Introducción a la ciencia de datos

Estimación de probabilidades

El objetivo de esta segunda parte de la materia es ofrecer un primer acercamiento a la estimación de la mayoría de los elementos vistos en los 8 capítulos anteriores, tales como probabilidades, esperanza, varianza, funciones de densidad y distribución y función de regresión basándonos en datos, tanto reales como simulados. También se busca reforzar el tema de Simulación, con más uso de la computadora, lo cual permite realizar simulaciones que ayudan a observar y terminar de entender los conceptos teóricos desarrollados a lo largo de la materia. La herramienta que utilizaremos en esta segunda parte será R. Aquí desarrollamos un pequeño apunte con los temas centrales, que deben ser complementados con material bibliográfico, se recomienda como material el libro https://mathstat.slu.edu/~speegle/_book/.

Ya hemos estudiado bien a fondo toda la teoría de probabilidades. Lo que buscamos ahora es poder, en base a la simulación de experimentos, estimar probabilidades. ¿Cuál es la probabilidad de que al tirar el dado que tengo en mi mano, salga el número 1? Yo no se si mi dado es equilibrado, lo que se me ocurre es tirarlo muchas veces, pero muchas veces, y tendré una idea de la probabilidad que busco calculando la frecuencia relativa con la cual observo el valor 1, esto es, cuento la cantidad de veces que observé el valor 1 y lo divido por la cantidad de veces que tiré el dado. ¿Tiene sentido? Si, esta es la forma en la que estimaremos probabilidades. La Ley de los grandes números nos da una herramienta para poder entender porqué esta será una buena forma de estimar una probabilidad. Desarrollemos un poco más la idea.

La ley de los grandes números dice que si tenemos $\{X_n\}_{n\geq 1}$ una sucesión de variables aleatorias independientes e idénticamente distribuidas, tales que $E(X_i) = \mu$, $var(X_i) = \sigma^2, \forall i$. Entonces si consideremos la sucesión de variables $\{\bar{X}_n\}_{n\geq 1}$ con $\bar{X}_n$ el promedio de las primeras n variables, y con $E(\bar{X}_n) = \mu_i$

entonces se tiene que

$P(|\bar{X}_n - \mu| > \epsilon) \overset{n\to\infty}{\to} 0$

Gracias a la desigualdad de Markov, podemos probar la ley débil de los grandes números, que lo que dice en términos de convergencia es que a medida que n aumenta y promediamos más variables, la probabilidad de que $\bar{X}$ se aleje de μ en más de un ϵ es casi cero, y tiende a cero cuando n tiende a infinito.\

Esta ley permite fundamentar el concepto de probabilidad de un evento

Si definimos $X_i = \mathbf{1}\{\text{en el experimento i ocurre el evento A}\}$, tenemos que $E(X_i) = P(A)$, $var(X_i) = P(A)(1 - P(A)) < \infty$ Como además las X_i son independientes, por la ley de los grandes números

$\bar{X}_n \to E(X_1) = P(A)$

Observemos que $\bar{X}_n$ es la frecuencia relativa de ocurrencia del evento **A** en n realizaciones independientes del experimento, ya que las variables X_i eran variables de bernpulli, y solo pueden tomar valores 0 y 1. Por lo tanto tenemos que la frecuencia relativa converge a la probabilidad del evento. Utilizando este resultado, podemos justificar que para un n considerablemente grande, calcular la frecuencia relativa con la que ocurre un evento es una buena aproximación para la probabilidad de dicho evento.

Para poder probar este resultado, es muy útil saber simular experimentos y poder ver que ocurre a medida que la cantidad de simulaciones aumenta.Simularemos experimentos, para lo cual necesitaremos un manejo mas o menos bueno de R, para lo cual es sumamente recomendable resolver los ejercicios del primer capítulo del libro de Speegle. Para poder simular un experimento, es escencial entenderlo bien, porque le voy a tener que explicar a la computadora como debe realizarlo.

Una de las ventajas de R es que podremos estimar probabilidades usando simulaciones.

Veamos un ejemplo: ¿Cómo simularíamos el tiro de un dado? Si tenemos un dado equilibrado de 6 caras, sabemos que hay seis valores posibles con igual probabilidad. Sería equivalente a tener una urna con seis

bolitas con los números del 1 al 6 y realizar extracciones con reposición. Entonces podemos elegir dos caminos: o usar la distribución uniforme y simular tal como lo aprendimos en la guía 2, o usar la función `salmple()`.

```
tiro_dado<-sample(1:6,size=1,replace=TRUE)
tiro_dado
```

```
## [1] 6
```

De esta manera, simulé el tiro del dado una vez. ¿Y si quiero tirarlo 8 veces? Facilísimo!

```
tiro_dado<-sample(1:6,size=8,replace=TRUE)
tiro_dado
```

```
## [1] 2 3 5 6 1 2 3 4
```

Hasta me puedo poner elegante y definir una función que dado un valor n, simule n tiros de un dado

```
tiro_dado<-function(n)
{
  D<-sample(1:6,size=n,replace=TRUE)
  return(D)
}

tiro_dado(10)
```

```
##  [1] 6 2 5 6 2 6 5 4 6 6
```

Observemos que devuelve lo que salió en cada tiro. Ya sabés que hacer cuando abras la caja de un juego de mesa y no tengas dados para jugar...

Una vez que tengo los valores al tirar el dado, puedo estimar la probabilidad de eventos simplemente calculando frecuencias relativas. Para tal fin, podemos usar vectores lógicos.

```
# A: sale el 1
# Estimamos la P(A)

fr_A<- sum(tiro_dado(10)==1)/10

# o promediando ceros y unos

fr_A<-mean(tiro_dado(10)==1)

# La funcion mean da el promedio
```

Sigamos. Ahora quiero estimar la probabilidad de que la suma de dos tiros del dado sea 8. Pensemos primero como haríamos para realizar el experimento, porque esa será la forma de que nos salga luego explicárselo a la computadora. Tengo que tirar el dado dos veces, sumar los resultados y ver si la suma es 8 o no. Si es 8, entonces tuve un éxito en ese ensayo, si no es 8 entonces no será éxito. Tengo que repetir ahora muchas veces ese mismo experimento, y finalmente, la estimación de la probabilidad que quiero calcular será la frecuencia con la que conseguí el éxito sobre la cantidad total que realicé mi experimento. Esto es: estimo la probabilidad usando la frecuencia relativa. A ver como lo hacemos.

```
# Una realización del experimento:

tiro_dado(1)+tiro_dado(1)
```

```
## [1] 8
```

```
# Lo que hice recién es tirar el dado dos veces y sumar los resultados. O también
```

```r
sum(tiro_dado(2))
```

```
## [1] 8
```

```r
# Son dos formas de hacer lo mismo.
```

Lo que quiero ahora no es saber cuanto vale la suma, es ver si es 8 o no. Para esto usamos vectores lógicos

```r
sum(tiro_dado(2))==8
```

```
## [1] FALSE
```

Con el == le pregunro a R si son iguales, y me devuelve TRUE o FALSE. No te olvides que al TRUE lo considera como 1 y a FALSE como 0, entonces si ahora repito, por ejemplo, $n = 100$ veces el experimento, tendré una tira de 1 y 0, dependiendo si la suma es 8 o no.

```r
resultados <- c() # Genero un vector vacio para llenarlo con mis resultados
for(i in 1:100)
{
  resultados[i]<-sum(tiro_dado(2))==8
}
```

Ahora, mi vector tiene los resultados de mis 100 experimentos, para estimar la probabilidad tengo que contar cuantas veces ocurrió éxito y dividirlo por 100.

```r
proba_estimada<-sum(resultados)/100
proba_estimada
```

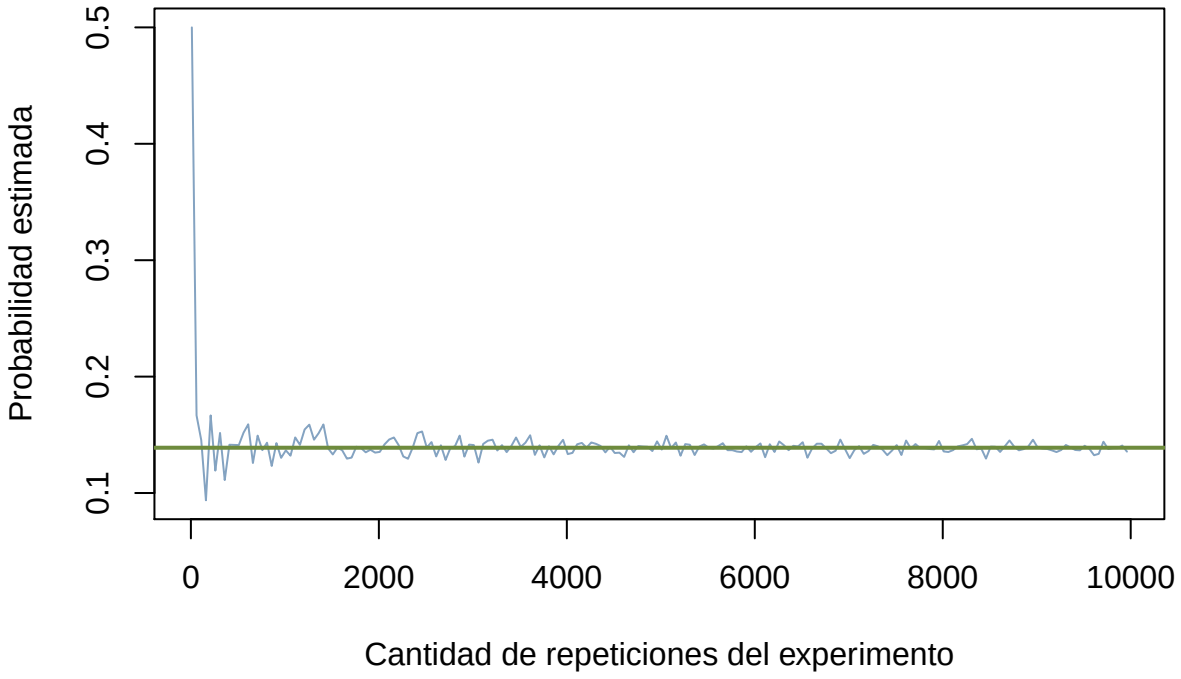
```
## [1] 0.16
```

Listo. Estimamos nuestra probabilidad. Ahora la parte interesante. Nosotros a esta altura sabemos perfectamente cuanto vale la probabilidad de que al sumar dos tiros del dado resulte 8, es una cuenta fácil de hacer, entonces podemos ver si nuestra estimación es buena. Más aun, podemos terminar de entender lo que interpretamos anteriormente a partir de la ley de los grandes números al decir que si repetimos muchas muchas veces el experimento, la frecuencia relativa va a tender a una constante a la cual llamamos probabilidad del evento. Eso es lo que estamos usando ahora. Cuantas más veces repita el experimento, mas me estaré acercando a la probabilidad que quiero calcular. Y ¿porqué lo hago para una probabilidad que se calcular? Porque quiero entender la idea, para luego usarla para estimar probabilidades que son muy difíciles de calcular, simplemente simulando el experimento y calculando frecuencias relativas. Pero eso sí, tenemos que aprender muy bien a simular.

Comprobamos. Sabemos que $P(D_1 + D_2 = 8) = 5/36$. Voy ahora a simular mi experimento para un valor de n cada vez más grande, para n=10,20,30,....,10000. Y voy a graficar la estimación de la probabilidad para cada uno de los valores de n. Luego voy a superponer con una línea horizontal el verdadero valor de la probabilidad estimada. ¿Qué te parece que vamos a observar?

