Tarea Modelización Predictiva

Luis Miguel Domínguez Pérez

7/2/2021

Parámetros inciales.

Vamos a comenzar el estudio estableciendo el working directory e importando tanto los datos como el código de las funciones que podamos necesitar durante la práctica.

```
setwd("C:/Users/luismiguel.dominguez/Escritorio/Tarea_mod_pred")
source("C:/Users/luismiguel.dominguez/Escritorio/Tarea_mod_pred/Funciones_R.R")
datos <- readRDS('C:/Users/luismiguel.dominguez/Escritorio/Tarea_mod_pred/Rad2018.RDS')
```

Carga de librerías necesarias.

A continuación cargamos las librerías que podamos llegar a necesitar durante el análisis. Las utilizadas han sido: ggpubr, psych, corrplot, mice, questionr, caret, pROC y lmSupport.

Análisis inicial de la estructura de los datos.

Vamos a comprobar que los tipos de las variables son las indicadas, detectando si alguna de ellas debería ser un factor o hay tipos asignados erróneamente.

El pico de radiación debería ser un factor así que los transformamos.

```
datos$PicoRad <- as.factor(datos$PicoRad)
str(datos)
```

Ahora vamos a inspeccionar la variedad de valores en las variables numéricas para ver si tienen una distribución razonable. En caso de que alguna de ellas tenga una distribución que no sea contínua podríamos calificarla como factor.

```
sapply(datos[, 2:12],function(x) length(unique(x)))
```

```
##       TD       Rn Desc.Rn     Pres     Temp       HR       HS   Isolar Vviento Temp.Su
##      617     6617     6644     6533     6567     6039     6768     4003     4506     3516
##   Lluvia
##       70
```

Ninguna de ellas parece adecuada para recategorizarla como un factor. La que menos niveles tendría sería la referente a las precipitaciones y serían 70, que es excesivo para tratarla como un factor.

Realizamos una primera inspección de las variables numéricas para ver si detectamos algún patrón extraño.

Podemos utilizar la función filter para quedarnos con las numéricas y no tener que introducir los índices manualmente como en el caso anterior.

```
summary(Filter(is.numeric, datos))
```

Lo primero que nos llama la atención es el elevado número de registros NA presentes.

TD -> Valores máximos y mínimos razonables, media y mediana muy similares sugiriendo distribución normal.

Rn y Desc.Rn -> Valores mínimos y máximos razonables. Presencia importante de NA's.

Pres y Temp -> Valores razonables pero con presencia de NA's.

HR y HS -> Valores en el dominio [0, 100], que es como debería ser al tratarse de porcentajes. También tienen presente un número importante de valores faltantes.

Isolar -> Presenta valores muy radicales, habrá que inspeccionarla más a fondo para evaluar la presencia o no de outliers. Presencia de NA's.

Vviento -> Parece que hay un valor máximo algo exagerado, habrá que inspeccionarla con más detalle. Presencia de NA's.

Temp.Su -> Los valores parecen dentro de lo normal. Presencia de NA's.

Lluvia -> Seguramente presentará uns distribución algo extrema pero aparte de la presencia de NA's nada más que señalar de momento.

Inspección de las variables categóricas.

Solo tenemos una variables categórica, que además será la objetivo de la regresión logística, por lo que podemos hacer una inspección del reparto de valores de la siguiente manera:

```
summary(Filter(is.factor, datos))
```

```
##  PicoRad
##  0:5767
##  1:2993
```

Hay un buen reparto, la proporción del reparto de valores es de 1/3 vs 2/3 que es bastante razonable y al no presentar valroes nulos no parece que esta variable vaya a necesitar ninún tratamiento especial.

Inspección de la variable Fecha.

Nos queda una variable por inspeccionar, la fecha. Dado su carácter, la inspección más importante que le podemos hacer es ver que no haya repeticiones de registros, es decir, fechas repetidas. Una manera de hacerlo es la siguiente. También podemos ver que no hay datos faltantes de fechas.

```
##                  Min.               1st Qu.                Median
## "2018-01-01 00:00:00" "2018-04-02 05:44:59" "2018-07-02 11:29:59"
##                  Mean               3rd Qu.                  Max.
## "2018-07-02 11:29:59" "2018-10-01 17:14:59" "2018-12-31 22:59:59"
```

```
## [1] "El número de fechas diferentes es: 8760"
```

```
## [1] "El número total de fechas es: 8760"
```

Análisis detallado de variables cuantitativas.

Parece que las únicas variables problemáticas detectadas han sido algunas de las numéricas, por lo que vamos a hacer un análisis más detallado de estas mediante boxplots para intentar detectar outliers. Lo haremos comparando gráficos de densidad con sus respectivos boxplots. Mostramos estos resultados en la Figura 1 del Anexo aunque el código fue el siguiente.

```
par(mfrow=c(3,2))
for(i in (colnames(Filter(is.numeric, datos)))){
  dens <- density(na.omit(datos[[i]]))
  plot(dens, main = paste("Densidad para ", i), col = "steelblue")
    polygon(dens, col = "steelblue")
  box <- boxplot(datos[i], main = paste("Boxplot para ", i), col = "Orange")
}
```

En mi opinión no hay outliers claramente descartables en ninguna de las variables numéricas.

En algunas como la lluvia o el viento lo que parecen outliers siguen estando dentro de valores que podrían esperarse físicamente (tanto los vientos de más de 150 km/h como las lluvias de 8 l/m**2 son fenómenos que pueden producirse de manera natural) por lo que descartarlos me parecería sesgar arbitrariamente los datos.

La variable objetivo TD que hace referencia a la “Tasa de Dosis” en microSieverts presenta valores que pueden considerarse como outliers pero al ser escasos y tratarse de la variable objetivo prefiero dejarlos tal y como están.

En mi opinión, los outliers, salvo que sus valores salgan de dominios de los que no pueden salir físicamente en condiciones normales (por ejemplo un viento de 8000 km/h, un porcentaje negativo o mayor que 100 o una temepratura del suelo de 400 ºC) deben ser conservados y en nuestro caso no hay ninguna variable numérica que presente este tipo de valores extremos claramente descartables.

Separamos las variables objetivo (target) de las predictoras.

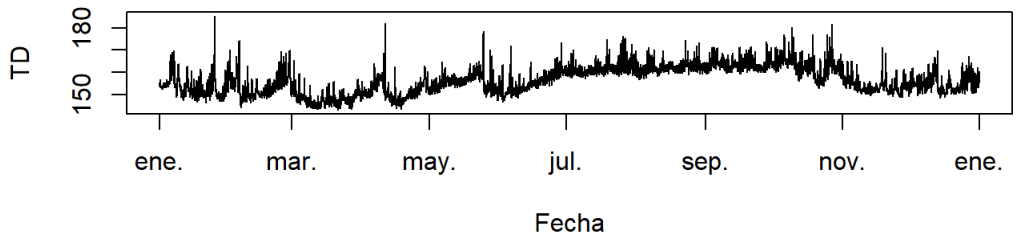
Tenemos dos variables target, “TD” que es cuantitativa e intentaremos predecir mediante una regresión lineal y PicoRad que es una variable dicotómica (factor de dos niveles) cuyo valor aproximaremos mediante una regresión logística.

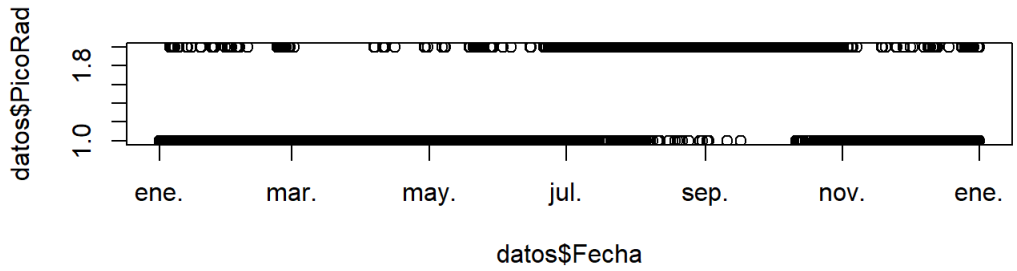
```
targetCont<-datos$TD
targetBin<-datos$PicoRad
input<-as.data.frame(datos[,-c(1,2,13)])
colnames(input)
str(input)
```

Tratamiento de la variable fecha.

Para poder utilizar esta variable vamos a estudiar el comportamineto de las variables a predecir con respecto a la variable fecha. Intentaremos ver si existe algún tipo de patrón entre diversas derivadas de la fecha(como el mes del año, el día del mes o la hora del día) e incluirlas en el modelo como variables predictoras. No estamos demasiado interesados en ver la relación de la fecha con el resto de variables predictoras ya que una relación entre ellas sólo nos indicaría que vamos a introducir colinealidad al modelo y esto lo podemos ver por otros métodos (VIF).

```
par(mfrow=c(2,1))
plot(datos$Fecha, datos$TD, type = "S", xlab = "Fecha", ylab = "TD")
plot(datos$Fecha, datos$PicoRad)
```





En estas dos anteriores gráficas podemos ver el carácter temporal de las dos variables target. En la primera vemos la relación entre la fecha y la variable “TD” y revela que es más probable encontrar mayores valores de “TD” durante los meses más cálidos del año.La segunda nos revela algo bastante parecido para la variable dicotómica, que es más probable que presente un valor de 1 (es el segundo factor, por eso aparece como 2 en el eje y de la gráfica) durante los meses cálidos. Ahora que está claro que existe una dependencia con la fecha podemos profundizar más en ella.

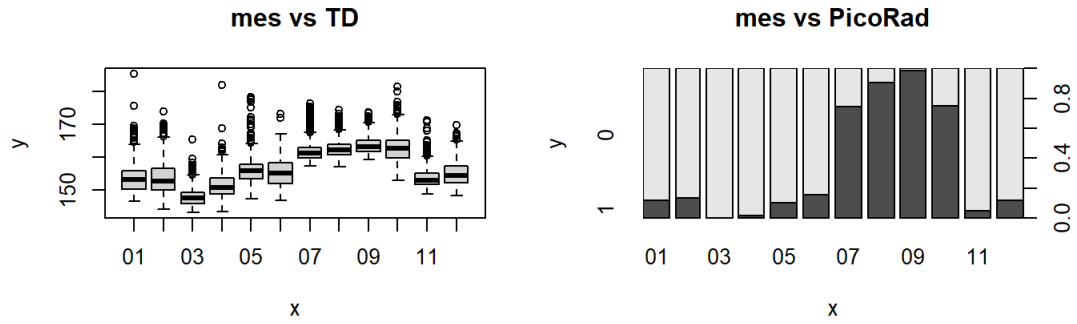
Vamos a trabajar con tres variables derivadas de la fecha, el mes del año, el día del año y la hora del día. Las vamos a tratar como factores ya que de tratarlas como variables cuantitativas estaríamos cometiendo un error peligroso (no es correcto considerar que, por ejemplo, existe una relación más fuerte entre el mes 9 y el 12 que entre el 12 y el 1, y una variable cuantitativa asume una ordinalidad en los valores aparte de una escala que está lejos de ser real).

