Next Steps

Sanjeev Subramanian

2025-05-23

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(readr)
library(stringr)
library(lmtest)
```

```
## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
library(ggplot2)
library(patchwork)
library(cluster)
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```r
edu_reg    <- read_csv("Data/Counties_Education_All_Counties.csv")
```

```
## Rows: 3241 Columns: 9

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): FIPS, Name
## dbl (7): 2023 Rural-urban Continuum Code*, 1970, 1980, 1990, 2000, 2008-2012...
##
## 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
income_reg <- read_csv("Data/County_Per_Capita_Income_Ratios.csv")
```

```
## Rows: 3118 Columns: 26
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (3): GeoFIPS, GeoName, Description
## dbl (23): 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, ...
##
## 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.
```

```
income_filtered_reg <- income_reg %>%
  filter(str_detect(Description, regex("income ratio", ignore_case = TRUE)))

edu_reg  <- edu_reg  %>% mutate(FIPS = str_pad(str_trim(FIPS), 5, pad = "0"))
income_filtered_reg <- income_filtered_reg %>%
  mutate(FIPS = str_pad(str_trim(GeoFIPS), 5, pad = "0"))

edu_clean_reg <- edu_reg %>%
  select(FIPS,
         edu_2000       = `2000`,
         edu_2008_2012  = `2008-2012`,
         edu_2019_2023  = `2019-2023`)

income_clean_reg <- income_filtered_reg %>%
  select(FIPS,
         inc_2001 = `2001`,
         inc_2010 = `2010`,
         inc_2019 = `2019`)

merged_reg <- inner_join(edu_clean_reg, income_clean_reg, by = "FIPS")

df_2001 <- merged_reg %>%
  select(FIPS, inc_2001, edu_2000) %>%
  na.omit() %>%
  filter(is.finite(inc_2001) & is.finite(edu_2000) & edu_2000 > 0 & inc_2001 > 0)

df_2010 <- merged_reg %>%
  select(FIPS, inc_2010, edu_2008_2012) %>%
  na.omit() %>%
  filter(is.finite(inc_2010) & is.finite(edu_2008_2012) & edu_2008_2012 > 0 & inc_2010 > 0)

df_2019 <- merged_reg %>%
  select(FIPS, inc_2019, edu_2019_2023) %>%
  na.omit() %>%
  filter(is.finite(inc_2019) & is.finite(edu_2019_2023) & edu_2019_2023 > 0 & inc_2019 > 0)

model_2001 <- lm(log(inc_2001) ~ log(edu_2000), data = df_2001)
model_2010 <- lm(log(inc_2010) ~ log(edu_2008_2012), data = df_2010)
model_2019 <- lm(log(inc_2019) ~ log(edu_2019_2023), data = df_2019)

z_test_coeffs <- function(model1, model2, coeff_name1, coeff_name2) {
  summary1 <- summary(model1)
  summary2 <- summary(model2)

  if (!coeff_name1 %in% rownames(summary1$coefficients)) {
    stop(paste("Coefficient", coeff_name1, "not found in model1 summary."))
```

```r
  }
  if (!coeff_name2 %in% rownames(summary2$coefficients)) {
    stop(paste("Coefficient", coeff_name2, "not found in model2 summary."))
  }

  b1 <- summary1$coefficients[coeff_name1, "Estimate"]
  se1 <- summary1$coefficients[coeff_name1, "Std. Error"]

  b2 <- summary2$coefficients[coeff_name2, "Estimate"]
  se2 <- summary2$coefficients[coeff_name2, "Std. Error"]

  z_stat <- (b1 - b2) / sqrt(se1^2 + se2^2)
  p_value <- 2 * pnorm(-abs(z_stat))

  return(list(b1 = b1, b2 = b2, se1 = se1, se2 = se2, z_stat = z_stat, p_value = p_value))
}

coeff_name_m2001 <- "log(edu_2000)"
coeff_name_m2010 <- "log(edu_2008_2012)"
coeff_name_m2019 <- "log(edu_2019_2023)"

z_2001_2010 <- z_test_coeffs(model_2001, model_2010, coeff_name_m2001, coeff_name_m2010)
cat("Z-test for difference in log education coefficients (2001 vs 2010):\n")
```

