Trabajo final

Matías Bajac-Aris Sarkisian

1-12-2023

Introducción

El proposito de este proyecto es utilizar las herramientas vistas en el curso , y aplicarlas en una base de datos real ajena a las vistas en el curso. Estos datos refieren a hogares de Uruguay, que fueron seleccionados bajo un diseño muestral: aleatorio, estratificado, por conglomerados y en un etapa de selección.

Lo que se intentará hacer es crear un sistema de ponderadores w_i para todas las unidades elegibles respondentes de una muestra recogida, y aprtir de ellos generar estimaciones de los parametros deseados.

Parte A

Se pide calcular estimaciones puntuales,junto con sus respectivas medidas de calidad, para tres parametros en específico, utilizando los ponderadores originales. Estos resultados se muestran en la siguiente tabla:

```
##                          coef      _se    _cv  _deff
## Ingreso promedio  22111.3302 242.1899 0.0110 0.9569
## Proporción pobres     0.0802   0.0023 0.0288 1.0556
## Tasa desempleo        0.0875   0.0034 0.0390 1.0530
```

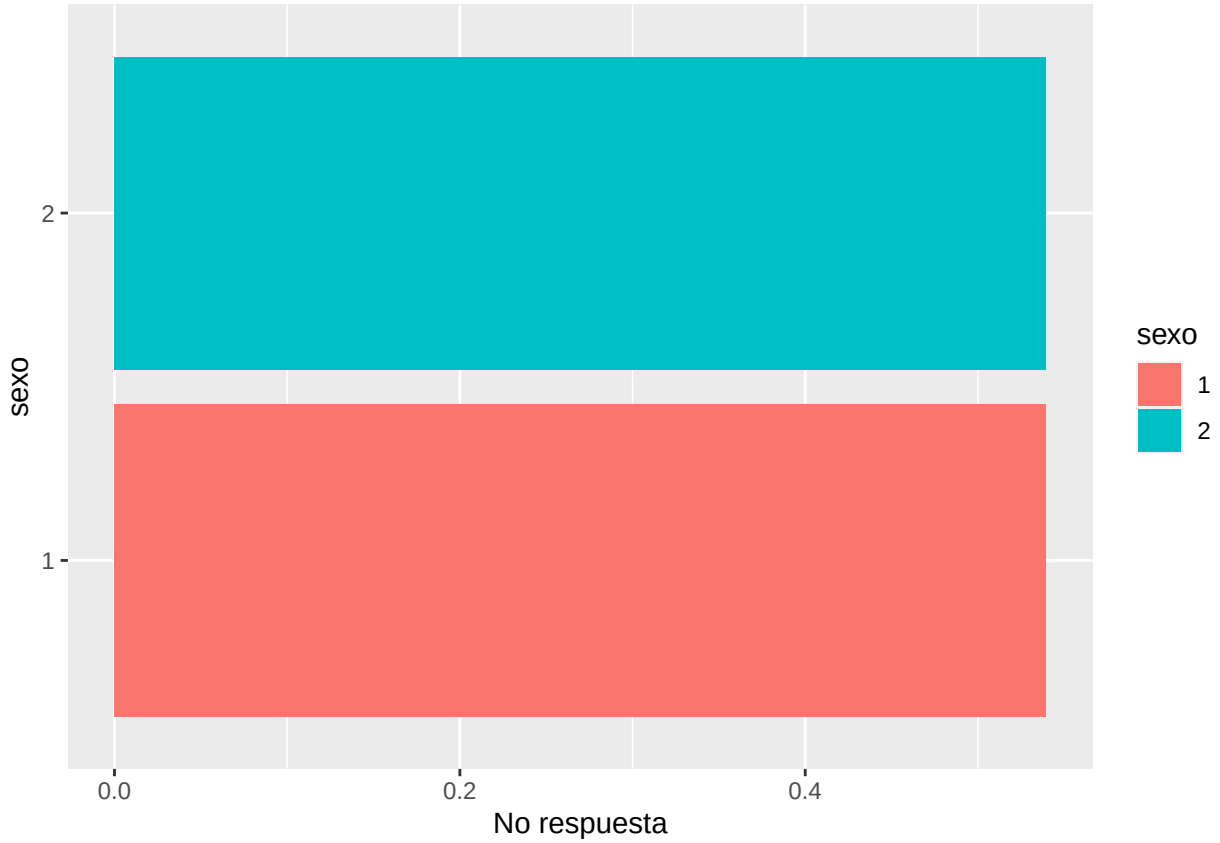
Para que sea correcto lo realizado, es necesario suponer que la no respuesta fue totalmente al azar, es decir, que no depende de ningúna variable. Haciendo este supuesto, quienés finalmente respondieron siguen siendo una muestra representativa de la población, y por lo tanto la no respuesta no induce sesgo.

