République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Abderrahmane Mira de Béjaïa

Faculté des Sciences Exactes
Département de Recherche Operationelles

Mémoire de Master
Option : Mathématique Financière

Thème

Prédiction des prix du Dogecoin : une analyse comparative de modèles d'apprentissage automatique

Présenté par :

Maouchi Fouad

encadré par :

Mr.Touche Nassim

Soutenu devant le jury composé de :

Président	Mme Djerroud Lamia	M.C.A	U. A/Mira Béjaïa
Examinatrice	Mme Amroun Sonia	M.C.B	U. A/Mira Béjaïa
Examinatrice	Mme Takhedmit Baya	M.C.A	U. A/Mira Béjaïa

2023 – 2024

Remerciments

Je tiens à exprimer ma profonde gratitude à toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce mémoire.

Tout d'abord, je souhaite remercier sincèrement mon encadrant, Monsieur Touche Nassim, pour son soutien constant, ses conseils avisés, et ses encouragements tout au long de ce travail. Sans son expertise et sa patience, ce mémoire n'aurait pas été possible.

Je remercie également les membres du jury, Mme Djerroud Lamia, Mme Amroun Sonia, et Mme Takhedmit Baya, pour leurs précieux commentaires et suggestions, qui ont grandement enrichi ce travail.

Je remercie tous les enseignants qui m'ont formé et guidé tout au long de mon parcours académique. Leur dévouement et leur passion pour l'enseignement ont été une source d'inspiration pour moi.

Je tiens également à exprimer ma profonde gratitude à ma mère, Boulkaria Karima, pour son amour inconditionnel, son soutien et ses sacrifices qui ont rendu possible la réalisation de ce mémoire. À ma sœur, Mayssen, et à mon frère, Islam, ainsi qu'à toute ma famille, merci pour votre soutien moral et vos encouragements constants.

Enfin, je souhaite remercier mes amis pour leur compréhension et leurs mots d'encouragement tout au long de cette période. Merci pour votre soutien et votre amitié.

Merci à tous pour votre aide et votre soutien.

Table des matières

Table des figures

Liste des tableaux

Introduction Générale

Les cryptomonnaies ont connu une croissance fulgurante ces dernières années, devenant une alternative attractive aux systèmes financiers traditionnels. Le Dogecoin, en particulier, s'est démarqué comme l'une des cryptomonnaies les plus populaires, attirant l'attention du grand public et des investisseurs du monde entier. Son succès est dû en grande partie à sa nature humoristique, à son association avec des célébrités comme Elon Musk et à son utilisation dans les micro-transactions[4]. Cependant, la volatilité élevée du marché des cryptomonnaies rend difficile la prédiction précise de l'évolution future des prix.

Récemment, le marché des cryptomonnaies a été marqué par des fluctuations importantes et des tendances changeantes[2]. Le Dogecoin n'a pas échappé à cette instabilité, ce qui rend d'autant plus crucial le développement de modèles de prédiction robustes pour aider les investisseurs à prendre des décisions éclairées.

Ce mémoire vise à comparer les performances de différents algorithmes de prédiction des prix du Dogecoin, en mettant l'accent sur les approches basées sur le machine learning et le deep learning. Plus précisément, nous allons évaluer la précision des modèles XGBoost, LightGBM, CatBoost et des réseaux neuronaux artificiels (ANN)[5, 8, 11] en utilisant des métriques de performance comme le RMSE, le MAE et le MSE[1].

Notre étude contribuera à identifier les modèles les plus performants pour la prédiction des prix du Dogecoin et à mettre en évidence les forces et les faiblesses de chaque approche. L'analyse de l'importance des différentes caractéristiques utilisées par les modèles permettra de mieux comprendre

les facteurs qui influencent le plus la volatilité du prix du Dogecoin.

En s'appuyant sur des données historiques et des indicateurs techniques[7], notre travail vise à fournir aux investisseurs un aperçu plus approfondi des dynamiques du marché des cryptomonnaies et à les aider à prendre des décisions d'investissement plus éclairées. De plus, l'analyse des limitations des modèles de prédiction permettra de mettre en lumière les risques inhérents à l'investissement dans les cryptomonnaies.

Ce mémoire est structuré en plusieurs chapitres :

Chapitre 1 : Ce chapitre explore les concepts fondamentaux du trading, des cryptomonnaies et des indicateurs techniques.

Chapitre 2 : Ce chapitre présente les méthodes de machine learning et de deep learning utilisées dans nos algorithmes, en mettant l'accent sur les algorithmes de boosting (XGBoost, LightGBM, CatBoost) et les réseaux neuronaux artificiels (ANN).

Chapitre 3 : Ce chapitre présente et analyse les résultats obtenus à partir des différentes méthodes de prédiction, et propose des recommandations basées sur ces résultats.

Chapitre 1

Concepts fondamentaux du trading et de l'analyse technique

1.1 Introduction

La prédiction des prix des cryptomonnaies est un sujet de grande importance dans le domaine financier. Ce chapitre explore les concepts fondamentaux liés au trading, aux cryptomonnaies et aux indicateurs techniques. Le trading des cryptomonnaies implique des stratégies spécifiques. Les indicateurs techniques jouent également un rôle crucial en fournissant des outils pour évaluer les mouvements de prix et la volatilité. Cette introduction pose les bases nécessaires pour comprendre les techniques et les outils utilisés dans la prédiction des prix des cryptomonnaies.

1.2 Les séries financières et leurs propriétés

Les séries financières, qui représentent des séquences chronologiques de prix d'actifs financiers, sont au cœur de notre étude sur la prédiction des prix des cryptomonnaies. Ces séries possèdent des propriétés distinctives telles que la non-stationnarité, la volatilité, l'auto-corrélation, et l'effet de levier. La compréhension de ces propriétés est essentielle pour l'application efficace des modèles de machine learning et du deep learning.

Les séries financières de cryptomonnaies comme Dogecoin sont particulièrement volatiles, ce qui signifie que les fluctuations de prix peuvent être rapides et significatives.

1.3 Les prévisions

Les prévisions, dans le contexte des marchés financiers , se réfèrent à l'art et la science de prédire les mouvements futurs des prix des actifs basés sur l'analyse des données historiques, des tendances de marché, et divers indicateurs économiques et techniques. Les prévisions sont essentielles pour les investisseurs, les traders et les analystes financiers pour prendre des décisions concernant l'achat, la vente d'un actif[10].

Les méthodes de prévision peuvent être divisées en plusieurs catégories :

— Analyse technique,
— Analyse fondamentale,
— Modèles de machine learning,
— Sentiment du marché.

1.4 Le trading

Le trading est une activité financière consistant à acheter et vendre des actifs tels que des actions, des devises, des matières premières ou des cryptomonnaies sur des marchés financiers dans le but de réaliser des bénéfices.

Les échanges offrent une variété d'instruments financiers, y compris les contrats à terme, les options, et les dérivés, permettant aux traders d'accéder à une gamme diversifiée d'opportunités de trading.

1.5 La cryptomonnaie

Une cryptomonnaie est une forme de monnaie numérique conçue pour fonctionner comme un moyen d'échange sécurisé et décentralisé. Contrairement aux monnaies traditionnelles émises par les gouvernements, les cryptomonnaies sont généralement basées sur une technologie appelée blockchain, qui utilise la cryptographie pour sécuriser les transactions et contrôler la création de nouvelles unités de monnaie[2].

Dogecoin est une cryptomonnaie créée en 2013 comme une parodie des cryptomonnaies existantes. Inspiré par le mème "Doge", son logo est un

Shiba Inu avec des phrases en anglais approximatif. La communauté Dogecoin est connue pour ses activités philanthropiques et ses collectes de fonds, soutenant des causes diverses. Dogecoin est populaire pour les microtransactions en ligne et est souvent influencé par les tendances des réseaux sociaux et des célébrités comme Elon Musk[4].

1.6 Les chandeliers japonais

Les chandeliers japonais, également connus sous le nom de ”candlestick charts” en anglais, sont un type de graphique utilisé dans l'analyse technique pour représenter les mouvements des prix d'un actif financier, comme une action, une devise ou une cryptomonnaie, sur une période donnée. Ces graphiques fournissent des informations sur le prix d'ouverture, le prix de clôture, ainsi que les prix les plus élevés et les plus bas pendant la période considérée[7].

Chaque chandelier japonais est composé de trois parties principales :

1. Le corps : représente la plage entre le prix d'ouverture et le prix de clôture de la période. Si le prix de clôture est supérieur au prix d'ouverture, le corps est généralement coloré en vert ou blanc pour indiquer une tendance haussière. Si le prix de clôture est inférieur au prix d'ouverture, le corps est coloré en rouge ou noir pour indiquer une tendance baissière.
2. Les mèches ou les ombres : représentent les prix les plus élevés et les plus bas atteints pendant la période, par rapport au prix d'ouverture ou de clôture.
3. La ligne médiane : une ligne fine qui connecte le prix le plus élevé et le plus bas de la période, mais qui n'est pas toujours incluse dans tous les graphiques de chandeliers.

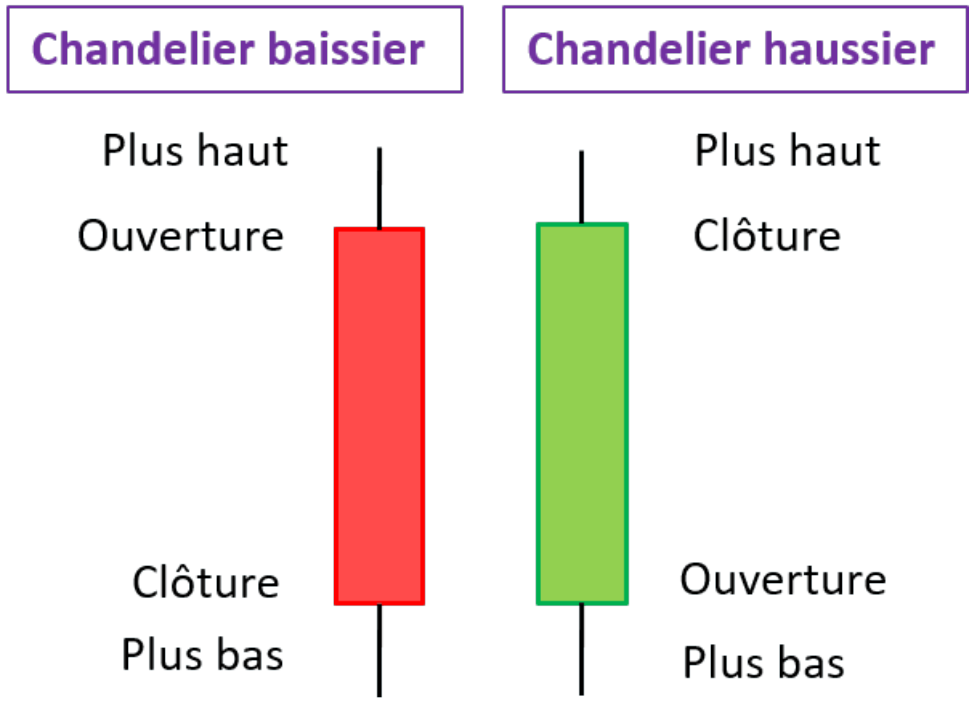

FIGURE 1.1 – Les chandeliers japonais

1.7 Les indicateurs techniques

Les indicateurs techniques sont des outils utilisés dans l'analyse technique [7] en finance pour aider à interpréter les mouvements de prix passés et actuels d'un actif financier. Ils sont basés sur des calculs mathématiques appliqués aux données de prix historiques. L'objectif principal des indicateurs techniques est d'aider les traders à identifier les tendances, les retournements de marché, les niveaux de support et de résistance, ainsi que les conditions de surachat ou de survente sur le marché.

Dans le cadre de la prédiction des prix des cryptomonnaies, les indicateurs techniques sont souvent utilisés comme caractéristiques d'entrée dans les modèles de machine learning et de deep learning. Ces indicateurs sont intégrés aux caractéristiques utilisées pour entraîner les modèles et peuvent jouer un rôle important dans la capacité des modèles à capturer les tendances et les motifs des données de prix.

1.7.1 La moyenne mobile

La moyenne mobile est un indicateur technique largement utilisé en analyse technique pour lisser les fluctuations à court terme d'une série de données, telles que les prix des actifs financiers. Elle est utilisée pour identifier la tendance générale des données en éliminant les variations aléatoires ou le bruit de marché.

La moyenne mobile est calculée en prenant la moyenne arithmétique des valeurs d'une série de données sur une période spécifiée. Cette moyenne est ensuite déplacée progressivement sur la série de données, point par point, pour créer une série de valeurs lissées.

Il existe plusieurs types de moyennes mobiles, notamment :

1. Moyenne Mobile Simple (SMA) : La moyenne mobile simple est calculée en prenant la moyenne arithmétique des prix sur une période donnée. Par exemple, pour une SMA sur une période de 10 jours, vous ajoutez les prix de clôture des 10 derniers jours et divisez le total par 10.

$$\text{SMA} = \frac{X_1 + X_2 + X_3 + \ldots + X_n}{n}.$$

2. Moyenne Mobile Exponentielle (EMA) : La moyenne mobile exponentielle donne plus de poids aux prix les plus récents, ce qui la rend plus réactive aux changements de prix. Elle est calculée en attribuant un poids décroissant exponentiellement aux prix passés.

$$\text{EMA}_t = \text{EMA}_{t-1} + \alpha \cdot (X_t - \text{EMA}_{t-1}).$$

3. Moyenne Mobile Pondérée (WMA) : La moyenne mobile pondérée attribue des poids différents aux prix sur la période spécifiée. Les prix les plus récents peuvent recevoir un poids plus élevé, ce qui les rend plus influents sur la moyenne.

$$\text{WMA} = \frac{w_1 \cdot X_1 + w_2 \cdot X_2 + \ldots + w_n \cdot X_n}{w_1 + w_2 + \ldots + w_n}.$$

1.7.2 Les ombres inférieures et supérieures des chandeliers japonais

Les ombres inférieures et supérieures des chandeliers japonais, également connues sous le nom de mèches ou de queues, représentent la différence entre le prix d'ouverture et le prix de clôture (pour l'ombre) ou entre le prix d'ouverture et les prix extrêmes de la période (pour les mèches). L'ombre inférieure est calculée en prenant la différence entre le prix d'ouverture et le prix le plus bas de la période, tandis que l'ombre supérieure est calculée en prenant la différence entre le prix le plus haut de la période et le prix d'ouverture. Ces calculs permettent de visualiser la volatilité et le comportement des prix pendant la période considérée :

Ombre inférieure (Lower Shadow) : $LowerShadow = Open - min(Close, Low)$.

Ombre supérieure (Upper Shadow) : $UpperShadow = max(High, Close) - Open$.

Où :

— Open représente le prix d'ouverture du chandelier.

— Close représente le prix de clôture du chandelier.

— Low représente le prix le plus bas atteint pendant la période.

— High représente le prix le plus haut atteint pendant la période.

1.7.3 La volatilité

La volatilité implicite est une estimation de la volatilité future de l'actif, dérivée des prix des options sur cet actif. Elle reflète les attentes du marché quant aux mouvements futurs des prix.Les actifs avec une volatilité élevée sont considérés comme plus risqués, car leurs prix peuvent varier considérablement sur de courtes périodes. Des augmentations soudaines de volatilité peuvent signaler des changements de tendance ou des événements importants sur le marché.

