Analyse de Données Financières : Scraping, Modèles de Machine Learning et NLP

M2 Data Science - 2025

Résumé

Ce document synthétise les huit travaux pratiques réalisés dans le cadre du cours de data science appliquée à la finance. Ces TPs couvrent un large spectre de techniques allant du scraping de données financières à l'application de modèles de machine learning classiques et profonds, en passant par des analyses de sentiment basées sur des modèles de langage de type BERT. L'objectif est de construire une chaîne de traitement complète permettant d'exploiter des données de marché et textuelles pour enrichir la prise de décision financière.

1 Introduction

Dans un environnement où les marchés financiers sont de plus en plus influencés par la disponibilité de l'information et la rapidité de son traitement, les méthodes de data science deviennent des outils incontournables pour l'analyste financier moderne.

Cette série de travaux pratiques vise à démontrer comment mettre en place une chaîne de traitement complète pour l'analyse financière assistée par l'intelligence artificielle. L'approche adoptée combine plusieurs disciplines :

- La collecte automatisée de données financières à l'aide d'API (notamment `yFinance` et `NewsAPI`).
- L'analyse exploratoire des ratios fondamentaux et des séries temporelles de prix.
- L'utilisation de techniques non supervisées (clustering) pour identifier des groupes d'actifs homogènes.
- L'apprentissage supervisé pour prédire des signaux d'achat ou de vente, ainsi que les prix futurs.
- L'usage de modèles de deep learning (RNN, LSTM) adaptés aux séries temporelles.
- Le traitement de données textuelles et l'analyse de sentiment via des modèles NLP de type BERT.

L'objectif final est de démontrer l'intérêt d'une approche intégrée, où les données numériques (prix, ratios) sont enrichies par des données qualitatives (news), pour construire des outils d'aide à la décision robustes, interprétables et réplicables.

Fondements mathématiques et techniques

Les méthodes utilisées dans ce projet s'appuient sur des fondements mathématiques solides, tels que :

- Les modèles linéaires et non-linéaires pour la prédiction de variables continues (régression).
- La théorie des probabilités et les statistiques inférentielles pour estimer la qualité des modèles.
- L'optimisation numérique, pour entraîner les modèles d'apprentissage automatique.
- Les processus stochastiques et les séries temporelles, outils incontournables pour modéliser l'évolution des prix d'actifs.

Par exemple, pour un processus de prix P_t, on modélise souvent l'évolution relative par le rendement logarithmique :

$$r_t = \ln\left(\frac{P_t}{P_{t-1}}\right) \tag{1}$$

Ces rendements sont ensuite utilisés dans divers contextes : mesure de la volatilité, création de labels, entrée dans des modèles prédictifs.

2 Ètudes récentes combinant sentiment et prédiction boursière

Plusieurs travaux montrent qu'un modèle combinant données de marché et analyse de sentiment améliore nettement les prévisions boursières. Voici trois exemples récents :

Shangshang Jin et al., 2023 Les auteurs étudient la prédiction boursière pilotée par le sentiment dans le secteur bancaire chinois. Ils présentent un modèle SB-LSTM qui intègre un score de sentiment extrait de l'indice Baidu. Le SB-LSTM surpasse nettement un LSTM classique en termes de précision de classification, ce qui montre l'intérêt de l'analyse de sentiment pour la prévision des cours. Les limites mentionnées sont la dépendance à une unique source de sentiments (Baidu Index) et l'absence de tests hors du secteur bancaire chinois. (thesai.org/...)

Zhuoyue Wang et al., 2024 Les auteurs présentent trois architectures de prévision : Fin-BERT Embedding LSTM, LSTM classique et DNN. Le modèle Fin-BERT Embedding LSTM combine le sentiment des actualités (via Fin-BERT) avec les tendances historiques des cours de clôture pour extraire conjointement caractéristiques numériques et textuelles. Après 77 epoch, il atteint une loss de test de 0,00083, un MAE de 173,67, un MAPE de 0,045 et une précision de 0,955, tout en nécessitant moins de temps de calcul que les autres architectures. Le LSTM arrive en deuxième position, la DNN en troisième, bien que l'écart entre Fin-BERT et LSTM reste faible. Les pistes futures incluent l'extension à de plus gros jeux de données, un approfondissement de l'analyse émotionnelle et l'ajout d'un mécanisme de détection des fausses informations. (arxiv.org/abs/2407.16150)

