Final Project : Complete Project

Dhyana

2025-05-03

Final Project

State Information

- **State Name**: New Jersey
- **State Abbreviation**: NJ
- **State FIPS Code**: 34

1. Data Preparation

1.1 SAIPE Data

```
SAIPE_NJ <- read.csv("~/Downloads/SAIPE_11-04-2025.csv")
clean_data <- SAIPE_NJ |>
  select(Year, ID, Name, Poverty.Universe, Number.in.Poverty) |>
  rename(
    FIPS = ID,
    County = Name,
    Population = Poverty.Universe,
    Poverty = Number.in.Poverty
  )

clean_data <- clean_data[nchar(clean_data$FIPS) == 5, ] # Exactly 5 characters
clean_data <- clean_data[clean_data$FIPS != 34000, ] # Removing new jersey
# Total number of counties in New Jersey
clean_data |>
  distinct(FIPS, County) |>
  count()
```

```
##    n
## 1 21
```

```
# Converting to numeric
clean_data <- clean_data |>
  filter(Population != "--") |>
  mutate(
    Population = gsub (",", "", Population),
    Population = as.numeric(Population))
```

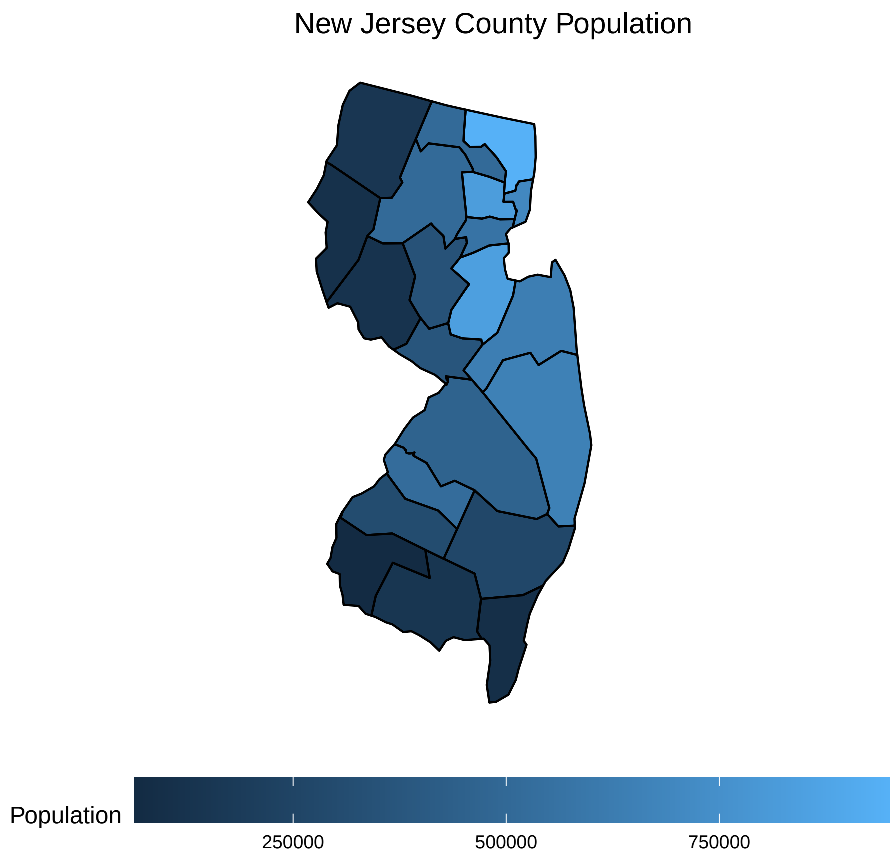
```r
# Largest county
largest_county <- clean_data |>
  filter(Year == 2023) |> # 2023 is the latest year
  arrange(desc(Population)) |>
  slice(1)
largest_county |>
  select(County, Population)
```

```
##          County Population
## 1 Bergen County     948615
```

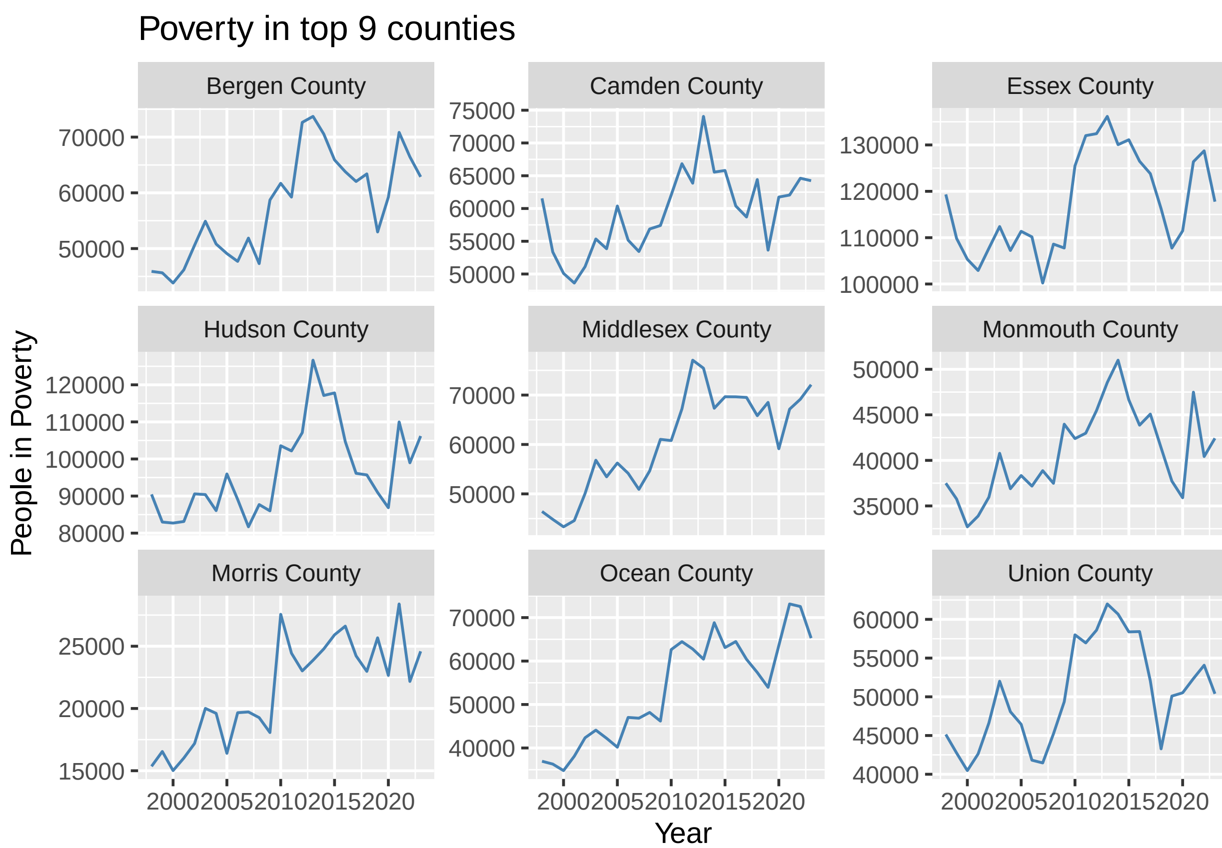
```r
# 9 largest counties
top_9 <- clean_data |>
  filter(Year == 2023) |>
  arrange(desc(Population)) |>
  slice_head(n = 9)
top_9 |>
  select(County, Population)
```

```
##             County Population
## 1    Bergen County     948615
## 2 Middlesex County     841362
## 3     Essex County     828312
## 4    Hudson County     697983
## 5     Ocean County     651520
## 6  Monmouth County     636773
## 7     Union County     567554
## 8    Camden County     521373
## 9    Morris County     507188
```

```r
# Population map
data_2023 <- clean_data[clean_data$Year == 2023, ]
data_2023 <- data_2023 |>
  rename(fips = FIPS)
plot_usmap(
  data = data_2023,
  regions = "counties",
  include = "NJ",
  values = "Population"
) +
  labs(
    title = "New Jersey County Population"
  ) +
  theme(
    legend.position = "bottom",
    legend.key.width = unit(2, "cm"),
    legend.margin = margin(t = 5, unit = "pt")
  )
```




```r
# Time Plot
clean_data$Poverty <- as.numeric(gsub(",", "", clean_data$Poverty))
top9_name <- top_9$County
top9_poverty <- clean_data[clean_data$County %in% top9_name, ]
ggplot(top9_poverty, aes(x = Year, y = Poverty)) +
  geom_line(color = "steelblue") +
  facet_wrap(~ County, scales = "free_y") +
  labs(
    title = "Poverty in top 9 counties",
    x = "Year",
    y = "People in Poverty"
  )
```



