Master Economie et Finance
2021 - 2022

Dauphine | PSL
UNIVERSITÉ PARIS

MEMOIRE :

Badr Eddine EL HAMZAOUI et Cyril RAMEAUX

SOMMAIRE

« *Nous déclarons sur l'honneur que ce mémoire a été rédigé de nos mains, sans aide extérieure non autorisée, qu'il n'a pas été présenté auparavant pour évaluation et qu'il n'a jamais été publié, dans sa totalité ou en partie. Toutes parties, groupes de mots ou idées, aussi limités soient-ils, y compris des tableaux, graphiques, cartes etc. qui sont empruntés ou qui font référence à d'autres sources bibliographiques sont présentés comme tels, sans exception aucune.* »

Badr Eddine EL HAMZAOUI
Cyril RAMEAUX

Introduction

La crise des Subprimes de 2007 est la plus importante des crises financières que le monde n'ait jamais connues. Cette crise a non seulement provoqué une défaillance généralisée du système financier mondial, mais elle s'est également traduite par une perte de confiance de la part d'une vaste majorité d'agents financiers, dans les institutions bancaires. C'est dans ce contexte que la première cryptomonnaie a vu le jour. Une cryptomonnaie se définit comme une devise numérique qu'un réseau de personnes peut faire circuler, en l'absence de toute autorité centrale pour valider les transactions. La circulation, la transparence et la sécurité des transactions sont assurées par la technologie de la blockchain. La blockchain consiste en un système de stockage et de protection de l'information, reposant sur des techniques d'écriture de messages chiffrés ; technique appelée cryptographie. Le Bitcoin, première cryptomonnaie, fut introduite par Satoshi Nakamoto, le 1er novembre 2008, en pleine post-crise. Le Bitcoin est une devise numérique universelle, pouvant s'échanger contre toute autre devise et caractérisée par l'absence d'institution financière la régulant. Son cours est donc influencé uniquement par la loi de l'offre et de la demande ; ce qui a révolutionné la manière dont les agents ont conçu jusque-là la monnaie. Une devise de ce type permet de s'affranchir de contraintes, telle que l'inflation par-exemple, qui a amplement impacté les devises mondiales pendant la crise. Malgré cet avantage et la création en 2011 d'une autre cryptomonnaie relativement renommée aujourd'hui, l'Ethereum, il a fallu attendre les années 2020 pour assister à la montée en ampleur définitive de ces instruments financiers. De nos jours, les cryptomonnaies sont considérées comme des devises échangées essentiellement dans un but spéculatif. Elles ont néanmoins réussi à se forger une place à part entière sur les marchés financiers. Prédire le cours de ces instruments, si volatils et particulièrement peu prédictibles, comme nous avons pu le voir avec la montée en flèche, plutôt inattendue, du cours de Bitcoin à partir de 2020, est devenu un enjeu crucial pour une vaste majorité d'investisseurs. Dans le monde digitalisé dans lequel nous vivons, l'utilisation de l'intelligence artificielle, ou plus particulièrement du Machine Learning, apporte une assistance essentielle dans la correcte évaluation du cours des cryptomonnaies. Le Machine Learning correspond à une branche de l'intelligence artificielle qui consiste à apprendre à un ordinateur à réaliser, de manière autonome, des tâches sans que ce dernier soit programmé explicitement pour les faire. Plus concrètement, le Machine Learning comprend une phase d'apprentissage et une phase de mise en pratique. A partir d'une base de données initiales, le programme reconnaît à l'aide d'outils mathématiques et statistiques, des éléments récurrents, puis parvient à élaborer un modèle prédictif avec un objectif particulier. Par-exemple, il parvient à identifier la maladie d'un patient à partir de données sanguines ou encore

prédire le cours d'une cryptomonnaie à partir de certains facteurs. Dans un premier temps, pendant la phase dite d'apprentissage, le programme s'entraîne sur les données initiales et apprend à partir de celles-ci, à élaborer un modèle. Ensuite il s'agit de mettre en pratique ce que le programme a appris, sur une autre base de données, en évaluant sa performance et si nécessaire, recommencer l'étape d'apprentissage.

Ainsi, la problématique de ce mémoire est la suivante : comment construire des modèles d'évaluation des cryptomonnaies à l'aide de l'intelligence artificielle ? Dans un premier temps, nous allons expliquer comment nous avons construit la base des données et l'ensemble des facteurs que nous avons utilisés. Dans un deuxième temps, nous mettrons en œuvre à l'aide du Machine Learning, les modèles de pricing les plus connus, tout en expliquant leur fonctionnement et les techniques de Machine Learning employées pour la modélisation. Nous tâcherons également d'analyser les résultats des différents modèles et finalement de comparer leur performance, ce qui constituera notre conclusion.

I. Construction et analyse de la base de données

L'ensemble des données étudiées ont été collectées depuis la section cryptomonnaies de Yahoo finance. Nous allons nous focaliser sur l'évaluation du rendement des deux cryptomonnaies les plus anciennes mais également les plus renommées : le Bitcoin et l'Ethereum. Afin de pouvoir correctement entraîner nos programmes et en déduire des modèles prédictifs assez performants, nous avons besoin d'un historique de rendements complet ou autrement dit sans valeurs manquantes. De plus, la construction des facteurs explicatifs que nous avons sélectionnés et que nous allons vous présenter dans la section suivante, a nécessité l'utilisation d'autres cryptomonnaies. Les facteurs explicatifs, devant également éviter des valeurs manquantes, nous avons opté pour le 04/01/2018 comme date de début de notre base de données, quand bien même le Bitcoin et l'Ethereum sont en circulation depuis bien avant cette date. La performance d'un bon modèle en Machine Learning dépend de l'apprentissage des programmes élaborés et de la taille des données d'entrainement. Ainsi nous avons codé notre programme de récupération des données afin que la date de fin de nos historiques, soit le plus loin possible de la date de début et que la fréquence des observations soit journalière. Cela nous permet d'avoir plus d'observations disponibles pour l'entraînement de nos programmes de modélisation.

Finalement notre base de données contient les rendements du Bitcoin et l'Ethereum ainsi que les facteurs explicatifs du 04/01/2018 au 26/04/2022, à une fréquence d'observations journalière. Cette base de données est stockée dans le fichier « Data_Memoire.csv » et directement importée à travers le code.

a. Présentation des variables à évaluer

	Bitcoin	Ethereum
count	1573.0000	1573.0000
mean	0.0014	0.0020
std	0.0391	0.0502
min	-0.3717	-0.4235
25%	-0.0158	-0.0214
50%	0.0014	0.0010
75%	0.0180	0.0272
max	0.1875	0.2595

Figure : Statistiques de base

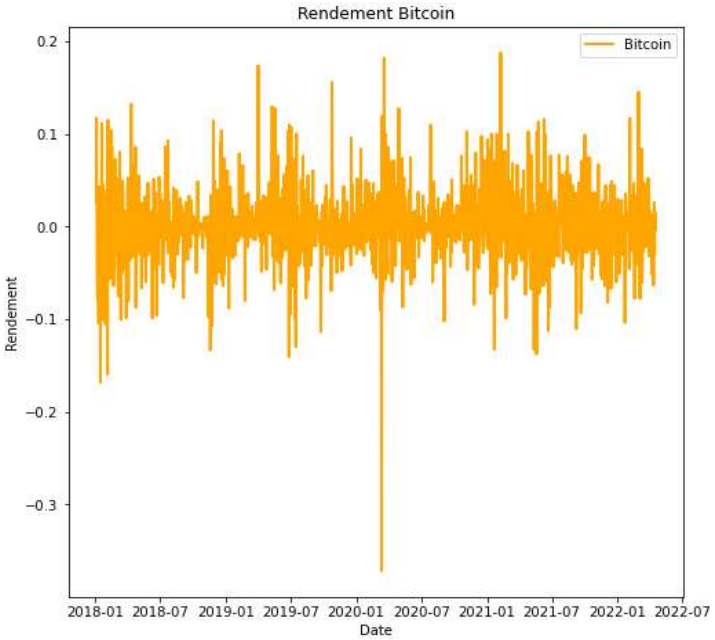

Figure 2 : Rendement du Bitcoin au cours du temps

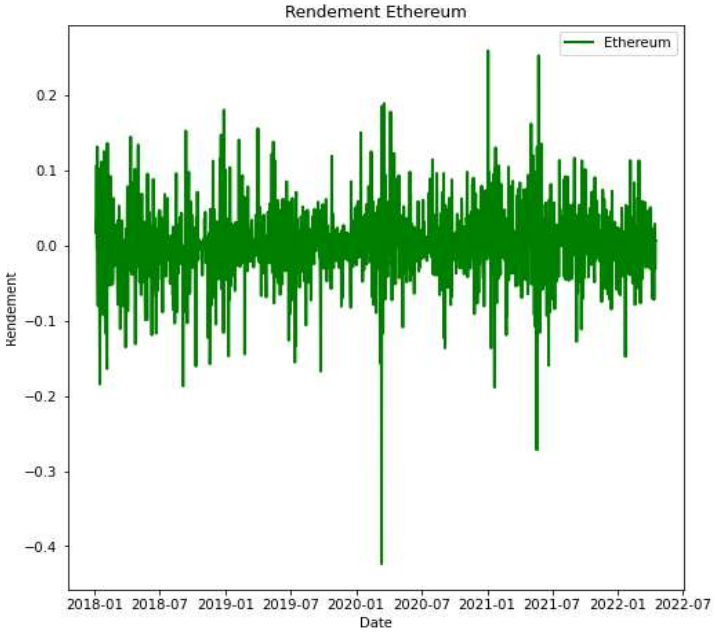

Figure 3 : Rendement de l' Ethereum au cours du temps

Notre base de données comporte 1573 observations journalières. Les écart-types de ces deux séries sont assez élevés. Nous avons un écart-type de 3,91% autour des valeurs dont la moyenne est 0,14% pour le Bitcoin et un écart-type d'environ 5% autour des valeurs dont la moyenne est de 0,2% pour l'Ethereum. Ce n'est pas une surprise, puisque ces instruments financiers sont renommés pour être assez volatiles ; ce qui est d'ailleurs une des principales raisons pour

laquelle nous nous intéressons à la construction de modèles pouvant correctement estimer leurs rendements. Trois quarts des rendements du Bitcoin sont inférieurs à 1,8% et le rendement maximal sur la période est de 18,75%. Pour ce qui concerne l'Ethereum le quantile à 75% vaut 2,72% et le rendement maximal vaut 25,95%. Ces valeurs du quantile à 75% couplées à celles du rendement maximal semblent suggérer l'existence de valeurs extrêmes dans nos deux séries de rendement. Ces valeurs extrêmes seront à supprimer ou à conserver en fonction de la performance des modèles que nous allons présenter dans la partie suivante. Les valeurs des différents quantiles nous indiquent également que nos observations sont assez concentrées dans une plage de valeurs. Les deux graphiques des rendements confirment ce constat. Les rendements du Bitcoin et de l'Ethereum oscillent entre -0,1% et 0,1%. Les graphiques permettent aussi de vérifier l'existence de quelques valeurs extrêmes : le creux exceptionnel en correspondance du début de la pandémie en 2020 en est un exemple. De plus, nous remarquons que l'évolution des rendements de ces deux cryptomonnaies est assez similaire, bien que l'Ethereum présente une performance moyenne légèrement supérieure à celle du Bitcoin. De ce fait, nous supposons que les modèles élaborés seront aussi performants à prédire le rendement du Bitcoin que celui de l'Ethereum. A ce stade, ceci est une simple conjecture que nous proposons de vérifier par la suite.

b. Construction des facteurs explicatifs

Les facteurs présentés dans cette section ont été construits à partir des douze premières cryptomonnaies par capitalisation boursière dont la période d'observation est compatible avec celle retenue pour le Bitcoin et l'Ethereum. Les douze cryptomonnaies retenues sont les suivantes : Bitcoin, Ethereum, Cardano, Bitcoin Cash, Binance Coin, Dogecoin, Chainlink, Litecoin, Decentraland, Tron, Tether et le Xrp. Pour la construction des facteurs, nous nous sommes inspirés, tout en apportant certaines modifications, des facteurs utilisés par Hujibregts dans sa thèse.

Indice de marché

Il n'existe pas vraiment d'indice représentatif du marché des cryptomonnaies. Nous avons donc opté pour le construire, de partir du portefeuille équipondéré des douze cryptomonnaies sélectionnées. Autrement dit, chaque valeur de cet indice correspond tout simplement au rendement moyen de ces cryptomonnaies.