Andrea Ajello et al., 2023 Les auteurs construisent en temps réel un indice de sentiment financier (TFSI) à partir de tweets liés aux marchés financiers et de crédit. Ils montrent que c'est surtout le nombre de retweets et de « likes » qui fait bouger l'indice, plus que le score moyen du contenu. Cet indice suit bien les évolutions du marché et aide à prévoir les rendements du jour suivant. Juste avant et pendant les annonces de la Fed, le sentiment baisse, surtout quand les hausses de taux sont inattendues. (arxiv.org/abs/2305.16164)

En résumé, l'ajout d'une source textuelle (tweets, dépêches d'actualité, articles de presse, etc.) apporte un gain constant, quel que soit l'algorithme employé.

3 TP1 — Collecte de données financières avec `yFinance`

Objectifs et justification

L'accès aux données financières est une condition sine qua non pour toute modélisation en finance quantitative. Le module `yFinance`, développé autour de l'API publique de Yahoo Finance, permet d'accéder facilement à un ensemble riche d'indicateurs fondamentaux et de séries temporelles.

Les données extraites incluent :

— Des ratios financiers (évaluation, rentabilité, levier, croissance).

— Des historiques de prix à diverses fréquences (journalier, hebdomadaire, mensuel).

— Des informations sur le marché : capitalisation boursière, volume échangé, dividendes.

Ces données servent à alimenter des modèles d'apprentissage supervisé (classification, régression), ou non supervisé (clustering).

Scraping de ratios financiers

Nous avons collecté plusieurs ratios par entreprise, tels que :
— `Forward P/E` (Price to Earnings anticipé)
— `Beta` (sensibilité au marché)
— `Return on Assets` (ROA)
— `Debt/Equity` (structure du capital)
Ces métriques sont fondamentales en analyse financière :

$$\text{Forward P/E} = \frac{\text{Prix de l'action}}{\text{Bénéfices estimés par action}} \tag{2}$$

$$\text{ROA} = \frac{\text{Résultat net}}{\text{Total de l'actif}} \tag{3}$$

$$\beta_i = \frac{\text{Cov}(R_i, R_m)}{\text{Var}(R_m)} \tag{4}$$

où R_i est le rendement de l'actif i, et R_m celui du marché (souvent S&P500).

Scraping de données historiques de prix

Pour chaque actif sélectionné (AAPL, MSFT, TSLA...), nous avons récupéré les prix historiques sur cinq ans avec une granularité journalière. Ces données sont ensuite transformées pour calculer les rendements simples :

$$r_t = \frac{P_t - P_{t-1}}{P_{t-1}} \tag{5}$$

et les rendements log :

$$r_t^{\log} = \ln\left(\frac{P_t}{P_{t-1}}\right) \tag{6}$$

Nous avons également extrait des features techniques :
— Moyenne mobile (SMA, EMA) sur différentes fenêtres
— Volatilité mobile (écart-type glissant)
— Momentum
— Volume moyen
Ces séries sont visualisées à l'aide de bibliothèques comme `matplotlib` ou `plotly`, facilitant l'inspection visuelle et la détection d'anomalies.

Préparation pour les modèles

Les données ont été stockées sous forme de DataFrame pandas. Une attention particulière a été portée au nettoyage : gestion des valeurs manquantes, homogénéisation des timestamps, et alignement des séries temporelles. Nous avons également utilisé :
— `ffill` pour la propagation des valeurs.
— `dropna()` pour supprimer les lignes corrompues.
— Normalisation des variables (z-score) :

$$z_i = \frac{x_i - \mu}{\sigma} \tag{7}$$

Ce TP sert donc de fondation technique pour tous les TPs suivants, en assurant la robustesse et la réplicabilité du pipeline de collecte.

4 TP2 — Clustering des profils financiers

Théorie et motivation

Le clustering, ou regroupement non supervisé, consiste à partitionner un ensemble de données en sous-groupes homogènes appelés *clusters*, sans connaître les étiquettes à l'avance. En finance, ce type de méthode est utile pour regrouper des entreprises selon :

— Leurs caractéristiques fondamentales (ratios financiers)
— Leurs profils de volatilité ou de rendement
— Leur comportement de marché (correlation structure)

Cette approche permet d'identifier des régimes de marché, des anomalies sectorielles, ou encore de construire des portefeuilles diversifiés sur des bases structurelles plutôt qu'arbitraires.

