MA615-HW4

Haochen Li

2024-09-27

B: Your next exercise is to identify and deal with the null data in the dataset...

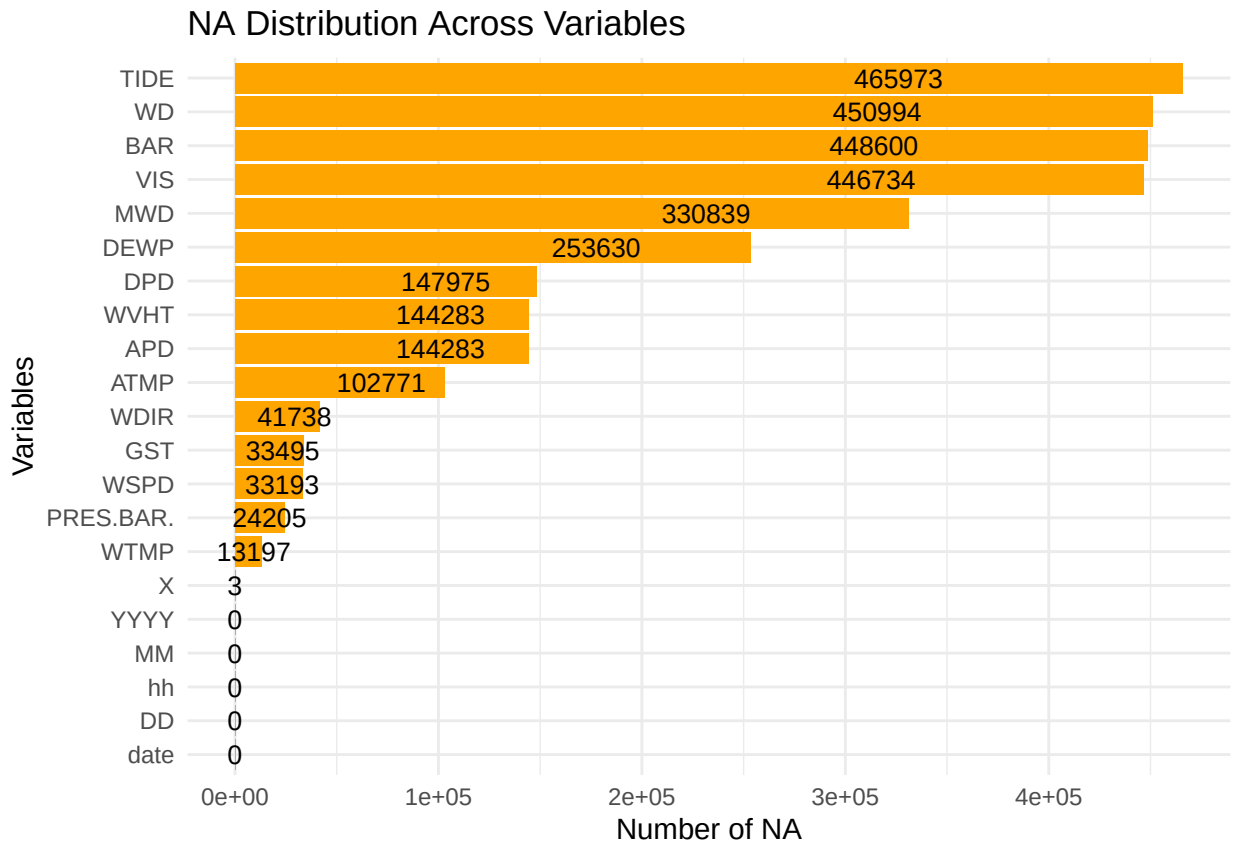
```
# Replace extreme values-NA

data <- data %>%
  mutate(YYYY = ifelse(YYYY < 100, YYYY + 1900, YYYY))%>%
  mutate(across(where(is.numeric), ~ na_if(., 99))) %>%
  mutate(across(where(is.numeric), ~ na_if(., 999))) %>%
  mutate(across(where(is.numeric), ~ na_if(., 9999)))

summary(data)
```

```
##        X                YYYY            MM              DD              hh
##  Min.   :     1   Min.   :1985   Min.   : 1.00   Min.   : 1.00   Min.   : 0.0
##  1st Qu.:116496   1st Qu.:1999   1st Qu.: 4.00   1st Qu.: 8.00   1st Qu.: 5.0
##  Median :232989   Median :2013   Median : 7.00   Median :16.00   Median :11.0
##  Mean   :232989   Mean   :2009   Mean   : 6.62   Mean   :15.73   Mean   :11.5
##  3rd Qu.:349481   3rd Qu.:2021   3rd Qu.:10.00   3rd Qu.:23.00   3rd Qu.:17.0
##  Max.   :465973   Max.   :2023   Max.   :12.00   Max.   :31.00   Max.   :23.0
##  NA's   :3
##       WDIR            WSPD            GST             WVHT
##  Min.   :  0.0   Min.   : 0.00   Min.   : 0.0    Min.   :0.00
##  1st Qu.:114.0   1st Qu.: 3.50   1st Qu.: 4.2    1st Qu.:0.41
##  Median :201.0   Median : 5.40   Median : 6.5    Median :0.66
##  Mean   :189.1   Mean   : 5.91   Mean   : 7.3    Mean   :0.87
##  3rd Qu.:277.0   3rd Qu.: 7.90   3rd Qu.: 9.7    3rd Qu.:1.06
##  Max.   :360.0   Max.   :25.70   Max.   :32.4    Max.   :9.10
##  NA's   :41738   NA's   :33193   NA's   :33495   NA's   :144283
##       DPD              APD              MWD           PRES.BAR.
