BÁO CÁO CUỘC THI DATAFLOW 2025

Đội thi minekraft - Product Recommendation

Giới thiệu:

Dự án này được chia thành ba giai đoạn chính, tương ứng với ba tập tin Jupyter Notebook riêng biệt, nhằm mục tiêu phân tích dữ liệu khách hàng, chuẩn bị dữ liệu và xây dựng mô hình dự đoán hành vi sử dụng sản phẩm. Ba giai đoạn bao gồm:

- **Xử Lý và Chuẩn Bị Dữ Liệu (data_cleaning_and_feature_engineering.ipynb)**: Giai đoạn này tập trung vào việc làm sạch, biến đổi và chuẩn bị dữ liệu khách hàng, đảm bảo dữ liệu đạt chất lượng tốt nhất cho quá trình huấn luyện mô hình học máy.
- **Phân Tích Dữ Liệu Khách Hàng (data_ana.ipynb)**: Tập trung vào việc khám phá dữ liệu, hiểu rõ hành vi sử dụng sản phẩm của khách hàng, cũng như xác định các xu hướng và mức độ ảnh hưởng của từng sản phẩm.
- **Huấn Luyện và Dự Đoán Mô Hình (model.ipynb)**: Mục tiêu chính là xây dựng một mô hình học máy có khả năng dự đoán sản phẩm mới mà khách hàng có khả năng sử dụng trong tương lai. Mô hình này sẽ được sử dụng để tạo ra tệp dự đoán cuối cùng cho cuộc thi.

1. Xử Lý và Chuẩn Bị Dữ Liệu (data_cleaning_and_feature_engineering.ipynb)

Mục tiêu:

Làm sạch dữ liệu thô, xử lý các vấn đề về dữ liệu, biến đổi và tạo ra các đặc trưng (features) mới từ dữ liệu khách hàng. Mục tiêu cuối cùng là chuẩn bị một bộ dữ liệu chất lượng cao, sẵn sàng cho việc xây dựng mô hình học máy.

Các công việc đã thực hiện:

1. **Xử lý Giá trị Thiếu (Missing Values):**
    - **Cột "age" (Tuổi):**
        - Tính toán tuổi trung bình theo các nhóm khách hàng khác nhau (ví dụ: theo khu vực, loại sản phẩm sử dụng).
        - Xử lý các giá trị ngoại lai (outliers) như tuổi quá nhỏ hoặc quá lớn so với thực tế.
        - Điền các giá trị thiếu trong cột "age" bằng giá trị tuổi trung bình đã tính toán cho nhóm tương ứng.
        - Chuyển đổi kiểu dữ liệu cột "age" về dạng số nguyên.
    - **Các cột khác:** Điền giá trị thiếu bằng các phương pháp khác nhau tùy theo cột:
        - Sử dụng **giá trị xuất hiện nhiều nhất (mode)** cho các cột phù hợp.
        - Sử dụng **giá trị trung vị (median)** cho các cột số có phân phối lệch.
        - Sử dụng **giá trị cố định** như '2020-01-01' (cho cột ngày tháng) hoặc '-1' (cho các cột số khác) khi thích hợp.
    - **Loại bỏ hàng:** Xóa bỏ các hàng dữ liệu vẫn còn chứa giá trị thiếu sau các bước xử lý trên (trong trường hợp số lượng hàng này không đáng kể).

