Université Abdelmalek Essaâdi

Faculté des Sciences et Techniques-Tanger

Département Génie Informatique

Réseaux Avancés

LSTM

Réalisé Par :

Yossra safi chetouan

Kaddour Ferdaous

Encadré par :

Mohammed BOUHORM

Plan de Rapport

- Introduction
- Objectif du Projet
- LSTM ET RNN
- Fonctionnement d'un LSTM
- Architecture d'un LSTM
- Avantages des LSTM
- Exemples d'utilisation d'un LSTM
- Limitations d'un LSTM
- Application d'un LSTM (Netflix Stock prédiction)
- Conclusion

1.Introduction

L'intégration de modèles LSTM (Long Short-Term Memory) marque une avancée significative dans le paysage de l'apprentissage automatique, spécifiquement dans le contexte de l'analyse de séquences temporelles. Les LSTM représentent une évolution des réseaux de neurones récurrents (RNN), conçus pour aborder des problèmes où la capture des dépendances à long terme au sein des données est essentielle. Leur émergence a été particulièrement marquée par leur efficacité dans la modélisation et la prédiction de séquences temporelles complexes, qu'il s'agisse de données financières, météorologiques ou de textes séquentiels. Cette adoption croissante atteste de la puissance des LSTM à relever des défis préalablement complexes dans divers domaines d'application.

2. Objectif du Projet

Ce projet impliquant l'utilisation de modèles LSTM a pour objectif de développer un système capable de comprendre et de prédire des tendances, des motifs ou des comportements dans des données séquentielles. Selon le domaine d'application, cela peut inclure la prédiction de valeurs futures, la détection d'anomalies, la classification de séquences, ou d'autres tâches liées aux séries temporelles.

3.LSTM ET RNN

Les RNN sont une classe de réseaux neuronaux conçus pour traiter des séquences de données, où l'ordre des éléments est important. Ils sont utilisés pour modéliser des dépendances temporelles dans des données séquentielles, comme le langage naturel, les séries temporelles, ou les séquences générées par des capteurs.

Cependant, les RNN traditionnels présentent des limitations. L'un des principaux problèmes est le "problème du gradient qui disparaît" ou "du gradient qui explose", où les gradients deviennent trop petits ou trop grands pendant la rétropropagation, rendant l'apprentissage inefficace sur de longues séquences.

LSTM sont une amélioration des Réseaux de Neurones Récurrents (RNN), conçue pour surmonter les limitations liées à la modélisation de séquences temporelles complexes. La principale innovation des LSTM réside dans leur capacité à gérer des dépendances à long terme, ce qui les rend idéaux pour des applications telles que la prédiction de séries temporelles, la modélisation du langage naturel, et d'autres tâches impliquant des données séquentielles. Les LSTM sont devenus populaires en raison de leur efficacité dans la gestion de l'information sur de longues séquences, comblant ainsi les lacunes observées dans les RNN traditionnels.

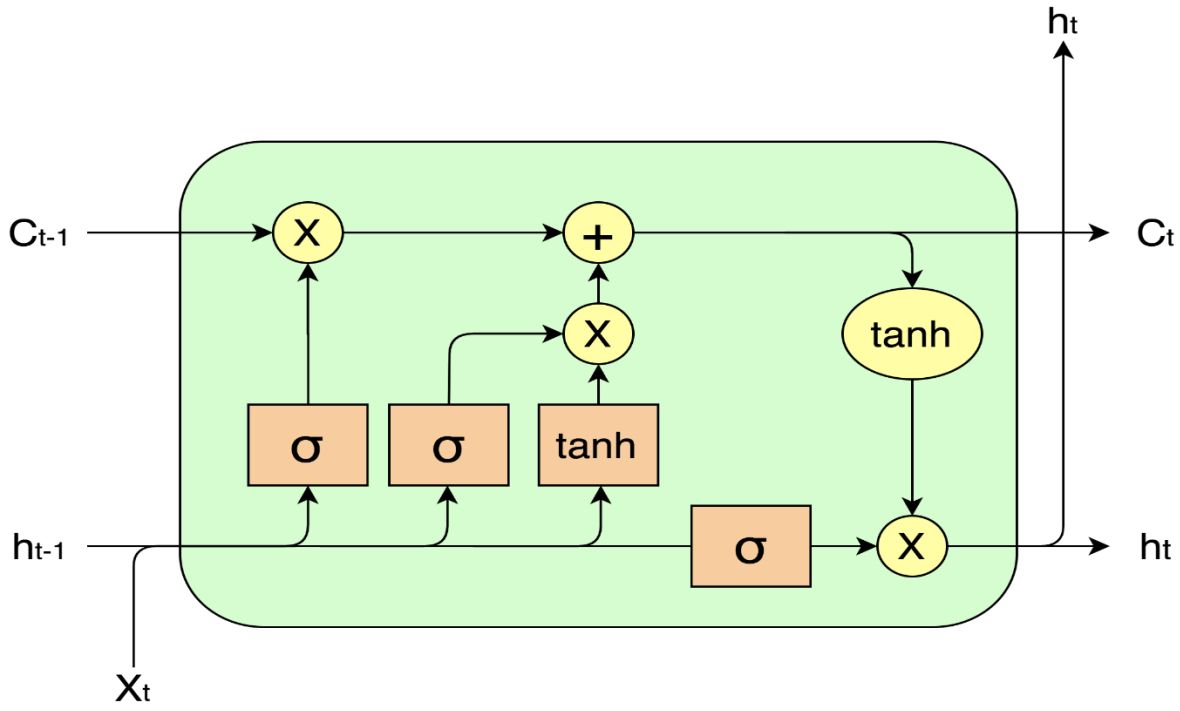
4. Fonctionnement d'un LSTM

Le fonctionnement d'un LSTM repose sur trois portes principales : la porte d'oubli, la porte d'entrée et la porte de sortie. Chaque porte a un rôle spécifique dans la gestion et la mise à jour de l'information stockée, permettant au LSTM de maintenir des dépendances à long terme et de traiter des données séquentielles de manière efficace.

- La porte d'oubli dans un LSTM (Long Short-Term Memory) est l'un des composants clés de son architecture. Elle a pour rôle de déterminer quelles informations stockées précédemment dans la cellule LSTM doivent être oubliées ou effacées. Cette fonctionnalité est cruciale pour permettre au LSTM de gérer des dépendances à long terme dans les données séquentielles.
- La porte d'entrée (Input Gate) est un élément clé de l'architecture d'un LSTM (Long Short-Term Memory). Elle a pour fonction de réguler les informations nouvelles qui doivent être intégrées dans la cellule LSTM à partir des données d'entrée et des informations de l'étape temporelle précédente.
- La porte de sortie (Output Gate) est une composante essentielle de l'architecture d'un LSTM (Long Short-Term Memory). Cette porte est responsable de réguler la sortie finale de la cellule LSTM, en décidant quelle partie de l'information stockée dans la cellule sera transmise comme sortie.