La volatilité historique est calculée en utilisant l'écart-type des rendements sur une période donnée. La formule est la suivante :

$$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(r_i - \bar{r})^2} \times \sqrt{T}.$$

où :

— σ est la volatilité historique.

— N est le nombre de périodes (jours, mois, etc.).

— r_i est le rendement du jour i.

— $\bar{r}$ est le rendement moyen sur N périodes.

— T est le nombre de périodes par an.

1.7.4 L'indice de force relative

L'indice de force relative (Relative Strength Index, ou RSI) est un indicateur technique utilisé dans l'analyse des marchés financiers pour mesurer la vitesse et la variation des mouvements de prix. Le RSI est principalement utilisé pour identifier les conditions de surachat ou de survente d'un actif, ce qui peut indiquer des opportunités d'achat ou de vente.

L'indice de force relative (RSI) est calculé en trois étapes :

1. Calcul des gains et des pertes moyens :

$$\text{Gain moyen} = \frac{\text{Somme des gains sur la période}}{\text{Nombre de périodes}},$$

$$\text{Perte moyenne} = \frac{\text{Somme des pertes sur la période}}{\text{Nombre de périodes}}.$$

2. Calcul de la force relative (RS) :

$$\text{RS} = \frac{\text{Gain moyen}}{\text{Perte moyenne}}.$$

3. Calcul du RSI :

$$\text{RSI} = 100 - \frac{100}{1 + \text{RS}}.$$

Interprétation du RSI :

RSI > 70 : L'actif est considéré comme suracheté, ce qui peut indiquer une correction ou un recul imminent.

RSI < 30 : L'actif est considéré comme survendu, ce qui peut signaler une opportunité d'achat.

RSI entre 30 et 70 : L'actif est généralement considéré comme étant dans une tendance stable.

1.7.5 Les rendements précédents

Les rendements précédents, également connus sous le nom de "returns" en anglais, sont des mesures qui indiquent le pourcentage de variation du prix d'un actif financier sur une période donnée. Ces rendements sont couramment utilisés dans l'analyse financière pour évaluer la performance passée d'un actif et pour aider à prédire ses mouvements futurs.

Les rendements précédents pour une période donnée sont calculés comme suit :

$$\text{Rendement} = \frac{\text{Prix de clôture actuel} - \text{Prix de clôture précédent}}{\text{Prix de clôture précédent}},$$

ou, exprimé en pourcentage :

$$\text{Rendement}(\%) = \left(\frac{\text{Prix de clôture actuel} - \text{Prix de clôture précédent}}{\text{Prix de clôture précédent}}\right) \times 100.$$

Dans le contexte des données financières, les rendements peuvent être calculés sur différentes périodes, telles que :

Rendement quotidien : Variation du prix d'un jour à l'autre.

Rendement hebdomadaire : Variation du prix d'une semaine à l'autre.

Rendement mensuel : Variation du prix d'un mois à l'autre.

Rendement annuel : Variation du prix d'une année à l'autre.

1.8 Les métriques d'évaluation

Erreur Quadratique Moyenne (MSE)

L'erreur quadratique moyenne (Mean Squared Error (MSE)) est une métrique d'évaluation utilisée pour mesurer la qualité d'un modèle de régression. Il représente la moyenne des différences au carré entre les valeurs observées (réelles) et les valeurs prédites par le modèle.[1]

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

où :

n est le nombre d'observations,

y_i est la valeur réelle,

$\hat{y}_i$ est la valeur prédite par le modèle.

Racine de l'erreur quadratique moyenne (RMSE)

La racine de l'erreur quadratique moyenne (Root Mean Squared Error (RMSE)) est la racine carrée de la MSE. Il a les mêmes unités que les valeurs observées, ce qui le rend plus interprétable par rapport à la MSE.[1]

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

Erreur absolue moyenne (MAE)

L'erreur absolue moyenne (Mean Absolute Error (MAE)) est une autre métrique d'évaluation utilisée pour mesurer la qualité des modèles de régression. Contrairement à la MSE, elle représente la moyenne des valeurs absolues des erreurs, c'est-à-dire la moyenne des différences absolues entre les valeurs observées et les valeurs prédites par le modèle.[1]

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

1.9 Conclusion

Ce chapitre a posé les bases en introduisant les concepts essentiels du trading, des cryptomonnaies, des chandeliers japonais et des indicateurs techniques ainsi les métriques d'évaluation. Ces éléments sont fondamentaux pour comprendre les dynamiques des marchés financiers et la manière dont les prix des cryptomonnaies peuvent être analysés. En établissant ce cadre de référence, nous sommes maintenant prêts à explorer des méthodes plus avancées de prédiction des prix. Le chapitre suivant se concentrera sur

les approches de machine learning et de deep learning, en examinant comment ces techniques avancées peuvent être appliquées pour améliorer la précision des prévisions et optimiser les stratégies de trading.

Chapitre 2

Quelques algorithmes d'apprentissage automatique et de l'apprentissage profond

2.1 Introduction

Dans ce chapitre, nous explorerons les concepts fondamentaux de l'apprentissage automatique et de l'apprentissage profond. Nous détaillerons les techniques et algorithmes spécifiques utilisés dans cette étude pour prédire les prix des cryptomonnaies, en mettant en lumière leurs avantages et inconvénients. Nous aborderons également les concepts clés et la terminologie liés à ces domaines pour mieux comprendre leur application dans notre contexte.

2.2 Apprentissage automatique (machine learning)

Le machine learning est une branche de l'intelligence artificielle qui se concentre sur le développement de systèmes capables d'apprendre et de s'améliorer automatiquement à partir de données sans être explicitement programmés. Les algorithmes de machine learning sont largement utilisés pour des tâches telles que la classification, la régression[9].

2.2.1 Apprentissage supervisé

L'apprentissage supervisé est un type de machine learning où l'algorithme apprend à partir d'un ensemble de données d'entraînement étiqueté. Chaque exemple d'entraînement est une paire constituée d'une entrée et de la sortie souhaitée, également appelée étiquette. L'objectif est de faire des prédictions ou des classifications basées sur de nouvelles données en s'appuyant sur les modèles appris à partir des données d'entraînement.

— **Régression** : Prédiction de valeurs continues. Exemple : Prédire le prix d'un actif.

— **Classification** : Attribution de catégories discrètes. Exemple : Déterminer si un email est un spam ou non.

2.2.2 Apprentissage non supervisé

L'apprentissage non supervisé est un type de machine learning où l'algorithme apprend à partir de données qui ne sont pas étiquetées. L'objectif est de découvrir des structures cachées ou des motifs dans les données. Les tâches courantes incluent le clustering et la réduction de dimension.

— **Clustering** : Regroupement d'exemples similaires. Exemple : Segmenter des clients en groupes ayant des comportements d'achat similaires.

— **Réduction de dimension** : Simplification des données tout en conservant leurs caractéristiques essentielles. Exemple : ACP (Analyse en Composantes Principales).

2.2.3 Méthodes d'ensemble

Les méthodes d'ensemble combinent plusieurs modèles de machine learning pour améliorer la performance prédictive et la robustesse par rapport à des modèles individuels. Elles exploitent la diversité des modèles pour réduire les erreurs de généralisation.

— **Bagging (Bootstrap Aggregating)** : Construction de multiples modèles en utilisant des sous-échantillons des données d'entraînement. Exemple : Random Forest.

— **Boosting** : Construction de modèles séquentiels où chaque modèle corrige les erreurs de prédiction de ses prédécesseurs. Exemple : XGBoost, LightGBM, CatBoost.
— **Stacking** : Combinaison de plusieurs modèles de base (ou de niveau 0) en utilisant un modèle de méta-apprentissage (ou de niveau 1) pour faire des prédictions finales.

2.3 Le boosting

Le boosting est une méthode d'ensemble en machine learning qui combine les prédictions de plusieurs modèles faibles pour produire un modèle fort . Le concept a été introduit par Robert Schapire en 1990 avec l'algorithme Adaboost (Adaptive Boosting), qui a marqué un tournant dans la manière dont les modèles de machine learning pouvaient être améliorés en séquentiellement corrigeant les erreurs des modèles précédents[13].

L'idée fondamentale derrière le boosting est de former un modèle faible à la fois, en mettant l'accent sur les erreurs des modèles précédents. À chaque étape, les observations mal prédites reçoivent une pondération plus élevée afin que le prochain modèle se concentre davantage sur ces observations difficiles. Le processus se poursuit jusqu'à ce qu'un nombre prédéfini de modèles soit formé, chaque modèle cherchant à corriger les erreurs de ses prédécesseurs.

2.4 Adaptive boosting

L'algorithme adaboost est le premier algorithme de boosting, combine plusieurs modèles faibles (souvent des arbres de décision de faible profondeur) en pondérant les erreurs de manière adaptative. Bien que puissant, Adaboost présente des limites, notamment une sensibilité au bruit dans les données[12].

Algorithme Adaboost pour la Régression

1. Initialiser les poids des observations $w_i = \frac{1}{N}$ pour $i = 1, 2, \ldots, N$.

2. Pour $m = 1$ à M :

   (a) Ajuster un régresseur $G_m(x)$ aux données d'entraînement en utilisant les poids w_i.

   (b) Calculer l'erreur err_m :

$$err_m = \frac{\sum_{i=1}^{N} w_i |y_i - G_m(x_i)|}{\sum_{i=1}^{N} w_i}.$$

   (c) Calculer le poids du régresseur α_m :

$$\alpha_m = \frac{1}{2} \log \left(\frac{1 - err_m}{err_m} \right).$$

   (d) Mettre à jour les poids des observations w_i :

$$w_i \leftarrow w_i \exp \left[\alpha_m \, |y_i - G_m(x_i)| \right], \quad i = 1, 2, \ldots, N.$$

   Normaliser les poids w_i pour qu'ils somment à 1.

3. Sortie finale $G(x)$:

$$G(x) = \sum_{m=1}^{M} \alpha_m G_m(x).$$

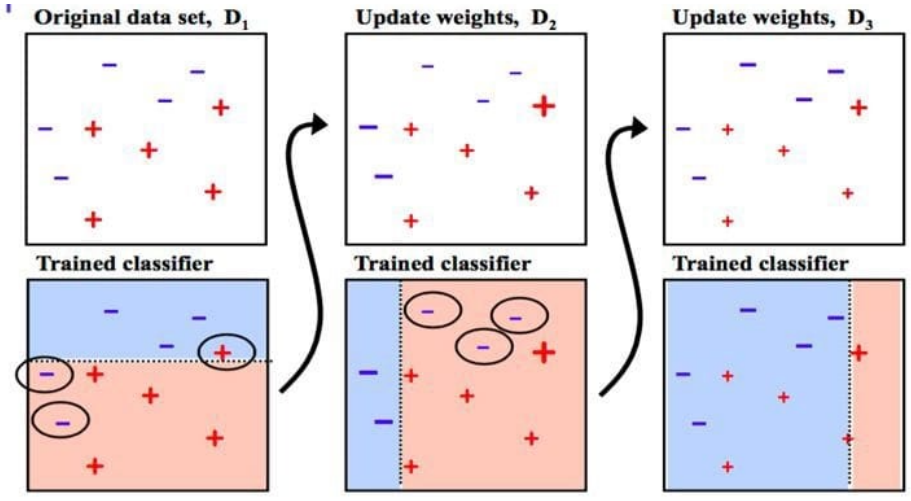

FIGURE 2.1 – La structure de adaboost

2.5 Le gradient boosting

Le gradient boosting, proposé par Jerome Friedman en 1999, a amélioré Adaboost en optimisant les erreurs de manière plus sophistiquée. Plutôt que d'utiliser une pondération adaptative, le gradient boosting utilise la descente de gradient pour minimiser l'erreur résiduelle de chaque modèle à chaque étape. Cela a conduit à des améliorations significatives en termes de précision et de robustesse.[12]

Algorithme de gradient boosting pour la régression

1. Initialiser le modèle avec une prédiction constante :

$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{N} \frac{1}{2}(y_i - \gamma)^2.$$

2. Pour $m = 1$ à M :
   (a) Calculer les résidus pseudo-résiduels pour chaque observation i :

$$r_{im} = y_i - F_{m-1}(x_i).$$

   (b) Ajuster un régresseur de base $h_m(x)$ aux résidus pseudo-résiduels r_{im}.
   (c) Calculer le facteur de mise à jour γ_m en résolvant :

$$\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{N} \frac{1}{2}(r_{im} - \gamma h_m(x_i))^2.$$

   (d) Mettre à jour le modèle :

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x).$$

3. Résultat final $F_M(x)$:

$$F_M(x) = F_0(x) + \sum_{m=1}^{M} \gamma_m h_m(x).$$

2.6 Extreme Gradient Boosting

Le XGBoost, est une méthode d'apprentissage supervisé basée sur le principe du gradient boosting qui utilise des arbres de décision pour les tâches de classification et de régression. Elle consiste à ajouter de nouveaux modèles qui corrigent les erreurs des modèles précédents. Les modèles sont ajoutés séquentiellement jusqu'à ce qu'aucune amélioration supplémentaire ne puisse être apportée. XGBoost est introduit par Tianqi Chen en 2016, a révolutionné le domaine grâce à ses optimisations de calcul, telles que la parallélisation des processus et la gestion efficace de la mémoire. Il est devenu populaire pour ses performances exceptionnelles dans de nombreuses compétitions de machine learning[11].

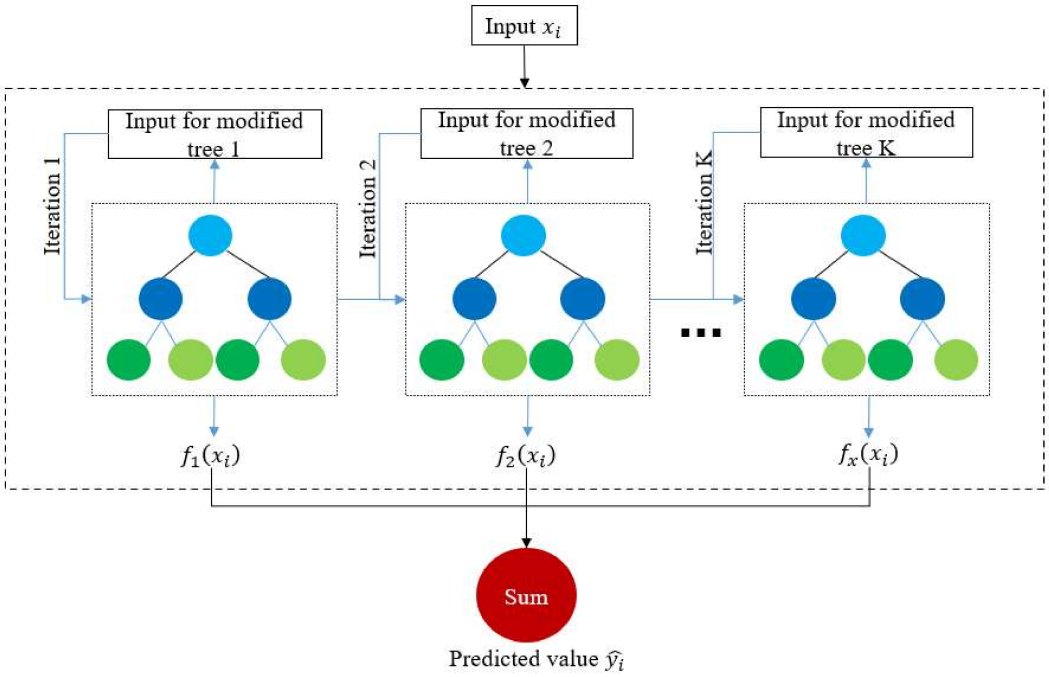

FIGURE 2.2 – La Structure générale de xgboost

2.6.1 Fonction de perte et terme de régularisation

La fonction de perte mesure l'écart entre les prédictions et les valeurs réelles. XGBoost optimise une fonction objectif composée de la fonction de perte et d'un terme de régularisation qui pénalise la complexité des arbres pour éviter le surapprentissage (overfitting).