Size

Dans un premier temps, nous avons classé le rendement des douze cryptomonnaies retenues pour la construction des facteurs, dans l'ordre croissant de la capitalisation boursière, puis nous avons découpé en deux sous-échantillons ces cryptomonnaies à l'aide des capitalisations affichées sur Yahoo Finance ; les cryptos à faible capitalisation - les «Small Caps» - et les cryptos dont les capitalisations sont élevées - les « Large Caps »-. Pour chaque sous-échantillon, nous avons calculé le rendement moyen par date, puis en soustrayant les rendements moyens des «Small Caps» à celui de «Large Caps», nous obtenons notre facteur size. Nous avons décidé d'inclure ce facteur dans notre modélisation, afin de tenir compte de l'effet de taille, une des fameuses anomalies de marché établies par les études menées sur la pertinence du « Capital Asset Pricing Model ». Dans le contexte des cryptomonnaies, l'effet de taille désigne le fait que les cryptomonnaies à faible capitalisation boursière ont tendance à surperformer celles à grande capitalisation.

Momentum

De la même manière que le facteur Size, nous avons construit ce facteur pour tenir compte d'une autre anomalie de marché : l'effet Momentum. Appliqué au monde des cryptomonnaies, l'effet Momentum indique le fait que des cryptomonnaies performantes par le passé ont tendance à l'être aussi bien, ou voire plus, dans les périodes à venir. Afin de construire ce facteur, nous avons commencé par fixer une période de référence, pour déterminer les « winners », les cryptomonnaies les plus performantes sur cette période, et les « loosers », celles qui sont les moins performantes. Comme période de référence, nous avons opté pour la période allant de la date de début de notre base de données à la fin de l'année 2019. Cela peut se justifier par notre volonté de ne pas prendre en compte l'année atypique qu'a été 2020, caractérisée par la pandémie de la Covid19 et la crise financière qui en a découlé. Nous avons ensuite calculé la moyenne des rendements en utilisant une fenêtre roulante de deux semaines. A partir de ces rendements, nous avons calculé les rendements moyens par cryptomonnaie, sur l'ensemble de la période de référence. A ce stade, nous obtenons un vecteur contenant un rendement moyen pour chaque cryptomonnaie, que nous avons classé dans l'ordre croissant et utilisé pour découper nos douze cryptomonnaies en deux sous-échantillons, en fonction du rendement moyen. Finalement, nous avons obtenu le facteur Momentum, en soustrayant le rendement des « winners » à celui des « loosers ».

Volatilité

Ce facteur a été construit sur le même principe que le facteur Momentum. Nous avons considéré la même période de référence. Nous avons ensuite calculé la moyenne des rendements en utilisant une fenêtre roulante de deux semaines. A partir de ces rendements, nous avons calculé les écart-types en obtenant ainsi un vecteur de douze volatilités, une par cryptomonnaie. De la même manière que le facteur Momentum, nous avons ensuite trié ce vecteur, puis nous l'avons utilisé pour découper l'historique initial de rendements en deux échantillons, un comportant les cryptomonnaies les moins volatiles et l'autre comportant celles les plus volatiles. Finalement, en soustrayant le rendement des cryptomonnaies moins volatiles au rendement de celles les plus volatiles, nous obtenons le facteur volatilité. La volatilité étant une caractéristique importante de ce type d'instrument financier, nous jugeons l'inclusion de ce facteur assez pertinente à la bonne prédiction du rendement du Bitcoin et de l'Ethereum.

Liquidité

Ce facteur a été également calculé en effectuant les mêmes étapes que celles réalisées pour le facteur Momentum. La seule différence est que nous avons considéré le volume de transactions (que nous avons retenu comme indice de liquidité des cryptomonnaies) au lieu des rendements des cryptomonnaies pour son calcul. Nous avons opté pour inclure ce facteur en supposant que plus la cryptomonnaie est liquide, ou autrement dit facilement disponible sur les marchés, plus son cours serait faible.

c. Présentation des autres facteurs explicatifs

L'ensemble des facteurs décrits dans cette section ont été directement collectés depuis Yahoo finance. Ces facteurs n'ont pas de cotations disponibles les week-ends. Pour pallier à l'existence de valeurs manquantes, nous avons remplacé les valeurs manquantes par la dernière valeur disponible dans l'historique, ou autrement dit par la cotation du vendredi. Puis, cela étant fait, nous avons transformé l'ensemble de ces cours en rendements. Ce choix implique, par construction, des rendements nuls le week-end. A ce stade, nous procédons ainsi, mais la pertinence de ce choix sera à remettre en question en fonction de la performance des modèles construits.

SP500, CAC40, Euro Stoxx 50

Le SP500 est un indice boursier regroupant les 500 sociétés américaines les plus importantes. Il est considéré comme l'indice le plus représentatif du marché boursier américain. Le CAC40 est le principal indice boursier français, réunissant les 40 plus grandes valeurs françaises en termes de volumes d'échanges de titres. Au même titre, l' Euro Stoxx 50 est un indice représentatif du marché de la zone euro, construit à partir des 50 plus grandes entreprises en termes de capitalisation boursière. Nous supposons que plus le rendement de ces indices de marché est élevé, plus les titres boursiers sont échangés, au détriment des cryptomonnaies, dont le rendement sera donc plus faible dans ces circonstances. De plus, ces indices sont représentatifs d'un marché dans des zones géographiques différentes. En comparant les expositions du rendement du Bitcoin et de l'Ethereum à ces indices, nous pourrions nous apercevoir de l'existence d'un lien entre ces rendements et la zone géographique.

VIX

Le VIX, connu aussi sous le nom d'indice de la peur, est un indice représentatif de la volatilité du marché américain. Lorsque cet indice est élevé, le marché américain est très mouvementé et sa volatilité élevée. Dans cette conjoncture, nous pouvons supposer que les agents financiers préfèreront alors s'orienter vers d'autres types d'investissements comme les cryptomonnaies par exemple. Or, nous savons que la majorité des cryptomonnaies sont très volatiles. Cette conjecture ne nous semble donc pas très pertinente. Nous décidons tout de même de garder ce facteur et de vérifier par la modélisation si le VIX influence ou non le cours des cryptomonnaies.

Or, Argent, Pétrole

Nous avons décidé d'inclure ces facteurs, pour tester si le rendement d'une cryptomonnaie est influencé par d'autres produits qui ne sont pas strictement financiers, telles que les valeurs refuge que sont l'or et l'argent ainsi qu'une matière première, à savoir le pétrole.

Treasury Yield Index-10 years

Cet indice est représentatif du rendement perçu par les investisseurs lorsqu'ils gardent des bons du trésor américains de maturité 10 ans, jusqu'à échéance. La prise en compte de ce facteur dans notre modélisation peut se justifier par notre volonté de vouloir mettre en relief un éventuel arbitrage existant entre des produits moins risqués, tels que les bons du Trésor, et des produits plus volatiles et risqués, telles que les cryptomonnaies.

d. Analyse de la relation Target / Facteurs

Dans cette section, nous nous occupons d'étudier la relation entre le rendement du Bitcoin et les différents facteurs explicatifs. L'enjeu est de vérifier l'existence d'éventuels liens de causalité en effectuant une analyse graphique des différentes relations.

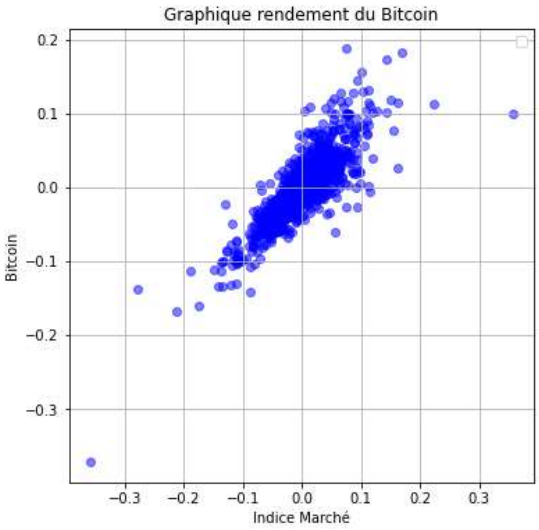

Figure 4 : Rendement du Bitcoin en fonction de l'indice du Marché

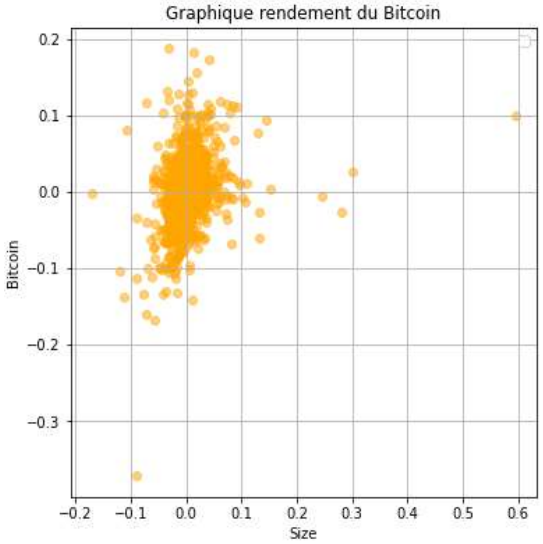

Figure 5 : Rendement du Bitcoin en fonction du facteur Size

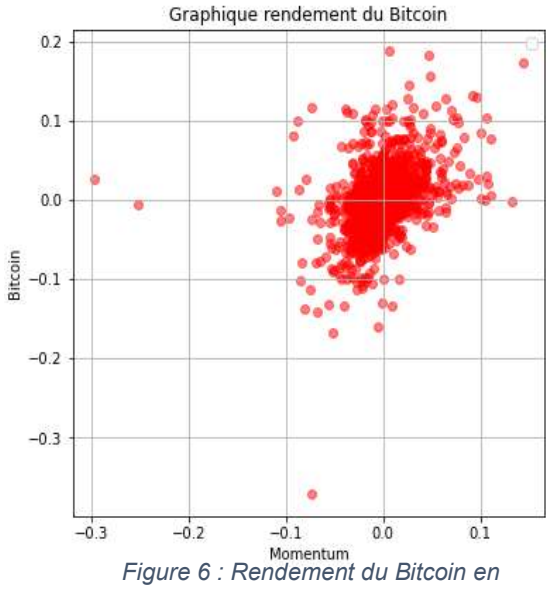

Figure 6 : Rendement du Bitcoin en fonction du facteur Momentum

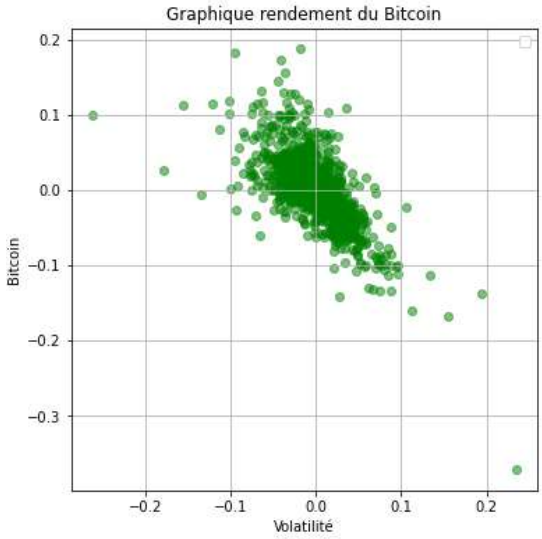

Figure 7 : Rendement du Bitcoin en fonction du facteur Volatilité

Nous remarquons que l'ensemble de facteurs Size, Momentum, Volatilité et Indice de Marché semblent influencer le rendement du Bitcoin. Plus la valeur de l'indice de marché est grande, plus le rendement du Bitcoin est élevé. Le Bitcoin semble ainsi évoluer dans le même sens que

l'indice représentatif du marché des cryptomonnaies. Ce même constat peut être observé pour le facteur Size et Momentum, même si dans le cas du facteur Momentum, la relation croissante est moins robuste et plus difficilement observable. A ce stade, nous en déduisons que l'effet taille ainsi que l'effet Momentum, dans une moindre mesure, influencent le rendement du Bitcoin.

