Trabajo Práctico Final

Introducción a la Estadística y la Ciencia de Datos, 1er Cuatrimeste 2024

Alan Erdei, Nicolás Ian Rozenberg

2024-07-11

1. Sección Teórica

(a)

Demostraremos que el predictor derivado del estimador de Nadaraya-Watson de la esperanza condicional Y_i = E(Y | X = X_i), 1 ≤ i ≤ n de la forma

$$\hat{\boldsymbol{Y}} = \begin{pmatrix} \hat{Y_1} & \hat{Y_2} & \cdots & \hat{Y_n} \end{pmatrix}^T$$

donde

$$\hat{Y_i} = \hat{m_h}(X_i)$$

es una transformación aplicada sobre **Y**.

$$\hat{Y_i} = \hat{m_h}(X_i) = \sum_{j=1}^{n} Y_j w_{j,h}(X_i)$$

$$= \begin{pmatrix} w_{1,h}(X_i) & w_{2,h}(X_i) & \cdots & w_{n,h}(X_i) \end{pmatrix} \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{pmatrix}$$

Por lo tanto

$$\underbrace{\begin{pmatrix} \hat{Y_1} \\ \hat{Y_2} \\ \vdots \\ \hat{Y_n} \end{pmatrix}}_{\hat{\mathbf{Y}}} = \underbrace{\begin{pmatrix} w_{1,h}(X_1) & w_{2,h}(X_1) & \cdots & w_{n,h}(X_1) \\ w_{1,h}(X_2) & w_{2,h}(X_2) & \cdots & w_{n,h}(X_2) \\ \vdots & \vdots & \ddots & \vdots \\ w_{1,h}(X_n) & w_{2,h}(X_n) & \cdots & w_{n,h}(X_n) \end{pmatrix}}_{S} \underbrace{\begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{pmatrix}}_{\mathbf{Y}}$$

Entonces obtuvimos una transformación lineal en **Y**, como se quería ver. ■

2. Sección Práctica

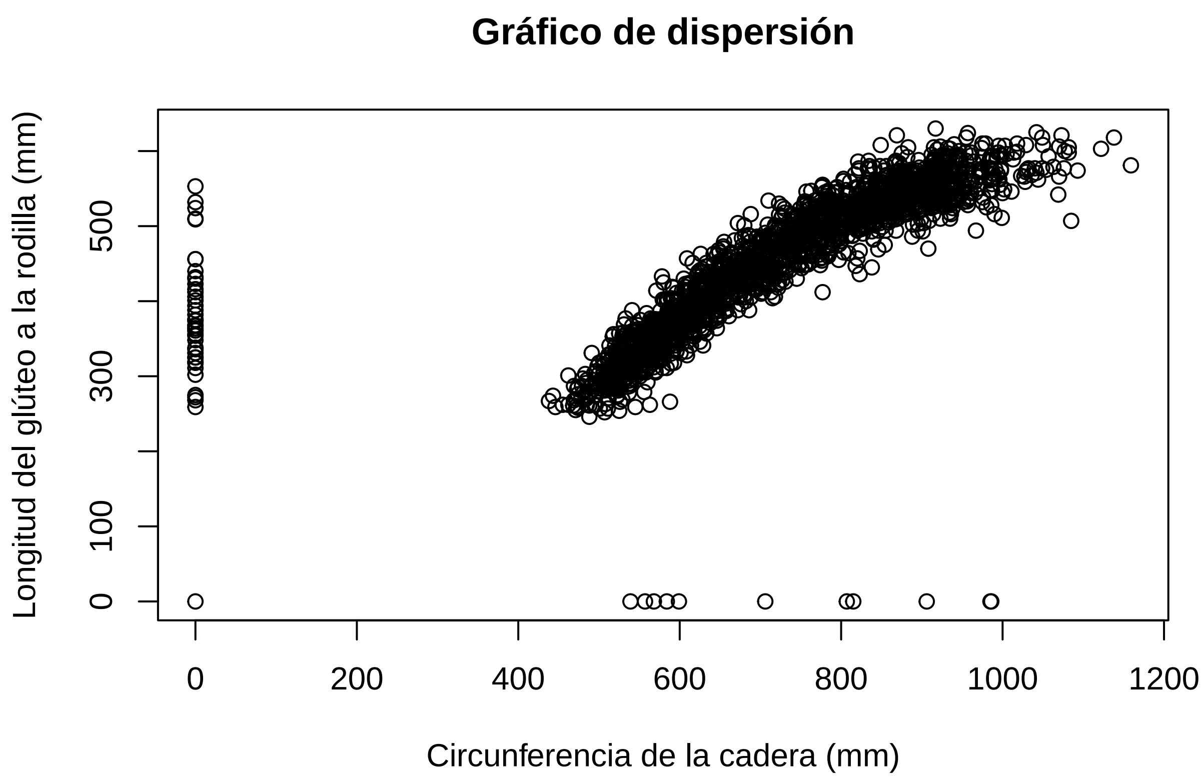
(a)

Cargamos los datos

```r
datos <- read.csv("individuals.csv", sep=";")
poblacion_femenina <- datos[datos$SEX == 2,]
attach(poblacion_femenina)
```

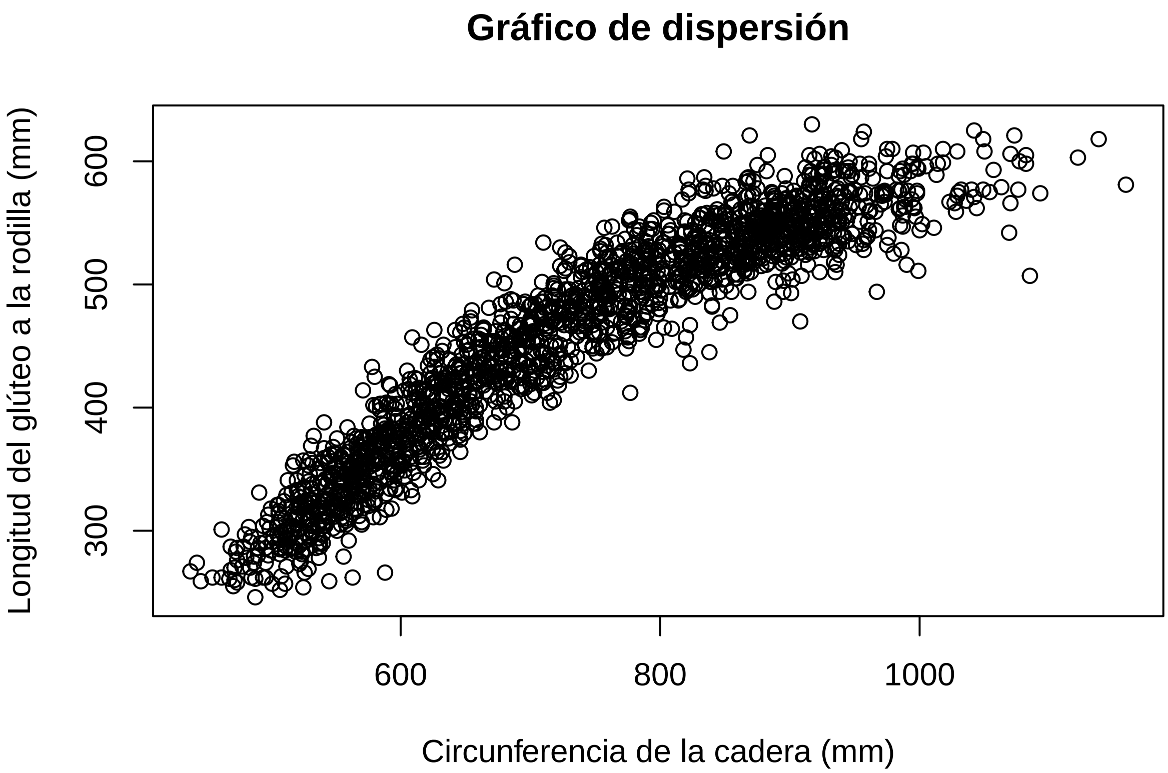
Primer gráfico de dispersión de Circunferencia de la cadera (HIP.CIRCUMFERENCE) (eje x) vs. Longitud del glúteo a la rodilla (BUTTOCK.KNEE.LENGTH) (eje y)

```r
plot(
  HIP.CIRCUMFERENCE, BUTTOCK.KNEE.LENGTH,
  main = "Gráfico de dispersión",
  xlab = "Circunferencia de la cadera (mm)",
  ylab = "Longitud del glúteo a la rodilla (mm)"
)
```



