HỘI NGHỊ SINH VIÊN NGHIÊN CỨU KHOA HỌC HUTECH 2020

TÌM HIỂU VỀ LONG SHORT-TERM MEMORY

TRƯƠNG TẤN PHÁT (16DTHC2 - 1611061139)

PHẠM NGUYỄN HOÀNG VĨNH PHÚC (16DTHC2 - 1611060524)

Khoa Công nghệ Thông tin, Trường Đại học Công Nghệ TP. Hồ Chí Minh (HUTECH)

Giảng viên hướng dẫn: thầy Bùi Mạnh Toàn

TÓM TẮT

Trong các giao dịch hàng ngày của thị trường chứng khoáng, các nhà đầu tư chiến lược và cả những người tham gia giao dịch luôn tìm cách để dự đoán ra những chuyển biến tiếp theo của giá trị hàng hoá mà họ đang tham gia theo dõi, giao dịch. Những cá nhân, tổ chức có khả năng đưa ra những dự đoán gần chính xác nhất đối với thị trường sẽ có khả năng dẫn đầu trong việc nắm bắt lợi thế thị trường, tăng cơ hội thành công trong giao dịch. Trong quá trình dự đoán, các thuật toán dự đoán bằng máy tính được ứng dụng bên cạnh việc phán đoán dựa trên kĩ thuật và kinh nghiệm của cá nhân. Máy học (marchine learning) là một trong những kĩ thuật tiềm năng luôn được nghiên cứu và ứng dụng trong ngành này. Trong bài viết này sử dụng kĩ thuật có tên “Long short-term memory” (bộ nhớ dài-ngắn hạn), để tìm hiểu khả năng dự đoán các bước chuyển biến của thị trường giao dịch tài chính.

*Từ khoá: finance, long short-term memory, lstm, marchine learning, python, time series forecasting.*

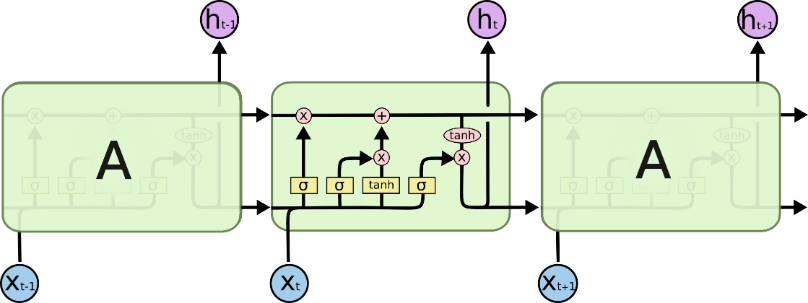
1. GIỚI THIỆU
   1. Long short-term memory[\*]
      1. Lý thuyết

Mạng bộ nhớ dài-ngắn (Long Short Term Memory networks), thường được gọi là LSTM - là một dạng đặc biệt của RNN, nó có khả năng học được các phụ thuộc xa. LSTM được giới thiệu bởi Hochreiter & Schmidhuber (1997), và sau đó đã được cải tiến và phổ biến bởi rất nhiều người trong ngành. Chúng hoạt động cực kì hiệu quả trên nhiều bài toán khác nhau nên dần đã trở nên phổ biến như hiện nay.

LSTM được thiết kế để tránh được vấn đề phụ thuộc xa (long-term dependency). Việc nhớ thông tin trong suốt thời gian dài là đặc tính mặc định của chúng, chứ ta không cần phải huấn luyện nó để có thể nhớ được. Tức là ngay nội tại của nó đã có thể ghi nhớ được mà không cần bất kì can thiệp nào.

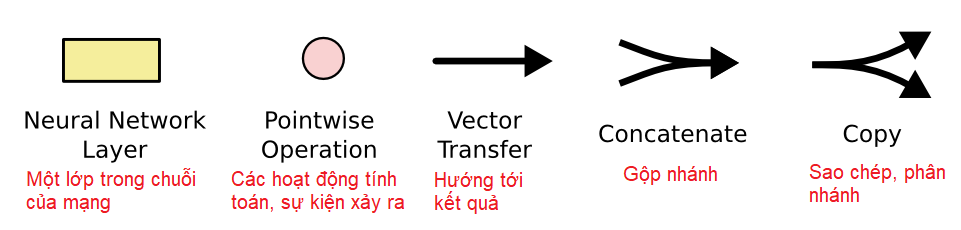
Mọi mạng hồi quy đều có dạng là một chuỗi các mô-đun lặp đi lặp lại của mạng nơ-ron. Với mạng RNN chuẩn, các mô-dun này có cấu trúc rất đơn giản, thường là một tầng *tanh*.

