UNIVERSITY OF ECONOMICS AND LAWS

FINAL PROJECT REPORT

DATA MINING COURSE

TOPIC:

FORECAST THE SALES IN CONDITIONS WITH COMPETITION BY USING DEEP LEARNING WITH REGRESSION

Lecturers:

M.A Phan Huy Tam

Student:

Tran Thu Hien – K214140938

Ho Chi Minh City, January 14, 2024

ACKNOWLEDGEMENTS

I would like to start by sending our most sincere gratitude to M.A.Phan Huy Tan, who helped and personally taught us in the Data mining course this semester. I have welcomed the teacher's attention, assistance, direction with great excitement as I have studied and understood this subject. I gained an overview and a deeper understanding of data analysis from the teacher's knowledge.

While it's possible that knowledge is endless, there are certain restrictions on what each person can learn. Because of this, I will inevitably make mistakes when writing the research paper. In order to make our research more thorough, I eagerly await instructor recommendations.

Finally, I send our best wishes for continued health, joy, and professional success in teaching!

Thu Hien

TABLE OF CONTENTS

LIST OF FIGURES

PROJECT OVERVIEW

1. Business Problem

Rossmann operates over 3,000 drug stores in 7 European countries. Currently, Rossmann store managers are tasked with predicting their daily sales for up to six weeks in advance. Store sales are influenced by many factors, including promotions, competition, school and state holidays, seasonality, and locality. With thousands of individual managers predicting sales based on their unique circumstances, the accuracy of results can be quite varied.

To address these challenges, Rossmann aims to develop a robust and reliable sales forecasting model that takes into account the various factors affecting store sales. The model should provide accurate predictions for up to six weeks in advance, allowing store managers to make informed decisions and optimize their sales planning efforts. By leveraging historical sales data, information about promotions and competition, and insights from seasonality and holidays, Rossmann aims to improve the accuracy and consistency of sales forecasts across its stores.

The successful development and implementation of a sales forecasting model will have several benefits. It will enable Rossmann to optimize its inventory levels, ensuring that stores have the right products in the right quantities at the right time. This will lead to improved customer satisfaction by reducing instances of stockouts and ensuring a smooth shopping experience. Additionally, accurate sales forecasts will aid in the planning and execution of effective promotional strategies, leading to increased sales and revenue.

2. Objectives

2.1. Overall objective

The objective is to develop a sales forecasting model for Rossmann drug stores that takes into account various factors such as promotions, competition, holidays, seasonality, and locality. The model should provide accurate predictions of daily sales for up to six weeks in advance, helping store managers make informed decisions and improve their sales planning.

2.2. Research questions

The project's main goal was attained by utilizing the following research questions.

1. What information may be utilized to pinpoint the sales revenue?

2. What percentage of customers will purchase in the store? How should businesses retain customers?

3. What techniques are available for sales prediction?

2.3. Research objectives

The above research questions then advised on the following research objectives.

1. Locate information about clients and sales for businesses.

2. List the elements that influence how consumers respond to the company's offerings.

3. Select the appropriate algorithms for sales prediction.

4. Create mental images of outcomes, conclusions, and solutions.

3. Objects and scopes

3.1. Objects

Sales statistics from Rossman sales, providing detailed information about the company’s event of promotions.

3.2. Scopes

Time scope: The data used for the dataset had a time range from Jan 01 2023 to July 31 2015.

4. Research method

The study used two methods: Qualitative and Quantitative.

5. Process

Data Collection: Gather historical sales data from Rossmann drug stores, including information about promotions, competition, holidays, seasonality, and locality. Collect additional relevant data sources, such as weather data or economic indicators, if available.

Data Preprocessing: Clean the collected data by handling missing values, outliers, and inconsistencies. Transform and aggregate the data as necessary for analysis. Ensure compatibility and consistency across different data sources.

Feature Engineering: Extract meaningful features from the data that capture the influence of promotions, competition, holidays, seasonality, and locality on sales.

Create lag variables, rolling averages, or other derived features that capture trends and patterns in the data. Consider incorporating external data sources to enhance the predictive power of the model.

Data Split: Split the preprocessed data into training and validation sets. The training set will be used to train the sales forecasting model, while the validation set will be used to evaluate its performance.

Model Selection: Choose an appropriate model for sales forecasting. Consider various models such as regression models (e.g., linear regression, decision trees), time series models (e.g., ARIMA, Prophet), or machine learning algorithms (e.g., random forest, gradient boosting). Select a model that can effectively capture the complexities and dynamics of sales data.

Model Training: Train the selected model on the training set using the prepared features. Optimize the model's hyperparameters to achieve the best performance. Consider techniques like cross-validation or grid search to find the optimal hyperparameter values.

Model Evaluation: Evaluate the trained model's performance on the validation set using appropriate evaluation metrics such as mean absolute error (MAE), root mean squared error (RMSE), or mean absolute percentage error (MAPE). Assess the model's accuracy and its ability to capture the variations in sales due to promotions, competition, holidays, seasonality, and locality.

Model Refinement: Analyze the model's performance and identify any limitations or areas for improvement. Refine the model by adjusting its parameters, exploring different modeling techniques, or incorporating additional features. Iterate on the model training and evaluation process to enhance its accuracy and predictive capabilities.

6. Tools and Programing language

6.1. Tools

Pycharm

6.2. Programing language

Python.

7. Structure of project

There are 4 chapters in the project.

CHAPTER 1: THEORETICAL BASIS

Chapter overview: The background information to assist the research is presented in this chapter includes linear regression, machine learning / deep learning, and other algorithms like model accuracy, etc.

1.1. Linear Regression

1.1.1. The definition

Linear regression analysis is used to predict the value of a variable based on the value of another variable. The variable you want to predict is called the dependent variable. The variable you are using to predict the other variable's value is called the independent variable.

This form of analysis estimates the coefficients of the linear equation, involving one or more independent variables that best predict the value of the dependent variable. Linear regression fits a straight line or surface that minimizes the discrepancies between predicted and actual output values. There are simple linear regression calculators that use a “least squares” method to discover the best-fit line for a set of paired data. You then estimate the value of X (dependent variable) from Y (independent variable). (MALI, 2023)

1.1.2. Key assumptions of effective linear regression

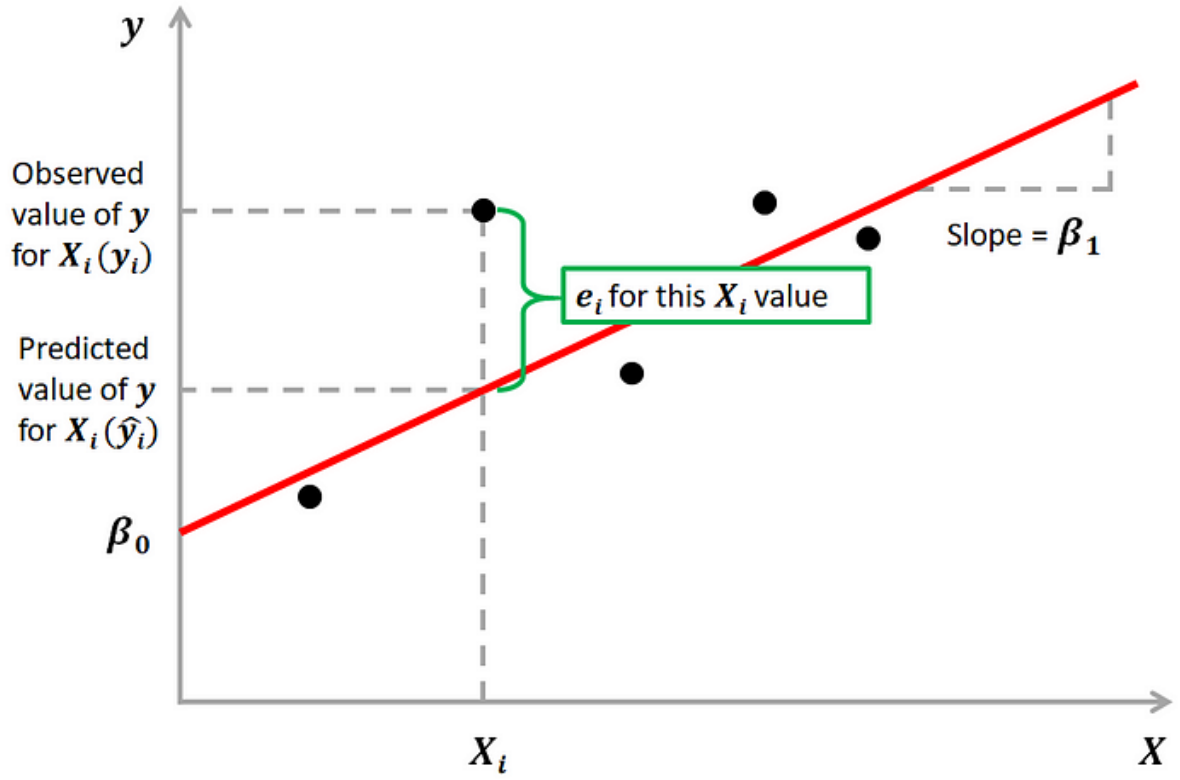


Figure 1: Linear Regression model and chart

Assumptions to be considered for success with linear-regression analysis:

Firstly, for each variable: Consider the number of valid cases, mean and standard deviation.