Se pueden observar puntos donde o bien la primera variable es 0, o bien la segunda lo es. Podemos interpretar que se desconoce el valor. Graficamos las mismas variables quitando dichos puntos.

```r
plot(HIP.CIRCUMFERENCE[HIP.CIRCUMFERENCE != 0 & BUTTOCK.KNEE.LENGTH != 0],
     BUTTOCK.KNEE.LENGTH[HIP.CIRCUMFERENCE != 0 & BUTTOCK.KNEE.LENGTH != 0],
     main = "Gráfico de dispersión",
     xlab = "Circunferencia de la cadera (mm)",
     ylab = "Longitud del glúteo a la rodilla (mm)"
)
```



Este gráfico sugiere una correlación positiva entre la circunferencia de la cadera y la longitud del femur.

Eliminamos los datos que posean 0 en estas columnas.

```
poblacion_femenina <- poblacion_femenina[
  HIP.CIRCUMFERENCE != 0 & BUTTOCK.KNEE.LENGTH != 0,
]
attach(poblacion_femenina)
```

(b)

Separamos a la población femenina en grupos etarios de acuerdo al cuartil, y calculamos intervalo de confianza bootstrap de nivel aproximado 0.95. Para cada grupo, generamos un número (`n_bootstrap`, 1000) de muestras bootstrap. Cada muestra bootstrap se obtiene al seleccionar observaciones aleatoriamente con reemplazo del conjunto de datos original del grupo. Esto significa que algunas observaciones pueden ser seleccionadas más de una vez mientras que otras pueden no ser seleccionadas en absoluto. Para cada una de estas muestras bootstrap, calculamos la mediana de `HIP.CIRCUMFERENCE`, y el intervalo de confianza normal. Esto se hizo ordenando las medianas bootstrap y seleccionando los percentiles correspondientes al 2.5% y al 97.5% para un intervalo de confianza del 95%. Estos percentiles representan los límites inferior y superior del intervalo de confianza.

```
quartiles <- quantile(poblacion_femenina$AGE.IN.MONTHS, probs = c(0.25, 0.5, 0.75))
poblacion_femenina$group <- cut(
  AGE.IN.MONTHS,
  breaks = c(-Inf, quartiles, Inf),
  labels = c("Q1", "Q2", "Q3", "Q4")
```

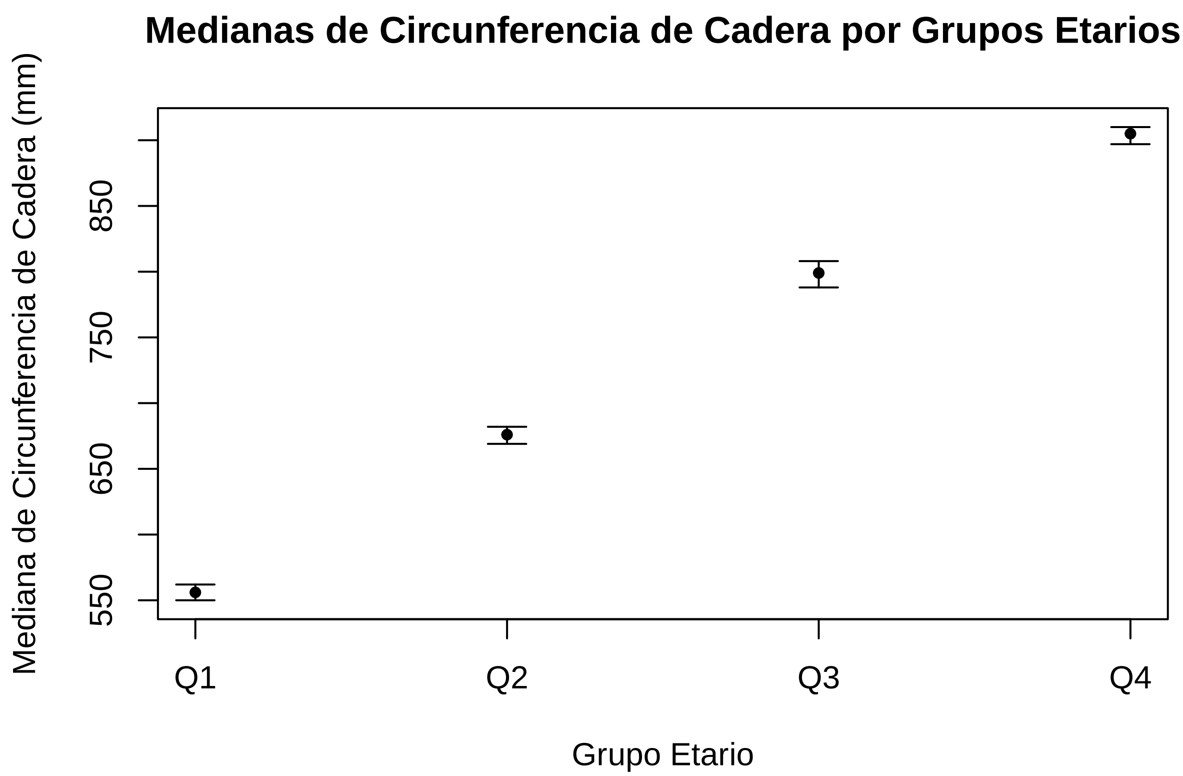
```r
)

ic_bootstrap <- function(data, n_bootstrap = 1000, conf_level = 0.95) {
  medians <- c()

  for (i in 1:n_bootstrap){
    medians <- c(medians, median(sample(data, replace = TRUE)))
  }

  lower_bound <- quantile(medians, (1 - conf_level) / 2)
  upper_bound <- quantile(medians, 1 - (1 - conf_level) / 2)
  list(median = median(data), lower = lower_bound, upper = upper_bound)
}

results <- lapply(
  split(HIP.CIRCUMFERENCE, poblacion_femenina$group), ic_bootstrap
)

for (i in 1:4) {
  cat("Grupo", names(results)[i], "\n")
  cat("Mediana:", results[[i]]$median, "\n")
  cat("Intervalo de confianza:", results[[i]]$lower, "-", results[[i]]$upper, "\n\n")
}
```

```
## Grupo Q1
## Mediana: 556
## Intervalo de confianza: 550 - 562
##
## Grupo Q2
## Mediana: 676
## Intervalo de confianza: 669 - 682
##
## Grupo Q3
## Mediana: 799
## Intervalo de confianza: 788 - 808.0375
##
## Grupo Q4
## Mediana: 905
## Intervalo de confianza: 897 - 910
```

