Construcción de modelos de regresión y reglas de predicción paramétricos y no-paramétricos.

Aplicación:Datos de inmisión SO_2 y NO_x

Técnicas de remuestreo. Curso 2010-2011

Leyenda Rodríguez, María

Índice

1. Introducción

En este documento, se analizará un episodio de calidad de aire que tiene como origen la central térmica de As Pontes, el cual se produjo entre las 05:00 y las 09:00 del día 12 de Marzo de 2007 en la estación de inmisión G2, la cual es es propiedad de U.P.T. de As Pontes(Endesa s.a.). Para ello trabajaremos con datos de inmisión minutales tomados en tiempo real de SO_2 y NO_x.

Cabe notar que hemos escogido este episodio porque al tener como origen del episodio la central térmica cabe esperar que el NO_x siga al SO_2, de modo que si los valores de SO_2 aumentan los de NO_x también. Esto tiene sentido debido a que el SO_2 proviene de la central térmica de As Pontes mientras que el NO_x puede ser debido a la central o a otros posibles focos como por ejemplo, el tráfico. Por tanto si se produce un episodio de alteración de calidad del aire y se observa que NO_x tiene un comportamiento parecedido a SO_2 entonces, probablemente, el origen del episodio de calidad del aire sea la central térmica.

En primer lugar, se realizará un estudio descriptivo de las variables que se van a estudiar. En segundo lugar, se estudiará la relación entre SO_2 y NO_x mediante un modelo de regresión lineal de diseño fijo y heterocedástico, dónde SO_2 es la variable independiente y NO_x la variable dependiente. Para ello,se realizarán varias estimaciones de la densidad de los coeficientes del modelo de regresión; mediante la teoría clásica y mediante los bootstraps uniforme y Wild bootstrap. También se realizará la construcción de modelos no paramétricos de regresión utilizando los estimadores de Nadaraya-Watson y Local-lineal. Finalmente, se interpretarán los modelos de regresión construídos como una regla de predicción y se calcula el error real, el error aparente y el optimismo esperado mediante Bootstrap uniforme. Además, se proporcionará una buena aproximación teórica del error cometido.

A la hora de interpretar las reglas de predicción hay que tener en cuenta que la muestra representa un tipo de episodios de calidad de aire; loa causado por la central térmica por tanto los resultados obtenidos proporcionaran información en esas situaciones.

2. Análisis descriptivo de las variables de inmisión de SO_2 y NO_x

En primer lugar, observamos si,efectivamente, los valores de NO_x siguen a los de SO_2. Para ello representamos los datos y construímos el modelo de regresión (Figura 1), de donde concluímos que éste explica un 80 %. Por lo que hay dependencia entre SO_2 y NO_x. Además, en este caso es razonablemente lineal. A continuación, se representa el gráfico de dispersión de las variables en estudio (Figura 2) y la tabla de los estadístiscos más representativos de las variables SO_2 y NO_x(Tabla 1) . A partir del gráfico de dispersión se observa una tendencia creciente y a partir de la tabla se observa que las variables tienen datos missing (-1) por tanto estos deben ser eliminados. Además también se puede observar que las medias de las dos variables de estudio son muy diferentes, pues la media de NO_2 es mucho mayor que la de la variable SO_2, lo que implica una diferencia de escala.

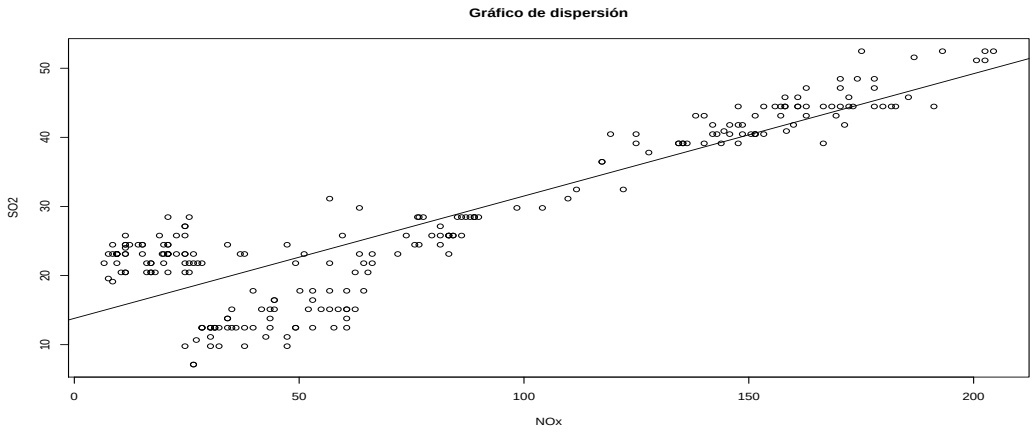

Figura 1: Gráfico de dispersión de NO_x frente a SO_2

Valores de interés	SO_2	NO_x
Mínimo	-1	-1
1º Cuantil	20.46	25.55
Mediana	24.46	58.68
Media	27.46	77.78
3ºCuantil	40.47	140.07
Maximo	52.47	204.42

Cuadro 1: Valores de los estadísticos de las variiables de inmisión SO_2 y NO_x

Completamos el estudio con la estimación de las densidades de las variables de inmisión de SO_2 (Figura3)y NO_x(Figura4) mediante el histograma (1),el histograma movil (y el estimador de

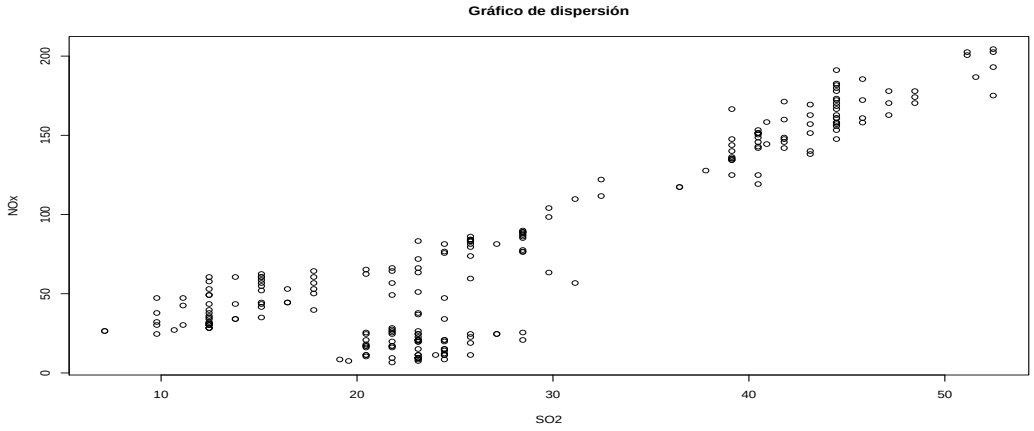

Figura 2: Gráfico de dispersión de NO_x frente a SO_2

Parzen-Rosemblat (3). A la vista de estos dos gráficos, se concluye que la estimación más suave es la proporcionada por el estimador de Parzen-Rosemblat. Además tras realizar el test de Shapiro-Wilks se obiene que ni SO_2 sigue una distribución normal (p.valor= 2.824e-09) ni NO_x sigue una distribución norma (p.valor=1.395e-12)

$$\hat{f}_h(x) = \frac{\#(x_i \in A_j)}{\frac{max(x_i)-min(x_i)}{N}}, \forall x \in I_j, j \in \{1, ..., N\} \tag{1}$$

$$\hat{f}_h(x) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{h} K(\frac{x - x_i}{h}), \tag{2}$$

siendo $K(\frac{x-x_i}{h}) = \frac{1}{2} I_{x_i \in (x-h, x+h)}$

$$\hat{f}_h(x) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{h} K(\frac{x - x_i}{h}), \tag{3}$$

