PSL UNIVERSITÉ PARIS

MIDO
MATHÉMATIQUES ET INFORMATIQUE
DE LA DÉCISION ET DES ORGANISATIONS

Dauphine UNIVERSITÉ PARIS

Pipeline d'analyse quotidienne pour la recommandation d'investissement sur le marché des actions

Shana BENATTAR
Ismael BENDIB
Diego LUQUE ROMANO

Table des matières

0.1 Résumé

Ce projet consiste en la mise en place d'un pipeline automatisé d'analyse de données financières, exécuté quotidiennement, dans le but de générer des recommandations d'investissement sur des actions cotées. Le pipeline intègre plusieurs modules développés durant les travaux pratiques, incluant :

— un clustering non supervisé pour identifier des groupes d'entreprises aux profils similaires,
— un modèle de classification supervisée pour prédire des signaux de type Buy / Sell / Hold,
— une régression (ou un modèle séquentiel) pour estimer les rendements à J+1,
— une analyse de sentiments basée sur des données textuelles issues de news financières, exploitant des techniques de NLP.

Ces signaux sont ensuite agrégés afin de produire une recommandation finale par entreprise. Le tout est orchestré via un fichier `main.py` permettant l'exécution complète et automatisée du pipeline pour chaque entreprise analysée.

0.2 Abstract

This project involves the development of an automated financial data analysis pipeline, executed on a daily basis, with the objective of generating investment recommendations for publicly traded companies. The pipeline integrates several modules developed during the practical sessions, including :

— an unsupervised clustering algorithm to identify groups of companies with similar profiles,
— a supervised classification model to predict Buy / Sell / Hold signals,
— a regression (or sequential) model to estimate returns at $t + 1$,
— a sentiment analysis module based on textual data from financial news, using NLP techniques.

These signals are then aggregated to produce a final recommendation for each company. The entire pipeline is orchestrated through a `main.py` file that enables the full and automated execution of the analysis for each company.

1 Introduction

Le marché financier génère quotidiennement une quantité massive d'informations complexes, structurées (prix, volumes, ratios financiers) ou non structurées (actualités, rapports, opinions). Dans un contexte où la rapidité d'analyse et la pertinence des décisions d'investissement sont devenues cruciales, les techniques issues de la science des données se révèlent être des outils de plus en plus incontournables.

L'objectif de ce projet est de concevoir un pipeline automatisé et modulaire capable de produire quotidiennement des recommandations d'investissement pour un ensemble d'actions cotées, en combinant des méthodes de clustering, de classification supervisée, de régression temporelle et d'analyse de sentiments sur des données textuelles issues de l'actualité financière. L'enjeu est de croiser différents types de signaux pour produire une recommandation robuste, dynamique et explicable.

Une des originalités majeures de notre démarche réside dans le fait que, plutôt que de s'appuyer directement sur des fonctions déjà implémentées dans les bibliothèques classiques de Python (comme `scikit-learn` ou `scipy`), nous avons choisi de développer notre propre bibliothèque maison. Celle-ci regroupe les principales méthodes nécessaires au projet, notamment :

— une implémentation manuelle du clustering hiérarchique ascendant (CHA),
— une version codée depuis zéro de la méthode des K-moyennes,
— des fonctions de prétraitement, de normalisation et de visualisation spécifiques.

Cette approche nous a permis de renforcer notre compréhension fine des algorithmes et d'assurer une maîtrise complète sur leur comportement, leur paramétrage et leur intégration dans le pipeline global. Elle s'inscrit également dans une logique d'extensibilité et de transparence du code, essentielle pour l'interprétation des résultats dans un contexte financier.

Le présent rapport documente la démarche suivie, les méthodes mises en œuvre, les résultats obtenus ainsi que la stratégie d'agrégation des signaux conduisant à la décision d'investissement finale.

1.1 Informations sur le code du projet

Le code du projet peut se retrouver sur Github à l'adresse suivante :

Il comprends les 8 TP principaux, numérotés de 1 à 8, menant au pipeline. De plus, 2 TP supplémentaires correspondant aux améliorations proposées dans les ennoncés ont été rajoutés. Ils sont nommées TP_3B et TP_4B.
Un fichier main permet de lancer tous les TP.
Les résultats d'une grande partie des codes sont également sauvegardées dans les dossiers "Résultats" (comprenant un grand nombre de graphiques), "Companies_historical_data", "Data" (data généraux) et pour finir "output" contenant les recommandations.
En cas de doute ou d'erreur, n'hésitez pas à contacter un des membres du groupe à l'adresse de l'université @dauphine.eu.

2 Scrapping / Données

La première partie de ce projet consiste en un scrapping d'informations d'entreprises permettant l'importation de :

- un fichier (CSV) global appelé "financial_ratios.csv", comprenant des ratios financiers intéressants pour ces entreprises.
- un fichier (CSV) pour chaque entreprise, comprenant des données journalières historiques de rendement de marché.