5. L'architecture d'un LSTM

L'architecture d'un LSTM comprend des cellules avec des mécanismes uniques. Chaque cellule contient des unités de mémoire capables de stocker des informations pendant des périodes prolongées. Ces cellules sont structurées de manière à faciliter l'apprentissage des dépendances complexes au sein des données séquentielles.



Ct-1 : C'est l'état de la cellule mémoire à l'instant précédent (t-1) dans une séquence temporelle. Cet état est utilisé comme base pour la mise à jour de la cellule mémoire à l'instant t.

ht-1 : C'est l'état caché (hidden state) à l'instant précédent (t-1). L'état caché est la sortie du LSTM à chaque pas de temps et est également utilisé comme l'état actuel pour les portes lors du pas de temps suivant

Xt : C'est l'entrée au pas de temps t. Cela peut être une partie de la séquence que le LSTM traite à cet instant.

Ct : Représente l'état de la cellule mémoire à un moment donné dans une séquence temporelle

ht : Représente l'état interne du réseau à un moment donné dans une séquence temporelle

tanh : La fonction tangente hyperbolique (tanh) est souvent utilisée dans les LSTM pour réguler les valeurs. Elle prend en entrée une valeur réelle et produit une valeur dans la plage [-1, 1].

+/X : Cela représente généralement les opérations de somme et de multiplication qui sont effectuées pendant la mise à jour de la cellule mémoire. Les opérations spécifiques dépendent des portes (porte d'oubli, porte d'entrée, porte de sortie) et de la mise à jour de la cellule mémoire.

6. Avantages des LSTM

Les LSTM offrent plusieurs avantages, tels que la capacité à traiter des séquences de données de longue durée, la conservation des dépendances temporelles, et la prévention des problèmes de disparition ou d'explosion de gradients, ce qui les rend efficaces pour des tâches comme la traduction automatique et la génération de textes

- Gestion des dépendances à long terme : Les LSTM sont conçus pour capturer et maintenir des dépendances temporelles à long terme dans les séquences. Grâce à leur architecture avec des portes (portes d'oubli, d'entrée et de sortie), ils peuvent réguler l'information stockée dans la cellule mémoire sur de longues périodes, évitant ainsi le problème de disparition du gradient que peuvent rencontrer les RNN classiques.

- Prévention du problème de disparition du gradient : Les LSTM incluent des mécanismes tels que les portes qui permettent de contrôler le flux d'information. Cela aide à résoudre le problème de disparition du gradient qui peut survenir lors de l'entraînement des RNN, en facilitant le transport des gradients sur de longues séquences.

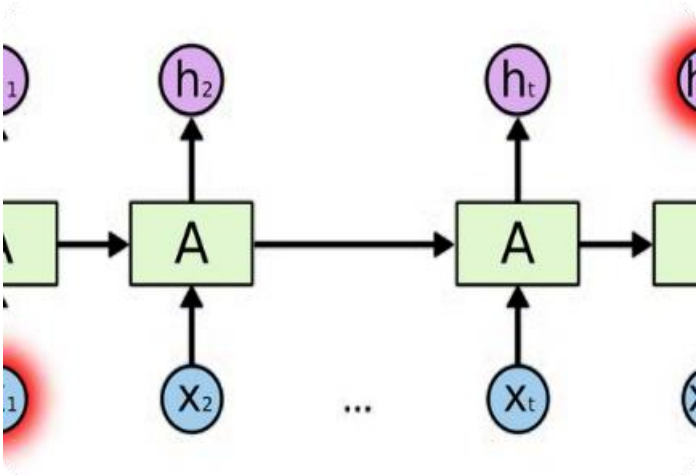
- Adaptabilité à différentes échelles de temps : Les LSTM peuvent traiter efficacement des informations sur différentes échelles de temps. Ils sont capables de maintenir des informations importantes à court terme tout en gérant simultanément des dépendances à plus long terme, ce qui les rend adaptés à une variété de tâches.

- Apprentissage de motifs complexes : Les LSTM sont capables d'apprendre des modèles complexes dans les séquences, ce qui les rend particulièrement efficaces pour des tâches telles que la modélisation du langage, la traduction automatique, la reconnaissance de la parole et la prédiction de séries temporelles.

7. Exemples d'utilisation d'un LSTM

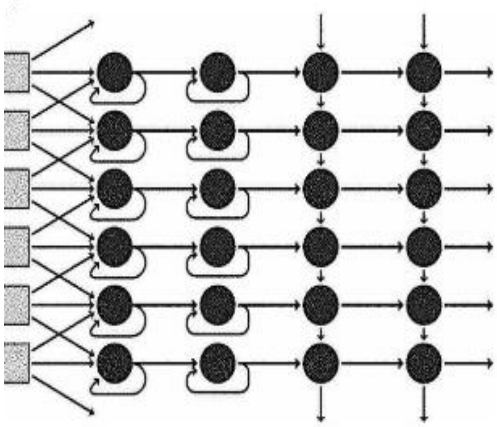
Les LSTM sont appliqués dans la traduction automatique pour produire des résultats de grande qualité. Ils sont également utilisés dans les systèmes de recommandation, le traitement du langage naturel, la génération de musique, et la prédiction de séries temporelles telles que les mouvements boursiers

➢ Traduction automatique



Exemple d'application réussie des LSTM dans la traduction automatique est le modèle Google Neural Machine Translation (GNMT), introduit par Google en 2016. GNMT a contribué à améliorer significativement la qualité des traductions automatiques fournies par Google Translate

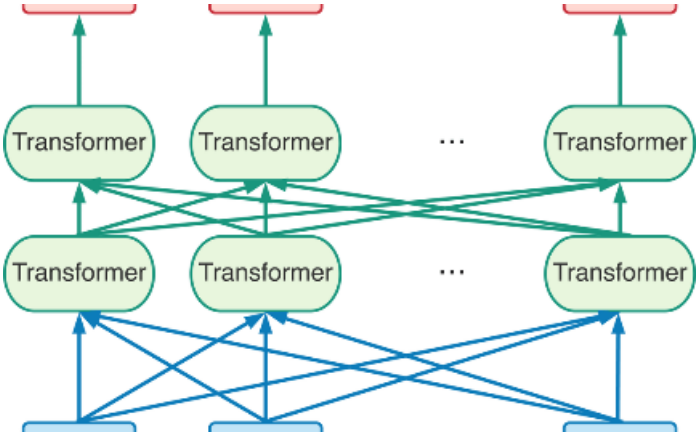
➢ Génération de musique



.27 Hexahedria architecture (folded). Reproduced from [94] with permission of the author

Les LSTM, utilisés pour créer de la musique, apprennent à partir de séquences de notes musicales et peuvent ensuite générer de nouvelles séquences harmonieuses. En encodant les motifs musicaux dans leur mémoire à long terme, ces réseaux peuvent produire des compositions originales en traitant les informations mélodiques au fil du temps. Cela offre une méthode créative pour générer de la musique avec une certaine structure et une touche d'originalité.

