PEC4: Final assessment

Marco Russo - Silvia Gamundi Sumando

Enero, 2025

Contents

Información del Estudiante

Nombre	Marco Russo
Email	mrussorb@uoc.edu
GitHub	https://github.com/marcusRB/uoc-ub-scientific-programming
LinkedIn	https://www.linkedin.com/in/marcusrb/
Fecha	January 13, 2026

Nombre	Silvia Gamundi Sumando
Email	sgamundis@uoc.edu
GitHub	https://github.com/
LinkedIn	https://www.linkedin.com/in/
Fecha	January 13, 2026

1 Contexto y objetivo del estudio. Datos (1 punto)

El dataset elegido es `Bacteremia` del autor Heinze, G. (2023). Bacteremia [Data set]. In PLoS One (Version S2, Vol. 9, Number 9, p. e106765). Zenodo. https://doi.org/10.5281/zenodo.7554815 [1]

El resto de información ha sido extracto de la fuente oficial:

> The data set consists of 14,691 observations from different patients with the clinical suspicion to suffer from bacteremia, for whom a a blood culture analysis was performed at the Vienna General Hospital, Austria, between January 2006 and December 2010.
>
> It contains the results of the blood culture analysis for bacteremia and the values of 51 potential predictors of bacteremia. To protect data privacy our version of this data was slightly modified compared to the original version, and this modified version was cleared by the Medical University of Vienna for public use (DC 2019-0054).
>
> The original version of the data set was used by Ratzinger et al (2014) to develop a model for screening bacteremic patients based on highly standardizable laboratory variables. This public version has been used by Gregorich et al (2021).

Basada en la descripción oficial del mismo, se indican que existen 14,691 observaciones de diferentes pacientes que podrían ser afectos de `bacteriemia`. De la información disponible en Wikipedia:Bacteriemia [^2], la bacteriemia es la presencia de bacterias en la sangre. La sangre es normalmente un medio estéril, por lo tanto la detección de bacterias es indicativa de infección.

Es importante entender este punto respecto al **diagnóstico**, muchas personas se recuperan completamente de la bacteriemia. Sin embargo, la bacteriemia es grave y puede provocar sepsis. Cuando tiene sepsis, el daño a los órganos principales puede ser irreversible.

Entre las **causas**, la entrada de bacterias en el torrente sanguíneo puede ser producto de una infección localizada (ej: neumonía, absceso en piel o mucosas), o por interrupción de la piel como barrera defensiva. Se destacan las intervenciones quirúrgicas, utilización de dispositivos invasivos (catéteres, sondas, asistencia mecánica respiratoria), heridas accidentales, o quemaduras.

La infección suele empezar en los pulmones, el tracto genitourinario, gastrointestinal o los tejidos blandos, entre ellos la piel de pacientes con úlceras. También puede ser secundaria a una intervención dental en pacientes de alto riesgo, especialmente en los que tienen prótesis intravasculares.

Respecto a las **consecuencias**, dependen del tipo de bacteria y el estado del paciente. La respuesta inmunológica a la infección puede causar sepsis y devenir en shock séptico. También puede ocurrir que la sangre transporte las bacterias a otros tejidos, que podrán ser infectados. Ejemplos incluyen endocarditis, osteomielitis, y meningitis. El tratamiento es fundamental para erradicar a las bacterias y requiere el uso de antibióticos por vía intravenosa.

Se dispone de un pequeño diccionario que incluye el significado de cada característica del dataset. Lo descargaremos y visualizaremos para entender mejor el contexto de cada característica del dataset.

[1] Heinze, G. (2023). Bacteremia [Data set]. In PLoS One (Version S2, Vol. 9, Number 9, p. e106765). Zenodo. https://doi.org/10.5281/zenodo.7554815

2 Prospección y preparación de los datos (2 puntos)

2.1 Descripción de los datos (1 punto)

Preparamos el entorno cargando el resto de librerías que serán útiles para realizar un análisis exploratorio de los datos.

Descargamos el dataset en un formato dataframe y comprobaremos su estructura.

```
# Define the URL and file path
url <- "https://zenodo.org/records/7554815/files/Bacteremia_public_S2.csv?download=1"
filename <- "data/bacteremia_dataset.csv"

# Create directory if it doesn't exist
if (!dir.exists("data")) {
  dir.create("data", recursive = TRUE)

  # Download the file
  download.file(url, destfile = filename, mode = "wb")
} else {
  sprintf("The dataset has been downloaded and is available in 'data' path")
}
```

```
## [1] "The dataset has been downloaded and is available in 'data' path"
```

```
# Read the CSV file
bacteremia_df <- read.csv(file = filename)
```

Finalmente mostramos los primeros datos y la naturaleza de las características.

```
# Mostramos los primeros datos con head()
head(bacteremia_df, 10)
```

```
##    ID SEX AGE   MCV  HGB  HCT PLT  MCH MCHC  RDW  MPV LYM MONO EOS BASO NT APTT
## 1   1   2  62  99.3 11.5 35.9 307 31.5 31.8 19.5 10.8 0.4  1.7 0.0  0.1 86 28.8
## 2   3   1  72  85.1 10.3 34.7 182 26.0 30.6 15.0  9.7 0.4  0.2 0.1  0.0 90 29.8
## 3   5   1  46  96.3  7.4 22.8  64 31.2 32.4 19.7 11.1 1.5  1.2 0.1  0.1 58 36.3
## 4   7   1  84  91.3 10.3 31.1 309 30.4 33.3 13.8  8.5 1.3  0.8 0.0  0.0 67 38.2
## 5   9   2  38  85.1 13.7 38.7 183 30.2 35.3 12.6 10.0 0.8  0.4 0.0  0.0 95 33.1
## 6  10   1  68 104.5 15.7 46.9 144 34.8 33.5 13.9 10.9 2.2  0.9 0.1  0.0 61 41.8
## 7  11   1  55  99.3 14.6 43.5 242 33.1 33.4 13.1 10.3 2.1  1.6 0.3  0.0 NA   NA
## 8  12   1  55  77.0 10.8 34.8  38 23.8 30.5 16.8   NA 0.4  0.1 0.1  0.0 93 36.3
## 9  13   1  67  95.3 10.9 30.4  88 33.6 35.3 13.3 10.7 0.4  0.2 0.0  0.0 57 33.8
## 10 19   2  52  83.0 10.3 30.1 105 28.6 34.1 13.2 11.3 0.9  0.9 0.3  0.1 69 28.1
##    FIB SODIUM POTASS   CA PHOS   MG CREA  BUN  HS GBIL   TP  ALB AMY PAMY LIP
## 1  578    137   3.88 2.29 1.20 0.66 0.65  5.7 5.3 0.59 67.0 36.7  30   16  10
## 2   NA    141     NA 2.21 0.58   NA 0.76 19.9  NA 0.48 65.3 37.4  NA   NA  NA
## 3  313    147   4.61 1.92 1.51 1.03 1.25 50.6  NA 8.42 40.5 22.1 146   NA  89
## 4  487    141   4.71 2.05 2.17 0.83 2.78 47.5 9.7 0.35 61.2 33.2  92   28  18
## 5  490    137     NA 2.34 0.97 0.74 0.65  8.5 3.0 0.42 78.4 43.8  84   50  50
## 6  400    141   4.41 2.08 0.99 0.56 0.82 15.3 5.5 2.40 57.5 30.1  95   57  25
## 7   NA    139   3.69   NA   NA   NA 1.21 13.0  NA 1.13   NA   NA 117   NA  73
## 8  413    142   4.67 2.31 1.16 0.87 1.77 29.8 6.2 0.45 70.8 43.6 177   43  30
```

```
## 9  431    143   2.35 2.10 0.51 0.36 1.00 15.0 4.7 1.21 67.4 35.4  NA   NA  NA
## 10 407    136   3.80 1.92 0.72 0.53 0.58 14.0 4.0 2.46 53.8 24.8  35   35  38
##     CHE  AP ASAT ALAT GGT  LDH   CK GLU TRIG CHOL   CRP     BASOR      EOSR
## 1  5.12  85   22   14  48  284   23 107  105  175  3.94 0.4132231 0.0000000
## 2  5.61  80   28   25  61   NA   36  84   NA   NA  1.42 0.0000000 0.8264463
## 3  2.52 119  124  135 134  696   40 107   NA   NA 12.09 0.5681818 0.5681818
## 4  4.10  94  774   72  23 1787 2422 105  134  141  3.78 0.0000000 0.0000000
## 5  6.91 108   35   22  72   NA   79  93  152  167 11.17 0.0000000 0.0000000
## 6  6.79  68   32   11  68  263   75  89   85  144  5.89 0.0000000 1.0000000
## 7    NA  51   29   20 138  303  230  91   NA   NA 17.84 0.0000000 2.3255814
## 8  7.40 153   26   32  96  181   87  96  129  156  1.29 0.0000000 5.5555556
## 9    NA 239   91   57 446  183   53  86   62  118  1.36        NA        NA
## 10 2.64 146   97  156 192  277   87 104  207  123  3.80 1.6666667 5.0000000
##         LYMR     MONOR  NEU     NEUR  PDW RBC   WBC BloodCulture
## 1   1.652893  7.024793 22.0 90.90909 10.6 3.7 24.10           no
## 2   3.305785  1.652893 11.4 94.21488 11.4 3.9 12.17           no
## 3   8.522727  6.818182 14.7 83.52273 14.1 2.5 17.45           no
## 4  11.016949  6.779661  9.7 82.20339  8.7 3.5 11.58           no
## 5   8.333333  4.166667  8.4 87.50000 12.2 4.4  9.86           no
## 6  22.000000  9.000000  6.8 68.00000 12.9 4.3  9.94           no
## 7  16.279070 12.403101  8.9 68.99225 12.5 4.5 13.06           no
## 8  22.222222  5.555556  1.2 66.66667   NA 4.7  1.78           no
## 9         NA        NA   NA       NA   NA  NA    NA          yes
## 10 15.000000 15.000000  3.8 63.33333 13.2 3.5  5.98           no
```

```
# Mostramos los últimos datos también con tail()
tail(bacteremia_df, 10)
```

```
##          ID SEX AGE  MCV  HGB  HCT PLT  MCH MCHC  RDW  MPV LYM MONO EOS BASO NT
## 14682 62410   1  72 91.5 11.5 33.7 126 31.3 34.1 14.0 10.1 0.7  1.1 0.1  0.0 61
## 14683 62411   1  30 81.9 14.7 41.2 255 29.4 35.9 12.3  9.5 1.9  1.1 0.0  0.0 84
## 14684 62417   1  63 89.7 13.0 38.0 157 30.8 34.4 13.7 10.2 0.3  0.8 0.0  0.0 60
## 14685 62420   1  28 87.5 10.4 29.8 217 31.0 35.3 12.4  9.7 1.4  2.0 0.1  0.0 95
## 14686 62432   1  37 82.0  9.0 25.9 145 28.6 34.6 16.4  9.9 0.0  0.3 0.0  0.0 44
## 14687 62436   2  44 97.4  7.9 22.5 248 33.6 34.4 13.3  9.8 1.3  0.4 0.1  0.0 NA
## 14688 62438   1  23 67.4 11.5 38.2  58 20.1 29.8 18.4   NA 0.6  0.5 0.0  0.0 NA
## 14689 62446   1  79 86.3  9.7 31.4 345 26.8 30.8 14.4  8.9 0.7  1.3 0.1  0.0 75
## 14690 62454   1  81 88.6 11.5 32.6 262 31.0 35.0 16.4 11.6 1.3  1.0 0.2  0.0 99
## 14691 62455   2  21 79.2 12.3 35.6 192 27.5 34.7 12.2  9.5 1.3  0.6 0.0  0.1 68
##       APTT FIB SODIUM POTASS   CA PHOS   MG CREA  BUN  HS GBIL   TP  ALB AMY
## 14682 45.4 483    136   4.10 2.09 0.84 0.93 0.83 10.3 2.9 1.41 55.2 30.6 143
## 14683 35.5 523    137   3.87 2.44 0.81 0.89 0.93 15.6 5.8 1.98 83.5 45.7  64
## 14684 34.9 667    135   3.90 1.91 0.91 0.79 1.20 27.2 6.0 0.93 57.5 30.0  44
## 14685 33.2 674    139   3.61 2.21 1.30 0.80 0.75 12.1 3.9 1.12 65.8 33.3  57
## 14686 44.1 457    138   5.59 2.17 2.76 1.03 4.10 63.1 5.8 3.06 58.0 32.1 535
## 14687   NA  NA    134   3.98 2.20 0.91 0.62 0.56  5.7 8.1 0.70 53.8 26.9  46
## 14688   NA  NA    137     NA 2.24 0.97   NA 0.64 12.6  NA   NA 66.3 42.8  31
## 14689 32.7  NA    137   4.02 2.32 0.81 0.92 1.65 23.4 5.3 0.78 72.2 32.9  62
## 14690 34.3 551    143   3.82 2.08 0.80 0.80 0.96 10.1 3.3 1.43 56.2 28.6  50
## 14691 43.0 347    135   3.49 2.22 1.02 0.78 0.69  8.4 5.1 1.76 72.9 38.4  32
##       PAMY LIP  CHE  AP ASAT ALAT GGT  LDH  CK GLU TRIG CHOL   CRP    BASOR
## 14682   NA  35 4.91  37   48   21  34  288 219 101   61   73 18.56 0.000000
## 14683   18  14 7.47  90   34   50  26  349 216  99   46  159  7.88 0.000000
## 14684   NA  12 3.59  36   19   15  28  156 144 160  123   96 21.05 0.000000
```

```
## 14685   35  50 3.59 111   38   36 187  370 379  77  162  162 11.85 0.000000
## 14686   14  NA 2.56  87   62   28 102  346 150 113   70   89 15.12 0.000000
## 14687   34 146 2.60  51   28   22 144  564  61  76  298   NA 26.80 0.000000
## 14688   NA  14   NA 155   60   72  36   NA  28  85   NA   NA  4.68 0.000000
## 14689   NA  15 4.32  71   56   14  34  272 274 113   NA   NA 12.76 0.000000
## 14690   32  55 2.89  68   28   24  88  304  24  88  117  115 11.18 0.000000
## 14691   18  33 5.22 352 2751 2487 332 2080  42  95   53  134  1.00 2.941176
##            EOSR      LYMR     MONOR  NEU     NEUR  PDW RBC   WBC BloodCulture
## 14682 0.8403361  5.882353  9.243697 10.0 84.03361 11.2 3.8 11.95           no
## 14683 0.0000000 13.013699  7.534247 11.6 79.45205 11.1 5.0 14.63           no
## 14684 0.0000000  7.894737 21.052632  2.7 71.05263 11.5 4.2  4.00           no
## 14685 1.0638298 14.893617 21.276596  5.9 62.76596 10.5 3.3  9.41           no
## 14686 0.0000000  0.000000  1.477833 20.0 98.52217 11.6 3.4 20.34           no
## 14687 2.0833333 27.083333  8.333333  3.0 62.50000 11.0 2.4  4.89           no
## 14688 0.0000000 10.714286  8.928571  4.5 80.35714   NA 5.7  5.62           no
## 14689 0.8771930  6.140351 11.403509  9.3 81.57895  9.3 3.8 11.39           no
## 14690 2.1276596 13.829787 10.638298  6.9 73.40426 14.1 3.7  9.30           no
## 14691 0.0000000 38.235294 17.647059  1.4 41.17647 10.8 4.4  3.46           no
```

Realizaremos un exploratorio genérico del dataset. Mostrando información básica del dataset, para pasar luego a los estadísticos básicos y comenzaremos a interactuar con las características luego.

