FinalProject

Megh

2025-04-15

1. DATA PREPARATIONS

1.1 SAIPE data

```
raw_saipe <- read.csv("SAIPE.csv")
saipe <- raw_saipe |> select(Year, FIPS = ID, Name, Population = Poverty.Universe,
                             Poverty = Number.in.Poverty)
saipe <- saipe |> mutate(Population = as.integer(gsub(",", "", Population)),
                         Poverty = as.integer(gsub(",", "", Poverty)))
```

```
## Warning: There were 2 warnings in `mutate()`.
## The first warning was:
## i In argument: `Population = as.integer(gsub(",", "", Population))`.
## Caused by warning:
## ! NAs introduced by coercion
## i Run `dplyr::last_dplyr_warnings()` to see the 1 remaining warning.
```

State Name: Georgia

State Abbv. : GA

FIPS Code: 13

```
saipe$Name[saipe$Name == "De Kalb County"] <- "DeKalb County"
saipe |> distinct(FIPS) |> count()
```

```
##     n
## 1 159
```

There are 159 counties in Georgia.

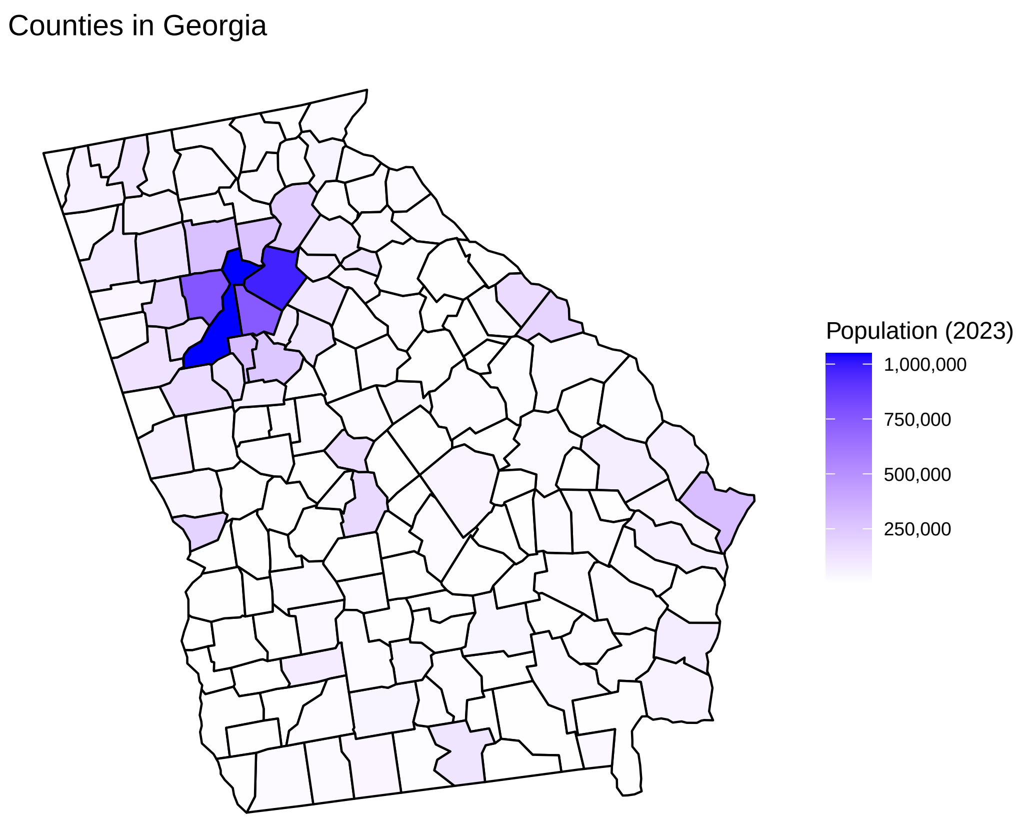
```
top_10counties <- saipe |> filter(Year == 2023) |> arrange(desc(Population)) |>
  head(n=10)
top_10counties
```

```
##    Year  FIPS            Name Population Poverty
## 1  2023 13121   Fulton County    1047709  136621
## 2  2023 13135 Gwinnett County     975728  111168
## 3  2023 13067     Cobb County     765204   67115
## 4  2023 13089   DeKalb County     749408  100015
## 5  2023 13063  Clayton County     292420   50474
```

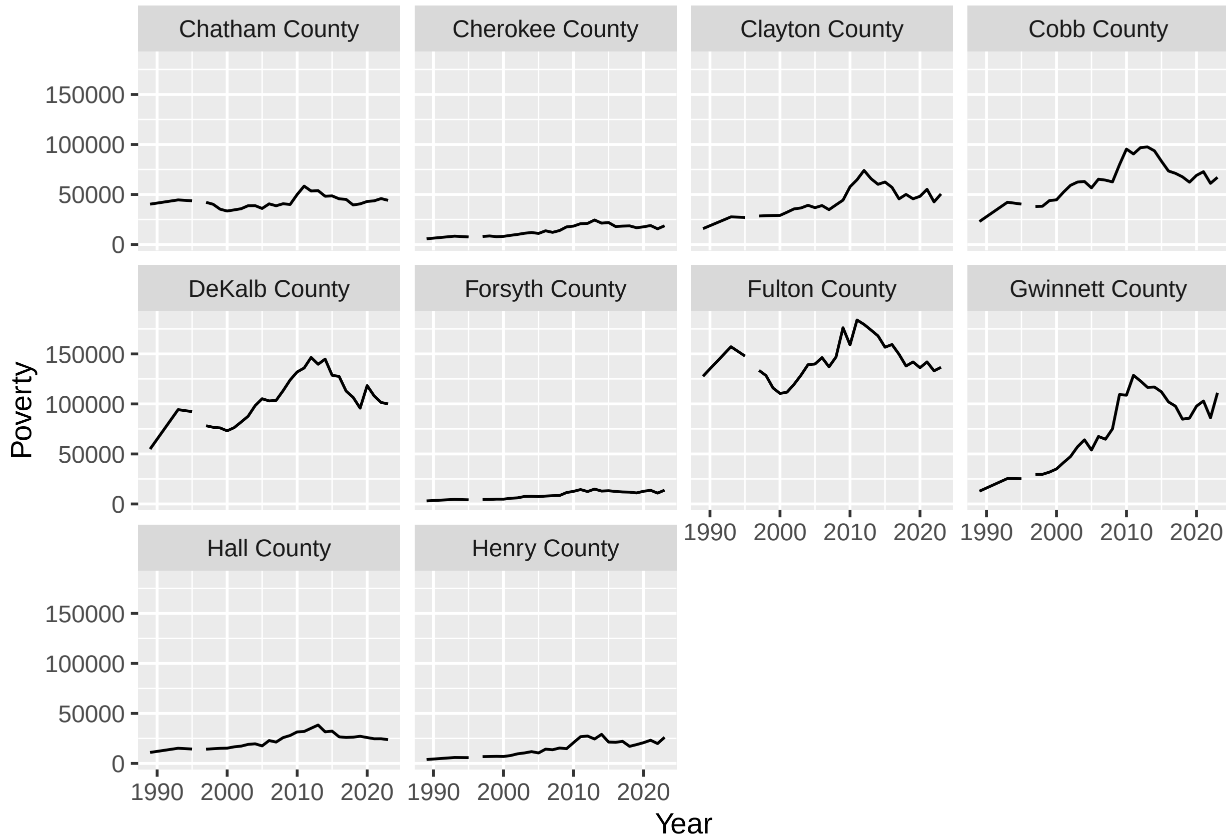
```
## 6  2023 13051  Chatham County     290391   44111
## 7  2023 13057 Cherokee County     284182   18708
## 8  2023 13117  Forsyth County     271213   13783
## 9  2023 13151    Henry County     252752   26087
## 10 2023 13139     Hall County     215151   23740
```