Préparation des données

Les observations utilisées pour le clustering sont des vecteurs de dimension d, où chaque dimension représente un ratio financier normalisé :

$$x_i = \begin{bmatrix} \text{ROA}_i & \text{Beta}_i & \text{Forward P/E}_i & \cdots \end{bmatrix}$$

On applique une normalisation standard (centrée-réduite) pour que les variables soient comparables :

$$z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j} \tag{8}$$

Algorithmes utilisés

Trois méthodes principales ont été testées :

— **K-Means** : minimise la somme des distances intra-cluster (inertie), selon :

$$\text{Inertie} = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2$$

avec C_i le i-ème cluster, et μ_i son centroïde.

— **Clustering hiérarchique agglomératif** : construit un arbre (dendrogramme) de fusions successives selon une distance. On a utilisé le critère de liaison de Ward, qui minimise la variance intra-groupe :

$$D_{Ward}(A, B) = \frac{|A||B|}{|A| + |B|} \|\mu_A - \mu_B\|^2$$

— **DBSCAN** : identifie des clusters denses selon deux hyperparamètres :
  — ε : rayon de voisinage
  — minPts : nombre minimal de points dans ce voisinage
  Il est robuste au bruit, mais difficile à calibrer en haute dimension.

Évaluation de la qualité du clustering

Même sans étiquettes, on peut utiliser des métriques internes :

— **Silhouette Score** :

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)}$$

où a_i est la distance intra-cluster moyenne, et b_i la plus faible distance moyenne à un autre cluster. $s_i \in [-1, 1]$.

— **Distortion, Calinski-Harabasz, Davies-Bouldin**

On sélectionne le nombre optimal de clusters k par la méthode du coude ("elbow method") ou en maximisant le score de silhouette.

Réduction de dimension pour visualisation

La visualisation des clusters est rendue possible par des techniques de projection :
— **PCA** : projection linéaire maximale de variance
— **t-SNE** : méthode non linéaire de préservation des distances locales
— **UMAP** : approche topologique pour la structure des voisinages

[Insérer ici les graphes TSNE, PCA et dendrogrammes]

Applications possibles

— Détection de groupes sectoriels ou de stratégies similaires
— Segmentation pour l'entraînement de modèles différenciés par cluster
— Détection d'anomalies : points ne s'intégrant dans aucun cluster
—

5 TP3 — Classification du signal Buy / Hold / Sell

Motivation et approche

L'objectif est de formuler une tâche de classification supervisée à partir des rendements futurs des actions. En définissant des règles de seuil sur le rendement à 20 jours, on peut générer automatiquement des étiquettes de type Buy, Hold ou Sell — une forme de *labellisation auto-supervisée*.

Construction des labels : Soit $r_t^{(20j)}$ le rendement à horizon 20 jours depuis la date t :

$$r_t^{(20j)} = \frac{P_{t+20} - P_t}{P_t} \tag{9}$$

On définit ensuite les classes comme suit :

$$y_t = \begin{cases} \text{"Buy"} & \text{si } r_t^{(20j)} > 5\% \\ \text{"Sell"} & \text{si } r_t^{(20j)} < -5\% \\ \text{"Hold"} & \text{sinon} \end{cases}$$

Ce type de signal est simplifié mais peut constituer une première approximation de stratégie de trading ou d'alerte.

Extraction des caractéristiques (features)

Les variables explicatives utilisées pour la prédiction sont des indicateurs techniques largement utilisés dans le domaine de l'analyse technique, calculés à partir des prix historiques.
— Moyenne mobile simple (SMA) :

$$\text{SMA}_n(t) = \frac{1}{n} \sum_{i=0}^{n-1} P_{t-i}$$

— Relative Strength Index (RSI), qui mesure la vitesse et l'amplitude des mouvements :

$$\text{RSI}_t = 100 - \frac{100}{1 + RS}, \quad RS = \frac{\text{moy. gains}}{\text{moy. pertes}}$$

— MACD (Moving Average Convergence Divergence) :

$$\text{MACD}_t = \text{EMA}_{12}(t) - \text{EMA}_{26}(t)$$

— Bandes de Bollinger :

$$\text{Bollinger Supérieure}_t = \text{SMA}_{20}(t) + 2\sigma_{20}(t)$$

Ces indicateurs sont calculés grâce à la bibliothèque Python `ta`, qui encapsule les méthodes techniques de finance.

