Code - A bit more organized

Dylan Koproski

2025-01-28

Libraries

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(jsonlite)
```

```
##
## Attaching package: 'jsonlite'
##
## The following object is masked from 'package:purrr':
##
##     flatten
```

```r
library(readxl)
library(rsample)
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(VGAM)
```

```
## Loading required package: stats4
## Loading required package: splines
##
## Attaching package: 'VGAM'
##
## The following object is masked from 'package:caret':
```

```
##
##     predictors
```

```r
library(COMPoissonReg)
```

```
## Loading required package: Rcpp
##
## Attaching package: 'Rcpp'
##
## The following object is masked from 'package:rsample':
##
##     populate
##
## Loading required package: numDeriv
##
## Attaching package: 'COMPoissonReg'
##
## The following object is masked from 'package:VGAM':
##
##     get.offset
```

```r
library(pscl)
```

```
## Classes and Methods for R originally developed in the
## Political Science Computational Laboratory
## Department of Political Science
## Stanford University (2002-2015),
## by and under the direction of Simon Jackman.
## hurdle and zeroinfl functions by Achim Zeileis.
```

```r
library(lme4)
```

```
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```r
library(zipcodeR)
library(maps)
```

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

```r
library(MASS)
```

```
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:dplyr':
##
##     select
```

```r
library(usmap)
library(scales)
```

```
##
## Attaching package: 'scales'
##
## The following object is masked from 'package:purrr':
##
##     discard
##
## The following object is masked from 'package:readr':
##
##     col_factor
```

```r
conflicted::conflict_prefer("select", "dplyr")
```

```
## [conflicted] Will prefer dplyr::select over any other package.
```

```r
conflicted::conflict_prefer("map", "purrr")
```

```
## [conflicted] Will prefer purrr::map over any other package.
```

```r
conflicted::conflict_prefer("filter", "dplyr")
```

```
## [conflicted] Will prefer dplyr::filter over any other package.
```

Preprocessing

```r
# Load data
df_visitor = read_csv("mobility.csv")
```

```
## Warning: One or more parsing issues, call `problems()` on your data frame for details,
## e.g.:
##   dat <- vroom(...)
##   problems(dat)
```

```
## Rows: 24583 Columns: 52
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (30): placekey, parent_placekey, safegraph_brand_ids, location_name, br...
## dbl  (14): naics_code, latitude, longitude, phone_number, wkt_area_sq_meters...
## lgl   (3): enclosed, is_synthetic, includes_parking_lot
## dttm  (2): date_range_start, date_range_end
## date  (3): opened_on, closed_on, tracking_closed_since
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
df_census = read_csv("tract_census.csv", skip = 1) |>
  janitor::clean_names()
```

```
## New names:
## * `` -> `...459`
```

```
## Warning: One or more parsing issues, call `problems()` on your data frame for details,
## e.g.:
##   dat <- vroom(...)
```

```
##   problems(dat)
```

```
## Rows: 85395 Columns: 459
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (272): Geography, Geographic Area Name, Estimate!!Total!!Total populatio...
## dbl (186): Estimate!!Total!!Total population, Margin of Error!!Total!!Total ...
## lgl   (1): ...459
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
df_tract_zip = read_excel("tract_zip.xlsx")



# Temporary restriction to NYC
state_county_code_str = c(36005, 36047, 36061, 36081, 36085)

# Process visitor data - none missing
filtered_df_visitor = df_visitor |>
  mutate(first_five_digits = substr(poi_cbg, 1, 5)) |>
  filter(first_five_digits %in% state_county_code_str) |>
  # Dropping missing values for home cbg
  filter(!is.na(visitor_home_aggregation)) |>
  mutate(identifier = row_number()) |>
  mutate(poi_zip = postal_code) |>
  mutate(visitor_home_aggregation = map(visitor_home_aggregation, ~fromJSON(as.character(.)))) |>
  select(location_name, date_range_start, date_range_end, visitor_home_aggregation, top_category, identi
  unnest_longer(visitor_home_aggregation) |>
  rename(visitor_census_tract = visitor_home_aggregation_id, visitors = visitor_home_aggregation) |>
  mutate(visitors = if_else(visitors == 4, 3, visitors)) |>
  mutate(poi_lat = latitude,
         poi_long = longitude)

# Census data processing
df_census = df_census |>
  rowwise() |>
  mutate(cbg = str_sub(geography, -11)) |>
#873 locations have an estimated 0 people, we should exclude these.
  filter(estimate_total_total_population > 0)

# Age group proportions in census data, separate into 3 age groups
filtered_df_census_totals =
  df_census |>
  rowwise() |>
  select(estimate_total_total_population, cbg, geographic_area_name, starts_with("estimate")) |>
  mutate(
    total_under_18 = sum(estimate_total_total_population_age_under_5_years,
                         estimate_total_total_population_age_5_to_9_years,
                         estimate_total_total_population_age_10_to_14_years,
                         estimate_total_total_population_age_15_to_19_years) / estimate_total_total_popu

    total_19_65 = sum(estimate_total_total_population_age_20_to_24_years,
                      estimate_total_total_population_age_25_to_29_years,
```

```
                      estimate_total_total_population_age_30_to_34_years,
                      estimate_total_total_population_age_35_to_39_years,
                      estimate_total_total_population_age_40_to_44_years,
                      estimate_total_total_population_age_45_to_49_years,
                      estimate_total_total_population_age_50_to_54_years,
                      estimate_total_total_population_age_55_to_59_years,
                      estimate_total_total_population_age_60_to_64_years,
                      estimate_total_total_population_age_65_to_69_years) / estimate_total_total_popula

    total_65_plus = sum(estimate_total_total_population_age_70_to_74_years,
                        estimate_total_total_population_age_75_to_79_years,
                        estimate_total_total_population_age_80_to_84_years,
                        estimate_total_total_population_age_85_years_and_over) / estimate_total_total_po
  ) |>
  rename("total" = estimate_total_total_population) |>
  select(cbg, geographic_area_name, total, total_under_18, total_19_65, total_65_plus)

# Define primary ZIP code per census tract
primary_tract_zip = df_tract_zip |>
  group_by(tract) |>
  summarize(zip = min(zip))  # Selects the minimum ZIP as primary for simplicity

# Merge filtered_df_census_totals with filtered_df_visitor
merged_df = filtered_df_visitor |>
  inner_join(filtered_df_census_totals, by = c("visitor_census_tract" = "cbg")) |>
  mutate(
    visitors_under_18 = visitors * total_under_18,
    visitors_19_65 = visitors * total_19_65,
    visitors_65_plus = visitors * total_65_plus
  )

# Map primary ZIP codes by merging with primary_tract_zip on the census tract
final_df = merged_df |>
  left_join(primary_tract_zip, by = c("visitor_census_tract" = "tract")) |>
  select(location_name, date_range_start, date_range_end, top_category,
         identifier, poi_cbg, poi_zip, visitors, visitors_under_18,
         visitors_19_65, visitors_65_plus, zip, poi_long, poi_lat) |>
  mutate(visitor_zip = zip)

vis_zip_lat_long = geocode_zip(final_df$visitor_zip)

final_df = final_df |>
  left_join(vis_zip_lat_long, by = join_by(visitor_zip == zipcode)) |>
  mutate(vis_lat = lat,
         vis_long = lng) |>
  select(-lat, -lng)

# Rounding, can be adjusted at will
final_df = final_df |>
  mutate(visitors_under_18 = ceiling(visitors_under_18),
         visitors_19_65 = ceiling(visitors_19_65),
         visitors_65_plus = ceiling(visitors_65_plus)) |>
  mutate(total_visitors = visitors_under_18 + visitors_19_65 + visitors_65_plus)
```

```r
# There are 2 rows in the above data that have missing values, the visitor zip is missing. We should fi

# Long dataframe for modelling purposes
df_long = final_df |>
  pivot_longer(
    cols = starts_with("visitors_"),
    names_to = "age_group",
    names_prefix = "visitors_",
    values_to = "visitor_count"
  ) |>
  mutate(top_category = factor(top_category),
         age_group = factor(age_group, levels = c("under_18", "19_65", "65_plus")))

