Predicting Walmart Sales, Exploratory Data Analysis, and Walmart Sales Dashboard

by

Rashmi Jeswani

A Project Report Submitted
in
Partial Fulfillment of the
Requirements for the Degree of
Master of Science
Supervised by

Michael McQuaid

School of Information

B. Thomas Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, New York

December 2021

The project report “Predicting Walmart Sales, EDA, and Sales Dashboard” by Rashmi Jeswani has been examined and approved by the following Examination Committee:

Michael McQuaid
Senior Lecturer
Project Committee Chair

Stephen Zilora
Professor

Dedication

To my family and Ani who have supported me endlessly throughout my Master's!

Abstract

Predicting Walmart Sales, Exploratory Data Analysis, and Walmart Sales Dashboard

Rashmi Jeswani

Supervising Professor: Michael McQuaid

Data is one of the most essential commodities for any organization in the 21st century. Harnessing data and utilizing it to create effective marketing strategies and making better decisions is extremely essential for organizations. For a conglomerate as big as Walmart, it is necessary to organize and analyze the large volumes of data generated to make sense of existing performance and identify growth potential.The main goal of this project is to understand how different factors affect the sales for this conglomerate and how these findings could be used to create more efficient plans and strategies directed at increasing revenue.

This paper explores the performance of a subset of Walmart stores and forecasts future weekly sales for these stores based on several models including linear and lasso regression, random forest, and gradient boosting. An exploratory data analysis has been performed on the dataset to explore the effects of different factors like holidays, fuel price, and temperature on Walmart's weekly sales. Additionally, a dashboard highlighting information about predicted sales for each of the stores and departments has been created in Power BI and provides an overview of the overall predicted sales.

Through the analysis, it was observed that the gradient boosting model provided the most accurate sales predictions and slight relationships were observed between factors like store size, holidays, unemployment, and weekly sales. Through the implementation of interaction effects, as part of the linear models, relationship between a combination of variables like temperature, CPI, and unemployment was observed and had a direct impact on the sales for Walmart stores.

Contents

List of Tables

List of Figures

1 Introduction

1.1 Project Goals and Background

The 21st century has seen an outburst of data that is being generated as a result of the continuous use of growing technology. Retail giants like Walmart consider this data as their biggest asset as this helps them predict future sales and customers and helps them lay out plans to generate profits and compete with other organizations. Walmart is an American multinational retail corporation that has almost 11,000 stores in over 27 countries, employing over 2.2 million associates (Wikipedia, n.d.).

Catering to their customers with the promise of ‘everyday low prices’, the range of products sold by Walmart draws its yearly revenue to almost 500 billion dollars thus making it extremely crucial for the company to utilize extensive techniques to forecast future sales and consequent profits. The world’s largest company by revenue, Walmart, sells everything from groceries, home furnishings, body care products to electronics, clothing, etc. and generates a large amount of consumer data that it utilizes to predict customer buying patterns, future sales, and promotional plans and creating new and innovative in-store technologies. The employment of modern technological approaches is crucial for the organization to survive in today’s cutting-edge global market and create products and services that distinguish them from its competitors.

The main focus of this research is to predict Walmart’s sales based on the available historic data and identify whether factors like temperature, unemployment, fuel prices, etc affect the weekly sales of particular stores under study. This study also aims to understand whether sales are relatively higher during holidays like Christmas and Thanksgiving than normal days so that stores can work on creating promotional offers that increase sales and generate higher revenue.

Walmart runs several promotional markdown sales throughout the year on days immediately following the prominent holidays in the United States; it becomes crucial for the organization to determine the impact of these promotional offerings on weekly sales to drive resources towards such key strategic initiatives. It is also essential for Walmart to understand user requirements and user buying patterns to create higher customer retention, increasing their demand adding to their profits. The findings from this study can help the organization understand market conditions at various times of the year and allocate resources according to regional demand and profitability.

Additionally, the application of big data analytics will help analyze past data efficiently to generate insights and observations and help identify stores that might be at risk, help predict as well as increase future sales and profits and evaluate if the organization is on the right track.

The analysis for this study has been done using SQL, R, Python, and Power BI on the dataset provided by Walmart Recruiting on Kaggle (“Walmart Recruiting - Store Sales Forecasting,” 2014). The modeling, as well as the exploratory data analysis for the research, have been performed in R and Python, aggregation and querying will be performed using SQL and the final dashboard has been created using Power BI.

1.2 Project Deliverables

The main objective of this study is to predict weekly sales for Walmart stores and create a Power BI dashboard that tracks the final predicted sales until 2013 through interactive and immersive visualizations.

The conclusion section highlights the findings from the exploratory data analysis as well as from the models implemented as part of this study.

The dashboard created compares the findings from the Exploratory Data Analysis with the findings from the dashboard. The user of the dashboard can filter data based on stores, departments, store type, store size, week of the year, holiday versus non-holiday sales, etc.

1.3 Tools and Technologies Applied

The analysis for this study have been performed using some main tools: R, Python, and Power BI. The models and Exploratory Data Analysis have been executed using development tools like R Studio and PyCharm.

Several packages have been used to perform the initial and final outcome EDA for the analysis. For the initial EDA, a combination of R and Python libraries like inspectdf, ggplot2, plotly, caret, matplotlib, seaborn, etc have been implemented. Packages like numpy, pandas, tidyverse, etc. have been used for data wrangling and manipulation. For the models that have been created, several packages like ‘scikit-learn’, ‘xgboost’, etc have been applied.

2 Purpose Statement

The purpose of this study is to predict the weekly sales for Walmart based on available historical data (collected between 2010 to 2013) from 45 stores located in different regions around the country. Each store contains a number of departments and the main deliverable is to predict the weekly sales for all such departments.

The data has been collected from Kaggle and contains the weekly sales for 45 stores, the size and type of store, department information for each of those stores, the amount of weekly sales, and whether the week is a holiday week or not. There is additional information in the dataset about the factors that might influence the sales of a particular week. Factors like Consumer Price Index (CPI), temperature, fuel price, promotional markdowns for the week, and unemployment rate have been recorded for each week to try and understand if there is a correlation between the sales of each week and their determinant factors.

Correlation testing has been performed to understand if there is a correlation between the individual factors and weekly sales and whether such factors have any impact on sales made by Walmart.

This study also includes an extensive exploratory data analysis on the provided Walmart dataset to understand the following:

- Identifying store as well as department-wide sales in Walmart
- Identifying sales based on store size and type
- Identifying how much sales increase during holidays
- Correlation between the different factors that affect sales
- Average sales per year
- Weekly sales as per region temperature, CPI, fuel price, unemployment

Apart from identifying these direct relationships between independent and dependent variables, some interaction effects have also been studied as part of the Multiple Linear Regression model to understand if a certain combination of the factors under study can directly impact the weekly sales for Walmart.

After employing different algorithms to predict future sales and correlation between factors for the retail store, a dashboard that tracks the above-mentioned outcomes has been created (in Power BI) and also includes the new predictions to collectively visualize the outcomes of this research and present them to amateur users more effectively.

3 Related Work

Studies have previously been performed to predict sales for retail industry corporations based on the availability of relevant historic data. Several authors from the Fiji National University and The University of the South Pacific analyzed the Walmart dataset to predict sales ("Walmart's Sales Data Analysis - A Big Data Analytics Perspective," 2017). Tools like Hadoop Distributed File Systems (HDFS), Hadoop MapReduce framework, and Apache Spark along with Scala, Java, and Python high-level programming environments were used to analyze and visualize the data. Their study also aimed at trying to understand whether the factors included in the dataset have any impact on the sales of Walmart.

In 2015, Harsoor and Patil (Harsoor & Patil, 2015) worked on forecasting Sales of Walmart Stores using big data applications: Hadoop, MapReduce, and Hive so that resources are managed efficiently. This paper used the same sales data set that has been used for analysis in this study, however, they forecasted the sales for the upcoming 39 weeks using Holt's winter algorithm. The forecasted sales are visually represented in Tableau using bubble charts.

Michael Crown (Crown, 2016), a data scientist, performed analysis on a similar dataset but instead focused on the usage of time series forecasting and non-seasonal ARIMA models to make his predictions. He worked on ARIMA modeling to create one year of weekly forecasts using 2.75 years of sales data, with features of the store, department, date, weekly sales, and holiday data. Performance was measured using normalized root-mean-square error (NRMSE).

Forecasting has not been limited to just business enhancement. Several researchers have tried to utilize machine learning and statistical analysis to build predictive models that can accurately predict the weather, monitor stock prices and analyze market trends, predict illnesses in a patient, etc.

Likewise, in 2017, Chouskey and Chauhan (Chouksey & Chauhan, 2017) worked on creating a weather forecasting model that accurately predicts the weather and sends out weather warnings for people and businesses so that they can better prepare for the unforeseeable weather. The authors make use of MapReduce and Spark to create their models and gather data from various weather sensors; weather forecasts can be essentially important as they influence all human aspects and the authors have made use of various parameters like temperature, humidity, pressure, wind speed, etc. to make better predictions.

Another approach followed by Rajat Panchotia (Panchotia, 2020) to create a predictive model using linear regression throws light on the various regression techniques and the metrics that need to be defined when creating such models. He talks about the importance of defining techniques that should be considered, like studying the number of independent variables and type of dependent variables, determining the best fit, etc., based on the nature of data and the most accurate regression model that should be selected based on results obtained. In his article, he also emphasizes on the use of regression coefficients, p-values, variable selection, and residual analysis to study the performance of regression models. While Panchotia only focuses on studying the direct relationship between the independent and dependent variables of the dataset, another theory by James Jaccard and Robert Turrisi (Jaccard & Turrisi, 2018) involves observing the change in the relationship between an independent and dependent variable as a result of the presence of a third variable, called the moderator variable.

Kassambara (kassambara, 2018), in his article, throws light on the implementation of interaction effects with a multiple linear regression in R. Taking a basic multiple regression model as a base where he tries to predict sales based on advertising budgets spent on youtube and facebook, he tries to create an additive model based on two relevant predictors (budget for youtube and budget for facebook). His model assumes that the effect on sales of youtube advertising is independent of the effect of facebook advertising and subsequently creates a regression model. With an R2 score of 0.98, he observes that there is an interactive relationship between the two predictor variables (youtube and facebook advertising) and this additive model performs better than the regular regression model.

Adopting a similar approach to Kassambara (kassambara, 2018), this study also studies the interaction effects between the multiple independent variables in the dataset like unemployment, fuel prices, CPI, etc. and tries to find if there is a relationship between a combination of these factors and weekly sales.

A further extension of predictive techniques relevant to this study involves the implementation of random forest algorithms to create predictive models. A study conducted by researchers at the San Diego State University (Lingjun et al., 2018) highlights the importance of this tree-based machine learning algorithm over other regression methods to create predictive models for the higher education sector. With their study, the authors use a standard classification and regression tree (CART) algorithm along with feature importance to highlight the importance of using random forest algorithms with prediction problems in Weka and R and compare their efficacy with several other models like lasso regression and logistic regression.

The purpose of this review was to identify similar practices utilized by other authors or researchers when creating a predictive model influenced by several independent variables. It is clear from the research reviewed above that a significant number of these authors make use of a combination of tools and techniques to create efficient models and tend to compare their findings across all models to select the best-performing model based on their dataset. Just like Harsoor and Patil, along with Chouskey and Chauhan, make use of Hadoop, MapReduce, and Hive to generate predictions, this study makes use of several algorithms like linear and lasso regression, random forest, etc. and also studies interaction effects for multiple regression to make predictions.

Performing a comparative analysis with several models is crucial to ensure that the predictions are accurate and that they are not limited in scope. Testing out multiple models is also necessary for this study as models tend to perform differently based on the nature and size of the data.

4 Methodology

The paper comprises of several different components that explore various aspects of the 45 Walmart stores used in this study. The methodology section is broken down into several sub-sections that follow a ‘top-down’ approach of the process that is followed in this analysis.

This section contains detailed information about the dataset, the exact techniques that have been used in forecasting weekly sales and the last section talks about how this study is significant in predicting the weekly sales for Walmart stores. It will also discuss the success of the applied models in identifying the effect of different factors on such weekly sales.

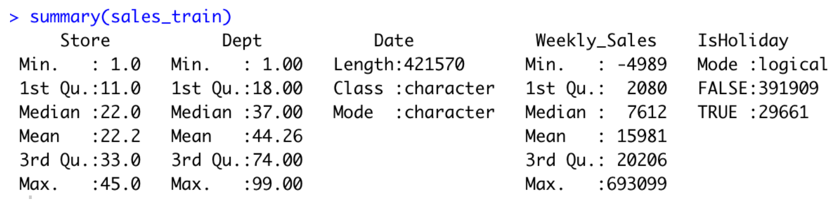
4.1 About the Dataset

The dataset for this study has been acquired from a past Kaggle competition hosted by Walmart, this can be found here: https://www.kaggle.com/c/walmart-recruiting-store-sales-forecasting/data. It contains historic weekly sales information about 45 Walmart stores across different regions in the country along with department-wide information for these stores.

The 'test.csv' data file that is a part of this dataset is only being used to predict values derived from the model with the lowest WMAE score. Because, the dataset contains no target variable, in our case 'Weekly Sales', it cannot be used for testing for this analysis. Instead, the training dataset 'train.csv' is being split into training and validation datasets for this study.

The main goal of this study is going to be to predict the department-wide weekly sales for each of these stores.

The dataset is already divided into separate training and testing data; the testing data is identical to the training dataset apart from the weekly sales information. The training dataset contains weekly sales information from 2010-02-05 to 2012-11-01 about the stores and departments. It also contains a column that suggests whether a particular date falls on a holiday or not. In total, there are 4,21,570 rows in the training dataset and 1,15,064 rows in the testing dataset. (Figure 1)



```
> summary(sales_train)
     Store           Dept            Date            Weekly_Sales     IsHoliday
 Min.   : 1.0   Min.   : 1.00   Length:421570      Min.   : -4989   Mode :logical
 1st Qu.:11.0   1st Qu.:18.00   Class :character   1st Qu.:  2080   FALSE:391909
 Median :22.0   Median :37.00   Mode  :character   Median :  7612   TRUE :29661
 Mean   :22.2   Mean   :44.26                      Mean   : 15981
 3rd Qu.:33.0   3rd Qu.:74.00                      3rd Qu.: 20206
 Max.   :45.0   Max.   :99.00                      Max.   :693099
```

Figure 1. A summary of the Training dataset

There is another dataset called 'stores.csv' that contains some more detailed information about the type and size of these 45 stores used in this study.

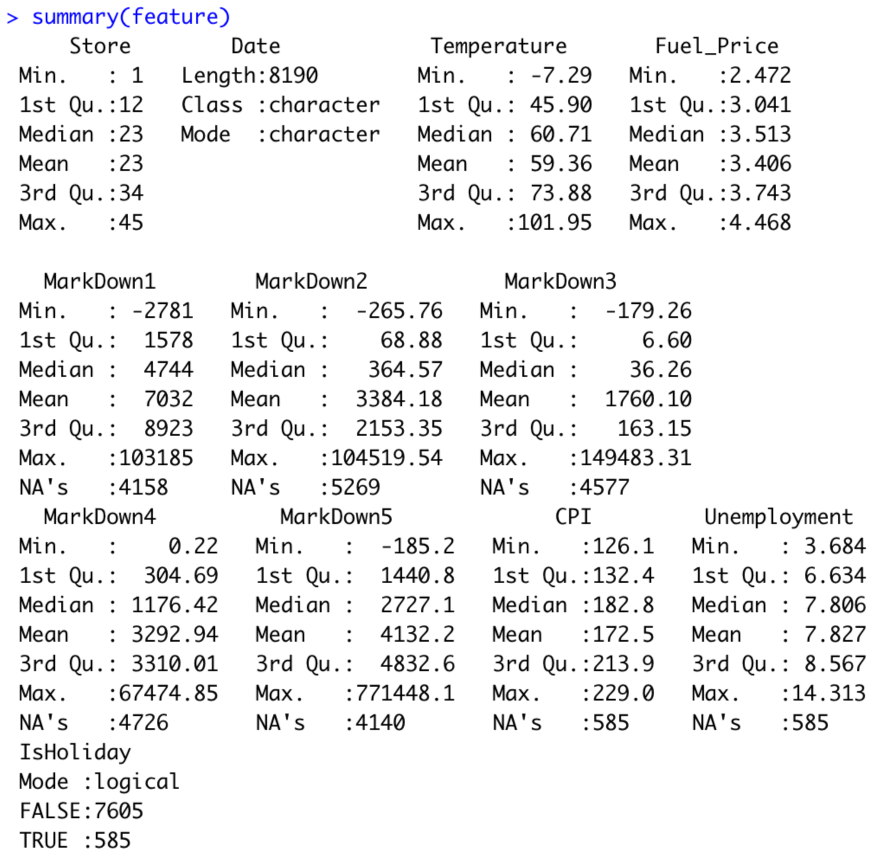
Another big aspect of this study is to determine whether there is an increase in the weekly store sales because of changes in temperature, fuel prices, holidays, markdowns, unemployment rate, and fluctuations in consumer price indexes, The file 'features.csv' contains all necessary information about these factors and is used in the analysis to study their impact on sale performances.

The holiday information listed in the study is:

Table 1. List of holidays from the dataset

Holiday Name	Date 1	Date 2	Date 3	Date 4
Super Bowl	12-Feb-10	11-Feb-11	10-Feb-12	8-Feb-13
Labor Day	10-Sep-10	9-Sep-11	7-Sep-12	6-Sep-13
Thanksgiving	26-Nov-10	25-Nov-11	23-Nov-12	29-Nov-13
Christmas	31-Dec-10	30-Dec-11	28-Dec-12	27-Dec-13

A summary of the features dataset is displayed in the image below. (Figure 2)



```
> summary(feature)
     Store          Date            Temperature       Fuel_Price   
 Min.   : 1   Length:8190        Min.   : -7.29   Min.   :2.472  
 1st Qu.:12   Class :character   1st Qu.: 45.90   1st Qu.:3.041  
 Median :23   Mode  :character   Median : 60.71   Median :3.513  
 Mean   :23                      Mean   : 59.36   Mean   :3.406  
 3rd Qu.:34                      3rd Qu.: 73.88   3rd Qu.:3.743  
 Max.   :45                      Max.   :101.95   Max.   :4.468  
                                                                 
   MarkDown1        MarkDown2           MarkDown3        
 Min.   : -2781   Min.   :  -265.76   Min.   :  -179.26  
 1st Qu.:  1578   1st Qu.:    68.88   1st Qu.:     6.60  
 Median :  4744   Median :   364.57   Median :    36.26  
 Mean   :  7032   Mean   :  3384.18   Mean   :  1760.10  
 3rd Qu.:  8923   3rd Qu.:  2153.35   3rd Qu.:   163.15  
 Max.   :103185   Max.   :104519.54   Max.   :149483.31  
 NA's   :4158     NA's   :5269        NA's   :4577       
   MarkDown4          MarkDown5             CPI          Unemployment   
 Min.   :    0.22   Min.   :  -185.2   Min.   :126.1   Min.   : 3.684  
 1st Qu.:  304.69   1st Qu.:  1440.8   1st Qu.:132.4   1st Qu.: 6.634  
 Median : 1176.42   Median :  2727.1   Median :182.8   Median : 7.806  
 Mean   : 3292.94   Mean   :  4132.2   Mean   :172.5   Mean   : 7.827  
 3rd Qu.: 3310.01   3rd Qu.:  4832.6   3rd Qu.:213.9   3rd Qu.: 8.567  
 Max.   :67474.85   Max.   :771448.1   Max.   :229.0   Max.   :14.313  
 NA's   :4726       NA's   :4140       NA's   :585     NA's   :585     
 IsHoliday      
 Mode :logical  
 FALSE:7605     
 TRUE :585      
```

Figure 2. A summary of the Features dataset

The final file called ‘sampleSubmission.csv’ contains two main columns: dates for each of the weeks in the study as well as a blank column that should be utilized to record predicted sales for that week based on the different models and techniques applied.