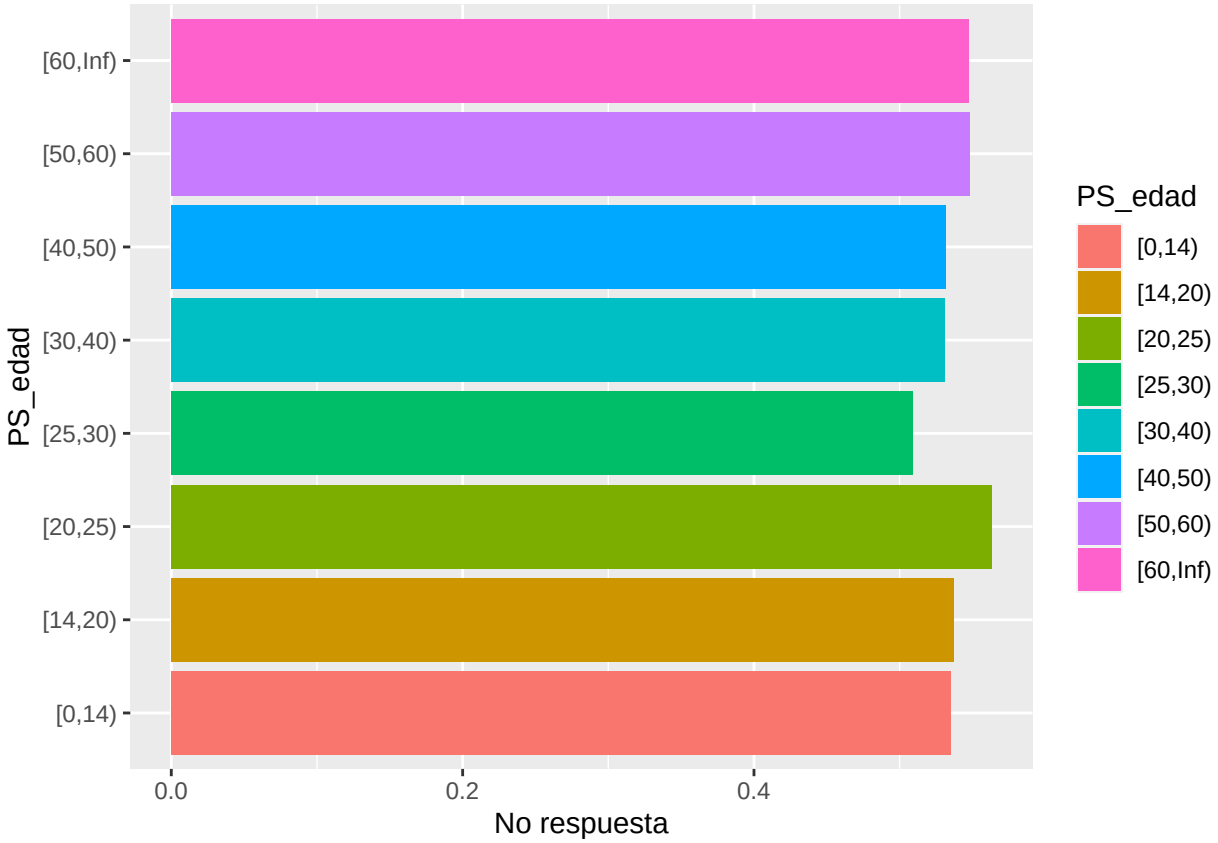
Para analizar la no respuesta, se presenta como primera aproximación,la tasa de respuesta a nivel global, sin incorporar ninguna información auxiliar de ningún tipo

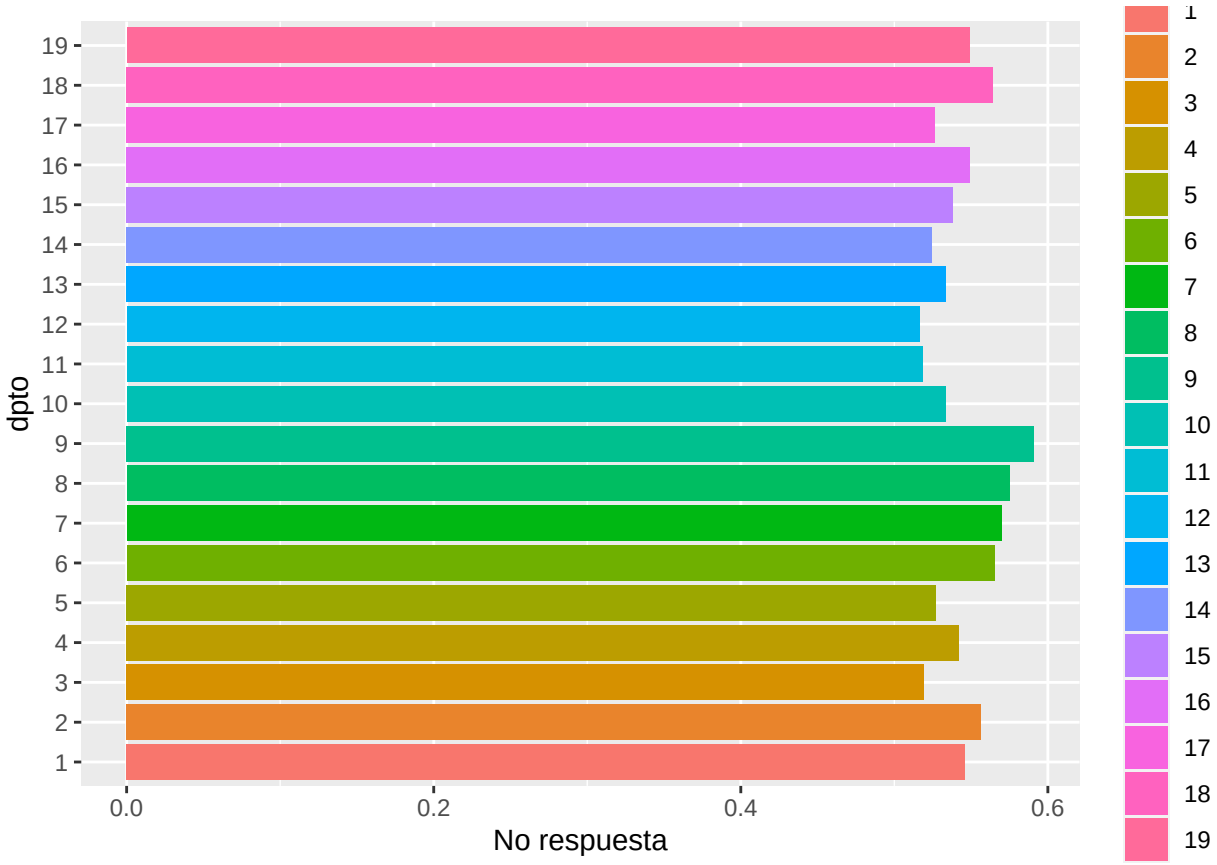
```
muestra  %>%  summarise(tr=mean(as.numeric(R)))
```

```
## # A tibble: 1 x 1
##      tr
##   <dbl>
## 1 0.540
```

A continuación se analiza si la tasa de no respuesta puede depender de alguna variable auxiliar disponible. Las candidatas son: Sexo, departamento y Edad, está última agrupandose por tramos debido la falta de practicidad de trabajar con 99 valores.


sexo
2
1
0.0
0.2
0.4
No respuesta
sexo
1
2


[60,Inf)
[50,60)
[40,50)
[30,40)
[25,30)
[20,25)
[14,20)
[0,14)
PS_edad
0.0
0.2
0.4
No respuesta
PS_edad
[0,14)
[14,20)
[20,25)
[25,30)
[30,40)
[40,50)
[50,60)
[60,Inf)



La tasa de respuesta global es de aproximadamente 54%, y al anzalizar la relación con las variables auxiliares, se puede ver que no hay tanta relación, dado que sin importar sexo,edad o departamento, la tasa de respuesta siempre está entre 50% y 60%.

