BÁO CÁO CUỘC THI

AUTOSCALING ANALYSIS
DỰ ĐOÁN LƯU LƯỢNG HTTP VÀ TỰ ĐỘNG MỞ RỘNG MÁY CHỦ

CUỘC THI: CUỘC THI DATAFLOW 2026 - NĂM: 2026

[Tên thành viên 1] [MSSV 1] [Email 1]
[Tên thành viên 2] [MSSV 2] [Email 2]
[Tên thành viên 3] [MSSV 3] [Email 3]

BAN TỔ CHỨC: DATAFLOW 2026

Mục lục

Chương 1

GIỚI THIỆU BÀI TOÁN

1.1 GIỚI THIỆU

Trong bối cảnh điện toán đám mây phát triển mạnh mẽ, việc tối ưu hóa tài nguyên thông qua Autoscaling (tự động mở rộng quy mô) đã trở thành một yêu cầu cấp thiết. Do lưu lượng truy cập web thường xuyên biến động, các hệ thống luôn phải đối mặt với bài toán cân bằng giữa chi phí và hiệu năng: cấp phát thừa tài nguyên (over-provisioning) sẽ gây lãng phí ngân sách, trong khi cấp phát thiếu (under-provisioning) lại dẫn đến quá tải và gián đoạn dịch vụ. Để giải quyết vấn đề này, các chiến lược Autoscaling thường được chia thành hai nhóm chính là Reactive Scaling và Predictive Scaling. Trong khi Reactive Scaling chỉ phản ứng dựa trên các chỉ số hiện tại như CPU hay RAM, thì Predictive Scaling lại tận dụng kỹ thuật dự báo chuỗi thời gian (Time series forecasting) để phân tích dữ liệu lịch sử, nhận diện các chu kỳ (mùa vụ) và chủ động điều chỉnh tài nguyên trước khi lưu lượng thực tế tăng lên, giúp giảm thiểu độ trễ phản ứng của hệ thống.

1.2 BÀI TOÁN HỒI QUY: DỰ ĐOÁN LƯU LƯỢNG

Trọng tâm của bài toán hồi quy là xây dựng các mô hình máy học có khả năng dự báo chính xác hai chỉ số quan trọng: số lượng HTTP request và lượng dữ liệu (bytes) được chuyển tải trong tương lai. Để đảm bảo tính thực tiễn và linh hoạt, các mô hình này sẽ được huấn luyện và đánh giá trên ba khung thời gian khác nhau: cửa sổ 1 phút giúp hệ thống phản ứng nhanh với các biến động tức thời, cửa sổ 5 phút để cân bằng giữa độ nhạy và độ ổn định, và cửa sổ 15 phút phục vụ cho các chiến lược dài hạn. Dữ liệu huấn luyện được trích xuất từ logs của máy chủ NASA (giai đoạn 1/7 - 31/8/1995), bao gồm các thông tin chi tiết về host, thời gian, request và trạng thái phản hồi. Cần lưu ý rằng dữ liệu được phân chia theo trình tự thời gian nghiêm ngặt với tập Training từ 1/7 đến 22/8 và tập Test cho giai đoạn còn lại, đồng thời đã xử lý các khoảng downtime do sự cố thiên tai bằng dữ liệu giả lập.

Về phương pháp luận, dự án sẽ triển khai và so sánh hiệu quả giữa các nhóm kỹ thuật khác nhau, bao gồm các phương pháp thống kê truyền thống như ARIMA/SARIMA, các mô hình Deep Learning như LSTM/RNN và các thuật toán Machine Learning hiện đại như XGBoost hay LightGBM. Mô hình tối ưu nhất cho mỗi khung thời gian sẽ được lựa chọn dựa trên bộ chỉ số đánh giá toàn diện. Cụ thể, RMSE sẽ được dùng để phạt nặng các sai số lớn, trong khi MSE và MAE cung cấp cái nhìn tổng quát ít nhạy cảm với ngoại lai hơn, và MAPE sẽ giúp

đánh giá mức độ sai số dưới dạng phần trăm trực quan.

1.3 Bài toán tối ưu: Autoscaling

Bài toán tối ưu hóa đóng vai trò chuyển đổi các kết quả dự báo thành hành động cụ thể nhằm tìm ra chính sách autoscaling cân bằng nhất giữa việc giảm thiểu chi phí vận hành (tính theo server-hours) và đảm bảo cam kết chất lượng dịch vụ (SLA). Hệ thống sẽ triển khai ba chiến lược để so sánh: Scaling dựa trên CPU (điều chỉnh khi tải vượt ngưỡng 70% hoặc giảm dưới 30%), Scaling dựa trên số lượng request thực tế, và Predictive Scaling sử dụng kết quả từ mô hình học máy để đón đầu xu hướng. Để đảm bảo hệ thống hoạt động ổn định và tránh hiện tượng dao động liên tục (oscillation), các cơ chế như thời gian chờ (Cooldown Period) và ngưỡng trễ (Hysteresis) cũng được áp dụng chặt chẽ trong quá trình vận hành. Hiệu quả cuối cùng của các chính sách này sẽ được đo lường dựa trên tổng chi phí, tỷ lệ thời gian hệ thống không bị quá tải, tần suất thay đổi trạng thái máy chủ và độ trễ trong phản ứng.

1.4 Mục tiêu dự án

Dự án hướng tới việc xây dựng một quy trình khép kín từ phát triển mô hình đến triển khai hệ thống thực nghiệm.

Ở giai đoạn phát triển, mục tiêu là triển khai tối thiểu hai mô hình thuộc các nhóm thuật toán khác nhau cho

Chương 2

XỬ LÝ DỮ LIỆU

2.1 TỔNG QUAN BỘ DỮ LIỆU NASA HTTP LOGS

Bộ dữ liệu được sử dụng trong dự án là **NASA WWW Server Logs**, một tập dữ liệu kinh điển trong lĩnh vực phân tích lưu lượng mạng và chuỗi thời gian. Dữ liệu này ghi lại toàn bộ các request HTTP đến máy chủ WWW của NASA trong khoảng thời gian hai tháng, từ ngày 1/7/1995 đến ngày 31/8/1995, với độ phân giải thời gian là 1 giây. Tổng số lượng request trong bộ dữ liệu là khoảng 3.46 triệu dòng log, cung cấp một bức tranh toàn diện về mẫu hình truy cập web vào giữa thập niên 1990.