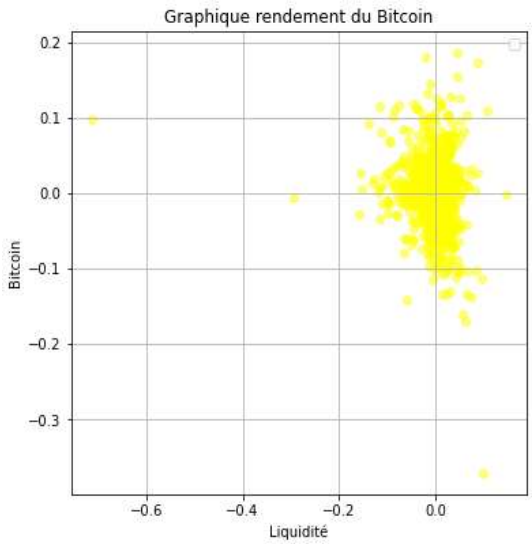

Figure 7 : Rendement du Bitcoin en fonction du facteur Liquidité

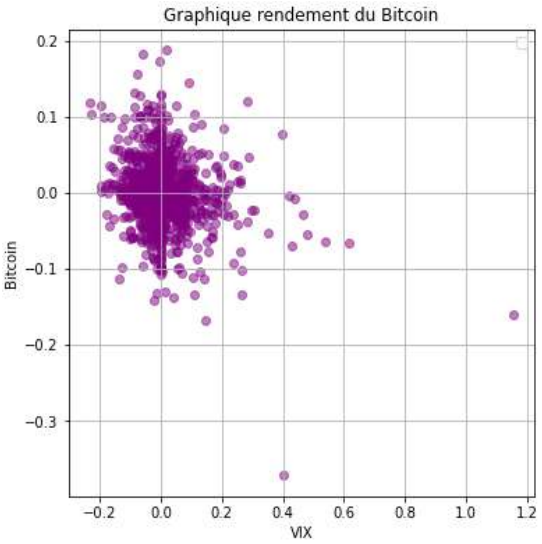

Figure 8 : Rendement du Bitcoin en fonction du VIX

Pour ce qui concerne le facteur Volatilité, plus les valeurs de ce facteur est élevé, plus le rendement du Bitcoin est faible. Cela est cohérent avec l'arbitrage rendement-risque que nous retrouvons facilement dans d'autres marchés, tels que le marché d'action par exemple.

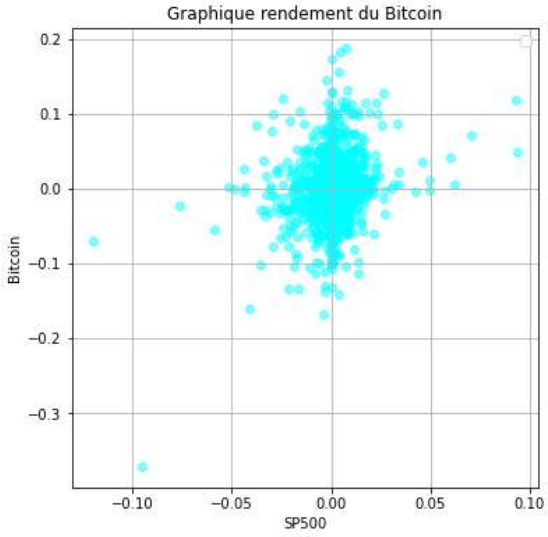

Figure 9 : Rendement du Bitcoin en fonction du SP500

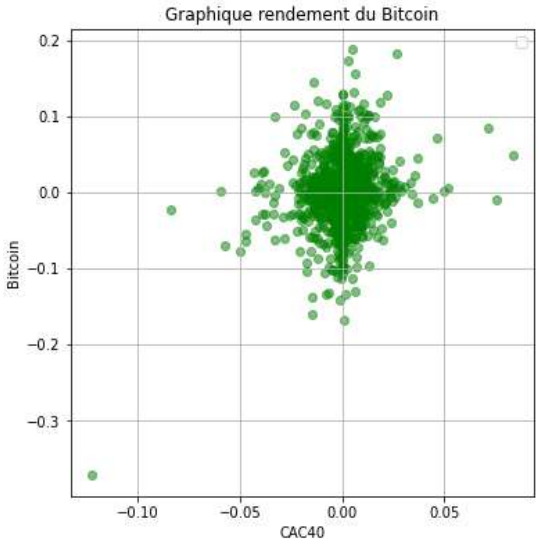

Figure 10 : Rendement du Bitcoin en fonction du CAC40

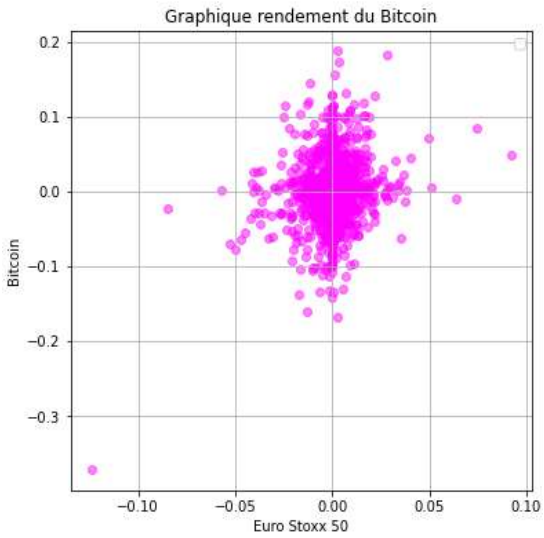

Figure 11 : Rendement du Bitcoin en fonction de l'Euro stoxx 50

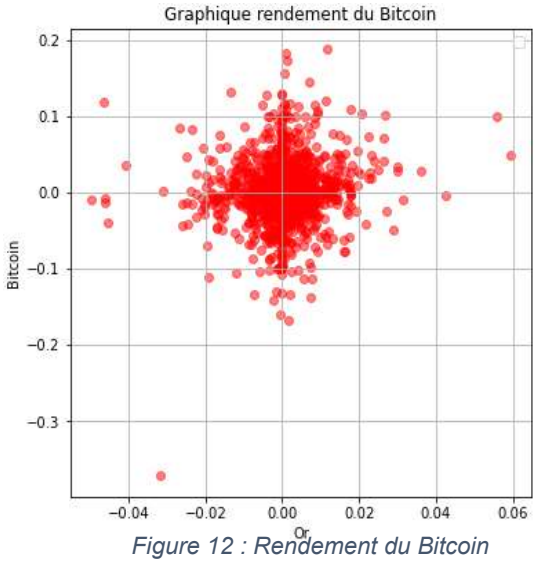

Figure 12 : Rendement du Bitcoin en fonction de l'Or

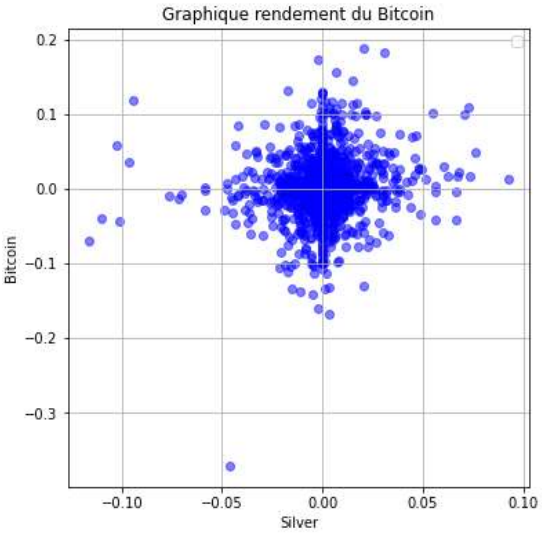

Figure 13 : Rendement du Bitcoin en fonction de l'Argent

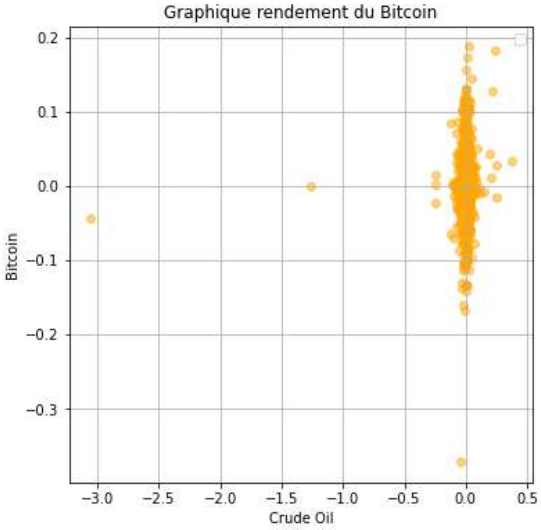

Figure 14 : Rendement du Bitcoin en fonction du pétrole

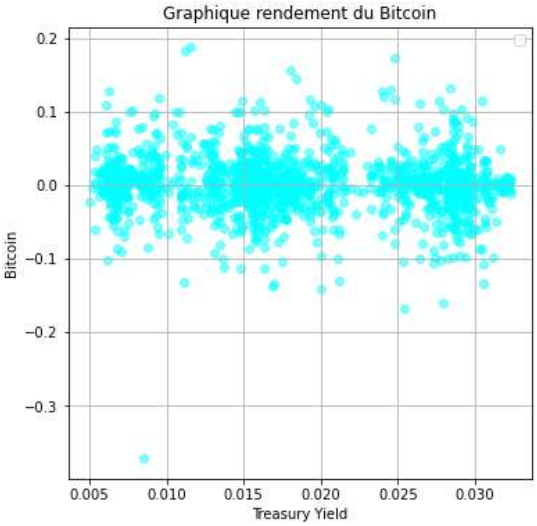

Figure 15 : Rendement du Bitcoin en fonction du taux sans risque

L'ensemble des facteurs représentés dans les figures de 7 à 15 ne semblent pas influencer significativement les rendements du Bitcoin selon les graphiques obtenus, contrairement à nos hypothèses initiales. Nous remarquons que pour les trois indices représentatifs du marché d'actions, que nous avons choisis, le nuage de points obtenu est assez similaire. Le même constat peut être effectué pour l'or et l'argent. Aucun lien de causalité n'est observable. Les rendements du pétrole sont assez faibles et concentrés. Le graphique qui en résulte est donc peu exploitable. Les graphiques du facteur liquidité, du facteur taux sans risque, et du Vix ne permettent pas de mettre en évidence l'existence d'un lien de causalité robuste avec le rendement du Bitcoin.

Nous avons réalisé les mêmes graphiques pour l'Ethereum et nous aboutissons aux mêmes conclusions observées pour le rendement du Bitcoin.

	Indice Marché	**Size**	**Momentum**	**Volatilité**	**Liquidité**	**SP500**	**CAC40**	**Euro Stoxx 50**
Bitcoin	0.84	0.29	0.4	-0.69	-0 .16	0.23	0.19	0.2
Ethereum	0.87	0.28	0.53	-0.79	-0.17	0.24	0.19	0.2

	Or	**Silver**	Pétrole	**Treasury Yield**	**VIX**
Bitcoin	0.097	0.12	0.055	-0.054	-0.22
Ethereum	0.081	0.12	0.05	-0.068	-0.23

Tableau 1 : Corrélation entre les variables à expliquer et les facteurs explicatifs

L'étude des corrélations entre les rendements du Bitcoin et de l'Ethereum et les facteurs, révèle que la plupart des facteurs que nous avons choisis sont assez corrélés avec les variables que nous chercherons à prédire, contrairement à ce que l'analyse graphique suggérait. Seuls les facteurs Or, Pétrole et Treausury Yield présentent des corrélations très faibles. Dans le cas de ces facteurs, les corrélations ne font que confirmer la moindre influence sur le rendement des cryptomonnaies, à laquelle nous avons conclu lors de l'analyse graphique.

Avant de supprimer définitivement les variables qui ne présentent pas un lien avec les rendements du Bitcoin et de l'Ethereum, au vu de constats que nous avons effectués ci-dessus,

nous allons construire quelques modèles en gardant la totalité des variables, puis nous réaliserons une sélection des variables à l'aide de tests de significativité.

II. Construction et analyse de modèles

Dans cette partie nous nous intéressons à la construction, l'évaluation et l'amélioration des modèles. Pour ce faire, nous avons commencé par découper notre jeu de données en trois échantillons : le « train set », le « validation set » et le « test set ». Le « train set » comprend 70% du jeu de données initiales alors que les deux autres se composent de 15% de données chacun. La découpe a été réalisée temporellement : le « train set » contient les données du 04/01/2018 au 09/01/2021, le « validation set » celles du 10/01/2021 au 02/09/2021 et le « test set » celles du 03/09/2021 au 26/04/2022. Le « train set », comme le nom l'indique, contient les données sur lesquelles nos modèles vont s'entraîner et apprendre les différents paramètres. Afin de pouvoir évaluer la performance des modèles et les comparer entre eux, il n'est pas pertinent d'utiliser le même jeu de données utilisé pour régler les paramètres du modèle. Pour réaliser le diagnostic final d'un modèle et choisir le meilleur modèle parmi plusieurs, il est nécessaire d'utiliser un jeu de données qui n'a jamais été utilisé. Si nous utilisons les données test pour améliorer nos modèles et les comparer entre eux, nous risquons d'obtenir des résultats biaisés, dans la mesure où ces données sont déjà connues par les modèles. Ainsi, pour améliorer les hyperparamètres d'un modèle, nous allons utiliser les données de validation, puis pour comparer plusieurs modèles concurrents, nous allons nous servir des scores de performance obtenus sur les données de test. L'utilisation du validation set est très importante, surtout pour les modèles plus sophistiqués ayant plus de réglages possibles. Autrement dit, pour ce qui concerne une régression simple, le fait d'utiliser les données d'entraînement ou de validation est indifférent, car il n'y a pas d' hyperparamètres à modifier pour améliorer le modèle.

