

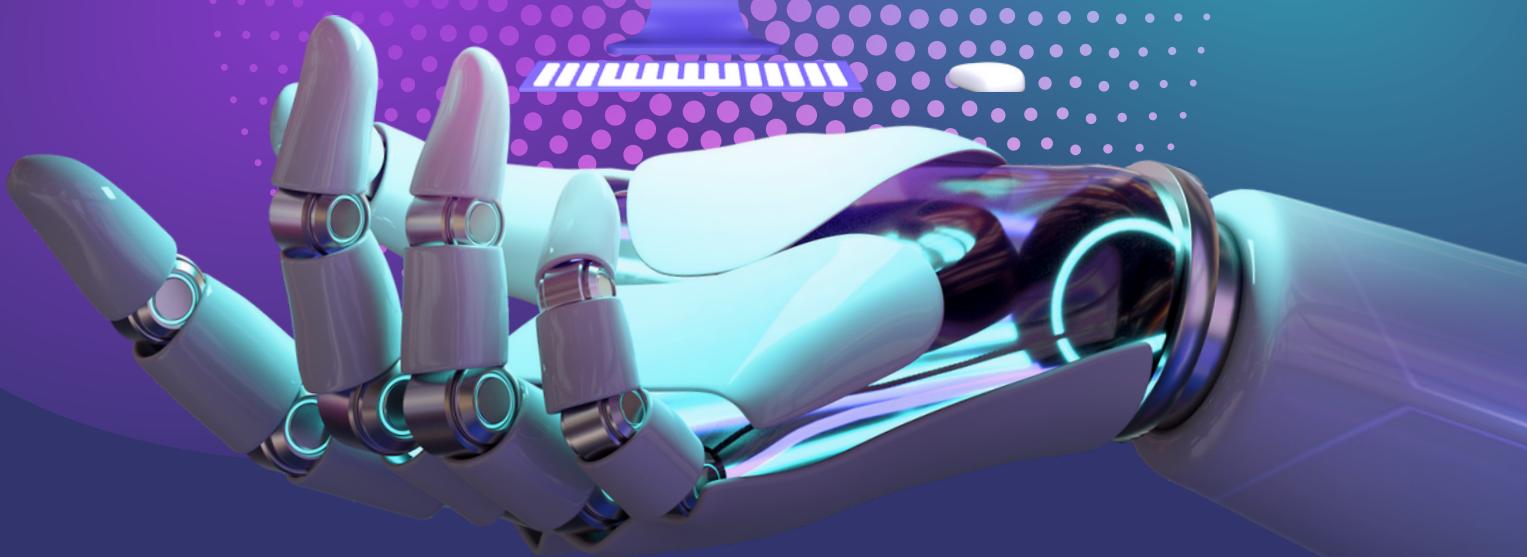


FPT UNIVERSITY



PROJECT REPORT

SALE FORECASTING



SUBJECT: DAP-301M
INSTRUCTOR: BUI VAN HIEU

Reported by Group 3



Table of content

1. About our group
2. Problem Statement
3. Data Collection
4. Data Preprocessing
5. Exploratory Data Analysis
6. Model Building, Evaluate and Optimization
7. Data - Centric. A new approach
8. Summary
9. References





1. About our group



Lê Anh Tú

Leader



Lê Minh Nghĩa

Data Analyst



Đỗ Huy Hoàng

Modeling



Nguyễn Tuấn Anh

Data



Đào Hữu Thủy

Optimization

MAP OF THE PROJECT'S JOBS

Lê Anh Tú
Nguyễn Tuấn Anh
Lê Minh Nghĩa
Đỗ Huy Hoàng
Đào Hữu Thủy

- Problem Statement
- Data Collection
- Data Centric - A new approach
- Summary

- Preprocessing Data

- Exploratory Data Analysis

- Model and Evaluate

- Optimization



2. Problem Statement

WHAT IS SALES FORECAST?

Dự báo bán hàng là sự ước đoán về lượng bán của doanh nghiệp, (tính bằng tiền hoặc theo đơn vị sản phẩm) có thể bán được trong một thời kì nhất định dưới một kế hoạch marketing đã được thông qua và dưới một tổ hợp các điều kiện kinh tế được giả định.

Nói chung, việc dự báo lượng bán là trách nhiệm của giám đốc marketing. Trong các doanh nghiệp lớn, do tính chất phức tạp của chính việc dự báo nên nó thường được giao cho một bộ phận chuyên môn trong nghiên cứu marketing.

Các dữ liệu dự báo cần được đưa vào hệ thống hỗ trợ ra quyết định marketing của doanh nghiệp để những quản lý bán hàng và các cán bộ quản lý những chức năng khác sử dụng. Đối với nhiều doanh nghiệp, việc dự báo lượng bán là công cụ then chốt trong việc lập kế hoạch và điều khiển hoạt động bán hàng.

WHY WE NEED IT?

1. Vì sao chủ kinh doanh cần dự báo doanh thu?

Dự báo doanh thu là công việc vô cùng quan trọng đóng vai trò là “kim chỉ nam” dẫn lối mọi hoạt động kinh doanh trong doanh nghiệp. Dựa vào những số liệu phân tích đó, bạn có thể đưa ra những quyết định kinh doanh đúng đắn và sáng suốt hơn, kiểm soát được dòng tiền và các hạng mục chi phí. Cụ thể, dự báo doanh thu chính xác đem lại những lợi ích nổi bật sau:

2. Cân đối dòng tiền hiệu quả

Nhiều nhà hàng có doanh thu không đều đặn vào các tháng khác nhau trong năm. Với dự báo doanh thu dự kiến hàng tháng hoặc hàng quý, chủ kinh doanh có thể điều chỉnh trích quỹ trong mùa cao điểm để có tiền trả cho các chi phí trong thời gian doanh thu ít hơn. Nhờ đó, nhà hàng sẽ không phải đổi mặt với tình trạng không có đủ tiền mặt để trả lương cho nhân viên hay thanh toán cho các hóa đơn và thuế vào những thời điểm khó khăn.



3. Kiểm soát nguyên vật liệu và hàng tồn kho

Dự báo được doanh số trong ngắn hạn giúp nhà hàng vừa đảm bảo lên kế hoạch nhập hàng đầy đủ phục vụ bán hàng, vừa giảm thiểu nguy cơ xảy ra tình trạng tồn kho lãng phí thực phẩm. Bộ phận mua hàng có thể đặt hàng và lên lịch giao nguyên vật liệu một cách chính xác so với nhu cầu ước tính. Điều này giúp tiết kiệm chi phí vận chuyển và bảo quản hàng hóa không cần thiết.

4. Sắp xếp số lượng nhân sự hợp lý

Chi phí nhân sự là một trong những hạng mục tốn kém nhất của các nhà hàng. Trong trường hợp này, bản dự báo doanh thu nhà hàng sẽ hướng dẫn chỉ đường để chủ kinh doanh lập kế hoạch cân đối số lượng, tiền lương, cách sắp xếp ca làm việc,... cho nhân viên như thế nào cho hiệu quả. Lập kế hoạch dựa trên cơ sở dữ liệu sẽ giúp nhà hàng tối ưu ngân sách, tăng số lượng nhân viên vào ngày doanh thu cao, đồng thời giảm số lượng nhân viên vào ngày doanh thu thấp để tránh việc dư thừa lãng phí.

5. Điều chỉnh kế hoạch và ngân sách marketing

Bán hàng và marketing luôn có mối quan hệ chặt chẽ với nhau. Chủ kinh doanh có thể quyết định chi “mạnh tay” hơn vào các chương trình marketing nếu nhà hàng dự kiến nhận được doanh thu bán hàng cao hơn. Tuy nhiên, nếu doanh thu dự báo quá thấp, nhà hàng cũng cần đẩy mạnh khuyến mãi, giảm giá với kênh truyền thông và chi phí hợp lý để đẩy mạnh doanh thu.

6. Đặt ra mục tiêu doanh thu kỳ vọng

Nhiều chủ kinh doanh thường cảm thấy phân vân giữa việc chọn một cái nhìn khiêm tốn, thận trọng về doanh thu hay là một giấc mơ đầy tham vọng để tạo động lực phấn đấu cho bản thân và thúc đẩy tinh thần làm việc của nhân viên. Thay vì chỉ dự báo một cách cảm tính, bạn nên có phương pháp tính toán hợp lý để có được con số chính xác gần với thực tế nhất. Doanh thu kỳ vọng để tạo cơ sở khen thưởng cho nhân viên có thể cao hơn doanh thu doanh thu dự báo để tạo động lực làm việc cho mọi người. Như vậy, hoạt động kinh doanh của nhà hàng sẽ có định hướng, mục tiêu rõ ràng thay vì theo phương châm “doanh thu được bao nhiêu tốt bấy nhiêu”.



TRADITIONAL SALES FORECAST

EXPERT OPINION

1. **Thành kiến:** Các chuyên gia có thể bị ảnh hưởng bởi thành kiến, sở thích hoặc liên kết cá nhân của họ, điều này có thể ảnh hưởng đến tính khách quan trong ý kiến của họ.
2. **Không nhất quán:** Các chuyên gia khác nhau có thể có những quan điểm khác nhau về một chủ đề, dẫn đến những ý kiến trái ngược nhau có thể gây khó khăn cho việc xác định thông tin chính xác hoặc đáng tin cậy nhất.
3. **Phạm vi hạn chế:** Ý kiến chuyên gia thường tập trung vào một lĩnh vực chuyên môn cụ thể, có nghĩa là họ có thể không xem xét bối cảnh rộng hơn hoặc các khía cạnh liên ngành của một vấn đề. Sự tập trung hạn hẹp này có thể dẫn đến việc thiếu hiểu biết toàn diện và bỏ qua các yếu tố quan trọng có thể ảnh hưởng đến tình hình.
4. **Chi phí và khả năng tiếp cận:** Việc thu thập ý kiến chuyên gia có thể tốn kém, đặc biệt nếu cần có sự tham gia của chuyên gia tư vấn hoặc chuyên gia. Hơn nữa, khả năng tiếp cận các chuyên gia có thể bị hạn chế, đặc biệt là ở một số khu vực địa lý nhất định hoặc đối với các cá nhân hoặc tổ chức có ít nguồn lực hơn.

RESEARCH MARKET

1. **Chi phí:** Tiến hành nghiên cứu thị trường toàn diện có thể tốn kém, đặc biệt khi liên quan đến cỡ mẫu lớn, công cụ chuyên dụng hoặc thuê các cơ quan nghiên cứu bên ngoài. Các doanh nghiệp nhỏ có thể gặp khó khăn trong việc phân bổ đủ nguồn lực cho nghiên cứu thị trường rộng rãi.
2. **Tốn thời gian:** Nghiên cứu thị trường thường đòi hỏi thời gian để lập kế hoạch, thực hiện, thu thập dữ liệu, phân tích và giải thích. Quá trình này có thể kéo dài, đặc biệt nếu nghiên cứu liên quan đến các phương pháp phức tạp hoặc đối tượng mục tiêu đa dạng.
3. **Dữ liệu không chính xác hoặc sai lệch:** Chất lượng của kết quả nghiên cứu thị trường phụ thuộc rất nhiều vào tính chính xác và tính đại diện của dữ liệu được thu thập. Luôn có nguy cơ thu được kết quả không chính xác hoặc sai lệch do các yếu tố như người trả lời hiểu sai câu hỏi, cố tình không trung thực hoặc lối lấy mẫu.
4. **Phạm vi giới hạn:** Nghiên cứu thị trường cung cấp những hiểu biết sâu sắc dựa trên các mục tiêu nghiên cứu cụ thể và các thông số được xác định trước. Nó có thể không nắm bắt được mọi khía cạnh của hành vi người tiêu dùng hoặc động lực thị trường, hạn chế chiều sâu của sự hiểu biết trong một số lĩnh vực nhất định.
5. **Thiếu thông tin theo thời gian thực:** Thu thập và phân tích dữ liệu thị trường là một quá trình tốn nhiều thời gian. Vào thời điểm có kết quả nghiên cứu, điều kiện thị trường hoặc sở thích của người tiêu dùng có thể đã thay đổi, khiến thông tin ít hữu ích hơn hoặc cần nghiên cứu bổ sung.



HISTORICAL SALES DATA

Bối cảnh hạn chế: Dữ liệu bán hàng trong quá khứ cung cấp thông tin về các xu hướng và mô hình bán hàng trong quá khứ, nhưng nó có thể không cung cấp hiểu biết đầy đủ về các yếu tố cơ bản ảnh hưởng đến các xu hướng đó. Các yếu tố như thay đổi sở thích của khách hàng, điều kiện thị trường, yếu tố kinh tế hoặc bối cảnh cạnh tranh có thể không được ghi lại chỉ trong dữ liệu lịch sử.



Nếu không xem xét các yếu tố bối cảnh này, các doanh nghiệp có thể đưa ra các giả định không chính xác hoặc không thích ứng với các điều kiện thay đổi.

Các sự kiện không lường trước được: Dữ liệu bán hàng trước đây có thể không tính đến các sự kiện hoặc khủng hoảng bất ngờ có thể ảnh hưởng đáng kể đến mô hình bán hàng. Những gián đoạn như thiên tai, bất ổn chính trị, suy thoái kinh tế hoặc đại dịch có thể làm thay đổi mạnh mẽ hành vi của người tiêu dùng và điều kiện thị trường. Trong những tình huống như vậy, dữ liệu bán hàng trong quá khứ có thể cung cấp ít hướng dẫn và doanh nghiệp cần phải linh hoạt và nhạy bén để vượt qua các tình huống không lường trước được.

REGRESSION

- Giả định tuyến tính:** Phân tích hồi quy giả định mối quan hệ tuyến tính giữa các biến phụ thuộc và biến độc lập. Tuy nhiên, trong các tình huống thực tế, giả định này có thể không đúng với mọi mối quan hệ. Nếu mối quan hệ là phi tuyến tính, phân tích hồi quy có thể tạo ra kết quả không chính xác.
- Đa cộng tuyến:** Đa cộng tuyến xảy ra khi có sự tương quan cao giữa các biến độc lập trong mô hình hồi quy. Điều này có thể dẫn đến các vấn đề như ước tính tham số không ổn định và khó diễn giải các tác động riêng lẻ của các biến.
- Các giá trị ngoại lệ và các quan sát có ảnh hưởng:** Phân tích hồi quy có thể nhạy cảm với các giá trị ngoại lai, là các giá trị cực đoan sai lệch đáng kể so với mẫu tổng thể của dữ liệu. Các ngoại lệ có thể làm sai lệch các hệ số ước tính và ảnh hưởng đến độ chính xác của mô hình. Tương tự như vậy, các quan sát có ảnh hưởng, có tác động mạnh mẽ đến kết quả của mô hình, cũng có thể ảnh hưởng đến tính hợp lệ của phân tích.



WHY NEED MACHINE LEARNING IN SALES FORECAST

1. Độ phức tạp của dữ liệu: Các phương pháp truyền thống thường hoạt động tốt khi xử lý các tập dữ liệu đơn giản và các mối quan hệ tương đối đơn giản. Tuy nhiên, họ có thể gặp khó khăn khi xử lý các mẫu phức tạp và khối lượng dữ liệu lớn. Mặt khác, các thuật toán học máy vượt trội trong việc phân tích các cấu trúc dữ liệu phức tạp và nắm bắt các mẫu phức tạp, cho phép dự đoán chính xác hơn.
2. Khả năng thích ứng với thay đổi: Các phương pháp dự báo truyền thống thường là tĩnh và yêu cầu cập nhật thủ công khi có dữ liệu mới hoặc khi điều kiện thị trường thay đổi. Các mô hình học máy có thể thích ứng và có thể tự động điều chỉnh dự đoán của chúng dựa trên thông tin mới. Họ có thể học hỏi từ dữ liệu thời gian thực và nhanh chóng thích ứng với môi trường kinh doanh đang thay đổi.
3. Độ chính xác của dự báo: Bằng cách tận dụng các kỹ thuật tiên tiến như mạng thần kinh, cây quyết định và phương pháp tập hợp, các mô hình học máy có thể nắm bắt một cách hiệu quả các mẫu phức tạp và đưa ra dự đoán chính xác. Các phương pháp truyền thống có thể gặp khó khăn trong việc nắm bắt các mối quan hệ phức tạp và phi tuyến tính, dẫn đến các dự báo kém chính xác hơn.
4. Khả năng mở rộng: Khi các tập dữ liệu ngày càng lớn và phức tạp hơn, các phương pháp dự báo truyền thống có thể gặp phải các vấn đề về khả năng mở rộng. Phân tích hồ sơ bán hàng mở rộng theo cách thủ công hoặc sử dụng các kỹ thuật thống kê truyền thống có thể trở nên không thực tế. Các mô hình máy học được thiết kế để xử lý các tập dữ liệu quy mô lớn một cách hiệu quả, khiến chúng phù hợp để phân tích một lượng lớn dữ liệu bán hàng.
5. Tự động hóa: Dự báo dựa trên máy học có thể tự động hóa quá trình tạo dự báo doanh số. Sau khi mô hình được đào tạo, nó có thể tự động tạo dự đoán mà không cần can thiệp thủ công nhiều. Các phương pháp truyền thống thường yêu cầu tính toán thủ công, thao tác dữ liệu và đánh giá chủ quan, có thể tốn thời gian và dễ mắc lỗi.
6. Học liên tục: Các mô hình máy học có thể liên tục học và cải thiện theo thời gian bằng cách kết hợp dữ liệu và phản hồi mới. Họ có thể điều chỉnh dự đoán của mình dựa trên kết quả bán hàng thực tế, cho phép tinh chỉnh liên tục và tăng độ chính xác. Các phương pháp truyền thống có thể thiếu khả năng học tập liên tục này.



3. Data Collection

Recruitment Prediction Competition

Walmart Recruiting - Store Sales Forecasting

Use historical markdown data to predict store sales

Walmart · 688 teams · 9 years ago

688	688	12,240
Teams	Competitors	Entries

Points This competition awarded ranking points
Medals This competition awarded medals for tiers

Một thách thức của việc lập mô hình dữ liệu bán lẻ là nhu cầu đưa ra quyết định dựa trên lịch sử hạn chế. Nếu Giáng sinh đến nhưng mỗi năm một lần, thì cơ hội để xem các quyết định chiến lược đã tác động đến điểm mấu chốt như thế nào.

Trong cuộc thi tuyển dụng này, người tìm việc được cung cấp dữ liệu bán hàng lịch sử của 45 cửa hàng Walmart ở các khu vực khác nhau. Mỗi cửa hàng có nhiều bộ phận và người tham gia phải dự kiến doanh số bán hàng cho từng bộ phận trong mỗi cửa hàng. Để thêm vào thử thách, các sự kiện giảm giá ngày lễ đã chọn sẽ được đưa vào bộ dữ liệu. Những khoản giảm giá này được biết là có ảnh hưởng đến doanh số bán hàng, nhưng rất khó để dự đoán bộ phận nào bị ảnh hưởng và mức độ ảnh hưởng.

EVALUATE

This competition is evaluated on the weighted mean absolute error (WMAE):

$$\text{WMAE} = \frac{1}{\sum w_i} \sum_{i=1}^n w_i |y_i - \hat{y}_i|$$

where

- n is the number of rows
- \hat{y}_i is the predicted sales
- y_i is the actual sales
- w_i are weights. $w = 5$ if the week is a holiday week, 1 otherwise

DATA DESCRIPTION



Bạn được cung cấp dữ liệu bán hàng lịch sử cho 45 cửa hàng Walmart ở các khu vực khác nhau. Mỗi cửa hàng có một số bộ phận và bạn có nhiệm vụ dự đoán doanh số bán hàng trên toàn bộ bộ phận cho mỗi cửa hàng.

Ngoài ra, Walmart tổ chức một số sự kiện giảm giá khuyến mại trong suốt cả năm. Những đợt giảm giá này diễn ra trước các ngày lễ nổi bật, trong đó có bốn ngày lễ lớn nhất là Super Bowl, Ngày Lao động, Lễ tạ ơn và Giáng sinh. Các tuần bao gồm các ngày lễ này được đánh giá có trọng số cao hơn năm lần so với các tuần không có ngày nghỉ. Một phần của thách thức do cuộc thi này đưa ra là lập mô hình tác động của việc giảm giá trong các tuần nghỉ lễ này khi không có dữ liệu lịch sử hoàn chỉnh/lý tưởng.

- **Stores.csv:** Tệp này chứa thông tin ẩn danh về 45 cửa hàng, cho biết loại và quy mô cửa hàng.

- **Train.csv**

Đây là dữ liệu đào tạo lịch sử, bao gồm từ 2010-02-05 đến 2012-11-01. Trong tệp này, bạn sẽ tìm thấy các trường sau:

1. Store - số cửa hàng
2. Dept - số phòng ban
3. Day- tuần
4. Weekly_Sales - doanh số cho bộ phận nhất định trong cửa hàng nhất định
5. IsHoliday - liệu tuần đó có phải là tuần lễ đặc biệt hay không

- **Test.csv:** Tệp này giống với train.csv, ngoại trừ việc chúng tôi đã giữ lại doanh thu hàng tuần. Bạn phải dự đoán doanh số bán hàng cho từng bộ ba cửa hàng, bộ phận và ngày trong tệp này.

