[DITK] Data Preprocessing

Pre-liminaries

Installing the packages needed.

Please update and run here when adding new packages

```
pkgs <- sort(c('readr', 'dplyr', 'tidyr', 'tidylog', 'lubridate', 'ggplot2', 'gridExtra', 'cowplot', 'gg

pkgs_install <- pkgs[!(pkgs %in% installed.packages()[,"Package"])]
if(length(pkgs_install)){
  install.packages(pkgs_install)
}
library(readr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyr)
library(tidylog)
```

```
## Warning: package 'tidylog' was built under R version 4.3.3

##
## Attaching package: 'tidylog'

## The following objects are masked from 'package:tidyr':
##
##     drop_na, fill, gather, pivot_longer, pivot_wider, replace_na,
##     spread, uncount

## The following objects are masked from 'package:dplyr':
##
##     add_count, add_tally, anti_join, count, distinct, distinct_all,
```

```
##     distinct_at, distinct_if, filter, filter_all, filter_at, filter_if,
##     full_join, group_by, group_by_all, group_by_at, group_by_if,
##     inner_join, left_join, mutate, mutate_all, mutate_at, mutate_if,
##     relocate, rename, rename_all, rename_at, rename_if, rename_with,
##     right_join, sample_frac, sample_n, select, select_all, select_at,
##     select_if, semi_join, slice, slice_head, slice_max, slice_min,
##     slice_sample, slice_tail, summarise, summarise_all, summarise_at,
##     summarise_if, summarize, summarize_all, summarize_at, summarize_if,
##     tally, top_frac, top_n, transmute, transmute_all, transmute_at,
##     transmute_if, ungroup

## The following object is masked from 'package:stats':
##
##     filter
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(ggplot2)
library(gridExtra)
```

```
## Warning: package 'gridExtra' was built under R version 4.3.3

##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(cowplot)
```

```
## Warning: package 'cowplot' was built under R version 4.3.3

##
## Attaching package: 'cowplot'

## The following object is masked from 'package:lubridate':
##
##     stamp
```

```
library(ggmap)
```

```
## Warning: package 'ggmap' was built under R version 4.3.3
```

```
## i Google's Terms of Service: <https://mapsplatform.google.com>
##   Stadia Maps' Terms of Service: <https://stadiamaps.com/terms-of-service/>
##   OpenStreetMap's Tile Usage Policy: <https://operations.osmfoundation.org/policies/tiles/>
## i Please cite ggmap if you use it! Use `citation("ggmap")` for details.

##
## Attaching package: 'ggmap'

## The following object is masked from 'package:cowplot':
##
##     theme_nothing
```

```
library(leaflet)
```

```
## Warning: package 'leaflet' was built under R version 4.3.3
```

```
library(viridis)
```

```
## Warning: package 'viridis' was built under R version 4.3.3

## Loading required package: viridisLite
```

```
library(htmltools)
library(DataCombine)
```

```
## Warning: package 'DataCombine' was built under R version 4.3.3
```

```
library(ggplotify)
```

```
## Warning: package 'ggplotify' was built under R version 4.3.3
```

```
library(ggpubr)
```

```
## Warning: package 'ggpubr' was built under R version 4.3.3

##
## Attaching package: 'ggpubr'

## The following object is masked from 'package:cowplot':
##
##     get_legend

## The following objects are masked from 'package:tidylog':
##
##     group_by, mutate
```

```
tinytex::install_tinytex()
library(tinytex)
```

Importing the datasets

We import the dataset using `read_csv` and use `view` `head` and `tail` to view some records.

```
dataset <- read_csv("datasets/2013-2023.csv")
```

```
## Rows: 97421 Columns: 11
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (9): Temperature, Dew Point, Humidity, Wind, Wind Speed, Wind Gust, Pre...
## date (1): Date
## time (1): Time
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
View(dataset)
head(dataset)
```

```
## # A tibble: 6 x 11
##   Date       Time   Temperature `Dew Point` Humidity Wind  `Wind Speed`
##   <date>     <time> <chr>       <chr>       <chr>    <chr> <chr>
## 1 2013-01-01 00:00  79 °F       70 °F       74 %     WNW   3 mph
## 2 2013-01-01 01:00  77 °F       70 °F       78 %     SE    3 mph
## 3 2013-01-01 02:00  77 °F       70 °F       78 %     CALM  0 mph
## 4 2013-01-01 03:00  77 °F       72 °F       83 %     CALM  0 mph
## 5 2013-01-01 04:00  79 °F       72 °F       78 %     WNW   3 mph
## 6 2013-01-01 05:00  79 °F       68 °F       69 %     WNW   5 mph
## # i 4 more variables: `Wind Gust` <chr>, Pressure <chr>, Precip. <chr>,
## #   Condition <chr>
```

```
tail(dataset)
```

```
## # A tibble: 6 x 11
##   Date       Time   Temperature `Dew Point` Humidity Wind  `Wind Speed`
##   <date>     <time> <chr>       <chr>       <chr>    <chr> <chr>
## 1 2023-12-31 18:00  86 °F       75 °F       70 %     E     8 mph
## 2 2023-12-31 19:00  84 °F       75 °F       74 %     ESE   5 mph
## 3 2023-12-31 20:00  84 °F       75 °F       74 %     ESE   6 mph
## 4 2023-12-31 21:00  84 °F       75 °F       74 %     SE    7 mph
## 5 2023-12-31 22:00  82 °F       75 °F       79 %     ESE   5 mph
## 6 2023-12-31 23:00  82 °F       73 °F       74 %     ESE   5 mph
## # i 4 more variables: `Wind Gust` <chr>, Pressure <chr>, Precip. <chr>,
## #   Condition <chr>
```

```
spec(dataset)
```

```
## cols(
##   Date = col_date(format = ""),
##   Time = col_time(format = ""),
##   Temperature = col_character(),
##   `Dew Point` = col_character(),
##   Humidity = col_character(),
##   Wind = col_character(),
##   `Wind Speed` = col_character(),
##   `Wind Gust` = col_character(),
##   Pressure = col_character(),
##   Precip. = col_character(),
##   Condition = col_character()
## )
```

Data Pre-processing

Changing column names For consistency and format

```
colnames(dataset) <- c('DATE', 'TIME', 'TEMP', 'DP', 'HUM', 'WND_DIR', 'WND_SPD', 'WND_GST', 'PRES', 'PR
```

Remove units and convert values to numerical

```
dataset$TEMP <- as.numeric(gsub('[^0-9.]','', dataset$TEMP))
dataset$DP <- as.numeric(gsub('[^0-9.]','', dataset$DP))
dataset$HUM <- as.numeric(gsub('[^0-9.]','', dataset$HUM))
dataset$WND_SPD <- as.numeric(gsub('[^0-9.]','', dataset$WND_SPD))
dataset$WND_GST <- as.numeric(gsub('[^0-9.]','', dataset$WND_GST))
dataset$PRES <- as.numeric(gsub('[^0-9.]','', dataset$PRES))
dataset$PRECIP <- as.numeric(gsub('[^0-9.]','', dataset$PRECIP))
head(dataset)
```

```
## # A tibble: 6 x 11
##   DATE       TIME    TEMP    DP   HUM WND_DIR WND_SPD WND_GST  PRES PRECIP COND
##   <date>     <time> <dbl> <dbl> <dbl> <chr>     <dbl>   <dbl> <dbl>  <dbl> <chr>
## 1 2013-01-01 00:00     79    70    74 WNW           3       0  29.8      0 Fair
## 2 2013-01-01 01:00     77    70    78 SE            3       0  29.8      0 Fair
## 3 2013-01-01 02:00     77    70    78 CALM          0       0  29.8      0 Fair
## 4 2013-01-01 03:00     77    72    83 CALM          0       0  29.8      0 Most~
## 5 2013-01-01 04:00     79    72    78 WNW           3       0  29.8      0 Most~
## 6 2013-01-01 05:00     79    68    69 WNW           5       0  29.8      0 Clou~
```

```
tail(dataset)
```

```
## # A tibble: 6 x 11
##   DATE       TIME    TEMP    DP   HUM WND_DIR WND_SPD WND_GST  PRES PRECIP COND
##   <date>     <time> <dbl> <dbl> <dbl> <chr>     <dbl>   <dbl> <dbl>  <dbl> <chr>
## 1 2023-12-31 18:00     86    75    70 E             8       0  29.8      0 Part~
## 2 2023-12-31 19:00     84    75    74 ESE           5       0  29.8      0 Fair
```

```
## 3 2023-12-31 20:00     84    75    74 ESE           6       0  29.8      0 Fair
## 4 2023-12-31 21:00     84    75    74 SE            7       0  29.8      0 Fair
## 5 2023-12-31 22:00     82    75    79 ESE           5       0  29.8      0 Part~
## 6 2023-12-31 23:00     82    73    74 ESE           5       0  29.8      0 Fair
```