Afin d'apprécier l'efficacité de nos modèles nous allons nous intéresser aux statistiques suivantes :

- MAE : Il mesure la moyenne des erreurs d'estimation sur l'ensemble des données. Il correspond à la moyenne de la différence absolue entre les valeurs réelles estimées de la variable à expliquer.
- MSE : l'erreur quadratique moyenne représente la moyenne du carré de la différence des valeurs réelles et prédites de la variable à expliquer. Il est un indicateur de la variance des erreurs de prédiction.

- RMSE : il correspond à la racine carrée de l'MSE. Il mesure l'écart-type des erreurs de prédiction.
- Coefficient de détermination R2 : correspond à la part de la variance de la variable cible qui est effectivement expliquée par les facteurs explicatifs dans le modèle choisi.

a. Modèles classiques

Dans cette section, nous allons recréer des modèles de base de la littérature financière, à l'aide du Machine Learning. Pour chaque modèle, nous allons dans un premier temps le présenter, puis nous allons le construire et analyser sa performance.

1. CPAM

Le Capital Pricing Asset Model (CPAM) est le modèle d'évaluation d'actifs financiers, le plus connu de la littérature financière. Ce modèle permet de déterminer la rentabilité attendue d'un actif risqué. Selon ce modèle, la prime de risque espérée pour l'investissement dans un actif, ou autrement dit la différence entre la rentabilité attendue de l'actif et le taux sans risque, dépend de la rémunération du risque systématique pour ce même actif. Cette rémunération correspond au produit entre le bêta (la sensibilité du titre au rendement de marché) et la prime de risque du marché (différence entre le rendement du marché et le taux sans risque). Le risque systématique ou risque de marché est le risque lié à l'évolution du marché. La présence du taux sans risque au sein de l'équation du modèle CPAM permet de tenir compte de la valeur des actifs non risqués. En d'autres termes, le taux sans risque représente un coût d'opportunité : il correspond au rendement des produits sans risque auxquels nous renoncerons en investissant dans un actif risqué supplémentaire. Malgré sa renommée, le CPAM est aussi un modèle assez contesté. En effet, les premiers tests économétriques pour ce modèle sont très décevants. Il a fallu attendre les travaux de Back, Jensen et Scholes qui ont l'idée de réaliser la régression sur des portefeuilles de très grandes tailles afin de minimiser les erreurs d'estimation, pour que la validité du modèle soit reconnue. Mais progressivement la validité empirique du modèle a été mise à l'épreuve par l'apparition des plusieurs anomalies, consistant dans la différence entre les rendements réels et ceux prédits par le modèle du CPAM.

Malgré cela, nous avons choisi de construire ce type de modèle pour le marché des cryptomonnaies, en utilisant l'indice de marché, que nous avons construit, et le rendement perçu par les investisseurs lorsqu'ils gardent des bons du trésor américains de maturité 10 ans jusqu'à échéance comme taux sans risque. L'équation du modèle que nous avons estimé est ainsi la suivante :

$$Er - r0 = \beta * (Erm - r0)$$

Avec Er le rendement du Bitcoin ou de l'Ethereum, r0 le taux sans risque, beta le coefficient que nous avons estimé et Erm le rendement de notre indice de marché.

Nous obtenons un beta égal à 0,83 pour le Bitcoin et un beta égal à 1.05 pour l'Ethereum. Les scores de performance du modèle sont résumés dans les tableaux suivants :

Bitcoin	RMSE (%)	MSE (%)	MAE (%)	R2 (%)
Entrainement	2,06	0,042	1,44	73,95
Validation	3,19	0,1	2,2	52,46

Tableau 2 : Scores modèle CPAM pour le Bitcoin

Ethereum	RMSE (%)	MSE (%)	MAE (%)	R2 (%)
Entrainement	2,19	0,045	1,4	81,69
Validation	3,9	0,16	2,49	58,05

Tableau 3 : Scores modèle CPAM pour l'Ethereum

Nous remarquons que pour les deux cryptomonnaies, le coefficient de détermination R^2 est plutôt élevé sur les données d'entraînement. Mais pour ce qui concerne les données de validation, le modèle ne parvient pas à généraliser efficacement les résultats de son apprentissage. En effet, uniquement 53% pour le Bitcoin et 58% de la prime de risque attendue sont expliqués par la prime de risque du marché. Nous retrouvons le même constat en observant les autres statistiques : la moyenne (MAE) et la variance (MSE) des erreurs de prédiction sont plus élevées pour les données de validation que pour celles d'entrainement. La validité empirique du CPAM est ainsi remise en question également dans le cadre du modèle que nous avons construit pour le marché des cryptomonnaies. Néanmoins cette perte de pouvoir explicatif peut s'expliquer par le constat suivant : une grande partie des données d'entrainement sont antérieures à la pandémie du covid-19, alors que les données de validation correspondent bien à la période après le début de la crise. Ainsi nous pensons que cette perte de pouvoir explicatif est due au fait que le modèle n'a pas été assez confronté à cette période atypique lors de sa phase d'apprentissage. A ce stade, nous nous limitons uniquement à relever ce constat, car nous disposons d'un troisième jeu de données (les données de test), correspondant à une période plus lointaine du début de la pandémie, pour laquelle nous nous attendons à avoir des scores similaires avec ceux obtenus sur le « train set ». Ayant utilisé la même découpe pour

l'ensemble des modèles que nous allons vous présenter, nous nous attendons à relever ce même constat aussi pour les autres modèles. Finalement, les betas sont tous les deux positifs, ce qui implique que les rendement du Bitcoin et de l'Ethereum évoluent dans le même sens du facteur de marché que nous avons construit.

2. Modèle Multifactoriel marché crypto

La perte de validité empirique du CPAM a conduit de nombreux chercheurs à pallier ce problème en développant des extensions du modèle. Parmi les extensions les plus célèbres, nous trouvons le modèle de Fama et French. Ce modèle comprend en plus du facteur marché, d'autres facteurs comme par exemple celui lié à la taille de la capitalisation boursière. Après sa publication en 1993, d'autres facteurs ont été ajoutés afin de tenir en compte des fameuses anomalies de marché que le modèle CPAM avait mis en lumière. Parmi ces facteurs, nous pouvons trouver le facteur Momentum qui permet de tenir compte des différences de rendement entre titres performants et ceux ayant une tendance à stagner plus ou voire, à baisser. Nous trouvons encore le facteur style qui permet de considérer la différence de rendement entre les actions sous-évaluées par le marché (les titres « value ») et les actions des entreprises avec un grand potentiel de croissance (les titres « growth »). L'estimation de facteurs supplémentaires a permis de tenir compte de ces anomalies de marché et d'augmenter le pouvoir explicatif de ce type de modèles, en leur rendant une alternative plus performante que le CPAM.

De ce fait, nous avons choisi de construire un modèle factoriel dans le cadre de l'estimation du rendement du Bitcoin et de l'Ethereum. Pour le choix des facteurs, nous nous sommes inspirés de la thèse de Hujibregts et du modèle de Fama et French. Nous rappelons que l'ensemble des facteurs utilisés ont été présentés dans la partie précédente. L'équation du modèle que nous avons estimé est ainsi la suivante :

$$Rit - RFt = 0{,}02 + 1{,}87 * (RMt - RFt) - 0{,}4 * T - 0{,}45 * M + 1{,}27 * V + 0{,}19 * L$$

Equation du modèle multifactoriel pour le Bitcoin

$$Rit - RFt = 0{,}003 + 1{,}13 * (RMt - RFt) - 0{,}31 * T + 0{,}09 * M - 0{,}08 * V + 0{,}25 * L$$

Equation du modèle multifactoriel pour l'Ethereum

Avec $Rit - RFt$ la prime de risque attendue, $RMt - RFt$ la prime de risque du marché, T le facteur taille, M le facteur Momentum, V le facteur volatilité et L le facteur liquidité. Les scores de performance du modèle sont résumés dans les tableaux suivants :

Bitcoin	RMSE (%)	MSE (%)	MAE (%)	R2 (%)
Entrainement	1,28	0,016	0,98	89,92
Validation	1,57	0,027	1,14	87,78

Tableau 4 : Scores modèle Multifactoriel pour le Bitcoin

Ethereum	RMSE (%)	MSE (%)	MAE (%)	R2 (%)
Entrainement	1,83	0,03	1,14	87,18
Validation	3,18	0,1	2,29	72,00

Tableau 5 : Scores modèle Multifactoriel pour l'Ethereum

Nous avons obtenu des modèles ayant un coefficient de détermination supérieur à 85% pour le Bitcoin et à 72% pour l'Ethereum. Le MAE et le MSE sont supérieurs pour le « validation set ». Le différentiel de pouvoir explicatif entre les données d'entraînement et celle de validation est assez réduit par rapport à celui obtenu pour le CPAM. En d'autres termes, le modèle multifactoriel parvient plus aisément à généraliser son apprentissage sur des données qu'il n'a jamais rencontrées. La constante dans ce modèle peut s'interpréter comme la part de la prime de risque espérée qui n'est pas expliquée par les facteurs. Pour les deux cryptomonnaies, cette part est presque nulle, ce qui démontre le fort pouvoir explicatif du modèle. De plus, nous remarquons que l'exposition de la prime de risque à l'indice de marché et au facteur taille est plus importante que les expositions aux autres facteurs pour l'Ethereum. En effet, comme nous l'avons précédemment montré, la corrélation avec ces facteurs est parmi les plus élevées. En revanche, le Bitcoin semble être aussi influencé significativement par le facteur volatilité et Momentum. De plus, les coefficients estimés sont globalement plus importants pour le Bitcoin que pour l'Ethereum. Cela semble suggérer que le type de cryptomonnaie influe sur l'exposition aux différents facteurs. Néanmoins ce constat est à nuancer. Le Bitcoin est la cryptomonnaie la plus ancienne et la plus célèbre. De ce fait, il n'est pas à exclure qu'elle ait un poids plus important dans la construction de nos facteurs. En effet, nous rappelons que l'ensemble des facteurs de ce modèle ont été construits à partir d'un échantillon de cryptomonnaies comprenant le Bitcoin. De ce fait, nous pensons que l'importance des expositions pour le Bitcoin est plutôt propre au jeu de données et à la manière dont nous avons construit nos facteurs.

b. Modèles de Machine Learning

Dans cette partie nous nous intéresserons à la construction de modèles propres au domaine du Machine Learning. La grande majorité de ces modèles utilisent dans leur phase d'entraînement, l'algorithme de la descente de gradient afin de correctement estimer les

paramètres du modèle. C'est le cas des modèles de régression comme le modèle CPAM, et Multifactoriel que nous avons présenté dans la partie précédente, ou bien d'autres modèles que nous présenterons dans cette partie (modèle de régression optimale et réseaux de neurones par exemple).

Ainsi, dans un premier temps, nous allons expliquer l'algorithme de la descente de gradient, puis nous réaliserons une sélection des variables à l'aide de tests de significativité. Nous finirons par illustrer le fonctionnement des différents modèles élaborés et commenter leurs résultats.

1) La descente de gradient

La majorité des algorithmes de Machine Learning sont basés sur la minimisation d'une fonction de coût. La fonction de coût décrit la manière dont les erreurs d'estimation réalisées par le modèle pendant la phase d'apprentissage, évoluent. Il existe différents types de fonctions de coût : parmi les plus utilisées, nous trouvons la fonction d'erreur quadratique moyenne : f=$\sum_{i=1}^{n}(y_i - a_i\ x_i)^2$où y est la variable expliquée et x la variable explicative (exemple d'un modèle à une seule variable). La minimisation de la fonction de coût permet de déterminer l'ensemble des coefficients des variables explicatives. Pour ce faire, nous pouvons utiliser la méthode des moindres carrés, mais aussi la descente de gradient. Ce type d'approche est à préférer à la méthode des moindres carrés ordinaires, qui est extrêmement chronophage lorsque le jeu de données est volumineux. Il est très important que la fonction de coût soit convexe. Une fonction convexe a un seul minimum, dit pour cette raison minimum global. Si nous réalisons une descente de gradient sur une fonction non-convexe, nous risquons que l'algorithme reste piégé dans un minimum local. Ce minimum ne correspond pas au minimum global de la fonction, ce qui amène donc à une estimation de paramètres ne minimisant pas totalement la fonction de coût.

Soit A le vecteur des coefficients a_i du modèle que l'on cherche à estimer, L la fonction de coût utilisé et α un paramètre appelé taux d'apprentissage. La descente de gradient consiste à appliquer à chaque paramètre a_i l'algorithme ci-dessous :

1. Nous initialisons le paramètre a_i de manière aléatoire ou utilisant un critère précis. Cette initialisation permet de déterminer le point de départ de la descente de gradient.
2. Pour chaque paramètre a_i nous réalisons l'opération suivante : $a_i = a_i - \alpha * \frac{d\,L}{d\,a_i}$

3. Tant que $\frac{d\,L}{d\,a_i} \neq 0$ ou en d'autres termes, tant qu'il n'y a pas de convergence du paramètre a_i nous reprenons à l'étape 2.