1.2 County SNAP Benefits

```r
snap_data <- read_csv("~/Creative Cloud Files/cleaned_cntysnap.csv")
```

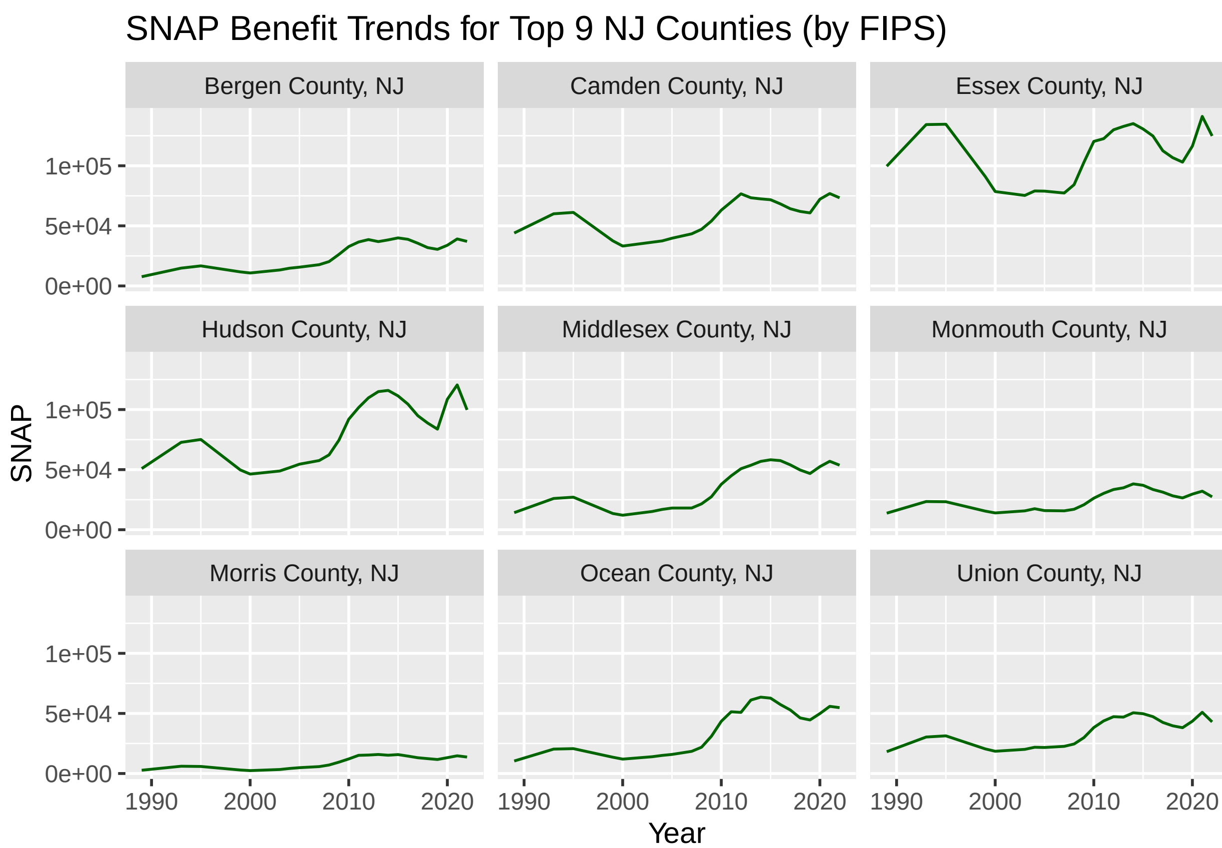
```
## Warning: One or more parsing issues, call `problems()` on your data frame for details,
## e.g.:
##   dat <- vroom(...)
##   problems(dat)

## Rows: 3214 Columns: 32
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (6): Name, Jul-2006, Jul-2002, Jul-2001, Jul-1998, Jul-1997
## dbl  (2): State FIPS code, County FIPS code
## num (24): Jul-2022, Jul-2021, Jul-2020, Jul-2019, Jul-2018, Jul-2017, Jul-20...
##
## 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
snap_data <- snap_data |>
  rename(
    state_fips = `State FIPS code`,
    county_fips = `County FIPS code`,
```

```r
    county_name = Name
  )
SNAP_NJ <- snap_data |>
  filter(state_fips == 34, county_fips != 0) |>
  mutate(FIPS = as.character(state_fips * 1000 + county_fips))
SNAP_NJ <- SNAP_NJ |>
  mutate(across(starts_with("Jul-"), as.numeric))
```

```
## Warning: There were 5 warnings in ‘mutate()‘.
## The first warning was:
## i In argument: ‘across(starts_with("Jul-"), as.numeric)‘.
## Caused by warning:
## ! NAs introduced by coercion
## i Run ‘dplyr::last_dplyr_warnings()‘ to see the 4 remaining warnings.
```

```r
snap_long <- SNAP_NJ |>
  pivot_longer(
    cols = starts_with("Jul-"),
    names_to = "Year",
    values_to = "SNAP"
  ) |>
  mutate(Year = as.numeric(sub("Jul-", "", Year)))
top_9$FIPS <- as.character(top_9$FIPS)
snap_long$FIPS <- as.character(snap_long$FIPS)
top9_fips <- top_9$FIPS # Top 9
top9_snap <- snap_long |>
  filter(FIPS %in% top9_fips)
top9_snap <- top9_snap |>
  filter(!is.na(SNAP))
# Time PLot
ggplot(top9_snap, aes(x = Year, y = SNAP)) +
  geom_line(color = "darkgreen") +
  facet_wrap(~ county_name) +
  labs(
    title = "SNAP Benefit Trends for Top 9 NJ Counties (by FIPS)",
    x = "Year",
    y = "SNAP"
  )
```



