DSIA_5102B

Time series and Machine Learning

RÉALISÉ PAR

Faissal KOUTTI

Benjamin NAHUM

SOMMAIRE

- Problème abordé, nature des données
- Questions posées et Features
- Méthodes utilisées, résultats obtenus
- Enseignements (Performances, Implantation, Difficultés rencontrées, Perspectives)

PROBLÈME ABORDÉ
NATURE DES DONNÉES

Etant intéressés par le monde de la finance et ce que nous pouvons y apporter à l'aide du machine learning, nous avons cherché un jeu de données nous permettant d'exploiter différents types de données temporelles. Ainsi, en appliquant nos connaissances personnelles et en utilisant des méthodes de prédictions variés, nous espérerons alors obtenir les prédictions les plus précises.

Notre jeu de données, obtenu sur Kaggle, renseigne des valeurs journalières sur les plus grande bourses mondiales comme des données indiquant des fluctuations, le montant le plus élevé atteint, le plus bas ou encore le volume. Bien sûr, toutes ces données structurées sont temporelles, et observées à partir d'une date précise, comme le 31 décembre 1965 pour la célèbre New York Stock Exchange.

Durant ce projet, nous aurons pour but de prédire l'évolution des plus grandes bourses mondiales et utilisant plusieurs algorithmes variés. Une fois les étapes de feature engineering terminées ainsi que l'application des modèles visés, nous comparerons alors les résultats obtenus en évaluant les taux d'erreurs obtenus et en analysant les prédictions obtenues directement sur les graphiques.

QUESTIONS POSÉES ET FEATURES

Afin d'obtenir les prédictions les plus réalistes, il nous est nécessaire de comprendre la signification des données présentes à travers les étapes de feature engineering.

En effet, nous chercherons à supprimer, ajouter, modifier ou amplifier certaines données afin d'obtenir le forecasting le plus cohérent. Lorsque nous aurons exploité au maximum les datasets en de multiples dataframes, nous dressons alors diverses études nous permettant d'étayer notre étude.

Nous tenterons alors de répondre à différentes questions:

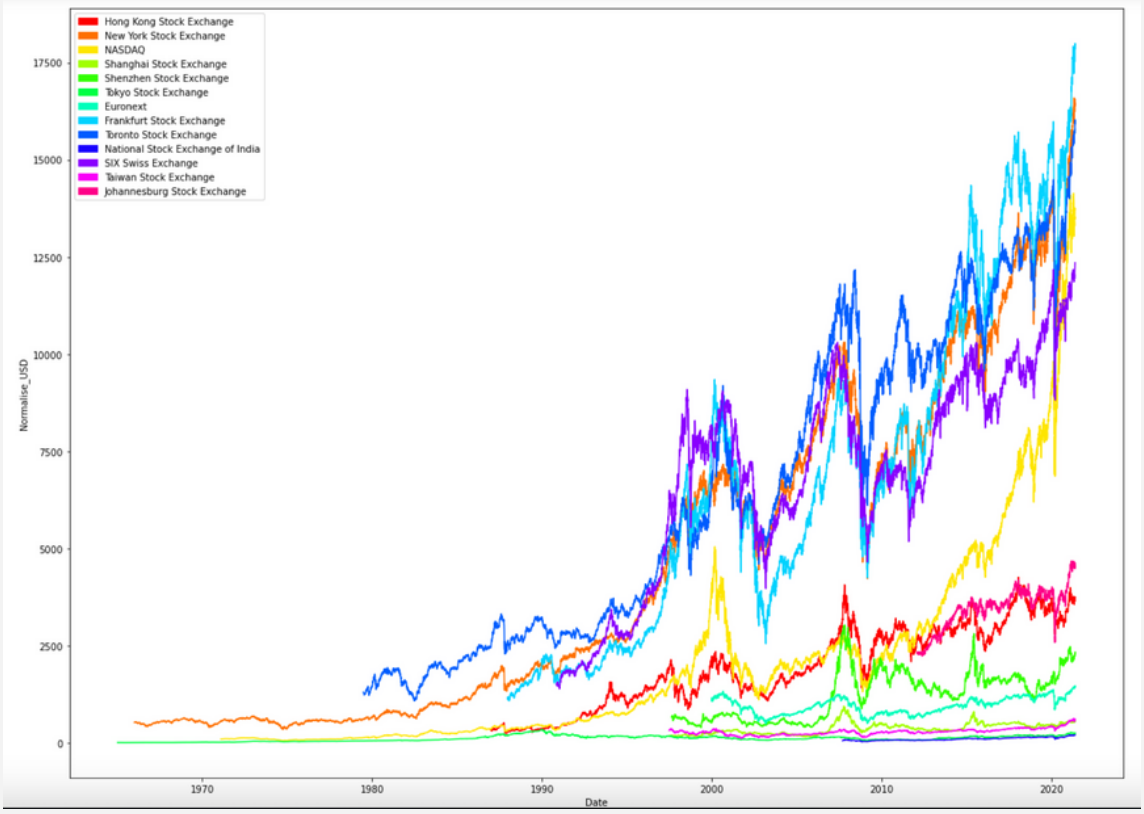
- Quels sont les étapes phares du pré-processing et feature engineering ? Quelles sont les ajouts et suppressions qui en résultent ?
- Nous réaliserons ensuite différentes études annexes nous permettant de mieux cerner et interpréter les données et d'étayer les futures prédictions.

