Project 2 EDA

```
library(sp)
library(spmodel)
```

```
## Warning: package ’spmodel’ was built under R version 4.3.3
```

```
library(sf)
```

```
## Warning: package ’sf’ was built under R version 4.3.3
```

```
## Linking to GEOS 3.11.0, GDAL 3.5.3, PROJ 9.1.0; sf_use_s2() is TRUE
```

```
library(spdep)
```

```
## Warning: package ’spdep’ was built under R version 4.3.3
```

```
## Loading required package: spData
```

```
## Warning: package ’spData’ was built under R version 4.3.3
```

```
## To access larger datasets in this package, install the spDataLarge
## package with: ‘install.packages(’spDataLarge’,
## repos=’https://nowosad.github.io/drat/’, type=’source’)‘
```

```
library(tmap)
```

```
## Warning: package ’tmap’ was built under R version 4.3.3
```

```
library(corrplot)
```

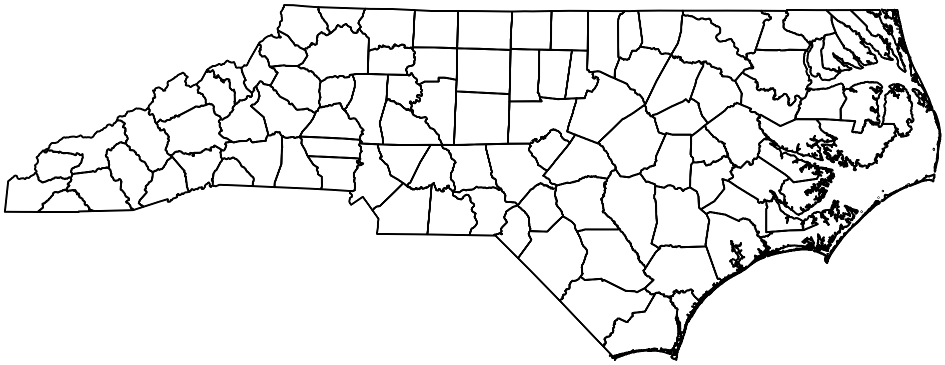
```
## Warning: package ’corrplot’ was built under R version 4.3.3
```

```
## corrplot 0.95 loaded
```

```
library(readr)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v purrr     1.0.2
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## -- 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(tinytex)
library(ggcorrplot)

## Load in the demographic data and county data for North Carolina
ncdata=read.csv("NCcounty2024data.csv",header=T)
load("nc_counties.Rdata")

## Plot the counties
plot(nc_counties)
```



```r
## Names of the counties
nc_counties[[7]]
```

```
##   [1] "Alamance"     "Alexander"    "Alleghany"    "Anson"        "Ashe"
##   [6] "Avery"        "Beaufort"     "Bertie"       "Bladen"       "Brunswick"
##  [11] "Buncombe"     "Burke"        "Cabarrus"     "Caldwell"     "Camden"
##  [16] "Carteret"     "Caswell"      "Catawba"      "Chatham"      "Cherokee"
##  [21] "Chowan"       "Clay"         "Cleveland"    "Columbus"     "Craven"
##  [26] "Cumberland"   "Currituck"    "Dare"         "Davidson"     "Davie"
##  [31] "Duplin"       "Durham"       "Edgecombe"    "Forsyth"      "Franklin"
##  [36] "Gaston"       "Gates"        "Graham"       "Granville"    "Greene"
##  [41] "Guilford"     "Halifax"      "Harnett"      "Haywood"      "Henderson"
##  [46] "Hertford"     "Hoke"         "Hyde"         "Iredell"      "Jackson"
##  [51] "Johnston"     "Jones"        "Lee"          "Lenoir"       "Lincoln"
##  [56] "Macon"        "Madison"      "Martin"       "McDowell"     "Mecklenburg"
##  [61] "Mitchell"     "Montgomery"   "Moore"        "Nash"         "New Hanover"
##  [66] "Northampton"  "Onslow"       "Orange"       "Pamlico"      "Pasquotank"
##  [71] "Pender"       "Perquimans"   "Person"       "Pitt"         "Polk"
##  [76] "Randolph"     "Richmond"     "Robeson"      "Rockingham"   "Rowan"
##  [81] "Rutherford"   "Sampson"      "Scotland"     "Stanly"       "Stokes"
##  [86] "Surry"        "Swain"        "Transylvania" "Tyrrell"      "Union"
##  [91] "Vance"        "Wake"         "Warren"       "Washington"   "Watauga"
##  [96] "Wayne"        "Wilkes"       "Wilson"       "Yadkin"       "Yancey"
## Obtain neighborhoods given the polygons
nc_neigh=poly2nb(nc_counties)
```

```
#nc_counties$NAME_2
```

```
# Convert to sf object before merge
nc_counties_sf <- st_as_sf(nc_counties)

# Make sure the name matches
nc_counties_sf$County <- nc_counties_sf$NAME_2

# Now merge
nc_full <- left_join(nc_counties_sf, ncdata, by = "County")
```

```
# First we make sure county names align
head(nc_counties[[7]])          # county names from earlier
```

```
## [1] "Alamance"  "Alexander" "Alleghany" "Anson"     "Ashe"      "Avery"
```

```
head(ncdata$County)
```

```
## [1] "Alamance"  "Alexander" "Alleghany" "Anson"     "Ashe"      "Avery"
```