2. **Chuyển đổi kiểu dữ liệu:**
    - Chuyển đổi kiểu dữ liệu của nhiều cột sang kiểu số nguyên (`IntegerType`) để tối ưu hóa việc lưu trữ và tính toán.
3. **Ánh xạ Giá trị (Label Encoding):**
    - Do model không thể sử dụng dạng biểu diễn chuỗi kí tự(string), do đó ta phải chuyển chúng thành số nguyên.
    - Chuyển đổi các giá trị dạng chuỗi (text) trong nhiều cột (ví dụ: tên tỉnh thành, loại sản phẩm) thành các mã số nguyên duy nhất.
    - Sử dụng các **từ điển LabelEncoding** đã được định nghĩa trước để đảm bảo tính nhất quán trong quá trình mã hóa.
4. **Xử lý cột Ngày tháng:**
    - Tách các cột ngày tháng gốc thành các cột thành phần riêng biệt: năm, tháng và ngày.
    - Xử lý các giá trị `NULL` (giá trị rỗng) trong các cột mới tạo bằng cách gán giá trị `-1` để thể hiện thông tin bị thiếu.
    - Xóa các cột ngày tháng gốc sau khi đã tách thành các cột thành phần.
5. **Tiền xử lý khác:**
    - Thay thế giá trị "P" bằng giá trị `-2` trong cột `indrel_1mes` (theo yêu cầu nghiệp vụ hoặc để đơn giản hóa dữ liệu).
    - Điền giá trị thiếu bằng `-1` và chuyển đổi kiểu dữ liệu thành số nguyên trong một số cột cụ thể (theo yêu cầu nghiệp vụ).
    - Xóa cột `nomprov` (nếu cột này không còn cần thiết cho phân tích hoặc mô hình).
6. **Tạo Đặc trưng Trễ (Lag Features):**
    - Tạo ra các cột mới bằng cách lấy giá trị của các cột sản phẩm ở các tháng trước đó (ví dụ: 1 tháng trước, 2 tháng trước, ..., 6 tháng trước).
    - Các đặc trưng trễ này giúp mô hình dựa vào sản phẩm của thời gian trước đó, để học và dự đoán các sản phẩm trong tháng hiện tại.
7. **Phát hiện Khoảng trống (Gap Detection) trong chuỗi thời gian:**
    - Xác định các khoảng thời gian bị thiếu dữ liệu trong chuỗi thời gian của mỗi khách hàng (ví dụ: khách hàng có dữ liệu tháng 1, 2, 5, 6 nhưng thiếu dữ liệu tháng 3 và 4).
    - Thêm cột `"gap_flag"` để đánh dấu các khoảng trống này: giá trị `1` nếu có khoảng trống trong chuỗi thời gian của khách hàng, giá trị `0` nếu không có.
8. **Lưu trữ dữ liệu đã xử lý:**
    - Lưu DataFrame đã được làm sạch và biến đổi vào file định dạng **Parquet**. Parquet là một định dạng file cột (columnar) tối ưu cho việc lưu trữ và truy vấn dữ liệu lớn, giúp tăng tốc độ đọc ghi dữ liệu trong các bước tiếp theo.
9. **Kiểm tra Giá trị NULL (sau xử lý):**
    - Đếm số lượng giá trị `NULL` còn lại trong DataFrame sau tất cả các bước xử lý.

- Xuất kết quả đếm giá trị `NULL` dưới dạng Dictionary (từ điển Python) để dễ dàng theo dõi và kiểm tra.

Công cụ sử dụng:

- **PySpark**: Thư viện Spark cho Python, tiếp tục được sử dụng cho các công việc xử lý và biến đổi dữ liệu quy mô lớn.

