

EST-24107: Simulación

Profesor: Alfredo Garbuno Iñigo — Otoño, 2022 — *Bootstrap*.

Objetivo: Que veremos.

Lectura recomendada: Referencia.

1. ERROR ESTÁNDAR BOOTSTRAP E INTERVALOS NORMALES

Ahora podemos construir nuestra primera versión de intervalos de confianza basados en la distribución *bootstrap*.

- Supongamos que queremos estimar una cantidad poblacional θ con una estadística $\hat{\theta} = t(X_1, \dots, X_n)$, donde X_1, \dots, X_n es una muestra independiente e idénticamente distribuida de la población.
- Suponemos además que la distribución muestral de $\hat{\theta}$ es aproximadamente normal (el teorema central del límite aplica), y está centrada en el verdadero valor poblacional θ .

Ahora queremos construir un intervalo que tenga probabilidad 95 % de cubrir al valor poblacional θ . Tenemos que

$$P(-2\text{ee}(\hat{\theta}) < \hat{\theta} - \theta < 2\text{ee}(\hat{\theta})) \approx 0.95, \quad (1)$$

por las propiedades de la distribución normal ($P(-2\sigma < X - \mu < 2\sigma) \approx 0.95$ si X es normal con media μ y desviación estándar σ). Entonces

$$P(\hat{\theta} - 2\text{ee}(\hat{\theta}) < \theta < \hat{\theta} + 2\text{ee}(\hat{\theta})) \approx 0.95. \quad (2)$$

Es decir, la probabilidad de que el verdadero valor poblacional θ esté en el intervalo

$$[\hat{\theta} - 2\text{ee}(\hat{\theta}), \hat{\theta} + 2\text{ee}(\hat{\theta})]$$

es cercano a 0.95. En este intervalo no conocemos el error estándar (es la desviación estándar de la distribución de muestreo de $\hat{\theta}$), y aquí es donde entra la distribución *bootstrap*, que aproxima la distribución de muestreo (en términos de varianza). Lo estimamos con

$$\hat{\text{ee}}_{\text{boot}}(\hat{\theta}), \quad (3)$$

que es la desviación estándar de la **distribución *bootstrap***.

1.0.1. Definición: El **error estándar *bootstrap*** $\hat{\text{ee}}_{\text{boot}}(\hat{\theta})$ se define como la desviación estándar de la distribución *bootstrap* de θ . El **intervalo de confianza normal *bootstrap*** al 95 % está dado por

$$[\hat{\theta} - 2\hat{\text{ee}}_{\text{boot}}(\hat{\theta}), \hat{\theta} + 2\hat{\text{ee}}_{\text{boot}}(\hat{\theta})]. \quad (4)$$

Nótese que hay varias cosas que checar aquí: que el teorema central del límite aplica y que la distribución de muestreo de nuestro estimador está centrado en el valor verdadero. Esto en algunos casos se puede demostrar usando la teoría, pero más abajo veremos comprobaciones empíricas.