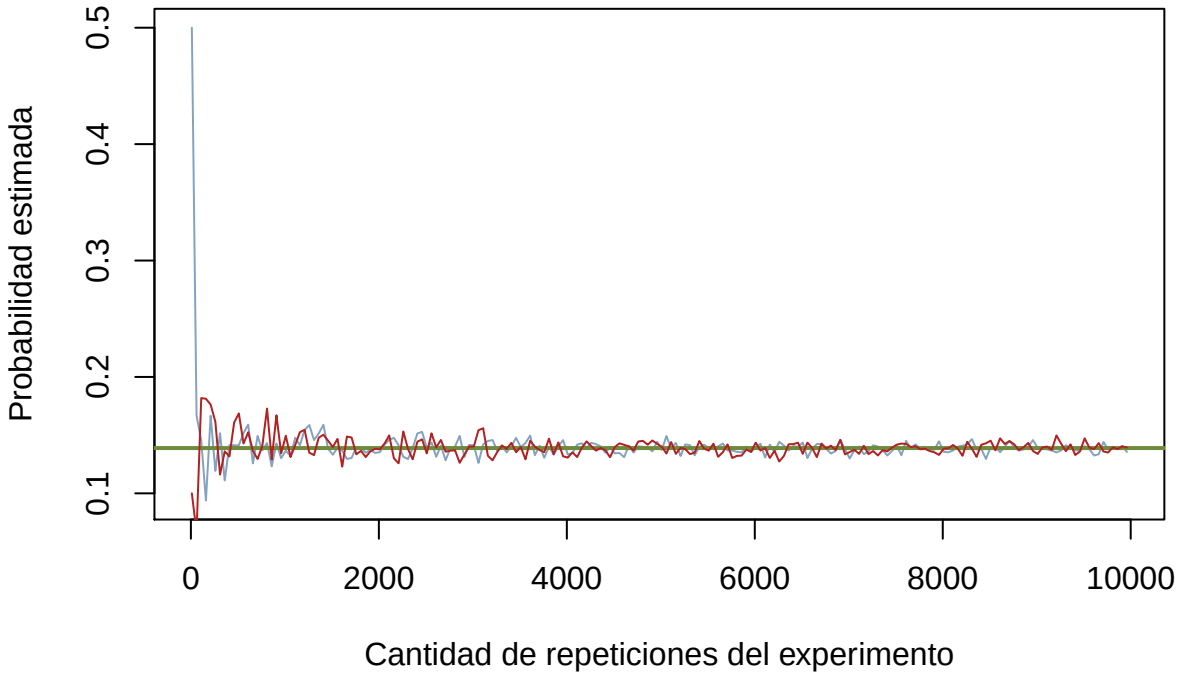
```r
Probabilidades<-c() # En este vector voy a guardar todas mis probabilidades

n<-seq(10,10000,50) # Me construyo un vector con todos los valores de n

for (j in 1:length(n))
{
  resultados <- c() # Genero un vector vacio para llenarlo con mis resultados
  for(i in 1:n[j])
    {
       resultados[i]<-sum(tiro_dado(2))==8
  }
  Probabilidades[j]<-sum(resultados)/n[j]
}
```

```r
plot(n,Probabilidades,col=rgb(0.2,0.4,0.6,0.6), pch=20, type = "l",xlab = "Cantidad de repeticiones del
     ylab= "Probabilidad estimada")
abline(h=5/36,col= "darkolivegreen4",lwd=2)
```



Podemos observar que a medida que la cantidad de repeticiones del experimento crece, la probabilidad estimada se acerca al valor verdadero de la probabilidad que se busca estimar.

Si lo hacemos nuevamente, simulando, vamos a obtener diferentes resultados, pero podremos observar la misma tendencia.



A partir de esta misma idea, podremos simular experimentos mucho más complicados. Pero cuidado, como ya

uno conoce la teoría de probabilidades,se ve tentado de usar las fórmulas que ya sabe que funcionan. Ahora lo que buscamos es simular el experimento para estimar la probabilidad como si no supieramos calcularla, solo usando la frecuencia relativa: tratando de reproducir el experimento, contando cantidad de veces que ocurre el evento a observar y dividirlo por la cantidad de veces que realizo el experimento. Si es posible hacer el cálculo exacto, se podrá comparar el valor simulado con el verdadero. Y para los casos en los que las probabilides sean muy difíciles de calcular, tendremos una muy buna herramienta para poder calcular un valor cercano al verdadero.

Variables aleatorias

En este segundo capítulo, buscaremos aprender a estimar la función de probabilidad, la función de densidad y función de distribución de una variable aleatoria X. A partir de ello también entenderemos como estimar la función conjunta de vectores aleatorios, usando la misma idea. En el capítulo 2 de la primera parte de la materia, ya estudiamos histograma y función empírica, que son las formas más simples de estimación de una función de densidad y función de distribución de una variable aleatoria. Volveremos a ello, ahora entendiendo un poco más lo que buscábamos con esas funciones, y veremos algunas formas más de estimacion.

Variables aleatorias discretas.

Ya aprendimos a estimar probabilidades de eventos, y eso es lo único que necesitamos para estimar una función de probabilidad. Si defino X como el valor observado al arrojar un dado equilibrado, encontrar su función de probabilidad consiste en econtrar la probabilidad de cada uno de los valores de su rango (o soporte). De la misma forma, estimar la función de probabilidad consistirá en estimar la probabilidad de cada uno de los valores posibles. ¿Cómo? Realizando (o simulando) muchas veces el experimento y calculando la frecuencia relativa para cada uno de los valores posibles. No olvidemos que las reglas que debe cumplir una función de probabilidad también deberá cumplirlas su función estimada.

```
set.seed(27)
# Seteo la semilla de trabajo cada vez que voy a comenzar un ejercicio de simulacion. El
# numero que elijo no es importante, siempre que elija el mismo para conseguir los mismos
# resultados del experimento la próxima vez que quiera simularlo

n <- 10000 #Realizo muchas simulaciones para que mi estimación sea mejor
tiros <- sample(1:6,n,replace = TRUE) # Simulo n tiros del dado
```

Una función muy útil es la función `table()` que lo que hace es una tabla en la que cuenta la cantidad de veces que se repite un valor en un vector dado. Es justo lo que necesitamos! Buscamos contar cuantas veces aparece cada valor, y dividir esa cantidad por n, y ese resultado será la frecuencia relativa para cada uno de los valores de X.

```
p_X_estimada<-table(tiros)/n
p_X_estimada
```

```
## tiros
##      1      2      3      4      5      6
## 0.1703 0.1673 0.1652 0.1687 0.1625 0.1660
```

Si buscara por ejemplo la probabilidad $P(X < 4)$, esa estimación resultaría de sumar los valores sobre la función de probabilidad estimada, siendo $\hat{P}(X < 4)$ (usamos el sombrero arriba de la P para decir que el valor es estimado):

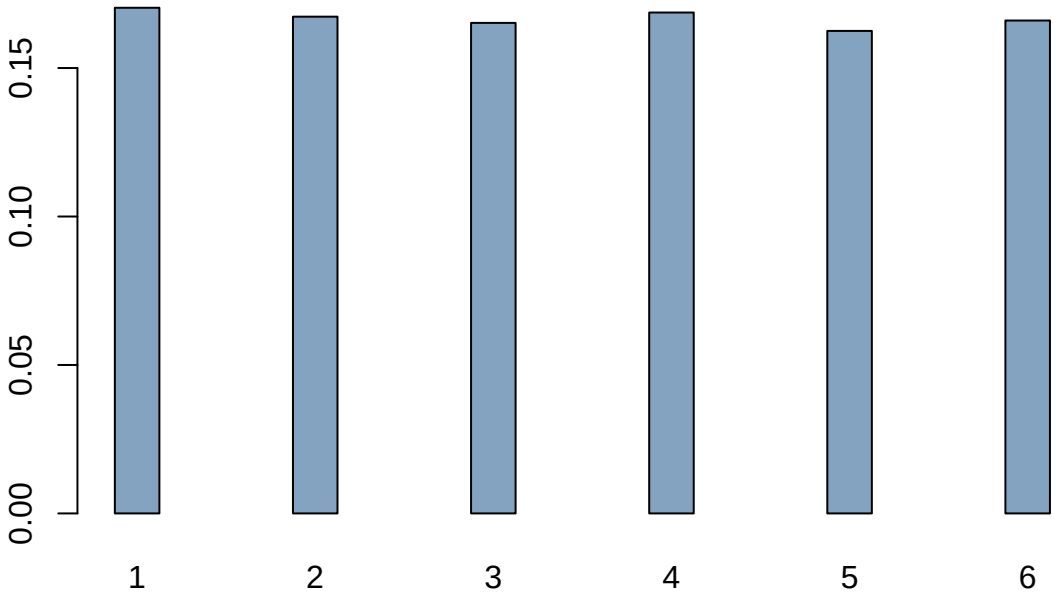
```
prob_estimada<- sum(p_X_estimada[1:3])
prob_estimada
```

```
## [1] 0.5028
```

No esta mal, siendo que el valor real es 0.5

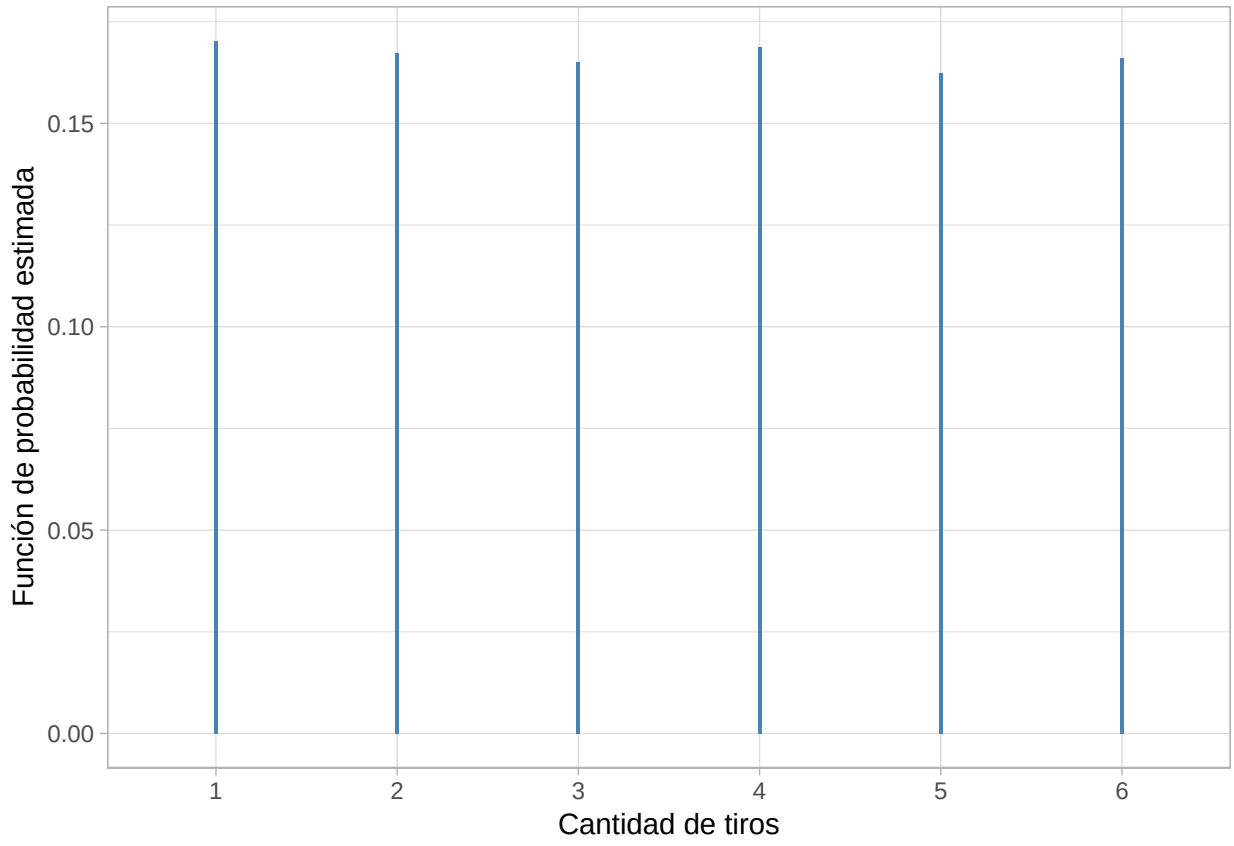
Finalmente, me interesa graficar esta función. Siempre nos interesa graficar y poder a partir del gráfico concluir muchísimas cosas. Los gráficos de las funciones de probabilidad estimada se realizan con gráficos de barras, o *bar plot*.

```
barplot(p_X_estimada,col=rgb(0.2,0.4,0.6,0.6),space=3)
```



O usando la libreria ggplot2 (próximamente video)

```
library(ggplot2)
tabla<- as.data.frame(p_X_estimada)
ggplot(data=tabla, aes(x=tiros, y=Freq)) +
  geom_bar(stat="identity", width=0.02, fill = "steelblue")+
  labs(x="Cantidad de tiros", y="Función de probabilidad estimada")+
  theme_light()
```



Las probabilidades efectivamente se encuentran entre 0 y 1, y observamos una gráfica muy similar a la de la distribución uniforme, como debería ser.

Variables aleatorias continuas.

En este caso, buscaremos estimar la función de densidad de la variable aleatoria, y usar esa estimación para estimar probabilidades. La forma mas simple para estimar a una función de densidad es el histograma.

El histograma es la forma no paramétrica mas común para estimar $f(x)$. La construcción es bastante simple. Dada una muestra de datos $x_1, x_2, \ldots, x_n$, que se asumen realizaciones de las variables aleatorias $X_1, X_2, \ldots, X_n$, todas con distribución $f(x)$ e independientes, se realizan los siguientes pasos:

- Se selecciona un origen x_0 y se divide la recta real en intervalos de longitud h

  $B_j = [x_0 + (j-1)h, x_0 + jh], j \in \mathbb{N}$

  No es necesario que todos los intervalos tengan la misma longitud, pero es recomendable que así sea. Esto facilita la lectura.

