DATA 606 Data Project Proposal

Justin WIlliams

```
library(tidycensus)
library(tidyverse)
library(tigris)
options(tigris_use_cache = TRUE)
library(RSocrata)
library(dotenv)
library(sf)
library(mapview)
```

Data Preparation

View different variables to choose from in latest available Census.

```
# variables available from 2020 decennial
(decennial_2020 <- load_variables(2020, "pl", cache = T))
```

```
## # A tibble: 301 x 3
##    name    label                                                    concept
##    <chr>   <chr>                                                    <chr>
##  1 H1_001N " !!Total:"                                              OCCUPA~
##  2 H1_002N " !!Total:!!Occupied"                                    OCCUPA~
##  3 H1_003N " !!Total:!!Vacant"                                      OCCUPA~
##  4 P1_001N " !!Total:"                                              RACE
##  5 P1_002N " !!Total:!!Population of one race:"                     RACE
##  6 P1_003N " !!Total:!!Population of one race:!!White alone"        RACE
##  7 P1_004N " !!Total:!!Population of one race:!!Black or African Americ~ RACE
##  8 P1_005N " !!Total:!!Population of one race:!!American Indian and Ala~ RACE
##  9 P1_006N " !!Total:!!Population of one race:!!Asian alone"        RACE
## 10 P1_007N " !!Total:!!Population of one race:!!Native Hawaiian and Oth~ RACE
## # ... with 291 more rows
```

```
# 2020 decennial didn't have income data, will have to use acs
(acs_5_2020 <- load_variables(2020, "acs5", cache = T))
```

```
## # A tibble: 27,850 x 3
##    name       label                                      concept
##    <chr>      <chr>                                      <chr>
##  1 B01001_001 Estimate!!Total:                           SEX BY AGE
##  2 B01001_002 Estimate!!Total:!!Male:                    SEX BY AGE
##  3 B01001_003 Estimate!!Total:!!Male:!!Under 5 years     SEX BY AGE
##  4 B01001_004 Estimate!!Total:!!Male:!!5 to 9 years      SEX BY AGE
##  5 B01001_005 Estimate!!Total:!!Male:!!10 to 14 years    SEX BY AGE
```

```
##  6 B01001_006 Estimate!!Total:!!Male:!!15 to 17 years  SEX BY AGE
##  7 B01001_007 Estimate!!Total:!!Male:!!18 and 19 years SEX BY AGE
##  8 B01001_008 Estimate!!Total:!!Male:!!20 years        SEX BY AGE
##  9 B01001_009 Estimate!!Total:!!Male:!!21 years        SEX BY AGE
## 10 B01001_010 Estimate!!Total:!!Male:!!22 to 24 years  SEX BY AGE
## # ... with 27,840 more rows
```

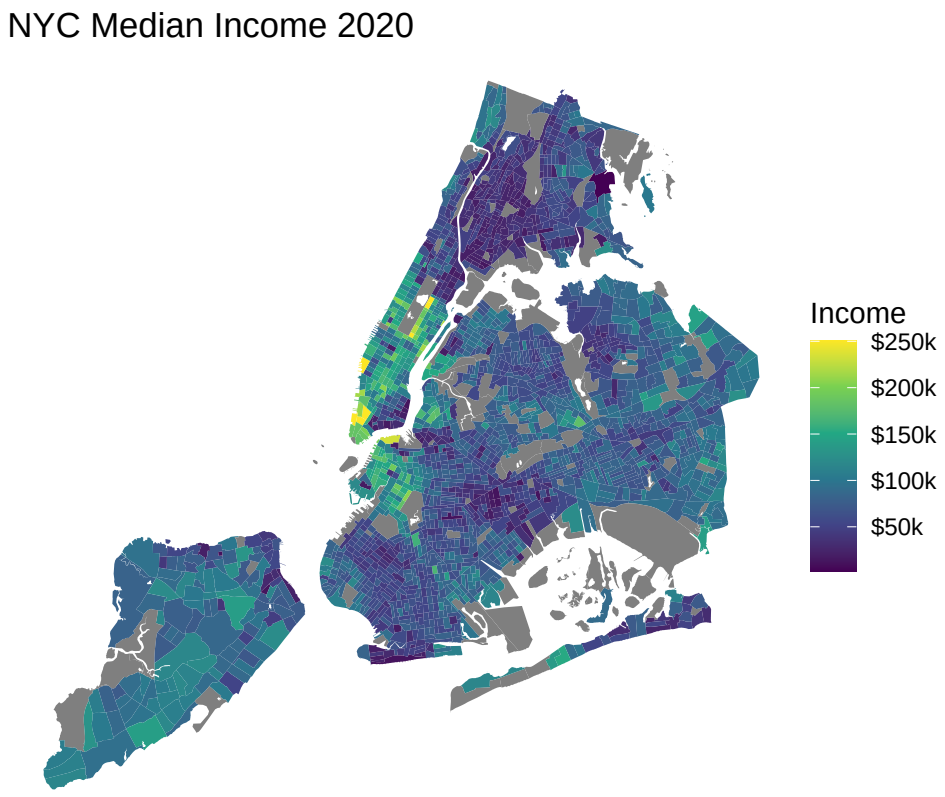
```
# preview and search for term
view(acs_5_2020)
```