Modèles d'apprentissage supervisé

Plusieurs algorithmes ont été testés et comparés :
— **Random Forest** : ensemble d'arbres de décision, robuste au surapprentissage.
— **XGBoost** : gradient boosting efficace, performant sur données tabulaires.
— **K-Nearest Neighbors (KNN)** : méthode paresseuse non paramétrique.
— **SVM (Support Vector Machine)** : trouve l'hyperplan maximal entre classes.
L'optimisation des hyperparamètres est effectuée avec `GridSearchCV`.

Évaluation Les performances sont évaluées à l'aide des métriques de classification :
— Précision (accuracy) :

$$\text{Accuracy} = \frac{\text{Nombre de prédictions correctes}}{\text{Nombre total de prédictions}}$$

— Matrice de confusion
— Précision, rappel, F1-score par classe
Un déséquilibre des classes peut apparaître si les seuils ± 5
— Rééchantillonner (oversampling, SMOTE)
— Ajuster les pondérations de classes dans la fonction de perte

Interprétabilité via SHAP

Pour comprendre les décisions des modèles comme XGBoost, nous avons utilisé la méthode SHAP (SHapley Additive exPlanations) (**?**), fondée sur la théorie des jeux coopératifs.

Chaque prédiction est décomposée comme une somme d'effets attributables à chaque variable :

$$f(x) = \phi_0 + \sum_{i=1}^{M} \phi_i \tag{10}$$

où ϕ_i est la contribution de la variable i à la prédiction.

[Insérer ici un SHAP summary plot avec les principales features]

Cette approche permet de rendre le modèle plus transparent pour un analyste métier, en révélant les variables qui influencent les prédictions de signal.

Limites et perspectives

Ce modèle reste basé sur une définition naïve des labels, ne tenant pas compte des frais de transaction, de la liquidité ou du timing réel des signaux. Pour aller plus loin :
— Intégrer des signaux multi-horizons
— Incorporer des variables fondamentales (ratios, news)
— Backtester les signaux dans un portefeuille simulé

6 TP4 — Régression sur les prix à J+1

Objectif et cadre du problème

Dans ce TP, nous cherchons à prédire la valeur future du prix d'une action à un horizon court (J+1), à partir d'une fenêtre glissante de données historiques. Cela constitue un problème classique de *régression supervisée sur séries temporelles.*

Plus formellement, pour une série de prix $\{P_t\}$, on définit :

$$X^{(t)} = [P_{t-30}, P_{t-29}, \ldots, P_{t-1}] \in \mathbb{R}^{30}, \quad Y^{(t)} = P_t$$

L'objectif est de modéliser la fonction $f : \mathbb{R}^{30} \to \mathbb{R}$ telle que :

$$\hat{P}_t = f(X^{(t)})$$

Prétraitement des données

Pour éliminer la non-stationnarité des prix, on peut travailler sur les rendements ou les prix normalisés :

$$x'_i = \frac{x_i - \mu}{\sigma} \quad \text{ou} \quad x_i^{\text{minmax}} = \frac{x_i - \min(x)}{\max(x) - \min(x)}$$

Cela permet de stabiliser les distributions et d'accélérer l'entraînement des modèles.

Modèles de régression testés

Plusieurs modèles classiques de régression ont été implémentés et comparés :

— **Régression Linéaire** :

$$\hat{y} = \beta_0 + \sum_{i=1}^{30} \beta_i x_i$$

Apprentissage par moindres carrés ordinaires (OLS). Sert de baseline interprétable.

— **Random Forest Regressor** : moyenne des prédictions d'un ensemble d'arbres construits sur des sous-échantillons aléatoires.

— **KNN Regressor** : prédit par moyenne locale des k voisins les plus proches.

— **XGBoost Regressor** : boosting par gradient, performant en tabulaire.

Évaluation des performances

Les métriques d'évaluation utilisées pour juger la qualité des prédictions sont classiques :

— **MAE — Mean Absolute Error** :

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

— **RMSE — Root Mean Squared Error** :

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$$

— R^2 — **Coefficient de détermination** :

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}$$

La performance réelle dépend largement du comportement du marché : en phase de forte volatilité, les erreurs augmentent. On observe aussi que les modèles non linéaires (XGBoost) surpassent souvent la régression linéaire, mais au prix d'une interprétabilité réduite.