##  Min.   : 0.00    Min.   : 0.00    Min.   :  0.0    Min.   : 964.6
##  1st Qu.: 4.55    1st Qu.: 3.85    1st Qu.: 77.0    1st Qu.:1010.5
##  Median : 7.69    Median : 4.70    Median : 94.0    Median :1015.8
##  Mean   : 7.38    Mean   : 4.95    Mean   :124.4    Mean   :1015.8
##  3rd Qu.:10.00    3rd Qu.: 5.84    3rd Qu.:130.0    3rd Qu.:1021.4
##  Max.   :25.00    Max.   :12.10    Max.   :360.0    Max.   :1045.8
##  NA's   :147975   NA's   :144283   NA's   :330839   NA's   :24205
```

```
##       ATMP             WTMP            DEWP             VIS
##  Min.   :-19.70   Min.   :-1.80   Min.   :-24.90   Min.   : 0.0
##  1st Qu.:  3.90   1st Qu.: 5.80   1st Qu.: -0.20   1st Qu.: 8.1
##  Median :  9.80   Median :10.50   Median :  7.10   Median : 9.4
##  Mean   :  9.87   Mean   :11.06   Mean   :  6.59   Mean   :12.5
##  3rd Qu.: 16.70   3rd Qu.:16.20   3rd Qu.: 14.60   3rd Qu.:11.6
##  Max.   : 32.10   Max.   :27.80   Max.   : 26.10   Max.   :36.0
##  NA's   :102771   NA's   :13197   NA's   :253630   NA's   :446734
##    TIDE             date                WD              BAR
##  Mode:logical   Length:465973      Min.   :  2.0    Min.   : 972.2
##  NA's:465973    Class :character   1st Qu.:128.0    1st Qu.:1010.0
##                 Mode  :character   Median :215.0    Median :1015.0
##                                    Mean   :201.2    Mean   :1014.9
##                                    3rd Qu.:288.0    3rd Qu.:1020.4
##                                    Max.   :360.0    Max.   :1040.9
##                                    NA's   :450994   NA's   :448600
```

