**TRƯỜNG ĐẠI HỌC KINH TẾ - LUẬT**

**MÔN HỌC: NHÀ KHO DỮ LIỆU VÀ KHAI PHÁ DỮ LIỆU**

**KHAI THÁC CÁC TẬP MỤC HỮU ÍCH CAO SẴN CÓ TỪ CƠ SỞ DỮ LIỆU ADVENTUREWORKS 2022 BẰNG THUẬT TOÁN FOSHU**

**GVHD:**

Thầy Nguyễn Thôn Dã

*Thành phố Hồ Chí Minh, 202**3*

# 

**MỤC LỤC**

[**DANH MỤC TỪ VIẾT TẮT 1**](#_Toc148902339)

[**DANH MỤC HÌNH ẢNH 2**](#_Toc148902340)

[**DANH MỤC BẢNG BIỂU 4**](#_Toc148902341)

[**A. TỔNG QUAN 5**](#_Toc148902342)

[**I.** **Lý do chọn đề tài** 5](#_Toc148902343)

[**II.** **Mục tiêu thực hiện đề tài** 6](#_Toc148902344)

[**III.** **Đối tượng và phạm vi nghiên cứu** 6](#_Toc148902345)

[**IV.** **Phương pháp nghiên cứu** 6](#_Toc148902346)

[**V.** **Công cụ** 6](#_Toc148902347)

[**VI.** **Quy trình thực hiện** 7](#_Toc148902348)

[**B. MÔ HÌNH ĐỀ XUẤT 10**](#_Toc148902349)

[**I.** **Lý thuyết mô hình** 10](#_Toc148902350)

[*1.* *Giới thiệu về thuật toán* 10](#_Toc148902351)

[*2.* *Ưu và nhược điểm của thuật toán* 10](#_Toc148902352)

[**II.** **Yêu cầu của thuật toán** 12](#_Toc148902353)

[**C. KẾT QUẢ THỰC NGHIỆM VÀ PHÂN TÍCH 15**](#_Toc148902354)

[**I.** **Chuyển đổi dữ liệu** 15](#_Toc148902355)

[*1.* *Truy xuất và chuyển đổi dữ liệu đầu vào* 15](#_Toc148902356)

[*2.* *Khai phá dữ liệu (Exploratory Data Analysis – EDA)* 18](#_Toc148902357)

[*3.* *Phân tích tập dữ liệu đầu vào* 27](#_Toc148902358)

[**II.** **Kết quả (đầu ra)** 28](#_Toc148902359)

[**III.** **Phân tích kết quả** 34](#_Toc148902360)

[1. Trực quan hóa và phân tích kết quả 34](#_Toc148902361)

[2. Phân tích hiệu suất thuật toán 41](#_Toc148902362)

[**D. KẾT LUẬN 44**](#_Toc148902363)

[**I.** **Nhận xét** 44](#_Toc148902364)

[**II.** **Kết luận** 45](#_Toc148902365)

[**III.** **Hướng phát triển** 45](#_Toc148902366)

[**THAM KHẢO 47**](#_Toc148902367)

# DANH MỤC TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| **Từ viết tắt** | **Từ gốc** | **Giải thích** |
| SPMF | Sequential Pattern Mining Framework | Khung khai thác mẫu tuần tự |
| EDA | Exploratory Data Analysis | Khai phá và xử lý dữ liệu |
| DB | Database | Cơ sở dữ liệu |
| CSV | Comma Separated Values | Các Giá Trị Phân Cách Bằng Dấu Phẩy (một định dạng file dữ liệu với đuôi file là “.csv”) |
| MB | Megabyte | Đơn vị thông tin hoặc dung lượng tin học |

# DANH MỤC HÌNH ẢNH

[*Hình 1. Quy trình thực hiện nghiên cứu 7*](#_Toc148994206)

[*Hình 2. Kiểm tra dữ liệu null của Dataset1 19*](#_Toc148994207)

[*Hình 3. Kiểm tra cột chứa giá trị null của Dataset1 19*](#_Toc148994208)

[*Hình 4. Biểu đồ cột và tròn thể hiện số lượng và tỷ trọng giá trị null 20*](#_Toc148994209)

[*Hình 5. Kiểm tra dữ liệu null của Dataset2 20*](#_Toc148994210)

[*Hình 6. Kiểm tra cột chứa giá trị null của Dataset2 21*](#_Toc148994211)

[*Hình 7. Kiểm tra giá trị null của Dataset3 22*](#_Toc148994212)

[*Hình 8. Kiểm tra cột chứa giá trị null của Dataset3 22*](#_Toc148994213)

[*Hình 9. Kiểm tra các cột dữ liệu sau khi chỉnh sửa kiểu dữ liệu ModifiedDate và loại bỏ cột Name 23*](#_Toc148994214)

[*Hình 10. Kiểm tra khoảng thời gian của tập dữ liệu 23*](#_Toc148994215)

[*Hình 11.Mô tả về dataset 24*](#_Toc148994216)

[*Hình 12. Mô tả về dataset (tt) 24*](#_Toc148994217)

[*Hình 13. Biểu đồ miêu tả sự phân phối của các chỉ số 25*](#_Toc148994218)

[*Hình 14. Biểu đồ hệ số tương quan giữa các dữ liệu 26*](#_Toc148994219)

[*Hình 15. Biểu đồ đường về các chỉ số Sales Order, Revenue, Profit qua thời gian 27*](#_Toc148994220)

[*Hình 16. Hình kết quả đầu ra dưới dạng dataframe 31*](#_Toc148994221)

[*Hình 17. Biểu đồ so sánh tiện ích (utility – lợi nhuận) và tiện ích tương đối (relative utility – tỷ lệ lợi nhuận) của các tập mục (sản phẩm) đi cùng nhau 35*](#_Toc148994222)

[*Hình 18. Biểu đồ tròn thể hiện tỷ lệ phần trăm tần suất xuất hiện của toàn bộ Itemsets qua 4 mùa 36*](#_Toc148994223)

[*Hình 19. Biểu đồ thể hiện mối quan hệ giữa tần suất xuất hiện của các Itemsets và 4 mùa trong năm 37*](#_Toc148994224)

[*Hình 20. Biểu đồ thể hiện mối quan hệ giữa tần suất xuất hiện của các Itemsets và 4 mùa trong năm 37*](#_Toc148994225)

[*Hình 21. Biểu đồ thể hiện mối quan hệ giữa tổng lợi nhuận của các Itemsets và 4 mùa trong năm 38*](#_Toc148994226)

[*Hình 22. Biểu đồ đường thể hiện tổng lợi nhuận của các Itemsets trong 4 mùa 39*](#_Toc148994227)

[*Hình 23. Biểu đồ đường thể hiện tổng lợi nhuận của các Itemsets trong 4 mùa 40*](#_Toc148994228)

[*Hình 24. Biểu đồ đường thể hiện tần suất xuất hiện của các Itemsets trong 4 mùa 40*](#_Toc148994229)

# DANH MỤC BẢNG BIỂU

[*Bảng 1. So sánh thuật toán FOSHU thực thi trên bộ dữ liệu được phân chia theo từng năm và tổng thể thời gian 40*](#_Toc148902692)

# 

# TỔNG QUAN

## **Lý do chọn đề tài**

Doanh thu và lợi nhuận bán hàng luôn là vấn đề quan trọng mà các doanh nghiệp, công ty, siêu thị luôn phải chú ý đến. Để có thể có một chiến lược bán hàng tốt nhất, các doanh nghiệp phải đặt việc tính toán doanh số và tối ưu hóa lợi nhuận bán lên hàng đầu. Tuy nhiên, mỗi hàng hóa có giá cả, hạn sử dụng và thời gian mà hàng hóa đó có mặt ở trên kệ khác nhau, mỗi giờ số lượng giao dịch là một con số rất lớn. Vì vậy việc tính toán các mặt hàng đem lại doanh số cao, mặt hàng bán không chạy trong một khoảng thời gian trở nên khó khăn do dữ liệu quá lớn và xảy ra quá dày đặc.

Bài toán khai thác tập mục lợi ích cao (High-utility itemsets mining - HUIM) để tìm ra các tổ hợp đem lại lợi ích cao nhất từ cơ sở dữ liệu giao dịch đã xảy ra và được lưu lại. Có khá nhiều thuật toán hiệu quả đã được đề xuất để giải quyết bài toán này, tuy nhiên, các thuật toán này đều có điểm chung là giả định các mặt hàng đều có thời hạn sử dụng là như nhau, tức là tất cả các mặt hàng đều được áp dụng chiến lược giá hay giảm giá trong cùng một khoảng thời gian, và lợi nhuận của mặt hàng này đều không có số âm. Tuy nhiên, trong thực tế các cửa hàng có thể chấp nhận bán lỗ để thu hút khách hàng và các mặt hàng có thể được bán trong khoảng thời gian ngắn. Điều này dẫn đến sự không cân bằng giữa các mặt hàng, và có ảnh hưởng lớn đến chiến lược bán hàng.

Thuật toán khai thác tập mục tiện ích cao có sẵn nhanh chóng (FOSHU) được Philippe Fournier-Viger và cộng sự giới thiệu vào năm 2015, dùng để khai thác các tập mục tiện ích cao trong cơ sở dữ liệu có giá trị tiện ích âm và thời hạn sử dụng của các mục đều được cân nhắc.

Từ những vấn đề và lý do trên, nhóm chúng tôi đã quyết định chọn đề tài ***“Khai thác các tập mục hữu ích cao sẵn có từ cơ sở dữ liệu AdventureWorks 2022 bằng thuật toán FOSHU”***.

## **Mục tiêu thực hiện đề tài**

Mục tiêu cụ thể của đồ án bao gồm:

*Một là*: Nghiên cứu các lý thuyết, cụm từ liên quan, thuật toán xử lý dữ liệu giao dịch như thuật toán FOSHU nói riêng và các thuật toán khác nói chung.

*Hai là*: Nắm bắt được tình hình của các cụm Itemsets trong các mùa trong năm. Phân tích được tình hình, thời gian trên kệ, cũng như lợi nhuận của từng Itemset để đưa ra các chiến lược xử lý nhằm tăng lợi nhuận.

## **Đối tượng và phạm vi nghiên cứu**

* *Đối tượng nghiên cứu*: Các sản phẩm, giá bán, lợi nhuận trong tập dữ liệu AdventureWorks qua các năm, thuật toán FOSHU.
* *Phạm vi nghiên cứu*: Tập dữ liệu AdventureWorks 2022.

## **Phương pháp nghiên cứu**

*Dựa vào tài liệu:*

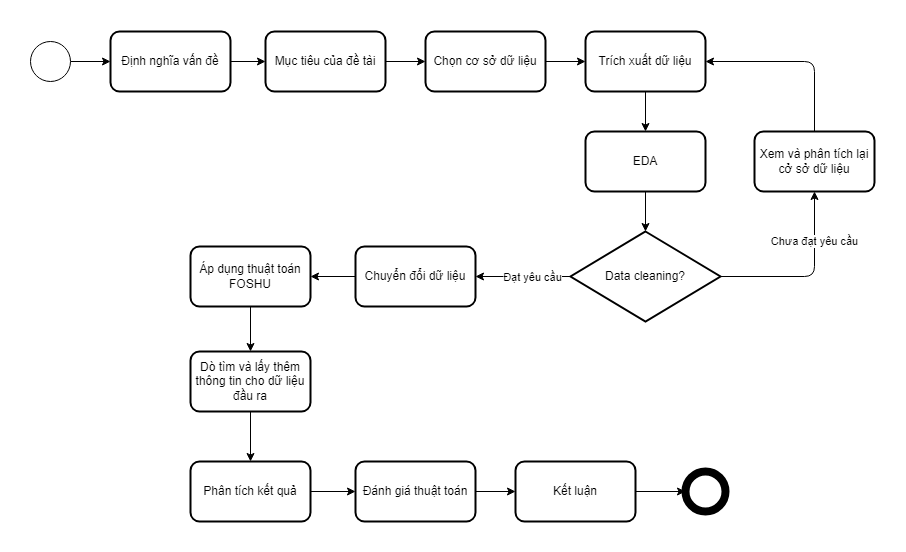
* Nghiên cứu những tài liệu giáo trình chính thức, internet, các bài báo khoa học có liên quan đến thuật toán và những vấn đề của thuật toán để phục vụ cho việc phân tích đề tài.
* Để có thể phân tích và xây dựng mô hình thì nhóm cần tìm hiểu về các phương pháp khai thác, xây dựng những mô hình, những công nghệ mới có liên quan.

*Dựa vào nghiên cứu thực nghiệm*: Nhóm sử dụng thuật toán FOSHU để khai thác các tập mục tiện ích cao trong cơ sở dữ liệu có giá trị tiện ích âm và thời hạn sử dụng của các mục đều được cân nhắc.

## **Công cụ**

* Công cụ: SPMF, SQL Server.
* Ngôn ngữ: Python.

## **Quy trình thực hiện**



Hình . Quy trình thực hiện nghiên cứu

*Bước 1: Đặt vấn đề và định nghĩa vấn đề.*

* Trong bước đầu tiên, nhóm chọn thuật toán để thực hiện báo cáo, cùng với đó là chọn đề tài phù hợp với thuật toán đã chọn.

*Bước 2: Xác định mục tiêu đề tài.*

* Trong bước tiếp theo, nhóm xác định mục tiêu cụ thể của đề tài, các phương pháp nghiên cứu và phạm vi nghiên cứu.

*Bước 3: Chọn cơ sở dữ liệu.*

* Vì đầu vào của thuật toán yêu cầu các tập mục (Itemsets) đi cùng nhau, kèm theo đó là tiện ích (utility - giá trị mà nó mang lại) và khoảng thời gian nó được xác định (period time). Nên yêu cầu ở đây là một bộ dữ liệu tập hợp các giao dịch kinh doanh (Transaction), trong đó phải có một số hay nhiều giao dịch có các tập mục (các sản phẩm) đi cùng nhau. Đồng thời, có các thông tin cần thiết để tính toán tiện ích và có dữ liệu về thời gian các mặt hàng được bán hay được xác định. Dựa vào các những yêu cầu đó, nhóm đã tiến hành xem xét, tìm kiếm và đã quyết định chọn cơ sở dữ liệu AdventureWorks2022 vì nó phù hợp và đáp ứng các yêu cầu đề ra.