The results of the most accurate and efficient model have been recorded in this file and the final Power BI dashboard has been created based on these predicted values, in conformity with the ‘stores’ and ‘features’ dataset.

4.2 Exploratory Data Analysis

It is crucial to have an in-depth understanding of the dataset that is used in this analysis to understand the models that would give the most accurate prediction. Several times there are underlying patterns or trends in the data that would not be identified as easily, hence the need for an extensive exploratory data analysis. This thorough examination is necessary to understand the underlying structure of the dataset and to draw conclusions or insight about the validity of our analysis.

The study is going to begin with a brief analysis of the available dataset to get a sense of the main characteristics and components that are relevant to the research. An exploratory data analysis is crucial to this study considering the numerous attributes that are a part of the dataset that will be essential when trying to draw insights and making predictions. As part of the exploratory data analysis, several visualizations have been created that will help us understand what it is that we are trying to achieve and to keep in mind the various attributes that we can use to improve results.

The EDA is like a primary investigation and tries to look at the relationships and nature of the different columns available to us. As part of this, the ‘inspectdf’ package (Ellis, 2019) and the ‘glimpse’ package (Sullivan, 2019) have been used and implemented in R that will answer questions related to the number and nature of columns and rows in the dataset, missing values, distribution of numeric and categorical variables, correlation coefficients, etc.

Several other packages like ‘ggplot2’, ‘matplotlib’, ‘seaborn’, and ‘plotly’ have also been used in this study to create visualizations that provide information about weekly sales by store and department, weekly sales on holidays versus on normal days, weekly sales based on region, store type and store size, average sales per year, change in sales as a result of factors like CPI, fuel price, temperature, and unemployment, etc in the form of heatmaps, correlation matrix (Kedia et al., 2013), histograms, scatterplots and several more. These visualizations are accompanied by brief descriptions that will discuss the findings and scope for potential modeling that will be performed in the next stages of this project.

4.2.1 EDA I: Exploring the Dataset with ‘inspectdf’

The ‘inspectdf’ package in R does basically what the name suggests: it inspects crucial components of a dataframe under study. For this study, it was essential to get a sense of the several datasets before they were used to create complex models.

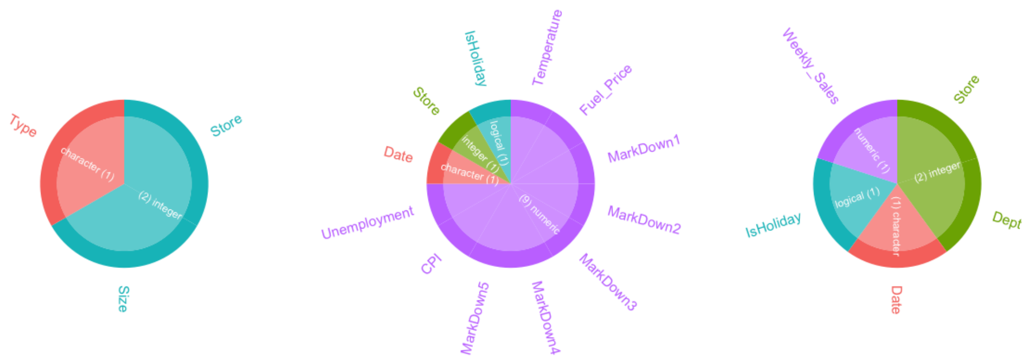
This package inspects, compares, and visualizes the different components associated with the above-mentioned datasets. It gives a brief visualized column-wise summary about missing or out of range values, distribution of values, types of columns, the correlation between numeric columns, etc. Starting with the initial explorations, it is imperative to understand the data types and ranges of the values in each column; the ‘inspecttypes()’ function from the package helps explore the data type for columns in the dataset.

The dataset contains the following five main CSV files: train, test, features, stores, and sampleSubmission. Each file contains crucial information relevant to this study, some of the important ones are discussed below:

Table 2. Description of Columns

Column Name	Column Type	Column Description
Store	Categorical	45 stores each with 143 observations
Department	Categorical	99 departments each
Date	Categorical	Weekly Sales data from 2010 until 2012
Weekly Sales	Numerical Continuous	Sales Ranging from 2,00,000 to 38,00,000
IsHoliday	Categorical Binary	0 and 1 values associated with date
Fuel Price	Numerical Continuous	Prices ranging from 2.4 to 4.4
CPI	Numerical Continuous	Values ranging from 126 to 227
Unemployment	Numerical Continuous	Values ranging from 3 to 14

A visualization of the ’inspect df()’ package for the features, train, and stores datasets can also be observed in Figure 3.



(a) Data Type: Stores Dataset (b) Data Type: Features Dataset (c) Data Type: Training Dataset

Figure 3. Data Types

Another important function from the package is the ‘inspect na()’ function that highlights the missing values in each column of the dataset used for this study. Apart from the ‘features.csv’ file, it was found that no other dataset used had any missing values in any of the columns. The major missing values in the features dataset come from the markdown columns that contain information about the different promotional activities happening at different Walmart stores. A reason behind such a massive amount of missing values in these columns is due to the seasonal promotional prices set by the stores during holidays (that mostly happen to start from November until January).

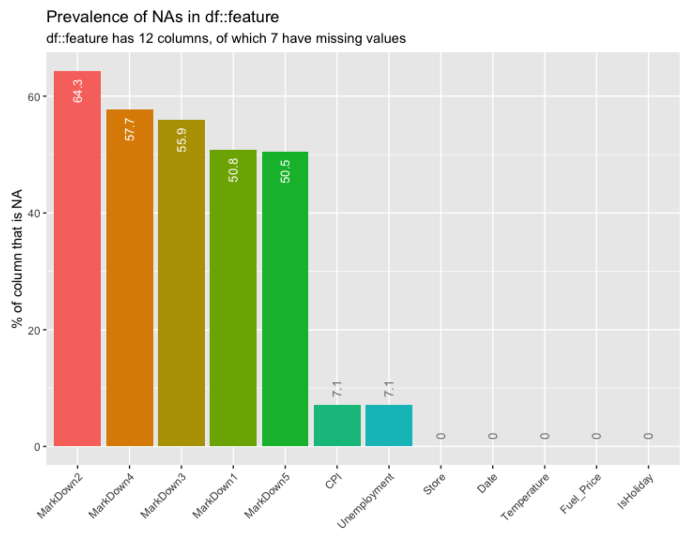


Figure 4. Missing Values in the Features dataset

The next function from the package looks at the distribution of the numeric variables using histograms created from identical bins. Considering that the features dataset has the most numeric variables, that is the only one that will be looked at in detail. According to the package website, 'The hist column is a list whose elements are tibbles each containing the relative frequencies of bins for each feature. These tibbles are used to generate the histograms when showplot = 'TRUE'.

The histograms are represented through heat plot comparisons using Fisher's exact test to highlight the significance of values within the column; the higher the significance, the redder the data label (Rushworth, n.d.).

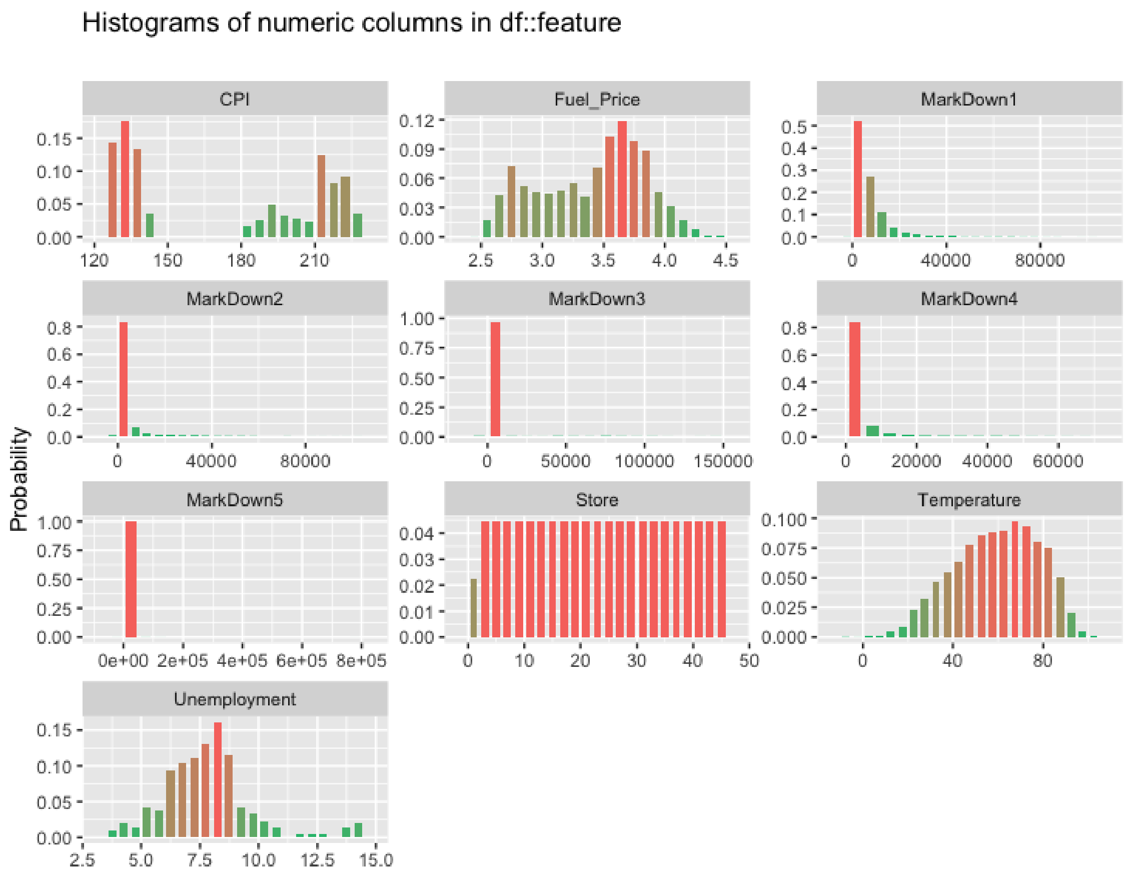


Figure 5. Distribution of Numerical attributes in the Features dataset

After looking at the distribution of numeric variables, the study proceeds to look at the distribution of categorical variables. This is done using ‘inspect cat()’; this function looks at the complete distribution of values in the categorical columns. Considering there are not many categorical variables in the datasets used for this study (as seen above through ’inspect types()’), the only relevant information that was gathered using the function was the distribution of types of stores in the ‘stores’ dataset.

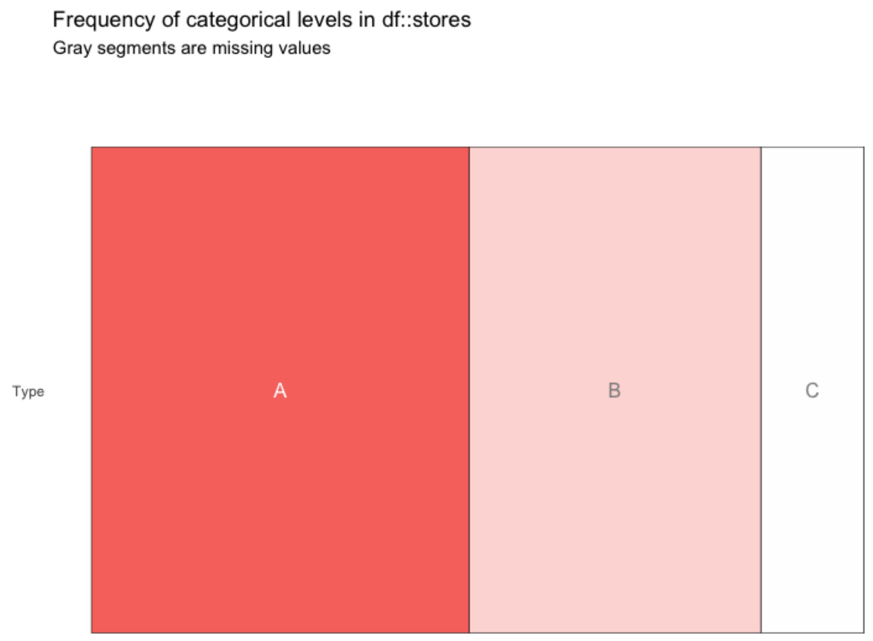


Figure 6. Distribution of Store Types

From the image above, it is clear that a majority of the Walmart stores included in this study belong to Type 'A'. This will be briefly discussed in the coming sections of the study when advanced EDA will be used to answer some important questions.

The last significant function from the 'inspectdf()' package is called 'inspect cor()'. This function, in a nutshell, enables users to look at Pearson's correlation between different variables in a dataframe. Understanding if there is an association between variables beforehand will answer a lot of questions about what variables affect the weekly sales the most significantly.

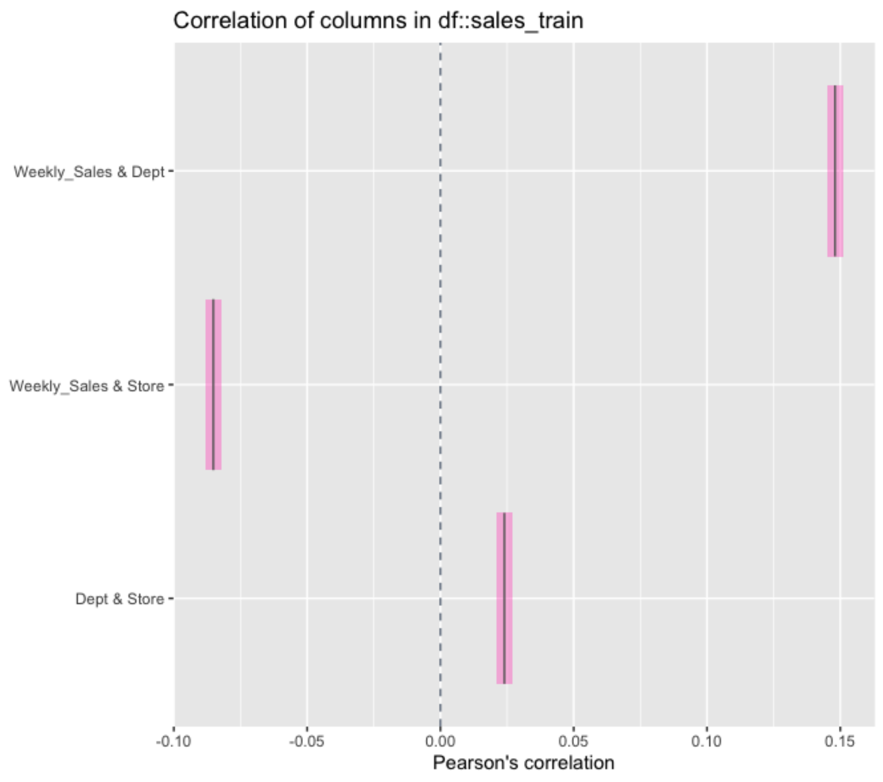


Figure 7. Correlation between attributes of the training dataset

A look at the correlation between the variables of the training dataset depicts that while there is a slight association between the weekly sales and department, it is still not as significant (higher the R-value, higher the significance).

4.2.2 EDA II

The second section under this Exploratory Data Analysis looks at advanced and extensive visualizations that answer some crucial questions about the Walmart dataset, as listed in the purpose statement.

After inspecting crucial elements in each of the data frames about the types of variables, their distribution, correlation, and association, etc. using 'inspectdf', more detailed and summarized information about the weekly sales for each department/store and the effect of various factors on the weekly sales are studied here. This is performed using a combination of R and Python packages like 'ggplot2', 'matplotlib', 'seaborn', 'plotly', and several others.

This section will aim at looking at the following aspects of the Walmart dataset and also possibly look at some more crucial information that stems out from the below-mentioned criteria:

- Identifying store as well as department-wide sales in Walmart
- Identifying sales based on store size and type
- Identifying how much sales increase during holidays
- Correlation between the different factors that affect sales
- Average sales per year
- Weekly sales as per region temperature, CPI, fuel price, unemployment

4.2.3 Identifying Storewide Sales

As depicted in Figure 8, we can conclude that Store 'A' is the most popular type of store included in this study. Apart from the number of stores, it is also essential to identify the average sale for each of the Walmart stores. For this, I have used the 'plotly express' library in Python and grouped the training datasets (that have previously been merged with the features dataset) based on Store Type and calculated average sales.

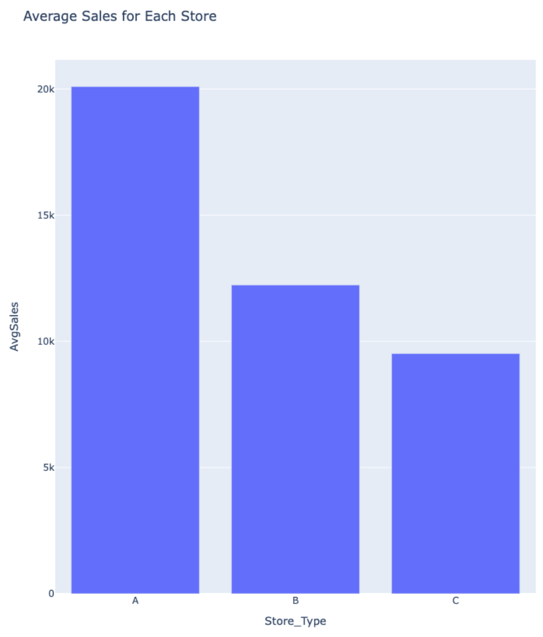


Figure 8. Average Sales for Each Store Type

Based on our bar graph, it is clear that Type 'A' stores have the highest sales compared to the other two stores. This tells us that there is a direct relationship between the size of the store and their corresponding sales. Type ‘B’ stores have the second-highest number of stores as well as average sales, thus proving this assumption.

After looking at the average sales for each store type, the next focus is to look at the sales for each department associated with the stores.

4.2.4 Identifying Department-wide Sales

The next step is looking at what departments for each of these stores have the highest average sales. From the image depicted below, it can be identified that Department number 92 has the highest number of average sales.

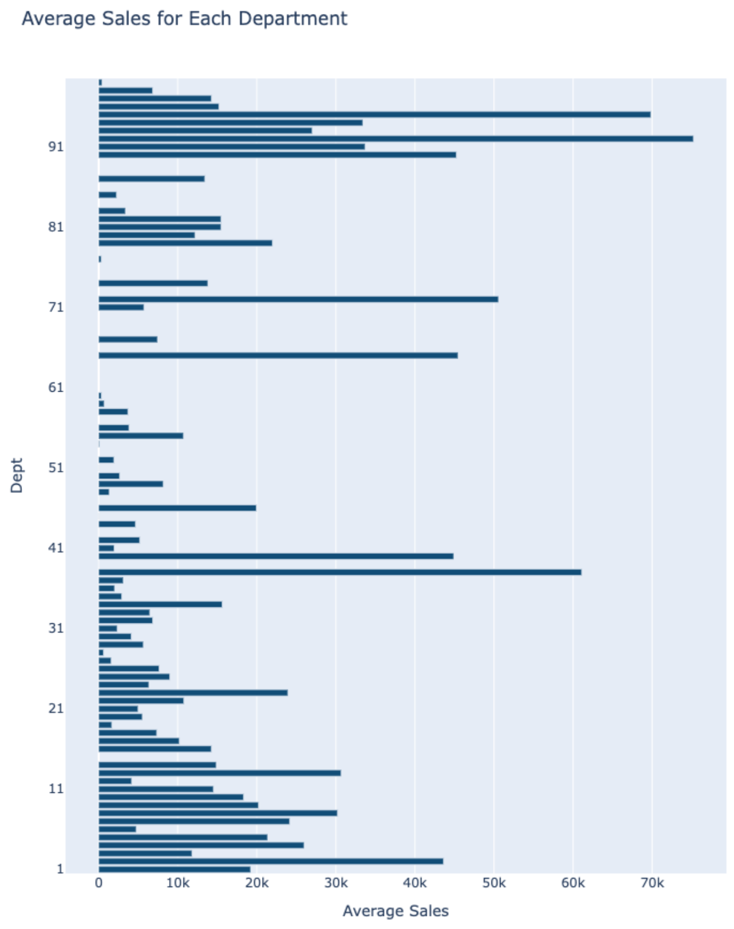


Figure 9. Department Wide Sales

After looking at the average sales for the stores and departments, it is also imperative to look at the breakdown of sales for both the stores as well as departments. This breakdown will include looking at average sales for each year, week over week sales, monthly sales, week of year sales, etc. Each of these will throw more light on customers' buying patterns over different periods.