Parte B

A partir de este punto, se asume que la no respuesta es MAR, que significa que su variabilidad puede ser explicada a partir de las variables auxiliares.

A continuación, se intenta observar si los estratos pueden ser una buena variable explicativa para la no respuesta, y cómo quedarían los ponderadores ajustados por este factor

A su vez nos creamos g clases, en el cual cada individuo dentro de cada clase tiene igual probabilidad de responder

$$\phi_{i,g} = TR_w = \frac{\sum_{i \in R} w_i}{\sum_{i \in s} w_i}$$

```
## Joining with `by = join_by(estrato)`
```

```
## # A tibble: 12 x 3
##    estrato    tr  tr_w
##    <fct>   <dbl> <dbl>
##  1 12       0.58  0.58
##  2 3        0.57  0.57
```

```
##  3 5        0.57  0.57
##  4 11       0.56  0.56
##  5 4        0.55  0.55
##  6 7        0.54  0.54
##  7 10       0.53  0.54
##  8 1        0.52  0.53
##  9 8        0.53  0.53
## 10 9        0.53  0.53
## 11 2        0.52  0.52
## 12 6        0.51  0.51
```

La tasa de respuesta no tiene una extrema variabilidad según estrato, y a su vez se puede apreciar tambien que tanto el ajuste por no respuesta como la propension teorica de no respuesta son similares entre si. Por consecuencia, parece razonable suponer que la propensión a responder no está fuertemente correlacionado con el estrato al que pertenece la unidad

El modelo utilizado es el MAR, en el cual se utiliza info auxiliar para poder modelar la no respuesta, con estas covariables , nos construimos clases. La idea es que la propensión a no responder de cada clase sean distintas entre si (iguales dentro de cada clase?), para buscar cuales grupos son mas propensos a no responder. Se asume que los individuos dentro de cada estrto, tienen misma probabilidad de responder

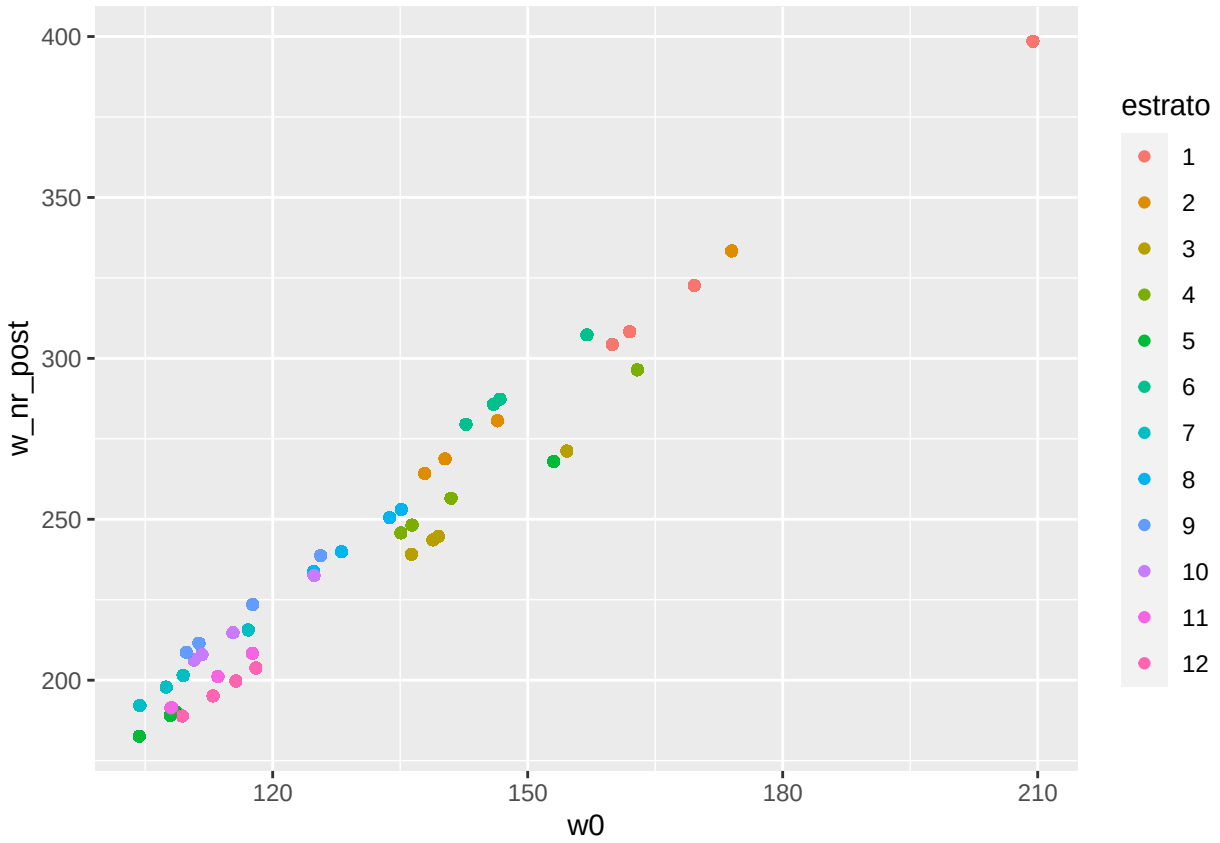
```
## [1] 1.029101
```

El deff_k indica que no hay problemas con la variabilidad de los nuevos ponderadores

Computamos las estimaciones usando los ponderadores por no respuesta de la manera

$$w_i^{nr} = \frac{1}{\pi_i \phi_i}$$

y luego se comparan con los ponderadores originales.



