

Inferencia Conformal

Mauro Loprete y Maximiliano Saldaña

1/1/22

Tabla de contenidos

1	Introducción	3
2	Inferencia conformal	4
2.1	Un resultado previo	4
2.2	Método <i>naive</i> de construcción de intervalos	4
2.3	Intervalos de predicción conformales	5
2.3.1	Teorema	6
2.4	Intervalos de predicción conformales con muestras separadas	7
2.4.1	Teorema	7
2.4.2	Teorema	8
2.5	Intervalos conformales con múltiples separaciones de la muestra	8
2.6	Intervalos predictivos mediante Jackknife	9
3	Aplicación	10
3.1	Método simple	13
3.2	Split	16
3.3	Paquete conformalInference	18
3.3.1	Implementación de un modelo de regresión lineal	19
3.3.2	Ejemplo para implementar Nadaraya–Watson	20
3.3.3	Ejemplo regresión múltiple	23
4	Conclusión	26
5	Bibliografía	27
6	TODO AGREGAR BIBLIOGRAFIA y PAQUETES DE R	28
7	Anexo	29

1 Introducción

En Lei et al. (2016) los autores plantean un marco general para realizar inferencia predictiva sin supuestos distribucionales en un contexto de regresión, empleando la *inferencia conformal*. Mediante la metodología planteada se pueden obtener intervalos de confianza con validez en muestra finitas (no asintótica) para una variable de respuesta, empleando cualquier estimador de la función de regresión.

El problema se plantea de la siguiente manera: Se considera $Z_1, \dots, Z_n \sim F$ i.i.d., donde $Z_i = (X_i, Y_i)$ es una variable aleatoria en $\mathbb{R}^d \times \mathbb{R}$, Y_i es la variable de respuesta y $X_i = X_i(1) \dots, X_i(d)$ son las covariables. Se tiene la función de regresión:

$$\mu(x) = E(Y|X = x), \quad x \in \mathbb{R}^d$$

Es de interés predecir la nueva respuesta Y_{n+1} a las covariables X_{n+1} , sin hacer supuestos sobre μ o F . Dado un nivel de cobertura α , el objetivo es construir un intervalo de predicción $C \subseteq \mathbb{R}^d \times \mathbb{R}$ basado en Z_1, \dots, Z_n que cumpla:

$$P(Y_{n+1} \in C(X_{n+1})) \geq 1 - \alpha$$

En esta expresión se supone que $Z_{n+1} = (X_{n+1}, Y_{n+1})$ proviene también de la distribución F y $C(x) = \{y \in \mathbb{R} : (x, y) \in C\}$, $x \in \mathbb{R}^d$

2 Inferencia conformal

La idea básica de la inferencia conformal, dadas las definiciones de la introducción, consiste en que para decidir si un valor y está incluido en el intervalo $C(X_{n+1})$ consideramos poner a prueba la hipótesis nula de que $Y_{n+1} = y$ y se construye un valor-p válido basado en los cuantiles empíricos de la muestra aumentada Z_1, \dots, Z_n, Z_{n+1} .

2.1 Un resultado previo

Sean U_1, \dots, U_n una muestra i.i.d de una variable aleatoria continua. Para un nivel de no cobertura $\alpha \in (0, 1)$ y una observación U_{n+1} , nótese que:

$$P(U_{n+1} \leq \hat{q}_{1-\alpha}) \geq 1 - \alpha \quad (2.1)$$

Donde $\hat{q}_{1-\alpha}$ es el cuantil de la muestra U_1, \dots, U_n definido por:

$$\hat{q}_{1-\alpha} = \begin{cases} U_{(\lceil (n+1)(1-\alpha) \rceil)} & \text{si } \lceil (n+1)(1-\alpha) \rceil \leq n \\ \infty & \text{en caso contrario} \end{cases} \quad (2.2)$$

Aquí $U_{(1)} \leq \dots \leq U_{(n)}$ son los estadísticos de orden de la muestra. Se verifica la cobertura en muestra finita de la Ecuación 2.1: dada la independencia de las variables, el rango de U_{n+1} en la muestra se distribuye uniforme en el conjunto $\{1, \dots, n+1\}$, entonces

2.2 Método *naive* de construcción de intervalos

Usando el resultado previo de la sección anterior y en el contexto de regresión planteado en la Sección Capítulo 1, un método sencillo para contruir un intervalo predictivo para Y_{n+1} ante el valor X_{n+1} es:

$$C_{naive}(X_{n+1}) = [\hat{\mu}(X_{n+1}) - \hat{F}_n^{-1}(1 - \alpha), \hat{\mu}(X_{n+1}) + \hat{F}_n^{-1}(1 - \alpha)] \quad (2.3)$$

donde $\hat{\mu}$ es un estimador de la función de regresión, \hat{F}_n la distribución empírica de los residuos dentro de la muestra $|Y_i - \hat{\mu}(X_i)|$, $i = 1, \dots, n$ y $\hat{F}_n^{-1}(1 - \alpha)$ el cuantil $1 - \alpha$ de \hat{F}_n .

Este método es aproximadamente válido para muestras grandes, bajo la condición de que $\hat{\mu}$ sea lo suficientemente preciso, es decir, que $F_n^{-1}(1 - \alpha)$ esté cerca del cuantil $1 - \alpha$ de $|Y_i - \mu(X_i)|$. Para que esto se cumpla en general es necesario el cumplimiento de condiciones de regularidad tanto para la distribución F de los datos y para $\hat{\mu}$, como que el modelo esté correctamente especificado.