Checking for null values

```
print("Number of NULL values under each column:")
```

```
## [1] "Number of NULL values under each column:"
```

```
colSums(is.na(dataset))
```

```
##    DATE    TIME    TEMP      DP     HUM WND_DIR WND_SPD WND_GST    PRES  PRECIP
##       0       0       0       0       0     342       0       0       0       0
##    COND
##       0
```

We will not be using WIND_DIR so we will not bother removing the NULL values.

Check for values

```
sort(table(dataset$TEMP), decreasing =T)
```

```
##
##    81    82    79    84    86    88    77    90    91    75    93    95    73
## 15047 14778 12919 12618 10408  8215  6763  5782  3452  2542  2036  1144   656
##    97     0    72    70    99    68   100    66    64   102   133   149
##   408   330   196    48    48    17     6     4     1     1     1     1
```

```
sort(table(dataset$DP), decreasing =T)
```

```
##
##    77    75    79    73    72    70    81    68    66    82    64    63     0
## 19658 17034 13391 11899  9601  7705  5492  5142  2821  1750  1416   562   330
##    84    61    59    57    86    50    54    90    55    93    95    52    91
##   257   218    69    24    13     9     5     5     4     4     4     2     2
##    97   131   133   165
##     1     1     1     1
```

```
sort(table(dataset$HUM), decreasing =T)
```

```
##
##   74  100   89   79   70   84   94   66   62   78   55   58   83   59   52   49
## 9981 9964 8984 8938 8302 7839 7422 6303 4429 2590 2560 2363 2260 1949 1708 1477
##   75   69   65   63   56   46   61   51   73    0   47   53   44   54   48   43
## 1425 1387 1240  710  653  587  567  493  464  375  354  321  302  221  195  169
```

```
##   41   57   88   50   45   71   39   40   42   37   67   38   36   34   64   35
##  158  123  119   90   84   78   50   40   26   22   19   17   14   10   10    7
##   30   32   68   26   29   13   15   27   28   31   33   80   95
##    4    3    3    2    2    1    1    1    1    1    1    1    1
```

We observe the there are 0 values in these 3 columns which the value should not appear on. These are cases of missing values. We will replace these values with the mean of the date they are observed.

We will make another dataframe without the 0 rows, calculate the mean for every date.

```
no_zero_data <- subset(dataset, dataset$TEMP != 0 & dataset$DP != 0 & dataset$HUM != 0)
```

```
mean_temp <- aggregate(no_zero_data$TEMP ~ no_zero_data$DATE, data = no_zero_data, FUN = mean)
mean_dp <- aggregate(no_zero_data$DP ~ no_zero_data$DATE, data = no_zero_data, FUN = mean)
mean_hum <- aggregate(no_zero_data$HUM ~ no_zero_data$DATE, data = no_zero_data, FUN = mean)
```

Let's turn the 0 values into NA first.

```
dataset$TEMP[dataset$TEMP == 0] <- NA
dataset$DP[dataset$DP == 0] <- NA
dataset$HUM[dataset$HUM == 0] <- NA
```

```
rows_with_na <- dataset[!complete.cases(dataset), ]
print(rows_with_na)
```

```
## # A tibble: 679 x 11
##    DATE       TIME   TEMP    DP   HUM WND_DIR WND_SPD WND_GST  PRES PRECIP COND
##    <date>     <tim> <dbl> <dbl> <dbl> <chr>     <dbl>   <dbl> <dbl>  <dbl> <chr>
##  1 2013-01-04 05:00    77    77   100 <NA>          0       0  29.8      0 Clou~
##  2 2013-01-08 05:00    77    75    94 <NA>          0       0  29.8      0 Fair
##  3 2013-01-13 12:00    NA    NA    NA VAR           7       0  29.7      0 Fair
##  4 2013-01-14 10:00    81    84    NA N            10       0  29.8      0 Clou~
##  5 2013-01-24 09:00    79    68    69 <NA>          0       0  29.9      0 Fair
##  6 2013-01-26 20:00    NA    NA    NA SE            7       0  29.8      0 Part~
##  7 2013-02-13 12:00    86    70    58 <NA>          0       0  29.9      0 Part~
##  8 2013-02-17 07:00    NA    NA    NA ESE           7       0   0        0 Fair
##  9 2013-02-21 22:00    77    77   100 <NA>          0       0  29.8      0 Clou~
## 10 2013-02-27 04:00    79    72    78 <NA>          0       0  29.8      0 Most~
## # i 669 more rows
```

```
dataset$TEMP[is.na(dataset$TEMP)] <- mean_temp$`no_zero_data$TEMP`[match(dataset$DATE, mean_temp$`no_ze
dataset$DP[is.na(dataset$DP)] <- mean_dp$`no_zero_data$DP`[match(dataset$DATE, mean_temp$`no_zero_data$
dataset$HUM[is.na(dataset$HUM)] <- mean_hum$`no_zero_data$HUM`[match(dataset$DATE, mean_temp$`no_zero_d
```

Let's check for NA values

```
colSums(is.na(dataset))
```

```
##    DATE    TIME    TEMP      DP     HUM WND_DIR WND_SPD WND_GST    PRES  PRECIP
##       0       0       0       0       0     342       0       0       0       0
##    COND
##       0
```

We've successfully replaced missing values with the mean.

Getting the average of the columns per day

We are more interested in daily instead of hourly observations so we will get the daily averages of each and also get the min and max temperature for each day.

```
average_temp <- aggregate(dataset$TEMP ~ dataset$DATE, data = dataset, FUN = mean)
average_dp <- aggregate(dataset$DP ~ dataset$DATE, data = dataset, FUN = mean)
average_hum <- aggregate(dataset$HUM ~ dataset$DATE, data = dataset, FUN = mean)
average_wndspd <- aggregate(dataset$WND_SPD ~ dataset$DATE, data = dataset, FUN = mean)
average_wndgst <- aggregate(dataset$WND_GST ~ dataset$DATE, data = dataset, FUN = mean)
average_pres <- aggregate(dataset$PRES ~ dataset$DATE, data = dataset, FUN = mean)
average_precip <- aggregate(dataset$PRECIP ~ dataset$DATE, data = dataset, FUN = mean)
min_temp <- aggregate(dataset$TEMP ~ dataset$DATE, data = dataset, FUN = min)
max_temp <- aggregate(dataset$TEMP ~ dataset$DATE, data = dataset, FUN = max)
```