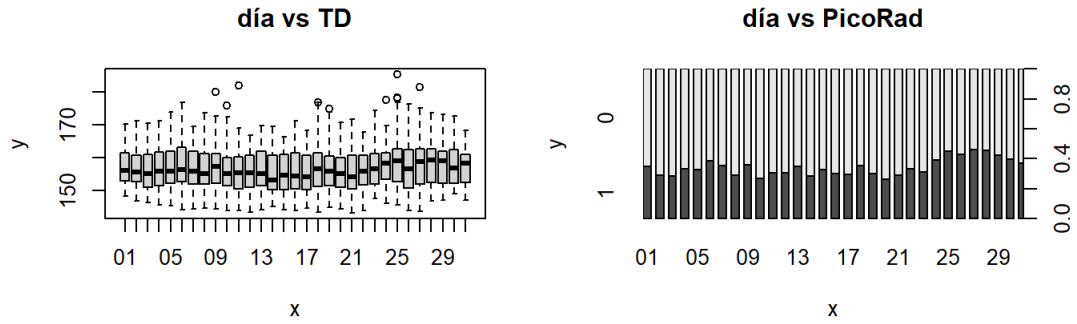
```
input$mes <- as.factor(format(datos$Fecha,"%m"))
input$día <- as.factor(format(datos$Fecha, "%d"))
input$hora <- as.factor(format(datos$Fecha, "%H"))
input$semana <- as.factor(format(datos$Fecha, "%W"))
str(input)
```

```
## 'data.frame':    8760 obs. of  14 variables:
##  $ Rn     : num  9.04 9.66 7.52 7.99 6.8 ...
##  $ Desc.Rn: num  1.42 1.9 2.59 2.99 3.15 ...
##  $ Pres   : num  951 951 951 951 951 ...
##  $ Temp   : num  7 6.77 6.07 5.73 5.77 ...
##  $ HR     : num  60.2 59.2 65.2 64 57.8 ...
##  $ HS     : num  11.3 11.2 11.2 11.2 11.1 ...
##  $ Isolar : num  0 0 0 0 0 0 0 0 0 60 ...
##  $ Vviento: num  26.5 17.6 25 29.8 19.9 ...
##  $ Temp.Su: num  5.4 5.03 4.8 4.57 4.3 ...
##  $ Lluvia : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ mes    : Factor w/ 12 levels "01","02","03",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ día    : Factor w/ 31 levels "01","02","03",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ hora   : Factor w/ 24 levels "00","01","02",..: 1 2 2 3 4 5 6 7 8 9 ...
##  $ semana : Factor w/ 53 levels "01","02","03",..: 1 1 1 1 1 1 1 1 1 1 ...
```

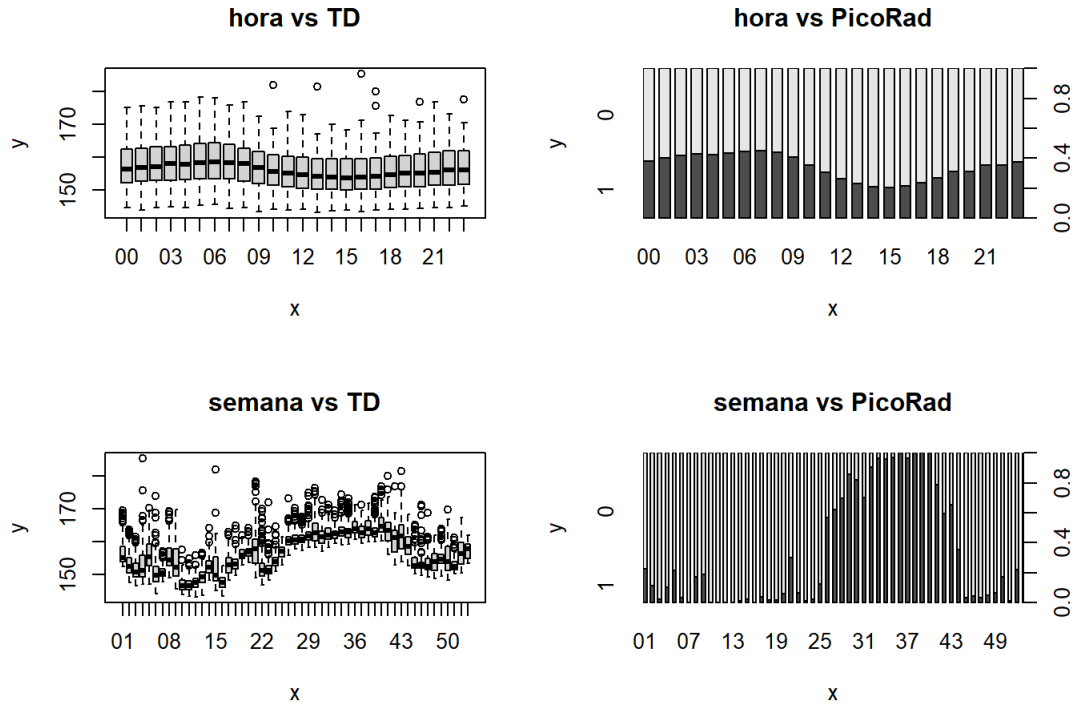
Podemos elaborar unos primeros gráficos para evaluar si hay dependencias entre las variables a predecir y estas predictoras derivadas de la fecha.

```
par(mfrow=c(2,2))
   plot(input$mes, targetCont, main = "mes vs TD")
   plot(input$mes, targetBin, main = "mes vs PicoRad")
   plot(input$día, targetCont, main = "día vs TD")
   plot(input$día, targetBin, main = "día vs PicoRad")
```






```
   plot(input$hora, targetCont, main = "hora vs TD")
   plot(input$hora, targetBin, main = "hora vs PicoRad")
   plot(input$semana, targetCont, main = "semana vs TD")
   plot(input$semana, targetBin, main = "semana vs PicoRad")
```



Relación entre targets y mes del año: Vemos que es probable que existe una dependencia de los dos targets con el mes del año. El target cuantitativo presenta medias claramente más altas (sólo hay que fijarse en la anchura de las cajas y la distancia entre las líneas que delimitan las medias para ver que las distribuciones tienen poca superposición) en los meses más cálidos del año y a su vez parece haber muchísima más probabilidad de pico de radiación durante este mismo periodo.

Relación entre targets y día del mes: Por esta parte no parece haber una relación muy clara, si bien es cierto que parece que hay más probabilidad de pico hacia finales de mes no hay una razón fuerte para sospechar. Es de remarcar que físicamente tiene menos sentido que pueda existir una infuencia del día del mes que de las horas del día o del mes del año ya que estas dos últimas van a tener relación con otras variables físicas del modelo como la temperatura o la insolación mientras que el día del mes en un principio no debería presentarlas. Podemos probar esto más adelante viendo las relaciones mutuas entre todas las variables predictoras.

Relación entre targets y hora del día: Es flagrante que existe algún tipo de relación. Las curvas parecen casi definidas por alguna función matemática ondulatoria y las dos variables target vienen a decirnos lo mismo: Existe más actividad (tanto en niveles como picos de radiación) durante las horas nocturnas que las diurnas.

Relación entre targets y semana: Básicamente el resultado es una versión detallada de los resultados que obtuvimos al tratar los meses.

Valores atípicos.

Ninguna variable tiene un nivel intolerable de valores atípicos.

```
sapply(Filter(is.numeric, input),function(x) atipicosAmissing(x)[[2]])/nrow(input)
```

```
##          Rn     Desc.Rn        Pres        Temp          HR          HS
## 0.012557078 0.016210046 0.002739726 0.000000000 0.000000000 0.000000000
##      Isolar     Vviento     Temp.Su      Lluvia
## 0.000000000 0.002397260 0.000000000 0.061415525
```

A continuación inspeccionamos el número de missings por variable y por registro (fila). En caso de que en alguna de ellas la proporción superase el 0.5 la eliminamos. Junto a cada línea de código se pueden encontrar detalles sobre el funcionamiento.

```
input$prop_missings<-apply(is.na(input),1,mean) #is.na(input) devuelve una máscara con true o
# false en función de si el registro de esa fila es nulo o no. Los nulos son verdadero y compu
#tan como valor 1. Calculando la media de cada secuencia de TRUE o FALSE sabemos la proporción
# de datos faltantes de la fila. El apply se encarga de recorrer cada una de las filas del
# dataFrame.
print("Resumen de NA's por fila:")
```

```
## [1] "Resumen de NA's por fila:"
```

```
summary(input$prop_missings)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.00000 0.00000 0.00000 0.01124 0.00000 0.07143
```

```
print("Resumen de NA's por variable:")
```

```
## [1] "Resumen de NA's por variable:"
```

```
(prop_missingsVars<-apply(is.na(input),2,mean))# Funciona de manera similar a la anterior pero recorre columnas en lugar de filas.
```

```
##            Rn       Desc.Rn          Pres          Temp            HR
##    0.01655251    0.01700913    0.01449772    0.01712329    0.01563927
##            HS        Isolar       Vviento       Temp.Su        Lluvia
##    0.01598174    0.01506849    0.01449772    0.01575342    0.01518265
##           mes           día          hora        semana prop_missings
##    0.00000000    0.00000000    0.00000000    0.00000000    0.00000000
```