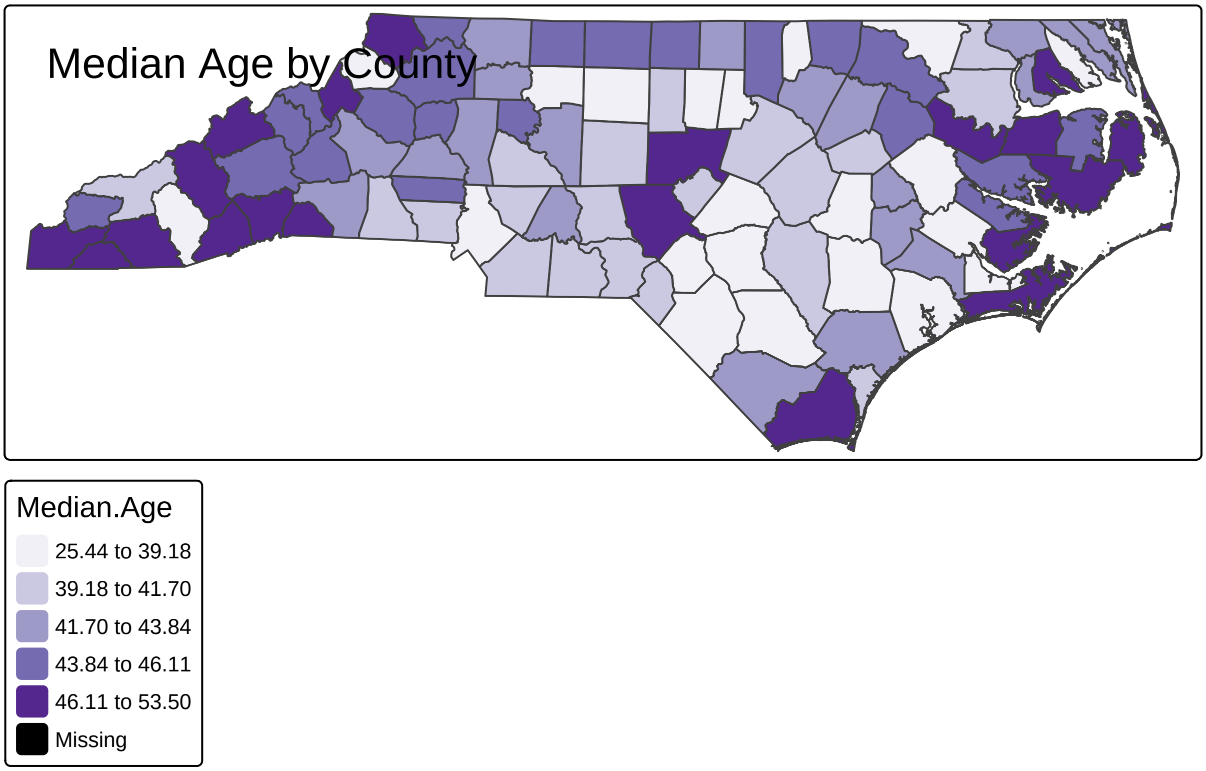
```
# Attacingh names as a column in nc_counties
nc_counties@data$County <- nc_counties[[7]]

# Merge by county name
nc_full <- merge(nc_counties, ncdata, by = "County")

# Convert to sf object before merge
nc_counties_sf <- st_as_sf(nc_counties)

# Make sure the name matches
nc_counties_sf$County <- nc_counties_sf$NAME_2

# Now merge
nc_full <- left_join(nc_counties_sf, ncdata, by = "County")
```

```
tm_shape(nc_full) +
  tm_fill("Median.Age", style = "quantile", palette = "Purples") +
  tm_borders() +
  tm_layout(title = "Median Age by County")
```