###Merge all average, min, and max values in one dataframe

```
dfs <- list(min_temp, max_temp, average_temp, average_dp, average_hum, average_pres, average_precip,  a
merged_df <- Reduce(function(x, y) merge(x, y, by = "dataset$DATE"), dfs)
colnames(merged_df) <- c('DATE', 'MIN. TEMP', 'MAX. TEMP', 'AVE. TEMP', 'AVE. DP', 'AVE. HUM', 'AVE. PR
head(merged_df)
```

```
##         DATE MIN. TEMP MAX. TEMP AVE. TEMP  AVE. DP AVE. HUM AVE. PRES
## 1 2013-01-01        77        84  79.76000 71.44000 75.76000  29.81800
## 2 2013-01-02        75        91  81.58333 71.12500 71.25000  29.83875
## 3 2013-01-03        77        86  81.04167 73.04167 77.04167  29.82000
## 4 2013-01-04        75        88  81.54167 73.54167 78.95833  29.75875
## 5 2013-01-05        77        91  82.87500 72.62500 73.33333  29.73125
## 6 2013-01-06        75        90  82.33333 71.91667 73.16667  29.77750
##   AVE. PRECIP AVE. WND_GST AVE. WND_SPD
## 1           0            0     5.440000
## 2           0            0     6.375000
## 3           0            0     6.291667
## 4           0            0     5.750000
## 5           0            0     6.208333
## 6           0            0     6.208333
```

```
tail(merged_df)
```

```
##            DATE MIN. TEMP MAX. TEMP AVE. TEMP  AVE. DP AVE. HUM AVE. PRES
## 4010 2023-12-26        79        86  81.65217 74.60870 80.17391  29.88625
## 4011 2023-12-27        77        90  82.30769 74.38462 78.84615  29.91077
## 4012 2023-12-28        79        90  82.50000 75.08333 79.62500  29.88125
## 4013 2023-12-29        77        90  83.04167 73.66667 74.37500  29.83250
## 4014 2023-12-30        77        91  83.60000 75.00000 76.44000  29.78680
## 4015 2023-12-31        79        91  84.45833 75.08333 74.75000  29.79500
##      AVE. PRECIP AVE. WND_GST AVE. WND_SPD
## 4010           0            0     4.791667
## 4011           0            0     3.730769
## 4012           0            0     5.083333
## 4013           0            0     4.958333
## 4014           0            0     4.400000
## 4015           0            0     5.375000
```

Date completeness

We will now check if we have all the days from 2013 to 2023 in our dataset

```
start_date <- as.Date("2013-01-01")  # Replace "yyyy-mm-dd" with the start date of your range
end_date <- as.Date("2023-12-31")    # Replace "yyyy-mm-dd" with the end date of your range
date_range <- seq(start_date, end_date, by = "day")

# Get the unique dates available in your dataset
data_dates <- unique(as.Date(merged_df$DATE))

# Check if there are any missing dates in the range
missing_dates <- setdiff(date_range, data_dates)

if (length(missing_dates) == 0) {
  print("Data is available for all days in the range of years.")
} else {
  print("Data is missing for the following days:")
  print(as.Date(missing_dates))
}
```

```
## [1] "Data is missing for the following days:"
## [1] "2020-07-15" "2020-07-16"
```

We have 2 days with missing data. Let's add their data using the average of the month.

```
tail(merged_df)
```

```
##            DATE MIN. TEMP MAX. TEMP AVE. TEMP  AVE. DP AVE. HUM AVE. PRES
## 4010 2023-12-26        79        86  81.65217 74.60870 80.17391  29.88625
## 4011 2023-12-27        77        90  82.30769 74.38462 78.84615  29.91077
## 4012 2023-12-28        79        90  82.50000 75.08333 79.62500  29.88125
## 4013 2023-12-29        77        90  83.04167 73.66667 74.37500  29.83250
## 4014 2023-12-30        77        91  83.60000 75.00000 76.44000  29.78680
## 4015 2023-12-31        79        91  84.45833 75.08333 74.75000  29.79500
##      AVE. PRECIP AVE. WND_GST AVE. WND_SPD
## 4010           0            0     4.791667
## 4011           0            0     3.730769
## 4012           0            0     5.083333
## 4013           0            0     4.958333
## 4014           0            0     4.400000
## 4015           0            0     5.375000
```

```
merged_df$MONTH <- format(merged_df$DATE, "%m")
merged_df$YEAR <- format(merged_df$DATE, "%Y")

monthly_average <- aggregate(. ~ MONTH + YEAR, data = merged_df, FUN = mean, na.rm = TRUE)
merged_df <- subset(merged_df, select = -MONTH)
merged_df <- subset(merged_df, select = -YEAR)
```

From the monthly_average table, get the mean of the month of 7-2019 and use it as values for the 2 rows

```
first_row <- c(DATE = as.Date("2020-07-15"), MIN.TEMP = 79.72414, MAX.TEMP = 91.75862, AVE.TEMP = 85.678
second_row <- c(DATE = as.Date("2020-07-16"), MIN.TEMP = 79.72414, MAX.TEMP = 91.75862, AVE.TEMP = 85.67
```

Let's specify the index where they'll be added.

```
first_index <- 2753
second_index <- 2754
```

We use rbind to merge the new rows with merged_df dataframe

```
merged_df <- InsertRow(merged_df, first_row, first_index)
merged_df <- InsertRow(merged_df, second_row, second_index)
```

Converting Fahrenheit to Celsius

Since we use Celsius commonly in the Philippines, we will be converting the temperature values into celsius.

```
fahrenheit_to_celsius <- function(fahrenheit) {
  celsius <- (fahrenheit - 32) * (5/9)
  return(celsius)
}
```

```
merged_df$`AVE. TEMP` <- fahrenheit_to_celsius(merged_df$`AVE. TEMP`)
merged_df$`MIN. TEMP` <- fahrenheit_to_celsius(merged_df$`MIN. TEMP`)
merged_df$`MAX. TEMP` <- fahrenheit_to_celsius(merged_df$`MAX. TEMP`)
merged_df$`AVE. DP` <- fahrenheit_to_celsius(merged_df$`AVE. DP`)
```

Dropping Columns

We see that there are only 0 values under Precipitation even if the Conditions column in previous data frames mentioned rain. We will drop this column.

```
merged_df <- subset(merged_df, select = -`AVE. PRECIP`)
```

View of Pre-processed Data Frame

```
View(merged_df)
head(merged_df)
```

```
##         DATE MIN. TEMP MAX. TEMP AVE. TEMP  AVE. DP AVE. HUM AVE. PRES
## 1 2013-01-01  25.00000  28.88889  26.53333 21.91111 75.76000  29.81800
## 2 2013-01-02  23.88889  32.77778  27.54630 21.73611 71.25000  29.83875
## 3 2013-01-03  25.00000  30.00000  27.24537 22.80093 77.04167  29.82000
## 4 2013-01-04  23.88889  31.11111  27.52315 23.07870 78.95833  29.75875
## 5 2013-01-05  25.00000  32.77778  28.26389 22.56944 73.33333  29.73125
## 6 2013-01-06  23.88889  32.22222  27.96296 22.17593 73.16667  29.77750
##   AVE. WND_GST AVE. WND_SPD
## 1            0     5.440000
```

```
## 2            0     6.375000
## 3            0     6.291667
## 4            0     5.750000
## 5            0     6.208333
## 6            0     6.208333
```

```
tail(merged_df)
```

```
##            DATE MIN. TEMP MAX. TEMP AVE. TEMP  AVE. DP AVE. HUM AVE. PRES
## 4012 2023-12-26  26.11111  30.00000  27.58454 23.67150 80.17391  29.88625
## 4013 2023-12-27  25.00000  32.22222  27.94872 23.54701 78.84615  29.91077
## 4014 2023-12-28  26.11111  32.22222  28.05556 23.93519 79.62500  29.88125
## 4015 2023-12-29  25.00000  32.22222  28.35648 23.14815 74.37500  29.83250
## 4016 2023-12-30  25.00000  32.77778  28.66667 23.88889 76.44000  29.78680
## 4017 2023-12-31  26.11111  32.77778  29.14352 23.93519 74.75000  29.79500
##      AVE. WND_GST AVE. WND_SPD
## 4012            0     4.791667
## 4013            0     3.730769
## 4014            0     5.083333
## 4015            0     4.958333
## 4016            0     4.400000
## 4017            0     5.375000
```

