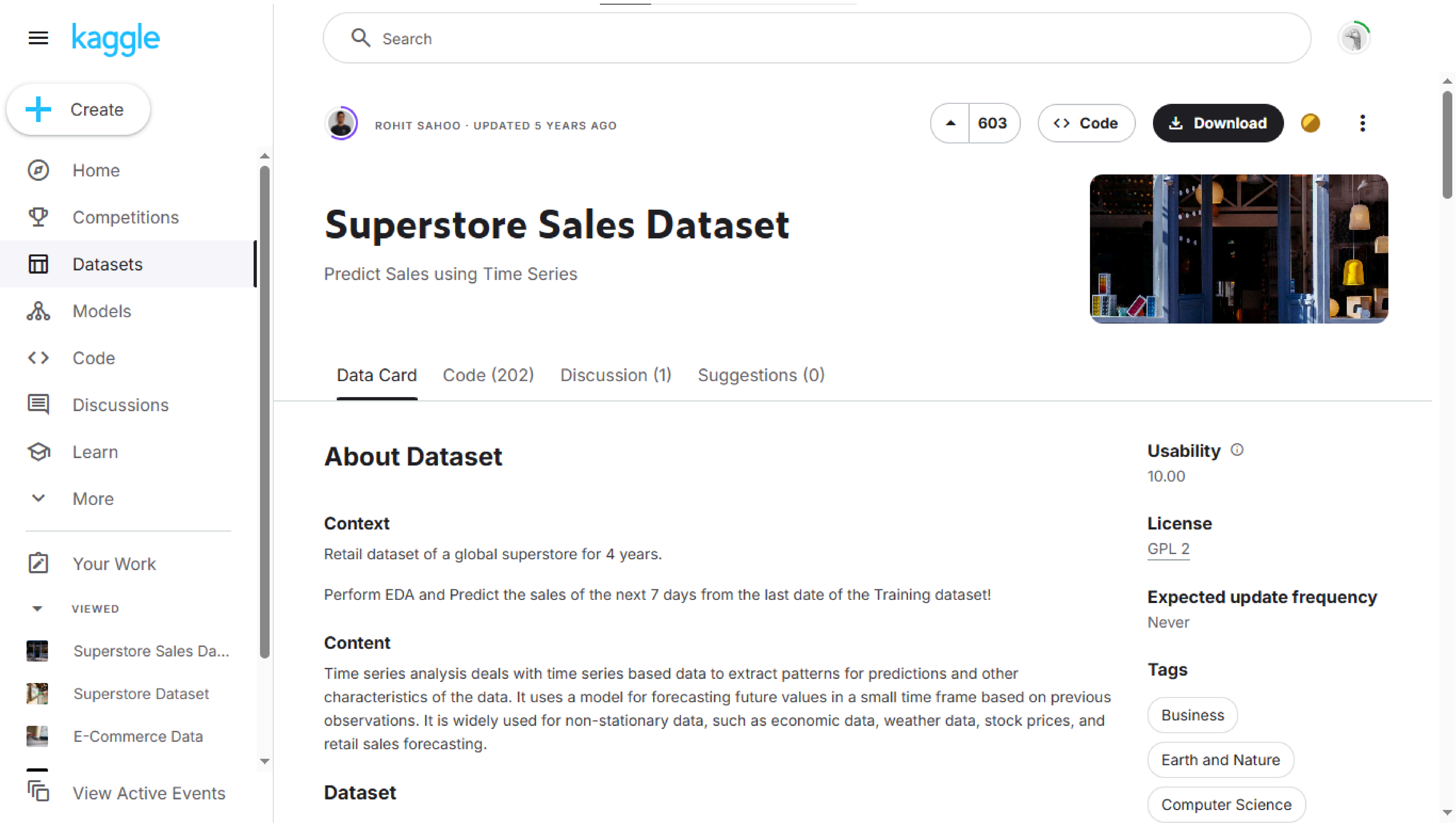
HULT
INTERNATIONAL
BUSINESS SCHOOL

Superstore Sales Dataset

Angel Lanto
MBAN 2025



ROHIT SAHOO · UPDATED 5 YEARS AGO

Superstore Sales Dataset

Predict Sales using Time Series

About Dataset

Context

Retail dataset of a global superstore for 4 years.

Perform EDA and Predict the sales of the next 7 days from the last date of the Training dataset!

Content

Time series analysis deals with time series based data to extract patterns for predictions and other characteristics of the data. It uses a model for forecasting future values in a small time frame based on previous observations. It is widely used for non-stationary data, such as economic data, weather data, stock prices, and retail sales forecasting.

Dataset

Usability
10.00

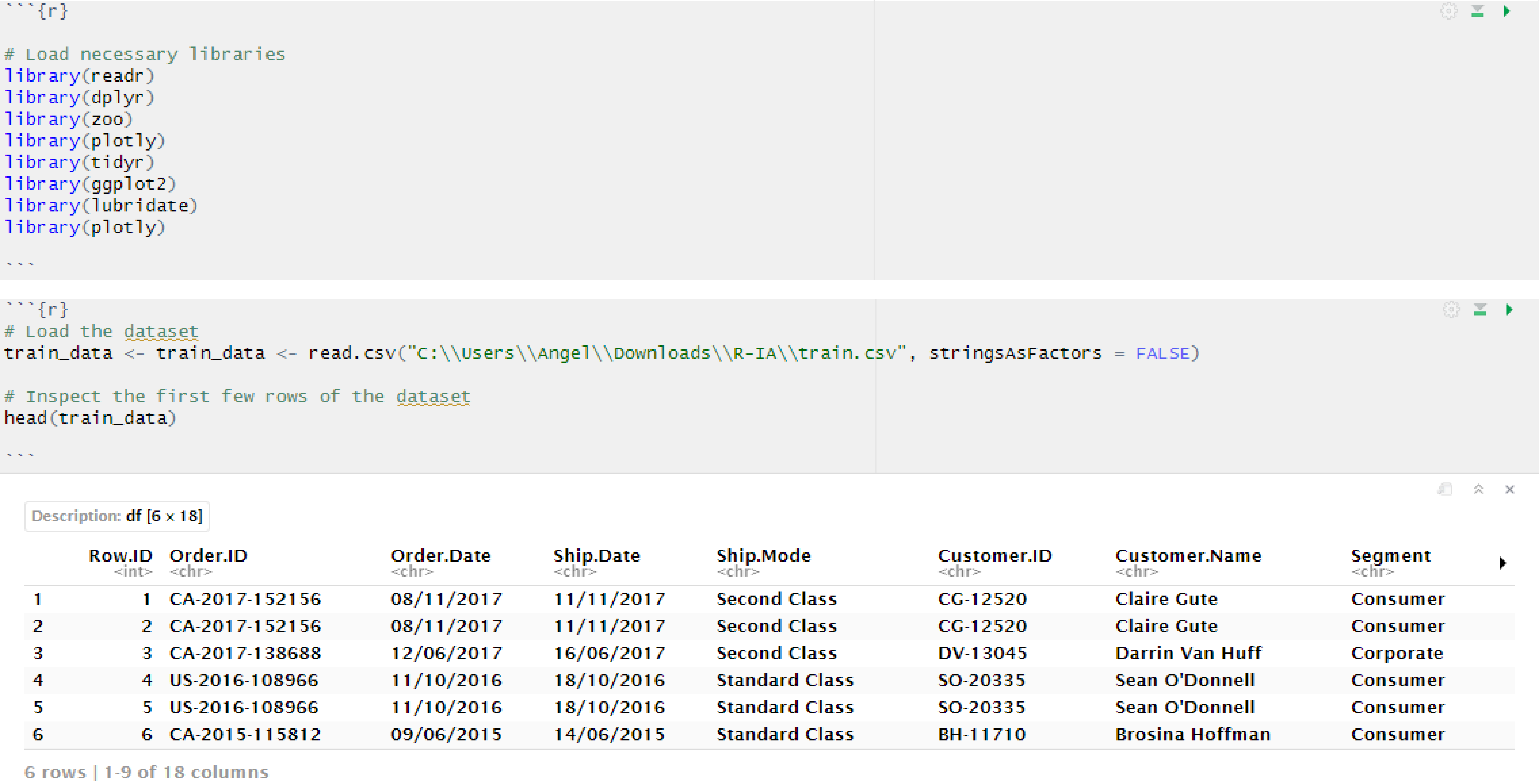
License
GPL 2

Expected update frequency
Never

Tags
Business
Earth and Nature
Computer Science

Kaggle Superstore Sales Dataset

Data Loading and Exploration



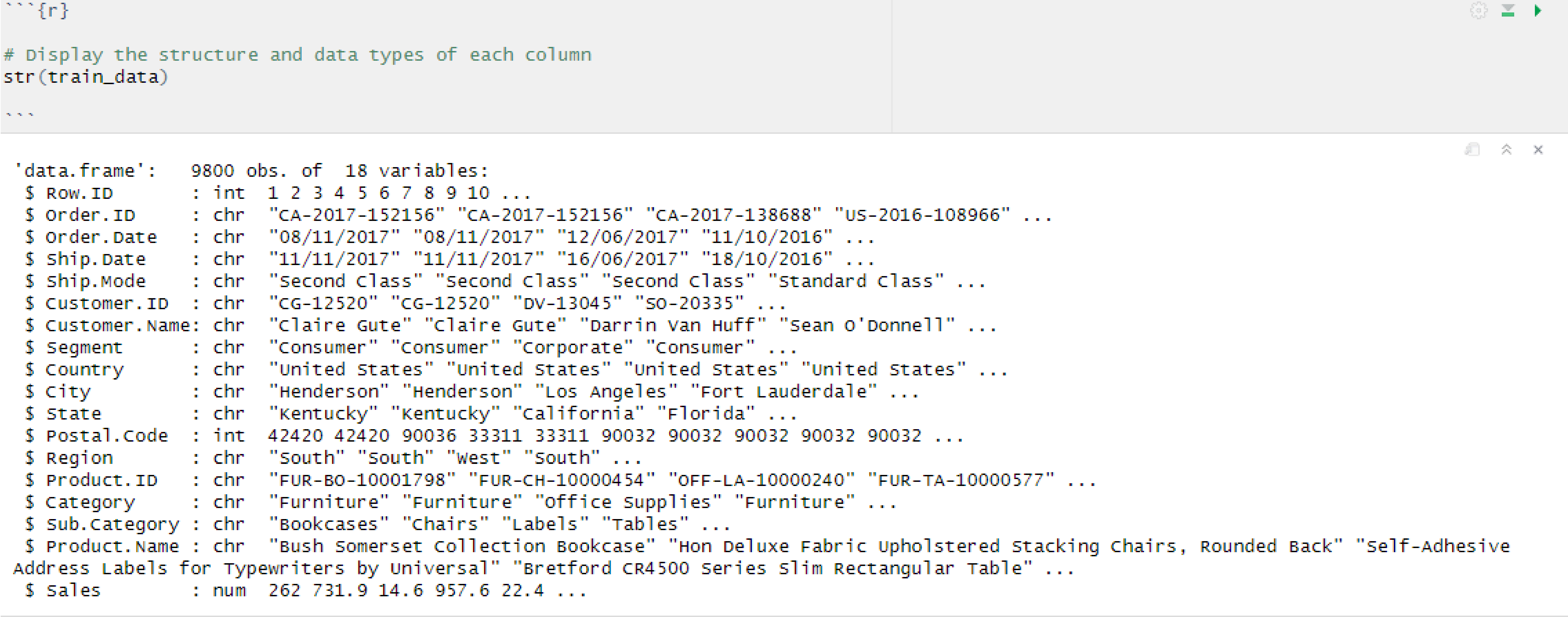
~~~
```{r}

# Load necessary libraries
library(readr)
library(dplyr)
library(zoo)
library(plotly)
library(tidyr)
library(ggplot2)
library(lubridate)
library(plotly)

```
~~~

~~~
```{r}
# Load the dataset
train_data <- train_data <- read.csv("C:\\Users\\Angel\\Downloads\\R-IA\\train.csv", stringsAsFactors = FALSE)

# Inspect the first few rows of the dataset
head(train_data)

```
~~~

Description: df [6 x 18]