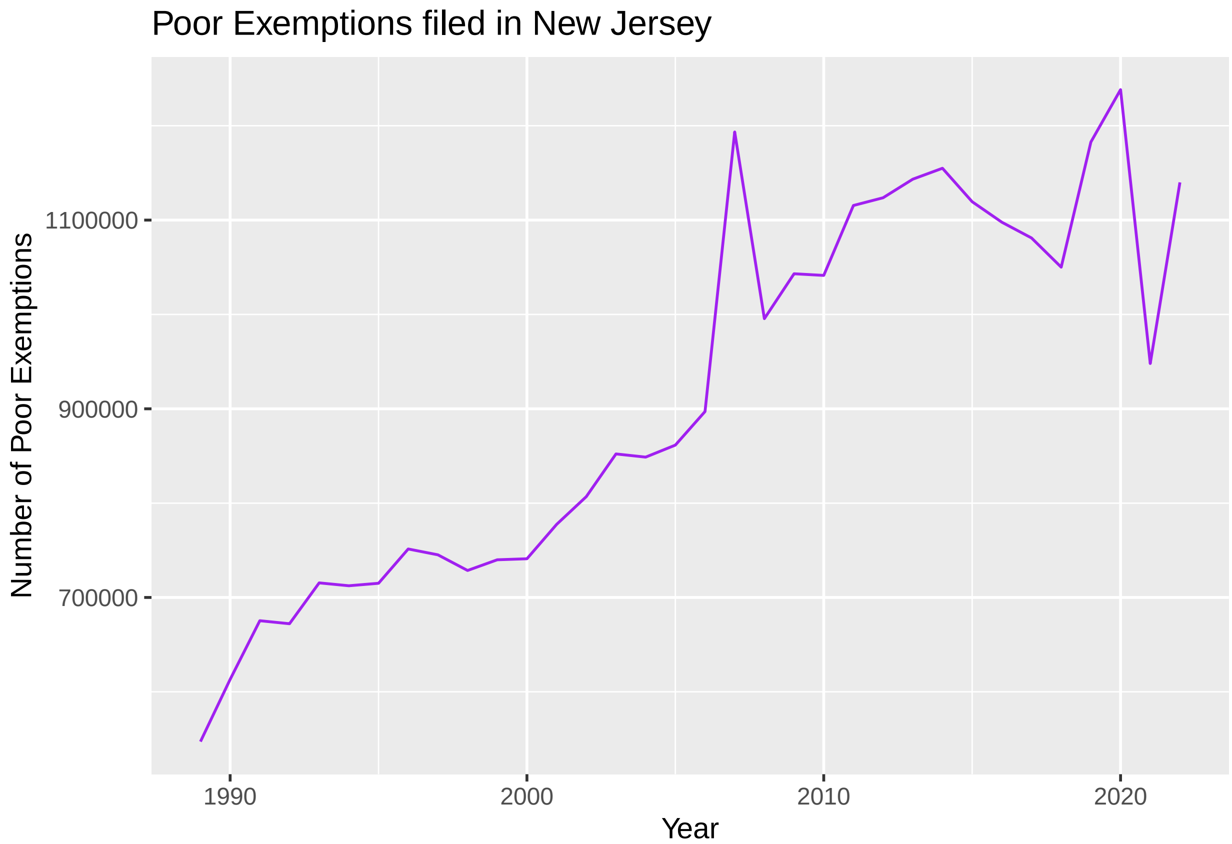
1.3 State IRS Data

```r
irs_data <- read_csv("~/Creative Cloud Files/cleaned_irs.csv")
```

```
## Rows: 1734 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): Name
## dbl  (2): State FIPS code, Year
## num (10): Total exemptions, Poor exemptions, Age 65 and over exemptions, Age...
##
## 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
# Filtering for New Jersey
IRS_NJ <- irs_data |>
  filter(Name == "New Jersey") |>
  mutate(`Poor exemptions` = as.numeric(`Poor exemptions`)) # Converting to numeric
# Time Plot
ggplot(IRS_NJ, aes(x = Year, y = `Poor exemptions`)) +
  geom_line(color = "purple") +
  labs(
    title = "Poor Exemptions filed in New Jersey",
```

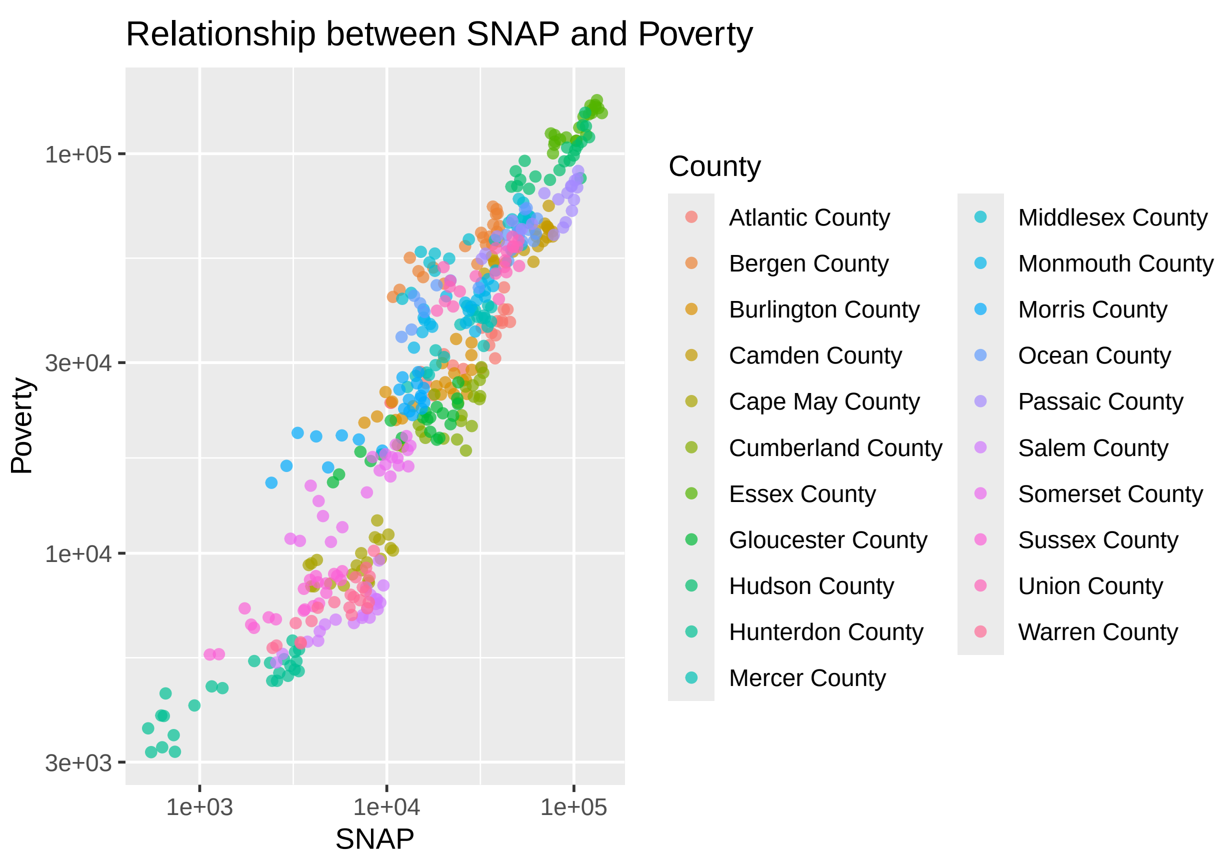
```
    x = "Year",
    y = "Number of Poor Exemptions"
  )
```



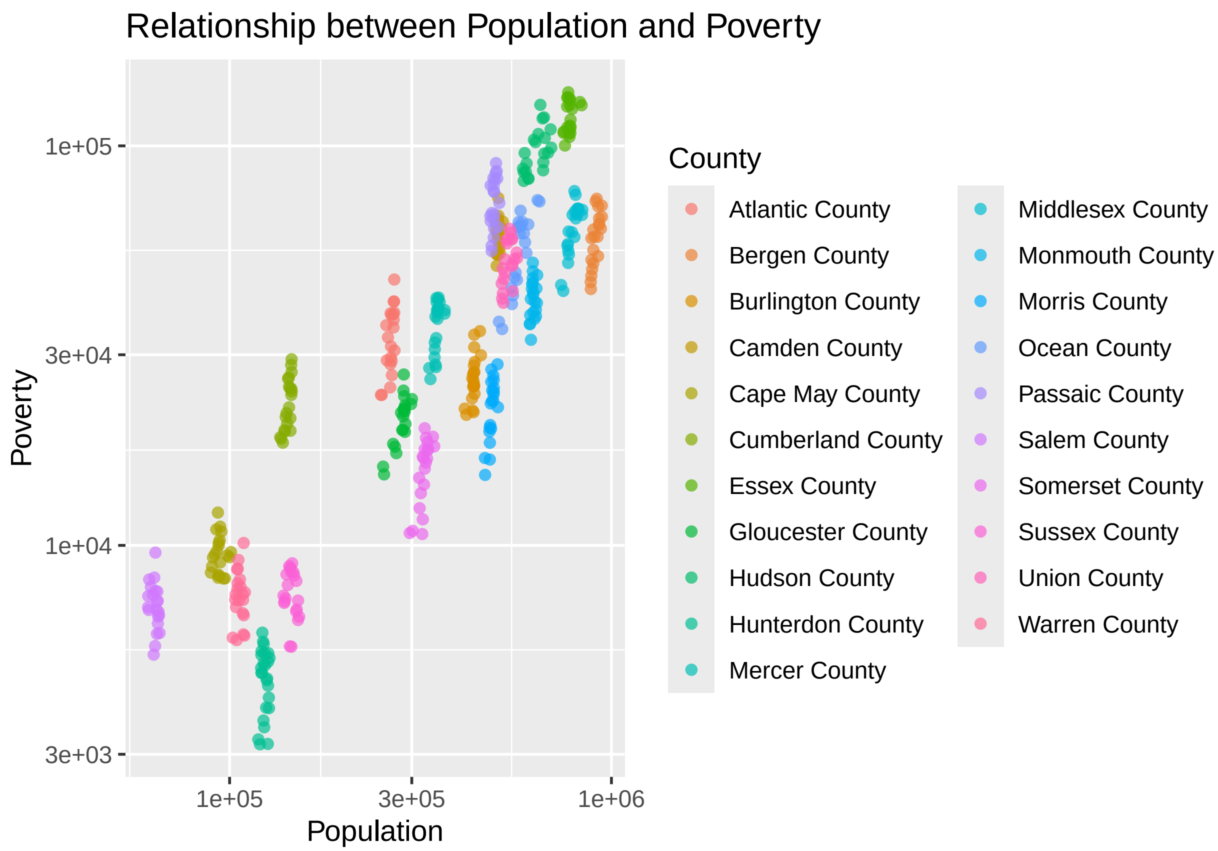
1.4 Merging the data

```
clean_data <- clean_data |>
  mutate(FIPS = as.character(FIPS))
snap_long <- snap_long |>
  mutate(FIPS = as.character(FIPS))
# Merging SAIPE and SNAP
merged_data <- clean_data |>
  filter(Year >= 1997) |>
  left_join(
    snap_long |>
      filter(Year >= 1997) |>
      select(FIPS, Year, SNAP),
    by = c("FIPS", "Year"))
# Adding IRS
merged_data <- merged_data |>
  left_join(
    IRS_NJ |>
      filter(Year >= 1997) |>
```