We will be using `B19013_001` (median income) from the ACS 5-year 2016-2020 estimate

```
# pull in median income data from acs 2016 - 2020
nyc_median_income <- get_acs(geography = "tract",
        state = "New York",
        county = c("Bronx",
                   "Kings",
                   "New York",
                   "Queens",
                   "Richmond"),
        year = 2020,
        variables = "B19013_001",
        key = Sys.getenv("CENSUS_API"),
        geometry = T,
        cb = F) %>% # use TIGER/Line shapefiles
  st_transform(crs = 2263) %>% # transform crs to best for NYC
  erase_water(area_threshold = 0.75) #erase water from boundaries
```

Let's visualize Median Income across the 5 boroughs.

```
nyc_median_income %>%
  ggplot(aes(fill = estimate)) +
    geom_sf(color = NA) +
    scale_fill_viridis_c(option = "viridis",
                         label = scales::dollar_format(scale = .001,
                                                       prefix = "$",
                                                       suffix = "k")) +
    theme_void() +
    labs(title = "NYC Median Income 2020",
         caption = "Data source: ACS 5-year 2016-2020 estimate",
         fill = "Income")
```



Pull in food scrap site data using Socrata API.

```
# load food scrap data
food_scrap <-
  read.socrata(
    "https://data.cityofnewyork.us/resource/if26-z6xq.json",
    app_token = Sys.getenv("SOCRATA_API"))

#preview data
view(food_scrap)

# look at census tract columns
food_scrap %>%
  group_by(borough, ct2010) %>%
  summarise(count = n()) %>%
  arrange(desc(count))
```

```
## `summarise()` has grouped output by 'borough'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 57 x 3
## # Groups:   borough [6]
##    borough  ct2010 count
##    <chr>    <chr>  <int>
##  1 Kings    <NA>      50
##  2 New York <NA>      40
```

```
##  3 Queens   <NA>      36
##  4 Bronx    <NA>      24
##  5 Richmond <NA>       5
##  6 <NA>     <NA>       2
##  7 Bronx    133        1
##  8 Bronx    189        1
##  9 Bronx    265        1
## 10 Bronx    281        1
## # ... with 47 more rows
```

```
# sum missing values
sum(is.na(food_scrap$ct2010))
```

```
## [1] 157
```

157 missing values in `ct2010` Census Tract, will need to get these values in order to eventually join with Median Income data. We can use the `call_geolocator_latlon()` function from the **tigris** package to get Census Codes from point locations. It's a 15 digit number where the first 2-digit code are state, second 3-digit code state, the next 6-digit code for Census Tract. Therefore we can get a sub-string that equates to census tracts and fill in this missing data.

```
# many na in census tract, get census code through geolocator
food_scrap$census_code <-
  apply(food_scrap, 1, function(row) call_geolocator_latlon(
  row['latitude'], row['longitude']
))

# get census tracts from census code
food_scrap <- food_scrap %>%
  mutate(ct2010_2 = as.numeric(substr(census_code, 6, 11)))

# convert to simple features object to utilize geometry features
food_scrap_sf <- food_scrap %>%
  st_as_sf(coords = c("longitude", "latitude"),
           crs = 4326
  ) %>%
  st_transform(2263)

#preview
view(food_scrap_sf)
```

Save and then recall so I don't have to redo geolocator function (takes time).

```
saveRDS(food_scrap_sf, file = "./data/food_scrap_sf.rds")
food_scrap_sf <- readRDS("./data/food_scrap_sf.rds")
```