The largest county is Fulton County.

```
map_data <- saipe |> filter(Year == 2023) |> mutate(fips = FIPS)
plot_usmap(regions = "counties", include = "GA", data = map_data, values = "Population") +
  scale_fill_continuous(low = "white", high ="blue", name = "Population (2023)",
                        label = scales::comma)  +
  labs(title = "Counties in Georgia") +
  theme(legend.position = "right")
```



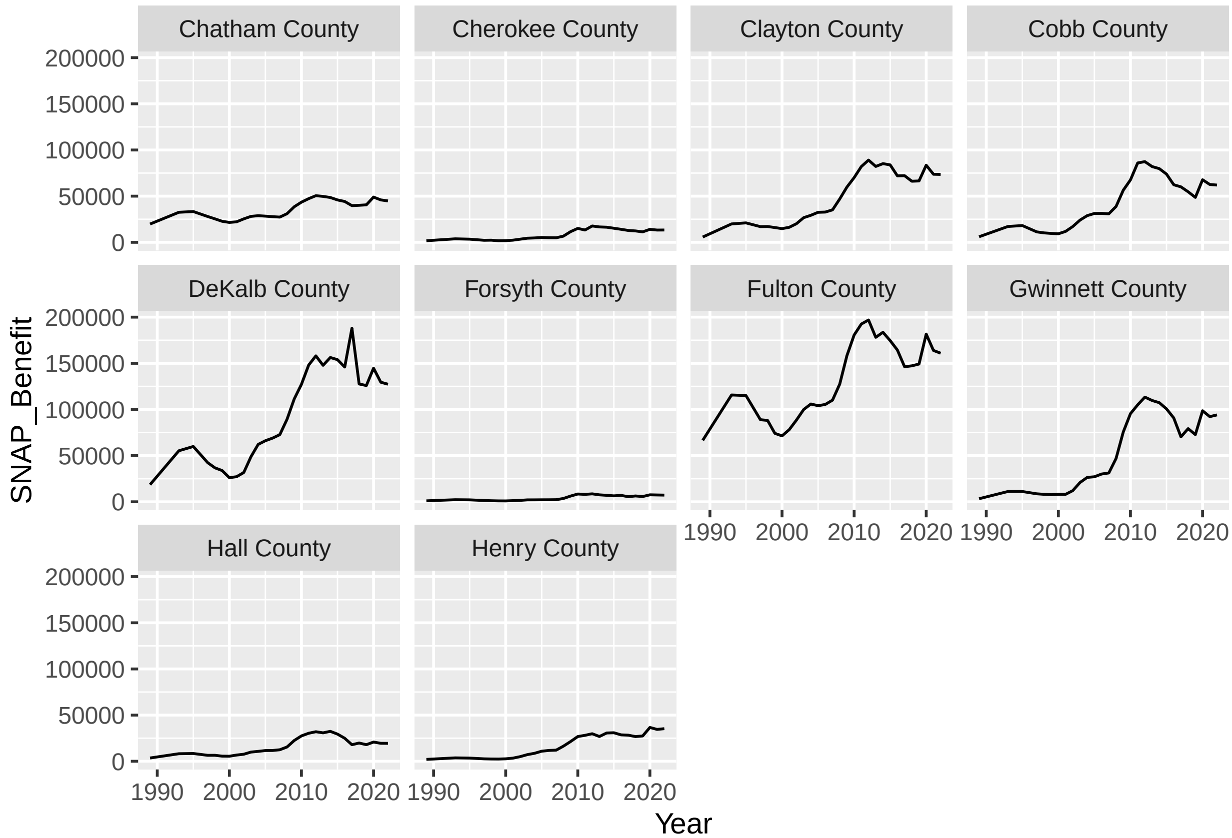

```
top_10names <- top_10counties$Name
saipe |> filter(Name %in% top_10names) |> ggplot(aes(x = Year, y = Poverty)) +
  geom_line() +
  facet_wrap(~ Name)
```



1.2 County SNAP Benefits

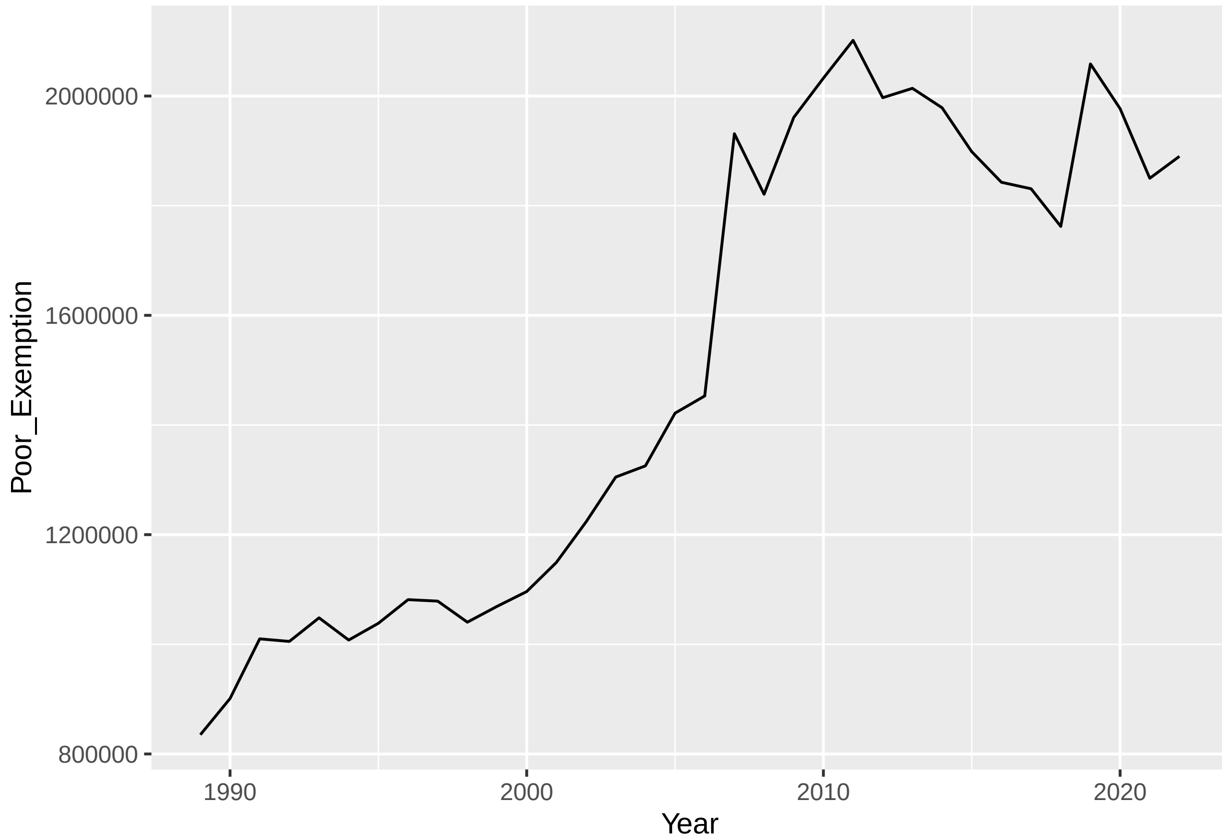
```
snap_raw <- read.csv("cntysnap.csv", skip = 5)
snap_untidy <- snap_raw |> filter(State.FIPS.code == 13, County.FIPS.code != 0) |>
  mutate(FIPS = as.integer(State.FIPS.code*1000 + County.FIPS.code))
snap <- snap_untidy |> pivot_longer(cols = starts_with("Jul"), names_to = "MonthYear",
                                    values_to = "SNAP_Benefit") |>
  mutate(Name = substr(Name, 1, nchar(Name)-4),
         Year = as.integer(substr(MonthYear, 5,8)),
         SNAP_Benefit = as.integer(gsub(",", "", SNAP_Benefit))) |>
  select(Name, FIPS, SNAP_Benefit, Year)

snap |> filter(Name %in% top_10names) |> ggplot(aes(x = Year, y = SNAP_Benefit)) +
  geom_line() +
  facet_wrap(~ Name)
```



