Rapport - TP Machine Learning

Présentation de la problématique

Objectif

Notre objectif, à travers ce TP, est de pouvoir être capable de prédire le cours du Bitcoin dans le temps.

Pour cela, nous allons utiliser un dataset disponible ici : https://www.kaggle.com/mczielinski/bitcoin-historical-data

Ce dataset comprend plus de 4 millions de données sur le Bitcoin amassées entre Janvier 2012 et Décembre 2020. On peut retrouver dans ce dataset les données suivantes :

- **Timestamp :** Valeur temporelle associée aux données. Représente le nombre de secondes qui se sont écoulées depuis le 1er janvier 1970 (Unix Epoch)
- **Open :** Prix d'ouverture du Bitcoin au temps T
- **High :** Prix le plus élevé entre T et T+1
- **Low :** Prix le plus bas entre T et T+1
- **Close :** Prix de fermeture à T+1
- **Volume (BTC) transacted :** Volume de Bitcoin échangé durant la période entre T et T+1
- **Volume ($) transacted :** Volume de $ échangé durant la période entre T et T+1
- **Average Price ($) :** Prix moyen du Bitcoin entre T et T+1

Afin d'atteindre notre objectif, nous souhaitons, à partir des données au temps T, être capable de prédire le prix le plus élevé (colonne High) que va atteindre le Bitcoin à T+1.

Analyse du dataset

Dans le dataset, on retrouve plusieurs lignes qui contiennent des données incomplètes (notamment début 2012). Nous avons donc choisi de retirer ces lignes qui pourraient impacter la qualité de notre prédicteur.

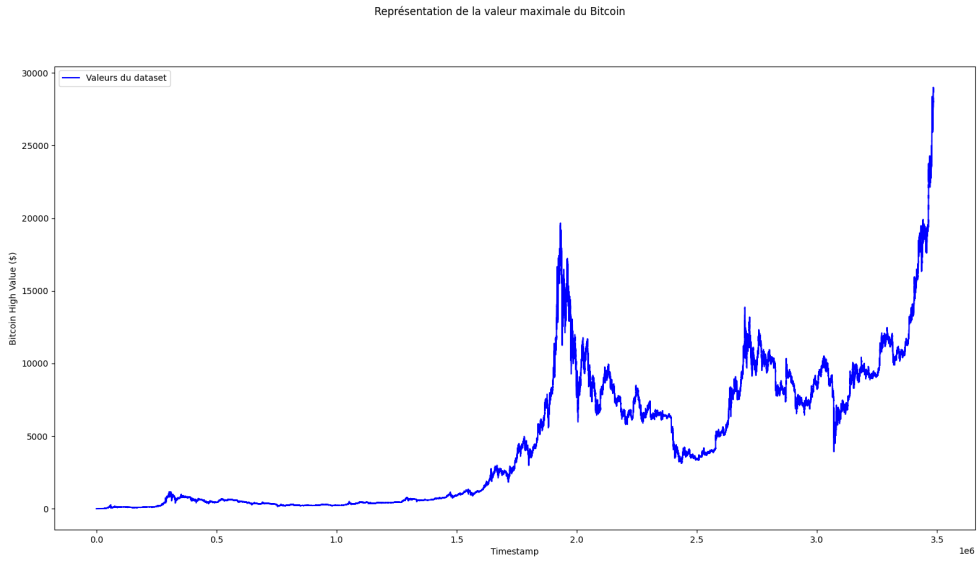
On se retrouve donc avec le dataset suivant :

Description du dataset

Dataset description :

	Timestamp	Open	High	Low	Close	Volume_(BTC)	Volume_(Currency)	Weighted_Price
count	3.484305e+06	3484305.000	3484305.000	3484305.000	3484305.000	3484305.000	3.484305e+06	3484305.000
mean	1.493612e+09	4570.658	4573.721	4567.413	4570.624	9.423	3.315691e+04	4570.625
std	7.234420e+07	4775.105	4778.620	4771.352	4775.094	31.013	1.129107e+05	4775.090
min	1.325318e+09	3.800	3.800	1.500	1.500	0.000	0.000000e+00	3.800
25%	1.430738e+09	432.190	432.430	432.000	432.200	0.392	4.147940e+02	432.175
50%	1.501915e+09	3085.010	3093.250	3078.010	3084.410	1.937	3.305199e+03	3085.478
75%	1.556349e+09	8158.010	8162.850	8152.790	8157.310	7.300	2.158297e+04	8157.964
max	1.609373e+09	28997.380	29010.320	28963.660	28997.380	5853.852	1.044599e+07	28985.886

Représentation de la valeur maximale du Bitcoin



On vois donc, dans cette représentation, l'objectif de prédiction que l'on souhaite obtenir.