Un problema de este método es que los intervalos pueden presentar una considerable subcobertura, dado que se están empleando los residuos dentro de la muestra. Para subsanar esto, en Lei et al. (2016) se plantea la metodología de los intervalos de predicción conformales.

2.3 Intervalos de predicción conformales

Para cada valor $y \in \mathbb{R}$ se construye un estimador de regresión aumentado $\hat{\mu}_y$, el cual se estima en el conjunto de datos aumentado $Z_1, \dots, Z_n, (X_{n+1}, y)$. Luego, se define:

$$R_{y,i} = |Y_i - \hat{\mu}_y(X_i)|, \quad i = 1, \dots, n \quad (2.4)$$

$$R_{y,n+1} = |y - \hat{\mu}_y(X_{n+1})| \quad (2.5)$$

Con el rango de $R_{y,n+1}$ entre los demás residuos de la muestra $R_{y,1}, \dots, R_{y,n}$ se calcula:

$$\pi(y) = \frac{1}{n+1} \sum_{i=1}^{n+1} \mathbb{I}\{R_{y,i} \leq R_{y,n+1}\} = \frac{1}{n+1} + \frac{1}{n+1} \sum_{i=1}^n \mathbb{I}\{R_{y,i} \leq R_{y,n+1}\} \quad (2.6)$$

que es la proporción de los puntos de la muestra aumentada cuyos residuos dentro de la muestra son más pequeños que el residuo $R_{y,n+1}$. Como los datos son i.i.d. y suponiendo la simetría de $\hat{\mu}$, se puede apreciar que el estadístico $\pi(Y_{n+1})$ se distribuye uniforme en $1/(n+1), 2/(n+1), \dots, 1$, lo cual implica¹:

$$P((n+1)\pi(Y_{n+1}) \leq \lceil (1-\alpha)(n+1) \rceil) \geq 1-\alpha \quad (2.7)$$

Esta expresión se puede interpretar como que $1 - \pi(Y_{n+1})$ da un valor-p válido conservador para la prueba de hipótesis donde $H_0)Y_{n+1} = y$.

Aplicando dicha prueba sobre todos los posibles valores de $y \in \mathbb{R}$, la ecuación Ecuación 2.7 lleva al intervalo de predicción conformal evaluado en X_{n+1} :

$$C_{conf}(X_{n+1}) = [y \in \mathbb{R} : (n+1)\pi(Y_{n+1}) \leq \lceil (1-\alpha)(n+1) \rceil] \quad (2.8)$$

¹Ver por qué

Cada vez que se quiere obtener un intervalo de predicción en un nuevo conjunto de covariables se tienen que recalcular los pasos Ecuación 2.4, Ecuación 2.5, Ecuación 2.6 y Ecuación 2.8. En la práctica, se restringen los valores de y a una grilla discreta.

El procedimiento para obtener el intervalo se puede resumir en el Algoritmo 1.

Algoritmo 1: Intervalo de predicción conformal

Entrada: Datos (X_i, Y_i) , $i = 1, \dots, n$, nivel de no cobertura $\alpha \in (0, 1)$, algoritmo de regresión \mathcal{A} , puntos \mathcal{X}_{nuevo} en los que construir intervalos de predicción y valores $\mathcal{Y}_{prueba} = \{y_1, y_2, \dots\}$ para comparar con la predicción.

Salida: Intervalos de predicción, en cada elemento de \mathcal{X}_{nuevo}

```

1 for  $x \in \mathcal{X}_{nuevo}$  do
2   for  $y \in \mathcal{Y}_{prueba}$  do
3      $\hat{\mu}_y = \mathcal{A}(\{(X_1, Y_1), \dots, (X_n, Y_n), (x, y)\})$ 
4      $R_{y,i} = |Y_i - \hat{\mu}_y(X_i)|$ ,  $i = 1, \dots, n$  y  $R_{y,n+1} = |y - \hat{\mu}_y(x)|$ 
5      $\pi(y) = (1 + \sum_{i=1}^n \mathbb{I}\{R_{y,i} \leq R_{y,n+1}\}) / (n + 1)$ 
6    $C_{conf}(x) = [y \in \mathbb{R} : (n + 1)\pi(Y_{n+1}) \leq \lceil (1 - \alpha)(n + 1) \rceil]$ 
7 Se devuelve  $C_{conf}(x)$  para cada  $X \in \mathcal{X}_{nuevo}$ .
```

2.3.1 Teorema

El intervalo Ecuación 2.8 tiene cobertura válida para muestras finitas por construcción y a su vez no presenta sobre cobertura. Esto se puede expresar mediante las expresiones Ecuación 2.9 y Ecuación 2.10, respectivamente:

Sea (X_i, Y_i) , $i = 1, \dots, n$ v.a. i.i.d, entonces para la nueva observación i.i.d. (X_{n+1}, Y_{n+1}) :

$$P(Y_{n+1} \in C_{conf}(X_{n+1})) \geq 1 - \alpha \quad (2.9)$$

Adicionalmente, si se hace el supuesto que para todo $y \in \mathbb{R}$ los residuos dentro de la muestra $R_{y,i} = |Y_i - \hat{\mu}_y(X_i)|$, $i = 1, \dots, n$ tienen una distribución conjunta continua se cumple que:

$$P(Y_{n+1} \in C_{conf}(X_{n+1})) \leq 1 - \alpha + \frac{1}{n + 1} \quad (2.10)$$

Observación. Nótese que las probabilidades aquí, al tomarse sobre la muestra aumentada i.i.d. implican cobertura promedio (o marginal). Esto no es lo mismo que la cobertura condicional $P(Y_{n+1} \in C_{conf}(x) | X_{n+1} = x) \geq 1 - \alpha \ \forall \ x \in \mathbb{R}^d$. Esta última es una propiedad más fuerte y no puede lograrse con intervalos predictivos de amplitud finita sin que el modelo y el estimador cumplan condiciones de regularidad y consistencia.

