Boosting Store Sales: Exploring Insights and Predictive Power with XGBoost.

Team

Name	Hawk-id	Email-id
JENITA SHIRLEY DEVADOSS SAMRAJ (lead)	A20513982	Jdevadosssamraj@hawk.iit.edu
VINITHA INAGANTI	A20514310	vinaganti@hawk.iit.edu
KUNDETI NAGA SRI ADITYA	A20520654	nkundeti@hawk.iit.edu
LAKSHMA REDDY BAIRI	A20513302	lbairi@hawk.iit.edu
PRAPUL KUMAR PODILI	A20523662	ppodili@hawk.iit.edu

Illinois Institute of Technology

CSP 571 - Data Preparation and Analysis

Professor: Jawahar Panchal

Table Of Contents

1. Data Preparation
    - 1.1. Merging of Data
    - 1.2. Extracting Date Information
    - 1.3. Columns Information
    - 1.4. Missing Values
2. Exploratory Data Analysis:
    - 2.1. Popularity of Store Types
    - 2.2. Average Sales - Store Type
    - 2.3. Average Monthly Sales - Per Year
    - 2.4. Average Weekly Sales - Per Year
    - 2.5. Average Store Sales
    - 2.6. Holidays Vs Non Holidays Sales
    - 2.7. Relationship: Week of Year vs Sales
    - 2.8. Relationship: Size of Store vs Sales
    - 2.9. Relationship: Temperature vs Sales
    - 2.10. Relationship: Fuel Price vs Sales
    - 2.11. Relationship: CPI vs Sales
    - 2.12. Relationship: Unemployment vs Sales
    - 2.13. Correlation Matrix
3. Data Preparation for Model Training
4. Machine Learning
    - 4.1. Linear Regression
    - 4.2. Ridge Regression
    - 4.3. Decision Tree
    - 4.4. Random Forest
        - 4.4.1. Tuning of Random Forest Hyperparameters
    - 4.5. Gradient Boosting Machine
        - 4.5.1. Tuning of Gradient Boosting Machine Hyperparameters
5. Models Comparison
6. Inferences and Conclusions

1. Data Preparation:The dataset comprises three CSV files:

   train.csv: This file contains 421,570 rows and 5 columns. The information in these columns pertains to a store and includes details such as store ID, department, date, weekly sales, and an indication of whether a given week is a holiday week.

   store.csv: With 45 rows and 3 columns, this file provides information about different stores. The columns include store ID, store type, and store size.

   features.csv: This file, with 8,190 rows and 12 columns, offers additional details about the stores and the regions they are located in. The columns include date, temperature, fuel price, consumer price index, and the unemployment rate for the region. Additionally, there are five columns labeled MarkDown1-5, which represent various promotional activities occurring in different stores.

   In summary, the dataset combines information about sales, stores, and regional factors, making it comprehensive and suitable for analyzing the relationships between weekly sales and different store-related and regional variables.

```
     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
```

```
     Store        Type                Size
 Min.   : 1   Length:45          Min.   : 34875
 1st Qu.:12   Class :character   1st Qu.: 70713
 Median :23   Mode  :character   Median :126512
 Mean   :23                      Mean   :130288
 3rd Qu.:34                      3rd Qu.:202307
 Max.   :45                      Max.   :219622
```

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

1.1 Merging of data and Extracting date Information: Combine the information from three separate data frames into a unified dataframe, and then continue the analysis using this consolidated dataset.

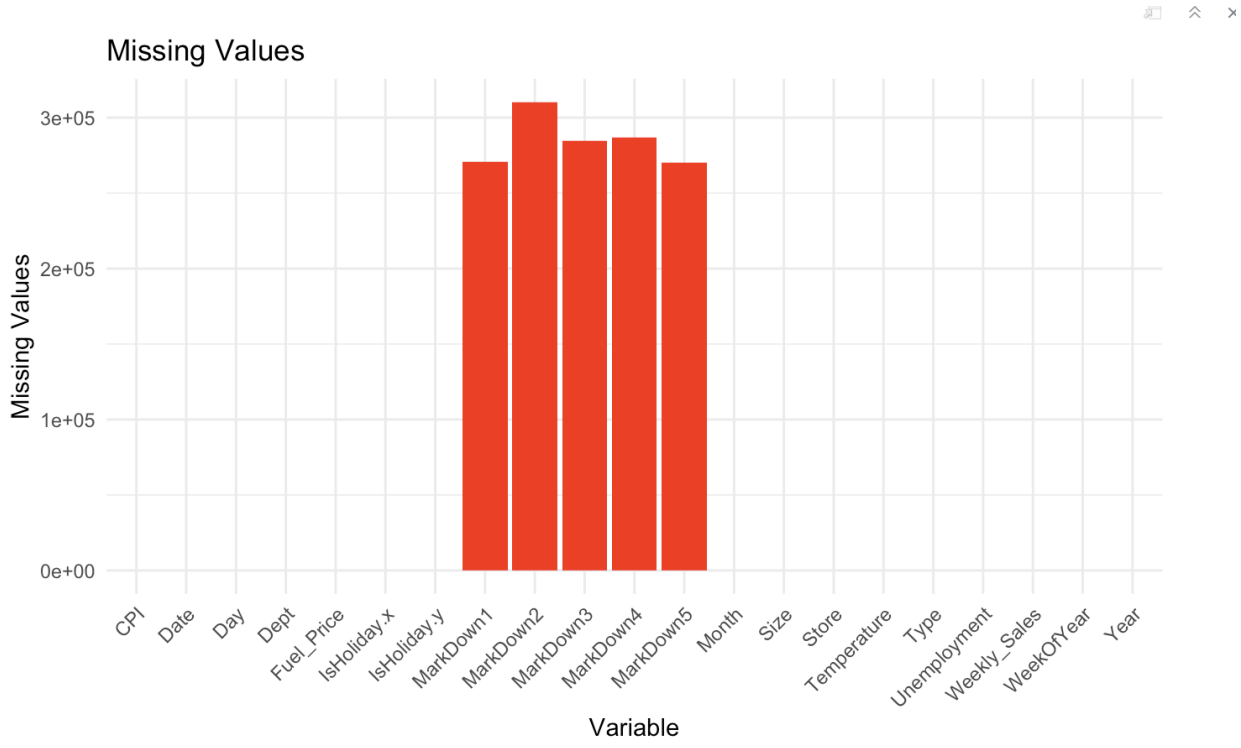
1.2 Extracting Date Information: The sales data is provided for the years 2012 to 2012, with a weekly breakdown.

1.3 Columns Information: To enhance our analysis, let's divide the date column to extract details such as the year, month, and week for a more granular understanding of the temporal patterns in the sales data. This will facilitate a more detailed exploration of how sales vary across different time intervals.

```r
library(dplyr)
library(lubridate)


mergeddf <- traindf %>%
    left_join(storesdf, by = "Store") %>%
    left_join(featuresdf, by = c("Store", "Date"))

testingdf_merged <- testingdf %>%
    left_join(storesdf, by = "Store") %>%
    left_join(featuresdf, by = c("Store", "Date"))

splitdf_date <- function(df) {
    df$Date <- as.Date(df$Date)
    df$Year <- year(df$Date)
    df$Month <- month(df$Date)
    df$Day <- day(df$Date)
    df$WeekOfYear <- week(df$Date)
    return(df)
}

mergeddf <- splitdf_date(mergeddf)
testingdf_merged <- splitdf_date(testingdf_merged)
mergeddf
testingdf_merged
```

1.4: **Missing Values:** With the exception of the Markdown 1 to 5 columns, all other columns in the dataset are complete. The Markdown columns exhibit a significant number of missing values, exceeding 250,000 in each Markdown column. These specific columns pertain to promotional activities conducted at various stores. Notably, the promotional markdowns commenced after November 2011 and are not consistently implemented across all stores at all times. The prevalence of NaN values in these columns is thus understandable.

```{r}
library(ggplot2)

missing_values <- sapply(mergeddf, function(x) sum(is.na(x)))
missing_values_df <- data.frame(Variable = names(missing_values), MissingValues =
as.integer(missing_values))

ggplot(missing_values_df, aes(x = Variable, y = MissingValues)) +
  geom_bar(stat = "identity", fill = "red") +
  theme_minimal() +
  labs(title = "Missing Values", x = "Variable", y = "Missing Values") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))

```



To gain insights into the dataset, we will embark on exploratory data analysis (EDA) to examine the relationship between these Markdown columns and the weekly sales. By doing so, we can assess the impact of these promotional activities on sales and make informed decisions about how to handle the missing values in these columns. This analysis will guide us in determining the relevance and significance of the Markdown data in relation to the overall sales patterns.

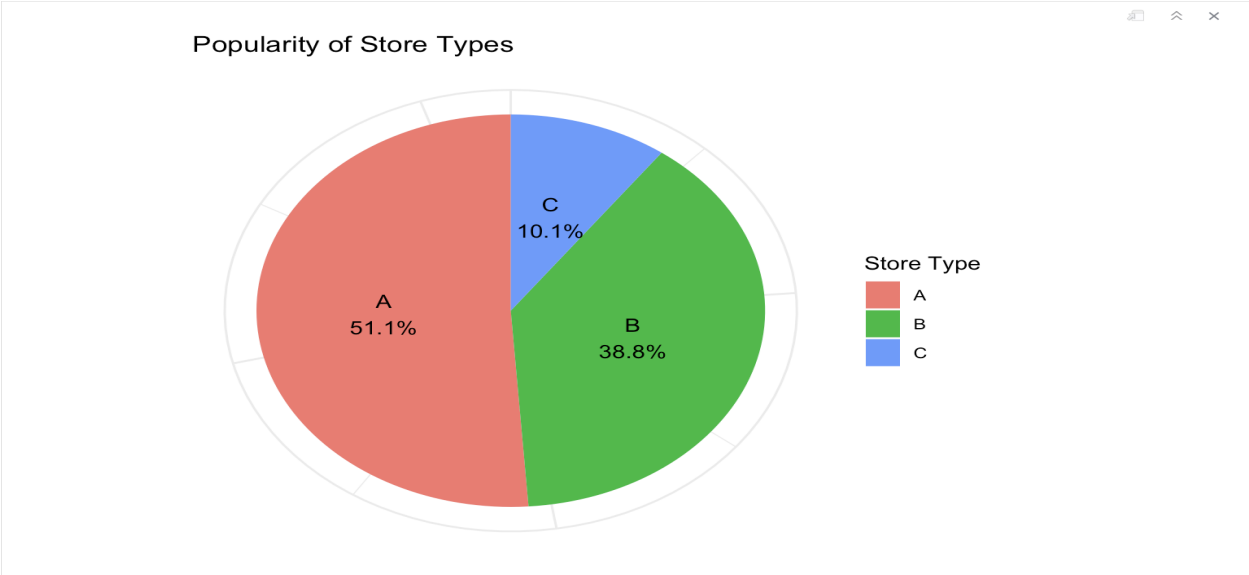
2.Exploratory Data analysis

2.1 Popularity of Store Type

```r
```{r}
library(ggplot2)

typecounts <- table(mergeddf$Type)
typecounts_df <- as.data.frame(typecounts)
typecounts_df$Percentage <- (typecounts_df$Freq / sum(typecounts_df$Freq)) * 100
typecounts_df$Label <- paste0(typecounts_df$Var1, "\n", round(typecounts_df$Percentage, 1), "%")
ggplot(typecounts_df, aes(x = "", y = Freq, fill = Var1, label = Label)) +
  geom_bar(stat = "identity", width = 1) +
  geom_text(position = position_stack(vjust = 0.5)) +
  coord_polar(theta = "y") +
  labs(title = "Popularity of Store Types", x = NULL, y = NULL, fill = "Store Type") +
  theme_minimal() +
  theme(axis.text.x = element_blank())

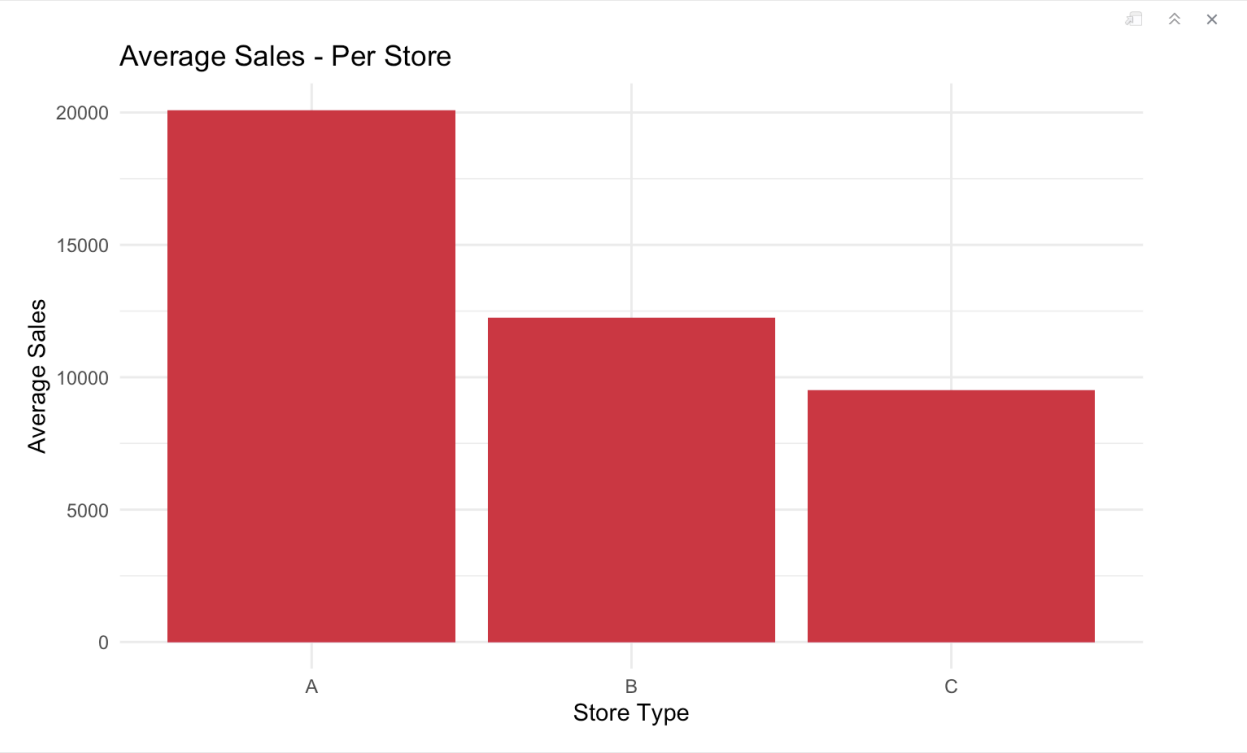

```
```



Observations: Type A stores exhibit higher popularity compared to B and C types.The data analysis reveals that among the store types (A, B, and C), Type A stores stand out as more favored or widely accepted. This conclusion is drawn from the available dataset, which could include factors such as customer footfall, sales figures, or any other metric indicating the level of popularity. Further investigation into the characteristics and performance of each store type may provide a more comprehensive understanding of the factors contributing to the observed popularity disparity.