- Se cuenta cuantas observaciones caen en cada intervalo armando una tabla de frecuencias. Denotamos a la cantidad de observaciones que caen en el intervalo j como n_j

- Para cada intervalo, se divide la frecuencia absoluta por la cantidad total de la muestra n (para convertirlas en frecuencias relativas, análogo a como se hace con las probabilidades) y por la longitud h (para asegurarse que el área debajo del histograma sea igual a 1):

  $f_j = \frac{n_j}{nh}$

- Se grafica el histograma realizando una barra vertical sobre cada intervalo con altura f_j y ancho h

Formalmente, el histograma está dado por:

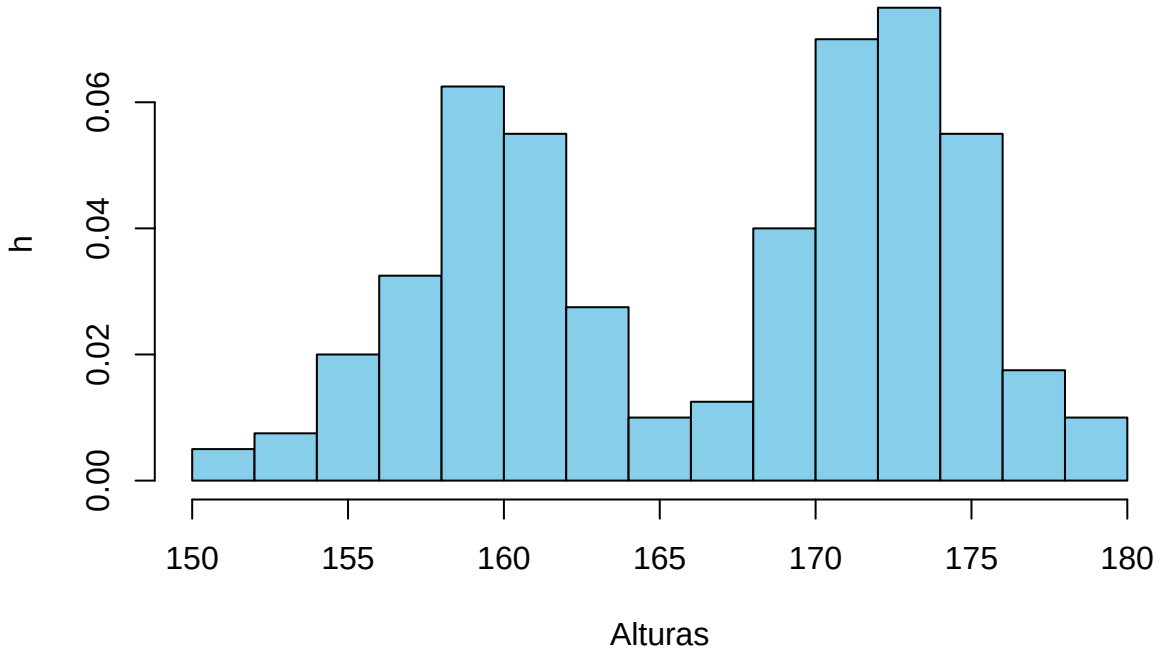
$$\hat{f}_h(x) = \frac{1}{nh} \sum_{i=1}^{n} \sum_{j} 1(x_i \in B_j) 1(x \in B_j)$$

Si m_j es el centro del intervalo j, podemos escribir $B_j = [m_j - \frac{h}{2}, m_j + \frac{h}{2})$.

Se puede verificar fácilmente que el área del histograma es igual a 1, propiedad que se requiere para cualquier estimador razonable de una función de densidad.

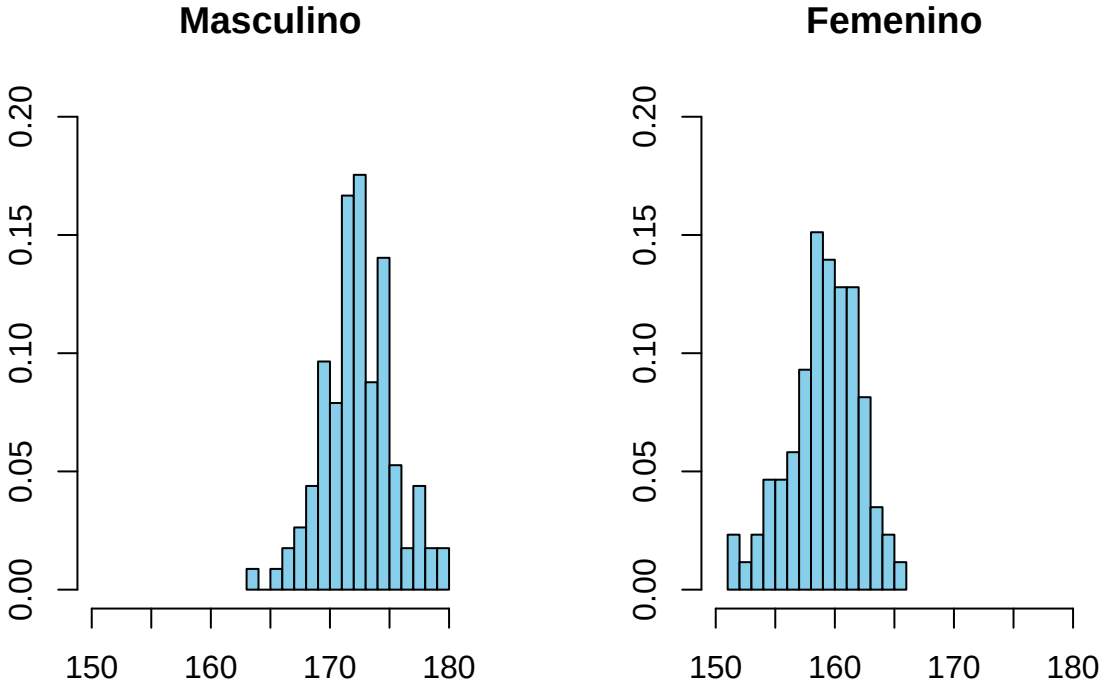
En R, es muy fácil hacer histogramas porque existen funciones hechas para ello. Veamos un ejemplo. Tenemos datos de mediciones de alturas de 200 personas, su género, la altura de su madre y su contextura física. Supongamos que queremos entender como se distribuye la variable A que mide la altura de las personas. La variable A es una variable continua, entonces para poder comenzar a entender su comportamiento realizamos un histograma. La forma del histograma podría darnos idea de la forma de la función de densidad, ya que es una estimación de la misma, y a partir de ello ver si se parece a alguna de las densidades conocidas.

```
alturas<-read.csv("alturas_n_200.csv")

hist(alturas$altura,breaks=15, freq = FALSE, col="skyblue",
     xlab="Alturas", ylab="h", main=" ")
```



Me da una idea de la densidad de la variable A. En principio al observar el histograma no veo que se parezca a una distribución conocida. Pero, ¿Qué pasa si separo en hombres y mujeres?

```
par(mfrow=c(1,2)) #le digo que parta en 1 fila y dos columnas la zona del grafico
hist(alturas$altura[alturas$genero=="M"],freq=FALSE,breaks=15,
     xlim=c(150,180),ylim = c(0,0.2), main = "Masculino", col="skyblue", xlab="", ylab="")
hist(alturas$altura[alturas$genero=="F"],freq=FALSE,breaks=15,
     xlim=c(150,180),ylim = c(0,0.2),main="Femenino", col="skyblue", xlab="", ylab="")
```



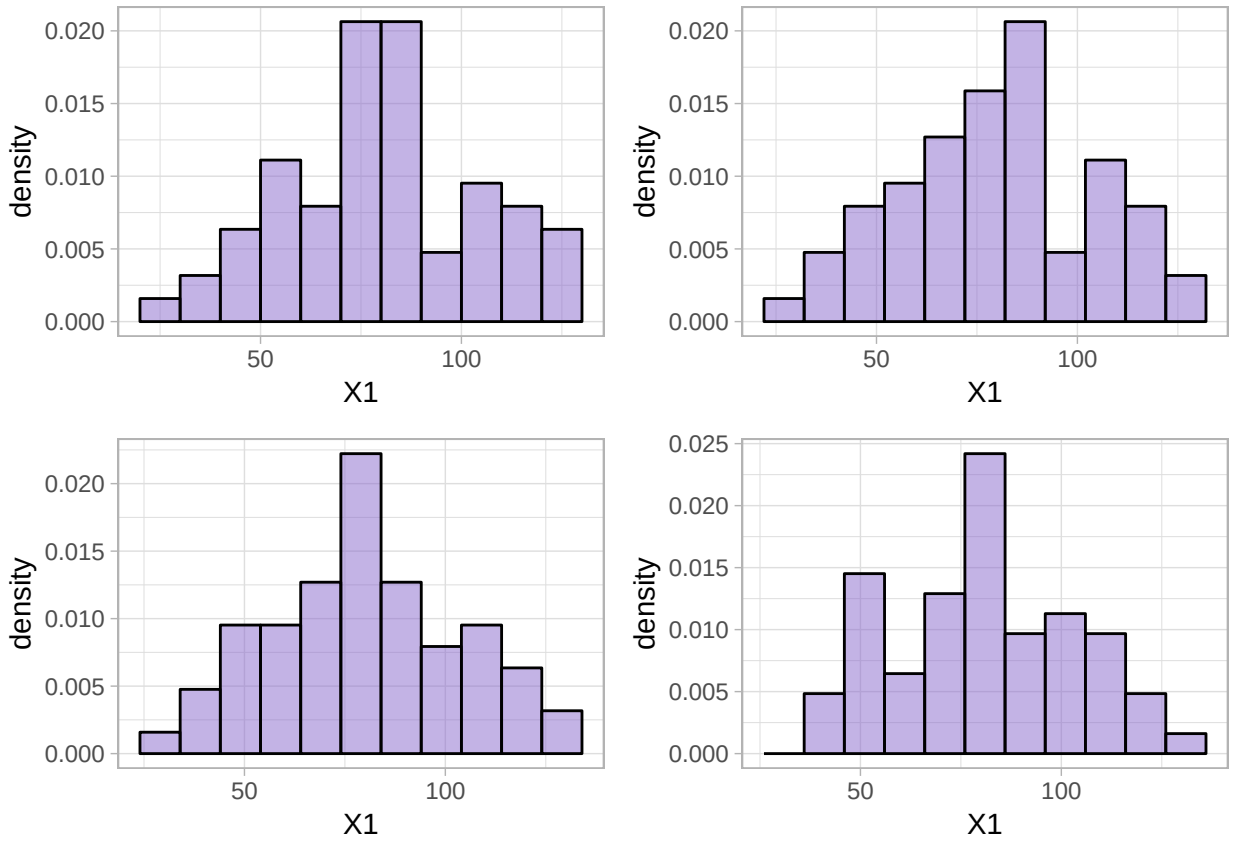
```r
par(mfrow=c(1,1)) #vuelve a una sola parte
```

Ahora sí, podríamos observar formas conocidas y a partir de ello empezar a suponer posibles distribuciones para las variables en estudio.

A partir de un histograma podemos estimar una probabilidad calculando áreas, de la misma forma que lo hacemos cuando calculamos probabilidades.

El histograma presenta varias desventajas al momento de estimar una densidad:

1. Depndiendo del ancho del intervalo dará ideas de densidades diferentes
2. Es muy sensible al punto de inicio del primer intervalo. Para un número fijo de intervalos, la forma puede cambiar simplemente moviendo la ubicación del punto inicial, lo vemos en el siguiente gráfico donde solament cambiando el punto de inicio las formas de los histogramas dan muy diferentes.



3. La densidad estimada no es suave, es escalonada, y esta característica no es propia de la densidad que busca estimar si no de la herramienta que usamos para estimarla.

Es por estas razones que el histograma se usa únicamente para tener una primera visualización. Veremos ahora un método de estimación que sí es efectivo y es el utilizado para estos fines. Se basa (nuevamente) en la ley de los grandes números. Esté método, llamado estimador de densidades basado en núcleos, busca estimar f a partir de la cantidad de observaciones que caen en un intervalo alrededor del punto x. Esta estimación resulta en funciones más suaves y precisas. Veamos el desarrollo para llegar a la estimación.

Sabemos que si X es una variable aleatoria continua, entonces

$$P(X \in (x-h, x+h)) = \int_{x-h}^{x+h} f_X(t)dt$$

Por otro lado, por la LGN podríamos decir que

$$P(X \in (x-h, x+h)) \cong \frac{\#\{X_i \in (x-h, x+h)\}}{n}$$

Por otro lado, cuando aprendimos integrales vimos que, para un h suficientemente pequeño

$$\int_{x-h}^{x+h} f_X(t)dt \cong 2hf_X(x)$$

Uniendo todas estas ideas llegamos a que

$$2hf_X(x) \cong P(X \in (x-h, x+h)) \cong \frac{\#\{X_i \in (x-h, x+h)\}}{n}$$

Por este motivo es que se propone como estimador para la densidad a

$$\hat{f}(x) = \frac{\#\{X_i \in (x-h, x+h)\}}{2hn}$$

Se puede probar que esta función es siempre mayor o igual que 0 y su integral para todos los reales vale 1.

