# Shiny aplicado al estudio de cohorte de pacientes de insuficiencia cardíaca tratados con terapia de resincronización en el Hospital Clínico de Barcelona

Trabajo final de grado

Estadística Aplicada

Universidad Autónoma de Barcelona

Autora: Natalia Sánchez Sierras
Tutor: Roger Borràs

14 de enero de 2020

# *Agradecimientos*

*En primer lugar, deseo expresar mi sincero agradecimiento a mi tutor, Roger Borràs, que, sin su ayuda, no habría sido posible la realización de este trabajo. En concreto, agradecer todas las tutorías hechas, los e-mails enviados, que haya resuelto la cantidad de dudas que me han ido surgiendo a lo largo del trabajo, y por las correcciones hechas. Con todo ello, ha hecho más fácil y llevadero la realización de este proyecto.*

*En segundo lugar, gracias a mis padres por ser los principales promotores de mis sueños, gracias a ellos por cada día confiar y creer en mí y en mis expectativas, por desear y anhelar siempre lo mejor para mi vida.*

*En tercer lugar, quiero dar las gracias a mis dos hermanos, Javi y Óscar, por ser los que más desinteresadamente me han ayudado, por ser mi mayor inspiración y motivación. Sois mi ejemplo a seguir, por lo que este trabajo es tanto mío como vuestro.*

*También me gustaría dar las gracias a Lorena por regalarme palabras de apoyo cuando más se han necesitado y por cuidarme y mimarme como a una hermana.*

*Y por último, infinitas gracias a Miquel, por ayudarme a superar los obstáculos a nivel moral que un proyecto de tal envergadura lleva consigo y por acompañarme en todos estos seis meses. Gracias por hacerme mejor persona.*

# Resumen

*Se estudia una cohorte de pacientes que padecen insuficiencia cardiaca, por lo que se les aplica la terapia de resincronización cardiaca. Las técnicas utilizadas han sido el análisis de la supervivencia y modelos lineales generalizados, donde se ha hecho un pequeño enfoque en las complicaciones del momento de la intervención. Mediante un shinydashboard se ha creado un aplicativo para automatizar todo el procedimiento de análisis, donde además de ser explicado el procedimiento de creación, se han extraído las conclusiones del análisis.*

**Palabras clave: R, Shiny, Shinydashboard, Análisis de supervivencia, Modelos lineales generalizados, Insuficiencia cardiaca, TRC.**

*A cohort have been studied with patients who have heart failure. The techniques used were the analysis of survival and generalized linear models, where a small focus was made on the complications at the time of the intervention. Through a shinydashboard, an application was created to automate the entire analysis procedure, where in addition to explaining the creation procedure, the conclusions of the analysis have been extracted.*

**Keywords: R, Shiny, Shinydashboard, Survival analysis, Generalized linear models, Heart failure, THF.**

# Índice

# 1. Introducción

Para el proyecto de fin de grado, se ha realizado un proyecto en el ámbito biomédico donde se analiza una cohorte de pacientes en el ámbito de la cardiología. Concretamente, se ha estudiado una cohorte de pacientes con la terapia de Resincronización Cardiaca (TRC). Los pacientes de resincronización cardiaca, son una tipología de pacientes que tienen una mala sincronización en el funcionamiento de las cavidades del corazón. El dispositivo de resincronización cardiaca es un dispositivo que tiene como objetivo volver a sincronizar las contracciones de los ventrículos del corazón mediante estimulaciones eléctricas artificiales. Se estudiará diferentes aspectos que son de interés en la práctica clínica diaria y se hará un balance de la experiencia de nuestro centro al largo de los años. Por otro lado, se dispondrá de una base de datos con 671 pacientes operados entre el año 1999 y 2015 en la unidad de arritmias del Hospital Clínico de Barcelona. Por lo que respecta a las técnicas utilizadas, se centrará en un análisis de supervivencia, modelos lineales entre otros.

Por último, y con el objetivo de hacer una visualización más óptima de los datos, y de una manera mucho más interactiva, se estudiará el paquete interno de R llamado "Shiny" el cual te permite hacer páginas web interactivas mediante el programa R, permitirán al usuario interactuar con los datos sin tener que manipular el código. Dicha aplicación web se hará a medida para esta base de datos. Se ha considerado que es una buena oportunidad para dar a conocer esta herramienta, ya que no es necesario tener conocimientos previos de HTML o JavaScript, solo es necesario conocer R.

## 1.1. Motivación

Juntando los dos objetivos explicados anteriormente, se podrá hacer una vista generalizada de los datos y conclusiones extraídas al final del estudio mediante el aplicativo web y además realizar un estudio profundo sobre el análisis de supervivencia que ofrecen estos datos. De esta manera, aquí radica la motivación para realizar este estudio, ya que durante todo el periodo de grado universitario se ha utilizado informes para exponer los resultados de un estudio mediante LaTeX, Word..., lo que conllevaba a unas ciertas restricciones tanto visuales como interactivas. Por lo que vi una oportunidad excepcional poder profundizar y acabar aprendiendo una herramienta como es Shiny, la cual te da la oportunidad de visualizar resultados de una manera diferente.

# 2. Contextualización

La insuficiencia cardiaca es una de las enfermedades cardiovasculares que causan más invalidez y más mortalidad, ya que el músculo cardíaco se debilita y es posible que no pueda bombear suficiente sangre para todo el cuerpo. En 1990 se introdujo en la práctica clínica el uso del marcapasos ventricular. Desde entonces, se ha producido una notable evolución de la terapia de resincronización cardiaca. Los objetivos actuales de la TRC son mejorar la tasa de pacientes que responden al tratamiento y potenciar la respuesta al tratamiento en los individuos que obtienen un efecto beneficioso con la TRC.

La terapia de resincronización cardiaca (TRC) emite señales eléctricas que hacen latir al corazón de tal forma que maximice la cantidad de sangre que bombea hacia afuera. Este tratamiento puede reducir los síntomas de la insuficiencia cardíaca y los riegos de sufrir complicaciones, incluso la muerte.

Requiere un procedimiento de cirugía donde se implanta un dispositivo en el pecho. Los tipos de dispositivos para la terapia son los siguientes:

- Terapia de resincronización cardíaca con un marcapasos.
- Terapia de resincronización cardíaca con un marcapasos y un desfibrilador cardiovascular implantable.

Dicha información queda recogida en una de las variables que se tendrán en cuenta en este estudio.

## 2.1. Base de datos

Se dispone de una base de datos con observaciones de 671 pacientes operados entre el año 1999 y 2015 en la unidad de arritmias del Hospital Clínico de Barcelona. Para modelar la variable respuesta, tiempo que transcurre dentro del estudio hasta que sucede el evento, se trabajará con unas 14 características de cada individuo. La tabla de la que se dispone es la siguiente:

| SEXE | EDAT | FE | CF | CREA | FGR | BBE | ISQUEMIC | INFART | FA |
|---|---|---|---|---|---|---|---|---|---|
| DONA | 67.00 | 20.00 | III | 1.10 | 52.66 | SI | NO | NO | NO |
| HOME | 65.00 | 16.00 | III | 1.10 | 71.40 | SI | NO | NO | NO |
| HOME | 65.00 | 28.00 | III | 1.90 | 38.00 | NO | SI | SI | SI |
| HOME | 66.00 | 18.00 | III | 0.40 | 228.75 | SI | SI | SI | SI |

| TV | QRS | MODE | COMPLICACIONS | VIU_O_MORT_lf | TEMPS_KM_EP1 |
|---|---|---|---|---|---|
| NO | 200.00 | TRC-P | NO | 0.00 | 15.89 |
| SI | 180.00 | TRC-D | NO | 0.00 | 0.22 |
| NO | 160.00 | TRC-P | NO | 1.00 | 4.89 |
| NO | 165.00 | TRC-D | NO | 0.00 | 1.00 |

**Cuadro 1:** Base de datos

Ya que el nombre de las variables no queda bien explicado en la tabla anterior, seguidamente se explicará brevemente qué información recoge cada una de estas características.

- **SEXE:** (HOME, DONA) Sexo del individuo.
- **EDAT:** Edad del individuo.
- **FE:** Es una variable numérica, que indica la fracción de eyección, es decir, es una medida del porcentaje de sangre que sale del corazón cada vez que se contrae.
- **CF:** Indica la clase funcional, en otras palabras, es la capacidad de movilidad que tiene la persona.
- **CREA:** Creatinina. La insuficiencia renal es un importante factor pronóstico en pacientes con insuficiencia cardiaca. Para valorar la función renal se suelen utilizar las cifras de creatinina.
- **FGR:** Es el filtrado glomerular, esto es el volumen de fluido filtrado por unidad de tiempo desde los capilares glomerulares renales hacia el interior de la cápsula de Bowman. Normalmente se mide en mililitros por minuto.
- **BBE:** Bloqueo de rama izquierda.
- **ISQUEMIC:** Si el individuo presenta isquemia cardiaca.
- **INFART:** Si ha tenido un infarto.
- **FA:** Fibrilación auricular.
- **TV:** Si ha presentado una taquicardia en la parte superior
- **QRS:** El complejo QRS es la representación gráfica de la despolarización de los ventrículos del corazón formando una estructura picuda en el electrocardiograma.
- **MODE:** Indica si la TRC se hace sola (TRC-P) o la TRC se le añade desfibrilador (TRC-D).
- **COMPLICACIONS:** Existencia de complicaciones en el proceso del tratamiento.

- **VIU_O_MORT_lf:** Indicador de si ha pasado el evento de interés.
- **TEMPS_KM_EP1:** Tiempo que pasa dentro del estudio.

Además de estas variables que son las que se recogen de la base de datos, durante el estudio se ha querido que el usuario pudiera categorizar las variables numéricas, para poder extraer más conclusiones, tanto en el análisis descriptivo como en el de supervivencia, ya se explicará con más detalle más adelante.

#### 2.1.1. Validación

Respecto a la validación de la base de datos para acabar realizando un buen estudio, únicamente se ha encontrado 1 valor nulo el cual se ha eliminado. Todas las demás variables se han dejado como estaban, exceptuando "$VIU\_O\_MORT\_lf$", ya que se ha recategorizado (VIU,MORT) en (0,1) respectivamente, ya que como ya se verá en secciones futuras, la función surv() necesaria para hacer el análisis de supervivencia necesita este formato.

### 2.2. Lenguaje utilizado

Para todo el procesamiento de datos, la herramienta utilizada ha sido `R`. Se han utilizado conceptos de análisis de supervivencia, ya que como se ha comentado anteriormente, se dispone de datos censurados por la derecha, por lo que se han aplicado paquetes como `survival`, `surfit`, `ggsurvplot`, etc. Además, para una mejor visualización de los resultados se ha utilizado una paquetería de `R` llamada `Shiny`. Todas estas herramientas serán explicadas con mayor detalle en los siguientes apartados.

## 3. Métodos

En esta sección se explicarán brevemente los conceptos ya vistos en clase sobre cómo hacer un análisis de supervivencia, y de modelos lineales generalizados, y se irá un poquito más lejos explicando todo el proceso de creación del aplicativo web donde el usuario podrá interaccionar con los inputs para tener una visión más general de los posibles resultados, sin necesidad de tocar el código.

### 3.1. Shiny

En primer lugar, hay que saber que Shiny es una herramienta para crear fácilmente aplicaciones web interactivas que permiten a los usuarios interaccionar con sus datos sin tener que manipular el código. Cada app es una carpeta que debe de contener un archivo llamado App.R. En el interior de este archivo .R debe de tener tres partes principales que son las siguientes:

- **ui:** Es una descripción de la interfaz, es decir, es una secuencia de comandos que controlan la estética visual de la aplicación.
- **server:** Es el código necesario que constituyen los componentes de R de la app. Son las instrucciones que necesita para construir la App.
- **ShinyApp(ui,server):** Es la función final para ejecutar la web.

Por otro lado, hace falta destacar dos conceptos importantes, la interactividad y la reactividad dentro de Shiny. Como ya se ha dicho anteriormente, la principal característica sería la interactividad, y esto está altamente correlacionado con la programación reactiva en la que el usuario cada vez que modifica algo, se renueva todo el proceso. A grandes rasgos hay tres tipos de objetos que tienen relación con la programación reactiva:

- **Reactive sources:** Son aquellos inputs que se ponen en la parte *ui* y se envían a *server*.
- **Reactive conductors:** Es básicamente una transformación de los inputs que se usan en el *server*.
- **Reactive endpoints:** Son los outputs que están en el la parte *server* y se envían al *ui*.

