QMSS Data Viz - Assignment #2

Julia Tache

Professor Thomas Brambor

GR5063: QMSS Data Visualization

03/30/20

Assignment #2: Mapping Fires (and hopefully putting them out)

```
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.2.1 ──
```

```
## ✓ ggplot2 3.2.1     ✓ purrr   0.3.3
## ✓ tibble  2.1.3     ✓ dplyr   0.8.4
## ✓ tidyr   1.0.2     ✓ stringr 1.4.0
## ✓ readr   1.3.1     ✓ forcats 0.4.0
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggmap)
```

```
## Google's Terms of Service: https://cloud.google.com/maps-platform/terms/.
```

```
## Please cite ggmap if you use it! See citation("ggmap") for details.
```

```
# fire_all <- read_csv("Incidents_Responded_to_by_Fire_Companies.csv")
# fire <- fire_all %>%
#   filter(INCIDENT_TYPE_DESC == "111 - Building fire")

# Register Google API Key
# register_google(key = "AIzaSyB-D2Q8JlnM8XET9G8D4roPEpEhHd6ubhE", write = TRUE)

# Create and geocode addresses
# fire <- fire %>%
#   mutate(address = str_c( str_to_title(fire$STREET_HIGHWAY),
#                    "New York, NY",
#                    fire$ZIP_CODE,
#                    sep=", ")) %>%
#   filter(is.na(address)==FALSE) %>%
#   mutate_geocode(address)

# Save File
# write_csv(fire, "building_fires.csv")
```

1. Location of Severe Fires

```
library(tidyverse)
library(leaflet)
library(sp)
library(maps)
```

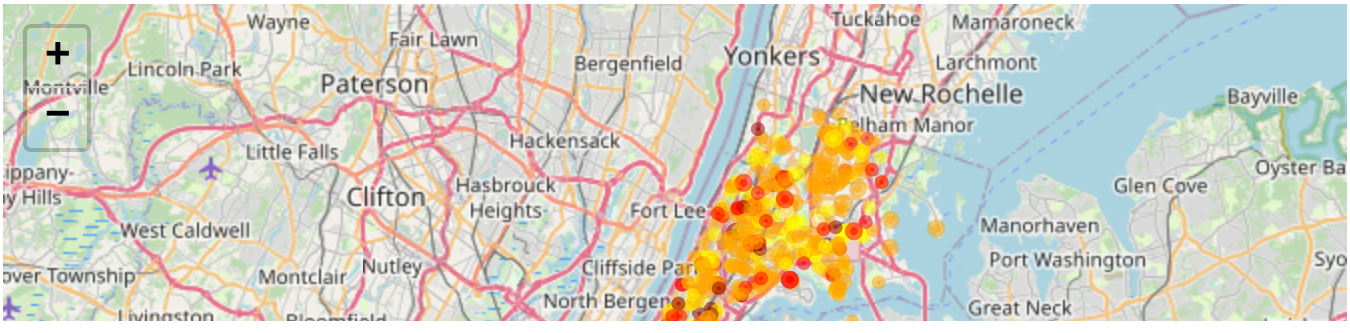
```
##
## Attaching package: 'maps'
```

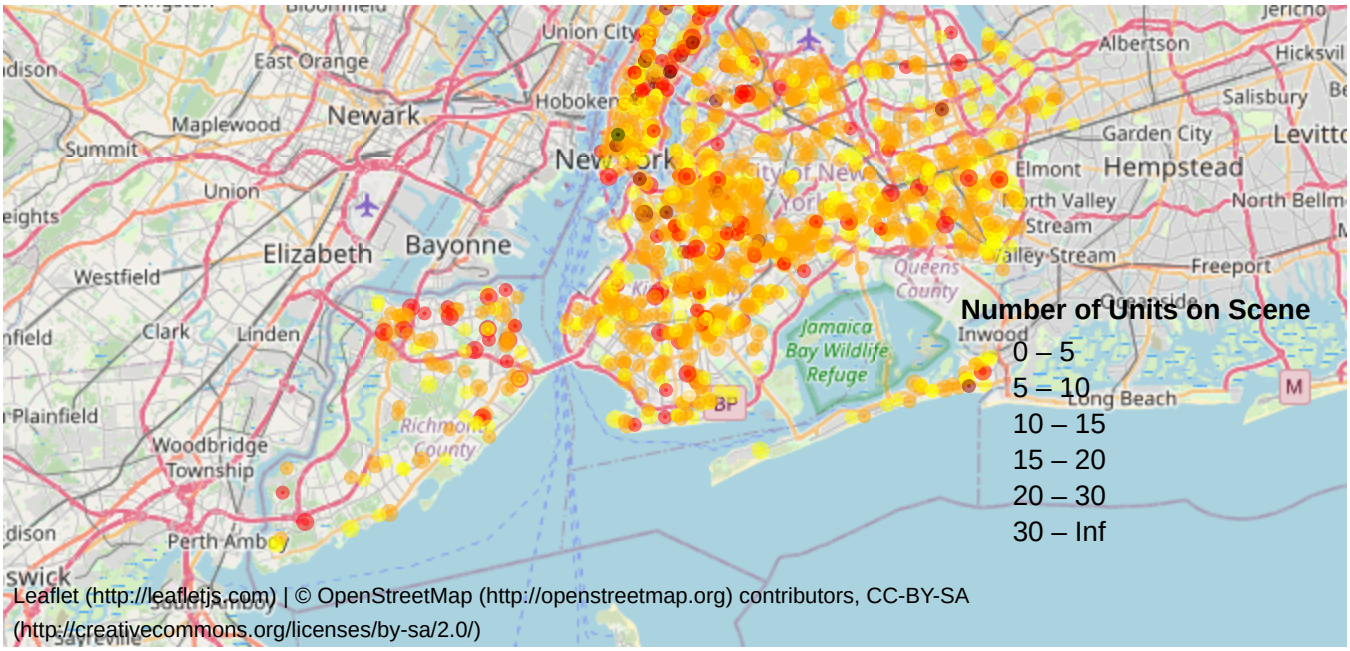
```
## The following object is masked from 'package:purrr':
##
##     map
```

```
firehouses <- read_csv("FDNY_Firehouse_Listing.csv") %>%
  dplyr::filter(!is.na(Latitude))
```

```
## Parsed with column specification:
## cols(
##   FacilityName = col_character(),
##   FacilityAddress = col_character(),
##   Borough = col_character(),
##   Postcode = col_double(),
##   Latitude = col_double(),
##   Longitude = col_double(),
##   `Community Board` = col_double(),
##   `Community Council` = col_double(),
##   `Census Tract` = col_double(),
##   BIN = col_double(),
##   BBL = col_double(),
##   NTA = col_character()
## )
```

```
fire <- read.csv("building_fires.txt")
fire <-rename(fire, "lng" = "lon")
fire <- fire %>% filter(BOROUGH_DESC == c("1 - Manhattan", "2 - Bronx", "3 - Staten Island",
                                          "4 - Brooklyn", "5 - Queens"))

severe_fire <- fire %>% filter(HIGHEST_LEVEL_DESC ==    c("7 - Signal 7-5"))
# subsets by the highest level fire

bins <- c(0, 5, 10, 15, 20, 30, Inf)
pal <- colorBin(c("light yellow", "yellow", "orange", "red", "dark red", "black"),
                domain = severe_fire$UNITS_ONSCENE, bins = bins)

map <- leaflet(data = severe_fire) %>%
  addTiles() %>%
  addCircles(lng = ~lng,
             lat = ~lat,
             color = ~pal(UNITS_ONSCENE),
             stroke =TRUE,
             fillOpacity = 0.5,
             radius = ~(as.numeric(FIRE_SPREAD_DESC)*50),
             popup = paste("Incident Type:", severe_fire$INCIDENT_TYPE_DESC, "<br>",
                           "Date and Time:",severe_fire$INCIDENT_DATE_TIME, "<br>",
                           "Fire Spread:", severe_fire$FIRE_SPREAD_DESC)) %>%
  addLegend("bottomright", pal = pal, values = ~UNITS_ONSCENE,
            title = "Number of Units on Scene",
            opacity = 1)

map
```