Let's visualize food scrap sites

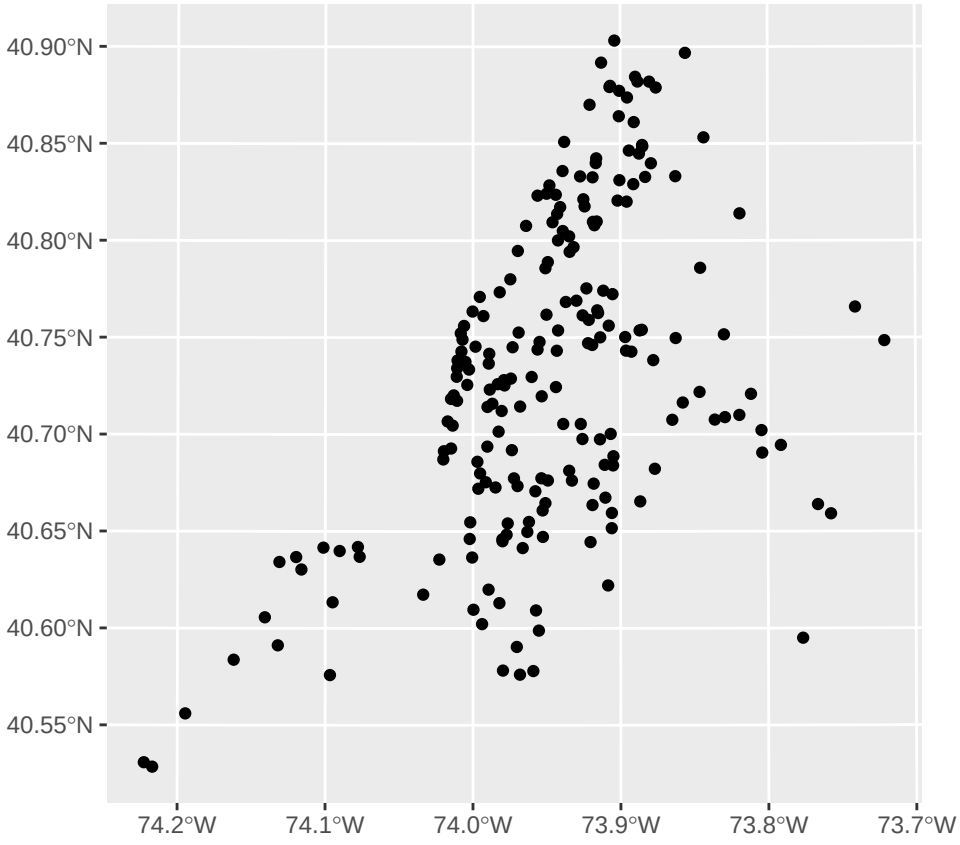
```
# drop point coordinates column
drop <- c("point.coordinates")
food_scrap_sf <- food_scrap_sf %>%
  select(-drop)
```

```
## Note: Using an external vector in selections is ambiguous.
## i Use ‘all_of(drop)‘ instead of ‘drop‘ to silence this message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

```
# preview food scrap sites
food_scrap_sf %>%
  mapview(
    col.regions = "red",
    legend = F
  )
```

```
## PhantomJS not found. You can install it with webshot::install_phantomjs(). If it is installed, pleas
```

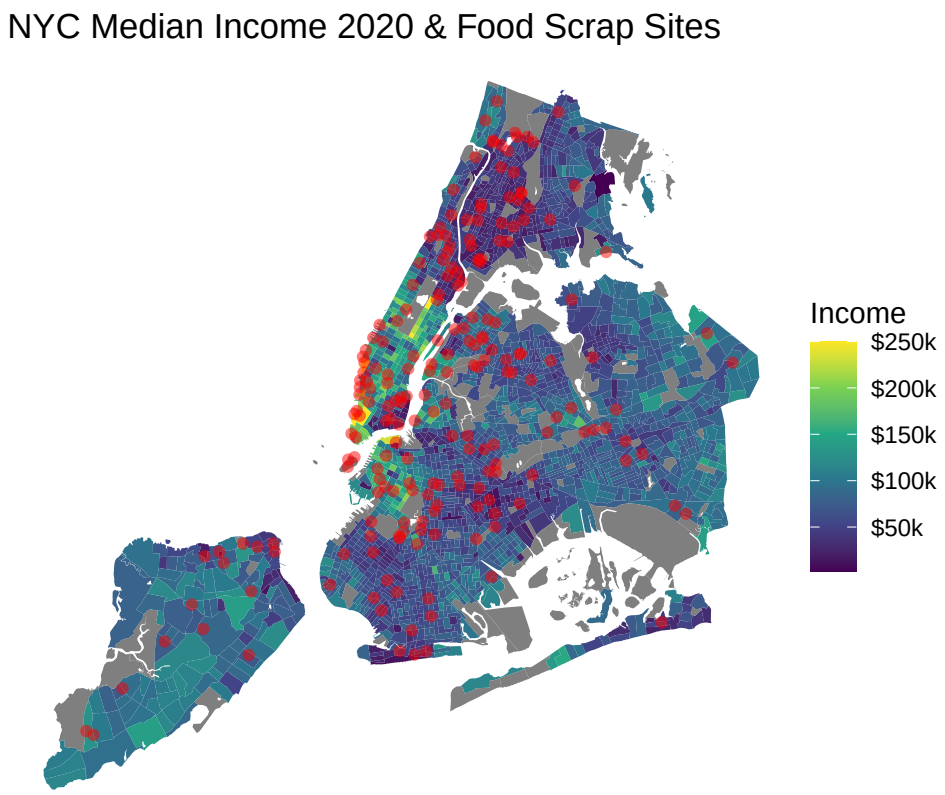
```
food_scrap_sf %>%
  ggplot() +
    geom_sf()
```



Let's look at this overlayed with the median income data.

