Modelo predictivo hepatitis C

Jorge Ballesta Cerezo

Contexto inicial

La **hepatitis C** es una infección hepática causada por el virus de la hepatitis C (VHC). La hepatitis C se transmite a través del contacto con la sangre de una persona infectada. Hoy en día, la mayoría de las personas se infectan con el virus de la hepatitis C al compartir agujas u otro equipo utilizado para preparar e inyectar drogas. Para algunas personas, la hepatitis C es una enfermedad de corta duración, pero para más de la mitad de las personas que se infectan con el virus de la hepatitis C, se convierte en una infección crónica a largo plazo. La hepatitis C crónica puede resultar en problemas de salud graves e incluso mortales, como la cirrosis y el cáncer de hígado. Las personas con hepatitis C crónica a menudo no presentan síntomas y no se sienten enfermas. Cuando los síntomas aparecen, a menudo son una señal de enfermedad hepática avanzada. No existe vacuna para la hepatitis C. La mejor manera de prevenir la hepatitis C es evitando comportamientos que puedan propagar la enfermedad, especialmente el consumo de drogas inyectables. Hacerse la prueba de hepatitis C es importante, porque los tratamientos pueden curar a la mayoría de las personas con hepatitis C en 8 a 12 semanas.

La prevención de la hepatitis C puede ser mejorada mediante la creación de un modelo predictivo que pueda detectar de forma prematura dicha infección y otras fases más avanzadas y derivadas del virus.

Descripción del dataset

Para poder analizar las características y consecuencias sobre pacientes hepáticos, necesitamos obtener una muestra de datos sobre la que realizar un estudio en profundidad. Para ello hemos elegido un dataset que ha sido donado por un laboratorio alemán que ha realizado un gran trabajo de investigación y ha registrado análisis de sangre tanto de pacientes como de donantes que mide los niveles de ciertas encimas y de proteinas. Además, también han anotado la edad y sexo de cada muestra tomada.

https://archive.ics.uci.edu/ml/datasets/HCV+data

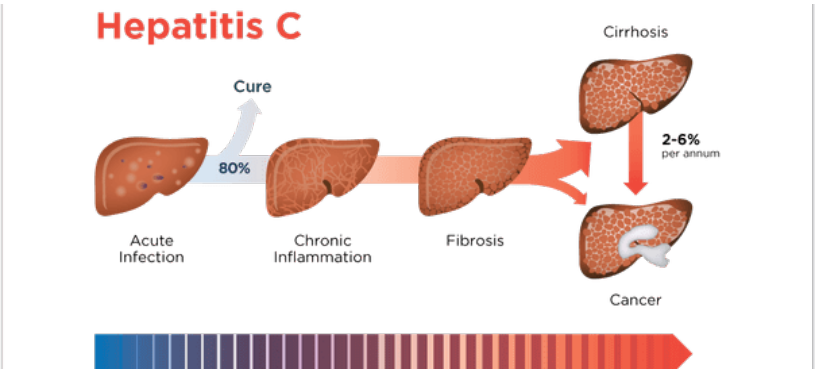
El contenido del dataset es:

615 filas (personas)

13 columnas (variables)

Atributo	Descripción y unidad de medida
Category (target)	Diagnóstico: 0=Donante de sangre ; 0s=Donante de sangre sospechoso ; 1=Hepatitis;2=Fibrosis; 3=Cirrosis
Age	Edad del paciente (años)
Sex	Sexo (f=femenino,m=masculino)
ALB	Albúmina (g/L):La albúmina es una de las varias proteínas producidas en el hígado. El cuerpo necesita estas proteínas para combatir infecciones y realizar otras funciones.
ALP	Fosfatasa alcalina (IU/L):Es una enzima encontrada en el hígado y en los huesos, importante para descomponer proteínas.

Atributo	Descripción y unidad de medida
ALT	Alanina transaminasa (U/L):Es una enzima encontrada en el hígado que ayuda a convertir proteínas en energía para las células hepáticas.
AST	Aspartato aminotransferasa (U/L):Es una enzima encontrada en el hígado que ayuda a metabolizar aminoácidos.
BIL	Bilirrubina (µmol/L):La bilirrubina es una sustancia producida durante la descomposición normal de los glóbulos rojos. La bilirrubina pasa por el hígado y se excreta en las heces.
CHE	Colinesterasa (kU/L):Pertenece a un grupo de enzimas que catalizan la hidrólisis de los ésteres de la colina hacia colina y un ácido graso correspondiente. Se produce en el hígado y se secreta a la sangre.
CHOL	Colesterol (mmol/L):El hígado elabora todo el colesterol que el organismo necesita para formar las membranas celulares y producir ciertas hormonas
CREA	Creatinina (µmol/L):La creatina es un aminoácido que se sintetiza en el hígado y se almacena en los músculos
GGT	Gamma-glutamil transferasa (U/L):Es una enzima qye hace las funciones de molécula de transporte, para mover otras por el cuerpo y ayuda al hígado a metabolizar fármacos y otras toxinas
PROT	Proteínas(g/L):Proteinas totales



Como podemos observar, nuestra variable respuesta es el diagnóstico de cada paciente y el resto de variables actuarán como predictoras. Tenemos 5 valores para nuestra variable respuesta. Los dos primeros pertenecen a los donantes de sangre, y englobaremos a estos donantes como pacientes sanos. El resto de valores son la hepatitis C y otras enfermedades de mayor gravedad desencadendenadas por el mismo virus. Nuestro objetivo principal es la predicción de riesgo por lo que podemos prescindir de las etapas derivadas de la hepatitis transformando nuestra variable respuesta a una variable categórica binaria (sano vs hepatitis). Posteriormente analizaremos si sería mejor (óptimo) separar la variable predictora en dos (sano, hepatitis) o cuatro grupos (sano,hepatitis, fibrosis,cirrosis) mediante análisis de clúster.

Preparación, tratamiento y limpieza de datos.

```
library(xtable)
```

```
# Lectura de datos
d <- read.csv('./data/HepatitisCdata.csv', row.names = 1) # la primera columna contiene el id de los pa
# View(d)
```

```r
# comprobar que existen datos perdidos (retorna booleano)
any(is.na(d))
```

```
## [1] TRUE
```

```r
# determinar el numero de valores perdidos
colSums(is.na(d))
```

```
## Category      Age      Sex      ALB      ALP      ALT      AST      BIL
##        0        0        0        1       18        1        0        0
##      CHE     CHOL     CREA      GGT     PROT
##        0       10        0        0        1
```

Como podemos observar, existen ciertos valores NA en las variables predictoras numéricas.Como solución, hemos optado por sustituir dichos valores por la mediana de su correspondiente variable (columna) ,debido a que gran parte de los NAs pertenecen a observaciones de pacientes hepáticos y solo disponemos de 75 respecto al cómputo total (615 observaciones).Es por ello, por lo que no podemos prescindir de la escasa información que disponemos acerca de los pacientes enfermos.

```r
valores <- list(4, 5, 6, 10, 13)
for(i in valores){
  d[[i]][is.na(d[[i]])] <- median(d[[i]], na.rm = TRUE)}
any(is.na(d)) # Comprobacion de valores nulos o perdidos
```

```
## [1] FALSE
```

Como hemos mencionado antes, nuestra variable respuesta debe ser transformada para que sea binaria.También, vamos a cambiar el nombre para una mayor representabilidad.

```r
colnames(d)[1] <- "Diagnosis"
d$Diagnosis <- ifelse(d$Diagnosis %in% c("0=Blood Donor", "0s=suspect Blood Donor"), 0, 1) # Sano 0, En
# Cambiamos el tipo de datos de la variable Sex y la convertimos en una variable dummmie (0,1) para fut
dum_sex <- ifelse(d$Sex == "m", 1, 0) # Masculino : 0, Femenino: 1
```

Tras las modificaciones aplicadas el dataset queda con la siguiente estructura:

```r
d[1:3,] # ejemplos
```

```
##   Diagnosis Age Sex  ALB  ALP  ALT  AST BIL   CHE CHOL CREA  GGT PROT
## 1         0  32   m 38.5 52.5  7.7 22.1 7.5  6.93 3.23  106 12.1 69.0
## 2         0  32   m 38.5 70.3 18.0 24.7 3.9 11.17 4.80   74 15.6 76.5
## 3         0  32   m 46.9 74.7 36.2 52.6 6.1  8.84 5.20   86 33.2 79.3
```

Estudio inicial de los datos

Análisis exploratorio de datos

Podemos obtener una descripción inicial de las características de los datos mediante:

```r
d_numeric <- d[,4:13] # almacenamos analisis sanguineo de nuestro dataset original
summary(d)
```

```
##    Diagnosis          Age             Sex                 ALB
##  Min.   :0.000   Min.   :19.00   Length:615         Min.   :14.90
##  1st Qu.:0.000   1st Qu.:39.00   Class :character   1st Qu.:38.80
##  Median :0.000   Median :47.00   Mode  :character   Median :41.95
##  Mean   :0.122   Mean   :47.41                      Mean   :41.62
##  3rd Qu.:0.000   3rd Qu.:54.00                      3rd Qu.:45.20
##  Max.   :1.000   Max.   :77.00                      Max.   :82.20
##       ALP              ALT              AST              BIL
##  Min.   : 11.30   Min.   :  0.90   Min.   : 10.60   Min.   :  0.8
##  1st Qu.: 52.95   1st Qu.: 16.40   1st Qu.: 21.60   1st Qu.:  5.3
##  Median : 66.20   Median : 23.00   Median : 25.90   Median :  7.3
##  Mean   : 68.22   Mean   : 28.44   Mean   : 34.79   Mean   : 11.4
##  3rd Qu.: 79.30   3rd Qu.: 33.05   3rd Qu.: 32.90   3rd Qu.: 11.2
##  Max.   :416.60   Max.   :325.30   Max.   :324.00   Max.   :254.0
##       CHE              CHOL             CREA              GGT
##  Min.   : 1.420   Min.   :1.430   Min.   :   8.00   Min.   :  4.50
##  1st Qu.: 6.935   1st Qu.:4.620   1st Qu.:  67.00   1st Qu.: 15.70
##  Median : 8.260   Median :5.300   Median :  77.00   Median : 23.30
##  Mean   : 8.197   Mean   :5.367   Mean   :  81.29   Mean   : 39.53
##  3rd Qu.: 9.590   3rd Qu.:6.055   3rd Qu.:  88.00   3rd Qu.: 40.20
##  Max.   :16.410   Max.   :9.670   Max.   :1079.10   Max.   :650.90
##       PROT
##  Min.   :44.80
##  1st Qu.:69.30
##  Median :72.20
##  Mean   :72.04
##  3rd Qu.:75.40
##  Max.   :90.00
```

El dataset contiene 11 variable numéricas y una categórica.

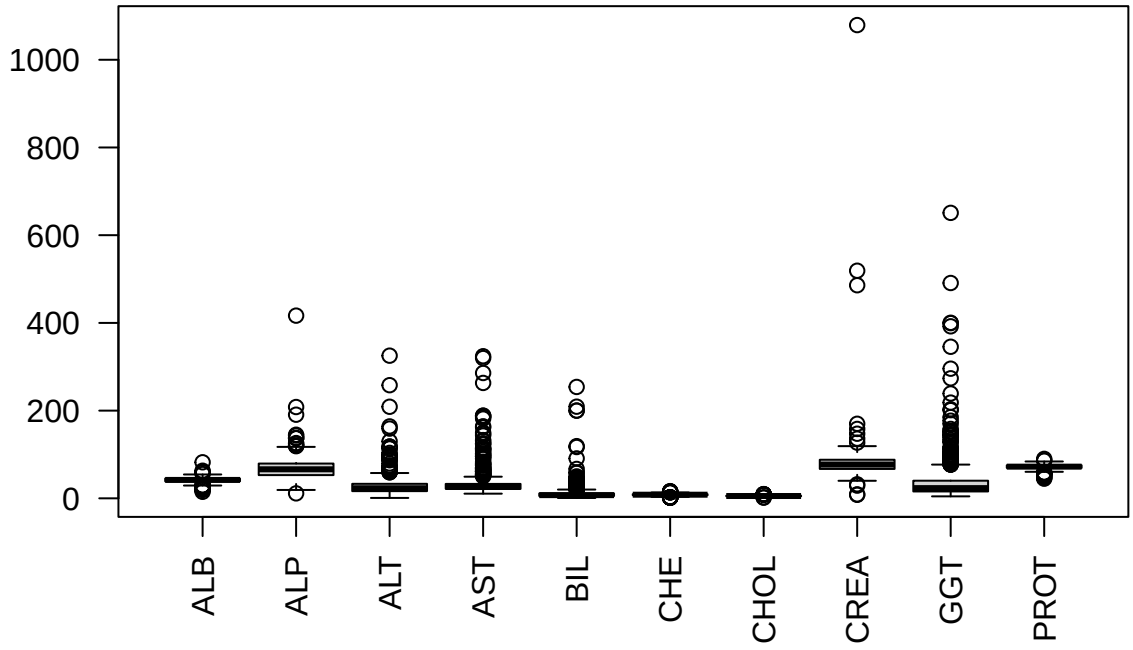
Datos relevantes en las variables bioquímicas

- **ALB (Albúmina)**: Tiene una media de **41.62 g/L**, dentro del rango normal (35-50 g/L). Sin embargo, hay valores mínimos bastante bajos (**14.9 g/L**), lo que podría indicar **desnutrición o enfermedad hepática avanzada**.
- **ALP (Fosfatasa alcalina)**: Tiene una media de **68.22 U/L**, con valores extremos de hasta **416.6 U/L**, lo que sugiere que hay casos con **daño hepático severo o colestasis**.
- **ALT y AST (Transaminasas)**: Ambas tienen valores elevados en algunos pacientes (**ALT hasta 325.3 U/L y AST hasta 324 U/L**), lo cual es típico en enfermedades hepáticas como **hepatitis viral o daño hepático**.
- **BIL (Bilirrubina)**: La media es **11.4 mol/L**, pero hay pacientes con valores extremadamente altos (**hasta 254 mol/L**), lo que sugiere casos con **ictericia o insuficiencia hepática**.
- **GGT (Gamma-Glutamil Transferasa)**: Valores extremadamente altos (**hasta 650.9 U/L**) pueden indicar **hepatopatía alcohólica o colestasis**.
- **CHE (Colinesterasa)**: Media de **8.197 kU/L**, lo que está dentro de lo normal, pero valores bajos pueden sugerir **disfunción hepática**.

- **CHOL (Colesterol)**: Media de **5.36 mmol/L**, con valores que van desde **1.43 a 9.67 mmol/L**. Valores bajos pueden estar relacionados con enfermedad hepática crónica.
- **CREA (Creatinina)**: La media es **81.29 mol/L**, pero hay valores extremos de **hasta 1079.1 mol/L**, lo que indica que algunos pacientes pueden tener **fallo renal asociado**.
- **PROT (Proteínas totales)**: Media de **72.04 g/L**, dentro del rango normal (60-80 g/L), pero valores bajos pueden reflejar **fallo hepático crónico**.

