Draft Documents

Team 1 (Hedgehog, Callista, Imtiyaaz, Issac)

2022-11-16

```
library(needs)

needs("dplyr", "MASS", "readr", "tidyr", "ggplot2",
    "e1071", "moments", "corrplot", "Hmisc", "PerformanceAnalytics",
    "mice", "car", "glmnet", "ggforce", "lmtest")

prioritize(dplyr)
```

1. Introduction (To be done)

2. Data Characteristic

2.1. Nature of Data

The data set is collection The World Bank Data, the variables of interest are extracted from the raw data files and combined into a single data frame for analysis. The final data set includes:

1. **country.code**: Country code
2. **country.name**: Country name
3. **year**: Year
4. **income**: Income class

- Low income (L)
- Lower middle income (LM)
- Upper middle income (UM)
- High income (H)

5. **reg**: Region
6. **pov**: Poverty headcount ratio based on cut-off value of $2.15 per day

7. **mpi**: Multidimensional Poverty Index
8. **edu.total**: Total expenditure on education (% of GDP)
9. **edu.pri**: Total expenditure on primary education (% of total education expenditure)
10. **edu.sec**: Total expenditure on secondary education (% of total education expenditure)
11. **edu.ter**: Total expenditure on tertiary education (% of total education expenditure)
12. **hlth**: Total expenditure on health (% of GDP)
13. **mil**: Total expenditure on military (% of GDP)
14. **fdi**: Foreign Direct Investment
15. **lbr.part**: Labour force participation (% of population ages 15+)
16. **unemp**: Unemployment rate
17. **pop.gwth.total**: Total population growth rate
18. **pop.gwth.rural**: Total rural population growth rate
19. **pop.gwth.urban**: Total urban population growth rate
20. **gdp.dflt**: GDP deflator
21. **gdr.eql**: Gender equality rating
22. **gcf**: Gross Capital Formation
23. **trade**: Trade = import + export (% of GDP)
24. **gdp.pc**: GDP per capita (current US$)

Data imports and combining:

```
# helper functions
importWDI <- function(filepath, value_name) {
    df <- read_csv(filepath, skip = 4)

    colnames(df) <- tolower(gsub(" ", ".", colnames(df)))

    df <- df %>%
        pivot_longer(5:ncol(.), names_to = "year",
            values_to = "value") %>%
        filter(!is.null(value) & !is.na(value)) %>%
        mutate(country.code = factor(country.code),
            country.name = factor(country.name), year = as.numeric(year)) %>%
        select(country.code, country.name, year, value)

    colnames(df)[4] <- value_name

    df
```

```
}

importRegionClass <- function(filepath) {
    df <- read_csv(filepath, skip = 4)

    colnames(df) <- tolower(gsub(" ", ".", colnames(df)))

    df %>%
        mutate(country.name = factor(country.name),
            region = factor(region)) %>%
        select(country.name, reg = region)
}

importIncomeClass <- function(filepath) {
    df <- read_csv(filepath, skip = 4)

    colnames(df) <- tolower(gsub(" ", ".", colnames(df)))

    df %>%
        pivot_longer(3:ncol(.), names_to = "year",
            values_to = "income") %>%
        filter(!is.null(income) & !is.na(income)) %>%
        mutate(country.code = factor(country.code),
            country.name = factor(country.name), year = as.numeric(year),
            income = factor(income)) %>%
        select(country.code, country.name, year, income)
}
```

```
# import data
setwd("../data")

poverty.headcount <- importWDI("poverty.headcount.215dollar.csv",
    "pov")
mpi <- importWDI("mpi.csv", "mpi")
education.expenditure.total <- importWDI("total.education.expenditure.csv",
    "edu.total")
education.expenditure.primary <- importWDI("primary.education.expenditure.csv",
    "edu.pri")
education.expenditure.secondary <- importWDI("secondary.education.expenditure.csv",
    "edu.sec")
education.expenditure.tertiary <- importWDI("tertiary.education.expenditure.csv",
    "edu.ter")
health.expenditure <- importWDI("health.expenditure.csv",
    "hlth")
military.expenditure <- importWDI("military.expenditure.csv",
    "mil")
fdi <- importWDI("fdi.csv", "fdi")
labour.force.participation <- importWDI("labour.force.participation.csv",
    "lbr.part")
unemployment.rate <- importWDI("unemployment.csv",
    "unemp")
population.growth <- importWDI("population.growth.csv",
    "pop.gwth.total")
```

```
rural.population.growth <- importWDI("rural.population.growth.csv",
    "pop.gwth.rural")
urban.population.growth <- importWDI("urban.population.growth.csv",
    "pop.gwth.urban")
gdp.deflator <- importWDI("gdp.deflator.csv", "gdp.dflt")
gender.equality <- importWDI("gender.equality.csv",
    "gdr.eql")
gross.capital.formation <- importWDI("gross.capital.formation.csv",
    "gcf")
trade <- importWDI("trade.csv", "trade")
region.class <- importRegionClass("region.class.csv")
income.class <- importIncomeClass("income.class.csv")
gdp.pc <- importWDI("gdp.pc.csv", "gdp.pc")

setwd("../src")
```

We found that the data sets collected from World Bank's Data helpdesk and The World Bank's Data have different naming convention for certain countries (e.g. "Czechia" vs. "Czechnia Republic"). So we need to rename these countries to avoid some error when joining.

Furthermore, WDI's data sets rate also account for non-country (e.g. country.name = "Low income" or "South Asia"). These special groups are not in our scope of interest, which is national, so we eliminate them.

```
# using poverty.headcount as a naming standard
# (as other data from WDI also use this
# convention) join a subset of data to process
# the names
d <- poverty.headcount %>%
    select(country.name) %>%
    mutate(inPov = T) %>%
    full_join(income.class %>%
        select(country.name) %>%
        mutate(inIncome = T), by = "country.name") %>%
    full_join(region.class %>%
        select(country.name) %>%
        mutate(inReg = T), by = "country.name") %>%
    mutate(inPov = !is.na(inPov), inIncome = !is.na(inIncome),
        inReg = !is.na(inReg))

d
```

```
## # A tibble: 62,759 x 4
##    country.name inPov inIncome inReg
##    <fct>        <lgl> <lgl>    <lgl>
##  1 Angola       TRUE  TRUE     TRUE
##  2 Angola       TRUE  TRUE     TRUE
##  3 Angola       TRUE  TRUE     TRUE
##  4 Angola       TRUE  TRUE     TRUE
##  5 Angola       TRUE  TRUE     TRUE
##  6 Angola       TRUE  TRUE     TRUE
##  7 Angola       TRUE  TRUE     TRUE
##  8 Angola       TRUE  TRUE     TRUE
##  9 Angola       TRUE  TRUE     TRUE
## 10 Angola       TRUE  TRUE     TRUE
```

```
## # ... with 62,749 more rows
## # i Use 'print(n = ...)' to see more rows
```

First, remove special economic groups from `poverty.headcount`. We figured these regions will not appear in `income.class` or `region.class`, so we might find something from looking at the countries **only** appear in `poverty.headcount`.

```
d %>%
    filter(inPov & (!inIncome | !inReg)) %>%
    distinct(country.name)
```

```
## # A tibble: 18 x 1
##    country.name
##    <fct>
##  1 Cote d'Ivoire
##  2 Czechia
##  3 East Asia & Pacific
##  4 Europe & Central Asia
##  5 Fragile and conflict affected situations
##  6 High income
##  7 IDA total
##  8 Latin America & Caribbean
##  9 Low income
## 10 Lower middle income
## 11 Low & middle income
## 12 Middle East & North Africa
## 13 South Asia
## 14 Sub-Saharan Africa
## 15 Sao Tome and Principe
## 16 Turkiye
## 17 Upper middle income
## 18 World
```

Lucky! We can look through these 18 results and compose a list of special regions.

```
spec.reg <- c("Fragile and conflict affected situations",
    "IDA total", "World", "East Asia & Pacific", "Europe & Central Asia",
    "Latin America & Caribbean", "Middle East & North Africa",
    "South Asia", "Sub-Saharan Africa", "Low income",
    "Low & middle income", "Lower middle income", "Upper middle income",
    "High income")
```

Then, we rename those countries with inconsistent naming convention. Since we should only care about countries whose poverty headcount is available, reusing the list generated above, we can identify:

1. Cote d'Ivoire (also Côte d'Ivoire)
2. Czechia (also Czechoslovakia or Czech Republic)
3. Curacao (also Curaçao)
4. Turkiye (formerly known as Turkey, also Türkiye)
5. Sao Tome and Principe (also São Tomé and Príncipe)

```
# mapping standard name and variation
nameMap <- tibble(standard = c("Cote d'Ivoire", "Czechia",
    "Czechia", "Curacao", "Turkiye", "Turkiye", "Sao Tome and Principe"),
    variation = c("Côte d'Ivoire", "Czechoslovakia",
        "Czech Republic", "Curaçao", "Turkey", "Türkiye",
        "São Tomé and Príncipe"))

correctName <- function(name) {
    tibble(name = name) %>%
        left_join(nameMap, by = c(name = "variation")) %>%
        mutate(standard = ifelse(is.na(standard), name,
            standard)) %>%
        select(standard) %>%
        pull()
}

orig.name <- c("Vietnam", "China", "Turkey", "Czechia Republic")
correctName(orig.name)
```

```
## [1] "Vietnam"          "China"            "Turkiye"          "Czechia Republic"
```

Let's test this out!

```
d <- poverty.headcount %>%
  # correct name here
  mutate(country.name = correctName(country.name)) %>%
  select(country.name) %>%
  mutate(inPov = T) %>%
  full_join(income.class %>%
              # correct name here
              mutate(country.name = correctName(country.name)) %>%
              select(country.name) %>%
              mutate(inIncome = T), by = "country.name") %>%
  full_join(region.class %>%
              # correct name here
              mutate(country.name = correctName(country.name)) %>%
              select(country.name) %>%
              mutate(inReg = T), by = "country.name") %>%
  filter(!(country.name %in% spec.reg)) %>%
  mutate(inPov = !is.na(inPov), inIncome = !is.na(inIncome), inReg = !is.na(inReg))

# countries not in region list, but is in Pov list
d %>%
  filter(!inReg & inPov) %>%
  distinct(country.name) %>%
  nrow()
```

```
## [1] 0
```

```
# countries not in income list, but is in Pov list
d %>%
  filter(!inIncome & inPov) %>%
```

```
  distinct(country.name) %>%
  nrow()
```

```
## [1] 0
```

We are *pretty* confident that there's no inconsistent naming left unprocessed in the data sets.

```
# Rename countries in all data sets.
poverty.headcount <- poverty.headcount %>%
    mutate(country.name = correctName(country.name))
mpi <- mpi %>%
    mutate(country.name = correctName(country.name))
education.expenditure.total <- education.expenditure.total %>%
    mutate(country.name = correctName(country.name))
education.expenditure.primary <- education.expenditure.primary %>%
    mutate(country.name = correctName(country.name))
education.expenditure.secondary <- education.expenditure.secondary %>%
    mutate(country.name = correctName(country.name))
education.expenditure.tertiary <- education.expenditure.tertiary %>%
    mutate(country.name = correctName(country.name))
health.expenditure <- health.expenditure %>%
    mutate(country.name = correctName(country.name))
military.expenditure <- military.expenditure %>%
    mutate(country.name = correctName(country.name))
fdi <- fdi %>%
    mutate(country.name = correctName(country.name))
labour.force.participation <- labour.force.participation %>%
    mutate(country.name = correctName(country.name))
unemployment.rate <- unemployment.rate %>%
    mutate(country.name = correctName(country.name))
population.growth <- population.growth %>%
    mutate(country.name = correctName(country.name))
rural.population.growth <- rural.population.growth %>%
    mutate(country.name = correctName(country.name))
urban.population.growth <- urban.population.growth %>%
    mutate(country.name = correctName(country.name))
gdp.deflator <- gdp.deflator %>%
    mutate(country.name = correctName(country.name))
gender.equality <- gender.equality %>%
    mutate(country.name = correctName(country.name))
gross.capital.formation <- gross.capital.formation %>%
    mutate(country.name = correctName(country.name))
trade <- trade %>%
    mutate(country.name = correctName(country.name))
region.class <- region.class %>%
    mutate(country.name = correctName(country.name))
income.class <- income.class %>%
    mutate(country.name = correctName(country.name))
gdp.pc <- gdp.pc %>%
    mutate(country.name = correctName(country.name))
```

Join the data

```
countries <- poverty.headcount %>%
    # We used a full join here so we can conduct
    # a separate analysis on mpi later
full_join(mpi, by = c("country.name", "country.code",
    "year")) %>%
    left_join(income.class, c("country.name", "country.code",
        "year")) %>%
    left_join(region.class, by = "country.name") %>%
    left_join(education.expenditure.total, by = c("country.name",
        "country.code", "year")) %>%
    left_join(education.expenditure.primary, by = c("country.name",
        "country.code", "year")) %>%
    left_join(education.expenditure.secondary, by = c("country.name",
        "country.code", "year")) %>%
    left_join(education.expenditure.tertiary, by = c("country.name",
        "country.code", "year")) %>%
    left_join(health.expenditure, by = c("country.name",
        "country.code", "year")) %>%
    left_join(military.expenditure, by = c("country.name",
        "country.code", "year")) %>%
    left_join(fdi, by = c("country.name", "country.code",
        "year")) %>%
    left_join(labour.force.participation, by = c("country.name",
        "country.code", "year")) %>%
    left_join(unemployment.rate, by = c("country.name",
        "country.code", "year")) %>%
    left_join(population.growth, by = c("country.name",
        "country.code", "year")) %>%
    left_join(rural.population.growth, by = c("country.name",
        "country.code", "year")) %>%
    left_join(urban.population.growth, by = c("country.name",
        "country.code", "year")) %>%
    left_join(gdp.deflator, by = c("country.name",
        "country.code", "year")) %>%
    left_join(gender.equality, by = c("country.name",
        "country.code", "year")) %>%
    left_join(gross.capital.formation, by = c("country.name",
        "country.code", "year")) %>%
    left_join(trade, by = c("country.name", "country.code",
        "year")) %>%
    left_join(gdp.pc, by = c("country.name", "country.code",
        "year")) %>%
    # filter special groups
filter(!(country.name %in% spec.reg))
```

Data preview

```
head(countries)
```

```
## # A tibble: 6 x 24
##   count~1 count~2  year   pov   mpi income reg   edu.t~3 edu.pri edu.sec edu.ter
##   <fct>   <chr>   <dbl> <dbl> <dbl> <fct>  <fct>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 AGO     Angola   2000  21.4    NA L      Sub-~    2.61      NA      NA      NA
```

```
## 2 AGO     Angola   2008  14.6    NA LM     Sub-~    2.69      NA      NA      NA
## 3 AGO     Angola   2018  31.1    NA LM     Sub-~    2.04      NA      NA      NA
## 4 ALB     Albania  1996   0.5    NA LM     Euro~    3.08      NA      NA      NA
## 5 ALB     Albania  2002   1.1    NA LM     Euro~    3.12      NA      NA      NA
## 6 ALB     Albania  2005   0.6    NA LM     Euro~    3.28      NA      NA      NA
## # ... with 13 more variables: hlth <dbl>, mil <dbl>, fdi <dbl>, lbr.part <dbl>,
## #   unemp <dbl>, pop.gwth.total <dbl>, pop.gwth.rural <dbl>,
## #   pop.gwth.urban <dbl>, gdp.dflt <dbl>, gdr.eql <dbl>, gcf <dbl>,
## #   trade <dbl>, gdp.pc <dbl>, and abbreviated variable names 1: country.code,
## #   2: country.name, 3: edu.total
## # i Use `colnames()` to see all variable names
```

```
str(countries)
```

```
## tibble [1,901 x 24] (S3: tbl_df/tbl/data.frame)
##  $ country.code  : Factor w/ 272 levels "AGO","ALB","ARE",..: 1 1 1 2 2 2 2 2 2 2 ...
##  $ country.name  : chr [1:1901] "Angola" "Angola" "Angola" "Albania" ...
##  $ year          : num [1:1901] 2000 2008 2018 1996 2002 ...
##  $ pov           : num [1:1901] 21.4 14.6 31.1 0.5 1.1 0.6 0.2 0.6 1 0.1 ...
##  $ mpi           : num [1:1901] NA NA NA NA NA NA NA NA NA NA ...
##  $ income        : Factor w/ 4 levels "H","L","LM","UM": 2 3 3 3 3 3 3 4 4 4 ...
##  $ reg           : Factor w/ 7 levels "East Asia & Pacific",..: 7 7 7 2 2 2 2 2 2 2 ...
##  $ edu.total     : num [1:1901] 2.61 2.69 2.04 3.08 3.12 ...
##  $ edu.pri       : num [1:1901] NA NA NA NA NA ...
##  $ edu.sec       : num [1:1901] NA NA NA NA NA ...
##  $ edu.ter       : num [1:1901] NA NA NA NA NA ...
##  $ hlth          : num [1:1901] 1.91 3.32 2.54 NA 6.91 ...
##  $ mil           : num [1:1901] 6.39 3.57 1.87 1.38 1.32 ...
##  $ fdi           : num [1:1901] 8.79e+08 1.68e+09 -6.46e+09 9.01e+07 1.35e+08 ...
##  $ lbr.part      : num [1:1901] NA NA NA 38.8 59.6 ...
##  $ unemp         : num [1:1901] NA NA NA 12.3 15.8 ...
##  $ pop.gwth.total: num [1:1901] 3.277 3.711 3.276 -0.622 -0.3 ...
##  $ pop.gwth.rural: num [1:1901] 0.921 1.91 1.338 -1.546 -2.169 ...
##  $ pop.gwth.urban: num [1:1901] 5.682 5.02 4.312 0.812 2.181 ...
##  $ gdp.dflt      : num [1:1901] 418.02 19.37 28.17 38.17 3.65 ...
##  $ gdr.eql       : num [1:1901] NA 3 NA NA NA 4 NA NA NA NA ...
##  $ gcf           : num [1:1901] 30.5 30.8 17.9 18.1 35.3 ...
##  $ trade         : num [1:1901] 152.5 121.4 66.4 44.9 68.5 ...
##  $ gdp.pc        : num [1:1901] 557 4081 2525 1010 1425 ...
```

```
summary(countries)
```

```
##   country.code  country.name            year           pov
##  BRA    :  36   Length:1901        Min.   :1967   Min.   : 0.0
##  CRI    :  34   Class :character   1st Qu.:2002   1st Qu.: 0.2
##  ARG    :  32   Mode  :character   Median :2009   Median : 1.5
##  USA    :  32                      Mean   :2007   Mean   :10.0
##  DEU    :  30                      3rd Qu.:2014   3rd Qu.:11.6
##  HND    :  30                      Max.   :2021   Max.   :91.5
##  (Other):1707                                     NA's   :58
##       mpi         income                              reg        edu.total
##  Min.   : 2.37   H   :644   East Asia & Pacific       :167   Min.   : 1.03
##  1st Qu.:18.30   L   :253   Europe & Central Asia     :883   1st Qu.: 3.52
```

```
##  Median :24.80   LM  :501   Latin America & Caribbean :416   Median : 4.52
##  Mean   :27.06   UM  :438   Middle East & North Africa:107   Mean   : 4.58
##  3rd Qu.:33.30   NA's: 65   North America             : 50   3rd Qu.: 5.46
##  Max.   :74.20              South Asia                : 61   Max.   :15.75
##  NA's   :1446               Sub-Saharan Africa        :217   NA's   :596
##     edu.pri         edu.sec         edu.ter          hlth            mil
##  Min.   : 0.66   Min.   : 2.72   Min.   : 0.0   Min.   : 1.72   Min.   : 0.00
##  1st Qu.:24.03   1st Qu.:30.14   1st Qu.:16.6   1st Qu.: 5.15   1st Qu.: 1.04
##  Median :30.47   Median :35.71   Median :20.6   Median : 6.91   Median : 1.47
##  Mean   :31.66   Mean   :35.63   Mean   :21.0   Mean   : 6.98   Mean   : 1.79
##  3rd Qu.:38.33   3rd Qu.:41.38   3rd Qu.:25.1   3rd Qu.: 8.56   3rd Qu.: 2.10
##  Max.   :70.10   Max.   :71.59   Max.   :50.4   Max.   :17.73   Max.   :19.39
##  NA's   :1090    NA's   :1094    NA's   :963    NA's   :464     NA's   :105
##       fdi               lbr.part        unemp        pop.gwth.total
##  Min.   :-3.44e+11   Min.   :30.5   Min.   : 0.25   Min.   :-3.630
##  1st Qu.: 2.98e+08   1st Qu.:56.2   1st Qu.: 4.51   1st Qu.: 0.266
##  Median : 1.71e+09   Median :61.2   Median : 6.88   Median : 1.032
##  Mean   : 1.60e+10   Mean   :60.8   Mean   : 8.14   Mean   : 1.056
##  3rd Qu.: 9.82e+09   3rd Qu.:65.5   3rd Qu.:10.08   3rd Qu.: 1.776
##  Max.   : 7.34e+11   Max.   :93.0   Max.   :49.70   Max.   : 5.614
##  NA's   :17          NA's   :320    NA's   :267
##  pop.gwth.rural    pop.gwth.urban     gdp.dflt         gdr.eql
##  Min.   :-8.5607   Min.   :-4.08   Min.   : -26.3   Min.   :1.50
##  1st Qu.:-0.8566   1st Qu.: 0.51   1st Qu.:   1.7   1st Qu.:3.00
##  Median :-0.0246   Median : 1.48   Median :   3.9   Median :3.50
##  Mean   : 0.0008   Mean   : 1.69   Mean   :  15.8   Mean   :3.59
##  3rd Qu.: 0.9636   3rd Qu.: 2.66   3rd Qu.:   8.5   3rd Qu.:4.00
##  Max.   : 4.5969   Max.   :13.80   Max.   :3333.6   Max.   :5.00
##  NA's   :14        NA's   :13      NA's   :18       NA's   :1635
##       gcf             trade            gdp.pc
##  Min.   : 0.0   Min.   :  1.38   Min.   :   120
##  1st Qu.:19.6   1st Qu.: 51.06   1st Qu.:  1928
##  Median :22.7   Median : 73.50   Median :  6032
##  Mean   :23.9   Mean   : 84.43   Mean   : 15220
##  3rd Qu.:26.8   3rd Qu.:105.46   3rd Qu.: 21490
##  Max.   :69.5   Max.   :380.10   Max.   :123679
##  NA's   :72     NA's   :57       NA's   :8
```

There's no NA in `reg`, which is a sign that all naming in the data is remedied. There's some expected NAs in `income` and `pov`, as these data are collected by year. There's a substantial amount of missing data in `mpi`, as this is a relative new concept. We will address the nature, and processing of missing data in the next sections.