For each model: Consider regression coefficients, correlation matrix, part and partial correlations, multiple R, R2, adjusted R2, change in R2, standard error of the estimate, analysis-of-variance table, predicted values and residuals. Also, consider 95-percent-confidence intervals for each regression coefficient, variance-covariance matrix, variance inflation factor, tolerance, Durbin-Watson test, distance measures (Mahalanobis, Cook and leverage values), DfBeta, DfFit, prediction intervals and case-wise diagnostic information.

Plots: Consider scatterplots, partial plots, histograms and normal probability plots.

Data: Dependent and independent variables should be quantitative. Categorical variables, such as religion, major field of study or region of residence, need to be recoded to binary (dummy) variables or other types of contrast variables.

Other assumptions: For each value of the independent variable, the distribution of the dependent variable must be normal. The variance of the distribution of the dependent variable should be constant for all values of the independent variable. The relationship between the dependent variable and each independent variable should be linear and all observations should be independent.

1.2. Machine learning

1.2.1. The definition

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. (Kanade, 2022)

Machine learning is an important component of the growing field of data science. Through the use of statistical methods, algorithms are trained to make classifications or predictions, and to uncover key insights in data mining projects. These insights subsequently drive decision making within applications and businesses, ideally impacting key growth metrics. As big data continues to expand and grow, the market demand for data scientists will increase. They will be required to help identify the most relevant business questions and the data to answer them.

Machine learning algorithms are typically created using frameworks that accelerate solution development, such as TensorFlow and PyTorch.

1.2.2. Machine learning method

1.2.2.1. Supervised machine learning

Supervised learning, also known as supervised machine learning, is defined by its use of labeled datasets to train algorithms to classify data or predict outcomes accurately. As input data is fed into the model, the model adjusts its weights until it has been fitted appropriately. This occurs as part of the cross validation process to ensure that the model avoids overfitting or underfitting. Supervised learning helps organizations solve a variety of real-world problems at scale, such as classifying spam in a separate folder from your inbox. Some methods used in supervised learning include neural networks, naïve bayes, linear regression, logistic regression, random forest, and

support vector machine (SVM).

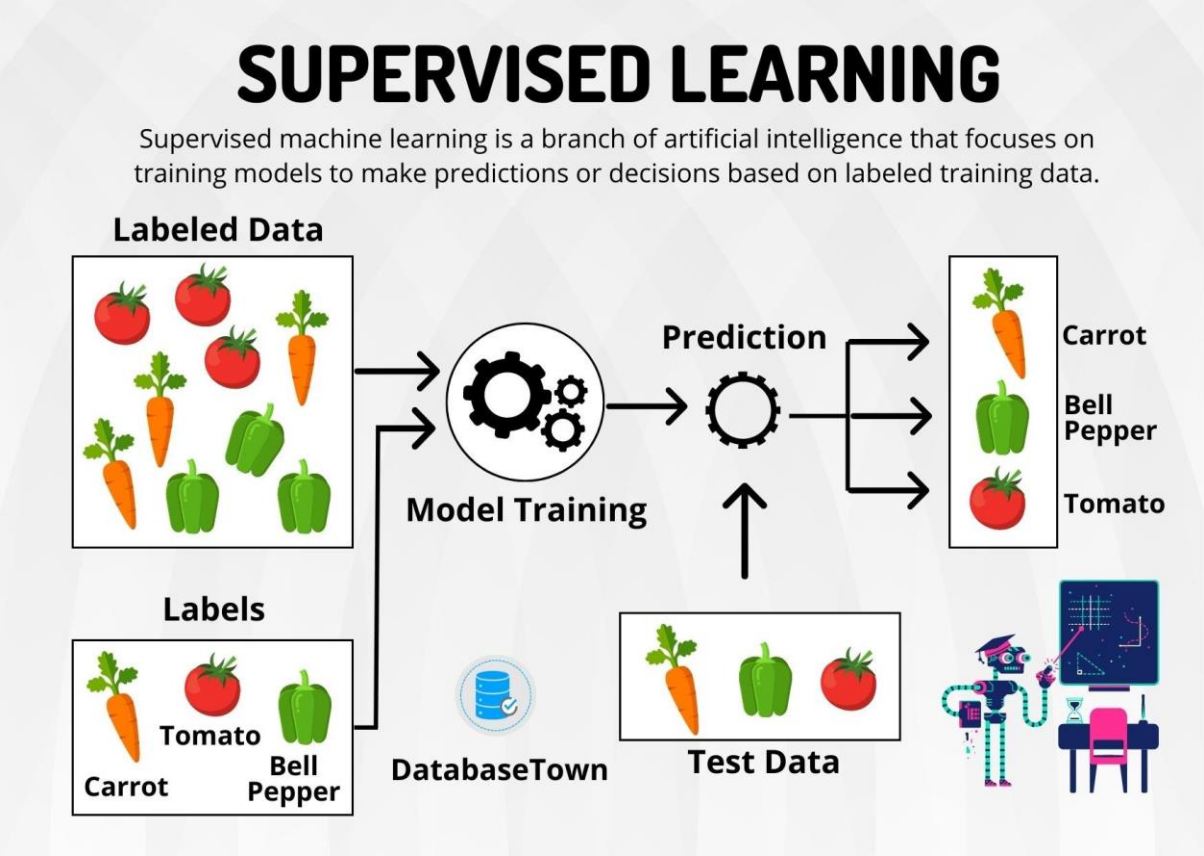


Figure 2: Supervised learning method

1.2.2.2. Unsupervised machine learning

Unsupervised learning, also known as unsupervised machine learning, uses machine learning algorithms to analyze and cluster unlabeled datasets. These algorithms discover hidden patterns or data groupings without the need for human intervention. This method's ability to discover similarities and differences in information make it ideal for exploratory data analysis, cross-selling strategies, customer segmentation, and image and pattern recognition. It's also used to reduce the number of features in a model through the process of dimensionality reduction. Principal component analysis (PCA) and singular value decomposition (SVD) are two common approaches for this. Other algorithms used in unsupervised learning include neural networks, k-means clustering, and probabilistic clustering methods.