Verificamos la estructura del juego de datos principal. Vemos el número de columnas que tenemos y ejemplos de los contenidos de las filas.

```
str(bacteremia_df)
```

```
## 'data.frame':    14691 obs. of  53 variables:
##  $ ID          : int  1 3 5 7 9 10 11 12 13 19 ...
##  $ SEX         : int  2 1 1 1 2 1 1 1 1 2 ...
##  $ AGE         : int  62 72 46 84 38 68 55 55 67 52 ...
##  $ MCV         : num  99.3 85.1 96.3 91.3 85.1 ...
##  $ HGB         : num  11.5 10.3 7.4 10.3 13.7 15.7 14.6 10.8 10.9 10.3 ...
##  $ HCT         : num  35.9 34.7 22.8 31.1 38.7 46.9 43.5 34.8 30.4 30.1 ...
##  $ PLT         : int  307 182 64 309 183 144 242 38 88 105 ...
##  $ MCH         : num  31.5 26 31.2 30.4 30.2 34.8 33.1 23.8 33.6 28.6 ...
##  $ MCHC        : num  31.8 30.6 32.4 33.3 35.3 33.5 33.4 30.5 35.3 34.1 ...
##  $ RDW         : num  19.5 15 19.7 13.8 12.6 13.9 13.1 16.8 13.3 13.2 ...
##  $ MPV         : num  10.8 9.7 11.1 8.5 10 10.9 10.3 NA 10.7 11.3 ...
##  $ LYM         : num  0.4 0.4 1.5 1.3 0.8 2.2 2.1 0.4 0.4 0.9 ...
##  $ MONO        : num  1.7 0.2 1.2 0.8 0.4 0.9 1.6 0.1 0.2 0.9 ...
##  $ EOS         : num  0 0.1 0.1 0 0 0.1 0.3 0.1 0 0.3 ...
##  $ BASO        : num  0.1 0 0.1 0 0 0 0 0 0 0.1 ...
##  $ NT          : int  86 90 58 67 95 61 NA 93 57 69 ...
##  $ APTT        : num  28.8 29.8 36.3 38.2 33.1 41.8 NA 36.3 33.8 28.1 ...
##  $ FIB         : int  578 NA 313 487 490 400 NA 413 431 407 ...
##  $ SODIUM      : int  137 141 147 141 137 141 139 142 143 136 ...
##  $ POTASS      : num  3.88 NA 4.61 4.71 NA 4.41 3.69 4.67 2.35 3.8 ...
##  $ CA          : num  2.29 2.21 1.92 2.05 2.34 2.08 NA 2.31 2.1 1.92 ...
##  $ PHOS        : num  1.2 0.58 1.51 2.17 0.97 0.99 NA 1.16 0.51 0.72 ...
##  $ MG          : num  0.66 NA 1.03 0.83 0.74 0.56 NA 0.87 0.36 0.53 ...
##  $ CREA        : num  0.65 0.76 1.25 2.78 0.65 0.82 1.21 1.77 1 0.58 ...
##  $ BUN         : num  5.7 19.9 50.6 47.5 8.5 15.3 13 29.8 15 14 ...
##  $ HS          : num  5.3 NA NA 9.7 3 5.5 NA 6.2 4.7 4 ...
```

```
##  $ GBIL        : num  0.59 0.48 8.42 0.35 0.42 2.4 1.13 0.45 1.21 2.46 ...
##  $ TP          : num  67 65.3 40.5 61.2 78.4 57.5 NA 70.8 67.4 53.8 ...
##  $ ALB         : num  36.7 37.4 22.1 33.2 43.8 30.1 NA 43.6 35.4 24.8 ...
##  $ AMY         : int  30 NA 146 92 84 95 117 177 NA 35 ...
##  $ PAMY        : int  16 NA NA 28 50 57 NA 43 NA 35 ...
##  $ LIP         : int  10 NA 89 18 50 25 73 30 NA 38 ...
##  $ CHE         : num  5.12 5.61 2.52 4.1 6.91 6.79 NA 7.4 NA 2.64 ...
##  $ AP          : int  85 80 119 94 108 68 51 153 239 146 ...
##  $ ASAT        : int  22 28 124 774 35 32 29 26 91 97 ...
##  $ ALAT        : int  14 25 135 72 22 11 20 32 57 156 ...
##  $ GGT         : int  48 61 134 23 72 68 138 96 446 192 ...
##  $ LDH         : int  284 NA 696 1787 NA 263 303 181 183 277 ...
##  $ CK          : int  23 36 40 2422 79 75 230 87 53 87 ...
##  $ GLU         : int  107 84 107 105 93 89 91 96 86 104 ...
##  $ TRIG        : int  105 NA NA 134 152 85 NA 129 62 207 ...
##  $ CHOL        : int  175 NA NA 141 167 144 NA 156 118 123 ...
##  $ CRP         : num  3.94 1.42 12.09 3.78 11.17 ...
##  $ BASOR       : num  0.413 0 0.568 0 0 ...
##  $ EOSR        : num  0 0.826 0.568 0 0 ...
##  $ LYMR        : num  1.65 3.31 8.52 11.02 8.33 ...
##  $ MONOR       : num  7.02 1.65 6.82 6.78 4.17 ...
##  $ NEU         : num  22 11.4 14.7 9.7 8.4 6.8 8.9 1.2 NA 3.8 ...
##  $ NEUR        : num  90.9 94.2 83.5 82.2 87.5 ...
##  $ PDW         : num  10.6 11.4 14.1 8.7 12.2 12.9 12.5 NA NA 13.2 ...
##  $ RBC         : num  3.7 3.9 2.5 3.5 4.4 4.3 4.5 4.7 NA 3.5 ...
##  $ WBC         : num  24.1 12.17 17.45 11.58 9.86 ...
##  $ BloodCulture: chr  "no" "no" "no" "no" ...
```

```r
# Observamos su composición
dim(bacteremia_df)
```

```
## [1] 14691    53
```

Vemos que tenemos **53** características y **14691** observaciones. Deberíamos profundizar ahora respecto al tipo de dato de cada variable y realizar un exploratorio básico para ver inconsistencias.

```r
# Obtenemos el datatype de cada característica
sapply(bacteremia_df, class)
```

```
##          ID         SEX         AGE         MCV         HGB         HCT
##   "integer"   "integer"   "integer"   "numeric"   "numeric"   "numeric"
##         PLT         MCH        MCHC         RDW         MPV         LYM
##   "integer"   "numeric"   "numeric"   "numeric"   "numeric"   "numeric"
##        MONO         EOS        BASO          NT        APTT         FIB
##   "numeric"   "numeric"   "numeric"   "integer"   "numeric"   "integer"
##      SODIUM      POTASS          CA        PHOS          MG        CREA
##   "integer"   "numeric"   "numeric"   "numeric"   "numeric"   "numeric"
##         BUN          HS        GBIL          TP         ALB         AMY
##   "numeric"   "numeric"   "numeric"   "numeric"   "numeric"   "integer"
##        PAMY         LIP         CHE          AP        ASAT        ALAT
##   "integer"   "integer"   "numeric"   "integer"   "integer"   "integer"
##         GGT         LDH          CK         GLU        TRIG        CHOL
##   "integer"   "integer"   "integer"   "integer"   "integer"   "integer"
```

```
##          CRP        BASOR         EOSR         LYMR        MONOR          NEU
##    "numeric"    "numeric"    "numeric"    "numeric"    "numeric"    "numeric"
##         NEUR          PDW          RBC          WBC BloodCulture
##    "numeric"    "numeric"    "numeric"    "numeric"  "character"
```

Respecto a las características, no tenemos transformaciones previas a realizar, exceptuando nuevas features que podríamos crear sucesivamente.

