**K.GOWTHAMI(192224229)**

**DSA0616-DATA HANDLING AND VISUALIZATION**

**DAY 02 PROGRAMS**

**11. Area Plot**

**Dataset: Yearly Revenue**

| **Year** | **Revenue (in million)** |
| --- | --- |
| 2015 | 30 |
| 2016 | 35 |
| 2017 | 40 |
| 2018 | 45 |
| 2019 | 50 |

# Create a data frame

revenue\_data <- data.frame(

Year = 2015:2019,

Revenue = c(30, 35, 40, 45, 50)

)

# Plot area plot

ggplot(revenue\_data, aes(x = Year, y = Revenue)) +

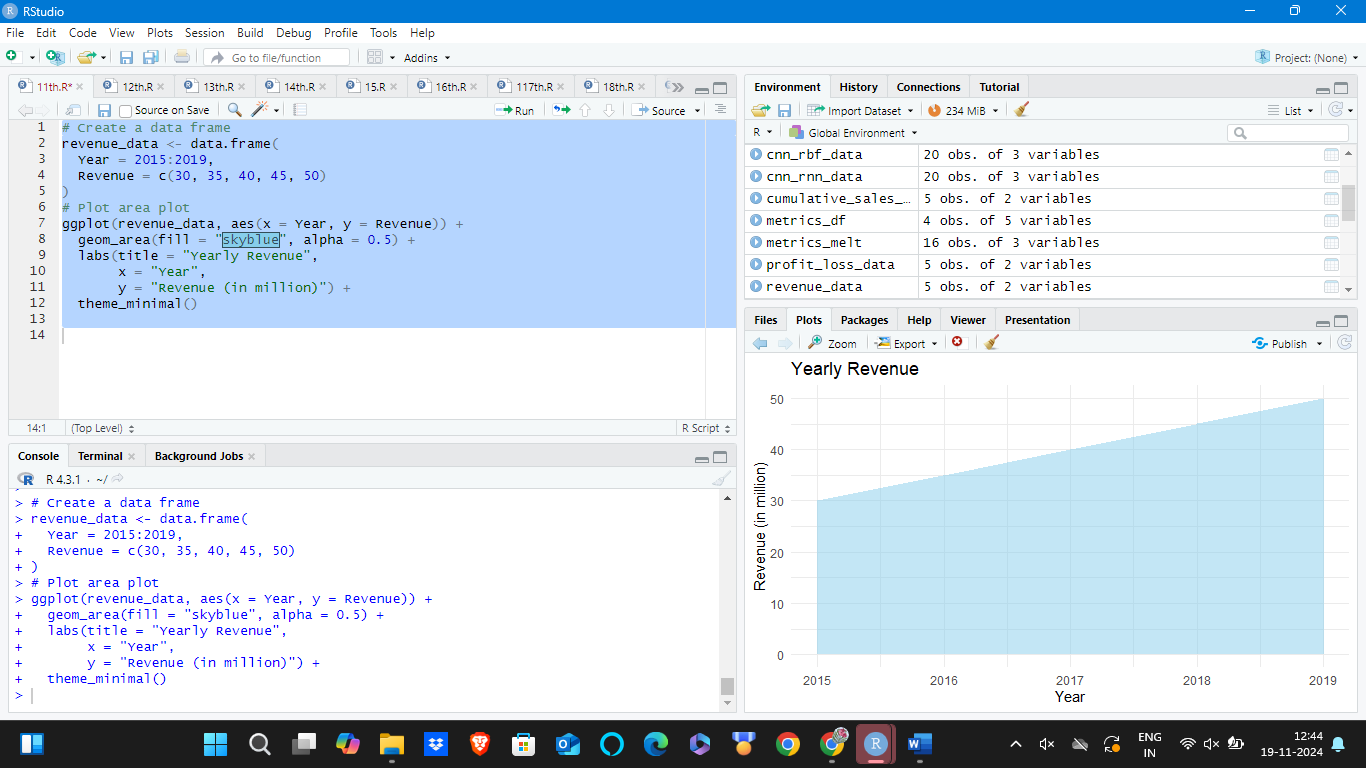
geom\_area(fill = "skyblue", alpha = 0.5) +

labs(title = "Yearly Revenue",

x = "Year",

y = "Revenue (in million)") +

theme\_minimal()



**12. Step Plot**

**Dataset: Cumulative Sales**

| **Month** | **Cumulative Sales** |
| --- | --- |
| Jan | 100 |
| Feb | 200 |
| Mar | 300 |
| Apr | 400 |
| May | 500 |

# Load necessary library

library(ggplot2)

# Create a data frame

cumulative\_sales\_data <- data.frame(

Month = factor(c("Jan", "Feb", "Mar", "Apr", "May"), levels = c("Jan", "Feb", "Mar", "Apr", "May")),

Cumulative\_Sales = c(100, 200, 300, 400, 500)

)