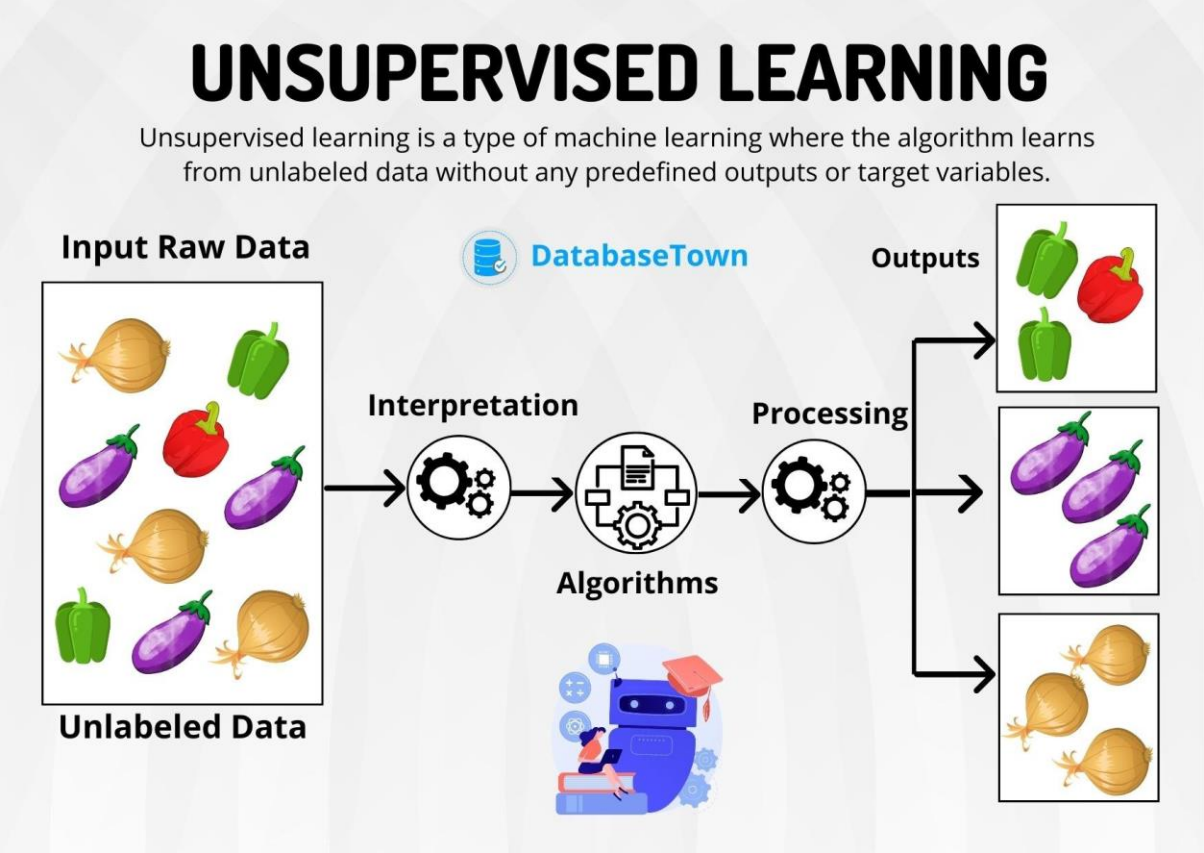


Figure 3:Un - Supervised learning method

1.2.3. Semi-supervised learning

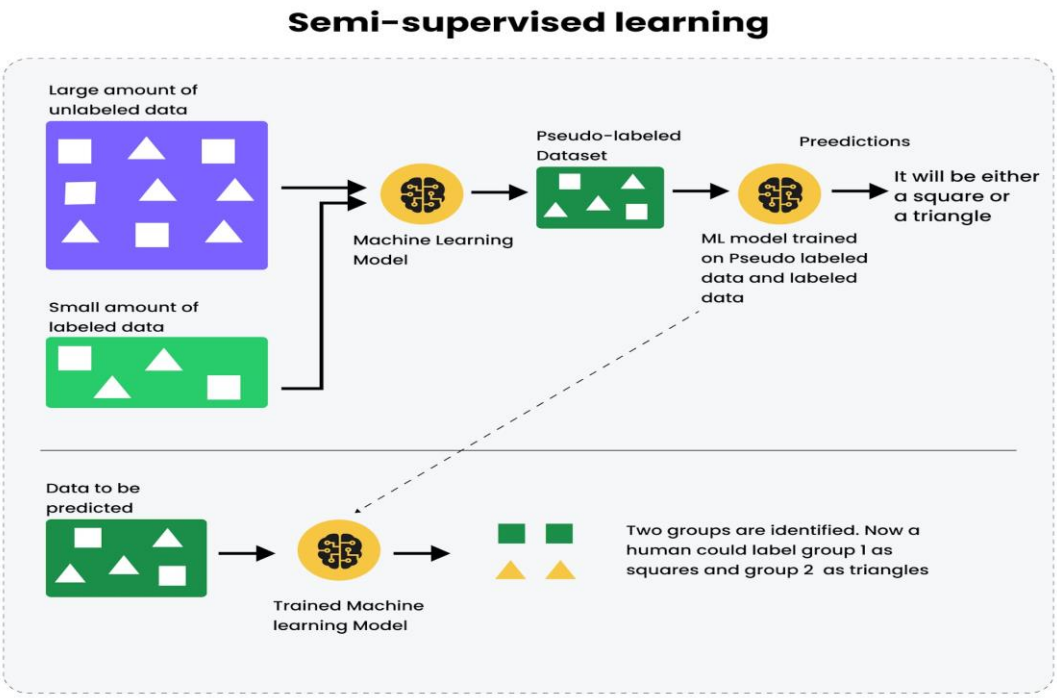


Figure 4: Semi-supervised learning model

Semi-supervised learning offers a happy medium between supervised and unsupervised learning. During training, it uses a smaller labeled data set to guide classification and feature extraction from a larger, unlabeled data set. Semi-supervised learning can solve the problem of not having enough labeled data for a supervised learning algorithm. It also helps if it's too costly to label enough data.

CHAPTER 2: DATA PREPARATION

Chapter overview: After learning the fundamentals in Chapter 1, Chapter 2 will start to process data.

2.1. Data Introduction

The Rossmann sales data provides a comprehensive record of historical sales across the Rossmann drug store chain. This dataset encompasses a range of information that is crucial for understanding and analyzing the factors influencing sales performance. The dataset includes details on store-specific sales, promotions, competition, holidays, seasonality, and locality. (Will Cukierski, 2015)

2.2. Data understanding

There are 3 data files: store, test, train. In it, the training file includes the following properties:

- **Id** - an Id that represents a (Store, Date) duple within the test set
- **Store** - a unique Id for each store
- **Sales** - the turnover for any given day (this is what you are predicting)
- **Customers** - the number of customers on a given day
- **Open** - an indicator for whether the store was open: 0 = closed, 1 = open
- **StateHoliday** - indicates a state holiday. Normally all stores, with few exceptions, are closed on state holidays. Note that all schools are closed on public holidays and weekends. a = public holiday, b = Easter holiday, c = Christmas, 0 = None
- **SchoolHoliday** - indicates if the (Store, Date) was affected by the closure of public schools
- **StoreType** - differentiates between 4 different store models: a, b, c, d
- **Assortment** - describes an assortment level: a = basic, b = extra, c = extended
- **CompetitionDistance** - distance in meters to the nearest competitor store
- **CompetitionOpenSince[Month/Year]** - gives the approximate year and month of the time the nearest competitor was opened
- **Promo** - indicates whether a store is running a promo on that day
- **Promo2** - Promo2 is a continuing and consecutive promotion for some stores: 0 = store is not participating, 1 = store is participating

- **Promo2Since[Year/Week]** - describes the year and calendar week when the store started participating in Promo2
- **PromoInterval** - describes the consecutive intervals Promo2 is started, naming the months the promotion is started anew. E.g. "Feb,May,Aug,Nov" means each round starts in February, May, August, November of any given year for that store

```
In 142  1  train.info()
           Executed at 2024.01.14 23:28:36 in 290ms

 #   Column           Non-Null Count   Dtype
---  ------           --------------   -----
 0   Store            844338 non-null  int64
 1   DayOfWeek        844338 non-null  int64
 2   Sales            844338 non-null  float64
 3   Customers        844338 non-null  int64
 4   Open             844338 non-null  int64
 5   Promo            844338 non-null  int64
 6   StateHoliday     844338 non-null  object
 7   SchoolHoliday    844338 non-null  int64
 8   Year             844338 non-null  int32
 9   Month            844338 non-null  int32
 10  Day              844338 non-null  int32
 11  WeekOfYear       844338 non-null  UInt32
 12  SalePerCustomer  844338 non-null  float64
dtypes: UInt32(1), float64(2), int32(3), int64(6), object(1)
memory usage: 78.1+ MB
```

Figure 5: Data set details prior to processing

```
In 143  1  store.info()
```

Executed at 2024.01.14 23:30:10 in 174ms

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1115 entries, 0 to 1114
Data columns (total 10 columns):
 #   Column                     Non-Null Count  Dtype  
---  ------                     --------------  -----  
 0   Store                      1115 non-null   int64  
 1   StoreType                  1115 non-null   object 
 2   Assortment                 1115 non-null   object 
 3   CompetitionDistance        1115 non-null   float64
 4   CompetitionOpenSinceMonth  1115 non-null   float64
 5   CompetitionOpenSinceYear   1115 non-null   float64
 6   Promo2                     1115 non-null   int64  
 7   Promo2SinceWeek            1115 non-null   float64
 8   Promo2SinceYear            1115 non-null   float64
 9   PromoInterval              1115 non-null   object 
dtypes: float64(5), int64(2), object(3)
memory usage: 87.2+ KB
```

Figure 6: Store 's data

```python
# data extraction
train['Year'] = train.index.year
train['Month'] = train.index.month
train['Day'] = train.index.day
train['WeekOfYear'] = train.index.isocalendar().week

# adding new variable
train['SalePerCustomer'] = train['Sales']/train['Customers']
train['SalePerCustomer'].describe()
```

Executed at 2024.01.14 21:36:01 in 465ms

8 rows Length: 8, dtype: float64

	SalePerCustomer
count	844340.000000
mean	9.493619
std	2.197494
min	0.000000
25%	7.895563
50%	9.250000
75%	10.899729
max	64.957854

Figure 7: Sale per customer distribution

On average customers spend about 9.50$ per day. Though there are days with Sales equal to zero. So this is the weird point in the data, need to be processing.

2.3. Data preprocessing

I used ECDF: empirical cumulative distribution function to find the reason why there are still have revenue even if that day the store is closed. About 20% of data has zero amount of sales / customers that I need to deal with and almost 80% of time daily

amount of sales was less than 1000. So what about zero sales, is it only due to the fact that the store is closed?

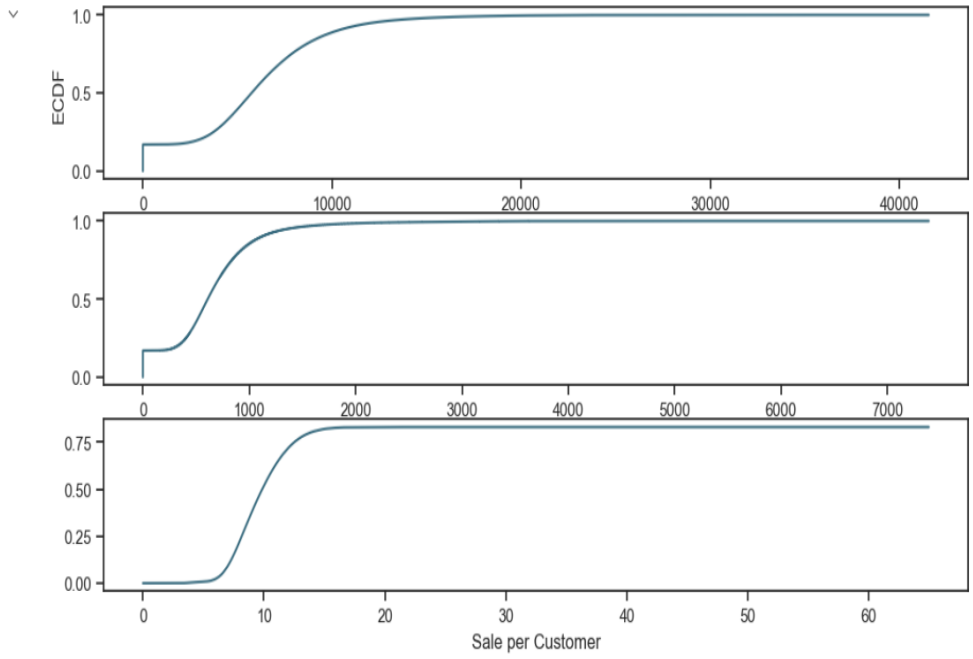


Figure 8: Sales per customer by using ECDF

And then proceed to deal with the missing value on this train's data. There're 172817 closed stores in the data. It is about 10% of the total amount of observations. To avoid any biased forecasts I will drop these values.

