Springboard – DSC
Capstone Project III

Sales Analyzing and Weekly Sales Forecasting for Drug Store Rossmann

Final Report

Yang Liu Kunz

Nikoloz Skhirtladze (Mentor)

Data Science Career Track

January 2021

Table of Contents

1 Introduction *3*

1.1 Objective **3**

1.2 Significance **3**

2 Dataset *4*

2.1 Data Description **4**

3 Package Introduction *5*

4 Data Wrangling *5*

4.1 Dataset Information **5**

4.2 Data Processing **7**

5 Exploratory Data Analysis *8*

5.1 Summary Statistic **8**

5.2 Overall Distribution **9**

5.3 Daily Sales and Customers **10**

5.4 Summarized daily Sales and Customers **11**

5.5 Daily Sales Per Customer **13**

5.6 Sales by Year **14**

5.7 Sales by Month **15**

5.8 Sales by Day of Week **17**

5.9 Summarized Sales Analysis **17**

5.10 Sales Analysis by Store Type **21**

5.11 Competition Distance by Store Type **22**

5.12 Sales Analysis by Assortment Type **23**

5.13 Promotion Effectiveness on Sales **23**

5.14 Holidays vs. Sales **25**

6 Weekly Sales Forecasting *26*

6.1 Data Processing and Testing Strategy **26**

6.2 Baseline Model Selection **28**

6.3 Modeling Strategy **28**

6.4 Model Evaluation **31**

7 Conclusion *34*

7.1 Future Strategy **34**

7.2 Models **34**
7.3 Feature Work **35**
References ***36***
Appendix ***37***

1 Introduction

Dirk Rossmann GmbH is one of the largest drug store chains in Europe, operates over 3,000 drug stores in 7 Europe countries. This project firstly looks into how we can transform the company's historical sales, promotion, school/state holidays and competitors' data into business insights. Secondly, we will look into how we can turn these insights into actionable business strategy for the company. On the top of that, we will study how to use the historical sales data to forecast the weekly sales for each Rossmann store with a certain level of accuracy. Currently Rossmann store managers are predicting sales based on their unique circumstances, the accuracy of the results can be quite varied. Therefore, we will attempt to create a unified modelling method for managers in order to increase the overall forecasting accuracy.

1.1 Objective

The objectives of this project are to:

- Explore and analyze historical sales related data for Rossmann.
- Identify the key factors that influence store sales.
- Provide overall sales performance report to the management team with feasible strategies to increase future sales.
- Provide individual store performance report to each store manager.
- Develop time series model that predict the future sales for each store with certain level of accuracy.

This report is divided into the following sections:

- Section 2: Dataset
- Section 3: Package Introduction
- Section 4: Data Wrangling
- Section 5: Exploratory Data Analysis
- Section 6: Time Series Modeling
- Section 7: Model Evaluation
- Section 8: Conclusion
- The programming codes used for this report can be found in this Github Repository Sales-Analyzing-and-Weekly-Sales-Forecasting-for-Drug-Store-Rossmann.

1.2 Significance

By analyzing the historical sales related data, we will identify the main factors that influence store sales, identify key sales drivers among 1115 stores that are categorized into 4 different store

types and 3 different assortment types. Diagnose the key issues that are negatively impacting sales growth. Advising management team with feasible business strategy based on the analyze results. In respect of providing individual report to each store manager, we will create an online interactive dashboard for them to track the historical sales related data of their store or of any other stores. Data and graphs will be automated for consistent and automatic delivery on a daily, weekly or monthly basis. Future 8 weeks' sales forecast for each store will also be provided with 95% confidence interval. By bringing individual store data together in an organization dashboard, we can provide a more integrated and concise view for the sales of each store, ensuring the managers know where their store stand and where the store is based on goals set.

2 Dataset

2.1 Data Description

The datasets used in this study are sourced from the website kaggle, a subsidiary of Google LLC, is an online community of data scientists and machine learning practitioners that allows users to find and publish datasets, explore and build models in a web-based data-science environment. You can find the original dataset from Rossmann datasets. The two datasets we used in the study are:

- train.csv containing 1,017,209 data samples, is composed of a total of 9 variables collected from 1115 Rossmann stores over 942 days from 01/01/2013 to 07/31/2015, with 7 numerical variables and 2 categorical variables.
- store.csv containing 1115 data samples, is composed of a total of 10 variables collected from 1115 Rossmann stores, with 6 numerical variables and 3 categorical variables.

A description of each of the columns of the two datasets are provided in *Table 2.1* and *Table 2.2.*

Table 2.1 Description of train.csv

No	Variable Name	Variable Description	Data Summary
0	Store	A unique Id for each store	Integer, 1115 unique values
1	DayOfWeek	The day of the week	Integer, 7 unique vales
2	Date	The date of the day	Object, 942 unique values
3	Sales	The turnover of any given day	Integer, 21734 unique values
4	Customers	The number of customers on a given day	Integer, 4086 unique values
5	Open	Indicates whether the store was open or closed on that day	Integer, 0 = closed, 1 = open
6	Promo	Indicates whether a store is running a promo on that day	Integer, 0 = no promo, 1 = running promo
7	StateHoliday	Indicates a state holiday	Object, a = public holiday, b = Easter holiday, c = Christmas, 0 = None
8	SchoolHoliday	Indicates if it's a school holiday on a given day.	Integers, 0 = not a school holiday, 1 = school holiday

Table 2.2 Description of store.csv

No	Variable Name	Variable Description	Data Summary
0	Store	A unique Id for each store	Integer, 1115 unique values
1	StoreType	Differentiates between 4 different store models	Object, a, b, c, d
2	Assortment	Describes an assortment level	Object, a = basic, b = extra, c = extended
3	CompetitionDistance	Distance in meters to the nearest competitor store	Float, 654 unique values
4	CompetitionOpenSinceMonth	Gives the approximate month of the time the nearest competitor was opened	Float, 12 unique values
5	CompetitionOpenSinceYear	Gives the approximate year of the time the nearest competitor was opened	Float, 23 unique values
6	Promo2	Indicates whether a store is running consecutive promo	Integer, 0 = not participating, 1 = participating
7	Promo2SinceWeek	Describes the calendar week when the store started participating in Promo2	Float, 24 unique values
8	Promo2SinceYear	Describes the year the store started participating in Promo2	Float, 7 unique values
9	PromoInterval	Describes the consecutive month intervals Promo2 is started of any given year of that store	Object, 3 unique values

3 Package Introduction

In this study we used JupyterLab (2.1.5) to run all the codes for data analyzing and PyCharm (2019.3.5) to run the codes for building interactive dashboard. Numpy (1.18.5), Pandas (1.0.5), Matplotlib (3.2.2), Seaborn (0.11.0) were installed as basic package. Pmdarima (4.14.1) and Scikit-learn (0.23.1) were installed as the machine learning library. Statsmodels (0.11.1) was installed for time series analysis. Plotly (4.14.1) and Dash (1.19.0) were installed for interactive dashboard creation. Heroku cloud platform was used to host the application for interactive dashboard.

4 Data Wrangling

4.1 Dataset Information

Please find the general information of the two original datasets from *Table 4.1* and *Table 4.2,* sorted by the count of unique value in each variable in descending order. It contains columns of original variable index, variable name, counts of unique values, percentage of unique value, percentage of missing value and data type of each variable.

Table 4.1 General information of dataset train.csv

Index	Variable Name	Counts	Unique Value Percentage	Missing Value Percentage	Data Type
3	Sales	21734	2.1366	0	int64
4	Customers	4086	0.4017	0	int64
0	Store	1115	0.1096	0	int64
2	Date	942	0.0926	0	object
1	DayOfWeek	7	0.0007	0	int64
7	StateHoliday	5	0.0005	0	object
5	Open	2	0.0002	0	int64
6	Promo	2	0.0002	0	int64
8	SchoolHoliday	2	0.0002	0	int64

Table 4.2 General information of dataset store.csv

Index	Variable Name	Counts	Unique Value Percentage	Missing Value Percentage	Data Type
0	Store	1115	100	0.0000	Int64
3	CompetitionDistance	654	58.6547	0.2691	float64
7	Promo2SinceWeek	24	2.1525	48.7892	float64
5	CompetitionOpenSinceYear	23	2.0628	31.7489	float64
4	CompetitionOpenSinceMonth	12	1.0762	31.7489	float64
8	Promo2SinceYear	7	0.6278	48.7892	float64
1	StoreType	4	0.3587	0	object
2	Assortment	3	0.2691	0	object
9	PromoInterval	3	0.2691	48.7892	object
6	Promo2	2	0.1794	0	int64

4.2 Data Processing

After examining all the variables in two datasets, we processed data by following 5 main steps. Please find the details of each step from table 4.3.

Tabel 4.3 Data processing step

Steps	Action	Variable Names	Detail explanation	Dataset name
Step 1	Data Type Correction	Date	Convert object to datetime	train.csv
		StateHoliday	Convert object to int	tain.csv
Step 2	New Variable Creation	Year	Add new variable 'Year' by extracting year value from variable 'Date'	train.csv
		Month	Add new variable 'Month by extracting month value from variable 'Date'	train.csv
Step 3	Aggregation	All variables	Sum up all the values of all the variables for each store in train.csv	store_sum.csv (name after aggregation)
Step 4	New Variable Creation	AverageDailySales	AverageDailySales = Sum of 'Sales' ÷ Sum of 'Open'	store_sum.csv
		AverageDailyCustomer	AverageDailyCustomer = Sum of 'Customers' ÷ Sum of 'Open'	store_sum.csv
		SalesPerCustomer	SalesPerCustomer = Sum of 'Sales' ÷ Sum of 'Customers'	store_sum.csv
Step 5	Datasets Combination	-	Combine store_sum.csv with store.csv	combine_data.csv (name after combination)

You can find more details about the cleaning process from this jupyter notebook DataWrangling and EDA01_All Stores and DataWrangling and EDA02_ByStoreType.