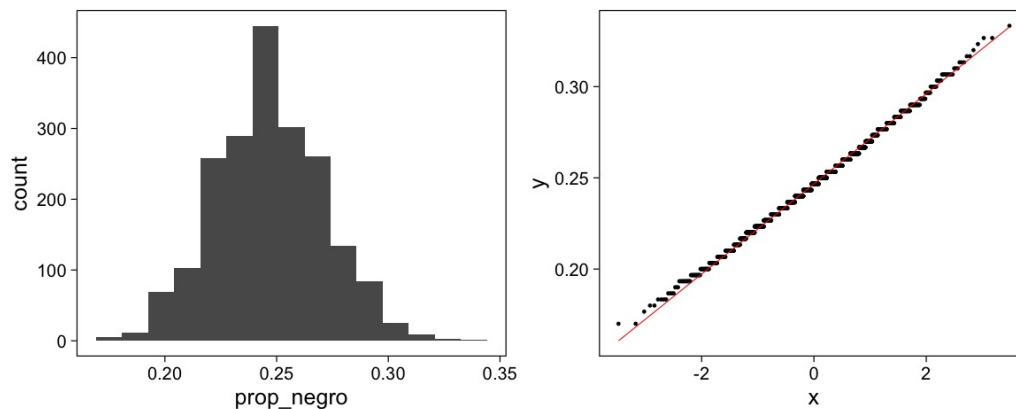
1.1. Ejemplo

Consideremos la estimación que hicimos de el porcentaje de tomadores de té que toma té negro:

```
1 ## paso 1: define el estimador
2 calc_estimador <- function(datos){
3   prop_negro <- datos >
4   mutate(negro = ifelse(Tea == "black", 1, 0)) >
5   summarise(prop_negro = mean(negro), n = length(negro)) >
6   pull(prop_negro)
7   prop_negro
8 }
9 prop_hat <- calc_estimador(te)
10 prop_hat > round(4)
```

```
1 Error in mutate(datos, negro = ifelse(Tea == "black", 1, 0)) :
2   object 'te' not found
3 Error: object 'prop_hat' not found
```

Podemos graficar su distribución bootstrap —la cual simulamos arriba—.



Y notamos que la distribución *bootstrap* es aproximadamente normal. Adicionalmente, vemos que el sesgo tiene un valor estimado de:

```
1 media_boot <- prop_negro_tbl > pull(prop_negro) > mean()
2 media_boot - prop_hat
```

```
1 Error in pull(prop_negro_tbl, prop_negro) :
2   object 'prop_negro_tbl' not found
3 Error: object 'media_boot' not found
```

De esta forma, hemos verificado que:

- La distribución *bootstrap* es aproximadamente normal (ver gráfica de cuantiles normales);
- La distribución *bootstrap* es aproximadamente insesgada.

Lo cual nos lleva a construir intervalos de confianza basados en la distribución normal. Estimamos el error estándar con la desviación estándar de la distribución *bootstrap*

```
1 ee_boot <- prop_negro_tbl > pull(prop_negro) > sd()
2 ee_boot
```

```
1 Error in pull(prop_negro_tbl, prop_negro) :
2   object 'prop_negro_tbl' not found
3 Error: object 'ee_boot' not found
```

y construimos un intervalo de confianza del 95 %:

```
1 intervalo_95 <- c(inf = prop_hat - 2 * ee_boot,
2                   centro = prop_hat,
3                   sup = prop_hat + 2 * ee_boot)
4 intervalo_95 > round(3)
```

```
1 Error: object 'prop_hat' not found
2 Error: object 'intervalo_95' not found
```

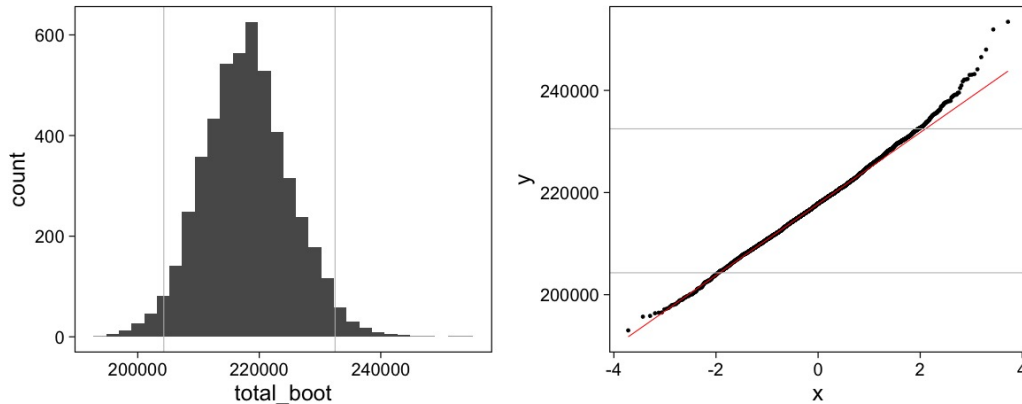
Este intervalo tiene probabilidad del 95 % de capturar al verdadero poblacional.