*Bước 4: Trích xuất dữ liệu.*

* Trong bước này nhóm sử dụng hệ quản trị cơ sở dữ liệu SQL Server để thực hiện viết các câu truy vấn để phân tích và trích xuất các dữ liệu cần thiết để có thể tính toán dữ liệu đầu vào phù hợp cho việc chạy thuật toán từ cơ sở dữ liệu đã chọn. Sau khi các câu truy vấn đã lấy được dữ liệu cần thiết nhóm thực hiện kết nối với cơ sở dữ liệu bằng Python để trích xuất dữ liệu vào file CSV và đi đến bước tiếp theo là EDA dữ liệu.

*Bước 5: EDA.*

* Nhóm tiến hành tiền xử lý dữ liệu, làm sạch dữ liệu với Python sau khi đã có được dữ liệu cần thiết, nếu dữ liệu chưa đạt yêu cầu, nhóm sẽ tiến hành xem và phân tích lại cơ sở dữ liệu, nếu dữ liệu cho đến khi dữ liệu đã đạt yêu cầu nhóm sẽ tiến hành bước tiếp theo.

*Bước 6: Chuyển đổi dữ liệu.*

* Từ tập dữ liệu đã đạt yêu cầu, nhóm thực hiện các câu lệnh theo ngôn ngữ Python để tiến hành chuyển đổi dữ liệu bằng cách tính toán, nhóm các dữ liệu liên quan lại với nhau và sau đó thực hiện nối chuỗi cho đúng định dạng đầu vào của thuật toán FOSHU trong SPMF. Sau đó lưu dữ liệu đã chuyển đổi định dạng vào file văn bản có đuôi “.txt” theo định dạng chuẩn SPMF.

*Bước 7: Áp dụng thuật toán FOSHU.*

* Từ tập dữ liệu đã được định dạng theo chuẩn đầu vào của thuật toán, nhóm tiến hành áp dụng thuật toán FOSHU (được gọi, chạy bằng ngôn ngữ Python) và chọn mức tiện ích tối thiểu phù hợp để tìm ra kết quả tốt nhất.

*Bước 8: Dò tìm và lấy thêm thông tin cho dữ liệu đầu ra.*

* Sau khi chạy thành công thuật toán FOSHU, nhóm có kết quả theo định dạng đầu ra của thuật toán, nhưng chưa thể xác định các thông tin chi tiết liên quan. Vì vậy, nhóm thực hiện dò kết quả sau khi chạy thuật toán có được tương ứng với dữ liệu trong bộ dữ liệu để lấy thêm thông tin cần thiệt cho việc phân tích và đánh giá.

*Bước 9: Phân tích kết quả.*

* Trong bước này nhóm tiến hành phân tích những kết quả đạt được bằng cách trực quan hóa dữ liệu với nhiều mô hình khác nhau. Đồng thời, so sánh các số liệu thể hiện hiệu suất cũng như chi phí mà thuật toán tiêu hao.

*Bước 10: Đánh giá kết quả thuật toán.*

* Sau khi phân tích kết quả, nhóm tiến hành đánh giá, nhận xét kết quả của thuật toán, đồng thời nêu ra những hạn chế của thuật toán, những khó khăn khi thực hiện khai phá dữ liệu bằng thuật toán.

*Bước 10: Kết luận.*

* Sau khi có kết quả và đánh giá thuật toán, nhóm rút ra những kết luận về thuật toán, về bộ dữ liệu cũng như đề tài của nhóm sau khi hoàn thành bài báo cáo.

# MÔ HÌNH ĐỀ XUẤT

## **Lý thuyết mô hình**

### *Giới thiệu về thuật toán*

FOSHU (Fournier-Viger et al., 2015) là một thuật toán dùng để phát hiện các tập hợp mặt hàng có lợi nhuận cao trong cơ sở dữ liệu giao dịch. Nhiệm vụ của nó là mở rộng nhiệm vụ khai thác các tập hợp mặt hàng có lợi nhuận cao lên cả việc xem xét thời gian mặt hàng được bán trên kệ hàng, được gọi là "on-shelf high-utility itemset mining" (Philippe & Souleymane, FOSHU: Faster On-Shelf High Utility Itemset Mining -with or without Negative Unit Profit, 2015).

FOSHU có thể giải quyết được hai hạn chế của thuật toán khai thác tập mục có tính tiện ích cao:

* ***Mặt hàng có giá trị âm***: FOSHU có khả năng xử lý mặt hàng có giá trị âm, điều mà các thuật toán truyền thống không thể làm được. Điều này quan trọng trong trường hợp cửa hàng áp dụng chiến lược giảm giá để kích thích mua sắm hoặc thu hút khách hàng đến cửa hàng.
* ***Thời gian được bán trên kệ hàng***: FOSHU không giả định rằng tất cả mặt hàng có cùng thời gian được bán trên kệ hàng, mà nó hiểu rõ rằng các mặt hàng có thời gian được bán trên kệ hàng khác nhau. Điều này loại bỏ sự thiên vị và giúp tìm ra các tập hợp mặt hàng có giá trị cao dựa trên thời gian sử dụng cụ thể mà mỗi mặt hàng có sẵn trên kệ (Philippe & Souleymane, FOSHU: Faster On-Shelf High Utility Itemset Mining -with or without Negative Unit Profit, 2015).

### *Ưu và nhược điểm của thuật toán*

#### Ưu điểm

* *Xử lý mặt hàng có giá trị âm*: FOSHU có khả năng xử lý mặt hàng có giá trị âm, giúp loại bỏ sự hạn chế của các thuật toán truyền thống. Điều này quan trọng trong trường hợp cửa hàng áp dụng chiến lược giảm giá để kích thích mua sắm hoặc thu hút khách hàng đến cửa hàng.
* *Tối ưu hóa lợi nhuận và trọng lượng*: FOSHU tích hợp thông tin về lợi nhuận và trọng lượng của từng mặt hàng trong quá trình khai thác. Điều này giúp tìm ra các tập hợp mặt hàng có lợi nhuận cao và trọng lượng lớn, giúp tối ưu hóa hiệu suất kinh doanh.
* *Xem xét thời gian trên kệ*: Thuật toán FOSHU hiểu rõ rằng các mặt hàng có thời gian trên kệ khác nhau và không giả định rằng tất cả chúng có cùng thời gian bán hàng. Điều này loại bỏ sự thiên vị và giúp tìm ra các tập hợp mặt hàng có giá trị cao dựa trên thời gian cụ thể mà mỗi mặt hàng có sẵn trên kệ (Philippe & Souleymane, FOSHU: Faster On-Shelf High Utility Itemset Mining -with or without Negative Unit Profit, 2015).
* *Hiệu quả với dữ liệu lớn*: FOSHU được thiết kế để xử lý cơ sở dữ liệu giao dịch lớn một cách hiệu quả. Nó sử dụng các kỹ thuật tối ưu hóa và phân tách giao dịch để giảm áp lực về tài nguyên tính toán và thời gian thực thi (Philippe, Example: Mining On-Shelf High-Utility Itemsets from a Transaction Database using the FOSHU Algorithm (SPMF - Java), 2015).
* *Ứng dụng đa dạng*: FOSHU có ứng dụng đa dạng trong nhiều lĩnh vực, bao gồm quản lý cửa hàng, tiếp thị, phân tích dữ liệu giao dịch, và nghiên cứu thị trường. Nó có khả năng giúp doanh nghiệp tối ưu hóa chiến lược kinh doanh và tăng cường lợi nhuận.

#### Nhược điểm

* *Yêu cầu dữ liệu chính xác*: FOSHU đòi hỏi dữ liệu phải chính xác và đầy đủ. Nó phụ thuộc vào thông tin về lợi nhuận và trọng lượng của từng mặt hàng. Nếu dữ liệu không đủ chính xác hoặc thiếu thông tin về lợi nhuận và trọng lượng, kết quả có thể không đáng tin cậy.
* *Khả năng tính toán*: Xử lý cơ sở dữ liệu giao dịch lớn có thể đòi hỏi nhiều tài nguyên tính toán và thời gian. Điều này đặc biệt đúng khi dữ liệu phức tạp hoặc chứa nhiều mặt hàng. FOSHU có thể đánh bại nguồn tài nguyên tính toán và kéo dài thời gian thực hiện quá trình khai thác (Philippe & Souleymane, FOSHU: Faster On-Shelf High Utility Itemset Mining -with or without Negative Unit Profit, 2015).
* *Khả năng dự đoán giá trị tương lai*: Mặc dù FOSHU có khả năng xem xét thời gian mặt hàng được bán trên kệ, nó không thể dự đoán giá trị tương lai của mặt hàng. Điều này đặc biệt quan trọng nếu bạn cần dự đoán sự thay đổi trong giá trị hoặc thời gian trên kệ trong tương lai.
* *Phụ thuộc vào điều kiện khai thác*: Hiệu suất của FOSHU có thể phụ thuộc vào cách điều chỉnh các điều kiện và thông số trong quá trình khai thác. Điều này đòi hỏi sự hiểu biết sâu rộng về cơ sở dữ liệu và mục tiêu cụ thể của việc khai thác (Philippe & Souleymane, FOSHU: Faster On-Shelf High Utility Itemset Mining -with or without Negative Unit Profit, 2015).
* *Phụ thuộc vào dữ liệu giao dịch*: FOSHU là một thuật toán đặc biệt dành cho việc khai thác dữ liệu giao dịch. Nó không thích hợp cho mọi loại dữ liệu hoặc ứng dụng. Nếu bạn không có dữ liệu giao dịch, thuật toán này không thể sử dụng.

## **Yêu cầu của thuật toán**

**Đầu vào**:

* Dữ liệu đầu vào cần cung cấp các thông tin sau:
* *Danh sách các giao dịch*: Đây là danh sách các giao dịch mà khách hàng đã thực hiện. Mỗi giao dịch là một tập hợp các mặt hàng được mua trong cùng một lần mua sắm (Philippe, Example: Mining On-Shelf High-Utility Itemsets from a Transaction Database using the FOSHU Algorithm (SPMF - Java), 2015).
* *Danh sách các mặt hàng trong mỗi giao dịch*: Đối với mỗi giao dịch, bạn cần biết danh sách các mặt hàng được mua. Mỗi mặt hàng cần có thông tin về lợi nhuận mà nó mang lại và thời gian trên kệ.
* Về định dạng: Đầu vào của thuật toán yêu cầu có định dạng “.txt”, mỗi dòng đại diện cho một giao dịch, mỗi dòng bao gồm các thông tin như sau:
* Các mục (item) có trong giao dịch được liệt kê, mỗi mục được thể hiện bằng số nguyên dương, mỗi mục được phân cách nhau bằng một khoảng trắng. Giả định tất cả các mục trong cùng một dòng được sắp xếp theo thứ tự (tăng dần/giảm dần), không có mục nào xuất hiện hai lần trong cùng một dòng (Philippe, Example: Mining On-Shelf High-Utility Itemsets from a Transaction Database using the FOSHU Algorithm (SPMF - Java), 2015).
* Sau ký hiệu “:” là tiện ích giao dịch.
* Sau ký hiệu “:” là tiện ích của từng mục trong giao dich, được thể hiện bằng số nguyên (âm/dương).
* Sau ký hiệu “:” là một số nguyên dương thể hiện cho khoảng thời gian của giao dịch.

**Đầu ra**:

* Dữ liệu đầu ra của thuật toán FOSHU bao gồm tập hợp các tập mục tiện ích cao sẵn có có tiện ích tương đối không nhỏ hơn ngưỡng min\_utility\_ratio do người dùng đặt, trong các tập mục đó bao gồm các thông tin sau:
* *Tập mục (Itemset):* Đây là một tập hợp không có thứ tự của các mục riêng biệt.
* *Tiện ích của tập mục trong giao dịch (Utility of an Itemset in a Transaction):* Được tính bằng tổng tiện ích của các mục trong giao dịch.
* *Tiện ích của tập mục trong cơ sở dữ liệu (Utility of an Itemset in the Database):* Được tính bằng tổng tiện ích của tập mục đó trong tất cả các giao dịch mà nó xuất hiện (Philippe, Example: Mining On-Shelf High-Utility Itemsets from a Transaction Database using the FOSHU Algorithm (SPMF - Java), 2015).
* *Tiện ích tương đối của tập mục (Relative Utility of an Itemset):* Được tính bằng cách chia tiện ích của tập mục cho tổng tiện ích giao dịch trong khoảng thời gian mà tập mục được bán (Philippe, Example: Mining On-Shelf High-Utility Itemsets from a Transaction Database using the FOSHU Algorithm (SPMF - Java), 2015).
* *Tập mục có tiện ích cao sẵn có (Frequent High-Utility Itemset):* Tập mục mà tiện ích tương đối của nó không nhỏ hơn giá trị ngưỡng tỷ lệ tiện ích tối thiểu (*min\_utility\_ratio*) (Philippe, Example: Mining On-Shelf High-Utility Itemsets from a Transaction Database using the FOSHU Algorithm (SPMF - Java), 2015).
* Về định dạng: Đầu ra của thuật toán cũng yêu cầu định dạng “.txt”, mỗi dòng thể hiện cho một tập mục hữu ích cao có sẵn. Trong mỗi dòng có các thông tin như sau:
* Mỗi mục được biểu diễn bằng một số nguyên, mỗi mục cách nhau bằng một khoảng trắng.
* Sau đó là “#UTIL”: tiện ích của tập mục.
* Sau đó là “#RUTIL”: tiện ích tương đối của tập mục.

