Triển khai hệ thống Big Data phân tích dữ liệu chứng khoán Việt Nam trong môi trường Docker sử dụng Hadoop và Spark

Vũ Văn Tới
University of engineering and technology
23020427@vnu.edu.vn

1 Giới thiệu

1.1 Bối cảnh

Trong kỷ nguyên dữ liệu hiện nay, khối lượng thông tin tài chính và chứng khoán tăng trưởng với tốc độ chóng mặt. Mỗi ngày, hàng triệu giao dịch được thực hiện trên thị trường, tạo ra một lượng dữ liệu khổng lồ về giá, khối lượng, biến động, và hành vi nhà đầu tư. Việc khai thác, xử lý và phân tích khối dữ liệu này đóng vai trò quan trọng trong việc dự đoán xu hướng thị trường, hỗ trợ ra quyết định đầu tư, và nâng cao hiệu quả giao dịch.

Tuy nhiên, dữ liệu chứng khoán thường có kích thước lớn, tốc độ sinh dữ liệu cao, và độ phức tạp đa dạng (bao gồm dữ liệu thời gian, dữ liệu phi cấu trúc từ tin tức hoặc mạng xã hội). Do đó, các công cụ truyền thống như Python thuần hoặc Pandas thường không đủ khả năng mở rộng và xử lý hiệu quả.

1.2 Mục tiêu

Mục tiêu chính dự án bao gồm :

- Xây dựng môi trường xử lý dữ liệu phân tán sử dụng Hadoop và Spark, triển khai trong Docker để đảm bảo tính linh hoạt và dễ dàng tái sử dụng.
- Thu thập và lưu trữ dữ liệu chứng khoán Việt Nam từ thư viện vnstock, bao gồm các chỉ số như giá mở cửa, giá đóng cửa, khối lượng giao dịch, mã cổ phiếu, v.v.
- Phân tích và trực quan hóa dữ liệu chứng khoán với các chỉ số thống kê, biến động giá, xu hướng thị trường, và mối tương quan giữa các cổ phiếu.
- Tối ưu hiệu năng xử lý dữ liệu lớn bằng cách khai thác sức mạnh tính toán song song của Spark trên cluster Hadoop.

2 Thu thập dữ liệu

2.1 Nguồn dữ liệu

- **Thư viện**: Sử dụng thư viện `vnstock` để truy cập dữ liệu thị trường chứng khoán Việt Nam.
- **API**: Dữ liệu được lấy thông qua phương thức `stock.quote.history` từ nguồn `VCI` (Vietcombank Securities).
- **Danh sách Công ty**: Lấy danh sách các công ty niêm yết từ lớp `Listing` của thư viện `vnstock`, bao gồm các mã cổ phiếu trên các sàn HOSE, HNX, và UPCOM.

- **Thời gian Dữ liệu**: Lịch sử giá cổ phiếu từ ngày 1/1/2010 (hoặc ngày niêm yết nếu muộn hơn) đến ngày hiện tại

2.2 Mục tiêu và Kết quả

- **Mục tiêu**:
  - Thu thập lịch sử giá cổ phiếu của tất cả các công ty niêm yết trên thị trường chứng khoán Việt Nam.
  - Lưu trữ dữ liệu dưới dạng tệp CSV để phục vụ phân tích sau này.
  - Đảm bảo quá trình thu thập dữ liệu diễn ra tự động, hiệu quả và xử lý tốt các lỗi như giới hạn API hoặc dữ liệu trống.
- **Kết quả**:
  - **Danh sách Công ty**: Tệp `companies_info.csv` chứa thông tin chi tiết của tất cả các công ty niêm yết
  - **Dữ liệu Giá**: Các tệp `[ticker].csv` chứa lịch sử giá của từng công ty, bao gồm các cột như ngày, giá mở cửa, giá đóng cửa, giá cao nhất, giá thấp nhất, và khối lượng giao dịch.

3 Kiến trúc hệ thống

3.1 Tổng quan

Hệ thống được thiết kế để xử lý và phân tích dữ liệu chứng khoán, sử dụng Docker Compose để triển khai các dịch vụ Hadoop, Spark và Jupyter Notebook. Các thành phần chính bao gồm HDFS để lưu

trữ phân tán, YARN để quản lý tài nguyên, Spark để xử lý dữ liệu nhanh chóng, và Jupyter Notebook để phân tích tương tác.