2. INVENTARIOS DE CASAS VENDIDAS

Ahora consideremos el problema de estimar el total del valor de las casas vendidas en un periodo. Tenemos una muestra de tamaño $n = 150$:

```
1 ## muestra original
2 set.seed(121)
3 muestra_casas <- sample_n(poblacion_casas, size = 150)
4 ## paso 1: define el estimador
5 calc_estimador_casas <- function(datos){
6   N <- nrow(poblacion_casas)
7   n <- nrow(datos)
8   total_muestra <- sum(datos$precio_miles)
9   estimador_total <- (N / n) * total_muestra
10  estimador_total
11 }
```

```
1 ## paso 2: define el proceso de remuestreo
2 muestra_boot <- function(datos){
3   ## tomar muestra con reemplazo del mismo tamaño
4   sample_n(datos, size = nrow(datos), replace = TRUE)
5 }
```



En este caso, distribución de muestreo presenta cierta asimetría, pero la desviación no es grande. En la parte central la aproximación normal es razonable. Procedemos a checar sesgo

```
1 total_est <- calc_estimador_casas(muestra_casas)
2 sesgo <- mean(totales_boot$total_boot) - total_est
3 sesgo
```

```
1 Error in nrow(poblacion_casas) : object 'poblacion_casas' not found
2 Error in mean(totales_boot$total_boot) : object 'totales_boot' not found
3 Error: object 'sesgo' not found
```

Este número puede parecer grande, pero sí calculamos la desviación relativa con respecto al estimador vemos que es chico en la escala de la distribución *bootstrap*:

```
1 sesgo_relativo <- sesgo / total_est
2 sesgo_relativo
```

```
1 Error: object 'sesgo' not found
2 Error: object 'sesgo_relativo' not found
```

De forma que procedemos a construir intervalos de confianza como sigue :

```
1 ee_boot <- sd(totales_boot$total_boot)
2 c(inf = total_est - 2*ee_boot, centro = total_est, sup = total_est + 2*ee_boot
  )
```

```
1 Error in is.data.frame(x) : object 'totales_boot' not found
2 Error: object 'total_est' not found
```

Que está en miles de dólares. En millones de dólares, este intervalo es:

```
1 intervalo_total <- c(inf = total_est - 2*ee_boot,
2                       centro = total_est,
3                       sup = total_est + 2*ee_boot) / 1000
4 intervalo_total > round(5)
```

```
1 Error: object 'total_est' not found
2 Error: object 'intervalo_total' not found
```

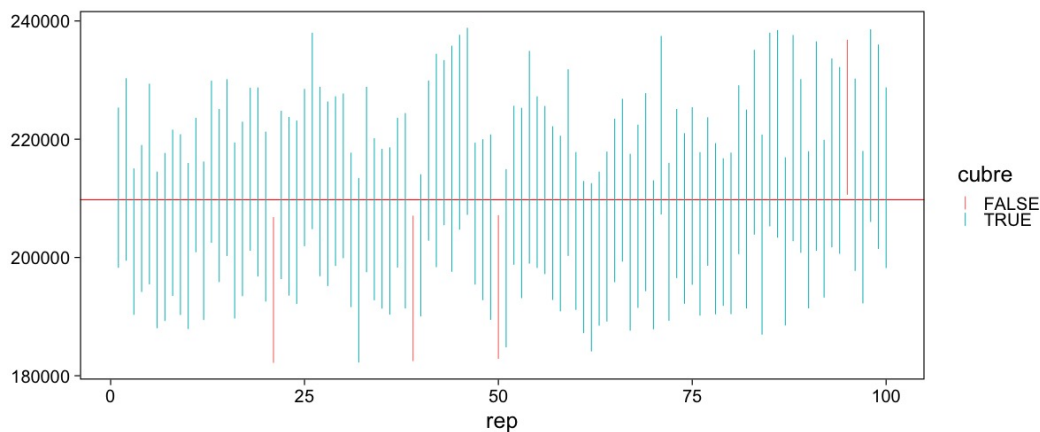
2.0.1. Nota: En este ejemplo mostraremos una alternativa de intervalos de confianza que es más apropiado cuando observamos asimetría. Sin embargo, primero tendremos que hablar de dos conceptos clave con respecto a intervalos de confianza: calibración e interpretación.