# C. KẾT QUẢ THỰC NGHIỆM VÀ PHÂN TÍCH

## **Chuyển đổi dữ liệu**

### *Truy xuất và chuyển đổi dữ liệu đầu vào*

Với cơ sở dữ liệu từ AdventureWork2022, thực hiện các câu truy vấn để kết nối với SQL Server, lấy các dữ liệu cần thiết cho bài làm, xuất kết quả từ các câu truy vấn đó sang file CSV.

|  |
| --- |
| import pyodbc  import pandas as pd  print(pyodbc.drivers())  SERVER = r'DESKTOP-5C33P4O\BCT'  DATABASE = 'AdventureWorks2022'  USERNAME = 'mkla123'  PASSWORD = '123'  Encrypt = 'no'  connectionString = f'DRIVER={{ODBC Driver 18 for SQL Server}};\  SERVER={SERVER};\  DATABASE={DATABASE};\  Encrypt={Encrypt};\  UID={USERNAME};PWD={PASSWORD}'  conn = pyodbc.connect(connectionString)  with open('QueryForDataset1.sql','r') as file:  sql\_query\_1 = file.read()  with open('QueryForDataset2.sql','r') as file:  sql\_query\_2 = file.read()  with open('QueryForDataset3.sql','r') as file:  sql\_query\_3 = file.read()  dataset1 = pd.read\_sql\_query(sql\_query\_1,conn)  dataset2 = pd.read\_sql\_query(sql\_query\_2,conn)  dataset3 = pd.read\_sql\_query(sql\_query\_3,conn)  df1 = pd.DataFrame(dataset1)  df2 = pd.DataFrame(dataset2)  df3 = pd.DataFrame(dataset3)  df1.to\_csv (r'D:\Nam4\_HKI\NhaKho\_DuLieu\Midterm\_Project\Dataset1.csv', index = False)  df2.to\_csv (r'D:\Nam4\_HKI\NhaKho\_DuLieu\Midterm\_Project\Dataset2.csv', index = False)  df3.to\_csv (r'D:\Nam4\_HKI\NhaKho\_DuLieu\Midterm\_Project\Dataset3.csv', index = False) |

Dùng các câu lệnh sau để chuyển đổi dữ liệu từ dataset (CSV) về định dạng đầu vào của thuật toán FOSHU trong SPMF. Sau đó lưu dữ liệu đã chuyển đổi định dạng vào file có đuôi “.txt”.

|  |
| --- |
| from EDA3 import \*  class Convert\_to\_SPMF:  def \_\_init\_\_(self):  self.df\_org = EDA\_3().get\_df\_eda\_processed()  self.df\_grouped\_lookup = None  def group\_by(self):  df = self.df\_org.copy()  df['Season'] = pd.cut(df['ModifiedDate'].dt.month, bins=[1, 4, 7, 10, 13], labels=[0, 1, 2, 3], include\_lowest=True)  df['Positive\_Profit\_Total'] = (df['UnitPrice'] \* (1 - df['UnitPriceDiscount']) - df['ProductCost']).clip(lower=0)  df['Profit\_Unit'] = (df['UnitPrice'] \* (1 - df['UnitPriceDiscount']) - df['ProductCost'])  grouped = df.groupby('SalesOrderID').agg({  'ProductID': lambda x: ' '.join(map(str, x)),  'Positive\_Profit\_Total': lambda x: round(x.sum(), 0).astype(int),  'Profit\_Unit': lambda x: ' '.join(map(str, round(x, 0).astype(int))),  'Season': 'max'  }).reset\_index()  return grouped  def format\_SPMF(self):  grouped = self.group\_by()  *# Tạo chuỗi kết quả*  grouped['Result'] = grouped['ProductID'] + ':' + grouped['Positive\_Profit\_Total'].astype(str) + ':' + grouped[  'Profit\_Unit'] + ':' + grouped['Season'].astype(str)  spmf\_format = grouped['Result'].copy()  return spmf\_format  def save\_into\_txt(self, spmf\_format):  with open('Dataset\_SPMF\_Format.txt', 'w') as file:  file.write('')  with open('Dataset\_SPMF\_Format.txt', 'a') as file:  for i, r in enumerate(spmf\_format):  file.writelines(r)  if i + 1 != len(spmf\_format):  file.writelines('\n')  def get\_df\_grouped\_lookup(self):  if self.df\_grouped\_lookup is None:  self.df\_grouped\_lookup = self.group\_by()  return self.df\_grouped\_lookup |

### *Khai phá dữ liệu (Exploratory Data Analysis – EDA)*

#### Khai phá dữ liệu

##### Tổng quan về dữ liệu

Sau khi tìm hiểu và truy vấn thực hiện lấy dữ liệu từ DB, nhóm được Dataset1. Dataset1 có các cột SalesOrderID, ProductID, Name, UnitPrice, UnitPriceDiscount, ProductCost và ModifiedDate. Trong đó, chi phí của sản phẩm (ProductCost) có hai loại, một là được nhập từ vendor về bán, hai là tự sản xuất để bán. Đồng thời, chi phí (kể cả tự sản xuất hay mua từ nhà cung cấp) có thể khác nhau theo thời gian. Dựa vào thông tin này nhóm sẽ xem xét mã sản phẩm (ProductID) đó thuộc loại nào và tìm bảng chứa chi phí tương ứng để lấy thông tin (nếu ProductID không có trong dữ liệu nhập từ vendor về bán thì nó sẽ nằm trong tự sản xuất rồi bán). Sau đó, dò thời gian xác định đơn hàng (ModifiedDate) tương ứng với thời gian xác định chi phí. Bằng lập luận này nhóm có Dataset1.

Nhóm viết câu lệnh để kiểm tra dataset có “missing value” hay không và nếu có thì “missing value” là bao nhiêu.

|  |
| --- |
| print("Null values: ",df1.isnull().values.sum())  print("NA values: ",df1.isna().values.any()) |

Sau khi thực hiện câu lệnh trên ta có kết quả là Dataset1 có “missing value” và hiện có 49599 giá trị rỗng.

A black background with white text

Description automatically generated

Hình . Kiểm tra dữ liệu null của Dataset1

Nhóm tiếp tục kiểm tra xem những cột nào có giá trị rỗng với câu lệnh:

|  |
| --- |
| print(df1.info()) |

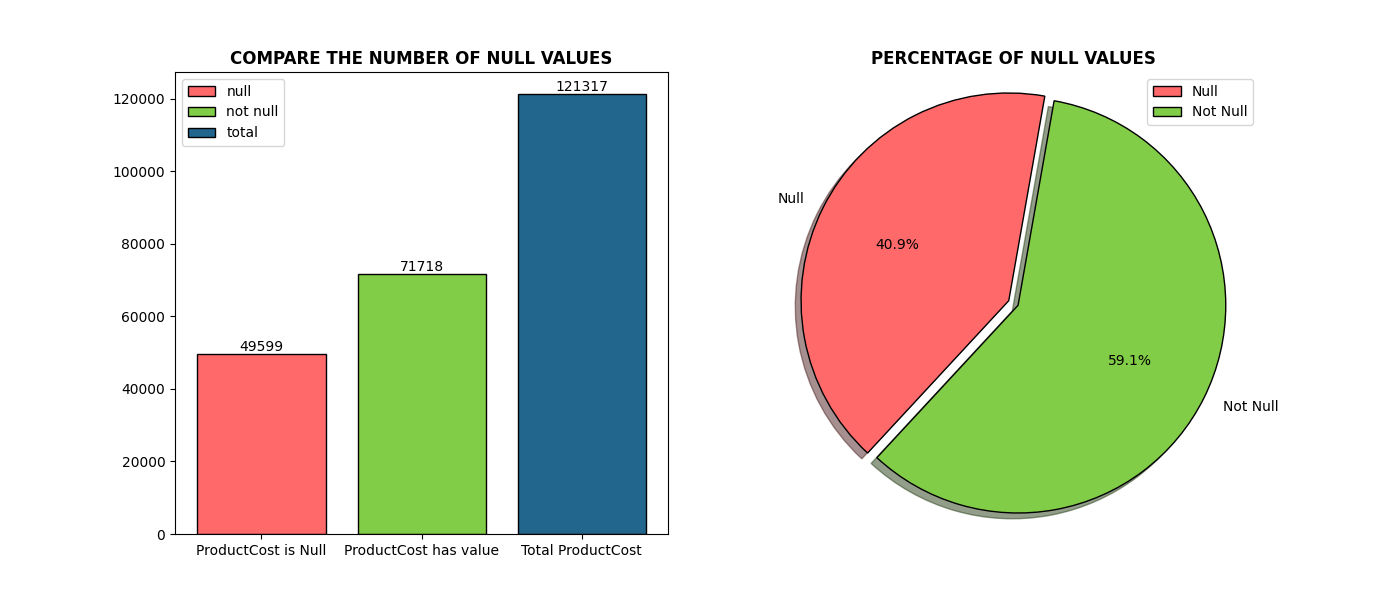
Bằng câu lệnh trên, chúng ta có thể thấy dataset có 121317 dòng dữ liệu, và trong tất cả các cột đều có đủ dữ liệu ngoại trừ cột ProductCost (71718 dòng).

A screenshot of a computer

Description automatically generated

Hình . Kiểm tra cột chứa giá trị null của Dataset1

Nhóm thực hiện trực quan hoá dữ liệu với mục đích có cái nhìn tổng quan về số lượng và tỷ trọng của giá trị null so với giá trị không null của cột ProductCost. Cụ thể giá trị null có số lượng là 49599, chiếm tỉ lệ 40.9%. Đây là một tỉ lệ rất lớn, nên việc xóa bỏ “missing value” gần như không khả thi, trong trường hợp này, phương án “replace” - thay thế missing value được nhóm chọn để làm đầy dữ liệu.



Hình . Biểu đồ cột và tròn thể hiện số lượng và tỷ trọng giá trị null

##### Thay thế các dữ liệu rỗng

Sau khi phân tích và xem xét lại DB, nhóm phát hiện tại bảng ProductCostHistory có ghi lại lịch sử chi phí của sản phẩm. Vì vậy nếu hai phân loại giá sản phẩm (một là được nhập từ vendor về bán, hai là tự sản xuất để bán) được trình bày ở phần trước (tổng quan về dữ liệu) không thỏa thì sẽ lấy chi phí sản phẩm (ProductCost) tại bảng này. Nhóm cho rằng độ tin cậy khi sử dụng dữ liệu của DB sẽ cao hơn là dùng các phương pháp thống kê thông thường (trung bình, trung vị, …) để thay thế các “missing value”. Từ đó, với lập luận này nhóm được Dataset2.

Nhóm thực hiện câu lệnh tương tự với phần trước để kiểm tra giá trị null của Dataset2 và được kết quả có 33 giá trị null.

A black background with white text

Description automatically generated

Hình . Kiểm tra dữ liệu null của Dataset2

Tiếp tục, kiểm tra xem những cột nào có giá trị rỗng và theo như kết quả trên màn hình nhóm thấy cột ProductCost vẫn bị thiếu 33 dòng dữ liệu.

A screenshot of a computer screen

Description automatically generated

Hình . Kiểm tra cột chứa giá trị null của Dataset2

Sau khi xem xét lại DB thì nhóm nhận ra 33 giá trị missing này đều có đặc điểm là không có chi phí niêm yết tính đến thời điểm hiện tại (hoặc tương lai). Tức là chi phí của sản phẩm đó chỉ tính đến thời điểm niêm yết cuối cùng mà nó được ghi nhận, còn sau khoảng thời gian đó thì nó không được xác định trong bảng ProductCostHistory. Sau khi tìm hiểu thì nhóm đặt ra giả định là do các sản phẩm đó đã ngừng sản xuất hoặc ngừng nhập hàng và tại thời điểm đó nghiệp vụ kế toán đã phát sinh lỗi nên không ghi nhận lịch sử chi phí của sản phẩm trong khoảng thời gian sau đó. Vì thuật toán nhóm chọn cần tìm ra những itemsets đi cùng nhau mang lại hiệu quả dựa vào lợi ích (utility) và khoảng thời gian (period time) của chúng. Nên nhóm đề ra phương án là sẽ tiếp tục tìm dữ liệu trong DB phù hợp để thay thế thay vì dùng các phương pháp thống kê thông thường.

##### Hoàn thiện tập dữ liệu

Sau khi xem xét DB, nhóm sử dụng chi phí được lưu trong bảng Product – bảng thông tin của sản phẩm để thay thế cho 33 giá trị bị missing ở phần trên. Lý do vì sao ngay từ đầu nhóm không dùng thông tin này cho chi phí, là vì chi phí được lưu trong bảng Product là chi phí cuối cùng được ghi nhận của sản phẩm đó. Trong khi đó, theo thời gian thì các sản phẩm có chi phí mua/sản xuất khác nhau nên nếu dùng chi phí trong bảng Product sẽ không đánh giá đúng được lợi ích (utility) thực sự mà sản phẩm mang về - làm giảm độ tin cậy. Dựa vào các phương án đã nêu và lập luận trên ta có Dataset3.

Dùng câu lệnh tương tự các phần trước, nhóm nhận thấy các cột đã đồng nhất về số lượng và không còn giá trị null nữa.

A black background with white text

Description automatically generated

Hình . Kiểm tra giá trị null của Dataset3

A screenshot of a computer screen

Description automatically generated

Hình . Kiểm tra cột chứa giá trị null của Dataset3

#### Xử lý và phân tích dữ liệu

Xét về kiểu dữ liệu, nhóm thấy cột ModifiedDate chưa đúng định dạng nên nhóm sẽ chuyển đổi định dạng của nó về datetime. Đồng thời cột Name không dùng đến trong việc triển khai thuật toán nên nhóm sẽ tạm bỏ nó ra.

|  |
| --- |
| def process\_data(self):  df = pd.read\_csv(self.file\_path)  df\_eda = df.copy()  df\_eda.drop('Name', axis=1, inplace=True)  df\_eda['ModifiedDate'] = pd.to\_datetime(df\_eda.ModifiedDate)  print(df\_eda.info())  return df\_eda |

Sau khi thực hiện, ta được kết quả như hình bên dưới với tập dữ liệu có 121317 dòng dữ liệu và không có giá trị null nào.

A screenshot of a computer

Description automatically generated

Hình . Kiểm tra các cột dữ liệu sau khi chỉnh sửa kiểu dữ liệu ModifiedDate và loại bỏ cột Name

Thực hiện kiểm tra quãng thời gian của tập dữ liệu với các câu lệnh sau:

|  |
| --- |
| def time\_period(self):  print("Starting date: ", self.df\_eda.iloc[0, -1])  print("Ending date: ", self.df\_eda.iloc[-1, -1])  print("Duration: ", (self.df\_eda.iloc[-1, -1]) - (self.df\_eda.iloc[0, -1]))  print("Actual Number of Date: " + str(self.df\_eda['ModifiedDate'].nunique())) |

Ta nhận thấy khoảng thời gian của tập dữ liệu bắt đầu vào ngày 31/05/2011 và kết thúc vào ngày 30/06/2014, tổng thời gian là 1126 ngày, trong đó số ngày hoàn chỉnh là 1124 ngày.

A black background with white numbers

Description automatically generated

Hình . Kiểm tra khoảng thời gian của tập dữ liệu

Để mô tả nhanh và sơ lược về dataset nhóm thực hiện câu lệnh đơn giản sau:

|  |
| --- |
| def describe\_dataset(self):  print(self.df\_eda.describe()) |

Sau khi thực hiện, ta có một bảng mô tả sơ lược về dataset, trong đó có các chỉ số trung vị, tứ phân vị, độ lệch chuẩn của các cột trong Dataset.