En el anterior grafico podemos observar como varian los ponderadores teoricos y los ajustados por n-r. Podemos observar que el rango en w_0 varia entre 100 y 215 mientras que el ajustado el rango es mas amplio. Vemos que el estratos con ingresos altos (Montevideo Alto con un 58 %) tiene la tasa de no respuesta mas grande mientras que el estrato 6 tiene la tasa de no respuesta mas chica con un 51%

Con estos nuevos ponderadores, se vuelven a estimar mismos parametros que en la parte anterior

```
##                         coef      _se    _cv  _deff
## Ingreso promedio  21957.1035 239.0505 0.0109 0.9453
## Proporción pobres     0.0809   0.0023 0.0288 1.0666
## Tasa desempleo        0.0879   0.0034 0.0392 1.0591
```

Se puede ver que las estimaciones son bastante parecidas en general. Lo único que cambia de una manera significativa es la estimación puntual del ingreso promedio, pero en términos de desvío y relación con el muestreo aleatorio simple, los hallazgos son bastane parecidos.

Pregunta 4

Para estimar las propensiones a responder, se elige utilizar el método de bosques aleatorios, entendiendo que es el algoritmo con mayor potencia para realizar las estimaciones. Se utilizan 100 árboles, y se utilizan estrato, sexo,edad y departamento como variables explicativas.

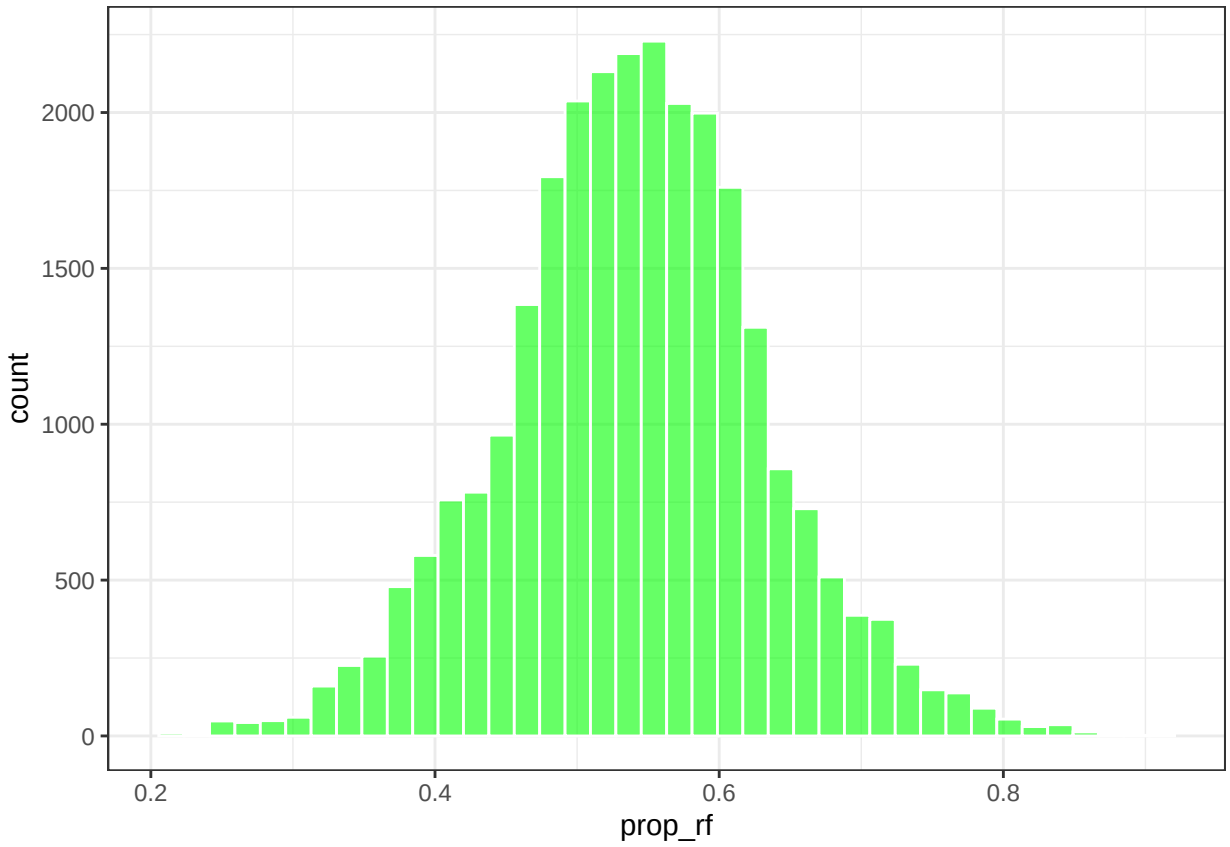
```
muestra = muestra %>% mutate(R=factor(R))
rf_model= parsnip::rand_forest( trees = 100) %>%
  set_engine('ranger') %>%
```

```
  set_mode('classification') %>%
  fit(R~estrato+edad+sexo+dpto,
              data=muestra)
```

Para evaluar su desempeo al predecir, se muestra la respectiva matriz de confusión, que compara la predicción con el valor real.

```
##            Truth
## Prediction     0     1
##          0  5549  2935
##          1  6826 11584
```

La tasa de acierto en la predicción no es tan buena (alrededor del 64%),y tiende a predecir que el resultado va a ser respuesta por sobre no respuesta. De igual manera, era esperable encontrar algo así, debido a que al analizar la tasa de respuesta en función de cada variable explicativa en la parte anterior, se había encontrado que en todos los subgrupos hubo una tasa de respuesta mayor al 50%, por lo que es entendible que el algoritmo tienda a predecir que el individuo va a responder.De igual manera, es importante aclarar que este algoritmo fue el que tuvo mayor tasa de acierto entre todos los evaluados.

