# Trabajo final Muestreo II

Fiorella Lúngaro, Emanuelle Marsella y Maximiliano Saldaña

Diciembre 2021

# Parte 1

```
# Carga la muestra
muestra <- read_xlsx("datos/muestra grupo 2.xlsx")

# Convertir las variables categóricas a su formato correspondiente

muestra <- muestra %>%
  mutate(across(where(is.double) & !c(ingreso, w0, edad, R), as.factor))

muestra <- muestra %>%
  mutate(
    edad = cut(edad, breaks=c(0, 14,20,25,30,40,50,60,Inf), right = FALSE)
    )
```

Se calculan las estimaciones puntuales de la tasa de desempleo, la proporción de personas pobres y del ingreso promedio, haciendo uso de los ponderadores originales $w_0$, es decir, sin ajustar por no respuesta. Esta estrategia de cómputo resulta correcta si el esquema de no respuesta que se considera es *Missing Completely at Random* (MCAR), bajo el cual la probabilidad de responder no depende de las variables de interés ni auxiliares y todas las unidad del marco tienen la misma probabilidad de responder (Ferreira y Zoppolo, 2017).

```
# Diseño usando los ponderadores originales, MCAR.
##FPC?
design1 <- muestra %>%
  filter(R==1) %>%
  as_survey_design(ids = id_hogar, strata = estrato, weights = w0)
```

```
## Tasa de desempleo (desempleados/activos)

#REVISAR ESTO
#Se piensa como un problema de estimación en dominios, nos interesan los desempleados considerando el g
design1 %>%
  filter(activo == 1) %>%
  group_by(desocupado) %>%
  summarise(tasa_desempleo = survey_mean(deff = TRUE, vartype = c('se','cv')))
```

```
## # A tibble: 2 x 5
##   desocupado tasa_desempleo tasa_desempleo_se tasa_desempleo_cv tasa_desempleo_~
##   <fct>               <dbl>             <dbl>             <dbl>            <dbl>
## 1 0                   0.918           0.00330           0.00360             1.07
## 2 1                   0.0824          0.00330           0.0400              1.07
```

La estimación puntual de la proporción de desempleados es 0,0824; mientras que el error estándar (la medida que empleamos para medir la variación del estimador entre muestra y muestra) es 0,0033. Otra medida de la calidad de un estimador $\hat{\theta}$ es su coeficiente de variación, que mide su dispersi'on relativa. Se define como (Zoppollo, x):

$$CV(\hat{\theta}) = \frac{\sqrt{\hat{V}(\hat{\theta})}}{\left|E(\hat{\theta})\right|}$$

Y en el caso del estimador de la proporción de desempleados su estimación es 0,04. El efecto diseño es una medida que permite comparar la eficiencia en términos de variabilidad del estimador para el diseño utilizado, respecto al diseño aleatorio simple sin reposición que. Siendo $p(s)$ el diseño medible considerado, se define como:

$$Deff(p(s), \hat{\theta}) = \frac{V_{p(s)}(\hat{\theta})}{V_{SI}(\hat{\theta})}$$

En el caso del estimador de la proporción de desempleados su valor es 1,07; lo que indica que en este caso el diseño SI es un 7 % más eficiente que el empleado.

```
## Proporción de personas pobres
design1 %>%
  group_by(pobreza) %>%
  summarise(prop_pobres = survey_mean(deff = TRUE, vartype = c('se','cv')))
```

```
## # A tibble: 2 x 5
##   pobreza prop_pobres prop_pobres_se prop_pobres_cv prop_pobres_deff
##   <fct>         <dbl>          <dbl>          <dbl>            <dbl>
## 1 0             0.919        0.00377        0.00410             2.84
## 2 1             0.0811       0.00377        0.0465              2.84
```