A screenshot of a computer screen

Description automatically generated

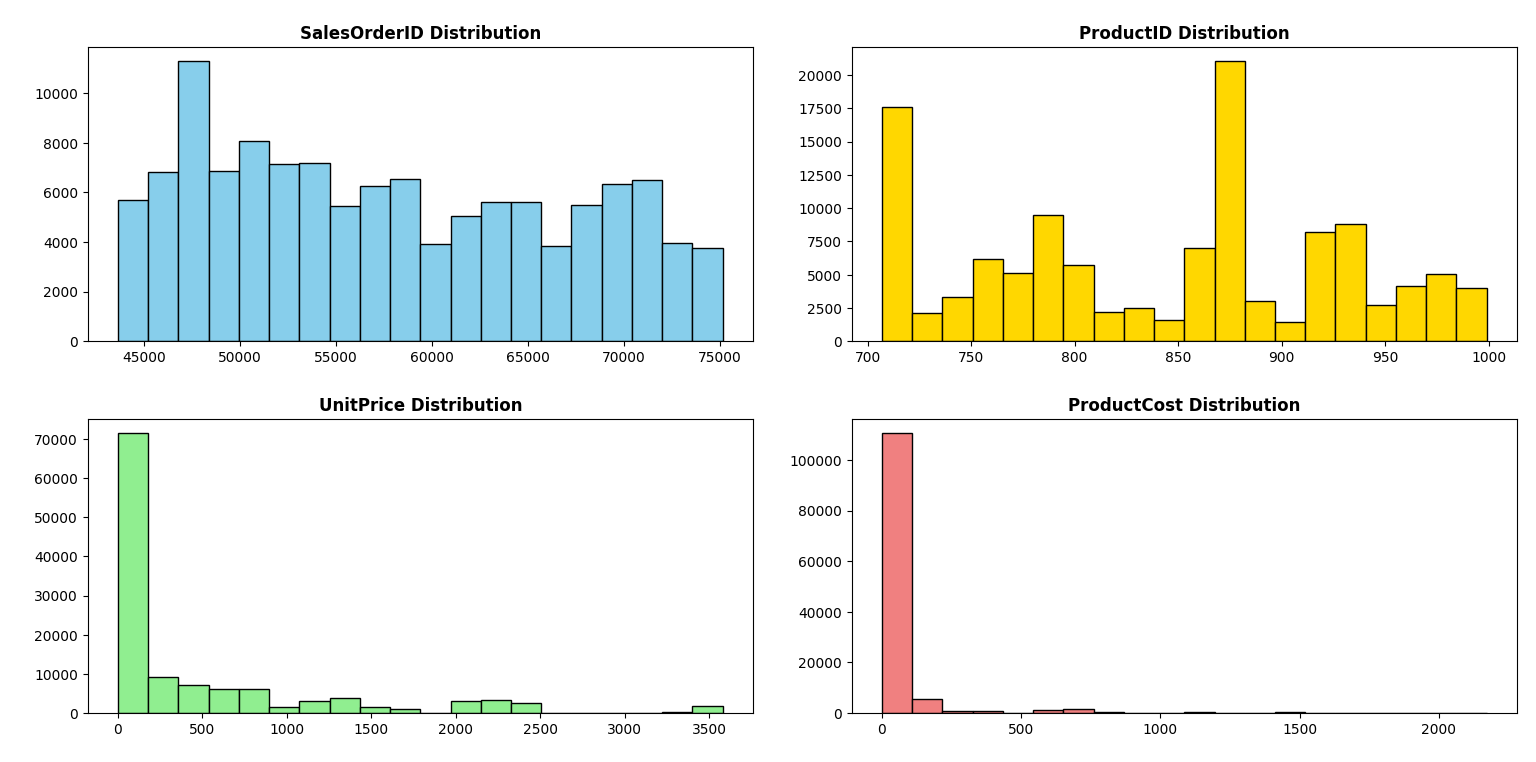
Hình .Mô tả về dataset

A black screen with white text

Description automatically generated

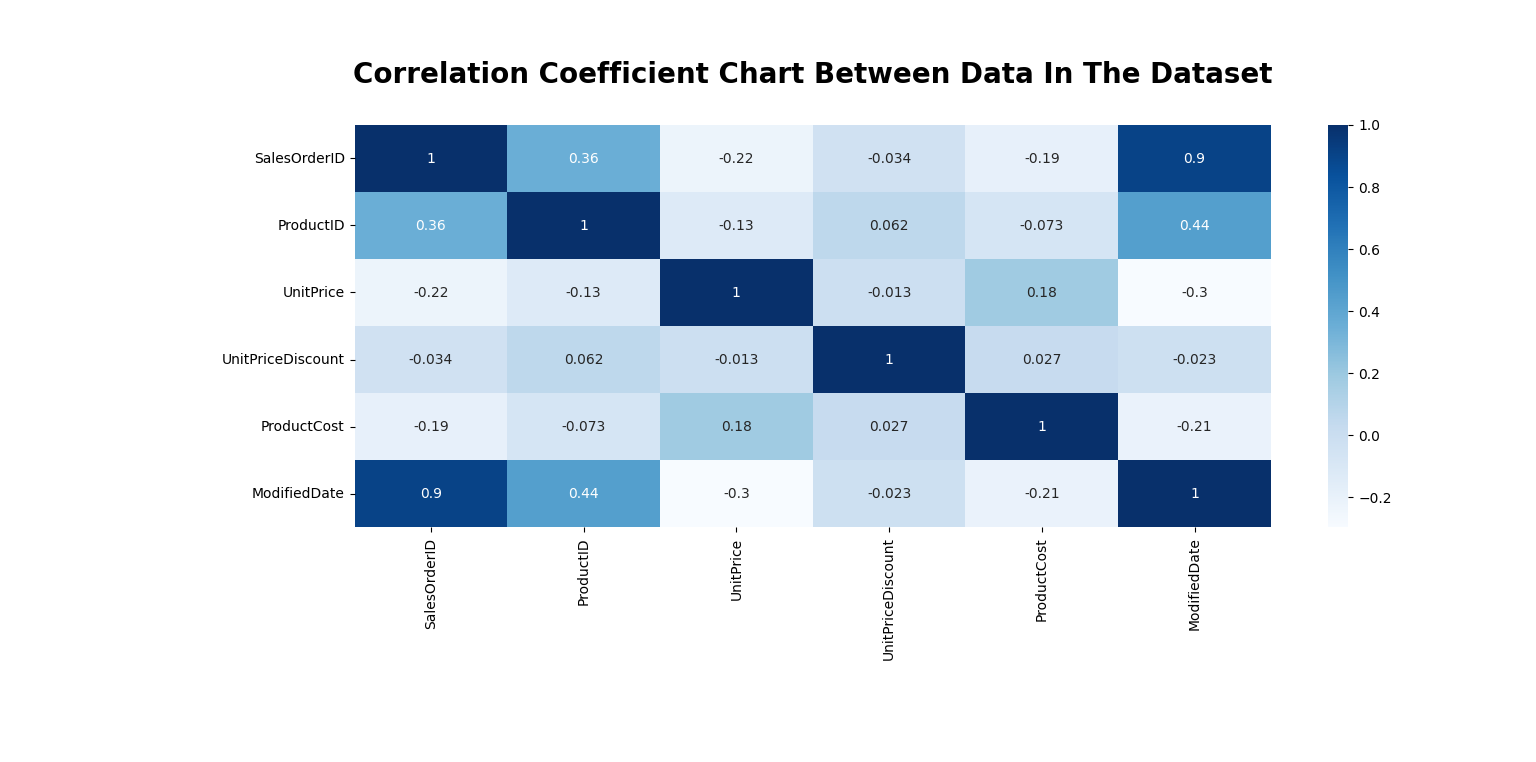
Hình . Mô tả về dataset (tt)

Nhóm tiến hành mô hình hoá tần suất hay sự phân phối dữ liệu của các cột trong tập dữ liệu. Đầu tiên, với mã đơn hàng (SalesOrderID) cho thấy sự phân bố khá đều về số lượng xuất hiện của chúng (mã đơn hàng xuất hiện càng nhiều thì số lượng sản phẩm trong mã đơn hàng đó càng nhiều). Trong đó, có thể thấy các mã đơn hàng ở đầu, đặc biệt là từ khoảng 46000 – 48000 xuất hiện nhiều hơn. Bảng sản phẩm (ProductID) thì lại có sự cao thấp không đồng đều, trong đó, tập trung nhiều ở các sản phẩm đầu và giữa. Bảng giá bán sản phẩm (UnitPrice) và bảng chi phí sản phẩm (ProductCost) khá tương đồng nhau khi có sự chênh lệch lớn vì giá bán và chi phí hầu hết tập trung ở phân khúc giá rẻ (từ khoảng 180 đô trở xuống với giá bán và khoảng 100 đô trở xuống với chi phí), còn lại thì không đáng kể và giảm dần về số lượng khi giá bán hay chi phí tăng cao.



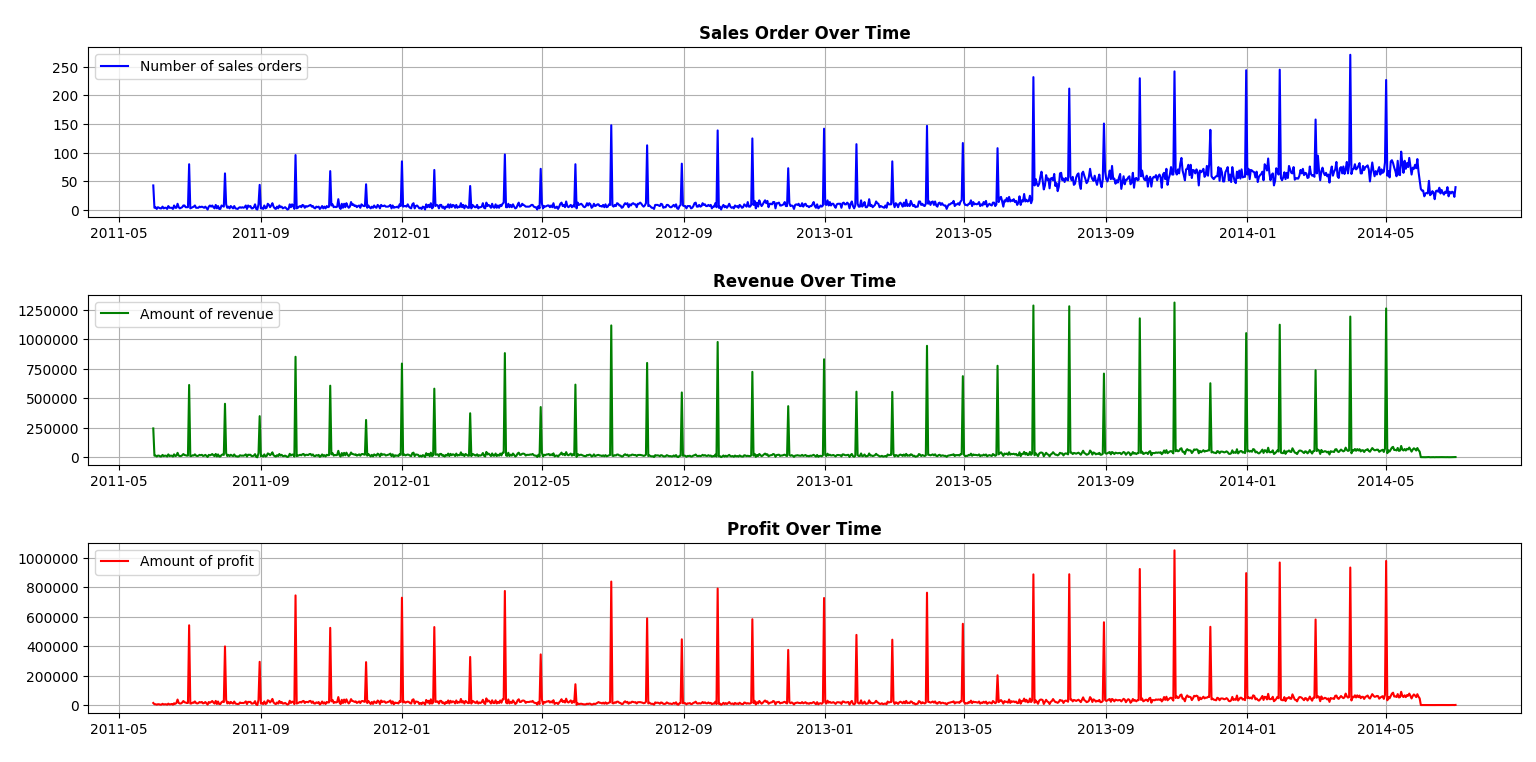
Hình . Biểu đồ miêu tả sự phân phối của các chỉ số

Nhóm thực hiện trực quan hoá biểu đồ hệ số tương quan giữa các cột dữ liệu trong tập dữ liệu. Ta có thể thấy mã đơn hàng (SalesOrderID) và mã sản phẩm (ProductID) có sự tương quan cao với ngày xác nhận đơn hàng (ModifiedDate), điều này có thể ngụ ý rằng có một sự tương quan mạnh giữa các đơn hàng hay sản phẩm cụ thể và thời gian. Có thể tồn tại một xu hướng cụ thể trong cách đơn hàng được đặt hàng và sản phẩm được lựa chọn theo thời gian. Đối với giảm giá đơn giá (UnitPriceDiscount) và chi phí sản phẩm (ProductCost) thì chúng có xu hướng đồng biến và có hệ số tương quan cao hơn so với giá bán sản phẩm (UnitPrice) có xu hướng nghịch biến với hệ số tương quan âm. Có thể nhận định rằng chi phí sản phẩm có sự ảnh hưởng đến việc quyết định mức giảm giá đơn giá cao hơn so với giá bán sản phẩm.