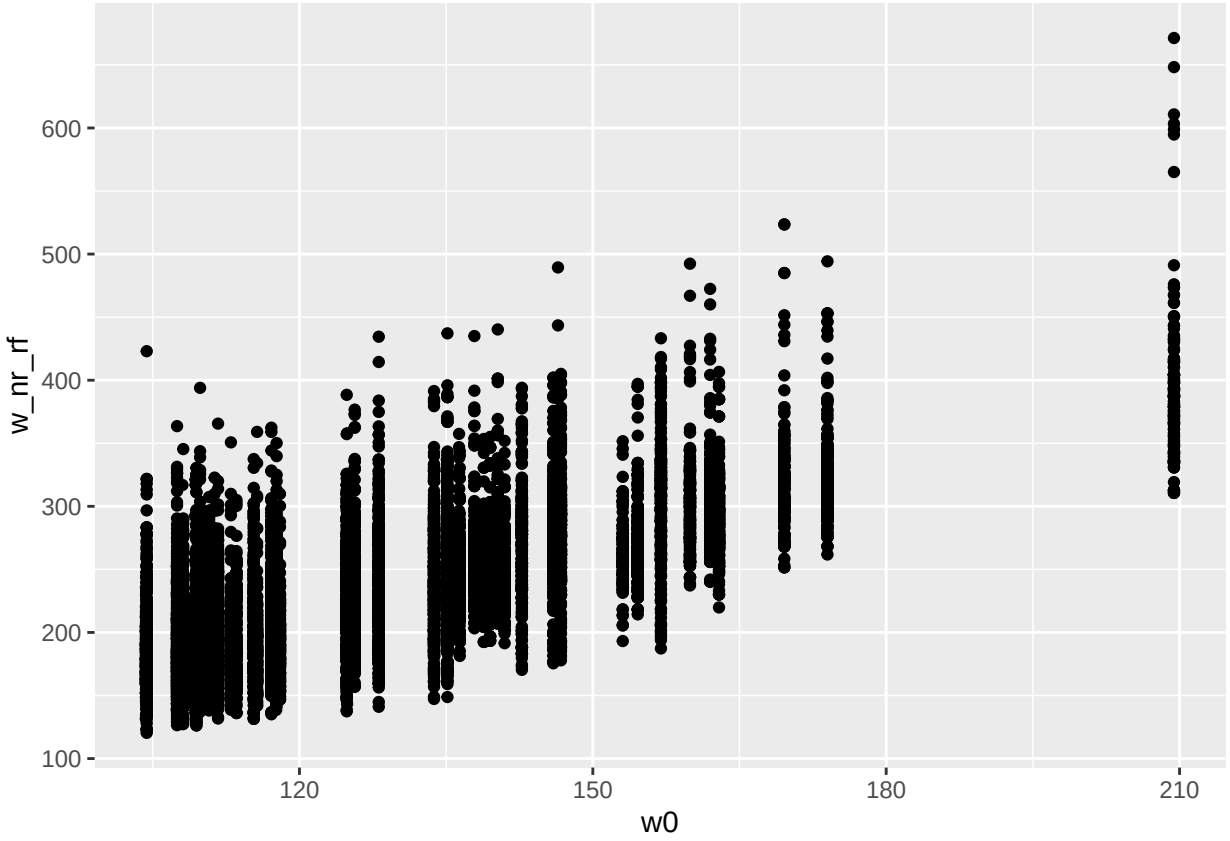


Como se había propuesto,la distribución de las propensiones se encuentra inclinada hacia el lado positivo, conllevando mayores predicciones de respuesta en detrimento de la no respuesta.

Al calcularle el efecto de Kish a los ponderadores ajustados, el resultado da menor a 1.5.

```
## [1] 1.054718
```

Nuevamente se comparan los nuevos ponderadores calculados con los ponderadores originales



Se puede observar que ponderadores en principio iguales ahora tienen una distancia significativa entre ellos. Una pregunta que se puede plantear es si esto impactará en gran medida a las nuevas estimaciones de los parametros en cuestión

```
##                         coef      _se    _cv  _deff
## Ingreso promedio  22015.9911 239.5663 0.0109 0.9371
## Proporción pobres     0.0814   0.0023 0.0285 1.0487
## Tasa desempleo        0.0880   0.0035 0.0394 1.0835
```

Las nuevas estimaciones están proximas a las realizadas en las partes anteriores, no se percibe evidencia clara de que el procedimineto realizado fue fructífero.

Pregunta 6

Se utilizan las $\hat{\phi}_i$ del random forest para crear clases de no respuesta y luego se le aplica un factor de ajuste comun, el cual en este caso es computado utilizando la mediana. En función de la estructura de los datos,se elige trabajar con quintiles

```
##   clase_rf total           Clase
## 1        1     1 (0.0195,0.4690]
## 2        2  5379 (0.4690,0.5198]
## 3        3  5380 (0.5198,0.5631]
## 4        4  5379 (0.5631,0.6109]
## 5        5 10755 (0.6109,0.9126]
```

```
## [1] 1.043413
```

Aquí no hay problemas con el efecto Kish. Por lo tanto, se pasa a las estimaciones

```
##                         coef      _se    _cv  _deff
## Ingreso promedio  21976.6434 237.0731 0.0108 0.9227
## Proporción pobres     0.0807   0.0023 0.0284 1.0340
## Tasa desempleo        0.0880   0.0035 0.0393 1.0712
```

Sellega a resultados parecidos

Ejercicio 3

EN esta parte el objetivo es calibrar los ponderadores anteriores, para que se ajusten a las predicciones poblacionales según edad, sexo y departamento

```
sum(depto$personas)-(sum(edadysexo$hombres)+sum(edadysexo$mujeres))
```

```
## [1] 5
```