En cuanto a los datos de los pacientes, podemos observar que el estudio se ha realizado en pacientes de entre 19 y 77 años, siendo la media de 47 años.

```
boxplot(d_numeric, las=2)
```



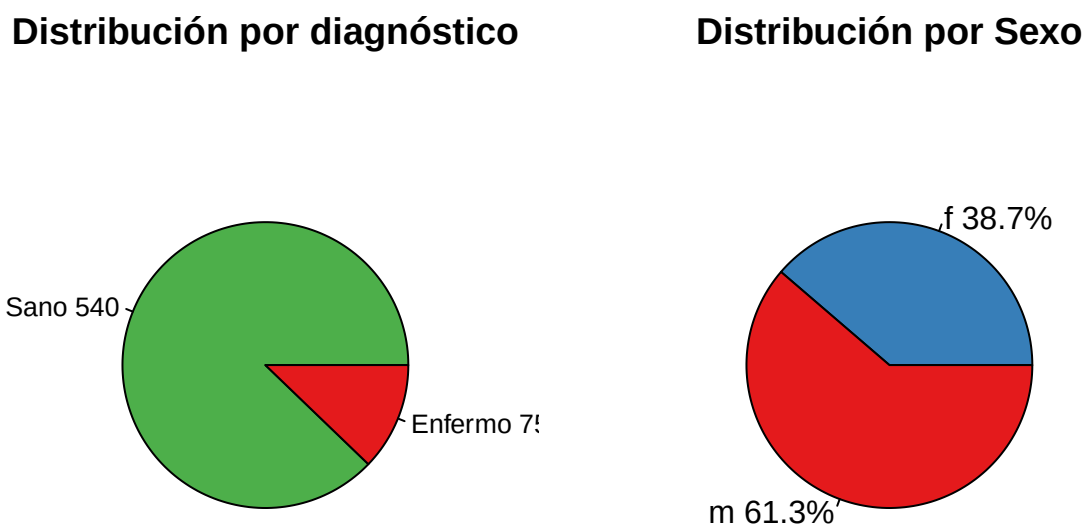
Como podemos observar en el **boxplot**, las variables numéricas utilizan escalas muy diferentes (diferentes magnitudes). Esto también se puede comprobar calculando el vector de medias de cada variable. Por otra parte, es inevitable mencionar la gran cantidad de valores atípicos observados a lo largo de nuestras variables de estudio.

```
mu <-colMeans(d_numeric)
mu
```

```
##       ALB       ALP       ALT       AST       BIL       CHE      CHOL      CREA
## 41.620732 68.222927 28.441951 34.786341 11.396748  8.196634  5.366992 81.287805
##       GGT      PROT
## 39.533171 72.044390
```

Esto nos plantea la posible necesidad de normalizar y estandarizar los datos para un análisis más preciso.

```
diag_counts <- table(d$Diagnosis)
diag_labels <- sprintf("%s %d", c("Sano", "Enfermo"), diag_counts)
sex_counts <- table(d$Sex)
sex_labels <- sprintf("%s %.1f%%", c("f", "m"), 100 * prop.table(sex_counts))
par(mfrow=c(1,2))
pie(table(d$Diagnosis), main = "Distribución por diagnóstico", col = c("#4DAF4A", "#E41A1C"), labels = d
pie(table(dum_sex), main = "Distribución por Sexo", col = c("#377EB8", "#E41A1C"), labels = sex_labels)
```



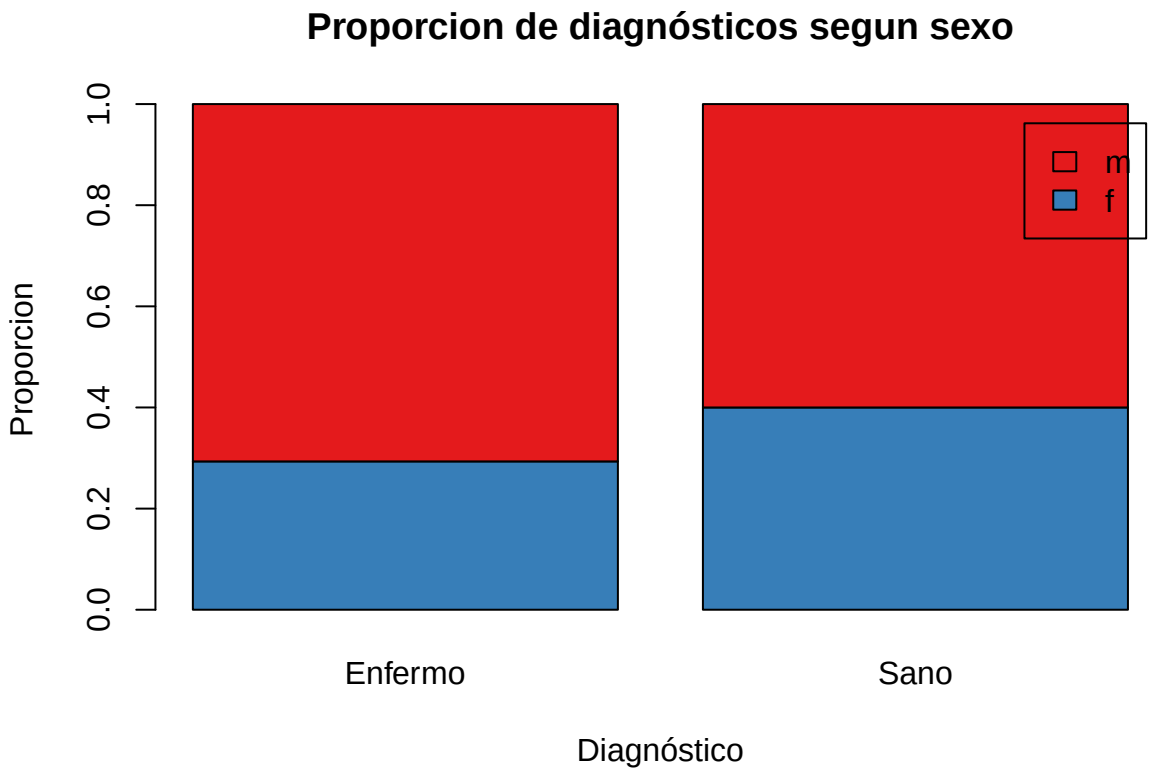
Podemos observar que **la cantidad de observaciones de pacientes sanos es aproximadamente 7 veces mayor que la de enfermos**.Ante este tipo de situación y dependiendo de los modelos que apliquemos podemos escalar el número de pacientes para igualar el número de observaciones de cada diagnóstico mediante algoritmos de ***oversampling***.En ciertas ocasiones podemos arriesgarnos a realizar un estudio posiblemente sesgado. Sin lugar a duda, es un factor fundamental que debemos tener en cuenta a lo largo del desarrollo de nuestro estudio

Aproximadamente, 3/5 de las observaciones provienen de hombres y el resto de mujeres.

La Hepatitis C afecta más comúnmente a los hombres que a las mujeres. Las mujeres tienen una probabilidad menor de contraer el virus y, cuando lo hacen, la progresión de la enfermedad suele ser más lenta.

```
diagnostico <- ifelse(d$Diagnosis == 0, "Sano", "Enfermo")
# Calculamos proporciones por grupo de Diagnostico
prop <- prop.table(table(diagnostico, d$Sex), 1)
barplot(t(prop),
        col =c("#377EB8", "#E41A1C"),
```

```
        xlab = "Diagnóstico",
        ylab = "Proporcion",
        main = "Proporcion de diagnósticos segun sexo",
        legend = TRUE
        )
```



La diferencia de proporción de cada sexo no es tan significativa entre pacientes enfermos y sanos.

Análisis del dataset

En este apartado vamos a realizar una visualización de los aspectos más importantes del dataset mediante un estudio general (estudio bivariante, densidad, diagrama de caja, correlación,...).

Podemos obtener una descripción general mediante

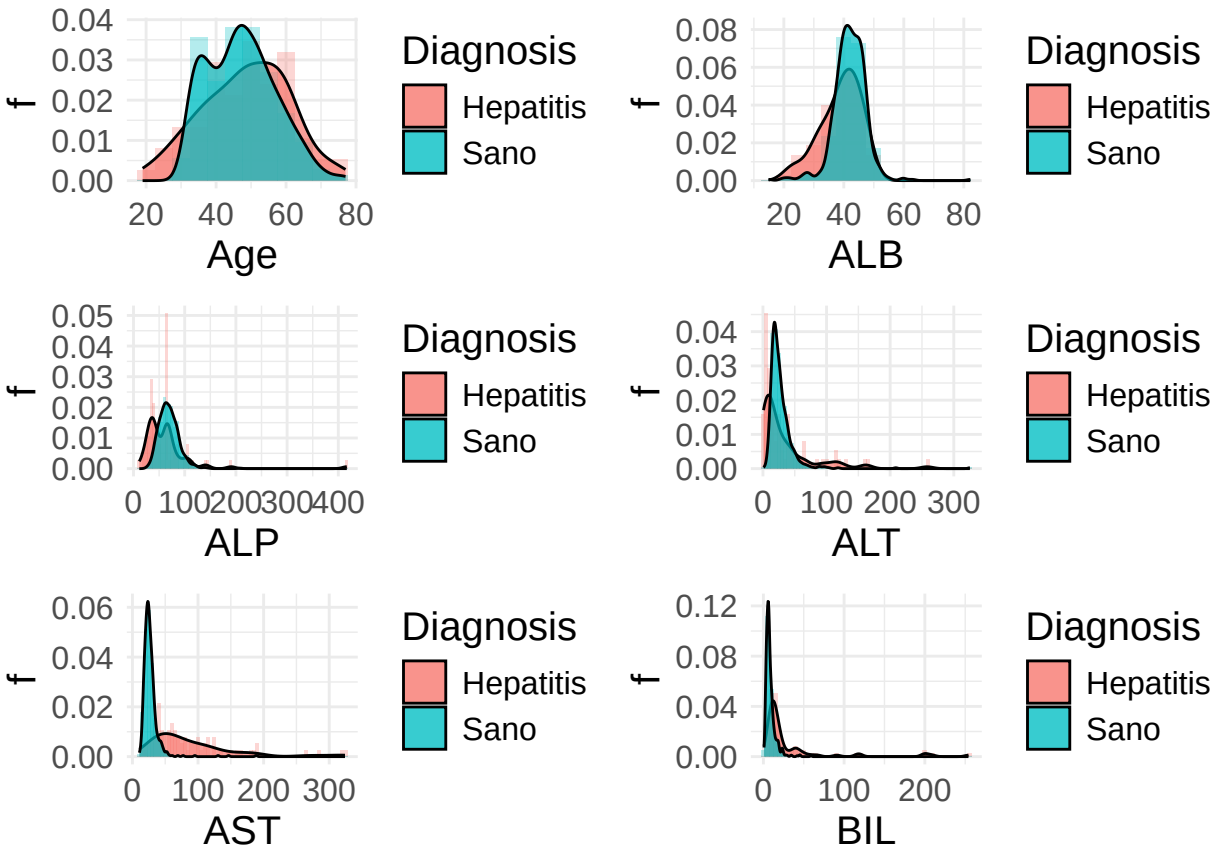
Análisis univariante de los datos

```
# Librerías necesarias
library(ggplot2)
library(cowplot)

# Variables categóricas a factor
d$Diagnosis <- ifelse(d$Diagnosis == 0, "Sano", "Hepatitis")
```

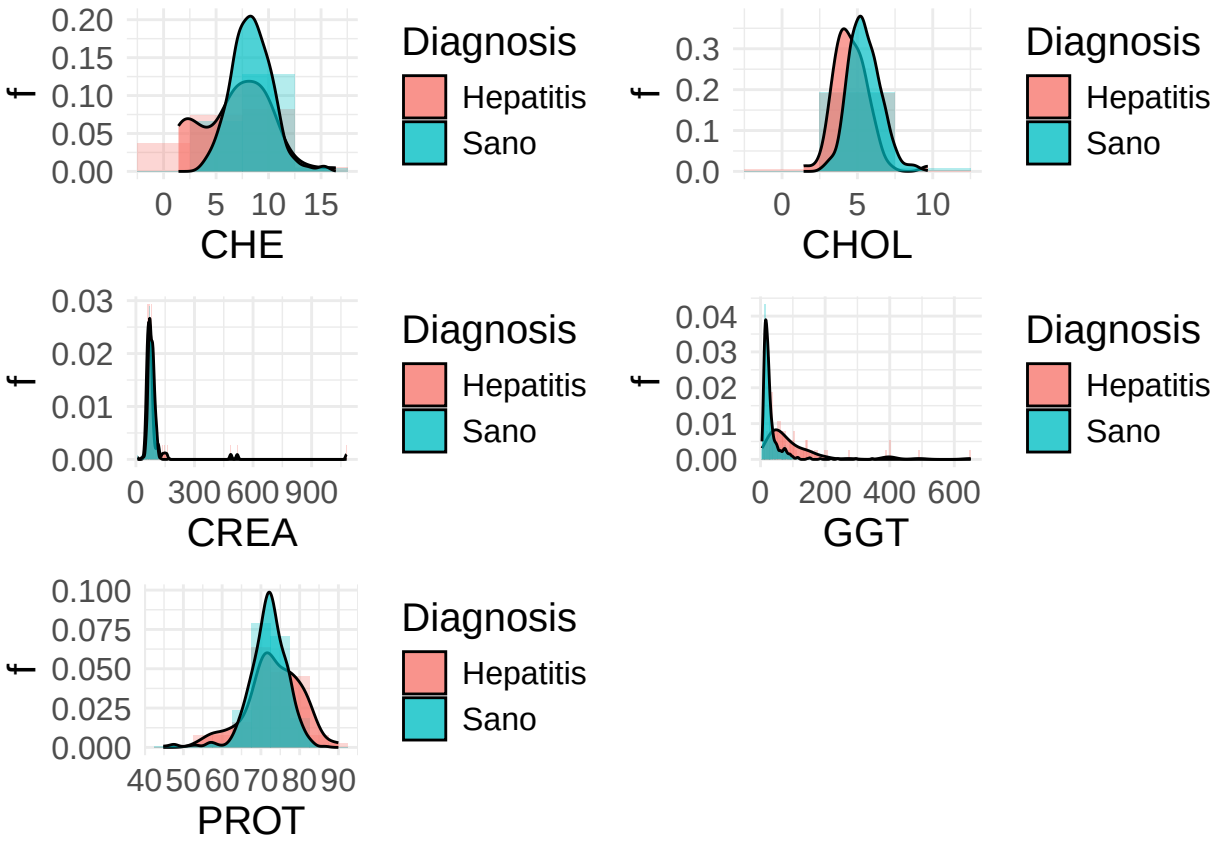
```r
d$Diagnosis <- as.factor(d$Diagnosis)
d$Sex <- as.factor(d$Sex)

# Variables a graficar
vars <- c("Age", "ALB", "ALP", "ALT", "AST", "BIL", "CHE", "CHOL", "CREA", "GGT", "PROT")

# gráficos mediante bucle lapply
plots <- lapply(vars, function(var) {
  ggplot(d, aes_string(x = var, fill = "Diagnosis")) +
    geom_histogram(aes(y = after_stat(density)), binwidth = 5, alpha = 0.3, position = "identity") +
    geom_density(alpha = 0.7, adjust = 1) +
    labs(x = var, y = "f") +
    theme_minimal() +
    theme(text = element_text(size = 15))
})
```

```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
# Visualización con cowplot
cowplot::plot_grid(plotlist = plots[1:6], ncol = 2, nrow = 3)
```



```
cowplot::plot_grid(plotlist = plots[7:11], ncol = 2, nrow = 3)
```

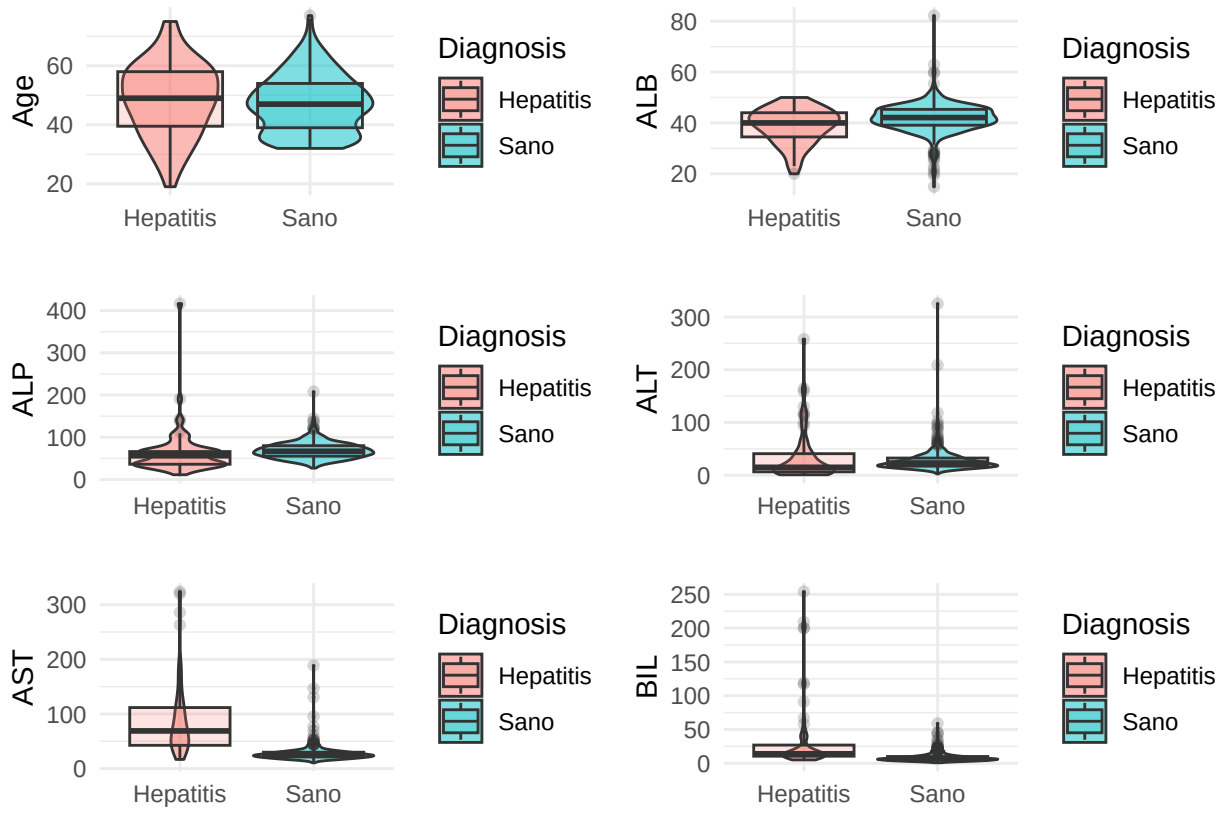


A partir de estos histogramas enriquecidos con la curva de densidad, podemos visualizar la distribución de cada una de las variables numéricas de nuestro dataset y obtener ciertas conclusiones:

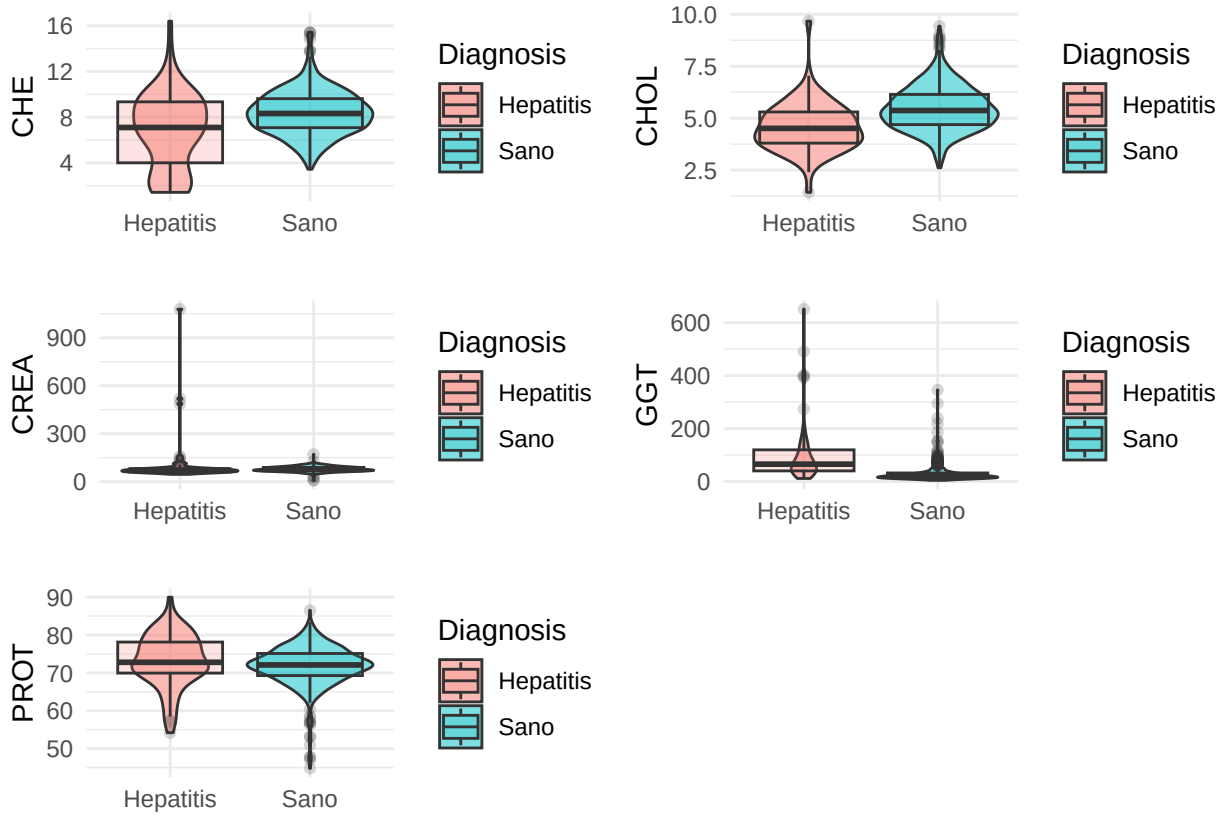
- La tasa de hepatitis en los jóvenes es **sorprendenetemente alta.**
- Niveles bajos en **ALB,ALP, CHE, CHOL** son indicadores de riesgo
- Niveles altos en **ALT, AST, BIL, GGT** son indicadores de riesgo
- La creatina (**CREA**) parece no tener relación con la hepatitis
- La cantidad proteína (**PROT**) parece no despuntar mucho cuando existen problemas de hígado

```
# Estructura similar a los histogramas
variables <- c("Age", "ALB", "ALP", "ALT", "AST", "BIL",
               "CHE", "CHOL", "CREA", "GGT", "PROT")

# gráficos con bucle
plot_list <- lapply(variables, function(var) {
  ggplot(d, aes(x = Diagnosis, y = .data[[var]])) +
    geom_violin(aes(fill = Diagnosis), alpha = 0.5) +
    geom_boxplot(aes(fill = Diagnosis), alpha = 0.2) +
    theme_minimal() +
    xlab("")
```

```
})


cowplot::plot_grid(plotlist = plot_list, ncol = 2, nrow = 3)
```




```
cowplot::plot_grid(plotlist = plot_list[7:11], ncol = 2, nrow = 3)
```



Observando las proyecciones de las cajas sobre el eje y, puede parecer que en algunos casos se puede aislar los diagnósticos en función de los valores de la variable (p.ej **AST**). De nuevo, volvemos a visualizar la gran cantidad de valores atípicos en las analíticas.

Conclusiones Generales:

- Los pacientes con hepatitis presentan alteraciones claras en biomarcadores hepáticos clave como **ALT, AST, BIL y GGT**, que están significativamente elevados.
- Marcadores relacionados con la síntesis proteica, como **ALB** y **CHE** , están reducidos, lo que refleja una función hepática comprometida.
- Estas diferencias entre los grupos pueden ser útiles para diagnosticar y monitorear la hepatitis, pero deben complementarse con análisis estadísticos para confirmar su significancia clínica.

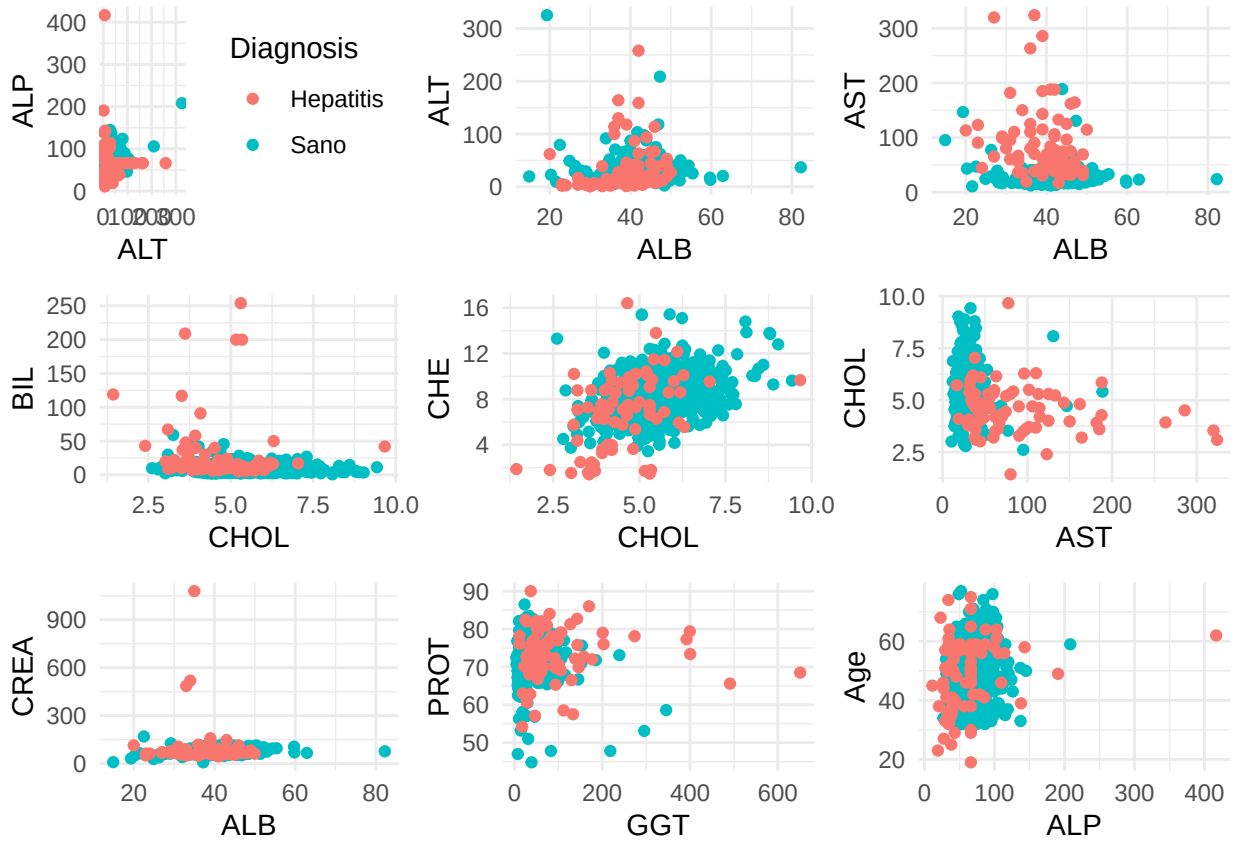
Análisis bivariante de los datos

¿Cómo es afectada la salud del hígado según dos variables?

Representamos alguna relaciones bivariadas para ver si existe alguna relación que nos permita encontrar patrones excluyentes entre los diagnósticos.

```
# pares de variables para cada gráfico
variable_pairs <- list(
  c("ALT", "ALP"),
  c("ALB", "ALT"),
```

```
  c("ALB", "AST"),
  c("CHOL", "BIL"),
  c("CHOL", "CHE"),
  c("AST", "CHOL"),
  c("ALB", "CREA"),
  c("GGT", "PROT"),
  c("ALP", "Age")
)


plot_list <- lapply(seq_along(variable_pairs), function(i) {
  pair <- variable_pairs[[i]]

  p <- ggplot(d, aes(x = .data[[pair[1]]], y = .data[[pair[2]]], color = Diagnosis)) +
    geom_point() +
    theme_minimal()

  # Eliminar leyenda excepto en el primer gráfico
  if(i != 1) {
    p <- p + theme(legend.position = "none")
  }

  return(p)
})

# Organizar gráficos en grid 3x3
cowplot::plot_grid(plotlist = plot_list, ncol = 3, nrow = 3)
```



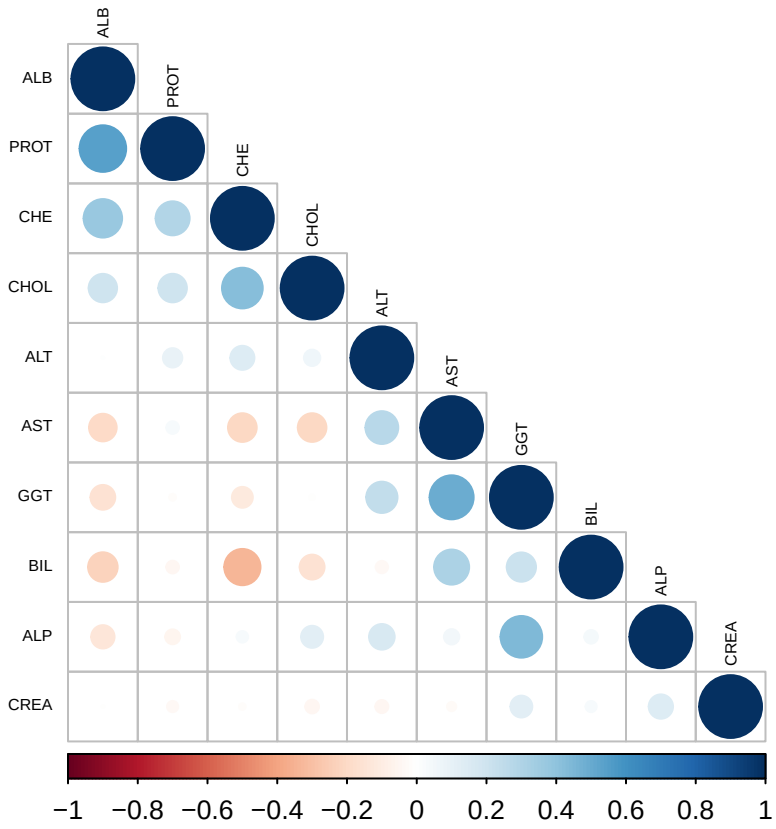
Podemos afirmar que nuestras sospechas eran ciertas (gran concentración de puntos y sin linealidad). No obstante, observamos que **AST** parece ser la mejor variable en cuanto a clasificación.

Para estudiar las correlaciones podemos hacer:

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
C <-cov(d_numeric)
#View(C)
M <-cor(d[,4:13])
corrplot(M, type = "lower", order = "hclust", tl.col = "black", tl.cex = 0.5)
```



Podemos observar notable correlación positiva entre **PROT y ALB**, **CHOL y CHE**, **GGT y AST** y entre **ALP y GGT**. También observamos una correlación negativa significativa entre **CHE y BIL**.

Estudio y análisis de las componentes principales

El **PCA** transforma los datos a un espacio de menor dimensión mientras conserva la mayor cantidad de varianza significativa. Es útil para evaluar conjuntos de datos multivariantes con múltiples características numéricas correlacionadas. Generalmente, los componentes iniciales retienen la mayor parte de la información de los datos originales, reduciendo significativamente la dimensionalidad del estudio.

Cálculo de las componentes principales

Calculamos las **componentes principales** mediante la **matriz de correlacion** (se puede realizar de forma alternativa con la **matriz de covarianzas**), debido a el uso de escalas diferentes. Además, en la matriz de covarianzas hemos visto grandes diferencias que podrían sesgar los resultados.