# Create the step plot

ggplot(cumulative\_sales\_data, aes(x = Month, y = Cumulative\_Sales, group = 1)) +

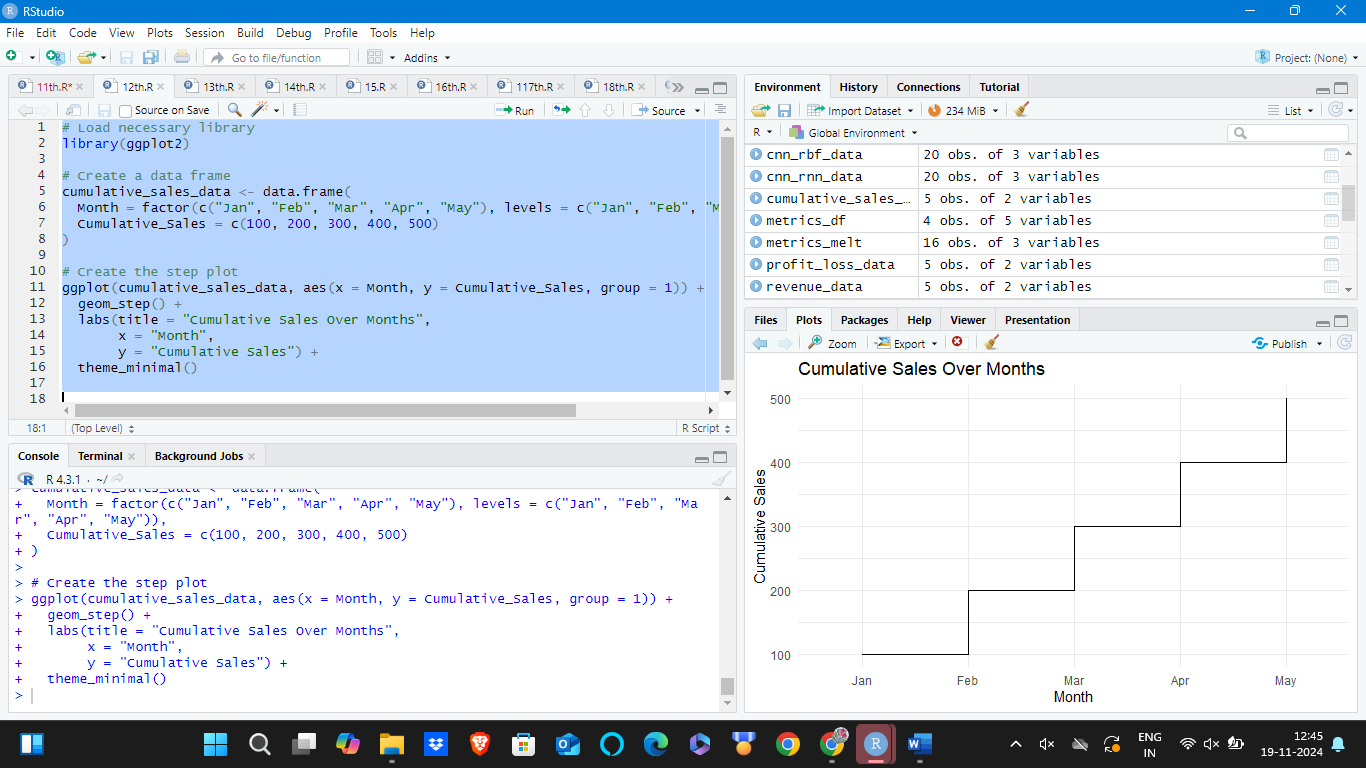
geom\_step() +

labs(title = "Cumulative Sales Over Months",

x = "Month",

y = "Cumulative Sales") +

theme\_minimal()

****

**13. Ridgeline Plot**

**Dataset: Temperature Distributions**

| **City** | **Temperature** |
| --- | --- |
| City1 | 20 |
| City1 | 21 |
| City1 | 19 |
| City2 | 22 |
| City2 | 23 |
| City2 | 24 |

# Load necessary library

library(ggridges)

# Create a data frame

temperature\_data <- data.frame(

City = c("City1", "City1", "City1", "City2", "City2", "City2"),

Temperature = c(20, 21, 19, 22, 23, 24)

)