```
ggplot() +
  geom_sf(data = nyc_median_income, color = NA, aes(fill = estimate)) +
    geom_sf(data = food_scrap_sf,
          color = "red",
          alpha = 0.5) +
```

```
  scale_fill_viridis_c(option = "viridis",
                       label = scales::dollar_format(scale = .001,
                                                     prefix = "$",
                                                     suffix = "k")) +
  theme_void() +
  labs(title = "NYC Median Income 2020 & Food Scrap Sites",
       caption = "Data source: ACS 5-year 2016-2020 estimate & DOS NYC Open Data",
       fill = "Income")
```



Ok, well we know we have a bunch throughout each borough, we need to aggregate by count for census tract and then join this to the median income data through Census Tracts. We will also create a boolean `food_scrap` which denotes whether a Census Tract has a food scrap composting site or not.

```
# aggreagte by borough
(food_scrap_borough <- food_scrap_sf %>%
  group_by(borough) %>%
  summarise(count = n()) %>%
  arrange(desc(count)))
```

```
## Simple feature collection with 6 features and 2 fields
## Geometry type: MULTIPOINT
## Dimension:     XY
## Bounding box:  xmin: 922387.6 ymin: 131898.4 xmax: 1061554 ymax: 268353.6
## Projected CRS: NAD83 / New York Long Island (ftUS)
## # A tibble: 6 x 3
```

```
##   borough  count                                                    geometry
##   <chr>    <int>                                        <MULTIPOINT [US_survey_foot]>
## 1 Kings       59 ((974893.2 164213.7), (977931 170812.1), (983635.9 174687), (9~
## 2 New York    55 ((978671.6 189627.2), (978788 191173.8), (979493.5 196768.1), ~
## 3 Queens      42 ((996386.9 210291.6), (996757.5 211712.6), (1000008 210061.5),~
## 4 Bronx       35 ((1004374 242828.3), (1004992 238526), (1005224 237190.9), (10~
## 5 Richmond    15 ((922387.6 132726.8), (923954.5 131898.4), (930167.4 141907.5)~
## 6 <NA>         2                       ((980642.3 201672.9), (994770.8 177882.6))
```

There are two missing values for borough, let's use the census code column imputed earlier to get the county code. Then we can fill in those missing values for county

```
food_scrap_sf <- food_scrap_sf %>%
  mutate(county_code = as.numeric(substr(census_code, start = 3, stop = 5)))

food_scrap_sf %>%
  group_by(borough, county_code) %>%
  summarise(count = n()) %>%
  arrange(desc(count))
```

```
## `summarise()` has grouped output by 'borough'. You can override using the
## `.groups` argument.
```

```
## Simple feature collection with 7 features and 3 fields
## Geometry type: GEOMETRY
## Dimension:     XY
## Bounding box:  xmin: 922387.6 ymin: 131898.4 xmax: 1061554 ymax: 268353.6
## Projected CRS: NAD83 / New York Long Island (ftUS)
## # A tibble: 7 x 4
## # Groups:   borough [6]
##   borough  county_code count                                          geometry
##   <chr>          <dbl> <int>                          <GEOMETRY [US_survey_foot]>
## 1 Kings             47    59 MULTIPOINT ((974893.2 164213.7), (977931 170812.1)~
## 2 New York          61    55 MULTIPOINT ((978671.6 189627.2), (978788 191173.8)~
## 3 Queens            81    42 MULTIPOINT ((996386.9 210291.6), (996757.5 211712.~
## 4 Bronx              5    35 MULTIPOINT ((1004374 242828.3), (1004992 238526), ~
## 5 Richmond          85    15 MULTIPOINT ((922387.6 132726.8), (923954.5 131898.~
## 6 <NA>              47     1                            POINT (994770.8 177882.6)
## 7 <NA>              61     1                            POINT (980642.3 201672.9)
```

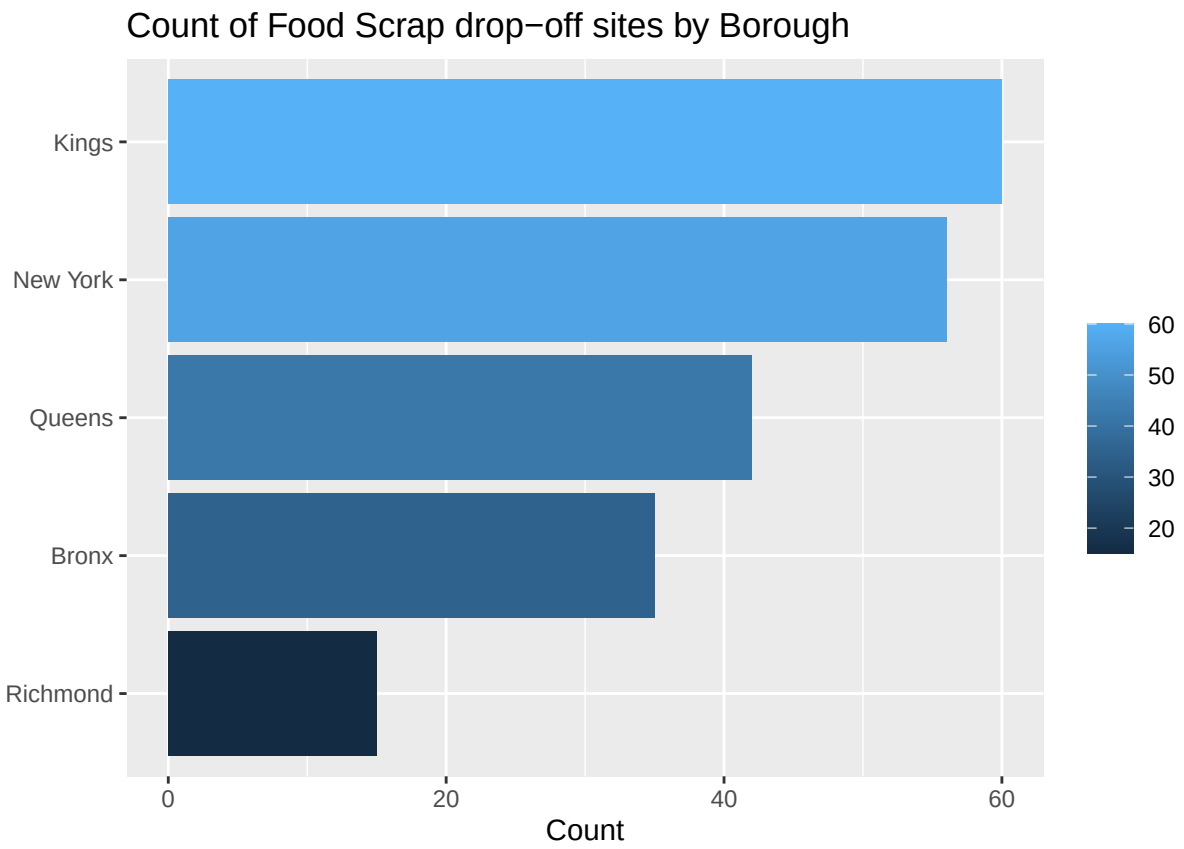
We can see the missing county codes are 047 Kings and 061 New York. Let's fill those in.