- **Features.csv**

Tệp này chứa dữ liệu bổ sung liên quan đến cửa hàng, bộ phận và hoạt động khu vực cho các ngày nhất định. Nó chứa các trường sau:

1. Store - number of store
2. Day- Week
3. Temperature - nhiệt độ trung bình trong khu vực
4. Fuel_Price - chi phí nhiên liệu trong khu vực
5. Markdown1-5 - dữ liệu ẩn danh liên quan đến giảm giá khuyến mại mà Walmart đang chạy. Dữ liệu Markdown chỉ khả dụng sau tháng 11 năm 2011 và không phải lúc nào cũng có sẵn cho tất cả các cửa hàng. Bất kỳ giá trị nào bị thiếu đều được đánh dấu bằng NA.
6. CPI - chỉ số giá tiêu dùng
7. Unemployed - tỷ lệ thất nghiệp
8. IsHoliday - liệu tuần đó có phải là tuần lễ đặc biệt hay không

Để thuận tiện, bốn ngày lễ rơi vào các tuần tiếp theo trong tập dữ liệu (không phải tất cả các ngày lễ đều có trong dữ liệu):

Super Bowl: 12-Feb-10, 11-Feb-11, 10-Feb-12, 8-Feb-13

Ngày Lao động: 10-Sep-10, 9-Sep-11, 7-Sep-12, 6-Sep-13

Lễ tạ ơn: 26-Nov-10, 25-Nov-11, 23-Nov-12, 29-Nov-13

Giáng sinh: 31-Dec-10, 30-Dec-11, 28-Dec-12, 27-Dec-13

4. Data PreProcessing

SỬ LÝ DỮ LIỆU CHUẨN BỊ CHO XÂY DỰNG MODEL

DATA PROCESSING LÀ PHƯƠNG PHÁP THU THẬP CÁC DỮ LIỆU THÔ VÀ CHUYỂN NÓ THÀNH NHỮNG THÔNG TIN HỮU ÍCH, CÓ Ý NGHĨA. NẾU DỮ LIỆU CHỈ Ở ĐẠNG THÔ THÌ DOANH NGHIỆP SẼ KHÔNG ĐỂ SỬ DỤNG DỮ LIỆU NÀY ĐỂ PHÂN TÍCH. CHÍNH VÌ THẾ MÀ DOANH NGHIỆP CẦN PHẢI XỬ LÝ DỮ LIỆU. DATA PROCESSING LÀ PHƯƠNG PHÁP THU THẬP CÁC DỮ LIỆU THÔ VÀ CHUYỂN NÓ THÀNH NHỮNG THÔNG TIN HỮU ÍCH, CÓ Ý NGHĨA.

CÁC BƯỚC CƠ BẢN TRONG DATA PROCESSING



THỰC TẾ TRONG BÀI TOÁN NÀY

Recruitment Prediction Competition

Walmart Recruiting - Store Sales Forecasting

Use historical markdown data to predict store sales

Walmart · 688 teams · 9 years ago

688 Teams 688 Competitors 12,240 Entries

Points This competition awarded ranking points
Medals This competition awarded medals for tiers

VẤN ĐỀ GẶP PHẢI Ở BƯỚC PREPROCESSING

KẾT QUẢ LẦN 1

	A	B
1	Id	Weekly_Sales
2	1_1_2012-11-02	30867.4365
3	1_1_2012-11-09	39315.1823
4	1_1_2012-11-16	37067.38
5	1_1_2012-11-23	22295.0514
6	1_1_2012-11-30	22443.7327
7	1_1_2012-12-07	28070.0585
8	1_1_2012-12-14	25619.0712
9	1_1_2012-12-21	28630.5976
10	1_1_2012-12-28	41914.9149
11	1_1_2013-01-04	37437.738
12	1_1_2013-01-11	23122.51
13	1_1_2013-01-18	21147.5742
14	1_1_2013-01-25	19115.736
15	1_1_2013-02-01	19838.5489
16	1_1_2013-02-08	18085.1853
17	1_1_2013-02-15	15854.6979
18	1_1_2013-02-22	15979.704
19	1_1_2013-03-01	17166.1112
20	1_1_2013-03-08	16735.3661
21	1_1_2013-03-15	16231.2479
22	1_1_2013-03-22	16163.2165
23	1_1_2013-03-29	16682.9472
24	1_1_2013-04-05	17302.596
25	1_1_2013-04-12	16852.6161
26	1_1_2013-04-19	15948.6742
27	1_1_2013-04-26	16118.2612
28	1_1_2013-05-03	16829.7904
29	1_1_2013-05-10	15786.3652
30	1_1_2013-05-17	15766.8409
31	1_1_2013-05-24	16103.405
32	1_1_2013-05-31	16993.5489
33	1_1_2013-06-07	17645.0819
34	1_1_2013-06-14	18245.9566
35	1_1_2013-06-21	17452.2479
36	1_1_2013-06-28	21763.2904
37	1_1_2013-07-05	25088.6411
38	1_1_2013-07-12	27599.8294
39	1_1_2013-07-19	25488.2677

KẾT QUẢ LẦN 2

	A	B
1	Id	Weekly_Sales
2	1_1_2012-11-02	14373.12557
3	1_1_2012-11-09	14251.41948
4	1_1_2012-11-16	14129.71338
5	1_1_2012-11-23	14823.34086
6	1_1_2012-11-30	13886.30119
7	1_1_2012-12-07	14468.75238
8	1_1_2012-12-14	14347.04628
9	1_1_2012-12-21	14225.34018
10	1_1_2012-12-28	14918.96766
11	1_1_2013-01-04	12361.07606
12	1_1_2013-01-11	12239.36996
13	1_1_2013-01-18	12117.66387
14	1_1_2013-01-25	11995.95777
15	1_1_2013-02-01	12601.88087
16	1_1_2013-02-08	13295.50835
17	1_1_2013-02-15	12358.46868
18	1_1_2013-02-22	12236.76258
19	1_1_2013-03-01	12772.26995
20	1_1_2013-03-08	12650.56386
21	1_1_2013-03-15	12528.85776
22	1_1_2013-03-22	12407.15166
23	1_1_2013-03-29	12285.44556
24	1_1_2013-04-05	12891.36867
25	1_1_2013-04-12	12769.66257
26	1_1_2013-04-19	12647.95647
27	1_1_2013-04-26	12526.25037
28	1_1_2013-05-03	13108.70157
29	1_1_2013-05-10	12986.99547
30	1_1_2013-05-17	12865.28937
31	1_1_2013-05-24	12743.58328
32	1_1_2013-05-31	12621.87718
33	1_1_2013-06-07	13227.80028
34	1_1_2013-06-14	13106.09418
35	1_1_2013-06-21	12984.38809
36	1_1_2013-06-28	12862.68199
37	1_1_2013-07-05	13445.13318
38	1_1_2013-07-12	13323.42708
39	1_1_2013-07-19	13201.72099

KẾT QUẢ LẦN 3

	A	B
1	Id	Weekly_Sales
2	1_1_2012-11-02	36399.78701
3	1_1_2012-11-09	20173.87937
4	1_1_2012-11-16	19713.98749
5	1_1_2012-11-23	20851.11334
6	1_1_2012-11-30	27037.11056
7	1_1_2012-12-07	29492.85217
8	1_1_2012-12-14	46525.35088
9	1_1_2012-12-21	48975.7962
10	1_1_2012-12-28	22626.54436
11	1_1_2013-01-04	16903.79691
12	1_1_2013-01-11	17181.49977
13	1_1_2013-01-18	18161.26257
14	1_1_2013-01-25	18515.29354
15	1_1_2013-02-01	23742.71872
16	1_1_2013-02-08	38531.35559
17	1_1_2013-02-15	54428.47167
18	1_1_2013-02-22	20237.25343
19	1_1_2013-03-01	20407.91446
20	1_1_2013-03-08	21300.87063
21	1_1_2013-03-15	21574.04632
22	1_1_2013-03-22	22004.08252
23	1_1_2013-03-29	31321.52271
24	1_1_2013-04-05	58744.429
25	1_1_2013-04-12	39217.56652
26	1_1_2013-04-19	18498.33743
27	1_1_2013-04-26	17122.94736
28	1_1_2013-05-03	18816.22029
29	1_1_2013-05-10	18039.74098
30	1_1_2013-05-17	17106.12773
31	1_1_2013-05-24	16676.88411
32	1_1_2013-05-31	16423.60863
33	1_1_2013-06-07	17509.76952
34	1_1_2013-06-14	17190.61279
35	1_1_2013-06-21	16384.27423
36	1_1_2013-06-28	16050.1786
37	1_1_2013-07-05	17296.23744
38	1_1_2013-07-12	16712.9871
39	1_1_2013-07-19	16326.00102

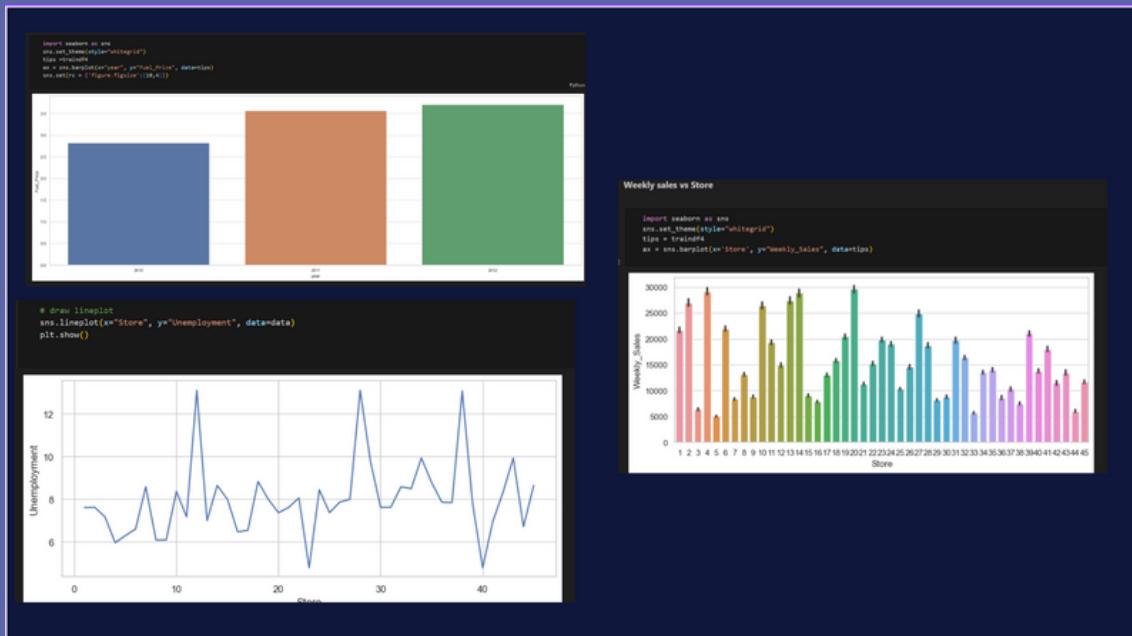
3 ảnh trên là 3 kết quả sau khi sử dụng model, chúng ta có thể thấy được nó đưa ra 3 kết quả khác nhau, và gần như sự lệch số rất lớn, nó nằm ở việc người sử lý data và người xây dựng model khác nhau, bình thường thì việc sử lý dữ liệu chuẩn bị và đưa ra nhận xét kết quả nó cũng nằm đến những yếu tố mà chúng ta quan tâm, hoặc nếu chuẩn bị cho việc ¹⁰ đưa vào model thì nó phải dựa trên yếu tố các model cần những cột dữ liệu gì. Tức là có model trước sẽ dễ dàng cho việc data preprocessing hơn.

VẤN ĐỀ GẶP PHẢI Ở BƯỚC PREPROCESSING

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	
64	1	1	#####	28762.37	0	71.27	3.743	0	0	0	0	0	215.2919	7.682	1	151315	2011	4	15	15	223	253	0	0	0	
65	1	1	#####	1050.31	0	72.99	3.807	0	0	0	0	0	215.4599	7.682	1	151315	2011	4	16	22	216	246	0	0	0	
66	1	1	#####	41512.39	0	72.03	3.81	0	0	0	0	0	215.628	7.682	1	151315	2011	4	17	29	209	239	0	0	0	
67	1	1	5/6/2011	20138.19	0	64.61	3.906	0	0	0	0	0	215.796	7.682	1	151315	2011	5	18	6	202	232	0	0	0	
68	1	1	#####	17235.15	0	75.64	3.899	0	0	0	0	0	215.9641	7.682	1	151315	2011	5	19	13	195	225	0	0	0	
69	1	1	#####	15136.78	0	67.63	3.907	0	0	0	0	0	215.7339	7.682	1	151315	2011	5	20	20	184	218	0	0	0	
70	1	1	#####	15741.6	0	77.72	3.786	0	0	0	0	0	215.5038	7.682	1	151315	2011	5	21	27	181	211	0	0	0	
71	1	1	6/3/2011	16434.15	0	83	3.699	0	0	0	0	0	215.2737	7.682	1	151315	2011	6	22	3	174	204	0	0	0	
72	1	1	#####	15883.52	0	83.13	3.648	0	0	0	0	0	215.0435	7.682	1	151315	2011	6	23	10	167	197	0	0	0	
73	1	1	#####	14978.09	0	86.41	3.637	0	0	0	0	0	214.9981	7.682	1	151315	2011	6	24	17	160	190	0	0	0	
74	1	1	#####	15682.81	0	83.58	3.594	0	0	0	0	0	215.0911	7.682	1	151315	2011	6	25	24	153	183	0	0	0	
75	1	1	7/8/2011	15863.5	0	85.55	3.524	0	0	0	0	0	215.1841	7.962	1	151315	2011	7	26	1	146	176	0	0	0	
76	1	1	7/8/2011	16148.87	0	85.83	3.48	0	0	0	0	0	215.2772	7.962	1	151315	2011	7	27	8	139	169	0	0	0	
77	1	1	#####	15654.85	0	88.54	3.575	0	0	0	0	0	215.3611	7.962	1	151315	2011	7	28	15	132	162	0	0	0	
78	1	1	#####	15766.6	0	85.77	3.651	0	0	0	0	0	215.4223	7.962	1	151315	2011	7	29	22	125	155	0	0	0	
79	1	1	#####	15922.41	0	86.83	3.682	0	0	0	0	0	215.2384	7.962	1	151315	2011	7	30	29	118	148	0	0	0	
80	1	1	8/5/2011	15295.55	0	91.05	3.684	0	0	0	0	0	215.5446	7.962	1	151315	2011	8	31	5	111	141	0	0	0	
81	1	1	#####	14539.79	0	90.76	3.638	0	0	0	0	0	215.6058	7.962	1	151315	2011	8	32	12	104	134	0	0	0	
82	1	1	#####	14689.24	0	89.94	3.554	0	0	0	0	0	215.6693	7.962	1	151315	2011	8	33	19	97	127	0	0	0	
83	1	1	#####	15277.27	0	87.96	3.523	0	0	0	0	0	215.7332	7.962	1	151315	2011	8	34	26	90	120	0	0	0	
84	1	1	9/7/2011	15277.27	0	87.83	3.533	0	0	0	0	0	215.7971	7.962	1	151315	2011	9	35	2	83	113	0	0	0	
85	1	1	9/7/2011	17746.68	1	76	3.546	0	0	0	0	0	215.8611	7.962	1	151315	2011	9	36	9	76	106	0	1	0	
86	1	1	#####	18535.48	0	79.94	3.526	0	0	0	0	0	216.0411	7.962	1	151315	2011	9	37	16	69	99	0	0	0	
87	1	1	#####	17859.3	0	75.8	3.467	0	0	0	0	0	216.3758	7.962	1	151315	2011	9	38	23	62	92	0	0	0	
88	1	1	#####	18337.68	0	79.69	3.355	0	0	0	0	0	216.1106	7.962	1	151315	2011	9	39	30	55	85	0	0	0	
89	1	1	#####	14539.79	0	69.31	3.185	0	0	0	0	0	217.0454	7.866	1	151315	2011	10	40	7	48	78	0	0	0	
90	1	1	#####	23077.55	0	71.74	3.24	0	0	0	0	0	215.5553	7.866	1	151315	2011	10	41	14	41	71	0	0	0	
91	1	1	#####	2338.02	0	63.21	3.353	0	0	0	0	0	217.5116	7.866	1	151315	2011	10	42	21	34	64	0	0	0	
92	1	1	#####	31379.9	0	66.57	3.372	0	0	0	0	0	217.6397	7.866	1	151315	2011	10	43	28	57	80	0	0	0	
93	1	1	#####	30866.06	0	54.98	3.332	0	0	0	0	0	217.8374	7.866	1	151315	2011	11	44	4	20	50	0	0	0	
94	1	1	#####	18689.54	0	59.11	3.297	10382.9	6115.67	0	215.07	2406.62	6551.47	217.0981	7.866	1	151315	2011	11	45	11	43	0	0	0	0
95	1	1	#####	18930.66	0	62.01	3.309	6974.12	254.39	0	54.98	427.39	5988.57	210.2389	7.866	1	151315	2011	11	46	18	36	0	0	0	0
96	1	1	#####	20911.25	1	60.01	4.134	210.31	54.98	55.01	42.38	427.39	5988.57	210.2389	7.866	1	151315	2011	11	47	25	1	29	0	0	1
97	1	1	#####	25293.49	0	48.91	3.172	5629.51	68	1308.11	2064.64	2047.5	210.7147	7.866	1	151315	2011	12	48	2	-8	22	0	0	0	
98	1	1	#####	33305.92	0	43.93	3.154	4640.65	19	15.02	3639.42	14461.32	218.9518	7.866	1	151315	2011	12	49	9	-15	15	0	0	0	
99	1	1	#####	45773.03	0	51.63	3.159	5011.32	67	347.37	225.79	4011.37	210.1795	7.866	1	151315	2011	12	50	16	-22	8	0	0	0	
100	1	1	#####	46768.75	0	47.96	3.111	2725.26	40.48	0	24.9	2739.43	210.3577	7.866	1	151315	2011	12	51	23	-29	1	0	0	0	
101	1	1	#####	23350.88	1	44.55	3.126	5761.2	46011.38	0	260.36	983.65	475.78	219.5136	7.866	1	151315	2011	12	52	30	-36	-6	0	0	0
102	1	1	1/6/2012	16567.69	0	49.01	3.157	6277.39	21813.16	143.1	14501.13	8483	219.7143	7.348	1	151315	2012	1	1	6	323	353	0	0	0	
103	1	1	#####	16904.4	0	49.53	3.261	5192.39	9015.82	42.34	427.02	3110.29	210.9016	7.348	1	151315	2012	1	7	17	216	246	0	0	0	

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	Store	Dept	Weekly_Sls	Holiday	Temperatur	Fuel_Price	CPI	Unemployment	Type	Size	year	month	week												
2	1	1	24924.5	0	42.31	2.572	211.0964	8.106	0	151315	2010	2	5												
3	1	1	46039.49	1	38.51	2.548	211.2422	8.106	0	151315	2010	2	6												
4	1	1	41595.55	0	39.93	2.514	211.2891	8.106	0	151315	2010	2	7												
5	1	1	19403.54	0	46.63	2.561	211.3196	8.106	0	151315	2010	2	8												
6	1	1	21827.79	0	46.5	2.625	211.3501	8.106	0	151315	2010	3	9												
7	1	1	21043.39	0	57.79	2.667	211.3806	8.106	0	151315	2010	3	10												
8	1	1	22136.64	0	54.58	2.72	211.3581	8.106	0	151315	2010	3	11												
9	1	1	26220.21	0	51.45	2.732	211.018	8.106	0	151315	2010	3	12												
10	1	1	15750.43	0	62.27	2.719	210.8204	7.808	0	151315	2010	4	13												
11	1	1	42960.91	0	65.86	2.77	210.6229	7.808	0	151315	2010	4	14</												

Vì vậy chúng tôi phải ngồi phân tích biểu đồ từ dữ liệu, để xem những dữ liệu nào quan trọng tới kết quả chúng tôi mong muốn để đưa ra việc sử lý và chọn lọc dữ liệu 1 cách tốt nhất





5. Exploratory Data Analysis

EDA (Exploratory Data Analysis – Phân tích Khám phá Dữ liệu) là một bước quan trọng trước khi làm bất kỳ một bài toán ML với dữ liệu dạng bảng nào.

Trước khi xây dựng mô hình, bạn cần xây dựng đặc trưng. Trước khi xây dựng đặc trưng, bạn phải làm bước khám phá dữ liệu.