# Plot ridgeline plot

ggplot(temperature\_data, aes(x = Temperature, y = City, fill = City)) +

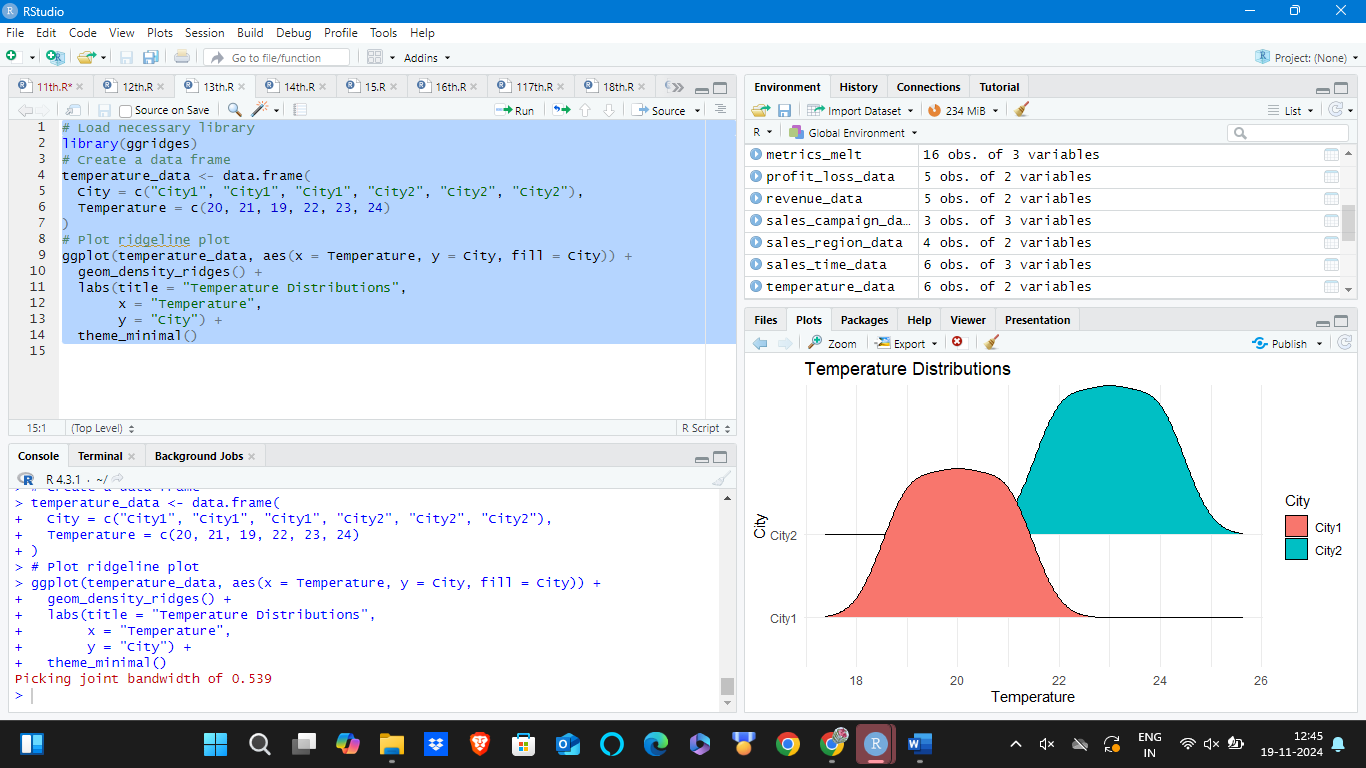
geom\_density\_ridges() +

labs(title = "Temperature Distributions",

x = "Temperature",

y = "City") +

theme\_minimal()

****

**14. Dumbbell Plot**

**Dataset: Sales Before and After Campaign**

| **Product** | **Before Campaign** | **After Campaign** |
| --- | --- | --- |
| A | 200 | 250 |
| B | 300 | 350 |
| C | 400 | 450 |
| D | 500 | 550 |
| E | 600 | 650 |

# Load necessary library

library(ggalt)

# Create a data frame

sales\_campaign\_data <- data.frame(

Product = c("A", "B", "C"),

Before = c(200, 300, 400),

After = c(250, 350, 450)

)