```
d_numeric$Sex <- dum_sex
d_numeric$Age <- d$Age
PCA<-princomp(d_numeric,cor=TRUE)
```

```
summary(PCA,loadings=TRUE)
```

```
## Importance of components:
##                           Comp.1    Comp.2    Comp.3     Comp.4     Comp.5
```

```
## Standard deviation     1.5689877 1.3781288 1.1908025 1.06983643 0.98988559
## Proportion of Variance 0.2051435 0.1582699 0.1181676 0.09537917 0.08165612
## Cumulative Proportion  0.2051435 0.3634135 0.4815810 0.57696018 0.65861630
##                            Comp.6     Comp.7     Comp.8     Comp.9    Comp.10
## Standard deviation     0.96269220 0.86471702 0.80585169 0.76619968 0.68487040
## Proportion of Variance 0.07723136 0.06231129 0.05411641 0.04892183 0.03908729
## Cumulative Proportion  0.73584766 0.79815895 0.85227536 0.90119719 0.94028448
##                           Comp.11    Comp.12
## Standard deviation     0.6184591 0.57800916
## Proportion of Variance 0.0318743 0.02784122
## Cumulative Proportion  0.9721588 1.00000000
##
## Loadings:
##      Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8 Comp.9 Comp.10
## ALB   0.445  0.207  0.255         0.236         0.234  0.228  0.164  0.265
## ALP  -0.185  0.341 -0.429  0.170  0.173  0.349 -0.182  0.428  0.283 -0.157
## ALT          0.428        -0.254 -0.534  0.196        -0.374  0.506
## AST  -0.369  0.300  0.294 -0.271                0.231 -0.161 -0.430 -0.163
## BIL  -0.342         0.290         0.420 -0.282 -0.542 -0.173  0.287 -0.227
## CHE   0.416  0.301 -0.149        -0.200        -0.152        -0.288 -0.695
## CHOL  0.286  0.253 -0.407 -0.109  0.196 -0.228 -0.378 -0.386 -0.234  0.438
## CREA         0.115         0.795  0.107  0.132  0.248 -0.507
## GGT  -0.348  0.451                0.140  0.146         0.205 -0.363  0.266
## PROT  0.316  0.314  0.301 -0.174  0.448         0.198         0.177 -0.150
## Sex          0.303  0.304  0.378 -0.356 -0.537 -0.142  0.342         0.181
## Age  -0.177        -0.452 -0.121  0.126 -0.603  0.524         0.260 -0.124
##      Comp.11 Comp.12
## ALB   0.565   0.338
## ALP  -0.188   0.359
## ALT   0.118
## AST  -0.100   0.558
## BIL   0.288
## CHE   0.247  -0.110
## CHOL -0.137   0.171
## CREA
## GGT   0.280  -0.552
## PROT -0.535  -0.300
## Sex  -0.296
## Age
```

```
PCA$loadings->L # almaceno los vectores propios unitarios
PCA$scores->S # Son los valores que obtendrian los individuos de la muestra (personas) en las CP
Y1<-S[,1]
Y2<-S[,2]
```

Los valores ausentes en Loadings son números pequeños pero no necesariamente 0

Obtenemos el cálculo de 12 componentes principales y sus respectivas importancias.La **primera componente mantiene un 20.51435%** de la información inicial, seguido de **la segunda con un 15.82699%**, sumando un **36.34134%**. A partir de los l**oadings** (vectores propios unitarios de las varianzas de cada componente (*sd^2*) podemos calcular los coeficientes asociados a las componentes principales:

```
L[,1]
```

```
##          ALB          ALP          ALT          AST          BIL          CHE
##  0.444759244 -0.184953269 -0.061613481 -0.368998128 -0.341957985  0.415830233
##         CHOL         CREA          GGT         PROT          Sex          Age
##  0.286193501 -0.065058588 -0.347537943  0.316292336 -0.001231504 -0.176667194
```

Y1 = 0.444759244 * ***ALB*** -0.184953269 ***ALP*** -0.061613481 ***ALT*** -0.368998128 * ***AST*** -0.341957985 * ***BIL*** + 0.415830233 * ***CHE*** + 0.286193501 ***CHOL*** -0.065058588 * ***CREA*** -0.347537943 * ***GGT*** + 0.316292336 * ***PROT*** -0.001231504 * ***Sex*** -0.176667194 * ***Age***

donde todas las variables como ***ALB*** han sido estandarizadas (debido al uso de la matriz de correlación, con la matriz de covarianza no es necesario).

Cuanto mayor es el valor absoluto de los coeficientes, mas participación y por ende importancia tiene dicha variable en el cálculo de la componente principal asociada

Análisis de componentes principales

Si observamos los coeficientes de **Y1**, podemos observar que las variables que más influyen en esta componente son : **ALB** (+), **CHE** (+), **AST** (-), **GGT** (-), **PROT** (+), **CHOL** (+). La creatina y el sexo del paciente, parecen no tener efecto apenas sobre **Y1**

Por lo tanto,si nos fijamos en la conclusiones que habíamos obtenido cuando hemos tratado con los indicadores de riesgo, podemos ver que **Y1** representa en gran medida pacientes cuyos niveles de enzimas y proteinas son opuestos a los indicadores de riesgo. Por tanto, estaríamos hablando de pacientes con un analítica que indica un estado saludable.

Podemos obtener un resumen de las características de la componente 1 y 2 mediante

```
summary(S[,1])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -9.2929 -0.3581  0.2791  0.0000  0.8534  2.8534
```

```
d[which.max(S[,1]),]
```

```
##     Diagnosis Age Sex  ALB  ALP ALT  AST BIL CHE CHOL CREA  GGT PROT
## 217      Sano  52   m 82.2 82.2  37 23.7 7.8 8.9 6.09   77 87.8 67.4
```

```
d[which.min(S[,1]),]
```

```
##     Diagnosis Age Sex ALB   ALP ALT   AST BIL  CHE CHOL CREA   GGT PROT
## 611 Hepatitis  62   f  32 416.6 5.9 110.3  50 5.57  6.3 55.7 650.9 68.5
```

```
summary(S[,2])
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -4.13848 -0.88434 -0.09358  0.00000  0.73806  9.02559
```

Observamos que el paciente con mayor **Y1** está sano, mientras que el paciente con menor Y1 está enfermo, por tanto, refuerza nuestra hipótesis inicial.

Para **Y2** los coeficientes son:

```
L[,2]
```

```
##        ALB        ALP        ALT        AST        BIL        CHE       CHOL
## 0.20673635 0.34146562 0.42760659 0.30039843 0.03538565 0.30134611 0.25318866
##       CREA        GGT       PROT        Sex        Age
## 0.11469558 0.45089271 0.31412830 0.30263924 0.06842644
```

Todos los coeficientes son positivos, por lo que hay ciertos indicadores de riesgo (**ALT, AST,GGT**) que denotan problemas en el hígado del paciente. Además, el sexo parece también ser un aspecto importante.

```
summary(S[,2])
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -4.13848 -0.88434 -0.09358  0.00000  0.73806  9.02559
```

```
d[which.max(S[,2]),]
```

```
##     Diagnosis Age Sex ALB   ALP ALT   AST BIL  CHE CHOL CREA   GGT PROT
## 611 Hepatitis  62   f  32 416.6 5.9 110.3  50 5.57  6.3 55.7 650.9 68.5
```

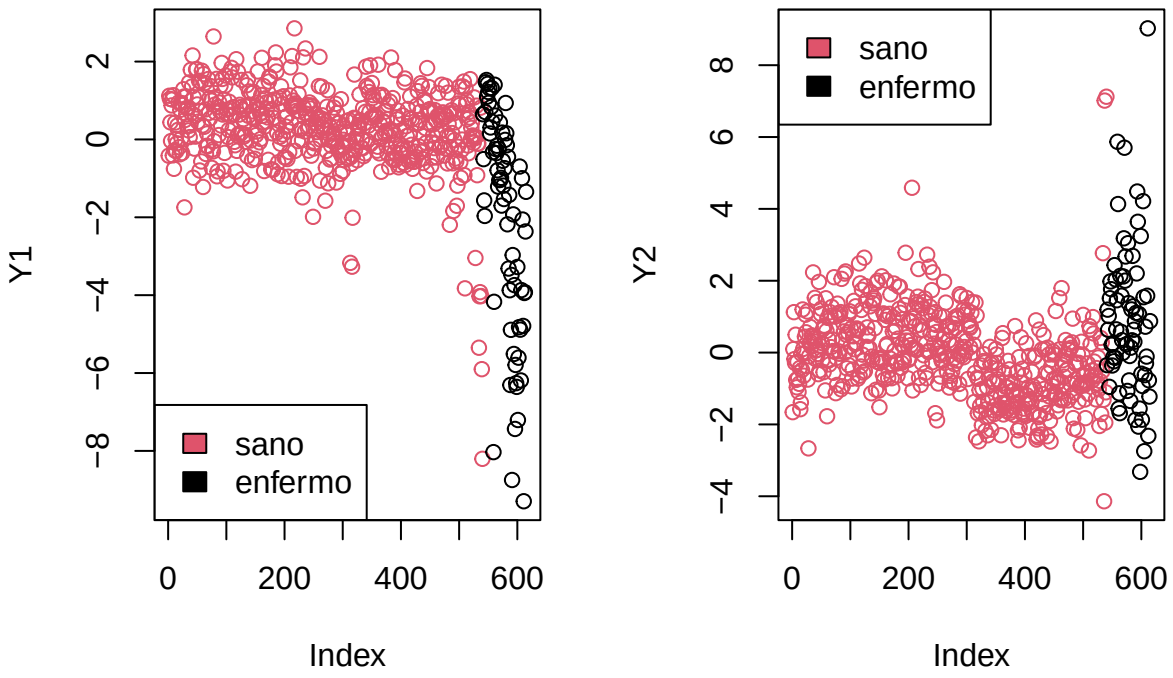
```
d[which.min(S[,2]),]
```

```
##     Diagnosis Age Sex  ALB  ALP ALT  AST BIL  CHE CHOL CREA  GGT PROT
## 536      Sano  49   m 21.6 42.2 9.5 10.6 2.4 3.75 3.01   64 38.9 44.8
```

Observamos que el paciente con mayor **Y2** está enfermo, mientras que el paciente con menor **Y2** está sano, por tanto, refuerza nuestra hipótesis acerca de esta componente. Por tanto, estaríamos hablando de pacientes con un análisis cuyos valores generales elevados.

Obs:El mínimo de la componente principal 1 es bastante bajo

```
par(mfrow=c(1,2))
plot(S[,1], ylab = 'Y1', col=as.integer(d$Diagnosis))
legend(x = "bottomleft",          # Position
       legend = c("sano", "enfermo"), fill = c(2,1))
plot(S[,2], ylab = 'Y2',col=as.integer(d$Diagnosis))
legend(x = "topleft",          # Position
       legend = c("sano", "enfermo"), fill = c(2,1))
```

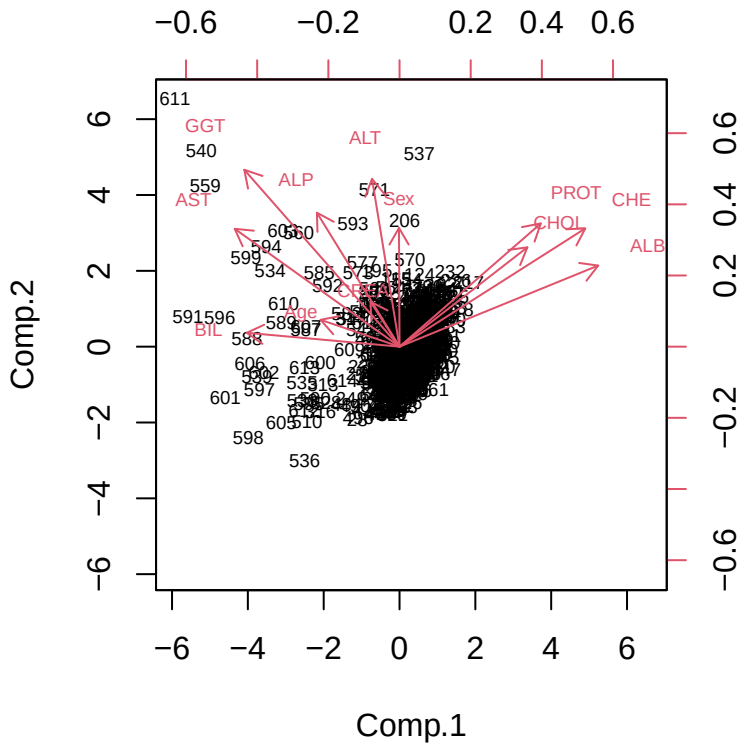


Podemos observar que generalizando, los valores de Y1 con mayor valor, pertenecen a pacientes sanos, no obstante, existen muchos pacientes enfermos que han obtenido similares a los pacientes sanos,por lo que la clasificación de diagnóstico, como veremos posteriormente no será fácil. Por otra parte,sorprenden algunos pacientes sanos cuyos valores en **Y1** muy bajos.

Por otro lado, sucede algo similar con **Y2** pero de forma inversa

Para representar las cargas y las puntuaciones estandarizadas de las dos primeras componentes haremos:

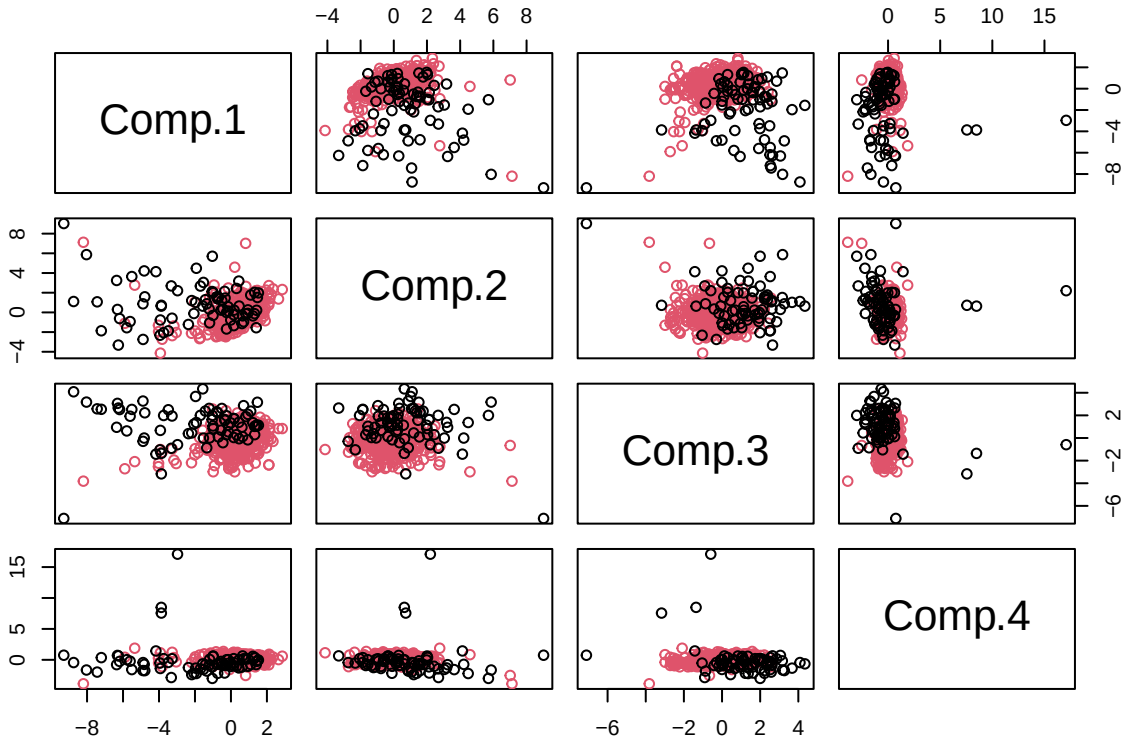
```
biplot(PCA, pc.biplot=TRUE, cex = 0.6)
```



Podemos observar una gran concentración de observaciones entre los intervalos -2 y 2 de ambas componentes

En el resto de componentes principales, tampoco resulta tarea fácil diferenciar a los pacientes

```
pairs(PCA$scores[,1:4], col=as.integer(d$Diagnosis))
```



En el resto de componentes principales, tampoco resulta tarea fácil diferenciar a los pacientes

Saturaciones

Para calcular las saturaciones

```
PCA$scores -> S
SAT <- cor(d_numeric, S)
#data.frame(SAT)
```

Estas saturaciones son los coeficientes de las variables originales en cada una de las componentes principales y sirven para interpretar la contribución de cada variable **original** a cada componente principal.

Si queremos ver cuales son las variables mejores respresentadas en cada componente principal, elevamos al cuadrado las saturaciones (información)

```
data.frame(SAT^2)
```

```
##            Comp.1      Comp.2       Comp.3       Comp.4       Comp.5
## ALB  4.869552e-01 0.081173323 9.251777e-02 1.126107e-03 5.436375e-02
## ALP  8.420989e-02 0.221448940 2.604387e-01 3.320071e-02 2.942533e-02
## ALT  9.345242e-03 0.347270921 1.655854e-03 7.407985e-02 2.794557e-01
## AST  3.351872e-01 0.171385855 1.227960e-01 8.382463e-02 7.255397e-06
## BIL  2.878622e-01 0.002378121 1.196033e-01 1.826964e-05 1.727183e-01
## CHE  4.256682e-01 0.172468908 3.160228e-02 1.804683e-03 3.933639e-02
```