1.3 State IRS Data

```
iris_raw <- read.csv("irs.csv", skip = 5)
iris <- iris_raw |> filter(State.FIPS.code == 13) |>
  mutate(Poor_Exemption = as.integer(gsub(",", "", Poor.exemptions))) |>
  select(Poor_Exemption, Year)
iris |> ggplot(aes(x = Year, y = Poor_Exemption)) + geom_line()
```



1.4 Merging the data

```
data_two <- saipe |> left_join(snap)
```

```
## Joining with `by = join_by(Year, FIPS, Name)`
```

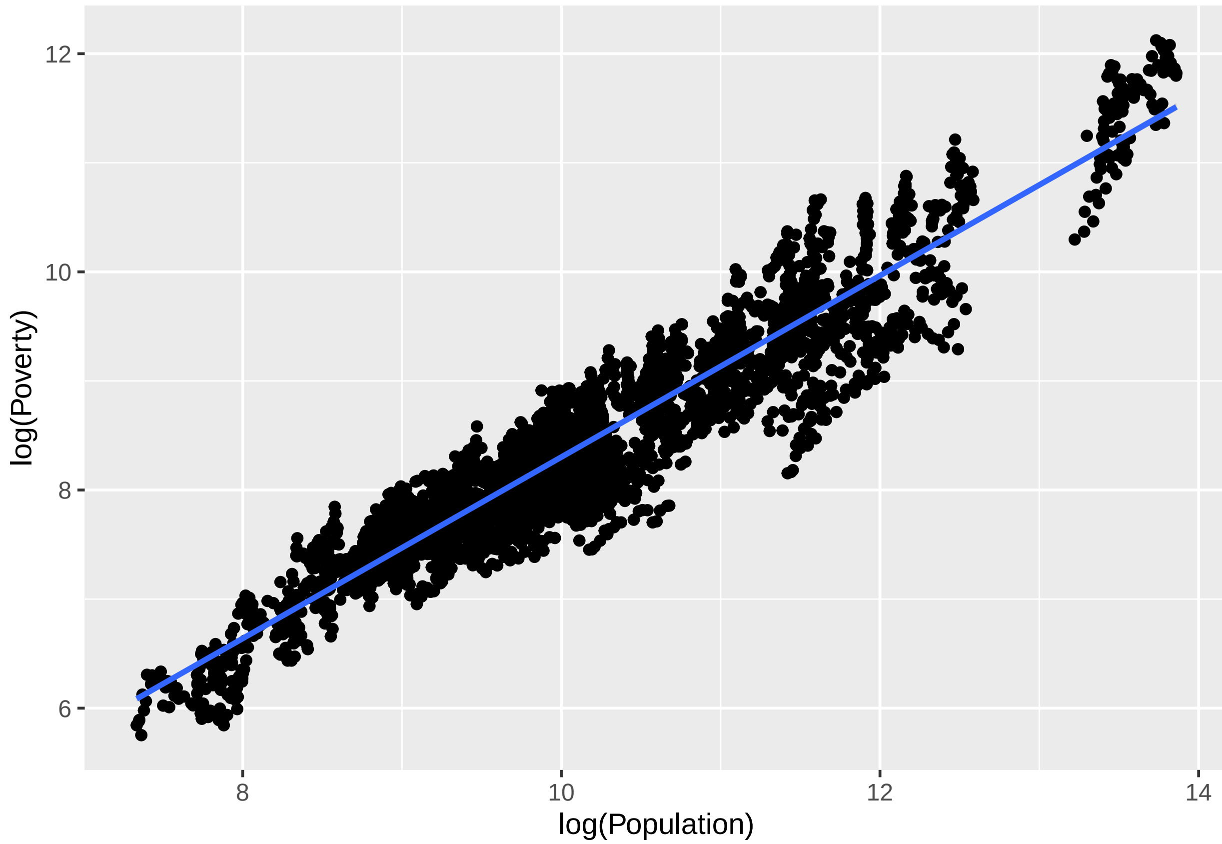
```
final_data <- data_two |> left_join(iris) |> filter(Year != 2023, Year > 1997)
```

```
## Joining with `by = join_by(Year)`
```

```
#Ignoring the Year 2023 because we don't have SNAP and IRIS data for that year.
```