Observación. Si se mejora el estimador $\hat{\mu}$, en general el intervalo de predicción conformal decrece en tamaño. Esto se da debido a que un $\hat{\mu}$ más preciso lleva a residuos más pequeños y los intervalos conformales están definidos por los cuantiles de la distribución aumentada de los residuos.

2.4 Intervalos de predicción conformales con muestras separadas

Un problema práctico de los intervalos de inferencia conformal de la sección anterior es que tienen mucho costo computacional. Para poder concluir si $y \in C_{conf}(X_{n+1})$, para cualquier X_{n+1} y y , se tiene que reestimar el modelo en la muestra aumentada que incluye el nuevo punto X_{n+1} y recalcular y reordenar los nuevos residuos obtenidos.

Para enfrentar esta problemática se puede hacer uso de una metodología denominada por Lei et al. (2016) como predicción conformal separada (*split conformal prediction*). Su costo computacional es menor (es el del paso de estimación únicamente) y tiene menos requerimientos de memoria (solo hay que guardar las variables seleccionadas cuando se evalúa el ajuste en los nuevos puntos X_i , $i \in \mathcal{J}_2$). Se presenta en el Algoritmo 2.

Algoritmo 2: Intervalos de predicción conformales con muestras separadas

Entrada: Datos (X_i, Y_i) , $i = 1, \dots, n$, nivel de no cobertura $\alpha \in (0, 1)$, algoritmo de regresión \mathcal{A} .

Salida: Intervalos de predicción, sobre $x \in \mathbb{R}^d$

- 1 Se separa la muestra al azar en dos subconjuntos de igual tamaño \mathcal{J}_1 e \mathcal{J}_2 .
 - 2 $\hat{\mu}_y = \mathcal{A}(\{(X_i, Y_i) : i \in \mathcal{J}_1\})$
 - 3 $R_i = |Y_i - \hat{\mu}_y(X_i)|$, $i \in \mathcal{J}_2$
 - 4 d es el k -ésimo valor más pequeño en $\{R_i : i \in \mathcal{J}_2\}$, donde $k = \lceil (n/2 + 1)(1 - \alpha) \rceil$
 - 5 Se devuelve $C_{split}(x) = [\hat{\mu} - d, \hat{\mu} + d]$ para todo $x \in \mathbb{R}^d$.
-

2.4.1 Teorema

Sea (X_i, Y_i) , $i = 1, \dots, n$ v.a. i.i.d, entonces para la nueva observación i.i.d. (X_{n+1}, Y_{n+1}) :

$$P(Y_{n+1} \in C_{split}(X_{n+1})) \geq 1 - \alpha \quad (2.11)$$

Adicionalmente, si se hace el supuesto que los residuos R_i , $i \in \mathcal{J}_2$ tienen una distribución conjunta continua se cumple que:

$$P(Y_{n+1} \in C_{split}(X_{n+1})) \leq 1 - \alpha + \frac{2}{n+2} \quad (2.12)$$

2.4.2 Teorema

Los intervalos de predicción conformales con muestras separadas dan una garantía aproximada de cobertura dentro de la muestra. Esto se puede expresar como que existe una constante $c > 0$ tal que, para cualquier $\epsilon > 0$:

$$P \left(\frac{2}{n} \sum_{i \in \mathcal{J}_2} \mathbb{I}\{Y_I \in C_{split}(X_i) - (1 - \alpha) \geq \epsilon\} \right) \leq 2 \exp(-cn^2(\epsilon - 4/n)^2) \quad (2.13)$$

Esto implica cobertura dentro de la muestra para la muestra \mathcal{J}_2 , revirtiendo los roles de \mathcal{J}_1 e \mathcal{J}_2 se puede extender para toda la muestra.

Observación. También se puede aplicar este método con una separación no balanceada de la muestra, con $|\mathcal{J}_1| = \rho n$ e $|\mathcal{J}_2| = (1 - \rho)n$, para $\rho \in (0, 1)$. Esto puede ser útil en situaciones donde el procedimiento de regresión es complejo y puede resultar beneficioso elegir $\rho > 0,5$, para que $\hat{\mu}$ sea más preciso.

2.5 Intervalos conformales con múltiples separaciones de la muestra

Al considerar diferentes divisiones estamos introduciendo una fuente **adicional** de aleatoriedad a nuestra estimación, por tanto, el método de dividir la muestra reduce el costo computacional pero introduce ruido provocando una mayor incertidumbre en nuestras predicciones.

