Problemas: estimación no paramétrica

José R. Berrendero

Realiza un análisis descriptivo de los datos británicos de ingresos familiares contenidos en el fichero Datos-ingresos.txt (http://matematicas.uam.es/~joser.berrendero/datos/Datos-ingresos.txt). En concreto, calcula su media, mediana y desviación típica, y un estimador del núcleo de la función de densidad. Comenta los resultados.

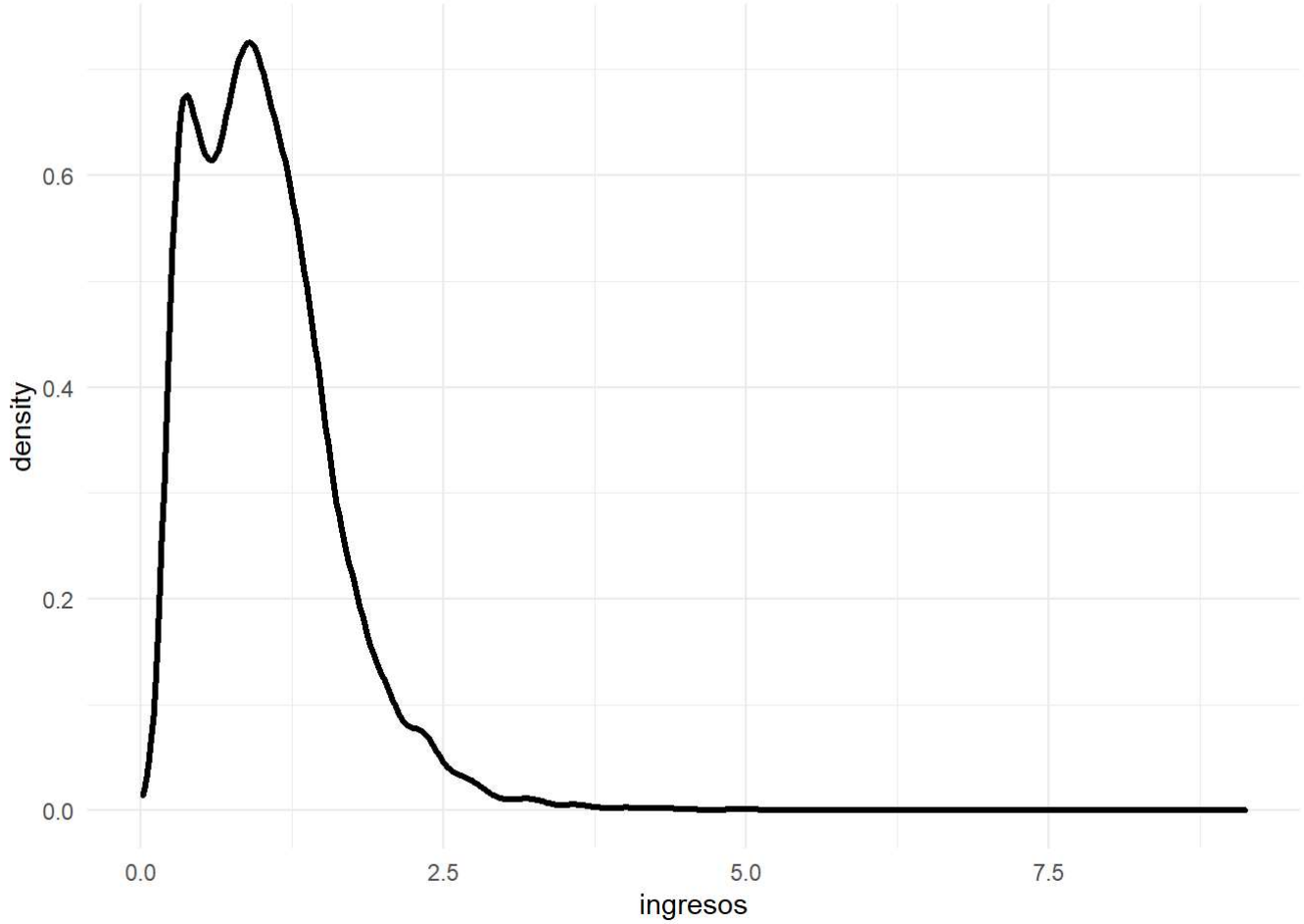
```
enlace <- "http://matematicas.uam.es/~joser.berrendero/datos/Datos-ingresos.txt"
ingresos <- read.table(enlace, col.names = c("ingresos"))

medidas_numericas <- ingresos %>%
  summarise(media = mean(ingresos),
            mediana = median(ingresos),
            desv_tipica = sd(ingresos))

medidas_numericas
```

```
##      media mediana desv_tipica
## 1 1.022779  0.9417   0.6048126
```

```
ggplot(ingresos) +
  geom_density(aes(x=ingresos),
                col = 'black',
                size = 1.1) +
  theme_minimal()
```