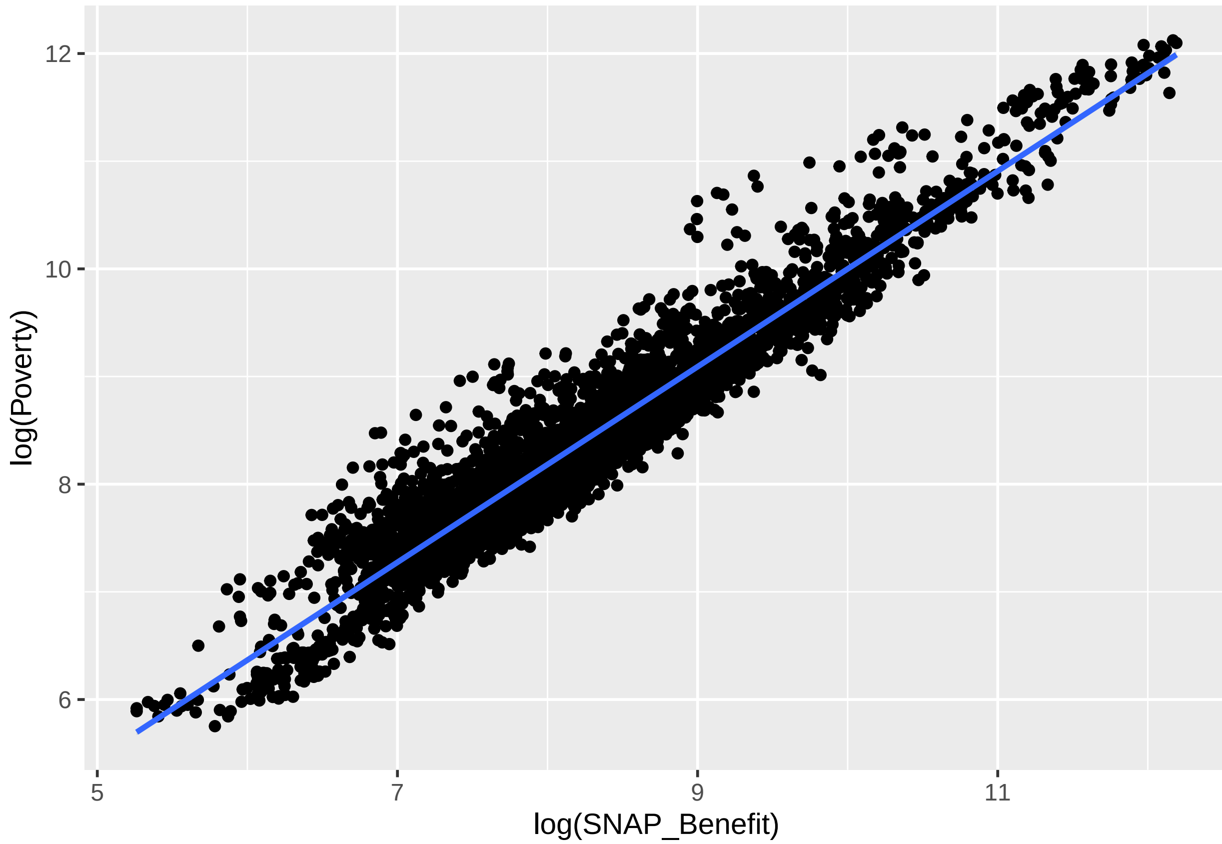
```
final_data |> ggplot(aes(x = log(Population), y = log(Poverty))) + geom_point() +
  geom_smooth(method = "lm")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

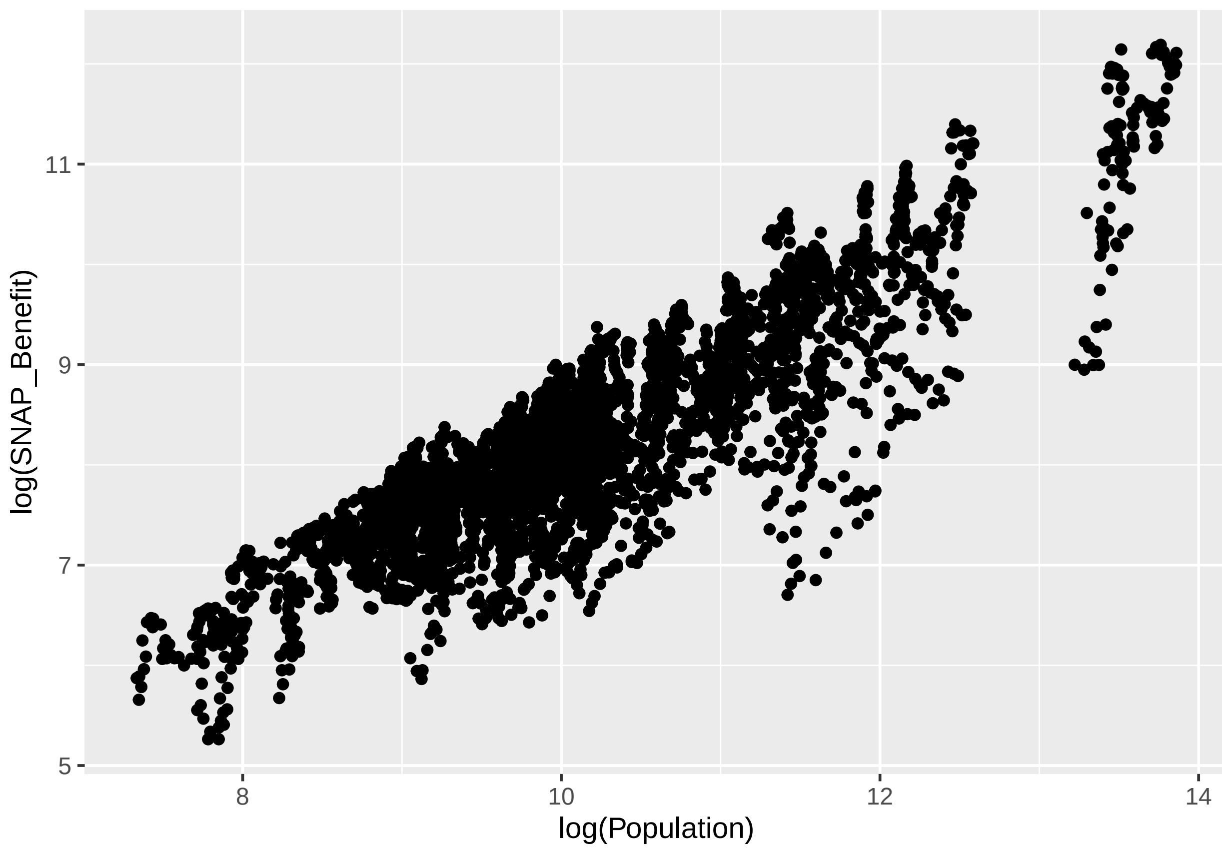



```
final_data |> ggplot(aes(x = log(SNAP_Benefit), y = log(Poverty))) + geom_point() +
  geom_smooth(method = "lm")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
final_data |> ggplot(aes(x = log(Population), y = log(SNAP_Benefit))) + geom_point()
```




```
georgia_data <- final_data |> as_tsibble(index = Year, key = c(FIPS, Name))
```

2. LINEAR MODELS

2.1 Variable Selection