# Data quality looks good, 2 missing zip codes may need to be handled, but data is large enough maybe d
```

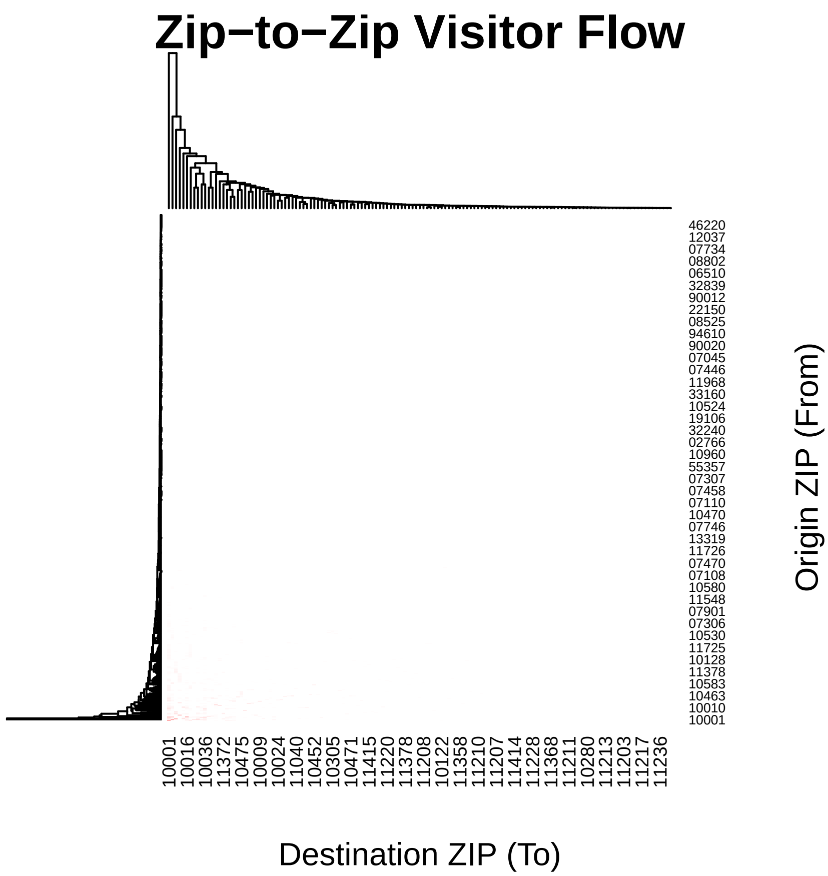
Exploratory Data Analysis (EDA)

Zip Code Flow Matrix

```r
# Remove rows with NA in visitor_zip or poi_zip
zip_matrix = final_df |>
  filter(!is.na(visitor_zip) & !is.na(poi_zip)) |>
  group_by(visitor_zip, poi_zip) |>
  summarize(total_visitors = sum(visitors, na.rm = TRUE)) |>
  pivot_wider(names_from = poi_zip, values_from = total_visitors, values_fill = 0)
```

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

```r
# Convert to matrix and plot
zip_matrix_plot = zip_matrix |>
  column_to_rownames("visitor_zip") |>
  as.matrix() |>
  heatmap(
    col = colorRampPalette(c("white", "red"))(100),
    scale = "none",
    main = "Zip-to-Zip Visitor Flow",
    xlab = "Destination ZIP (To)",
    ylab = "Origin ZIP (From)"
  )
```



State to Destination ZIP in NYC

```
final_df_with_state = final_df |>
  left_join(df_tract_zip, by = c("visitor_zip" = "zip")) |>
  select(!zip) |>
  rename(visitor_state = usps_zip_pref_state)
```

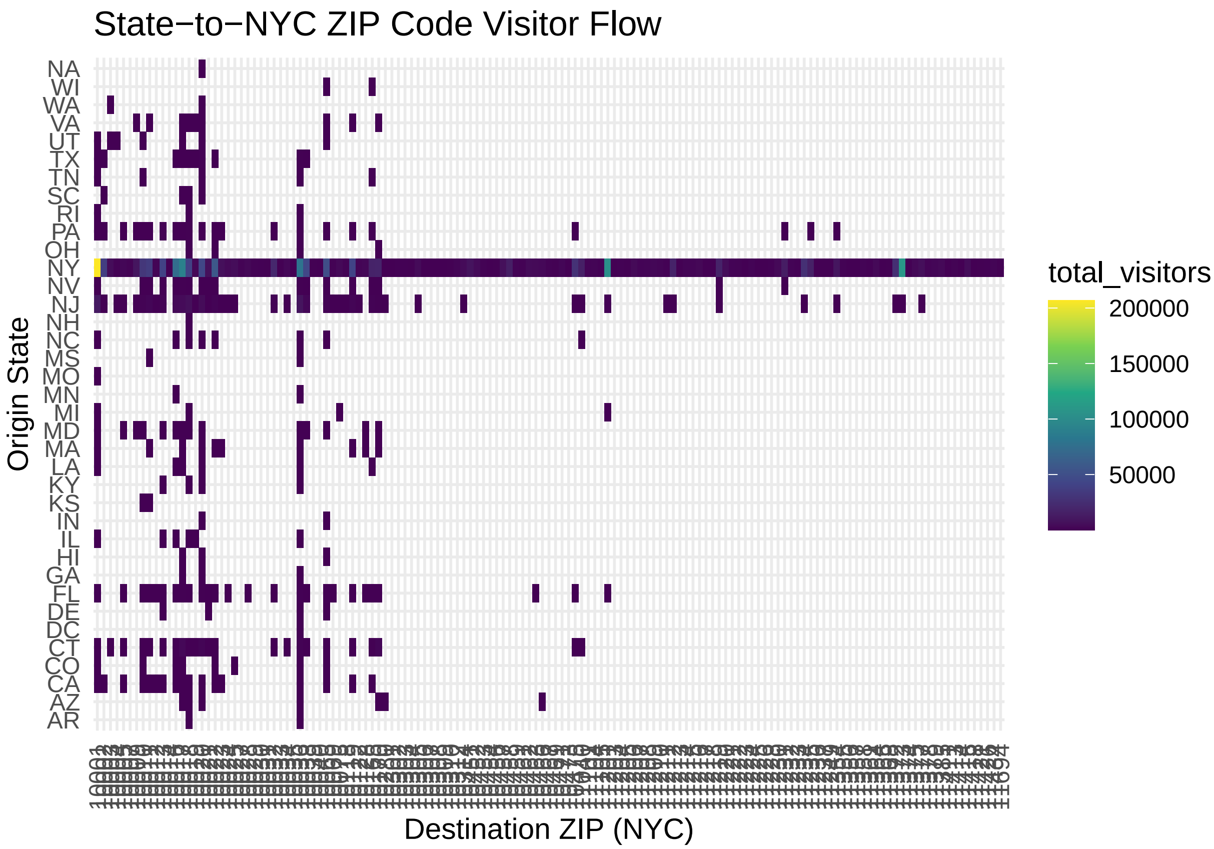
```
## Warning in left_join(final_df, df_tract_zip, by = c(visitor_zip = "zip")): Detected an unexpected man
## i Row 1 of `x` matches multiple rows in `y`.
## i Row 103268 of `y` matches multiple rows in `x`.
## i If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```
state_zip_matrix = final_df_with_state |>
  group_by(visitor_state, poi_zip) |>
  summarize(total_visitors = sum(visitors, na.rm = TRUE))
```

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

```
ggplot(state_zip_matrix, aes(x = poi_zip, y = visitor_state, fill = total_visitors)) +
  geom_tile() +
  scale_fill_viridis_c() +
  labs(
    title = "State-to-NYC ZIP Code Visitor Flow",
    x = "Destination ZIP (NYC)",
    y = "Origin State"
```