LSTM cũng có kiến trúc dạng chuỗi như vậy, nhưng các mô-đun trong nó có cấu trúc khác với mạng RNN chuẩn. Thay vì chỉ có một tầng mạng nơ-ron, chúng có tới 4 tầng tương tác với nhau một cách rất đặc biệt.



*Hình 1. Cấu tạo của một modul trong kiến trúc chuỗi LSTM*

Các kí hiệu dùng trong LSTM:

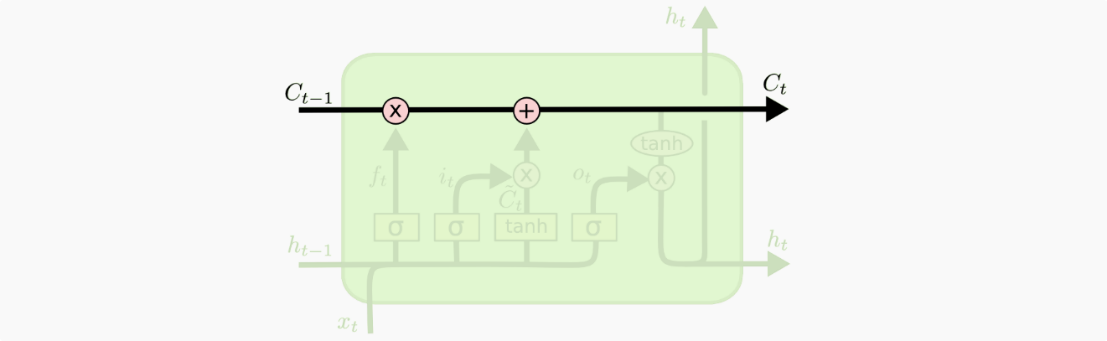


*Hình 2. Các kí hiệu dùng trong LSTM*

* + 1. Ý tưởng cốt lõi của LSTM

Chìa khóa của LSTM là trạng thái tế bào (cell state)

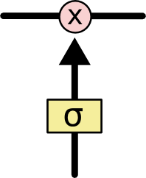
Trạng thái tế bào là một dạng giống như băng truyền. Nó chạy xuyên suốt tất cả các mắt xích (các nút mạng) và chỉ tương tác tuyến tính đôi chút. Vì vậy mà các thông tin có thể dễ dàng truyền đi thông suốt mà không sợ bị thay đổi.



*Hình 3. Trạng thái tế bào*

LSTM có khả năng bỏ đi hoặc thêm vào các thông tin cần thiết cho trạng thái tế báo, chúng được điều chỉnh cẩn thận bởi các nhóm được gọi là cổng (gate).

Các cổng là nơi sàng lọc thông tin đi qua nó, chúng được kết hợp bởi một tầng mạng sigmoid và một phép nhân.



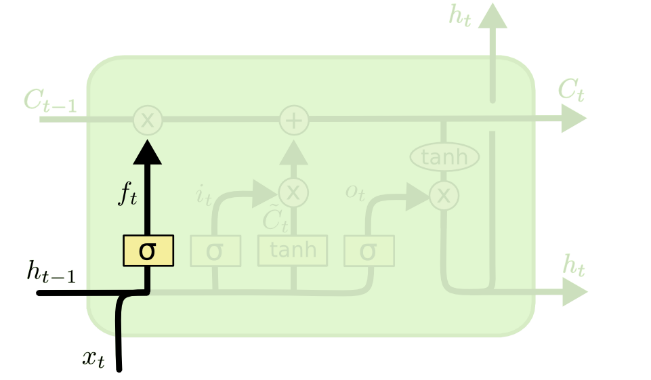
*Hình 4. Cổng sàng lọc*

Tầng sigmoid sẽ cho đầu ra là một số trong khoản [0, 1][0,1], mô tả có bao nhiêu thông tin có thể được thông qua. Khi đầu ra là 00 thì có nghĩa là không cho thông tin nào qua cả, còn khi là 11 thì có nghĩa là cho tất cả các thông tin đi qua nó.

