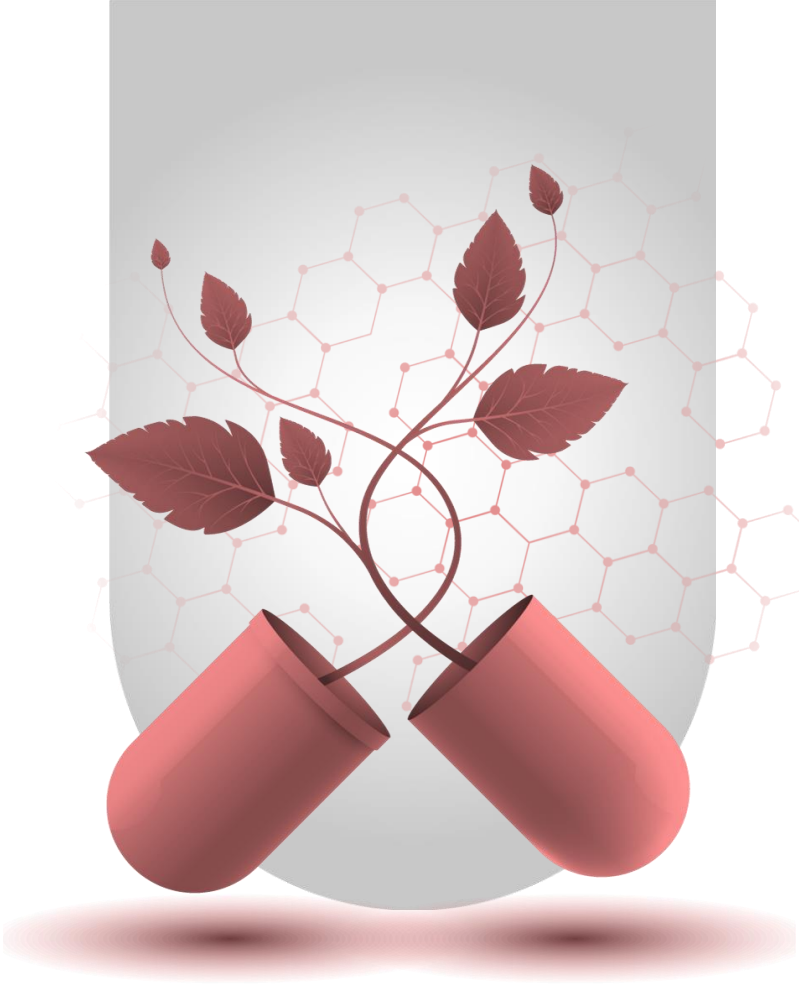


Dirk Rossmann GmbH

Forecast the daily sales for the next six weeks

Submitted by: Walter Edward

Introduce About the Company

Rossmann is a large pharmaceutical and healthcare retail chain in Europe. Founded in 1972 by Dirk Rossmann in Germany, the company now has more than 4,000 stores in many European countries such as Germany, Poland, Hungary, and the Czech Republic. Rossmann offers a wide range of products, from cosmetics, skin care, over-the-counter pharmaceuticals, to dietary supplements and personal hygiene products. With reasonable price policies and attractive promotions, Rossmann attracts many loyal customers. The company focuses on checking product quality and encouraging the use of recycled packaging. In addition to business, Rossmann actively participates in social and charitable activities, building a positive image in the community. With a commitment to quality and strategic vision, Rossmann continues to be a trusted partner in health and beauty care in Europe.

Table of contents

ROSSMANN

ROSSMANN

01

Preparation

Problem Statement

Rossmann is a European drug distributor which operates over 3,000 drug stores across seven European countries. Since a lot of drugs come with a short shelf life, that is, they do not have a long expiry date, it becomes imperative for Rossmann to accurately forecast sales at their individual stores. Currently, the forecasting is taken care of by the store managers who are tasked with forecasting daily sales for the next six weeks.

As expected, store sales are influenced by many factors, including promotional campaigns, competition, state holidays, seasonality, and locality.

With thousands of individual managers predicting sales based on their unique circumstances and intuitions, the accuracy of the forecasts is quite varied. To overcome this problem, the company has decided to build a forecasting model to forecast the daily sales for the next six weeks.

Note - Since the company is just embarking on this project, the scope has been kept to nine key stores across Europe. The stores are key for the company keeping in mind the revenue and historical prestige associated with them. These stores are numbered - 1, 3, 8, 9, 13, 25, 29, 31 and 46.

Goal of the Study

To apply data preprocessing and preparation techniques in order to obtain clean data (EDA)

To build models able to predict sales based on stores and sales history features

To analyze and compare models performance in order to choose the best model

Data Understand

Store DataFrame

Data contains:
- 10 Features
- 1115 rows
- CompetitionDistance, CompetitionOpenSinceMonth, CompetitionOpenSinceYear, Promo2SinceWeek, Promo2SinceYear, PromoInterval columns have Null values.

=> Need processing for next step.

Store DataFrame - Data Information		
Features	Non-Null Count	Dtype
Store	1115	Int
StoreType	1115	Object
Assortment	1115	Object
CompetitionDistance	1112	Float
CompetitionOpenSinceMonth	761	Float
CompetitionOpenSinceYear	761	Float
Promo2	1115	Int
Promo2SinceWeek	571	Float
Promo2SinceYear	571	Float
PromoInterval	571	Object

Data Processing

Store DataFrame: Promo2SinceWeek

Goods with a Promo2SinceWeek Null value are mostly because the store does not apply promotions (or in other words, the promo at these stores is 0). Since there is no promotion, there will be no promotion period.

So we will fill these null values with 0.

Store DataFrame - Data Information			
Store	1	...	1114
StoreType	C	...	A
Assortment	A	...	C
CompetitionDistance	1270	...	870
CompetitionOpenSinceMonth	9	...	NaN
CompetitionOpenSinceYear	2008	...	NaN
Promo2	0	...	0
Promo2SinceWeek	0	...	0
Promo2SinceYear	NaN	...	NaN
PromoInterval	NaN	...	NaN

Data Processing

Store DataFrame: Promo2SinceYear

Considering the Promo2SinceYear column is similar to the Promo2SinceWeek column, we will also fill null values in this column with 0.

Store DataFrame - Data Information			
Store	1	...	1114
StoreType	C	...	A
Assortment	A	...	C
CompetitionDistance	1270	...	870
CompetitionOpenSinceMonth	9	...	NaN
CompetitionOpenSinceYear	2008	...	NaN
Promo2	0	...	0
Promo2SinceWeek	0	...	0
Promo2SinceYear	0	...	0
PromoInterval	NaN	...	NaN

Data Processing

Store DataFrame: PromoInterval

Because these stores do not apply promotions.
So we will fill these null values with “Not Applicable”.

Store DataFrame - Data Information			
Store	1	...	1114
StoreType	C	...	A
Assortment	A	...	C
CompetitionDistance	1270	...	870
CompetitionOpenSinceMonth	9	...	NaN
CompetitionOpenSinceYear	2008	...	NaN
Promo2	0	...	0
Promo2SinceWeek	0	...	0
Promo2SinceYear	0	...	0
PromoInterval	Not Applicable	...	Not Applicable

Data Processing

Store DataFrame: CompetitionDistance

We calculate the average value of the CompetitionDistance column based on each Store Type.
The mean Competition Distance for :

- Store Type A: 5123
- Store Type B: 1060
- Store Type C: 3522
- Store Type D: 6913

Use these calculated values to fill in null values in the CompetitionDistance column according to each respective Store Type.

Store DataFrame - Data Information			
Store	291	622	879
StoreType	D	A	D
Assortment	A	C	A
CompetitionDistance	5123	3522	5123
CompetitionOpenSinceMonth	NaN	NaN	NaN
CompetitionOpenSinceYear	NaN	NaN	NaN
Promo2	0	0	1
Promo2SinceWeek	0	0	5
Promo2SinceYear	0	0	2013
PromoInterval	Not Applicable	Not Applicable	Feb,May, Aug,Nov

Data Processing

Store DataFrame: CompetitionOpenSinceMonth

Hypothesize: since these stores are starting to operate, we need to run promotions to compete. Therefore, we can use the data in the Promo2SinceWeek column as a basis for interpolation for CompetitionOpenSinceMonth. Since the data at CompetitionOpenSinceMonth is a month, while at Promo2SinceWeek the data is a week, we need to convert them from weeks to months.

Week_to_month = Week_Number / 4.35

Because some stores do not have promo, the returned result is 0.

We take one more step, replacing these 0 values with the interpolate() method from the available library.

Store DataFrame - Data Information			
Store	12	...	1115
StoreType	A	...	D
Assortment	C	...	C
CompetitionDistance	1070	...	5350
CompetitionOpenSinceMonth	2	...	5
CompetitionOpenSinceYear	NaN	...	NaN
Promo2	1	...	1
Promo2SinceWeek	13	...	22
Promo2SinceYear	2010	...	2012
PromoInterval	Jan,Apr, Jul,Oct	...	Mar,Jun, Sept,Dec

Data Processing

Store DataFrame: CompetitionOpenSinceYear

Similar to CompetitionOpenSinceMonth, we use the Promo2SinceYear column to fill null values in the CompetitionOpenSinceYear column.

Because some stores do not have promo, the returned result is 0.

We take one more step, replacing these 0 values with the interpolate() method from the available library.

Store DataFrame - Data Information			
Store	12	...	1115
StoreType	A	...	D
Assortment	C	...	C
CompetitionDistance	1070	...	5350
CompetitionOpenSinceMonth	2	...	5
CompetitionOpenSinceYear	2010	...	2012
Promo2	1	...	1
Promo2SinceWeek	13	...	22
Promo2SinceYear	2010	...	2012
PromoInterval	Jan,Apr,Jul,Oct	...	Mar,Jun,Sept,Dec

Data Understand

Store DataFrame

Data contains:
- 10 Features
- 1115 rows

=> Data have no-null columns
=> Ready for next step.

Store DataFrame - Data Information after processing		
Features	Non-Null Count	Dtype
Store	1115	Int
StoreType	1115	Object
Assortment	1115	Object
CompetitionDistance	1115	Float
CompetitionOpenSinceMonth	1115	Int
CompetitionOpenSinceYear	1115	Int
Promo2	1115	Int
Promo2SinceWeek	1115	Int
Promo2SinceYear	1115	Int
PromoInterval	1115	Object

Data Understand

Sales DataFrame

Data contains:
- 9 Features
- 1017209 rows
- Data have no-null column.

=> Let's go into detail

Sales DataFrame - Data Information		
Features	Non-Null Count	Dtype
Store	1017209	Int
DayOfWeek	1017209	Int
Date	1017209	Object
Sales	1017209	Int
NumberOfCustomers	1017209	Int
Open	1017209	Int
Promo	1017209	Int
StateHoliday	1017209	Object
SchoolHoliday	1017209	Int

Data Processing

Sales DataFrame: Feature Engineering related to Date column

Convert the data type of the ‘Date’ column to datetime.

Then, features engineer the Day, Month, Year, WeekOfYear, Quarter columns based on the Date column.

Sales DataFrame - Data Information			
Store	1	...	1115
DayOfWeek	Friday	...	Tuesday
Date	2015-07-31	...	2013-01-01
Sales	5263	...	0
NumberOfCustomers	555	...	0
...	...	...	
Day	31	...	1
Month	7	...	1
Year	2015	...	2013
WeekOfYear	31	...	1
Quarter	3	...	1

Data Processing

Sales DataFrame: Filter Data

When the Open column is 0 (or the store is closed), the sales column is also 0.

There are about 172817 Open is Zero values in the data, accounting for about 17%.

Finally, we exclude values with Open = 0, and the Open column from the sales data frame.

Sales DataFrame - Data Information		
Features	Non-Null Count	Dtype
Store	844392	Int
DayOfWeek	844392	Int
Date	844392	Object
Sales	844392	Int
NumberOfCustomers	844392	Int
...	...	...
Day	844392	Int
Month	844392	Int
Year	844392	Int
WeekOfYear	844392	Int
Quarter	844392	Int

Data Processing

Sales DataFrame: Feature Engineering AST column

Average Sales per Transactions (AST) is an important index in the business and retail sector. AST measures the average value of the orders or transactions over a certain period of time. This index helps business evaluate sales efficiency and optimize revenue.

Calculating Formular:
AST = Total Sales / Total Transactions

We assume that 1 customer will make 1 transaction, so the total transaction will be replaced by the total number of customers.

These null values may be caused by the values at Sales and Number Of Customers both being 0. Therefore, we will fill these Null values with 0.

Sales DataFrame - Data Information			
Store	1	...	1097
DayOfWeek	Friday	...	Tuesday
Date	2015-07-31	...	2013-01-01
Sales	5263	...	5961
NumberOfCustomers	555	...	1405
...	...	...	...
Month	07	...	1
Year	2015	...	2013
WeekOfYear	31	...	1
Quarter	3	...	1
AST	9.48	...	4.24

Data Understand

Sales DataFrame

Data contains:
- 14 Features
- 844392 rows

=> Data have no-null columns
=> Ready for next step.

Sales DataFrame - Data Information after processing		
Features	Non-Null Count	Dtype
Store	844392	Int
DayOfWeek	844392	Object
Date	844392	Datetime
Sales	844392	Int
NumberOfCustomers	844392	Int
…	...	…
Year	844392	Int
WeekOfYear	844392	Int
Quarter	844392	Int
AST	844392	Float

Data Processing

Key_store_data DataFrame		
Features	Non-Null Count	Dtype
Store	6681	Int
DayOfWeek	6681	Object
Date	6681	Datetime
Sales	6681	Int
NumberOfCustomers	6681	Int
…	...	…
Promo2	6681	Int
Promo2SinceWeek	6681	Int
Promo2SinceYear	6681	Int
PromoInterval	6681	Object

To prepare data for the next steps of the job, we combine the two existing dataframes together using the merge method, through the link of the Store column between the two dataframes.

Since the analysis only focuses on Rossmann's main stores, we will filter the data based on the list of main stores:
Key_stores = [1, 3, 8, 9, 13, 25, 29, 31, 46]

Dataframe has 23 columns and 6681 rows.

Data Processing

Key_store Dataframe: CompetitionOpenMonth

Create a new feature called CompetitionOpenMonth that represents the number of months the competitor has been active. Calculated using the formula:

12*(Year-CompetitionOpenSinceYear) + (Month - CompetitionOpenSinceMonth)

Because previously we used the data of promotion-related columns to fill in the null values of the CompetitionOpenSinceMonth column, and CompetitionOpenSinceYear should appear some values less than 0.

We replace these negative values with 0.

Key_store DataFrame - Data Information

Store	Year	Competition OpenSinceYear	Month	Competition OpenSinceMonth	Competition OpenMonth
1	2015	2008	7	9	82
3	2015	2006	7	12	103
8	2015	2014	7	10	9
	…	…		…	…
29	2013	2014	1	10	-21
31	2013	2012	1	7	6
46	2013	2005	1	9	88

Data Processing

Key_store Dataframe: Promo2OpenMonth

Key_store DataFrame - Data Information						
Store	Promo2	Year	Promo2SinceYear	WeekOfYear	Promo2SinceWeek	Promo2OpenMonth
1	0	2015	0	31	0	0
3	1	2015	2011	31	14	51
8	0	2015	0	31	0	0
...	...	…	…	…	…	…
29	0	2013	0	1	0	0
31	0	2013	0	1	0	0
46	1	2013	2011	1	14	21

Create a new feature called Promo2OpenMonth representing the number of months that the store has promoted. Calculated using the formula:

12*(Year-Promo2SinceYear) + (WeekOfYear – Promo2SinceWeek)/4.35

For stores with Promo2 equal to 0 (no promotion), the corresponding Promo2OpenMonth will also be equal to 0.

Data Processing

Key_store Dataframe: IsPromoInterval

PromoInterval contains promotion months (from "January" to "Dec"). From this data, we create a new column called IsPromoInterval.

If the value in the Month Column is in the list of values in PromoInterval, the value 1 will be returned, otherwise it will return 0. Especially if the value in PromoInterval is "Not Applicable", it will return 0.

Key_store DataFrame - Data Information

Store	Month	PromoInterval	IsPromoInterval
1	7	Not Applicable	0
3	7	Jan,Apr,Jul,Oct	1
8	7	Not Applicable	0
		…	…
29	1	Not Applicable	0
31	1	Not Applicable	0
46	1	Jan,Apr,Jul,Oct	1

Data Processing

Features	Non-Null Count	Dtype
Store	6681	Int
Date	6681	Datetime
DayOfWeek	6681	Object
Day	6681	Int
Month	6681	Int
WeekOfYear	6681	Int
…	...	...
Promo	6681	Int
Promo2	6681	Int
Promo2OpenMonth	6681	Int
IsPromoInterval	6681	Int
CompetitionDistance	6681	Float
CompetitionOpenMonth	6681	Int

Remove some unimportant features to reduce the burden on the model such as:
- 'CompetitionOpenSinceYear'
- 'CompetitionOpenSinceMonth'
- 'Promo2SinceYear'
- 'Promo2SinceWeek'
- 'PromoInterval'

Then check the data information.
At this time, the key_store_data dataframe has 21 columns and 6681 rows.

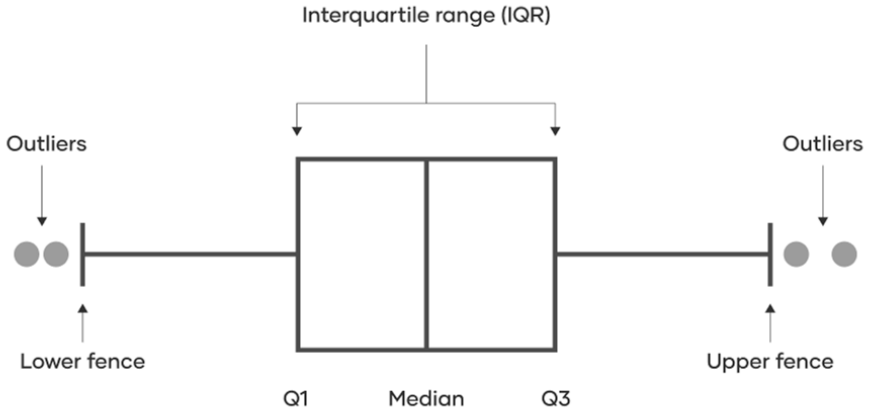
Outlier Processing

Use IQR to detect and handle outliers.