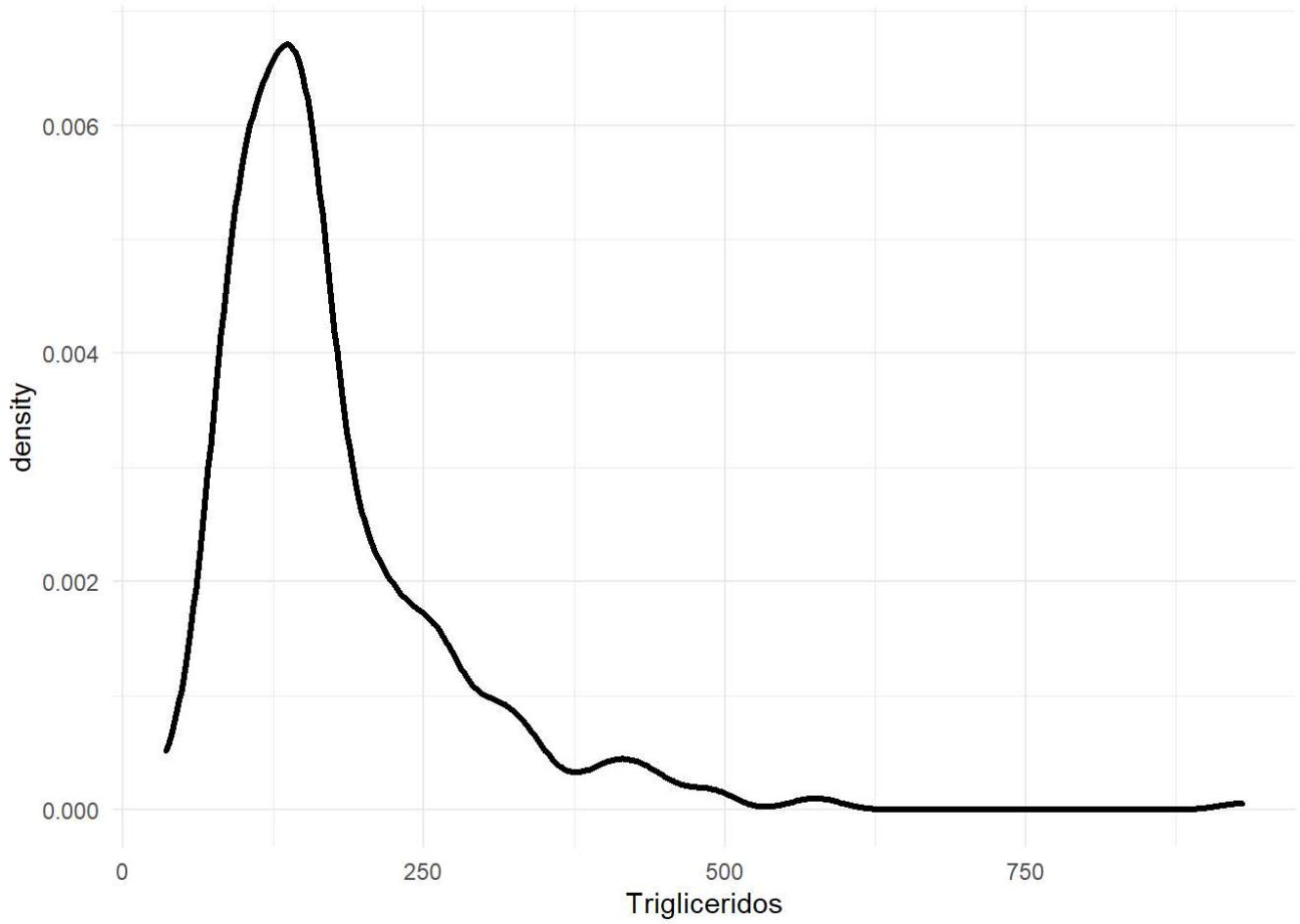


Los datos del fichero lipidos.txt (http://verso.mat.uam.es/~joser.berrendero/datos/lipidos.txt) corresponden a la concentración de colesterol y triglicéridos (mg/dl) en pacientes evaluados por tener un dolor en el pecho. De estos pacientes, 51 no presentaron evidencia