Dự đoán giá cổ phiếu của Apple

Trần Nhật Tiến[1][18521493], Kiên Tiến Đạt[2][18520568], Nguyễn Tấn Phong[3][18521239], Trần Cao Phát[3][18521233], and Đỗ Trọng Hợp

Trường Đại học Công nghệ Thông tin - Đại học Quốc gia Thành phố Hồ Chí Minh

Tóm tắt nội dung Trong bài báo này, chúng tôi đề xuất các phương pháp dự đoán mô hình hồi quy được xây dựng trên kiến trúc học sâu để dự đoán khả năng chính xác về giá trị tương lai của một cổ phiếu.

Keywords: LSTM · Pyspark · Gradient-boosted Tree Regression

1 Giới thiệu

Xây dựng các mô hình dự đoán để dự đoán chính xác và mạnh mẽ về giá cổ phiếu và biến động giá cổ phiếu là một vấn đề nghiên cứu đầy thách thức vì giả định giá cổ phiếu hoàn toàn là ngẫu nhiên về bản chất. Việc dự đoán giá cổ phiếu có thể giúp các nhà đầu tư trên thị trường chứng khoán trong việc xây dựng chiến lược đầu tư trên thị trường để tối đa hóa lợi nhuận của họ. Trong bài toán này chúng tôi đề xuất các mô hình để dự đoán giá cổ phiếu Apple gồm ... Mỗi mô hình có một kiến trúc khác, các hình dạng khác nhau của dữ liệu đầu vào và các giá trị siêu tham số khác nhau.

- Input : các giá trị của giá cổ phiếu trong thị trường chứng khoán
- Output : giá cổ phiếu dự đoán

2 Công trình liên quan

Tài liệu về hệ thống và phương pháp dự báo giá chứng khoán khá phong phú. Nhiều đề xuất tồn tại trên các cơ chế, cách tiếp cận và khuôn khổ cho dự đoán giá cổ phiếu trong tương lai và mô hình biến động giá cổ phiếu. Trong bài báo này, chúng tôi tập trung vào sự phát triển phương pháp hồi quy trong PySpark, dựa vào thư viện học máy đã được tích hợp sẵn trong PySpark : spark.ml [1]. Apache Spark là một khung mã nguồn mở được tối ưu hóa cho tính toán phân tán của các tập dữ liệu lớn trên các cụm máy tính được giới thiệu bởi Zaharia và cộng sự 2010 [2]. Hồi quy tuyến tính được sử dụng để dự đoán một số biến phụ thuộc, y, dựa trên một số biến độc lập x1, x2, x3,... [3]. Rừng ngẫu nhiên là quần thể của các cây quyết định . Rừng ngẫu nhiên là một trong những mô hình máy học thành công nhất để phân loại và hồi quy[4]. Họ kết hợp nhiều cây quyết định để giảm nguy cơ trang bị quá mức. Giống như cây quyết định, các khu rừng ngẫu nhiên xử lý các đặc điểm phân loại, mở rộng sang cài đặt phân loại đa lớp, không yêu cầu chia tỷ lệ đối tượng và có thể nắm bắt các điểm không tuyến tính và các tương tác đối tượng . Trong bài toán, tôi sẽ sử dụng thuật

toán Random Forest (RF) làm công cụ phân loại và hồi quy với Spark 2.0 được giới thiệu bởi Huynh Mai Khue và cộng sự[5]. Gradient-boosted Tree (GBT) là tập hợp của cây quyết định . GBT đào tạo lặp đi lặp lại các cây quyết định để giảm thiểu hàm mất mát. Giống như cây quyết định, GBT xử lý các tính năng phân loại, mở rộng sang cài đặt phân loại đa lớp, không yêu cầu chia tỷ lệ tính năng và có thể nắm bắt các điểm không tuyến tính và tương tác tính năng[6] được giới thiệu bởi Joseph Bradley và Manish Made[7]. Các mô hình LSTM đã cải thiện nhược điểm của RNN thông thường hav-ing các gradient biến mất và bùng nổ được giới thiệu bởi Hochreitar etal [8]. Trong bài báo cáo này, chúng tôi sẽ sử dụng các mô hình này để giải quyết bài toán.