Notemos que h es el paramétro de suavización o ventana y se escoge de tal manera que minimice el error cuadrático medio asintótico (AMISE) y K es una función tipo núcleo que no es más que una función de densidad cualquiera; en este caso N(0,1)

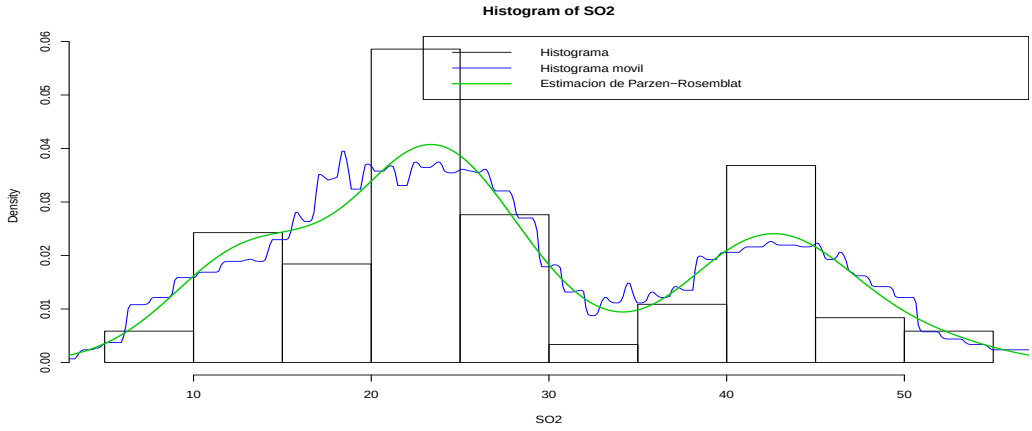

Figura 3: Estimación de la densidad de SO_2

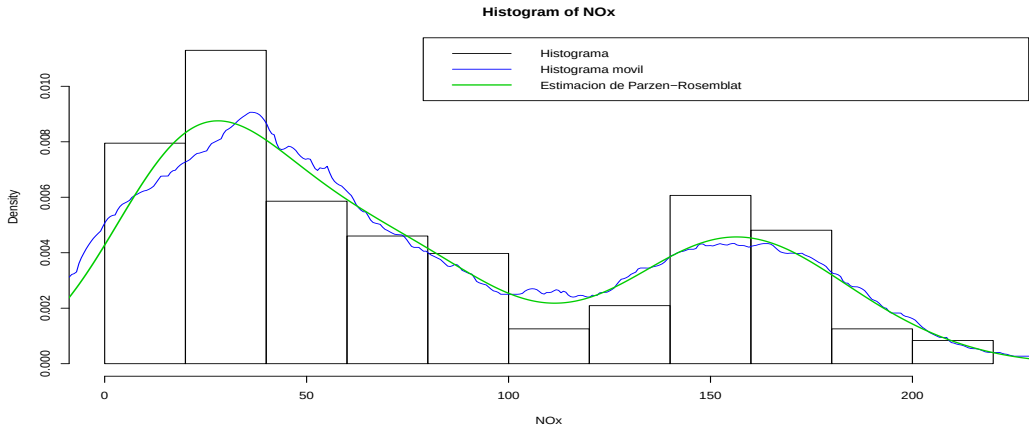

Figura 4: Estimación de la densidad de NO_x

3. Estudio de modelos de regresión paramétricos y no parametricos de NO_x frente a SO_2

En este trabajo, nos interesa analizar la relación existente entre dos variables, X=SO_2e Y=NO_x. El análisis de regresión estudia de que forma NO_x (la variable dependiente) se puede explicar a partir de SO_2. Si NO_x depende de SO_2 entonces NO_x = m(SO_2),donde m es una función. El análisis de la información empírica disponible nos debería de proporcionar información sobre m.

$$E(Y|X = x) = \frac{\int yf(x, y)}{f_X(x)} \tag{4}$$

3.1. Modelo de regresión paramétrico. Cálculo de la densidad de sus coeficientes mediante la Teoría Clásica, Bootstrap y Wild-Bootstrap

En regresión paramétrica habitualmente se supone que m depende linealmente de un vector de parámetros. En nuestro caso particular, hemos empleado la regresión lineal simple, por tanto hemos supuesto que m(SO_2) = a + bSO_2. Construimos el modelo de regresión paramétrica (4) aunque dicho modelo no ajusta bastante bien a los datos pues explica el 80 % pues $R^2 = 0{,}78$; con lo que concluímos que es un buen modelo para modelizar la relación entre NO_x y SO_2, es decir, para explicar NO_x a partir de SO_2.

$$NO_x = -43{,}6821 + 4{,}4098 * SO_2 \tag{5}$$

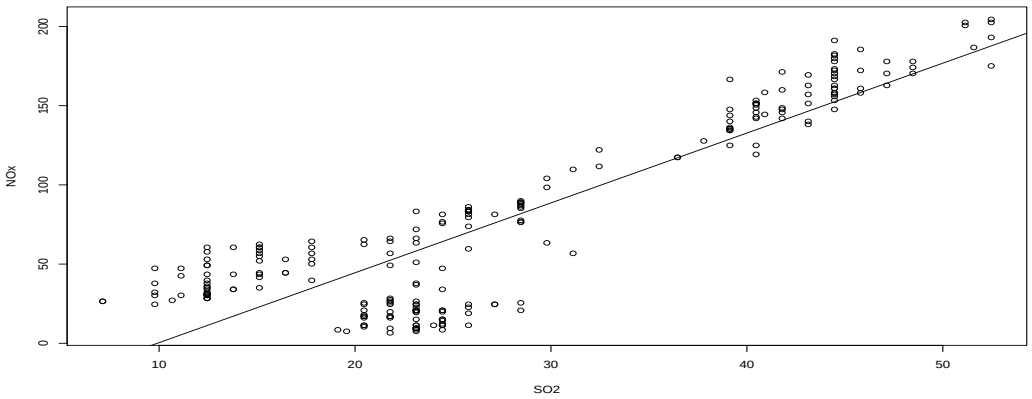

Figura 5: Modelo de regresion

En (Figura 6)se puede observar que la Teoría Clásica nos proporciona densidades normales. En este caso, la teoría dice que Wild Bootstrap funciona mejor que Bootstrap uniforme. Lo que es debido a que el modelo de regresión que estamos considerando es heterocedástico, lo que significa

que la varianza del error es función de la parte predictora del dato. Luego, el Bootstrap uniforme es inadecuado ya que no disponemos de n datos undependientes e identicamente distribuídos, sino de un único dato para cada uno de los n errores correspondientes; es decir, no disponemos de una muestra de n datos i.i.d. sino de n muestras de un dato. A pesar de todo esto no podemos afirmar nada ya que no tenemos las densidades reales con las que comparar.