EXTRACTING DATE INFORMATION

Doanh số bán hàng được thu thập trên cơ sở hàng tuần cho năm 2010-2012. Vì vậy, sẽ chia cột date để trích xuất thông tin cho year, month và week.

```
[ ] #Extracting Date Information
def split_date(df):
    df['Date'] = pd.to_datetime(df['Date'])
    df['Year'] = df.Date.dt.year
    df['Month'] = df.Date.dt.month
    df['Day'] = df.Date.dt.day
    df['WeekOfYear'] = (df.Date.dt.isocalendar().week)*1.0

    feature_store = features.merge(stores, how='inner', on = "Store")
```

THE DIMENSIONS OF THE DATASET

Sau khi tải tập dữ liệu, trước tiên chúng ta sẽ xem xét kích thước của tập dữ liệu.
→ Dữ liệu có 421570 mẫu và 20 biến.

```
[ ] train_df.shape
(421570, 20)
```

INFO OF THE FEATURES



```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 421570 entries, 0 to 421569
Data columns (total 20 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   Store        421570 non-null   int64  
 1   Dept         421570 non-null   int64  
 2   Date         421570 non-null   datetime64[ns]
 3   Weekly_Sales 421570 non-null   float64 
 4   IsHoliday    421570 non-null   bool   
 5   Temperature  421570 non-null   float64 
 6   Fuel_Price   421570 non-null   float64 
 7   MarkDown1   150681 non-null   float64 
 8   MarkDown2   111248 non-null   float64 
 9   MarkDown3   137091 non-null   float64 
 10  MarkDown4   134967 non-null   float64 
 11  MarkDown5   151432 non-null   float64 
 12  CPI          421570 non-null   float64 
 13  Unemployment 421570 non-null   float64 
 14  Type         421570 non-null   object  
 15  Size          421570 non-null   int64  
 16  Year          421570 non-null   int64  
 17  Month         421570 non-null   int64  
 18  Day           421570 non-null   int64  
 19  WeekOfYear   421570 non-null   float64 
dtypes: Float64(1), bool(1), datetime64[ns](1), float64(10), int64(6), object(1)
memory usage: 61.9+ MB

```

STATISTICAL SUMMARY OF THE FEATURES

	count	mean	std	min	25%	50%	75%	max
Store	421570.0	22.200546	12.785297	1.0	11.0	22.0	33.0	45.0
Dept	421570.0	44.260317	30.492054	1.0	18.0	37.0	74.0	99.0
Weekly_Sales	421570.0	15981.258123	22711.183519	-4988.94	2079.65	7612.03	20205.8525	693099.36
Temperature	421570.0	-9.108007	5.693806	-28.290123	-13.246914	-8.490741	-4.728395	3.253086
Fuel_Price	421570.0	3.361027	0.458515	2.472	2.933	3.452	3.738	4.468
MarkDown1	150681.0	7246.420196	8291.221345	0.27	2240.27	5347.45	9210.9	88646.76
MarkDown2	111248.0	3334.628621	9475.357325	-265.76	41.6	192.0	1926.94	104519.54
MarkDown3	137091.0	1439.421384	9623.07829	-29.1	5.08	24.6	103.99	141630.61
MarkDown4	134967.0	3383.168256	6292.384031	0.22	504.22	1481.31	3595.04	67474.85
MarkDown5	151432.0	4628.975079	5962.887455	135.16	1878.44	3359.45	5563.8	108519.28
CPI	421570.0	171.201947	39.159276	126.064	132.022667	182.31878	212.416993	227.232807
Unemployment	421570.0	7.960289	1.863296	3.879	6.891	7.866	8.572	14.313
Size	421570.0	136727.915739	60980.583328	34875.0	93638.0	140167.0	202505.0	219622.0
Year	421570.0	2010.968591	0.796876	2010.0	2010.0	2011.0	2012.0	2012.0
Month	421570.0	6.44951	3.243217	1.0	4.0	6.0	9.0	12.0
Day	421570.0	15.673131	8.753549	1.0	8.0	16.0	23.0	31.0
WeekOfYear	421570.0	25.826762	14.151887	1.0	14.0	26.0	38.0	52.0

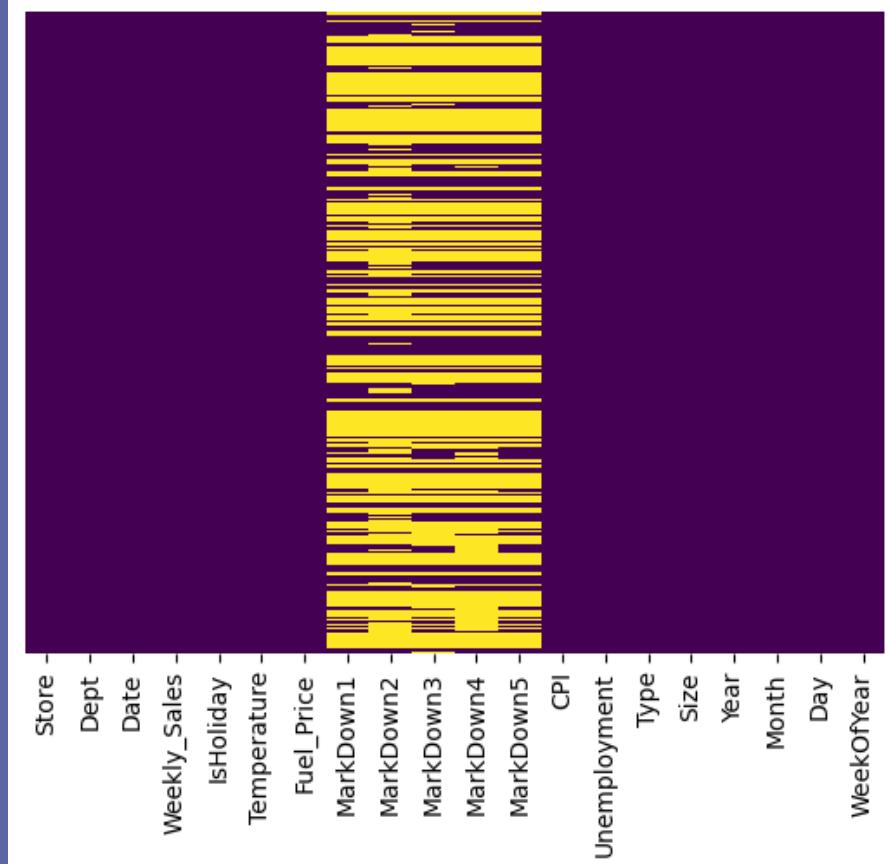
MISSING VALUES



Tìm số lượng của các giá trị bị thiếu trong tập dữ liệu.

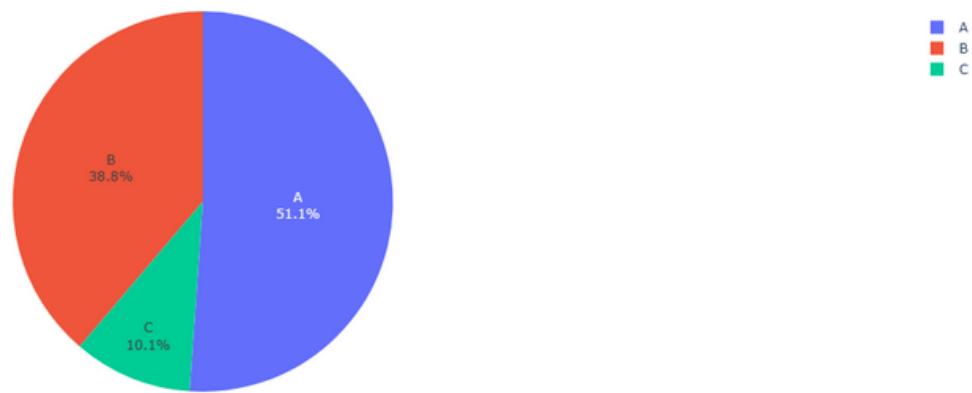
Store	0
Dept	0
Date	0
Weekly_Sales	0
IsHoliday	0
Temperature	0
Fuel_Price	0
MarkDown1	270889
MarkDown2	310322
MarkDown3	284479
MarkDown4	286603
MarkDown5	270138
CPI	0
Unemployment	0
Type	0
Size	0
Year	0
Month	0
Day	0
WeekOfYear	0
dtype: int64	0

Biểu đồ nhiệt của các giá trị bị thiếu



POPULARITY OF STORE TYPES

Popularity of Store Types



Cửa hàng loại A phổ biến hơn loại B và C

AVERAGE SALES - STORE TYPE

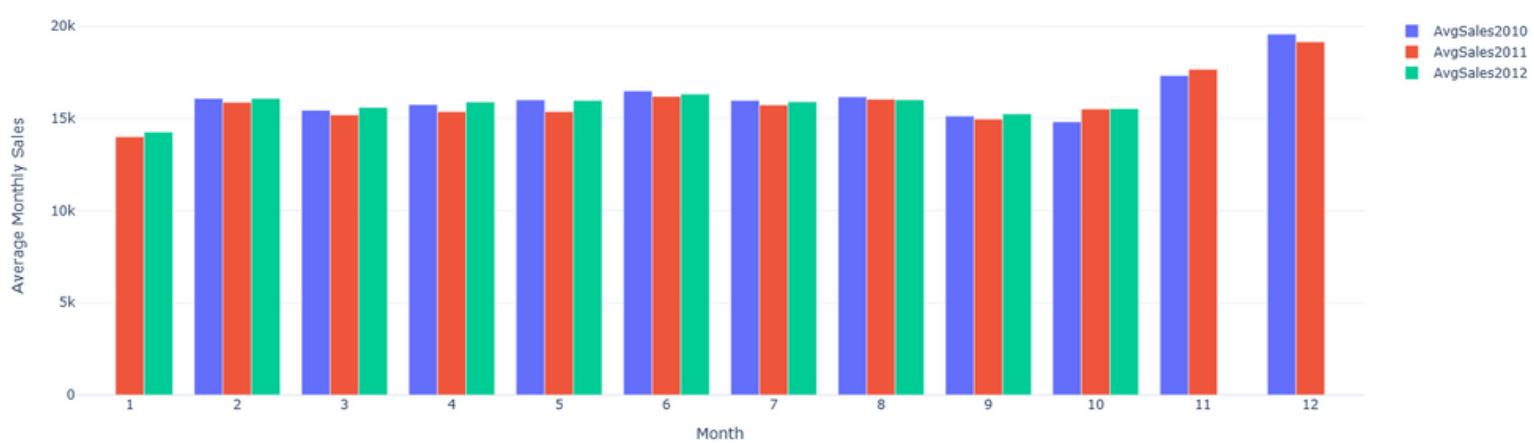


Average Sales - Per Store



Về doanh số, cửa hàng loại A vượt trội so với 2 loại còn lại

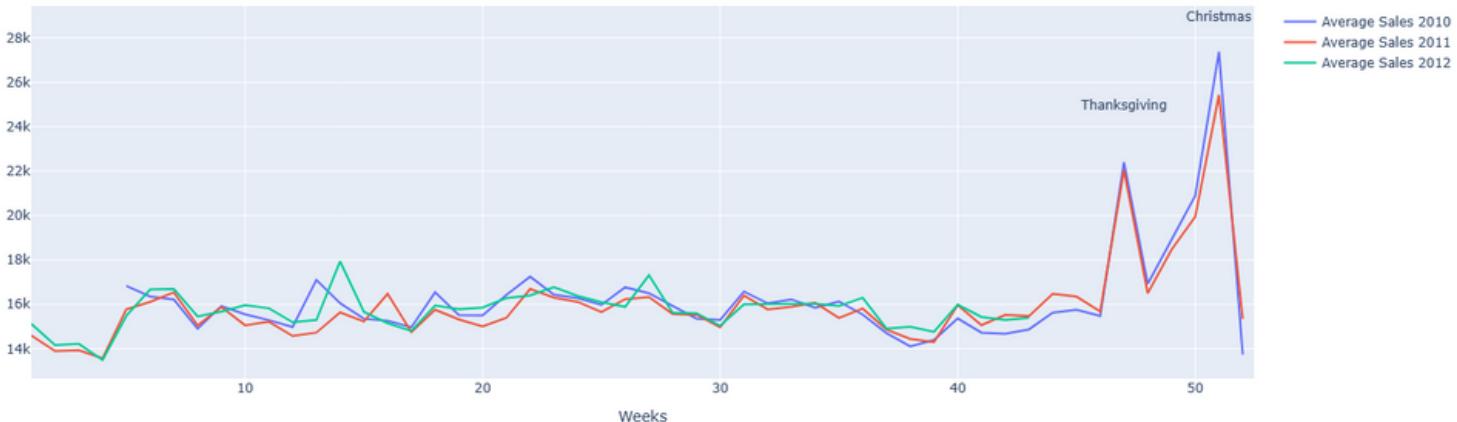
AVERAGE MONTHLY SALES - PER YEAR



- Tháng 1 chứng kiến doanh số bán hàng thấp nhất trong năm 2011 và 2012 trong khi đối với năm 2010 doanh số hàng tuần không được đưa ra trong dữ liệu
- Từ tháng 2 đến tháng 10, doanh số hàng tuần gần như không đổi vào khoảng 15000 trong 3 năm
- Tháng 11 và tháng 12 có doanh số bán hàng cao nhất trong năm 2010 và 2011 trong khi dữ liệu bán hàng của năm 2012 chưa được cung cấp

AVERAGE WEEKLY SALES - PER YEAR

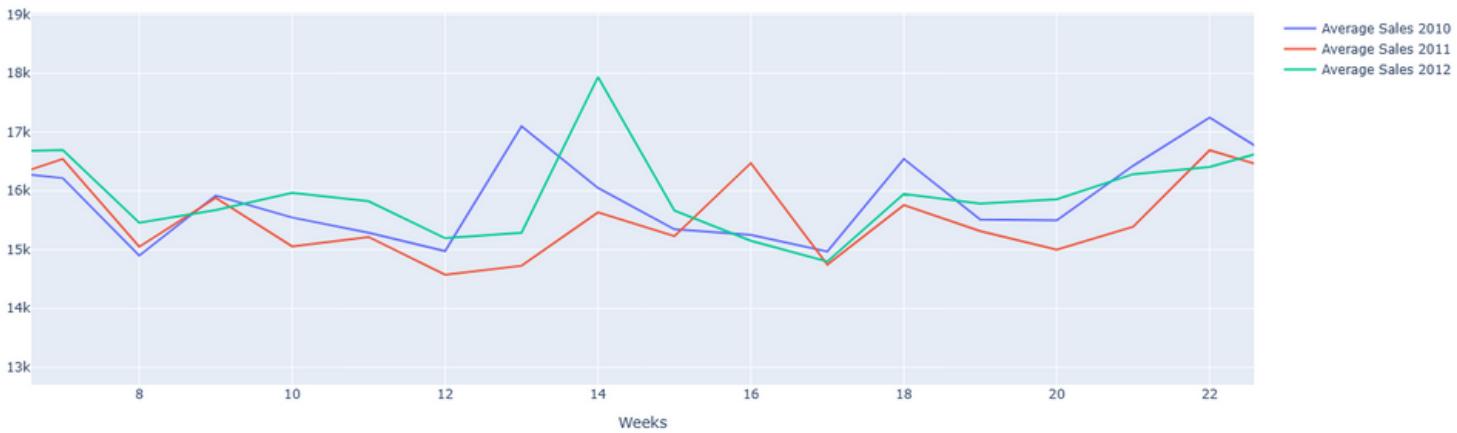
Sales 2010, 2011, 2012



Trên cơ sở hàng tuần, tuần lễ Tạ ơn và một tuần trước Giáng sinh chứng kiến doanh số bán hàng cao nhất trong các năm 2010 và 2011.

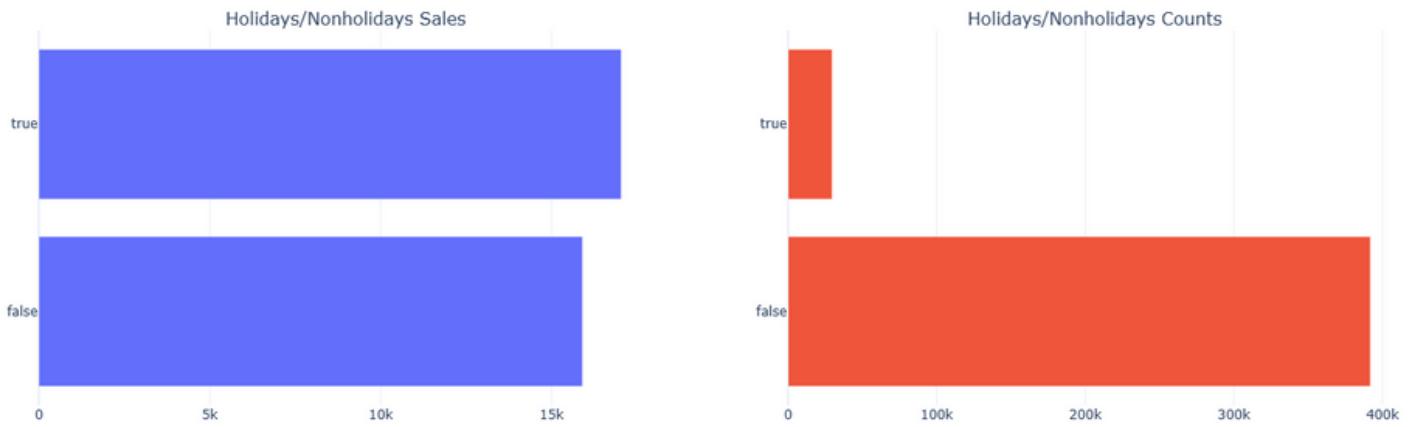
→ Có một mô hình rõ ràng về doanh số bán hàng trong các năm, vào Lễ Tạ ơn và Giáng sinh, doanh số bán hàng tăng lên một mức rất lớn.

Sales 2010, 2011, 2012



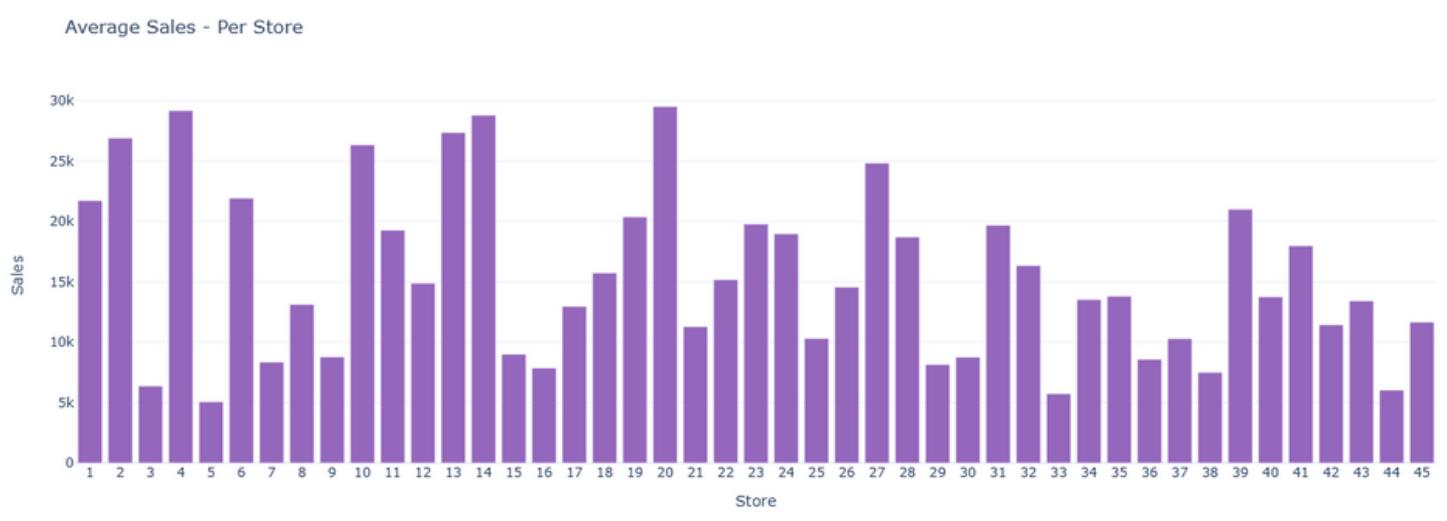
Trong năm 2012, tuần thứ 14 có doanh thu cao nhất so với các tuần khác trong năm nhưng trong tuần đó không có trùng với bất kỳ ngày lễ hay sự kiện đặc biệt nào.

HOLIDAYS VS NONHOLIDAYS SALES



- Chỉ có 7% số các tuần trong dữ liệu là những tuần nghỉ lễ
- Mặc dù tỷ lệ các tuần nghỉ lễ ít hơn nhưng doanh số bán hàng trong những tuần lễ trung bình cao hơn so với các tuần không nghỉ lễ

AVERAGE STORE SALES

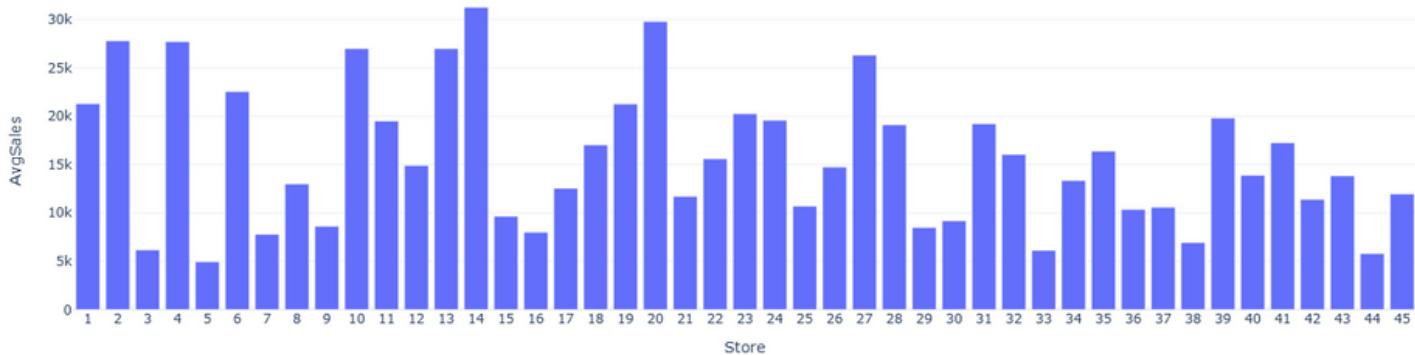


- Doanh số bán hàng có sự khác biệt lớn trong số 45 cửa hàng được thu thập.
- Doanh số có sự phụ thuộc vào loại cửa hàng và vào tuần nào của năm cụ thể đều được xem xét.