Mỗi dòng log trong bộ dữ liệu NASA được lưu trữ theo định dạng ASCII chuẩn, chứa các trường thông tin quan trọng cho việc phân tích chuỗi thời gian và autoscaling. Các trường này bao gồm: **Host** là địa chỉ IP hoặc tên miền của máy khách gửi request; **Timestamp** là thời điểm request được thực hiện với định dạng `[dd/Mon/YYYY:HH:MM:SS -0400]`; **Request** là chuỗi request HTTP bao gồm phương thức (GET, POST, v.v.), đường dẫn URL và phiên bản giao thức; **Status Code** là mã phản hồi HTTP (ví dụ: 200 cho thành công, 404 cho not found, 500 cho lỗi server); và **Bytes** là số byte dữ liệu được trả về cho request.

Ví dụ một dòng log điển hình có dạng sau:

```
199.72.81.55 - - [01/Jul/1995:00:00:01 -0400] "GET /history/apollo/ HTTP/1.0" 200 6245
```

Trong ví dụ này, `199.72.81.55` là địa chỉ IP của client, `[01/Jul/1995:00:00:01 -0400]` là thời gian request với timezone UTC-4 (Eastern Time), `"GET /history/apollo/ HTTP/1.0"` là request HTTP GET đến đường dẫn `/history/apollo/`, `200` là mã trạng thái HTTP 200 (Success), và `6245` là số byte dữ liệu được trả về.

Một điểm quan trọng cần lưu ý về bộ dữ liệu này là sự tồn tại của các khoảng thời gian *system downtime* do bão. Theo tài liệu chính thức của NASA, máy chủ đã gặp sự cố từ ngày 01/08/1995 14:52:01 đến ngày 03/08/1995 04:36:13 do ảnh hưởng của bão. Tuy nhiên, thay vì để lại các khoảng trống trong dữ liệu, các khoảng thời gian này đã được điền bằng dữ liệu giả lập (mock data) để duy trì tính liên tục của chuỗi thời gian. Ngoài ra, dự án cũng phát hiện thêm một khoảng thời gian có dấu hiệu bất thường từ ngày 28/07/1995 13:32:26 đến ngày 01/08/1995 00:00:00, trong đó số lượng request giảm đáng kể so với bình thường. Để đảm bảo chất lượng dữ liệu cho việc huấn luyện mô hình, các khoảng thời gian này được đánh dấu bằng cột `is_system_down` và các giá trị request được khôi phục bằng cách sử dụng dữ liệu từ tuần trước (shift 7 ngày).

2.2 Cấu trúc dữ liệu và Pipeline xử lý

2.2.1 Log Parser và Trích xuất trường

Pipeline xử lý dữ liệu bắt đầu bằng việc đọc và parse các file log thô từ thư mục `data/raw/`. Do dữ liệu được lưu trữ dưới dạng ASCII với cấu trúc cố định, chúng tôi sử dụng Regular Expression để trích xuất các trường thông tin một cách hiệu quả và chính xác. Biểu thức Regular Expression được sử dụng có dạng sau:

```
(?P<host>\S+) - - \[(?P<timestamp>.*?)\] "(?P<request>.*?)" (?P<status>\d{3}) (?P<bytes>
```

Hàm `parse_log_line` chịu trách nhiệm chuyển đổi mỗi dòng log thành một dictionary với các trường đã được trích xuất. Trong quá trình này, chúng tôi thực hiện các bước xử lý sau. Đầu tiên, trích xuất **Host** bằng cách lấy địa chỉ IP hoặc tên miền của client. Tiếp theo, parse **Timestamp** và chuyển đổi chuỗi timestamp sang định dạng datetime với timezone UTC-4 để đảm bảo tính chính xác về thời gian. Sau đó, trích xuất **Request** để lấy chuỗi request HTTP đầy đủ bao gồm phương thức, đường dẫn và phiên bản giao thức. Tiếp theo, parse **Status Code** và chuyển đổi mã trạng thái sang dạng integer để dễ dàng xử lý. Cuối cùng, parse **Bytes** và chuyển đổi số byte sang dạng integer, xử lý trường hợp giá trị là `'-'` (không có dữ liệu trả về) bằng cách gán giá trị 0.

Sau khi parse tất cả các dòng log, dữ liệu được chuyển đổi thành **pandas DataFrame** và sắp xếp theo thứ tự thời gian để đảm bảo tính liên tục của chuỗi thời gian. Điều này rất quan trọng vì các mô hình chuỗi thời gian yêu cầu dữ liệu được sắp xếp theo trình tự thời gian để học được các mẫu hình phụ thuộc vào quá khứ.

2.2.2 Aggregation theo thời gian

Do dữ liệu gốc có độ phân giải là 1 giây, chúng tôi thực hiện aggregation theo các cửa sổ thời gian khác nhau để phù hợp với các mục đích phân tích và mô hình hóa. Ba cửa sổ thời gian được sử dụng trong dự án là: cửa sổ **1-minute (1m)** với 89,280 điểm dữ liệu, phù hợp cho việc phản ứng nhanh với các biến động tức thời; cửa sổ **5-minute (5m)** với 17,856 điểm dữ liệu, cân bằng tốt giữa độ nhạy và độ ổn định; và cửa sổ **15-minute (15m)** với 5,952 điểm dữ liệu, phù hợp cho các chiến lược dài hạn và giảm nhiễu.

Hàm `resample_traffic` thực hiện aggregation với các bước sau. Đầu tiên, thiết lập cột `timestamp` làm index của DataFrame. Tiếp theo, sử dụng phương thức `resample` của pandas với các cửa sổ thời gian `'1min'`, `'5min'`, và `'15min'`. Sau đó, áp dụng các hàm aggregation cho từng trường: `requests` sử dụng `count` để đếm số lượng request trong mỗi cửa sổ; `bytes` sử dụng `sum` để tổng số byte được chuyển tải; `hosts` sử dụng `nunique` để đếm số lượng host duy nhất (đại diện cho số lượng người dùng); và `errors` sử dụng `lambda x: (x >= 400).sum()` để đếm số lượng lỗi (status code ≥ 400). Cuối cùng, **Fill missing values** bằng cách điền 0 cho các khoảng thời gian không có request để đảm bảo tính liên tục của chuỗi thời gian.

2.2.3 Xử lý missing values và anomalies

Sau khi aggregation, chúng tôi thực hiện các bước xử lý để đảm bảo chất lượng dữ liệu. Đầu tiên, xử lý **System Downtime** bằng cách tạo cột `is_system_down` để đánh dấu các khoảng

thời gian có dữ liệu giả lập. Các giá trị request trong khoảng này được khôi phục bằng cách sử dụng dữ liệu từ tuần trước (shift 2016 bước cho 5-minute window, tương ứng với 7 ngày). Sau đó, thực hiện **Forward-fill và Backward-fill** bằng cách sử dụng phương thức forward-fill (sử dụng giá trị trước đó) và backward-fill (sử dụng giá trị sau đó) để đảm bảo tính liên tục của chuỗi thời gian. Tiếp theo, loại bỏ các giá trị infinity có thể xuất hiện sau khi tính toán các đặc trưng thống kê. Cuối cùng, thực hiện **Type casting** bằng cách chuyển đổi các cột sang kiểu dữ liệu phù hợp (int cho requests, bytes, hosts; float cho các đặc trưng thống kê) để tối ưu hóa bộ nhớ và tốc độ xử lý.