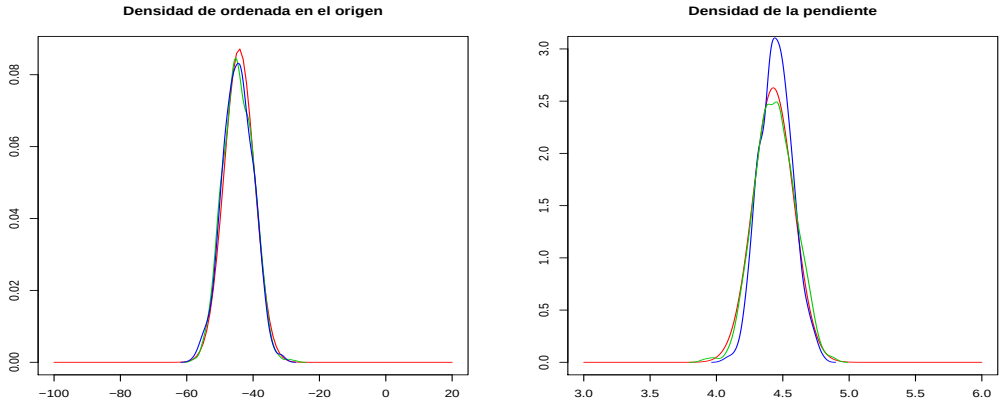

Figura 6: Densidades de los coeficientes estimadas por la Teoría Clásica (roja), Bootstrap uniforme (verde) y Wild Bootstrap (azul)

3.2. Construcción de modelos de regresión no-paramétricos: Nadaraya-Watson, Local-lineal

El modelo anterior es muy restrictivo, pues no permite que cambios en el aumento de NO_x, puesto que este se incrementa hasta un cierto nivel a partir del cual baja o se mantiene estable y vuelve a aumentar. En los modelos de regresión no paramétricos no se impone ninguna restricción a priori sobre m. Obviamente existe un precio a pagar por esta flexibilidad.

Para obtener un estimador no paramétrico de la regresión, basta con estimar m. Para ello habrá que estimar $f_{SO_2}(x)$ y $f(x, y)$. Para estimar la densidad bivariante f(x, y) es habitual emplear el estimador tipo núcleo con núcleo producto.

$$\hat{f}_{n,K}(x, y) = \frac{1}{n}\sum_{j=1}^{n} K_h(x - X_j)K_g(y - Y_j) \tag{6}$$

donde X=SO_2 e Y=NO_x y x son valores de SO_2 e y son valores de NO_x

Por tanto el estimador del numerador de la media condicional sería

$$\hat{m}_{n,K}(x, y) = \frac{1}{n}\sum_{j=1}^{n} K_h(x - X_j)Y_j \tag{7}$$

Tanto el estimador Nadaraya-Watson como el estimador lineal-local son estimadores lineales. Luego, aplicaremos un resultado general para estimadores lineales. Sea

$$\hat{m}(x) = \sum_{j=1}^{n} l_j(x)Y_j \tag{8}$$

un estimador lineal. Entonces la función de validación cruzada para un estimador lineal se define como

$$CV(h) = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{m}_{-(i)}(X_i))^2 \tag{9}$$

Dónde

$$\hat{m}_{(-i)}(x) = \sum_{j=1}^{n} l_{j,(-i)}(x)Y_j \tag{10}$$

con

$$l_{j,(-i)}(x) = \begin{cases} 0 si j = i \\ \dfrac{l_j(x)}{\sum_{k\neq i} l_k(x)} si j \neq i \end{cases} \tag{11}$$

Teorema

$$CV(h) = \frac{1}{n}\sum_{i=1}^{n}(\frac{Y_i - \hat{m}(X_i)}{1 - L_{ii}})^2 \tag{12}$$

donde L_{ii} es el elemento i-de la diagonal de la matriz de suavizado L necesaria para calcular el estimador en los puntos $(X_1, ..., X_n)$. Es decir

$$L_{ii} = l_i(x_i) \tag{13}$$

3.2.1. Estimador Nadaraya-Watson

El estimador de la función m resultante de reemplazar las cantidades desconocidas por sus estimadores en la fórmula de la esperanza condicional fue propuesto por Nadaraya y Watson en 1964

$$\hat{m}_{n,K}(x) = \frac{\sum_{j=1}^{n} K_h(x - X_j)Y_j}{\sum_{k=1}^{n} K_h(x - X_j)} = \sum_{j=1}^{n} W_{h,j}(x)Y_j \tag{14}$$

donde $K_h(u) = \frac{1}{h}K(\frac{u}{h})$ y siendo $W_{h,j}$

$$W_{h,j}(x) = \frac{K_h(x - X_j)}{\sum_{k=1}^{n} K_h(x - X_j)} \tag{15}$$

Por tanto el estimador tipo núcleo de la función de regresión es una media (local) ponderada de los valores observados de la variable Y donde

$$\sum W_{h,j}(x) = 1 \tag{16}$$

Una posibilidad para elegir el parámetro de suavizado, es usar el método de validación cruzada convenientemente adaptado al contexto de regresión.

$$\sum CV(h) = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{m}_{n,K}(X_i))^2 \tag{17}$$

Para medir la bondad de ajuste que se consigue con una ventana h podríamos usar el error cuadrático medio. Esta medida de error global, aproximaría el error de predicción. Sin embargo, la aproximación sería un tanto optimista ya que estaríamos usando el valor de Y_i dos veces: una a la hora de medir el error, y otra a la hora de construir el estimador.

Para evaluar mejor el error de predicción se suele eliminar el dato i-ésimo cuando calculamos el error de predicción para Y_i. Así la función de validación cruzada se define como

$$\sum CV(h) = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{m}_{-(i),K}(X_i))^2 \tag{18}$$

donde $\hat{m}_{-(i),K}$ denota el estimador de Nadaraya-Watson construido a partir de la muestra original después de eliminar el par (X_i, Y_i). La idea sería tomar aquel h que haga que CV sea mínimo. Aunque se podría calcular directamente CV , esto requeriría evaluar, para cada h, n veces el estimador de Nadaraya-Watson,construido a partir de una muestra de (n - 1) puntos. Muchos de estos cálculos serían redundantes y se pueden simplificar.

Teorema

La función de validación cruzada del estimador de Nadaraya-Watson se puede escribir de la siguiente forma

$$CV(h) = \frac{1}{n}\sum_{i=1}^{n}(\frac{Y_i - \hat{m}_{n,K}(X_i)}{1 - L_{ii}})^2 \tag{19}$$

donde L_{ii} es el elemento i-de la diagonal de la matriz de suavizado L necesaria para calcular el estimador en los puntos $(X_1, ..., X_n)$. Es decir,

$$L_{ii} = \frac{K(0)}{\sum_{k=1}^{n} K(X_i - X_k/h)} \tag{20}$$

Notemos que K es una función de densidad a la cual se le denomina función núcleo y en nuestro caso particular hemos escogido núcleo gausiano y h es el paramétro de suavizado o ventana, que la escogemos mediante validación cruzada, es decir, tomamos h de modo que haga que CV sea mínimo. En nuestro caso particular h=0.84. Por tanto, en estas condiciones hemos obtenido la estimación no paramétrica de la regresión (Figura 13)

