seminararbeit_R.R

User

Sun Jan 13 21:02:24 2019

```
rm(list =ls())
library(dplyr)
```

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

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

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

```
library(RPostgreSQL)
```

```
## Loading required package: DBI
```

```
library(DBI)
library(survey)
```

```
## Loading required package: grid
```

```
## Loading required package: Matrix
```

```
## Loading required package: survival
```

```
##
## Attaching package: 'survey'
```

```
## The following object is masked from 'package:graphics':
##
##     dotchart
```

```
library(convey)
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------- tidy
```

```
## v ggplot2 3.1.0     v readr   1.3.1
## v tibble  1.4.2     v purrr   0.2.5
## v tidyr   0.8.2     v stringr 1.3.1
## v ggplot2 3.1.0     v forcats 0.3.0
```

```
## -- Conflicts ------------------------------------------------------------------------- tidyverse_
## x tidyr::expand() masks Matrix::expand()
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(eurostat)

#Daten abrufen

pg <- src_postgres(dbname="datacube", host="ineq.wu.ac.at", user='lvineq',password = 'palma', options="
```

```
#Daten laden und formatatieren

silc.p <- tbl(pg, 'pp') %>% filter(pb010 %in% c(2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 20
silc.p <- collect(silc.p, n=Inf)

silc.r <- tbl(pg, 'rr') %>% filter(rb010 %in% c(2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 20
silc.r <- collect(silc.r, n=Inf)

silc.d <- tbl(pg, 'dd') %>% filter(db010 %in% c(2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 20
silc.d <- collect(silc.d, n=Inf)

silc.h <- tbl(pg, 'hh') %>% filter(hb010 %in% c(2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 20
silc.h <- collect(silc.h, n=Inf)

#Download der py021g variable

c07p <- tbl(pg, "c07p") %>% filter(pb020 %in% c("NL")) %>% select(pb010, pb030,

c08p <- tbl(pg, "c08p") %>% filter(pb020 %in% c("NL")) %>% select(pb010, pb030,
                                                                  py021g) %>% collect(n = Inf)

c09p <- tbl(pg, "c09p") %>% filter(pb020 %in% c("NL")) %>% select(pb010, pb030,
                                                                  py021g) %>% collect(n = Inf)

c10p <- tbl(pg, "c10p") %>% filter(pb020 %in% c("NL")) %>% select(pb010, pb030,
                                                                  py021g) %>% collect(n = Inf)

c11p <- tbl(pg, "c11p") %>% filter(pb020 %in% c("NL")) %>% select(pb010, pb030,
                                                                  py021g) %>% collect(n = Inf)

c12p <- tbl(pg, "c12p") %>% filter(pb020 %in% c("NL")) %>% select(pb010, pb030,
                                                                  py021g) %>% collect(n = Inf)

c13p <- tbl(pg, "c13p") %>% filter(pb020 %in% c("NL")) %>% select(pb010, pb030,
                                                                  py021g) %>% collect(n = Inf)
cxxp <- bind_rows(c07p, c08p, c09p, c10p, c11p, c12p, c13p)
rm(c07p, c08p, c09p, c10p, c11p, c12p, c13p)

silc.p <- left_join(silc.p, cxxp %>% select(py021g, pb010, pb030))