It will also help in evaluating whether sales go up during certain time periods as compared to normal average sales. I will also look at the average sales for different stores associated with the three store types and evaluate which store number has the highest sales.

4.2.5 Identifying Average Store Sales

There are several stores associated with the three store types and it is crucial to look at the average sales for each of these stores. In total there are 45 stores presented in the Walmart dataset and from the image below it can be concluded that store numbers 4, 14, and 20 have the highest average sales. It should also be noted that there is a very high difference between the average sales for each of the stores; while some stores record huge sales, some others lack vastly in the area.

This could be dependent on factors like the kinds of products sold by the store, the geographic location, temperature, unemployment in the vicinity, etc. Some further study reveals that all three of these stores belong to the Store Type A that gather the largest sales out of all three store types.

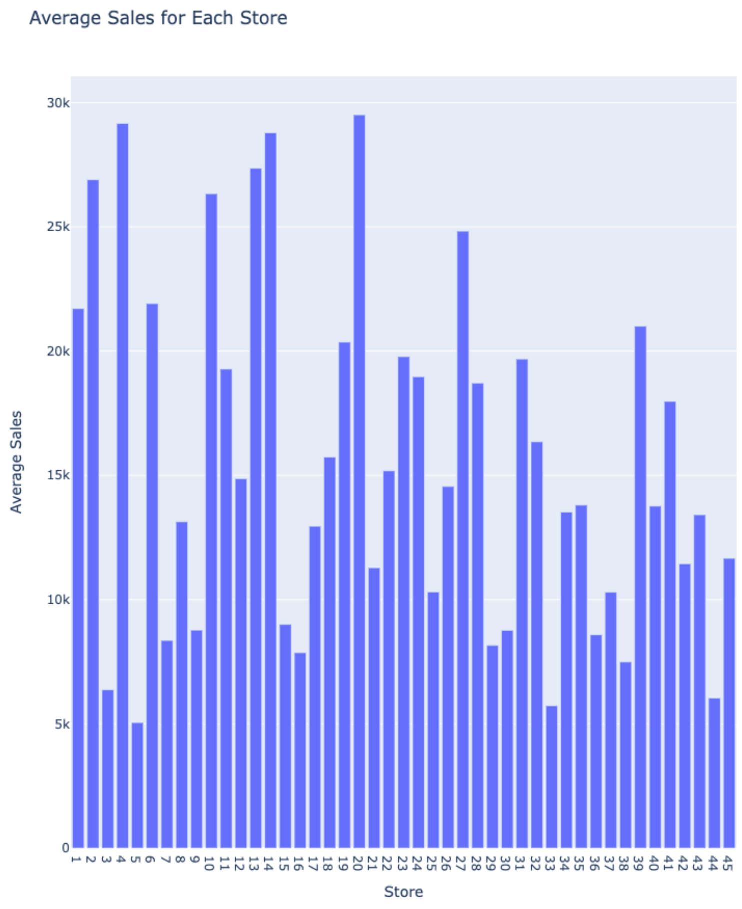


Figure 10. Store Wide Sales

4.2.6 Identifying Specific Stores and Departments with Highest Sales

After looking at specific stores and departments with the highest sales, it is also imperative to analyze whether these specific stores and departments with the highest sales belong to a specific store type. It is necessary to highlight whether specific stores and their respective departments generate higher sales for Walmart.

We know that Type A stores tend to generate the most revenue for Walmart, hence, it is important to look at the composition of such stores to identify high revenue generating departments.

I have created two histograms for the same, one that visualizes the highest revenue generating departments for each store type and the second that breaks down the store and store type information for four of the highest revenue generating departments. The following visuals for this section have been generated in Power BI after combining the training dataset with the stores dataset.

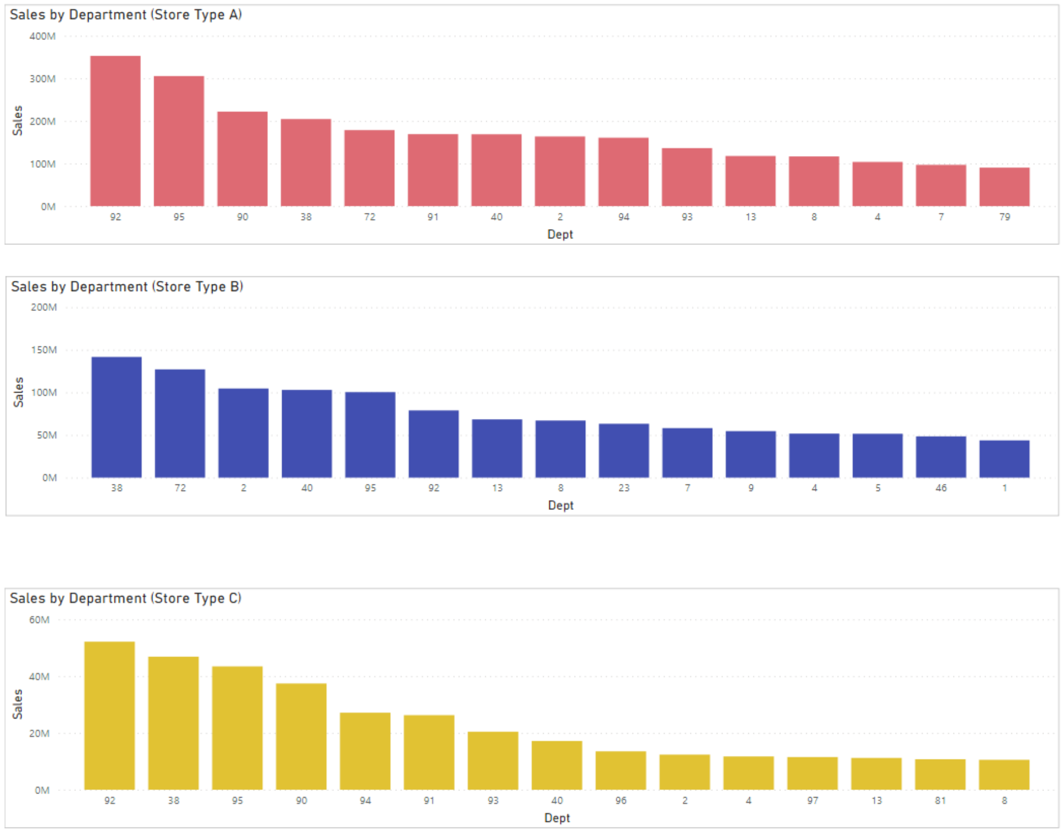


Figure 11. Departments with highest sales for each store type

The histogram generated in Figure 11 highlights the top 15 departments with the highest revenue for each of the three store types. As observed in the figure, it is certain that departments 92, 95, 38, 72 and 90 are the top revenue generating departments across all the three store types. Although there is no information in this dataset about the nature of these departments, it is still safe to establish that Walmart can increase revenue by including the above mentioned departments in their stores.

The figure below (Figure 12) consists of a breakdown of stores and their respective types for the four largest departments.

In conclusion, for all store types listed in this study, departments 95, 92, 38, 90, and 72 generate some of the highest revenue for Walmart stores.

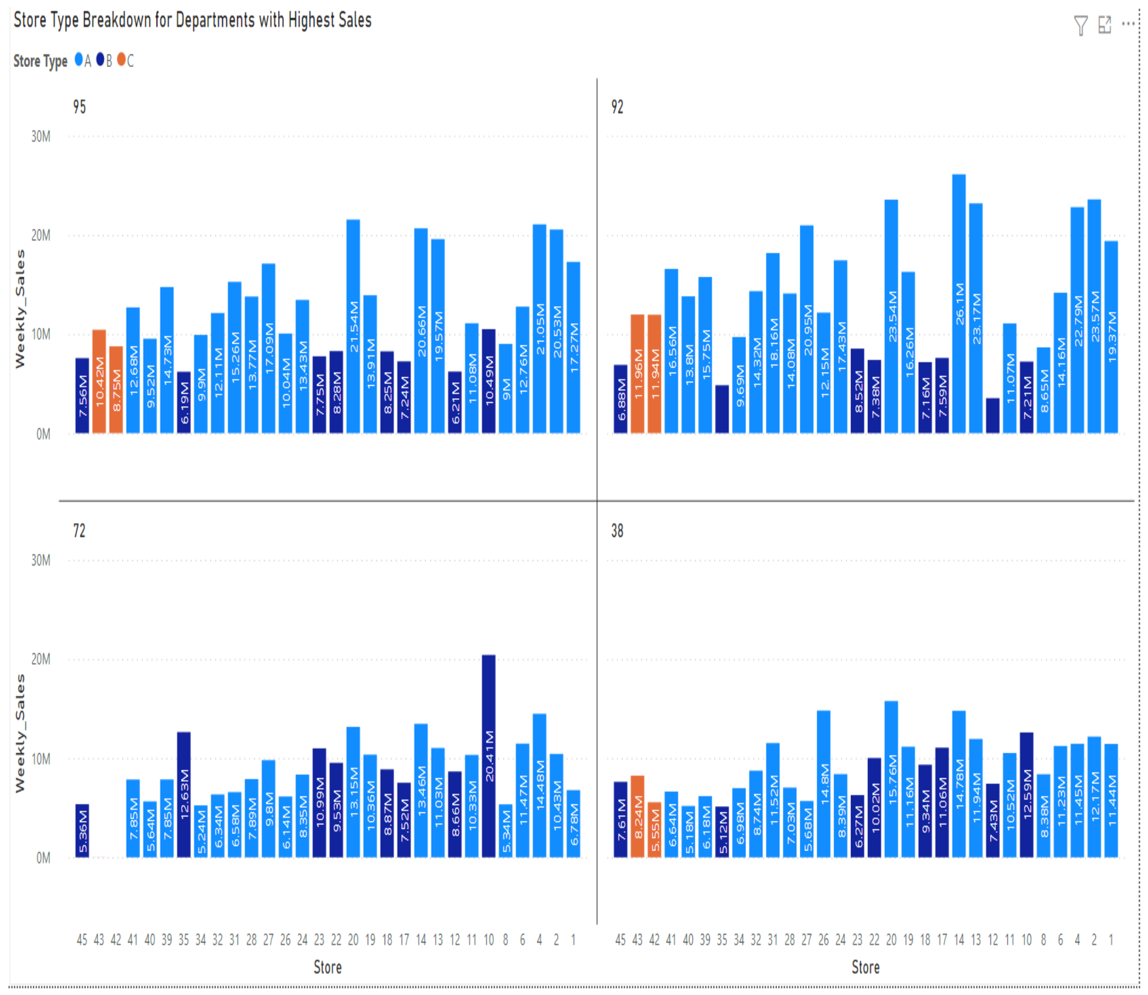


Figure 12. Stores and departments with highest sales for each store type

4.2.7 Identifying Monthly Sales for Each Year

With the holiday information provided in the original dataset, it is known that the major holidays fall at the end of the year.

The graph below clearly depicts that the months of November and December recorded the highest average sales for 2010 and 2011. The dataset provided by Walmart contained no weekly sales information for the last two months of the year 2012, hence no conclusion can be drawn for that year. This graph also shows that the month of January tends to have the lowest average sales in the whole year.

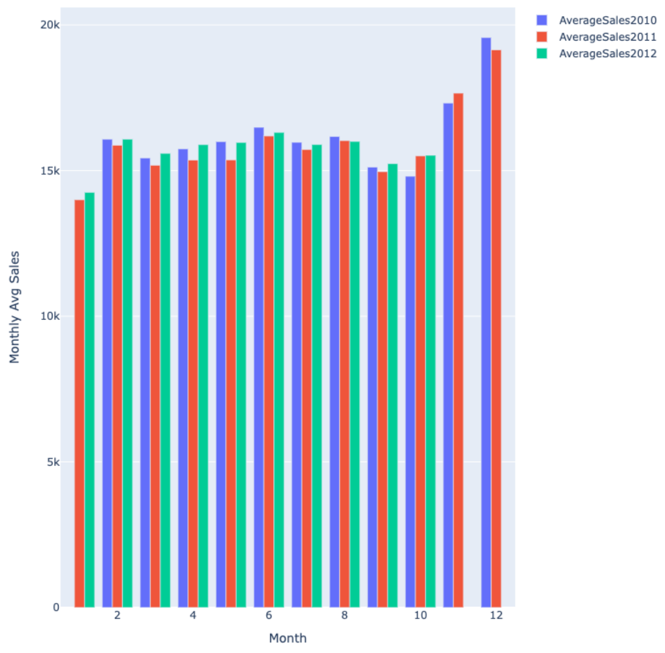


Figure 13. Overall Monthly Sales

4.2.8 Identifying Week Over Week Sales for Each Year

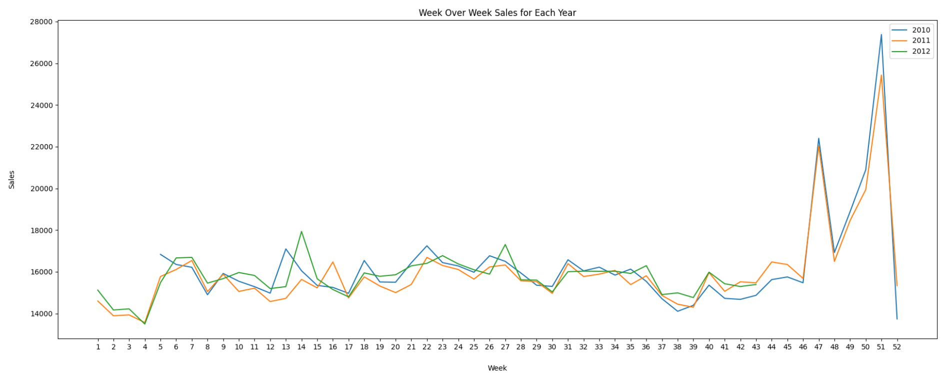


Figure 14. Week Over Week Sales

The week over week overview again helps us in understanding if there is an increase in sales during holiday weeks each year, i.e. the weeks of Thanksgiving, Christmas, Labor

Day, etc.

There is an evident hike in sales in weeks 47 and 51 that correspond to Thanksgiving and Christmas respectively, proving again that sales rise during the holiday season. Due to the insufficiency of data for the year 2012, these conclusions have only been made based on the data available from 2010 and 2011. This graph also tells that there is a distinguished pattern of decline immediately following Christmas and New Year's.

After studying the overall sales summaries of different components of the Walmart dataset, this report will now throw light upon the effect of different factors (such as holidays, markdowns, CPIs, unemployment, etc.) on the weekly sales. It has always been an integral part of this study to understand the effect that these factors have on Walmart's sales performance. I will also create several visualizations that shed light on the difference in Walmart store sales on holidays versus non-holiday days, the impact of store size and type on weekly sales, and finally create a correlation matrix to examine the correlation between the many factors included in the study.

4.2.9 Impact of Size of Store on Sales

It is known from the previous visualization that Type 'A' is the biggest store type followed by Type 'B' and 'C'. The graph below shows a linear relationship between the size of a store and their consequent sales, with some exceptions. A few Type B stores, as depicted below, acquire more average sales than Type A stores, going against the general idea of the bigger the size of the store, the higher the sales. But in general, type A stores still show a high amount of sales while Type C stores show a significantly small amount of sales. To summarize, sales generally increase with an increase in the size of the store, with some minor exceptions.

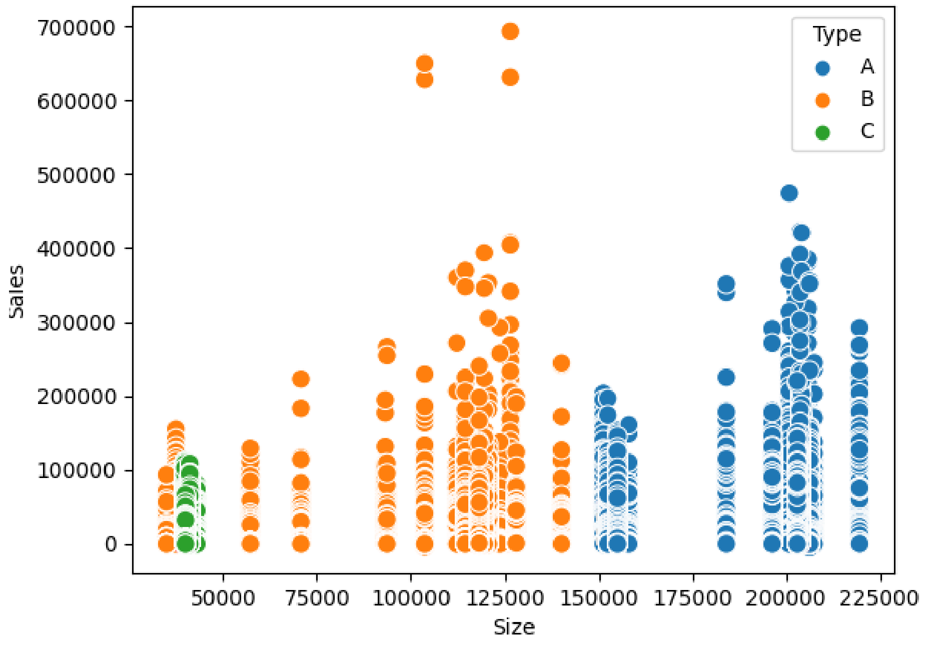


Figure 15. Impact of Size of Store on Sales

4.2.10 Week of Year Sales by Store Type

This visualization is similar to Figure 14 in the way that it shows a summarized view of the average sales for each week of a year. The difference with this scatter plot is that it shows the average weekly sales for each store type and helps understand if sales go up for each store at the end of the year. With this plot, it is clear to understand that, unlike Store Type A and B, the average sales do not necessarily go up for Type C at the end of the year around Thanksgiving and Christmas. It also shows that Type A stores typically have higher weekly sales as compared to the other two stores, proving once again that a bigger store size signifies higher sales.

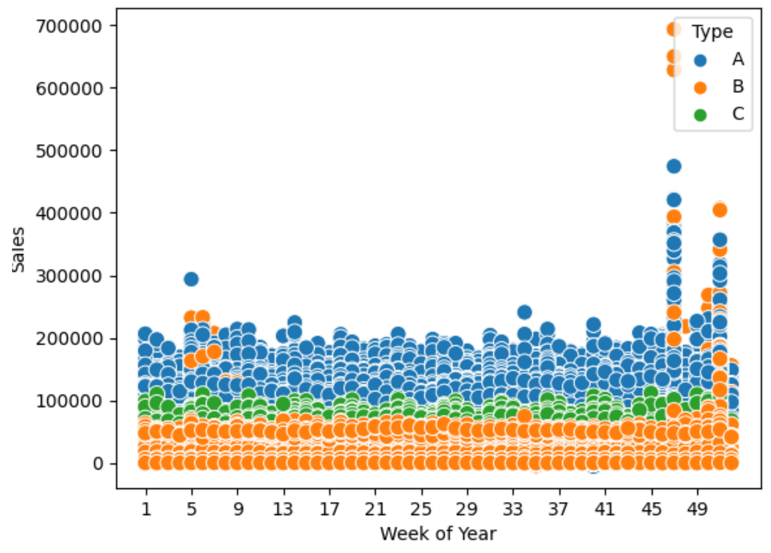


Figure 16. Week of Year Sales Based on Store Type

4.2.11 Impact of Temperature on Sales

It has widely been known in the retail sector that weather has a profound effect on sales. While warmer weather promotes sales, cold/harsh or extremely hot weather is generally not a great encouragement for shoppers to get outdoors and spend money. Generally speaking, temperatures between 40 to 70 degrees Fahrenheit are considered as favorable for humans to live in considering they are not as hot or cold.