Procederemos con un exploratorio básico extrayendo la información estadístico de las características del dataset.

```
# Mostramos los estadísticos en pantalla, los valores nulos y frecuentes
summary(bacteremia_df)
```

```
##        ID             SEX             AGE              MCV
##  Min.   :    1   Min.   :1.000   Min.   : 16.00   Min.   : 51.00
##  1st Qu.:13586   1st Qu.:1.000   1st Qu.: 43.00   1st Qu.: 84.70
##  Median :28755   Median :1.000   Median : 58.00   Median : 88.30
##  Mean   :29353   Mean   :1.419   Mean   : 56.17   Mean   : 88.35
##  3rd Qu.:44670   3rd Qu.:2.000   3rd Qu.: 70.00   3rd Qu.: 92.00
##  Max.   :62455   Max.   :2.000   Max.   :101.00   Max.   :128.70
##                                                   NA's   :42
##       HGB             HCT             PLT             MCH             MCHC
##  Min.   : 3.00   Min.   : 0.00   Min.   :   0   Min.   :14.90   Min.   :23.70
##  1st Qu.: 9.90   1st Qu.:29.80   1st Qu.: 140   1st Qu.:28.40   1st Qu.:32.60
##  Median :11.40   Median :34.30   Median : 204   Median :29.70   Median :33.50
##  Mean   :11.57   Mean   :34.48   Mean   : 220   Mean   :29.58   Mean   :33.47
##  3rd Qu.:13.20   3rd Qu.:39.10   3rd Qu.: 277   3rd Qu.:31.00   3rd Qu.:34.40
##  Max.   :21.00   Max.   :66.60   Max.   :2092   Max.   :47.40   Max.   :43.50
##  NA's   :41      NA's   :42      NA's   :42     NA's   :42      NA's   :42
##       RDW             MPV             LYM              MONO
##  Min.   :10.6   Min.   : 7.30   Min.   :  0.000   Min.   : 0.0000
##  1st Qu.:13.4   1st Qu.: 9.70   1st Qu.:  0.700   1st Qu.: 0.5000
##  Median :14.5   Median :10.30   Median :  1.000   Median : 0.8000
##  Mean   :15.0   Mean   :10.38   Mean   :  1.366   Mean   : 0.8527
##  3rd Qu.:16.0   3rd Qu.:11.00   3rd Qu.:  1.600   3rd Qu.: 1.1000
##  Max.   :31.8   Max.   :15.00   Max.   :578.100   Max.   :20.4000
##  NA's   :56     NA's   :702     NA's   :262       NA's   :246
##       EOS              BASO              NT              APTT
##  Min.   : 0.0000   Min.   :0.00000   Min.   :  4.00   Min.   : 21.40
##  1st Qu.: 0.0000   1st Qu.:0.00000   1st Qu.: 67.00   1st Qu.: 34.10
##  Median : 0.1000   Median :0.00000   Median : 83.00   Median : 37.70
##  Mean   : 0.1148   Mean   :0.01725   Mean   : 83.22   Mean   : 40.06
##  3rd Qu.: 0.1000   3rd Qu.:0.00000   3rd Qu.:101.00   3rd Qu.: 42.70
##  Max.   :15.8000   Max.   :6.50000   Max.   :152.00   Max.   :176.10
##  NA's   :135       NA's   :146       NA's   :2467     NA's   :2549
##       FIB             SODIUM          POTASS             CA
##  Min.   :  55.0   Min.   :106.0   Min.   : 1.920   Min.   :1.030
##  1st Qu.: 397.0   1st Qu.:135.0   1st Qu.: 3.660   1st Qu.:2.090
##  Median : 529.0   Median :137.0   Median : 3.950   Median :2.220
##  Mean   : 547.4   Mean   :137.2   Mean   : 4.003   Mean   :2.214
##  3rd Qu.: 674.0   3rd Qu.:140.0   3rd Qu.: 4.290   3rd Qu.:2.350
##  Max.   :1593.0   Max.   :170.0   Max.   :36.620   Max.   :4.400
##  NA's   :2567     NA's   :1282    NA's   :2008     NA's   :1276
```

```
##       PHOS             MG              CREA             BUN
##  Min.   :0.300   Min.   :0.2000   Min.   : 0.260   Min.   :  2.50
##  1st Qu.:0.810   1st Qu.:0.7200   1st Qu.: 0.810   1st Qu.: 11.60
##  Median :0.990   Median :0.8100   Median : 1.000   Median : 16.60
##  Mean   :1.048   Mean   :0.8136   Mean   : 1.329   Mean   : 22.66
##  3rd Qu.:1.200   3rd Qu.:0.8900   3rd Qu.: 1.350   3rd Qu.: 26.90
##  Max.   :6.220   Max.   :2.2200   Max.   :20.750   Max.   :184.80
##  NA's   :1242    NA's   :1869     NA's   :159      NA's   :172
##        HS              GBIL             TP             ALB
##  Min.   : 1.300   Min.   : 0.110   Min.   : 29.9   Min.   :10.00
##  1st Qu.: 3.700   1st Qu.: 0.530   1st Qu.: 56.9   1st Qu.:27.90
##  Median : 5.000   Median : 0.770   Median : 65.7   Median :33.60
##  Mean   : 5.413   Mean   : 1.406   Mean   : 64.9   Mean   :33.42
##  3rd Qu.: 6.600   3rd Qu.: 1.230   3rd Qu.: 73.3   3rd Qu.:39.10
##  Max.   :22.700   Max.   :51.770   Max.   :120.9   Max.   :55.70
##  NA's   :3061     NA's   :1441     NA's   :1583    NA's   :1676
##       AMY               PAMY               LIP               CHE
##  Min.   :    8.00   Min.   :    1.00   Min.   :    0.00   Min.   : 0.98
##  1st Qu.:   33.00   1st Qu.:   14.00   1st Qu.:   14.00   1st Qu.: 3.15
##  Median :   49.00   Median :   22.00   Median :   23.00   Median : 4.60
##  Mean   :   90.83   Mean   :   41.66   Mean   :   63.82   Mean   : 4.79
##  3rd Qu.:   76.00   3rd Qu.:   36.00   3rd Qu.:   40.00   3rd Qu.: 6.22
##  Max.   :56146.00   Max.   :38369.00   Max.   :45991.00   Max.   :13.89
##  NA's   :3913       NA's   :7114       NA's   :3699       NA's   :2447
##        AP              ASAT              ALAT               GGT
##  Min.   :  11.0   Min.   :    3.0   Min.   :    0.00   Min.   :   3.0
##  1st Qu.:  63.0   1st Qu.:   22.0   1st Qu.:   16.00   1st Qu.:  25.0
##  Median :  84.0   Median :   31.0   Median :   26.00   Median :  49.0
##  Mean   : 118.8   Mean   :   86.9   Mean   :   67.66   Mean   : 115.1
##  3rd Qu.: 123.0   3rd Qu.:   56.0   3rd Qu.:   48.00   3rd Qu.: 117.0
##  Max.   :2995.0   Max.   :13991.0   Max.   :15059.00   Max.   :5171.0
##  NA's   :1400     NA's   :1154      NA's   :987        NA's   :1262
##       LDH               CK              GLU              TRIG
##  Min.   :   39.0   Min.   :    8   Min.   :  19.0   Min.   :  14.0
##  1st Qu.:  187.0   1st Qu.:   42   1st Qu.:  97.0   1st Qu.:  83.0
##  Median :  239.0   Median :   80   Median : 113.0   Median : 115.0
##  Mean   :  331.2   Mean   :  385   Mean   : 126.4   Mean   : 141.7
##  3rd Qu.:  332.0   3rd Qu.:  184   3rd Qu.: 138.0   3rd Qu.: 165.0
##  Max.   :13906.0   Max.   :98801   Max.   :1403.0   Max.   :5440.0
##  NA's   :1714      NA's   :2080    NA's   :4192     NA's   :5061
##       CHOL             CRP             BASOR              EOSR
##  Min.   :  25.0   Min.   : 0.00   Min.   : 0.0000   Min.   : 0.0000
##  1st Qu.: 113.0   1st Qu.: 2.87   1st Qu.: 0.0000   1st Qu.: 0.0000
##  Median : 145.0   Median : 8.57   Median : 0.0000   Median : 0.5882
##  Mean   : 150.8   Mean   :10.92   Mean   : 0.1451   Mean   : 1.2974
##  3rd Qu.: 182.0   3rd Qu.:16.45   3rd Qu.: 0.0000   3rd Qu.: 1.7857
##  Max.   :1104.0   Max.   :76.32   Max.   :23.6559   Max.   :73.4884
##  NA's   :5045     NA's   :155     NA's   :732       NA's   :732
##       LYMR             MONOR              NEU              NEUR
##  Min.   :  0.000   Min.   :  0.000   Min.   : 0.000   Min.   :  0.00
##  1st Qu.:  6.757   1st Qu.:  5.634   1st Qu.: 4.600   1st Qu.: 69.23
##  Median : 11.340   Median :  8.000   Median : 7.300   Median : 78.33
##  Mean   : 14.614   Mean   :  8.793   Mean   : 8.367   Mean   : 75.15
##  3rd Qu.: 18.182   3rd Qu.: 10.870   3rd Qu.:10.800   3rd Qu.: 85.32
```

```
##  Max.   :100.000   Max.   :100.000   Max.   :83.800   Max.   :100.00
##  NA's   :732       NA's   :732       NA's   :728      NA's   :732
##       PDW              RBC              WBC         BloodCulture
##  Min.   : 6.60   Min.   :1.000   Min.   :  0.00   Length:14691
##  1st Qu.:10.80   1st Qu.:3.400   1st Qu.:  6.63   Class :character
##  Median :12.00   Median :3.900   Median :  9.60   Mode  :character
##  Mean   :12.29   Mean   :3.936   Mean   : 11.23
##  3rd Qu.:13.40   3rd Qu.:4.500   3rd Qu.: 13.53
##  Max.   :25.30   Max.   :8.200   Max.   :604.47
##  NA's   :1102    NA's   :461     NA's   :462
```

De las **53 características** observamos que existen:

- 2 variables del tipo numérica discretas (SEX, BloodCulture), aunque podríamos tratar esta última como variable dependiente.
- 50 variables son del tipo numérica continua a excepción de la característica id del paciente `ID`, que no es considerada para la modelización, peró en caso de realizar segmentación, podría ser útil.

Hay que tener en cuenta que la variable `AGE` ha sido modificada de origen ligermente por lo que los estudios analíticos a continuación, no serán del todo precisas y fiables. Por lo que propondríamos suposiciones a lo largo del estudio.

Podemos revisar la descripción de las variables contenidas en el fichero y si los tipos de variables se corresponden con las que hemos cargado. Las organizamos lógicamente para darles sentido y construimos un pequeño diccionario de datos utilizando la documentación auxiliar.

Respecto al significado de cada una de las features utilizaremos el fichero DataDictionary - https://zenodo.org/records/7554815/files/bacteremia-DataDictionary.csv disponible en la misma página oficial para interpretar los valores.

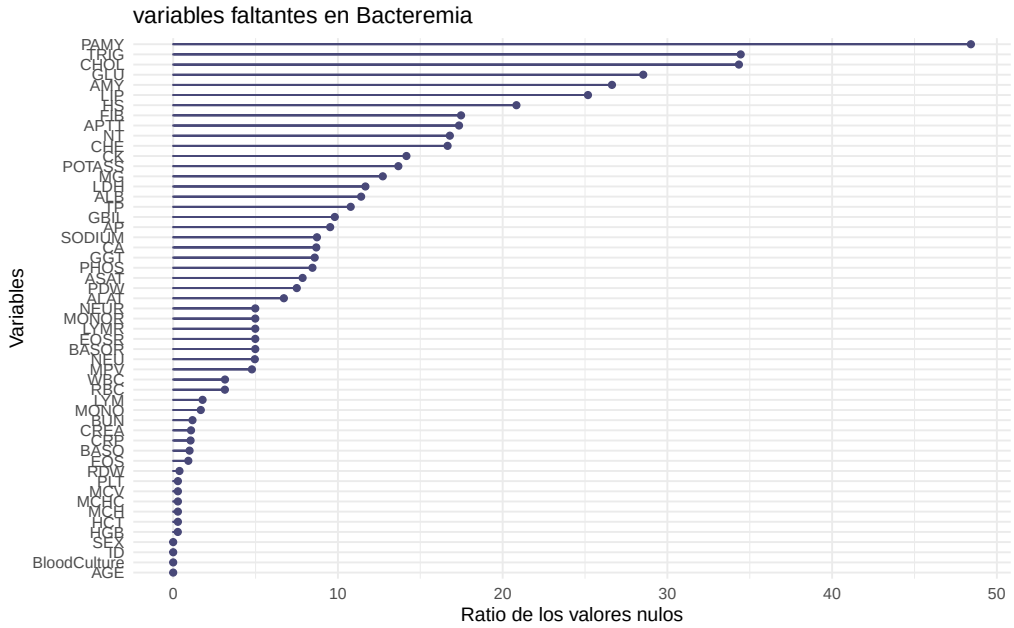
A continuación comprobaremos los valores nulos que podrían estar en nuestro dataset.

```r
# Verificación de valores perdidos
na_count <- sapply(bacteremia_df, function(x) sum(is.na(x)))
print(na_count[na_count > 0])
```

```
##    MCV    HGB    HCT    PLT    MCH   MCHC    RDW    MPV    LYM   MONO    EOS
##     42     41     42     42     42     42     56    702    262    246    135
##   BASO     NT   APTT    FIB SODIUM POTASS     CA   PHOS     MG   CREA    BUN
##    146   2467   2549   2567   1282   2008   1276   1242   1869    159    172
##     HS   GBIL     TP    ALB    AMY   PAMY    LIP    CHE     AP   ASAT   ALAT
##   3061   1441   1583   1676   3913   7114   3699   2447   1400   1154    987
##    GGT    LDH     CK    GLU   TRIG   CHOL    CRP  BASOR  EOSR  LYMR  MONOR
##   1262   1714   2080   4192   5061   5045    155    732    732    732    732
##    NEU   NEUR    PDW    RBC    WBC
##    728    732   1102    461    462
```

De las 14691 observaciones, disponemos de muchos valores nulos, podemos realizar un análisis para comprobar el porcentaje que podría este afectar a la estabilidad de los análisis.

```r
# Visualize missing values by variable
gg_miss_var(bacteremia_df, show_pct = TRUE) +
  labs(title = "variables faltantes en Bacteremia",
       x = "Variables",
       y = "Ratio de los valores nulos")
```



Calcularemos los valores totales y su ratio.

```
missing_summary <- data.frame(
  Variable = names(bacteremia_df),
  Missing_Count = colSums(is.na(bacteremia_df)),
  Missing_Percent = round(colSums(is.na(bacteremia_df))/nrow(bacteremia_df)*100, 2)
) %>%
  arrange(desc(Missing_Percent))

print("Missing Data Summary:")
```

```
## [1] "Missing Data Summary:"
```

```
print(missing_summary)
```

```
##                  Variable Missing_Count Missing_Percent
## PAMY                PAMY          7114           48.42
## TRIG                TRIG          5061           34.45
## CHOL                CHOL          5045           34.34
## GLU                  GLU          4192           28.53
## AMY                  AMY          3913           26.64
## LIP                  LIP          3699           25.18
## HS                    HS          3061           20.84
## FIB                  FIB          2567           17.47
## APTT                APTT          2549           17.35
## NT                    NT          2467           16.79
## CHE                  CHE          2447           16.66
## CK                    CK          2080           14.16
## POTASS            POTASS          2008           13.67
## MG                    MG          1869           12.72
## LDH                  LDH          1714           11.67
## ALB                  ALB          1676           11.41
```

```
## TP                     TP          1583          10.78
## GBIL                 GBIL          1441           9.81
## AP                     AP          1400           9.53
## SODIUM             SODIUM          1282           8.73
## CA                     CA          1276           8.69
## GGT                   GGT          1262           8.59
## PHOS                 PHOS          1242           8.45
## ASAT                 ASAT          1154           7.86
## PDW                   PDW          1102           7.50
## ALAT                 ALAT           987           6.72
## BASOR               BASOR           732           4.98
## EOSR                 EOSR           732           4.98
## LYMR                 LYMR           732           4.98
## MONOR               MONOR           732           4.98
## NEUR                 NEUR           732           4.98
## NEU                   NEU           728           4.96
## MPV                   MPV           702           4.78
## RBC                   RBC           461           3.14
## WBC                   WBC           462           3.14
## LYM                   LYM           262           1.78
## MONO                 MONO           246           1.67
## BUN                   BUN           172           1.17
## CREA                 CREA           159           1.08
## CRP                   CRP           155           1.06
## BASO                 BASO           146           0.99
## EOS                   EOS           135           0.92
## RDW                   RDW            56           0.38
## MCV                   MCV            42           0.29
## HCT                   HCT            42           0.29
## PLT                   PLT            42           0.29
## MCH                   MCH            42           0.29
## MCHC                 MCHC            42           0.29
## HGB                   HGB            41           0.28
## ID                     ID             0           0.00
## SEX                   SEX             0           0.00
## AGE                   AGE             0           0.00
## BloodCulture BloodCulture             0           0.00
```

Notamos que no existe características que superen al umbral del 70% de los valores nulos, lo que podríamos considerar a eliminar la característica de nuestros análisis.