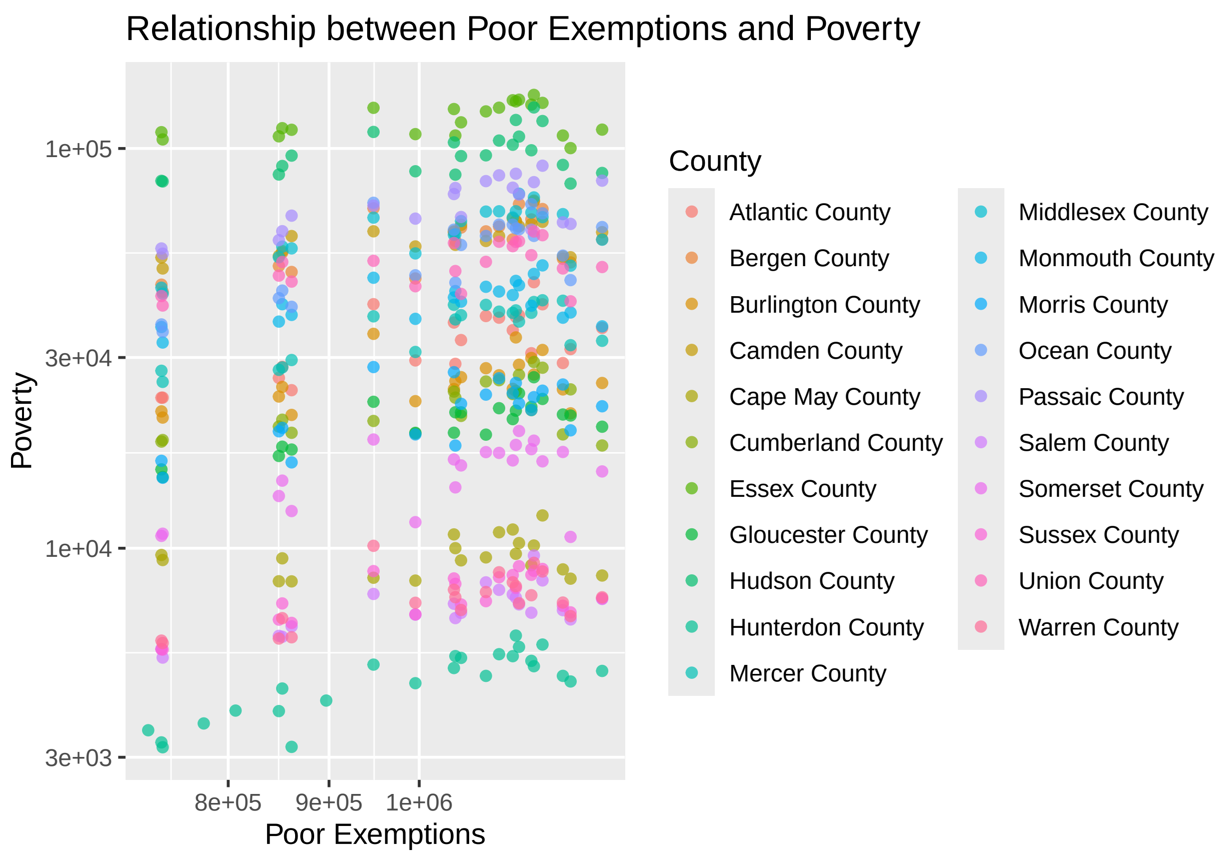
```r
    select(Year, `Poor exemptions`),
    by = "Year"
  )
# Removing missing values
merged_data <- merged_data |>
  filter(
    !is.na(Poverty),
    !is.na(Population),
    !is.na(SNAP),
    !is.na(`Poor exemptions`)
  )
# Converting to a tsibble
merged_tsibble <- merged_data |>
  as_tsibble(key = c(FIPS, County), index = Year)
# Visualization 1 - Poverty vs SNAP
ggplot(merged_tsibble, aes(x = SNAP, y = Poverty, color = County)) +
  geom_point(alpha = 0.7) +
  scale_x_log10() +
  scale_y_log10() +
  labs(
    title = "Relationship between SNAP and Poverty",
    x = "SNAP",
    y = "Poverty"
  )
```



```r
# Visualization 2 - Poverty vs Population
ggplot(merged_tsibble, aes(x = Population, y = Poverty, color = County)) +
  geom_point(alpha = 0.7) +
  scale_x_log10() +
  scale_y_log10() +
  labs(
    title = "Relationship between Population and Poverty",
    x = "Population",
    y = "Poverty"
  )
```




```r
# Visualization 3 - Poverty vs Poor Exemptions
ggplot(merged_tsibble, aes(x = `Poor exemptions`, y = Poverty, color = County)) +
  geom_point(alpha = 0.7) +
  scale_x_log10() +
  scale_y_log10() +
  labs(
    title = "Relationship between Poor Exemptions and Poverty",
    x = "Poor Exemptions",
    y = "Poverty"
  )
```



2. Linear Models

2.1 Variable Selection

```
final_data <- merged_tsibble
# Log-transformed variables
final_data <- final_data |>
  mutate(
    log_poverty = log(Poverty),
    log_population = log(Population),
    log_snap = log(SNAP),
    log_poor_exemptions = log(`Poor exemptions`)
  )
# 7 Linear Models
model_1 <- lm(log_poverty ~ log_population, data = final_data)
model_2 <- lm(log_poverty ~ log_snap, data = final_data)
model_3 <- lm(log_poverty ~ log_poor_exemptions, data = final_data)
model_4 <- lm(log_poverty ~ log_population + log_snap, data = final_data)
model_5 <- lm(log_poverty ~ log_population + log_poor_exemptions, data = final_data)
model_6 <- lm(log_poverty ~ log_snap + log_poor_exemptions, data = final_data)
model_7 <- lm(log_poverty ~ log_population + log_snap + log_poor_exemptions, data = final_data)
models_list <- list(model_1, model_2, model_3, model_4, model_5, model_6, model_7)
# Summary of each model
summary_1 <- glance(model_1)
```

```r
summary_2 <- glance(model_2)
summary_3 <- glance(model_3)
summary_4 <- glance(model_4)
summary_5 <- glance(model_5)
summary_6 <- glance(model_6)
summary_7 <- glance(model_7)
# All summaries into a table
model_summary <- bind_rows(
  summary_1,
  summary_2,
  summary_3,
  summary_4,
  summary_5,
  summary_6,
  summary_7
)
model_summary <- model_summary |>
  mutate(model_number = 1:7) |>
  select(model_number, adj.r.squared, AIC, BIC)
print(model_summary)
```