En cuanto a la proporción de personas pobres, la estimación puntual es de 0,0811. El error estándar se estima que es 0,004 aproximadamente, mientras que el coeficiente de variación se estima que es 0,05 aproximadamente. La estimación del efecto diseño es 2,84; un elevado valor que indica que el diseño empleado es altamente ineficiente en comparación con el SI, en particular casi tres veces más.

```
## Ingreso promedio
design1 %>%
  summarise(ingreso_prom = survey_mean(ingreso, deff = TRUE, vartype = c('se','cv')))
```

```
##   ingreso_prom ingreso_prom_se ingreso_prom_cv ingreso_prom_deff
## 1     21798.64        239.5809      0.01099063         0.9354982
```

La estimación puntual del ingreso promedio es 21799, siendo la estimación de su error estándar 240. Por otro lado, el coeficiente de variación toma el valor 0,011. La estimación del efecto diseño es 0,94 aproximadamente, por lo que en este caso el diseño empelado resulta más eficiente que el SI, un 6 % más.

```
muestra %>%
  summarise(
    # tasa de no respuesta no ponderada
    nr_np = 1 - mean(R),
    # tasa de no respuesta ponderada
    nr_p = 1 - weighted.mean(R, w0)
  )
```

```
## # A tibble: 1 x 2
##   nr_np  nr_p
##   <dbl> <dbl>
## 1 0.474 0.476
```

```r
summary(muestra$w0)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   104.4   110.8   124.8   125.6   140.3   162.0
```

```r
#considerando por estratos

muestra %>%
  group_by(estrato) %>%
  summarise(
    # tasa de no respuesta no ponderada
    nr_np = mean(R),
    # tasa de no respuesta ponderada
    nr_p = weighted.mean(R, w0)
    )
```

```
## # A tibble: 12 x 3
##    estrato nr_np  nr_p
##    <fct>   <dbl> <dbl>
##  1 1       0.455 0.455
##  2 2       0.530 0.530
##  3 3       0.533 0.533
##  4 4       0.547 0.547
##  5 5       0.538 0.538
##  6 6       0.487 0.487
##  7 7       0.543 0.543
##  8 8       0.543 0.543
##  9 9       0.526 0.526
## 10 10      0.508 0.508
## 11 11      0.552 0.552
## 12 12      0.564 0.564
```

```r
#considerando por departamento

muestra %>%
  group_by(dpto) %>%
  summarise(
   # tasa de no respuesta no ponderada
    nr_np = mean(R),
    # tasa de no respuesta ponderada
    nr_p = weighted.mean(R, w0)
    )
```

```
## # A tibble: 19 x 3
##    dpto  nr_np  nr_p
##    <fct> <dbl> <dbl>
##  1 1     0.523 0.521
##  2 2     0.545 0.545
##  3 3     0.515 0.512
##  4 4     0.542 0.542
##  5 5     0.511 0.511
##  6 6     0.570 0.570
```

```
##  7 7     0.552 0.552
##  8 8     0.576 0.576
##  9 9     0.556 0.556
## 10 10    0.534 0.534
## 11 11    0.517 0.517
## 12 12    0.498 0.498
## 13 13    0.535 0.535
## 14 14    0.533 0.533
## 15 15    0.545 0.545
## 16 16    0.498 0.494
## 17 17    0.499 0.499
## 18 18    0.540 0.540
## 19 19    0.556 0.556
```

```
#considerando por departamento

muestra %>%
  group_by(edad) %>%
  summarise(
   # tasa de no respuesta no ponderada
    nr_np = mean(R),
    # tasa de no respuesta ponderada
    nr_p = weighted.mean(R, w0)
    )
```

```
## # A tibble: 8 x 3
##   edad     nr_np  nr_p
##   <fct>    <dbl> <dbl>
## 1 [0,14)   0.499 0.486
## 2 [14,20)  0.576 0.566
## 3 [20,25)  0.574 0.567
## 4 [25,30)  0.549 0.549
## 5 [30,40)  0.541 0.541
## 6 [40,50)  0.542 0.547
## 7 [50,60)  0.516 0.521
## 8 [60,Inf) 0.499 0.500
```