Hình . Biểu đồ hệ số tương quan giữa các dữ liệu

Nhóm thực hiện trực quan hóa các biểu đồ đường về Sales Order, Revenue và Profit qua thời gian. Trong khoảng thời gian từ tháng 6/2011 đến tháng 5/2014, cả 3 biểu đồ đều khá tương đồng nhau khi có xu hướng tăng mạnh vào khoảng thời gian giao nhau giữa đầu và cuối tháng, sau đó giảm mạnh và ổn định cho đến chu kỳ tăng mạnh ở khoảng thời gian đầu và cuối tháng tiếp theo. Đồng thời, xu hướng tăng giảm của cả 3 biểu đồ đều khá ổn định, trong đó, biểu đồ Sales Order là có sự tăng trưởng mạnh vào cuối tháng 6/2013. Đến tháng 6/2014, cả 3 biểu đồ đều có sự suy giảm sâu thay vì tăng mạnh như các chu kỳ trước đó. Điều này cho thấy, đã có biến cố hay sự kiện nào đó đã diễn ra và làm ảnh hưởng lớn đến việc kinh doanh (cụ thể là gây ra suy giảm mạnh).



Hình . Biểu đồ đường về các chỉ số Sales Order, Revenue, Profit qua thời gian

### *Phân tích tập dữ liệu đầu vào*

Định dạng đầu vào của thuật toán là một tập tin có định dạng “.txt”, mỗi dòng đại diện cho một giao dịch, mỗi dòng bao gồm các thông tin:

* Các mục trong cơ sở dữ liệu giao dịch (Items): các mã sản phẩm được bán.
* Sau kí hiệu “:” là tiện ích giao dịch: tổng lợi nhuận dương của các sản phẩm đem lại trong một giao dịch.
* Sau kí hiệu “:” là tiện ích của từng mục trong giao dịch (Item utilities): lợi nhuận của từng sản phẩm.
* Khoảng thời gian khi giao dịch xảy ra (Time period): theo thứ tự 0, 1, 2, 3 đại diện cho các khoảng thời gian “xuân”, “hạ”, “thu”, “đông”.

Dưới đây là một vài dòng trong tập dữ liệu đầu vào theo định dạng SPMF mà nhóm đã thực hiện:

|  |
| --- |
| 716 709 777 712 711 714 778 771 773 772 774 776:903:-3 2 127 -1 8 -3 127 128 128 128 128 127:1  758 762:6:-10 6:1  712 741 715 708 716 775 742 745 743 777 776 711 747 773 778:1168:-1 112 -3 8 -3 127 99 111 98 127 127 8 98 128 127:1  729 732 760 726 733 753 725 722 756 754 749 762 758 755 765 730 766 763 764 738 770 768:75:2 5 6 2 5 -24 2 2 -10 -10 -24 6 -10 -10 6 2 6 6 6 2 6 6:1  760:6:6:1  714 777 771 775 773 716 772 778:764:-3 127 128 127 128 -3 128 127:1  709 777 773 715 775 707 776 711 778 712:654:2 127 128 -3 127 8 127 8 127 -1:1  753 756 768 766 732 764:24:-24 -10 6 6 5 6:1  778 773 775 710:384:127 128 127 2:1  707 715 716 725 711 768 708 754 712 722 726 764 756 729 730 770 714 738 732 760 733 761 758 766 753 763 762 765 755:105:8 -3 -3 2 8 6 8 -10 -1 2 2 6 -10 2 2 6 -3 2 5 6 5 6 -10 6 -24 6 6 6 -10:1  747:98:98:1 |

Theo như dòng đầu tiên, giao dịch {716, 709, 777, 712, 711, 714, 778, 771, 773, 772, 774, 776} có tổng lợi nhuận dương là 903$ và lợi nhuận cụ thể của từng mã sản phẩm là -3$, 2$, 127$, -1$, 8$, -3$, 127$, 128$, 128$, 128$, 128$, 127$, các sản phẩm này được mua vào mùa hạ.

## **Kết quả (đầu ra)**

Sau khi áp dụng thuật toán FOSHU vào tập dữ liệu theo câu lệnh sau:

|  |
| --- |
| from spmf import Spmf  import pandas as pd  pd.set\_option('display.max\_columns', None)  class FOSHU\_SPMF:  def \_\_init\_\_(self):  self.df\_result = None  def Run\_SPMF(self,name\_input, name\_output, arg):  spmf = Spmf("FOSHU", input\_filename=name\_input,  output\_filename=name\_output, arguments=[arg])  spmf.run()  return spmf  def get\_df\_result(self):  if self.df\_result is None:  spmf = self.Run\_SPMF('Dataset\_SPMF\_Format.txt','output.txt',0.03)  self.df\_result = spmf.to\_pandas\_dataframe\_for\_FOSHU(pickle=True)  return self.df\_result |

Đầu ra của thuật toán FOSHU là tập hợp các mục tiện ích cao có sẵn, có tiện ích tương đối không nhỏ hơn tỷ lệ tiện ích tối thiểu (min\_uitility\_ratio). Định dạng đầu ra của thuật toán FOSHU cũng có định dạng “.txt”, mỗi dòng đại diện cho một tập hữu ích cao có sẵn. Trên mỗi dòng có các thông tin như sau:

* Đầu tiên là các mục của tập mục (Itemsets): mã sản phẩm (ProductID).
* Sau từ “ #UTIL” là tiện ích của tập mục (Utility): lợi nhuận của các sản phẩm.
* Sau từ “#RUTIL” là tiện ích tương đối của tập mục (Relative utility): tỷ lệ lợi nhuận được tạo ra bởi một tập hợp mặt hàng nhất định trong khoảng thời gian bán ra.

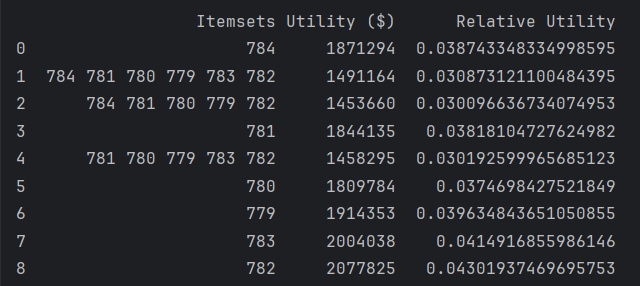
Bên cạnh đó, do hàm “to\_pandas\_dataframe” bên trong thư viện spmf chỉ hỗ trợ cho thuật toán có tham số “sup” (support) nên không thể sử dụng hàm mặc định của spmf để hiển thị kết quả và chuyển nó thành dataframe dùng để phân tích về sau. Vì vậy, nhóm đã viết lại một hàm cho thuật toán FOSHU bên trong thư viện spmf:

|  |
| --- |
| def to\_pandas\_dataframe\_for\_FOSHU(self, pickle=False):  if not self.patterns\_:  self.parse\_output()  patterns\_dict\_list = []  for pattern in self.patterns\_:  itemsets = pattern[0].split("#UTIL:")[0].strip()  utility = pattern[0].split("#UTIL:")[1].split("#RUTIL:")[0].strip()  relative\_utility = pattern[0].split("#RUTIL:")[1].strip()  patterns\_dict\_list.append({'Itemsets': itemsets, 'Utility ($)': utility, 'Relative Utility': relative\_utility})  df = pd.DataFrame(patterns\_dict\_list)  self.df\_ = df  if pickle:  df.to\_pickle(self.output\_.replace(".txt", ".pkl"))  return df |

Dưới đây là kết quả đầu ra sau khi áp dụng thuật toán FOSHU với tiện ích tối thiểu (minimum utility) là 0,03:

|  |
| --- |
| 784 #UTIL: 1871294 #RUTIL: 0.038743348334998595  784 781 780 779 783 782 #UTIL: 1491164 #RUTIL: 0.030873121100484395  784 781 780 779 782 #UTIL: 1453660 #RUTIL: 0.030096636734074953  781 #UTIL: 1844135 #RUTIL: 0.03818104727624982  781 780 779 783 782 #UTIL: 1458295 #RUTIL: 0.030192599965685123  780 #UTIL: 1809784 #RUTIL: 0.0374698427521849  779 #UTIL: 1914353 #RUTIL: 0.039634843651050855  783 #UTIL: 2004038 #RUTIL: 0.0414916855986146  782 #UTIL: 2077825 #RUTIL: 0.04301937469695753 |

Theo như dòng đầu tiên, tập mục {784} có tiện ích là 1871294$ và tiện ích tương đối là 0.0387 (không nhỏ hơn tiện ích tối thiểu). Tương tự các dòng còn lại, có thể thấy tiện ích tương đối không nhỏ hơn 0,03 - mức tiện tối thiểu mà nhóm đã đặt ra. Có thể xem nó dưới dạng một dataframe mà nhóm đã chuyển đổi bên dưới:



Hình . Hình kết quả đầu ra dưới dạng dataframe

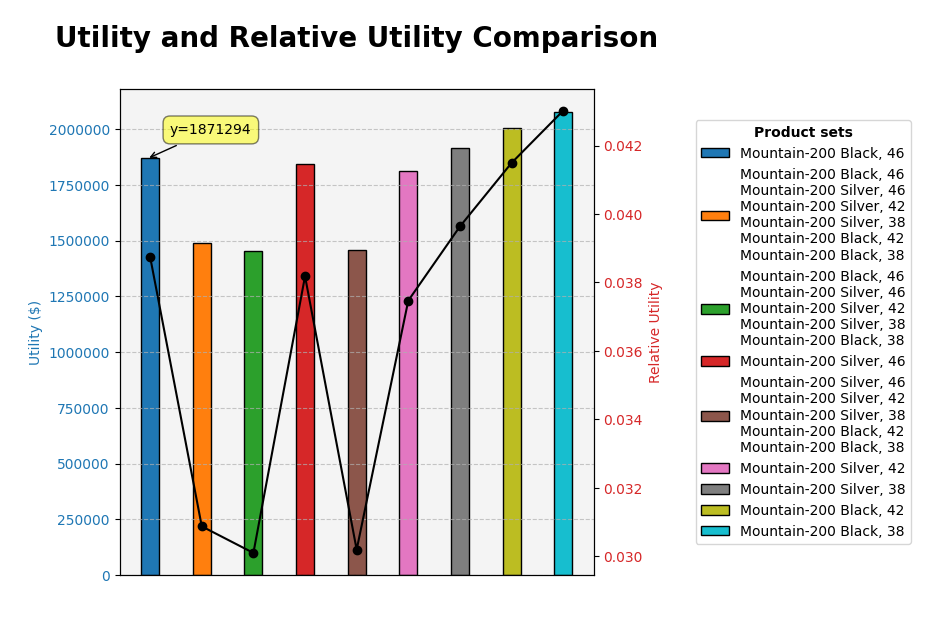
Sau khi thực đã chạy ra kết quả, nhóm nhận thấy cần lấy thêm thông tin cho đầu ra để có thể phân tích kết quả tốt hơn. Vì vậy, nhóm đã thực hiện chuyển đổi, dò tìm và lấy thêm dữ liệu cho kết quả đầu ra:

|  |
| --- |
| from ConvertToSPMF import \* from Run\_FOSHU\_SPMF import \*  pd.set\_option('display.max\_columns', None)  class Convert\_Output\_To\_Model:  def \_\_init\_\_(self,df\_eda,df\_groupby,df\_result):  self.df\_eda = df\_eda  self.df\_groupby = df\_groupby  self.df\_result = df\_result   self.df\_result\_splited = None  self.df\_looked\_for = None   self.unitprice = 0  self.discount = 0  self.cost = 0   def split\_itemsets\_into\_list(self):  df\_result\_split = []  for item in self.df\_result['Itemsets']:  df\_result\_split.append({'Itemsets': item.split()})  self.df\_result\_splited = pd.DataFrame(df\_result\_split)  return self.df\_result\_splited   def get\_name(self):  df\_dataset = pd.read\_csv('Dataset3.csv')  df\_result = self.df\_result   df\_new = pd.DataFrame(columns=['ProductName', 'Utility ($)', 'Relative Utility'])   *# Tách chuỗi Itemsets thành các mã sản phẩm* for index, row in df\_result.iterrows():  itemsets = row['Itemsets'].split()  product\_names = []  for item in itemsets:  item = int(item)  *# Kiểm tra sự trùng khớp với cột ProductID của df\_dataset* match = df\_dataset[df\_dataset['ProductID'] == item]  if not match.empty:  product\_name = match['Name'].values[0]  product\_names.append(product\_name)   *# Tạo một dòng mới cho df\_new* if product\_names:  df\_new = pd.concat([df\_new, pd.DataFrame({'ProductName': ['\n'.join(product\_names)],  'Utility ($)': [row['Utility ($)']],  'Relative Utility': [row['Relative Utility']]})])   *# return df\_new* df\_new.to\_csv(r'D:\Nam4\_HKI\NhaKho\_DuLieu\Midterm\_Project\TrucQuanKQ\_Improve.csv', index=False)   def look\_for\_output\_in\_dataset(self):  df\_check = []  df\_result\_splited = self.split\_itemsets\_into\_list()  df\_groupby = self.df\_groupby.copy()  mapping = {0: 'Spring', 1: 'Summer', 2: 'Fall', 3: 'Winter'}  df\_groupby['Season'] = df\_groupby['Season'].replace(mapping)  for i, r in df\_groupby.iterrows():  salesorderid = r['SalesOrderID']  itemsets = r['ProductID']  season = r['Season']   for row in df\_result\_splited['Itemsets']:  sodem = 0  profit = 0  for j in range(0, len(row), 1):  self.unitprice = 0  self.discount = 0  self.cost = 0  if row[j] in itemsets:  sodem += 1  self.unitprice = self.df\_eda.loc[(self.df\_eda['SalesOrderID'] == salesorderid) & (  self.df\_eda['ProductID'] == int(row[j])), 'UnitPrice'].values[0]  self.discount = self.df\_eda.loc[(self.df\_eda['SalesOrderID'] == salesorderid) & (  self.df\_eda['ProductID'] == int(row[j])), 'UnitPriceDiscount'].values[0]  self.cost = self.df\_eda.loc[(self.df\_eda['SalesOrderID'] == salesorderid) & (  self.df\_eda['ProductID'] == int(row[j])), 'ProductCost'].values[0]  profit += (self.unitprice \* (1 - self.discount) - self.cost)  continue  else:  break  if sodem == len(row):  df\_check.append(  {'SalesOrderID': salesorderid, 'Itemsets': row, 'Profit': int(round(profit, 0)), 'Season': season})   self.df\_looked\_for = pd.DataFrame(df\_check)  return self.df\_looked\_for   def Convert\_df\_for\_AoV(self):  df\_looked\_for = self.look\_for\_output\_in\_dataset()   copy = df\_looked\_for[['Itemsets', 'Profit', 'Season']].copy()  copy['Itemsets'] = copy['Itemsets'].apply(lambda x: ' '.join(x))  copy['Count'] = copy['Itemsets']   df\_pt1 = copy.groupby(['Itemsets', 'Season']).agg({  'Profit': 'sum',  'Count': 'size'  }).reset\_index()  df\_pt1.columns = ['Itemsets', 'Season', 'Total Profit', 'Count']   print(df\_pt1)  df\_pt1.to\_csv(r'D:\Nam4\_HKI\NhaKho\_DuLieu\Midterm\_Project\TrucQuanKQ1.csv', index=False)  print('Xong rồi')  return df\_pt1 |

