

```
#####
# Appendix A2. R code used to validate self-reported census tracts and
# ZIP Code Tabulation Areas (ZCTAs)
#
# This script compares self-reported geographic identifiers with official
# 2024 TIGER/Line reference datasets from the U.S. Census Bureau, and creates
# clean geographic identifier columns (TIGER_CODE, TIGER_ZIP) for use in
# downstream analyses (e.g., joining ACS data).
#
# Validation is performed progressively: each step is only applied to
# respondents who did NOT already match in an earlier.
#####

##-----
## Load packages
##-----

library(tidyverse)
library(sf)
library(spatialEco)

##-----
## Read survey and reference datasets
##-----

survey      <- read_csv("data/Nevada mining_v4.csv")
census_tracts <- read_sf("data/CA_NV_CensusTracts.shp")
zip_codes    <- read_sf("data/tl_2024_us_zcta520.shp")
states       <- read_sf("data/tl_2024_us_state.shp")

# Full U.S. ZCTA dataset (not clipped to CA/NV).
# Used for ZIP validation below, since a mistyped ZIP could plausibly
# resolve to a ZCTA outside California/Nevada.
zip_codes_all <- st_read("data/tl_2024_us_zcta520.shp")

##-----
## Rename survey variables
##
## Q62 = self-reported census tract
## Q64 = self-reported ZIP code
##-----

survey <- survey %>%
  rename(
    reported_tract = Q62,
    reported_zip   = Q64
  ) %>%
  mutate(
    reported_tract = as.character(reported_tract),
    reported_zip   = as.character(reported_zip)
  )

##-----
## Prepare California and Nevada state boundary
##-----

states_ca_nv <- states %>%
  filter(STUSPS %in% c("CA", "NV"))

ca_nv_boundary <- st_union(states_ca_nv)
plot(states_ca_nv[1])

##-----
## Clip ZIP Code Tabulation Areas to California and Nevada
##
## Retained for mapping/plotting purposes. NOT used for ZIP validation
```

```

## below (see zip_codes_all).
##-----

zip_codes <- st_intersection(
  zip_codes,
  ca_nv_boundary
)

#####
# PART 1: CENSUS TRACT VALIDATION
#####

##-----
## Step 1. Exact match to TIGER NAME field
##-----

match_step1 <- survey$reported_tract %in% census_tracts$NAME
idx_step1 <- match(survey$reported_tract, census_tracts$NAME)

exact_matches <- sum(match_step1, na.rm = TRUE)
exact_percent <- exact_matches / nrow(survey) * 100
exact_percent
##-----
## Step 2. Remove leading zeros and compare to NAME
## Example: 0032.06 -> 32.06
## Only applied to respondents who did not match in Step 1.
##-----

reported_tract_nozeros <- str_replace(survey$reported_tract, "^0+", "")

match_step2 <- !match_step1 &
  (reported_tract_nozeros %in% census_tracts$NAME)
idx_step2 <- match(reported_tract_nozeros, census_tracts$NAME)

leading_zero_matches <- sum(match_step2, na.rm = TRUE)
leading_zero_percent <- leading_zero_matches / nrow(survey) * 100
leading_zero_percent
##-----
## Step 3. Remove decimal point and compare to TRACTCE
## Example: 0032.06 -> 003206
## Only applied to respondents who did not match in Step 1 or Step 2.
##-----

reported_tract_code <- str_remove_all(survey$reported_tract, "\\.")

match_step3 <- !match_step1 & !match_step2 &
  (reported_tract_code %in% census_tracts$TRACTCE)
idx_step3 <- match(reported_tract_code, census_tracts$TRACTCE)

decimal_removed_matches <- sum(match_step3, na.rm = TRUE)
decimal_removed_percent <- decimal_removed_matches / nrow(survey) * 100
decimal_removed_percent
##-----
## Summary table: cumulative tract match rates
##-----

tract_match_summary <- tibble(
  Validation = c(
    "Exact match to TIGER tract number",
    "After removing leading zeros",
    "After removing decimal point"
  ),
  Matches = c(
    exact_matches,
    exact_matches + leading_zero_matches,