2.2. Missing values

As observed from the summary above, the data set contains a lot of missing values in some of the variables.

```
mean(is.na(countries))
```

```
## [1] 0.18197
```

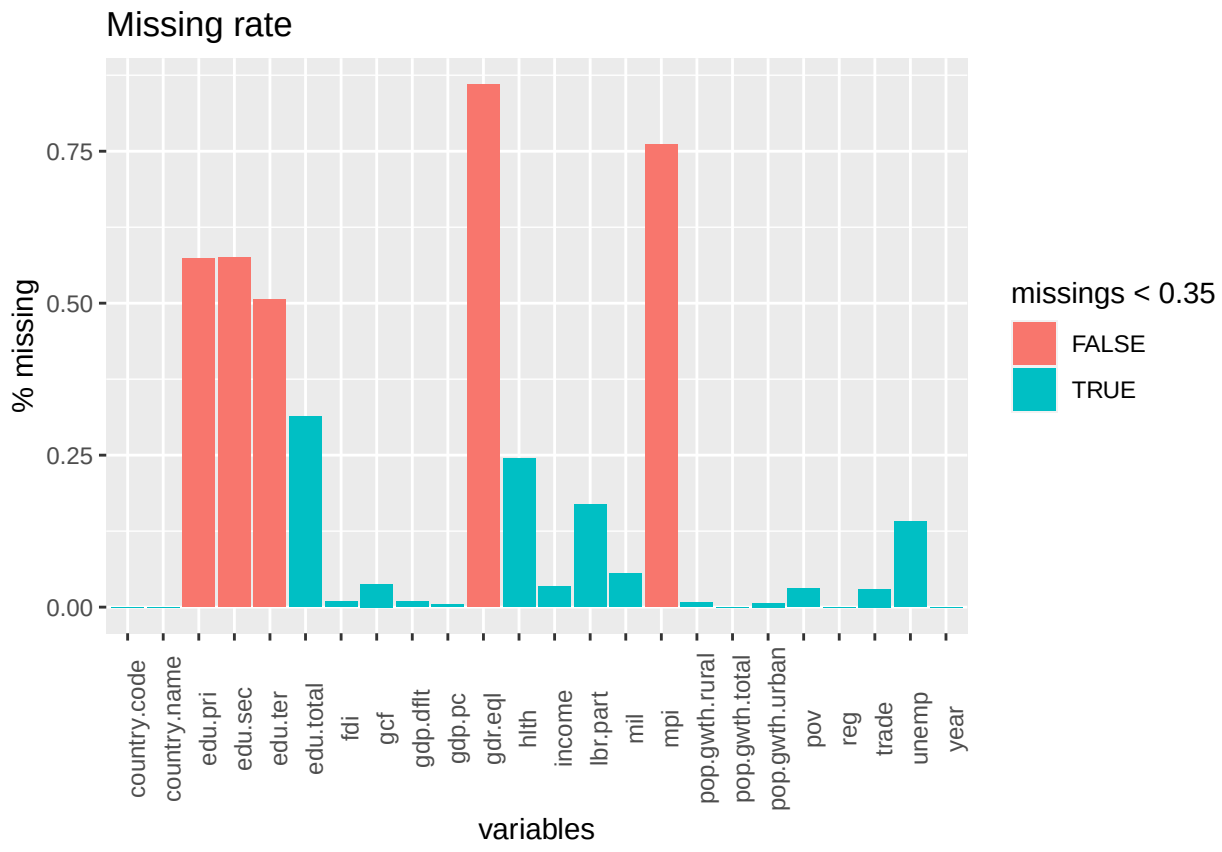
About 19% of the data set is missing.

```
nCompleteObs <- sum(complete.cases(countries))
print(paste("No. of complete cases:", nCompleteObs))
```

```
## [1] "No. of complete cases: 3"
```

There are only 3 complete cases where all the variable is available. This is nowhere near acceptable to conduct any meaningful analysis. Therefore, we need to eliminate some variables for a more balance data set.

```
missings <- colMeans(is.na(countries))
ggplot(mapping = aes(x = names(missings), y = missings,
    fill = missings < 0.35)) + geom_bar(stat = "identity") +
    ggtitle("Missing rate") + xlab("variables") + ylab("% missing") +
    theme(axis.text.x = element_text(size = 9, angle = 90))
```




```
missings[missings > 0.35]
```

```
##     mpi edu.pri edu.sec edu.ter gdr.eql
## 0.76065 0.57338 0.57549 0.50658 0.86007
```

There are 5 variables with missing rate >35%.

expenditure in primary, secondary, and tertiary edication can be very useful and relevant information to predict poverty reduction (Akbar et al. 2019). However, we would like to exclude these variables from some

first analyses to make use of the richer set of data. We can conduct a separate analysis with these variable to gain more insight.

```
# variables with high missing rate
hMiss <- names(missings[missings > 0.35])
# exclude these variables in countries1
countries1 <- countries %>%
    select(!hMiss) %>%
    filter(!is.na(pov))
```

```
## Note: Using an external vector in selections is ambiguous.
## i Use ‘all_of(hMiss)‘ instead of ‘hMiss‘ to silence this message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

```
str(countries1)
```

```
## tibble [1,843 x 19] (S3: tbl_df/tbl/data.frame)
##  $ country.code  : Factor w/ 272 levels "AGO","ALB","ARE",..: 1 1 1 2 2 2 2 2 2 2 ...
##  $ country.name  : chr [1:1843] "Angola" "Angola" "Angola" "Albania" ...
##  $ year          : num [1:1843] 2000 2008 2018 1996 2002 ...
##  $ pov           : num [1:1843] 21.4 14.6 31.1 0.5 1.1 0.6 0.2 0.6 1 0.1 ...
##  $ income        : Factor w/ 4 levels "H","L","LM","UM": 2 3 3 3 3 3 3 4 4 4 ...
##  $ reg           : Factor w/ 7 levels "East Asia & Pacific",..: 7 7 7 2 2 2 2 2 2 2 ...
##  $ edu.total     : num [1:1843] 2.61 2.69 2.04 3.08 3.12 ...
##  $ hlth          : num [1:1843] 1.91 3.32 2.54 NA 6.91 ...
##  $ mil           : num [1:1843] 6.39 3.57 1.87 1.38 1.32 ...
##  $ fdi           : num [1:1843] 8.79e+08 1.68e+09 -6.46e+09 9.01e+07 1.35e+08 ...
##  $ lbr.part      : num [1:1843] NA NA NA 38.8 59.6 ...
##  $ unemp         : num [1:1843] NA NA NA 12.3 15.8 ...
##  $ pop.gwth.total: num [1:1843] 3.277 3.711 3.276 -0.622 -0.3 ...
##  $ pop.gwth.rural: num [1:1843] 0.921 1.91 1.338 -1.546 -2.169 ...
##  $ pop.gwth.urban: num [1:1843] 5.682 5.02 4.312 0.812 2.181 ...
##  $ gdp.dflt      : num [1:1843] 418.02 19.37 28.17 38.17 3.65 ...
##  $ gcf           : num [1:1843] 30.5 30.8 17.9 18.1 35.3 ...
##  $ trade         : num [1:1843] 152.5 121.4 66.4 44.9 68.5 ...
##  $ gdp.pc        : num [1:1843] 557 4081 2525 1010 1425 ...
```

Re-evaluate the `countries1` set.

```
mean(is.na(countries1))
```

```
## [1] 0.054716
```

```
sum(complete.cases(countries1))
```

```
## [1] 937
```

```
mean(complete.cases(countries1))
```

```
## [1] 0.50841
```

On average, each column has 6% missing rate, results in 937 complete data point (i.e. 49%). This can be a sufficient number for the analysis. However, the missing data can induce loss of power due to the reduced sample size, and some other biases depending on which variables is missing.

```
# complete rate of data by regions
countries1 %>%
    mutate(isComplete = complete.cases(.)) %>%
    group_by(reg) %>%
    summarise(complete.rate = mean(isComplete)) %>%
    arrange(desc(complete.rate))
```

```
## # A tibble: 7 x 2
##   reg                          complete.rate
##   <fct>                                <dbl>
## 1 Europe & Central Asia                0.646
## 2 Latin America & Caribbean            0.505
## 3 Middle East & North Africa           0.462
## 4 East Asia & Pacific                  0.437
## 5 South Asia                           0.245
## 6 Sub-Saharan Africa                   0.192
## 7 North America                        0.14
```

Countries from North America, Sub-Saharan Africa, and South Asia have the highest rate of missing data. We suspect that Sub-Saharan Africa, and South Asia are comparably less accessible regions. We also know that Americans don't like filling out forms, so their high rate of missing data is understandable as well.

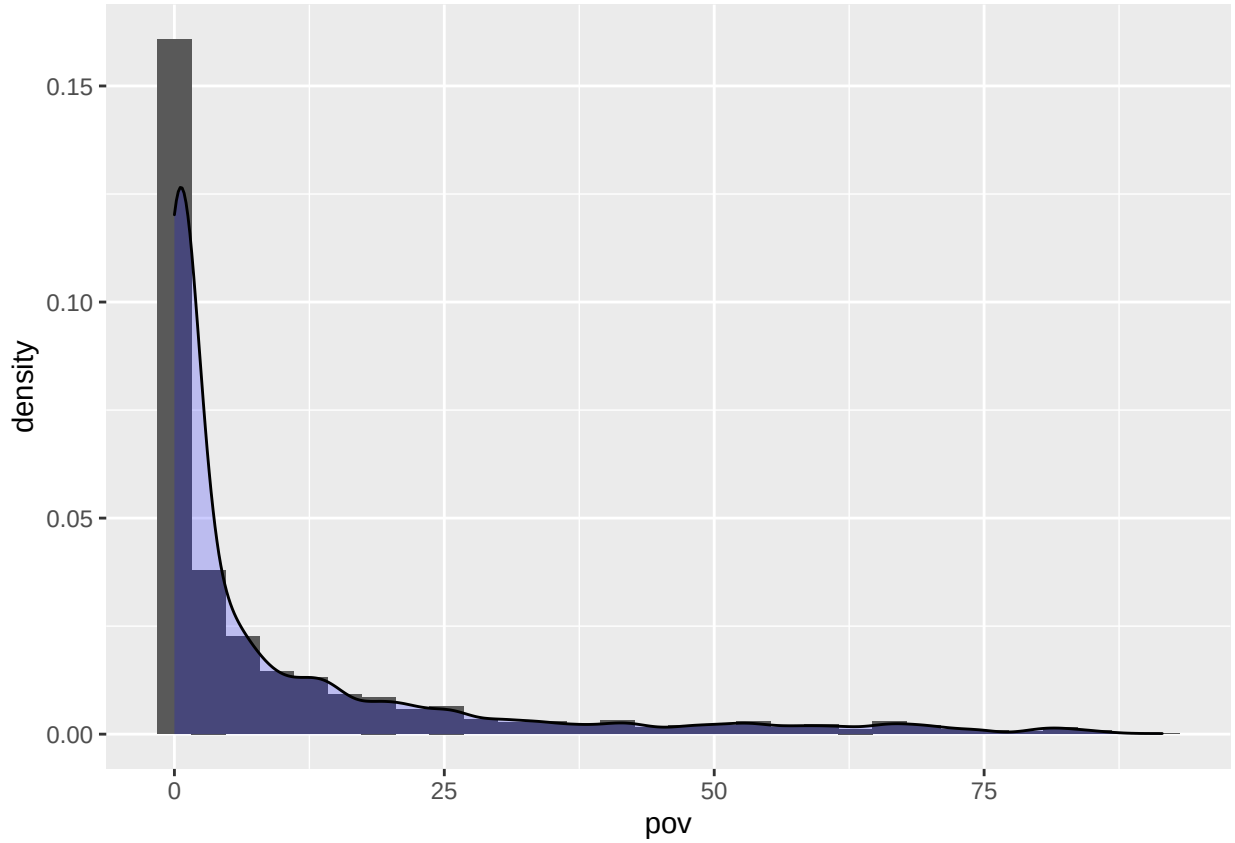
Still, we need to find a way to address this issue. we propose several approaches:

1. **Use complete cases**: Only use the complete cases for the analysis. This is a straightforward approach, but doesn't resolve the bias resulted from the mass loss of data.
2. **Selectively remove variables with high missing rate**: The same as we did before, but this process should be carried out carefully as we run the chance of dropping an important variable.
3. **Update the data set as we select variables**: As we drop insignificant variables (in backward selection), the number of NAs are changed as well. We can utilize the extra complete cases to build the next model in the steps.
4. **Imputation**: The idea is to replace the missing observations on the response or the predictors with artificial values that try to preserve the data set structure. This is a quite complex topic of its own, but we think why not. You can read more from Arel-Bundock and Pelc (2018).

2.3. Descriptive Analytics

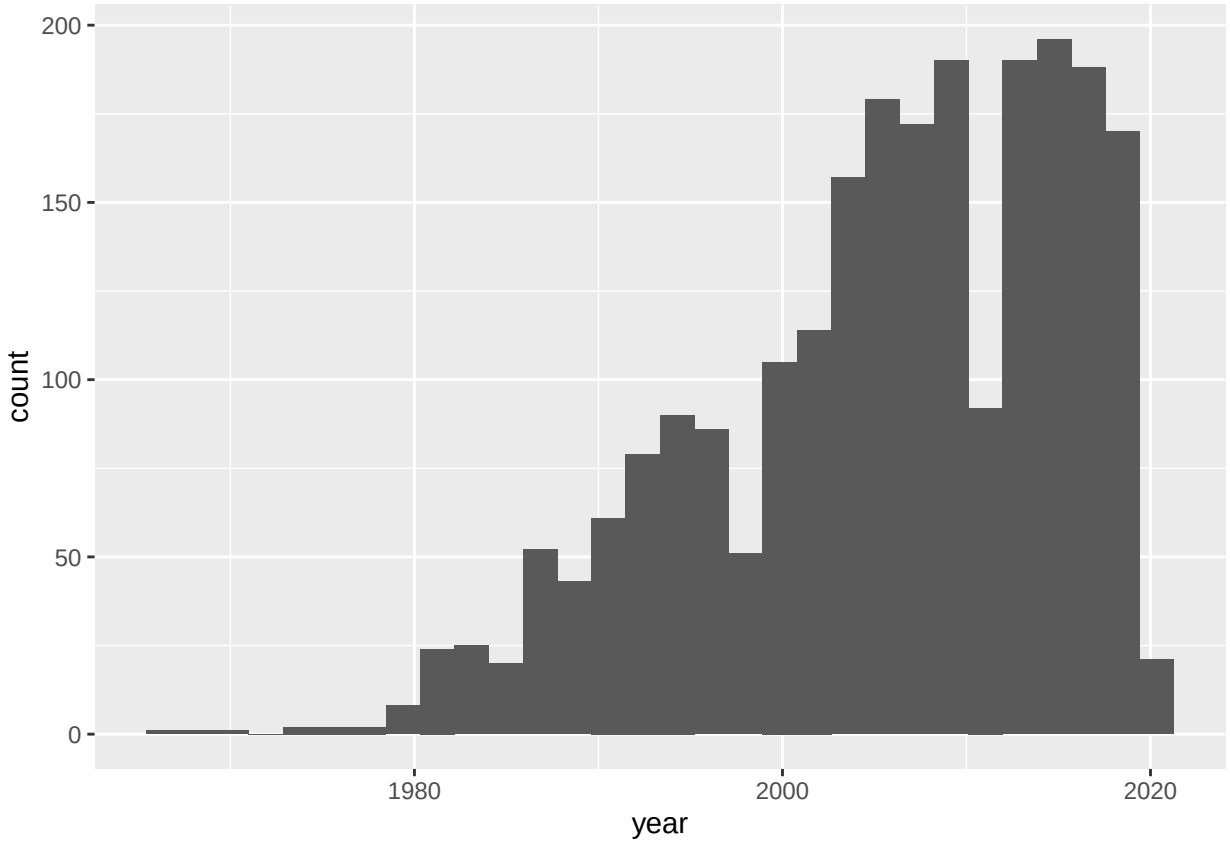
Distribution of the predicted variable `pov`

```
ggplot(countries, aes(x = pov)) + geom_histogram(aes(y = ..density..),
    bins = 30) + geom_density(alpha = 0.2, fill = "blue")
```



The graph displays a decreasing rate as poverty indicator increasing. This might not be representative of the current state of poverty in the world, but of the number presented in our data. For example, more recent data is likely to be more inclusive than ancient data, when poverty is more prevalent. We should look at data from the same period.

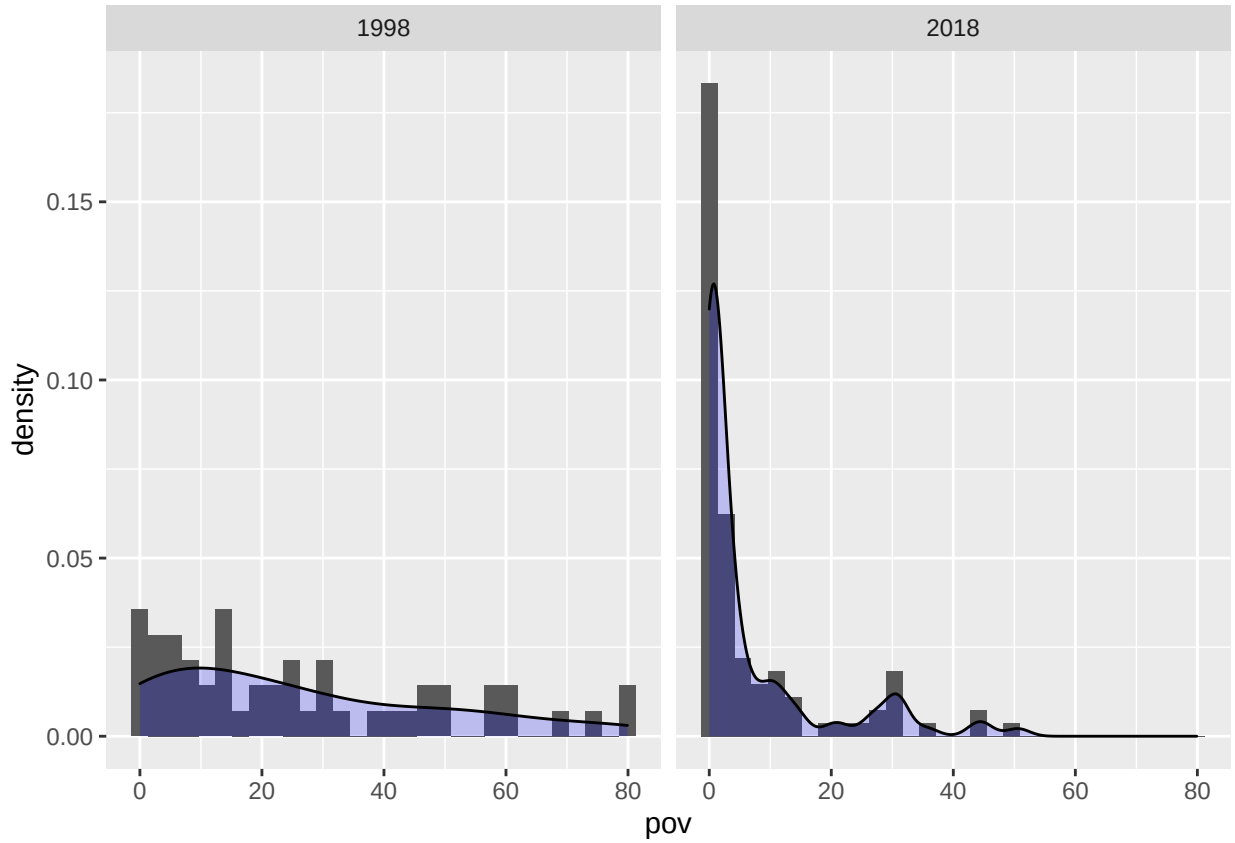
```
# number of data point available from 1967 to
# 2021
ggplot(poverty.headcount, aes(x = year)) + geom_histogram(bins = 30)
```