```
(food_scrap_borough <- food_scrap_sf %>%
  mutate(borough = ifelse(county_code == 047, "Kings",
                          ifelse(county_code == 061, "New York", borough))
) %>%
  group_by(borough) %>%
  summarise(count = n()) %>%
  arrange(desc(count)))
```

```
## Simple feature collection with 5 features and 2 fields
## Geometry type: MULTIPOINT
```

```
## Dimension:     XY
## Bounding box:  xmin: 922387.6 ymin: 131898.4 xmax: 1061554 ymax: 268353.6
## Projected CRS: NAD83 / New York Long Island (ftUS)
## # A tibble: 5 x 3
##   borough  count                                                  geometry
##   <chr>    <int>                                    <MULTIPOINT [US_survey_foot]>
## 1 Kings       60 ((974893.2 164213.7), (977931 170812.1), (983635.9 174687), (9~
## 2 New York    56 ((978671.6 189627.2), (978788 191173.8), (979493.5 196768.1), ~
## 3 Queens      42 ((996386.9 210291.6), (996757.5 211712.6), (1000008 210061.5),~
## 4 Bronx       35 ((1004374 242828.3), (1004992 238526), (1005224 237190.9), (10~
## 5 Richmond    15 ((922387.6 132726.8), (923954.5 131898.4), (930167.4 141907.5)~
```

```
food_scrap_borough %>%
  ggplot(aes(reorder(borough, count), count, fill = count)) +
    geom_bar(stat = "identity") +
    coord_flip() +
    labs(x="", y = "Count", fill = "", title = "Count of Food Scrap drop-off sites by Borough")
```



So we can see Kings County has the most sites followed by New York and Queens. Let's aggregate by borough and Census Tract.

```
# aggreate by borough and census tract
(food_scrap_ct <- food_scrap_sf %>%
  group_by(borough, ct2010_2) %>%
  summarise(count = n()) %>%
  arrange(desc(count)))
```

```
## `summarise()` has grouped output by 'borough'. You can override using the
## `.groups` argument.
```

