

A faint, light-gray network graph with numerous circular nodes and connecting lines, centered behind the text.

IAI-UET

VIỆN TRÍ TUỆ NHÂN TẠO

ĐẠI HỌC CÔNG NGHỆ - ĐẠI HỌC QUỐC GIA HÀ NỘI

YOUR ONLY LIMIT IS YOU



XÂY DỰNG HỆ THỐNG GIÁM SÁT RỦI RO TÀI CHÍNH VÀ PHÁT HIỆN GIAN LẬN THỜI GIAN THỰC

JQK TEAM

Nguyễn Minh Quân

23020417

Cao Minh Quang

23020411

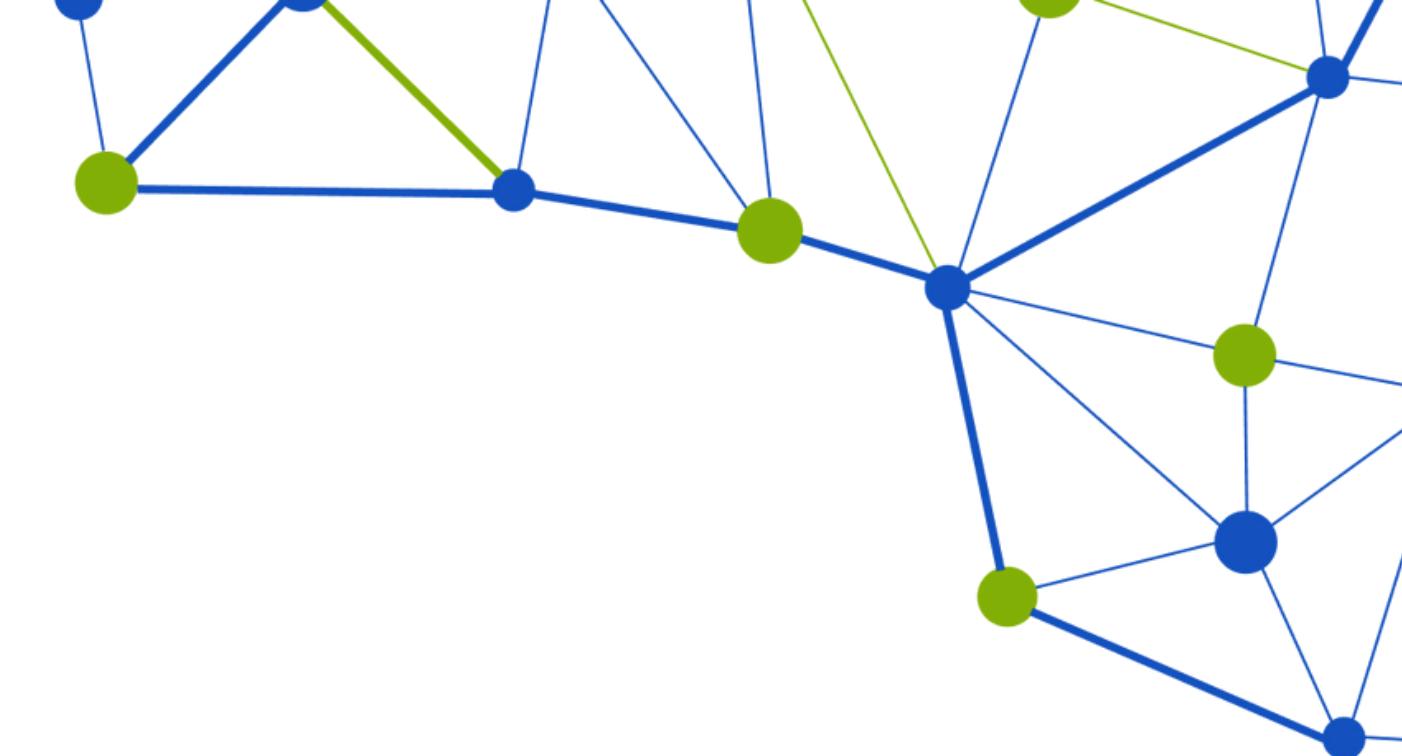
Nguyễn Năng Thịnh

23020439



NỘI DUNG

TRÌNH BÀY



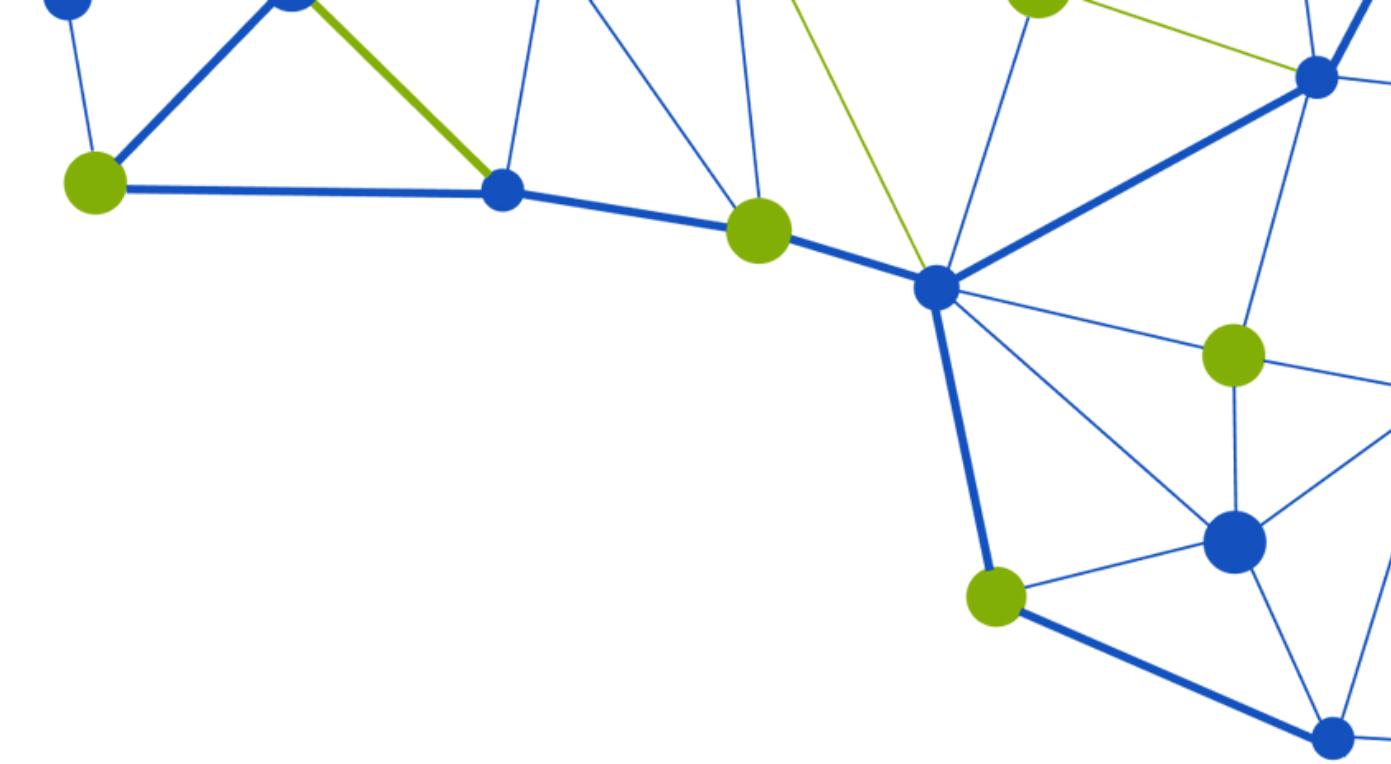
- 1 TỔNG QUAN
- 2 PHƯƠNG PHÁP
- 3 KẾT QUẢ
- 4 TỔNG KẾT

1.1.1 Bối cảnh và Tầm quan trọng

Bối cảnh: Ngành tài chính hiện đại vận hành dựa trên dữ liệu lớn, tốc độ cao và xử lý tức thì



Phát hiện gian lận và Phân tích rủi ro sẽ quyết định lợi thế cạnh tranh của một tổ chức.



1.1.2 Vấn đề và thách thức trong Big Data

Volume: Phải lưu trữ và xử lý hàng TB

dữ liệu lịch sử

Velocity: Hàng ngàn giao dịch mỗi
giây cần xử lý

Variety: Đa dạng nguồn dữ liệu từ
transaction, log, market data

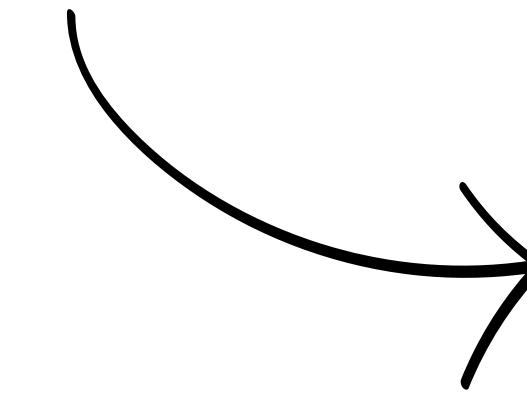