```
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5))
```



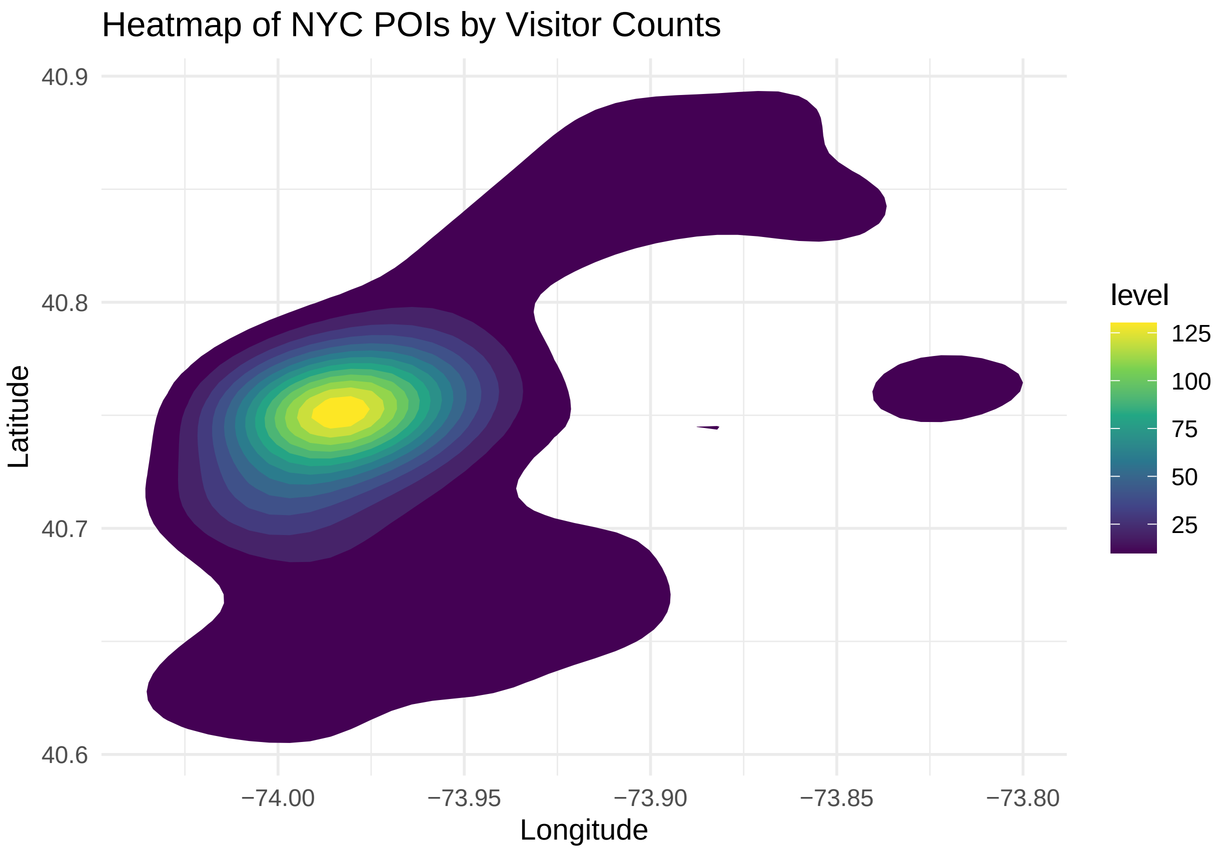
Visualizations

Heatmap - no basemap, just blobs. This can be improved

```
nyc_zip_visitors = final_df |>
  group_by(poi_long, poi_lat) |>
  summarize(total_visitors = sum(visitors, na.rm = TRUE))
```

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

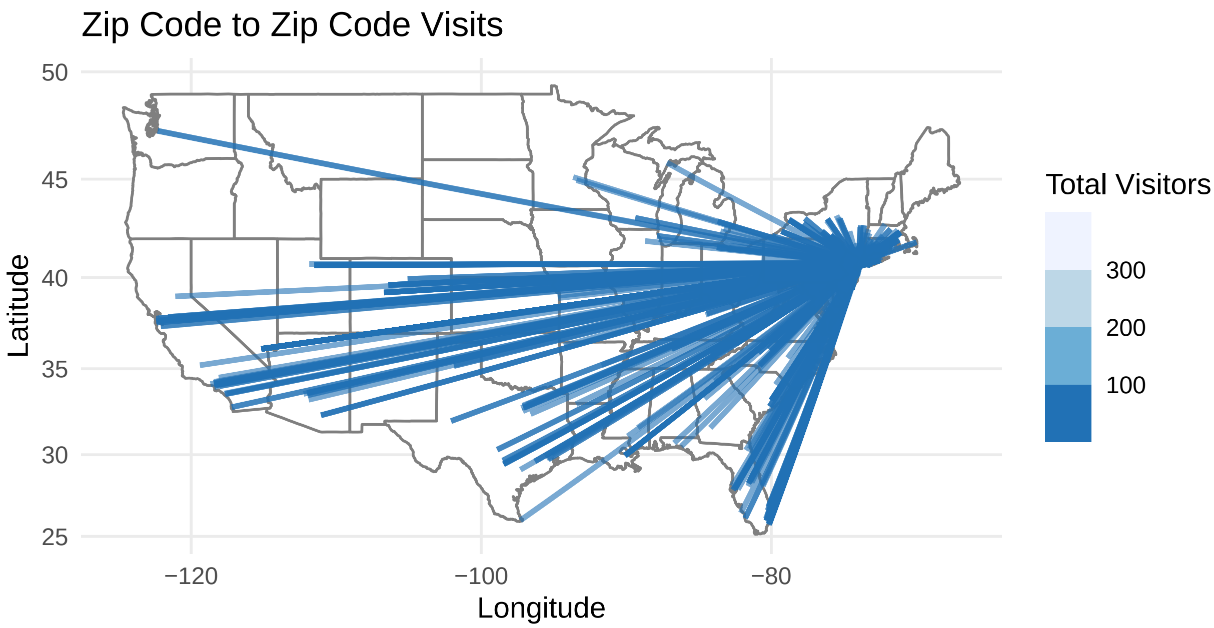
```
ggplot(nyc_zip_visitors, aes(x = poi_long, y = poi_lat)) +
  stat_density_2d(aes(fill = after_stat(level)), geom = "polygon", color = NA) +
  scale_fill_viridis_c() +
  labs(
    title = "Heatmap of NYC POIs by Visitor Counts",
    x = "Longitude",
    y = "Latitude"
  ) +
  theme_minimal()
```



Flow map - work in progress

```
#excise visitors from hawaii, include the second filter argument to get a better look at lower ends of
flow_df = final_df |>
  filter(vis_long > -150) #|> filter(total_visitors < 15)

# if you want to include visitors from hawaii
# flow_df = final_df

usa = map_data("state")

usa = rename(usa, state = "region")

usa$state = str_to_title(usa$state)

stateData = usa |>
  arrange(state, group, order)

ggplot() +
  geom_polygon(data = stateData,
               aes(x = long, y = lat, group = group),
               fill = "white", color = "gray50") +

  geom_segment(data = flow_df,
               aes(x = vis_long, y = vis_lat,
                   xend = poi_long, yend = poi_lat,
```

```r
                    color = total_visitors),
                alpha = 0.6, linewidth = 1) +
  scale_color_fermenter(name = "Total Visitors", direction = -1) +
  coord_map() +
  theme_minimal() +
  labs(color = "Volume of Visits",
       title = "Zip Code to Zip Code Visits",
       x = "Longitude",
       y = "Latitude")
```



Modeling

Category definitions

```r
all_cat = c(
  "Accounting, Tax Preparation, Bookkeeping, and Payroll Services",
  "Activities Related to Credit Intermediation",
  "Activities Related to Real Estate",
  "Advertising, Public Relations, and Related Services",
  "Agencies, Brokerages, and Other Insurance Related Activities",
  "Architectural, Engineering, and Related Services",
  "Automotive Parts, Accessories, and Tire Stores",
  "Automotive Repair and Maintenance",
  "Bakeries and Tortilla Manufacturing",
  "Beer, Wine, and Liquor Stores",
  "Book Stores and News Dealers",
  "Building Equipment Contractors",
  "Building Finishing Contractors",
  "Building Material and Supplies Dealers",
  "Child Day Care Services",
  "Clothing Stores",
  "Consumer Goods Rental",
```