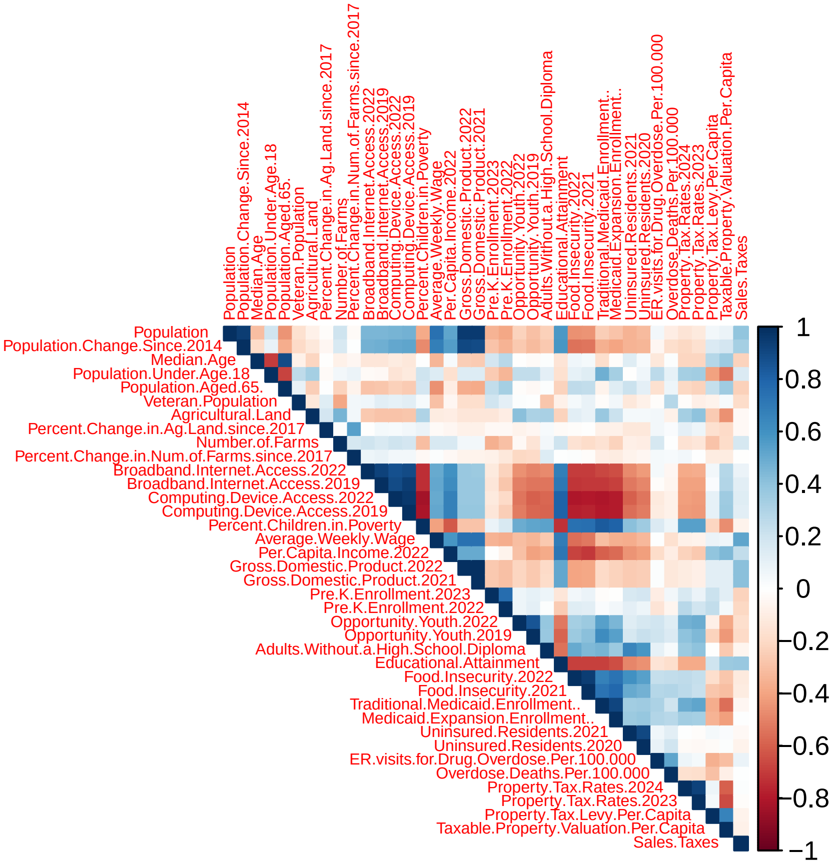
```
##
```

```
## -- tmap v3 code detected -------------------------------------------------------
```