Exploratory Data Analysis

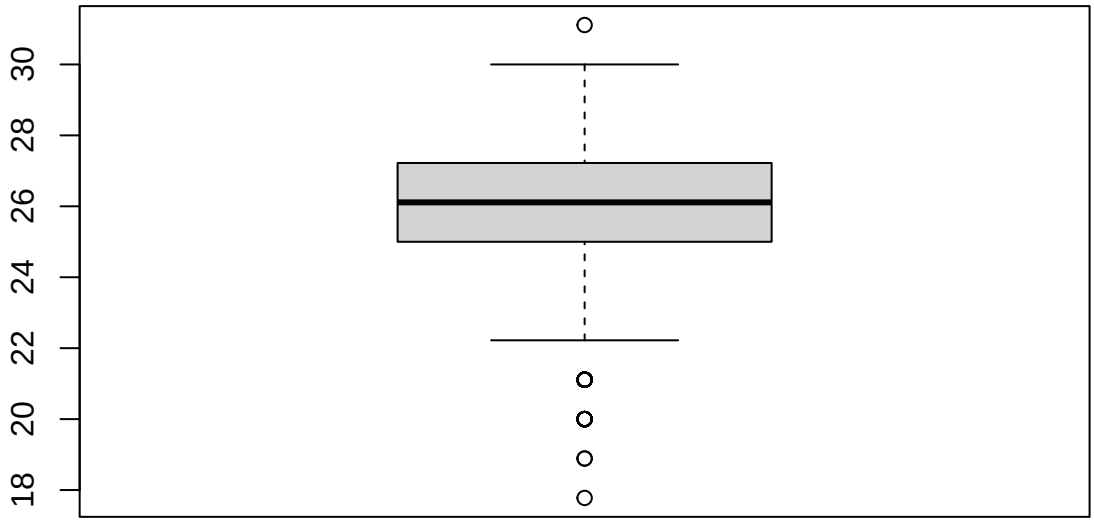
Summary

```
summary(merged_df)
```

```
##       DATE              MIN. TEMP       MAX. TEMP       AVE. TEMP
##  Min.   :2013-01-01   Min.   :17.78   Min.   :23.89   Min.   :22.64
##  1st Qu.:2015-10-02   1st Qu.:25.00   1st Qu.:31.11   1st Qu.:27.47
##  Median :2018-07-02   Median :26.11   Median :32.22   Median :28.50
##  Mean   :2018-07-02   Mean   :25.64   Mean   :32.12   Mean   :28.50
##  3rd Qu.:2021-04-01   3rd Qu.:27.22   3rd Qu.:32.78   3rd Qu.:29.49
##  Max.   :2023-12-31   Max.   :31.11   Max.   :65.00   Max.   :33.64
##     AVE. DP         AVE. HUM        AVE. PRES      AVE. WND_GST
##  Min.   :14.58   Min.   : 54.25   Min.   :27.54   Min.   : 0.000
##  1st Qu.:22.31   1st Qu.: 68.29   1st Qu.:29.71   1st Qu.: 0.000
##  Median :24.03   Median : 75.88   Median :29.76   Median : 0.000
##  Mean   :23.65   Mean   : 76.31   Mean   :29.75   Mean   : 0.285
##  3rd Qu.:25.19   3rd Qu.: 83.25   3rd Qu.:29.81   3rd Qu.: 0.000
##  Max.   :28.61   Max.   :100.00   Max.   :31.14   Max.   :19.625
##   AVE. WND_SPD
##  Min.   : 1.962
##  1st Qu.: 4.808
##  Median : 6.000
##  Mean   : 6.347
##  3rd Qu.: 7.458
##  Max.   :25.923
```

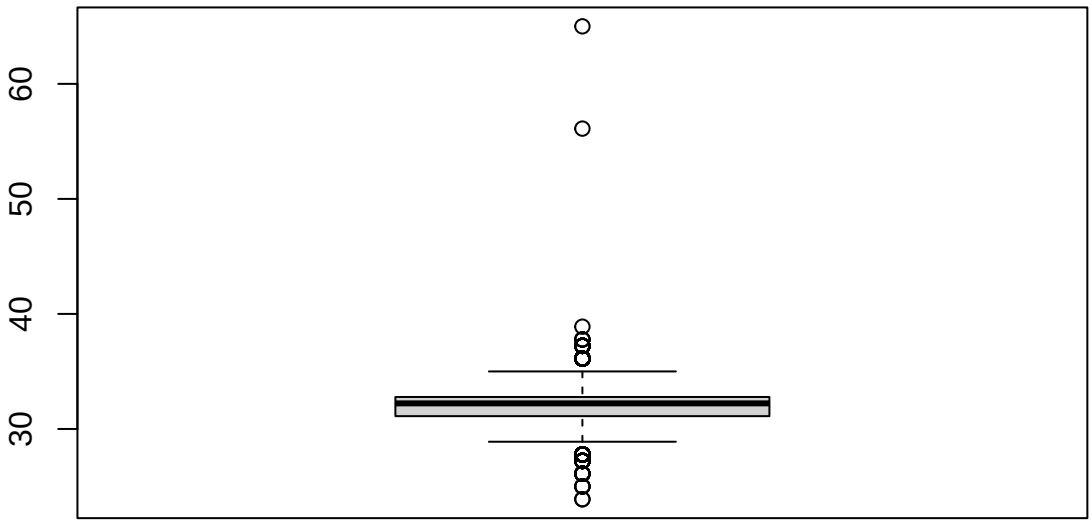
Visualizations

```
boxplot(merged_df$`MIN. TEMP`)
```