3. CALIBRACIÓN DE INTERVALOS DE CONFIANZA

¿Cómo sabemos que nuestros intervalos de confianza del 95 % nominal tienen cobertura real de 95 %? Es decir, tenemos que checar:

- El procedimiento para construir intervalos debe dar intervalos tales que el valor poblacional está en el intervalo de confianza para 95 % de las muestras.

Como solo tenemos una muestra, la calibración depende de argumentos teóricos o estudios de simulación previos. Para nuestro ejemplo de casas tenemos la población, así que podemos checar qué cobertura real tienen los intervalos normales:



La cobertura para estos 100 intervalos simulados da

```
1 total <- sum(poblacion_casas$precio_miles)
2 sims_tbl %>%
3   summarise(cobertura = mean(cubre))
```

```
1 Error: object 'poblacion_casas' not found
2 Error in summarise(., cobertura = mean(cubre)) :
3   object 'sims_tbl' not found
```

que es **consistente** con una cobertura real del 95 % (¿qué significa “consistente”? ¿Cómo puedes checarlo con el *bootstrap*?)

3.0.1. Observación: En este caso teníamos la población real, y pudimos verificar la cobertura de nuestros intervalos. En general no la tenemos. Estos ejercicios de simulación se pueden hacer con poblaciones sintéticas que se generen con las características que creemos va a tener nuestra población (por ejemplo, sesgo, colas largas, etc.).

En general, no importa qué tipo de estimadores o intervalos de confianza usemos, requerimos checar la calibración. Esto puede hacerse con ejercicios de simulación con poblaciones sintéticas y tanto los procedimientos de muestreo como los tamaños de muestra que nos interesa usar.

Verificar la cobertura de nuestros intervalos de confianza por medio simulación está bien estudiado para algunos casos. Por ejemplo, cuando trabajamos con estimaciones para poblaciones teóricas. En general sabemos que los procedimientos funcionan bien en casos:

- con distribuciones simétricas que tengan colas no muy largas;
- estimación de proporciones donde no tratamos con casos raros o casos seguros (probabilidades cercanas a 0 o 1).

4. INTERPRETACIÓN INTERVALOS DE CONFIANZA

Como hemos visto, “intervalo de confianza” (de 90 % de confianza, por ejemplo) es un término **frecuentista**, que significa:

- **Cada muestra produce un intervalo distinto.** Para el 90 % de las muestras posibles, el intervalo cubre al valor poblacional.
- La afirmación es **sobre el intervalo y el mecanismo para construirlo.**
- Así que con **alta probabilidad**, el intervalo contiene el valor poblacional.
- Intervalos más anchos nos dan más incertidumbre acerca de dónde está el verdadero valor poblacional (y al revés para intervalos más angostos).

Existen también “intervalos de credibilidad” (de 90 % de probabilidad, por ejemplo), que se interpretan de forma **bayesiana**:

- Con 90 % de probabilidad (relativamente alta), creemos que el valor poblacional está dentro del intervalo de credibilidad.

Esta última interpretación es más natural. Obsérvese que para hablar de intervalos de confianza frecuentista tenemos que decir:

- Este intervalo particular cubre o no al verdadero valor, pero nuestro procedimiento produce intervalos que contiene el verdadero valor para el 90 % de las muestras.
- Esta es una interpretación relativamente débil, y muchos intervalos poco útiles pueden satisfacerla.
- La interpretación bayesiana es más natural porque expresa más claramente incertidumbre acerca del valor poblacional.