de enfermedad cardiaca mientras que 320 sí la presentaron.

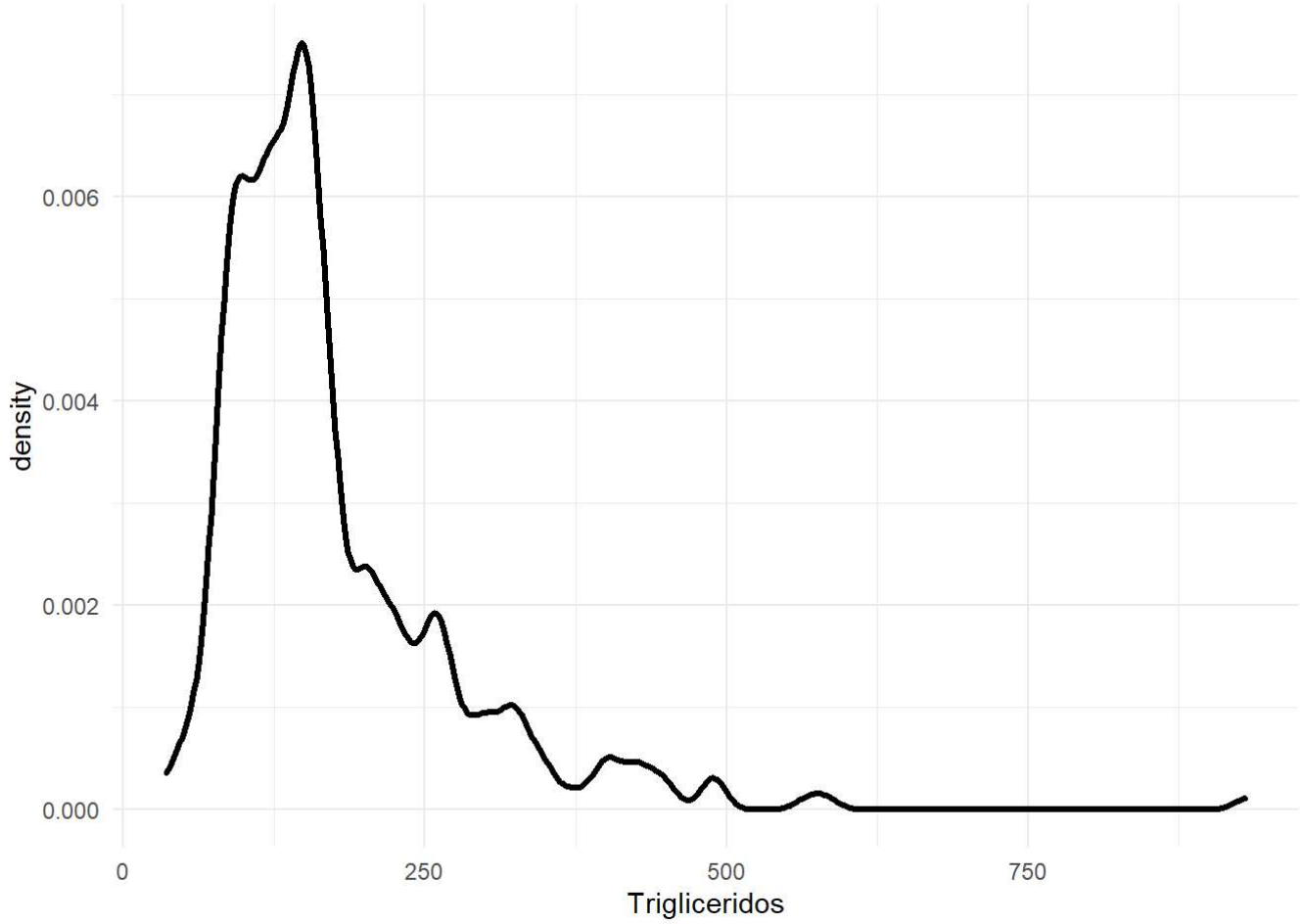
```
# importa y prepara los datos
load(url("http://verso.mat.uam.es/~joser.berrendero/datos/lipidos.RData"))
lipidos <- lipidos %>%
  mutate(Enfermo = factor(Enfermo)) %>%
  rename(Enfermedad = Enfermo)
```