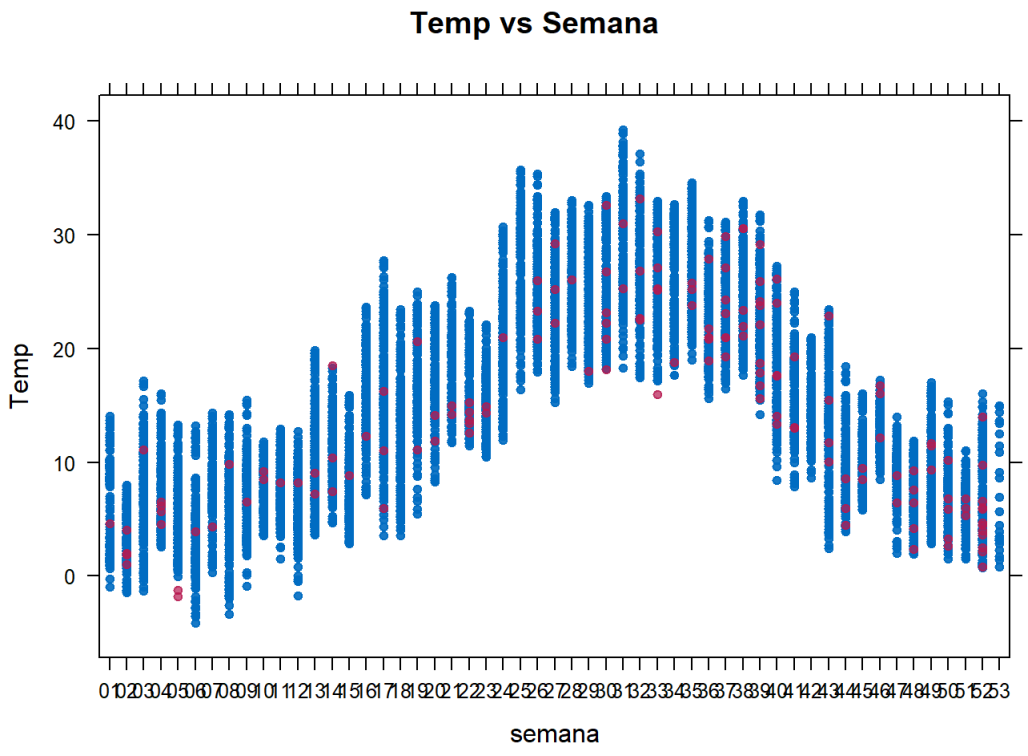
Imputación de variables cuantitativas.

Tenemos que imputar bastantes valores en variables cuantitativas. No tenemos cualitativas que imputar. Podríamos utilizar imputaciones tradicionales como la media, la mediana o valores aleatorios de la variable. Vamos a intentar antes de nada hacer alguna imputación más lógica utilizando las relaciones entre las variables a imputar y las derivadas de la fecha con el paquete mice.

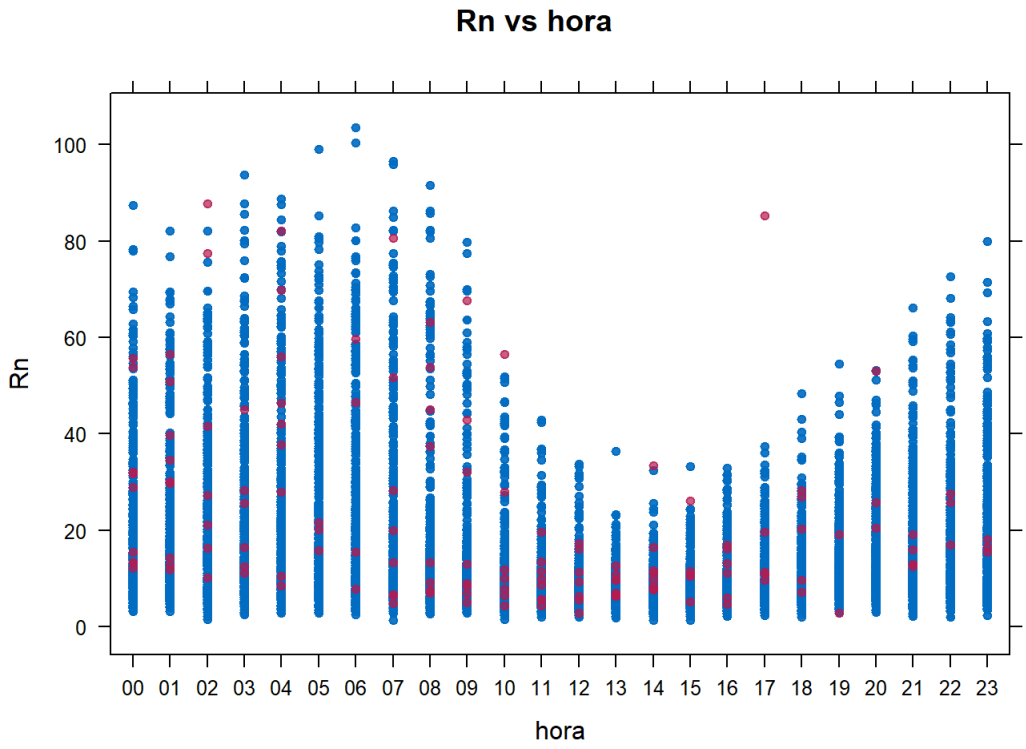
```
imputation <- mice(input, m= 1, maxit = 10)
```

Podemos mostrar la bondad de los resultados de este métod de imputación graficando las variables cuantitativas imputadas frente a algunas de las temporales que mejor capturan su carácter. Podemos generar muchos de estos gráfico, a cada cual más interesante pero lamentablemente, debido a la escasez de espacio que tenemos para mostrar resultados nos limitaremos a dos de ellos para mostrar los resultados aunque el resto se muestran en las Figuras 2 y 3 del Anexo:

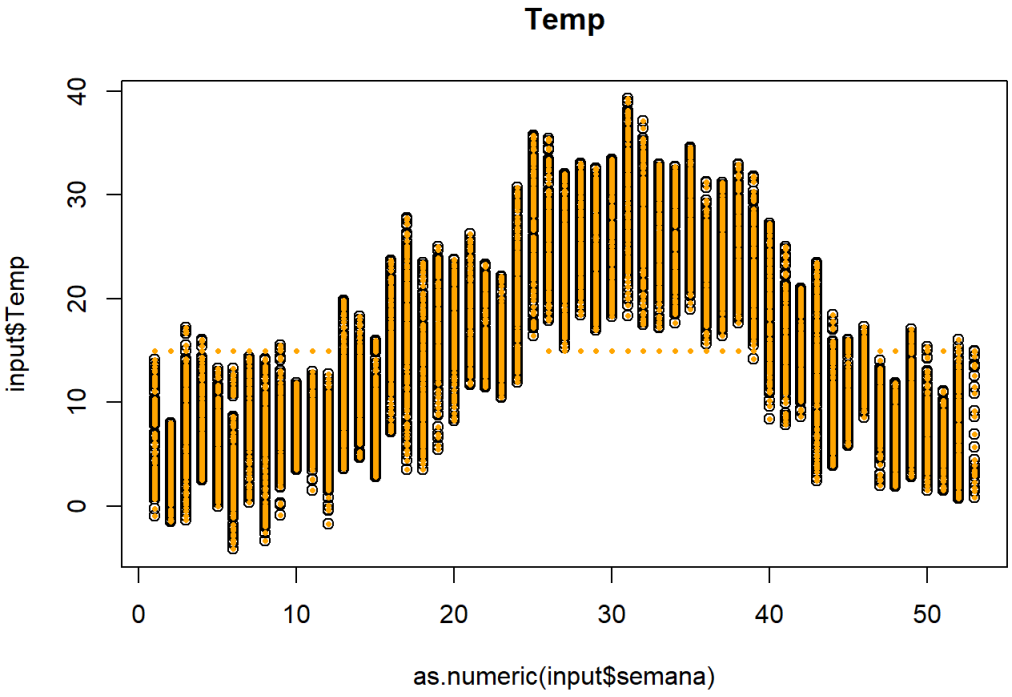
```
xyplot (imputation, Temp~semana, pch = 19, cex = 0.7, main = "Temp vs Semana")
```




```
xyplot (imputation, Rn~hora, pch = 19, cex = 0.7, main = "Rn vs hora")
```



Al estar los resultados de la imputación (mostrados en rojo) obtenidos a través de las relaciones con el resto de las variables conseguimos una adaptación muy exitosa. Si lo hubiéramos hecho imputando con, por ejemplo, la media, hubiéramos obtenido resultados del siguiente aspecto:



Donde la línea horizontal de puntos amarillos sin borde negro representa los datos introducidos al imputar con la media.

Relación entre la variable objetivo cuantitativa y las predictoras numéricas.

Para finalizar la inspección de los datos antes de montar el modelo de predicción de la variable cuantitativa podemos ver la relación entre las variables regresoras y la variable target cuantitativa. Podemos generar gráficos gráficos para hacernos una idea de cuales van a tener más importancia la hora de elaborar los modelos de regresión. Mostramos estos gráficos en la Figura 4 del Anexo, pero el código utilizado fue el siguiente.

```
par(mfrow=c(3,2))
for (i in (colnames(input[,-c(11,12,13,14,15)]))){
  plot( input[[i]],targetCont,
        pch = 16,
        main = paste("TD vs ", i),
        xlab = i,
        ylab = "TD",
        col = "steelblue",
        cex = 0.5)
  abline(lm(TD~input[[i]], data = datos), col = "orange")
}
```

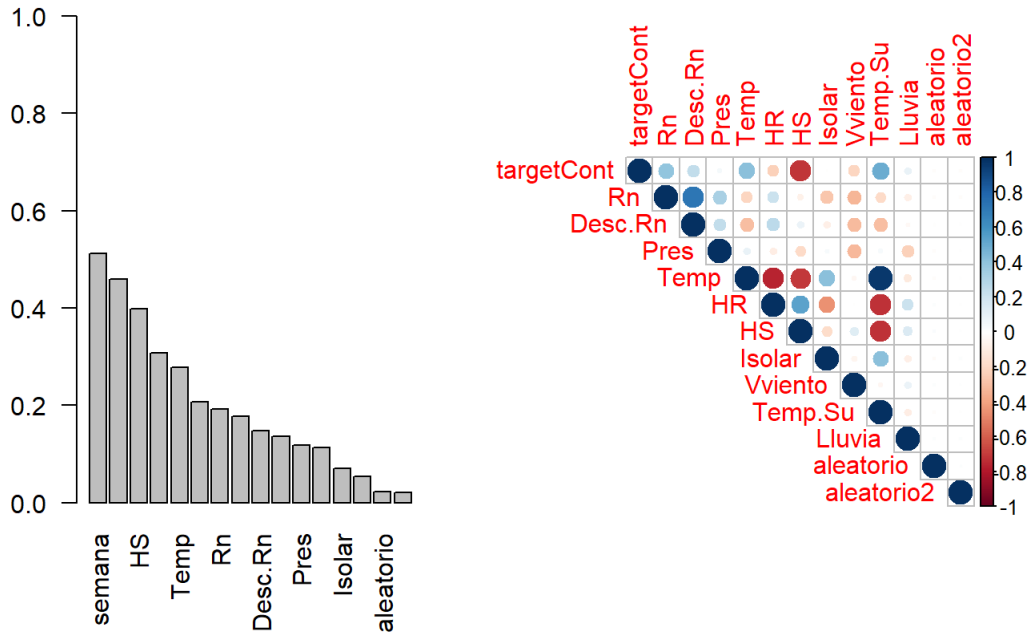
Regresión lineal manual:

Comenzamos insertando dos variables aleatorias en el input que marcarán el punto de corte a partir del cual las variables no son significativas para nuestro modelo.

```
datos<-readRDS("datos_dep")
targetCont<-datos$targetCont
targetBin<-datos$targetBin
input<-datos[,-c(1,2,17)]
input$aleatorio<-runif(nrow(input))
input$aleatorio2<-runif(nrow(input))
summary(input)
```

Inspeccionamos la correlación entre variables y la importancia que tendrán a la hora de realizar un modelo.

```
par(mfrow=c(1,2))
graficoVcramer(input,targetCont)
corrplot(cor(cbind(targetCont,Filter(is.numeric, input)), use="pairwise", method="pearson"), method = "circle",type = "upper")
```

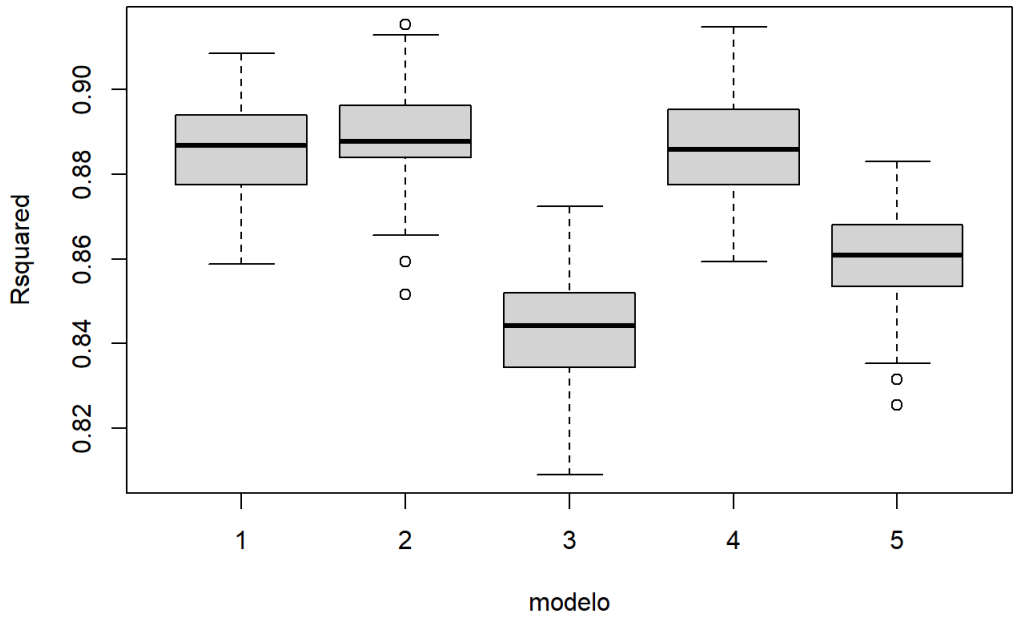


Comenzamos con la regresión lineal sin variables modificadas. PAra ello partimos nuestro dataset en train y test. En input figuran las variables regresoras y targetCont es la variable TD a predecir.

```
todo<-data.frame(input,targetCont)
set.seed(123456)
trainIndex <- createDataPartition(todo$targetCont, p=0.8, list=FALSE)
data_train <- todo[trainIndex,]
data_test <- todo[-trainIndex,]
```

Elaboración del modelo manual.

A continuación vamos a elaborar varios modelos tanteando y jugando con la importancia de las variables, tras ello los compararemos, en cuanto a capacidad de predicción como en complejidad ya que si tienen demasiados parámetros no nos interesarán.




```
##   modelo  Rsquared
## 1      1 0.8859438
## 2      2 0.8885421
## 3      3 0.8435401
## 4      4 0.8859312
## 5      5 0.8597633
```

```
##   modelo   Rsquared
## 1      1 0.01074871
## 2      2 0.01157136
## 3      3 0.01200385
## 4      4 0.01168121
## 5      5 0.01048228
```

```
## [1] 94
```

```
## [1] 94
```

```
## [1] 42
```

```
## [1] 83
```

```
## [1] 20
```

De momento, el mejor modelo manual parece el quinto, con solo 20 parámetros consigue un ajuste bastante bueno, pero vamos a hacerle algunos cambios para remodelar niveles del factor poco significativos y evitar colinealidad.

```
modeloManual<-lm(targetCont~mes+HS+Temp.Su+Rn+Desc.Rn+Vviento+Pres+Lluvia+as.numeric(hora),data=data_train)
```

Si añadiéramos la variable temperatura aparte de la del suelo el VIF de estas se inflaría peligrosamente indicando colinealidad, por eso no está. De la misma manera ocurriría si intentáramos trabajar con la variable semana, que básicamente va a contener una información muy parecida al mes pero aumentando muchísimo el número de parámetros. Lo que sí podemos hacer es juntar algunos de los niveles de la variable “mes”.

```
##
## Call:
## lm(formula = targetCont ~ mes + HS + Temp.Su + Rn + Desc.Rn +
##     Vviento + Pres + Lluvia + as.numeric(hora), data = data_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -16.6836  -1.4000  -0.0614   1.3346  29.3895
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      383.487532   5.312121  72.191  < 2e-16 ***
## mes02             -1.205743   0.152761  -7.893 3.40e-15 ***
## mes03             -4.986263   0.165088 -30.204  < 2e-16 ***
## mes04             -4.169757   0.159040 -26.218  < 2e-16 ***
## mes05             -0.539072   0.169390  -3.182 0.001467 **
## mes06              1.563356   0.191830   8.150 4.29e-16 ***
## mes09              4.759569   0.205692  23.139  < 2e-16 ***
## mes10              3.125674   0.176552  17.704  < 2e-16 ***
## mes11             -1.801787   0.151997 -11.854  < 2e-16 ***
## mes12             -0.821030   0.144982  -5.663 1.55e-08 ***
## HS                -0.317104   0.005562 -57.009  < 2e-16 ***
## Temp.Su           -0.025093   0.007129  -3.520 0.000434 ***
## Rn                 0.110888   0.003211  34.538  < 2e-16 ***
## Desc.Rn            0.047524   0.002690  17.666  < 2e-16 ***
## Vviento           -0.015776   0.003094  -5.099 3.50e-07 ***
## Pres              -0.237687   0.005613 -42.345  < 2e-16 ***
## Lluvia             4.315866   0.114902  37.561  < 2e-16 ***
## as.numeric(hora)  -0.063617   0.004489 -14.172  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.43 on 6992 degrees of freedom
## Multiple R-squared:  0.8456, Adjusted R-squared:  0.8452
## F-statistic:  2252 on 17 and 6992 DF,  p-value: < 2.2e-16
```

```
##                       GVIF Df GVIF^(1/(2*Df))
## mes               13.729718  9        1.156654
## HS                 3.164995  1        1.779043
## Temp.Su            6.679323  1        2.584439
## Rn                 2.547894  1        1.596213
## Desc.Rn            2.381828  1        1.543317
## Vviento            1.286782  1        1.134364
## Pres               1.771225  1        1.330874
## Lluvia             1.080068  1        1.039263
## as.numeric(hora)   1.147956  1        1.071427
```

```
## $r2
## [1] 0.8455636
##
## $r2_adj
## [1] 0.845166
```

```
## $r2
## [1] 0.8138297
##
## $r2_adj
## [1] 0.8118938
```

Ahora el modelo está bastante mejor, consigue unos niveles de predicción bastante aceptables sin incurrir en un exceso de parámetros (que solo dependamos de 17 parámetros está genial) y sin presentar niveles altos de colinealidad. Este será el modelo manual que entrará luego en las comparaciones con los demás. En este primer modelo no se han tenido en cuenta interacciones entre variables ni varaibles transformadas.

Criterios clásicos de elaboración del modelo de regresión lineal.

Vamos a hacer una transformación de nuestro modelo basándonos en resultados previos. Vamos a introducir las variables hora, día y semana como numéricas (el día es poco significativo, la semana como factor introduce demasiados niveles y posiblemente sea descartada

a favor del mes y la hora introduce niveles adicionales y puede que nos funcione como transformada numérica como ya ocurrió en el modelo manual). Extraeremos variables transformadas que linealicen mejor que las originales con la variable target, entre ellas estarán las numéricas de las fechas que acabamos de cambiar de tipo.

```
datos<-readRDS("datos_dep")
targetCont<-datos$targetCont
datos$aleatorio<-runif(nrow(datos))
datos$aleatorio2<-runif(nrow(datos))
datos$día <- as.numeric(datos$día)
datos$hora <- as.numeric(datos$hora)
datos$semana <- as.numeric(datos$semana)
datos_mod<-cbind(datos,Transf_Auto(Filter(is.numeric, datos),targetCont))
datos_mod <- datos_mod[,-c(1,17,18,19,20,34,35,36)]
set.seed(123456)
trainIndex <- createDataPartition(datos_mod$targetCont, p=0.8, list=FALSE)
data_train <- datos_mod[trainIndex,]
data_test <- datos_mod[-trainIndex,]
```

Las variables con las que trabajaremos ahora serán:

```
##  [1] "targetCont"   "Rn"           "Desc.Rn"      "Pres"         "Temp"
##  [6] "HR"           "HS"           "Isolar"       "Vviento"      "Temp.Su"
## [11] "Lluvia"       "mes"          "día"          "hora"         "semana"
## [16] "raiz4Rn"      "logxDesc.Rn"  "expxPres"     "sqrxTemp"     "raiz4HR"
## [21] "raiz4HS"      "logxIsolar"   "sqrtxVviento" "xTemp.Su"     "xLluvia"
## [26] "cuartaxdía"   "xhora"        "sqrtxsemana"
```

Vamos a correr nuestros algoritmos con los criterior AIC y BIC de selección de variables:

```
null<-lm(targetCont~1, data=data_train)#Modelo minimo
full<-lm(targetCont~., data=data_train) # Tenemos en cuenta las variables modificadas.

modeloStepAIC<-step(null, scope=list(lower=null, upper=full), direction="both")
summary(modeloStepAIC)
Rsq(modeloStepAIC,"targetCont",data_test)
modeloStepAIC$rank

modeloStepBIC<-step(null, scope=list(lower=null, upper=full), direction="both",k=log(nrow(data_train)))
summary(modeloStepBIC)
Rsq(modeloStepBIC,"targetCont",data_test)
modeloStepBIC$rank
```

Como la pestaña de resultados de este proceso es infame los resumo aquí:

El criterio AIC elaboró el siguiente modelo:

targetCont ~ mes + Rn + Pres + raiz4HS + Lluvia + Temp + Desc.Rn + cuartaxdía + sqrtxVviento + HS + hora + expxPres + Temp.Su + raiz4Rn + Vviento + raiz4HR + logxDesc.Rn + HR + día + sqrtxsemana + sqrxTemp + Isolar

Con R.squared = 0.8624 y ajustado = 0.8598. Presenta 33 parámetros.