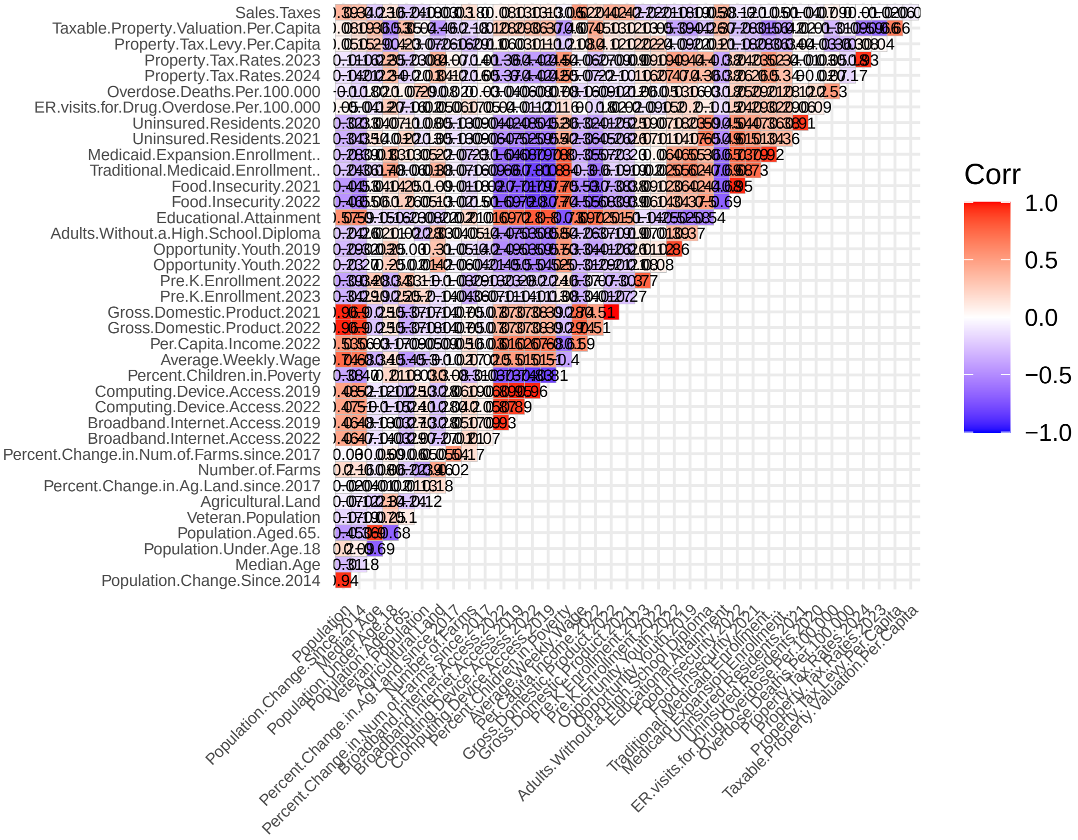
```
## [v3->v4] ‘tm_fill()‘: instead of ‘style = "quantile"‘, use fill.scale =
## ‘tm_scale_intervals()‘.
## i Migrate the argument(s) ’style’, ’palette’ (rename to ’values’) to
##   'tm_scale_intervals(<HERE>)'
## [v3->v4] `tm_layout()`: use `tm_title()` instead of `tm_layout(title = )`
## [cols4all] color palettes: use palettes from the R package cols4all. Run
## `cols4all::c4a_gui()` to explore them. The old palette name "Purples" is named
## "brewer.purples"
## Multiple palettes called "purples" found: "brewer.purples", "matplotlib.purples". The first one, "bre
```



```r
numeric_vars <- ncdata %>%
  select(where(is.numeric))

corr_matrix <- cor(numeric_vars, use = "complete.obs")

corrplot(cor(numeric_vars, use = "complete.obs"),
         method = "color",
         type = "upper",
         tl.cex = 0.5)  # try 0.4 or 0.3 if needed
```



```r
corr_matrix <- cor(numeric_vars, use = "complete.obs")

ggcorrplot(corr_matrix,
           lab = TRUE,
           type = "upper",
           lab_size = 2,
           tl.cex = 6,
           colors = c("blue", "white", "red"))
```



Based on the heat map, some interesting variables might be (for select topics): - Population.Change.Since.2014 (urbanization, migration, policy) - Food.Insecurity.2022 (health equity & access) - Overdose.Deaths.Per.100.000 (public health) - Percent.Children.in.Poverty (long-term socioeconomic outcomes)

We want to pick predictors that are: - Correlated (but not too correlated) with our outcome - Interpretable and likely to explain the response - Not multicollinear (check pairwise correlation < 0.8 if possible)

Again using the heatmap, it looks like the best 3 predictors are: - If we use Population.Change.Since.2014 as our target, then predictors could be Median.Age, Broadband.Internet.Access, Poverty.Rate, Educational.Attainment (Since young, educated areas with internet might grow more).