```
## Z-test for difference in log education coefficients (2001 vs 2010):
```

```r
print(z_2001_2010)
```

```
## $b1
## [1] 0.3840932
##
## $b2
## [1] 0.3635128
##
## $se1
## [1] 0.006534761
##
## $se2
## [1] 0.007174538
##
## $z_stat
## [1] 2.120716
##
## $p_value
## [1] 0.03394574
```

```r
z_2001_2019 <- z_test_coeffs(model_2001, model_2019, coeff_name_m2001, coeff_name_m2019)
cat("\nZ-test for difference in log education coefficients (2001 vs 2019):\n")
```

```
##
## Z-test for difference in log education coefficients (2001 vs 2019):
```

```r
print(z_2001_2019)
```

```
## $b1
## [1] 0.3840932
```

```
##
## $b2
## [1] 0.4038046
##
## $se1
## [1] 0.006534761
##
## $se2
## [1] 0.007713133
##
## $z_stat
## [1] -1.949847
##
## $p_value
## [1] 0.05119439
```

```r
z_2010_2019 <- z_test_coeffs(model_2010, model_2019, coeff_name_m2010, coeff_name_m2019)
cat("\nZ-test for difference in log education coefficients (2010 vs 2019):\n")
```

```
##
## Z-test for difference in log education coefficients (2010 vs 2019):
```

```r
print(z_2010_2019)
```

```
## $b1
## [1] 0.3635128
##
## $b2
## [1] 0.4038046
##
## $se1
## [1] 0.007174538
##
## $se2
## [1] 0.007713133
##
## $z_stat
## [1] -3.824908
##
## $p_value
## [1] 0.0001308208
```

```r
raw_income_data <- read_csv("Data/County_Only_Full_Income_Data.csv", col_types = cols(.default = "c"))

per_capita_income <- raw_income_data %>%
  filter(LineCode == "30") %>%
  select(GeoFIPS, `2001`, `2023`) %>%
  rename(FIPS = GeoFIPS, income_2001 = `2001`, income_2023 = `2023`) %>%
  mutate(FIPS = str_pad(str_trim(FIPS), 5, pad = "0"),
         income_2001 = as.numeric(income_2001),
         income_2023 = as.numeric(income_2023)) %>%
  na.omit()

edu_data_full <- read_csv("Data/Counties_Education_All_Counties.csv")
```

```
## Rows: 3241 Columns: 9
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): FIPS, Name
## dbl (7): 2023 Rural-urban Continuum Code*, 1970, 1980, 1990, 2000, 2008-2012...
##
## 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.
```

```
edu_data_cleaned <- edu_data_full %>%
  mutate(FIPS = str_pad(str_trim(FIPS), 5, pad = "0")) %>%
  select(FIPS,
         Name,
         edu_2000 = `2000`,
         edu_2019_2023 = `2019-2023`) %>%
  mutate(edu_2000 = as.numeric(edu_2000),
         edu_2019_2023 = as.numeric(edu_2019_2023)) %>%
  na.omit()

income_growth_data <- per_capita_income %>%
  filter(income_2001 > 0 & income_2023 > 0) %>%
  mutate(
    growth_income_county = (income_2023 / income_2001)^(1/(2023-2001-1))
  ) %>%
  select(FIPS, income_2001, income_2023, growth_income_county)

cat("Summary of calculated income growth:\n")
```

```
## Summary of calculated income growth:
```

```
summary(income_growth_data$growth_income_county)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##
```

```
head(income_growth_data)
```

```
## # A tibble: 0 x 4
## # i 4 variables: FIPS <chr>, income_2001 <dbl>, income_2023 <dbl>,
## #   growth_income_county <dbl>
```

```
education_change_data <- edu_data_cleaned %>%
  mutate(education_change = edu_2019_2023 - edu_2000) %>%
  select(FIPS, Name, edu_2000, edu_2019_2023, education_change)

cat("Summary of calculated education change:\n")
```

```
## Summary of calculated education change:
```

```
summary(education_change_data$education_change)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -9.700   5.100   7.200   7.593   9.900  26.600
```