```
# closed stores
train[(train.Open == 0) & (train.Sales == 0)].head()
```

Executed at 2024.01.14 21:36:01 in 91ms

5 rows × 13 columns

Date	Store	DayOfWeek	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday	Year
2015-07-31	292	5	0	0	0	1	0	1	2015
2015-07-31	876	5	0	0	0	1	0	1	2015
2015-07-30	292	4	0	0	0	1	0	1	2015
2015-07-30	876	4	0	0	0	1	0	1	2015
2015-07-29	292	3	0	0	0	1	0	1	2015

```
len(train[(train.Open == 0) & (train.Sales == 0)])
```

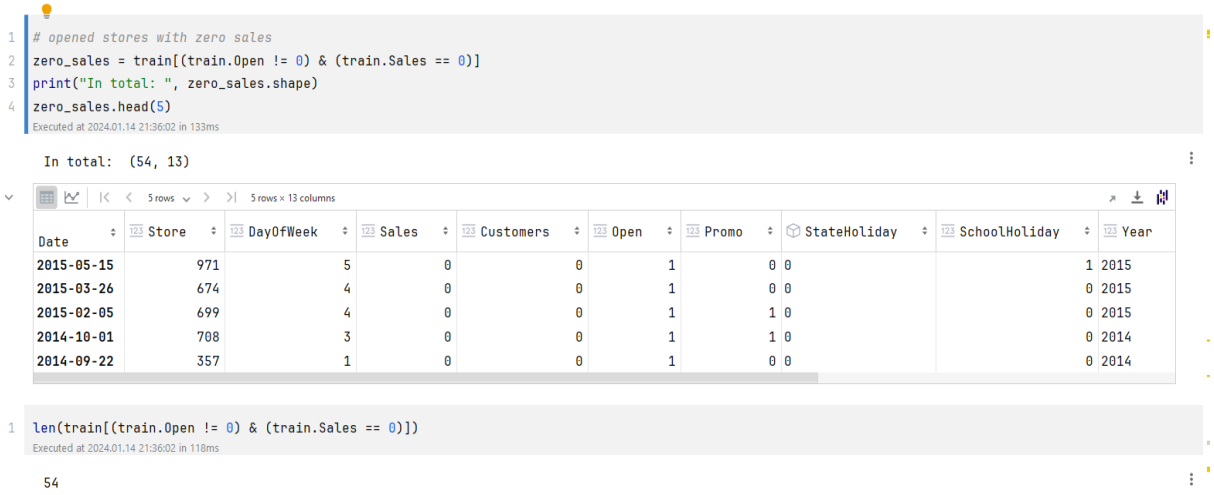
Executed at 2024.01.14 21:36:01 in 88ms

```
172817
```

Figure 9: Statistic the number of closed store

Upon analyzing the Rossmann sales data, it was observed that certain stores had recorded zero sales on working days. This occurrence could be attributed to external

factors that influenced the store's operations during that period. It is important to note that the dataset covers a limited timeframe of only 54 days, which further supports the possibility of external influences impacting the recorded sales figures.



```
# opened stores with zero sales
zero_sales = train[(train.Open != 0) & (train.Sales == 0)]
print("In total: ", zero_sales.shape)
zero_sales.head(5)
```

Executed at 2024.01.14 21:36:02 in 133ms

```
In total:  (54, 13)
```

5 rows × 13 columns

Date	Store	DayOfWeek	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday	Year
2015-05-15	971	5	0	0	1	0	0	1	2015
2015-03-26	674	4	0	0	1	0	0	0	2015
2015-02-05	699	4	0	0	1	1	0	0	2015
2014-10-01	708	3	0	0	1	1	0	0	2014
2014-09-22	357	1	0	0	1	0	0	0	2014

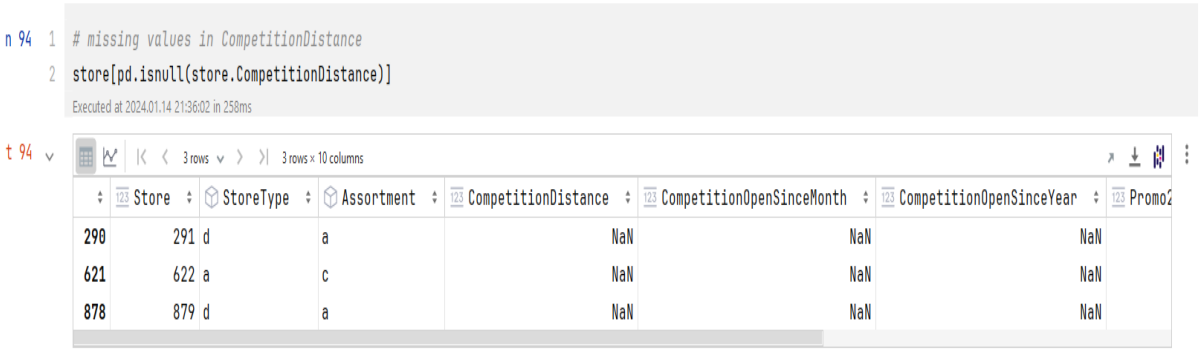
```
len(train[(train.Open != 0) & (train.Sales == 0)])
```

Executed at 2024.01.14 21:36:02 in 118ms

```
54
```

Figure 10: Stores without sales on working days

Upon closer examination of the data, it appears that the information regarding sales on certain working days is missing. There is no discernible pattern or clear explanation for the absence of sales data on these days. In such cases, it is reasonable to replace the missing values (NaN) with the median values from the available data.



```
# missing values in CompetitionDistance
store[pd.isnull(store.CompetitionDistance)]
```

Executed at 2024.01.14 21:36:02 in 258ms

3 rows × 10 columns

	Store	StoreType	Assortment	CompetitionDistance	CompetitionOpenSinceMonth	CompetitionOpenSinceYear	Promo2
290	291	d	a	NaN	NaN	NaN	
621	622	a	c	NaN	NaN	NaN	
878	879	d	a	NaN	NaN	NaN	

Figure 11: Missing data on store's data

After filling missing value, now I can base on this normal data to continue the next processing step.

Add Code Cell

```
# replace NA's by 0
store.fillna(0, inplace = True)
```

Executed at 2024.01.14 21:36:02 in 233ms

```
print("Joining train set with an additional store information.")

# by specifying inner join we make sure that only those observations
# that are present in both train and store sets are merged together
train_store = pd.merge(train, store, how = 'inner', on = 'Store')

print("In total: ", train_store.shape)
train_store.head()
```

Executed at 2024.01.14 21:36:02 in 497ms

```
Joining train set with an additional store information.
In total:  (844338, 22)
```

Figure 12: Final data to investigate

2.5. Exploratory Data Analysis

To determine the most selling and crowded StoreType, let's calculate the overall sum of Sales and Customers across all StoreTypes. While StoreType B may have the highest average of Sales, it is essential to consider the overall performance by aggregating the data from all StoreTypes. By doing so, we can gain a comprehensive understanding of the sales and customer traffic across the different StoreTypes.