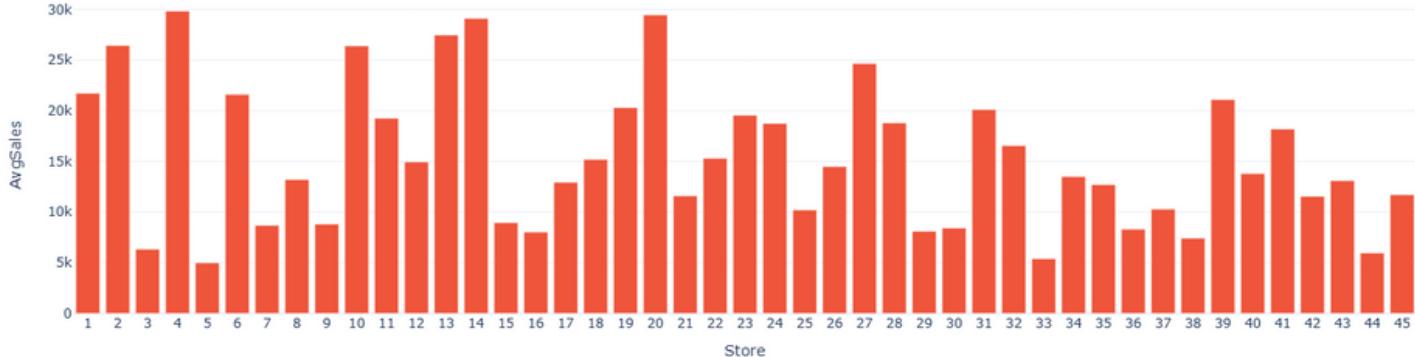
AVERAGE STORE SALES - YEAR WISE



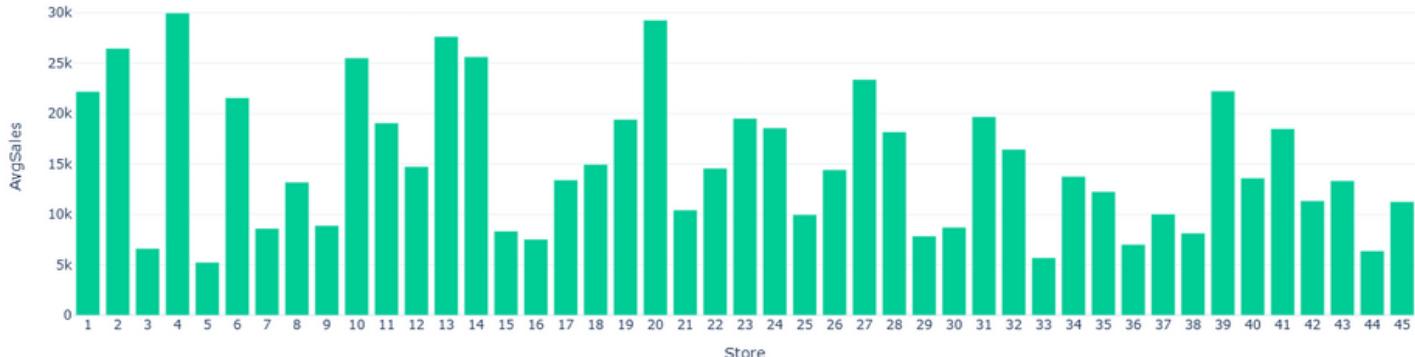
Average Store Sales 2010



Average Store Sales 2011



Average Store Sales 2012

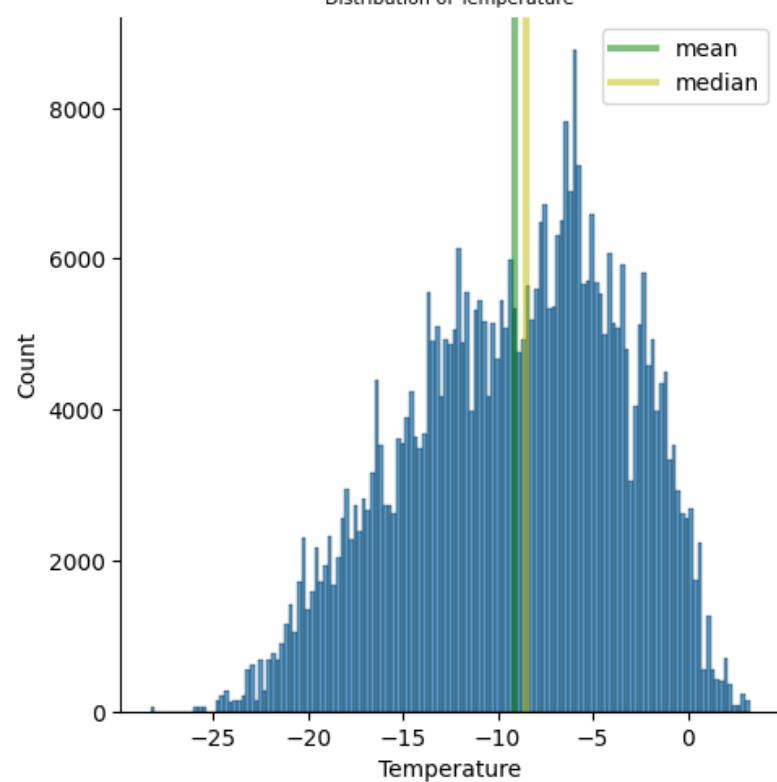


- Xu hướng chung về doanh số bán hàng của cửa hàng trong 3 năm vẫn không thay đổi vì nó phụ thuộc vào loại cửa hàng và quy mô cửa hàng
- Các cửa hàng 2,4,13,14 và 20 đạt doanh số cao nhất trong cả 3 năm

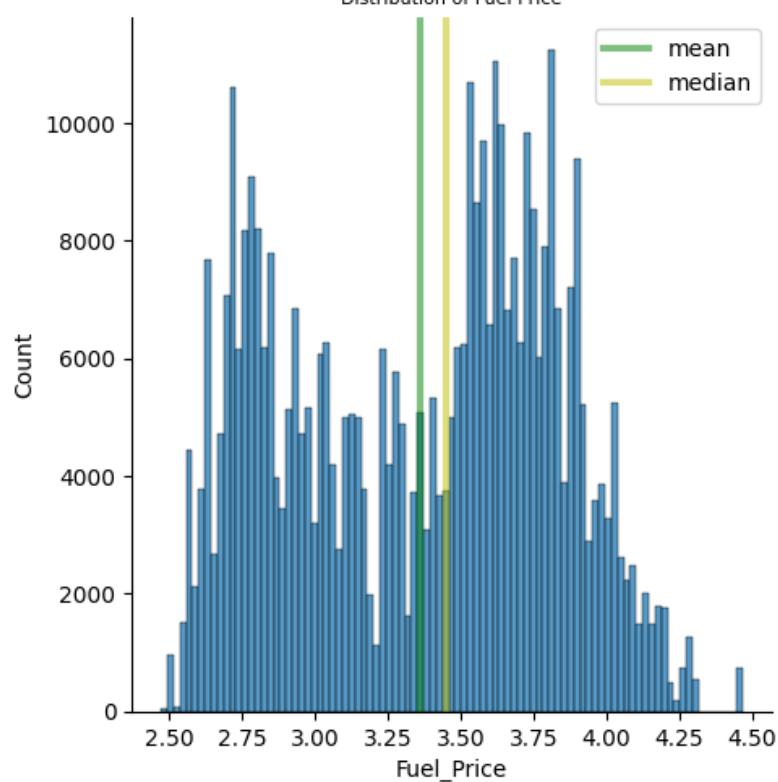
THE DISTRIBUTION



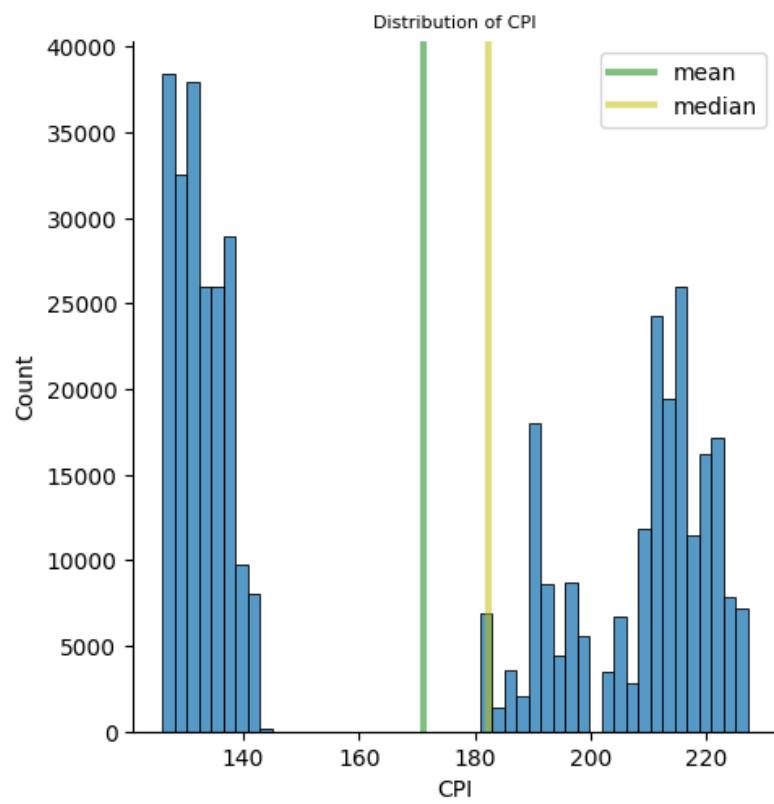
Distribution of Temperature



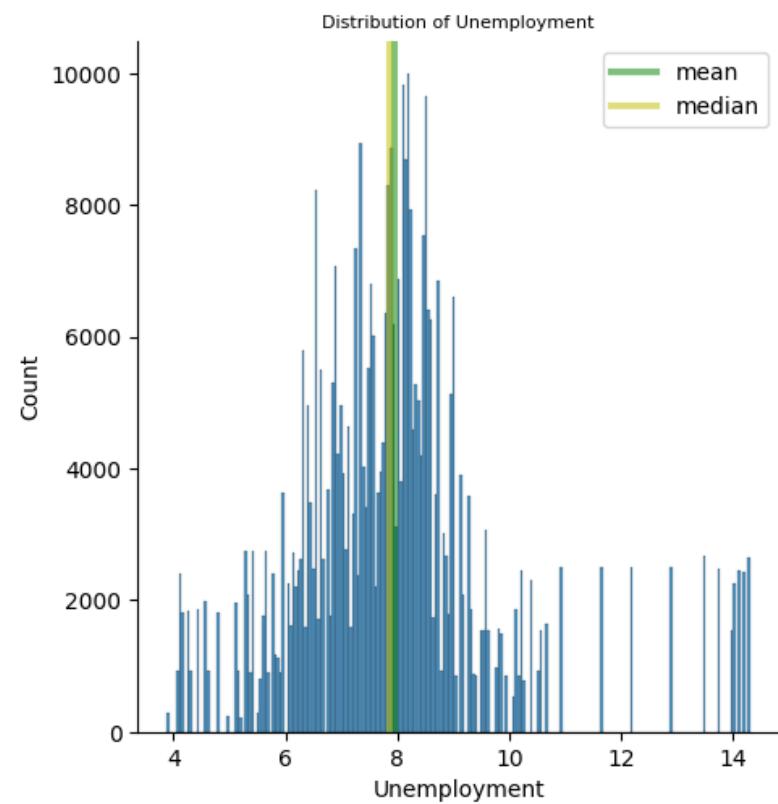
Distribution of Fuel Price



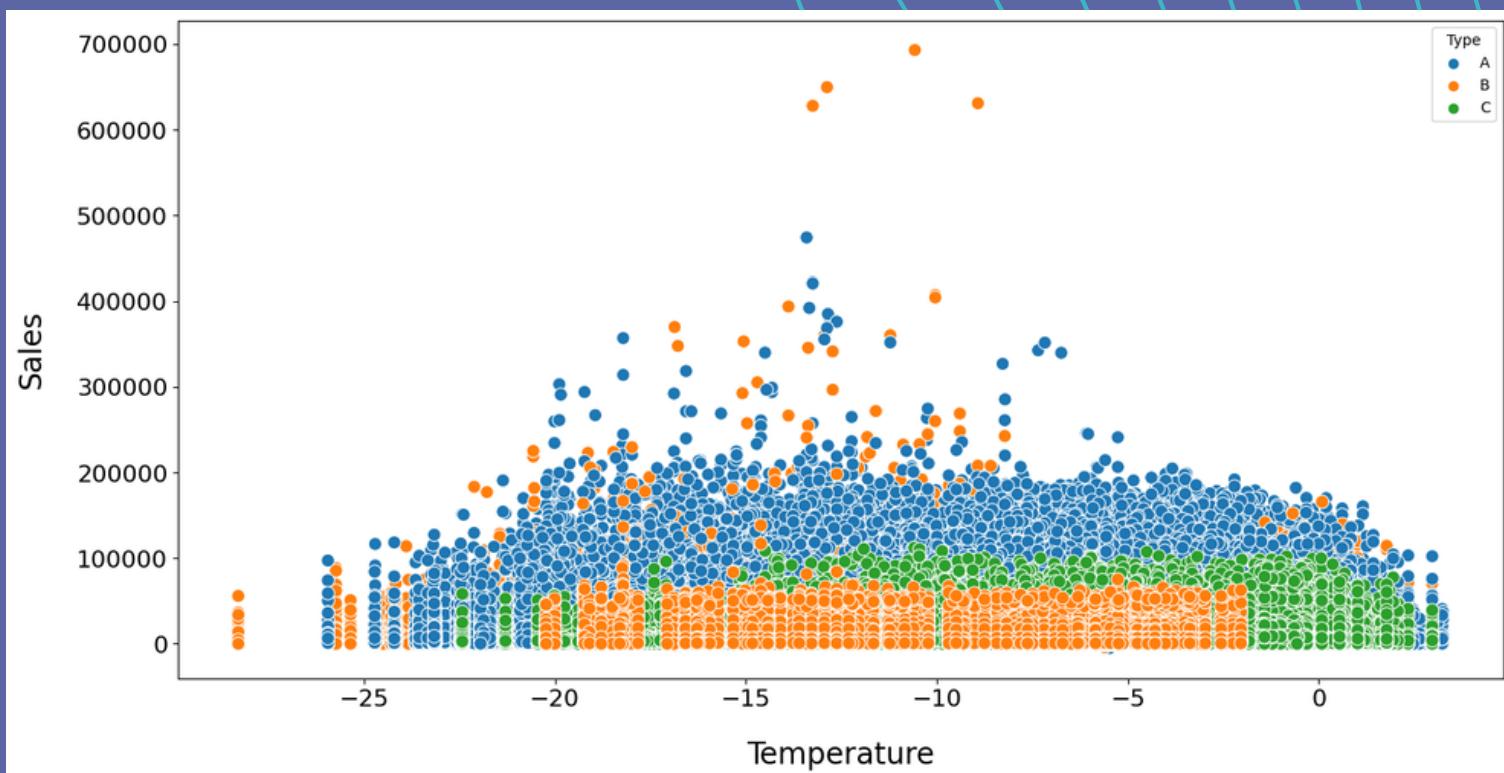
Distribution of CPI



Distribution of Unemployment



RELATIONSHIP: TEMPERATURE VS SALES

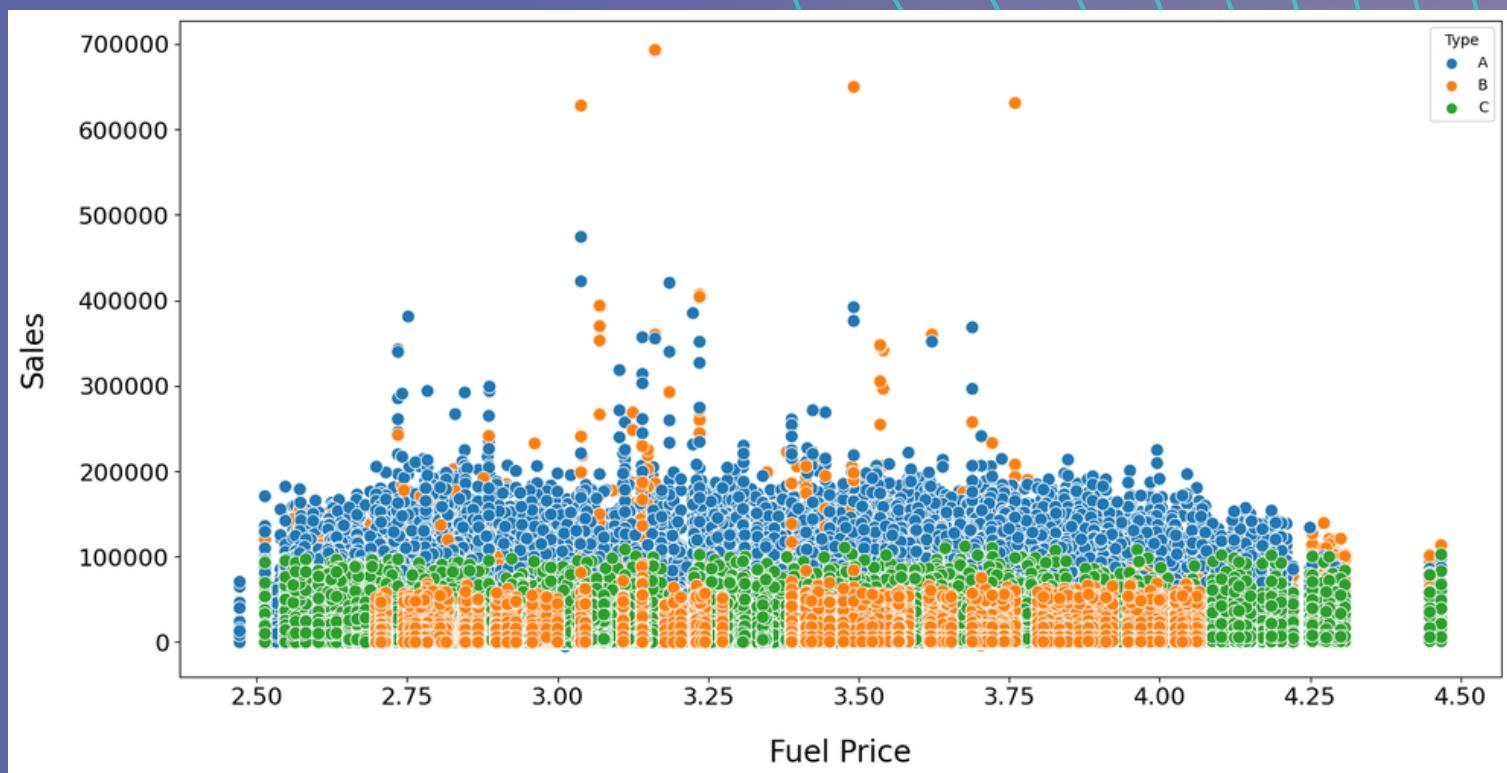


- Ở đây dường như không có mối quan hệ nào giữa nhiệt độ trong khu vực và doanh thu hàng tuần của các cửa hàng. Ở nhiệt độ thấp và rất cao, doanh số bán hàng dường như giảm một chút nhưng nhìn chung không tồn tại mối quan hệ rõ ràng



- Ta có thể biết được là có một mô hình giữa nền nhiệt lạnh với doanh số bán hàng, rõ ràng là liên quan đến thực tế là Hoa Kỳ nằm ở bán cầu bắc và đây là quốc gia mà phần lớn diện tích của nó phải trải qua nền nhiệt độ thấp vào thời điểm này.

RELATIONSHIP: FUEL PRICE VS SALES

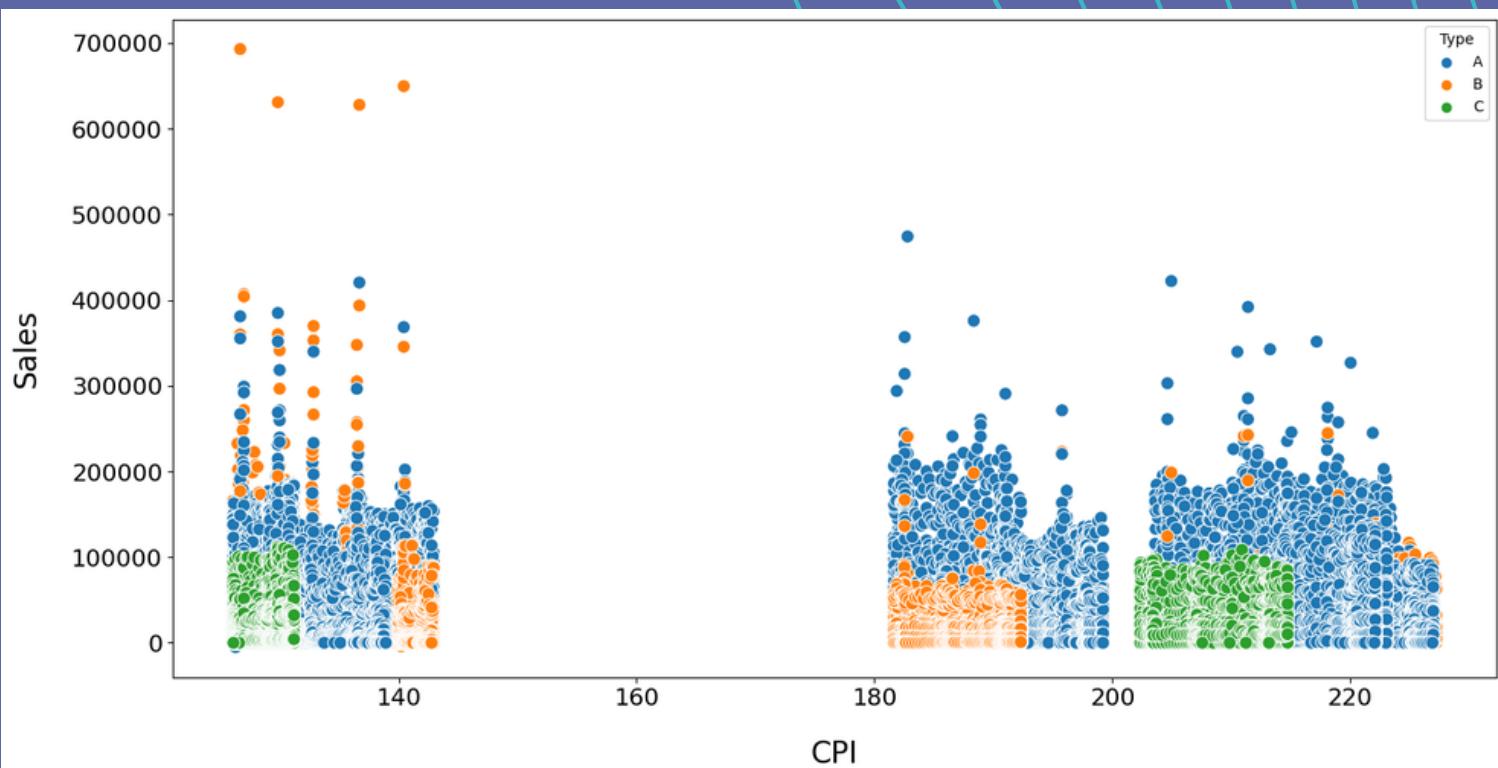


- Giữa giá xăng dầu và doanh số đường như cũng không tồn tại mối quan hệ rõ ràng nào

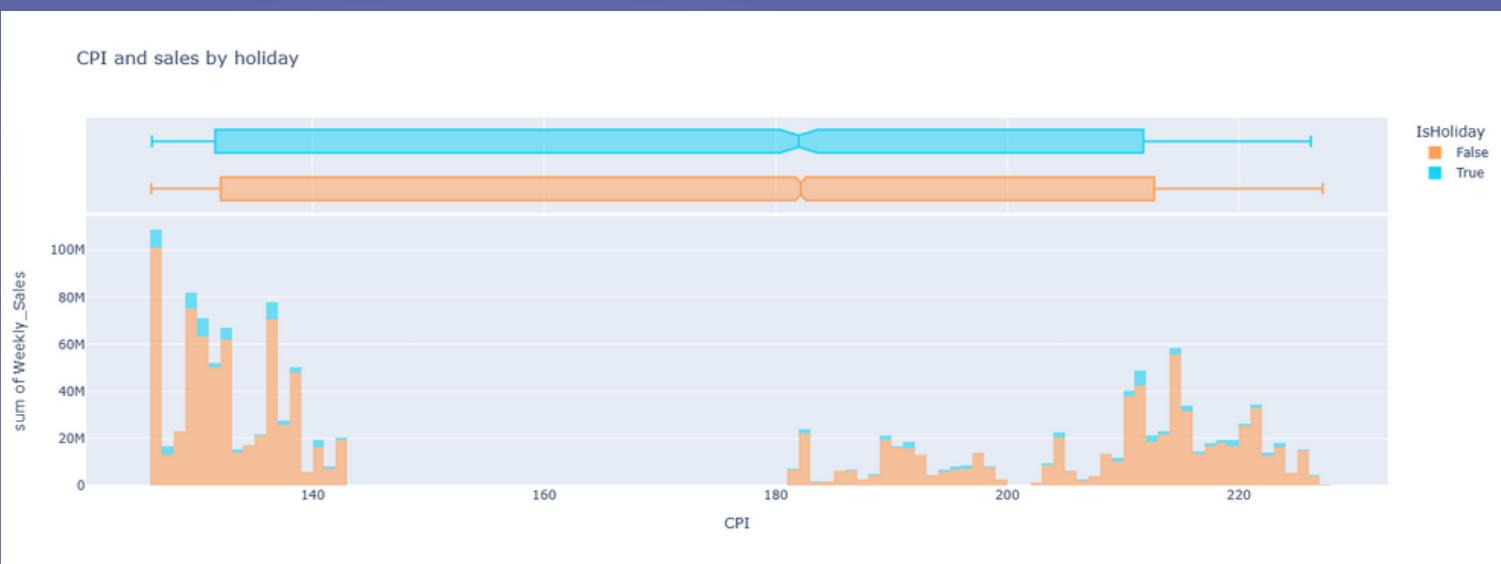


- Không có một mô hình rõ ràng nào ở đây, nhưng ta có thể thấy rằng giá nhiên liệu càng thấp thì doanh số bán hàng càng nhiều.