```
head(education_change_data)
```

```
## # A tibble: 6 x 5
##   FIPS  Name                  edu_2000 edu_2019_2023 education_change
##   <chr> <chr>                    <dbl>         <dbl>            <dbl>
```

```
## 1 01001 Autauga County, AL     18            28.3           10.3
## 2 01003 Baldwin County, AL     23.1          32.8            9.7
## 3 01005 Barbour County, AL     10.9          11.5            0.600
## 4 01007 Bibb County, AL         7.1          11.5            4.4
## 5 01009 Blount County, AL       9.6          15.6            6
## 6 01011 Bullock County, AL      7.7           9              1.3
```

```
cluster_data_prep1 <- inner_join(income_growth_data, education_change_data, by = "FIPS")

per_capita_income_2001_for_cluster <- per_capita_income %>%
  select(FIPS, per_capita_income_2001 = income_2001)

cluster_data_merged <- cluster_data_prep1 %>%
  inner_join(per_capita_income_2001_for_cluster, by = "FIPS") %>%
  select(FIPS, Name,
         growth_income_county,
         education_change,
         per_capita_income_2001,
         edu_2000) %>%
  na.omit() %>%
  filter(is.finite(growth_income_county) &
         is.finite(education_change) &
         is.finite(per_capita_income_2001) &
         is.finite(edu_2000))

cluster_vars <- cluster_data_merged %>%
  select(growth_income_county, education_change, per_capita_income_2001, edu_2000)

if (nrow(cluster_vars) < 4) {
  cat("Not enough data points for K-means clustering after NA/non-finite removal. Skipping K-means.\n")
} else {
  scaled_cluster_vars <- scale(cluster_vars)

  cat(paste("Number of observations for clustering:", nrow(scaled_cluster_vars), "\n"))

  cat("\nDetermining optimal number of clusters:\n")

  max_k_elbow <- min(10, floor(nrow(scaled_cluster_vars)/2) -1)
  if (max_k_elbow > 1) {
    elbow_plot <- fviz_nbclust(scaled_cluster_vars, kmeans, method = "wss", k.max = max_k_elbow)
    print(elbow_plot)
    cat("Elbow method: Look for an 'elbow' in the plot of total within-cluster sum of squares against k
  } else {
    cat("Not enough data points to use elbow method with k.max > 1.\n")
  }

  max_k_silhouette <- min(10, nrow(scaled_cluster_vars) - 1)
   if (max_k_silhouette > 1) {
    silhouette_plot <- fviz_nbclust(scaled_cluster_vars, kmeans, method = "silhouette", k.max = max_k_s
    print(silhouette_plot)
    cat("Silhouette method: Look for the peak in average silhouette width.\n")
  } else {
    cat("Not enough data points to use silhouette method with k.max > 1.\n")
  }
```

```
optimal_k <- 3
if (max_k_silhouette > 1) {
   nb_clusters_obj <- fviz_nbclust(scaled_cluster_vars, kmeans, method = "silhouette", k.max = max_k_s
   optimal_k_auto <- which.max(nb_clusters_obj$data$y)
   if (length(optimal_k_auto) == 0 || optimal_k_auto < 2) {
      cat("Could not automatically determine optimal k from silhouette, defaulting to 3 if possible, o
      optimal_k <- if(nrow(scaled_cluster_vars) >=3) 3 else if(nrow(scaled_cluster_vars) >=2) 2 else 1
   } else {
     optimal_k <- optimal_k_auto
   }
} else {
  optimal_k <- if(nrow(scaled_cluster_vars) >=2) 2 else 1
}
 if(nrow(scaled_cluster_vars) < optimal_k) {
  optimal_k <- nrow(scaled_cluster_vars)
}
if(optimal_k < 2 && nrow(scaled_cluster_vars) >=2) {
    optimal_k <- 2
} else if (optimal_k < 1) {
    optimal_k <- 1
}