2.2 Average Sales - Store Type

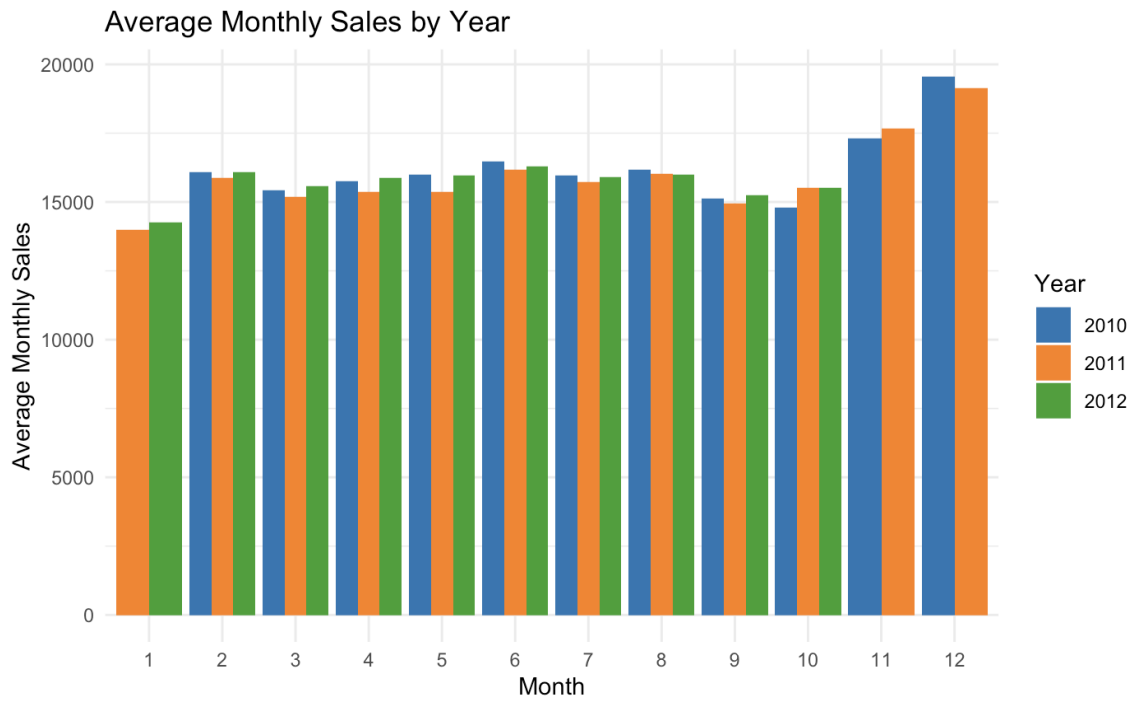
```r
library(ggplot2)
library(dplyr)

avgweeklysales <- mergeddf %>%
  group_by(Type) %>%
  summarize(AvgSales = mean(Weekly_Sales, na.rm = TRUE))

ggplot(avgweeklysales, aes(x = Type, y = AvgSales)) +
  geom_bar(stat = "identity", fill = "#DC143C") +
  labs(title = "Average Sales - Per Store", x = "Store Type", y = "Average Sales") +
  theme_minimal() +
  theme(legend.position = "none")


```



Observation: When it comes to sales, Type A stores outperform the other two types.The analysis of sales data indicates that Type A stores demonstrate a superior performance compared to Type B and Type C stores. This superiority in sales could be attributed to various factors such as location, product assortment, promotional strategies, or other business-related dynamics. Further investigation into the specific aspects driving the sales variations among the store types can provide valuable insights for strategic decision-making and optimization of overall retail performance.

2.3 Average Monthly Sales - Per Year

```r
library(dplyr)
library(ggplot2)
avg_sales_by_year <- mergeddf %>%
  filter(Year %in% c(2010, 2011, 2012)) %>%
  group_by(Year, Month) %>%
  summarize(AvgSales = mean(Weekly_Sales, na.rm = TRUE))
ggplot(avg_sales_by_year, aes(x = factor(Month), y = AvgSales, fill = as.factor(Year))) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Average Monthly Sales by Year",
       x = "Month", y = "Average Monthly Sales",
       fill = "Year") +
  theme_minimal() +
  scale_fill_manual(values = c("2010" = "#1f77b4", "2011" = "#ff7f0e", "2012" = "#2ca02c"))
```

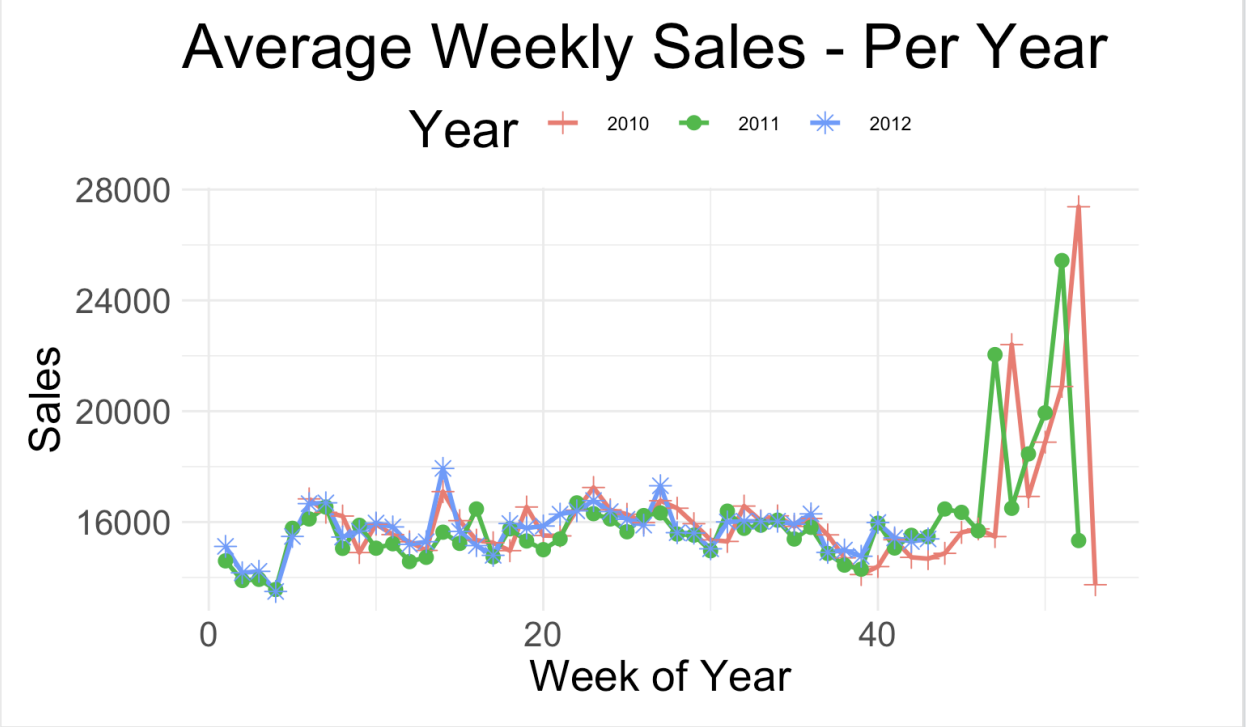


Observations:

- The month of January experienced the lowest sales for both 2011 and 2012, and unfortunately, the weekly sales data for 2010 is not available in the dataset.
- From February to October, consistent weekly sales of around 15,000 were observed across the three years. November and December exhibited the highest sales for 2010 and 2011. However, the sales data for these months in 2012 is not provided in the dataset.
- Analysis of the sales data reveals distinct patterns in monthly sales trends. January consistently emerges as a period of lower sales in both 2011 and 2012. Unfortunately, the lack of weekly sales data for 2010 prevents a comprehensive understanding of that year's January sales.
- In contrast, from February through October, the weekly sales figures exhibit a relatively stable pattern, hovering around the 15,000 mark across the three years. This period of consistent sales suggests a baseline level of demand or a stable market condition during these months.
- A notable trend is observed in November and December, where sales peaked for 2010 and 2011. However, it's important to note the absence of sales data for these months in 2012, limiting the ability to draw conclusions for that specific year. Further exploration and analysis of contributing factors during these peak months could provide valuable insights into the seasonality or specific events influencing sales performance during the holiday season.

2.4 Average Weekly Sales - Per Year

```r
library(dplyr)
library(ggplot2)

avg_weekly_sales <- mergeddf %>%
  filter(Year %in% c(2010, 2011, 2012)) %>%
  group_by(Year, WeekOfYear) %>%
  summarize(AvgSales = mean(Weekly_Sales, na.rm = TRUE))

ggplot(avg_weekly_sales, aes(x = WeekOfYear, y = AvgSales, color = as.factor(Year), group = Year, shape = as.factor(Year))) +
  geom_line(size = 1) +
  geom_point(size = 3) +
  labs(title = "Average Weekly Sales - Per Year",
       x = "Week of Year", y = "Sales",
       color = "Year", shape = "Year") +
  theme_minimal() +
  scale_shape_manual(values = c("2010" = 3, "2011" = 16, "2012" = 8)) +
  theme(legend.position = "top", axis.text = element_text(size = 16), axis.title = element_text(size = 20), title = element_text(size = 24))

```

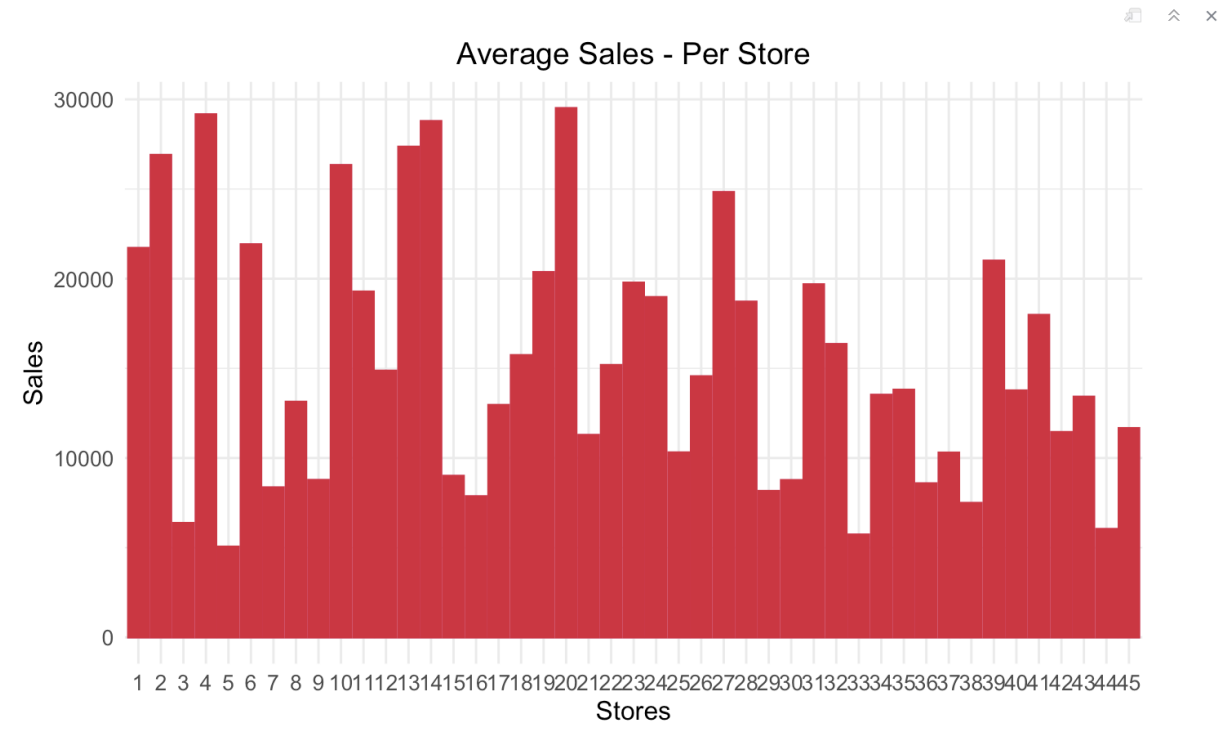


Observations:

- Weekly sales data indicates that, in the years 2010 and 2011, the weeks encompassing Thanksgiving and the week just prior to Christmas consistently experienced the highest sales.
- In 2012, unlike the preceding years, week number 14 stood out with the highest sales. Notably, this spike in sales does not align with any recognized holiday or special event.
- A distinct sales trend emerges when examining weekly data for 2010 and 2011. During these years, the weeks coinciding with the Thanksgiving holiday and the week immediately preceding Christmas consistently boasted the highest sales figures. This aligns with expectations, as these periods typically witness increased consumer spending due to holiday-related activities and gift shopping.
- However, in 2012, a noteworthy deviation from this pattern is observed. Week number 14 in that year recorded the highest sales, even though it doesn't correspond to any recognized holiday or special event. Further analysis is warranted to understand the factors contributing to this anomaly. It could be influenced by unique market dynamics, promotional activities, or other external factors not explicitly tied to holidays. Investigating the specifics of week 14 in 2012 could uncover insights that contribute to a more nuanced understanding of the sales dynamics during that particular period.

2.5 Average Store Sales

```r
library(dplyr)
library(ggplot2)

store_sales <- mergeddf %>%
  group_by(Store) %>%
  summarize(AvgSales = mean(Weekly_Sales, na.rm = TRUE)) %>%
  arrange(AvgSales)

p_bar <- ggplot(store_sales, aes(x = factor(Store, levels = 1:45), y = AvgSales)) +
  geom_bar(stat = "identity", color = "#DC143C", fill = "#DC143C") +
  labs(title = "Average Sales - Per Store",
       x = "Stores", y = "Sales") +
  theme_minimal() +
  scale_x_discrete(breaks = 1:45) +
  theme(legend.position = "none",
        axis.text.x = element_text(angle = 0, hjust = 0.5, size = 10),
        axis.text.y = element_text(size = 10),
        axis.title = element_text(size = 12),
        plot.title = element_text(hjust = 0.5, size = 14))

p_bar
```



Observations:

- Considerable variations in sales are evident across the 45 stores.The extent of these variations is influenced by the specific category of store and the particular week within the given year.
- The analysis highlights substantial differences in sales performance among the 45 stores. The magnitude of these variations is not uniform and is contingent upon two key factors: the category of the store and the specific week within the given year.
- Different categories of stores exhibit diverse sales patterns, suggesting that factors such as store size, location, or other category-specific attributes play a significant role in influencing sales outcomes. Additionally, the specific week within a given year also emerges as a crucial determinant of sales variations. This indicates that certain weeks may witness a surge in consumer activity, promotions, or other factors that impact sales differently across the diverse set of stores. Understanding these dynamics is essential for tailoring strategies and optimizing performance based on the unique characteristics of each store category and the temporal context within a given year.

2. 6 Average Store Sales - Year Wise

```r
```{r}
library(dplyr)
library(ggplot2)
library(gridExtra)


store_sales_2010 <- mergeddf %>% filter(Year == 2010) %>%
  group_by(Store) %>%
  summarize(AvgSales2010 = mean(Weekly_Sales, na.rm = TRUE))

store_sales_2011 <- mergeddf %>% filter(Year == 2011) %>%
  group_by(Store) %>%
  summarize(AvgSales2011 = mean(Weekly_Sales, na.rm = TRUE))

store_sales_2012 <- mergeddf %>% filter(Year == 2012) %>%
  group_by(Store) %>%
  summarize(AvgSales2012 = mean(Weekly_Sales, na.rm = TRUE))

p_2010 <- ggplot(store_sales_2010, aes(x = Store, y = AvgSales2010)) +
  geom_bar(stat = "identity", fill = scales::seq_gradient_pal("blue", "red")(0.7)) +
  labs(title = "Average Store Sales 2010", x = "Store", y = "AvgSales") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 0, hjust = 0.5, size = 10))

p_2011 <- ggplot(store_sales_2011, aes(x = Store, y = AvgSales2011)) +
  geom_bar(stat = "identity", fill = scales::seq_gradient_pal("blue", "red")(0.7)) +
  labs(title = "Average Store Sales 2011", x = "Store", y = "AvgSales") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 0, hjust = 0.5, size = 10))

p_2012 <- ggplot(store_sales_2012, aes(x = Store, y = AvgSales2012)) +
  geom_bar(stat = "identity", fill = scales::seq_gradient_pal("blue", "red")(0.7)) +
  labs(title = "Average Store Sales 2012", x = "Store", y = "AvgSales") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 0, hjust = 0.5, size = 10))

grid.arrange(p_2010, p_2011, p_2012, ncol = 1)