5 Exploratory Data Analysis

5.1 Summary Statistic

After adding new variables 'Year and 'Month', statistic including mean, standard deviation, minimum values, maximum values and percentile for each variable in train.csv were summarized in *Table 5.1*

Table 5.1 Summary statistics for train.csv

variable name	count	mean	std	min	25%	50%	75%	max
Store	1017209	558.429727	321.908651	1	280	558	838	1115
DayOfWeek	1017209	3.998341	1.997391	1	2	4	6	7
Sales	1017209	5773.818972	3849.926175	0	3727	5744	7856	41551
Customers	1017209	633.145946	464.411734	0	405	609	837	7388
Open	1017209	0.830107	0.375539	0	1	1	1	1
Promo	1017209	0.381515	0.485759	0	0	0	1	1
StateHoliday	1017209	0.030525	0.172026	0	0	0	0	1
SchoolHoliday	1017209	0.178647	0.383056	0	0	0	0	1
Month	1017209	5.846762	3.326097	1	3	6	8	12
Year	1017209	2013.832292	0.777396	2013	2013	2014	2014	2015

5.2 Overall Distribution

The overall distribution of numerical variables of train.csv was visualized in *Figure 5.1*.

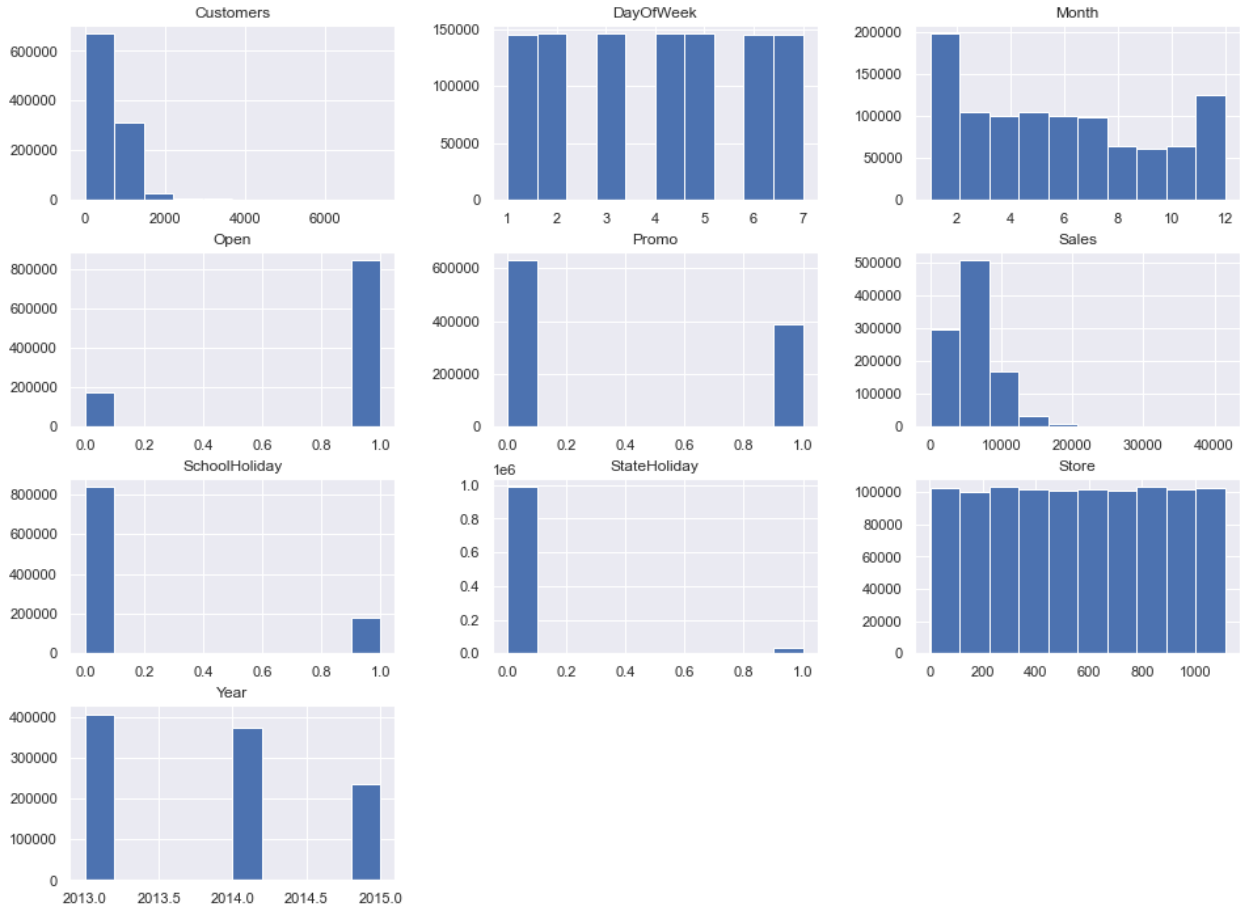


Figure 5.1 Distributions of numerical variables of train.csv

5.3 Daily Sales and Customers

By observing the distribution plots of daily Sales and Customers of 1115 stores over 942 days, we can see both of these two variables' distribution occurs more on the left side with a long tail on the right side, with an average value of 5774 (Sales), and 633 (Customers).

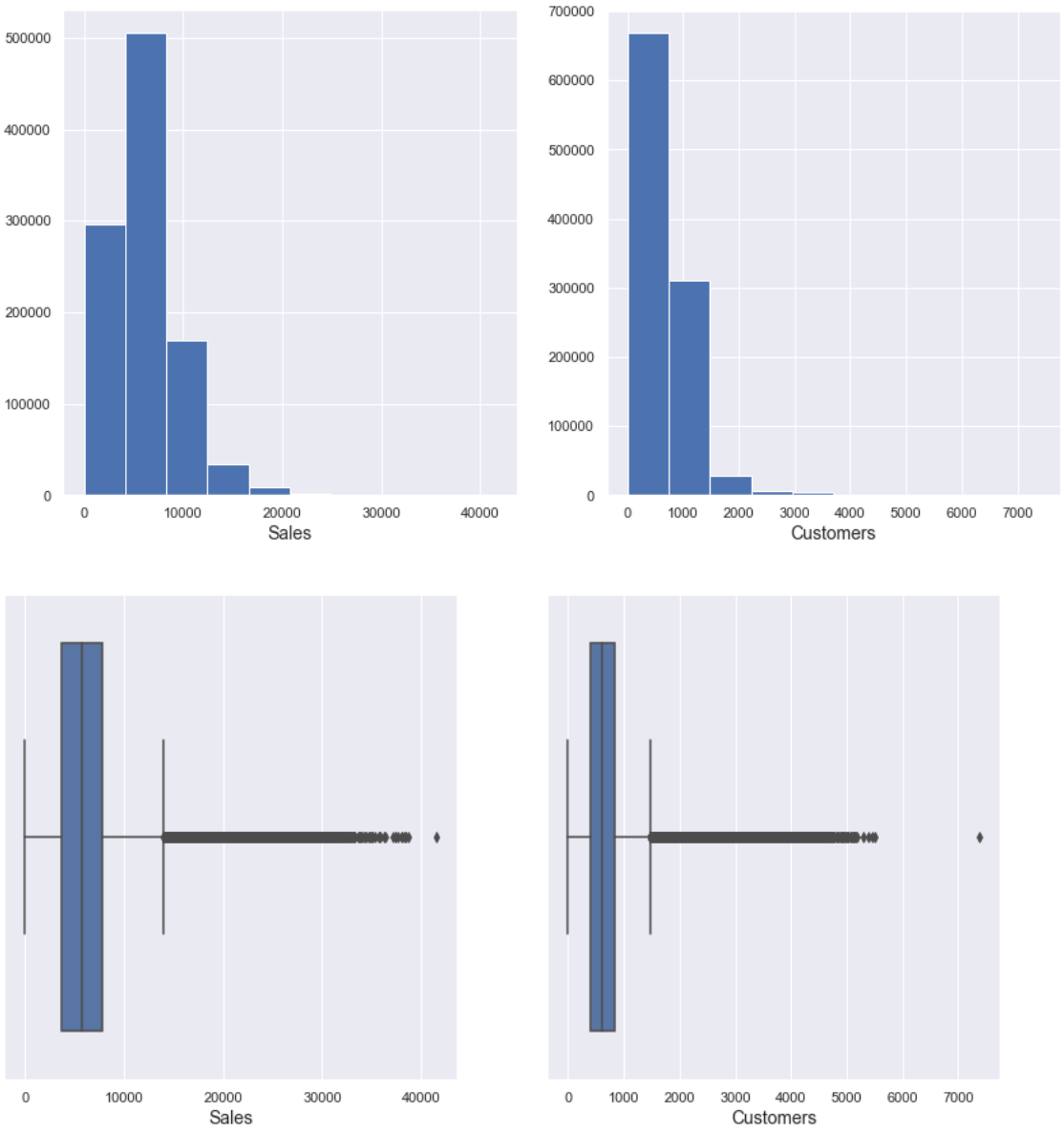


Figure 5.2 Distributions of daily Sales and daily Customers

5.4 Summarized daily Sales and Customers

To understand how the amounts of sales and customers fluctuate over time, we visualized both of them with different graphs of varying time spans in *Figure 5.3* (from ①~ ④).