```
# pov data from 1998 and 2018
pov.98.18 <- poverty.headcount %>%
    filter(year == 1998 | year == 2018)
pov.98.18 %>%
    group_by(year) %>%
    summarise(sum = n())
```

```
## # A tibble: 2 x 2
##    year   sum
##   <dbl> <int>
## 1  1998    51
## 2  2018    99
```

```
ggplot(pov.98.18, aes(x = pov)) + geom_histogram(aes(y = ..density..),
    bins = 30) + geom_density(alpha = 0.2, fill = "blue") +
    facet_grid(cols = vars(year))
```



The graph for 1998 has a much gentler slope, meaning poverty was more popular during that time, as predicted from out intuition. What about the general progress of the world?

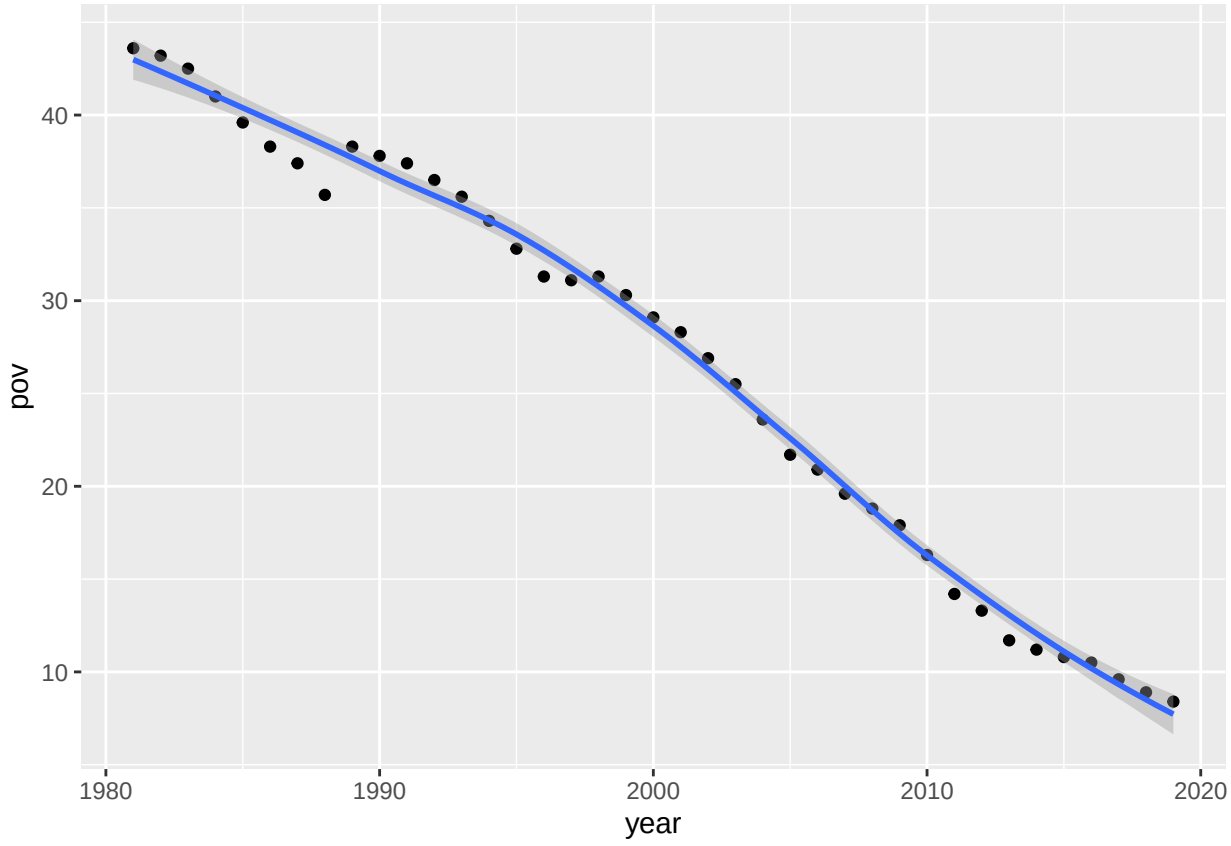
```
# Re-import pov and only take special regions
# geographic
geo.regs <- c("EAS", "ECS", "LCN", "MEA", "SAS", "SSF",
    "WLD")
# economics
eco.regs <- c("HIC", "LIC", "LMC", "LMY", "UMC")

pov.reg <- importWDI("../data/poverty.headcount.215dollar.csv",
    "pov") %>%
    filter(country.code %in% c(geo.regs, eco.regs)) %>%
    mutate(type = ifelse(country.code %in% geo.regs,
        "Geographic", "Economics"))
```

```
## New names:
## Rows: 266 Columns: 67
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (4): Country Name, Country Code, Indicator Name, Indicator Code dbl (50): 1967,
## 1969, 1971, 1974, 1975, 1977, 1978, 1979, 1980, 1981, 1982, ... lgl (13): 1960,
## 1961, 1962, 1963, 1964, 1965, 1966, 1968, 1970, 1972, 1973, ...
## 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.
## * `` -> `...67`
```

```
pov.reg %>%
    distinct(country.code, country.name, type) %>%
    arrange(type)
```

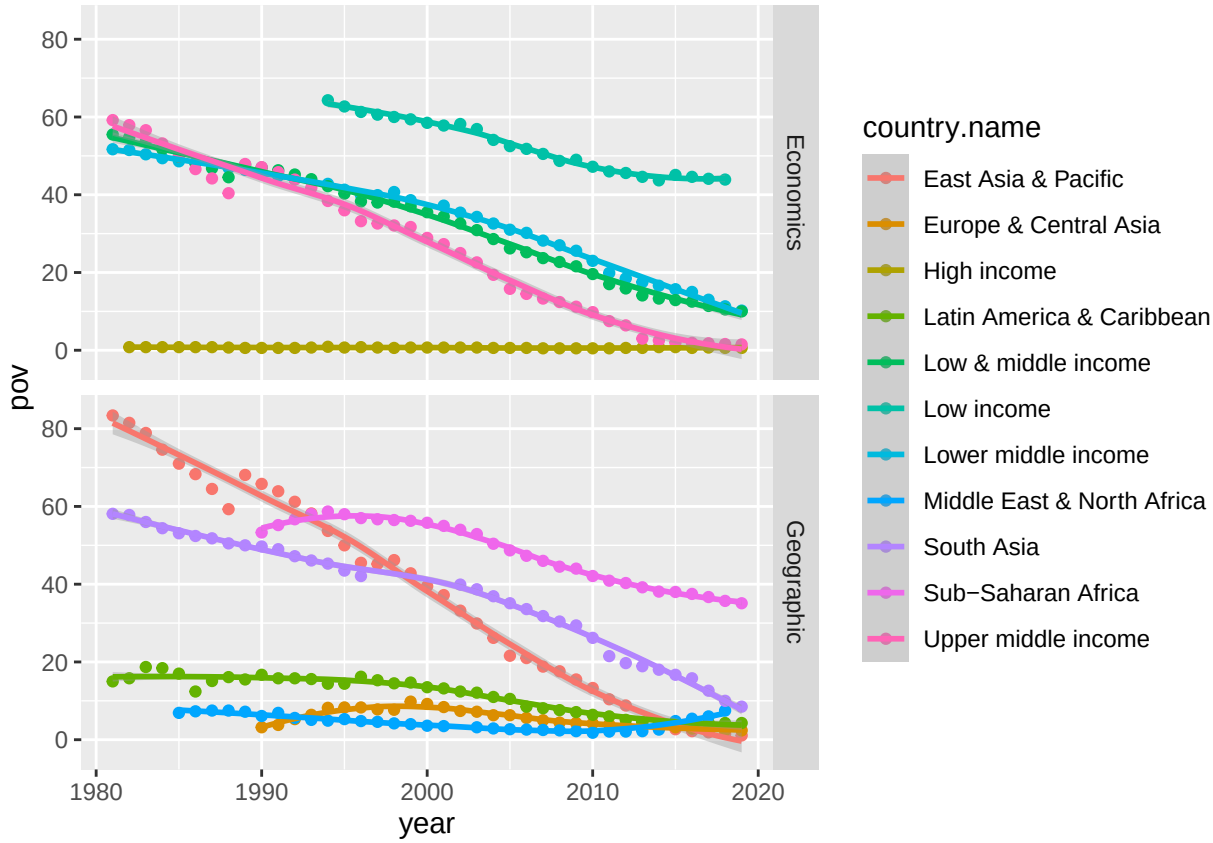
```
## # A tibble: 12 x 3
##    country.code country.name               type
##    <fct>        <fct>                      <chr>
##  1 HIC          High income                Economics
##  2 LIC          Low income                 Economics
##  3 LMC          Lower middle income        Economics
##  4 LMY          Low & middle income        Economics
##  5 UMC          Upper middle income        Economics
##  6 EAS          East Asia & Pacific        Geographic
##  7 ECS          Europe & Central Asia      Geographic
##  8 LCN          Latin America & Caribbean  Geographic
##  9 MEA          Middle East & North Africa Geographic
## 10 SAS          South Asia                 Geographic
## 11 SSF          Sub-Saharan Africa         Geographic
## 12 WLD          World                      Geographic
```

```
# World
ggplot(pov.reg %>%
    filter(country.code == "WLD"), aes(x = year, y = pov)) +
    geom_point() + geom_smooth(formula = y ~ x, method = loess)
```



An overall very steady decrease of poverty. How about each region?

```
ggplot(pov.reg %>%
    filter(country.code != "WLD"), aes(x = year, y = pov,
    color = country.name)) + geom_point() + geom_smooth(formula = y ~
    x, method = loess) + facet_grid(rows = vars(type))
```



There's a general steady, but distinct decline of poverty over time in each type region of respective type. Latin America & Caribbean, Europe & Central Asia, Middle East & North Africa, and High Income group has a more gradual decline as they are not very poor to begin with.

Among the income groups, Low & Middle Income, Lower & Middle Income, and Upper Middle Income have quite similar in term of poverty indicator and slope over the year. While these values vary greatly among different geographical regions.

Let's see some important statistics

```
stats <- poverty.headcount %>%
    summarise(count = n(), skewness = skewness(pov),
        kurtosis = kurtosis(pov), std.deviation = sd(pov))

kable(stats)
```

count	skewness	kurtosis	std.deviation
2322	1.5982	1.6611	19.486

2.4. Data Source

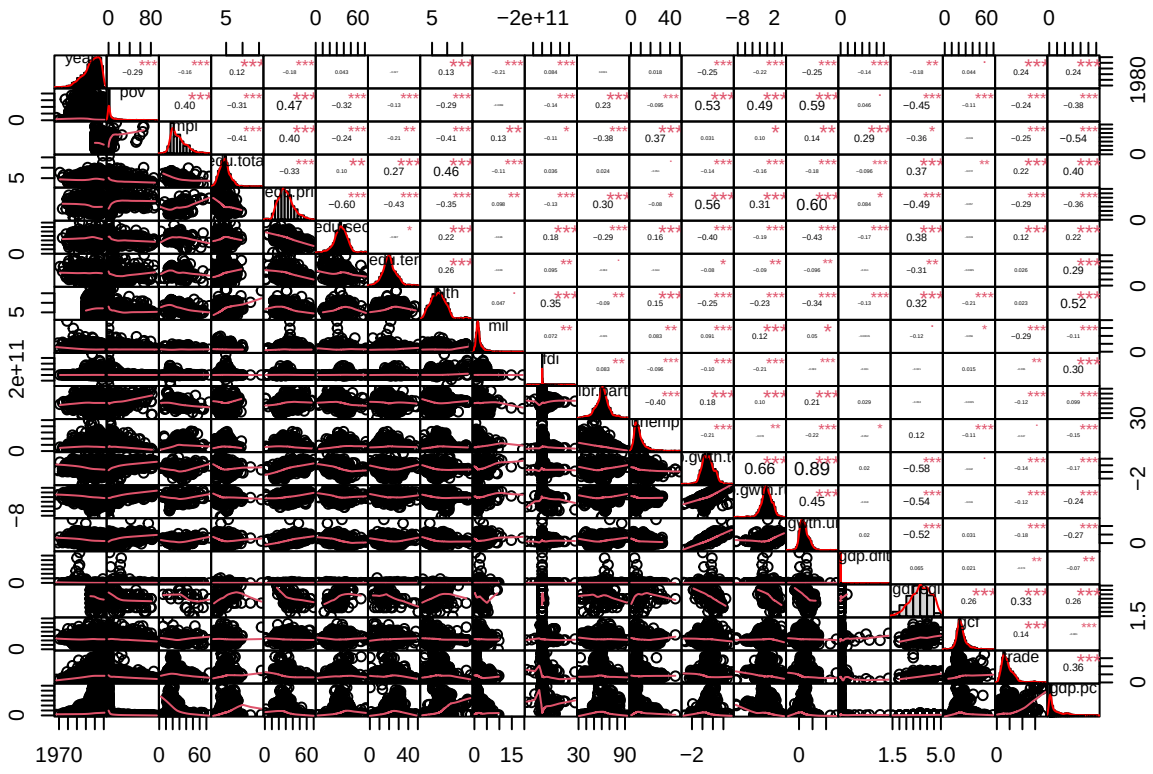
- poverty.headcount
- mpi
- education.expenditure.primary
- education.expenditure.secondary
- education.expenditure.tertiary
- education.expenditure.total
- health.expenditure
- military.expenditure
- fdi
- unemployment.rate
- labour.force.participation
- gender.equality
- population.growth
- urban.population.growth
- rural.population.growth
- gdp.deflator
- gross capital formation
- trade
- region.class
- income.class
- gross.capital.formation

3. Model Selection and Interpretation

3.1. Assumption Check

We can conduct some preliminary checks on linearity and correlation of predictors to have a better picture of the data. Checklist **1. Linear relationship:**
Use correlation matrix to check linearity

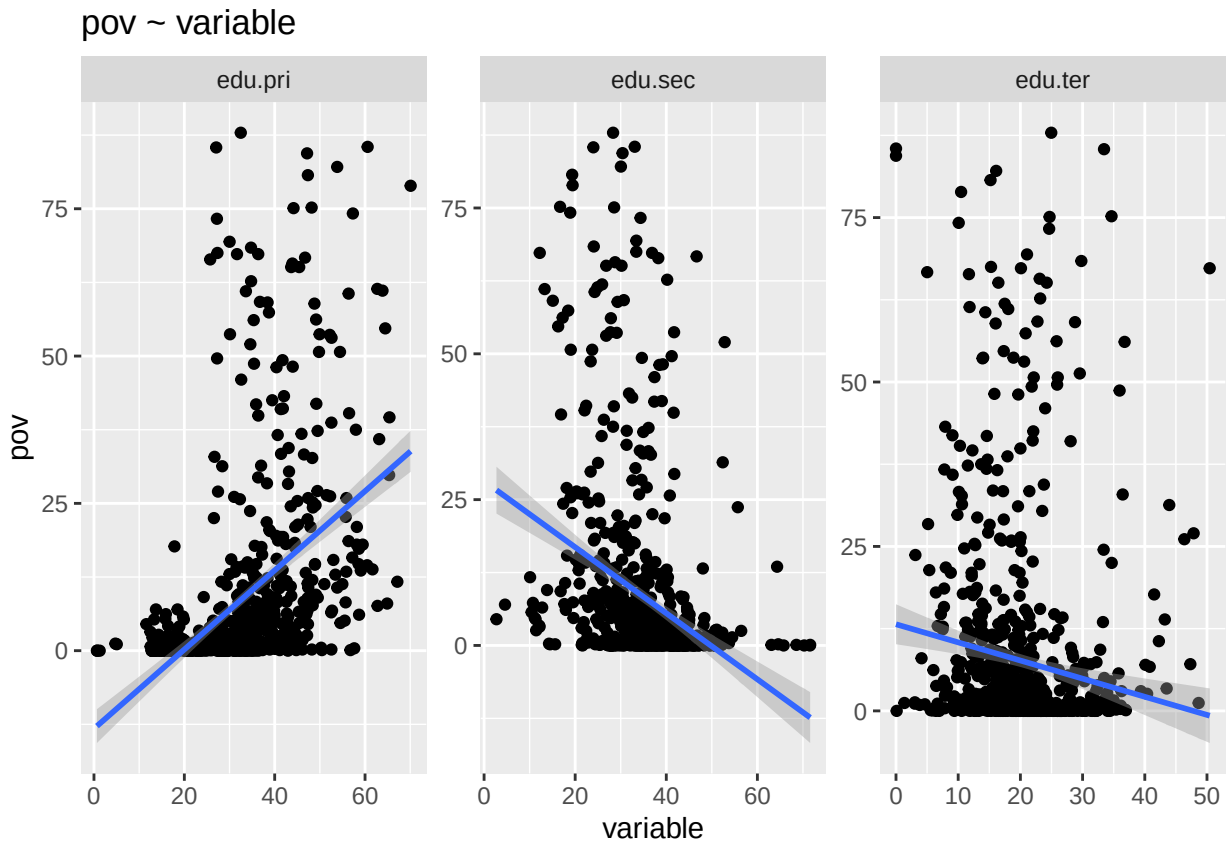
```
# select only numeric data
countries.num <- countries %>%
    select(where(is.numeric))
chart.Correlation(countries.num, histogram = TRUE,
    pch = 19)
```

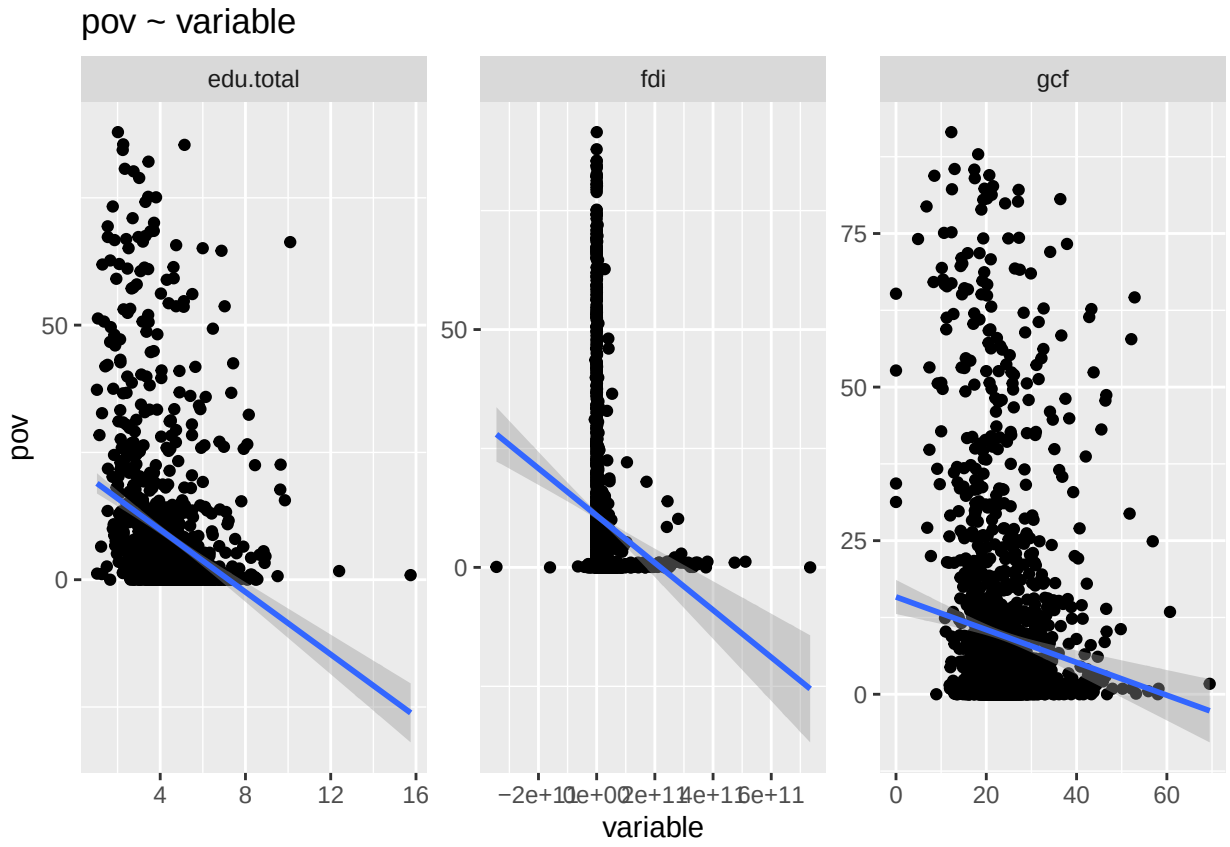


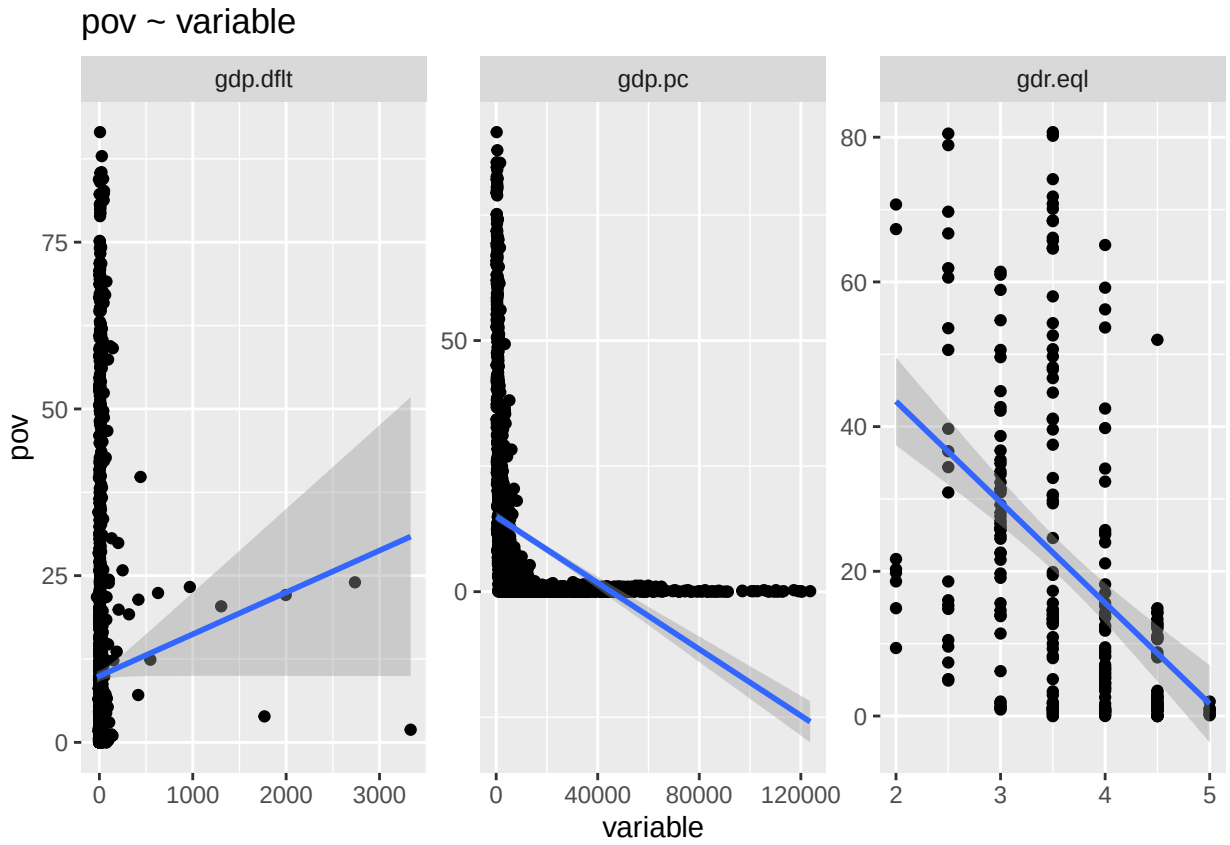
Which is utterly intelligible, but a majority of the fitted lines are linear at first glance. We should render separate graphs for the relationship between pov & other variables.

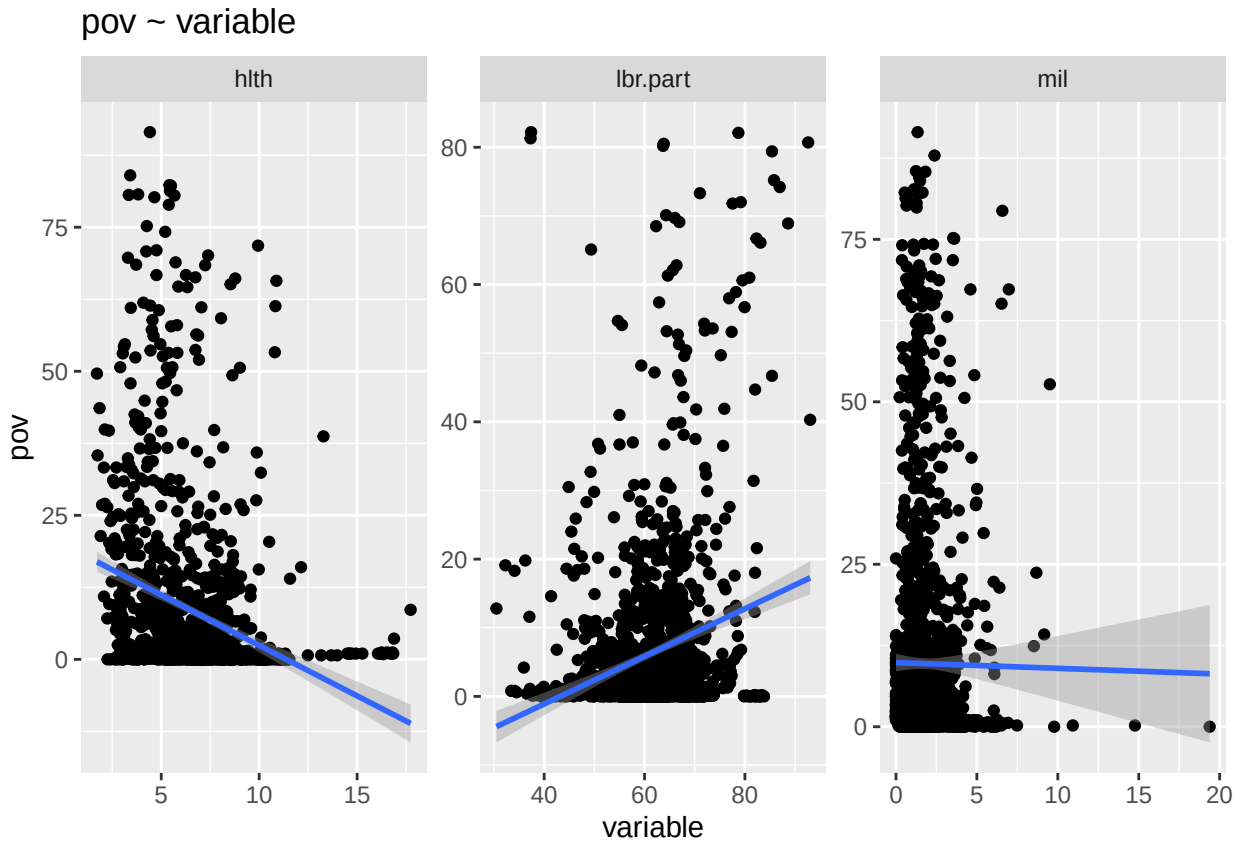
```
# lengthen data table to variable-value pairs,
# with pov as predicted variable and others as
# predictive
countries.num2 <- countries.num %>%
    pivot_longer(cols = 3:ncol(.), names_to = "variable",
        values_to = "value") %>%
    filter(!is.na(value) & !is.na(pov))