→ Do đó, một hệ thống tài chính hiện đại không thể chỉ dựa vào cơ sở dữ liệu quan hệ truyền thống mà bắt buộc phải áp dụng các kiến trúc và công nghệ dữ liệu lớn phân tán.

1.2 Kiến trúc hệ thống

- **Pipeline Big Data:** Sử dụng kiến trúc **Lambda Layer** (Hybrid Architecture) với khả năng xử lý song song 2 luồng dữ liệu thị trường và gian lận trên một hạ tầng.
- **Speed Layer 1:** Phát hiện gian lận giao dịch độ trễ thấp. **Kafka** đóng vai trò vùng đệm chịu tải cao, **Spark Streaming** xử lý từng micro-batch.
- **Speed Layer 2:** Tính toán > 30 chỉ số rủi ro chứng khoán thời gian thực và đẩy kết quả vào **Redis** để dashboard hiển thị. **Redis** là bộ nhớ đệm giúp Dashboard đọc dữ liệu nhanh, không làm nghẽn luồng tính toán.
- **Batch Layer:** Sử dụng dữ liệu lịch sử khổng lồ trên **HDFS** để re-train mô hình trên HDFS giúp hệ thống thông minh hơn theo thời gian.
- **Serving Layer:** Dashboard gom kết quả từ cả 2 luồng xử lý rồi rác vào một màn hình duy nhất để ra quyết định dựa trên dữ liệu tổng hợp.

1.2 Kiến trúc hệ thống

Vậy tại sao là Lambda Lai?