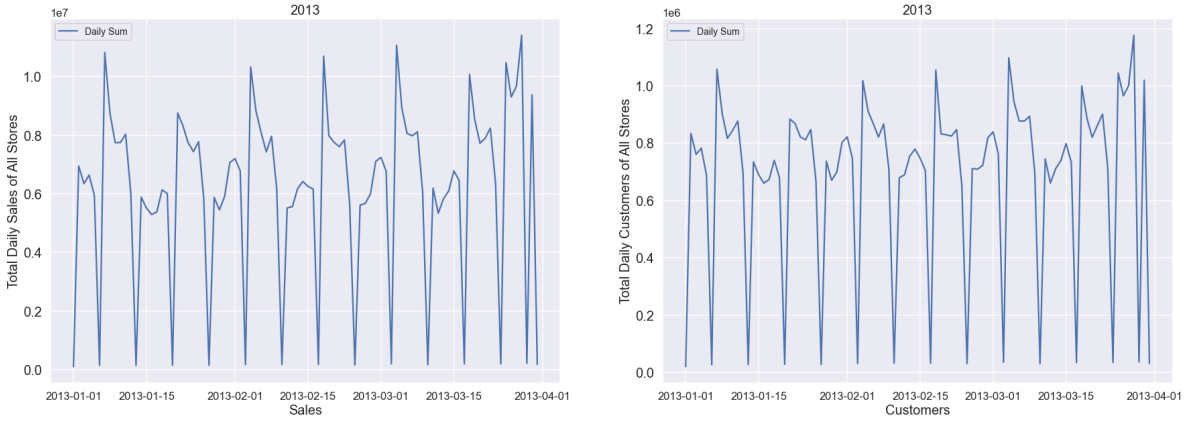
In ① of *Figure 5.3*, we visualized daily total sales and daily total customers of 1115 stores from 2013-01-01 to 2015-07-31.

①Daily sum of all store from 2013 ~ 2015:

Figure 5.3 Daily and weekly sum of sales & customers

② Daily sum of all store in year 2013:

③ Daily sum of all store from Jan to Mar in 2013:



④ Weekly sum of all store from 2013 ~ 2015:

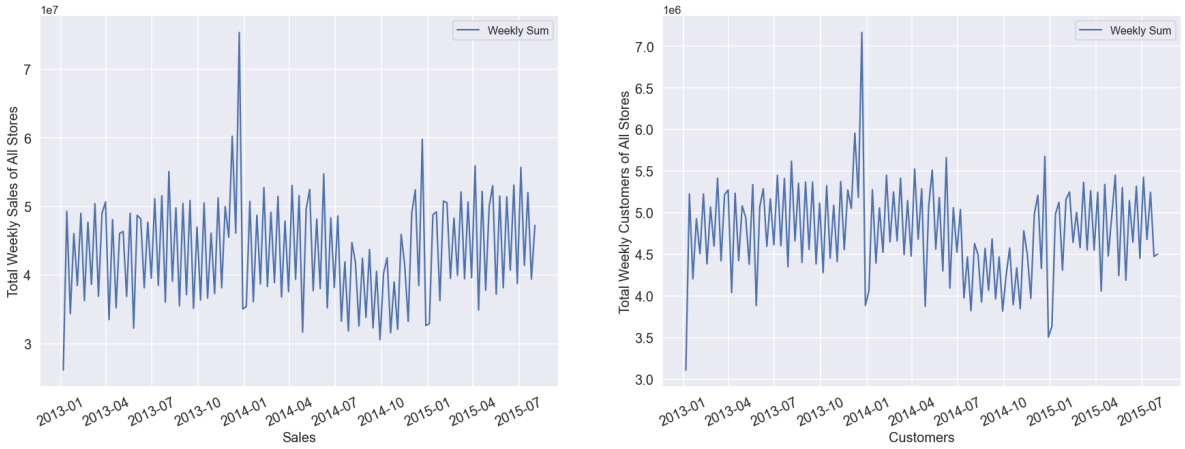


Figure 5.3 Daily and weekly sum of sales & customers

In ② we visualized total daily sales and customers in year 2013. In ③ we focused only on the period of Jan ~ Mar in 2013. Lastly In ④ we visualized weekly total sales and customers from 2013-01-01 to 2015-07-31. We can see the amounts of sales and costumers fluctuate together in all these graphs, so we will focus on sales analysis in this study. Also, in ④ we observed a big decline in both sales and customers around July 2014 ~ Jan 2015. As 180 stores were closed for renovation over that period (*Figure 5.4)*

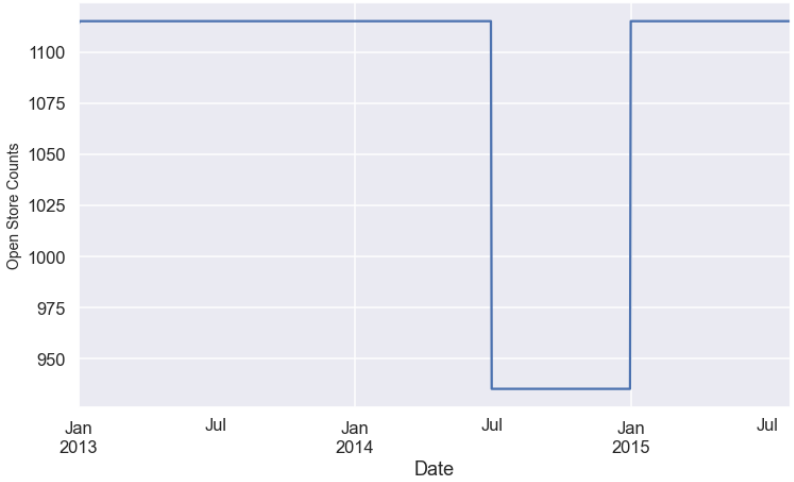


Figure 5.4 Number of stores that were closed for renovation: 180

5.5 Daily Sales Per Customer

In *Figure 5.5* we visualized daily sales-per-customer with a rolling window of 7 days of all stores from 2013-01-01 to 2015-07-31. Daily sales per customer is calculated by: (sum of daily Sales of 1115 stores) ÷ (sum of daily Customers of 1115 stores), we can see this value roughly varies from 7 to 10 over the 942 days. This value reaches its peak in December every year.

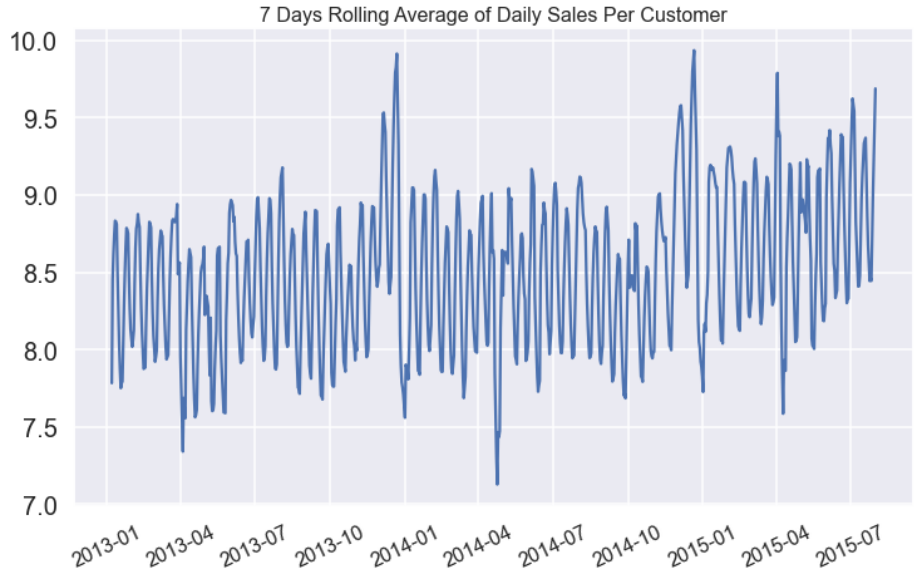


Figure 5.5 Seven days average of daily Sales per customer

5.6 Sales by Year

In *Table 5.2* and *Figure 5.6* we visualized all store total sales by year, average daily sales per store by year and sales per customer by year.

Table 5.2 Sales percentage change by year

Year	Total Sales	Percentage Change	Daily Sales Per Store	Percentage Change	Sales Per Customer	Percentage Change
2013	2.303E+09	NaN	5658.534	NaN	8.995	NaN
2014	2.181E+09	-5.3%	5833.291	3.1%	9.068	0.8%
2015	1.389E+09	-36.3%	5878.245	0.7%	9.417	3.8%

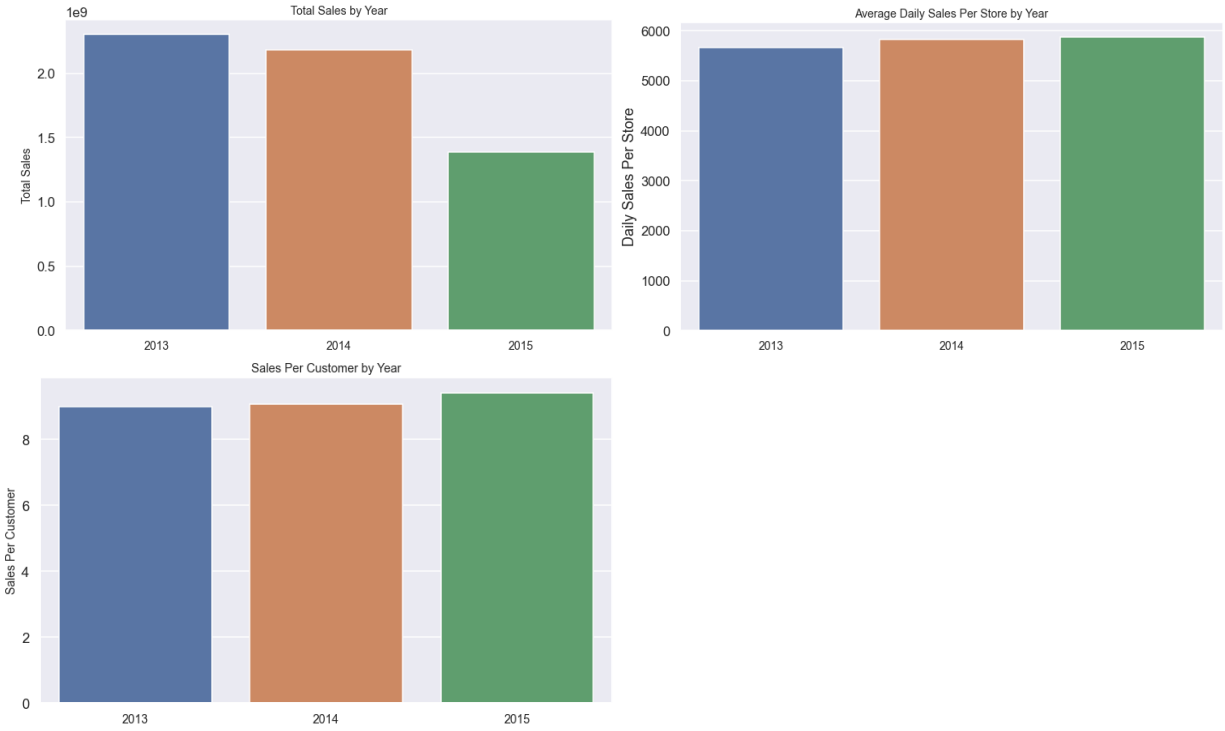


Figure 5.6 Sales by year

Comparing to 2013, total sales decreased by 5.3 % in 2014. (Please be notified that the total sales of 2015 are not a full year data). On the other hand, daily sales per store keeps increasing from 2013 with a percentage of 3.1% and 0.7%. Sales per customer also showed an upward trend with a percentage of 0.8% and 3.8%.