- Prédiction de séries temporelles



Les LSTM sont utilisés pour prédire les mouvements boursiers en analysant les tendances passées et en apprenant à partir des données historiques, ce qui aide à anticiper les fluctuations du marché financier.

8.Limitations d'un LSTM

Bien que puissants, les LSTM présentent certaines limitations telles que le besoin de puissance de calcul élevé pour l'entraînement, le risque de surapprentissage sur de petites quantités de données, et la difficulté à capturer des dépendances très longues dans les séquences.

Besoin de puissance de calcul élevé : L'entraînement des LSTM peut être intensif en termes de puissance de calcul, en particulier avec des ensembles de données volumineux ou des architectures de modèles complexes. Cela peut rendre l'utilisation des LSTM coûteuse en termes de ressources matérielles et financières.

Risque de surapprentissage : Les LSTM peuvent être sujets au surapprentissage, en particulier lorsque le modèle est complexe et que la quantité de données d'entraînement est limitée. Cela signifie que le modèle peut bien fonctionner sur les données d'entraînement, mais il peut avoir du mal à généraliser efficacement sur de nouvelles données.

Difficulté à capturer des dépendances très longues : Bien que les LSTM soient conçus pour gérer les dépendances à long terme dans les séquences, ils peuvent encore rencontrer des difficultés à capturer des dépendances très étendues. Certains modèles plus récents, tels que les Transformers, ont montré des performances améliorées dans la capture de dépendances à long terme dans certaines tâches.

Interprétabilité limitée : Les LSTM sont souvent considérés comme des "boîtes noires" en raison de la complexité de leur architecture. Comprendre comment et pourquoi un LSTM

prend une décision particulière peut être difficile, ce qui peut être problématique dans des domaines où l'interprétabilité est cruciale.

Sensibilité aux hyperparamètres : Les performances des LSTM peuvent dépendre fortement du réglage des hyperparamètres tels que la taille du modèle, le taux d'apprentissage, etc. Trouver les paramètres optimaux peut nécessiter des efforts d'expérimentation et d'ajustement.

9.Application d'un LSTM (Netflix Stock prédiction)

Les étapes générales pour créer un projet de prédiction des actions Netflix en utilisant un LSTM

Étape 1 : Importer des bibliothèques :