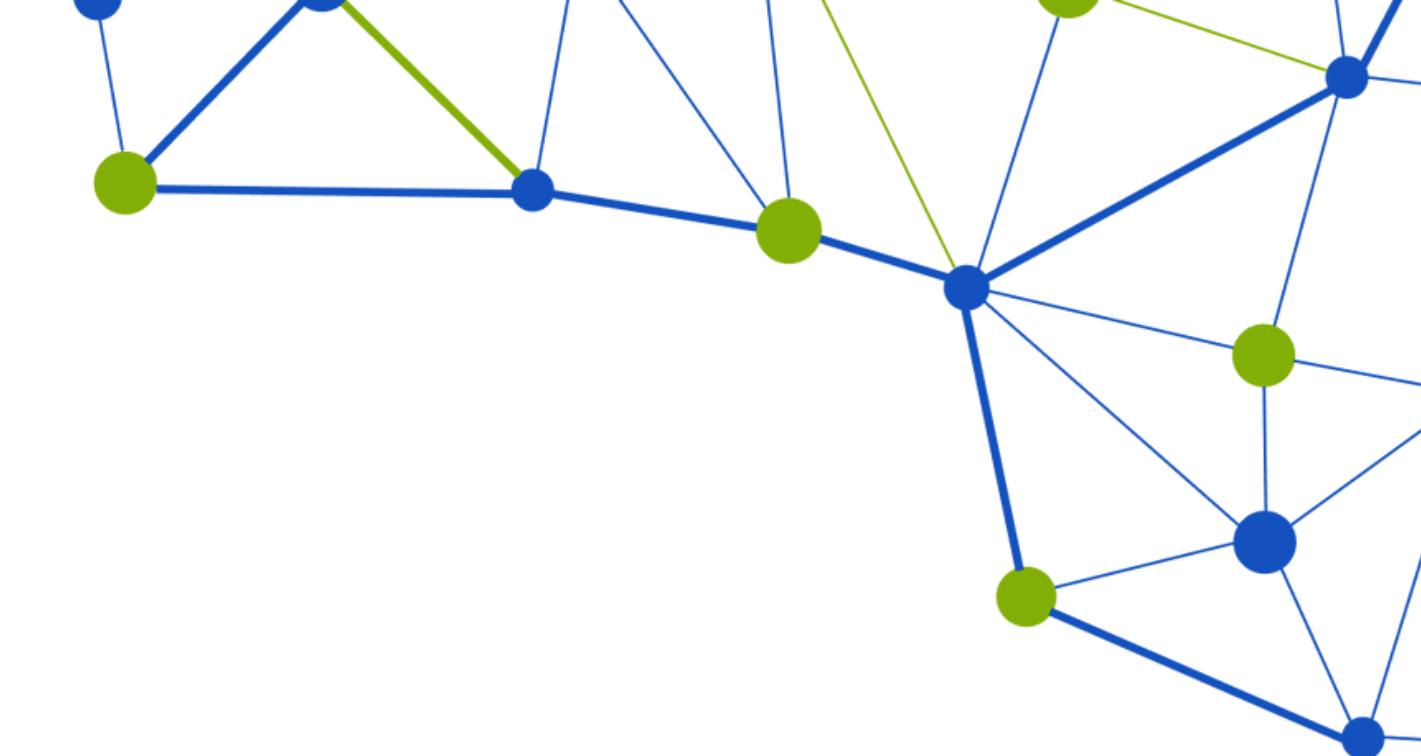


Kiến trúc Lambda truyền thống có 3 lớp (**Batch**, **Speed**, **Serving**). "Lai" ở đây nhấn mạnh việc tối ưu hóa: **Batch** không tính toán lại views mà **Speed** đã tính trước đó dẫn tới lãng phí tài nguyên, mà chỉ tập trung vào training, còn **Speed Layer** đảm nhiệm việc tính toán chỉ số.

NỘI DUNG

TRÌNH BÀY

- 1 TỔNG QUAN
- 2 PHƯƠNG PHÁP
- 3 KẾT QUẢ
- 4 TỔNG KẾT



2.1 Công nghệ sử dụng

- **Ingestion:** Apache Kafka và Zookeeper - "Xương sống" phân phối dữ liệu
- **Storage:** Hadoop - Data Lake lưu trữ dữ liệu lịch sử, Yarn quản lý cấp bộ nhớ cho Spark
- **Processing:** Apache Spark (Streaming, SQL, MLlib) - Bộ não xử lý in-memory

- **Serving:** Redis - Cache kết quả tốc độ cao cho Dashboard
- **Environment:** Docker và Docker Compose - Đóng gói và triển khai nhất quán
- **IDE:** Jupyter Notebook
- **Gateway:** Nginx

2.1 Công nghệ sử dụng

Vai trò	Lựa chọn	Lý do
Ingestion	Kafka, Zookeeper	Chịu được thông lượng cao, chịu lỗi tốt
Processing	Spark	Tốc độ xử lý in-memory nhanh cho ML và Streaming
Serving	Redis	Độ trễ thấp nhờ lưu trữ trên RAM
Environment	Docker	Nhỏ, khởi động nhanh, tiết kiệm tài nguyên và triển khai nhất quán

2.2 Thu thập dữ liệu

Dữ liệu Gian lận (PaySim):

- Giả lập giao dịch thời gian thực
- Mất cân bằng dữ liệu nghiêm trọng (99.9% hợp lệ vs 0.1% gian lận).

Dữ liệu phân tích rủi ro (VNStock):

- Dữ liệu thực từ thị trường chứng khoán Việt Nam (API vnstock).
- Gồm: Giá, khối lượng, mã cổ phiếu (Blue-chip: VCB, HPG, FPT...)

2.3 Phân tích và tiền xử lý dữ liệu

- Phân tích: Thống kê mô tả và trực quan hóa dữ liệu
- Tiền xử lý: Làm sạch, xử lý Missing Values, Outliers

- Feature Engineering:
 - Tạo đặc trưng mới: errorBalanceOrig (chênh lệch số dư bất thường).
 - Tính daily return, MA, RSI
- Chuẩn hóa: VectorAssembler và StandardScaler

2.4 Mô hình hóa và phân tích

Phát hiện gian lận:

- Lựa chọn mô hình GBTCClassifier bởi Ensemble mạnh mẽ, xử lý tốt dữ liệu phi tuyến và tương tác phức tạp giữa các đặc trưng.
- Random Undersampling hoặc SMOTE để cân bằng tập dữ liệu huấn luyện.

Phân tích rủi ro:

- Đọc dữ liệu từ Kafka, áp dụng hàng loạt các phép biến đổi và tính toán bằng **Spark SQL** để tạo ra một bộ gồm hơn 30 chỉ số chia thành 3 nhóm Thị trường, Rủi ro và Tiềm năng.
- Kết quả được đẩy vào **Redis** để Dashboard hiển thị và có thể truy vấn ngay lập tức.

2.4.1 Triển khai hệ thống

Batch Layer: Lưu trữ toàn bộ dữ liệu giao dịch lịch sử trên **HDFS** sau đó định kỳ re-train mô hình trên tập dữ liệu đó rồi lưu lại vào **HDFS** để **Speed Layer** sử dụng.