Graficamos los resultados

```r
groups <- names(results)

medians <- sapply(results, function(x) x$median)
lower_bounds <- sapply(results, function(x) x$lower)
upper_bounds <- sapply(results, function(x) x$upper)

plot(1:4, results$medians, ylim = range(c(lower_bounds, upper_bounds)), xaxt = "n",
     xlab = "Grupo Etario", ylab = "Mediana de Circunferencia de Cadera (mm)",
     main = "Medianas de Circunferencia de Cadera por Grupos Etarios")
axis(1, at = 1:4, labels = groups)
```

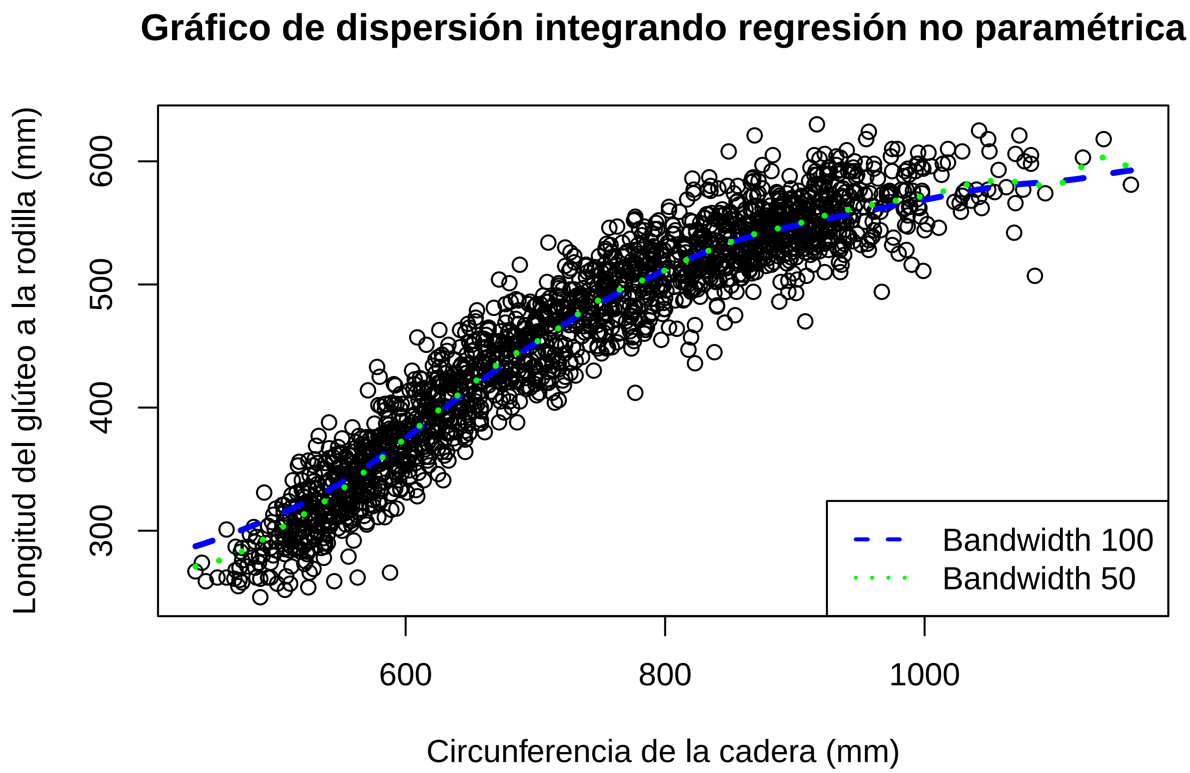
```r
arrows(1:4, lower_bounds, 1:4, upper_bounds, angle = 90, code = 3, length = 0.1)
points(1:4, medians, pch=20)
```



(c)

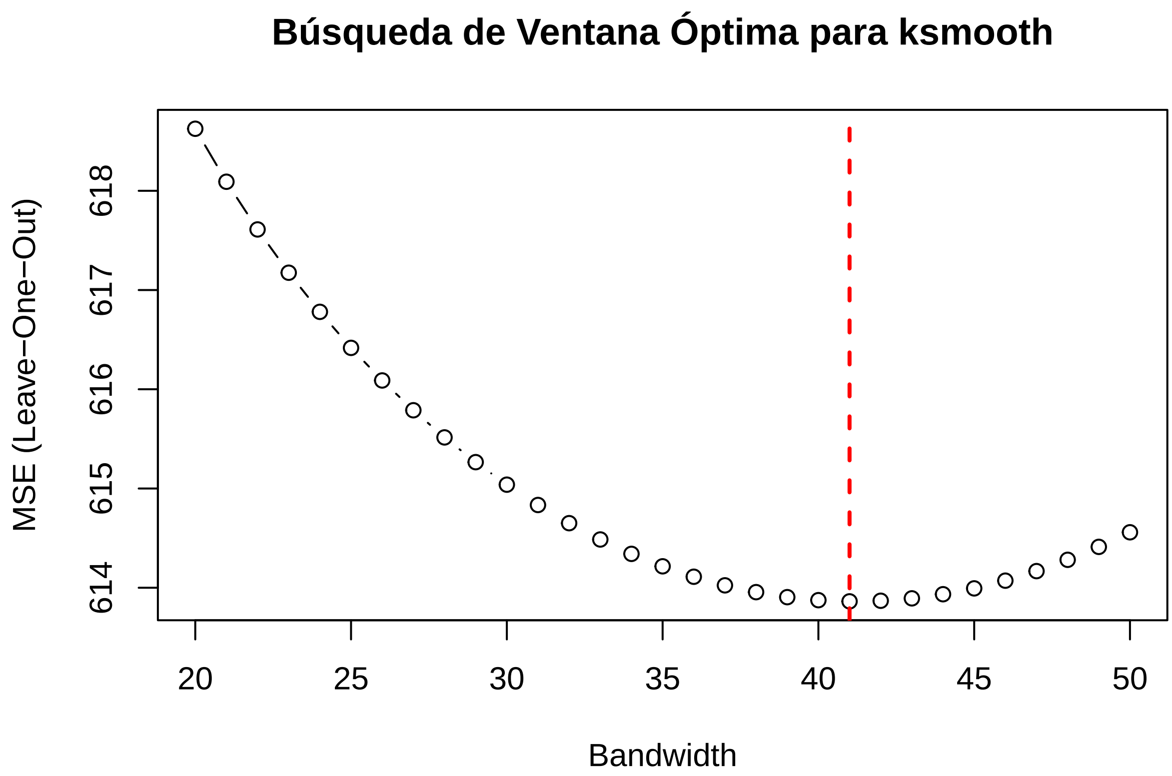
Agregamos las líneas de regresión al diagrama de dispersión original.

```r
X <- HIP.CIRCUMFERENCE
Y <- BUTTOCK.KNEE.LENGTH

plot(X, Y,
     main = "Gráfico de dispersión integrando regresión no paramétrica",
     xlab = "Circunferencia de la cadera (mm)",
     ylab = "Longitud del glúteo a la rodilla (mm)"
)
smooth_100 <- ksmooth(X, Y, kernel = "normal", bandwidth = 100)
smooth_50 <- ksmooth(X, Y, kernel = "normal", bandwidth = 50)

lines(smooth_100, col = "blue", lwd = 3, lty = 2)
lines(smooth_50, col = "green", lwd = 3, lty = 3)
legend("bottomright", legend = c("Bandwidth 100", "Bandwidth 50"),
       col = c("blue", "green"), lty = c(2, 3), lwd = 2)
```