Một LSTM gồm có 3 cổng như vậy để duy trì và điều hành trạng thái của tế bào.

* 1. Hoạt động
     1. Bước 1

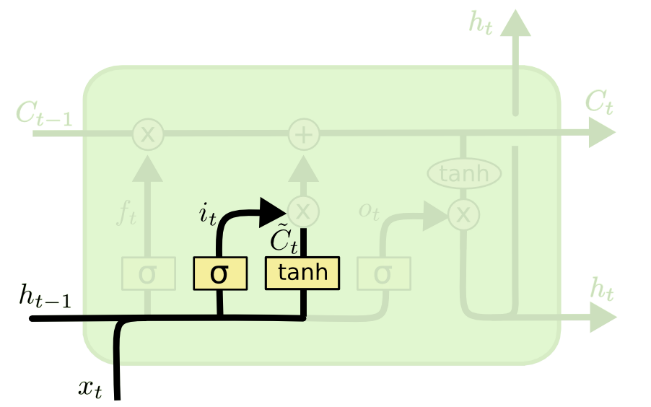
Bước đầu tiên của LSTM là quyết định xem thông tin nào cần bỏ đi từ trạng thái tế bào. Quyết định này được đưa ra bởi tầng sigmoid - gọi là “tầng cổng quên” (forget gate layer). Nó sẽ lấy đầu vào là *ht*−1 và xt ds rồi đưa ra kết quả là một số trong khoảng [0, 1][0,1] cho mỗi số trong trạng thái tế bào C\_{t-1}*Ct*−1​. Đẩu ra là 11 thể hiện rằng nó giữ toàn bộ thông tin lại, còn 00 chỉ rằng toàn bộ thông tin sẽ bị bỏ đi.



*Hình 5. Chọn lọc thông tin đầu vào cho lstm*

* + 1. Bước 2

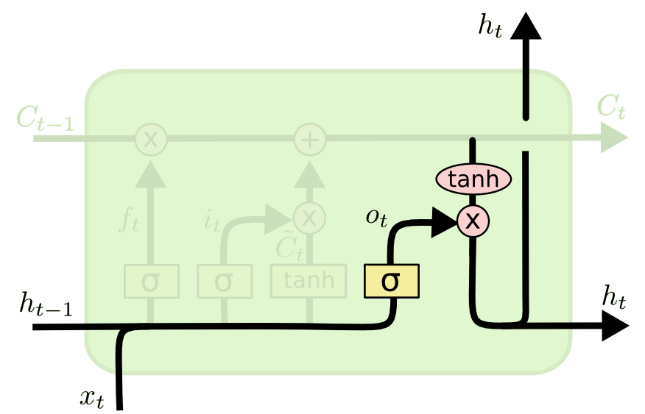
Bước tiếp theo là quyết định xem thông tin mới nào ta sẽ lưu vào trạng thái tế bào. Việc này gồm 2 phần. Đầu tiên là sử dụng một tầng sigmoid được gọi là “tầng cổng vào” (input gate layer) để quyết định giá trị nào ta sẽ cập nhập. Tiếp theo là một tầng *tanh* tạo ra một véc-tơ cho giá trị mới  nhằm thêm vào cho trạng thái. Trong bước tiếp theo, ta sẽ kết hợp 2 giá trị đó lại để tạo ra một cập nhập cho trạng thái.



*Hình 6. Chọn lọc thông tin để lưu vào tế bào*

* + 1. Bước 3

Giá trị đầu ra sẽ dựa vào trạng thái tế bào, nhưng sẽ được tiếp tục sàng lọc. Đầu tiên, ta chạy một tầng sigmoid để quyết định phần nào của trạng thái tế bào ta muốn xuất ra. Sau đó, ta đưa nó trạng thái tế bảo qua một hàm *tanh* để có giá trị nó về khoảng [-1, 1][−1,1], và nhân nó với đầu ra của cổng sigmoid để được giá trị đầu ra ta mong muốn.