Définition du problème de Machine Learning

Quel est notre objectif ?

L'objectif est de mettre en place un modèle prédictif capable de prédire le prix maximum que va atteindre le Bitcoin à T+1 en se basant sur les données au temps T. Pour cela, nous avons presque 4 millions de données, ce qui nous semble suffisant pour mettre en place notre modèle.

Notre problème peut-il être exprimé comme un problème d'apprentissage automatique ?

Nous pouvons organiser notre problème comme un problème d'apprentissage automatique en utilisant le cadre supervisé, et plus précisément en utilisant un problème de régression.

Les données utilisées vont-elles changer au cours du temps ?

Non, ces données ne changeront pas avec le temps car elles représentent des valeurs immuables.

Comment allons nous évaluer les performances de notre modèle ?

Pour cela, nous allons utiliser des métriques tels que k-fold cross, calcul du R^2, MAE, ME et également le traçage de la prédiction et la comparaison visuelle avec ce qui est attendu.

Mise en place de modèles

Gaussian Process Regressor

La régression du processus gaussien est une approche bayésienne puissante et non paramétrique des problèmes de régression, qui peut être utilisée dans des scénarios d'exploration et d'exploitation.

```
Training score: 0.5371537545575811
Test score: 0.536790104509991
```

```
K-Fold cross scores :
 [-71.47857366  -3.6063671   -3.26479682  -7.15979169  -3.66764475]
Mean Absolute Error : 1016.8972573215261
Maximum error : 2192.553251484893
```

Dans un résultat parfait, la valeur du R^2 devrait être aux alentours de 1. Or, on remarque que la valeur du R^2 sur le jeu de test est d'environ $0,54$ ce qui n'est pas un bon résultat.

De plus, les valeurs de $Mean\ Square\ Error$ retournées pendant le k-fold cross validation ne sont pas bonnes non plus car très éloignées de 0.

Le $Mean\ Absolute\ Error$ qui représente l'erreur moyenne est à plus de 1 000$ d'écart entre ce qui est attendu et ce qui est prédit.

L'erreur maximale ($Maximum\ Error$) n'est elle aussi pas concluante car nous sommes à plus de 2 000$ d'écart au maximum lors d'une prédiction.

On retrouve ces résultats peu encourageants lorsque l'on compare les résultats obtenus par rapport aux données initiales. On se rend alors compte que le modèle n'est pas utilisable en l'état car beaucoup trop imprécis.

Attention : Pour des raisons de temps, nous avons dû limiter le nombre de données utilisées pour ce modèle au vu des ressources nécessaires à son bon fonctionnement. Nous n'avons donc utilisé que 10 000 données afin de permettre à notre modèle de s'entraîner. La conclusion de ce rapport se basera donc sur les résultats obtenus avec le jeu de données utilisé.

Support Vector Regression

Le SVR nous donne la possibilité de définir le niveau d'erreur acceptable dans notre modèle et de trouver une ligne appropriée (ou un hyperplan dans les dimensions supérieures) pour ajuster les données.

Contrairement aux MCO, la fonction objective du SVR est de minimiser les coefficients - plus précisément, la norme l2 du vecteur de coefficient - et non l'erreur quadratique. Le terme d'erreur est plutôt traité dans les contraintes, où nous fixons l'erreur absolue inférieure ou égale à une marge spécifiée, appelée l'erreur maximale, ϵ (epsilon). Nous pouvons régler epsilon pour obtenir la précision souhaitée de notre modèle.

Dans notre cas, nous allons utiliser les paramètres suivants pour notre modèle :

```
SVR(kernel='poly', C=1.0, epsilon=0.2)
```

```
------
Support Vector Regression model :

Training score: 0.7930045406135825
Test score: 0.7907189675950895
k-kold cross with 5 sections :

K-Fold cross scores :
 [-318.80606176   -8.53938638   -3.47066956   -9.45979284  -21.20145497]
Mean Absolute Error : 585.4422241312659
Maximum error : 4095.4630851416478
```

Avec le modèle SVR, on remarque que la valeur du R^2 sur le jeu de test est d'environ $0,79$. Ce résultat est meilleur que lors de l'utilisation de la régression du processus gaussien.

