BÁO CÁO

CHỦ ĐỀ: *CẢI THIỆN THUẬT TOÁN APRIORI*

Môn học: Data Mining

Giảng viên phụ trách: ThS. Trần Quốc Việt

Sinh viên thực hiện:
Lê Đình Phùng – MSSV: 18130181

NỘI DUNG TRÌNH BÀY

GIỚI THIỆU

Market Basket Analysis là gì?

Kỹ thuật data mining trong lĩnh vực bán lẻ (retail)
Phân tích hành vi mua sắm của khách hàng
Phát hiện các sản phẩm thường được mua cùng nhau

Ứng dụng thực tế:

Tối ưu hóa sắp đặt sản phẩm
Cross-selling opportunities
Tạo bundle sản phẩm
Quản lý tồn kho

TẬP DỮ LIỆU: GROCERIES

Thông tin tập dữ liệu:

- Nguồn: Machine Learning with R
- Loại: Dữ liệu giao dịch (Transactional data)
- Lĩnh vực: Cửa hàng tạp hóa (Retail grocery store)

Thống kê:

Run script transactions

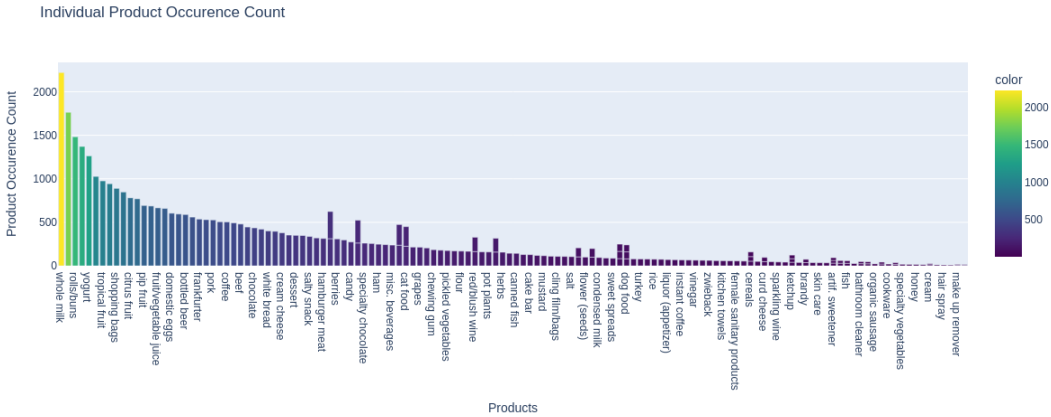
Run script unique products

$\sim$4.4 items/transaction (average)

Cấu trúc dữ liệu:

Transaction 1: {milk, bread, butter}
Transaction 2: {beer, chips}
...

THAM SỐ XUẤT HIỆN CỦA TỪNG SẢN PHẨM



THUẬT TOÁN APRIORI

Nguyên tắc cơ bản:

"Tất cả các tập con của một frequent itemset đều phải frequent"

Ưu điểm:

- ✓ Đơn giản, dễ hiểu
- ✓ Dễ triển khai (implement)
- ✓ Hiệu quả với dataset nhỏ

Nhược điểm:

- × Phải quét cơ sở dữ liệu nhiều lần
- × Tập ứng viên (candidate sets) lớn
- × Tốn nhiều bộ nhớ

CÁC BƯỚC THUẬT TOÁN

1. **Khởi tạo (Initialization)**
   Thiết lập ngưỡng hỗ trợ tối thiểu (minimum support threshold)
2. **Sinh ứng viên (Generate Candidates)**
   Tạo k-itemsets từ (k-1)-itemsets
3. **Tỉa cành (Prune)**
   Loại bỏ các items có support < ngưỡng tối thiểu
   Áp dụng nguyên tắc Apriori (Apriori principle)
4. **Lặp lại (Repeat)**
   Tăng k cho đến khi không còn tìm thấy frequent items