```
## Simple feature collection with 196 features and 3 fields
## Geometry type: GEOMETRY
## Dimension:     XY
## Bounding box:  xmin: 922387.6 ymin: 131898.4 xmax: 1061554 ymax: 268353.6
## Projected CRS: NAD83 / New York Long Island (ftUS)
## # A tibble: 196 x 4
## # Groups:   borough [6]
##    borough  ct2010_2 count                                          geometry
##    <chr>       <dbl> <int>                       <MULTIPOINT [US_survey_foot]>
##  1 Bronx       37100     3 ((1015435 247117.7), (1016022 248731.7), (1016066 24~
##  2 New York      500     3 ((978671.6 189627.2), (978788 191173.8), (980137.3 1~
##  3 Bronx        3900     2             ((1006789 234285.8), (1007066 233652.6))
##  4 Bronx       28900     2             ((1009893 259624.8), (1010007 259844.2))
##  5 Bronx       40900     2             ((1013193 257657.8), (1015151 260659.5))
##  6 Kings       11902     2          ((985248.4 184132.2), (986668.8 185346.6))
##  7 New York     9901     2             ((982078 209908.6), (982278.8 212151.9))
##  8 New York    17800     2             ((1002410 228656.2), (1003155 229532.7))
##  9 Queens       5900     2             ((1004858 216710.3), (1006008 215861.6))
## 10 Richmond    22601     2          ((922387.6 132726.8), (923954.5 131898.4))
## # ... with 186 more rows
```

So it looks like no census tract has more then 3 food scrap drop-off sites. Let's join the data sets on census tract and then create a boolean column for whether the census tract has food scrap drop off site or not.

Research question

You should phrase your research question in a way that matches up with the scope of inference your dataset allows for.

Does income predict food scrap composting sites in NYC's 5 boroughs?

Or perhaps a different question which could involve more data from Census API such as:

What factors contribute to whether or not a census tract in NYC has a food scrap drop off site or not?

Additional variables could include, race, percent below poverty level, percent with vehicle etc...

Cases

What are the cases, and how many are there?

The cases are census tracts within NYC 5 boros.

```
cat("There are ", nrow(nyc_median_income), " cases in the dataset",sep = "")
```

```
## There are 2324 cases in the dataset
```

Data collection

Describe the method of data collection.

I used the most recent American Community Survey (ACS 2016 - 2020). The ACS is mailed annually to ~3.5 million households per year representing about 3 percent of the total US population). The 5-year which I will be utilizing in this project, is a moving average of data over a 5-year period that covers geographies down to Census block group. ACS are different from decennial Census data in that data represent estimates rather than precise counts, and as a result, are characterized by margins of error (moe) around estimates.

The Food Scrap drop-off data was provided by the NYC Department of Sanitation (DOS). I am uncertain of their collection methods.

Type of study

What type of study is this (observational/experiment)?

This is observational.

Data Source

If you collected the data, state self-collected. If not, provide a citation/link.

Census data was downloaded via the `tidycensus` package using the Census API. Food scrap data was downloaded through NYC Open Data using `RSocrata` through the Socrata API.

Response Variable

What is the response variable? Is it quantitative or qualitative?

The response variable is whether or not a Census Tract has a food scrap drop-off location and is categorical or boolean.

Explanatory Variable

You should have two independent variables, one quantitative and one qualitative.

The explanatory variable is mean median rent of NYC Census Tracts and is numerical.

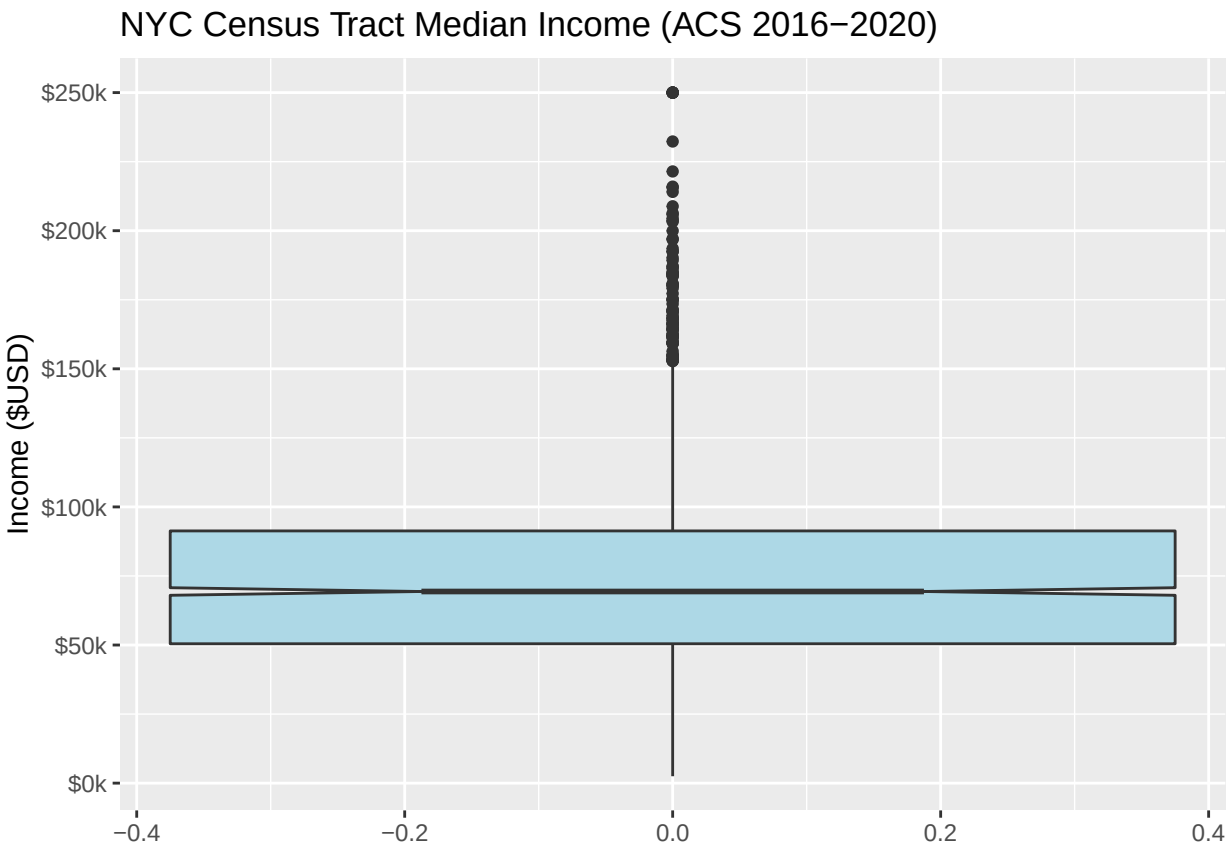
Relevant summary statistics

Provide summary statistics for each the variables. Also include appropriate visualizations related to your research question (e.g. scatter plot, boxplots, etc). This step requires the use of R, hence a code chunk is provided below. Insert more code chunks as needed.

Let's look at descriptive statistics of median income per census tract in NYC.