Vamos a reescribir la función usando indicadoras.

$$\hat{f}(x) = \frac{1}{hn}\sum_{i=1}^{n}\frac{1}{2}\mathbf{1}\{x-h < X_i < x+h\} = \frac{1}{hn}\sum_{i=1}^{n}\frac{1}{2}\mathbf{1}\{-1 < \frac{x-X_i}{h} < 1\}$$

Si llamamos $k(t) = \frac{1}{2}\mathbf{1}\{-1 < t < 1\}$ resulta

$$\hat{f}_h(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x-x_i}{h}\right).$$

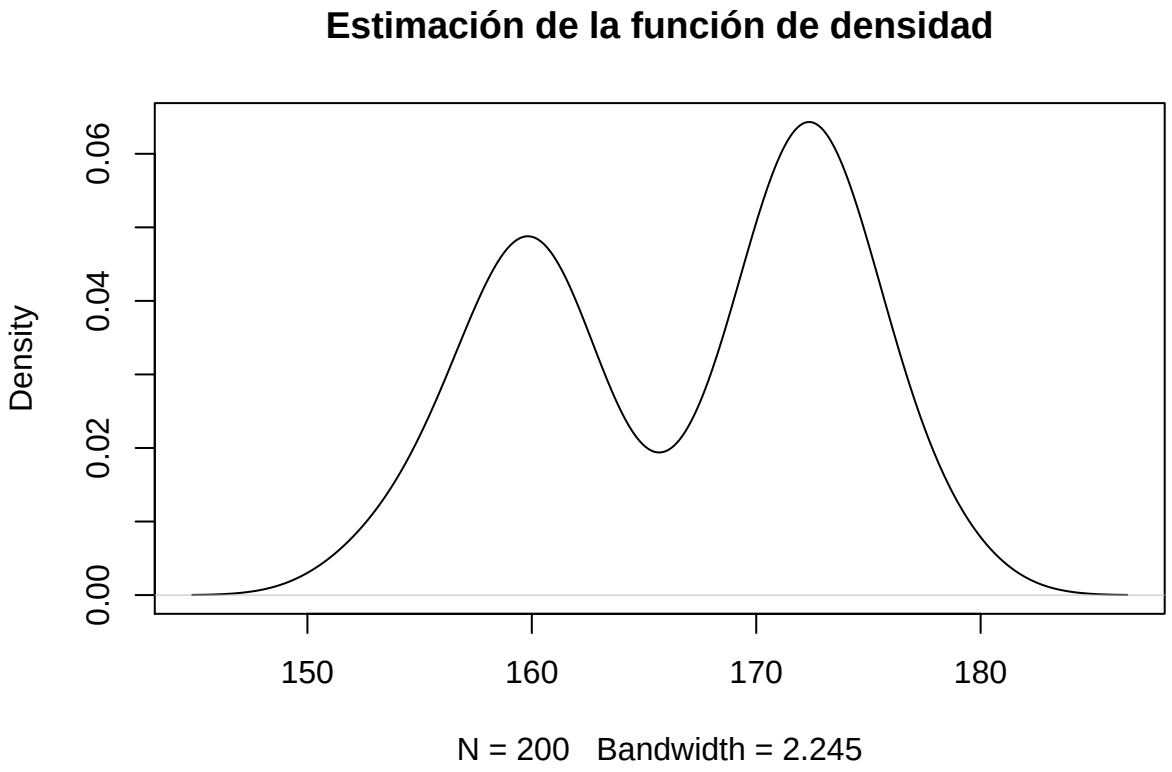
De esta manera la estimación de f se ve coo una suma en la que se usa una función de peso $K : \mathbb{R} \to \mathbb{R}$ llamada *núcleo* (o *kernel*), resultando en el estimador de la densidad por núcleos de la densidad de x de Rosenblatt (1956) y Parzen (1962). A h se lo llama ventana o parámetro de suavizado, y encontrar los valores óptimos de h es la tarea más complicada e importante del método.

A partir de esto, pueden definirse otros núcleos para que la densidad estimada de más *suave*.

En general, las funciones $K(u)$ son funciones de densidad de probabilidad (integran a uno y son mayores o iguales a cero para u en el dominio de K) continuas, acotadas y simétricas con parámetros de suavizado h que ajustan el tamaño y la forma de los pesos cercanos a x, de esta manera $\hat{f}$ hereda muchas propiedades de K. Algunos ejemplos de núcleos son el núcleo Epanechnikov dado por $K(u) = \frac{3}{4}(1-u^2)I(|u| \leq 1)$, el bicuadrado definido como $K(u) = \frac{15}{16}(1-u^2)^2 I(|u| \leq 1)$ y el gausiano $K(u) = \frac{1}{\sqrt{2\pi}}\exp(-u^2/2)$. El histograma sería la versión de estimación basada en núcleos, utilizando un núcleo uniforme.

Nuevamente, en R es muy simple de realizar. Volviendo al ejemplo de las alturas

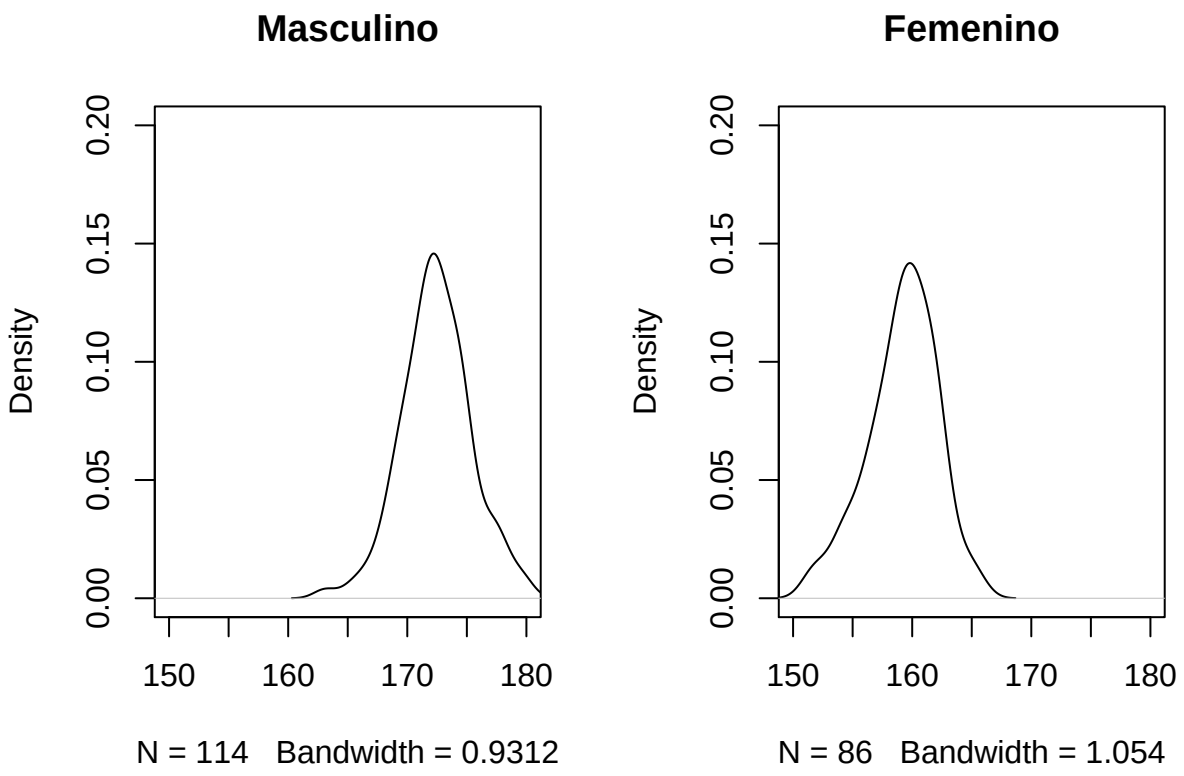
```
plot(density(alturas$altura), main= "Estimación de la función de densidad")
```



Si no se especifica, la función `density()` utiliza el núcleo gaussiano.

Separando por género podemos hallar estimaciones que ayudan a visualizar mejor la posible distribución de la variable en estudio.

```
par(mfrow=c(1,2))
plot(density(alturas$altura[alturas$genero=="M"]),
     xlim=c(150,180),ylim = c(0,0.2), main = "Masculino")
plot(density(alturas$altura[alturas$genero=="F"]),
     xlim=c(150,180),ylim = c(0,0.2),main="Femenino")
```




```
par(mfrow=c(1,1))
```

Función empírica

La Función empírica es una estimación de la función de distribución acumulada de una variable aleatoria. Lo mínimo que puede pedirse a esta función es que cumpla con las condiciones que debe tener una función de distribución:

1. $\hat{F}_X(x) \in [0,1], \forall x \in \mathbb{R}$
2. $\hat{F}_X(x)$ es monótona no decreciente
3. $\hat{F}_X(x)$ es continua a derecha
4. $lim_{x\to-\infty}\hat{F}_X(x) = 0$ y $lim_{x\to\infty}\hat{F}_X(x) = 1$

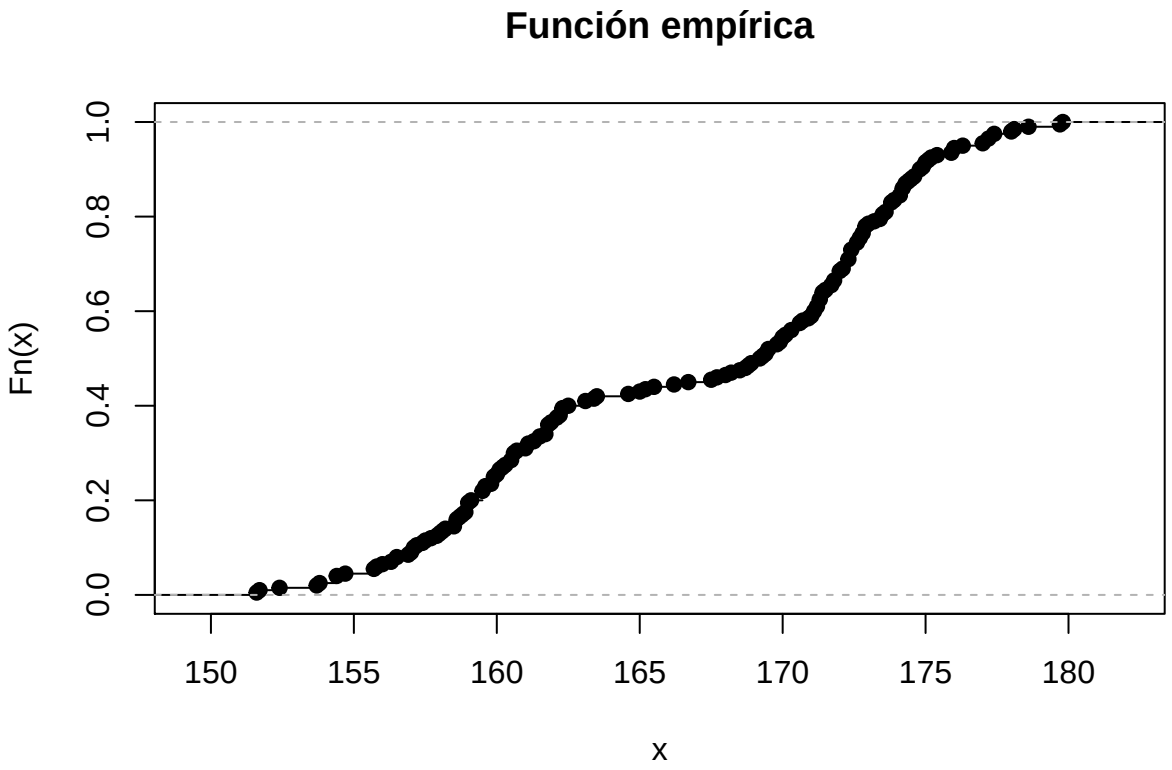
Se define la Función empírica como

$$\hat{F}(x) = \frac{1}{n}\sum_{i=1}^{n} 1(x_i \leq x)$$

Donde $x_1, x_2, \ldots, x_n$ se asumen realizaciónes de las variables aleatorias $X_1, X_2, \ldots, X_n$, todas con distribución $F_X(x)$ e independientes