El criterio BIC elaboró el siguiente modelo:

targetCont ~ mes + Rn + Pres + raiz4HS + Lluvia + Temp + Desc.Rn + cuartaxdía + sqrtxVviento + HS + hora + expxPres + Temp.Su + raiz4Rn + Vviento + raiz4HR + logxDesc.Rn

Con R.squared = 0.8614 y ajustado = 0.8591. Presenta 28 parámetros.

Consiguen un buen ajuste pero el número de parámetros es bastante elevado, aparte habría que hacer una limpieza de variable consultando el valor del VIF para cada variable de cada modelo y así salvar problemas de colinealidad.

Se hizo también un análisis de interacciones:

Los resultados obtenidos fueron:

El criterio AIC elaboró un modelo en el que ninguna interacción tomaba importancia salvo algunas con el mes. Como nuestra cantidad de parámetros es grande no vamos a incluirlas.

Con R.squared = 0.9061 y ajustado = 0.8984. Presenta 132 parámetros.

El criterio BIC elaboró un modelo en el que ninguna interacción tomaba importancia salvo algunas con el mes. Como nuestra cantidad de parámetros es grande no vamos a incluirlas.

Con R.squared = 0.8998 y ajustado = 0.8930. Presenta 111 parámetros.

Son demasiados parámetros los que se introducen y las interacciones solo se producían con los diferentes niveles del mes por lo que no serán tenidos en cuenta estos modelos.

Modelos de selección aleatoria.

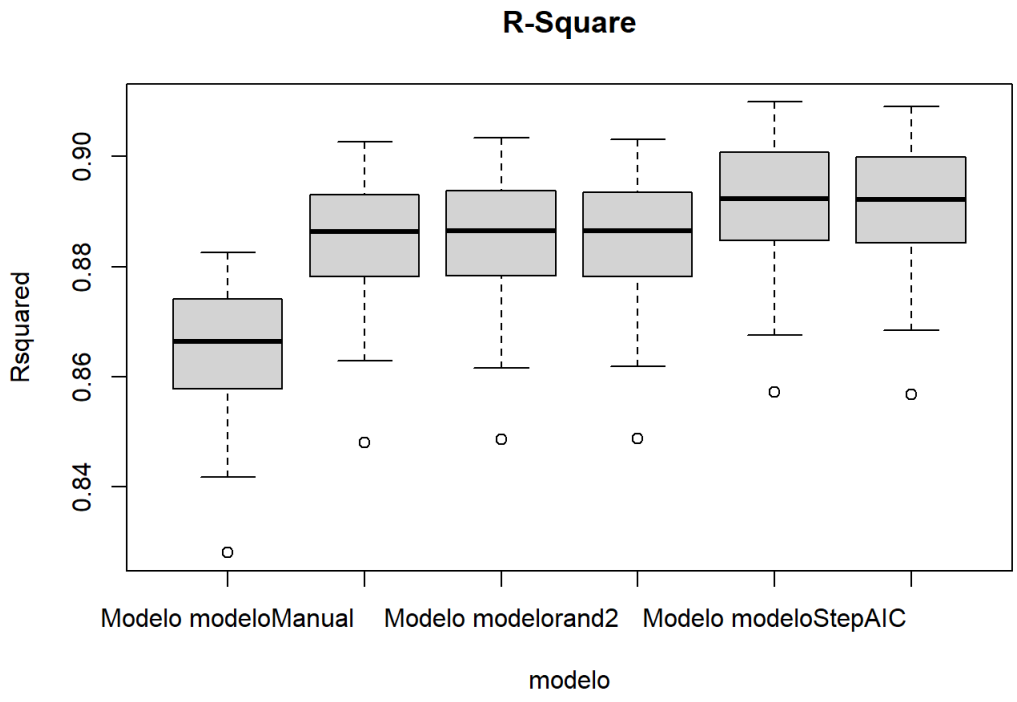
Ahora podemos aplicar un proceso de selección aleatoria de modelos en los que se va cambiando la semilla y se generan secuencialmente modelos a partir de submuestras aleatorias dependientes de la semilla de nuestros datos. Generamos 100 modelos y los ordenamos por frecuencia de aparición (en cuanto a las variables que contienen). No se tienen en cuenta interacciones. Nos quedaremos con los 3 modelos más repetidos a lo largo del proceso.

Los modelos elegidos de la selección aleatoria han sido, después de limpiarlos siguiendo motivos de eliminación de variables por alta colinealidad:

```
modelorand1 <- lm(targetCont~cuartaxdía+Desc.Rn+expxPres+hora+Lluvia+mes+Pres+raiz4HR+raiz4HS+raiz4Rn+Rn+sqr
txVviento+Temp.Su+Vviento, data = data_train)

modelorand2 <- lm(targetCont~cuartaxdía+Desc.Rn+día+expxPres+hora+Lluvia+logxDesc.Rn+mes+Pres+raiz4HR+raiz4H
S+raiz4Rn+Rn+sqrtxsemana+sqrtxVviento+Temp.Su+Vviento, data = data_train)

modelorand3 <- lm(targetCont~cuartaxdía+Desc.Rn+día+expxPres+hora+Lluvia+mes+Pres+raiz4HR+raiz4HS+raiz4Rn+Rn
+sqrtxsemana+sqrtxVviento+Temp.Su+Vviento, data = data_train)
```

Ahora comparamos nuestros 6 modelos, el manual, AIC, BIC y los 3 mejores aleatorios:

```
total<-c()
lista <- c("modeloManual", "modelorand1", "modelorand2", "modelorand3", "modeloStepAIC", "modeloStepBIC")
modelos<-sapply(list(modeloManual, modelorand1, modelorand2, modelorand3, modeloStepAIC, modeloStepBIC),form
ula)
for (i in 1:length(modelos)){
  set.seed(1712)
  vcr<-train(as.formula(modelos[[i]]), data = data_train,
             method = "lm",
             trControl = trainControl(method="repeatedcv", number=5, repeats=20,
                                      returnResamp="all")
  )
  total<-rbind(total,cbind(vcr$resample[,1:2],modelo=rep(paste("Modelo",lista[i]),
                                                         nrow(vcr$resample))))
}
boxplot(Rsquared~modelo,data=total,main="R-Square")
```




```
aggregate(Rsquared~modelo, data = total, mean) #el 5 y el 8 son mejores
```

```
##                 modelo  Rsquared
## 1  Modelo modeloManual 0.8653394
## 2   Modelo modelorand1 0.8851300
## 3   Modelo modelorand2 0.8855022
## 4   Modelo modelorand3 0.8853496
## 5 Modelo modeloStepAIC 0.8918841
## 6 Modelo modeloStepBIC 0.8914405
```

```
aggregate(Rsquared~modelo, data = total, sd)
```

```
##                 modelo    Rsquared
## 1  Modelo modeloManual 0.010688022
## 2   Modelo modelorand1 0.010169146
## 3   Modelo modelorand2 0.010233858
## 4   Modelo modelorand3 0.010194573
## 5 Modelo modeloStepAIC 0.010021222
## 6 Modelo modeloStepBIC 0.009981184
```

```
length(coef(modeloManual)); length(coef(modelorand1));length(coef(modelorand2));length(coef(modelorand3));le
ngth(coef(modeloStepAIC));length(coef(modeloStepBIC))
```

```
## [1] 18
```

```
## [1] 25
```

```
## [1] 28
```

```
## [1] 27
```

```
## [1] 33
```

```
## [1] 28
```

Selección del mejor modelo.

Ya que lo que buscamos es un modelo que pueda ser interpretado y por lo tanto deba tener un número aceptable de parámetros (y no incurrir en un gran VIF que nos deforme los coeficientes) mi elección va a ser el modelo aleatorio 1, ya que consigue un ajuste medio de 0.8851 que es bastante elevado y contiene 25 parámetros que es un número aceptable teniendo en cuenta que tenemos una variable (mes) factor con 12 niveles. A continuación vamos a comprobar que no tenemos grandes valores para los VIF (aunque ya lo hicimos antes para este modelo) y veremos si hay algunos niveles del factor que podamos simplificar por baja relevancia para el modelo.

```
modelorand1 <- lm(targetCont~cuartaxdía+Desc.Rn+expxPres+hora+Lluvia+mes+Pres+raiz4HR+raiz4HS+raiz4Rn+Rn+sqr
txVviento+Temp.Su+Vviento, data = data_train)
summary(modelorand1)
car::vif(modelorand1)
```

```
modelorand1 <- lm(targetCont~cuartaxdía+Desc.Rn+hora+Lluvia+mes+Pres+raiz4HS+Rn+sqrtxVviento+Temp.Su, data =
data_train)
summary(modelorand1)
```

```
##
## Call:
## lm(formula = targetCont ~ cuartaxdía + Desc.Rn + hora + Lluvia +
##     mes + Pres + raiz4HS + Rn + sqrtxVviento + Temp.Su, data = data_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.3547  -1.1716  -0.1092   1.0013  29.8222
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.716e+02  4.444e+00  83.629  < 2e-16 ***
## cuartaxdía    8.444e-03  7.183e-04  11.755  < 2e-16 ***
## Desc.Rn       4.758e-02  2.365e-03  20.120  < 2e-16 ***
## hora         -4.501e-02  3.934e-03 -11.441  < 2e-16 ***
## Lluvia        4.248e+00  1.005e-01  42.259  < 2e-16 ***
## mes02        -8.699e-01  1.333e-01  -6.524 7.33e-11 ***
## mes03        -4.797e+00  1.417e-01 -33.852  < 2e-16 ***
## mes04        -3.210e+00  1.379e-01 -23.273  < 2e-16 ***
## mes06         9.641e-01  1.430e-01   6.744 1.67e-11 ***
## mes07         3.776e+00  2.052e-01  18.400  < 2e-16 ***
## mes08         4.582e+00  2.095e-01  21.874  < 2e-16 ***
## mes09         5.394e+00  1.936e-01  27.856  < 2e-16 ***
## mes10         3.339e+00  1.611e-01  20.726  < 2e-16 ***
## mes11        -5.994e-01  1.105e-01  -5.422 6.08e-08 ***
## Pres         -2.141e-01  4.670e-03 -45.839  < 2e-16 ***
## raiz4HS      -1.456e+01  2.267e-01 -64.243  < 2e-16 ***
## Rn            9.160e-02  2.861e-03  32.011  < 2e-16 ***
## sqrtxVviento -8.920e-01  8.514e-02 -10.477  < 2e-16 ***
## Temp.Su      -1.341e-01  6.518e-03 -20.572  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.129 on 6991 degrees of freedom
## Multiple R-squared:  0.8814, Adjusted R-squared:  0.8811
## F-statistic:  2887 on 18 and 6991 DF,  p-value: < 2.2e-16
```

```
car::vif(modelorand1)
```

```
##                   GVIF Df GVIF^(1/(2*Df))
## cuartaxdía    1.034727  1        1.017216
## Desc.Rn       2.396529  1        1.548073
## hora          1.147999  1        1.071447
## Lluvia        1.076252  1        1.037425
## mes          15.167159  9        1.163070
## Pres          1.596428  1        1.263498
## raiz4HS       5.399422  1        2.323666
## Rn            2.635140  1        1.623311
## sqrtxVviento  1.468069  1        1.211639
## Temp.Su       7.270828  1        2.696447
```

