```r
library(readxl)

report_data <- read_excel("C:/Users/Abrar Labib/Desktop/Temperature dataset.xlsx")



#1. DATA ACQUISITION AND PREPARATION

#Data Preprocessing and Cleaning

#Exploring the dataset


#1.#Checking the dimension of the dataset (Row x Column)

dim(report_data)

ncol(report_data)

nrow(report_data)



#2.

#Understanding the data structure

glimpse(report_data)                    #Shows a short structure of our dataset

str(report_data)                        #Shows the characterstics of the data present



#3.

#Data inspection

head(report_data,n=20)                  #Shows first few (20) rows of the dataset

tail(report_data,n=20)                  #Seeing the last 20 rows of the dataset
```

```
#4.

#Sampling

sample_n(report_data,15)                    #Seeing random 15 rows of the dataset to inspect existence of any unexpexted change in dataset structure
```

```
#5.

#Duplicacy checking

duplicated(report_data)

sum(duplicated(report_data))                #No duplicated values came as output. Even if there were any, we wouldn't be much bothered about that. Because having same temperature at multiple days is possible.
```

```
#6.

#Formatting dates

library(lubridate)

library(dplyr)

df <- report_data |>

  mutate(Date = make_date(YEAR, MO, DY))      # Creating a separate column for Date (YYYY-MM-DD)

print(df)
```

```
#7.

#Deleting the regular YEAR, Month, Day variable/columns

df1 <- df %>%
```

```
  select(-YEAR, -MO, -DY)
print(df1)
```

```
#8.
#Organizing the variables/columns
df1 <- df1[, c(ncol(df1), 1:(ncol(df1)-1))]
print(df1)
```

```
#9.
#Renaming the T2M_MAX variable/column to Max Temperature
colnames(df1)[colnames(df1) == "T2M_MAX"] <- "Max Temperature"
```

```
#10.
#Checking missing values in the dataset
is.na(df1)                    #Outcome is FALSE
sum(is.na(df1))                  #Outcome is 0. Means there aren't any missing values present in the dataset.
```

```
#11.
#Checking extreme outliers (Assuming temperature > 46 degrees Celcius not possible)
library(readxl)
any(df1$"Max Temperature" > 46, na.rm = TRUE)     #Outcome's FALSE. Means there aren't any such values.
```

```
#12.
#Checking out presence of any infinite values
is.infinite("df1")                         #Output's FALSE.
sum(is.infinite("df1"))
```

```
#2. DESCRIPTIVE STASTICAL ANALYSIS
#Central Tendency determining
#MEAN
mean_value <- mean(df1$`Max Temperature`, na.rm = TRUE)
cat("Mean:", mean_value, "\n")                         # Mean= 30.38461

#MEDIAN
median_value <- median(df1$`Max Temperature`, na.rm = TRUE)
cat("Median:", median_value, "\n")                     # Median = 30.92

#MODE
get_mode <- function(x) {
  uniq_x <- unique(x)
  uniq_x[which.max(tabulate(match(x, uniq_x)))]
}

mode_value <- get_mode(df1$`Max Temperature`)
cat("Mode:", mode_value, "\n")                         # Mode= 30.56


#Inter-Quartile Range
```

```
iqr<- IQR(df1$`Max Temperature`, na.rm = TRUE)


# Print the result
cat("Interquartile Range:", iqr_value, "\n")            #IQR= 6.54









# Calculating the skewness and kurtosis values of Max Temperature variable/column
library(e1071)
skewness_value <- skewness(df1$`Max Temperature`, na.rm = TRUE)
kurtosis_value <- kurtosis(df1$`Max Temperature`, na.rm = TRUE)



# Ensure Date column is in Date format
df1$Date <- as.Date(df1$Date)


# Extract months and store in a separate variable
months_extracted <- format(df1$Date, "%m")


# Print extracted months
print(months_extracted)







#Viewing the skewness of the dataset
library(e1071)
```

```r
library(ggplot2)

skewness_value <- skewness(df1$`Max Temperature`, na.rm = TRUE)

kurtosis_value <- kurtosis(df1$`Max Temperature`, na.rm = TRUE)


# Creating  a histogram having a  density curve

ggplot(df1, aes(x = `Max Temperature`)) +

  geom_histogram(aes(y = ..density..), bins = 30, fill = "lightgreen", color = "black", alpha = 0.7) +

  geom_density(color = "red", size = 1.2) +

  ggtitle(paste("Skewness:", round(skewness_value, 2),

                " | Kurtosis:", round(kurtosis_value, 2))) +

  theme_minimal() +

  labs(x = "Max Temperature", y = "Density")

ggsave(filename = "C:/Users/Abrar Labib/Desktop/Skewness_Kurtosis_Plot.png", plot = plot, width = 8, height = 6, dpi = 300, bg="white")
```

```r
#Calculating Percentiles and Quartiles

percentiles <- quantile(df1$`Max Temperature`, probs = c(0.10, 0.50, 0.90), na.rm = TRUE)

quartiles <- quantile(df1$`Max Temperature`, probs = c(0.25, 0.5, 0.75), na.rm = TRUE)
```

```
#MONTHLY MAX TEMPERATURE AVERAGE

#Extracting each months

Mar_2019 <- df1[1:31, ]

print(Mar_2019)



Apr_2019 <- df1[32:61, ]

print(Apr_2019)



May_2019 <- df1[62:92, ]

print(May_2019)



Jun_2019 <- df1[93:122, ]

print(Jun_2019)



Jul_2019 <- df1[123:153, ]

print(Jul_2019)



Aug_2019 <- df1[154:184, ]

print(Aug_2019)



Sep_2019 <- df1[185:214, ]

print(Sep_2019)
```

```
Oct_2019 <- df1[215:245, ]

print(Oct_2019)

Nov_2019 <- df1[246:275, ]

print(Nov_2019)

Dec_2019 <- df1[276:306, ]

print(Dec_2019)

Jan_2020 <- df1[307:337, ]

print(Jan_2020)

Feb_2020 <- df1[338:366, ]

print(Feb_2020)

Mar_2020 <- df1[367:397, ]

print(Mar_2020)

Apr_2020 <- df1[398:427, ]
```

```r
print(Apr_2020)

May_2020 <- df1[428:458, ]
print(May_2020)

Jun_2020 <- df1[459:488, ]
print(Jun_2020)

Jul_2020 <- df1[489:519, ]
print(Jul_2020)

Aug_2020 <- df1[520:550, ]
print(Aug_2020)

Sep_2020 <- df1[551:580, ]
print(Sep_2020)

Oct_2020 <- df1[581:611, ]
print(Oct_2020)

Nov_2020 <- df1[612:641, ]
print(Nov_2020)
```

```r
Dec_2020 <- df1[642:672, ]
print(Dec_2020)

Jan_2021 <- df1[673:703, ]
print(Jan_2021)

Feb_2021 <- df1[704:731, ]
print(Feb_2021)

Mar_2021 <- df1[732:762, ]
print(Mar_2021)

Apr_2021 <- df1[763:792, ]
print(Apr_2021)

May_2021 <- df1[793:823, ]
print(May_2021)

Jun_2021 <- df1[824:853, ]
print(Jun_2021)
```

```
Jul_2021 <- df1[854:884, ]

print(Jul_2021)

Aug_2021 <- df1[885:915, ]

print(Aug_2021)

Sep_2021 <- df1[916:945, ]

print(Sep_2021)

Oct_2021 <- df1[946:976, ]

print(Oct_2021)

Nov_2021 <- df1[977:1006, ]

print(Nov_2021)

Dec_2021 <- df1[1007:1037, ]

print(Dec_2021)

Jan_2022 <- df1[1038:1068, ]
```

```
print(Jan_2022)

Feb_2022 <- df1[1069:1096, ]
print(Feb_2022)

Mar_2022 <- df1[1097:1099, ]
print(Mar_2022)

#MONTHLY MAX TEMPERATURE AVERAGE
monthly_vars <- ls(pattern = "^[A-Z][a-z]{2}_\\d{4}$")
monthly_avg_temp <- c()
for (month in monthly_vars) {
  data <- get(month)
  # Calculating the average Max Temperature for each  months
  monthly_avg_temp[month] <- mean(data$'Max Temperature', na.rm = TRUE)
}
print(monthly_avg_temp)

#SEASONAL AVERAGE MAX TEMPERATURE
```

```
Summer_2019 <- df1[32:122, ]

print(Summer_2019)


Monsoon_2019 <- df1[107:184, ]