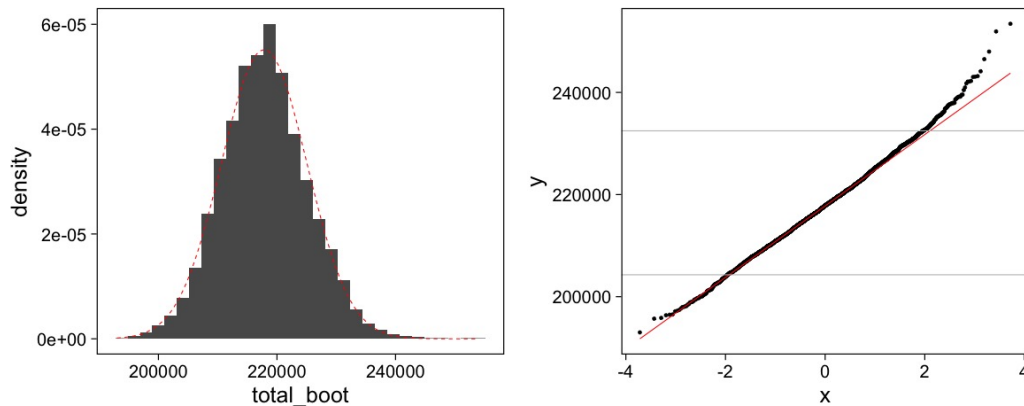
Sin embargo,

- La interpretación frecuentista nos da maneras empíricas de probar si los intervalos de confianza están bien calibrados o no: es un mínimo que “intervalos del 90 %” deberían satisfacer.

Así que tomamos el punto de vista bayesiano en la interpretación, pero buscamos que nuestros intervalos cumplan o aproximen bien garantías frecuentistas (discutimos esto más adelante). Los intervalos que producimos en esta sección pueden interpretarse de las dos maneras.

5. INTERVALOS BOOTSTRAP DE PERCENTILES

Retomemos nuestro ejemplo del valor total del precio de las casas. A través de remuestras bootstrap hemos verificado gráficamente que la distribución de remuestreo es **ligeramente** asimétrica (ver la figura de abajo).



Anteriormente hemos calculado intervalos de confianza basados en supuestos normales por medio del error estándar. Este intervalo está dado por

```
1 intervalo_total %>% round(1)
```

```
1 Error in intervalo_total %>% round(1) :  
2 object 'intervalo_total' not found
```

y por construcción sabemos que es simétrico con respecto al valor estimado, pero como podemos ver la distribución de muestreo no es simétrica, lo cual podemos confirmar por ejemplo calculando el porcentaje de muestras bootstrap que caen por arriba y por debajo del intervalo construido:

```
1 totales_boot >>  
2 mutate(upper = total_boot ≥ max(intervalo_total * 1000),  
3         lower = total_boot ≤ min(intervalo_total * 1000)) >>  
4 summarise(prop_inf = mean(lower),  
5            prop_sup = mean(upper))
```

```
1 Error in mutate(totales_boot, upper = total_boot ≥ max(intervalo_total * :  
2 object 'totales_boot' not found
```

los cuales se han calculado como el porcentaje de medias *bootstrap* por debajo (arriba) de la cota inferior (superior), y vemos que no coinciden con el nivel de confianza preestablecido (2.5 % para cada extremo).

Otra opción común que se usa específicamente cuando la distribución bootstrap no es muy cercana a la normal son los intervalos de percentiles *bootstrap*:

5.0.1. Definición: El **intervalo de percentiles *bootstrap*** al 95 % de confianza está dado por

$$[q_{0.025}, q_{0.975}], \quad (5)$$

donde q_f es el percentil f de la distribución *bootstrap*.

Otros intervalos comunes son el de 80 % o 90 % de confianza, por ejemplo, que corresponden a $[q_{0.10}, q_{0.90}]$ y $[q_{0.05}, q_{0.95}]$. **Ojo:** intervalos de confianza muy alta (por ejemplo 99.5 %) pueden tener mala calibración o ser muy variables en su longitud pues dependen del comportamiento en las colas de la distribución.