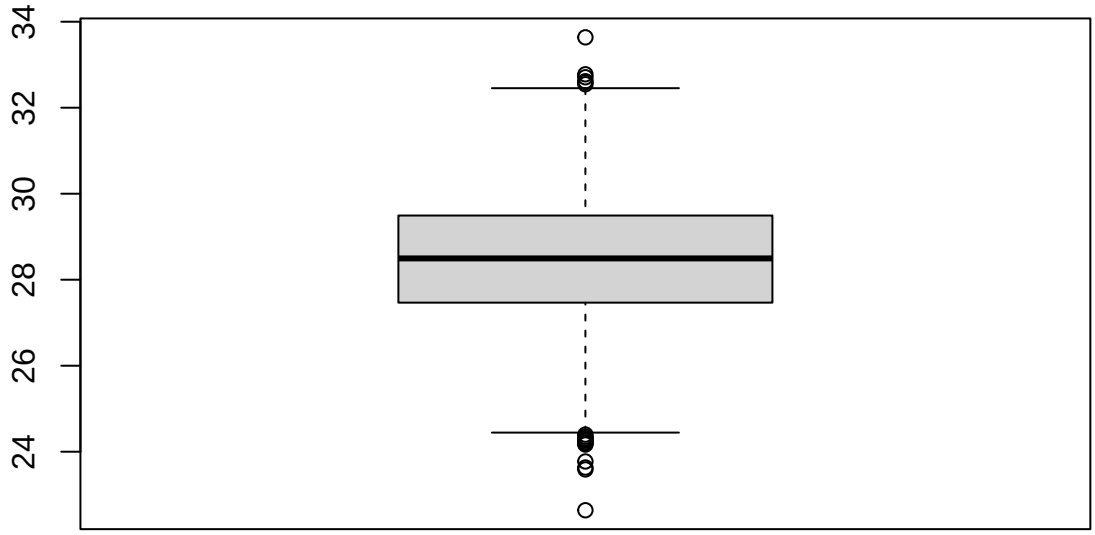


Box Plots

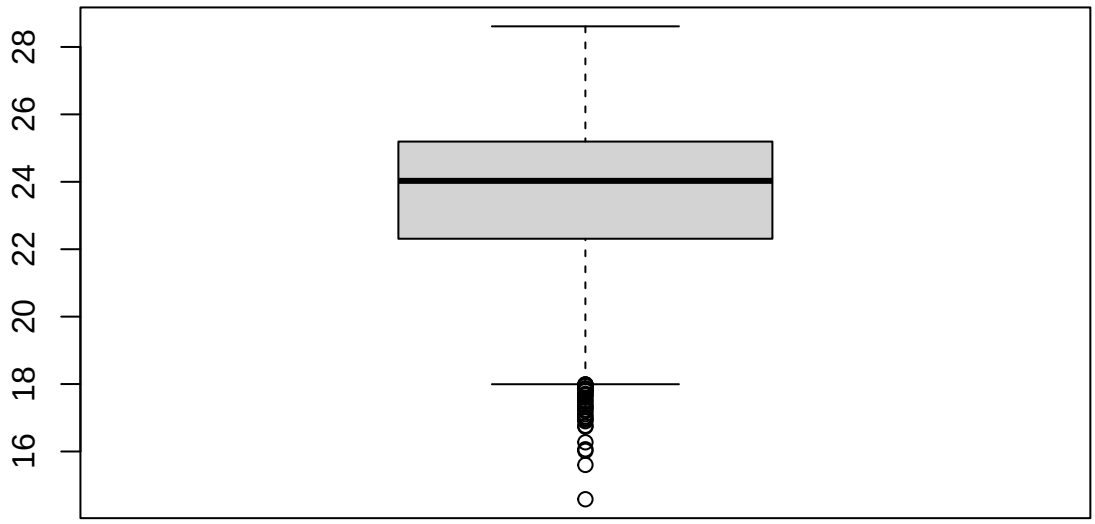
```
boxplot(merged_df$`MAX. TEMP`)
```



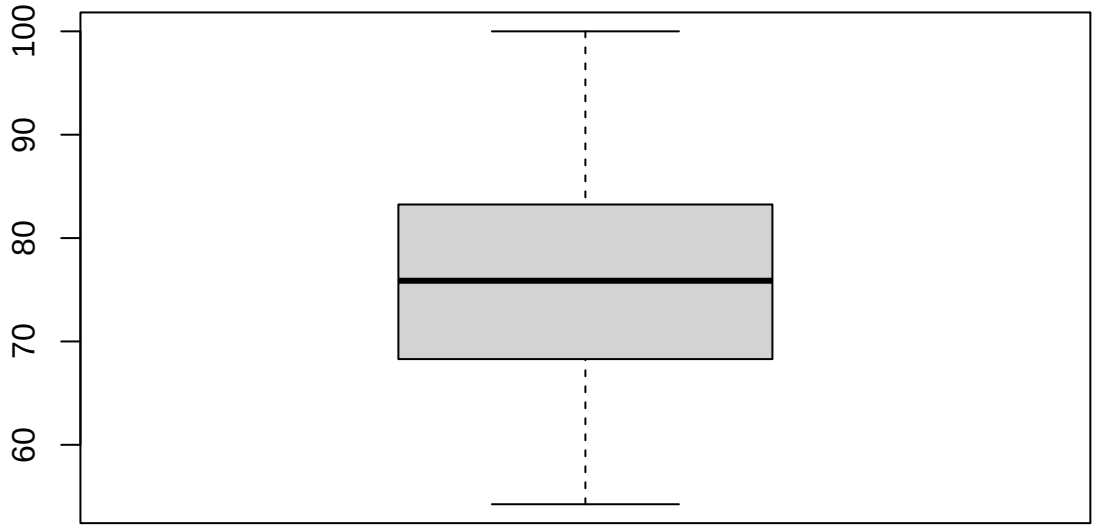

```
boxplot(merged_df$`AVE. TEMP`)
```



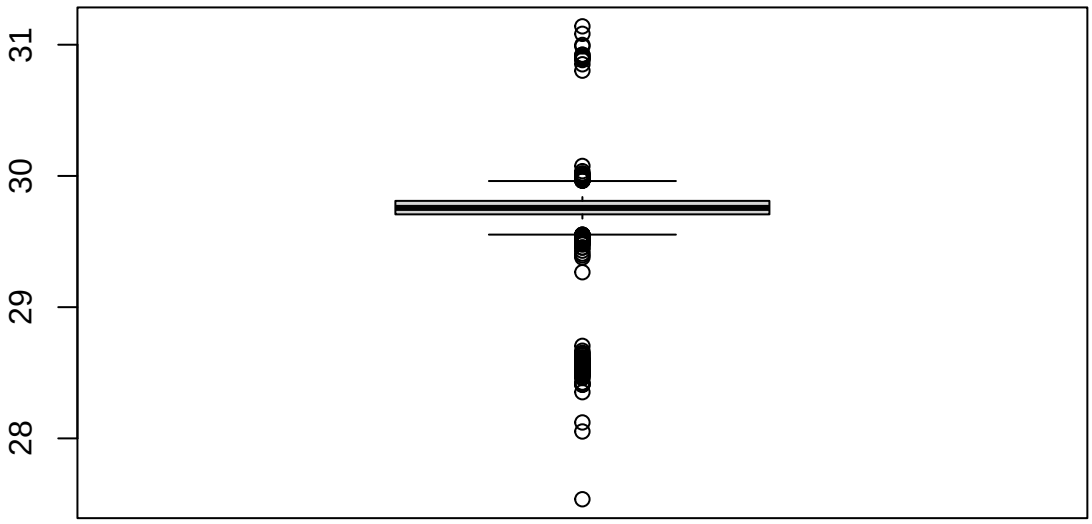

```
boxplot(merged_df$`AVE. DP`)
```



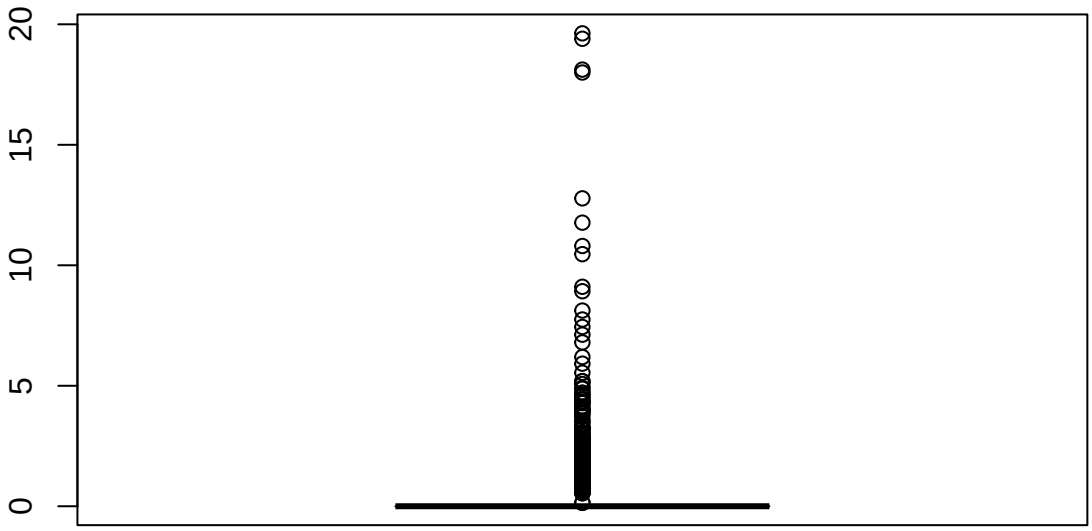

```
boxplot(merged_df$`AVE. HUM`)
```



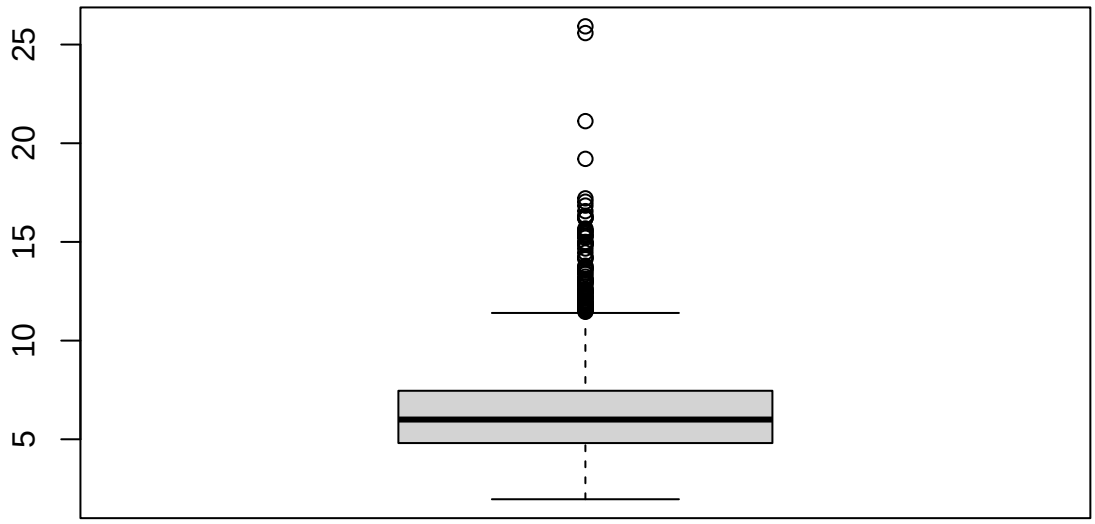

```
boxplot(merged_df$`AVE. PRES`)
```




```
boxplot(merged_df$`AVE. WND_GST`)
```




```
boxplot(merged_df$`AVE. WND_SPD`)
```

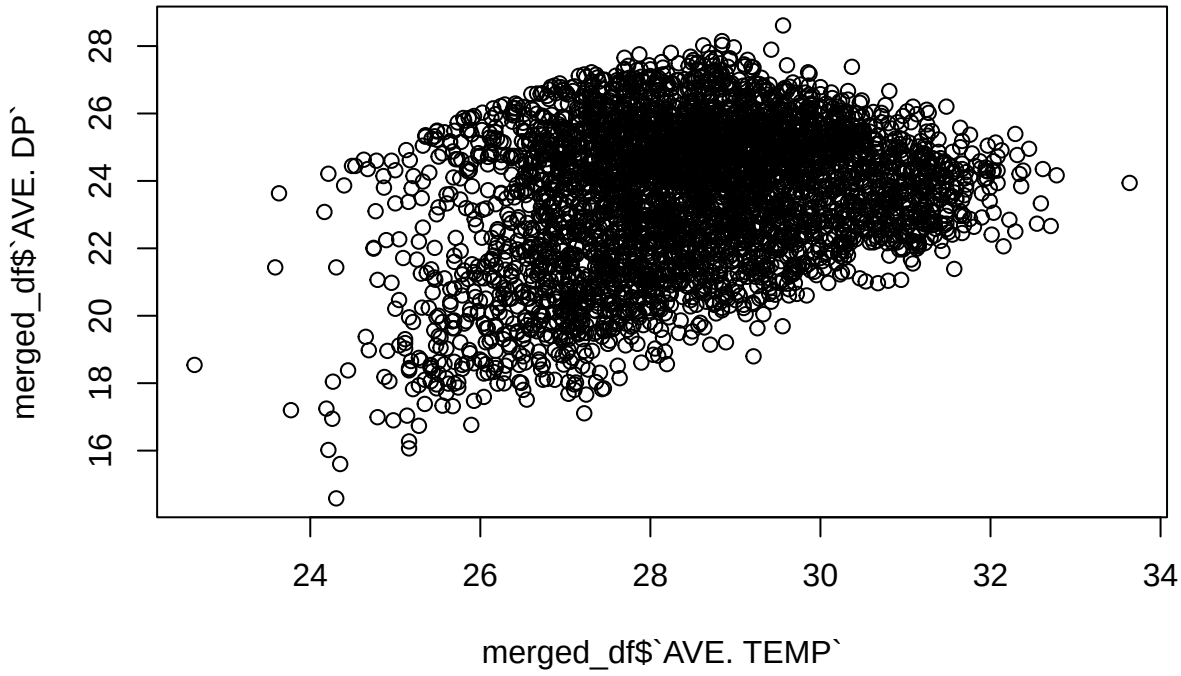


Each attribute of the merged data frame was displayed in the form of a box plot:

- **Minimum Temperature Box Plot:** This box plot shows that the minimum temperature recorded is ranged between 22 to 30 degrees, without the outliers. The common minimum temperature range however is in between 25 to 27 degrees with the median sitting at 26 degrees. This means that the usual lowest temperature experienced in Pasay throughout the years is around 25 to 27 degrees with the most common being at 26.
- **Maximum Temperature Box Plot:** The average maximum temperature recorded in Pasay is about in the range of 30 to 35 degrees. The highest minimum and maximum range, without considering the outliers, is between sub-30 to 39. There is however outliers which surpass this maximum in where the highest recorded temperature is above 60 degrees and the lowest being around 25 or lower.
- **Average Temperature Box Plot:** The average recorded temperature falls between 24 to 33 degrees with the vast majoprity of the records being between 27 to 29.5 degrees. Based on the given data, Pasay usually experiences 29 degrees of average temperature in the past years.