RELATIONSHIP: CPI VS SALES

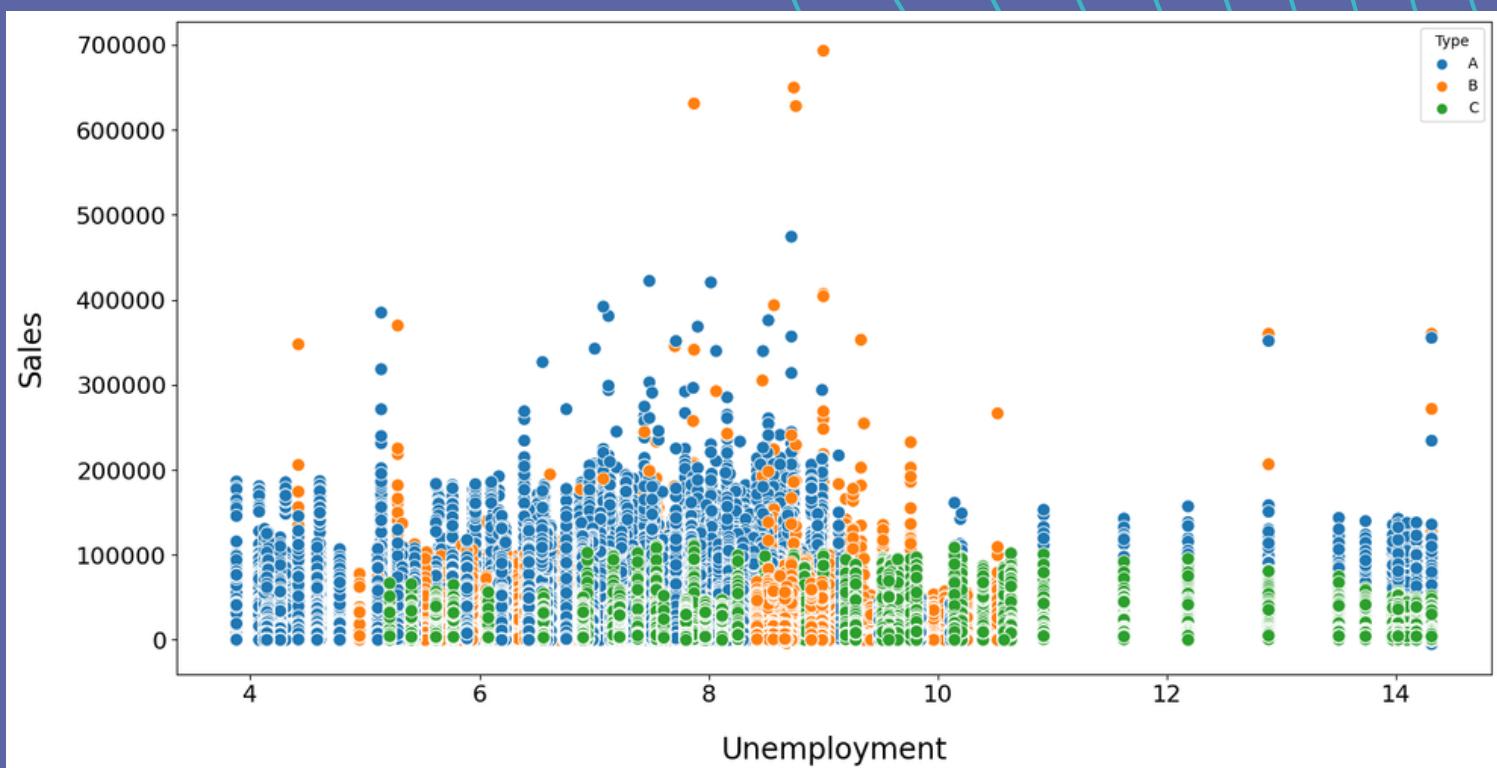


- Có 3 cụm phân bố rõ ràng nhưng lại không tồn tại bất kỳ mối tương quan rõ ràng nào giữa CPI và doanh số hàng tuần



- Ở đây cũng không có một khuôn mẫu rõ ràng cho lắm, ta có thể thấy có 3 nhóm nhưng nhóm nào cũng có doanh số bán ra, mặc dù thực tế là CPI cao hơn.

RELATIONSHIP: UNEMPLOYMENT VS SALES



- Theo biểu đồ tỷ lệ thất nghiệp dường như không có bất kỳ ảnh hưởng nào đến doanh số hàng tuần

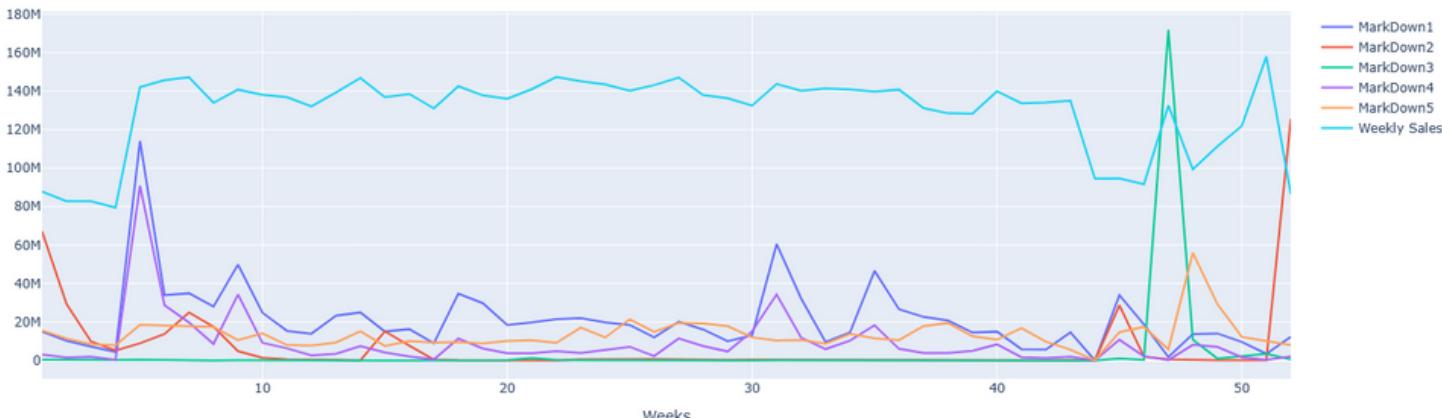


- Liên quan đến tỷ lệ thất nghiệp, có thể thấy rằng giá trị càng thấp, doanh số càng cao, điều đó cũng có lý.

RELATIONSHIP: MARKDOWNS VS SALES



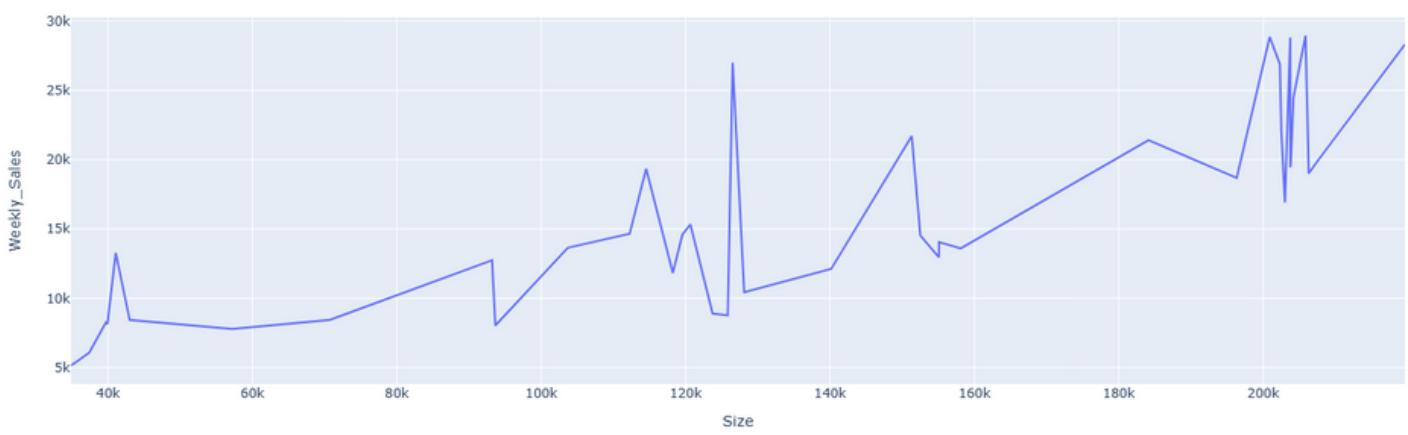
Sales vs Markdown's



- Các cột Markdown từ 1-5 chứa rất nhiều giá trị bị thiếu với hơn 250.000 giá trị NaN trong mỗi cột Markdown. Những cột này tương ứng với các hoạt động khuyến mãi đang được tiến hành tại các cửa hàng khác nhau. Các chiết khấu khuyến mãi chỉ bắt đầu sau tháng 11 năm 2011 và không diễn ra liên tục tại tất cả các cửa hàng. Do đó, việc có nhiều giá trị NaN trong các cột này là hợp lý.
- Những chương trình giảm giá hàng tuần sẽ có được số doanh số lớn và trải đều trong năm

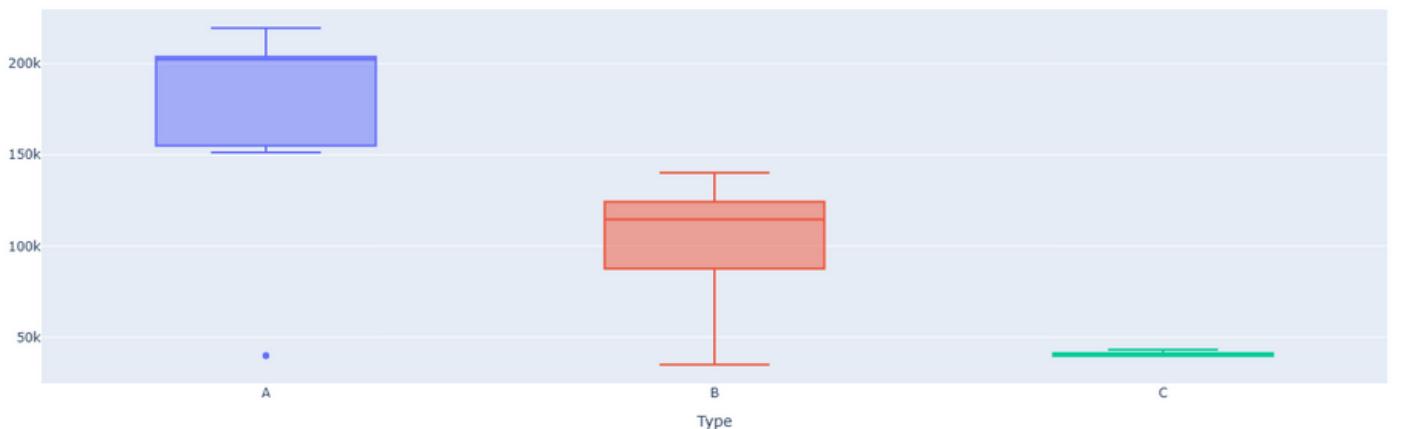
OTHER FEATURES ANALYSIS

Store size and sales



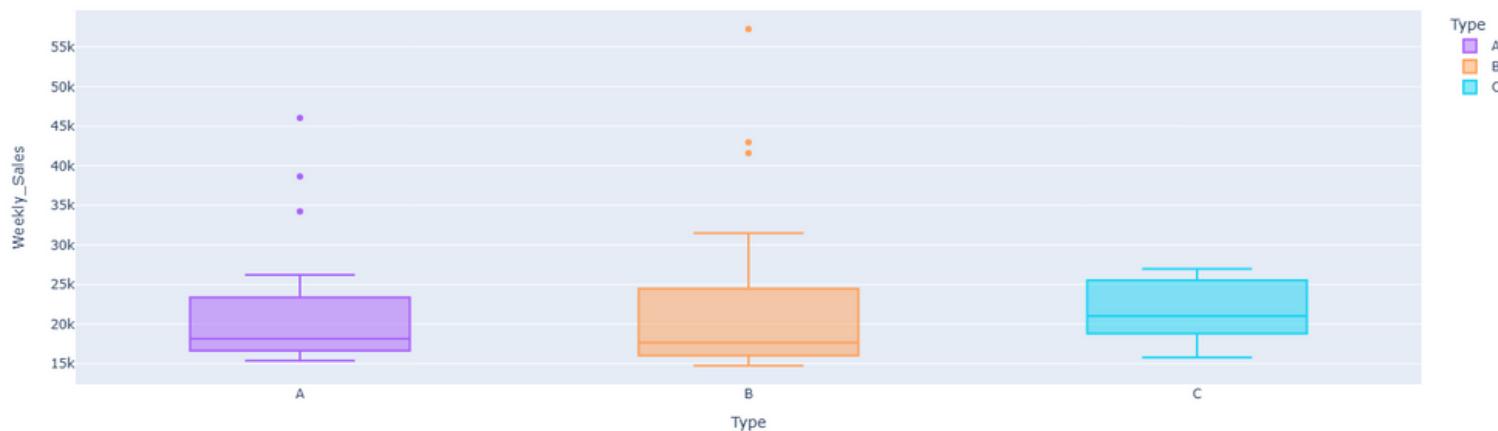
- Kích thước cũng là một yếu tố quan trọng khi bán hàng, như có thể thấy ở đây

Store size and Store type



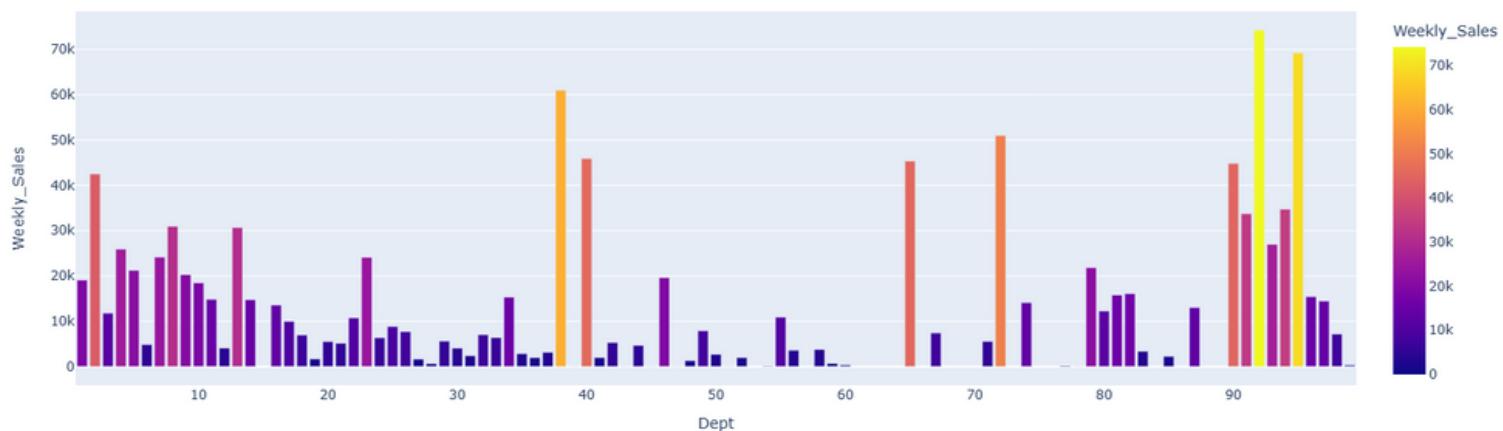
- Trong phạm vi về kích thước, chúng ta có thể thấy rằng có 3 loại cửa hàng, loại A là loại phổ biến nhất.

Store type and sales



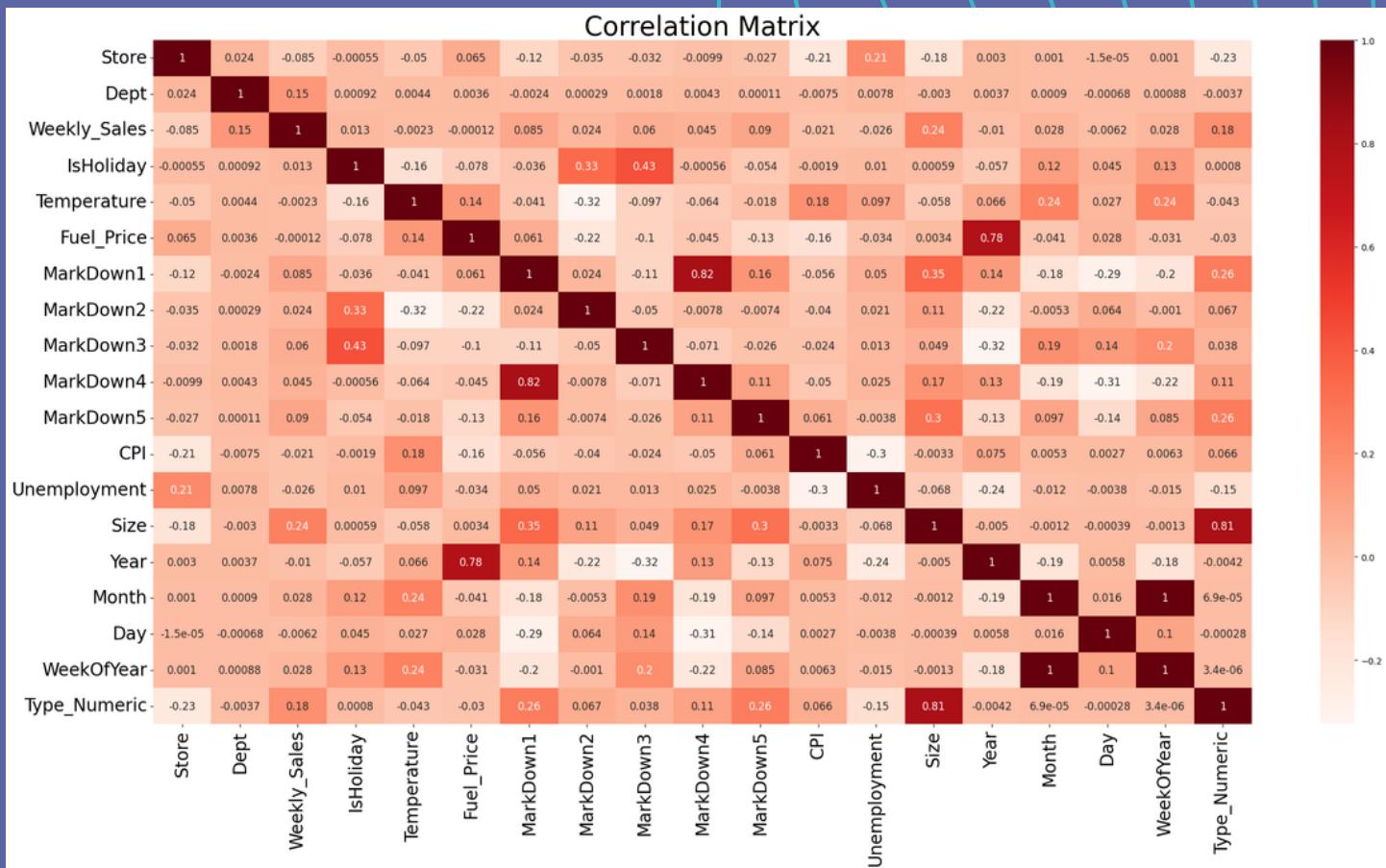
- Liên quan đến loại cửa hàng, chúng ta có thể thấy rằng mặc dù loại C có kích thước nhỏ nhất, nhưng lại là những cửa hàng có doanh thu trung bình cao nhất.