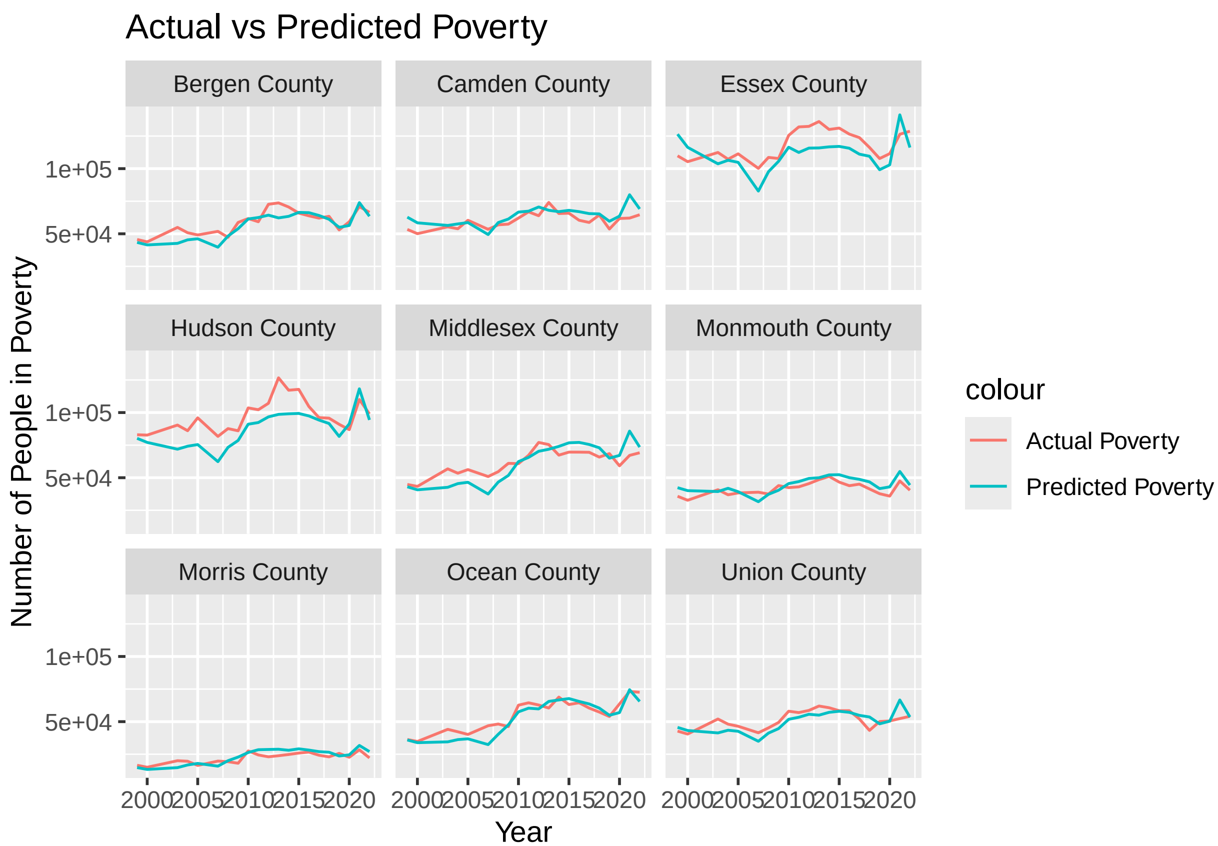
```
## # A tibble: 7 x 4
##   model_number adj.r.squared   AIC   BIC
##          <int>         <dbl> <dbl> <dbl>
## 1            1        0.774   553.  565.
## 2            2        0.872   299.  312.
## 3            3        0.0143 1208. 1220.
## 4            4        0.971  -364. -348.
## 5            5        0.784   533.  550.
## 6            6        0.906   163.  179.
## 7            7        0.981  -537. -516.
```

```r
# Best model
best_model <- model_summary |>
  filter(AIC == min(AIC))
print(paste("Best model is model number:", best_model$model_number))
```

```
## [1] "Best model is model number: 7"
```

```r
best_model <- model_7
# Predicting values
predict_values <- predict(best_model, newdata = final_data)
final_data <- final_data |>
  mutate(predict_log_poverty = predict_values)
# Converting log values back to normal
final_data <- final_data |>
  mutate(
    real_poverty = exp(log_poverty),
    predict_poverty = exp(predict_log_poverty)
  )
# Top 9 Counties
top_9_data <- final_data |>
```

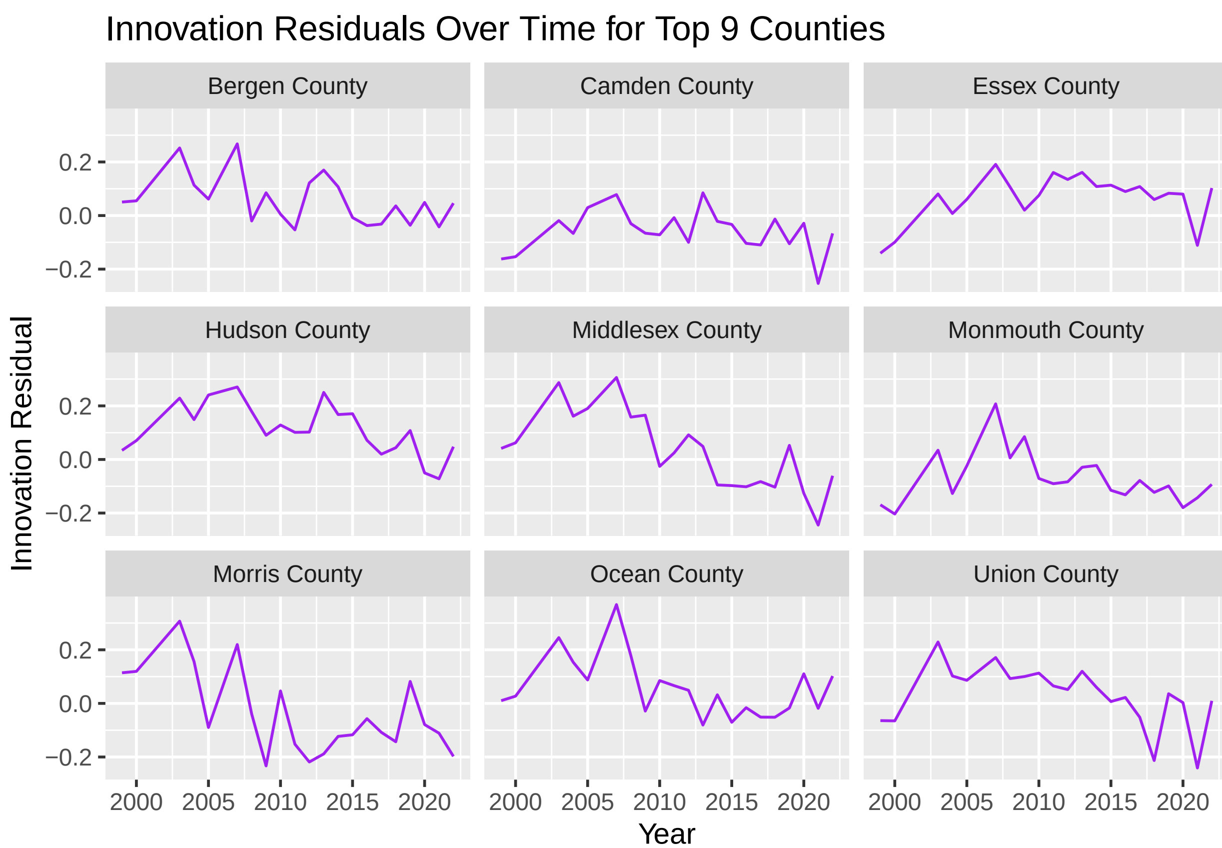
```r
  filter(County %in% top_9$County)
# Plot
ggplot(top_9_data, aes(x = Year)) +
  geom_line(aes(y = real_poverty, color = "Actual Poverty")) +
  geom_line(aes(y = predict_poverty, color = "Predicted Poverty")) +
  facet_wrap((~County)) +
  labs(
    title = "Actual vs Predicted Poverty",
    x = "Year",
    y = "Number of People in Poverty"
  )
```