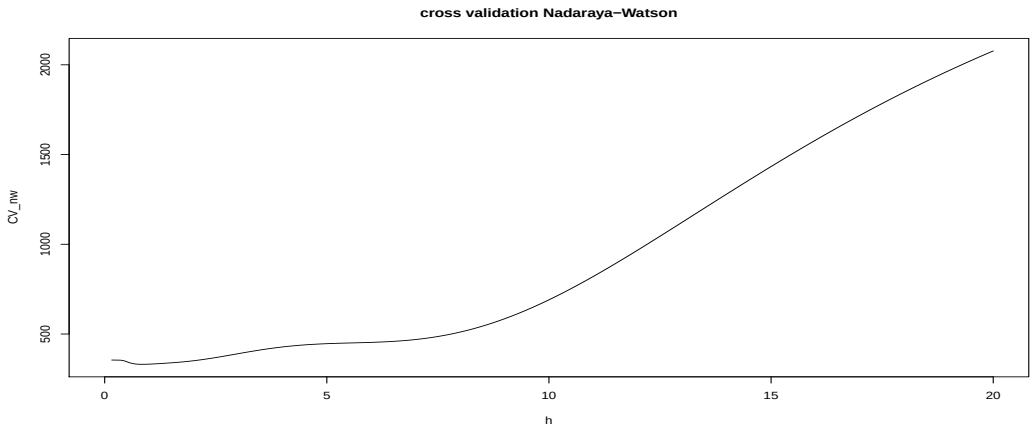

Figura 7: Función de validación cruzada para el estimador Nadaraya-Watson

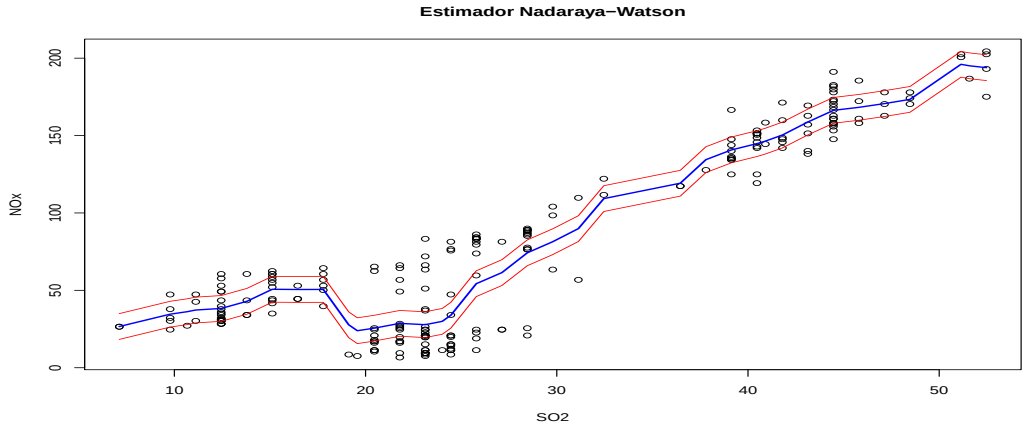

Figura 8: Estimación de la regresión dada por Nadaraya-Watson. Bandas de confianza(rojo)

3.2.2. Estimador Local-lineal

La idea de este método es muy sencilla. En lugar de hacer un ajuste global por mínimos cuadrados de una recta podemos intentar buscar una recta que ajuste bien sólo en los puntos próximos a x. Dado h >0 podemos proponer un modelo lineal válido sólo en el entorno (x - h, x + h)

$$Y_i = \alpha(x) + \beta(x)X_i + e_i; siendo X_i \in (x-h, x+h) \tag{21}$$

El estimador lineal local en el punto x vendrá dado por

$$\hat{m}_{n,LL}(x) = a(x) + b(x)x \tag{22}$$

donde a(x), b(x) son los valores que minimizan la suma de cuadrados ponderada

$$\sum_{i=1}^{n}(Y_i - \alpha(x) - \beta(x)X_i)K_h(x - X_i) \tag{23}$$

donde donde K una función de densidad unimodal y simétrica alrededor del cero que proporciona diferentes pesos a los errores del intervalo (x - h, x + h),dependiendo de su proximidad a x.

Al igual que ocurría con el estimador de Nadaraya-Watson, el estimador lineal-local también es un estimador lineal. Por tanto, la función de validación cruzada es bastante sencilla de calcular.

Teorema

El estimador local lineal se puede escribir de la forma

$$\hat{m}_{n,LL}(x) = \sum_{j=1}^{n} l_j(x)Y_j, \tag{24}$$

donde $l_j(x) = \frac{b_j(x)}{\sum_{k=1}^{n} b_k(x)}$ con

$$b_j(x) = K(\frac{x_j - x}{h})(S_{n,2}(x) - (x - x_j)S_{n,1}(x)), \tag{25}$$

y

$$S_{n,r}(x) = \sum_{j=1}^{n} K(\frac{x_j - x}{h})(x - x_j)^r, r = 1, 2 \tag{26}$$

Notemos que hemos construído el estimador Local-lineal seleccionando la ventana por validación cruzada, h=2.38,(Figura 10)

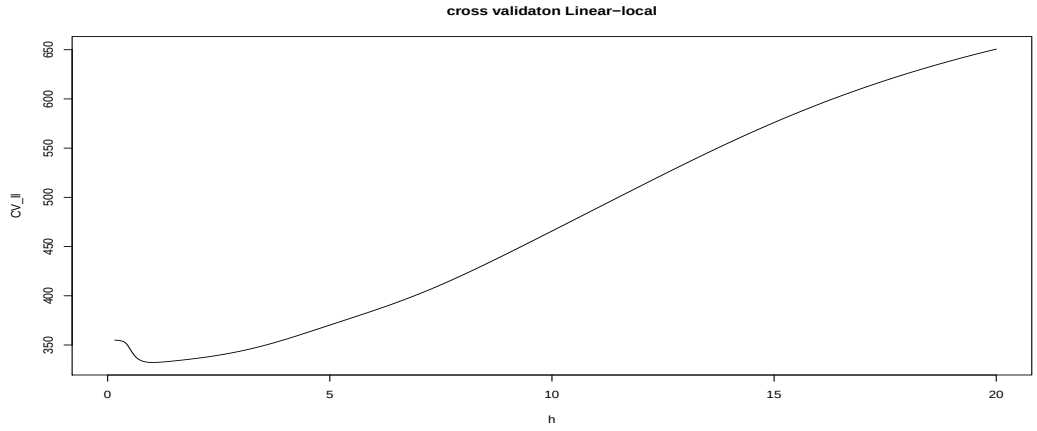

Figura 9: Función de validación cruzada para el estimador Local-lineal

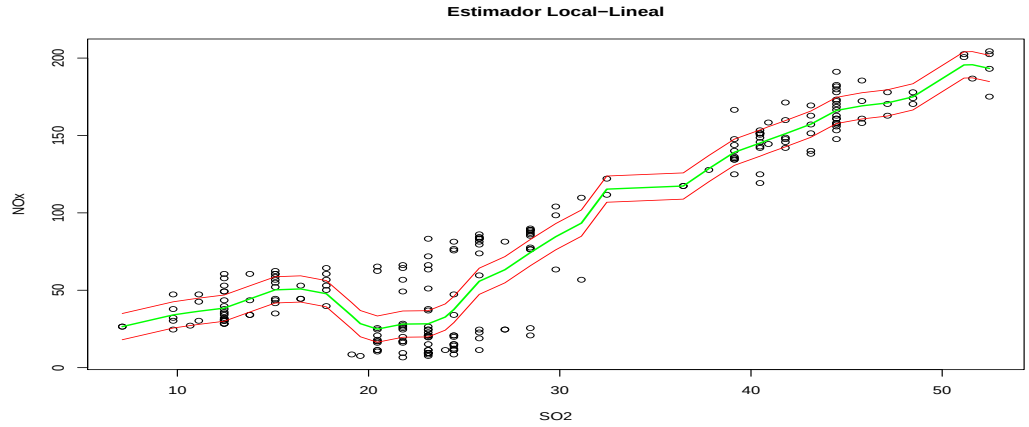

Figura 10: Estimación de la regresión dada por Local-lineal con ventana calculada mediante VC. Bandas de confianza (rojo)

3.3. Comparación de las estimaciones paramétricas y no-paramétricas de la regresión

En (Figura 11) se observa que las mejores estimaciones son las dadas por los estimadores Nadaraya-Watson y Local-lineal, ambos con el parámetro de suavizado escogido mediante validación cruzada.