```
## CHOL 2.016316e-01 0.121749766 2.348557e-01 1.352905e-02 3.763217e-02
## CREA 1.041954e-02 0.024984633 2.462870e-06 7.225815e-01 1.129490e-02
## GGT  2.973333e-01 0.386123354 5.142566e-03 1.129730e-05 1.927754e-02
## PROT 2.462728e-01 0.187410432 1.283385e-01 3.457491e-02 1.967143e-01
## Sex  3.733454e-06 0.173952276 1.310826e-01 1.631757e-01 1.241150e-01
## Age  7.683355e-02 0.008892575 2.899749e-01 1.662334e-02 1.553281e-02
##            Comp.6       Comp.7       Comp.8       Comp.9      Comp.10
## ALB  4.504903e-06 0.0410371640 0.0336754855 1.574964e-02 0.0329661315
## ALP  1.126605e-01 0.0247162030 0.1187769152 4.690527e-02 0.0115746419
## ALT  3.568858e-02 0.0001481942 0.0908534490 1.505060e-01 0.0041988765
## AST  8.572266e-04 0.0400238654 0.0167503330 1.087659e-01 0.0124098126
## BIL  7.375236e-02 0.2194658852 0.0195311063 4.838468e-02 0.0242110163
## CHE  9.065839e-03 0.0173548084 0.0000360824 4.862337e-02 0.2265867876
## CHOL 4.812978e-02 0.1067633166 0.0966443812 3.216167e-02 0.0899379747
## CREA 1.625582e-02 0.0459036992 0.1670234137 1.982994e-05 0.0009502316
## GGT  1.966276e-02 0.0027873093 0.0273687381 7.744917e-02 0.0330844775
## PROT 6.400358e-03 0.0292982763 0.0026874753 1.835678e-02 0.0105432888
## Sex  2.672110e-01 0.0151263601 0.0759052271 5.617106e-04 0.0153976881
## Age  3.370875e-01 0.2051104390 0.0001443468 3.957796e-02 0.0071865371
##           Comp.11      Comp.12
## ALB  0.1222780525 3.815284e-02
## ALP  0.0135856601 4.305722e-02
## ALT  0.0053096175 1.487723e-03
## AST  0.0038388164 1.041532e-01
## BIL  0.0317339408 3.408096e-04
## CHE  0.0233758401 4.076821e-03
## CHOL 0.0071405920 9.823958e-03
## CREA 0.0003061069 2.578815e-04
## GGT  0.0299045421 1.018550e-01
## PROT 0.1093566883 3.004621e-02
## Sex  0.0334417312 2.703887e-05
## Age  0.0022200526 8.159516e-04
```

Variable mejor representada en **Y1**: **ALB** (48.69552%)

Variable peor representada en **Y1**: **Sex** (3.733454e-04%)

Variable mejor representada en **Y2**: **GGT**(38.6123354%)

Variable peor representada en **Y2**: **BIL**(0.2378121%)

Comunalidad

Indica en tanto por 1, la información que mantienen en este caso las primeras dos componentes sobre cada variable (Suma de las saturaciones al cuadrado de cada componente principal en el rango deseado)

```
COM2 <- SAT[,1]^2 + SAT[,2]^2
data.frame(COM2)
```

```
##           COM2
## ALB  0.56812856
## ALP  0.30565883
## ALT  0.35661616
## AST  0.50657304
## BIL  0.29024028
## CHE  0.59813710
```

```
## CHOL 0.32338137
## CREA 0.03540417
## GGT  0.68345664
## PROT 0.43368321
## Sex  0.17395601
## Age  0.08572612
```

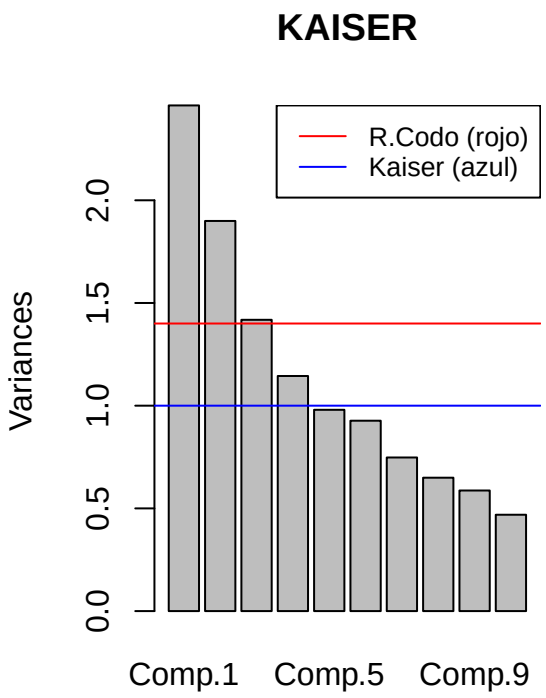
```r
COM2 <- SAT[,1]^2 + SAT[,2]^2 + SAT[,3]^2 + SAT[,4]^2
data.frame(COM2)
```

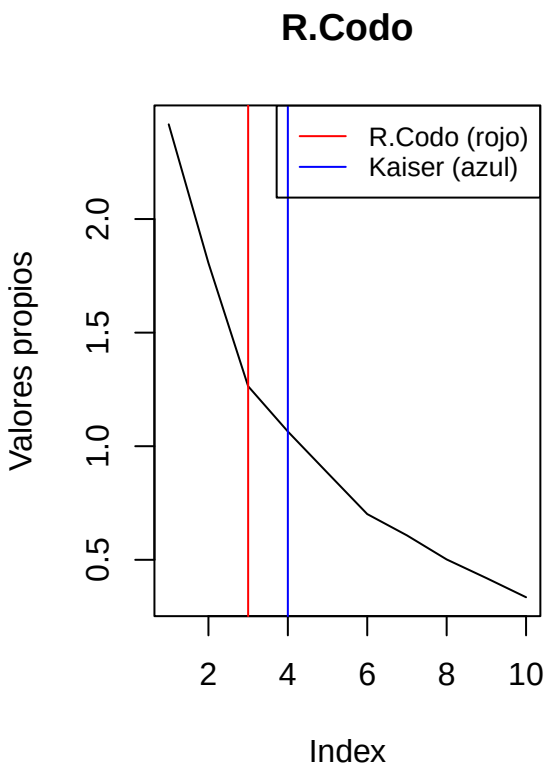
```
##           COM2
## ALB  0.6617724
## ALP  0.5992982
## ALT  0.4323519
## AST  0.7131936
## BIL  0.4098619
## CHE  0.6315441
## CHOL 0.5717662
## CREA 0.7579881
## GGT  0.6886105
## PROT 0.5965966
## Sex  0.4682142
## Age  0.3923244
```

Numero de componentes

Existen números reglas y técnicas para determinar el número de componentes, utilizaremos dos muy comunes:

```r
par(mfrow = c(1, 2))

screeplot(PCA, main = "KAISER")
abline(h = 1, col = 'blue') # Línea horizontal en 1 (criterio de Kaiser)
abline(h = 1.4, col = "red")
legend("topright", legend = c("R.Codo (rojo)", "Kaiser (azul)"),
       col = c("red", "blue"), lty = 1, cex = 0.8)

# Gráfico de valores propios con título "R.Codo"
plot(eigen(M)$values, type = 'l', ylab = 'Valores propios')
abline(v = 3, col = "red") # Línea vertical en 3 (regla del codo)
abline(v = 4, col = "blue") # Línea vertical en 4 (criterio de Kaiser)
title(main = "R.Codo")

# Añadir leyenda a la ventana gráfica
legend("topright", legend = c("R.Codo (rojo)", "Kaiser (azul)"),
       col = c("red", "blue"), lty = 1, cex = 0.8)
```





Regla del codo o sedimentacion

La regla del codo establece que serán representativas las componentes hasta el primer codo (sin incluirlo),

si observamos el gráfico obtenido, podemos ver que el primer codo surge cuando Index = 3, por lo que tomaríamos las dos primeras componentes.

Basándonos en las representación de la regla del codo, podemos concluir que **las primeras dos componentes principales son suficientes para capturar la mayoría de la información**.

Regla de kaiser

La regla de kaiser establece que que solo serán relevantes las componentes que tengan una variabilidad mayor que la variabilidad mediad de las variables originales. Como hemos utilizado la matriz de correlaciones para calcular las componentes principales, nuestras variables están estandarizadas y por ende, la varianzas medias e iniciales son 1.

Si usamos la regla de kaiser podemos concluir que el numero de componentes óptimo es 4

Conclusiones PCA

EL resultado de las reglas aplicadas difiere.Esto abre el dilema del compromiso entre complejidad y explicación.Es decir, a mayor número de componentes, obtenemos una imagen más completa pero a costa de una mayor complejidad. No obstante, si las componentes se reducen, podemos omitir detalles fundamentales.

Debido a la concentración y dificultad de clasificación a simple vista entre pacientes enfermos y sanos, desde mi punto de vista, un aumento de la complejidad es necesario y por tanto, optaría por **4 componentes principales.**

Con 4 componentes principales, retenemos aproximadamente un 66% de la información, un 24% más que si utilizaramos dos componentes.

Medainte los resultados obtenidos, llegamos a la conclusión de que podemos conglomerar los datos en un número menor de variables, capturando los conceptos más importantes en 4 dimensiones. Esto nos demuestra que el **PCA** es es de gran utilidad para hacer más manejables los conjuntos de datos grandes y nos ayuda a entender mejor el comportamiento de los datos.

Clasificación supervisada

La clasificación supervisada es especialmente útil en nuestro contexto ya que proporciona un medio adecuado y efectivo para hacer predicciones precisas en base a la muestras de otros pacientes que han sido analizado en el laboratorio.

Análisis discriminante lineal LDA

El análisis discriminante lineal es una técnica de clasificación supervisada muy útil en nuestro campo de estudio, ya que puede ser de gran ayuda para diferenciar entre pacientes sanos y enfermos.Se basa en los resultados de comparación de analíticas, encontrando un patrón preciso. Esta técnica es especialmente útil si la variable respuesta es **linealmente separable** (podemos trazar un hiperplano dividiendo efectivamente) y los datos se ajustan razonablemente bien a la suposición de normalidad.

En nuestro caso trataremos de estudiar la diferencias de las variables según el tipo de diagnóstico para comprobar si serán de utilidad a la hora de diagnosticar a los individuos. Un ejemplo sería:

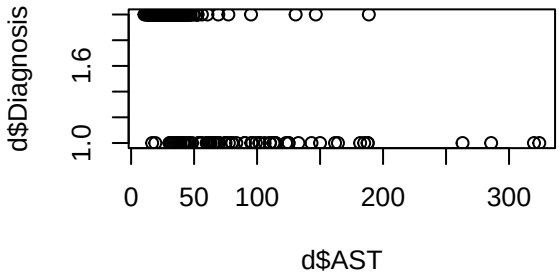
```
par(mfrow=c(2,2))
tapply(d$AST, d$Diagnosis,summary)
```

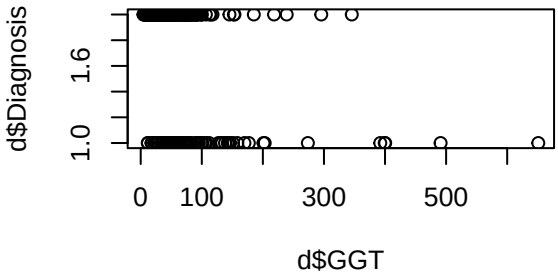
```
## $Hepatitis
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   16.70   42.55   68.90   89.95  111.65  324.00
##
## $Sano
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   10.60   21.18   24.90   27.13   29.93  188.70
```

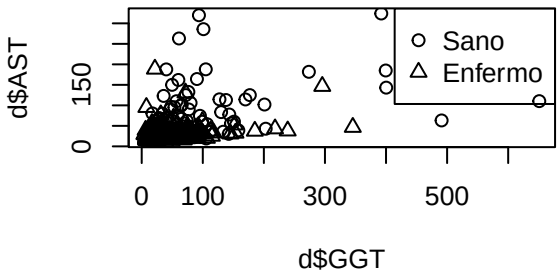
```
plot(d$AST, d$Diagnosis)
tapply(d$GGT, d$Diagnosis,summary)
```

```
## $Hepatitis
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   11.50   40.15   65.60  103.68  119.65  650.90
##
## $Sano
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    4.50   15.10   21.50   30.62   32.35  345.60
```

```r
plot(d$GGT, d$Diagnosis)
plot(d$GGT, d$AST, pch=as.integer(d$Diagnosis))
legend('topright', legend = c('Sano','Enfermo'), pch=1:2)
```







Es redundante realizar el cálculo sobre todas las variables al haber sido realizado en el estudio inicial, donde hemos comprobado que no se aprecia que los datos sean linealmente separables por diagnóstico. En cuanto a la distribución, algunas densidades se ajustan razonablemente a la curva gaussiana aunque posteriormente lo comprobaremos realizando ciertos tests de normalidad.

Para realizar la clasificación debemos seguir ciertos pasos:

1.Calcular la función discriminante lineal L(Z) Como el dataset está desbalanceado (mayor número de pacientes sanos), debemos ajustar las probabilidades a priori.

Probabilidad a priori de paciente sano : $540/615 = 31/36$

Probabilidad a priori de paciente enfermo: $1 - 31/36 = 5/36$

```r
library('MASS')
d$Diagnosis<-ifelse(d$Diagnosis == "Sano",0,1)
d$Diagnosis<- as.factor(d$Diagnosis)
d$Sex<-ifelse(d$Sex == "m",0,1)
LDA<-lda(d[,2:13],d[,1],prior=c(31/36,5/36))
#LDA
```

Las medias pueden confundir a nuestro modelo ya que son muy afectadas por los valores atipicos

Almacenamos los coeficientes de la FDL y construimos L

```
a <- LDA$scaling
L <- function(z) sum(a*z)
```

```
m_sano <- L(LDA$means[1,])
print(paste('media sano',m_sano))
```

```
## [1] "media sano 2.08147831381127"
```

```
m_enfermo <- L(LDA$means[2,])
print(paste('media enfermo',m_enfermo))
```

```
## [1] "media enfermo 5.42224535764278"
```

Para clasificar a un individuo, nos basaremos en la distancia mínima de su función discriminante a las medias de las funciones discriminantes según el diagnóstico.

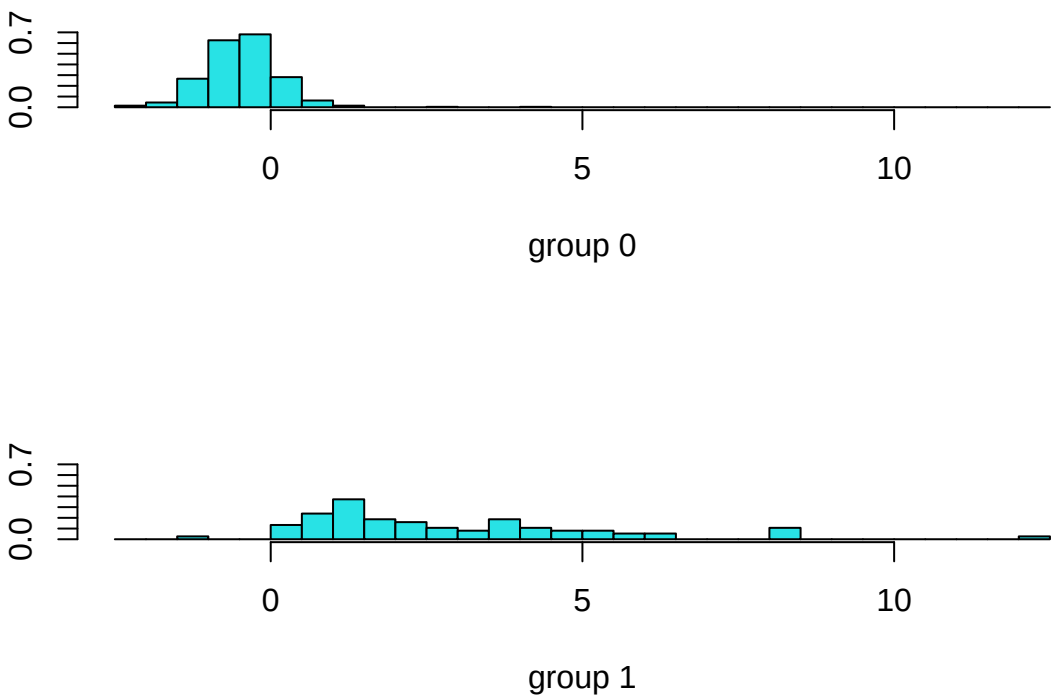
2 PASO: CALCULAR K (FRONTERA ENTRE CLASES) La frontera define un hiperplano y viene dado por la media de las proyecciones de la medias.