Sin embargo, hay valores que necesitaríamos analizar a posteriori por tener unos valores faltantes entre el 10% y 40% que supone tener que tomar decisiones.

```
filter(missing_summary, missing_summary$Missing_Percent > 20)
```

```
##      Variable Missing_Count Missing_Percent
## PAMY     PAMY          7114           48.42
## TRIG     TRIG          5061           34.45
## CHOL     CHOL          5045           34.34
## GLU       GLU          4192           28.53
## AMY       AMY          3913           26.64
## LIP       LIP          3699           25.18
## HS         HS          3061           20.84
```

```r
filter(missing_summary, missing_summary$Missing_Percent > 0 & missing_summary$Missing_Percent < 20)
```

```
##         Variable Missing_Count Missing_Percent
## FIB          FIB          2567           17.47
## APTT        APTT          2549           17.35
## NT            NT          2467           16.79
## CHE          CHE          2447           16.66
## CK            CK          2080           14.16
## POTASS    POTASS          2008           13.67
## MG            MG          1869           12.72
## LDH          LDH          1714           11.67
## ALB          ALB          1676           11.41
## TP            TP          1583           10.78
## GBIL        GBIL          1441            9.81
## AP            AP          1400            9.53
## SODIUM    SODIUM          1282            8.73
## CA            CA          1276            8.69
## GGT          GGT          1262            8.59
## PHOS        PHOS          1242            8.45
## ASAT        ASAT          1154            7.86
## PDW          PDW          1102            7.50
## ALAT        ALAT           987            6.72
## BASOR      BASOR           732            4.98
## EOSR        EOSR           732            4.98
## LYMR        LYMR           732            4.98
## MONOR      MONOR           732            4.98
## NEUR        NEUR           732            4.98
## NEU          NEU           728            4.96
## MPV          MPV           702            4.78
## RBC          RBC           461            3.14
## WBC          WBC           462            3.14
## LYM          LYM           262            1.78
## MONO        MONO           246            1.67
## BUN          BUN           172            1.17
## CREA        CREA           159            1.08
## CRP          CRP           155            1.06
## BASO        BASO           146            0.99
## EOS          EOS           135            0.92
## RDW          RDW            56            0.38
## MCV          MCV            42            0.29
## HCT          HCT            42            0.29
## PLT          PLT            42            0.29
## MCH          MCH            42            0.29
## MCHC        MCHC            42            0.29
## HGB          HGB            41            0.28
```

Tenemos 7 características superior al 20% de valores nulos y 42 de muy residual, debajo del 1% hasta el 20%. Podemos realizar un análisis exhaustivo para poder tomar decisiones al respecto, cuáles tareas de imputaciones de los valores faltantes por segmentación de valores cercanos (por ejemplo algún valor estadístico tipo media, mediana) o a través de técnicas de aprendizaje automático (utilizar segmentación a través de k-Means u otros métodos supervisado tipo kNN).

2.2 Preguntas "objetivo" (1 punto)

Antes de comenzar con las preguntas objetivos, obtenemos una clasificación de los valores indicadores como rango de referencia para el hemocultivo.

De esta manera agruparemos los valores de las features por categoría:

A. Marcadores de inflamación e infección (Alta correlación esperada)

Variable	Meaning	Typical Range
CRP	C-reactive protein	< 5 mg/L
HS	Hypersensitive CRP	< 1 mg/L
NT	Neutrophils	40–75 %
WBC	White blood cells	4–10 $\times 10$ /L
LDH	Cell damage / inflammation	140–280 U/L
PCT	**Procalcitonin**	**< 0.5 ng/mL**

B. Coagulación y hemostasia (Correlación indirecta/moderada)

Variable	Meaning	Typical Range
FIB	Fibrinogen	200–400 mg/dL
APTT	Coagulation time	25–35 s
PLT	Platelets	150–400 $\times 10$ /L
PT	**Prothrombin time**	**11–13.5 s**
INR	**International normalized ratio**	**0.8–1.2**
DD	**D-dimer**	**< 500 ng/mL**

C. Perfil metabólico y lipídico (Correlación indirecta/débil)

Variable	Meaning	Typical Range
GLU	Glucose	70–100 mg/dL
TRIG	Triglycerides	< 150 mg/dL
CHOL	Cholesterol	< 200 mg/dL
HDL	**HDL cholesterol**	**> 40 mg/dL**
LDL	**LDL cholesterol**	**< 100 mg/dL**

D. Función hepática y proteínas (Correlación indirecta/moderada)

Variable	Meaning	Typical Range
ALB	Albumin	3.5–5.0 g/dL
TP	Total proteins	6.0–8.3 g/dL
CHE	Cholinesterase	5,000–12,000 U/L
TBIL	**Total bilirubin**	**0.1–1.2 mg/dL**
DBIL	**Direct bilirubin**	**0–0.3 mg/dL**
AST	**Aspartate aminotransferase**	**10–40 U/L**
ALT	**Alanine aminotransferase**	**7–56 U/L**
GGT	**Gamma-glutamyl transferase**	**8–61 U/L**
ALP	**Alkaline phosphatase**	**30–120 U/L**

E. Daño tisular/muscular (Correlación variable)

Variable	Meaning	Typical Range
CK	Creatine kinase	30–200 U/L
LDH	Cell damage	140–280 U/L
CKMB	**CK-MB (cardiac marker)**	**< 25 U/L**

F. Enzimas pancreáticas (Correlación débil - contextual)

Variable	Meaning	Typical Range
AMY	Amylase	30–110 U/L
PAMY	Pancreatic amylase	13–53 U/L
LIP	Lipase	< 160 U/L

G. Electrolitos y minerales (Correlación indirecta/moderada)

Variable	Meaning	Typical Range
POTASS	Potassium	3.5–5.1 mmol/L
MG	Magnesium	1.7–2.2 mg/dL
NA	**Sodium**	**136–145 mmol/L**
CL	**Chloride**	**98–107 mmol/L**
CA	**Calcium**	**8.5–10.5 mg/dL**
PHOS	**Phosphate**	**2.5–4.5 mg/dL**

H. Función renal (Alta correlación esperada)

Variable	Meaning	Typical Range
CREA	**Creatinine**	**0.7–1.3 mg/dL**
BUN	**Blood urea nitrogen**	**7–20 mg/dL**
UA	**Uric acid**	**3.5–7.2 mg/dL**

I. Hematología completa (Correlación variable)

Variable	Meaning	Typical Range
WBC	White blood cells	4–10 ×10 /L
RBC	**Red blood cells**	**4.5–5.5 $\times 10^{12}$/L**
HGB	**Hemoglobin**	**13–17 g/dL**
HCT	**Hematocrit**	**40–50 %**
MCV	**Mean corpuscular volume**	**80–100 fL**
MCH	**Mean corpuscular hemoglobin**	**27–33 pg**
MCHC	**Mean corpuscular Hb concentration**	**32–36 g/dL**
PLT	Platelets	150–400 ×10 /L
NT	Neutrophils	40–75 %
LY	**Lymphocytes**	**20–40 %**
MONO	**Monocytes**	**2–8 %**
EO	**Eosinophils**	**1–4 %**
BASO	**Basophils**	**0.5–1 %**

J. Variables demográficas y clínicas

Variable	Meaning	Notes
Age	Patient age	Modificada de origen
Sex	Patient sex	Revisión agrupaciones

2.2.1 ¿Existen diferencias descriptivas en los marcadores inflamatorios entre pacientes con y sin bacteriemia?

Podemos contestar a esta pregunta comparando la tendencia y dispersión de variables según el estado del hemocultivo.

2.2.2 ¿Cómo es la distribución de las variables metabólicas y qué asimetría presentan?

Podemos describir la forma de distribución de cada feature y detectar asimetría y valores extremos (outliers).

2.2.3 ¿Cuál es la relación descriptiva entre variables fisiológicas y existen agrupaciones naturales?

Aquí podemos explorar asociaciones entre variables y detectar patrones comunes.

2.2.4 ¿Existen diferencias descriptivas en la proporción de bacteriemia según categorías clínicas simples?

Podemos describir frecuencias y proporciones.

#Análisis exploratorio de los datos (2,5 puntos)
##Análisis descriptivo y gráfico (1 punto)
###Análisis descriptivo y gráfico general.
`r # Guardamos una copia del dataset bak_bacteremia <- bacteremia_df # bacteremia_df <- bak_bacteremia`
El primer dato que tenemos es la proporción que hay entre las dos clases de la variable target:
`r class_ratio <- prop.table(table(bacteremia_df$BloodCulture)) sprintf("La proporción de la variable dependiente es: %.2f para los valores %s y %.2f y para los valores %s.", class_ratio[1]*100, names(class_ratio)[1], class_ratio[2]*100, names(class_ratio)[2])`
`## [1] "La proporción de la variable dependiente es: 91.97 para los valores no y 8.03 y para los valores yes."`
Vemos que la diferencia del ratio de la posible variable dependiente está bastante desbalanceada a favor de los casos de valores negativos que positivos, 92-8.
A partir del conjunto de datos citado, mostraremos los estadísticos descriptivos más relevantes y comentaremos los resultados, teniendo en cuenta el tipo de variables.
"' r # Estadísticos descriptivos para variables numéricas excluyendo la ID y # la variable target y el género num_summary <- bacteremia_df %>% select(-ID, -BloodCulture, -SEX) %>% psych::describe() num_summary "'

```
##        vars     n   mean      sd median trimmed    mad    min      max    range ## AGE
1 14691  56.17   18.15  58.00   56.69  19.27  16.00   101.00    85.00 ## MCV       2
14649  88.35    6.46  88.30   88.32   5.34  51.00   128.70    77.70 ## HGB       3 14650
11.57    2.25  11.40   11.51   2.37   3.00    21.00    18.00 ## HCT       4 14649  34.48
6.51  34.30   34.45   6.97   0.00    66.60    66.60 ## PLT       5 14649 220.03  122.84
204.00  209.31 100.82   0.00  2092.00  2092.00 ## MCH       6 14649  29.58    2.53  29.70
29.66   1.93  14.90    47.40    32.50 ## MCHC      7 14649  33.47    1.40  33.50   33.51
1.33  23.70    43.50    19.80 ## RDW       8 14635  15.00    2.29  14.50   14.69   1.93
10.60    31.80    21.20 ## MPV       9 13989  10.38    1.01  10.30   10.34   1.04   7.30
15.00     7.70 ## LYM      10 14429   1.37    7.46   1.00    1.10   0.59   0.00   578.10
578.10 ## MONO     11 14445   0.85    0.65   0.80    0.79   0.44   0.00    20.40    20.40
## EOS      12 14556   0.11    0.27   0.10    0.07   0.15   0.00    15.80    15.80 ##
BASO     13 14545   0.02    0.08   0.00    0.00   0.00   0.00     6.50     6.50 ## NT
14 12224  83.22   27.15  83.00   83.70  25.20   4.00   152.00   148.00 ## APTT     15
12142  40.06   10.97  37.70   38.41   6.08  21.40   176.10   154.70 ## FIB      16 12124
547.36  208.13 529.00  535.68 204.60  55.00  1593.00  1538.00 ## SODIUM   17 13409 137.21
4.73 137.00  137.33   4.45 106.00   170.00    64.00 ## POTASS   18 12683   4.00    0.64
3.95    3.97   0.47   1.92    36.62    34.70 ## CA       19 13415   2.21    0.20   2.22
2.22   0.19   1.03     4.40     3.37 ## PHOS     20 13449   1.05    0.40   0.99    1.01
0.30   0.30     6.22     5.92 ## MG       21 12822   0.81    0.15   0.81    0.81   0.12
0.20     2.22     2.02 ## CREA     22 14532   1.33    1.17   1.00    1.09   0.34   0.26
20.75    20.49 ## BUN      23 14519  22.66   18.11  16.60   19.19   9.19   2.50   184.80
182.30 ## HS       24 11630   5.41    2.45   5.00    5.16   2.08   1.30    22.70    21.40
## GBIL     25 13250   1.41    2.75   0.77    0.89   0.44   0.11    51.77    51.66 ## TP
26 13108  64.90   11.46  65.70   65.23  12.01  29.90   120.90    91.00 ## ALB      27
13015  33.42    7.44  33.60   33.50   8.30  10.00    55.70    45.70 ## AMY      28 10778
90.83  805.23  49.00   54.93  29.65   8.00 56146.00 56138.00 ## PAMY     29  7577  41.66
448.01  22.00   25.38  14.83   1.00 38369.00 38368.00 ## LIP      30 10992  63.82  603.89
23.00   27.40  16.31   0.00 45991.00 45991.00 ## CHE      31 12244   4.79    2.10   4.60
4.68   2.25   0.98    13.89    12.91 ## AP       32 13291 118.78  132.87  84.00   93.42
38.55  11.00  2995.00  2984.00 ## ASAT     33 13537  86.90  404.69  31.00   39.77  17.79
3.00 13991.00 13988.00 ## ALAT     34 13704  67.66  311.05  26.00   32.69  17.79   0.00
15059.00 15059.00 ## GGT      35 13429 115.06  208.98  49.00   71.42  44.48   3.00
5171.00  5168.00 ## LDH      36 12977 331.15  475.23 239.00  259.60  94.89  39.00
13906.00 13867.00 ## CK       37 12611 385.01 2241.21  80.00  120.40  71.16   8.00
98801.00 98793.00 ## GLU      38 10499 126.41   56.91 113.00  117.60  28.17  19.00
1403.00  1384.00 ## TRIG     39  9630 141.71  120.69 115.00  124.12  56.34  14.00
5440.00  5426.00 ## CHOL     40  9646 150.80   55.51 145.00  147.40  50.41  25.00
1104.00  1079.00 ## CRP      41 14536  10.92    9.58   8.57    9.70   9.43   0.00
76.32    76.32 ## BASOR    42 13959   0.15    0.59   0.00    0.01   0.00   0.00    23.66
23.66 ## EOSR     43 13959   1.30    2.36   0.59    0.84   0.87   0.00    73.49    73.49
## LYMR     44 13959  14.61   12.74  11.34   12.50   7.90   0.00   100.00   100.00 ##
MONOR    45 13959   8.79    5.81   8.00    8.22   3.82   0.00   100.00   100.00 ## NEU
46 13963   8.37    5.61   7.30    7.73   4.45   0.00    83.80    83.80 ## NEUR     47
13959  75.15   15.53  78.33   77.26  11.44   0.00   100.00   100.00 ## PDW      48 13589
12.29    2.19  12.00   12.10   1.93   6.60    25.30    18.70 ## RBC      49 14230   3.94
0.77   3.90    3.93   0.89   1.00     8.20     7.20 ## WBC      50 14229  11.23   12.92
9.60   10.05   4.95   0.00   604.47   604.47 ##          skew kurtosis    se ## AGE
-0.24    -0.80  0.15 ## MCV     0.03     2.18  0.05 ## HGB     0.22    -0.31  0.02 ## HCT
-0.10     0.87  0.05 ## PLT     1.54     8.02  1.01 ## MCH    -0.38     2.67  0.02 ##
MCHC   -0.38     1.34  0.01 ## RDW     1.59     3.80  0.02 ## MPV     0.47     0.22  0.01
## LYM    57.60  3694.43  0.06 ## MONO    7.08   145.66  0.01 ## EOS    24.63  1149.47
0.00 ## BASO   39.50  2863.86  0.00 ## NT     -0.16    -0.03  0.25 ## APTT    4.17
29.74  0.10 ## FIB     0.66     0.83  1.89 ## SODIUM -0.25     3.24  0.04 ## POTASS 11.88
556.12  0.01 ## CA      0.10     4.27  0.00 ## PHOS    2.17    11.17  0.00 ## MG
0.88     4.31  0.00 ## CREA    5.03    37.06  0.01 ## BUN     2.52     8.84  0.15 ## HS
1.28     2.95  0.02 ## GBIL    8.15    88.89  0.02 ## TP     -0.22    -0.16  0.10 ## ALB
-0.10    -0.58  0.07 ## AMY    58.06  3575.67  7.76 ## PAMY   82.74  7066.98  5.15 ## LIP
51.06  3363.21  5.76 ## CHE     0.47    -0.20  0.02 ## AP      6.80    77.02  1.15 ##
ASAT   19.04   461.45  3.48 ## ALAT   23.80   801.62  2.66 ## GGT     6.73    78.28  1.80
```