- **Average Dew Point:** The overall average dew point falls in to the range of 18 to 28 degrees with its outliers being below 18 degrees. The common dew point temperature value is in the 22 to 25 degrees range. The most common dew point is at 24 degrees.
- **Average Humidity:** Unlike the other box plots, the average humidity doesn't have recorded outliers. This means the maximum value is at 100 percent and the minimum being below around 50 percent. The vast majority of recorded humidity values is between 65 to 85 percent with the median being at 75 percent, which indicates a rather humid average environment over the years.
- **Average Pressure:** The average pressure box plot has a smaller interquartile range, falling in between 29.5 to 30.0 and with the rest of its values being outliers. This means that the common recorded pressure in Pasay over the past decade is at 29.5 to 29.7

- **Average Wind Gust:** In the given result in the box plot, it can be seen that the most common value of recorded wind gust is zero which makes the box plot being skewed close to zero. This indicates that Pasay rarely records any wind gust that is above 5 mph.
- **Average Wind Speed:** The overall average wind speed in Pasay is observed to be rather slow, evident in the shown box plot in where the recorded range is between 0 to 12 mph. The common recorded speed however is in the range ot 5 to 7 mph. There are however recorded cases where the wind speed is slightly above average which can be seen with the present outliers, with the highest being at 25 mph or above which can imply aberrant weather cases.

```
scat_temp_dp <-plot(merged_df$`AVE. TEMP`, merged_df$`AVE. DP`)
```