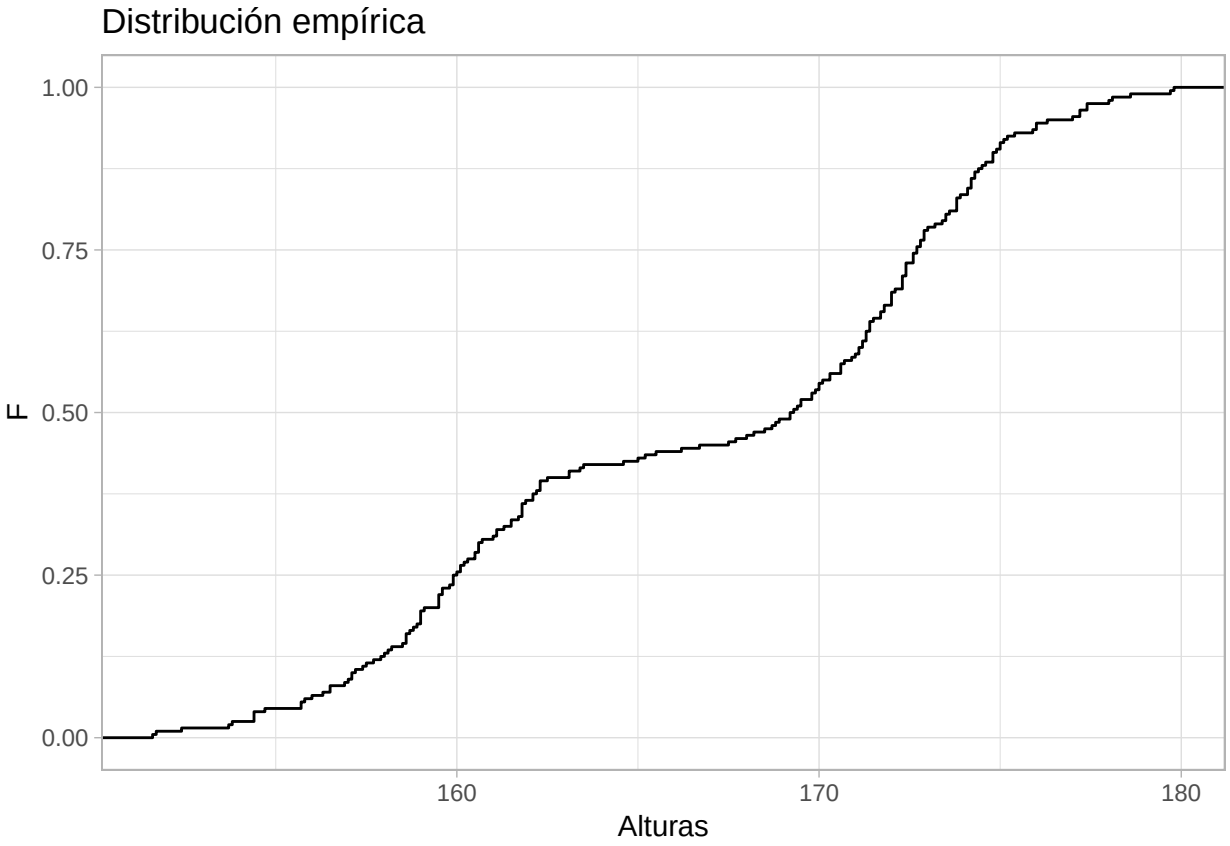
La función empírica nos sirve para entender la distribución de cualquier tipo de variable, así como la función de distribución. En el ejemplo

```
plot(ecdf(alturas$altura), main="Función empírica")
```



Usando ggplot2 (para que den ganas de aprenderlo ya que los gráficos quedan mucho más bonitos)

```
ggplot(alturas, aes(altura))+stat_ecdf(geom = "step")+
  labs(title="Distribución empírica",y = "F", x= "Alturas")+
  theme_light()
```



Nuevamente, cuantas más observaciones tengamos, más se acercaran estas estimaciones a los verdaderos valores de las funciones. Para aprender un poco más y encontrar las demostraciones teóricas, se recomienda el libro de Wolfgang Härdle: Nonparametric & semiparamtric models.

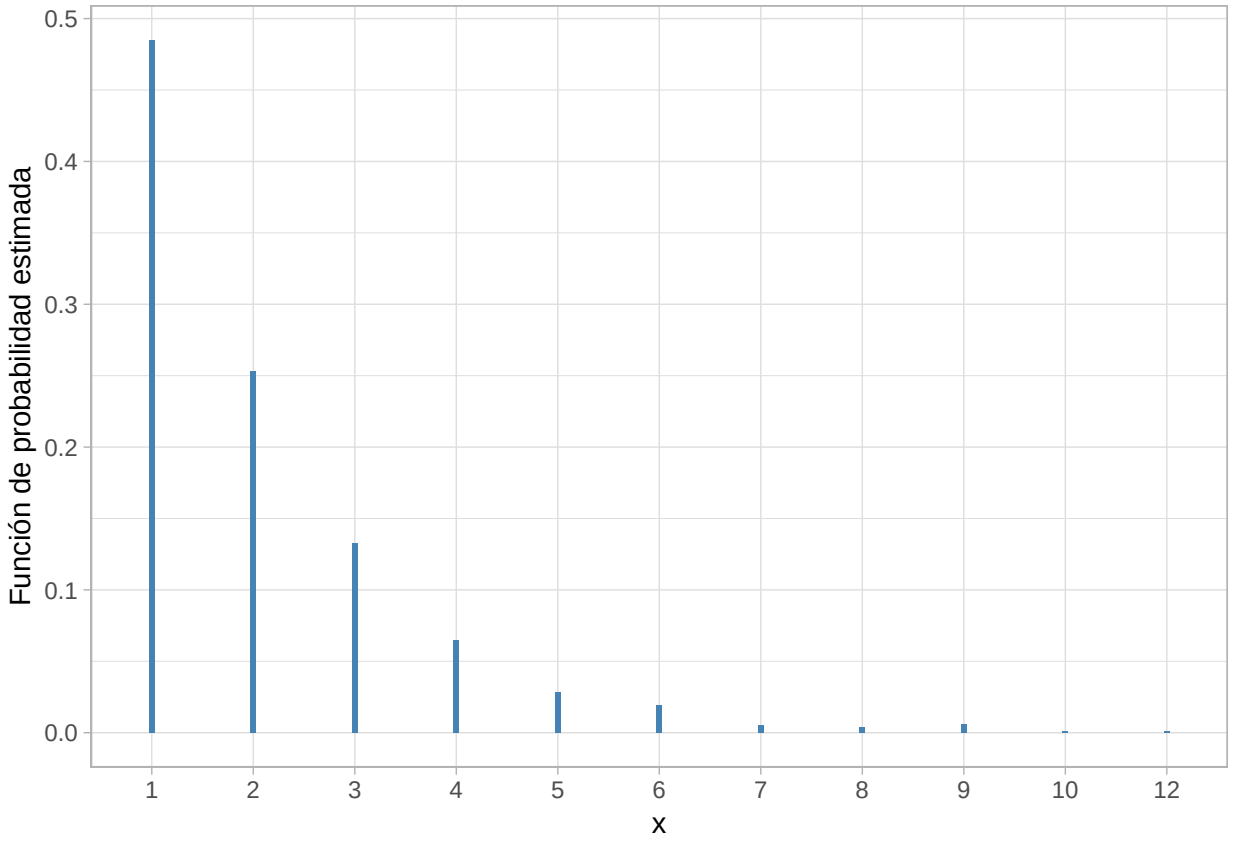
Función de variable aleatoria

Muchas veces buscaremos estimar no la distribución de una variable aleatoria, si no la distribución de una función de dicha variable. Si tenemos una variable aleatoria X de la cual conocemos su distribución, mediante simulaciones resulta muy simple estimar la distribución de la nueva variable. Veamos algunos ejemplos.

Supongamos que $X \sim \mathcal{G}(0.5)$. Podemos tal como lo aprendimos simular muchos valores de X, ya sea utilizando todas las herramientas teóricas que tenemos, o utilizando las funciones de R.

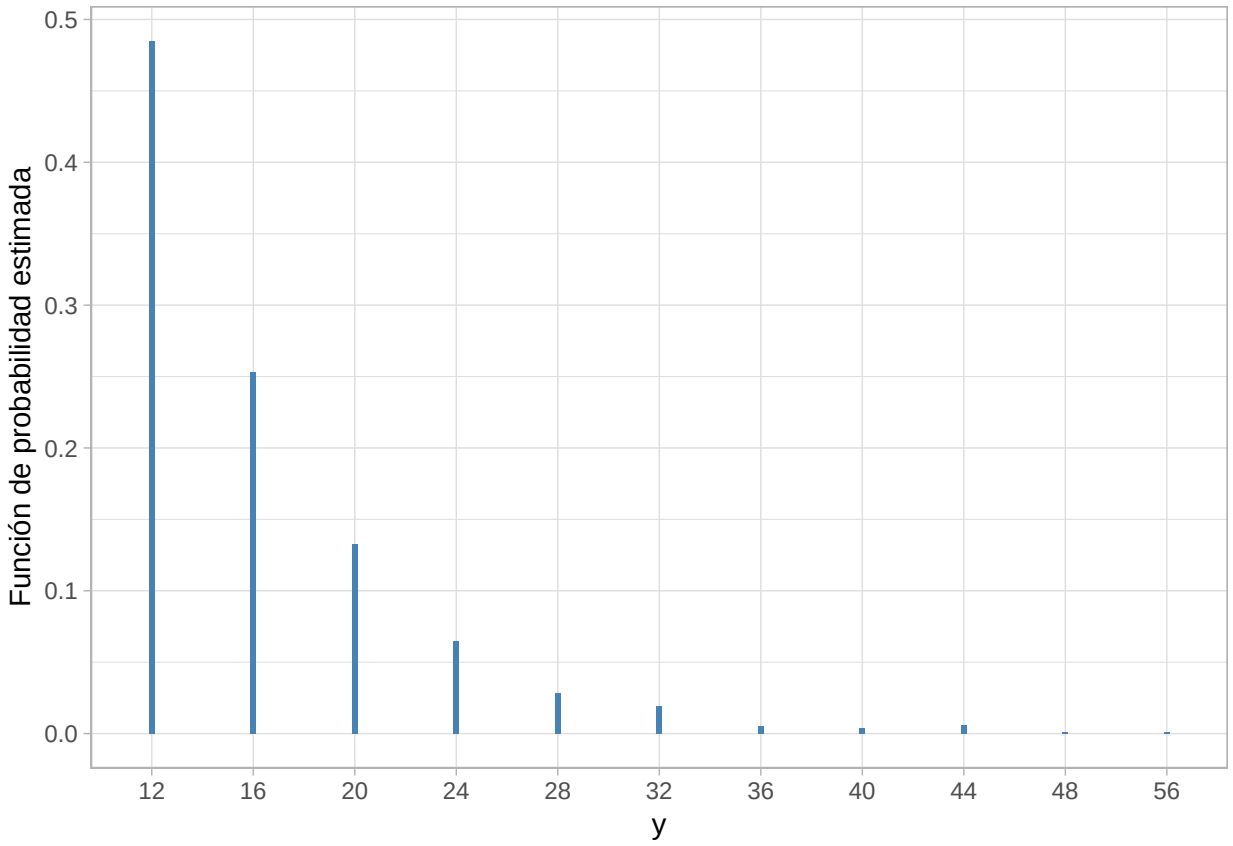
```
set.seed(27)
n<-1000 # cantidad de simulaciones
x<-rgeom(n, 0.5)+1 #simulo n observaciones de una variable con distribucion geometrica

# le sume 1 porque en R la geom cuenta los fracasos hasta el primer exito.

p_X_estimada<-table(x)/length(x) # estimo la funcion de probabilidad

# grafico la funcion de probabilidad estimada

library(ggplot2)
tabla<- as.data.frame(p_X_estimada)
ggplot(data=tabla, aes(x=x, y=Freq)) +
```

```
  geom_bar(stat="identity", width=0.05, fill = "steelblue")+
  labs(x="x", y="Función de probabilidad estimada")+
  theme_light()
```



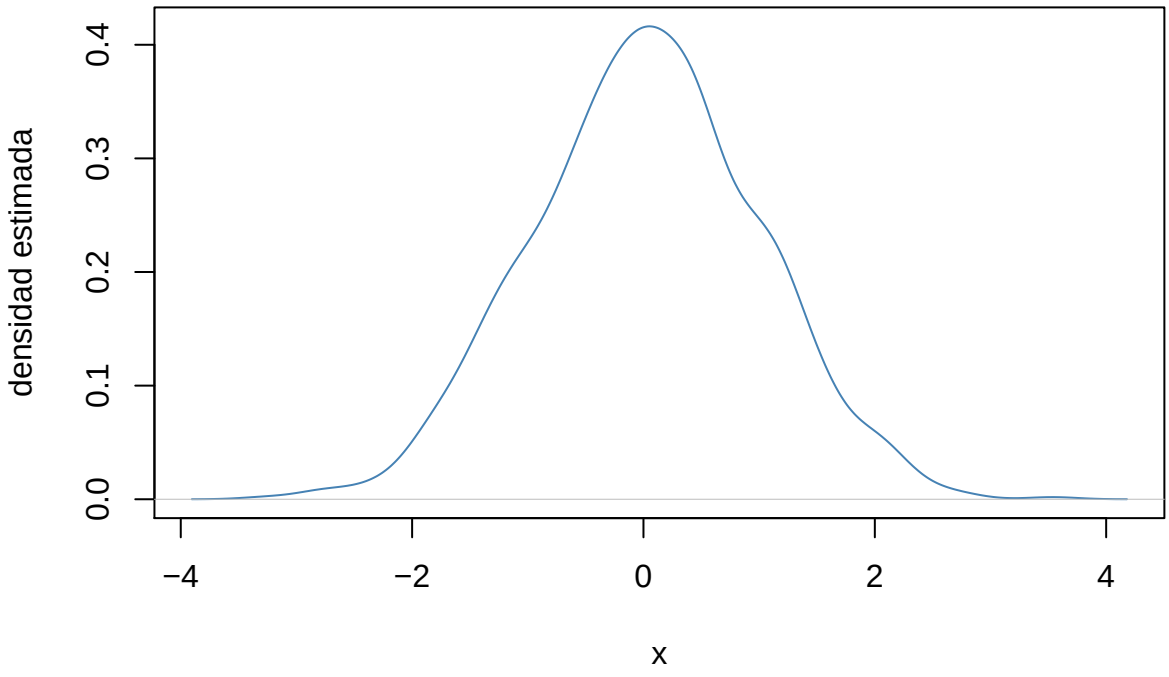
Supongamos ahora que queremos conocer la distribución de $Y = 4X + 8$. De forma teórica podríamos encontrarla realizando la transformación, pero podemos realizarlo de forma estimada usando R

```
y<-4*x+8 #listo, solo con este renglon ya hice la transformacion...

p_Y_estimada<-table(y)/length(y) # estimo la funcion de probabilidad

# grafico la funcion de probabilidad estimada

tabla<- as.data.frame(p_Y_estimada)
ggplot(data=tabla, aes(x=y, y=Freq)) +
  geom_bar(stat="identity", width=0.05, fill = "steelblue")+
  labs(x="y", y="Función de probabilidad estimada")+
  theme_light()
```