	Row.ID <int>	Order.ID <chr>	Order.Date <chr>	Ship.Date <chr>	Ship.Mode <chr>	Customer.ID <chr>	Customer.Name <chr>	Segment <chr>
1	1	CA-2017-152156	08/11/2017	11/11/2017	Second Class	CG-12520	Claire Gute	Consumer
2	2	CA-2017-152156	08/11/2017	11/11/2017	Second Class	CG-12520	Claire Gute	Consumer
3	3	CA-2017-138688	12/06/2017	16/06/2017	Second Class	DV-13045	Darrin Van Huff	Corporate
4	4	US-2016-108966	11/10/2016	18/10/2016	Standard Class	SO-20335	Sean O'Donnell	Consumer
5	5	US-2016-108966	11/10/2016	18/10/2016	Standard Class	SO-20335	Sean O'Donnell	Consumer
6	6	CA-2015-115812	09/06/2015	14/06/2015	Standard Class	BH-11710	Brosina Hoffman	Consumer

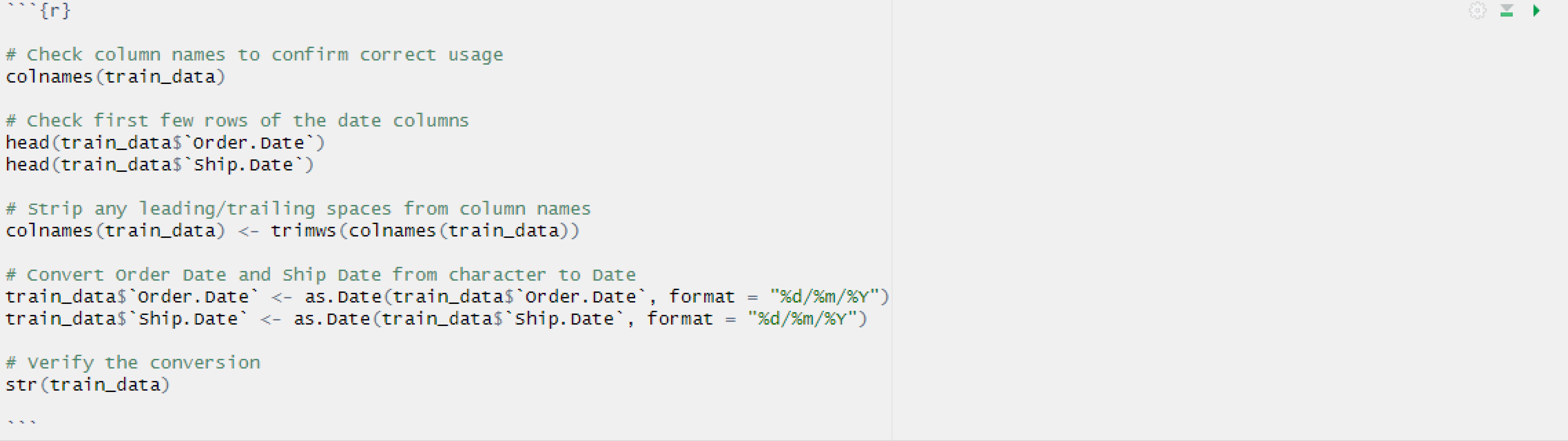
6 rows | 1-9 of 18 columns



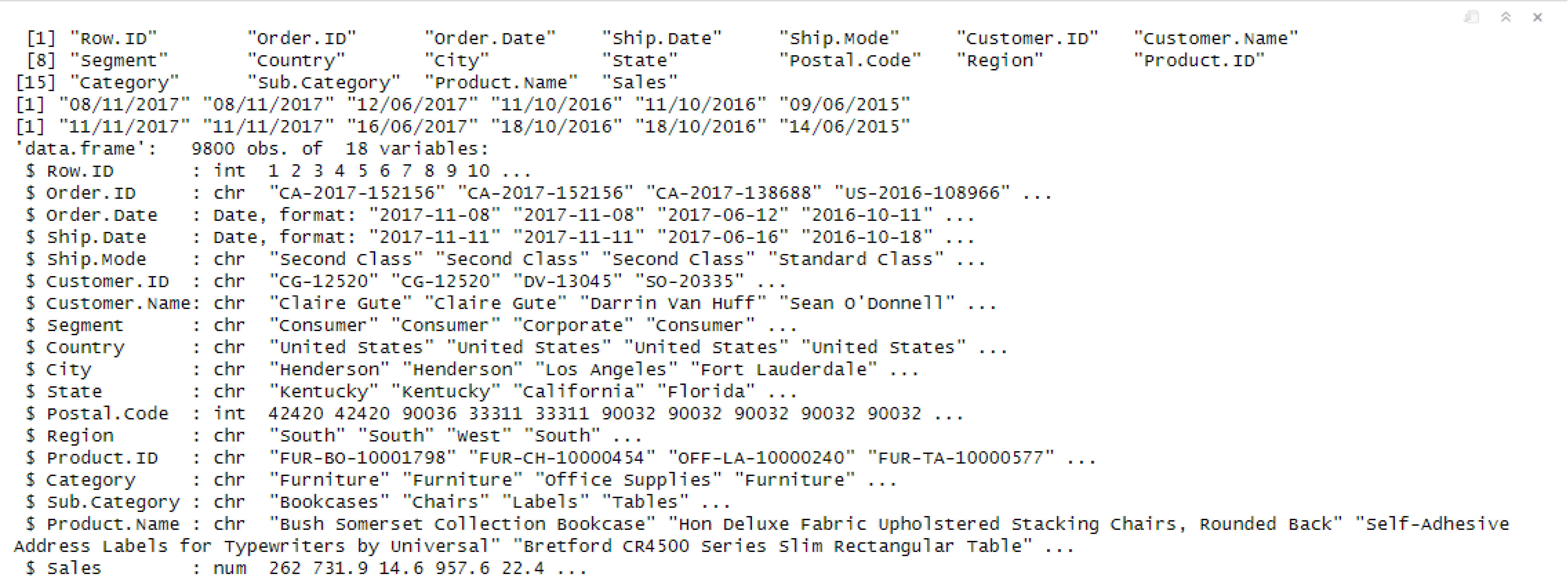
~~~
```{r}

# Display the structure and data types of each column
str(train_data)

```
~~~

```
'data.frame':   9800 obs. of  18 variables:
 $ Row.ID       : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Order.ID     : chr  "CA-2017-152156" "CA-2017-152156" "CA-2017-138688" "US-2016-108966" ...
 $ Order.Date   : chr  "08/11/2017" "08/11/2017" "12/06/2017" "11/10/2016" ...
 $ Ship.Date    : chr  "11/11/2017" "11/11/2017" "16/06/2017" "18/10/2016" ...
 $ Ship.Mode    : chr  "Second Class" "Second Class" "Second Class" "Standard Class" ...
 $ Customer.ID  : chr  "CG-12520" "CG-12520" "DV-13045" "SO-20335" ...
 $ Customer.Name: chr  "Claire Gute" "Claire Gute" "Darrin Van Huff" "Sean O'Donnell" ...
 $ Segment      : chr  "Consumer" "Consumer" "Corporate" "Consumer" ...
 $ Country      : chr  "United States" "United States" "United States" "United States" ...
 $ City         : chr  "Henderson" "Henderson" "Los Angeles" "Fort Lauderdale" ...
 $ State        : chr  "Kentucky" "Kentucky" "California" "Florida" ...
 $ Postal.Code  : int  42420 42420 90036 33311 33311 90032 90032 90032 90032 90032 ...
 $ Region       : chr  "South" "South" "West" "South" ...
 $ Product.ID   : chr  "FUR-BO-10001798" "FUR-CH-10000454" "OFF-LA-10000240" "FUR-TA-10000577" ...
 $ Category     : chr  "Furniture" "Furniture" "Office Supplies" "Furniture" ...
 $ Sub.Category : chr  "Bookcases" "Chairs" "Labels" "Tables" ...
 $ Product.Name : chr  "Bush Somerset Collection Bookcase" "Hon Deluxe Fabric Upholstered Stacking Chairs, Rounded Back" "Self-Adhesive
Address Labels for Typewriters by Universal" "Bretford CR4500 Series Slim Rectangular Table" ...
 $ Sales        : num  262 731.9 14.6 957.6 22.4 ...
```

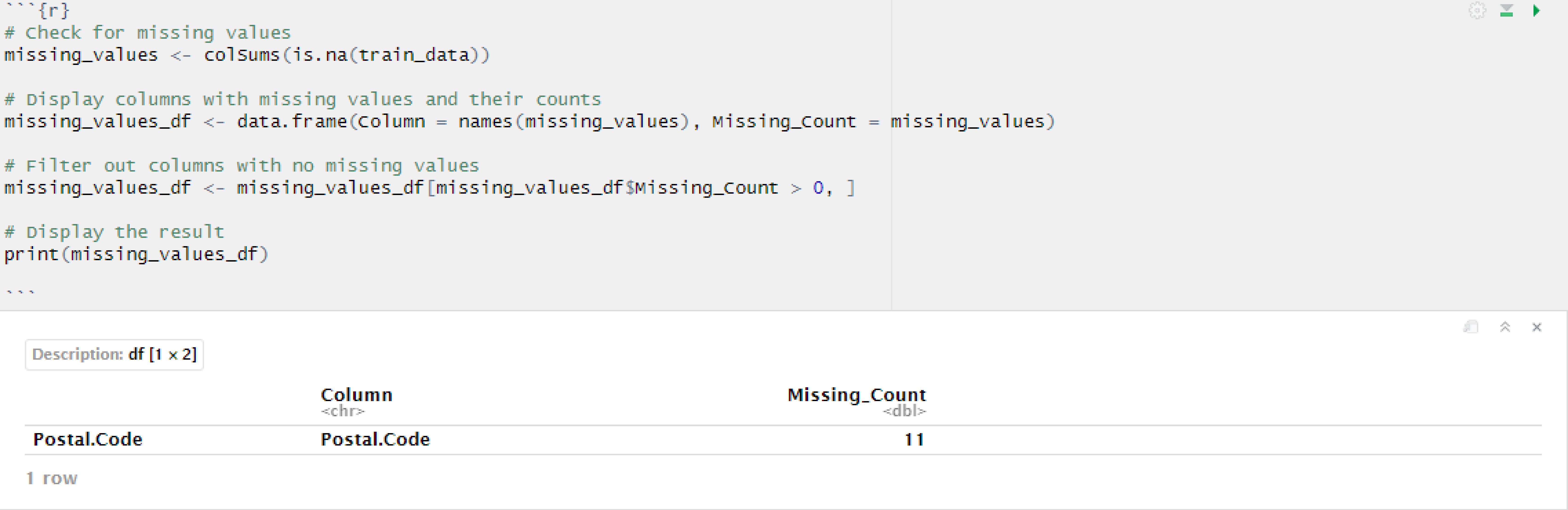
Data Cleaning and Handling Missing Values



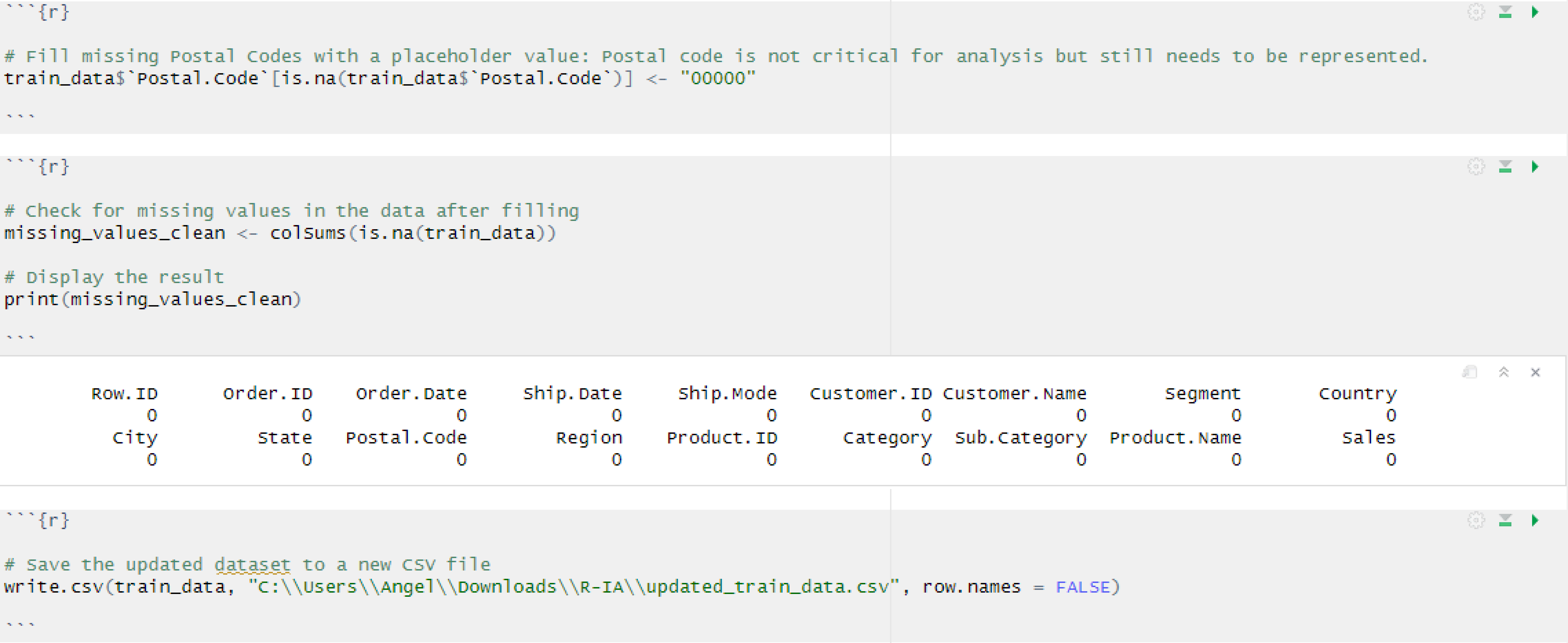
~~~
```{r}

# Check column names to confirm correct usage
colnames(train_data)

# Check first few rows of the date columns
head(train_data$`Order.Date`)
head(train_data$`Ship.Date`)

# Strip any leading/trailing spaces from column names
colnames(train_data) <- trimws(colnames(train_data))

# Convert Order Date and Ship Date from character to Date
train_data$`Order.Date` <- as.Date(train_data$`Order.Date`, format = "%d/%m/%Y")
train_data$`Ship.Date` <- as.Date(train_data$`Ship.Date`, format = "%d/%m/%Y")