- Representa un estimador del núcleo de la densidad para la variable correspondiente a la concentración de triglicéridos (utiliza primero todos los datos y después trata por separado a los individuos sanos y enfermos). Experimenta utilizando distintos núcleos y distintos valores del parámetro de suavizado. Comenta el resultado.

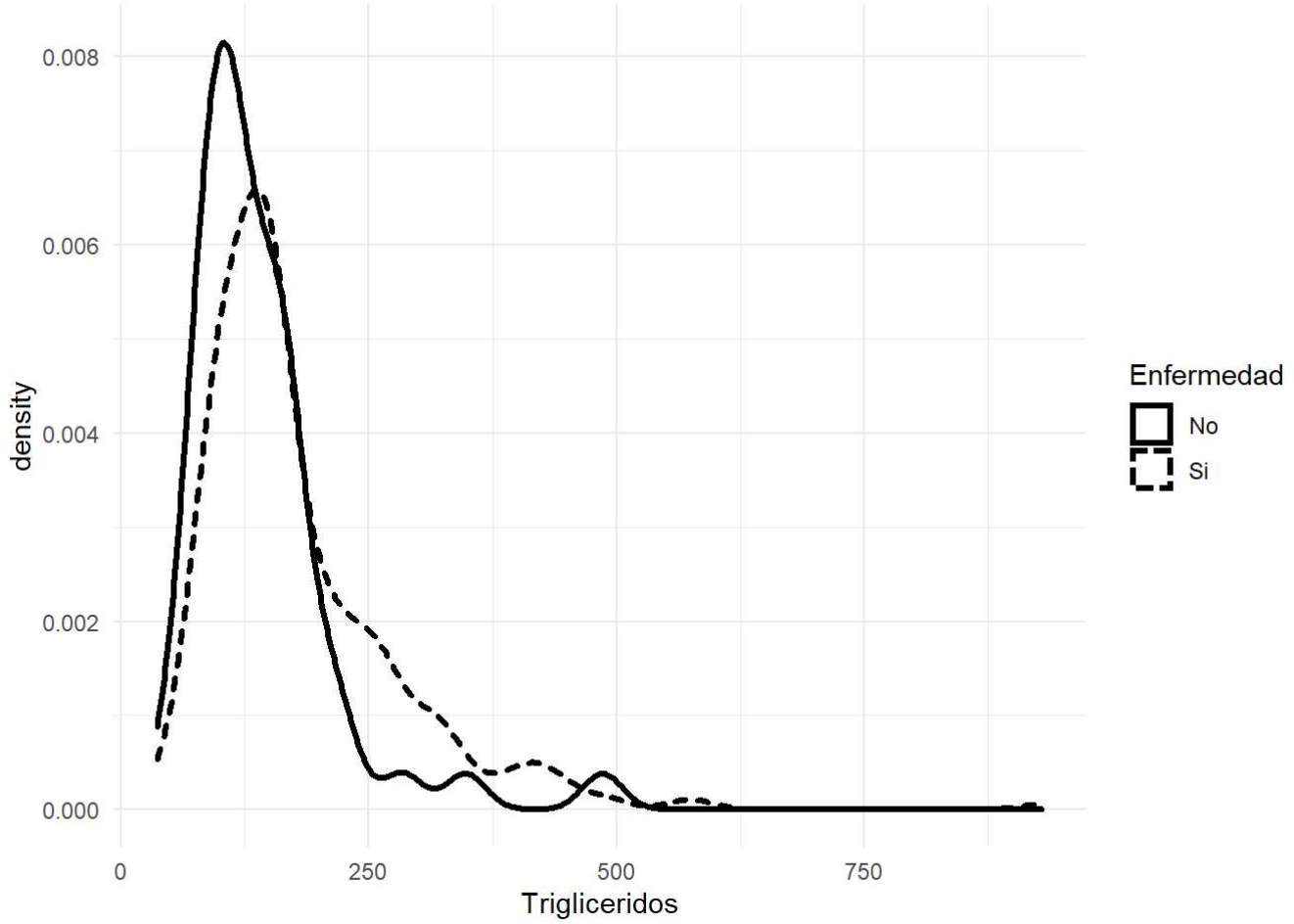
```
ggplot(lipidos) +
  geom_density(aes(x = Trigliceridos),
               size = 1.1) +
  theme_minimal()
```




```
ggplot(lipidos) +
  geom_density(aes(x = Trigliceridos),
               size = 1.1, bw = 10) +
  theme_minimal()
```