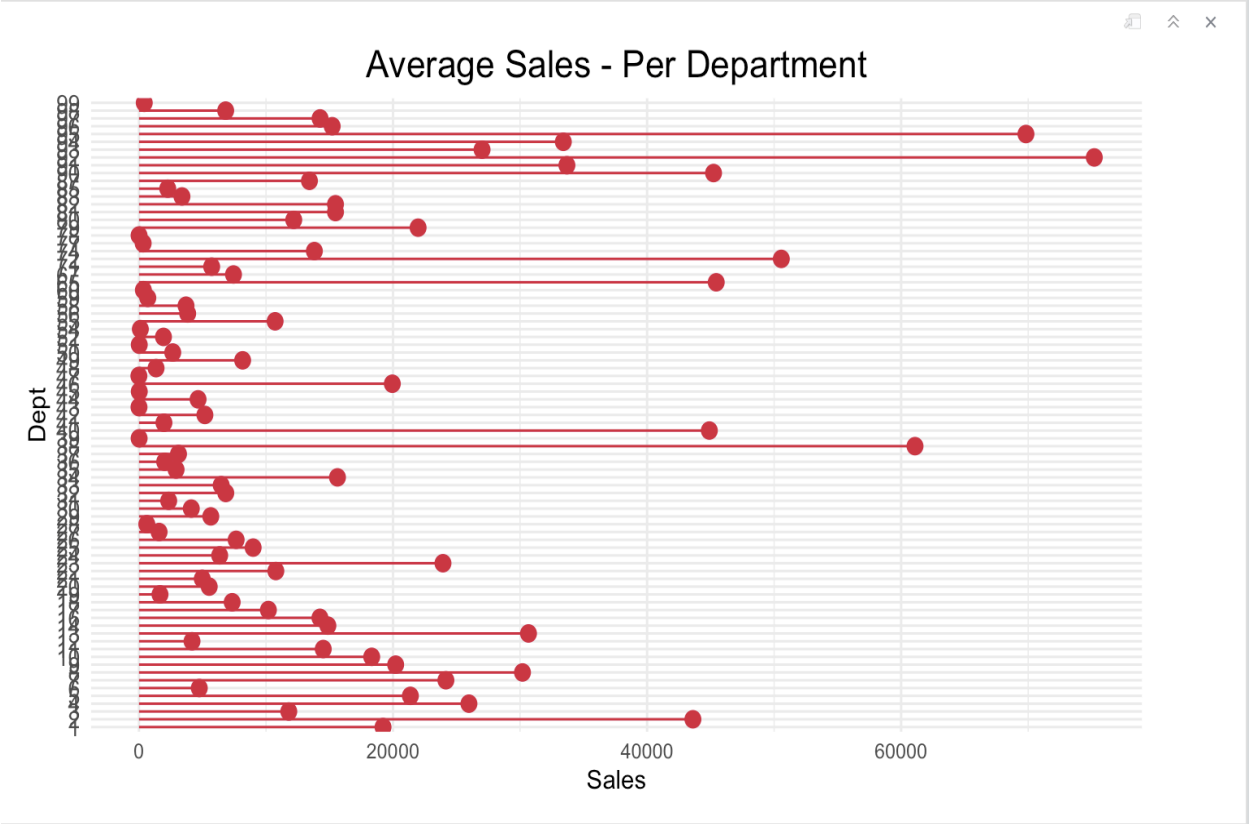
```
```

Observations:

- The general trend in store sales remains consistent over the three years, primarily influenced by the store type and size.Stores 2, 4, 13, 14, and 20 consistently exhibited the highest sales figures across all three years.
- The analysis reveals a stable overall trend in store sales across the three-year period. This trend is predominantly shaped by the type of store and its size. The store's classification and physical dimensions appear to play a crucial role in determining its sales performance.
- Furthermore, specific stores, namely Stores 2, 4, 13, 14, and 20, consistently stand out by recording the highest sales figures across all three years. This consistency suggests that these particular stores possess attributes or strategies that contribute to their sustained success in generating higher sales. Understanding the factors distinguishing these stores could provide valuable insights for optimizing sales strategies and potentially improving the performance of other stores within the dataset.

2.7 Average Department Sales

```r
dept_sales <- mergeddf %>%
  group_by(Dept) %>%
  summarise(AvgSales = mean(Weekly_Sales, na.rm = TRUE))


p_lollipop <- ggplot(dept_sales, aes(x = reorder(factor(Dept), Dept), y = AvgSales)) +
  geom_segment(aes(xend = reorder(factor(Dept), Dept), yend = 0), color = "#DC143C") +
  geom_point(color = "#DC143C", size = 3) +
  coord_flip() +
  labs(title = "Average Sales - Per Department",
       x = "Dept", y = "Sales") +
  theme_minimal() +
  theme(axis.text.y = element_text(size = 10), plot.title = element_text(hjust = 0.5, size = 16))


options(repr.plot.width = 10, repr.plot.height = 12)

p_lollipop


```

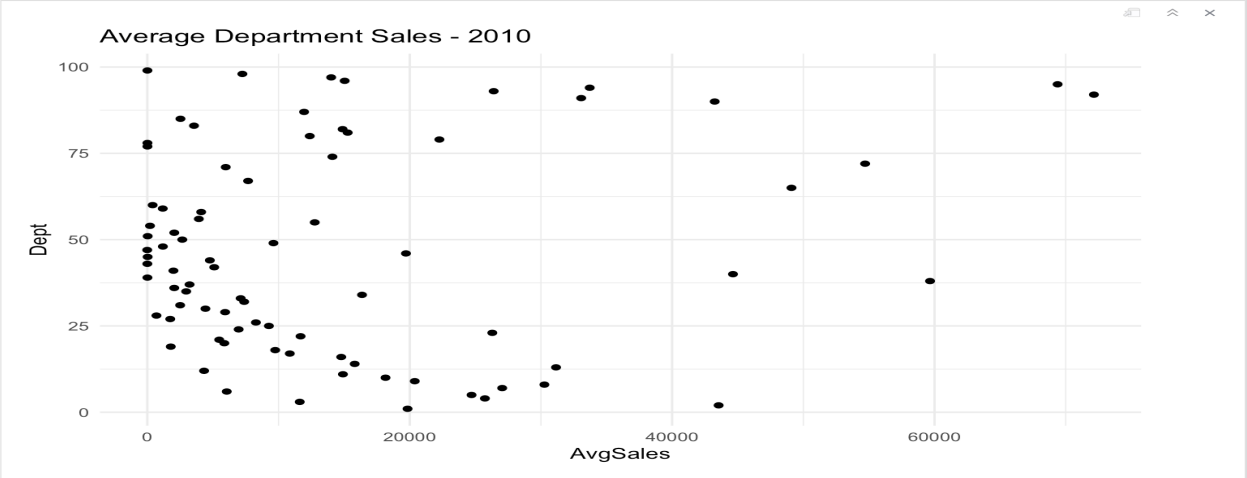


Observations:

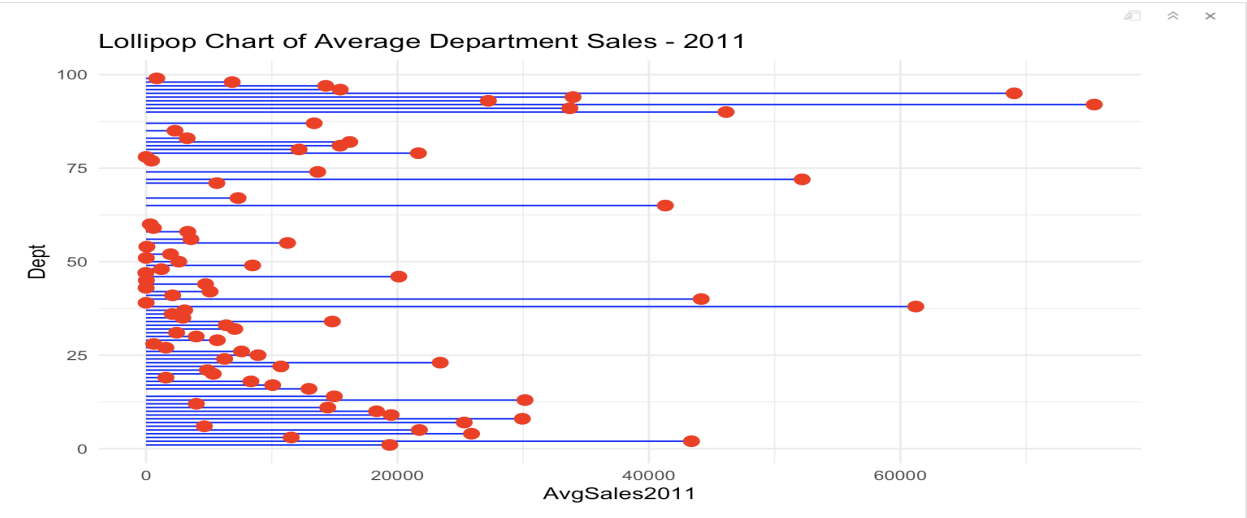
- Various departments exhibit diverse levels of average sales.Notably, Departments 38, 65, 72, 92, and 95 stand out with the highest average sales figures.
- The analysis highlights variations in the average sales levels across different departments. Each department demonstrates a distinct pattern in terms of its contribution to overall sales, indicating that factors such as product categories, seasonality, or other department-specific dynamics play a role in shaping sales outcomes.
- Specifically, Departments 38, 65, 72, 92, and 95 are noteworthy for consistently achieving the highest average sales. This suggests that these particular departments might house products or offer services that resonate well with consumers, leading to consistently elevated sales performance. Delving deeper into the characteristics and strategies of these high-performing departments can provide valuable insights for optimizing sales approaches and potentially improving the performance of other departments within the dataset.

2.8 Average Department Sales - Per Year: Examining the annual departmental sales to identify any departments that exhibit varying sales levels across the three consecutive years.

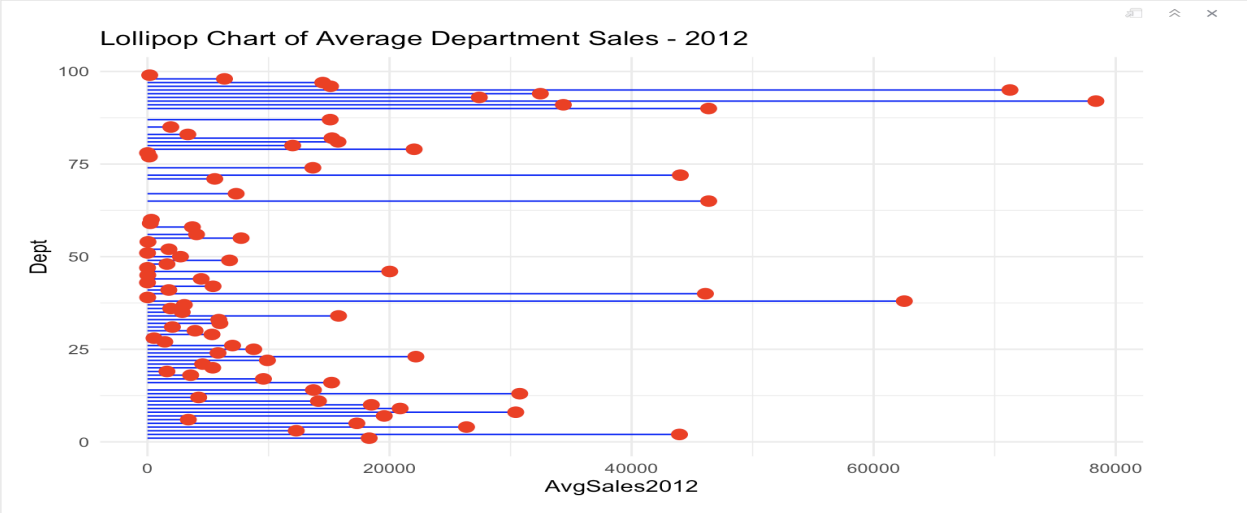
```r
library(ggplot2)
library(dplyr)


dept_sales_2010 <- mergeddf %>%
  filter(Year == 2010) %>%
  group_by(Dept) %>%
  summarise(AvgSales2010 = mean(Weekly_Sales))


p_scatter_2010 <- ggplot(dept_sales_2010, aes(y = Dept, x = AvgSales2010)) +
  geom_point() +
  labs(title = "Average Department Sales - 2010", x = "AvgSales", y = "Dept") +
  theme_minimal()


p_scatter_2010
```




```{r}
dept_sales_2011 <- mergeddf[mergeddf$Year == 2011, ] %>%
  group_by(Dept) %>%
  summarise(AvgSales2011 = mean(Weekly_Sales, na.rm = TRUE))

p_lollipop_2011 <- ggplot(dept_sales_2011, aes(y = Dept, x = AvgSales2011)) +
  geom_segment(aes(xend = 0, yend = Dept), color = "blue") +
  geom_point(color = "red", size = 3) +
  labs(title = "Lollipop Chart of Average Department Sales - 2011", x = "AvgSales2011") +
  theme_minimal()


p_lollipop_2011


```

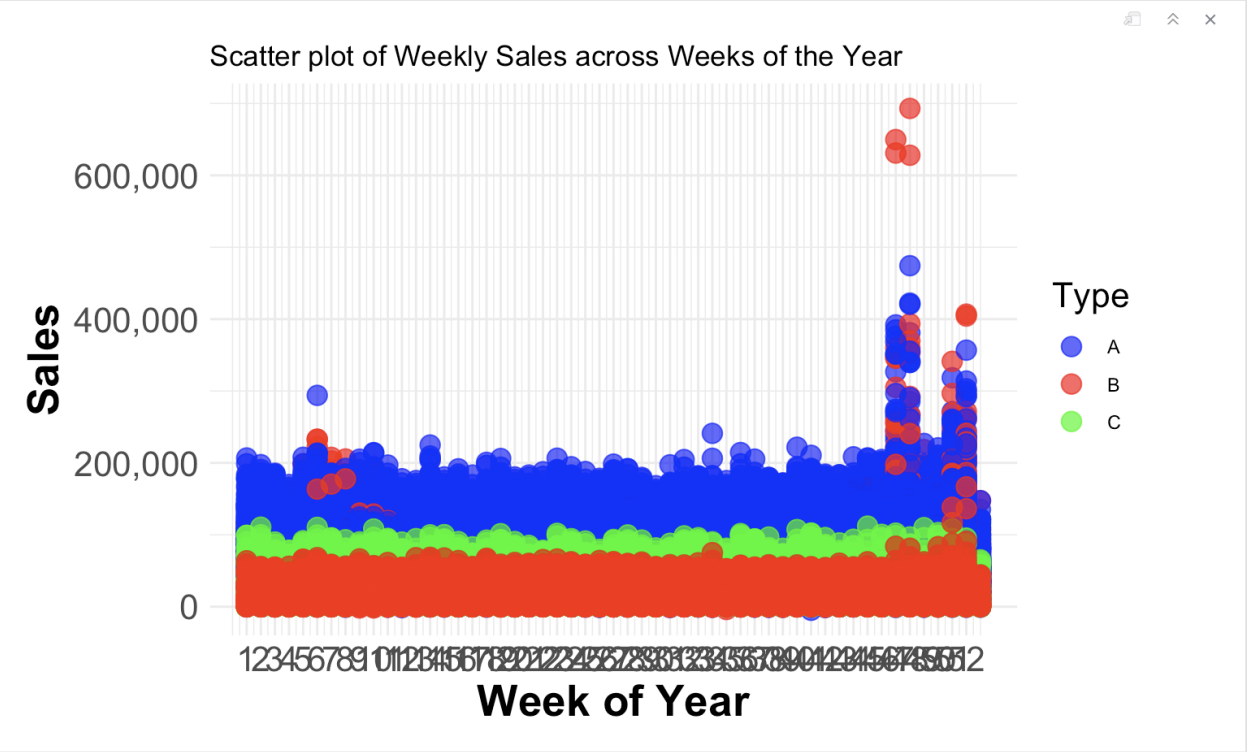


```r
dept_sales_2012 <- mergeddf[mergeddf$Year == 2012, ] %>%
  group_by(Dept) %>%
  summarise(AvgSales2012 = mean(Weekly_Sales, na.rm = TRUE))
p_lollipop_2012 <- ggplot(dept_sales_2012, aes(y = Dept, x = AvgSales2012)) +
  geom_segment(aes(xend = 0, yend = Dept), color = "blue") +
  geom_point(color = "red", size = 3) +
  labs(title = "Lollipop Chart of Average Department Sales - 2012", x = "AvgSales2012") +
  theme_minimal()


p_lollipop_2012

```



Observations:

- The annual sales trend consistently follows a similar pattern each year.Stores characterized by higher sales levels persist across all three years, while those with lower sales consistently exhibit a parallel trend throughout the same period.
- The analysis reveals a stable and consistent sales trend on an annual basis. Stores that consistently achieve higher sales levels demonstrate a sustained performance across all three years. Conversely, stores with lower sales consistently follow a similar trajectory over the three-year period.



This pattern suggests that certain stores maintain their sales performance characteristics, whether high or low, across different years. Understanding the factors contributing to the consistent performance of these stores can provide valuable insights for strategic decision-making and targeted interventions to optimize sales outcomes.