Answer: Replacing 'extreme' data may not always be suitable. For instance, in cases where 999 could represent a valid extreme measurement rather than missing data, replacing it with NA might obscure significant insights.

```r
# Convert- Plot
data$date <- as.Date(paste(data$YYYY, data$MM, data$DD, sep="-"))

# Count
na_counts <- sapply(data, function(x) sum(is.na(x)))
na_df <- data.frame(Variable = names(na_counts), NA_Counts = na_counts)

# Plot
ggplot(na_df, aes(x = NA_Counts, y = reorder(Variable, NA_Counts))) +
  geom_bar(stat = "identity", fill = "Orange") +
  geom_text(aes(label = NA_Counts),
            position = position_stack(vjust = 0.7),
            size = 3.5, color = "Black") +
  labs(title = "NA Distribution Across Variables", x = "Number of NA", y = "Variables") +
  theme_minimal()
```



The NA distribution reveals that variables like TIDE, WD, BAR, and VIS have the most missing data, with each having over 400,000 entries marked as NA. On the other hand, variables such as WTMP and PRES.BAR show much fewer missing values, with as low as 13,000.

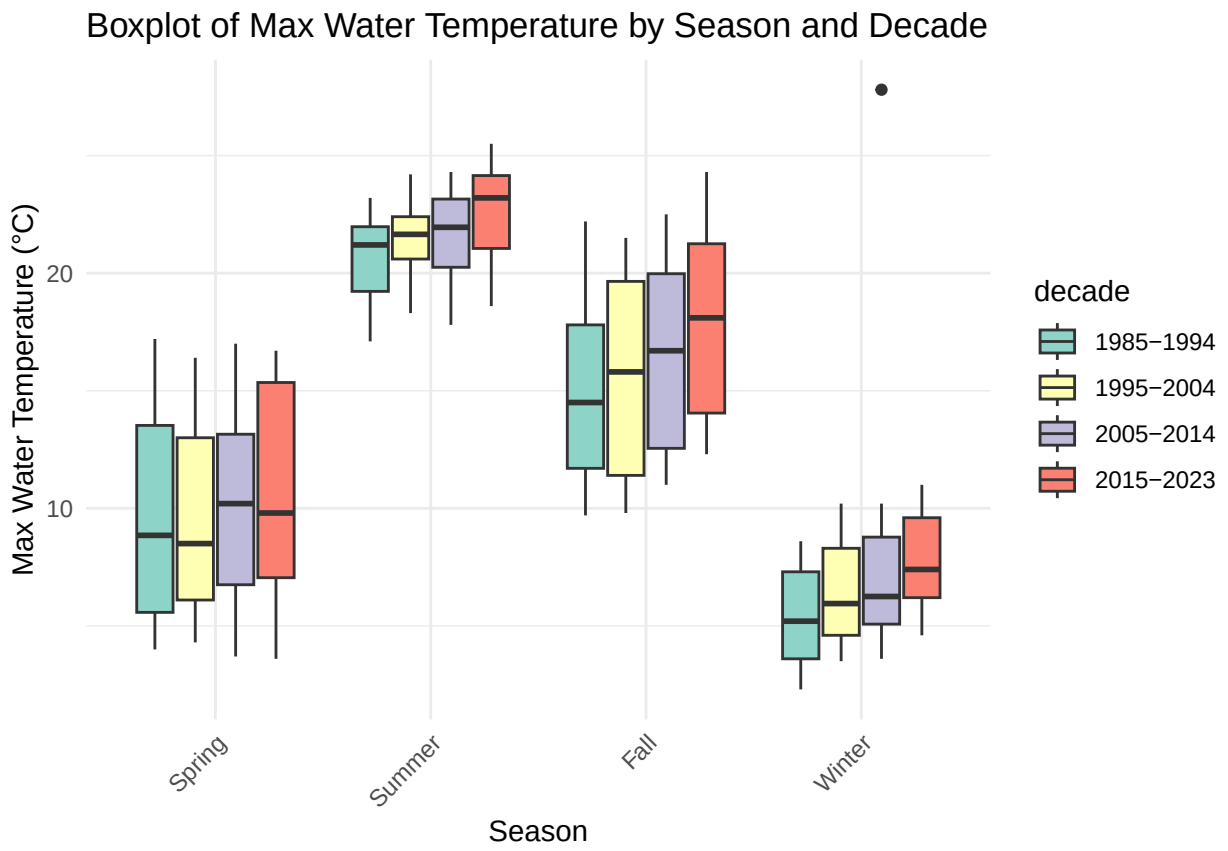
C: Can you use the Buoy data to see the effects of climate change?...