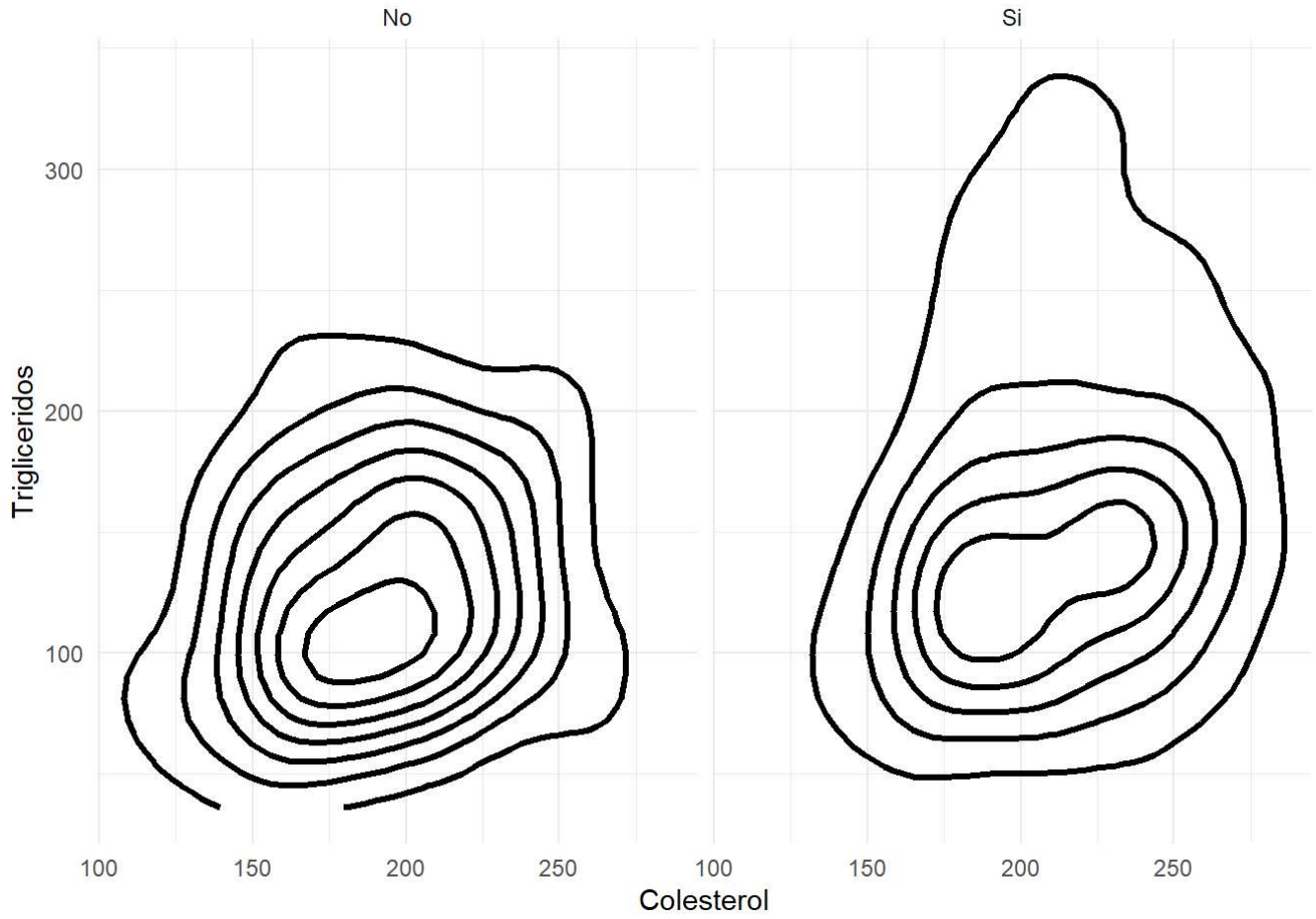



```
ggplot(lipidos) +
  geom_density(aes(x = Trigliceridos, linetype = Enfermedad),
               size = 1.1) +
  theme_minimal()
```