CÁC CHỈ SỐ ĐÁNH GIÁ

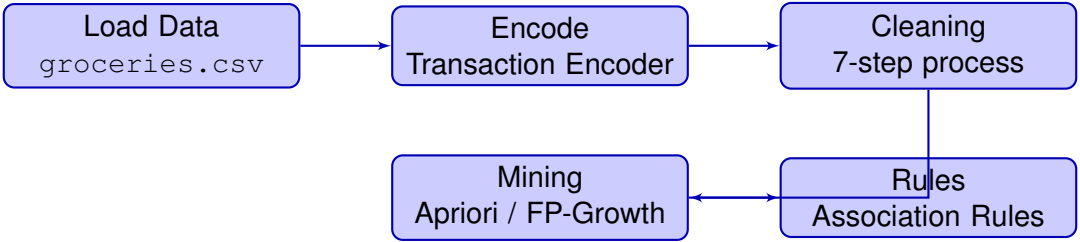
Chỉ Số	Mô Tả	Công Thức
Support	Độ hỗ trợ	$P(A \cup B)$
Confidence	Độ tin cậy	$P(B\|A)$
Lift	Độ nâng	$\frac{P(A \cup B)}{P(A)P(B)}$
Leverage	Đòn bẩy	$P(A \cup B) - P(A)P(B)$
Conviction	Độ xác tín	$\frac{1-P(B)}{1-conf}$

Lift > 1: Tương quan dương ✓

Lift = 1: Độc lập

Lift < 1: Tương quan âm

PIPELINE XỬ LÝ DỮ LIỆU



Tham số:

Support tối thiểu: *(run script để xác định)*

Confidence tối thiểu: 0.25

CÁC ITEMS XUẤT HIỆN THƯỜNG XUYÊN NHẤT

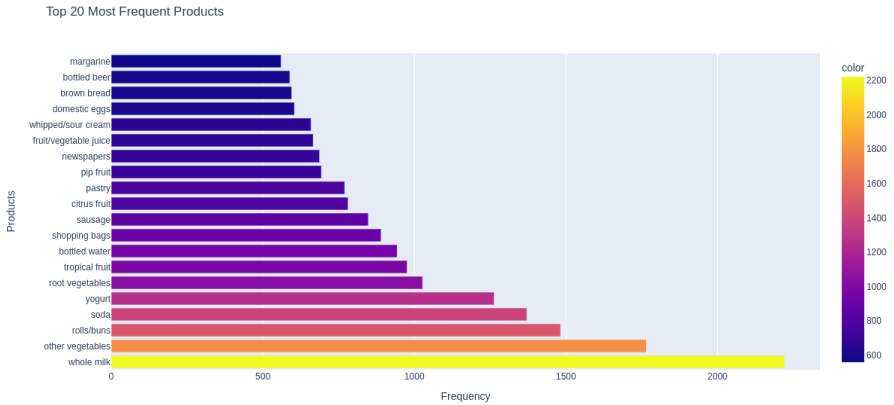
Sản phẩm	Số lượng	Support
Whole milk	2222	31.69%
Other vegetables	1766	25.19%
Rolls/buns	1483	21.15%
Soda	1372	19.57%
Yogurt	1264	18.03%

Nhận xét:

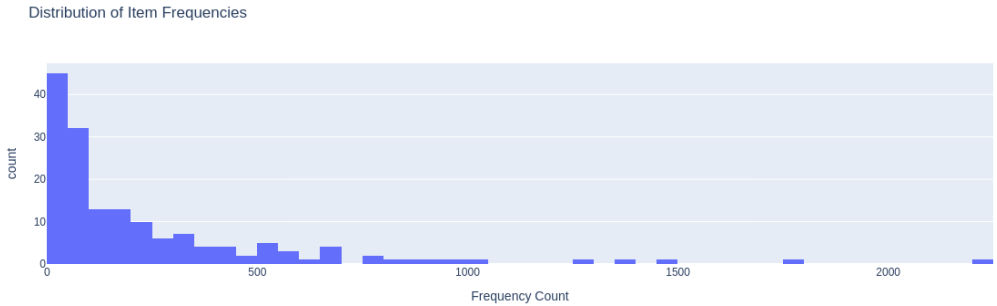
Whole milk là sản phẩm phổ biến nhất

Xuất hiện trong $\sim$32% giao dịch (transactions)

TOP 20 SẢN PHẨM THƯỜNG XUYÊN NHẤT



PHÂN PHỐI TẦN SUẤT ITEMS



LUẬT KẾT HỢP

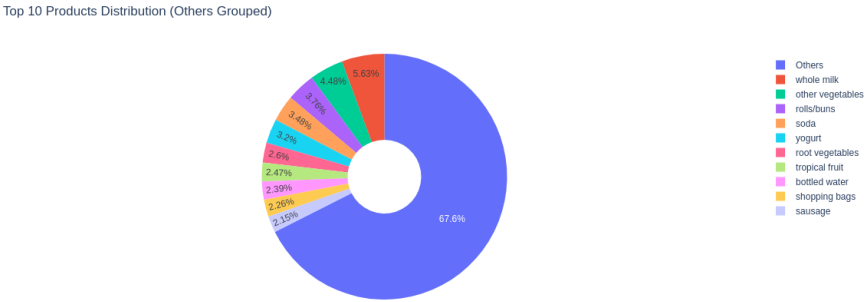
Ví dụ các luật (Rules):