print(Monsoon_2019)


Autumn_2019 <- df1[154:245, ]

print(Autumn_2019)


Winter_2020 <- df1[246:366, ]

print(Winter_2019_2020)


Summer_2020 <- df1[398:488, ]

print(Summer_2020)


Monsoon_2020 <- df1[473:550, ]

print(Monsoon_2020)


Autumn_2020 <- df1[551:611, ]

print(Autumn_2020)


Winter_2021 <- df1[612:731, ]
```

```r
print(Winter_2020_2021)

Summer_2021 <- df1[763:853, ]
print(Summer_2021)

Monsoon_2021 <- df1[854:915, ]
print(Monsoon_2021)

Autumn_2021 <- df1[916:976, ]
print(Autumn_2021)

Winter_2022 <- df1[977:1096, ]
print(Winter_2022)

# Listing all seasonal variables that I've assigned
seasonal_vars <- ls(pattern = "^(Summer|Winter|Autumn|Spring|Monsoon|Rainy)_[0-9]{4}$")
seasonal_avg_temp <- data.frame(Season = character(), Avg_Temperature = numeric(), stringsAsFactors = FALSE)

# Looping through the each seasonal variable and calculating the average temperature for each seasons of the months
for (season in seasonal_vars) {
  # Get the dataset for the season
  data <- get(season)
```

```
  if ("Max Temperature" %in% colnames(data)) {
    valid_temps <- data$'Max Temperature'[!is.na(data$'Max Temperature')]  # Remove NA values

    if (length(valid_temps) > 0) {  # If there are valid temperatures
      avg_temp <- mean(valid_temps)
    } else {
      avg_temp <- NA  # If no valid temperatures, return NA
    }
  } else {
    avg_temp <- NA  # If 'Max Temperature' column is missing
  }
  seasonal_avg_temp <- rbind(seasonal_avg_temp, data.frame(Season = season, Avg_Temperature = avg_temp))
}
library(knitr)
kable(seasonal_avg_temp)
```

```
#Distribution
# Required packages are loaded
library(dplyr)
library(ggplot2)
mean_temp <- mean(df1$`Max Temperature`, na.rm = TRUE)
sd_temp <- sd(df1$`Max Temperature`, na.rm = TRUE)
threshold <- 40
```

```r
extreme_days <- df1 %>%
  filter(`Max Temperature` > threshold)

extreme_days_count <- nrow(extreme_days)
total_days <- nrow(df1)
extreme_percentage <- (extreme_days_count / total_days) * 100

# Printing the outcomes
cat("Mean Temperature: ", mean_temp, "\n")
cat("Standard Deviation: ", sd_temp, "\n")
cat("Threshold for extreme temperatures: ", threshold, "°C\n")
cat("Number of extreme days (above 40°C): ", extreme_days_count, "\n")
cat("Percentage of extreme days: ", round(extreme_percentage, 2), "%\n")

# A histogram to visualize the distribution of max temperatures
ggplot(df1, aes(x = `Max Temperature`)) +
  geom_histogram(binwidth = 1, fill = "#4C9F70", color = "black", alpha = 0.8) +
  geom_vline(xintercept = threshold, color = "#D84B16", linetype = "dashed", size = 1) +
  labs(
    title = paste("Distribution of Max Temperature\nExtreme Threshold: > 40°C"),
    x = "Max Temperature (°C)", y = "Frequency"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold", color = "#333333"),
    panel.grid.major = element_line(color = "#F0F0F0", size = 0.5), # Light grid lines
    panel.grid.minor = element_blank(),
    axis.text = element_text(size = 12, color = "#333333"), # Axis text styling
    axis.title = element_text(size = 14, face = "bold", color = "#333333")
  )
```

```r
#Exporting the generated distribution graph
file_path <- "C:/Users/Abrar Labib/Desktop/distribution.png"
ggsave(file_path,
       plot = last_plot(),
       width = 8, height = 6,
       dpi = 300)
```

```r
#Calculating days with extreme temperatures >40 degrees celcius
library(dplyr)
threshold <- 40
extreme_days <- df1 %>%
  filter(`Max Temperature` > threshold)

extreme_days_count <- nrow(extreme_days)
total_days <- nrow(df1)
extreme_percentage <- (extreme_days_count / total_days) * 100          # This is the percentage of extreme days

extreme_stats <- extreme_days %>%
  summarise(
    mean_temp = mean(`Max Temperature`, na.rm = TRUE),
    median_temp = median(`Max Temperature`, na.rm = TRUE),
    sd_temp = sd(`Max Temperature`, na.rm = TRUE)
  )

# Combining all results into a single tablular format for easiness
```

```
result_table <- data.frame(
  Total_Days = total_days,
  Extreme_Days_Count = extreme_days_count,
  Extreme_Days_Percentage = round(extreme_percentage, 2),
  Mean_Extreme_Temperature = round(extreme_stats$mean_temp, 2),
  Median_Extreme_Temperature = round(extreme_stats$median_temp, 2),
  SD_Extreme_Temperature = round(extreme_stats$sd_temp, 2)
)
print(result_table)
```

```
#HEATMAP
library(ggplot2)
library(readxl)
library(dplyr)
library(lubridate)

df2 <- read_excel("C:/Users/Abrar Labib/Desktop/df2.xlsx")

# Filtering the  high-risk days (>37°C)
```

```
high_risk_df <- df2 %>% filter(Max_Temperature > 37)


# Extracting the Month and Year from my dataset

high_risk_df$Month <- factor(month(high_risk_df$Date, label = TRUE))  # Convert month to factor

high_risk_df$Year <- factor(year(high_risk_df$Date))  # Convert year to factor


# Heatmap creation of high-risk days. I've assumed temperature > 37 as high enough temperature

heatmap_plot <- ggplot(high_risk_df, aes(x = Month, y = Year, fill = Max_Temperature)) +
  geom_tile(color = "white") +  # White grid lines
  scale_fill_gradient(low = "yellow", high = "red", name = "Max Temp (°C)") +
  labs(
    title = "High-Risk Temperature Days (>37°C) by Month & Year",
    x = "Month",
    y = "Year"
  ) +
  theme_minimal() +
  theme(
    panel.background = element_rect(fill = "white", color = "white"),
    plot.background = element_rect(fill = "white", color = "white"),
    plot.title = element_text(hjust = 0.5, face = "bold"),
    legend.title = element_text(face = "bold"),
    axis.title.x = element_text(face = "bold"),
    axis.title.y = element_text(face = "bold")
  )


print(heatmap_plot)



# Saving th heatmap

ggsave("C:/Users/Abrar Labib/Desktop/heatmap.png", plot = heatmap_plot, width = 8, height = 6, dpi = 300, bg = "white")
```

```
# Overlay the map with geospatial layers (e.g., administrative boundaries) to provide contextual insights.

library(sf)
library(ggplot2)
library(ggspatial)

# Locating the  file paths to R
temperature_map_path <- "E:/GIS data for Report/Rajshahi/rajdiv.shp"
admin_boundary_path <- "E:/GIS data for Report/BGD_adm2.shp"

# Reading both  the shapefiles
temperature_map <- st_read(temperature_map_path)
admin_boundary <- st_read(admin_boundary_path)

# Plotting the maps together using ggplot2 package
ggplot() +
  geom_sf(data = temperature_map, aes(fill = Temperature), alpha = 0.7) +
  geom_sf(data = admin_boundary, color = "black", fill = NA, size = 0.8) +
  scale_fill_viridis_c(option = "magma", name = "Temperature (°C)") +
  annotation_north_arrow(location = "tl", which_north = "true") +
  annotation_scale(location = "br") + # Add scale bar
  labs(title = "Temperature Map Overlaid with Administrative Boundaries",
       caption = "Data Source: Your Dataset") +
  theme_minimal()
```