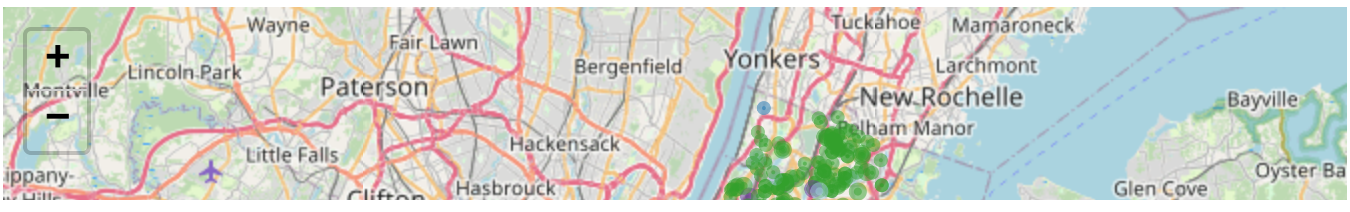


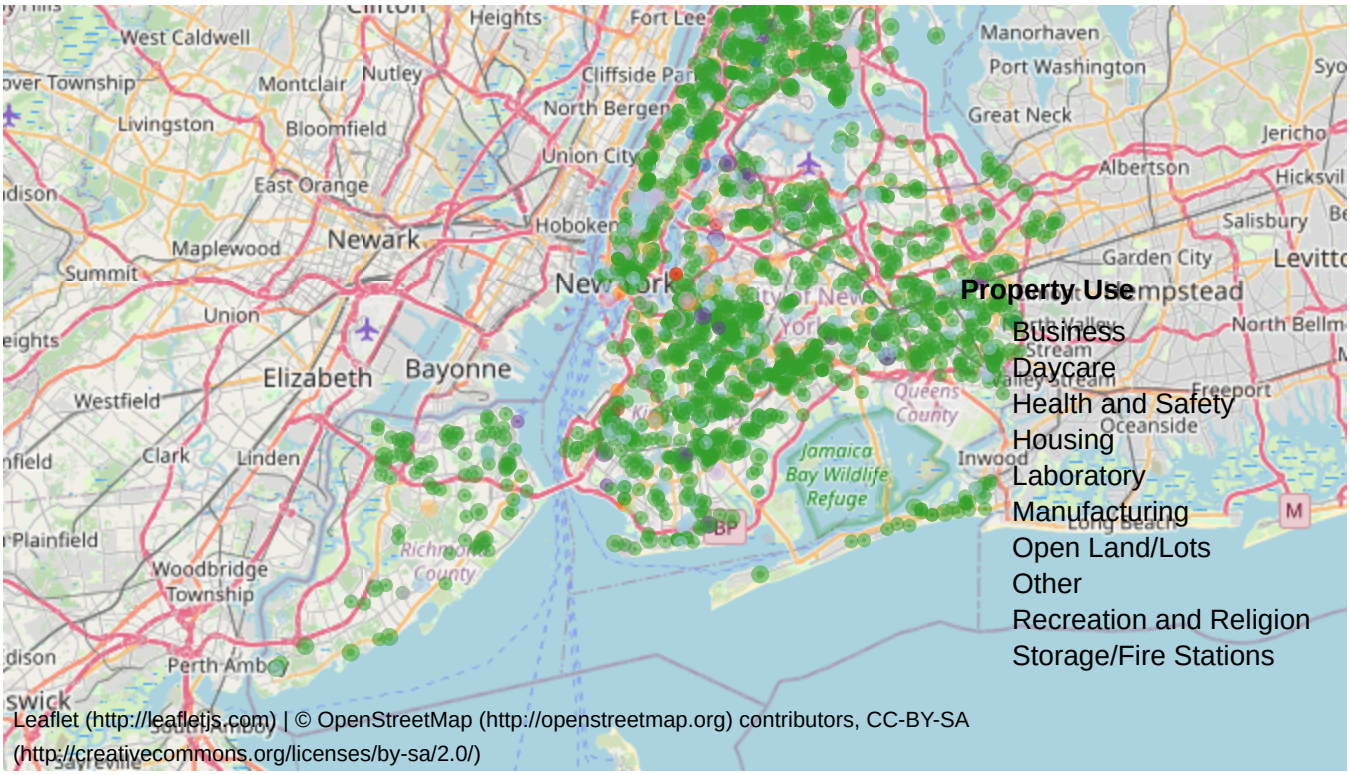


2. Layers and Clusters

a) Color by Type of Property

```
# clean the data and collapse property types into main categories
severe_fire$PROPERTY_USE_DESC <- str_replace_all(severe_fire$PROPERTY_USE_DESC,
                                                 "24-hour care Nursing homes, 4 or more persons",
                                                 "Twenty-Four Hour Nursing Homes")

severe_fire$PROPERTY_USE_DESC <- str_replace_all(severe_fire$PROPERTY_USE_DESC,
                                                 "1 or 2", "One or two")

severe_fire$PROPERTY_USE_DESC <- str_replace_all(severe_fire$PROPERTY_USE_DESC,
                                                 "UUU - Undetermined", "000")

severe_fire$PROPERTY_USE_DESC <- str_replace_all(severe_fire$PROPERTY_USE_DESC,
                                                 "\\D", "")

severe_fire$PROPERTY_USE_DESC <- gsub("000", "Other", severe_fire$PROPERTY_USE_DESC)
severe_fire$PROPERTY_USE_DESC <- gsub("1[0-9][0-9]", "Recreation and Religion", severe_fire$PROPER
TY_USE_DESC)
severe_fire$PROPERTY_USE_DESC <- gsub("2[0-9][0-9]", "Daycare", severe_fire$PROPERTY_USE_DESC)
severe_fire$PROPERTY_USE_DESC <- gsub("3[0-9][0-9]", "Health and Safety", severe_fire$PROPERTY_USE
_DESC)
severe_fire$PROPERTY_USE_DESC <- gsub("4[0-9][0-9]", "Housing", severe_fire$PROPERTY_USE_DESC)
severe_fire$PROPERTY_USE_DESC <- gsub("5[0-9][0-9]", "Business", severe_fire$PROPERTY_USE_DESC)
severe_fire$PROPERTY_USE_DESC <- gsub("6[0-9][0-9]", "Laboratory", severe_fire$PROPERTY_USE_DESC)
severe_fire$PROPERTY_USE_DESC <- gsub("7[0-9][0-9]", "Manufacturing", severe_fire$PROPERTY_USE_DES
C)
severe_fire$PROPERTY_USE_DESC <- gsub("8[0-9][0-9]", "Storage/Fire Stations", severe_fire$PROPERTY
_USE_DESC)
severe_fire$PROPERTY_USE_DESC <- gsub("9[0-9][0-9]", "Open Land/Lots", severe_fire$PROPERTY_USE_DE
SC)

pal <- colorFactor(c("Paired"), domain = severe_fire$PROPERTY_USE_DESC)

map <- leaflet(data = severe_fire) %>%
  addTiles() %>%
  addCircles(lng = ~lng,
             lat = ~lat,
             color = ~pal(PROPERTY_USE_DESC),
             stroke =TRUE,
             fillOpacity = 0.5,
             radius = ~(as.numeric(FIRE_SPREAD_DESC)*50),
             popup = paste("Property Type:", severe_fire$PROPERTY_USE_DESC, "<br>",
                           "Date and Time:",severe_fire$INCIDENT_DATE_TIME, "<br>",
                           "Fire Spread:", severe_fire$FIRE_SPREAD_DESC)) %>%
  addLegend("bottomright", pal = pal, values = ~PROPERTY_USE_DESC,
            title = "Property Use",
            opacity = 1)

map
```



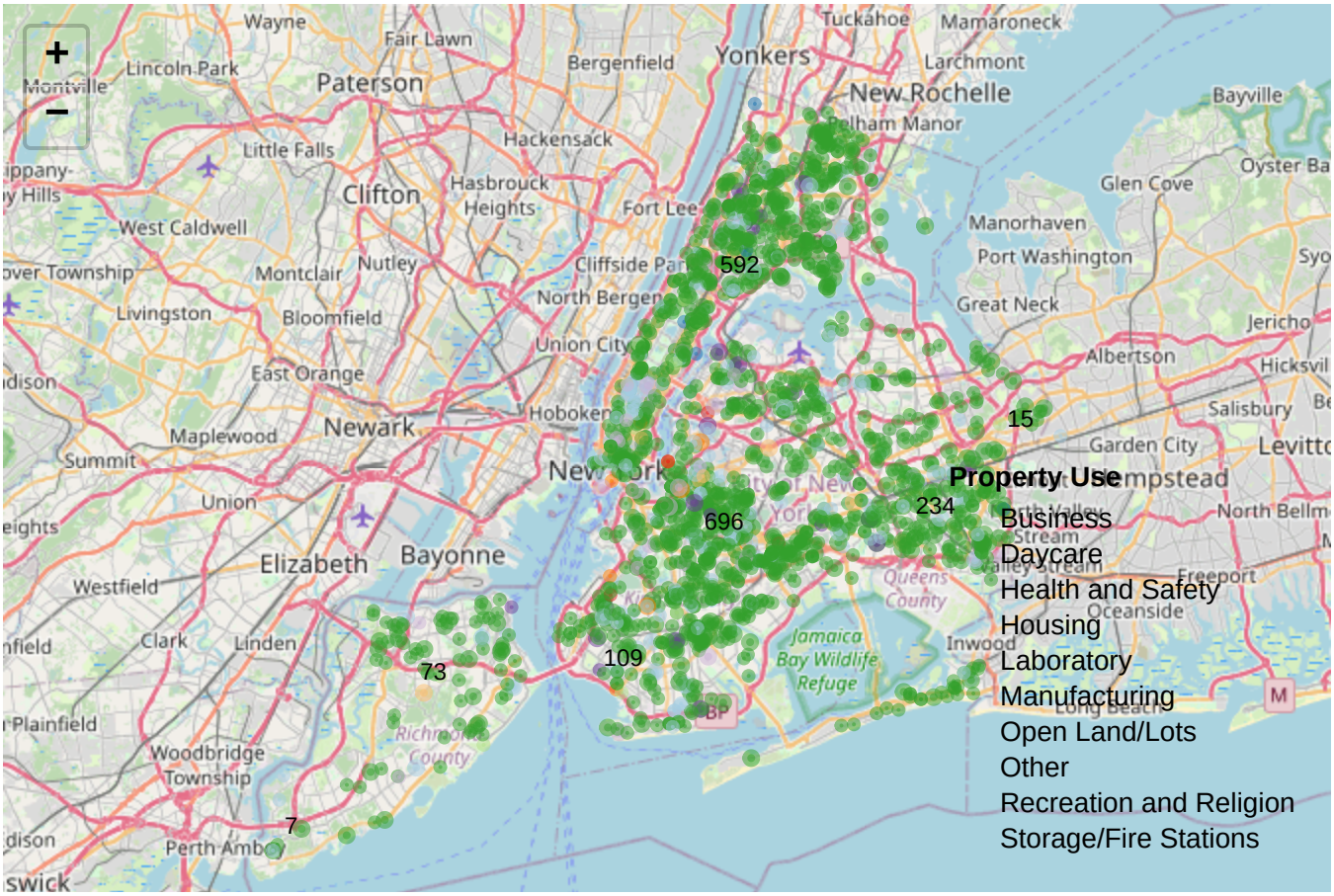


b) Cluster

```
map <- map %>% addMarkers(clusterOptions = markerClusterOptions())
```

```
## Assuming "lng" and "lat" are longitude and latitude, respectively
```

```
map
```



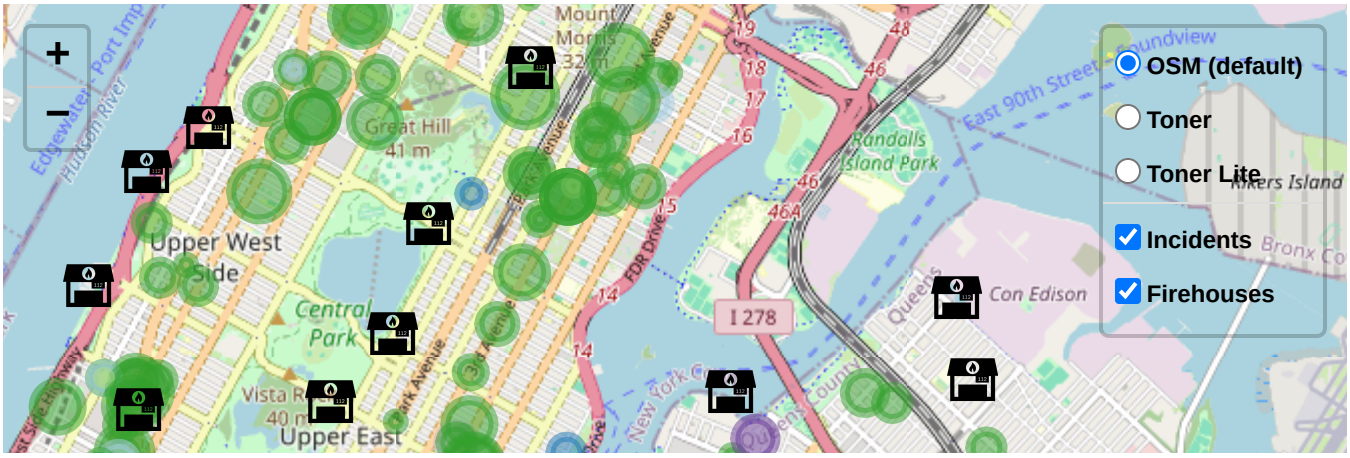
Leaflet (http://leafletjs.com) | © OpenStreetMap (http://openstreetmap.org) contributors, CC-BY-SA (http://creativecommons.org/licenses/by-sa/2.0/)