De aquí salen otros dos conceptos que hay que conocer antes de explicar la App, los inputs y outputs. Los llamados inputs pueden ser tanto números como parámetros lógicos, por lo que Shiny dispone de los siguientes inputs que se incorporan mediante lo que se denominan widgets:

| Inputs | Descripción |
|---|---|
| actionButton | Botón para accionar |
| checkboxGroupInput | Grupo de casillas de verificación |
| checkboxInput | Una sola casilla de verificación |
| dateInput | Calendario para ayudar a seleccionar la fecha |
| dateRangeInput | Un par de calendarios para seleccionar un rango de fechas |
| fileInput | Asistente de control de carga de archivos |
| helpText | Texto de ayuda que se puede agregar a un formulario de entrada |
| numericInput | Campo para ingresar números |
| radioButtons | Conjunto de botones de radio |
| selectInput | Cuadro con opciones para elegir |
| sliderInput | Barra deslizante |
| submitButton | Botón de enviar |
| textInput | Campo para ingresar texto |

**Cuadro 2:** Los inputs y su respectiva descripción

Una vez ya se ha insertado un input en la parte *ui* se utilizará como *input$nombre_variable* dentro de la estructura *server*.

Seguidamente, para saber cómo se pasa de inputs a outputs hay que tener en cuenta que los inputs anteriores se introducen en la parte *ui* y se envían a la parte *server* y su utilidad radica en obtener los outputs. Son de tipo reactivos aquellos procesos del server que hacen servir los inputs para encontrar un resultado de outputs. Los posibles objetos que se pueden obtener son los siguientes:

| Outputs | Descripción |
|---|---|
| renderImage | Imágenes(guarda el link de donde se ubica la imagen) |
| renderPlot | Gráficos |
| renderPrint | Muestra el contenido del output |
| renderTable | Muestra una tabla o matriz |
| renderText | Muestra un conjunto de carácteres |
| renderUI | Etiqueta de objecto, o un HTML |

**Cuadro 3:** Los outputs y su respectiva descripción

Por último, los resultados que se han obtenido con el proceso anterior se devuelven al *ui* donde se deberán de mostrar los resultados que se hayan indicado. Según el tipo de output, debe de poner en la parte *ui* los siguientes tipos de outputs:

| Tipo de Output | Significado |
|---|---|
| htmlOutput | raw HTML |
| imageOutput | Imagen |
| plotOutput | Gráfico |
| tableOutput | Tabla |
| textOutput | Texto |
| uiOutput | visualización con HTML |
| verbatimTextOutput | Texto |

**Cuadro 4:** Tipo de Outputs

Por lo tanto, teniendo ya los pilares fundamentales de la estructura de una aplicación Shiny, ya se podría comenzar a explicar en profundidad como se ha llevado a cabo dicho estudio y como se ha hecho y estructurado la App en cuestión. Dicha información estará recogida en la Sección 4. A continuación, se hará un repaso sobre los conocimientos utilizados en el ámbito de la supervivencia.

## 3.2. Análisis de supervivencia

Se dispone de una base de datos donde existen valores censurados por la derecha, que esto es, de aquellos individuos donde no ha sido posible ver al completo el tiempo de seguimiento, ya sea porque en el último instante de observación del individuo aún no ha sucedido el acontecimiento de interés, o también por un abandono o la posibilidad de que a un individuo le haya sucedido algo externo al estudio que le imposibilite acceder a él.

### 3.2.1. Kaplan i Meier

Para comenzar se ha hecho un análisis donde se observa la curva de Kaplan i Meier. Una de las mejores virtudes que tiene dicho método es que no se tiene que asumir previamente casi nada sobre los datos que se quieren tratar, únicamente que los datos censurados se comportarán del mismo modo que los que han sido seguidos hasta la última observación del investigador. De esta manera hace mucho más llevadero el estudio.

La principal característica de dichas gráficas es que son curvas escalonadas. Mediante vaya transcurriendo el tiempo los escalones se van formando, de manera que cada vez que sucede el acontecimiento del estudio (fallecimiento) se crea un escalón hacia abajo, lo cual lleva a la reducción de la supervivencia en ese mismo instante.

El método Kaplan i Meier también nos aporta los valores de la media y mediana del tiempo de supervivencia, donde hay que tener en cuenta que, en este tipo de estudios, la media puede verse afectada por los valores censurados, en cambio la mediana aportará una información más invariante a cualquier cambio de escala, de este modo al ser más robusta, hace que no le influyan los datos censurados.

Haciendo una definición más exacta, siendo $S(t)$ la función de supervivencia de una determinada población, es decir, la probabilidad de que uno de sus integrantes viva más allá de un tiempo t. Para una muestra de esta población de tamaño N, sean

$$t_1 \leq t_2 \leq t_3 \leq ... \leq t_N \tag{1}$$

los tiempos que pasan hasta la muerte de todos ellos. Entonces, para cada $t_i$ se define:

- $d_i$, el número de muertes en el momento $t_i$ y
- $n_i$, el número de individuos en riesgo justo antes de $t_i$.

El estimador de Kaplan y Meier de $S(t)$ es:

$$\widehat{S}(t) = \prod_{t_i<t} \frac{n_i - di}{n_i} \tag{2}$$

Por último, para comparar las curvas de supervivencia se ha hecho el test log-Rank ya que es el método favorito para comparar curvas de supervivencia para diferentes niveles de una misma variable. Dicha prueba tiene en cuenta todo el rango de tiempo y no únicamente un momento arbitrario. Otra ventaja es que no necesita que el investigador sepa nada sobre la distribución de los datos.

### 3.2.2. Modelo de Cox

El modelo de Cox, a la hora de interpretar los coeficientes, es algo más simple que el resto de los modelos. Este modelo trabajo primordialmente con la función de riesgo y es utilizado para detectar relaciones existentes entre el riesgo que se produce en un determinado individuo en el estudio y algunas variables independientes o explicativas; por lo que este modelo nos permite evaluar dentro de un conjunto de variables cuáles tienen relación o influencia sobre la función de riesgo y por ello también en la función de supervivencia, ya que ambas funciones están conectadas.

La función de riesgo en este modelo es la siguiente:

$$h(t|x) = h_o(t)exp(x'\beta) \tag{3}$$

Tenemos un modelo semiparamétrico porque mientras que el riesgo basal, $h_o(t)$, puede tomar cualquier forma, las covariables entran a través de un modelo lineal con sus correspondientes parámetros. En dos puntos diferentes, $x_1$ y $x_2$, la proporción de riesgos para dichas observaciones son:

$$\frac{h(t|x_1)}{h(t|x_2)} = \frac{h_o(t)exp(x_1'\beta)}{h_o(t)exp(x_2'\beta)} = \frac{exp(x_1^t\beta)}{exp(x_2^t\beta)} = exp[(x_1' - x_2')\beta] \tag{4}$$

La expresión anterior es conocida como la hipótesis de riesgos proporcionales. Esta proporción también es conocida como la razón de riesgos.

## 3.3. Modelo lineal generalizado

Se aplicará una regresión logística, en la cual se hará mediante la función glm() implementada en el R. En este casi se utiliza el modelo LOGIT que nos da ciertas ventajas en comparación a un modelo lineal de probabilidad, estimada por mínimos cuadrados ordinarios (MCO). Los modelos de regresión con respuesta cualitativa son modelos de regresión en los cuales la variable dependiente puede ser de naturaleza cualitativa, mientras que las variables independientes pueden ser cualitativas o cuantitativas, o una mezcla de las dos. La variable cualitativa en estos tipos de modelos no tiene que restringirse simplemente a respuestas de sí o no, la variable respuesta puede tomar más de dos valores, ser tricotómica o politómica, también se establecen modelos en lo que la variable dependiente es de carácter ordinal o

de carácter nominal, en donde no hay preestablecido ningún tipo de orden. En este trabajo se analizará el modelo LOGIT en donde la variable dependiente, complicaciones, es de carácter binario o dicotómico (sí o no). Se trata pues de adoptar una formulación no lineal que obligue a que los valores estimados estén entre 0 y 1 ya que, la regresión con una variable binaria dependiente Y modeliza la probabilidad de que Y = 1. La regresión LOGIT utiliza una función de distribución logística, su función de distribución de probabilidad da lugar a probabilidades ente 0 y 1, y presenta un crecimiento no lineal (con mayores incrementos en la parte central).

$$Logit(D) = log(\frac{\pi}{1-\pi}) = L(\mathbf{X}) = \beta_0 + \sum_{j=1}^{p} \beta_j X_{j.} \tag{5}$$

Además de la interpretación de las estimaciones, se hace la interpretación de los parámetros OR (odds ratio) y RR (riesgo relativo), los cuales se calculan a partir de las siguientes fórmulas:

$$\hat{OR}(D) = \exp^{\hat{\beta_j}} \tag{6}$$

$$CI_{1-\alpha}(OR_{X_j}) = exp(\beta_j \pm z_{1-\frac{\alpha}{2}} \sqrt{\hat{Var}(\hat{\beta_j})}) \tag{7}$$

$$RR = \frac{P(D|E)}{P(D|\bar{E})} \tag{8}$$

$$CI_{1-\alpha}(RR) = \hat{RR}exp(\pm z_{1-\frac{\alpha}{2}} \sqrt{\hat{Var}(log(\hat{RR}))}) \tag{9}$$

donde

$$\hat{RR} = \frac{n_{++}(n_{-+} + n_{--})}{n_{-+}(n_{++} + n_{+-})} \tag{10}$$

y

$$\hat{Var}(log(\hat{RR})) = \frac{n_{+-}}{n_{++}(n_{++} + n_{+-})} + \frac{n_{--}}{n_{-+}(n_{-+} + n_{--})} \tag{11}$$

## 3.4. Enlaces - Repositorio GitHub y página web creada

Para una mejor visualización de los resultados, y para facilitar todo el proceso de visualización tanto del código como de la página web, se ha creado un repositorio GitHub donde se encuentra el código necesario para la creación de la página web, el pdf y la base de datos utilizada.

El enlace del GitHub es el siguiente:

**https://github.com/ntlsanchez/TFG**

Por otro lado, el enlace donde se ubica mi página web, y donde podéis interactuar con ella es el siguiente enlace:

**https://nataliasanchezapp.shinyapps.io/sanchezsierrasnataliatfg/**

# 4. Resultados

Después de consolidar las bases necesarias para poder empezar con dicho estudio, se comenzará a explicar cómo se ha llevado a cabo el aplicativo web y se explicará el código utilizado. Seguidamente se irán mostrando los diferentes apartados desarrollados en la web. Por último, se extraerán los resultados más destacados sobre el estudio.

## 4.1. Aplicativo web

En esta sección se mostrará y explicará la parte del código que se crea necesaria destacar de cada apartado del estudio.

Para empezar, hay que recordar que el código web se divide en dos partes diferenciadas, el *ui* y el *server*. Como página principal, donde se puede visualizar en la Figura 1, se ve bastante bien los objetos principales que se utilizan para crear la página web:

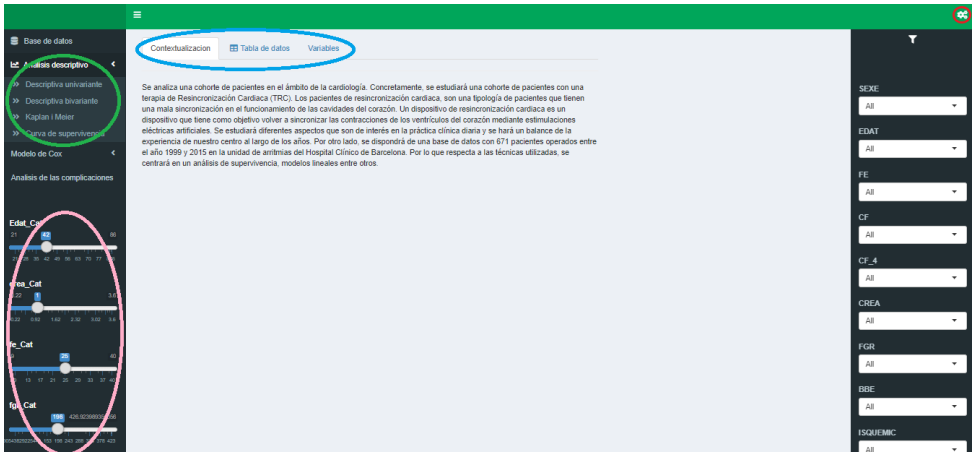


**Figura 1:** Primera sección de la página web

El círculo verde indica como se estructura el estudio, es decir, los subapartados que se quieren estudiar. Una parte del código es la siguiente:

```
ui <- dashboardPagePlus(skin = "green",
        header = dashboardHeaderPlus(
        enable_rightsidebar = TRUE,
        rightSidebarIcon = "gears"
      ),
dashboardSidebar(
    sidebarMenu(##Lado izquierdo

menuItem("Base de datos", tabName = "Basededatos", icon=icon("database")),

menuItem("Análisis descriptivo", tabName = "Descriptivos",  icon=icon("chart-line"),
         menuSubItem("Descriptiva univariante", tabName="uni"),
         menuSubItem("Descriptiva bivariante", tabName="biv"),
         menuSubItem("Kaplan i Meier",  tabName ="KM1"),
         menuSubItem("Curva de supervivencia", tabName="Sup")),
```

Dentro de la función dashboardSidebar(), que indica lo que incluirá la barra lateral izquierda, debe de tener incluido un sidebarMenu(), y dentro de este un menuItem() por cada sección que se quiera hacer.