Una manera de corregirlo es combinar las diferentes inferencias realizadas en N particiones independientes, construyendo así, $C_{split,1}, \dots, C_{split,N}$, en donde cada intervalo es construido a un nivel de significación $\alpha^* = 1 - \alpha/N$ y su relación esta dada de la siguiente manera:

$$C_{split}^N(x) = \cap_{j=1}^N C_{split,j}(x) \quad x \in R^d$$

Como se puede apreciar, al utilizar este procedimiento hay un precio que debemos de pagar, los intervalos se vuelven mas anchos a medida que N crece, esto puede verse de la siguiente manera:

Como sabemos, la cobertura marginal de nuestra predicción tiene una cobertura de al menos $1 - \alpha$ por construcción, además con el algoritmo 2 podemos acotar aún mas esta probabilidad:

$$1 - \alpha \leq P(Y_{n+1} \in C_{split}(X_{n+1})) \leq 1 - \alpha + \frac{2}{n+2} \leq 1 \quad (2.14)$$

En nuestro caso, cada partición esta construida con un nivel de significación $\alpha^* = \alpha/N$, por tanto remplazando esta identidad en 2.14, obtenemos la siguiente expresión

$$1 - \frac{\alpha}{N} \leq P(Y_{n+1} \in C_{split}(X_{n+1})) \leq 1 - \frac{\alpha}{N} + \frac{2}{n+2} \leq 1$$

Es por esto que si aumentamos N el término α^* se vuelve aún mas chico y debido a que estamos frente a una medida de probabilidad esta se acerca cada vez más a uno, por tanto, nuestros intervalos se vuelven aún mas estrechos.

2.6 Intervalos predictivos mediante Jackknife

Esta metodología emplea los cuantiles de los residuos de validación cruzada dejando una observación fuera (*leave-one-out*) para calcular los intervalos de predicción.

Algoritmo 3: Intervalo de predicción conformal mediante Jackknife.

Entrada: Datos (X_i, Y_i) , $i = 1, \dots, n$, nivel de no cobertura $\alpha \in (0, 1)$, algoritmo de regresión \mathcal{A} .

Salida: Intervalos de predicción sobre $x \in \mathbb{R}^d$.

- 1 **for** $i \in \{1, \dots, n\}$ **do**
 - 2 $\hat{\mu}^{(-i)} = \mathcal{A}(\{(X_l, Y_l) : l \neq i\})$
 - 3 $R_i = |Y_i - \hat{\mu}^{(-i)}(X_i)|$
 - 4 $d =$ el k -ésimo valor más pequeño en $\{R_i : i \in \{1, \dots, n\}\}$, con $k = \lceil n(1 - \alpha) \rceil$
 - 5 Se devuelve $C_{jack}(x) = [\hat{\mu}(x) - d, \hat{\mu}(x) + d]$ para todo $x \in \mathbb{R}^d$
-

Tiene la ventaja que emplea más de la muestra que se aparta para entrenar cuando se calculan los residuos, lo cual frecuentemente lleva a intervalos de menor amplitud. Como desventaja, los intervalos que se obtienen no garantizan cobertura válida fuera de la muestra cuando se trabaja con muestras finitas e incluso asintóticamente la cobertura depende de condiciones del estimador.

3 Aplicación

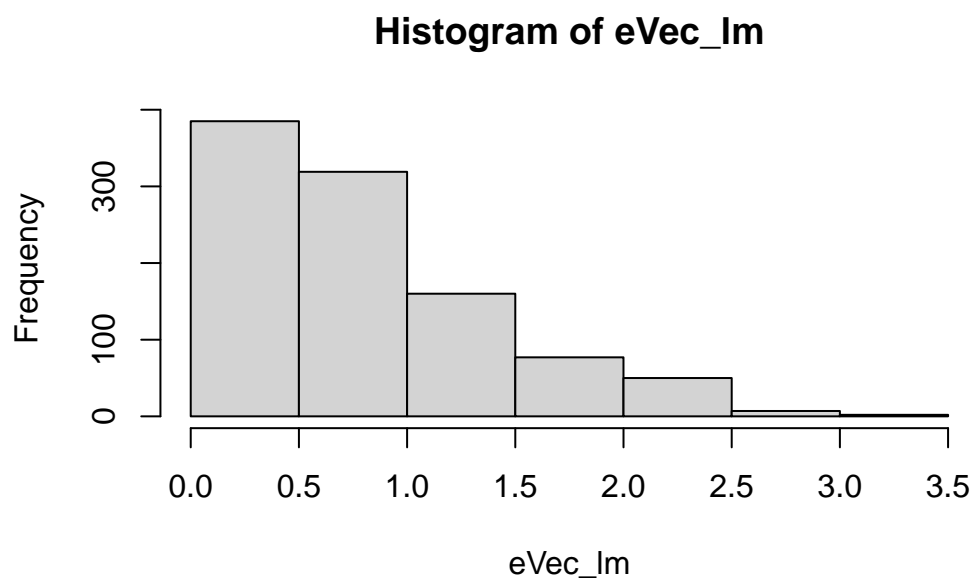
En primera instancia, simulamos un conjunto de datos para realizar la estimación. En este caso es una mixtura de normales de media 0 y varianza 1.

```
# Generamos un conjunto de datos al cual ajustar la regresion
set.seed(12345)
n <- 1000

U <- rnorm(n)
X <- rnorm(n)

Y <- X + U
regData <- data.frame(X,Y)

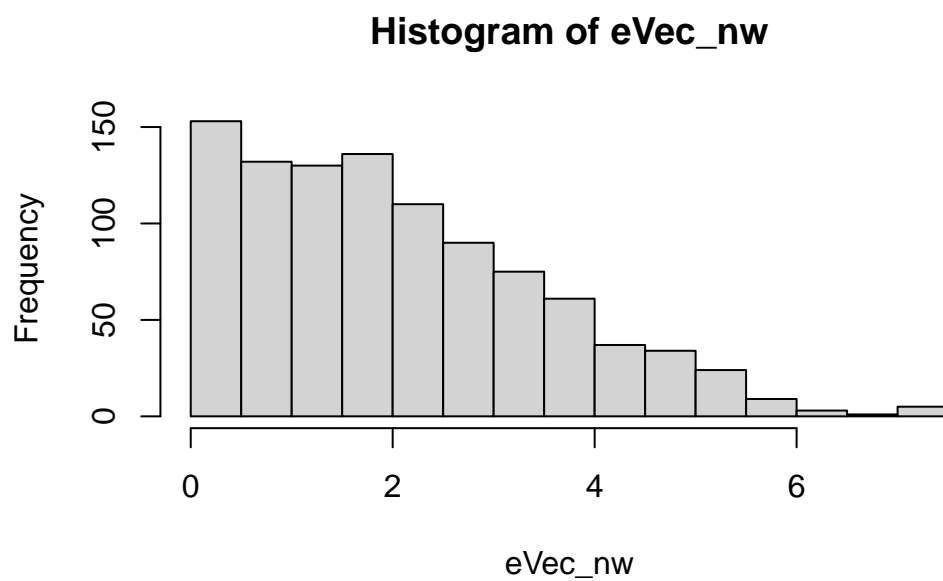
# Estimacion paramétrica
fitlm <- lm(Y~X, data=regData)
eVec_lm <- abs(fitlm$residuals)
hist(eVec_lm)
```