Scatter Plots

```
scat_temp_dp
```

```
## NULL
```

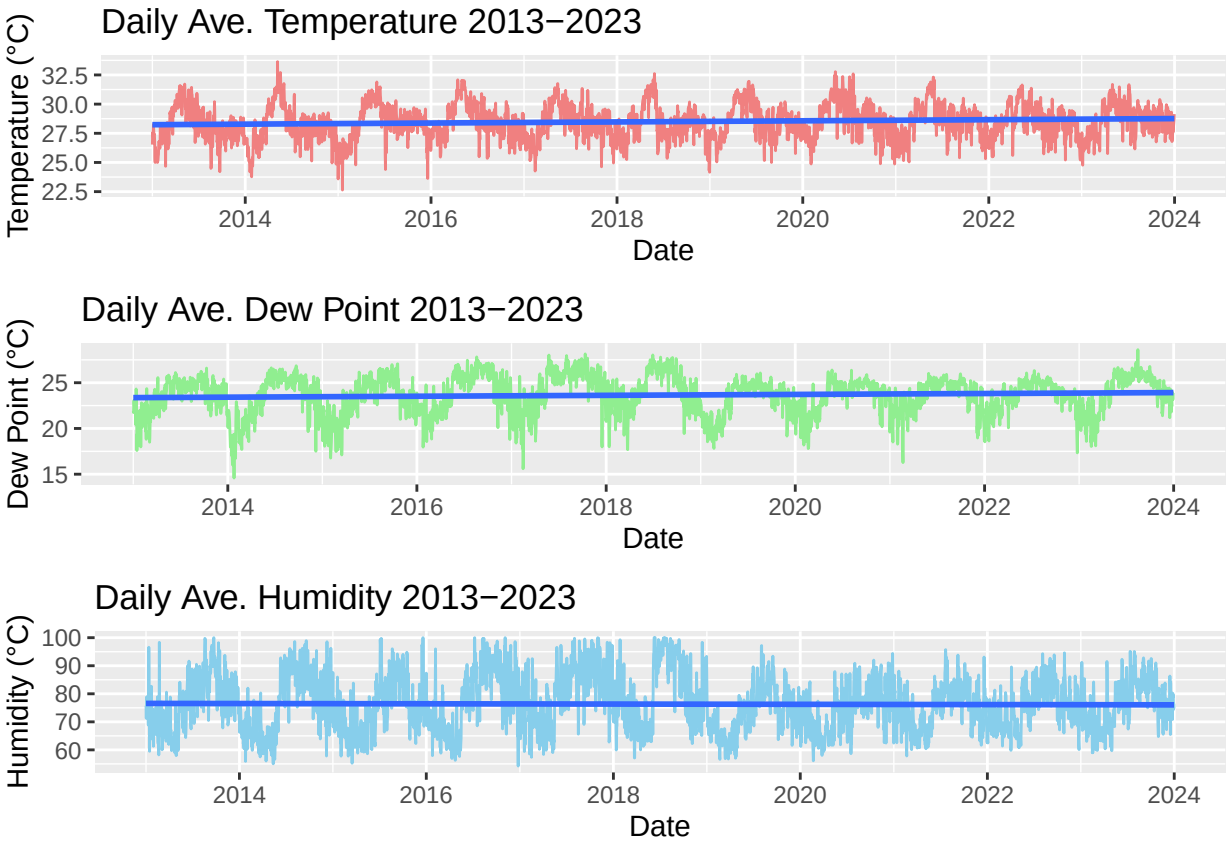
Based on the given scatter plot between the average dew point and average temperature, it can be observed that the two atrributes follows a positive correlation as the point values of the dew point increases as the average temperature increases. With this, it can be inferred that the temperature experienced in Pasay is affected by its dew point and vice versa. Upon looking more on the data, there is a vast cluster of points in the middle which makes up around the range of 27 to 29 degrees of temperature with a variation of dew points with it peaking at around 28 degrees.

```r
tsp_temp <- ggplot(merged_df, aes(x = DATE, y = `AVE. TEMP`)) +
  geom_line(color = "lightcoral") +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title = "Daily Ave. Temperature 2013-2023", x = "Date", y = "Temperature (°C)")
tsp_dp <- ggplot(merged_df, aes(x = DATE, y = `AVE. DP`)) +
  geom_line(color = "lightgreen") +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title = "Daily Ave. Dew Point 2013-2023", x = "Date", y = "Dew Point (°C)")
tsp_hum <- ggplot(merged_df, aes(x = DATE, y = `AVE. HUM`)) +
  geom_line(color = "skyblue") +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title = "Daily Ave. Humidity 2013-2023", x = "Date", y = "Humidity (°C)")


all_tsp <- grid.arrange(tsp_temp, tsp_dp, tsp_hum, nrow = 3)
```

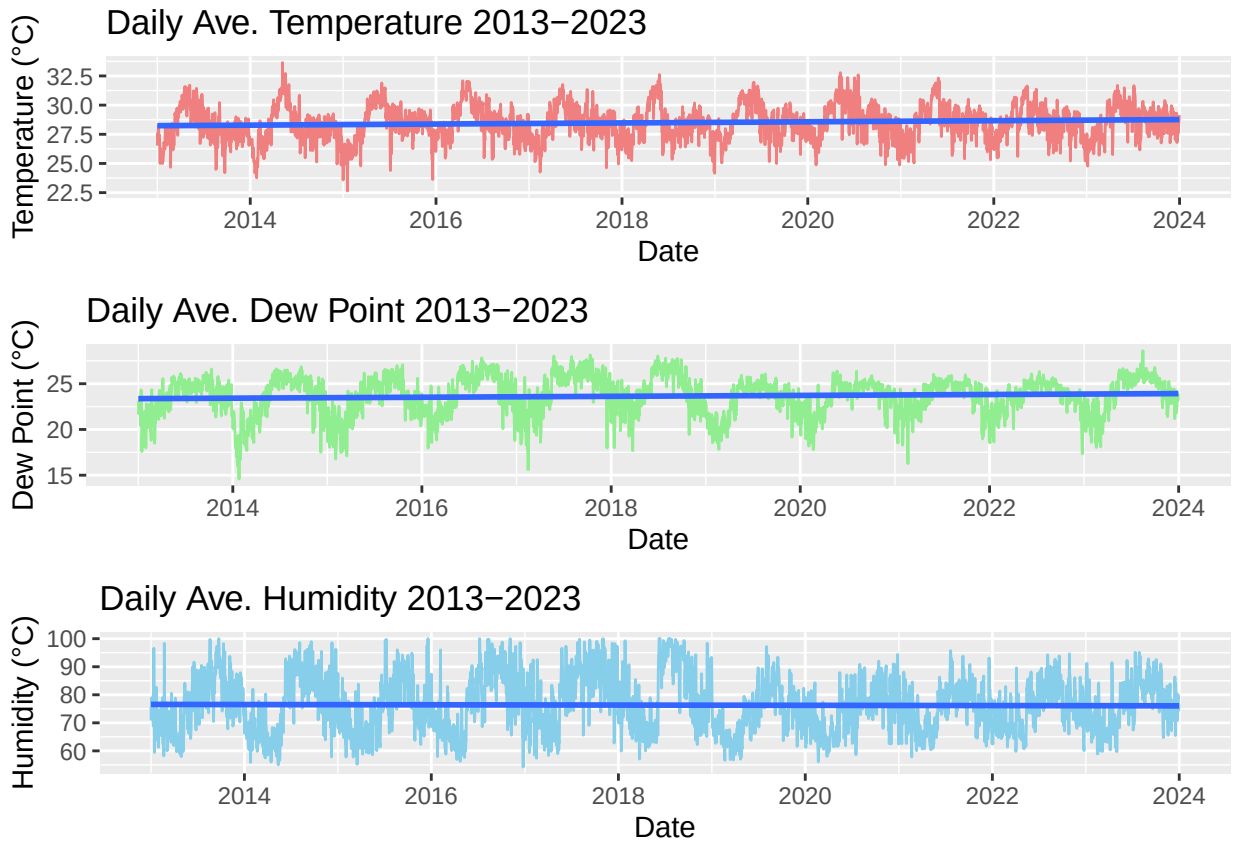
Time-Series Plots

```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```