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense,Dropout
from sklearn.metrics import mean_squared_error, mean_absolute_error
from plotly.subplots import make_subplots
import plotly.graph_objects as go
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping
import matplotlib.dates as dates
import plotly.express as px
import datetime as dt
import os
import seaborn as sns
```

La première étape consiste à importer les bibliothèques nécessaires pour travailler avec les données et construire le modèle LSTM

Étape 2 : Charger vos données :

	Date	Open	High	Low	Close	Adj Close	Volume
0	2015-12-16	119.800003	123.000000	118.089996	122.639999	122.639999	13181000
1	2015-12-17	123.970001	126.349998	122.419998	122.510002	122.510002	17284900
2	2015-12-18	120.849998	122.190002	117.919998	118.019997	118.019997	17948100
3	2015-12-21	119.510002	119.589996	115.660004	116.629997	116.629997	11670000
4	2015-12-22	117.300003	117.430000	114.860001	116.239998	116.239998	9689000
...	...	...	...	...	...	...	...
1002	2019-12-10	296.119995	298.940002	292.019989	293.119995	293.119995	10476100
1003	2019-12-11	294.489990	299.429993	294.200012	298.929993	298.929993	5589800
1004	2019-12-12	295.670013	299.170013	295.059998	298.440002	298.440002	4766600
1005	2019-12-13	298.500000	301.799988	297.250000	298.500000	298.500000	3879700
1006	2019-12-16	300.850006	305.709991	298.630005	304.209991	304.209991	4658900

Date : La date à laquelle les données ont été enregistrées.

