PM566-Lab5

AUTHOR
Breyonne Williams

Setup 1

```
setwd("~/Documents")
dir.create("PM566-Lab5")
```

```
Warning in dir.create("PM566-Lab5"): 'PM566-Lab5' already exists
```

```
setwd("PM566-Lab5")

download.file(
  "https://raw.githubusercontent.com/USCbiostats/PM566/master/website/content/assignment
  destfile = "README.Rmd"
  )
```

Setup 2

```
library(data.table)
download.file(
  "https://raw.githubusercontent.com/USCbiostats/data-science-data/master/02_met/met_all
  destfile = "met_all.gz",
  method   = "libcurl",
  timeout  = 60
)
met <- data.table::fread("met_all.gz")

stations <- data.table::fread("ftp://ftp.ncdc.noaa.gov/pub/data/noaa/isd-history.csv")
stations[, USAF := as.integer(USAF)]
```

```
Warning in eval(jsub, SDenv, parent.frame()): NAs introduced by coercion
```

```
stations[, USAF   := fifelse(USAF == 999999, NA_integer_, USAF)]
stations[, CTRY   := fifelse(CTRY == "", NA_character_, CTRY)]
stations[, STATE  := fifelse(STATE == "", NA_character_, STATE)]
stations <- unique(stations[, list(USAF, CTRY, STATE)])
stations <- stations[!is.na(USAF)]
stations[, n := 1:.N, by = .(USAF)]
stations <- stations[n == 1,][, n := NULL]
```

Setup 3

```
combined_data <- merge(
  x = met,
  y = stations,
  by.x = "USAFID",
  by.y = "USAF",
  all.x = TRUE,
  all.y = FALSE
)

stations[, n := 1:.N, by = .(USAF)]
stations <- stations[n == 1,][, n := NULL]

dat <- merge(
  x = met,
  y = stations,
  by.x = "USAFID",
  by.y = "USAF",
  all.x = TRUE,
  all.y = FALSE
)

head(dat[, list(USAFID, WBAN, STATE), n = NULL])
```

```
   USAFID  WBAN STATE
1: 690150 93121    CA
2: 690150 93121    CA
3: 690150 93121    CA
4: 690150 93121    CA
5: 690150 93121    CA
6: 690150 93121    CA
```

Question 1

```
library(magrittr)
dat[, .(
  temp_med = median(temp, na.rm = TRUE),
  wind.sp_med = median(wind.sp, na.rm = TRUE),
  atm.press_med = median(atm.press, na.rm = TRUE)
  ),
  by = STATE
  ][order(STATE)] %>% head(n = 4)
```

```
   STATE temp_med wind.sp_med atm.press_med
1:    AL     25.3         1.5        1014.8
2:    AR     25.6         2.1        1014.5
3:    AZ     29.0         3.1        1010.8
4:    CA     21.1         2.6        1012.8
```

```
median_temp <- quantile(dat$temp, probs = 0.5, na.rm = TRUE)
median_wind.sp <- quantile(dat$wind.sp, probs = 0.5, na.rm = TRUE)
median_atm.press <- quantile(dat$atm.press, probs = 0.5, na.rm = TRUE)

closest_temp_station <- dat[which.min(abs(dat$temp - median_temp)), ]
closest_wind.sp_station <- dat[which.min(abs(dat$wind.sp - median_wind.sp)), ]
closest_atm.press_station <- dat[which.min(abs(dat$atm.press - median_atm.press)), ]

percentiles_temp <- quantile(dat$temp, probs = c(0.25, 0.5, 0.75), na.rm = TRUE)
percentiles_wind.sp <- quantile(dat$wind.sp, probs = c(0.25, 0.5, 0.75), na.rm = TRUE)
percentiles_atm.press <- quantile(dat$atm.press, probs = c(0.25, 0.5, 0.75), na.rm = TRU

representative_temp_stations <- lapply(percentiles_temp, function(p) {
  dat[which.min(abs(dat$temp - p)), ]
})
representative_wind.sp_stations <- lapply(percentiles_wind.sp, function(p) {
  dat[which.min(abs(dat$wind.sp - p)), ]
})
representative_atm.press_stations <- lapply(percentiles_atm.press, function(p) {
  dat[which.min(abs(dat$atm.press - p)), ]
})

# top 3 median stations are located in CA, MI, & AR. the median for all three variables
```