2.3 PHÂN TÍCH KHÁM PHÁ DỮ LIỆU (EDA)

2.3.1 Thống kê tổng quan

Sau khi xử lý và làm sạch dữ liệu, chúng tôi thực hiện phân tích khám phá dữ liệu (Exploratory Data Analysis - EDA) để hiểu rõ các đặc điểm và mẫu hình của lưu lượng truy cập NASA. Thống kê cơ bản cho khung 5-minute cho thấy tổng số điểm dữ liệu là 17,856, khoảng thời gian từ 01/07/1995 00:00:00 đến 31/08/1995 23:55:00 (UTC-4). Trung bình số request mỗi 5 phút là khoảng 200-300 request. Peak traffic đạt mức cao nhất vào các giờ làm việc trong ngày (9h-17h) phản ánh hành vi người dùng điển hình vào giờ làm việc. Low traffic giảm đáng kể vào các giờ đêm (0h-6h) và cuối tuần, phản ánh sự khác biệt trong hành vi truy cập giữa ngày làm việc và thời gian nghỉ ngơi.

2.3.2 Phân phối Status Codes

Phân tích mã trạng thái HTTP cho thấy các đặc điểm sau. Mã **200 (Success)** chiếm phần lớn các request, cho thấy hệ thống hoạt động ổn định và trả về dữ liệu thành công cho hầu hết các request. Mã **304 (Not Modified)** xuất hiện thường xuyên do client sử dụng cache để giảm tải cho server. Mã **404 (Not Found)** có tỷ lệ thấp, cho thấy cấu trúc website được duy trì tốt và các đường dẫn không bị hỏng nhiều. Mã **500 (Server Error)** rất hiếm, cho thấy hệ thống có độ tin cậy cao và ít gặp sự cố nghiêm trọng.

Tỷ lệ lỗi được tính toán bằng công thức sau để đánh giá chất lượng hệ thống:

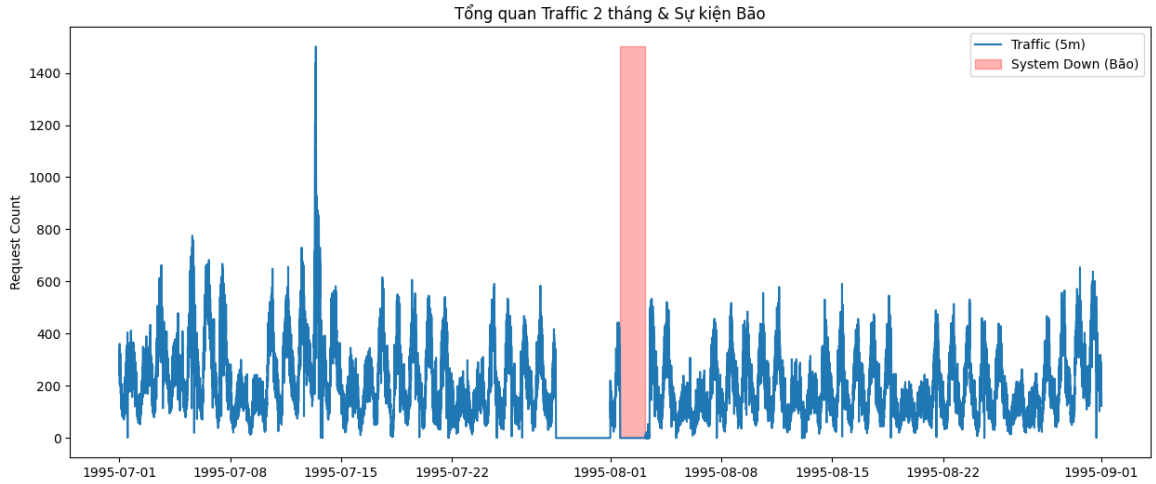
$$\text{error\_rate} = \frac{\text{errors}}{\text{requests}} \tag{2.1}$$

Trong đó `errors` là số lượng request có status code ≥ 400. Tỷ lệ lỗi trung bình trong toàn bộ bộ dữ liệu rất thấp, thường dưới 1%, cho thấy hệ thống hoạt động ổn định và đáng tin cậy.

2.3.3 Mẫu hình Traffic theo thời gian

Biểu đồ traffic theo thời gian (Hình 2.1) cho thấy các mẫu hình rõ ràng về chu kỳ hàng ngày và hàng tuần. Chu kỳ **hàng ngày** thể hiện rõ nét với traffic tăng mạnh vào các giờ làm việc (9h-17h) và giảm vào ban đêm (0h-6h), phản ánh hành vi người dùng điển hình trong ngày làm việc. Chu kỳ **hàng tuần** cho thấy traffic cao hơn vào các ngày trong tuần (Thứ 2 - Thứ 6) và thấp hơn vào cuối tuần (Thứ 7 - Chủ Nhật), phản ánh sự khác biệt trong hành vi truy cập giữa ngày làm việc và cuối tuần. Các **sự kiện bất thường** được đánh dấu bằng vùng màu

đỏ thể hiện các khoảng thời gian system downtime do bão, trong đó traffic giảm đáng kể do dữ liệu giả lập.

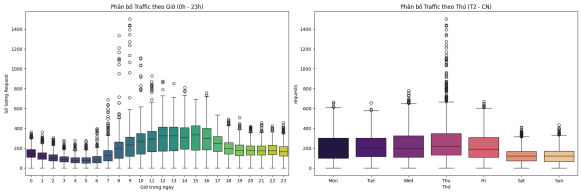


Hình 2.1: Tổng quan Traffic 2 tháng và Sự kiện Bão

2.3.4 Phân tích theo Giờ và Thứ

Biểu đồ phân bố traffic theo giờ trong ngày (Hình 2.2) cho thấy các đặc điểm sau. **Giờ cao điểm** từ 10h-16h có lượng request cao nhất, phản ánh thời gian làm việc chính thức. **Giờ thấp điểm** từ 0h-6h có lượng request thấp nhất, phản ánh thời gian nghỉ ngơi của người dùng. Độ biến động của traffic ở các giờ cao điểm lớn hơn, phản ánh sự không ổn định của traffic trong giờ làm việc.

