Autour de quelques techniques de prédiction du cours d'un cryptoactif

Aurélien Grenard - Yannis Oulefki

Résumé

Aujourd'hui, le marché des cryptomonnaies est plus que jamais en expansion. Alors que le bitcoin est présenté comme l'alternative principale aux monnaies fiduciaires, des dizaines de cryptomonnaies naissent (et disparaissent) chaque jour sur les plateformes de négociation dédiées à ces actifs. Simple monnaie créée par un informaticien en herbe ou bien projet faramineux voulant mettre la blockchain au service de la société, chaque cryptomonnaie est liée à un projet plus ou moins influant sur l'économie. Cependant, depuis que les banques (et les banques centrales) et autres grandes institutions se sont appropriées ces produits, nous commençons sérieusement à envisager ces cryptoactifs comme de produits financiers réels, pouvant avoir un impact considérable sur les stratégies financières de certains acteurs économique. Contrats futures, options et autres produits dérivés portant des cryptomonnaies comme sous-jacents apparaissent de plus en plus sur les marchés, d'où la nécessité pour les acteurs de bien comprendre le fonctionnement de ces nouveaux actifs. Dans notre article, nous allons mettre en avant certains outils qui peuvent se révéler utiles pour le trading sur ces cryptomonnaies. Notre projet consiste à concevoir des modèles de prédiction de la variation future des cours de cryptomonnaies ainsi que la mise en place de certains principes de trading en exploitant les résultats de ces modèles. Le machin learning offre de nombreuses solutions notamment pour la prédiction des prix. Bien qu'étant l'outil principalement utilisé dans nos travaux, nous allons d'abord mettre en avant les technologies mise en place par les plateformes de trading qui nous ont permit d'accéder aux marchés, mais aussi, via leurs API, de récolter continument des données de marchés et de les exploiter. Tous les résultats statistique énoncés dans cet article sont issue de données que nous avons récolté et stocké dans notre base de donnée privé et qui sont donc pas trouvable sur Internet. Elles pourront néanmoins être partagées au lecteur à sa demande, s'il souhaite reproduire ou vérifier les calcules.

Table des matières

1 Récupération des données **2**
- 1.1 Plateformes et API . . . 2
- 1.2 Récupération des données . . . 2
- 1.3 Certains problèmes techniques . . . 3

2 Modèles de prédiction **5**
- 2.1 Choix de la structure des données . . . 5
- 2.2 Une première approche classique : la moyenne mobile . . . 5
- 2.3 Prédiction par régression linéaire . . . 6
  - 2.3.1 Un exemple jouet : régression sur les données sans décalage . . . 6
  - 2.3.2 Introduction du décalage . . . 7
- 2.4 Forêt aléatoire . . . 8
- 2.5 Prédiction par XGBoost . . . 8
- 2.6 Comparaisons des modèles . . . 9

3 Stratégie de trading algorithmique **9**
- 3.1 Une première stratégie : un monde parfait . . . 9
- 3.2 Slippage et frais de transaction . . . 10

4 Conclusion **10**

1 Récupération des données

1.1 Plateformes et API

L'engouement actuel pour la technologie de la cryptomonnaie et de la blockchain a poussé les développeurs à créer des interfaces d'application programmées robustes pour intégrer les logiciels d'entreprise existants aux nouveaux systèmes cryptographiques. En 2018, il existe plusieurs plateformes de change virtuelles qui fournissent un accès API aux développeurs. Celles-ci sont importantes dans la génération de portefeuilles de monnaie virtuelle et d'améliorer les transactions. Définissons d'abord clairement ce qu'est une API.

Définition 1. *Une interface de programmation d'application ou API est un ensemble normalisé de classes, de méthodes, de fonctions et de constantes qui sert de façade par laquelle un logiciel offre des services à d'autres logiciels.*

Dans le cadre de notre projet, nous serons plus sensibles aux API offrant d'une part un accès constant, libre et rapide aux marchés pour optimiser la partie, et d'autre part un large panel de solutions pour l'accès aux données présentes et passées. Le deuxième point est crucial si l'on veut donner un sens à notre travail.

Les principaux atouts que nous cherchons dans une API sont : la mise à disposition d'un bon connecteur qui permet de maintenir une connexion continue et stable entre notre machine et la plateforme, des classe et des méthodes qui permettent le passage d'ordres (market et limite) et leur exécution rapide et enfin des classes et méthode permettant de se procurer des informations de marchés en temps réel ou en temps décalé, avec un certain degré de liberté concernant leurs précisions, leurs quantités et leurs formats ainsi que d'autre critères plus détaillés. Les données sur Internet sont présentes sous certains formats limités, à des intervalles de temps constants et parfois avec peu de précisions. C'est pour se défaire de ces contraintes que nous avons choisi d'opter pour la plateforme qui nous donnait le plus de liberté à l'accès aux données de marché afin de les récupérer et de les façonner à notre besoin. Nous avons sélectionné les 5 meilleures plateformes dont voici une description rapide :