-For Food.Insecurity.2022 as the target, we could look at Percent.Children.in.Poverty, Adults.Without.a.High.School.Diploma, Uninsured.Residents.2020 etc. (due to the link between poverty and food insecurity, education and food scarcity, and lack of healthcare and food insecurity).

- If we use Overdose.Deaths.Per.100.000 as the target, we could look at things like Uninsured.Residents.2020, Adults.Without.High.School.Diploma, Food.Insecurity.2021 (structural vulnerabilities).

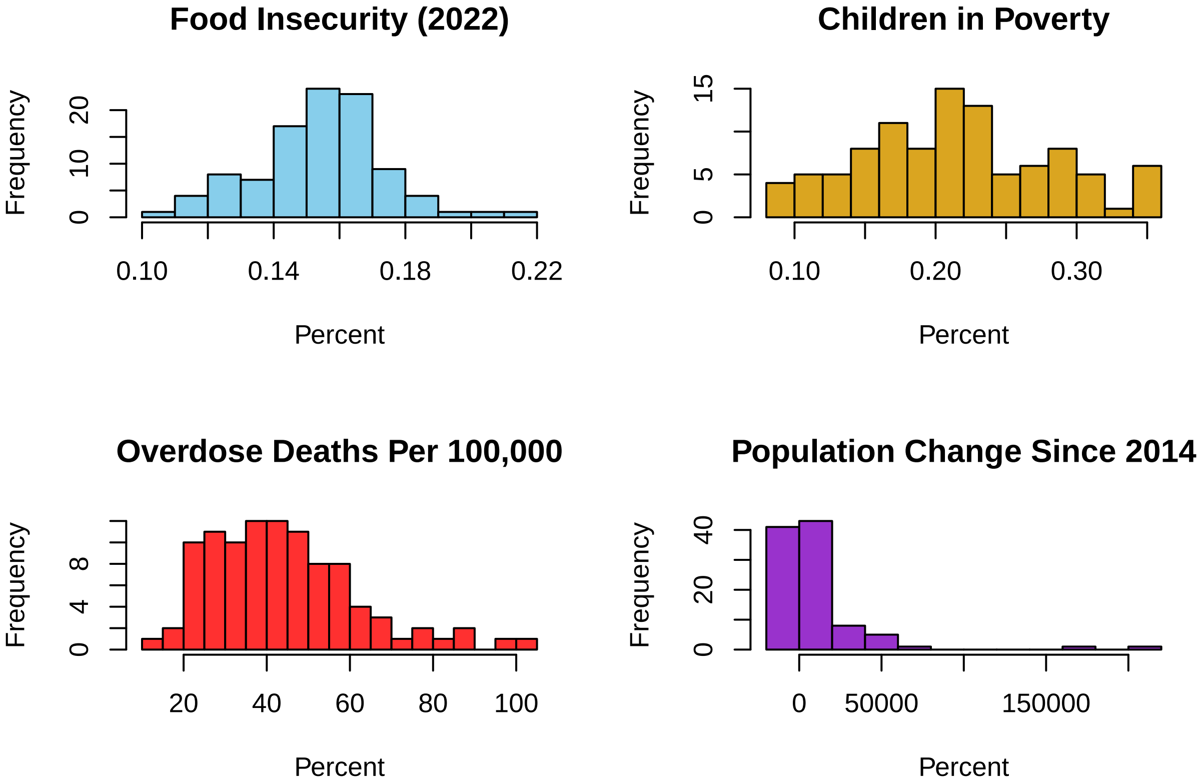
-If we look at Percent.Children.in.Poverty, we might look at things like: Unemployment.Rate, Food.Insecurity, Educational.Attainment, Average.Weekly.Wage (since these speak to parental resources and economic stability).

```
nc_full$Population.Change.Since.2014
```

```
##  [1]  28556   -359    540  -3079    514   -357  -2822  -3011  -4001  51407
## [11]  26947   2034  55137    608   1264   2709  -1225  12196  15220   2151
## [21]   -493   1272   4098  -5390   1956  13614   9007   2847  14790   3788
## [31]  -5278  46239  -6177  32110  18271  31688   -816   -780   5175   -895
## [41]  40463  -5576  20909   4818  11790  -4159   6308   -929  39622   2495
## [51]  68170   -817   8126  -4197  19118   3958    317  -2355     63 178869
## [61]   -399  -1117  19381   3999  29145  -4193  22409  11045   -568   1398
## [71]  14550   -151    615   8033   -183   4050  -2369  -8645   -379  12761
## [81]   -677  -2325  -2309   3971    302  -1196   -789    680   -663  48061
## [91]  -3009 217857  -1293  -1817   5946  -4968  -1397   -894     40    850
```

