Appending Census Data

In this section we'll append Census metadata to our survey responses. The variables we'll include can be found on the NCCS census page.

Mapping survey respondents to Census Counties

The first step in this process is mapping census variables at the county level. We can accomplish this using the FCC's API.

```
# Function to access API and retrive county FIPS codes
census_api <- function(lat, lon, year, unit) {
  if(! lat %in% c(0, NA) & ! lon %in% c(0, NA)) {
    query_params <- list(
      latitude = as.character(lat),
      longitude = as.character(lon),
      censusYear = year,
      format = "json"
    )

    response <- httr::GET("https://geo.fcc.gov/api/census/area?",
                          query = query_params)

    body <- httr::content(response,
                          "parsed")
    if (unit == "Block"){
      rs <- body$Block$FIPS
    } else if (unit == "County"){
      rs <- body$County$FIPS
    }
    return(rs)
  } else {
    return(NA)
  }
}

# Create new column with County FIPs
survey_census_df <- survey_geocoded_df %>%
  dplyr::mutate(County_FIPS = purrr::map2_chr(Latitude,
```

```
                                    Longitude,
                                    census_api,
                                    year = "2010",
                                    unit = "County"))
```

Appending Census metaddata with crosswalks

With county FIPs appended, we can now join census metadata from NCCS' census files. The census variables found in these metadata files are listed in the table below:

variable_name	variable_description
year	Year of data
geoid	Geographic identifier
total_population	Total population
housing_units	Number of housing units
occupied	Number of occupied housing units
vacant	Number of vacant housing units
renter_occ	Number of renter occupied units
white_perc	Percent of population that is white
black_perc	Percent of population that is black
asian_perc	Percent of population that is asian
hawaiian_perc	Percent of population that is hawaiian
american_alaskan_perc	Percent of population that is american indican or alaskan native
two_or_more_perc	Percent of population with two or more races
other_perc	Percent of population with race classified as other
rural_perc	Percent of population living in rural areas
bachelors_perc	Percent of population 25 and over that have a bachelors degree or more
hispanic_perc	Percent of population that is hispanic
poverty_perc	Percent of population for whom poverty status is determined living in poverty
unemployment	Percent of population aged 16 or over and in labor force that are unemployed
turnover_perc	Percent of population that moved in the past year
med_family_income_adj	Median family income, inflation adjusted to 2021 dollars
med_gross_rent_adj	Median gross rent, inflation adjusted to 2021 dollars
med_household_income_adj	Median household income, inflation adjusted to 2021 dollars
median_value_adj	Median housing value, inflation adjusted to 2021 dollars

```r
# Download county-level census metadata
county_metadata_df <- readr::read_csv(
  "https://nccsdata.s3.us-east-1.amazonaws.com/geo/data/county/county_2017-2021.csv"
) %>%
  dplyr::mutate(
    geoid_2010 = as.character(geoid_2010)
  ) %>%
  dplyr::rename(
    "County_FIPS" = "geoid_2010",
    "Census_Year" = "year"
  ) %>%
  dplyr::select(-Census_Year) %>%
  dplyr::rename_at(vars(-County_FIPS), function(x)
    paste0(x, "_County"))

# Join census metadata
survey_census_df <- survey_census_df %>%
  tidylog::left_join(
    county_metadata_df,
    by = "County_FIPS"
  )
```

Appending FIPS Codes for Additional Geographic Units

We can extend this process to include metadata from additional geographic units. This requires pulling block FIPs codes and working with a crosswalk file.