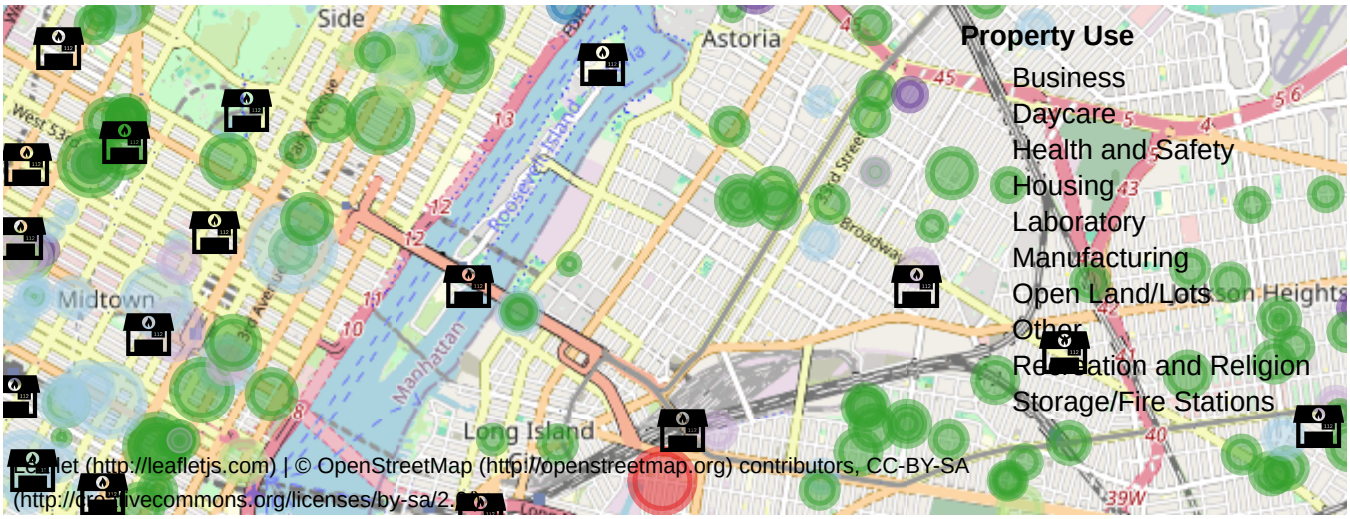
3. Fire Houses

```
firehouse_icon <- makeIcon(
  iconUrl = "https://www.pngrepo.com/png/45189/180/fire-station.png",
  iconWidth = 25, iconHeight = 25,
  iconAnchorX = 22, iconAnchorY = 94,
)

map <- leaflet(data = severe_fire) %>%
  addTiles(group = "OSM (default)") %>%
  addProviderTiles(providers$Stamen.Toner, group = "Toner") %>%
  addProviderTiles(providers$Stamen.TonerLite, group = "Toner Lite") %>%
  addCircles(lng = ~lng,
             lat = ~lat,
             color = ~pal(PROPERTY_USE_DESC),
             stroke = TRUE,
             fillOpacity = 0.5,
             radius = ~(as.numeric(UNITS_ONSCENE)*10),
             group = "Incidents",
             popup = paste("Property Type:", severe_fire$PROPERTY_USE_DESC, "<br>",
                           "Date and Time:",severe_fire$INCIDENT_DATE_TIME, "<br>",
                           "UNITS_ONSCENE:", severe_fire$UNITS_ONSCENE)) %>%
  addLegend("bottomright", pal = pal, values = ~PROPERTY_USE_DESC,
            title = "Property Use",
            opacity = 1) %>%
  addMarkers(data = firehouses,
              group = "Firehouses",
              icon = firehouse_icon,
              popup = paste("Facility Name:", firehouses$FacilityName)) %>%
  addLayersControl(
    baseGroups = c("OSM (default)", "Toner", "Toner Lite"),
    overlayGroups = c("Incidents", "Firehouses"),
    options = layersControlOptions(collapsed = FALSE)
  )
```

```
## Assuming "Longitude" and "Latitude" are longitude and latitude, respectively
```

```
map
```





Firehouse Icon Source: https://www.pngrepo.com/png/45189/180/fire-station.png (https://www.pngrepo.com/png/45189/180/fire-station.png)

4. Distance from Firehouse and Response Time

a) Calculate Distance

```
library(geosphere)
library(lubridate)
```

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

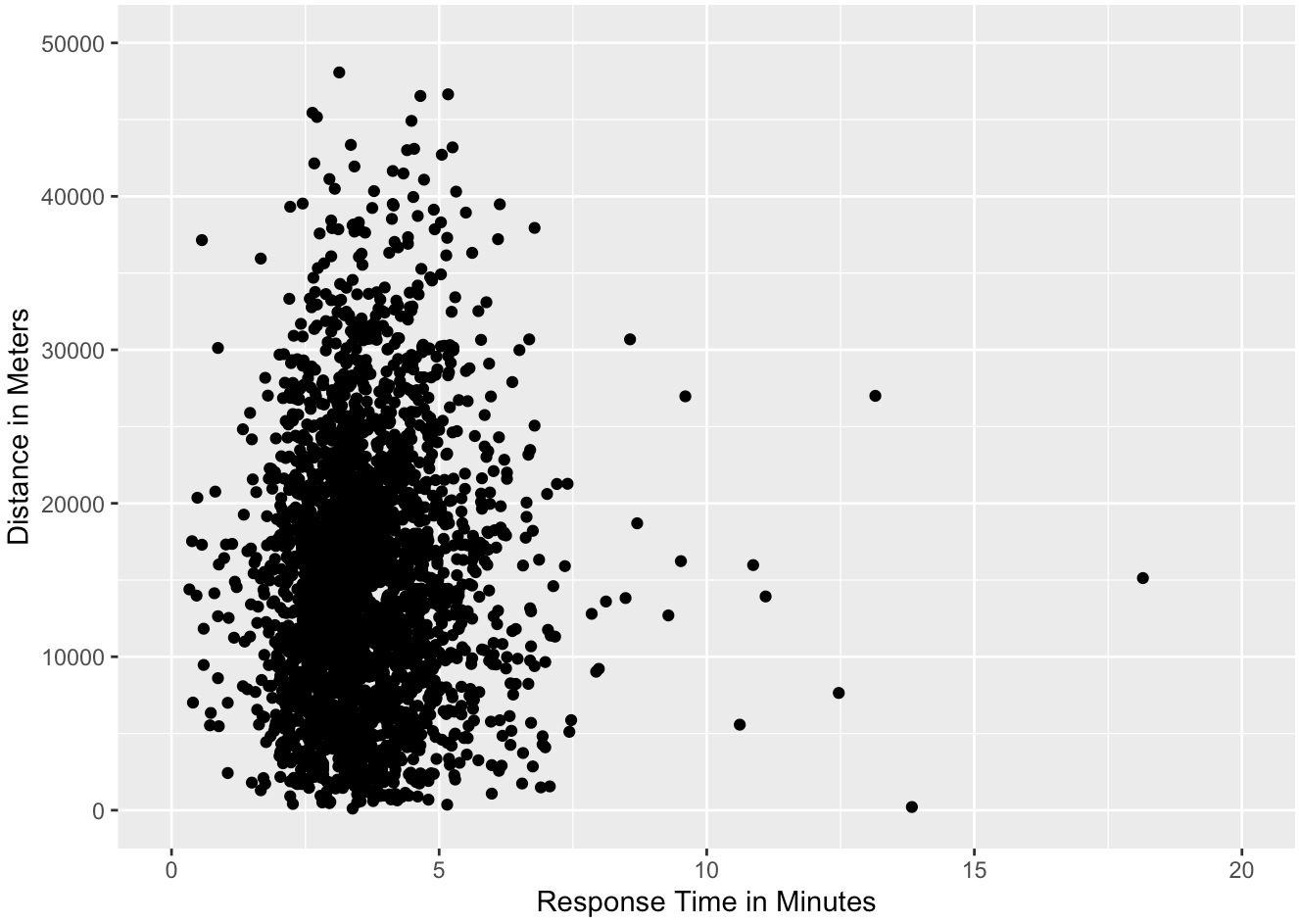
```
## The following object is masked from 'package:base':
##
##     date
```

```
# calculate shortest distance between fire and firehouse
fire <- mutate(fire,
       `Distance in Meters` = distHaversine(cbind(fire$lng, fire$lat),
                                cbind(firehouses$Longitude, firehouses$Latitude), r = 6378137))
```

```
## Warning in cbind(p1[, 1], p1[, 2], p2[, 1], p2[, 2], as.vector(r)): number
## of rows of result is not a multiple of vector length (arg 3)
```

```
fire$ARRIVAL_DATE_TIME <- mdy_hms(fire$ARRIVAL_DATE_TIME)
fire$INCIDENT_DATE_TIME <- mdy_hms(fire$INCIDENT_DATE_TIME)

# calculate response time
fire <- mutate(fire, `Response Time in Minutes` =
                 difftime(fire$ARRIVAL_DATE_TIME, fire$INCIDENT_DATE_TIME) / 60)

fire$`Response Time in Minutes` <- as.numeric(fire$`Response Time in Minutes`)
fire$`Response Time in Minutes` <- round(fire$`Response Time in Minutes`, 3)

library(ggplot2)

ggplot(fire, aes(x =`Response Time in Minutes`, y = `Distance in Meters`)) +
  geom_point() + xlim(0, 20) + ylim(0, 50000)
```

```
## Warning: Removed 4 rows containing missing values (geom_point).
```