## **Phân tích kết quả**

### Trực quan hóa và phân tích kết quả

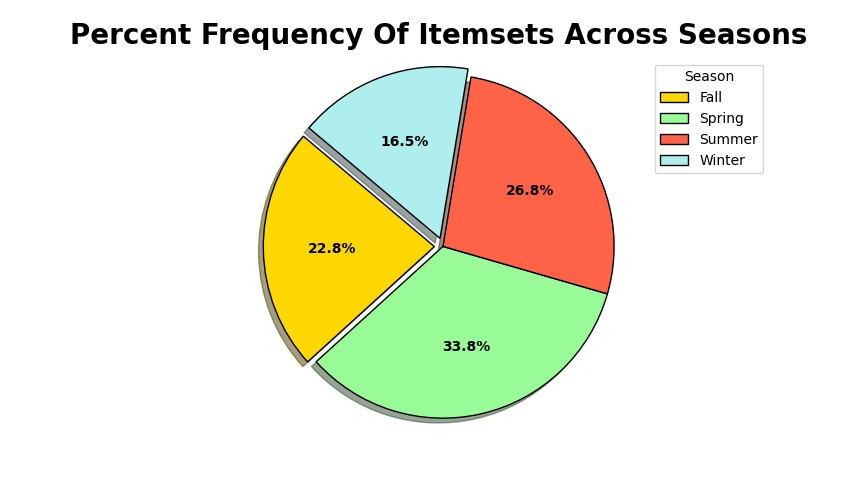
*So sánh tiện ích và tiện ích tương đối của các Itemsets:*



Hình . Biểu đồ so sánh tiện ích (utility – lợi nhuận) và tiện ích tương đối (relative utility – tỷ lệ lợi nhuận) của các tập mục (sản phẩm) đi cùng nhau

Qua biểu đồ trên, ta có thể thấy sản phẩm “Mountain-200 Black, 38” mang lại lợi nhuận và có tỷ lệ lợi nhuận cao nhất. Đồng thời các tập mục với sản phẩm đơn lẻ như “Mountain-200 Black, 42”, “Mountain-200 Silver, 38”, “Mountain-200 Silver, 42”, “Mountain-200 Silver, 46” và “Mountain-200 Black, 46” đều mang lại lợi nhuận với tỷ lệ lợi nhuận tương đối cao. Cho thấy các tập mục sản phẩm có tỷ lệ lợi nhuận cao thì thường có tổng lợi nhuận cao tương ứng. Đối với các tập mục chứa sản phẩm đơn lẻ, tổng lợi nhuận và tỷ lệ lợi nhuận tương quan theo cùng một hướng (cao thì cả hai đều cao, thấp thì cả hai đều thấp). Điều này cho thấy các sản phẩm trong tập mục đã đem lại lợi nhuận đáng kể và hợp lý. Tuy nhiên, đối với các tập mục chứa nhiều sản phẩm đi cùng nhau, mặc dù tổng lợi nhuận mang lại khá tốt, nhưng tỷ lệ lợi nhuận lại rất thấp. Có thể đưa ra nhận định rằng khi bán các tập mục sản phẩm đi cùng nhau, có thể cần áp dụng các chính sách giảm giá hoặc chiết khấu lớn hơn để khuyến khích người mua mua nhiều sản phẩm cùng một lúc. Điều này có thể làm giảm đi tỷ lệ lợi nhuận của chúng.

*Tỷ lệ phần trăm tần suất xuất hiện của toàn bộ Itemsets qua 4 mùa:*

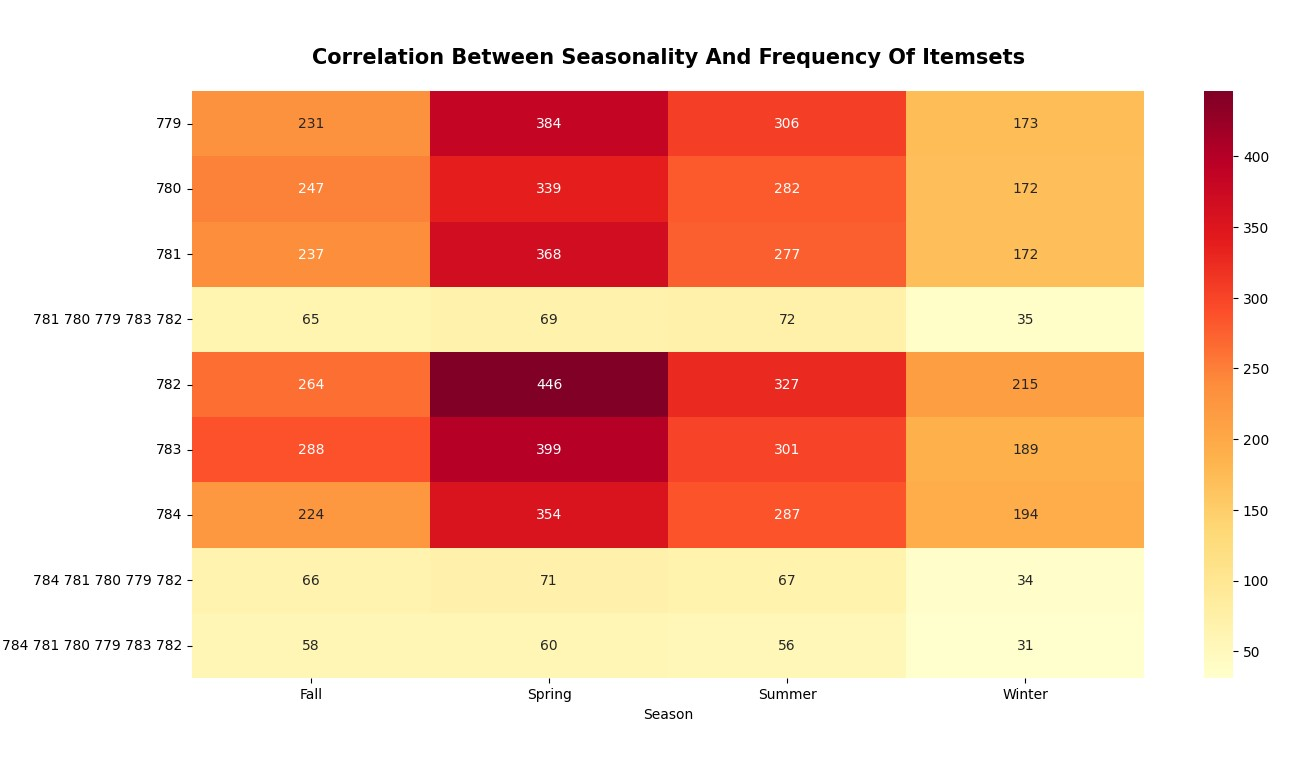


Hình . Biểu đồ tròn thể hiện tỷ lệ phần trăm tần suất xuất hiện của toàn bộ Itemsets qua 4 mùa

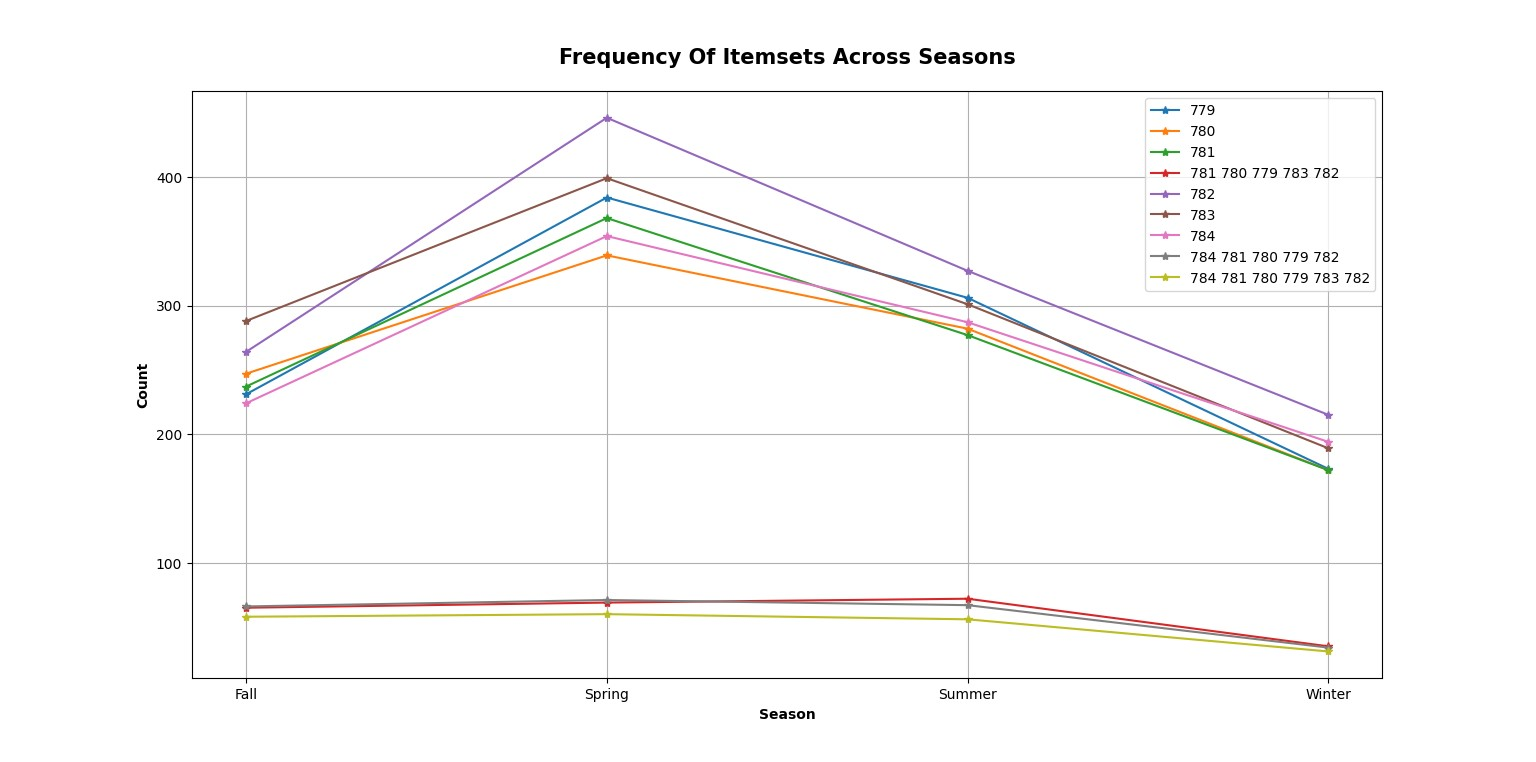
Biểu đồ tròn trên thể hiện sự tương quan giữa các Itemsets trong 4 mùa với tỷ lệ phần trăm tần suất xuất hiện của chúng trong tập dữ liệu. Mùa xuân là mùa có tỷ lệ xuất hiện của các Itemsets cao nhất với tỷ lệ 33.8% và mùa đông chiếm tỷ lệ thấp nhất với 16.5%.

Để có thể phân tích sâu hơn với nhiều thông tin hơn, nhóm sẽ sử dụng mã sản phẩm thay cho tên của các sản phẩm để có thể thuận tiện cho việc truy xuất lấy thêm dữ liệu cũng như trực quan hoá các biểu đồ hiệu quả hơn.

*Mối quan hệ giữa các mùa và tần suất xuất hiện của các Itemset*



Hình . Biểu đồ thể hiện mối quan hệ giữa tần suất xuất hiện của các Itemsets và 4 mùa trong năm

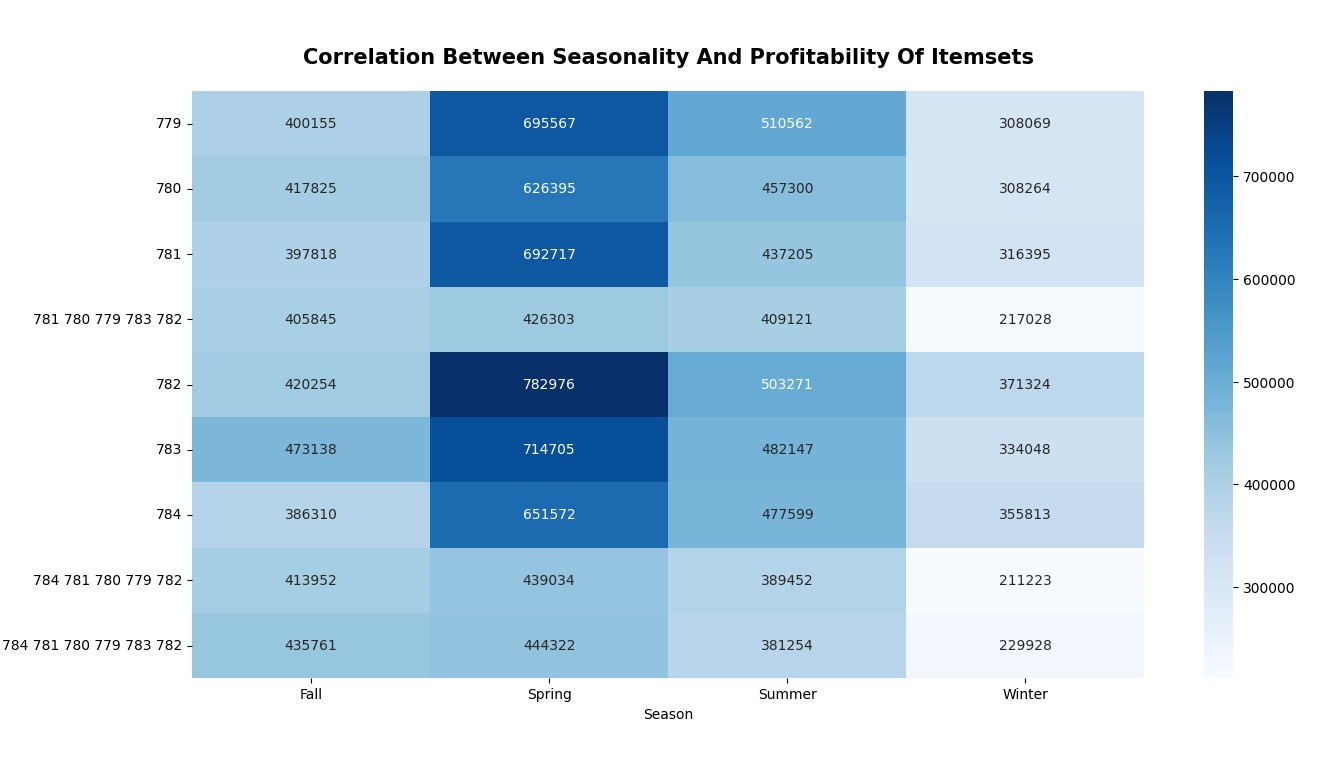


Hình . Biểu đồ thể hiện mối quan hệ giữa tần suất xuất hiện của các Itemsets và 4 mùa trong năm