```
"Couriers and Express Delivery Services",
"Depository Credit Intermediation",
"Drinking Places (Alcoholic Beverages)",
"Drycleaning and Laundry Services",
"Electronic and Precision Equipment Repair and Maintenance",
"Electronics and Appliance Stores",
"Elementary and Secondary Schools",
"Florists",
"Furniture Stores",
"Gasoline Stations",
"General Medical and Surgical Hospitals",
"General Merchandise Stores, including Warehouse Clubs and Supercenters",
"Glass and Glass Product Manufacturing",
"Grocery Stores",
"Health and Personal Care Stores",
"Home Furnishings Stores",
"Investigation and Security Services",
"Jewelry, Luggage, and Leather Goods Stores",
"Justice, Public Order, and Safety Activities",
"Legal Services",
"Machinery, Equipment, and Supplies Merchant Wholesalers",
"Museums, Historical Sites, and Similar Institutions",
"Offices of Dentists",
"Offices of Other Health Practitioners",
"Offices of Physicians",
"Offices of Real Estate Agents and Brokers",
"Other Amusement and Recreation Industries",
"Other Financial Investment Activities",
"Other Miscellaneous Manufacturing",
"Other Miscellaneous Store Retailers",
"Other Personal Services",
"Other Professional, Scientific, and Technical Services",
"Other Schools and Instruction",
"Other Specialty Trade Contractors",
"Personal and Household Goods Repair and Maintenance",
"Personal Care Services",
"Printing and Related Support Activities",
"Promoters of Performing Arts, Sports, and Similar Events",
"Radio and Television Broadcasting",
"Religious Organizations",
"Restaurants and Other Eating Places",
"Shoe Stores",
"Sound Recording Industries",
"Special Food Services",
"Specialized Design Services",
"Specialty (except Psychiatric and Substance Abuse) Hospitals",
"Specialty Food Stores",
"Sporting Goods, Hobby, and Musical Instrument Stores",
"Support Activities for Road Transportation",
"Technical and Trade Schools",
"Transit and Ground Passenger Transportation",
"Traveler Accommodation",
"Warehousing and Storage",
```

```r
  "Wired and Wireless Telecommunications Carriers"
)

medical_services = c(
  "General Medical and Surgical Hospitals",
  "Health and Personal Care Stores",
  "Offices of Dentists",
  "Offices of Other Health Practitioners",
  "Specialty (except Psychiatric and Substance Abuse) Hospitals",
  "Offices of Physicians"
)

essential_services = c(
  "General Medical and Surgical Hospitals",
  "Health and Personal Care Stores",
  "Pharmacies and Drug Stores",
  "Grocery Stores",
  "Gasoline Stations",
  "Depository Credit Intermediation",
  "Public Transport Hubs",
  "Government Offices"
)

retail_shopping = c(
  "General Merchandise Stores, including Warehouse Clubs and Supercenters",
  "Clothing Stores",
  "Shoe Stores",
  "Jewelry, Luggage, and Leather Goods Stores",
  "Electronics and Appliance Stores",
  "Furniture Stores",
  "Home Furnishings Stores",
  "Specialty Food Stores",
  "Sporting Goods, Hobby, and Musical Instrument Stores",
  "Book Stores and News Dealers"
)

entertainment_recreation = c(
  "Other Amusement and Recreation Industries",
  "Museums, Historical Sites, and Similar Institutions",
  "Promoters of Performing Arts, Sports, and Similar Events",
  "Radio and Television Broadcasting",
  "Sound Recording Industries"
)

personal_services = c(
  "Personal Care Services",
  "Drycleaning and Laundry Services",
  "Other Personal Services",
  "Personal and Household Goods Repair and Maintenance"
)

hospitality_lodging = c(
  "Traveler Accommodation",
```

```r
  "Bed and Breakfast Inns",
  "Resorts",
  "Extended Stay Hotels"
)

office_professional = c(
  "Accounting, Tax Preparation, Bookkeeping, and Payroll Services",
  "Legal Services",
  "Architectural, Engineering, and Related Services",
  "Agencies, Brokerages, and Other Insurance Related Activities",
  "Offices of Physicians",
  "Offices of Dentists",
  "Offices of Other Health Practitioners",
  "Real Estate Agencies"
)


target_categories = c("Drinking Places (Alcoholic Beverages)", "Restaurants and Other Eating Places", "
```

All categories vs. categories of interest

```r
df_long_model_filtered_1 = df_long |>
  mutate(non_restaurant = if_else(top_category %in% target_categories, "No", "Yes"))

poisson_model_interact_1 = glm(visitor_count ~ age_group * non_restaurant, family = poisson(link = "log
summary(poisson_model_interact_1)
```

```
##
## Call:
## glm(formula = visitor_count ~ age_group * non_restaurant, family = poisson(link = "log"),
##     data = df_long_model_filtered_1)
##
## Coefficients:
##                                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)                          0.16988    0.02238   7.592 3.16e-14 ***
## age_group19_65                       0.96487    0.02630  36.691  < 2e-16 ***
## age_group65_plus                    -0.10608    0.03252  -3.262  0.00111 **
## non_restaurantYes                    0.06969    0.02323   3.000  0.00270 **
## age_group19_65:non_restaurantYes     0.01149    0.02730   0.421  0.67383
## age_group65_plus:non_restaurantYes -0.03016    0.03378  -0.893  0.37195
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 85790  on 65684  degrees of freedom
## Residual deviance: 52109  on 65679  degrees of freedom
## AIC: 207746
##
## Number of Fisher Scoring iterations: 5
```

```r
dispersion_test = sum(residuals(poisson_model_interact_1, type = "pearson")^2) / poisson_model_interact_
print(dispersion_test)
```

```
## [1] 2.619864
```

```r
#Overdispersion present, use NB
```

NB models, overdispersion was present

NB model on whole data

"Are older individuals visiting restaurants/bars at lower rates compared to other age groups?" A negative estimate implies that an age group is visiting a location at a lower rate than the reference

```r
nb_whole = glm.nb(visitor_count ~ age_group, data = df_long)
summary(nb_whole)
```

```
##
## Call:
## glm.nb(formula = visitor_count ~ age_group, data = df_long, init.theta = 9.54321176,
##     link = log)
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)      0.234380   0.006397   36.64   <2e-16 ***
## age_group19_65   0.975531   0.007702  126.66   <2e-16 ***
## age_group65_plus -0.134037   0.009328  -14.37   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(9.5432) family taken to be 1)
##
##     Null deviance: 60126  on 65684  degrees of freedom
## Residual deviance: 32394  on 65682  degrees of freedom
## AIC: 199382
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  9.543
##           Std. Err.:  0.170
##
##  2 x log-likelihood:  -199373.606
```

```r
df_model_filtered = df_long |>
  filter(top_category %in% target_categories)

nb_rest = glm.nb(visitor_count ~ age_group, data = df_model_filtered)
summary(nb_rest)
```

```
##
## Call:
## glm.nb(formula = visitor_count ~ age_group, data = df_model_filtered,
##     init.theta = 45.18341845, link = log)
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)      0.16988    0.02267   7.494 6.68e-14 ***
## age_group19_65   0.96487    0.02679  36.013  < 2e-16 ***
```

```
## age_group65_plus -0.10608    0.03292  -3.222  0.00127 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(45.1834) family taken to be 1)
##
##     Null deviance: 4199.5  on 5054  degrees of freedom
## Residual deviance: 1982.1  on 5052  degrees of freedom
## AIC: 14056
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  45.18
##           Std. Err.:  9.56
##
##  2 x log-likelihood:  -14048.09
```

NB with interaction

"Are older individuals visiting restaurants/bars at lower rates compared to other location types?"