Let's take a look at histograms since understanding the shape of our distribution will help us understand what type of model we should use (linear or poisson, etc), help us avoid too many outliers dominating the model, decide if a transformation of the target variable is needed, and to visually asses normality and skewness.

```r
# Set up 2x2 grid of plots
par(mfrow = c(2, 2))

# Histogram 1: Food Insecurity
hist(nc_full$Food.Insecurity.2022,
     main = "Food Insecurity (2022)",
     xlab = "Percent",
     col = "skyblue", breaks = 15)

# Histogram 2: Percent Children in Poverty
hist(nc_full$Percent.Children.in.Poverty,
     main = "Children in Poverty",
     xlab = "Percent",
     col = "goldenrod", breaks = 15)

# Histogram 3: Overdose Deaths Per 100,000
hist(nc_full$Overdose.Deaths.Per.100.000 ,
     main = "Overdose Deaths Per 100,000",
     xlab = "Percent",
     col = "firebrick1", breaks = 15)

# Histogram 4: Popluation Change Since 2014
hist(nc_full$Population.Change.Since.2014,
     main = "Population Change Since 2014",
     xlab = "Percent",
     col = "darkorchid", breaks = 15)
```



```r
# Reset layout
par(mfrow = c(1, 1))
```

Which seems to be the best for the target variable?

For the Food Insecurity target, we have:

- Distribution: Fairly normal (bell-shaped), centered around ~15–18%.
- Pros: Good for regression — a nice, symmetric target.
- Cons: If you're interested in identifying outliers or extremes, this might not be ideal since it's quite balanced.
- Verdict: Solid choice for standard regression, but not particularly surprising or extreme.

For Children in Poverty, we have: - Distribution: Roughly uniform or slightly bimodal, with a wide spread.

- Pros: Could be interesting for classification (e.g., high vs. low poverty), or exploring clusters.
- Cons: Might be more difficult for regression due to less clear central tendency.
- Verdict: Good for classification or policy-based segmentation.

For Overdose Deaths, we have:

- Distribution: Right-skewed, long tail — many counties with low deaths, few with very high rates.
- Pros: Super interesting for prediction. That skew suggests some counties are outliers worth understanding. Great for modeling rare but serious outcomes.

- Cons: Requires care in handling outliers (e.g., log transformation).
- Verdict: Most interesting if your goal is identifying high-risk areas or understanding disparities.

For - Distribution: Highly skewed right, many near zero or negative, with a few extreme positives.

- Pros: Could be fascinating to model migration or growth dynamics.
- Cons: Huge range, lots of zeros or small changes — may require normalization or feature engineering.
- Verdict: Also very interesting, but potentially noisier.

My personal opinion is that we should go with Overdose Deaths.

Overdose deaths are heavily spatially structured and certain regions (like Appalachia, parts of the South, or the Rust Belt) have had historically higher rates. That makes it a perfect candidate for a spatial regression model.

The right skew also gives us the challenge of transforming or modeling a non-normal target — often the more statistically interesting and policy-relevant scenario. You might log-transform the response or try a Poisson or negative binomial model depending on how it's measured.

Furthermore, modeling overdose deaths allows us to make a policy case — identifying predictors like poverty, unemployment, healthcare access, or education could point to real solutions. It's a meaningful and timely topic.

Even after controlling for variables like poverty or population change, we'll probably have leftover spatial structure — so your spatial dependence test (e.g., Moran's I on residuals) will likely be fruitful and interesting.