Table 1 – Comparaison des différentes plateformes de trading sur cryptomonnaies

Plateforme	Volume/jour (euro)	API	Frais de transaction	Paire cryptomonnaies
$Binance$	$900M$	Oui	0.10	603
$Bitfinex$	$127M$	Oui	0.20	401
$Kraken$	$98M$	Oui	0.26	75
$Bittrex$	$20M$	Oui	0.25	346
$Poloniex$	$11M$	Oui	0.25	124
Cex	$2,2M$	Oui	0.25	44

On remarque que la plateforme Binance sort du lot avec ses faibles frais de transaction, sa richesse en terme de diversité de paire de cyptomonaies négociable, mais aussi par un volume de transactions par jour bien plus élevé que ses concurrents. Ce dernier point nous intéresse particulièrement. En effet, il est synonyme de grande liquidité sur cette plateforme et donc de plus de données par jour. De plus, l'API de Binance est de loin celle qui permet le plus de liberté en terme de trading et autres manipulations cité plus haut. Nous avons choisi de traiter sur celle-ci dans le cadre de notre projet.

1.2 Récupération des données

Détaillons à présent la démarche de récupération des données. Commençons cette section par une remarque importante :

Remarque 1. *Les transactions d'achat et de vente ont lieu entre deux ou plusieurs acteurs du marché et pour que nous réalisions un profit, quelqu'un d'autre doit faire une perte. Qu'est-ce qui nous rend meilleurs que tous les autres commerçants qui tentent également d'être rentables ? Dans le commerce, un avantage concurrentiel est appelé un avantage et peut provenir de divers endroits :*

Latence : nous avons une connexion à l'échange plus rapide que les autres. Cela signifie que nous pouvons observer les nouvelles données plus rapidement et soumettre des commandes avant les autres. Sur les marchés financiers, les institutions dépensent des millions de dollars pour minimiser la latence des échanges.

Infrastructure : Notre infrastructure peut être plus tolérante aux pannes, plus performante ou gérer les meilleurs cas que les concurrents.

Données : Nous pouvons avoir de meilleures données que d'autres, où mieux peut signifier beaucoup de choses. Nos données peuvent être collectées de manière plus fiable avec moins de pannes, provenir d'une autre API ou nettoyées et post-traitées plus soigneusement. La reconstruction du carnet de commandes à partir de données API brutes en temps réel peut être un processus sujet aux erreurs en raison de données bruyantes, retardées ou en double.

Modèle : Nous pourrons peut-être construire un meilleur modèle prédictif basé sur des modèles dans les données. Peut-être que nous utilisons de nouvelles techniques de Deep Learning sophistiquées, avons une meilleure fonction d'optimisation (plus d'informations ci-dessous), de meilleures fonctionnalités ou un algorithme d'entraînement différent.

Accès au marché : Nous pouvons avoir accès à un marché que tout le monde ne peut pas échanger. Par exemple, certains échanges en Corée du Sud exigent que vous soyez citoyen pour y accéder. De nombreux échanges internationaux n'acceptent pas les citoyens américains comme clients pour éviter de traiter avec l'IRS.

N'étant pas des professionnels du trading, nous partons d'avance avec un handicap de taille concernant la latence et l'infrastructure. L'idée de récupérer nos propres données et de les exploiter le plus possible avec nos outils probabilistes est motivée par le fait de vouloir compenser ce déséquilibre face aux autres acteurs.

Notre objectif est de récupérer toutes les information de chaque transaction sous le format :

$$I_t = (p_t, v_t, ask/bid)$$

où p est le prix, v le volume et $askorbid$ signifiant si le market maker est vendeur ou acheteur lors de la transaction. Nous allons utiliser pour cela la classe $MarketDataEndpoint$ qui contient les fonctions nécessaires pour récupérer les dernières informations de marché. La première étape de notre processus est d'établir une connexion constante et fiable entre notre serveur et la plateforme Binance. C'est primordiale si nous voulons une suite continue de données. Notons que le marché des cryptomonnaies sont extrêmement volatiles (on peut observer assez fréquemment des variations quotidiennes de l'ordre de centaine de pourcent !). Ainsi, il est assez important d'avoir les moyens de récupérer absolument toutes les transactions afin d'obtenir une base de données propre pour nos modèles. Il arrive néanmoins que des problèmes de connexions surviennent et qu'on n'enregistre pas les données durant un intervalle de temps. Nous discuterons des solutions de ce problème dans la section suivante. Afin d'assurer cette connexion, nous utilisons la classe $BinanceSocketManagerr$ de Binance qui fournit les méthodes nécessaires à la mise ne place de la connexion sécurisée à la plateforme, la vérification de l'état de connexion (avec des pings) mais aussi à la résolution automatique de certains problèmes de connexions.

1.3 Certains problèmes techniques

Nous faisons ici une liste non-exhaustive des principaux problèmes que nous avons rencontrés lors de la mise en place de l'algorithme de récupération des données.