Fonction de perte

Pour une tâche de régression, la fonction de perte courante est l'erreur quadratique moyenne (MSE) :

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

Pour une tâche de classification binaire, on utilise souvent la log-perte (logistic loss) :

$$L(y, \hat{y}) = -\sum_{i=1}^{n} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)].$$

Terme de régularisation

Le terme de régularisation Ω pour un arbre f_k est donné par :

$$\Omega(f_k) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2.$$

Où :

— T : Nombre de feuilles dans l'arbre.
— w_j : Poids de la feuille j.
— γ : Paramètre de régularisation pour le nombre de feuilles.
— λ : Paramètre de régularisation pour la taille des poids.

2.6.2 Objectif à minimiser

L'objectif total à minimiser dans XGBoost est[6] :

$$\text{Obj}(\theta) = \sum_{i=1}^{n} L(y_i, \hat{y}_i^{(t)}) + \sum_{k=1}^{t} \Omega(f_k).$$

2.6.3 Gradient et hessien

Pour chaque étape de boosting, XGBoost utilise une approximation de second ordre de la fonction de perte pour simplifier l'optimisation :

$$\text{Obj}^{(t)} \approx \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t(x_i)^2 \right] + \Omega(f_t).$$

Où :

— $g_i = \frac{\partial L(y_i, \hat{y}_i^{(t-1)})}{\partial \hat{y}_i^{(t-1)}}$: Gradient de la perte.

— $h_i = \frac{\partial^2 L(y_i, \hat{y}_i^{(t-1)})}{\partial \hat{y}_i^{(t-1)^2}}$: Hessien (seconde dérivée) de la perte.

2.6.4 Construction de l'arbre

Gain de l'information

Pour chaque nœud, XGBoost cherche à maximiser le gain de l'information pour déterminer la meilleure division :

$$\text{Gain} = \frac{1}{2}\left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda}\right] - \gamma.$$

où :

— G_L et G_R sont les sommes des gradients pour les feuilles gauche et droite.

— H_L et H_R sont les sommes des hessiens (secondes dérivées) pour les feuilles gauche et droite.

— λ est un paramètre de régularisation L2.

— γ est un terme de régularisation pour les divisions (splits).

Poids des feuilles

Le poids optimal w_j pour une feuille j est donné par :

$$w_j = -\frac{\sum_{i \in \text{leaf}_j} g_i}{\sum_{i \in \text{leaf}_j} h_i + \lambda}.$$

Valeur du gain pour une feuille

La valeur du gain pour une feuille avec le poids optimal est :

$$\text{Gain}_j = -\frac{1}{2}\frac{(\sum_{i \in \text{leaf}_j} g_i)^2}{\sum_{i \in \text{leaf}_j} h_i + \lambda}.$$

2.6.5 Prédiction finale

La prédiction finale pour une observation x est la somme des prédictions de tous les arbres :

$$\hat{y}(x) = \sum_{k=1}^{K} f_k(x).$$

2.6.6 Algorithme complet

1. **Initialisation** : Initialiser les prédictions $\hat{y}_i^{(0)}$ à une valeur initiale (par exemple, la moyenne pour la régression).
2. **Pour chaque itération t (de 1 à T)** :
   (a) Calculer les gradients g_i et les hessiens h_i.
   (b) Construire un nouvel arbre f_t en minimisant l'objectif avec les gradients et hessiens.
   (c) Mettre à jour les prédictions $\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t(x_i)$.
3. **Sortie** : La somme des arbres construits.

2.6.7 Avantages et inconvénients de XGBoost

Avantages :

— **Performance** : Très performant grâce à son implémentation optimisée.
— **Régularisation** : Offre à la fois la régularisation L1 et L2 pour éviter le sur-ajustement.
— **Simplicité** : Relativement facile à utiliser avec une bonne documentation.
— **Flexibilité** : Supporte différentes fonctions de perte et peut être utilisé pour des tâches de classification, régression.

Inconvénients :

— **Temps de Formation** : Peut être lent pour de très grands ensembles de données.
— **Consommation Mémoire** : Nécessite une quantité significative de mémoire.

— **Hyperparamètres** : Nombreux hyperparamètres à ajuster, ce qui peut être complexe.

2.7 Light Gradient Boosting Machine

LightGBM , développé par Microsoft c'est un algorithme d'apprentissage automatique performant, largement utilisé dans le domaine de l'apprentissage supervisé pour les tâches de régression et de classification LightGBM améliore encore le gradient boosting en utilisant une technique appelée arbres de décision basés sur les histogrammes, qui accélère considérablement l'entraînement sur de grands ensembles de données tout en maintenant une haute précision[5].

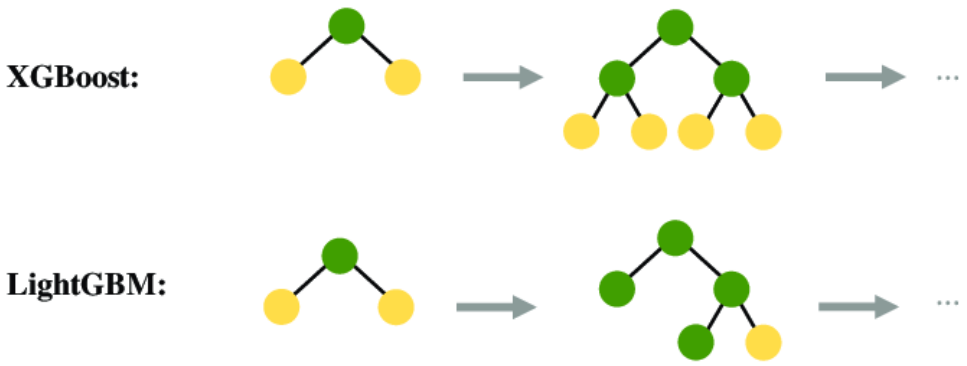

FIGURE 2.3 – La structure de XGBoost vs LightGBM

2.7.1 Pré-traitement des données

Encodage des variables catégorielles

LightGBM gère les variables catégorielles en les transformant en variables indicatrices (one-hot encoding) ou en utilisant des techniques d'encodage intégrées qui optimisent les splits dans les arbres.

2.7.2 Construction des arbres

Croissance feuille-par-feuille (Leaf-wise growth)

LightGBM choisit la feuille qui donne la plus grande réduction de perte pour se développer. Cela peut entraîner des arbres plus profonds et plus efficaces.

Formule de réduction de la perte

LightGBM utilise une formule similaire de gain de l'information à XGBoost pour déterminer la meilleure division de chaque nœud.

2.7.3 Boosting par gradient

LightGBM utilise le boosting par gradient pour construire les arbres successivement, de manière à corriger les erreurs des prédictions précédentes.

$$F_{m+1}(x) = F_m(x) + \gamma_m h_m(x).$$

où $F_m(x)$ est la prédiction du modèle au pas m, $h_m(x)$ est l'arbre ajouté au pas m, et γ_m est le taux d'apprentissage.

2.7.4 Division basée sur les histogrammes

LightGBM utilise une technique de binning pour accélérer le processus de split. Les valeurs continues des caractéristiques sont discrétisées en un nombre fixe de bins[5].

Formule de binning

Les valeurs d'une caractéristique x sont regroupées en k bins :

$$\text{Bin}_i = \left\{ x \mid \frac{i}{k} \leq \frac{\text{rank}(x)}{n} < \frac{i+1}{k} \right\}.$$

où $\text{rank}(x)$ est le rang de x parmi les valeurs de la caractéristique.

2.7.5 Regroupement de caractéristiques exclusives (EFB)

LightGBM utilise une technique appelée (Exclusive Feature Bundling EFB) pour réduire la dimensionnalité des données en regroupant les caractéristiques mutuellement exclusives[5].

Identification des caractéristiques mutuellement exclusives

Les caractéristiques qui ne sont jamais non nulles simultanément sont identifiées. Deux caractéristiques x_1 et x_2 sont considérées comme mutuellement exclusives si elles ne prennent jamais des valeurs non nulles en même temps.

Regroupement des caractéristiques mutuellement exclusives

Les caractéristiques identifiées comme mutuellement exclusives sont regroupées en une nouvelle caractéristique combinée. Cela permet de réduire le nombre total de caractéristiques tout en préservant les informations importantes.

Formellement, si nous avons deux caractéristiques x_1 et x_2, le regroupement peut être représenté comme suit :

$$x' = \text{combine}(x_1, x_2).$$

où x' est la nouvelle caractéristique résultant du regroupement de x_1 et x_2.

En pratique, cela signifie que les valeurs de x_1 et x_2 sont combinées en une seule caractéristique avant d'être utilisées pour la construction des arbres, réduisant ainsi la complexité computationnelle et améliorant l'efficacité du modèle.

2.7.6 Échantillonnage unilatéral basé sur le gradient (GOSS)

Identification des grands gradients absolus

Les échantillons avec de grands gradients absolus (forte influence sur la perte) sont identifiés[5].

Sous-échantillonnage des petits gradients

Les échantillons avec de petits gradients absolus sont sous-échantillonnés aléatoirement pour accélérer l'apprentissage.

Formule de GOSS

La formule de GOSS est la suivante :

$$\text{GOSS} = \frac{n}{2} \cdot \frac{g_i^2}{g_i^2 + \lambda}.$$

où g_i est le gradient de l'échantillon i et λ est un paramètre de régularisation.

2.7.7 Fonction de perte

LightGBM utilise des formules similaires à XGBoost.

2.7.8 Régularisation

LightGBM applique plusieurs techniques de régularisation pour éviter le sur-apprentissage, telles que :

— Régularisation L2 pour les feuilles des arbres.

— Pénalité de complexité des arbres en limitant la profondeur maximale des arbres et le nombre minimum d'échantillons par feuille.

2.7.9 Traitement des valeurs manquantes

LightGBM gère les valeurs manquantes en les traitant comme une catégorie distincte ou en apprenant des splits optimisés pour les valeurs manquantes.

2.7.10 Algorithme complet

1. **Initialisation** : Initialiser les prédictions $\hat{y}_i^{(0)}$ à une valeur initiale .
2. **Pré-traitement des données** :

(a) **Encodage des variables catégorielles** : Transformer les variables catégorielles en variables indicatrices (one-hot encoding) ou utiliser des techniques d'encodage intégrées.

(b) **exclusive feature bundling (Regroupement de caractéristiques)** : Identifier et regrouper les caractéristiques mutuellement exclusives en une seule caractéristique combinée.

3. **Pour chaque itération t (de 1 à T)** :

(a) Calculer les gradients g_i et les hessiens h_i pour chaque échantillon i.

(b) **Gradient-based One-Side Sampling (GOSS)** :

i. Identifier les échantillons avec de grands gradients absolus (forte influence sur la perte).

ii. Sous-échantillonner aléatoirement les échantillons avec de petits gradients absolus pour accélérer l'apprentissage.

(c) **Histogram-based splitting (Optimisation basée sur les histogrammes)** :

i. Discrétiser les valeurs des caractéristiques continues en un nombre fixe de bins.

ii. Calculer les histogrammes des gradients et hessiens pour chaque bin.

(d) Construire un nouvel arbre f_t en minimisant l'objectif avec les gradients et hessiens, en utilisant une croissance en profondeur des feuilles (leaf-wise growth).

(e) Mettre à jour les prédictions $\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + \gamma_t f_t(x_i)$, où γ_t est le taux d'apprentissage.

4. **Sortie** : La somme des arbres construits, c'est-à-dire la prédiction finale $\hat{y} = \sum_{t=1}^{T} \gamma_t f_t(x)$.

2.7.11 Avantages et Inconvénients de LightGBM

Avantages :

— **Vitesse** : Plus rapide que XGBoost, surtout pour les grandes bases de données.

— **Efficacité Mémoire** : Utilise moins de mémoire grâce à des techniques de histogram-based decision tree learning.
— **Scalabilité** : Bien adapté pour les très grandes bases de données avec des millions de lignes et des centaines de features.
— **Support pour les Catégories** : Gère efficacement les variables catégorielles.

Inconvénients :

— **Hyperparamètres** : Complexité dans le réglage des hyperparamètres.
— **Sensibilité aux Données Bruitées** : Peut être plus sensible aux données bruitées par rapport à XGBoost.

2.8 L'algorithme Catboost

CatBoost ,créé par Yandex ,conçue pour l'apprentissage automatique supervisé pour les tâches de régression et de classification. se distingue par sa gestion efficace des variables catégorielles. Il utilise des techniques avancées pour traiter les variables catégorielles directement, sans nécessiter de prétraitement comme encodage one-hot, réduisant ainsi le risque de surapprentissage et améliorant la performance[8].

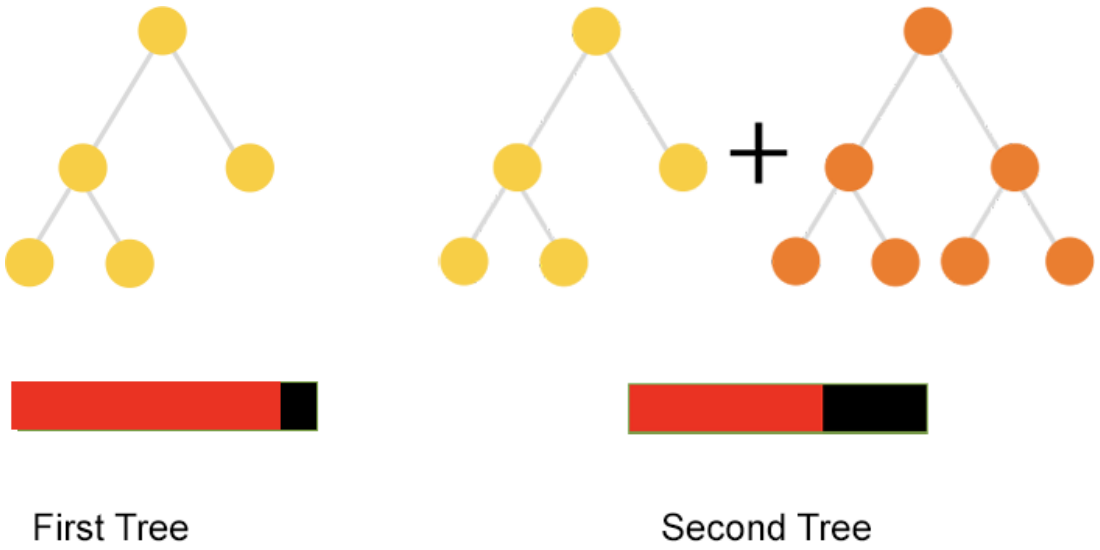

FIGURE 2.4 – La structure de Catboost

2.8.1 Fonction de perte et terme de régularisation

CatBoost utilise des formules similaires à XGBoost pour la fonction de perte et la régularisation.