Ahora aplicamos validación cruzada Leave One Out para ksmooth. Por ahora no aplicamos el criterio demostrado en el apartado teórico.

```r
loo_mse <- function(bandwidth) {
  n <- length(X)
  mse <- 0

  for (i in 1:n) {
    X_train <- X[-i]
    Y_train <- Y[-i]
    smooth <- ksmooth(X_train, Y_train, kernel = "normal", bandwidth = bandwidth, x.points = X[i])
    mse <- mse + (Y[i] - smooth$y)^2
  }

  return(mse / n)
}

# Búsqueda de la ventana óptima en la grilla de 20 a 50 con paso 1
bandwidths <- 20:50
mse_values <- sapply(bandwidths, loo_mse)

optimal_bandwidth <- bandwidths[which.min(mse_values)]
plot(bandwidths, mse_values, type = "b", main = "Búsqueda de Ventana Óptima para ksmooth",
     xlab = "Bandwidth", ylab = "MSE (Leave-One-Out)")
abline(v = optimal_bandwidth, col = "red", lwd = 2, lty = 2)
```




```r
cat("La ventana óptima obtenida es", optimal_bandwidth)
```

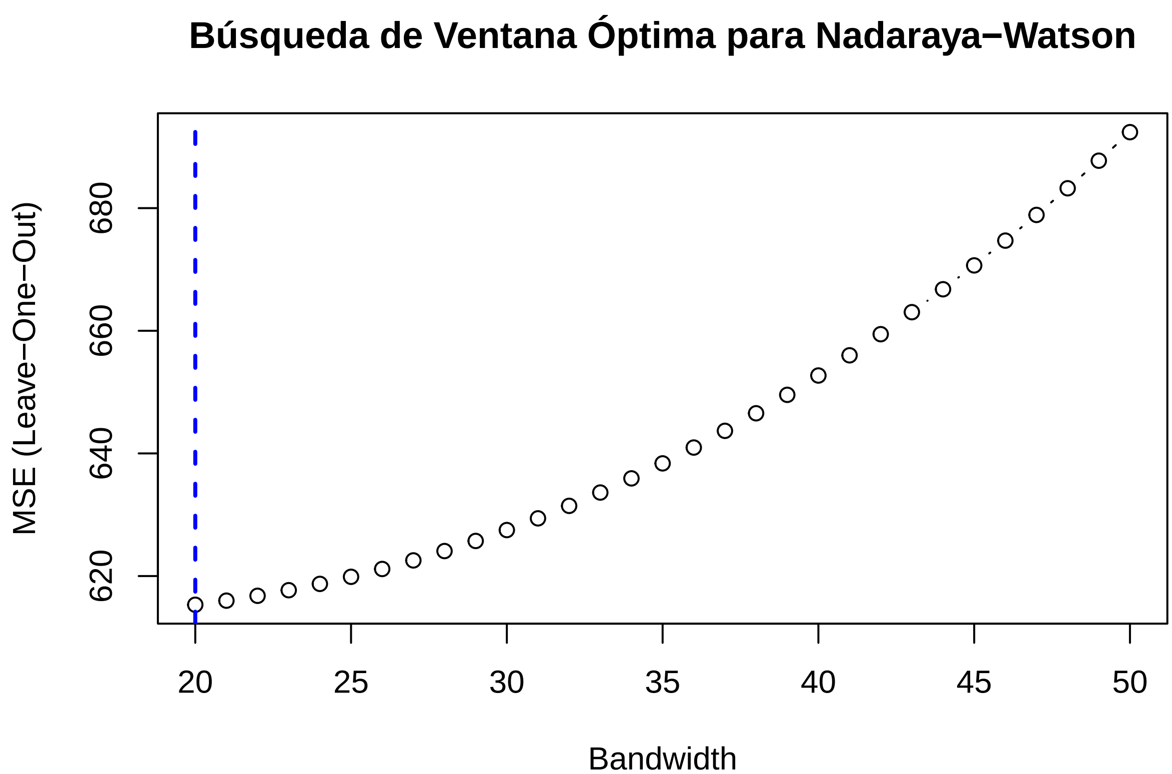
```
## La ventana óptima obtenida es 41
```

Ahora, implementamos manualmente el estimador de Nadaraya-Watson con kernel gaussiano

```r
# Función para el estimador de Nadaraya-Watson
nwsmooth <- function(x, X, Y, h) {
  K <- function(u) dnorm(u)
  n <- length(X)
  m <- numeric(length(x))

  # Si x == X, weights_matrix corresponde a la matriz asociada con
  # la transformación lineal de Nadaraya-Watson
  weights_matrix <- matrix(nrow = length(x), ncol = n)

  for (i in 1:length(x)) {
    weights_matrix[i, ] <- K((X - x[i]) / h)
    weights_matrix[i, ] <- weights_matrix[i, ] / sum(weights_matrix[i, ])
    m[i] <- sum(weights_matrix[i, ] * Y)
  }

  return(list(m = m, weights = weights_matrix))
}
```