# Plot dumbbell plot

ggplot(sales\_campaign\_data) +

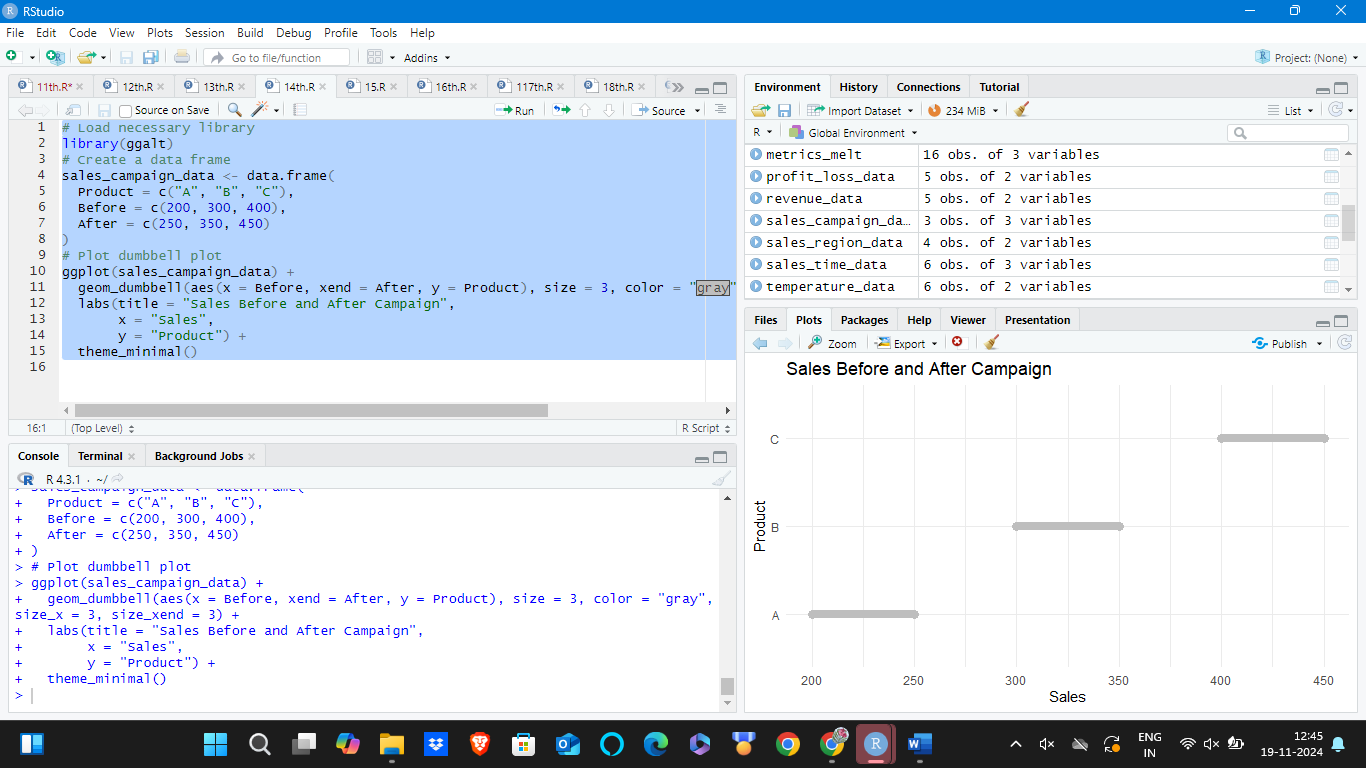
geom\_dumbbell(aes(x = Before, xend = After, y = Product), size = 3, color = "gray", size\_x = 3, size\_xend = 3) +

labs(title = "Sales Before and After Campaign",

x = "Sales",

y = "Product") +

theme\_minimal()



**15. Lollipop Plot**

**Dataset: Sales by Region**

| **Region** | **Sales** |
| --- | --- |
| North | 150 |
| South | 200 |
| East | 180 |
| West | 210 |

# Create a data frame

sales\_region\_data <- data.frame(

Region = c("North", "South", "East", "West"),

Sales = c(150, 200, 180, 210)

)

# Plot lollipop plot

ggplot(sales\_region\_data, aes(x = Region, y = Sales)) +

geom\_segment(aes(x = Region, xend = Region, y = 0, yend = Sales), color = "gray") +