As seen below, the highest sales occur for most store types between the range of 40 to 80 degrees Fahrenheit, thus proving the idea that pleasant weather encourages higher sales. Sales are relatively lower for very low and very high temperatures but seem to be adequately high for favorable climate conditions.

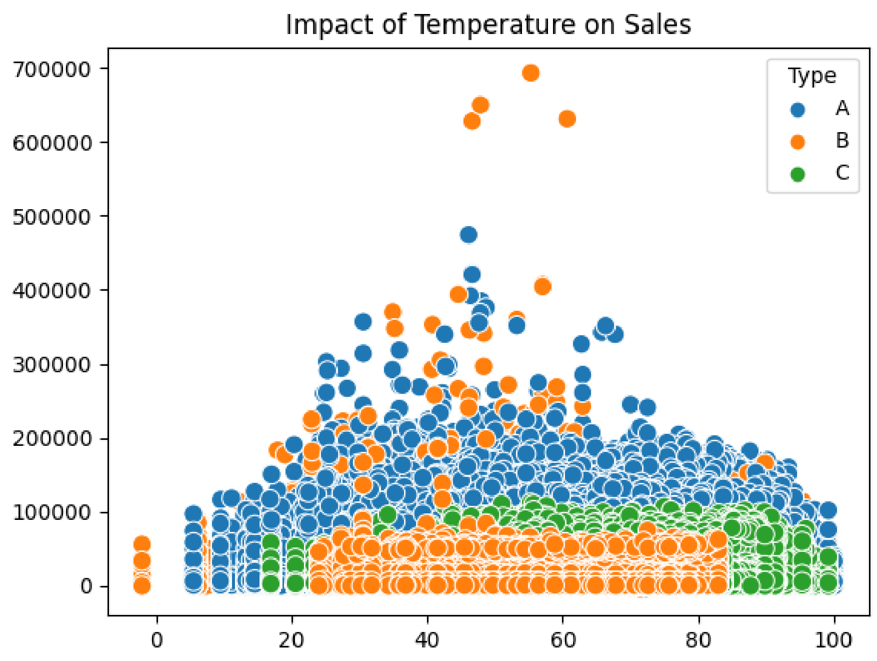


Figure 17. Impact of Temperature on Sales

4.2.12 Impact of Unemployment on Sales

Spending sharply drops on the onset of unemployment; a higher unemployment index would generally result in a dip in sales as individuals tend to decrease overall spending. In our dataset, unemployment is presented through an index of the unemployment rate during that week in the region of the store. From our scatter plot, it is easier to gather the following information:

- For the given store types, there seems to be a visible decrease in sales when the unemployment index is higher than 11
- Even when the unemployment index is higher than 11, there is no significant change in the average sales for Type C stores when compared to the overall sales
- There seems to be a significant drop in sales for store types A and B when the unemployment index increases
- Highest recorded sales for store types A and B occur around the unemployment index of 8 to 10; this gives ambiguous ideas about the impact of unemployment on sales for each of the stores

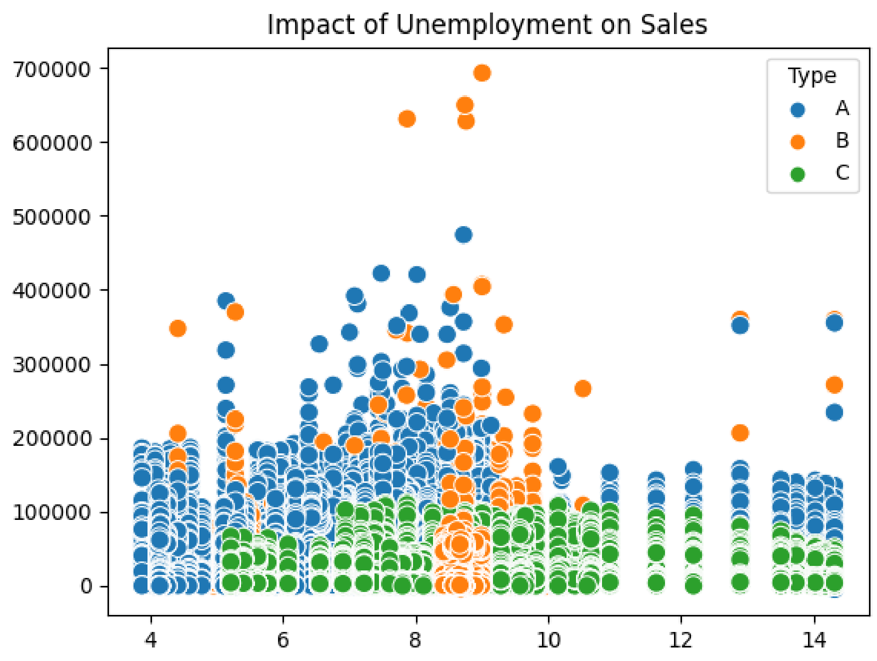


Figure 18. Impact of Unemployment on Sales

4.2.13 Impact of CPI on Sales

According to the US Bureau of Labor Statistics, CPI (Consumer Price Index) is defined as the measure of the average change over time in the prices paid by urban consumers for a market basket of consumer goods and services (STATISTICS, n.d.).

In layman's terms, CPI is a measure that assesses the price changes that are associated with the cost of living for each individual. CPI is a great measure for the government when studying inflation (i.e. an increase in the prices of a representative basket of goods consumed) and is often used to evaluate and adjust government assistance needs based on income levels and provide wage adjustments with respect to changing cost of living expenses. A higher CPI generally means that the price of goods has increased and that an individual needs to spend more money to maintain the same standard of living.

In our scatter plot above, we can identify three different clusters around different ranges of CPI; while there seems to be no visible relationship between the change in CPI and weekly sales for Walmart stores (sales still occur at high CPI rates), the only negligible observation that can be made is the high amount of sales for store Type B when CPI is at a low rate of 140.

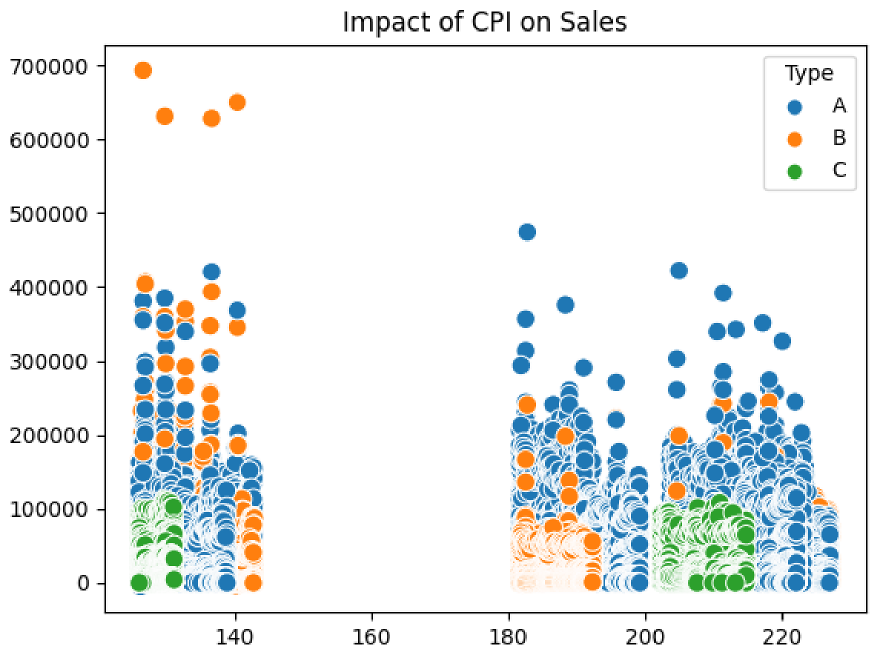


Figure 19. Impact of CPI on Sales

4.2.14 Impact of Fuel Price on Sales

An economist from Brown University used gasoline price data to study if consumers change their buying behavior based on changes in fuel prices (Baum, 2011). The economist assumes that even a slight increase in fuel prices significantly adds up to the annual expenditure and thus discourages consumers from actively buying their required goods and services.

This can also slightly be observed in the visualization below; while there seems to be a decrease in sales when fuel price is higher than 4.25 dollars, sales are higher when fuel price ranges between 2.75 to 3.75 dollars. Some of the highest occurring sales for store types A and B happen during this period. While there is no definite pattern that proves this, some observations do support the theory that lower fuel prices encourage higher sales.

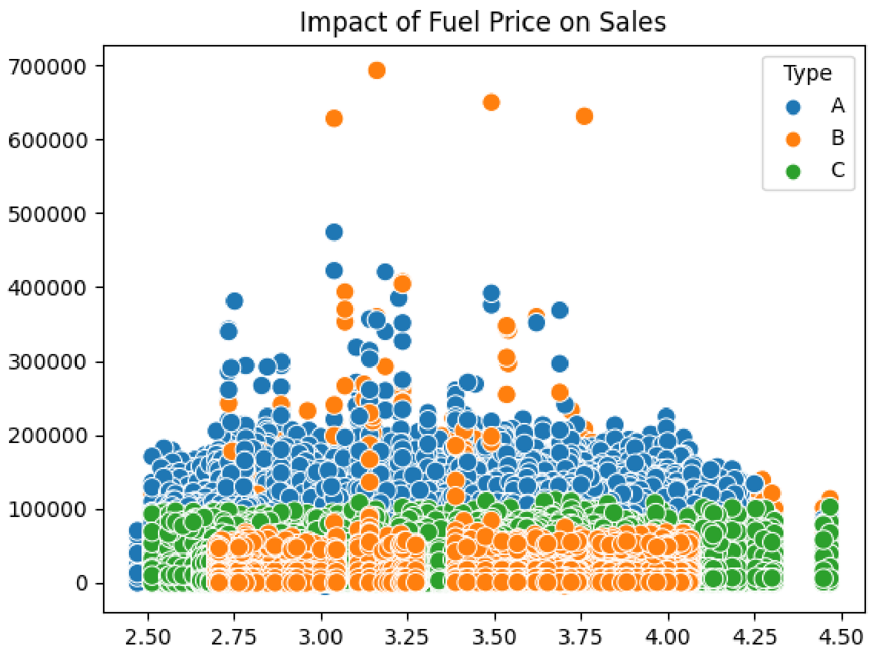


Figure 20. Impact of Fuel Price on Sales

4.2.15 Holiday VS Non-Holiday Sales

The dataset provided contains data about weekly Walmart sales over various periods of time in a year, this includes data about the sales that occur during holiday periods of Thanksgiving, Christmas, etc. It was crucial to compare the difference between sales during holidays and normal weeks to understand if the holiday season gathers higher sales. For this comparison, I first counted the number of holidays in a year and compared sales during the holiday dates versus the normal days. While the holiday dates only accounted for almost 7 percent of the days in the year, they still have higher weekly sales than the rest of the year combined (as seen in the image below).

The 'IsHoliday' = TRUE sales and counts can be observed in Figure 21 and highlight that even though the number of holiday dates is far less than the normal dates in the dataset, they still accumulate more sales.

```
# Holiday VS Non-Holiday Sales

sales = merge_train.groupby('IsHoliday')['Weekly_Sales'].mean()
counts = merge_train.IsHoliday.value_counts()
```

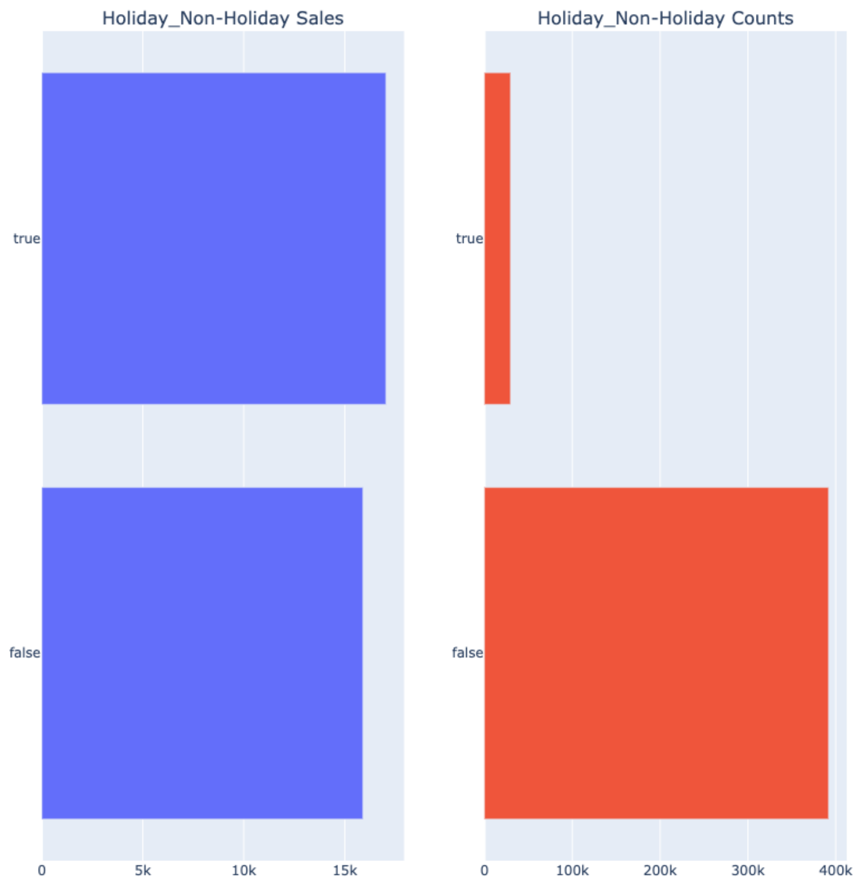


Figure 21. Holiday Versus Non-Holiday Sales

4.2.16 Correlation Matrix

A correlation matrix describes the correlation between the various variables of a dataset. Each variable in the table is correlated to each of the other variables in the table and helps in understanding which variables are more closely related to each other (Glen, 2016).

With the numerous variables available through this dataset, it became imperative to study correlations between some of them. By default, this matrix also calculates correlation through Pearson's Correlation Coefficient (Johnson, 2021) that calculates the linear relationship between two variables, within a range of -1 to $+1$. The closer the correlation to $|1|$, the higher the linear relationship between the variables and vice versa.

Pearson's Correlation Coefficient (Johnson, 2021) is calculated as:

$$r = \frac{\mathrm{Cov}(x,y)}{\sigma_x \sigma_y} \tag{1}$$

with

- $\mathrm{Cov}(x,y) = \frac{\sum(x-\bar{x})(y-\bar{y})}{n-1}$
- $\sigma_x = \sum(x-\bar{x})^2$
- $\sigma_y = \sum(y-\bar{y})^2$

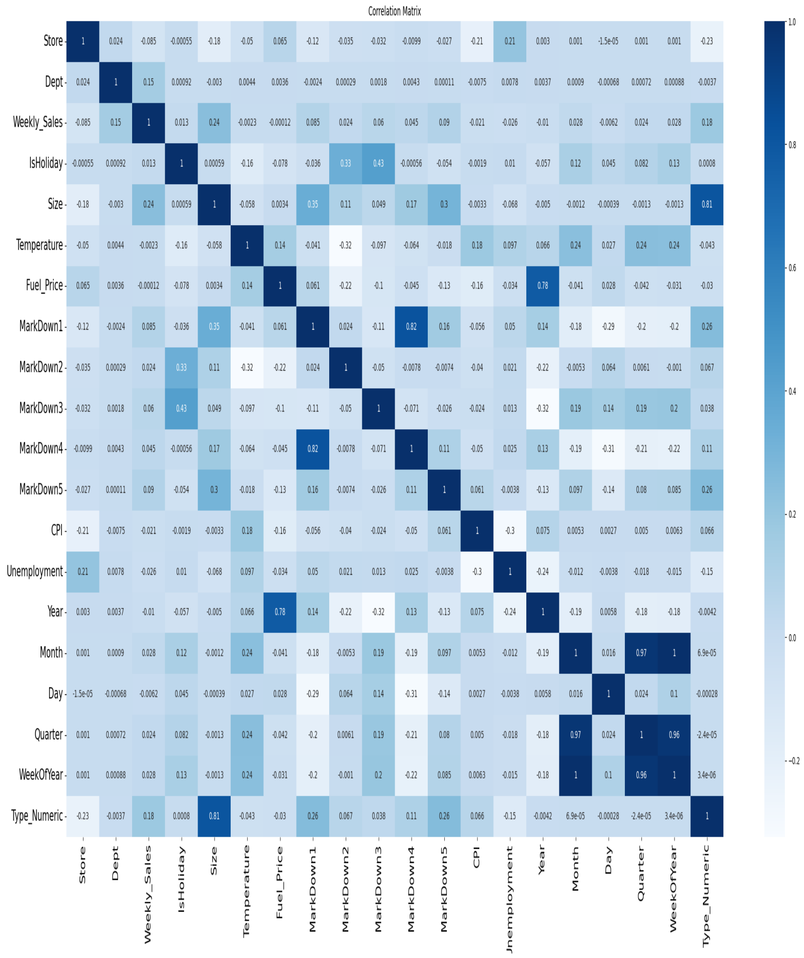


Figure 22. Correlation Matrix

The heatmap/correlation matrix in Figure 22, created using the seaborn library in Python (Szabo, 2020) gives the following information:

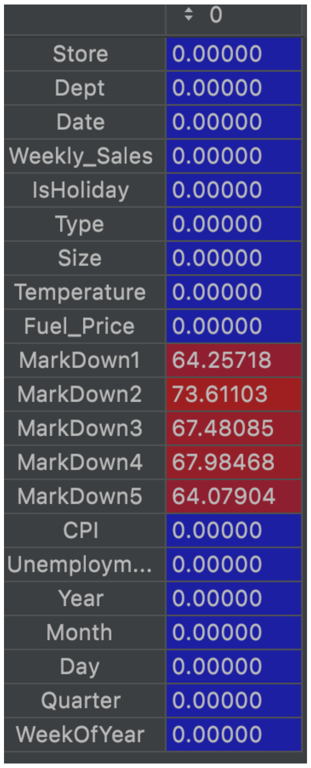
- There is a slight correlation between weekly sales and store size, type, and department
- There seems to be a negative correlation between weekly sales and temperature, unemployment, CPI, and fuel price. This could suggest that sales are not impacted by changes in these factors
- Markdowns 1-5 also seem to have no distinct correlation with weekly sales, thus they are not as important a factor in the study

4.3 Data Cleaning and Preprocessing

The data contains 421,570 rows, with some store-specific departments missing a few too many weeks of sales. As observed in Figure 4, some columns in the features dataset contain missing values, however, after the features dataset is merged with the training dataset, the only missing values that exist are in the Markdown columns (as shown in figure 23).

After the extensive EDA, it was determined that these five markdown files, with missing values, have barely any correlation to the weekly sales for Walmart, hence these five columns have been eliminated from the subsequent training and testing dataset. Because the source already provides training and testing datasets, there is no need to create them for our study.

Because the main focus of this study is to accurately predict weekly sales for different Walmart stores, the previously modified 'Date', 'Month', 'Quarter', and 'Day' columns have been dropped and only the 'Week of Year' column has been used in the upcoming models.



	0
Store	0.00000
Dept	0.00000
Date	0.00000
Weekly_Sales	0.00000
IsHoliday	0.00000
Type	0.00000
Size	0.00000
Temperature	0.00000
Fuel_Price	0.00000
MarkDown1	64.25718
MarkDown2	73.61103
MarkDown3	67.48085
MarkDown4	67.98468
MarkDown5	64.07904
CPI	0.00000
Unemploym...	0.00000
Year	0.00000
Month	0.00000
Day	0.00000
Quarter	0.00000
WeekOfYear	0.00000

Figure 23. Missing Values

Data has been checked for inaccuracies, missing or out of range values using the 'inspectdf' package in R as part of the initial EDA. Columns with missing values have been dropped. The dataset contains information about weekly sales which was initially broken down to acquire information about monthly as well as quarterly sales for our analysis, however, that information is not going to be utilized during the modeling process.

The boolean 'isHoliday' column in the dataset contains information about whether the weekly date was a holiday week or not. As observed in the EDA above, sales have been higher during the holiday season as compared to non-holiday season sales, hence the 'isHoliday' column has been used for further analysis.