```
models <- georgia_data |> model(model_pop = TSLM(log(Poverty) ~ log(Population)),
                                model_snap = TSLM(log(Poverty) ~ log(SNAP_Benefit)),
                                model_pe = TSLM(log(Poverty) ~ log(Poor_Exemption)),
                                model_pop_snap = TSLM(log(Poverty) ~ log(Population)
                                                      + log(SNAP_Benefit)),
                                model_pop_pe = TSLM(log(Poverty) ~ log(Population)
                                                    + log(Poor_Exemption)),
                                model_snap_pe = TSLM(log(Poverty) ~ log(SNAP_Benefit)
                                                     + log(Poor_Exemption)),
                                model_all = TSLM(log(Poverty) ~ log(Population)
                                                 + log(SNAP_Benefit) + log(Poor_Exemption)))
model_results <- models |> report()
```

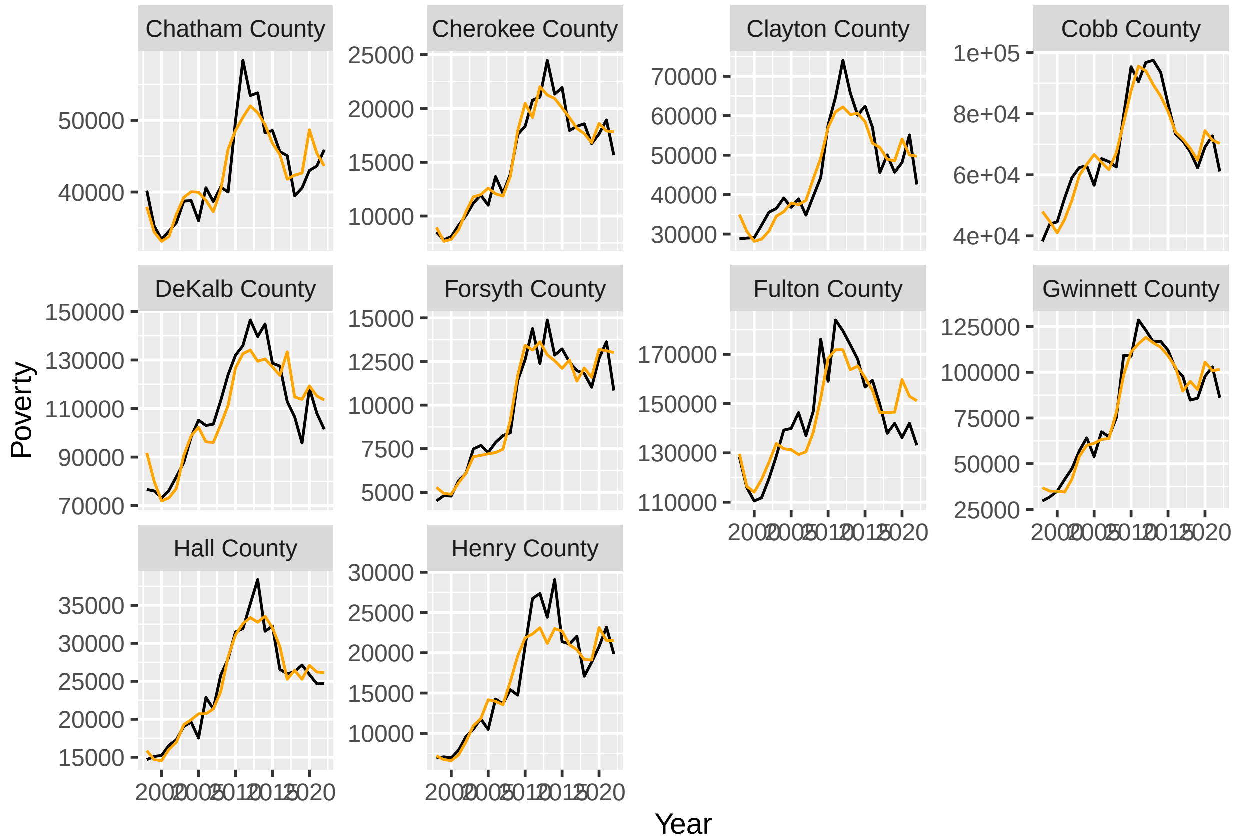
```
## Warning in report.mdl_df(models): Model reporting is only supported for
## individual models, so a glance will be shown. To see the report for a specific
## model, use `select()` and `filter()` to identify a single model.
```

```r
# To find the best model across all counties, first we find which model performs
# the best for each county
best_models_per_county <- model_results |> group_by(Name) |>
  slice_max(order_by = adj_r_squared, n = 1) |> ungroup()
# Simply count the number of times each model performs best for the counties
best_model_counts <- best_models_per_county |> count(.model) |> arrange(desc(n))
best_model_counts
```

```
## # A tibble: 7 x 2
##   .model             n
##   <chr>          <int>
## 1 model_pop_snap    52
## 2 model_all         41
## 3 model_snap        31
## 4 model_pop_pe      17
## 5 model_snap_pe     11
## 6 model_pe           5
## 7 model_pop          2
```

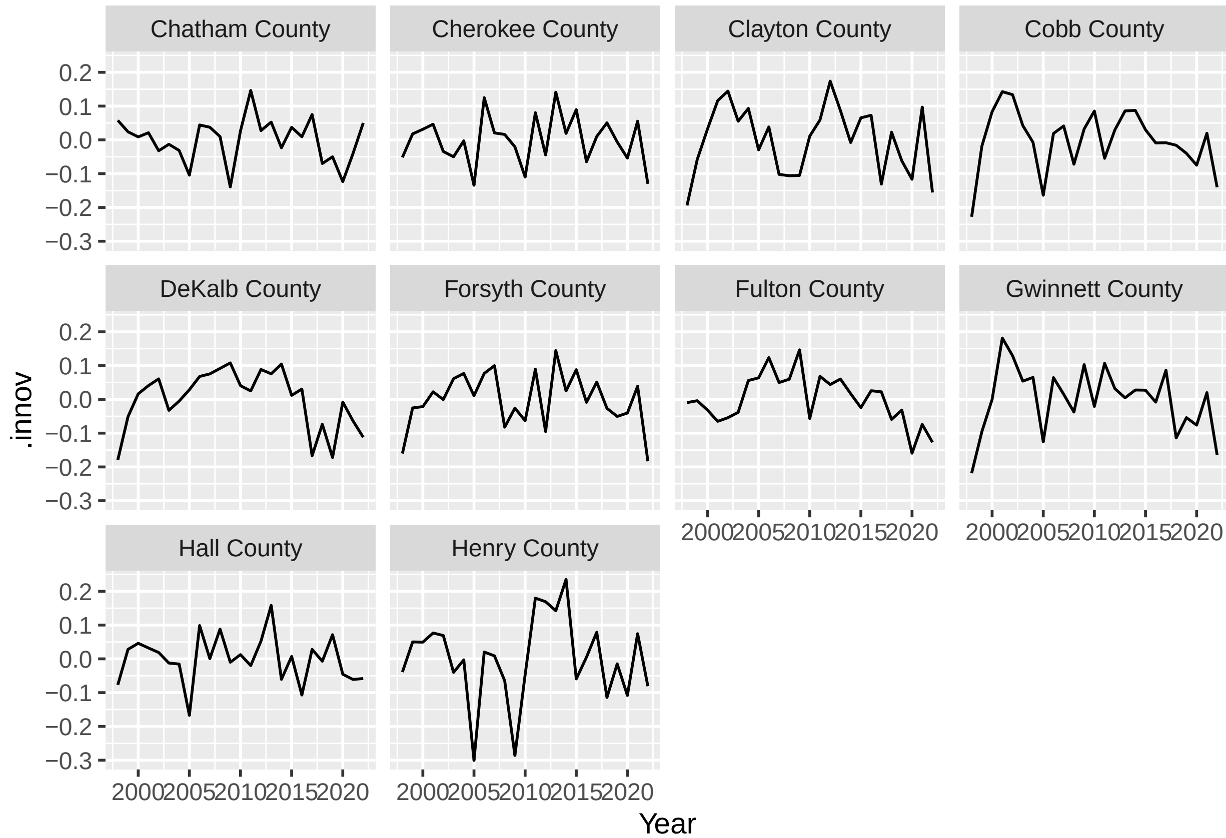
It seems that the best model include two input variables which are Population and SNAP_Benefit.