```
all_tsp_plot <- as_ggplot(all_tsp)

all_tsp_plot
```



The following time series plots can be further analyzed:

- **Daily Average Temperature:** In the given data, it can be observed that the daily average temperature became less and less varied over the years with the most amount of variations can be seen during the 2014 to 2015 time period where it also peaked at around 32.5 and beyond and also has the recorded low point of 22.5 degrees. As the years go by, the average temperature can be seen to steady between 25 to 30 degrees with the most common daily temperature sitting at 27 to 27.5 degrees. It can also be noted that the following graph seems to follow a wave-like pattern which means the temperature changes based on seasons. This trend can be observed further when looking at each start of the year having a relatively low temperature and fluctuates as time goes by. Going into depth with the time-series plot of the daily average temperature, it can be observed that the time plot fluctuates regularly as the year changes and a norm that is seen is that the temperature for that year is usually is at its highest around the start to middle. This can signify a hotter season which is usually around March to May. Beyond that, the temperature then dips at its average temperature around 27.5 degrees and goes lower as the year ends. Here, it can also be seen the highest recorded average temperature, being at above 32.5 and the lowest being slightly above 22.5 which is all recorded during mid-2014 to early 2015.
- **Daily Average Dew Point:** The dew point has more a consistent trend over the years with there being a noticeable wave-like pattern. At the start of the year the dew point is usually is at its lowest point with the lowest recorded value being at 15 degrees which was in 2014. The highest recorded dew point was actually during 2023 to 2024 with the value going over 30 degrees. The lowest amount of

fluctuations seen in the dew point data can be observed during 2021 to 2023 with the average dew point being set around the median point while also having a relatively low peak and dip.

- **Daily Average Humidity:** It is observable in the past years that the average levels is much higher as compared to recent years. There is also a foreseeable pattern in where the humidity level is at its highest during the middle of the given year and usually has its lowest point at the start of each year. Around mid-2018 is where the humidity levels of Pasay peaked, around 100 degrees and the lowest being below 60 degrees during 2014. In the most recent years of 2021 to 2024 it can be observed that the humidity levels are less varied is comparatively has less fluctuations.

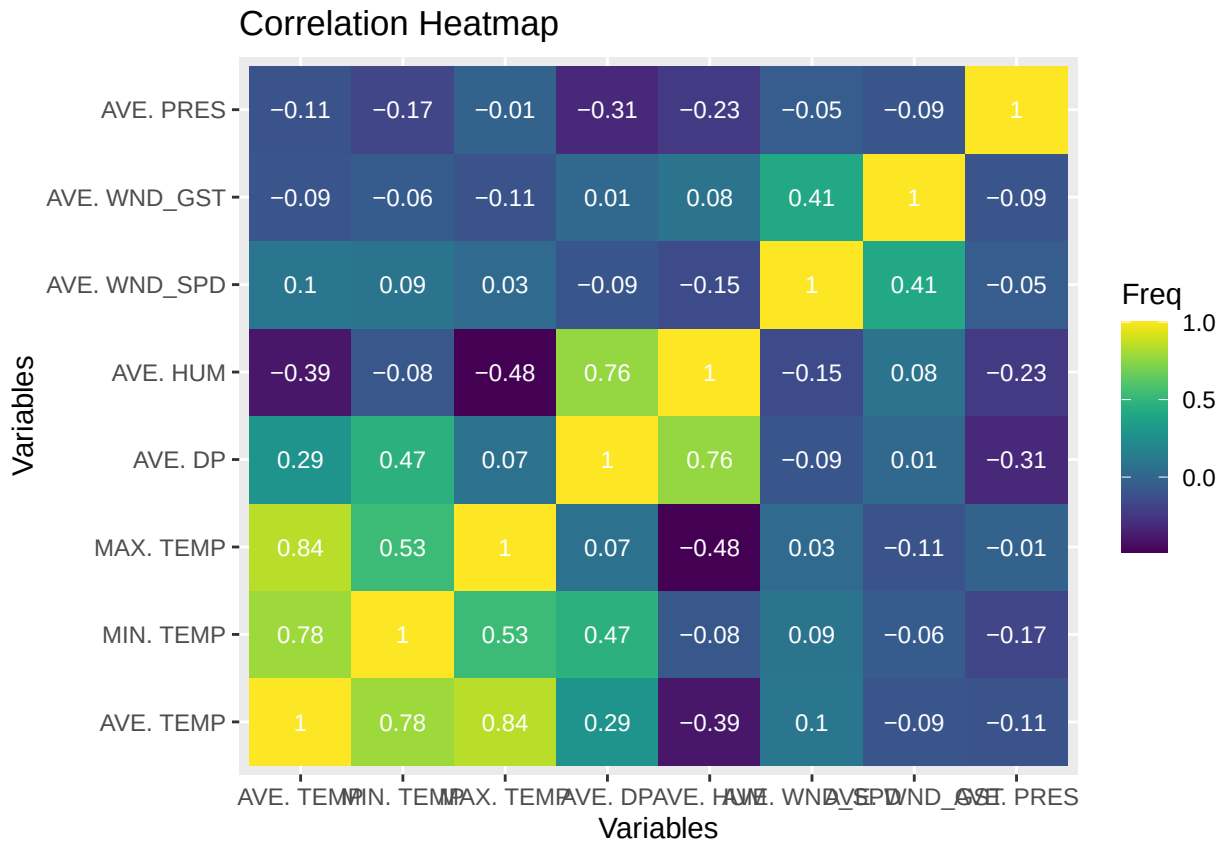
```
cor_matrix <- cor(merged_df[, c("AVE. TEMP", "MIN. TEMP", "MAX. TEMP", "AVE. DP", "AVE. HUM", "AVE. WND_
```

Correlation Analysis Let's visualize the correlations

```
cor_df <- as.data.frame(as.table(cor_matrix))

cor_heatmap <- ggplot(data = cor_df, aes(x = Var1, y = Var2, fill = Freq)) +
  geom_tile() +
  scale_fill_viridis(option = "viridis") +
  geom_text(aes(label = round(Freq, 2)), color = "white", size = 3) +
  labs(title = "Correlation Heatmap", x = "Variables", y = "Variables")

cor_heatmap
```



Upon looking at the heatmap we can the view various correlations each attribute has to each other. Based on this, it is expected that the minimum and maximum temperature has a correlation to the average temperature. There is also a positive correlation with the average dew point and temperature with frequency values being close to 0.5, which is backed up by the results of the scatter plot. Similar to this, the average humidity and dew point also have a positive correlation as the nature of both deal with the moisture. It is also noticeable that the average humidity and temperature has the darkest color, meaning they do not have any sort of correlation. This also applies to the average pressure with it being the next darkest. Among all of the attributes, the avearge pressure has the least amounts of correlations with the other present attributes as all of its values are negative. This implies that the average pressure is an independent factor and is not affected with other environmental attributes.

Exporting the dataset

```
write.csv(merged_df, "preprocessed_2013-2023.csv", row.names=FALSE)
```

Exporting the plots as images

```
ggsave("plots/heatmap_plot.png", plot = cor_heatmap, dpi = 300)
```

```
## Saving 6.5 x 4.5 in image
```

```
ggsave("plots/tsp_temp.png", plot = tsp_temp, dpi = 300)
```

```
## Saving 6.5 x 4.5 in image
## `geom_smooth()` using formula = 'y ~ x'
```

```
ggsave("plots/tsp_hum.png", plot = tsp_hum, dpi = 300)
```

```
## Saving 6.5 x 4.5 in image
## `geom_smooth()` using formula = 'y ~ x'
```

```
ggsave("plots/tsp_dp.png", plot = tsp_dp, dpi = 300)
```

```
## Saving 6.5 x 4.5 in image
## `geom_smooth()` using formula = 'y ~ x'
```

```
ggsave("plots/sp_temp_dp.png", plot = scat_temp_dp, dpi = 300)
```

```
## Saving 6.5 x 4.5 in image
```

```
ggsave("plots/all_tsp.png", plot = all_tsp_plot, dpi = 300)
```

```
## Saving 6.5 x 4.5 in image
```