La tasa de no respuesta no ponderada es del 47,4 % mientras que la ponderada es del 47,6 %. El hecho de que ambas tasas de no respuesta sean similares puede deberse a que los pesos $w_0$ no son muy disímiles entre sí, siendo su mínimo 104,4; su media 125.6 y su máximo 162. Al considerar la proporción de no respondentes por estrato se puede apreciar que ocurre lo mismo. En este caso se puede observar que la tasa de no respuesta varía según el estrato considerado, siendo el primero (Montevideo bajo) el que cuenta con la mayor tasa, del 56 %, y el doceavo el que cuenta con la menor, del 46 %. Estas diferencias se ven reflejadas también al considerar la tasa de no respondentes por departamento. Resultan bastante diferentes las tasas de no respuesta considerando los distintos segmentos de edad, el de 0 a 14 años y el de 60 en adelante son los que presentan menor tasa de no respuesta, siendo las de los primeros 50 % (no ponderada) y 49 % (ponderada) y 50 % (no ponderada y ponderada) la de los segundos.

# Parte 2

## Ajuste por no respuesta por medio de post-estratos de no respuesta

Bajo el enfoque de no respuesta considerado, el MAR (*Missing at Random*), se trabaja bajo el supuesto de que la no respuesta no depende de las variables de interés, pero sí es completamente explicada por variables auxiliares. Lo que se puede hacer en este caso es construir un modelo de respuesta basado en la información auxiliar (Ferreira y Zoppolo, ).

Siguiendo este enfoque, una manera de realizar el ajuste es mediante clases de no respuesta, creadas en base a información de las unidades presente en el marco muestral. Se crean $g$ clases y se asume que todas las unidades dentro de cada una de ellas tiene la misma probabilidad de responder, cuya fórmula es:

$$\hat{\phi}_{i,g} = TR_w = \frac{\sum_{i \in R} w_i}{\sum_{i \in s} w_i}, \quad i \in g$$

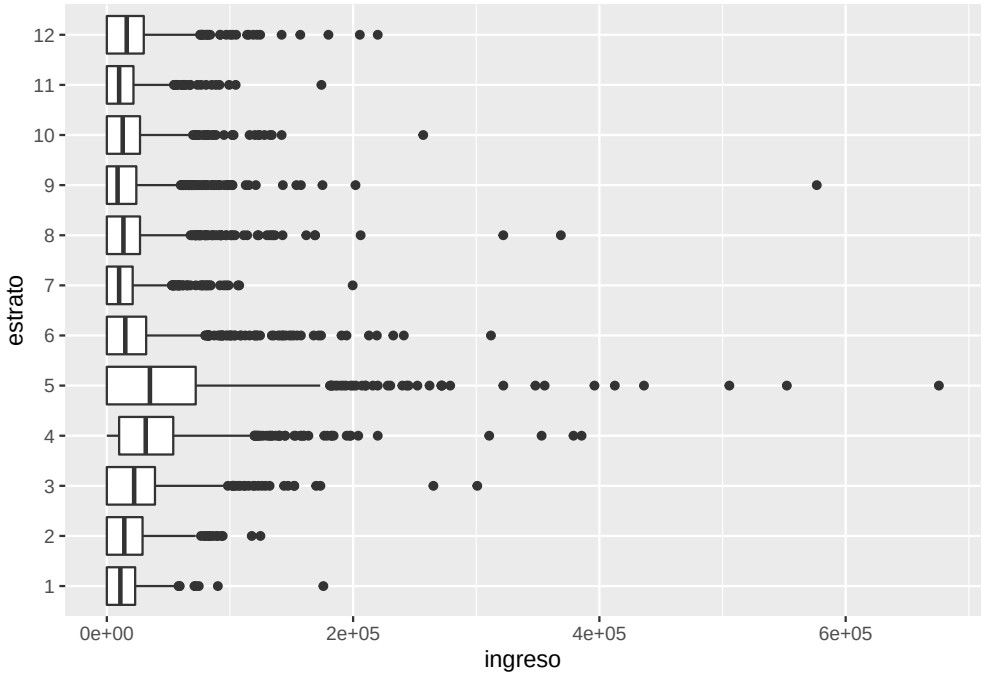
Luego, los ponderadores por no respuesta son:

$$w_i^{nr} = \frac{1}{\pi_i \times \hat{\phi}_{i,g}}$$

($\pi_i$ son las probabilidad de inclusión originales)