```r
best_model <- georgia_data |> model(TSLM(log(Poverty) ~ log(Population) + log(SNAP_Benefit)))
georgia_pred <- best_model |> augment()
georgia_pred |> filter(Name %in% top_10names) |> ggplot(aes(x = Year, y = Poverty)) +
  geom_line() +
  geom_line(aes(y = .fitted), color = "Orange") +
  facet_wrap(~Name, scales = "free_y")
```



2.2 Residual Analysis

```
georgia_pred |> filter(Name %in% top_10names) |> ggplot(aes(x = Year, y = .innov)) +
  geom_line() + facet_wrap(~Name)
```




```
georgia_pred |> features(.innov, ljung_box, lag=10) |> filter(lb_pvalue < 0.05) |> count()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1    37
```

37 counties residuals are significantly different from white noise

I think the models does a good job of predicting the number in poverty because there are only 37 counties whose residuals does not look like white noise and have some sort of relation. That means, there are 122 counties, where the model predicts the value of poverty without leaving any information in the residuals.

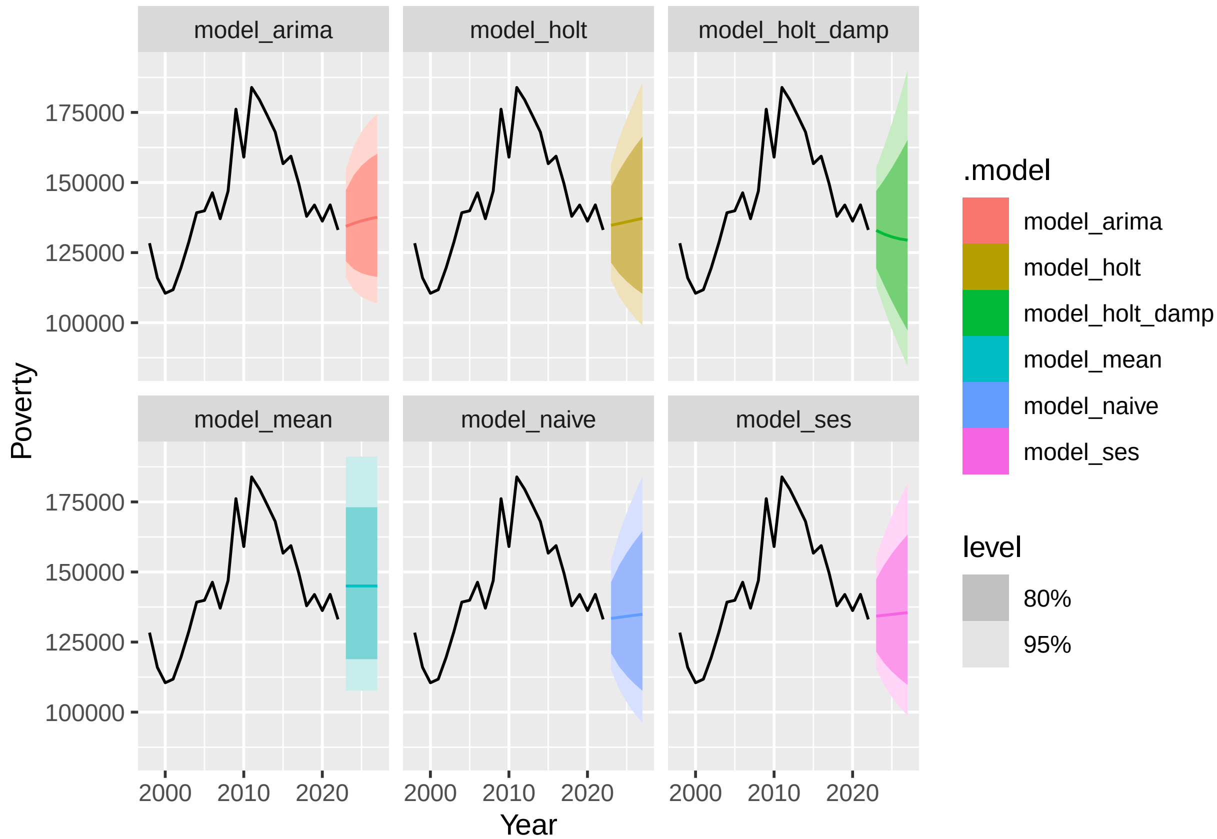
3. STOCHASTIC MODELS

3.1 Single County Forecasts

```
fulton_data <- georgia_data |> filter(Name == "Fulton County")
fulton_models <- fulton_data |> model(model_naive = NAIVE(log(Poverty)),
                                      model_mean = MEAN(log(Poverty)),
                                      model_ses = ETS(log(Poverty) ~ error("A")
                                                      + trend("N") + season("N")),
                                      model_holt = ETS(log(Poverty) ~ error("A")
                                                       + trend("A") + season("N")),
                                      model_holt_damp = ETS(log(Poverty) ~ error("A")
```

```
                                                + trend("Ad", phi = 0.9)
                                                + season("N")),
                              model_arima = ARIMA(log(Poverty)))

fulton_pred <- fulton_models |> forecast(h = 5)
autoplot(fulton_pred, fulton_data) + facet_wrap(~ .model)
```




```
fulton_models |> accuracy()
```

```
## # A tibble: 6 x 12
##    FIPS Name         .model .type     ME   RMSE    MAE     MPE  MAPE  MASE RMSSE
##   <int> <chr>        <chr>  <chr>  <dbl>  <dbl>  <dbl>   <dbl> <dbl> <dbl> <dbl>
## 1 13121 Fulton Coun~ model~ Trai~   197.  11271.  9170. -0.119    6.18 1     1
## 2 13121 Fulton Coun~ model~ Trai~ 1423.  20393. 16305. -0.996   11.4  1.78  1.81
## 3 13121 Fulton Coun~ model~ Trai~   340.  10889.  8641. -0.0240   5.85 0.942 0.966
## 4 13121 Fulton Coun~ model~ Trai~   -10.1 10886.  8603. -0.272    5.83 0.938 0.966
## 5 13121 Fulton Coun~ model~ Trai~   -56.9 10929.  9081. -0.0701   6.13 0.990 0.970
## 6 13121 Fulton Coun~ model~ Trai~ 1025.  10686.  8039.  0.191    5.48 0.877 0.948
## # i 1 more variable: ACF1 <dbl>
```

It seems that the arima model performs the best when it comes to predicting the Poverty for Fulton County.It has the least RMSE as well as MAE which means that it does a better job than others.

3.2 Exponential smoothing model

```r
exp_models <- georgia_data |> model(SES = ETS(log(Poverty) ~ error("A") + trend("N")
                                        + season("N")),
                                    Holt = ETS(log(Poverty) ~ error("A") + trend("A")
                                         + season("N")),
                                    Damped_Holt = ETS(log(Poverty) ~ error("A")
                                                + trend("Ad") + season("N")))
exp_model_results <- exp_models |> glance()
best_exp_models_per_county <- exp_model_results |> group_by(Name) |>
  slice_min(order_by = AICc, n = 1) |> ungroup()
best_exp_model_counts <- best_exp_models_per_county |> count(.model) |> arrange(desc(n))
best_exp_model_counts
```

```
## # A tibble: 2 x 2
##   .model          n
##   <chr>       <int>
## 1 SES           157
## 2 Damped_Holt     2
```

The results might be quite surprising as the more complex Holt and Holt Damped models were completely dominated by the performance of the simple exponential smoothing model. The results are completely one sided with SES model performing the best for 157 out of 159 counties. So, it is obvious to select simple exponential smoothing model.

3.3 ARIMA Models

```r
arima_model <- georgia_data |> group_by(Name) |> model(auto_arima = ARIMA(log(Poverty)))
arima_model <- arima_model |> mutate(ideal_model = as.character(auto_arima))
arima_model |> group_by(ideal_model) |> count() |> arrange(desc(n))
```