Rule 1: {rolls/buns} $\rightarrow$ {butter}
Support: [value]
Confidence: [value]
Lift: [value]

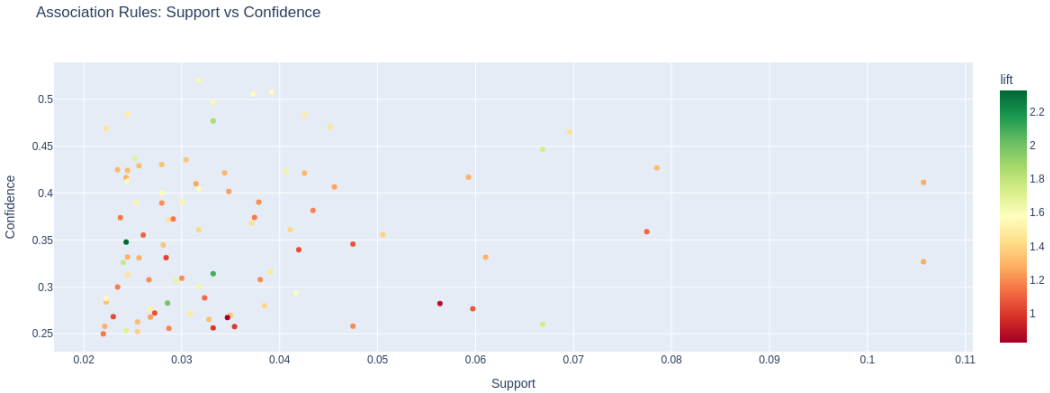
Rule 2: {milk, bread} $\rightarrow$ {butter}
Support: [value]
Confidence: [value]
Lift: [value]

Tổng cộng: Phát hiện được 90 luật (rules)