En nuestro caso, la información que podríamos emplear que se encuentra en el marco son los estratos.

```
ggplot(muestra, aes(x=ingreso, y=estrato))+ geom_boxplot()
```



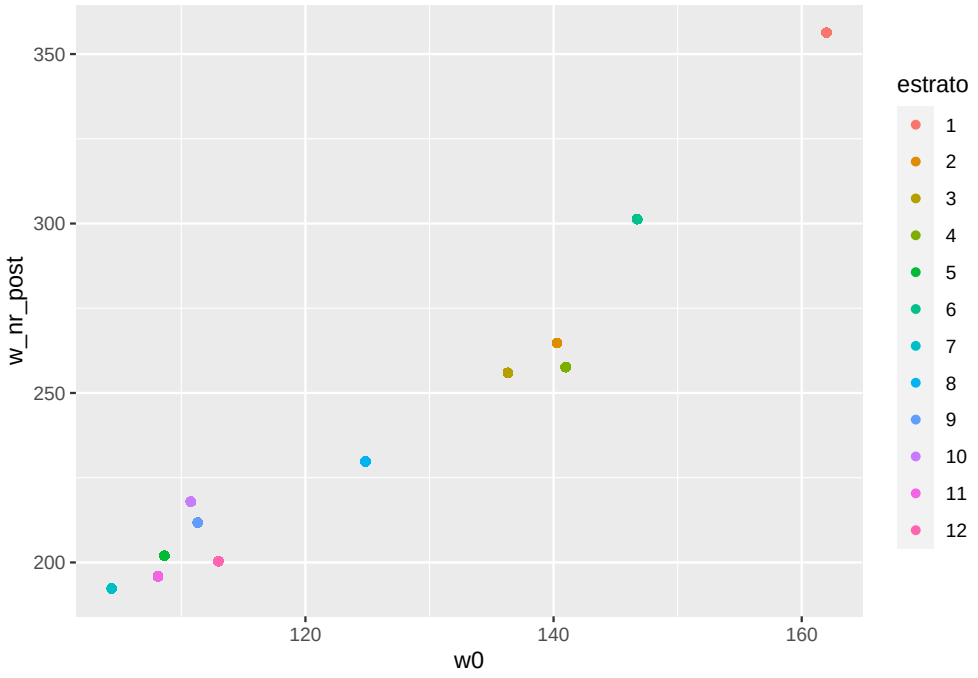

```
# Parecería que el estrato permitiría explicar mejor el ingreso
```

```
ajuste_nr_estrato <-  muestra %>%
  group_by(estrato) %>%
  summarise(
    tr = mean(R),
    tr_w = weighted.mean(R, w0))


muestra <- left_join(muestra , select(ajuste_nr_estrato, estrato, tr_w)) %>%
               mutate(w_nr_post = w0/tr_w)
```

```
## Joining, by = "estrato"
```

```
ggplot(muestra) +
 geom_point(aes(w0, w_nr_post, color = estrato))
```




```
# los ponderadores originales se ven bastante alterados
```

Una medida global frecuentemente utilizada conocida como efecto diseño debido a la ponderación o efecto de Kish (Kish, 1965, 1992) y representa el incremento en la variabilidad de los estimadores por usar ponderadores distintos respecto al uso de el mismo ponderador para todos los caso y se define como uno más la varianza relativa de los ponderadores,

Una forma de medir la variabilidad global de los ponderadores (que en caso de ser alta puede resultar en una variabilidad alta de los estimadores) es el efecto diseño de Kish, que representa el incremento en la variabilidad de los estimadores causada por usar ponderadores distintos para las unidades de la muestra con respecto a usar el mismo ponderador. Su fórmula es:

$$deff_w = 1 + \frac{1}{n}\frac{\sum_s (w_k - \bar{w})^2}{\bar{w}^2}$$

donde $\bar{w} = n^{-1} \sum_s w_k$ es el promedio de los ponderadores.