2.2 Residual Analysis

```r
# Innovation Residuals
top_9_data <- top_9_data |>
  mutate(
    innovation_residual = log_poverty - predict_log_poverty
  )
# Time Plot
ggplot(top_9_data, aes(x = Year, y = innovation_residual)) +
  geom_line(color = "purple") +
  facet_wrap((~County)) +
```

```
  labs(
    title = "Innovation Residuals Over Time for Top 9 Counties",
    x = "Year",
    y = "Innovation Residual"
  )
```




```
# Ljung - Box Test
ljung_box <- top_9_data |>
  as_tibble() |>
  group_by(County) |>
  summarize(
    p_value = Box.test(innovation_residual, lag = 10, type = "Ljung-Box")$p.value
  )
print(ljung_box)
```

```
## # A tibble: 9 x 2
##   County             p_value
##   <chr>                <dbl>
## 1 Bergen County     0.972
## 2 Camden County     0.632
## 3 Essex County      0.412
## 4 Hudson County     0.399
## 5 Middlesex County 0.000253
## 6 Monmouth County   0.729
## 7 Morris County     0.131
```

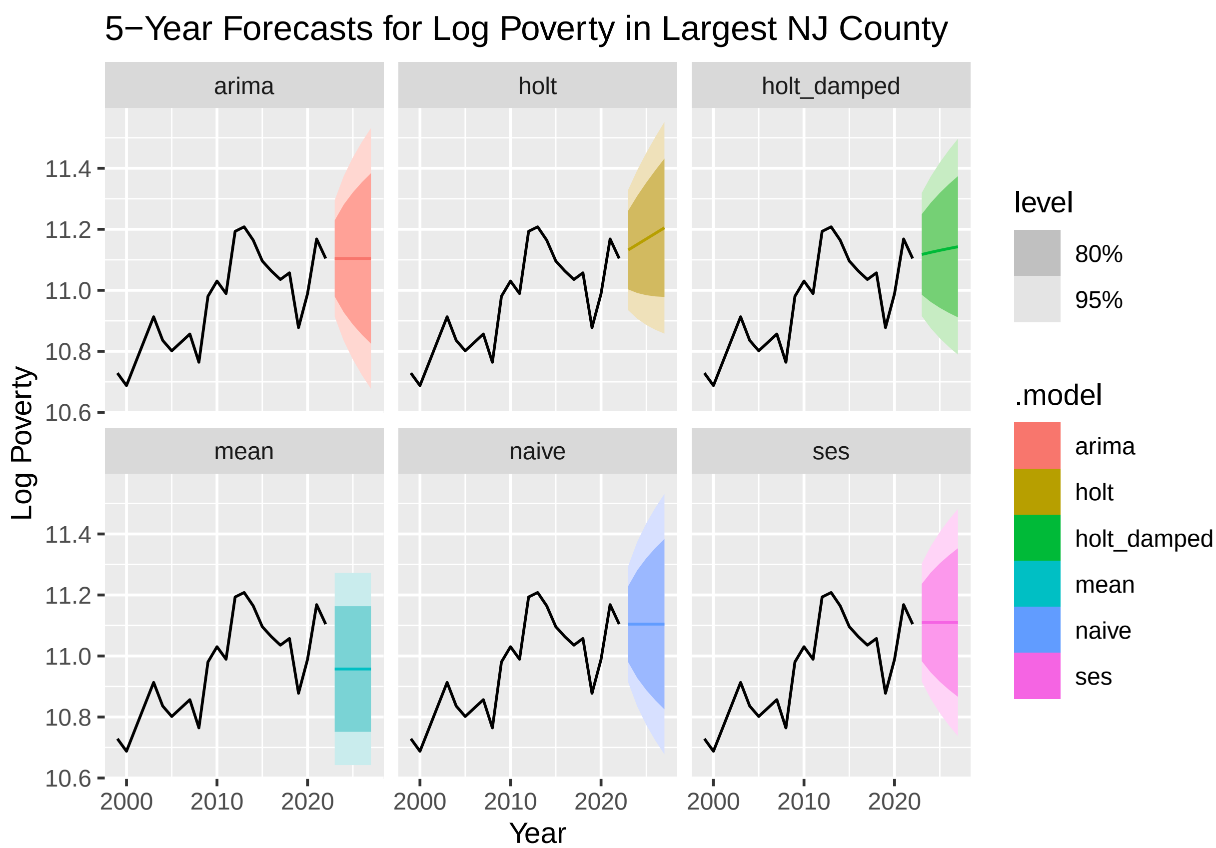
```
## 8 Ocean County      0.0677
## 9 Union County      0.255
```

```r
# Only 1 county has residuals different from white noise that is the Middlesex County
# With a high adjusted R^2 (0.981) and a strong relationship between actual and predicted values,
#the linear model performs well in predicting poverty
# According to the Ljung-Box test, innovation residuals seemed random for 8 of the 9 counties,
#indicating that the model fits the data well generally
```

3. Stochastic Models

3.1 Single County Forecasts

```r
largest_county_name <- largest_county$County[1]
county_data <- final_data |>
  filter(County == largest_county_name) |>
  select(Year, log_poverty)
largest_county_ts <- county_data |>
  as_tsibble(index = Year) |>
  tsibble::fill_gaps()
largest_county_ts <- largest_county_ts |>
  mutate(log_poverty = na.approx(log_poverty, x = Year, na.rm = FALSE))
# Models
county_models <- largest_county_ts |>
  model(
  naive = NAIVE(log_poverty),
  mean = MEAN(log_poverty),
  ses = ETS(log_poverty ~ error("A") + trend("N") + season("N")),
  holt = ETS(log_poverty ~ error("A") + trend("A") + season("N")),
  holt_damped = ETS(log_poverty ~ error("A") + trend("Ad") + season("N")),
  arima = ARIMA(log_poverty)
)
forecasts <- county_models |>
  forecast(h = 5)
autoplot(forecasts, largest_county_ts) +
  labs(
    title = "5-Year Forecasts for Log Poverty in Largest NJ County",
    x = "Year",
    y = "Log Poverty"
  ) +
  facet_wrap(~.model)
```




```r
# Model Quality
model_accuracy <- accuracy(county_models)
print(model_accuracy)
```