We perform calculations, and find 2 points:

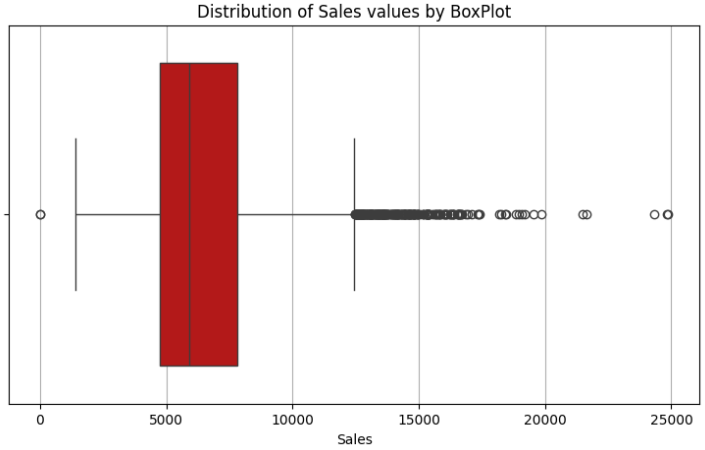
lower_fence = q1 – 1.5 * iqr
upper_fence = q3 + 1.5 * iqr

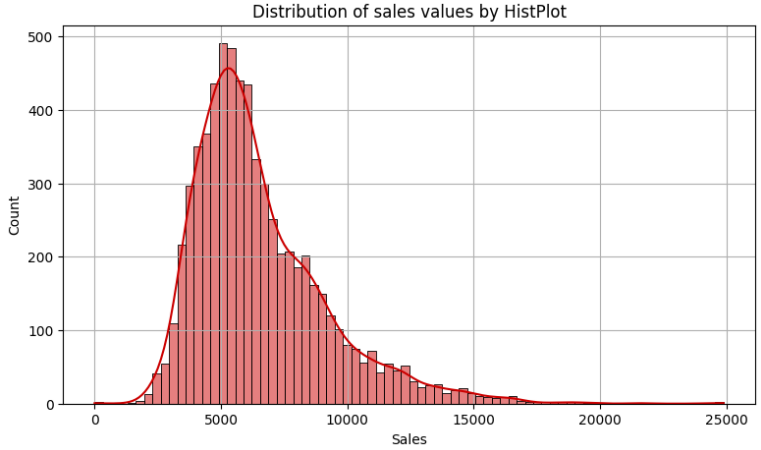
Values greater than upper_fence, or smaller than lower_fence, will be labeled as outlier.



Outlier Processing

Sales Outlier





- The sales data has a long-tailed distribution, skewed to the right, with most data points ranging from 0 to 17,000 and a few outliers extending higher.
- A skewness value of 1.42 indicates a slight positive skew, meaning the right tail is slightly longer than the left but the distribution is fairly symmetrical.
- Skewed distributions and outliers in sales data should be carefully considered during analysis and interpretation.

Outlier Processing

Sales Outlier

The data in the sales column has a lower bound of 103, and an upper bound of 12463.

Based on the data distribution in the previous sales column we learned about, the data is skewed to the right. Therefore, we will only focus on further exploration of the upper bound. Let's move on

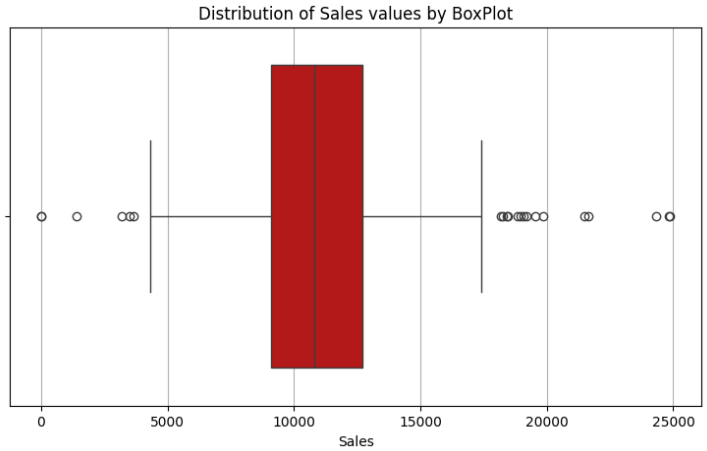
Outliers Information				
Index	Lower bound	Median point	Upper bound	Percentage of Sales Outliers
Value	103	5919	12463	3.68%

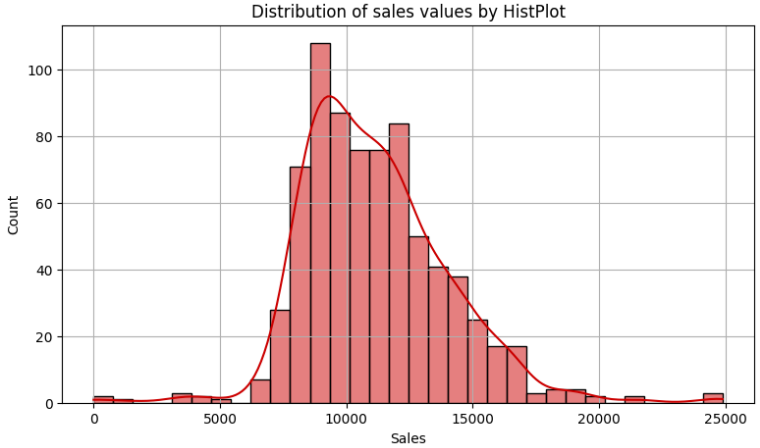
In general, most sales outliers come from store number 25, with 205 values out of a total of 244 (about 84%).
Over there, we'll go see what store 25 has.

Outliers Information					
Store	25	29	3	9	31
Count	205	27	9	2	1

Outlier Processing

Explore Store 25 Sales Outliers





- The sales data for store 25 has a long-tailed distribution, skewed to the right, with most data points ranging from 0 to 17,000 and a few outliers extending higher.
- A skewness value of 0.72 indicates a slight positive skew, meaning the right tail is slightly longer than the left but the distribution is fairly symmetrical.

Outlier Processing

Explore Store 25 Sales Outliers

Outliers Information						
Index	Min point	Lower bound	Median point	Upper bound	Max point	Percentage of Sales Outliers
Value	0	3702	10808	18119	24882	2.79%

Check how many sales values are greater than upper bound of store 25 (18119), we archive 15 rows.
Then when we check in key store data frame, we have 15 rows.

Overall, there appear to be 15 sales values in the key_store_data dataframe that are larger than the upper bound of store 25.

Considering these outliers values, which are much larger (more than 2 times) when compared to the median values of the main dataset, can affect the overall analysis so we will remove rows with sales values greater than upper bound of store 25 from the key_store_data DataFrame.

Outlier Processing

Review Summary Statistics after Cleaning

Let’s quickly review some of our summary statistics now that we’ve cleaned up our data a bit. We can look at the sales summary statistics to see how they change before and after handing outliers.

Let's move on

Outliers Information								
Index	Count	Mean	Std	Min	25%	50%	75%	Max
Value	6666	6516.58	2552.12	0	4735	5911	7806.5	17391

We can see that the cleaning has narrowed our standard deviation for sales (Decreased from 3849.93 to 2552.12), as well as an increase in the average value (increased from 5773.82 to 6516.58)

ROSSMANN

02

Data Exploration

Exploratory Data Analyst

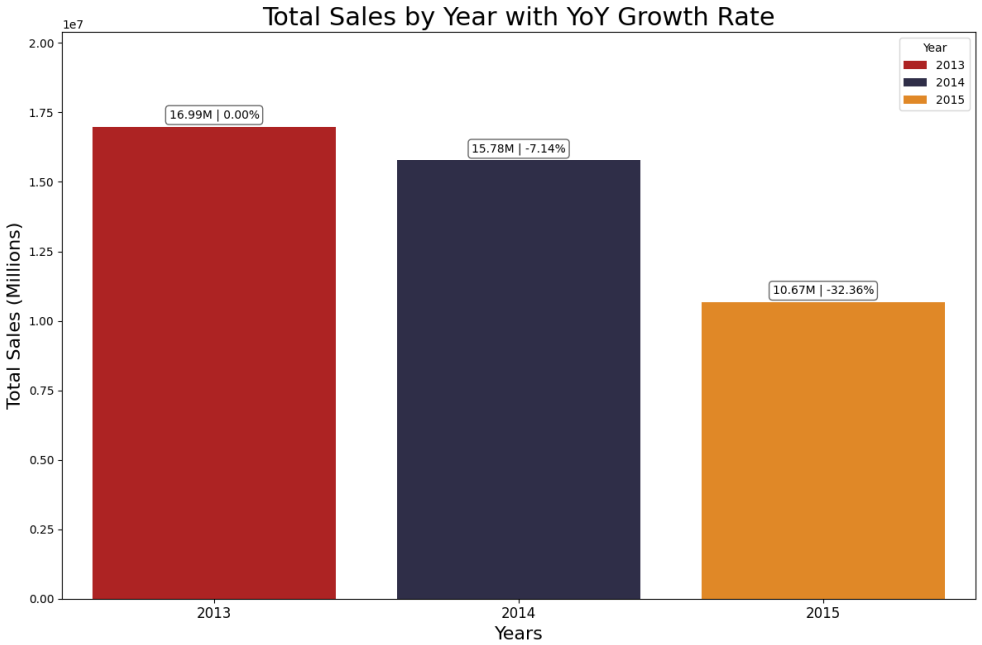
Sales Analysis: Total Sales over Three Years

- 2013: Sales reached 16.99 million, the highest in 3 years, possibly due to new stores and strong marketing.

- 2014: Sales decreased to 15.78 million, a 7.14% drop, likely due to increased competition.

- 2015: Sales as of July are 10.67 million. Full year comparison is not possible, and trends depend on the remaining months.

The chart shows a slight downward trend from 2013 to 2014. More data is needed for 2015.



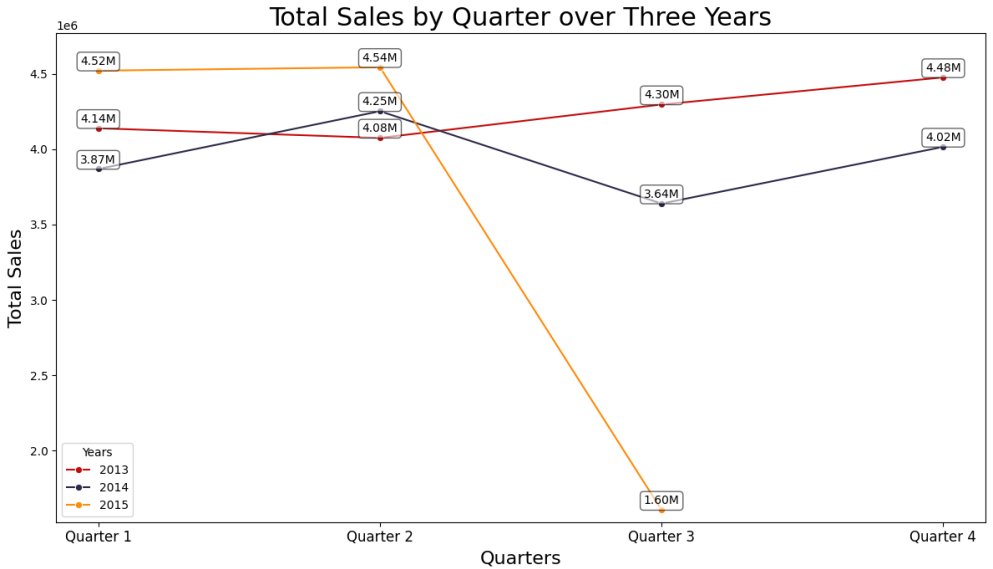
Exploratory Data Analyst

Sales Analysis: Total Sales by Quarter over Three Years

- 2013: Sales were stable with slight decreases in Q2, but grew steadily in Q3 and Q4, showing a positive and stable year.

- 2014: Sales started weakly in Q1, recovered in Q2, but dropped significantly in Q3 before recovering in Q4. Overall, the year was more volatile, with sales ending lower than in 2013.

- 2015: Sales surged in Q1 and grew slightly in Q2. Incomplete data for Q3 suggests continued growth, but a full assessment awaits complete figures.



Exploratory Data Analyst

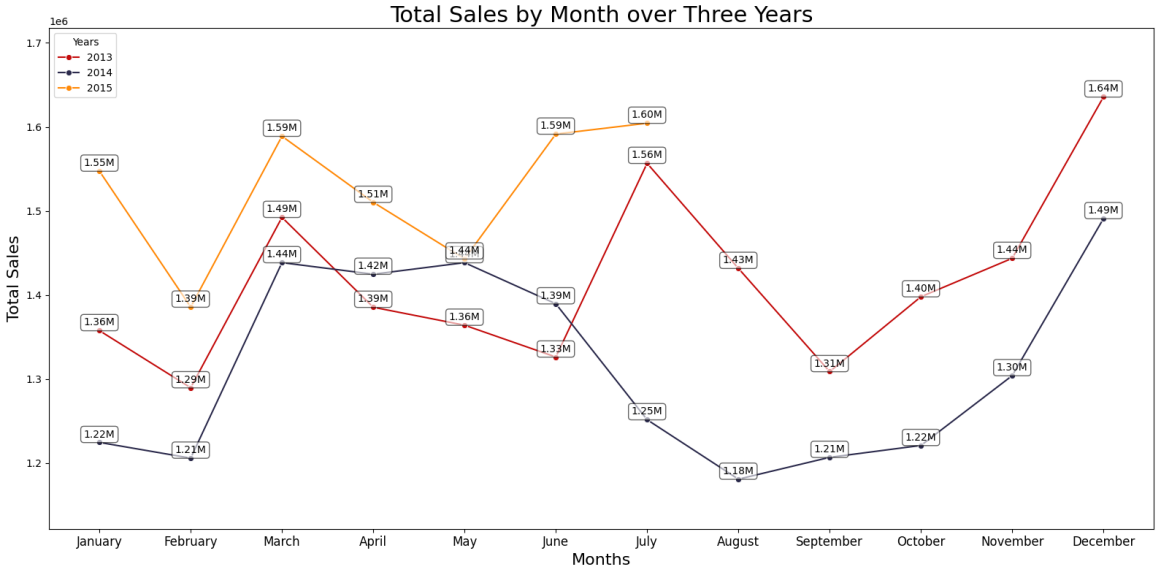
Sales Analysis: Total Sales by Month over Three Years

2013:
- Sales peaked in December ($1.64M) and dipped in February ($1.29M).
- Strongest growth in March, July, and December; largest declines in April, August, and September.

2014:
- Sales fell compared to 2013, with notable drops in July and August.
- Highest month was December ($1.49M), but still lower than December 2013.
- Strongest growth in March and December.

2015:
- January and July saw significant growth compared to previous years.
- Incomplete data for the latter part of the year.

Overview:
- Sales often decline in Q3 and recover in Q4.
- March and July generally have high sales.



ROSSMANN

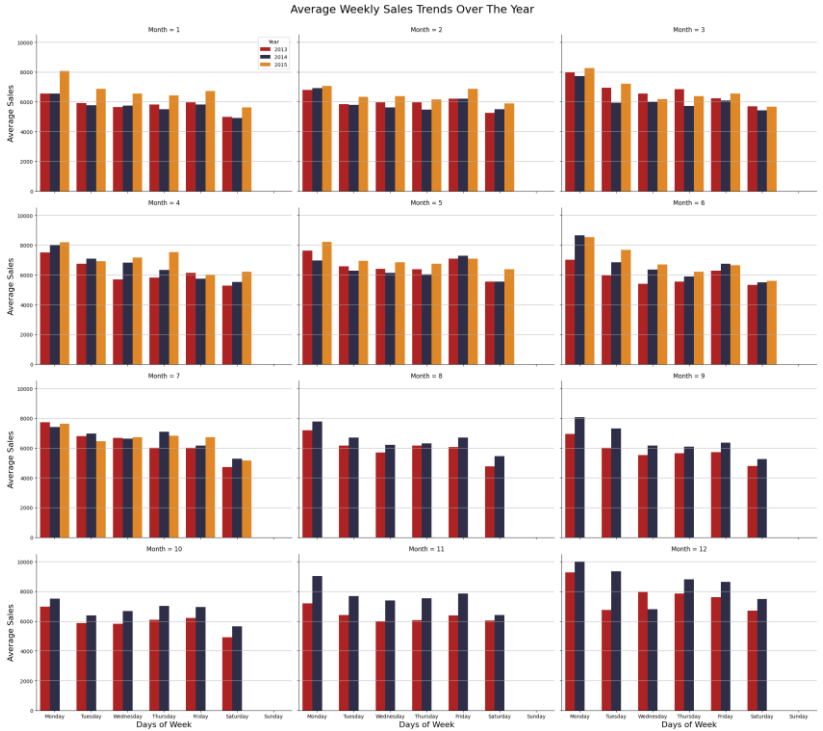
Exploratory Data Analyst

Sales Analysis: Average Weekly Sales Trends over The Year

- **January, February, May, August, October, November**: Mondays have the highest average sales; Saturdays have the lowest; other days are stable.

- **March, April, June, July, September, December**: Mondays have the highest average sales; sales decline from Monday to Saturday, with Saturday being the lowest.

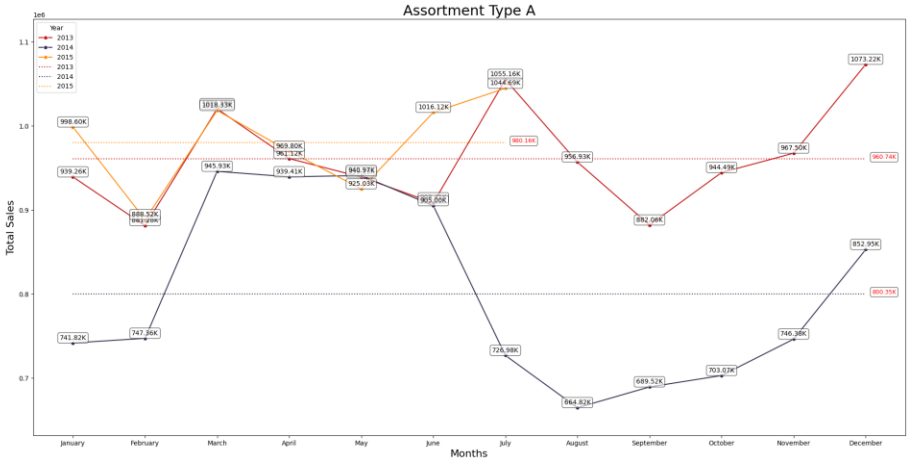
General Trends:
- Mondays generally show higher sales, while Saturdays have the lowest.
- Sales decrease throughout the week, with a notable drop on Saturday. No data is available for Sunday due to it being a holiday.