drawGraph <- function(indvar, data) {
    ggplot(data %>%
        filter(variable == indvar), aes(x = value,
        y = pov)) + geom_point() + geom_smooth(method = lm) +
        labs(title = paste("Relationship pov ~", indvar),
            x = indvar, y = "pov")
}

ivs <- distinct(countries.num2, variable)[[1]]

# for (indvar in ivs) { print(
# ggplot(countries.num2 %>% filter(variable ==
# indvar), aes(x = value, y = pov)) +
# geom_point() + geom_smooth(formula = y~x,
# method = lm) + labs(title = paste('Relationship
# pov ~', indvar), x = indvar, y = 'pov') ) }
```

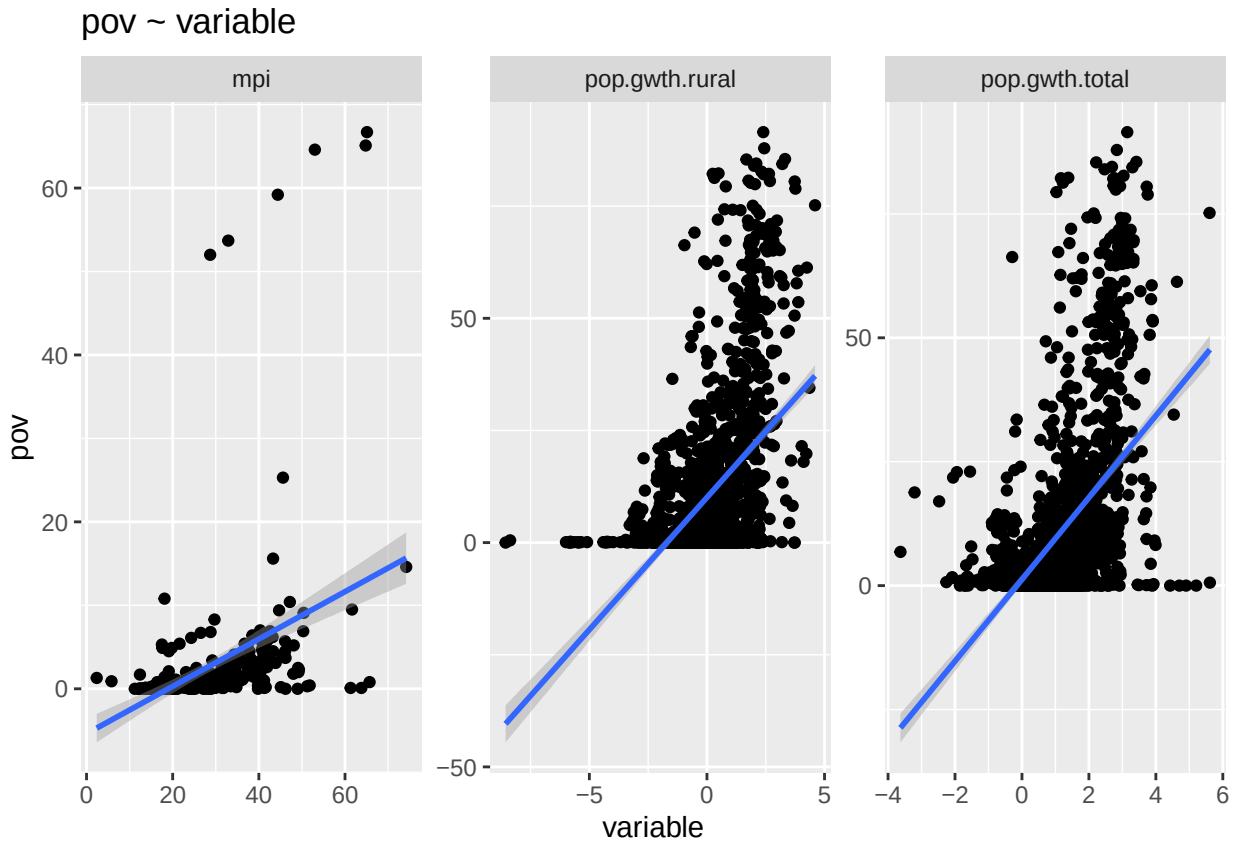
```
p <- ggplot(countries.num2, aes(x = value, y = pov)) +
    geom_point() + geom_smooth(formula = y ~ x, method = lm) +
    labs(title = paste("pov ~ variable"), x = "variable",
        y = "pov") + facet_wrap_paginate(~variable,
    ncol = 3, nrow = 1, scales = "free")

requiredPages <- n_pages(p)

for (i in 1:requiredPages) {
    ggplot(countries.num2, aes(x = value, y = pov)) +
        geom_point() + geom_smooth(formula = y ~ x,
        method = lm) + labs(title = paste("pov ~ variable"),
        x = "variable", y = "pov") + facet_wrap_paginate(~variable,
        ncol = 3, nrow = 1, scales = "free", page = i) ->
        p
    print(p)
}
```

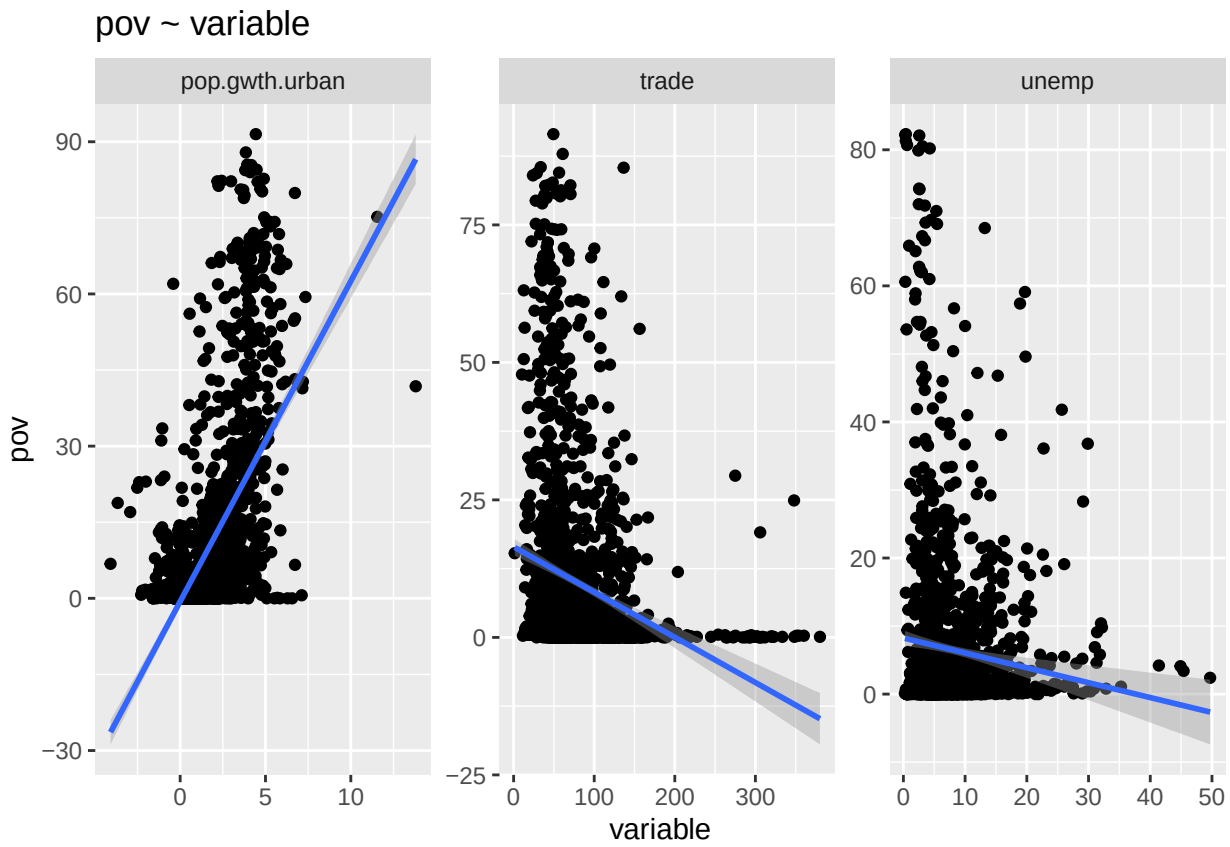



pov ~ variable
edu.total
fdi
gcf
pov
50
0
75
25
4
8
12
16
−2e+11
0e+00
2e+11
4e+11
6e+11
0
20
40
60
variable


pov ~ variable
gdp.dflt
gdp.pc
gdr.eql
pov
variable
0
25
50
75
0
1000
2000
3000
0
40000
80000
120000
0
20
40
60
80
2
3
4
5


pov ~ variable
hlth
lbr.part
mil
pov
variable
0
25
50
75
0
20
40
60
80
0
5
10
15
20
40
60
80

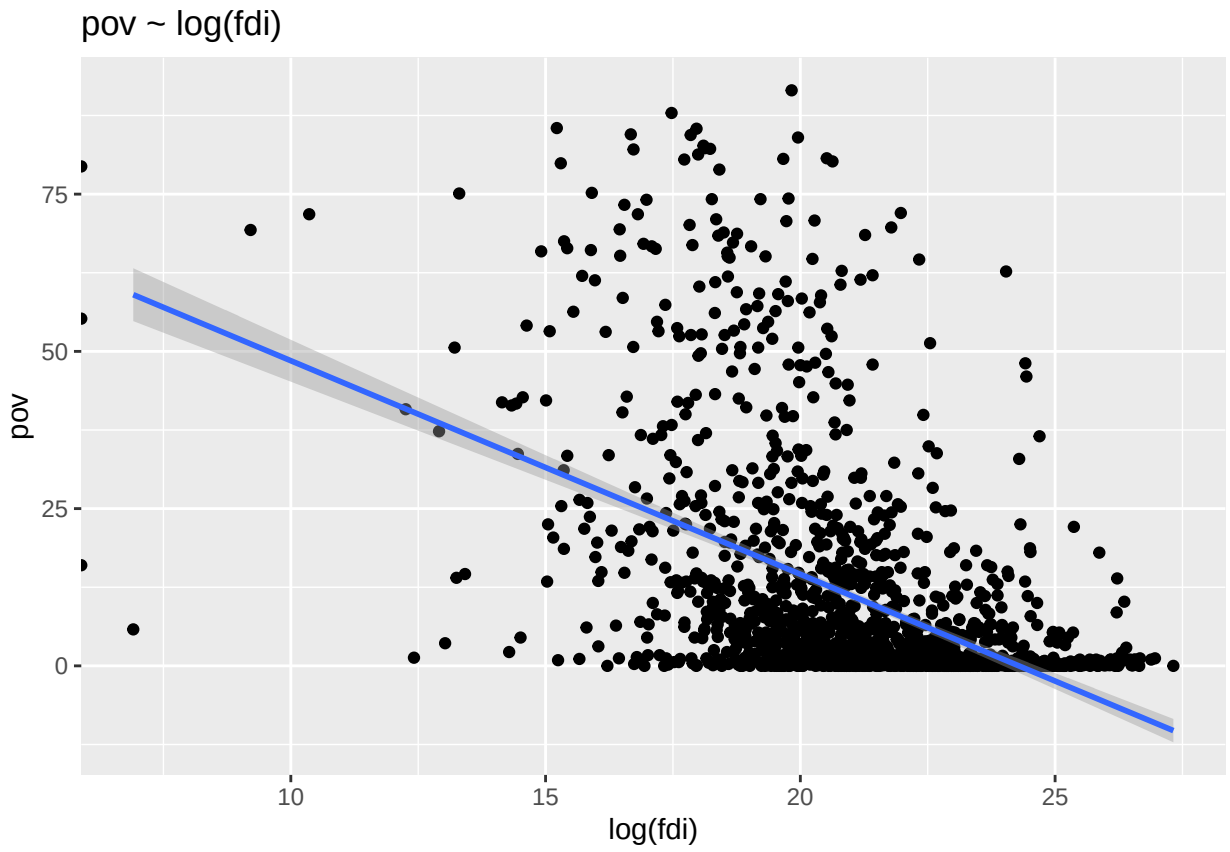

pov ~ variable
mpi
pop.gwth.rural
pop.gwth.total
pov
variable
0
20
40
60
−50
50
−5
5
−4
−2
2
4
6



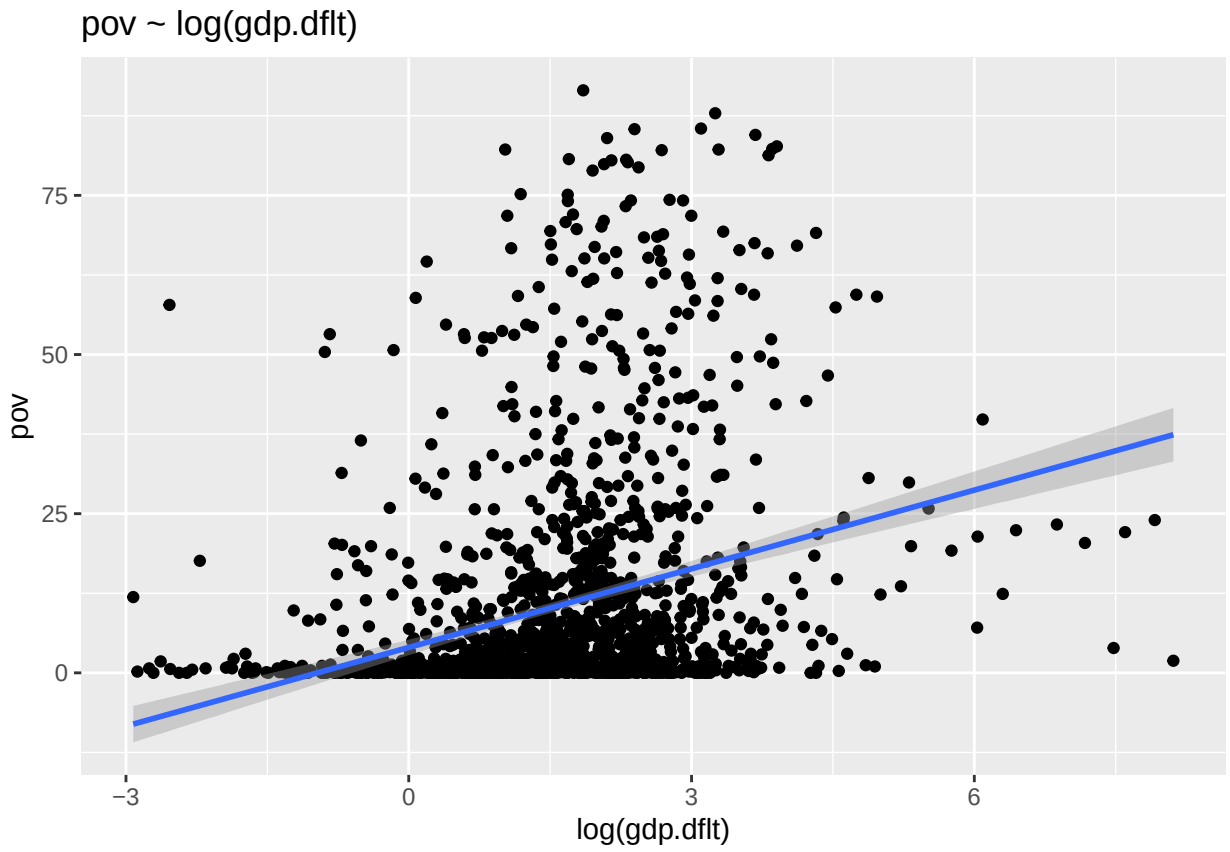
Most variables display linear relationship with some exceptions, which are more appropriate to assume a logarithmic relationship. Gender equality should be viewed as a categorical data.