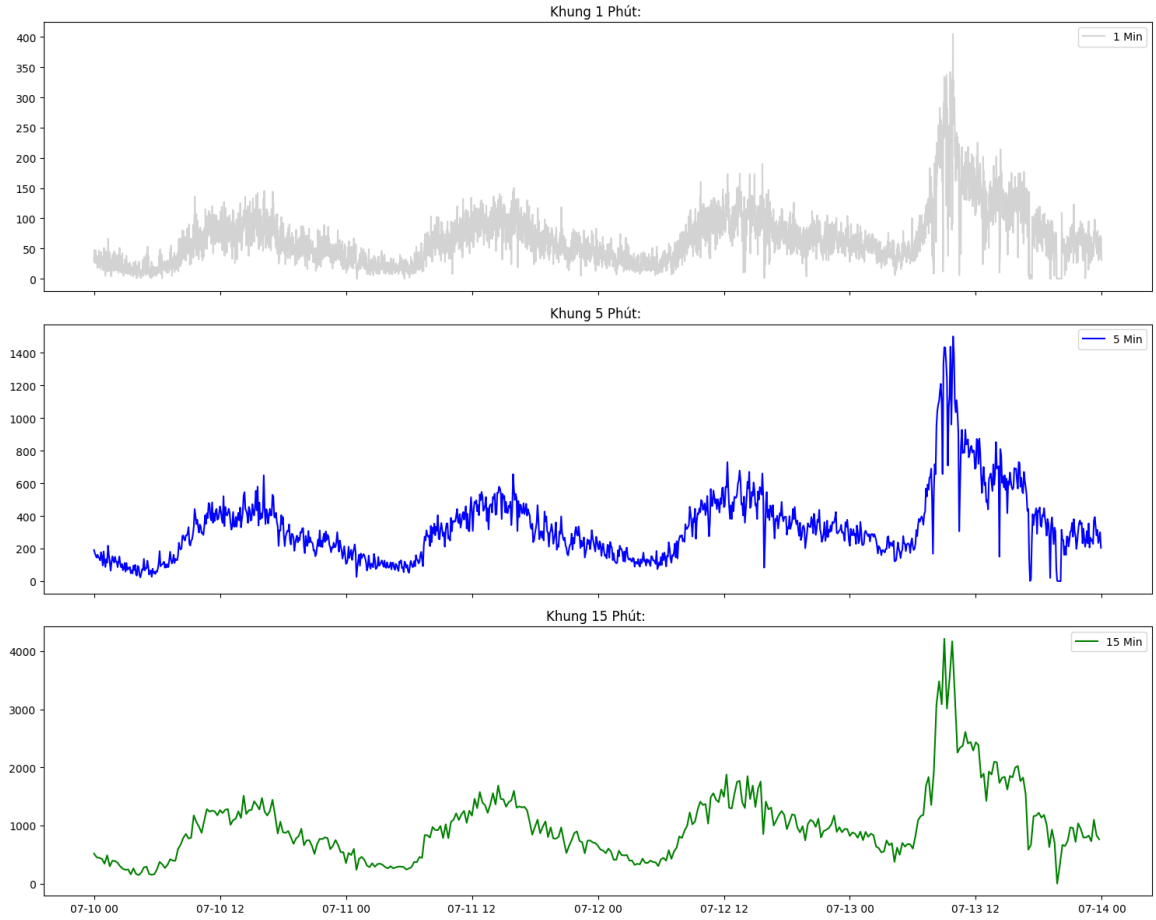
Biểu đồ phân bố traffic theo thứ trong tuần cho thấy các đặc điểm sau. **Ngày trong tuần** như Thứ 3, Thứ 4, Thứ 5 có traffic cao nhất, có thể do đây là những ngày làm việc bận rộn nhất trong tuần. **Cuối tuần** như Thứ 7 và Chủ Nhật có traffic thấp hơn đáng kể, phản ánh hành vi người dùng ít truy cập vào ngày nghỉ ngơi.



Hình 2.2: Phân bố Traffic theo Giờ và Thứ trong tuần

2.3.5 So sánh 3 khung thời gian

Biểu đồ so sánh traffic giữa 3 khung thời gian (Hình 2.3) cho thấy các đặc điểm khác nhau của mỗi cửa sổ. **Khung 1m** cho thấy chi tiết cao nhất nhưng có nhiều nhiễu, phù hợp cho việc phản ứng nhanh với các biến động tức thời nhưng khó dự báo do nhiễu lớn. **Khung 5m** cân bằng tốt giữa chi tiết và độ ổn định, giảm nhiễu đáng kể so với khung 1m nhưng vẫn giữ được các mẫu hình quan trọng. **Khung 15m** mượt mà hơn nhưng mất đi một số chi tiết quan trọng, phù hợp cho các chiến lược dài hạn và giảm nhiễu tối đa.



Hình 2.3: So sánh Traffic giữa 3 khung thời gian

2.4 FEATURE ENGINEERING

2.4.1 Time-based Features

Để nắm bắt các mẫu hình theo thời gian, chúng tôi tạo ra các đặc trưng dựa trên thời gian. **hour_of_day** là giờ trong ngày (0-23) phản ánh chu kỳ hàng ngày của traffic. **day_of_week** là thứ trong tuần (0-6, với 0 là Thứ 2) phản ánh chu kỳ hàng tuần của traffic. **is_weekend** là biến nhị phân (0/1) chỉ ra ngày cuối tuần, giúp phân biệt hành vi người dùng giữa ngày làm việc và cuối tuần.

2.4.2 Cyclical Encoding

Để mô hình hiểu được tính chu kỳ của thời gian (ví dụ: 23h và 0h gần nhau hơn 23h và 12h), chúng tôi sử dụng *Cyclical Encoding* với các hàm sin và cos. Cách tiếp cận này giúp mô hình học được các mẫu hình chu kỳ mà không bị nhầm lẫn bởi sự gián đoạn tại các điểm biên. Các công thức được sử dụng như sau:

$$\text{hour\_sin} = \sin\left(\frac{2\pi \times \text{hour\_of\_day}}{24}\right) \tag{2.2}$$

$$\text{hour\_cos} = \cos\left(\frac{2\pi \times \text{hour\_of\_day}}{24}\right) \tag{2.3}$$

2.4.3 Lag Features

Để nắm bắt tính autoregressive của chuỗi thời gian, chúng tôi tạo ra các đặc trưng lag. **req_lag_1** là giá trị request tại thời điểm $t - 1$, giúp capture các mẫu hình ngắn hạn và phản ứng nhanh với các biến động tức thời. **req_lag_12** là giá trị request tại thời điểm $t - 12$ (1 giờ trước cho khung 5m), giúp capture chu kỳ hàng ngày của traffic. **req_lag_288** là giá trị request tại thời điểm $t - 288$ (24 giờ trước cho khung 5m), giúp capture chu kỳ hàng tuần của traffic. Các đặc trưng lag này đặc biệt quan trọng cho các mô hình như ARIMA và LSTM, giúp mô hình học được các mẫu hình phụ thuộc vào quá khứ.