Speed Layer:

- Luồng phát hiện gian lận: Nhận dữ liệu giao dịch từ topic, áp dụng mô hình trên **HDFS** để dự đoán gian lận cho từng giao dịch sau đó lưu kết quả vào **Redis**.
- Luồng phân tích rủi ro thị trường: Nhận dữ liệu giá cổ phiếu từ topic, tính hơn 30 chỉ số và đẩy kết quả vào **Redis**.

Serving Layer:

Dashboard kết nối với **Redis** để hiển thị cảnh báo gian lận, trực quan hóa các chỉ số phân tích rủi ro bằng biểu đồ tương tác, bảng phân tích và cảnh báo rủi ro cao.

2.4.2 Quy trình thực hiện

Speed Layer 1:

- Producer đóng vai trò là nguồn phát, liên tục gửi dữ liệu vào Topic.
- Kafka hoạt động như vùng đệm trung gian đảm bảo giao dịch đáng tin cậy khi truyền tới Consumer.
- **Spark Streaming** liên tục đọc dữ liệu từ **Kafka** theo các micro-batch.
- Consumer tải mô hình GBTCClassifier đã được huấn luyện trước đó từ HDFS lên bộ nhớ.
- Nếu một giao dịch bị dự đoán là gian lận, **Dashboard** có thể trực tiếp đọc log hoặc tham chiếu đến bộ đếm trong Redis, hiển thị số lượng giao dịch nghi ngờ theo thời gian thực.

2.4.2 Quy trình thực hiện

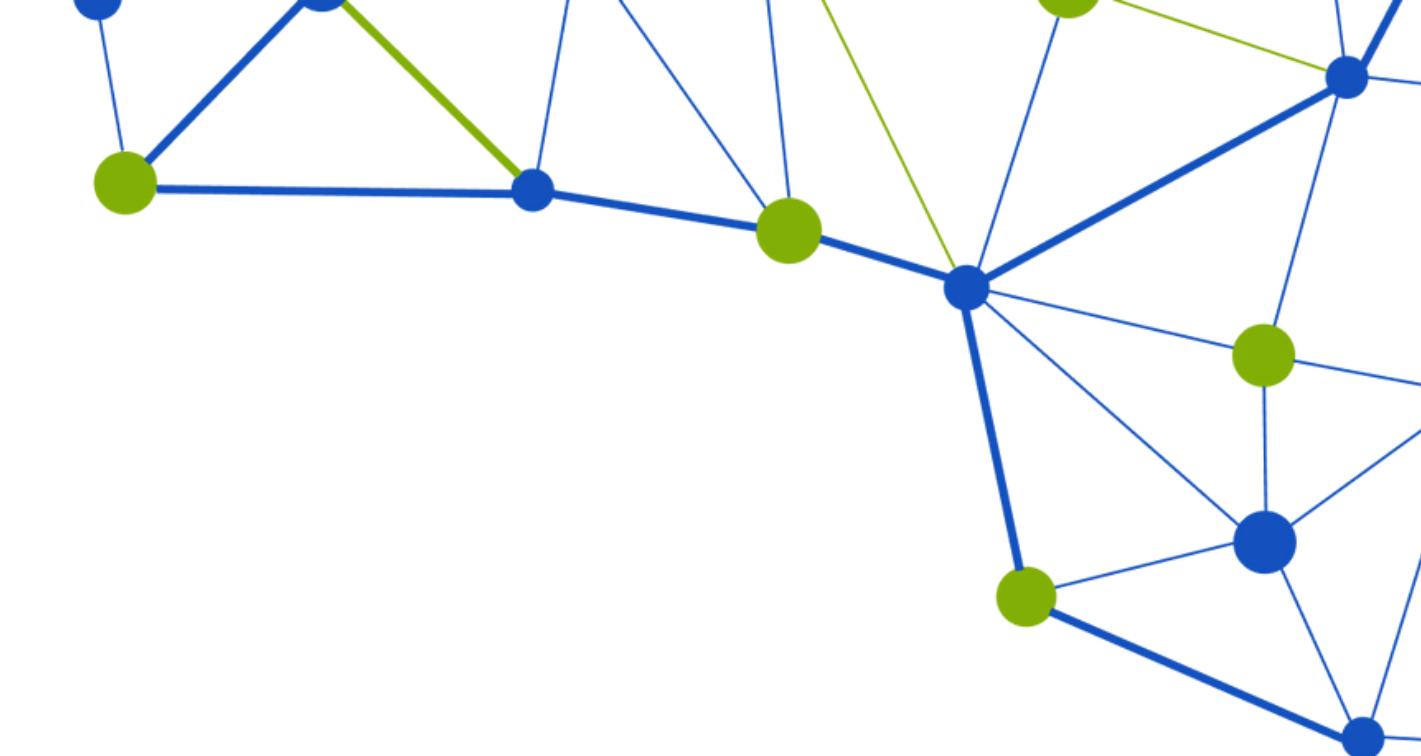
Speed Layer 2:

- Producer mỗi 15 giây thu thập dữ liệu bảng giá từ API vnstock, chuẩn hoá và gửi bản ghi vào Topic vietnam_stocks.
- **Spark Streaming** liên tục đọc dữ liệu từ **Kafka** theo các micro-batch.
- Hệ thống sẽ tính hơn 30 chỉ số thuộc 3 nhóm Thị trường, Rủi ro, Tiềm năng ngay khi dữ liệu xuất hiện, cho phép hệ thống phản ánh kịp thời biến động thị trường.
- Kết quả lưu vào **Redis** dưới dạng key-value giúp truy xuất nhanh.
- Các cổ phiếu có mức rủi ro cao được đưa vào Sorted Set để truy vấn, sắp xếp và cảnh báo theo thời gian thực.
- **Dashboard** giúp người dùng có thể xem biểu đồ tương tác, danh sách cảnh báo, và phân tích rủi ro được cập nhật liên tục từng giây.