Problème 1. *Les coupures de connexions entre nos machines et le serveur Binance sont de loin le problème le plus rencontré lors de la mise en place de l'algorithme. En réalité, bien que le problème tout le temps le même, il soulevait à chaque fois de nouveaux problèmes. Par exemple, il est arrivé que la connexion ai été coupée par notre opérateur internet à cause de l'excès de requêtes envoyé au serveur Binance. Il a fallu contourner alors ce problème (d'où l'utiilsation d'un websocket qui assure un lien continue d'information sans envoyer de requêtes).*

Problème 2. *Les données manquantes lors des coupures de connexion sont le plus important à gérer (car le but de toute cette manipulation est d'avoir une base de données parfaite sans trous). Nous avons réussi à contourner ce problème en déclenchant automatiquement un algorithme sur un serveur parallèle indépendant de notre connexion qui récupère les données le temps de la déconnexion et qui les envoie ensuite à notre base de donnée. (Que se passe-t-il si les deux machines étaient privés de connexion ? Ce*

n'est pour le moment jamais arrivé car l'algorithme de secours est sur un serveur sécurisé, mais c'est un cas à envisager)

Problème 3. *Le plantage dû à la masse importante de données récoltées. Nous recevons en continue les informations de chaque trade qui à lieu sur le marché, et cela, pour chacune des 600 paires de cryptomonnaies, la plupart d'entre elles enregistrant des dizaines de transactions par secondes. Cela fait une grande quantité d'information à traiter, surtout lorsqu'on ne possède pas le matériel adéquat. Nous ne pouvions pas stocker l'information dans notre base de donnée à chaque fois que nous la recevons. Nous avons donc créé un pont de données qui permet de stocker les données une fois un certain quota d'information reçu. Cela réduit considérablement le nombre d'opérations et le nombre de plantages de l'algorithme.*

Problème 4. *Il est arrivé aussi que la connexion se coupait au moment où stockage devait avoir lieu ce qui faisait perdre des milliers de données d'un coup. Nous avons donc décider a chaque fois qu'il y avait une coupure de connexion de récupérer les données des 5 dernières minutes (ce qui est suffisant pour couvrir une perte maximale de données en attente d'être stockées) en plus des données en direct.*

La liste est encore très longue, mais ces quelques points donnent une idée des problématiques que nous avons rencontrées et du genre de solutions que nous avons imaginées et mises en œuvre pour les contourner. Nous obtenons enfin un algorithme stable qui récupère toutes les données et les stocks sous le format montré dans la figure 1.

tradetime	isbuyermm	price	qty
2019-03-01 00:00:00.789	vendeur	3819.52	0.018366
2019-03-01 00:00:00.897	vendeur	3819.52	0.039891
2019-03-01 00:00:01.739	vendeur	3819.52	0.004991
2019-03-01 00:00:02.178	vendeur	3819.50	0.032655
2019-03-01 00:00:02.947	acheteur	3819.28	0.055265
...	...	...	...
2019-03-01 23:59:56.642	vendeur	3829.49	0.002972
2019-03-01 23:59:57.448	vendeur	3829.49	0.033193
2019-03-01 23:59:59.518	acheteur	3829.32	0.011303
2019-03-01 23:59:59.733	vendeur	3829.49	0.047896
2019-03-01 23:59:59.809	acheteur	3829.32	0.008950

FIGURE 1 – Données récupérées pour la paire BTCUSDT le 01.03.2019

Ce degrés de précision nous permet d'avoir le choix de l'échelle de temps à laquelle nous souhaitons effectuer nos prédictions. En effet, la figure 2 nous montre que nous pouvons remodeler nos données en fonction du temps choisie tout en gardant l'information à sont échelle la plus fine/

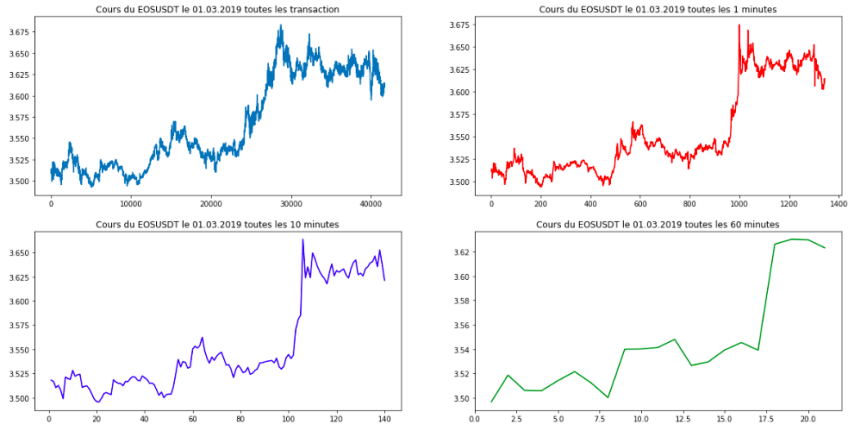

FIGURE 2 – Différents courbes pour le pour la paire EOSUSDT le 01.03.2019

2 Modèles de prédiction

Tous les tests présentés ci dessous ont été effectués sur la paire de cryptomonnaie $BNBUSDT$ sur une période de 15 jours entre le 1er mars 2019 et le 15 mars 2019.