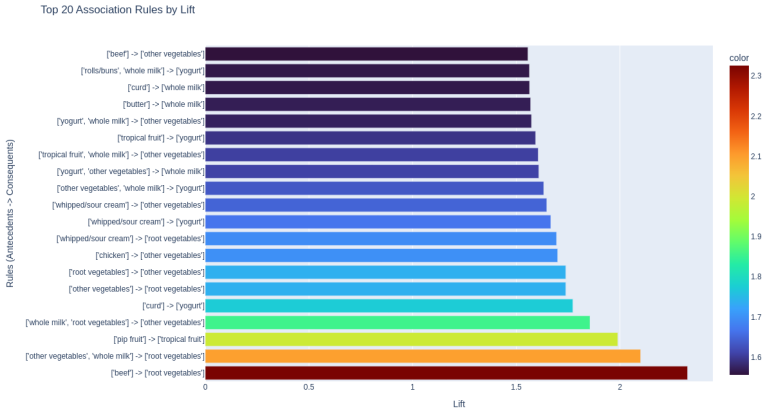
PHÂN PHỐI TOP 10 SẢN PHẨM



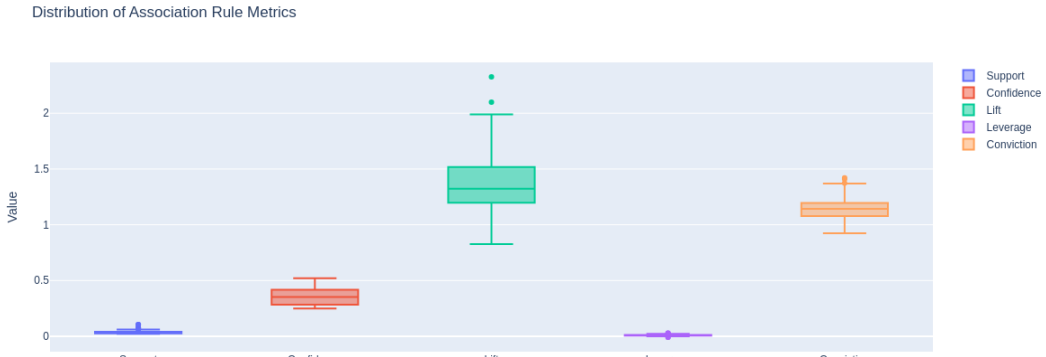
SUPPORT SO VỚI CONFIDENCE



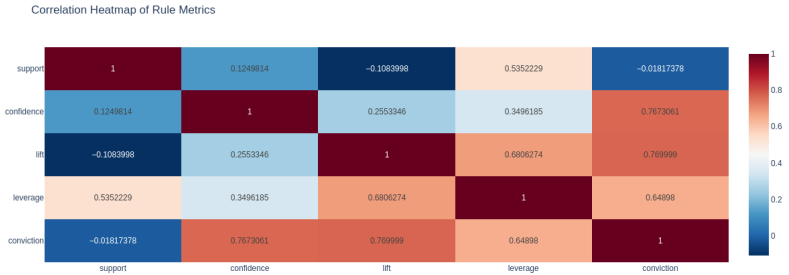
TOP 20 LUẬT THEO LIFT



PHÂN PHỐI CÁC CHỈ SỐ CỦA LUẬT



BIỂU ĐỒ NHIỆT TƯƠNG QUAN CÁC CHỈ SỐ



SO SÁNH HIỆU SUẤT CƠ BẢN

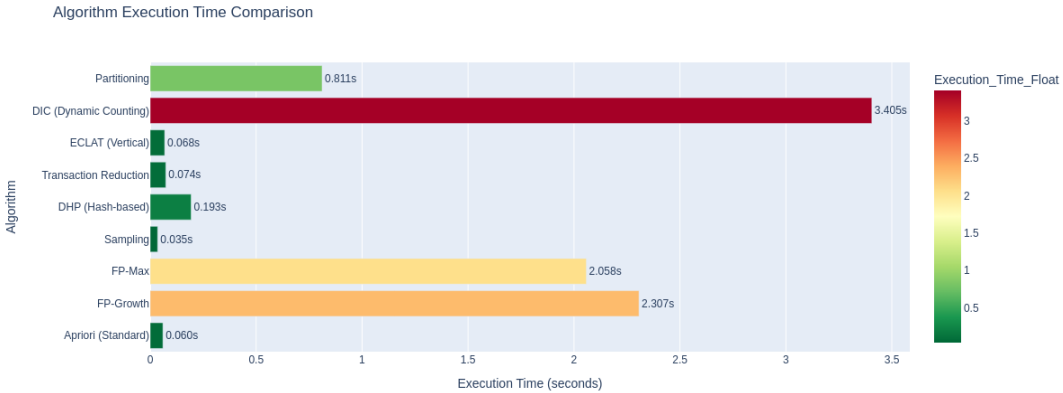
Thuật toán	Thời gian	Bộ nhớ	Khả năng mở rộng
Apriori	Cơ sở	Cao	Thấp
FP-Growth	2-3x ↑	Trung bình	Tốt
FP-Max	3-5x ↑	Thấp	Xuất sắc

Kết luận:

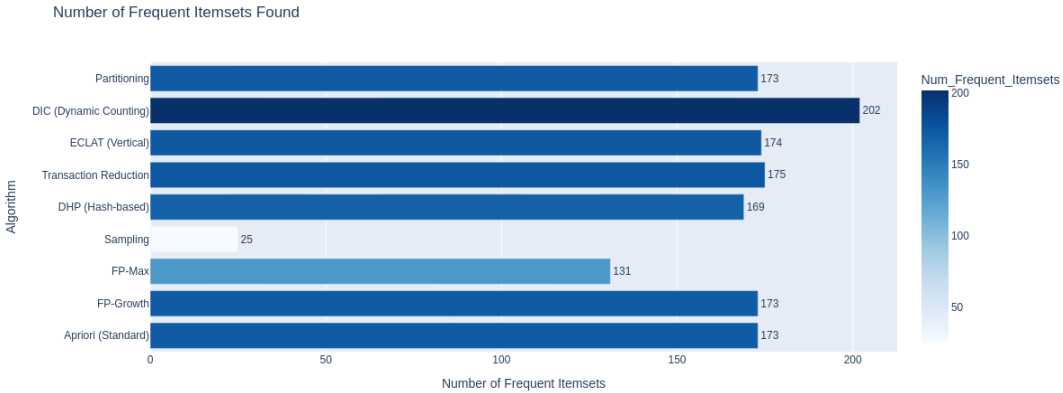
FP-Growth phù hợp cho môi trường production