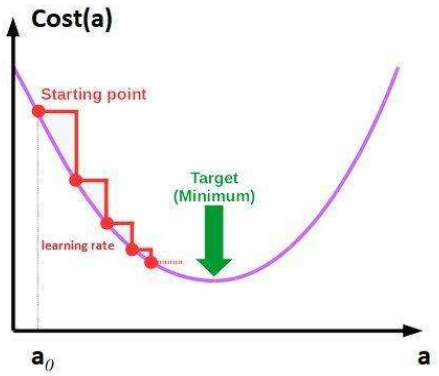

Figure 16 : Descente de gradient

Si le point de départ est à gauche comme dans la figure ci-dessus, alors la dérive de la fonction de coût par rapport au paramètre a est négative, car la pente de la tangente à la fonction en ce point est négative. Le taux d'apprentissage α est tout le temps positif, donc la quantité $a_i - \alpha * \frac{d\,L}{d\,a_i}$ est positive. Après une première itération, la valeur du paramètre a a augmenté. Cela se poursuit jusqu'à ce que la dérive de la fonction de coût soit nulle. En effet, en correspondance du minimum global, la pente de la tangente est nulle. Le raisonnement inverse s'applique si le point de départ est à droite du minimum global. Globalement, si nous considérons l'ensemble de coefficients a_i, l'algorithme s'arrête lorsque le vecteur A est convergent, ou autrement dit, lorsque le gradient de la fonction de coût par rapport à A est environ nul.
De plus, il est très important de bien choisir le taux d'apprentissage. Il correspond à la distance parcourue par l'algorithme à chaque itération. Plus il est important, plus l'algorithme converge rapidement vers le minimum. Néanmoins, un taux trop élevé implique également que l'algorithme oscille autour du minimum sans jamais l'atteindre.

2) Sélection des variables optimales

Pour obtenir des modèles de Machine Learning performants, il convient tout d'abord de trouver les bonnes variables explicatives. Pour ce faire, nous allons partir d'un modèle de régression linéaire et nous allons essayer de trouver la combinaison de variables qui nous donne le meilleur score, grâce à différents outils économétriques. Pour réaliser cette sélection, nous avons utilisé un test dit « f-régression ». Ce dernier commence par calculer la corrélation croisée entre

chaque facteur et la variable cible en utilisant la formule suivante : $\frac{(Fi-\underline{F})*(Ri-\underline{R})}{\sigma(F)*\sigma(R)}$, avec F et $\underline{F}$ le rendement et la moyenne des rendements du facteur, R et $\underline{R}$ le rendement et la moyenne des rendements de la variable à expliquer et σ l'écart-type. Puis ces corrélations sont transformées en une statistique de Fisher et classées par ordre d'importance. Nous avons également utilisé le sélecteur « SelectKBest » qui prend en argument un test économétrique et un nombre de variables k, et permet de sélectionner les k variables ayant les scores de Fisher les plus importants. Finalement, à l'aide de ces outils, nous avons bouclé sur le nombre de variables k en affichant à chaque itération le coefficient de détermination et la variance des résidus obtenus sur les données de validation, afin d'identifier le nombre optimal de variable k. Notre code affiche également le nom des k variables à sélectionner. Pour l'Ethereum, notre algorithme a conclu à la prise en compte de 4 variables supplémentaires par rapport à celles du Bitcoin : Liquidité, CaC40, Euro Stoxx 50 et VIX. Nous constatons qu'à partir de neuf variables, la variance des résidus pour le Bitcoin augmente pour après converger, alors que pour l'Ethereum, la variance des résidus commence à stagner à partir de huit variables. Le coefficient de détermination est assez similaire, que nous considérions huit ou neuf variables pour les deux cryptomonnaies. Nous décidons donc d'opter pour un modèle à huit facteurs, en supprimant le facteur Liquidité. Le modèle obtenu comprend les facteurs suivants : l'indice de marché (IM), le facteur taille (T), le facteur Momentum (M), le facteur volatilité (V), le SP&500 (SP500), le CAC40, l'Euro Stoxx 50 (EU50) et le VIX.

L'ensemble des modèles présentés ci-dessous seront ainsi construits en utilisant uniquement ces huit facteurs.

3) Modèle de régression linéaire

Une fois la sélection optimale opérée, nous allons pouvoir entraîner et évaluer le premier modèle. Nous tenons à rappeler que le CPAM et le Multifactoriel sont des modèles de régression linéaire pour lesquels nous avons uniquement considéré les facteurs inhérents au marché des cryptomonnaies. Dans cette section, nous allons également construire un modèle de régression linéaire sur la base de la sélection optimale de variables que nous avons précédemment réalisée.

Nous allons commencer par un modèle de régression linéaire qui vise à minimiser une fonction de coûts de type erreur quadratique moyenne grâce à une descente de gradient. Nous résolvons donc le programme suivant : Min L , où L est la fonction de coûts :

$L=\sum_{i=1}^{n}(y_i - a0 - a1_i\ IM_i - a2_i\ T_i - a3_i\ M_i - a4_i\ V_i - a5_i\ SP500_i - a6_i\ CAC40_i - a7_i\ EU50_i - a8_i\ VIX_i)^2$.

En minimisant cette fonction, nous trouvons l'ensemble des coefficients aj où j ∈ {1,..,8}. Nous allons ainsi déterminer l'équation de régression qui géométriquement correspond à un hyperplan.

Les équations du modèle que nous avons estimé sont ainsi les suivantes :

$$R = 2{,}74IM - 0{,}62T - 0{,}66M + 2{,}42V - 0{,}003 * SP500 + 0{,}05 * CAC40 + 0{,}04 * EU50 + 0{,}01VIX$$

Equation du modèle de régression optimale pour le Bitcoin

$$R = 1{,}22IM - 0{,}31T + 0{,}22M + 0{,}2 * V - 0{,}1 * SP500 + 0{,}09 * CAC40 + 0{,}11 * EU50 + 0{,}005VIX$$

Equation du modèle de régression optimale pour l'Ethereum

Nous n'avons pas reporté la constante car elle très proche de 0. Cette équation est obtenue sur les données du « train set » et pour évaluer notre modèle, nous reporterons les résultats de cette équation sur les données du test et du validation set.

La performance de ce modèle est résumé dans les tableaux ci-dessous :

Bitcoin	RMSE (%)	MSE (%)	MAE (%)	R2 (%)
Entrainement	0,75	0,005	0,51	96,24
Validation	0,93	0,009	0,59	95,63

Tableau 6 : Scores modèle de régression optimale pour le Bitcoin

Ethereum	RMSE (%)	MSE (%)	MAE (%)	R2 (%)
Entrainement	1,87	0,034	1,17	85,91
Validation	3,28	0,11	2,36	70,08

Tableau 7 : Scores modèle de régression optimale pour l'Ethereum

Pour le Bitcoin, la différence entre statistiques d'entraînement et celles de validation est très petite. De plus, le pouvoir explicatif du modèle est extrêmement élevé : seulement 4% environ de la variance du rendement du Bitcoin n'est pas expliqué par les huit facteurs explicatifs. En

revanche, pour l'Ethereum, le différentiel de pouvoir explicatif est beaucoup plus important. Le modèle rencontre plus de difficultés à généraliser pour l'Ethereum. Malgré cela, le score du modèle vaut environ 70% sur les données de validation ; ce qui est plutôt acceptable. De plus, nous remarquons encore une fois, que les expositions aux facteurs que nous avons construits sont plus élevées pour le Bitcoin que pour l'Ethereum. Comme indiqué dans la section précédente, cela est peut-être dû à l'importance du Bitcoin dans la construction des facteurs. Au contraire, les expositions des facteurs liés au marché financier (CAC40, EU50,SP500 et VIX) sont plutôt faibles par rapport à celles des facteurs que nous avons construits. De plus, lorsque nous nous intéressons aux trois indices du marché actions présents dans notre modèle, nous remarquons que le rendement du Bitcoin est influencé davantage par l'indice représentatif du marché français (le CAC40). En revanche, pour le rendement de l'Ethereum, l'exposition aux trois indices représentatifs des marchés actions que nous avons choisis, est assez proche (elle vaut environ 0,1 pour les trois indices). Selon ces résultats, il ne semble pas qu'il existe un lien entre les rendements de ces cryptomonnaies et la zone géographique des différents marchés financiers, comme nous l'avons conjecturé en première partie. Néanmoins nous avons constaté cela uniquement en considérant deux cryptomonnaies. De ce fait, un lien pourrait exister mais notre étude ne permet pas de le relever.

4) Modèle d'arbre de décision

Ce type de modèle permet la prédiction de la valeur cible en construisant des règles de décision à partir des caractéristiques des différents facteurs. En d'autres termes, à partir du jeu de données d'entraînement, le modèle parvient à construire un arbre de décision qui est utilisé pour attribuer une valeur à la variable à expliquer. Cette arbre de décision se compose d'un nœud racine qui constitue l'entrée de la totalité des donnés d'entraînement, de nœuds terminaux appelés feuilles et des nœuds intermédiaires. Ce type d'apprentissage revient à partitionner les jeux de données du nœud racine jusqu'aux feuilles en choisissant à chaque nœud, une variable explicative et une règle de décision permettant de réaliser le meilleur partage de données à chaque nœud de décision. Ce partage des données est réalisé de la manière suivante : pour chaque nœud, un facteur explicatif parmi ceux disponibles est sélectionné, puis une règle de partage sur ce facteur explicatif est choisie et utilisée pour découper la variable explicative en deux sous-échantillons. Plusieurs critères peuvent être utilisés pour déterminer le meilleur facteur explicatif et la meilleure règle de partage à utiliser. Dans le cadre d'un problème de régression, le critère le plus souvent utilisé est la variance de la variable cible. L'idée derrière l'utilisation de cette variance, est la recherche d'une règle de partage qui permet de réduire la

variance par rapport au nœud qui précède. Considérons l'exemple suivant : nous avons un jeu de données contenant une variable y et deux facteurs explicatifs x0 et x1. Des points rouges correspondent à des valeurs élevées de la variable y, à l'inverse des points jaunes correspondent à des valeurs faibles. Nous considérons deux règles de partage : soit x0 inférieur ou égal à 1 soit x1 inférieur ou égal à 2. Cet exemple est résumé par la figure ci-dessous :

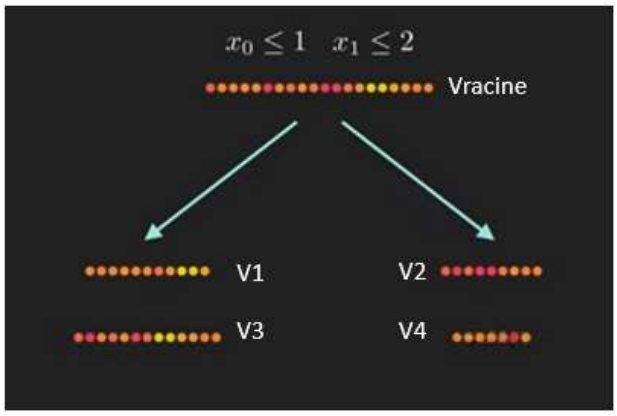

Figure 17 : Schéma explicatif du partage dans un arbre de décision

Soit V la variance. V1 et V2 correspondent aux variances des nœuds du partage de la variable cible, suivant la règle $x0 \leq 1$, V3 et V4 correspondent aux variances des nœuds du partage de la variable cible, suivant la règle $x1 \leq 2$ et Vracine correspond à la variance de la variable cible dans le nœud racine. Pour décider de la meilleure règle de partage entre les deux, l'algorithme de l'arbre de décision commence par calculer la moyenne pondérée par le nombre d'observations dans chaque nœud des variances et ce, pour l'ensemble des règles de décision candidat. A titre illustratif, pour la première règle de décision, le calcul réalisé est le suivant : $\underline{Vnoeud} = V1 * \frac{11}{20} + V2 * \frac{9}{20}$. Finalement, l'algorithme choisit le partage qui correspond à la réduction de variance la plus importante ou autrement dit, le partage qui minimise la différence entre la variance au nœud précédent (dans notre exemple Vracine) et la quantité $\underline{Vnoeud}$.

Ainsi, ce type de modèle traite l'ensemble des combinaisons de facteurs et de règles de partage et utilise cette démarche afin d'en déduire le meilleur partage à chaque nœud, en parvenant à construire ainsi un arbre de décision. Généralement, lorsque le nombre d'observations dans un nœud est inférieur ou égal à 5, le partage des données est arrêté. Une fois l'arbre construit, l'estimation de la variable à expliquer est réalisé de la manière suivante : en fonction des différentes valeurs des facteurs explicatifs et des règles de partage à chaque nœud, nous arrivons à un nœud feuille, puis l'estimation réalisée par l'algorithme correspond simplement à la moyenne des observations de la variable cible pour ce nœud.