Por lo que respecta al círculo rosa, dichas variables son utilizadas para que el usuario pueda categorizar las variables numéricas del estudio y escoja el punto de corte por el cual quiera visualizar el estudio. Dichas variables se pueden utilizar en el análisis descriptivo. Teniendo en cuenta que seguimos estando dentro de la función dashboardSidebar(), el código es el siguiente:

```
(...)
sliderInput("edat",
                  "Edat_Cat",
                  min = min(dataset$EDAT),
                  max = max(dataset$EDAT),
                  value = 42),
      sliderInput("crea",
                  "crea_Cat",
                  min = min(dataset$CREA),
                  max = max(dataset$CREA),
                  value = 1),
(...)



server <- function(input, output, session){
  bajas<- reactive({

dataset$Edat_Cat <- ifelse(dataset$EDAT < input$edat, paste0("Edat<",input$edat),
                           paste0("Edat>=",input$edat))
    dataset$Edat_Cat <- as.factor(dataset$Edat_Cat)

    dataset$crea_Cat <- ifelse(dataset$CREA < input$crea, paste0("Crea<",input$crea),
                               paste0("crea>=",input$crea))
    dataset$crea_Cat <- as.factor(dataset$crea_Cat)

   (...)

    data <- dataset
    (...)
```

Se observa que la función sliderInput() se utiliza para que el usuario indique que input quiere exactamente, donde se le indica el rango del mínimo al máximo. En la parte *server* se hace el filtro, que guarda el número que el usuario ha indicado y crea una variable que categoriza la variable numérica original. Todo esto está dentro de una función reactiva que cambia cada vez que el individuo decida cambiar el valor de la variable que quiere categorizar.

En segundo lugar, mirando el contenido del círculo verde, tenemos los llamados Tabs, es una forma de organizar el cuerpo de la web:

```
dashboardBody(
   tabItems(
      tabItem(tabName = "Basededatos", mainPanel(
        tabsetPanel(type = "tabs",
 tabPanel("Contextualizacion",hr(),p(...)),

 tabPanel("Tabla de datos", icon = icon("table"),dataTableOutput("table1"),
          dataTableOutput("table2")),
 tabPanel("Variables", hr(), h2(...)))))),
```

Dentro de la función dashboardBody(), se crea un tabItem para indicarle con que botón de la barra lateral izquierda se quiere visualizar dicha estructura, en este caso se le indica "tabName- "Basededatos", se añade un mainPanel() que servirá para crear los tabPanel().

Además, se ha querido crear un filtro general, por si el usuario quiere filtrar la base de datos, y centrarse solo en un conjunto de datos en específico. Esto se ve reflejado en la parte superior derecha, círculo rojo.

```
(...)
rightsidebar= rightSidebar(
        background = "dark",
        rightSidebarTabContent(
          id = "right_sidebar_tab_1",
          icon = "filter",
          active = TRUE,
          uiOutput("right_menu")))

(...)

  dim_SEXE <- dataset %>% distinct(SEXE) %>% as.data.frame() %>% c("All", .)
  dim_EDAT   <- dataset %>% distinct(EDAT) %>% as.data.frame() %>% c("All", .)
(...)

  selected_CREA <- reactive({
    ifelse(is.null(input$CREA), "All", input$CREA)
  })

(...)


  bajas<- reactive({

(...)
    req(NoneEmpty(input$SEXE, input$EDAT, input$FE,
                  input$CF,input$CF_4, input$CREA, input$FGR, input$BBE, input$ISQUEMIC,
                  input$INFART, input$FA, input$TV, input$QRS))
    data <- dataset

    if (selected_SEXE() != "All") {
      data %<>% filter(SEXE == selected_SEXE())
    }

(...)

    return(data)
  })


  ## Right sidebar
  output$right_menu <- renderMenu({
    tagList(
      selectInput(
        "SEXE",
        "SEXE",#Titulo
        dim_SEXE,
        selected = isolate(selected_SEXE())
      ),
(...)
```

Este código es algo más complejo de lo que se ha mostrado hasta el momento. En primer lugar, en la parte *ui* se ponen las primeras líneas de código, es decir, la función rightSidebar(), seguidamente en la parte server se ha de definir la "*dim_nombrevariable*"donde se hace una distinción de la base de datos para las diferentes categorías de cada variable. En segundo lugar se tiene que hacer una función reactive() donde se tendrá en cuenta lo que el usuario indique, teniendo en cuenta que siempre está presente la opción *All*, la cual indica que se coge todas las categorías de la variable en cuestión. En tercer lugar, se crea la base de datos nueva con las nuevas indicaciones que el usuario quiera, por lo que se crea un if() que se ejecuta si la opción es diferente a *All* y filtra la base de datos. Por último, dentro de la función renderMenu(), se crea para todas las variables del estudio un input con la función selectInput(), donde se indica la función isolate() que servirá para que reaccione cada vez que alguien cambie algún valor.

Los puntos suspensivos indican que hay más líneas de código.

Otras de las cosas interesantes que hay que destacar es la visualización de tablas en Shiny. En la Figura 2:

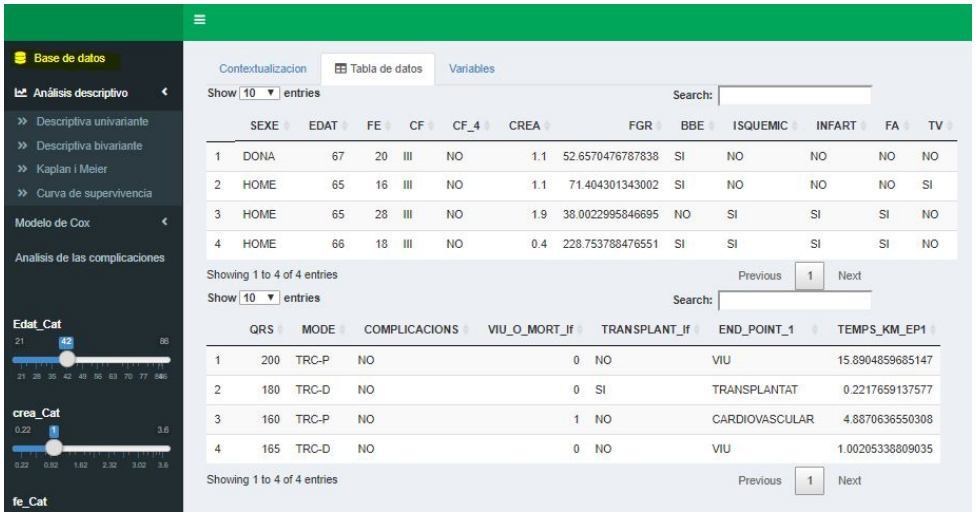


**Figura 2:** Visualización de la base de datos

```
(...)
tabPanel("Tabla de datos", icon = icon("table"),dataTableOutput("table1"),
         dataTableOutput("table2"))
(...)
 output$table1 <- renderDataTable({
    B <- bajas()
   head(B[,1:12],4)
  })

  output$table2 <- renderDataTable({
    B <- bajas()
    head(B[,c(13:15,18:21)],4)
  })
```

La primera línea es la que se ubica en la parte *ui* y se utiliza el dataTableOutput() donde se le indica el nombre que se ha puesto en la parte server que son las otras líneas siguientes donde se utiliza la función

renderDataTable().

Respecto a los otros apartados, dentro de la descriptiva univariante se ven cosas bastante interesantes como la que se ve en la siguiente Figura 3:

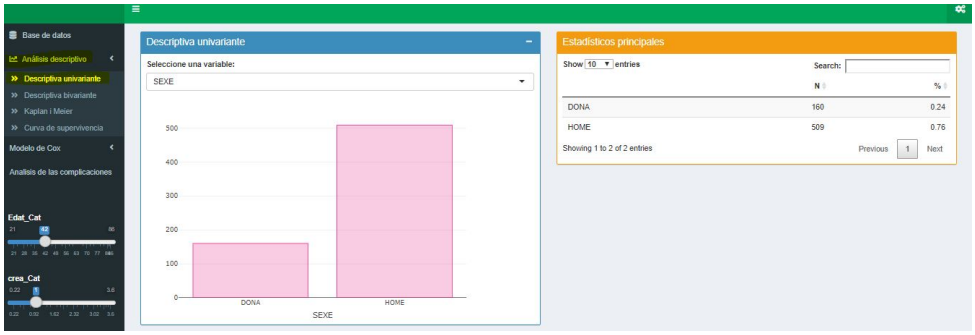


**Figura 3:** Descriptiva univariante - Gráficos

Este gráfico, como todos los demás que hay en el aplicativo web, son completamente interactivos y cambian según lo que el usuario indique en las barras laterales y en la misma opción de dicho gráfico. Se visualizan los estadísticos principales tanto en la tabla de la derecha como encima del gráfico si pasas el cursor por encima y además depende de la clase de la variable saldrá un gráfico u otro.

```
(...)
  selectInput(inputId = "n", label = "Seleccione una variable:",
  choices = c(names(dataset)),selected = "SEXE"),plotlyOutput(outputId = "univ"))
(...)
output$univ <- renderPlotly({
    B <- bajas()
    variable1 <- variable()

    if(class(B[[variable1]])=="factor"){
      B <- bajas()

      B %>%
        group_by(!!as.name(variable1)) %>%
        summarise(models=n()) %>%
        ungroup %>%
        plot_ly() %>%

        add_bars(
          x = as.formula(paste0("~`", variable1, "`")),
          y = ~models,
          name = "2019",
          yaxis = "y2",
          marker = list(
            color = "rgba(226, 0, 116, 0.2)",
            line = list(color = "rgb(226, 0, 116, 0.1)", width = 0.9)))
      }else{
      plot_ly(B, x = B[[variable1]], type="box",mode = 'markers')}})
```

En la parte *ui* se escriben las dos primeras líneas de código donde se utiliza la función selectInput() para indicar que variable se quiere ver, y plotlyOutput() para que aparezca el gráfico. Las demás líneas con las que se ubican en la parte *server* donde se realiza el gráfico. Se crea un if() para distinguir entre

la clase de las variables y haga un gráfico u otro dependiendo de la variable. La parte del código donde se define B, es la base de datos reactiva que depende de lo que el usuario haya indicado. En este caso es muy importante el paquete llamado *plotly* el cual te deja hacer gráficos como este.

La penúltima cosa del aplicativo web que quiero resaltar es lo que sale en la siguiente Figura 4:


**Figura 4:** Descriptiva univariante - Tabla de individuos

Este gráfico y tabla se ha hecho con el siguiente código:

```
(...)
fluidRow(box(status="danger",
        width = 12,
        title = "Click or select on the plot to fill the table!",
        plotOutput('seleccionable',
                   click = "user_click",
                   brush = "user_brush"),
        dataTableOutput("tabla_seleccionable")
(...)
 interaction_type <- "click"
  observeEvent(input$user_click, interaction_type «- "click")
  observeEvent(input$user_brush, interaction_type «- "brush")

   output$seleccionable <- renderPlot({
    variable1 <- input$s
    B <- bajas()
    B %>%
      ggplot(aes(!!as.name(input$n), VIU_O_MORT_lf)) +
      geom_jitter(width = 0.25, aes(colour =  VIU_O_MORT_lf)) +
      theme(legend.position = "none") +
      theme(
        axis.text.x = element_text(angle = 45, hjust = 1,
                                   size=14, face = "bold"),
        axis.text.y = element_text(size=14, face = "bold")) +
      xlab("Grupo tarifa")
  })


  brush_data <- reactive({
```

```
    B <- bajas()
    user_brush <- input$user_brush
    user_click <- input$user_click
    if (interaction_type == "brush") {
      brushedPoints(B, user_brush)
    } else if (interaction_type == "click") {
      nearPoints(B, user_click, threshold = 10)
    }
  })

  output$tabla_seleccionable <- renderDT({
    df <- brush_data()
    req(nrow(df) > 0)
    df %>% datatable(
      rownames = FALSE,
      colnames = colnames(dataset),
      selection = "single",
      extensions = "Buttons",
      filter = "top",
      options = list(
        dom = "Blfrtip",
        scrollX = TRUE
        ,buttons = list("copy", list(
          extend = "collection",
          buttons = c("csv", "excel", "pdf"),
          text = "Download"
        ))
```

Esto es algo digno a destacar, ya que se ha conseguido crear un gráfico en el cual el usuario puede seleccionar cualquiera de los puntos que dispone el gráfico y te aparece en la parte inferior una tabla con los individuos que son. Para llevar a cabo esto han sido necesarias funciones como renderDT(), reactive(), observeEvent() y renderPlot().