```
K <- (m_sano + m_enfermo)/2
print(paste('K =', K))
```

```
## [1] "K = 3.75186183572703"
```

3 PASO : CALCULAR LA PROYECCION DEL INDIVIDUO A CLASIFICAR Por lo tanto, la regla de decisión óptima según este criterio sería: Si $L(z) > K$ se clasifica como paciente con hepatitis Si $L(z) < K$ como paciente ssano. Podemos calcular las proyecciones de los pacientes muestrales haciendo:

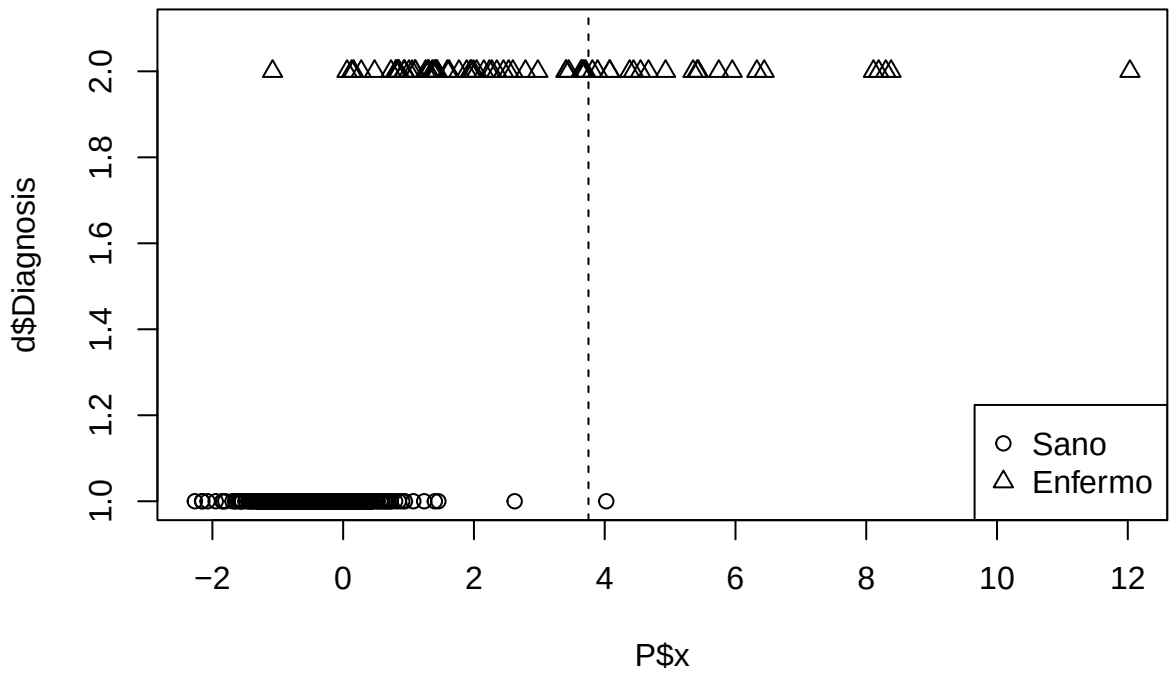
```
P<-predict(LDA,d[,2:13])
ldahist(P$x, g=d$Diagnosis)
```



Vemos que existe cierta solapación de valroe entre el grupo 0 (pacientes sanos) y el grupo 2 (pacientes enfermos). Podemos observar una mayor dispersión en los valores del grupo de los pacientes con hepatitis, mientras que los pacientes sanos se encuentran más concentrados y se asemejan a una distribución normal.

Podemos dibujar la frontera y los valores de la función discriminante para cada paciente separándolo según su diagnóstico

```
plot(P$x, d$Diagnosis, ylim=c(1, 2.1), pch = as.integer(d$Diagnosis))
abline(v=K, lty = 2)
legend('bottomright', legend = c('Sano','Enfermo'), pch=1:2)
```



Observamos ciertos valores extremos tanto en pacientes sanos como enfermos. Recordamos que este método se basa en las medias de las observaciones.La media se ve muy afectada por dichos valores extremos y estos, puede estar afectando muy negativamente al modelo. De ahí la importancia de una tendencia normal en los datos.

Podemos obtener una matriz de confusión de manera que podemos observar la eficacia de nuestro modelo y concluir si modeliza bien nuestros datos.

Resumen	Prediction: Positive (PP)	Prediction: Negative (PN)	Total
Real value: Positive	True Positive (TP)	False Negative (FN)	P
Real value: Negative	False Positive (FP)	True Negative (TN)	N
Total	PP	PN	Total Population (P+N)

```
table(d[,1],P$class)
```

```
##
##       0   1
##   0 538   2
##   1  31  44
```

Resumen	Clasificados en 0 (Sano)	Clasificados en 1 (Hepatitis)	Total
Grupo verdadero: 0 (Sano)	538	2	540

Resumen	Clasificados en 0 (Sano)	Clasificados en 1 (Hepatitis)	Total
Grupo verdadero: 1 (Hepatitis)	31	44	75
Total	569	46	615

Estas estimaciones suelen dar valores ligeramente optimistas respecto a los valores reales ya que al clasificar los pacientes, se ha usado información proporcionada por él mismo. Por ello, es recomendable utilizar **validación cruzada**.La validación cruzada es una técnica que consiste en clasificar pacientes sin haber utilizado su propia información, de manera que obtengamos una clasificación insesgada.

```
LDACV<-lda(d[,2:13],d[,1],prior=c(31/36,5/36),CV=TRUE)
table(d[,1],LDACV$class)
```

```
##
##       0   1
##   0 537   3
##   1  34  41
```

Resumen	Clasificados en 0 (Sano)	Clasificados en 1 (Hepatitis)	Total
Grupo verdadero: 0 (Sano)	537	3	540
Grupo verdadero: 1 (Hepatitis)	34	41	75
Total	571	44	615

Podemos analizar las métrica de bondad del modelo calculadas a partir de nuestra matriz de confusión. Son un aspecto clave para determinar la naturaleza del modelo.

```
sensibilidad <- table(d[,1],LDACV$class)[1] / (table(d[,1],LDACV$class)[1] + table(d[,1],LDACV$class)[3]
sensibilidad
```

```
## [1] 0.9944444
```

```
especifidad <- table(d[,1],LDACV$class)[4] / (table(d[,1],LDACV$class)[4] + table(d[,1],LDACV$class)[2]
especifidad
```

```
## [1] 0.5466667
```

```
acc <- (table(d[,1],LDACV$class)[1] + table(d[,1],LDACV$class)[4]) / (table(d[,1],LDACV$class)[1]+table
acc
```

```
## [1] 0.9398374
```

Sensibilidad: **99.44%**

Especifidad: **54.67%**

Eficiencia: **94%**

Conclusiones LDA

- **Alta Tasa de Verdaderos Positivos para Sanos**:

  El modelo ha clasificado correctamente a 538 de 540 pacientes sanos. Esto sugiere que el modelo es muy efectivo para identificar a los pacientes que no tienen hepatitis.

- **Tasa Moderada de Verdaderos Negativos para Hepatitis:**

  De los 75 pacientes con hepatitis, el modelo identificó correctamente a 44 como tales. Esto indica la existencia de un margen de mejora en la sensibilidad del modelo para detectar casos reales de hepatitis.

- **Alta eficiencia y sensibilidad**:

  Estas dos medidas de bondad han obtenido resultados bastante esperanzadores.Sin embargo, estas métricas se ven muy influenciadas por el desbalanceo del modelo y nos dan una falsa realidad del modelo

- **Baja especifidad**

El resultado de esta métrica es el principal indicador de las restricciones que supone la linealidad en nuestro modelo y de la dificultad de la detección de los casos de hepatitis.

La detección de los casos de hepatitis es crucial en nuestro contexto, pues no podemos pasar por alto casos de hepatitis.La incapacidad de realizar esta tarea con cierta bondad, nos lleva a rehuir de este modelo.

QDA

El Análisis Discriminante Cuadrático (**QDA**) es similar al Análisis Discriminante Lineal (**LDA**) en que ambos buscan definir límites de decisión entre las clases de una manera que maximice la distancia entre las medias de las clases y minimice la variación dentro de cada clase. Sin embargo, **QDA y LDA difieren** significativamente en la forma en que modelan estos límites. Mientas que LDA asume que las clase comparten la misma matriz de covarianza, QDA permite que cada clase tenga su propia matriz de covarianza, estableciendo límites de decisión cuadráticos.

Para realizar el QDA hacemos:

```
QDA<-qda(d[,2:13],d[,1],prior=c(31/36,5/36))
#QDA
```

Para obtener los coeficientes de la QDF que convierten a los datos en esféricos y las constantes, debemos teclear repectivamente:

```
#QDA$scaling
#QDA$ldet
```

Para obtener predicciones bassadas en estas funciones:

```
P<-predict(QDA,d[,2:13])
table(d[,1],P$class)
```

```
##
##       0   1
##   0 526  14
##   1  21  54
```

Al igual que en LDA, debemos aplicar la validación cruzada para obtener un resultado más realista

Resumen	Prediction: Positive (PP)	Prediction: Negative (PN)	Total
Real value: Positive	True Positive (TP)	False Negative (FN)	P
Real value: Negative	False Positive (FP)	True Negative (TN)	N
Total	PP	PN	Total Population (P+N)

```
QDACV<-qda(d[,2:13],d[,1],prior=c(31/36,5/36),CV=TRUE)
table(d[,1],QDACV$class)
```

```
##
##       0   1
##   0 525  15
##   1  22  53
```

Resumen	Clasificados en 0 (Sano)	Clasificados en 1 (Hepatitis)	Total
Grupo verdadero: 0 (Sano)	525	15	540
Grupo verdadero: 1 (Hepatitis)	22	53	75
Total	547	68	615

```
sensibilidad <- table(d[,1],QDACV$class)[1] / (table(d[,1],QDACV$class)[1] + table(d[,1],QDACV$class)[3]
sensibilidad
```

```
## [1] 0.9722222
```

```
especifidad <- table(d[,1],QDACV$class)[4] / (table(d[,1],QDACV$class)[4] + table(d[,1],QDACV$class)[2]
especifidad
```

```
## [1] 0.7066667
```

```
acc <- (table(d[,1],QDACV$class)[1] + table(d[,1],QDACV$class)[4]) / (table(d[,1],QDACV$class)[1]+table
acc
```

```
## [1] 0.9398374
```

Sensibilidad:**97.22%**

Especificdad: **70.67%**

Eficiencia:**0.94%**

Conclusiones QDA respecto a LDA

1. **Mejora de la especifidad(capacidad de identificar correctamente los casos de hepatitis)**

   QDA parece tener una mejor especifidad que LDA, ya que identifica correctamente a más pacientes con hepatitis (53 vs. 44) obteniendo una mejora del 16%.

2. **Sensibilidad (capacidad de identificar correctamente los casos de pacientes sanos)**

   LDA tiene una tasa mucho más baja de falsos negativos (2 vs 15), indicando que es menos probable que clasifique incorrectamente a un paciente sano como enfermo.

3. **Eficiencia (capacidad de identificar correctamente el diagnóstico de los pacientes)**

   En el cómputo global apenas ha sufrido cambios.Sin embargo, ha habido un nuevo compromiso entre las falsas alarmas y los pacientes enfermos no detectados

Si el objetivo es minimizar el número de casos de hepatitis no detectados (falsos positivos), **QDA** podría ser preferible debido a su mayor sensibilidad. Sin embargo, **si es crítico minimizar la tasa de falsos negativos**(por ejemplo, para evitar tratamientos innecesarios o ansiedad en pacientes sanos), **LDA** podría ser más adecuado.

Comprobaciones Como hemos comentado anteriormente, LDA funciona de manera correcta si las matrices de covarianzas teóricas son iguales. Por otra parte, QDA funciona bien si los datos son normales en cada grupo.

Observemos las matrices de covarianzas según el diagnóstico

```
d1<-d[d$Diagnosis==0,2:13]
d2<-d[d$Diagnosis==1,2:13]
S1 <- cov(d1)
S2 <- cov(d2)
View(S1)
View(S2)
```

Es más que evidente que las matrices son totalmente diferentes

Vamos a realizar un test de normalidad según el diagnóstico para cada variable

```
library('mvnormtest')
mshapiro.test(t(d[1:540,2:13]))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  Z
## W = 0.52113, p-value < 2.2e-16
```

```
mshapiro.test(t(d[541:615,2:13]))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  Z
## W = 0.40496, p-value = 7.106e-16
```

Ambas clases no pasan el tets de normalidad.

Por ende,hemos comprobado que ninguno de los supuesto iniciales se cumplen.

Regresión Logística

El Análisis Discriminante Lineal (LDA) y el Análisis Discriminante Cuadrático (QDA) son métodos estadísticos útiles para la clasificación, especialmente cuando las clases son bien diferenciadas (no es nuestro caso). Además, estos métodos tienen supuestos importantes que afectan su efectividad. Ante las limitaciones de LDA y QDA mencionadas, la Regresión Logística se presenta como una alternativa robusta.

La Regresión Logística ofrece una solución más general que puede adaptarse mejor a las irregularidades comunes en los datos reales. Su capacidad para manejar no normalidad, proporciona una interpretación clara, y su flexibilidad la convierten en una técnica robusta y preferible en muchos casos de estudio donde la clasificación precisa es crucial.

Para calcular el modelo de regresión logística en R, hacemos:

```
mylogit <- glm(Diagnosis ~., data = d, family = "binomial")
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

Nos encontramos con un error a la hora de calcula el modelo logístico

Tras cierto investigación online, este aviso se debe principalmente a dos razones:

El mensaje de advertencia "glm.fit: fitted probabilities numerically 0 or 1 occurred" se refiere a un problema común en modelos de regresión logística, este sucede cuando algunas de las probabilidades predichas por el modelo son esencialmente 0 o 1. Este fenómeno, se denomina separación perfecta, y suele ser causada por:

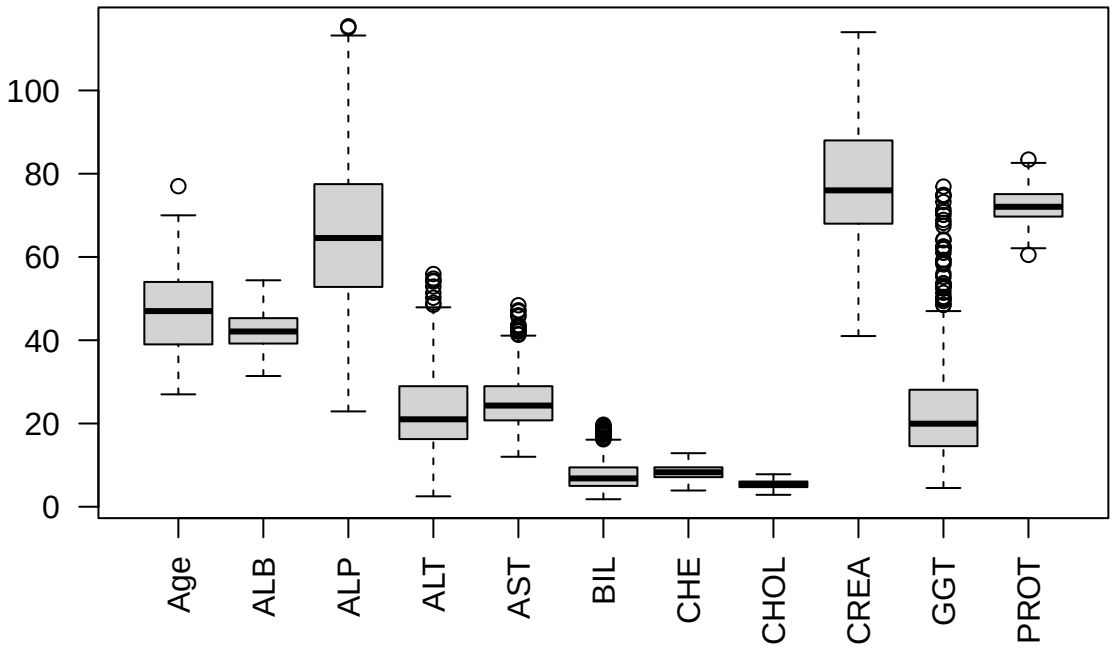
- Valores atípicos influyentes
- Falta de observaciones en alguna de las clases

Ambos casos se cumplen, por lo que nos vemos obligados a transformar y filtrar el dataset para ver si llegamos a la solución del problema

Para comprobar si este es el problema, vamos a reducir los valores atípicos y filtrar los datos en base al rango intercuartílico

```
Q1 <- apply(d[,4:13], 2, quantile, probs = 0.25)
Q3 <- apply(d[,4:13], 2, quantile, probs = 0.75)
IQR <- Q3 - Q1
# Calcula los límites para identificar valores atípicos para cada variable
lim_inf <- Q1 - 1.5 * IQR
lim_sup <- Q3 + 1.5 * IQR
# Identifica los valores atípicos para cada variable numérica
outliers <- sapply(4:13,
                   function(x) d[[x]] < lim_inf[x-3]
                   | d[[x]] > lim_sup[x-3])
# Elimina las filas que contienen valores atípicos en al menos una variable numérica
datos_sin_outliers <- d[!apply(outliers, 1, any),]
```

```
boxplot(datos_sin_outliers[,-c(1,3)], las=2)
```



Podemos observar que hemos reducido en gran medida los valores atípicos de nuetro dataset

Por último, debemos balancear nuestro dataset (evitar resultados sesgados), es decir igualar el número de datos de ambas clases. Existen dos opciones:

- Reducir el número de observaciones de la clases dominante (pacientes sanos)
- Aumentar el número de observacioness de la clase minoritaria (pacientes con hepatitis)

En nuestro caso al no tener un dataset demasiado extenso, optaremos por la segunda opción.Para ello,optaremoss por un algoritmo denomido **SMOTE**.