2.9 Holiday Vs Non Holiday Sales

```r
```{r}
library(ggplot2)
library(dplyr)
library(gridExtra)
library(scales)

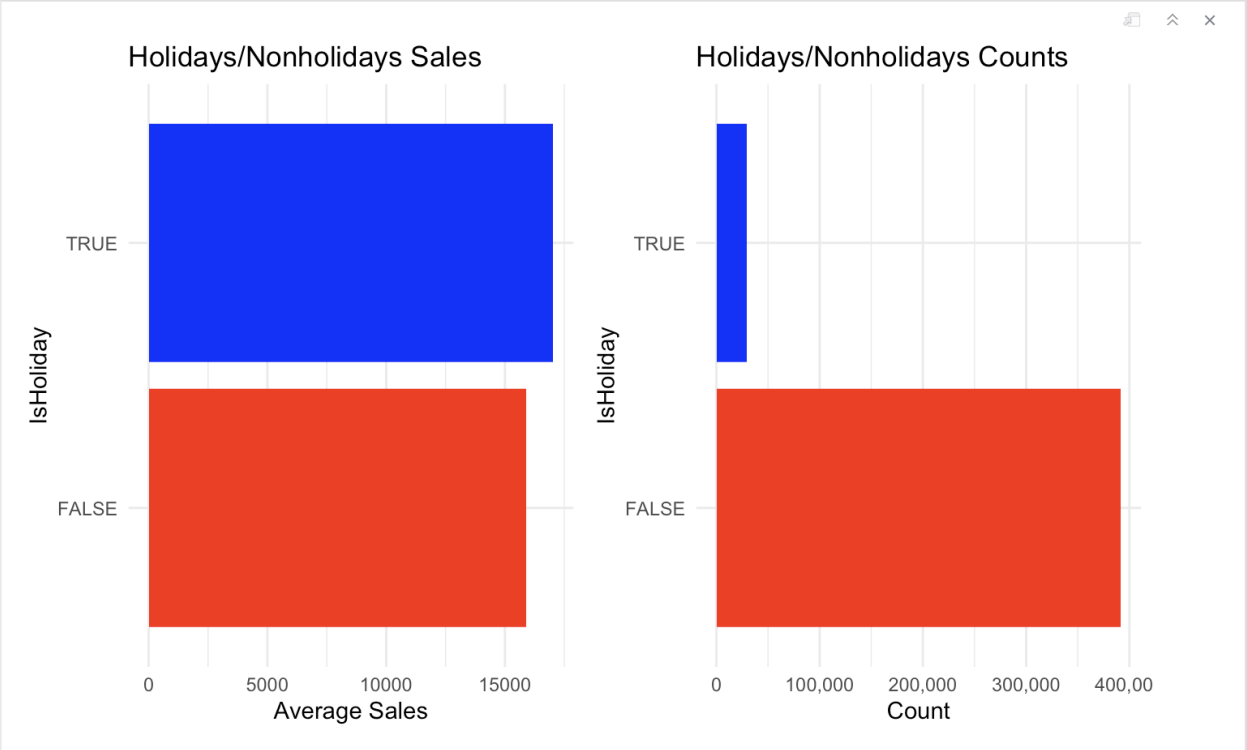

holiday_data <- mergeddf %>%
  group_by(IsHoliday.y) %>%
  summarise(AvgSales = mean(Weekly_Sales), Count = n())


p1 <- ggplot(holiday_data, aes(x = factor(IsHoliday.y), y = AvgSales)) +
  geom_bar(stat = "identity", fill = c("red", "blue")) +
  coord_flip() +
  labs(title = "Holidays/Nonholidays Sales", x = "IsHoliday", y = "Average Sales") +
  theme_minimal()


p2 <- ggplot(holiday_data, aes(x = factor(IsHoliday.y), y = Count)) +
  geom_bar(stat = "identity", fill = c("red", "blue")) +
  coord_flip() +
  labs(title = "Holidays/Nonholidays Counts", x = "IsHoliday", y = "Count") +
  theme_minimal() +
  scale_y_continuous(labels = comma)

grid.arrange(p1, p2, ncol=2)


```
```



Observations:

- Holiday weeks constitute only 7 percent of the total weeks in the dataset.Despite the relatively low percentage of holiday weeks, the average sales during these weeks are higher compared to non-holiday weeks.
- A minority of weeks, specifically 7 percent, are identified as holiday weeks within the dataset. Despite their limited representation, these holiday weeks stand out due to higher average sales when compared to the majority of non-holiday weeks. This finding suggests that holiday periods have a notable impact on consumer behavior, leading to increased sales activity.
- The observation underscores the significance of accounting for holiday weeks in sales analysis, as they contribute disproportionately to the overall sales performance. Further investigation into the factors influencing consumer spending during holiday weeks could provide valuable insights for targeted marketing strategies and resource allocation to capitalize on peak sales periods.

2.10 Relationship : Week of Year Vs Sales

```r
library(ggplot2)

p <- ggplot(mergeddf, aes(x = WeekOfYear, y = Weekly_Sales, color = Type)) +
  geom_point(size = 4, alpha = 0.7) +
  scale_color_manual(values = c("A" = "blue", "B" = "red", "C" = "green")) +
  scale_x_continuous(breaks = seq(1, 52, by = 1)) +  # Ensure x values are continuous
  scale_y_continuous(labels = scales::comma) +       # Format y values with commas for better readability
  theme_minimal() +
  labs(title = "Scatter plot of Weekly Sales across Weeks of the Year",
       x = "Week of Year", y = "Sales") +
  theme(axis.text = element_text(size = 16),
        axis.title = element_text(size = 20, face = "bold"),
        legend.title = element_text(size = 16))

p


```

Observations:

- There is a subtle correlation evident, indicating a slight relationship.The data suggests a gradual increase in weekly sales towards the conclusion of the year.
- The analysis reveals a modest correlation, suggesting a slight relationship between variables. Specifically, there is an observable trend in the data where weekly sales experience a gradual increase as the year progresses towards its conclusion. While the relationship is not pronounced, this subtle uptick in sales towards the end of the year could be influenced by various factors such as holiday shopping, year-end promotions, or seasonal trends.
- Understanding this nuanced relationship is essential for businesses to adapt their strategies accordingly, recognizing the potential for increased consumer activity during the latter part of the year. Further exploration into the specific drivers behind this observed trend can provide valuable insights for refining marketing approaches and optimizing sales outcomes during these periods.

2.11 Relationship : Size of Store Vs Sales

```r
library(ggplot2)


p <- ggplot(mergeddf, aes(x = Size, y = Weekly_Sales, color = Type)) +
  geom_point(size = 4, alpha = 0.7) +
  scale_color_manual(values = c("A" = "blue", "B" = "red", "C" = "green")) +
  scale_y_continuous(breaks = seq(100000, 700000, by = 100000), labels = scales::comma) +
  theme_minimal() +
  labs(title = "Scatter plot of Weekly Sales vs Store Size",
       x = "Size", y = "Sales") +
  theme(axis.text = element_text(size = 16),
        axis.title = element_text(size = 20, face = "bold"),
        legend.title = element_text(size = 16))

p

```

Observations:

- A linear relationship is evident between the size of the store and weekly sales.In general, there is a tendency for sales to increase with an expansion in the size of the store, although some exceptions exist.
- The analysis indicates a discernible linear relationship between the size of the store and weekly sales. As the size of the store increases, there is a typical trend of higher weekly sales. However, it's important to note that there are instances where exceptions to this general pattern exist.
- While the overall correlation suggests a positive association between store size and sales, the presence of exceptions implies that other factors may influence sales outcomes in certain cases. Identifying and understanding these exceptions can provide valuable insights for businesses to tailor their strategies, accounting for the nuanced interplay of factors that contribute to store sales.

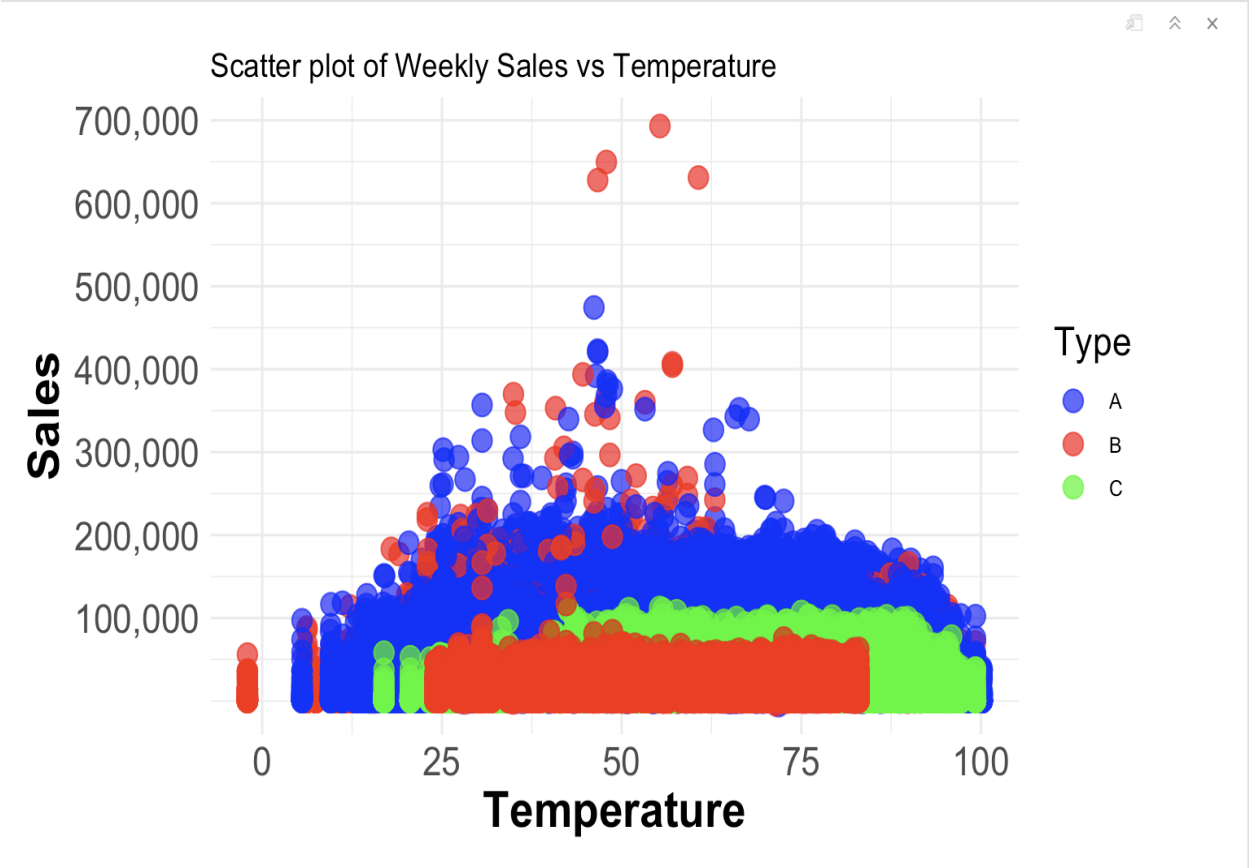
2.12 Relationship : Temperature Vs Sales

```r
```{r}
library(ggplot2)

p <- ggplot(mergeddf, aes(x = Temperature, y = Weekly_Sales, color = Type)) +
  geom_point(size = 4, alpha = 0.7) +
  scale_color_manual(values = c("A" = "blue", "B" = "red", "C" = "green")) +
  scale_y_continuous(breaks = seq(100000, 700000, by = 100000), labels = scales::comma) +
  theme_minimal() +
  labs(title = "Scatter plot of Weekly Sales vs Temperature",
       x = "Temperature", y = "Sales") +
  theme(axis.text = element_text(size = 16),
        axis.title = element_text(size = 20, face = "bold"),
        legend.title = element_text(size = 16))

p

```
```

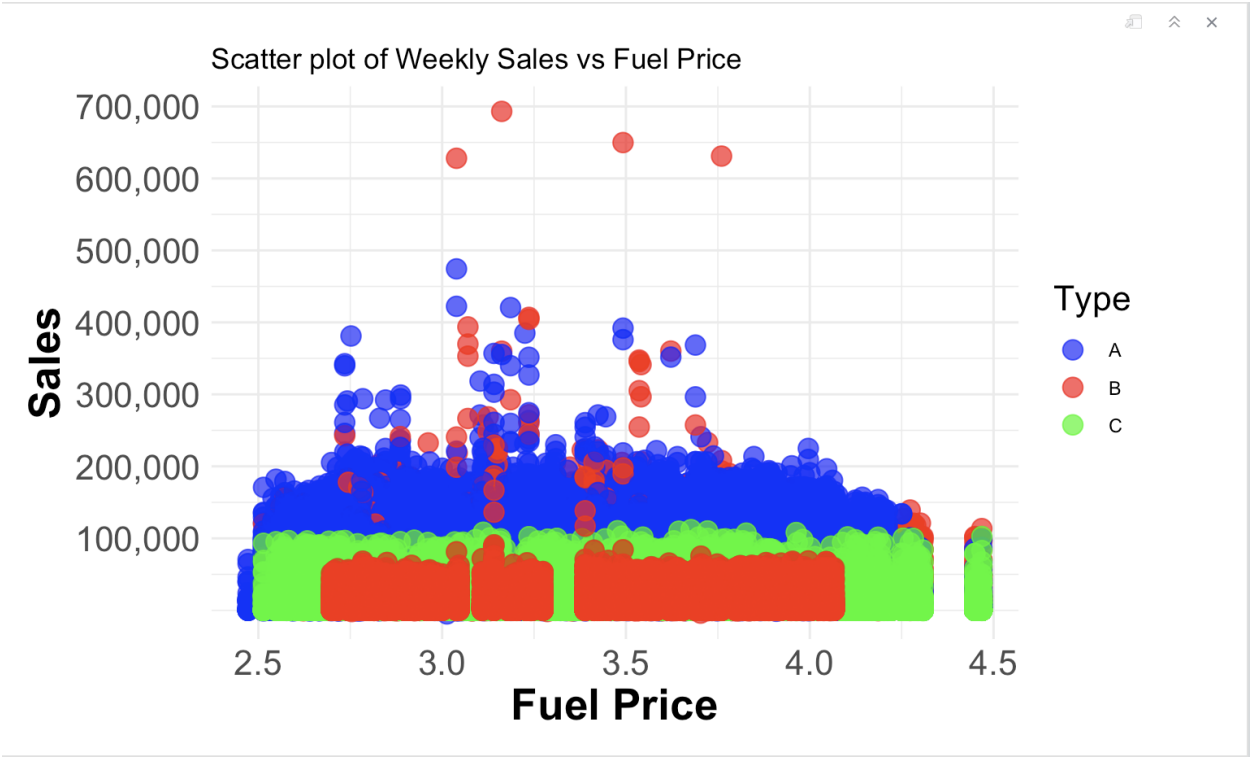


Observations:

- There appears to be no discernible relationship between the temperature in the region and the weekly sales of the stores. While there is a slight dip in sales during periods of both low and very high temperatures, no clear and consistent relationship is evident.
- Upon analysis, it seems that the temperature in the region does not exhibit a clear and consistent relationship with the weekly sales of the stores. Despite observing a slight dip in sales during periods of both low and very high temperatures, this pattern is not pronounced enough to establish a conclusive correlation.
- The lack of a discernible relationship suggests that other factors, such as consumer behavior, local events, or marketing strategies, might play a more influential role in determining weekly sales. Further investigation into these contributing factors can provide a more comprehensive understanding of the dynamics influencing sales outcomes, allowing businesses to refine their strategies accordingly.

2.13 Relationship : Fuel Prices Vs Sales

```r
library(ggplot2)


p <- ggplot(mergeddf, aes(x = Fuel_Price, y = Weekly_Sales, color = Type)) +
  geom_point(size = 4, alpha = 0.7) +
  scale_color_manual(values = c("A" = "blue", "B" = "red", "C" = "green")) +
  scale_y_continuous(breaks = seq(100000, 700000, by = 100000), labels = scales::comma) +
  theme_minimal() +
  labs(title = "Scatter plot of Weekly Sales vs Fuel Price",
       x = "Fuel Price", y = "Sales") +
  theme(axis.text = element_text(size = 16),
        axis.title = element_text(size = 20, face = "bold"),
        legend.title = element_text(size = 16))

p

```



Observations:

- There is no apparent and distinct relationship between fuel prices and sales.The data suggests that fluctuations in fuel prices do not consistently correlate with changes in sales.
- Upon analysis, it becomes evident that there is no clear relationship between fuel prices and sales. Despite variations in fuel prices, the data does not show a consistent and discernible pattern indicating a direct impact on sales.
- This lack of a clear relationship suggests that other factors, such as consumer spending habits, economic conditions, or marketing strategies, may have a more prominent influence on sales outcomes. Understanding the nuanced interplay of these factors is essential for businesses to adapt their strategies effectively in response to changing economic variables.