2.1 Choix de la structure des données

Nous avons une base de données riche et complète. Cependant, nous devons donner un sens à nos données, c'est-à-dire les organiser en fonction de nos besoins de prédiction. Chaque ligne de notre base de données nous donne les informations d'une seule transaction. Ce format peut être utile si l'on veut prédire les caractéristiques de la prochaine transaction qui aura lieu dans le marché. Notre objectif est légèrement différent. Plaçons-nous dans un intervalle de temps $[0, T]$ et considérons une subdivision $t_0 < t_1 < ... < t_N$ de pas égale à δ. Notons par V_k la variation du prix p_t entre $[t_{k-1}, t_k]$. Nous voulons alors prédire V_{k+1} en fonction de l'information disponible avant t_k. Le paramètre δ peut être vu comme un hyperparamètre à optimiser. Le choix est assez libre en fonction du modèle utilisé et des objectif qu'on se fixe. Cependant, nous allons nous baser sur une idée assez intuitive pour restreindre le choix du pas de temps. L'idée est que le mouvement du prix d'un cryptoactif dans une heure, un jour ou un mois est fortement aléatoire et dépendant de facteurs qui ne sont pas observables dans nos données (actualité politique, déclaration sur tweeter de certains gouvernements, apparition d'une nouvelle technologie ...). Par contre, la variation du prix les prochaines secondes est beaucoup moins dépendante de ce type de paramètre et on peut considérer qu'elle est en grande partie liée aux données de marchés comme les derniers prix de transaction, les derniers volumes traités ou encore le nombre d'achats ou de ventes durant les dernières secondes. Nous allons donc nous concentrer sur les δ variant entre $30s$ et $300s$. À présent, comme nous voulons observer le comportement du marcher intervalle de temps par intervalle de temps et non transaction par transaction, nous pouvons nous servir de nos données pour extraire de nouvelle informations concernant le marché comme par exemple le nombre d'achats et le nombre de ventes qu'il y a eu entre chaque pas de temps, les volumes traité et leurs variations et tous les autres informations qui pourraient devenir des paramètre pour notre modèle(ou $features$). La fonction $WIdowsData$ dans notre $ToolBox$ nous permet de remanier notre base de données et d'obtenir un nouveau format de données adapté à notre besoin en fonction du pas de temps choisi

2.2 Une première approche classique : la moyenne mobile

La moyenne mobile est un indicateur très utilisé en trading pour avoir une idée de la tendance future d'un cours. Commençons par déterminer la fenêtre de temps à observer pour avoir une prédiction optimale. Pour ce faire, si N est la taille de la fenêtre, nous cherchons

$$argmin_N(\left\|y_{test}^N - y_{model}^N\right\|^2)$$

Nous obtenons la courbe d'erreur ci dessous.

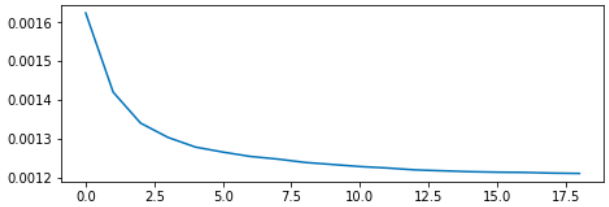

FIGURE 3 – Erreur quadratique en fonction de N

Ainsi, on remarque que le modèle arête de gagner significativement en précision à partir de $N = 7$. Nous allons alors choisir cette fenêtre pour l'entraînement du modèle. Voici les résultat obtenus

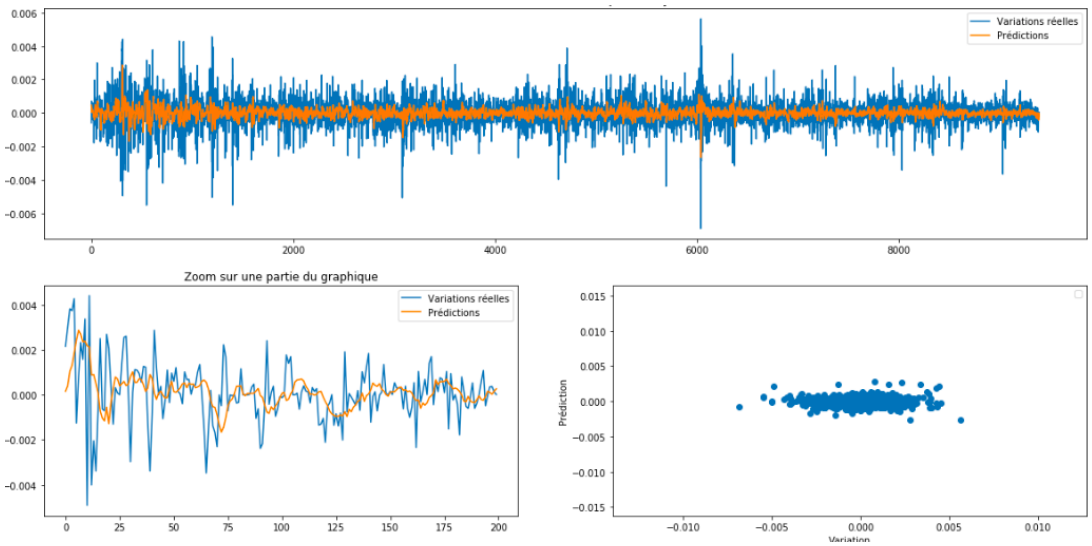

FIGURE 4 – Analyse du modèle Moyenne Mobile

Le premier graphique nous sert à avoir une vision globale de la différence entre la prédiction et la réalité (différence d'échelle, décalage ...).(différence d'échelle, décalage ...) La dernière est une représentation des prédictions en fonction des variation, une prédiction parfaite est censé donner un ensemble de point formant une courbe affine. On remarque avant tout sur les courbes que le modèle ne capte pas du tout les intensités de variations. Cela peut s'expliquer facilement, car le modèle étant basé sur des moyennes, les variations sont rarement extrêmes et nous obtenons une courbe beaucoup plus plate que la réalité. De plus, on remarque un retard dans les changements de tendance ce qui est aussi expliqué par le fait de prendre en compte uniquement la moyenne des données passées. Le modèle est assez peu satisfaisant d'un point de vu de la prédiction, mais il va s'avérer surprenant quand il s'agira de mettre en place des stratégies de trading comme on le verra dans la section 3.