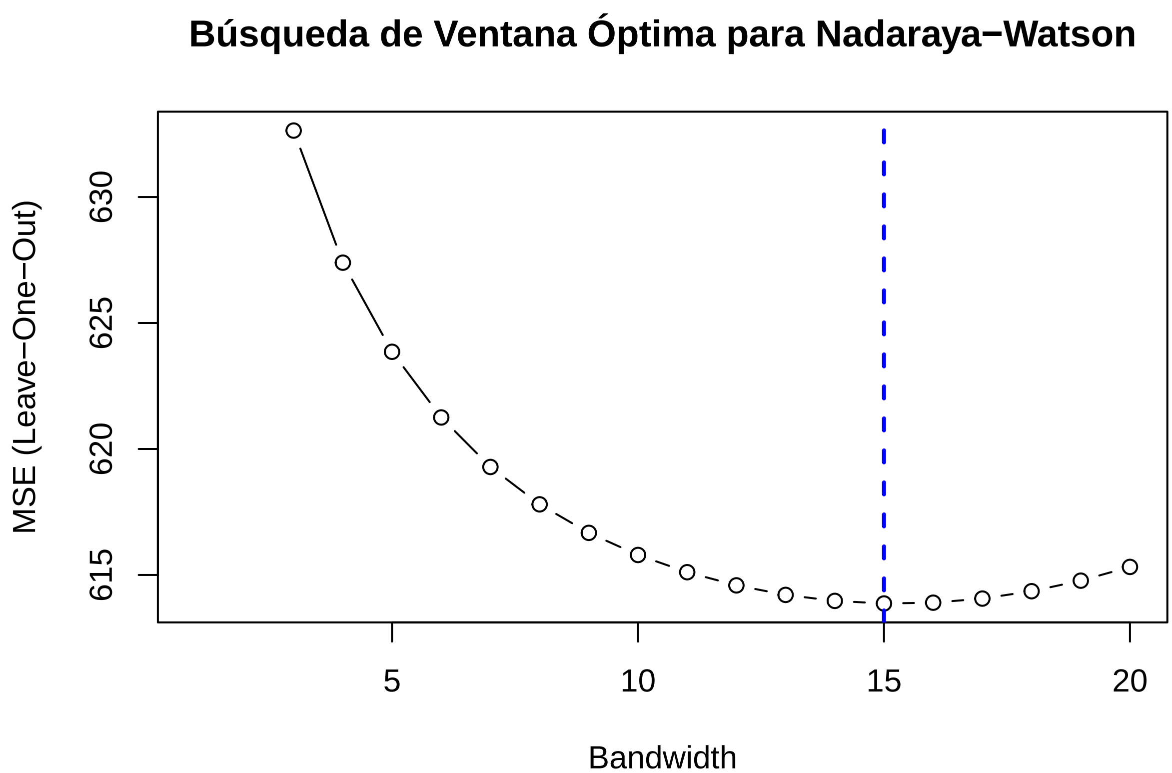
Y ahora realizamos validación cruzada. Ahora sí, utilizando el criterio del apartado teórico.

```
loo_mse_nw <- function(bandwidth) {
  n <- length(X)
  nw_response <- nwsmooth(X, X, Y, bandwidth)
  Y_hat <- nw_response$m
  weights <- nw_response$weights
  mse <- 0

  for (i in 1:n){
    mse <- mse + (Y[i]-Y_hat[i])^2 / (1 - weights[i, i])^2
  }

  mse <- mse / n

  return(mse)
}

mse_values_nw <- sapply(bandwidths, loo_mse_nw)

optimal_bandwidth_nw <- bandwidths[which.min(mse_values_nw)]
plot(bandwidths, mse_values_nw, type = "b", main = "Búsqueda de Ventana Óptima para Nadaraya-Watson",
     xlab = "Bandwidth", ylab = "MSE (Leave-One-Out)")
abline(v = optimal_bandwidth_nw, col = "blue", lwd = 2, lty = 2)
```



La función objetivo se ve distinta a cuando aplicamos ksmooth. Ahora el bandwidth óptimo es un extremo de la grilla. Crearemos una nueva grilla corrida hacia la izquierda, con ventanas que poseen valores que van de 1 a 20

```
bandwidths_nw <- 1:20
mse_values_nw <- sapply(bandwidths_nw, loo_mse_nw)

optimal_bandwidth_nw <- bandwidths_nw[which.min(mse_values_nw)]
plot(bandwidths_nw, mse_values_nw, type = "b", main = "Búsqueda de Ventana Óptima para Nadaraya-Watson"
     xlab = "Bandwidth", ylab = "MSE (Leave-One-Out)")
abline(v = optimal_bandwidth_nw, col = "blue", lwd = 2, lty = 2)
```



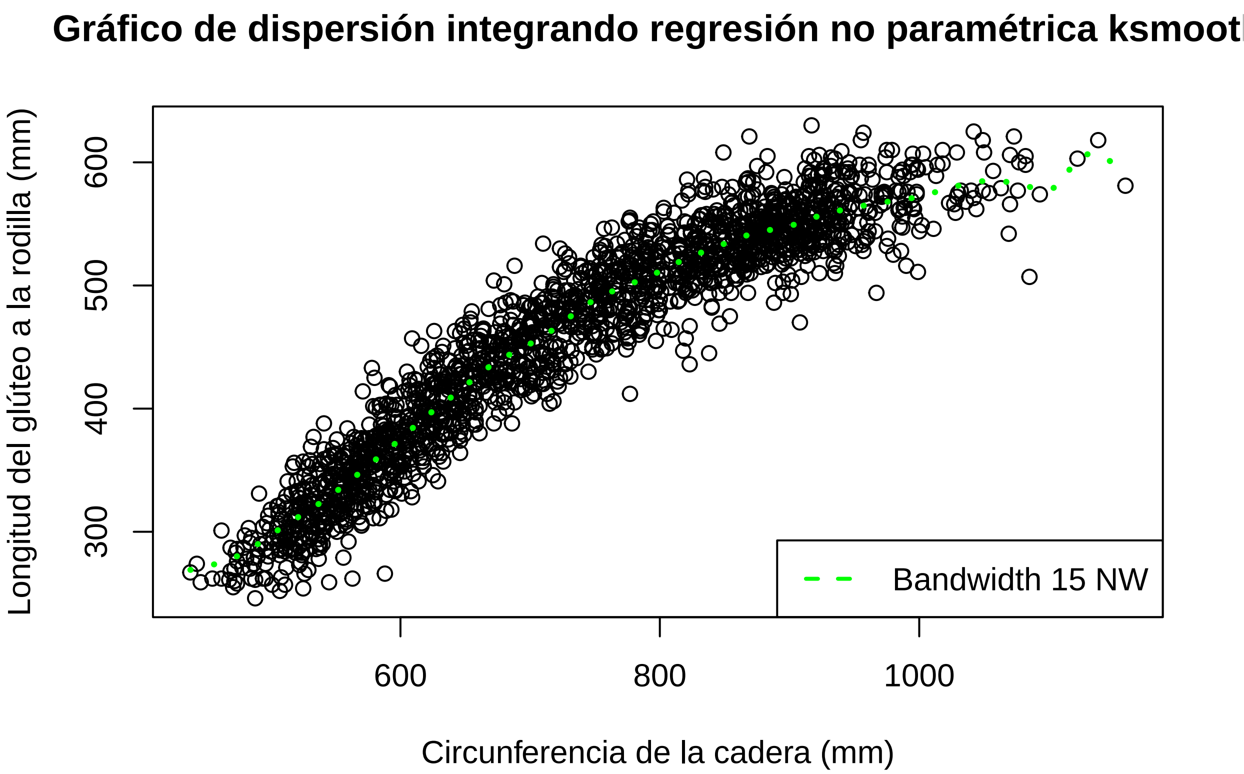
Ahora el óptimo en nuestra grilla no se encuentra en un extremo, por lo que podemos quedarnos con dicho valor.

```
cat("La ventana óptima obtenida es", optimal_bandwidth_nw)
```

```
## La ventana óptima obtenida es 15
```