Ces données permettront d'être une base d'étude par la suite.

Données utilisées

Liste des entreprises : Apple, Microsoft, Amazon, Alphabet, Meta, Tesla, NVIDIA, Samsung, Tencent, Alibaba, IBM, Intel, Oracle, Sony, Adobe, Netflix, AMD, Qualcomm, Cisco, JP Morgan, Goldman Sachs, Visa, Johnson & Johnson, Pfizer, ExxonMobil, ASML, SAP, Siemens, Louis Vuitton (LVMH), TotalEnergies, Shell, Baidu, JD.com, BYD, ICBC, Toyota, SoftBank, Nintendo, Hyundai, Reliance Industries, Tata Consultancy Services.

Ratios extraits : forwardPE, beta, priceToBook, priceToSales, dividendYield, trailingEps, debtToEquity, currentRatio, quickRatio, returnOnEquity, returnOnAssets, operatingMargins, profitMargins.

Colonnes des fichiers historiques : Date, Close, Next Day Close, Rendement.

3 Stratégies d'analyse financière existantes

Plusieurs travaux académiques ont exploré, de manière partielle, les différentes composantes du pipeline que nous cherchons à construire. Notre approche se distingue toutefois par l'intégration complète de ces briques dans un système modulaire, avec l'implémentation de certains de nos propres packages, destiné à produire quotidiennement des recommandations d'investissement. Nous présentons ci-dessous trois contributions évoquant des aspects similaires.

News-Based Trading Strategies (Hagenau et al., 2013) [1] Ce travail propose une stratégie de trading basée sur l'analyse automatique des nouvelles financières. Les auteurs exploitent le traitement du langage naturel (NLP) pour extraire des indicateurs de sentiment à partir de flux d'actualités, et les combinent avec des modèles de classification supervisée afin de générer des signaux d'achat ou de vente. Ce projet démontre l'intérêt d'intégrer les dimensions textuelles dans les stratégies de décision financière.

Financial News Predicts Stock Market Volatility Better than Close Price (Tetlock et al., 2008) [2]

Cette étude explore la relation entre le contenu des articles financiers et la volatilité des marchés boursiers. Les auteurs montrent que certaines métriques de sentiment extraites de la presse économique peuvent prédire plus efficacement la volatilité des marchés que les prix de clôture eux-mêmes. Cette conclusion renforce la pertinence de l'analyse de sentiment dans un contexte prédictif.

Stock Movement Prediction from Tweets and Historical Prices (Xu & Cohen, 2018) [3] Dans ce projet, les auteurs combinent des données de séries temporelles historiques avec des données textuelles issues des réseaux sociaux pour prédire les variations de prix des actions. L'étude met en évidence le gain en performance obtenu par la fusion de données hétérogènes, ce qui valide le principe de notre pipeline fondé sur l'agrégation de signaux multiples.

1. https ://arxiv.org/abs/1807.06824
2. https ://www.researchgate.net/publication/323013379_Financial_News_Predicts_Stock_Market_Volatility_Better_Than_Close_Price
3. Disponible en ligne : https ://www.researchgate.net/publication/334116360_Stock_Movement_Prediction_from_Tweets_and_Historical_Price

4 Packages personnalisés

Lors de ce projet, nous avons choisi d'élaborer certains packages personnalisés, nous permettant d'avoir un meilleur contrôle des actions réellement effectuées par les algorithmes, nommés iads.

Cette bibliothèque Python a été conçue pour implémenter manuellement des méthodes essentielles en science des données et en machine learning. Cette approche nous a permis de mieux comprendre les algorithmes sous-jacents, tels que CHA, K-moyennes, ou les méthodes de classification, en les codant de zéro plutôt qu'en utilisant des bibliothèques prêtes à l'emploi comme scikit-learn.

Notre bibliothèque iads est structurée en plusieurs modules Python, chacun dédié à une fonctionnalité spécifique en science des données. Le fichier **Clustering.py** contient les implémentations manuelles des algorithmes de clustering, incluant le calcul des centroïdes, les mesures de distance (euclidienne, Manhattan, etc.), ainsi que les méthodes K-moyennes et CHA (incluant la visualisation avec des dendrogrammes). Ces fonctions permettent de comprendre précisément comment les clusters se forment et évoluent itérativement, sans dépendre de bibliothèques externes.

Le module **evaluation.py** regroupe des méthodes d'évaluation des modèles, notamment la validation croisée et des métriques comme le score de silhouette ou la pureté des clusters. Ces outils sont essentiels pour mesurer la robustesse et la performance de nos algorithmes sur des données financières volatiles.

Enfin, **utils.py** offre des fonctions utilitaires pour la génération et la visualisation de datasets. On y trouve des méthodes pour créer des distributions artificielles (uniforme, gaussienne) afin de tester nos modèles dans des conditions contrôlées, ainsi que des visualisations 2D pour illustrer les partitions de clusters ou les frontières de décision.