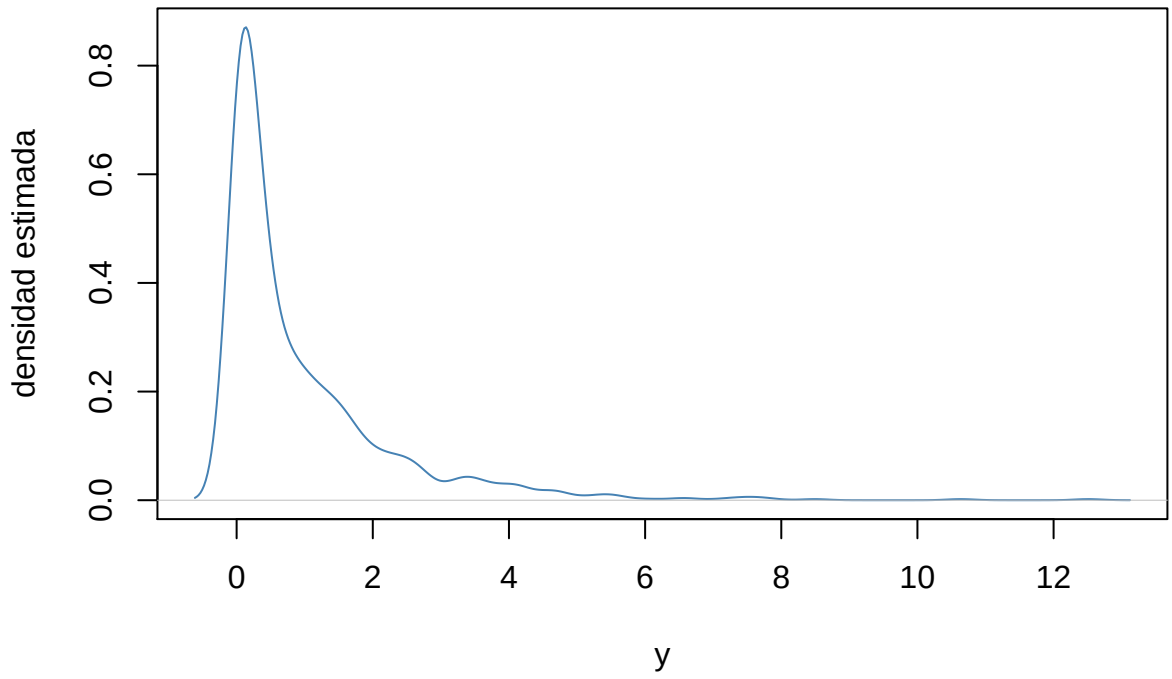


Ahora con una variable continua. Supongamos que $X \sim \mathcal{N}(0,1)$. Ahora, queremos conocer la distribución de $Y = X^2$. La forma teórica es un poco mas complicada, en general es un tema largo de aprender. Pero simulando, podemos observar muy rápidamente la forma de la densidad estimada para la nueva variable.

```
x<-rnorm(n,0,1) # rnorm simula variables normales. Siempre help() para chequear los argumentos.
plot(density(x),xlab="x", ylab="densidad estimada", col="steelblue", main="")
```

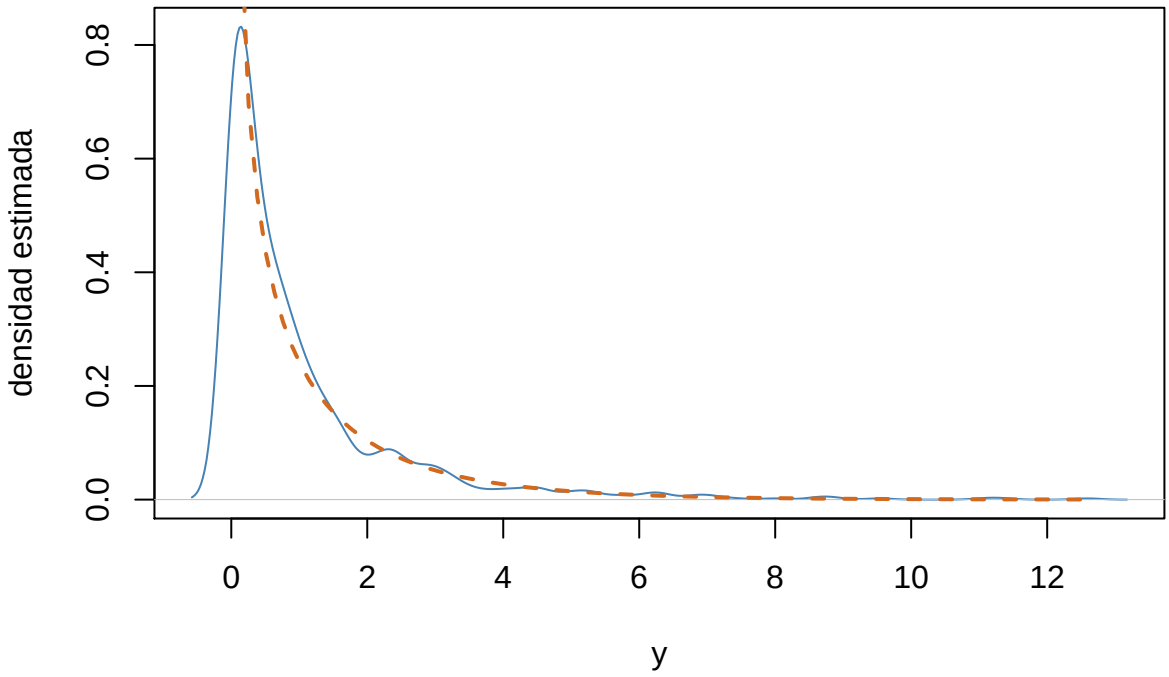


```r
y<-x^2

plot(density(y),xlab="y", ylab="densidad estimada", col="steelblue", main="")
```



Si quisieramos podemos hacer la transformación de forma teórica y superponer los gráficos, de manera que podamos verificar si nuestras estimaciones son buenas. Lo hacemos con el caso de la normal ya que sabemos que si elevamos al cuadrado una normal estándar eso nos devuelve una variable con distribución chi cuadrado.

```r
x<-rnorm(n,0,1) # rnorm simula variables normales. Siempre help() para chequear los argumentos.
y<-x^2
plot(density(y),xlab="y", ylab="densidad estimada", col="steelblue", main="")
grilla<-seq(0, max(y),length=100)
lines(grilla, dchisq(grilla,1), col="chocolate", lwd=2, lty=2)
```

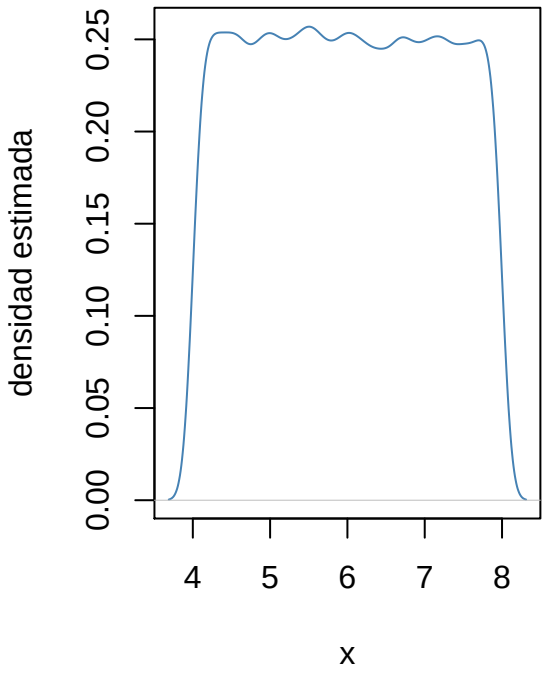


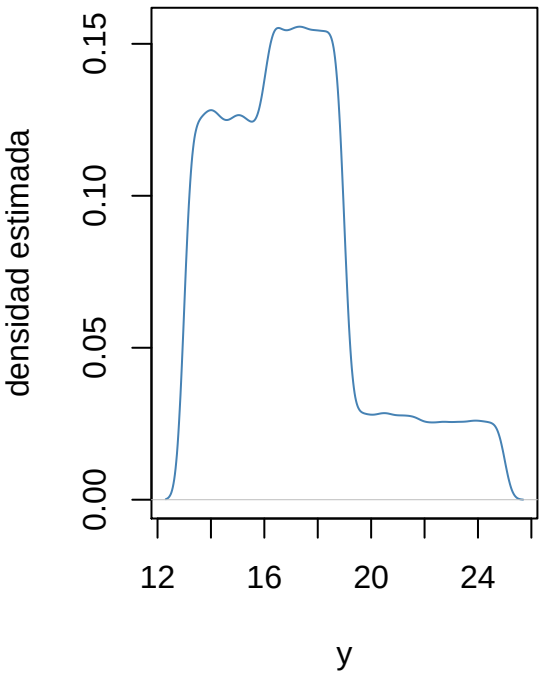
Se ve muy bien, y tan solo con $n = 1000$ simulaciones.

De esta forma es muy sencillo observar como se comportan no solo las variables que podemos medir si no también transformaciones de dichas variables, simulando y graficando tal como lo aprendimos desde el comienzo.

Si la nueva variable es una función partida, entonces necesitaremos poner en práctica lo aprendido con respecto a ciclos if, ifelse y funciones en R, pero nada muy complicado. Último ejemplo

```
n<-100000
x<-runif(n,4,8) #simulo uniformes sobre el intervalo (4,8)

y<-ifelse(x<5, x^2, 2*x+3) # y es una funcion partida de x. Ya hice la transformacion, ahora grafico

par(mfrow=c(1,2))
plot(density(x),xlab="x", ylab="densidad estimada", col="steelblue", main="")
plot(density(y),xlab="y", ylab="densidad estimada", col="steelblue", main="")
```






```
par(mfrow=c(1,1))
```

Momentos y función de regresión

Momentos

Supongamos que realizamos un experimento estadístico muchas veces, y observamos cada vez el valor de la variable aleatoria X. El promedio de dichas observaciones (bajo la mayoría de las circunstancias) va a converger a un valor fijo a medida que la cantidad de observaciones aumenta. Este valor es la Esperanza de X, $E[X]$. Esta afirmación se basa en la **Ley de los grandes números** que dice que

$$\frac{1}{n}\sum_{i=1}^{n} X_i \to E[X]$$

si las variables $X_1, \ldots, X_n$ son *i.i.d.*

Veamos un ejemplo simple. Usando simulacion, estimemos la esperanza del valor observado al tirar un dado. Debemos simular tiros de un dado, y luego estimar la esperanza de la variable promediando los resultados de las estimaciones. En R, el promedio se calcula con la función `mean()` (Que en inglés es media).

```
tiros<- sample(1:6,30, replace=TRUE)
tiros
```

```
##  [1] 2 3 6 3 3 1 3 2 6 2 1 2 3 3 3 1 1 6 1 3 2 5 4 3 4 3 1 4 5 5
```

```
mean(tiros)
```

```
## [1] 3.033333
```

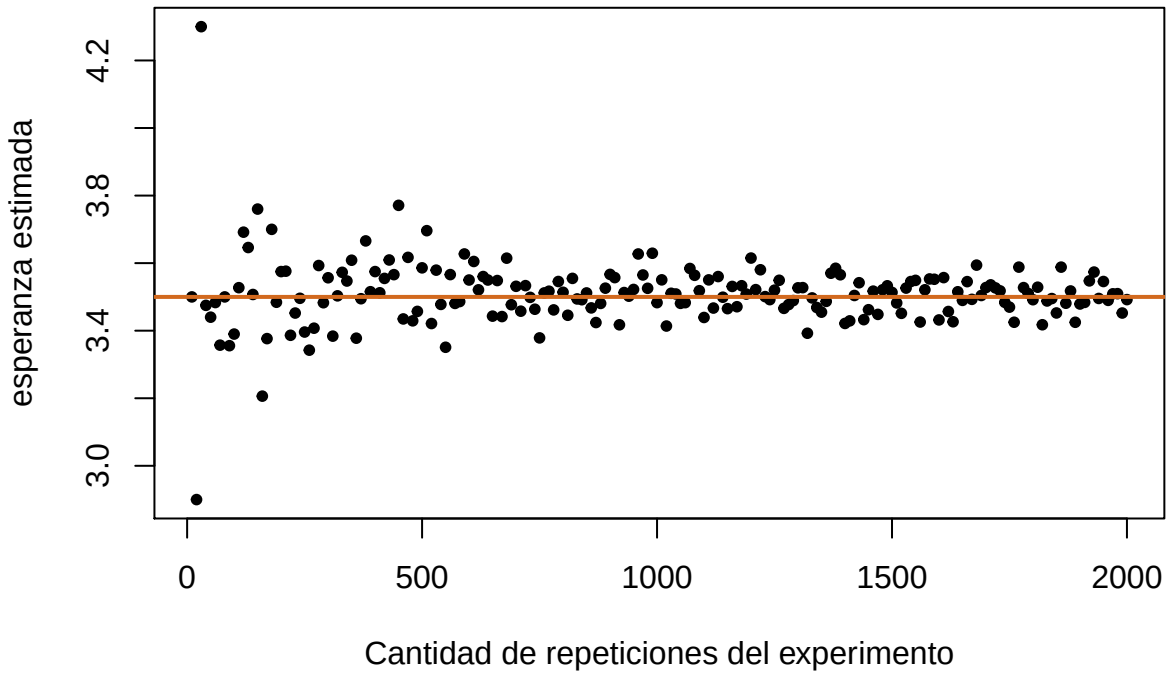
Sabemos que el valor verdadero es 3.5. Si simulamos más tiros

```
tiros<- sample(1:6,3000, replace=TRUE)

mean(tiros)
```

```
## [1] 3.466333
```

Al igual que como lo hicimos con la estimación de una probabilidad, puedo realizar la simulación para un valor de n cada vez mayor y observar como las estimaciones se acercan cada vez más al valor verdadero.