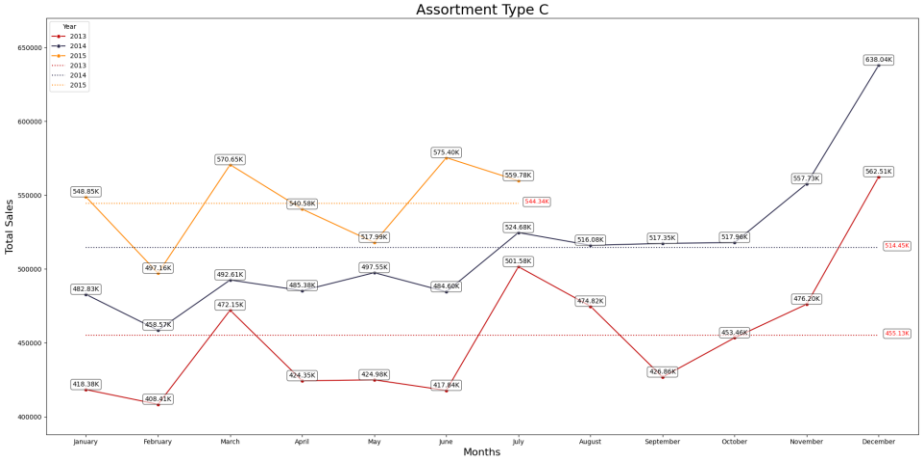


ROSSMANN

Exploratory Data Analyst

Sales Analysis: Total Sales and Average Sales by Assortment over Three Years





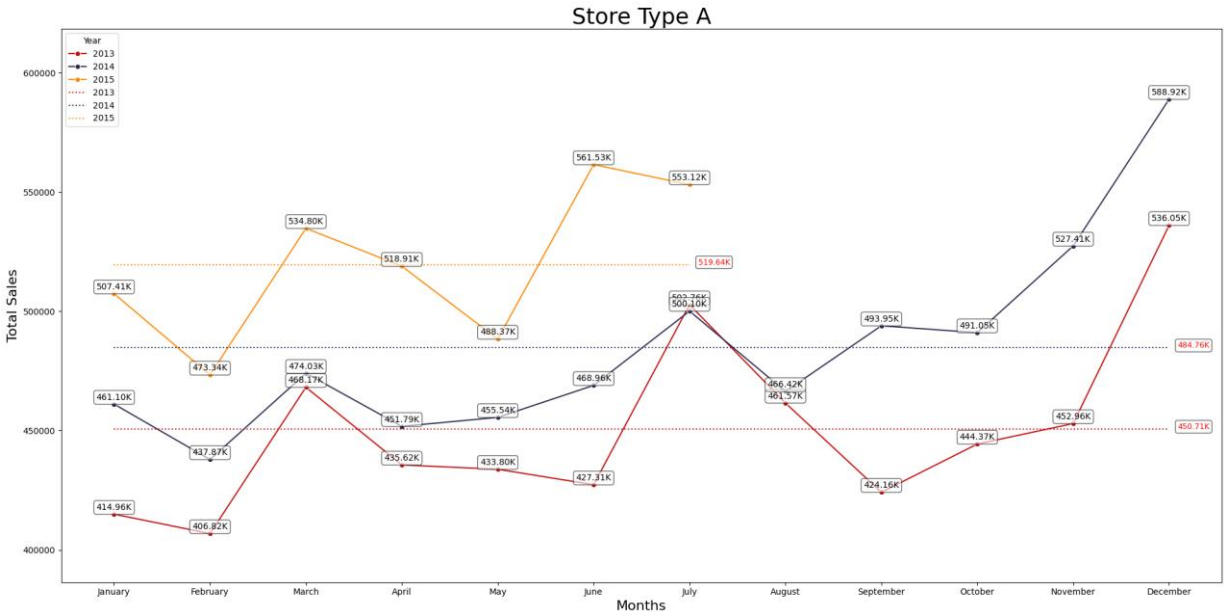
- Assortment A: 2013: Average $960K; peaks in March, July, December; lows in February, June, September.
2014: Average $800K; peaks in March, April, May; lows in August, September, October. **2015**: Average $980K; peak in July; lows in February, May.
- Assortment C: 2013: Average $455K; peaks in December; low in February. **2014**: Average $514K; peaks in December; low in February.
2015: Average $544K; peaks in March, June; lows in February, May.

Summary:
- Assortment A: Significant monthly fluctuations; peaks in early and mid-year; lows at year's start and end.
- Assortment C: More stable; growth towards year-end, especially December; low in February.

ROSSMANN

Exploratory Data Analyst

Sales Analysis: Total Sales and Average Sales by StoreType over Three Years



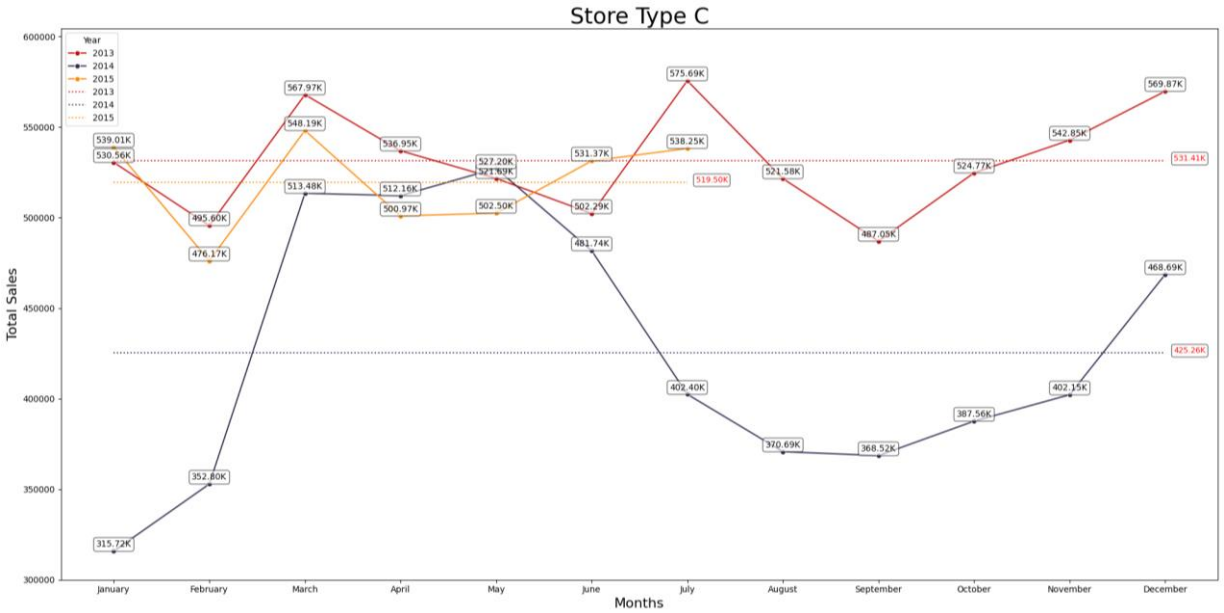
Store Type A:

Sales tend to increase steadily over the years (average in 2013 is about 450K, in 2014 is 484K, in 2015 is 519K), with sales usually tending to grow strong in the last 4 months of the year, and usually peaks in December.

ROSSMANN

Exploratory Data Analyst

Sales Analysis: Total Sales and Average Sales by StoreType over Three Years



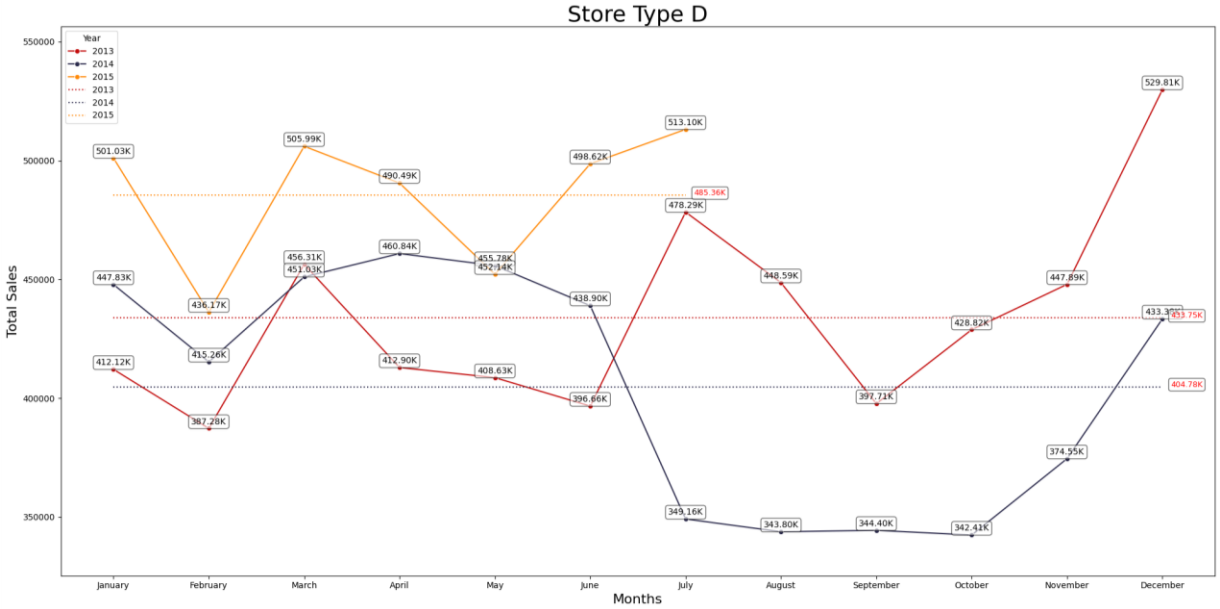
Store Type C:

Sales were stable but decreased sharply in the last 6 months of 2014 compared to 2013. Although there was a recovery in 2015, it still did not reach the level of 2013.

ROSSMANN

Exploratory Data Analyst

Sales Analysis: Total Sales and Average Sales by StoreType over Three Years



Store Type D:

Sales were relatively stable but tended to decrease in 2014. Although they increased again in 2015, sales were still lower than those of category A and C stores.

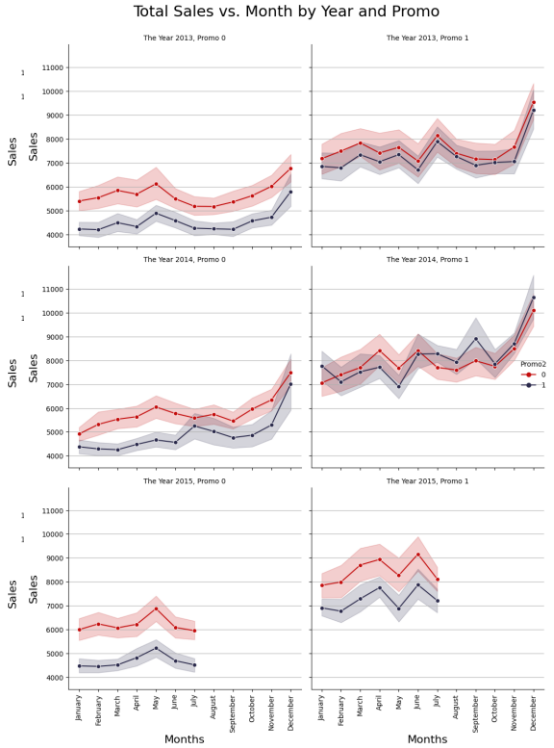
ROSSMANN

Exploratory Data Analyst

Sales Analysis: Sales Trend During Promotion over Three Years

- Promo:
Sales of all 3 years tend to increase gradually from the beginning of the year to the end of the year whether the store runs promotions (Promo = 1) or not (Promo = 0), with peak sales usually in December. However, for stores running promotions (Promo = 1), sales can often be 30% higher than stores without promotions (Promo = 0).

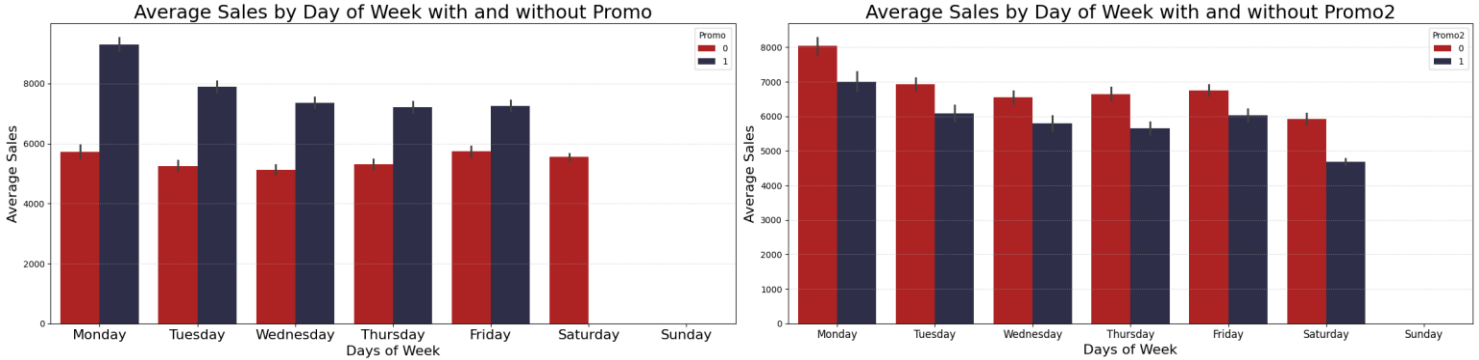
- Promo2:
Stores that participate in consecutive promotions (Promo2 = 1) typically have lower sales than stores that do not participate (Promo2 = 0). This could be because the stores participating in Promo2 have different characteristics or they are in need of continuous promotions to attract more customers.



ROSSMANN

Exploratory Data Analyst

Sales Analysis: Average Sales by DayOfWeek with and without Promotion



Promo:
- Without Promo: Average sales are stable at $5600-$6000 daily.
- With Promo: Sales increase significantly, staying above $7000, with the highest sales on Monday (> $8000) and gradually decreasing through the week.

Promo2:
- With Promo2: Sales trends are similar to without Promo2, highest on Monday and lower on Saturday. Stores with Promo2 show slightly reduced sales compared to those without it.

Conclusion:
- Promotions boost sales daily, especially on Mondays.
- Maintain and enhance promotional programs, but reassess strategies for Saturdays as they are less effective.

ROSSMANN

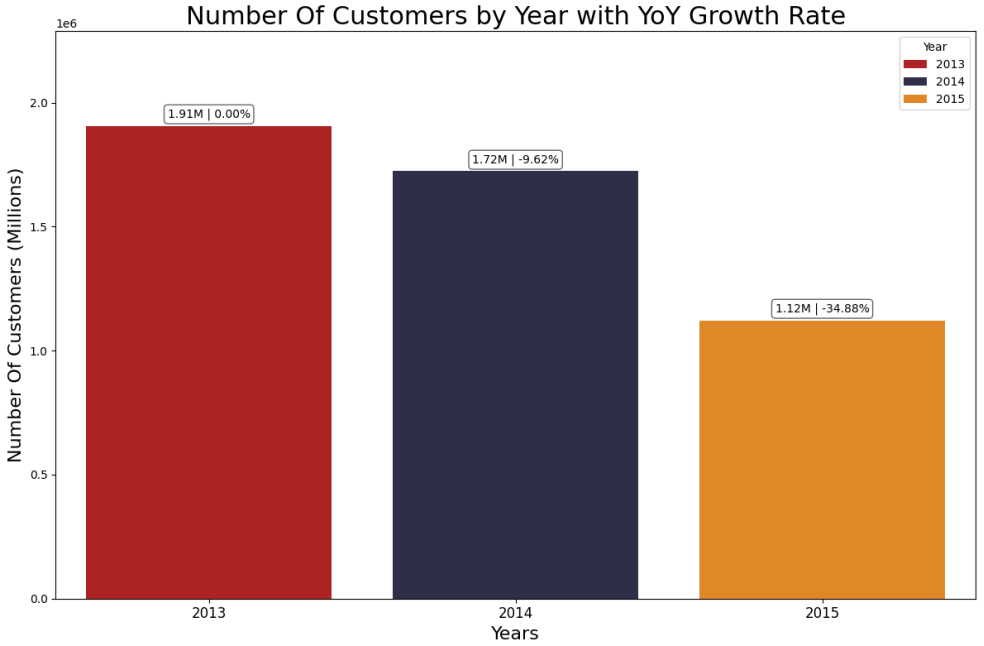
Exploratory Data Analyst

NOC Analysis: Number of Customers over Three Years

- 2013 to 2014: Number of customers decreased by 9.62%, from 1.91 million to 1.72 million, indicating business difficulties.

- 2015: A significant decline of 34.88% was observed, though only data up to July was available.

Summary:
The continuous decrease in customer numbers year by year suggests Rossmann is struggling to retain and attract customers.



ROSSMANN

Exploratory Data Analyst

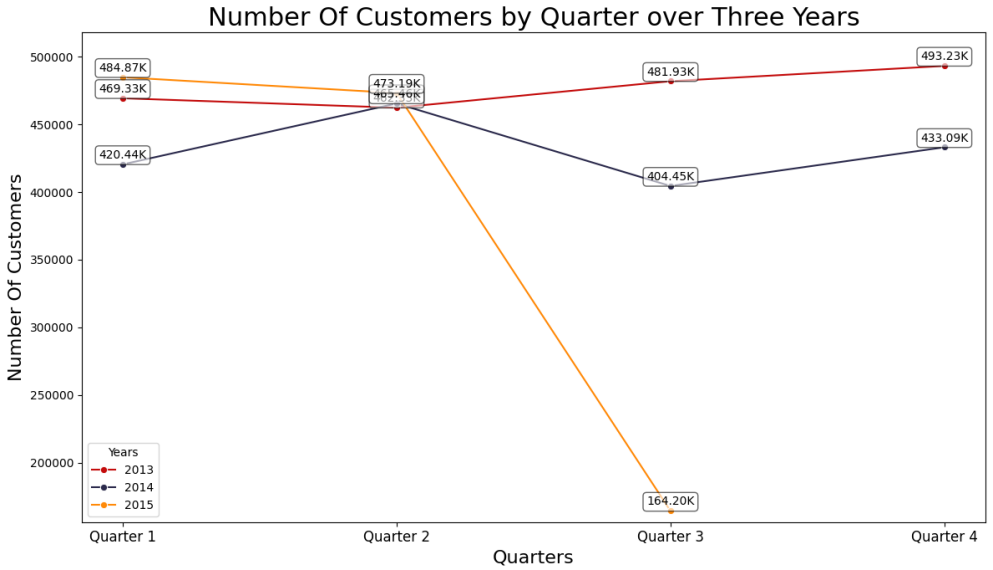
NOC Analysis: NOC by Quarter over Three Years

- 2013: Steady growth throughout the year with a slight dip in Q2, but overall positive performance.

- 2014: Significant volatility with declines in early and mid-year, though a recovery in Q4. End-of-year numbers were lower than 2013.

- 2015: Strong start in Q1 with 15.32% growth. Slight decrease in Q2, and incomplete Q3 data requires further review.

Overall:
2013 was marked by consistent growth, 2014 experienced notable fluctuations with a year-end recovery, and 2015 started strong with potential for further positive trends.