Este es mi modelo final para la regresión lineal. Hemos unido algunos niveles de la variable mes que presentaban bajos niveles de importancia para el modelo. También hemos eliminado algunas variables que presentaban algo de colinealidad, como por ejemplo exp.Pres y Pres. Para hacerlo se ha seguido el procedimiento de evaluar el modelo en la ausencia de cada una de ellas alternamente y quedarnos con el modelo que mejores resultados ofrezca (prescindir de la que menos aporte aparentemente al modelo).

La interpretación de nuestros coeficientes sería:

- Un aumento de 1 en la variable sqrtxVviento produce una reducción de 0.892 en TD.
- Un aumento de 1ºC en Temp.Su produce una reducción de 0.134 en TD.
- Un aumento de 1bq/m^3 en Rn produce un aumento de 0.091 en TD
- Un aumento de 1 en la variable HR produce una reducción de 14.56 en TD.
- Un aumento de 1mB en Pres produce un descenso de 0.214 en TD.
- Un aumento de 1l/m^2 en Lluvia produce un aumento de 4.248 en TD.
- Un aumento de una hora medido desde la hora incial del día produce un descenso de 0.045 en TD.
- Un aumento de 1bq/m^3 en Desc.Rn produce un aumento de 0.047 en TD.

- Un aumento de 1 en el día desde el primer día del mes produce un aumento de 0.0084 en TD.

  Esta última variable aunque parece infuir en los modelos me temo que sólo se debe a la muestra ya que no soy capaz de darle significado físico.

Regresión logística.

Vamos a agilizar un poco el proceso por que disponemos de poco espacio para presentar los resultados. El modelo manual se ha hecho siguiendo procedimientos similares al del modelo lineal, se compusieron varios modelos y se compararon, el mejor modelo fue:

```
datos$mes<-car::recode(datos$mes, "'02'='03';'05'='06'")
data_train <- datos[trainIndex,]
data_test <- datos[-trainIndex,]
modeloManual_bin<-glm(targetBin~mes+HS+HR+Rn+Desc.Rn+as.numeric(hora)+Vviento+Pres+Lluvia,data=data_train, f
amily = binomial)
summary(modeloManual_bin)
```

```
##
## Call:
## glm(formula = targetBin ~ mes + HS + HR + Rn + Desc.Rn + as.numeric(hora) +
##     Vviento + Pres + Lluvia, family = binomial, data = data_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.4227  -0.2261  -0.0622   0.1766   3.3585
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)      175.785827  10.953517  16.048  < 2e-16 ***
## mes03             -1.740025   0.291911  -5.961 2.51e-09 ***
## mes04             -3.862557   0.431407  -8.953  < 2e-16 ***
## mes06             -0.997449   0.256983  -3.881 0.000104 ***
## mes07              1.946873   0.301616   6.455 1.08e-10 ***
## mes08              3.301113   0.318106  10.377  < 2e-16 ***
## mes09              5.679789   0.470734  12.066  < 2e-16 ***
## mes10              1.352124   0.278658   4.852 1.22e-06 ***
## mes11             -3.552285   0.361898  -9.816  < 2e-16 ***
## mes12             -1.529648   0.265532  -5.761 8.38e-09 ***
## HS                -0.232772   0.014238 -16.348  < 2e-16 ***
## HR                 0.025061   0.003147   7.963 1.68e-15 ***
## Rn                 0.066854   0.005097  13.116  < 2e-16 ***
## Desc.Rn            0.024716   0.004282   5.772 7.83e-09 ***
## as.numeric(hora)  -0.049957   0.008008  -6.238 4.42e-10 ***
## Vviento           -0.029583   0.008039  -3.680 0.000233 ***
## Pres              -0.187254   0.011577 -16.175  < 2e-16 ***
## Lluvia             2.018411   0.180271  11.197  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 9001.8  on 7008  degrees of freedom
## Residual deviance: 2881.6  on 6991  degrees of freedom
## AIC: 2917.6
##
## Number of Fisher Scoring iterations: 7
```

```
pseudoR2(modeloManual_bin,data_train,"targetBin")
```

```
## [1] 0.6798841
```

```
pseudoR2(modeloManual_bin,data_test,"targetBin")
```

```
## [1] 0.6945839
```

```
modeloManual_bin$rank #nÃºmero de parÃ¡metros
```

```
## [1] 18
```

```
car::vif(modeloManual_bin)
```

```
##                       GVIF Df GVIF^(1/(2*Df))
## mes                11.771045  9        1.146805
## HS                  4.323766  1        2.079367
## HR                  3.054639  1        1.747753
## Rn                  3.400680  1        1.844093
## Desc.Rn             3.009937  1        1.734917
## as.numeric(hora)    1.307817  1        1.143598
## Vviento             1.416170  1        1.190029
## Pres                2.419172  1        1.555369
## Lluvia              1.577552  1        1.256007
```

Hemos juntado algunos de los niveles del factor mes, como curiosidad, tenemos un pseudo R^2 mayor en el conjunto test que en el propio train. El modelo presente 18 parámetros que no está nada mal.

A continuación vamos a mejorar un poco los datos de entrada, elaboramos variables transformadas para ajustar de mejor manera con el target y transformamos definitivamente la hora, la semana y el día a numérico para que puedan ser transformadas de la misma manera de ser necesario. Las variables que han resultado:

```
##  [1] "targetBin"    "Rn"           "Desc.Rn"      "Pres"         "Temp"
##  [6] "HR"           "HS"           "Isolar"       "Vviento"      "Temp.Su"
## [11] "Lluvia"       "mes"          "día"          "hora"         "semana"
## [16] "raiz4Rn"      "raiz4Desc.Rn" "sqrxPres"     "sqrxTemp"     "sqrxHR"
## [21] "raiz4HS"      "xIsolar"      "logxVviento"  "sqrxTemp.Su"  "sqrtxLluvia"
## [26] "expxdía"      "sqrxhora"     "sqrxsemana"
```

Selección de modelos con métodos clásicos:

Al igual que en el apartado anterior vamos a utilizar los criterios AIC y BIC para elaborar modelos progresivos.

```
null<-glm(targetBin~1, data=data_train, family = binomial)
full<-glm(targetBin~., data=data_train, family = binomial)

modeloStepAIC_bin<-step(null, scope=list(lower=null, upper=full), direction="both")
summary(modeloStepAIC_bin)
pseudoR2(modeloStepAIC_bin,data_train,"targetBin")
pseudoR2(modeloStepAIC_bin,data_test,"targetBin")
modeloStepAIC_bin$rank

modeloStepBIC_bin<-step(null, scope=list(lower=null, upper=full), direction="both",k=log(nrow(data_train)))
summary(modeloStepBIC_bin)
pseudoR2(modeloStepBIC_bin,data_train,"targetBin")
pseudoR2(modeloStepBIC_bin,data_test,"targetBin")
modeloStepBIC_bin$rank
```

Los modelos calculados por los algoritmos son:

AIC: targetBin ~ mes + raiz4Rn + sqrtxLluvia + raiz4HS + Temp + sqrxPres + raiz4Desc.Rn + sqrxTemp.Su + Temp.Su + HS + semana + sqrxhora + sqrxTemp + Rn + Pres + sqrxHR + Lluvia + sqrxsemana + xIsolar

Con Pseudo R^2 training 0.7663 y test 0.7686 y con 30 parámetros.

BIC: targetBin ~ mes + raiz4Rn + sqrtxLluvia + raiz4HS + Temp + sqrxPres + raiz4Desc.Rn + sqrxTemp.Su + Temp.Su + HS + semana + sqrxhora + sqrxTemp + Rn + Pres + sqrxHR + Lluvia + sqrxsemana + xIsolar

Con Pseudo R^2 training 0.7644 y test 0.7693 y con 25 parámetros.

Los modelos con interacciones son descartados por la misma razón que en el apartado anterior, son mayormente con los factores del mes y los parámetros se diaparan hasta más de la centena.

Evaluación de los modelos aleatorios.