```
nyc_median_income %>%
  ggplot(aes(estimate)) +
    geom_boxplot(fill = "lightblue",
                 notch = T,
                 na.rm = T) +
    coord_flip() +
    scale_x_continuous(labels =
```

```
                          scales::dollar_format(
                            scale = .001,
                            prefix = "$",
                            suffix = "k")) +
    labs(x = "Income ($USD)", title = "NYC Census Tract Median Income (ACS 2016-2020)")
```



There were 118 rows with `NA`, so those were removed. Otherwise we can see median income per NYC Census Tract is ~$70k and the Interquartile Range is between spans ~50k - ~90k. Max outliers are upwards of ~250k and Min is very low.

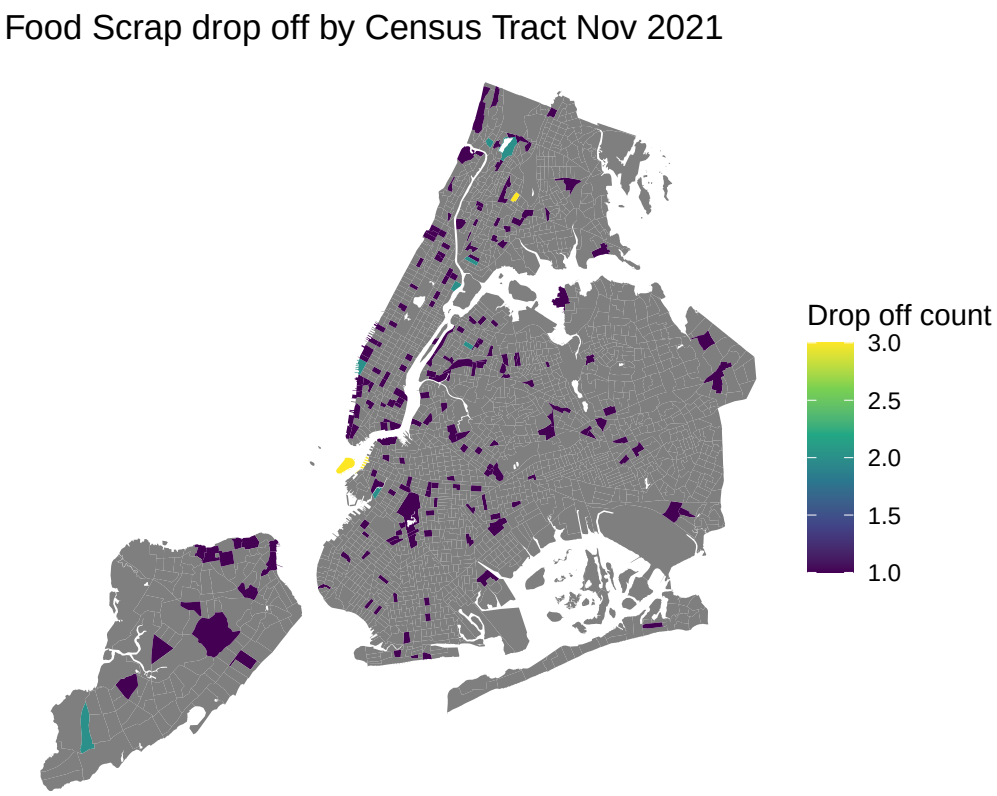
Let's take a look at distributions of food scrap drop off sites.