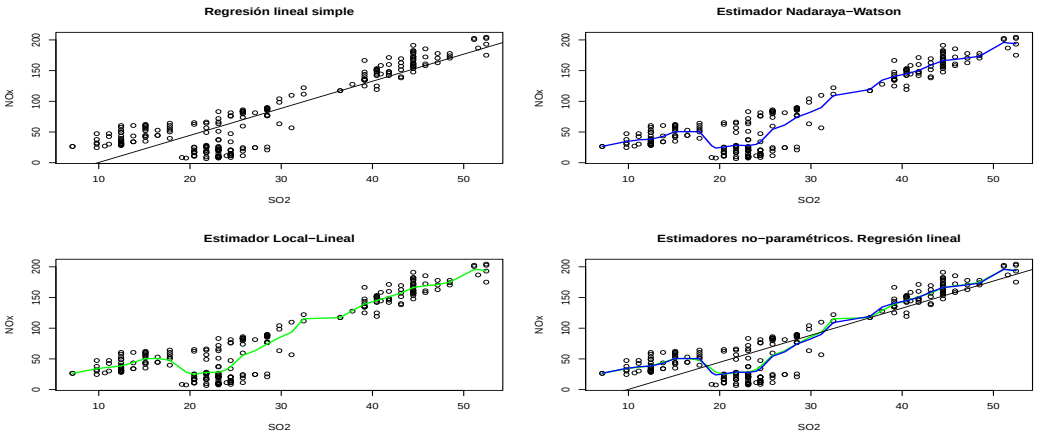

Figura 11: Estimaciones no-paramétricas: Nadaraya-Watson(azul), Local-lineal VC(verde). Estimación paramétrica: Regresión lineal simple(negro)

4. Interpretación de los modelos de regresión construídos como reglas de predicción

Hasta ahora, hemos trabajado con un problema de estimación dónde el objetivo es fijo, aunque es desconocido. Por tanto, la única fuente de variabilidad reside en los datos. A continuación, transformamos el problema de estimación anterior en un problema de predicción. En este nuevo problema existen dos fuentes de variabilidad los datos y el propio objetivo, pues este último es una variable aleatoria. por tanto, a la hora de estimar el error de una regla de predicción habrá que tener en cuenta las dos fuentes de variabilidad. A la hora de cuantificar el error de la regla de de predicción,$y = \eta(t, \vec{x}), \vec{x} = (t, x)$ se obtiene mediante los siguientes términos:

- **Función de pérdida** Cuantifica el error cometido al predecir NO_x mediante el modelo de regresión considerado, en general, el error cometido al predecir y_0 con $\eta(t, \bar{x})$ viene dado por

$$Q(y_0, \eta(t, \vec{x})) = (y_0 - \eta(t, \vec{x}))^2 \tag{27}$$

- **Error verdadero de la regla de predicción** Es el valor esperado de la función de pérdida respecto de NO_x con el modelo de regresión considerado, en general, es el valor esperado de la función de la función de pérdida respecto de (Y_0, T_0).

$$Err(\vec{x}; F) = E_F\left[(Y_0 - \eta(T_0, \vec{x}))^2\right] \tag{28}$$

- **Error aparente de la regla de predicción** Es la evaluación del error verdadero sobre los datos muestrales. Notemos que tiende a infraestimar el error cometido pues la regla de predicción ha sido reconstruída con dichos datos.

$$err(\vec{x}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \eta(t_i, \vec{x}))^2 \tag{29}$$

- **Optimismo de la regla de predicción** Es la diferencia entre el error verdadero y el aparente.

$$op(\vec{x}; F) = Err(\vec{x}, F) - err(\vec{x}) \tag{30}$$

- **Optimismo esperado de la regla de predicción** Es el valor esperado del optimismo respecto de los datos de SO_2.

$$w(F) = E_F[op(\vec{X}; F)] \tag{31}$$

Notemos que en nuestro caso particular no conocemos F por tanto usaremos remuestreo Bootstrap uniforme para obtener una aproximación de las cantidades definidas anteriormente. En primer lugar obtenemos réplicas mediante el Bootstrap uniforme y con cada una de ellas obtenemos $\eta(t*, \vec{x}^{*b})$ por tanto la ecuación (27) se transfoma en $Q(Y*_0, \eta(t*_0, \vec{x}^{*b}))$. Luego, ya podemos obtener una aproximación para cada remuestra Bootstrap del error real (28) $Err(\vec{x}^{*b}; \hat{F}) = \frac{1}{n}\sum_{i=1}^{n} Q(y_i, \eta(t_i, \vec{x}^{*b}))$ del error aparente (29) $err(\vec{x}^{*b}) = \sum_{i=1}^{n} \frac{(x_j^{*b} == x_i)}{n} Q(y_i, \eta(t_i, \vec{x}^{*b}))$ y del optmismo de la regla de predicción (30) $op(\vec{x}^{*b}; \hat{F}) = \sum_{i=1}^{n} \frac{1}{n} - \frac{(x_j^{*b} == x_i)}{n} Q(y_i, \eta(t_i, \vec{x}^{*b}))$

4.1. Construcción de la regla de predicción mediante el modelo de regresión lineal simple

En primer lugar, definimos la regla de predicción como se había definido la recta de regresión lineal simple (5) solo que en esta ocasión la variable $y = NO_x$ es desconocida, por lo tanto, nuestro objetivo es predecir los valores de NO_x a partir de los valores de SO_2. Estimamos el error real, el error aparente y el optimismo mediante Bootstrap(Tabla 2). Podemos decir que una buena estimación del error real es el optimismo Bootstrap más el error aparente, 16.00752+ 768.3317=784.3392. Notemos que dicho argumento es el que se utilizará para obtener las estimaciones de los errores reales dados por las reglas de predicción no paramétricas.

A continuación representamos las rectas de regresión con cada una de las replicas Bootstrap. Dónde podemos observar que estas no recogen el comportamiento no lineal de los datos.

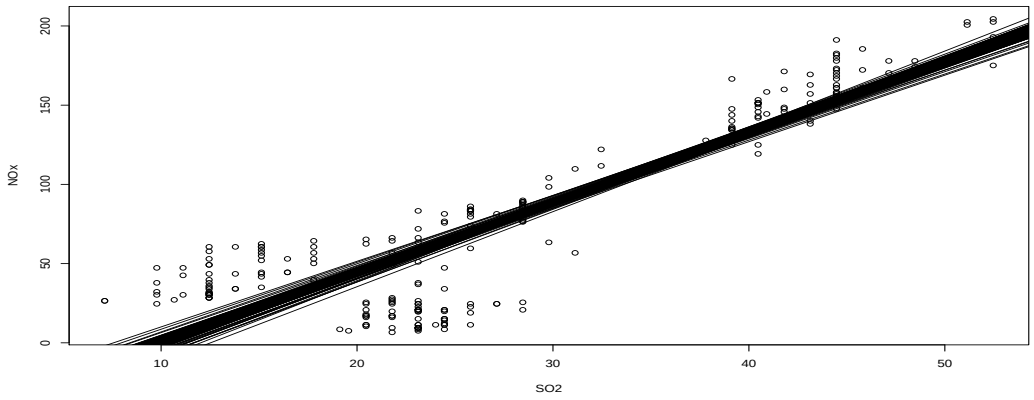

Figura 12: Función de validación cruzada para el estimador Local-lineal

n=36	Error real Bootstrap	error aparente Bootstrap	optimismo Bootstrap	error aparente
Media(B=100)	774.7842	758.7767	16.00752	768.3317

Cuadro 2: Valores del error real, error aparente y optimismo de la regla de predicción estimados mediante Bootstrap y error aparente real

4.2. Construcción de la regla de predicción mediante el modelo de regresión dado por el estimador de Nadaraya-Watson

A continuación, usamos la estimación del modelo de regresión dada por Nadaraya-Watson(Figura 13) como regla de prección. De esta forma, obtenemos los resultados expuestos a continuación (Cuadro 3).

Representamos, tambien en este caso, las réplicas Bootstrap; las cuales si recogen la no linealidad de nuestros datos. Sin embargo, observamos dificultades en la parte central dónde los datos presentan mayor variabilidad.