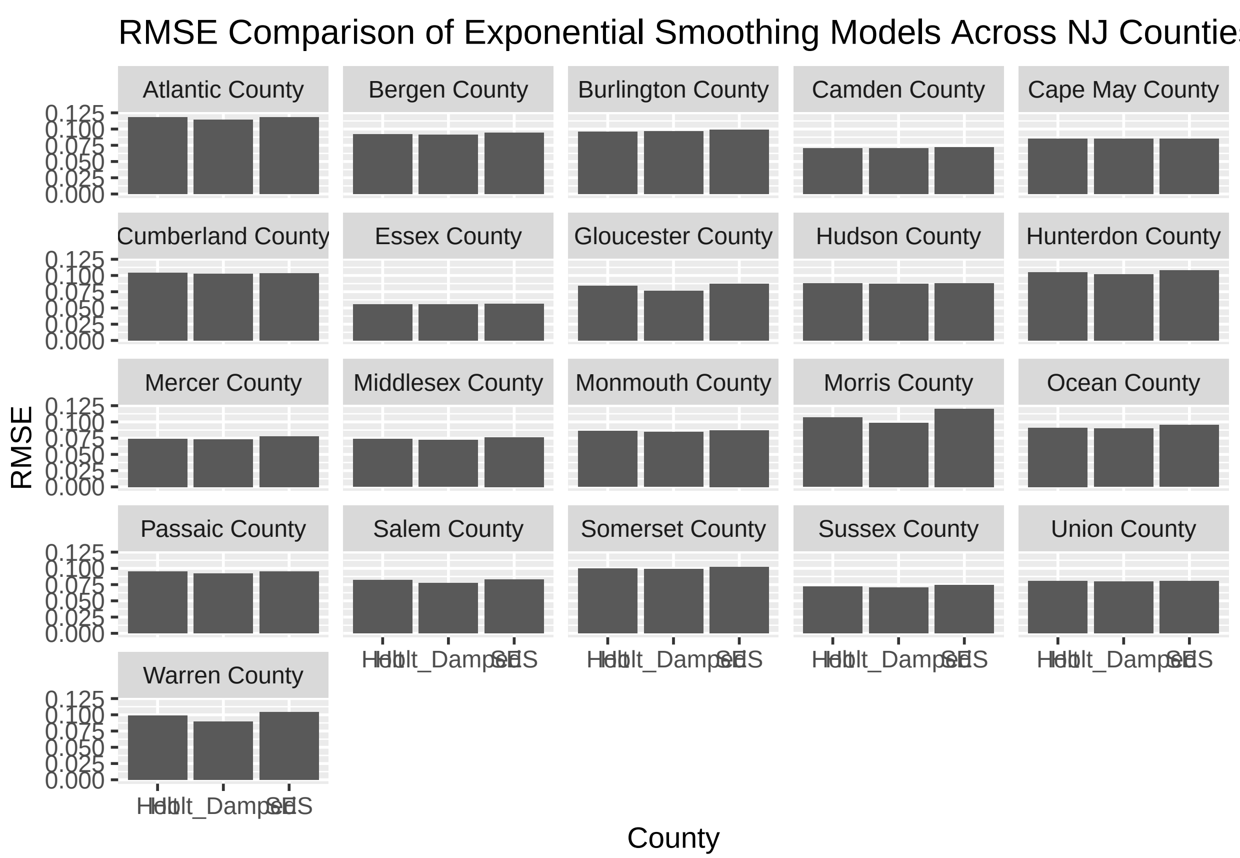
```
## # A tibble: 6 x 10
##   .model      .type           ME   RMSE    MAE      MPE  MAPE  MASE RMSSE    ACF1
##   <chr>       <chr>        <dbl>  <dbl>  <dbl>    <dbl> <dbl> <dbl> <dbl>   <dbl>
## 1 naive       Traini~  1.63e- 2 0.0975 0.0774  0.146    0.705 1     1     -0.155
## 2 mean        Traini~ -5.74e-16 0.154  0.136  -0.0198   1.24  1.76  1.58   0.744
## 3 ses         Traini~  1.95e- 2 0.0944 0.0743  0.174    0.676 0.960 0.968 -0.0222
## 4 holt        Traini~ -1.26e- 3 0.0920 0.0745 -0.0151   0.679 0.963 0.944  0.0556
## 5 holt_damped Traini~  5.18e- 5 0.0916 0.0729 -0.00399 0.663 0.941 0.939  0.0444
## 6 arima       Traini~  1.61e- 2 0.0954 0.0746  0.144    0.680 0.964 0.979 -0.154
```

```r
# Among all models, the Holt's damped trend method performed best
# It has lowest RMSE and MAE
# It's forecast trend is stable, with tighter confidence intervals compared to
# ARIMA and other methods
```

3.2 Exponential Smoothing Models

```r
final_data <- final_data |>
  mutate(log_poverty = log(Poverty))
```

```r
exp_models <- final_data |>
  as_tsibble(key = c(FIPS, County), index = Year) |>
  tsibble::fill_gaps() |>
  group_by(County) |>
  mutate(log_poverty = zoo::na.approx(log_poverty, x = Year, na.rm = FALSE)) |>
  ungroup()
exp_models_fitted <- exp_models |>
  model(
    SES = ETS(log_poverty ~ error("A") + trend("N") + season("N")),
    Holt = ETS(log_poverty ~ error("A") + trend("A") + season("N")),
    Holt_Damped = ETS(log_poverty ~ error("A") + trend("Ad") + season("N"))
  )
model_accuracy <- accuracy(exp_models_fitted)
ggplot(model_accuracy, aes(x = .model, y = RMSE, fill.model)) +
  geom_col() +
  facet_wrap(~ County) +
  labs(
    title = "RMSE Comparison of Exponential Smoothing Models Across NJ Counties",
    x = "County",
    y = "RMSE",
    color = "Model"
  )
```




```r
# I selected the Holt's damped trend model as it has the lowest RMSE in most
# counties, indicating the best overall forecast accuracy for poverty trends in
```

```r
# New Jersey
```

3.3 ARIMA Models

```r
arima_models <- final_data |>
  as_tsibble(key = c(FIPS, County), index = Year) |>
  tsibble::fill_gaps() |>
  group_by(County) |>
  mutate(log_poverty = zoo::na.approx(log_poverty, x = Year, na.rm = FALSE)) |>
  ungroup() |>
  model(auto_arima = ARIMA(log_poverty))
# ARIMA structure for each county
for (i in 1:nrow(arima_models)) {
  cat("County:", arima_models$County[i], "\n")
  print(report(arima_models$auto_arima[[i]]))
  cat("\n")
}
```

```
## County: Atlantic County
## Series: log_poverty
## Model: ARIMA(0,1,0)
##
## sigma^2 estimated as 0.01531:  log likelihood=15.43
## AIC=-28.86   AICc=-28.67   BIC=-27.73
## NULL
##
## County: Bergen County
## Series: log_poverty
## Model: ARIMA(0,1,0)
##
## sigma^2 estimated as 0.009504:  log likelihood=20.91
## AIC=-39.83   AICc=-39.64   BIC=-38.69
## NULL
##
## County: Burlington County
## Series: log_poverty
## Model: ARIMA(0,1,0)
##
## sigma^2 estimated as 0.01096:  log likelihood=19.28
## AIC=-36.56   AICc=-36.37   BIC=-35.42
## NULL
##
## County: Camden County
## Series: log_poverty
## Model: ARIMA(1,1,0)
##
## Coefficients:
##           ar1
##       -0.4810
## s.e.   0.1788
##
```

```
## sigma^2 estimated as 0.005338:  log likelihood=27.93
## AIC=-51.87   AICc=-51.27   BIC=-49.6
## NULL
##
## County: Cape May County
## Series: log_poverty
## Model: ARIMA(1,0,0) w/ mean
##
## Coefficients:
##          ar1  constant
##       0.6039    3.6242
## s.e.  0.1535    0.0155
##
## sigma^2 estimated as 0.007112:  log likelihood=26.11
## AIC=-46.23   AICc=-45.03   BIC=-42.7
## NULL
##
## County: Cumberland County
## Series: log_poverty
## Model: ARIMA(1,0,0) w/ mean
##
## Coefficients:
##          ar1  constant
##       0.7138    2.8601
## s.e.  0.1383    0.0186
##
## sigma^2 estimated as 0.01077:  log likelihood=21.01
## AIC=-36.01   AICc=-34.81   BIC=-32.48
## NULL
##
## County: Essex County
## Series: log_poverty
## Model: ARIMA(1,0,0) w/ mean
##
## Coefficients:
##          ar1  constant
##       0.7961    2.3804
## s.e.  0.1157    0.0097
##
## sigma^2 estimated as 0.003238:  log likelihood=35.28
## AIC=-64.56   AICc=-63.36   BIC=-61.02
## NULL
##
## County: Gloucester County
## Series: log_poverty
## Model: ARIMA(0,1,0)
##
## sigma^2 estimated as 0.008564:  log likelihood=22.11
## AIC=-42.22   AICc=-42.03   BIC=-41.09
## NULL
##
## County: Hudson County
## Series: log_poverty
## Model: ARIMA(1,0,0) w/ mean
```

```
##
## Coefficients:
##          ar1  constant
##       0.7020    3.4148
## s.e.  0.1398    0.0160
##
## sigma^2 estimated as 0.007986:  log likelihood=24.61
## AIC=-43.22   AICc=-42.02   BIC=-39.69
## NULL
##
## County: Hunterdon County
## Series: log_poverty
## Model: ARIMA(0,1,0)
##
## sigma^2 estimated as 0.0128:  log likelihood=18.25
## AIC=-34.49   AICc=-34.31   BIC=-33.31
## NULL
##
## County: Mercer County
## Series: log_poverty
## Model: ARIMA(0,1,1)
##
## Coefficients:
##           ma1
##       -0.3123
## s.e.   0.1665
##
## sigma^2 estimated as 0.006618:  log likelihood=25.54
## AIC=-47.08   AICc=-46.48   BIC=-44.81
## NULL
##
## County: Middlesex County
## Series: log_poverty
## Model: ARIMA(0,1,0)
##
## sigma^2 estimated as 0.006126:  log likelihood=25.97
## AIC=-49.94   AICc=-49.75   BIC=-48.8
## NULL
##
## County: Monmouth County
## Series: log_poverty
## Model: ARIMA(1,0,0) w/ mean
##
## Coefficients:
##          ar1  constant
##       0.6350    3.8718
## s.e.  0.1536    0.0165
##
## sigma^2 estimated as 0.008036:  log likelihood=24.62
## AIC=-43.24   AICc=-42.04   BIC=-39.7
## NULL
##
## County: Morris County
## Series: log_poverty
```

```
## Model: ARIMA(0,1,1) w/ drift
##
## Coefficients:
##           ma1  constant
##       -0.7347    0.0197
## s.e.   0.2199    0.0074
##
## sigma^2 estimated as 0.01418:  log likelihood=16.97
## AIC=-27.94   AICc=-26.67   BIC=-24.53
## NULL
##
## County: Ocean County
## Series: log_poverty
## Model: ARIMA(0,1,0)
##
## sigma^2 estimated as 0.009453:  log likelihood=20.98
## AIC=-39.95   AICc=-39.76   BIC=-38.82
## NULL
##
## County: Passaic County
## Series: log_poverty
## Model: ARIMA(0,1,0)
##
## sigma^2 estimated as 0.009788:  log likelihood=20.58
## AIC=-39.15   AICc=-38.96   BIC=-38.02
## NULL
##
## County: Salem County
## Series: log_poverty
## Model: ARIMA(0,1,1)
##
## Coefficients:
##           ma1
##       -0.3638
## s.e.   0.1712
##
## sigma^2 estimated as 0.007441:  log likelihood=24.17
## AIC=-44.34   AICc=-43.74   BIC=-42.07
## NULL
##
## County: Somerset County
## Series: log_poverty
## Model: ARIMA(0,1,0)
##
## sigma^2 estimated as 0.01085:  log likelihood=19.39
## AIC=-36.78   AICc=-36.59   BIC=-35.64
## NULL
##
## County: Sussex County
## Series: log_poverty
## Model: ARIMA(0,1,0)
##
## sigma^2 estimated as 0.005761:  log likelihood=26.67
## AIC=-51.34   AICc=-51.15   BIC=-50.21
```

```
## NULL
##
## County: Union County
## Series: log_poverty
## Model: ARIMA(1,0,0) w/ mean
##
## Coefficients:
##          ar1  constant
##       0.7877    2.2956
## s.e.  0.1201    0.0140
##
## sigma^2 estimated as 0.006625:  log likelihood=26.71
## AIC=-47.42   AICc=-46.22   BIC=-43.88
## NULL
##
## County: Warren County
## Series: log_poverty
## Model: ARIMA(0,1,1)
##
## Coefficients:
##           ma1
##       -0.4713
## s.e.   0.1805
##
## sigma^2 estimated as 0.01182:  log likelihood=18.79
## AIC=-33.58   AICc=-32.98   BIC=-31.31
## NULL
```

```r
# The most commonly selected models by auto ARIMA across New Jersey counties are
# ARIMA(0,1,0) = selected for 9 counties
# ARIMA(1,0,0) with mean = selected for 6 counties
# ARIMA(0,1,1) and variations = selected for 4 counties
# 3 ARIMA Models
common_models_data <- final_data |>
  as_tsibble(key = c(FIPS, County), index = Year) |>
  tsibble::fill_gaps() |>
  group_by(County) |>
  mutate(log_poverty = zoo::na.approx(log_poverty, x = Year, na.rm = FALSE)) |>
  ungroup()
common_models <- common_models_data |>
  model(
    ARIMA_010 = ARIMA(log_poverty ~ pdq(0,1,0)),
    ARIMA_100_mean = ARIMA(log_poverty ~ pdq(1,0,0)),
    ARIMA_011 = ARIMA(log_poverty ~ pdq(0,1,1))
    )
# Model Quality
comparison <- accuracy(common_models)
comparison_summary <- comparison |>
  group_by(.model) |>
  summarise(mean_RMSE = mean(RMSE, na.rm = TRUE)) |>
  arrange(mean_RMSE)
print(comparison_summary)
```

```
## # A tibble: 3 x 2
```

```
##   .model         mean_RMSE
##   <chr>              <dbl>
## 1 ARIMA_011         0.0904
## 2 ARIMA_100_mean    0.0908
## 3 ARIMA_010         0.0945
```

```
# ARIMA(0,1,1) is the best model because it has the lowest RMSE
```