5.7 Sales by Month

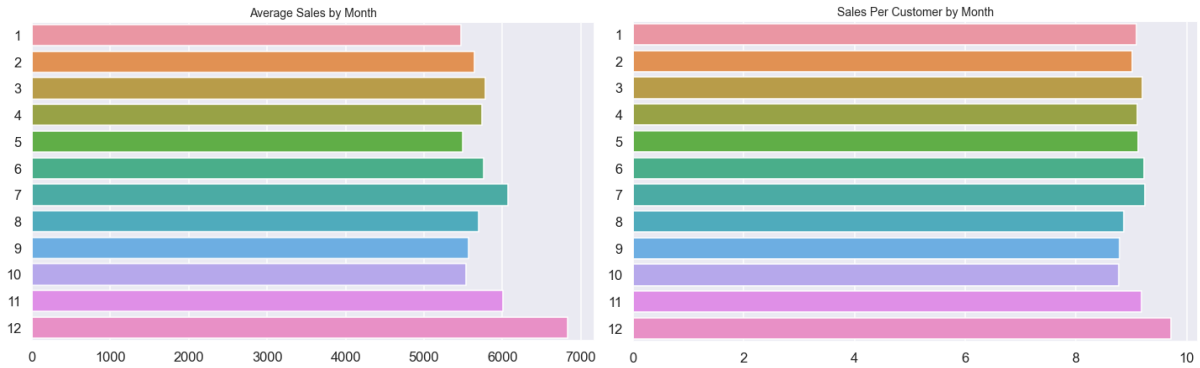


Figure 5.7 Average daily sales per store and sales per customer by month

By visualizing sales by month in *Figure 5.7*, we can see December and July have the highest and second highest average daily sales per store and sales per customers comparing to other months. In *Figure 5.8* we visualized average daily sales per store in each month from 2013 to 2015 on the left side, and on the right side we plotted the sum of daily promotions in each month from 2013 to 2015.

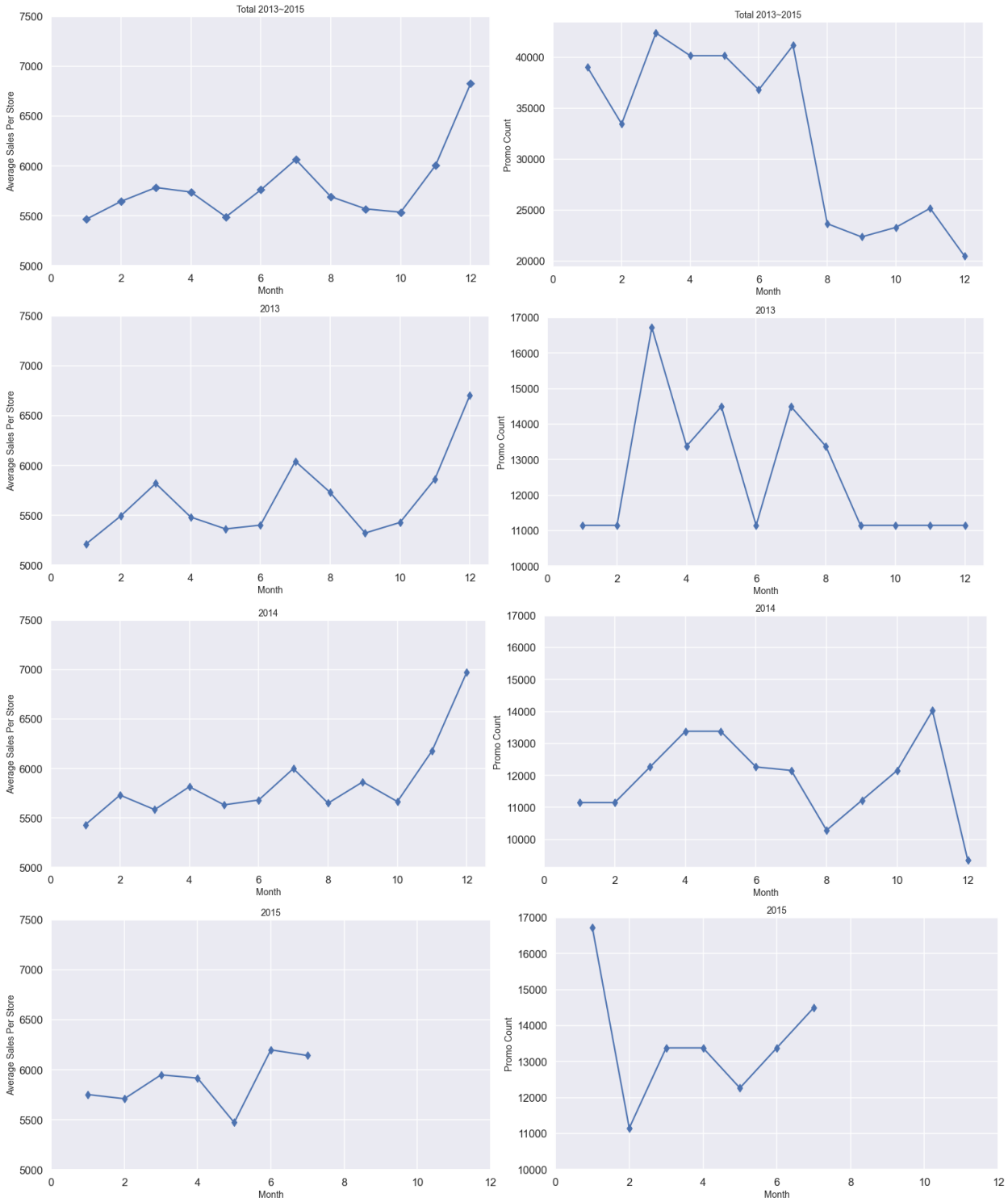


Figure 5.8 Daily Promotion count by month

(please be notified that data in 2015 ended on July 31st)

We can see comparing to 2013, company focused on running more daily promotions in autumn season in 2014, and this brought positive impact in average daily sales per store as we saw an increase from September to December comparing to 2013. In 2015, we increased daily

promotions in January and again, we see significant increase in terms of average daily sales per store. In February 2015, we decreased the amounts of daily promotions among stores, and the average daily sales stayed flat comparing to 2014.

5.8 Sales by Day of Week

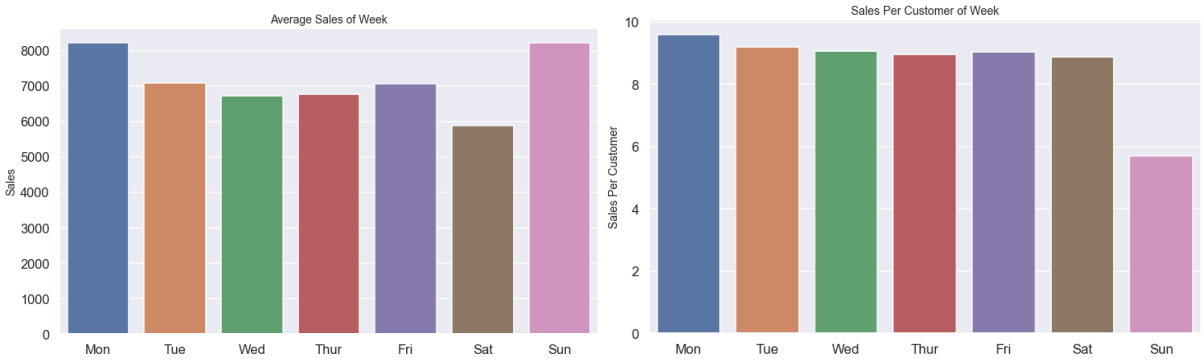


Figure 5.9 Sales by day of week

Despite being the second highest average sales in a week, Sunday has the lowest sales per customer comparing to other days of week.

5.9 Summarized Sales Analysis

In this part we will analyze the summarized sales of each store over the 942 days. You can find the top 10 stores in terms of total sales and their sales share among all stores from *Table 5.3*. We can see the share of the top 10 stores' total sales is about 2.6% among all stores. From *Figure 5.10* we can see the ratio of top 400 stores and the rest 715 stores' sales shares are roughly about 1:1. This means half of the total sales 5,873,180,623 (over the 942 days) came from the top 400 stores and the rest 751 stores generated the other half.

Table 5.3 Total sales top 10 stores

Ranking	Store id	Sales	Percentage %
1	262	19,516,842	0.332304
2	817	17,057,867	0.290437
3	562	16,927,322	0.288214
4	1114	16,202,585	0.275874
5	251	14,896,870	0.253642
6	513	14,252,406	0.242669
7	788	14,082,141	0.23977
8	733	14,067,158	0.239515
9	383	13,489,879	0.229686
10	756	12,911,782	0.219843

Figure 5.10 Sales share breakdown of top 10 to top 400 stores and the rest stores

As we mentioned previously, 180 stores were closed for certain period of months due to renovation, so total sales probably will not be the best metrics to compare store performance in this case. Therefore, we also use another two metrics: 1) average daily sales, which were calculated by using sum of total sales divided by sum of total store open days, to identify stores with high performance; 2) sales per customer, which were calculated by using sum of total sales divided by sum of total customers over 942 days. From *Figure 5.11* you can find the top 10 stores that have the highest daily sales and their daily sales distribution. In *Figure 5.12* we visualized the top 10 stores that have highest sales per customer and their daily sales distribution.