```
# create ct column for joining
nyc_median_income <- nyc_median_income %>%
  mutate(ct2010 = str_sub(string = GEOID,
                          start = 6))
# join ct and food scrap data.
merged_df <- nyc_median_income %>%
  st_join(food_scrap_ct)

# create bool column for food scrap
merged_df <- merged_df %>%
  mutate(food_scrap = ifelse(count != 0, 1, 0))

# visualize food scrap drop off
merged_df %>%
```

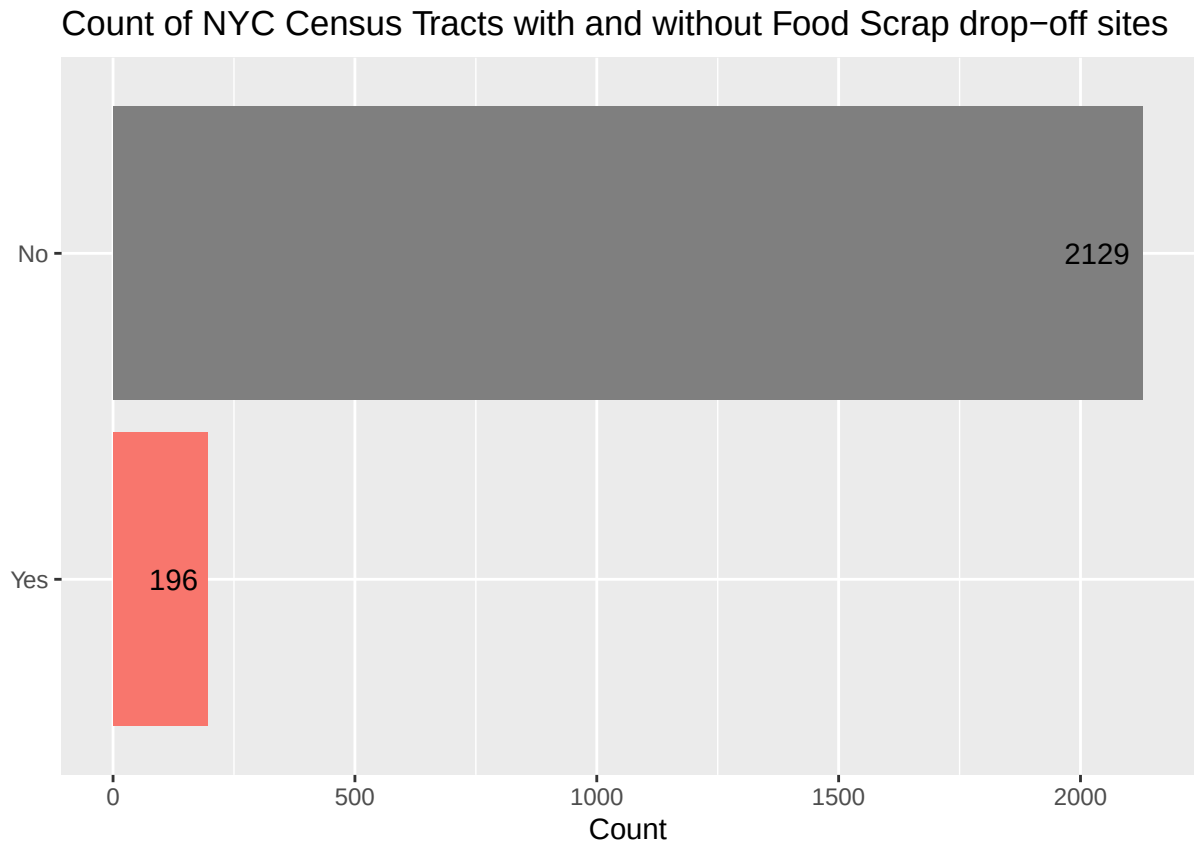
```
  ggplot(aes(fill = count)) +
  geom_sf(color = NA) +
  scale_fill_viridis_c(option = "viridis") +
  labs(title = "Food Scrap drop off by Census Tract Nov 2021",
       caption = "Data source: NYC Department of Sanitation",
       fill = "Drop off count") +
  theme_void()
```



We can see that most Census Tract's DO NOT have food scrap drop offs. Additionally, only a few have multiple food scrap drop offs.

Total census tracts with food drop off sites versus those that do not.

```
merged_df %>%
  group_by(food_scrap = as.factor(food_scrap)) %>%
  summarise(count = n()) %>%
  ggplot(aes(food_scrap, count, fill = food_scrap)) +
    geom_bar(stat = "identity") +
    coord_flip() +
    labs(x="", y="Count", title = "Count of NYC Census Tracts with and without Food Scrap drop-off sites
    scale_x_discrete(labels=c("Yes", "No")) +
    geom_text(aes(label = count, hjust=1.2)) +
    theme(legend.position = "none")
```



Imbalanced data set with 196 yes and 2129 no.

Next step would be to look at what percentage of census tracts within each borough have food scrap drop-off sites or not.