```

```
## Joining, by = c("pb010", "pb030")
```

```
#Daten 2014-17 werden hinzugefügt hineingeladen ueber cyyp,cyyh, cyyd, cyyr
#cyyp:
c14p <- tbl(pg, "c14p") %>% filter(pb020 %in% c("NL")) %>%
  select(py010g, py021g, py050g, py080g, py090g, py100g, py110g, py120g, py130g, py140g, pb010, pb020, p

c15p <- tbl(pg, "c15p") %>% filter(pb020 %in% c("NL")) %>%
  select(py010g, py021g, py050g, py080g, py090g, py100g, py110g, py120g, py130g, py140g, pb010, pb020, p

c16p <- tbl(pg, "c16p") %>% filter(pb020 %in% c("NL")) %>%
  select(py010g, py021g, py050g, py080g, py090g, py100g, py110g, py120g, py130g, py140g, pb010, pb020, p

c17p <- tbl(pg, "c17p") %>% filter(pb020 %in% c("NL")) %>%
```

```
  select(py010g, py021g, py050g, py080g, py090g, py100g, py110g, py120g, py130g, py140g, pb010, pb020, p

cyyp <- bind_rows(c14p, c15p, c16p, c17p)
rm(c14p, c15p, c16p, c17p)

#hinzufuegen zu silc.p
silc.p <- bind_rows(silc.p, cyyp)

#cyyh:
c14h <- tbl(pg, "c14h") %>%
  filter(hb020 %in% c("NL")) %>%
  select(hb010, hb020, hb030, hy010, hy020, hy040g, hy050g, hy060g, hy070g, hy080g,
         hy090g, hy110g, hy120g, hy130g, hy140g, hx010, hx050) %>%
  collect(n = Inf)

c15h <- tbl(pg, "c15h") %>%
  filter(hb020 %in% c("NL")) %>%
  select(hb010, hb020, hb030, hy010, hy020, hy040g, hy050g, hy060g, hy070g, hy080g,
         hy090g, hy110g, hy120g, hy130g, hy140g, hx010, hx050) %>%
  collect(n = Inf)

c16h <- tbl(pg, "c16h") %>%
  filter(hb020 %in% c("NL")) %>%
  select(hb010, hb020, hb030, hy010, hy020, hy040g, hy050g, hy060g, hy070g, hy080g,
         hy090g, hy110g, hy120g, hy130g, hy140g, hx010, hx050) %>%
  collect(n = Inf)

c17h <- tbl(pg, "c17h") %>%
  filter(hb020 %in% c("NL")) %>%
  select(hb010, hb020, hb030, hy010, hy020, hy040g, hy050g, hy060g, hy070g, hy080g,
         hy090g, hy110g, hy120g, hy130g, hy140g, hx010, hx050) %>%
  collect(n = Inf)

cyyh <- bind_rows(c14h, c15h, c16h, c17h)
rm(c14h, c15h, c16h, c17h)

#zu silc.h hinzufügen
silc.h <- bind_rows(silc.h, cyyh)

#cyyd:

c14d <- tbl(pg, "c14d") %>%
  filter(db020 %in% c("NL")) %>%
  select(db010, db020, db030, db040, db090) %>%
  collect(n = Inf)

c15d <- tbl(pg, "c15d") %>%
  filter(db020 %in% c("NL")) %>%
  select(db010, db020, db030, db040, db090) %>%
  collect(n = Inf)

c16d <- tbl(pg, "c16d") %>%
  filter(db020 %in% c("NL")) %>%
```

```
  select(db010, db020, db030, db040, db090) %>%
  collect(n = Inf)

c17d <- tbl(pg, "c17d") %>%
  filter(db020 %in% c("NL")) %>%
  select(db010, db020, db030, db040, db090) %>%
  collect(n = Inf)

cyyd <- bind_rows(c14d, c15d, c16d, c17d)
rm(c14d, c15d, c16d, c17d)

# merge data to silc.d
silc.d <- bind_rows(silc.d, cyyd)

### 3.4 prepare cyyr:

c14r <- tbl(pg, "c14r") %>%
  filter(rb020 %in% c("NL")) %>%
  select(rb010, rb020, rb030, rb050, rb080, rb090, rx010, rx030, rl010) %>%
  collect(n = Inf)

c15r <- tbl(pg, "c15r") %>%
  filter(rb020 %in% c("NL")) %>%
  select(rb010, rb020, rb030, rb050, rb080, rb090, rx010, rx030, rl010) %>%
  collect(n = Inf)

c16r <- tbl(pg, "c16r") %>%
  filter(rb020 %in% c("NL")) %>%
  select(rb010, rb020, rb030, rb050, rb080, rb090, rx010, rx030, rl010) %>%
  collect(n = Inf)