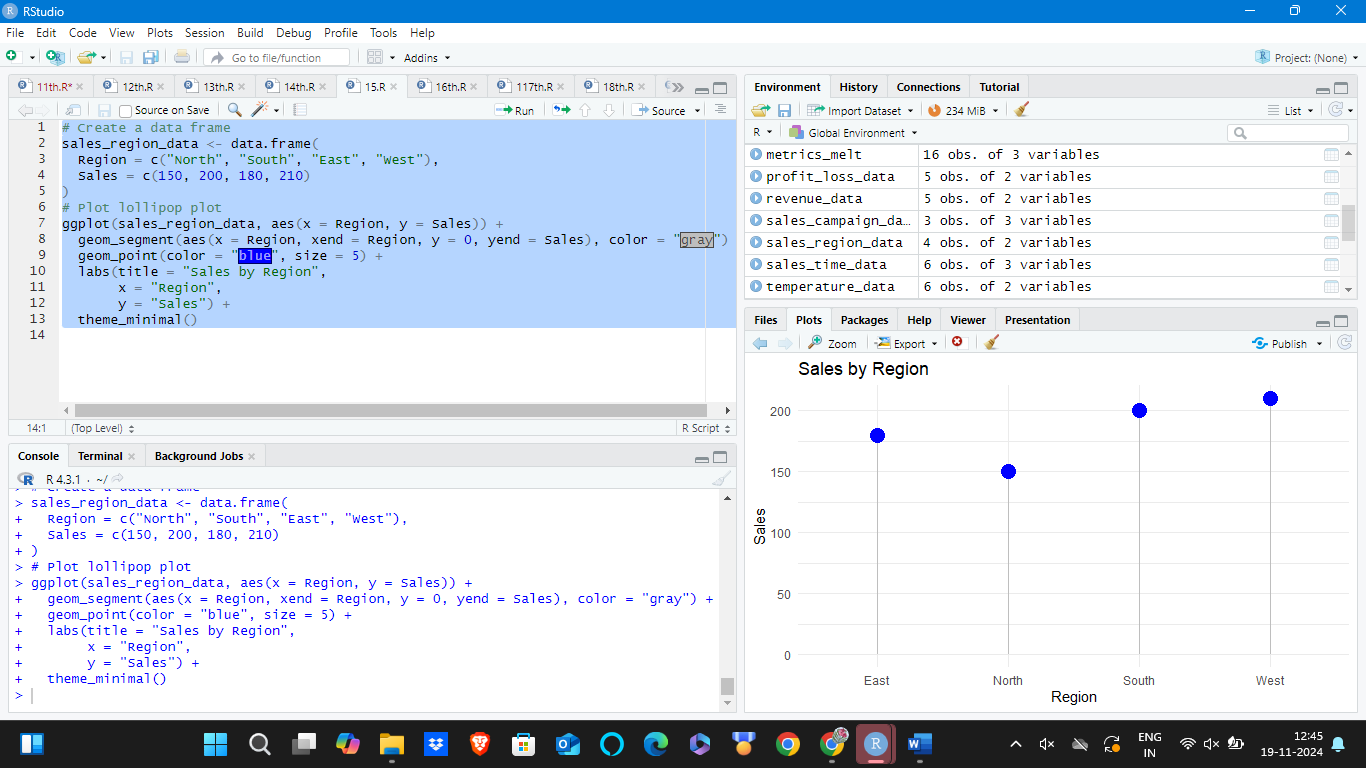
geom\_point(color = "blue", size = 5) +

labs(title = "Sales by Region",

x = "Region",

y = "Sales") +

theme\_minimal()

****

**16. Spaghetti Plot**

**Dataset: Sales over Time by Product**

| **Month** | **Product** | **Sales** |
| --- | --- | --- |
| Jan | A | 100 |
| Feb | A | 150 |
| Mar | A | 130 |
| Jan | B | 120 |
| Feb | B | 140 |
| Mar | B | 160 |

# Create a data frame

sales\_time\_data <- data.frame(

Month = c("Jan", "Feb", "Mar", "Jan", "Feb", "Mar"),

Product = c("A", "A", "A", "B", "B", "B"),

Sales = c(100, 150, 130, 120, 140, 160)

)

# Plot spaghetti plot

ggplot(sales\_time\_data, aes(x = Month, y = Sales, group = Product, color = Product)) +