We repeat the API call from above and append block FIPS codes to the data set.

```
# Create new column with Block FIPs
survey_census_df <- survey_census_df %>%
  dplyr::mutate(Block_FIPS = purrr::map2_chr(Latitude,
                                             Longitude,
                                             census_api,
                                             year = "2010",
                                             unit = "Block"))
```

We now use the NCCS crosswalk file to map these block FIPS codes to additional geographic units. We use the data.table package to process the data and extract the EINs.

```
# Load in Block crosswalk
block_xwalk_df <- data.table::fread(
  "https://nccsdata.s3.us-east-1.amazonaws.com/geo/xwalk/BLOCKX.csv"
)
# Convert into a lazy data frame
block_xwalk_lazy <- dtplyr::lazy_dt(block_xwalk_df)

# Process with tidyverse commands
block_xwalk_lazy <- block_xwalk_lazy %>%
  dplyr::mutate(Block_FIPS = as.character(block.census.geoid),
                Tract_FIPS = as.character(tract.census.geoid)) %>%
  dplyr::filter(Block_FIPS %in% unique(survey_census_df$Block_FIPS)) %>%
  dplyr::select(
    Block_FIPS,
    TRACT_FIPS
  )
```

```r
# Convert into a data.frame
block_xwalk_df <- as.data.frame(block_xwalk_lazy)
```

We now merge this crosswalk with NCCS' Census Tract crosswalk to include CBSA FIPs.

```r
tract_xwalk_df <- readr::read_csv(
  "https://nccsdata.s3.us-east-1.amazonaws.com/geo/xwalk/TRACTX.csv"
) %>%
  dplyr::rename(
    "Tract_FIPS" = "tract.census.geoid",
    "CBSA_FIPS" = "metro.census.cbsa10.geoid"
  ) %>%
  dplyr::mutate(
    CBSA_FIPS = as.character(CBSA_FIPS)
  ) %>%
  dplyr::select(
    Tract_FIPS,
    CBSA_FIPS
  )

# Merge both crosswalks together
xwalk_df <- block_xwalk_df %>%
  tidylog::left_join(
    tract_xwalk_df,
    by = "Tract_FIPS"
  )

# Print summary
summary(xwalk_df)
```

Appending Census Metadata at Tract and CBSA Levels

Tract-level Metadata

We can now repeat the same process as above but with census crosswalks at the tract and Metro CBSA levels using Tract FIPS and CBSA FIPS from our crosswalks.

```
# Download Tract-level census metadata
tract_metadata_df <- readr::read_csv(
  "https://nccsdata.s3.us-east-1.amazonaws.com/geo/data/tract/tract_2017-2021.csv"
) %>%
  dplyr::mutate(geoid = as.character(geoid)) %>%
  dplyr::rename("Tract_FIPS" = "geoid",
                "Census_Year" = "year") %>%
  dplyr::select(-Census_Year) %>% # We do not need Census Year
  rename_at(vars(-Tract_FIPS), function(x)
    paste0(x, "_Tract"))

# Join census metadata via crosswalk
survey_census_df <- survey_census_df %>%
  tidylog::left_join(
    xwalk_df,
    by = "Block_FIPS"
  ) %>%
  tidylog::left_join(
    tract_metadata_df,
    by = "Tract_FIPS"
  )
```

CBSA Level Metadata

```r
# Download CBSA-level census metadata
cbsa_metadata_df <- readr::read_csv(
  "https://nccsdata.s3.us-east-1.amazonaws.com/geo/data/msa/msa_2017-2021.csv"
) %>%
  dplyr::mutate(cbsa_code = as.character(cbsa_code)) %>%
  dplyr::rename("CBSA_FIPS" = "cbsa_code",
                "Census_Year" = "year") %>%
  dplyr::select(-Census_Year) %>%
  rename_at(vars(-CBSA_FIPS), function(x)
    paste0(x, "_CBSA"))

# Merge into survey dataset
survey_census_df <- survey_census_df %>%
  tidylog::left_join(
    cbsa_metadata_df,
    by = "CBSA_FIPS"
  )
```

Save data

```
setwd("Y:/CNP/Generosity Commission/")
readr::write_csv(
  survey_census_df,
  "DATA-PREP/02-data-intermediate/02-wave-two/wave-02-data-intermediate-census.csv"
)
```