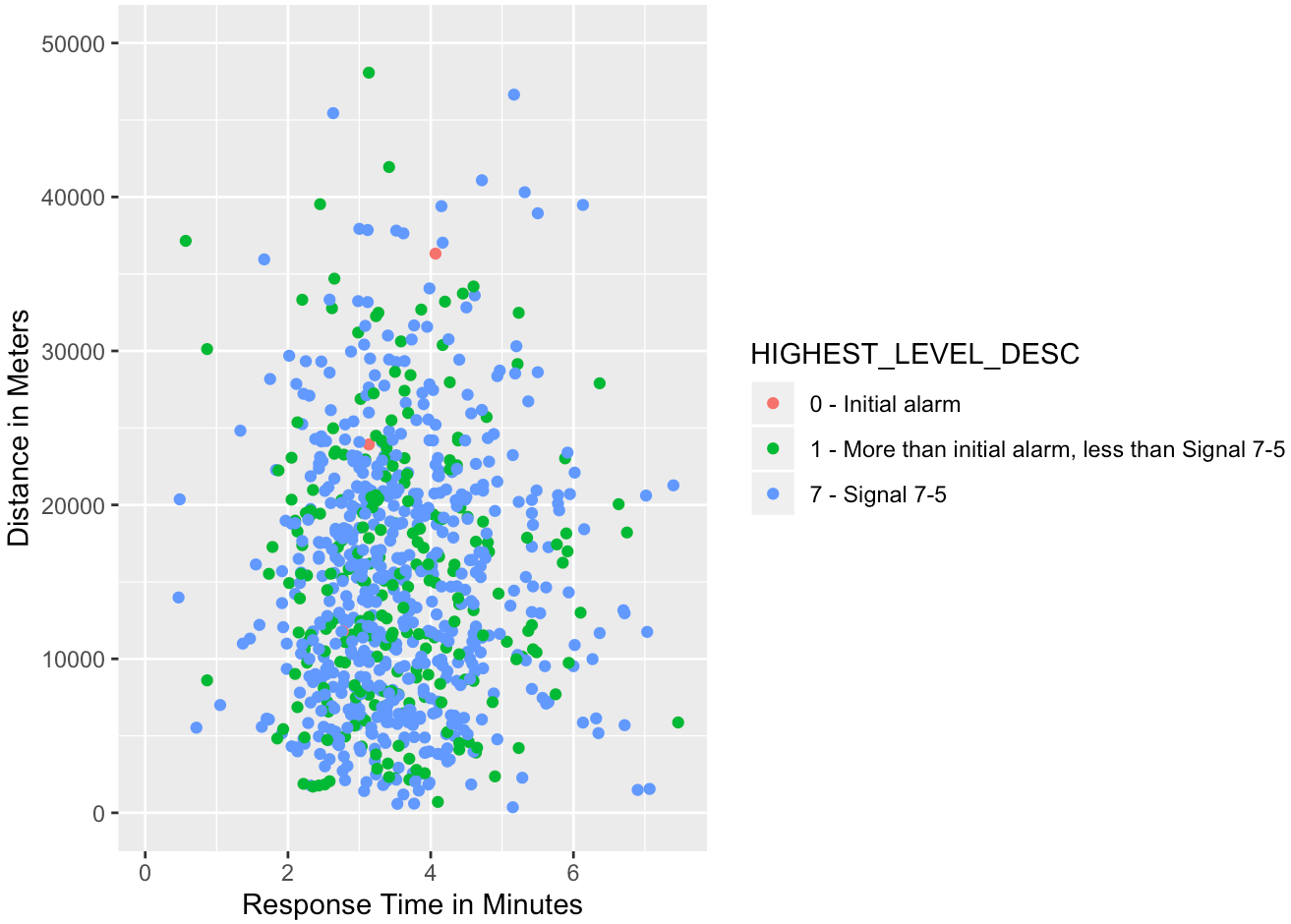

```
fire3 <- filter(fire, HIGHEST_LEVEL_DESC == c("0 - Initial alarm",
                                              "1 - More than initial alarm, less than Signal 7-5",
                                              "7 - Signal 7-5"))
```

```
## Warning in `==.default`(HIGHEST_LEVEL_DESC, c("0 - Initial alarm", "1 -
## More than initial alarm, less than Signal 7-5", : longer object length is
## not a multiple of shorter object length
```

```
## Warning in is.na(e1) | is.na(e2): longer object length is not a multiple of
## shorter object length
```

```
ggplot(fire3, aes(x =`Response Time in Minutes`, y = `Distance in Meters`,
                  color = HIGHEST_LEVEL_DESC)) +
  geom_point() + xlim(0, 7.5) + ylim(0, 50000)
```

```
## Warning: Removed 3 rows containing missing values (geom_point).
```



I had hypothesized that there would be a clear difference between the response time for high severity fires and low severity fires, but this does not seem to be the case. The scatterplot shows that there are similar response times for both Alarm 1 and Alarm 7 fires, with a slight upward trend of response times and the distance of the fire to the nearest firehouse. The response times overall, excluding outliers which are not shown in the graphs above, were very quick (2-10 minutes). A lot can happen in 2-10 minutes of a fire, so hopefully some of these less severe fires are able to be put out quickly and not become higher level fires.

b) Map of Response Times

```
library(tigris)
```

```
## To enable
## caching of data, set `options(tigris_use_cache = TRUE)` in your R script or .Rprofile.
```

```
##
## Attaching package: 'tigris'
```

```
## The following object is masked from 'package:graphics':
##
##     plot
```

```
library(dplyr)
library(leaflet)
library(sp)
library(ggmap)
library(maptools)
```

```
## Checking rgeos availability: TRUE
```

```
library(broom)
library(httr)
library(rgdal)
```

```
## rgdal: version: 1.4-8, (SVN revision 845)
##  Geospatial Data Abstraction Library extensions to R successfully loaded
##  Loaded GDAL runtime: GDAL 2.4.2, released 2019/06/28
##  Path to GDAL shared files: /Library/Frameworks/R.framework/Versions/3.6/Resources/library/sf/g
dal
##  GDAL binary built with GEOS: FALSE
##  Loaded PROJ.4 runtime: Rel. 5.2.0, September 15th, 2018, [PJ_VERSION: 520]
##  Path to PROJ.4 shared files: /Library/Frameworks/R.framework/Versions/3.6/Resources/library/s
f/proj
##  Linking to sp version: 1.3-2
```

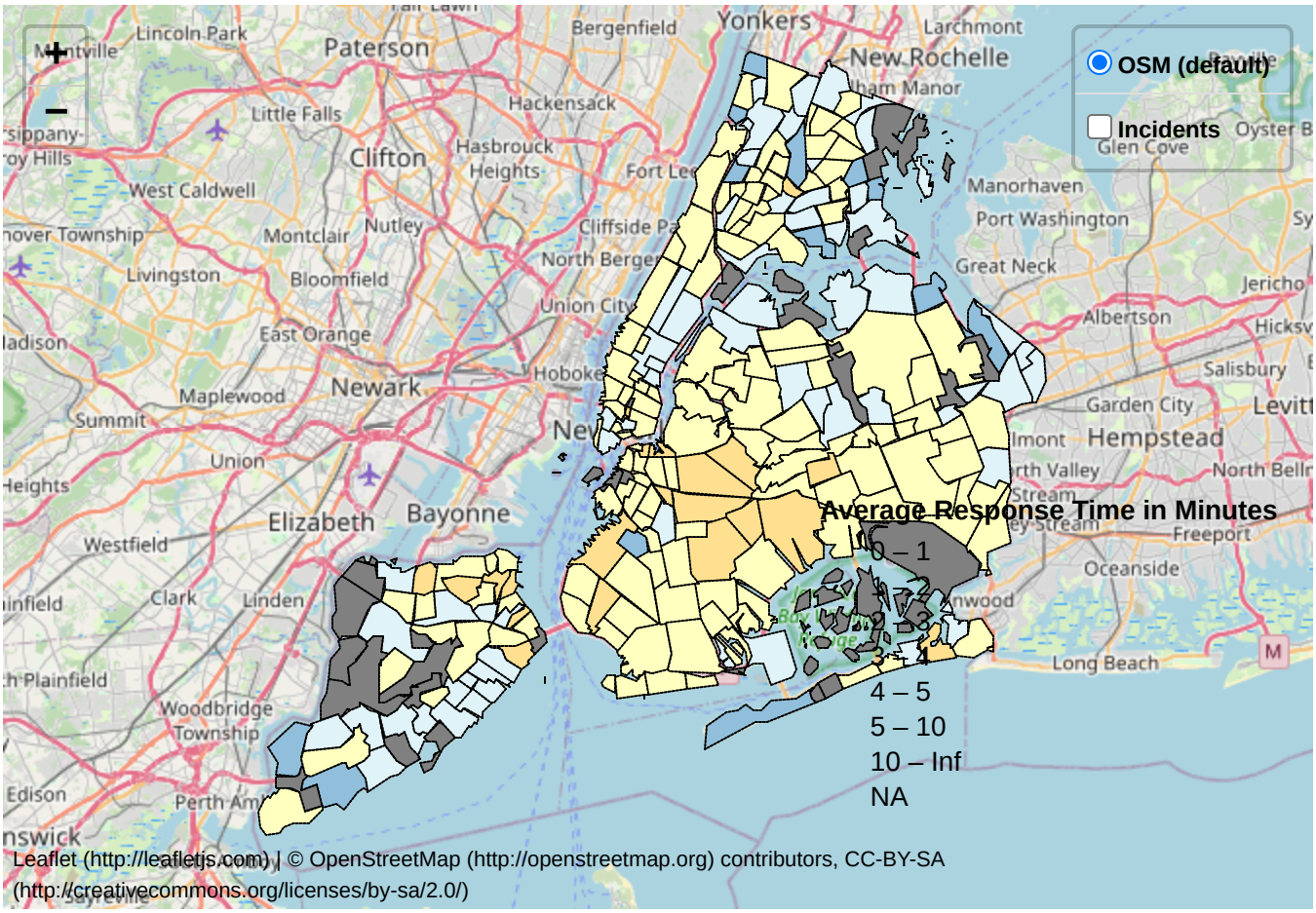
```
# open data map of nyc neighborhoods
nyc_neighborhoods <- readOGR("nyc_neighborhoods.json")
```

```
## OGR data source with driver: GeoJSON
## Source: "/Users/JuliaTache/Desktop/QMSS - MA/Spring 2020/Data Viz/course_materials/Exercises/07
_fire/nyc_neighborhoods.json", layer: "nyc_neighborhoods"
## with 310 features
## It has 4 fields
```

```
# binds existing fire data frame with nyc neighborhoods
fire_spdf <- fire
coordinates(fire_spdf) <- ~lng + lat
proj4string(fire_spdf) <- proj4string(nyc_neighborhoods)
matches <- over(fire_spdf, nyc_neighborhoods)
points <- cbind(fire, matches)

# summarizing average response time by neighborhood for easy mapping
time_by_neighborhood <- points %>%
  group_by(neighborhood) %>%
  summarize(Average = round(mean(`Response Time in Minutes`, na.rm = TRUE), digits = 3))
```

```
## Warning: Factor `neighborhood` contains implicit NA, consider using
## `forcats::fct_explicit_na`
```

```
map_data <- geo_join(nyc_neighborhoods, time_by_neighborhood, "neighborhood", "neighborhood")

bins <- c(0, 1, 2, 3, 4, 5, 10, Inf)
pal <- colorBin("RdYlBu", domain = map_data$Average, bins = bins)

leaflet(map_data) %>%
  addTiles() %>%
  addPolygons(fillColor = ~pal(Average),
              popup = paste("Neighborhood:", map_data$neighborhood, "<br>",
                            "Average Response Time (in minutes):", map_data$Average),
              opacity = 1,
              color = "black",
              dashArray = "1",
              fillOpacity = 1,
              weight = 1,
              highlight = highlightOptions(
                weight = 5,
                color = "white",
                dashArray = "",
                fillOpacity = 0.7,
                bringToFront = TRUE)) %>%
                addLegend(pal = pal, values = ~Average, opacity = 0.7,
                          position = "bottomright", title = "Average Response Time in Minutes") %
>%
                addMarkers(data = fire,
                           lng = ~lng,
                           lat = ~lat,
                           group = "Incidents",
                           popup = paste("Property Type:", fire$PROPERTY_USE_DESC, "<br>",
                                         "Response Time (in minutes):", fire$`Response Time in Min
utes`, "<br>",
                                         "Level of Fire:", fire$HIGHEST_LEVEL_DESC)) %>%
  addLayersControl(
                  baseGroups = c("OSM (default)"),
                  overlayGroups = c("Incidents"),
                  options = layersControlOptions(collapsed = FALSE))
```



Based on the markers, it does not seem like the property type or level of fire really play much of a role in the response time. However, the map showcases how distance does play a role, as we can see neighborhoods out in Queens and Staten Island have sometimes the longest wait times of 8+ minutes.