2.8.2 Prétraitement des données catégorielles

Une des innovations majeures de CatBoost est la gestion efficace des caractéristiques catégorielles. Il utilise une méthode basée sur le codage des valeurs des catégories en fonction de l'historique des données pour chaque ensemble de données (training et validation).

Codage des catégories

CatBoost convertit les caractéristiques catégorielles en valeurs numériques en utilisant la statistique de codage basé sur la cible avec méthode de validation croisée de type un-à-un (leave-one-out) :

$$\text{EncodedValue}(x_i) = \frac{\sum_{j \neq i} \mathbb{I}(x_j = x_i) y_j + \alpha}{\sum_{j \neq i} \mathbb{I}(x_j = x_i) + \alpha}.$$

Où :

— $\mathbb{I}$ est une fonction indicatrice.

— α est un paramètre de lissage pour éviter les divisions par zéro.

2.8.3 Gradient et hessien

CatBoost utilise une formules similaires à XGBoost pour optimiser la fonction objectif.

2.8.4 Construction de l'arbre

Gain de l'information et les poids des feuilles

CatBoost utilise des formules similaires à XGBoost.

Boosting ordonné

CatBoost utilise une technique appelée Ordered Boosting pour améliorer l'efficacité de l'apprentissage en exploitant l'ordre naturel des données[8]. Cette méthode réduit la complexité computationnelle tout en maintenant de bonnes performances prédictives :

$$\text{OrderedBoost} = \frac{1}{N} \sum_{i=1}^{N} \text{loss}(y_i, F(x_i)) + \lambda \cdot \Omega(F).$$

Où :

— N est le nombre total d'échantillons dans le jeu de données.
— y_i est la vraie valeur de l'échantillon i.
— $F(x_i)$ est la prédiction du modèle pour l'échantillon i.
— loss est la fonction de perte utilisée pour évaluer les prédictions du modèle.
— λ est un paramètre de régularisation.
— $\Omega(F)$ est une fonction de régularisation qui contrôle la complexité du modèle.

2.8.5 Prédiction finale

La prédiction finale pour une observation x est la somme des prédictions de tous les arbres :

$$\hat{y}(x) = \sum_{k=1}^{K} f_k(x).$$

2.8.6 Algorithme complet

1. **Initialisation** : Initialiser les prédictions $\hat{y}_i^{(0)}$ à une valeur initiale.
2. **Pour chaque itération** t :
   (a) Calculer les gradients g_i et les hessiens h_i.
   (b) Construire un nouvel arbre f_t en minimisant l'objectif avec les gradients et hessiens.
   (c) Mettre à jour les prédictions $\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t(x_i)$.

3. **Sortie** : La somme des arbres construits.

2.8.7 Avantages et inconvénients de CatBoost

Avantages :

— **Gestion des Variables Catégorielles** : Excellente gestion des variables catégorielles avec un encodage ordonné.

— **Performance** : Bonne performance sur des données déséquilibrées et bruitées.

— **Facilité d'Utilisation** : Moins de réglages d'hyperparamètres nécessaires comparé à XGBoost et LightGBM.

— **Efficacité** : Rapide et efficace, même sans prétraitement intensif des données.

Inconvénients :

— **Temps de Formation** : Peut être plus lent que LightGBM pour des ensembles de données très larges.

— **Complexité** : Implémentation plus complexe due à l'encodage ordonné.

— **Support Communautaire** : Communauté et support peut-être moins vastes que ceux de XGBoost.

2.9 Paramètres et mécanismes d'optimisation

Dans les algorithmes XGBoost LightGBM et Catboost, il existe plusieurs paramètres et mécanismes qui jouent un rôle crucial dans l'optimisation des performances du modèle.

Paramètres spécifiques à XGBoost a. Max Depth :

— **Description** : Limite la profondeur maximale des arbres pour spécifier la complexité maximale de l'arbre.

— **Effet** : Les arbres peu profonds sont rapides à calculer, mais moins précis dans la capture de relations complexes. Les arbres plus profonds peuvent capturer des relations complexes mais risquent de surajuster les données bruitées.

b. Taux d'apprentissage (eta) :

— **Description** : Le taux d'apprentissage contrôle l'importance des mises à jour apportées lors de l'ajout de nouveaux arbres. Une valeur plus petite favorise l'ajout de nombreux arbres pour mieux généraliser tout en évitant l'overfitting.

— **Effet** : Réduction du taux d'apprentissage aide souvent à mieux généraliser le modèle mais nécessite plus d'itérations.

c. Min Child Weight :

— **Description** : Détermine la somme minimale des poids Hessiens (h_i) dans une feuille.

— **Effet** : Une valeur plus élevée empêche les arbres de croître trop profondément et conduit à une régularisation plus forte.

d. Gamma :

— **Description** : Contrôle la pénalisation pour ajouter une nouvelle feuille.

— **Effet** : Une valeur plus élevée augmente la pénalisation pour ajouter une nouvelle feuille, ce qui limite la croissance de l'arbre et réduit donc le risque d'overfitting.

e. Alpha :

— **Description** : Contrôle la régularisation L1.

— **Effet** : Une valeur plus élevée conduit à une pénalisation plus stricte des poids élevés, aidant ainsi à éviter le sur-ajustement.

Paramètres spécifiques à LightGBM a. Max Bin :

— **Description** : Nombre maximal de bins que les features peuvent prendre.

— **Effet** : Augmenter cette valeur peut améliorer la précision, mais augmente également la complexité et le temps de calcul.

b. Min Data in Leaf :

— **Description** : Nombre minimal d'échantillons nécessaires dans une feuille.

— **Effet** : Empêche les feuilles de devenir trop petites, réduisant ainsi le sur-ajustement.

c. Feature Fraction :

— **Description** : Fraction des features à considérer lors de la création de chaque arbre.

— **Effet** : Réduit la corrélation entre les arbres et améliore la généralisation.

d. Boosting Type :

— **Description** : Spécifie le type de boosting à utiliser (Gradient Boosting Decision Tree (gbdt), Random Forest, etc.)

— **Effet** : Permet de choisir différentes approches de boosting selon le problème et les données.

Paramètres spécifiques à CatBoost a. Depth :

— **Description** : La profondeur des arbres.

— **Effet** : Les arbres plus profonds peuvent capturer des relations plus complexes mais risquent de sur-ajuster les données bruitées.

b. L2 Leaf Regularization :

— **Description** : Terme de régularisation L2 sur les poids des feuilles.

— **Effet** : Pénalise les poids élevés sur les feuilles, aidant ainsi à éviter le sur-ajustement.

c. Border Count :

— **Description** : Nombre de frontières utilisées pour le bining des features numériques.

— **Effet** : Plus il y a de frontières, plus le modèle peut capturer des informations détaillées, mais au coût d'une plus grande complexité.

d. One-Hot Max Size :

— **Description** : Taille maximale pour utiliser l'encodage one-hot pour les features catégorielles.

— **Effet** : Les features catégorielles ayant moins de valeurs uniques que ce seuil seront encodées en one-hot, impactant la performance et la complexité.

Optimisation des modèles

1. Recherche par grille

— **Description** : Méthode pour chercher les meilleurs paramètres en explorant systématiquement une grille des valeurs possibles des paramètres.

— **Avantages** : Facile à implémenter et garantit de trouver la combinaison optimale dans l'espace défini.
— **Inconvénients** : Coût de calcul élevé pour des espaces de paramètres large.

2. Recherche aléatoire

— **Description** : Recherche des meilleurs paramètres en échantillonnant au hasard dans une distribution des valeurs possibles des paramètres.
— **Avantages** : Moins coûteux que la Grid Search et peut trouver une solution optimale ou quasi-optimale.
— **Inconvénients** : Pas exhaustif, risque de manquer la vraie combinaison optimale.

3. Hyperopt / Optimisation bayésienne

— **Description** : Méthodes avancées qui utilisent des approches bayésiennes pour modéliser la fonction but et choisir les paramètres d'entraînement.
— **Avantages** : Peut réduire significativement le nombre d'itérations nécessaires.
— **Inconvénients** : Plus complexe à mettre en œuvre et nécessite une bonne compréhension des approches bayésiennes.

2.10 Apprentissage profond (deep learning)

Le deep learning , est une sous-discipline du machine learning qui utilise des réseaux de neurones artificiels composés de nombreuses couches (ou profondeur) pour modéliser et résoudre des problèmes complexes.

2.10.1 Réseaux de neurones artificiels (Artificial Neural Networks)

Inspirés par le fonctionnement du cerveau humain, les ANN sont constitués de couches de neurones artificiels interconnectés qui permettent de réaliser des tâches d'apprentissage profond.

2.10.2 Structure des réseaux de neurones artificiels

Un réseau de neurone artificiel est composé de trois couches de neurones,

— **Couche d'entrée** : Reçoit les données brutes sous forme de vecteurs de caractéristiques.

— **Couches cachées** : Constituent le cœur du réseau, où chaque neurone applique une transformation linéaire suivie d'une fonction d'activation non linéaire. Ces couches permettent de capturer des structures et des relations complexes dans les données.

— **Couche de sortie** : Produit les prédictions finales du modèle, adaptées au type de tâche (par exemple, une classification ou une régression).

il existe plusieurs architectures de réseau de neurone :

— **Perceptron monocouche** : Un réseau de neurones simple composé uniquement d'une couche d'entrée et d'une couche de sortie avec une couche cachée.

— **Perceptron multicouches** : Un réseau de neurones plus complexe, composé d'une couche d'entrée, une couche de sortie et plusieurs couches cachées entre les deux.

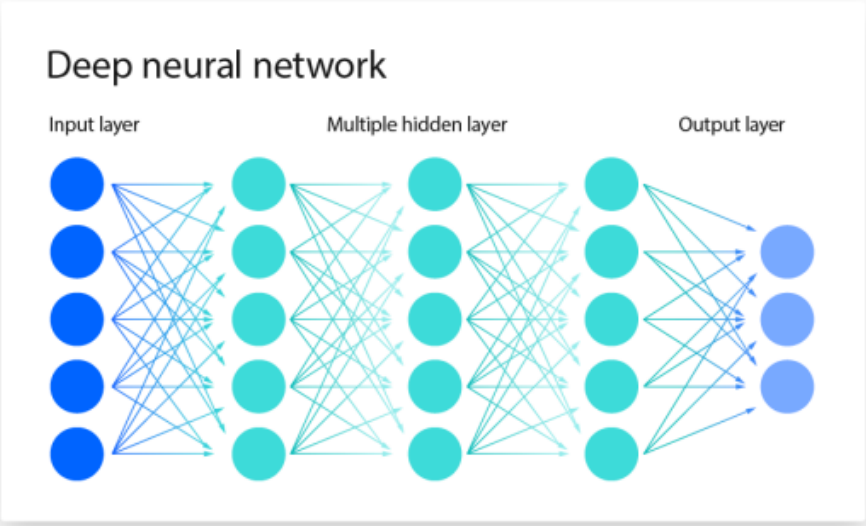

FIGURE 2.5 – La structure des ANN profond

2.10.3 Fonctionnement d'un neurone

Chaque neurone dans un réseau de neurones artificiels fonctionne en suivant des étapes spécifiques :

Entrées et poids

Le neurone reçoit des signaux d'entrée x_i, chacun étant associé à un poids w_i. Les poids déterminent l'importance de chaque entrée.

$$\text{Entrée pondérée} = \sum_i w_i x_i,$$

Ajout de biais

Un biais b est ajouté à l'entrée pondérée pour ajuster davantage le modèle et aider à mieux le décaler.

$$\text{Entrée totale} = \sum_i w_i x_i + b,$$

Fonctions d'activation La fonction d'activation f est appliquée à l'entrée totale pour introduire de la non-linéarité au modèle. Les fonctions d'activation les plus courantes sont : :

— **Sigmoïde** : $\sigma(x) = \frac{1}{1+e^{-x}}$,
— **Tanh** : $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$,
— **ReLU (Rectified Linear Unit)** : $\text{ReLU}(x) = \max(0, x)$,

Sortie

La sortie y du neurone est le résultat de l'application de la fonction d'activation à l'entrée totale.

$$y = f\left(\sum_i w_i x_i + b\right).$$

Apprentissage et optimisation

Les réseaux de neurones artificiels utilisent deux étapes clés pour apprendre à partir des données : la propagation avant (forward propagation) et la rétropropagation (backward propagation). Ces étapes sont essentielles pour entraîner efficacement le modèle.[14]

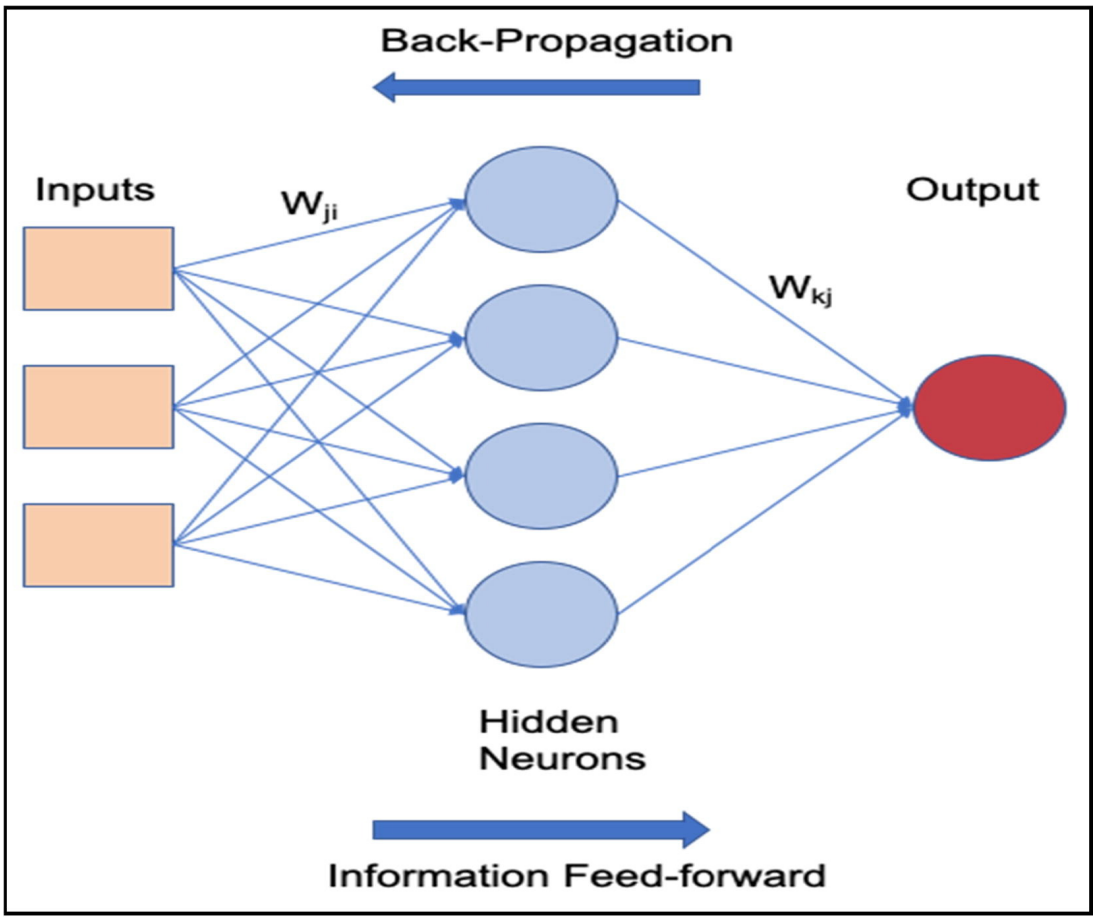

FIGURE 2.6 – La propagation arrière et avant

La propagation avant consiste à passer les données d'entrée à travers les couches du réseau, de la couche d'entrée à la couche de sortie, pour produire une prédiction.