```
1 train_store.groupby('StoreType')['Sales'].describe()
2
```

Executed at 2024.01.14 21:36:02 in 132ms

4 rows 4 rows × 8 columns

StoreType	count	mean	std	min	25%	50%	75%	max
a	457042.0	6925.697986	3277.351589	46.0	4695.25	6285.0	8406.00	41551.0
b	15560.0	10233.380141	5155.729868	1252.0	6345.75	9130.0	13184.25	38722.0
c	112968.0	6933.126425	2896.958579	133.0	4916.00	6408.0	8349.25	31448.0
d	258768.0	6822.300064	2556.401455	538.0	5050.00	6395.0	8123.25	38037.0

```
train_store.groupby('StoreType')[['Customers', 'Sales']].sum()
```

Executed at 2024.01.14 21:36:02 in 85ms

4 rows × 2 columns

StoreType	Customers	Sales
a	363541431	3165334859
b	31465616	159231395
c	92129705	783221426
d	156904995	1765392943

Figure 13: Total sum off customers and sales base on each storetype

Based on the analysis, it is evident that stores of type A have the highest total Sales and Customers. Following closely behind is StoreType D, ranking second in both categories. Now, let's examine the sales and customer trends across different date periods. To accomplish this, we can utilize Seaborn's facet grid, which is a powerful visualization tool.

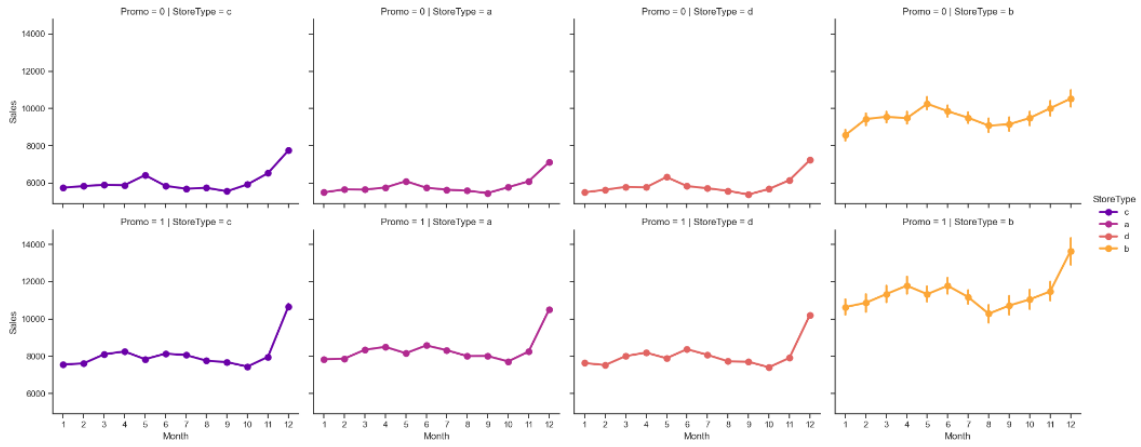


Figure 14: Store type with the affected promotions

StoreType B stands out as its sales scale is notably higher compared to other store types. This suggests that StoreType B may have specific attributes or strategies that contribute to increased sales performance. Additionally, the presence of the first promotion

appears to have a significant impact on sales across all store types, further influencing the observed trends.

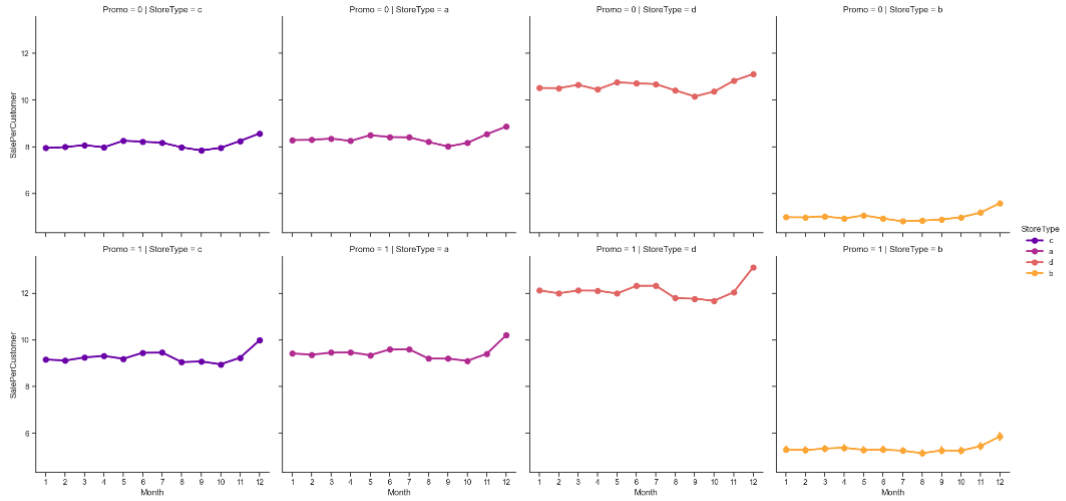


Figure 15: Sales per customer in month with promotions

Surprisingly, despite the initial impression from the previous plots, StoreType B is not actually the most selling or performant store type. Upon further analysis, it is revealed that StoreType D has the highest SalePerCustomer amount. When the first promotion (Promo) is in effect, StoreType D achieves an impressive SalePerCustomer amount of approximately 12€, and even without the promotion, it still maintains a high SalePerCustomer of around 10€. StoreType A and C closely follow with a SalePerCustomer of about 9€. The low SalePerCustomer amount observed for StoreType B indicates that customers at this store tend to have smaller shopping carts, purchasing items in relatively small quantities or with lower price points. This suggests

that the majority of customers at StoreType B may primarily buy "small" items rather than large or expensive products.

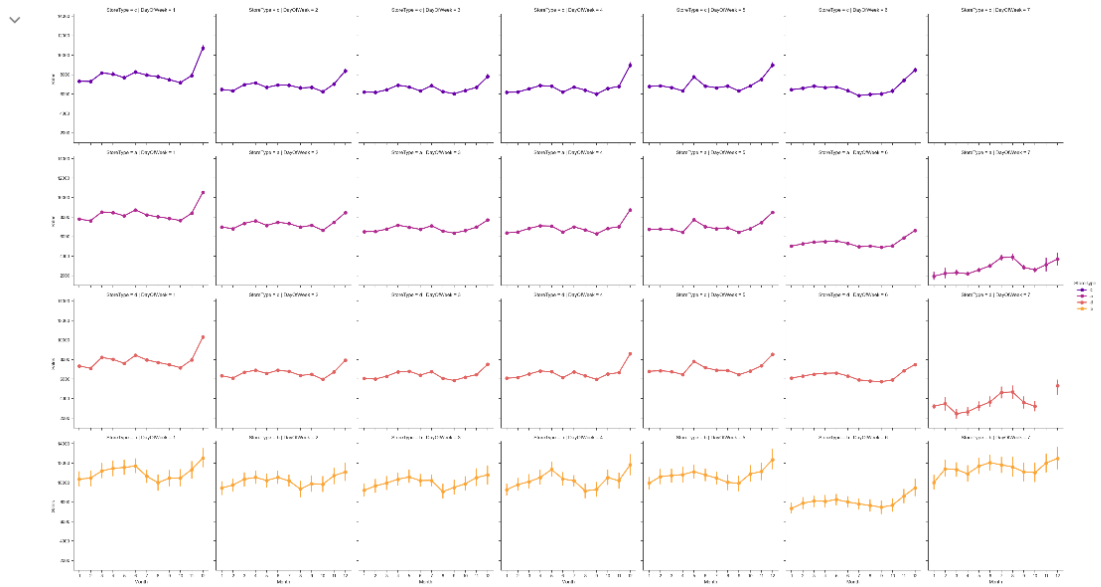


Figure 16: Store working days with sales

The analysis reveals that stores belonging to StoreType C are consistently closed on Sundays, whereas stores of other StoreTypes are generally open on Sundays. This observation suggests that StoreType C follows a regular practice of closing on Sundays across all stores. On the other hand, stores of StoreType D exhibit a slightly different

pattern. Specifically, from October to December, stores of StoreType D are closed on Sundays, while they may be open on Sundays during other months.

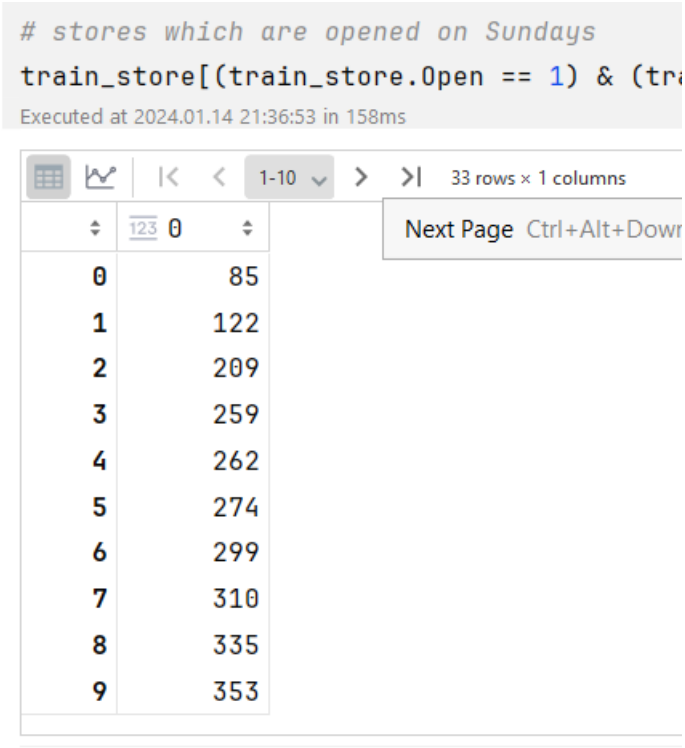