Highlighting regions with the highest temperature extremes in the region of interest

#I chose to work on Rajshahi division. Because Rajshahi is a prime location in case of heat related issues throughout the country. Every year, Rajshahi faces severe exposure leading to heat sensitive cases.

#Latitude and Longitude for Rajshahi = 24.3631° N and 88.6073° E respectively. Temperature data were downloaded from the NASA POWER website and were imported to R later on.

#Arcmap 10.8 version (a Geographical Information System software) was used to extract the whole shape file of Rajshahi division from BDG_adm file for the whole country.

```
#Map Visualization

#Adding latitude and longitude  to df1 dataset

# Creating a new dataset df2 by adding "LON" and "LAT" columns in df1 dataset


df2 <- df1 %>%

  mutate(LON = 88.6073,  # Adding longitude

        LAT = 24.3631)  # Adding latitude

head(df2)



# Mapping the high risk zones in the map

library(sf)

library(dplyr)

library(tmap)

library(readxl)

shapefile <- st_read("E:/GIS data for Report/Rajshahi/rajdiv.shp")


# Ensuring that the  the shapefile uses the correct CRS

if (st_crs(shapefile)$epsg != 4326) {

  shapefile <- st_transform(shapefile, crs = 4326)

}
```

```r
# Temperature dataset df2 loadning
df2 <- read_excel("C:/Users/Abrar Labib/Desktop/df2.xlsx")


# MAking sure that the  Max Temperature is numeric and filter for high-temperature points (e.g., >= 37°C)
df2$`Max Temperature` <- as.numeric(df2$Max_Temperature)
df2 <- df2 %>% filter(`Max_Temperature` >= 37)



# Converting temperature data to spatial data(LON first, then LAT)
ext_temps_sf <- st_as_sf(df2, coords = c("LON", "LAT"), crs = 4326)


#  Making sure that all the points are within the desired (Rajshahi) region using `st_intersection()`
ext_temps_sf <- st_intersection(ext_temps_sf, shapefile)


# Viewing the mapp
tmap_mode("view")


tm_shape(shapefile) +
  tm_polygons(col = "lightgray", border.col = "black", lwd = 0.5) +
  tm_shape(ext_temps_sf) +
  tm_dots(col = "Max_Temperature",
        style = "quantile",
        palette = "YlOrRd",                      # Yellow color >  Red color palette
        size = 0.5,
        title = "Temperature Extremes") +
  tm_layout(main.title = "Extreme Temperature Zones in Rajshahi Division",
          legend.outside = TRUE)
```

```
# Construct the file path for saving to Desktop

file_path <- file.path("C:/Users/Abrar Labib/Desktop", "high risk.png")


# Save the map to the Desktop

tmap_save(map, file_path)
```

#All of the temperatue events are on the same spot because of using the same LAT and LON. But Using different LAT and LON for all of the Rajshahi districts create dispersion in the shape file that doesn't resemble with the crs of all the points. Which shows the points situate outside of the region. That's why, despite of being imperfect visualization, only single point LAT and LON has been kept final.