Cependant, les valeurs de $Mean\ Square\ Error$ retournées pendant le k-fold cross validation sont bien moins bonnes.

Le $Mean\ Absolute\ Error$ qui représente l'erreur moyenne est à plus de 585$ d'écart entre ce qui est attendu et ce qui est prédit. On a ici divisé par 2 l'erreur moyenne.

L'erreur maximale ($Maximum\ Error$) est elle bien plus importante que précédemment. On a ici une erreur maximale de 4 000$

Lorsque l'on compare les données souhaités et les données prédites, on remarque que le profil de la courbe prédite correspond bien plus à ce qui est visé. Cependant, des erreurs trop importantes sont notables. Nous avons donc décidé de ne pas nous arrêter sur ce modèle car les erreurs possibles sont trop importantes.

Attention : Pour des raisons de temps, nous avons dû limiter le nombre de données utilisées pour ce modèle au vu des ressources nécessaires à son bon fonctionnement. Nous n'avons donc utilisé que 10 000 données afin de permettre à notre modèle de s'entraîner. La conclusion de ce rapport se basera donc sur les résultats obtenus avec le jeu de données utilisé.

Linear Regression

La régression linéaire est une approche linéaire de la modélisation de la relation entre une réponse scalaire et une ou plusieurs variables explicatives (également appelées variables dépendantes et indépendantes).

Dans la régression linéaire, les relations sont modélisées à l'aide de fonctions prédicteurs linéaires dont les paramètres inconnus du modèle sont estimés à partir des données. Ces modèles sont appelés modèles linéaires.

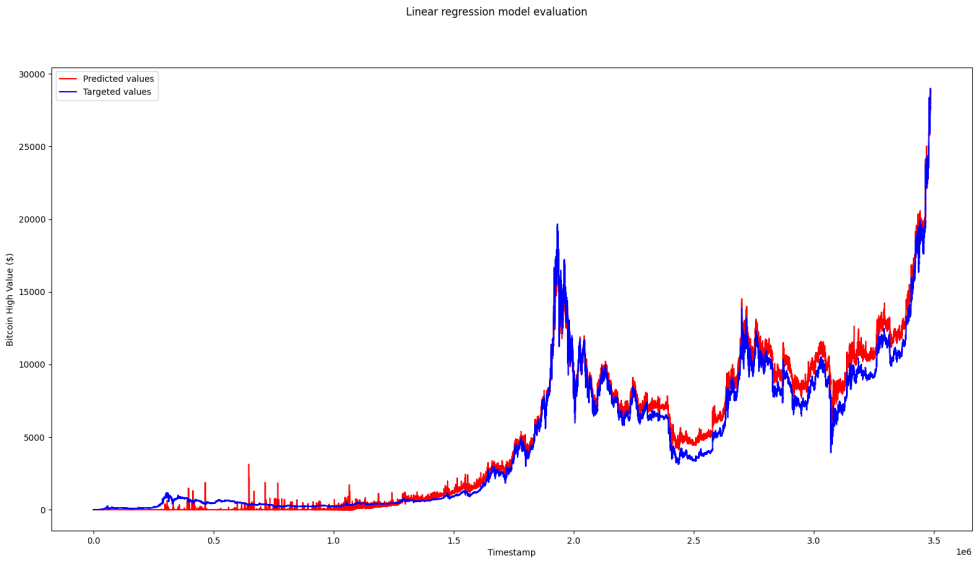
```
------
LinearRegression model :

Training score: 0.9642231136104561
Test score: 0.9641005154716021
k-kold cross with 5 sections :

K-Fold cross scores :
 [0.99996244 0.99997447 0.99999368 0.99999236 0.9999949 ]
Mean Absolute Error : 518.1291760837054
Maximum error : 2912.9062558169535
```

En utilisant le modèle Linear Regression, on remarque que la valeur du R^2 est bien meilleure avec une valeur de $0,96$.

De plus, les valeurs de $Mean\ Square\ Error$ retournés pendant le k-fold cross validation sont très bonnes car très proche de 1.

Le $Mean\ Absolute\ Error$ qui représente l'erreur moyenne est à plus de 518$ d'écart entre ce qui est attendu et ce qui est prédit. On est très proche des derniers résultats obtenus avec le SVR.