Figure 17: List of store opened on Sundays

As previously mentioned, there is a notable positive correlation between the amount of Sales and the number of Customers at a store. This suggests that as the number of Customers increases, there is a corresponding increase in Sales, indicating a strong relationship between these two variables. Additionally, another positive correlation can be observed between the presence of a running promotion (Promo equal to 1) and the

number of Customers. This implies that when a promotion is active, there tends to be a higher influx of Customers to the store. The promotional offers or discounts likely attract more individuals, resulting in increased customer traffic.

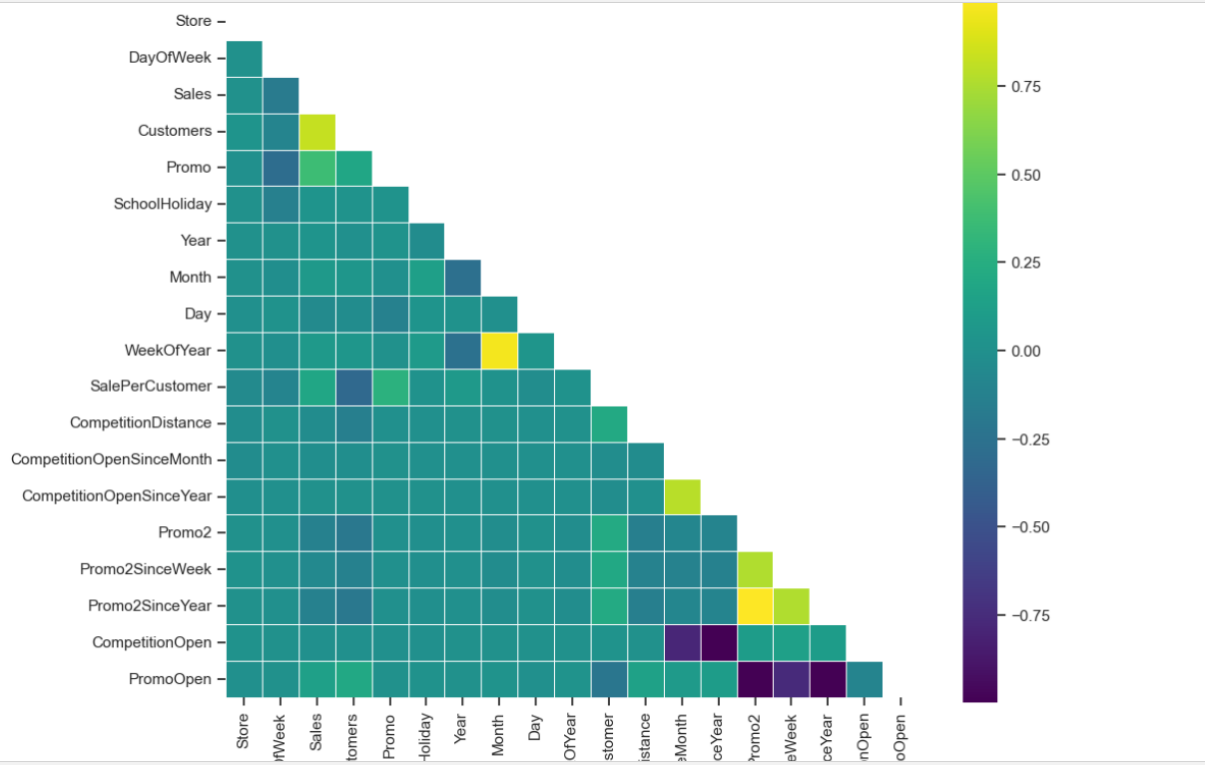


Figure 18: Correlation between factors

When there is no promotion, Sales tend to peak on Sundays. However, this trend is mainly driven by StoreType A, B, and D, as StoreType C does not operate on Sundays. Stores that run promotions experience the highest Sales on Mondays. This finding suggests that Monday is a strategically significant day for driving sales when promotions are active. The same trend applies to stores that have both promotions running simultaneously (Promo and Promo2). Promo2 alone does not show a significant correlation with changes in the Sales amount. This is supported by the blue-colored area

on the heatmap, indicating a weak correlation between Promo2 and Sales.

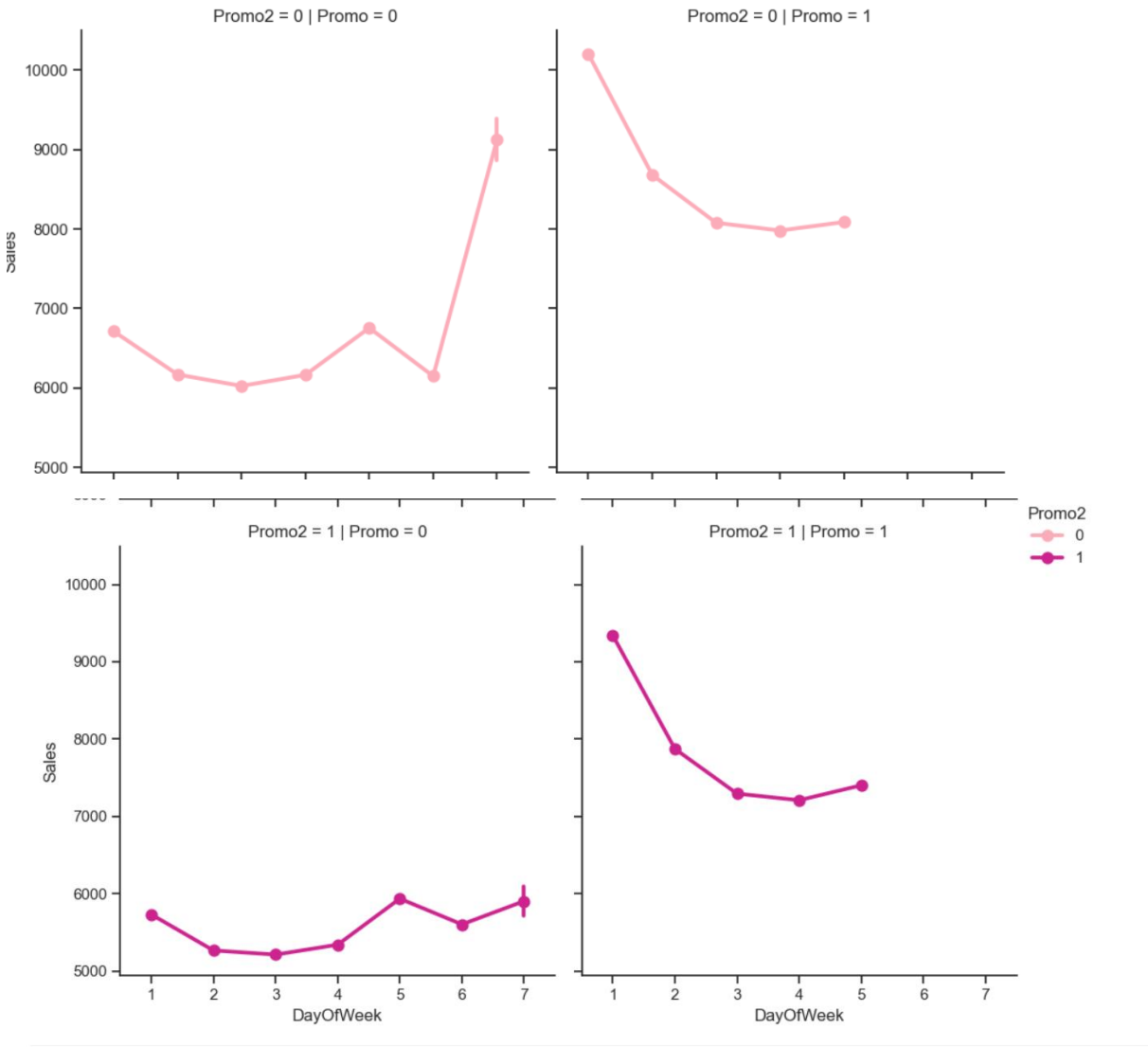


Figure 19: Promotions effective to the sales

In summary, the exploratory data analysis reveals that StoreType A is the most successful in terms of sales and customer traffic, while StoreType D has the highest "Sale per Customer" metric, indicating larger shopping carts, potentially in rural areas. StoreType B attracts customers who make smaller purchases more frequently, likely due to its accessibility. Mondays see increased sales when a single promotion is active, and Sundays experience high sales even without promotions. However, Promo2 alone does not show a significant correlation with sales. These insights can guide businesses in optimizing their marketing strategies and promotions to maximize sales and cater to customer preferences.

CHAPTER 3: Time-Series Analysis per Store Type

Chapter Overview: Once we completed our exploratory data analysis, we discovered some crucial insights. We found that sales are influenced not only by the promotions themselves but also by other factors such as seasonality and customer preferences. With this in mind, our next step is to gain a deeper understanding of the daily sales for each store type. By examining the sales patterns in more detail, we can uncover valuable information about how different store types perform on a day-to-day basis.

3.1. Seasonality

To represent different store types, we have selected four stores from our dataset:

- Store number 2 will represent StoreType A.
- Store number 85 will represent StoreType B.
- Store number 1 will represent StoreType C.
- Store number 13 will represent StoreType D.

In order to gain a clearer understanding of the current trends, it is beneficial to downsample the data from a daily frequency to a weekly frequency. This can be achieved using the resample method, which aggregates the daily data into weekly