*Hình 7. Cho ra kết quả sau quá trình phân loại*

1. THỬ NGHIỆM
   1. Dữ liệu thử nghiệm

Dữ liệu thử nghiệm được dùng trong bài được trích từ nguồn dữ liệu giao dịch của Yahoo Finance.

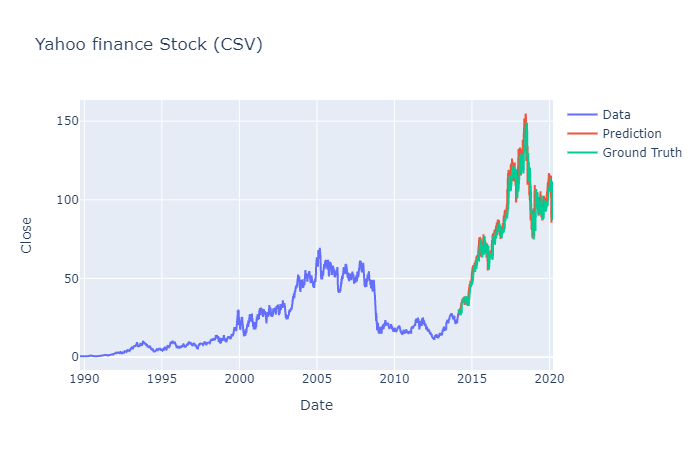
Luồng dữ liệu giá trị giao dịch của 2 công ty: Google và Apple với khung thời gian từ 01-01-1998 đến 12-03-2020.

* 1. Môi trường thử nghiệm
* Môi trường thực hiện: Google Colab
* Ngôn ngữ thực hiện: Python 3
* Các thư viện (dùng cho python):
  + Math
  + Pandas\_datareader
  + Numpy
  + Pandas
  + MinMaxScaler (sklearn.preprocessing)
  + Sequential (keras.models)
  + Dense, LSTM (keras.layers)
  + Mathplotlib.pyplot
  + Plotly.express
  + Plotly.graph\_objects
  + TimeseriesGenerator (keras.preprocessing.sequencec)
* Cài đặt của lstm:

Bảng 1. Giá trị cài đặt của thử nghiệm

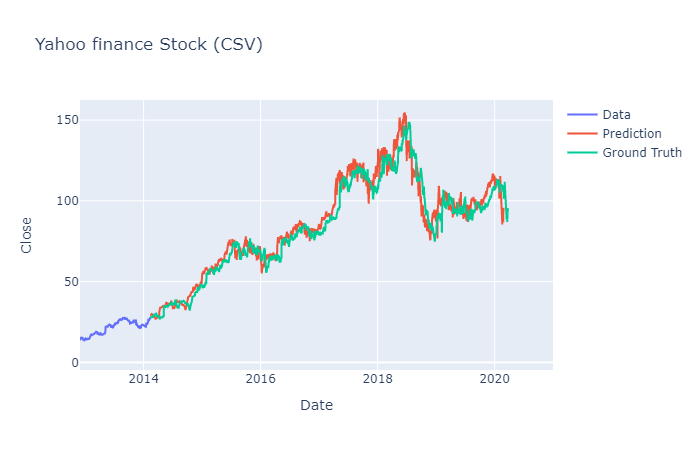
|  |  |  |
| --- | --- | --- |
| Thứ tự | Chỉ số | Giá trị |
| 1 | Batch\_size train | 10 |
| 2 | Batch size test | 10 |
| 3 | Epochs | 5 |
| 4 | Lookback | 5 |
| 5 | Optimizer | adam |
| 6 | Loss | mean\_squared\_error |
| 7 | Lượng dữ liệu training | 80% |
| 8 | Lượng dữ liệu testing | 20% |
| 9 | Số ngày dự đoán | 30 |
| 10 | Thời gian dự đoán | 24/03/2020 ~ 24/04/2020 |