```

```

    exact_matches + leading_zero_matches + decimal_removed_matches
  ),
  Percent = round(
    c(
      exact_percent,
      exact_percent + leading_zero_percent,
      exact_percent + leading_zero_percent + decimal_removed_percent
    ),
    1
  )
)

tract_match_summary

##-----
## Add match type and official TIGER identifiers back into survey
##-----

survey <- survey %>%
  mutate(
    tract_match_type = case_when(
      match_step1 ~ "Exact TIGER NAME match",
      match_step2 ~ "Leading zeros removed",
      match_step3 ~ "Decimal removed",
      TRUE ~ "No match"
    )
  )

survey$TIGER_NAME <- NA_character_
survey$TIGER_CODE <- NA_character_

survey$TIGER_NAME[match_step1] <- census_tracts$NAME[idx_step1[match_step1]]
survey$TIGER_CODE[match_step1] <- census_tracts$TRACTCE[idx_step1[match_step1]]

survey$TIGER_NAME[match_step2] <- census_tracts$NAME[idx_step2[match_step2]]
survey$TIGER_CODE[match_step2] <- census_tracts$TRACTCE[idx_step2[match_step2]]

survey$TIGER_NAME[match_step3] <- census_tracts$NAME[idx_step3[match_step3]]
survey$TIGER_CODE[match_step3] <- census_tracts$TRACTCE[idx_step3[match_step3]]

## Sanity check: table counts should equal tract_match_summary above
table(survey$tract_match_type)

#####
# PART 2: ZIP CODE VALIDATION
#
# ASSUMPTION normalization proceeds
# Step 1: exact match
# Step 2: restore a dropped leading zero (e.g., "2138" -> "02138")
# Step 3: strip a ZIP+4 suffix (e.g., "89701-1234" -> "89701")
# As with tracts, each step only applies to respondents not already matched.
#####

##-----
## Step 1. Exact match to ZCTA5CE20
##-----

zip_match_step1 <- survey$reported_zip %in% zip_codes_all$ZCTA5CE20
zip_idx_step1 <- match(survey$reported_zip, zip_codes_all$ZCTA5CE20)

zip_exact_matches <- sum(zip_match_step1, na.rm = TRUE)
zip_exact_percent <- zip_exact_matches / nrow(survey) * 100
zip_exact_percent
##-----
## Step 2. Restore dropped leading zero (pad to 5 digits)

```

```

## Example: 2138 -> 02138
## Only applied to respondents who did not match in Step 1.
##-----

reported_zip_padded <- str_pad(survey$reported_zip, width = 5, pad = "0")

zip_match_step2 <- !zip_match_step1 &
  (reported_zip_padded %in% zip_codes_all$ZCTA5CE20)
zip_idx_step2 <- match(reported_zip_padded, zip_codes_all$ZCTA5CE20)

zip_leading_zero_matches <- sum(zip_match_step2, na.rm = TRUE)
zip_leading_zero_percent <- zip_leading_zero_matches / nrow(survey) * 100
zip_leading_zero_percent
##-----
## Step 3. Strip ZIP+4 suffix
## Example: 89701-1234 -> 89701
## Only applied to respondents who did not match in Step 1 or Step 2.
##-----

reported_zip_5digit <- str_sub(
  str_remove(survey$reported_zip, "-.*$"),
  1, 5
)

zip_match_step3 <- !zip_match_step1 & !zip_match_step2 &
  (reported_zip_5digit %in% zip_codes_all$ZCTA5CE20)
zip_idx_step3 <- match(reported_zip_5digit, zip_codes_all$ZCTA5CE20)

zip_plus4_matches <- sum(zip_match_step3, na.rm = TRUE)
zip_plus4_percent <- zip_plus4_matches / nrow(survey) * 100
zip_plus4_percent
##-----
## Summary table: cumulative ZIP match rates
##-----

zip_match_summary <- tibble(
  Validation = c(
    "Exact match to ZCTA",
    "After restoring leading zero",
    "After removing ZIP+4 suffix"
  ),
  Matches = c(
    zip_exact_matches,
    zip_exact_matches + zip_leading_zero_matches,
    zip_exact_matches + zip_leading_zero_matches + zip_plus4_matches
  ),
  Percent = round(
    c(
      zip_exact_percent,
      zip_exact_percent + zip_leading_zero_percent,
      zip_exact_percent + zip_leading_zero_percent + zip_plus4_percent
    ),
    1
  )
)

zip_match_summary
##-----
## Add match type and official ZCTA identifier back into survey
##-----

survey <- survey %>%
  mutate(
    zip_match_type = case_when(