2.4.4 Rolling Statistics

Để nắm bắt xu hướng và độ biến động của traffic, chúng tôi tính toán các thống kê rolling. **rolling_mean_1h** là trung bình động trong 1 giờ (12 bước cho khung 5m), phản ánh xu hướng ngắn hạn của traffic. **rolling_std_1h** là độ lệch chuẩn động trong 1 giờ, phản ánh độ biến động ngắn hạn của traffic. **rolling_mean_24h** là trung bình động trong 24 giờ (288 bước cho khung 5m), phản ánh xu hướng dài hạn của traffic. Các đặc trưng này giúp mô hình nhận diện các giai đoạn tăng trưởng, giảm sút và ổn định của traffic.

2.5 AGGREGATION VÀ TRAIN/TEST SPLIT

2.5.1 Train/Test Split theo trình tự thời gian

Đối với bài toán chuỗi thời gian, việc split dữ liệu phải tuân thủ nguyên tắc *chronological split* - không được random shuffle vì sẽ làm mất tính liên tục của thời gian. Chúng tôi thực hiện split như sau: **Training set** từ 01/07/1995 đến 22/08/1995 (khoảng 52 ngày), chiếm khoảng 85% dữ liệu. **Test set** từ 23/08/1995 đến 31/08/1995 (khoảng 9 ngày), chiếm khoảng 15% dữ liệu. Tỷ lệ train/test là khoảng 85%/15%, phù hợp cho việc đánh giá khả năng dự báo của mô hình.

Việc split theo trình tự thời gian có những ưu điểm quan trọng. Đầu tiên, **Giả lập thực tế**: Mô hình được huấn luyện trên quá khứ và dự báo cho tương lai, giả lập đúng cách mà các hệ thống dự báo hoạt động trong thực tế. Thứ hai, **Tránh data leakage**: Không có thông tin từ tương lai "rò rỉ"vào tập training, đảm bảo tính công bằng của đánh giá mô hình. Tuy nhiên, việc split theo trình tự thời gian cũng có nhược điểm. Nếu mẫu hình traffic thay đổi đáng kể giữa train và test (concept drift), mô hình có thể hoạt động kém. Ngoài ra, nếu test set chứa các sự kiện đặc biệt không có trong train set, dự báo sẽ kém chính xác.

2.5.2 Lưu trữ dữ liệu

Sau khi xử lý hoàn tất, 3 bộ dữ liệu được lưu trữ trong thư mục `data/cleaned/` để sử dụng cho việc huấn luyện và đánh giá các mô hình. `data_1m.csv` là dữ liệu aggregation theo 1 phút với 89,280 điểm dữ liệu. `data_5m.csv` là dữ liệu aggregation theo 5 phút với 17,856 điểm dữ liệu. `data_15m.csv` là dữ liệu aggregation theo 15 phút với 5,952 điểm dữ liệu. Các file này được sử dụng trực tiếp cho việc huấn luyện và đánh giá các mô hình trong các chương tiếp theo.

Chương 3

MÔ HÌNH ARIMA

3.1 TỔNG QUAN MÔ HÌNH ARIMA

Mô hình ARIMA (AutoRegressive Integrated Moving Average) là một phương pháp thống kê kinh điển trong dự báo chuỗi thời gian, kết hợp ba thành phần chính để nắm bắt và dự báo các mẫu hình trong dữ liệu. Ba thành phần này bao gồm: **AR (AutoRegressive)** sử dụng các giá trị quá khứ để dự báo giá trị tương lai; **I (Integrated)** áp dụng phép sai phân (differencing) để biến đổi chuỗi thời gian thành chuỗi dừng (stationary); và **MA (Moving Average)** sử dụng các sai số dự báo quá khứ để cải thiện độ chính xác của dự báo.

Mô hình ARIMA được ký hiệu là `ARIMA(p,d,q)`, trong đó: `p` là bậc của thành phần tự hồi quy (số lượng giá trị quá khứ được sử dụng), `d` là số lần áp dụng sai phân để đạt được tính dừng, và `q` là bậc của thành phần trung bình động (số lượng sai số dự báo quá khứ được sử dụng). Công thức toán học của mô hình ARIMA có thể được biểu diễn như sau:

$$y_t = c + \sum_{i=1}^{p} \phi_i y_{t-i} + \sum_{j=1}^{q} \theta_j \varepsilon_{t-j} \tag{3.1}$$

Trong đó: `y_t` là giá trị tại thời điểm `t`, `c` là hằng số, ϕ_i là các hệ số tự hồi quy, θ_j là các hệ số trung bình động, và ε_t là sai số trắng (white noise error).

Mô hình ARIMA đặc biệt phù hợp cho bài toán dự báo lưu lượng truy cập web vì: (1) dữ liệu chuỗi thời gian đơn biến (số lượng request) có tính dừng hoặc có thể biến đổi thành dừng thông qua sai phân; (2) có cấu trúc tự tương quan (autocorrelation) rõ rệt trong dữ liệu traffic; (3) các tham số có thể giải thích được, giúp hiểu rõ các yếu tố ảnh hưởng đến lưu lượng; và (4) phù hợp làm baseline trước khi thử nghiệm các mô hình phức tạp hơn như Prophet, LSTM, hay XGBoost.

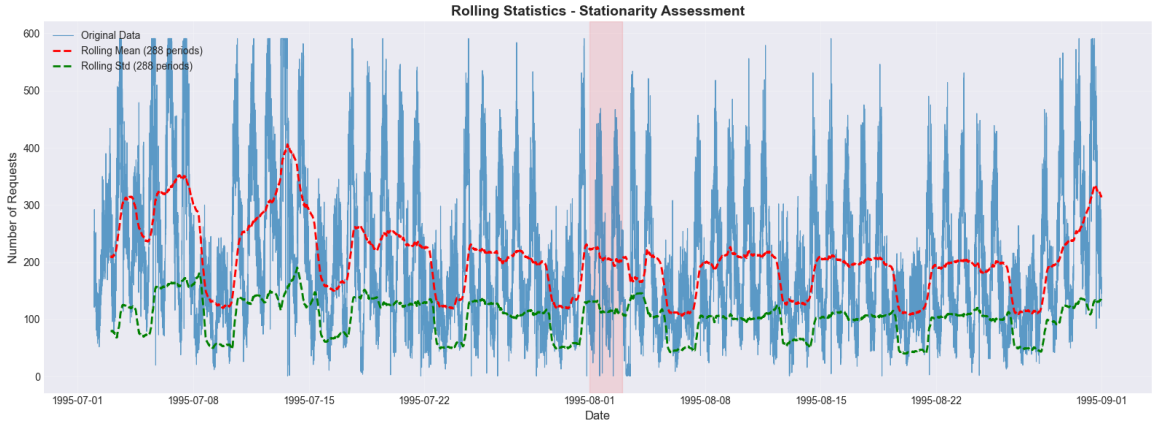
3.2 PHƯƠNG PHÁP HUẤN LUYỆN

Trước khi huấn luyện mô hình ARIMA, chúng tôi thực hiện các bước chuẩn bị dữ liệu và phân tích đặc điểm để đảm bảo dữ liệu đáp ứng các giả định của mô hình.