3.1.1 Lưu trữ Dữ liệu - HDFS

Hệ thống HDFS được triển khai với một NameNode và bốn DataNode để lưu trữ dữ liệu chứng khoán:

- **NameNode**: Sử dụng image bde2020/hadoop-namenode:2.0.0-hadoop3.2.1-java8, chịu trách nhiệm quản lý metadata của toàn bộ hệ thống tệp. NameNode được ánh xạ cổng 9870 cho giao diện web và 9000 cho giao tiếp với client. Metadata được lưu trữ tại thư mục ./hdfs/namenode, trong khi dữ liệu chứng khoán thô được lưu tại ./data.
- **DataNode**: Gồm bốn nút (datanode1 đến datanode4), sử dụng image bde2020/hadoop-datanode:2.0.0-hadoop3.2.1-java8, chịu trách nhiệm lưu trữ dữ liệu thực tế. Mỗi DataNode lưu dữ liệu tại các thư mục riêng ./hdfs/datanode[1-4] trên máy host. Các DataNode sử dụng tệp cấu hình ./hadoop.env để kết nối và giao tiếp ổn định với NameNode.

3.1.2 Quản lý Tài nguyên - YARN

YARN quản lý tài nguyên và lập lịch công việc:

- **ResourceManager**: Sử dụng image bde2020/hadoop-resourcemanager:2.0.0-hadoop3.2.1-java8, ánh xạ cổng 8088 cho giao diện web. Phụ thuộc vào NameNode và DataNode.
- **NodeManager**: Sử dụng image bde2020/hadoop-nodemanager:2.0.0-hadoop3.2.1-java8, thực thi các tác vụ, phụ thuộc vào ResourceManager.
- **HistoryServer**: Sử dụng image bde2020/hadoop-historyserver:2.0.0-hadoop3.2.1-java8, lưu trữ lịch sử công việc, ánh xạ cổng 8188, phụ thuộc vào NameNode, DataNode và ResourceManager.

3.1.3 Xử lý Dữ liệu - Spark

Cụm Spark được triển khai để xử lý dữ liệu quy mô lớn:

- **Spark Master**: Sử dụng image bde2020/spark-master:3.2.1-hadoop3.2, ánh xạ cổng 8080 cho giao diện web và 7077 cho giao tiếp worker.
- **Spark Workers**: Bốn nút (spark-worker-1 đến spark-worker-4) sử dụng image bde2020/spark-worker:3.2.1-hadoop3.2, kết nối với Spark Master qua spark://spark-master:7077.

3.1.4 Môi trường Phân tích - Jupyter Notebook

- Sử dụng image jupyter/pyspark-notebook:spark-3.2.1, cung cấp môi trường tương tác với PySpark. Cổng 8888 được ánh xạ để truy cập giao diện web với token toivu. Thư mục ./notebook và ./data được gắn để lưu trữ notebook và dữ liệu chứng khoán. Phụ thuộc vào spark-master để kết nối với cụm Spark.

4 Phân tích và xử lý dữ liệu

Phần này trình bày chi tiết quá trình phân tích và xử lý dữ liệu giá cổ phiếu lịch sử, sử dụng PySpark để tận dụng khả năng xử lý phân tán trên cụm Spark (`spark://spark-master:7077`). Dữ liệu đầu vào được lưu trữ trên HDFS dưới dạng các tệp CSV, mỗi tệp tương ứng với một mã cổ phiếu. Cuối cùng, một mô hình dự đoán giá sử dụng LSTM được xây dựng và huấn luyện cho mã cổ phiếu FPT.