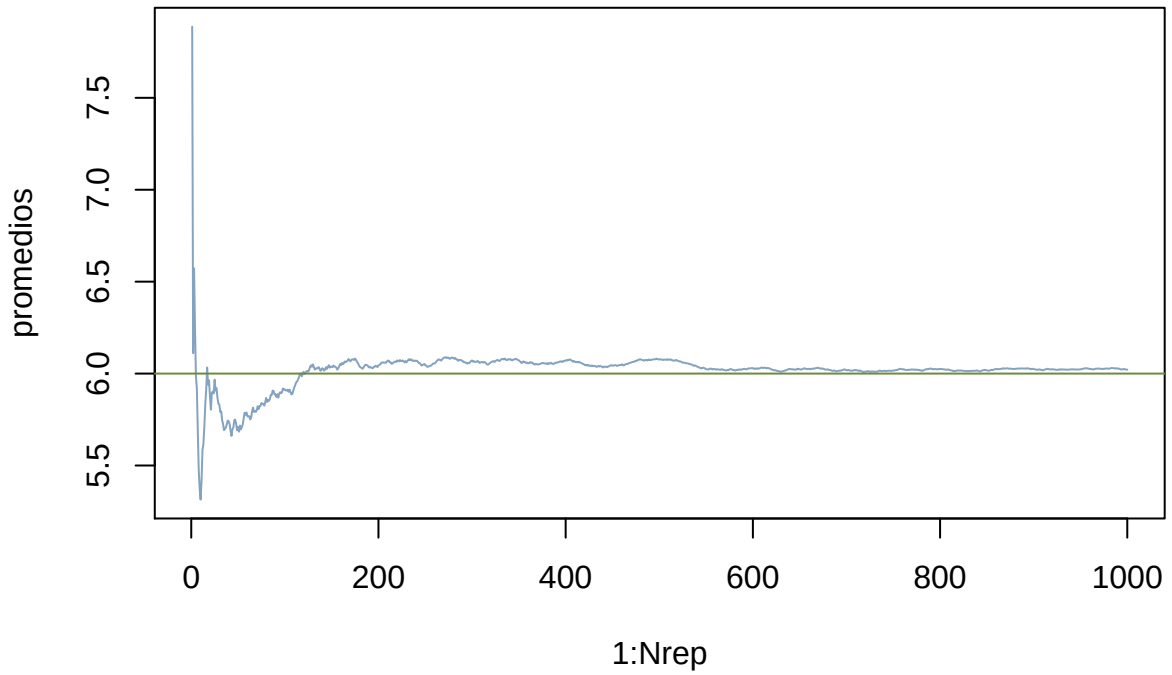
```
n<-seq(10,2000,10)
promedios<-c()
for (j in 1:length(n))
{
  tiros<- sample(1:6,n[j], replace=TRUE)
  promedios[j]<-mean(tiros)
}
plot(n,promedios,pch=20,xlab = "Cantidad de repeticiones del experimento",
     ylab= "esperanza estimada")
abline(h=3.5,col="chocolate",lwd=2)
```



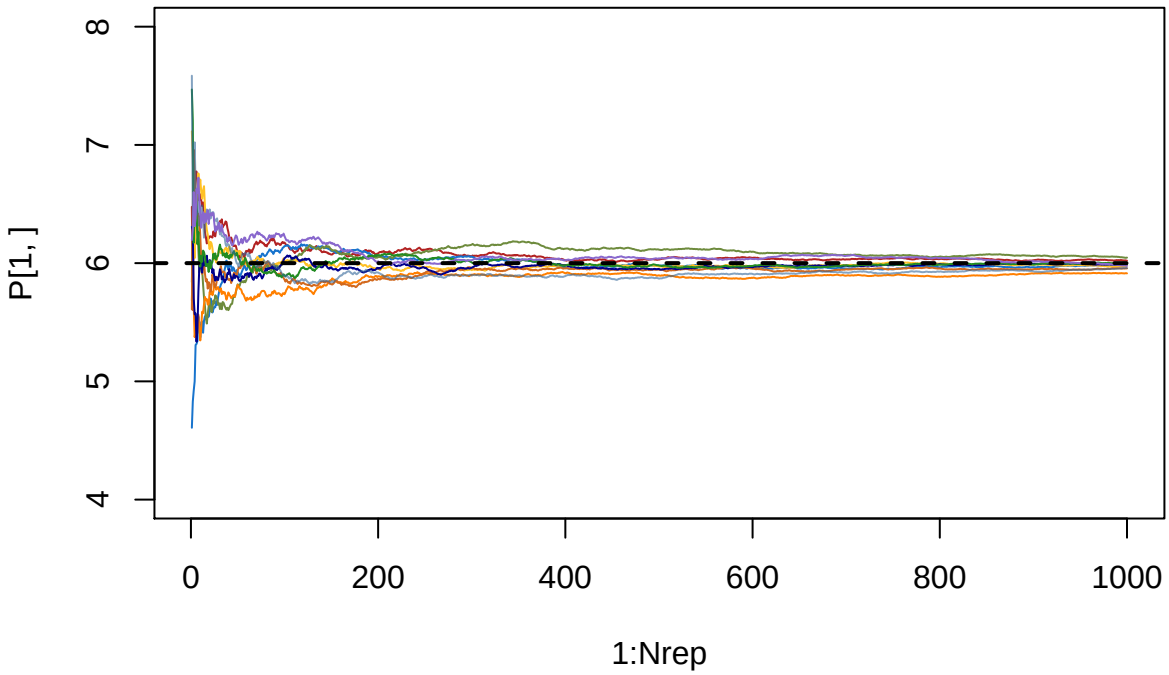
Esta herramienta nos será de mucha utilidad, ya que hay experimentos que son muy complicados, con funciones difíciles de integrar o sumar, y si logramos simular el experimento, el cálculo estimado de la esperanza es muy sencillo.

Veamos otro ejemplo. Supongamos que tenemos una variable con distribución uniforme sobre el intervalo $(4, 8)$. Vamos a simular observaciones de una variable con dicha distribución y vamos a tratar de ver como se comporta el promedio a medida que n aumenta. Pero no lo voy a hacer una vez sola, lo voy a hacer 10 veces.

```
set.seed(27)
a<-4
b<-8
Nrep<-1000 #cantidad de simulaciones
x<-runif(Nrep,a,b) #simulo las uniformes

#vamos a promediar las primeras n componentes del vector, para n desde 1 hasta Nrep

promedios<-c()
for(i in 1:Nrep)
{
  promedios[i]<-mean(x[1:i])
}

plot(1:Nrep, promedios,col=rgb(0.2,0.4,0.6,0.6), pch=20, type = "l" )
abline(h=(a+b)/2, col = "darkolivegreen4")
```




```
#repitamos esto varias veces y grafiquemos en distintos colores
n<-10
P<-matrix(0,ncol=Nrep, nrow = n)
for(j in 1:n)
{
  x<-runif(Nrep,a,b)
  for(i in 1:Nrep)
  {
    promedios[i]<-mean(x[1:i])
  }
  P[j,]<-promedios
}
plot(1:Nrep, P[1,],col=col1, pch=20, type = "l", ylim = c(4,8))
lines(1:Nrep, P[2,],col=col2, pch=20)
lines(1:Nrep, P[3,],col=col3, pch=20)
lines(1:Nrep, P[4,],col=col4, pch=20)
lines(1:Nrep, P[5,],col=col5, pch=20)
lines(1:Nrep, P[6,],col=col6, pch=20)
lines(1:Nrep, P[7,],col=col7, pch=20)
lines(1:Nrep, P[8,],col=col8, pch=20)
lines(1:Nrep, P[9,],col=col9, pch=20)
lines(1:Nrep, P[10,],col=col10, pch=20)

# ¿que dice la LGN?
abline(h=(a+b)/2, lwd=2, lty=2)
```



Observemos que para todos los experimentos, las curvas dan diferentes, pero lo importante es observar que siempre convergen al mismo valor. Eso es lo que quiere decirnos la ley de los grandes números.

La estimación de la varianza es un poco más complicada, tal como lo fue la definición de varianza. Lo mismo con la estimación de la covarianza. Serán cosas que aprenderás en detalle en las próximas materias de estadística. Te cuento la idea: recordemos que la varianza se define como

$$var(X) = E[(X - E[X])^2]$$

La varianza es una esperanza. Vimos que una forma de estimar a la esperanza, de tener una aproximación de la esperanza, es calculando el promedio, bueno, me quedo con esa idea para conseguir un valor aproximado de la varianza. Lo que voy a promediar ahora es la distancia entre cada valor observado y el promedio, todo elevado al cuadrado

$$\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Usamos el sombrero para decir que es una estimación del parámetro.

Ahora veamos como en R se puede hacer la cuenta y conseguir las estimaciones.

```
# Si seguimos con el dado:

# Tiramos 1000 veces el dado
tiros<-sample(1:6,1000, replace = TRUE)

# Estimamos la varianza

var_estimada<-sd(tiros)^2 # sd es por standard deviation
```

Si usamos los datos de alturas, tenemos la variable correspondiente a la altura del hije y la altura de la madre, podríamos querer estimar la correlación entre estas dos variables.

```r
# Defino mis variables
Xh<-alturas$altura
Xm<-alturas$altura_madre

# Estimo la covarianza, misma idea que para la varianza.
cov(Xh,Xm)
```

```
## [1] 4.909725
```

Da positiva como podíamos suponerlo.

Para poder pensar en todos los temas vistos en el capítulo 3 de la primera parte de la materia, con todos estos elementos podríamos pensar en una forma de estimar la recta de regresión, ya que para ellos necesitaríamos estimar esperanzas, varianzas y covarianzas.

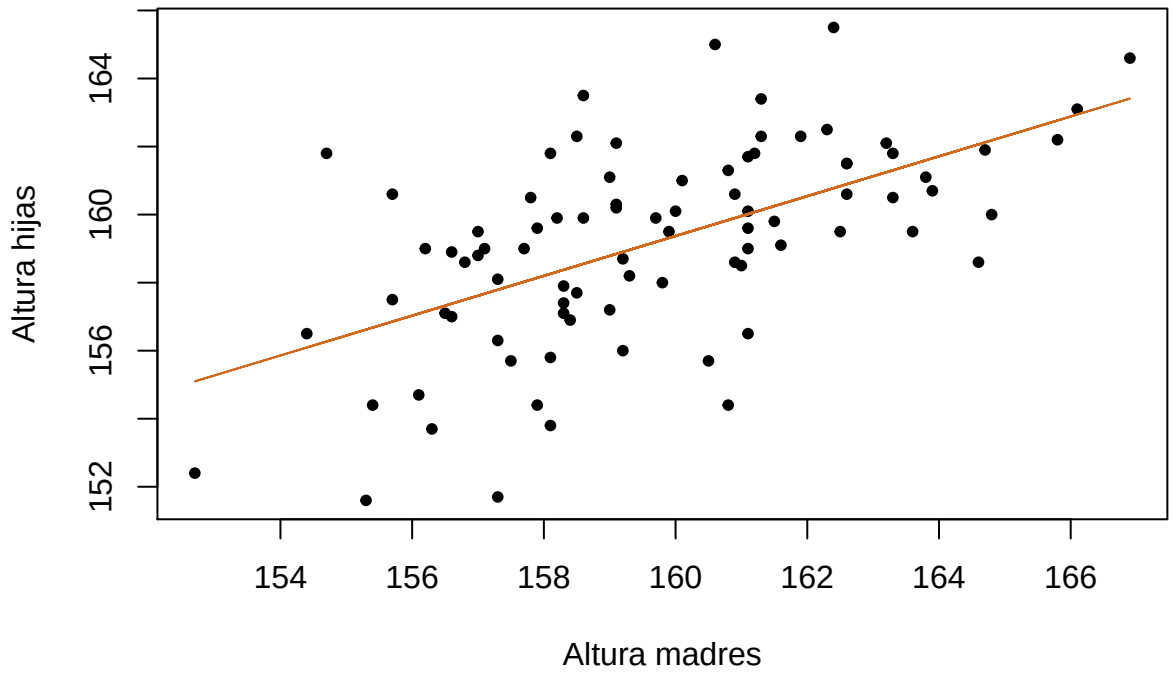
Supongamos que queremos predecir la altura del hije de género femenino, conociendo la altura de la madre. El mejor predictor lineal será la recta de regresión de X_F dado X_M.