3.2.1 Đánh giá tính dừng (Stationarity Assessment)

Mô hình ARIMA yêu cầu chuỗi thời gian phải có tính dừng, nghĩa là: giá trị trung bình không thay đổi theo thời gian; phương sai không thay đổi theo thời gian; và tự tương quan giữa các giá trị tại các khoảng thời gian khác không thay đổi theo thời gian. Để đánh giá tính dừng, chúng tôi sử dụng hai phương pháp: kiểm tra trực quan bằng cách vẽ các thống kê rolling (rolling mean và rolling standard deviation) và kiểm tra thống kê Augmented Dickey-Fuller (ADF test).

Hình 3.1 cho thấy thống kê rolling của chuỗi thời gian số lượng request theo cửa sổ 5 phút. Rolling mean (đường nét đứt màu đỏ) và rolling standard deviation (đường nét đứt màu xanh) tương đối ổn định trong suốt giai đoạn huấn luyện, cho thấy chuỗi có tính dừng. Kết quả kiểm tra ADF cho thấy `p-value = 0.000000`, nhỏ hơn nhiều so với mức ý nghĩa 0.05, cho phép chúng tôi bác bỏ giả thuyết về unit root và kết luận chuỗi là dừng. Do đó, tham số `d = 0` (không cần áp dụng sai phân).

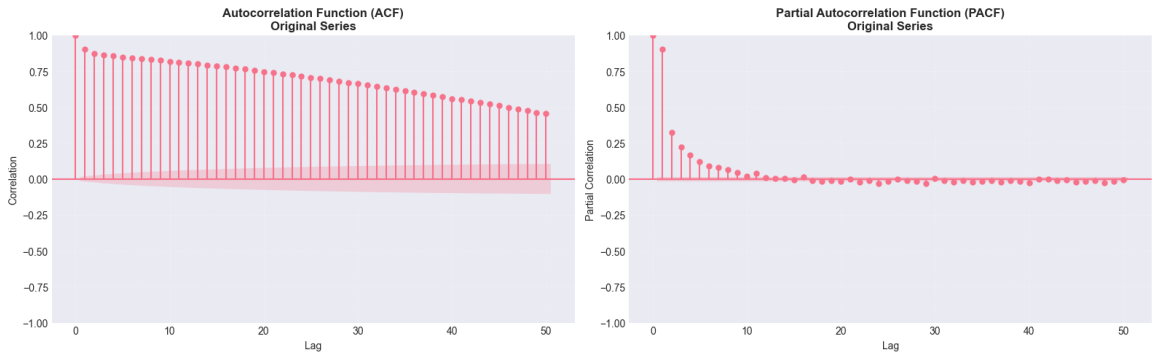


Hình 3.1: Rolling Statistics - Đánh giá tính dừng của chuỗi thời gian

3.2.2 Phân tích tự tương quan (ACF và PACF)

Sau khi xác định chuỗi là dừng, chúng tôi phân tích cấu trúc tự tương quan để xác định các tham số p và q cho mô hình ARIMA. **ACF (Autocorrelation Function)** đo lường tương quan giữa chuỗi thời gian và các giá trị lag của nó, trong khi **PACF (Partial Autocorrelation Function)** đo lường tương quan loại bỏ ảnh hưởng của các lag trung gian.

Hình 3.2 cho thấy đồ thị ACF và PACF của chuỗi thời gian. Đồ thị ACF (bên trái) cho thấy tương quan giảm rất chậm theo thời gian, biểu hiện tính persistence cao của chuỗi. Đồ thị PACF (bên phải) cho thấy có đỉnh rõ rệt tại lag 1 (tương quan rất mạnh) và đỉnh nhỏ hơn tại lag 2, sau đó các giá trị giảm vào vùng tin cậy (confidence interval). Dựa trên các mẫu hình này, chúng tôi đề xuất mô hình ban đầu là `ARIMA(1,0,0)` hoặc `ARIMA(2,0,0)`.



Hình 3.2: Đồ thị ACF và PACF cho phân tích tham số mô hình

3.2.3 Lựa chọn tham số tự động (Auto-ARIMA)

Thay vì chọn tham số `(p,d,q)` thủ công dựa trên phân tích ACF/PACF, chúng tôi sử dụng thuật toán `auto_arima` từ thư viện `pmdarima` để tìm kiếm tự động các tham số tối ưu. Thuật toán này tìm kiếm qua lưới các tổ hợp tham số và chọn mô hình có tiêu chuẩn thông tin Akaike (AIC) thấp nhất, cân bằng giữa độ phù hợp và độ phức tạp của mô hình.

Chúng tôi huấn luyện mô hình ARIMA trên cả ba cửa sổ thời gian khác nhau (1m, 5m, 15m) để so sánh hiệu quả dự báo theo các mức độ phân giải dữ liệu khác nhau. Kết quả lựa chọn tham số cho từng cửa sổ được tóm tắt trong Bảng 3.1: `ARIMA(5,0,1)` cho cửa sổ 1m với `AIC = 605677.82`; `ARIMA(2,0,1)` cho cửa sổ 5m với `AIC = 159089.09`; và `ARIMA(3,0,2)` cho cửa sổ 15m với `AIC = 62299.72`.

Bảng 3.1: Tham số ARIMA tối ưu cho ba cửa sổ thời gian

Cửa sổ	ARIMA(p,d,q)	AIC	BIC	Số lượng huấn luyện
1m	(5,0,1)	605677.82	605751.48	74,880
5m	(2,0,1)	159089.09	159127.16	14,976
15m	(3,0,2)	62299.72	62341.01	4,992

3.2.4 So sánh thủ công và tự động

Để đánh giá hiệu quả của việc lựa chọn tham số thủ công so với tự động, chúng tôi so sánh ba mô hình trên dữ liệu cửa sổ 5m: `ARIMA(1,0,0)` (thủ công dựa trên PACF), `ARIMA(2,0,0)` (thủ công dựa trên PACF), và `ARIMA(2,0,1)` (tự động). Kết quả cho thấy mô hình tự động có `AIC` thấp nhất (159089.09), thấp hơn đáng kể so với hai mô hình thủ công (162404.80 và 160806.62). Điều này cho thấy thuật toán tự động tìm kiếm được các tổ hợp tham số tốt hơn so với phân tích trực quan, đặc biệt là khả năng kết hợp cả thành phần AR và MA.