Lo hacemos de la siguiente manera, hemos puesto sólo 30 repeticiones por que el tiempo de procesamiento de cada uno de estos modelos es muy superior al de los modelos lineales. Seleccionamos los tres que más se han repetido.

```
rep<-30
prop<-0.7
modelosGenerados<-c()
for (i in 1:rep){
  set.seed(12345+i)
  subsample<-data_train[sample(1:nrow(data_train),prop*nrow(data_train),replace = T),]
  full<-glm(targetBin~.,data=subsample, family = binomial)
  null<-glm(targetBin~1,data=subsample, family = binomial)
  modeloAux<-step(null,scope=list(lower=null,upper=full),direction="both",trace=0,k=log(nrow(subsample)))
  modelosGenerados<-c(modelosGenerados,paste(sort(unlist(strsplit(as.character(formula(modeloAux))[3]," [+]
")))),collapse = "+"))
}
freq(modelosGenerados,sort="dec")
```

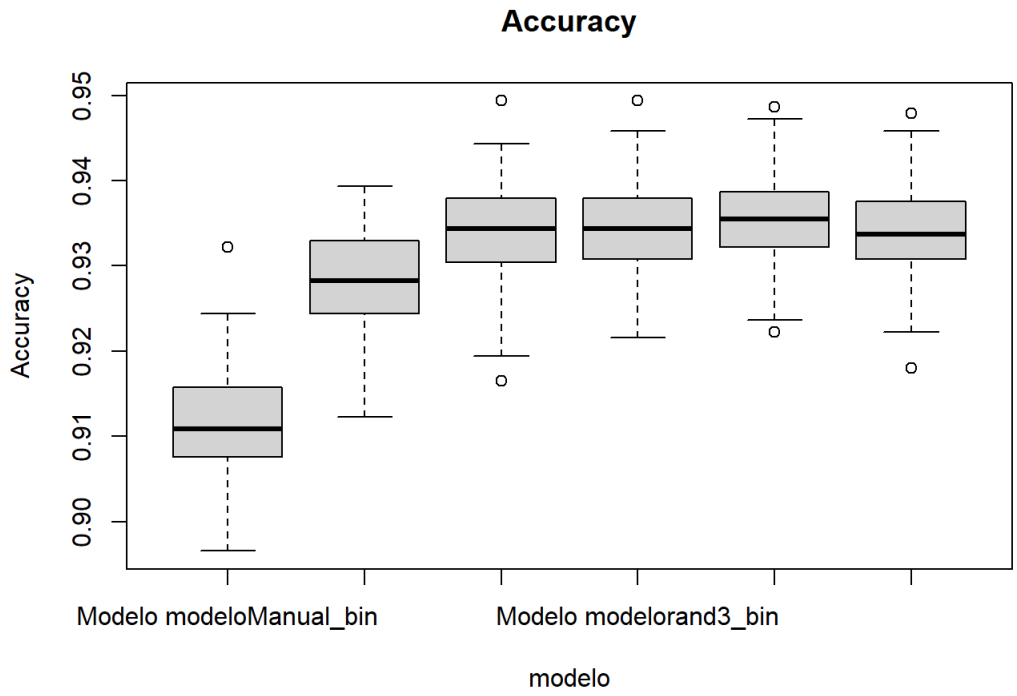
Los más exitosos fueron:

```
modelorand1_bin = glm(targetBin~HS+mes+raiz4Desc.Rn+raiz4HS+raiz4Rn+sqrtxLluvia+sqrxhora+sqrxPres+sqrxTemp,
data = data_train, family = binomial)

modelorand2_bin = glm(targetBin~expxdía+HS+mes+raiz4Desc.Rn+raiz4HS+raiz4Rn+sqrtxLluvia+sqrxhora+sqrxPres+sq
rxTemp+sqrxTemp.Su+Temp+Temp.Su, data = data_train, family = binomial)

modelorand3_bin = glm(targetBin~HS+mes+raiz4Desc.Rn+raiz4HS+Rn+semana+sqrtxLluvia+sqrxhora+sqrxHR+sqrxPres+s
qrxTemp+sqrxTemp.Su+Temp+Temp.Su , data = data_train, family = binomial)
```

Ponemos nuestros mejores modelos a prueba con validación cruzada.

```
total<-c()
lista <- c("modeloManual_bin", "modelorand1_bin", "modelorand2_bin", "modelorand3_bin", "modeloStepAIC_bin",
"modeloStepBIC_bin")
modelos<-sapply(list(modeloManual_bin, modelorand1_bin, modelorand2_bin, modelorand3_bin, modeloStepAIC_bin,
modeloStepBIC_bin),formula)
for (i in 1:length(modelos)){
  set.seed(1712)
  vcr<-train(as.formula(modelos[[i]]), data = data_train,
             method = "glm",
             trControl = trainControl(method="repeatedcv", number=5, repeats=20,
                                      returnResamp="all")
  )
  total<-rbind(total,cbind(vcr$resample[,1:2],modelo=rep(paste("Modelo",lista[i]),
                                                         nrow(vcr$resample))))
}
```

```
boxplot(Accuracy~modelo, data = total, main = "Accuracy")
```




```
aggregate(Accuracy~modelo, data = total, mean) #el 5 y el 8 son mejores
```

```
##                     modelo  Accuracy
## 1  Modelo modeloManual_bin 0.9114427
## 2   Modelo modelorand1_bin 0.9283851
## 3   Modelo modelorand2_bin 0.9338638
## 4   Modelo modelorand3_bin 0.9342061
## 5 Modelo modeloStepAIC_bin 0.9349766
## 6 Modelo modeloStepBIC_bin 0.9338565
```

```
aggregate(Accuracy~modelo, data = total, sd)
```

```
##                     modelo    Accuracy
## 1  Modelo modeloManual_bin 0.006494595
## 2   Modelo modelorand1_bin 0.005502527
## 3   Modelo modelorand2_bin 0.005722624
## 4   Modelo modelorand3_bin 0.005401060
## 5 Modelo modeloStepAIC_bin 0.005685248
## 6 Modelo modeloStepBIC_bin 0.005579204
```

```
length(coef(modeloManual_bin));length(coef(modelorand1_bin));length(coef(modelorand2_bin));length(coef(model
orand3_bin));length(coef(modeloStepAIC_bin));length(coef(modeloStepBIC_bin))
```

```
## [1] 18
```

```
## [1] 20
```

```
## [1] 24
```

```
## [1] 25
```

```
## [1] 30
```

```
## [1] 25
```

Mi modelo definitivo elegido proviene de pulir el modelo ModeloStepAIC_bin, es el que mejores predicciones presenta pero tiene muchos parámetros, pero vamos a ver si eliminando variables por motivos de colinealidad y fucionando niveles de factores no significativos

podemos reducirlo sustancialmente.

```
##
## Call:
## glm(formula = targetBin ~ mes + sqrtxLluvia + raiz4HS + sqrxPres +
##     raiz4Desc.Rn + sqrxTemp.Su + sqrxhora + Rn, family = binomial,
##     data = data_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.2685  -0.1904  -0.0438   0.1086   3.8053
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)   10.951840   0.757952  14.449  < 2e-16 ***
## mes03         -2.354132   0.298494  -7.887 3.10e-15 ***
## mes04         -3.975893   0.445648  -8.922  < 2e-16 ***
## mes05         -0.919943   0.257543  -3.572 0.000354 ***
## mes07          1.857136   0.349803   5.309 1.10e-07 ***
## mes08          3.670847   0.390901   9.391  < 2e-16 ***
## mes09          5.623532   0.501814  11.206  < 2e-16 ***
## mes10          0.477571   0.292343   1.634 0.102343
## mes11         -3.855305   0.376241 -10.247  < 2e-16 ***
## mes12         -1.063618   0.269830  -3.942 8.09e-05 ***
## sqrtxLluvia    2.171129   0.131341  16.530  < 2e-16 ***
## raiz4HS      -12.487156   0.605985 -20.606  < 2e-16 ***
## sqrxPres      -0.150043   0.009692 -15.481  < 2e-16 ***
## raiz4Desc.Rn   3.033469   0.370358   8.191 2.60e-16 ***
## sqrxTemp.Su   -0.333426   0.022403 -14.883  < 2e-16 ***
## sqrxhora      -0.101333   0.017326  -5.849 4.96e-09 ***
## Rn             0.075797   0.005305  14.287  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 9001.8  on 7008  degrees of freedom
## Residual deviance: 2464.5  on 6992  degrees of freedom
## AIC: 2498.5
##
## Number of Fisher Scoring iterations: 7
```

```
## [1] 0.7262203
```

```
## [1] 0.748485
```

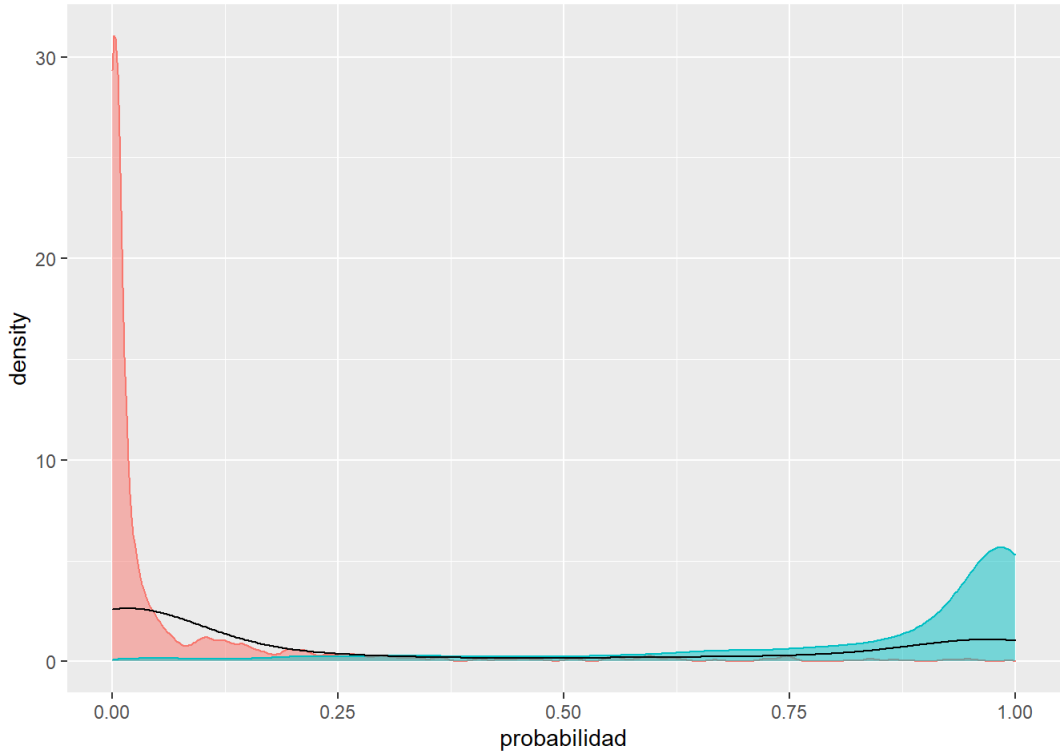
```
## [1] 17
```

```
##                   GVIF Df GVIF^(1/(2*Df))
## mes          21.785722  9        1.186705
## sqrtxLluvia   1.963830  1        1.401367
## raiz4HS       6.681115  1        2.584785
## sqrxPres      2.503371  1        1.582204
## raiz4Desc.Rn  3.054947  1        1.747841
## sqrxTemp.Su   5.383800  1        2.320302
## sqrxhora      1.186987  1        1.089489
## Rn            3.554161  1        1.885248
```

Este sería nuestro modelo definitivo, donde hemos juntado algunos niveles del factor. Puede que el nivel mes10 no sea significativo pero fusionarlo con otros niveles nos quita bastante ajuste (los R^2 ajustados bajan hasta 0.69-0.70).