geom\_line(size = 1) +

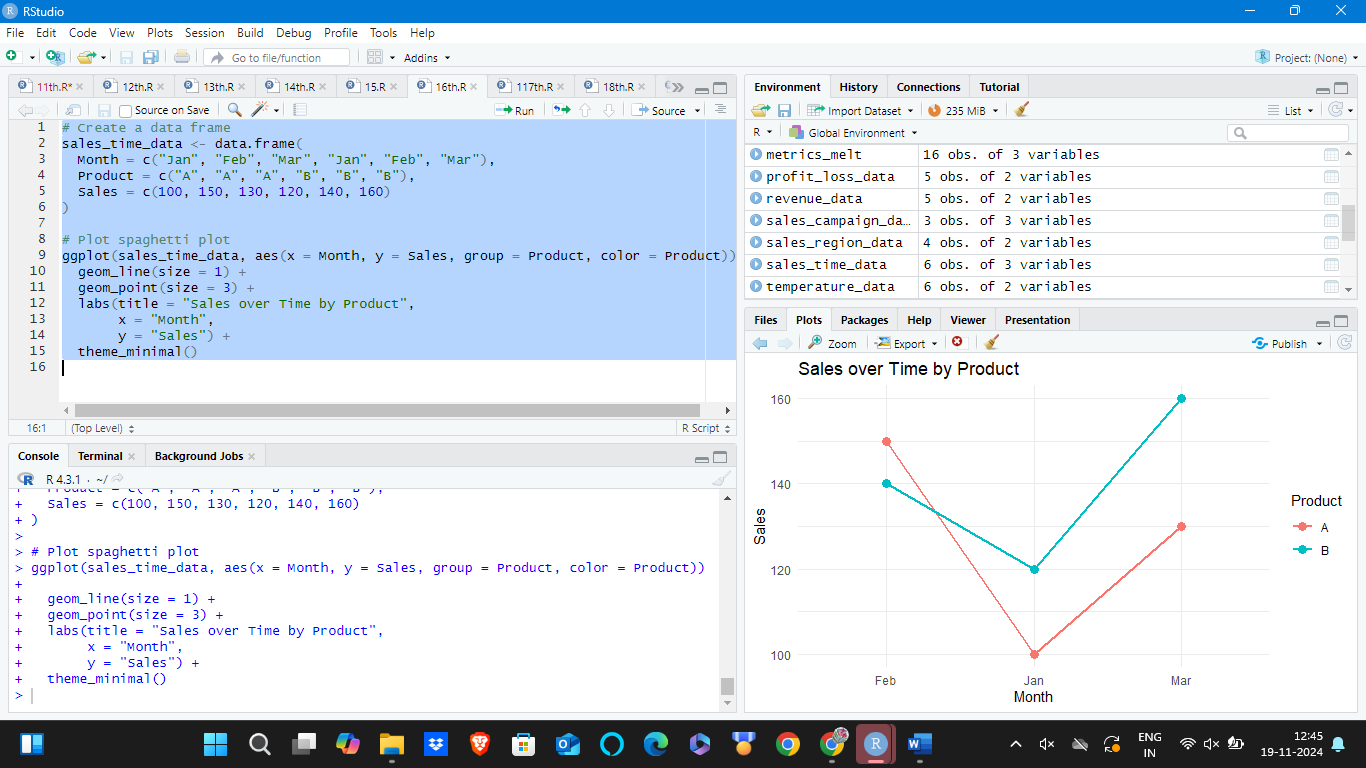
geom\_point(size = 3) +

labs(title = "Sales over Time by Product",

x = "Month",

y = "Sales") +

theme\_minimal()



**17. Waterfall Plot**

**Dataset: Profit and Loss by Month**

| **Month** | **Amount** |
| --- | --- |
| Jan | 100 |
| Feb | -20 |
| Mar | 50 |
| Apr | -10 |
| May | 80 |

# Load necessary library

library(waterfalls)

# Create a data frame with the appropriate structure

profit\_loss\_data <- data.frame(

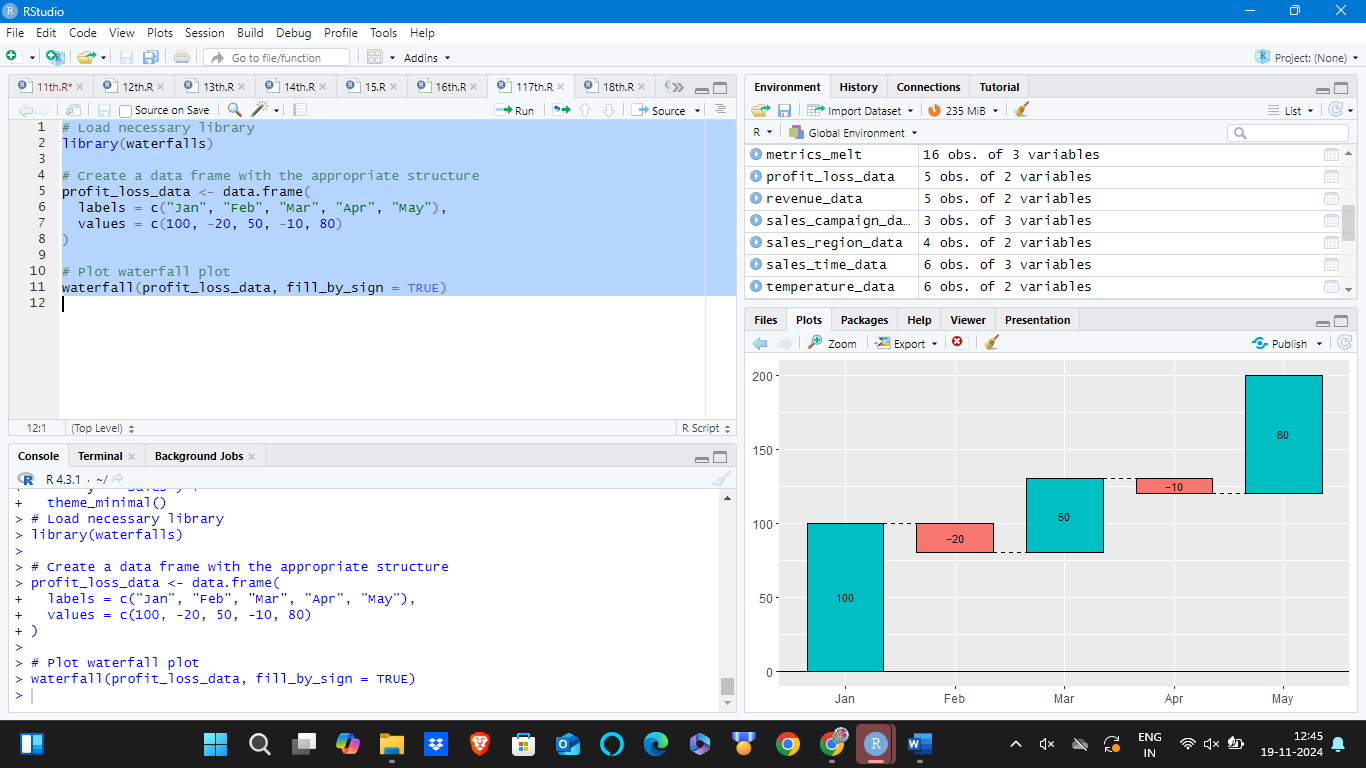
labels = c("Jan", "Feb", "Mar", "Apr", "May"),

values = c(100, -20, 50, -10, 80)