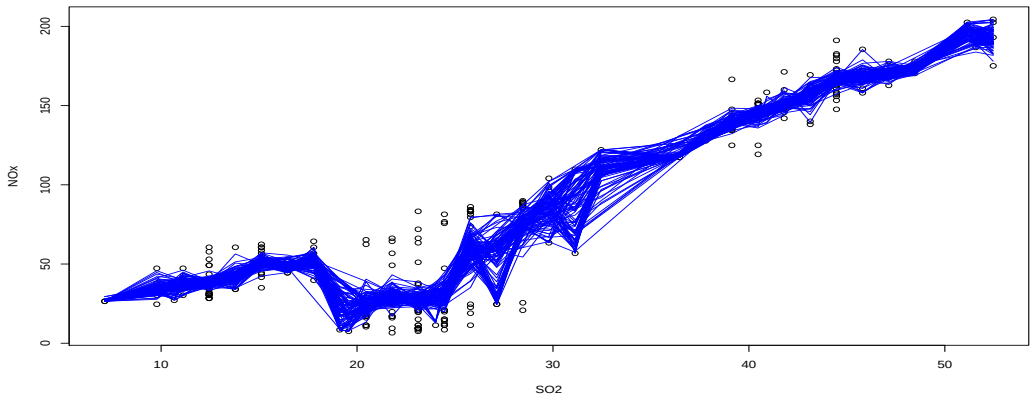

Figura 13: Construcción del estimador Nadaraya-Watson para la regresión con cada una de las réplicas Bootstrap

n=36	Error real Bootstrap	error aparente Bootstrap	optimismo Bootstrap	error aparente
Media(B=100)	307.2037	254.2634	52.94036	289.5022

Cuadro 3: Valores del error real, error aparente y optimismo de la regla de predicción estimados mediante Bootstrap y error aparente real

4.3. Construcción de la regla de predicción mediante el modelo de regresión dado por el estimador Local-lineal

Finalmente usamos como regla de predicción la dada por la estimación Local-lineal del modelo de regresión (Figura 10); obteniendo los siguientes resultados (Cuadro 4).

Representamos las replicas Bootstrap que se comportan de forma análoga a las descritas en 4.2.

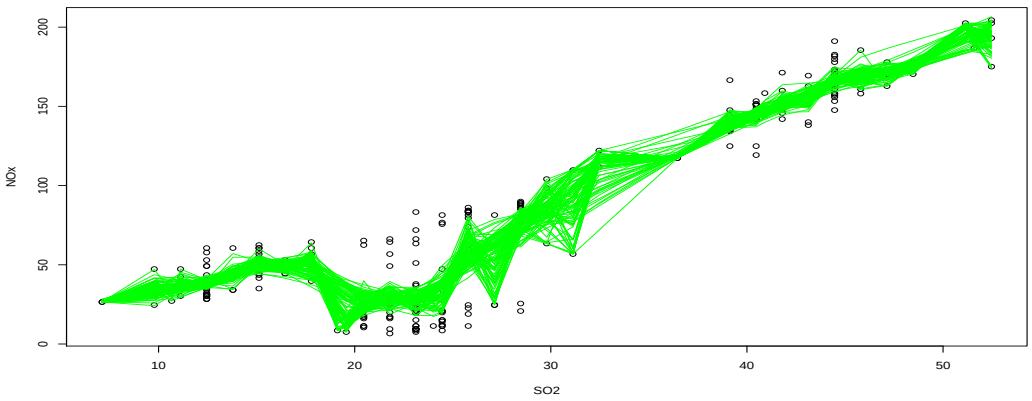

Figura 14: Construcción del estimador Local-lineal para la regresión con cada una de las réplicas Bootstrap

	Empresa	Incio	Tamaño	Media
Desv. típica	Coef. Asimetría	Coef. curtosis		
Log-returns	Gas natural	02/01/200	2854	-0.0002
0.0179	-0.3769	6.8867		

Cuadro 4: Valores del error real, error aparente y optimismo de la regla de predicción estimados mediante Bootstrap y error aparente real

4.4. Comparación de las tres reglas de predicción

En (Tabla 5) podemos observar que las mejores reglas de predicción son las dadas por las estimaciones no-parámericas de los modelos de regresión. Además estas reglas son las que proporcionan menores estimaciones del error real (Tabla 6).

n=36	Error real Bootstrap	error aparente Bootstrap	optimismo Bootstrap	e. aparente
Regresión lineal	774.7842	758.7767	16.00752	768.3317
Nadaraya-Watson	307.2037	254.2634	52.94036	289.5022
Local-lineal	309.2613	262.1891	47.07215	293.4831

Cuadro 5: Valores del error real, error aparente y optimismo de la regla de predicción estimados mediante Bootstrap y error aparente real

n=36	ESTIMACIÓN ERROR REAL
Regresión lineal	784.3392
Nadaraya-Watson	342.4426
Local-lineal	340.5553