ROSSMANN

Exploratory Data Analyst

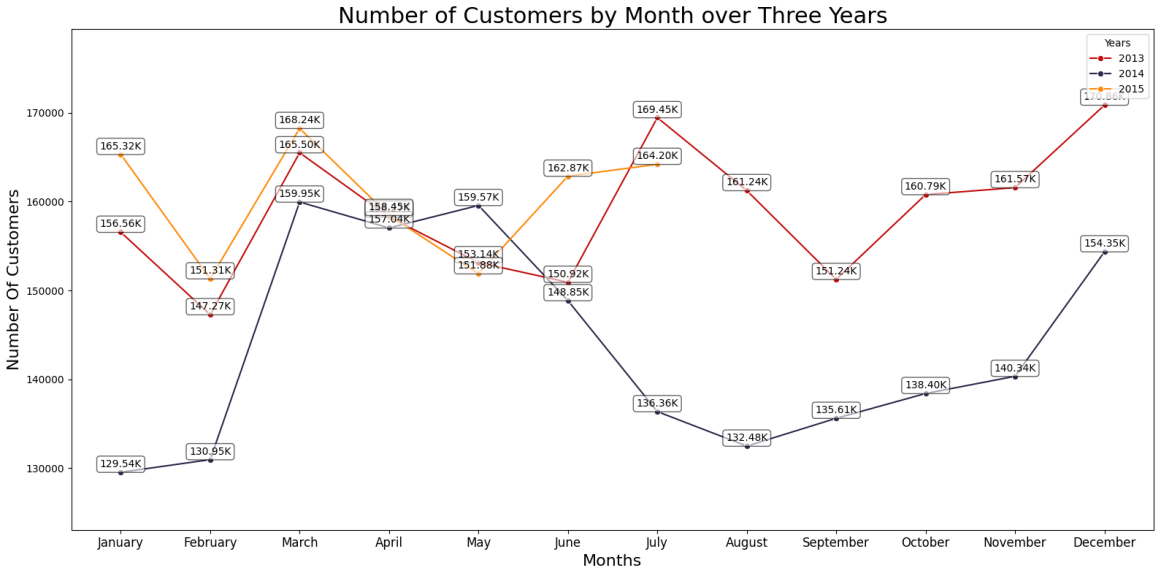
NOC Analysis: NOC by Month over Three Years

- 2013: Notable growth in March, July, and December, with December peaking at 170.86 thousand. Largest drops in February and September.

- 2014: Significant decline from 2013, with the lowest numbers in January and February. March, April, and May had the highest counts. Declines in summer, with some recovery in the fourth quarter.

- 2015: Strong start in January and peak in March. Incomplete data for the remaining months.

Overview:
Customer numbers decreased in Q3 and increased in Q4. March, July, and December had consistently high numbers.



ROSSMANN

Exploratory Data Analyst

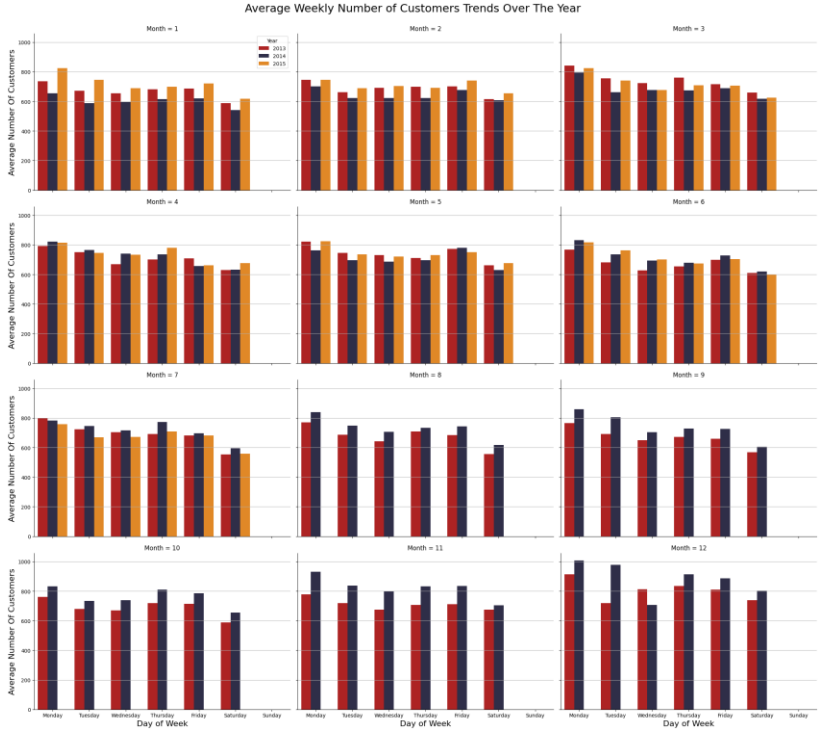
NOC Analysis: Average NOC by DayOfWeek over Months

- **Monday:** Highest average customer count.

- **Saturday:** Lowest average customer count.

- **Other Days:** Generally stable with little variation.

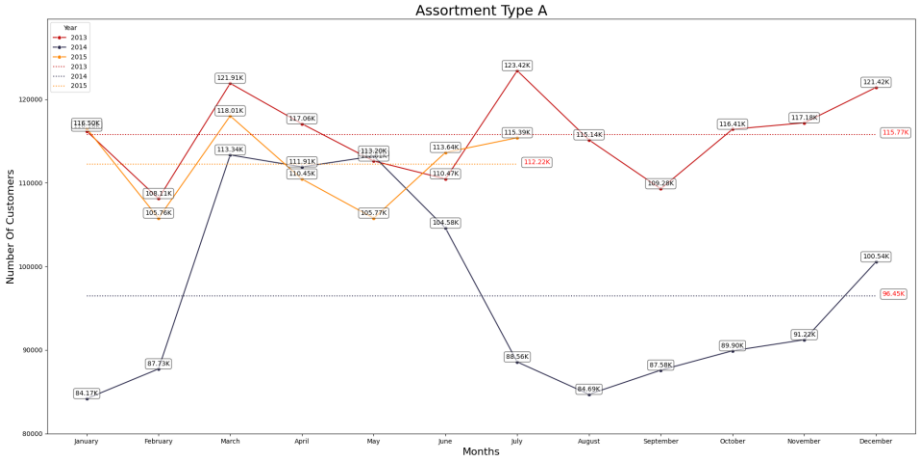
General Trends:
Mondays usually see higher sales, while Saturdays have lower sales. This indicates a shopping peak early in the week, with a decline towards the weekend, especially on Saturdays. No data is available for Sundays due to it being a holiday.

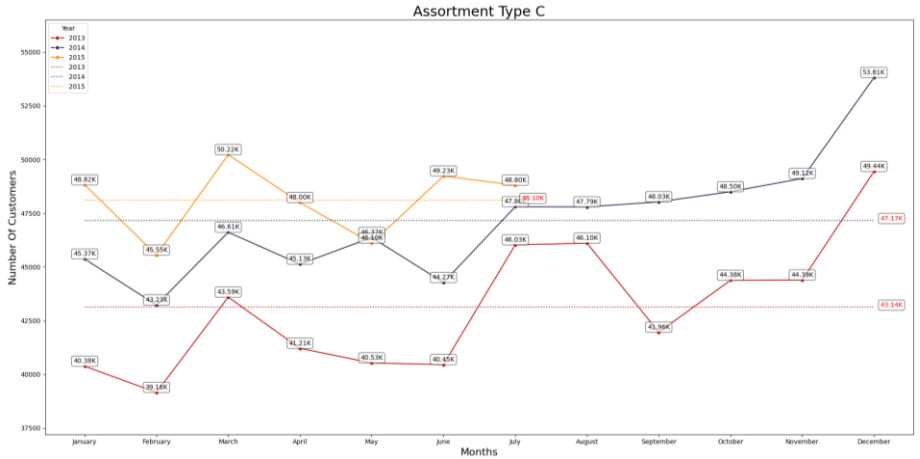


ROSSMANN

Exploratory Data Analyst

NOC Analysis: NOC and Average NOC by Assortment over Three Years





Assortment A:

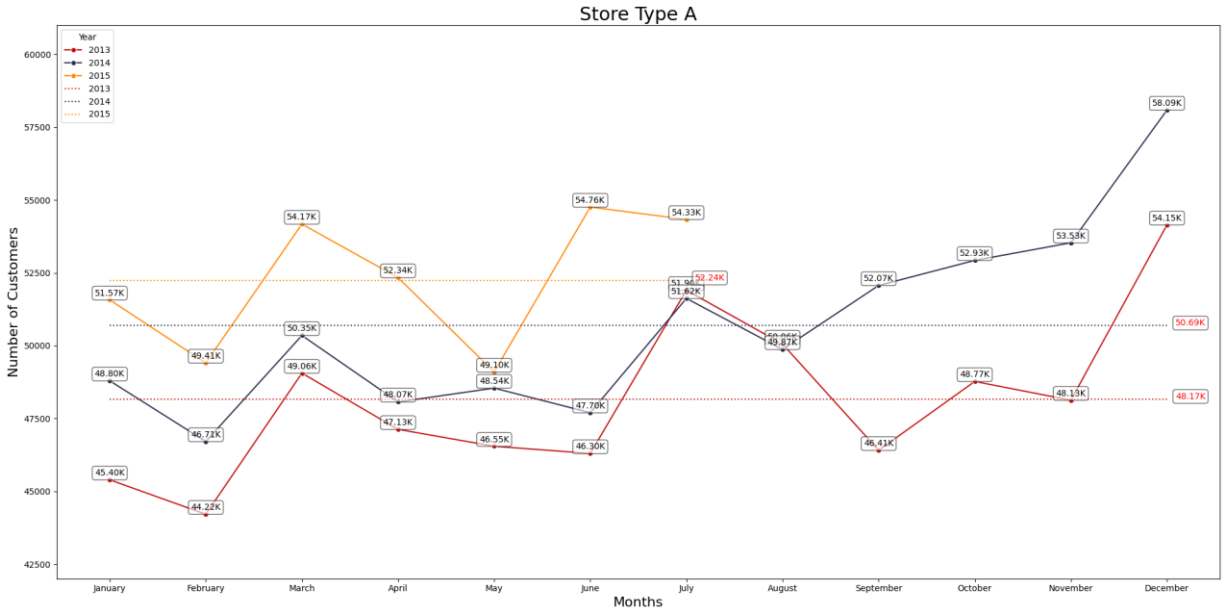
- Customer numbers steadily increased: 115.76K (2013), 96.5K (2014), 112.22K (2015).
- Typically shows strong growth in the last 4 months of the year.

Assortment C:

- Highest number of customers among all assortments.
- Significant growth usually occurs in March and July, with increases in the last 4 months of the year.
- Although there was a recovery in 2015 compared to 2014, it still did not reach 2013 levels.

Exploratory Data Analyst

NOC Analysis: NOC and Average NOC by StoreType over Three Years



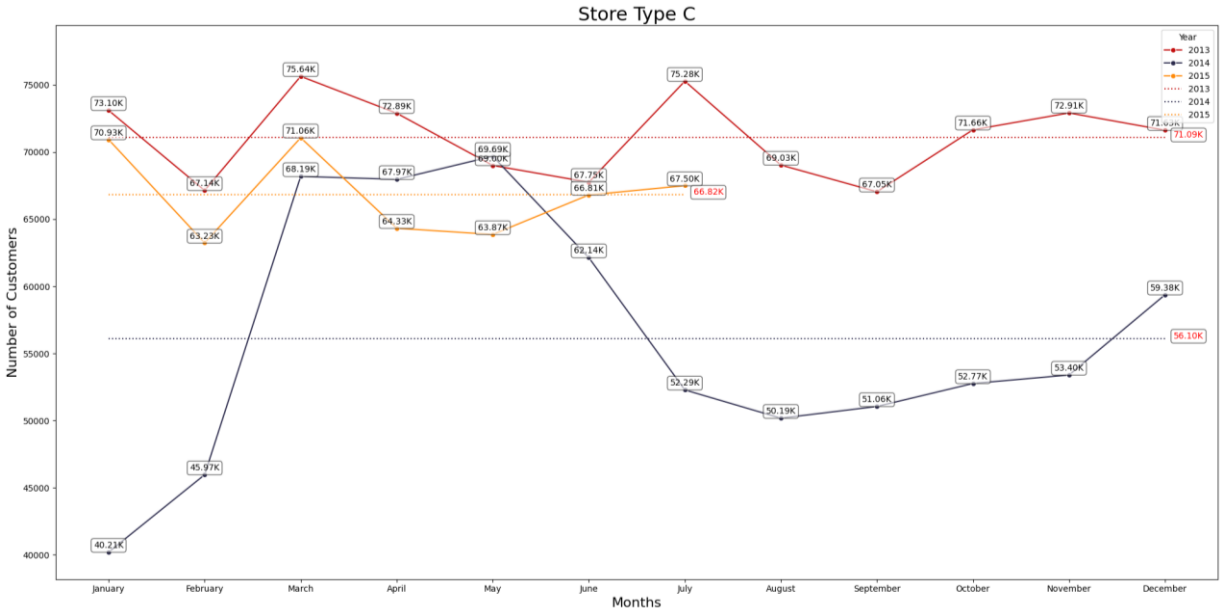
Store Type A:

- Steady increase in customers: 48.17K (2013), 50.69K (2014), 52.24K (2015)
- Strong growth in the last 4 months of the year.

ROSSMANN

Exploratory Data Analyst

NOC Analysis: NOC and Average NOC by StoreType over Three Years



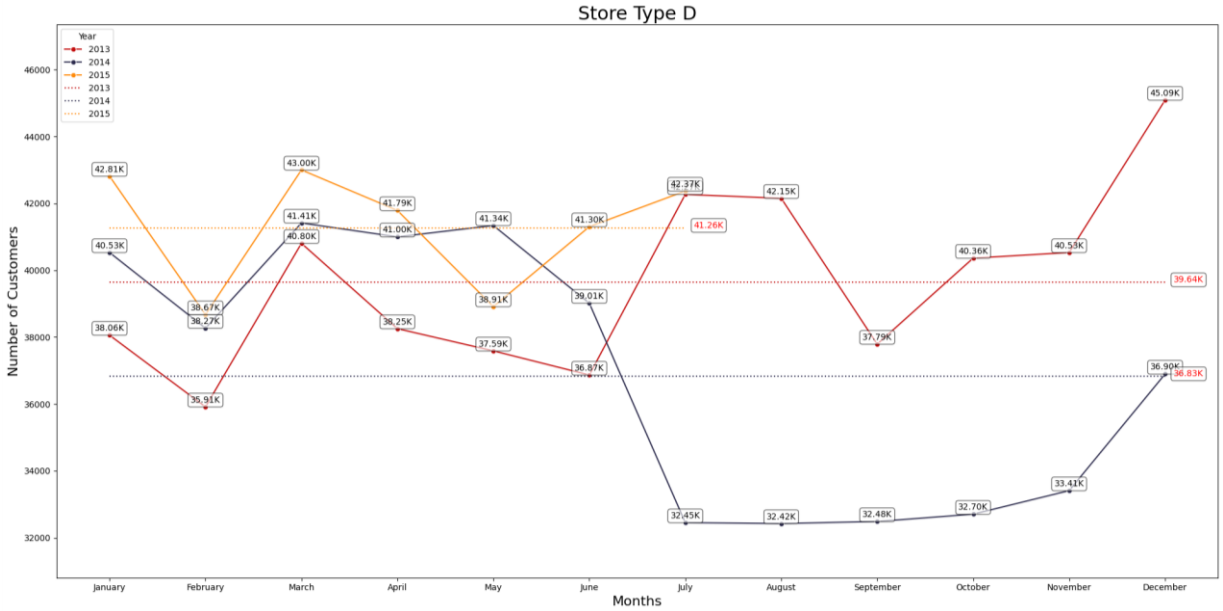
Store Type C:

- Highest number of customers among all types.
- Growth peaks in March and July, with increases in the last 4 months.
- 2015 showed recovery but did not reach 2013 levels.

ROSSMANN

Exploratory Data Analyst

NOC Analysis: NOC and Average NOC by StoreType over Three Years



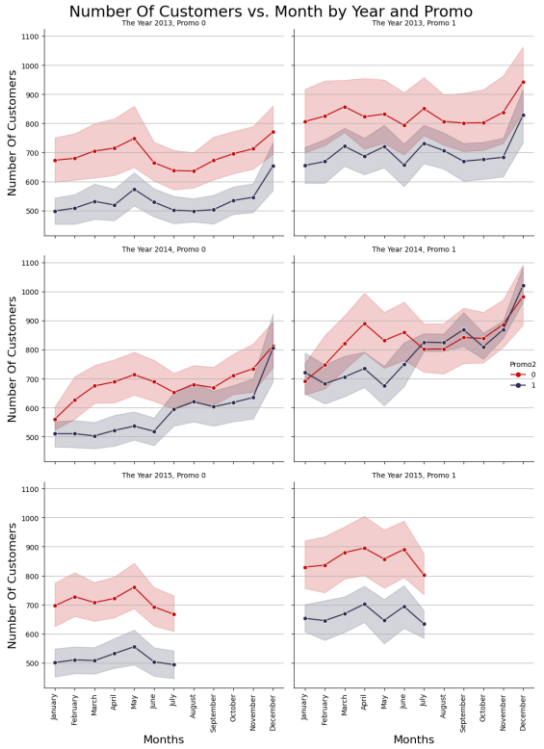
Store Type D:

- Lowest number of customers; relatively stable.
- Decrease in 2014, slight increase in 2015, but still below Type A and C stores.

ROSSMANN

Exploratory Data Analyst

NOC Analysis: NOC Trend During Promotion over Three Years



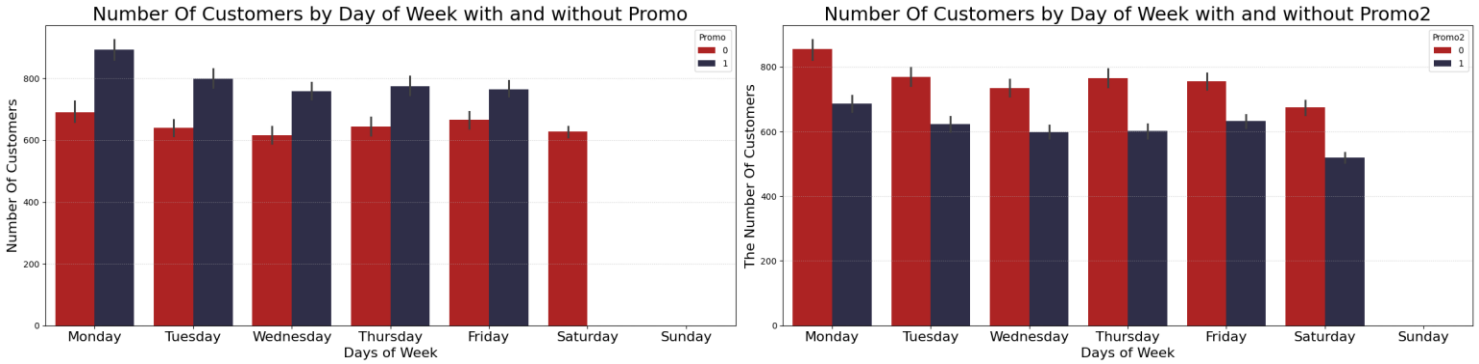
Promo:
- Customer numbers increase gradually throughout the year, peaking in December, whether the store has promotions (Promo = 1) or not (Promo = 0).
- Stores with promotions (Promo = 1) typically have 15% to 20% more customers compared to stores without promotions (Promo = 0).

