les appels de fonctions. Cette absence d'architecture cohérente a rendu l'intégration des différentes contributions complexe et chronophage.

Pour éviter de telles situations à l'avenir, il serait pertinent d'imposer, dès le début du projet, une architecture modulaire claire, avec des conventions de nommage, une organisation des

répertoires définie et une documentation adéquate. Cela faciliterait non seulement l'intégration des modules, mais aussi la maintenance et l'évolution du projet informatique.

Absence de leadership désigné

L'absence d'un chef de projet clairement désigné a engendré une coordination insuffisante, entraînant une perte de temps dans la gestion des priorités, la prise de décision et le suivi des contributions. Cette carence a également impacté l'architecture du code, avec des divergences dans les conventions de nommage, l'organisation des répertoires et la structure des modules. Lors de la phase d'intégration, ces incohérences ont compliqué l'assemblage des différentes parties du projet, nécessitant des efforts considérables pour harmoniser l'ensemble. Un rôle de coordination clair aurait facilité la gestion du projet, notamment dans la définition d'une architecture cohérente et dans l'intégration fluide des modules.

Sous-exploitation de GitHub

Bien que GitHub ait été utilisé, son utilisation s'est limitée à un simple espace de stockage partagé. Un autre outil de gestion de projet aurait été essentiel pour estimer les jours-hommes nécessaires, identifier les points critiques, suivre l'avancement des tâches et assurer une gestion de projet efficace, tel qu'un diagramme de Gantt, par exemple.

Difficulté dans la rédaction du rapport

Enfin, un aspect plus personnel : j'ai éprouvé des difficultés à trouver un juste équilibre entre la dimension technique et la clarté de la présentation dans la rédaction du rapport. Étant dans une unité d'enseignement informatique, mais avec un projet axé sur le traitement de données dans un contexte statistique, je ne savais pas s'il fallait opter pour un format de type "spécification fonctionnelle" ou plutôt une approche plus narrative pour décrire les résultats. L'équilibre entre ces deux aspects s'est avéré délicat : il ne fallait pas tomber dans un projet purement statistique, mais il était essentiel d'utiliser les données de manière adéquate pour mener des analyses statistiques. Cette incertitude m'a ralenti et a souligné l'importance, dans tout projet, de définir clairement les attentes en matière de documentation dès le début.

III. Enseignements tirés

Ce projet m'a permis de tirer plusieurs enseignements précieux pour l'avenir, tant dans un contexte académique que professionnel :

Sur le plan individuel, j'ai confirmé que l'approfondissement d'un concept par le biais de sa propre implémentation est un excellent moyen d'apprentissage. Cela renforce non seulement ma maîtrise technique, mais aussi ma confiance face à des problématiques inédites.

Sur le plan collectif, j'ai réalisé que l'organisation du projet est tout aussi cruciale que son contenu. Structurer le travail, harmoniser les pratiques de développement et établir des conventions dès le départ permet d'éviter bien des pertes de temps ultérieures.

Sur le plan humain, j'ai pris conscience que la réussite d'un projet repose non seulement sur les compétences techniques, mais aussi sur la communication, l'écoute et la capacité à collaborer efficacement. La désignation d'un coordinateur ou chef de projet est un levier indéniable pour améliorer la fluidité du travail en équipe.

Enfin, sur le plan technique, ce projet m'a montré que nous disposons d'une solide maîtrise de Python, et que des outils comme pandas ou matplotlib sont à la fois puissants et parfaitement adaptés à notre raisonnement "data science", similaire à celui en R. Cela m'encourage à continuer à pratiquer ces outils pour en améliorer la maîtrise. J'ai également appris à créer une interface graphiques avec Tkinter.