intervals. By doing so, we can observe the overarching patterns and trends in a more concise and informative manner.

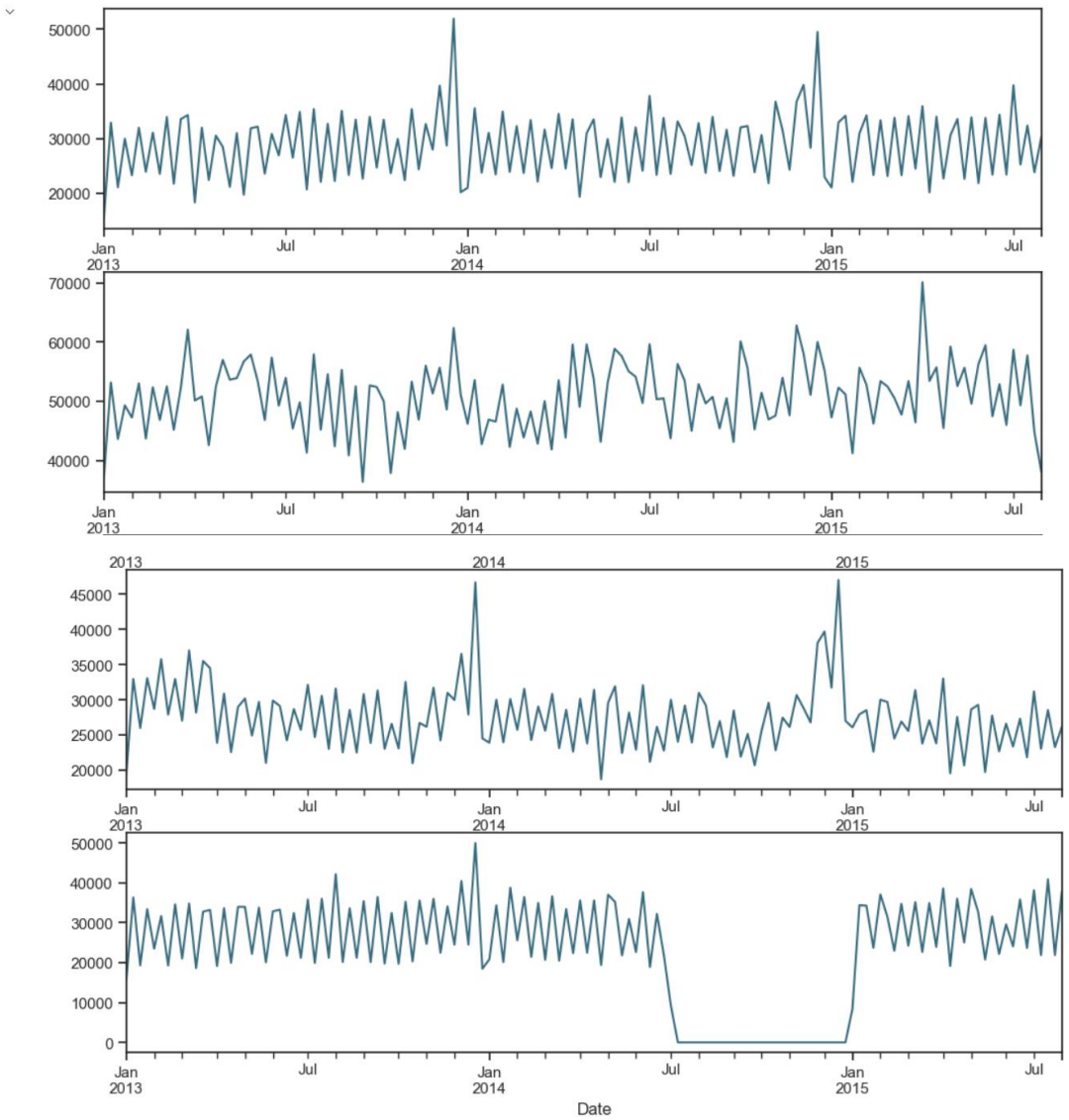


Figure 20: The trends for sales depend on holidays

During the Christmas season, we typically observe a peak in retail sales for StoreType A and StoreType C. However, following the holiday period, sales for both store types tend to decline. It is worth noting that StoreType D, located at the bottom, may have exhibited a similar trend. Due to a lack of information for the period between July 2014

and January 2015, we are unable to analyze the sales patterns for these stores as they were closed during that time.

3.2. Yearly trend

In general, the overall sales show an increasing trend. However, StoreType C, which is positioned third from the top, does not follow this pattern. Despite being one of the store types in the dataset, it does not experience the same upward trajectory as the others. On the other hand, StoreType A, which is the highest-selling store type in the dataset, appears to have the potential to follow a similar decreasing trajectory as StoreType C did.

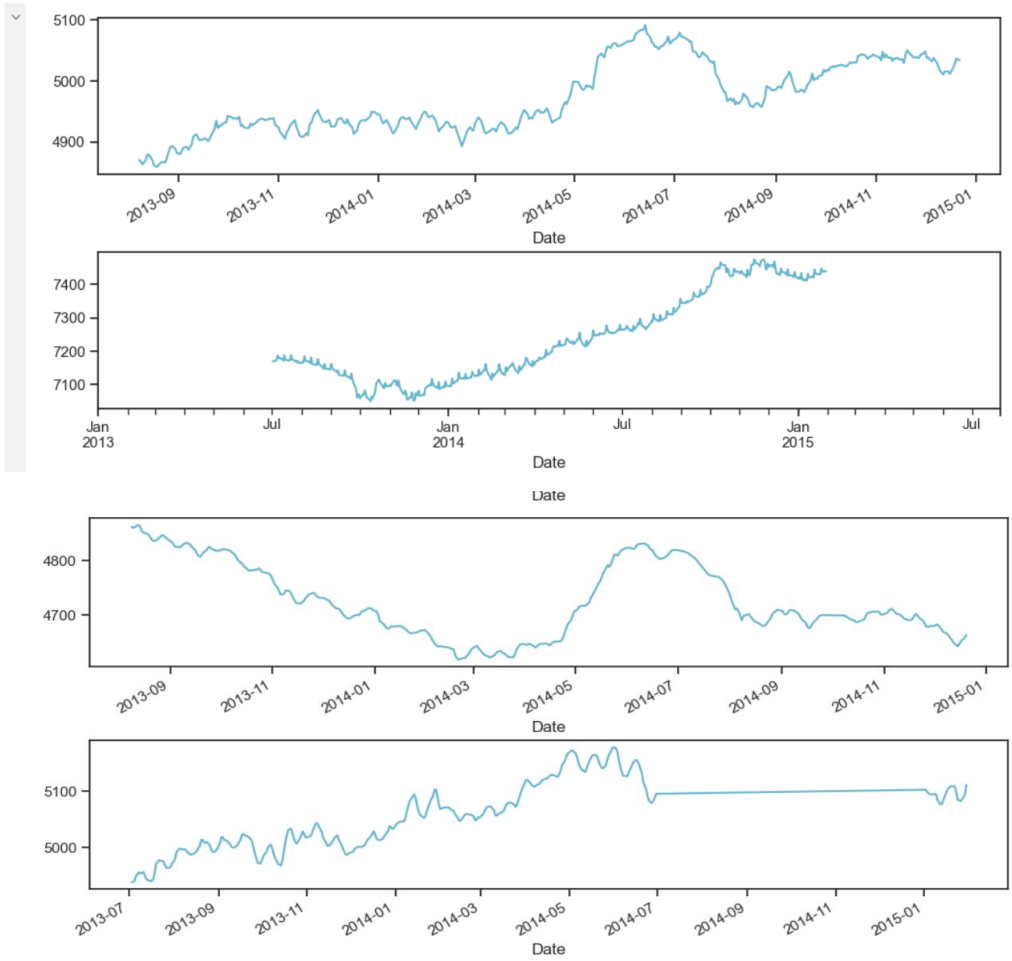


Figure 21: Trending of sales in year

CHAPTER 4: VISUALIZATION OF REGRESSION MODEL

Chapter overview: Analyzing the seasonal nature of the dataset helps improve the accuracy of predictions. Each store type is influenced by specific holiday seasons throughout the year. We will utilize a regression model to predict the revenue of the stores based on these characteristics.

4.1. Create the quarter data

To accurately account for the impact of special features on sales predictions, it is necessary to create quarterly data. This approach allows for a comprehensive analysis of the specific factors that influence sales performance. By organizing the data into quarterly segments, we can readily incorporate and consider the unique characteristics and dynamics associated with each quarter. This enables a more effective prediction of sales by incorporating important variables such as seasonal patterns, economic influences, and other time-dependent factors that affect sales.

In 119
```
df_new['Date'] = pd.to_datetime(df_new['Date'], infer_datetime_format=True)
df_new["Month"] = df_new["Date"].dt.month
df_new["Quarter"] = df_new["Date"].dt.quarter
df_new["Year"] = df_new["Date"].dt.year
df_new["Day"] = df_new["Date"].dt.day
df_new["Week"] = df_new["Date"].dt.isocalendar().week
df_new["Season"] = np.where(df_new["Month"].isin([3,4,5]),"Spring",np.where(df_new["Month"].isin([6,7,8]),"Summer",np.where(df_new["Month"].isin([9,10,11]),"Fall",np
 .where(df_new["Month"].isin([12,1,2]),"Winter","None"))))
print(df_new[["Date","Year","Month","Day","Week","Quarter","Season"]].head())
```
Executed at 2024.01.14 21:37:07 in 940ms

	Date	Year	Month	Day	Week	Quarter	Season
0	2015-07-31	2015	7	31	31	3	Summer
1	2015-07-30	2015	7	30	31	3	Summer
2	2015-07-29	2015	7	29	31	3	Summer
3	2015-07-28	2015	7	28	31	3	Summer
4	2015-07-27	2015	7	27	31	3	Summer

Figure 22: Group by the data into 4 seasons

4.2. Create train , test, validation data