2.3 Prédiction par régression linéaire

2.3.1 Un exemple jouet : régression sur les données sans décalage

Nous commençons par faire une régression directement sur les données que nous avons remaniées en prenant comme paramètres toutes les colonnes du tableau de la figure 3 à part la dernière qui est la variable à prédire.

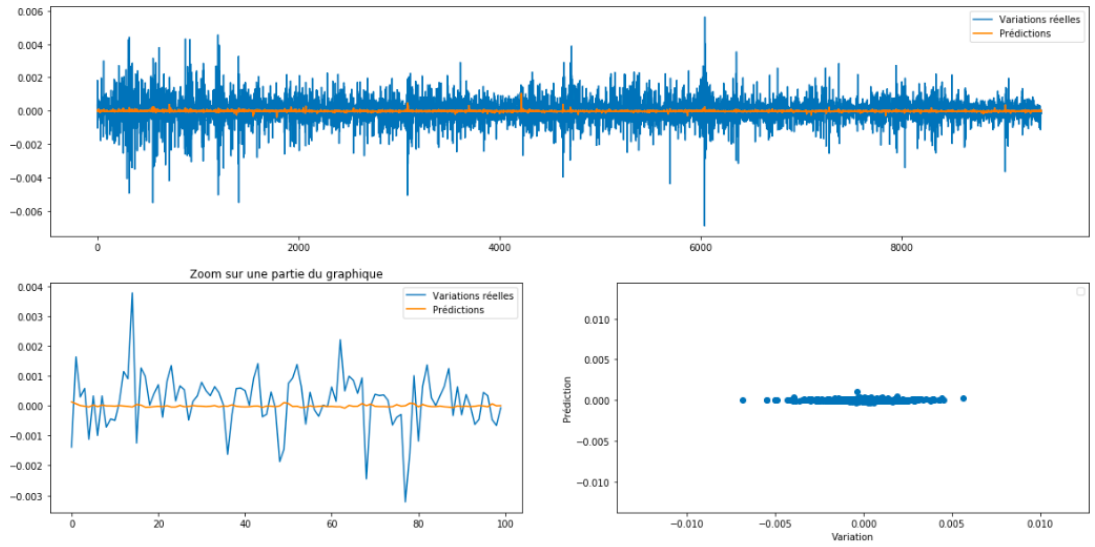

FIGURE 5 – Analyse du modèle Régression linéaire sans lag

Cet exemple nous montre l'intérêt de choisir les bons paramètres pour la prédiction. En effet, nous avons un résultat moins bon que la moyenne mobile alors que théoriquement il devrait le surpasser en terme d'erreur.

2.3.2 Introduction du décalage

Un des facteurs essentiel à prendre en compte dans nos données est qu'elles sont en réalité un ensemble de séries temporelles. Une bonne pratique alors est de calculer la corrélation entre les données passés et présentes. Celle ci s'avère non nulle pour la plus part des cas y compris le notre. Nous allons considérer alors les trois séries temporelle *Pricevariation*, *Orderbuy*, et *Ordersell* que nous allons affecter de leurs matrices décalées afin de pouvoir entrainer le modèle dessus. Par exemple pour un décalage de taille N (qui signifie que pour prédire une variation, nous allons regarder $N \cdot \delta$ secondes dans le passé) nous obtenons un ensemble de données de la forme

	Price variation	Buy orders	Sell order	Price variation n-1	Buy orders n-1	Sell order n-1	Price variation n-2	Buy orders n-2	Sell order n-2	Future price variation
0	0.018080	-0.514082	-0.500692	0.026152	0.352607	0.436782	0.028664	-0.293797	-0.276843	-0.024526
1	-0.024526	0.306435	0.239147	0.018080	-0.514082	-0.500692	0.026152	0.352607	0.436782	0.012073
2	0.012073	-0.413984	-0.180612	-0.024526	0.306435	0.239147	0.018080	-0.514082	-0.500692	0.000595
3	0.000595	-0.666343	-0.639692	0.012073	-0.413984	-0.180612	-0.024526	0.306435	0.239147	0.008795
4	0.008795	-0.247273	0.030130	0.000595	-0.666343	-0.639692	0.012073	-0.413984	-0.180612	0.029145

FIGURE 6 – Ensemble de données avec décalage $N = 2$

Comme pour la moyenne mobile, nous commençons d'abord par séléctionner le N optimal pour notre modèle grâce à la courbe d'erreur ci dessous