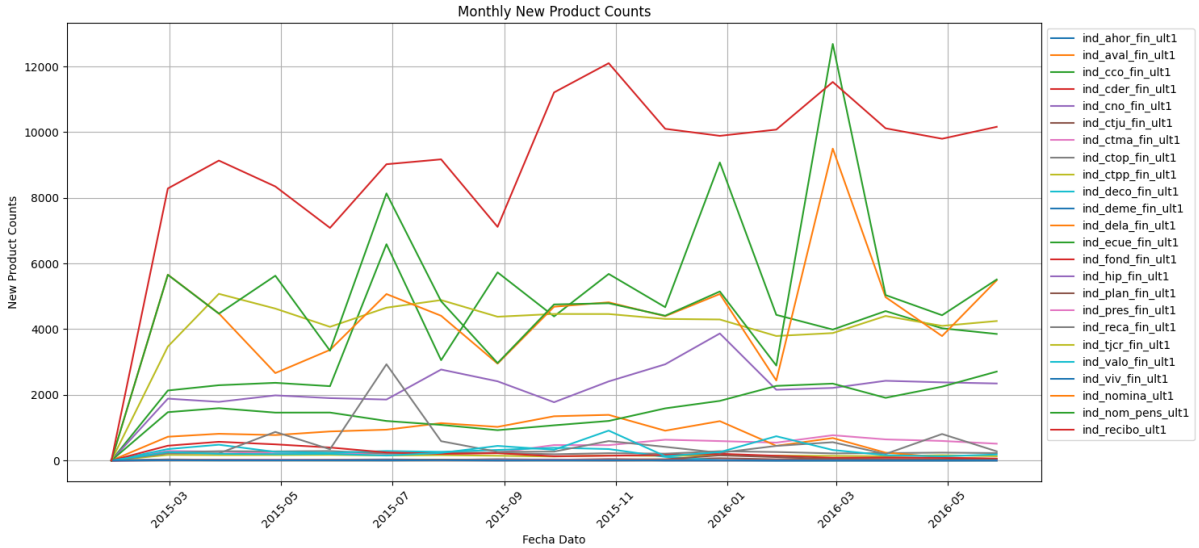
2. Phân Tích Dữ Liệu Khách Hàng (data_ana.ipynb)

Mục tiêu:

Phân tích sâu dữ liệu khách hàng để hiểu rõ hành vi sử dụng sản phẩm, xác định các xu hướng quan trọng và đánh giá mức độ ảnh hưởng của từng sản phẩm đến hành vi khách hàng.

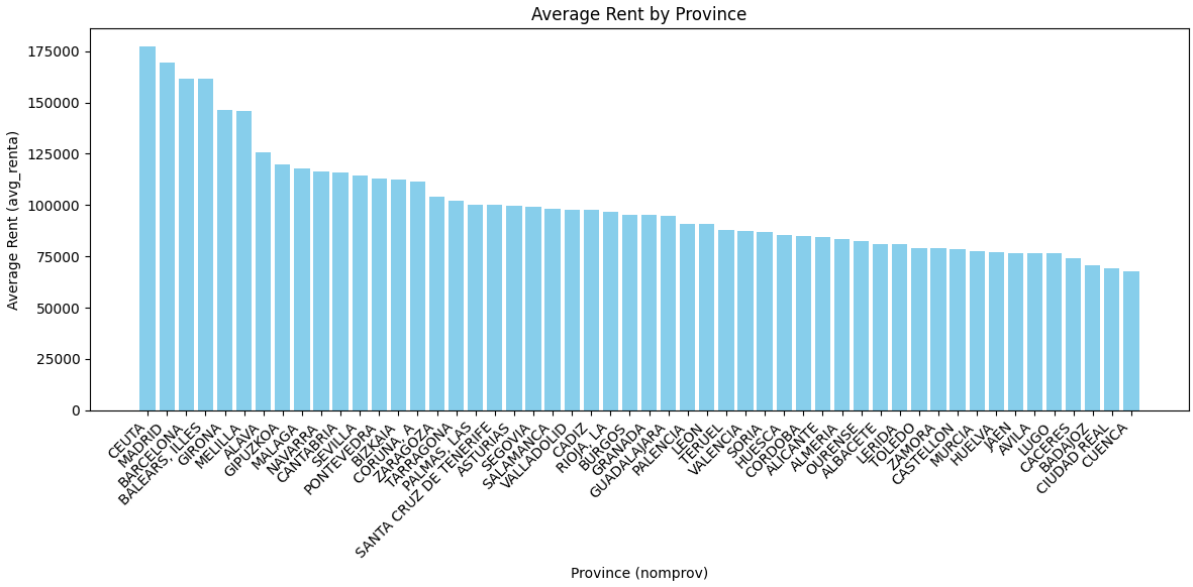
Các công việc đã thực hiện:

1. **Xử lý dữ liệu ban đầu:**
   - Chuyển đổi kiểu dữ liệu của các cột liên quan đến số lượng, đảm bảo chúng ở dạng số nguyên để phục vụ cho các phép tính.
   - Chuẩn hóa định dạng cột ngày tháng, đưa về dạng chuẩn để dễ dàng phân tích theo thời gian.
2. **Phân tích xu hướng sử dụng sản phẩm theo thời gian:**
   - **Thống kê theo tháng:**
     - Đếm số lượng khách hàng *mới* bắt đầu sử dụng từng sản phẩm trong mỗi tháng.
     - Đếm số lượng khách hàng *ngừng* sử dụng từng sản phẩm trong mỗi tháng.
   - **Thống kê tổng quan về hành vi khách hàng:**
     - Tính tổng số lần xuất hiện của mỗi khách hàng trong toàn bộ dữ liệu.
     - Xác định số tháng hoạt động của mỗi khách hàng, đo lường mức độ gắn bó.
   - **Xây dựng Ma trận Khách hàng - Tháng:** Tạo một ma trận để biểu diễn sự hiện diện của từng khách hàng trong mỗi tháng (có mặt/không có mặt).
   - **Xác định Lần Xuất Hiện Đầu Tiên:** Tìm ra tháng đầu tiên mà mỗi khách hàng xuất hiện trong dữ liệu, cho biết thời điểm gia nhập của khách hàng.