Furthermore, as part of this data preprocessing step, I have also created input and target data frames along with the training and validation datasets that help accurately measure the performance of applied models. In addition, as part of this data preprocessing, feature scaling (Vashisht, 2021) has been applied to normalize different data attributes. This has primarily been done to unionize the independent variables in the training and testing datasets so that these variables will be centered around the same range (0,1) and provide more accuracy.

Also referred to as normalization, this method uses a simple min-max scaling technique (implemented in Python using the Scikit-learn (Sklearn) library (Pedregosa et al., 2011).

Lastly, as part of this competition, Walmart wished that participants assess the accuracies of models using the Weighted Mean Absolute Error (WMAE) ("Walmart Recruiting - Store Sales Forecasting," 2014), a brief description of which is displayed as follows.

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

where

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

The Weighted Mean Absolute Error is one of the most common metrics used to measure accuracy for continuous variables (JJ, 2016).

A WMAE function has been created that provides a measure of success for the different models applied. It is the average of errors between prediction and actual observations, with a weighting factor. In conclusion, the smaller the WMAE, the more efficient the model.

4.4 Model Selection and Implementation

Trying to find and implement the most effective model is the biggest challenge of this study. Selecting a model will depend solely on the kind of data available and the analysis that has to be performed on the data (UNSW, 2020).

Several models have been studied as part of this study that were selected based on different aspects of our dataset; the main purpose of creating such models is to predict the weekly sales for different Walmart stores and departments, hence, based on the nature of models that should be created, the following four machine learning models have been used:

- Linear Regression
- Lasso Regression
- Gradient Boosting Machine
- Random Forest

Each of these methods have been discussed briefly in the upcoming report. For each of the models, why they were chosen, their implementation and their success rate (through WMAE) have been included.

4.4.1 Linear Regression

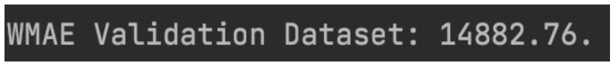
Regression analysis helps find a relationship between two or more variables, an independent variable (predictor) and a dependent variable (target), in a dataset. Regression analysis specifies how the value of a target variable changes with a change in the value of the predictor variable when all other variables in the dataset are constant and evaluate the accuracy of this relationship (Panchotia, 2020).

Starting with the most basic and straightforward model for this analysis, linear regression aims at finding relationships between two linear variables, a predictor and a target. For the purpose of this study, multiple linear regression helps in predicting the future value of a numeric variable based on past data (Bari et al., n.d.).

Linear regression is most suitable for linear data, i.e. data without the presence of outliers as these disrupt the linear nature of the dataset, hence resulting in a high error rate and low model performance. It is imperative to deal with outliers (influential points) in the training dataset before creating linear regression models as the presence of such outliers affects the regular distribution of points, i.e. slope of the model, resulting in inaccuracies.

The ‘scikit-learn’ library along with the ‘Linear Regression’ function in Python has been used to create the linear regression model. The training dataset has been further divided into the training and validation datasets; the validation dataset is primarily created to test models based on different parameters to see what parameters result in higher accuracy. (Myrianthous, n.d.).

After implementing the linear regression model and generating predictions on both datasets (training and validation), the WMAE (weighted mean absolute error) was calculated (based on the previously defined function in the data cleaning and preprocessing section) which was as follows:



```
WMAE Validation Dataset: 14882.76.
```

Figure 24. WMAE Rate for Linear Regression

Considering that the WMAE values for the validation data is extremely high, linear regression cannot be considered as a very efficient model for this analysis based on this model.

Apart from the ’scikit-learn’ library in Python, further analysis and cross-validation has been performed on the training dataset using the lm() function in R (created another linear regression model using the variables pre-selected in data processing above). The lm() function takes in two main arguments, namely:

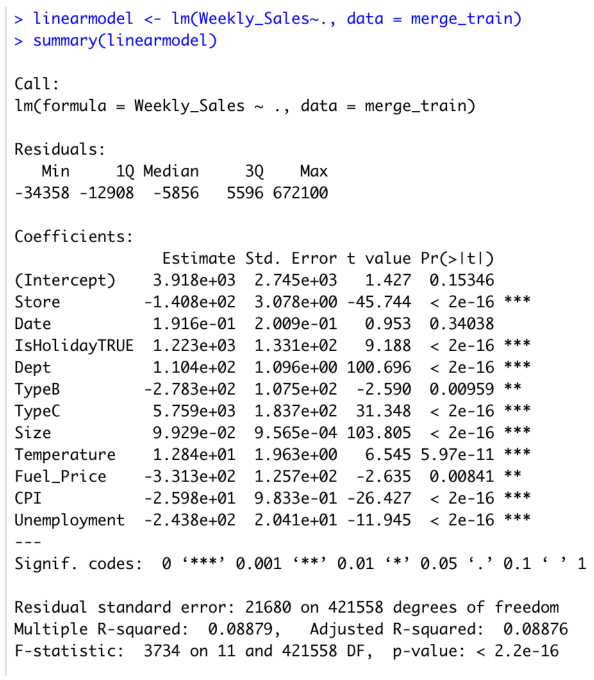
- 1. Formula
- 2. Data

The data is typically a data.frame and the formula is an object of the class formula (Prabhakaran, 2016). Based on the linear model, the coefficients, F-statistic, RSE score, etc. have been calculated as shown below and in Figure 25.

```
linearmodel <- lm(Weekly_Sales~.,data=merge_train)
summary(linear)
summary(linear)$coefficient
summary(linear)$fstatistic
# Residual Standard Error
sigma(linearmodel)/mean(merge_train$Weekly_Sales)
```

An important step in analyzing the linear model is to look at the variable coefficients ad the F-statistic at the end of the model summary. This summary table highlights the estimate of regression beta coefficients and their respective p-values and helps in identifying the variables that are significantly related to weekly sales.

The link between a predictor variable and the response is described by regression coefficients, which are estimations of unknown population parameters. Coefficients are the values that multiply the predictor values in linear regression (Frost, 2021). The value of a coefficient determines the relationship between a predictor and response variable: a positive relationship is indicated by a positive sign and vice versa. Another metric calculated in the model is the F-statistic value which tells if a group of variables are statistically significant and helps analyze whether to support or reject the null hypothesis. The value is mostly always used with the p-value to determine the significance of a model by studying all variables that are significant.



```
> linearmodel <- lm(Weekly_Sales~., data = merge_train)
> summary(linearmodel)

Call:
lm(formula = Weekly_Sales ~ ., data = merge_train)

Residuals:
   Min     1Q Median     3Q    Max 
-34358 -12908  -5856   5596 672100 

Coefficients:
               Estimate Std. Error t value Pr(>|t|)    
(Intercept)   3.918e+03  2.745e+03   1.427  0.15346    
Store        -1.408e+02  3.078e+00 -45.744  < 2e-16 ***
Date          1.916e-01  2.009e-01   0.953  0.34038    
IsHolidayTRUE  1.223e+03  1.331e+02   9.188  < 2e-16 ***
Dept          1.104e+02  1.096e+00 100.696  < 2e-16 ***
TypeB        -2.783e+02  1.075e+02  -2.590  0.00959 ** 
TypeC         5.759e+03  1.837e+02  31.348  < 2e-16 ***
Size          9.929e-02  9.565e-04 103.805  < 2e-16 ***
Temperature   1.284e+01  1.963e+00   6.545 5.97e-11 ***
Fuel_Price   -3.313e+02  1.257e+02  -2.635  0.00841 ** 
CPI          -2.598e+01  9.833e-01 -26.427  < 2e-16 ***
Unemployment -2.438e+02  2.041e+01 -11.945  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 21680 on 421558 degrees of freedom
Multiple R-squared:  0.08879,	Adjusted R-squared:  0.08876 
F-statistic:  3734 on 11 and 421558 DF,  p-value: < 2.2e-16
```

Figure 25. Multiple Linear Regression in R Summary: Coefficients and F-Statistic

As per the figure above, variables with p-value of the F-statistic lower than 2.2e-16 have some significant relationship with the dependent variable. This means that variables like department, temperature, size, and isHoliday have some statistical relationship with weekly sales. For the F-statistic, the lm function runs an ANOVA test to check the significance of the overall model. Here the null hypothesis is that the model is not significant. According to the $p < 2.2e - 16$, our model is significant.

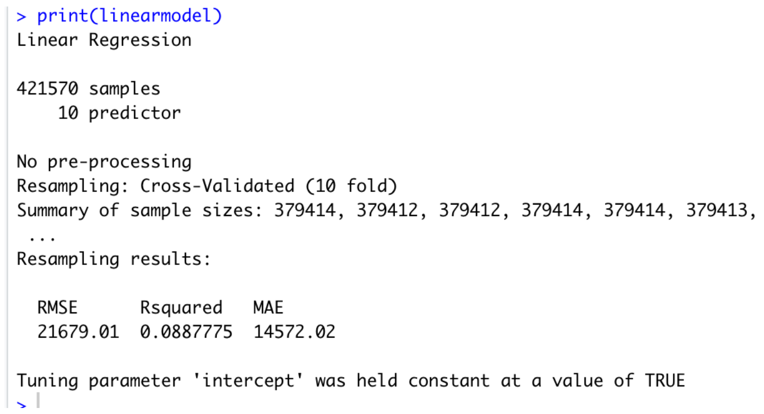
The summary also provides the R squared values that determine how well our data fits our linear model, in simple terms, it tells us about the goodness of fit of the model. This value tells us how close the predicted data values are to the fitted regression line. In general, if the differences between the actual values and the model's predicted values are minimal and unbiased, the model fits the data well (Editor, 2013). R square values range from 0 to 1, hence the closer the value of R-square to 1, the better the fit. However, as shown in Figure 25, the model has a relatively small R-square value of 0.08 which suggests that the model might not exactly be a good fit.

The RSE (Residual Standard Error) or SIGMA gives a measure of the error of prediction. The lower the RSE, the more accurate the model. The RSE has been calculated as follows:

sigma(linearmodel)/mean(WeeklySales)

The RSE value for the model comes out at 1.35 which is relatively higher denoting high standard error and less efficiency.

As the next step, I utilized a cross-validation or rotation estimation technique that resamples the data under study and trains the data iteratively using different chunks of data to avoid overfitting.



```
> print(linearmodel)
Linear Regression

421570 samples
    10 predictor

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 379414, 379412, 379412, 379414, 379414, 379413,
 ...
Resampling results:

  RMSE      Rsquared   MAE
  21679.01  0.0887775  14572.02

Tuning parameter 'intercept' was held constant at a value of TRUE
>
```

Figure 26. Trained Multiple Linear Regression Model Summary

According to Figure above, even with the ten-fold cross-validation, the R square value is still extremely low to establish statistical significance for the model.

4.4.2 Interaction Effects with Regression

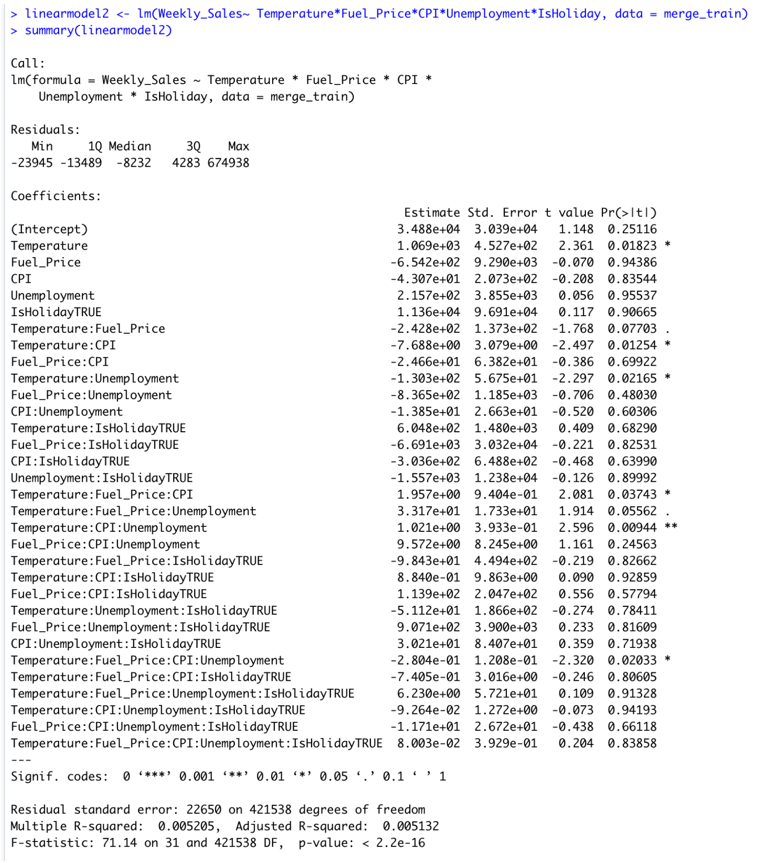
After observing the effects of factors like temperature, unemployment, fuel price, etc. on the dependent variable (i.e. weekly sales), it is also imperative to also study the effect of a combination of such factors on weekly sales. If an interaction effect exists, the value of a single variable depends on the level of different variables; with the help of the regression model, interaction effects can be tested if the independent variables are either continuous or categorical. Some of the relationship effects studied in the exploratory data analysis might not be true reflections of the effect of such factors on the weekly sales, hence, the need to study relationships between several factors and their effect on weekly Walmart sales.

Considering that an important part of this study has been to study the effects of such factors on weekly sales, the interaction effects utilizes five main components/variables from the dataset to study such interaction effects:

- Temperature
- Unemployment
- CPI
- Fuel Price
- IsHoliday

A combination of these factors has been studied to understand if more than one factor jointly contributes towards the increase or decrease of weekly sales for Walmart stores under study.

For the same, the 'tidyverse' and 'caret' packages have been used in R for data manipulation and implementing machine learning model functions respectively. Just as the linear regression model has been created, a similar model has been created to study these interaction effects, however, instead of using all independent variables from the dataset, only the five main ones listed above are used in the linear regression model. The '*' operator helps study interaction effects between several independent and dependent variables in the linear regression model (kassambara, 2018).



```
> linearmodel2 <- lm(Weekly_Sales~ Temperature*Fuel_Price*CPI*Unemployment*IsHoliday, data = merge_train)
> summary(linearmodel2)

Call:
lm(formula = Weekly_Sales ~ Temperature * Fuel_Price * CPI * 
    Unemployment * IsHoliday, data = merge_train)

Residuals:
   Min     1Q Median     3Q    Max 
-23945 -13489  -8232   4283 674938 

Coefficients:
                                                 Estimate Std. Error t value Pr(>|t|)   
(Intercept)                                     3.488e+04  3.039e+04   1.148  0.25116   
Temperature                                     1.069e+03  4.527e+02   2.361  0.01823 * 
Fuel_Price                                     -6.542e+02  9.290e+03  -0.070  0.94386   
CPI                                            -4.307e+01  2.073e+02  -0.208  0.83544   
Unemployment                                    2.157e+02  3.855e+03   0.056  0.95537   
IsHolidayTRUE                                   1.136e+04  9.691e+04   0.117  0.90665   
Temperature:Fuel_Price                         -2.428e+02  1.373e+02  -1.768  0.07703 . 
Temperature:CPI                                -7.688e+00  3.079e+00  -2.497  0.01254 * 
Fuel_Price:CPI                                 -2.466e+01  6.382e+01  -0.386  0.69922   
Temperature:Unemployment                       -1.303e+02  5.675e+01  -2.297  0.02165 * 
Fuel_Price:Unemployment                        -8.365e+02  1.185e+03  -0.706  0.48030   
CPI:Unemployment                               -1.385e+01  2.663e+01  -0.520  0.60306   
Temperature:IsHolidayTRUE                       6.048e+02  1.480e+03   0.409  0.68290   
Fuel_Price:IsHolidayTRUE                       -6.691e+03  3.032e+04  -0.221  0.82531   
CPI:IsHolidayTRUE                              -3.036e+02  6.488e+02  -0.468  0.63990   
Unemployment:IsHolidayTRUE                     -1.557e+03  1.238e+04  -0.126  0.89992   
Temperature:Fuel_Price:CPI                      1.957e+00  9.404e-01   2.081  0.03743 * 
Temperature:Fuel_Price:Unemployment             3.317e+01  1.733e+01   1.914  0.05562 . 
Temperature:CPI:Unemployment                    1.021e+00  3.933e-01   2.596  0.00944 **
Fuel_Price:CPI:Unemployment                     9.572e+00  8.245e+00   1.161  0.24563   
Temperature:Fuel_Price:IsHolidayTRUE           -9.843e+01  4.494e+02  -0.219  0.82662   
Temperature:CPI:IsHolidayTRUE                   8.840e-01  9.863e+00   0.090  0.92859   
Fuel_Price:CPI:IsHolidayTRUE                    1.139e+02  2.047e+02   0.556  0.57794   
Temperature:Unemployment:IsHolidayTRUE         -5.112e+01  1.866e+02  -0.274  0.78411   
Fuel_Price:Unemployment:IsHolidayTRUE           9.071e+02  3.900e+03   0.233  0.81609   
CPI:Unemployment:IsHolidayTRUE                  3.021e+01  8.407e+01   0.359  0.71938   
Temperature:Fuel_Price:CPI:Unemployment        -2.804e-01  1.208e-01  -2.320  0.02033 * 
Temperature:Fuel_Price:CPI:IsHolidayTRUE       -7.405e-01  3.016e+00  -0.246  0.80605   
Temperature:Fuel_Price:Unemployment:IsHolidayTRUE      6.230e+00  5.721e+01   0.109  0.91328   
Temperature:CPI:Unemployment:IsHolidayTRUE     -9.264e-02  1.272e+00  -0.073  0.94193   
Fuel_Price:CPI:Unemployment:IsHolidayTRUE      -1.171e+01  2.672e+01  -0.438  0.66118   
Temperature:Fuel_Price:CPI:Unemployment:IsHolidayTRUE  8.003e-02  3.929e-01   0.204  0.83858   
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 22650 on 421538 degrees of freedom
Multiple R-squared:  0.005205,	Adjusted R-squared:  0.005132 
F-statistic: 71.14 on 31 and 421538 DF,  p-value: < 2.2e-16
```

Figure 27. Multiple Linear Regression: Studying Interaction Effects

The additive effect applied to the model uses only the specified variables and the model creates and evaluates the significance of every possible combination of these variables. As per the Figure above, significant interactions can be identified using the coefficient estimates as well as the p-value for the combinations of variables. Based on the Figure, the only significant relationship seems to exist between 'Temperature',

’CPI’ and ’Unemployment’. This suggests that a combination of these three independent variables can have a significant impact on the predictor variable, i.e. ideal values for the three independent variables can signify a direct relationship with the independent variable, meaning higher sales in ideal conditions. There also seems to be a slight relationship between ’Temperature’, ’Fuel Price’ and ’CPI’ but this is not as significant as the previous relationship.

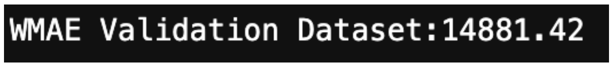
One important thing to note about the model is that the R-square value for the model is very low, denoting a very poor fit of the data points around its mean. Hence, as a result of low statistical significance, this model might not be the best method to establish proof of the existence of such relationships.

4.4.3 Lasso Regression

Lasso (least absolute shrinkage and selection operator) regression, also known as regularized linear regression, is another regression analysis model that handles data that suffer from multicollinearity and is primarily suitable when there are several predictor variables in the data. Lasso regression is an extension of linear regression in which the loss function is modified to minimize the complexity of the model by limiting the sum of the absolute values of the model coefficients. (Singh, 2019)

The algorithm creates a penalty against complexity by adding a regularization term such that with increasing value of the regularization parameter, the weights are reduced (Kumar, 2020). As the value of the regularization parameter (here alpha) goes up, it reduces the absolute weight values shrink and the loss function is curtailed. Lasso is also considered as an effective feature selection method as when the loss function decreases because of regularization, some of the features from the dataset are removed as their weights become zero. One of the key differences between linear and several other regression models is that while linear regression does not exactly tune the parameters, the other models allow for tuning of the hyperparameter, in this case, lambda.