Department and sales

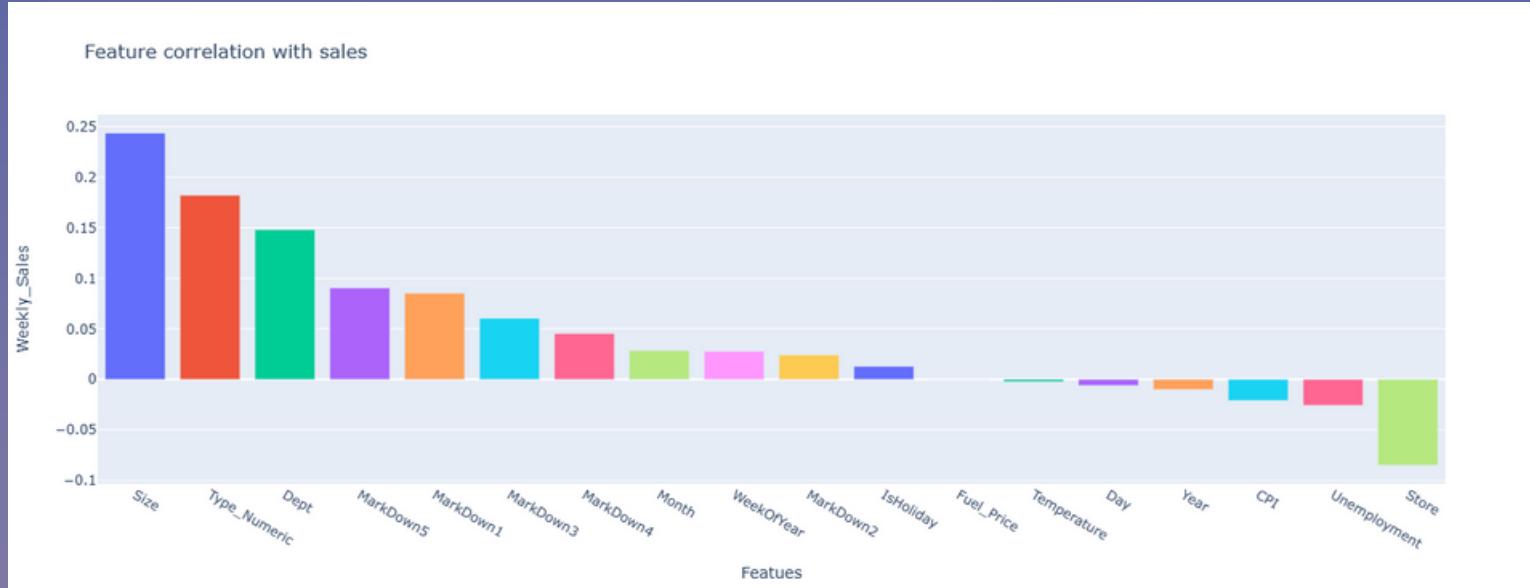


- Một số bộ phận đóng góp vào doanh số bán hàng nhiều hơn các bộ phận khác.

HEATMAP AND CORRELATION BETWEEN FEATURES



- Phần "Department" (Bộ phận), "Store size" (Kích thước cửa hàng) và "Type" (Loại) có mức tương quan cao với doanh số hàng tuần.
- Cột "Temperature" (Nhiệt độ), "Fuel price" (Giá nhiên liệu), "CPI" (Chỉ số giá tiêu dùng) và "Unemployment" (Tỉ lệ thất nghiệp) có mức tương quan rất yếu với doanh số hàng tuần.
- Các cột "Markdown1-5" có mức tương quan rất yếu với doanh số hàng tuần, vì vậy chúng ta sẽ bỏ qua các cột này.
- Chúng ta cũng sẽ bỏ qua các cột "Month" (Tháng) và "Day" (Ngày) vì thông tin này đã được chứa trong "WeekOfYear" (Tuần trong năm).



6.1 MODELING - SAI LẦN THỨ N

```
[ ] import numpy as np
    import pandas as pd

[ ] data = pd.read_csv(r"D:\data_clean.csv")

[ ] from sklearn.ensemble import RandomForestRegressor
    import matplotlib.pyplot as plt

[ ] Features=data.drop(['Weekly_Sales'],axis=1)
    Target=data['Weekly_Sales']
```



Target

```
0      24924.50
1      46039.49
2      41595.55
3      19403.54
4      21827.90
...
420207    508.37
420208    628.10
420209    1061.02
420210    760.01
420211    1076.80
Name: Weekly_Sales, Length: 420212, dtype: float64
```

```
rf = RandomForestRegressor(n_estimators=100)
rf.fit(Features,Target)
```

▼ RandomForestRegressor

RandomForestRegressor()

	Store	Dept	IsHoliday	Temperature	Fuel_Price	CPI	Unemployment	Type	Size	year	month	week
0	1	1	0	42.31	2.572	211.096358	8.106	0	151315	2010	2	5
1	1	1	1	38.51	2.548	211.242170	8.106	0	151315	2010	2	6
2	1	1	0	39.93	2.514	211.289143	8.106	0	151315	2010	2	7
3	1	1	0	46.63	2.561	211.319643	8.106	0	151315	2010	2	8
4	1	1	0	46.50	2.625	211.350143	8.106	0	151315	2010	3	9
...
420207	45	98	0	64.88	3.997	192.013558	8.684	1	118221	2012	9	39
420208	45	98	0	64.89	3.985	192.170412	8.667	1	118221	2012	10	40
420209	45	98	0	54.47	4.000	192.327265	8.667	1	118221	2012	10	41
420210	45	98	0	56.47	3.969	192.330854	8.667	1	118221	2012	10	42
420211	45	98	0	58.85	3.882	192.308899	8.667	1	118221	2012	10	43

420212 rows × 12 columns

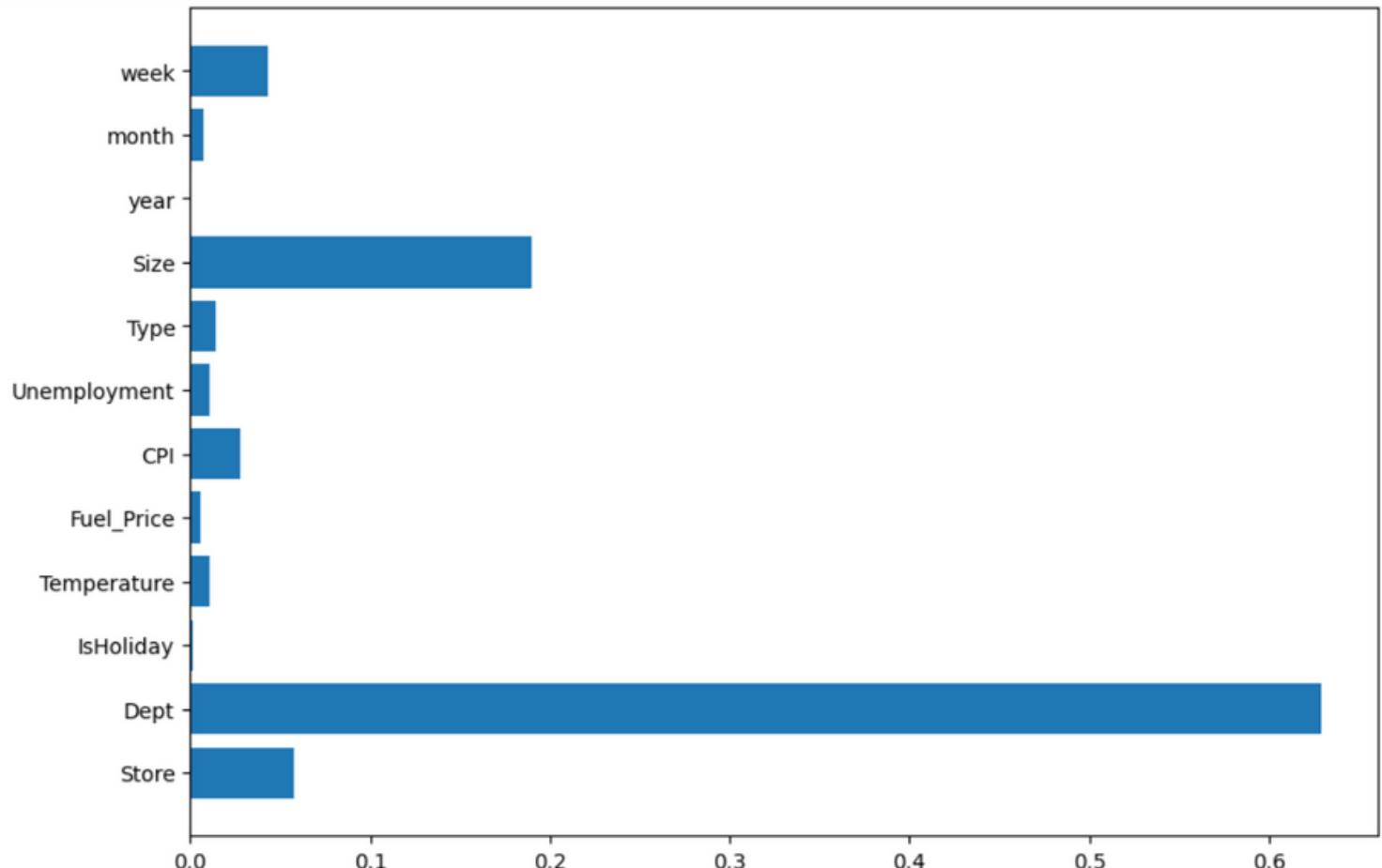
- Clean data xong , so sánh thì thấy bị thiếu mất 1 đống data, so với bản hoàn chỉnh sau đó ,nhưng em bất chấp dùng nó để làm model :)))
- Đây là bản clean hoàn chỉnh sau đó của bọn em

Unnamed: 0	Store	Dept	Date	Weekly_Sales	IsHoliday	Temperature	Fuel_Price	MarkDown1	MarkDown2	...	MarkDown4	MarkDown5	CPI	Unemployment	Type	Size	Year
0	0	1	1 2010-02-05	24924.50	0	42.31	2.572	0.00	0.00	...	0.00	0.00	211.096358	8.106	1	151315	2010
1	1	1	1 2010-02-12	46039.49	1	38.51	2.548	0.00	0.00	...	0.00	0.00	211.242170	8.106	1	151315	2010
2	2	1	1 2010-02-19	41595.55	0	39.93	2.514	0.00	0.00	...	0.00	0.00	211.289143	8.106	1	151315	2010
3	3	1	1 2010-02-26	19403.54	0	46.63	2.561	0.00	0.00	...	0.00	0.00	211.319643	8.106	1	151315	2010
4	4	1	1 2010-03-05	21827.90	0	46.50	2.625	0.00	0.00	...	0.00	0.00	211.350143	8.106	1	151315	2010
...
421565	421565	45	98 2012-09-28	508.37	0	64.88	3.997	4556.61	20.64	...	1601.01	3288.25	192.013558	8.684	2	118221	2012
421566	421566	45	98 2012-10-05	628.10	0	64.89	3.985	5046.74	0.00	...	2253.43	2340.01	192.170412	8.667	2	118221	2012
421567	421567	45	98 2012-10-12	1061.02	0	54.47	4.000	1956.28	0.00	...	599.32	3990.54	192.327265	8.667	2	118221	2012
421568	421568	45	98 2012-10-19	760.01	0	56.47	3.969	2004.02	0.00	...	437.73	1537.49	192.330854	8.667	2	118221	2012
421569	421569	45	98 2012-10-26	1076.80	0	58.85	3.882	4018.91	58.08	...	211.94	858.33	192.308899	8.667	2	118221	2012

421570 rows × 21 columns

MODELINNG BẤT CHẤP

- Vẽ biểu đồ để so sánh độ quan trọng của các feature



LINEAREGRESSION

- Đoạn này e xóa 1 vài cột không quan trọng đi để làm model nhưng không hiểu sao cột size với week to đúng thế kia nhưng vẫn xóa đi :)))

Linear Regression

```
[ ] x = data.drop(['Weekly_Sales', 'Size', 'IsHoliday', 'year', 'month', 'week'], axis = 1)
y = data['Weekly_Sales']

[ ] x.columns
Index(['Store', 'Dept', 'Temperature', 'Fuel_Price', 'CPI', 'Unemployment',
       'Type'],
      dtype='object')

[ ] y.head()
0    24924.50
1    46039.49
2    41595.55
3    19403.54
4    21827.90
Name: Weekly_Sales, dtype: float64

[ ] from sklearn.model_selection import train_test_split

x_train, x_val, y_train, y_val = train_test_split(x, y, test_size = 0.3, random_state = 10)
```

- Em sử dụng hàm train_test_split từ sklearn.model_selection để chia dữ liệu thành tập huấn luyện và tập validation. Dữ liệu đặc trưng (x) và mục tiêu (y) được chia thành bốn phần: x_train, x_val, y_train, và y_val. Tỉ lệ chia tập validation là 30%, được xác định bằng tham số test_size=0.3. Đồng thời, để đảm bảo sự nhất quán, giá trị random_state được đặt là 10. Kết quả là ta có tập dữ liệu huấn luyện (x_train, y_train) và tập dữ liệu validation (x_val, y_val) để sử dụng trong quá trình đào tạo và đánh giá mô hình.

```
[ ] from sklearn.linear_model import LinearRegression
lr = LinearRegression()

lr.fit(x_train, y_train)
y_pred = lr.predict(x_val)
```

▶ lr.score(x_val, y_val)

```
0.05794610101444586
```

```
[ ] from sklearn.metrics import mean_squared_error, r2_score
mse = mean_squared_error(y_pred, y_val)
r2 = r2_score(y_pred, y_val)
print('Mean Square Error = ', mse)
print('R2 Score = ', r2)
```

```
Mean Square Error =  479286745.4245957
R2 Score = -14.323014131318908
```

- Em sử dụng mô hình Linear Regression để huấn luyện trên dữ liệu huấn luyện (x_train và y_train). Sau đó, em sử dụng mô hình đã huấn luyện để dự đoán giá trị trên tập validation (x_val) và tính toán MSE (Mean Squared Error) và R2 Score để đánh giá hiệu suất của mô hình trên tập validation.
- Kết quả MSE và R2 Score được in ra màn hình để cung cấp thông tin về độ chính xác và hiệu suất của mô hình trên tập validation.

DECISION TREE REGRESSION

```
[ ] from sklearn.tree import DecisionTreeRegressor  
  
[ ] dt = DecisionTreeRegressor()  
dt_model = dt.fit(x_train, y_train)  
y_pred_dt = dt_model.predict(x_val)  
  
[ ] rms_dt = np.sqrt(mean_squared_error(y_pred_dt, y_val))  
r2_dt = r2_score(y_pred_dt, y_val)  
print('RMSE of DT = ', rms_dt)  
print('R2 Score of DT = ', r2_dt)  
  
RMSE of DT = 7505.389009546623  
R2 Score of DT = 0.8903981694619187
```

- Em sử dụng mô hình Decision Tree Regressor (DecisionTreeRegressor) để huấn luyện trên dữ liệu huấn luyện (x_{train} và y_{train}). Sau đó, ta sử dụng mô hình đã huấn luyện để dự đoán giá trị trên tập validation (x_{val}) và tính toán RMSE (Root Mean Squared Error) và R2 Score để đánh giá hiệu suất của mô hình trên tập validation.
- Kết quả RMSE và R2 Score được in ra màn hình để cung cấp thông tin về độ chính xác và hiệu suất của mô hình Decision Tree trên tập validation.

RANDOM FOREST REGRESSION

```
from sklearn.ensemble import RandomForestRegressor
```

```
rf = RandomForestRegressor()  
rf_model = rf.fit(x_train, y_train)  
y_pred_rf = rf_model.predict(x_val)
```

```
rms_rf = np.sqrt(mean_squared_error(y_pred_rf, y_val))  
r2_rf = r2_score(y_pred_rf, y_val)  
print('RMSE of RF = ', rms_rf)  
print('R2 Score of RF = ', r2_rf)
```

```
RMSE of RF =  5694.180282880063  
R2 Score of RF =  0.9331227217005335
```

- Em sử dụng mô hình RandomForestRegressor (RandomForestRegressor) để huấn luyện trên dữ liệu huấn luyện (x_train và y_train). Sau đó, chúng ta sử dụng mô hình đã huấn luyện để dự đoán giá trị trên tập validation (x_val) và tính toán RMSE (Root Mean Squared Error) và R2 Score để đánh giá hiệu suất của mô hình trên tập validation.
- Kết quả RMSE và R2 Score được in ra màn hình để cung cấp thông tin về độ chính xác và hiệu suất của mô hình RandomForest trên tập validation.

SO SÁNH GIỮA 3 MODEL

```
import matplotlib.pyplot as plt

models = ['Linear Regression', 'Decision Tree', 'Random Forest']
rmse_scores = [mse, rms_dt, rms_rf]
r2_scores = [r2, r2_dt, r2_rf]

# Vẽ biểu đồ RMSE
plt.figure(figsize=(10, 6))
plt.bar(models, rmse_scores)
plt.xlabel('Models')
plt.ylabel('RMSE')
plt.title('RMSE Comparison')

# Hiển thị giá trị RMSE trên cột
for i in range(len(models)):
    plt.text(i, rmse_scores[i], str(round(rmse_scores[i], 2)), ha='center', va='bottom')

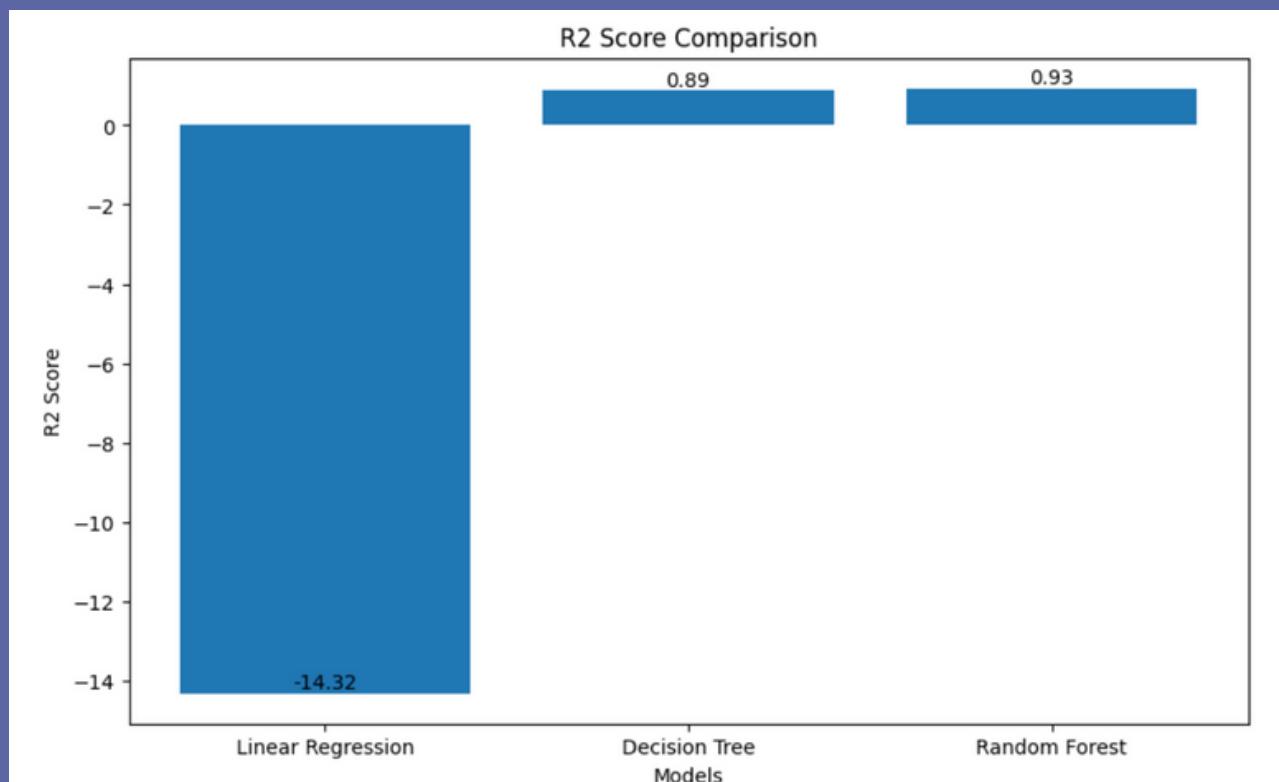
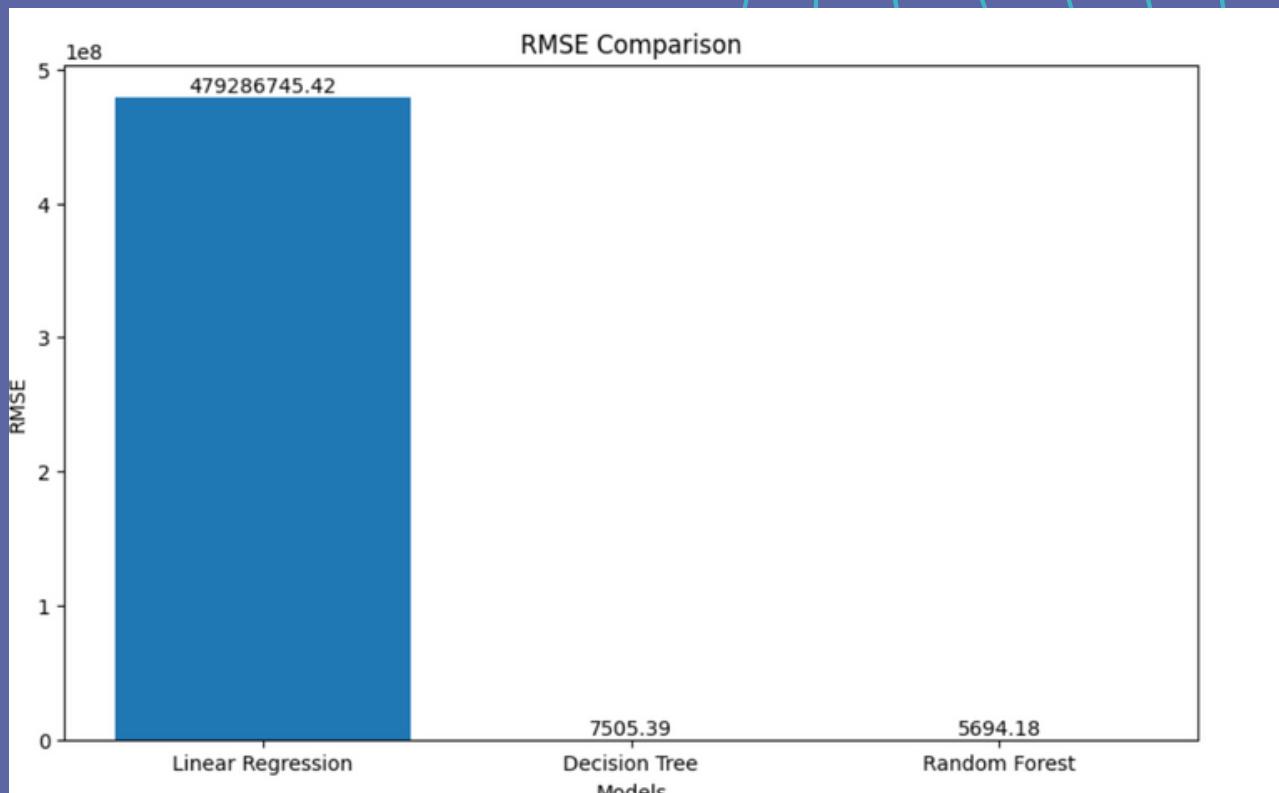
plt.show()

# Vẽ biểu đồ R2
plt.figure(figsize=(10, 6))
plt.bar(models, r2_scores)
plt.xlabel('Models')
plt.ylabel('R2 Score')
plt.title('R2 Score Comparison')

# Hiển thị giá trị R2 trên cột
for i in range(len(models)):
    plt.text(i, r2_scores[i], str(round(r2_scores[i], 2)), ha='center', va='bottom')

plt.show()
```