L'erreur maximale ($Maximum\ Error$) n'est pas très élevée. En effet, on est proche du résultat obtenu avec la régression du processus gaussien.



Côté profil de courbe, on commence à se rapprocher d'une courbe très proche de celle souhaitée. Les résultats sont donc très concluants avec ce modèle car, bien que des erreurs soient bien visibles, le profil de courbe est quasiment identique.

Nous pensons donc que ce modèle est utilisable. Mais nous avons tout de même cherché à améliorer nos résultats avec un nouveau modèle.

SGD Regressor

La descente stochastique par gradient (SGD) est une approche simple mais très efficace pour ajuster les régresseurs linéaires sous des fonctions de perte convexes, telles que les machines à vecteurs de soutien (linéaires) et la régression logistique. Bien que la SGD existe depuis longtemps dans la communauté de l'apprentissage machine, elle a reçu une attention considérable tout récemment dans le contexte de l'apprentissage à grande échelle.

C'est un modèle souvent utilisé lors d'apprentissages sur d'importantes données.

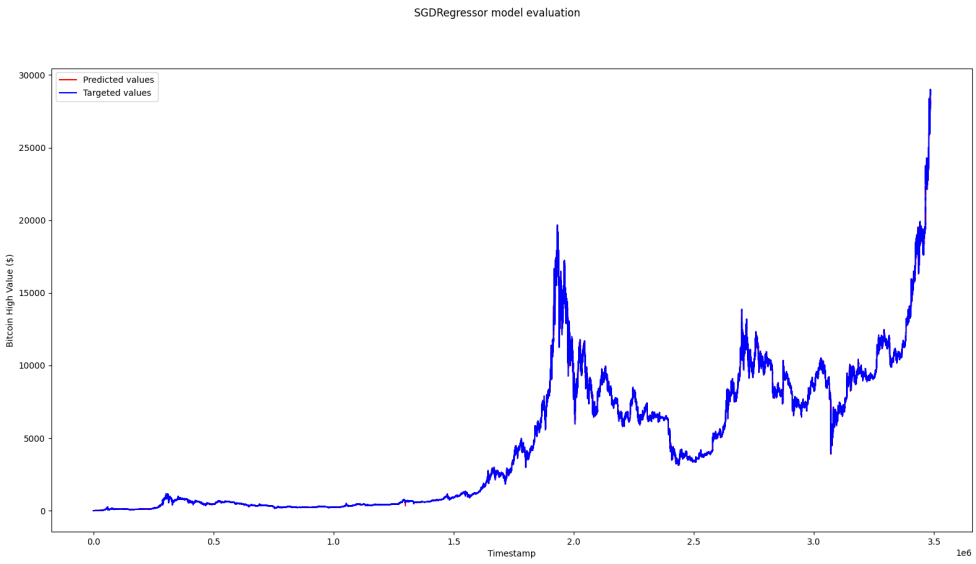
```
------
SGDRegressor model :

Training score: 0.999997356435299
Test score: 0.9999973362988467
k-kold cross with 5 sections :

K-Fold cross scores :
 [0.99988298 0.99997385 0.9999921  0.99999009 0.99999279]
Mean Absolute Error : 2.0727596293902093
Maximum error : 429.53699299157233
```

En utilisant le modèle SGD Regressor, on remarque que la valeur du R^2 est la meilleure parmi les modèles testés. On obtiens une valeur de 1 à 10^{-6} près.

De plus, les valeurs de $Mean\ Square\ Error$ retournées pendant le k-fold cross validation sont également très bonnes car très proche de 1.

Le $Mean\ Absolute\ Error$ qui représente l'erreur moyenne est à seulement 2$. On a donc une erreur en moyenne presque nulle.

L'erreur maximale ($Maximum\ Error$) est bluffante par rapport aux autres modèles. On a une erreur maximale de 429$.



Quand on porte attention aux courbes, on se rend compte que le modèle arrive à prédire presque parfaitement les valeurs maximales du Bitcoin à chaque fois. Les profils de courbes sont presque superposés, ce qui indique des résultats très intéressants.

Conclusion

Au vue des différents tests effectués, nous avons constaté que le modèle **SGD Regressor** est de loin le plus intéressant et le plus performant. Grâce à ce modèle, nous sommes certain de réduire au maximum les erreurs possibles que peut faire notre prédicteur.