Cette architecture modulaire permet une transparence totale sur les mécanismes des algorithmes, tout en facilitant leur réutilisation pour d'autres projets d'analyse prédictive. L'objectif de ce package personnalisé est de maîtriser les mécanismes des algorithmes, leurs paramètres, et leurs impacts sur les prédictions. Par exemple, l'implémentation manuelle de CHA nous a permis d'analyser finement les critères de division des données, tandis que notre version de K-moyennes a clarifié l'optimisation des centroïdes. Ce travail a aussi renforcé notre compréhension des pré-traitements nécessaires avant l'application des modèles.

5 Approche de clustering des entreprises

5.1 K-Means

Dans un premier temps, nous avons identifié des groupes d'entreprises présentant des caractéristiques financières similaires. Pour déterminer le nombre optimal de clusters, nous utilisons la méthode du coude, qui consiste à analyser la courbe d'inertie en fonction du nombre de clusters. Le "coude" de la courbe indique généralement un bon compromis entre précision et complexité. Dans notre cas, le coude apparaît pour un nombre de clusters égal à 5.

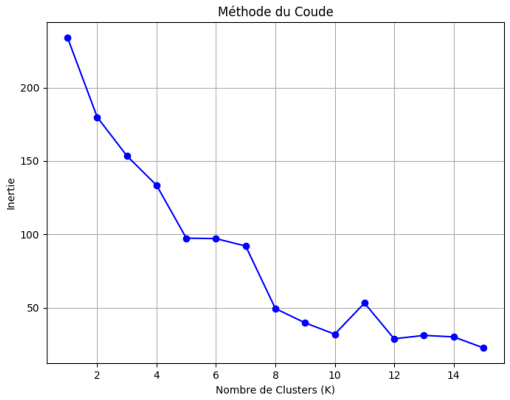

FIGURE 1 – Méthode du coude

Nous appliquons ensuite l'algorithme K-Means avec ce nombre de clusters. Chaque entreprise est ainsi affectée à un groupe, et nous pouvons visualiser les résultats grâce à une réduction de dimension avec la méthode t-SNE.

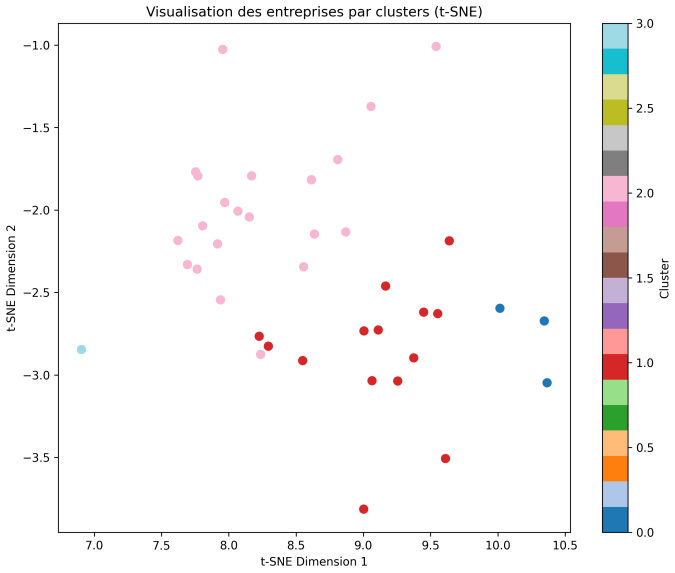

FIGURE 2 – Clusters obtenus avec K-Means

5.2 Clustering des corrélations de rendements journaliers

Nous introduisons à présent le dendrogramme des corrélations entre les rendements journaliers des entreprises. Celui-ci permet de visualiser les proximités entre entreprises en fonction de l'évolution de leur performance boursière. Une forte similarité entre deux séries temporelles implique un rapprochement sur l'arbre hiérarchique, et donc une appartenance probable à un même groupe.

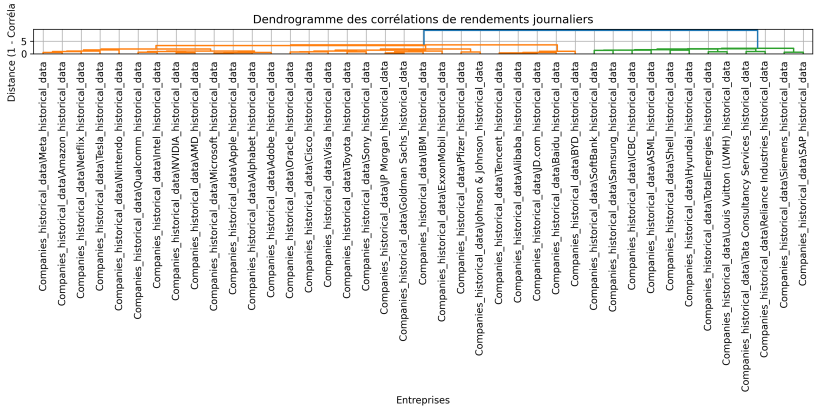

FIGURE 3 – Dendrogramme des corrélations des rendements journaliers

6 Classification Buy / Sell / Hold

6.1 Préparation des données pour la classification

Dans cette partie, nous introduisons une approche de self-supervised learning afin de générer automatiquement les étiquettes à prédire. L'idée est de formuler une consigne d'investissement à partir

du rendement à 20 jours :

— **Buy (2)** : si le rendement est supérieur à +5%,
— **Sell (0)** : si le rendement est inférieur à -5%,
— **Hold (1)** : sinon.