```
## # A tibble: 14 x 2
## # Groups:   ideal_model [14]
##    ideal_model                  n
##    <chr>                    <int>
##  1 <ARIMA(1,0,0) w/ mean>      61
##  2 <ARIMA(0,1,0)>              58
##  3 <ARIMA(1,1,0)>              14
##  4 <ARIMA(0,0,1) w/ mean>       5
##  5 <ARIMA(0,1,1)>               5
##  6 <ARIMA(0,2,1)>               4
##  7 <ARIMA(3,0,0) w/ mean>       3
##  8 <ARIMA(0,0,0) w/ mean>       2
##  9 <ARIMA(2,0,0) w/ mean>       2
## 10 <ARIMA(0,1,0) w/ drift>      1
## 11 <ARIMA(1,0,1) w/ mean>       1
## 12 <ARIMA(1,1,0) w/ drift>      1
## 13 <ARIMA(1,1,1)>               1
## 14 <ARIMA(3,1,0)>               1
```

Arima(1,0,0) with mean dominating in 61 counties but surprisingly Random Walk model which is Arima(0,1,0) is the second best performing model.

3.4 Cross-Validation

```r
ideal_models <- georgia_data |> stretch_tsibble(.init = 15) |>
  model(ideal_arima = ARIMA(log(Poverty) ~ pdq(1,0,0)+1),
        ideal_ets = ETS(log(Poverty) ~ error("A") + trend("N") + season("N")))
```

```
## Warning in sqrt(diag(best$var.coef)): NaNs produced
## Warning in sqrt(diag(best$var.coef)): NaNs produced

## Warning in wrap_arima(y, order = c(p, d, q), seasonal = list(order = c(P, :
## possible convergence problem: optim gave code = 1
## Warning in wrap_arima(y, order = c(p, d, q), seasonal = list(order = c(P, :
## possible convergence problem: optim gave code = 1

## Warning in sqrt(diag(best$var.coef)): NaNs produced

## Warning in wrap_arima(y, order = c(p, d, q), seasonal = list(order = c(P, :
## possible convergence problem: optim gave code = 1

## Warning in sqrt(diag(best$var.coef)): NaNs produced
## Warning in sqrt(diag(best$var.coef)): NaNs produced
## Warning in sqrt(diag(best$var.coef)): NaNs produced
## Warning in sqrt(diag(best$var.coef)): NaNs produced
## Warning in sqrt(diag(best$var.coef)): NaNs produced
## Warning in sqrt(diag(best$var.coef)): NaNs produced

## Warning in wrap_arima(y, order = c(p, d, q), seasonal = list(order = c(P, :
## possible convergence problem: optim gave code = 1

## Warning in sqrt(diag(best$var.coef)): NaNs produced
## Warning in sqrt(diag(best$var.coef)): NaNs produced

## Warning in wrap_arima(y, order = c(p, d, q), seasonal = list(order = c(P, :
## possible convergence problem: optim gave code = 1

## Warning in sqrt(diag(best$var.coef)): NaNs produced

## Warning: 8 errors (3 unique) encountered for ideal_arima
## [1] Lapack routine dgesv: system is exactly singular: U[1,1] = 0
## [6] non-stationary AR part from CSS
## [1] system is computationally singular: reciprocal condition number = 8.26522e-22
```

```r
ideal_models |> forecast(h = 5) |> accuracy(georgia_data) |> group_by(.model) |>
  summarise(mean_rmse = mean(RMSE))
```

```
## Warning: The future dataset is incomplete, incomplete out-of-sample data will be treated as missing.
## 5 observations are missing between 2023 and 2027

## # A tibble: 2 x 2
##   .model      mean_rmse
##   <chr>           <dbl>
## 1 ideal_arima     1255.
## 2 ideal_ets       1627.
```

From the results of combined RMSE which fits the whole state overall, we can see that ARIMA model which was Arima(1,0,0)~mean is performing better than the ETS/SES model by a significant margin because its mean RMSE is lower than the SES model.

4. FORECASTS

```
winning_model <- georgia_data |> model(ARIMA(log(Poverty) ~ pdq(1,0,0)+1))
winning_pred <- winning_model |> forecast(h=5)
winning_pred
```

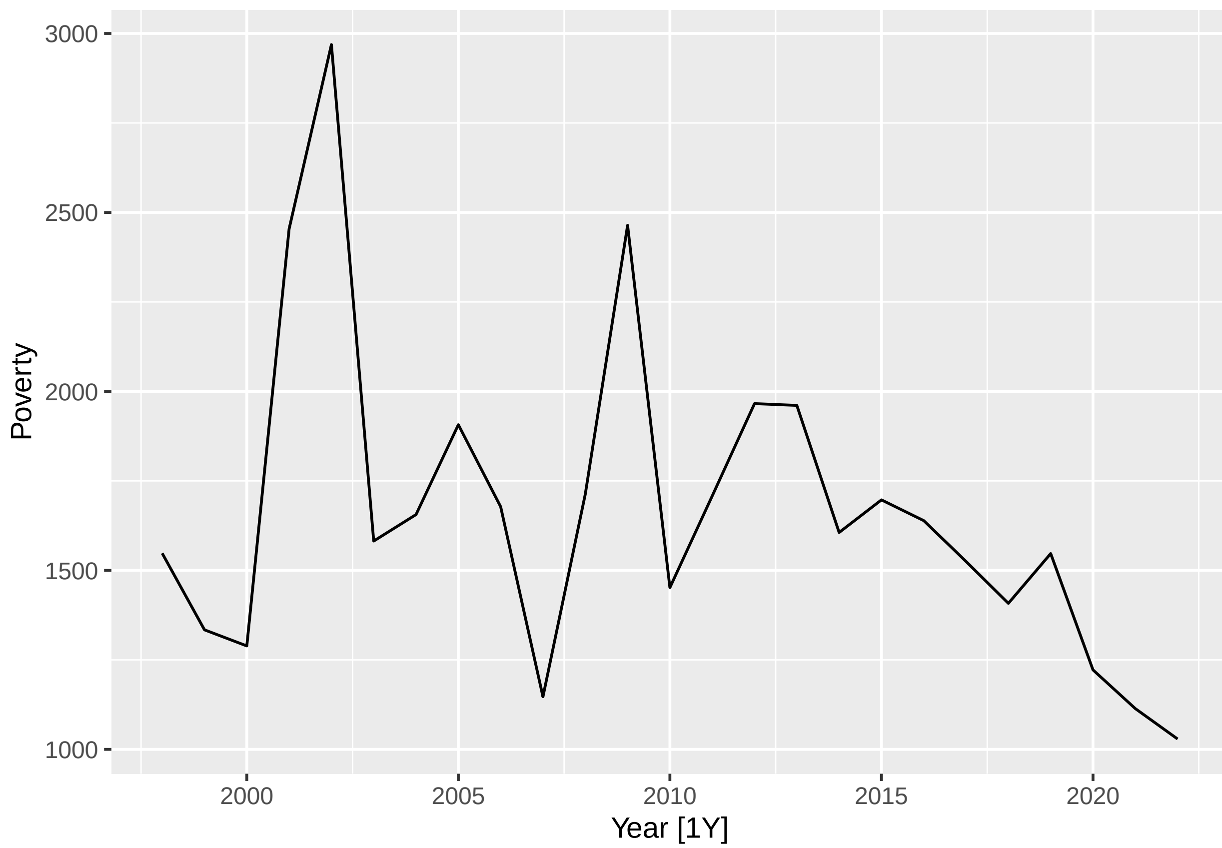
```
## # A fable: 795 x 6 [1Y]
## # Key:     FIPS, Name, .model [159]
##     FIPS Name             .model                         Year         Poverty .mean
##    <int> <chr>            <chr>                         <dbl>          <dist> <dbl>
##  1 13001 Appling County  ARIMA(log(Poverty) ~ pdq(~  2023  t(N(8.2, 0.01)) 3639.
##  2 13001 Appling County  ARIMA(log(Poverty) ~ pdq(~  2024 t(N(8.2, 0.016)) 3631.
##  3 13001 Appling County  ARIMA(log(Poverty) ~ pdq(~  2025 t(N(8.2, 0.018)) 3623.
##  4 13001 Appling County  ARIMA(log(Poverty) ~ pdq(~  2026  t(N(8.2, 0.02)) 3617.
##  5 13001 Appling County  ARIMA(log(Poverty) ~ pdq(~  2027  t(N(8.2, 0.02)) 3612.
##  6 13003 Atkinson County ARIMA(log(Poverty) ~ pdq(~  2023 t(N(7.5, 0.014)) 1785.
##  7 13003 Atkinson County ARIMA(log(Poverty) ~ pdq(~  2024  t(N(7.5, 0.02)) 1832.
##  8 13003 Atkinson County ARIMA(log(Poverty) ~ pdq(~  2025 t(N(7.5, 0.023)) 1863.
##  9 13003 Atkinson County ARIMA(log(Poverty) ~ pdq(~  2026 t(N(7.5, 0.025)) 1885.
## 10 13003 Atkinson County ARIMA(log(Poverty) ~ pdq(~  2027 t(N(7.5, 0.026)) 1900.
## # i 785 more rows
```