3 Bộ dữ liệu

Bộ dữ liệu chúng tôi xây dựng và sử dụng trong bài báo có tên là bộ dữ liệu giá cổ phiểu Apple. Bộ dữ liệu có 6 biến bao gồm : Date, Open, High, Low, Close, Volumn và biến mục tiêu là Adj Close.

Mô tả các biến :

- Date : Ngày giao dịch
- Open : Giá tại thời điểm bắt đầu giao dịch khi tiếng chuông mở cửa vang lên
- High : Giá cao nhất mà cổ phiếu được giao dịch trong một khoảng thời gian
- Low : Giá thấp nhất mà cổ phiếu được giao dịch trong một khoảng thời gian
- Close : Giá mua của một cổ phiếu riêng lẻ khi sàn giao dịch chứng khoán đóng cửa trong ngày
- Volume : Tổng số cổ phiếu được giao dịch trong một khoảng thời gian
- Adj Close : Giá đóng của điều chỉnh để phản ảnh giá trị chứng khoán sau khi tính toàn bất kỳ hành động của công ty

	Date	Open	High	Low	Close	Volume	Adj Close
109	2008-05-09	183.16	184.25	181.37	183.45	24038300	183.45
3461	1995-01-17	44.50	45.50	44.13	45.00	11806400	11.12
834	2005-06-22	38.26	38.60	38.14	38.55	15175900	38.55
4799	1989-10-02	44.50	44.75	43.75	44.38	4922400	10.34
4098	1992-07-10	46.00	46.25	44.88	45.75	5144400	10.96