```r
run_nb_model = function(df, category_name, category_vector, reference_name, target_categories) {
  df_model = df |>
    filter(top_category %in% c(target_categories, category_vector)) |>  # Filter to only relevant POIs
    mutate(category_indicator = if_else(top_category %in% target_categories, reference_name, category_na

  nb_model = glm.nb(visitor_count ~ age_group * category_indicator, data = df_model)

  print(summary(nb_model))

  return(nb_model)
}

# All groups v. restaurant and bar
nb_model_1 = run_nb_model(df_long, "Non-Restaurant", all_cat, "Restaurant/Bar", target_categories)
```

```
##
## Call:
## glm.nb(formula = visitor_count ~ age_group * category_indicator,
##     data = df_model, init.theta = 9.554525783, link = log)
##
## Coefficients:
##                                                    Estimate Std. Error z value
## (Intercept)                                        0.239574   0.006642  36.069
## age_group19_65                                     0.976356   0.007998 122.078
## age_group65_plus                                  -0.136244   0.009691 -14.059
## category_indicatorRestaurant/Bar                  -0.069693   0.024637  -2.829
## age_group19_65:category_indicatorRestaurant/Bar   -0.011490   0.029661  -0.387
## age_group65_plus:category_indicatorRestaurant/Bar  0.030160   0.035717   0.844
##                                                   Pr(>|z|)
## (Intercept)                                        < 2e-16 ***
## age_group19_65                                     < 2e-16 ***
## age_group65_plus                                   < 2e-16 ***
## category_indicatorRestaurant/Bar                   0.00467 **
```

```
## age_group19_65:category_indicatorRestaurant/Bar    0.69848
## age_group65_plus:category_indicatorRestaurant/Bar  0.39843
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(9.5545) family taken to be 1)
##
##     Null deviance: 60143  on 65684  degrees of freedom
## Residual deviance: 32371  on 65679  degrees of freedom
## AIC: 199353
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  9.555
##           Std. Err.:  0.171
##
##  2 x log-likelihood:  -199338.680
```

```r
# Healthcare v. restaurant and bar
nb_model_2 = run_nb_model(df_long, "Healthcare", medical_services, "Restaurant/Bar", target_categories)
```

```
##
## Call:
## glm.nb(formula = visitor_count ~ age_group * category_indicator,
##     data = df_model, init.theta = 16.46988646, link = log)
##
## Coefficients:
##                                                    Estimate Std. Error z value
## (Intercept)                                         0.19046    0.01381  13.791
## age_group19_65                                      0.94419    0.01652  57.155
## age_group65_plus                                   -0.09558    0.01998  -4.783
## category_indicatorRestaurant/Bar                   -0.02058    0.02697  -0.763
## age_group19_65:category_indicatorRestaurant/Bar     0.02068    0.03220   0.642
## age_group65_plus:category_indicatorRestaurant/Bar -0.01051    0.03910  -0.269
##                                                    Pr(>|z|)
## (Intercept)                                         < 2e-16 ***
## age_group19_65                                      < 2e-16 ***
## age_group65_plus                                   1.73e-06 ***
## category_indicatorRestaurant/Bar                      0.446
## age_group19_65:category_indicatorRestaurant/Bar       0.521
## age_group65_plus:category_indicatorRestaurant/Bar     0.788
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(16.4699) family taken to be 1)
##
##     Null deviance: 15923.6  on 19010  degrees of freedom
## Residual deviance:  8366.6  on 19005  degrees of freedom
## AIC: 54874
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  16.470
```

```
##           Std. Err.:  0.813
##
##  2 x log-likelihood:  -54860.302
```

```r
# Essential Services v. restaurant and bar
nb_model_3 = run_nb_model(df_long, "Essential Services", essential_services, "Restaurant/Bar", target_ca
```

```
##
## Call:
## glm.nb(formula = visitor_count ~ age_group * category_indicator,
##     data = df_model, init.theta = 26.13995157, link = log)
##
## Coefficients:
##                                                 Estimate Std. Error z value
## (Intercept)                                      0.15415    0.02114   7.293
## age_group19_65                                   0.96781    0.02507  38.608
## age_group65_plus                                -0.07688    0.03046  -2.524
## category_indicatorRestaurant/Bar                 0.01573    0.03115   0.505
## age_group19_65:category_indicatorRestaurant/Bar  -0.00294    0.03695  -0.080
## age_group65_plus:category_indicatorRestaurant/Bar -0.02921    0.04506  -0.648
##                                                 Pr(>|z|)
## (Intercept)                                     3.04e-13 ***
## age_group19_65                                   < 2e-16 ***
## age_group65_plus                                  0.0116 *
## category_indicatorRestaurant/Bar                  0.6136
## age_group19_65:category_indicatorRestaurant/Bar   0.9366
## age_group65_plus:category_indicatorRestaurant/Bar   0.5169
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(26.14) family taken to be 1)
##
##     Null deviance: 9178.0  on 11066  degrees of freedom
## Residual deviance: 4541.3  on 11061  degrees of freedom
## AIC: 31175
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  26.14
##           Std. Err.:  2.42
##
##  2 x log-likelihood:  -31161.46
```

```r
# Retail shopping v. restaurant and bar
nb_model_4 = run_nb_model(df_long, "Retail Shopping", retail_shopping, "Restaurant/Bar", target_categor
```

```
##
## Call:
## glm.nb(formula = visitor_count ~ age_group * category_indicator,
##     data = df_model, init.theta = 9.004769194, link = log)
##
## Coefficients:
##                                                   Estimate Std. Error z value
## (Intercept)                                        0.16988    0.02380   7.136
## age_group19_65                                     0.96487    0.02869  33.626
```

```
## age_group65_plus                                      -0.10608    0.03449  -3.076
## category_indicatorRetail Shopping                      0.14392    0.02756   5.222
## age_group19_65:category_indicatorRetail Shopping      -0.01954    0.03327  -0.587
## age_group65_plus:category_indicatorRetail Shopping    -0.09149    0.04014  -2.279
##                                                       Pr(>|z|)
## (Intercept)                                           9.58e-13 ***
## age_group19_65                                         < 2e-16 ***
## age_group65_plus                                        0.0021 **
## category_indicatorRetail Shopping                     1.77e-07 ***
## age_group19_65:category_indicatorRetail Shopping        0.5570
## age_group65_plus:category_indicatorRetail Shopping      0.0227 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(9.0048) family taken to be 1)
##
##     Null deviance: 17166.1  on 18149  degrees of freedom
## Residual deviance:  9447.9  on 18144  degrees of freedom
## AIC: 55979
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  9.005
##           Std. Err.:  0.304
##
##  2 x log-likelihood:  -55965.252
```

```
# Entertainment/Recreation v. restaurant and bar
nb_model_5 = run_nb_model(df_long, "Entertainment/Recreation", entertainment_recreation, "Restaurant/Ba
```

```
##
## Call:
## glm.nb(formula = visitor_count ~ age_group * category_indicator,
##     data = df_model, init.theta = 46.87244572, link = log)
##
## Coefficients:
##                                                       Estimate Std. Error z value
## (Intercept)                                           0.163602   0.035466   4.613
## age_group19_65                                        0.914571   0.042194  21.676
## age_group65_plus                                     -0.122549   0.051726  -2.369
## category_indicatorRestaurant/Bar                      0.006278   0.042086   0.149
## age_group19_65:category_indicatorRestaurant/Bar       0.050295   0.049972   1.006
## age_group65_plus:category_indicatorRestaurant/Bar     0.016465   0.061306   0.269
##                                                      Pr(>|z|)
## (Intercept)                                          3.97e-06 ***
## age_group19_65                                        < 2e-16 ***
## age_group65_plus                                       0.0178 *
## category_indicatorRestaurant/Bar                       0.8814
## age_group19_65:category_indicatorRestaurant/Bar        0.3142
## age_group65_plus:category_indicatorRestaurant/Bar      0.7883
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(46.8724) family taken to be 1)
```

```
##
##     Null deviance: 5810.5  on 7130  degrees of freedom
## Residual deviance: 2776.8  on 7125  degrees of freedom
## AIC: 19750
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  46.87
##           Std. Err.:  8.76
##
##  2 x log-likelihood:  -19736.01
```