* 1. Kết quả
     1. Kết quả training



*Hình 8. Kết quả sau quá trình training lstm*

* Dòng “Data”: thể hiện giá trị được dùng cho việc tự học của thuật toán (màu xanh dương)
* Dòng “Prediction”: thể hiện giá trị được dùng cho việc tự thử nghiệm quá trình dự đoán giá trị tương lai của thuật toán (màu đỏ cam)
* Dòng “Ground Truth”: thể hiện giá trị thật của dữ liệu đưa vào
* Dòng giá trị dự đoán “Prediction” cho thấy kết quả tự học của thuật toán có phần hi vọng, luôn bám sát dữ liệu thực tế. Làm tăng tỉ lệ dự đoán thành công của thuật toán



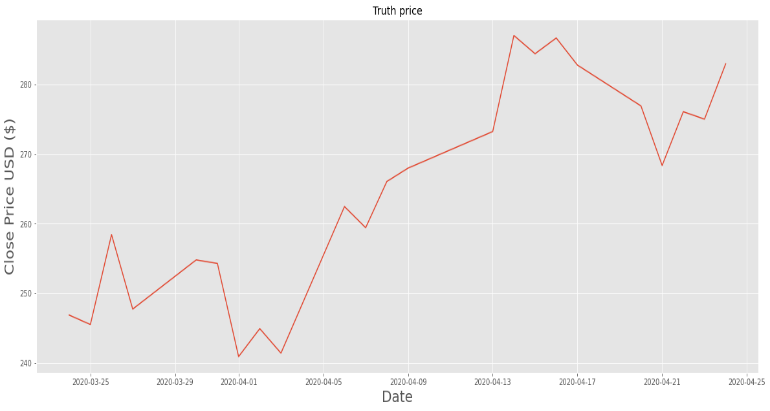
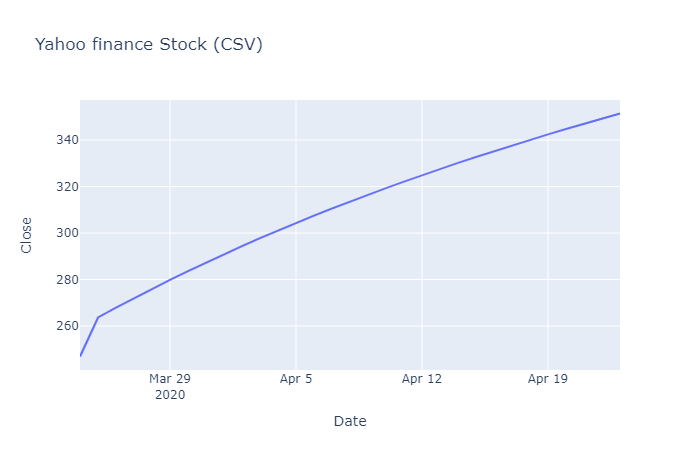
*Hình 9. Dữ liệu dự đoán luôn bám sát dữ liệu thật*

* + 1. Dự đoán 30 ngày tiếp theo
       1. Dữ liệu đầu vào
* Nguồn dữ liệu: Yahoo Finance
* Ngày dự đoán: 24/03/2020 đến 24/04/2020