# Verify the conversion
str(train_data)

```
~~~



```
 [1] "Row.ID"        "Order.ID"      "Order.Date"    "Ship.Date"     "Ship.Mode"     "Customer.ID"   "Customer.Name"
 [8] "Segment"       "Country"       "City"          "State"         "Postal.Code"   "Region"        "Product.ID"
[15] "Category"      "Sub.Category"  "Product.Name"  "Sales"
[1] "08/11/2017" "08/11/2017" "12/06/2017" "11/10/2016" "11/10/2016" "09/06/2015"
[1] "11/11/2017" "11/11/2017" "16/06/2017" "18/10/2016" "18/10/2016" "14/06/2015"
'data.frame':	9800 obs. of  18 variables:
 $ Row.ID       : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Order.ID     : chr  "CA-2017-152156" "CA-2017-152156" "CA-2017-138688" "US-2016-108966" ...
 $ Order.Date   : Date, format: "2017-11-08" "2017-11-08" "2017-06-12" "2016-10-11" ...
 $ Ship.Date    : Date, format: "2017-11-11" "2017-11-11" "2017-06-16" "2016-10-18" ...
 $ Ship.Mode    : chr  "Second Class" "Second Class" "Second Class" "Standard Class" ...
 $ Customer.ID  : chr  "CG-12520" "CG-12520" "DV-13045" "SO-20335" ...
 $ Customer.Name: chr  "Claire Gute" "Claire Gute" "Darrin Van Huff" "Sean O'Donnell" ...
 $ Segment      : chr  "Consumer" "Consumer" "Corporate" "Consumer" ...
 $ Country      : chr  "United States" "United States" "United States" "United States" ...
 $ City         : chr  "Henderson" "Henderson" "Los Angeles" "Fort Lauderdale" ...
 $ State        : chr  "Kentucky" "Kentucky" "California" "Florida" ...
 $ Postal.Code  : int  42420 42420 90036 33311 33311 90032 90032 90032 90032 90032 ...
 $ Region       : chr  "South" "South" "West" "South" ...
 $ Product.ID   : chr  "FUR-BO-10001798" "FUR-CH-10000454" "OFF-LA-10000240" "FUR-TA-10000577" ...
 $ Category     : chr  "Furniture" "Furniture" "Office Supplies" "Furniture" ...
 $ Sub.Category : chr  "Bookcases" "Chairs" "Labels" "Tables" ...
 $ Product.Name : chr  "Bush Somerset Collection Bookcase" "Hon Deluxe Fabric Upholstered Stacking Chairs, Rounded Back" "Self-Adhesive
Address Labels for Typewriters by Universal" "Bretford CR4500 Series Slim Rectangular Table" ...
 $ Sales        : num  262 731.9 14.6 957.6 22.4 ...
```



~~~r
```{r}
# Check for missing values
missing_values <- colSums(is.na(train_data))

# Display columns with missing values and their counts
missing_values_df <- data.frame(Column = names(missing_values), Missing_Count = missing_values)

# Filter out columns with no missing values
missing_values_df <- missing_values_df[missing_values_df$Missing_Count > 0, ]

# Display the result
print(missing_values_df)

~~~
```

Description: df [1 × 2]

	Column <chr>	Missing_Count <dbl>
Postal.Code	Postal.Code	11

1 row
```



~~~
```{r}

# Fill missing Postal Codes with a placeholder value: Postal code is not critical for analysis but still needs to be represented.
train_data$`Postal.Code`[is.na(train_data$`Postal.Code`)] <- "00000"

```
~~~

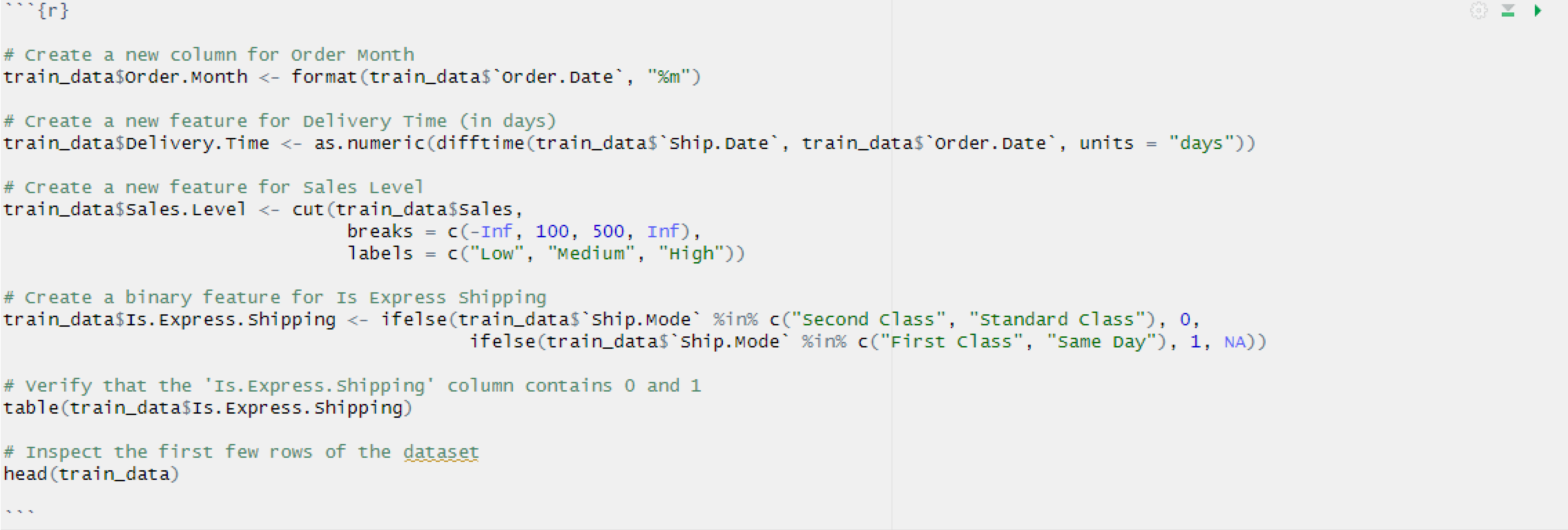
~~~
```{r}

# Check for missing values in the data after filling
missing_values_clean <- colSums(is.na(train_data))

# Display the result
print(missing_values_clean)

```
~~~

```
       Row.ID      Order.ID    Order.Date     Ship.Date     Ship.Mode   Customer.ID Customer.Name       Segment       Country
            0             0             0             0             0             0             0             0             0
         City         State   Postal.Code        Region    Product.ID      Category  Sub.Category  Product.Name         Sales
            0             0             0             0             0             0             0             0             0
```

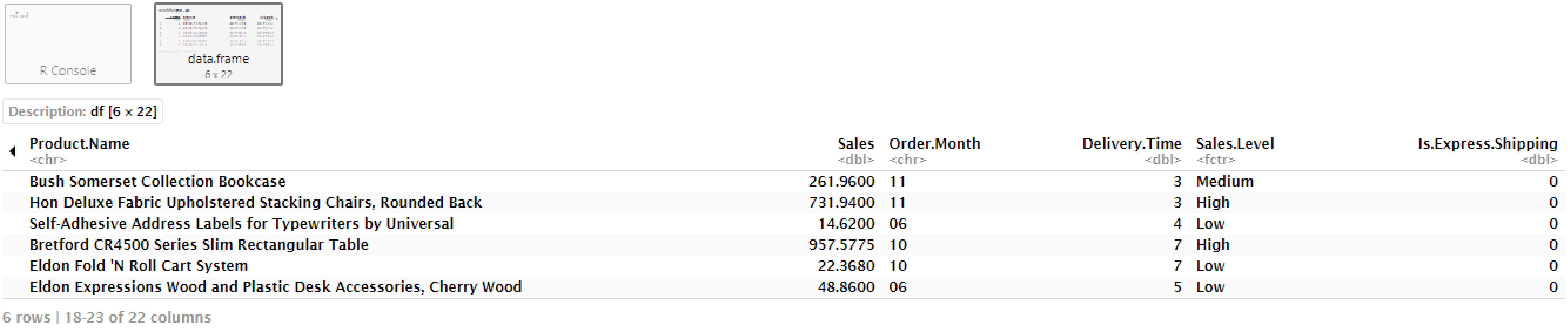
~~~
```{r}

# Save the updated dataset to a new CSV file
write.csv(train_data, "C:\\Users\\Angel\\Downloads\\R-IA\\updated_train_data.csv", row.names = FALSE)

```
~~~

HULT
INTERNATIONAL
BUSINESS SCHOOL

Data Transformation and Feature Engineering



~~~
```{r}

# Create a new column for Order Month
train_data$Order.Month <- format(train_data$`Order.Date`, "%m")

# Create a new feature for Delivery Time (in days)
train_data$Delivery.Time <- as.numeric(difftime(train_data$`Ship.Date`, train_data$`Order.Date`, units = "days"))

# Create a new feature for Sales Level
train_data$Sales.Level <- cut(train_data$Sales,
                               breaks = c(-Inf, 100, 500, Inf),
                               labels = c("Low", "Medium", "High"))

# Create a binary feature for Is Express Shipping
train_data$Is.Express.Shipping <- ifelse(train_data$`Ship.Mode` %in% c("Second Class", "Standard Class"), 0,
                                          ifelse(train_data$`Ship.Mode` %in% c("First Class", "Same Day"), 1, NA))

# Verify that the 'Is.Express.Shipping' column contains 0 and 1
table(train_data$Is.Express.Shipping)

# Inspect the first few rows of the dataset
head(train_data)