#Temporal Analysis:

#Time Series

```
library(ggplot2)

library(readxl)

df2 <- read_excel("C:/Users/Abrar Labib/Desktop/df2.xlsx")


# Assuring the "Date" column is in Date format.

df2$Date <- as.Date(df2$Date, format = "%Y-%m-%d")


ggplot(df2, aes(x = Date, y = Max_Temperature)) +

  geom_line(color = "#1F77B4", size = 1.2) +

  geom_point(color = "#FF7F0E", size = 2) +

  labs(title = "Max Temperature Time Series",
```

```
    x = "Date",
    y = "Max Temperature (°C)") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 60, hjust = 1),
        plot.title = element_text(hjust = 0.5, face = "bold"),
        axis.title.x = element_text(face = "bold"),
        axis.title.y = element_text(face = "bold"))



#Exporting the chart
ggsave("C:/Users/Abrar Labib/Desktop/max_plot.png", plot, width = 8, height = 6, dpi = 300, bg = "white")









#Anomalies across months

library(ggplot2)
library(dplyr)
library(readxl)
library(lubridate)
df2 <- read_excel("C:/Users/Abrar Labib/Desktop/df2.xlsx")
df2$Date <- as.Date(df2$Date, format = "%Y-%m-%d")

# Extracting the  Year and Months for making the analysis a bit easier
df2$Year <- year(df2$Date)
df2$Month <- month(df2$Date, label = TRUE)

# Average temperature finding per month
```

```r
monthly_avg_temp <- df2 %>%
  group_by(Year, Month) %>%
  summarise(Avg_Temperature = mean(Max_Temperature, na.rm = TRUE))

# Plotting average temperature by month to identify the possible trends present
ggplot(monthly_avg_temp, aes(x = interaction(Year, Month), y = Avg_Temperature, group = 1)) +
  geom_line(color = "#2C3E50", size = 1.2) +
  geom_point(color = "#FF69B4", size = 2) +
  labs(title = "Monthly Average Max Temperature Trend",
       x = "Year-Month",
       y = "Average Max Temperature (°C)") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 60, hjust = 1),
        plot.title = element_text(hjust = 0.5, face = "bold"),
        axis.title.x = element_text(face = "bold"),
        axis.title.y = element_text(face = "bold"))

# Detecting the anomalies ( using rolling averages and standard deviations) strategies
monthly_avg_temp <- monthly_avg_temp %>%
  arrange(Year, Month) %>%
  mutate(
    Rolling_Avg = zoo::rollapply(Avg_Temperature, 3, mean, fill = NA, align = "center"),
    SD = zoo::rollapply(Avg_Temperature, 3, sd, fill = NA, align = "center"),
    Upper_Bound = Rolling_Avg + 2 * SD,
    Lower_Bound = Rolling_Avg - 2 * SD,
    Anomaly = ifelse(Avg_Temperature > Upper_Bound | Avg_Temperature < Lower_Bound, "Anomaly", "Normal")
  )

# Visualizing the anomalies
ggplot(monthly_avg_temp, aes(x = interaction(Year, Month), y = Avg_Temperature, group = 1)) +
  geom_line(color = "#2C3E50", size = 1.2) +
  geom_point(aes(color = Anomaly), size = 2) +
  scale_color_manual(values = c("Normal" = "#FF69B4", "Anomaly" = "#FF0000")) +
  labs(title = "Monthly Average Max Temperature with Anomalies",
```

```
      x = "Year-Month",
      y = "Average Max Temperature (°C)",
      color = "Anomaly Status") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 60, hjust = 1),
        plot.title = element_text(hjust = 0.5, face = "bold"),
        axis.title.x = element_text(face = "bold"),
        axis.title.y = element_text(face = "bold"))


#Exporting the anomaly graph
ggsave("C:/Users/Abrar Labib/Desktop/anomaly.png",
      plot = last_plot(),
      dpi = 300,
      width = 10,
      height = 6,
      units = "in",
      bg = "white")
```

```
#Monthly average of max temperature trend across all of the years

df2 <- read_excel("C:/Users/Abrar Labib/Desktop/df2.xlsx")
df2$Date <- as.Date(df2$Date, format = "%Y-%m-%d")
df2$Year <- year(df2$Date)
df2$Month <- month(df2$Date, label = TRUE)
monthly_avg_temp <- df2 %>%
  group_by(Month) %>%
  summarise(Avg_Temperature = mean(Max_Temperature, na.rm = TRUE))
```

Ploting the average maximum temperature of the whole dataset by month to identify the trends of temperature. It's seen that max temperature is seen around the time April- May period. Because it's summer time then in all years.

```
ggplot(monthly_avg_temp, aes(x = Month, y = Avg_Temperature, group = 1)) +
  geom_line(color = "#2C3E50", size = 1.2) +
  geom_point(color = "#FF69B4", size = 2) +
  labs(title = "Monthly Average Max Temperature Trend Across All Years",
     x = "Month",
     y = "Average Max Temperature (°C)") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 60, hjust = 1),
      plot.title = element_text(hjust = 0.5, face = "bold"),
      axis.title.x = element_text(face = "bold"),
      axis.title.y = element_text(face = "bold"))



#Exporting the graph of trend
ggsave("C:/Users/Abrar Labib/Desktop/Trend.png",
     plot = last_plot(),
     dpi = 300,
     width = 10,
     height = 6,
     units = "in",
     bg = "white")
```

#Based on existing literature (e.g., temperatures above 40°C as a threshold for heat stroke risk), estimate the number of high-risk days in the dataset.