Open : Le prix d'ouverture des actions de Netflix le jour donné. Il s'agit du prix auquel la première transaction a été effectuée lors de la journée de négociation.

High : Le prix le plus élevé atteint par les actions de Netflix au cours de la journée.

Low : Le prix le plus bas atteint par les actions de Netflix au cours de la journée.

Close : Le prix de clôture des actions de Netflix à la fin de la journée de négociation. Il s'agit du dernier prix auquel une transaction a été effectuée.

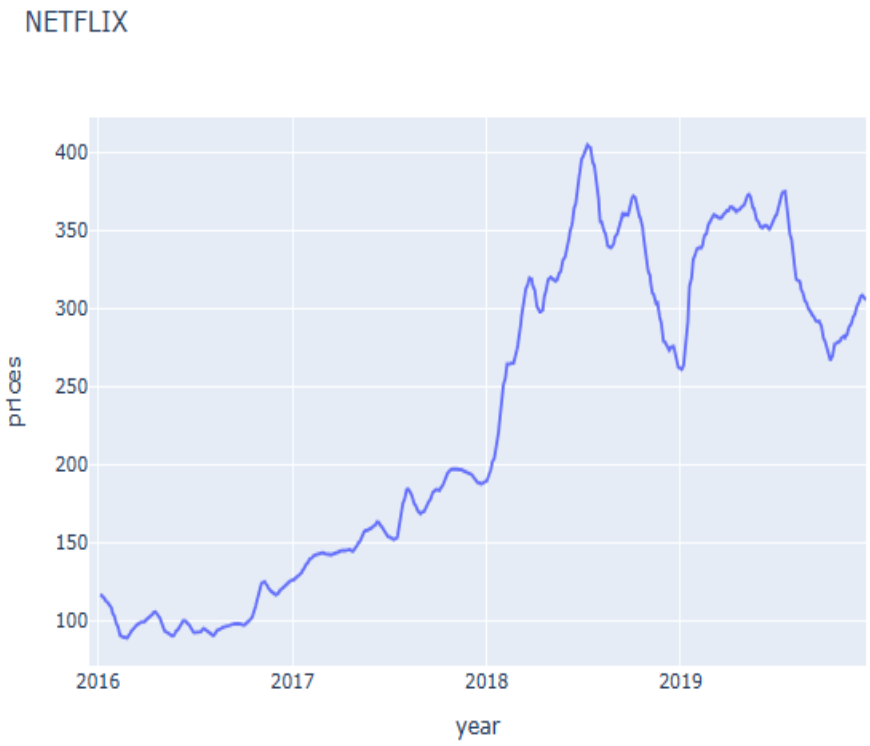
Adj Close : Le prix de clôture ajusté en fonction des événements tels que les dividendes, les fusions, etc. Il donne une vision plus précise des performances historiques des actions.

Volume : Le nombre total d'actions de Netflix échangées pendant la journée. Il représente la liquidité du marché et indique l'intérêt des investisseurs pour les actions.

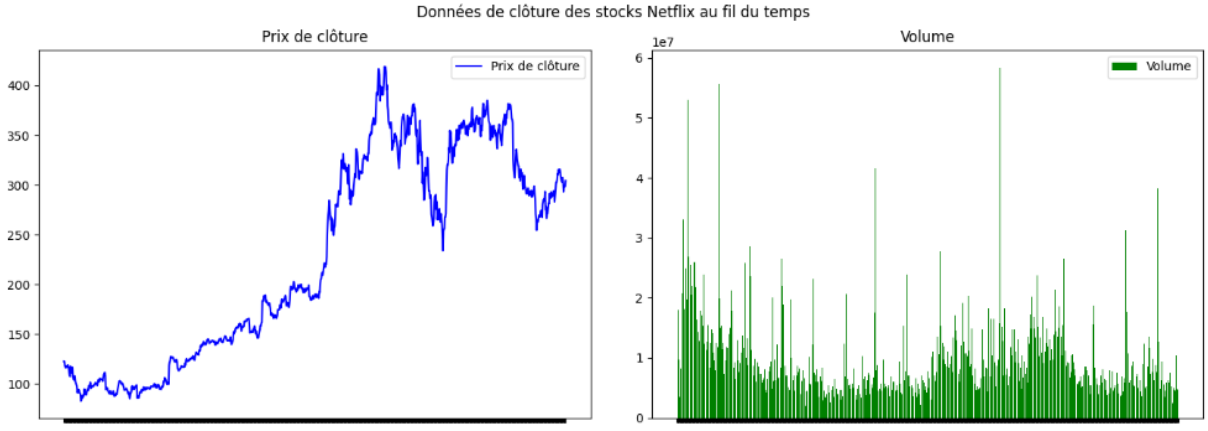
Étape 3 : Visualisation des données

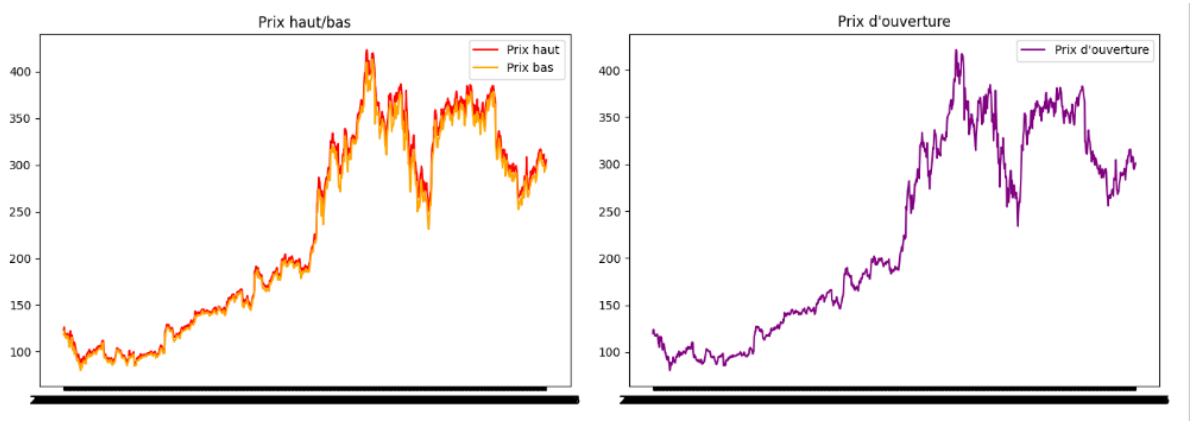
La visualisation des données est une étape importante pour comprendre les tendances et les motifs présents dans votre ensemble de données.

Le graphique interactif illustre l'évolution du prix d'ouverture des actions de Netflix au fil du temps, avec une moyenne mobile sur 14 jours pour mieux percevoir les tendances à moyen terme. Cette visualisation offre un aperçu lisse des variations de prix, facilitant l'identification de motifs significatifs dans le comportement des actions.

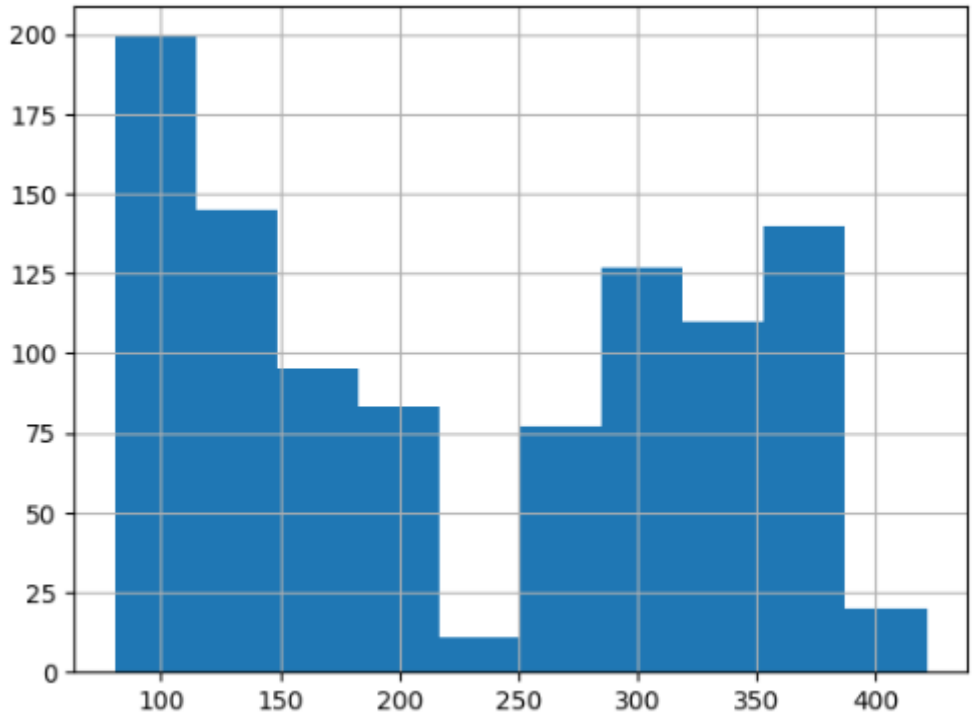