Préparation des données. Pour chaque entreprise, nous utilisons les données de clôture (`Close`) pour créer une colonne `Close Horizon` (valeur dans 20 jours) et en déduire le rendement sur 20 jours (`Horizon Return`), qui sert à générer les labels.

À partir de la colonne `Close`, nous extrayons ensuite des indicateurs techniques à l'aide de la librairie `ta` : `SMA20`, `EMA20`, `RSI14`, `MACD`, bandes de Bollinger, volatilité roulante, etc. Chaque entreprise génère un DataFrame structuré.

Les données de toutes les entreprises sont ensuite concaténées, puis :

— les variables explicatives (`X`) sont séparées des labels (`Y`) ;
— les colonnes inutiles sont supprimées (`Label`, `Close Horizon`, etc.) ;
— `X` est standardisé, puis séparé en jeux d'entraînement et de test.

Taille des données :

— Entraînement : (39 494, 9)
— Test : (9 874, 9)

6.2 Modélisation.

Chaque algorithme est encapsulé dans une fonction dédiée. Nous appliquons les modèles suivants :

— `Random Forest`
— `XGBoost`
— `K-Nearest Neighbors (KNN)`
— `Support Vector Machine (SVM)`
— `Régression logistique`

Pour chacun d'eux :

— une recherche d'hyperparamètres est effectuée via `GridSearchCV`,
— le `classification_report` est utilisé pour évaluer les performances (précision, rappel, F1-score),
— les graphiques SHAP sont générés pour interpréter les décisions du modèle (notamment pour les classes `buy` et `sell`).

6.3 Résultats :

Les résultats détaillés des modèles selon les métriques usuelles sont présentés ci-dessous :

2*Modèle	Classe 0 (Sell)				Classe 1 (Hold)				Classe 2 (Buy)			
	Préc.	**Recall**	**F1**	**Supp.**	**Préc.**	**Recall**	**F1**	**Supp.**	**Préc.**	**Recall**	**F1**	**Supp.**
Random Forest	0.70	0.06	0.12	2164	0.50	0.92	0.65	4505	0.26	0.36	0.36	3205
XGBoost	0.65	0.22	0.33	2164	0.56	0.84	0.67	4505	0.44	0.52	0.52	3205
KNN	0.29	0.15	0.19	2164	0.50	0.66	0.57	4505	0.34	0.37	0.37	3205
SVM	0.00	0.00	0.00	113	0.45	1.00	0.62	226	0.00	0.00	0.00	161
Régression logistique	0.00	0.00	0.00	2164	0.46	1.00	0.63	4505	0.00	0.00	0.00	3205

TABLE 1 – Rapport de classification complet : précision, rappel, F1-score et support par classe

Les précisions (accuracy) des meilleurs modèles sont comparées ci-dessous :

Modèle	**Meilleure accuracy**
Random Forest	0.516
XGBoost	**0.577**
KNN	0.447
SVM	0.456
Régression logistique	0.456

TABLE 2 – Comparaison des modèles de classification

Analyse : XGBoost se distingue comme le meilleur modèle. Cependant, les scores restent globalement modestes, ce qui peut s'expliquer par un fort déséquilibre des classes, notamment une surreprésentation des labels hold. Les `classification_reports` confirment que les modèles ont plus de difficulté à bien prédire les classes minoritaires (buy et sell).

6.4 Améliorations

Nous avons décidé de mettre en place des améliorations proposées aux TP3.

6.4.1 SMOTE

Pour combler les déséquilibres de classes, nous avons intégré la méthode SMOTE afin de :
— équilibrer les classes buy, hold et sell dans y_{train},
— permettre aux modèles de mieux généraliser sur les classes minoritaires,
— améliorer les scores de recall et F1 dans les `classification_report`.

Résultats
— Entraînement : (39494, 9)
— Test : (9874, 9)

Modèle	**Meilleure accuracy**
Random Forest	0.4829
XGBoost	0.5323
KNN	0.4019
SVM	0.4562
Régression logistique	0.3917

Table 3 – Comparaison des performances des modèles après application de SMOTE

6.4.2 Ajout de nouvelles variables macroéconomiques

Nous avons ensuite ajouté les indicateurs économiques suivants : **VIX**, **US10Y** et **SP500**, afin d'enrichir les données des entreprises avec des signaux représentatifs du marché global.