```
# Personal Services v. restaurant and bar
nb_model_6 = run_nb_model(df_long, "Personal Services", personal_services, "Restaurant/Bar", target_cat
```

```
##
## Call:
## glm.nb(formula = visitor_count ~ age_group * category_indicator,
##     data = df_model, init.theta = 9.692034869, link = log)
##
## Coefficients:
##                                                    Estimate Std. Error z value
## (Intercept)                                         0.42828    0.02622  16.334
## age_group19_65                                      1.00102    0.03162  31.663
## age_group65_plus                                   -0.22537    0.03905  -5.771
## category_indicatorRestaurant/Bar                   -0.25839    0.03535  -7.310
## age_group19_65:category_indicatorRestaurant/Bar    -0.03616    0.04259  -0.849
## age_group65_plus:category_indicatorRestaurant/Bar   0.11929    0.05201   2.294
##                                                    Pr(>|z|)
## (Intercept)                                         < 2e-16 ***
## age_group19_65                                      < 2e-16 ***
## age_group65_plus                                   7.86e-09 ***
## category_indicatorRestaurant/Bar                   2.67e-13 ***
## age_group19_65:category_indicatorRestaurant/Bar      0.3958
## age_group65_plus:category_indicatorRestaurant/Bar    0.0218 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(9.692) family taken to be 1)
##
##     Null deviance: 8178.5  on 8348  degrees of freedom
## Residual deviance: 4160.4  on 8343  degrees of freedom
## AIC: 25686
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  9.692
##           Std. Err.:  0.477
##
##  2 x log-likelihood:  -25671.812
```

```
#Hospitality/Lodging v. restaurant and bar
nb_model_7 = run_nb_model(df_long, "Hospitality/Lodging", hospitality_lodging, "Restaurant/Bar", target_
```

```
##
## Call:
## glm.nb(formula = visitor_count ~ age_group * category_indicator,
##     data = df_model, init.theta = 12.01832498, link = log)
##
## Coefficients:
##                                                  Estimate Std. Error z value
## (Intercept)                                       0.65818    0.03774  17.441
## age_group19_65                                    1.04095    0.04533  22.965
## age_group65_plus                                 -0.40057    0.05876  -6.817
## category_indicatorRestaurant/Bar                 -0.48830    0.04443 -10.990
## age_group19_65:category_indicatorRestaurant/Bar   -0.07608    0.05334  -1.426
## age_group65_plus:category_indicatorRestaurant/Bar  0.29449    0.06789   4.338
##                                                  Pr(>|z|)
## (Intercept)                                       < 2e-16 ***
## age_group19_65                                    < 2e-16 ***
## age_group65_plus                                 9.30e-12 ***
## category_indicatorRestaurant/Bar                  < 2e-16 ***
## age_group19_65:category_indicatorRestaurant/Bar      0.154
## age_group65_plus:category_indicatorRestaurant/Bar 1.44e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(12.0183) family taken to be 1)
##
##     Null deviance: 6441.0  on 6320  degrees of freedom
## Residual deviance: 2858.1  on 6315  degrees of freedom
## AIC: 19022
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  12.018
##           Std. Err.:  0.737
##
##  2 x log-likelihood:  -19008.466
```

```
#Office/Professional v. restaurant and bar
nb_model_8 = run_nb_model(df_long, "Office/Professional", office_professional, "Restaurant/Bar", target_
```

```
##
## Call:
## glm.nb(formula = visitor_count ~ age_group * category_indicator,
##     data = df_model, init.theta = 13.85877443, link = log)
##
## Coefficients:
##                                                   Estimate Std. Error z value
## (Intercept)                                       0.187900   0.013578  13.838
## age_group19_65                                    0.956841   0.016257  58.857
## age_group65_plus                                 -0.098487   0.019655  -5.011
## category_indicatorRestaurant/Bar                 -0.018020   0.026981  -0.668
## age_group19_65:category_indicatorRestaurant/Bar    0.008025   0.032272   0.249
## age_group65_plus:category_indicatorRestaurant/Bar -0.007597   0.039109  -0.194
##                                                  Pr(>|z|)
## (Intercept)                                       < 2e-16 ***
```

```
## age_group19_65                                     < 2e-16 ***
## age_group65_plus                                  5.42e-07 ***
## category_indicatorRestaurant/Bar                      0.504
## age_group19_65:category_indicatorRestaurant/Bar       0.804
## age_group65_plus:category_indicatorRestaurant/Bar     0.846
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(13.8588) family taken to be 1)
##
##     Null deviance: 16524  on 19712  degrees of freedom
## Residual deviance:  8660  on 19707  degrees of freedom
## AIC: 57223
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  13.859
##           Std. Err.:  0.575
##
##  2 x log-likelihood:  -57209.489
```

```r
extract_nb_results = function(model, category_name) {
  results = broom.mixed::tidy(model) |>
    filter(grepl("category_indicator", term)) |>
    mutate(category = category_name) |>
    relocate(category) |>
    mutate(significance = case_when(
      p.value < 0.001 ~ "***",
      p.value < 0.01  ~ "**",
      p.value < 0.05  ~ "*",
      TRUE ~ ""
    ))

  return(results)
}

nb_summary_table = bind_rows(
  extract_nb_results(nb_model_1, "Non-Restaurant"),
  extract_nb_results(nb_model_2, "Healthcare"),
  extract_nb_results(nb_model_3, "Essential Services"),
  extract_nb_results(nb_model_4, "Retail Shopping"),
  extract_nb_results(nb_model_5, "Entertainment/Recreation"),
  extract_nb_results(nb_model_6, "Personal Services"),
  extract_nb_results(nb_model_7, "Hospitality/Lodging"),
  extract_nb_results(nb_model_8, "Office/Professional")
)