Analyse des résidus

Un bon modèle de régression doit produire des erreurs (résidus) sans structure identifiable :

$$\epsilon_t = y_t - \hat{y}_t$$

Nous avons observé que les résidus présentent parfois une autocorrélation significative, ce qui suggère que des dépendances temporelles plus longues (non captées par la fenêtre glissante) peuvent être exploitées — d'où la transition vers des modèles séquentiels.

Limites et extensions possibles

- Les modèles ne prennent pas en compte d'autres variables explicatives comme les volumes, les indicateurs techniques ou les news.
- La prédiction du prix brut est plus difficilement exploitable en stratégie que la prédiction du *directional move* (hausse ou baisse).
- On pourrait enrichir le modèle par :
  - Des séries multivariées (prix, RSI, MACD, etc.)
  - Des données exogènes (indicateurs macroéconomiques)
  - Des modèles séquentiels (RNN, LSTM), explorés dans le TP suivant

7 TP5 — Réseaux de neurones pour la régression

Objectif

Ce TP poursuit la tâche de prédiction des prix à J+1, mais en utilisant des modèles d'apprentissage profond. Le but est d'exploiter la capacité des réseaux de neurones à modéliser des relations non linéaires complexes, en particulier dans le cadre des séries temporelles financières.

Présentation des architectures testées

Nous avons implémenté trois types de modèles dans `TensorFlow / Keras` :

1. Réseau de neurones multi-couches (MLP) Le MLP (Multi-Layer Perceptron) est constitué de plusieurs couches denses entièrement connectées :

$$\text{Layer } l: \quad h^{(l)} = \sigma\left(W^{(l)} h^{(l-1)} + b^{(l)}\right)$$

où σ est la fonction d'activation (ReLU, tanh, etc.).

Ce réseau est efficace pour capturer les interactions non linéaires entre les variables d'entrée, mais il ne tient pas compte de la structure temporelle.

2. Réseaux Récurrents (RNN) Les RNN sont adaptés aux données séquentielles. Chaque neurone d'une couche RNN prend en compte à la fois l'entrée actuelle et l'état caché précédent :

$$h_t = \tanh(W_x x_t + W_h h_{t-1} + b)$$

Cependant, les RNN classiques souffrent de problèmes de gradients (explosion/dispersion) pour les longues séquences.

3. Long Short-Term Memory (LSTM) Les LSTM améliorent les RNN en introduisant des mécanismes de mémoire avec des portes (input, forget, output), ce qui permet de mieux capter les dépendances longues.

Le fonctionnement interne peut se résumer par les équations suivantes :

$$\begin{aligned} f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \\ i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \\ \tilde{c}_t &= \tanh(W_c x_t + U_c h_{t-1} + b_c) \\ c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o) \\ h_t &= o_t \odot \tanh(c_t) \end{aligned}$$

où $\odot$ désigne le produit élément par élément, et σ la fonction sigmoïde.

Fonction de perte

Tous les modèles sont entraînés à minimiser l'erreur quadratique moyenne (MSE), définie par :

$$\mathcal{L}(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{11}$$

Elle punit davantage les grandes erreurs, ce qui est pertinent dans un cadre de prédiction de prix.

Fonctions d'activation et régularisation

Nous avons utilisé des activations non linéaires pour permettre au réseau d'apprendre des fonctions complexes :

— `ReLU`(x) = max(0, x)
— `tanh`(x) = $\frac{e^x - e^{-x}}{e^x + e^{-x}}$

Afin de réduire le risque de surapprentissage (*overfitting*), nous avons ajouté :

— `Dropout` : mise à zéro aléatoire de neurones pendant l'entraînement
— `EarlyStopping` : arrêt anticipé si la perte de validation ne s'améliore plus
— `BatchNormalization` pour stabiliser l'apprentissage

Résultats et interprétation

Nous avons comparé les performances des trois architectures sur un jeu de données de séries temporelles boursières. Le LSTM a généralement obtenu les meilleures performances sur le RMSE, confirmant son aptitude à capter les dynamiques temporelles. Le MLP est rapide à entraîner mais moins performant, et le RNN est instable sans réglages fins.

Limites et perspectives

— L'utilisation de réseaux plus profonds (stacked LSTM, GRU) pourrait améliorer la capture des régimes de marché complexes.
— Une approche **seq2seq** permettrait de prédire plusieurs jours à l'avance plutôt qu'un seul point.
— L'intégration d'attention (*transformers temporels*) est prometteuse pour les séries temporelles irrégulières.
— Il serait pertinent d'intégrer des données multi-sources (textuelles, fondamentales) pour enrichir les entrées du réseau.