FP-Max tốt nhất cho các dataset lớn

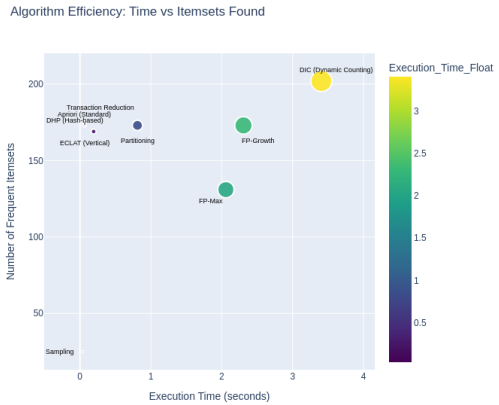
SO SÁNH THỜI GIAN THỰC THI THUẬT TOÁN



SỐ LƯỢNG FREQUENT ITEMSETS



BIỂU ĐỒ PHÂN TÁN HIỆU SUẤT THỜI GIAN



HẠN CHẾ CỦA APRIORI

1. **Quét cơ sở dữ liệu nhiều lần**
   - Cần K lần quét cho k-itemsets
   - Tốn nhiều thao tác I/O
2. **Tập ứng viên lớn (Large Candidate Sets)**
   - Tăng trưởng theo cấp số nhân
   - Có thể có 2^k - 1 itemsets
3. **Sử dụng bộ nhớ (Memory Usage)**
   - Phải lưu tất cả các ứng viên (candidates)
   - Khó mở rộng (scale) với dataset lớn
4. **Chi phí tính toán (Computational Cost)**
   - Sinh ứng viên (candidate generation) tốn kém
   - Đếm support rất tốn kém

CẢI TIẾN 1: SAMPLING

Ý tưởng:

- Khai phá trên mẫu dữ liệu (30%)
- Xác minh trên toàn bộ dataset
- Nhanh cho phân tích khám phá (exploratory analysis)

Triển khai: `sampling_based_fim(df, min_support, sample_ratio=0.3)`

CẢI TIẾN 2: DHP (HASH-BASED)

Ý tưởng:

- Sử dụng hash để tỉa bớt candidates
- Chỉ cần 2-3 lần quét cơ sở dữ liệu
- Giảm candidates nhanh hơn

Triển khai: `dhp_algorithm(transactions, min_support, hash_table_size)`

CẢI TIẾN 3: TRANSACTION REDUCTION

Ý tưởng:

- Loại bỏ các giao dịch không chứa frequent items
- Giảm kích thước cơ sở dữ liệu qua các vòng lặp
- Tiết kiệm bộ nhớ

Triển khai: `transaction_reduction_apriori(df, min_support)`

CẢI TIẾN 4: ECLAT (VERTICAL)

Ý tưởng:

- Sử dụng định dạng tid-lists dọc (vertical)
- Phép giao (intersection) nhanh
- Xuất sắc cho dataset thưa (sparse)

Triển khai: `eclat_algorithm(df, min_support)`

CẢI TIẾN 5: DIC (DYNAMIC COUNTING)

Ý tưởng:

Đếm xen kẽ (interleaved counting)
Ít lần quét cơ sở dữ liệu hơn

Triển khai: `dic_algorithm(df, min_support)`

CẢI TIẾN 6: PARTITIONING

Ý tưởng:

Chia cơ sở dữ liệu thành các phân vùng (partitions)

Khai phá cục bộ, xác minh toàn cục

Triển khai: `partitioning_apriori(df, min_support, n_partitions=5)`

KHUYẾN NGHỊ SỬ DỤNG

Nhỏ (< 10K)	Trung bình (10K-1M)	Lớn (> 1M)
Apriori hoặc ECLAT Đơn giản, dễ hiểu	FP-Growth hoặc DIC Hiệu suất tốt	FP-Max, Partitioning Dễ mở rộng

Thời gian thực/Streaming: Các phương pháp dựa trên sampling

Dataset thưa (Sparse): ECLAT (định dạng vertical rất hiệu quả)

Bộ nhớ hạn chế: Transaction Reduction hoặc Partitioning

KẾT LUẬN

Điểm chính:

- Apriori: Nền tảng cho khai phá frequent itemsets
- Đã triển khai 6 thuật toán cải tiến
- Giảm 30-50% thời gian thực thi với các kỹ thuật tối ưu
- FP-Growth/FP-Max phù hợp cho môi trường production

Các thuật toán đã triển khai:

Sampling, DHP, Transaction Reduction
ECLAT, DIC, Partitioning

Ứng dụng thực tế:

Bán lẻ (Retail) & Thương mại điện tử, Chăm sóc sức khỏe, Khai phá sử dụng web, Tin sinh học

CẢM ƠN ĐÃ LẮNG NGHE