Answer: Yes. The analysis of maximum water, air, and dew point temperatures over the decades shows a consistent upward trend across all seasons, reinforcing the impact of climate change. Across Spring, Summer, Fall, and Winter, the data reveals that the highest temperatures for water (WTMP), air (ATMP), and dew point (DEWP) have steadily increased from the 1985-1994 period to the 2015-2023 period. This rise in both sea surface and atmospheric temperatures reflects broader environmental changes, including the potential for more extreme weather events and disruptions in ecosystems. The evidence strongly supports the conclusion that climate change is driving significant increases in both water and atmospheric temperatures.

```r
# Convert- date
data <- data %>%
  mutate(date = as.Date(date))

# Group by month
data <- data %>%
  mutate(year_month = floor_date(date, "month"))
```

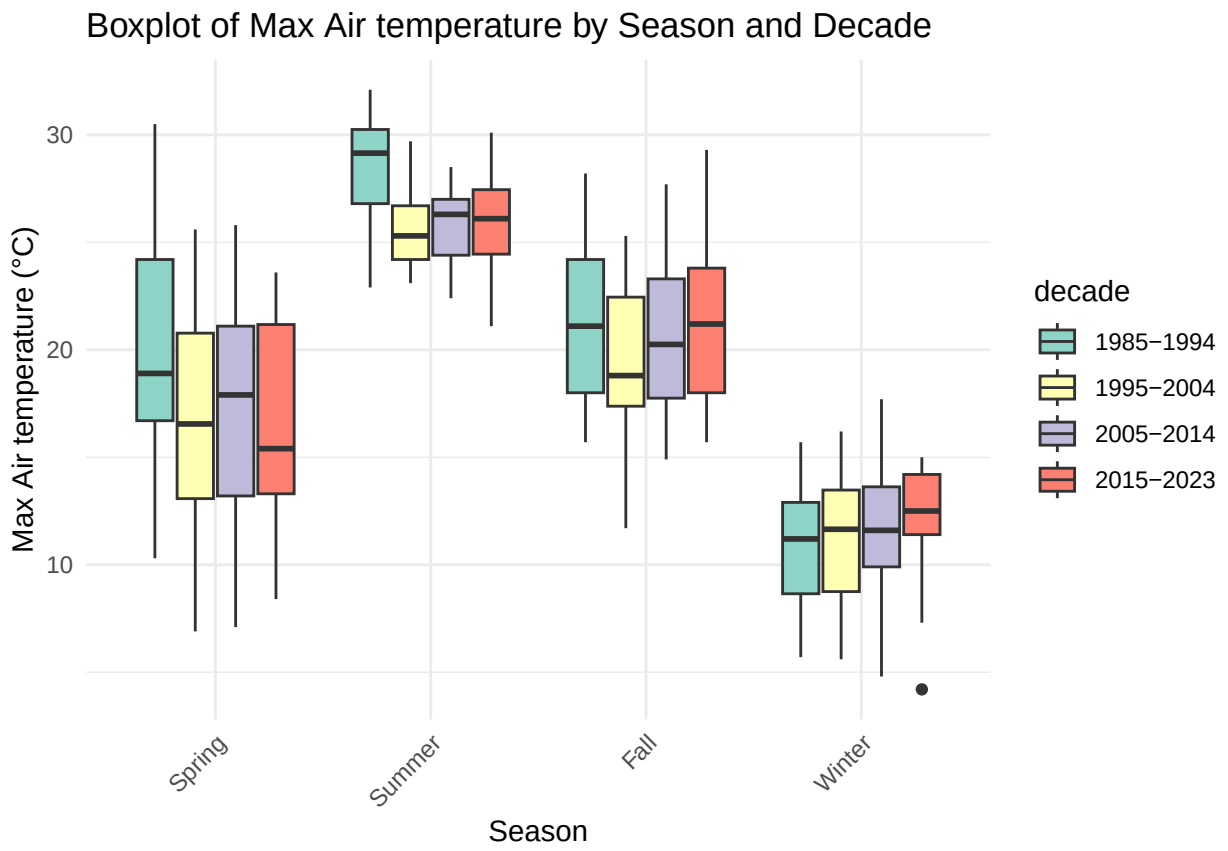
```r
# Group by decade
data <- data %>%
  mutate(decade = case_when(
    YYYY >= 1985 & YYYY <= 1994 ~ "1985-1994",
    YYYY >= 1995 & YYYY <= 2004 ~ "1995-2004",
    YYYY >= 2005 & YYYY <= 2014 ~ "2005-2014",
    YYYY >= 2015 & YYYY <= 2023 ~ "2015-2023"
  ))