8 TP6 — Scraping de News via NewsAPI

Objectifs

L'objectif de ce TP est d'automatiser la collecte d'actualités financières via l'API publique `NewsAPI`. Les articles ainsi collectés alimenteront par la suite des modèles d'analyse de sentiment, qui seront croisés avec les mouvements de prix pour détecter des signaux potentiels.

API : fonctionnement général

`NewsAPI` est une API REST qui permet d'effectuer des requêtes HTTP paramétrées selon :
— `q=` : mot-clé ou nom d'entreprise (ex : "Tesla")
— `from=` et `to=` : bornes temporelles (ex : "2024-12-01")
— `language=` : langue de l'article (ex : "en")
— `sources=` : filtrage sur des médias fiables (Bloomberg, WSJ, etc.)
— `pageSize=` : nombre maximum d'articles renvoyés
Chaque requête renvoie un document JSON contenant une liste d'articles.

Structure des articles collectés

Un exemple typique d'article collecté ressemble à ceci :

```
{
  "source": { "name": "Bloomberg" },
  "author": "John Smith",
  "title": "Tesla beats expectations in Q4 earnings",
  "description": "Tesla's stock rose after it reported stronger-than-expected results...",
  "url": "https://www.bloomberg.com/article...",
  "publishedAt": "2024-12-22T15:45:00Z",
  "content": "Full article text..."
}
```

Les champs importants pour nos traitements ultérieurs sont :
— **title** : résumé très condensé de l'information
— **description** : paragraphe d'introduction
— **publishedAt** : timestamp utile pour aligner avec les prix
— **source.name** : permet de restreindre à des médias reconnus

Stratégie de collecte et stockage

Nous avons écrit un script Python basé sur la bibliothèque `requests` pour effectuer des appels API en batch, sur plusieurs entreprises du S&P500. Pour chaque requête, on stocke les articles dans un fichier JSON local, en respectant la structure suivante :

```
{
  "ticker": "TSLA",
  "date": "2024-12-22",
  "articles": [ {...}, {...}, {...} ]
}
```

Nous avons géré les limitations de l'API gratuite (nombre de requêtes par jour) en répartissant les requêtes sur plusieurs jours et en sauvegardant systématiquement les logs.

Nettoyage et prétraitement du texte

Les textes collectés sont souvent bruyants (majuscule, ponctuation, mots de liaison). Un pipeline de prétraitement standard a été mis en place :
— Passage en minuscules
— Suppression de la ponctuation et des chiffres
— Suppression des mots vides (stopwords)
— Lemmatization (réduction à la racine des mots)
Ces étapes sont essentielles pour préparer les textes à l'analyse de sentiment (TP7), ou pour des modèles BERT nécessitant une tokenization propre.

Analyse exploratoire

Nous avons réalisé une première analyse exploratoire :
— Histogramme des articles par jour / entreprise
— Nuages de mots (*wordcloud*) sur les titres
— Distribution des longueurs de texte
Ces visualisations ont permis d'évaluer la richesse sémantique des articles et d'adapter les paramètres de scraping (langue, sources, fréquence).

Perspectives

Une telle base d'articles structurée ouvre la voie à de nombreuses applications :
— Analyse de sentiment supervisée (FinBERT)
— Détection de buzz médiatique (*news volume spike*)
— Construction d'un indicateur de news-to-price
— Génération automatique de résumés ou alertes (via NLP)

9 TP7 — Fine-tuning de BERT et FinBERT pour l'analyse de sentiment

Objectif

Ce TP vise à entraîner et comparer des modèles de traitement du langage (NLP) sur des articles de presse financière, afin de prédire le sentiment véhiculé par les textes (positif, neutre, négatif). Cette tâche est cruciale pour quantifier l'impact psychologique de l'actualité sur les marchés.

Présentation des modèles

BERT — Bidirectional Encoder Representations from Transformers Le modèle BERT (**?**) est un transformer bidirectionnel pré-entraîné sur un grand corpus de texte générique. Il repose sur deux tâches de pré-entraînement :
— *Masked Language Modeling (MLM)* : prédire un mot masqué dans une phrase
— *Next Sentence Prediction (NSP)* : prédire si deux phrases se suivent
BERT est un encodeur profond qui produit un vecteur dense pour chaque token d'entrée. La représentation du token `[CLS]` est utilisée pour les tâches de classification.