```
~~~



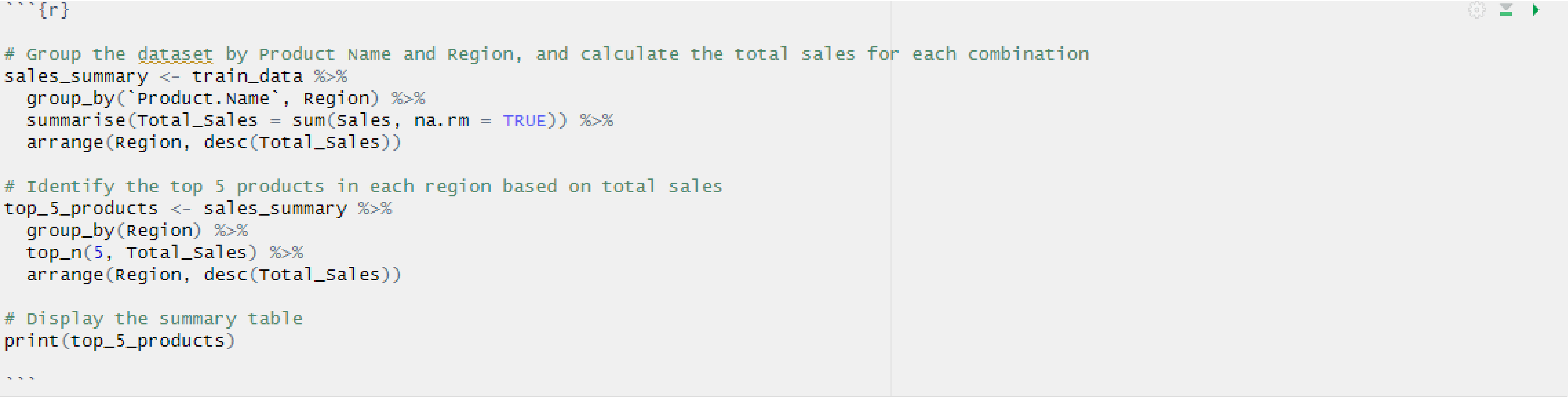
Description: df [6 × 22]

Product.Name	Sales	Order.Month	Delivery.Time	Sales.Level	Is.Express.Shipping
<chr>	<dbl>	<chr>	<dbl>	<fctr>	<dbl>
Bush Somerset Collection Bookcase	261.9600	11	3	Medium	0
Hon Deluxe Fabric Upholstered Stacking Chairs, Rounded Back	731.9400	11	3	High	0
Self-Adhesive Address Labels for Typewriters by Universal	14.6200	06	4	Low	0
Bretford CR4500 Series Slim Rectangular Table	957.5775	10	7	High	0
Eldon Fold 'N Roll Cart System	22.3680	10	7	Low	0
Eldon Expressions Wood and Plastic Desk Accessories, Cherry Wood	48.8600	06	5	Low	0

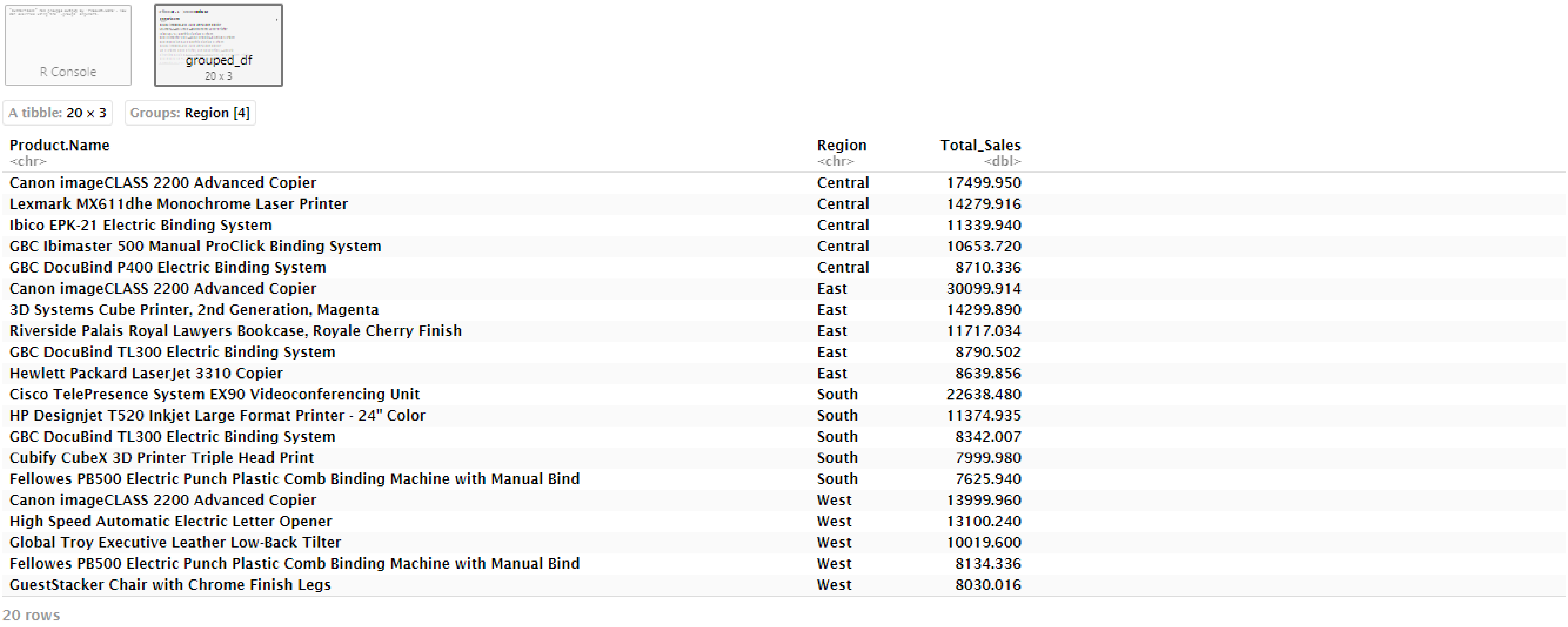
6 rows | 18-23 of 22 columns

HULT
INTERNATIONAL
BUSINESS SCHOOL

Grouping and Aggregation



```{r}

# Group the dataset by Product Name and Region, and calculate the total sales for each combination
sales_summary <- train_data %>%
  group_by(`Product.Name`, Region) %>%
  summarise(Total_Sales = sum(Sales, na.rm = TRUE)) %>%
  arrange(Region, desc(Total_Sales))

# Identify the top 5 products in each region based on total sales
top_5_products <- sales_summary %>%
  group_by(Region) %>%
  top_n(5, Total_Sales) %>%
  arrange(Region, desc(Total_Sales))