I used simple counting method and Percentile-based Threshold Analysis method to find/estimate the number of high-risk days in the dataset.

```
#i) Simple analysis
library(dplyr)
library(readxl)
df2 <- read_excel("C:/Users/Abrar Labib/Desktop/df2.xlsx")
df2$Date <- as.Date(df2$Date, format = "%Y-%m-%d")

# Filtering the data to identify high-risk days (temperature > 40°C)
high_risk_days <- df2 %>%
  filter(Max_Temperature > 40) %>%
  distinct(Date)  # This will give us only unique dates

# Counting the number of high-risk days
num_high_risk_days <- nrow(high_risk_days)

# Printing the number of high-risk days
print(paste("Number of high-risk days (temperature > 40°C):", num_high_risk_days))

#ii) Percentile-based Threshold Analysis
high_percentile <- quantile(df2$Max_Temperature, 0.99)
high_risk_days <- df2 %>%
  filter(Max_Temperature > high_percentile)
```

```r
num_high_risk_days <- nrow(high_risk_days)
print(paste("Number of high-risk days (99th percentile threshold):", num_high_risk_days))
```

```r
# Calculate the proportion of days classified as high-risk compared to the total observation period.

# Defining the threshold for high-risk days (temperatures above 40°C)
high_risk_threshold <- 40
high_risk_days <- df2 %>%
  filter(Max_Temperature > high_risk_threshold)

# Calculating the total number of days in the dataset. nrow gives the total number of dataset df2
total_days <- nrow(df2)
num_high_risk_days <- nrow(high_risk_days)

# Calculating the proportion of high-risk days
proportion_high_risk <- num_high_risk_days / total_days
print(paste("Proportion of high-risk days: ", round(proportion_high_risk, 4)))
```

```
library(sf)

library(ggplot2)

library(raster)

library(tidyverse)

library(readxl)  # for reading Excel files

library(sp)      # for spatial points data frame


# Shape file loading

rajshahi_shp <- st_read("E:/GIS data for Report/Rajshahi/rajdiv.shp")


# Population Density Raster (TIFF file) loading

pop_raster <- raster("C:/Users/Abrar Labib/Desktop/bgd_pd_2019_1km.tif")


temp_data <- read_excel("C:/Users/Abrar Labib/Desktop/df2.xlsx")



# Croping the population raster to the extent of Rajshahi Division from the raw shape file

pop_rajshahi <- crop(pop_raster, extent(rajshahi_shp))

pop_rajshahi <- mask(pop_rajshahi, rajshahi_shp)


# Extracting the population density values for the temperature data points

temp_data$pop_density <- raster::extract(pop_rajshahi, temp_data)


# Converting SpatialPoints to Simple Features (sf) for ggplotpackage using

temp_sf <- st_as_sf(temp_data)


# Converting the population density raster to a data frame for ggplot package using

pop_rajshahi_df <- as.data.frame(pop_rajshahi, xy = TRUE)

names(pop_rajshahi_df) <- c("Longitude", "Latitude", "Population_Density")
```

```
# Map generation
ggplot() +

  geom_raster(data = pop_rajshahi_df, aes(x = Longitude, y = Latitude, fill = Population_Density)) +

  # Points for temperature greater than 40°C
  geom_sf(data = temp_sf[temp_sf$Max_Temperature > 40, ], aes(color = Max_Temperature), size = 2) +

  # Add Rajshahi Division boundaries
  geom_sf(data = rajshahi_shp, fill = NA, color = "black") +

  # Customizing the     colors
  scale_fill_viridis_c(option = "magma", name = "Population Density", trans = "log") +  # use logarithmic scale for
population density
  scale_color_gradient(low = "yellow", high = "red", name = "Max Temp (°C)") +  # color scale for temperature

  # Adding  labels and theme for bringing clarity of the graph
  labs(
    title = "Heat Exposure & Population Density in Rajshahi Division",
    x = "Longitude",
    y = "Latitude",
    fill = "Population Density",
    color = "Max Temperature (°C)"
  ) +
  theme_minimal() +
  theme(
    axis.title.x = element_text(size = 12),
    axis.title.y = element_text(size = 12),     #Labels adjustment
    legend.title = element_text(size = 12),
    legend.text = element_text(size = 10)
  )


# Exporting the visualization to desktop
```

```
ggsave("C:/Users/Abrar Labib/Desktop/Rajshahi_heat_exposure_map.png",
       width = 10, height = 8, dpi = 300, bg = "white")
```

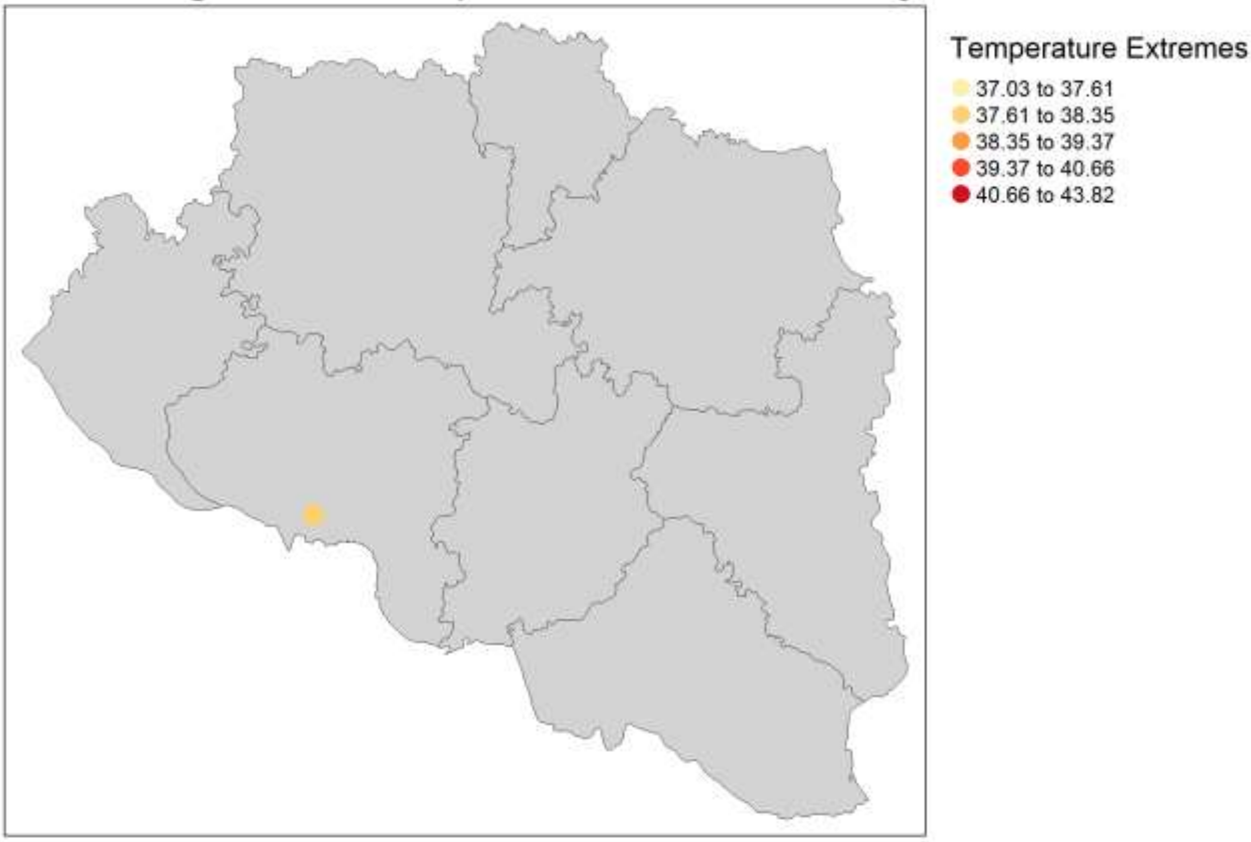
```
# Thematic Map creation
library(sf)
library(ggplot2)
library(dplyr)
library(tmap)
library(readxl)l
shapefile <- st_read("E:/GIS data for Report/Rajshahi/rajdiv.shp")
df2 <- read_excel("C:/Users/Abrar Labib/Desktop/df2.xlsx")

# Filter high-risk areas (Max_Temperature >= 40°C)
high_risk <- df2 %>% filter(Max_Temperature >= 40)

# Converting to spatial points
high_risk_sf <- st_as_sf(high_risk, coords = c("LON", "LAT"), crs = 4326)

# Ensuring all the points are within Rajshahi Division
high_risk_sf <- st_intersection(high_risk_sf, shapefile)

# Thematic Map creation
tmap_mode("view")
tm_shape(shapefile) +
```