Promo2:
- Stores with consecutive promotions (Promo2 = 1) generally have fewer customers than those without consecutive promotions (Promo2 = 0).
- This may be due to different store characteristics or a higher need for ongoing promotions to attract customers.

ROSSMANN

Exploratory Data Analyst

NOC Analysis: Average NOC by DayOfWeek with and without Promotion



Promo:

- Stable at 600-640 customers without promotions.
- With promotions, customer numbers rise to around 800, peaking on Monday and decreasing through Friday. No promotions apply to Saturday.

Promo2:

- Highest numbers on Monday, decreasing through the week, with a sharp drop on Saturday. Promo2 stores see a slight decrease compared to non-Promo2 stores.

Insights:

- Promotions boost customer numbers significantly, especially on Mondays.
- Enhance promotional strategies and reconsider approaches for Saturdays.

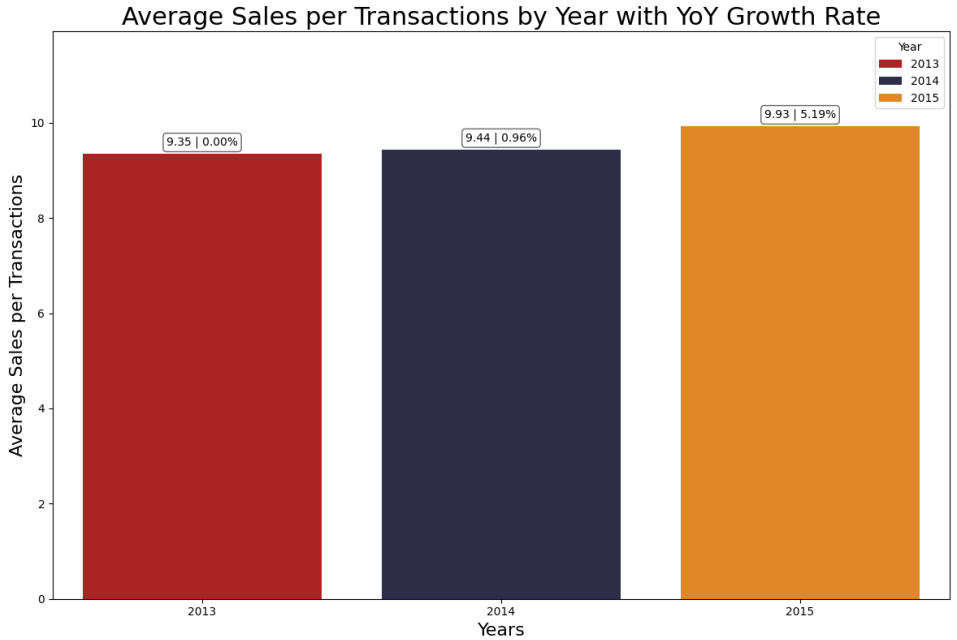
ROSSMANN

Exploratory Data Analyst

AST Analysis: Average Sales per Transaction over Three Years

- Although AST shows growth, this may be misleading. Both total sales and the number of customers have sharply decreased, with the number of customers declining faster than total sales. This results in an increased AST without actual improvement in sales performance.

- Data for 2015 is incomplete (only until July), so the full-year figures may show higher total sales and customer numbers.



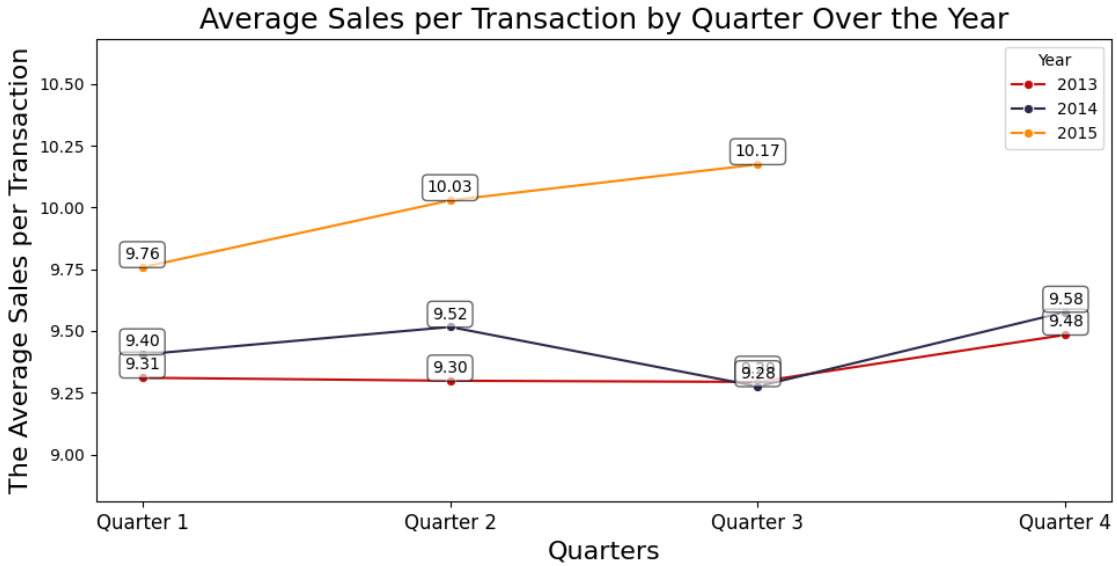
ROSSMANN

Exploratory Data Analyst

AST Analysis: AST by Quarter over Three Years

- Correlation: Total sales and Number of Customers are positively correlated. An increase in customers generally leads to higher sales, and AST follows these trends.

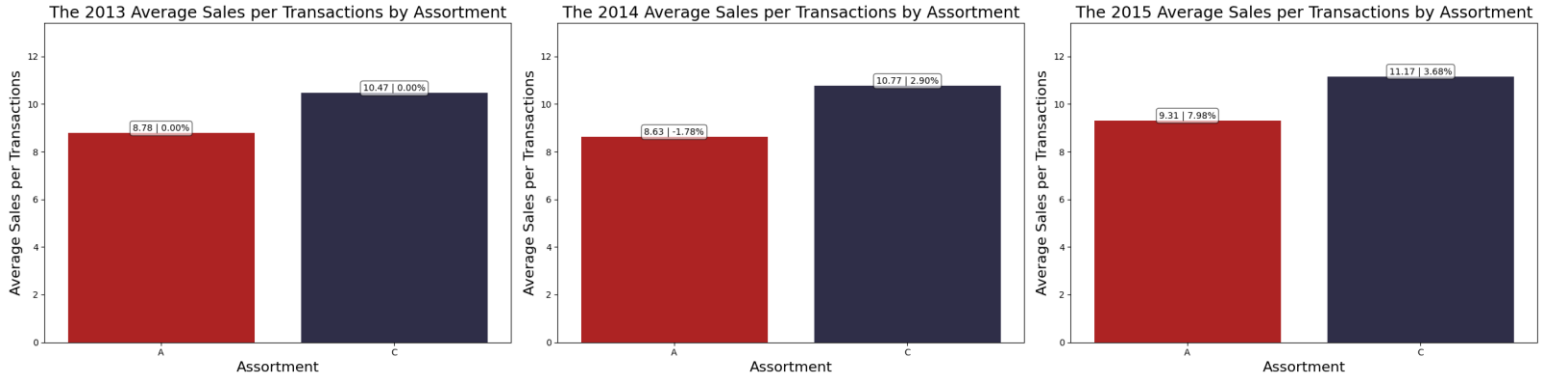
- Q2 2015: Despite a slight decrease in the number of customers (from 485K to 473K, or -2.41%), Total sales stayed at 4.5 million. AST increased by 2.78% (from 9.76% to 10.03%) due to higher sales per order. This indicates that Rossmann maintained revenue despite fewer customers by increasing value per order.



ROSSMANN

Exploratory Data Analyst

AST Analysis: AST by Assortment over Three Years

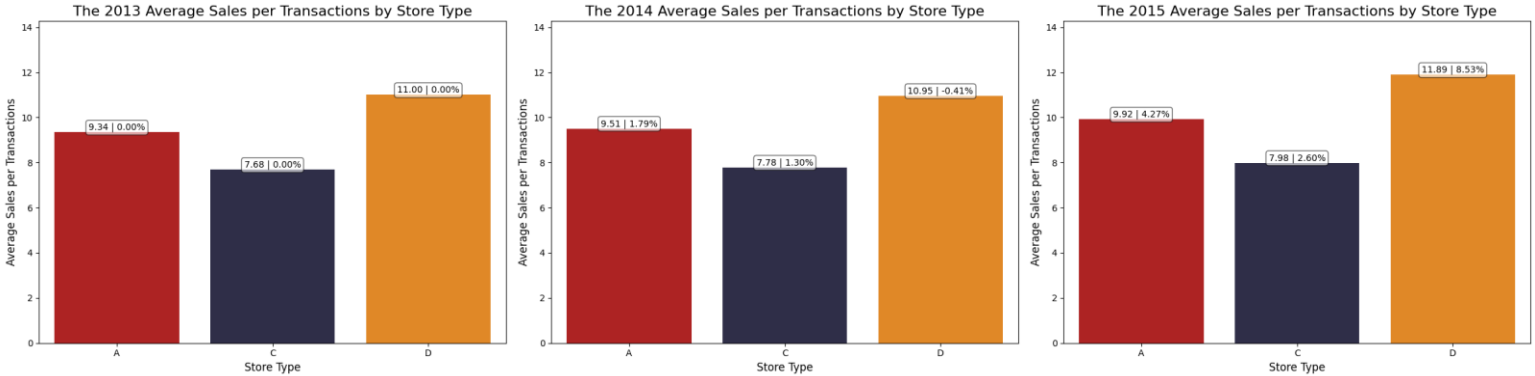


- **Assortment A**: Avg_AST increased in 2015 despite declines in total sales and number of customers, likely due to price increases or higher transaction values. The decline may be due to market saturation, competition, or changing customer demands.

- **Assortment C**: Avg_AST also grew consistently, reflecting stable transaction values. However, total sales and customer numbers dropped significantly in 2015, possibly due to similar factors affecting Assortment A.

Exploratory Data Analyst

AST Analysis: AST by StoreType over Three Years



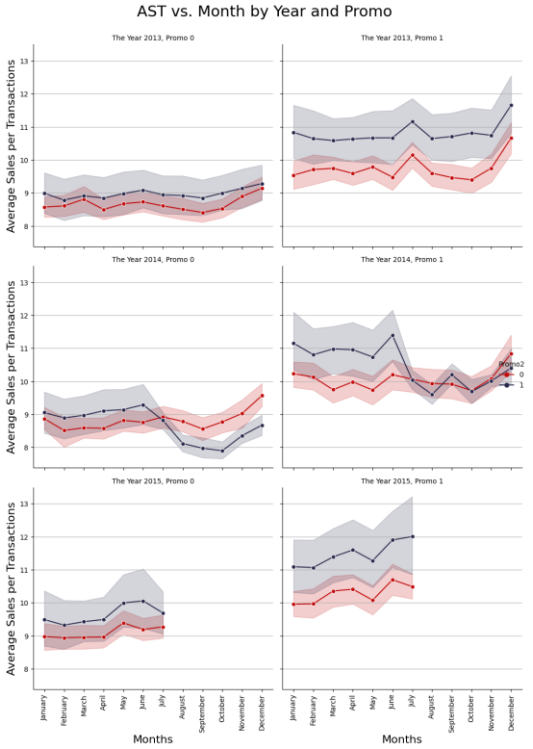
- **Store Type A**: Avg_AST increased from 2013 to 2015, with a 7.55% rise in total sales and a 5.23% increase in customers in 2014, indicating higher transaction values. However, total revenue and customer numbers dropped significantly in 2015, revealing ongoing challenges.

- **Store Type C**: Avg_AST grew steadily over the years. Despite significant decreases in total revenue and customer numbers, the average transaction value remained stable, which is a positive outcome.

- **Store Type D**: Avg_AST decreased slightly in 2014 but surged in 2015, reflecting higher transaction values. Nonetheless, total sales and customer numbers fell significantly, suggesting fundamental issues that need addressing.

Exploratory Data Analyst

AST Analysis: AST Trend During Promotion over Three Years



- Promo: AST is higher with promotions (Promo = 1) compared to without (Promo = 0).Promotions notably boost AST, particularly in June, July, and December.

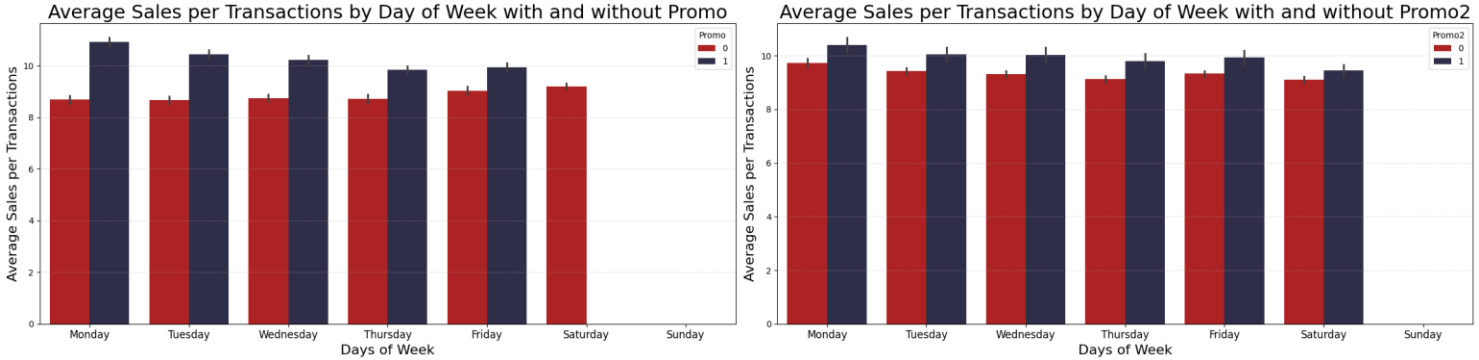
- Promo2: Continuous promotions (Promo2 = 1) generally result in higher AST compared to no Promo2 (Promo2 = 0).Promo2's effect on AST is often positive in the first half of the year and varies in the second half.

Strategic Awareness: Increase promotional activities in June, July, and December to maximize AST. Continuously monitor and analyze advertising impacts to optimize strategies for year-round AST improvement.

ROSSMANN

Exploratory Data Analyst

AST Analysis: AST by DayOfWeek with and without Promotion



- Promo:
AST increases on all days of the week with promotions (Promo = 1), especially on Mondays and Tuesdays, compared to no promotion (Promo = 0).

- Promo2:
Continuous promotions (Promo2 = 1) show less impact on AST compared to Promo but still result in increased AST on some days, notably Mondays.

Insight:
Both Promo and Promo2 positively impact AST, with varying effects based on the type of promotion and day of the week.

Exploratory Data Analyst

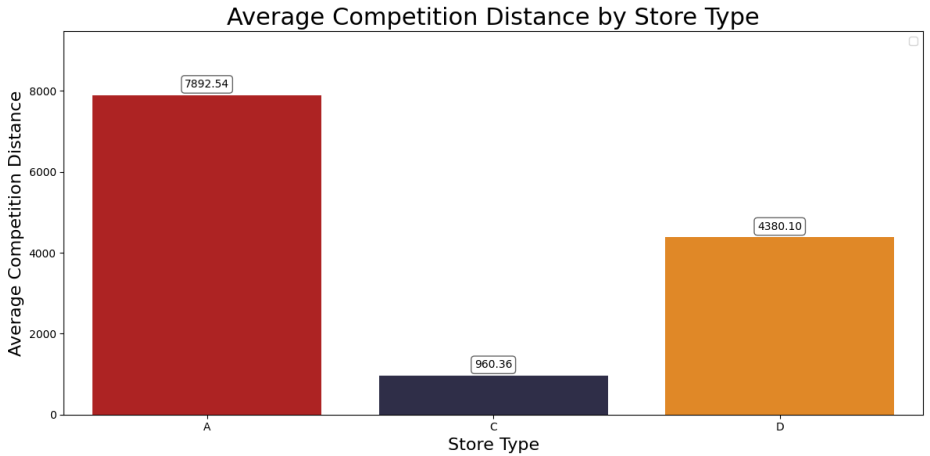
Relationship between StoreType and CompetitionDistance

- Store Type A: Likely located in areas with fewer competitors compared to Types C and D.

- Store Type C: Faces the highest competitive density with the shortest average distance to competitors (around 960 meters).

- Store Type D: Has an average distance to competitors, positioned between Types A and C.

Insight:
Store Type C is typically in more competitive, urban areas, which may require a larger inventory to meet customer needs due to the higher density of nearby stores.



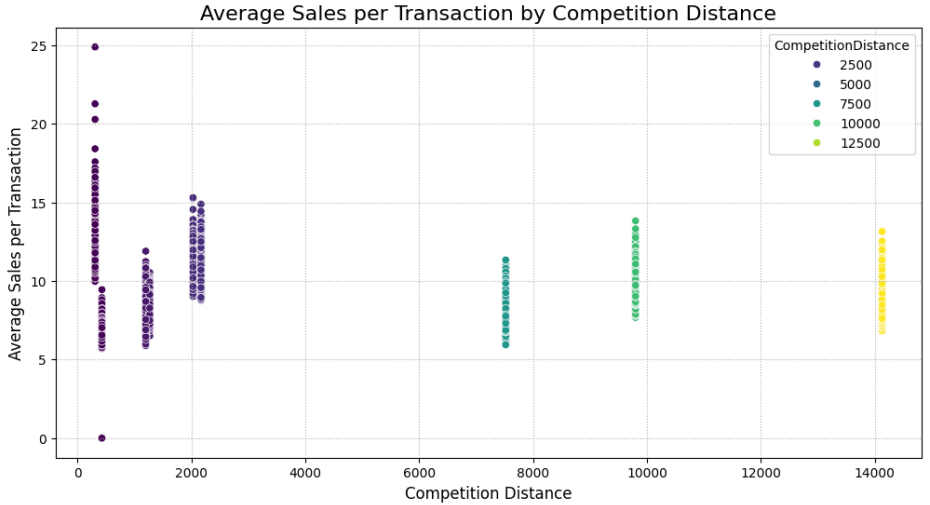
Exploratory Data Analyst

Relationship between StoreType and CompetitionDistance

- Stores close to competitors (0 to 2000 meters) show a wide range in average sales per transaction, from $0 to $20.

- Stores farther from competitors (above 4000 meters) exhibit more stable average sales per transaction, ranging from $5 to $15.

Insight:
Average sales per transaction are generally higher but more variable near competitors, while stores further away have more stable but potentially lower average sales.