With the same ‘scikit-learn’ library in Python, a ’Lasso’ function helps create the lasso regression model on the training and validation datasets. The WMAE function helps calculate the error rate for this model.



```
WMAE Validation Dataset:14881.42
```

Figure 28. WMAE Rate for Lasso Regression

```
# Create an instance of Lasso Regression implementation
lasso = Lasso(alpha=1.0)


# Creating model
```

```
lr = lasso.fit(train_inputs, train_targets)

# Generate prediction (training data)
pred_train2 = lr.predict(train_inputs)

# WMAE for training dataset
WMAE_train2 = WMAE(train_inputs, train_targets, pred_train2)

# Generate predictions (validation data)
pred_valid2 = lr.predict(val_inputs)

# WMAE for validation data
WMAE_valid2 = WMAE(val_inputs, val_targets, pred_valid2)
print('WMAE Validation Datatset: {}.'.format(WMAE_valid2))
```

Just like the linear regression model, the lasso regression model does not perform well on the validation data, giving a relatively higher WMAE value. Hence, the model cannot exactly be considered as an efficient model for the study.

Some further analysis has been done for the model in R using the ’glmnet’ package. With the help of the cross-validation glmnet function (i.e. cv.glmnet()), an optimal value for lambda is obtained from the model. The lowest point on the curve represents the optimal lambda: the log value of lambda that best-minimized cross-validation error.

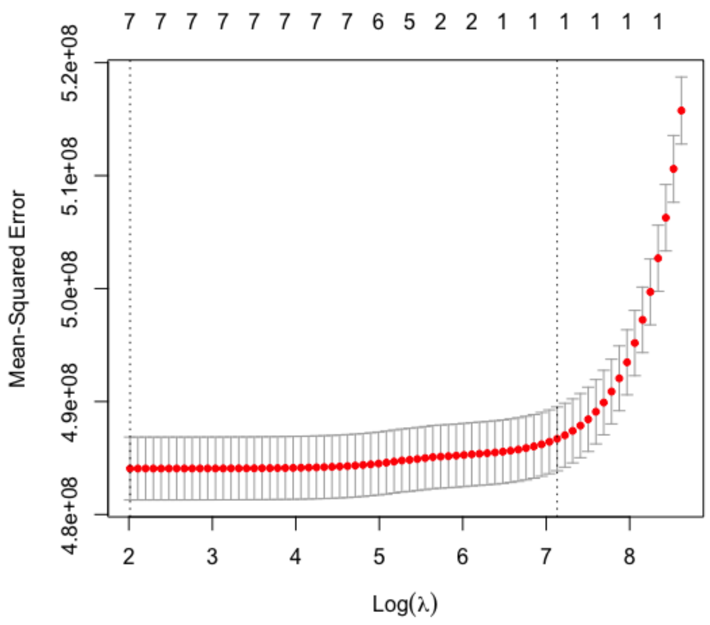
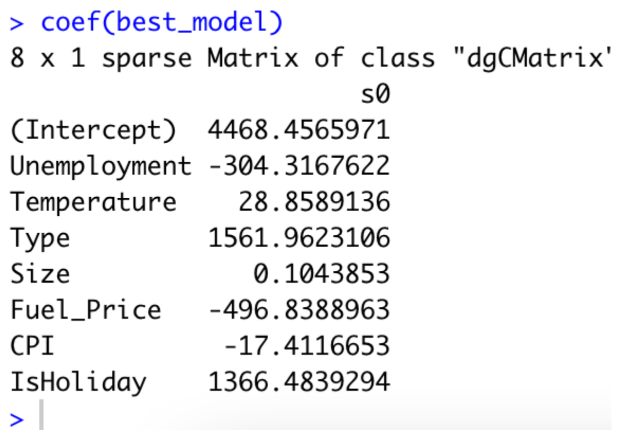


Figure 29. Lasso Regression: Optimal Lambda value (7.49)

The model is then rebuilt with this optimal lambda value, with alpha as 1, and coefficients are calculated. We can again observe positive coefficients for variables like temperature, type, and isHoliday that depict the slight statistical significance of such variables with weekly sales as per Figure 30.



```
> coef(best_model)
8 x 1 sparse Matrix of class "dgCMatrix'
                        s0
(Intercept)  4468.4565971
Unemployment -304.3167622
Temperature    28.8589136
Type         1561.9623106
Size            0.1043853
Fuel_Price   -496.8388963
CPI           -17.4116653
IsHoliday    1366.4839294
> 
```

Figure 30. Lasso Regression Coefficients

The R-squared value for the model has been calculated using the Sum of Squares

Total (SST) and Sum of Squares Error (SSE) values as observed below. With a very low R2 value of 0.06, the model cannot be considered a good fit for predicting weekly sales for Walmart.

```
# Sum of Squares Total and Error
sst <- sum((y_var - mean(y_var))^2)
sse <- sum((predicted - y_var)^2)
# R squared
rsq <- 1 - sse / sst
rsq
```

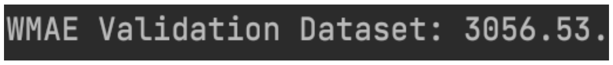
4.4.4 Gradient Boosting Machine

Gradient boosting is a sequential technique, based on ensemble learning, which combines the decisions from multiple machine learning models to improve the overall performance of the model. In boosting, the model outcomes for any instance in the sequence are weighed based on the results of the last instance. The correct predictions are assigned a lower weight, while the incorrect predictions are assigned a higher weight. The model will then focus on higher weight points as it might go wrong with the lower weight points. After many iterations, a combined result is generated, using the accuracy of all the models.

Gradient Boosting Machine uses 3 types of parameters: Tree-Specific Parameters, Boosting Parameters, and other miscellaneous parameters (Jain, 2016).

Using GBM provides multiple advantages: it usually provides better predictive accuracy compared to other models, it provides various hyperparameter tuning options, is time-efficient for larger datasets, and handles missing data (Guide, n.d.).

Using the XGBRegressor object from the xgboost library under ‘scikit-learn’, the training and validation datasets were used to create a basic gradient boosting model and the WMAE was calculated. The following error rate was calculated with the ‘random state’ and ‘n jobs’ parameters applied to this initial model:



```
WMAE Validation Dataset: 3056.53.
```

Figure 31. WMAE Rate for Gradient Boosting Machine

As observed from the image above, the error rate for the initial model is high; several tuning parameters have been applied to the model below to reduce this error.

One important aspect of feature importance with gradient boosting is that with the creation of the boosted trees with the model, it is easier to capture the importance of each attribute. An importance score is assigned to each attribute that is calculated by the gradient boosting model by ranking and comparing attributes with each other. The importance of an attribute is weighted by the amount that each split improves on the performance metric, weighted by the number of observations for which the node is responsible. The performance measure could be purity (Gini index) used to select split points or another more specific error function. The significance of the features is then averaged over all decision trees of the model (Brownlee, 2016).

Harmonious to the findings in the EDA, the ‘Dept’ and ‘Size’ attributes are the two most important features that affect weekly sales for the retail giant.

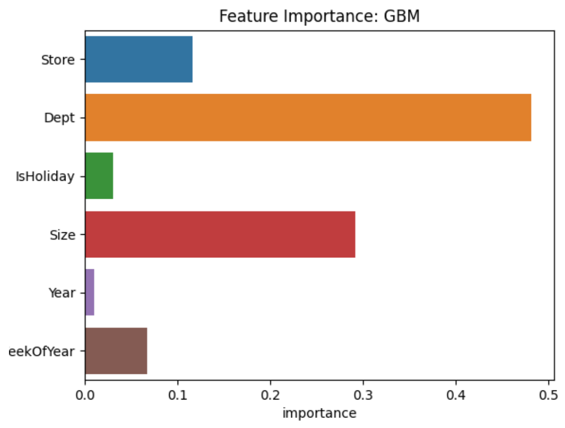
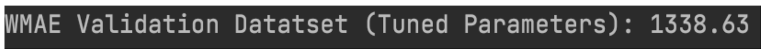


Figure 32. Feature Importance for Gradient Boosting Machine

There are several tuning parameters associated with the model which can assist in reducing the WMAE rate and attaining higher accuracy for the model. Under the scikit-learn library in python, several functions like ’min samples split’, ’max Depth’, ’min samples leaf’, ’max features’, ’n estimators’, ’random state’, etc. can be used to tune the GBM model.

For the purpose of this study, the ‘random state’ (seed that generates random numbers for parameter tuning), ‘n jobs’ (reduce computation time for parallel processes), ’n estimators’ (modeling of sequential trees), ’max depth’ (define the maximum depth of a tree to control overfitting), and ’learning rate’ (impact of each tree on the model outcome) (Jain, 2016) functions have been applied and this results in following WMAE rate:



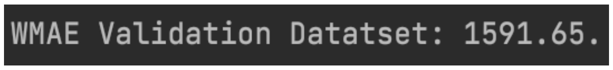
```
WMAE Validation Datatset (Tuned Parameters): 1338.63
```

Figure 33. WMAE Rate for Gradient Boosting Machine: Tuned Parameters

4.4.5 Random Forest

The random forest regression operates by making multiple and different regression decision trees at the time of training. Each tree predicts a decision based on the criteria it picked. The random forest then makes a prediction by taking the average of individual predictions (Bakshi, 2020).

Random forest usually has good accuracy compared to other linear models and scales well with new features or samples. This regression model can handle missing data and outliers which makes it time-saving and easy to use (Keboola, 2020). This model is powerful because it performs well on various problems, including attributes with non-linear relationships. Using the 'RandomForestRegressor' object in the xgboost library, a basic model was created, with some initial parameters, and the WMAE rate was calculated for the training and validation sets:



```
WMAE Validation Datatset: 1591.65.
```

Figure 34. WMAE Rate for Random Forest

As observed from the image above, the error rate for the initial model is high; several tuning parameters have been applied to the model below to reduce this error.

Just like the gradient boosting model, the feature importance plot below highlights the most important attributes in the random forest model; the 'feature importance' function in python's scikit-learn library is used with the 'RandomForestRegressor' and computes importance for each of the attributes using Gini index (Płoński, 2020). According to the feature importance barplot below, the 'Dept' and 'Size' attributes impact weekly sales the most, which corresponds to the correlation matrix findings in the EDA.

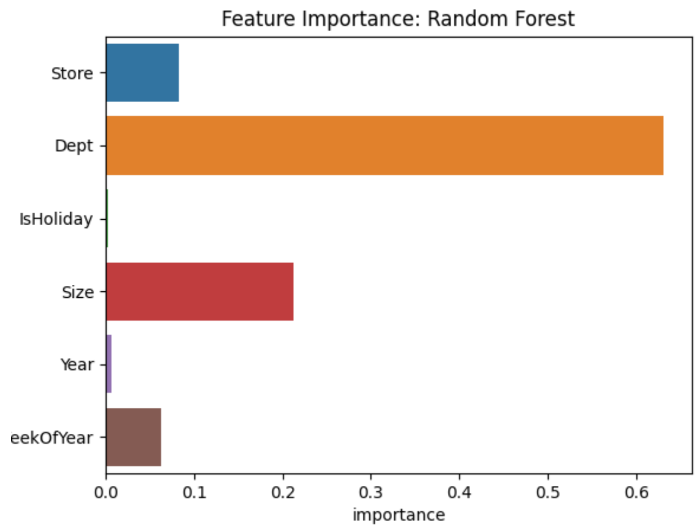
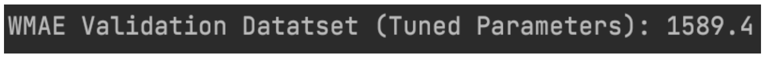


Figure 35. Feature Importance for Random Forest

This model was also tuned based on similar parameters that were previously applied for the tuning of the GBM model and includes some additional parameters like 'min samples split' (minimum samples required in a node for a split), 'min samples leaf' (minimum required samples in a leaf), 'max samples', and 'max features' (maximum samples required for a split) (Jain, 2016).

With the parameters set, the new WMAE rate for the Random Forest model is:



```
WMAE Validation Datatset (Tuned Parameters): 1589.4
```

Figure 36. WMAE Rate for Random Forest: Tuned Parameters

The general idea when trying to select the most efficient model would have suggested looking at the R^2 (proportion of variance explained), the Root Average Squared Error (RASE), or the p-values to generate a predictive model that can inform decision-makers about what output is expected given certain input (Yu et al., 2018).

Hence, in addition to WMAE, several other metrics have been created for the sake of this study.

Table 3. Model Performance

WMAE Rate	Linear Regression	Lasso Regression	GBM (Tuned)	Random Forest (Tuned)
WMAE:Validation	14882.76	14881.42	1338.63	1589.4

4.5 Selection of the Most Accurate Model

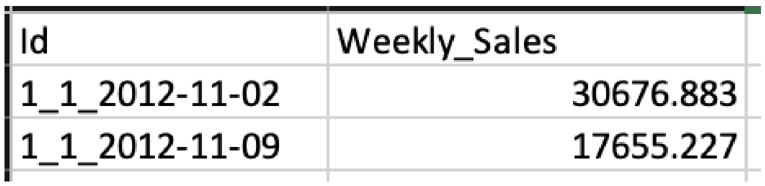
The model with the lowest WMAE score is the most efficient model for our predictions. The WMAE score measures the magnitude of errors in our study; it measures the value of the difference between the predicted values and input values and defines accuracy for a model. Considering all this, the lower the WMAE values, the better the model's performance.

Based on Table 3, the lowest WMAE score is provided by the Gradient Boosting Machine model with tuned parameters and predictions generated from this model that have been used to create the final Power BI dashboard.

4.6 Building the Power BI Dashhoard

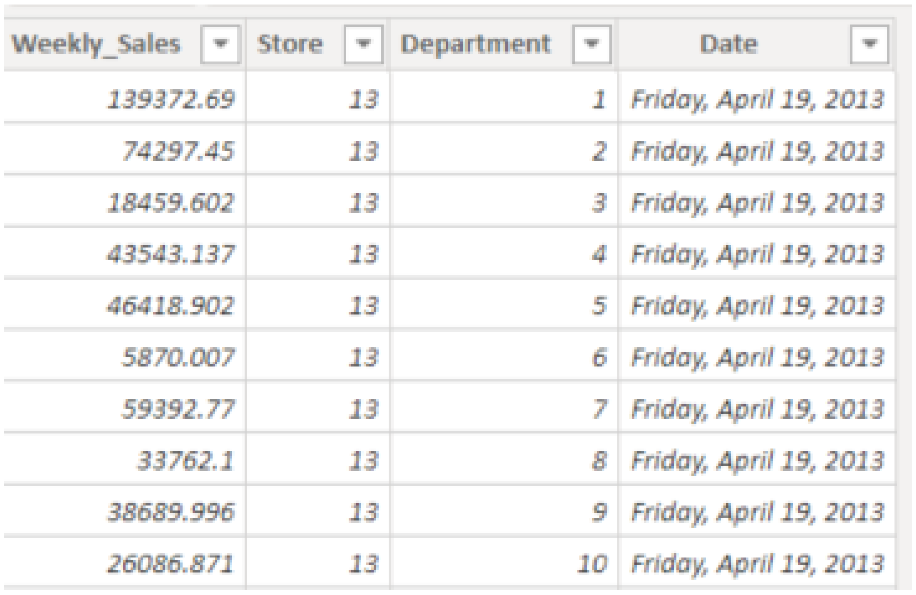
As an end product, this Power BI dashboard is going to serve as the final product of this research. The dashboard contains detailed information about the original data related to the 45 Walmart stores as well as displays their respective predicted weekly sales. Most of the explorations that have been performed as part of the EDA will be included in this dashboard in the form of a story and users can filter data based on their requirements in the dashboard.

After the final predicted weekly sales are exported in the 'sampleSubmissionFinal' file, the id column is split to separate the store, department, and date information into different columns through Power BI data transformations (as shown in the figures below).



Id	Weekly_Sales
1_1_2012-11-02	30676.883
1_1_2012-11-09	17655.227

Figure 37. Predicted Sales Submission File Snippet



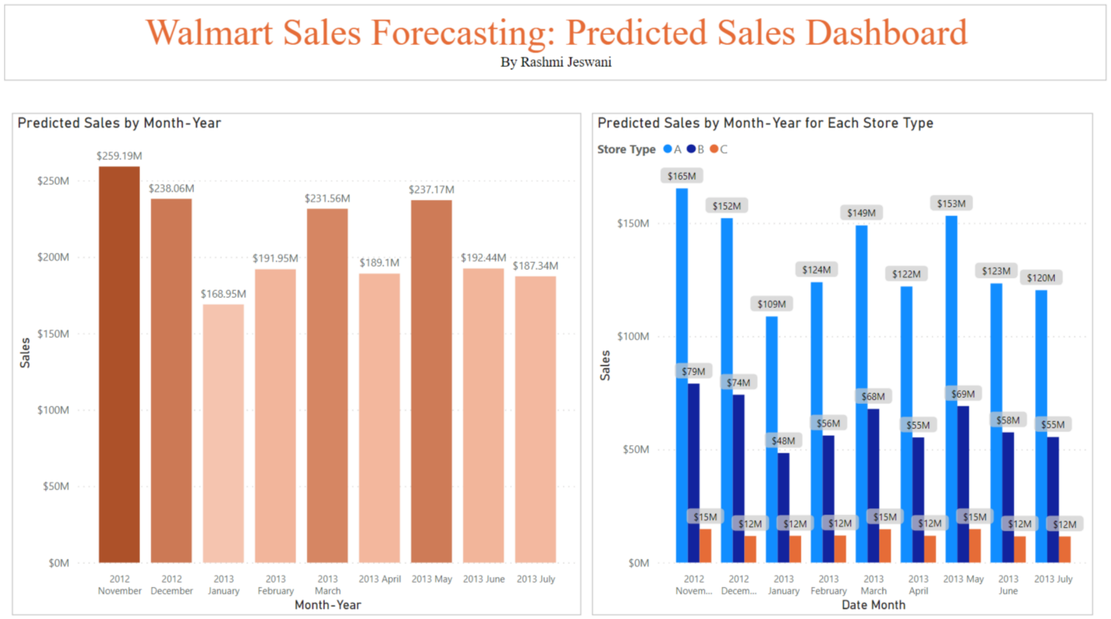
Weekly_Sales	Store	Department	Date
139372.69	13	1	Friday, April 19, 2013
74297.45	13	2	Friday, April 19, 2013
18459.602	13	3	Friday, April 19, 2013
43543.137	13	4	Friday, April 19, 2013
46418.902	13	5	Friday, April 19, 2013
5870.007	13	6	Friday, April 19, 2013
59392.77	13	7	Friday, April 19, 2013
33762.1	13	8	Friday, April 19, 2013
38689.996	13	9	Friday, April 19, 2013
26086.871	13	10	Friday, April 19, 2013

Figure 38. Power BI Data Transformation

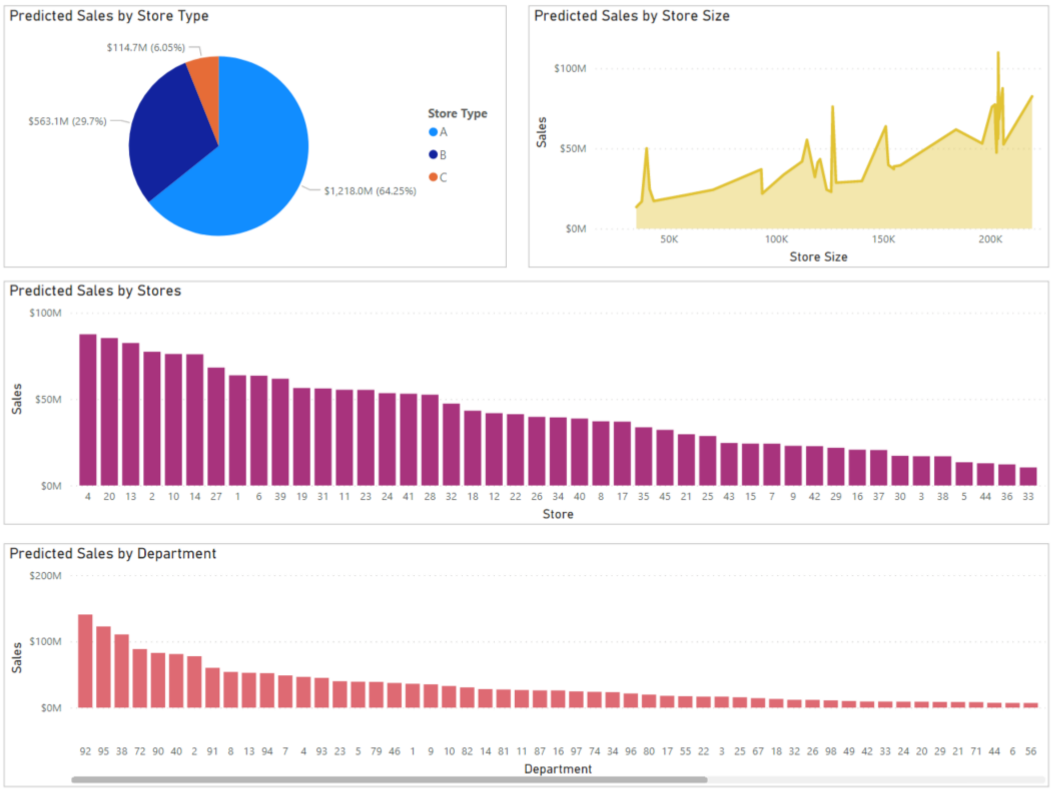
This file is then merged with the 'stores' file that contains information about the type and size of the store as well as holiday information. All these columns will be used to create several visualizations that track weekly predicted sales for various stores and departments, sales based on store size and type, etc. The dashboard also provides detailed information about stores and departments that generate the highest revenue and their respective store types. The PDF file contains brief information about all the visualizations created in the dashboard.