SMOTE (Synthetic Minority Over-sampling Technique) es una técnica de sobremuestreo desarrollada para abordar el problema del desequilibrio de clases en conjuntos de datos utilizados para entrenar modelos de aprendizaje automático. En R, se encuentra implementada en la librería smotefamily.

```
library(smotefamily)
d_balance <- SMOTE(datos_sin_outliers[,-1],datos_sin_outliers$Diagnosis,K=5) # nos devuelve una lista c
d_balance <- d_balance$data # almacenamos el dataset balanceado
```

Para comprobar que el dataset ha sido balanceado podemos calcular la proporcion de cada clases

```
prop.table(table(d_balance$class))
```

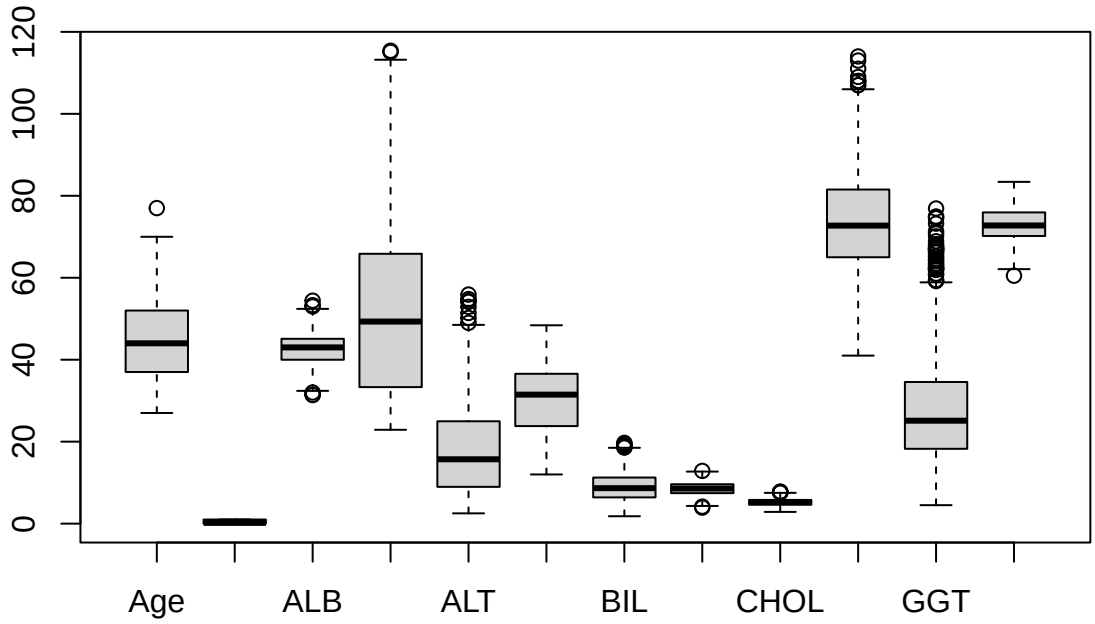
```
##
##         0         1
## 0.5034722 0.4965278
```

```r
nrow(d_balance)
```

```
## [1] 864
```

Observamos que las proporciones son casi idénticas, y nuestro dataset ahora cuenta con 864 observaciones

```r
boxplot(d_balance[,-13])
```



Para obtener un modelo más robusto y evitar el sobreajuste, debemos realizar un data split

Dividimos los datos en entrenamiento (80%) y test (20%). Los datos de entrenamiento serán utilizados para la construcción del modelo, y los datos de test serán utilizados para medir la bondad de cara a nuevos datos.

```r
d_balance$class <- as.factor(d_balance$class)
set.seed(246)
indices_entrenamiento <- sample(1:nrow(d_balance), 0.8*nrow(d_balance))
datos_entrenamiento <- d_balance[indices_entrenamiento,]
#View(datos_entrenamiento)
datos_prueba <- d_balance[-indices_entrenamiento,]
```

Datos de entrenamiento: 691 observaciones

Datos de prueba: 173 observaciones que difieren de los datos de entrenamiento

```r
mylogit <- glm(class~., data = datos_entrenamiento, family = "binomial")
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

A nuestra sorpresa, el error no se ha conseguido resolver. Como solución, podría basar el modelo en las variables más influyentes de las componentes principales:

```r
mylogit <- glm(class~ ALB+ALP+ALT+AST+BIL+CHE+CHOL+GGT+PROT+Sex , data = datos_entrenamiento, family = 
summary(mylogit)
```

```
##
## Call:
## glm(formula = class ~ ALB + ALP + ALT + AST + BIL + CHE + CHOL +
##     GGT + PROT + Sex, family = "binomial", data = datos_entrenamiento)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.25992    5.94210   0.549  0.58327
## ALB          0.10985    0.08965   1.225  0.22048
## ALP         -0.19457    0.02893  -6.726 1.74e-11 ***
## ALT         -0.26192    0.04653  -5.629 1.81e-08 ***
## AST          0.41521    0.06490   6.397 1.58e-10 ***
## BIL          0.18187    0.09148   1.988  0.04681 *
## CHE          1.05204    0.25568   4.115 3.88e-05 ***
## CHOL        -0.94075    0.35798  -2.628  0.00859 **
## GGT          0.05854    0.02162   2.708  0.00677 **
## PROT        -0.18105    0.06442  -2.811  0.00495 **
## Sex          1.67479    0.77934   2.149  0.03164 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 957.93  on 690  degrees of freedom
## Residual deviance: 134.23  on 680  degrees of freedom
## AIC: 156.23
##
## Number of Fisher Scoring iterations: 9
```

Si observamos los p-valores correspondientes a cada variable, observamos que no todos son significativos (p-valor inferior a 0.05). Esto nos sugiere que el modelo va a ser reducible. Para comprobarlo, aplicamos los tres métodos de selección de regresores.

Hemos desactiva la traza de los métodos de selección de regresores mediante trace=FALSE para no afectar a la estética del documento.

```r
modelo_backward <- step(mylogit, direction = "backward", trace = FALSE)
```

```r
modelo_nulo <- glm(class ~ 1, data = datos_entrenamiento, family = "binomial", trace = FALSE)
modelo_forward <- step(modelo_nulo, scope = formula(mylogit), direction = "forward",trace = FALSE)
```

```r
modelo_stepwise <- step(modelo_nulo, scope = formula(mylogit), direction = "both",trace = FALSE)
```

Los tres métodos convergen en un mismo modelo, asi que este será nuestro modelo de referencia. La bondad del ajuste de este modelo lo medimos en funcion criterio de Akaike, siendo mejor el ajuste cuanto menor valor tenga AIC.

```r
mylogit$aic
```

```
## [1] 156.2257
```

```r
modelo_backward$aic
```

```
## [1] 155.7743
```

La mejora entre el modelo completo y el resto de modelos es bastante poco relevante.También podemos basar la elección del modelo en la devianza (cuanto menor, mejor)

```r
mylogit$deviance
```

```
## [1] 134.2257
```

```r
modelo_backward$deviance
```

```
## [1] 135.7743
```

La devianza del modelo completo es algo mejor. En cómputo global, el modelo completo es ligeramente mejor, así que lo tomaremos como modelo de referencia.

Antes de realizar predicciónes en base a nuestro modelo, **debemos validarlo**

Validación del modelo

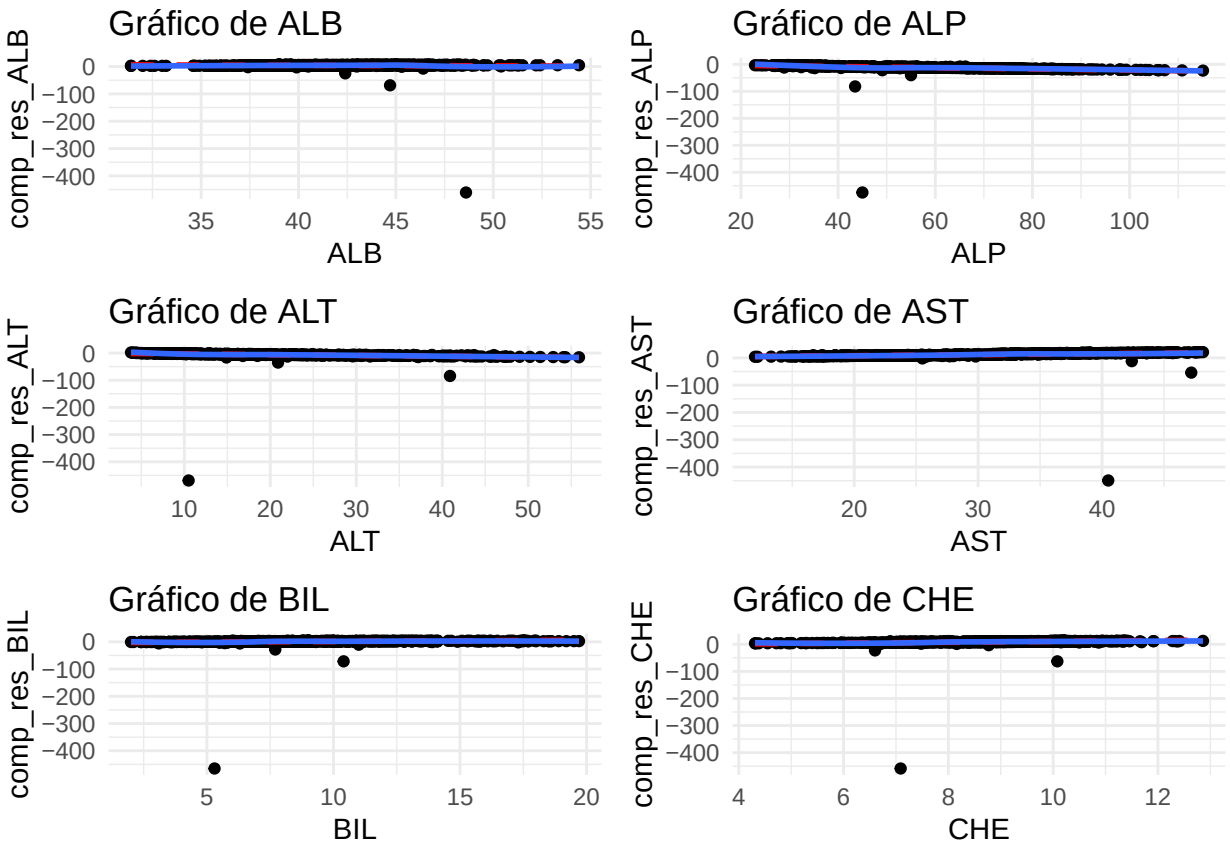
Debemos verificar:

Linealidad

```r
mydata2 <- datos_entrenamiento
# Lista de nombres de las variables
variables <- c("ALB", "ALP", "ALT", "AST", "BIL",
               "CHE", "CHOL", "GGT", "Sex", "PROT")

# Crear los componentes residuales
comp_res <- list()
for (var in variables) {
  comp_res[[var]] <- coef(mylogit)[var] * datos_entrenamiento[[var]] + mylogit$residuals
}

# Convertir la lista en un data frame para trabajar más fácilmente
comp_res_df <- as.data.frame(comp_res)

# Asignar los componentes residuales al dataset mydata2
for (var in variables) {
  mydata2[[paste0("comp_res_", var)]] <- comp_res[[var]]
}
```

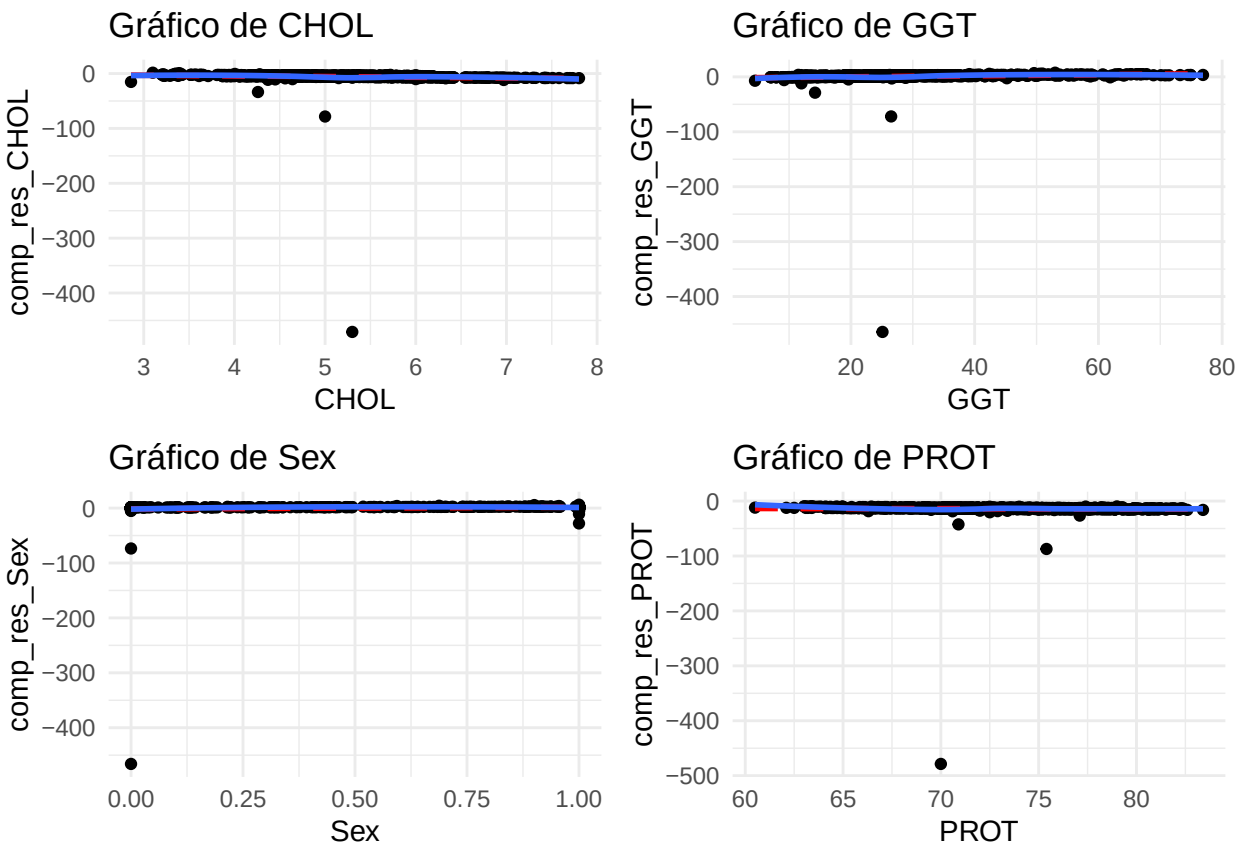
```r
# Lista de variables y componentes residuales
variables <- c("ALB", "ALP", "ALT", "AST", "BIL",
               "CHE", "CHOL", "GGT", "Sex", "PROT")