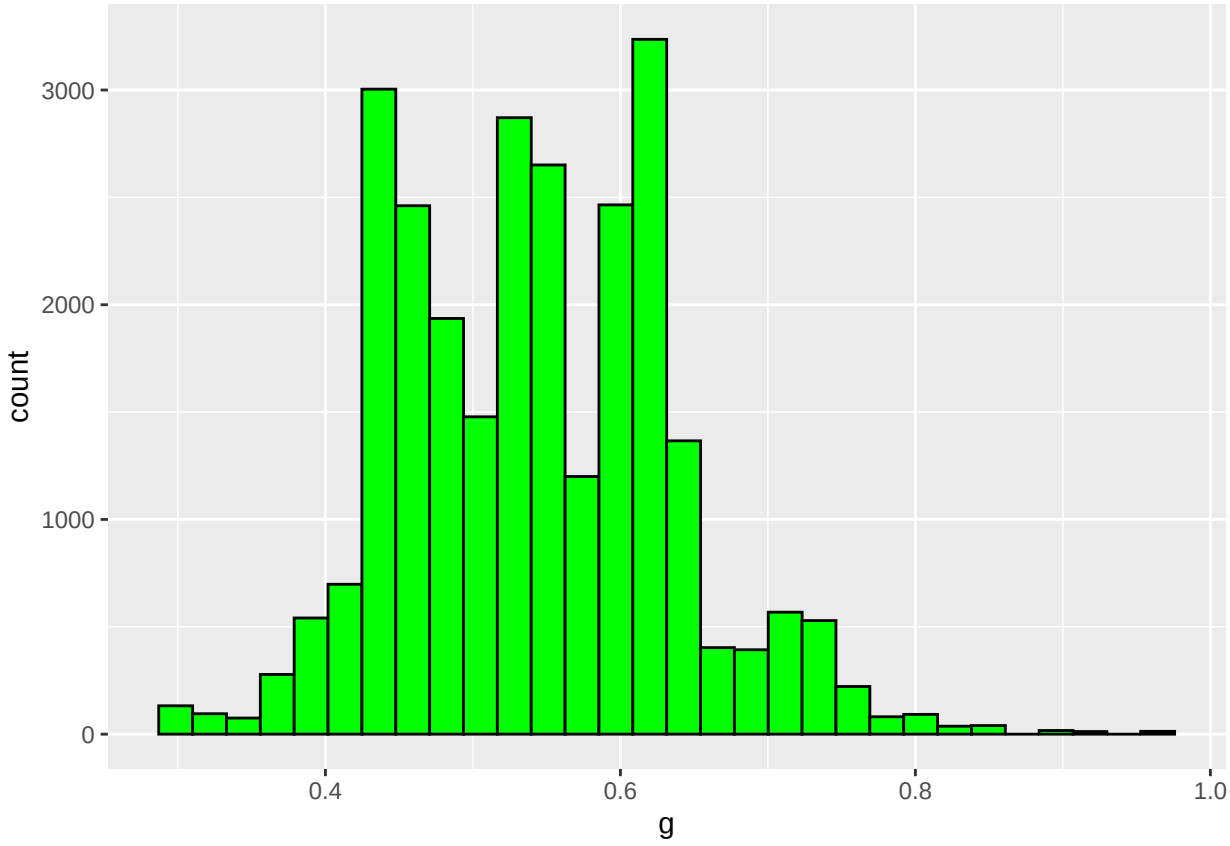
En primer lugar, hay que arreglar la diferencia en el total poblacional estimado entre las diferentes bases de datos, La diferencia es que la base de departamente cuenta en total 5 personas mas que la base de edad y sexo. Por lo tanto, se decide agregar 5 personas a la base de edad y sexo, en un lugar aleatorio

```
set.seed(4)
i<-sample(nrow(edadysexo),1)
j<-sample(c(2,3),1)
edadysexo[i,j]<-edadysexo[i,j]+5
```

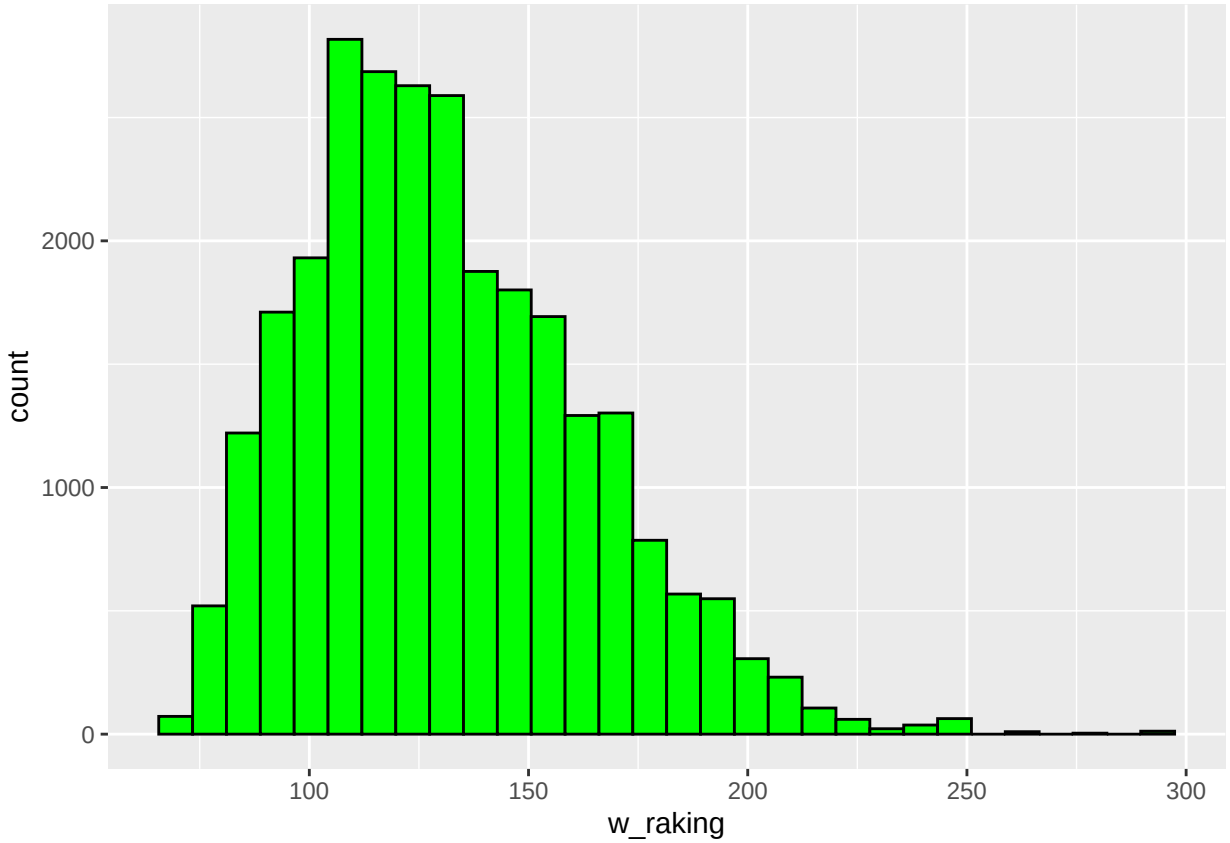
Ya resuelto esto, se procede a presentar los totales poblaciones de la manera requerida por la función calibrate

```
## `summarise()` has grouped output by 'PS_edad'. You can override using the
## `.groups` argument.
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



No se detectan poneradores negativos, y hay un par que pueden ser considerados muy altos,pero tampoco se considera que sea algo inaceptable, por lo que se aceptan y se sigue adelante

Ejercico 4

Habiendo obtenido los ponderadores calibrados y ponderados por la no respuesta, se vuelven a calcular los parametros.