4.1 Đọc và Chuẩn bị Dữ liệu Ban đầu

- **Đọc Dữ liệu:** Dữ liệu từ tất cả các file CSV trong thư mục `hdfs://namenode:9000/data/` được nạp vào một Spark DataFrame duy nhất. Các tùy chọn `header=True` và `inferSchema=True` được sử dụng để tự động nhận diện tiêu đề cột và kiểu dữ liệu.
- **Xử lý Cột Thời gian:** Cột `time` được chuyển đổi từ kiểu chuỗi sang kiểu ngày tháng (`DateType`) bằng hàm `F.to_date()` với định dạng `yyyy-MM-dd`.
- **Trích xuất Mã Cổ phiếu (Ticker):** Một cột mới `ticker` được tạo ra bằng cách trích xuất tên mã cổ phiếu từ tên của file CSV gốc sử dụng hàm `input_file_name()` và `regexp_extract()`.
- **Dữ liệu sau khi xử lý sẽ có dạng sau :**

time	open	high	low	close	volume	ticker
2010-01-04	12.95	12.95	12.95	12.95	247810	CII
2010-01-05	13.59	13.59	13.59	13.59	288720	CII
2010-01-06	14.22	14.22	13.9	14.22	1306580	CII
2010-01-07	14.43	14.85	14.11	14.43	2804340	CII
2010-01-08	14.53	14.64	13.9	13.9	1504570	CII
2010-01-11	14.22	14.22	13.59	13.59	629400	CII
2010-01-12	13.59	13.69	12.95	12.95	762290	CII
2010-01-13	12.95	13.17	12.43	13.06	1180890	CII
2010-01-14	13.71	13.71	13.71	13.71	41960	CII
2010-01-15	14.39	14.39	14.29	14.39	2314920	CII
2010-01-18	14.13	14.13	13.68	13.68	977220	CII
2010-01-19	13.68	13.91	13.49	13.49	720030	CII
2010-01-20	13.75	13.75	13.39	13.42	403860	CII
2010-01-21	13.01	13.39	12.78	12.78	794460	CII
2010-01-22	12.24	12.81	12.17	12.36	1003320	CII
2010-01-25	12.52	12.65	12.2	12.46	305720	CII
2010-01-26	12.78	13.07	12.78	13.07	886390	CII
2010-01-27	13.17	13.33	12.52	12.69	592510	CII
2010-01-28	12.69	12.75	12.46	12.52	285240	CII
2010-01-29	12.52	12.78	12.27	12.52	246990	CII

only showing top 20 rows

- **Dưới đây là giải thích cho từng cột dữ liệu hiển thị trong bảng**:

`time` : biểu thị ngày giao dịch của dữ liệu cổ phiếu được ghi lại. Định dạng Năm-Tháng-Ngày (YYYY-MM-DD).

`open` : Giá mở cửa: Đây là mức giá mà cổ phiếu được giao dịch lần đầu tiên khi thị trường mở cửa vào ngày đó.

`high` : Giá cao nhất: Đây là mức giá cao nhất mà cổ phiếu đạt được trong suốt phiên giao dịch của ngày đó.

`low` : Giá thấp nhất: Đây là mức giá thấp nhất mà cổ phiếu giảm xuống trong suốt phiên giao dịch của ngày đó.

`close` : Giá đóng cửa: Đây là mức giá cuối cùng mà cổ phiếu được giao dịch khi thị trường đóng cửa vào ngày đó.

`volume` : Khối lượng giao dịch: Cột này cho biết tổng số lượng cổ phiếu đã được giao dịch (mua và bán) trong ngày đó.

`ticker` : Mã cổ phiếu: Đây là ký hiệu hoặc mã định danh duy nhất cho cổ phiếu của một công ty cụ thể trên sàn giao dịch (ví dụ: CII trong hình ảnh).

4.2 Thống kê Mô tả và Phân tích Tổng quan Thị trường

Các bước phân tích sau được thực hiện trên toàn bộ dữ liệu để hiểu rõ hơn về đặc điểm chung của thị trường:

- **Thống kê Mô tả Chung:** Sử dụng hàm `summary()` trên các cột giá (`open`, `high`, `low`, `close`) và khối lượng (`volume`) để tính các chỉ số cơ bản như count, mean, stddev, min, max, và các phân vị.