)

# Plot waterfall plot

waterfall(profit\_loss\_data, fill\_by\_sign = TRUE)



**18. Hexbin Plot**

**Dataset: X and Y Coordinates**

| **X** | **Y** |
| --- | --- |
| 1 | 2 |
| 2 | 3 |
| 3 | 4 |
| 4 | 5 |
| 5 | 6 |
| 6 | 7 |
| 7 | 8 |
| 8 | 9 |

# Load necessary library

library(hexbin)

# Create a data frame

xy\_data <- data.frame(

X = c(1, 2, 3, 4, 5, 6, 7, 8),

Y = c(2, 3, 4, 5, 6, 7, 8, 9)

)

# Plot hexbin plot

ggplot(xy\_data, aes(x = X, y = Y)) +

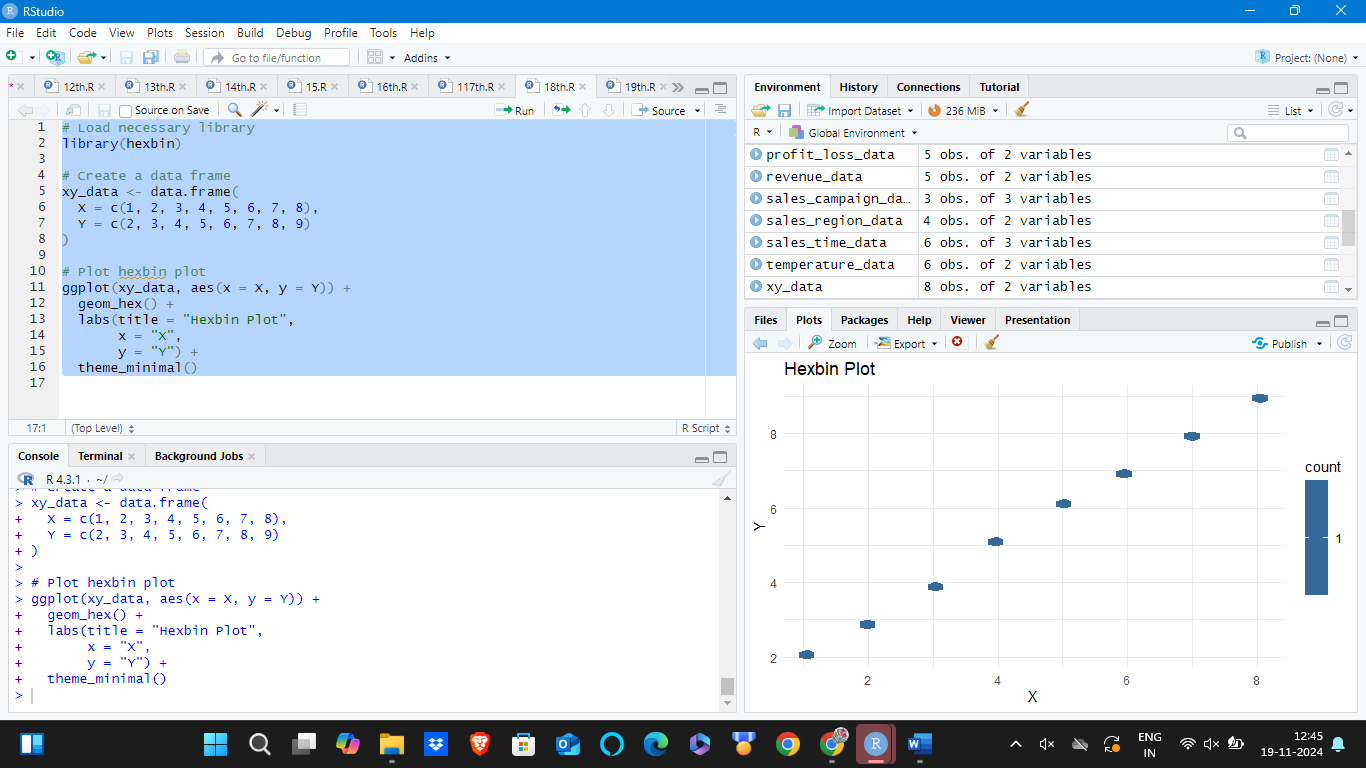
geom\_hex() +

labs(title = "Hexbin Plot",

x = "X",

y = "Y") +

theme\_minimal()



**19. Chord Diagram**

**Dataset: Connections between Categories**

| **From** | **To** | **Value** |
| --- | --- | --- |
| A | B | 10 |
| A | C | 20 |
| B | C | 15 |
| B | D | 25 |
| C | D | 30 |