The dashboard can be found in the final submitted folder. If a user does not have access to Power BI, a PDF export of the entire dashboard is included along with the .pbix file that contains all of the created visualizations and reports in the dashboard.

Some views of the dashboard created are included below:



(a) Dashboard 1


Predicted Sales by Store Type
$114.7M (6.05%)
$563.1M (29.7%)
$1,218.0M (64.25%)
Store Type
● A
● B
● C
Predicted Sales by Store Size
Sales
Store Size
Predicted Sales by Stores
Sales
Store
Predicted Sales by Department
Sales
Department

(b) Dashboard 2

Figure 39. Power BI Dashboard Views

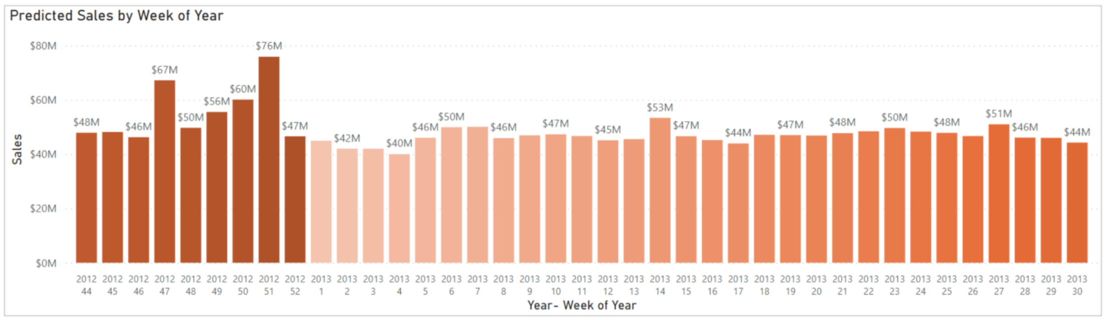


Figure 40. Power BI Dashboard View

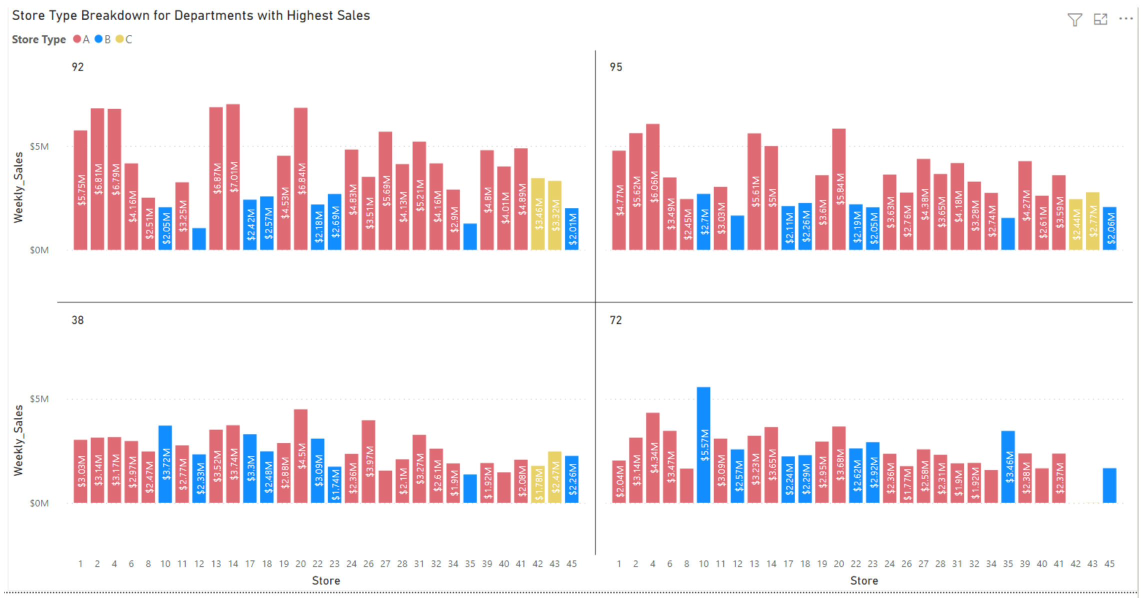


Figure 41. Breakdown of the Highest Revenue Generating Departments

Based on the visualizations created based on the predicted sales, the following observations can be made:

- Sales still seem to be the highest during the holiday season (in the months of November and December)
- Store size is still a great factor that affects sales; the bigger the store, the higher the sales. Store A still has the highest sales, followed by stores B and C
- Stores 4, 14, and 20 are the three stores with the highest sales; similar to this, other than store 14, stores 4 and 20 still have the highest predicted sales

- Departments 92, 95. 38, and 72 still have the highest sales for all three store types as per figure 41

5 Conclusion

5.1 Overall Results

The main purpose of this study was to predict Walmart's sales based on the available historic data and identify whether factors like temperature, unemployment, fuel prices, etc affect the weekly sales of particular stores under study. This study also aims to understand whether sales are relatively higher during holidays like Christmas and Thanksgiving than normal days so that stores can work on creating promotional offers that increase sales and generate higher revenue.

As observed through the exploratory data analysis, store size and holidays have a direct relationship with high Walmart sales. It was also observed that out of all the store types, Type A stores gathered the most sales for Walmart. Additionally, departments 92, 95, 38, and 72 accumulate the most sales for Walmart stores across all three store types; for all of the 45 stores, the presence of these departments in a store ensures higher sales.

Pertaining to the specific factors provided in the study (temperature, unemployment, CPI, and fuel price), it was observed that sales do tend to go up slightly during favorable climate conditions as well as when the prices of fuel are adequate. However, it is difficult to make a strong claim about this assumption considering the limited scope of the training dataset provided as part of this study. By the observations in the exploratory data analysis, sales also tend to be relatively higher when the unemployment level is lower. Additionally, with the dataset provided for this study, there does not seem to be a relationship between sales and the CPI index. Again, it is hard to make a substantial claim about these findings without the presence of a larger training dataset with additional information available.

Interaction effects were studied as part of the linear regression model to identify if a combination of different factors could influence the weekly sales for Walmart. This was necessary because of the presence of a high number of predictor variables in the dataset. While the interaction effects were tested on a combination of significant variables, a statistically significant relationship was only observed between the independent variables of temperature, CPI and unemployment, and weekly sales (predictor variable). However, this is not definite because of the limitation of training data.

Relationships between independent and target variables were tried to be identified through EDA components like the correlation matrix and scatter plots, feature importance plots created as part of the random forest and gradient boosting models as well as the interaction effects. It was discovered that, although, there were no significant relationships between weekly sales and factors like temperature, fuel price, store size, department, etc. in the correlation matrix (Figure 22), some significant relationships were observed between weekly sales and store size and department in the feature importance plots created as part of the gradient boosting and random forest models. Considering that the performance of both these models was significantly better than the performance of the regression models, it can be concluded that a non-linear statistically significant relationship exists between these independent and target variables.

Finally, the tuned Gradient Boosting model, with the lowest WMAE score, is the main model used to create the final predictions for this study. These predictions can be found in the 'sampleSubmissionFinal.csv' file and a visualized report of the outcome can be seen in the Power BI dashboard.

5.2 Limitations

A huge constraint of this study is the lack of sales history data available for analysis. The data for the analysis only comes from a limited number of Walmart stores between the years 2010 and 2013. Because of this limited past history data, models cannot be trained as efficiently to give accurate results and predictions. Because of this lack of availability, it is harder to train and tune models as an over-constrained model might reduce the accuracy of the model. An appropriate amount of training data is required to efficiently train the model and draw useful insights.

Additionally, the models created have been developed based on certain preset assumptions and business conditions; it is harder to predict the effects of certain economic, political, or social policies on the sales recorded by the organization. Also, it is tough to predict how the consumer buying behavior changes over the years or how the policies laid down by the management might affect the company's revenue; these factors can have a direct impact on Walmart sales and it is necessary to constantly study the market trends and compare them with existing performance to create better policies and techniques for increased profits.

5.3 Future Work

With growing technology and increasing consumer demand, Walmart can shift its focus on the e-commerce aspects of the business. Taking inspiration from Amazon's business model, Walmart can grow its online retail business massively and gather huge profits. With already established stores and warehouses, it is easier for the organization to create a nationwide reach, limiting the presence of physical stores and helping their customers save on fuel costs by delivering goods at their doorstep. It also makes it a lot easier to identify consumer buying patterns.

An important aspect of this study is also to try and understand customer buying behavior based on regional and departmental sales. This customer segmentation can help the organization in creating and communicating targeted messages for customers belonging to a particular region, establishing better customer relationships, focusing on profitable regions, and identifying ways to improve products and services in specific regions or for specific customers.

Another aspect that would be worth exploring with this study is identifying trends with sales for each of the stores and predicting future trends based on the available sales data. Time series forecasting can be utilized (ARMA and ARIMA modeling) to predict future sales for each of the stores and their respective departments.

References

Bakshi, C. (2020). Random forest regression. https://levelup.gitconnected.com/random-forest-regression-209c0f354c84

Bari, A., Chaouchi, M., & Jung, T. (n.d.). How to utilize linear regressions in predictive analytics. https://www.dummies.com/programming/big-data/data-science/how-to-utilize-linear-regressions-in-predictive-analytics/

Baum, D. (2011). How higher gas prices affect consumer behavior. https://www.sciencedaily.com/releases/2011/05/110512132426.htm

Brownlee, J. (2016). Feature importance and feature selection with xgboost in python. https://machinelearningmastery.com/feature-importance-and-feature-selection-with-xgboost-in-python/

Chouksey, P., & Chauhan, A. S. (2017). A review of weather data analytics using big data. *International Journal of Advanced Research in Computer and Communication Engineering*, *6*. https://doi.org/https://ijarcce.com/upload/2017/january-17/IJARCCE%2072.pdf

Crown, M. (2016). Weekly sales forecasts using non-seasonal arima models. http://mxcrown.com/walmart-sales-forecasting/

Editor, M. B. (2013). Regression analysis: How do i interpret r-squared and assess the goodness-of-fit? https://blog.minitab.com/en/adventures-in-statistics-2/regression-analysis-how-do-i-interpret-r-squared-and-assess-the-goodness-of-fit

Ellis, L. (2019). Simple eda in r with inspectdf. https://www.r-bloggers.com/2019/05/part-2-simple-eda-in-r-with-inspectdf/

Frost, J. (2021). Regression coefficients- statistics by jim. https://statisticsbyjim.com/glossary/regression-coefficient/

Glen, S. (2016). Elementary statistics for the rest of us. https://www.statisticshowto.com/correlation-matrix/

Guide, U. B. A. R. P. (n.d.). Gradient boosting machines. http://uc-r.github.io/gbm_regression

Harsoor, A. S., & Patil, A. (2015). Forecast of sales of walmart store using big data applications. *International Journal of Research in Engineering and Technology eIS*, *04*, 51–59. https://doi.org/https://ijret.org/volumes/2015v04/i06/IJRET20150406008.pdf

Jaccard, J., & Turrisi, R. (2018). *Interaction effect in multiple regression second edition*. Sage Publications, Thousand Oaks CA.

Jain, A. (2016). Complete machine learning guide to parameter tuning in gradient boosting (gbm) in python. https://www.analyticsvidhya.com/blog/2016/02/complete-guide-parameter-tuning-gradient-boosting-gbm-python/

JJ. (2016). Mae and rmse — which metric is better? https://medium.com/human-in-a-machine-world/mae-and-rmse-which-metric-is-better-e60ac3bde13d

Johnson, D. (2021). Correlation in r: Pearson and spearman correlation matrix. https://www.guru99.com/r-pearson-spearman-correlation.html

kassambara. (2018). Interaction effect in multiple regression: Essentials statistical tools for high-throughput data analysis (sthda). http://www.sthda.com/english/articles/40-regression-analysis/164-interaction-effect-in-multiple-regression-essentials/

Keboola. (2020). The ultimate guide to random forest regression. https://www.keboola.com/blog/random-forest-regression

Kedia, J., Nguyen, G. H., Pasteur, R. D., Snyder, R., & III, R. W. (2013). Sales forecasting using regression and artificial neural networks. https://doi.org/https://www.researchgate.net/publication/280742365_Sales_Forecasting_Using_Regression_and_Artificial_Neural_Networks

Kumar, A. (2020). Lasso regression explained with python example. https://vitalflux.com/lasso-ridge-regression-explained-with-python-example/

Lingjun, H., Levine, R. A., Fan, J., Beemer, J., & Stronach, J. (2018). Random forest as a predictive analytics alternative to regression in institutional research. *Practical Assessment, Research, and Evaluation*, *23*. https://doi.org/https://doi.org/10.7275/1wpr-m024

Myrianthous, G. (n.d.). Training vs testing vs validation sets. https://towardsdatascience.com/training-vs-testing-vs-validation-sets-a44bed52a0e1

Panchotia, R. (2020). Predictive modelling using linear regression. https://medium.com/swlh/predictive-modelling-using-linear-regression-e0e399dc4745

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, B., V.and Thirion, Grisel, O., Blondel, M., Prettenhofer, R., P.and Weiss, Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Sklearn.preprocessing.minmaxscaler. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html

Płoński, P. (2020). Random forest feature importance computed in 3 ways with python. https://mljar.com/blog/feature-importance-in-random-forest/

Prabhakaran, S. (2016). Linear regression: R-statistics.co. http://r-statistics.co/Linear-Regression.html

Rushworth, A. (n.d.). Numeric column summaries and visualisations. https://alastairrushworth.github.io/inspectdf/articles/pkgdown/inspect_num_examples.html

Singh, D. (2019). Linear, lasso, and ridge regression with r. https://www.pluralsight.com/guides/linear-lasso-and-ridge-regression-with-r

STATISTICS, U. B. O. L. (n.d.). Consumer price index. https://www.bls.gov/cpi/

Sullivan, J. (2019). Data cleaning with r and the tidyverse: Detecting missing values. https://towardsdatascience.com/data-cleaning-with-r-and-the-tidyverse-detecting-missing-values-ea23c519bc62

Szabo, B. (2020). How to create a seaborn correlation heatmap in python? https://medium.com/@szabo.bibor/how-to-create-a-seaborn-correlation-heatmap-in-python-834c0686b88e

UNSW. (2020). Descriptive, predictive and prescriptive analytics: What are the differences? https://studyonline.unsw.edu.au/blog/descriptive-predictive-prescriptive-analytics

Vashisht, R. (2021). When to perform a feature scaling? https://www.atoti.io/when-to-perform-a-feature-scaling/

Walmart recruiting - store sales forecasting. (2014). https://www.kaggle.com/c/walmart-recruiting-store-sales-forecasting/data

Walmart's sales data analysis - a big data analytics perspective. (2017). https://doi.org/10.1109/APWConCSE.2017.00028

Wikipedia, t. f. e. (n.d.). Walmart. https://en.wikipedia.org/w/index.php?title=Walmart&oldid=1001006854

Yu, C. H., Lee, H. S., Lara, E., & Gan, S. (2018). The ensemble and model comparison approaches for big data analytics in social sciences. https://scholarworks.umass.edu/pare/vol23/iss1/17