3.4 Cross Validation

```
cross_validation_data <- final_data |>
  as_tsibble(key = c(FIPS, County), index = Year) |>
  tsibble::fill_gaps() |>
  group_by(County) |>
  filter(!is.na(Poverty)) |>
  mutate(log_poverty = log(Poverty)) |>
  filter(!is.infinite(log_poverty)) |>
  ungroup() |>
  stretch_tsibble(.init = 10, .step = 1)
cross_validation_model <- cross_validation_data |>
  model(
    ETS = ETS(log_poverty ~ error("A") + trend("Ad") + season("N")),
    ARIMA = ARIMA(log_poverty ~ pdq(1,0,0))
  )
```

```
## Warning: 240 errors (1 unique) encountered for ETS
## [240] .data contains implicit gaps in time. You should check your data and convert implicit gaps into
```

```
## Warning: 240 errors (1 unique) encountered for ARIMA
## [240] .data contains implicit gaps in time. You should check your data and convert implicit gaps into
```

```
# Forecast 5 years
cross_validation_forecast <- cross_validation_model |>
  forecast(h = 5)
cross_validation_accuracy_data <- final_data |>
  as_tsibble(key = c(FIPS, County), index = Year)
cross_validation_accuracy <- cross_validation_forecast |>
  accuracy(cross_validation_accuracy_data)
```

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

```
rmse_model <- cross_validation_accuracy |>
  group_by(.model) |>
  summarise(mean_RMSE = mean(RMSE, na.rm = TRUE)) |>
  arrange(mean_RMSE)
print(rmse_model)
```

```
## # A tibble: 2 x 2
##   .model mean_RMSE
##   <chr>      <dbl>
## 1 ETS        0.144
## 2 ARIMA      0.177
```

```
# ETS is the better choice for statewide poverty forecasting as it has the
# lowest RMSE that is 0.144
```

4. Forecasts

```
forecast_data <- final_data |>
  as_tsibble(key = c(FIPS, County), index = Year) |>
  tsibble::fill_gaps() |>
  group_by(County) |>
  mutate(
    log_poverty = log(Poverty),
    log_poverty = na.approx(log_poverty, x = Year, na.rm = FALSE)
  ) |>
  ungroup() |>
  filter(!is.infinite(log_poverty))
ets_forecast_model <- forecast_data |>
  model(ETS = ETS(log_poverty ~ error("A") + trend("Ad") + season("N")))
ets_forecast <- ets_forecast_model |>
  forecast(h = 5)
latest_year <- max(forecast_data$Year)
forecast_2028 <- ets_forecast |>
  filter(Year == latest_year + 5) |>
  as_tibble() |>
  mutate(predicted_poverty = exp(.mean)) |>
  select(FIPS, County, predicted_poverty)
current_data <- final_data |>
  filter(Year == latest_year) |>
  select(FIPS, County, Poverty, Population)
poverty_change <- forecast_2028 |>
  left_join(current_data, by = c("FIPS", "County")) |>
  mutate(
    increase = predicted_poverty - Poverty,
    percent_increase = 100 * (increase / Population)
  )
# Top 5 Counties
top_5_counties <- poverty_change |>
  arrange(desc(percent_increase)) |>
  slice_head(n = 5)
print(top_5_counties |> select(County, percent_increase))
```

```
## # A tibble: 5 x 2
##   County          percent_increase
##   <chr>                      <dbl>
## 1 Atlantic County             1.49
## 2 Ocean County                1.38
```

```
## 3 Passaic County              1.21
## 4 Warren County               0.868
## 5 Morris County               0.859
```

```
# Map
map_data <- poverty_change |>
  mutate(fips = FIPS) |>
  select(fips, percent_increase)
# Plot
plot_usmap(
  data = map_data,
  regions = "counties",
  include = "NJ",
  values = "percent_increase"
) +
  labs(
    title = "Forecasted 5-Year % Increase in Poverty by NJ County",
    fill = "% Increase"
  )
```