Cette visualisation en quatre parties présente différentes facettes des données de clôture des actions Netflix au fil du temps. Elle inclut le prix de clôture, le volume d'échanges, les prix hauts/bas, et les prix d'ouverture. L'utilisation de sous-graphiques offre une vue complète de la dynamique des actions de Netflix, permettant une analyse plus approfondie des tendances et des événements marquants.

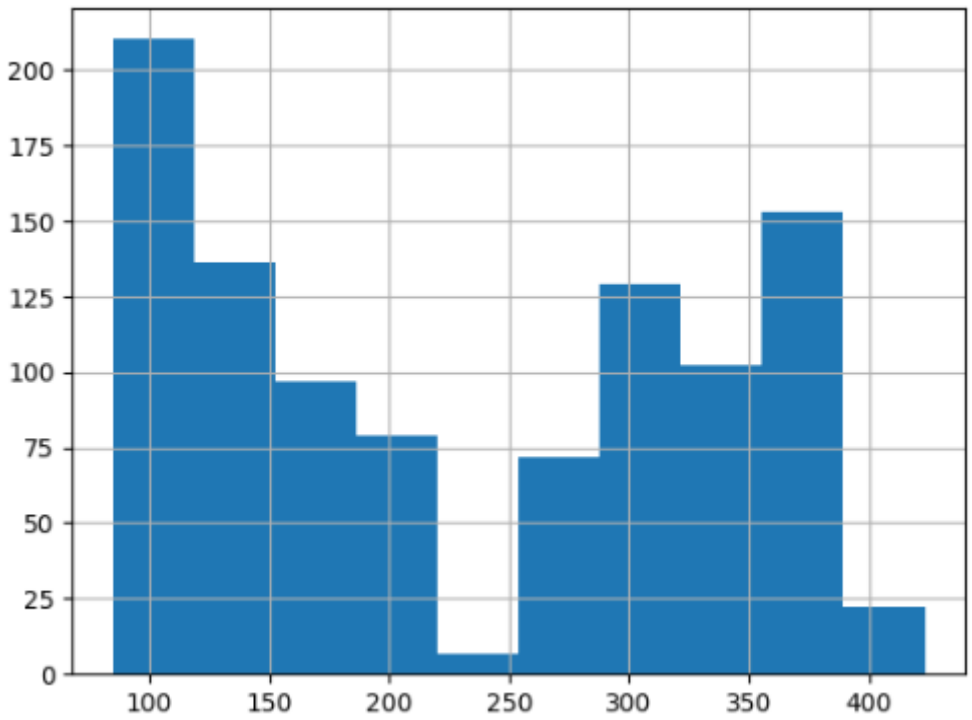




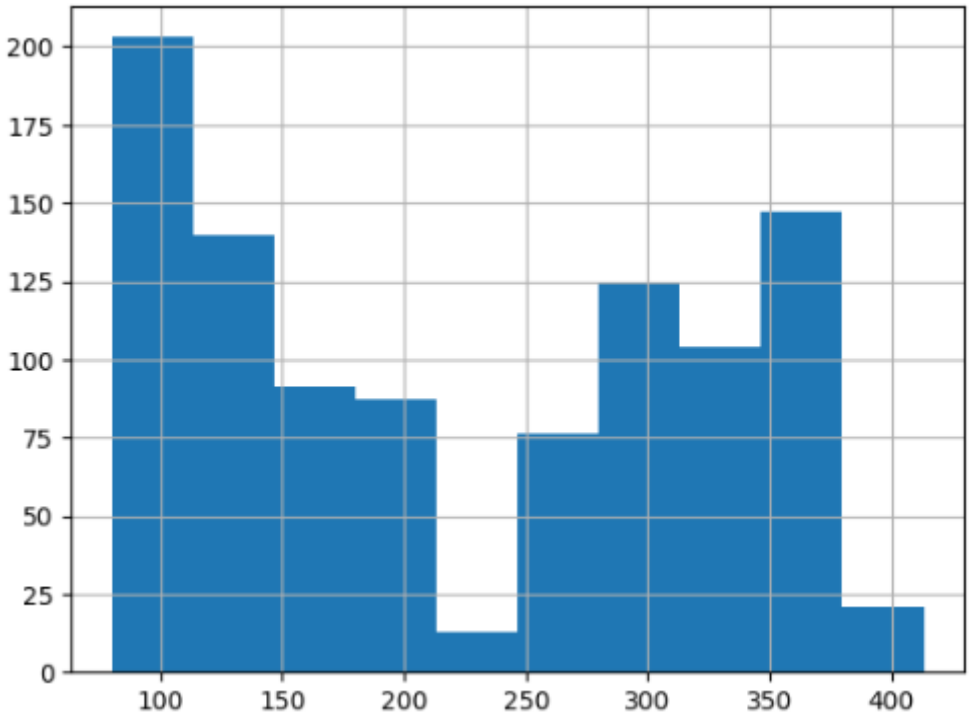
Histogramme représentant la distribution des prix d'ouverture des actions de Netflix :



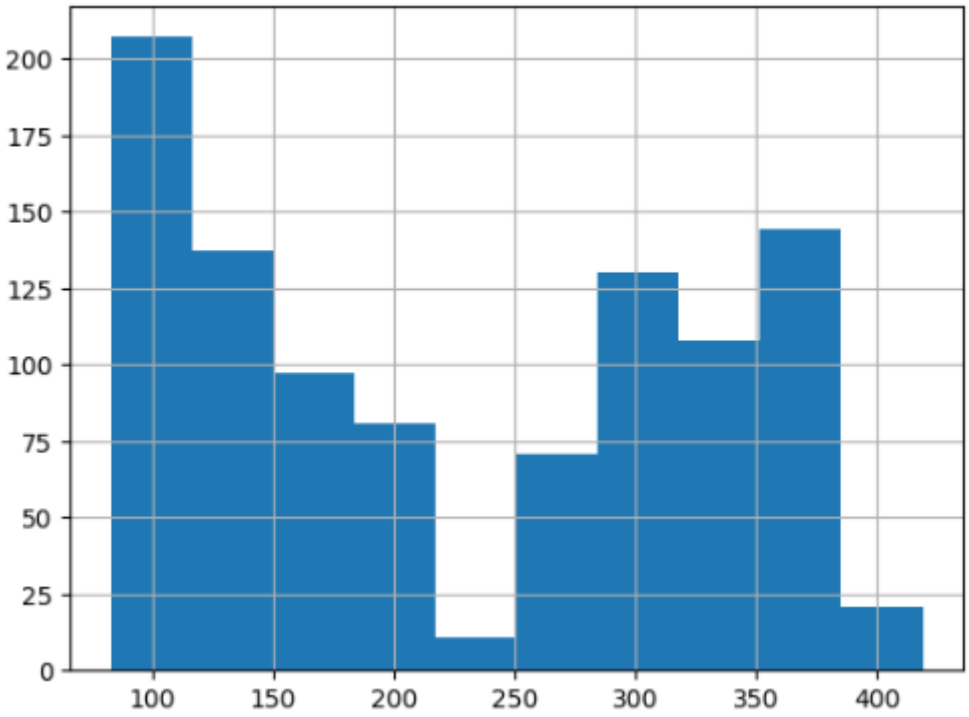
Histogramme représentant la distribution des prix les plus élevés (High) des actions de Netflix



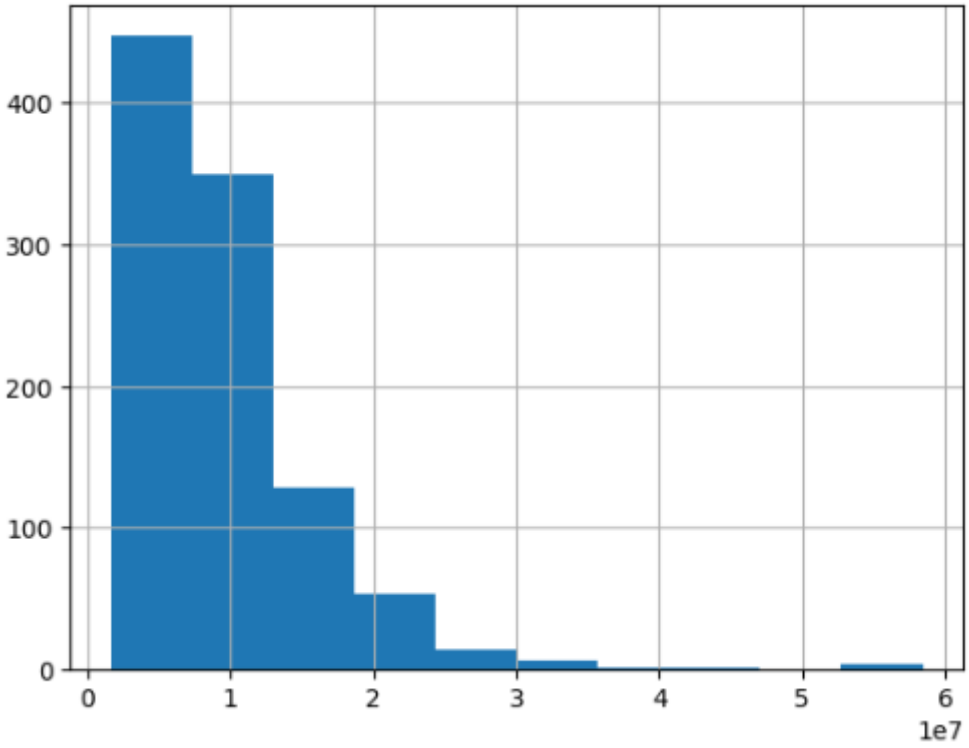
Histogramme représentant la distribution des prix les plus bas (Low) des actions de Netflix



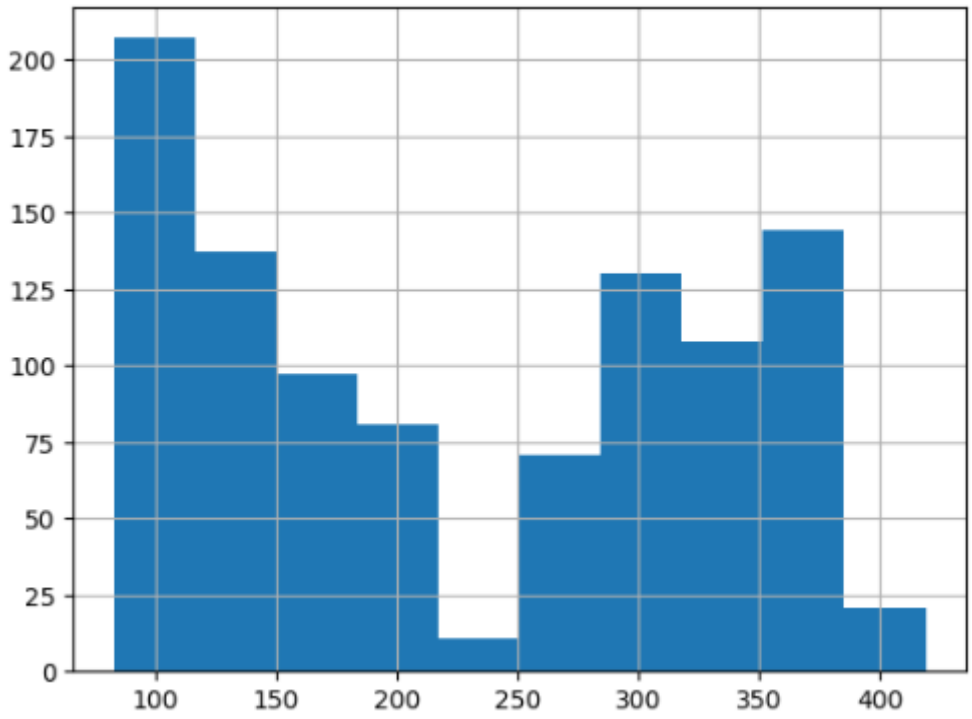
Histogramme représentant la distribution des prix de clôture (Close) des actions de Netflix



Histogramme représentant la distribution du volume d'échanges des actions de Netflix



Histogramme représentant la distribution des prix de clôture ajustés (Adj Close) des actions de Netflix



Étape 4 : préparation des données

La préparation des données est une étape cruciale avant de construire et d'entraîner un modèle LSTM.

- **Gestion des valeurs manquantes :** Vérifiez et gérez les éventuelles valeurs manquantes dans vos données. Vous pouvez choisir de les supprimer, les remplacer par des valeurs moyennes ou utiliser d'autres techniques de gestion des valeurs manquantes.