ROSSMANN

Exploratory Data Analyst

SUMMARY:

- Sales and number of customers tend to decrease over the years, about 7% of sales and 10% of the number of customers.

- Sales and number of customers often decrease in the second and third quarters, and recover in the fourth quarter (December is usually the month with the strongest growth). March and December are two particularly high growth months of the year.

- For weekdays, Mondays usually have the highest number of customers and sales, gradually decreasing towards the end of the week, and are lowest on Saturdays. Sundays are not counted because stores are closed on this day.

- Sales and number of customers by Assortment: Assortment A often fluctuates greatly, high in March, July, and December every year. Assortment C grows steadily over the years, usually high at the end of the year, especially December.

- Sales and number of customers by Store Type: StoreType A has grown steadily over the years, usually highest in December. The remaining types have fluctuations, but still maintain a relatively stable level.

- Sales and number of customers according to Promo and Promo2: Promo helps these values grow by about 30%, especially on Mondays. On the contrary, Promo2 reduces the values.

- Average Sales per Transaction increasing over the years does not mean sales performance has improved. The number of customers decreased significantly, artificially increasing the AST.

- Sales and number of customers have a positive correlation. The second quarter of 2015 was a bright spot. Although the number of customers decreased slightly, sales still increased (AST also increased), showing an increase in sales per transaction.

- Correlation Insights: Strong correlations exist between sales, customers, and promos.

ROSSMANN

03

Model Prediction

Correlation Check

Based on the data table above, we can see that NumberOfCustomers (87%) is a variable that has a high correlation with the sales target variable. Because the correlation is too high, we consider removing variables to avoid variables affecting the prediction too much and causing the possibility of overfitting. At the same time, the AST variable is a variable created from the NumberOfCustomers variable, so we also consider removing it.
Therefore, we will focus on the remaining variables such as Promo, CompetitionOpenMonth, and Monday,… in the next work. Because they have the highest correlation after NumberOfCustomers.

Correlation Information

Features	The Weights
NumberOfCustomers	0.87
Promo	0.46
CompetitionOpenMonth	0.27
Monday	0.21
Store	0.14
...	...
IsPromoInterval	-0.10
Promo2	-0.17
Saturday	-0.17
Promo2OpenMonth	-0.18
StateHoliday	NaN

Machine Learning

Preparing Data for Model

Main data have 6666 rows, and 23 columns.

Key_store = [1, 3, 8, 9, 13, 25, 29, 31, 46]

Split main data to 2 sets, one for training, and the other for testing.

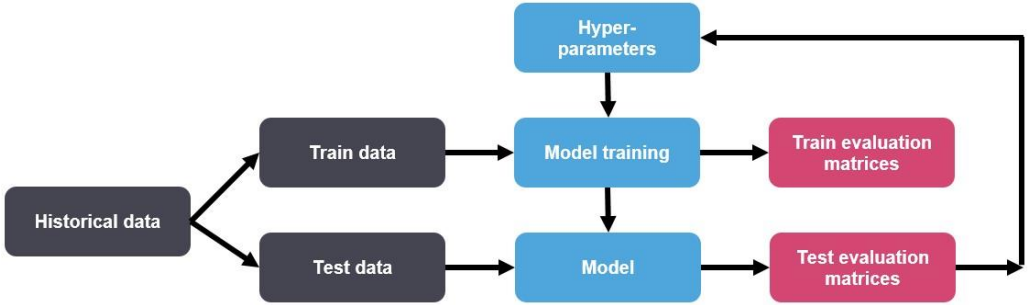
With the data to test, we loop through each store in the main store list, and retrieve the last 36 days of each store. Then connect the data of the stores you just retrieved together into a test dataframe.

With the data to train, we exclude the data already in the test dataframe from the main dataframe.

Correlation Information			
Datasets	Variables	Rows	Columns
Train_df	X_train	6342	22
	y_train	6342	-
Test_df	X_test	324	22
	y_test	324	-

Machine Learning

HyperParameters Tuning



Hyperparameter Tuning is the process of selecting optimal values for parameters set before training a machine learning model.
Proper tuning improves model performance.
Types:

- Manual Tuning: Based on experience and experimentation. Time-consuming but offers control.
- Automated Tuning:
  - Grid Search: Tests all combinations in a specified range.
  - Random Search: Randomly selects combinations
  - Bayesian Optimization: Uses a probabilistic model for efficient search.

Importance: Enhances accuracy and prevents overfitting or underfitting. Use with techniques like cross-validation for best results.

Machine Learning

HyperParameters Tuning: Create Function

The test_params function is used to evaluate machine learning models with different hyperparameters. It performs the following steps:

1. Model Initialization: Initializes the provided model with specified hyperparameters, using all available CPU cores and ensuring reproducibility.
2. Training: Trains the model on the training dataset.
3. Performance Evaluation: Computes the Root Mean Square Error (RMSE) for both the training and testing datasets.
4. Results Display: Prints the RMSE values for comparison, aiding in model performance assessment and hyperparameter tuning.

This function is useful for systematically testing and optimizing models to achieve better predictive accuracy.

Machine Learning

Evaluating Models: Create Function

The “evaluate_models” function is used to evaluate the performance of a prediction model. This function takes the model name, actual values (y_test), and predicted values (y_pred). The steps performed by the function are as follows:

1. Define column names: Create a list of column names to store the evaluation metrics.

2. Calculate evaluation metrics:
   - R2 Score: Measures the goodness of fit of the model to the actual data.
   - Mean Absolute Error (MAE): Measures the average absolute difference between predicted values and actual values.
   - Mean Square Error (MSE): Measures the average squared difference between predicted values and actual values.
   - Root Mean Square Error (RMSE): Measures the square root of MSE to bring the error back to the same unit as the original data.

3. Create DataFrame: Create a DataFrame containing these evaluation values, with the column names defined in the first step.

4. Return DataFrame: Return the DataFrame containing the evaluation metrics for the model.

Machine Learning

Random Forest Regressor: Define

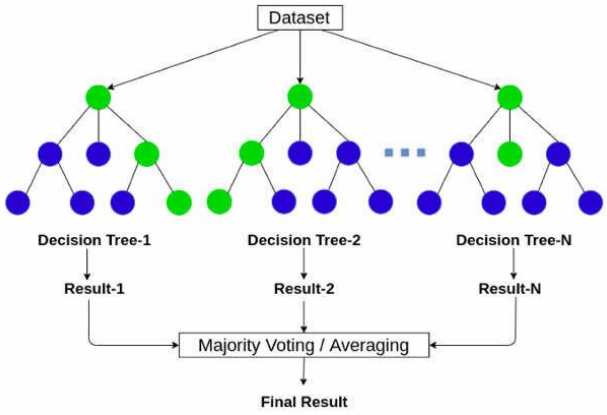
Overview: The Random Forest Regressor is an ensemble model that builds multiple decision trees and combines their predictions to improve accuracy and reduce overfitting.

How it Works:

1) Multiple Trees: Trains several decision trees on different bootstrap samples of the data.
2) Aggregation: Averages predictions from all trees for the final output.
3) Random Feature Selection: Each tree uses a random subset of features to determine splits.

Benefits:

1) High Accuracy: Effective for predicting complex and fluctuating data like revenue.
2) Handles Non-Linearity: Manages non-linear relationships well.
3) Reduces Overfitting: Combats overfitting, especially in volatile data.
4) Manages Temporal Features: Automatically finds important relationships in time series data.
1) Scalable and Flexible: Adapts to more data and integrates with other techniques.



Machine Learning

Random Forest Regressor: HyperParameters Tuning

We use the "test_params" function we created earlier for HyperParameters Tuning, which is useful for systematically testing and optimizing models to achieve better prediction accuracy.

The table below is the tuned range, and the evaluation results on the train and test data sets:

HyperParameters Tuning Results			
Parameters	Range	RMSE on Train	RMSE on Test
n_estimators	300-650	318-316	1084-1082
max_depth	5-10	1160-636	1080-1061
min_samples_split	50-65	874-910	989-990
min_samples_leaf	5-8	694-794	970-972

Machine Learning

Random Forest Regressor: Training and Evaluating Model

On the right is the parameter table of the trained model.

And the table below contains the evaluation results of the prediction model on the test dataset.

Evaluating Models				
Index	R2 Score	MAE	MSE	RMSE
Random Forest Regressor	0.83	706.86	980610.21	990.26

HyperParameters	
Best Parameters	Values
bootstrap	False
ccp_alpha	0.0
criterion	Squared_error
max_depth	10
max_features	log2
max_leaf_nodes	None
...	...
n_estimators	400
n_jobs	None
oob_score	False
random_state	None
verbose	0
warm_start	False

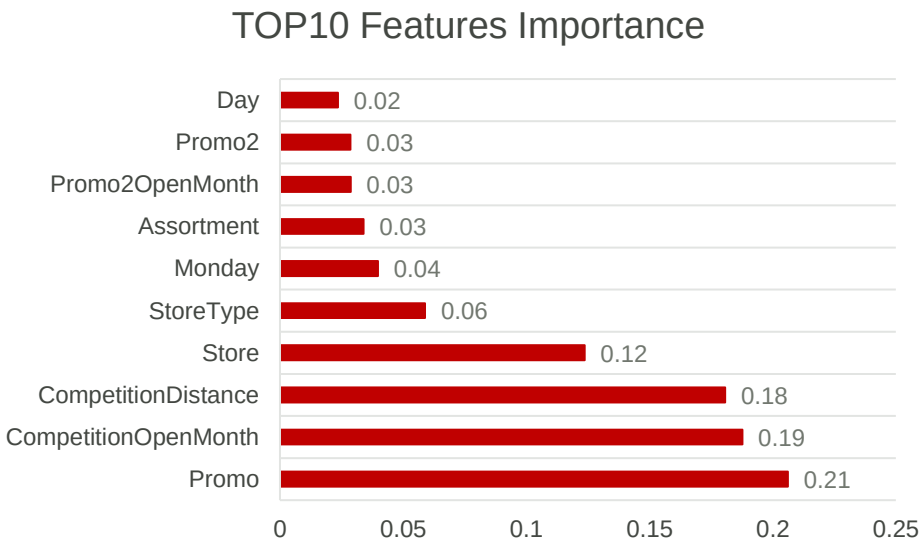
Machine Learning

Random Forest Regressor: Features Importance

- Promo: This feature is of the highest importance, accounting for a very large proportion in predicting sales.

- CompetitionOpenMonth, CompetitionDistance and Store: these features have almost equal importance, but their contribution is still significantly smaller than 'Promo'. only ranked 2nd, 3rd, and 4th after Promo.

- The remaining characteristics are also of relative importance. This shows that they have an impact on Sales prediction, but not greatly.



ROSSMANN

Machine Learning

XGBoost Regressor: Define

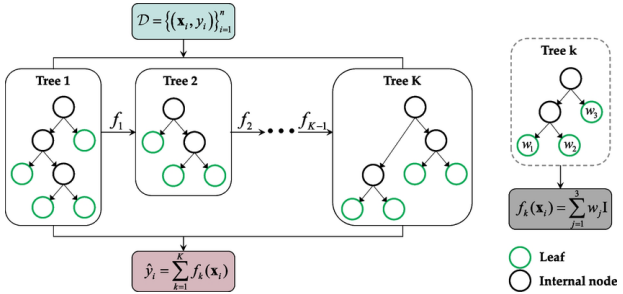
Overview: XGBoost Regressor is a high-performance machine learning model based on Gradient Boosting Machines (GBM), developed to enhance accuracy and speed.

How it Works:

1) Boosting Algorithm: Combines weak learners (simple decision trees) to form a stronger model.
2) Gradient Boosting: Iteratively optimizes a loss function to reduce prediction errors.
3) Regularization: Uses L1 and L2 regularization to control overfitting.
4) Parallelization: Efficiently runs on multiple CPU cores for faster training.
5) Hyperparameters: Offers extensive tuning options to improve performance.

Benefits:

1) High Performance: Provides accurate and stable results, even on large datasets.
2) Regularization Support: Helps prevent overfitting.
3) Fast Training: Optimized for speed and efficiency.
4) Versatile: Suitable for various prediction and classification tasks.



Machine Learning

XGBoost Regressor: HyperParameters Tuning

We use the "test_params" function we created earlier for HyperParameters Tuning, which is useful for systematically testing and optimizing models to achieve better prediction accuracy.

The table below is the tuned range, and the evaluation results on the train and test data sets:

HyperParameters Tuning Results			
Parameters	Range	RMSE on Train	RMSE on Test
n_estimators	25-55	607-463	918-897
max_depth	5-10	471-41	943-1028
learning_rate	0.1-0.15	544-899	477-915
subsample	0.5-0.7	396-353	974-979
colsample_bytree	0.6-0.9	390-354	892-908
min_child_weight	1-5	356-380	904-906

Machine Learning

XGBoost Regressor: Training and Evaluating Model

On the right is the parameter table of the trained model.

And the table below contains the evaluation results of the prediction model on the test dataset.

Evaluating Models				
Index	R2 Score	MAE	MSE	RMSE
XGBoost Regressor	0.86	663.32	835501.55	914.06

HyperParameters	
Best Parameters	Values
objective	squarederror
base_score	None
booster	None
callbacks	None
colsample_bylevel	None
colsample_bynode	None
...	...
feature_types	None
gamma	0.0069
subsample	0.8895
tree_method	None
validate_parameters	None
verbosity	None

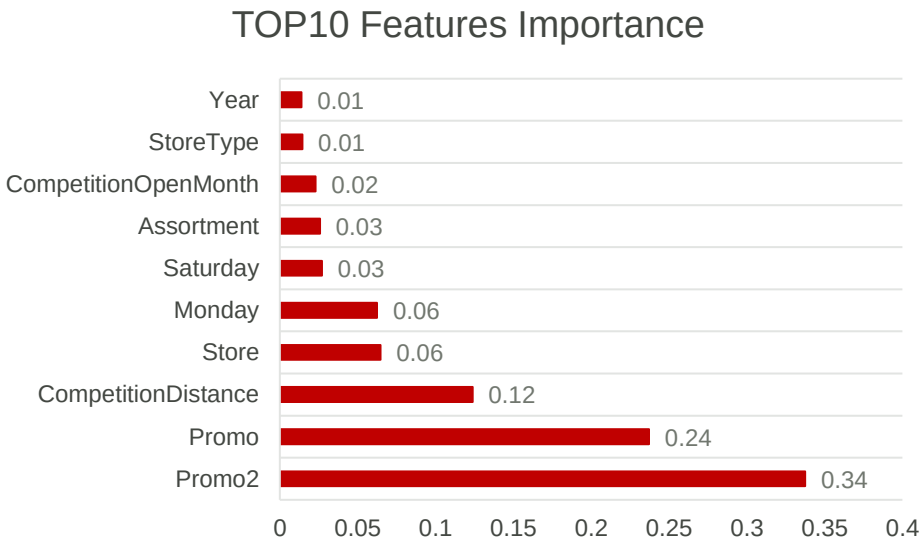
Machine Learning

XGBoost Regressor: Features Importance

- The feature 'Promo2' takes on the greatest importance, showing that the Promo2 is the main factor determining sales.

- 'Promo', and 'CompetitionDistance' are also of significant importance, ranked 2nd, and 3rd after 'Promo2'.

- The remaining features are of low importance, but still contribute to Sales prediction. This shows that these factors also have an impact, although not large.



Machine Learning

LightGBM Regressor: Define

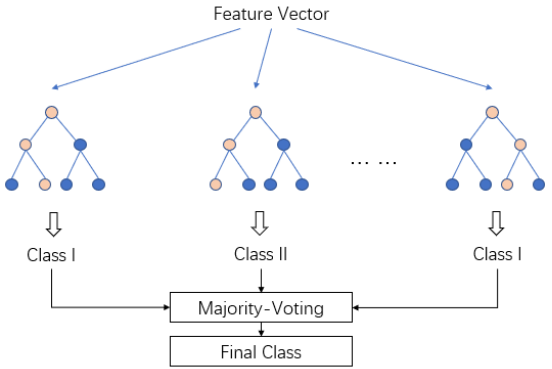
Overview: LightGBM (Light Gradient Boosting Machine) is a fast and efficient Gradient Boosting algorithm developed by Microsoft, known for handling large datasets effectively.

How it Works:

1) Gradient Boosting: Builds models by iteratively optimizing a loss function with decision trees.
2) Leaf-wise Growth: Uses a leaf-wise growth strategy for deeper, sparser trees, optimizing splits based on loss reduction.
3) Regularization: Includes techniques like pruning to prevent overfitting.
4) Parallel and GPU Computing: Supports parallel processing and GPU acceleration for faster training.
5) Categorical Features: Directly handles categorical features without needing encoding.

Benefits:

1) High Performance: Efficiently handles and trains on large datasets.
2) Big Data Handling: Optimized for large-scale data with GPU support.
3) Flexibility: Applicable to both regression and classification tasks.



Machine Learning

LightGBM Regressor: HyperParameters Tuning

We use the "test_params" function we created earlier for HyperParameters Tuning, which is useful for systematically testing and optimizing models to achieve better prediction accuracy.

The table below is the tuned range, and the evaluation results on the train and test data sets:

HyperParameters Tuning Results			
Parameters	Range	RMSE on Train	RMSE on Test
n_estimators	100-500	611 – 391	921-885
max_depth	5-10	680-613	905-927
num_leaves	10-80	793-491	963-925
learning_rate	0.1-0.35	611-449	921-892
colsample_bytree	0.6-0.9	626-618	948-931
min_child_samples	5-35	584-635	921-919

Machine Learning

LightGBM Regressor: Training and Evaluating Model

On the right is the parameter table of the trained model.

And the table below contains the evaluation results of the prediction model on the test dataset.

Evaluating Models				
Index	R2 Score	MAE	MSE	RMSE
LightGBM Regressor	0.85	681.11	859962.06	927.34

HyperParameters	
Best Parameters	Values
boosting_type	gbdt
class_weight	None
colsample_bytree	0.8745
importance_type	split
learning_rate	0.2548
max_depth	5
...	...
num_leaves	10
objective	None
reg_alpha	0.4774
reg_lambda	0.3689
subsample	0.7663
subsample_for_bin	200000

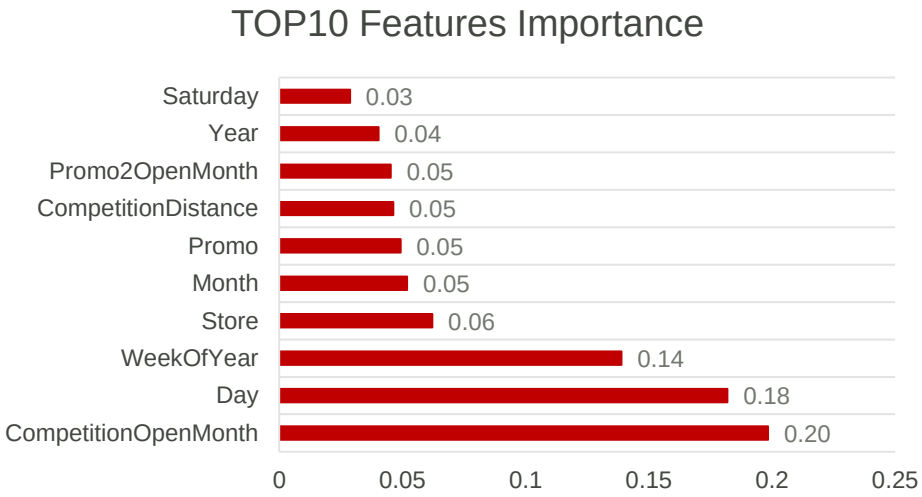
Machine Learning

LightGBM Regressor: Features Importance

- The feature 'CompetitionOpenMonth' takes on the greatest importance, showing that the 'CompetitionOpenMonth' is the main factor determining sales.