- Các giá trị RMSE và R2 Score đã tính toán từ các mô hình trước đó được lưu trong các danh sách rmse_scores và r2_scores tương ứng.
- Vẽ biểu đồ RMSE bằng cách sử dụng plt.bar và truyền vào các thông số models và rmse_scores.
- Vẽ biểu đồ R2 Score tương tự như RMSE



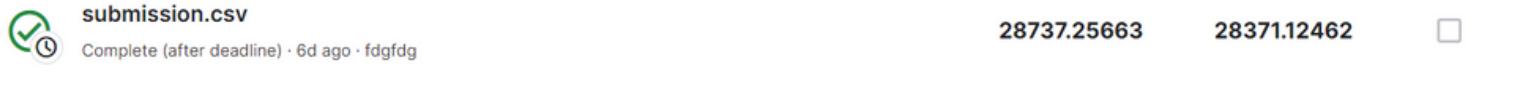
- Em quyết định sử dụng kết quả dự đoán của model RandomForest để làm kết quả cuối cùng

- Sau khi xử lí xong thì em tạo file submission :

```
[ ] submission['Weekly_Sales'] = prediction
```

```
[ ] submission.to_csv('submission1.csv', index=False)
```

- Và đây là kết quả từ model của em



- Submit lên và thấy gần như là kết quả không đúng 1 chút nào bởi vì điểm kaggle cho càng cao thì kết quả sai càng nhiều



- Trong bài báo cáo này, chúng em đã mở rộng cơ sở dữ liệu bằng cách thêm hai tính năng mới, bao gồm "tổng số lượng bán" (total sell) và "số lượng bán trung bình" (average sell), nhằm cung cấp thông tin hữu ích cho việc dự đoán giá cả.

```
# Tính total sales cho mỗi (year, month)
data['Total_Sales'] = data.groupby(['year', 'month'])['Weekly_Sales'].transform('sum')
print(data['Total_Sales'])

0      1.903360e+08
1      1.903360e+08
2      1.903360e+08
3      1.903360e+08
4      1.819236e+08
...
420207  1.806480e+08
420208  1.843628e+08
420209  1.843628e+08
420210  1.843628e+08
420211  1.843628e+08
Name: Total_Sales, Length: 420212, dtype: float64
```

```
#Tính average sells
data['Average_Sell'] = data.groupby(['year', 'month'])['Weekly_Sales'].transform('mean')
print(data['Average_Sell'])

0      16123.339183
1      16123.339183
2      16123.339183
3      16123.339183
4      15480.222209
...
420207  15283.248527
420208  15567.236930
420209  15567.236930
420210  15567.236930
420211  15567.236930
Name: Average_Sell, Length: 420212, dtype: float64
```



- Sau đó, chúng em đã triển khai hai mô hình mới để dự đoán giá cả, bao gồm XGBoost Regression và Gradient Boost Regression.
- Cả hai mô hình đều có khả năng xử lý dữ liệu phức tạp và tạo ra dự đoán chính xác

```
from sklearn.ensemble import GradientBoostingRegressor
# Khởi tạo và huấn luyện mô hình Gradient Boosting Regression
gb = GradientBoostingRegressor()
gb.fit(x_train, y_train)
```

```
▼ GradientBoostingRegressor
GradientBoostingRegressor()
```

```
# Khởi tạo và huấn luyện mô hình XGBoost Regression
xgb = XGBRegressor()
xgb.fit(x_train, y_train)
```

```
▼ XGBRegressor
XGBRegressor(base_score=None, booster=None, callbacks=None,
             colsample_bylevel=None, colsample_bynode=None,
             colsample_bytree=None, early_stopping_rounds=None,
             enable_categorical=False, eval_metric=None, feature_types=None,
             gamma=None, gpu_id=None, grow_policy=None, importance_type=None,
             interaction_constraints=None, learning_rate=None, max_bin=None,
             max_cat_threshold=None, max_cat_to_onehot=None,
             max_delta_step=None, max_depth=None, max_leaves=None,
             min_child_weight=None, missing=nan, monotone_constraints=None,
             n_estimators=100, n_jobs=None, num_parallel_tree=None,
             predictor=None, random_state=None, ...)
```



- Sau khi thực hiện quá trình cross-validation cho tất cả các mô hình, chúng em đã đưa ra hai kết luận quan trọng

```
models = [('Linear Regression', lr), ('Decision Tree Regression', dt),
          ('Gradient Boosting Regression', gbr), ('XGBoost Regression', xgb)]
for model_name, model in models:
    scores = cross_val_score(model, x, y, cv=5, scoring='neg_mean_squared_error')
    mse_scores = -scores
    mean_mse_cv = np.mean(mse_scores)
    print(model_name + ' Mean Square Error:', mean_mse_cv)

Linear Regression Mean Square Error: 505797699.9903086
Decision Tree Regression Mean Square Error: 363772293.91352737
Gradient Boosting Regression Mean Square Error: 211993848.21874815
XGBoost Regression Mean Square Error: 193581681.82906988
```

- Mô hình Random Forest không đảm bảo thời gian huấn luyện để đưa ra kết quả dự đoán.
- XGBoost (XGB) là mô hình có chỉ số sai số bình phương trung bình (MSE) tốt nhất.
- => XGBoost có khả năng ứng dụng tốt và đưa ra dự đoán chính xác hơn các mô hình khác





- Sau đó, chúng em quyết định đo thời gian chạy cụ thể của các mô hình để đạt được sự cân bằng giữa tốc độ xử lý và hiệu quả dự đoán.

```
# Đo thời gian xử lý của mô hình Linear Regression
start_time = time.time()
lr.fit(x_train, y_train)
end_time = time.time()
execution_time_lr = end_time - start_time
```

```
# Đo thời gian xử lý của mô hình Decision Tree Regression
start_time = time.time()
dt.fit(x_train, y_train)
end_time = time.time()
execution_time_dt = end_time - start_time
```

```
# Đo thời gian xử lý của mô hình Gradient Boosting Regression
start_time = time.time()
gb.fit(x_train, y_train)
end_time = time.time()
execution_time_gb = end_time - start_time
```

```
# Đo thời gian xử lý của mô hình XGBoost Regression
start_time = time.time()
xgb.fit(x_train, y_train)
end_time = time.time()
execution_time_xgb = end_time - start_time
```



```
print("Execution Time (Linear Regression):", execution_time_lr, "seconds")
print("Execution Time (Decision Tree Regression):", execution_time_dt, "seconds")
print("Execution Time (Gradient Boosting Regression):", execution_time_gb, "seconds")
print("Execution Time (XGBoost Regression):", execution_time_xgb, "seconds")
```

```
Execution Time (Linear Regression): 0.08773469924926758 seconds
Execution Time (Decision Tree Regression): 2.4512083530426025 seconds
Execution Time (Gradient Boosting Regression): 37.00614309310913 seconds
Execution Time (XGBoost Regression): 11.936166286468506 seconds
```

- Dựa trên kết quả đo thời gian chạy của các mô hình, mặc dù mô hình XGBoost không phải là mô hình có tốc độ nhanh nhất trong số các mô hình được thử nghiệm, tuy nhiên khoảng cách về thời gian giữa XGBoost và các mô hình khác không quá lớn.
- Vì vậy, bằng cách xem xét cân bằng giữa tốc độ xử lý và hiệu suất dự đoán, chúng tôi kết luận rằng mô hình XGBoost là mô hình tối ưu nhất





- Các bước submit dự án lên kaggle

```
submission = pd.read_csv("/kaggle/input/sample-submission/sampleSubmission.csv")  
submission
```

	Id	Weekly_Sales
0	1_1_2012-11-02	0
1	1_1_2012-11-09	0
2	1_1_2012-11-16	0
3	1_1_2012-11-23	0
4	1_1_2012-11-30	0
...
115059	45_98_2013-06-28	0
115060	45_98_2013-07-05	0
115061	45_98_2013-07-12	0
115062	45_98_2013-07-19	0
115063	45_98_2013-07-26	0

115064 rows × 2 columns

```
# print(submission.head())  
# print(prediction[:5])  
prediction = prediction[:115064]  
submission['Weekly_Sales'] = prediction  
print(prediction)
```

```
[26975.969 36654.395 34417.977 ... 17215.266 16851.223 16358.039]
```

```
submission.to_csv('submission5.csv', index=False)
```



- Model trước khi được tối ưu hóa

submission.csv	Complete (after deadline) · 1h ago · fdgfdg	28737.25663	28371.12462	<input type="checkbox"/>
----------------	---	-------------	-------------	--------------------------



- Model sau khi được tối ưu hóa

submission5.csv	Complete (after deadline) · now · test	28569.45057	28230.6069	<input type="checkbox"/>
-----------------	--	-------------	------------	--------------------------

- Mặc dù kết quả chấm điểm cho thấy tiến bộ của mô hình, nhưng vẫn chưa đạt đủ mức độ mong muốn, do số lượng lỗi sai vẫn còn khá cao.

6.2 MODELING-HOÀN THIỆN

- Để chuẩn bị cho việc dự đoán thì em chia data thành 2 phần:
 - Phần 1 : tạo một danh sách gồm các tính năng từ cột dữ liệu trong data_train, trừ đi các cột có tên là 'Date' và 'Weekly_Sales' và sao đó copy , gán vào biến X
 - Phần 2: sao chép dữ liệu từ cột 'Weekly_Sales' của data_train và gán vào biến y.

```
[33] features = [feature for feature in data_train.columns  
|   |   |   if feature not in ('Date', 'Weekly_Sales')]
```

```
[34] X = data_train[features].copy()  
y = data_train.Weekly_Sales.copy()
```

- Kiểm tra kích thước của X và y để xác định xem có trùng khớp dữ liệu hay không

```
x.shape
```

```
✓ 0.0s
```

```
(421570, 19)
```

```
y.shape
```

```
✓ 0.0s
```

```
(421570, )
```

- Lấy 25% từ data gốc để tạo biến data_sample nhằm kích thước dữ liệu huấn luyện và giảm thời gian huấn luyện cho các mô hình.
- tạo ra biến X_sample bằng cách loại bỏ các cột 'Date' và 'Weekly_Sales' từ data_sample, và tạo ra biến y_sample từ cột 'Weekly_Sales' của data_sample

data sample

```
data_sample = data_train.copy().sample(frac=0.25)
X_sample = data_sample.drop(['Date', 'Weekly_Sales'], axis='columns').copy()
y_sample = data_sample.Weekly_Sales.copy()
```

- Kiểm tra kích thước của X_sample và y_sample để xác định xem có trùng khớp dữ liệu hay không

```
X_sample.shape
✓ 0.0s
(105392, 19)

y_sample.shape
✓ 0.0s
(105392,)
```

- Tiếp theo, ta sử dụng hàm `train_test_split` từ `sklearn.model_selection` để chia dữ liệu thành tập huấn luyện và tập kiểm tra. Tỷ lệ của tập kiểm tra là 0.15 (15%), và `random_state` được sử dụng để đảm bảo việc chia dữ liệu này có thể được lặp lại một cách nhất quán. Kết quả là ta nhận được các biến `X_train`, `X_test`, `y_train`, `y_test` chứa dữ liệu đã được chia thành các tập tương ứng.

```
from sklearn.model_selection import train_test_split
X_train , X_test , y_train , y_test = train_test_split(X_sample,y_sample,test_size = 0.15, random_state = 0)
```

MODEL

- import class GradientBoostingRegressor và sử dụng phương thức fit của reg để huấn luyện mô hình Gradient Boosting trên tập dữ liệu huấn luyện X_train và y_train để tạo ra mô hình dự đoán.

```
from sklearn.ensemble import GradientBoostingRegressor  
reg = GradientBoostingRegressor(random_state=0).fit(x_train,y_train)
```

- Import module eli5 và class PermutationImportance từ eli5.sklearn.
 - Tạo một đối tượng perm thông qua PermutationImportance và huấn luyện nó trên tập dữ liệu kiểm tra X_test và y_test được tạo ở trên. Phương pháp Permutation Importance được sử dụng để đo lường tầm quan trọng của các tính năng trong mô hình.
 - Sử dụng eli5.show_weights để hiển thị các trọng số của các tính năng, dựa trên đối tượng perm. Kết quả được gán cho biến features.

- `features_weights` chứa bảng độ quan trọng của các tính năng được tính toán bằng phương pháp Permutation Importance.

- Bảng này sẽ cho thấy độ quan trọng của các tính năng theo thứ tự giảm dần, trong đó tính năng quan trọng nhất được xếp đầu bảng. Mỗi hàng trong bảng sẽ bao gồm tên tính năng và độ quan trọng tương ứng của nó.

Weight	Feature
1.1385 ± 0.0337	Dept
0.3484 ± 0.0103	Size
0.0517 ± 0.0015	Unnamed: 0
0.0185 ± 0.0009	CPI
0.0146 ± 0.0011	Week
0.0044 ± 0.0033	MarkDown3
0.0025 ± 0.0001	Type
0.0014 ± 0.0003	Store
0.0011 ± 0.0002	Unemployment
0.0005 ± 0.0001	IsHoliday
0.0002 ± 0.0003	Month
0.0001 ± 0.0000	MarkDown4
0.0001 ± 0.0001	Day
0.0001 ± 0.0000	Temperature
0 ± 0.0000	Fuel_Price
0 ± 0.0000	MarkDown2
0 ± 0.0000	MarkDown5
0 ± 0.0000	Year
-0.0001 ± 0.0000	MarkDown1

- Biến f_importances là một Series pandas chứa độ quan trọng của các tính năng được tính bằng phương pháp Permutation Importance

```
f_importances = pd.Series(dict(zip(X_test.columns.tolist(),
| | | | | perm.feature_importances_))).sort_values(ascending=False)
f_importances
```

- Series f_importances với tính năng có độ quan trọng cao nhất được đặt ở đầu. Các tính năng sẽ được sắp xếp theo thứ tự giảm dần của độ quan trọng. Mỗi hàng trong Series sẽ chứa tên tính năng và độ quan trọng tương ứng của nó.

Dept	1.138506
Size	0.348378
Unnamed: 0	0.051675
CPI	0.018493
Week	0.014554
MarkDown3	0.004395
Type	0.002522
Store	0.001359
Unemployment	0.001144
IsHoliday	0.000534
Month	0.000246
MarkDown4	0.000085
Day	0.000068
Temperature	0.000052
MarkDown5	0.000000
MarkDown2	0.000000
Year	0.000000
Fuel_Price	0.000000
MarkDown1	-0.000079
dtype:	float64

- sử dụng eli5.show_weights để hiển thị độ quan trọng của các tính năng và lưu kết quả vào biến weights

```
weights = eli5.show_weights(perm, top=len(x_train.columns),
                             feature_names=x_test.columns.tolist())
result = pd.read_html(weights.data)[0]
result
```

- sử dụng pd.read_html để đọc dữ liệu từ weights.data và lấy DataFrame đầu tiên từ kết quả để lưu vào biến result.
- Kết quả là result sẽ chứa DataFrame chứa độ quan trọng của các tính năng, được sắp xếp theo thứ tự giảm dần.

	Weight	Feature
0	1.1385 ± 0.0337	Dept
1	0.3484 ± 0.0103	Size
2	0.0517 ± 0.0015	Unnamed: 0
3	0.0185 ± 0.0009	CPI
4	0.0146 ± 0.0011	Week
5	0.0044 ± 0.0033	MarkDown3
6	0.0025 ± 0.0001	Type
7	0.0014 ± 0.0003	Store
8	0.0011 ± 0.0002	Unemployment
9	0.0005 ± 0.0001	IsHoliday
10	0.0002 ± 0.0003	Month
11	0.0001 ± 0.0000	MarkDown4
12	0.0001 ± 0.0001	Day
13	0.0001 ± 0.0000	Temperature
14	0 ± 0.0000	Fuel_Price
15	0 ± 0.0000	MarkDown2
16	0 ± 0.0000	MarkDown5
17	0 ± 0.0000	Year
18	-0.0001 ± 0.0000	MarkDown1

```
def WMAE(dataset, real, predicted):  
    weights = dataset.IsHoliday.apply(lambda x: 5 if x else 1)  
    return np.round(np.sum(weights*abs(real-predicted))/(np.sum(weights)), 2)
```

- Tạo hàm có tên là WMAE (Weighted Mean Absolute Error). Hàm này tính toán WMAE dựa trên các thông số đầu vào là dataset (tập dữ liệu), real (giá trị thực tế), và predicted (giá trị dự đoán).
- Cụ thể, các bước thực hiện trong hàm WMAE như sau:
- Tạo một mảng weights dựa trên cột IsHoliday trong dataset. Giá trị trong mảng weights được xác định bằng 5 nếu IsHoliday là True và 1 nếu IsHoliday là False. Điều này áp dụng trọng số cao hơn cho các mẫu trong ngày lễ (IsHoliday=True).
- Tính WMAE bằng cách tính tổng trọng số nhân với giá trị tuyệt đối của hiệu giữa real và predicted, sau đó chia cho tổng trọng số. Kết quả WMAE được làm tròn đến 2 chữ số thập phân.
- Trả về giá trị WMAE tính được.
- Hàm WMAE này giúp tính toán WMAE trong mô hình dự đoán, trong đó mức độ sai lệch được đánh trọng số dựa trên cột IsHoliday của dữ liệu.

Tạo dictionary `models` chứa các mô hình hồi quy khác nhau.

```
models = {
    'LinearReg': LinearRegression(),
    'GBoost': ensemble.GradientBoostingRegressor(random_state = 0, loss = 'squared_error'),
    'HGBR': HistGradientBoostingRegressor(random_state = 0),
    'ExtraTr': ensemble.ExtraTreesRegressor(bootstrap = True, random_state = 0),
    'RandomF': ensemble.RandomForestRegressor(random_state = 0),
    'KNN': KNeighborsRegressor(),
    'SVR': SVR(),
    'DecisionTree': DecisionTreeRegressor(random_state=0)
}
```

```
def model_evaluation (name, model, models, x_train, y_train, x_test, y_test):
    rmsses = []
    for i in range(len(models)):
        # Model fit
        model.fit(x_train, y_train)

        # Model predict
        y_preds = model.predict(x_test)

        # RMSE
        rmse = np.sqrt(np.mean((y_test - y_preds)**2))
        rmsses.append(rmse)

    return np.mean(rmsses)
```

Hàm này tạo ra để đánh giá hiệu suất của một mô hình hồi quy và so sánh với các mô hình khác, dựa trên giá trị RMSE trên dữ liệu test

```
for name, model in models.items():
    print(name + ' valid RMSE {:.4f}'.format(
        model_evaluation(name, model, models, x_train, y_train, x_test, y_test)))
```

- Trong mỗi vòng lặp, tên của mô hình và đối tượng mô hình tương ứng được trích xuất từ từ điển models. Hàm model_evaluation được gọi để đánh giá hiệu suất của mô hình, và giá trị RMSE trả về được lưu trong biến rmse. Cuối cùng, thông tin về tên mô hình và RMSE được hiển thị lên màn hình.

```
LinearReg Valid RMSE 21969.0168
GBoost Valid RMSE 12064.2005
HGBR Valid RMSE 7299.0327
ExtraTr Valid RMSE 5404.9242
RandomF Valid RMSE 4539.1281
KNN Valid RMSE 21309.8979
SVR Valid RMSE 24122.9689
DecisionTree Valid RMSE 6325.3055
```

- Đây là kết quả đánh giá hiệu suất của từng mô hình hồi quy trên dữ liệu kiểm tra, dựa trên giá trị Root Mean Squared Error (RMSE). Kết quả RMSE càng thấp thì mô hình càng có hiệu suất tốt hơn.

- so sánh thì thấy RMSE của RandomForest thấp nhất sau đó thì đến ExtraTrees nên quyết định sử dụng kết quả dự đoán của model RandomForest để làm kết quả cuối cùng
- Đây là các bước để em lấy kết quả dự đoán dựa vào vào file sample_submission

```
x_baseline = X[['Store', 'Dept', 'IsHoliday', 'Size', 'Week', 'Type', 'Year', 'Day']].copy()
```

- Biến X_baseline được tạo ra bằng cách sao chép một phần của dữ liệu từ biến X. Các cột được chọn trong X_baseline là 'Store', 'Dept', 'IsHoliday', 'Size', 'Week', 'Type', 'Year', và 'Day'.
- Do đó, X_baseline chứa các tính năng này từ X và được sử dụng như một bộ dữ liệu cơ sở (baseline) để thực hiện huấn luyện và dự đoán trong các phần sau của mã.

```
x_train, x_valid, y_train, y_valid =  
model_selection.train_test_split(X_baseline, y, random_state=0, test_size=0.1)
```

sử dụng hàm `train_test_split` từ module `model_selection` của `sklearn` để chia dữ liệu huấn luyện và dữ liệu kiểm tra từ X_baseline và y. Kết quả của việc chia sẽ được gán cho các biến `X_train`, `X_valid`, `y_train`, và `y_valid`.