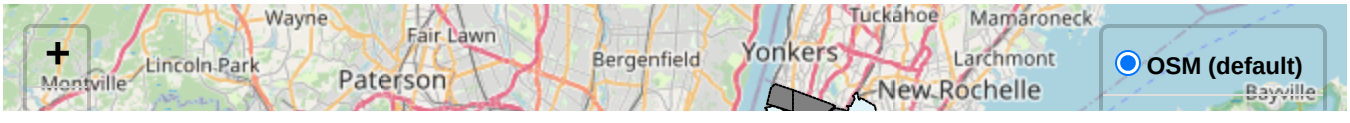
```
fire$INCIDENT_DATE_TIME <- str_sub(fire$INCIDENT_DATE_TIME, 1, 4)

fire_13 <- filter(fire, INCIDENT_DATE_TIME == "2013")
fire_14 <- filter(fire, INCIDENT_DATE_TIME == "2014")
fire_15 <- filter(fire, INCIDENT_DATE_TIME == "2015")
fire_16 <- filter(fire, INCIDENT_DATE_TIME == "2016")
fire_17 <- filter(fire, INCIDENT_DATE_TIME == "2017")
fire_18 <- filter(fire, INCIDENT_DATE_TIME == "2018")

nyc_neighborhoods <- readOGR("nyc_neighborhoods.json")
```

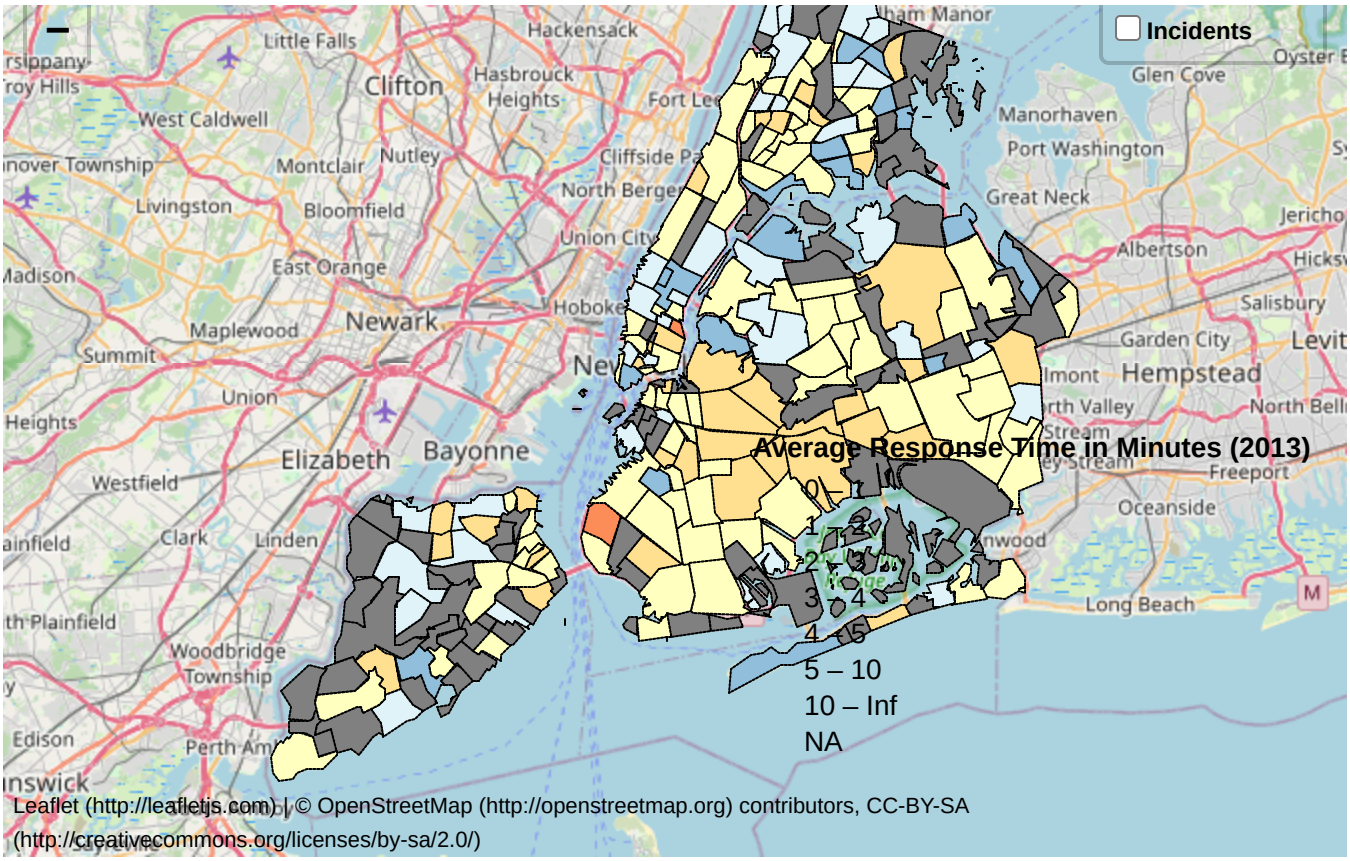
```
## OGR data source with driver: GeoJSON
## Source: "/Users/JuliaTache/Desktop/QMSS - MA/Spring 2020/Data Viz/course_materials/Exercises/07
_fire/nyc_neighborhoods.json", layer: "nyc_neighborhoods"
## with 310 features
## It has 4 fields
```

```
fire_spdf_13 <- fire_13
coordinates(fire_spdf_13) <- ~lng + lat
proj4string(fire_spdf_13) <- proj4string(nyc_neighborhoods)
matches <- over(fire_spdf_13, nyc_neighborhoods)
points <- cbind(fire_13, matches)

time_by_neighborhood <- points %>%
  group_by(neighborhood) %>%
  summarize(Average = round(mean(`Response Time in Minutes`, na.rm = TRUE), digits = 3))
```

```
## Warning: Factor `neighborhood` contains implicit NA, consider using
## `forcats::fct_explicit_na`
```

```
map_data <- geo_join(nyc_neighborhoods, time_by_neighborhood, "neighborhood", "neighborhood")

bins <- c(0, 1, 2, 3, 4, 5, 10, Inf)
pal <- colorBin("RdYlBu", domain = map_data$Average, bins = bins)

leaflet(map_data) %>%
  addTiles() %>%
  addPolygons(fillColor = ~pal(Average),
              popup = paste("Neighborhood:", map_data$neighborhood, "<br>",
                            "Average Response Time (in minutes):", map_data$Average),
              opacity = 1,
              color = "black",
              weight = 1,
              dashArray = "1",
              fillOpacity = 1,
              highlight = highlightOptions(
                weight = 5,
                color = "white",
                dashArray = "",
                fillOpacity = 0.7,
                bringToFront = TRUE)) %>%
                addLegend(pal = pal, values = ~Average, opacity = 0.7,
                          position = "bottomright", title = "Average Response Time in Minutes (201
3)") %>%
                addMarkers(data = fire_13,
                           lng = ~lng,
                           lat = ~lat,
                           group = "Incidents",
                           popup = paste("Property Type:", fire_13$PROPERTY_USE_DESC, "<br>",
                                         "Response Time (in minutes):", fire_13$`Response Time in
Minutes`, "<br>",
                                         "Level of Fire:", fire_13$HIGHEST_LEVEL_DESC)) %>%
  addLayersControl(
                  baseGroups = c("OSM (default)"),
                  overlayGroups = c("Incidents"),
                  options = layersControlOptions(collapsed = FALSE))
```





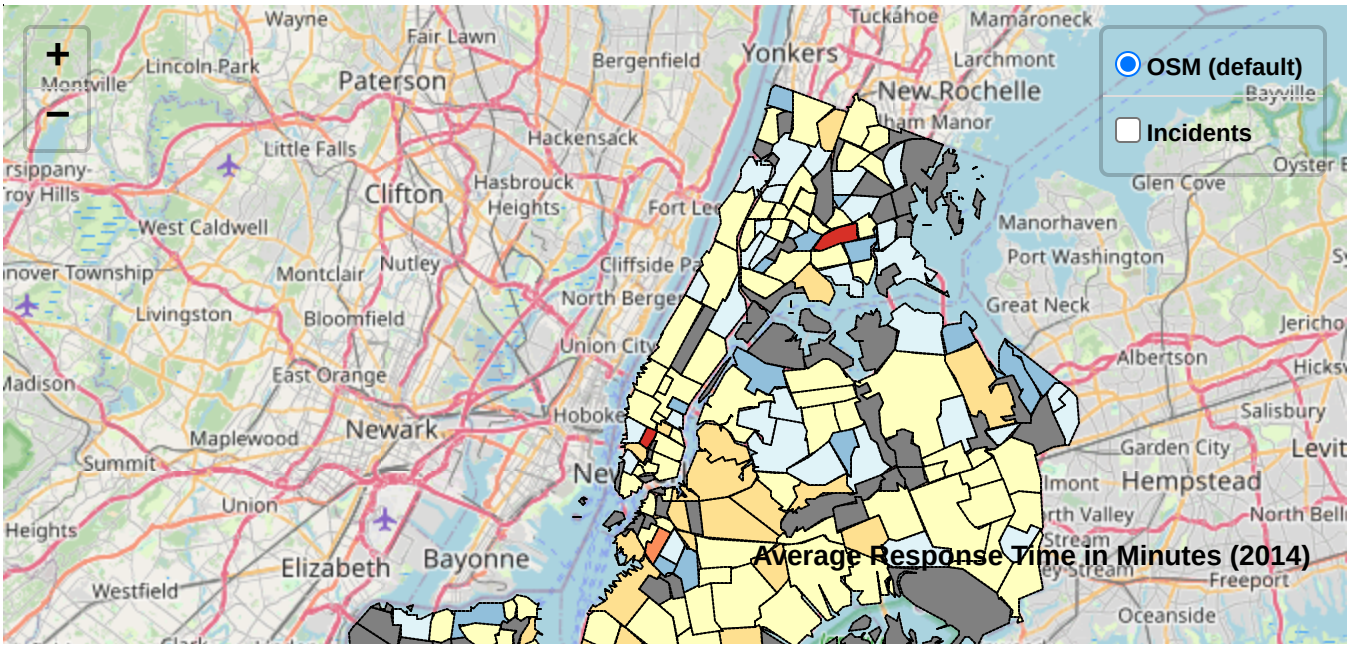

```
nyc_neighborhoods <- readOGR("nyc_neighborhoods.json")
```