Nous étudierons par la suite les prédiction à partir de l'algorithme Sarimax et un réseau de neurones de type RNN.

L'objectif sera alors de les comparer et expliquer ces résultats.

QUESTIONS POSÉES ET FEATURES

Différentes transformations ont requis notre attention dans cette partie.
Premièrement, toutes les données boursières mises à disposition sont fournies dans leur propre monnaies. Ce qui n'est pas un problème lors d'une étude sur les variations sur cette même bourse. En revanche, pour effectuer des corrélations entre les différentes bourses, il nous est préférable de tout convertir selon une monnaie commune. Ainsi, nous avons décidé de créer une nouvelle colonne nommées "normalise_USD". En ajoutant les taux de conversion actuels des devises des différentes bourses en dollar américain, la nouvelle colonne nous permettra de comparer les données boursières mondiales.

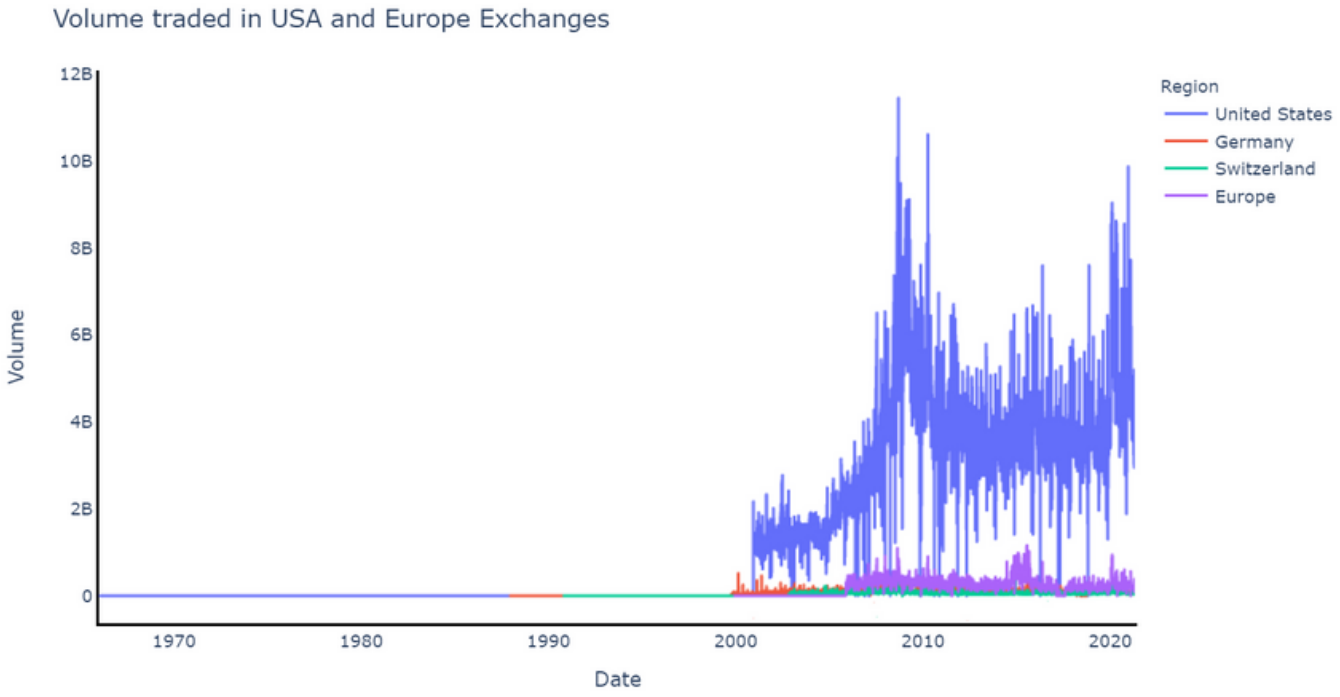


Nous avons également utilisé MinMaxScaler() afin de normaliser les données dans l'optique de favoriser le traitement des données de nos différents modèles de prédictions.
Un ajout d'une colonne représentant la moyenne du prix d'entrée et de sortie, nous permettant d'analyser les fluctuations journalières de chaque Bourse.
