San Francisco Crime Density Map (2003-2018)

Pauline I. Alvarado

12/19/2018

This coding exercise was submitted for the University of Pennsylvania's Criminal Justice Data Science Course taught by Dr. Gregory Ridgeway. Code has been modified from course examples to fit the needs of dataset and description of process. Data consists of San Francisco Police Department's Incident Reports from 2003 to May 2018 taken from the city of San Francisco's Open Data site. There are 2.21 million incidents included in this .csv file.

Key skills: Data Preprocessing, Data Wrangling, Creating SQLite Databases, SQL Queries, Regular Expressions, Density/Hot Spot Mapping

Data Preprocessing and Wrangling

Load packages and view the first 5 rows of the data set.

```
library(sqldf)
library(ggmap)
scan("sfpd-incident-2003to2018.csv", what="", nlines=5, sep="\n")
```

```
## [1] "IncidntNum;Category;Descript;DayOfWeek;Date;Time;PdDistrict;Resolution;Address;X;Y"
## [2] "150060275;NON-CRIMINAL;LOST PROPERTY;Monday;01/19/2015;14:00;MISSION;NONE;18TH ST / VALENCIA ST
## [3] "150098210;ROBBERY;\"ROBBERY, BODILY FORCE\";Sunday;02/01/2015;15:45;TENDERLOIN;NONE;300 Block o
## [4] "150098210;ASSAULT;AGGRAVATED ASSAULT WITH BODILY FORCE;Sunday;02/01/2015;15:45;TENDERLOIN;NONE;
## [5] "150098210;SECONDARY CODES;DOMESTIC VIOLENCE;Sunday;02/01/2015;15:45;TENDERLOIN;NONE;300 Block o
```

Set up a "while loop" to ensure that commas and semicolons are correctly used in addition to removing periods in field names.

```
# Create input and output files
  infile  <- file("sfpd-incident-2003to2018.csv",       'r')
  outfile <- file("sfpd-incident-2003to2018-clean.csv", 'w')

# Get variable names & make the symbols SQL friendly
  a <- readLines(infile, n = 1)
  a <- gsub(",", ";", a)

  writeLines(a, con = outfile)
  flush(outfile)
```

```
  # Loop through the rest of the data, 100,000 rows at a time
    c.lines <- 0

    system.time(
        while ((length(a <- readLines(infile, n=100000)) >  0))
        {
            c.lines <- c.lines + length(a)
            print(c.lines)

        # Turn commas outside double quotes to semicolons and use ?= to "lookahead" for paired quotes
        a <- gsub("(,)(?=(?:[^\"]|\"[^\"]*\")*$)",";", a, perl = TRUE)

        # Manually handle problematic lines
        a <- gsub("AEROSOL CONTAINER; SALE, PURCHASE OR POSSESSION OF",
                  "AEROSOL CONTAINER, SALE, PURCHASE OR POSSESSION OF", a)
        a <- gsub("DOG, FIGHTING;  OWNING, FIGHTING, OR ATTENDING FIGHT",
                  "DOG, FIGHTING, OWNING, FIGHTING, OR ATTENDING FIGHT", a)

        writeLines(a, con=outfile)
        }
    )
```

```
## [1] 1e+05
## [1] 2e+05
## [1] 3e+05
## [1] 4e+05
## [1] 5e+05
## [1] 6e+05
## [1] 7e+05
## [1] 8e+05
## [1] 9e+05
## [1] 1e+06
## [1] 1100000
## [1] 1200000
## [1] 1300000
## [1] 1400000
## [1] 1500000
## [1] 1600000
## [1] 1700000
## [1] 1800000
## [1] 1900000
## [1] 2e+06
## [1] 2100000
## [1] 2200000
## [1] 2215024
```

```
##    user  system elapsed
##  38.542   2.307  41.248
```

```
    close(infile)
    close(outfile)
```

Set Up SQL Database

```
con <- dbConnect(SQLite(), dbname="sfcrime.db")
a <- read.table("sfpd-incident-2003to2018-clean.csv", sep = ";", nrows = 5, header = TRUE)
variabletypes <- dbDataType(con, a)
```

Import the cleaned data file into RSQLite

```
if(dbExistsTable(con, "crime")) dbRemoveTable(con, "crime")

dbWriteTable(con, "crime",
             "sfpd-incident-2003to2018-clean.csv",
             row.names = FALSE,
             header = TRUE,
             field.types = variabletypes,
             sep = ";")
```

Check if the new "crime" table exists and view columns

```
dbListFields(con, "crime")
```

```
##  [1] "IncidntNum" "Category"   "Descript"   "DayOfWeek"  "Date"
##  [6] "Time"       "PdDistrict" "Resolution" "Address"    "X"
## [11] "Y"
```

Disconnect to finalize

```
dbDisconnect(con)
```

Create a Crime Hot Spot / Density Map

Connect to the SQL database and examine min/max coordinates in the dataset

```
con <- dbConnect(SQLite(), dbname="sfcrime.db")

res <- dbSendQuery(con, "
                        SELECT MIN(X)  AS min_lat,
                               MAX(X)  AS max_lat,
                               MIN(Y) AS min_lon,
                               MAX(Y) AS max_lon
                         FROM  crime
                         WHERE (X IS NOT NULL) AND
                               (Y IS NOT NULL)")
fetch(res, n = -1)
```

```
##     min_lat max_lat  min_lon max_lon
## 1 -122.5136  -120.5 37.70788      90
```

```
  dbClearResult(res)
```