```
logIvs <- c("fdi", "gdp.dflt", "gdp.pc")

for (indvar in logIvs) {
    print(ggplot(countries.num2 %>%
        filter(variable == indvar), aes(x = log(value),
        y = pov)) + geom_point() + geom_smooth(method = lm) +
        labs(title = paste0("pov ~ log(", indvar, ")"),
            x = paste0("log(", indvar, ")"), y = "pov"))
}
```

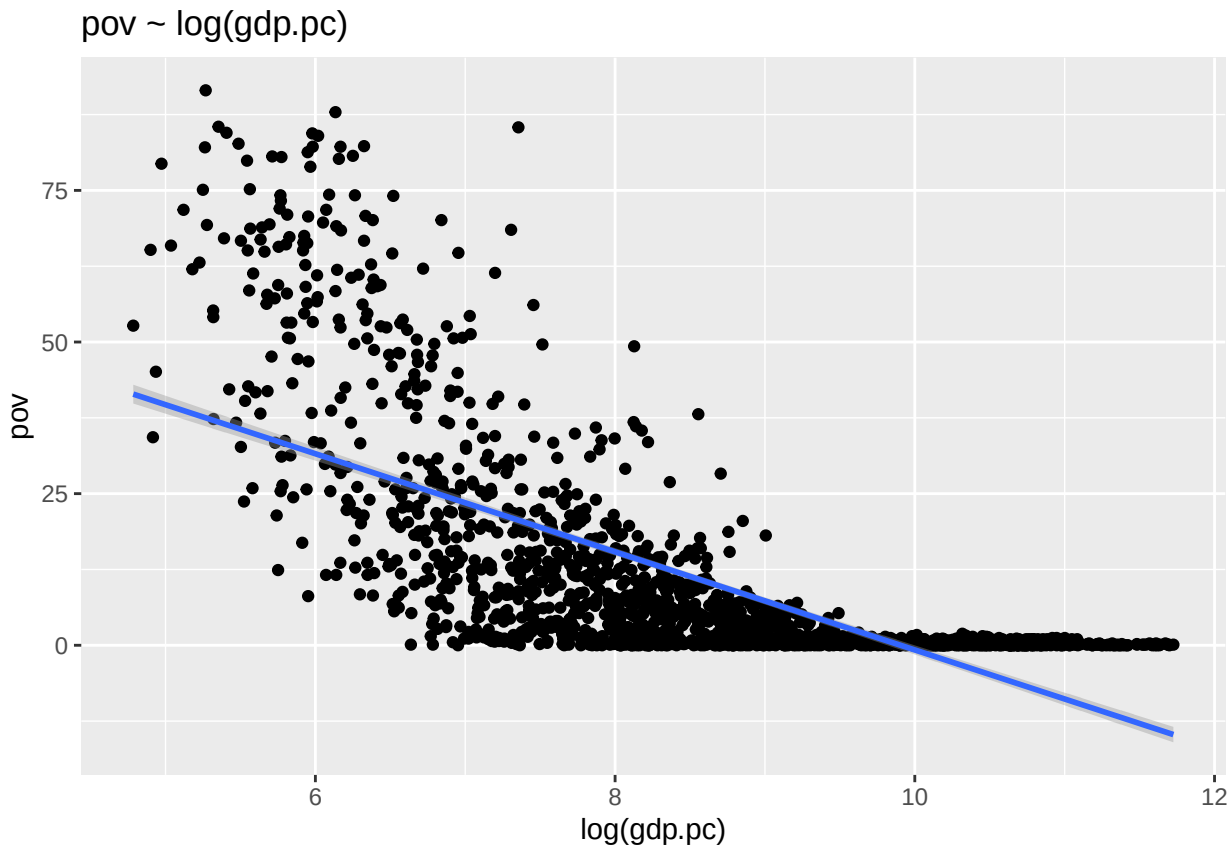
```
## `geom_smooth()` using formula 'y ~ x'
```




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

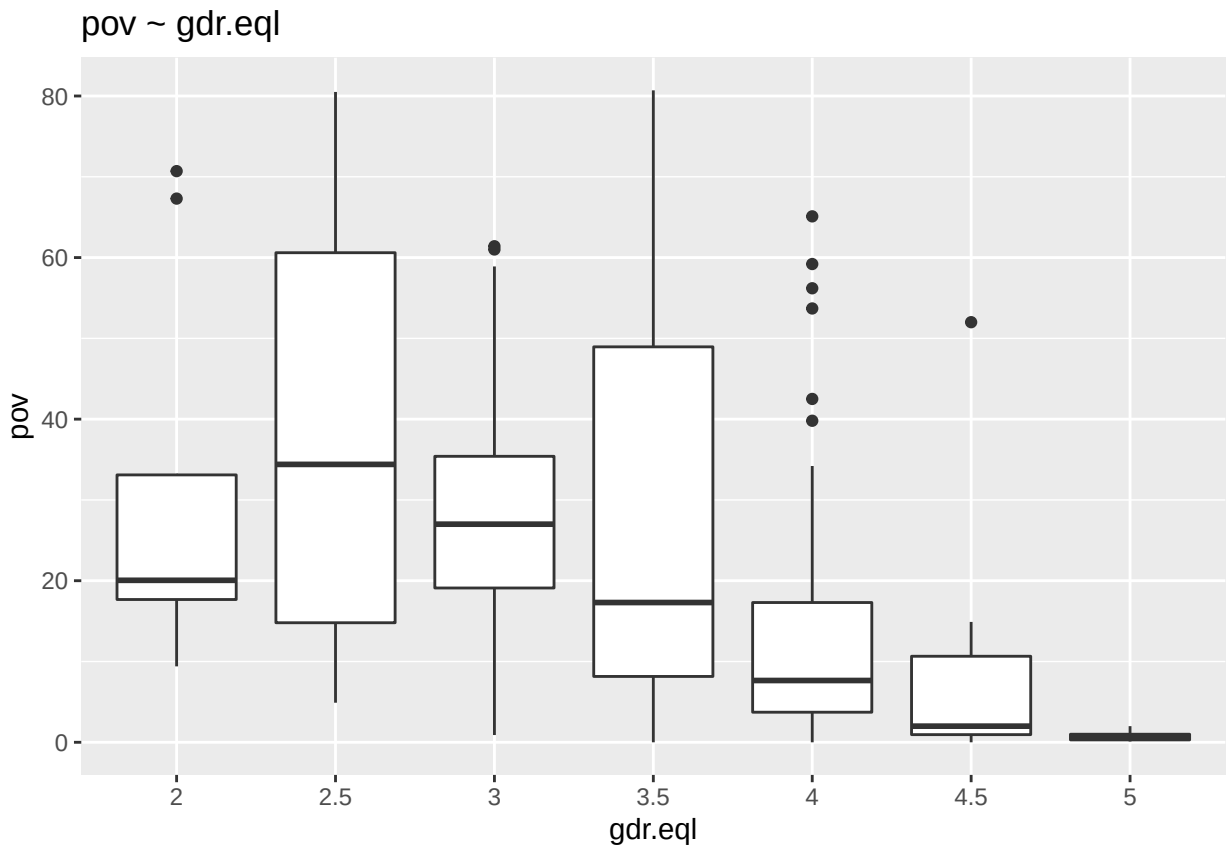



```
## `geom_smooth()` using formula 'y ~ x'
```



The relationship are more linear after the transformation.

```
gdr.eql <- countries.num2 %>%
    filter(variable == "gdr.eql") %>%
    mutate(value = factor(value))

ggplot(gdr.eql, aes(x = value, y = pov)) + geom_boxplot() +
    labs(title = paste("pov ~ gdr.eql"), x = "gdr.eql",
        y = "pov")
```



There's a significant correlation between gender equality and poverty.
Correlation coefficients between pov and predictors.

```
# transform some variables
countries.num3 <- countries.num %>%
    mutate(lfdi = log(fdi), lgdp.dflt = log(gdp.dflt),
        lgdp.pc = log(gdp.pc)) %>%
    mutate(lfdi = ifelse(is.nan(lfdi) | lfdi == -Inf,
        NA, lfdi))

# name of the independent variables
cn <- colnames(countries.num3)[-c(1, 2)]

# correlation of independent variable and pov
corWithPov <- function(indevar, data) {
    cor(data[[indevar]], data[["pov"]], use = "complete.obs")
}

cor.pov <- sapply(cn, corWithPov, data = countries.num3)
kable(cor.pov)
```

	x
mpi	0.40360
edu.total	-0.30981
edu.pri	0.47113

	x
edu.sec	-0.31729
edu.ter	-0.12848
hlth	-0.29384
mil	-0.00694
fdi	-0.14417
lbr.part	0.23168
unemp	-0.09472
pop.gwth.total	0.52906
pop.gwth.rural	0.49432
pop.gwth.urban	0.58665
gdp.dflt	0.04571
gdr.eql	-0.45056
gcf	-0.11085
trade	-0.23872
gdp.pc	-0.37735
lfdi	-0.48760
lgdp.dflt	0.29930
lgdp.pc	-0.70540

`edu.ter`, `mil`, `fdi`, `lbr.part`, `unemp`, `gdp.dflt`, `gcf` have negligible correlation with `pov`. `lgdp.pc`, `lfdi`, and `lgdp.dflt` have stronger linear relationship with `pov` than their un-transformed counterparts.

```
countries1 <- countries1 %>%
    mutate(lfdi = log(fdi), lgdp.dflt = log(gdp.dflt),
        lgdp.pc = log(gdp.pc)) %>%
    mutate(lfdi = ifelse(is.nan(lfdi) | lfdi == -Inf,
        NA, lfdi))
```

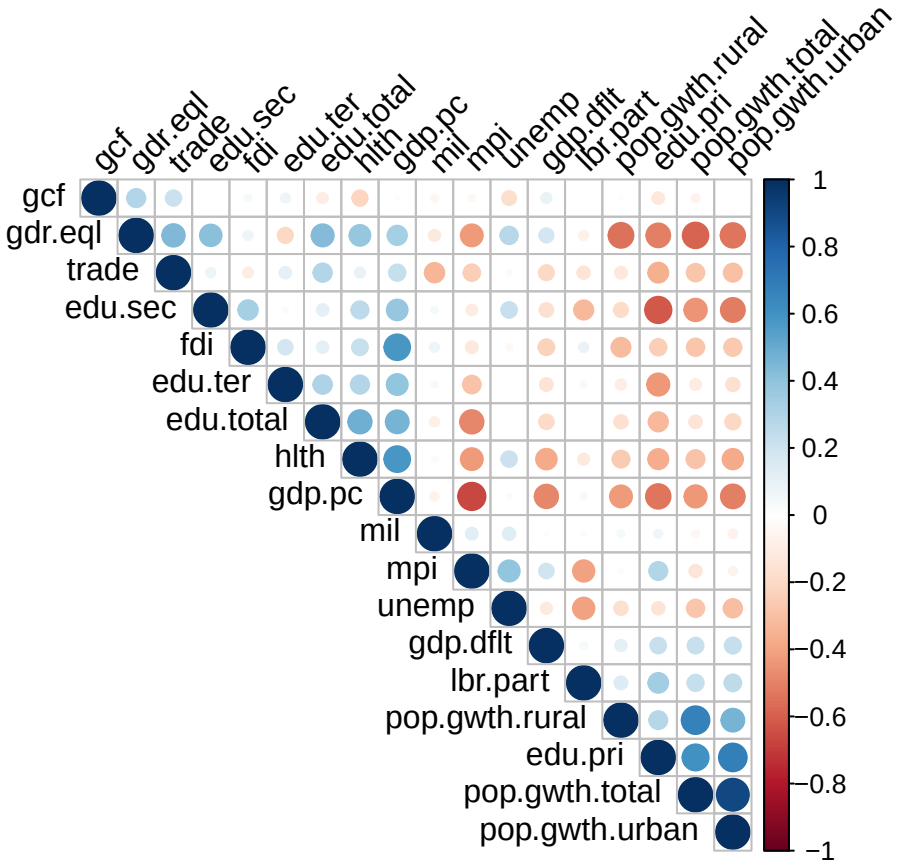
2. Predictors are independent There should be no correlation/multicolinearity between each pair of predictors.

```
countries.arr <- simplify2array(countries.num %>%
    select(-c("year", "pov")))

cor.mtx <- rcorr(countries.arr, type = "spearman")

round(cor.mtx$r, 2)
```

```
##                  mpi edu.total edu.pri edu.sec edu.ter  hlth   mil   fdi
## mpi             1.00     -0.49    0.29   -0.10   -0.28 -0.43  0.13 -0.13
## edu.total      -0.49      1.00   -0.33    0.12    0.31  0.48 -0.08  0.12
## edu.pri         0.29     -0.33    1.00   -0.62   -0.44 -0.36  0.06 -0.24
## edu.sec        -0.10      0.12   -0.62    1.00    0.02  0.26  0.04  0.34
## edu.ter        -0.28      0.31   -0.44    0.02    1.00  0.30  0.04  0.19
## hlth           -0.43      0.48   -0.36    0.26    0.30  1.00 -0.03  0.22
## mil             0.13     -0.08    0.06    0.04    0.04 -0.03  1.00  0.07
## fdi            -0.13      0.12   -0.24    0.34    0.19  0.22  0.07  1.00
## lbr.part       -0.41      0.01    0.35   -0.33   -0.04 -0.11  0.02  0.08
## unemp           0.40      0.00   -0.14    0.22    0.00  0.22  0.14 -0.04
## pop.gwth.total -0.14     -0.14    0.60   -0.45   -0.11 -0.29 -0.05 -0.27
## pop.gwth.rural -0.02     -0.16    0.29   -0.18   -0.10 -0.26  0.04 -0.32
```

```
## pop.gwth.urban -0.06     -0.21    0.68   -0.51   -0.16 -0.37 -0.07 -0.27
## gdp.dflt        0.19     -0.19    0.22   -0.17   -0.15 -0.38  0.01 -0.22
## gdr.eql        -0.43      0.44   -0.50    0.41   -0.21  0.39 -0.11  0.08
## gcf            -0.04     -0.10   -0.12    0.01    0.08 -0.21 -0.05  0.04
## trade          -0.24      0.30   -0.35    0.08    0.12  0.09 -0.34 -0.09
## gdp.pc         -0.66      0.47   -0.53    0.38    0.39  0.59 -0.06  0.59
##                lbr.part unemp pop.gwth.total pop.gwth.rural pop.gwth.urban
## mpi               -0.41  0.40          -0.14          -0.02          -0.06
## edu.total          0.01  0.00          -0.14          -0.16          -0.21
## edu.pri            0.35 -0.14           0.60           0.29           0.68
## edu.sec           -0.33  0.22          -0.45          -0.18          -0.51
## edu.ter           -0.04  0.00          -0.11          -0.10          -0.16
## hlth              -0.11  0.22          -0.29          -0.26          -0.37
## mil                0.02  0.14          -0.05           0.04          -0.07
## fdi                0.08 -0.04          -0.27          -0.32          -0.27
## lbr.part           1.00 -0.41           0.24           0.15           0.25
## unemp             -0.41  1.00          -0.27          -0.16          -0.30
## pop.gwth.total     0.24 -0.27           1.00           0.68           0.91
## pop.gwth.rural     0.15 -0.16           0.68           1.00           0.46
## pop.gwth.urban     0.25 -0.30           0.91           0.46           1.00
## gdp.dflt           0.05 -0.11           0.22           0.11           0.23
## gdr.eql           -0.08  0.27          -0.58          -0.55          -0.53
## gcf                0.01 -0.17          -0.06           0.01           0.00
## trade             -0.14 -0.03          -0.28          -0.13          -0.29
## gdp.pc            -0.04  0.02          -0.44          -0.43          -0.50
##                gdp.dflt gdr.eql   gcf trade gdp.pc
## mpi                0.19   -0.43 -0.04 -0.24  -0.66
## edu.total         -0.19    0.44 -0.10  0.30   0.47
## edu.pri            0.22   -0.50 -0.12 -0.35  -0.53
## edu.sec           -0.17    0.41  0.01  0.08   0.38
## edu.ter           -0.15   -0.21  0.08  0.12   0.39
## hlth              -0.38    0.39 -0.21  0.09   0.59
## mil                0.01   -0.11 -0.05 -0.34  -0.06
## fdi               -0.22    0.08  0.04 -0.09   0.59
## lbr.part           0.05   -0.08  0.01 -0.14  -0.04
## unemp             -0.11    0.27 -0.17 -0.03   0.02
## pop.gwth.total     0.22   -0.58 -0.06 -0.28  -0.44
## pop.gwth.rural     0.11   -0.55  0.01 -0.13  -0.43
## pop.gwth.urban     0.23   -0.53  0.00 -0.29  -0.50
## gdp.dflt           1.00    0.18  0.09 -0.20  -0.49
## gdr.eql            0.18    1.00  0.30  0.45   0.33
## gcf                0.09    0.30  1.00  0.21  -0.01
## trade             -0.20    0.45  0.21  1.00   0.24
## gdp.pc            -0.49    0.33 -0.01  0.24   1.00
```

```
corrplot(cor.mtx$r, type = "upper", order = "hclust",
    tl.col = "black", tl.srt = 45)
```



Some significant correlations can be found between `gdr.eql` with `edu.pri`, `pop.gwth.total`, and `fdi` with `mil`, etc. We expect these variables to be eliminated in the VIF test. This might be a good property for imputation (3.2.4)

3.2. Ordinary Multiple Linear Regression

We conduct a normal linear regression, following the approaches mentioned above to address missing values issues.

3.2.1. Use Complete Cases (To be done)

3.2.1.1. Model Fitting

3.2.1.2. Assessment

3.2.1.3. Interpretation

3.2.2. Selectively remove variables with high missing rate (To be done)

3.2.2.1. Model Fitting

3.2.2.2. Assessment

3.2.2.3. Interpretation

3.2.3. Update the data set as we select variables (To be done)

3.2.3.1. Model Fitting

3.2.3.2. Assessment

3.2.3.3. Interpretation

3.2.4. Imputation

1. What is imputation
Imputation is a technique to handle missing values by replacing missing data with substitute values. In our study, we focus on "item imputation", which means substituting for a component of a data point (i.e. a variable). The general idea is to take a value that preserve the property of the data (e.g., distribution, mean, standard deviation), and use other variables to predict the missing values (much like regression). Since we observed some correlations during descriptive analysis, we expect the imputation algorithm to work well.

We use **Multivariate Imputation By Chained Equations (MICE)** as primary tool for this technique, because of its wide acceptance in scientific studies (Alruhaymi and Kim (2021)). In order to use this technique, we have to assume that the "missingness" of a field can be explained by the values in other columns, (e.g., If the countries is in North America, it's more likely to be missing as we have discovered in 2.2). The general ideas of the algorithm is to fill in the missing values, and improve it iteratively until predicted values converse to a stable point. The algorithmic details of MICE is very concisely (and enthrallingly) explained in Gopalan (2020).

We use relatively small parameters in the interest of time.

- `m = 5` imputed data sets
- `maxit = 30` iterations

The imputation method is `cart` (Classification and Regression Trees).