Par la suite, de nombreuses dataframes normalisées seront crées afin de dresser des études spécifiques comme l'analyse de la volatilité ou le volume échangé des différentes bourses.

QUESTIONS POSÉES ET FEATURES

Les features disponibles dans le jeu de données nous permettent d'effectuer diverses études, et d'en visualiser un contenu intéressant. Par exemple, dans le premier graphique, nous comparons le volume boursier des deux plus grands pôles, américain et européen.

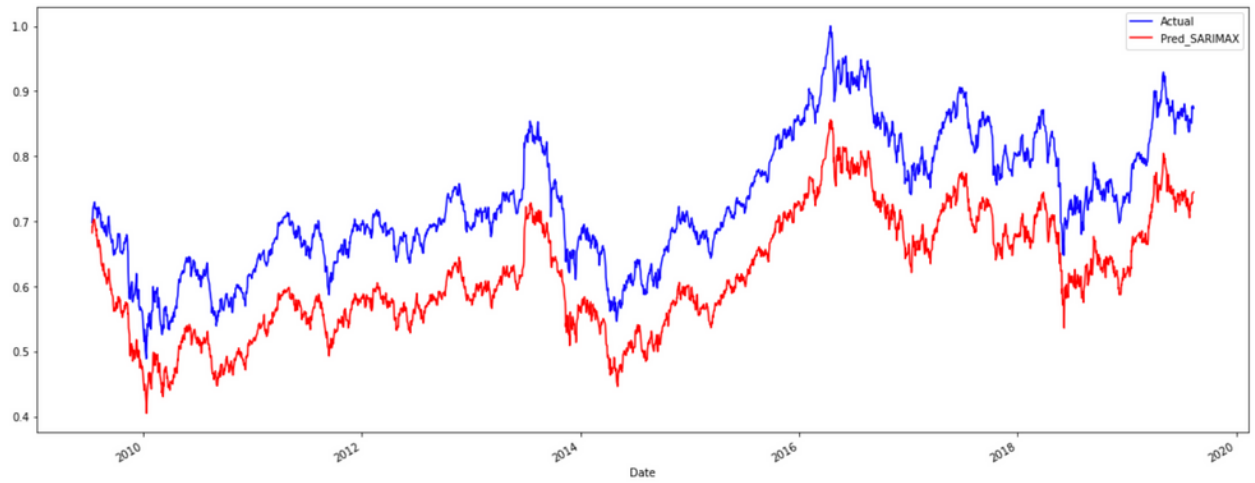
On remarque alors que les célèbres New York Stock Exchange et Nasdaq américains surplombent les marchés européens



MÉTHODES UTILISÉES, RÉSULTATS OBTENUS

Prédiction sur notre jeu de données avec le modèle Sarimax

```
Entrée [588]: predictions["Actual"].plot(figsize=(20,8), legend=True, color="blue")
              predictions["Pred_SARIMAX"].plot(legend=True, color="red", figsize=(20,8))
Out[588]: <AxesSubplot:xlabel='Date'>
```




```
Entrée [593]: from statsmodels.tools.eval_measures import rmse
              error=rmse(predictions['Pred_SARIMAX'], predictions['Actual'])
              error
Out[593]: 0.11385790036587945
```

Nous avons obtenu un score de rmse de 0.11 qui correspond à un bon score. Après avoir effectué un Auto-Arima, le choix d'un Sarimax avec pour paramètres (2,1,0) pour l'ordre semblait être le modèle qui allait nous donner les résultats les plus concluants.