```
Valeurs manquantes avant le traitement :
Date         0
Open         0
High         0
Low          0
Close        0
Adj Close    0
Volume       0
dtype: int64
```

Normalisation des données : Normalisez les données pour mettre toutes les fonctionnalités à la même échelle

```
[[0.11510813]
 [0.12734368]
 [0.11818901]
 [0.11425721]
 [0.10777267]
 [0.10662834]
 [0.11047211]
 [0.1076553 ]
 [0.11038409]
 [0.11261406]
 [0.10457439]
 [0.08341891]
 [0.08767347]
 [0.07253308]
 [0.10501453]
 [0.1049265 ]
 [0.09260291]
 [0.10428098]
```

Création de séquences temporelles : Transformez vos données en séquences temporelles pour l'entraînement du modèle LSTM**.**

```
prediction_days = 30

x_train = []
y_train = []

for x in range(prediction_days, len(scaled_data)-10):      ######
    x_train.append(scaled_data[x-prediction_days:x, 0])
    y_train.append(scaled_data[x+10, 0])      ###### predict 10 days after

x_train, y_train = np.array(x_train), np.array(y_train)
x_train = np.reshape(x_train, (x_train.shape[0], x_train.shape[1], 1))
```

```
print(x_train.shape)
print(y_train.shape)
```

```
(860, 30, 1)
(860,)
```

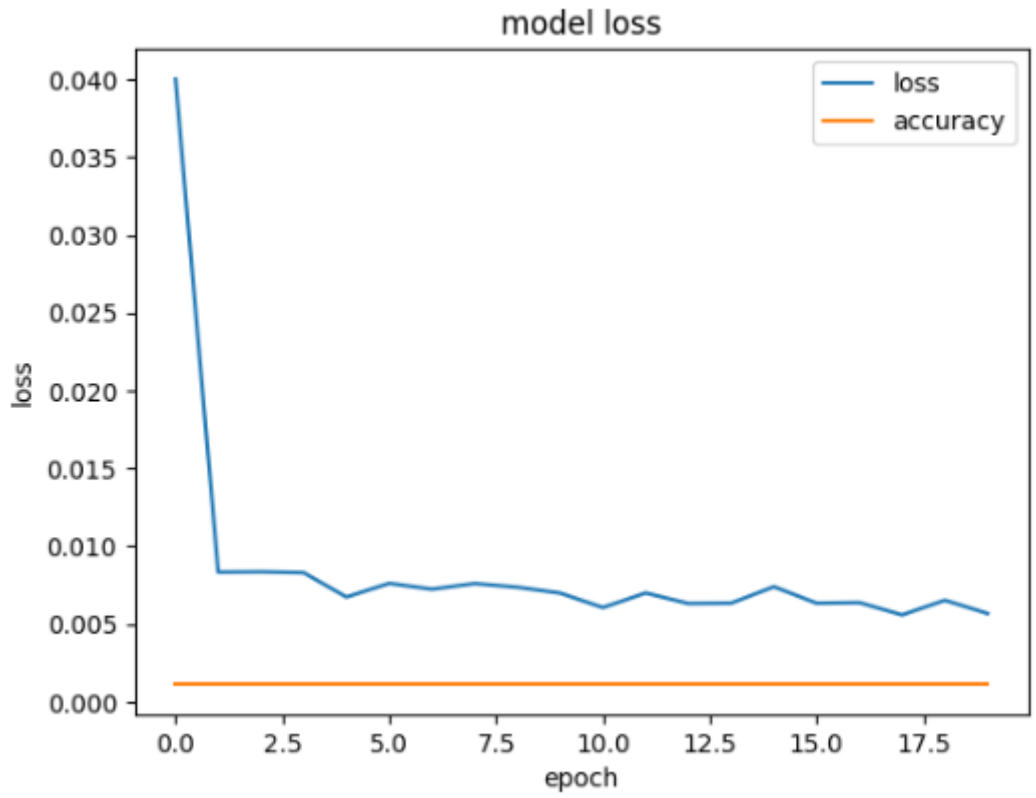
Étape 5 : Création du modèle LSTM

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_3 (LSTM)               (None, 30, 50)            10400

 dropout_3 (Dropout)         (None, 30, 50)            0

 lstm_4 (LSTM)               (None, 30, 50)            20200

 dropout_4 (Dropout)         (None, 30, 50)            0

 lstm_5 (LSTM)               (None, 50)                20200

 dropout_5 (Dropout)         (None, 50)                0

 dense_1 (Dense)             (None, 1)                 51

=================================================================
Total params: 50851 (198.64 KB)
Trainable params: 50851 (198.64 KB)
Non-trainable params: 0 (0.00 Byte)
```