summary	open	high	low	close	volume
count	4400101	4400061	4400061	4400060	4400101
mean	12.95034035355092	13.083267059252044	12.79381014945016	12.949878106207564	328725.8568994212
stddev	18.789752069983088	18.990656300827915	18.590629554462556	18.793004270793418	1964588.9818515815
min	0.0	0.0	0.0	0.0	0
25%	4.4	4.42	4.3	4.39	0
50%	8.2	8.3	8.1	8.2	1500
75%	14.86	15.01	14.68	14.87	45170
max	1562.5	1562.5	1358.7	1358.7	249760712

- **So sánh Các Mã Cổ phiếu:**
  - Dữ liệu được nhóm theo `ticker` để tính toán các thống kê riêng lẻ cho từng mã, bao gồm: `count`, `avg_close`, `std_close`, `min_close`, `max_close`, `avg_volume`.
  - Các mã được xếp hạng theo `avg_close` giảm dần.

ticker	count	avg_close	std_close	min_close	max_close	avg_volume
F88	51	1073.6882352941175	44.82081055518216	888.8	1180.0	4317.64705882353
VNZ	684	569.5385964912274	221.59933063923012	240.0	1358.7	2038.43567251462
IDP	1046	170.75557361376693	65.91339313268556	42.31	302.72	1056.5181644359466
XDC	68	168.4352941176471	223.56510939482075	13.5	999.9	400.0
GAB	1254	162.26692185007704	58.19751617690133	7.68	196.9	42740.47129186603
CMF	2151	156.12218503021822	70.86473565009824	41.35	390.4	150.13621571362157
HLB	2125	146.42398588235326	100.34450618699215	0.0	430.0	348.7854117647059
VCF	3674	117.52073761567779	57.81979664480004	18.13	334.98	2194.047359825803
NTC	2204	116.43990471869311	64.84695182993673	0.0	240.81	40947.05444646098
VJC	2163	114.92817383263952	21.1669132297023	60.7	184.84	706195.1215903837

- **Phân tích Rủi ro (Độ biến động):**
  - Tính lợi suất hàng ngày (`daily_return`) bằng cách sử dụng hàm cửa sổ (`Window`) để so sánh giá đóng cửa hiện tại và ngày trước đó.
  - Tính trung bình lợi suất (`avg_return`) và độ lệch chuẩn lợi suất (`volatility`) cho mỗi `ticker`. Các mã được xếp hạng theo `volatility` giảm dần.

ticker	avg_return	volatility
DNN	0.10187643867728115	1.7806845259321122
PTG	0.022516605067223746	0.6765261938940811
DKW	0.3	0.4242640687119285
PTX	0.020913662679470427	0.33901377412467304
XDC	0.03479982616170331	0.16189148574393228
NS3	-0.03406195964607938	0.14718524296935043
HHR	0.0037616430465172443	0.14204573893335298
SGS	0.0030034968148438855	0.1303604999019848
DKG	0.004405092030349224	0.12843838980170094
NSS	0.004549897387364335	0.11585282923734551

- **Phân tích Tăng trưởng Dài hạn:**
  - Tính toán tỷ lệ tăng trưởng (`growth_rate`) dựa trên giá đóng cửa đầu tiên và cuối cùng của mỗi mã.
  - Các mã được xếp hạng theo `growth_rate` giảm dần.

ticker	first_close	last_close	growth_rate
CAP	0.35	41.4	117.28571428571429
DHT	1.13	84.5	73.77876106194691
TV2	0.54	33.7	61.40740740740741
VE4	5.06	259.4	50.26482213438735
DNC	1.08	51.0	46.22222222222222
PTB	1.24	49.65	39.04032258064516
PDN	2.4	93.5	37.958333333333336
THG	1.43	51.7	35.15384615384616
L14	0.96	32.1	32.4375
LHC	3.07	99.9	31.540716612377857

- **Phân tích Thanh khoản:**
  - Tính giá trị giao dịch hàng ngày (`traded_value`).
  - Tính `avg_volume` và `avg_traded_value` cho mỗi `ticker`. Các mã được xếp hạng theo `avg_traded_value` giảm dần để xác định mã thanh khoản nhất.