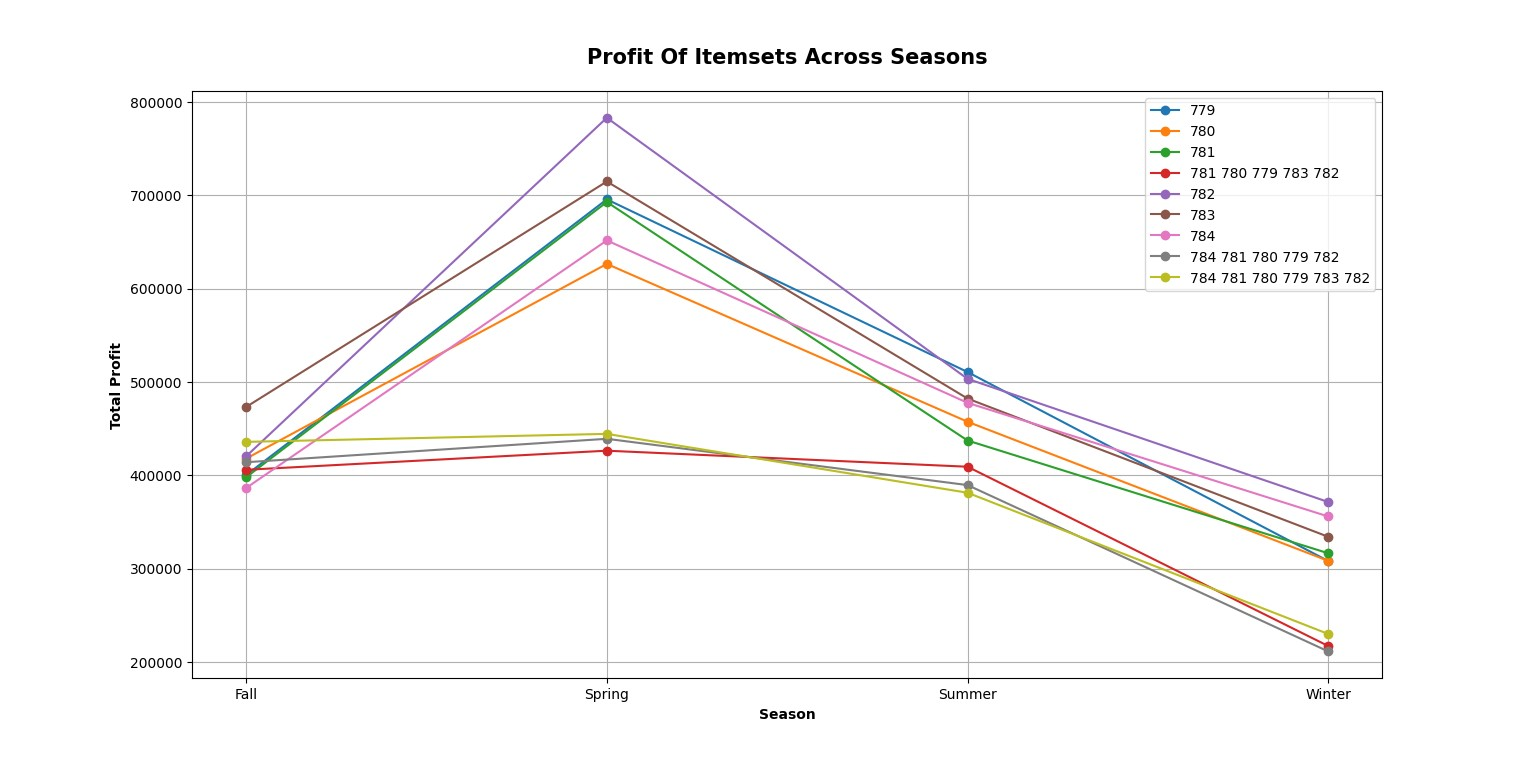
Hai biểu đồ trên cho ta thấy được kết quả số lần xuất hiện của các cụm Itemsets như {779};{780};{ 781}; {781, 780, 779, 783, 782}; {782}; {783}; {784}; {784, 781, 780, 779, 782}; {784, 781, 780, 779, 783, 782} được thể hiện qua rõ bốn mùa.

Cụ thể, hình 20 tần suất xuất hiện của các cụm Itemsets được thể hiện rõ bằng độ đậm nhạt của màu trên biểu đồ, như {782} với tần suất xuất hiện lớn nhất trong các Itemset vào mùa xuân với số lần là 446 ,cụm {784, 781, 780, 779, 783, 782} với số lần xuất hiện thấp nhất trong các cụm Itemsets rơi vào mùa đông với 31 lần xuất hiện . Kết quả nhìn chung ở cả 2 biểu đồ cho thấy tần suất xuất hiện của các cụm Itemsets này phân bố tập trung cao nhất vào mùa xuân, thấp nhất vào mùa đông. Ngoài ra, việc các Itemset đơn lẻ có tần suất xuất hiện nhiều hơn so với các cụm Itemset đi với nhau là điều hiển nhiên, vì trong thực tế để một sản phẩm xuất hiện trong đơn hàng sẽ phổ biến và dễ hơn nhiều so với việc để một cụm các sản phẩm xuất hiện cùng nhau. Nhưng nhìn chung các cụm Itemset trên biểu đồ vẫn cho thấy chúng duy trì được sự ổn định ở các mùa xuân, hạ và thu.

*Mối quan hệ giữa lợi nhuận của các cụm Itemset giữa các mùa:*



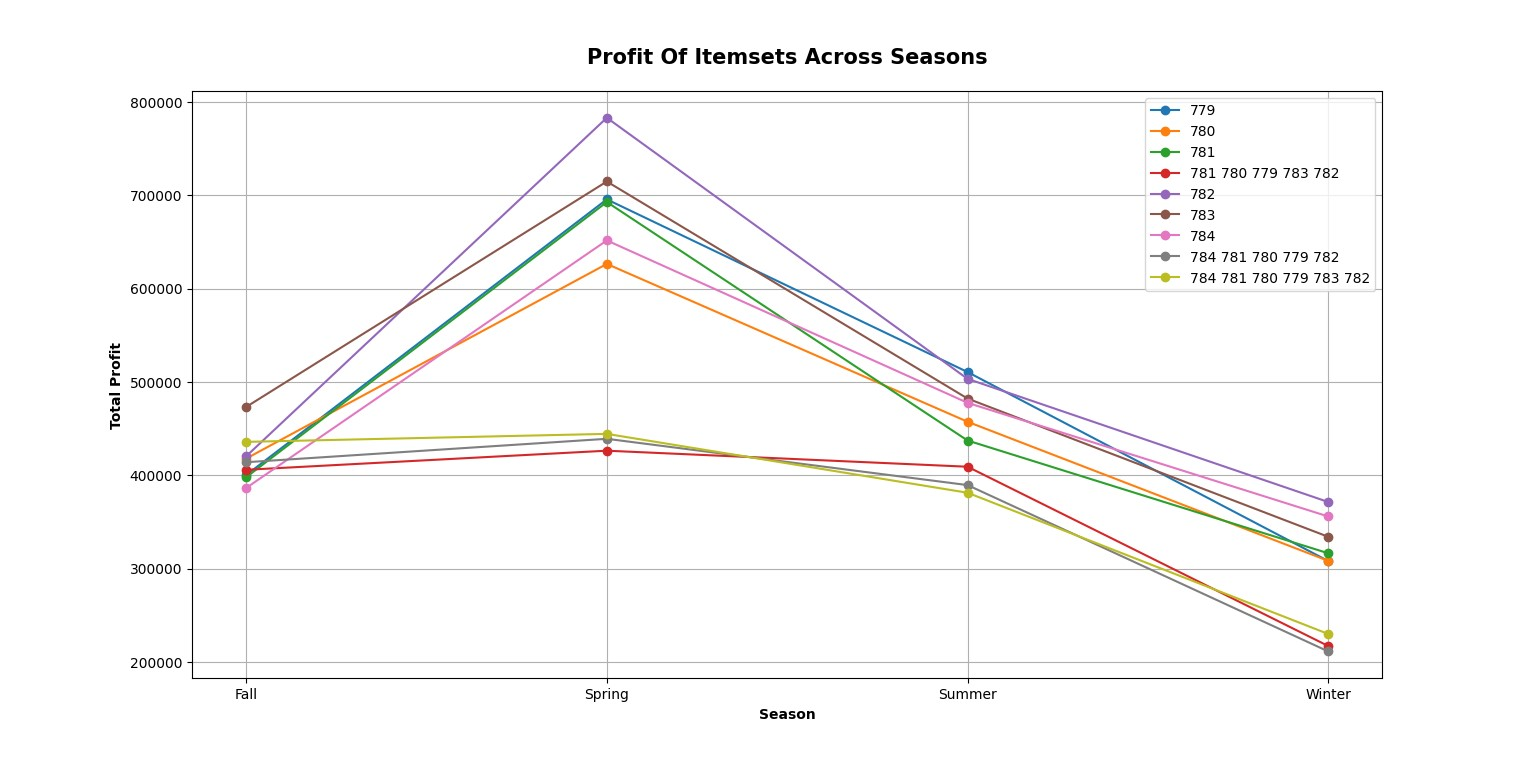
Hình . Biểu đồ thể hiện mối quan hệ giữa tổng lợi nhuận của các Itemsets và 4 mùa trong năm



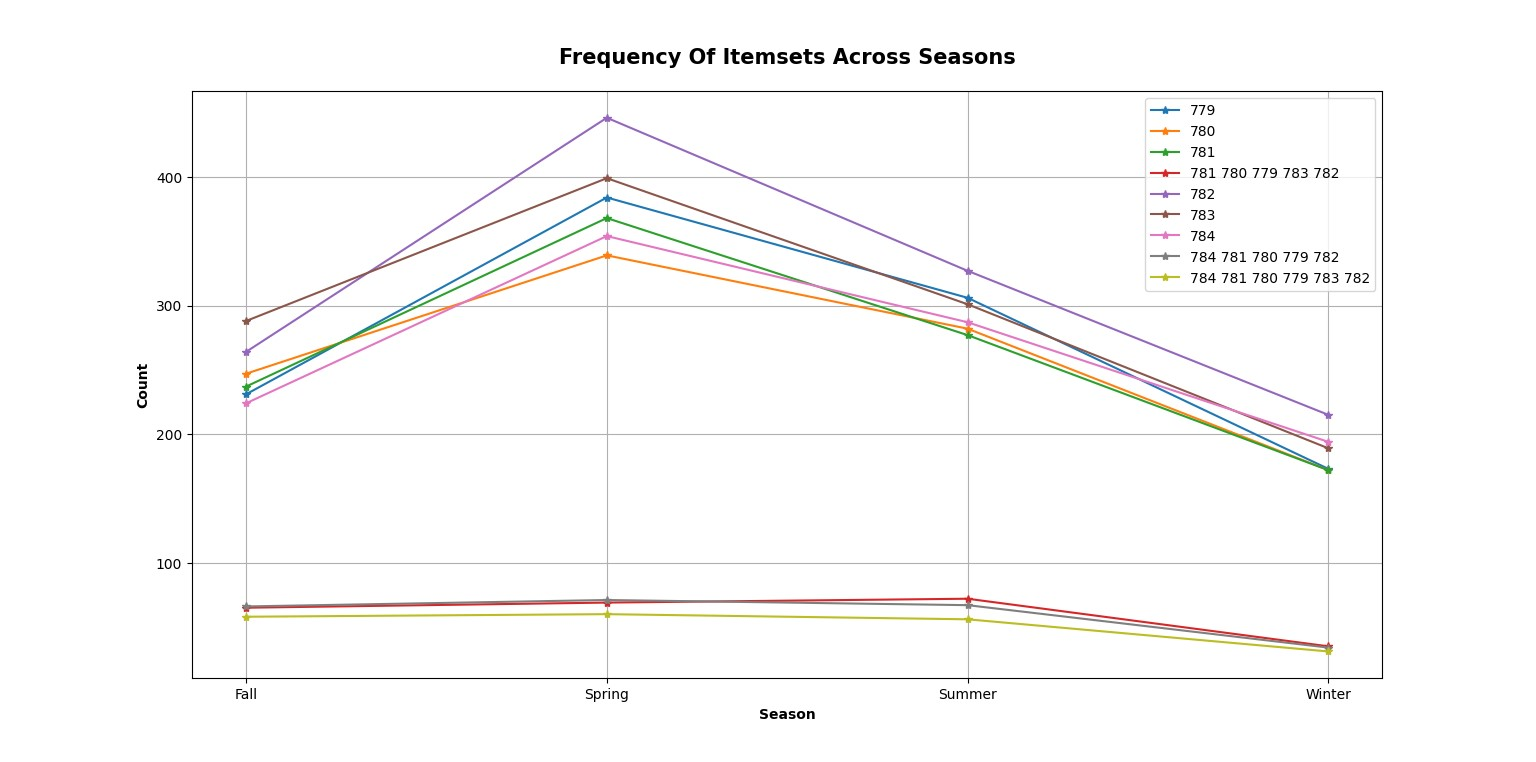
Hình . Biểu đồ đường thể hiện tổng lợi nhuận của các Itemsets trong 4 mùa

Hai biểu đồ trên cung cấp kết quả thông tin về tổng lợi nhuận của các Itemset qua bốn mùa trong năm. Để diễn giải cụ thể, ta sẽ đi vào phân tích cụ thể từng biểu đồ, biểu đồ hình 22, tổng lợi nhuận của Itemset {782} cao nhất trong các Itemset với 782976 ( đơn vị tiền tệ) vào mùa xuân; cụm {781, 780, 779, 783, 782} có tổng lợi nhuận thấp nhất: 217028 ( đơn vị tiền tệ) rơi vào mùa đông. Chúng ta có thể dễ dàng nhận thấy mùa xuân là mùa mang lại tổng lợi nhuận cao nhất của các Itemset, và mùa đông là mùa mang lại lợi nhuận thấp nhất. Điều này cũng dễ hiểu vì tần suất xuất hiện của chúng cũng tập trung nhiều ở mùa xuân và ít ở mùa đông. Bên cạnh sự nổi bật về lợi nhuận lớn thu được ở mùa xuân thì mùa thu là mùa có tính ổn định tốt và sự chênh lệch ít nhất.