FinBERT FinBERT (**?**) est une version spécialisée de BERT, fine-tunée sur des corpus financiers. Il capture des nuances de langage spécifiques aux marchés :
— Ambiguïtés liées aux résultats financiers (« beat estimates » vs « fell short »)
— Tournures techniques propres aux rapports d'entreprises
— Volatilité émotionnelle dans les nouvelles économiques

Données utilisées

Nous avons utilisé deux jeux de données labellisées :
— **Financial Phrase Bank** : phrases issues de rapports annuels et d'actualités, labellisées manuellement (positif / neutre / négatif)
— **Twitter Financial Sentiment** : tweets financiers, plus courts mais plus volatils
Pour chaque texte, nous avons construit une triple (x, y) où :
— x : texte (titre ou phrase)
— $y \in \{0, 1, 2\}$: sentiment (négatif, neutre, positif)

Prétraitement NLP

Avant l'entraînement, nous avons appliqué une procédure standard :
— **Tokenization** avec le tokenizer BERT/FinBERT
— **Troncature/padding** à une longueur maximale (ex : 128 tokens)
— **Encodage** en tenseurs : input_ids, attention_mask
Les textes ont été batchés et passés dans un modèle de classification fine-tuné avec la bibliothèque `Transformers` de Hugging Face.

Fine-tuning et entraînement

L'entraînement repose sur la minimisation de l'entropie croisée :

$$\mathcal{L}(y, \hat{y}) = -\sum_{i=1}^{3} y_i \log(\hat{y}_i)$$

où $\hat{y}_i$ est la probabilité prédite pour la classe i, et y_i la variable indicatrice.
— Optimiseur : AdamW
— Taux d'apprentissage initial : 2×10^{-5}
— Epochs : entre 3 et 5 selon le jeu de données
— Batch size : 16 ou 32

Comparaison BERT vs FinBERT

Nous avons évalué les performances des deux modèles sur un jeu de validation :
— **Accuracy globale** : FinBERT > BERT
— **F1-score par classe** : meilleure détection du neutre avec FinBERT
— **Courbes de confusion** : FinBERT fait moins de confusions "positif/negatif"
FinBERT se montre bien plus adapté au langage économique et réduit l'ambiguïté des prédictions sur les phrases techniques.

Perspectives et limites

— Les textes longs (> 512 tokens) doivent être tronqués, ce qui peut faire perdre du contexte.
— L'analyse phrase par phrase ne capte pas toujours l'effet global d'un article.
— L'apprentissage auto-supervisé ou distant (via réaction des prix) pourrait enrichir l'étiquetage.
— Des architectures récentes comme RoBERTa, DeBERTa ou Longformer pourraient améliorer la performance sur des textes longs.

10 TP8 — Impact des sentiments sur les cours de bourse

Objectif général

Ce dernier TP vise à mettre en relation les signaux de sentiment extraits des articles de presse avec les mouvements de prix des actions concernées. L'objectif est de détecter une corrélation — voire une causalité — entre la tonalité des nouvelles et le comportement des actifs financiers.

Alignement temporel des données

Les données textuelles (issues du scraping de TP6 et de l'analyse de sentiment du TP7) sont associées à un timestamp précis, souvent au format UTC. Pour chaque article traité, nous avons associé :

— Le ticker de l'entreprise concernée (ex : AAPL, TSLA)
— Le score de sentiment prédéfini : $s \in \{0, 1, 2\}$ (négatif, neutre, positif)
— L'horodatage de publication t_{news}

Les données de prix étant collectées à fréquence horaire (via `yFinance` ou `alpaca`), il a été nécessaire d'aligner les timestamps selon les heures d'ouverture du NYSE :

$$t_{\text{aligned}} = \text{prochaine heure ouverte suivant } t_{\text{news}}$$

Méthodologie d'analyse

L'approche retenue consiste à regrouper les articles par tranche horaire, et à calculer :

— La **distribution de sentiment** par heure
— Le **rendement horaire** correspondant :

$$r_t = \frac{P_{t+1} - P_t}{P_t}$$

— La **corrélation entre sentiment et rendement**

Nous avons ensuite généré des séries temporelles synchronisées :

$$(\text{sentiment}_t, r_t) \quad \text{pour chaque intervalle horaire}$$

Des fenêtres mobiles de type rolling average (sur 3 ou 5 points) ont été utilisées pour lisser les courbes.