Por lo que respecta a la descriptiva bivariante, no se expondrá el código ya que es muy similar que el que ya se ha explicado en la descriptiva univariante, pero sí que se visualiza la Figura 5:

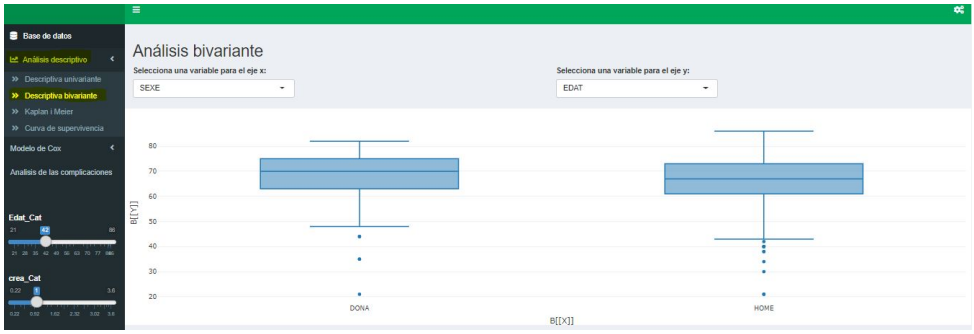


**Figura 5:** Descriptiva bivariante - Gráfico

Aparece un gráfico, donde el usuario puede especificar que variable ocupa el eje izquierdo o derecho. Además igual que en la descriptiva univariante, se crea un if() para distinguir entre las clases de las variables y poner un gráfico u otro. Por otro lado, se muestra abajo el p-valor de la prueba que se ha llevado a cabo con el objetivo de hacer un mejor estudio de ambas variables.

En la siguiente tabla se hace un resumen de cuales son:

| x | y | Test | Descripción |
|---|---|---|---|
| numeric | numeric | Spearman | Correlación |
| numeric | factor | Willcoxon | Comparación de medias |
| factor | numeric | Willcoxon | Comparación de medias |
| factor | factor | ji-cuadrado de Pearson | Asociación |

**Cuadro 5:** Test aplicado en la descriptiva bivariante dependiendo de la clase de ambas variables

Hace falta destacar que se aplica la correlación de Spearman ya que a priori se hace una prueba de normalidad para ambas variables, y al dar el p-valor significativo se aplica este y no el de Pearson, ya que este último requiere normalidad en los datos.

Otro de los apartados es el de Kaplan-Meier donde además de la curva de supervivencia se muestra el p-valor de la prueba Log-Rank ya explicado en la Sección 3.2.1.

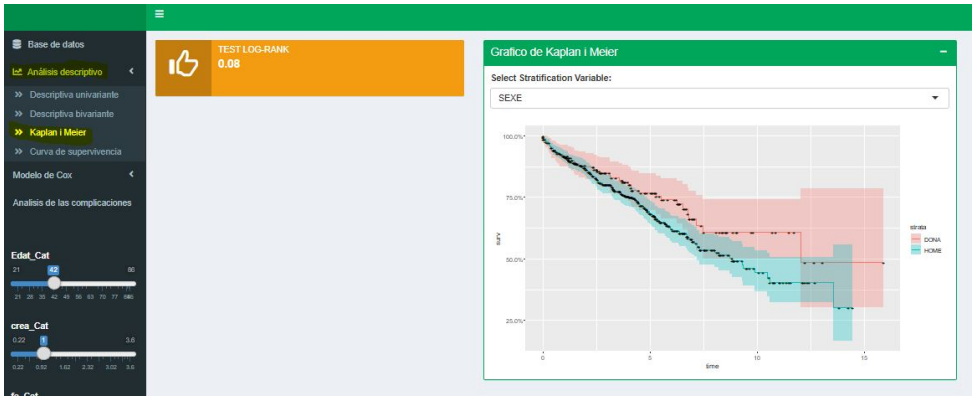


**Figura 6:** Sección de Kaplan-Meier

En la Figura 6 es importante destacar la manera en la que hay que definir el modelo de supervivencia:

```
kmdata <- surv_fit(as.formula(paste('Surv(TEMPS_KM_EP1,VIU_O_MORT_lf)~',input$s)),
                   data=B)
```

Shiny no admite el símbolo que es necesario para definir un modelo, por lo que hay que definirlo como una fórmula y utilizar la función paste(). Esto es muy importante en el momento de definir un modelo, ya que no se ha encontrado otra forma de hacerlo para que funcione.

Para finalizar se quiere destacar tanto para el apartado del modelo de Cox como para el análisis de las complicaciones, como se ha hecho para que el usuario pueda escoger que variables insertar en el modelo. El código ha sido el siguiente:

```
(...)
 uiOutput("independent")
(...)

 output$independent <- renderUI({
    checkboxGroupInput("independent", "Independent Variables:",choices =
                         names(B)[2:14])
  })

 runRegression <- reactive({
    B <- bajas()
    coxph(as.formula(paste("Surv(TEMPS_KM_EP1,VIU_0_MORT_lf)~",paste0(
      input$independent,collapse="+"))),data=B)
  })
```

Se define como *independent* aquellas variables que el individuo quiera añadir en el modelo *runRegression*. Sobre todo, es necesario destacar que hace falta añadir el argumento *collapse*=+ para que se pueda añadir más de una variable.

La estética de esta parte sería la que se visualiza en la Figura 7:

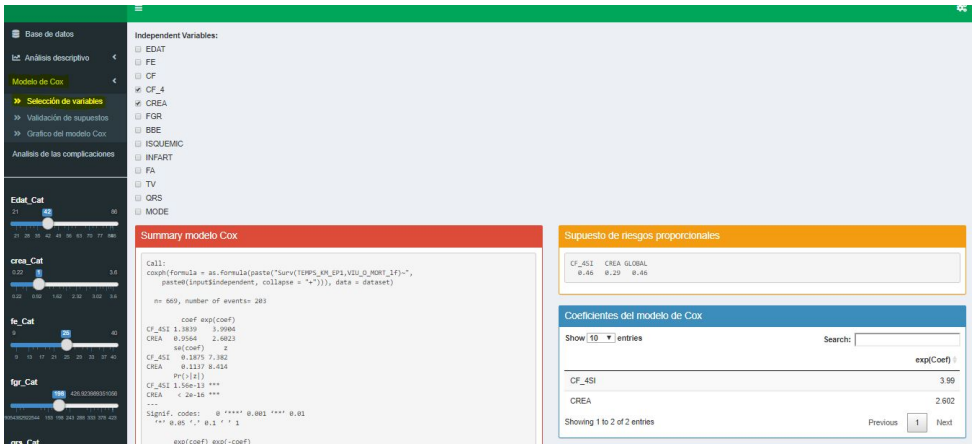


**Figura 7:** Modelo de Cox - Añadir variables

Para terminar con esta sección es necesario explicar cómo se ha hecho para subir este aplicativo web a internet. Se ha utilizado la web *shiny.io*, la cual te deja tener como máximo 5 aplicaciones, con un límite de uso de 25 horas al mes.

## 4.2. Resultados del estudio

Una vez explicado cómo se ha construido el aplicativo web, se ha podido utilizar para sacar un mayor número de conclusiones del estudio.

En primer lugar, respecto a la descriptiva se observan los puntos siguientes:

- El 76 % de la muestra son hombres y lo restante mujeres.
- Hay más pacientes que presentan una clase funcional de estado III, exactamente 444 personas.
- A cuanta más edad más valores elevados tanto en creatina como en QRS, esto se puede observar en los siguientes gráficos:

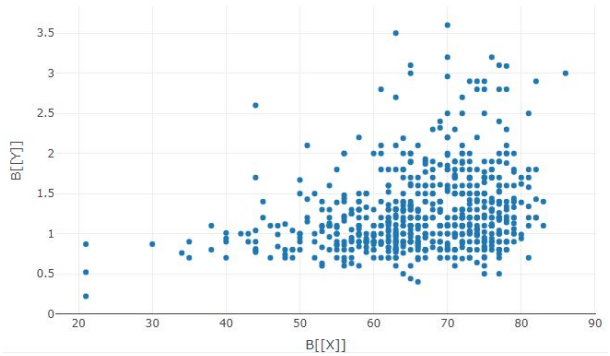


**Figura 8:** Gráfico de puntos donde en el eje x está la edad y en el eje y la creatina

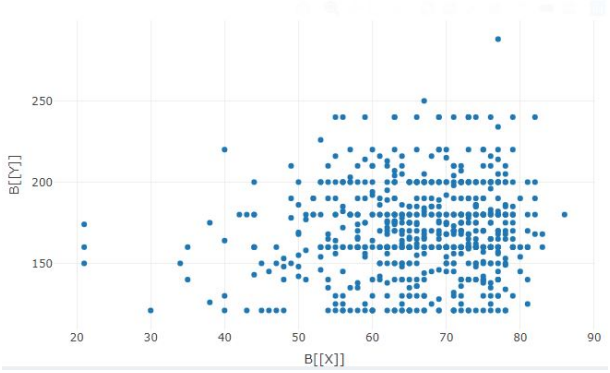


**Figura 9:** Gráfico de puntos donde en el eje x está la edad y en el eje y el qrs.

- Se observan correlaciones negativas, aunque muy leves, entre los valores de FGR y Creatina, y entre los valores de FE respecto a la creatina y QRS.
- Respecto a la variable sexo se ven diferencias significativas en la media tanto con la variable edad como en los valores de la creatina. El sexo se ve asociada con la variable FA.

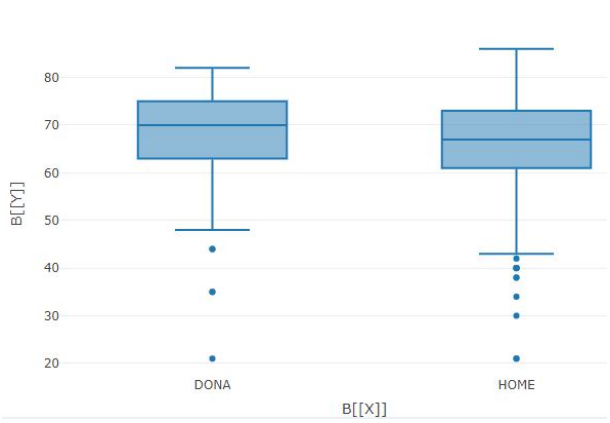


**Figura 10:** Gráfico de caja donde en el eje x está el sexo y en el eje y la edad

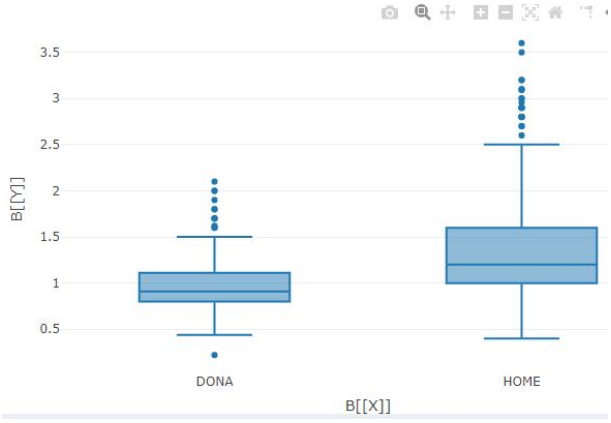


**Figura 11:** Gráfico de caja donde en el eje x está el sexo y en el eje y la creatina

Por otro lado, respecto a Kaplan-Meier se ven diferencias significativas entre las curvas de supervivencia de las variables: *CF,ISQUEMIC, INFART, FA, TV y MODE.*


**Figura 12:** Variable indicadora de la clase funcional del paciente

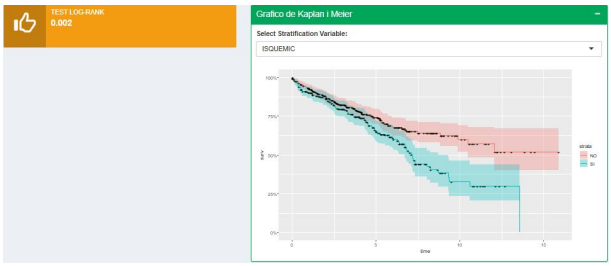


**Figura 13:** Variable indicadora de si el individuo ha tenido isquemia

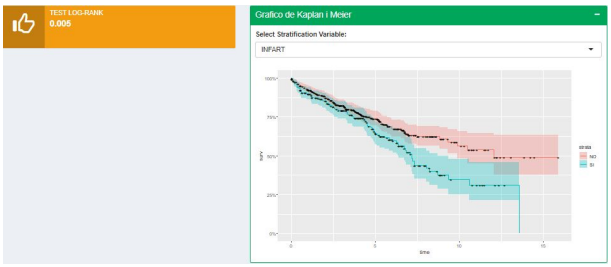


**Figura 14:** Variable indicadora de si el individuo ha tenido un infarto

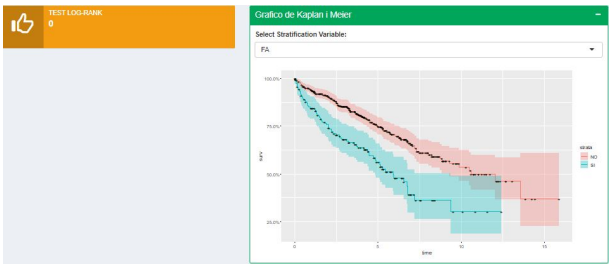


**Figura 15:** Variable indicadora de si el individuo ha tenido fibrilación auricular

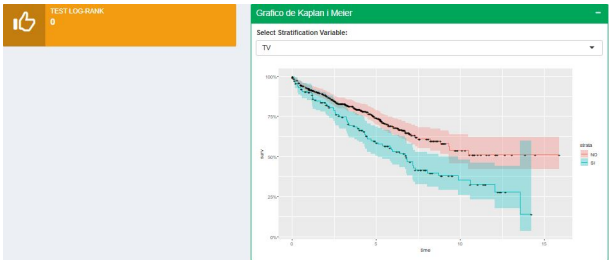


**Figura 16:** variable indicadora de la presencia de taquicardias en la parte superior.


**Figura 17:** Variable indicadora de si la TRC se ha hecho sola o con desfibrilador

Sobre las variables numéricas, las cuales se pueden categorizar, se ven curvas significativamente diferentes entre sus categorías para los siguientes valores:

- Edad: Si se categoriza la edad en el punto 53 se ven curvas significativamente diferentes.
- Creatina: Casi todos los valores de corte hacen que las curvas presenten diferencias significativas.
- FE: Si se hacen puntos de corte desde el valor 19 hasta el 34 saldrían diferencias significativas.
- FGR: Si hacemos puntos de corte del valor 23 al 110 también salen diferencias significativas.
- QRS: Lo mismo pasa para los valores de QRS que oscilan alrededor del valor 125.