Nous avons donc construit ce modèle d'arbre de décision. Nous remarquons que les scores de performance sur les données de validation sont assez faibles par rapport aux scores obtenus jusqu'ici. Nos modèles ont un pouvoir explicatif de 100% sur les données d'entraînement. En revanche, pour les données de validation, nous remarquons une baisse d'au moins 35% en pouvoir explicatif. En effet, nous avons un R^2 d'environ 40% pour l'Ethereum et un R^2 de 64,88% pour le Bitcoin. Malgré une phase d'apprentissage parfaite, nos algorithmes d'arbre de décision ne parviennent pas à aussi bien performer sur des données nouvelles.

De ce fait, nous allons chercher à améliorer la performance du modèle en modifiant ses hyperparamètres. Un hyperparamètre d'un modèle est un paramètre qui n'est pas estimé lors de la phase d'apprentissage, mais qui est plutôt utilisé comme argument du modèle afin de contrôler et modifier le processus d'apprentissage. Dans le cas de notre modèle d'arbre de décision, nous avons considéré les hyperparamètres suivants : l'étendue maximale de l'arbre de décision, le nombre minimal d'observations dans un nœud pour effectuer un partage supplémentaire, le nombre maximal de nœuds, le nombre maximal de variables explicatives à considérer lorsque nous cherchons le meilleur partage et le pourcentage minimal d'observations pour chaque nœud. Comme nous pouvons le remarquer, ces valeurs ne sont pas estimées par le modèle lors de sa phase d'apprentissage. Ils correspondent bien à des hyperparamètres. Pour chaque hyperparamètre, nous avons listé une série de valeurs possibles. Puis nous avons utilisé l'outil « GridSearchCV » fourni par le module « sickit learn » de python afin d'identifier la meilleure combinaison possible des valeurs des hyperparamètres, parmi celles que nous avons listées. Nous avons indiqué à l'outil « GridSearchCV » d'utiliser le score R^2 comme métrique de comparaison, qui nous a sorti les valeurs des hyperparamètres à prendre en compte, permettant d'obtenir le modèle d'arbre de décision ayant le plus fort pouvoir explicatif. Les statistiques de ce dernier modèle sont résumées dans les tableaux suivants :

Bitcoin	**RMSE (%)**	**MSE (%)**	**MAE (%)**	**R2 (%)**
Entrainement	1,87	0,03	1,13	76,64
Validation	2,41	0,058	1,75	71,14

Tableau 8 : Scores modèle d'arbre de décision pour le Bitcoin

Ethereum	**RMSE (%)**	**MSE (%)**	**MAE (%)**	**R2 (%)**
Entrainement	2,13	0,04	1,3	81,65
Validation	3,67	0,13	2,55	62,65

Tableau 9 : Scores modèle d'arbre de décision pour l'Ethereum

Après modification des hyperparamètres, nous remarquons que le score sur les données d'entraînement a clairement baissé. Cela ne nous a pas surpris, car la modification des hyperparamètres impacte la manière dont notre algorithme apprend à construire l'arbre de décision à partir des facteurs explicatifs. Cette perte de pouvoir explicatif lors de la phase d'apprentissage, a permis en revanche d'augmenter d'environ 20% le R^2 sur les données de validation pour l'Ethreum et d'environ 9% pour le Bitcoin. Malgré cette hausse, le pouvoir explicatif du modèle d'arbre de décision n'est pas assez satisfaisant et cela, en particulier pour l'Ethreum. De ce fait, nous soupçonnons que ce type de modèle n'est pas vraiment adapté à l'estimation de rendements des cryptomonnaies et nous décidons donc de passer à un modèle plus sophistiqué : le Random Forest.

5) Modèle Random Forest

Le modèle Random Forest est une extension du modèle d'arbre de décision qui consiste, comme son nom l'indique, à construire une « forêt d'arbres de décision ». L'algorithme du Random Forest repose sur la construction d'une multitude d'arbres de décision indépendants. Pour y parvenir, les données d'entraînement sont partagées en plusieurs sous-ensembles, de manière aléatoire, et chaque sous-ensemble de données est utilisé pour construire un arbre de décision. La technique utilisée pour le découpage des données d'entraînement est « l'échantillonnage par bootstrap » qui consiste en la création de plusieurs échantillons en effectuant un tirage avec remise des observations du « train set ». De plus, ce modèle repose aussi sur une sélection aléatoire des facteurs explicatifs. Pendant la phase d'apprentissage, à chaque nœud d'un arbre, l'algorithme du Random Forest considère uniquement un sous-ensemble de facteurs explicatifs, sélectionnés de manière aléatoire, afin de partitionner les données. En découpant de manière aléatoire les données d'entraînement et en tirant aléatoirement les facteurs disponibles pour diviser les données à chaque nœud de décision, ce modèle permet d'obtenir des arbres de décision peu corrélés entre eux. Cela est d'autant plus vrai que le nombre d'arbres de décision dans la forêt aléatoire est important. Ce procédé de réduction de la dépendance des différents arbres est très important pour l'estimation de la variable cible. L'estimation de la variable à expliquer dans ce type de modèle et pour un problème de régression comme le nôtre, correspond à la moyenne des valeurs prédites par l'ensemble des arbres de décision. En réduisant la corrélation entre les arbres, nous réduisons également la corrélation entre les erreurs d'estimation et ainsi le risque d'erreur d'estimation. De plus, ce modèle permet de pallier un problème souvent rencontré par le modèle d'arbre de décision simple : le surapprentissage. Ce dernier correspond à la situation où l'arbre de décision a trop appris les caractéristiques des

donnés d'entraînement. Il affiche alors un score élevé pour les données d'entraînement mais une performance médiocre lorsqu'il est confronté à des nouvelles observations.

Comme pour le modèle d'arbre de décision, les scores sur les données de validation de nos premiers modèles de Random Forest n'est pas assez satisfaisant. Nous avons un R2 d'environ 40 % pour l'Ethereum et de 64,8% pour le Bitcoin. Compte tenu de ces scores, nous avons cherché à améliorer le modèle en modifiant les hyperparamètres. Aux hyperparamètres considérés pour le modèle d'arbre de décision, nous avons ajouté le nombre d'arbres dans la forêt aléatoire. Puis en exploitant les mêmes outils utilisés pour le modèle d'arbre de décision, nous aboutissons aux modèles dont les performances sont résumées ci-dessous :

Bitcoin	RMSE (%)	MSE (%)	MAE (%)	R2 (%)
Entrainement	1,54	0,024	0,93	84,32
Validation	2,22	0,049	1,65	75,42

Tableau 9 : Scores modèle Random Forest pour le Bitcoin

Ethereum	RMSE (%)	MSE (%)	MAE (%)	R2 (%)
Entrainement	1,86	0,04	1,09	86,00
Validation	3,41	0,11	2,43	67,65

Tableau 10 : Scores modèle Random Forest pour l'Ethereum

La modification des hyperparamètres amène à une augmentation du pouvoir explicatif sur les données de validation. Cette modification permet d'améliorer davantage la performance du modèle pour l'Ethereum, qui affiche une hausse de 27% en pouvoir explicatif. En ayant utilisé les mêmes hyperparamètres, à l'exception d'un seul (le nombre d'arbres dans le Random Forest) pour le modèle d'arbre de décision simple et le modèle de forêt aléatoire, nous pouvons comparer leurs performances sur le « validation set ». Nous remarquons que la construction de plusieurs arbres de décision permet d'augmenter le R2 d'environ 4%. Ainsi nous pouvons conclure que le modèle de forêt aléatoire est plus performant.

6) Support Vecteur Machine

Pour comprendre le principe du modèle de support vecteur machine, revenons rapidement sur le modèle de régression linéaire. Nous avons vu qu'une régression linéaire classique vise à minimiser la fonction de coûts $f=\sum_{i=1}^{n}(y_i - w_i\ x_i)^2$, si nous sommes dans le cas d'un modèle avec une seule variable explicative. Cette fonction correspond à la somme au carré des erreurs

entre la valeur réelle y et l'estimation $\hat{y}$. Par exemple, si nous voulons étudier la relation entre le prix des maisons et le nombre de pièces, y correspond au prix et x au nombre de pièces. Le modèle de régression linaire nous donnera une équation qui en 2 dimensions correspond à la droite rouge suivante :

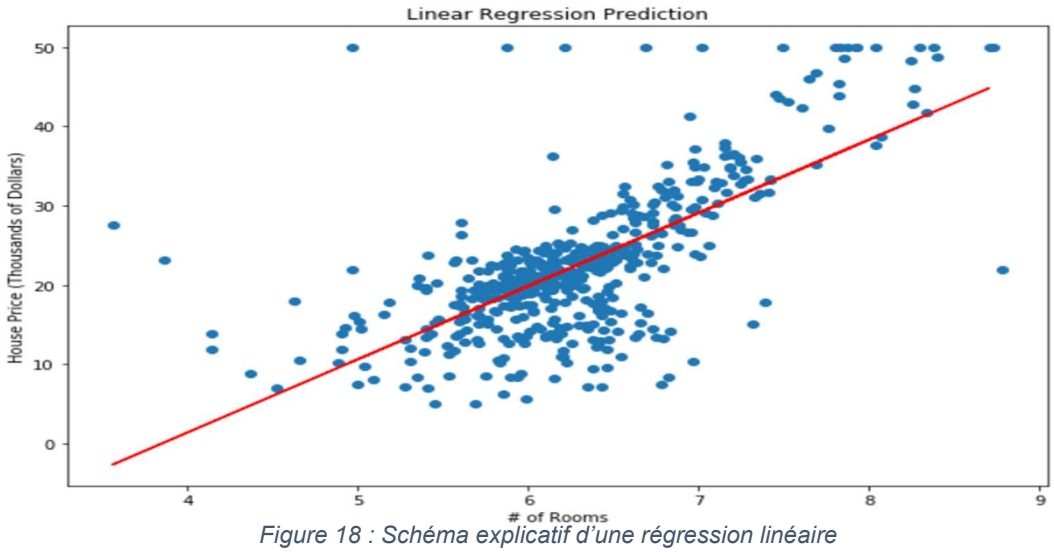

Figure 18 : Schéma explicatif d'une régression linéaire

Mais que se passe-t-il si nous nous fixons comme objectif de réduire la fonction de coût que dans une certaine mesure ? Comment réagirait le modèle, si nous ne nous soucions pas de l'ampleur de nos erreurs, tant qu'elles se situent dans une plage acceptable ? Pour ce faire, nous utilisons le modèle de Support Vecteur Machine appliqué à la régression (SVR).

Contrairement au modèle de régression, la fonction objective du SVR ne consiste pas à minimiser les erreurs, mais à minimiser les coefficients, et plus précisément la norme du vecteur de coefficients. Le terme d'erreur est géré dans une contrainte où nous définissons l'erreur absolue de manière qu'elle soit inférieure ou égale à une marge spécifiée (ε). Ainsi, dans un modèle à une seule variable, nous résolvons le programme suivant :

$$\begin{cases} Min\ \frac{1}{2}\ \|w\|^2 \\ sc:\ |y_i - w_i\, x_i| \leq \varepsilon \end{cases}$$

Nous obtenons ainsi un résultat de ce type :

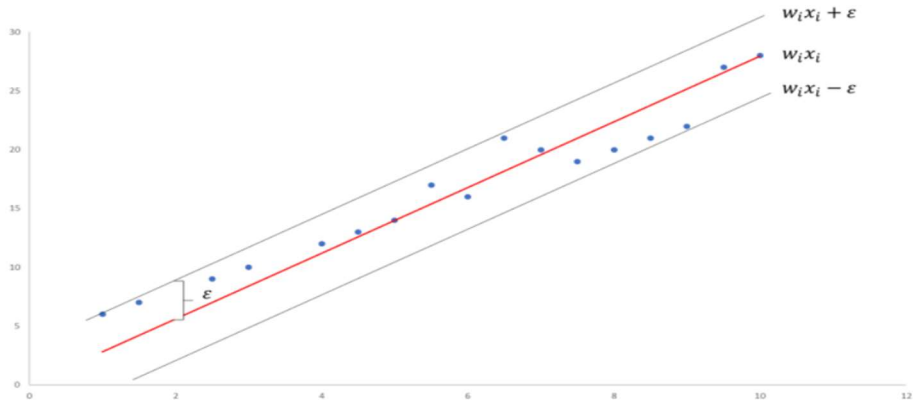

Figure 19 : Schéma explicatif de la marge d'erreur

Mais il se peut que tous les points de données ne se situent pas à l'intérieur de ces marges. Ce sont des écarts que nous voulons minimiser afin d'obtenir le modèle le plus efficace. Pour ce faire, nous pouvons déterminer une variable d'écart. Le principe de ce type de variable est que pour toute valeur qui tombe en dehors de ε, nous définissons son écart par rapport à la marge que l'on note ξ et que nous allons chercher à minimiser. Le programme à résoudre est désormais :

$$\begin{cases} Min\ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}|\xi_i| \\ sc:\ |y_i - w_i x_i| \leq \varepsilon + |\xi_i| \end{cases}$$

Nous avons donc un nouveau hyperparamètre C que nous pouvons ajuster. Lorsque C augmente, la tolérance pour les points en dehors de la marge augmente et inversement, si C diminue.