c17r <- tbl(pg, "c17r") %>%
  filter(rb020 %in% c("NL")) %>%
  select(rb010, rb020, rb030, rb050, rb080, rb090, rx010, rx030, rl010) %>%
  collect(n = Inf)

cyyr <- bind_rows(c14r, c15r, c16r, c17r)
rm(c14r, c15r, c16r, c17r)

#alles zu hinzufügen silc.r
silc.r <- bind_rows(silc.r, cyyr)


#Erstellen der Ids
silc.p <- silc.p %>% mutate(id_h = paste0(pb020, px030),
                            id_p = paste0(pb020, pb030))

silc.h <- silc.h %>% mutate(id_h = paste0(hb020, hb030))

silc.d <- silc.d %>% mutate(id_h = paste0(db020, db030))

silc.r <- silc.r %>% mutate(id_h = paste0(rb020, rx030),
                            id_p = paste0(rb020, rb030))
```

```r
#Mergen
silc.pd <- left_join(silc.p, silc.d %>% select(id_h, db010, db020, db040, db090)
                     , by = c('id_h'='id_h', 'pb010'='db010'))

silc.hd <- left_join(silc.h, silc.d %>% select(id_h, db010, db020, db040, db090)
                     , by = c('id_h' = 'id_h', 'hb010'='db010'))

silc.rp <- left_join(silc.r, silc.p, by= c('id_p' = 'id_p', 'rb010' = 'pb010'))

rm(cxxp, cyyp, cyyh, cyyd, cyyr)

#Variable von autos erstellen

int1 <- seq(2004,2006,1)
int2 <- seq(2007,2017,1)
df1 <- silc.pd %>% filter(pb010 %in% int1)
df2 <- silc.pd %>% filter(pb010 %in% int2)
df1$auto <- df1$py020g
df2$auto <- df2$py021g

silc.pd <- bind_rows(df1,df2)

df3 <- silc.rp %>% filter(rb010 %in% int1)
df4 <- silc.rp %>% filter(rb010 %in% int2)
df3$auto <- df3$py020g
df4$auto <- df4$py021g

silc.rp <- bind_rows(df3,df4)

rm(int1,int2,df1,df2, df3, df4)


#verbinden von P und h dataset
silc.rph <- left_join(silc.rp, silc.hd, by = c('id_h.x' = 'id_h',
                                               'rb010' = 'hb010' ))
#Methode P1 Eurostat
# alle Personen aus dem R file die Na's haben 0 setzen

silc.rph[is.na(silc.rph)] <- 0

#Pre-tax factor income
# summe Einkommen aus Arbeit :p_inc
silc.rph <- silc.rph %>% mutate(p_inc = auto + py010g + py050g + py080g)

#Haushaltseinkommen: h_inc
silc.rph <- silc.rph %>% mutate(h_inc = hy040g + hy080g + hy090g + hy110g)

silc.rph <- silc.rph %>% group_by(rb010, id_h.x) %>% mutate(sum_p_inc = sum(p_inc))

#Pre-tax factor income [income_pti1]
silc.rph <- silc.rph %>% mutate(income_pti1 = (sum_p_inc + h_inc)/hx050)

#Pre-tax national income
```

```r
#Pensionen + Arbeitslosengeld
silc.rph <- silc.rph %>% mutate(benefit = py090g + py100g)
silc.rph <- silc.rph %>% group_by(id_h.x, rb010) %>% mutate(sum_benefit = sum(benefit))


#Pre-tax national income
silc.rph <- silc.rph %>% mutate(income_ptni1 =( income_pti1 +
                                                  sum_benefit/hx050))

#Post-tax disposable income
#summe transfer
silc.rph <- silc.rph %>% mutate(ptransfers = py110g + py120g + py130g + py140g)
silc.rph <- silc.rph %>% mutate(htransfers = hy050g + hy060g + hy070g + hy080g)

silc.rph <- silc.rph %>% group_by(id_h.x, rb010) %>% mutate(sum_ptransfers = sum(ptransfers))
#summe tax
silc.rph <- silc.rph %>% mutate(tax = hy120g + hy130g + hy140g)

#Post-tax disposable income [income_ptdi1]
silc.rph <- silc.rph %>% mutate(income_ptdi1 = income_ptni1 + (ptransfers +
                                                                 htransfers - tax)/hx050)