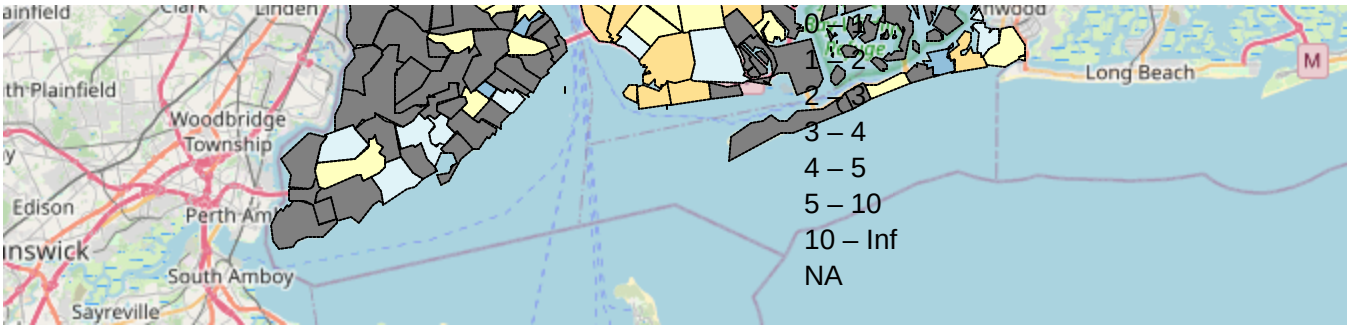
```
## OGR data source with driver: GeoJSON
## Source: "/Users/JuliaTache/Desktop/QMSS - MA/Spring 2020/Data Viz/course_materials/Exercises/07
_fire/nyc_neighborhoods.json", layer: "nyc_neighborhoods"
## with 310 features
## It has 4 fields
```

```
fire_spdf_14 <- fire_14
coordinates(fire_spdf_14) <- ~lng + lat
proj4string(fire_spdf_14) <- proj4string(nyc_neighborhoods)
matches <- over(fire_spdf_14, nyc_neighborhoods)
points <- cbind(fire_14, matches)

time_by_neighborhood <- points %>%
  group_by(neighborhood) %>%
  summarize(Average = round(mean(`Response Time in Minutes`, na.rm = TRUE), digits = 3))
```

```
## Warning: Factor `neighborhood` contains implicit NA, consider using
## `forcats::fct_explicit_na`
```

```
map_data <- geo_join(nyc_neighborhoods, time_by_neighborhood, "neighborhood", "neighborhood")

bins <- c(0, 1, 2, 3, 4, 5, 10, Inf)
pal <- colorBin("RdYlBu", domain = map_data$Average, bins = bins)

leaflet(map_data) %>%
  addTiles() %>%
  addPolygons(fillColor = ~pal(Average),
              popup = paste("Neighborhood:", map_data$neighborhood, "<br>",
                            "Average Response Time (in minutes):", map_data$Average),
              opacity = 1,
              color = "black",
              dashArray = "1",
              weight = 1,
              fillOpacity = 1,
              highlight = highlightOptions(
                weight = 5,
                color = "white",
                dashArray = "",
                fillOpacity = 0.7,
                bringToFront = TRUE)) %>%
                addLegend(pal = pal, values = ~Average, opacity = 0.7,
                          position = "bottomright", title = "Average Response Time in Minutes (201
4)") %>%
                addMarkers(data = fire_14,
                           lng = ~lng,
                           lat = ~lat,
                           group = "Incidents",
                           popup = paste("Property Type:", fire_14$PROPERTY_USE_DESC, "<br>",
                                         "Response Time (in minutes):", fire_14$`Response Time in
Minutes`, "<br>",
                                         "Level of Fire:", fire_14$HIGHEST_LEVEL_DESC)) %>%
  addLayersControl(
                  baseGroups = c("OSM (default)"),
                  overlayGroups = c("Incidents"),
                  options = layersControlOptions(collapsed = FALSE))
```





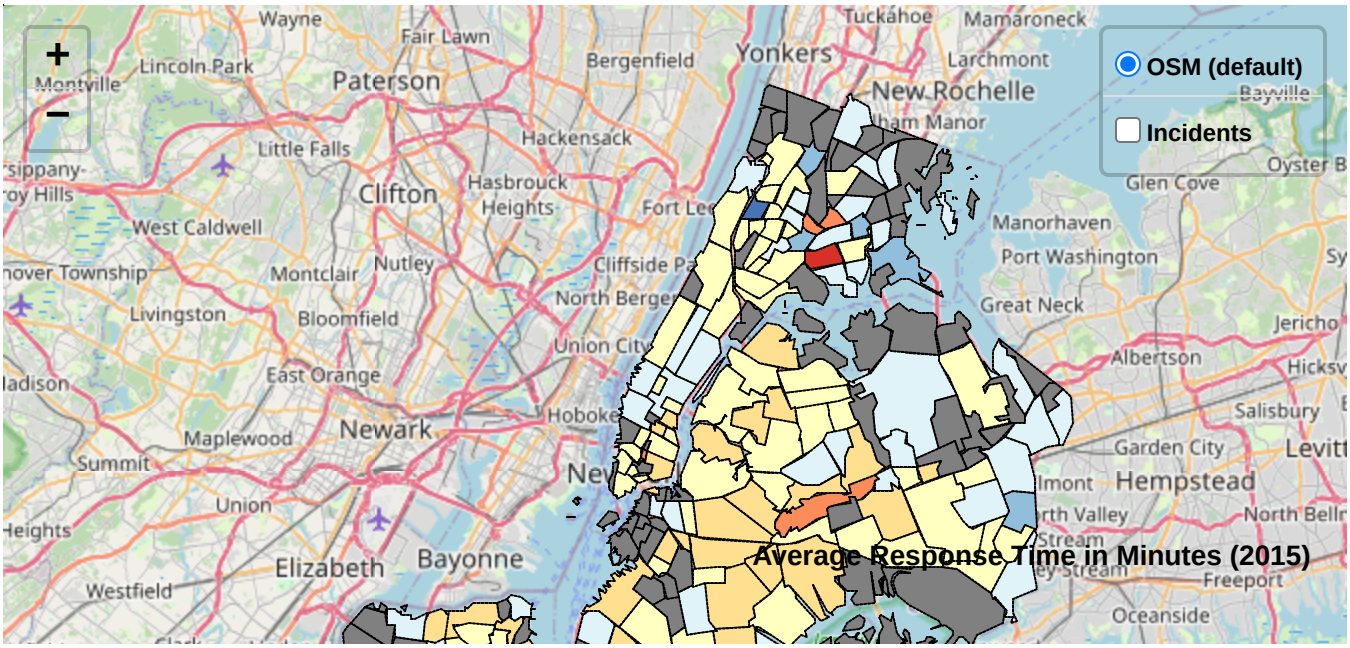

```
nyc_neighborhoods <- readOGR("nyc_neighborhoods.json")
```

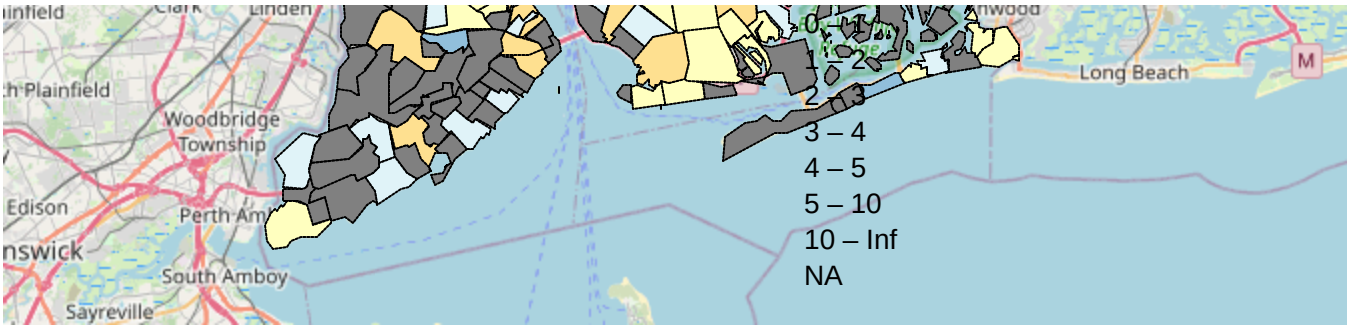
```
## OGR data source with driver: GeoJSON
## Source: "/Users/JuliaTache/Desktop/QMSS - MA/Spring 2020/Data Viz/course_materials/Exercises/07
_fire/nyc_neighborhoods.json", layer: "nyc_neighborhoods"
## with 310 features
## It has 4 fields
```

```
fire_spdf_15 <- fire_15
coordinates(fire_spdf_15) <- ~lng + lat
proj4string(fire_spdf_15) <- proj4string(nyc_neighborhoods)
matches <- over(fire_spdf_15, nyc_neighborhoods)
points <- cbind(fire_15, matches)

time_by_neighborhood <- points %>%
  group_by(neighborhood) %>%
  summarize(Average = round(mean(`Response Time in Minutes`, na.rm = TRUE), digits = 3))
```

```
## Warning: Factor `neighborhood` contains implicit NA, consider using
## `forcats::fct_explicit_na`
```

```
map_data <- geo_join(nyc_neighborhoods, time_by_neighborhood, "neighborhood", "neighborhood")

bins <- c(0, 1, 2, 3, 4, 5, 10, Inf)
pal <- colorBin("RdYlBu", domain = map_data$Average, bins = bins)

leaflet(map_data) %>%
  addTiles() %>%
  addPolygons(fillColor = ~pal(Average),
              popup = paste("Neighborhood:", map_data$neighborhood, "<br>",
                            "Average Response Time (in minutes):", map_data$Average),
              opacity = 1,
              color = "black",
              dashArray = "1",
              weight = 1,
              fillOpacity = 1,
              highlight = highlightOptions(
                weight = 5,
                color = "white",
                dashArray = "",
                fillOpacity = 0.7,
                bringToFront = TRUE)) %>%
                addLegend(pal = pal, values = ~Average, opacity = 0.7,
                          position = "bottomright", title = "Average Response Time in Minutes (201
5)") %>%
                addMarkers(data = fire_15,
                           lng = ~lng,
                           lat = ~lat,
                           group = "Incidents",
                           popup = paste("Property Type:", fire_15$PROPERTY_USE_DESC, "<br>",
                                         "Response Time (in minutes):", fire_15$`Response Time in
Minutes`, "<br>",
                                         "Level of Fire:", fire_15$HIGHEST_LEVEL_DESC)) %>%
  addLayersControl(
                  baseGroups = c("OSM (default)"),
                  overlayGroups = c("Incidents"),
                  options = layersControlOptions(collapsed = FALSE))
```