- `X_baseline`: Tập dữ liệu huấn luyện ban đầu.
- `y`: Tập dữ liệu mục tiêu ban đầu.
- `random_state`: Giá trị ngẫu nhiên sử dụng để chia dữ liệu. Đặt cùng một `random_state` sẽ đảm bảo việc chia dữ liệu là nhất quán trong các lần chạy khác nhau.
- `test_size`: Kích thước dữ liệu kiểm tra, được đặt là 0.1 (10% của dữ liệu được chọn làm dữ liệu kiểm tra).

```
RF = ensemble.RandomForestRegressor(n_estimators=60,  
| | | max_depth=25, min_samples_split=3, min_samples_leaf=1)  
RF.fit(X_train, y_train)
```

```
RandomForestRegressor
```

```
RandomForestRegressor(max_depth=25, min_samples_split=3, n_estimators=60)
```

tạo một đối tượng mô hình RandomForestRegressor từ sklearn.ensemble với các tham số cụ thể như sau:

- n_estimators=60: Số lượng cây quyết định trong mô hình là 60.
- max_depth=25: Độ sâu tối đa của các cây quyết định là 25.
- min_samples_split=3: Số lượng mẫu tối thiểu để tiếp tục chia một nút trong quá trình xây dựng cây là 3.
- min_samples_leaf=1: Số lượng mẫu tối thiểu yêu cầu để tạo một lá của cây là 1.

```
test = data_test[['Store', 'Dept', 'IsHoliday', 'Size', 'Week', 'Type', 'Year', 'Day']].copy()  
predict_rf = RF.predict(test)
```

Đoạn mã trên sử dụng mô hình RandomForestRegressor đã được huấn luyện (RF) để dự đoán giá trị trên dữ liệu kiểm tra (data_test). Các tính năng được sử dụng để dự đoán là 'Store', 'Dept', 'IsHoliday', 'Size', 'Week', 'Type', 'Year', và 'Day'. Kết quả dự đoán được lưu trong biến predict_rf. Kết quả dự đoán (predict_rf) chứa các giá trị dự đoán tương ứng với dữ liệu kiểm tra, được tạo ra bởi mô hình RandomForestRegressor.

```
sample_submission = pd.read_csv("D:\Học\DAT\project\sampleSubmission.csv")
sample_submission['Weekly_Sales'] = predict_rf
sample_submission.to_csv('D:\submission_13.csv', index=False)
```

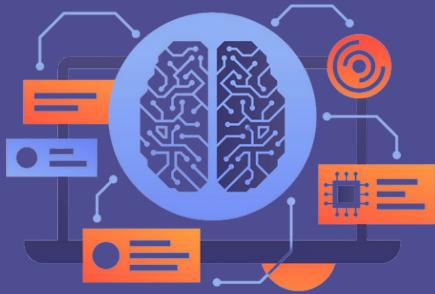
- Đoạn mã trên đọc tệp tin "sampleSubmission.csv" và tạo DataFrame sample_submission. Sau đó, cột 'Weekly_Sales' của DataFrame này được cập nhật với các giá trị dự đoán từ mô hình RandomForestRegressor. Cuối cùng, DataFrame sample_submission được lưu thành tệp tin "submission_13.csv" mà chứa các dự đoán được xuất ra dưới định dạng CSV.

```
sample_submission['Weekly_Sales'] = avg_predicts
sample_submission.to_csv('submission2_11.csv', index=False)
```

Làm tương tự các bước như trên và sử dụng model LinearRegression để lấy kết quả dự đoán và so sánh

Chúng ta dễ dàng nhìn bằng mắt thường độ chênh lệch từ kết quả của 2 model:

 	submission2_11.csv Complete (after deadline) · now	19926.70953	19486.66072
 	submission_13.csv Complete (after deadline) · 1d ago	2792.0604	2714.02302



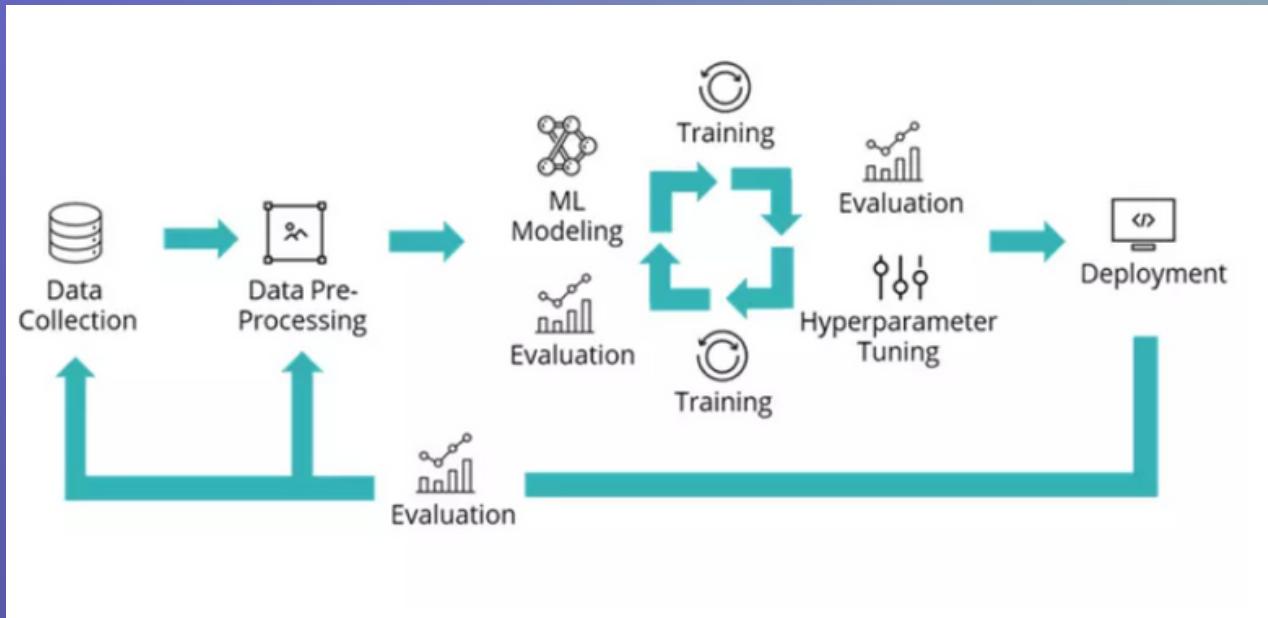
7. DATA - CENTRIC

A NEW APPROACH





Data Centric AI



Phương pháp tiếp cận lấy dữ liệu làm trung tâm đối với AI là tập trung vào việc lấy đúng loại dữ liệu có thể được sử dụng để xây dựng các mô hình học máy chất lượng cao, hiệu suất cao. Không giống như AI tập trung vào mô hình, trọng tâm chuyển sang lấy dữ liệu chất lượng cao cho các mô hình đào tạo hơn là các mô hình.

Trong thời đại hiện giờ, khi mà mô hình AI đã phát triển và các công ty đa phần cũng đều có một lượng dữ liệu của riêng mình, thì dữ liệu trở thành cốt lõi của mọi quy trình ra quyết định. Một số công ty lấy dữ liệu làm trung tâm hay còn gọi là tiếp cận theo hướng dữ liệu có thể dựa vào dữ liệu để phân tích thông tin về hoạt động của công ty, doanh nghiệp để điều chỉnh chiến lược phù hợp với mình nhằm tăng lợi ích cho chính công ty. Bằng cách tiếp cận này, kết quả có thể chính xác hơn, có tổ chức và minh bạch hơn, có thể giúp tổ chức hoạt động trơn tru hơn. Cách tiếp cận này liên quan đến việc thay đổi / cải tiến bộ dữ liệu một cách có hệ thống để tăng độ chính xác của các ứng dụng học máy. Làm việc trên dữ liệu là mục tiêu trọng tâm của phương pháp này.



APPLY FEATURE ENGINEERING

Dùng các dữ liệu cuộc thi cung cấp để tạo ra các feature mới, tăng sự đa dạng của dữ liệu.

```
df_train['Days_to_Thanskgiving'] = (pd.to_datetime(train_df["Year"].astype(str)+"-11-24", format="%Y-%m-%d") -
```

```
+ Code + Markdown
```

```
df_test['Days_to_Thanskgiving'] = (pd.to_datetime(test_df["Year"].astype(str)+"-11-24", format="%Y-%m-%d") -
```

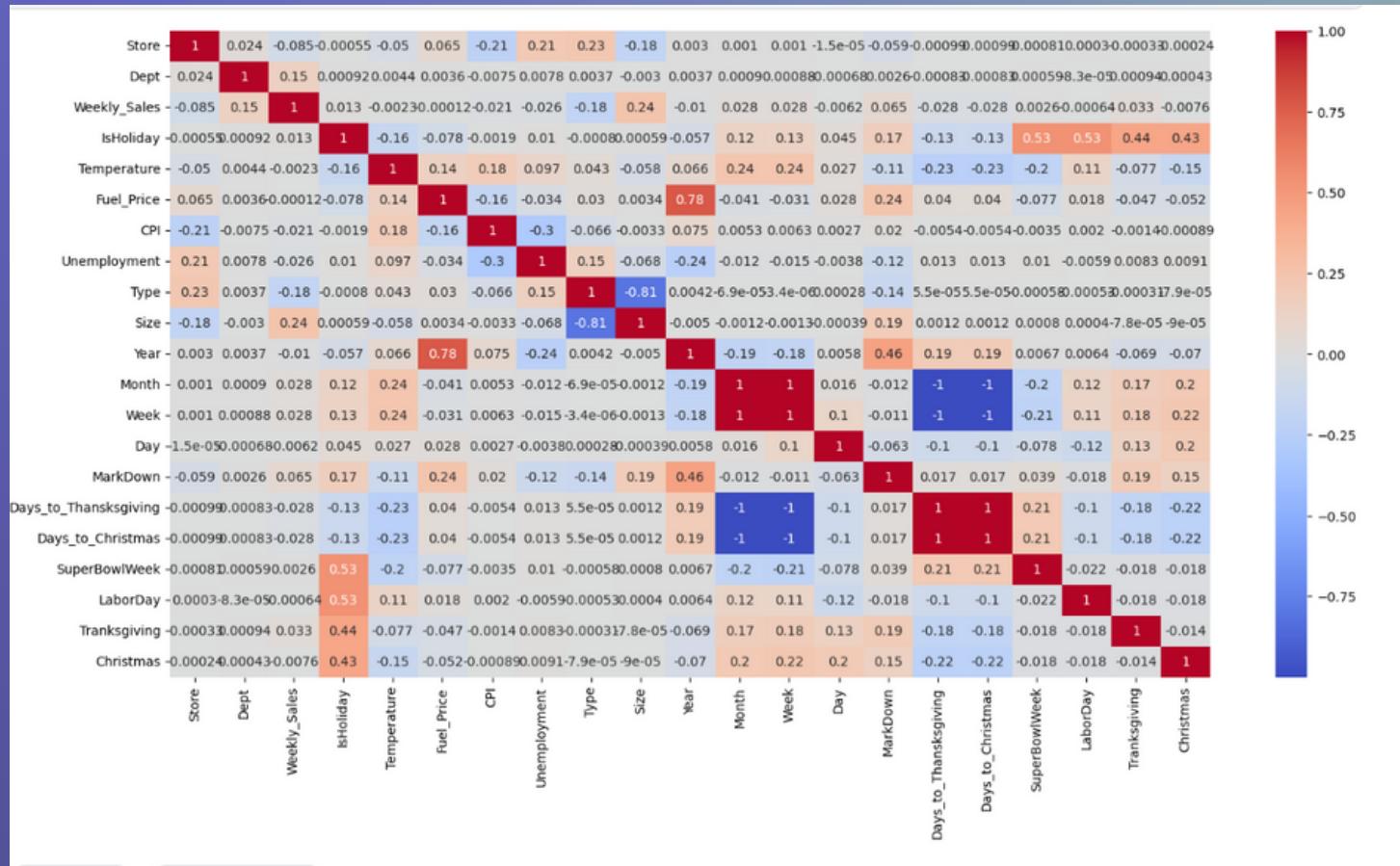
```
df_train['SuperBowlWeek'] = train_df['Week'].apply(lambda x: 1 if x == 6 else 0)
df_train['LaborDay'] = train_df['Week'].apply(lambda x: 1 if x == 36 else 0)
df_train['Thanksgiving'] = train_df['Week'].apply(lambda x: 1 if x == 47 else 0)
df_train['Christmas'] = train_df['Week'].apply(lambda x: 1 if x == 52 else 0)
```

```
df_test['SuperBowlWeek'] = test_df['Week'].apply(lambda x: 1 if x == 6 else 0)
df_test['LaborDay'] = test_df['Week'].apply(lambda x: 1 if x == 36 else 0)
df_test['Thanksgiving'] = test_df['Week'].apply(lambda x: 1 if x == 47 else 0)
df_test['Christmas'] = test_df['Week'].apply(lambda x: 1 if x == 52 else 0)
```

MarkDown	Days_to_Thanskgiving	Days_to_Christmas	SuperBowlWeek	LaborDay	Thanksgiving	Christmas
0.000000	292	322	0	0	0	0
0.000000	285	315	1	0	0	0
0.000000	278	308	0	0	0	0
0.000000	271	301	0	0	0	0
0.000000	264	294	0	0	0	0
...
1.172471	57	87	0	0	0	0
1.196122	50	80	0	0	0	0
0.811618	43	73	0	0	0	0
0.493163	36	66	0	0	0	0
0.649795	29	59	0	0	0	0



NEW HEATMAP AND FEATURE SELECTION



Ngoài các feature đã được chọn ở mô hình mục, các ngày lễ cũng nên được xem xét kĩ lương, đặc biệt là Giáng Sinh và Lễ Tạ Ơn

names = ['Store', 'Dept', 'IsHoliday', 'Size', 'Week', 'Day', 'Month', 'Days_to_Thanksgiving', 'Thanksgiving', 'CPI', 'Christmas', 'Days_to_Christmas', 'SuperBowlWeek', 'LaborDay']

ROBUST SCALER

```
from sklearn import preprocessing
transform_columns = ['Size', 'MarkDown', 'CPI', 'Temperature', 'Fuel_Price', 'Unemployment']
for column_label in transform_columns:
    robust_scaler = preprocessing.RobustScaler().fit(df_train[[column_label]])
    df_train[column_label] = robust_scaler.transform(df_train[[column_label]])
    df_test[column_label] = robust_scaler.transform(df_test[[column_label]])
```



WHY ROBUST INSTEAD OF STANDARDIZATION SCALER

Robust Scaling:

- Uses quartiles (25th and 75th percentiles) rather than mean and standard deviation.
- More robust to outliers in the data.
- Removes the median and scales the interquartile range to 1.
- Works well for data with outliers.
- Can work better for non-normally distributed data.

Standard Scaling:

- Uses mean and standard deviation for scaling.
- Scales the data to have a mean of 0 and standard deviation of 1.
- Assumes data is normally distributed.
- More sensitive to outliers.
- If data is skewed, may scale majority of data to a small range.
- Works well when data is normally distributed.

In general:

- Robust scaling is preferred if your data contains outliers. It avoids letting the outliers skew the transformation.
- Standard scaling is preferred for normally distributed data as it scales according to the distribution shape.
- Standard scaling tends to be the default scaling technique for many ML algorithms. But robust scaling is a good option to handle outliers.
- So in summary, robust scaling is more robust to non-normal distributions but standard scaling takes the distribution into account directly. Choose based on data and outliers.



TESTING AND SUBMISSION CHECK

```
# # Prepare data
names = ['Store', 'Dept', 'IsHoliday', 'Size', 'Week', 'Day', 'Month', 'Type', 'Days_to_Thansksgiving',
'Thanksgiving', 'CPI', 'Christmas', 'Days_to_Christmas', 'LaborDay']
X_test = df_test[names]
X = df_train[names]
y = df_train['Weekly_Sales']

# Create a Random Forest regressor
rf = RandomForestRegressor(random_state=42)

# Train the model
rf.fit(X, y)

# Make predictions on the test set
rf_pred = rf.predict(X_test)
```



tulahe172220 - Version 13

Complete (after deadline) · 1h ago · Notebook tulahe172220 | Version 13

3494.8122

3384.60896



TESTING AND SUBMISSION FINAL

Cắt bỏ các feature đa tuyến tính và 1 số feature không có tính tương quan

```
# # Prepare data
names = ['Store', 'Dept', 'IsHoliday', 'Size', 'Week', 'Type', 'Year', 'Day']
X_test = df_test[names]
X = df_train[names]
y = df_train['Weekly_Sales']

# Create a Random Forest regressor
rf = RandomForestRegressor(random_state=42)

# Train the model
rf.fit(X, y)

# Make predictions on the test set
rf_pred = rf.predict(X_test)
```



tulahe172220 - Version 14

Complete (after deadline) · 31m ago · Notebook tulahe172220 | Version 14

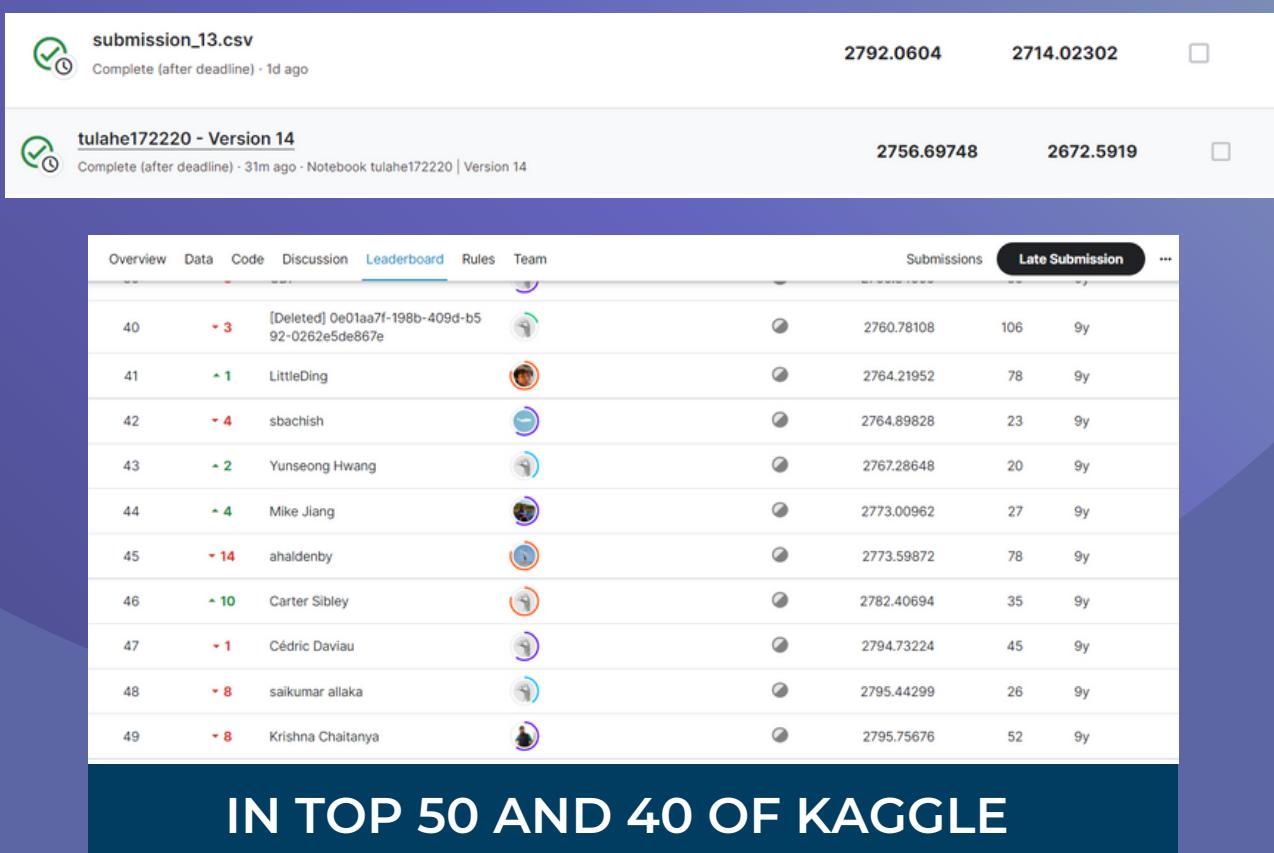
2756.69748

2672.5919



8. Summary

PROJECT OUTCOMES



The screenshot shows two sections of a Kaggle competition interface. The top section displays two recent submissions: 'submission_13.csv' (Complete, 1d ago) and 'tulahe172220 - Version 14' (Complete, 31m ago). The bottom section is a 'Leaderboard' table for a competition, showing the top 49 submissions. The columns include rank, name, description, profile picture, submissions count, late submissions count, and time since submission. The table shows a mix of individual and team submissions, with names like LittleDing, sbachish, Yunseong Hwang, Mike Jiang, ahaldenby, Carter Sibley, Cédric Daviau, saikumar allaka, and Krishna Chaitanya.

Rank	User	Description	Profile	Submissions	Late Submissions	Time
40	▼ 3	[Deleted] 0e01aa7f-198b-409d-b592-0282e5de867e	...	2760.78108	106	9y
41	▲ 1	LittleDing	...	2764.21952	78	9y
42	▼ 4	sbachish	...	2764.89828	23	9y
43	▲ 2	Yunseong Hwang	...	2767.28648	20	9y
44	▲ 4	Mike Jiang	...	2773.00962	27	9y
45	▼ 14	ahaldenby	...	2773.59872	78	9y
46	▲ 10	Carter Sibley	...	2782.40694	35	9y
47	▼ 1	Cédric Daviau	...	2794.73224	45	9y
48	▼ 8	saikumar allaka	...	2795.44299	26	9y
49	▼ 8	Krishna Chaitanya	...	2795.75676	52	9y

IN TOP 50 AND 40 OF KAGGLE

Với vấn đề dự đoán doanh thu đầy thử thách khi có tới trên 70% giá trị bị thiếu ở những tính năng chứa dữ liệu quan trọng có thể ảnh hưởng đến biến mục tiêu, Walmart đã mở ra một bài toán phức tạp hơn rất nhiều so với thông thường.

Điều cốt lõi để giải quyết được bài toán này chính nằm ở khâu xử lý dữ liệu. Walmart có lẽ muốn hướng tới một hướng đi khác mà thường nhiều người bỏ quên khi chỉ chuyên tâm vào việc huấn luyện mô hình=> AI tập trung dữ liệu (Data centric AI)

Bằng việc tìm kiếm được hướng đi đúng đắn cùng với nhiều lần thử nghiệm với các mô hình Boosting (Gradient Boosting, XGBoost) hoặc Regressor (Random Forest, Linear Regression), nhóm em đã đưa ra kết luận mới: tập trung vào dữ liệu kết hợp với tập trung vào mô hình mới là lựa chọn tối ưu.

Model - Centric và Data - Centric, sự kết hợp hài hòa giữa hai phương thức này mới cho ra được kết quả đáng mong đợi nhất.

9. References



[Walmart Competition](#)

[Robust Scaling: Why and How to Use It to Handle Outliers](#)

[Optuna - Optimize Your Optimization](#)

[Dự báo bán hàng là gì?](#)

Tip: Use links to go to a different page inside your presentation.

How: Highlight the text, click on the link symbol on the toolbar, and select the page you want to link in your document.

[Data Centric AI](#)