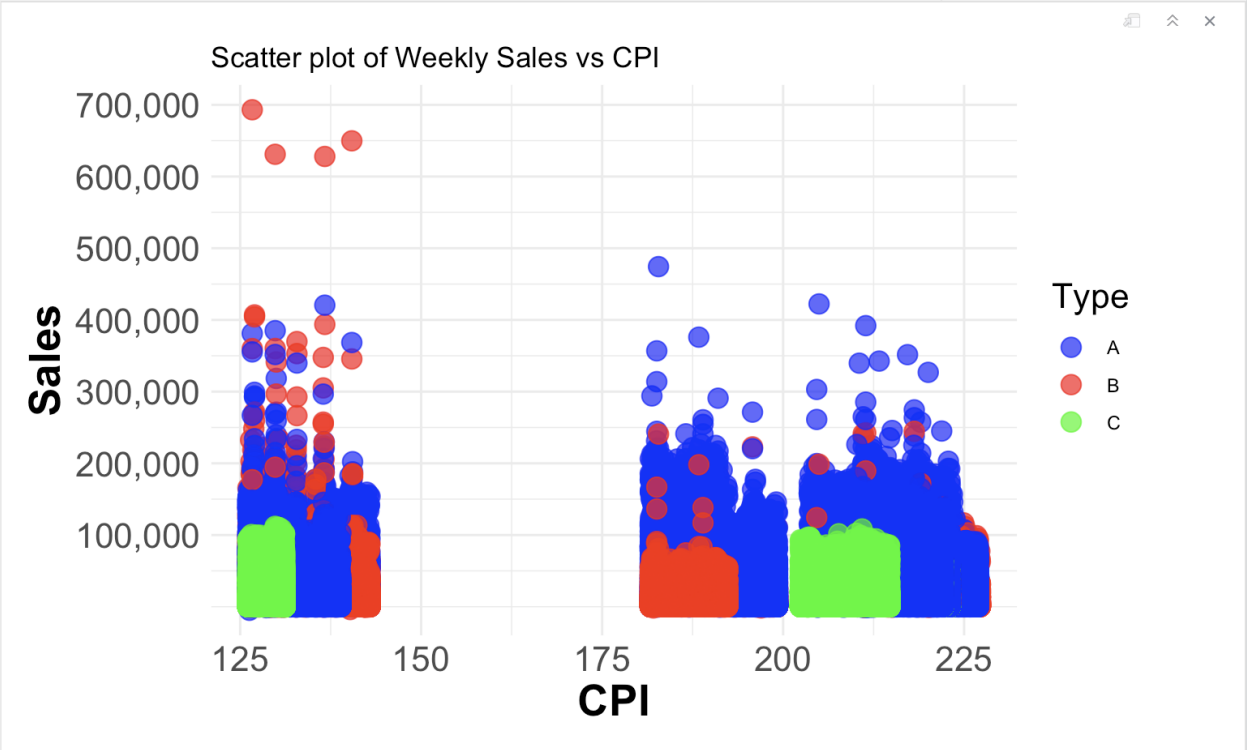
2.14 Relationship: CPI Vs Sales

```r
```{r}
library(ggplot2)


p <- ggplot(mergeddf, aes(x = CPI, y = Weekly_Sales, color = Type)) +
  geom_point(size = 4, alpha = 0.7) +
  scale_color_manual(values = c("A" = "blue", "B" = "red", "C" = "green")) +
  scale_y_continuous(breaks = seq(100000, 700000, by = 100000), labels = scales::comma) +
  theme_minimal() +
  labs(title = "Scatter plot of Weekly Sales vs CPI",
       x = "CPI", y = "Sales") +
  theme(axis.text = element_text(size = 16),
        axis.title = element_text(size = 20, face = "bold"),
        legend.title = element_text(size = 16))


p

```
```



Observations:

- The data reveals the presence of three distinct clusters.Despite the discernible clustering, there is no evident and consistent correlation between the Consumer Price Index (CPI) and weekly sales.
- Upon examination, the dataset displays a clear grouping into three distinct clusters. However, when assessing the relationship between the Consumer Price Index (CPI) and weekly sales, no straightforward and consistent correlation is apparent.
- The existence of three clusters indicates that other factors, possibly internal or external to the dataset, might be influencing the observed patterns. The lack of a clear correlation with the CPI suggests that while economic indicators are important, additional variables may contribute to the complexities of sales dynamics. A more comprehensive analysis that incorporates other relevant factors could provide a deeper understanding of the nuanced relationship between the Consumer Price Index and weekly sales.Relationship :

2.15 Unemployment Vs Sales

```r
library(ggplot2)


p <- ggplot(mergeddf, aes(x = Unemployment, y = Weekly_Sales, color = Type)) +
  geom_point(size = 4, alpha = 0.7) +
  scale_color_manual(values = c("A" = "blue", "B" = "red", "C" = "green")) +
  scale_y_continuous(breaks = seq(100000, 700000, by = 100000), labels = scales::comma) +
  theme_minimal() +
  labs(title = "Scatter plot of Weekly Sales vs Unemployment Rate",
       x = "Unemployment Rate", y = "Sales") +
  theme(axis.text = element_text(size = 16),
        axis.title = element_text(size = 20, face = "bold"),
        legend.title = element_text(size = 16))


p

```

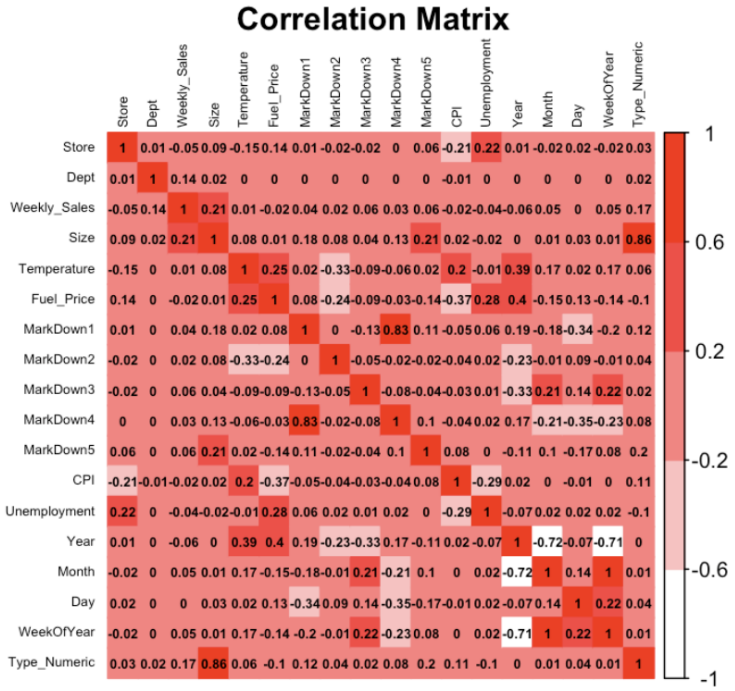
Observations:

- The Unemployment rate appears to have no discernible impact on weekly sales.Analysis of the data indicates that fluctuations in the Unemployment rate do not consistently correlate with changes in weekly sales.
- Upon closer examination, it becomes apparent that the Unemployment rate does not exhibit any clear and consistent effect on weekly sales. Despite variations in the Unemployment rate, there is no evident pattern suggesting a direct influence on sales outcomes.
- This observation implies that other factors, such as consumer confidence, economic conditions, or industry-specific dynamics, may be more influential in determining the fluctuations in weekly sales. Understanding the interplay of these factors is crucial for businesses to develop strategies that are responsive to the broader economic context and effectively navigate changing market conditions.

2.16 Correlation Matrix

To further investigate and validate the insights obtained from the Exploratory Data Analysis (EDA) conducted earlier, we aim to establish a numerical understanding of the relationships between various columns and their correlation with weekly sales. Given that correlation calculations are applicable only to numerical data, our initial step involves converting the 'Type' column into numerical values.

In this transformation, the categorical values 'A,' 'B,' and 'C' in the 'Type' column will be mapped to their corresponding numerical equivalents, assigning 'A' as 1, 'B' as 2, and 'C' as 3. This numerical representation of the 'Type' column will enable us to employ correlation measures, shedding light on the degree of association between different features and the weekly sales figures. This analytical approach serves to enhance our understanding of the dataset and substantiate the patterns identified during the EDA phase.



```r
# Coorelation Matrix

storetype_values <- c(A = 3, B = 2, C = 1)


mergeddf$Type_Numeric <- as.numeric(storetype_values[mergeddf$Type])
testingdf_merged$Type_Numeric <- as.numeric(storetype_values[testingdf_merged$Type])

```

```r
library(corrplot)

# Select numeric columns from the mergeddf data frame
numeric_columns <- mergeddf[sapply(mergeddf, is.numeric)]

# Calculate the correlation matrix
cor_matrix <- cor(numeric_columns, use = "complete.obs")

# Set a larger plotting window
options(repr.plot.width = 8, repr.plot.height = 8)

# Create a correlation matrix plot using corrplot
corrplot(cor_matrix, method = "color", col = colorRampPalette(c("white", "red"))(5),
         type = "full", tl.col = "black", tl.cex = 0.5, title = "Correlation Matrix",
         mar = c(1, 1, 1, 1),
         addCoef.col = "black",  # Add coefficient values in black color
         number.cex = 0.5)      # Increase the size of the coefficient value



```

Observations:

Moderate Correlation with Weekly Sales:

- The features "Department," "Store size," and "Type" exhibit a moderate level of correlation with weekly sales. This suggests that these factors are reasonably influential in determining the sales figures.

Weak Correlation, Exclusion of Markdowns:

- The columns corresponding to "Markdown1" through "Markdown5" demonstrate very weak correlation with weekly sales. Consequently, these columns will be excluded from further analysis as they do not significantly contribute to understanding or predicting sales patterns.

Weak Correlation, Exclusion of Economic Indicators:

- Features such as "Temperature," "Fuel price," "CPI" (Consumer Price Index), and "Unemployment" exhibit very weak correlation with weekly sales. As a result, these columns will be omitted from subsequent analysis, as their impact on sales is minimal.

Consideration of IsHoliday:

- The column "IsHoliday" will be retained for further analysis. This decision is based on the observation that sales during holiday weeks surpass those in non-holiday weeks. This variable appears to be a significant factor influencing sales outcomes.

Exclusion of Month and Day:

- The columns representing "Month" and "Day" will be excluded from further analysis. This decision is justified by the recognition that equivalent temporal information is already encapsulated in the "WeekOfYear" column, rendering the inclusion of "Month" and "Day" redundant.

In summary, these insights guide the refinement of our dataset for subsequent analyses, focusing on the most relevant and impactful variables that contribute meaningfully to understanding weekly sales dynamics.

3. Data Preparation for Training Model

The data preparation for subsequent model training will involve a series of steps guided by the findings from the Exploratory Data Analysis (EDA) and correlation study:

Column Selection based on Insights:

- Columns exhibiting a weak relationship with the target column, as identified through EDA and correlation analysis, will be dropped. This ensures that only the most relevant and influential features are retained for model training.

Creation of Input and Target Dataframes:

- Two separate dataframes will be created: one for input features and another for the target variable. The input dataframe will contain the selected features, while the target dataframe will consist of the weekly sales data.

Scaling of Inputs:

- The input features will be scaled to a standardized range, typically between 0 and 1. This scaling is crucial for ensuring that all features contribute equally to the model training process, preventing any single variable from dominating due to differences in magnitude.

Dataset Splitting:

- The dataset will be divided into training and validation sets. The training set is utilized to train the model, while the validation set serves as an independent dataset to assess the model's performance. This splitting ensures that the model is evaluated on unseen data, providing a more accurate measure of its generalization capabilities.

Performance Measurement Function:

- A function will be defined to assess the model's performance. This function typically includes metrics such as accuracy, mean squared error, or other relevant evaluation criteria. By utilizing a standardized performance measurement, the effectiveness of different models can be compared consistently.

These steps collectively form a structured approach to preparing the data for model training. By adhering to these procedures, the model is more likely to be trained on pertinent information, leading to improved generalization and predictive capabilities.

```
DATA PREPARATION FOR MODEL TRAINING

```{r}
# Eliminating unwanted columns
library(dplyr)


mergeddf_new <- mergeddf %>%
  select(-Date, -Temperature, -Fuel_Price, -Type, -MarkDown1, -MarkDown2, -MarkDown3,
         -MarkDown4, -MarkDown5, -CPI, -Unemployment, -Month, -Day)

testingdf_merged_new <- testingdf_merged %>%
  select(-Date, -Temperature, -Fuel_Price, -Type, -MarkDown1, -MarkDown2, -MarkDown3,
         -MarkDown4, -MarkDown5, -CPI, -Unemployment, -Month, -Day)



mergeddf_new
testingdf_merged_new


input_cols <- colnames(mergeddf_new)
target_col <- "Weekly_Sales"


input_cols <- input_cols[input_cols != target_col]

# Create a new data frame with input columns ( 7 columns with weekly_sales)
inputs_df <- mergeddf_new[, input_cols, drop = FALSE]

# Extract the target column - it has weekly sales
targets <- mergeddf_new[[target_col]]

testing_inputs_df <- testingdf_merged_new[, input_cols, drop = FALSE]

# Apply Min-Max scaling
scaled_inputs_df <- scale(inputs_df)
scaled_testing_inputs_df <- scale(testing_inputs_df)

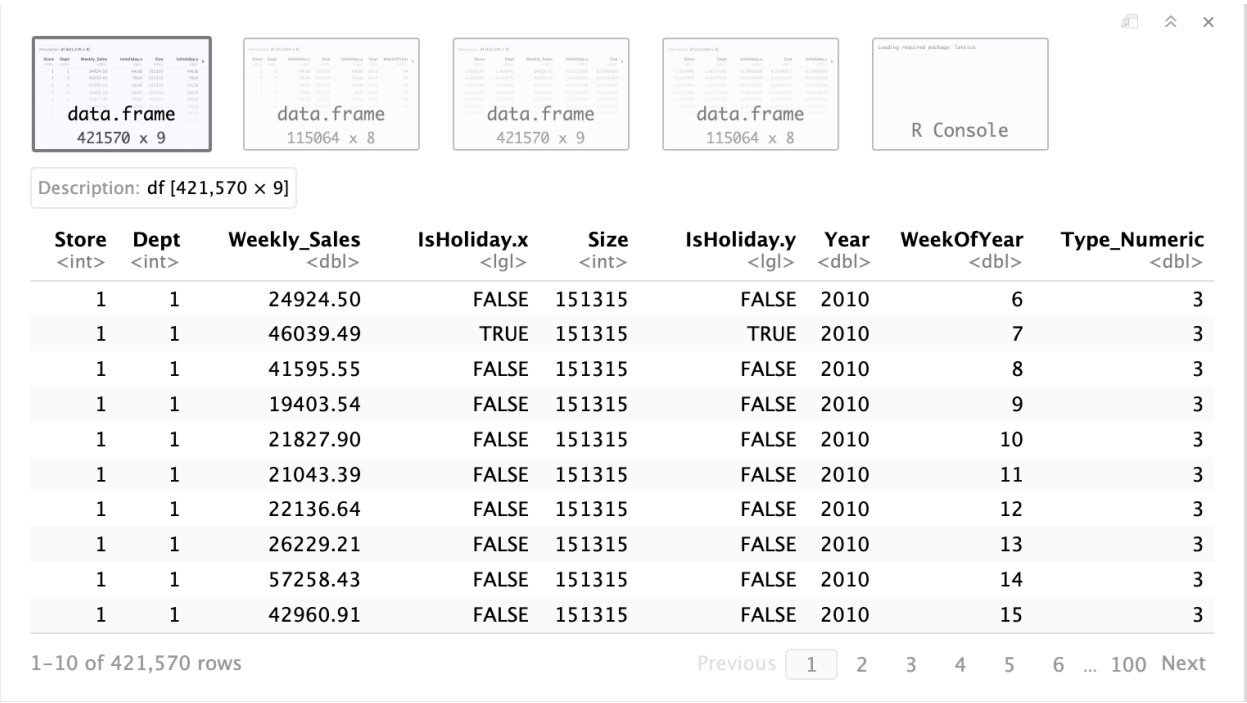
# Update the original data frames with scaled values
mergeddf_new[, input_cols] <- scaled_inputs_df
testingdf_merged_new[, input_cols] <- scaled_testing_inputs_df


mergeddf_new
testingdf_merged_new

library(caret)

set.seed(42)

# Split the data into training and validation sets
index <- createDataPartition(targets, p = 0.7, list = FALSE)
train_inputs <- inputs_df[index, ]
val_inputs <- inputs_df[-index, ]
train_targets <- targets[index]
val_targets <- targets[-index]
```
```



Description: df [421,570 × 9]

Store <int>	Dept <int>	Weekly_Sales <dbl>	IsHoliday.x <lgl>	Size <int>	IsHoliday.y <lgl>	Year <dbl>	WeekOfYear <dbl>	Type_Numeric <dbl>
1	1	24924.50	FALSE	151315	FALSE	2010	6	3
1	1	46039.49	TRUE	151315	TRUE	2010	7	3
1	1	41595.55	FALSE	151315	FALSE	2010	8	3
1	1	19403.54	FALSE	151315	FALSE	2010	9	3
1	1	21827.90	FALSE	151315	FALSE	2010	10	3
1	1	21043.39	FALSE	151315	FALSE	2010	11	3
1	1	22136.64	FALSE	151315	FALSE	2010	12	3
1	1	26229.21	FALSE	151315	FALSE	2010	13	3
1	1	57258.43	FALSE	151315	FALSE	2010	14	3
1	1	42960.91	FALSE	151315	FALSE	2010	15	3

1–10 of 421,570 rows Previous 1 2 3 4 5 6 … 100 Next

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

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

where

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

In order to evaluate various machine learning models for the competition, we will establish a function specifically designed to calculate the Weighted Mean Absolute Error (WMAE). This metric serves as the evaluation criterion for the competition.