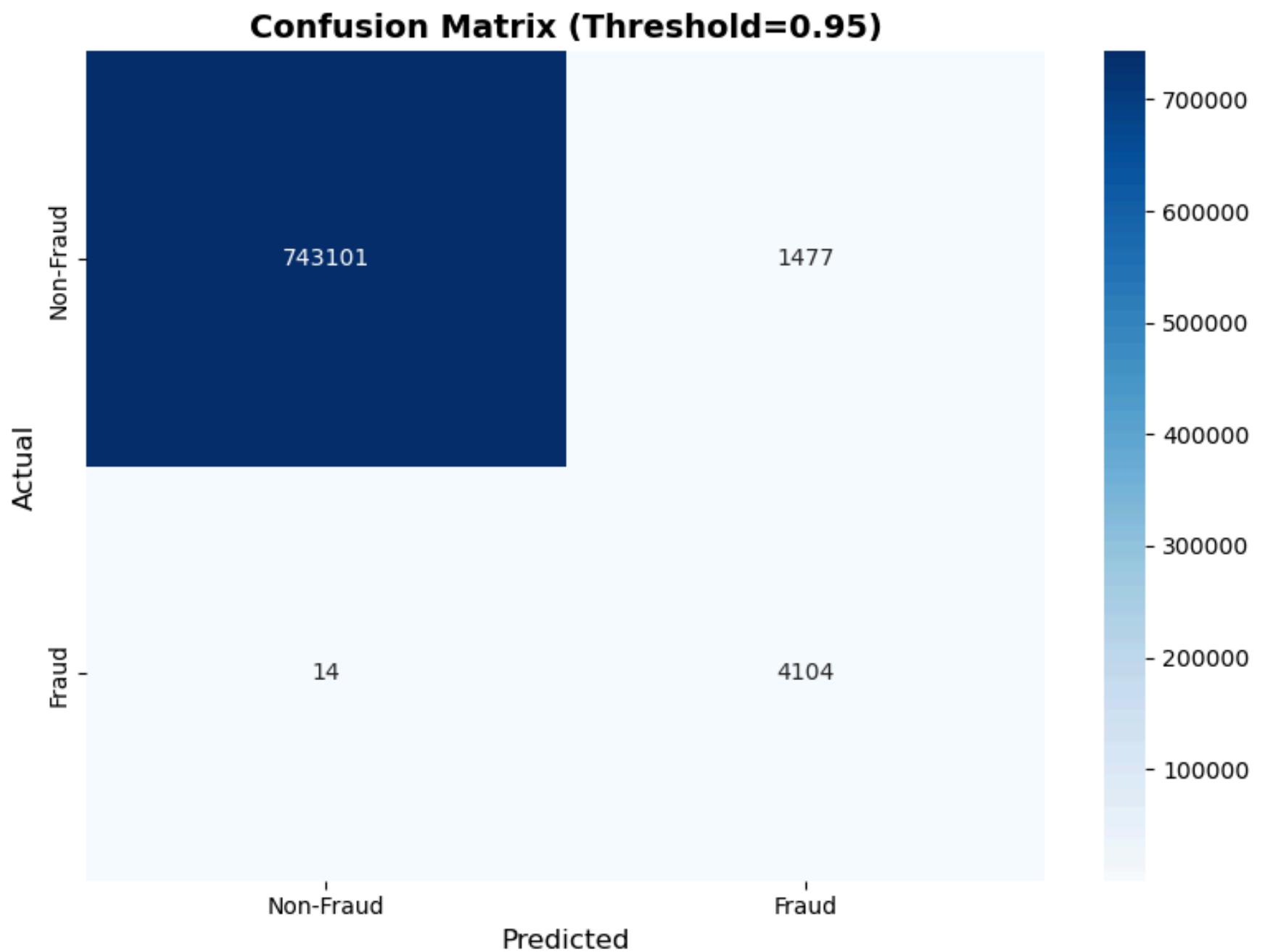
NỘI DUNG

TRÌNH BÀY

- 1 TỔNG QUAN
- 2 PHƯƠNG PHÁP
- 3 KẾT QUẢ
- 4 TỔNG KẾT



3.1.1 Kết quả Phát hiện gian lận



Mô hình nhận diện được **742,993** giao dịch trong đó có **4,104** giao dịch gian lận thực sự và **14** giao dịch gian lận bị bỏ sót. Bên cạnh đó, **1,645** giao dịch bị dự đoán sai.