La rétropropagation est utilisée pour ajuster les poids des connexions synaptiques afin de minimiser l'erreur entre la sortie prédite par le réseau et la sortie réelle, Ce processus est souvent réalisé à l'aide la descente de gradient stochastique (SGD) et Adam.[3]

2.10.4 Algorithme du réseau de neurone artificiel

1. **Initialisation** :
   — Initialiser les poids des connexions du réseau avec de petites valeurs aléatoires.
   — Définir le taux d'apprentissage η et le nombre d'itérations M.

2. **Pour chaque itération** $m = 1$ **à** M :
   (a) **Propagation avant** :
      — Pour chaque neurone de la couche cachée et de la couche de sortie, calculer l'activation en utilisant la somme pondérée des entrées plus le biais.
      — Appliquer une fonction d'activation (ex. sigmoïde, ReLU) pour obtenir la sortie de chaque neurone.
   (b) **Calcul de l'erreur** :
      — Calculer l'erreur de sortie en comparant les sorties du réseau avec les cibles réelles.
      — Calculer la dérivée de l'erreur par rapport aux sorties pour obtenir les gradients d'erreur.
   (c) **Propagation arrière** :
      — Propager les gradients d'erreur à travers le réseau en utilisant la règle de la chaîne.
      — Mettre à jour les poids en utilisant le taux d'apprentissage η et les gradients calculés.
3. **Sortie** :
   — Après M itérations, utiliser le réseau avec les poids appris pour effectuer des prédictions sur de nouvelles données.

2.10.5 Hyperparamètres des réseaux de neurones artificiels

Les hyperparamètres des réseaux de neurones artificiels (ANN) sont des paramètres définis avant la formation du modèle et influençant directement son apprentissage et sa performance. Voici une liste détaillée des hyperparamètres courants :

Taux d'apprentissage (Learning Rate)

— **Description :** Le taux d'apprentissage détermine la taille des étapes lors de l'ajustement des poids.
— **Impact :** Un taux trop élevé peut rendre le modèle instable et faire osciller les poids autour des minima. Un taux trop bas peut ralentir

considérablement l'entraînement, nécessitant plus d'itérations pour converger.

Nombre d'époques (Number of Epochs)

— **Description :** Le nombre d'époques représente le nombre de fois que l'algorithme parcourt l'ensemble des données d'entraînement.
— **Impact :** Un nombre trop élevé peut entraîner le surapprentissage (overfitting), où le modèle apprend trop bien les détails et le bruit des données d'entraînement.

Taille du lot (Batch Size)

— **Description :** La taille du lot correspond au nombre d'échantillons traités avant de mettre à jour les paramètres.
— **Impact :** Des tailles de lot plus petites peuvent offrir une meilleure généralisation car les mises à jour sont plus bruyantes, mais cela peut ralentir l'entraînement.

Nombre de couches cachées (Number of Hidden Layers)

— **Description :** Le nombre de couches cachées influe sur la capacité de modélisation du réseau.
— **Impact :** Plus de couches peuvent augmenter la capacité, mais accroissent également le risque de surapprentissage et allongent le temps d'entraînement.

Nombre de neurones par couche (Number of Neurons per Layer)

— **Description :** Le nombre de neurones par couche cachée augmente la capacité du modèle et sa complexité.
— **Impact :** Un plus grand nombre de neurones permet de modéliser des relations plus complexes, mais augmente également le risque de surapprentissage et le temps de calcul.

Fonction d'activation (Activation Function)

— **Description :** Les fonctions d'activation courantes incluent ReLU, Sigmoïde et Tanh.
— **Impact :** La fonction choisie peut influencer la vitesse et la qualité de l'apprentissage. Par exemple, ReLU est souvent utilisée pour ses bonnes propriétés de convergence.

Régularisation (Regularization)

— **Description :** Les méthodes courantes de régularisation incluent L1, L2 et Dropout.
— **Impact :** La régularisation aide à prévenir le surapprentissage en ajoutant des pénalités dans la fonction de coût pour des poids élevés ou en omettant aléatoirement des neurones pendant l'entraînement.

Momentum

— **Description :** Utilisé pour accélérer la descente de gradient et éviter les oscillations.
— **Impact :** Le momentum accumule les gradients des étapes précédentes pour mettre à jour les poids, ce qui permet de converger plus rapidement et d'éviter de rester coincé dans des minima locaux.

Algorithme d'optimisation (Optimization Algorithm)

— **Description :** Des algorithmes comme SGD, Adam et RMSprop sont utilisés pour optimiser le processus d'apprentissage.
— **Impact :** Chaque algorithme a ses propres avantages et inconvénients en termes de vitesse de convergence et de qualité d'optimisation. Adam, par exemple, combine les avantages d'Adagrad et de RMSprop et est souvent utilisé pour ses bonnes performances en pratique.

2.10.6 Avantages et inconvénients des réseaux de neurones artificiels

Les réseaux de neurones artificiels (ANN) sont des outils puissants pour résoudre une variété de problèmes complexes, mais comme toute technologie, ils présentent des avantages et des inconvénients.

Avantages

- **Capacité d'Apprentissage Complexe :** Les ANN sont capables d'apprendre et de modéliser des relations non linéaires complexes.
- **Adaptabilité :** Ils peuvent être ajustés et adaptés à presque tous les types de données (images, texte, son, etc.).
- **Performance Supérieure :** Lorsqu'ils sont bien entraînés, les ANN peuvent surpasser d'autres méthodes de machine learning pour des tâches complexes comme la reconnaissance d'images, la traduction automatique et le jeu stratégique.
- **Généralisation :** Avec une bonne régularisation et des données suffisantes, les ANN peuvent généraliser bien sur des données non vues.
- **Approche End-to-End :** Ils permettent une approche d'apprentissage de bout en bout, où les caractéristiques sont automatiquement extraites des données brutes.

Inconvénients

- **Besoins en Données :** Les ANN nécessitent de grandes quantités de données d'entraînement pour obtenir des performances optimales.
- **Nécessité de Puissance de Calcul :** L'entraînement des ANN, en particulier les réseaux profonds, peut être très exigeant en termes de ressources de calcul, nécessitant souvent du matériel spécialisé comme des GPU.
- **Complexité des Hyperparamètres :** Il y a de nombreux hyperparamètres à ajuster (nombre de couches, neurones par couche, taux d'apprentissage, etc.), ce qui peut rendre le processus de formation complexe et long.

— **Risques de Surapprentissage :** Les ANN peuvent facilement surentraîner (overfitting) sur les données d'entraînement, nécessitant des techniques de régularisation.
— **Temps de Convergence :** L'entraînement des ANN peut être lent, et il peut être difficile d'atteindre une convergence optimale, nécessitant de nombreux essais et ajustements.

2.11 Conclusion

Dans ce chapitre, nous avons introduit les concepts de machine learning et du deep learning, et décrit les algorithmes spécifiques utilisés dans cette étude : XGBoost, LightGBM, CatBoost et les réseaux de neurones artificiels. Chacun de ces algorithmes a ses propres avantages et inconvénients, que nous avons discutés en détail. Dans le prochain chapitre, nous présenterons les résultats obtenus en appliquant ces algorithmes à la prédiction des prix des cryptomonnaies, et nous comparerons leurs performances.

Chapitre 3

Comparaison des performances des algorithmes de machine learning pour la prédiction des prix du Dogecoin

3.1 Introduction

Dans ce chapitre, nous présentons une analyse des résultats obtenus à partir des différentes méthodes de prédiction des prix de la cryptomonnaie Dogecoin. Nous commencerons par une description des données utilisées dans cette étude. Ensuite, pour chaque méthode de prédiction (XGBoost, LightGBM, CatBoost, et les réseaux de neurones artificiels), nous discuterons de l'importance des caractéristiques, des performances sur les ensembles de données d'entraînement et de test, et des métriques d'évaluation. Nous conclurons ce chapitre par une comparaison entre les différentes méthodes.

3.2 Description des Données

— **Source des données** :il existe de nombreuses plate-forme et ressources à utiliser pour l'obtention des données sur les crypto-monnaies Nos données historiques des prix de Dogecoin ont été récupérées à partir de **YahooFinance**.

— **Caractéristiques (features)** : Les données comprennent des caractéristiques telles que :

  - Date,
  - Prix d'ouverture (Open),
  - Prix de clôture (Close),
  - Haut (High),
  - Bas (Low),
  - Volume des transactions (Volume),
  - les valeurs précédentes des prix de clôture,
- Ainsi des **indicateurs techniques** :
  - La moyenne mobile,
  - Ombres inférieures et supérieures des chandeliers japonais,
  - Volatilété,
  - L'indice de force relative,
  - Les rendements précédents.
- **Cible (target)** : Le prix de clôture (Close) est la variable cible (Y) que nous essayons de prédire.
- **Division des Données** : Pour évaluer la performance du modèle, nous avons divisé notre ensemble de données en deux parties : 80% des données ont été utilisées pour l'entraînement et 20% des données ont été réservées pour les tests. Cette division permet d'entraîner le modèle sur une grande partie des données tout en conservant un ensemble indépendant pour évaluer sa performance.
  - **Données d'entraînement (80%) :** Ces données sont utilisées pour entraîner le modèle. Le modèle apprend les relations et les patterns à partir de cet ensemble de données.
  - **Données de test (20%) :** Ces données sont utilisées pour évaluer la performance du modèle. Elles permettent de tester la capacité du modèle à généraliser à des données qu'il n'a jamais vues auparavant.

3.2.1 Exploration des données

Les données contient 7 variables et 2363 individues de la pair DOGE/USD de 2017-11-09 jusqu'a 2024-04-28 sous la forme suivante :

Date	Open	High	Low	Close	Adj Close	Volume
2017-11-09	0.001207	0.001415	0.001181	0.001415	0.001415	6259550
2017-11-10	0.001421	0.001431	0.001125	0.001163	0.001163	4246520
2017-11-11	0.001146	0.001257	0.001141	0.001201	0.001201	2231080
2017-11-12	0.001189	0.001210	0.001002	0.001038	0.001038	3288960
2017-11-13	0.001046	0.001212	0.001019	0.001211	0.001211	2481270
....	.....	.....	.....	.....	.....	.....
2024-04-28	0.147720	0.150610	0.147701	0.149782	0.149782	661729024

TABLE 3.1 – Le contenu du fichier DOGE/USD

Nous présenterons dans la figure 3.1 la visualisations graphiques des données de tendances générales observées sur la période étudiée sous formre des chandeliers japonais.

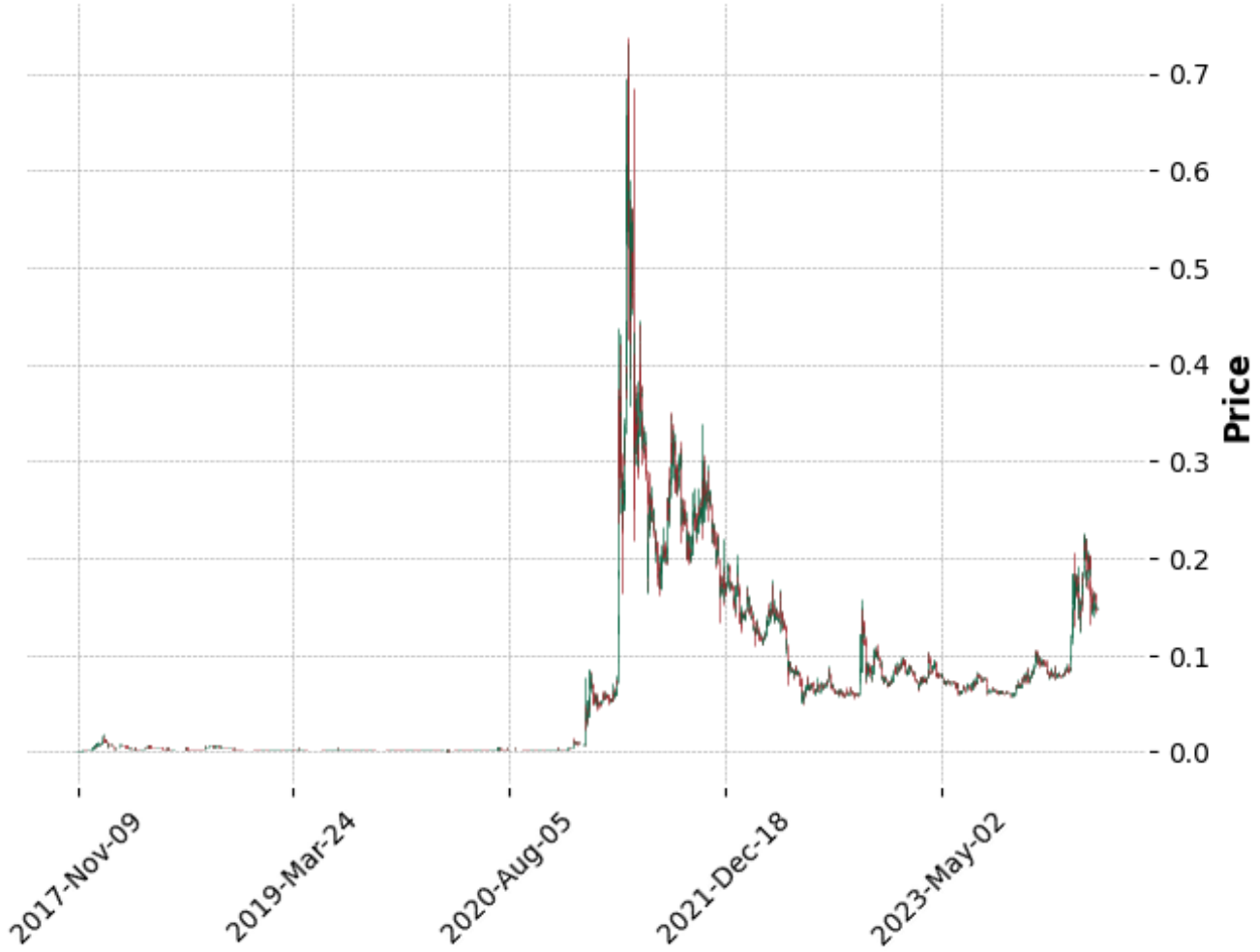

FIGURE 3.1 – La visualisation des données (DOGE/USD)

3.3 Application de l'algorithme XGBoost

Dans cette section, nous présentons les résultats de l'application du modèle XGBoost pour la prédiction des prix de cloture.