# Remove rows where ATMP, WTMP, and DEWP are all NA
data <- data %>%
  filter(!(is.na(ATMP) & is.na(WTMP) & is.na(DEWP)))

# Data 4 plot
Max_temp_MM <- data %>%
  group_by(decade, year_month) %>%
  summarise(
    max_ATMP = ifelse(all(is.na(ATMP)), NA, max(ATMP, na.rm = TRUE)),
    max_WTMP = ifelse(all(is.na(WTMP)), NA, max(WTMP, na.rm = TRUE)),
    max_DEWP = ifelse(all(is.na(DEWP)), NA, max(DEWP, na.rm = TRUE)),
    .groups = 'drop'
  )

# Group by Season
Max_temp_MM <- Max_temp_MM %>%
  mutate(
    month = month(year_month),
    season = case_when(
      month %in% c(3, 4, 5) ~ "Spring",
      month %in% c(6, 7, 8) ~ "Summer",
      month %in% c(9, 10, 11) ~ "Fall",
      month %in% c(12, 1, 2) ~ "Winter"
    ),
    season = factor(season, levels = c("Spring", "Summer", "Fall", "Winter"))
  )


# Boxplot of Max Water Temperature by Season and Decade
ggplot(Max_temp_MM, aes(x = season, y = max_WTMP, fill = decade)) +
  geom_boxplot() +
  labs(
    title = "Boxplot of Max Water Temperature by Season and Decade",
    x = "Season",
    y = "Max Water Temperature (°C)"
  ) +
  theme_minimal() +
  scale_fill_brewer(palette = "Set3") +  # Apply color palette
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