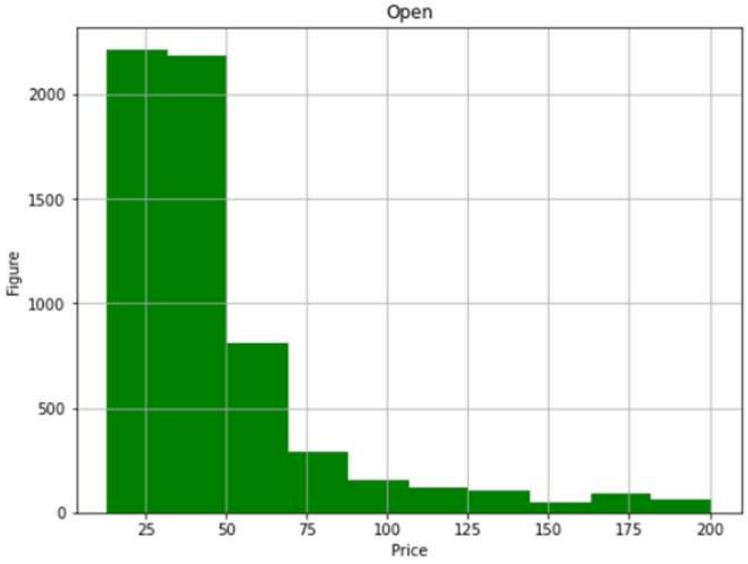
Hình 1. Ví dụ về bộ dữ liệu

- Mô tả bộ dữ liệu

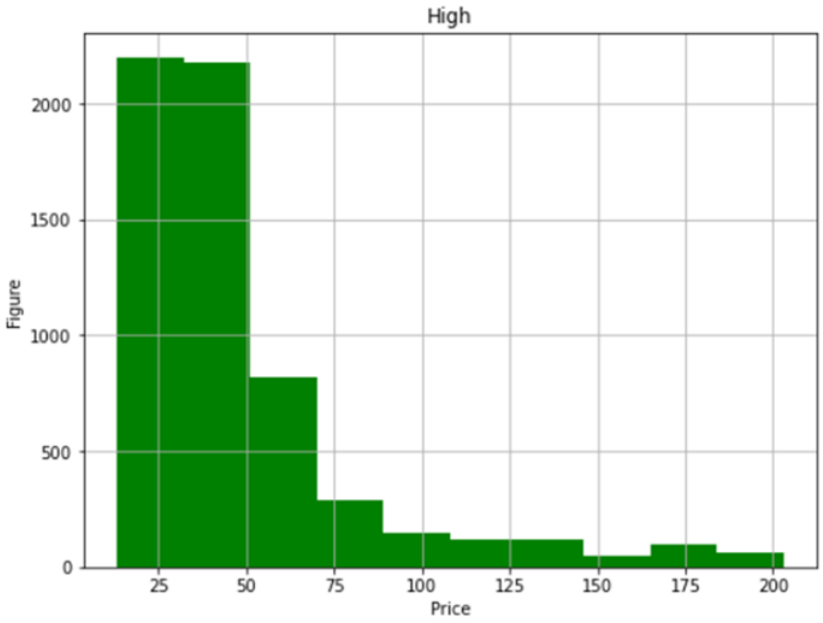
	Open	High	Low	Close	Volume	Adj Close
count	6081.000000	6081.000000	6081.000000	6081.000000	6.081000e+03	6081.000000
mean	46.823511	47.681506	45.913595	46.798619	1.363986e+07	23.529794
std	33.993517	34.578077	33.273106	33.947235	1.352107e+07	37.375601
min	12.880000	13.190000	12.720000	12.940000	8.880000e+04	1.650000
25%	24.730000	25.010000	24.200000	24.690000	5.530000e+06	7.380000
50%	38.250000	38.880000	37.460000	38.130000	8.976400e+06	9.910000
75%	53.500000	54.550000	52.500000	53.610000	1.631920e+07	14.360000
max	200.590000	202.960000	197.800000	199.830000	2.650690e+08	199.830000