#P2 (wid.world)
#nur über 20 jährige filtern
silc.rph2 <- silc.rph %>% filter(rx010 >= 20) %>% add_count(id_h.y) %>%
  rename(n_hh = n)

#Pre-tax factor income

silc.rph2 <- silc.rph2 %>% mutate(income_ptfi2 = p_inc + h_inc/n_hh)

#Pre-tax national income

silc.rph2 <- silc.rph2 %>% mutate(income_ptni2 = income_ptfi2 + benefit)


#Post-tax disposable income

silc.rph2 <- silc.rph2 %>% mutate(income_ptdi2 = income_ptni2 + ptransfers +
                                    htransfers/n_hh - tax/n_hh)

#Creating Survey Objects

silc.rph <- silc.rph %>% filter(income_pti1 >0, income_ptni1 > 0, income_ptdi1 > 0)
silc.rph2 <- silc.rph2 %>% filter(income_pti1 >0, income_ptni1 > 0, income_ptdi1 > 0,
                                  income_ptfi2 >0, income_ptni2 > 0, income_ptdi2 > 0)

# Inflation

inf <- get_eurostat("prc_hicp_aind", time_format = "raw")
```

```
## Table prc_hicp_aind cached at C:\Users\User\AppData\Local\Temp\RtmpY9Qarm/eurostat/prc_hicp_aind_raw
```

```r
inf <- inf %>% filter(unit == "INX_A_AVG", coicop == "CP00",
                                  geo == "NL", time %in% 2004:2017) %>%
  select(time, values) %>% arrange(time)

inf$values <- inf$values/100

inf$time <- as.integer(inf$time)
silc.rph <- left_join(silc.rph, inf, by = c('rb010' = 'time'))
silc.rph <- silc.rph %>% mutate(income_pti1 = income_pti1/values,
                                income_ptni1 = income_ptni1/values,
                                income_ptdi1 = income_ptdi1/values)

silc.rph2 <- left_join(silc.rph2, inf, by = c('rb010' = 'time'))
silc.rph2 <- silc.rph2 %>% mutate(income_ptfi2 = income_ptfi2/values,
                                  income_ptni2 = income_ptni2/values,
                                  income_ptdi2 = income_ptdi2/values)


P1.svy <- svydesign(ids =  ~ id_p,
                    strata = ~rb020,
                    weights = ~rb050,
                    data = silc.rph) %>% convey_prep()

P2.svy <- svydesign(ids = ~id_p,
                    strata = ~rb020,
                    weights = ~rb050,
                    data = silc.rph2) %>% convey_prep()


# pre-tax factor income
mean_pti1 <- svyby(~income_pti1, ~rb010, P1.svy, svymean)
#mittelwert muss durch inflation dividiert werden, wurde vorher schon durch 100 dividert
mean_pti1$income_pti1 <- mean_pti1$income_pti1


median_pti1 <- svyby(~income_pti1, ~rb010, P1.svy, svyquantile, quantiles = 0.5, ci = T)
```

```
## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available
```

```
## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available
```

```
#median durch inf div.
median_pti1$income_pti1 <-  median_pti1$income_pti1

gini_pti1 <- svyby(~income_pti1, ~rb010, P1.svy, svygini)
p82_pti1 <- svyby(~income_pti1, ~rb010, P1.svy, svyqsr)
tq10_pti1 <- subset(P1.svy, income_pti1 >= as.numeric(
  svyquantile(~income_pti1, P1.svy, quantile=c(0.9))))
t10_pti1 <- svyby(~income_pti1, ~rb010, tq10_pti1, svytotal)
t10_pti1_t <- svyby(~income_pti1, ~rb010, P1.svy, svytotal)
top10_pti1 <- t10_pti1 / t10_pti1_t
#erstellen einer tabelle pti1
tbl_pti1 <- data.frame(mean_pti1$rb010, mean_pti1$income_pti1, median_pti1$income_pti1,