Observando ahora el modelo de Cox, se ha decidido añadir las variables *Edat, FE, Creatina, FA y TV*, ya que son las variables que son significativas dentro del modelo, y presenta el supuesto de riesgos proporcionales.

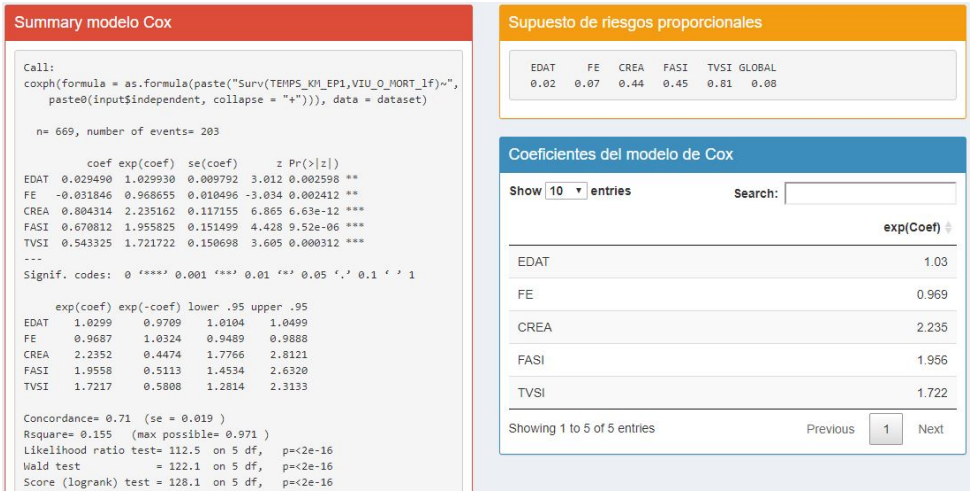


**Figura 18:** Modelo de Cox

Si nos fijamos en los valores de *FA* y de *TV* de la Figura 18, si la categoría de estas variables es sí, aumenta en un 95 % y en un 72 % respectivamente, el riesgo de morir respecto a la categoría de no. Por otro lado, si los valores de la creatina aumentan, se puede decir que hay casi más del doble de riesgo de morir respecto a valores inferiores de Creatina.

De este mismo modelo que se ha definido, se han comprobado los residuos, en este caso se muestran en la Figura 19 y 20:

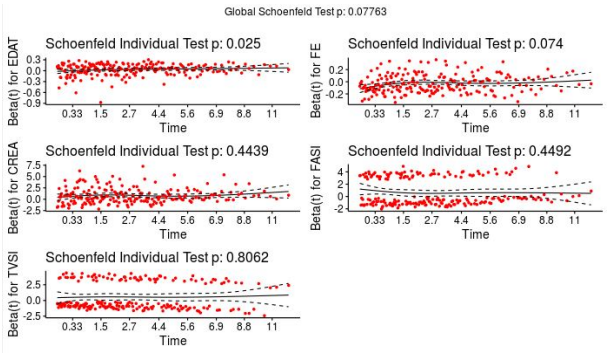


**Figura 19:** Gráfico de los residuos de Schoenfeld

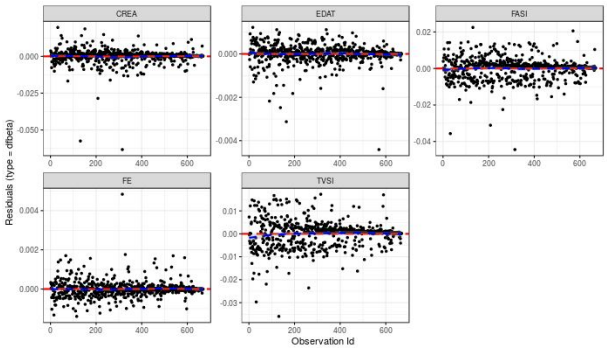


**Figura 20:** Gráfico de residuos Beta

Por otro lado, también se muestra el gráfico del modelo de Cox escogido, se visualiza en la Figura 21

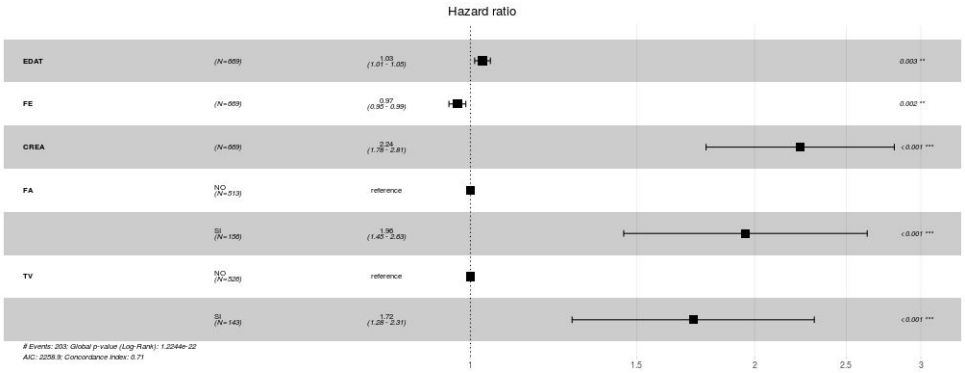


**Figura 21:** Gráfico del modelo de Cox

Por último, observando la Figura 22 sobre el estudio de las complicaciones:

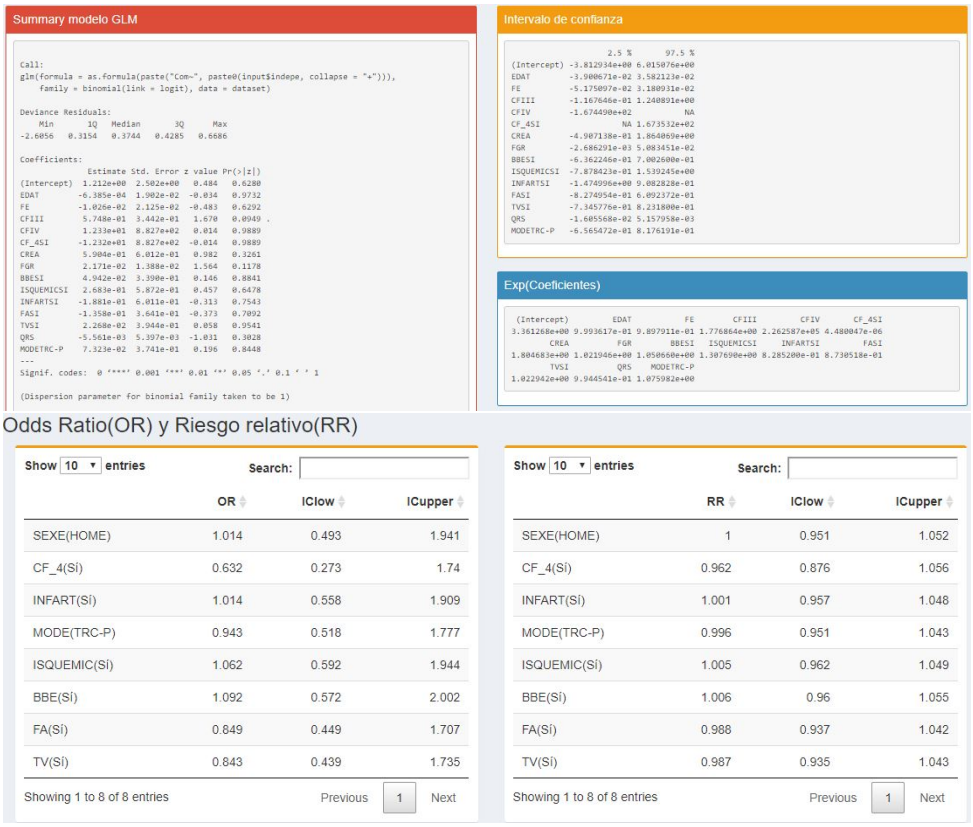


**Figura 22:** Análisis de las complicaciones

Después de hacer varios modelos se ha concluido que ninguna variable es significativa, esto se observa también en los intervalos de confianza ya que contienen el 0, y observando el AIC se ven variaciones poco significativas, pero hace falta remarcar que la clase funcional III sube un 77 % de posibilidades de morir respecto a los de clase funcional II.

Hace falta añadir que en un estudio de cohortes la interpretación de la OR es similar a la del RR, pero la OR tiende a alejarse más del valor nulo (1) tanto para valores inferiores a 1 (serán inferiores con la OR que con el RR) como para valores superiores a 1 (serán superiores con la OR que con

el RR). Sólo cuando el riesgo global de la enfermedad sea bajo (<10 %) la diferencia entre RR y OR puede considerarse insignificante, pero si el riesgo absoluto de la enfermedad es alto, la OR puede suponer una gran exageración del RR. Seguidamente si observamos los OR se podría decir que tanto la variable *FA* como la *TV* si están en la categoría sí, esto hace disminuir 0.16 veces la posibilidad de sufrir complicaciones respecto a los pacientes con la categoría no. Por lo que respecta al RR, se ve una disminución de la probabilidad del 2 % en los pacientes que presentan taquicardia y fibrilación auricular respecto a los que no.

# 5. Conclusiones

## 5.1. Conclusiones clínicas

Respecto al modelo de Cox seleccionado, hace falta remarcar que se ve un aumento del riesgo de morir casi del doble en aquellas personas que presentan fibrilación auricular, respecto a aquellas que no la presentan, y un aumento del riesgo del 72 % en aquellos pacientes que han sufrido taquicardias respecto a los que no las han tenido. Se observa que para cada valor más elevado de creatina aparece aproximadamente 2 veces más las posibilidades de morir que en los pacientes que tienen valores más bajos.

En segundo lugar, sobre el análisis de las complicaciones se observa que ninguna de las variables es significativa dentro del modelo, ya que todos los intervalos de confianza contienen el 1. Además, el AIC varía mínimamente entre los diferentes modelos, quitando o añadiendo variables. Mirando los coeficientes se aprecia un aumento en las posibilidades de sufrir complicaciones del 30 % en los individuos que presentan isquemia respecto a los que no tienen y un aumento del 77 % de las posibilidades de tener complicaciones en aquellos individuos que están en clase funcional III respecto a los de clase funcional II.

Por último, en un estudio de cohortes la interpretación de la OR es similar a la del RR, pero la OR tiende a alejarse más del valor nulo (1) tanto para valores inferiores a 1 (serán inferiores con la OR que con el RR) como para valores superiores a 1 (serán superiores con la OR que con el RR). Seguidamente si observamos los OR se podría decir que tanto la variable *FA* como la *TV* si están en la categoría sí, esto hace disminuir 0.16 veces la posibilidad de sufrir complicaciones respecto a los pacientes con la categoría no. Por lo que respecta al RR, se ve una disminución de la probabilidad del 2 % en los pacientes que presentan taquicardia y fibrilación auricular respecto a los que no.

## 5.2. Conclusiones metodológicas

En primer lugar, focalizándome en la librería *Shiny* he encontrado las siguientes ventajas y desventajas:

**Ventajas:**

- Interactividad y reactividad.
- Presenta una gran ayuda para hacer un amplio conjunto de conclusiones de un estudio estadístico sin la necesidad de ir ejecutando el código de cada caso en concreto.
- Tanto de cara a un cliente como a un trabajo personal, lo veo una buena forma de mostrar características de los datos de manera más entretenida y dinámica.