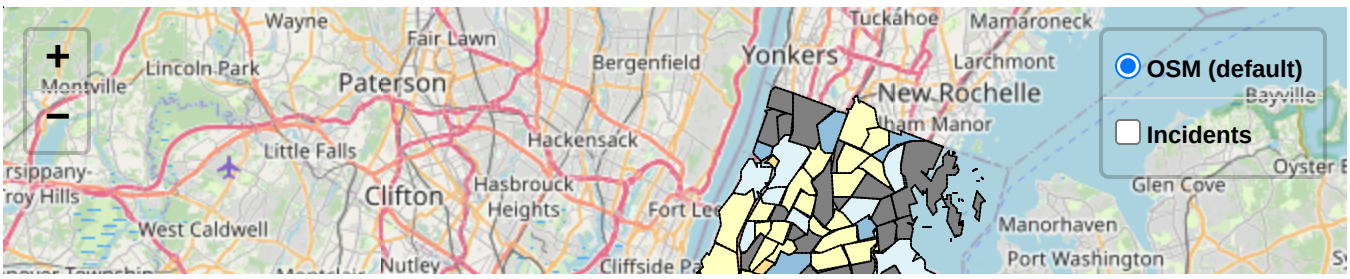


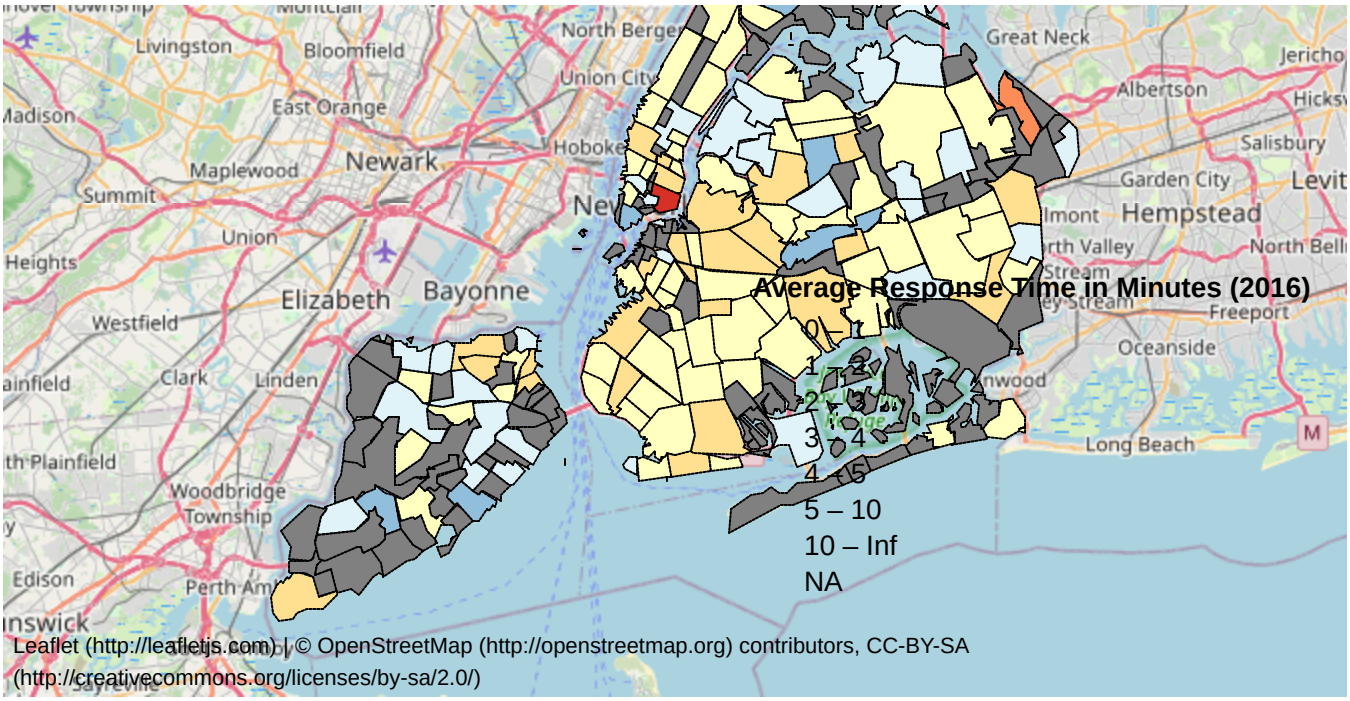



```
nyc_neighborhoods <- readOGR("nyc_neighborhoods.json")
```

```
## OGR data source with driver: GeoJSON
## Source: "/Users/JuliaTache/Desktop/QMSS - MA/Spring 2020/Data Viz/course_materials/Exercises/07
_fire/nyc_neighborhoods.json", layer: "nyc_neighborhoods"
## with 310 features
## It has 4 fields
```

```
fire_spdf_16 <- fire_16
coordinates(fire_spdf_16) <- ~lng + lat
proj4string(fire_spdf_16) <- proj4string(nyc_neighborhoods)
matches <- over(fire_spdf_16, nyc_neighborhoods)
points <- cbind(fire_16, matches)

time_by_neighborhood <- points %>%
  group_by(neighborhood) %>%
  summarize(Average = round(mean(`Response Time in Minutes`, na.rm = TRUE), digits = 3))

map_data <- geo_join(nyc_neighborhoods, time_by_neighborhood, "neighborhood", "neighborhood")

bins <- c(0, 1, 2, 3, 4, 5, 10, Inf)
pal <- colorBin("RdYlBu", domain = map_data$Average, bins = bins)

leaflet(map_data) %>%
  addTiles() %>%
  addPolygons(fillColor = ~pal(Average),
              popup = paste("Neighborhood:", map_data$neighborhood, "<br>",
                            "Average Response Time (in minutes):", map_data$Average),
              opacity = 1,
              color = "black",
              dashArray = "1",
              weight = 1,
              fillOpacity = 1,
              highlight = highlightOptions(
                weight = 5,
                color = "white",
                dashArray = "",
                fillOpacity = 0.7,
                bringToFront = TRUE)) %>%
                addLegend(pal = pal, values = ~Average, opacity = 0.7,
                          position = "bottomright", title = "Average Response Time in Minutes (201
6)") %>%
                addMarkers(data = fire_16,
                           lng = ~lng,
                           lat = ~lat,
                           group = "Incidents",
                           popup = paste("Property Type:", fire_16$PROPERTY_USE_DESC, "<br>",
                                         "Response Time (in minutes):", fire_16$`Response Time in
Minutes`, "<br>",
                                         "Level of Fire:", fire_16$HIGHEST_LEVEL_DESC)) %>%
  addLayersControl(
                  baseGroups = c("OSM (default)"),
                  overlayGroups = c("Incidents"),
                  options = layersControlOptions(collapsed = FALSE))
```





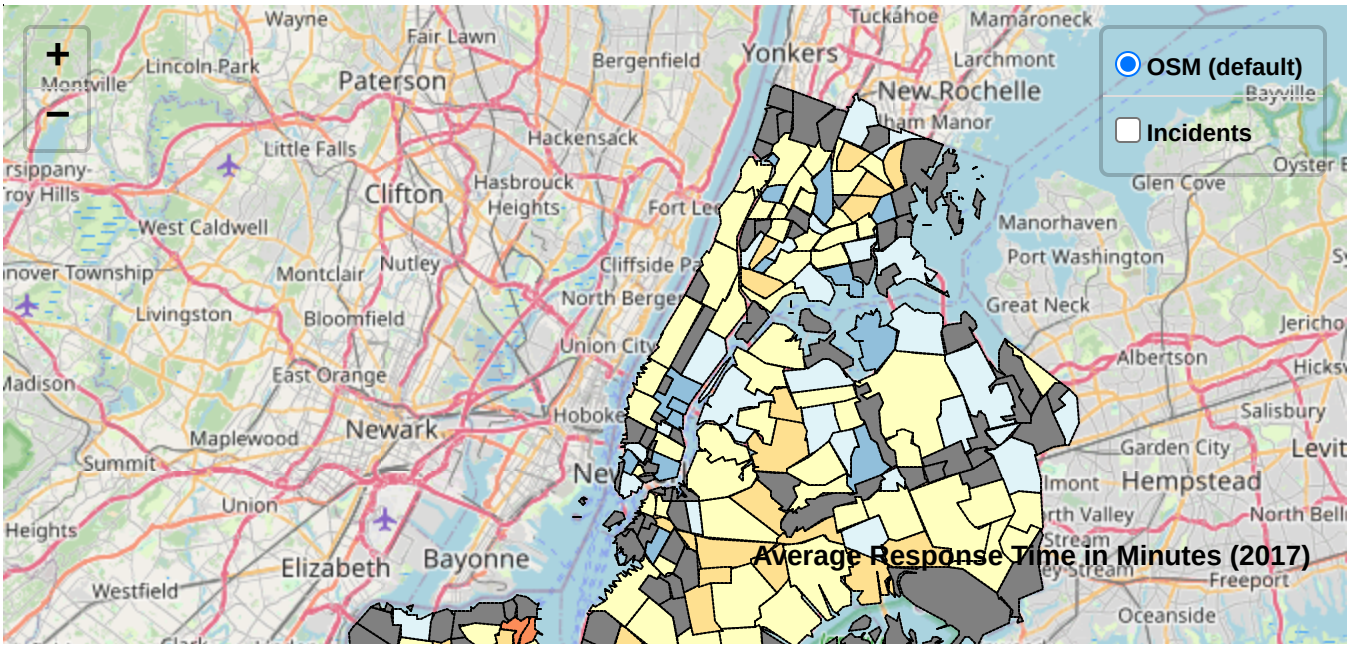

```
nyc_neighborhoods <- readOGR("nyc_neighborhoods.json")
```