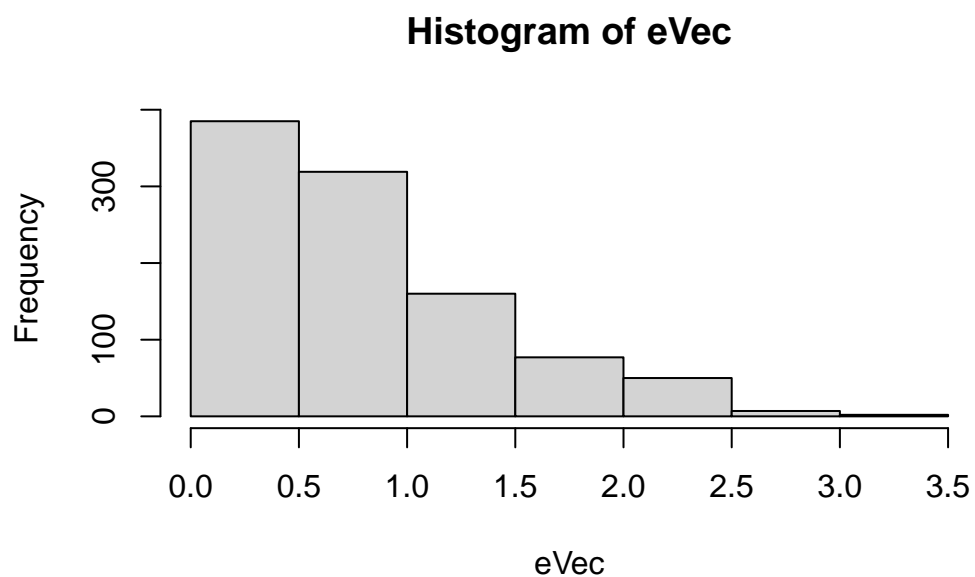
```
# Estimacion no parametrica, Naradaya-Watson
fitnw <- ksmooth(
  X,
  Y,
  kernel = "normal",
  # Seleccionar este parametro mejor
  bandwidth = 0.2
)

eVec_nw <- abs(fitnw$y-Y)

hist(eVec_nw)
```



```
# Estimacion paramétrica  
fitlm <- lm(Y~X, data=regData)  
eVec <- abs(fitlm$residuals)  
hist(eVec)
```



3.1 Método simple

```
plot(X,Y)
Xnew <- -1.5
abline(fitlm)
abline(v=Xnew, lty="dashed")

muHat <- predict(
  fitlm,
  newdata=data.frame(X=Xnew)
)

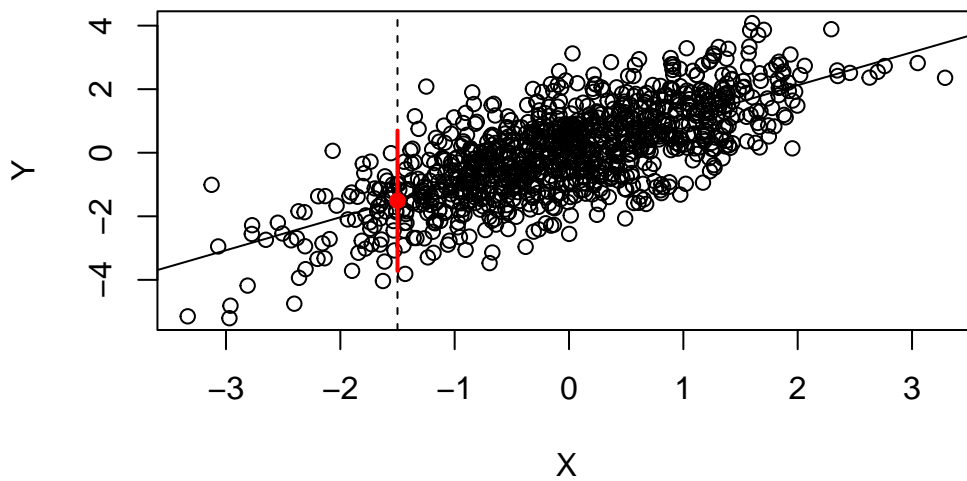
points(
  Xnew,
  muHat,
  pch=19,
  col="red"
)
```

```

C.X <- c(
  muHat - quantile(eVec, .975),
  muHat + quantile(eVec, .975)
)

points(
  rep(Xnew, 2),
  C.X,
  type="l",
  col="red",
  lwd=2
)