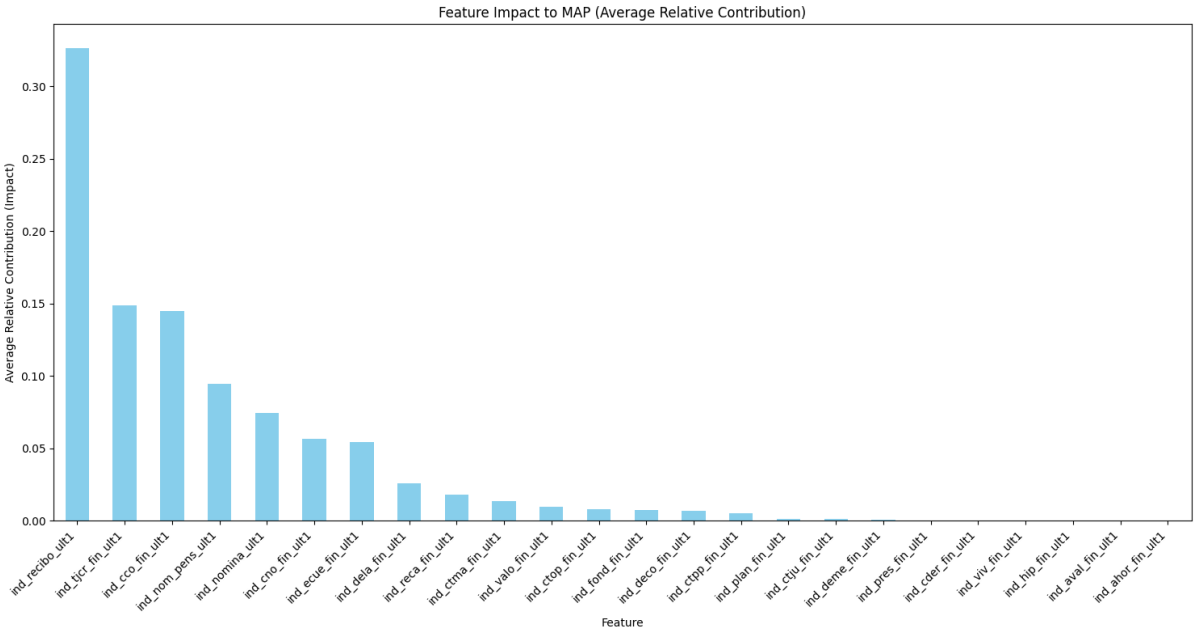


3. **Phân tích sâu về sản phẩm:**

- **Thu nhập trung bình theo tỉnh thành:** Tính toán thu nhập trung bình của khách hàng theo từng tỉnh thành, tìm hiểu sự khác biệt về kinh tế giữa các khu vực.



- **Phân tích sự đón nhận sản phẩm mới:**
  - Tập trung vào phân tích các khách hàng "chưa từng sử dụng sản phẩm" trong quá khứ, nhưng có sử dụng sản phẩm trong tháng hiện tại.
  - Phân tích xu hướng chấp nhận sản phẩm mới theo thời gian, đặc biệt tập trung vào tháng 5 để tìm hiểu các yếu tố đặc biệt.
- **Phân tích chuyên sâu về tháng 5:**
  - Tính toán tỷ lệ tương đối hàng tháng của các chỉ số và làm nổi bật tháng 5 để so sánh.
  - Tính tổng số lượng "đón nhận sản phẩm mới" trong toàn bộ chu kỳ thời gian được xem xét.
  - So sánh tỷ lệ "đón nhận sản phẩm mới" của tháng 5 so với trung bình của 3 tháng trước đó.
  - Phân tích tần suất sử dụng từng sản phẩm cụ thể trong tháng 5.
- **Đánh giá Ảnh hưởng của Tính năng Sản phẩm:** Đánh giá mức độ đóng góp của từng sản phẩm mới (tính năng mới) đến chỉ số MAP (Mean Average Precision), một chỉ số đo lường hiệu quả tổng thể.



4. **Trực quan hóa dữ liệu:**
    - Vẽ các biểu đồ trực quan để biểu diễn các số liệu và kết quả phân tích đã tính toán, giúp dễ dàng nhận diện xu hướng và mô hình.

Công cụ sử dụng:

- **PySpark**: Thư viện Spark cho Python, được sử dụng để xử lý và phân tích dữ liệu lớn một cách hiệu quả.

3. Huấn Luyện và Dự Đoán Mô Hình (model.ipynb)

Mục tiêu:

Xây dựng một mô hình học máy có khả năng dự đoán danh sách các sản phẩm mới mà mỗi khách hàng có khả năng sẽ sử dụng trong tháng tiếp theo. Kết quả dự đoán sẽ được sử dụng để tạo ra tệp submission (nộp bài) cho cuộc thi.

Các công việc đã thực hiện:

1. **Giải phóng bộ nhớ GPU (nếu có):**
    - Đảm bảo giải phóng bộ nhớ GPU (Graphics Processing Unit) trước khi bắt đầu huấn luyện mô hình, đặc biệt nếu máy tính đang sử dụng GPU để tăng tốc quá trình huấn luyện. Điều này đảm bảo không có dữ liệu không cần thiết chiếm dụng bộ nhớ GPU.
2. **Chuẩn bị dữ liệu cho mô hình:**
    - **Đọc dữ liệu đã xử lý:** Đọc dữ liệu đã được làm sạch và biến đổi từ file Parquet đã lưu ở bước trước.
    - **Định nghĩa danh sách sản phẩm và đặc trưng trễ:** Xác định danh sách các sản phẩm cần dự đoán và danh sách các cột đặc trưng trễ (lagged features) đã tạo ra ở giai đoạn trước sẽ được sử dụng làm đầu vào cho mô hình.
    - **Chia dữ liệu thành các tập:** Chia dữ liệu thành ba tập con:
        - **Tập Huấn luyện (Train):** Sử dụng dữ liệu từ các tháng trước tháng 15 để huấn luyện mô hình.
        - **Tập Kiểm định (Validation):** Sử dụng dữ liệu của tháng 15 để đánh giá hiệu năng của mô hình trong quá trình huấn luyện và điều chỉnh siêu tham số.
        - **Tập Kiểm tra (Test):** Sử dụng dữ liệu của tháng 16 (tháng cần dự đoán) để tạo ra dự đoán cuối cùng cho tệp submission.
    - **Chuẩn bị dữ liệu huấn luyện, kiểm định và kiểm tra chi tiết:**
        - **Xác định khách hàng mới sử dụng sản phẩm:** Xác định những khách hàng nào đã bắt đầu sử dụng sản phẩm mới trong tháng mục tiêu (tháng 15 cho tập kiểm định, tháng 16 cho tập kiểm tra).
        - **Tạo ma trận đặc trưng:** Tạo ma trận đặc trưng từ các cột đặc trưng trễ (lagged features) cho từng khách hàng trong mỗi tập dữ liệu.
        - **Gán nhãn (label):** Gán nhãn cho dữ liệu huấn luyện và kiểm định. Nhãn ở đây chính là danh sách các sản phẩm mới mà khách hàng đã sử dụng trong tháng mục tiêu (tháng 15).
3. **Xây dựng và Huấn luyện mô hình XGBoost:**

- **Chuyển đổi dữ liệu sang định dạng DMatrix:** Chuyển đổi dữ liệu từ DataFrame sang định dạng `DMatrix`, là định dạng dữ liệu đặc biệt được tối ưu hóa cho thư viện XGBoost.
- **Thiết lập tham số mô hình XGBoost:** Thiết lập các siêu tham số (hyperparameters) cho mô hình XGBoost, ví dụ: số lượng cây, độ sâu cây, tốc độ học, ... Các tham số này ảnh hưởng đến hiệu năng và tốc độ huấn luyện của mô hình.
  - `objective="multi:softprob"`: Bài toán Product Recommendation là phân loại nhiều lớp
  - `eta=0.1`: Tốc độ học thấp, tránh overfitting
  - `min_child_weight=10`: Kiểm soát độ phức tạp của mô hình
  - `max_depth=8`: Giới hạn độ sâu cây quyết định để tránh overfitting
  - `silent=1`: Tắt thông báo log không cần thiết
  - `eval_metric="mlogloss"`: Đánh giá mô hình bằng lỗi logarit đa lớp
  - `colsample_bytree=0.8, colsample_bylevel=0.9`: Chọn ngẫu nhiên các đặc trưng để tăng tính đa dạng
  - `num_class=len(products)`: Số lượng lớp bằng số sản phẩm cần dự đoán
  - `device="cuda"`: Sử dụng GPU để tăng tốc huấn luyện
- **Huấn luyện mô hình:** Huấn luyện mô hình XGBoost trên tập huấn luyện, sử dụng tập kiểm định để theo dõi hiệu năng của mô hình trong quá trình huấn luyện. Sử dụng kỹ thuật **dừng sớm (early stopping)** để ngăn chặn hiện tượng quá khớp (overfitting) và tìm ra điểm dừng huấn luyện tối ưu.
- **In ra Độ Quan Trọng của Đặc trưng (Feature Importance):** Sau khi huấn luyện, in ra độ quan trọng của từng đặc trưng đầu vào, cho biết mức độ đóng góp của mỗi đặc trưng vào kết quả dự đoán của mô hình.
- **Kết quả MAP@3 trên tập test:** 0.2406 tối thiểu, tối đa 0.2803.