Vamos a crear histogramas y describir los valores para ver los datos en general de estos atributos para hacer una primera aproximación a los datos:
"‘ r # Histograma de edad con densidad library(modeest)
modeest::mfv(bacteremia_df$AGE) "‘

```
## [1] 66
```

```
r ggplot(bacteremia_df, aes(x = AGE)) + geom_histogram(aes(y = after_stat(density)), bins
= 30, fill = "skyblue", color = "black") + geom_density(color = "red", size = 1.5) +
geom_vline(xintercept = mean(bacteremia_df$AGE), color = "blue", linetype = "dashed",
size = 1) + geom_vline(xintercept = median(bacteremia_df$AGE), color = "darkgreen",
linetype = "dashed", size = 1) + geom_vline(xintercept = modeest::mfv(bacteremia_df$AGE),
color = "purple", linetype = "dashed", size = 1) + labs(title = "Distribución de Edad",
subtitle = paste("Media =", round(mean(bacteremia_df$AGE), 1), "años", "\nMediana =",
round(median(bacteremia_df$AGE), 1), "años", "\nModa =",
round(modeest::mfv(bacteremia_df$AGE), 1), "años"), x = "Edad (años)", y = "Densidad" ) +
theme_minimal() + theme(plot.title = element_text(hjust = 0.5, face = "bold"))
```

Podemos detectar que la media está situada en 56.2 años con una mediana situada en 58. Pero los valores frecuentes está por los 66 años.
Realizaremos ahora un estudio por edad respecto a la variable dependiente

```
r # Extraeremos los estadísticos por edad bacteremia_df %>% group_by(BloodCulture) %>%
summarise( count = n(), mean_age = mean(AGE, na.rm = TRUE), median_age = median(AGE,
na.rm = TRUE), sd_age = sd(AGE, na.rm = TRUE), min_age = min(AGE, na.rm = TRUE), max_age
= max(AGE, na.rm = TRUE) )
```

```
## # A tibble: 2 x 7 ##   BloodCulture count mean_age median_age sd_age min_age max_age
##   <chr>        <int>    <dbl>      <dbl>  <dbl>   <int>   <int> ## 1 no
13511     55.6          58   18.2      16     101 ## 2 yes           1180     62.5
65   16.2      17      99
```

Graficaremos los casos negativos a la prueba realizada de BloodCulture, donde la media de la edad es 55, con un mínimo de 16 y máximo 101. Los casos positivos, que vimos anteriormente son un 8%, la media es 62 con un mínimo de 17 y la máxima edad 99. Aunque los datos podrían ser manipulados a origen, intentamos observar una cierta coherencia entre los dos casos. A conitnuación graficaremos con un boxplot.

```
r # Box plot: Edad por estado del hemocultivo ggplot(bacteremia_df, aes(x =
factor(BloodCulture, labels = c("Negativo", "Positivo")), y = AGE, fill =
factor(BloodCulture))) + geom_boxplot(alpha = 0.7) + scale_fill_manual(values =
c("#2ecc71", "#e74c3c")) + labs( title = "Distribución de Edad según el hemocultivo", x =
"Estado del hemocultivo ", y = "Edad (años)" ) + theme_minimal() + theme( plot.title =
element_text(hjust = 0.5, face = "bold"), legend.position = "none" )
```

Análisis descriptivo y gráfico: estudio de correlaciones:
Crearemos una nueva variable binaria para usarla en análisis posteriores:

```
r # Seleccionaremos los valores True y False bacteremia_df$BloodCulture_bin <-
ifelse(bacteremia_df$BloodCulture == "yes", 1, 0)
```

Mostramos gráficamente las correlaciones entre las variables de nuestro dataset:
"‘ r cor_data <- bacteremia_df %>% select(-ID, -BloodCulture, -BloodCulture_bin) %>% drop_na() # Creamos una matriz de correlación cor_matrix <- cor(cor_data)
Mostramos gráficamente corrplot(cor_matrix, method = "color", type = "upper", tl.srt = 45, tl.cex = 0.7, number.cex = 0.6) "‘

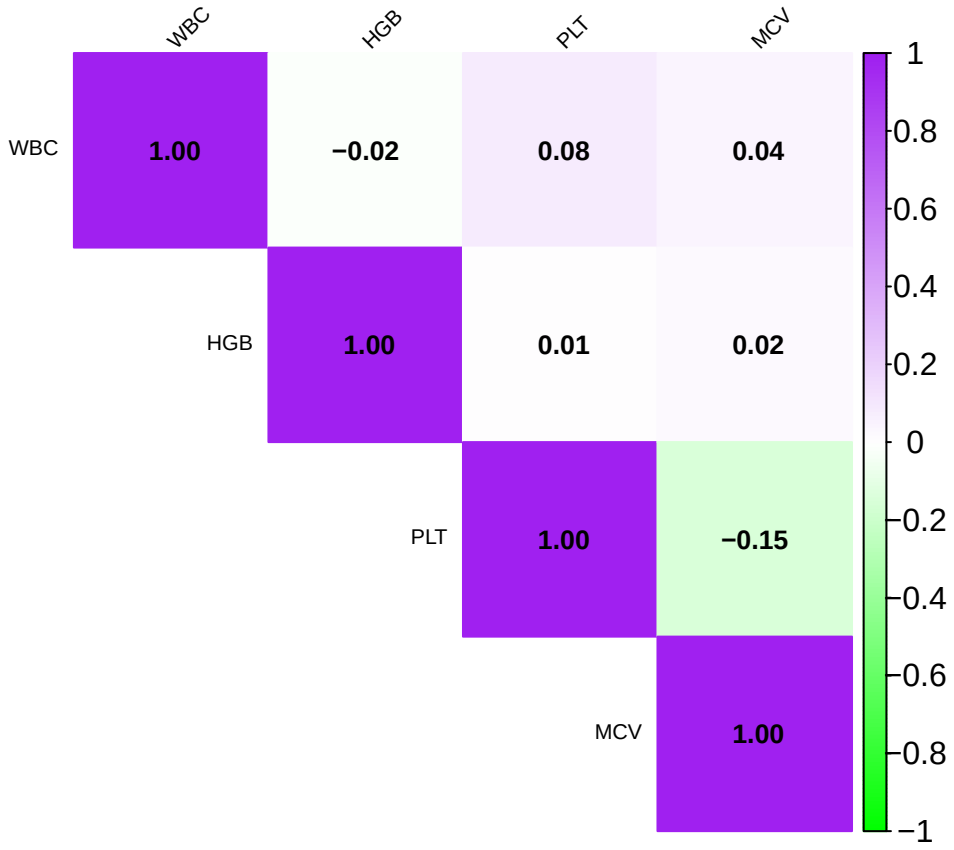
Al tener 53 variables, vamos a realizar un análisis de correlaciones separando las variables por las categorías anteriormente definidas (mapa de calor con escala azul-rojo). En este mapa, nos fijaremos en las variables intra categoría que estén áltamente correlacionadas. Después, se obtendrán otros mapas de calor (verde-morado) que mostrarán las correlaciones finales tras descartar las variables identificadas en el mapa previo. Finalmente, con las variables seleccionadas por categoría, realizaremos otro análisis de correlaciones para observar posibles relaciones considerando todas las ellas.
Nota: como criterio, se han eliminado las variables que presentaban una correlación media-alta, siendo el corte de 0.3 en valor absoluto. A pesar de que este corte puede ser considerado bajo, se ha decidido usar ya que debido al amplio número de variables, se dificultaba su viasulización e interpretación.
1. Variables del tipo Complete Blood Count (CBC) - Hematology
"' r # Correlación variables del tipo Complete Blood Count (CBC) - Hematology vars_big <- c("WBC", "RBC", "HGB", "HCT", "PLT", "MCV", "MCH", "MCHC", "RDW", "MPV", "PDW")
cor_big <- cor(bacteremia_df[, vars_big], use = "complete.obs")
corrplot(cor_big, method = "color", type = "upper", col = colorRampPalette(c("blue", "white", "red"))(200), addCoef.col = "black", number.cex = 0.8, # correlation numbers tl.col = "black", tl.cex = 0.7, # SMALLER labels tl.srt = 45, # ROTATED labels (45 degrees) cl.cex = 1.0, title = "Matriz de correlación", mar = c(0, 0, 3, 0)) "'

- RBC y HCT: correlación muy elevada con la Hemoglobina, 0.90 y 0.98 respectivamente y 0.92 entre HCT u RBC.

- MCH y PDW: MCH está muy ligada a MCV (0.87) y PDW a MPV (0.93).
- MVP: correlacionada a un nivel medio con PLT.
- RDW: correlacionada a un nivel medio (en torno al -0.34 y -0.45).
- MCHC: correlacionada a un nivel medio con MCH de 0.52

Correlacion tras no considerar las variables:

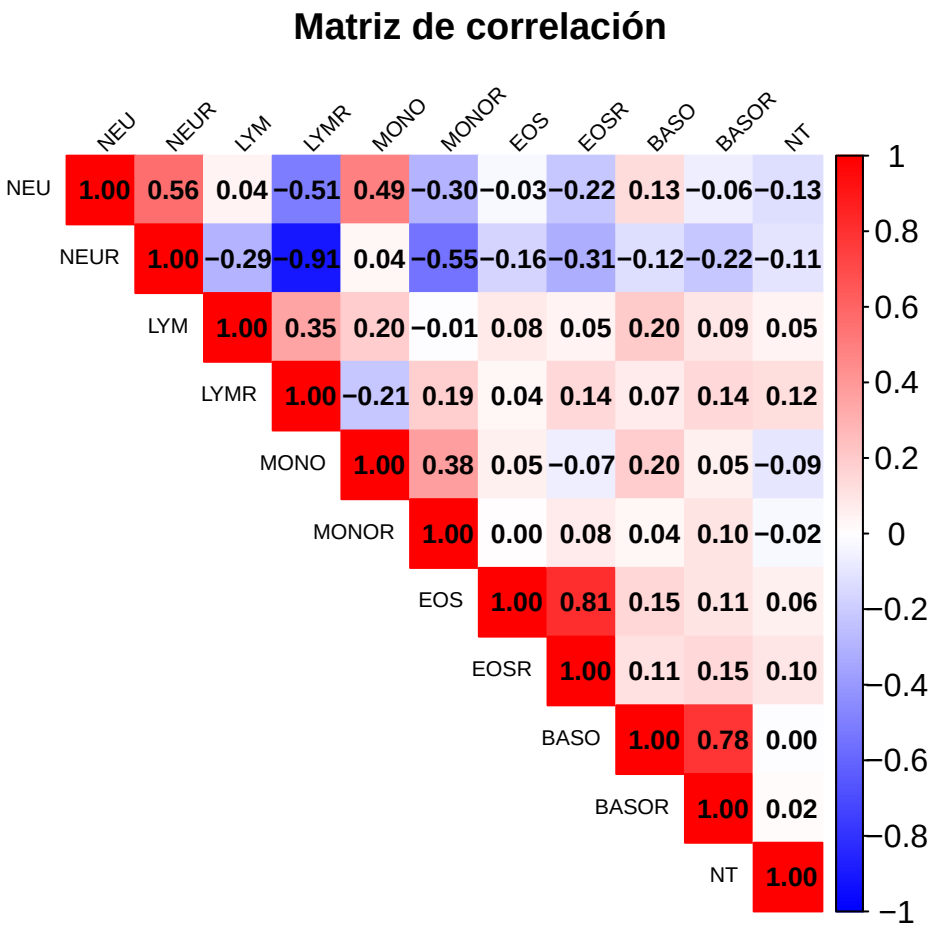
```r
# Correlación variables del tipo Complete Blood Count (CBC) - Hematology
vars_big <- c("WBC", "HGB", "PLT", "MCV"
              )

cor_big <- cor(
  bacteremia_df[, vars_big],
  use = "complete.obs"
)

corrplot(
  cor_big,
  method = "color",
  type = "upper",
  col = colorRampPalette(c("green", "white", "purple"))(200),
  addCoef.col = "black",
  number.cex = 0.8,      # correlation numbers
  tl.col = "black",
  tl.cex = 0.7,          # SMALLER labels
  tl.srt = 45,           # ROTATED labels (45 degrees)
  cl.cex = 1.0,
  title = "Matriz de correlación",
  mar = c(0, 0, 3, 0)
)
```



2. Variables del tipo White Blood Cell Differential

```
# Correlación variables del tipo White Blood Cell Differential
vars_big <- c("NEU", "NEUR", "LYM", "LYMR", "MONO", "MONOR",
              "EOS", "EOSR", "BASO", "BASOR", "NT"
              )

cor_big <- cor(
  bacteremia_df[, vars_big],
  use = "complete.obs"
)

corrplot(
  cor_big,
  method = "color",
  type = "upper",
  col = colorRampPalette(c("blue", "white", "red"))(200),
  addCoef.col = "black",
  number.cex = 0.8,      # correlation numbers
  tl.col = "black",
  tl.cex = 0.7,          # SMALLER labels
  tl.srt = 45,           # ROTATED labels (45 degrees)
  cl.cex = 1.0,
  title = "Matriz de correlación",
```

```
  mar = c(0, 0, 3, 0)
)
```