Résultats
— Entraînement : (36476, 14)
— test : (9119, 14)

Modèle	**Meilleure accuracy**
Random Forest	0.6055
XGBoost	0.6837
KNN	0.5123
SVM	0.4536
Régression logistique	0.4552

Table 4 – Performances des modèles avec SMOTE et variables macroéconomiques

L'ajout de ces variables améliore significativement la performance, notamment pour les modèles XGBoost et Random Forest.

7 Approche de prédiction à J+1

Dans cette partie, nous avons cherché à prédire le prix de clôture d'une action à J+1 à partir des 30 jours précédents. Pour cela, nous avons appliqué plusieurs modèles de régression (Régression linéaire, Random Forest, KNN, XGBoost) à chaque entreprise séparément.
Chaque modèle a été entraîné à l'aide d'un GridSearchCV pour optimiser ses hyperparamètres. Les performances ont été évaluées via les erreurs MSE et RMSE.
Les graphique ci-dessous illustre les résultats pour certaines entreprises (au hasard), en comparant

les valeurs réelles aux prédictions des différents modèles. On observe que certains modèles (comme la régression linéaire et XGBoost) parviennent à suivre de très près la tendance réelle du prix, tandis que d'autres comme KNN peuvent présenter plus d'instabilité.

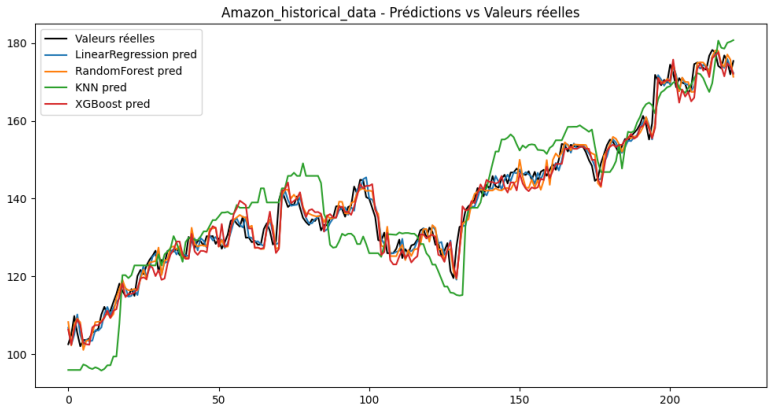

FIGURE 4 – Prédictions et résultats J+1 pour Amazon

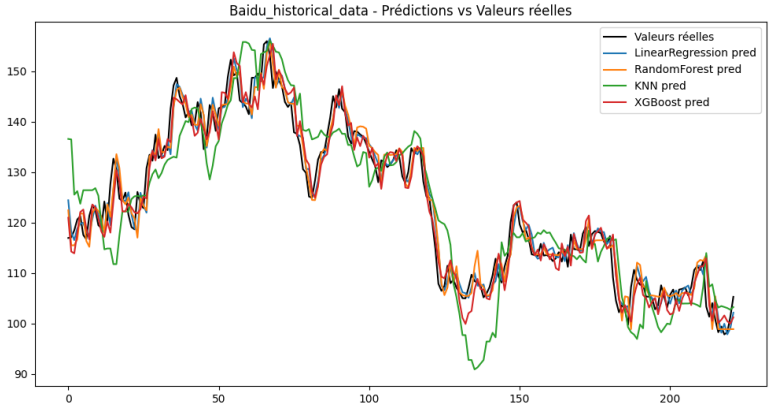

FIGURE 5 – Prédictions et résultats J+1 pour Baidu

7.1 Améliorations - Données à J+2

Nous avons mis en place une stratégie de prediction à J+2 selon notre compréhension du sujet en prenant des données J+1 et sur les 30 jours précédents (TP4_B). A continuation, quelque exemples de résultats :