*So sánh tần suất xuất hiện và tổng lợi nhuận của các Itemsets.*



Hình . Biểu đồ đường thể hiện tổng lợi nhuận của các Itemsets trong 4 mùa



Hình . Biểu đồ đường thể hiện tần suất xuất hiện của các Itemsets trong 4 mùa

Phân tích kết quả từ việc so sánh 2 biểu đồ trên, ta có thể thấy có sự không tương xứng giữa tổng lợi nhuận của từng Itemset và tần suất xuất hiện của chúng so với các Itemset khác.

Cụ thể, cụm {784, 781, 780, 779, 782} có tần suất xuất hiện vào mùa xuân là 71 cao hơn cụm {784, 781, 780, 779, 783, 782} có tần suất xuất hiện vào mùa xuân là 60 nhưng tổng lợi nhuận của {784, 781, 780, 779, 783, 782} là 444322 (đơn vị tiền tệ) lại cao hơn so với tổng lợi nhuận của {784, 781, 780, 779, 782} với 439039 (đơn vị tiền tệ). Hay cụm {781, 780, 779, 783, 782} với tần suất xuất hiện vào mùa xuân là 69 cao hơn cụm {784, 781, 780, 779, 783, 782} với 60 lần xuất hiện nhưng tổng lợi nhuận của {784, 781, 780, 779, 783, 782} lại cao hơn so với cụm {781, 780, 779, 783, 782} có tổng 426303 (đơn vị tiền tệ). Nếu chỉ xem sơ qua thì có vẻ kỳ lạ nhưng thực tế việc này là hoàn toàn khả thi và hợp lý. Các cụm có tần suất xuất hiện ít hơn nhưng lại có lợi nhuận cao hơn là vì chúng có nhiều tập mục (item) hơn, vậy nên lợi thế về tổng số lợi nhuận của chúng sẽ cao hơn.

Tuy nhiên, đối với cụm {779} và {781}, vào mùa xuân tần suất xuất hiện của {779} nhiều hơn {781} nhưng về lợi nhuận thì cả hai gần như xấp xỉ nhau ({779} với lợi nhuận 695567 và {781} với lợi nhuận là 692717). Điều này cho thấy, tỷ lệ lợi nhuận của {781} vào mùa xuân sẽ cao hơn so với {779}.

Như vậy, giữa các Itemsets có sự không tương xứng nhau về tần suất và lợi nhuận thì cần phải dựa vào đúng tiêu chí, đặt chúng đúng ngữ cảnh thì mới có thể đánh giá và đưa ra quyết định phù hợp.

### Phân tích hiệu suất thuật toán

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Parameters/Years** | **2011 - 2014** | **2011** | **2012** | **2013** | **2014** |
| **Minimum utility ratio** | 0.03 | 0.055 | 0.064 | 0.0674 | 0.03 |
| **Time** | 970 ms | 172 ms | 2394 ms | 10867 ms | 630 ms |
| **Memory** | 47.34 MB | 24.33 MB | 168.34 MB | 40.62 MB | 40.48 MB |
| **HOU count** | 9 | 7 | 9 | 8 | 10 |
| **Join count** | 49223 | 7972 | 624368 | 7254976 | 226018 |

Bảng . So sánh thuật toán FOSHU thực thi trên bộ dữ liệu được phân chia theo từng năm và tổng thể thời gian

Để đánh giá hiệu suất kết quả thực thi thuật toán FOSHU, bộ dữ liệu được phân chia theo từng năm và tổng thể thời gian. Kết quả cho thấy việc lựa chọn khoảng thời gian cũng như độ lớn của dữ liệu ảnh hưởng đến kết quả khai phá.

Thứ nhất, về tỷ lệ tiện ích tối thiểu, khi phân tích từng năm ta có thể khai phá được các tập mục có lợi nhuận cao hơn của những năm đó. Đồng thời nó cũng cho ta biết rõ hơn về tình hình và chiến lược kinh doanh của doanh nghiệp. Ví dụ, nhóm đặt ra “minimum utility ratio” để khai phá được khoảng 7 - 10 Itemsets thì ta có thể nhận thấy từ năm 2011 đến 2013 tỷ lệ lợi nhuận đã tăng trưởng, trong khi đó, để khai phá được khoảng Itemsets đó trong năm 2014 thì phải đặt giá trị “minimum utility ratio” thấp hơn rất nhiều. Điều này cũng cho thấy việc kinh doanh trong năm 2014 không thuận lợi (như đã phân tích ở phần trực quan hoá kết quả).

Kết quả hiệu suất thực thi cho thấy sự khác biệt về độ lớn thời gian và bộ nhớ cần thiết để thực hiện khai thác chuỗi trên các bộ dữ liệu khác nhau. Như trên bảng so sánh, khi chạy trên toàn bộ bộ dữ liệu từ năm 2011 đến năm 2014, thời gian và bộ nhớ sử dụng ít hơn nhiều so với khi chạy riêng lẻ cho từng năm 2012 (2394 ms - 168.34 MB) và 2013 (10867 ms - 40.62 MB). Điều này có thể được giải thích bằng hệ số “Join count”, tức số lần phải thực hiện phép nối (join) trong quá trình khai thác chuỗi. Hệ số này phản ánh mức độ tận dụng cấu trúc của bộ dữ liệu để giảm thiểu số lần ghép nối. Như vậy, ta có thể nhận thấy cấu trúc dữ liệu của năm 2012 và 2013 phức tạp hơn so với các phần dữ liệu còn lại (các Item có mối liên hệ phức tạp hơn). Sự ảnh hưởng của “Join count” càng rõ ràng khi so sánh năm 2013 và 2014, trong khi cả hai đều sử dụng bộ nhớ gần bằng nhau nhưng do hệ số “Join count” của năm 2014 nhỏ hơn rất nhiều so với năm 2013, nên thời gian sử dụng của nó cũng giảm đáng kể.

Do đó, ta có thể kết luận rằng việc chia tách bộ dữ liệu sẽ giúp ta khám phá nhiều khía cạnh hơn, có thể xem xét việc các Itemset có thể bị bỏ sót. Đây là chiến lược ngắn hạn, còn về dài hạn thì việc chạy trên bộ dữ liệu tổng thể sẽ tìm được các Itemset mang lại lợi nhuận tốt và ổn định, từ đó, định hình chiến lược kinh doanh. Bên cạnh đó, việc lựa chọn đánh đổi chi phí về thời gian, bộ nhớ cũng cung cấp những thông tin và hiệu quả nhất định, qua đó, hỗ trợ cho người ra quyết định có nhiều thông tin và phương án lựa chọn hơn.

# D. KẾT LUẬN

## **Nhận xét**

Thuật toán FOSHU với thời gian chạy được đánh giá là có tiến bộ và khả năng áp dụng vào dữ liệu thực tế tốt hơn so với một số thuật toán được tạo ra để khai thác mục tiện ích cao (HUI).

Thuật toán FOSHU có thể giải quyết các vấn đề thời gian được bán của mặt hàng và cơ sở dữ liệu chứa các mặt hàng có lợi nhuận/ khối lượng âm. Đồng thời, thuật toán được chứng minh là vượt trội hơn thuật toán trước đó của nó là TS-HOUN về thời gian thực hiện. Vì vậy, nó là sự lựa chọn phù hợp để khai thác tập mục có tiện ích cao sẵn có gắn với yếu tố thời gian và tiện ích có thể âm.

Bên cạnh đó, thuật toán FOSHU còn có những hạn chế nhất định. Đầu tiên, đây là thuật toán không được gắn nhãn, vì vậy, có thể gây ra một số hạn chế trong việc xử lý dữ liệu đầu vào. Thông số đầu vào (minimum utility ratio) nhạy cảm và chưa có cách xác định. Tùy thuộc vào tập dữ liệu, việc tối ưu hóa các tham số có thể khó khăn và đòi hỏi kiểm thử nhiều giá trị khác nhau. Sự lựa chọn không chính xác của các tham số này có thể dẫn đến kết quả không tốt hoặc tốn nhiều thời gian. Thuật toán này có thể gặp khó khăn nếu tập dữ liệu chứa nhiều chuỗi có sự biến đổi lớn và phức tạp, dẫn đến tiêu tốn nhiều thời gian và tài nguyên để khai thác.

Thông qua thuật toán, có thể thấy được khoảng thời gian bán hàng và thời gian trên kệ của các mặt hàng vô cùng quan trọng trong khi một số thuật toán đã bỏ qua yếu tố thời gian này. Ngoài ra, nhóm nhận thấy một số các cặp sản phẩm đem lại nhiều lợi nhuận hơn khi đi cùng nhau, hay các sản phẩm được bán ra có xu hướng tập trung ở một khoảng thời gian, mùa vụ nào đó. Qua đó, nếu kết hợp các sản phẩm với nhau kèm theo chương trình khuyến mãi và chiến lược giảm giá bán phù hợp có thể thúc đẩy việc bán sản phẩm, tăng doanh thu, và hạn chế nhập kho các sản phẩm cụ thể vào khoảng thời gian cụ thể khi các sản phẩm đó không được bán chạy.

Thực hiện và phân tích kết quả thuật toán đối với đề tài đặt ra, nhóm có thể nói công ty AdventureWorks đã có kết quả kinh doanh tương đối tốt, khi duy trì được sự ổn định cũng như xu hướng mua hàng của khách đều đặn theo thời gian. Cùng với đó là chiến lược giảm giá để kích thích, thu hút khách hàng có hiệu quả và đã mang lại kết quả tốt. Ngoài ra, kết quả cho thấy tỷ lệ lợi nhuận của các cặp Itemsets không chiếm tỷ lệ lớn. Do công ty AdventureWorks có rất nhiều sản phẩm, do đó, có thể nhận định rằng các sản phẩm của AdventureWorks được kinh doanh khá đồng đều. Đây có thể là dấu hiệu tốt vì nó cho thấy AdventureWorks không bị lệ thuộc vào một hay một nhóm mặt hàng cụ thể nào.

## **Kết luận**

Thuật toán FOSHU được đánh giá là có thời gian chạy nhanh hơn và ít tốn bộ nhớ hơn so với các thuật toán được đề xuất để giải quyết các bài toán khai thác tập mục lợi ích cao với yếu tố thời gian và tiện ích có thể âm. FOSHU mang đến một số cải tiến so với phiên bản trước. Thông qua đề tài, áp dụng thuật toán FOSHU, nhóm đã nhận ra được tầm quan trọng của thời gian mặt hàng được bán trên kệ hàng và lợi nhuận âm trong cơ sở dữ liệu giao dịch. Điều này góp phần hạn chế hàng tồn kho, tăng lợi nhuận, tạo ra một chiến lược bán hàng phù hợp cho doanh nghiệp. Đồng thời, nhóm cũng đã phân tích và đánh giá khái quát được tình hình, chiến lược kinh doanh của bộ dữ liệu, cũng như đạt được các mục tiêu đề ra.

## **Hướng phát triển**

Sau quá trình nhóm thực hiện và tham khảo các bài nghiên cứu khác, nhóm nhận thấy thuật toán FOSHU mang lại kết quả tốt trong quá trình xử lý tập dữ liệu giao dịch ở từng mốc thời gian, tuy nhiên dựa vào kết quả của thuật toán này không đủ để đưa ra các chiến lược trong kinh doanh trong tương lai.

Nhóm có các hướng như kết hợp thuật toán khám phá các tập mục có tiện ích cao xuyên cấp (FEACP) trong cơ sở dữ liệu giao dịch chứa thông tin tiện ích và có phân loại nhằm khai thác thêm mối liên hệ về sự phân loại của các tập mục đi cùng nhau, từ đó, tìm ra sự tương quan tốt hơn giữa các tập mục. Hay kết hợp với thuật toán khám phá các tập mục tiện ích cao chi phí thấp trong cơ sở dữ liệu giao dịch chứa thông tin tiện ích và chi phí (LCIM) nhằm đánh giá các tập mục không chỉ thông qua tiện ích (lợi nhuận, thời gian) mà còn dựa vào chi phí để tạo ra tiện ích đó, vì đôi khi cần phải bỏ ra nhiều chi phí để đạt được lợi nhuận cao, trong khi một số tập mục có thể không yêu cầu nhiều chi phí nhưng vẫn có lợi nhuận tốt. Bên cạnh đó, còn có thể áp dụng thêm thuật toán khám phá các tập mục hữu ích cao tối thiểu (MinFHM), vì trên thực tế các tập mục dài thường khó xuất hiện hơn so với các tập mục ngắn.

Vì vậy, trong tương lai nhóm hy vọng có thể áp dụng những gì đã tìm hiểu được, những vấn đề đã khai thác cũng như mở rộng vấn đề, kết hợp các thuật toán với nhau để giải quyết đề tài nhóm đặt ra nói riêng và các chủ đề khai phá dữ liệu khác nói chung được cải thiện hơn, cung cấp được nhiều thông tin có ích cho việc ra quyết định hơn.

# THAM KHẢO

Philippe, F.-V. (2015). *Example: Mining On-Shelf High-Utility Itemsets from a Transaction Database using the FOSHU Algorithm (SPMF - Java).* Retrieved from SPMF: https://www.philippe-fournier-viger.com/spmf/FOSHU.php

Philippe, F.-V., & Souleymane, Z. (2015). *FOSHU: Faster On-Shelf High Utility Itemset Mining -with or without Negative Unit Profit.* Retrieved from SPMF: https://www.philippe-fournier-viger.com/spmf/SAC2015\_FOSHU\_on-shelf-high-utility-itemset-mining.pdf