|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Tiêu chí** | **AirQuality (Thô)** | **Final (Đã xử lý)** | **Giải thích sự khác biệt** | **Code xử lý** | | **Giá trị lỗi** | Chứa các giá trị -200 ở nhiều cột như CO(GT), NOx(GT), PT08.S1(CO), v.v., biểu thị lỗi cảm biến hoặc dữ liệu không ghi nhận được. | -200 được thay bằng NaN, sau đó điền đầy bằng nội suy tuyến tính, ví dụ: -200 trong CO(GT) thành một số hợp lý như 2.5. | File thô có giá trị lỗi (-200) không phản ánh nồng độ thật. File final loại bỏ lỗi này, điền giá trị hợp lý để dữ liệu đầy đủ và chính xác hơn. | df.replace(-200, np.nan, inplace=True) <br> df.interpolate(method='linear', inplace=True) | | **Định dạng số** | Các cột số dạng chuỗi, dùng dấu phẩy làm thập phân (e.g., 2,6), do chuẩn định dạng file CSV châu Âu. | Các cột số được chuyển thành kiểu số thực, dùng dấu chấm (e.g., 2.6), phù hợp với Python. | File thô không dùng được trực tiếp vì Python không hiểu chuỗi 2,6. File final chuẩn hóa thành 2.6 để tính toán và học máy. | numeric\_cols = df.select\_dtypes(include=['object']).columns <br> for col in numeric\_cols: if col not in ['Date', 'Time']: df[col] = pd.to\_numeric(df[col], errors='coerce') | | **Kích thước** | (9471, 17) – 9471 dòng, 17 cột (bao gồm Date, Time, các cột cảm biến, và có thể có cột thừa như Unnamed). | (9356, 19) hoặc tương tự – giảm dòng do lọc dữ liệu lỗi, tăng cột do thêm đặc trưng mới. | File thô chứa toàn bộ dữ liệu gốc, có thể có dòng lỗi. File final giảm dòng (do lọc NaN còn sót hoặc outlier), thêm cột để tăng thông tin. | Không trực tiếp trong code chính, nhưng thấy trong dự đoán mới và quá trình xử lý ngầm (e.g., X có 19 cột khi thêm đặc trưng). | | **Dữ liệu thiếu** | Có ô trống hoặc giá trị lỗi (-200) ở nhiều cột, đặc biệt khi cảm biến không hoạt động. | Tất cả ô trống (NaN) được điền đầy bằng nội suy tuyến tính, không còn giá trị thiếu. | File thô thiếu dữ liệu làm mô hình không học được. File final điền đầy để tận dụng hết dữ liệu, phù hợp với dữ liệu thời gian liên tục. | df.interpolate(method='linear', inplace=True) | | **Cột thời gian** | Có 2 cột riêng: Date (e.g., 10/03/2004) và Time (e.g., 18:00:00), đều là chuỗi. | Gộp thành một cột Timestamp (e.g., 10/03/2004 18:00:00), thêm cột DayOfWeek (thứ), Month (tháng). | File thô khó xử lý thời gian vì tách riêng. File final gộp lại và thêm cột để tận dụng xu hướng thời gian (e.g., CO cao hơn vào giờ cao điểm). | Chỉ thấy trong phần dự đoán mới: <br> new\_df['DayOfWeek'] = pd.to\_datetime('15/03/2025 18:00:00').dayofweek <br> new\_df['Month'] = pd.to\_datetime('15/03/2025 18:00:00').month | | **Đặc trưng mới** | Chỉ có 17 cột gốc: các cảm biến (PT08.S1(CO), NOx(GT), v.v.), Date, Time, T, RH, AH. | Thêm các cột như NOx\_NO2\_Ratio (tỷ lệ NOx/NO2), CO\_Lag1 (CO của thời điểm trước), DayOfWeek, Month. | File thô thiếu thông tin bổ sung. File final thêm đặc trưng để mô hình hiểu rõ hơn mối quan hệ giữa các biến và xu hướng thời gian. | new\_df['NOx\_NO2\_Ratio'] = new\_df['NOx(GT)'] / (new\_df['NO2(GT)'] + 1e-5) <br> new\_df['CO\_Lag1'] = df['CO(GT)'].iloc[-1] (trong dự đoán mới) | | **Chuẩn hóa** | Giá trị thô, thang đo khác nhau (e.g., PT08.S1(CO) ~1000-2000, CO(GT) ~0-10). | Được chuẩn hóa về trung bình 0, độ lệch chuẩn 1 (e.g., PT08.S1(CO) và CO(GT) cùng thang 0-1). | File thô có thang đo lệch, ảnh hưởng mô hình. File final chuẩn hóa để các cột đồng đều, tránh biến lớn lấn át biến nhỏ khi học. | scaler = StandardScaler() <br> X\_train\_scaled = scaler.fit\_transform(X\_train) <br> X\_val\_scaled = scaler.transform(X\_val) <br> X\_test\_scaled = scaler.transform(X\_test) | | **Outlier (giá trị bất thường)** | Có thể chứa các giá trị bất thường (e.g., nồng độ cực cao hoặc âm ngoài -200). | Không thấy rõ trong code, nhưng nếu có thì đã lọc bớt qua quá trình làm sạch hoặc nội suy. | File thô có outlier làm mô hình học nhầm. File final giảm outlier để dữ liệu ổn định hơn (dù không rõ trong code chính). | Không trực tiếp trong code, nhưng có thể ngầm qua df.interpolate hoặc lọc dòng khi tạo X, y. | |