La résolution de ce programme va nous donner une équation qui nous permettra de réaliser les estimations. Pour mieux comprendre le principe, nous pouvons regarder le graphique suivant :

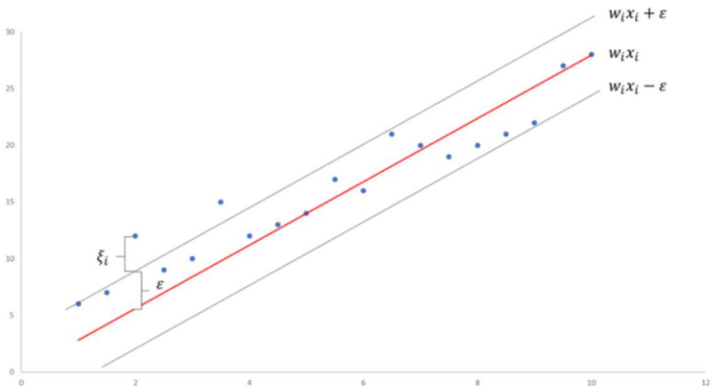

Figure 20: Schéma explicatif de la variable d'erreur

La droite rouge représente l'équation du modèle. Ici, il s'agit d'une droite car nous avons un modèle avec une seule variable et que notre problème est de type linéaire.

Pour appliquer cet algorithme à notre étude, nous avons utilisé LinearSVR du module scikit_learn. En lançant une première fois le modèle, sans jouer sur les hyperparamètres, nous obtenons des scores satisfaisants de R2 sur le validation set, avec 69% pour l'Ethereum et 92% pour le Bitcoin. Comme nous venons de le voir, ε et C sont des hyperparamètres qui vont affecter le programme à résoudre. Comme pour les modèles précédents, nous avons essayé de trouver les meilleures combinaisons possibles, pour obtenir un C égal à 4.5 et un ε égal à 0.001 qui, au départ étaient égaux à 1 et 0 par défaut. Nous obtenons alors les valeurs suivantes :

Bitcoin	**RMSE (%)**	**MSE (%)**	**MAE (%)**	**R2 (%)**
Entrainement	0.77	0.01	0,52	96,05
Validation	0,94	0,09	0,62	95,59

Tableau 10 : Scores modèle SVR pour le Bitcoin

Ethereum	**RMSE (%)**	**MSE (%)**	**MAE (%)**	**R2 (%)**
Entrainement	1,867	0,04	1,17	85,83
Validation	3,28	0,11	2,36	70 ,1

Tableau 11 : Scores modèle SVR pour l'Ethereum

Nous remarquons que l'augmentation des paramètres permet d'augmenter les scores du modèle sur le validation set même si la hausse n'est pas très importante. Par rapport aux résultats des modèles construits précédemment, nous pouvons envisager qu'il s'agit d'un modèle intéressant, notamment par rapport aux modèles d'arbres.

7) Le réseau de neurones

Les réseaux de neurones est un modèle qui a été inspiré des neurones humains. Le premier réseau de neurones, à proprement dit, est apparu en 1986 lorsque David Rumelhart réussit à généraliser son modèle à un seul neurone artificiel et publie le modèle du perceptron multicouche reposant sur la rétropropagation du gradient. Aujourd'hui, ce modèle est amplement utilisé avec beaucoup de succès dans différents domaines.

Un neurone est l'unité de base d'un réseau de neurones. Il est représenté dans la figure suivante :

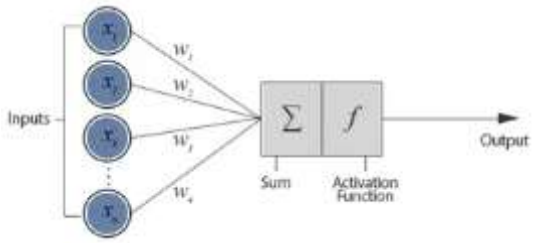

Figure 21 : Schéma explicatif d'un neurone artificiel

Un neurone artificiel reçoit en entrée les données d'entraînement et renvoie une sortie. Pour ce faire, il attribue à chaque entrée xi un poids wi. A partir de ces poids, le neurone opère une transformation sur les données d'entrée, traduite par une fonction dite d'agrégation. Le plus souvent, la fonction d'agrégation correspond à la somme pondérée par le poids wi des entrées xi. Le résultat de cette fonction est ensuite passé à une fonction d'activation. La fonction d'activation détermine l'état d'activation du neurone. Dans le cas d'un problème de régression, l'état d'activation du neurone va impacter la réponse élaborée par le neurone. De la même manière qu'un neurone biologique, la fonction d'activation permet, dans le cas du neurone artificiel, d'impacter le passage d'information lorsqu'un seuil d'activation du neurone est atteint ou en d'autres termes, lorsque le neurone est assez stimulé pour être actif. Il existe plusieurs types de fonctions d'activation. Le type de fonction d'activation utilisée est ainsi un hyperparamètre qui peut amener à des résultats différents, selon le jeu de données d'entraînement. Finalement, les similitudes avec le neurone biologique sont multiples : le neurone artificiel reçoit des entrées qu'il traite et qui déterminent son état d'activation et la conséquente sortie, transmise à la couche suivante.

Il existe également plusieurs types de réseaux de neurones. Nous avons opté pour la construction du réseau multicouche classique avec rétropropagation tel que celui présenté par Rumelhart en 1986.

Un réseau correspond simplement à un ensemble de neurones organisés en couches. Il est représenté dans la figure ci-dessous :

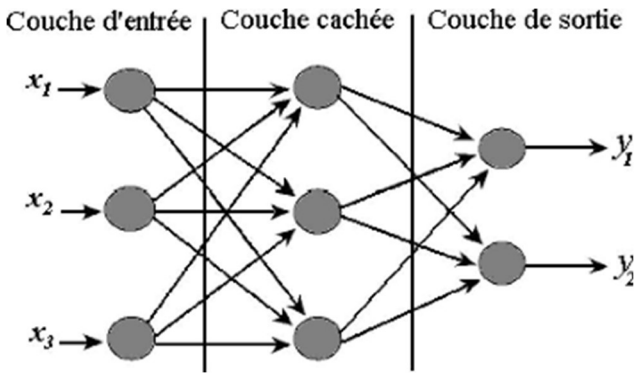

Figure 22 : Schéma explicatif d'un réseau de neurones

Un réseau de neurones est composé de plusieurs couches : une d'entrée, une de sorties et des couches intermédiaires dites aussi couches cachées. Chaque couche est caractérisée par un ensemble de neurones qui ne sont pas liés entre eux. Chaque neurone reçoit des entrées et lui associe des poids. Généralement, la couche d'entrée est composée d'autant neurones que le nombre de variables explicatives. Cette couche permet simplement le passage des entrées sans opérer de traitement dessus. Chaque entrée d'un neurone est reliée à tous les neurones de la couche précédente. La couche de sortie peut se composer de plusieurs neurones. Finalement, la dernière couche permet l'estimation de la variable cible, à partir des observations fournies en entrée à la couche initiale.

L'apprentissage dans un réseau de neurones se compose de deux étapes : une de propagation en avant et une dite rétropropagation du gradient. Dans un premier temps, les différents poids wi sont initialisés de manière aléatoire. Pendant la phase de propagation en avant, l'algorithme parcourt tous les neurones de chaque couche intermédiaire et calcule la fonction d'agrégation, puis celle d'activation. A chaque couche, les valeurs de la fonction d'activation sont ensuite transmises aux neurones de la couche suivante jusqu'à parvenir aux neurones de la couche de sortie. Les neurones de cette dernière couche calculent les valeurs d'activation ou en d'autres termes, les sorties du réseau à partir des sorties de la couche précédente et en déduisent la valeur de la variable à expliquer. A ce stade commence la phase d'ajustement des poids. Cette dernière consiste dans une descente du gradient telle que nous l'avons décrite précédemment. Nous rappelons que la fonction de coût permet d'évaluer les erreurs effectuées par le modèle. La descente permet d'ajuster la valeur des différents poids wi sur l'ensemble des neurones de manière à minimiser la fonction de coût. Nous mentionnons que les poids sont ajustés en utilisant les formules de la descente de gradient suivantes : $w_i = w_i - \alpha * \frac{d\,L}{d\,w_i}$ avec α le taux d'apprentissage et L la fonction de coût. La difficulté dans le cas d'un réseau de neurones se trouve dans le calcul des dérivés de la fonction de coût. Par pallier à ce problème, l'algorithme utilise la méthode de rétropropagation. Cette méthode consiste à calculer les dérives de la fonction de coût par rapport aux paramètres du modèle (les poids wi) de la couche de sortie jusqu'à celle d'entrée. Afin d'expliciter plus facilement le déroulement de cette méthode, considérons le réseau à deux couches, représenté dans la figure ci-dessous :

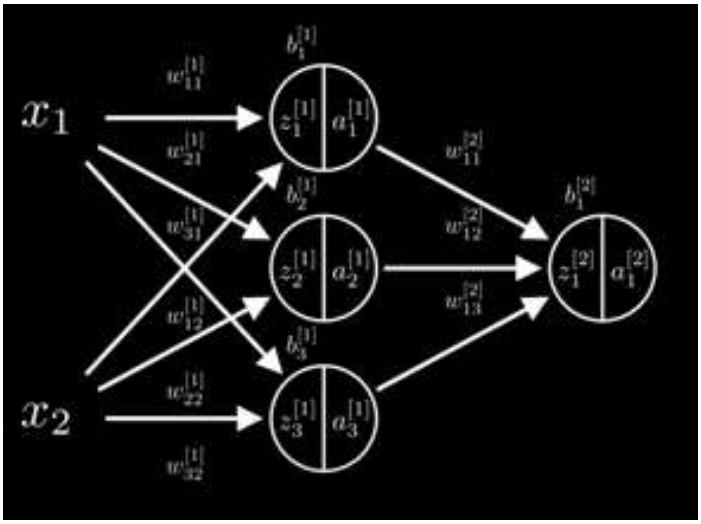

Figure 23 : Schéma explicatif d'un de la rétropropagation

Les valeurs z et a correspondent respectivement aux valeurs de la fonction d'agrégation et de la fonction d'activation et w_{ij}^2 les poids de neurones de la couche numéro 2 et w_{ij}^1 ceux de la couche numéro 1. Nous partons ainsi de la dernière couche. Afin d'obtenir les dérivées suivantes $\frac{d\,L}{w_{ij}^2}$ nous calculons la dérivée de la fonction de coût L par rapport à a_1^2, puis nous calculons la dérivée de a_1^2 par rapport à z_1^2 et au final la dérivée de z_1^2 par rapport à w_{ij}^2 . Nous remarquons que $\frac{d\,L}{w_{ij}^2} = \frac{d\,L}{a_1^2} * \frac{a_1^2}{z_1^2} * \frac{z_1^2}{w_{ij}^2}$. Ainsi, effectuer le produit de ces dérivées revient bien à calculer $\frac{d\,L}{w_{ij}^2}$. Cela étant fait, il suffit d'appliquer la formule de la descente pour en déduire les valeurs des poids ajustés. Pour calculer les dérivées de la première couche, nous réalisons les mêmes opérations, mais en allant encore plus en arrière. Nous repartons de la dérivée de a_1^2 par rapport à z_1^2, puis nous calculons la dérivée de z_1^2 par rapport à a_i^1 et nous poursuivons ce processus de dérivation en arrière jusqu'à arriver à la dérivée de z_i^1 par rapport à w_{ij}^1. A nouveau, en utilisant la formule de la descente de gradient, l'algorithme en déduit la valeur des poids w_{ij}^1 permettant de réduire les erreurs d'estimation.

En répétant plusieurs fois la phase de propagation en avant et celle de rétropropagation du gradient, le modèle parvient à optimiser les poids w_{ij}^1et w_{ij}^2 de manière à minimiser la fonction de coût, et ainsi améliorer son estimation de la variable cible.

Comme pour les deux modèles précédents, nous avons construit le réseau de neurones, puis nous avons modifié ces hyperparamètres afin d'accroître le pouvoir explicatif du modèle sur les données de validation. La méthode et les outils utilisés sont les mêmes que ceux appliqués

pour l'amélioration du modèle d'arbre de décision. Dans le cas du réseau neuronale, les hyperparamètres que nous avons considérés sont par exemple, le nombre de neurones dans les couches cachées et le nombre d'itérations de la phase de propagation en avant et de rétropropagation du gradient réalisé, afin de minimiser la fonction de coût. Après modification des hyperparamètres, le modèle de réseau de neurones affiche des performances plutôt satisfaisantes. Le coefficient de détermination R^2 sur les données de validation est de bien 69% pour l'Ethereum. Pour le Bitcoin 4,5% de la variance des ces rendements n'est pas expliquée par les facteurs explicatifs.