Para el ejemplo de las casas, calcularíamos simplemente

```

1 intervalo_95 <- totales_boot > pull(total_boot) %>%
2   quantile(., probs = c(0.025, 0.50, 0.975)) / 1000
3 (intervalo_95) %>% round(1)

```

```

1 Error in pull(totales_boot, total_boot) : object 'totales_boot' not found
2 Error in (intervalo_95) %>% round(1) : object 'intervalo_95' not found

```

que está en millones de dólares. Nótese que es similar al intervalo de error estándar.

Otro punto interesante sobre los intervalos *bootstrap* de percentiles es que lidian naturalmente con la asimetría de la distribución bootstrap. Ilustramos esto con la distancia de los extremos del intervalo con respecto a la media:

```

1 abs(intervalo_95 - total_est/1000)

```

```

1 Error: object 'intervalo_95' not found

```

Los intervalos de confianza nos permiten presentar un rango de valores posibles para el parámetro de interés. Esto es una notable diferencia con respecto a presentar sólo un candidato como estimador. Nuestra fuente de información son los datos. Es por esto que si vemos valores muy chicos (grandes) en nuestra muestra, el intervalo se tiene que extender a la izquierda (derecha) para compensar dichas observaciones.

5.0.2. Ejercicio: Explica por qué cuando la aproximación normal es apropiada, el intervalo de percentiles al 95% es muy similar al intervalo normal de 2 errores estándar.

5.1. Ejemplo

Consideramos los datos de propinas. Queremos estimar la media de cuentas totales para la comida y la cena. Podemos hacer bootstrap de cada grupo por separado:

```

1 ## en este ejemplo usamos rsample, pero puedes escribir tu propio código
2 library(rsample)
3 propinas <- read_csv("data/propinas.csv",
4                     progress = FALSE,
5                     show_col_types = FALSE) >
6   mutate(id = 1:244)
7 propinas

```

```

1 # A tibble: 244 × 7
2   cuenta_total propina fumador dia momento num_personas id
3   <dbl>      <dbl> <chr>   <chr> <chr>      <dbl> <int>
4 1      17.0      1.01 No     Dom  Cena         2     1
5 2      10.3      1.66 No     Dom  Cena         3     2
6 3      21.0      3.5  No     Dom  Cena         3     3
7 4      23.7      3.31 No     Dom  Cena         2     4
8 5      24.6      3.61 No     Dom  Cena         4     5
9 6      25.3      4.71 No     Dom  Cena         4     6
10 7       8.77      2    No     Dom  Cena         2     7
11 8      26.9      3.12 No     Dom  Cena         4     8
12 9      15.0      1.96 No     Dom  Cena         2     9
13 10     14.8      3.23 No     Dom  Cena         2    10
14 # ... with 234 more rows
15 # Use 'print(n = ...)' to see more rows

```



```

1 ## paso 1: define el estimador
2 estimador <- function(split, ...){
3   muestra <- analysis(split) > group_by(momento)
4   muestra >
5     summarise(estimate = mean(cuenta_total), .groups = 'drop') >
6     mutate(term = momento)
7 }

```

```

1 ## paso 2: remuestrea y calcula estimador
2 boot_samples <- bootstraps(propinas, strata = momento, 1000) >
3 mutate(res_boot = map(splits, estimador))
4 ## paso 3: construye intervalos de confianza
5 intervalo_propinas_90 <- boot_samples >
6   int_pctl(res_boot, alpha = 0.10) >
7   mutate(across(where(is.numeric), round, 2))
8 intervalo_propinas_90