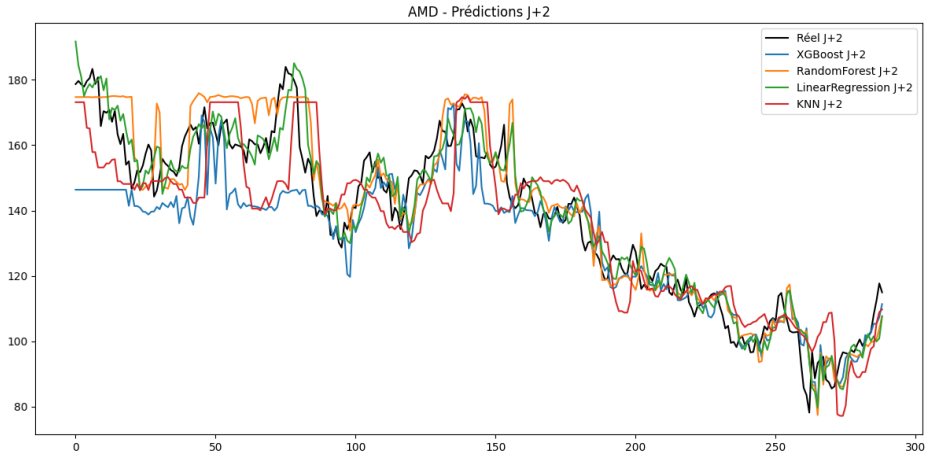

FIGURE 6 – Prédictions et résultats J+2 pour AMD

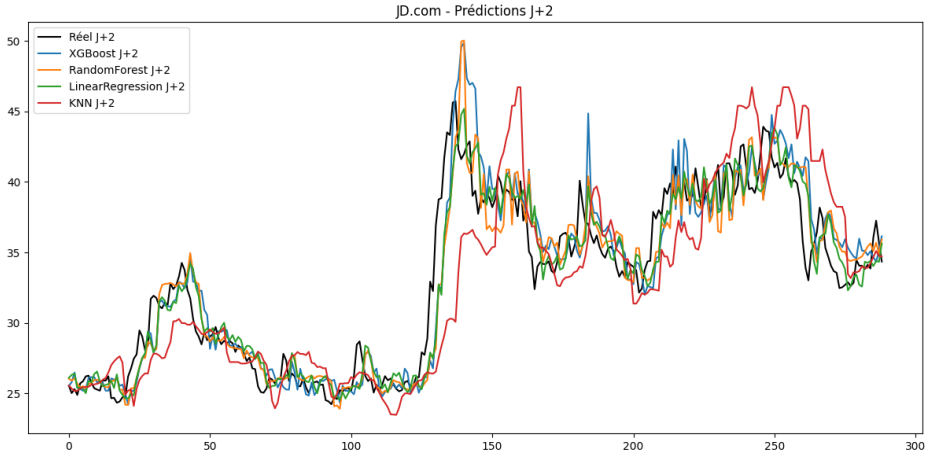

FIGURE 7 – Prédictions et résultats J+2 pour JD

Pour l'étude J+2, les modèles sont moins précis que J+1 par rapport aux valeurs réelles.

8 Réseaux de neuronnes

Dans cette section, nous avons utilisé des réseaux de neurones pour prédire le prix de clôture d'une action à l'horizon J+1. Ce type de modèle est particulièrement adapté à la modélisation de séries temporelles financières, car il est capable de capturer les dépendances temporelles longues.
Les figure suivante montre un exemple de résultats obtenus. On y compare les valeurs réelles (en bleu) aux prédictions du modèle LSTM (en orange).

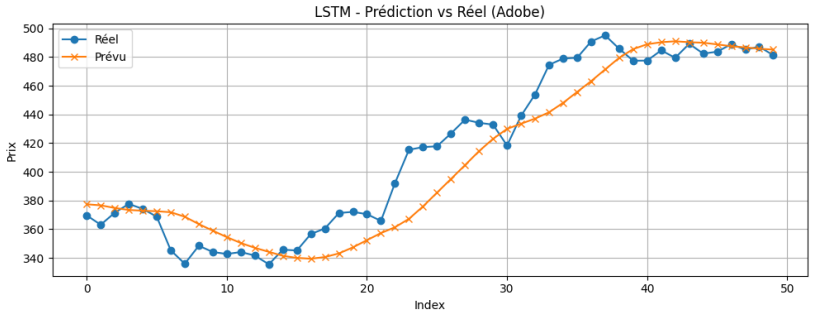

FIGURE 8 – LSTM Adobe

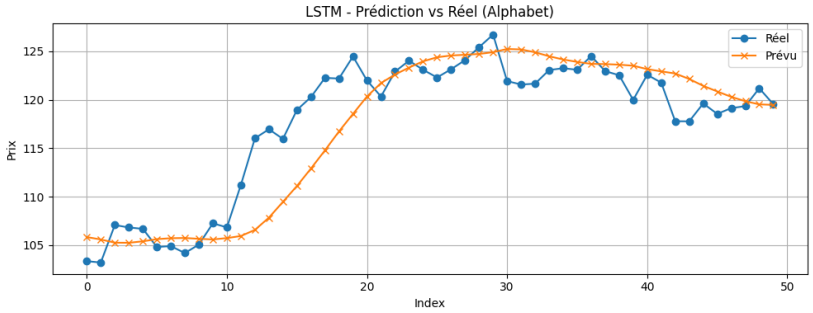

FIGURE 9 – LSTM Alphabet

8.1 Comparaison des modèles

Nous avons appliqué trois types de réseaux de neurones (MLP, RNN, LSTM) pour prédire les prix des actions à J+1. Les performances obtenues en termes d'erreur absolue moyenne (MAE) et d'erreur quadratique moyenne (RMSE), ainsi qu'un aperçu des 10 premières prédictions, sont présentées ci-dessous pour l'exemple de l'entreprise ASML.