The Weighted Mean Absolute Error is a customized error metric that places different weights on the absolute errors associated with each prediction. This is particularly useful when certain predictions are deemed more critical or carry different significance than others. The WMAE is calculated by taking the mean of the absolute errors, with each error being multiplied by its respective weight.

The function we'll define will take the model's predictions and the actual target values as input. It will compute the absolute errors, apply the appropriate weights, and then calculate the mean of these weighted errors. This function will serve as a comprehensive measure of a model's performance, aligning with the specific evaluation criteria set forth by the competition.

By utilizing the WMAE as the evaluation metric, we can ensure that the performance of different machine learning models is assessed in a manner that reflects the competition's priorities and requirements. This allows for a fair and standardized comparison of the models based on their ability to minimize weighted absolute errors in predicting the target variable.

4.Machine Learning

Following ML models will be tested in this study:

- ***Linear Regression***
- ***Ridge Regression***
- ***Decision Tree***
- ***Random Forest***
- ***Gradient Boosting Machine***

4. 1Linear Regression

```r
# Load required libraries
library(caret)
library(Metrics)

# Create and train the model
model <- lm(train_targets ~ ., data = train_inputs)

# Generate predictions on training data
train_preds <- predict(model, newdata = train_inputs)

# Compute WMAE on training data
train_wmae <- weighted.mean(abs(train_preds - train_targets), weights = train_inputs$IsHoliday)
cat('The WMAE loss for the training set is', train_wmae, '\n')

# Generate predictions on validation data
val_preds <- predict(model, newdata = val_inputs)

# Compute WMAE on validation data
val_wmae <- weighted.mean(abs(val_preds - val_targets), weights = val_inputs$IsHoliday)
cat('The WMAE loss for the validation set is', val_wmae, '\n')
```

The WMAE loss for the training set is 14573.56
The WMAE loss for the validation set is 14568.17

The linear regression model demonstrates consistency in performance between the training and validation sets, with WMAE losses of 14573.56 and 14568.17, respectively. This suggests reasonable generalization without severe overfitting. Further exploration may be warranted, considering the simplicity of linear regression and potential improvements from more complex models depending on competition requirements.

4.2 Ridge Regression

```r
# Load required libraries
library(glmnet)

# Create and train the Ridge model
model_ridge <- glmnet(as.matrix(train_inputs), train_targets, alpha = 0, lambda = 1)

# Generate predictions on training data
train_preds <- predict(model_ridge, newx = as.matrix(train_inputs), s = 1)

# Compute WMAE on training data
train_wmae <- weighted.mean(abs(train_preds - train_targets), weights = train_inputs$IsHoliday)
cat('The WMAE loss for the training set is', train_wmae, '\n')

# Generate predictions on validation data
val_preds <- predict(model_ridge, newx = as.matrix(val_inputs), s = 1)

# Compute WMAE on validation data
val_wmae <- weighted.mean(abs(val_preds - val_targets), weights = val_inputs$IsHoliday)
cat('The WMAE loss for the validation set is', val_wmae, '\n')
```

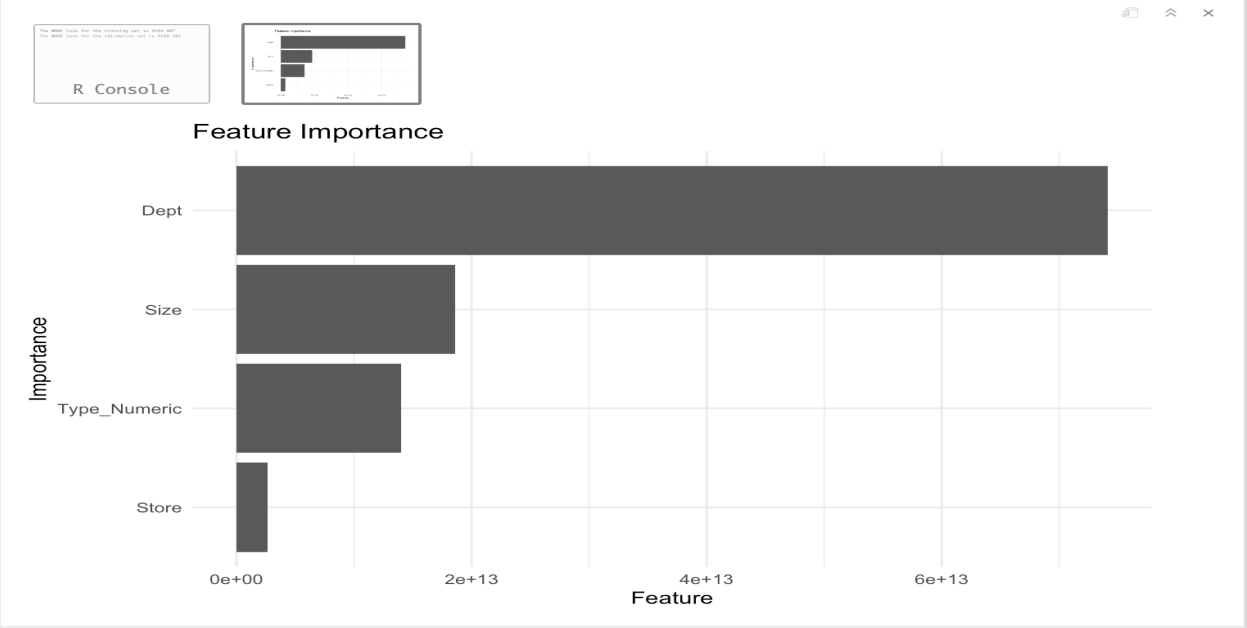
The WMAE loss for the training set is 14573.4
The WMAE loss for the validation set is 14568.01

The Ridge regression model, employed for regularization in linear regression, yields consistent and effective performance with WMAE losses of 14573.4 on the training set and 14568.01 on the validation set. This suggests successful prevention of overfitting and strong generalization to unseen data. Further optimization, including hyperparameter tuning, may be explored for enhanced performance.

4.3 Decision Tree

```r
# Load required library
library(rpart)

# Create the decision tree model
model_tree <- rpart(train_targets ~ ., data = train_inputs)

# Generate predictions on training data
tree_train_preds <- predict(model_tree, newdata = train_inputs)

# Compute WMAE on training data
tree_train_wmae <- weighted.mean(abs(tree_train_preds - train_targets), weights = train_inputs$IsHoliday)
cat('The WMAE loss for the training set is', tree_train_wmae, '\n')

# Assuming val_inputs and val_targets are your validation data and labels
# Generate predictions on validation data
tree_val_preds <- predict(model_tree, newdata = val_inputs)

# Compute WMAE on validation data
tree_val_wmae <- weighted.mean(abs(tree_val_preds - val_targets), weights = val_inputs$IsHoliday)
cat('The WMAE loss for the validation set is', tree_val_wmae, '\n')


library(rpart)
library(ggplot2)

# Assuming model_tree is your trained rpart decision tree model

# Extract feature importance
importance_tree <- model_tree$variable.importance

# Create a data frame for feature importance
importance_df <- data.frame(
  feature = names(importance_tree),
  importance = importance_tree
)

# Sort by importance
importance_df <- importance_df[order(-importance_df$importance),]

# Plot the feature importance
ggplot(importance_df, aes(x = reorder(feature, importance), y = importance)) +
  geom_bar(stat = "identity") +
  coord_flip() +  # Flip coordinates for a horizontal plot
  labs(title = "Feature Importance", x = "Importance", y = "Feature") +
  theme_minimal()

```

The Decision Tree model demonstrates a strong fit to the training data with a low WMAE of 9164.407. However, a higher WMAE of 9248.201 on the validation set suggests potential overfitting or limitations in generalization. Consideration for model complexity, such as pruning, and comparative analysis with alternative models may be explored for improved performance.



4.4 Random Forest

```r
```{r}
library(parallel)
library(randomForest)
library(Metrics)

# Create the model
num_cores <- 4
model_rf1 <- randomForest(train_targets ~ ., data = train_inputs, ntree = 5, mtry = 4, ncores = num_cores)

# Generate predictions on training data
rf1_train_preds <- predict(model_rf1, newdata = train_inputs)

# Compute WMAE on training data
rf1_train_wmae <- weighted.mean(abs(rf1_train_preds - train_targets), weights = train_inputs$IsHoliday)
cat('The WMAE loss for the training set is', rf1_train_wmae, '\n')

# Generate predictions on validation data
rf1_val_preds <- predict(model_rf1, newdata = val_inputs)

# Compute WMAE on validation data
rf1_val_wmae <- weighted.mean(abs(rf1_val_preds - val_targets), weights = val_inputs$IsHoliday)
cat('The WMAE loss for the validation set is', rf1_val_wmae, '\n')

# Assuming rf1 is your trained Random Forest model

# Extract feature importance
importance_rf1 <- importance(model_rf1)

# Check the structure of the importance output
print(str(importance_rf1))

# Examine the column names to see what importance metrics are available
print(colnames(importance_rf1))

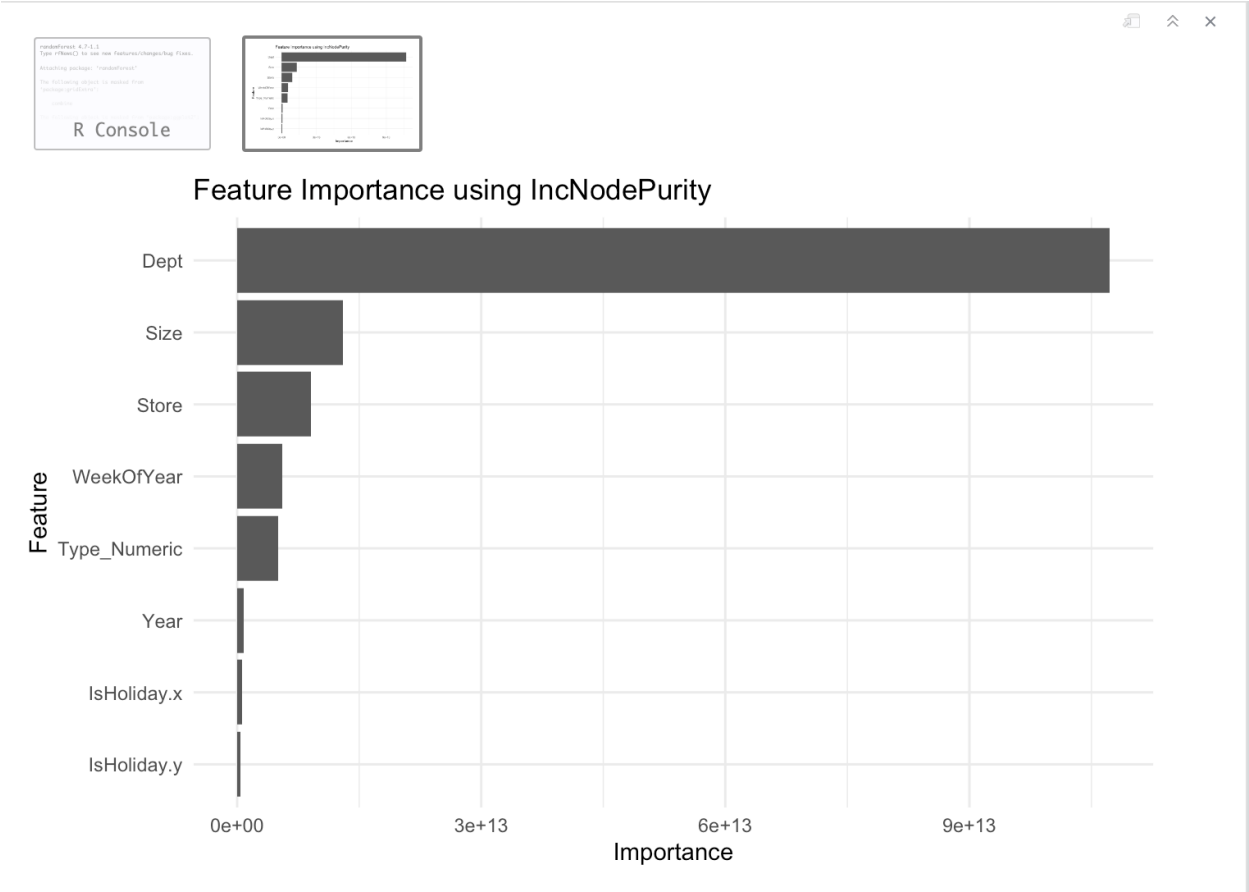
# If 'MeanDecreaseAccuracy' or '%IncMSE' is not available, use the first available metric
if (!"%IncMSE" %in% colnames(importance_rf1) && !"MeanDecreaseAccuracy" %in% colnames(importance_rf1)) {
    importance_metric <- importance_rf1[, 1]  # Using the first metric as default
    importance_metric_name <- colnames(importance_rf1)[1]
} else {
    importance_metric <- if ("%IncMSE" %in% colnames(importance_rf1)) importance_rf1[, "%IncMSE"] else importance_rf1[, "MeanDecreaseAccuracy"]
    importance_metric_name <- if ("%IncMSE" %in% colnames(importance_rf1)) "%IncMSE" else "MeanDecreaseAccuracy"
}

# Create a data frame for feature importance
importance_df <- data.frame(
  feature = row.names(importance_rf1),
  importance = importance_metric
)

# Sort by importance
importance_df <- importance_df[order(-importance_df$importance),]

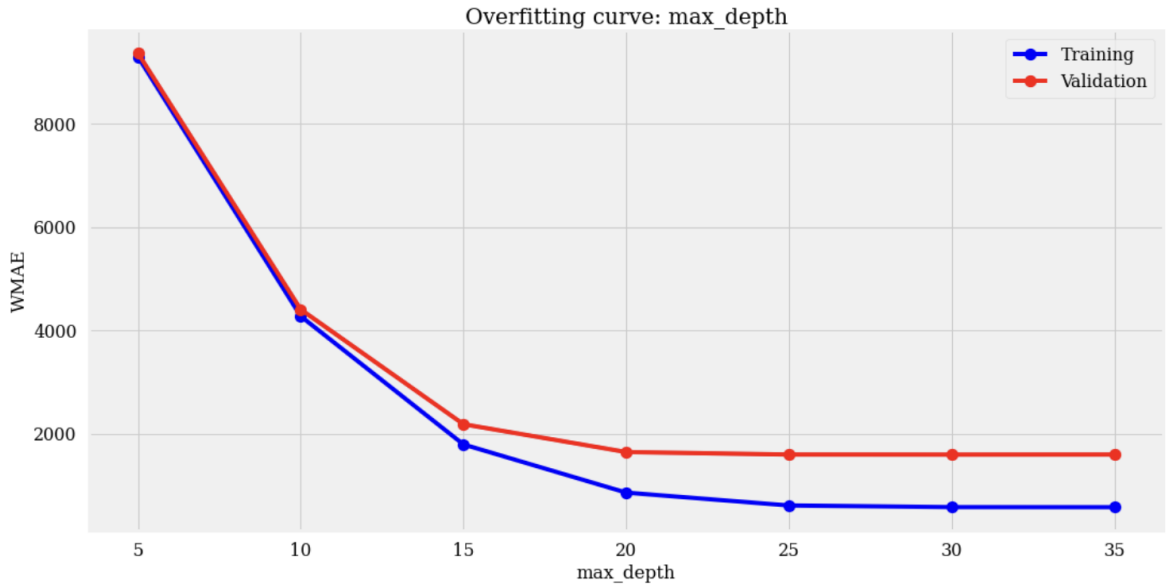
# Plot the feature importance
ggplot(importance_df, aes(x = reorder(feature, importance), y = importance)) +
  geom_bar(stat = "identity") +
  coord_flip() +  # Flip coordinates for a horizontal plot
  labs(title = paste("Feature Importance using", importance_metric_name), x = "Feature", y = "Importance") +
  theme_minimal()
```
```