# Display the summary table
print(top_5_products)

```



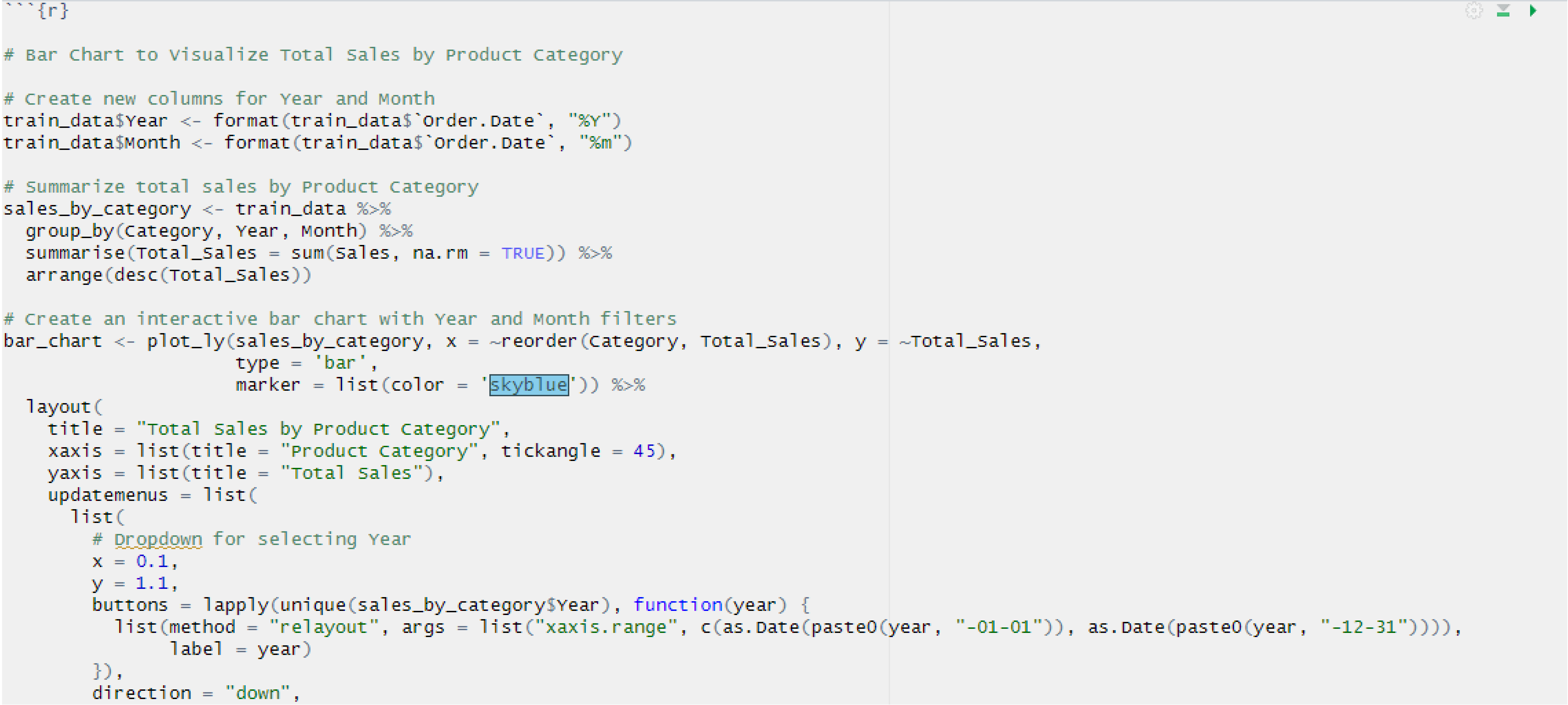
A tibble: 20 × 3 Groups: Region [4]

Product.Name <chr>	Region <chr>	Total_Sales <dbl>
Canon imageCLASS 2200 Advanced Copier	Central	17499.950
Lexmark MX611dhe Monochrome Laser Printer	Central	14279.916
Ibico EPK-21 Electric Binding System	Central	11339.940
GBC Ibimaster 500 Manual ProClick Binding System	Central	10653.720
GBC DocuBind P400 Electric Binding System	Central	8710.336
Canon imageCLASS 2200 Advanced Copier	East	30099.914
3D Systems Cube Printer, 2nd Generation, Magenta	East	14299.890
Riverside Palais Royal Lawyers Bookcase, Royale Cherry Finish	East	11717.034
GBC DocuBind TL300 Electric Binding System	East	8790.502
Hewlett Packard LaserJet 3310 Copier	East	8639.856
Cisco TelePresence System EX90 Videoconferencing Unit	South	22638.480
HP Designjet T520 Inkjet Large Format Printer - 24" Color	South	11374.935
GBC DocuBind TL300 Electric Binding System	South	8342.007
Cubify CubeX 3D Printer Triple Head Print	South	7999.980
Fellowes PB500 Electric Punch Plastic Comb Binding Machine with Manual Bind	South	7625.940
Canon imageCLASS 2200 Advanced Copier	West	13999.960
High Speed Automatic Electric Letter Opener	West	13100.240
Global Troy Executive Leather Low-Back Tilter	West	10019.600
Fellowes PB500 Electric Punch Plastic Comb Binding Machine with Manual Bind	West	8134.336
GuestStacker Chair with Chrome Finish Legs	West	8030.016

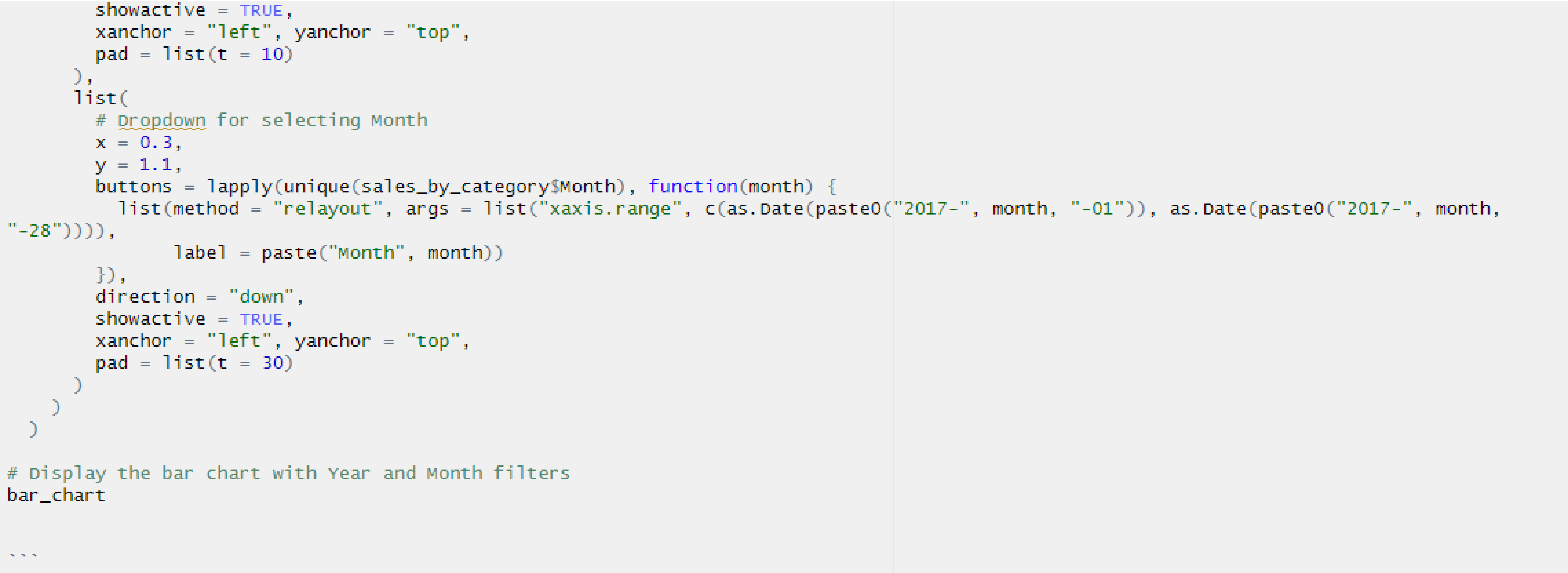
20 rows

HULT
INTERNATIONAL
BUSINESS SCHOOL

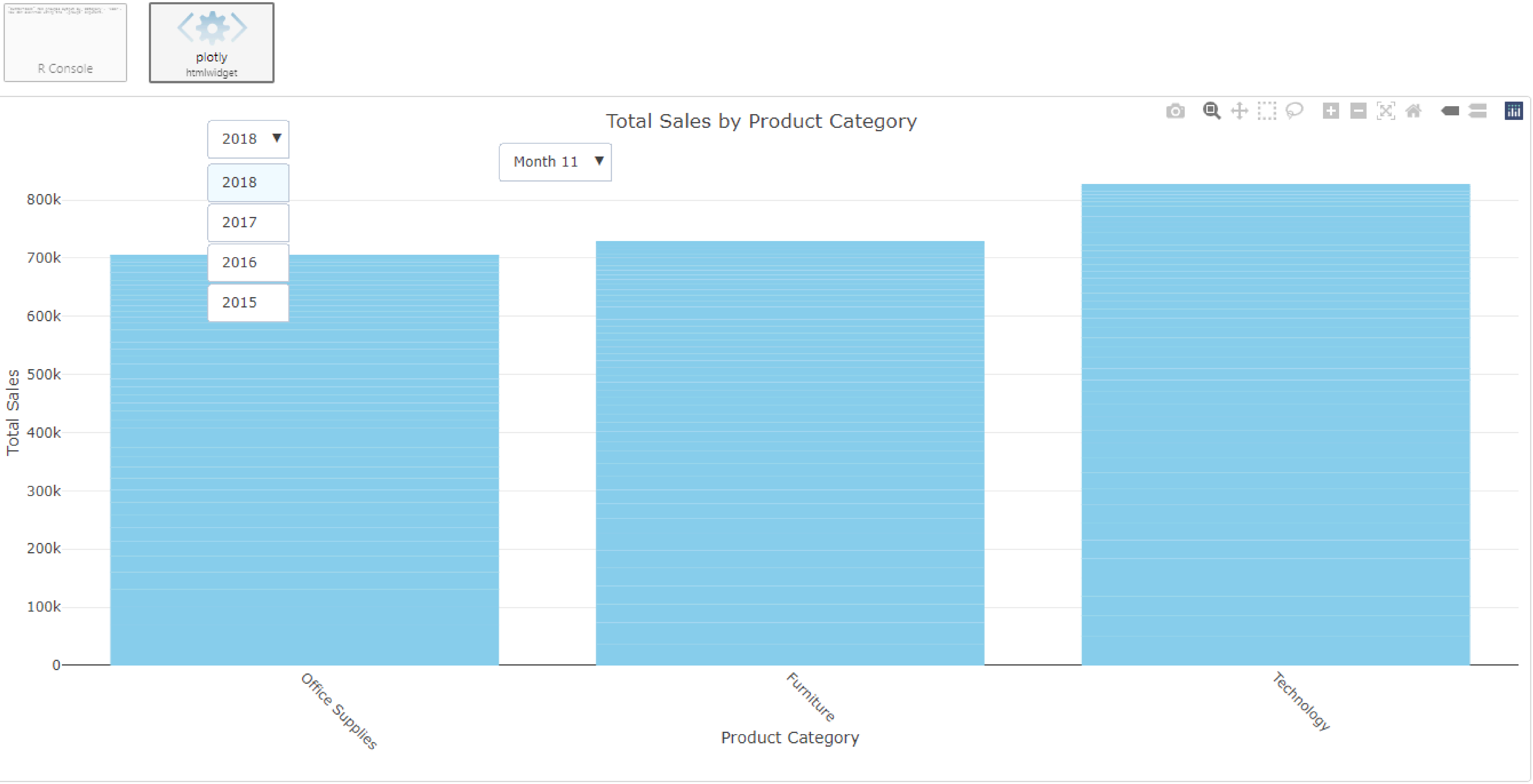
Data Visualization

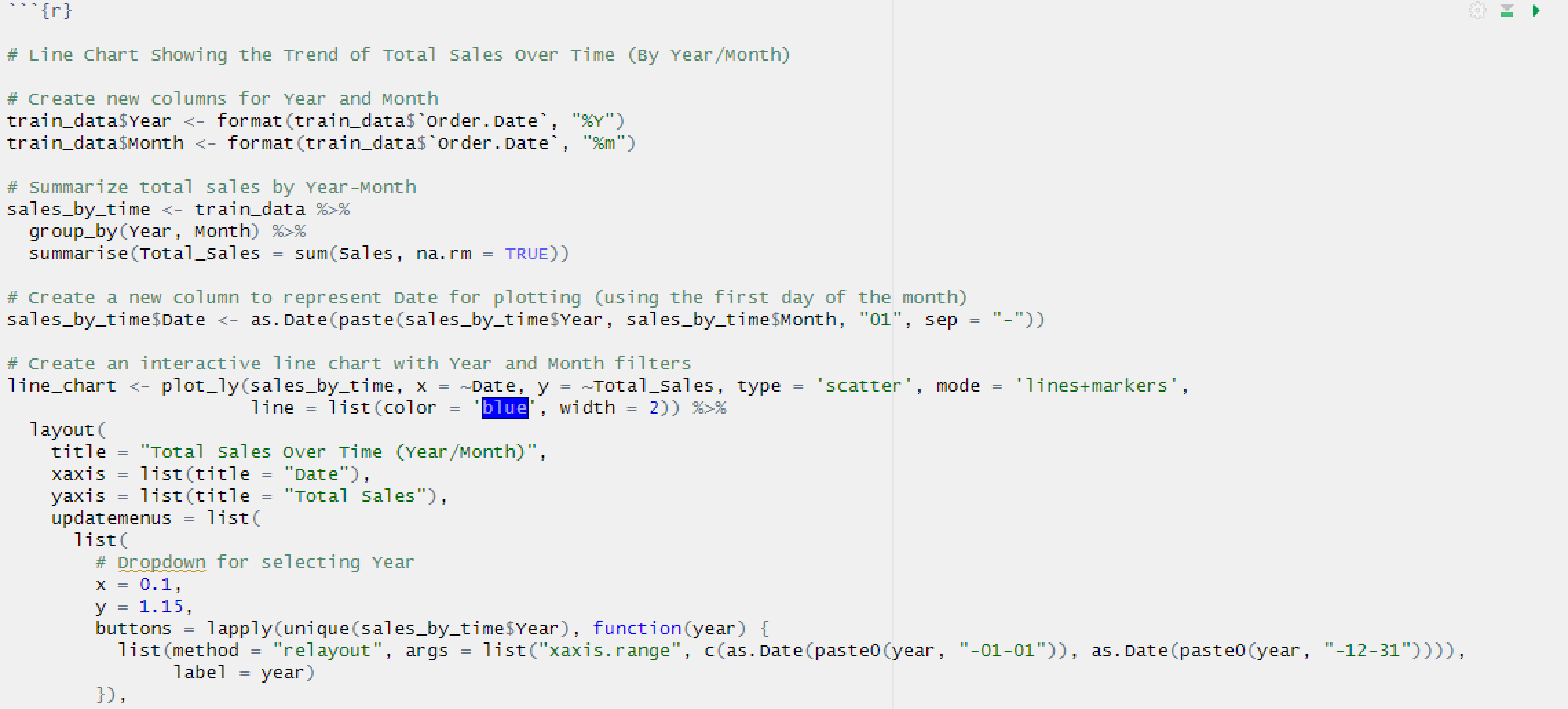


~~~r
```{r}

# Bar Chart to Visualize Total Sales by Product Category

# Create new columns for Year and Month
train_data$Year <- format(train_data$`Order.Date`, "%Y")
train_data$Month <- format(train_data$`Order.Date`, "%m")

# Summarize total sales by Product Category
sales_by_category <- train_data %>%
  group_by(Category, Year, Month) %>%
  summarise(Total_Sales = sum(Sales, na.rm = TRUE)) %>%
  arrange(desc(Total_Sales))

# Create an interactive bar chart with Year and Month filters
bar_chart <- plot_ly(sales_by_category, x = ~reorder(Category, Total_Sales), y = ~Total_Sales,
                     type = 'bar',
                     marker = list(color = 'skyblue')) %>%
  layout(
    title = "Total Sales by Product Category",
    xaxis = list(title = "Product Category", tickangle = 45),
    yaxis = list(title = "Total Sales"),
    updatemenus = list(
      list(
        # Dropdown for selecting Year
        x = 0.1,
        y = 1.1,
        buttons = lapply(unique(sales_by_category$Year), function(year) {
          list(method = "relayout", args = list("xaxis.range", c(as.Date(paste0(year, "-01-01")), as.Date(paste0(year, "-12-31")))),
               label = year)
        }),
        direction = "down",
~~~

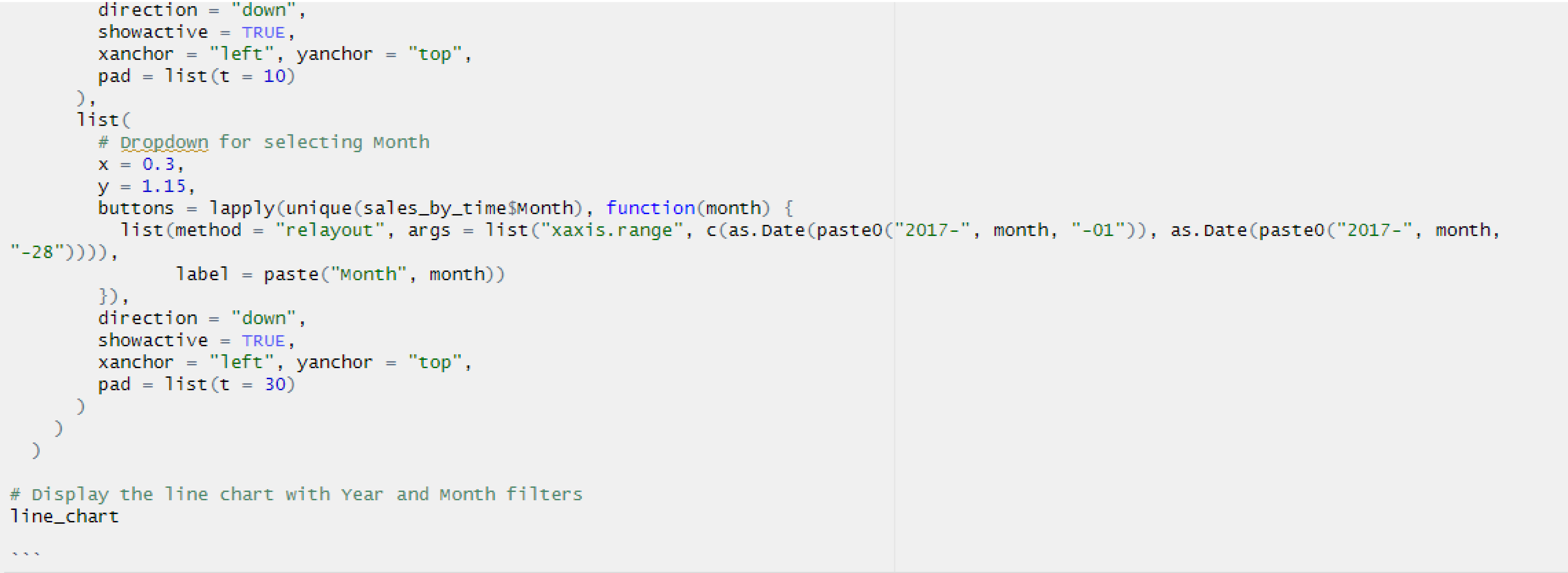


```
        showactive = TRUE,
        xanchor = "left", yanchor = "top",
        pad = list(t = 10)
      ),
      list(
        # Dropdown for selecting Month
        x = 0.3,
        y = 1.1,
        buttons = lapply(unique(sales_by_category$Month), function(month) {
          list(method = "relayout", args = list("xaxis.range", c(as.Date(paste0("2017-", month, "-01")), as.Date(paste0("2017-", month,
"-28")))),
               label = paste("Month", month))
        }),
        direction = "down",
        showactive = TRUE,
        xanchor = "left", yanchor = "top",
        pad = list(t = 30)
      )
    )
  )

# Display the bar chart with Year and Month filters
bar_chart