Bảng 2. Giá trị đầu vào

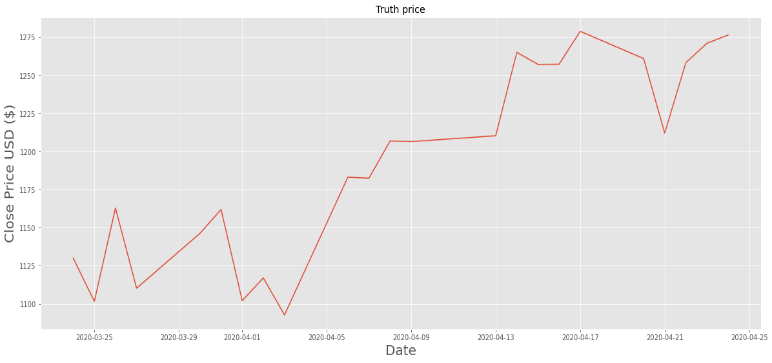
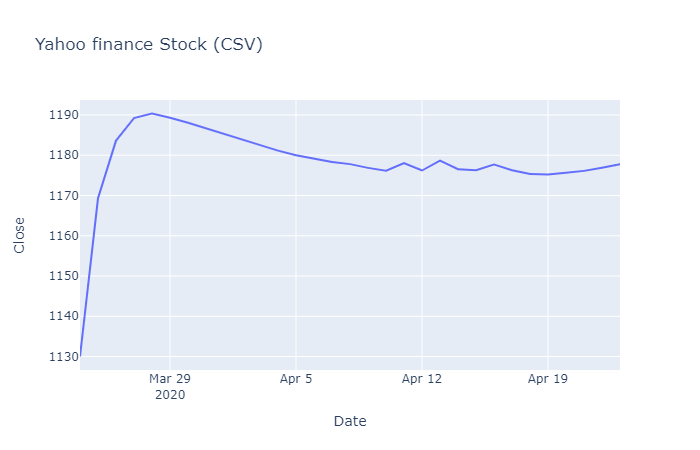
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Số thứ tự | Công ty | Mã thị trường | Ngày bắt đầu | Ngày kết thúc |
| 1 | Apple | AAPL | 12/12/1980 | 24-03-2020 |
| 2 | Alphabet | GOOGL | 19/08/2004 |
| 3 | Carriage Services | CSV | 09/08/1996 |
| 4 | IBM | IMB | 02/01/1962 |
| 5 | Electronic Art | EA | 20/09/1989 |
| 6 | Activision blizzard | ATVI | 25/10/1993 |

* + - 1. Kết quả dự đoán
         1. Apple

**

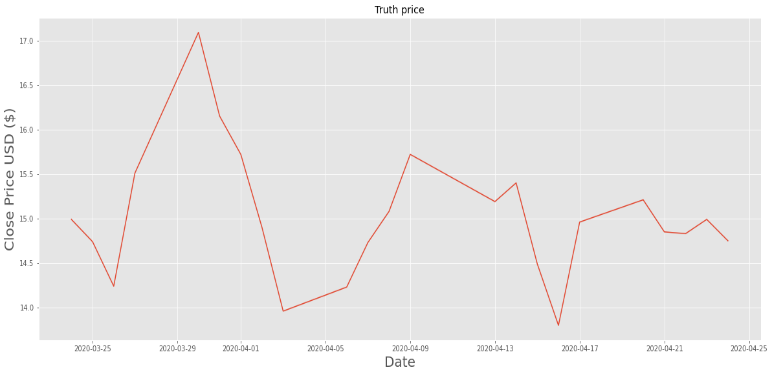
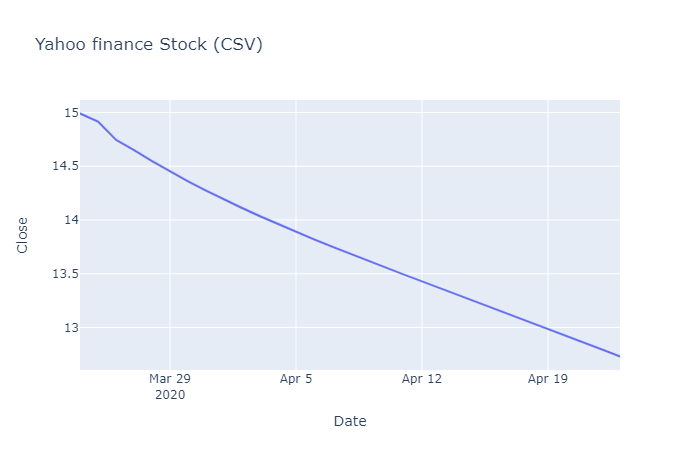
*Hình 9. Dự đoán của lstm cho 30 ngày tiếp theo Hình 10. Giá trị thực*

