Data Documentation Example

Linda Kim

Note:

This document records the process of collecting air quality information for data reproducibility. See our shared folders for API keys and passwords and data to run this code.

Data Collection

Objective

Collecting air quality data for criteria pollutants: PM2.5, PM10, NO2, O3 over a ten-year period (2011-2020). Retrieved data from US EPA Air Quality System datamart (RAQSAPI package). All air quality data for BREATHE study was last collected 04/28/2022.

Libraries required

```
#libraries required
library(RAQSAPI)
library(tidyverse)
library(readxl)
library(keyring)
library(openxlsx)
library(vroom)
library(tidygeocoder)
```

Setting up keyrings

```
#### Setting up keys ####
# set up keys and use password. See RAQSAPI help pages on how to set up account.
keyring::key_set(service = "AQSDatamart", username = "enail.com")
```

```
datamartAPI_user <- "email.com"
server <- "AQSDatamart"

aqs_credentials(username = datamartAPI_user,
                key = key_get(service = server, username = datamartAPI_user ))
```

Reading in data

Total 16 tracks identified, and 4 were duplicates due to name changes over the years. The 4 duplicated tracks were merged below.

```
#### Reading in CA data ####

address1 <- read_excel("ca_address.xlsx")

##### racetrack addresses with latitude and longitude #####

geocodes <- tidygeocoder::geocode(address1, street = street, city = city, state = state, postalcode = zip)
geocodes %>% colnames()

#Changed location #6: LA to match previous analysis.
geocodes$lat[6] <- 40.58581512 # from google.
geocodes$long[6] <- -124.2658941 # from google.
geocodes

address<- geocodes %>%
  mutate(trk = ifelse(trk == "BHP", "HOL", trk),
         trk = ifelse(trk == "BSR", "FPX", trk),
         trk = ifelse(trk == "LRC", "LA", trk),
         trk = ifelse(trk == "OTP", "PLN", trk)

address
```

Calculating perimeter

Assign latitudes and longitudes to create square perimeter around each racetracks. AQS mart require square perimeter (box) to download data. To account for the roundness of the earth use this equation.

```
##### Calculate distance to use AQS by box #####
```

```
#length of longitude needs to account for roundness of earth.
#use (cos(lat/57.2958)*69172)

##### calculate distance to use aqs by box #####
vdistTrk <-32 # miles approx 50 km

distinct_address <- address %>%
  mutate( minlat =lat - vdistTrk/69,
          maxlat = lat + vdistTrk/69,
          lenLon = vdistTrk /(cos(lat/57.2958)*69.172),
          minlong = long -lenLon,
          maxlong = long + lenLon) %>%
  rename(latitude = lat,
         longitude = long)
```

