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)

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")

#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, ]

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)

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, ]

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

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")

)

#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)

#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)

}

# 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

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)

num\_high\_risk\_days <- nrow(high\_risk\_days)

print(paste("Number of high-risk days (99th percentile threshold):", num\_high\_risk\_days))

# 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")