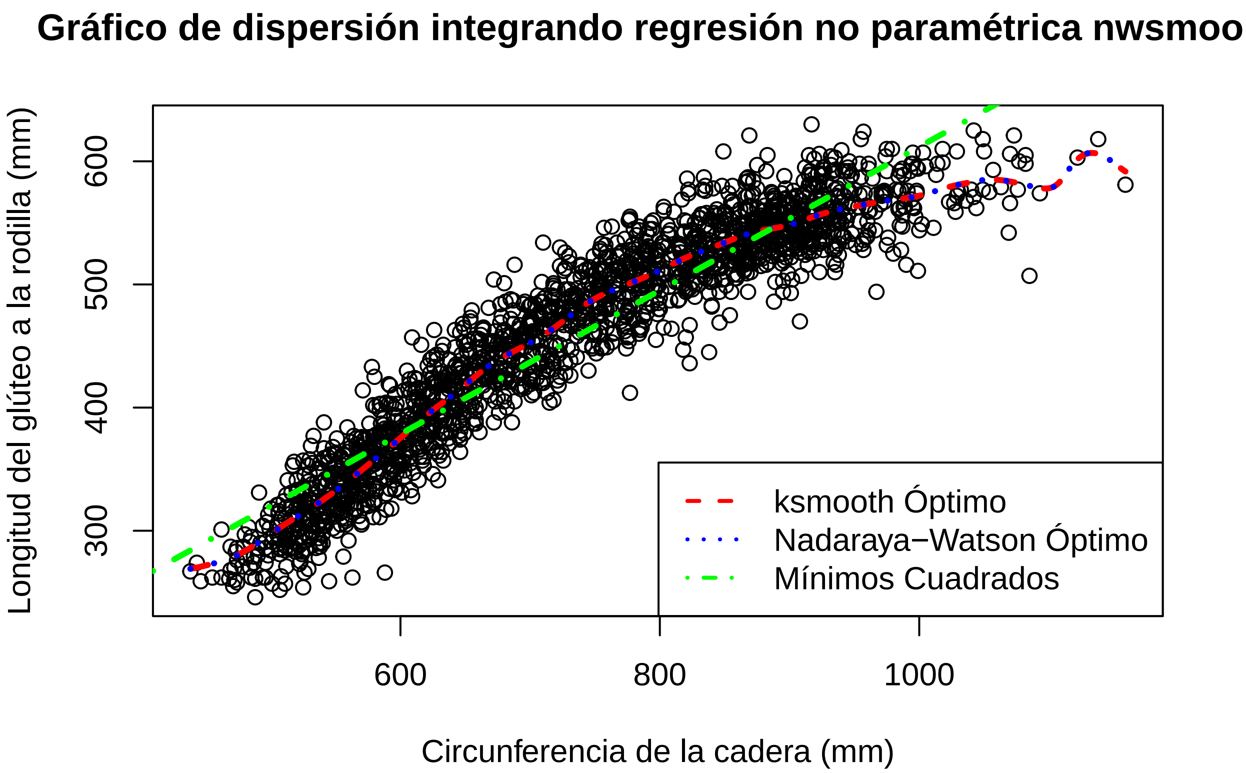
```
plot(X, Y,
     main = "Gráfico de dispersión integrando regresión no paramétrica ksmooth",
     xlab = "Circunferencia de la cadera (mm)",
     ylab = "Longitud del glúteo a la rodilla (mm)"
)

x <- seq(min(X), max(X), length.out = 2*length(X))
smooth_nw_opt <- nwsmooth(x, X, Y, optimal_bandwidth_nw)
```

```r
lines(x, smooth_nw_opt$m, col = "green", lwd = 3, lty = 3)
legend("bottomright", legend = sprintf("Bandwidth %s NW", optimal_bandwidth_nw),
       col = "green", lty = 2, lwd = 2)
```



Ahora nos falta realizar un gráfico viendo cómo queda la regresión obtenida por cuadrados mínimos, y por las mejores ventanas tanto de `ksmooth` como de `nwsmooth` (iv)

```r
smooth_opt <- ksmooth(X, Y, kernel = "normal", bandwidth = optimal_bandwidth)

fit_lm <- lm(Y ~ X)

plot(X, Y,
     main = "Gráfico de dispersión integrando regresión no paramétrica nwsmooth",
     xlab = "Circunferencia de la cadera (mm)",
     ylab = "Longitud del glúteo a la rodilla (mm)"
)

lines(smooth_opt, col = "red", lwd = 3, lty = 2)
lines(x, smooth_nw_opt$m, col = "blue", lwd = 3, lty = 3)
abline(fit_lm, col = "green", lwd = 3, lty = 4)
legend("bottomright", legend = c("ksmooth Óptimo", "Nadaraya-Watson Óptimo", "Mínimos Cuadrados"),
       col = c("red", "blue", "green"), lty = c(2, 3, 4), lwd = 2)
```



Observamos que `ksmooth` y `nwsmooth` se comportan de forma prácticamente idéntica en sus ventanas óptimas. Sólo observando el gráfico, consideramos que no se puede determinar cuál de los dos enfoques preferiríamos. Considerando que existe cierto cambio en la correlación entre las dos variables en valores altos de circunferencia de la cadera, el modelo no paramétrico podría servir para predecir mejor dichas situaciones. Sin embargo, a medida que crece el valor de las variables, los datos se vuelven más escasos. Entonces la desventaja del enfoque no paramétrico es que necesita de una cantidad mayor de datos para ser estable que en el enfoque paramétrico dado por el modelo lineal. Calculemos los R^2 (coeficiente de determinación)

```
cat("R^2 modelo lineal", summary(fit_lm)$r.squared)
```

```
## R^2 modelo lineal 0.887662
```

```
calcular_r2 <- function(Y_real, Y_hat_ksmooth){
  ss_total <- sum((Y_real - mean(Y))^2)
  ss_res_ksmooth <- sum((Y_real - Y_hat_ksmooth)^2)
  rsq <- 1 - (ss_res_ksmooth / ss_total)

  return(rsq)
}
```

```
smooth_nw_opt <- nwsmooth(X, X, Y, optimal_bandwidth_nw)
cat("R^2 NW", calcular_r2(Y, smooth_nw_opt$m))
```

```
## R^2 NW 0.9272791
```

```r
cat("R^2 ksmooth", calcular_r2(sort(Y), smooth_opt$y))
```

```
## R^2 ksmooth 0.914528
```