```
```
```

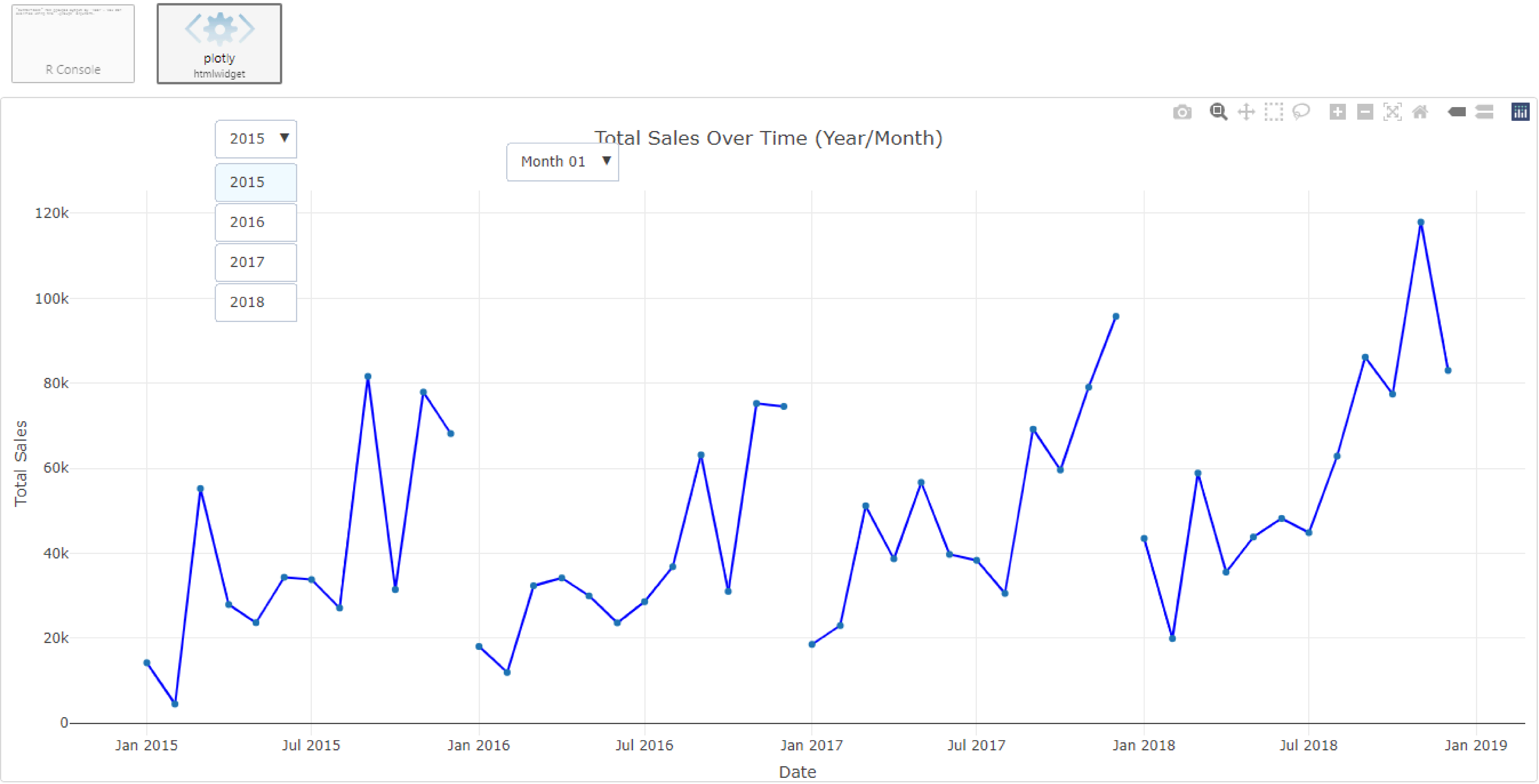

R Console
plotly
htmlwidget
Total Sales by Product Category
2018 ▼
2018
2017
2016
2015
Month 11 ▼
800k
700k
600k
500k
400k
300k
200k
100k
0
Total Sales
Office Supplies
Furniture
Technology
Product Category



~~~
```{r}

# Line Chart Showing the Trend of Total Sales Over Time (By Year/Month)

# Create new columns for Year and Month
train_data$Year <- format(train_data$`Order.Date`, "%Y")
train_data$Month <- format(train_data$`Order.Date`, "%m")

# Summarize total sales by Year-Month
sales_by_time <- train_data %>%
  group_by(Year, Month) %>%
  summarise(Total_Sales = sum(Sales, na.rm = TRUE))

# Create a new column to represent Date for plotting (using the first day of the month)
sales_by_time$Date <- as.Date(paste(sales_by_time$Year, sales_by_time$Month, "01", sep = "-"))

# Create an interactive line chart with Year and Month filters
line_chart <- plot_ly(sales_by_time, x = ~Date, y = ~Total_Sales, type = 'scatter', mode = 'lines+markers',
                      line = list(color = 'blue', width = 2)) %>%
  layout(
    title = "Total Sales Over Time (Year/Month)",
    xaxis = list(title = "Date"),
    yaxis = list(title = "Total Sales"),
    updatemenus = list(
      list(
        # Dropdown for selecting Year
        x = 0.1,
        y = 1.15,
        buttons = lapply(unique(sales_by_time$Year), function(year) {
          list(method = "relayout", args = list("xaxis.range", c(as.Date(paste0(year, "-01-01")), as.Date(paste0(year, "-12-31")))),
               label = year)
        }),
~~~

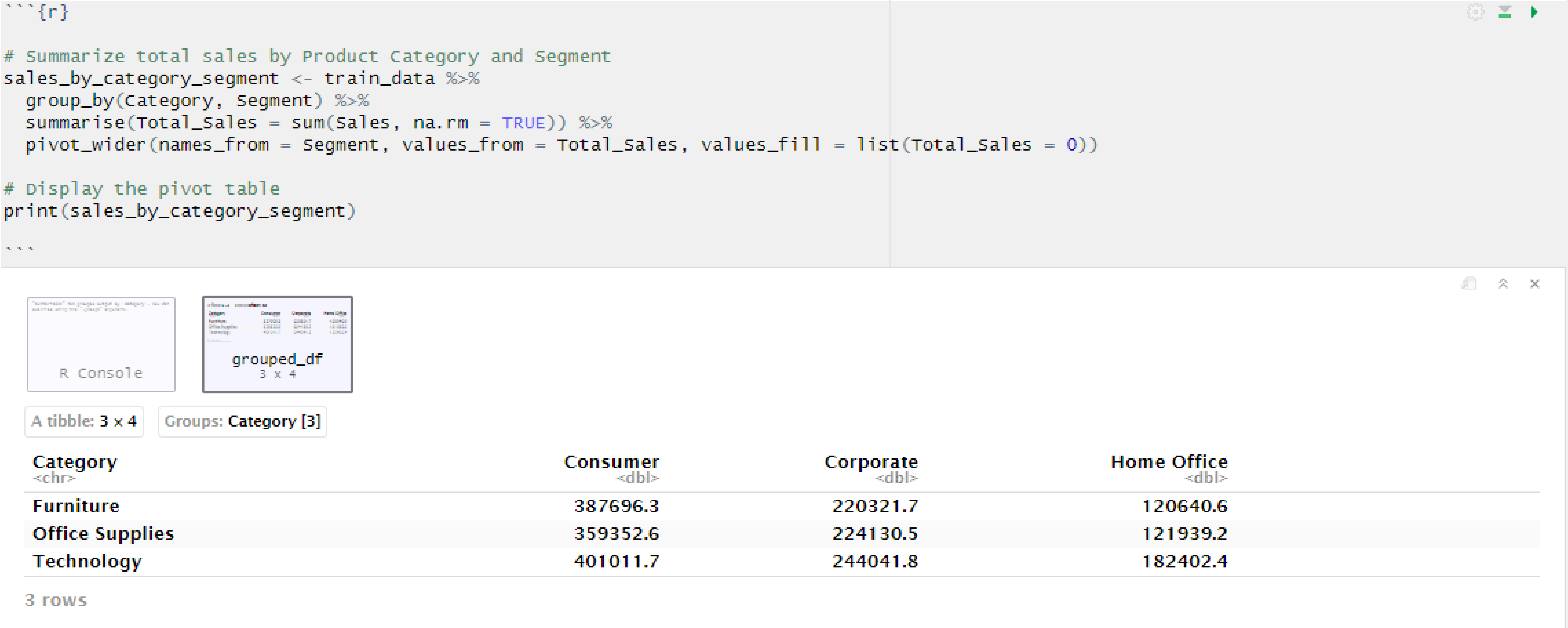


~~~
    direction = "down",
    showactive = TRUE,
    xanchor = "left", yanchor = "top",
    pad = list(t = 10)
   ),
   list(
    # Dropdown for selecting Month
    x = 0.3,
    y = 1.15,
    buttons = lapply(unique(sales_by_time$Month), function(month) {
     list(method = "relayout", args = list("xaxis.range", c(as.Date(paste0("2017-", month, "-01")), as.Date(paste0("2017-", month,
"-28")))),
       label = paste("Month", month))
    }),
    direction = "down",
    showactive = TRUE,
    xanchor = "left", yanchor = "top",
    pad = list(t = 30)
   )
  )
 )

# Display the line chart with Year and Month filters
line_chart

```
~~~


R Console
plotly
htmlwidget
Total Sales Over Time (Year/Month)
2015
2015
2016
2017
2018
Month 01
120k
100k
80k
60k
40k
20k
0
Total Sales
Jan 2015
Jul 2015
Jan 2016
Jul 2016
Jan 2017
Jul 2017
Jan 2018
Jul 2018
Jan 2019
Date