**Desventajas:**

- Cuando llega el momento de ejecutar y aparece un error es bastante difícil encontrar en que parte del código se encuentra, ya que Shiny únicamente te permite ejecutar código completo, por lo que pierde la facultad que presentaba R de poder ir ejecutando línea a línea para focalizar el error.
- Veo un inconveniente, la dificultad que presenta subir la aplicación web a internet, ya que si se quiere hacer de manera rápida y sencilla se tiene que utilizar el portal web de *shinyapp.io* el cual te deja acceder a él durante un límite de 25 horas mensuales, en el caso de que se quiera disponer de más tiempo se tiene que contratar una tarifa de pago.

Por lo que respecta al tema de programación, una de las dificultades a las que me he tenido que enfrentar ha sido crear la reactividad entre el panel lateral y el cuerpo de la web. Quiero añadir que una vez entendido no creo que sea algo difícil, pero considero que sí que es de los conceptos más

importantes que hay que tener en cuenta cuando se crea la primera aplicación web.

Por otro lado, he tenido que dedicar bastante tiempo en entender como referirme a una variable de una base de datos en concreto. Una de las maneras que R tiene para conseguir esto es utilizando el signo del dólar, en cambio, aun estando en R, la librería Shiny obliga a substituir este símbolo por los dobles corchetes o las dos exclamaciones antes de introducir el input, debido a que la variable es reactiva ya que el usuario puede escoger la que le interesa y la base de datos puede ser filtrada, esto también hace que esta sea reactiva.

Por último, y no menos importante, me costó encontrar la manera de poder compartir mi página web al público. Lo cierto es que existen tres maneras de hacerlo, pero dos de ellas eran necesario dinero y conocimientos elevados sobre programación y servidores, cosa que yo desconocía por completo. Y por último encontré una manera fácil de subirla que no suponía ningún coste económico desde la web *shinyapp.io*, el único inconveniente es que te limita el tiempo a 25 horas mensuales.

Todas estas dificultades se han ido presentando a lo largo del desarrollo de este proyecto, pero todas han sido solucionadas. Sin embargo, el único inconveniente que no he sabido resolver hoy en día ha sido hacer un gráfico con el paquete plotly donde la variable de x o y tenga más de una categoría.

### 5.3. Perspectivas futuras

Con la mirada puesta en el futuro, se podrían considerar cuatro aspectos que no se han llevado a cabo, estos son los siguientes:

- Hacer la página web en tres idiomas y de esta manera poder seleccionar el idioma en el que se quiere visualizar el aplicativo.
- Acceder con un usuario y contraseña por tal de poder restringir el acceso, por si hay información confidencial.
- Poner una imagen en la barra superior, como por ejemplo, el logotipo de la empresa.
- Conseguir hacer un gráfico plotly bivariante con variables con más de dos categorías.

## 6. Referencias

[1] https://www.revespcardiol.org/es-resincronizacion-cardiaca-\
insuficiencia-cardiaca-bases-articulo-13064194

[2] https://www.revespcardiol.org/es-terapia-resincronizacion\
-cardiaca-indicaciones-contraindicaciones-articulo-S0300893212002527

[3] https://rua.ua.es/dspace/bitstream/10045/54325/1/shiny.pdf

[4] https://www.elsevier.es/es-revista-cirugia-cardiovascular-\
358-articulo-terapia-resincronizacion-insuficiencia-cardiaca-S1134009611700675

[5] https://docs.rstudio.com/shinyapps.io/

[6] https://rstudio.github.io/shiny/tutorial/#welcome

[7] https://www.r-bloggers.com/lang/spanish/2818

# 7. Anexo

## 7.1. Tratamiento de la base de datos

```
library(shiny);library(foreign);library(forestplot);library(grid);library(foreign)
library(survival);library(KMsurv);library(ggplot2);library(survminer);
 library(shinydashboard);library(shinydashboardPlus);library(flexdashboard);
 library(plotly);library(stringr);library(DT);library(crosstalk);
 library(quantmod);library(openxlsx);library(lubridate)#paquete para fechas;
library(rlang);library(purrr);library(magrittr);library(dplyr);library(ggfortify);
library(nortest);library(epitools)


##Función que te devuelve TRUE si está vacío, NULL o NA
IsNullOrEmpty <- function(s) {
  return(is.null(s) || is.na(s) || s == "" || rlang::is_empty(s))
}

AnyNullOrEmpty <- function(...) {
  list(...) %>% some(IsNullOrEmpty)
}

NoneEmpty <- function(...) {
  !AnyNullOrEmpty(...)
}

###################
##Carga de datos##
###################

dat <- read.spss("TRC_TFG.sav", to.data.frame = TRUE)
dat <- as.data.frame(dat)

##Eliminamos columnas que no necesitamos##
dat <- dat[,-c(1,20,21,24,26:35)]

##0=alive,1=dead##
##Se define la variable de forma binaria para que fincione la función Surv##
dat$VIU_O_MORT_lf <- ifelse(dat$VIU_O_MORT_lf=="VIU",0,1)

##Validación de la base de datos##
suma_de_NA <- sum(is.na(dat))
#Se observa que hay 1 NA por lo que al solo haber 1, podemos omitirlo
dat <- na.omit(dat)

dataset <- as.data.frame(dat)

##Categorizamos la edad##
dataset$Edat_Cat[dataset$EDAT<62]<- "[21,62)"
dataset$Edat_Cat[dataset$EDAT>=62 & dataset$EDAT<68]<- "[62,68)"
dataset$Edat_Cat[dataset$EDAT>=68 & dataset$EDAT<74]<- "[68,74)"
dataset$Edat_Cat[dataset$EDAT>=74]<- "[74,86]"
dataset$Edat_Cat <- as.factor(dataset$Edat_Cat)

##Se definen las varibales numéricas que categorizará el usuario##
```

```
dataset$crea_Cat <- rep(0,nrow(dataset))
dataset$fe_Cat<- rep(0,nrow(dataset))
dataset$fgr_Cat<- rep(0,nrow(dataset))
dataset$qrs_Cat<- rep(0,nrow(dataset))


###Definir la clase de cada variable###
#Variables factor:#
dataset$SEXE <- as.factor(dataset$SEXE)
dataset$CF  <- as.factor(dataset$CF)
dataset$CF_4  <- as.factor(dataset$CF_4)
dataset$BBE  <- as.factor(dataset$BBE)
dataset$ISQUEMIC  <- as.factor(dataset$ISQUEMIC)
dataset$INFART  <- as.factor(dataset$INFART)
dataset$FA  <- as.factor(dataset$FA)
dataset$TV  <- as.factor(dataset$TV)
dataset$MODE  <- as.factor(dataset$MODE)
dataset$COMPLICACIONS  <- as.factor(dataset$COMPLICACIONS)
dataset$COMPLICACIONS_CAT  <- as.factor(dataset$COMPLICACIONS_CAT)
dataset$SUCCES_IMPLANT  <- as.factor(dataset$SUCCES_IMPLANT)
dataset$Edat_Cat <-as.factor(dataset$Edat_Cat)
dataset$crea_Cat <-as.factor(dataset$crea_Cat)
dataset$fe_Cat<- as.factor(dataset$fe_Cat)
dataset$fgr_Cat<-as.factor(dataset$fgr_Cat)
dataset$qrs_Cat<-as.factor(dataset$qrs_Cat)
dataset$END_POINT_1 <- as.factor(dataset$END_POINT_1)
dataset$TRANSPLANT_lf <- as.factor(dataset$TRANSPLANT_lf)

##Variables numéricas##
dataset$EDAT <- as.numeric(dataset$EDAT)
dataset$FE <- as.numeric(dataset$FE)
dataset$CREA <- as.numeric(dataset$CREA)
dataset$FGR <- as.numeric(dataset$FGR)
dataset$QRS <- as.numeric(dataset$QRS)

##Se recodifica la variable COMPLICACIONS en si o no, para un futuro análisis##
dataset$Com <- ifelse(dataset$COMPLICACIONS=="NO","Sense complicacions",
                      "Amb complicacions")
dataset$Com <- as.factor(dataset$Com)
```

## 7.2. Parte ui

```
###################
########UI########
###################

ui <- dashboardPagePlus(skin = "green",
        header = dashboardHeaderPlus(
        enable_rightsidebar = TRUE,
        rightSidebarIcon = "gears"
      ),
  dashboardSidebar(
    sidebarMenu(##Barra lateral izquierda
      menuItem("Base de datos", tabName = "Basededatos", icon=icon("database")),
```

```
      menuItem("Análisis descriptivo", tabName = "Descriptivos",
               icon=icon("chart-line"),
               menuSubItem("Descriptiva univariante", tabName="uni"),
               menuSubItem("Descriptiva bivariante", tabName="biv"),
               menuSubItem("Kaplan i Meier",  tabName ="KM1"),
               menuSubItem("Curva de supervivencia", tabName="Sup")),
      menuItem("Modelo de Cox", tabName = "Cox",
              menuSubItem("Selección de variables", tabName="SV"),
              menuSubItem("Validación de supuestos",  tabName ="VS"),
              menuSubItem("Grafico del modelo Cox", tabName="Plot_Cox")),

      menuItem("Analisis de las complicaciones", tabName = "AnalisisC"),hr(),
      sliderInput("edat",
                  "Edat_Cat",
                  min = min(dataset$EDAT),
                  max = max(dataset$EDAT),
                  value = 42),
      sliderInput("crea",
                  "crea_Cat",
                  min = min(dataset$CREA),
                  max = max(dataset$CREA),
                  value = 1),
      sliderInput("fe",
                  "fe_Cat",
                  min = min(dataset$FE),
                  max = max(dataset$FE),
                  value = 25),
      sliderInput("fgr",
                  "fgr_Cat",
                  min = min(dataset$FGR),
                  max = max(dataset$FGR),
                  value = 198),
      sliderInput("qrs",
                  "qrs_Cat",
                  min = min(dataset$QRS),
                  max = max(dataset$QRS),
                  value = 206))),
    dashboardBody(##Cuerpo de la página
    tabItems(
      tabItem(tabName = "Basededatos", mainPanel(
        tabsetPanel(type = "tabs",
tabPanel("Contextualizacion",hr(),
p("Se analiza una cohorte de pacientes en el ámbito
de la cardiología. Concretamente, se estudiará una cohorte de pacientes con una
terapia de Resincronización Cardiaca (TRC). Los pacientes de resincronización
cardiaca, son una tipología de pacientes que tienen una mala
sincronización en el funcionamiento de las cavidades del corazón. Un
dispositivo de resincronización cardiaca es un dispositivo que tiene como
objetivo volver a sincronizar las contracciones de los ventrículos del corazón
mediante estimulaciones eléctricas artificiales. Se
estudiará diferentes aspectos que son de interés en la práctica clínica
diaria y se hará un balance de la experiencia de nuestro centro al largo de
los años. Por otro lado, se dispondrá de una base de datos con 671 pacientes
operados entre el año 1999 y 2015 en la unidad de
arritmias del Hospital Clínico de Barcelona. Por lo que respecta a las técnicas
```

```
utilizadas, se centrará en un análisis de supervivencia, modelos lineales
entre otros.")),
tabPanel("Tabla de datos", icon = icon("table"),dataTableOutput("table1"),
         dataTableOutput("table2")),
tabPanel("Variables", hr(), h2(strong("Lista de variables")),
         p(strong("SEXE:"),"(HOME, DONA) Sexo del individuo."),
p(strong("EDAT"),": Edad del individuo."),
p(strong("FE:")," Es una variable numérica, que indica la fracción de
eyección, es decir, es una medida del porcentaje de sangre que sale
del corazón cada vez que se contrae."),
p(strong("CF:")," Indica la clase funcional, en otras palabras, es la capacidad de
  movilidad que tiene la persona."),
p(strong("CREA:")," Creatinina. La insuficiencia renal es un importante
factor pronóstico
  en pacientes con insuficiencia cardiaca. Para valorar la función renal
se suelen
utilizar las cifras de creatinina."),
p(strong("FGR:")," Es el filtrado glomerular, esto es el volumen de
fluido filtrado
por unidad de
  tiempo desde los capilares glomerulares renales hacia el interior de
la cápsula de Bowman.
  Normalmente se mide en mililitros por minuto."),
p(strong("BBE:")," Bloqueo de rama izquierda."),
p(strong("ISQUEMIC:")," Si el individuo presenta isquemia cardiaca."),
p(strong("INFART:")," Si ha tenido un infarto."),
p(strong("FA:")," Fibrilación auricular."),
p(strong("TV:")," Si ha presentado una taquicardia en la parte superior."),
p(strong("QRS:")," El complejo QRS es la representación gráfica de la
despolarización de los
  ventrículos del corazón formando una estructura picuda en el
electrocardiograma."),
p(strong("MODE:")," Indica si la TRC se hace sola (TRC-P) o la TRC se le añade
  desfibrilador (TRC-D)."),
p(strong("COMPLICACIONS:")," Existencia de complicaciones en el
  proceso del tratamiento."),
p(strong("VIU_0_MORT_lf:")," Indicador de si ha pasado el evento de interés."),
p(strong("TEMPS_KM_EP1:")," Tiempo que pasa dentro del estudio."))))),
      tabItem(tabName = "KM1", fluidRow(valueBoxOutput("pval"),
                                        box( solidHeader = TRUE,
                                             title="Grafico de Kaplan i Meier",
            collapsible = TRUE,status = "success",selectInput(inputId = "s",
            label = "Select Stratification Variable:",
            choices = c(names(dataset)),
            selected = "SEXE"),
            plotOutput(outputId = "km")))),
      tabItem(tabName = "uni", box(status = "primary", solidHeader = TRUE,
                                   title="Descriptiva univariante",
            collapsible = TRUE,selectInput(inputId = "n", label =
                                           "Seleccione una variable:",
            choices = c(names(dataset)),selected = "SEXE"),
            plotlyOutput(outputId = "univ")),box(status = "warning",
                                                 solidHeader = TRUE,
                                                 title="Estadísticos principales",
            dataTableOutput("summary")),
```

```
            fluidRow(
              box(status="danger",
                width = 12,
                title = "Click or select on the plot to fill the table!",
                plotOutput('seleccionable',
                           click = "user_click",
                           brush = "user_brush"),
                dataTableOutput("tabla_seleccionable")))),

    tabItem(tabName = "biv",h2("Análisis bivariante"),
            fluidRow(column(6,
                  selectInput(inputId = "x",
                              label = "Selecciona una variable para el eje x:",
                  choices = names(dataset),selected = "SEXE")),
                  column(6,
                  selectInput(inputId = "y",
                              label = "Selecciona una variable para el eje y:",
                  choices = names(dataset),selected = "EDAT"))),
                  fluidRow(plotlyOutput("plot_P")),hr(),
            box(status="primary",title="Variable X",
                verbatimTextOutput("lillie_X")),
            box(status="primary",title="Variable Y",
                verbatimTextOutput("lillie_Y")),

            fluidRow(
              box(solidHeader = TRUE,status="primary",
                  title="Tabla de contingencia",
                  verbatimTextOutput("tabla_factor")),infoBoxOutput("corre")),
            fluidRow(
              box(status="danger",
                  width  = 12,
                  title = "Click or select on the plot to fill the table!",
                  plotOutput('seleccionable2',
                             click = "user_click",
                             brush = "user_brush"),
                  dataTableOutput("tabla_seleccionable2")))),

    tabItem(tabName = "Sup",h4("Curva de supervivencia"),
            plotOutput(outputId = "plot_sup")),

    tabItem(tabName = "AnalisisC",uiOutput("indepe"),
            fluidRow(
            box(status="danger",solidHeader = TRUE,
                title="Summary modelo GLM",verbatimTextOutput("summary_glm")),
            box(status="warning", solidHeader = TRUE,
                title="Intervalo de confianza",verbatimTextOutput("confint_glm")),
            box(status="primary",solidHeader = TRUE,
                title="Exp(Coeficientes)",verbatimTextOutput("exp_glm"))),
            fluidRow(h3("Odds Ratio(OR) y Riesgo relativo(RR)"),
                     box(status="warning",dataTableOutput("OR")),
                     box(status="warning",dataTableOutput("RR")))),
    tabItem(tabName = "SV",
            uiOutput("independent"),
            fluidRow(box(status="danger",solidHeader = TRUE,
                         title="Summary modelo Cox",
```

```
                          verbatimTextOutput("sum1")),
                box(status="warning", solidHeader = TRUE,
                    title="Supuesto de riesgos proporcionales",
                    verbatimTextOutput("RP")),
                box(status="primary",solidHeader = TRUE,
                    title="Coeficientes del modelo de Cox",
                    dataTableOutput("tabla_cox")))
                ),
      tabItem(tabName="Plot_Cox", plotOutput("grafico_cox")),

      tabItem(tabName = "VS", mainPanel(
        tabsetPanel(type = "tabs",#Residuos del modelo de Cox
                    tabPanel("Residuos Schoenfeld",hr(),
                             plotOutput("cox_resi")),
                    tabPanel("Linealidad", icon = icon("table"),hr(),
                             plotOutput("plot_l")),
                    tabPanel("Residuos beta", hr(),plotOutput("plot_beta")))
        ))),
    rightsidebar= rightSidebar(#Barra lateral derecha
        background = "dark",
        rightSidebarTabContent(
          id = "right_sidebar_tab_1",
          icon = "filter",
          active = TRUE,
          uiOutput("right_menu"))))
```