```
## Warning: Removed 15 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```



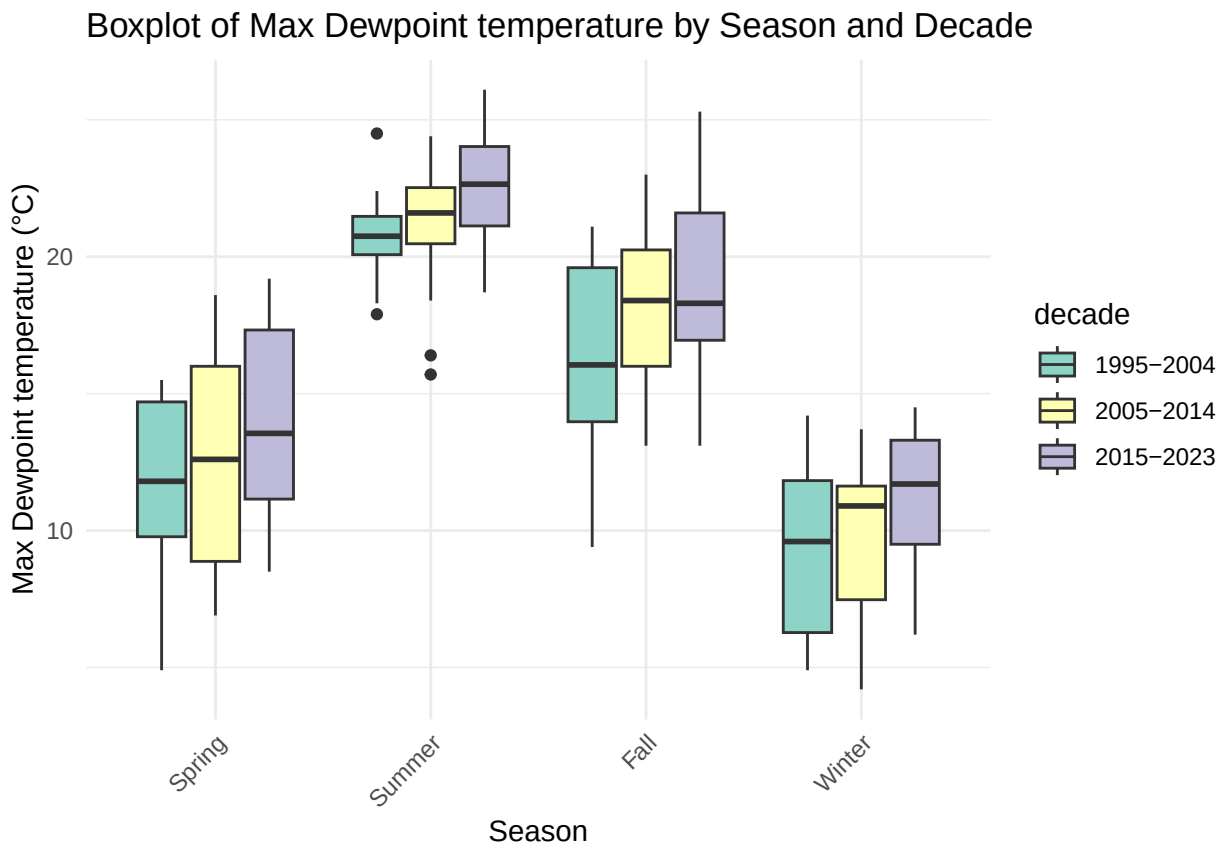

```
# Boxplot of Max Air temperature by Season and Decade
ggplot(Max_temp_MM, aes(x = season, y = max_ATMP, fill = decade)) +
  geom_boxplot() +
  labs(
    title = "Boxplot of Max Air temperature by Season and Decade",
    x = "Season",
    y = "Max Air temperature (°C)"
  ) +
  theme_minimal() +
  scale_fill_brewer(palette = "Set3") +  # Apply color palette
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