ticker	avg_volume	avg_traded_value
TCX	1.4668E7	7.179986E8
VPB	1.2148206955479452E7	2.2870221234667817E8
HPG	1.0398341804160325E7	2.0345462962350082E8
STB	7991610.270167428	1.9179722911278564E8
TCB	8567674.38332431	1.8860507084791 56E8
ROS	1.0106185350404313E7	1.79836987359366 54E8
NVL	8444376.389368469	1.7627726154750 12E8
SSI	7810239.729325215	1.5987470988124543E8
GEX	6777247.702204409	1.5341377621849 27E8
VHM	2408054.075512076	1.4059804584106717E8

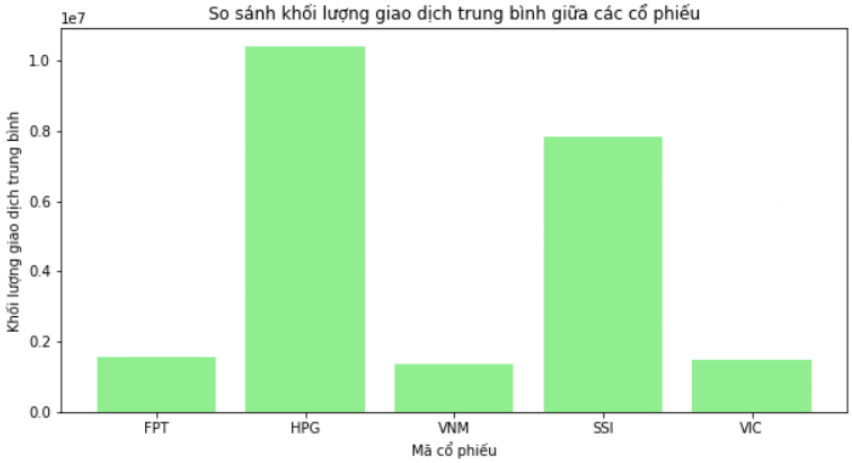
- **Phân tích Biến động theo Tháng:**
  - Trích xuất thông tin tháng từ cột `time`.
  - Tính `avg_month_return` và `month_volatility` cho từng `ticker` theo từng tháng.

ticker	month	avg_month_return	month_volatility
A32	1	0.005225978127063213	0.046559763135155204
A32	2	1.4260255430765634E-4	0.06329855163773913
A32	3	-0.001903069602235603	0.041724967354696794
A32	4	0.0015482929124923446	0.04702956155542099
A32	5	0.0012763706628027812	0.03689349113546448
A32	6	3.775462781556017E-4	0.03768769416938311
A32	7	0.002459934163447651	0.04696042422424292
A32	8	0.002148001776672375	0.03946234486016893
A32	9	0.00399768155289915	0.040616274005141666
A32	10	-0.0028582102800716195	0.04363906019026453
A32	11	0.002981947147462188	0.04221551139035171
A32	12	0.0035379274108610684	0.03973713650203381
AAA	1	0.0010904287479937202	0.02653227652018985
AAA	2	0.0030968544760234832	0.02803021002304573
AAA	3	4.942831285020446E-4	0.02578230075631313
AAA	4	2.2099263230239657E-4	0.02803183276725723
AAA	5	7.965329130743047E-4	0.02900859702578488
AAA	6	0.0010186889789549863	0.02637577423649819
AAA	7	5.894863792346425E-4	0.023401730454812385
AAA	8	0.0013400638313324216	0.026450880794032322