Out[586]:

Date	Pred_SARIMAX	Actual
2009-07-14	0.682575	0.698500
2009-07-15	0.692431	0.704356
2009-07-16	0.692082	0.707923
2009-07-17	0.700448	0.719861
2009-07-20	0.701493	0.725119
...	...	...
2019-08-06	0.727668	0.862372
2019-08-07	0.740158	0.876135
2019-08-08	0.743345	0.873362
2019-08-09	0.743437	0.877361
2019-08-12	0.744932	0.873765

2491 rows × 2 columns

Voici la DataFrame que l'on obtient avec les valeurs de départ et les valeurs prédisent. Notre modèle effectue une prédiction assez proche de la réalité.

MÉTHODES UTILISÉES, RÉSULTATS OBTENUS

```
step_wise = auto_arima(train_y, exogenous= train_X, start_p=1, start_q=1,max_p=7, max_q=7,d=1, max_d=7,trace=True, error_action='ignore', suppress_warnings=True, stepwise=True)
```

```
Performing stepwise search to minimize aic
 ARIMA(1,1,1)(0,0,0)[0] intercept   : AIC=-43744.132, Time=3.63 sec
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=-43679.761, Time=2.83 sec
 ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=-43735.051, Time=2.90 sec
 ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=-43728.836, Time=2.93 sec
 ARIMA(0,1,0)(0,0,0)[0]             : AIC=-39777.221, Time=6.08 sec
 ARIMA(2,1,1)(0,0,0)[0] intercept   : AIC=-43747.486, Time=3.49 sec
 ARIMA(2,1,0)(0,0,0)[0] intercept   : AIC=-43749.348, Time=3.33 sec
 ARIMA(3,1,0)(0,0,0)[0] intercept   : AIC=-43748.704, Time=3.62 sec
 ARIMA(3,1,1)(0,0,0)[0] intercept   : AIC=-43746.613, Time=3.86 sec
 ARIMA(2,1,0)(0,0,0)[0]             : AIC=-43749.944, Time=3.23 sec
 ARIMA(1,1,0)(0,0,0)[0]             : AIC=-43735.488, Time=2.66 sec
 ARIMA(3,1,0)(0,0,0)[0]             : AIC=-43749.342, Time=3.23 sec
 ARIMA(2,1,1)(0,0,0)[0]             : AIC=-43748.303, Time=3.28 sec
 ARIMA(1,1,1)(0,0,0)[0]             : AIC=-43744.734, Time=3.25 sec
 ARIMA(3,1,1)(0,0,0)[0]             : AIC=-43747.255, Time=3.99 sec

Best model:  ARIMA(2,1,0)(0,0,0)[0]
Total fit time: 52.414 seconds
```

SARIMAX Results

Dep. Variable:	y	**No. Observations:**	5814
Model:	SARIMAX(2, 1, 0)	**Log Likelihood**	21884.972
Date:	Sun, 07 Nov 2021	**AIC**	-43749.944
Time:	22:13:25	**BIC**	-43683.265
Sample:	0	**HQIC**	-43726.751
	- 5814		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
Open	0.1644	0.013	12.650	0.000	0.139	0.190
High	1.341e+12	0.012	1.13e+14	0.000	1.34e+12	1.34e+12
Low	1.318e+12	0.012	1.12e+14	0.000	1.32e+12	1.32e+12
Close	-5.011e+07	0.006	-8.55e+09	0.000	-5.01e+07	-5.01e+07
Volume	0.0103	0.001	9.731	0.000	0.008	0.012
Adj Close	5.011e+07	0.006	8.55e+09	0.000	5.01e+07	5.01e+07
Mean	-2.659e+12	0.007	-3.73e+14	0.000	-2.66e+12	-2.66e+12
ar.L1	-0.0933	0.015	-6.221	0.000	-0.123	-0.064
ar.L2	0.0536	0.008	7.136	0.000	0.039	0.068
sigma2	3.143e-05	2.43e-07	129.086	0.000	3.1e-05	3.19e-05