La práctica usual es calcularlo luego de realizar cada ajuste a los estimadores (no respuesta, calibración, en el caso de este trabajo). Se usa la regla empírica que $deff_w > 1,5$ indican que hay valores extremos de los ponderadores que repercuten en los finales.

```
# Efecto diseño de Kish
deffK(muestra$w_nr_post)
```

```
## [1] 1.033431
```

```
# Ver si solo se calcula con los respondentes
deffK(muestra %>%
        filter(R ==1) %>%
        select(w_nr_post) %>%
        pull())
```

```
## [1] 1.031808
```

En nuestro caso, luego de realizar el ajuste por no respuesta $deff_w \simeq 1,033 < 1,5$, por lo que parecería que no resultó en valores extremos de los ponderadores que repercutirían en la variabilidad de los estimadores.

```
#estimaciones empleando ajuste por no respuesta mediante clases de no respuesta

design2 <- muestra %>%
  filter(R==1) %>%
  as_survey_design(ids = id_hogar, strata = estrato, weights = w_nr_post)

# Tasa de desempleo
design2 %>%
  filter(activo == 1) %>%
  group_by(desocupado) %>%
  summarise(tasa_desempleo = survey_mean(deff = TRUE, vartype = c('se','cv')))
```

```
## # A tibble: 2 x 5
##   desocupado tasa_desempleo tasa_desempleo_se tasa_desempleo_cv tasa_desempleo_~
##   <fct>               <dbl>             <dbl>             <dbl>            <dbl>
## 1 0                 0.917           0.00336           0.00366             1.09
## 2 1                 0.0831          0.00336           0.0404              1.09
```

Una vez se realiza el ajuste por no respuesta mediante clases de no respuesta, la estimación puntual de la tasa de desempleo cambia, pasa de 8,24 % a 8,31 %. El desvío disminuye, pasando de 0,0033 a 0,00336; lo mismo ocurre con el coeficiente de variación, pasando de 0,04 a 0,0404. El efecto diseño pasa de 1,07 a 1,09.

```
## Proporción de personas pobres
design2 %>%
  group_by(pobreza) %>%
  summarise(prop_pobres = survey_mean(deff = TRUE, vartype = c('se','cv')))
```

```
## # A tibble: 2 x 5
##   pobreza prop_pobres prop_pobres_se prop_pobres_cv prop_pobres_deff
##   <fct>         <dbl>          <dbl>          <dbl>            <dbl>
## 1 0             0.916        0.00389        0.00424             2.92
## 2 1             0.0837       0.00389        0.0465              2.92
```

En cuanto a la estimación de la proporción de personas pobres, ahora aumenta de 0,0811 a 0,0837. La estimación del error estándar aumenta a 0,00389, mientras que el coeficiente de vairación no presenta cambio al corregir por no respuesta. El efecto diseño presenta un aumento, de 2,84 a 2,92.

```
## Ingreso promedio
design2 %>%
  summarise(ingreso_prom = survey_mean(ingreso, deff = TRUE, vartype = c('se','cv')))
```

```
##   ingreso_prom ingreso_prom_se ingreso_prom_cv ingreso_prom_deff
## 1     21686.43        238.2187      0.01098469         0.9313662
```

La estimación puntual del ingreso promedio disminuye, de 21799 a 21686, el error estándar presenta una disminución en una unidad. El coeficiente de variación con y sin ajuste son iguales hasta 4 lugares después de la coma, mientras que el efecto diseño presenta una leve disminución; pasa de 0,9355 a 0,9314.