```
set.seed(1984)
# split data
isComplete <- which(complete.cases(countries1))
idx <- sample(isComplete, replace = F, 0.3 * nrow(countries1))
countries.train.4 <- countries1[-idx, ]
countries.test.4 <- countries1[idx, ]

# blank for no imputation, cart for variable
# requiring imputation
meth <- c(rep("", 4), "cart", "", rep("cart", 16))
names(meth) <- colnames(countries1)
meth
```

```
##   country.code   country.name           year            pov         income
##             ""             ""             ""             ""         "cart"
##            reg      edu.total           hlth            mil            fdi
##             ""         "cart"         "cart"         "cart"         "cart"
```

```
##       lbr.part          unemp pop.gwth.total pop.gwth.rural pop.gwth.urban
##         "cart"         "cart"         "cart"         "cart"         "cart"
##       gdp.dflt            gcf          trade         gdp.pc           lfdi
##         "cart"         "cart"         "cart"         "cart"         "cart"
##      lgdp.dflt        lgdp.pc
##         "cart"         "cart"
```

```
# run the algorithm countries1.imputed <-
# mice(countries1, m = 3, maxit = 20, method =
# meth) saveRDS(countries1.imputed,
# 'countries1.imputed.RData')
countries1.imputed <- readRDS("countries1.imputed.RData")
summary(countries1.imputed)
```

```
## Class: mids
## Number of multiple imputations:  3
## Imputation methods:
##   country.code   country.name           year            pov         income
##             ""             ""             ""             ""         "cart"
##            reg      edu.total           hlth            mil            fdi
##             ""         "cart"         "cart"         "cart"         "cart"
##       lbr.part          unemp pop.gwth.total pop.gwth.rural pop.gwth.urban
##         "cart"         "cart"             ""         "cart"         "cart"
##       gdp.dflt            gcf          trade         gdp.pc           lfdi
##         "cart"         "cart"         "cart"         "cart"         "cart"
##      lgdp.dflt        lgdp.pc
##         "cart"         "cart"
## PredictorMatrix:
##              country.code country.name year pov income reg edu.total hlth mil
## country.code            0            0    1   1      1   1         1    1   1
## country.name            1            0    1   1      1   1         1    1   1
## year                    1            0    0   1      1   1         1    1   1
## pov                     1            0    1   0      1   1         1    1   1
## income                  1            0    1   1      0   1         1    1   1
## reg                     1            0    1   1      1   0         1    1   1
##              fdi lbr.part unemp pop.gwth.total pop.gwth.rural pop.gwth.urban
## country.code   1        1     1              1              1              1
## country.name   1        1     1              1              1              1
## year           1        1     1              1              1              1
## pov            1        1     1              1              1              1
## income         1        1     1              1              1              1
## reg            1        1     1              1              1              1
##              gdp.dflt gcf trade gdp.pc lfdi lgdp.dflt lgdp.pc
## country.code        1   1     1      1    1         1       1
## country.name        1   1     1      1    1         1       1
## year                1   1     1      1    1         1       1
## pov                 1   1     1      1    1         1       1
## income              1   1     1      1    1         1       1
## reg                 1   1     1      1    1         1       1
## Number of logged events:  961
##   it im       dep     meth
## 1  0  0           constant
## 2  1  1    income     cart
## 3  1  1 edu.total     cart
```

```
## 4  1  1      hlth     cart
## 5  1  1       mil     cart
## 6  1  1       fdi     cart
##
## 1
## 2
## 3            country.codeARE, country.codeBIH, country.codeCOD, country.codeDZA, country.codeEAS, cou
## 4
## 5 country.codeBTN, country.codeCOM, country.codeDJI, country.codeEAS, country.codeECS, country.codeF
## 6
```

We can take a look into one of the imputated data sets.

```
countries1.imputed.1 <- complete(countries1.imputed,
    1)
summary(countries1.imputed.1)
```

```
##   country.code  country.name            year           pov       income
##  BRA    :  36   Length:1843        Min.   :1967   Min.   : 0.0   H :623
##  CRI    :  34   Class :character   1st Qu.:2001   1st Qu.: 0.2   L :261
##  ARG    :  32   Mode  :character   Median :2008   Median : 1.5   LM:509
##  USA    :  32                      Mean   :2007   Mean   :10.0   UM:450
##  HND    :  30                      3rd Qu.:2014   3rd Qu.:11.6
##  GBR    :  29                      Max.   :2021   Max.   :91.5
##  (Other):1650
##                           reg        edu.total          hlth
##  East Asia & Pacific       :167   Min.   : 1.03   Min.   : 1.72
##  Europe & Central Asia     :845   1st Qu.: 3.44   1st Qu.: 4.98
##  Latin America & Caribbean :416   Median : 4.39   Median : 6.77
##  Middle East & North Africa:104   Mean   : 4.48   Mean   : 6.81
##  North America             : 50   3rd Qu.: 5.38   3rd Qu.: 8.46
##  South Asia                : 53   Max.   :15.75   Max.   :17.73
##  Sub-Saharan Africa        :208
##       mil             fdi               lbr.part        unemp
##  Min.   : 0.00   Min.   :-3.44e+11   Min.   :30.5   Min.   : 0.25
##  1st Qu.: 1.04   1st Qu.: 2.98e+08   1st Qu.:56.3   1st Qu.: 4.42
##  Median : 1.47   Median : 1.70e+09   Median :61.5   Median : 6.77
##  Mean   : 1.78   Mean   : 1.61e+10   Mean   :61.1   Mean   : 8.08
##  3rd Qu.: 2.10   3rd Qu.: 9.79e+09   3rd Qu.:65.9   3rd Qu.:10.07
##  Max.   :19.38   Max.   : 7.34e+11   Max.   :93.0   Max.   :49.70
##
##  pop.gwth.total   pop.gwth.rural    pop.gwth.urban      gdp.dflt
##  Min.   :-3.630   Min.   :-8.5607   Min.   :-4.078   Min.   : -26.3
##  1st Qu.: 0.284   1st Qu.:-0.8468   1st Qu.: 0.517   1st Qu.:   1.7
##  Median : 1.038   Median :-0.0211   Median : 1.485   Median :   3.9
##  Mean   : 1.064   Mean   : 0.0053   Mean   : 1.692   Mean   :  16.1
##  3rd Qu.: 1.776   3rd Qu.: 0.9597   3rd Qu.: 2.652   3rd Qu.:   8.6
##  Max.   : 5.614   Max.   : 4.5969   Max.   :13.805   Max.   :3333.6
##
##       gcf            trade             gdp.pc            lfdi
##  Min.   : 0.0   Min.   :  1.38   Min.   :   120   Min.   : 6.91
##  1st Qu.:19.6   1st Qu.: 50.62   1st Qu.:  1904   1st Qu.:19.51
##  Median :22.7   Median : 72.61   Median :  5913   Median :21.26
```

```
##  Mean   :23.9   Mean   : 83.51   Mean   : 14984   Mean   :20.77
##  3rd Qu.:26.8   3rd Qu.:104.71   3rd Qu.: 20863   3rd Qu.:23.01
##  Max.   :69.5   Max.   :380.10   Max.   :123679   Max.   :27.32
##
##    lgdp.dflt        lgdp.pc
##  Min.   :-2.92   Min.   : 4.78
##  1st Qu.: 0.55   1st Qu.: 7.55
##  Median : 1.36   Median : 8.69
##  Mean   : 1.23   Mean   : 8.67
##  3rd Qu.: 2.15   3rd Qu.: 9.95
##  Max.   : 8.11   Max.   :11.73
##
```

```
sum(!complete.cases(countries1.imputed.1))
```

```
## [1] 0
```

We found no missing cases as expected.

3.2.4.1. Model Fitting

A. Ordinary Linear Regression We generated 3 sets of imputated (training) data. We can build separate models using each data set, and combine the estimates using pooling rule. With all the generated data sets.

```
formula.str <- "pov ~ country.code+year+income+edu.total+hlth+mil+fdi+lbr.part+unemp+pop.gwth.total+pop

# Using single imputed data set
fit1 <- lapply(1:countries1.imputed$m, function(i) lm(as.formula(formula.str),
    data = complete(countries1.imputed, i)))

# Build models with all generated data set and
# pool the estimates
fit2 <- with(data = countries1.imputed, exp = lm(as.formula(formula.str)))
fit2.combined <- pool(fit2)
# dummy lm model
fit2.dummy <- lm(as.formula(formula.str), data = complete(countries1.imputed,
    1))
# replace coefficients of dummy model
name.coef <- names(fit2.dummy$coefficients)
fit2.dummy$coefficients <- fit2.combined$pooled$estimate
names(fit2.dummy$coefficients) <- name.coef
```

Helper function to calculate R-Squared

```
r2 <- function(pred, orig) {
    RSS <- sum((pred - orig)^2)
    TSS <- sum((orig - mean(orig))^2)
    R2 <- 1 - RSS/TSS
    return(R2)
}
```

```
adjR2 <- function(pred, orig, k) {
    R2 <- r2(pred, orig)
    n <- length(pred)
    adjr2 <- 1 - (1 - R2) * (n - 1)/(n - k - 1)
    return(adjr2)
}
```

Adjusted R-Squared on train and test

```
fit1.summ <- lapply(fit1, function(f) {
    # number of variables
    k <- length(f$coefficients) - 1
    # predict value of test
    pred <- predict(f, countries.test.4)
    test.resid <- pred - countries.test.4$pov
    data.frame(`Train MSE` = mean(summary(f)$residuals^2),
        `Test MSE` = mean(test.resid^2), `Train MAE` = mean(abs(summary(f)$residuals)),
        `Test MAE` = mean(abs(test.resid)), `Train Adj.R2` = summary(f)$adj.r.squared,
        `Test Adj.R2` = adjR2(pred, countries.test.4$pov,
            k))
})

# number of variables
k <- length(fit2.dummy$coefficients) - 1
# number of variables
fit2.pred <- predict(fit2.dummy, countries.test.4)
test.resid <- fit2.pred - countries.test.4$pov
fit2.summ <- data.frame(`Train MSE` = mean(summary(fit2.dummy)$residuals^2),
    `Test MSE` = mean(test.resid^2), `Train MAE` = mean(abs(summary(fit2.dummy)$residuals)),
    `Test MAE` = mean(abs(test.resid)), `Train Adj.R2` = pool.r.squared(fit2,
        adjusted = T)[, "est"], `Test Adj.R2` = adjR2(fit2.pred,
        countries.test.4$pov, k))

res <- do.call(rbind.data.frame, fit1.summ)
res <- rbind(res, fit2.summ)
res <- cbind(data.frame(Set = c(1:3, "Pooled")), res)

kable(res)
```

Set	Train.MSE	Test.MSE	Train.MAE	Test.MAE	Train.Adj.R2	Test.Adj.R2
1	25.124	12.035	3.0606	2.1255	0.90902	0.70000
2	26.225	11.650	3.0897	2.0850	0.90504	0.70958
3	26.671	11.673	3.0974	2.0768	0.90343	0.70901
Pooled	25.124	11.718	3.0606	2.0883	0.90586	0.70790

Pooled model slightly improve performance. Test data fitting has better MSE and MAE than train data. However, Adjusted R-Squared is lower in test data. There might be some over-fitting in our models.

```
# find coef of variable other than country.code
var <- c("year", "incomeL", "incomeLM", "incomeUM",
    "edu.total", "hlth", "mil", "fdi", "lbr.part",
```

```r
    "unemp", "pop.gwth.total", "pop.gwth.rural", "pop.gwth.urban",
    "gdp.dflt", "gcf", "trade", "gdp.pc", "lfdi", "lgdp.dflt",
    "lgdp.pc")

res <- do.call(rbind, lapply(fit1, function(f) f$coefficients[var]))
res <- rbind(res, fit2.dummy$coefficients[var])
res <- cbind(data.frame(Set = c(1:3, "Pooled")), res)
kable(res)
```

Set	year	incomeL	incomeLM	incomeUM	edu.total	hlth	mil	fdi	lbr.part	unemp	pop.gwth.total	pop.gwth.rural	pop.gwth.urban	gdp.dflt	gcf	trade	gdp.pc	lfdi	lgdp.dflt	lgdp.pc
1	-0.17936	-1.1220	-8.0159	-9.2676	-0.49380	0.33154	-0.22233	0	0.23734	-0.06299	-1.1178	0.68142	2.3650	0.00023	-0.18652	-0.01465	0.00030	-0.00018	0.03332	-8.1819
2	-0.23886	-8.6675	-5.0584	-4.7288	-0.44528	0.45655	-0.33947	0	0.20415	-0.06834	-3.0483	0.64041	2.3354	0.00043	-0.22869	-0.00471	0.00028	0.00039	0.11563	-6.7202
3	-0.21786	0.09565	-5.4153	-2.9036	-0.54513	0.25428	-0.08665	0	0.13605	-0.08213	-2.3296	0.71585	2.4598	-0.00001	-0.18836	-0.00816	0.00003	0.09561	0.03814	-7.1215
Pooled	-0.21204	-0.56465	-5.4965	-2.9666	-0.49474	0.34746	-0.21615	0	0.19251	-0.07131	-5.1652	0.67922	2.3867	0.00018	-0.20101	-0.00917	0.00029	0.03194	0.06236	-7.3412

In contradiction to expectation, `hlth` is positively correlated with `pov`, and lower-middle income countries (incomeLM) are less poor than high income countries (incomeH). `pop.gwth.total` is negatively related to `pov`, while `pop.gwth.rural` and `pop.gwth.urban` are positively related. They might be indications of over-fitting model.

```r
# fit model on imputated set no. 1
summary(fit1[[1]])
```

```
##
## Call:
## lm(formula = as.formula(formula.str), data = complete(countries1.imputed,
##     i))
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -25.33  -1.72   0.11   1.63  32.72
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      4.40e+02   6.20e+01    7.10  1.8e-12 ***
## country.codeALB -1.82e+01   3.75e+00   -4.84  1.4e-06 ***
## country.codeARE -1.17e+01   5.21e+00   -2.25  0.02440 *
## country.codeARG -1.43e+01   3.58e+00   -4.00  6.5e-05 ***
## country.codeARM -1.59e+01   3.57e+00   -4.46  8.8e-06 ***
## country.codeAUS -1.53e+01   3.97e+00   -3.86  0.00012 ***
## country.codeAUT -1.37e+01   3.98e+00   -3.45  0.00057 ***
## country.codeAZE -2.67e+01   3.93e+00   -6.79  1.5e-11 ***
## country.codeBDI  1.51e+01   4.35e+00    3.47  0.00053 ***
## country.codeBEL -9.96e+00   3.98e+00   -2.51  0.01234 *
## country.codeBEN  8.38e+00   4.20e+00    1.99  0.04625 *
## country.codeBFA  9.86e+00   3.98e+00    2.48  0.01333 *
## country.codeBGD -1.66e+01   3.77e+00   -4.41  1.1e-05 ***
```

```
## country.codeBGR -9.82e+00   3.76e+00   -2.61  0.00903 **
## country.codeBIH -1.40e+01   4.35e+00   -3.21  0.00136 **
## country.codeBLR -1.37e+01   3.61e+00   -3.80  0.00015 ***
## country.codeBLZ  1.62e+00   3.96e+00    0.41  0.68196
## country.codeBOL -1.45e+01   3.46e+00   -4.18  3.0e-05 ***
## country.codeBRA -7.94e+00   3.49e+00   -2.28  0.02300 *
## country.codeBTN -1.36e+01   4.25e+00   -3.20  0.00139 **
## country.codeBWA  3.19e+00   4.06e+00    0.79  0.43190
## country.codeCAF  3.21e+01   4.96e+00    6.47  1.3e-10 ***
## country.codeCAN -1.58e+01   3.90e+00   -4.06  5.1e-05 ***
## country.codeCHE -2.03e+01   4.22e+00   -4.80  1.7e-06 ***
## country.codeCHL -1.14e+01   3.61e+00   -3.15  0.00164 **
## country.codeCHN -3.39e+00   3.68e+00   -0.92  0.35631
## country.codeCIV -1.10e+01   3.60e+00   -3.05  0.00232 **
## country.codeCMR -2.09e+00   4.17e+00   -0.50  0.61704
## country.codeCOD  3.26e+01   5.01e+00    6.51  9.9e-11 ***
## country.codeCOG  2.28e+01   4.90e+00    4.67  3.3e-06 ***
## country.codeCOL -8.34e+00   3.47e+00   -2.40  0.01631 *
## country.codeCOM -1.04e+01   4.95e+00   -2.10  0.03557 *
## country.codeCPV -5.64e+00   4.47e+00   -1.26  0.20749
## country.codeCRI -1.40e+01   3.49e+00   -4.00  6.7e-05 ***
## country.codeCYP -1.15e+01   3.81e+00   -3.02  0.00259 **
## country.codeCZE -1.20e+01   3.75e+00   -3.20  0.00142 **
## country.codeDEU -1.51e+01   3.95e+00   -3.82  0.00014 ***
## country.codeDJI  6.45e+00   4.26e+00    1.51  0.13090
## country.codeDNK -1.52e+01   4.08e+00   -3.74  0.00019 ***
## country.codeDOM -1.35e+01   3.46e+00   -3.89  0.00010 ***
## country.codeDZA -1.53e+01   4.46e+00   -3.44  0.00060 ***
## country.codeECU -9.71e+00   3.45e+00   -2.82  0.00489 **
## country.codeEGY -1.81e+01   3.67e+00   -4.94  8.5e-07 ***
## country.codeESP -1.22e+01   3.77e+00   -3.25  0.00119 **
## country.codeEST -9.84e+00   3.81e+00   -2.58  0.00985 **
## country.codeETH -8.00e+00   4.11e+00   -1.95  0.05170 .
## country.codeFIN -1.38e+01   3.96e+00   -3.48  0.00052 ***
## country.codeFJI -1.39e+01   4.23e+00   -3.29  0.00103 **
## country.codeFRA -1.39e+01   3.91e+00   -3.55  0.00040 ***
## country.codeFSM -5.26e+00   5.17e+00   -1.02  0.30904
## country.codeGAB -8.62e+00   4.95e+00   -1.74  0.08190 .
## country.codeGBR -1.62e+01   3.86e+00   -4.20  2.8e-05 ***
## country.codeGEO -1.16e+01   3.61e+00   -3.20  0.00138 **
## country.codeGHA  9.15e+00   3.82e+00    2.40  0.01662 *
## country.codeGIN  3.75e+00   3.86e+00    0.97  0.33152
## country.codeGMB  9.85e-01   4.18e+00    0.24  0.81386
## country.codeGNB  1.06e+00   4.02e+00    0.26  0.79127
## country.codeGRC -1.15e+01   3.81e+00   -3.01  0.00266 **
## country.codeGTM -6.42e+00   3.89e+00   -1.65  0.09956 .
## country.codeGUY -6.99e+00   5.22e+00   -1.34  0.18080
## country.codeHND -6.04e+00   3.35e+00   -1.80  0.07181 .
## country.codeHRV -1.13e+01   3.89e+00   -2.90  0.00380 **
## country.codeHTI -2.48e+00   6.23e+00   -0.40  0.69081
## country.codeHUN -1.01e+01   3.81e+00   -2.64  0.00825 **
## country.codeIDN  1.10e+00   3.40e+00    0.32  0.74710
## country.codeIND  5.79e-01   3.61e+00    0.16  0.87241
## country.codeIRL -1.44e+01   3.98e+00   -3.62  0.00031 ***
```

```
## country.codeIRN -1.09e+01   3.56e+00   -3.06  0.00225 **
## country.codeIRQ -1.38e+01   4.92e+00   -2.81  0.00509 **
## country.codeISL -1.91e+01   4.07e+00   -4.71  2.7e-06 ***
## country.codeISR -1.20e+01   3.74e+00   -3.22  0.00132 **
## country.codeITA -1.18e+01   3.87e+00   -3.04  0.00238 **
## country.codeJAM -1.74e+01   3.89e+00   -4.46  8.7e-06 ***
## country.codeJOR -1.42e+01   3.88e+00   -3.65  0.00027 ***
## country.codeJPN -1.24e+01   5.10e+00   -2.44  0.01470 *
## country.codeKAZ -1.41e+01   3.52e+00   -4.00  6.8e-05 ***
## country.codeKEN -1.16e+01   4.02e+00   -2.89  0.00395 **
## country.codeKGZ -2.11e+01   3.45e+00   -6.12  1.2e-09 ***
## country.codeKIR -1.68e+01   5.02e+00   -3.34  0.00085 ***
## country.codeKOR -1.05e+01   4.17e+00   -2.51  0.01216 *
## country.codeLAO -1.65e+01   3.91e+00   -4.21  2.7e-05 ***
## country.codeLBN -1.12e+01   6.23e+00   -1.80  0.07260 .
## country.codeLBR -6.87e+00   4.56e+00   -1.51  0.13192
## country.codeLCA  8.07e+00   5.04e+00    1.60  0.10979
## country.codeLKA -1.78e+01   3.74e+00   -4.74  2.3e-06 ***
## country.codeLSO  1.01e+01   4.31e+00    2.36  0.01860 *
## country.codeLTU -1.02e+01   3.89e+00   -2.63  0.00862 **
## country.codeLUX -1.90e+01   4.55e+00   -4.17  3.2e-05 ***
## country.codeLVA -9.50e+00   3.84e+00   -2.47  0.01353 *
## country.codeMAR -1.49e+01   3.88e+00   -3.85  0.00012 ***
## country.codeMDA -1.26e+01   3.65e+00   -3.45  0.00058 ***
## country.codeMDG  2.20e+01   3.79e+00    5.81  7.5e-09 ***
## country.codeMDV -1.04e+01   4.23e+00   -2.46  0.01417 *
## country.codeMEX -8.81e+00   3.60e+00   -2.45  0.01455 *
## country.codeMHL -8.44e+00   6.65e+00   -1.27  0.20435
## country.codeMKD -6.35e+00   3.80e+00   -1.67  0.09506 .
## country.codeMLI  5.77e+00   4.04e+00    1.43  0.15323
## country.codeMLT -7.51e+00   4.21e+00   -1.78  0.07477 .
## country.codeMMR -2.05e+01   5.02e+00   -4.08  4.7e-05 ***
## country.codeMNE -9.31e+00   3.94e+00   -2.36  0.01827 *
## country.codeMNG -1.24e+01   3.62e+00   -3.43  0.00063 ***
## country.codeMOZ  3.32e+01   4.23e+00    7.85  7.7e-15 ***
## country.codeMRT -7.79e+00   3.75e+00   -2.07  0.03814 *
## country.codeMUS -1.16e+01   4.59e+00   -2.52  0.01166 *
## country.codeMWI  1.96e+01   4.11e+00    4.77  2.0e-06 ***
## country.codeMYS -1.63e+01   3.66e+00   -4.46  8.8e-06 ***
## country.codeNAM  4.84e+00   4.56e+00    1.06  0.28944
## country.codeNER  2.78e+01   3.89e+00    7.15  1.3e-12 ***
## country.codeNGA  7.59e+00   3.76e+00    2.02  0.04389 *
## country.codeNIC -1.26e+01   3.91e+00   -3.21  0.00133 **
## country.codeNLD -1.32e+01   4.08e+00   -3.24  0.00120 **
## country.codeNOR -2.00e+01   4.16e+00   -4.80  1.7e-06 ***
## country.codeNPL -1.63e+01   4.60e+00   -3.54  0.00042 ***
## country.codeNRU -9.91e+00   6.31e+00   -1.57  0.11657
## country.codePAK -8.71e+00   3.54e+00   -2.46  0.01406 *
## country.codePAN -6.79e+00   3.49e+00   -1.94  0.05204 .
## country.codePER -1.06e+01   3.49e+00   -3.04  0.00244 **
## country.codePHL -1.15e+01   3.80e+00   -3.03  0.00249 **
## country.codePNG  2.24e+01   4.99e+00    4.49  7.6e-06 ***
## country.codePOL -1.13e+01   3.75e+00   -3.01  0.00266 **
## country.codePRT -1.48e+01   3.86e+00   -3.82  0.00014 ***
```