                       gini_pti1$income_pti1, p82_pti1$income_pti1, top10_pti1$income_pti1)
colnames(tbl_pti1) <- c('Jahr', 'Mittelwert', 'Median', 'Gini', 'P80/20', 'Top10%')
rm(mean_pti1, median_pti1, gini_pti1, p82_pti1, tq10_pti1, t10_pti1, t10_pti1_t, top10_pti1)
#  pre-tax national income
mean_ptni1 <- svyby(~income_ptni1, ~rb010, P1.svy, svymean)
mean_ptni1$income_ptni1 <- mean_ptni1$income_ptni1
median_ptni1 <- svyby(~income_ptni1, ~rb010, P1.svy, svyquantile, quantiles = 0.5, ci = T)
```

```
## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available
```

```r
median_ptni1$income_ptni1 <- median_ptni1$income_ptni1
gini_ptni1 <- svyby(~income_ptni1, ~rb010, P1.svy, svygini)
p82_ptni1 <- svyby(~income_ptni1, ~rb010, P1.svy, svyqsr)
tq10_ptni1 <- subset(P1.svy, income_ptni1 >= as.numeric(
  svyquantile(~income_ptni1, P1.svy, quantile=c(0.9))))
t10_ptni1 <- svyby(~income_ptni1, ~rb010, tq10_ptni1, svytotal)
t10_ptni1_t <- svyby(~income_ptni1, ~rb010, P1.svy, svytotal)
top10_ptni1 <- t10_ptni1 / t10_ptni1_t
#erstellen einer tabelle ptni1
tbl_ptni1 <- data.frame(mean_ptni1$rb010, mean_ptni1$income_ptni1, median_ptni1$income_ptni1,
                        gini_ptni1$income_ptni1, p82_ptni1$income_ptni1, top10_ptni1$income_ptni1)
colnames(tbl_ptni1) <- c('Jahr', 'Mittelwert', 'Median', 'Gini', 'P80/20', 'Top10%')
rm(mean_ptni1, median_ptni1, gini_ptni1, p82_ptni1, tq10_ptni1, t10_ptni1, t10_ptni1_t, top10_ptni1)
#ptdi1
mean_ptdi1 <- svyby(~income_ptdi1, ~rb010, P1.svy, svymean)
mean_ptdi1$income_ptdi1 <- mean_ptdi1$income_ptdi1
median_ptdi1 <- svyby(~income_ptdi1, ~rb010, P1.svy, svyquantile, quantiles = 0.5, ci = T)
```

```
## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available
```

```r
median_ptdi1$income_pti1 <- median_ptdi1$income_ptdi1
gini_ptdi1 <- svyby(~income_ptdi1, ~rb010, P1.svy, svygini)
p82_ptdi1 <- svyby(~income_ptdi1, ~rb010, P1.svy, svyqsr)
tq10_ptdi1 <- subset(P1.svy, income_ptdi1 >= as.numeric(
  svyquantile(~income_ptdi1, P1.svy, quantile=c(0.9))))
t10_ptdi1 <- svyby(~income_ptdi1, ~rb010, tq10_ptdi1, svytotal)
t10_ptdi1_t <- svyby(~income_ptdi1, ~rb010, P1.svy, svytotal)
top10_ptdi1 <- t10_ptdi1 / t10_ptdi1_t
#erstellen einer tabelle ptdi1
tbl_ptdi1 <- data.frame(mean_ptdi1$rb010, mean_ptdi1$income_ptdi1, median_ptdi1$income_ptdi1,
```

```r
                        gini_ptdi1$income_ptdi1, p82_ptdi1$income_ptdi1, top10_ptdi1$income_ptdi1)
colnames(tbl_ptdi1) <- c('Jahr', 'Mittelwert', 'Median', 'Gini', 'P80/20', 'Top10%')
rm(mean_ptdi1, median_ptdi1, gini_ptdi1, p82_ptdi1, tq10_ptdi1, t10_ptdi1, t10_ptdi1_t, top10_ptdi1)
#ptfi2
mean_ptfi2 <- svyby(~income_ptfi2, ~rb010, P2.svy, svymean)
mean_ptfi2$income_ptfi2 <- mean_ptfi2$income_ptfi2
median_ptfi2 <- svyby(~income_ptfi2, ~rb010, P2.svy, svyquantile, quantiles = 0.5, ci = T)
```

```
## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available
```

```r
median_ptfi2$income_ptfi2 <- median_ptfi2$income_ptfi2
gini_ptfi2 <- svyby(~income_ptfi2, ~rb010, P2.svy, svygini)
p82_ptfi2 <- svyby(~income_ptfi2, ~rb010, P2.svy, svyqsr)
tq10_ptfi2 <- subset(P2.svy, income_ptfi2 >= as.numeric(
  svyquantile(~income_ptfi2, P2.svy, quantile=c(0.9))))