- Representa un estimador del núcleo bidimensional para el vector de variables correspondiente a las concentraciones de triglicéridos y colesterol, tratando por separado los datos de los individuos enfermos y los sanos. Comenta el resultado.

```
ggplot(lipidos, aes(x=Colesterol, y = Trigliceridos)) +
  # geom_point(alpha = 0.3) +
  geom_density_2d(size = 1.1, col = 'black') +
  facet_wrap( ~ Enfermedad) +
  theme_minimal()
```



Considera una variable aleatoria con distribución beta de parámetros $\alpha = 3, \beta = 6$.

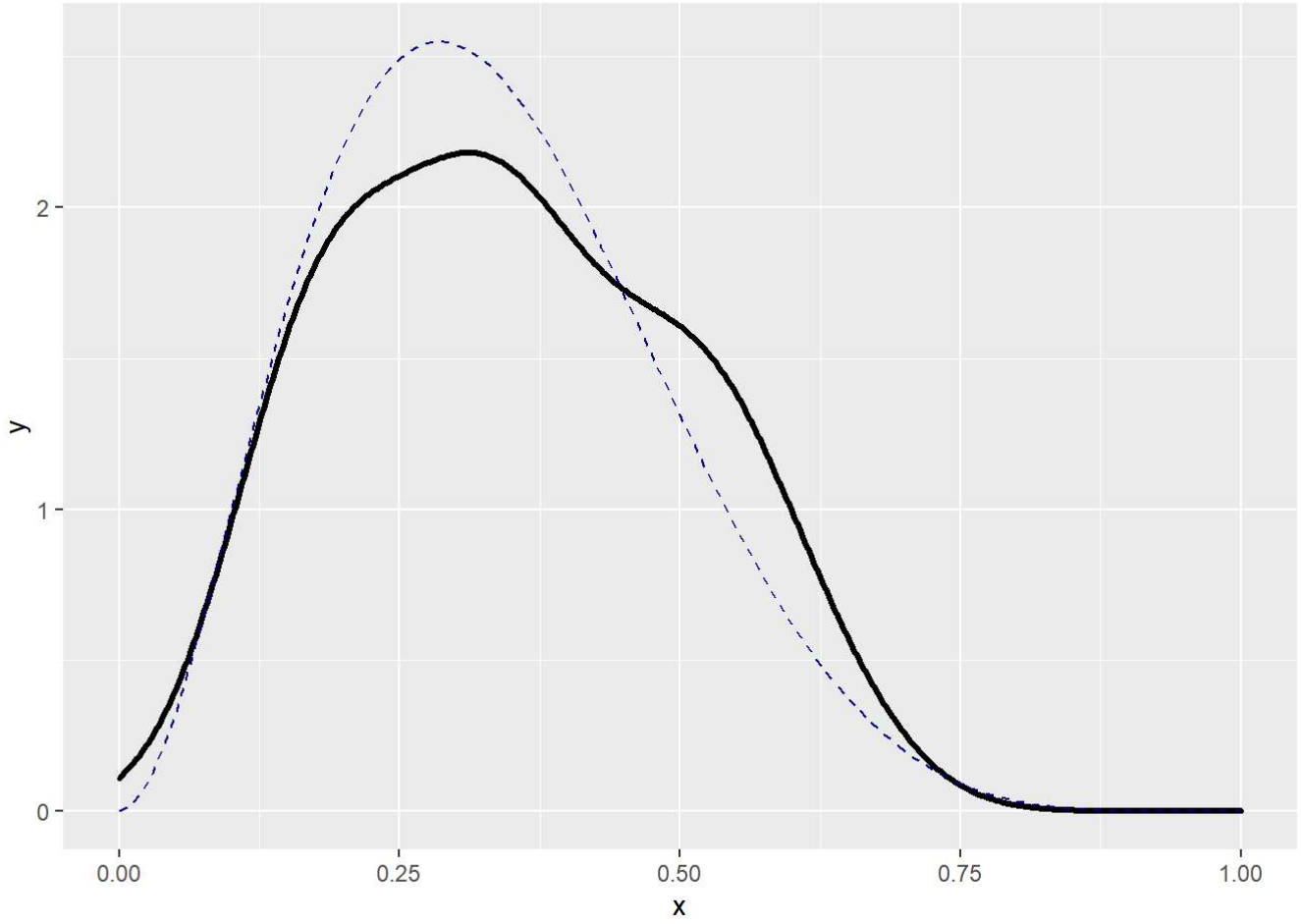
- Representa gráficamente la función de densidad y la función de distribución.
- Simula una muestra de tamaño 20 de esta distribución. A continuación, representa en los mismos gráficos del apartado (a) las estimaciones de F y f obtenidas respectivamente mediante la función de distribución empírica F_n y un estimador del núcleo $\hat{f}$ obtenidos a partir de la muestra simulada.
- Verifica empíricamente el grado de aproximación alcanzado en las estimaciones de F y f. Para ello, genera 200 muestras de tamaño 20 y para cada una de ellas evalúa el error (medido en la norma del supremo, es decir, el máximo de las diferencias entre las funciones) cometido al aproximar F por F_n y f por $\hat{f}$. Por último, calcula el promedio de los 200 errores obtenidos.