## Estimación de propensiones simples de responder utilizando el algoritmo random forest.

El ajuste por propensiones simples consiste en:

$$w_i^{nr} = \frac{1}{\pi_i \times \hat{\phi}_i}$$

donde $\hat{\phi}_i$ es la propensión a responder de la unidad $i$, la cual se estima a partir de un modelo o un algoritmo, haciendo uso de variables auxiliares conocidas tanto para respondentes como no respondentes.

El algoritmo elegido es *Random Forest* (RF), un método no paramétrico mediante el cual se hace uso de múltiples árboles de decisión para obtener una estimación de la propensión a responder de cada individuo a partir de alguna medida de resumen de la clasificación que hacen los árboles (por ejemplo el modo o la media).

```
# modelamos la no respuesta con random forest
modelo_rf <-  rand_forest(trees = 100) %>%
  set_engine("ranger") %>%
  set_mode("classification") %>%
  fit(as.factor(R) ~ estrato + sexo + edad + dpto, data = muestra)
```

```
# Para ver que tan bien predice el algoritmo
pred_rf <- tibble(predict(modelo_rf, muestra, type= "prob"), predict(modelo_rf, muestra) )

conf_mat(data = bind_cols(select(muestra, R), select(pred_rf, .pred_class)),
         truth = R,
         estimate = .pred_class)
```

```
##           Truth
## Prediction    0    1
##          0 7645 4949
##          1 5616 9794
```

El modelo predice correctamente al 57 % de los no respondentes y al 67 % de los respondentes.

```r
# Agregamos las propensiones estimadas con random forest a la muestra
pred_rf <-  pred_rf %>% rename(prop_rf = .pred_1)

muestra <-  muestra %>% bind_cols(select(pred_rf, prop_rf))

# Calculamos los ponderadores ajustados por no respuesta usando las propensiones de arriba
muestra <-  muestra %>%
  mutate(w_nr_rf = w0/prop_rf)
```

```r
# Calculamos el efecto diseño de Kish
deffK(muestra$w_nr_rf)
```

```
## [1] 1.08304
```

```r
### VER SI SE CALCULA SOLO CON RESPONDENTES
deffK(muestra %>%
        filter(R ==1) %>%
        select(w_nr_rf) %>%
        pull())
```

```
## [1] 1.076543
```

Una vez ajustados los ponderadores por no respuesta por propensiones simples empleando las estimaciones obtenidas mediante *Random Forest*, el efecto diseño de Kish es de 1,074, que si bien es menor a 1,5, es mayor al valor obtenido luego del ajuste anterior.

```r
# Estimaciones empleando ajuste por no respuesta mediante propensiones simples estimadas por random for

design3 <- muestra %>%
  filter(R==1) %>%
  as_survey_design(ids = id_hogar, strata = estrato, weights = w_nr_rf)

## Tasa de desempleo
design3 %>%
  filter(activo == 1) %>%
  group_by(desocupado) %>%
  summarise(tasa_desempleo = survey_mean(deff = TRUE, vartype = c('se','cv')))
```

```
## # A tibble: 2 x 5
##   desocupado tasa_desempleo tasa_desempleo_se tasa_desempleo_cv tasa_desempleo_~
##   <fct>              <dbl>             <dbl>             <dbl>            <dbl>
## 1 0                  0.918           0.00336           0.00366             1.12
## 2 1                  0.0816          0.00336           0.0412              1.12
```

La estimación de la tasa de desempleo al ajustar los ponderadores mediante propensiones simples estimadas por RF es del 8,2 % aproximadamente, disminuyendo en comparación a la estimación realizada con los ponderadores ajustados mediante clases de no respuesta. Se destaca además un aumento del efecto diseño, que ahora es de 1,12 en comparación al de 1,09 del ajuste anterior.

```r
## Proporción de personas pobres
design3 %>%
  group_by(pobreza) %>%
  summarise(prop_pobres = survey_mean(deff = TRUE, vartype = c('se','cv')))
```

```
## # A tibble: 2 x 5
##   pobreza prop_pobres prop_pobres_se prop_pobres_cv prop_pobres_deff
##   <fct>         <dbl>          <dbl>          <dbl>            <dbl>
## 1 0             0.917        0.00392        0.00427             2.98
## 2 1             0.0832       0.00392        0.0471              2.98
```