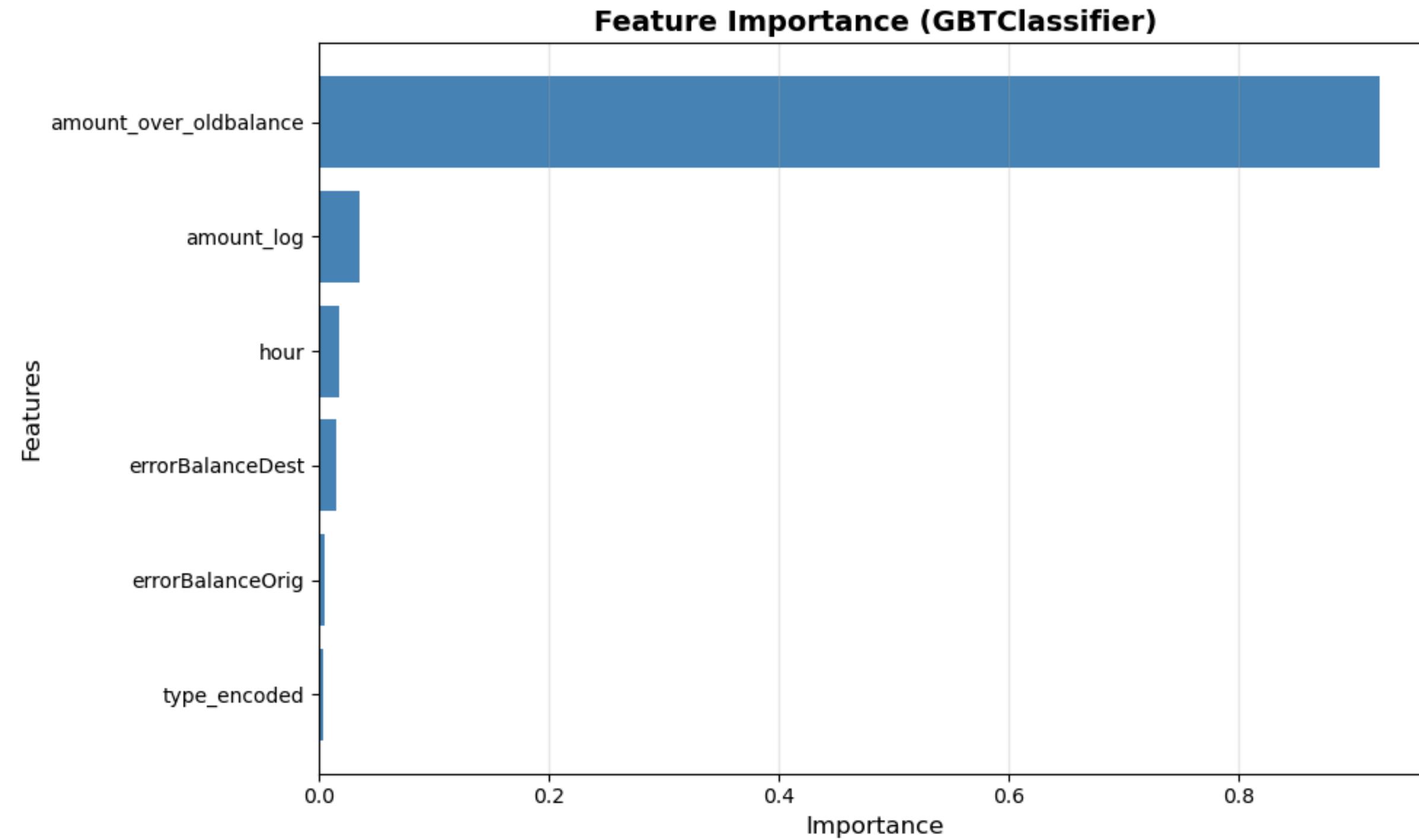
Từ đó, ta tính toán được độ phủ **99.66%** và nhận diện chính xác **71.41%** giao dịch. F1-Score đạt mức khoảng **83.2%** cho thấy hiệu suất của mô hình khá tốt. Điều này chứng tỏ mô hình không đoán bừa mà cố gắng giữ tỷ lệ nhầm ở mức kiểm soát được.

3.1.1 Kết quả Phát hiện gian lận

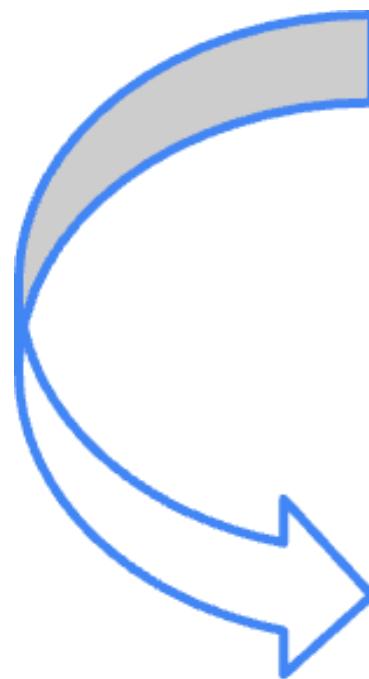
KẾT LUẬN

- Với độ phủ rất cao và chỉ có **14** ca bị bỏ sót cho thấy mô hình ưu tiên sự an toàn, giảm thiểu rủi ro.
- Tỉ lệ giao dịch thật **71.41%** ở mức chấp nhận được vì chi phí để xác minh **1,645** ca nghi ngờ thấp hơn nhiều so với thiệt hại nếu để lọt hơn **4,000** vụ gian lận.
- Thuật toán **Gradient-Boosted Trees** xử lý rất tốt bài toán mất cân bằng dữ liệu phân tách tốt giữa nhóm giao dịch gian lận và nhóm thường.

3.1.1 Kết quả Phát hiện gian lận



3.1.1 Kết quả Phát hiện gian lận



Đặc trưng **amount_over_oldbalance** chiếm tỉ trọng áp đảo với khoảng trên **92%** trong khi các đặc trưng còn lại đều không thể vượt trên **5%**. Đây là đặc trưng phái sinh, quan trọng nhất của mô hình.

Kẻ gian lận thường có xu hướng rút sạch tiền trong tài khoản hoặc chuyển một số tiền bất thường so với số dư gốc của tài khoản. Việc tập trung vào sự chênh lệch này là chìa khoá để ngăn chặn gian lận trong giao dịch.



3.1.2 Kết quả Phân tích Rủi ro Thị trường



Mã CP Giá (VND) Thay đổi % Điểm Rủi ro Rủi ro/Lợi nhuận Dòng vốn ngoại

Mã CP	Giá (VND)	Thay đổi %	Điểm Rủi ro	Rủi ro/Lợi nhuận	Dòng vốn ngoại
VCB	90,200	+1.32%	1.2 (Thấp)	1.8 (Hấp dẫn)	BUYING
HPG	28,500	-0.87%	1.5 (TB)	1.1 (Cân bằng)	SELLING
VNM	67,800	+0.44%	1.4 (Thấp)	1.5 (Hấp dẫn)	NEUTRAL
TCB	45,100	+2.15%	1.3 (Thấp)	2.1 (Rất hấp dẫn)	HEAVY_BUY
ABC	15,200	-4.10%	2.8 (Cao)	0.6 (Không hấp dẫn)	HEAVY_SELL