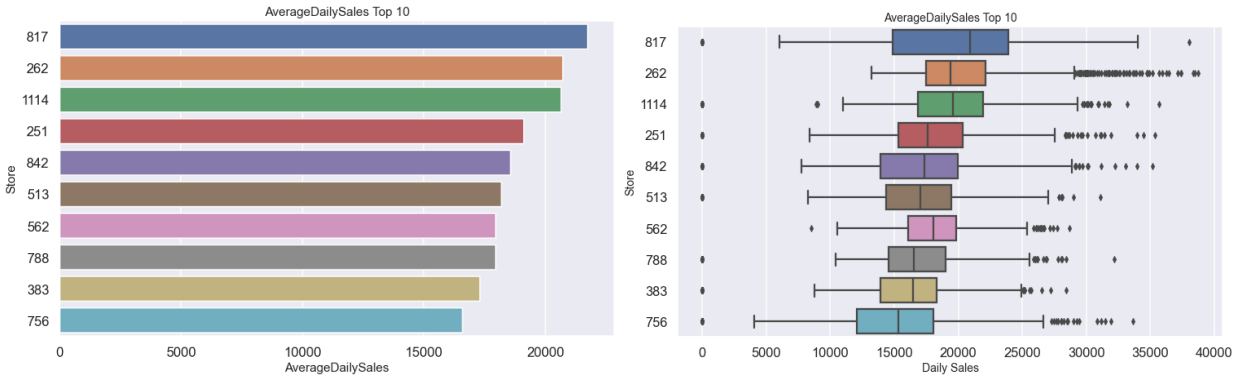


Figure 5.11 Top 10 average daily sales & their daily sales distribution

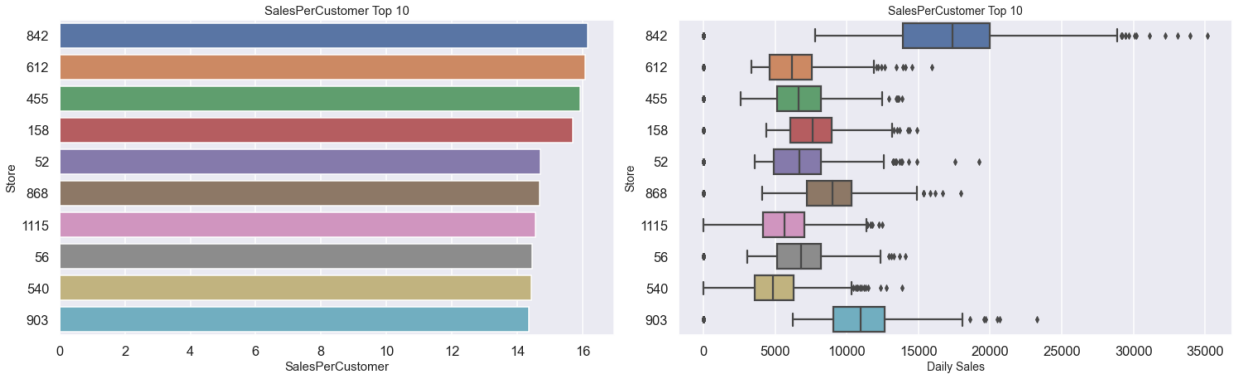


Figure 5.12 Top 10 sales per customers & their daily sales distribution

After checking the sales per customer ranking for these top 10 stores in *Table 5.4,* we noticed that in spite of their high average daily sales, their sales per customer are overall very low among all stores, except store 842 generated the highest value in terms of this metrics.

Table 5.4 Sales per customer ranking

Store id	Sales Per Customer Ranking
817	1049
262	1099
1114	1081
251	919
842	1
513	717
562	1103
788	367
383	909
756	1038

Figure 5.13 Comparison between store 842, store 817, store 262

In *Figure 5.13*, after comparing store 842 with store 817 and store 262 which have the highest and second highest average daily sales, we noticed that generally, store 842's daily customers were less than half of store 817 and store 262's daily customer, but as it has high sales per customer, the weekly sales generated from store 842 were very close to store 817 and store 262. However due to store renovation, store 842 stayed closed for 6 months. If we could make strategies to increase daily customers for stores like store 842 that has a high value of sales per customer, or make sure we don't shut them down over the same period of time for renovation, the sales increase brought in by these stores should leave a big positive impact in the company's total sales. More details of the analysis could be found in this jupyter notebook DataWrangling and EDA01_AllStores.ipynb and rossmann_dashboard.

5.10 Sales Analysis by Store Type

In this section we analyze sales break down by store type. All the Rossmann stores are categorized into 4 store types: a, b, c, d. The sales break down information of 4 different store types are provided in *Table 5.5, Figure 5.14, Figure 5.15, and Figure 5.16*.

Table 5.5 Sales breakdown by store type

Store Type	Count	Total Sales	Total Sales Per Store	Average Daily Sales	Sales Per Customer
a	602	3.16E+09	5.26E+06	6912.54	8.96
d	348	1.77E+09	5.07E+06	6823.74	11.43
c	148	7.83E+08	5.29E+06	6917.20	8.74
b	17	1.59E+08	9.37E+06	10110.78	5.17

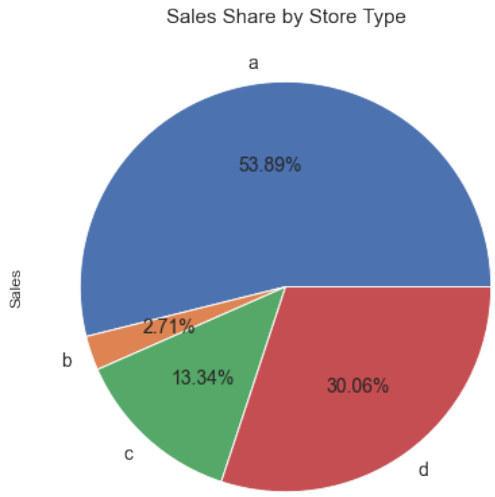


Figure 5.14 Sales share by store type

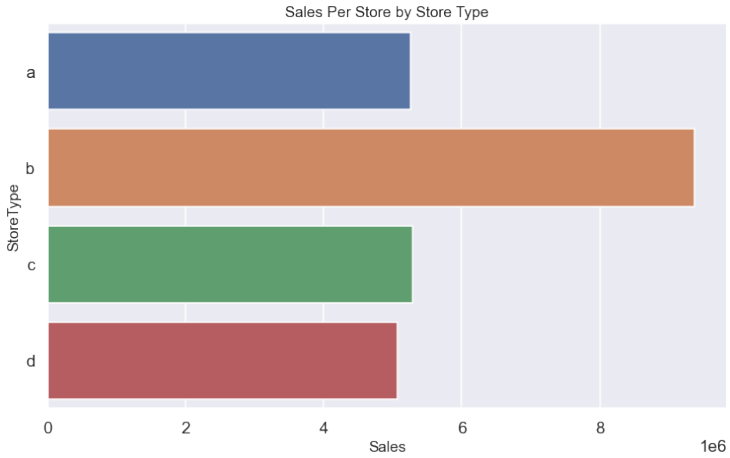


Figure 5.15 Total sales per store by store type

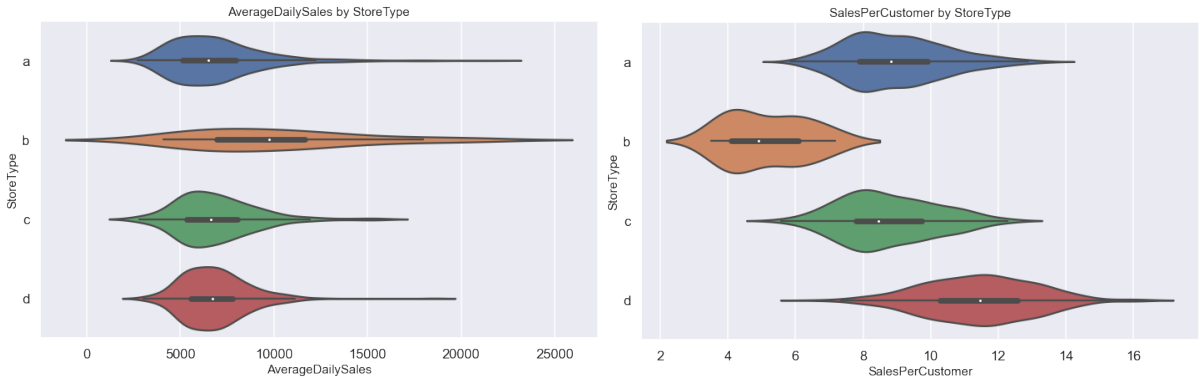


Figure 5.16 Average daily sales & sales per customer distribution by store type

From the analysis above we can see despite the fact that B type stores generate the highest sales per store but its sales per customer is the lowest among the 4 types, and there's an obvious gap between B type and the third lowest C type in terms of sales per customer. To understand the cause of the difference better, we will check the competitors' distance for the 4 types of stores.