Cuadro 6: Valores del Error real Bootsrap

5. Implementación en R

```
setwd("C:/Users/leyenda/Desktop/master/Remuestreo/Trabajo")
datos<- read.delim("C:/Users/leyenda/Desktop/master/Remuestreo/Trabajo/Datos_G2.txt",
dec=".", sep="\t",fill=TRUE,na.strings="NA",header=TRUE)
attach(datos)
names(datos)
summary(datos)
dim(datos)
ind1<-which(SO2==-1)
ind2<-which(NOx==-1)
SO2<-SO2[-ind1]
NOx<-NOx[-ind2]
windows()
plot(NOx[order(NOx)],SO2[order(NOx)],xlab="NOx",ylab="SO2", main="Gráfico de dispersión")
abline(lm(SO2[order(NOx)]~NOx[order(NOx)]))
summary(lm(SO2[order(NOx)]~NOx[order(NOx)]))
windows()
plot(NOx[order(NOx)],SO2[order(NOx)],xlab="NOx",ylab="SO2", main="Gráfico de dispersión")

##########################################
# Estimación de las funciones de densidad #
##########################################
windows()
hist(SO2,freq=FALSE)
lines(density(SO2,kernel="rectangular"),col=4,xlim=c(0,300),ylim=c(0.0,0.5))
lines(density(SO2),lwd=2,xlim=c(0,300),ylim=c(0.0,0.5),col=3)
legend("topright",legend=c("Histograma","Histograma movil",
"Estimacion de Parzen-Rosemblat"),lwd=c(1,1,2),col=c(1,4,3))
shapiro.test(SO2)
windows()
hist(NOx,freq=FALSE)
lines(density(NOx,kernel="rectangular"),col=4,xlim=c(0,300),ylim=c(0.0,0.5))
lines(density(NOx),lwd=2,xlim=c(0,300),ylim=c(0.0,0.5),col=3)
legend("topright",legend=c("Histograma","Histograma movil",
"Estimacion de Parzen-Rosemblat"),lwd=c(1,1,2),col=c(1,4,3))
shapiro.test(NOx)
```

```
####################################
# Modelos de regresión             #
####################################

# Modelo de regresión lineal simple #
####################################

## Cálculo de los coeficientes del modelo de regresion
regresion<-lm(NOx[order(SO2)]~SO2[order(SO2)])
windows()
plot(SO2[order(SO2)],NOx[order(SO2)],xlab="SO2",ylab="NOx")
abline(regresion)
#plot(regresion)
summary(regresion)
attach(regresion)
names(regresion)

# Estimaciones no-paramétricas:Nadaraya-Watson Local-lineal#
###########################################################

##########################################
#       Funciones auxilares              #
##########################################
K<-function(mas){
#--Núcleo gausiano
return(1/sqrt(2*pi)*exp(-mas^2/2))}
S<-function(t,mas,bandwidth,exponent){
#-- t vector de puntos dónde S será evaluada.
#-- mas valores de la variable independiente.
#-- bandwidth ventana.
#-- r exponent.
   return(diag(t(K(outer(mas,t,"-")/bandwidth))%*%(outer(mas,t,"-")^exponent)))}
```

```
##########################################################
#     Nadaraya-Watson estimator                          #
##########################################################
nw<-function(t,mas,y,bandwidth){
#-- Esta función devuelve la estimación calculada por el estimador no-paramétrico
#- de regresión Nayadara-Watson.
#-- t vector de puntos dónde evaluamos.
#-- mas valores de la variable independiente.
#-- y valores de la variable independiente.
#-- bandwidth es la ventana.
    return(apply(as.matrix(bandwidth),1,function(x)(K(outer(t,mas,"-")/x)/
    apply(K(outer(t,mas,"-")/x),1,sum))%*%y))}

##########################################################
#     Local-lineal estimator                             #
##########################################################
EstLL<-function(t,mas,y,bandwidth){
#-- Esta función devuelve la estimación calculada por el estimador
#- no-paramétrico de regresión Local-lineal.
#-- t vector de puntos dónde evaluamos.
#-- mas valores de la variable independiente.
#-- y valores de la variable independiente.
#-- bandwidth es la ventana.
return(apply(as.matrix(bandwidth),1,function(h){(t(K(outer(mas,t,"-")/h))*
(S(t,mas,h,2)-t(outer(mas,t,"-"))*S(t,mas,h,1))/apply(t(K(outer(mas,t,"-")/h))*
(S(t,mas,h,2)-t(outer(mas,t,"-"))*S(t,mas,h,1)),1,sum ))%*%y}))}
```

```
####################################
##                                ##
##   Validación cruzada           ##
##                                ##
####################################
cv_nw<-function(mas,y,bandwidth){
#-- Es la función de validación cruzada aplicada al estimador
#- no-paramétrico de la regresión Nadaraya-Watson.
    return(apply(as.matrix(bandwidth),1,function(x){1/length(mas) *
    sum( ((y - nw(mas,mas,y,x)) / (1 - (K(0)/
    apply( K(outer(mas,mas,"-")/x) ,1,sum) ) ) )^2 )}))}
cv_ll<-function(mas,y,bandwidth){
#--Es la función de validación cruzada aplicada al estimador
no-paramétrico de la regresión Local-lineal.
return(apply(as.matrix(bandwidth),1,function(h){1/length(mas)*
sum(((as.numeric(y -EstLL(mas,mas,y,h)))/(1-( K(0)*S(mas,mas,h,2)/
apply(t(K(outer(mas,mas,"-")/h))*(S(mas,mas,h,2)
-t(outer(mas,mas,"-"))*S(mas,mas,h,1)),1,sum))))^2)}))}

##- Secuencia de ventanas
h<-seq(0,20,by=0.02)
##--Se guardan los valores de la función de validación cruzada aplicada
a los estimadores no-paramétricos:Nadaraya-Watson y Local-lineal.
CV_nw<-numeric()
CV_nw<-cv_nw(SO2[order(SO2)],NOx[order(SO2)],h)
CV_ll<-numeric()
CV_ll<-cv_ll(SO2[order(SO2)],NOx[order(SO2)],h)


##-- Representación gráfica de la función de validación cruzada aplicada
#-a los estimadores no-paramétricos: Nadaraya-Watson y Local-lineal.
windows()
plot(CV_nw~h,type="l",main="cross validation Nadaraya-Watson")
windows()
plot(CV_ll~h,type="l",main="cross validaton Linear-local")
```

```
##-- Estimador de la regresión Nadaraya-Watson con la ventana de validación cruzada.
nadaraya_watson<-nw(SO2[order(SO2)],SO2[order(SO2)],NOx[order(SO2)],
h[which(CV_nw==min(CV_nw,na.rm=T))])
h_opt_nw<-h[which(CV_nw==min(CV_nw,na.rm=T))] # 0.84
##-- Estimador de la regresión Local-lineal con la ventana de validación cruzada.
local_lineal<-EstLL(SO2[order(SO2)],SO2[order(SO2)],NOx[order(SO2)],
h[which(CV_ll==min(CV_ll,na.rm=T))])
h_opt_ll<-h[which(CV_ll==min(CV_ll,na.rm=T))] # 1.02

#####################################################################
#           Representación gráfica                                  #
#####################################################################
windows()
plot(SO2[order(SO2)],NOx[order(SO2)],xlab="SO2", ylab="NOx",
main="Estimador Nadaraya-Watson")
lines(nadaraya_watson~SO2[order(SO2)],col="blue",lwd=2)
#CONSTRUCCIÓN DE INTERVALOS DE CONFIANZA
y0<-sort(NOx)
y1<-y0[-1]
y2<-y0[-length(NOx)]
sigma<-(1/(2*(length(y0)-1)))*sum((y1-y2)^2)
alpha=0.05
L_nw<-K(outer(SO2[order(SO2)],SO2[order(SO2)],"-")/h_opt_nw)/
apply(K(outer(SO2[order(SO2)],SO2[order(SO2)],"-")/h_opt_nw),1,sum)
inf<-nadaraya_watson-qnorm(1-alpha/2)*sqrt(sum(L_nw^2)*sigma)