Visualisation et interprétation

Des graphiques superposant le score de sentiment et l'évolution du prix ont été produits :

— Série du sentiment moyen pondéré par heure
— Variation du prix alignée sur les timestamps
— Identification visuelle de l'impact des news positives/négatives

[Insérer ici une figure comparative : prix + signal FinBERT]

On observe parfois une réactivité forte du marché aux nouvelles très polarisées, notamment lorsqu'elles paraissent en dehors des heures normales, provoquant un "gap" à l'ouverture suivante.

Méthodes alternatives testées

En plus de l'analyse exploratoire, plusieurs pistes ont été testées :

— **Régression logistique** : prédire si le prix monte ou baisse selon le sentiment
— **Analyse de Granger** : tester la causalité entre le signal textuel et les rendements
— **Modèles hybrides** combinant BERT + LSTM pour intégrer texte + séries temporelles

Cependant, la faiblesse des données (échantillon réduit, bruit) limite la robustesse statistique.

Limites et biais

— Le sentiment ne garantit pas un impact immédiat : l'effet peut être différé ou annulé par d'autres nouvelles.
— Les articles peuvent être redondants ou non pertinents malgré un score positif.
— Le signal peut être noyé dans la volatilité naturelle du marché.

Perspectives

Pour améliorer la précision de l'analyse, on pourrait :
— Pondérer le sentiment par la source ou la visibilité médiatique
— Segmenter selon le moment de publication (avant-market, after-market)
— Intégrer le sentiment dans une stratégie de trading simulée (backtest)

11 Conclusion

Cette série de huit travaux pratiques illustre la construction progressive et cohérente d'un pipeline de data science appliquée à la finance. En partant de la collecte brute de données numériques et textuelles, nous avons progressivement enrichi notre analyse par des techniques d'apprentissage automatique et d'intelligence artificielle.

Les points saillants sont les suivants :
— **TP1–TP2** ont permis de comprendre la richesse des données financières disponibles, et comment les segmenter de manière non supervisée.
— **TP3–TP4** ont montré comment formaliser une tâche prédictive réaliste, en construisant des étiquettes simples (Buy/Hold/Sell) et en utilisant des modèles de régression pour anticiper les prix futurs.
— **TP5** a permis de franchir un cap en modélisation, en passant à des architectures de deep learning adaptées aux séries temporelles, comme les LSTM.
— **TP6–TP7** ont étendu le champ d'analyse au traitement automatique du langage, en intégrant des signaux textuels au cœur de la modélisation.
— **TP8** a enfin permis de croiser ces dimensions — textuelle et numérique — pour évaluer leur impact croisé sur le comportement des marchés.

Synthèse méthodologique

Ce travail met en évidence l'intérêt d'une chaîne intégrée de traitement :

1. Extraction automatisée des données financières et textuelles (scraping via API)
2. Prétraitement avancé (nettoyage, feature engineering, NLP)
3. Modélisation supervisée (classification, régression) et non supervisée (clustering)
4. Utilisation de modèles modernes (BERT, LSTM, XGBoost) et de visualisations interprétables (SHAP, t-SNE)
5. Alignement des temporalités pour des analyses croisées robustes

Apports et originalité

Ce projet illustre une complémentarité forte entre :
— Les **données quantitatives** : prix, ratios, indicateurs techniques
— Les **données qualitatives** : news, titres, descriptions, sentiments

Cette dualité permet de construire une représentation enrichie de l'environnement financier, ouvrant la voie à des modèles plus robustes et adaptatifs.

Pistes futures

Les extensions possibles sont nombreuses :
— Développement d'une application ou d'un tableau de bord interactif pour agréger ces analyses
— Utilisation de données exogènes : météo, mobilité, réseau social, alertes ESG
— Exploration de modèles récents comme les **transformers temporels** (Informer, TimeGPT)
— Apprentissage en ligne ou incrémental pour suivre l'évolution dynamique du marché
— Intégration dans des stratégies de trading algorithmique, avec backtest et gestion du risque

En somme, ce projet montre qu'il est aujourd'hui possible, avec des outils open-source et des données accessibles, de mettre en œuvre une intelligence artificielle appliquée à la finance — depuis le traitement de données brutes jusqu'à l'aide à la décision stratégique.