```
## Warning: Removed 21 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```




```
# Boxplot of Max Dewpoint temperature by Season and Decade
ggplot(Max_temp_MM, aes(x = season, y = max_DEWP, fill = decade)) +
  geom_boxplot() +
  labs(
    title = "Boxplot of Max Dewpoint temperature by Season and Decade",
    x = "Season",
    y = "Max Dewpoint temperature (°C)"
  ) +
  theme_minimal() +
  scale_fill_brewer(palette = "Set3") +  # Apply color palette
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

```
## Warning: Removed 228 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```



D: As part of this Homework, you have been given data for rainfall in Boston from 1985 to the end of 2013.

```
rain$DATE <- ymd_hms(rain$DATE)
```

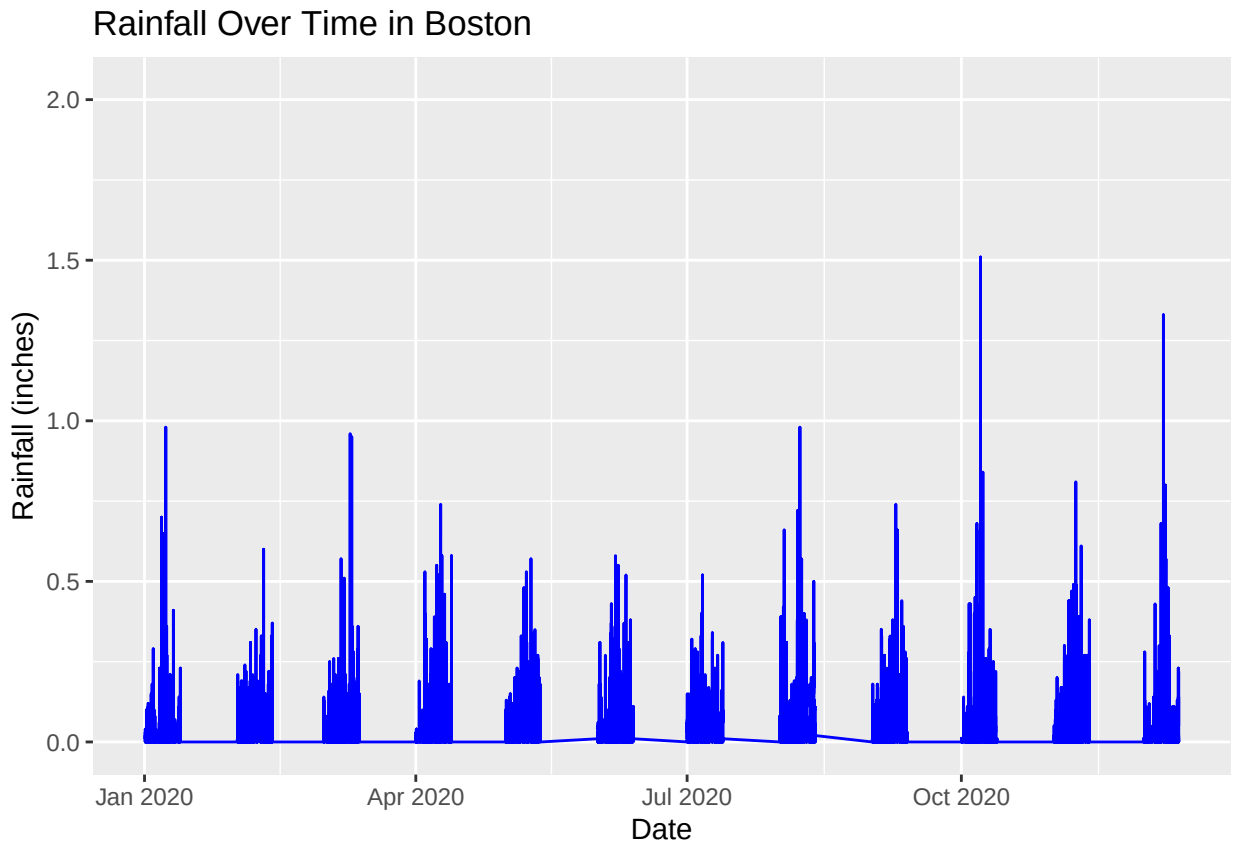
```
## Warning: 18921 failed to parse.
```

```
rain <- rain %>% arrange(DATE)

summary_stats <- rain %>%
  summarise(
    mean_rainfall = mean(HPCP, na.rm = TRUE),
    median_rainfall = median(HPCP, na.rm = TRUE),
    total_count = n(),
    max_rainfall = max(HPCP, na.rm = TRUE),
    min_rainfall = min(HPCP, na.rm = TRUE),
    sd_rainfall = sd(HPCP, na.rm = TRUE)
  )

# Display the summary table
print(summary_stats)
```

```
##   mean_rainfall median_rainfall total_count max_rainfall min_rainfall
## 1     0.0387485            0.01       31714         2.03            0
##   sd_rainfall
## 1  0.07634701
```

```r
ggplot(rain, aes(x = DATE, y = HPCP)) +
  geom_line(color = "blue") +
  labs(title = "Rainfall Over Time in Boston", x = "Date", y = "Rainfall (inches)")
```

```
## Warning: Removed 18921 rows containing missing values or values outside the scale range
## (`geom_line()`).
```