5.11 Competition Distance by Store Type

Table 5.6 Competition distance by store type

StoreType	a	b	c	d
count	601	17	148	346
mean	5123	1061	3523	6913
25%	460	590	490	1880
50%	1790	900	1660	5040
75%	5020	1270	3985	9703

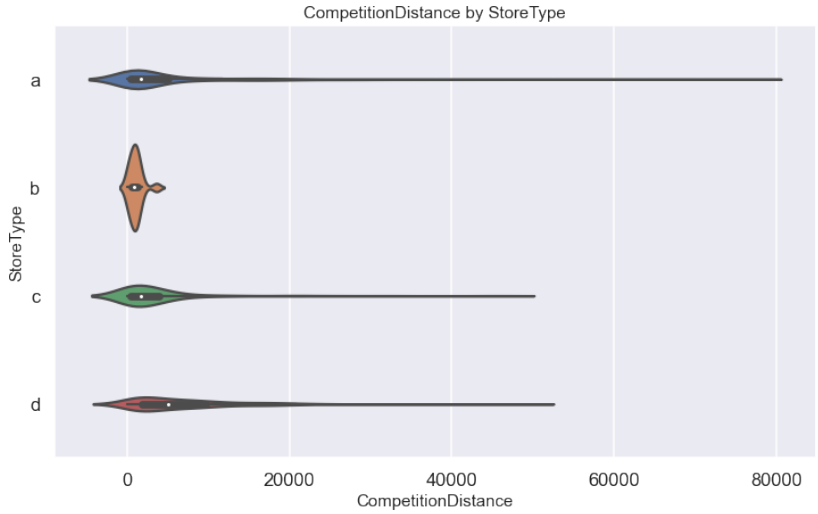


Figure 5.17 Competition distance distribution by store type

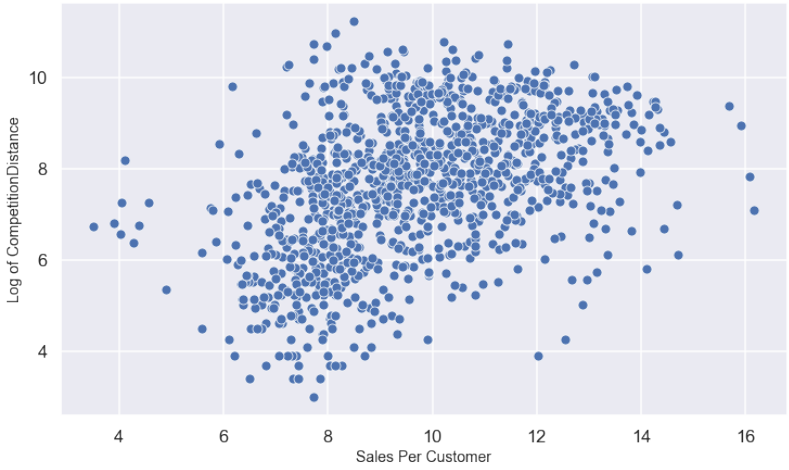


Figure 5.18 Log of competition distance vs. sales per customer

By checking competition distance for 4 store types in *Table 5.6* and *Figure 5.17,* we realized that B type of stores are the one that have competitors more near by comparing to other 3 types. In *Figure 5.18,* we were able to observe a liner relationship between the log of competition distance and sales per customer, as the log of competition increase the sales per customer increase correspondingly. This explained why B type stores has the lowest customer sales, also should be considered as a clear sign that we are losing our customers to the competitors as they tend to spend less with us when the competitors are close by. More details of this analysis could be found in this jupyter notebook DataWrangling and EDA02_ByStoreType.ipynb and rossmann_dashboard.

5.12 Sales Analysis by Assortment Type

All the Rossmann stores are categorized to a, b, c, 3 types of assortment, with 'a' stands for basic, 'b' stands for extra, and 'c' stands for extended. After analyzing sales break down by assortment, we noticed that assortment have very similar problem as store type: b assortment has the highest sales per store, however its sales per customer value is the lowest comparing to a, c assortment type, as b assortment has the shortest competitors' distance. Therefore we will skip the repetitive analysis here in this report, for more details you can find them in this jupyter notebook EDA03_ByAssortment.ipynb.

5.13 Promotion Effectiveness on Sales

Over the 942 days, Rossmann stores were running 2 types of promotions: daily promotion and consecutive promotion. After aggregating the number of daily promotions for 1115 stores, we noticed there were 286 days and 360 days type of daily promotions. 180 stores ran a total 286 daily promotions, while 935 stores ran 360 days of daily promotion over the 942 days. From *Table 5.7* we can see stores that ran 360 days of daily promotions generated higher average total

sales and average daily sales than stores that ran 286 days of daily promotions. And they both resulted a very close value of sales per customer with a small difference of 1.43.

In *Table 5.8* we summarized the comparison between stores that joined long term promotions (consecutive promotions) and stores that didn't. Stores that didn't run long term promotions generated higher average total sales and average daily sales than stores that did run long term promotions. Again, despite joining long term promotion or not, the value of sales per customers were very close with a difference of 0.99. The detailed code and analysis of this part are provided in this jupyter notebook EDA04_ByPromotions.ipynb.

Table 5.7 Daily promotion & sales performance

Count of Daily Promotion	Store Count	Average Total Sales	Average Daily Sales	Sales Per Customer
286 Days	180	4.12E+06	6,634	10.84
360 Days	935	5.49E+06	6,992	9.41

Table 5.8 Consecutive promotions & sales performance

Status of Participant	Store Count	Average Total Sales	Average Daily Sales	Sales Per Customer
Not Participating	544	5.72E+06	7,337	9.14
Participating	571	4.84E+06	6,550	10.13

5.14 Holidays vs. Sales

Table 5.9 Holiday and sales

Holiday	Average Daily Sales	Sales Per Customer
Yes	7614	8.59
No	7208	8.53

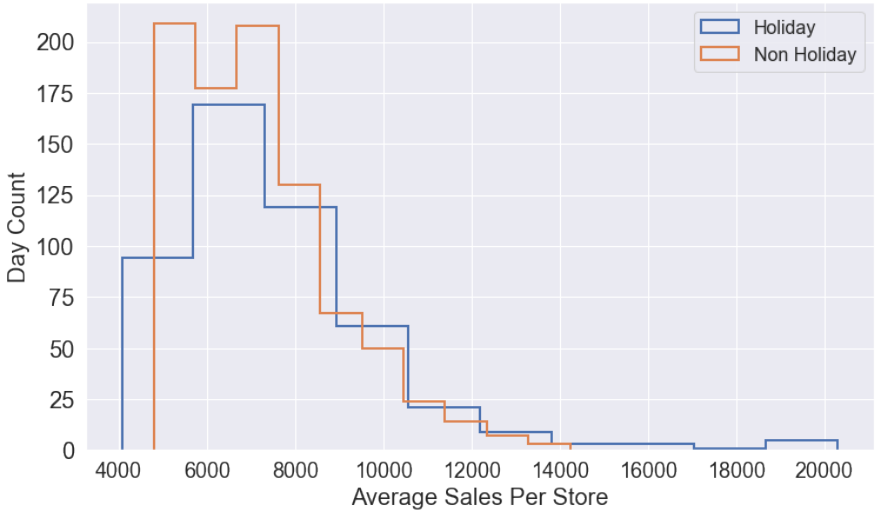


Figure 5.19 Holidays daily sales vs. non holiday daily sales

In Table 5.9 and Figure 5.19 we visualized how holidays (state or school holidays) are affecting the amounts of sales. We can see when it's a holiday, average sales per store is 5% higher than when it's not a holiday, sales per customer is also slightly higher when it's a holiday. Furthermore, there were more daily sales higher than 14,000 when it's a holiday.

6 Weekly Sales Forecasting

6.1 Data Processing and Testing Strategy

In this section, we will work on building a unified time series model for 1115 stores based on their historical weekly sales data. As our mission is to forecast the future 8 weekly sales from week 2015-08-02 to week 2015-09-20, therefore we will split our dataset following **Strategy 1** when there's no zero weekly sales over the 135 weeks, 942 days:

Strategy 1:

Train (!27 weeks)	Test (8 weeks)	Forecast (8 weeks)
1/6/2013 ~ 5/31/2015	6/7/2015 ~ 7/26/2015	8/2/2015 ~ 9/20/2015

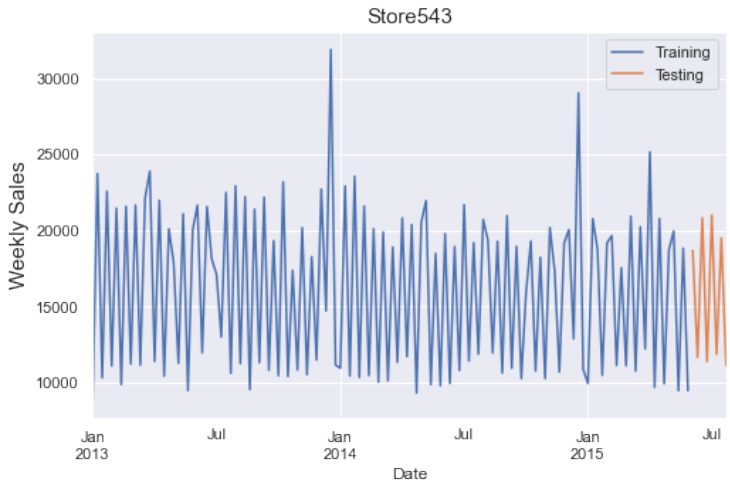


Figure 6.1 Example of train/test split following strategy 1

However, as we explained previously, there are 180 stores stay closed for multiple months due to renovation, so for those stores, there was a time period that contains only zero values for weekly sales. Let's use Store 186 as an example, you can see in *Figure 6.2*, from 07/13/2014 to 12/28/2014 no weekly sales were generated over this time.

Figure 6.2 Weekly sales for Store 186

In the case of Store 186, we choose to use the most recent datapoints that starts after the store reopened as the beginning of our dataset, and we will then do the train/test split following strategy 2, as this strategy enables us to gain higher predicting and forecasting accuracy.

Strategy 2:

Train (22 weeks)	Test (8 weeks)	Forecast (8 weeks)
1/4/2015 ~ 5/31/2015	6/7/2015 ~ 7/26/2015	8/2/2015 ~ 9/20/2015

Figure 6.3 Example of train/test split following strategy 2

However, **Strategy 2** will not work if the data points we select after the store reopened are not enough for train/test splitting (If the length of the datapoints is shorter than 16). In this case we will use **Strategy 3**: use the whole dataset but drop the time periods that contains zero weekly sales, and train/test split the data after dropping zero values. Ensure after splitting, the test set contains 8 weekly sales data points.