t10_ptfi2 <- svyby(~income_ptfi2, ~rb010, tq10_ptfi2, svytotal)
t10_ptfi2_t <- svyby(~income_ptfi2, ~rb010, P2.svy, svytotal)
top10_ptfi2 <- t10_ptfi2 / t10_ptfi2_t
#erstellen einer tabelle ptfi2
tbl_ptfi2 <- data.frame(mean_ptfi2$rb010, mean_ptfi2$income_ptfi2, median_ptfi2$income_ptfi2,
                        gini_ptfi2$income_ptfi2, p82_ptfi2$income_ptfi2, top10_ptfi2$income_ptfi2)
colnames(tbl_ptfi2) <- c('Jahr', 'Mittelwert', 'Median', 'Gini', 'P80/20', 'Top10%')
rm(mean_ptfi2, median_ptfi2, gini_ptfi2, p82_ptfi2, tq10_ptfi2, t10_ptfi2, t10_ptfi2_t, top10_ptfi2)

# pre-tax national income
mean_ptni2 <- svyby(~income_ptni2, ~rb010, P2.svy, svymean)
mean_ptni2$income_ptni2 <- mean_ptni2$income_ptni2
median_ptni2 <- svyby(~income_ptni2, ~rb010, P2.svy, svyquantile, quantiles = 0.5, ci = T)
```

```
## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available
```

```
## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available
```

```
median_ptni2$income_ptni1 <- median_ptni2$income_ptni2
gini_ptni2 <- svyby(~income_ptni2, ~rb010, P2.svy, svygini)
p82_ptni2 <- svyby(~income_ptni2, ~rb010, P2.svy, svyqsr)
tq10_ptni2 <- subset(P2.svy, income_ptni2 >= as.numeric(
  svyquantile(~income_ptni2, P2.svy, quantile=c(0.9))))
t10_ptni2 <- svyby(~income_ptni2, ~rb010, tq10_ptni2, svytotal)
t10_ptni2_t <- svyby(~income_ptni2, ~rb010, P2.svy, svytotal)
top10_ptni2 <- t10_ptni2 / t10_ptni2_t
#erstellen einer tabelle ptni2
tbl_ptni2 <- data.frame(mean_ptni2$rb010, mean_ptni2$income_ptni2, median_ptni2$income_ptni2,
                        gini_ptni2$income_ptni2, p82_ptni2$income_ptni2, top10_ptni2$income_ptni2)
colnames(tbl_ptni2) <- c('Jahr', 'Mittelwert', 'Median', 'Gini', 'P80/20', 'Top10%')
rm(mean_ptni2, median_ptni2, gini_ptni2, p82_ptni2, tq10_ptni2, t10_ptni2, t10_ptni2_t, top10_ptni2)
# post-tax disposable income
mean_ptdi2 <- svyby(~income_ptdi2, ~rb010, P2.svy, svymean)
mean_ptdi2$income_ptdi2 <- mean_ptdi2$income_ptdi2
median_ptdi2 <- svyby(~income_ptdi2, ~rb010, P2.svy, svyquantile, quantiles = 0.5, ci = T)
```

```
## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available
```

```
## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available

## Warning in vcov.svyquantile(X[[i]], ...): Only diagonal of vcov() available
```

```r
median_ptdi2$income_ptdi2 <- median_ptdi2$income_ptdi2
gini_ptdi2 <- svyby(~income_ptdi2, ~rb010, P2.svy, svygini)
p82_ptdi2 <- svyby(~income_ptdi2, ~rb010, P2.svy, svyqsr)
tq10_ptdi2 <- subset(P2.svy, income_ptdi2 >= as.numeric(
  svyquantile(~income_ptdi2, P2.svy, quantile=c(0.9))))
t10_ptdi2 <- svyby(~income_ptdi2, ~rb010, tq10_ptdi2, svytotal)
t10_ptdi2_t <- svyby(~income_ptdi2, ~rb010, P2.svy, svytotal)
top10_ptdi2 <- t10_ptdi2 / t10_ptdi2_t
#erstellen einer tabelle ptdi2
tbl_ptdi2 <- data.frame(mean_ptdi2$rb010, mean_ptdi2$income_ptdi2, median_ptdi2$income_ptdi2,
                        gini_ptdi2$income_ptdi2, p82_ptdi2$income_ptdi2, top10_ptdi2$income_ptdi2)
colnames(tbl_ptdi2) <- c('Jahr', 'Mittelwert', 'Median', 'Gini', 'P80/20', 'Top10%')
rm(mean_ptdi2, median_ptdi2, gini_ptdi2, p82_ptdi2, tq10_ptdi2, t10_ptdi2, t10_ptdi2_t, top10_ptdi2)
```