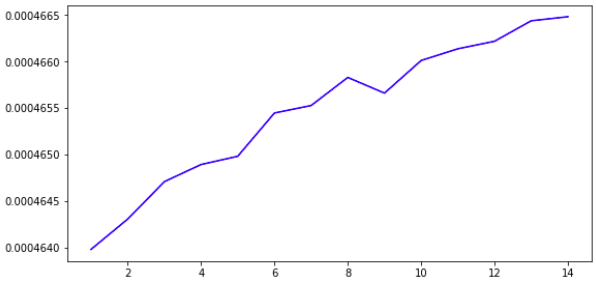

FIGURE 7 – Erreur quadratique en fonction de N

On obtient que le N optimal est le même que pour le modèle à moyenne mobile , c'est à dire égale à 1. Alors on effectue l'entraînement du modèle et les prédiction et on obtient :

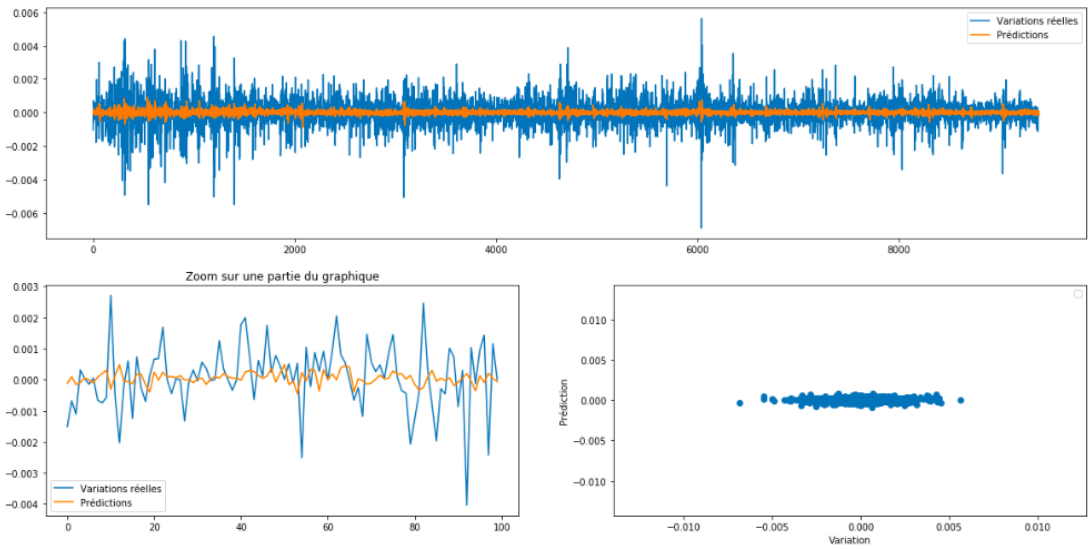

FIGURE 8 – Analyse du modèle Régression linéaire avec lag $N = 1$

On remarque une légère amélioration dans la prédiction, les mouvements de la courbe sont moins plats et ressemble un peu plus à la réalité. On voit aussi que le modèle arrive à capter et prédire de gros mouvements. Cependant, on voit qu'il à du mal à prédire le bon sens de variation.

2.4 Forêt aléatoire

Dans tout ce qui suit, nous considérons le même jeu de données que la dernière sous partie, c'est à dire avec décalage. Nous nous intéressons ici au modèle des forêts aléatoires qui selon notre expérience, apporte souvent des solutions lorsque la régression linéaire semble impuissante. Nous entrainons le modèle sans optimisation des hyperparamétres ici. Nous obtenons

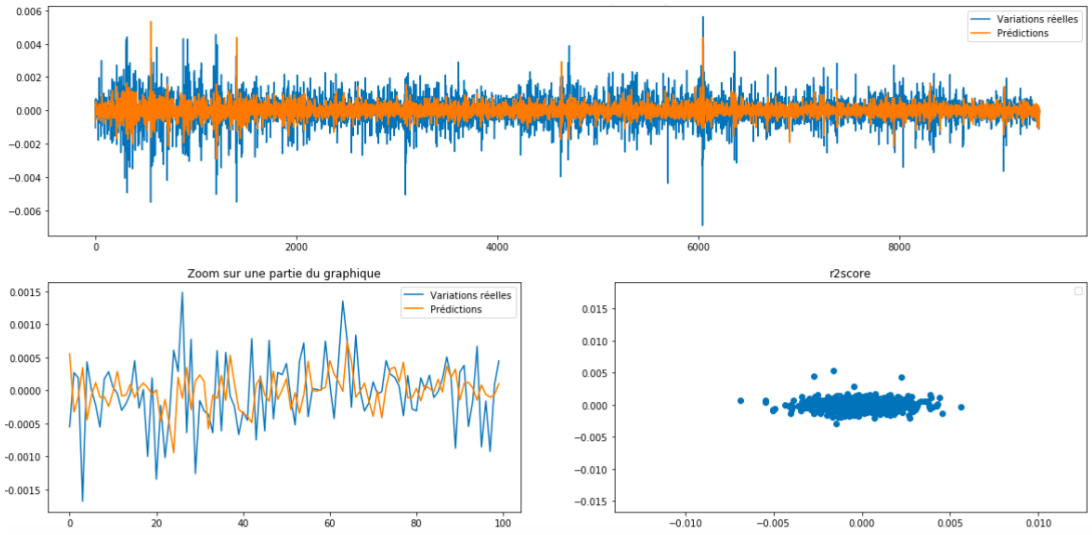

FIGURE 9 – Analyse du modèle des Forêts aléatoires avec lag $N = 1$

On observe clairement une amélioration dans l'intensité des prédictions, et dans la capacité du modèle à capter les signe des variations. Il serait alors intéressant de se pencher sur ce modèle afin d'en optimiser les hyperparamétres. Nous allons passer cette étape et allons passer au dernier modèle étudié dans ce rapport qui lui sera optimisé sur la grille de ses hyperparamétres.