```

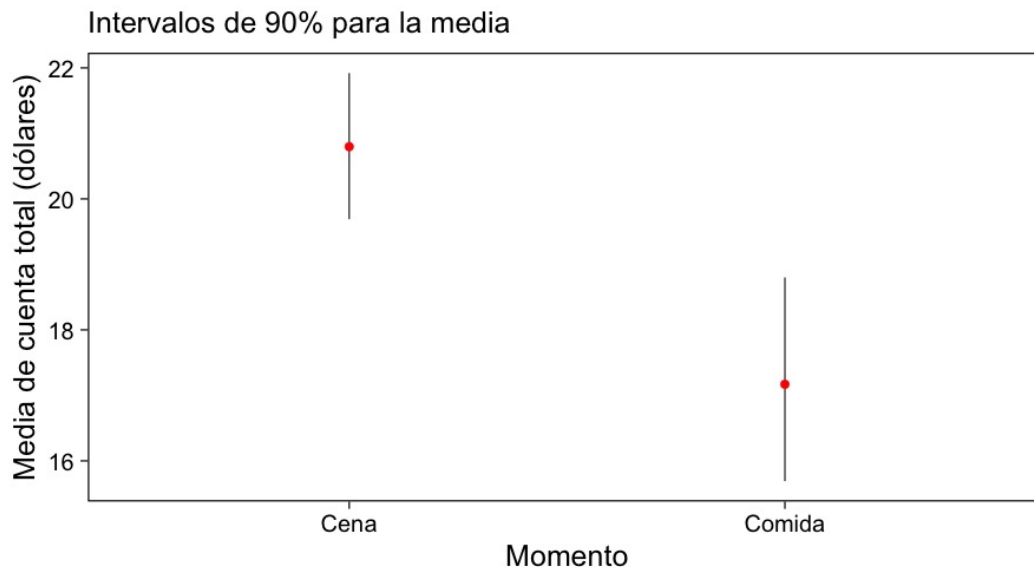
```

1 # A tibble: 2 × 6
2   term   .lower .estimate .upper .alpha .method
3   <chr>   <dbl>   <dbl>   <dbl>   <dbl> <chr>
4 1 Cena     19.7     20.8     21.9     0.1 percentile
5 2 Comida    15.7     17.2     18.8     0.1 percentile

```

Nota: `.estimate` es la media de los valores de la estadística sobre las remuestras, **no** es el estimador original.

De la tabla anterior inferimos que la media en la cuenta en la cena es más grande que la de la comida. Podemos graficar agregando los estimadores *plug-in*:



Nótese que el *bootstrap* lo hicimos por separado en cada momento del día (por eso el argumento `strata` en la llamada a `bootstraps`):

5.1.1. Funciones de cómputo: Es común crear nuestras propias funciones cuando usamos *bootstrap*, sin embargo, en R también hay alternativas que pueden resultar convenientes:

1. El paquete `rsample` (forma parte de la colección [tidymodels](#) y tiene una función para realizar el remuestreo: `bootstraps()` que regresa un arreglo cuadrangular (`tibble`, `data.frame`) que incluye una columna con las muestras bootstrap y un identificador del número y tipo de muestra.

```
1 boot_samples
```

```
1 # Bootstrap sampling using stratification
2 # A tibble: 1,000 × 3
3   splits      id      res_boot
4   <list>    <chr>    <list>
5 1 <split [244/83]> Bootstrap0001 <tibble [2 × 3]>
6 2 <split [244/82]> Bootstrap0002 <tibble [2 × 3]>
7 3 <split [244/90]> Bootstrap0003 <tibble [2 × 3]>
8 4 <split [244/93]> Bootstrap0004 <tibble [2 × 3]>
9 5 <split [244/85]> Bootstrap0005 <tibble [2 × 3]>
10 6 <split [244/90]> Bootstrap0006 <tibble [2 × 3]>
11 7 <split [244/83]> Bootstrap0007 <tibble [2 × 3]>
12 8 <split [244/93]> Bootstrap0008 <tibble [2 × 3]>
13 9 <split [244/87]> Bootstrap0009 <tibble [2 × 3]>
14 10 <split [244/98]> Bootstrap0010 <tibble [2 × 3]>
15 # ... with 990 more rows
16 # Use 'print(n = ...)' to see more rows
```