6 Appendix A: Python Code Exploratory Data Analysis

```
# Rashmi Jeswani: Capstone
# Walmart Sales Forecasting
# EDA


import pip


def install(package):
    if hasattr(pip, 'main'):
        pip.main(['install', package])
    else:
        pip._internal.main(['install', package])


install('pandas')
install('seaborn')
install('plotly')
install('jovian')
install('opendatasets')
install('numpy')
install('matplotlib')
install('inline')
install('zipFile')

# Installing the required packages

import numpy as np
import pandas as pd
import matplotlib
import matplotlib.style as style
import seaborn as sns
import opendatasets as od
from matplotlib import pyplot as plt
```

```python
import plotly.express as px
import plotly.graph_objs as go
from plotly.subplots import make_subplots

# Reading the datasets

train = pd.read_csv('train.csv')
stores = pd.read_csv('stores.csv')
features = pd.read_csv('features.csv')
test = pd.read_csv('test.csv')

# Joining the train and test datatsets with features and stores datasets

merge_train = train.merge(stores, how='left').merge(features,
                                                    how='left')
merge_test = test.merge(stores, how='left').merge(features,
                                                  how='left')


# Extracting Date, Month, Quarter and Week columns from the week column

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


split_date(merge_train)
split_date(merge_test)

# EDA: Initial

# Average Store & Department Sales

averagweeklysales = merge_train.groupby('Type')['Weekly_Sales'].mean().to_dict()
```

```
df = pd.DataFrame(list(averagweeklysales.items()), columns=['Store_Type', 'AvgSales'

fig1 = px.bar(df,
              x="Store_Type",
              y="AvgSales",
              title="Average Sales for Each Store")

fig1.show()

departmentsales = merge_train.groupby('Dept')['Weekly_Sales'].mean().sort_values(asc

fig2 = px.bar(departmentsales,
              x=departmentsales.values,
              y=departmentsales.index,
              title="Average Sales for Each Department",
              color_discrete_sequence=["#114D77"], orientation='h',
              labels={'x': 'Average Sales', 'y': 'Department'})

fig2.update_yaxes(tick0=1, dtick=10)

fig2.show()

########################################################
# Department: Alternative Vis #

departament = train.groupby(['Dept']).agg({'Weekly_Sales': ['mean', 'median']})

plt.figure(figsize=(20, 7))
plt.bar(departament.index, departament['Weekly_Sales']['mean'])
plt.xticks(np.arange(1, 100, step=2))
plt.ylabel('Week Sales', fontsize=16)
plt.xlabel('Departament', fontsize=16)
plt.show()

#########################################################

# Average Sales for Each month of the year
```

```
sales_2010 = merge_train[(merge_train.Year == 2010)].groupby('Month')['Weekly_Sales'
df_2010 = pd.DataFrame(list(sales_2010.items()), columns=['Month', 'AverageSales2010

sales_2011 = merge_train[(merge_train.Year == 2011)].groupby('Month')['Weekly_Sales'
df_2011 = pd.DataFrame(list(sales_2011.items()), columns=['Month', 'AverageSales2011

sales_2012 = merge_train[(merge_train.Year == 2012)].groupby('Month')['Weekly_Sales'
df_2012 = pd.DataFrame(list(sales_2012.items()), columns=['Month', 'AverageSales2012

monthly_merge = df_2010.merge(df_2011, how='right', on='Month').merge(df_2012, how='

trace1 = go.Bar(
    x=df_2010.Month,
    y=df_2010.AverageSales2010,
    name="AverageSales2010")

trace2 = go.Bar(
    x=df_2011.Month,
    y=df_2011.AverageSales2011,
    name="AverageSales2011")

trace3 = go.Bar(
    x=df_2012.Month,
    y=df_2012.AverageSales2012,
    name="AverageSales2012")

data = [trace1, trace2, trace3]

layout = go.Layout(barmode="group",
                   xaxis_title="Month",
                   yaxis_title="Monthly Avg Sales")

fig3 = go.Figure(data=data, layout=layout)

fig3.show()

# Average Week Over Week Sales
```

```
weeklysales2010 = merge_train[merge_train.Year == 2010].groupby('WeekOfYear')['Weekl
weeklysales2011 = merge_train[merge_train.Year == 2011].groupby('WeekOfYear')['Weekl
weeklysales2012 = merge_train[merge_train.Year == 2012].groupby('WeekOfYear')['Weekl

plt.plot(weeklysales2010.index, weeklysales2010.values)
plt.plot(weeklysales2011.index, weeklysales2011.values)
plt.plot(weeklysales2012.index, weeklysales2012.values)

plt.xticks(np.arange(1, 53, step=1))
plt.xlabel('Week', fontsize=10, labelpad=20)
plt.ylabel('Sales', fontsize=10, labelpad=20)

plt.title("Week Over Week Sales for Each Year")
plt.legend(['2010', '2011', '2012'])
plt.show()

# Average Sales for Stores

store_sales = merge_train.groupby('Store')['Weekly_Sales'].mean().sort_values(ascend

fig4 = px.bar(store_sales,
              x=store_sales.index,
              y=store_sales.values,
              title="Average Sales for Each Store",
              labels={'x': 'Stores', 'y': 'Average Sales'})

fig4.update_xaxes(tick0=1, dtick=1)

fig4.show()

# Impact of Size of Store on Sales

sns.scatterplot(x=merge_train.Size, y=merge_train.Weekly_Sales, hue=merge_train.Type

plt.xlabel('Size')
plt.ylabel('Sales')
plt.title('Impact of Size of Store on Sales')
plt.show()
```

```
# Week of Year Sales by Store Type

sns.scatterplot(x=merge_train.WeekOfYear, y=merge_train.Weekly_Sales, hue=merge_trai

plt.xticks(np.arange(1, 53, step=4))
plt.xlabel('Week of Year')
plt.ylabel('Sales')
plt.title('Week of Year Sales by Store Type')
plt.show()

# Impact of Temperature on Sales

sns.scatterplot(x=merge_train.Temperature, y=merge_train.Weekly_Sales, hue=merge_tra
plt.xlabel('Temperature', fontsize=10, labelpad=20)
plt.ylabel('Sales', fontsize=10, labelpad=20)
plt.title('Impact of Temperature on Sales')
plt.show()

# Impact of Unemployment on Sales

sns.scatterplot(x=merge_train.Unemployment, y=merge_train.Weekly_Sales, hue=merge_tr

plt.xlabel('Unemployment', fontsize=10, labelpad=20)
plt.ylabel('Sales', fontsize=10, labelpad=20)
plt.title('Impact of Unemployment on Sales')
plt.show()

# Impact of CPI on Sales

sns.scatterplot(x=merge_train.CPI, y=merge_train.Weekly_Sales, hue=merge_train.Type,

plt.xlabel('CPI', fontsize=10, labelpad=20)
plt.ylabel('Sales', fontsize=10, labelpad=20)
plt.title('Impact of CPI on Sales')
plt.show()

# Impact of Fuel Price on Sales
```

```
sns.scatterplot(x=merge_train.Fuel_Price, y=merge_train.Weekly_Sales, hue=merge_trai

plt.xlabel('Fuel Price', fontsize=10, labelpad=20)
plt.ylabel('Sales', fontsize=10, labelpad=20)
plt.title('Impact of Fuel Price on Sales')
plt.show()

# Holiday VS non-Holiday Sales

sales = merge_train.groupby('IsHoliday')['Weekly_Sales'].mean()
counts = merge_train.IsHoliday.value_counts()

fig5 = make_subplots(rows=1, cols=2, subplot_titles=("Holiday_Non-Holiday Sales", "H
fig5.add_trace(go.Bar(x=sales.values, y=sales.index, orientation='h', ), 1, 1)
fig5.add_trace(go.Bar(x=counts.values, y=counts.index, orientation='h', ), 1, 2)
fig5.show()

# Correlation Matrix

storetype_values = {'A': 3, 'B': 2, 'C': 1}
merge_train['Type_Numeric'] = merge_train.Type.map(storetype_values)
merge_test['Type_Numeric'] = merge_test.Type.map(storetype_values)
plt.figure(figsize=(30, 15))
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)

sns.heatmap(merge_train.corr(), cmap='Blues', annot=True, annot_kws={'size': 10})
plt.title('Correlation Matrix')
plt.show()
```

7 Appendix B: Python Code Data Cleaning and Modeling

```
# Rashmi Jeswani: Capstone
# Walmart Sales Forecasting
# Data Cleaning and Modeling


import pip


def install(package):
    if hasattr(pip, 'main'):
        pip.main(['install', package])
    else:
        pip._internal.main(['install', package])


install('pandas')
install('seaborn')
install('plotly')
install('jovian')
install('opendatasets')
install('numpy')
install('matplotlib')
install('inline')
install('zipFile')
install('scikit-learn')
install('xgboost')

# Installing the required packages

import numpy as np
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Lasso
from xgboost import XGBRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split

# Reading the datasets

train = pd.read_csv('train.csv')
stores = pd.read_csv('stores.csv')
features = pd.read_csv('features.csv')
test = pd.read_csv('test.csv')

# Joining the train and test datasets with features and stores datasets

merge_train = train.merge(stores, how='left').merge(features, how='left')
merge_test = test.merge(stores, how='left').merge(features, how='left')


# Extracting Date, Month, Quarter and Week columns from the week column

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


split_date(merge_train)
split_date(merge_test)

# DATA CLEANING & TRANSFORMATION

# Checking the NaN percentage

df = merge_train.isnull().mean() * 100
```

```
# Dropping columns that are not required for modeling

merge_train = merge_train.drop(['Date', 'Temperature', 'Fuel_Price', 'Type', 'MarkDo
                                'MarkDown4', 'MarkDown5', 'CPI', 'Unemployment', 'Mo

merge_test = merge_test.drop(['Date', 'Temperature', 'Fuel_Price', 'Type', 'MarkDown
                              'MarkDown4', 'MarkDown5', 'CPI', 'Unemployment', 'Mont

# Identifying the input and target column (Weekly Sales)

input_column = merge_train.columns.to_list()
input_column.remove('Weekly_Sales')
target_column = 'Weekly_Sales'

inputs = merge_train[input_column].copy()
targets = merge_train[target_column].copy()

# Feature Scaling (Min Max Scaler)

minmax_scaler = MinMaxScaler().fit(merge_train[input_column])

inputs[input_column] = minmax_scaler.transform(inputs[input_column])
merge_test[input_column] = minmax_scaler.transform(merge_test[input_column])

# Training & Validation Sets

train_inputs, val_inputs, train_targets, val_targets = train_test_split(inputs, targ


# WMAE Function Create
def WMAE(df, targets, predictions):
    weights = df.IsHoliday.apply(lambda x: 5 if x else 1)

    return np.round(np.sum(weights * abs(targets - predictions)) / (np.sum(weights))


# LINEAR REGRESSION
# Creating model
lm = LinearRegression().fit(train_inputs, train_targets)

# Generate predictions (training data)
pred_train = lm.predict(train_inputs)

# WMAE for training dataset
WMAE_train = WMAE(train_inputs, train_targets, pred_train)

# Generate predictions (validation data)
pred_valid = lm.predict(val_inputs)

# WMAE for Validation dataset
WMAE_valid = WMAE(val_inputs, val_targets, pred_valid)
print('WMAE Validation Dataset: {}.'.format(WMAE_valid))

###########################################################################

# LASSO REGRESSION

# Create an instance of Lasso Regression implementation
lasso = Lasso(alpha=1.0)

# Creating model
lr = lasso.fit(train_inputs, train_targets)

# Generate prediction (training data)
pred_train2 = lr.predict(train_inputs)

# WMAE for training dataset
WMAE_train2 = WMAE(train_inputs, train_targets, pred_train2)

# Generate predictions (validation data)
pred_valid2 = lr.predict(val_inputs)

# WMAE for validation data
WMAE_valid2 = WMAE(val_inputs, val_targets, pred_valid2)
```

```
print('WMAE Validation Datatset: {}.'.format(WMAE_valid2))

######################################################################

# GRADIENT BOOSTING MACHINE

# Creating model
gbm = XGBRegressor(random_state=42, n_jobs=-1)

# Fitting the inputs and targets for the model
gbm.fit(train_inputs, train_targets)

# Generate prediction (training data)
gbm_pred_train = gbm.predict(train_inputs)

# WMAE for training dataset
WMAE_train_gbm = WMAE(train_inputs, train_targets, gbm_pred_train)

# Generate predictions (validation data)
gbm_pred_valid = gbm.predict(val_inputs)

# WMAE for validation dataset
WMAE_valid_gbm = WMAE(val_inputs, val_targets, gbm_pred_valid)
print('WMAE Validation Dataset: {}.'.format(WMAE_valid_gbm))

# GBM: Feature Importance

feature_imp = pd.DataFrame({
    'feature': train_inputs.columns,
    'importance': gbm.feature_importances_
})

plt.title('Feature Importance: GBM')
sns.barplot(data=feature_imp, x='importance', y='feature')
plt.show()


# GBM: Tuning Parameters
```

```
def test_parameters_xgb(**params):
    model = XGBRegressor(random_state=42, n_jobs=-1, **params).fit(train_inputs, tra
    train_wmae = WMAE(train_inputs, train_targets, model.predict(train_inputs))
    val_wmae = WMAE(val_inputs, val_targets, model.predict(val_inputs))
    return train_wmae, val_wmae


def test_parameters(param_name, param_values):
    train_errors, val_errors = [], []
    for value in param_values:
        params = {param_name: value}
        train_wmae, val_wmae = test_parameters_xgb(**params)
        train_errors.append(train_wmae)
        val_errors.append(val_wmae)


# Recreating the GBM model with tuned parameters

# Create the model
gbm = XGBRegressor(random_state=42, n_jobs=-1, n_estimators=400, max_depth=15, learn

# Fit the model
gbm.fit(train_inputs, train_targets)

gbm_train_preds = gbm.predict(train_inputs)

# Compute WMAE on training data
gbm_train_wmae = WMAE(train_inputs, train_targets, gbm_train_preds)

gbm_val_preds = gbm.predict(val_inputs)

# Compute WMAE on validation data
gbm_val_wmae = WMAE(val_inputs, val_targets, gbm_val_preds)
print('WMAE Validation Datatset (Tuned Parameters): {}'.format(gbm_val_wmae))

##################################################################
```

```
# RANDOM FOREST

# Create the model
rf1 = RandomForestRegressor(n_jobs=-1, random_state=42)

# Fit the model
rf1.fit(train_inputs, train_targets)

rf1_train_preds = rf1.predict(train_inputs)

# Compute WMAE on training data
rf1_train_wmae = WMAE(train_inputs, train_targets, rf1_train_preds)

rf1_val_preds = rf1.predict(val_inputs)

# Compute WMAE on validation data
rf1_val_wmae = WMAE(val_inputs, val_targets, rf1_val_preds)
print('WMAE Validation Datatset: {}.'.format(rf1_val_wmae))

# Random Forest: Feature Importance

importance_df = pd.DataFrame({
    'feature': train_inputs.columns,
    'importance': rf1.feature_importances_})

plt.title('Feature Importance: Random Forest')
sns.barplot(data=importance_df, x='importance', y='feature')
plt.show()


# RANDOM FOREST: Tuning Parameters

def test_parameters_rf(**params):
    model = RandomForestRegressor(random_state=42, n_jobs=-1, **params).fit(train_in
    trainWMAE = WMAE(train_inputs, train_targets, model.predict(train_inputs))
    valWMAE = WMAE(val_inputs, val_targets, model.predict(val_inputs))
    return trainWMAE, valWMAE
```

```
def test_parameters_and_plot(param_name, param_values):
    train_errors, val_errors = [], []
    for value in param_values:
        params = {param_name: value}
        trainWMAE, valWMAE = test_parameters_rf(**params)
        train_errors.append(trainWMAE)
        val_errors.append(valWMAE)


# Recreating the Random Forest model with tuned parameters

# Create the model
randomforest1 = RandomForestRegressor(n_jobs=-1, max_depth=30, n_estimators=130, min
                            max_samples=0.99999, max_features=6, random_state=42)

# Fit the model
randomforest1.fit(train_inputs, train_targets)

rf1_train_preds = randomforest1.predict(train_inputs)

# Compute WMAE on training data
rf1_train_wmae = WMAE(train_inputs, train_targets, rf1_train_preds)

rf1_val_preds = rf1.predict(val_inputs)

# Compute WMAE on validation data
rf1_val_wmae = WMAE(val_inputs, val_targets, rf1_val_preds)
print('WMAE Validation Datatset (Tuned Parameters): {}'.format(rf1_val_wmae))


# EXPORTING PREDICTED SALES DATA TO SAMPLE SUBMISSION.CSV FILE

predicted_df = gbm.predict(merge_test)
merge_test['Weekly_Sales'] = predicted_df

# Reading sampleSubmission.csv file and exporting predicted results into the final s
```

```
sampleSubmission = pd.read_csv('sampleSubmission.csv')

sampleSubmission['Weekly_Sales'] = predicted_df
sampleSubmission.to_csv('sampleSubmissionFinal.csv', index=False)
```

8 Appendix C: R Code Exploratory Data Analysis and Modeling

```
#************* CAPSTONE PROJECT 2021 **********************#
#************ RASHMI JESWANI, MS IST **********************#
#**************** EDA and Data Modeling*****************#

##LOAD ALL PACKAGES
# Walmart Data Analysis
'''
install.packages("stringi")
install.packages("zoo")
install.packages("dplyr")
install.packages("ggplot2")
install.packages("tidyverse")
install.packages("CARS")
install.packages("VIF")
install.packages("car")
install.packages("inspectdf")
install.packages("data.table")
install.packages("reshape")
install.packages("forecast")
install.packages("tidyr")
install.packages("glimpse")
install.packages("corrplot")
install.packages("scales")
install.packages("lubridate")
install.packages("caret")
install.packages("glmnet")
install.packages("broom")
install.packages("coefplot")
install.packages("sjPlot")
install.packages("sjmisc")
install.packages("phia")
'''

library(stringi)
```

```r
library(zoo)
library(dplyr)
library(ggplot2)
library(tidyverse)
library(lubridate)
library(CARS)
library(VIF)
library(car)
library(inspectdf)
library(data.table)
library(reshape)
library(forecast)
library(tidyr)
library(corrplot)
library(scales)
library(caret)
library(glmnet)
library(broom)
library(modelr)
library(coefplot)

getwd()
setwd("/Users/nikki09/Desktop/Capstone Research/FINAL/Data")

## READ FILES. BASIC SUMMARY
feature <- read.csv("features.csv")
sales_train <- read.csv("train.csv")
sales_test <- read.csv("test.csv")
stores <- read.csv("stores.csv")

## datatset basic summaries ##
summary(feature)
summary(sales_train)
summary(stores)

############# PACKAGE INSPECTDF EDA ###################
## Data Type for each column in the following datasets
```

```r
inspect_types(feature) %>% show_plot()
inspect_types(sales_train) %>% show_plot()
inspect_types(stores) %>% show_plot()

## Column sizing information in the dataset

inspect_mem(feature) %>% show_plot()
inspect_mem(sales_train) %>% show_plot()
inspect_mem(stores) %>% show_plot()

## Missing values in each column of the dataset

inspect_na(feature) %>% show_plot()
inspect_na(sales_train) %>% show_plot()
inspect_na(stores) %>% show_plot()
inspect_na(sales_test) %>% show_plot()

## numerical distribution in the datasets

inspect_num(feature) %>% show_plot()
inspect_num(sales_train) %>% show_plot()
inspect_num(stores) %>% show_plot()

## categorical dostribution of variables

inspect_cat(feature) %>% show_plot()
inspect_cat(sales_train) %>% show_plot()
inspect_cat(stores) %>% show_plot()

## correlation between columns

inspect_cor(feature) %>% show_plot()
inspect_cor(sales_train) %>% show_plot()
inspect_cor(stores) %>% show_plot()

## Data Modeling ##

## JOINING THE TRAIN AND TEST DATASET WITH STORES & FEATURES
```

```
merge_train <- merge(x=sales_train, y=stores, by='Store')
merge_train <- merge(x=merge_train, y=feature, by=c('Store','Date','IsHoliday'))

merge_test <- merge(x=sales_test, y=stores, by='Store')
merge_test <- merge(x=merge_test, y=feature, by=c('Store','Date','IsHoliday'))

## Creating the Year, Month, Week of Year, Day, Quarter columns from Date Column

merge_train <- merge_train %>%
         dplyr::mutate(Quarter = quarter(merge_train$Date),
          Year = year(merge_train$Date),
          Month =  month(merge_train$Date),
          Day = day(merge_train$Date),
          Date = date(merge_train$Date),
          WeekofYear = week(merge_train$Date))

merge_test <- merge_test %>%
  dplyr::mutate(Quarter = quarter(merge_test$Date),
                Year = year(merge_test$Date),
                Month =  month(merge_test$Date),
                Day = day(merge_test$Date),
                Date = date(merge_test$Date),
                WeekofYear = week(merge_test$Date))

## DROPPING SOME COLUMNS THAT ARE NOT REQUIRED FOR MODELING

merge_train <- select(merge_train, -c(MarkDown1, MarkDown2, MarkDown3,
MarkDown4, MarkDown5, Quarter, Year, Month, Day,  WeekofYear))
merge_test <- select(merge_test, -c(MarkDown1, MarkDown2, MarkDown3,
MarkDown4, MarkDown5, Quarter, Year, Month, Day,  WeekofYear))

############## LINEAR REGRESSION

linearmodel <- lm(Weekly_Sales~., data = merge_train)
summary(linearmodel)
summary(linearmodel)$coefficient
summary(linearmodel)$fstatistic
```

```r
# Residual Standard Error
sigma(linearmodel)/mean(merge_train$Weekly_Sales)

# K-Fold Cross Validation

# Define training control
set.seed(123)
train.control <- trainControl(method = "cv", number = 10)

linearmodel <- train(Weekly_Sales ~., data = merge_train,
method = "lm", trControl = train.control)

print(linearmodel)

############# LASSO REGRESSION
# Getting the independent variable
x_var <- data.matrix(merge_train[, c("Unemployment",
"Temperature", "Type", "Size", "Fuel_Price", "CPI","IsHoliday")])

# Getting the dependent variable
y_var <- merge_train[, "Weekly_Sales"]

# Setting lambda = 1
# cv.glmnet automatically performs cross validation

#perform k-fold cross-validation to find optimal lambda value
cv_model = cv.glmnet(x_var, y_var, alpha = 1)

best_lambda <- cv_model$lambda.min
best_lambda

plot(cv_model)

#find coefficients
best_model <- glmnet(x_var, y_var, alpha = 1, lambda = best_lambda)
coef(best_model)
```

```
# R2 Value for both training and testing dataset

predicted <- predict(best_model, s = best_lambda, newx = x_var)

# Sum of Squares Total and Error
sst <- sum((y_var - mean(y_var))^2)
sse <- sum((predicted - y_var)^2)

# R squared
rsq <- 1 - sse / sst
rsq
# lower R square; inefficient model

##### Plotting Interaction Effects of Regression Models

theme_set(theme_sjplot())

linearmodel <- lm(Weekly_Sales~., data = merge_train)

linearmodel2 <- lm(Weekly_Sales~ Temperature*Fuel_Price*CPI*Unemployment*IsHoliday,
     data = merge_train)
summary(linearmodel2)
```