```
percent_change <- ((winning_pred |> filter(Year == 2027) |> pull(.mean)) -
  (georgia_data |> filter(Year == 2022) |> pull(Poverty)))*100 / (georgia_data |> filter(Year == 2022)
                                                                  pull(Poverty))

latest_georgia_data <- georgia_data |> filter(Year==2022)  |> mutate(percent_change = percent_change)
latest_georgia_data |> arrange(desc(percent_change))
```

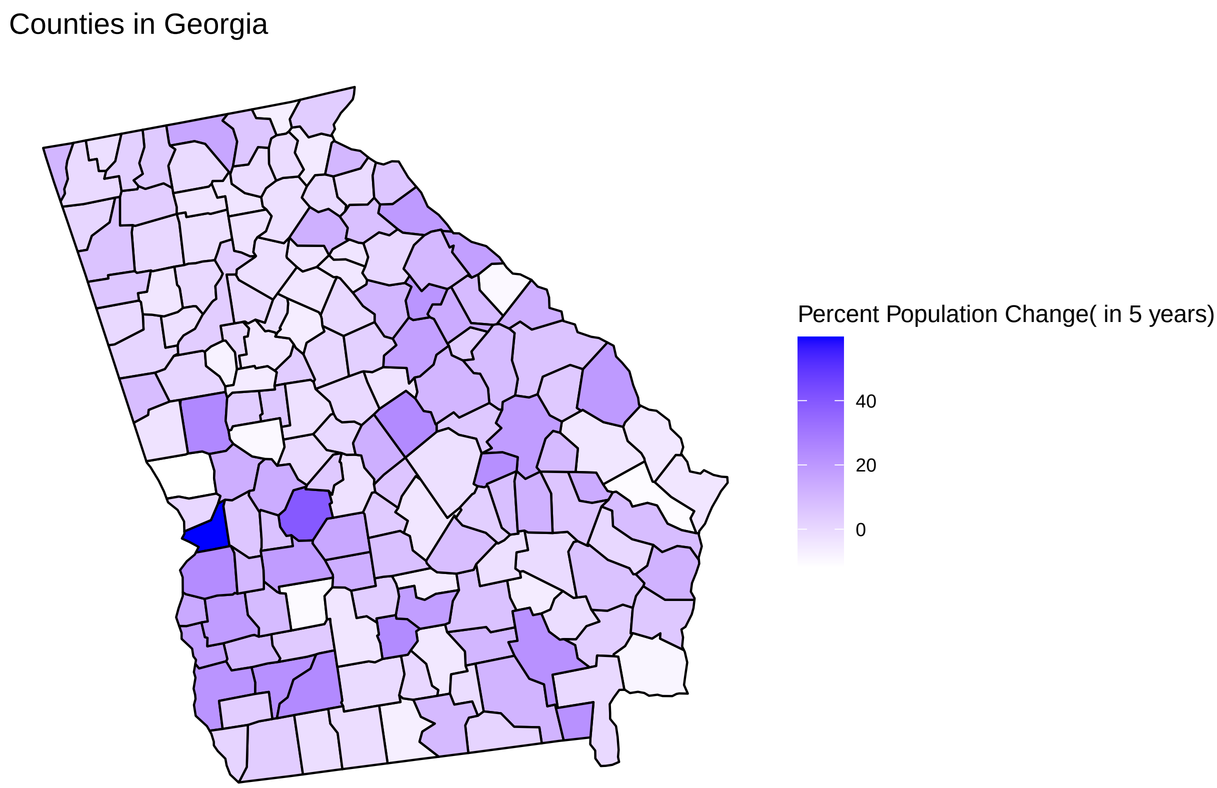
```
## # A tsibble: 159 x 8 [1Y]
## # Key:       FIPS, Name [159]
##     Year  FIPS Name               Population Poverty SNAP_Benefit Poor_Exemption
##    <int> <int> <chr>                   <int>   <int>        <int>          <int>
##  1  2022 13053 Chattahoochee Cou~       6608    1029          721        1890000
##  2  2022 13193 Macon County             9940    2365         2651        1890000
##  3  2022 13199 Meriwether County       20720    3435         4443        1890000
##  4  2022 13205 Mitchell County         19661    4470         5671        1890000
##  5  2022 13319 Wilkinson County         8554    1456         1948        1890000
##  6  2022 13277 Tift County             40365    6697         9153        1890000
##  7  2022 13259 Stewart County           3896    1010         1132        1890000
##  8  2022 13283 Treutlen County          5979    1297         1679        1890000
##  9  2022 13007 Baker County             2779     599          798        1890000
## 10  2022 13299 Ware County             33714    5943         9078        1890000
## # i 149 more rows
## # i 1 more variable: percent_change <dbl>
```

```r
georgia_data |> filter(Name == "Chattahoochee County") |> autoplot(Poverty)
```



The top five counties with the highest percentage increase in poverty over the next five years are Chattahoochee County, Macon County, Meriwether County, Mitchell County and Wilkinson County. The ARIMA model predicts that the Chattahoochee County will have 60% increase in the poverty percentage which might be quite surprising and might feel unbelievable. Also, in the recent years the number in poverty is decreasing in the county but the model predicts an increase in almost 60% in poverty. This is because ARIMA(1,0,0)~mean model forces the predictions to move towards the mean of the data. So, for this county, the poverty is decreasing but the mean is above its range in recent years, so the prediction rises.

```r
latest_georgia_data <- latest_georgia_data |> mutate(fips = FIPS)
plot_usmap(regions = "counties", include = "GA", data = latest_georgia_data, values = "percent_change")
  scale_fill_continuous(low = "white", high ="blue", name = "Percent Population Change( in 5 years)",
                        label = scales::comma)  +
  labs(title = "Counties in Georgia") +
  theme(legend.position = "right")
```


Counties in Georgia
Percent Population Change(in 5 years)
40
20
0