# Crear una lista vacía para almacenar los gráficos
plots <- list()

# Generar gráficos con un bucle
for (var in variables) {
  plots[[var]] <- ggplot(data = mydata2, aes_string(x = var, y = paste0("comp_res_", var))) +
    geom_point() +
    geom_smooth(color = "red", method = "lm", linetype = 2, se = FALSE) +
    geom_smooth(se = FALSE) +
    labs(title = paste("Gráfico de", var), x = var, y = paste0("comp_res_", var)) +
    theme_minimal()
}

# Organizar los gráficos en grids usando cowplot
cowplot::plot_grid(plotlist = plots[1:6], ncol = 2, nrow = 3) # Primer conjunto de gráficos
```

```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```




```
cowplot::plot_grid(plotlist = plots[7:10], ncol = 2, nrow = 2) # Segundo conjunto de gráficos
```

```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

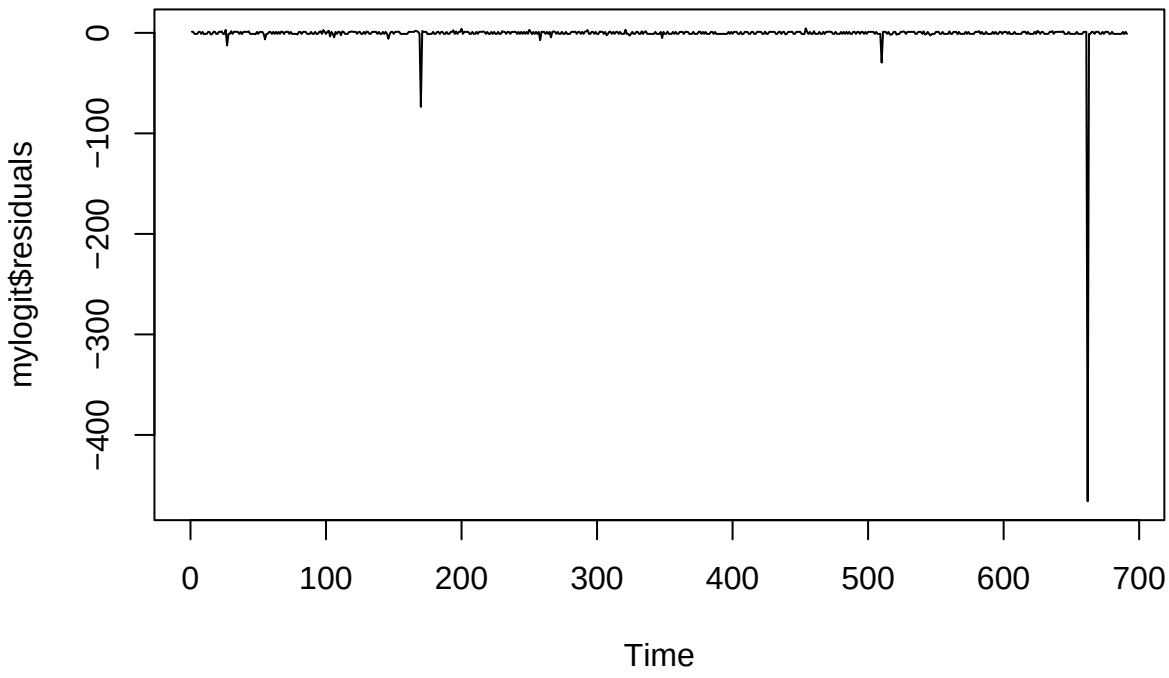


No podemos observar violaciones de la linealidad (línea azul se asemeja a una recta)

Independencia de las observaciones

Para analizar la independencia, podemos hacer un gráfico temporal de los residuos

```
ts.plot(mylogit$residuals)
```



No observamos patrones ni tendencias que puedan suponer dependencia entre las observaciones

Inexistencia de multicolinealidad entre predictores

Para comprobarlo calculamos los VIFs, valores superiores a 7 son indicativos de existencia de multicolinealidad

```
rms::vif(mylogit)
```

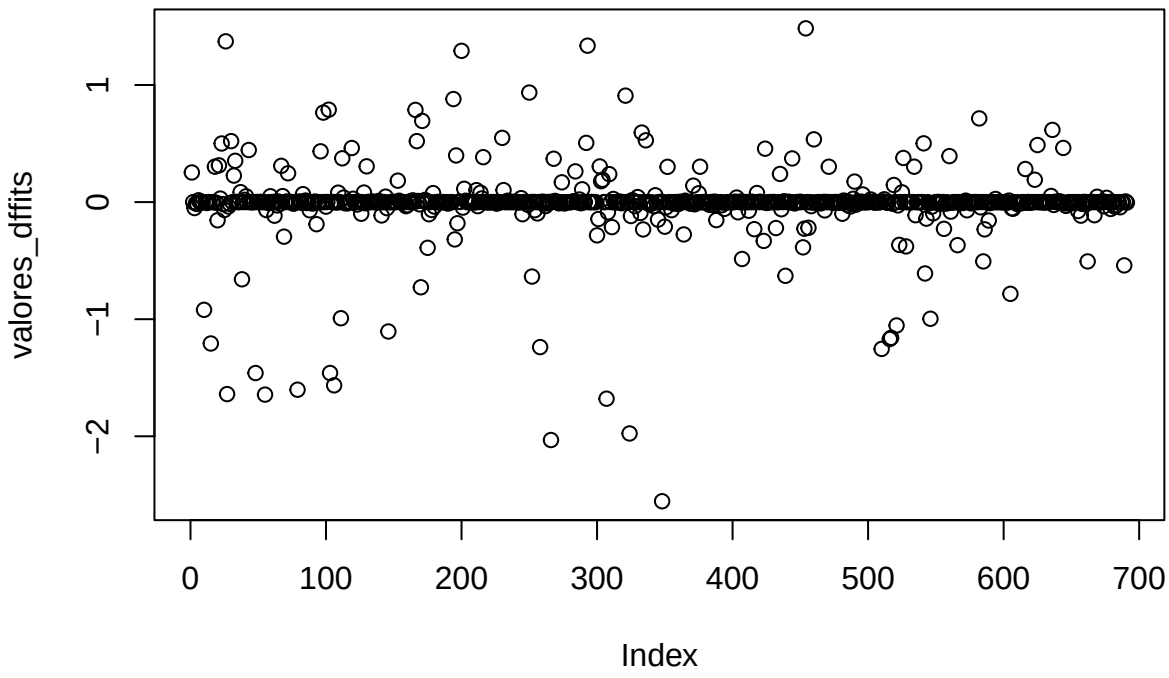
```
##      ALB      ALP      ALT      AST      BIL      CHE     CHOL      GGT
## 1.522195 2.478988 5.832663 3.782699 1.509534 2.783087 1.952013 1.872180
##     PROT      Sex
## 2.305369 2.551959
```

Ninguna de las variables supera a 7 aunque cabe destacar que ALT tiene un valor inferior pero cercano.

Inexsitencia de observaciones influyentes

Para ello, vamos a basarnos en los valores **DFFITS**. A cada observación le corresponde un **DFFITS** midiendo cuánto cambiaría la predicción para dicho punto

```
valores_dffits <- dffits(mylogit)
valores_dfbetas <- as.data.frame(dfbetas(mylogit))
plot(valores_dffits)
```



No obseravamos ningún valor demasiado alejado por lo que podemos concluir que nuestro modelo es válido tras haber superado todos los requisitos.

CLASIFICACIÓN MEDIANTE REGRESIÓN LOGÍSTICA

Para terminar con el estudio de este dataset, vamos a aplicar este modelo a una tarea de clasificación binaria. Podemos usar como criterio razonable que si dicha probabilidad es de al menos 0.5, entonces clasificaremos la observación en el grupo Y = 1, mientras que si la probabilidad es menor de 0.5 clasificaremos la observación en el grupo Y = 0.

```
predic_grupos <- ifelse(mylogit$fitted.values >= 0.5, 1, 0)
```

Construimos la matriz de confusión

```
matriz_confusion <- table(datos_entrenamiento$class, predic_grupos,
dnn = c("observado", "predicciones"))
matriz_confusion
```

```
##          predicciones
## observado   0   1
##         0 330  15
##         1  10 336
```

Calculamos las metricas de bondad:

```
VP <- matriz_confusion[2, 2]
FN <- matriz_confusion[2, 1]
VN <- matriz_confusion[1, 1]
FP <- matriz_confusion[1, 2]
sensibilidad <- VP/(VP+FN)
sensibilidad
```

```
## [1] 0.9710983
```

```
especificidad <- VN/(VN+FP)
especificidad
```

```
## [1] 0.9565217
```

```
accuracy <- (VP+VN)/(VP+FP+VN+FN)
accuracy
```

```
## [1] 0.9638205
```

Estos resultados son muy buenos y nos indican una gran bondad del modelo en parámetros generales. Con la aplicación de la regresión logística, hemos conseguido evitar las falsas alarmas y la incapacidad de detectar a pacientes con hepatitis, aspectos que dominaban en los modelos anteriores.

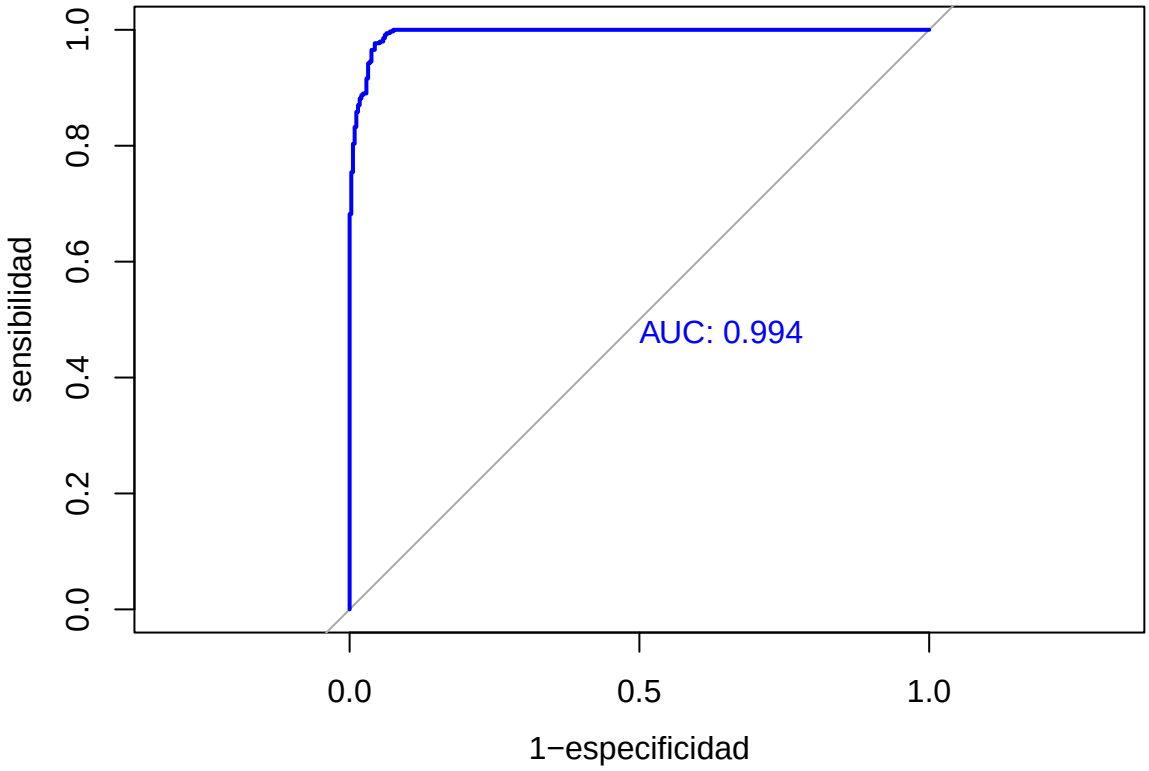
Curva ROC y AUC

```
library("pROC")
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
roc(datos_entrenamiento$class, mylogit$fitted.values, plot = TRUE,
legacy.axes = TRUE, percent = FALSE,
xlab = "1-especificidad", ylab = "sensibilidad",
col = "blue", lwd = 2, print.auc = TRUE)
```

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

## Setting direction: controls < cases
```




```
##
## Call:
## roc.default(response = datos_entrenamiento$class, predictor = mylogit$fitted.values,     percent = FA
##
## Data: mylogit$fitted.values in 345 controls (datos_entrenamiento$class 0) < 346 cases (datos_entrenam
## Area under the curve: 0.994
```

Obsérvese que en este caso obtenemos un AUC cercano a 0.995, indicativo de que el método clasificador (en este caso la Regresión Logística) tiene un rendimiento sorprendemente casi perfecto . Un clasificador perfecto daría un valor de **AUC igual a 1**, por lo que puede que nuestro modelo esté sobreajustado. Para comprobar si existe sobreajuste, vamos a realizar la predicción de los datos test, es decir, aquellos que no han sido utilizados en la construcción del modelo.

```
predic_grupos <- ifelse(predict(mylogit, newdata=datos_prueba) >= 0.5,
1, 0)
prediccion<-predict(mylogit, newdata=datos_prueba, type = "response")
 d_predict<-cbind(datos_prueba,prediccion)
```

```
matriz_confusion <- table(datos_prueba$class, predic_grupos,
dnn = c("observado", "predicciones"))
matriz_confusion
```

```
##          predicciones
## observado  0  1
##         0 90  0
##         1  2 81
```

```r
accuracy <- sum(diag(matriz_confusion)) / sum(matriz_confusion)
 accuracy
```

```
## [1] 0.9884393
```

La eficacia del modelo es sorprendentemente mejor que la obtenida por la predicción de los datos del propio modelo. Esto nos permite concluir que este modelo es muy robusto y de gran bondad. Sin duda, es el modelo adecuado para un diagnóstico consolidado. No obstante, debemos tener en cuenta las transformaciones que hemos realizado para fundar este modelo.

Conclusión final

El estudio realizado sobre la hepatitis C ha destacado la importancia crucial del análisis de datos clínicos para mejorar la detección temprana y el diagnóstico de esta enfermedad. A través del uso de técnicas estadísticas avanzadas como el Análisis de Componentes Principales (PCA) y modelos de clasificación supervisada, hemos logrado identificar patrones significativos en los datos que ayudan a diferenciar entre pacientes sanos y enfermos. Estos modelos han demostrado ser herramientas valiosas para la identificación temprana de la hepatitis C, lo que es esencial para prevenir complicaciones graves como la cirrosis y el cáncer de hígado.

Desafíos en el Análisis de Datos

Uno de los principales desafíos enfrentados durante el análisis ha sido el desequilibrio en el número de observaciones entre pacientes sanos y enfermos. Este desequilibrio puede sesgar los modelos de clasificación, afectando su capacidad para predecir correctamente a los pacientes con hepatitis C. Para abordar este problema, se han aplicado técnicas de sobremuestreo, como el algoritmo SMOTE (Synthetic Minority Oversampling Technique), que generan nuevas observaciones sintéticas para el grupo minoritario (pacientes con hepatitis C), mejorando así la precisión de los modelos.

Además, la presencia de valores atípicos en algunas variables ha requerido un cuidadoso manejo. Estos valores pueden influir negativamente en la precisión de los modelos, por lo que se han aplicado técnicas de detección y limpieza de datos para minimizar su impacto.

Selección de Variables Relevantes

La selección de variables relevantes ha sido otro aspecto crucial en el análisis. Dado que el conjunto de datos incluye múltiples variables clínicas y demográficas, se ha realizado un análisis exhaustivo para identificar aquellas que tienen un impacto significativo en la predicción de la hepatitis C. Variables como la alanina transaminasa (ALT), la aspartato aminotransferasa (AST), la bilirrubina (BIL) y la gamma-glutamil transferasa (GGT) han demostrado ser indicadores importantes de riesgo. Estas variables han sido seleccionadas cuidadosamente para mejorar la precisión de los modelos predictivos.

Implicaciones Clínicas

En resumen, el análisis de datos clínicos ha demostrado ser una herramienta poderosa en la lucha contra la hepatitis C. A través del uso de técnicas estadísticas avanzadas y la selección cuidadosa de variables relevantes, podemos mejorar significativamente la detección temprana y el diagnóstico de esta enfermedad, lo que a su vez puede mejorar las tasas de curación y reducir el riesgo de complicaciones graves.