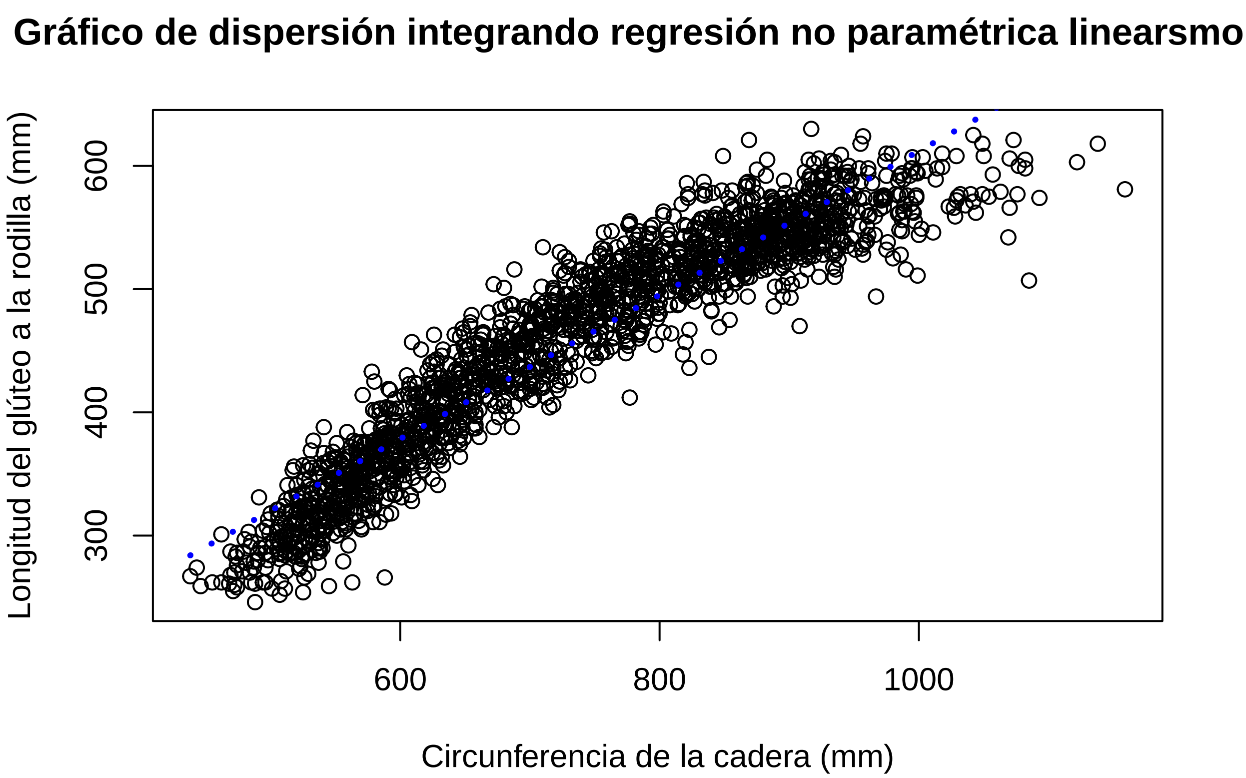
El método que mejor score R^2 tuvo es el enfoque no paramétrico con el estimador de Nadaraya Watson implementado manualmente. Por lo tanto, es el que mejor explica la variación en los datos. Sin embargo, hay que tener en consideración las observaciones previas.

(d)

Implementaremos dicho predictor basándonos en las cuentas que se encuentran en Loader (2004, p. 4-5).

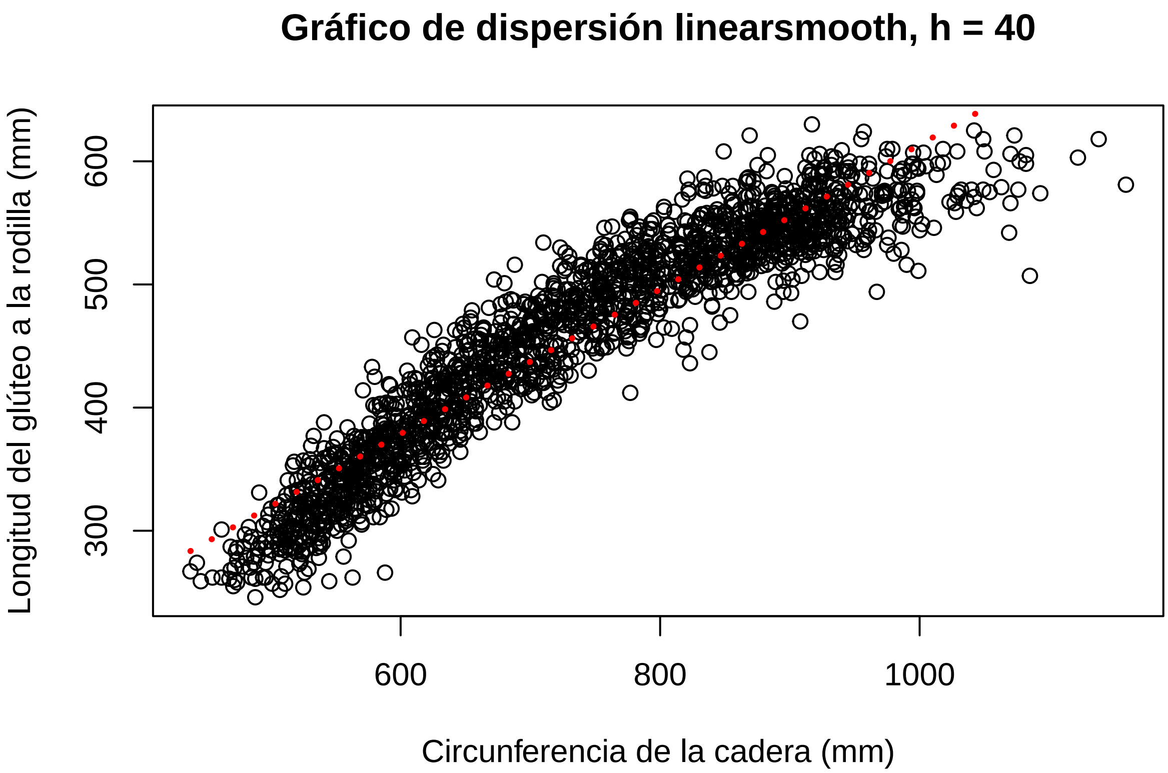
```r
linearsmooth <- function(x, X, Y, h){
  #' x: vector de valores en los que se evaluará el predictor
  #' X: vector de valores de la variable independiente
  #' Y: vector de valores de la variable dependiente
  #' h: ancho de la ventana

  K <- function(u) dnorm(u)
  n <- length(X)
  m <- numeric(length(x))

  weights_matrix <- matrix(nrow = n, ncol = n)

  for (i in 1:n) {
    weights_matrix[i, ] <- K((X - X[i]) / h)
    weights_matrix[i, ] <- weights_matrix[i, ] / sum(weights_matrix[i, ])
  }

  for (i in 1:length(x)) {
    X_design <- cbind(rep(1, n), X - x[i])
    M1 <- solve(t(X_design) %*% weights_matrix %*% X_design)
    M2 <- t(X_design) %*% weights_matrix
    coefs <- M1 %*% M2 %*% Y

    m[i] <- coefs[1]
  }

  return(m)
}

Y_pred <- linearsmooth(seq(min(X), max(X), length.out=500), X, Y, optimal_bandwidth_nw)
```

```r
plot(X, Y,
     main = "Gráfico de dispersión integrando regresión no paramétrica linearsmooth",
     xlab = "Circunferencia de la cadera (mm)",
     ylab = "Longitud del glúteo a la rodilla (mm)"
)

lines(seq(min(X), max(X), length.out=500), Y_pred, col = "blue", lwd = 3, lty = 3)
```