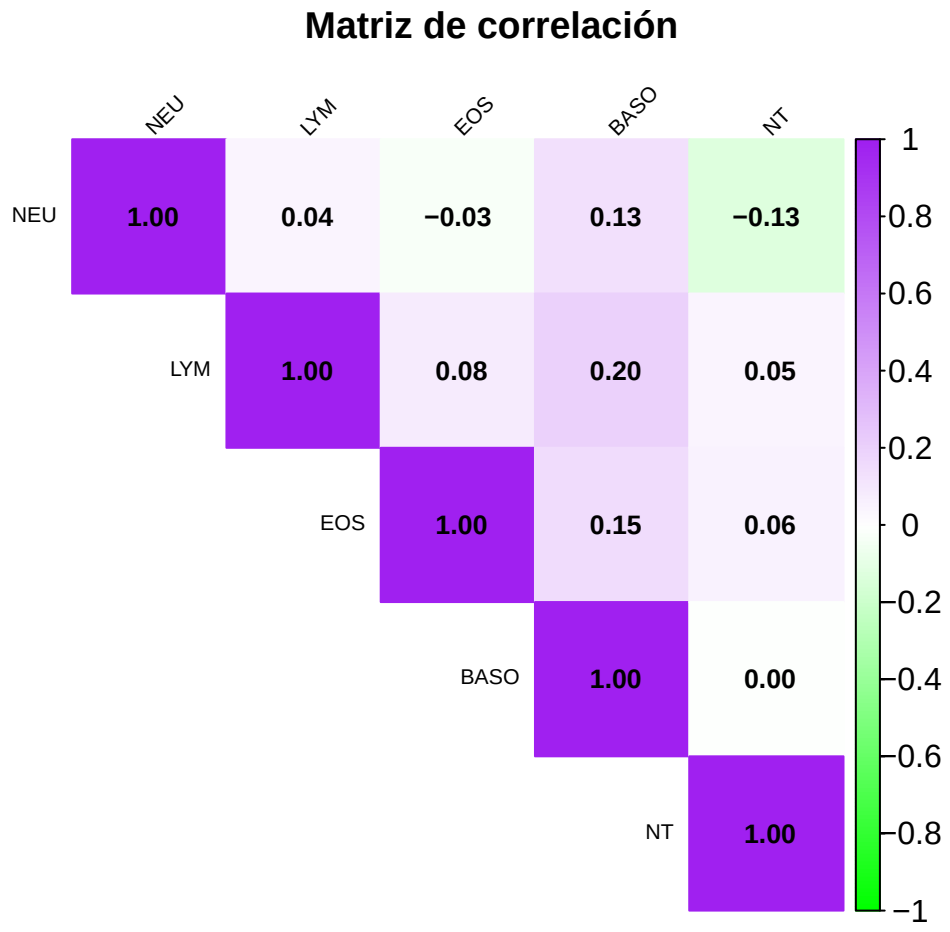
De froma análoga al caso anterior, descartamos las variables que están a un nivel alto medio de correlación.

- Variables correlacionadas a un nivel medio-alto: NEUR, LYMR, EOSR, BASOR, MONOR y MONO

Correlacion tras no considerar las variables:

```
# Correlación variables del tipo White Blood Cell Differential
vars_big <- c("NEU", "LYM", "EOS",  "BASO",  "NT"
              )

cor_big <- cor(
  bacteremia_df[, vars_big],
  use = "complete.obs"
)

corrplot(
  cor_big,
  method = "color",
  type = "upper",
  col = colorRampPalette(c("green", "white", "purple"))(200),
```

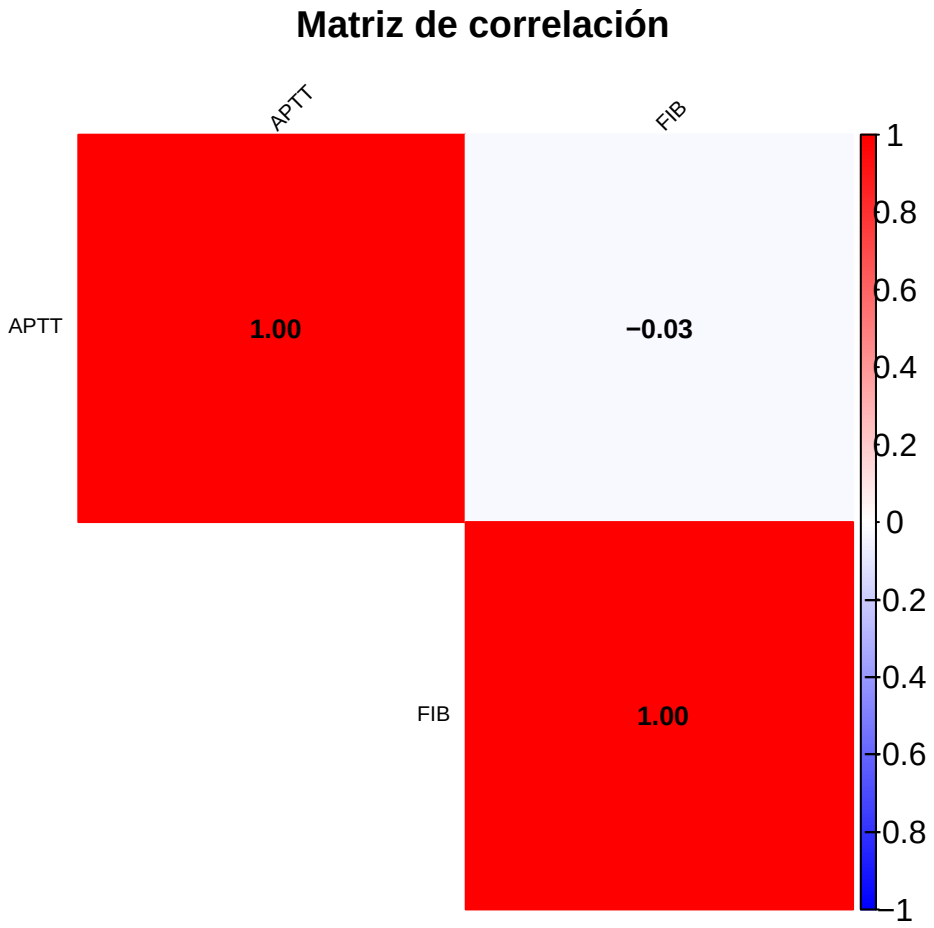
```
  addCoef.col = "black",
  number.cex = 0.8,      # correlation numbers
  tl.col = "black",
  tl.cex = 0.7,          # SMALLER labels
  tl.srt = 45,           # ROTATED labels (45 degrees)
  cl.cex = 1.0,
  title = "Matriz de correlación",
  mar = c(0, 0, 3, 0)
)
```



3. Variables del tipo Coagulation

```
# Correlación variables del tipo Coagulation
vars_big <- c("APTT", "FIB"
              )

cor_big <- cor(
  bacteremia_df[, vars_big],
  use = "complete.obs"
)

corrplot(
  cor_big,
```

```
  method = "color",
  type = "upper",
  col = colorRampPalette(c("blue", "white", "red"))(200),
  addCoef.col = "black",
  number.cex = 0.8,      # correlation numbers
  tl.col = "black",
  tl.cex = 0.7,          # SMALLER labels
  tl.srt = 45,           # ROTATED labels (45 degrees)
  cl.cex = 1.0,
  title = "Matriz de correlación",
  mar = c(0, 0, 3, 0)
)
```

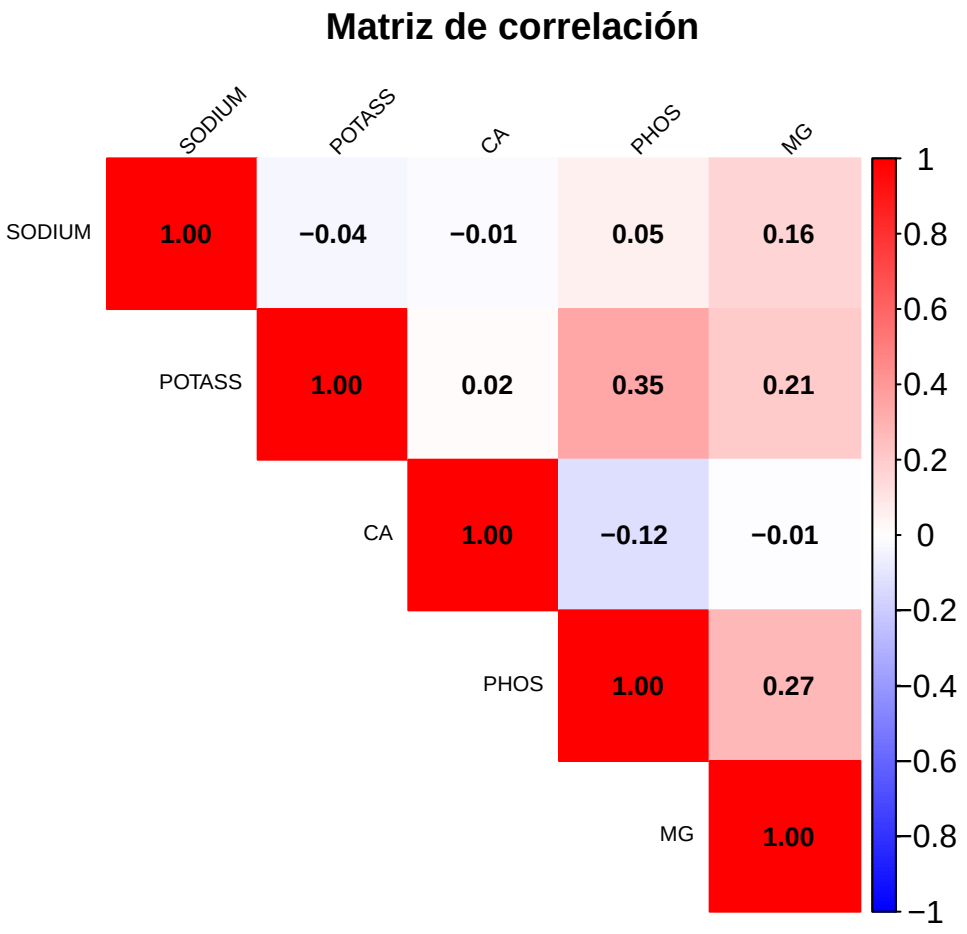


Consideraremos ambas variables ya que a penas están correlacionadas entre sí.

4. Variables del tipo Electrolytes & Minerals

```
# Correlación variables del tipo Electrolytes & Minerals
vars_big <- c("SODIUM", "POTASS", "CA", "PHOS", "MG"
              )

cor_big <- cor(
  bacteremia_df[, vars_big],
  use = "complete.obs"
```

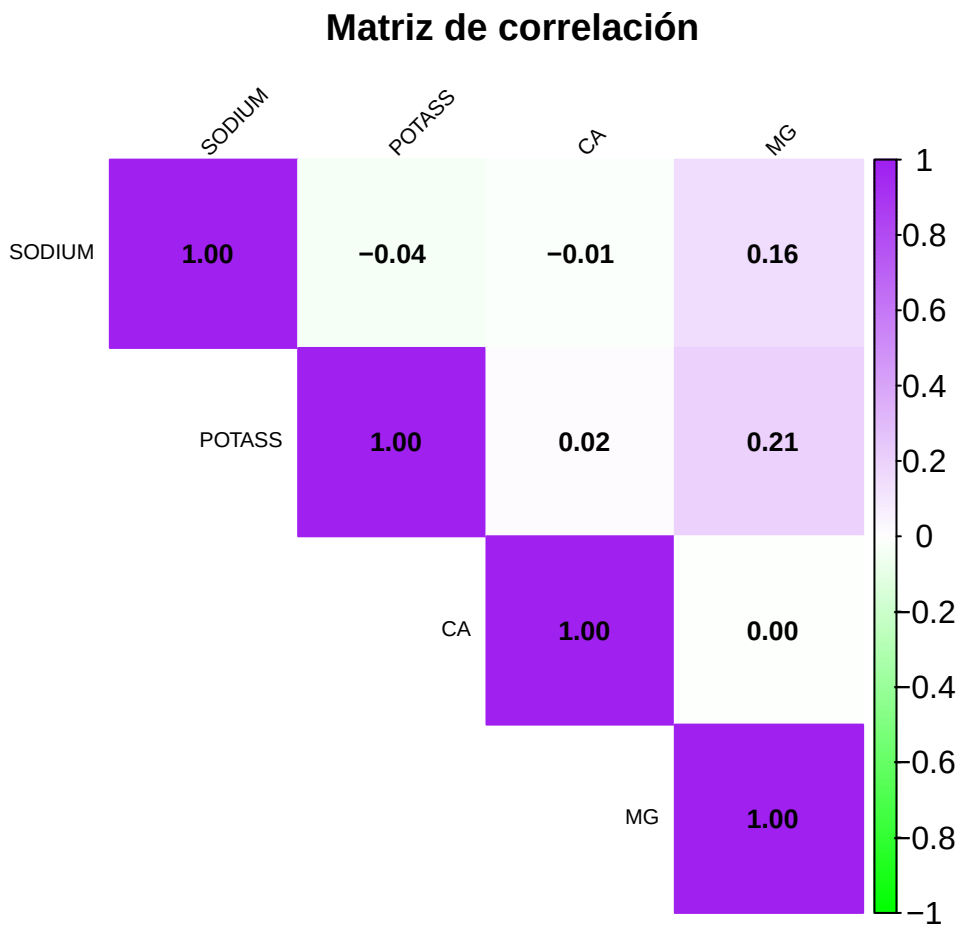
```
)

corrplot(
  cor_big,
  method = "color",
  type = "upper",
  col = colorRampPalette(c("blue", "white", "red"))(200),
  addCoef.col = "black",
  number.cex = 0.8,      # correlation numbers
  tl.col = "black",
  tl.cex = 0.7,          # SMALLER labels
  tl.srt = 45,           # ROTATED labels (45 degrees)
  cl.cex = 1.0,
  title = "Matriz de correlación",
  mar = c(0, 0, 3, 0)
)
```



- Variables descartadas: PHOS.