Question 2

```
library(geosphere)
```

```
The legacy packages maptools, rgdal, and rgeos, underpinning the sp package,
which was just loaded, will retire in October 2023.
Please refer to R-spatial evolution reports for details, especially
https://r-spatial.org/r/2023/05/15/evolution4.html.
It may be desirable to make the sf package available;
package maintainers should consider adding sf to Suggests:.
The sp package is now running under evolution status 2
     (status 2 uses the sf package in place of rgdal)
```

```
variables_matrix <- dat[, c("temp", "wind.sp", "atm.press")]


median_distance_index <- which.min(apply(variables_matrix, 1, median))
median_distance_stations <- dat[median_distance_index, ]
if (length(median_distance_index) > 1) {
  lowest_latitude_station <- median_distance_stations[which.min(median_distance_stations
  median_distance_stations <- lowest_latitude_station
}

#CA is the most representative
```

Question 3

```
library(leaflet)
library(dplyr)
```

```

Attaching package: 'dplyr'

The following objects are masked from 'package:data.table':

    between, first, last

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

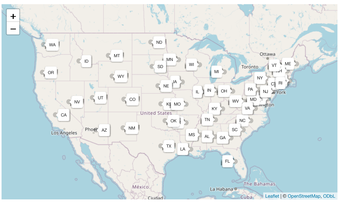
```
library(geosphere)

find_closest_station <- function(dat, midpoint) {
  distances <- apply(dat[, c("lat", "lon")], 1, function(row) {
    distM(midpoint, row)
  })
  closest_station_index <- which.min(distances)
  return(dat[closest_station_index, ])
}

state_midpoints <- dat %>%
  group_by(STATE) %>%
  summarise(
    Midpoint_Latitude = mean(lat),
    Midpoint_Longitude = mean(lon)
  ) %>%
  ungroup()

map <- leaflet() %>%
  addTiles()

map <- map %>%
  addCircleMarkers(
    data = state_midpoints,
    lng = ~state_midpoints$Midpoint_Longitude,
    lat = ~state_midpoints$Midpoint_Latitude,
    color = ifelse(state_midpoints$STATE, "red", "blue"),
    radius = 6,
    label = ~state_midpoints$STATE,
    labelOptions = labelOptions(noHide = TRUE)
  )
map
```



Question 4

```
dat <- data.frame(
  State = c("CA", "CA", "NY", "NY", "TX", "TX"),
  Temperature = c(10, 22, 23, 28, 19, 27),
  WindSpeed = c(10, 8, 7, 12, 9, 11),
  Pressure = c(1013, 1012, 1010, 1008, 1015, 1011)
)

dat$Avg_Temperature_Level <- cut(
  dat$Temperature,
  breaks = c(-Inf, 20, 25, Inf),
  labels = c("Low", "Mid", "High"),
  right = FALSE
)

summary_table <- dat %>%
  group_by(Avg_Temperature_Level) %>%
  summarise(
    Number_of_Entries = n(),
    Number_of_NA_Entries = sum(is.na(Temperature)),
    Number_of_Stations = n_distinct(State),
    Number_of_States_Included = n_distinct(State),
    Mean_Temperature = mean(Temperature, na.rm = TRUE),
    Mean_WindSpeed = mean(WindSpeed, na.rm = TRUE),
    Mean_Pressure = mean(Pressure, na.rm = TRUE)
  )

print(summary_table)
```

```
# A tibble: 3 × 8
  Avg_Temperature_Level Number_of_Entries Number_of_NA_Entries
  <fct>                            <int>                 <int>
1 Low                                  2                     0
2 Mid                                  2                     0
3 High                                 2                     0
# ℹ 5 more variables: Number_of_Stations <int>,
#   Number_of_States_Included <int>, Mean_Temperature <dbl>,
#   Mean_WindSpeed <dbl>, Mean_Pressure <dbl>
```