Retrieving Data from AQS

This loop retrieves air pollution data (PM2.5, NO2, PM10, O3) for 12 years. Monitors within 50 km of the respective racetracks and with at least 30% of data available will return data. AQS mart allows only one year of data to be retrieved at a time. Loop is written to retrieve one year at a time as AQS mart requested and bound together later. This process takes about 8 hours. *WARNING: Some racetracks will have no monitors because of our parameters (within 50 km and 30% of data available.

```
#### by box ####
#parameter numbers are in this order: PM2.5, NO2, PM10, O3
parameters <- c("88101", "42602", "81102", "44201")
startYear <- "20100101"
endYear <- "20101231"
startDate <- as.Date("20100101", format="%Y%m%d")
endDate <- as.Date("20211231", format="%Y%m%d")
numYears <- 12

# return monitors that have 30% data available.
dateRaceMonitorOverlapThreshold <- 0.30

addressMonitors <- list()
startTime <- Sys.time()

print(paste0("Start Time: ", startTime))
for(indexParameter in 1:length(parameters))
```

```r
{
  print(paste0("Parameter ", indexParameter, ": ", parameters[indexParameter]))
  addressMonitors[[indexParameter]] <- list()
  for(indexAddress in 1:nrow(distinct_address))
  {
    print(paste0("Address # ", indexAddress))
    vParameter <- parameters[indexParameter]
    vMinlat <- distinct_address$'minlat'[indexAddress]
    vMaxlat <- distinct_address$'maxlat'[indexAddress]
    vMinlon <- distinct_address$'minlong'[indexAddress]
    vMaxlon <- distinct_address$'maxlong'[indexAddress]

    box <- aqs_monitors_by_box(parameter=vParameter,
                               bdate=startDate,
                               edate=endDate,
                               minlat=vMinlat,
                               maxlat=vMaxlat,
                               minlon=vMinlon,
                               maxlon=vMaxlon)

   refbox <- box %>%
          mutate(open_date = as.Date(open_date),
          close_date = as.Date(close_date)) %>%
          filter(open_date<startDate & (close_date>endDate| is.na(close_date)))

    refbox$orglat <- distinct_address$latitude[indexAddress]
    refbox$orglon <- distinct_address$longitude[indexAddress]

# if no monitors were found within 50 km and without 30% data available.
# refbox files provide information about the monitor.
    if(nrow(refbox) == 0)
    {
      print("Shouldn't be here! AHHH!")
      next
    }
    if(nrow(refbox) > 1)
    {
      for( i in 1:nrow(refbox))
```

```
 {
  refbox$dist[i] <- distm(c(refbox$orglon[1], refbox$orglat[1]),
                          c(refbox$longitude[i], refbox$latitude[i]),
                          fun = distHaversine)
 }

 refbox <- refbox %>%
      arrange(dist)
}
addressMonitors[[indexParameter]][[indexAddress]] <- list()
refbox$dateRaceMonitorOverlap <- NA
for(indexMonitor in 1: nrow(refbox))
{
print(paste0("Monitor # ", indexMonitor))
 addressMonitors[[indexParameter]][[indexAddress]][[indexMonitor]] <- data.frame()
 for(indexYear in 1:numYears)
 {
  vStartDate <- as.Date(startYear, format="%Y%m%d") + years(indexYear-1)
  vEndDate <- as.Date(endYear, format="%Y%m%d") + years(indexYear-1)
  print(vStartDate)
  addressMonitors[[indexParameter]][[indexAddress]][[indexMonitor]] <- rbind(
   addressMonitors[[indexParameter]][[indexAddress]][[indexMonitor]],
   aqs_dailysummary_by_site(parameter = vParameter,
           bdate = vStartDate,
           edate = vEndDate,
           stateFIPS = refbox$state_code[indexMonitor],
           countycode = refbox$county_code[indexMonitor],
           sitenum = refbox$site_number[indexMonitor]))
 }

 dateMonitor <- unique(ymd(addressMonitors[[indexParameter]][[indexAddress]][[indexMonitor]]$date_local))
 dateRace <- ymd(trk_dates$data[[indexAddress]])
 dateRaceMonitorOverlap <- ((sum(dateMonitor%in%dateRace))/length(dateRace))
 refbox$dateRaceMonitorOverlap[indexMonitor] <- dateRaceMonitorOverlap
 print(paste0("Overlap: ", dateRaceMonitorOverlap))
 if(dateRaceMonitorOverlap > dateRaceMonitorOverlapThreshold)
 {
  print("Threshold Reached")
```

```
        write.xlsx(refbox, file=paste0("Parameter", indexParameter, "_Address", indexAddress, "_Refbox.xlsx"))
        break
      }
      if(indexMonitor == nrow(refbox))
      {
        print("Last Monitor Reached, No Sufficient Overlap Found")
        write.xlsx(refbox, file=paste0("Parameter", indexParameter, "_Address", indexAddress, "_Refbox.xlsx"))
        write.xlsx(refbox, file=paste0("Parameter", indexParameter, "_Address", indexAddress, "_AHHHHHHHHH.xlsx"))
      }
    }
  }
}
endTime <- Sys.time()
print(paste0("End Time: ", endTime))
print(endTime - startTime)

#### Exporting excel with naming convention Parameter#_Address#_Monitor# #####
for(i in seq_along(addressMonitors))
{
  for(j in seq_along(addressMonitors[[i]]))
  {
    for(k in seq_along(addressMonitors[[i]][[j]]))
    {
      write.xlsx(addressMonitors[[i]][[j]][[k]], file = paste0("Parameter", i, "_Address", j, "_Monitor", k, ".xlsx"))
    }
  }
}
```