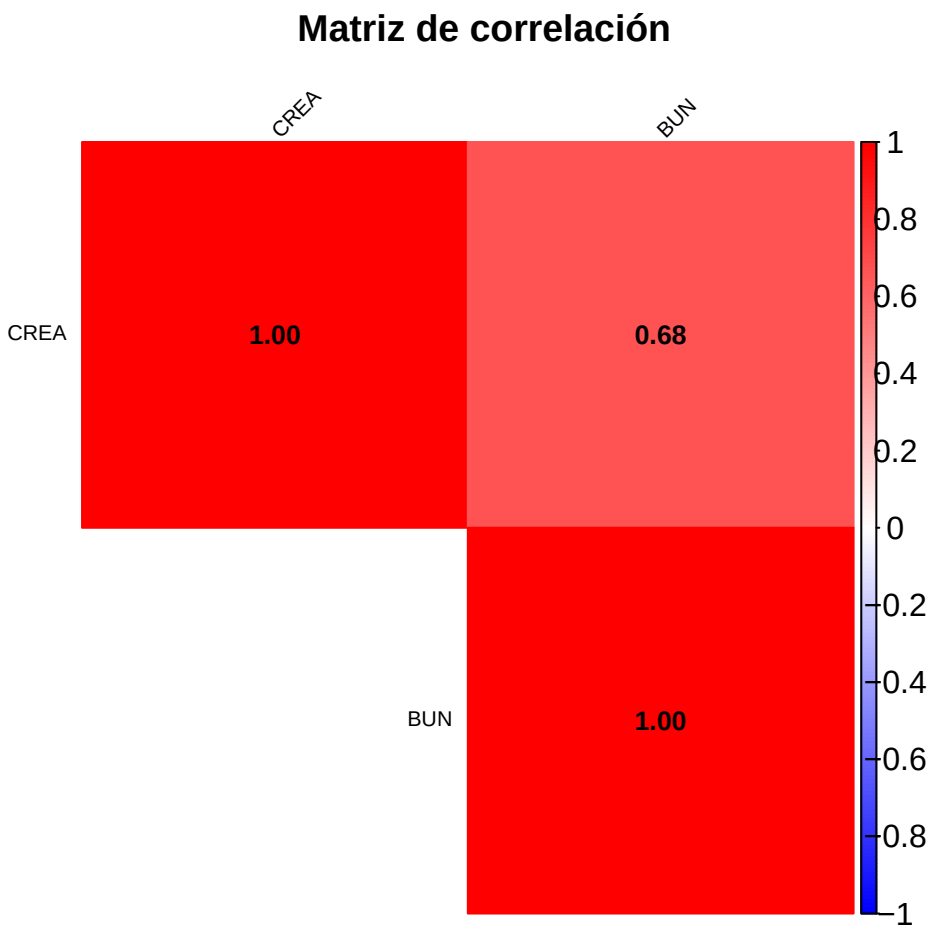
Correlacion tras descartar las variables:

```r
# Correlación variables del tipo Electrolytes & Minerals
vars_big <- c("SODIUM", "POTASS", "CA", "MG"
              )

cor_big <- cor(
  bacteremia_df[, vars_big],
  use = "complete.obs"
)

corrplot(
  cor_big,
  method = "color",
  type = "upper",
  col = colorRampPalette(c("green", "white", "purple"))(200),
  addCoef.col = "black",
  number.cex = 0.8,      # correlation numbers
  tl.col = "black",
  tl.cex = 0.7,          # SMALLER labels
  tl.srt = 45,           # ROTATED labels (45 degrees)
  cl.cex = 1.0,
  title = "Matriz de correlación",
  mar = c(0, 0, 3, 0)
)
```



5. Variables del tipo Kidney Function

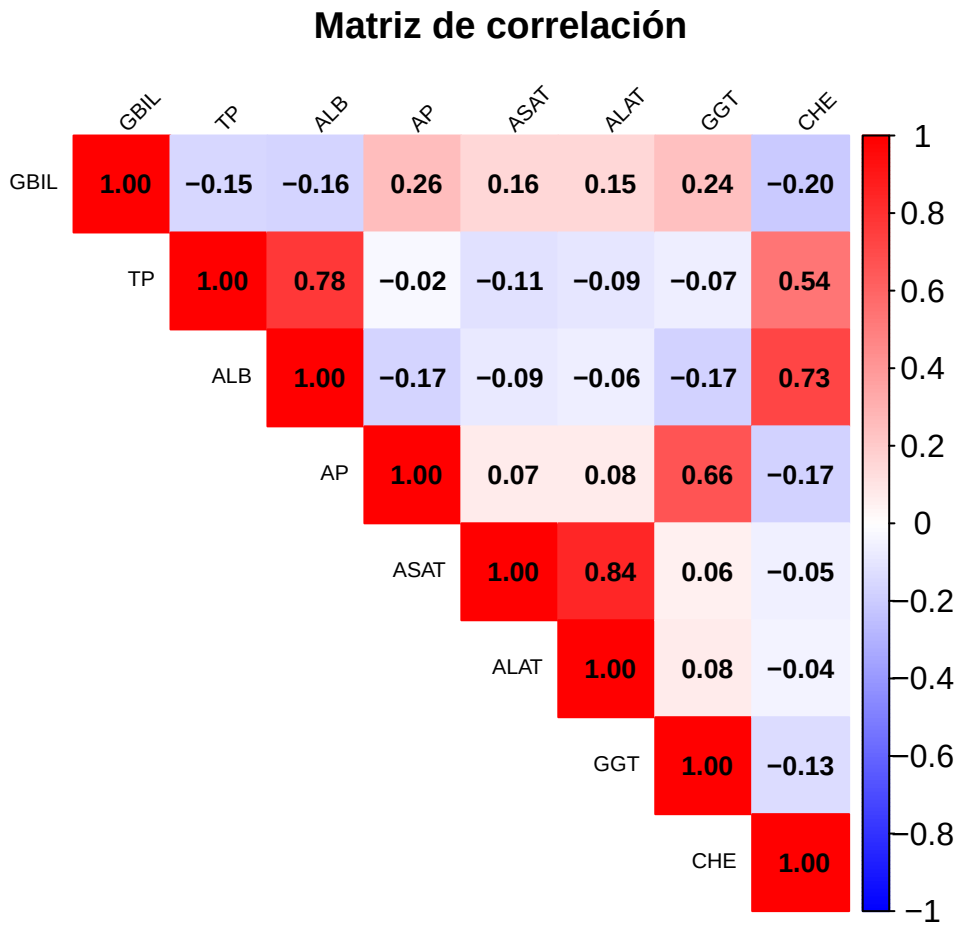
```
# Correlación variables del tipo  Kidney Function
vars_big <- c("CREA", "BUN"
              )

cor_big <- cor(
  bacteremia_df[, vars_big],
  use = "complete.obs"
)

corrplot(
  cor_big,
  method = "color",
  type = "upper",
  col = colorRampPalette(c("blue", "white", "red"))(200),
  addCoef.col = "black",
  number.cex = 0.8,      # correlation numbers
  tl.col = "black",
  tl.cex = 0.7,          # SMALLER labels
  tl.srt = 45,           # ROTATED labels (45 degrees)
  cl.cex = 1.0,
  title = "Matriz de correlación",
  mar = c(0, 0, 3, 0)
)
```



- Variables descartadas: CREA.

6. Variables del tipo Liver Function

```
# Correlación variables del tipo   Liver Function
vars_big <- c("GBIL", "TP", "ALB", "AP", "ASAT", "ALAT", "GGT", "CHE"
              )

cor_big <- cor(
  bacteremia_df[, vars_big],
  use = "complete.obs"
)

corrplot(
  cor_big,
  method = "color",
  type = "upper",
  col = colorRampPalette(c("blue", "white", "red"))(200),
  addCoef.col = "black",
  number.cex = 0.8,      # correlation numbers
  tl.col = "black",
  tl.cex = 0.7,          # SMALLER labels
  tl.srt = 45,           # ROTATED labels (45 degrees)
  cl.cex = 1.0,
  title = "Matriz de correlación",
  mar = c(0, 0, 3, 0)
)
```

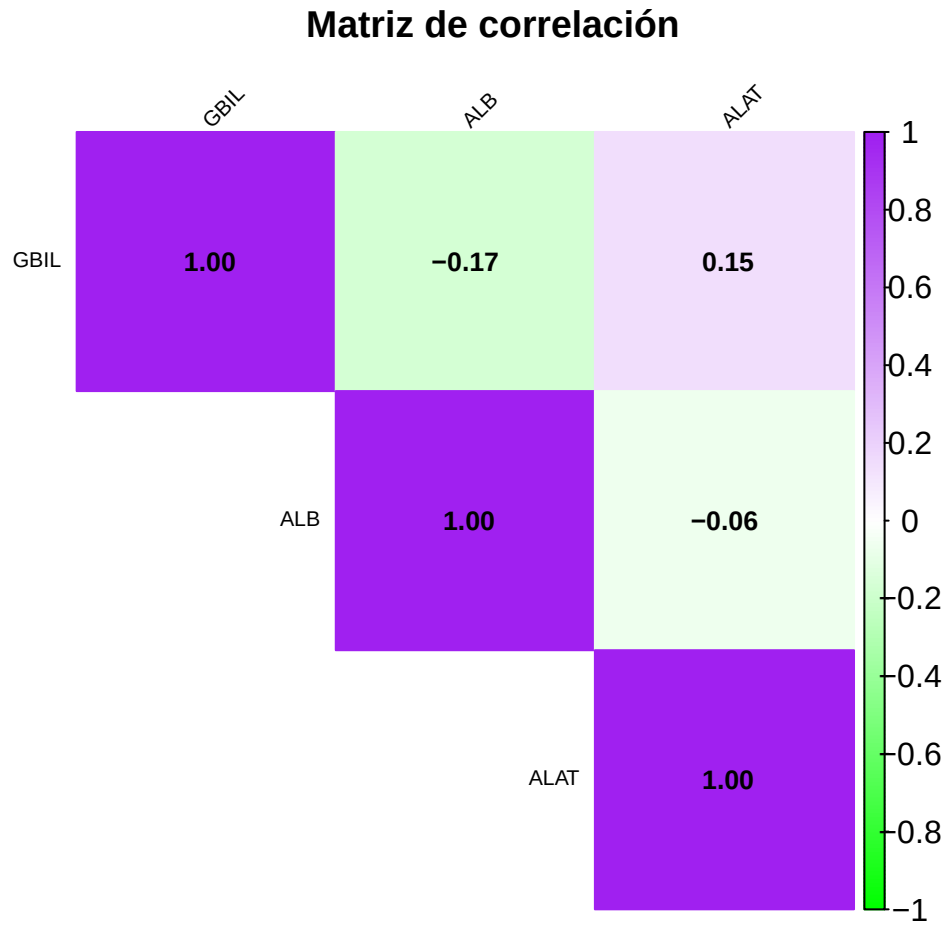


- Variables descartadas: TP, ASAT, CHE, GGT y AP

Correlacion tras descartar las variables:

```
# Correlación variables del tipo   Liver Function
vars_big <- c("GBIL", "ALB", "ALAT"
              )

cor_big <- cor(
  bacteremia_df[, vars_big],
  use = "complete.obs"
)

corrplot(
  cor_big,
  method = "color",
  type = "upper",
  col = colorRampPalette(c("green", "white", "purple"))(200),
  addCoef.col = "black",
  number.cex = 0.8,      # correlation numbers
  tl.col = "black",
  tl.cex = 0.7,          # SMALLER labels
  tl.srt = 45,           # ROTATED labels (45 degrees)
  cl.cex = 1.0,
```

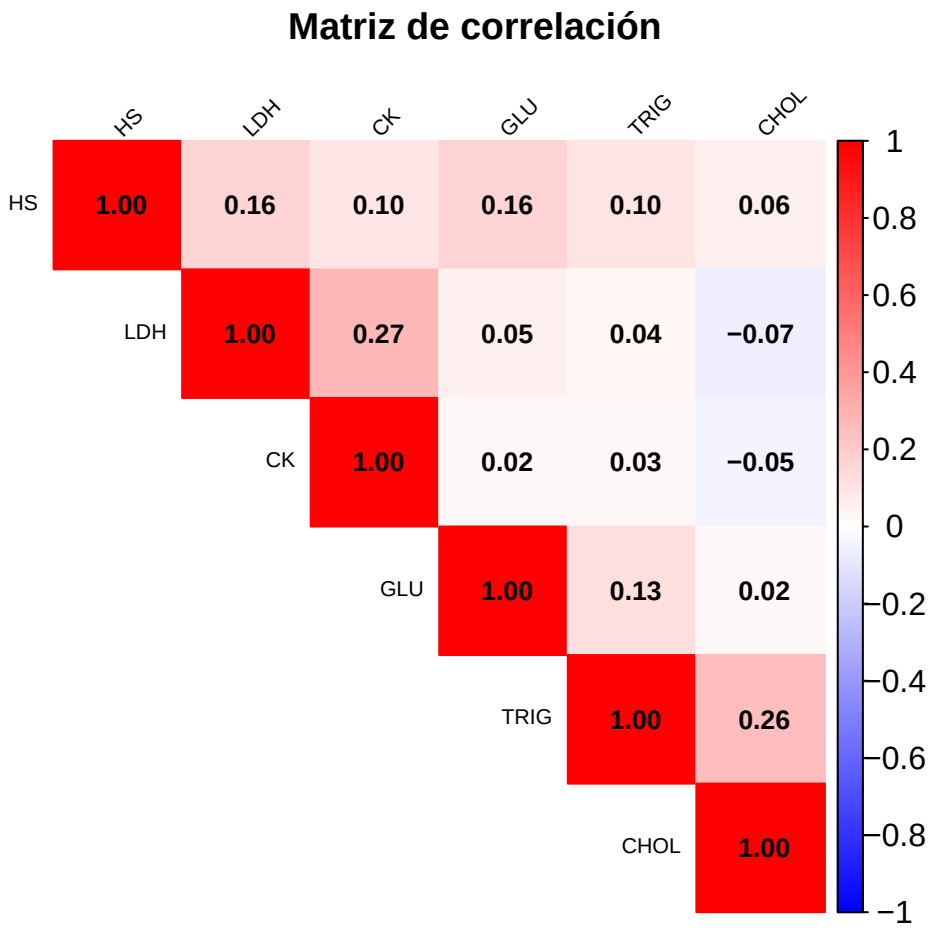
```
  title = "Matriz de correlación",
  mar = c(0, 0, 3, 0)
)
```