3.3 KẾT QUẢ CHO CỬA SỔ 5 PHÚT

Sau khi huấn luyện xong mô hình ARIMA cho cửa sổ 5 phút, chúng tôi đánh giá chi tiết hiệu quả dự báo trên tập kiểm tra (test set) từ 23/8/1995 đến 31/8/1995. Cửa sổ 5 phút được chọn để phân tích sâu vì nó đạt được sự cân bằng tốt nhất giữa độ chi tiết và độ ổn định trong ba cửa sổ thời gian được đánh giá.

3.3.1 Cấu trúc mô hình

Mô hình `ARIMA(2,0,1)` được huấn luyện trên 14,976 quan sát dữ liệu 5 phút. Các hệ số tự hồi quy là $\phi_1 = 1.228, \phi_2 = -0.234$ và hệ số trung bình động là $\theta_1 = -0.771$. Cấu trúc mô hình đơn giản hơn so với cửa sổ 1 phút, cho thấy dữ liệu 5 phút đã được làm mượt mà hơn nhờ quá trình aggregation.

Tất cả các tham số đều có ý nghĩa thống kê (p-value < 0.001), cho thấy mô hình phù hợp tốt với dữ liệu huấn luyện. Tham số $\phi_1 = 1.228$ cho thấy giá trị hiện tại phụ thuộc mạnh vào giá trị ngay trước đó, trong khi tham số $\phi_2 = -0.234$ cho thấy sự điều chỉnh từ giá trị hai bước trước đó. Tham số $\theta_1 = -0.771$ cho thấy mô hình sử dụng sai số dự báo quá khứ để điều chỉnh dự báo hiện tại.

3.3.2 Hiệu quả dự báo

Khi dự báo trên tập kiểm tra (2,592 quan sát), mô hình đạt được các chỉ số hiệu quả sau: `RMSE = 121.86`, `MAE = 99.20`, và `MAPE = 86.26%`. Cửa sổ 5 phút đạt được sự cân bằng tốt giữa độ chi tiết và độ ổn định: `RMSE` và `MAE` thấp hơn so với baseline (mean baseline), nhưng `MAPE` vẫn cao do ảnh hưởng của các giá trị nhỏ vào ban đêm.

Bảng 3.2 tóm tắt hiệu quả dự báo của mô hình ARIMA cho cửa sổ 5 phút.

Bảng 3.2: Hiệu quả dự báo ARIMA cho cửa sổ 5 phút

Cửa sổ	MSE	RMSE	MAE	MAPE (%)
5m	14849.92	121.86	99.20	86.26

3.4 MODEL DIAGNOSTICS

Sau khi huấn luyện xong mô hình, chúng tôi thực hiện các kiểm tra chẩn đoán để xác nhận mô hình có đáp ứng các giả định của ARIMA.

3.4.1 Kiểm tra Ljung-Box

Kiểm tra Ljung-Box được sử dụng để đánh giá xem các phần dư (residuals) của mô hình có tự tương quan hay không. Giả thuyết null là các phần dư không có tự tương quan (white noise). Nếu `p-value > 0.05`, chúng tôi không thể bác bỏ giả thuyết null, cho thấy các phần dư là white noise (tốt). Nếu `p-value` $\leq$ `0.05`, các phần dư có tự tương quan, cho thấy mô hình chưa nắm bắt được hết các mẫu hình trong dữ liệu.

Kết quả kiểm tra Ljung-Box cho cửa sổ 5 phút có `p-value = 0.082677` (> 0.05), cho thấy các phần dư là white noise (tốt). Điều này cho thấy mô hình đã nắm bắt được hầu hết các mẫu hình trong dữ liệu.

3.4.2 Kiểm tra Jarque-Bera

Kiểm tra Jarque-Bera được sử dụng để đánh giá xem các phần dư có phân phối chuẩn hay không. Giả thuyết null là các phần dư có phân phối chuẩn. Nếu `p-value > 0.05`, chúng tôi không thể bác bỏ giả thuyết null, cho thấy các phần dư có phân phối chuẩn (tốt). Nếu `p-value` $\leq$ `0.05`, các phần dư không có phân phối chuẩn.

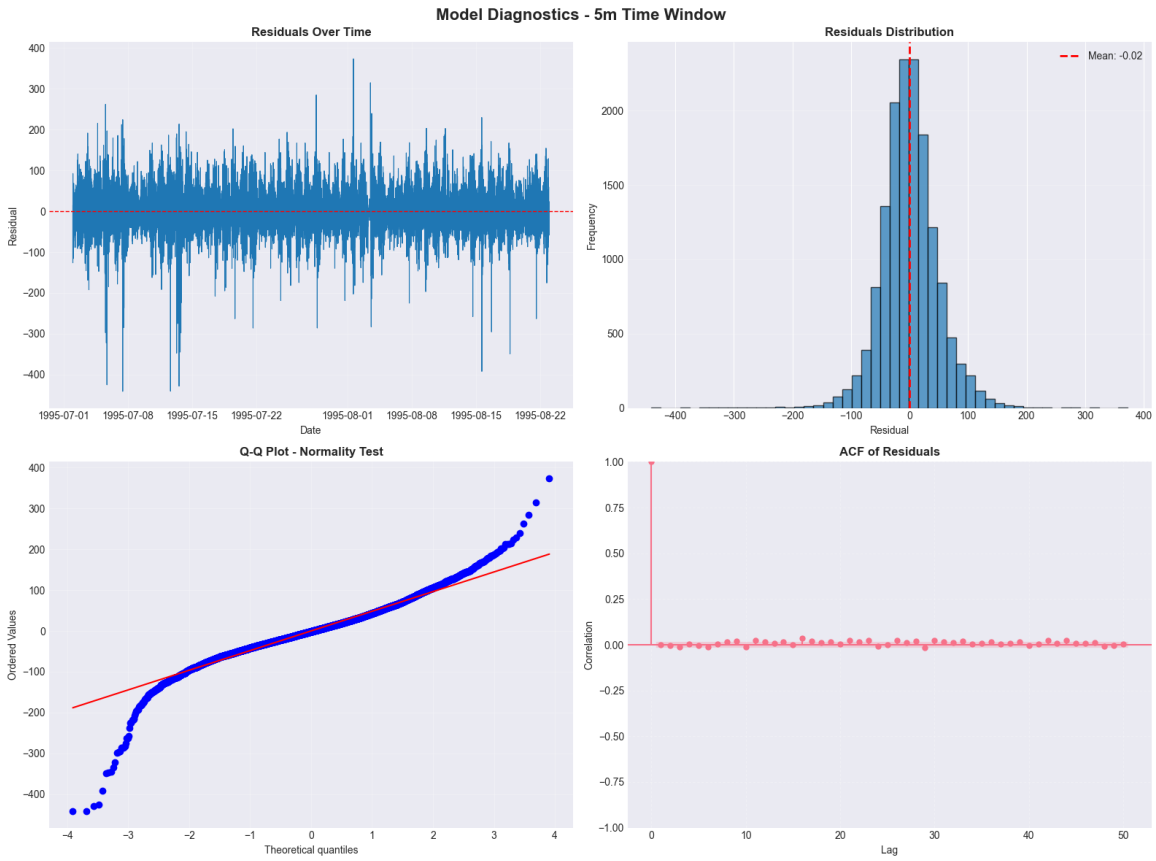
Kết quả kiểm tra Jarque-Bera cho cửa sổ 5 phút có `p-value = 0.000000`, nhỏ hơn nhiều so với mức ý nghĩa 0.05, cho thấy các phần dư không có phân phối chuẩn. Điều này cho thấy dữ liệu traffic có phân phối lệch (skewed) với nhiều giá trị outlier, đặc biệt vào giờ cao điểm.