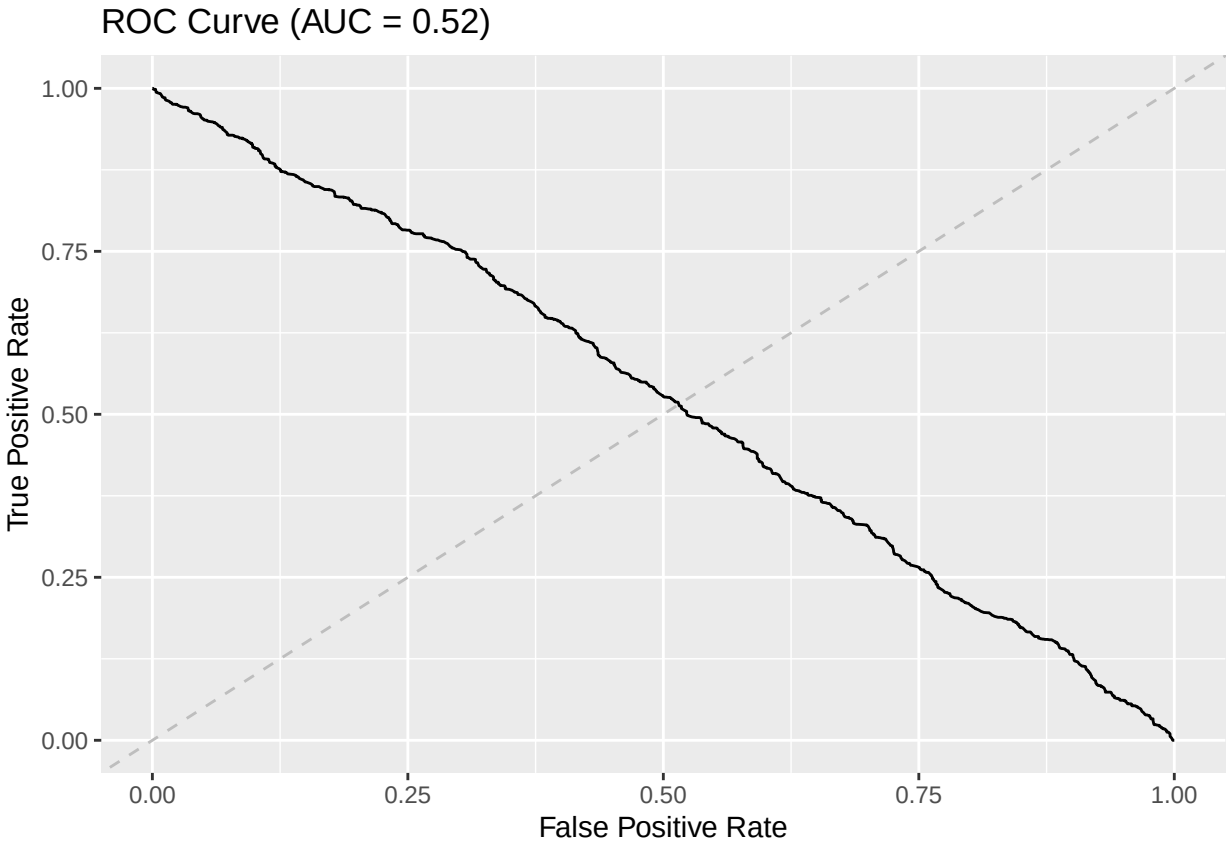

```r
# Create target variable
rain <- rain %>%
  mutate(HPCP_next_hour = lead(HPCP, n = 1),# shift HPCP for next hour prediction
         Rain_next_hour = ifelse(HPCP_next_hour > 0, 1, 0)) %>%
  drop_na(HPCP_next_hour)

# Add time features
rain <- rain %>%
  mutate(hour = hour(DATE),
         day_of_week = wday(DATE),
         month = month(DATE))

# Data for logistic regression
```

```r
rainfall_data_clean <- rain %>% select(hour, day_of_week, month, Rain_next_hour)
X <- rainfall_data_clean[, c('hour', 'day_of_week', 'month')]
y <- rainfall_data_clean$Rain_next_hour

# Split data into train and test sets
set.seed(123)
train_index <- sample(1:nrow(X), 0.7 * nrow(X))
X_train <- X[train_index,]
X_test <- X[-train_index,]
y_train <- y[train_index]
y_test <- y[-train_index]

# Fit the logistic regression model
train_data <- data.frame(X_train, Rain_next_hour = y_train)

# Fit the logistic regression model all data
logistic_model <- glm(Rain_next_hour ~ hour + day_of_week + month,
                      data = train_data, family = binomial)

# Make predictions
P_y <- predict(logistic_model, newdata = X_test, type = "response")
y_pred <- ifelse(P_y > 0.5, 1, 0)

# Evaluate model performance
accuracy <- mean(y_pred == y_test)
Roc <- roc(y_test, P_y)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```r
auc_value <- auc(Roc)

# Plot ROC Curve
ggplot(data = data.frame(fpr = Roc$specificities, tpr = Roc$sensitivities), aes(x = fpr, y = tpr)) +
  geom_line() +
  geom_abline(linetype = "dashed", color = "grey") +
  labs(title = sprintf("ROC Curve (AUC = %.2f)", auc_value),
       x = "False Positive Rate", y = "True Positive Rate")
```



The graph shows periodic rainfall spikes in Boston throughout 2020, with rainfall reaching up to 2 inches during certain events, followed by periods of no rain.

The model has weak predictive power, with the ROC curve near the diagonal and an AUC slightly above 0.5. Time-based features alone aren't sufficient for accurate rainfall prediction. Adding more relevant features like temperature, pressure, and humidity would likely improve performance.