```
##                         coef      _se    _cv  _deff
## Ingreso promedio  21443.1607 232.4330 0.0108 0.9154
## Proporción pobres     0.0866   0.0024 0.0283 1.1093
## Tasa desempleo        0.0901   0.0036 0.0394 1.1016
```

Y se agrega la determinación para cada departamento, además del intervalo de confianza para cada uno.

Proporción de personas pobres:

```
## # A tibble: 20 x 6
##    dpto    coef   `_se`  `_cv`  `_low` `_upp`
##    <fct>  <dbl>   <dbl>  <dbl>   <dbl>  <dbl>
##  1 1     0.133  0.00490 0.0369 0.123   0.142
##  2 2     0.117  0.0172  0.147  0.0834  0.151
##  3 3     0.0458 0.00477 0.104  0.0364  0.0552
##  4 4     0.0745 0.0131  0.176  0.0487  0.100
```

```
##  5 5     0.0285 0.00705 0.248  0.0147  0.0423
##  6 6     0.0826 0.0156  0.189  0.0521  0.113
##  7 7     0.0340 0.0150  0.441  0.00463 0.0633
##  8 8     0.0568 0.0127  0.224  0.0319  0.0817
##  9 9     0.0612 0.0159  0.260  0.0300  0.0924
## 10 10    0.0145 0.00458 0.316  0.00550 0.0235
## 11 11    0.0839 0.0121  0.144  0.0603  0.108
## 12 12    0.0528 0.0138  0.261  0.0258  0.0797
## 13 13    0.0747 0.0113  0.151  0.0526  0.0967
## 14 14    0.0490 0.0112  0.230  0.0269  0.0710
## 15 15    0.0712 0.0105  0.148  0.0506  0.0919
## 16 16    0.0140 0.00529 0.378  0.00364 0.0244
## 17 17    0.0732 0.0137  0.187  0.0464  0.100
## 18 18    0.106  0.0159  0.150  0.0748  0.137
## 19 19    0.0878 0.0193  0.220  0.0500  0.126
## 20 Total 0.0866 0.00245 0.0283 0.0818  0.0914
```

Ingreso promedio

Para las personas empleadas mayores de 25 años

```
## # A tibble: 20 x 6
##    dpto    coef `_se`  `_cv` `_low` `_upp`
##    <fct>  <dbl> <dbl>  <dbl>  <dbl>  <dbl>
##  1 1     45542.  812. 0.0178 43950. 47133.
##  2 2     28683. 1848. 0.0644 25059. 32306.
##  3 3     36483. 1065. 0.0292 34395. 38571.
##  4 4     27689. 1620. 0.0585 24514. 30865.
##  5 5     32289. 1433. 0.0444 29479. 35099.
##  6 6     29484. 2460. 0.0834 24662. 34307.
##  7 7     35707. 2484. 0.0696 30838. 40576.
##  8 8     33569. 1970. 0.0587 29706. 37432.
##  9 9     32440. 2597. 0.0801 27349. 37532.
## 10 10    35916. 1573. 0.0438 32833. 38999.
## 11 11    32281. 2001. 0.0620 28358. 36205.
## 12 12    33111. 2064. 0.0623 29064. 37158.
## 13 13    25341. 1290. 0.0509 22813. 27870.
## 14 14    29422. 1833. 0.0623 25829. 33015.
## 15 15    35006. 2358. 0.0674 30384. 39629.
## 16 16    33254. 1263. 0.0380 30778. 35731.
## 17 17    35366. 2045. 0.0578 31357. 39375.
## 18 18    26401. 1358. 0.0514 23740. 29063.
## 19 19    26602. 2006. 0.0754 22669. 30536.
## 20 Total 38412.  435. 0.0113 37559. 39266.
```

Tasa de desempleo:

```
## # A tibble: 20 x 6
##    dpto    coef   `_se`  `_cv` `_low` `_upp`
##    <fct>  <dbl>   <dbl>  <dbl>  <dbl>  <dbl>
##  1 1     0.0905 0.00573 0.0633 0.0793 0.102
##  2 2     0.0791 0.0211  0.267  0.0377 0.120
```

```
##  3 3     0.105  0.00991 0.0948 0.0852 0.124
##  4 4     0.0626 0.0193  0.308  0.0248 0.100
##  5 5     0.0439 0.0125  0.285  0.0193 0.0684
##  6 6     0.151  0.0274  0.182  0.0968 0.204
##  7 7     0.0750 0.0276  0.368  0.0209 0.129
##  8 8     0.0854 0.0213  0.249  0.0437 0.127
##  9 9     0.0863 0.0251  0.291  0.0371 0.136
## 10 10    0.0761 0.0142  0.186  0.0483 0.104
## 11 11    0.0968 0.0202  0.209  0.0572 0.136
## 12 12    0.0761 0.0233  0.307  0.0304 0.122
## 13 13    0.0568 0.0148  0.260  0.0278 0.0858
## 14 14    0.0991 0.0237  0.239  0.0526 0.146
## 15 15    0.106  0.0187  0.177  0.0690 0.142
## 16 16    0.0797 0.0171  0.215  0.0462 0.113
## 17 17    0.0918 0.0202  0.220  0.0522 0.131
## 18 18    0.113  0.0253  0.223  0.0636 0.163
## 19 19    0.0776 0.0285  0.367  0.0218 0.133
## 20 Total 0.0901 0.00355 0.0394 0.0832 0.0971
```

Todos estos cálculos se obtienen estimando la varianza por el método del último conglomerado, que es el predeterminado del program.

Para comparar, se plantea también cómo quedarían esas varianzas si fueran estimamadas por el método Boostrap Rao-Wu