Considerando ambos ajuste por no respuesta realizados hasta el momento, la estimación de la proporción de personas pobres resulta muy similar, siendo la primera realizada 0,0837 y 0,0834 con el último ajuste considerado. En este caso el efecto diseño cuando se ajusta con propensiones simples y estimadas por RF se eleva a 2,99, en comparación al valor del ajuste anterior; 2,92.

```r
## Ingreso promedio
design3 %>%
  summarise(ingreso_prom = survey_mean(ingreso, deff = TRUE, vartype = c('se','cv')))
```

```
##   ingreso_prom ingreso_prom_se ingreso_prom_cv ingreso_prom_deff
## 1     21513.57        237.0361      0.01101798         0.9486114
```

La estimación puntual del ingreso promedio considerando el último ajuste realizado es 21491, en comparación a 21686 de la estimación bajo el ajuste por clases de no respuesta. El efecto diseño sigue siendo menor a 1, pero aumenta a 0,947 en el ajuste mediante propensiones simples estimadas por RF. Por otro lado, el desvío estimado disminuye en el caso del último ajuste, siendo de 236,5 en comparación a 238,2.

### Ajuste por no respuesta creando clases de no respuesta, utilizando las propensiones estimadas en el punto anterior.

Este tipo de ajuste tiene la intención de hacer más estables las estimaciones finales de la respuesta. Para realizarlo se forman grupos de unidades de la muestra en base a las propensiones estimadas, en nuestro caso creamos clases en base a los quintiles de las propensiones. Una vez se cuenta con los grupos, se resumen los valores de las propensiones dentro de los mismo para contar con un valor representante dentro de la clase, con la mediana o la media (se opta por esta última).

```r
quintiles_phi <- quantile(muestra$prop_rf, c(0.2, 0.4, 0.6 , 0.8, 1))

muestra <- muestra %>%
  mutate(
    clase_nr_rf = case_when(
      prop_rf <=  quintiles_phi[1] ~ 1,
      prop_rf > quintiles_phi[1] & prop_rf <= quintiles_phi[2] ~ 2,
      prop_rf > quintiles_phi[2] & prop_rf <= quintiles_phi[3] ~ 3,
      prop_rf > quintiles_phi[3] & prop_rf <= quintiles_phi[4] ~ 4,
      prop_rf > quintiles_phi[4] ~ 5,
    ) %>%  as.factor()
  )

post_estratos_rf <- muestra %>%
```

```
  group_by(clase_nr_rf) %>%
  summarise(prop_clase_rf = mean(prop_rf))

muestra <- muestra %>%
  mutate(
    prop_clase_rf =  case_when(
      clase_nr_rf == 1 ~ post_estratos_rf$prop_clase_rf[1],
      clase_nr_rf == 2 ~ post_estratos_rf$prop_clase_rf[2],
      clase_nr_rf == 3 ~ post_estratos_rf$prop_clase_rf[3],
      clase_nr_rf == 4 ~ post_estratos_rf$prop_clase_rf[4],
      clase_nr_rf == 5 ~ post_estratos_rf$prop_clase_rf[5]
    ),
  w_post_nr_rf = w0/prop_clase_rf
  )
```

```
# Efecto diseño de Kish
deffK(muestra %>%
        filter(R ==1) %>%
        select(w_post_nr_rf) %>%
        pull())
```

```
## [1] 1.070663
```

El efecto diseño en este caso es de 1,067, con lo que al ser menor a 1,5 esto indicaría que el ajuste realizado parece no haber generado valores extremos de los ponderadores. Este valor es menor al obtenido en el ajuste por propensiones simples, pero mayor que el ajuste por clases de no respuesta.