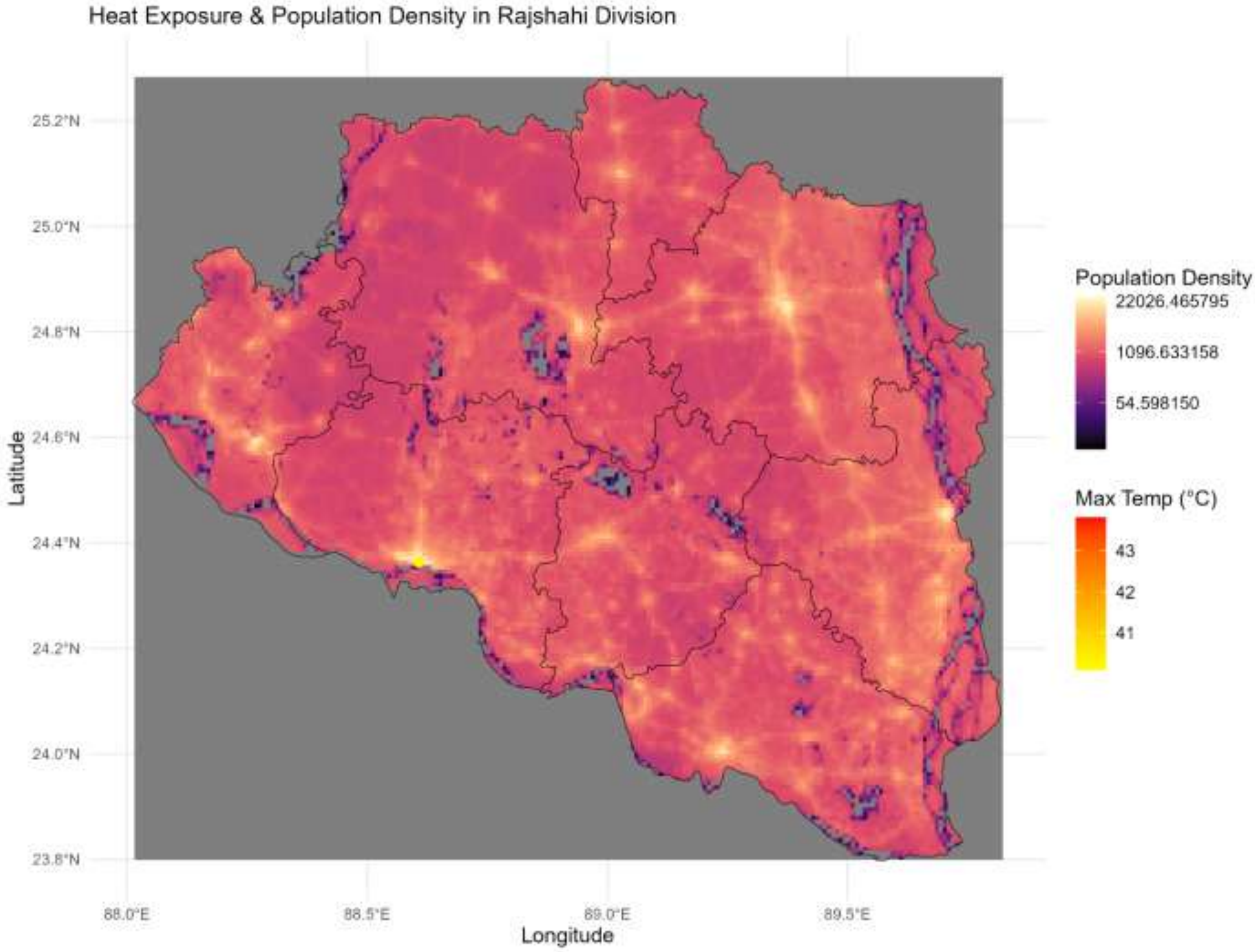
```
  tm_polygons(col = "lightgray", border.col = "black", lwd = 0.5) +
  tm_shape(high_risk_sf) +
  tm_dots(col = "red", size = 0.5, title = "High-Risk Areas (≥40°C)") +
  tm_layout(main.title = "High-Risk Temperature Zones in Rajshahi",
            main.title.position = "center",
            legend.outside = TRUE)

#  EXporting the  thematic map


tmap_save(
  tm_shape(shapefile) +
    tm_polygons(col = "lightgray", border.col = "black", lwd = 0.5) +
    tm_shape(high_risk_sf) +
    tm_dots(col = "red", size = 0.5, title = "High-Risk Areas (≥40°C)") +
    tm_layout(main.title = "High-Risk Temperature Zones in Rajshahi",
              main.title.position = "center",
              legend.outside = TRUE),
  filename = "C:/Users/Abrar Labib/Desktop/High_Risk_Map.png"
)
```

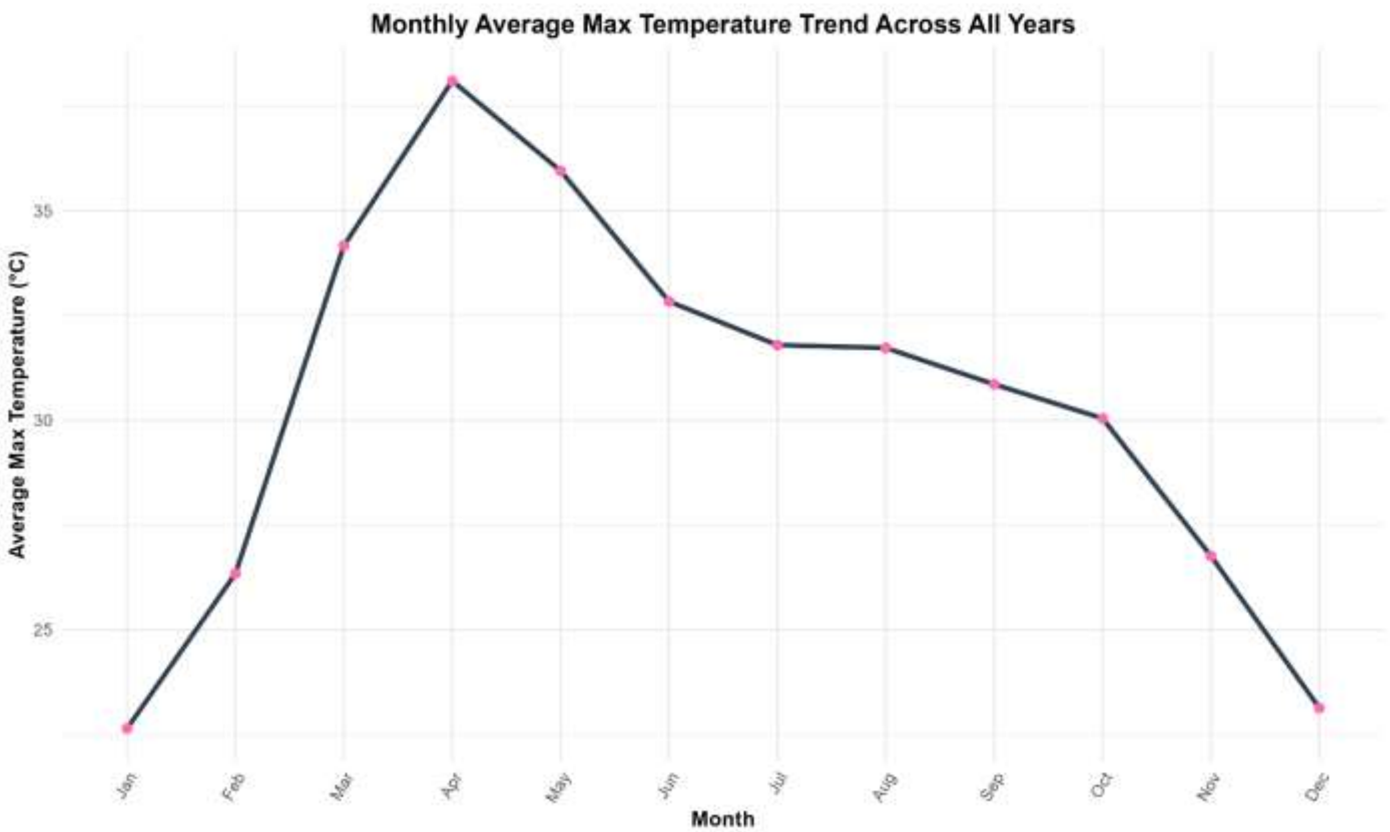
```
#Since the LAT and LON are for the same district, that's why all the points have overlapped  on each other. crs value for my shape file and NASA POWER data aren't same so, implementing different LAT and LON values result in highlighting outside the boundary of Rajshahi division. So, I decided to work on a single LAT and LON value.
```