# Load necessary library

library(circlize)

# Create a data frame

chord\_data <- data.frame(

From = c("A", "A", "B", "B", "C"),

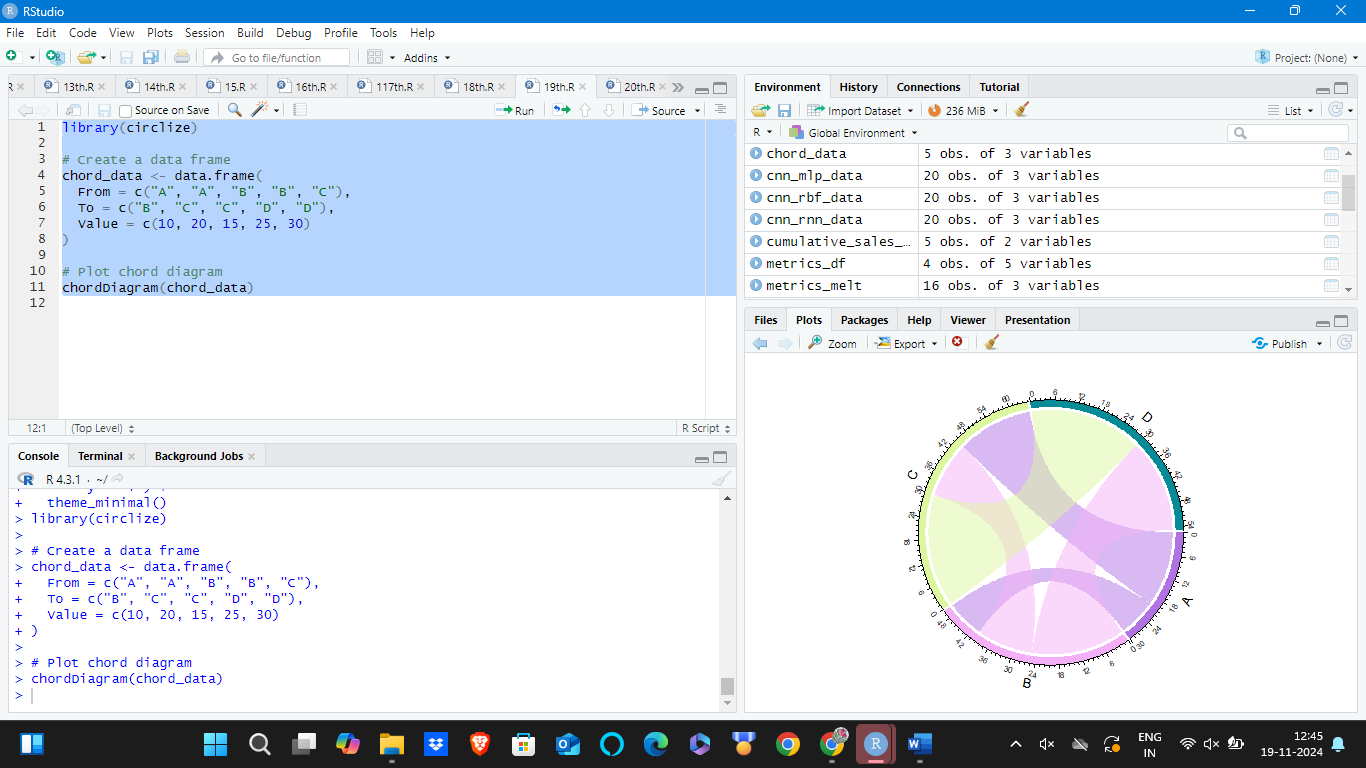
To = c("B", "C", "C", "D", "D"),

Value = c(10, 20, 15, 25, 30)

)

# Plot chord diagram

chordDiagram(chord\_data)



**20. Calendar Heatmap**

**Dataset: Daily Activity Count**

| **Date** | **Count** |
| --- | --- |
| 2023-01-01 | 10 |
| 2023-01-02 | 12 |
| 2023-01-03 | 15 |
| 2023-01-04 | 8 |
| 2023-01-05 | 20 |

# Load necessary library

library(lubridate)

library(ggplot2)

# Create a data frame

activity\_data <- data.frame(

Date = as.Date(c("2023-01-01", "2023-01-02", "2023-01-03", "2023-01-04", "2023-01-05")),

Count = c(10, 12, 15, 8, 20)

)

# Add weekday and week of year columns

activity\_data$Weekday <- wday(activity\_data$Date, label = TRUE)

activity\_data$Week <- week(activity\_data$Date)

# Plot calendar heatmap

ggplot(activity\_data, aes(x = Weekday, y = Week, fill = Count)) +

geom\_tile(color = "white") +

scale\_fill\_gradient(low = "white", high = "blue") +

labs(title = "Daily Activity Count",

x = "Weekday",

y = "Week",

fill = "Count") +

theme\_minimal()