```



```

nEval <- 200
yCand <- seq(
  from=min(Y),
  to=max(Y),
  length=nEval
)

```

```

confPredict <- function(y, Xin){

  nData <- nrow(regData)
  regData.a <- rbind(regData,c(Xin, y))

  fitlm.a <- lm(Y~X, data=regData.a)

  resOut <- abs(fitlm.a$residuals)
  resOut_new <- resOut[length(resOut)]

  pi.y <- mean(
    apply(
      as.matrix(resOut),
      1,
      function(x){x<=resOut_new}
    )
  )

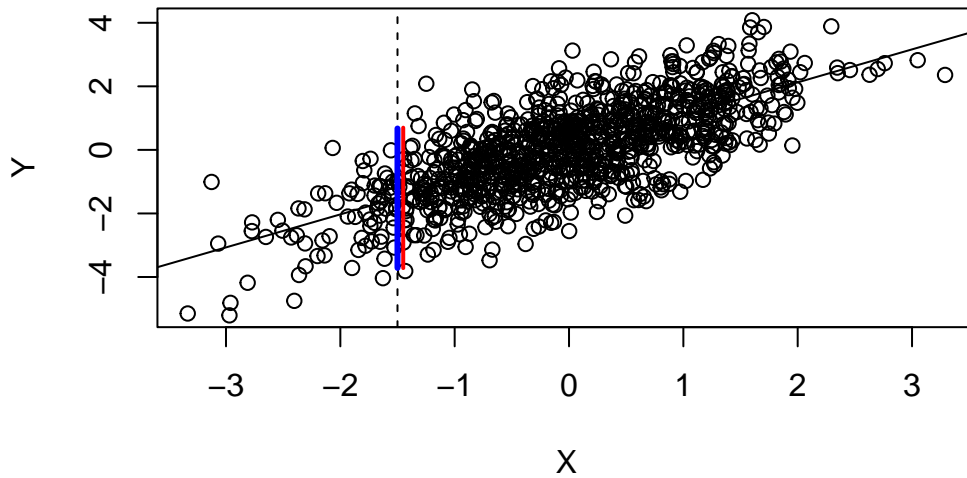
  testResult <- pi.y*(nData+1) <= ceiling(.975*(nData+1))

  return(testResult)
}

Cxa <- range(
  yCand[apply(yCand, confPredict, Xin=Xnew)]
)

plot(X,Y)
abline(fitlm)
abline(v=Xnew, lty="dashed")
points(rep((Xnew+.05), 2), C.X, type="l", col="red", lwd=2)
points(rep(Xnew, 2), Cxa, type="l", col="blue", lwd=3)

```



3.2 Split

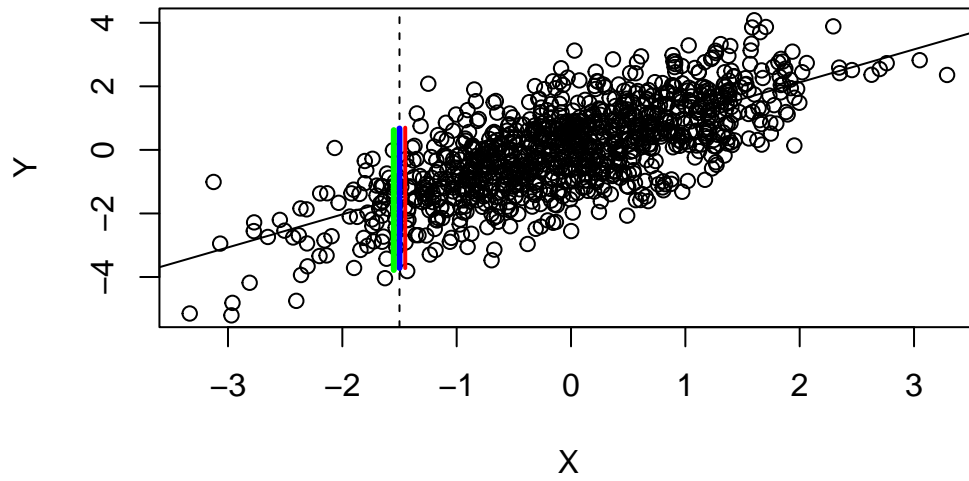
```
splitConfPredict <- function(Xin){
  nData <- nrow(regData)
  regData$index <- 1:nData
  regData$split <- 1
  regData$split[sample(regData$index, floor(nrow(regData)/2), replace=F)] <- 2
  fitlm.spl <- lm(Y~X, data=subset(regData, split==1))
  resOut <- abs(subset(regData, split==2)$Y - predict(fitlm.spl,
                                                       newdata=subset(regData, split==2)))

  kOut <- ceiling(((nData/2)+1)*(.975))
  resUse <- resOut[order(resOut)][kOut]
  Y.hat <- predict(fitlm.spl, newdata=data.frame(X=Xin))
  C.split <- c(Y.hat-resUse,Y.hat+resUse)
  return(C.split)
}

plot(X,Y)
abline(fitlm)
abline(v=Xnew, lty="dashed")
```



```
points(rep((Xnew+.05), 2), C.X, type="l", col="red", lwd=2)  
points(rep(Xnew, 2), C.xa, type="l", col="blue", lwd=3)  
points(rep(Xnew-.05, 2), splitConfPredict(Xnew), type="l", col="green", lwd=3)
```



3.3 Paquete conformalInference

Existe un paquete disponible en CRAN, que ya tiene implementado todas estas herramientas.