3.1.2 Kết quả Phân tích Rủi ro Thị trường

NHẬN XÉT

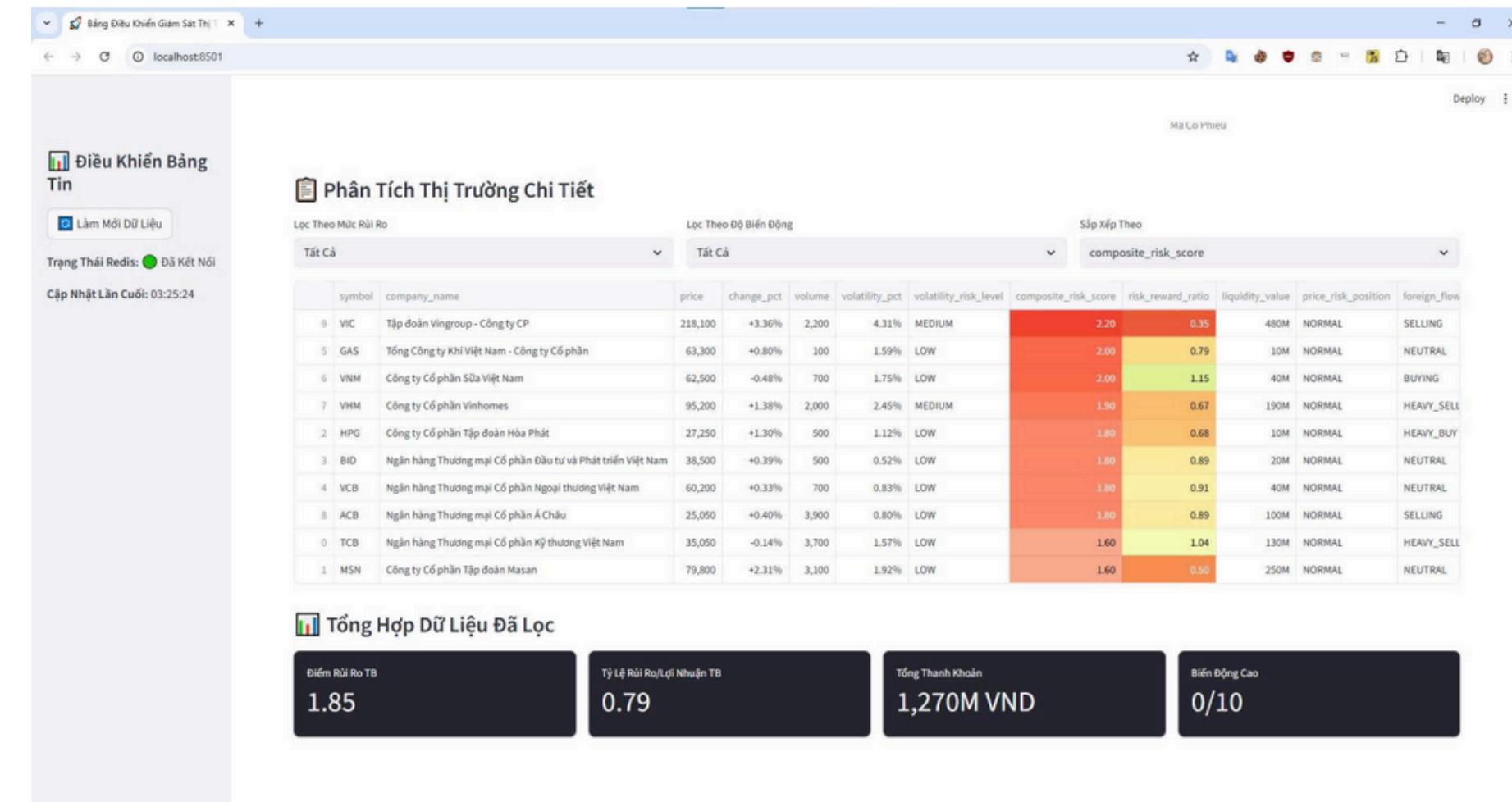
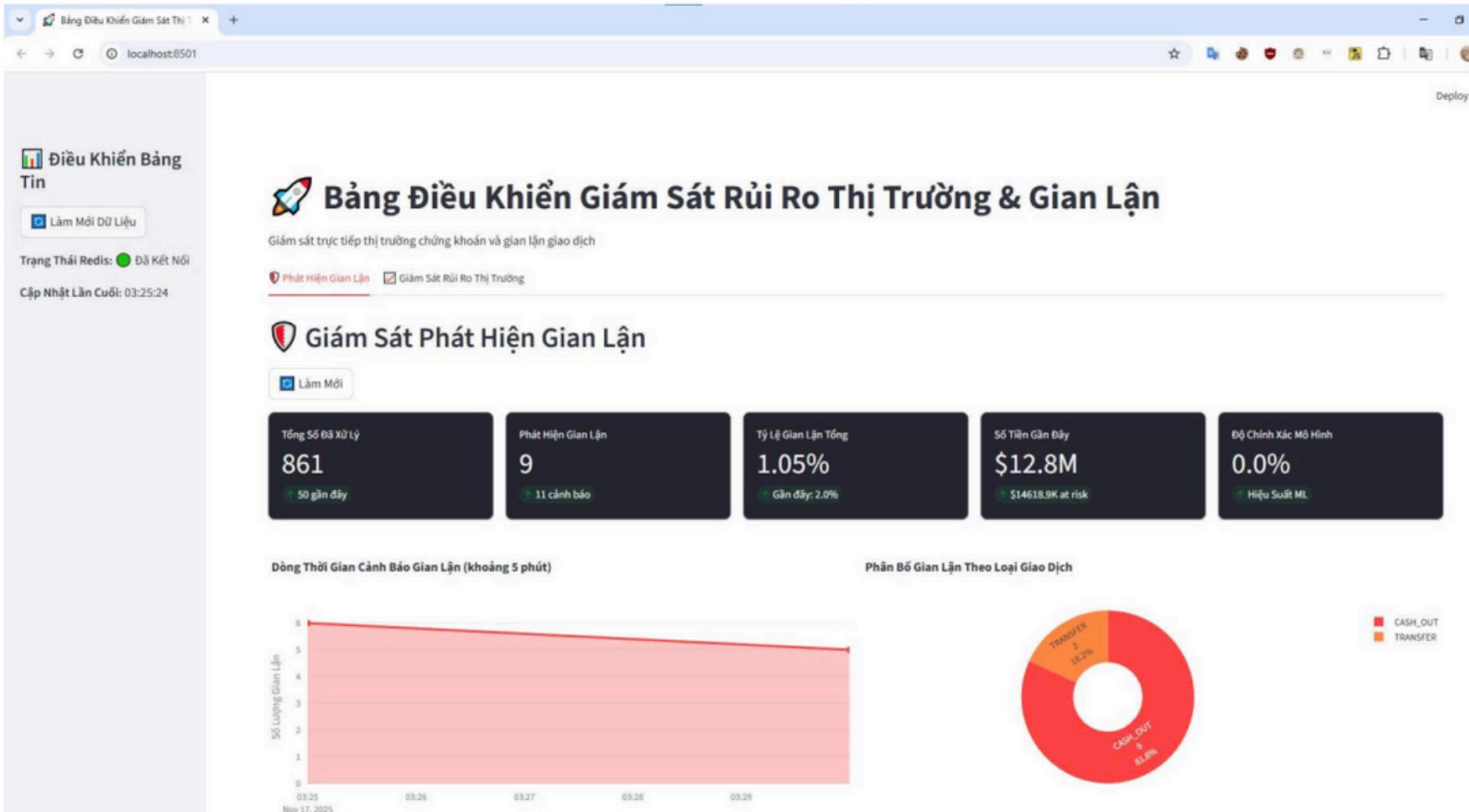
- Mã TCB giá tăng mạnh 2.15% và có dòng vốn ngoại mua mạnh (HEAVY_BUY). Từ đó, hệ thống chấm điểm rủi ro ở mức thấp là 1.3 và có tỷ lệ lợi nhuận ở mức rất hấp dẫn 2.1. Hệ thống nhận diện đúng đây là cổ phiếu đang có lực cầu tốt, an toàn để đầu tư tính toán chính xác.
- Mã ABC giá giảm mạnh 4.1% và bị bán tháo dẫn tới điểm rủi ro cao 2.8 và không hấp dẫn. Hệ thống phản ứng tức thì, đưa ra cảnh báo rủi ro cao.