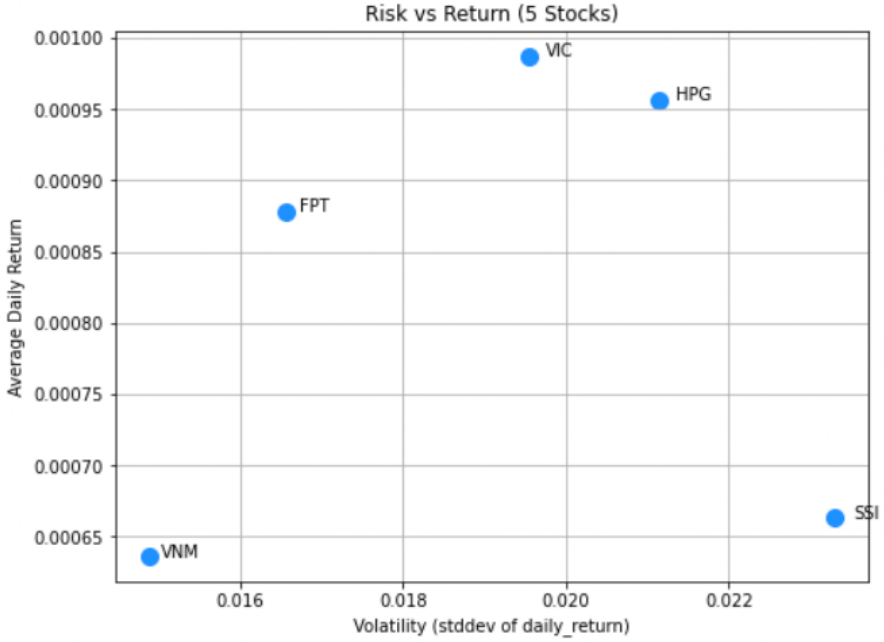
4.3 Phân tích Chi tiết 5 Mã Cổ phiếu Tiêu biểu

Phân tích sâu hơn được thực hiện trên nhóm 5 mã cổ phiếu: FPT, VNM, HPG, SSI, VIC.

- **Lọc Dữ liệu:** Tạo DataFrame mới chỉ chứa dữ liệu của 5 mã này.
- **Thống kê Mô tả:** Tính toán các chỉ số thống kê cơ bản cho nhóm 5 mã.
- **Trực quan hóa So sánh:** Chuyển đổi kết quả thống kê sang Pandas DataFrame và sử dụng Matplotlib để vẽ biểu đồ
  - Biểu đồ cột so sánh `avg_volume`.



  - Biểu đồ phân tán thể hiện mối quan hệ giữa Rủi ro (`volatility`) và Lợi suất (`avg_return`).

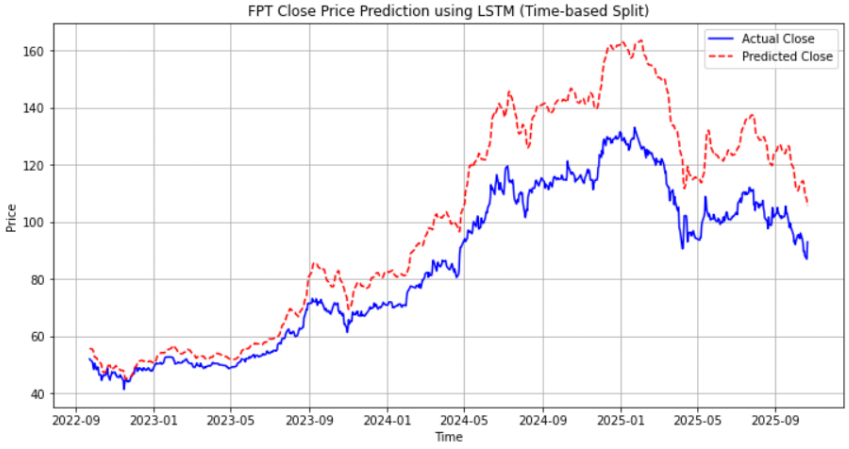


  - Biểu đồ đường biểu diễn xu hướng `avg_close` theo thời gian.
  - **Phân tích Hiệu suất Tháng:** Tính toán và hiển thị `avg_month_return` và `month_volatility` theo tháng cho 5 mã này.

4.4 Dự đoán Giá Cổ phiếu FPT bằng Mô hình LSTM

Phần cuối cùng tập trung vào việc xây dựng mô hình dự đoán chuỗi thời gian cho giá đóng cửa của cổ phiếu FPT.