Funciones de densidad y de distribución

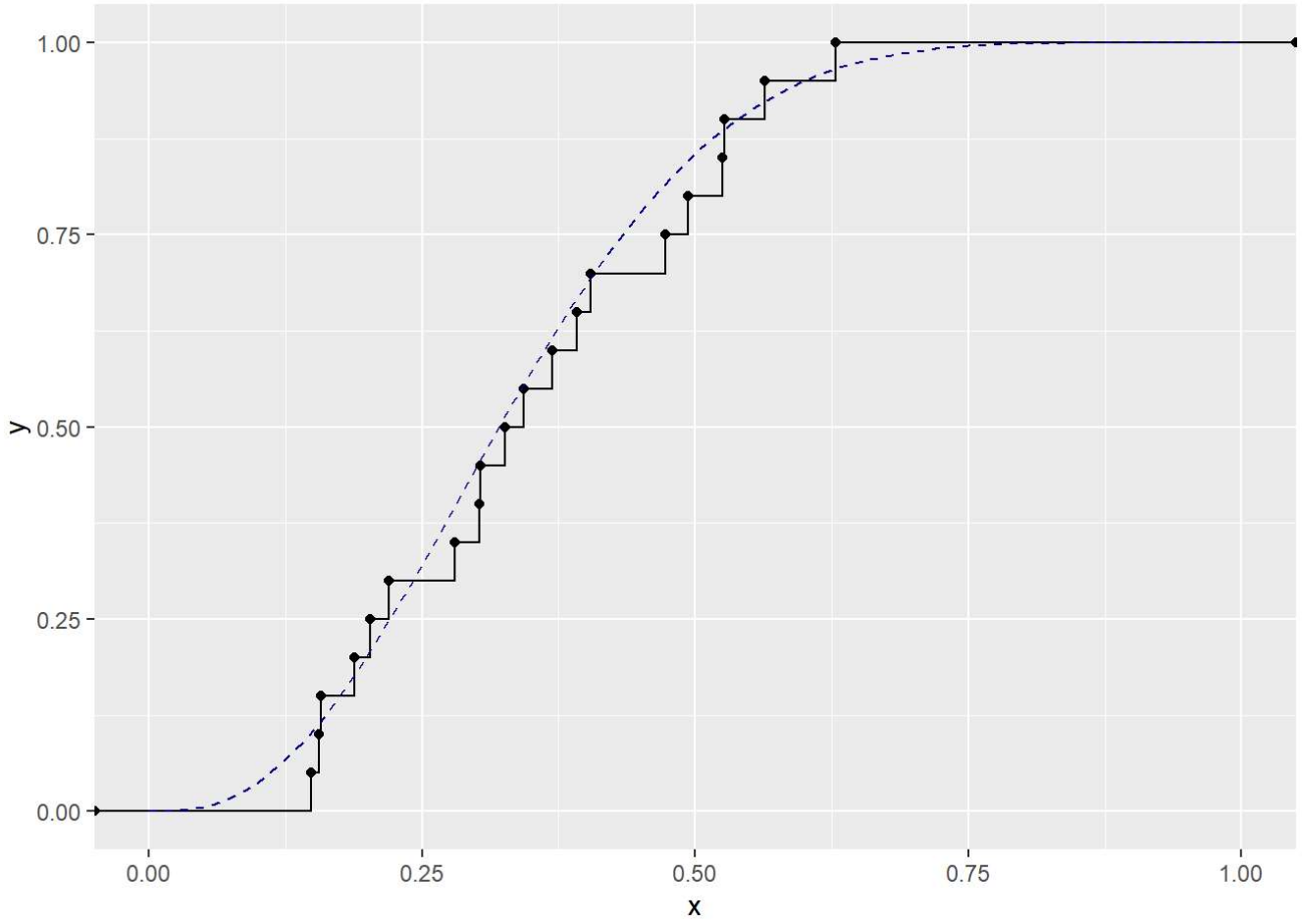
```
set.seed(1123)

a <- 3
b <- 6
n <- 20
datos <- data.frame(x = rbeta(n, a, b))

# Función de densidad
ggplot(datos) +
  geom_density(aes(x), size = 1.1) +
  geom_function(fun = dbeta, args = list(shape1=a, shape2=b), linetype = 2, col = 'darkblue')
+
  xlim(0, 1)  # fija el rango de las x
```




```
# Función de distribución
ggplot(datos) +
  stat_ecdf(aes(x)) +
  stat_ecdf(aes(x), geom = 'point') +
  geom_function(fun = pbeta, args = list(shape1=a, shape2=b), linetype = 2, col = 'darkblue')
+
  xlim(0, 1)
```



Grado de aproximación

```
num_muestras = 200

Supremo1 = rep(0, num_muestras)
Supremo2 = rep(0, num_muestras)

for (i in 1:num_muestras){
  datos <- rbeta(n, a, b)
  est_nucleo <- density(datos)
  densidad <- dbeta(est_nucleo$x, a, b)
  Supremo1[i] <- max(abs(est_nucleo$y - densidad))
  empirica <- ecdf(datos)
  t <- seq(0, 1, 0.01)
  Supremo2[i] <- max(abs(empirica(t) - pbeta(t, a, b)))
}

error_densidad <- mean(Supremo1)
error_distribucion <- mean(Supremo2)
print(error_densidad)
```

```
## [1] 0.8318475
```

```
print(error_distribucion)
```

```
## [1] 0.1770916
```

```
df <- data.frame(errores_densidad = Supremo1,
                 errores_distribucion = Supremo2)

densidad <- ggplot(df) +
  geom_histogram(aes(x = errores_densidad), col = 'black', fill = 'lightblue', bins=10) +
  geom_vline(xintercept = error_densidad, size = 1.1) +
  labs(y = NULL)

distribucion <- ggplot(df) +
  geom_histogram(aes(x = errores_distribucion), col = 'black', fill = 'lightblue', bins=10) +
  geom_vline(xintercept = error_distribucion, size = 1.1) +
  labs(y = NULL)

densidad + distribucion
```