Modèle : MLP

— **MAE** = 23.5683
— **RMSE** = 31.3675

Réel	**Prédit**
870.88	880.95
847.85	882.76
860.79	870.39
859.61	866.37
864.45	873.37
912.69	843.32
895.69	865.15
902.51	885.81
890.05	881.67
884.91	863.94

TABLE 5 – Prédictions MLP vs valeurs réelles (ASML)

Modèle : RNN

— **MAE** = 28.1454
— **RMSE** = 35.0986

Réel	Prédit
870.88	838.22
847.85	836.50
860.79	839.60
859.61	829.84
864.45	820.57
912.69	835.08
895.69	834.35
902.51	859.70
890.05	860.73
884.91	868.09

Table 6 – Prédictions RNN vs valeurs réelles (ASML)

Modèle : LSTM

- **MAE** = 28.3033
- **RMSE** = 35.7782

Réel	Prédit
870.88	874.54
847.85	871.33
860.79	865.31
859.61	859.32
864.45	853.35
912.69	848.35
895.69	848.39
902.51	849.66
890.05	852.84
884.91	855.85

Table 7 – Prédictions LSTM vs valeurs réelles (ASML)

Les prédictions sont donc proches des valeurs réelles, LTSM étant plus performant que MLP, qui est plus performant que RNN.

9 Stratégie d'aggrégation pour fournir des recommendations pertinentes

9.1 Collecte quotidienne de données textuelles

Afin de fournir des recommandations d'investissement basées sur l'actualité, nous procédons dans un premier temps à un scraping des nouvelles financières les plus récentes. Un algorithme développé lors du TP6 permet d'extraire ces données depuis diverses sources et de les structurer automatiquement dans un fichier au format JSON.

L'objectif est d'exécuter ce processus quotidiennement, afin de constituer progressivement une base de données textuelles suffisamment riche et représentative pour alimenter les modèles d'analyse de sentiment.

9.2 Analyse de sentimentsFine-tuning de modèles BERT sur des données financières

Dans un second temps, nous procédons à l'adaptation de modèles de type BERT à notre problématique. Ces modèles, initialement pré-entraînés sur de vastes corpus de texte, sont ensuite affinés (fine-tuning) sur un jeu de données spécifique à la classification de nouvelles financières, issu de la plateforme HuggingFace.

L'objectif est d'évaluer dans quelle mesure un entraînement ciblé améliore la compréhension du contenu financier. Pour cela, nous comparons deux variantes :

— **BERT** : modèle de base, entraîné sur un corpus généraliste (Wikipedia, livres, etc.).
— **FinBERT** : modèle dérivé de BERT, pré-entraîné spécifiquement sur des textes issus du domaine financier.

9.3 Intégration des sentiments aux données boursières

L'analyse dans le TP8 repose sur l'intersection de deux séries temporelles : les variations de cours boursiers et les sentiments extraits des actualités financières.

Nous avons codé la fonction d'exécution du TP8 dans le fichier main.py, en intégrant la possibilité d'utiliser une version fine-tunée de FinBERT pour les nouvelles du secteur technologique en tant que modèle NLP. Toutefois, faute de temps, nous n'avons pas pu entraîner ce modèle ni le tester sur nos données. Par conséquent, le programme utilise uniquement la version de base de FinBERT.

Les sentiments sont tri-catégorisés (négatif / neutre / positif) et alignés temporellement avec les horaires de marché, conformément à la méthodologie décrite dans les consignes du TP.

Pour chaque entreprise analysée, les sentiments extraits des articles récents sont comptabilisés et agrégés afin de produire un score de sentiment normalisé (compris entre -1 et 1). Ce score est intégré dans la décision finale via une pondération de 30% dans le score de recommandation pour les entreprises incluses dans l'analyse du TP8.

En visualisant les résultats du TP8 pour Microsoft, on observe que la corrélation entre sentiments négatifs et mouvements baissiers est plus marquée lors des périodes de forte volatilité (janvier-février), mais s'atténue dans les phases de consolidation (mars).