Ljung-Box (L1) (Q):	0.01	**Jarque-Bera (JB):**	33483.64
Prob(Q):	0.92	**Prob(JB):**	0.00
Heteroskedasticity (H):	11.93	**Skew:**	-0.26
Prob(H) (two-sided):	0.00	**Kurtosis:**	14.75

L'auto ARIMA prend en compte les valeurs AIC et BIC générées pour déterminer la meilleure combinaison de paramètres. Les valeurs AIC (Akaike Information Criterion) et BIC (Bayesian Information Criterion) sont des estimateurs permettant de comparer les modèles. Plus ces valeurs sont faibles, plus le modèle est bon.

Après avoir effectué un Auto-Arima, le meilleur modèle que l'on puisse utiliser sur notre dataset est un sarimax avec pour paramètres (2,1,0).

MÉTHODES UTILISÉES, RÉSULTATS OBTENUS

Prédiction sur notre jeu de données avec le modèle RNN

```
modelo_hongkong = modelo(x_train_hongkong, y_train_hongkong)
```

```
Epoch 1/3
6734/6734 [==============================] - 260s 38ms/step - loss: 0.0030
Epoch 2/3
6734/6734 [==============================] - 223s 33ms/step - loss: 4.0558e-04
Epoch 3/3
6734/6734 [==============================] - 220s 33ms/step - loss: 3.3934e-04
```

Nous avons opté pour un réseau de type RNN avec 2 couches LSTM. En entraînant le modèle, nous obtenons une belle prédiction, obtenue en vert, par rapport aux résultats réels en orange. Ces résultats ont été obtenus en entraînant le modèle selon les mêmes features que précédemment, mais avant 2016, pour une prédiction de 2016 à aujourd'hui.

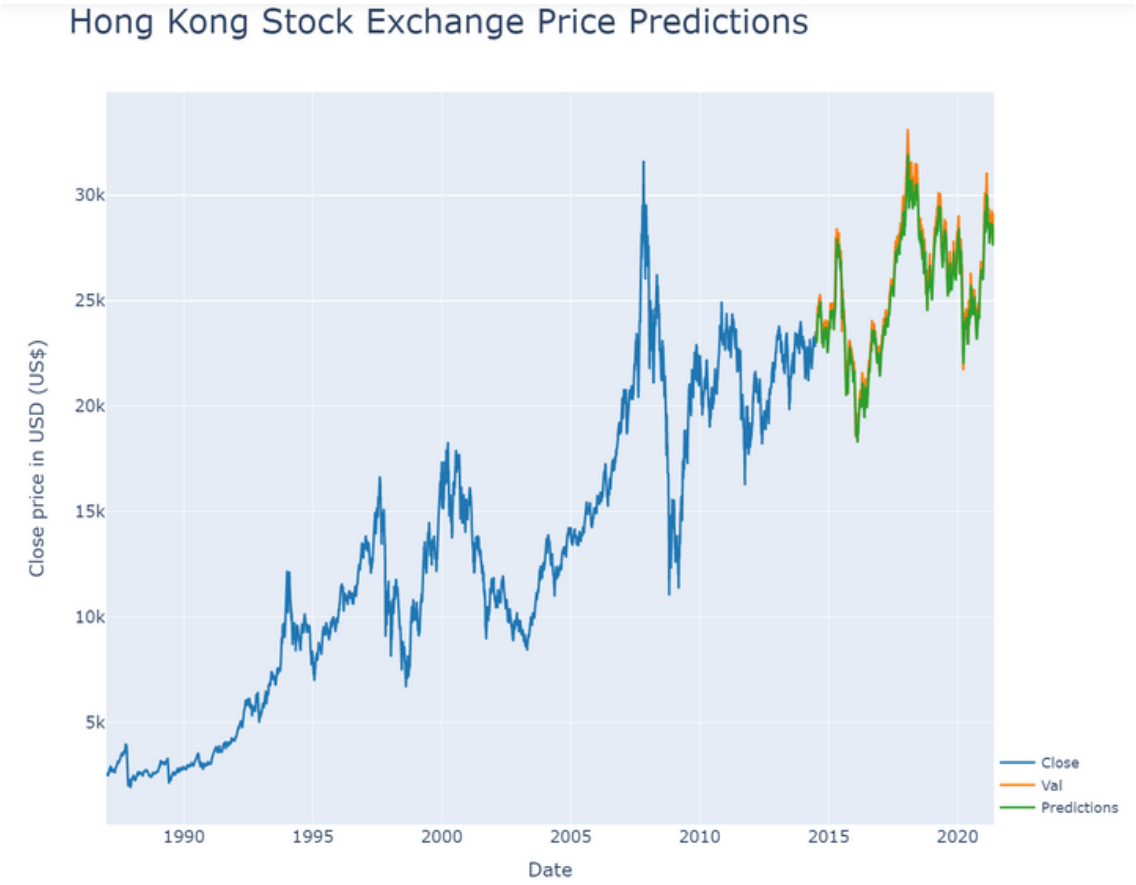
```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_10 (LSTM)               (None, 60, 50)            10400
_________________________________________________________________
dropout_2 (Dropout)          (None, 60, 50)            0
_________________________________________________________________
lstm_11 (LSTM)               (None, 50)                20200
_________________________________________________________________
dense_7 (Dense)              (None, 25)                1275
_________________________________________________________________
dense_8 (Dense)              (None, 1)                 26
=================================================================
Total params: 31,901
Trainable params: 31,901
Non-trainable params: 0
_________________________________________________________________
```