Hình 2. Mô tả bộ dữ liệu

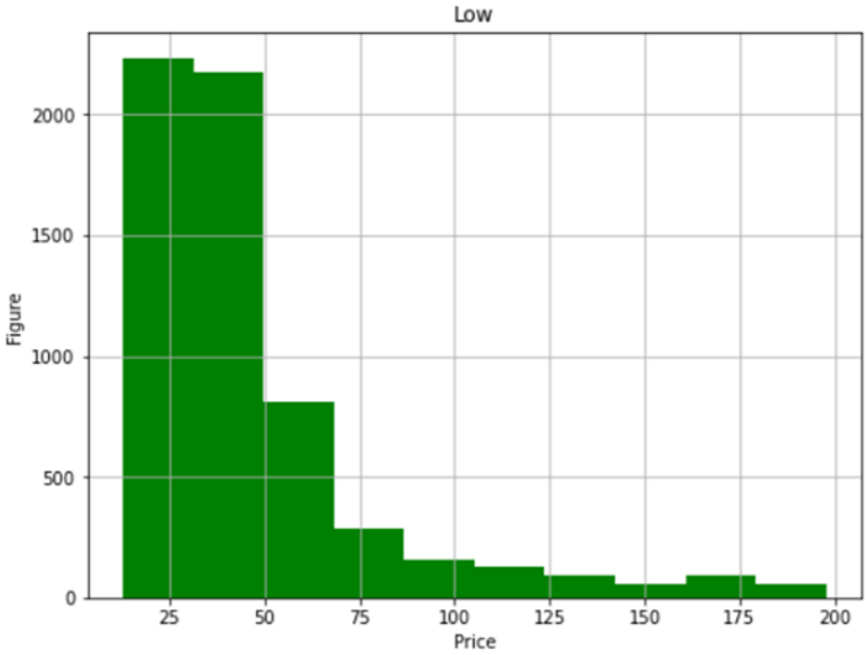
• Phân phối của dữ liệu theo từng nhãn



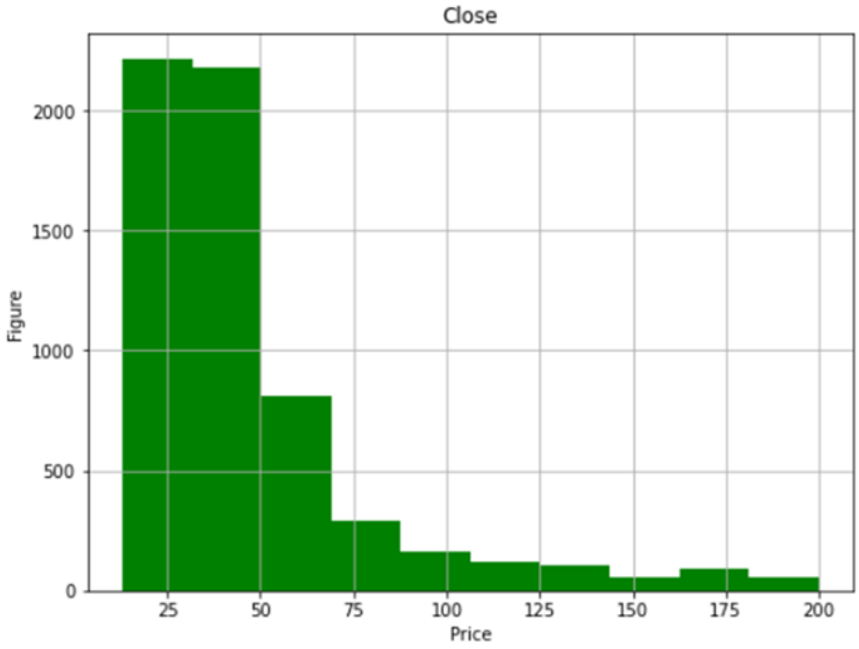
Hình 3. Phân phối của nhãn "Open"



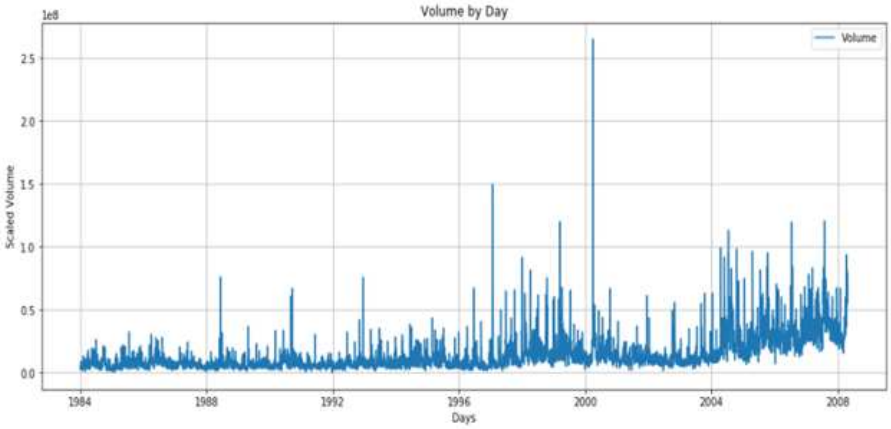
Hình 4. Phân phối của nhãn "High"



Hình 5. Phân phối của nhãn "Low"



Hình 6. Phân phối của nhãn "Close"



Hình 7. Trực quan nhãn "Volume"

4 Phương pháp thử nghiệm

Chúng tôi đã áp dụng 4 phương pháp trên bộ dữ liệu này: • Linear Regression: là một phương pháp thống kê để hồi quy dữ liệu với biến phụ thuộc có giá trị liên tục, trong khi các biến độc lập là giá trị phân loại. Nói cách khác "Hồi quy tuyến tính" là một phương pháp để dự đoán biến phụ thuộc (Y) dựa trên giá trị của biến độc lập (X).

• Random Forest Regression: là thuật toán supervised learning, có thể giải quyết cả bài toán hồi quy và phân loại. Random là ngẫu nhiên, Forest là rừng, nên ở thuật toán Random Forest mình sẽ xây dựng nhiều cây quyết định bằng thuật toán Decision Tree, tuy nhiên mỗi cây quyết định sẽ khác nhau (có yếu tố random).

• Gradient-boosted Tree Regression: là một tập hợp các mô hình cây quyết định để giải quyết các bài toán hồi quy và phân loại trong học máy. Mô hình sẽ tối thiểu hoá hàm mất mát, đồng thời giúp tăng độ chính xác của mô hình.

• Long Short Term Memory: 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 và 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.