knitr::kable(nb_summary_table)
```

category	term	estimate	std.error	statistic	p.value	significance
Non-Restaurant	category_indicatorRestaurant/Bar	-0.0696934	0.0246370	-2.8288065	0.0046722	**

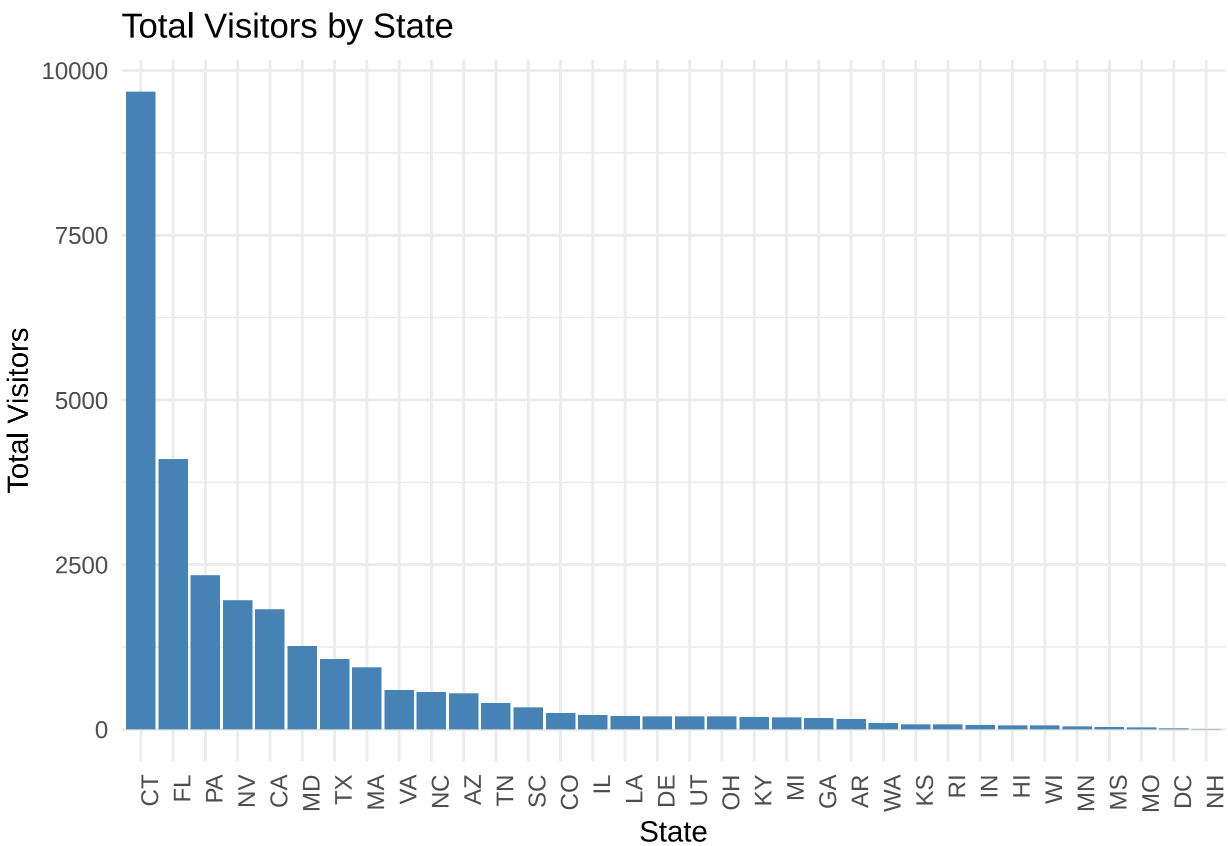
category	term	estimate	std.error	statistic	p.value	significance
Non-Restaurant	age_group19_65:category_indicatorRestaurant/Bar	-0.0114896	0.0296605	-0.3873693	0.6984828	
Non-Restaurant	age_group65_plus:category_indicatorRestaurant/Bar	0.0301600	0.0357168	0.8444211	0.3984341	
Healthcare	category_indicatorRestaurant/Bar	-0.0205764	0.0269726	-0.7628643	0.4455443	
Healthcare	age_group19_65:category_indicatorRestaurant/Bar	0.0206811	0.0321951	0.6423670	0.5206349	
Healthcare	age_group65_plus:category_indicatorRestaurant/Bar	-0.0105073	0.0391012	-0.2687220	0.7881436	
Essential Services	category_indicatorRestaurant/Bar	0.0157298	0.0311490	0.5049860	0.6135687	
Essential Services	age_group19_65:category_indicatorRestaurant/Bar	-0.0029401	0.0369507	-0.0795695	0.9365796	
Essential Services	age_group65_plus:category_indicatorRestaurant/Bar	-0.0292086	0.0450637	-0.6481616	0.5168804	
Retail Shopping	category_indicatorRetail Shopping	0.1439180	0.0275589	5.2222048	0.0000002	***
Retail Shopping	age_group19_65:category_indicatorRetail Shopping	-0.0195402	0.0332668	-0.5873768	0.5569507	
Retail Shopping	age_group65_plus:category_indicatorRetail Shopping	-0.0914854	0.0401446	-2.2788963	0.0226732	*
Entertainment/Recreation	category_indicatorRestaurant/Bar	0.0062783	0.0420861	0.1491784	0.8814128	
Entertainment/Recreation	age_group19_65:category_indicatorRestaurant/Bar	0.0502948	0.0499717	1.0064651	0.3141919	
Entertainment/Recreation	age_group65_plus:category_indicatorRestaurant/Bar	0.0164649	0.0613065	0.2685669	0.7882630	
Personal Services	category_indicatorRestaurant/Bar	-0.2583947	0.0353472	-7.3101897	0.0000000	***
Personal Services	age_group19_65:category_indicatorRestaurant/Bar	-0.0361583	0.0425856	-0.8490732	0.3958406	
Personal Services	age_group65_plus:category_indicatorRestaurant/Bar	0.1192875	0.0520088	2.2936024	0.0218133	*
Hospitality/Lodging	category_indicatorRestaurant/Bar	-0.4883023	0.0444331	-10.9896169	0.0000000	***
Hospitality/Lodging	age_group19_65:category_indicatorRestaurant/Bar	-0.0760822	0.0533379	-1.4264211	0.1537468	
Hospitality/Lodging	age_group65_plus:category_indicatorRestaurant/Bar	0.2944851	0.0678907	4.3376364	0.0000144	***
Office/Professional	category_indicatorRestaurant/Bar	-0.0180197	0.0269805	-0.6678778	0.5042116	
Office/Professional	age_group19_65:category_indicatorRestaurant/Bar	0.0080249	0.0322721	0.2486651	0.8036199	
Office/Professional	age_group65_plus:category_indicatorRestaurant/Bar	-0.0075972	0.0391087	-0.1942590	0.8459731	

Additional Visualizations

Visitor counts

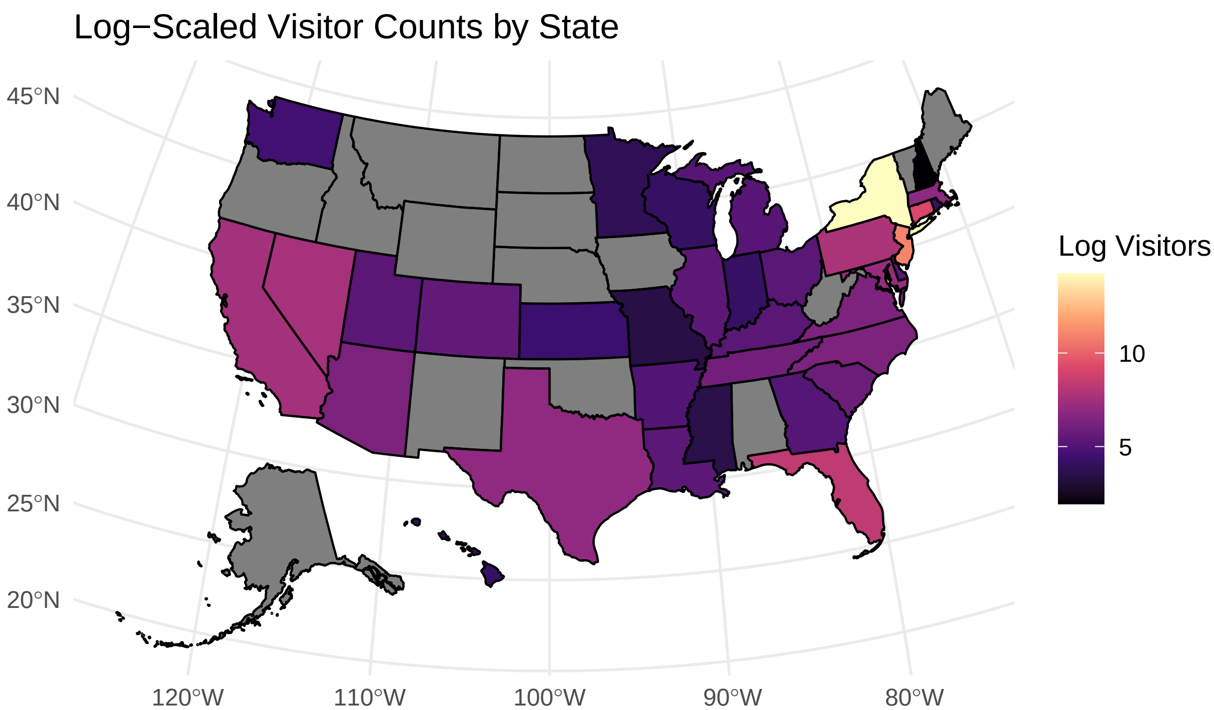
```r
state_visitors = final_df_with_state |>
  filter(visitor_state != "NY") |>
  filter(visitor_state != "NJ") |>
  group_by(visitor_state) |>
  summarize(total_visitors = sum(visitors, na.rm = TRUE)) |>
  arrange(desc(total_visitors))

ggplot(state_visitors, aes(x = reorder(visitor_state, -total_visitors), y = total_visitors)) +
  geom_col(fill = "steelblue") +
  labs(
```