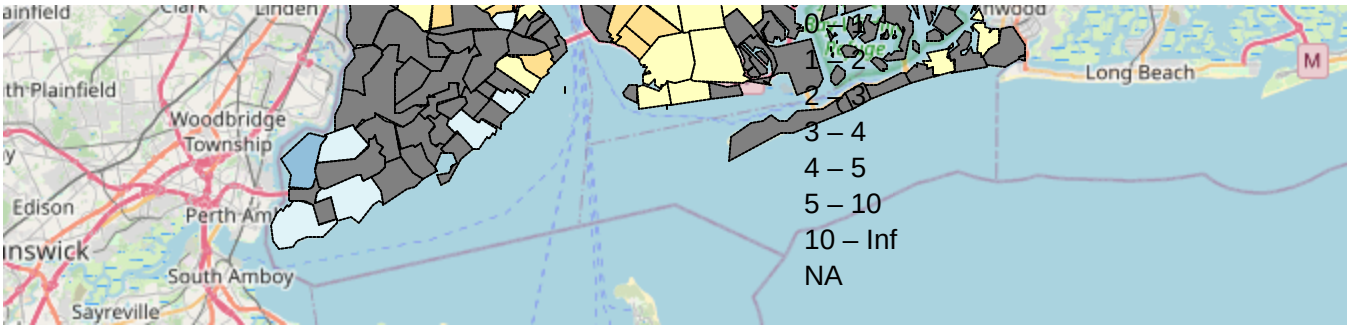
```
## OGR data source with driver: GeoJSON
## Source: "/Users/JuliaTache/Desktop/QMSS - MA/Spring 2020/Data Viz/course_materials/Exercises/07
_fire/nyc_neighborhoods.json", layer: "nyc_neighborhoods"
## with 310 features
## It has 4 fields
```

```
fire_spdf_17 <- fire_17
coordinates(fire_spdf_17) <- ~lng + lat
proj4string(fire_spdf_17) <- proj4string(nyc_neighborhoods)
matches <- over(fire_spdf_17, nyc_neighborhoods)
points <- cbind(fire_17, matches)

time_by_neighborhood <- points %>%
  group_by(neighborhood) %>%
  summarize(Average = round(mean(`Response Time in Minutes`, na.rm = TRUE), digits = 3))
```

```
## Warning: Factor `neighborhood` contains implicit NA, consider using
## `forcats::fct_explicit_na`
```

```
map_data <- geo_join(nyc_neighborhoods, time_by_neighborhood, "neighborhood", "neighborhood")

bins <- c(0, 1, 2, 3, 4, 5, 10, Inf)
pal <- colorBin("RdYlBu", domain = map_data$Average, bins = bins)

leaflet(map_data) %>%
  addTiles() %>%
  addPolygons(fillColor = ~pal(Average),
              popup = paste("Neighborhood:", map_data$neighborhood, "<br>",
                            "Average Response Time (in minutes):", map_data$Average),
              opacity = 1,
              color = "black",
              dashArray = "1",
              fillOpacity = 1,
              weight = 1,
              highlight = highlightOptions(
                weight = 5,
                color = "white",
                dashArray = "",
                fillOpacity = 0.7,
                bringToFront = TRUE)) %>%
                addLegend(pal = pal, values = ~Average, opacity = 0.7,
                          position = "bottomright", title = "Average Response Time in Minutes (201
7)") %>%
                addMarkers(data = fire_17,
                           lng = ~lng,
                           lat = ~lat,
                           group = "Incidents",
                           popup = paste("Property Type:", fire_17$PROPERTY_USE_DESC, "<br>",
                                         "Response Time (in minutes):", fire_17$`Response Time in
Minutes`, "<br>",
                                         "Level of Fire:", fire_17$HIGHEST_LEVEL_DESC)) %>%
  addLayersControl(
                  baseGroups = c("OSM (default)"),
                  overlayGroups = c("Incidents"),
                  options = layersControlOptions(collapsed = FALSE))
```

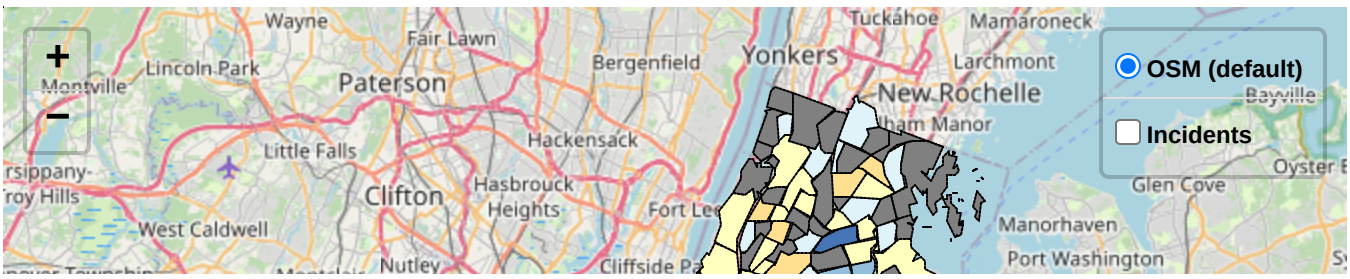


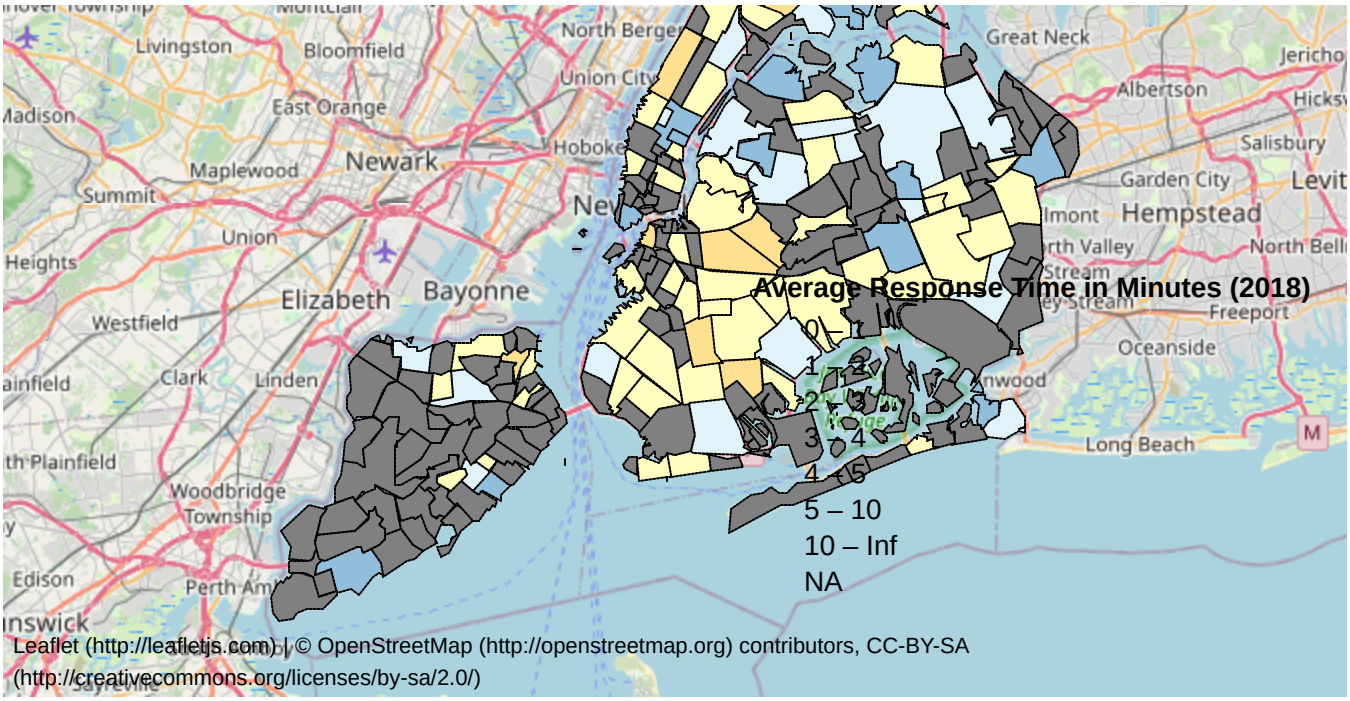



```
nyc_neighborhoods <- readOGR("nyc_neighborhoods.json")
```

```
## OGR data source with driver: GeoJSON
## Source: "/Users/JuliaTache/Desktop/QMSS - MA/Spring 2020/Data Viz/course_materials/Exercises/07
_fire/nyc_neighborhoods.json", layer: "nyc_neighborhoods"
## with 310 features
## It has 4 fields
```

```
fire_spdf_18 <- fire_18
coordinates(fire_spdf_18) <- ~lng + lat
proj4string(fire_spdf_18) <- proj4string(nyc_neighborhoods)
matches <- over(fire_spdf_18, nyc_neighborhoods)
points <- cbind(fire_18, matches)

time_by_neighborhood <- points %>%
  group_by(neighborhood) %>%
  summarize(Average = round(mean(`Response Time in Minutes`, na.rm = TRUE), digits = 3))

map_data <- geo_join(nyc_neighborhoods, time_by_neighborhood, "neighborhood", "neighborhood")

bins <- c(0, 1, 2, 3, 4, 5, 10, Inf)
pal <- colorBin("RdYlBu", domain = map_data$Average, bins = bins)

leaflet(map_data) %>%
  addTiles() %>%
  addPolygons(fillColor = ~pal(Average),
              popup = paste("Neighborhood:", map_data$neighborhood, "<br>",
                            "Average Response Time (in minutes):", map_data$Average),
              opacity = 1,
              color = "black",
              dashArray = "1",
              fillOpacity = 1,
              weight = 1,
              highlight = highlightOptions(
                weight = 5,
                color = "white",
                dashArray = "",
                fillOpacity = 0.7,
                bringToFront = TRUE)) %>%
                addLegend(pal = pal, values = ~Average, opacity = 0.7,
                          position = "bottomright", title = "Average Response Time in Minutes (201
8)") %>%
                addMarkers(data = fire_18,
                           lng = ~lng,
                           lat = ~lat,
                           group = "Incidents",
                           popup = paste("Property Type:", fire_18$PROPERTY_USE_DESC, "<br>",
                                         "Response Time (in minutes):", fire_18$`Response Time in
Minutes`, "<br>",
                                         "Level of Fire:", fire_18$HIGHEST_LEVEL_DESC)) %>%
  addLayersControl(
                  baseGroups = c("OSM (default)"),
                  overlayGroups = c("Incidents"),
                  options = layersControlOptions(collapsed = FALSE))
```





Over time (2013 - 2018), response times seem to have gotten faster given the increased amount of warm colors on the map, but there seems to have been a slowdown of response times from 2017-2018. There are also a lot of neighborhoods with “NAs” especially in later years in the data, which I am hoping means that there were less fires that the FDNY responded to as the years went on. Some trends remain consistent: heavily populated areas/areas close to the city center (Manhattan) seem to have faster response times, while fires out on the edges of the boroughs tend to have units arrive on the scene later.

Citations:

https://www.r-graph-gallery.com/327-chloropleth-map-from-geojson-with-ggplot2.html (https://www.r-graph-gallery.com/327-chloropleth-map-from-geojson-with-ggplot2.html)

https://rpubs.com/jhofman/nycmaps (https://rpubs.com/jhofman/nycmaps)

https://rstudio.github.io/leaflet/ (https://rstudio.github.io/leaflet/)