6.2 Baseline Model Selection

There are multiple persistence methods we can use for time series, such as mean method, naïve method:

- Mean Method: $\hat{y}_{T+h|T} = \bar{y} = (y_1 + \cdots + y_T)/T$.
  Forecasts of all future values are equal to the average of the historical data.
- Naïve method: $\hat{y}_{T+h|T} = y_T$.
  Forecasts of all future values are equal to the last observation.

As we mentioned primarily, all the store managers are forecasting their store future sales based on their own unique methods and based on our investigation, we found out that most of them were using average weekly sales to forecast the future values. Thus, in this study we will select Mean Method as our baseline model.

6.3 Modeling Strategy

There are many different classical time series forecasting methods including:

- Autoregression (AR)
- Moving Average (MA)
- Autoregressive Moving Average (ARMA)
- Autoregressive Integrated Moving Average (ARIMA)
- Seasonal Autoregressive Integrated Moving-Average (SARIMA)
- Seasonal Autoregressive Integrated Moving-Average with Exogenous Regressors (SARIMAX)

You can find more detailed explanations for each method from here. Usually with these methods, we need to provide the p, d, and q values.

- p: stands for auto-regressive (AR), represents the number of autoregressive terms.
- d: stands for integrated (I), represents the degree of differencing to allow for the time series to become stationary
- q: stands for moving average (MA), represents the number of lagged forecast errors in the prediction equation.

For AR method we need to provide the optimal value of p, for MA model we need to prove the optimal value of q. For ARMA method we need to find the optimal value for both p and q. For ARIMA model, besides p and q, we also need to find optimal value of d. Furthermore, if our data set has seasonality, then we need to use SARIMA or SARIMAX, that will also need us to provide optimal values for (P, D, Q, m)— represents the (p, d, q) for the seasonal part of the time series, and m represents the number of time steps for a single seasonal period. This could be written as follow:

$$\text{ARIMA} \quad \underbrace{(p,d,q)}_{\text{Non-seasonal part of the model}} \quad \underbrace{(P,D,Q)_m}_{\text{Seasonal part of the model}}$$

It is very similar to hyperparameter tuning in order to find the optimal values for p, d, q, or p, d, q, P, D, Q, m with different combinations. The final value will be determined for each parameter when that combination gives us the lowest AIC. If you need to know more about AIC, please visit AIC.

Normally with time series modeling, before determining the value for p, q, d, the steps we will follow could be explained as below:

- Step 1: Decompose the time series to trend, seasonal and resid to understand the pattern.
- Step 2: Use Augmented Dickey-Fuller Test to check stationarity for the time series. (Ideally, we need stationary time series for modeling, if not, we can transform the time series to stationary data using different methods. You can find more information here.)
- Step 3: Check ACF and PACF plot to define the range of values for p, q.

Below in *Figure 6.4* we visualized step 1 and step 3 for Store 186. ACF and PACF plots allow you to determine the AR and MA components of an ARIMA model. Both the Seasonal and the non-Seasonal AR and MA components can be determined from the ACF and PACF plots. If you would like to learn more about how to find p, q value by examining ACF and PACF plot, you may find this article helpful.

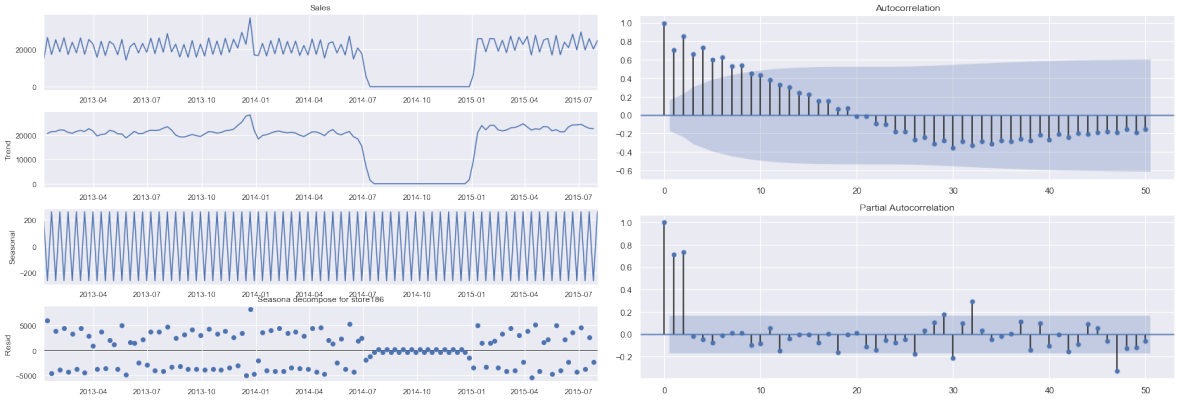


Figure 6.4 seasonal decompose plot (left) and ACF_PACF plot (right) of Store 186

In our case, we need to forecast the future 8 weeks' sales for 1115 stores, and each store is a time series with unique pattern. It would be an immense amount of work if we tried to manually puzzle out the optimal value of p, d, q or P, D, Q for each store time series. To solve this

problem, we will use a model called Auto ARIMA, the model itself will generate the optimal p, d, q and P, D, Q, m values which would result the lowest AIC and gives us higher accuracy of forecasting. After finalizing our model, from here the steps we took could be summarized as below:

- Step 1: determine values of main parameters for Auto Arima model as below:

```
arima_model =  auto_arima(train,start_p=0, d=0, start_q=0,
                          max_p=5, max_d=5, max_q=5, start_P=0,
                          D=1, start_Q=0, max_P=5, max_D=5,
                          max_Q=5, m=2, seasonal=True,
                          error_action='warn',trace = False,
                          supress_warnings=True,stepwise = False,
                          random=True,
                          random_state=20, n_fits=50)
```

- Step 2: run Auto Arima model we set up in Step 1 on each training set, get prediction values, calculate mean absolute error (MAE) for each store.
- Step 3: calculate mean absolute error using mean method for 1115 stores.
- Step 4: compare MAE we obtained from step 2 and step3.
- Step 5: calculate WAPE [1]of each week for Auto Arima model and mean method.
- Step 6: run Auto Arima model again to get forecast values for future 8 weeks' sales.
- Step 7: implement Step 1 and Step 5 on the interactive dashboard, enabling the store manager to confirm the forecast together with 95% confidence interval and with past 8 weeks sales and prediction values (*Figure 6.5)*.

The details of data process and modeling strategy are provided in this notebook [06_Modeling.ipynb]

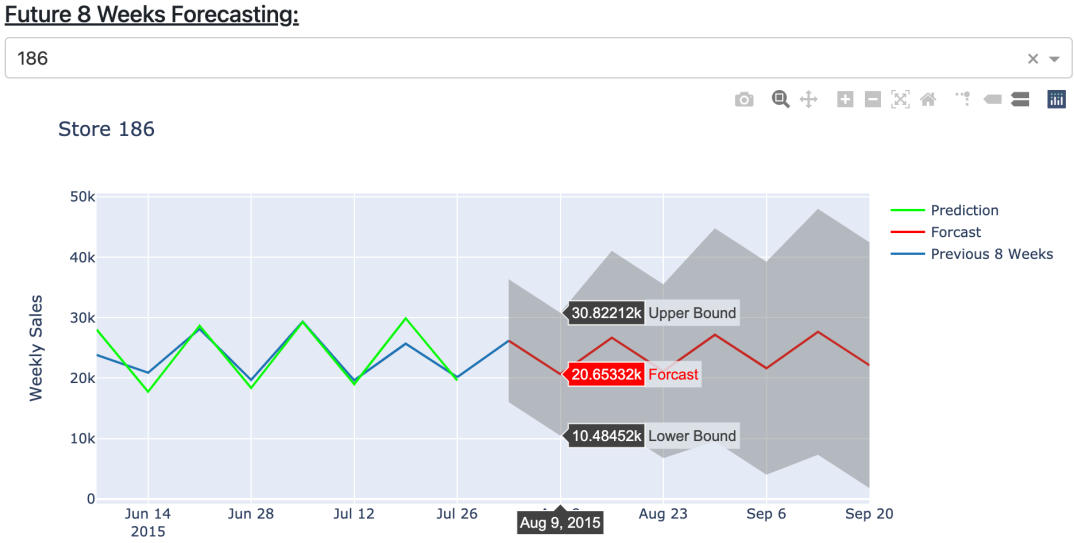


Figure 6.5 Forecast result for Store 186 on the interactive dashboard

[1] WAPE: Weighted Average Percentage Error. It weights the error by adding the total sales: $WAPE = \frac{\sum_{t=1}^{n}|A_t - F_t|}{\sum_{t=1}^{n}|A_t|}$

6.4 Model Evaluation

In this section, we will display the model performance result we obtained from the baseline model (mean method) and Auto Arima model. In *Table 6.1* and *Figure 6.6* we visualized the MAE of 1115 stores between two models.