4. **Dự đoán và tạo tệp submission:**

   - **Dự đoán trên tập kiểm tra:** Sử dụng mô hình XGBoost đã huấn luyện để dự đoán sản phẩm mới mà khách hàng sẽ sử dụng trên tập dữ liệu kiểm tra (tháng 16).
   - **Định nghĩa hàm tạo tệp submission:** Xây dựng một hàm để định dạng kết quả dự đoán thành tệp submission theo đúng yêu cầu của cuộc thi.
   - **Tạo tệp submission trong bộ nhớ:** Tạo tệp submission ở định dạng CSV hoặc TXT trong bộ nhớ máy tính.
   - **Ghi tệp submission ra file csv:** Lưu tệp submission từ bộ nhớ xuống file TXT để nộp bài.

Công cụ sử dụng:

- **pyarrow**: Hỗ trợ đọc và xử lý dữ liệu định dạng Parquet một cách hiệu quả, tối ưu RAM hơn RẤT NHIỀU so với pandas.
- **xgboost**: Thư viện hỗ trợ thuật toán XGBoost, mô hình dự đoán.
- **numpy**: Xử lý các phép tính toán số học và xử lý mảng đa chiều.
- **torch**: Kiểm tra và giải phóng bộ nhớ GPU (nếu có sử dụng GPU).
- **io, csv**: Các thư viện Python chuẩn để làm việc với input/output (IO) và định dạng file CSV, phục vụ cho việc tạo tệp submission.

Lưu ý:

Mô hình được thiết kế để có thể chạy trên GPU, giúp tăng tốc đáng kể quá trình huấn luyện. Ban giám khảo có thể truy cập đường dẫn Kaggle Notebook để chạy mô hình và kiểm tra kết quả.

Hình ảnh:

Tất cả hình ảnh minh họa (nếu có) đã được đính kèm trong thư mục code dưới dạng file Jupyter Notebook.

Chú thích thuật ngữ (đã được tích hợp trong báo cáo):

- **Data Type Conversion:** Chuyển đổi kiểu dữ liệu.
- **Date Formatting:** Định dạng ngày tháng.
- **Time Series Analysis:** Phân tích chuỗi thời gian.
- **Aggregated Statistics:** Thống kê tổng hợp.
- **Customer-Month Matrix:** Ma trận Khách hàng - Tháng.
- **First Occurrence:** Lần xuất hiện đầu tiên.
- **Average Income:** Thu nhập trung bình.
- **Positive Flank:** Sự đón nhận sản phẩm mới (thuật ngữ riêng của dự án).
- **MAP (Mean Average Precision):** Chỉ số đo lường hiệu quả mô hình.
- **Data Cleaning:** Làm sạch dữ liệu.
- **Feature Engineering:** Tạo đặc trưng.
- **Missing Values:** Giá trị thiếu.
- **Outliers:** Giá trị ngoại lai.
- **Mode:** Giá trị xuất hiện nhiều nhất.
- **Median:** Giá trị trung vị.
- **IntegerType:** Kiểu dữ liệu số nguyên.
- **Label Encoding:** Mã hóa nhãn.
- **Mapping Dictionaries:** Từ điển ánh xạ.
- **NULL:** Giá trị rỗng.
- **Lag Features:** Đặc trưng trễ.
- **Gap Detection:** Phát hiện khoảng trống.
- **Parquet:** Định dạng file cột tối ưu cho dữ liệu lớn.
- **DataFrame:** Cấu trúc dữ liệu dạng bảng hai chiều.
- **Dict:** Dictionary (từ điển Python).
- **GPU Memory Release:** Giải phóng bộ nhớ GPU.
- **Lagged Features:** Đặc trưng trễ.
- **Train, Validation, Test Sets:** Tập huấn luyện, tập kiểm định, tập kiểm tra.
- **XGBoost (Extreme Gradient Boosting):** Thuật toán học máy XGBoost.
- **DMatrix:** Định dạng dữ liệu của XGBoost.
- **Hyperparameters:** Siêu tham số.
- **Early Stopping:** Dừng sớm.
- **Feature Importance:** Độ quan trọng của đặc trưng.
- **Submission File:** Tệp submission.