En concreto, tenemos la función `conformal.pred` que nos permite realizar predicciones conformales para cualquier modelo de regresión, además de su versión menos costosa en términos computacionales `conformal.split`.

Antes de ir a un ejemplo práctico, veamos que argumentos recibe la función `conformal.pred`:

```
names(formals(conformal.pred))
```

```
[1] "x"           "y"           "x0"          "train.fun"
[5] "predict.fun" "alpha"       "w"           "mad.train.fun"
[9] "mad.predict.fun" "num.grid.pts" "grid.factor" "verbose"
```

Los argumentos de la función son los siguientes:

- `x` -> Vector de variables independientes (`nxp`)
- `y` -> Vector de variables dependientes `n`
- `x0` -> Matriz de variables independientes para predecir un nuevo `y` (`n0xp`)
- `train.fun` -> Función para mejorar la performance del modelo, varía en cada una de sus especificaciones.
- `predcit.fun` -> Función que se utilizará para predecir el nuevo valor de `y`
- `alpha` -> Valor de cobertura
- ...

La documentación entera se encuentra en el siguiente link: <https://github.com/ryantibs/conformal/blob/master/conformalInference.pdf>

Para cada argumento `*.fun` se tienen implementado los siguientes modelos por defecto:

- **Modelo de regresión lineal** (Implementación propia de MCO usando descomposición de Cholesky) `lm.funs()`
- **Random Forest** (via `randomForests`) `rf.funs()`
- **Splines** (via `splines`) `smooth.splines.funs()`
- **Elastic Net** (via `glmnet`) `elastic.funs()`

Para el caso de `train.fun` y `predict.fun` se pueden implementar funciones propias, dependiendo del modelo su implementación puede variar.

3.3.1 Implementación de un modelo de regresión lineal

```
train.fun = function(x,y,out=NULL) {  
  n = nrow(x); p = ncol(x)  
  v = rep(1,p)  
  
  if (intercept) {  
    x = cbind(rep(1,n),x)  
    v = c(0,v)  
  }  
  if (!is.null(out)) {  
    chol.R = out$chol.R  
  }  
  else {  
    chol.R = vector(mode="list",length=m)  
    for (j in 1:m) {  
      chol.R[[j]] = chol(crossprod(x) + lambda[j]*diag(v))  
    }  
  }  
  beta = matrix(0,p+intercept,m)  
  for (j in 1:m) {  
    beta[,j] = chol.solve(chol.R[[j]], t(x) %*% y)  
  }  
  
  return(list(beta=beta,chol.R=chol.R))  
}  
  
predict.fun = function(out,newx) {  
  if (intercept) {  
    newx = cbind(rep(1,nrow(newx)),newx)  
  }  
  return(newx %*% out$beta)  
}
```

3.3.2 Ejemplo para implementar Nadaraya–Watson

```
train.fun_ks <- function(x,y,out=NULL) {  
  
  kernel = "normal"  
  bandwidth = 0.2  
  
  out <- ksmooth(  
    X,  
    Y,  
    kernel,  
    bandwidth = bandwidth  
  )  
  
  list(  
    x = out$x,  
    y = out$y,  
    kernel = kernel,  
    bandwidth = bandwidth  
  )  
}  
  
predict.fun_ks <- function(out, newx) {  
  return(  
    ksmooth(  
      x = out$x,  
      y = out$y,  
      out$kernel,  
      out$bandwidth,  
      x.points = newx  
    )$y  
  )  
}
```

Si bien en el paquete tiene implementado diferentes modelos, como es el ejemplo de la regresión lineal no paramétrica de Nadaraya–Watson, su resultado final no tiene asociado el método `predict` y devuelve una lista. Haciendo las modificaciones correspondientes, es posible utilizarlo en la función `conformal.pred`.

3.3.2.1 Ejemplo de predicción conformal para Nadaraya–Watson

Hagamos una prueba, intentado predecir el valor de Y para $X = -1.5$, como en el ejemplo anterior:

```
res <- conformal.pred(  
  x = X,  
  y = Y,  
  x0 = Xnew,  
  train.fun = train.fun_ks,  
  predict.fun = predict.fun_ks,  
  num.grid.pts = 100  
)  
  
res$pred
```

```
      [,1]  
[1,] -1.590337
```

Ahora, probemos con la versión utilizando particiones:

```
resplit <- conformal.pred.split(  
  x = X,  
  y = Y,  
  x0 = Xnew,  
  train.fun = train.fun_ks,  
  predict.fun = predict.fun_ks  
)  
  
resplit$pred
```

```
      [,1]  
[1,] -1.590337
```