Table 6.1 Comparison of average MAE between Auto Arima and mean method

	MAE_AutoArima	MAE_MeanMethod	Performance Improvement
average	4084	7143	1.75

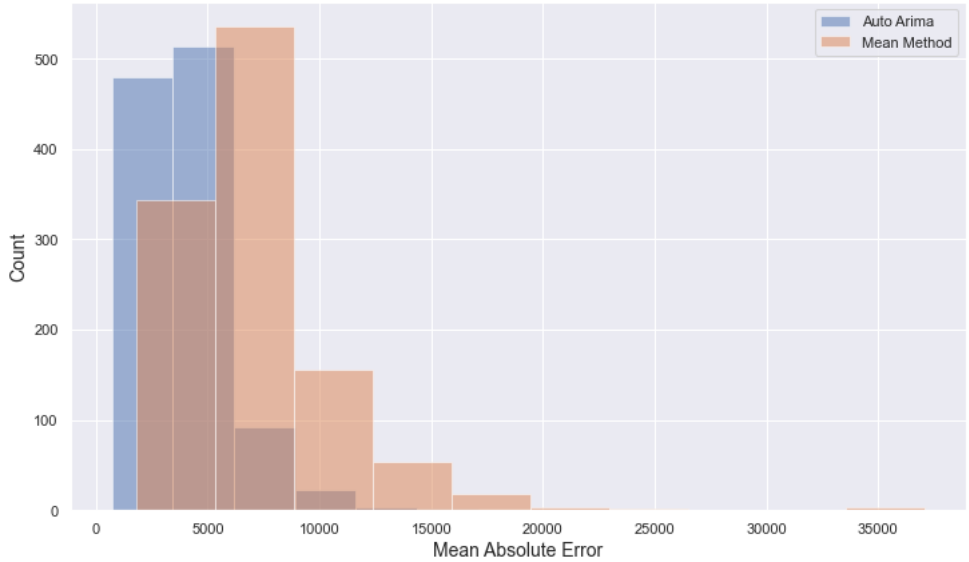


Figure 6.6 MAE distribution of Auto Arima and mean method

We can see by comparing the average value of MAE we obtained from 1115 stores, Auto Arima outperformed mean method by 1.75 times. This is the overall result. Next, we will have a look at the best, middle and worst 20 values of MAE that obtained by using Auto Arima model and compare them with the MAE values we got when using mean method. By visualizing the comparison in *Figure 6.7* we can see Auto Arima generated lower MAE overall, except in the 'Auto Arima Worst 20 vs Mean Method' plot we observed only 4 stores got lower MAE results by using mean method.

Figure 6.7 MAE comparison between Auto Arima and mean method

After comparing MAE, we will compare WAPE we obtained from 8 testing weeks:

Table 6.2 WAPE comparison of each testing week

Date	WAPE_AutoArima	WAPE_MeanMethod
2015-06-07	0.0030	0.1568
2015-06-14	0.0249	0.0622
2015-06-21	0.0430	0.1850
2015-06-28	0.1262	0.1174
2015-07-05	0.0885	0.2233
2015-07-12	0.0846	0.0460
2015-07-19	0.0233	0.1691
2015-07-26	0.1542	0.0934

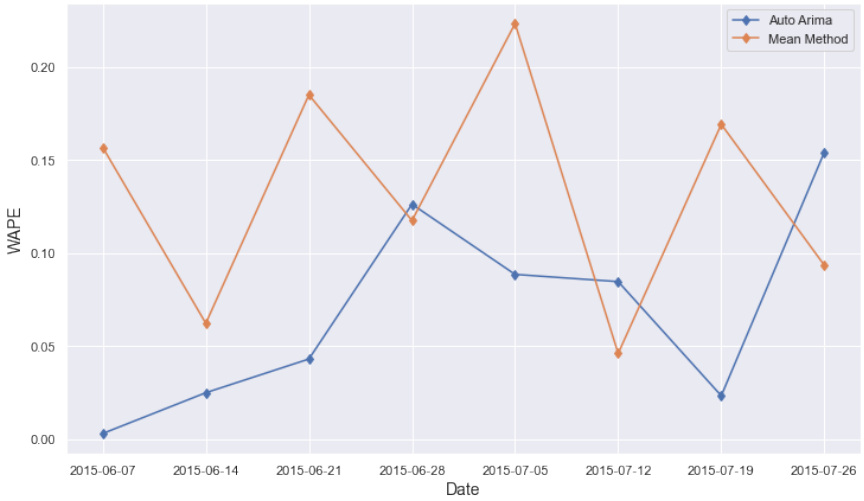


Figure 6.8 WAPE comparison between Auto Arima and mean method for each testing week

From *Table 6.2 and Figure 6.8* we can see that within 8 WAPE results, there were 5 of them are showing that Auto Arima model performed better than mean method, except week 6/28, week 7/12 and week 7/26. For more details of this part please visit this jupyter notebook 07_Model Evaluation.

7 Conclusion

7.1 Future Strategy

- Comparing to 2013, sales in 2014 decreased by 5.3% due to 180 stores closed for renovation. Among those 180 stores 5 of them ranked in top 10 stores that generated high values of sales per customer (Store 842, Store 612, Store 52, Store 540 and Store 903). Our first suggestion to the management team would be: try not to close these stores at the same time for renovation in the future if possible.
- Winter and autumn time are the off seasons of our business. In 2014 we increased the number of daily promotions over all stores in autumn (from September) and we saw a significant increase in average daily sales per store from September to December that year. We should apply this strategy to January and February each year to boost the average daily sales.
- Increase sales per customer for stores that have high average daily sales. Such as Store 262, Store 1114, Store 817, Store 251 As they are the ones have huge amounts of daily customers. If we could encourage each customer spend more in those stores, the impact on increasing total sales for the company would be huge.
- Increase sales per customer for stores that are categorized to Type B, and stores who belongs to assortment extra (assortment b). As these two types of store have the lowest sales per customer comparing to other types. The reason that caused this are due to having competitors more nearby comparing to other types of stores. Therefore, we need to differentiate our products and customer service from our competitors. Suggestions to improve 3 and 4 could be:
    - ❖ Seasonality: Make sure our products are right for the season—whether it's the cold season or allergy season in order to maximize on customer needs and boost their spend. Stock the product before the season hits and have it there when consumers need it
    - ❖ Large Display: Use big displays for seasonal items and redesign our display more frequently to make it more organized and interesting and dynamic with each season or holiday so customers will want to come into the store more often.
    - ❖ Know Our Customer: Conduct customer surveys to find out 1) reasons of why customers would choose competitors over us; 2) what are our customers' destination purchases are vs. impulse purchases, then invest in displays of the former, make sure it's easy to find and easily called out.

7.2 Models

1. In this study in order to build a unified model to forecast for 1115 stores we used Auto Arima, a python package that will automatically find the optimal non-seasonal and seasonal p, d, q values for each time series. We used mean method as the problem agnostic naïve forecast method. Metrics MAE and WAPE were selected as evaluation metrics.

2. We fed 1115 different time series to the Auto Arima model, calculated the MAE for each store. Then we calculated MAE when using mean method for each store.
3. Next, we calculated WAPE of Auto Arima and Mean method for each test week. Compared the results.
4. We came to the conclusion that Auto Arima model outperformed mean method overall, as the average MAE of all stores when using mean method was 1.75 times higher than when using Auto Arima. Furthermore, when we compared WAPE of each week, overall Auto Arima resulted much lower WAPE than mean method. Which convinced us that Auto Arima is a more robust forecasting method comparing to mean method that are currently most commonly used by many store managers.

7.3 Feature Work

1. We realized that Auto Arima model didn't always perform better than mean method in the case of all the stores. For instance, in the case of Store 262, Store 817 and Store 562, we obtained lower MAE from using mean method than from using Auto Arima. When we went back and check the time series pattern for these stores, we noticed they all follow the pattern of white noise and we considered this to be the reason explaining why Auto Arima failed to beat the mean method. As for white noise time series, forecasting with its average value normally is the most ideal solution. Thus, we could optimize our modeling part by adding a white noise test for each time series and the logic should be: when a time series suits the condition of being white noise, we will instead of using Auto Arima, use mean method to forecast the future sales. Additionally, we will also implement this optimized process to our interactive dashboard.
2. In this study we didn't add any exogenous features to Auto Arima model. For the next step of this study, we can try to explore more by adding features to the model such as week number or daily promotion count in each week in order to improve forecasting accuracy.

References

1. https://www.kaggle.com/c/rossmann-store-sales/data?select=train.csv (assessed Dec 14, 2020)
2. https://www.kaggle.com/c/rossmann-store-sales/data?select=store.csv (assessed Dec 14, 2020)
3. https://towardsdatascience.com/time-series-forecasting-using-auto-arima-in-python-bb83e49210cd , *Sushmitha Pulagam* (assessed Dec 20, 2020)
4. *Rob J Hyndman, George Athanasopoulos (2018) Forecasting: Principles and Practice,* 2nd edition, OTexts: Melbourne, Australia.
5. https://www.linkedin.com/pulse/reading-acf-pacf-plots-missing-manual-cheatsheet-saqib-ali/ , *Saqib Ali* (assessed Dec 28, 2020)
6. https://towardsdatascience.com/introduction-to-aic-akaike-information-criterion-9c9ba1c96ced , *Alexandre Zajic* (assessed Jan 2, 2021)
7. https://uhurunetwork.com/business-dashboard/ (assessed Jan 14, 2021)
8. https://mashmetrics.com/marketing-reports-dashboards-performance-alerts-pros-cons-of-each/ (assessed Jan 14, 2021)
9. https://machinelearningmastery.com/how-to-develop-baseline-forecasts-for-multi-site-multivariate-air-pollution-time-series-forecasting/ „*Jason Brownlee* (assessed Jan 20, 2021)
10. https://machinelearningmastery.com/time-series-forecasting-methods-in-python-cheat-sheet/ , *Jason Brownlee* (assessed Jan 20, 2021)
11. https://machinelearningmastery.com/remove-trends-seasonality-difference-transform-python/, *Jason Brownlee* (assessed Jan 22, 2021)
12. https://www.baeldung.com/cs/mape-vs-wape-vs-wmape (assessed Jan 22, 2021)
13. https://www.thebalancesmb.com/pharmacy-merchandising-how-to-boost-sales-2663853, *Amanda Baltazar* (assessed Jan 22, 2021)
14. https://en.wikipedia.org/wiki/Kaggle (assessed Jan 1, 2021)
15. Springboard – DSC Capstone Project II Detecting Potential Candidates Who are Looking for a New Job ((Nov 2020) *Yang Liu Kunz*

Appendix

1. Autoregressive models of order p: $y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t$, where ε_t is white noise.

2. Moving average model of order q: $y_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} \cdots + \theta_q \varepsilon_{t-q}$, where ε_t is white noise.

3. Non-seasonal ARIMA model:
   $y_t' = c + \phi_1 y'_{t-1} + \cdots + \phi_p y'_{t-p} + \theta_1 \varepsilon_{t-1} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t$
   where y_t' is the differenced series (it may have been differenced more than once). The "predictors" on the right hand side include both lagged values of y_t and lagged errors.

4. Mean absolute error: $MAE = \frac{1}{n}\sum_{t=1}^{n} |A_t - F_t|$