### 7.3. Parte server

```
###################
######Server######
###################

server <- function(input, output, session){
  #Se crea el filtro de la base de datos
  dim_SEXE <- dataset %>% distinct(SEXE) %>% as.data.frame() %>% c("All", .)
  dim_EDAT   <- dataset %>% distinct(EDAT) %>% as.data.frame() %>% c("All", .)
  dim_FE   <- dataset %>% distinct(FE) %>% as.data.frame() %>% c("All", .)
  dim_CF <- dataset %>% distinct(CF) %>% as.data.frame() %>% c("All", .)
  dim_CF_4 <- dataset %>% distinct(CF_4) %>% as.data.frame() %>% c("All", .)
  dim_CREA <- dataset %>% distinct(CREA) %>% as.data.frame() %>% c("All", .)
  dim_FGR <- dataset %>% distinct(FGR) %>% as.data.frame() %>% c("All", .)
  dim_BBE <- dataset %>% distinct(BBE) %>% as.data.frame() %>% c("All", .)
  dim_ISQUEMIC <- dataset %>% distinct(ISQUEMIC) %>% as.data.frame() %>% c("All", .)
  dim_INFART <- dataset %>% distinct(INFART) %>% as.data.frame() %>% c("All", .)
  dim_FA <- dataset %>% distinct(FA) %>% as.data.frame() %>% c("All", .)
  dim_TV <- dataset %>% distinct(TV) %>% as.data.frame() %>% c("All", .)
  dim_QRS <- dataset %>% distinct(QRS) %>% as.data.frame() %>% c("All", .)

  selected_CREA <- reactive({
    ifelse(is.null(input$CREA), "All", input$CREA)
  })

  selected_FGR <- reactive({
```

```
  ifelse(is.null(input$FGR), "All", input$FGR)
})

selected_BBE <- reactive({
  ifelse(is.null(input$BBE), "All", input$BBE)
})

selected_ISQUEMIC <- reactive({
  ifelse(is.null(input$ISQUEMIC), "All", input$ISQUEMIC)
})

selected_INFART <- reactive({
  ifelse(is.null(input$INFART), "All", input$INFART)
})

selected_FA <- reactive({
  ifelse(is.null(input$FA), "All", input$FA)
})

selected_TV <- reactive({
  ifelse(is.null(input$TV), "All", input$TV)
})

selected_QRS <- reactive({
  ifelse(is.null(input$QRS), "All", input$QRS)
})

selected_SEXE <- reactive({
  ifelse(is.null(input$SEXE), "All", input$SEXE)
})

selected_EDAT <- reactive({
  ifelse(is.null(input$EDAT), "All", input$EDAT)
})

selected_FE <- reactive({
  ifelse(is.null(input$FE), "All", input$FE)
})

selected_CF <- reactive({
  ifelse(is.null(input$CF), "All", input$CF)
})

selected_CF_4 <- reactive({
  ifelse(is.null(input$CF_4), "All", input$CF_4)
})

bajas<- reactive({

  dataset$Edat_Cat <- ifelse(dataset$EDAT < input$edat,
     paste0("Edat<",input$edat), paste0("Edat>=",input$edat))
  dataset$Edat_Cat <- as.factor(dataset$Edat_Cat)

  dataset$crea_Cat <- ifelse(dataset$CREA < input$crea,
     paste0("Crea<",input$crea), paste0("crea>=",input$crea))
```

```
    dataset$crea_Cat <- as.factor(dataset$crea_Cat)

    dataset$fe_Cat <- ifelse(dataset$FE < input$fe,
      paste0("FE<",input$fe), paste0("FE>=",input$fe))
    dataset$fe_Cat <- as.factor(dataset$fe_Cat)

    dataset$fgr_Cat <- ifelse(dataset$FGR < input$fgr,
      paste0("FGR <",input$fgr), paste0("FGR>=",input$fgr))
    dataset$fgr_Cat <- as.factor(dataset$fgr_Cat)

    dataset$qrs_Cat <- ifelse(dataset$QRS < input$qrs,
      paste0("QRS <",input$qrs), paste0("QRS>=",input$qrs))
    dataset$qrs_Cat <- as.factor(dataset$qrs_Cat)

    req(NoneEmpty(input$SEXE, input$EDAT, input$FE,
                  input$CF,input$CF_4, input$CREA, input$FGR,
                  input$BBE, input$ISQUEMIC,
                  input$INFART, input$FA, input$TV, input$QRS))
    data <- dataset

    if (selected_SEXE() != "All") {
      data %<>% filter(SEXE == selected_SEXE())
    }

    if (selected_EDAT() != "All") {
      data %<>% filter(EDAT == selected_EDAT())
    }

    if (selected_FE() != "All") {
      data %<>% filter(FE == selected_FE())
    }

    if (selected_CF() != "All") {
      data %<>% filter(CF == selected_CF())
    }

    if (selected_CF_4() != "All") {
      data %<>% filter(CF_4 == selected_CF_4())
    }

    if (selected_CREA() != "All") {
      data %<>% filter(CREA == selected_CREA())
    }

    if (selected_FGR() != "All") {
      data %<>% filter(FGR == selected_FGR())
    }

    if (selected_BBE() != "All") {
      data %<>% filter(BBE == selected_BBE())
    }

    if (selected_ISQUEMIC() != "All") {
      data %<>% filter(ISQUEMIC == selected_ISQUEMIC())
    }
```

```
  if (selected_INFART() != "All") {
    data %<>% filter(INFART == selected_INFART())
  }

  if (selected_FA() != "All") {
    data %<>% filter( FA== selected_FA())
  }

  if (selected_TV() != "All") {
    data %<>% filter(TV == selected_TV())
  }

  if (selected_QRS() != "All") {
    data %<>% filter(QRS == selected_QRS())
  }

  return(data)
})


## Right sidebar
output$right_menu <- renderMenu({
  tagList(
    selectInput(
      "SEXE",
      "SEXE",#titulo
      dim_SEXE,
      selected = isolate(selected_SEXE())
    ),
    selectInput(
      "EDAT",
      "EDAT",#titulo
      dim_EDAT,
      selected = isolate(selected_EDAT())
    ),
    selectInput(
      "FE",
      "FE",
      dim_FE,
      selected = isolate(selected_FE())
    ),
    selectInput(
      "CF",
      "CF",
      dim_CF,
      selected = isolate(selected_CF())
    ),
    selectInput(
      "CF_4",
      "CF_4",
      dim_CF_4,
      selected = isolate(selected_CF_4())
    ),
    selectInput(
      "CREA",
```

```
        "CREA",
        dim_CREA,
        selected = isolate(selected_CREA())
      ),
      selectInput(
        "FGR",
        "FGR",
        dim_FGR,
        selected = isolate(selected_FGR())
      ),
      selectInput(
        "BBE",
        "BBE",
        dim_BBE,
        selected = isolate(selected_BBE())
      ),
      selectInput(
        "ISQUEMIC",
        "ISQUEMIC",
        dim_ISQUEMIC,
        selected = isolate(selected_ISQUEMIC())
      ),
      selectInput(
        "INFART",
        "INFART",
        dim_INFART,
        selected = isolate(selected_INFART())
      ),
      selectInput(
        "FA",
        "FA",
        dim_FA,
        selected = isolate(selected_FA())
      ),
      selectInput(
        "TV",
        "TV",
        dim_TV,
        selected = isolate(selected_TV())
      ),
      selectInput(
        "QRS",
        "QRS",
        dim_QRS,
        selected = isolate(selected_QRS())
      )
    )

})

##Código para la tabla de la que al seleccionar sale la tabla correspondiente
interaction_type <- "click"
observeEvent(input$user_click, interaction_type «- "click")
observeEvent(input$user_brush, interaction_type «- "brush")
```

```
output$seleccionable <- renderPlot({
  variable1 <- input$s
  B <- bajas()
  B %>%
    ggplot(aes(!!as.name(input$n), VIU_O_MORT_lf)) +
    geom_jitter(width = 0.25, aes(colour =  VIU_O_MORT_lf)) +
    theme(legend.position = "none") +
    theme(
      axis.text.x = element_text(angle = 45, hjust = 1,
                                 size=14, face = "bold"),
      axis.text.y = element_text(size=14, face = "bold")) +
    xlab("Grupo tarifa")
})


brush_data <- reactive({
  B <- bajas()
  user_brush <- input$user_brush
  user_click <- input$user_click
  if (interaction_type == "brush") {
    brushedPoints(B, user_brush)
  } else if (interaction_type == "click") {
    nearPoints(B, user_click, threshold = 10)
  }
})

output$tabla_seleccionable <- renderDT({
  df <- brush_data()
  req(nrow(df) > 0)
  df %>% datatable(
    rownames = FALSE,
    colnames = colnames(dataset),
    selection = "single",
    extensions = "Buttons",
    filter = "top",
    options = list(
      dom = "Blfrtip",
      scrollX = TRUE
      ,buttons = list("copy", list(
        extend = "collection",
        buttons = c("csv", "excel", "pdf"),
        text = "Download"))))
})


variable <- reactive({
  variable1 <- input$n
    })

output$summary <- renderDataTable({
  B <- bajas()
  variable1 <- input$n
  if(class(B[[variable1]])=="factor"){
  a <- B[[variable1]] %>% table()
  b <- B[[variable1]] %>% table() %>% prop.table()
```

```
  b <- round(b,2)
  c <- as.data.frame(t(rbind(a,b)))
  colnames(c) <- c("N","%")

  return(c)
  }else{
    dataset
    s_dat <- as.matrix(summary(B[[variable1]]))
    s_dat <- round(as.data.frame(t(s_dat)),2)
    return(s_dat)
  }
})


output$cox_resi <- renderPlot({
  mod0 <- runRegression()
  riesgo0 <- cox.zph(mod0)
  #Riesgos proporcionales
  ggcoxzph(riesgo0)#Schoenfeld
})

output$plot_l <- renderPlot({
  mod0 <- runRegression()
 #Linealidad
 ggcoxdiagnostics(mod0, type = "schoenfeld", ox.scale = "time")
})

output$plot_beta <- renderPlot({
  mod0 <- runRegression()
  #Residuales Dfbeta
  ggcoxdiagnostics(mod0, type = "dfbeta")
})

output$univ <- renderPlotly({
  B <- bajas()
  variable1 <- variable()

  if(class(B[[variable1]])=="factor"){
    B <- bajas()

    B %>%
      group_by(!!as.name(variable1)) %>%
      summarise(models=n()) %>%
      ungroup %>%
      plot_ly() %>%

      add_bars(
        x = as.formula(paste0("~`", variable1, "`")),
        y = ~models,