5 Kết quả thử nghiệm

Chúng tôi sử dụng R-square để đánh giá mô hình, kết quả được trình bày trong bảng sau:

Mô hình	**R-square**
Linear Regression	0.8077
Random Forest Regression	0.9080
Gradient-boosted Tree Regression	0.9144
LSTM	0.8942

Bảng 1. Kết quả thử nghiệm

```
best_predictions.show(10)
```

```
+----------+-----+-----+-----+-----+--------+---------+------------------+
|      Date| Open| High|  Low|Close|  Volume|Adj Close|        prediction|
+----------+-----+-----+-----+-----+--------+---------+------------------+
|1984-09-17|28.62| 29.0|28.62|28.62| 6886400|     3.27|2.6305895628498046|
|1984-10-02|24.75|25.62|24.75|24.75| 4258400|     2.82|  2.55648784716353|
|1984-10-04|25.37|25.62|25.37|25.37| 4482400|     2.89| 2.585717013830197|
|1984-10-05|25.37|25.37|24.75|24.87| 3510400|     2.84| 2.585717013830197|
|1984-10-10|24.62|24.62|23.87|23.87|13070400|     2.72|  2.55648784716353|
|1984-10-11|23.87| 24.5|23.75|23.75| 6553600|     2.71|  2.55648784716353|
|1984-10-15| 24.0|24.25| 24.0| 24.0| 8715200|     2.74|  2.55648784716353|
|1984-10-16| 24.0|24.12|23.87|23.87| 4246400|     2.72|  2.55648784716353|
|1984-11-06|26.25|26.37|26.25|26.25| 8073600|      3.0| 2.585717013830197|
|1984-11-08|25.75|25.75|24.75|24.75| 3162400|     2.82|  2.55648784716353|
+----------+-----+-----+-----+-----+--------+---------+------------------+
only showing top 10 rows
```

Hình 8. Kết quả dự đoán và giá trị thực trên một sô mẫu thử

Mô hình Linear Regression là 1 mô hình cơ bản. Tuy nhiên, độ chính xác tính theo R-square khá cao, đạt 0.8077. Tiếp đến, 2 mô hình máy học tiên tiến hơn là Random Forest Regression và Gradient-boosted Tree Regression đạt được R-square khá tương đương nhau, cụ thể là 0.9080 và 0.9144. Đối với mô hình học sâu LSTM, độ chính xác giảm so với 2 mô hình vừa kể trên, chỉ đạt được 0.8942, tính theo R-square.

6 Kết luận và hướng phát triển

Bài báo cáo này mô tả quá trình áp dụng các mô hình máy học về hồi quy, cùng mô hình học sâu trên Apache Spark. Các mô hình đã được áp dụng vào thực nghiệm là Linear Regression, Random Forest Regression, Gradient-boosted Tree Regression và LSTM. Trong đó, mô hình Gradient-boosted Tree Regression đạt được kết quả R-square cao nhất, 0.9144.

Trong tương chúng tôi sẽ cố gắng cải thiện chất lượng của mô hình bằng việc thử nghiệm thêm các phương pháp tối ưu hóa tham số mô hình, cải tiến các phương pháp tiền xử lý đồng thử thử nghiệm trên các bộ dữ liệu mới hơn để có thể đem lại kết quả tốt nhất.

Tài liệu

1. https://spark.apache.org/docs/latest/ml-classification-regression.html
2. https://www2.eecs.berkeley.edu/Pubs/TechRpts/2010/EECS-2010-53.html
3. https://spark.apache.org/docs/latest/api/python/reference/api/pyspark.ml.regression.LinearRegressio
4. https://spark.apache.org/docs/latest/mllib-ensembles.html#random-forestsx
5. https://helpex.vn/article/rung-ngau-nhien-nhu-mot-phan-loai-mot-giai-phap-dua-tren-spark-5c6b0e16a

6. https://spark.apache.org/docs/latest/mllib-ensembles.html#gradient-boosted-trees-gbts
7. https://databricks.com/blog/2015/01/21/random-forests-and-boosting-in-mllib.html.
8. M. Zaharia, M. Chowdhury, M. Franklin, S. Shenker, I. Stoica, "Spark: cluster computing with working sets.",HotCloud, vol.10, pp. 10-10.
9. Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. Neural computation 9, 8 (1997), 1735–1780.
10. Elephas: Distributed Deep Learning with Keras and Spark, https://github.com/maxpumperla/elephas.
11. E. Nardo, "Distributed implementation of a LSTM on Spark and Tensorflow", 2016