```

```

    zip_match_step1 ~ "Exact ZCTA match",
    zip_match_step2 ~ "Leading zero restored",
    zip_match_step3 ~ "ZIP+4 suffix removed",
    TRUE ~ "No match"
  )
)

survey$TIGER_ZIP <- NA_character_

survey$TIGER_ZIP[zip_match_step1] <-
  zip_codes_all$ZCTA5CE20[zip_idx_step1[zip_match_step1]]
survey$TIGER_ZIP[zip_match_step2] <-
  zip_codes_all$ZCTA5CE20[zip_idx_step2[zip_match_step2]]
survey$TIGER_ZIP[zip_match_step3] <-
  zip_codes_all$ZCTA5CE20[zip_idx_step3[zip_match_step3]]

## Sanity check: table counts should equal zip_match_summary above
table(survey$zip_match_type)

#####
# PART 3: SUMMARY BY DEMOGRAPHIC VARIABLES
#
# tract_match / zip_match are TRUE/FALSE flags for "matched at any step"
# (i.e., match_type != "No match").
#
#####

survey <- survey %>%
  mutate(
    tract_match = tract_match_type != "No match",
    zip_match    = zip_match_type != "No match"
  )

##-----
## Label demographic variables using the survey codebook
##-----

survey <- survey %>%
  mutate(
    age_group = factor(Q50, levels = 1:6, labels = c(
      "18-24", "25-34", "35-44", "45-54", "55-64", "65 or older"
    )),
    gender = factor(Q51, levels = 1:3, labels = c(
      "Male", "Female", "Non-binary / third gender"
    )),
    race = factor(Q52, levels = 1:7, labels = c(
      "White/Caucasian", "Black or African American",
      "Asian or Asian American", "Native American or Alaska Native",
      "Native Hawaiian or Pacific Islander", "Other", "Multi-racial"
    )),
    hispanic_origin = factor(Q53, levels = 1:2, labels = c(
      "Hispanic", "Not Hispanic"
    )),
    education = factor(Q54, levels = 1:5, labels = c(
      "Less than high school", "High school graduate or equivalent",
      "Some college or Associate degree", "Bachelor's degree",
      "Graduate or professional degree"
    )),
    employment = factor(Q55, levels = 1:8, labels = c(
      "Employed full-time", "Employed part-time", "Self-employed",
      "Unemployed, looking for work", "Unemployed, not looking for work",
      "Retired", "Student", "Other"
    )),
    income = factor(Q56, levels = 1:6, labels = c(
      "Less than $25,000", "$25,000-$49,999", "$50,000-$74,999",

```

```

"$75,000-$99,999", "$100,000-$149,999", "$150,000 or more"
)),
party_lean = factor(Q57, levels = 1:5, labels = c(
  "Strongly Republican", "Lean Republican", "Independent",
  "Lean Democratic", "Strongly Democratic"
)),
ideology = factor(Q58, levels = 1:5, labels = c(
  "Strongly conservative", "Conservative", "Moderate",
  "Liberal", "Strongly liberal"
)),
mining_household = factor(Q59, levels = 1:3, labels = c(
  "Yes, currently employed", "Yes, previously employed", "No"
))
)

##-----
## Simple helper to reduce repetition below.
## Not a normalization/matching function -- just group_by() + summarise(),
## kept transparent so each output table is easy to inspect on its own.
##-----

summarize_match_by <- function(data, group_var) {
  data %>%
    group_by({{ group_var }}) %>%
    summarise(
      n = n(),
      tract_match_rate = round(mean(tract_match) * 100, 1),
      zip_match_rate = round(mean(zip_match) * 100, 1),
      .groups = "drop"
    )
}

##-----
## Match rate by each demographic variable
## (each is its own inspectable object -- print individually as needed)
##-----

match_by_age <- summarize_match_by(survey, age_group)
match_by_gender <- summarize_match_by(survey, gender)
match_by_race <- summarize_match_by(survey, race)
match_by_hispanic <- summarize_match_by(survey, hispanic_origin)
match_by_education <- summarize_match_by(survey, education)
match_by_employment <- summarize_match_by(survey, employment)
match_by_income <- summarize_match_by(survey, income)
match_by_party <- summarize_match_by(survey, party_lean)
match_by_ideology <- summarize_match_by(survey, ideology)
match_by_mining <- summarize_match_by(survey, mining_household)

match_by_age
match_by_gender
match_by_race
match_by_hispanic
match_by_education
match_by_employment
match_by_income
match_by_party
match_by_ideology
match_by_mining

#####
# PART 4: SAVE VALIDATED SURVEY DATASET
#####

write_csv(
  survey,

```

```
"clean_data/survey_location_validation.csv"
)

#####

# End of script
#####
```