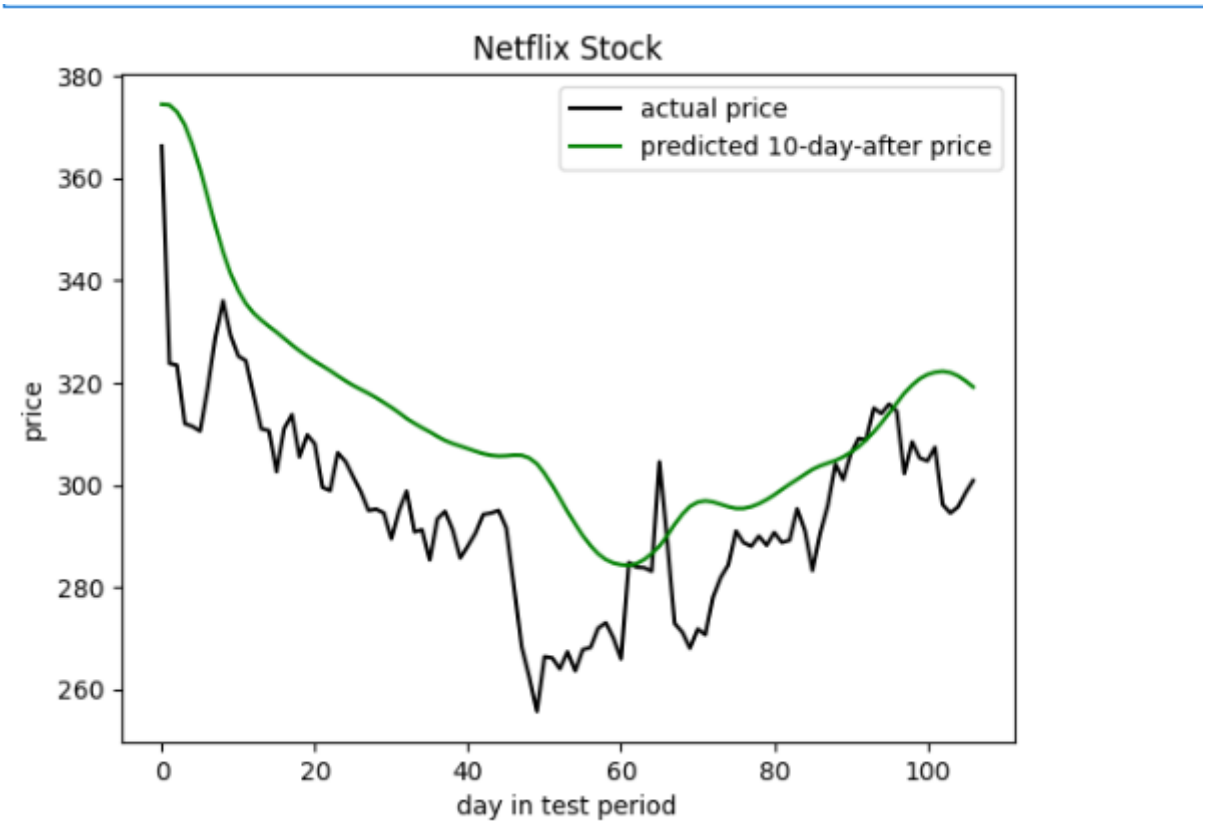
D'entraîner un modèle LSTM en utilisant les données d'entraînement (x_train et y_train) et de sauvegarder les poids du modèle à chaque époque si la performance sur les données de validation s'améliore

```
Epoch 1/20
26/27 [===========================>..] - ETA: 0s - loss: 0.0408 - accuracy: 0.0012WARNING:tensorflow:Can save best model only with val_loss available, sk
ipping.
27/27 [==============================] - 10s 47ms/step - loss: 0.0400 - accuracy: 0.0012
Epoch 2/20
27/27 [==============================] - ETA: 0s - loss: 0.0083 - accuracy: 0.0012  WARNING:tensorflow:Can save best model only with val_loss available,
skipping.
27/27 [==============================] - 1s 46ms/step - loss: 0.0083 - accuracy: 0.0012
Epoch 3/20
26/27 [===========================>..] - ETA: 0s - loss: 0.0084 - accuracy: 0.0012WARNING:tensorflow:Can save best model only with val_loss available, sk
ipping.
27/27 [==============================] - 1s 46ms/step - loss: 0.0084 - accuracy: 0.0012
Epoch 4/20
27/27 [==============================] - ETA: 0s - loss: 0.0083 - accuracy: 0.0012WARNING:tensorflow:Can save best model only with val_loss available, sk
ipping.
27/27 [==============================] - 1s 47ms/step - loss: 0.0083 - accuracy: 0.0012
Epoch 5/20
26/27 [===========================>..] - ETA: 0s - loss: 0.0068 - accuracy: 0.0012  WARNING:tensorflow:Can save best model only with val_loss available,
skipping.
27/27 [==============================] - 1s 50ms/step - loss: 0.0067 - accuracy: 0.0012
Epoch 6/20
27/27 [==============================] - ETA: 0s - loss: 0.0076 - accuracy: 0.0012  WARNING:tensorflow:Can save best model only with val_loss available,
skipping.
27/27 [==============================] - 1s 47ms/step - loss: 0.0076 - accuracy: 0.0012
Epoch 7/20
26/27 [===========================>..] - ETA: 0s - loss: 0.0073 - accuracy: 0.0012    WARNING:tensorflow:Can save best model only with val_loss availabl
e, skipping.
27/27 [==============================] - 1s 47ms/step - loss: 0.0072 - accuracy: 0.0012
Epoch 8/20
26/27 [===========================>..] - ETA: 0s - loss: 0.0074 - accuracy: 0.0012WARNING:tensorflow:Can save best model only with val_loss available, sk
ipping.
27/27 [==============================] - 1s 46ms/step - loss: 0.0076 - accuracy: 0.0012
Epoch 9/20
26/27 [===========================>..] - ETA: 0s - loss: 0.0075 - accuracy: 0.0012WARNING:tensorflow:Can save best model only with val_loss available, sk
ipping.
```

Tracer les courbes de perte (loss) et d'exactitude (accuracy) du modèle au fil des époques



Un graphique comparatif entre les prix réels et les prix prédits par le modèle pour une période de test



Préparer les données réelles pour être utilisées par le modèle LSTM afin d'effectuer une prédiction

```
# Sélection des données réelles pour la prédiction
real_data = [model_inputs[len(model_inputs)+1-prediction_days:len(model_inputs+1), 0]]

# Conversion de la liste en un tableau numpy
real_data = np.array(real_data)

# Remodelage du tableau pour qu'il ait la forme appropriée pour le modèle LSTM
real_data = np.reshape(real_data, (real_data.shape[0], real_data.shape[1], 1))

# Affichage de la forme du tableau
print(real_data.shape)
```

```
(1, 29, 1)
```

Utiliser le modèle LSTM entraîné pour effectuer une prédiction sur les données réelle

```
# Utilisation du modèle pour effectuer une prédiction sur les données réelles
prediction = model.predict(real_data)

# Inversion de la transformation effectuée précédemment avec le scaler
prediction = scaler.inverse_transform(prediction)

# Affichage de la valeur prédite
print(f"prediction: {prediction[0][0]}")
```

```
1/1 [==============================] - 2s 2s/step
prediction: 308.8885803222656
```

Conclusion

En Conclusion, les LSTM (Long Short-Term Memory) représentent des éléments cruciaux dans le domaine du traitement des données séquentielles, en fournissant un équilibre délicat entre la mémoire à court terme et la mémoire à long terme. Leur capacité à capturer des dépendances complexes dans les séquences en fait des outils incontournables pour résoudre une variété de problèmes de modélisation séquentielle. Les applications des LSTM sont étendues et continuent de s'élargir, soulignant leur pertinence constante dans des domaines tels que la prédiction de séries temporelles, la génération de texte, la traduction automatique, et bien d'autres. Leur flexibilité et leur efficacité en font des composants essentiels pour aborder des défis complexes liés aux données séquentielles.