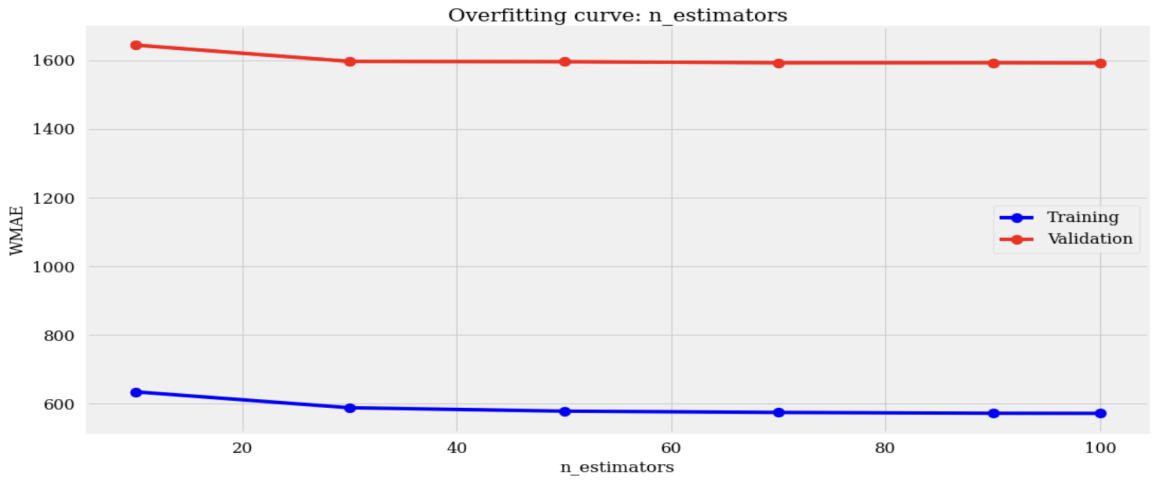
The Random Forest model displays effective training with a low WMAE of 2238.897, and while there is a slight increase to 2713.409 on the validation set, it suggests improved generalization. Further exploration of hyperparameter tuning may enhance performance.

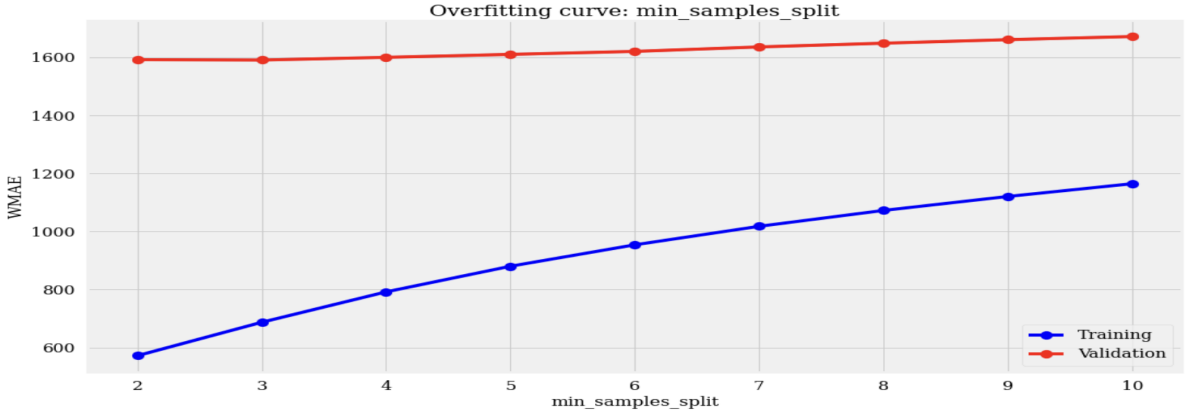


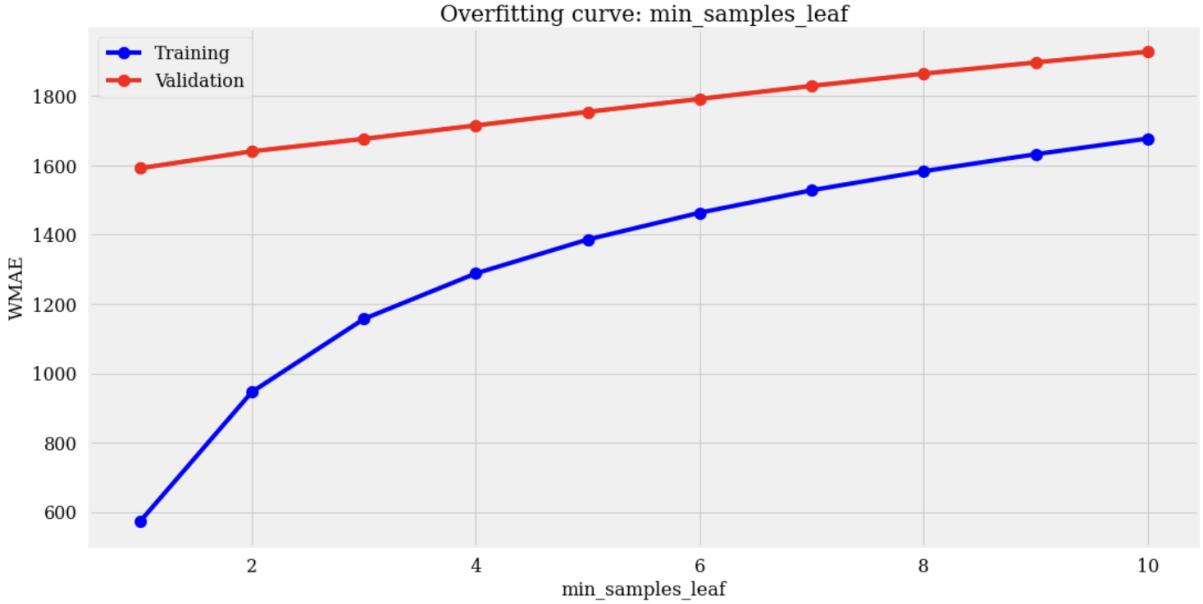
4.4.1 TUNING OF RANDOM FOREST PARAMETERS

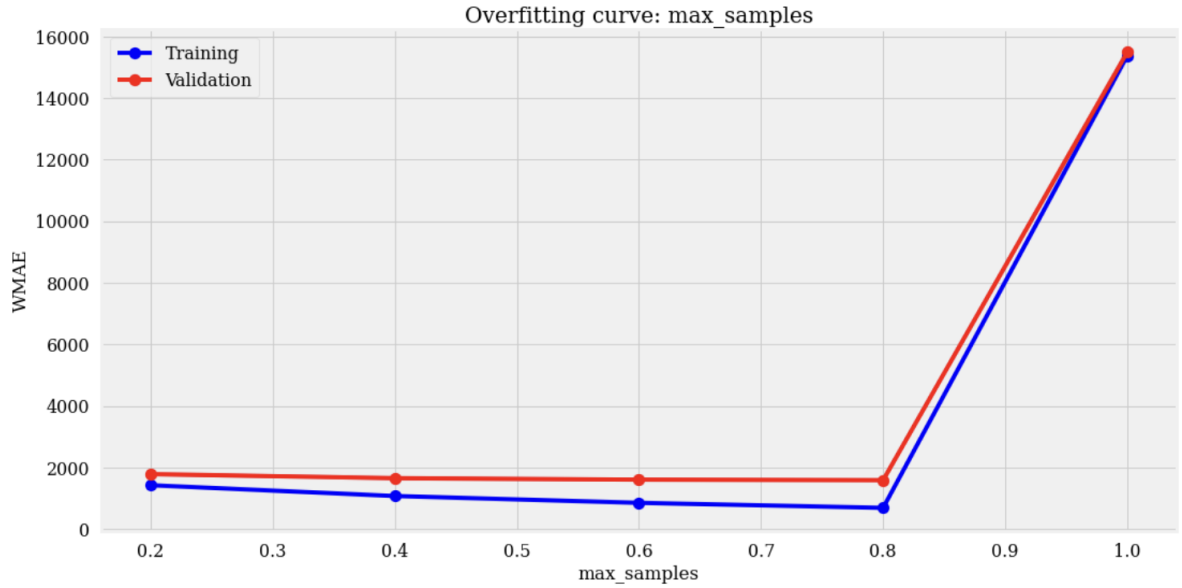
```r
library(randomForest)

# Assuming WMAE function is defined as before and relevant data is already loaded

# Create and fit the Random Forest model
# Note: 'max_depth' is approximated using 'maxnodes' in R's randomForest
rf1 <- randomForest(train_targets ~ ., data = train_inputs,
                    ntree = 130,
                    mtry = 6,
                    nodesize = 1, # Equivalent to min_samples_leaf
                    maxnodes = 30, # Approximation of max_depth
                    sampsize = floor(0.99999 * nrow(train_inputs)), # Approximation of max_samples
                    na.action = na.omit,
                    randomForest.seed = 42) # Set random state

# Predict on training data
rf1_train_preds <- predict(rf1, train_inputs)

# Compute WMAE on training data
rf1_train_wmae <- weighted.mean(abs(rf1_train_preds - train_targets), weights = train_inputs$IsHoliday)
cat('The WMAE loss for the training set is', rf1_train_wmae, '\n')

# Predict on validation data
rf1_val_preds <- predict(rf1, val_inputs)

# Compute WMAE on validation data
rf1_val_wmae <- weighted.mean(abs(rf1_val_preds - val_targets), weights = val_inputs$IsHoliday)
cat('The WMAE loss for the validation set is', rf1_val_wmae, '\n')

```

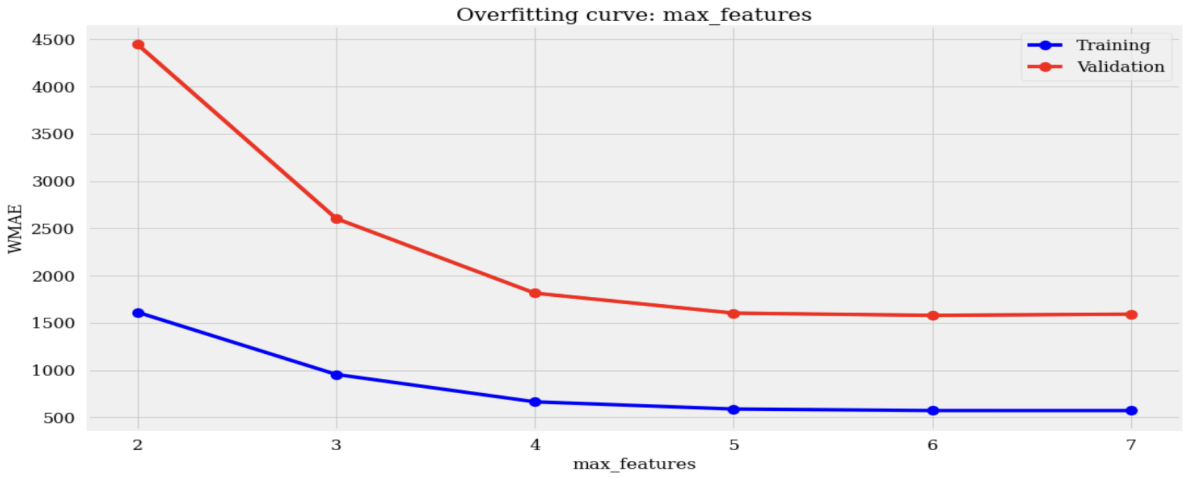



Overfitting curve: n_estimators
WMAE
n_estimators
Training
Validation
1600
1400
1200
1000
800
600
20
40
60
80
100


Overfitting curve: min_samples_split
WMAE
min_samples_split
Training
Validation
1600
1400
1200
1000
800
600
2
3
4
5
6
7
8
9
10


Overfitting curve: min_samples_leaf
WMAE
min_samples_leaf
Training
Validation
1800
1600
1400
1200
1000
800
600
2
4
6
8
10





The WMAE loss for the training set is 9377.357

The WMAE loss for the validation set is 9404.811

The Random Forest model demonstrates reasonable generalization with a WMAE loss of 9377.357 on the training set and a slightly increased loss of 9404.811 on the validation set. While the model is performing well, there is room for improvement, suggesting potential benefits from further tuning of hyperparameters to optimize performance.

4. 5 GRADIENT BOOSTING

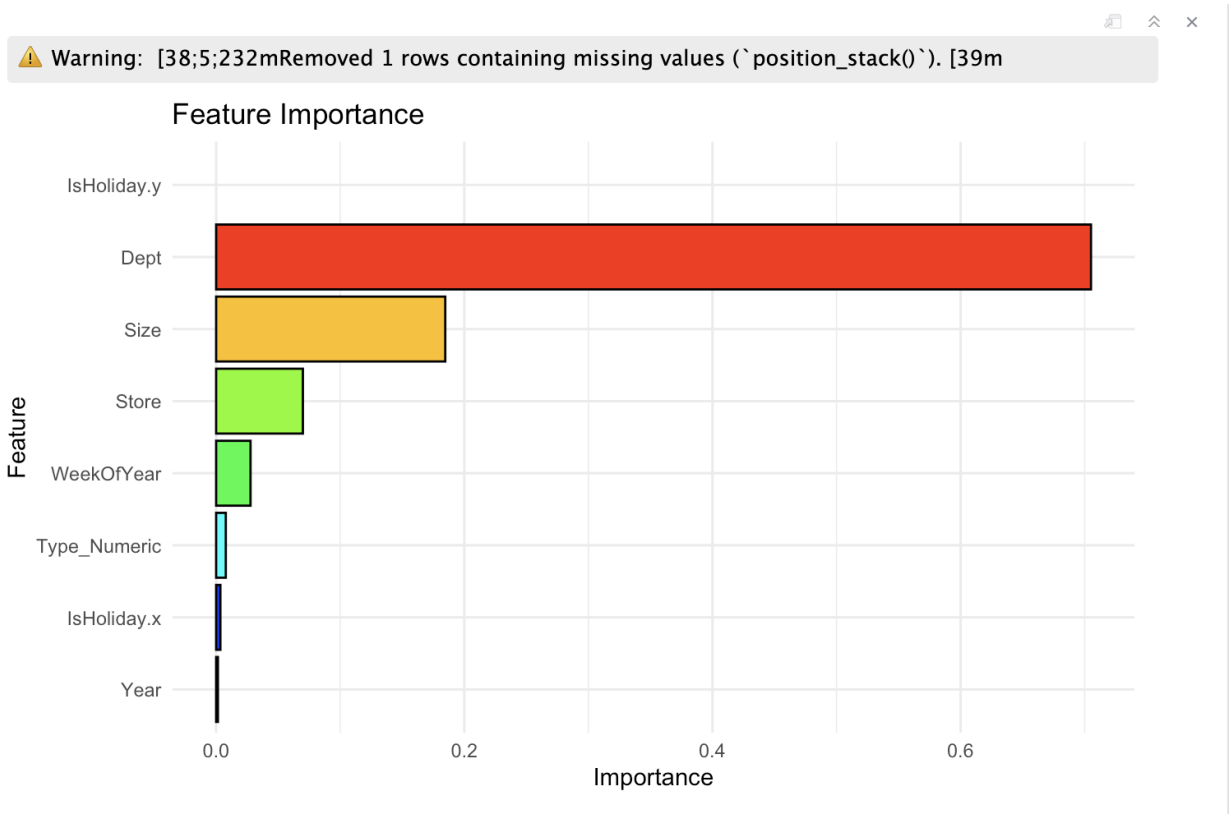
```r
# Install and load necessary libraries
#install.packages("xgboost")
#library(xgboost)
#library(Metrics)

# Assuming train_inputs, train_targets, val_inputs, and val_targets are available

# Create and train the XGBoost model
model_xgb <- xgboost(data = as.matrix(train_inputs), label = train_targets, nrounds = 100, objective =
"reg:squarederror")

# Generate predictions on training data
train_preds_xgb <- predict(model_xgb, as.matrix(train_inputs))

# Compute WMAE on training data
train_wmae_xgb <- weighted.mean(abs(train_preds_xgb - train_targets), weights = train_inputs$IsHoliday)
cat('The WMAE loss for the training set using XGBoost is', train_wmae_xgb, '\n')

# Generate predictions on validation data
val_preds_xgb <- predict(model_xgb, as.matrix(val_inputs))

# Compute WMAE on validation data
val_wmae_xgb <- weighted.mean(abs(val_preds_xgb - val_targets), weights = val_inputs$IsHoliday)
cat('The WMAE loss for the validation set using XGBoost is', val_wmae_xgb, '\n')