3.3.1 Importance des caractéristiques

Pour le modèle XGBoost, nous illustrons l'importance des différentes caractéristiques utilisées dans le modèle. Cela nous permettra d'identifier les variables les plus influentes dans la prédiction .

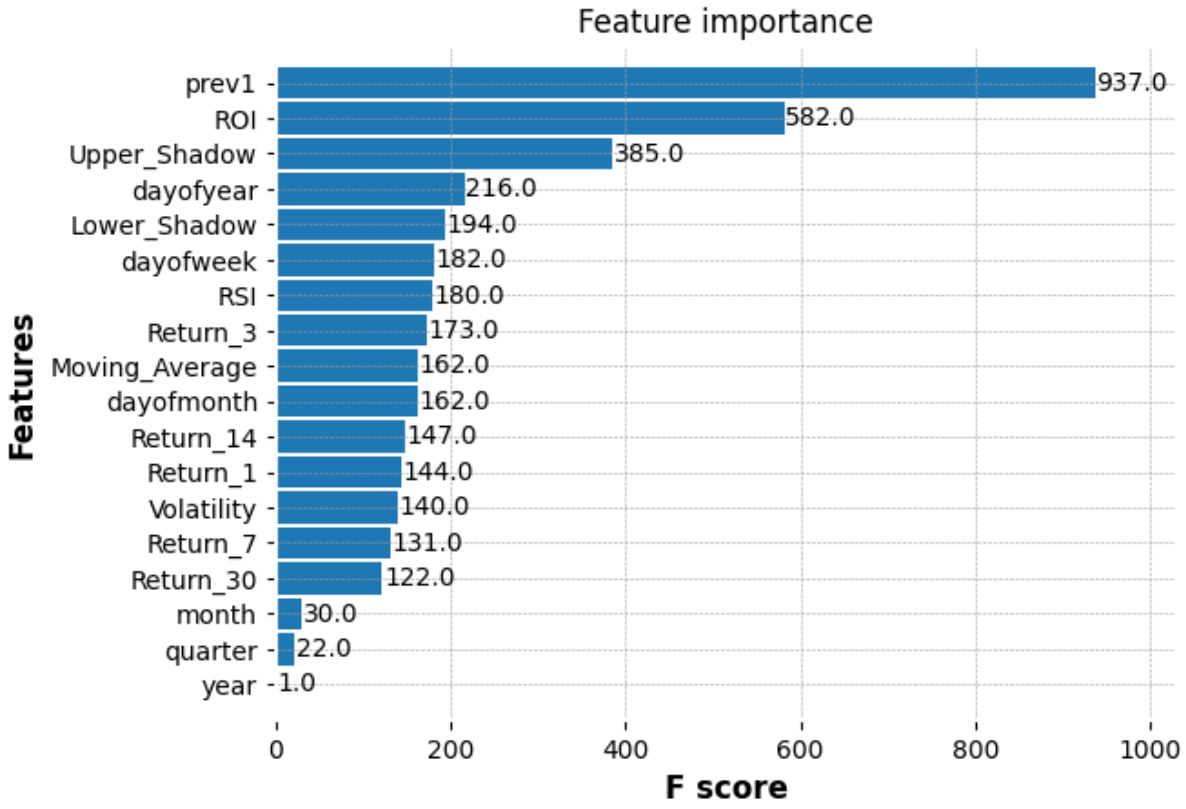

FIGURE 3.2 – L'importance des caractéristiques influentes dans XGBoost

Dans la figure 3.2 on remarque que l'analyse de l'importance des caractéristiques révèle que les variables **prev1, ROI et upper-shadow** ont significativement influencé les prédictions du modèle, indiquant que ces facteurs sont cruciaux pour expliquer les variations observées dans les résultats prédits.

3.3.2 Données d'entraînement

Nous visualisons les prédictions du modèle XGBoost sur les données d'entraînement, comparées aux prix réels, afin de vérifier comment le modèle s'adapte aux données d'apprentissage.

FIGURE 3.3 – Les prédictions sur les données d'entraînement de XGBoost

3.3.3 Données de test

Nous visualisons les prédictions sur les données de test, comparées aux valeurs réelles, pour évaluer la capacité du modèle .

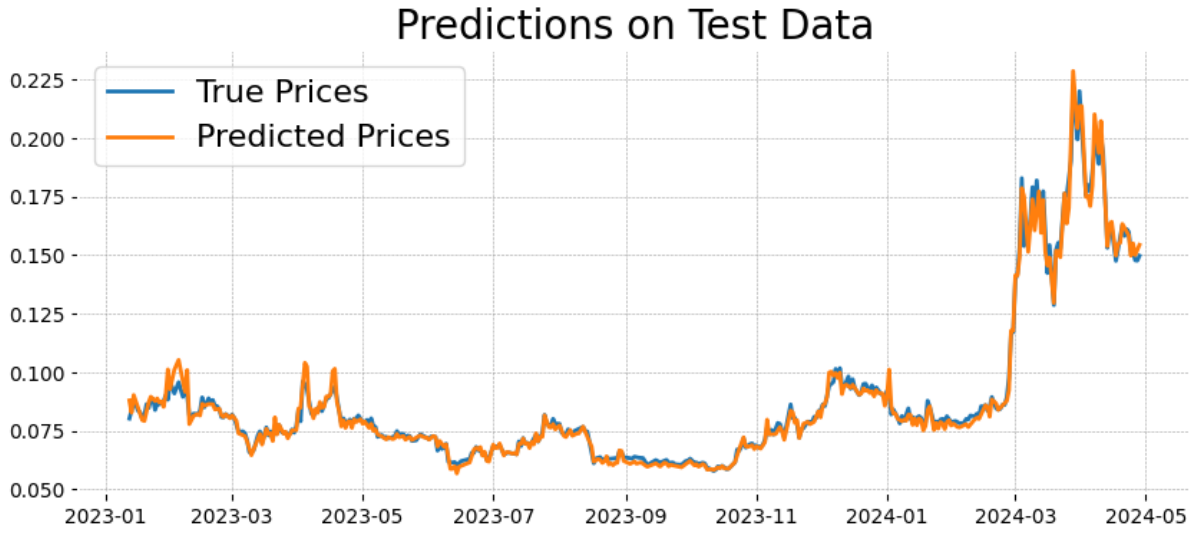

FIGURE 3.4 – Les prédictions sur les données de test de XGBoost

Dans la figure 3.4 on remarque que le modèle parvient à bien capturer les fluctuations des prix réels avec ses prédictions .

3.3.4 Les métriques d'évaluation

Nous évaluons les performances du modèle XGBoost en utilisant les métriques RMSE, MSE, et MAE, ainsi les prédictions des 5 derniers prix de clôture avec les prix réels.

	Données d'entraînement	Données de test
RMSE	0.0007466984799878073	0.003158622995854305
MAE	0.0002845053012160 9097	0.0019118318050865388
MSE	5.575586200161019e-07	9.976899229939626e-06

TABLE 3.2 – La performance du modèle XGBoost

	Prédictions	Prix Réels
2358	0.149840	0.151354
2359	0.155052	0.151394
2360	0.150105	0.147837
2361	0.153031	0.147716
2362	0.154432	0.149782

TABLE 3.3 – Les 5 dernières prédictions et réels de XGBoost

Dans le tableau 3.3 on peut dire que les valeurs prédites et les valeurs réelles sont très proches, ce qui un bon signe de la performance du modèle XGBoost .

3.4 Application de l'algorithme LightGBM

Nous présentons les résultats de l'application du modéle LightGBM pour la prédiction des prix de cloture.

3.4.1 Importance des caractéristiques

Nous illustrons l'importance des différentes caractéristiques utilisées dans le modèle LightGBM. Cela nous permettra d'identifier les variables les plus influentes dans la prédiction .

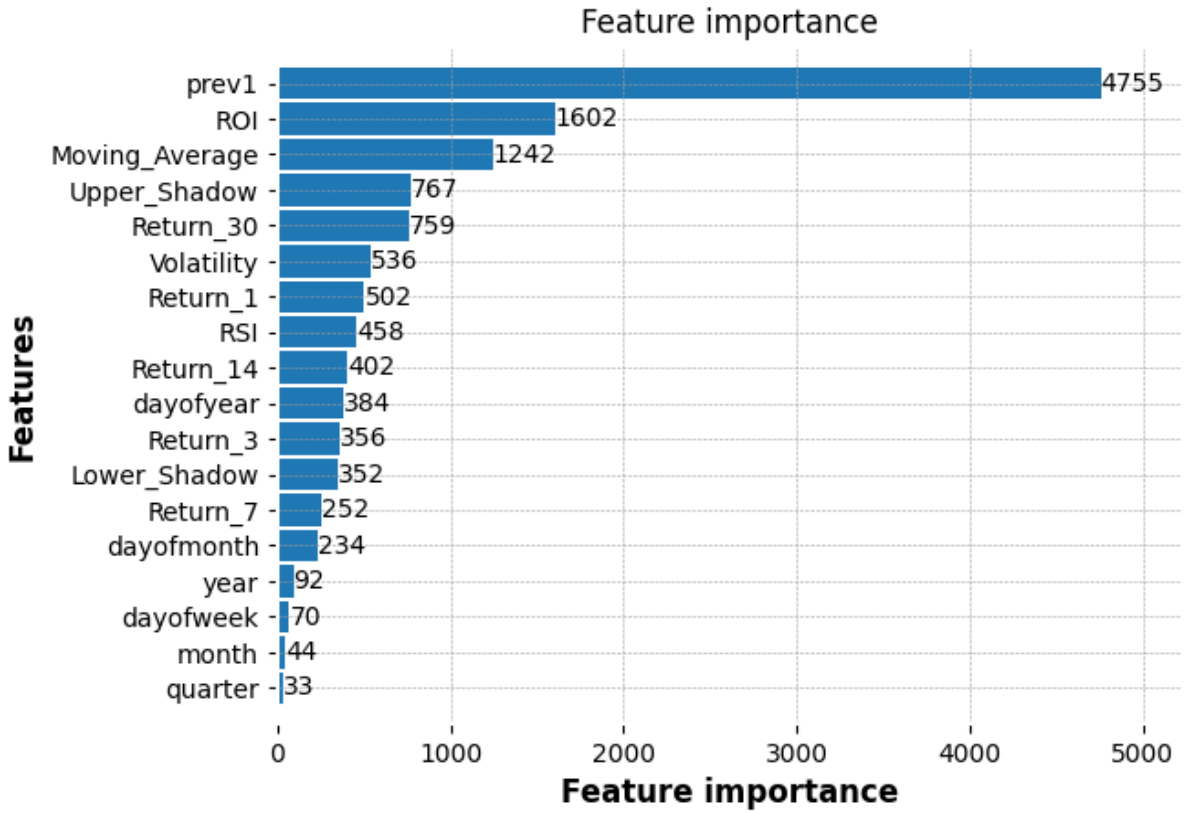

FIGURE 3.5 – L'importance des caractéristiques influentes dans LightGBM

Dans la figure 3.5 on remarque que l'analyse de l'importance des caractéristiques révèle que les variables **prev1, ROI et Moving-average** ont significativement influencé les prédictions du modèle, indiquant que ces facteurs sont cruciaux pour expliquer les variations observées dans les résultats prédits.

3.4.2 Données d'entraînement

Nous visualisons les prédictions du modèle LightGBM sur les données d'entraînement, comparées aux prix réels, afin de vérifier comment le modèle s'adapte aux données d'apprentissage.

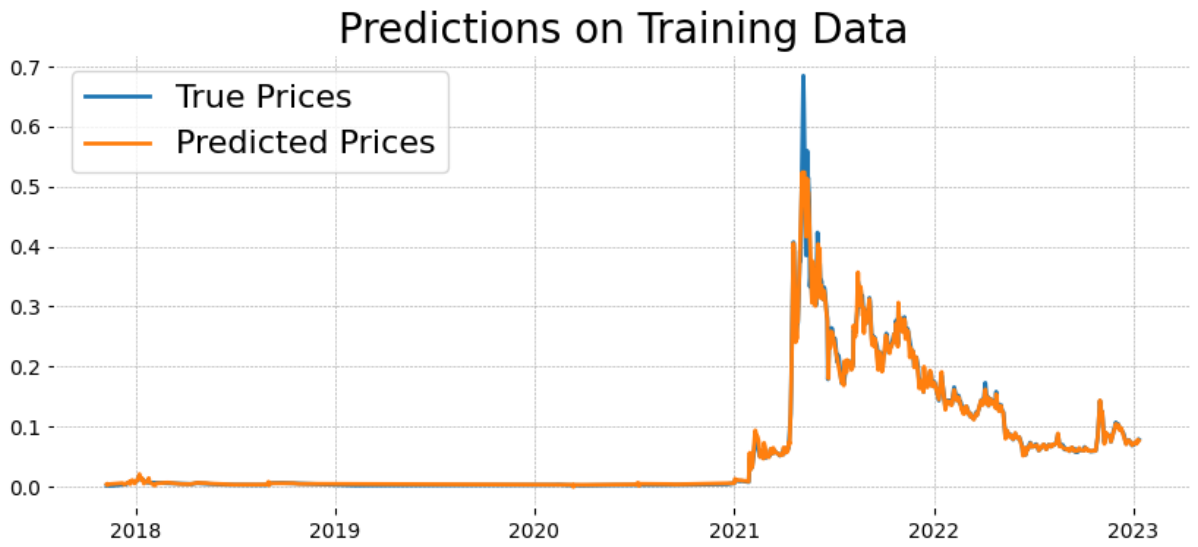

FIGURE 3.6 – Les prédictions sur les données d'entraînement de LightGBM

3.4.3 Données de test

Nous visualisons les prédictions sur les données de test, comparées aux valeurs réelles, pour évaluer la capacité du modèle LightGBM .