- **Chuẩn bị Dữ liệu Dự đoán:**
  - Lọc dữ liệu chỉ cho mã FPT, chọn cột `time` và `close`, và chuyển đổi sang Pandas DataFrame, sắp xếp theo thời gian.
  - Chia dữ liệu thành tập huấn luyện (80%) và tập kiểm tra (20%).
  - Chuẩn hóa dữ liệu giá đóng cửa về khoảng [0, 1] sử dụng `MinMaxScaler`.
  - Tạo các chuỗi dữ liệu đầu vào (X) và đầu ra (y) cho mô hình LSTM. Mỗi chuỗi đầu vào bao gồm giá đóng cửa của `seq_length` (đặt là 20) ngày liên tiếp, và đầu ra là giá đóng cửa của ngày tiếp theo.
  - Định dạng lại dữ liệu đầu vào thành dạng 3 chiều phù hợp với lớp LSTM của Keras.
- **Xây dựng Mô hình LSTM:**
  - Sử dụng Keras `Sequential` API để xây dựng mô hình.
  - Mô hình bao gồm hai lớp LSTM (với 50 units mỗi lớp) và các lớp `Dropout` (với tỷ lệ 0.2) để giảm overfitting. Lớp LSTM đầu tiên có `return_sequences=True`.
  - Lớp cuối cùng là một lớp `Dense` với 1 unit để dự đoán giá đóng cửa.
  - Mô hình được biên dịch (compile) với optimizer là `adam` và hàm mất mát là `mean_squared_error`.
- **Huấn luyện Mô hình:**
  - Mô hình được huấn luyện trên dữ liệu `X_train`, `y_train` trong 50 epochs, kích thước batch là 32. Một phần dữ liệu huấn luyện (10%) được sử dụng làm tập validation.
- **Dự đoán và Trực quan hóa:**
  - Mô hình đã huấn luyện được sử dụng để dự đoán giá trên tập `X_test`.
  - Kết quả dự đoán (đã được chuẩn hóa) được biến đổi ngược lại về thang đo giá gốc bằng `scaler.inverse_transform()`. Giá trị thực tế `y_test` cũng được biến đổi ngược.
  - Sử dụng Matplotlib để vẽ biểu đồ so sánh giá đóng cửa thực tế và giá dự đoán trên tập kiểm tra theo thời gian



5 Kết luận

Trong project này, em đã xây dựng và triển khai một hệ thống Big Data phân tích dữ liệu chứng khoán sử dụng Docker, Hadoop và Spark. Hệ thống đã được thiết kế với các đặc điểm chính:

1. **Thu thập dữ liệu tự động và đầy đủ**: Dữ liệu chứng khoán được crawl từ VNStock cho tất cả các mã niêm yết, đảm bảo dữ liệu lịch sử từ 2010 đến hiện tại.
2. **Lưu trữ phân tán với HDFS**: Sử dụng NameNode và bốn DataNode, hệ thống đảm bảo tính sẵn sàng cao, khả năng mở rộng và bảo vệ dữ liệu trước sự cố.

3. **Xử lý song song bằng Spark**: Cluster Spark Master – Worker cho phép xử lý lượng lớn dữ liệu chứng khoán một cách nhanh chóng và hiệu quả.
4. **Trực quan hóa và phân tích dữ liệu**: Jupyter Notebook kết nối trực tiếp với Spark giúp thực hiện thống kê, phân tích và trực quan hóa dữ liệu một cách linh hoạt.

Hệ thống đã chứng minh được khả năng tích hợp công nghệ Big Data để thu thập, lưu trữ, xử lý và phân tích dữ liệu chứng khoán một cách hiệu quả. Đồng thời, việc triển khai trên Docker giúp môi trường phát triển và thử nghiệm nhanh chóng, đồng nhất và dễ nhân rộng.

Trong tương lai, hệ thống có thể được mở rộng để:

- Thu thập dữ liệu từ nhiều nguồn hơn, bao gồm dữ liệu tài chính quốc tế hoặc dữ liệu thời gian thực.
- Xây dựng các mô hình dự đoán giá cổ phiếu hoặc phân tích rủi ro dựa trên dữ liệu Big Data.
- Tối ưu hóa cluster Spark và HDFS để xử lý dữ liệu lớn hơn, phức tạp hơn.

Nhìn chung, dự án đã cung cấp một mô hình minh họa hoàn chỉnh về pipeline Big Data, từ thu thập, lưu trữ, xử lý đến phân tích dữ liệu, đồng thời mở ra cơ hội nghiên cứu và ứng dụng thêm các kỹ thuật phân tích dữ liệu nâng cao trong tài chính.