```
boot=    design4 %>%  as.svrepdesign(design=., type='subbootstrap', replicates=1000)
```

Tasa de desempleo

```
##      dpto  desocupado/activo se.desocupado/activo       coef         _se
## 1       1 0.0905425222224118  0.00592587063707942 0.09054252 0.005730238
## 2       2 0.0790729101505123   0.0216690555114019 0.07907291 0.021115060
## 3       3  0.104616066453344  0.00992493704017421 0.10461607 0.009912395
## 4       4 0.0626071066954783   0.0191917263122693 0.06260711 0.019293888
## 5       5 0.0438617457156407   0.0124851322967923 0.04386175 0.012511256
## 6       6  0.150574311657827   0.0274655171785906 0.15057431 0.027446851
## 7       7 0.0750331854914854   0.0274219704321955 0.07503319 0.027640171
## 8       8 0.0853864349782683    0.021519530268609 0.08538643 0.021286333
## 9       9 0.0863243749531366   0.0256447914074933 0.08632437 0.025118703
## 10     10 0.0761168507691768   0.0143379050191727 0.07611685 0.014169907
## 11     11 0.0967796468243062   0.0205809646038721 0.09677965 0.020201168
## 12     12 0.0761190770562147   0.0232923234997219 0.07611908 0.023349331
## 13     13 0.0567818327023344    0.015242167854445 0.05678183 0.014784783
## 14     14 0.0991101994597546   0.0237744631630257 0.09911020 0.023730569
## 15     15  0.105602862581018   0.0182541734351887 0.10560286 0.018654502
## 16     16 0.0796742378805422   0.0173333336035752 0.07967424 0.017096306
## 17     17 0.0917940393599448   0.0199728023203716 0.09179404 0.020210481
## 18     18  0.113126113907656   0.0266437286606147 0.11312611 0.025271061
## 19     19 0.0776086473247003   0.0268247389079811 0.07760865 0.028475859
## 110 Total 0.0901229438612022  0.00353356023635981 0.09012294 0.003554071
```

Las varianzas estimadas son muy parecidas a las conseguidas con el método del último conglomerado

Pobreza

```
## Warning in `[<-.factor`(`*tmp*`, ri, value = "Total"): invalid factor level, NA
## generated
```

```
##    dpto            pobreza                   se       coef         _se
## 1     1  0.132890313287982 0.00486181834696093 0.13289031 0.004898673
## 2     2  0.117081178619493  0.0168596752386714 0.11708118 0.017203438
## 3     3 0.0458010794761907 0.00467401378352402 0.04580108 0.004771547
## 4     4 0.0744595491000405  0.0130231464000374 0.07445955 0.013134326
## 5     5 0.0284737581343477 0.00698665834886529 0.02847376 0.007050053
## 6     6 0.0826330419023514  0.0149838699693488 0.08263304 0.015601953
## 7     7 0.0339868911633081  0.0155776552273407 0.03398689 0.014975981
## 8     8 0.0567821793030592  0.0123691223875986 0.05678218 0.012694316
## 9     9 0.0611806901916739  0.0158374175097005 0.06118069 0.015931077
## 10   10 0.0144836252320002 0.00434709298382041 0.01448363 0.004583082
## 11   11 0.0839266565137773  0.0121800883063661 0.08392666 0.012051924
## 12   12 0.0527587867406834  0.0135764443653989 0.05275879 0.013759070
## 13   13 0.0746575784484269  0.0115595521215304 0.07465758 0.011267321
## 14   14 0.0489600394598582    0.01134724808839 0.04896004 0.011248305
## 15   15 0.0712145073154481  0.0105920436809319 0.07121451 0.010540174
## 16   16 0.0140021156091095  0.0050068905090159 0.01400212 0.005288928
## 17   17 0.0731998319210566  0.0138102837213538 0.07319983 0.013666687
## 18   18  0.105921467721873  0.0162781981295779 0.10592147 0.015901140
## 19   19 0.0877770601868102  0.0196658389114456 0.08777706 0.019281425
## 20 <NA>         0.08656063         0.002475025 0.08656063 0.002448510
```

También se llega a desvios similares

Ingreso Promedio

```
## Warning in `[<-.factor`(`*tmp*`, ri, value = "Total"): invalid factor level, NA
## generated
```

```
##    dpto          ingreso               se     coef       _se
## 1     1 45541.5808752753 727.584522542542 45541.58  811.9552
## 2     2 28682.8075311933 1822.29033371272 28682.81 1848.3821
## 3     3 36482.8579392181  1047.6242964458 36482.86 1065.2202
## 4     4 27689.2919218972  1620.4523645124 27689.29 1619.7348
## 5     5 32289.2084775338 1450.03466075682 32289.21 1433.3086
## 6     6 29484.3079110023 2485.10035888772 29484.31 2459.9664
## 7     7 35707.1392997103 2577.49922776872 35707.14 2483.6125
## 8     8 33568.9547440251 1916.40589289786 33568.95 1970.4321
## 9     9 32440.1728222287 2515.76534611147 32440.17 2597.2738
## 10   10 35915.9815359889 1533.31574404554 35915.98 1572.6366
## 11   11 32281.3486465908 1974.58897048262 32281.35 2001.3101
## 12   12 33111.2351250115 2028.52339482608 33111.24 2064.3422
## 13   13 25341.4150813443 1265.44634272124 25341.42 1289.6099
## 14   14  29422.078797793 1831.06332977019 29422.08 1832.8587
## 15   15 35006.3196020646  2342.5656831047 35006.32 2358.0699
## 16   16 33254.0211777799 1293.52315423334 33254.02 1263.2738
## 17   17  35366.114997322 2077.16717166437 35366.11 2044.8477
## 18   18 26401.0947407142 1317.16207340305 26401.09 1357.6200
```

```
## 19   19 26602.1715477249 1941.85539647617 26602.17 2006.4820
## 20 <NA>         38412.47         420.2542 38412.47  435.2365
```

En este caso si difieren un poco más lo desvios aparentemente. Aunque es posible que ahora sea más evidente debido a la mayor magnitud de los valores,cuando en realidad antes diferían en la misma proporción