2.5 Prédiction par XGBoost

Pour finir cette section, nous allons utiliser le modèle XGBoost, qui est le modèle star du moment. Nous avons entraîné puis optimisé ce modèle comme il est décrit dans le fichier *.pyng*. Nous obtenons alors

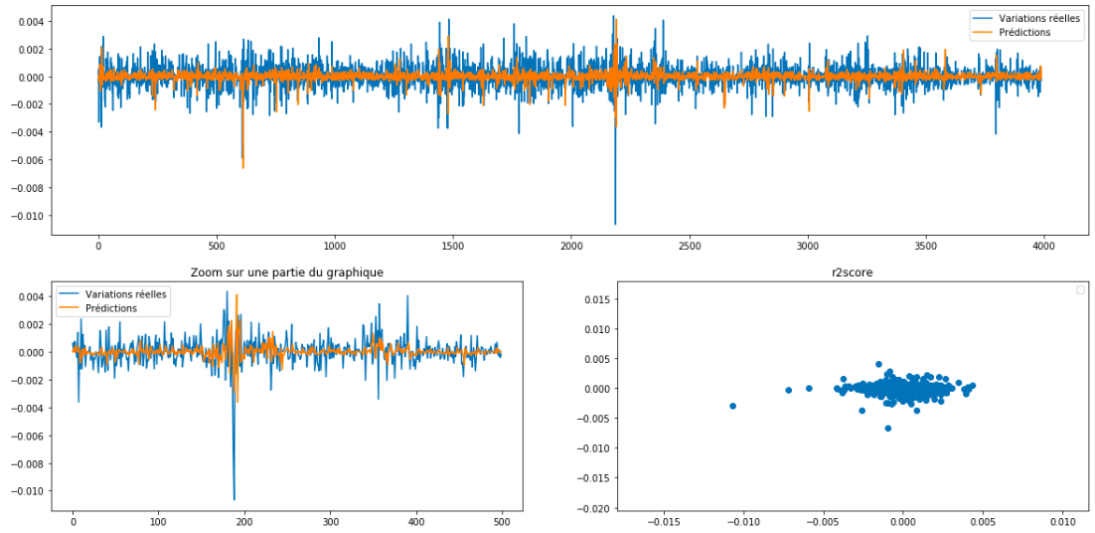

FIGURE 10 – Analyse du modèle XGBoost

On observe alors une nette amélioration des prédictions et le modèle semble mieux capter les tendances et les intensités des variations que les précédents modèles.

2.6 Comparaisons des modèles

Nous pouvons comparer la qualité de prédiction des modèles ci dessus en regardant leurs erreurs quadratiques respectives.

	RMSE	% succes signe predicted
Moving average	0.000750797	59.7358
Linear regresion without lag	0.000707208	47.7382
Linear regresion with lag	0.000708513	48.2006
Random forest	0.000724156	48.4987
XGBoost	0.000645192	47.4518

FIGURE 11 – Comparaison des modèles de prédiction

On observe que c'est bien (comme on l'attendait) le modèle XGBoost qui à l'erreur la plus faible donc qui en moyenne prédit le mieux les variations du cours de la cryptomonnaie. La moyenne mobile offre quant à elle la moins bonne prédiction. Néanmoins, on constate que ce simple modèle est le meilleur quand il s'agit de déterminer non pas la variation exacte, mais uniquement le signe avec près de 6 sucés sur 10 contre environ 5/10 pour les autres modèles.

Remarque 2. *Notons qu'un succès de 5/10 dans la prédiction du signe de variation est aussi bonne qu'un pile ou face, car le résultat est binaire (hausse ou baisse) et qu'a priori sans aucune information, les deux événements sont équiprobables. Cependant, ce ratio n'est qu'une indication primaire et ne vaut pas l'analyse détaillée des prédictions. Par exemple, pour un trader, il serait plus sûr d'avoir un modèle qui se trompe plus en prédisant la baisses qu'en prédisant la hausse du cours : cela lui ferait perdre certaines opportunités, mais lui assurera plus de gain en moyenne à l'achat. En générale, pour diminuer le risque de modèle, il vaut mieux un modèle qui surestime les baisses et sous-estime les hausses. Mais cela ne s'applique pas pour trader désireux au contraire de prendre beaucoup de risques.*

3 Stratégie de trading algorithmique

3.1 Une première stratégie : un monde parfait

Pour commencer, nous allons considérer un marché parfait, sans friction, sans coût de transaction où les ordres sont passés instantanément. Nous allons alors commencer par une la plus simple stratégie qui soit : acheter une unité si notre modèle prédit une hausse puis liquider la position à l'issue de la période de temps où la prédiction est valable. Pour chaque modèle, le PnL de la stratégie est donné dans la figure suivante