**Chi tiết từng sự khác biệt**

1. **Giá trị lỗi**:
   * **Thô**: -200 xuất hiện ở nhiều cột, là lỗi cảm biến, không phải nồng độ thật.
   * **Final**: Đổi thành NaN, rồi điền bằng nội suy (ví dụ: -200 giữa 2 và 4 thành 3).
   * **Code**: df.replace(-200, np.nan, inplace=True) thay -200, df.interpolate(method='linear', inplace=True) điền giá trị.
2. **Định dạng số**:
   * **Thô**: Cột số dạng chuỗi (e.g., 2,6), do chuẩn CSV châu Âu.
   * **Final**: Chuyển thành số thực (e.g., 2.6), đúng chuẩn Python.
   * **Code**: Vòng lặp for col in numeric\_cols với pd.to\_numeric đổi định dạng.
3. **Kích thước**:
   * **Thô**: (9471, 17) – toàn bộ dữ liệu gốc, có thể có cột thừa.
   * **Final**: (9356, 19) – giảm dòng do lọc lỗi, tăng cột do thêm đặc trưng.
   * **Code**: Không trực tiếp, nhưng thấy trong X (19 cột) và quá trình xử lý ngầm.
4. **Dữ liệu thiếu**:
   * **Thô**: Có ô trống hoặc -200, đặc biệt khi cảm biến hỏng.
   * **Final**: Điền đầy bằng nội suy tuyến tính, không còn thiếu.
   * **Code**: df.interpolate(method='linear', inplace=True).
5. **Cột thời gian**:
   * **Thô**: Date và Time tách riêng, dạng chuỗi.
   * **Final**: Gộp thành Timestamp, thêm DayOfWeek, Month (chỉ thấy trong dự đoán mới).
   * **Code**: new\_df['DayOfWeek'] = ..., new\_df['Month'] = ... (trong phần dự đoán).
6. **Đặc trưng mới**:
   * **Thô**: Chỉ có cột gốc từ cảm biến và môi trường.
   * **Final**: Thêm NOx\_NO2\_Ratio, CO\_Lag1, v.v., để tăng thông tin.
   * **Code**: new\_df['NOx\_NO2\_Ratio'] = ..., new\_df['CO\_Lag1'] = ... (trong dự đoán mới).
7. **Chuẩn hóa**:
   * **Thô**: Giá trị thô, thang đo lệch nhau (hàng nghìn vs đơn vị).
   * **Final**: Chuẩn hóa về mean=0, std=1, đồng đều.
   * **Code**: scaler.fit\_transform(X\_train), scaler.transform(X\_val/test).
8. **Outlier**:
   * **Thô**: Có thể có giá trị bất thường ngoài -200.
   * **Final**: Giảm outlier qua nội suy hoặc lọc ngầm.
   * **Code**: Không rõ trong code chính, nhưng có thể qua df.interpolate.

**Trả lời giáo viên**

* "File AirQuality thô có lỗi -200, định dạng chuỗi 2,6, thiếu đặc trưng, và chưa chuẩn hóa. Em dùng df.replace để thay -200 bằng NaN, df.interpolate để điền thiếu, pd.to\_numeric để đổi định dạng, thêm cột như NOx\_NO2\_Ratio trong dự đoán mới, và StandardScaler để chuẩn hóa. File final sạch, đầy đủ, có thêm cột, và đồng đều, giúp mô hình stacking dự đoán CO(GT) chính xác hơn."