```
## country.codePRY -1.71e+01   3.42e+00   -4.98  6.9e-07 ***
## country.codePSE -1.41e+01   3.72e+00   -3.78  0.00016 ***
## country.codeROU -6.29e+00   3.76e+00   -1.67  0.09439 .
## country.codeRUS -1.62e+01   3.58e+00   -4.52  6.8e-06 ***
## country.codeRWA  1.79e+01   4.08e+00    4.39  1.2e-05 ***
## country.codeSDN -3.06e+00   4.95e+00   -0.62  0.53707
## country.codeSEN  1.46e+01   3.85e+00    3.80  0.00015 ***
## country.codeSLB  5.58e+00   4.97e+00    1.12  0.26191
## country.codeSLE  4.24e+00   4.27e+00    0.99  0.32068
## country.codeSLV -1.32e+01   3.50e+00   -3.79  0.00016 ***
## country.codeSOM  3.13e+01   6.25e+00    5.02  5.8e-07 ***
## country.codeSRB -7.87e+00   3.92e+00   -2.01  0.04501 *
## country.codeSSD  1.75e+01   4.97e+00    3.52  0.00045 ***
## country.codeSTP -1.98e-01   4.50e+00   -0.04  0.96489
## country.codeSUR  2.16e+00   6.24e+00    0.35  0.72900
## country.codeSVK -1.02e+01   3.85e+00   -2.65  0.00801 **
## country.codeSVN -1.20e+01   3.87e+00   -3.11  0.00191 **
## country.codeSWE -1.57e+01   3.96e+00   -3.97  7.4e-05 ***
## country.codeSWZ  3.38e+01   4.22e+00    8.01  2.1e-15 ***
## country.codeSYC -8.22e+00   5.08e+00   -1.62  0.10559
## country.codeSYR -1.18e+01   4.96e+00   -2.39  0.01696 *
## country.codeTCD  8.19e+00   4.46e+00    1.83  0.06687 .
## country.codeTGO  1.18e+01   4.21e+00    2.81  0.00496 **
## country.codeTHA -2.02e+01   3.51e+00   -5.75  1.1e-08 ***
## country.codeTJK -1.04e+01   3.91e+00   -2.65  0.00802 **
## country.codeTKM  9.98e+00   6.26e+00    1.59  0.11100
## country.codeTLS -1.49e+01   4.48e+00   -3.32  0.00093 ***
## country.codeTON -1.47e+01   4.50e+00   -3.27  0.00110 **
## country.codeTTO -2.03e+01   5.13e+00   -3.96  7.8e-05 ***
## country.codeTUN -1.29e+01   3.82e+00   -3.37  0.00078 ***
## country.codeTUR -9.88e+00   3.47e+00   -2.85  0.00445 **
## country.codeTUV -1.88e+01   6.46e+00   -2.91  0.00370 **
## country.codeTZA  1.41e+01   4.09e+00    3.45  0.00058 ***
## country.codeUGA  4.78e+00   3.77e+00    1.27  0.20490
## country.codeUKR -1.88e+01   3.58e+00   -5.25  1.7e-07 ***
## country.codeURY -1.35e+01   3.75e+00   -3.61  0.00031 ***
## country.codeUSA -1.74e+01   4.04e+00   -4.30  1.8e-05 ***
## country.codeUZB  3.84e+01   4.24e+00    9.04  < 2e-16 ***
## country.codeVEN -1.02e+01   3.63e+00   -2.80  0.00522 **
## country.codeVNM -2.06e+01   3.68e+00   -5.60  2.5e-08 ***
## country.codeVUT -3.41e+00   4.89e+00   -0.70  0.48589
## country.codeWSM -1.28e+01   4.52e+00   -2.83  0.00475 **
## country.codeXKX -7.62e+00   3.81e+00   -2.00  0.04557 *
## country.codeYEM -1.66e+01   4.44e+00   -3.73  0.00020 ***
## country.codeZAF  8.33e+00   4.00e+00    2.08  0.03735 *
## country.codeZMB  2.17e+01   3.65e+00    5.95  3.3e-09 ***
## country.codeZWE  1.66e+00   4.49e+00    0.37  0.71155
## year            -1.79e-01   3.17e-02   -5.66  1.8e-08 ***
## incomeL         -1.12e+00   1.42e+00   -0.79  0.43054
## incomeLM        -6.02e+00   1.01e+00   -5.93  3.7e-09 ***
## incomeUM        -3.27e+00   7.50e-01   -4.36  1.4e-05 ***
## edu.total       -4.94e-01   1.37e-01   -3.61  0.00031 ***
## hlth             3.32e-01   1.17e-01    2.83  0.00471 **
## mil             -2.22e-01   1.88e-01   -1.18  0.23774
```

```
## fdi             -4.91e-12   3.85e-12   -1.27  0.20295
## lbr.part         2.37e-01   2.61e-02    9.10  < 2e-16 ***
## unemp           -6.30e-02   4.08e-02   -1.54  0.12280
## pop.gwth.total  -3.12e+00   6.54e-01   -4.77  2.0e-06 ***
## pop.gwth.rural   6.81e-01   2.68e-01    2.54  0.01105 *
## pop.gwth.urban   2.37e+00   3.55e-01    6.66  3.8e-11 ***
## gdp.dflt         2.30e-04   1.10e-03    0.21  0.83470
## gcf             -1.87e-01   2.74e-02   -6.81  1.4e-11 ***
## trade           -1.47e-02   8.50e-03   -1.72  0.08506 .
## gdp.pc           3.00e-04   2.35e-05   12.76  < 2e-16 ***
## lfdi            -1.85e-04   5.22e-02    0.00  0.99717
## lgdp.dflt        3.33e-02   1.18e-01    0.28  0.77827
## lgdp.pc         -8.18e+00   5.40e-01  -15.15  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.29 on 1655 degrees of freedom
## Multiple R-squared:  0.918,  Adjusted R-squared:  0.909
## F-statistic: 99.4 on 187 and 1655 DF,  p-value: <2e-16
```

There are also some weak variable, we should perform some variable selection.

B. VIF Our data set contains a lots of variables. We can perform some variable selection to reduce over-fitting.

```
gvif <- lapply(fit1, vif)
gvif
```

```
## [[1]]
##                      GVIF  Df GVIF^(1/(2*Df))
## country.code   6.5973e+08 167          1.0627
## year           5.6783e+00   1          2.3829
## income         1.0859e+02   3          2.1842
## edu.total      3.0739e+00   1          1.7533
## hlth           5.7126e+00   1          2.3901
## mil            4.2146e+00   1          2.0529
## fdi            2.5332e+00   1          1.5916
## lbr.part       3.4724e+00   1          1.8634
## unemp          3.5053e+00   1          1.8723
## pop.gwth.total 3.5572e+01   1          5.9643
## pop.gwth.rural 1.0227e+01   1          3.1980
## pop.gwth.urban 2.2105e+01   1          4.7016
## gdp.dflt       1.2919e+00   1          1.1366
## gcf            2.5453e+00   1          1.5954
## trade          1.1508e+01   1          3.3923
## gdp.pc         1.4562e+01   1          3.8161
## lfdi           2.1600e+00   1          1.4697
## lgdp.dflt      2.2771e+00   1          1.5090
## lgdp.pc        4.4841e+01   1          6.6964
##
## [[2]]
##                      GVIF  Df GVIF^(1/(2*Df))
## country.code   8.2907e+08 167          1.0634
```

```
## year           6.0082e+00   1          2.4512
## income         1.1182e+02   3          2.1949
## edu.total      3.8234e+00   1          1.9553
## hlth           6.0754e+00   1          2.4648
## mil            4.3002e+00   1          2.0737
## fdi            2.4809e+00   1          1.5751
## lbr.part       3.6749e+00   1          1.9170
## unemp          3.8875e+00   1          1.9717
## pop.gwth.total 3.6091e+01   1          6.0076
## pop.gwth.rural 1.0243e+01   1          3.2005
## pop.gwth.urban 2.2287e+01   1          4.7209
## gdp.dflt       1.2841e+00   1          1.1332
## gcf            2.5320e+00   1          1.5912
## trade          1.0483e+01   1          3.2378
## gdp.pc         1.4800e+01   1          3.8471
## lfdi           2.1060e+00   1          1.4512
## lgdp.dflt      2.2579e+00   1          1.5026
## lgdp.pc        4.4022e+01   1          6.6349
##
## [[3]]
##                       GVIF  Df GVIF^(1/(2*Df))
## country.code   1.0873e+09 167          1.0643
## year           5.8451e+00   1          2.4177
## income         1.0927e+02   3          2.1865
## edu.total      2.9561e+00   1          1.7193
## hlth           4.9931e+00   1          2.2345
## mil            4.2748e+00   1          2.0676
## fdi            2.3709e+00   1          1.5398
## lbr.part       3.2992e+00   1          1.8164
## unemp          4.2174e+00   1          2.0536
## pop.gwth.total 3.6524e+01   1          6.0435
## pop.gwth.rural 1.0263e+01   1          3.2037
## pop.gwth.urban 2.2411e+01   1          4.7340
## gdp.dflt       1.3044e+00   1          1.1421
## gcf            2.6585e+00   1          1.6305
## trade          1.0572e+01   1          3.2514
## gdp.pc         1.4302e+01   1          3.7818
## lfdi           7.5258e+00   1          2.7433
## lgdp.dflt      2.2610e+00   1          1.5037
## lgdp.pc        4.6713e+01   1          6.8347
```

Some forum suggested that we should use the standard $GVIF^{1/(2 \cdot df)} < 2$ as equivalent to $GVIF < 4$ to account for high degree of freedom of some variables.

```
# adjusted gvif higher than 3
lapply(gvif, function(g) {
    g[g[, 3] > 3, 3]
})
```

```
## [[1]]
## pop.gwth.total pop.gwth.rural pop.gwth.urban          trade         gdp.pc
##         5.9643         3.1980         4.7016         3.3923         3.8161
##        lgdp.pc
##         6.6964
```

```
##
## [[2]]
## pop.gwth.total pop.gwth.rural pop.gwth.urban          trade         gdp.pc
##         6.0076         3.2005         4.7209         3.2378         3.8471
##        lgdp.pc
##         6.6349
##
## [[3]]
## pop.gwth.total pop.gwth.rural pop.gwth.urban          trade         gdp.pc
##         6.0435         3.2037         4.7340         3.2514         3.7818
##        lgdp.pc
##         6.8347
```

Remove the highest-ranked variables (`pop.gwth.total`, `pop.gwth.urban`, `pop.gwth.rural`, `gdp.pc`, `lgdp.pc`) and rebuild the models.

```
formula.str <- "pov ~ country.code+year+income+edu.total+hlth+mil+fdi+lbr.part+unemp+gdp.dflt+gcf+trade+

# Using single imputed data set
fit1 <- lapply(1:countries1.imputed$m, function(i) lm(as.formula(formula.str),
    data = complete(countries1.imputed, i)))
```

Re-run GVIF analysis

```
gvif <- lapply(fit1, vif)

lapply(gvif, function(g) {
    g[g[, 3] > 2, 3]
})
```

```
## [[1]]
## income   hlth    mil  trade
## 2.0363 2.3674 2.0330 3.2449
##
## [[2]]
## income   hlth    mil  trade
## 2.0546 2.4400 2.0468 3.1082
##
## [[3]]
## income   hlth    mil  unemp  trade   lfdi
## 2.0487 2.2214 2.0398 2.0226 3.1350 2.6337
```

There're still some variable with adjusted GVIF > 2. We are interested in the effects of `hlth` (expenditure in Healthcare), and `mil` (expenditure in Military), so we won't remove those variables. We can remove `trade`.

```
# remove trade
formula.str <- "pov ~ country.code+year+income+edu.total+hlth+mil+fdi+lbr.part+unemp+gdp.dflt+gcf+lfdi+l

# Using single imputed data set
fit1 <- lapply(1:countries1.imputed$m, function(i) lm(as.formula(formula.str),
    data = complete(countries1.imputed, i)))
```

```
# gvif
gvif <- lapply(fit1, vif)

lapply(gvif, function(g) {
    g[g[, 3] > 2, 3]
})
```

```
## [[1]]
## income   hlth    mil
## 2.0285 2.3604 2.0293
##
## [[2]]
## income   hlth    mil
## 2.0494 2.4355 2.0468
##
## [[3]]
## income   hlth    mil  unemp   lfdi
## 2.0448 2.2120 2.0394 2.0147 2.6331
```

The adjusted GVIF are acceptably low.

Coefficients overview.

```
# find coef of variable other than country.code
var <- c("year", "incomeL", "incomeLM", "incomeUM",
    "edu.total", "hlth", "mil", "fdi", "lbr.part",
    "unemp", "gdp.dflt", "gcf", "lfdi", "lgdp.dflt")

res <- do.call(rbind, lapply(fit1, function(f) f$coefficients[var]))
kable(res)
```

year	incomeL	incomeLM	incomeUM	edu.total	hlth	mil	fdi	lbr.part	unemp	gdp.dflt	gcf	lfdi	lgdp.dflt
-0.34856	10.5954	0.08441	-2.4575	-0.66096	0.32845	-0.35281	0	0.27971	-0.08621	0.00217	-0.25952	-0.16084	0.08664
-0.36010	8.7583	-0.39497	-2.4961	-0.80362	0.56099	-0.36051	0	0.26125	-0.05543	0.00202	-0.29136	-0.12780	0.15781
-0.32162	9.5811	-0.50762	-2.5267	-0.87342	0.32653	-0.39554	0	0.18156	-0.06540	0.00101	-0.24237	-0.36766	0.09228

There are several sign-switching in `income` coefficients, indicating the effect of removing some multicollinearity.

```
fit1.summ <- lapply(fit1, function(f) {
    k <- length(f$coefficients) - 1
    pred <- predict(f, countries.test.4)
    data.frame(`Train Adj.R2` = summary(f)$adj.r.squared,
        `Test Adj.R2` = adjR2(pred, countries.test.4$pov,
            k))
})

res <- do.call(rbind.data.frame, fit1.summ)
res <- cbind(data.frame(Set = 1:3), res)
kable(res)
```

Set	Train.Adj.R2	Test.Adj.R2
1	0.88976	0.63284
2	0.88914	0.64535
3	0.88623	0.64498

Directly removing variables seems to be penalising our test results. As our interest is to investigate the effect of certain variables on poverty, the yielded R-Squared is within acceptable range. We can continue variable selection on the new models.

```
formula.str <- "pov ~ country.code+year+income+edu.total+hlth+mil+fdi+lbr.part+unemp+gdp.dflt+gcf+lfdi+l
```

B. Step-wise AIC We can conduct a step-wise AIC variable selection. It is similar to the procedure we use in class, but based on a metric call AIC (Akaike Information Criterion), which is an estimator of prediction error and relative quality of statistical models. The lower AIC is, the better the model fits. \

```
# helper
fitAIC <- function(i) {
    set.seed(1984)
    fit <- lm(as.formula(formula.str), complete(countries1.imputed,
        i))
    aic.fit <- stepAIC(fit, trace = F, direction = "backward")
    return(aic.fit)
}

# build models
aic.fits <- lapply(1:countries1.imputed$m, fitAIC)

# check final terms of each model
lapply(aic.fits, function(mod) formula(mod$terms))
```

```
## [[1]]
## pov ~ country.code + year + income + edu.total + hlth + mil +
##     lbr.part + unemp + gdp.dflt + gcf + lfdi
## <environment: 0x561359857b28>
##
## [[2]]
## pov ~ country.code + year + income + edu.total + hlth + mil +
##     lbr.part + gdp.dflt + gcf + lfdi + lgdp.dflt
## <environment: 0x561356149c48>
##
## [[3]]
## pov ~ country.code + year + income + edu.total + hlth + mil +
##     lbr.part + unemp + gcf + lfdi
## <environment: 0x561350fc5d30>
```

The most commonly removed variables are:

- gdp.dflt
- ldgp.dflt
- unemp

- fdi

```
# predict with each model
aic.pred <- lapply(aic.fits, predict, newdata = countries.test.4)
# calculate adj r2
aic.train.r2 <- unlist(lapply(aic.fits, function(model) summary(model)$adj.r.squared))

k <- lapply(aic.fits, function(m) length(m$coefficients))
aic.test.r2 <- unlist(mapply(adjR2, aic.pred, k = k,
    MoreArgs = list(orig = countries.test.4$pov)))

res <- data.frame(`set no.` = 1:3, train = aic.train.r2,
    test = aic.test.r2)
```

We can compare Adjusted R-Squared to estimate relative over-fitting in the new models.

```
kable(res)
```

set.no.	train	test
1	0.88986	0.63311
2	0.88918	0.64804
3	0.88632	0.64679

Performance on train set and test set are not affected. However, we have simplified the model by removing some unnecessary terms. We can update our models.

```
formula.str <- "pov ~ country.code+year+income+edu.total+hlth+mil+lbr.part+gcf+lfdi"

fit3 <- lapply(1:countries1.imputed$m, function(i) lm(as.formula(formula.str),
    data = complete(countries1.imputed, i)))

# Build models with all generated data set and
# pool the estimates
fit4 <- with(data = countries1.imputed, exp = lm(as.formula(formula.str)))
fit4.combined <- pool(fit4)
# dummy lm model
fit4.dummy <- lm(as.formula(formula.str), data = complete(countries1.imputed,
    1))
# replace coefficients of dummy model & predict
fit4.dummy$coefficients <- fit4.combined$pooled$estimate

fit3.summ <- lapply(fit3, function(f) {
    k <- length(f$coefficients) - 1
    pred <- predict(f, countries.test.4)
    test.resid <- pred - countries.test.4$pov
    data.frame(`Train MSE` = mean(summary(f)$residuals^2),
        `Test MSE` = mean(test.resid^2), `Train MAE` = mean(abs(summary(f)$residuals)),
        `Test MAE` = mean(abs(test.resid)), `Train Adj.R2` = summary(f)$adj.r.squared,
        `Test Adj.R2` = adjR2(pred, countries.test.4$pov,
            k))
})
```

```
k <- length(fit4.dummy$coefficients) - 1
fit4.pred <- predict(fit4.dummy, countries.test.4)
test.resid <- fit4.pred - countries.test.4$pov
fit4.summ <- data.frame(`Train MSE` = mean(summary(fit4.dummy)$residuals^2),
    `Test MSE` = mean(test.resid^2), `Train MAE` = mean(abs(summary(fit4.dummy)$residuals)),
    `Test MAE` = mean(abs(test.resid)), `Train Adj.R2` = pool.r.squared(fit4,
        adjusted = T)[, "est"], `Test Adj.R2` = adjR2(fit4.pred,
        countries.test.4$pov, k))

res <- do.call(rbind.data.frame, fit3.summ)
res <- rbind(res, fit4.summ)
res <- cbind(data.frame(Set = c(1:3, "Pooled")), res)

kable(res)
```

Set	Train.MSE	Test.MSE	Train.MAE	Test.MAE	Train.Adj.R2	Test.Adj.R2
1	30.724	14.845	3.4440	2.3418	0.88942	0.63984
2	30.873	14.396	3.4327	2.3342	0.88888	0.65074
3	31.607	14.430	3.4308	2.2813	0.88624	0.64992
Pooled	30.724	14.479	3.4440	2.3095	0.88819	0.64871

Coefficients overview.