	PnL	Mean	Variance	Max win	Max loss	Positives transactions
Moving average	1.2183	0.000255035	0.000106286	0.0647	-0.0829	54.8671
Linear regresion without lag	0.5111	0.000215019	0.000167265	0.0821	-0.101	53.1763
Linear regresion with lag	-4.2904	-0.000775841	0.000106394	0.0821	-0.101	48.8969
Random forest	0.8427	0.000179145	0.000114835	0.0821	-0.101	53.4014
XGBoost	2.9946	0.00071402	9.53403e-05	0.0821	-0.101	54.6733

FIGURE 12 – Comparaison des PnL

Nous avons le PnL du modèle XGBoost qui est largement en tête suivi par le moyenne mobile et le random forest. Les régressions linéaires sont en bas du classement avec notamment un PnL négatif pour le régression sans lag. Une analyse plus détaillée montre bien que le modèle XGBoost est meilleurs que les autres.

Remarque importante 1. *On remarque que la plupart des modèles ont enregistré un taux de transaction positif supérieur à 50prc. Pourtant, on a vu que les modèles (à part XGBoost qui était un peu plus convainquant que les autres, mais pas totalement sûr non plus) n'avaient pas un taux de réussite supérieur à la moyenne en ce qui concernait la prédiction du signe de la variation future du cours. Cela rejoint d'une part ce que nous disions dans la remarque 2. D'autre part, cela veut dire que si l'on veut faire du trading en utilisant des modèles de machin learning, il ne faut pas perdre de vue l'objectif principal qui est la maximisation du profit. Il faut donc prendre ce facteur en paramètre lorsqu'on évalue la pertinence d'un modèle et ne pas se concentrer uniquement sur les métriques conventionnelles d'évaluation de modèle.*

Mais alors comment se fait il que l'on puisse avoir un PnL positif avec des modèles qui se trompent 1 fois sur 2 lors de la prédiction du bon sens de variation du prix ? Essayons de donner une explication possible. Supposons que le modèle se trompe une fois sur 2 lors de sa prédiction du bon sens de variation uniformément à la hausse et à la baisse (c'est à dire qu'il se trompe autant en prédisant la hausse que la baisse). Ainsi, lorsque le modèle se trompe dans sa prédiction, on perd de l'argent et lorsque la prédiction est bonne, on en gagne. Le tout est censé se compenser pour donner à la fin un rendement nul. À présent, si on suppose que notre modèle est capable de nous prédire correctement un sens lorsque le marché présente certains paramètres. Nous aurons un léger avantage sur le marché, car on saurait prédire le cours si jamais ce paramètre se produit sur le marché. Ce léger avantage va en fait-tout tout changé, car il vient faire notre espérance de gain.

3.2 Slippage et frais de transaction

Pour être rentables, nos transactions doivent être suffisamment bonnes pour compenser tous les coûts de transaction. Les frais de change, mais aussi les coûts tels que le slippage sont cruciaux et ne doivent pas être négligés. Disons que nous achetons une quantité q de BTC et le revendons après une certaine période de temps. Voici ce que nous paierions en frais de transaction purs :

$$C = 2 \cdot q \cdot r + q \cdot s + f_s(q) + f_b(q)$$

où r représente le frais d'une transaction, s le ask-bid spread , $f_s(q)$ le slippage à la vente et $f_b(q)$ le slippage à l'achat. Ces slippage sont due à la liquidité insuffisante du marché et varient avec le temps. On peut les estimer sur le marché, mais nous avons besoin de l'historique des carnets d'ordres pour les estimer efficacement. On peut néanmoins supposer ici que le slippage est une variable aléatoire de loi $|N(0,1)|$ par exemple.

4 Conclusion

Certaines problématiques ont été abordées durant le projet mais pas énoncées dans le rapport comme par exemple le choix du marché ($i - e$ le choix des cryptomonnaies que l'on veut prédire) qui peut s'avérer crucial pour un projet de trading algorithmique sur cryptomonnaie (par exemple, le bitcoin sera beaucoup plus sensible à des facteurs extérieurs aux données de marché et donc des prédictions sur le cours du bitcoin peuvent s'avérer plus compliqué que si on voulait prédire une paire de cryptomonnaie moins connue du grand publique et dont le cours dépend en grande partie des données de marchés). Cependant, on a observé à travers ce projet qu'il est très difficile de prédire le cours d'une cryptomonnaie avec exactitude. Même si nous disposons des infrastructure nécessaire pour construire un bon modèle, il est fort probable qu'à notre échelle, cela soit insuffisant pour obtenir le bon taux de réussite de prédiction. Cependant, ce problème qui est important du point de vue du data scientist ne doit pas nous freiner dans notre démarche de trader, car on a vu qu'il suffit d'avoir un modèle qui nous donne un minimum de bonnes prédiction pour pouvoir construire une stratégie au profit positif en moyenne. Ce qui nous amène à une nouvelle problématique qui est celle de la détermination du niveau de performance nécessaire à un modèle pour pouvoir le considérer comme étant exploitable pour une stratégie de trading.