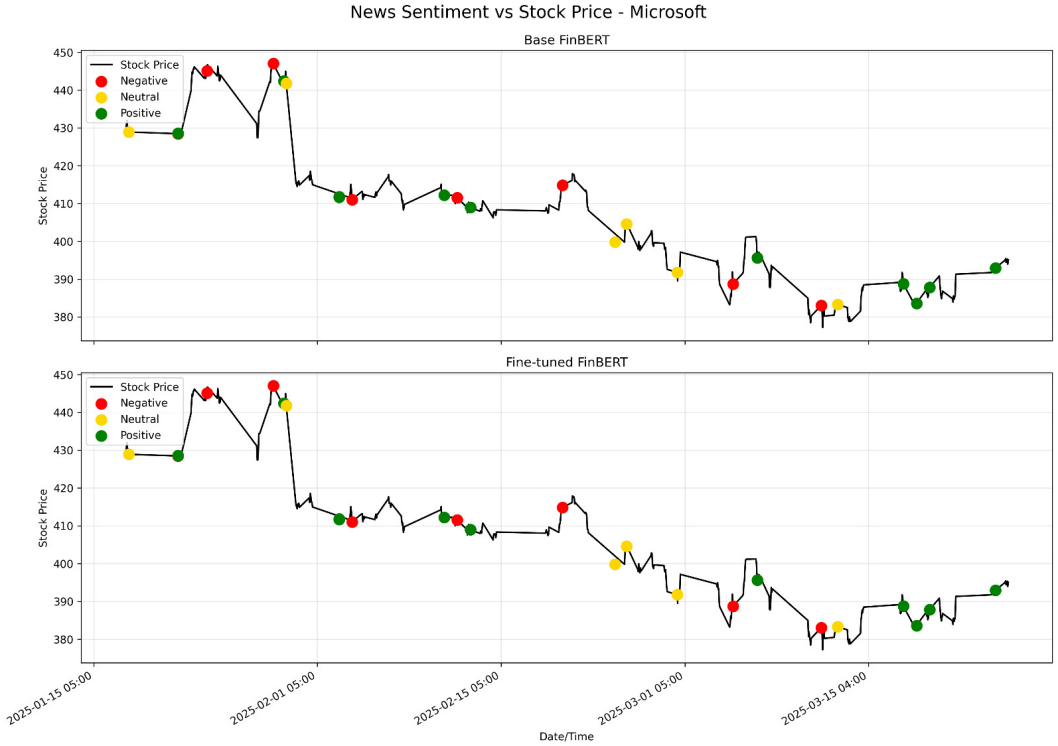

FIGURE 10 – Analyse de sentiments pour le stock Microsoft

Quelques observations notables :

— Les sentiments positifs de mi-janvier (cours autour de 429 dollars) ont précédé une forte hausse.
— Un cluster de sentiments négatifs début février coïncide avec une chute brutale du cours.
— La récupération graduelle après mars montre une indifférence aux signaux médiatiques positifs.

En conclusion, l'analyse de sentiment s'avère efficace pour anticiper certains mouvements du marché en lien avec l'actualité économique. Cependant, la corrélation directe entre le ton des articles et l'évolution du cours reste limitée. C'est pourquoi cette analyse est couplée à d'autres signaux (techniques, fondamentaux, clustering) afin de produire une recommandation plus contextualisée et robuste.

9.4 Conseil d'investissement

Finalement, nous obtenons un fichier json et txt résumant les conseils d'investissement pour chaque entreprise. Il donne pour une entreprise la recommandation, le retour espéré, les entreprises similaires, le sentiment des articles récents ainsi que certains exemples de ces derniers. Un exemple est :

```
Company: Qualcomm
Recommendation: HOLD
expected return: 4.07%
news sentiment: positive
similar companies: Microsoft, Alphabet, Meta, Tencent, Adobe
recent news:
  - Nothing Phone 3 leak points to a top-tier Qualcomm chip, flagship cameras, and massi
  - Qualcomm Snapdragon 7 Gen 4 announced
  - Snapdragon 7 Gen 4 brings big CPU and GPU upgrades
  - Lenovo Thinkbook 16 X1 Plus Laptop Sees Linux Patches But Not Yet Ready For Daily Us
  - Xiaomi's self-developed smartphone chipset has a name and a release date
```

10 Conclusion

Ce projet a permis de concevoir et de mettre en œuvre un pipeline automatisé d'analyse financière, combinant des techniques avancées de science des données pour générer des recommandations d'investissement quotidiennes.

En intégrant des méthodes de clustering non supervisé, de classification supervisée, de prédiction temporelle et d'analyse de sentiments, nous avons développé une approche holistique pour évaluer les opportunités d'investissement sur le marché des actions.

Les résultats obtenus démontrent l'efficacité de certaines méthodes, comme XGBoost pour la classification Buy/Sell/Hold et les réseaux de neurones LSTM pour la prédiction des prix à J+1.

Cependant, les défis liés au déséquilibre des classes et à la volatilité des marchés ont mis en lumière la nécessité d'améliorations continues, telles que l'utilisation de SMOTE ou l'ajout de variables macroéconomiques pour enrichir les données.

L'analyse de sentiments, bien que prometteuse, a révélé ses limites lorsqu'elle est utilisée isolément, soulignant l'importance de croiser plusieurs signaux pour des recommandations robustes.

La stratégie d'agrégation finale, qui combine ces différents signaux, offre une vision plus complète et contextualisée des opportunités d'investissement.

En conclusion, ce pipeline modulaire et extensible constitue une base solide pour des analyses financières automatisées. Les perspectives d'amélioration incluent l'incorporation de données supplémentaires, l'optimisation des modèles existants et l'exploration de nouvelles techniques d'apprentissage automatique.

Ce travail ouvre également la voie à des recherches futures sur l'interprétabilité des modèles et l'adaptation à des contextes économiques en constante évolution.

Références

Cours et TP de Python de M. Pierre FIHEY