```
var <- c("year", "incomeL", "incomeLM", "incomeUM",
    "edu.total", "hlth", "mil", "lbr.part", "gcf",
    "lfdi")

res <- do.call(rbind, lapply(fit3, function(f) f$coefficients[var]))
kable(res)
```

year	incomeL	incomeLM	incomeUM	edu.total	hlth	mil	lbr.part	gcf	lfdi
-0.35080	10.3710	-0.12820	-2.5993	-0.72961	0.28514	-0.36045	0.28462	-0.23934	-0.16645
-0.36826	8.5566	-0.57113	-2.6243	-0.85308	0.53171	-0.36941	0.26253	-0.27449	-0.13582
-0.32501	9.4070	-0.67532	-2.6324	-0.90330	0.30749	-0.40532	0.18336	-0.22585	-0.39267

We can further treat over-fitting with regularization.

C. Regularization Helper functions to standardise data.

```
sd0 <- function(vct) {
    if (!is.numeric(vct)) {
        return(NA)
    }

    return(sd(vct, na.rm = T))
}
```

```
std0 <- function(vct, scl) {
    if (!is.numeric(vct)) {
        return(vct)
    }

    return(vct/scl)
}

standardise <- function(train, test) {
    # save number of train and combine both data
    # set
    trainCases <- 1:nrow(train)
    data <- rbind(train, test)
    # calc scaler
    scaler <- unlist(lapply(data, sd0))
    # get the numeric columns
    numCols <- which(unlist(lapply(data, is.numeric)))
    # divide numeric columns by scalers, and
    # combine with factor columns
    num <- as.data.frame(mapply(std0, vct = data[,
        numCols], scl = scaler[numCols], SIMPLIFY = T))
    fct <- data[, -numCols]
    final <- cbind(fct, num)
    # split to train and test
    trainSet <- final[trainCases, ]
    testSet <- final[-trainCases, ]

    res <- list(final = final, train = trainSet, test = testSet,
        scaler = scaler)
    return(res)
}
```

Helper function to build the optimal model.

```
optimalModel <- function(formula, train, test) {
    set.seed(1984)
    # standise and save number of train cases (to
    # split later)
    std.data <- standardise(train, test)
    trainCases <- 1:nrow(train)
    # matrix-ise
    xs <- model.matrix(formula, std.data$final)[, -1]
    ys <- std.data$final$pov
    # split data
    train.x <- xs[trainCases, ]
    train.y <- ys[trainCases]
    test.x <- xs[-trainCases, ]
    test.y <- ys[-trainCases]
    # list of tested alpha, resolution = 0.1
    alphas <- seq(0, 1, 0.1)
    # build a bunch of models
    models <- lapply(alphas, function(a) cv.glmnet(train.x,
        train.y, type.measure = "mse", alpha = a))
```

```
    cv.error <- unlist(lapply(models, function(model) model$cvm[model$lambda ==
        model$lambda.min]))
    # best model
    best.model.idx <- which.min(cv.error)
    # optimal alpha and lambda
    alpha.opt <- alphas[best.model.idx]
    lambda.opt <- models[[best.model.idx]]$lambda.min
    best.model <- glmnet(train.x, train.y, alpha = alpha.opt,
        lambda = lambda.opt)
    # predict for test data
    train.fit <- predict(best.model, train.x)
    test.fit <- predict(best.model, test.x)
    # evaluation - train
    k <- best.model$df
    train.r2 <- best.model$dev.ratio
    train.adjR2 <- adjR2(train.fit, train.y, k)
    train.resid <- train.fit - train.y
    train.mse <- mean(train.resid^2)
    train.rmse <- sqrt(train.mse)
    train.mae <- mean(abs(train.resid))
    train.mape <- mean(abs(train.resid/train.y))
    # evaluation - test
    test.r2 <- r2(test.fit, test.y)
    test.adjR2 <- adjR2(test.fit, test.y, k)
    test.resid <- test.fit - test.y
    test.mse <- mean(test.resid^2)
    test.rmse <- sqrt(test.mse)
    test.mae <- mean(abs(test.resid))
    test.mape <- mean(abs(test.resid/test.y))
    # final results
    results <- list(train = list(data = train, r2 = train.r2,
        adj.r2 = train.adjR2, residuals = train.resid,
        mse = train.mse, rmse = train.rmse, mae = train.mae,
        mape = train.mape), test = list(data = test,
        r2 = test.r2, adj.r2 = test.adjR2, residuals = test.resid,
        mse = test.mse, rmse = test.rmse, mae = test.mae,
        mape = test.mape), scaler = std.data$scaler,
        alpha = alpha.opt, lambda = lambda.opt, model = best.model)

    return(results)
}
```

Build the optimal models using each imputated data set.

```
# helper function to build from a specific
# imputated data
buildWith <- function(set, imputed, test, formula,
    fun) {
    train <- complete(imputed, set)
    model <- fun(formula, train, test)
    return(model)
}
```

```
set.seed(1984)

models <- lapply(1:countries1.imputed$m, buildWith,
    imputed = countries1.imputed, test = countries.test.4,
    formula = as.formula(formula.str), fun = optimalModel)
```

Display parameters and evaluation.

```
result.list <- lapply(models, function(mod) {
    data.frame(alpha = mod$alpha, lambda = mod$lambda,
        `Train MSE` = mod$train$mse, `Test MSE` = mod$test$mse,
        `Train MAE` = mod$train$mae, `Test MAE` = mod$test$mae,
        `Train R2` = mod$train$r2, `Test R2` = mod$test$r2,
        `Train Adj.R2` = mod$train$adj.r2, `Test Adj.R2` = mod$test$adj.r2)
})

result.df <- do.call(rbind.data.frame, result.list)
result.df <- cbind(data.frame(Set = 1:countries1.imputed$m),
    result.df)
kable(result.df)
```

Set	alpha	lambda	Train.MSE	Test.MSE	Train.MAE	Test.MAE	Train.R2	Test.R2	Train.Adj.R2	Test.Adj.R2
1	0.5	0.00016	0.12003	0.05762	0.21462	0.14512	0.89950	0.75581	0.88882	0.64024
2	0.9	0.00009	0.12057	0.05587	0.21360	0.14460	0.89904	0.76321	0.88831	0.65115
3	0.5	0.00018	0.12349	0.05605	0.21365	0.14147	0.89660	0.76243	0.88561	0.65000

There are some improvement on the test data performance based on adjusted R-Squared. MSE and MAE remain low on both data sets.

Coefficients overview.

```
var <- c("year", "incomeL", "incomeLM", "incomeUM",
    "edu.total", "hlth", "mil", "lbr.part", "gcf",
    "lfdi")

res <- do.call(rbind, lapply(models, function(f) coefficients(f$model)[var,
    ]))
kable(res)
```

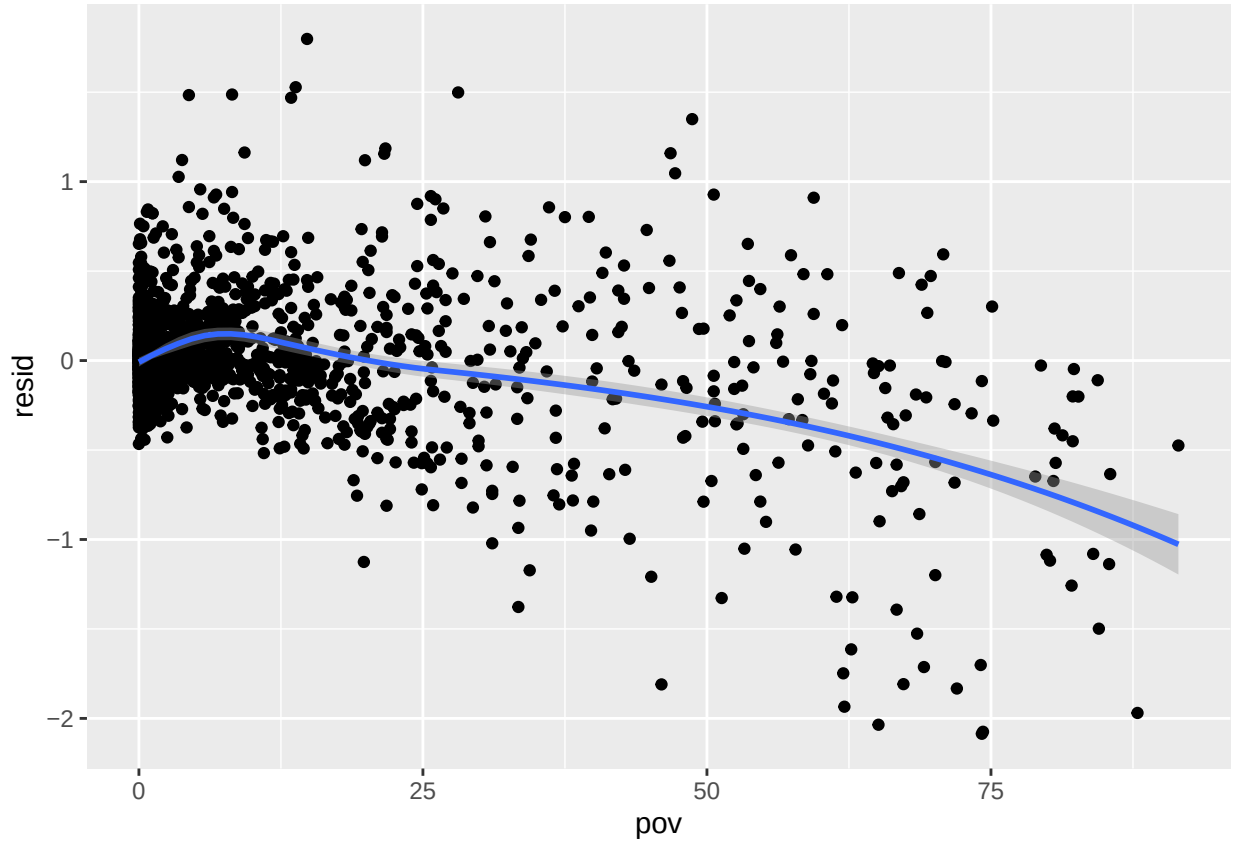
year	incomeL	incomeLM	incomeUM	edu.total	hlth	mil	lbr.part	gcf	lfdi
-0.18233	0.67869	0.01234	-0.15352	-0.07256	0.03431	-0.02238	0.14872	-0.10348	-0.03578
-0.19175	0.57080	-0.01042	-0.15340	-0.08424	0.07337	-0.02313	0.14399	-0.11839	-0.02884
-0.16845	0.62599	-0.01630	-0.15303	-0.08902	0.04033	-0.02585	0.09944	-0.09838	-0.06001

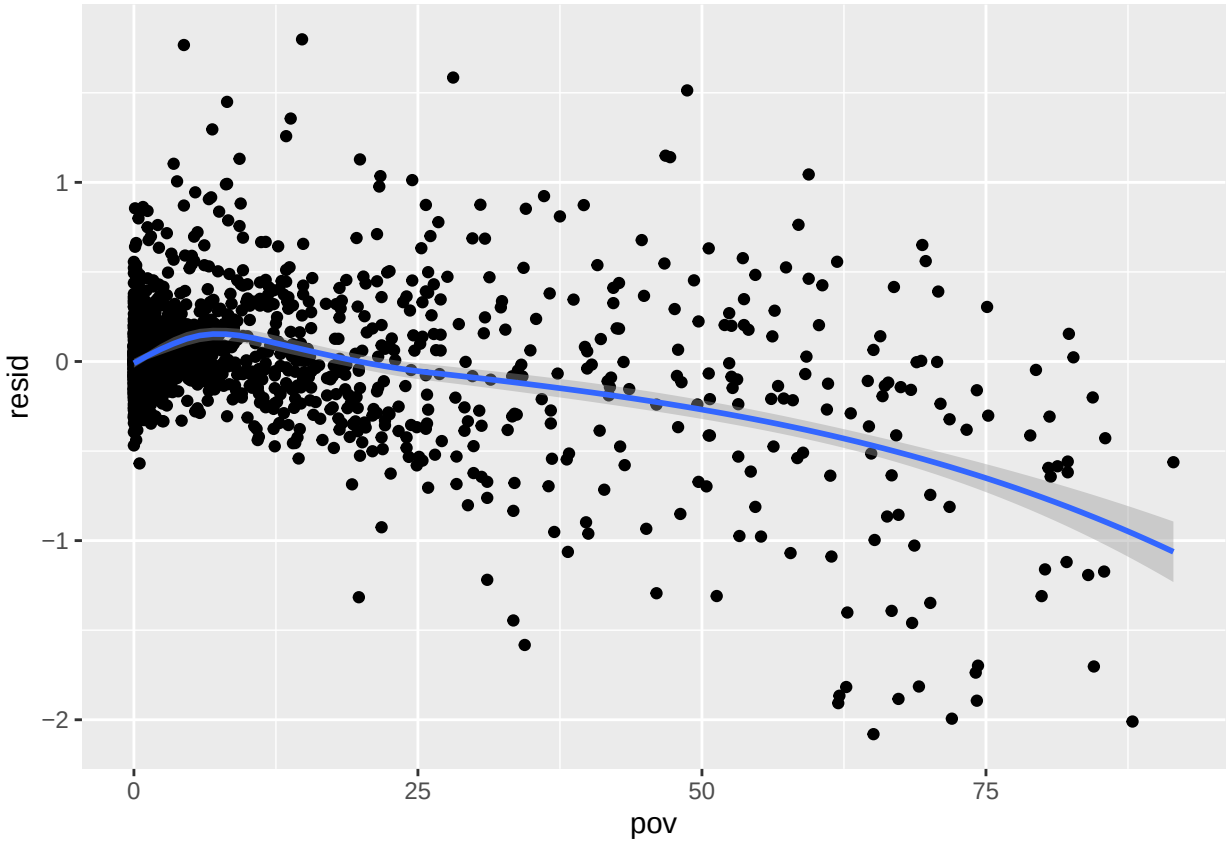
All non-country.code variables are non-zero. We can observe that the regularisation models just remove the effect of `country.code`, for countries that have similar "baseline".

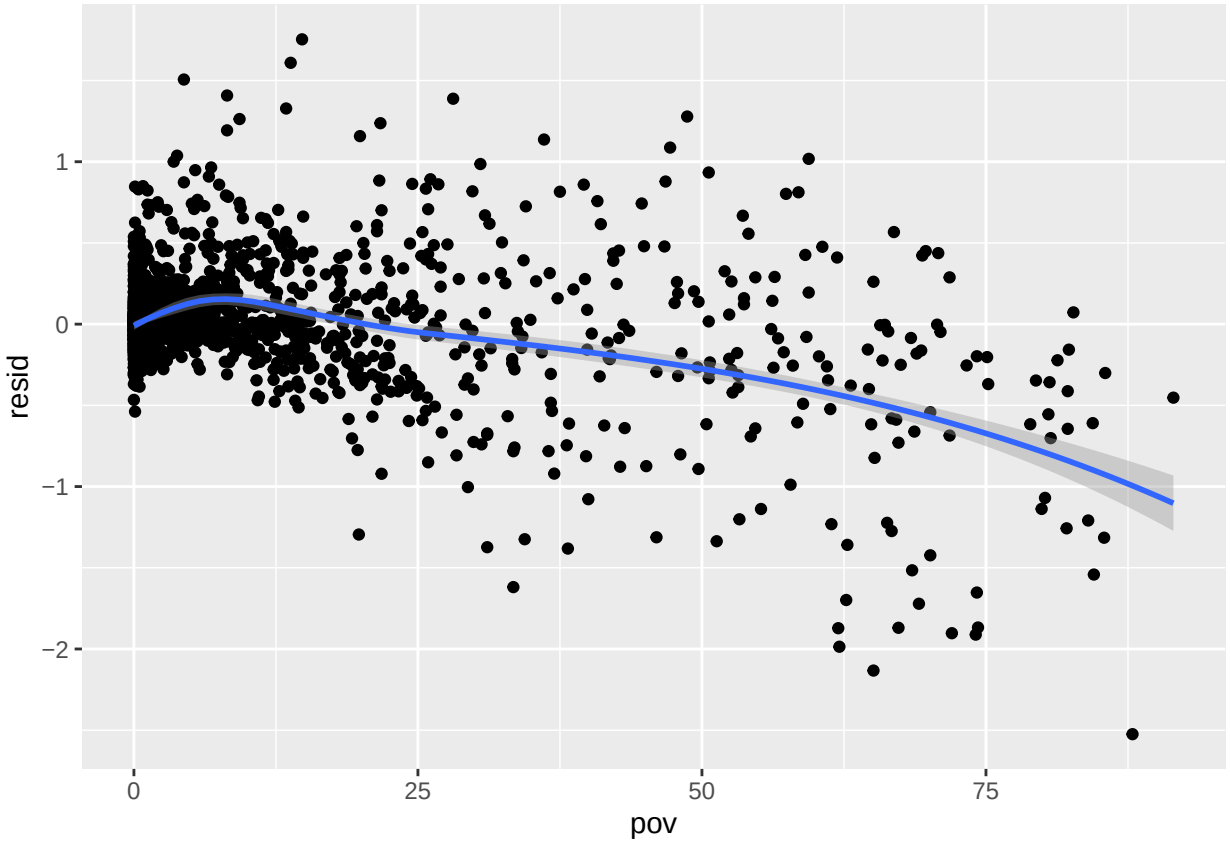
3.2.4.2. Assessment Now we are evaluating some performance metrics, checking assumptions, and remedy some potential problems.

A. Homoscedasticity Plot the residuals ~ pov

```
i <- 1
for (mod in models) {
    resid <- mod$train$residuals
    pov <- mod$train$data$pov
    # plot(resid ~ pov, main = paste('Model', i))
    print(ggplot(mapping = aes(y = resid, x = pov)) +
        geom_point() + geom_smooth(formula = y ~ x,
        method = loess))
    i <- i + 1
}
```




resid
1
0
−1
−2
0
25
50
75
pov



There seems to be some larger variance with higher values of `pov`. We can conduct a statistical test to confirm the present of heteroscedasticity.

The Goldfeld-Quandt test is performed by eliminating a certain number of observations from the dataset's center, then comparing the spread of residuals between the two datasets on either side of the central observations.

The Goldfeld-Quandt test examines two submodels' variances divided by a defined breakpoint and rejects if the variances disagree.

Under H0, the Goldfeld-Quandt test's test statistic follows an F distribution with degrees of freedom as specified in the parameter.

- **Null (H0):** Heteroscedasticity is not present.
- **Alternative (H1):** Heteroscedasticity is present.

```
lapply(models, function(m) {
    resid <- m$train$residuals
    # apply Goldfeld-Quandt test
    gqtest(formula = resid ~ 1, order.by = m$train$data$pov)
})
```

```
## [[1]]
##
##  Goldfeld-Quandt test
##
## data:  resid ~ 1
```

```
## GQ = 7.09, df1 = 921, df2 = 920, p-value <2e-16
## alternative hypothesis: variance increases from segment 1 to 2
##
##
## [[2]]
##
##  Goldfeld-Quandt test
##
## data:  resid ~ 1
## GQ = 7.42, df1 = 921, df2 = 920, p-value <2e-16
## alternative hypothesis: variance increases from segment 1 to 2
##
##
## [[3]]
##
##  Goldfeld-Quandt test
##
## data:  resid ~ 1
## GQ = 7.67, df1 = 921, df2 = 920, p-value <2e-16
## alternative hypothesis: variance increases from segment 1 to 2
```

There is heteroscedasticity in our models. Generalized Least Squares With Unknown For of Variance is a possible remedy of this problem, but at the cost of model interpretation. The high variance in higher end of `pov` can be explained by the relative volatile economics and politic climate in highly poor countries ($pov > 0.25$), leading to unstable effect of predictors.

B. Independence Some possible source of dependency are poverty measured on the same country, or in the same year. These are accounted in our models by controlling those v

3.2.4.3. Interpretation

3.3. Panel Data Analysis (To be done)

4. Conclusion

5. Appendix

6. References

Akbar, Muhammad, Mukaram Khan, Haidar Farooqe, and Kaleemullah. 2019. "Public Spending, Education and Poverty: A Cross Country Analysis" 4 (April): 12–20.

Alruhaymi, Abdullah Z, and Charles J Kim. 2021. "Why Can Multiple Imputations and How (MICE) Algorithm Work?" *Open J. Stat.* 11 (05): 759–77.

Arel-Bundock, Vincent, and Krzysztof J. Pelc. 2018. "When Can Multiple Imputation Improve Regression Estimates?" *Political Analysis* 26 (2): 240–45. https://doi.org/10.1017/pan.2017.43.

Gopalan, Bhuvaneswari. 2020. "Mice Algorithm to Impute Missing Values in a Dataset." *Numpy Ninja*. Numpy Ninja. https://www.numpyninja.com/post/mice-algorithm-to-impute-missing-values-in-a-dataset.