Remove data outside of city limits

```
res <- dbSendQuery(con, "
                   UPDATE crime SET Y=NULL
                   WHERE Y > 37.9")
dbClearResult(res)
```

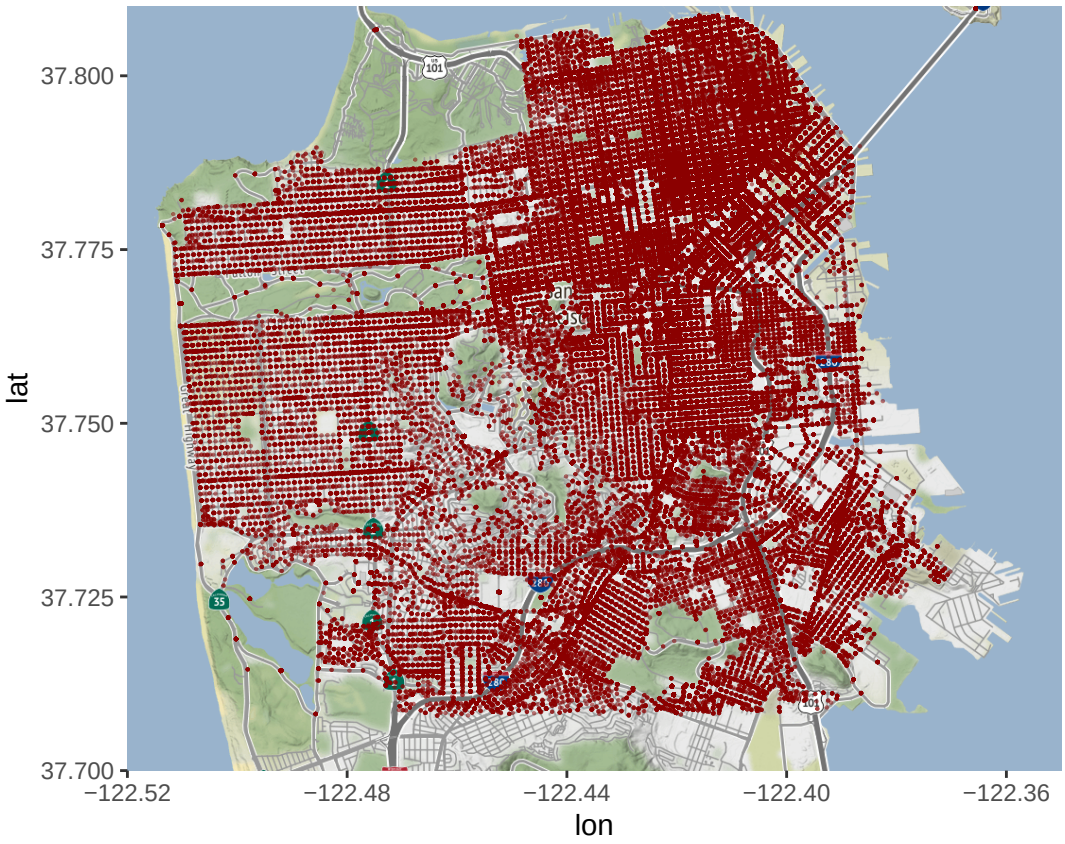
Create a SQL Query to get data for mapping

```
  res <- dbSendQuery(con, "
                     SELECT X,Y
                     FROM crime")
      a <- fetch(res, n = -1)
      dbClearResult(res)
```

Get a map of San Francisco taken from Stamen and add some random points to the map

```
  sf.map <- ggmap(get_map(c(-122.52, 37.7, -122.35, 37.81), scale = "auto",source = "stamen"))

  i <- sample(1:nrow(a),250000)
      sf.map +
        geom_point(aes(x = X,y = Y), data = a[i,], alpha = 0.5, color = "darkred", size = 0.1)
```



Format a hot spot / density map

```
sf.map +
  stat_density2d(aes(x = X, y = Y, fill =..level.., alpha = ..level..),
                 bins = 60, data = a[i,], geom = 'polygon') +
  scale_fill_gradient('Crime Density',low = "green",high = "red") +
  scale_alpha(range = c(.4, .75), guide = FALSE) +
  guides(fill = guide_colorbar(barwidth = 1.5, barheight = 10)) +
  labs(title = "San Francisco Crime Hotspot Map (2003 - 2018)",
       subtitle = "Number of Crime Incidents Reported to San Francisco Police Department",
       x = "Longitude",
       y = "Latitude") +

  theme_bw() +

  theme(panel.border = element_blank(),
        panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),

        axis.line = element_line(color = "#D8D8D8"),
        axis.ticks.y = element_blank(),
        axis.ticks.x = element_blank(),

        plot.title = element_text(color = "#404040", size = 12, face = "bold"),
        plot.subtitle = element_text(color = "#404040", size = 9, face = "italic"),
```

```
        axis.title.x = element_text(color = "#404040", size = 9, face = "bold"),
        axis.title.y = element_text(color = "#404040", size = 9, face = "bold"),

        strip.text.x = element_text(size=9, face = "bold"),
        strip.background = element_rect(colour="#D8D8D8", fill="#D8D8D8")
    )
```