sup<-nadaraya_watson+qnorm(1-alpha/2)*sqrt(sum(L_nw^2)*sigma)
lines(inf~SO2[order(SO2)],col="red")
lines(sup~SO2[order(SO2)],col="red")
windows()
plot(SO2[order(SO2)],NOx[order(SO2)],xlab="SO2", ylab="NOx",
main="Estimador Local-Lineal")
lines(local_lineal~SO2[order(SO2)],col="green",lwd=2)
#CONSTRUCCIÓN DE INTERVALOS DE CONFIANZA
y0<-sort(NOx)
y1<-y0[-1]
y2<-y0[-length(NOx)]
sigma<-(1/(2*(length(y0)-1)))*sum((y1-y2)^2)
alph_opt_lla=0.05
L_ll<-t(K(outer(SO2[order(SO2)],SO2[order(SO2)],"-")/h_opt_ll))*
(S(SO2[order(SO2)],SO2[order(SO2)],h_opt_ll,2)-
t(outer(SO2[order(SO2)],SO2[order(SO2)],"-"))*
S(SO2[order(SO2)],SO2[order(SO2)],h_opt_ll,1))/
```

```
apply(t(K(outer(SO2[order(SO2)],SO2[order(SO2)],"-")/h_opt_ll))*
(S(SO2[order(SO2)],SO2[order(SO2)],h_opt_ll,2) -
t(outer(SO2[order(SO2)],SO2[order(SO2)],"-")) * S(SO2[order(SO2)],
SO2[order(SO2)],h_opt_ll,1)),1,sum )
inf_ll<-local_lineal-qnorm(1-alph_opt_lla/2)*sqrt(sum(L_ll^2)*sigma)
sup_ll<-local_lineal+qnorm(1-alph_opt_lla/2)*sqrt(sum(L_ll^2)*sigma)
lines(inf_ll~SO2[order(SO2)],col="red")
lines(sup_ll~SO2[order(SO2)],col="red")
windows()
par(mfrow=c(2,2))
plot(SO2[order(SO2)],NOx[order(SO2)],xlab="SO2", ylab="NOx",
main="Regresión lineal simple")
abline(lm(NOx[order(SO2)]~SO2[order(SO2)]))
plot(SO2[order(SO2)],NOx[order(SO2)],xlab="SO2", ylab="NOx",
main="Estimador Nadaraya-Watson")
lines(nadaraya_watson~SO2[order(SO2)],col="blue",lwd=2)
plot(SO2[order(SO2)],NOx[order(SO2)],xlab="SO2", ylab="NOx",
main="Estimador Local-Lineal")
lines(local_lineal~SO2[order(SO2)],lwd=2,col="green")
plot(SO2[order(SO2)],NOx[order(SO2)],xlab="SO2", ylab="NOx",
main="Estimadores no-paramétricos. Regresión lineal")
lines(local_lineal~SO2[order(SO2)],lwd=2,col="green")
lines(nadaraya_watson~SO2[order(SO2)],col="blue",lwd=2)
abline(regresion)
```

```
#############################################################################
#WILD BOOTSTRAP. REGRESION LINEAL (Y=a+bx: Modelo de diseño fijo y heterocedástico). #
#Estimación de la densidad con teoría clásica, bootstraps uniforme y Wild bootstrap. #
#############################################################################
x<-SO2
y<-NOx
Varx<-var(x)*(length(x)-1)/length(x)
n<-length(SO2)
#TEORÍA CLÁSICA
a0<-mean(y)-mean(x)*(cov(x,y)/Varx)
b0<-cov(x,y)/Varx
windows()
plot(x,y)
abline(a=a0,b=b0)
u<-y-a0-b0*x
e<-u^2
Vare<-(n/(n-2))*mean(e)
#BOOTSTRAP UNIFORME Y WILD BOOTSTRAP
muestra<-numeric(n)
wmuestra<-numeric(n)
muestra=u-mean(u)
yboot<-numeric(n)
ywboot<-numeric(n)
B=1000
aboot<-numeric(B)
bboot<-numeric(B)
awboot<-numeric(B)
bwboot<-numeric(B)
for (k in 1:B){
l<-sample(1:n,replace=TRUE)
yboot=a0+b0*SO2+muestra[l]
aboot[k]<-mean(yboot)-mean(x)*(cov(x,yboot)/Varx)
bboot[k]<-cov(x,yboot)/Varx
for (i in 1:n){
r<-sample(c(u[i]*(1-sqrt(5))/2,u[i]*(1+sqrt(5))/2),replace=TRUE,
prob=c((5+sqrt(5))/10,1-(5+sqrt(5))/10))
wmuestra[i]=r[1]}
ywboot=a0+b0*x+wmuestra
awboot[k]<-mean(ywboot)-mean(x)*(cov(x,ywboot)/Varx)
bwboot[k]<-cov(x,ywboot)/Varx}
za<-seq(-100,20,length(n))
```

```
windows()
par(mfrow=c(1,2))
plot(za,dnorm(za,mean=a0,sd=sqrt((Vare/n)*(1+(mean(x)^2)/Varx))),col=2,xlab="",
ylab="",type="l",main="Densidad de ordenada en el origen")
lines(density(aboot),col=3, ylim=c(0,0.04))
lines(density(awboot),col=4)
zb<-seq(3,6,by=0.01)
plot(zb,dnorm(zb,mean=b0,sd=(sqrt(Vare/(n*Varx)))),col=2,xlab="",
ylab="", ylim=c(0,3),type="l",main="Densidad de la pendiente")
lines(density(bwboot),col=4)
lines(density(bboot),col=3)
```

```
############################################################
# REGLA DE PREDICCIÓN: Modelo de regresión lineal simple #
############################################################
B=100
Errbu<-numeric(B)
errbu<-numeric(B)
opbbu<-numeric(B)
t<-SO2
y<-NOx
mean((y-a0-b0*t)^2)
windows()
plot(SO2[order(SO2)],NOx[order(SO2)],xlab="SO2",ylab="NOx")

for(b in 1:B){
l<-sample(1:n,replace=TRUE)
tbu<-t[l]
ybu<-y[l]
cuantos<-numeric(n)
for(i in 1:n){
cuantos[i]<-length(which(tbu==t[i]& ybu==y[i]))}

Vartbu<-var(tbu)*(length(tbu)-1)/length(tbu)
a0bu<-mean(ybu)-mean(tbu)*(cov(tbu,ybu)/Vartbu)
b0bu<-cov(tbu,ybu)/Vartbu
abline(a=a0bu,b=b0bu)
ubu<-y-a0bu-b0bu*t
ebu<-ubu^2
Errbu[b]<-mean(ebu)
ubbu<-ybu-a0bu-b0bu*tbu
ebbu<-ubbu^2
errbu[b]<-mean(ebbu)
#errbu[b]=sum((cuantos/n)*ebbu)
opbbu[b]<-Errbu[b]-errbu[b]
}
op_Bootstrap<-mean(opbbu)
err_Bootstrap<-mean(errbu)
Err_Bootstrap<-mean(Errbu)
op_Bootstrap
err_Bootstrap
Err_Bootstrap
```

```
############################################################
# REGLA DE PREDICCIÓN: Modelo de regresión Nadaraya-Watson #
############################################################
B=100
Errbu<-numeric(B)
errbu<-numeric(B)
opbbu<-numeric(B)
t<-SO2
y<-NOx
n=length(t)
mean((y[order(t)]-as.numeric(nadaraya_watson))^2)
windows()
plot(SO2[order(SO2)],NOx[order(SO2)],xlab="SO2",ylab="NOx")
for(b in 1:B){
l<-sample(1:n,replace=TRUE)
tbu<-t[l]
ybu<-y[l]
CV_nwbu<-cv_nw(tbu[order(tbu)],ybu[order(tbu)],h)
est_bu0<-nw(t[order(t)],tbu[order(t)],ybu[order(t)],
h[which(CV_nwbu==min(CV_nwbu,na.rm=T))])
est_bu<-nw(tbu[order(tbu)],tbu[order(tbu)],ybu[order(tbu)],
h[which(CV_nwbu==min(CV_nwbu,na.rm=T))])
lines(est_bu~tbu[order(tbu)],col="blue")
ubu<-y[order(t)]-as.numeric(est_bu0)
ebu<-ubu^2
Errbu[b]<-mean(ebu)
ubbu<-ybu[order(tbu)]-as.numeric(est_bu)
ebbu<-ubbu^2
errbu[b]<-mean(ebbu)
opbbu[b]<-Errbu[b]-errbu[b]
}
op_Bootstrap_NAD<-mean(opbbu)
err_Bootstrap_NAD<-mean(errbu)
Err_Bootstrap_NAD<-mean(Errbu)
op_Bootstrap_NAD
err_Bootstrap_NAD
Err_Bootstrap_NAD
```

```
#############################################################
# REGLA DE PREDICCIÓN: Modelo de regresión Local-lineal    #
#############################################################
B=100
Errbu<-numeric(B)
errbu<-numeric(B)
opbbu<-numeric(B)
t<-SO2
y<-NOx
n=length(SO2)
mean((y[order(t)]-as.numeric(local_lineal))^2)
windows()
plot(SO2[order(SO2)],NOx[order(SO2)],xlab="SO2",ylab="NOx")
for(b in 1:B){
l<-sample(1:n,replace=TRUE)
tbu<-t[l]
ybu<-y[l]
CV_llbu<-cv_ll(tbu[order(tbu)],ybu[order(tbu)],h)
est_ll_bu0<-EstLL(t[order(t)],tbu[order(t)],ybu[order(t)],
h[which(CV_llbu==min(CV_llbu,na.rm=T))])
est_ll_bu<-EstLL(tbu[order(tbu)],tbu[order(tbu)],ybu[order(tbu)],
h[which(CV_llbu==min(CV_llbu,na.rm=T))])
lines(est_ll_bu~tbu[order(tbu)],col="green")
ubu<-y[order(t)]-as.numeric(est_ll_bu0)
ebu<-ubu^2
Errbu[b]<-mean(ebu)
ubbu<-ybu[order(tbu)]-as.numeric(est_ll_bu)
ebbu<-ubbu^2
errbu[b]<-mean(ebbu)
opbbu[b]<-Errbu[b]-errbu[b]
}
op_Bootstrap_LL<-mean(opbbu)
err_Bootstrap_LL<-mean(errbu)
Err_Bootstrap_LL<-mean(Errbu)
op_Bootstrap_LL
err_Bootstrap_LL
Err_Bootstrap_LL
```