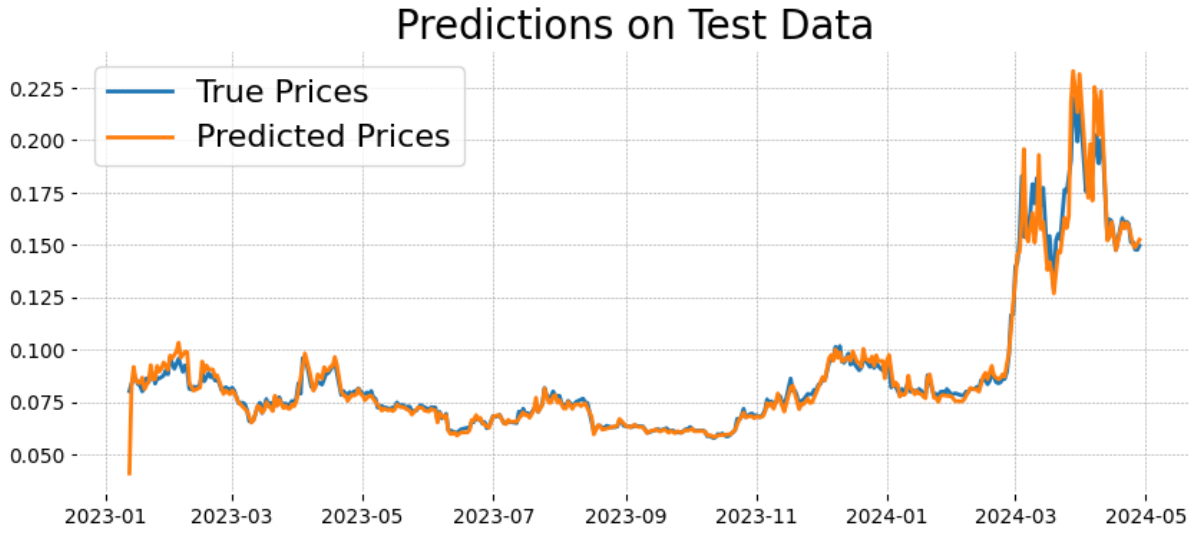

FIGURE 3.7 – Les prédictions sur les données de test de LightGBM

Dans la figure 3.7 on remarque que le modèle parvient légerment moins bien à capturer les fluctuations des prix réels, avec des prédictions moins

précises par rapport à XGBoost.

3.4.4 Les métriques d'évaluation

Nous évaluons les performances du modèle LightGBM en utilisant les métriques RMSE, MSE, et MAE, ainsi les prédictions des 5 derniers prix de clôture avec les prix réels.

	Données d'entraînement	Données de test
RMSE	0.007810971606172456	0.005476892603491752
MAE	0.0020601587626473974	0.0027044347400315234
MSE	6.1011277432432316e-05	2.999635259018266e-05

TABLE 3.4 – La performance du modèle LightGBM

	Prédictions	Réels
2358	0.152516	0.151354
2359	0.150137	0.151394
2360	0.148784	0.147837
2361	0.150316	0.147716
2362	0.152586	0.149782

TABLE 3.5 – Les 5 dernières prédictions et réels de LightGBM

Dans le tableau 3.5 on peut dire que les valeurs prédites et les valeurs réelles sont très proches, ce qui un bon signe de la performance du modèle LightGBM .

3.5 Application de l'algorithme CatBoost

Nous présentons les résultats de l'application du modéle CatBoost pour la prédiction des prix de cloture.

3.5.1 Importance des caractéristiques

Nous illustrons l'importance des différentes caractéristiques utilisées dans le modèle CatBoost. Cela nous permettra d'identifier les variables les plus influentes dans la prédiction .

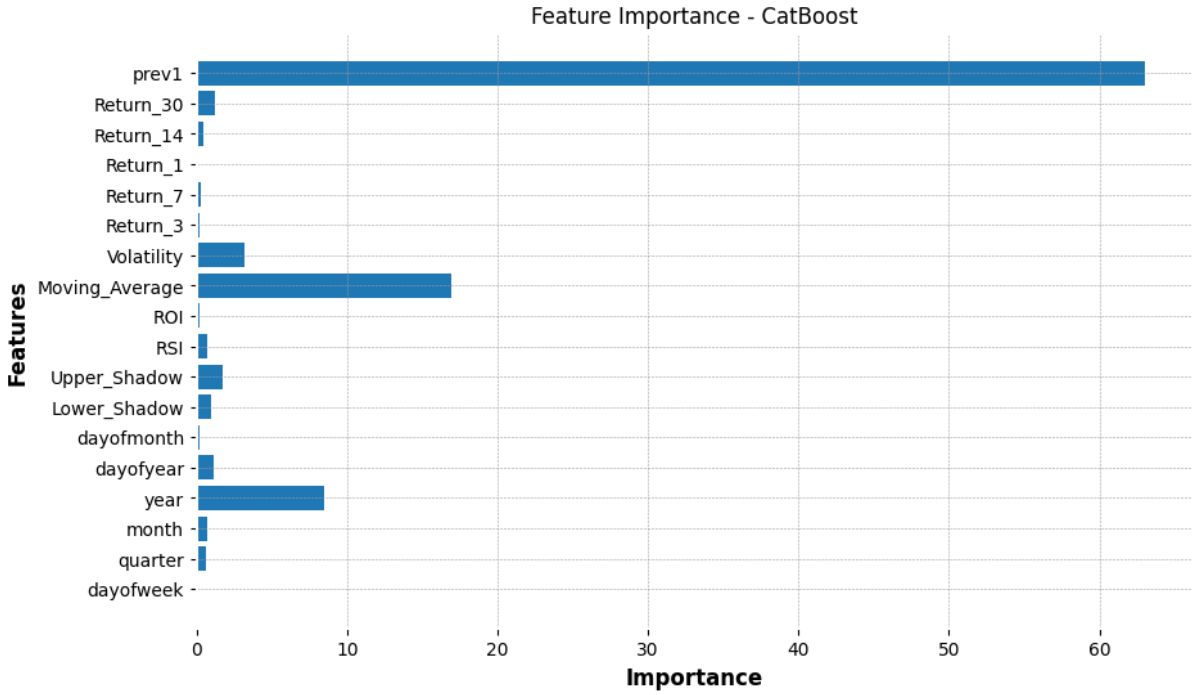

FIGURE 3.8 – L'importance des caractéristiques influentes dans CatBoost

Dans la figure 3.8 on remarque que l'analyse de l'importance des caractéristiques révèle que les variables **prev1, Moving-average et year** ont significativement influencé les prédictions du modèle, indiquant que ces facteurs sont cruciaux pour expliquer les variations observées dans les résultats prédits.

3.5.2 Données d'entraînement

Nous visualisons les prédictions du modèle CatBoost sur les données d'entraînement, comparées aux prix réels, afin de vérifier comment le modèle s'adapte aux données d'apprentissage.

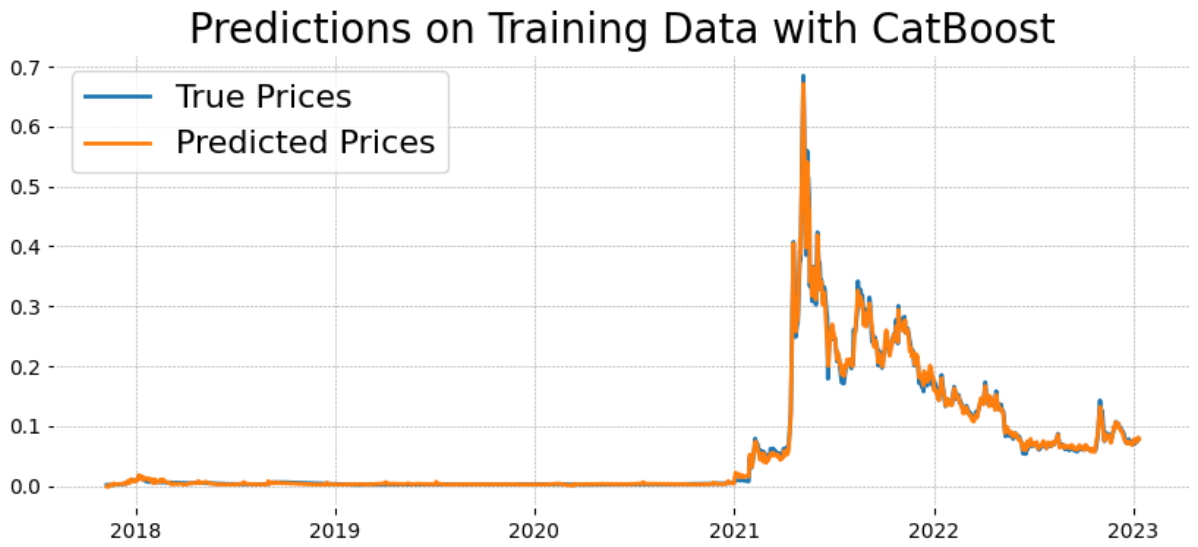

FIGURE 3.9 – Les prédictions sur les données d'entraînement de CatBoost

3.5.3 Données de test

Nous visualisons les prédictions sur les données de test, comparées aux valeurs réelles, pour évaluer la capacité du modèle CatBoost .

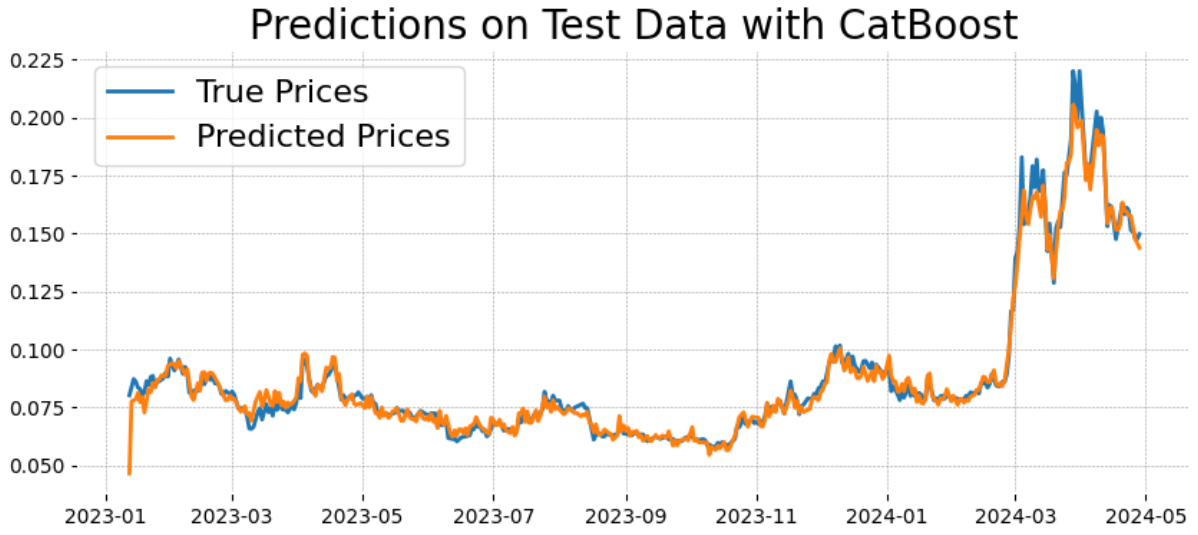

FIGURE 3.10 – Les prédictions sur les données de test de CatBoost

Dans la figure 3.10 on remarque que le modèle parvient à bien capturer les fluctuations des prix réels, avec des prédictions plus précises par rapport

à LightGBM.

3.5.4 Les métriques d'évaluation

Nous évaluons les performances du modèle CatBoost en utilisant les métriques RMSE, MSE, et MAE, ainsi les prédictions des 5 derniers prix de clôture avec les prix réels.

	Données d'entraînement	Données de test
RMSE	0.003939751317440907	0.003976554059226391
MAE	0.002193168462059232	0.002634098371160389
MSE	1.552164044327736e-05	1.581298218594989e-05

TABLE 3.6 – La performance du modèle CatBoost

	Prédictions	Réels
2358	0.157458	0.151354
2359	0.152116	0.151394
2360	0.147760	0.147837
2361	0.145136	0.147716
2362	0.143902	0.149782

TABLE 3.7 – Les 5 dernières prédictions et réels de CatBoost

Dans le tableau 3.7 on peut dire que les valeurs prédites et les valeurs réelles sont proches, ce qui un bon signe de la performance du modèle CatBoost .

3.6 Application des réseaux de neurones artificiels

Nous présentons les résultats de l'application du modéle ANN pour la prédiction des prix de cloture.

3.6.1 Importance des caractéristiques

Nous illustrons l'importance des différentes caractéristiques utilisées dans le modèle ANN. Cela nous permettra d'identifier les variables les plus influentes dans la prédiction .

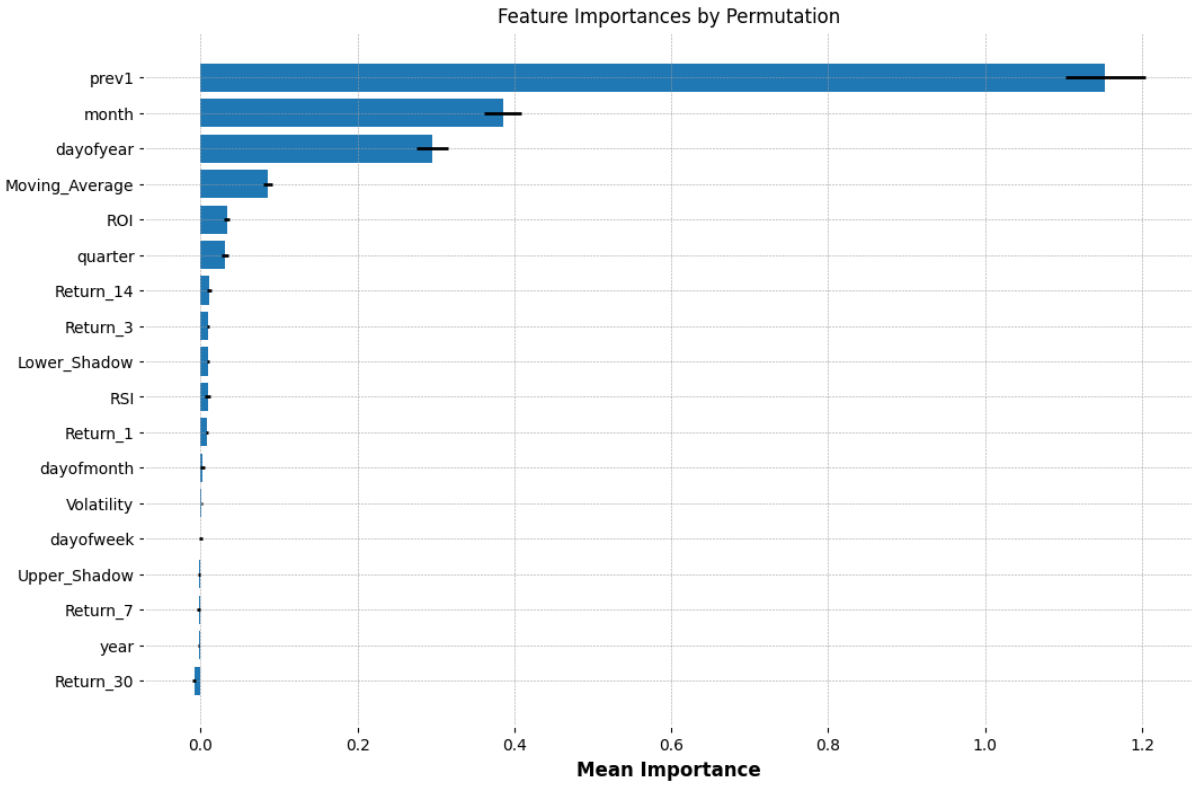

FIGURE 3.11 – L'importance des caractéristiques influentes dans ANN

Dans la figure 3.11 on remarque que l'analyse de l'importance des caractéristiques révèle que les variables **prev1, month et dayofyear** ont significativement influencé les prédictions du modèle, indiquant que ces facteurs sont cruciaux pour expliquer les variations observées dans les résultats prédits.

3.6.2 Données d'entraînement

Nous visualisons les prédictions du modèle ANN sur les données d'entraînement, comparées aux prix réels, afin de vérifier comment le modèle s'adapte aux données d'apprentissage.