3.4.3 Kiểm tra Durbin-Watson

Kiểm tra Durbin-Watson được sử dụng để đánh giá xem có tự tương quan bậc nhất trong các phần dư hay không. Thống kê Durbin-Watson có giá trị từ 0 đến 4, trong đó giá trị 2 cho thấy không có tự tương quan bậc nhất. Kết quả kiểm tra cho cửa sổ 5 phút có `DW statistic = 1.9967`, gần 2.0, cho thấy không có tự tương quan bậc nhất đáng kể trong các phần dư.

3.4.4 Đồ thị chẩn đoán mô hình

Hình 3.3 cho thấy đồ thị chẩn đoán mô hình cho cửa sổ 5 phút. Đồ thị bao gồm: (1) Residuals over time (phần trên trái) cho thấy các phần dư dao động quanh giá trị 0; (2) Residuals distribution (phần trên phải) cho thấy phân phối gần chuẩn với mean gần 0; (3) Q-Q plot (phần dưới trái) cho thấy các điểm gần như nằm trên đường chéo, cho thấy phân phối gần chuẩn; và (4) ACF of residuals (phần dưới phải) cho thấy không có cột nào vượt quá vùng tin cậy, cho thấy không có tự tương quan đáng kể trong các phần dư.



Hình 3.3: Chẩn đoán mô hình ARIMA cho cửa sổ 5 phút

3.5 PHÂN TÍCH VÀ ĐÁNH GIÁ

3.5.1 So sánh với baseline

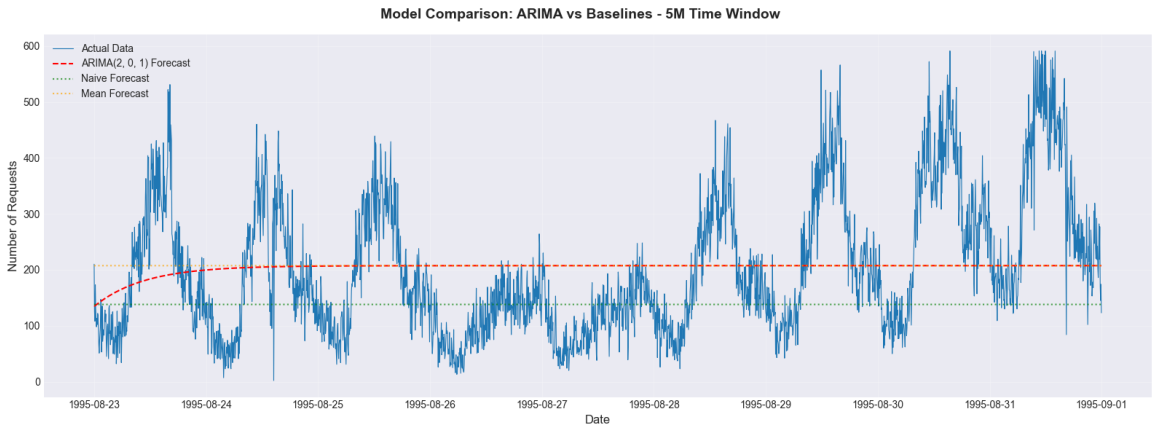
Để đánh giá giá trị thực tế của mô hình ARIMA, chúng tôi so sánh với hai mô hình baseline đơn giản: **Naive baseline** (dự báo bằng giá trị cuối cùng của tập huấn luyện) và **Mean baseline** (dự báo bằng giá trị trung bình của tập huấn luyện). Kết quả so sánh cho cửa sổ 5 phút được tóm tắt trong Bảng 3.3.

Bảng 3.3: So sánh mô hình ARIMA với các baseline cho cửa sổ 5 phút

Cửa sổ	Naive RMSE	Mean RMSE	ARIMA RMSE	Cải thiện (%)
5m	138.74	122.50	121.86	+0.5

Kết quả cho thấy mô hình ARIMA chỉ cải thiện nhẹ 0.5% so với baseline. Điều này cho thấy mô hình ARIMA gần như chỉ dự báo bằng giá trị trung bình, cho thấy giá trị gia tăng của mô hình là rất hạn chế.

Hình 3.4 cho thấy so sánh trực quan giữa dự báo của mô hình ARIMA (đường nét đứt màu đỏ), dự báo naive (đường nét đứt màu xanh lá), và dự báo mean (đường nét đứt màu cam). Mô hình ARIMA gần như trùng với dự báo mean, cho thấy giá trị gia tăng của mô hình là rất hạn chế.



Hình 3.4: So sánh mô hình ARIMA với các baseline cho cửa sổ 5 phút

3.5.2 Phân tích overfitting

Để đánh giá mức độ overfitting của mô hình, chúng tôi so sánh hiệu quả trên tập huấn luyện (in-sample) và tập kiểm tra (out-of-sample). Kết quả cho thấy mô hình có hiện tượng overfitting rõ rệt: hiệu quả trên tập kiểm tra kém hơn đáng kể so với tập huấn luyện.

Bảng 3.4: So sánh hiệu quả in-sample và out-of-sample cho cửa sổ 5 phút

Cửa sổ	In-sample RMSE	Out-of-sample RMSE	Tỷ lệ overfitting
5m	49.02	121.86	2.48x

Tỷ lệ overfitting được tính bằng tỷ lệ giữa `RMSE out-of-sample` và `RMSE in-sample`. Cửa sổ 5 phút có tỷ lệ 2.48x, cao hơn 2, cho thấy mô hình ARIMA overfitting nghiêm trọng, với hiệu quả trên tập kiểm tra kém hơn 2.5 lần so với tập huấn luyện.