HULT
INTERNATIONAL
BUSINESS SCHOOL

Data Manipulation and Reshaping

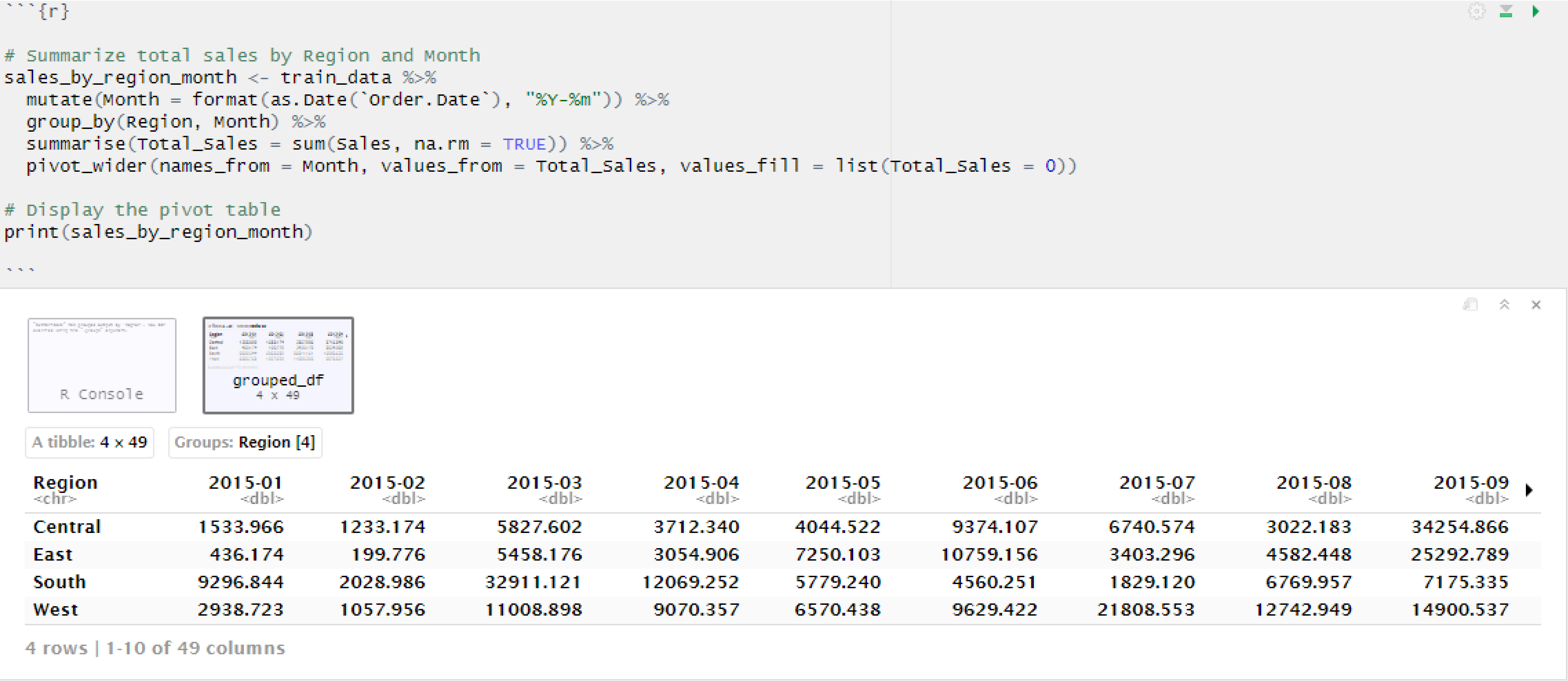


~~~
```{r}

# Summarize total sales by Product Category and Segment
sales_by_category_segment <- train_data %>%
  group_by(Category, Segment) %>%
  summarise(Total_Sales = sum(Sales, na.rm = TRUE)) %>%
  pivot_wider(names_from = Segment, values_from = Total_Sales, values_fill = list(Total_Sales = 0))

# Display the pivot table
print(sales_by_category_segment)

```
~~~

A tibble: 3 x 4 Groups: Category [3]

Category <chr>	Consumer <dbl>	Corporate <dbl>	Home Office <dbl>
Furniture	387696.3	220321.7	120640.6
Office Supplies	359352.6	224130.5	121939.2
Technology	401011.7	244041.8	182402.4

3 rows



~~~
```{r}

# Summarize total sales by Region and Month
sales_by_region_month <- train_data %>%
  mutate(Month = format(as.Date(`Order.Date`), "%Y-%m")) %>%
  group_by(Region, Month) %>%
  summarise(Total_Sales = sum(Sales, na.rm = TRUE)) %>%
  pivot_wider(names_from = Month, values_from = Total_Sales, values_fill = list(Total_Sales = 0))

# Display the pivot table
print(sales_by_region_month)

```
~~~

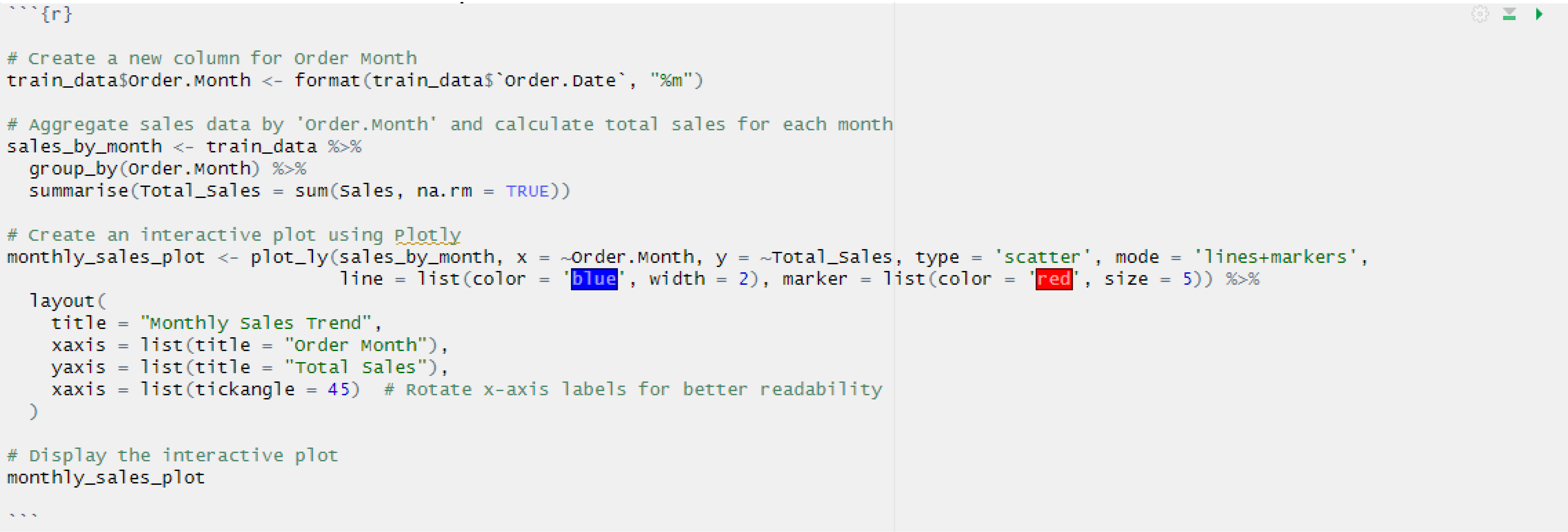
A tibble: 4 x 49 Groups: Region [4]

Region <chr>	2015-01 <dbl>	2015-02 <dbl>	2015-03 <dbl>	2015-04 <dbl>	2015-05 <dbl>	2015-06 <dbl>	2015-07 <dbl>	2015-08 <dbl>	2015-09 <dbl>
Central	1533.966	1233.174	5827.602	3712.340	4044.522	9374.107	6740.574	3022.183	34254.866
East	436.174	199.776	5458.176	3054.906	7250.103	10759.156	3403.296	4582.448	25292.789
South	9296.844	2028.986	32911.121	12069.252	5779.240	4560.251	1829.120	6769.957	7175.335
West	2938.723	1057.956	11008.898	9070.357	6570.438	9629.422	21808.553	12742.949	14900.537