cat(paste("\nPerforming K-means with k=4 (initial) and k=", optimal_k, "(chosen based on silhouette me

set.seed(123)
kmeans_4 <- NULL
if (nrow(scaled_cluster_vars) >= 4) {
  kmeans_4 <- kmeans(scaled_cluster_vars, centers = 4, nstart = 25)
  cluster_data_merged$cluster_k4 <- as.factor(kmeans_4$cluster)
} else {
  cat("Not enough data points for k=4, skipping k=4 clustering.\n")
}

kmeans_optimal <- NULL
if (nrow(scaled_cluster_vars) >= optimal_k && optimal_k > 0) {
  kmeans_optimal <- kmeans(scaled_cluster_vars, centers = optimal_k, nstart = 25)
  cluster_data_merged$cluster_optimal_k <- as.factor(kmeans_optimal$cluster)
} else {
   cat(paste("Not enough data points for k=", optimal_k, " or optimal_k is invalid. Skipping optimal k
}

if (ncol(scaled_cluster_vars) >= 2) {
  if (!is.null(kmeans_4)) {
      cluster_plot_k4 <- fviz_cluster(kmeans_4, data = scaled_cluster_vars,
                                    ellipse.type = "confidence",
                                    geom = "point",
                                    ggtheme = theme_minimal(),
                                    main = "K-means Clustering (k=4)")
      print(cluster_plot_k4)
  }

  if (!is.null(kmeans_optimal) && (is.null(kmeans_4) || optimal_k != 4 || kmeans_4$centers != kmeans_o
     cluster_plot_optimal <- fviz_cluster(kmeans_optimal, data = scaled_cluster_vars,
```

```r
                                   ellipse.type = "confidence",
                                   geom = "point",
                                   ggtheme = theme_minimal(),
                                   main = paste0("K-means Clustering (k=", optimal_k, ")"))
      print(cluster_plot_optimal)
    }

  } else {
    cat("Cannot generate fviz_cluster plot with less than 2 variables for clustering.\n")
  }

  if (!is.null(kmeans_4)) {
    cat("\nSummary of cluster assignments (k=4):\n")
    print(table(cluster_data_merged$cluster_k4))
    cat("\nCentroids (k=4):\n")
    print(kmeans_4$centers)
  }

  if (!is.null(kmeans_optimal)) {
    cat(paste0("\nSummary of cluster assignments (k=", optimal_k, "):\n"))
    print(table(cluster_data_merged$cluster_optimal_k))
    cat(paste0("\nCentroids (k=", optimal_k, "):\n"))
    print(kmeans_optimal$centers)
  }

  cat("\nFirst few rows of data with cluster assignments:\n")
  head(cluster_data_merged)
}
```

```
## Not enough data points for K-means clustering after NA/non-finite removal. Skipping K-means.
```

```r
p1 <- ggplot(df_2001,
             aes(x = log(edu_2000), y = log(inc_2001))) +
  geom_point(alpha = .6) +
  geom_smooth(method = "lm", se = FALSE, linewidth = .8) +
  labs(title = "ln(Income Ratio 2001) vs ln(Education 2000)",
       x = "ln(Education 2000)",
       y = "ln(Income ratio 2001)")

p2 <- ggplot(df_2010,
             aes(x = log(edu_2008_2012), y = log(inc_2010))) +
  geom_point(alpha = .6) +
  geom_smooth(method = "lm", se = FALSE, linewidth = .8) +
  labs(title = "ln(Income Ratio 2010) vs ln(Education 2008-2012)",
       x = "ln(Education 2008-2012)",
       y = "ln(Income ratio 2010)")

p3 <- ggplot(df_2019,
             aes(x = log(edu_2019_2023), y = log(inc_2019))) +
  geom_point(alpha = .6) +
  geom_smooth(method = "lm", se = FALSE, linewidth = .8) +
  labs(title = "ln(Income Ratio 2019) vs ln(Education 2019-2023)",
       x = "ln(Education 2019-2023)",
       y = "ln(Income ratio 2019)")
```

```r
(p1 | p2) / p3
```

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