III. Bilan final, comparaison des modèles

Dans la partie précédente nous avons présenté, puis construit des modèles classiques, tels que le CPAM, mais aussi des modèles plus sophistiqués, propres au domaine « Machine Learning », comme le réseau de neurones ou le modèle de forêt aléatoire. Pour les modèles les plus compliqués, nous avons cherché systématiquement à modifier les hyperparamètres afin d'augmenter leur pouvoir explicatif sur les données de validation. Le « validation set » correspond à une période d'après début de crise, ce qui peut expliquer pourquoi les scores de performance ne dépassent pas le seuil de 70%, malgré la modification des hyperparamètres. Dans cette dernière partie, il s'agit d'évaluer l'ensemble des modèles sur les données de test afin de pouvoir les comparer et déterminer le modèle le plus performant. Nous rappelons que les données de test correspondent aux observations allant du 03/09/2021 au 26/04/2022. Cette période est plus contemporaine et semblable à celle correspondante aux données d'entrainement. De ce fait, nous nous attendons à des scores plus élevés. De plus, l'ensemble des modèles construits n'a jamais été confronté à ce jeu de données jusqu'à ici. Pour cette raison, il est pertinent de l'utiliser afin de comparer correctement la performance des différents modèles. Nous avons synthétisé dans le tableau suivant les performances des modèles construits, en les évaluant sur les données de test.

Modèle	Bitcoin	Ethereum
CPAM	74,06	78,54
Multifactoriel marché crypto	92,09	86,13
Régression linéaire	97,57	85,22
Arbre de décision	73,04	76,7
Foret aléatoire	80,91	82,23
Support Vector Machine	97,15	85,15
Réseau de neurones	97,26	84,00

Tableau 12 : Récapitulatif des R^2 en percentage des modèle sur les données de test

Nous remarquons que le pouvoir explicatif de l'ensemble des modèles évalués sur les données de test dépasse les 70%, ce qui est plutôt acceptable. Pour ce qui concerne le Bitcoin, les modèles de régression linéaire, de support vecteur machine et de réseau neuronale affichent des performances similaires et largement supérieures à celles des autres modèles. En effet, le R^2 pour ces modèles est compris entre 97% et 98%. Le modèle multifactoriel a un score inférieur à ces trois modèles mais reste tout de même très important (92%). De plus, ce modèle affiche également le meilleur coefficient de détermination pour l'Ethereum. Nous pouvons donc en déduire que les facteurs que nous avons construits sur le marché des cryptomonnaies, sont les plus explicatifs pour la construction d'un modèle d'évaluation du rendement d'une cryptomonnaie. Le fort pouvoir explicatif de ces facteurs sur les modèles peut se justifier par la structure du marché des cryptomonnaies. Premièrement, la taille du marché des cryptos est relativement faible par rapport à d'autres marchés tel que le marché d'actions. Deuxièmement, alors que les actions sont influencées par l'activité économique des entreprises, les cryptos sont principalement impactées par la spéculation, caractéristique commune à l'ensemble du marché des cryptomonnaies. Troisièmement, le Bitcoin a une représentation médiatique et une capitalisation majeure par rapport aux autres cryptos, phénomène de sur-représentation que l'on ne retrouve pas sur le marché des actions. Ainsi, si le Bitcoin subit une chute importante, nous pouvons envisager que l'ensemble des cryptos sera impacté. En ayant considéré douze cryptomonnaies pour construire nos facteurs (Bitcoin compris), ces effets sont accentués et permettent d'expliquer l'influence significative de ces facteurs. De plus, contrairement à notre conjecture initiale, nous constatons que le Bitcoin a des meilleurs scores que l'Ethereum, ce qui peut se justifier par le poids majeur du Bitcoin dans les facteurs.

Finalement, pour le rendement des deux cryptomonnaies que nous avons étudiées, il est aussi bien estimé par quatre modèles que nous jugeons être les modèles les plus performants de notre

analyse : le modèle Multifactoriel (malgré un moins bon score pour le Bitcoin), le modèle de régression linéaire avec sélection des variables, le modèle de support vecteur machine et le réseau de neurones. Ces modèles présentent un pouvoir explicatif très élevé malgré la présence de valeurs extrêmes et le traitement réservé au rendement du week-end pour les indices représentatifs du marché actions. La non-influence des valeurs extrêmes peut se justifier par le fait que le nombre de valeurs extrêmes, pour les rendements du Bitcoin et de l'Ethereum, est assez faible par rapport au nombre total de nos observations. Pour ce qui concerne le rendement du week-end des indices actions, nous rappelons que nous les avons mis à 0. Cette opération n'impacte pas la performance des modèles. En effet, cela peut se justifier par la moindre importance des indices actions dans l'évaluation du rendement des cryptos, par rapport aux autres facteurs.

Pour visualiser nos résultats, nous avons décidé d'afficher des graphiques montrant les prix réels des deux cryptomonnaies et les prix estimés par nos modèles. Le quatre modèles ayant des scores assez proches, nous reportons uniquement les graphiques pour le modèle de régression linéaire.

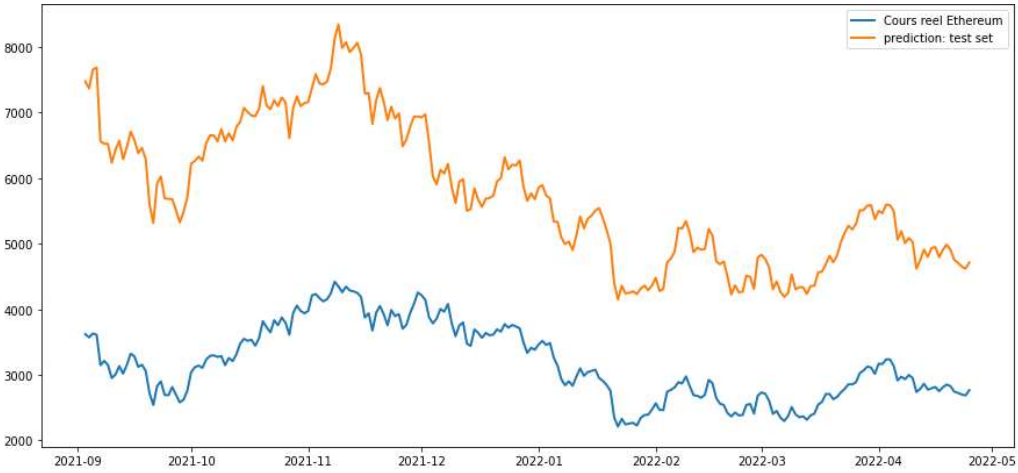

Figure 24 : Cours réel et estimé de l'Ethereum sur le test set

Figure 25 : Cours réel et estimé du Bitcoin sur le test set

La lecture des graphiques permet de bien confirmer et visualiser le différentiel de pouvoir explicatif des modèles évalués sur le Bitcoin et l'Ethereum.

Conclusion

Ce mémoire avait pour objet la construction des modèles d'évaluation des cryptomonnaies à l'aide du « Machine Learning ». Nous avons choisi de construire ces modèles afin d'estimer le rendement du Bitcoin, la cryptomonnaie la plus célèbre, mais aussi de l'Ethereum. Pour ce faire, nous avons commencé par construire une base de données. Cette base de données est constituée d'une part de facteurs propres au marché des cryptos (facteurs créés par nos soins) et d'autre part, par des facteurs inhérents à d'autres marchés (actions, commodities …). Nous avons ensuite dû établir des modèles de prédiction. Dans un premier temps, nous avons construit des modèles classiques de la littérature financière, tels que le modèle CPAM et le modèle multifactoriel, qui s'est avéré parmi les plus performants. Ensuite, après avoir sélectionné les variables les plus explicatives à l'aide d'outils économétriques, nous avons présenté, construit et testé nos modèles à l'aide d'outils informatiques. Dans une dernière partie, nous avons comparé la performance des modèles pour sélectionner finalement quatre modèles qui se prêtent relativement bien à l'évaluation des cryptomonnaies. Nous avons pu constater que les modèles les plus sophistiqués n'améliorent pas, voire font moins bien que des modèles que l'on peut qualifier de plus basiques.

Nous avons obtenu des résultats satisfaisants surtout pour le Bitcoin. Néanmoins, notre étude ne s'est pas portée sur l'ensemble des modèles qu'il est possible de construire en exploitant les techniques de Machine Learning. Il existe bien d'autres modèles comme par exemple les modèles dits de « boosting », très à la mode dans le domaine de l'intelligence artificielle en ce moment.

Références :

Livre : Madjid Khichane : *Le Machine Learning avec python, de la théorie à la pratique*

Série de vidéo YouTube : Chaine Machine Lernia. *Machine Learning avec Python*

Documentation Sickit Learn : https://scikit-learn.org/stable/

Introduction

cryptoast.fr*: Histoire, nature et philosophie de la crypto-monnaie.*
Lien : https://cryptoast.fr/crypto-monnaie-explication-definition/

presse-citron.net *: Comprendre la crypto-monnaie : qu'est-ce que c'est, comment investir ?*
Lien : https://www.presse-citron.net/crypto/faq/crypto-monnaie/

lebigdata.fr*: Machine Learning et Big Data : définition et explications*
Lien : https://www.lebigdata.fr/machine-learning-et-big-data

Construction des facteurs :

Huijbregts. Thèse : *"An Asset Pricing Model for Cryptocurrencies"*

Consultable dans le dossier bibliographie.

Statistiques de performance :

Meidum.com*: MAE, MSE, RMSE, Coefficient of Determination, Adjusted R Squared Which Metric is Better*
Lien : https://medium.com/analytics-vidhya/mae-mse-rmse-coefficient-of-determination-adjusted-r-squared-which-metric-is-better-cd0326a5697e

CPAM :

investopedia.com*: Capital Asset Pricing Model (CAPM)*
Lien : https://www.investopedia.com/terms/c/capm.asp

journaldunet.fr*: CAPM (Capital asset pricing model) : définition simple, formule et traduction*
Lien : https://www.journaldunet.fr/business/dictionnaire-comptable-et-fiscal/1445102-capm-capital-asset-pricing-model-definition-simple-calcul-et-traduction/

Multifactoriel :

cmcmarkets.com*: Modèles multifactoriels : 3 facteurs, 4, 6 ou davantage ?*
Lien : https://www.cmcmarkets.com/fr-fr/actualites-et-analyses/modeles-multifactoriels-3-facteurs-4-6-ou-davantage

Descente de gradient :

machinelearnia.com*: La Descente de Gradient, qu'est-ce-que c'est ?*
Lien : https://machinelearnia.com/descente-de-gradient/#:~:text=La%20Descente%20de%20Gradient%20est,au%20centre%20un%20minimum%20global.

Arbre de décision :

fr.wikipedia.org *: Arbre de décision (apprentissage)*
Lien :
https://fr.wikipedia.org/wiki/Arbre_de_d%C3%A9cision_(apprentissage)#:~:text=L'apprentissage%20par%20arbre%20de%20d%C3%A9cision%20est%20une%20m%C3%A9thode%20classique,de%20plusieurs%20variables%20d'entr%C3%A9e

python-course.eu *: Regression Trees in Python*
Lien : https://python-course.eu/machine-learning/regression-trees-in-python.php

youtube.com*: Decision Tree Regression Clearly Explained!*
Lien : https://www.youtube.com/watch?v=UhY5vPfQIrA

Random Forest :

stat4decision.com*: Forêt aléatoire avec python et scikit-learn*
Lien : https://www.stat4decision.com/fr/foret-aleatoire-avec-python/

larevueia.fr*: Random Forest*
Lien : https://larevueia.fr/random-forest/

machinelearningmastery.com *: Random Forest for Time Series Forecasting*
Lien : https://machinelearningmastery.com/random-forest-for-time-series-forecasting/

Livre : Madjid Khichane : *Le Machine Learning avec python, de la théorie à la pratique*
Chapitre 4.7

Support Vector Machine :

medium.com*: Support Vector Regression (SVR) Model: A Regression-Based Machine Learning Approach*
Lien :https://medium.com/analytics-vidhya/support-vector-regression-svr-model-a-regression-based-machine-learning-approach-f4641670c5bb

towardsdatascience.com*: An Introduction to Support Vector Regression (SVR)*
Lien : https://towardsdatascience.com/an-introduction-to-support-vector-regression-svr-a3ebc1672c2

Réseau de neurones :

youtube.com*: réseau de neurones (2 couches)*
Lien : https://www.youtube.com/watch?v=YMP-lU-xqyc&t=1261s