        name = "2019",
        yaxis = "y2",
        marker = list(
          color = "rgba(226, 0, 116, 0.2)",
          line = list(color = "rgb(226, 0, 116, 0.1)", width = 0.9)))
    }else{
```

```
      plot_ly(B, x = B[[variable1]], type="box",mode = 'markers') }

})

 output$table1 <- renderDataTable({
   B <- bajas()
  head(B[,1:12],4)
 })

 output$table2 <- renderDataTable({
   B <- bajas()
   head(B[,c(13:15,18:21)],4)
 })

 output$km <- renderPlot({
   B <- bajas()
   kmdata <-  surv_fit(as.formula(paste('Surv(TEMPS_KM_EP1,VIU_O_MORT_lf)~',
                                        input$s)), data=B)
   autoplot(kmdata)
 })

 output$pval <- renderValueBox({
   B <- bajas()
   sdf <- survdiff(as.formula(paste('Surv(TEMPS_KM_EP1,VIU_O_MORT_lf)~',
                                    input$s)), data=B,rho=0)

   infoBox(
     "Test Log-Rank",round(1-pchisq(sdf$chisq,length(sdf$n)-1),3),
     icon = icon("thumbs-up", lib = "glyphicon"),
     color = "yellow", fill = TRUE)
 })


 runSur <- reactive({
   B <- bajas()
   survfit(as.formula(paste("Surv(TEMPS_KM_EP1,VIU_O_MORT_lf) ~ ",
                            paste("1"))),data=B)
 })

 # Plot the survival graph
 output$plot_sup <- renderPlot({
   cosa <- runSur()
   autoplot(runSur(),xlab="Days", ylab="S(t)", surv.colour = 'orange')
 })

 output$independent <- renderUI({
   checkboxGroupInput("independent", "Independent Variables:",
                      choices =  names(dataset)[2:14])
 })

 runRegression <- reactive({
   B <- bajas()
   coxph(as.formula(paste("Surv(TEMPS_KM_EP1,VIU_O_MORT_lf)~",
    paste0(input$independent,collapse="+"))),data=dataset)
 })
```

```
output$sum1 <- renderPrint({
  if(!is.null(input$independent)){
    summary(runRegression())
  } else {
    print(data.frame(Warning="Debe seleccionar al menos 1 variable."))
  }
})


output$RP <- renderPrint({
  if(!is.null(input$independent)){
    mod0 <- runRegression()
    riesgo0 <- cox.zph(mod0)
    round(riesgo0$table[,3],2)
  } else {
    print(data.frame(Warning="Debe seleccionar al menos 1 variable."))
  }
})


output$tabla_cox <- renderDataTable({
  if(!is.null(input$independent)){
    mod0 <- runRegression()
    smod1 <- summary(mod0)
    coxcoe <- as.data.frame(round(smod1$coefficients[,2],3))
    colnames(coxcoe) <- c("exp(Coef)")
    coxcoe
  } else {
    print(data.frame(Warning="Debe seleccionar al menos 1 variable."))
  }

})


output$plot_P <- renderPlotly({
  B <- bajas()
  X <- input$x
  Y <- input$y
  if((class(B[[Y]])=="numeric") && (class(B[[X]])=="numeric")){
   p <-  plot_ly(B, x =~B[[X]], y=~B[[Y]])
    return(p)
  }else{
    if((class(B[[Y]])=="factor") && (class(B[[X]])=="factor")){
      pp_f <- B %>% ggplot(data = B,
      mapping=aes(x =!!as.name(X), fill =!!as.name(Y)))+
        geom_bar(position = "fill")
    return(ggplotly(pp_f))
    }else{
      return(plot_ly(B, x =~B[[X]], y=~B[[Y]],
                     type="box",mode = 'markers'))
    }}
    })
```

```
  output$lillie_X <- renderPrint({
    B <- bajas()
    X <- input$x
    Y <- input$y
    if(class(B[[X]])=="numeric"){
      return(lillie.test(B[[X]]))
    }else{return(paste0("Esta variable es categórica"))}
  })

  output$lillie_Y <- renderPrint({
    B <- bajas()
    X <- input$x
    Y <- input$y

    if(class(B[[Y]])=="numeric"){
      return(lillie.test(B[[Y]]))
    }else{return(paste0("Esta variable es categórica"))}

  })


  output$corre <- renderInfoBox({
    B <- bajas()
    X <- input$x
    Y <- input$y

    if((class(B[[X]])=="numeric") && (class(B[[Y]])=="numeric")){
      co<- infoBox(
        "Correlación",round(cor(B[[X]],B[[Y]],
    method = "spearman"),3), icon = icon("thumbs-up", lib = "glyphicon"),
        color = "yellow", fill = TRUE)
      #utilizo spearman porque no se distribuyen como una normal
      return(co)
    }else{
      if((class(B[[X]])=="factor") && (class(B[[Y]])=="factor")){
      co <- infoBox(
        "Grado de asociación", round(chisq.test(table(B[[X]],
    B[[Y]]))$p.value,3) , icon = icon("thumbs-up", lib = "glyphicon"),
        color = "red", fill = TRUE)
      return(co)}
    else{
      pv <- wilcox.test(as.formula(paste(input$y,"~",input$x)),
                        alternative="two.sided", data=B)
      pv<- round(pv$p.value,3)
      co <- infoBox(
      h6("Willcoxon Test"), paste(pv) , icon = icon("thumbs-up",
                                                    lib = "glyphicon"),
      color = "red", fill = TRUE)
      return(co)
      }}
})


  output$tabla_factor <- renderPrint({
    B <- bajas()
```

```
  X <- input$x
  Y <- input$y

 if(class(B[[X]])=="factor" && class(B[[Y]])=="factor"){
  prop <- table(B[[X]],B[[Y]])
  return(prop)
 }else{return(paste0("Alguna variable es numérica"))}

  })


output$seleccionable2 <- renderPlot({
  X <- input$x
  Y <- input$y
  B <- bajas()
  B %>%
    ggplot(aes(!!as.name(input$x), !!as.name(input$y))) +
    geom_jitter(width = 0.25, aes(colour =  !!as.name(input$y))) +
    theme(legend.position = "none") +
    theme(
      axis.text.x = element_text(angle = 45, hjust = 1,
                                 size=14, face = "bold"),
      axis.text.y = element_text(size=14, face = "bold"))
})


brush_data2 <- reactive({
  B <- bajas()
  user_brush <- input$user_brush
  user_click <- input$user_click
  if (interaction_type == "brush") {
    brushedPoints(B, user_brush)
  } else if (interaction_type == "click") {
    nearPoints(B, user_click, threshold = 10)
  }
})

output$tabla_seleccionable2 <- renderDT({
  df <- brush_data()
  req(nrow(df) > 0)
  df %>% datatable(
    rownames = FALSE,
    colnames = colnames(dataset),
    selection = "single",
    extensions = "Buttons",
    filter = "top",
    options = list(
      dom = "Blfrtip",
      scrollX = TRUE
      ,buttons = list("copy", list(
        extend = "collection",
        buttons = c("csv", "excel", "pdf"),
        text = "Download"
      ))
    )
```

```
  )
})

output$grafico_cox <- renderPlot({
  mod0 <- runRegression()
  ggforest(mod0, data=dataset)

})



output$indepe <- renderUI({
  checkboxGroupInput("indepe", "Independent Variables:",
                     choices =  names(dataset)[2:14])
})

runRegression_glm <- reactive({
  fit <- glm(as.formula(paste("Com~",paste0(input$indepe,
    collapse="+"))),data=dataset,family=binomial(link=logit))
})

output$summary_glm <- renderPrint({

  if(!is.null(input$independent)){
    fit <- runRegression_glm()
    summary(fit)
  } else {
    print(data.frame(Warning="Debe seleccionar al menos 1 variable."))
  }

})

output$confint_glm <- renderPrint({

  if(!is.null(input$independent)){
    fit <- runRegression_glm()
    confint(fit)
  } else {
    print(data.frame(Warning="Debe seleccionar al menos 1 variable."))
  }

})
# 95% CI for the coefficients
output$exp_glm <- renderPrint({

  if(!is.null(input$independent)){
    fit <- runRegression_glm()
    exp(coef(fit))
  } else {
    print(data.frame(Warning="Debe seleccionar al menos 1 variable."))
  }
})


output$OR <- renderDataTable({
```

```
    OR0 <- oddsratio(dataset$SEXE,dataset$Com)
    a <- OR0$measure[2,]
    OR1 <- oddsratio(dataset$CF_4,dataset$Com)
    b <- OR1$measure[2,]
    OR2 <- oddsratio(dataset$INFART,dataset$Com)
    c <- OR2$measure[2,]
    OR3 <- oddsratio(dataset$MODE,dataset$Com)
    d <- OR3$measure[2,]
    OR4 <- oddsratio(dataset$ISQUEMIC,dataset$Com)
    e <- OR4$measure[2,]
    OR5 <- oddsratio(dataset$BBE,dataset$Com)
    f <- OR5$measure[2,]
    OR6 <- oddsratio(dataset$FA,dataset$Com)
    g <- OR6$measure[2,]
    OR7 <- oddsratio(dataset$TV,dataset$Com)
    h <- OR7$measure[2,]

    OR <- rbind(a,b,c,d,e,f,g,h)
    rownames(OR) <- c("SEXE(HOME)","CF_4(Sí)","INFART(Sí)",
          "MODE(TRC-P)", "ISQUEMIC(Sí)","BBE(Sí)","FA(Sí)","TV(Sí)")
    colnames(OR) <- c("OR", "IClow", "ICupper")

    return(round(OR,3))
  })

  output$RR <- renderDataTable({
    OR0 <- riskratio(dataset$SEXE,dataset$Com)
    a <- OR0$measure[2,]
    OR1 <- riskratio(dataset$CF_4,dataset$Com)
    b <- OR1$measure[2,]
    OR2 <- riskratio(dataset$INFART,dataset$Com)
    c <- OR2$measure[2,]
    OR3 <- riskratio(dataset$MODE,dataset$Com)
    d <- OR3$measure[2,]
    OR4 <- riskratio(dataset$ISQUEMIC,dataset$Com)
    e <- OR4$measure[2,]
    OR5 <- riskratio(dataset$BBE,dataset$Com)
    f <- OR5$measure[2,]
    OR6 <- riskratio(dataset$FA,dataset$Com)
    g <- OR6$measure[2,]
    OR7 <- riskratio(dataset$TV,dataset$Com)
    h <- OR7$measure[2,]


    OR <- rbind(a,b,c,d,e,f,g,h)
    rownames(OR) <- c("SEXE(HOME)","CF_4(Sí)","INFART(Sí)",
      "MODE(TRC-P)", "ISQUEMIC(Sí)","BBE(Sí)","FA(Sí)","TV(Sí)")
    colnames(OR) <- c("RR", "IClow", "ICupper")

    return(round(OR,3))
  })
}
```

### 7.4. ShinyApp

```
shinyApp(ui, server)
```