* + - * 1. Alphabet

**

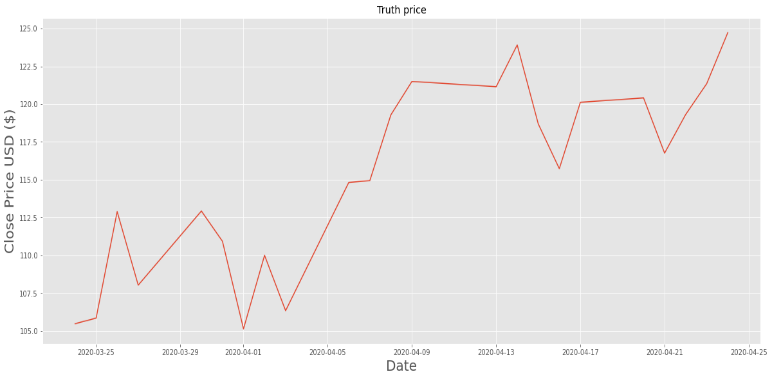
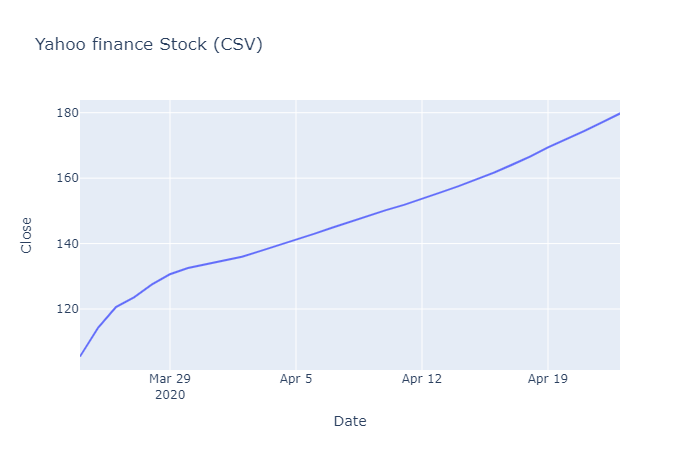
*Hình 9. Dự đoán của lstm cho 30 ngày tiếp theo Hình 10. Giá trị thực*

* + - * 1. Carriage Services

**

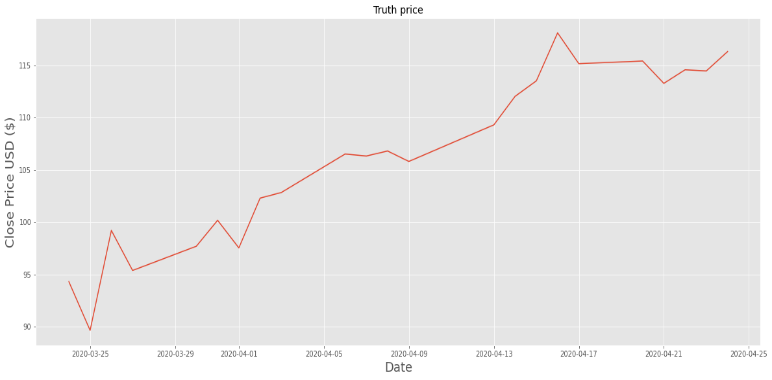
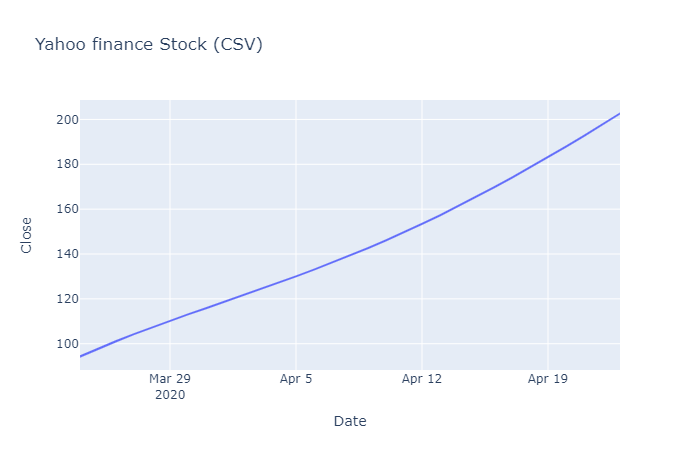
*Hình 9. Dự đoán của lstm cho 30 ngày tiếp theo Hình 10. Giá trị thực*