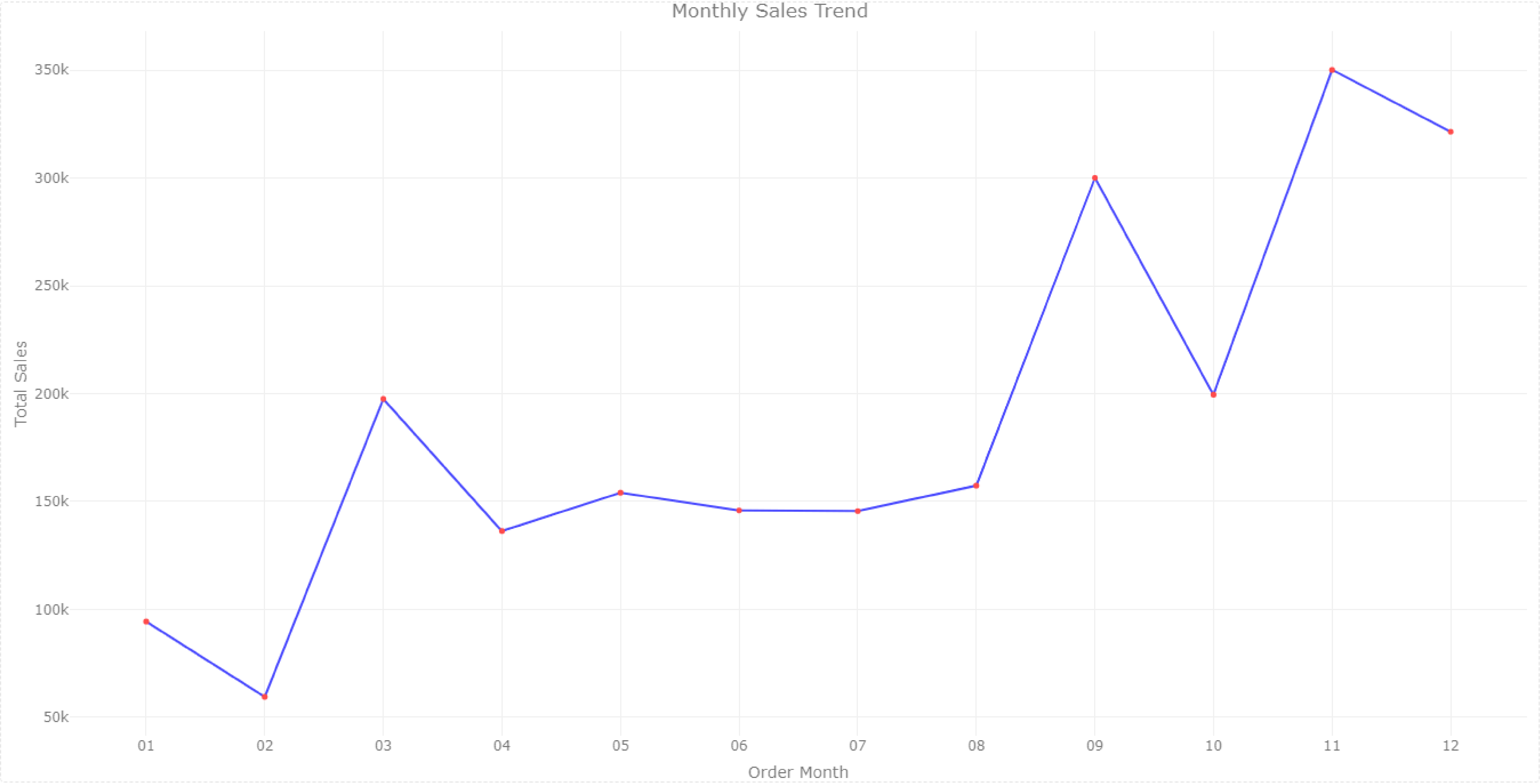
4 rows | 1-10 of 49 columns

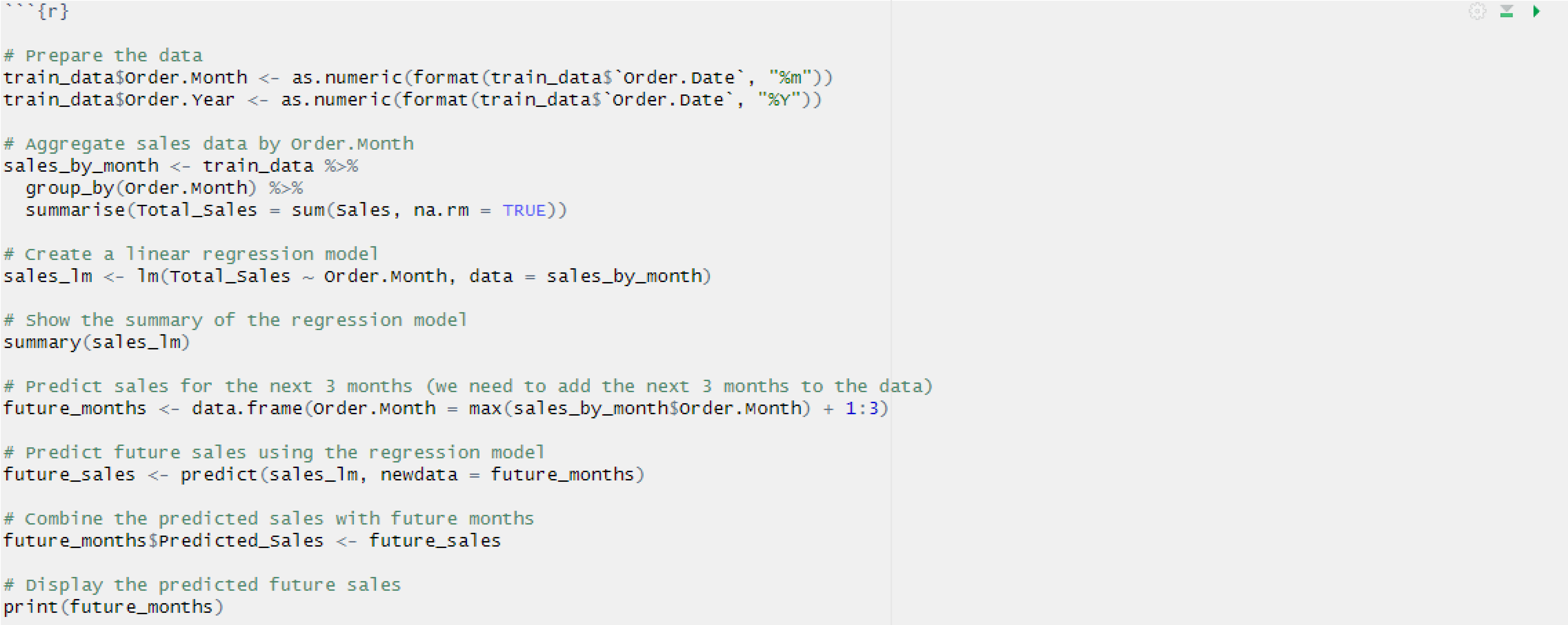
HULT
INTERNATIONAL
BUSINESS SCHOOL

Predicting Future Sales with Regression

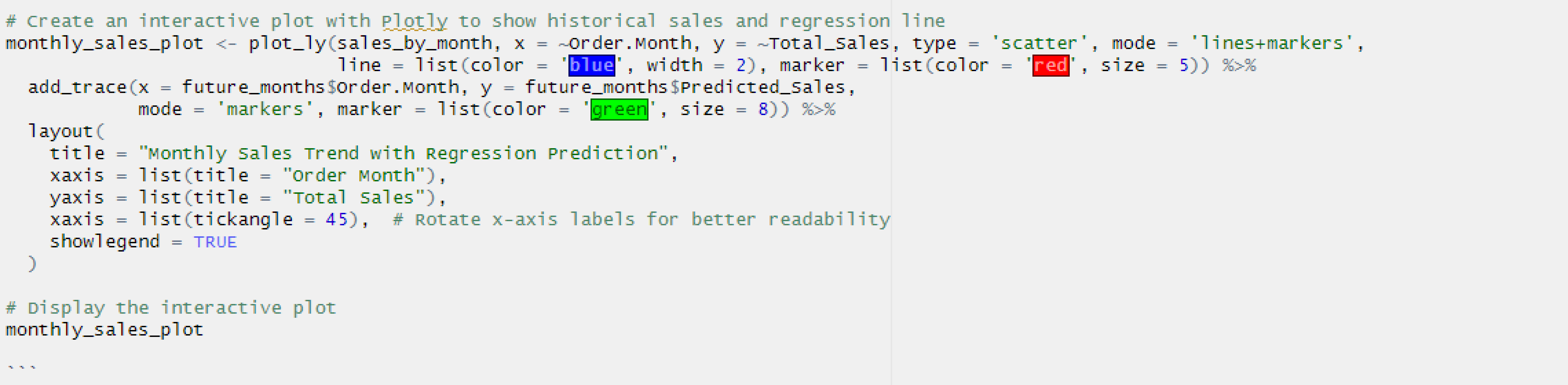


~~~
```{r}

# Create a new column for Order Month
train_data$Order.Month <- format(train_data$`Order.Date`, "%m")

# Aggregate sales data by 'Order.Month' and calculate total sales for each month
sales_by_month <- train_data %>%
  group_by(Order.Month) %>%
  summarise(Total_Sales = sum(Sales, na.rm = TRUE))

# Create an interactive plot using Plotly
monthly_sales_plot <- plot_ly(sales_by_month, x = ~Order.Month, y = ~Total_Sales, type = 'scatter', mode = 'lines+markers',
                              line = list(color = 'blue', width = 2), marker = list(color = 'red', size = 5)) %>%
  layout(
    title = "Monthly Sales Trend",
    xaxis = list(title = "Order Month"),
    yaxis = list(title = "Total Sales"),
    xaxis = list(tickangle = 45)  # Rotate x-axis labels for better readability
  )

# Display the interactive plot
monthly_sales_plot

```
~~~

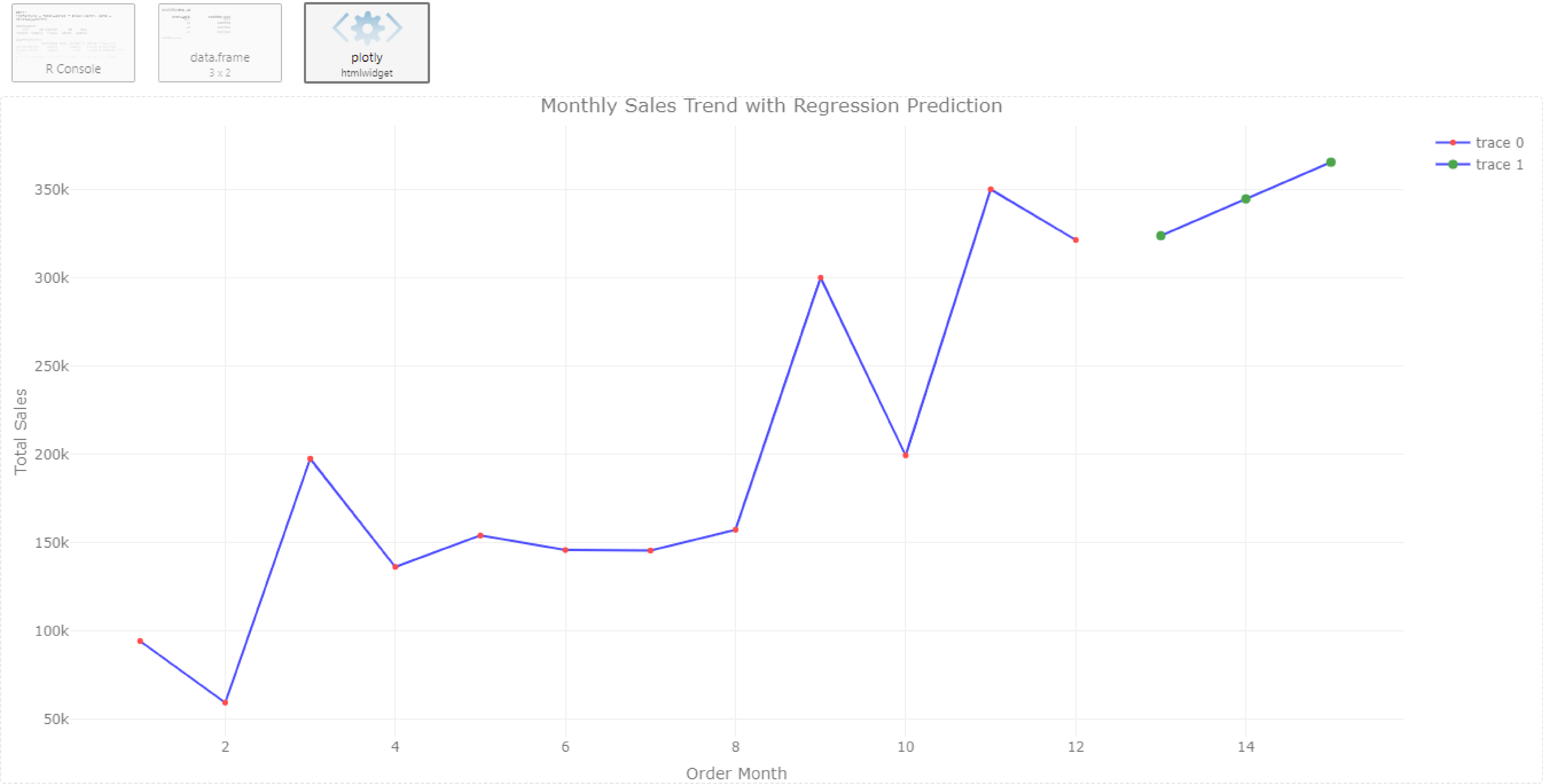

Monthly Sales Trend
Total Sales
350k
300k
250k
200k
150k
100k
50k
01
02
03
04
05
06
07
08
09
10
11
12
Order Month

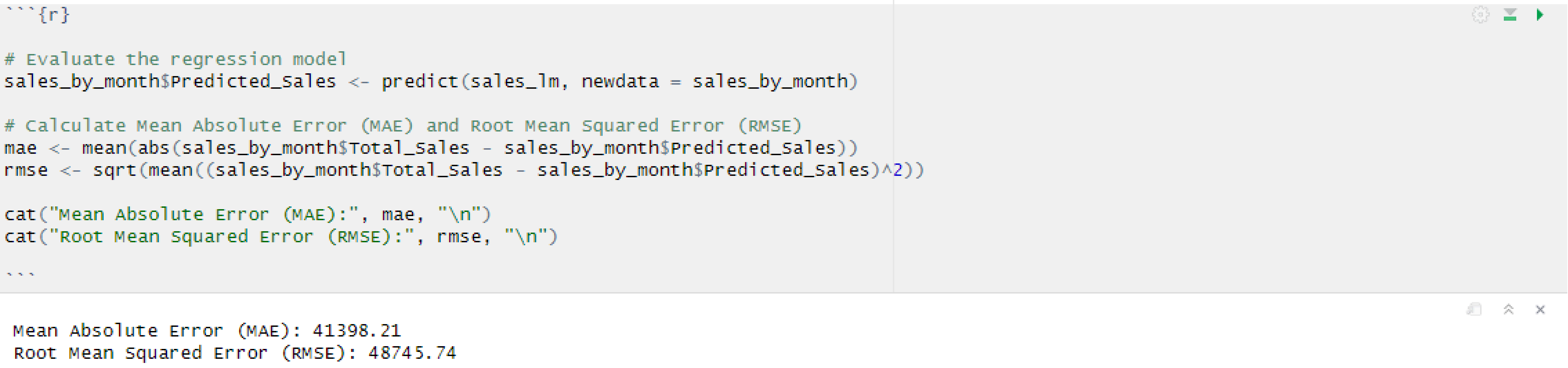


~~~
```{r}

# Prepare the data
train_data$Order.Month <- as.numeric(format(train_data$`Order.Date`, "%m"))
train_data$Order.Year <- as.numeric(format(train_data$`Order.Date`, "%Y"))

# Aggregate sales data by Order.Month
sales_by_month <- train_data %>%
  group_by(Order.Month) %>%
  summarise(Total_Sales = sum(Sales, na.rm = TRUE))

# Create a linear regression model
sales_lm <- lm(Total_Sales ~ Order.Month, data = sales_by_month)

# Show the summary of the regression model
summary(sales_lm)

# Predict sales for the next 3 months (we need to add the next 3 months to the data)
future_months <- data.frame(Order.Month = max(sales_by_month$Order.Month) + 1:3)

# Predict future sales using the regression model
future_sales <- predict(sales_lm, newdata = future_months)

# Combine the predicted sales with future months
future_months$Predicted_Sales <- future_sales

# Display the predicted future sales
print(future_months)
~~~



```
# Create an interactive plot with Plotly to show historical sales and regression line
monthly_sales_plot <- plot_ly(sales_by_month, x = ~Order.Month, y = ~Total_Sales, type = 'scatter', mode = 'lines+markers',
                              line = list(color = 'blue', width = 2), marker = list(color = 'red', size = 5)) %>%
  add_trace(x = future_months$Order.Month, y = future_months$Predicted_Sales,
            mode = 'markers', marker = list(color = 'green', size = 8)) %>%
  layout(
    title = "Monthly Sales Trend with Regression Prediction",
    xaxis = list(title = "Order Month"),
    yaxis = list(title = "Total Sales"),
    xaxis = list(tickangle = 45),  # Rotate x-axis labels for better readability
    showlegend = TRUE
  )

# Display the interactive plot
monthly_sales_plot

```
```
```


R Console
data.frame
3 x 2
plotly
htmlwidget
Monthly Sales Trend with Regression Prediction
trace 0
trace 1
350k
300k
250k
200k
150k
100k
50k
Total Sales
2
4
6
8
10
12
14
Order Month



~~~
```{r}

# Evaluate the regression model
sales_by_month$Predicted_Sales <- predict(sales_lm, newdata = sales_by_month)

# Calculate Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE)
mae <- mean(abs(sales_by_month$Total_Sales - sales_by_month$Predicted_Sales))
rmse <- sqrt(mean((sales_by_month$Total_Sales - sales_by_month$Predicted_Sales)^2))

cat("Mean Absolute Error (MAE):", mae, "\n")
cat("Root Mean Squared Error (RMSE):", rmse, "\n")

```
~~~

```
Mean Absolute Error (MAE): 41398.21
Root Mean Squared Error (RMSE): 48745.74
```

HULT
INTERNATIONAL
BUSINESS SCHOOL

Superstore Sales Dataset

Angel Lanto
MBAN 2024