Geospatial Code Replication Workflow

Jyotishka Datta

6/12/2019

Contents

Package Dependencies

```
# install.packages("devtools")
# devtools::install_github("thomasp85/patchwork")

library("sf")             # Spatial data objects and methods
library("spatstat")       # KDE and other spatial functions
library("raster")         # cell-based spatial operations
library("tidyverse")      # data manipulation framework
library("lubridate")      # Power tools for handling dates
library("tidycensus")     # Get census data
library("lwgeom")
library("Hmisc")
library("hrbrthemes")
library("gridExtra")
library("patchwork")
library("spdep")          # KNN functions
library("foreach")        # loops in parallel
library("doParallel")     # parallel backend
library("corrplot")       # correlation plot
library("ranger")         # randomforest implimentation
library("glmnet")         # for Ridge and Lasso Regression
library("knitr")          # for kable table
library("kableExtra")     # nake nice tables in Rmarkdown
library("FNN")            # KNN for CPS vs. NN plots
library("groupdata2")
library("htmltools")
library("viridis")        # color palette
library("viridisLite")    # color palette
```

Report Appendix 1: Data wrangling

Step 1. Detect all xls/xlsx and csv files in a directory and read them into a list

This Code example of data import function. The inputs are `.xls`, `.xlsx`, and `.csv` files in a folder. The output is a list data type where each element of the list if one of the input data sets in the sf spatial data format.

```
# requires all data in *.csv or *.xls files containing coordinate field names "X" and "Y"
# `crs` in the call to `st_as_sf()` needs to be set to the ESPG code of your data projection
# `base_dir` file path and many feature names are specified for the current project.

##1.1 Global Variables
# mapviewOptions(basemaps = c("Stamen.TonerLite", "OpenStreetMap.DE"))
base_dir = "C:/Users/jd033/Box/Child Maltreatment"
```

```
##2.1 Load Data
files <-list.files(file.path(base_dir,"/Little Rock Data/BuiltEnvironment/Factors2015"), pattern = "*\\
var_list <- vector(mode = "list")
var_names <- NULL
for(i in seq_along(files)){
  filename <- str_sub(files[i], start = 1, end = -5)
  sf_i <- tryCatch({
    if(tools::file_ext(files[i]) == "xlsx"){
      dat <- readxl::read_xlsx(file.path(base_dir,"/Little Rock Data/BuiltEnvironment/Factors2015",files
    } else if(tools::file_ext(files[i]) == "csv"){
      dat <- read.csv(file.path(base_dir,"/Little Rock Data/BuiltEnvironment/Factors2015",files[i]))
    }
    dat %>%
      filter(!is.na(X) | !is.na(Y)) %>%
      st_as_sf(., coords = c("X", "Y"), crs = 2765)
  }, error = function(e){
    cat(filename, "error = ",e$message,"\n")
    return(e)
  }
  )
  if(!inherits(sf_i, "error")){
    var_list[[length(var_list)+1]] <- sf_i
    var_names[length(var_list)] <- filename
  }
}
(names(var_list) <- var_names)
```

```
## [1] "Banks."                    "BarberAndBeautyShops."
## [3] "HighSchoolsPublic."        "HotelMotel."
## [5] "LiquorStores."             "MajorDeptRetailDiscount."
## [7] "Rental_MobileHomes."       "Rental_SingleToQuad."
## [9] "TattooPiercing."
```

Next steps

- Step 2: Import spatial neighborhood data with `read_sf()` and `get_decennial()`
- Step 3: Create spatial fishnet grid with `st_make_grid()` and calculate spatial weights with `poly2nb()` and `nb2listw()`
- Step 4: Intersect fishnet and census blocks to create populations estimates and weights per fishnet cell