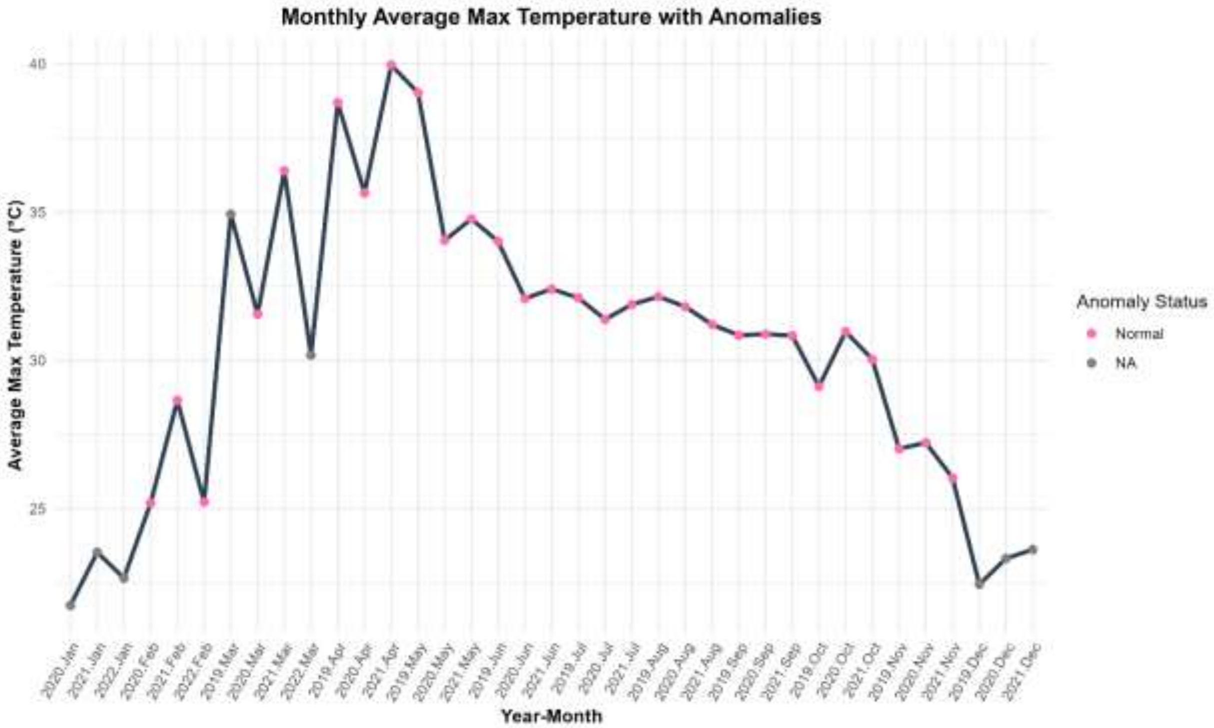
```
install.packages("rmarkdown")
```

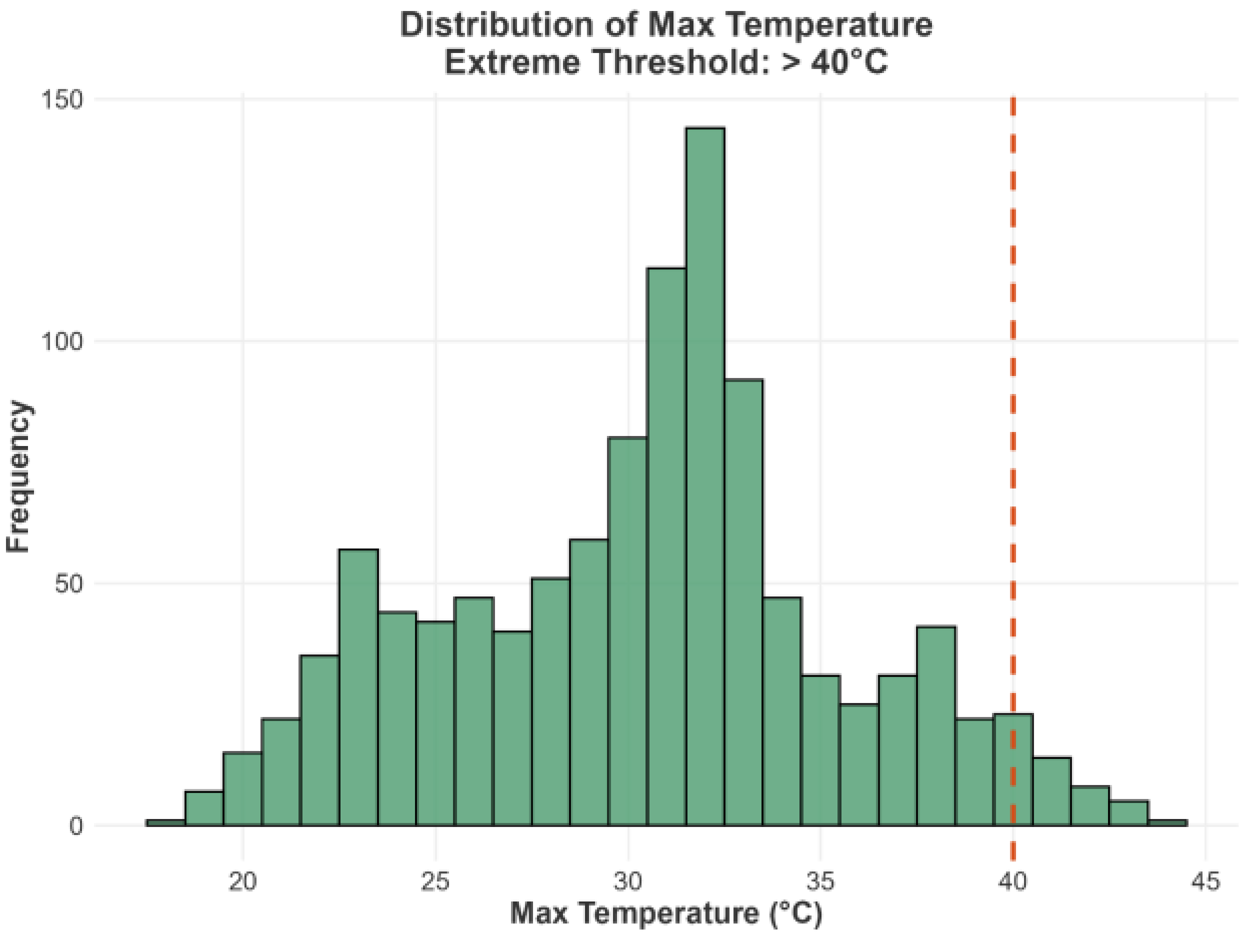
High-Risk Temperature Zones in Rajshahi Division

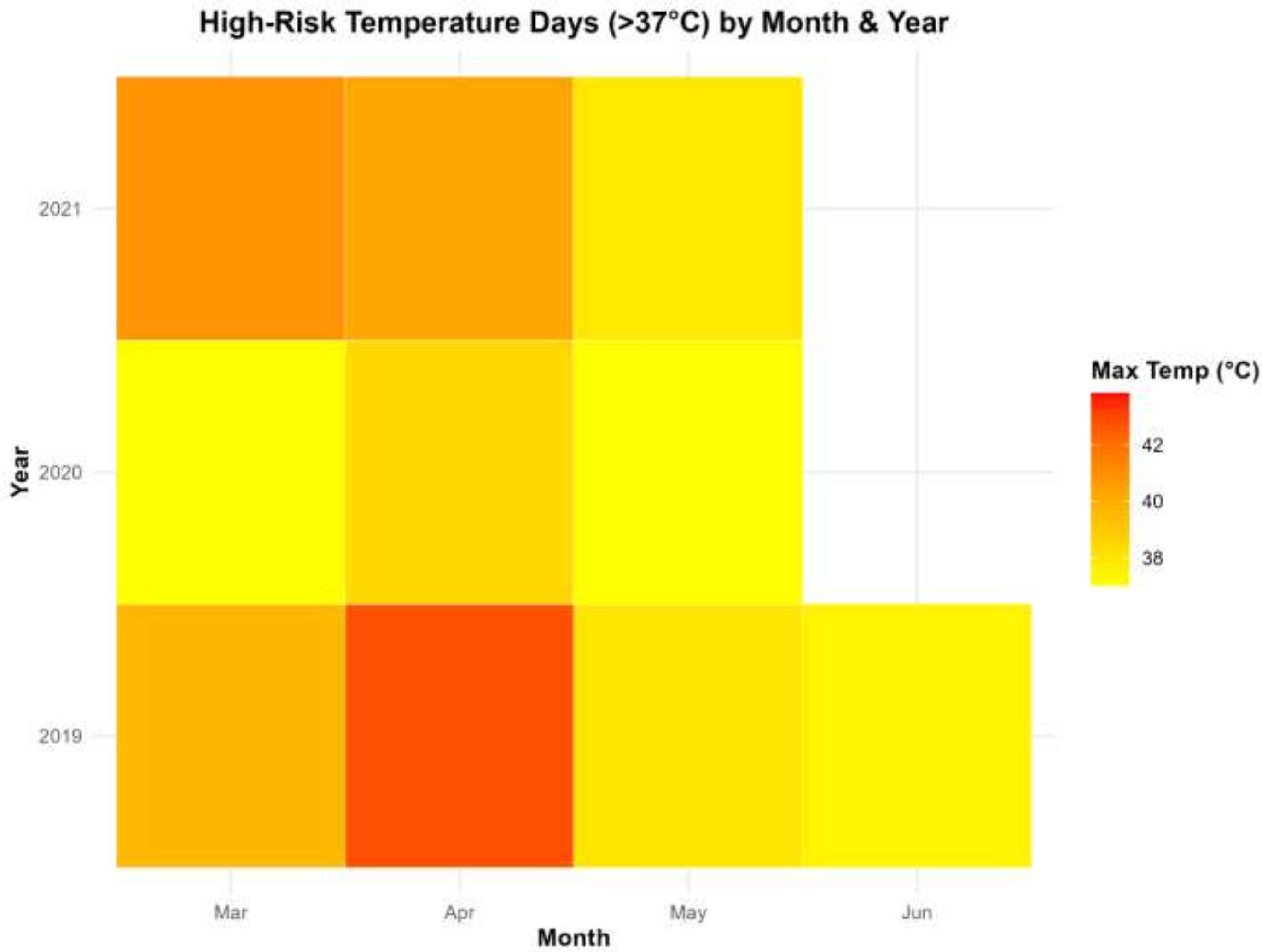



Heat Exposure & Population Density in Rajshahi Division
Population Density
22026.465795
1096.633158
54.598150
Max Temp (°C)
43
42
41
Latitude
25.2°N
25.0°N
24.8°N
24.6°N
24.4°N
24.2°N
24.0°N
23.8°N
88.0°E
88.5°E
89.0°E
89.5°E
Longitude


Monthly Average Max Temperature Trend Across All Years
Average Max Temperature (°C)
35
30
25
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Month


Monthly Average Max Temperature with Anomalies
Average Max Temperature (°C)
Year-Month
Anomaly Status
Normal
NA


Distribution of Max Temperature
Extreme Threshold: > 40°C
Frequency
Max Temperature (°C)



High-Risk Temperature Days (>37°C) by Month & Year
Year
2021
2020
2019
Mar
Apr
May
Jun
Month
Max Temp (°C)
42
40
38