La recta de regresión está dada por

$\hat{X_F} = \frac{cov(X_F,X_M)}{var(X_M)}(X_M - E[X_M]) + E[X_F]$

Una forma de estimar esta recta podría ser estimando cada uno de los elementos que la componen y luego evaluando los valores estimados donde irían los verdaderos. Esto se conoce como el método *plug-in*. Realicemos nuestra estimación y grafiquemos a ver que resulta.

```r
X_F<-Xh[alturas$genero=="F"]
X_M<-Xm[alturas$genero=="F"]

XF_sombrero<-cov(X_F,X_M)/var(X_M)*(X_M-mean(X_M))+mean(X_F)

plot(X_M,X_F,pch=20,xlab = "Altura madres",
     ylab= "Altura hijas")

lines(X_M,XF_sombrero, col="chocolate")
```



Función de regresión

Supongamos que $\varphi(x)$ es la verdadera función de regresión. Hay diferentes formas de estimar a $\varphi(x)$ que solo asumen que es suave. Estos métodos se llaman métodos de estimación no paramétrica. Sabemos, de la teoría de probabilidades, que la función de regresión está dada por

$$\varphi(x) = E[Y|X = x]$$

Lo que buscamos en estudiar alguna relación entre Y y X mediante la forma $Y \sim \varphi(X)$ de manera de minimizar el error cuadrático medio.

Una forma razonable para estimar a $\varphi(x)$ sería calcular el promedio de todos los valores de y para cada valor de x, siguiendo la idea de que el promedio es una muy buena estimación para la esperanza de una variable aleatoria.

Entonces, si X es discreta y tenemos un conjunto de observaciones del vector (X, Y), (x, y), podríamos estimar a $\varphi(x)$ por

$$\hat{\varphi}(x) = \frac{\sum_{i=1}^{n} y_i * 1\{x_i = x\}}{\sum_{i=1}^{n} 1\{x_i = x\}}$$

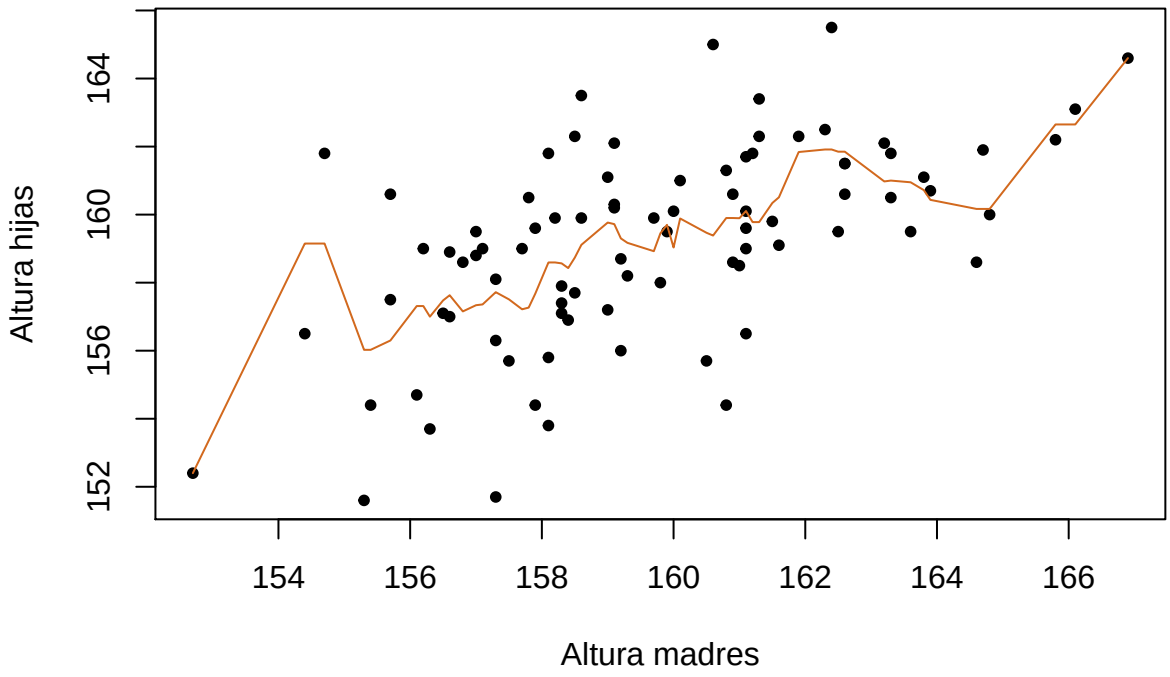
Observamos que esta fórmula calcula exactamente lo explicado, sumamos los valores de y que corresponden a la observación $X = x$ y lo dividimos por el total de esas observaciónes, entonces para cada valor de x estamos calculando el promedio de las y. Eso dará una buena estimación de la función de regresión cuando las variables son discretas.

Ahora, ¿qué pasa si las variables son continuas? Es muy probable que estos casos ocurra que para cada valor de x obtenga un solo valor de y, por lo que la curva resultante pasará por todas las observaciones y no será suave, tampoco dará una buena estimación de la función de regresión. Entonces en este caso, una forma razonable para estimar a $\varphi(x)$ puede ser calcular el promedio de las respuestas cercanas a x, por ejemplo dentro de una ventana definida en un entorno de x, $(x - h, x + h)$. El procedimiento de encontrar promedios locales es la idea central de lo que se conoce como suavizado.

Si volvemos al ejemplo de las alturas de las madres y de las hijas, podríamos buscar una estimación de la función de regresión para poder lograr conseguir una mejor predicción de la altura de la hija si conocemos la altura de la madre. Lo habíamos hecho en el capítulo 3 con una estimación para la recta de regresión. Veamos como queda con esta nueva información

```
X_F<-Xh[alturas$genero=="F"]
X_M<-Xm[alturas$genero=="F"]

funcionRegresion<-function(X,Y,xNuevo,h)
{
  M<-mean(Y[X>=(xNuevo-h) & X<=(xNuevo+h)])#promediamos los y para los x que caen dentro de la ventana
  return(M)
}

mamas<-sort(X_M)
m_sombrero<-c()
for(i in 1: length(X_M))
{
  m_sombrero[i]<-funcionRegresion(X_M,X_F,mamas[i],0.5)
}

plot(X_M,X_F,pch=20,xlab = "Altura madres",
```

```
     ylab= "Altura hijas")

lines(mamas,m_sombrero, col="chocolate")
```



Como ya aprendimos a encontrar funciones de regresion en forma teórica, en varios casos de estudio podremos superponer el gráfico de la función real y obserbar cuan buena es la aproximación.

Así como en el capítulo 9 hablamos de histograma, pero luego mejoramos esos gráficos usando núcelos, haremos lo mismo ahora. Veremos ahora uno de los mejores estimadores para la función de regresión: El estimador de NAdaraya-Watson.

Volvamos a la definición de función de regresión (para el caso continuo)

$$\varphi(x) = E[Y|X = x] = \frac{\int_{-\infty}^{\infty} y * f(x, y)dy}{f(x)}$$

Entonces podemos usar estimaciones para las densidades usando núcleos. Recordemos que

$$\hat{f}_h(x) = \frac{1}{nh_X} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h_X}\right).$$

Con la misma idea, tendremos una estimación para la densidad conjunta

$$\hat{f}_h(x, y) = \frac{1}{nh_X h_Y} \sum_{i=1}^{n} K_X\left(\frac{x - x_i}{h_X}\right) K_Y\left(\frac{y - y_i}{h_Y}\right).$$

Si definimos

$$\hat{\varphi}(x) = \frac{\int_{-\infty}^{\infty} y * \hat{f}(x, y)dy}{\hat{f}(x)}$$

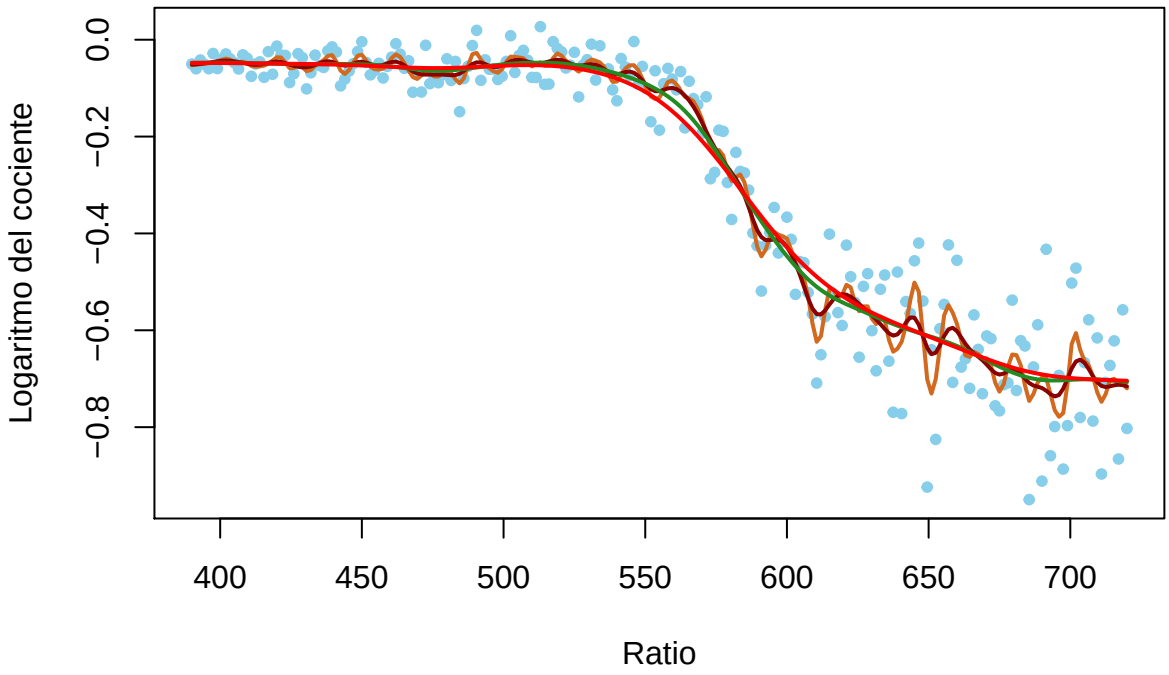
Reemplazando y resolviendo resulta el estimador de Nadaraya-Watson

$$\hat{\varphi}(x) = \frac{\frac{1}{n}\sum_{i=1}^{n} K\left(\frac{x-x_i}{h}\right) * y_i}{\frac{1}{n}\sum_{i=1}^{n} K\left(\frac{x-x_i}{h}\right)}$$

Este estimador es un promedio ponderado de las variables respuesta y. La forma de los pesos está dado por la función K y el tamaño de los pesos está parametrizado por h. El valor óptimo de h se consigue tratando de minimizar el error cuadrático medio estimado mediante una técnica que lleva el nombre de validación cruzada. Es un tema muy interesante que excede el alcance de esta materia. Es muy importante entender que un valor muy pequeño de h consigue estimaciones que reproducen los datos, y valores muy grandes dan curvas super suaves que tampoco resultan útiles.

En R, para usar el estimador de N-W se puede utilizar la función **ksmooth**, donde aclaramos cual es el núcleo que queremos usar y el valor de h (bandwith). Veamos un ejemplo con una base de datos ya armada.

```
# ejemplo con datos IDAR (light detection and ranging)
# range --> x
# logratio --> y

datos<-read.table("lidar.txt",header = TRUE)

plot(datos$range,datos$int.conc,pch=20,col="skyblue",
     xlab= "Ratio", ylab = "Logaritmo del cociente")

# Nadaraya watson usando R, si no te armas la funcion
regNP<-ksmooth(datos$range,datos$int.conc,kernel = "normal", bandwidth = 5,
               x.points = datos$range)
lines(regNP$x,regNP$y,col= "chocolate",lwd=2)

regNP2<-ksmooth(datos$range,datos$int.conc,kernel = "normal", bandwidth = 10,
                x.points = datos$range)
lines(regNP2$x,regNP2$y,col= "darkred",lwd=2)

regNP3<-ksmooth(datos$range,datos$int.conc,kernel = "normal", bandwidth = 30,
                x.points = datos$range)
lines(regNP3$x,regNP3$y,col= "forestgreen",lwd=2)

regNP4<-ksmooth(datos$range,datos$int.conc,kernel = "normal", bandwidth = 50,
                x.points = datos$range)
lines(regNP4$x,regNP4$y,col= "red",lwd=2)
```



Se observa claramente en el gráfico como a medida que aumentamos h, la curva se suaviza.