→ Thay vì để con người tự dự đoán thì mô hình đưa ra con số **Composite Risk Score**. Các nhãn như (Hấp dẫn), (Cân bằng), (Thấp/Cao) giúp người dùng không chuyên cũng có thể hiểu ngay trạng thái thị trường.

3

KẾT QUẢ

3.1.3 Dashboard trực quan hóa



3.1.3 Dashboard trực quan hóa

- Được xây dựng bằng Dash/Plotly
- Là Serving Layer trong kiến trúc Lambda tinh chỉnh giúp hiển thị thông tin và kết nối 2 Speed Layer, cung cấp một tầm nhìn thống nhất hỗ trợ người dùng ra quyết định

- Speed Layer 1: Hiển thị số lượng và danh sách giao dịch đáng ngờ, có độ trễ < 2 giây
- Speed Layer 2: Kết nối với Redis, hiển thị Risk Heat Map, Risk-Reward Scatter Plot và bảng phân tích hơn 15 chỉ số rủi ro quan trọng với độ trễ dưới 10 giây

NỘI DUNG

TRÌNH BÀY

1

TỔNG QUAN

2

PHƯƠNG PHÁP

3

KẾT QUẢ

4

TỔNG KẾT



4.1 Ưu điểm của hệ thống

Hệ thống cung cấp kiến trúc Lambda lai tối ưu, với:

- **Speed Layer** xử lý song song 2 nghiệp vụ phát hiện gian lận và phân tích rủi ro.
- **Batch Layer** huấn luyện định kỳ mô hình học máy.

Hệ thống có khả năng mở rộng ngang (horizontal scaling) nhờ các công nghệ **Kafka/HDFS/Spark**.

Dễ dàng triển khai, quản lý hệ thống với hệ sinh thái Docker.

Dashboard cho phép người dùng theo dõi, phân tích và đưa ra quyết định dựa trên kết quả mô hình.

Hệ thống có tính đặc thù cao của ngành tài chính.

4.2 Nhược điểm và thách thức

Hệ thống có độ phức tạp cao, gồm nhiều thành phần khác nhau, gây khó khăn cho việc vận hành, bảo trì và debug.

Mô hình **GBTClassifier** còn khá cơ bản do chưa áp dụng những kỹ thuật tiên tiến như **Deep Learning** hay **GNN**.

Việc tinh chỉnh cấu hình để đạt hiệu quả cao trong môi trường thực tế là một công việc khó khăn và cần nhiều kinh nghiệm.

Speed Layer chỉ xử lý sự kiện một cách độc lập, điều này làm hạn chế khả năng phát hiện các mẫu phức tạp hơn.

4.3 Hướng phát triển cho hệ thống

Nghiên cứu thêm về việc áp dụng các mô hình **Deep Learning** ví dụ như **LSTM** để phát hiện mâu thuẫn gian lận dựa trên chuỗi giao dịch theo thời gian hoặc **GNN**, **GraphX** để phát hiện các mạng lưới gian lận dựa trên mối quan hệ giữa các tài khoản.

Xây dựng mô hình dự đoán giá cổ phiếu và tính VaR bằng phương pháp **Monte Carlo Simulation**.

Thêm biểu đồ tương tác để người dùng đi sâu vào các cảnh báo.

Dùng **Structured Streaming**, tích hợp **stateful streaming** để phân tích các mâu thuẫn vi phức tạp.

Tích hợp **Apache Airflow** để tự động hóa quy trình huấn luyện lại (MLOps).