```
In 125  1  x_train, x_test, y_train, y_test = train_test_split(temp,df_new[target],test_size=0.2,random_state=2018)
           Executed at 2024.01.14 21:37:10 in 472ms

In 126  1  #Further divide training dataset into train and validation dataset with an 90:10 split
        2  x_train, x_val, y_train, y_val = train_test_split(x_train, y_train,test_size=0.1,random_state=2018)
           Executed at 2024.01.14 21:37:10 in 366ms

In 127  1  #Check the sizes of all newly created datasets
        2  print("Shape of x_train:",x_train.shape)
        3  print("Shape of x_val:",x_val.shape)
        4  print("Shape of x_test:",x_test.shape)
        5  print("Shape of y_train:",y_train.shape)
        6  print("Shape of y_val:",y_val.shape)
        7  print("Shape of y_test:",y_test.shape)
           Executed at 2024.01.14 21:37:10 in 13ms

           Shape of x_train: (732390, 44)
           Shape of x_val: (81377, 44)
           Shape of x_test: (203442, 44)
           Shape of y_train: (732390, 1)
           Shape of y_val: (81377, 1)
           Shape of y_test: (203442, 1)
```

Figure 23: Output shape of train ,test and validation of data

We will split the dataset into two parts: a training set and a test set with a ratio of 80% and 20% respectively. Additionally, to enhance the model's accuracy, we will further divide the training set into a training set and a validation set, with a ratio of 90% and 10% respectively.

4.3. Metric before data modeling

```
In 128  1  #calculate the average score of the train dataset
        2  mean_sales = y_train.mean()
        3  print("Average Sales :",mean_sales)
           Executed at 2024.01.14 21:37:10 in 47ms

           Average Sales : Sales    5773.099997
           dtype: float64

In 129  1  #Calculate the Mean Absolute Error on the test dataset
        2  print("MAE for Test Data:",abs(y_test - mean_sales).mean()[0])
           Executed at 2024.01.14 21:37:10 in 47ms

           MAE for Test Data: 2883.587604303127
```

Figure 24: Output shape of train ,test and validation of data

4.4. Data modeling process

```
model = Sequential()
model.add(Dense(150,input_dim = 44,activation="relu"))
#The input_dim =44, since the width of the training data=44 (refer data engg section)
model.add(Dense(1,activation = "linear"))
```
Executed at 2024.01.14 21:37:10 in 49ms

```
#Configure the model
model.compile(optimizer='adam',loss="mean_absolute_error", metrics=["mean_absolute_error"])
```
Executed at 2024.01.14 21:37:10 in 29ms

```
#Train the model
model.fit(x_train.values,y_train.values, validation_data=(x_val,y_val),epochs=10,batch_size=64)
```
Executed at 2024.01.14 21:39:43 in 2m 32s 562ms

Figure 25: The process of modeling

```
11444/11444 [==============================] - 23s 2ms/step - loss: 906508.0625 - mean_absolute_error: 640.2001 - val_loss: 864167.9375 -
val_mean_absolute_error: 627.8195
Epoch 14/15
11444/11444 [==============================] - 23s 2ms/step - loss: 889609.5625 - mean_absolute_error: 634.5693 - val_loss: 866429.2500 -
val_mean_absolute_error: 637.5680
Epoch 15/15
11444/11444 [==============================] - 26s 2ms/step - loss: 882059.7500 - mean_absolute_error: 631.2745 - val_loss: 835286.4375 -
val_mean_absolute_error: 622.0830
6358/6358 [==============================] - 8s 1ms/step - loss: 834229.5000 - mean_absolute_error: 618.6281
Metric  loss : 834229.5
Metric  mean_absolute_error : 618.63
```

Figure 26: MAE after training model

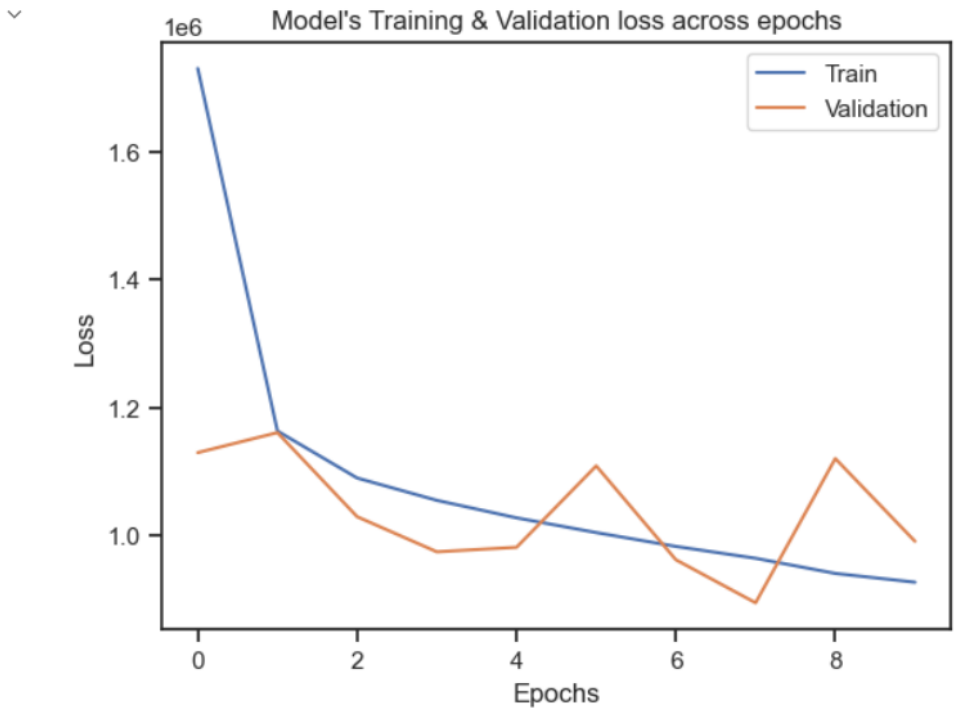


Figure 27: Model's training and validation loss across epochs

About the loss: The training loss values are decreasing over time, starting from 1,735,986.125 and gradually reducing to 943,925.8125. This indicates that the model is improving its performance and reducing the errors between predicted and actual values during the training process. Validation Loss: The validation loss values show some fluctuations but generally decrease from 1,148,203.5 to 897,943.625. This suggests that the model is also performing well on unseen data during the validation process, as the loss values are decreasing.

Add Code Cell | Add Markdown Cell

```
#Manually predicting from the model, instead of using model's evaluate function
y_test["Prediction"] = model.predict(x_test)
y_test.columns = ["Actual Sales","Predicted Sales"]
print(y_test.head(5))
```

Executed at 2024.01.15 13:42:27 in 11s 362ms

```
6358/6358 [==============================] - 7s 1ms/step
        Actual Sales  Predicted Sales
115563             0         0.175753
832654             0         0.175753
769112          2933      3271.041992
350588          8602      7523.621582
141556          6975      6431.605469
```

Figure 28: Actual Sales vs Predicted Sales

It is evident that the model's performance improved after training. The MAE decreased from 2883.5876 to 621.0521, indicating that the model's predictions became more accurate on average. However, it's worth noting that the MSE value after training (832146.3359852177) is still relatively high, suggesting that there is room for further improvement in the model's performance.

Conclusion and Future works

After analyzing the Rossman Sales dataset and training a predictive model, several key findings have emerged. The model's performance was evaluated using various metrics, including mean squared error (MSE) and mean absolute error (MAE).

The initial evaluation of the model on the test data revealed a high MAE of 2883.59, indicating significant differences between the predicted and actual sales values. The model's performance was further assessed after training, where it demonstrated notable improvement.

Post-training, the MAE decreased significantly to 621.05, reflecting a substantial enhancement in the accuracy of sales predictions. However, it is important to note that the MSE after training remained relatively high at 832146.34, indicating the presence of outliers or larger errors in certain predictions.

To achieve more precise and reliable predictions, further analysis and model refinement are necessary. It is advisable to explore additional evaluation metrics and compare the model's performance against other benchmark models or industry standards. Furthermore, feature engineering, hyperparameter tuning, or exploring different algorithms could potentially enhance the model's predictive capabilities.

In conclusion, while the model has shown promising progress in predicting sales for the Rossman dataset, there is still room for improvement. Continued analysis, fine-tuning, and optimization are essential to enhance the model's accuracy and ensure its effectiveness in predicting sales for Rossman stores.

References

Kanade, V. (2022, 8 30). *What Is Machine Learning? Definition, Types, Applications, and Trends for 2022*. Retrieved from https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-ml/

MALI, K. (2023, 11 28). *Everything you need to Know about Linear Regression!* Được truy lục từ https://www.analyticsvidhya.com/blog/2021/10/everything-you-need-to-know-about-linear-regression/

Will Cukierski, F. (2015). *Rossmann Store Sales*. Retrieved from https://kaggle.com/competitions/rossmann-store-sales