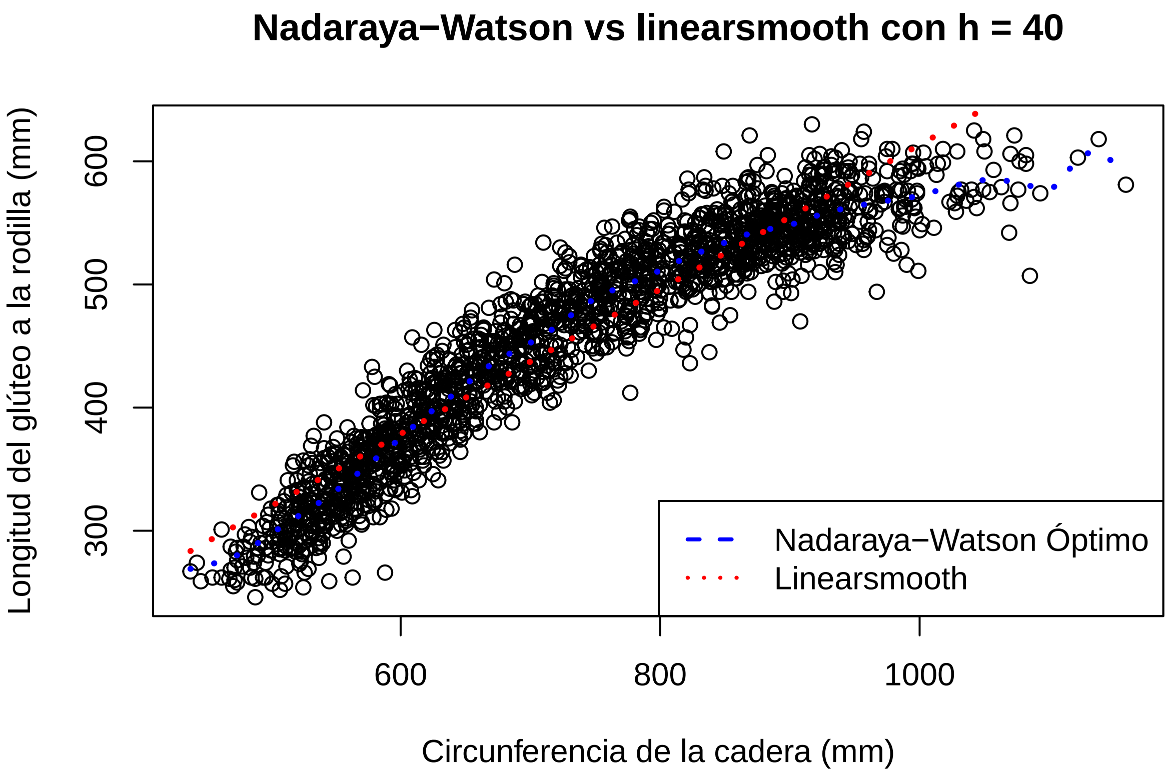
Ahora aplicamos el estimador implementado recien, pero usando una ventana de h = 40

```
h <- 40
Y_pred40 <- linearsmooth(seq(min(X), max(X), length.out=500), X, Y, h)
plot(X, Y,
     main = "Gráfico de dispersión linearsmooth, h = 40",
     xlab = "Circunferencia de la cadera (mm)",
     ylab = "Longitud del glúteo a la rodilla (mm)"
)

lines(seq(min(X), max(X), length.out=500), Y_pred40, col = "red", lwd = 3, lty = 3)
```



Graficamos ahora el estimador de Nadaraya-Watson obtenida con la ventana optima del item (c)iii, en conjunto con el estimador del grafico de arriba:

```r
smooth_nw_opt <- nwsmooth(x, X, Y, optimal_bandwidth_nw)
plot(X, Y,
     main = "Nadaraya-Watson vs linearsmooth con h = 40",
     xlab = "Circunferencia de la cadera (mm)",
     ylab = "Longitud del glúteo a la rodilla (mm)"
)

lines(x, smooth_nw_opt$m, col = "blue", lwd = 3, lty = 3)
lines(seq(min(X), max(X), length.out=500), Y_pred40, col = "red", lwd = 3, lty = 3)
legend("bottomright", legend = c("Nadaraya-Watson Óptimo", "Linearsmooth"),
       col = c("blue", "red"), lty = c(2, 3), lwd = 2)
```



Por lo que puede verse, el estimador de Nadaraya-Watson parece ajustar mejor a los datos, salvo para circunferencias de cadera grandes (>1000) y fémures largos(>520). En estos rangos, en donde esperaríamos ver una correlacion positiva entre ambas estas variables, vemos sin embargo un comportamiento extraño de Nadaraya_Watson, con subidas y bajadas, que no es lo que uno esperaria ver dados todos los demas datos. Creemos que este comportamiento es simplemente un sobreajuste de N-W debido a la falta de datos en esos rangos. Por el contrario, el ajuste de regresión local lineal pierde un poco la flexibilidad de N-W. Sin embaego esta rigidez sólo provoca que sobreestime la longitud de los femures en el rango de cirumferencias de cadera muy pequeñas, pero aun asi conserva un ajuste, al menos a la vista, bastante bueno. Ademas, al ser lineal, no muestra el comportamiento errático de N-W en valores grandes de las variables (partr de arriba a la derecha del grafico), y su rigidez permite no sobreajustar, y mostrar una tendencia razonable aun en esta region donde los datos son escasos.