Los objetos `splits` tienen muestras de tamaño 244. Sin embargo, utilizan (por el muestreo aleatorio con reemplazo) una fracción de los datos.

```
1 boot_samples$splits[[1]]
```

```
1 <Analysis/Assess/Total>
2 <244/83/244>
```

```
1 analysis(boot_samples$splits[[1]]) ▷
2 group_by(id)
```

```
1 # A tibble: 244 × 7
2 # Groups:   id [161]
3   cuenta_total propina fumador dia      momento num_personas id
4   <dbl> <dbl> <chr> <chr> <chr> <dbl> <int>
5 1 17.0 1.01 No Dom Cena 2 1
6 2 10.3 1.66 No Dom Cena 3 2
7 3 10.3 1.66 No Dom Cena 3 2
8 4 23.7 3.31 No Dom Cena 2 4
9 5 23.7 3.31 No Dom Cena 2 4
10 6 25.3 4.71 No Dom Cena 4 6
11 7 8.77 2 No Dom Cena 2 7
12 8 26.9 3.12 No Dom Cena 4 8
13 9 26.9 3.12 No Dom Cena 4 8
14 10 15.0 1.96 No Dom Cena 2 9
15 # ... with 234 more rows
16 # Use 'print(n = ...)' to see more rows
```

El paquete de `rsample` es un paquete muy eficiente para la creación de los conjunto de remuestreo y es una de sus principales ventajas.

```
1 library(pryr)
2 c(objeto_boot = object_size(boot_samples),
3   original     = object_size(propinas),
4   remuestra    = object_size(boot_samples)/nrow(boot_samples),
5   incremento   = object_size(boot_samples)/object_size(propinas))
```

```
1 objeto_boot:  2.39 MB
2 original    : 15.43 kB
3 remuestra   :  2.39 kB
4 incremento  : 155.13 B
```

2. El paquete `boot` está asociado al libro **Bootstrap Methods and Their Applications** y tiene, entre otras, funciones para calcular replicaciones *bootstrap* y para construir intervalos de confianza usando *bootstrap*:
 - a) calculo de replicaciones *bootstrap* con la función `boot()`,
 - b) intervalos normales, de percentiles y BC_a con la función `boot.ci()`,
 - c) intervalos ABC con la función `abc.ci()`.
3. El paquete `bootstrap` contiene datos usados en [1], y la implementación de funciones para calcular replicaciones y construir intervalos de confianza:
 - a) calculo de replicaciones *bootstrap* con la función `bootstrap()`,
 - b) intervalos BC_a con la función `bcanon()`,
 - c) intervalos ABC con la función `abcnon()`.

5.1.2. Ejercicio: Justifica el procedimiento de hacer el *bootstrap* separado para cada grupo. ¿Qué supuestos acerca del muestreo se deben satisfacer? ¿Deben ser muestras aleatorias simples de cada momento del día, por ejemplo? ¿Qué harías si no fuera así, por ejemplo, si se escogieron al azar tickets de todos los disponibles en un periodo?

6. CONCLUSIONES Y OBSERVACIONES

- El principio fundamental del *bootstrap* es que podemos estimar la distribución poblacional con la distribución empírica. Por tanto para hacer inferencia tomamos muestras con reemplazo de la distribución empírica y analizamos la variación de la estadística de interés a lo largo de las muestras.
- El *bootstrap* nos da la posibilidad de crear intervalos de confianza cuando no contamos con fórmulas para hacerlo de manera analítica y sin supuestos distribucionales de la población.
- Hay muchas opciones para construir intervalos *bootstrap*, los que tienen mejores propiedades son los intervalos BC_a , sin embargo los más comunes son los intervalos normales con error estándar *bootstrap* y los intervalos de percentiles de la distribución *bootstrap*.
- Antes de hacer intervalos normales (o con percentiles de una t) vale la pena graficar la distribución *bootstrap* y evaluar si el supuesto de normalidad es razonable.
- En cuanto al número de muestras *bootstrap* se recomienda al menos 1,000 al hacer pruebas, y 10,000 o 15,000 para los resultados finales, sobre todo cuando se hacen intervalos de confianza de percentiles.

- La función de distribución empírica es una mala estimación en las colas de las distribuciones, por lo que es difícil construir intervalos de confianza (usando bootstrap no paramétrico) para estadísticas que dependen mucho de las colas.

REFERENCIAS

- [1] B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Springer US, Boston, MA, 1993. ISBN 978-0-412-04231-7 978-1-4899-4541-9. . [11](#)