Como sabemos, el resultado es el mismo. Elegir un método u otro impactará en la estimación de los intervalos de predicción, veamos cada uno de ellos:

```
intervalo <- function(puntual, lo, uper) {  
  glue::glue(  
    "[{puntual} - {lo}; {puntual} + {uper}]",  
  )  
}
```

```

    lo = round(puntual - lo,2),
    uper = round(uper - puntual,2),
    puntual = round(puntual,2)
  )
}

```

```

# Intervalo con el método conformal

```

```

intervalo(
  res$pred,
  res$lo,
  res$up
)

```

```

[-1.59 - -1.13;-1.59 + 6.92]

```

```

# Intervalo con el método de split

```

```

intervalo(
  resplit$pred,
  resplit$lo,
  resplit$up
)

```

```

[-1.59 - 2.95;-1.59 + 2.95]

```

Como podemos ver el método de split es más amplio, en concreto existe una mayor incertidumbre a la izquierda del valor predicho. Para comparar la eficiencia de los intervalos, comparemos su apertura relativa, respecto al método conformal entero:

```

(resplit$lo-resplit$up) / (res$lo-res$up)

```

```

[,1]
[1,] 1.020071

```

Podemos ver que el método de splitting es más amplio, pero solo en poco más de un 2%.

3.3.3 Ejemplo regresión múltiple

Tomando como ejemplo el dataset `state.x77` de R, vamos a intentar predecir la variable `life_exp` utilizando como variables explicativas `income`, `hs_grad`, `frost` y `area`.

Vamos a entrenar el modelo con todas las observaciones, a excepción de una, que será la que queremos predecir, en concreto, la ciudad de Colorado.

```
# Semilla

set.seed(1234)

# Para arreglar los nombres

normalize_names <- function(names) {
  upper = tolower(names)
  newNames = gsub(" ", "_", upper)
  return(newNames)
}

df <- as.data.frame(state.x77)

names(df) <- normalize_names(names(df))

# Para elegir la observación a predecir

test <- runif(1,1,nrow(df))

# Vector de variables independientes

Y = df[-test,c('life_exp')]

# Matriz de diseño

X = df[-test,c('income','hs_grad','frost','area')]

row.names(X) <- NULL
X = data.matrix(X, rownames.force = TRUE)
```

```

# Matriz de diseño en los nuevos puntos

x0 = data.matrix(
  df[test,c('income','hs_grad','frost','area')],
  rownames.force = TRUE
)

# No funciona con lm.funs, NW usamos RF!

engine_rf = rf.funs(
  ntree = 500,
  varfrac = 0.333
)

train.fun = engine_rf$train.fun
predict.fun = engine_rf$predict.fun

res <- conformal.pred(
  x = X,
  y = Y,
  x0 = x0,
  train.fun = train.fun,
  predict.fun = predict.fun
)

intervalo(
  res$pred,
  res$lo,
  res$up
)

```

[71.01 - 1.32;71.01 + 0.54]

```

glue::glue(
  "La expecativa de vida de la {ciudad} es {df[test,'life_exp']}",
  ciudad = row.names(df[test,])
)

```

La expecativa de vida de la Colorado es 72.06

Para la realización de este ejemplo, se intentó realizar la predicción con el modelo de regresión lineal tanto en su versión paramétrica como su versión de núcleo, el resultado está repleto de valores NA, por lo que se optó por probar diferentes modelos de regresión no paramétrica, en concreto, Random Forest, el cual nos dio un resultado satisfactorio.

4 Conclusión

Si bien en la práctica, los intervalos de predicción pueden llegar a ser confusos para personas no técnicas sobre estadística es importante destacar su utilidad en la toma de decisiones, cualquier metodología que aborde sobre la mejora de estos, es de gran relevancia como la metodología expuesta en este artículo.

Con este artículo hemos revisado como implementar intervalos de predicción conformales para cualquier modelo de regresión, estos algoritmos requieren únicamente asumir que los datos tienen independencia entre sí, pero ninguna forma funcional sobre los regresores, matriz de diseño o distribución de probabilidad para los residuos. Esto último puede llegar a ser una gran ventaja para modelos en donde no es tan claro la construcción de intervalos de predicción que usualmente son aproximados con bootstrap, como pueden ser los métodos de ensamble o en aquellos modelos donde se hacen supuestos fuertes sobre la distribución de los datos basados en resultados asintóticos.

El considerar este tipo de metodologías nos permite reducir la incertidumbre respecto a un intervalo de predicción clásico como pudimos ver en el Sección 2.3.1. Es importante mencionar que se cuenta con una implementación por parte de los autores en la librería **conformalInfer** de R con modelos de regresión ampliamente utilizados en la práctica, como son modelos de regresión lineal, splines, glm, random forest, etc.

Cabe destacar que la inferencia conformal es aplicada frecuentemente en la actualidad a modelos de machine learning. Estas metodologías pueden complementar los métodos basados únicamente en bootstrap para estimación de varianzas en para cualquier algoritmo de regresión, demostrando que en base al estimador consistente, se puede obtener un intervalo de predicción consistente en la estimación de la cobertura condicional

5 Bibliografía

6 TODO AGREGAR BIBLIOGRAFIA y PAQUETES DE R

Lei, J., G'Sell, M., Rinaldo, A., Tibshirani, R. J., & Wasserman, L. (2016). *Distribution-Free Predictive Inference For Regression*. arXiv. <https://doi.org/10.48550/ARXIV.1604.04173>

7 Anexo