```
    title = "Total Visitors by State",
    x = "State",
    y = "Total Visitors"
  ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Map view of visitor counts

```
# Aggregate visitor counts by state
state_visitors_map = final_df_with_state |>
  group_by(visitor_state) |>
  summarize(total_visitors = sum(visitors, na.rm = TRUE)) |>
  filter(!is.na(visitor_state)) |>
  mutate(log_visitors = log1p(total_visitors)) |>
  rename(state = visitor_state)

# Plot using log scale
plot_usmap(data = state_visitors_map, regions = "states", values = "log_visitors") +
  scale_fill_viridis_c(name = "Log Visitors", option = "magma") +
  labs(title = "Log-Scaled Visitor Counts by State") +
  theme_minimal()
```

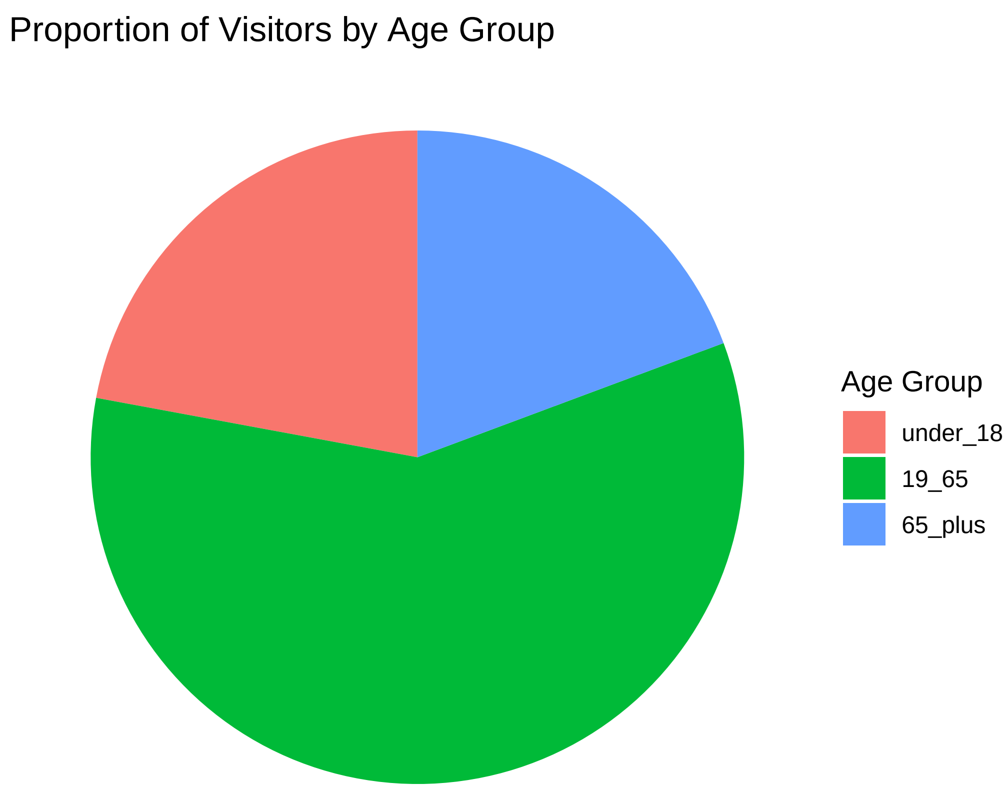


Bar Plot

```r
age_group_summary = df_long_model_filtered_1 |>
  group_by(age_group, top_category) |>
  summarize(total_visitors = sum(visitor_count), .groups = "drop")

ggplot(age_group_summary, aes(x = age_group, y = total_visitors, fill = top_category)) +
  geom_col(position = "dodge") +
  labs(
    title = "Visitor Counts by Age Group and Location Type",
    x = "Age Group",
    y = "Total Visitors",
    fill = "Location Type"
  ) +
  theme_minimal()
```

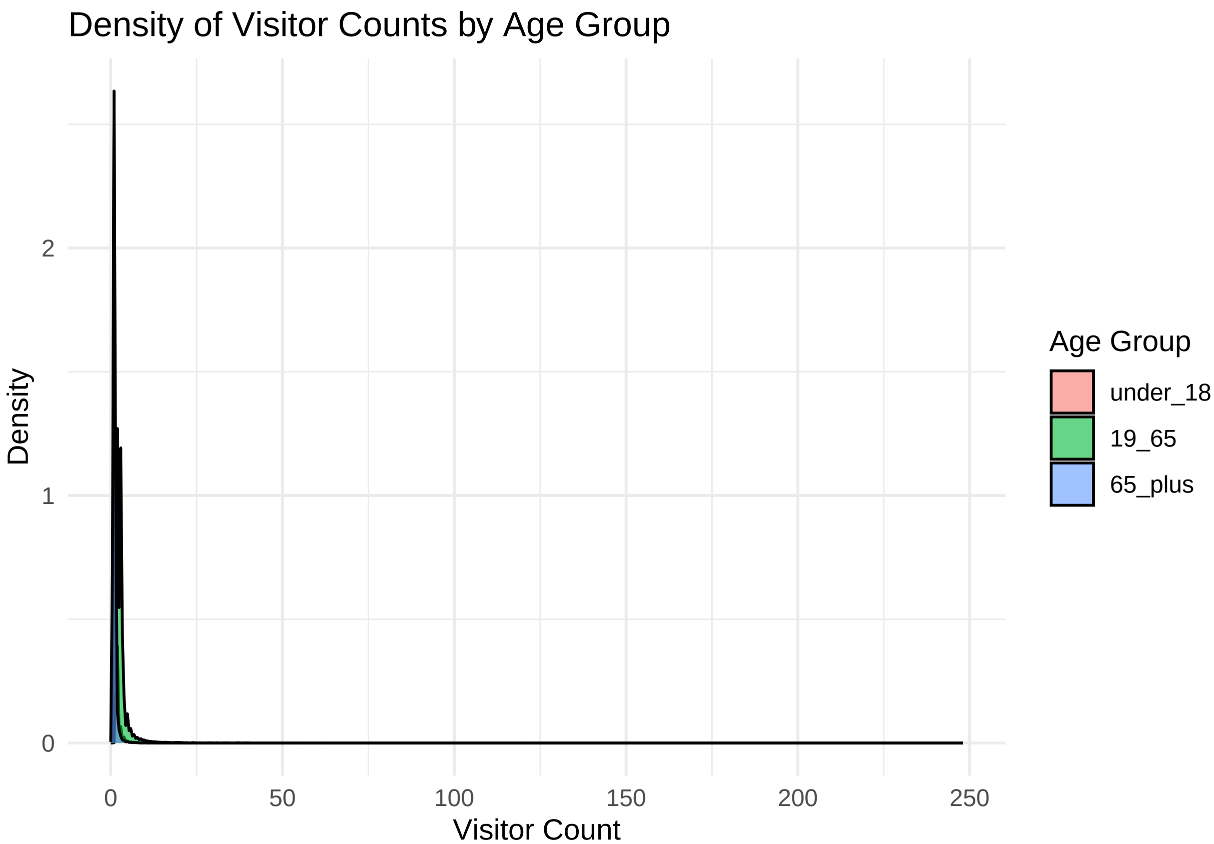
Pie chart (alternative to above)

```
age_group_proportions = df_long_model_filtered_1 |>
  group_by(age_group) |>
  summarize(total_visitors = sum(visitor_count), .groups = "drop") |>
  mutate(proportion = total_visitors / sum(total_visitors))

ggplot(age_group_proportions, aes(x = "", y = proportion, fill = age_group)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar(theta = "y") +
  labs(
    title = "Proportion of Visitors by Age Group",
    fill = "Age Group"
  ) +
  theme_void()
```



Density Plot

```r
ggplot(df_long_model_filtered_1, aes(x = visitor_count, fill = age_group)) +
  geom_density(alpha = 0.6) +
  labs(
    title = "Density of Visitor Counts by Age Group",
    x = "Visitor Count",
    y = "Density",
    fill = "Age Group"
  ) +
  theme_minimal()
```


Density of Visitor Counts by Age Group
Density
2
1
0
0
50
100
150
200
250
Visitor Count
Age Group
under_18
19_65
65_plus