* + - * 1. IBM

**

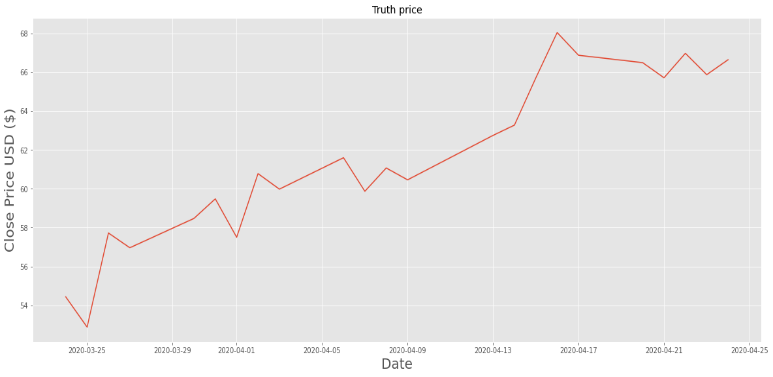
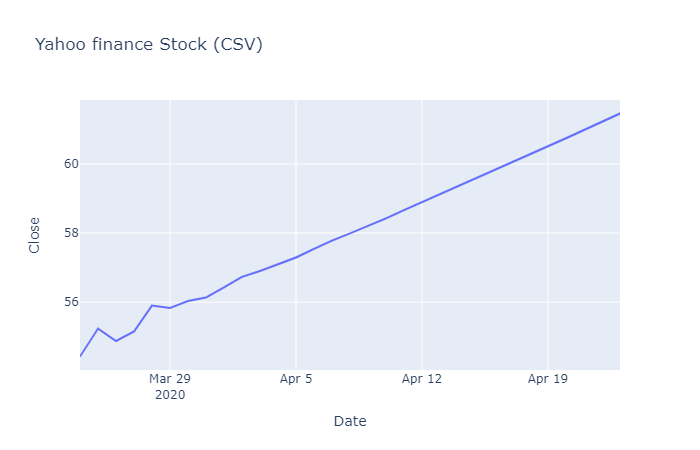
*Hình 9. Dự đoán của lstm cho 30 ngày tiếp theo Hình 10. Giá trị thực*

* + - * 1. Electronic Art

**

*Hình 9. Dự đoán của lstm cho 30 ngày tiếp theo Hình 10. Giá trị thực*

* + - * 1. Activistion Blizzard

**

*Hình 9. Dự đoán của lstm cho 30 ngày tiếp theo Hình 10. Giá trị thực*

* Các kết quả dự đoán có hướng phát triển trùng với hướng phát triển của dữ liệu thật, nhưng số liệu dự đoán đa phần thì không khớp.

1. KẾT LUẬN

* Mô hình long short-term memory cho khả năng dự đoán trong thời gian ngắn có tính chính xác ở mức khá và có thể tin tưởng
* Trong trường hợp dự đoán những dữ liệu về lâu dài thì thông tin dự đoán chỉ mang tính chất tương đối
* Mô hình được thực hiện nhiều lần thử và các kết quả có thể thay đổi giữa các lần
* Mô hình hứa hẹn sẽ có thể trở thành một tham chiếu để tham khảo trong quá trình dự đoán các giá trị tương lai của thị trường giao dịch
* Dựa vào mô hình dự đoán, ta có thể nắm được xu thế của thị trường sẽ đi lên hoặc đi xuống, nhưng không thể nắm được tình hình thay đổi của xu hướng thị trường
* Mô hình cần được thực hiện thêm nhiều lần nữa cùng với những cải tiến khác để có thể đưa ra dự đoán chính xác hơn

TÀI LIỆU THAM KHẢO

1. Keras: The Sequential model API
2. Keras: Getting started with the Keras Sequential model
3. George V Jose (20/12/2019) Predicting Sequential Data using LSTM: An Introduction
4. Hai's Blog (20/10/2017) [RNN] LSTM là gì?
5. Nttuan8 (2/6/2019) Long short term memory (LSTM)
6. phamduytung (02/10/2018) Phân biệt Epoch – Batch
7. plotly: Scatter Plots in Python
8. Pranathi (2018): Exporting Data from google colab to local machine
9. randerson112358 (24/12/2019) Stock Price Prediction Using Python & Machine Learning

TÀI LIỆU ĐƯỢC LƯU TẠI: <https://github.com/OrangeFoxie/Long-short-term-memory-predict-stock-market-Python-.git>