Evaluación del punto de corte.

Una vez tenemos nuestro modelo elegido podemos ver cómo se comporta, recordemos que el modelo logístico dará de salida un valor entre 0 y 1 pero el resultado que queremos realmente es un 0 o un 1, por lo que podemos ajustar el valor a partir del cual colapsa a 1 o si no llega a ese valor, a 0. Podemos ver que hay una distinción muy calra de los dos conjuntos sobre nuestro eje de regresión.



Los gráficos de rejilla de Youden y de especificidad que ayudan a elegir el valor de corte de manera óptima se pueden encontrar en la Figura 5 del Anexo. A continuación se van los valores del punto de corte para los que se alcanza el máximo de estos dos gráficos y por lo tanto el máximo en la sensibilidad (capacidad de discernir verdaderos positivos) y la precisión del modelo logístico así como las especificidades, sensibilidades y demás parámetros proporcionados por ese punto de corte.

```
## [1] 0.3
```

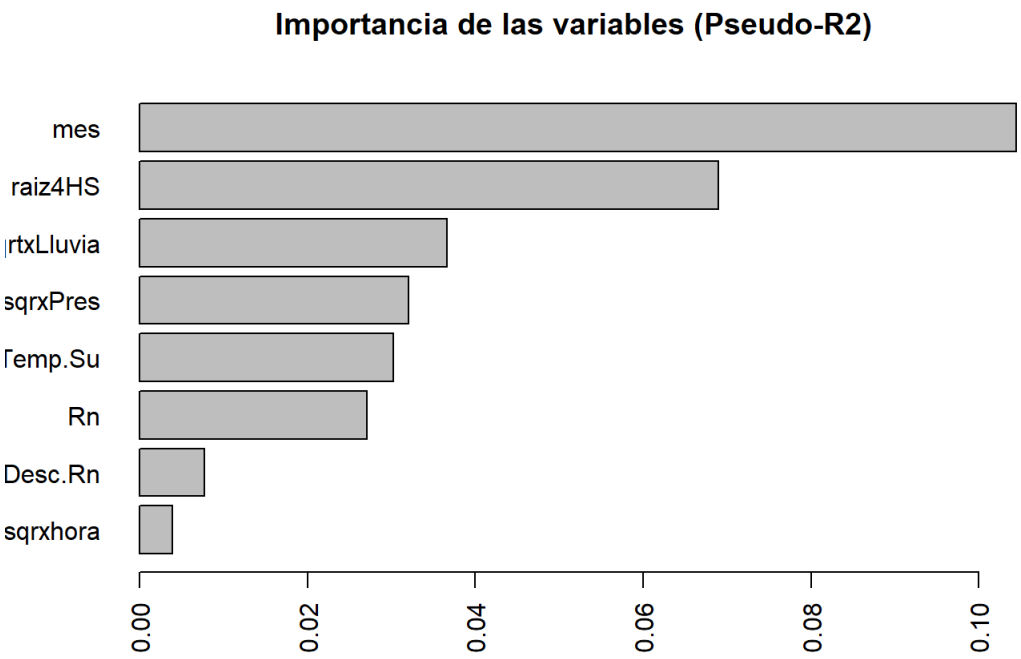
```
## [1] 0.31
```

```
##       Accuracy    Sensitivity    Specificity Pos Pred Value Neg Pred Value
##      0.9360366      0.9414716      0.9332177      0.8796875      0.9684968
```

```
##       Accuracy    Sensitivity    Specificity Pos Pred Value Neg Pred Value
##      0.9366077      0.9381271      0.9358196      0.8834646      0.9668459
```

Podemos observar que el primer punto de corte nos proporcionamás sensibilidad que el segundo pero el segundo más precisión total que el primero. Podríamos elegir a placer con cual hacer las predicciones, cualquiera de los dos va a funcionar bien, pero a la hora de evaluar este tipo de modelos y sus puntos de corte siempre tenemos que identificar las necesidades del modelo (identificar siempre los verdaderos positivos, no tener falsos negativos, no tener falsos positivos, etc…) y elegir en función de estas.

Para finalizar, se muestra la importancia de las variables para nuestro modelo, tras ello una lista de lo que serían los coeficientes beta de la regresión logística que se pueden ver como los coeficientes de la regresión lineal del logit de la variable. LA manera de interpretarlas es que e^(beta) es el odd ratio de presentar el evento frente a no presentarlo, es decir, en el caso de una variable regresora cuantitativa, cuántas veces es más probable presentar el evento por cada aumento de una unidad de esa variable y en el caso de una variable regresora categórica con varios niveles cuántas veces es más probable presentar el evento en caso de estar presente este nivel con respecto al factor de referencia. En el caso de los meses el nivel del factor que se toma como referencia es mes1 (Enero). En lugar de consultar estos betas podemos mirar directamente los odd ratios, donde los ajustados dan directamente e^(beta) y se interpretan como el incremento del ratio entre probabilidad de evento y no evento manteniendo el resto de variables del modelo constantes.




```
##  (Intercept)        mes03        mes04        mes05        mes07        mes08
##  10.95184016  -2.35413194  -3.97589262  -0.91994268   1.85713584   3.67084670
##        mes09        mes10        mes11        mes12  sqrtxLluvia      raiz4HS
##   5.62353168   0.47757093  -3.85530510  -1.06361773   2.17112872 -12.48715607
##     sqrxPres raiz4Desc.Rn  sqrxTemp.Su     sqrxhora           Rn
##  -0.15004305   3.03346855  -0.33342604  -0.10133264   0.07579733
```

```
## Setting levels: control = 0, case = 1
```

```
##
## Call:
## roc.default(response = data_train$targetBin, predictor = predict(modeloDefinitivo_bin,     data_train, ty
pe = "response"), direction = "<")
##
## Data: predict(modeloDefinitivo_bin, data_train, type = "response") in 4614 controls (data_train$targetBin
0) < 2395 cases (data_train$targetBin 1).
## Area under the curve: 0.9789
```

```
## Setting levels: control = 0, case = 1
```

```
##
## Call:
## roc.default(response = data_test$targetBin, predictor = predict(modeloDefinitivo_bin,     data_test, type
= "response"), direction = "<")
##
## Data: predict(modeloDefinitivo_bin, data_test, type = "response") in 1153 controls (data_test$targetBin 0
) < 598 cases (data_test$targetBin 1).
## Area under the curve: 0.9825
```

```
##
## Logistic regression predicting targetBin : 1 vs 0
##
##                          crude OR(95%CI)          adj. OR(95%CI)
## mes: ref.=01
##    03                    0.51 (0.36,0.73)         0.09 (0.05,0.17)
##    04                    0.16 (0.08,0.3)          0.02 (0.01,0.04)
##    05                    1.15 (0.85,1.56)         0.4 (0.24,0.66)
##    07                    23.49 (17.17,32.14)      6.41 (3.23,12.71)
##    08                    69.26 (48.04,99.85)      39.29 (18.26,84.52)
##    09                    641.85 (292.45,1408.65)  276.87 (103.54,740.31)
##    10                    22.94 (16.8,31.31)       1.61 (0.91,2.86)
##    11                    0.38 (0.24,0.61)         0.02 (0.01,0.04)
##    12                    1.14 (0.8,1.62)          0.35 (0.2,0.59)
##
## sqrtxLluvia (cont. var.)  1.2 (1.07,1.36)          8.77 (6.78,11.34)
##
## raiz4HS (cont. var.)      0 (0,0)                  0 (0,0)
##
## sqrxPres (cont. var.)     1 (0.99,1)               0.86 (0.84,0.88)
##
## raiz4Desc.Rn (cont. var.) 2.91 (2.34,3.61)         20.77 (10.05,42.92)
##
## sqrxTemp.Su (cont. var.)  1.3 (1.28,1.32)          0.72 (0.69,0.75)
##
## sqrxhora (cont. var.)     0.95 (0.94,0.97)         0.9 (0.87,0.93)
##
## Rn (cont. var.)           1.03 (1.03,1.04)         1.08 (1.07,1.09)
##
##                          P(Wald's test) P(LR-test)
## mes: ref.=01                            < 0.001
##    03                    < 0.001
##    04                    < 0.001
##    05                    < 0.001
##    07                    < 0.001
##    08                    < 0.001
##    09                    < 0.001
##    10                    0.102
##    11                    < 0.001
##    12                    < 0.001
##
## sqrtxLluvia (cont. var.)  < 0.001        < 0.001
##
## raiz4HS (cont. var.)      < 0.001        < 0.001
##
## sqrxPres (cont. var.)     < 0.001        < 0.001
##
## raiz4Desc.Rn (cont. var.) < 0.001        < 0.001
##
## sqrxTemp.Su (cont. var.)  < 0.001        < 0.001
##
## sqrxhora (cont. var.)     < 0.001        < 0.001
##
## Rn (cont. var.)           < 0.001        < 0.001
##
## Log-likelihood = -1232.253
## No. of observations = 7009
## AIC value = 2498.506
```

Por ejemplo, tenemos que en el mes 9 el ratio de probabilidades entre presentar pico y no presentarlo se eleva hasta 276.87 tomando el mes 1 como referencia, por lo que el mes 9 actúa como un factor a favor del pico, mientras que el mes 04 sería un mes con unas probabilidades de pico muy inferiores a las de no presentarlo. En cuanto a alguna variable cuantitativa tenemos por ejemplo que por cada unidad de incremento en raiz4Desc.Rn se incrementa el odd ratio de presentar pico frente a no presentarlo en 20.77, por lo que es un factor a favor del pico mientras que la raiz4HS sería el factor protector más fuerte, jugando muy a favor de las probabilidades de no presentar pico de estar presente.Por lo general, si el odd ratio de una variable es 1, es un factor neutro, si está por debajo de la unidad es un factor que se suele llamar “protector” en caso de que el evento 1 sea perjudicial, es decir, que juega a favor del evento 0 y si es superior a la unidad es un factor agravante que juega a favor del evento 1.