Dans notre réseau de neurones, nous avons choisi 3 types de couches: LSTM, Dropout et Dense, pour un total de 5 couches.
Nous nous sommes limités à ce nombre car ce sont les couches les plus essentielles au vu de notre entraînement, et il faut garder un temps d'exécution raisonnable en limitant ces layers.

Nous avons alors deux couches de LSTM (Long Short Term Memory), type de RNN performant divisant la donnée à court et à long terme.

Nous avons ajouté une couche de Dropout désactivant aléatoirement un certain pourcentage de neurones dans une couche à chaque itération afin d'empêcher l'overfitting, et enfin les couches Dense.

MÉTHODES UTILISÉES, RÉSULTATS




```
from statsmodels.tools.eval_measures import rmse
error=rmse(predictions_hongkong, y_test_hongkong)
error
```

```
array([0.03826654])
```

En analysant la rmse, nous nous rendons compte que nous obtenons un score plus faible pour le modèle RNN que pour le modèle Sarimax.

Ainsi, la prédiction s'avère plus efficace pour le modèle Sarimax.
Cependant, l'entraînement pour le modèle RNN s'est réalisé que sur 3 epochs, compte tenu du temps beaucoup plus conséquent conséquent.

En comparant les temps d'exécutions, nous remarquons que nous avons 52 secondes pour le modèle Sarimax, et 422 secondes pour le réseau de neurones.

De plus, en réalisant l'entraînement sur une seul epoch, la prédiction est plus faible, et est moins réaliste pour plus de 3 epochs. Nous pouvons donc en conclure que pour notre étude et les contraintes de temps, le modèle RNN est meilleur pour une prédiction à long terme termes avec un entraînement plus robuste, que le modèle Sarimax.

ENSEIGNEMENTS (PERFORMANCES, IMPLANTATION, DIFFICULTÉS RENCONTRÉES, PERSPECTIVES)

Ce projet nous as permis à la fois d'allié nos compétences en Machine Learning mais aussi en Deep Learning. Ce projet nous as aussi permis de travailler sur un cas de finance, un domaine qui nous intéresse particulièrement. Nous avons pu mettre en oeuvre deux modèles un Sarimax et un RNN pour prédire la valeur des différentes bourses au fil des années.

La difficulté majeure rencontrée était la compréhension des modèles Sarimax, RNN le modèle rattaché au Deep Learning et leurs fonctionnement.

La notion de time series était assez nouveau pour nous, nous n'avons jamais travaillé sur des projets composé de données qui évolue au fur et à mesure du temps. Il fallait pour ce faire de s'adapter a ces nouvelles notions.

Ce projet nous as permis de consolider nos compétences en Machine Learning et nous as apporté des connaissances supplémentaires sur la finance grâce à la notion de Time Series.