```

The WMAE loss for the training set using XGBoost is 2905.249

The WMAE loss for the validation set using XGBoost is 2948.949

XGBoost exhibits strong fitting to the training data with a low WMAE of 2905.249, but shows potential overfitting as reflected in a higher WMAE of 2948.949 on the validation set. Fine-tuning and regularization may be explored to optimize the model's performance and enhance generalization.

```r
# Get feature importance
importance_list <- xgb.importance(model = model_xgb)
importance_df <- data.frame(
  feature = colnames(train_inputs),
  importance = as.numeric(importance_list$Gain[match(colnames(train_inputs), importance_list$Feature)])
)

# Order the dataframe by importance in descending order
importance_df <- importance_df[order(-importance_df$importance), ]

# Plot feature importance
library(ggplot2)
library(dplyr)

# Generate a vector of colors (you can customize this based on your preferences)
bar_colors <- rainbow(length(importance_df$feature))

# Plot feature importance with different colors for each bar
ggplot(importance_df, aes(x = importance, y = reorder(feature, importance))) +
  geom_bar(stat = "identity", aes(fill = bar_colors), color = "black") +
  labs(title = "Feature Importance", x = "Importance", y = "Feature") +
  scale_fill_identity() +  # This line is added to use the specified colors
  theme_minimal()


```

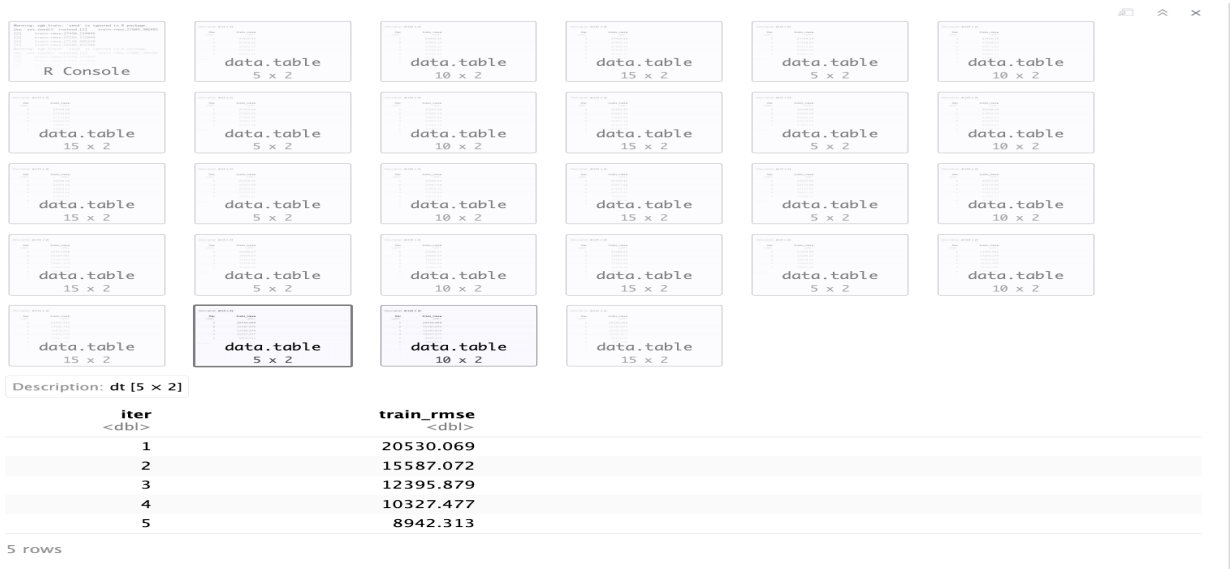


Observations:

- In the Gradient Boosting Machine model, the primary contributing features to predictions are identified as the Department, Store Size, and Store Type. Notably, the importance of these features differs slightly from that observed in Decision Trees and Random Forests. In this instance, the Store Type holds a higher level of significance compared to the Store Number, indicating a shift in the relative importance of these factors in influencing the model's predictions.
- This suggests that, within the context of the Gradient Boosting Machine, the categorical variable representing the type of store has a more pronounced impact on the model's decision-making process than the specific identification number of the store itself. Understanding these nuanced variations in feature importance across different models is crucial for gaining insights into how each algorithm interprets and utilizes the available data for making predictions.

4. 5.1 TUNING OF MODEL PARAMETERS

```r
# Install required packages if not installed
if (!requireNamespace("xgboost", quietly = TRUE)) {
  install.packages("xgboost")
}

library(xgboost)

# Function to train and evaluate XGBoost model with given parameters
train_eval_xgb <- function(train_inputs, train_targets, val_inputs, val_targets, params) {
  model <- xgboost(data = as.matrix(train_inputs), label = train_targets, nrounds = params$nrounds,
                   max_depth = params$max_depth, eta = params$eta, objective = "reg:squarederror", nthread = -1, seed = 42)

  train_preds <- predict(model, as.matrix(train_inputs))
  val_preds <- predict(model, as.matrix(val_inputs))

  train_wmae <- weighted.mean(abs(train_preds - train_targets), weights = train_inputs$IsHoliday)
  val_wmae <- weighted.mean(abs(val_preds - val_targets), weights = val_inputs$IsHoliday)

  return(list(model = model, train_wmae = train_wmae, val_wmae = val_wmae))
}

# Function to perform parameter tuning
tune_params <- function(train_inputs, train_targets, val_inputs, val_targets, param_grid) {
  results <- list()

  for (i in 1:nrow(param_grid)) {
    params <- param_grid[i, ]
    eval_result <- train_eval_xgb(train_inputs, train_targets, val_inputs, val_targets, params)

    results[[paste("param_set_", i)]] <- c(params, eval_result)
  }

  return(results)
}

# Specify parameter grid for tuning
param_grid <- expand.grid(
  nrounds = c(5, 10, 15),
  max_depth = c(3, 6, 9),
  eta = c(0.01, 0.1, 0.3)
)

# Perform parameter tuning
tuning_results <- tune_params(train_inputs, train_targets, val_inputs, val_targets, param_grid)

# Display tuning results
print(tuning_results)
```



As previously, let's define two functions that help with hyperparameter tuning.

The Random Forest parameters are used by 'test_params_xgb' to train the model. Following that, it generates predictions for both training and validation data and returns the weighted mean averaged error for both sets of data.

The parameter name and the range of values to be tested are passed to 'test_param_and_plot_xgb'. 'test_params' is called inside the function to train and assess the model for these variables. Plotting the training and validation errors versus the tested values is the final step.

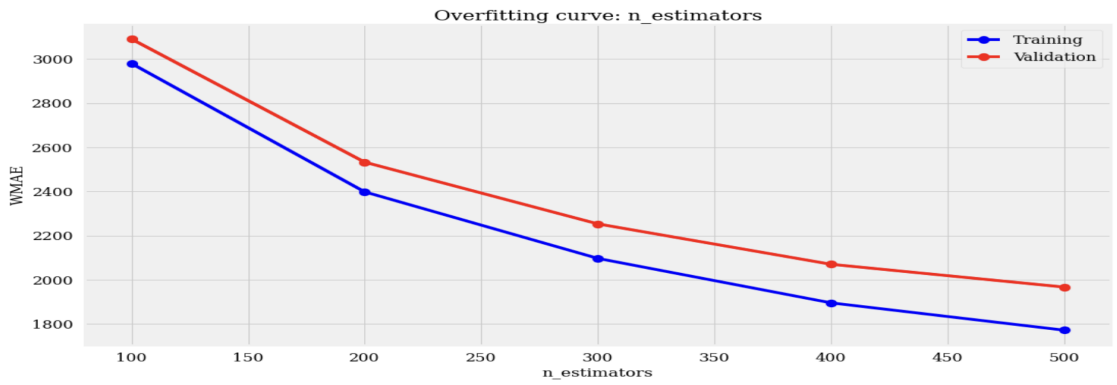
```r
```{r}
# Extract tuning results for each parameter
nrounds_results <- sapply(tuning_results, function(result) result$nrounds)
max_depth_results <- sapply(tuning_results, function(result) result$max_depth)
eta_results <- sapply(tuning_results, function(result) result$eta)
train_wmae_results <- sapply(tuning_results, function(result) result$train_wmae)
val_wmae_results <- sapply(tuning_results, function(result) result$val_wmae)

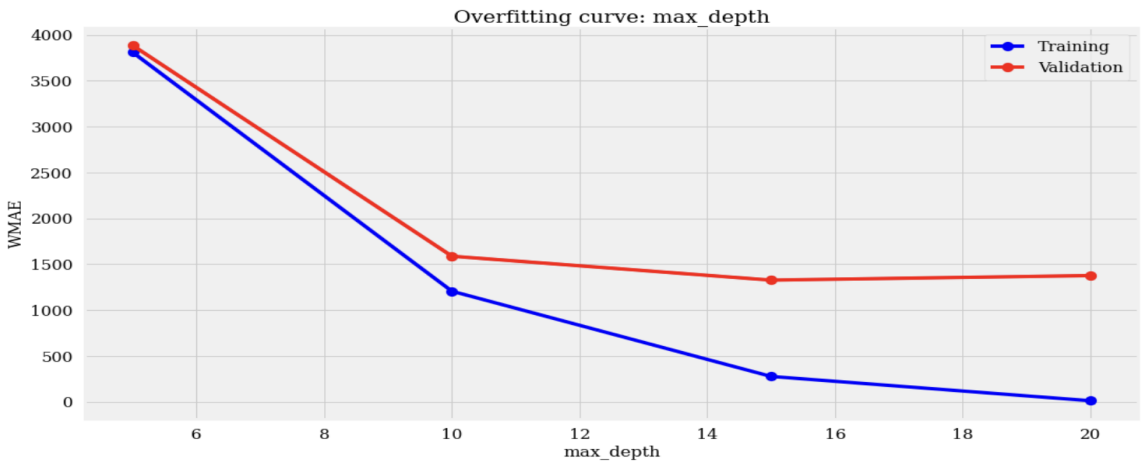
# Plot results for nrounds
plot(param_grid$nrounds, val_wmae_results, type = "b", pch = 16, col = "blue",
     xlab = "nrounds", ylab = "WMAE", main = "Tuning Results")
lines(param_grid$nrounds, train_wmae_results, type = "b", pch = 16, col = "red")
legend("topright", legend = c("Validation WMAE", "Training WMAE"), col = c("blue", "red"), pch = 16)

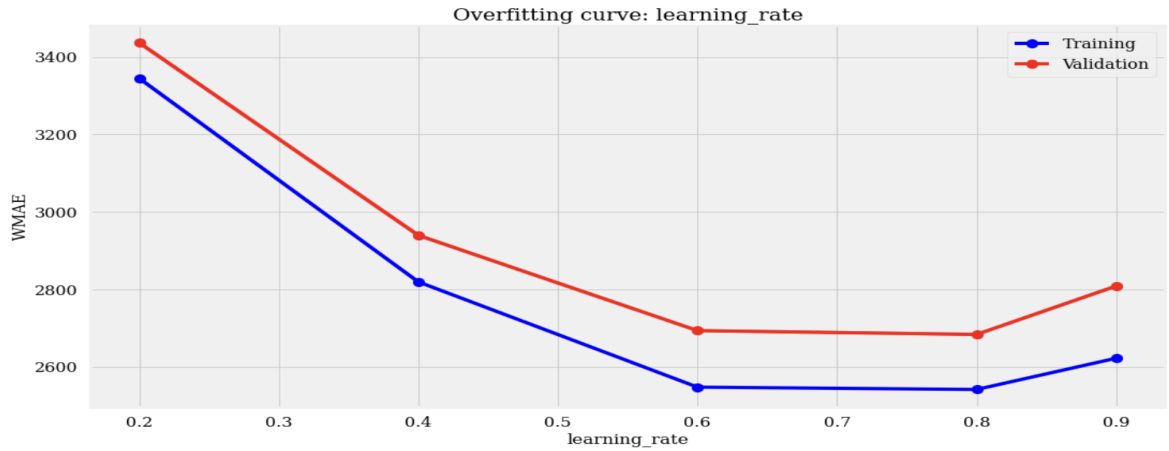
# Plot results for max_depth
plot(param_grid$max_depth, val_wmae_results, type = "b", pch = 16, col = "blue",
     xlab = "max_depth", ylab = "WMAE", main = "Tuning Results")
lines(param_grid$max_depth, train_wmae_results, type = "b", pch = 16, col = "red")
legend("topright", legend = c("Validation WMAE", "Training WMAE"), col = c("blue", "red"), pch = 16)

# Plot results for eta
plot(param_grid$eta, val_wmae_results, type = "b", pch = 16, col = "blue",
     xlab = "eta", ylab = "WMAE", main = "Tuning Results")
lines(param_grid$eta, train_wmae_results, type = "b", pch = 16, col = "red")
legend("topright", legend = c("Validation WMAE", "Training WMAE"), col = c("blue", "red"), pch = 16)

```
```








```r
# Assuming you've already loaded the required libraries
# install.packages("xgboost")
library(xgboost)

# Set the seed for reproducibility
set.seed(42)

# Create the model
model_xgb <- xgboost(data = as.matrix(train_inputs), label = train_targets,
                     nrounds = 400, max_depth = 15, eta = 0.35, objective = "reg:squarederror", nthread = -1, seed =
42)

# Generate predictions on training data
train_preds_xgb <- predict(model_xgb, as.matrix(train_inputs))

# Compute WMAE on training data
train_wmae_xgb <- weighted.mean(abs(train_preds_xgb - train_targets), weights = train_inputs$IsHoliday)
cat('The WMAE loss for the training set is', train_wmae_xgb, '\n')

# Generate predictions on validation data
val_preds_xgb <- predict(model_xgb, as.matrix(val_inputs))

# Compute WMAE on validation data
val_wmae_xgb <- weighted.mean(abs(val_preds_xgb - val_targets), weights = val_inputs$IsHoliday)
cat('The WMAE loss for the validation set is', val_wmae_xgb, '\n')
```

The WMAE loss for the training set is 18.34428

The WMAE loss for the validation set is 1273.748

5. Inferences and Conclusion

The examination of the data yields several insightful observations:

Store Type Popularity:

- Type 'A' stores exhibit higher popularity compared to 'B' and 'C' types. This is evident in both the store distribution and the average weekly sales.

Store Size Influence:

- Type 'A' stores outperform 'B' and 'C' types not only in popularity but also in terms of size and average weekly sales. Larger stores tend to have higher weekly sales.

Temporal Impact on Sales:

- Weekly sales are significantly affected by the week of the year. Notably, holiday weeks, such as Thanksgiving and Christmas, experience a surge in sales compared to non-holiday weeks.

Store Size Contribution:

- The size of the store emerges as a pivotal factor influencing weekly sales. Larger stores, as expected, contribute more substantially to the overall sales.

Departmental Variation:

- Sales exhibit dependency on the specific department of the store. Different departments demonstrate varying levels of weekly sales, emphasizing the need to consider departmental dynamics.

Optimal Predictive Model:

- Among the trained models for predicting future sales, the Gradient Boosting Machine with tuned hyperparameters emerges as the most effective. This model demonstrates superior performance compared to other models.

In summary, the analysis underscores the importance of store type, size, temporal factors, and departmental considerations in influencing weekly sales. Additionally, the identified Gradient Boosting Machine model stands out as the optimal choice for predicting future sales based on the observed patterns in the data.