- 'Day', and 'WeekOfYear' are also of significant importance, only ranked second and third after 'CompetitionOpenMonth' a little.

- The remaining features are of low importance, but still contribute to Sales prediction. This shows that these factors also have an impact, although not large.



Deep Learning

Long Short Term Memory(LSTM): Define

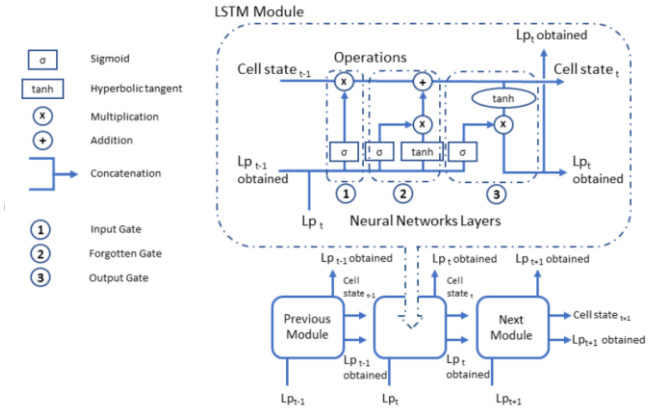
Overview: LSTM is a type of recurrent neural network (RNN) designed to handle time-series data and mitigate issues like vanishing gradients in traditional RNNs.

How it Works:

1) Cell State and Gates: Uses a cell state to maintain and transfer information across time steps with gate mechanisms to control updates.
2) Forget Gate: Decides what information to discard from the cell state using a sigmoid function.
3) Input Gate: Chooses new information to add to the cell state, using sigmoid and tanh functions.
4) Output Gate: Determines the output based on the updated cell state.

Benefits:

1) Complex Pattern Learning: Remembers past information to predict future values, handling complex temporal patterns.
2) Non-linear Relationships: Captures non-linear relationships in time-series data.
3) Sparse Data Handling: Effectively manages missing data by maintaining and updating the cell state.



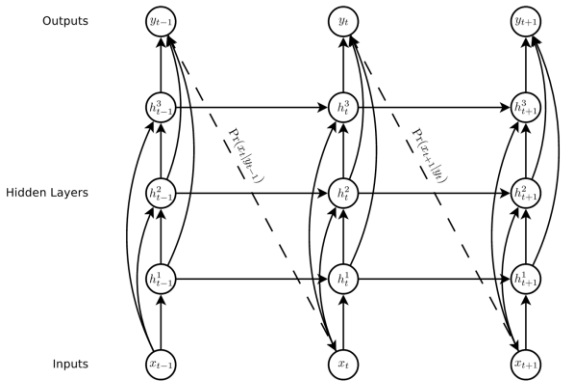
Deep Learning

Long Short-Term Memory: Build the Model

LSTM Model overview:
1. Architecture:
- Layers: three bidirectional LSTM layers with 128, 64, and 32 units, respectively.
- Dropout: added with a rate of 0.2 to prevent overfitting
- Output: final dense layer with 1 unit for predicting sales

2. Compilation:
- optimizer: adam
- Loss function: mean squared error.
- Metric: mean absolute error

3. Training:
- epochs: 100
- Batch size: 32
- Validation: 20% splits.



Deep Learning

Long Short-Term Memory: Training and Evaluating Model

The model has quite high r2 on both training and testing sets, showing that the model has captured the trends and factors affecting sales.

However, the model appears to perform better on the training set, which suggests that the model may have overlearned the characteristics of the training data, leading to poor generalization on new data. And the values of evaluation indices such as MAE, MSE, RMSE on the test set are still quite large, showing that the model's predictive ability is not really high.

Evaluating Long Short-Term Memory Models				
Index	R2 Score	MAE	MSE	RMSE
Train Data	0.88	560.91	810436.67	900.24
Test Data	0.7	988.82	1727934.65	1314.51

Comparating Models

1) XGBoost and LightGBM: These two models have the best performance with the highest R^2 Score and lowest error metrics. This shows they are capable of capturing complex relationships in data and making more accurate predictions.

2) Random Forest: This model also has quite good performance, but not as good as XGBoost and LightGBM.

1) LSTM: This model has the worst performance compared to the remaining models. This may be because your data is not yet suitable for applying LSTM (for example, there is no clear temporality or no trend over time).

=> XGB Regressors is the best model for predicting sales in this context, while LSTM performs the worst.

Evaluating Models				
Index	R2 Score	MAE	MSE	RMSE
Random Forest Regressor	0.83	706.86	980610.21	990.26
XGBoost Regressor	0.86	663.32	835501.55	914.06
LightGBM Regressor	0.85	681.11	859962.06	927.34
Long Short-Term Memory	0.7	988.83	1727934.65	1314.51

Predicting Future Daily Sales

We find the most suitable model for prediction based on 2 criteria:

- R2 Score: model has the highest R2 Score
- RMSE: the model has the lowest index

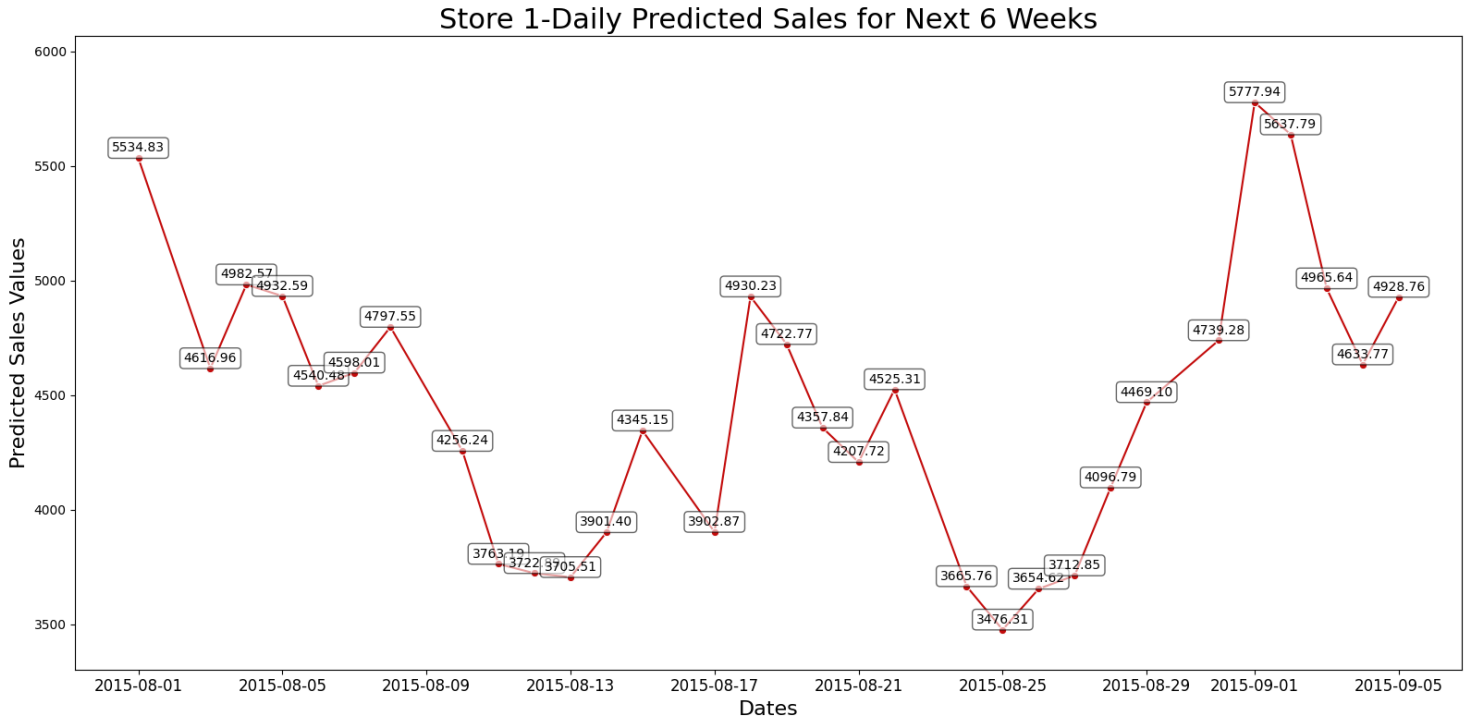
As a result, the XGBoost Regressor model is a model that meets both criteria. We use this model to predict daily sales for the next 36 days.

Evaluating Models				
Index	R2 Score	MAE	MSE	RMSE
XGBoost Regressor	0.86	663.32	835501.55	914.06

HyperParameters	
Best Parameters	Values
objective	squarederror
base_score	None
booster	None
callbacks	None
colsample_bylevel	None
colsample_bynode	None
...	...
feature_types	None
gamma	0.0069
subsample	0.8895
tree_method	None
validate_parameters	None
verbosity	None

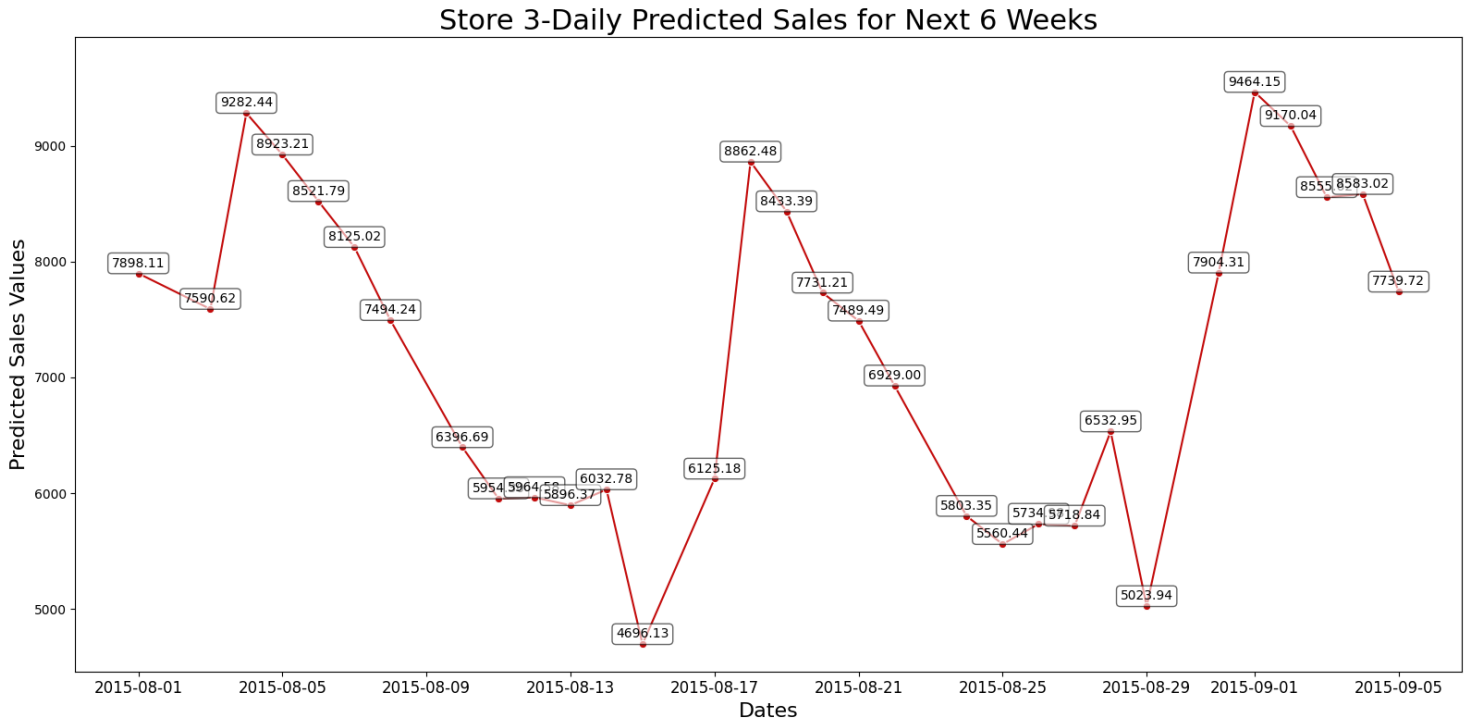
Visualization Future Daily Sales

Store 1:



Visualization Future Daily Sales

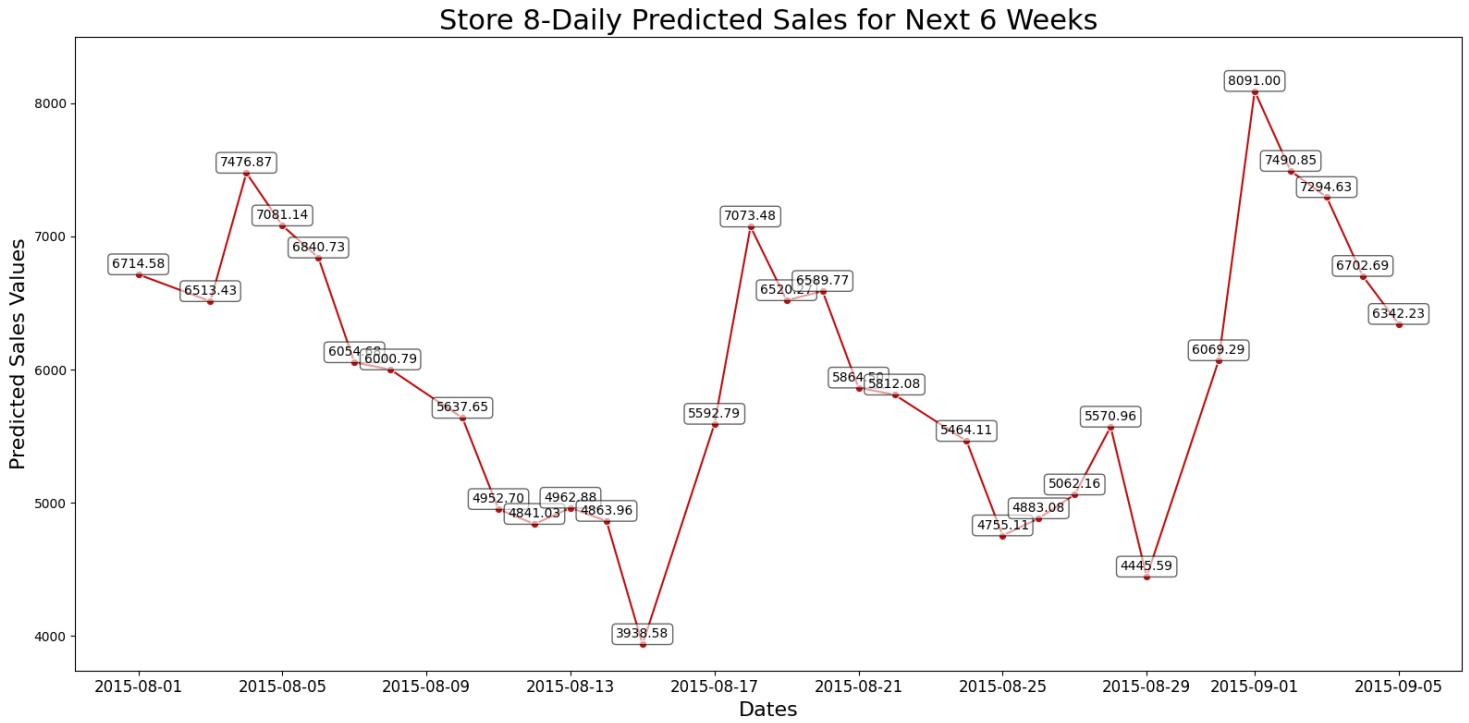
Store 3:



ROSSMANN

Visualization Future Daily Sales

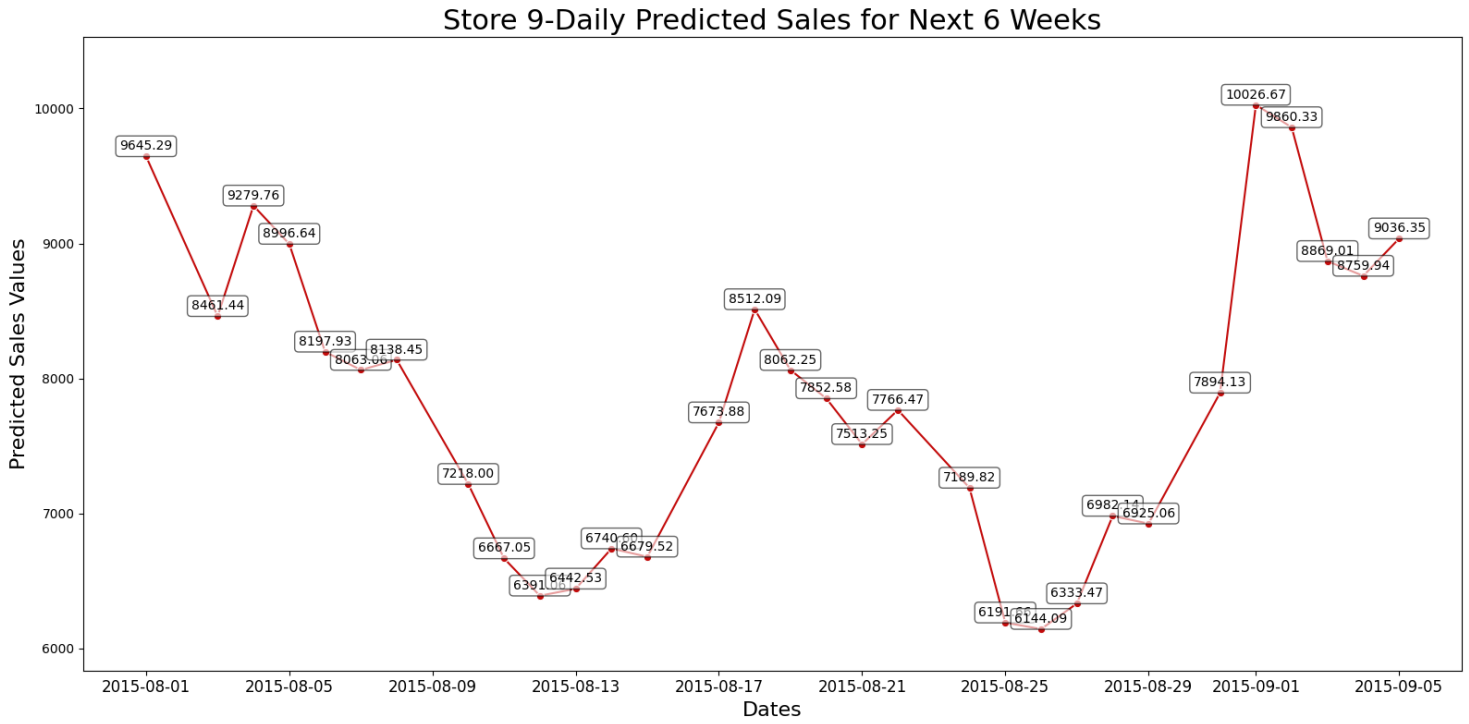
Store 8:



ROSSMANN

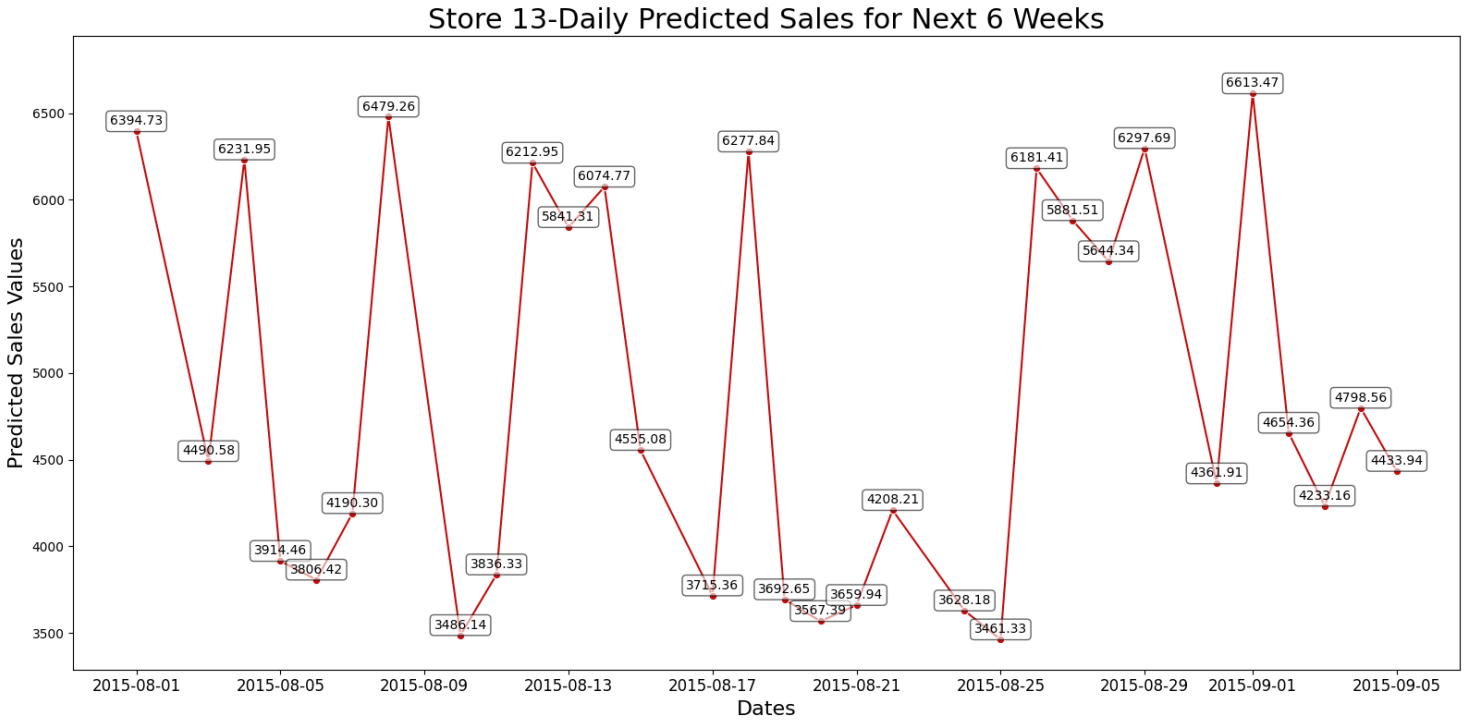
Visualization Future Daily Sales

Store 9:



Visualization Future Daily Sales

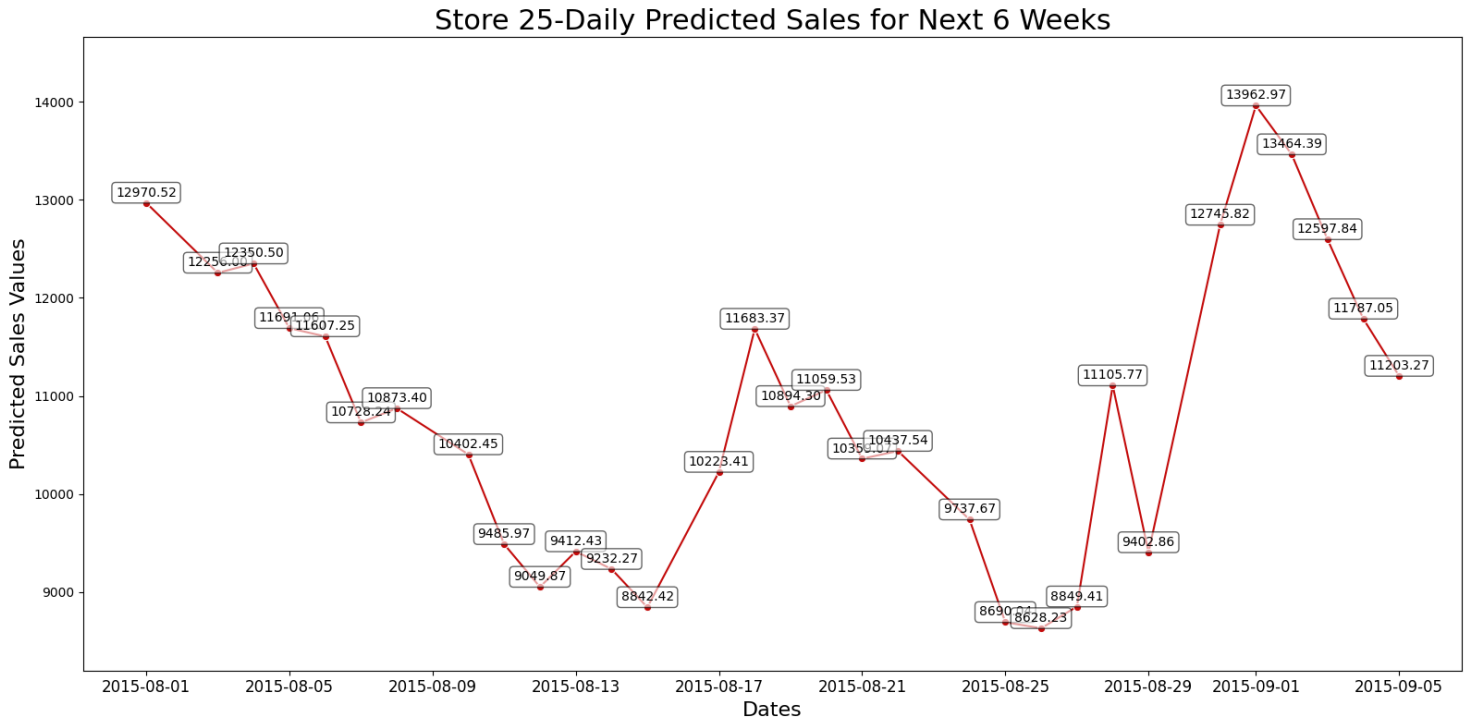
Store 13:



ROSSMANN

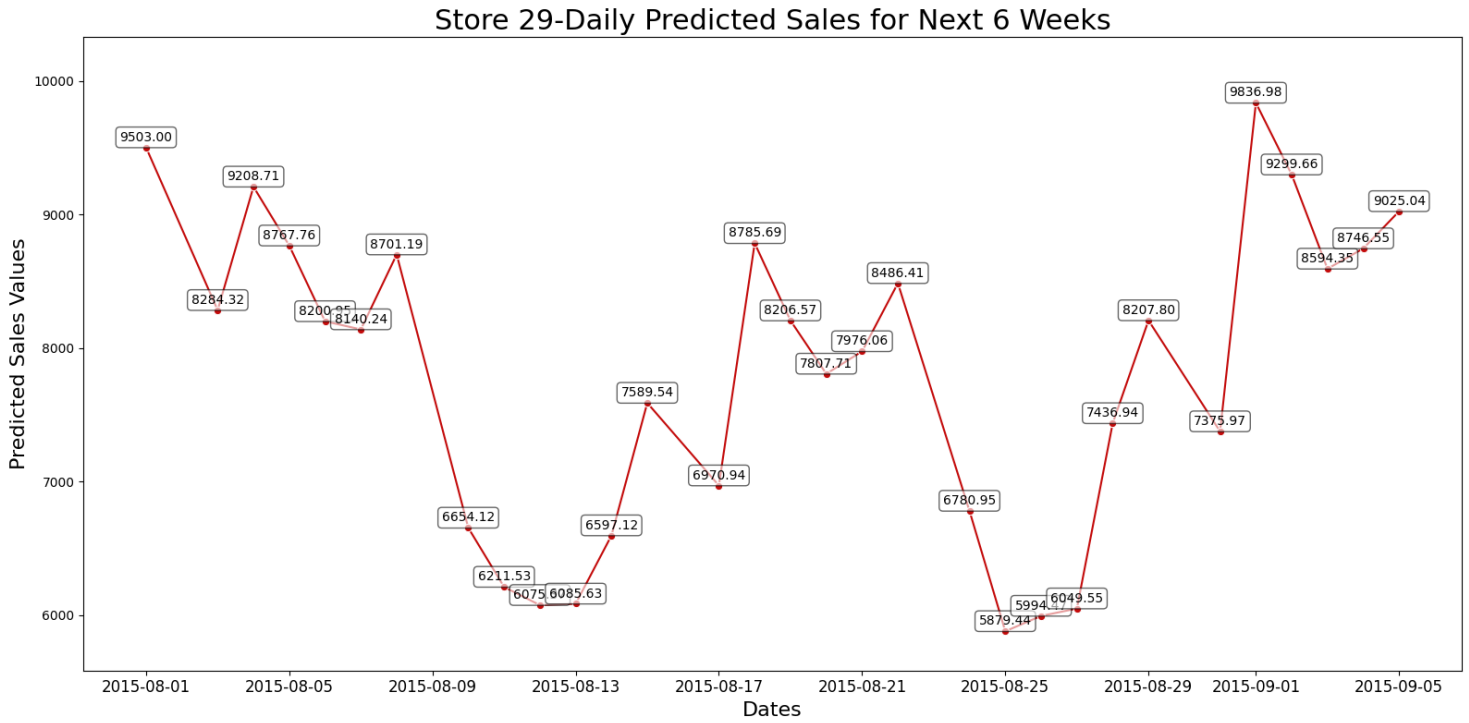
Visualization Future Daily Sales

Store 25:



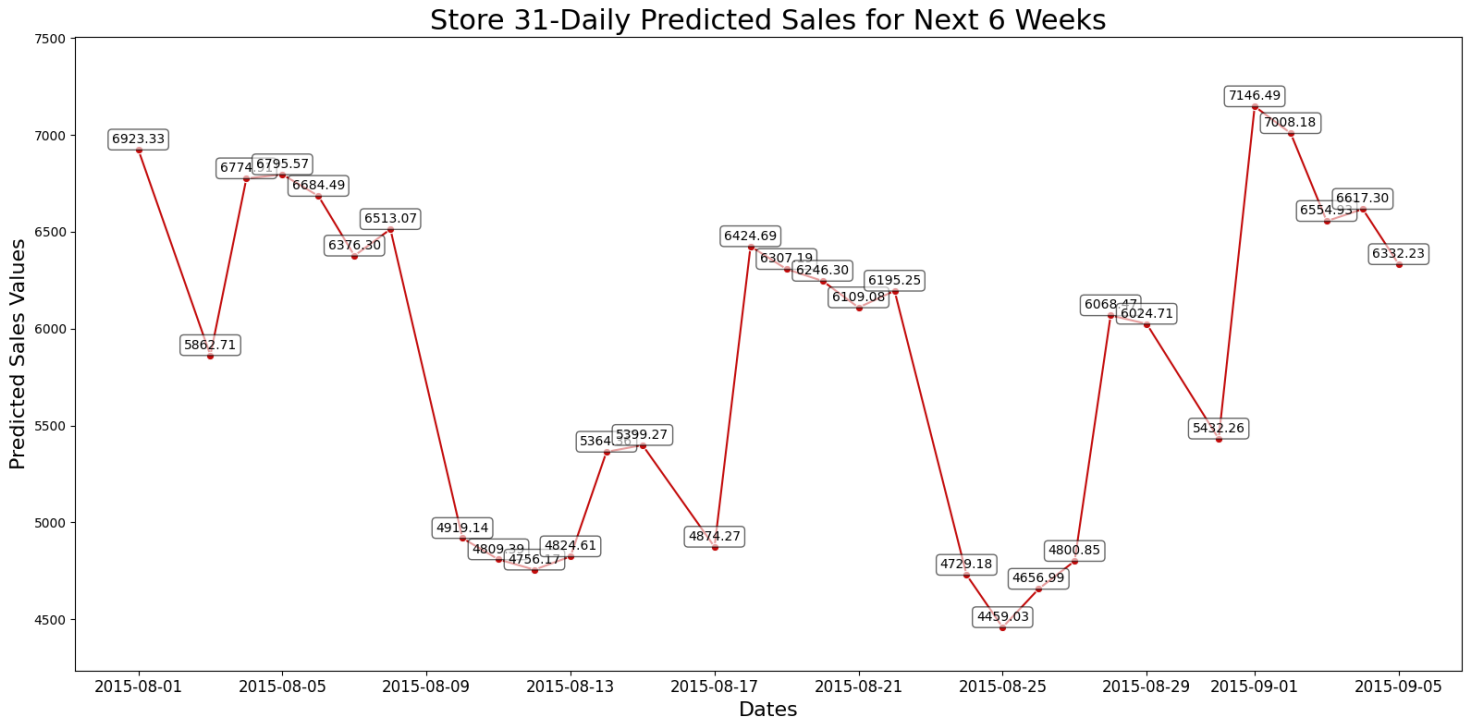
Visualization Future Daily Sales

Store 29:



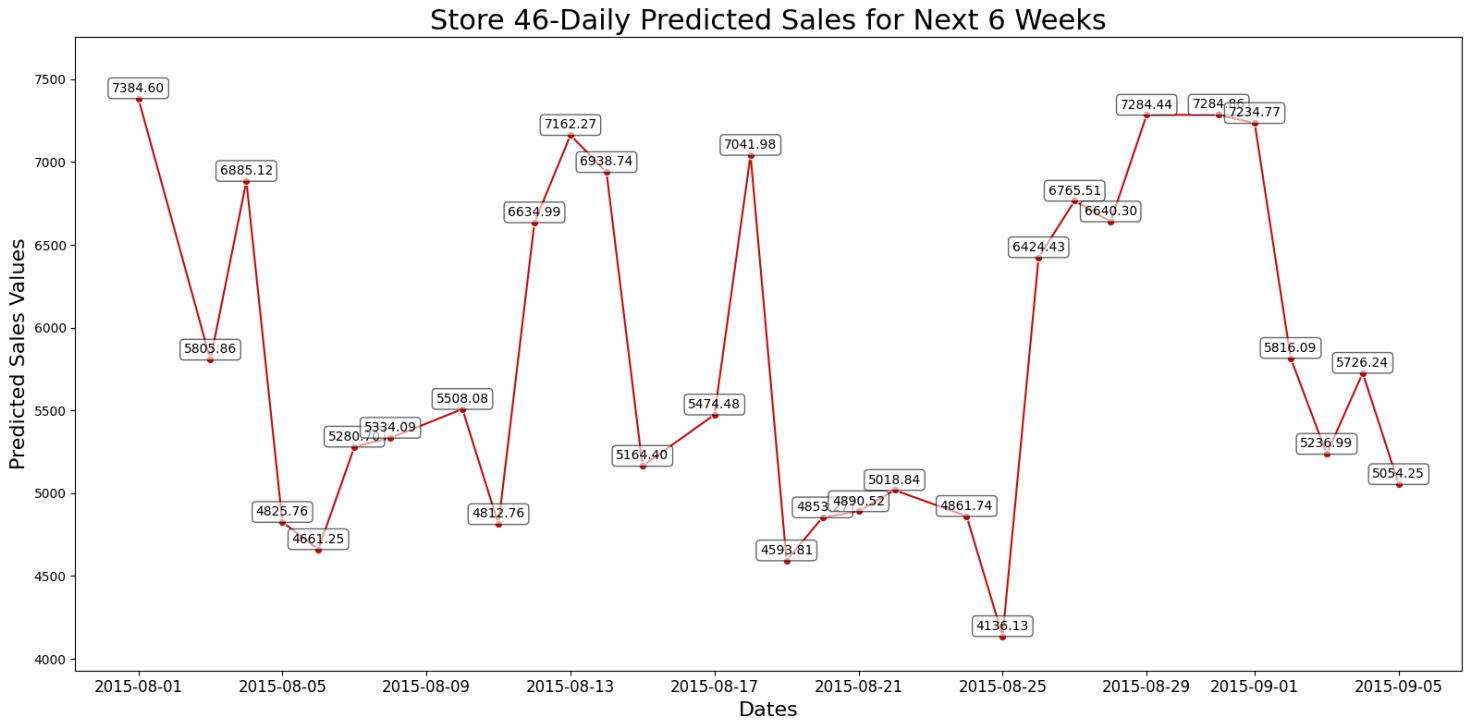
Visualization Future Daily Sales

Store 31:



Visualization Future Daily Sales

Store 46:



ROSSMANN

ROSSMANN

04

Suggestion

Suggestion

1. Prioritize using XGBoost and LightGBM: 2 models show superior performance compared to the remaining models. Hyperparameters can be fine-tuned to further improve performance.
2. Re-evaluate the LSTM model:
    - Check the data again: ensure the data has clear timing and trends over time.
    - Adjust network structure: Try changing the number of hidden layers, number of neurons, activation function,...
    - Try some other models: other RNN models like GRU
3. Build ensemble: combine predictions of the best models (XGBoost, LightGBM) to get better prediction results.
4. Analyze feature importance: determine the factors that have the biggest impact on sales, focus on these factors when making decisions.
5. Evaluate the test data set: ensure the model's performance is not only good on the training data set but also good on the unprecedented data set.
6. In addition, review data quality, data preprocessing, hyperparameters tuning, Interpretability,...

Thanks!

Do you have any questions?

nhudaitran1510@gmail.com
+84 98 629 364



Responsible for Content: Walter Edward