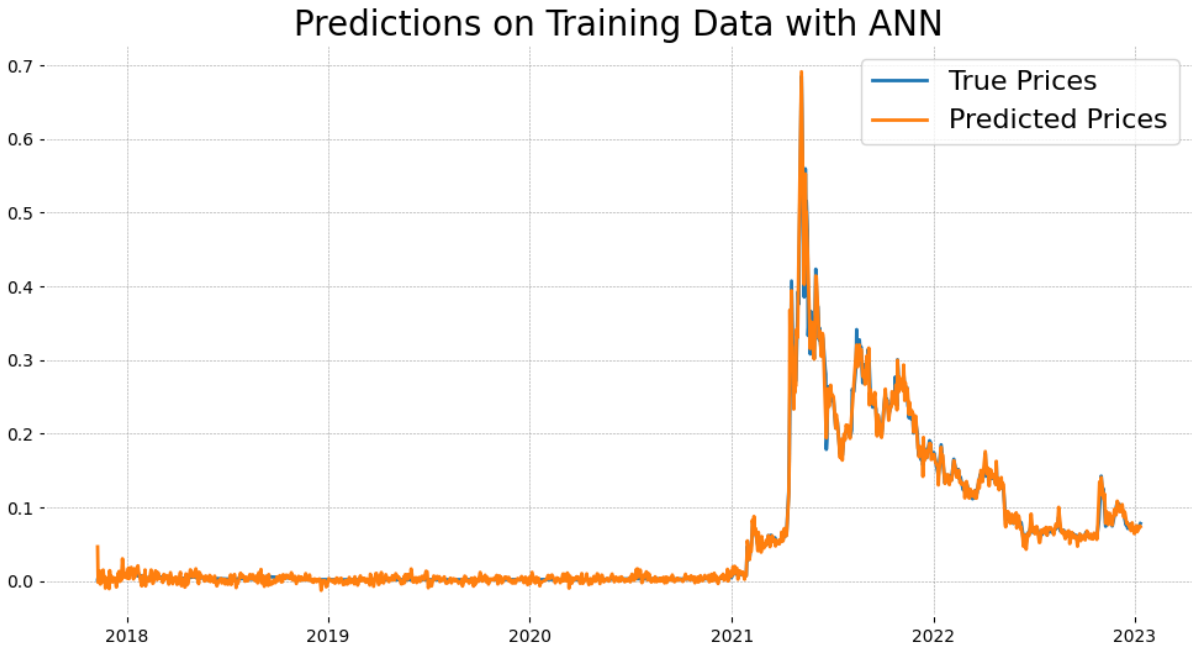

FIGURE 3.12 – Les prédictions sur les données d'entraînement de ANN

3.6.3 Données de test

Nous visualisons les prédictions sur les données de test, comparées aux valeurs réelles, pour évaluer la capacité du modèle ANN .

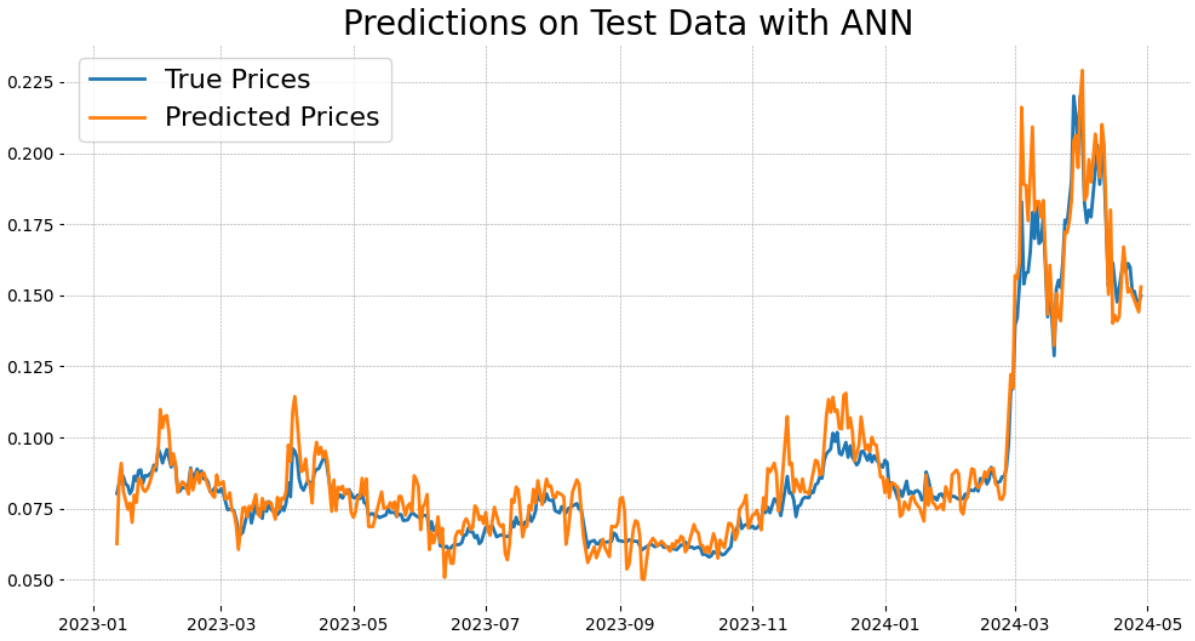

FIGURE 3.13 – Les prédictions sur les données de test de ANN

Dans la figure 3.13 on remarque que le modèle parvient moins bien à capturer les fluctuations des prix réels avec des prédictions par rappot aux algorithmes précédents.

3.6.4 Les métriques d'évaluation

Nous évaluons les performances du modèle ANN en utilisant les métriques RMSE, MSE, et MAE, ainsi les prédictions des 5 derniers prix de clôture avec les prix réels.

	les données d'entraînement	les données de test
RMSE	0.005142257557103528	0.007446566220581054
MAE	0.0037776875519755066	0.0055057554176022325
MSE	2.6442812783588346e-05	5.545134847749881e-05

TABLE 3.8 – La performance du modèle ANN

	Prédictions	Réels
2358	0.150181	0.151354
2359	0.147105	0.151394
2360	0.144635	0.147837
2361	0.144098	0.147716
2362	0.152912	0.149782

TABLE 3.9 – Les 5 dernières prédictions et réels ANN

Dans le tableau 3.9 on peut dire que les valeurs prédites et les valeurs réelles sont moins proches par rapport aux algorithmes précédents. Mais le modèle montre une bonne performance.

3.7 Comparaison entre les algorithmes

Dans cette section nous comparons les performances des différents modèles en termes de RMSE, MSE, et MAE des données de test pour déterminer lequel des modèles étudiés est le plus performant pour la prédiction des prix du Dogecoin.

Algorithme	RMSE	MAE	MSE
XGBoost	0.003159	0.001912	0.000010
LightGBM	0.005477	0.002704	0.000030
CatBoost	0.003977	0.002634	0.000016
ANN	0.007447	0.005506	0.000055

TABLE 3.10 – La comparaison des donneés de test entre les algorithmes

L'interprétation des résultats

Le modèle XGBoost est le meilleur en termes de RMSE, MAE et MSE. Cela suggère qu'il a la meilleure performance globale parmi les modèles comparés.

Le modèle CatBoost suit XGBoost en termes de RMSE MSE et MAE.

Le modèle LightGBM a des performances légèrement moins bonnes que XGBoost et CatBoost en termes de RMSE, MAE et MSE.

Le modèle ANN a les performances les plus faibles parmi les modèles comparés, avec le plus haut RMSE, MAE et MSE.

3.8 Conclusion

Ce chapitre a examiné et comparé les performances de quatre algorithmes largement utilisés dans la prédiction des prix du cryptomonnaies : XGBoost, LightGBM, CatBoost et les réseaux de neurones artificiels (ANN). Chaque algorithme a été évalué à travers des métriques d'évaluation telles que le RMSE, le MSE et le MAE, sur des ensembles de données de test spécifiques au marché du Dogecoin.

Les résultats de cette étude indiquent que les algorithmes de machine learning tels que XGBoost, LightGBM et CatBoost ont surpassé les réseaux de neurones artificiels (ANN) dans la prédiction des prix du Dogecoin. Cela souligne l'efficacité distinctive des approches basées sur XGBoost, LightGBM et CatBoost pour les applications de prédiction dans le domaine volatile des cryptomonnaies comme le Dogecoin.

Conclusion génerale

Ce mémoire a exploré la prédiction des prix du Dogecoin, une cryptomonnaie volatile et populaire, en utilisant une variété d'algorithmes d'apprentissage automatique et de deep learning. L'objectif était de déterminer quel modèle offrait la meilleure performance en termes de précision et de robustesse pour prédire les fluctuations de prix de cette cryptomonnaie.

Nos résultats ont révélé que les modèles de boosting, tels que XGBoost et CatBoost, surpassaient les réseaux neuronaux artificiels (ANN) en termes de précision et de robustesse. XGBoost et CatBoost ont démontré des performances supérieures avec des erreurs quadratiques moyennes (RMSE), des erreurs absolues moyennes (MAE) et des erreurs quadratiques moyennes (MSE) plus faibles. Ces résultats suggèrent que les modèles de boosting sont particulièrement bien adaptés à la prédiction de prix dans les marchés volatils des cryptomonnaies.

L'analyse de l'importance des caractéristiques a mis en évidence le rôle crucial de variables telles que la moyenne mobile, la volatilité et les rendements précédents dans la prédiction des prix du Dogecoin. Ces insights offrent une compréhension plus approfondie des facteurs qui influencent le plus la volatilité de cette cryptomonnaie.

Il est important de noter que les modèles de prédiction des prix des cryptomonnaies sont sujets à des limitations. La volatilité extrême du marché, la complexité des facteurs économiques et politiques, ainsi que la difficulté à prédire les événements futurs rendent la prédiction des prix des cryptomonnaies une tâche difficile.

Pour améliorer la précision des prédictions à l'avenir, des travaux sup-

plémentaires pourraient explorer les techniques suivantes :

- Intégration de données externes : L'intégration de données supplémentaires, comme les nouvelles des médias sociaux, les sentiments du marché, les données économiques et les événements majeurs, pourrait améliorer la capacité des modèles à capturer les influences externes sur les prix des cryptomonnaies.
- Modèles de séries temporelles plus avancés : L'utilisation de modèles de séries temporelles plus sophistiqués, tels que les réseaux neuronaux récurrents (RNN) ou les modèles autorégressifs intégrés à moyenne mobile (ARIMA), pourrait permettre de mieux capturer les tendances et les motifs temporels dans les données de prix.
- Techniques d'apprentissage par renforcement : L'exploration de techniques d'apprentissage par renforcement pourrait permettre de développer des modèles capables d'apprendre et de s'adapter aux changements du marché des cryptomonnaies en temps réel.

En conclusion, cette étude offre une contribution à la compréhension des méthodes de prédiction des prix des cryptomonnaies et met en lumière les avantages et les limitations des différentes approches. Les résultats suggèrent que les modèles de boosting, comme XGBoost et CatBoost, représentent des outils prometteurs pour la prédiction des prix du Dogecoin. Des travaux futures devraient explorer les approches mentionnées ci-dessus pour améliorer la précision et la robustesse des modèles de prédiction dans le contexte volatil et en constante évolution des cryptomonnaies.

Bibliographie

[1] C.Davide, M.J.Warrens, and G.Jurman. The coefficient of determination r-squared is more informative than smape, mae, mape, mse, and rmse in regression evaluation. *Journal of Machine Learning Research*, 22(1) :1–26, 2021. Disponible en ligne : https://www.jmlr.org/papers/v22/20-729.html.

[2] Coinmarketop. Guides d'investissement en cryptomonnaies. https://www.coinmarketop.com/, 2024.

[3] D.E.Rumelhart, G.Hinton, and R.J.Williams. Learning representations by back-propagating errors. *Nature*, 323(6088) :533–536, 1986.

[4] Dogecoin. History of dogecoin. https://dogecoin.com/dogepedia/articles/history-of-dogecoin/, 2024.

[5] G.Ke, Q.Meng, T.Finley, T.Wang, W.Chen, W.Ma, Q.Ye, and T.Liu. Lightgbm : A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, pages 3146–3154, 2017.

[6] J.H.Friedman. Greedy function approximation : A gradient boosting machine. *Annals of Statistics*, 29(5) :1189–1232, 2001.

[7] J.Murphy. *Technical Analysis of the Financial Markets : A Comprehensive Guide to Trading Methods and Applications*. New York Institute of Finance, 1999.

[8] L.Prokhorenkova, G.Gusev, A.Vorobev, A.V.Dorogush, and A.Gulin. Catboost : unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems*, pages 6639–6649, 2018.

[9] L.De Matteis, S.Janny, S.Nathan, and W.Shu-Quartier. Introduction à l'apprentissage automatique. 2022.

[10] R.J.Hyndman and G.Athanasopoulos. *Forecasting : Principles and Practice*. OTexts, 2018. Disponible en ligne : https://otexts.com/fpp3/.

[11] T.Chen and C.Guestrin. Xgboost : A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794. ACM, 2016.

[12] T.Hastie, R.Tibshirani, and J.Friedman. *The Elements of Statistical Learning : Data Mining, Inference, and Prediction*. Springer, 2 edition, 2009.

[13] Y.Freund and R.E.Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. Technical report, AT&T Labs, 180 Park Avenue, Florham Park, New Jersey 07932, December 1996.

[14] Y.LeCun, Y.Bengio, and G.Hinton. Deep learning. *Nature*, 521(7553) :436–444, 2015.

Résumé :
Ce mémoire explore les approches de prédiction des prix du Dogecoin en utilisant des techniques de l'apprentissage automatique et de l'apprentissage profond. Les cryptomonnaies, notamment le Dogecoin, sont devenues populaires en raison de leur nature décentralisée et de leur potentiel de rendement élevé, malgré une volatilité marquée. L'étude évalue plusieurs algorithmes, tels que XGBoost, LightGBM, CatBoost et les réseaux de neurones artificiels (ANN), afin de déterminer lequel offre les meilleures performances en termes de précision et de robustesse. Les résultats permettent de mieux comprendre les forces et les faiblesses de chaque modèle et fournissent des recommandations pour une utilisation pratique dans la prédiction des prix des cryptomonnaies.

Mots clés : Apprentissage automatique, Apprentissage profond, Prédiction des prix, Cryptomonnaies, XGBoost, LightGBM, CatBoost, Réseaux de neurones artificiels, Dogecoin, Méthodes d'ensemble,

Abstract :
This thesis investigates methods for predicting Dogecoin prices using machine learning and deep learning techniques. Cryptocurrencies, particularly Dogecoin, have risen in popularity due to their decentralized nature and potential for high returns, despite significant volatility. The study evaluates several algorithms, including XGBoost, LightGBM, CatBoost, and artificial neural networks (ANN), to determine which offers the best performance in terms of accuracy and robustness. The findings provide insights into the strengths and weaknesses of each model and offer practical recommendations for their application in cryptocurrency price prediction.

Keywords : Machine learning, Deep learning, Price prediction, Cryptocurrencies, XGBoost, LightGBM, CatBoost, Artificial neural networks, Dogecoin, Ensemble methods,