7. Variables del tipo Cardiac & Metabolic

```
# Correlación variables del tipo Cardiac & Metabolic
vars_big <- c("HS", "LDH", "CK", "GLU", "TRIG", "CHOL"
              )

cor_big <- cor(
  bacteremia_df[, vars_big],
  use = "complete.obs"
)

corrplot(
  cor_big,
  method = "color",
  type = "upper",
  col = colorRampPalette(c("blue", "white", "red"))(200),
  addCoef.col = "black",
  number.cex = 0.8,      # correlation numbers
  tl.col = "black",
```

```
  tl.cex = 0.7,          # SMALLER labels
  tl.srt = 45,           # ROTATED labels (45 degrees)
  cl.cex = 1.0,
  title = "Matriz de correlación",
  mar = c(0, 0, 3, 0)
)
```

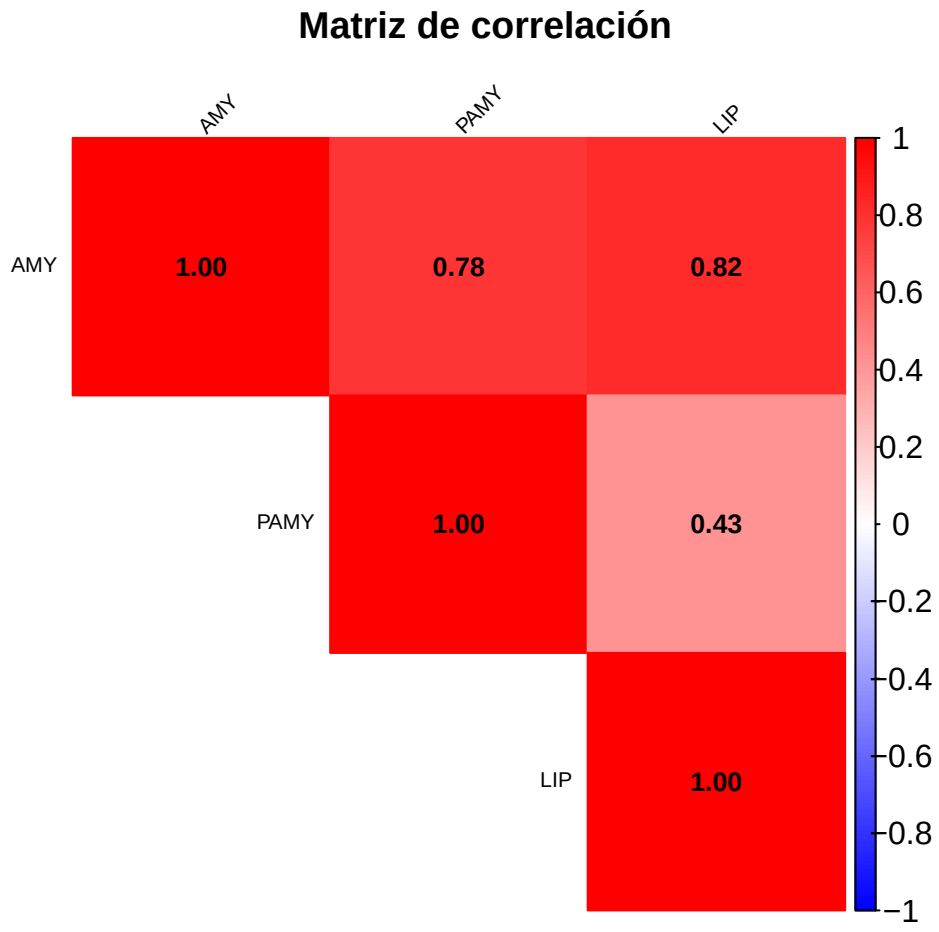


No descartamos variables al no estar a penas correlacionadas entre sí.

8. Variables del tipo Pancreatic Enzymes

```
# Correlación variables del tipo Cardiac & Metabolic
vars_big <- c("AMY", "PAMY", "LIP"
              )

cor_big <- cor(
  bacteremia_df[, vars_big],
  use = "complete.obs"
)

corrplot(
  cor_big,
  method = "color",
  type = "upper",
```

```
  col = colorRampPalette(c("blue", "white", "red"))(200),
  addCoef.col = "black",
  number.cex = 0.8,      # correlation numbers
  tl.col = "black",
  tl.cex = 0.7,          # SMALLER labels
  tl.srt = 45,           # ROTATED labels (45 degrees)
  cl.cex = 1.0,
  title = "Matriz de correlación",
  mar = c(0, 0, 3, 0)
)
```



- Variables descartadas: LIP y PAMY.

9. Variables del tipo Inflammatory Markers

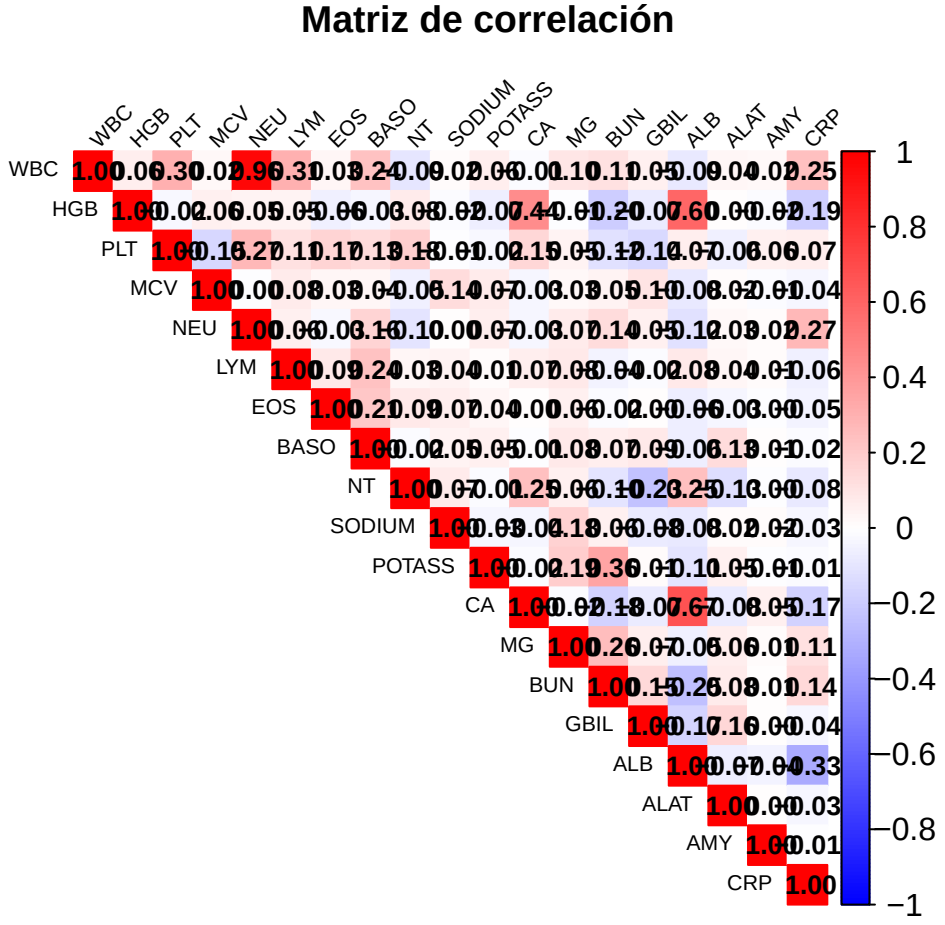
No la eliminamos al ser la única de su categoría.

10. Correlación entre categorías.

A continuación procedemos a estudiar las correlaciones con las variables restantes que no hemos descartado:

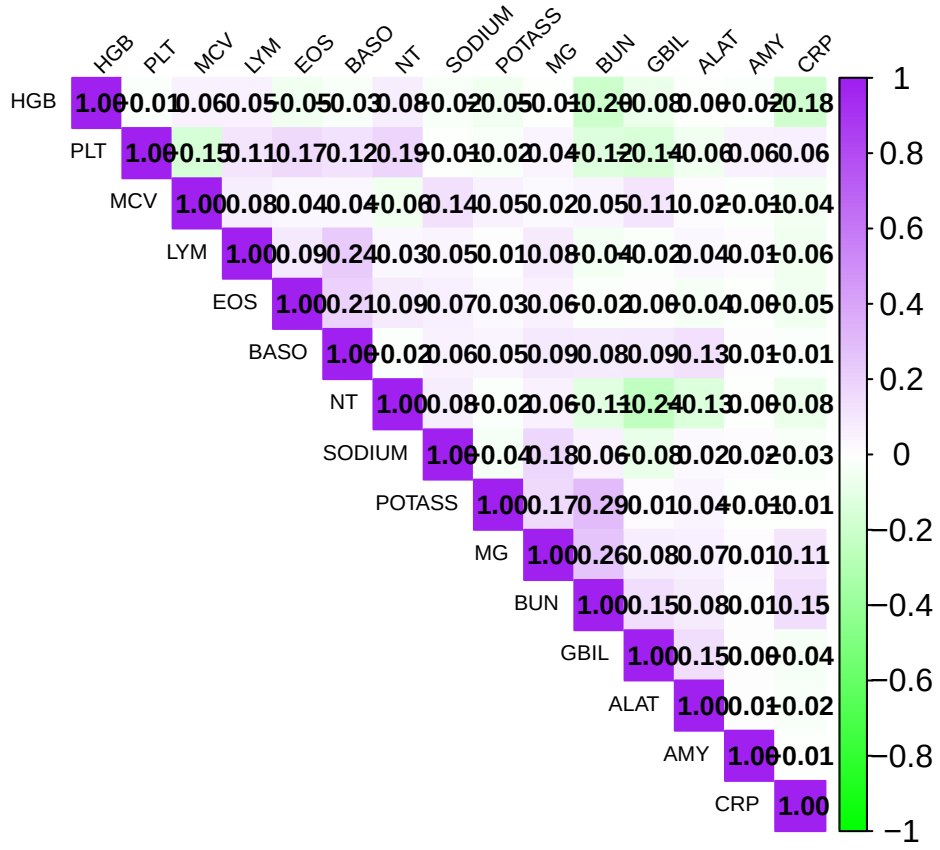
```
# Correlación todas
vars_big <- c("WBC", "HGB", "PLT", "MCV", "NEU", "LYM",  "EOS",  "BASO",
              "NT", "SODIUM", "POTASS", "CA", "MG", "BUN", "GBIL", "ALB",
```

```
              "ALAT", "AMY", "CRP"
              )

cor_big <- cor(
  bacteremia_df[, vars_big],
  use = "complete.obs"
)

corrplot(
  cor_big,
  method = "color",
  type = "upper",
  col = colorRampPalette(c("blue", "white", "red"))(200),
  addCoef.col = "black",
  number.cex = 0.8,      # correlation numbers
  tl.col = "black",
  tl.cex = 0.7,          # SMALLER labels
  tl.srt = 45,           # ROTATED labels (45 degrees)
  cl.cex = 1.0,
  title = "Matriz de correlación",
  mar = c(0, 0, 3, 0)
)
```



- Variables descartadas: WCB, NEU, CA y ALB

```r
# Correlación todas
vars_big <- c("HGB", "PLT", "MCV", "LYM",  "EOS",  "BASO",
              "NT", "SODIUM", "POTASS", "MG", "BUN", "GBIL",
              "ALAT", "AMY", "CRP"
              )

cor_big <- cor(
  bacteremia_df[, vars_big],
  use = "complete.obs"
)

corrplot(
  cor_big,
  method = "color",
  type = "upper",
  col = colorRampPalette(c("green", "white", "purple"))(200),
  addCoef.col = "black",
  number.cex = 0.8,      # correlation numbers
  tl.col = "black",
  tl.cex = 0.7,          # SMALLER labels
  tl.srt = 45,           # ROTATED labels (45 degrees)
  cl.cex = 1.0,
  title = "Matriz de correlación",
  mar = c(0, 0, 3, 0)
)
```



Observamos que tras el análisis de correlaciones se ha reducido considerablemente el número de variables, de 53 a 16, quedándonos únicamente con aquellas que presentan muy poca relación entre sí.

Este análisis exhausitivo de correlaciones es un paso previo necesario para poder realizar la sección 4 de modelos de aprendizaje automático, lo cual se desarrollará en dicha sección.

2.2.5 Limpieza de los datos.

```
feature_categories <- list(

  # Complete Blood Count (CBC) - Hematology
  hematology = c("WBC", "RBC", "HGB", "HCT", "PLT", "MCV", "MCH", "MCHC",
                 "RDW", "MPV", "PDW"),

  # White Blood Cell Differential
  wbc_differential = c("NEU", "NEUR", "LYM", "LYMR", "MONO", "MONOR",
                       "EOS", "EOSR", "BASO", "BASOR", "NT"),

  # Coagulation
  coagulation = c("APTT", "FIB"),

  # Electrolytes & Minerals
  electrolytes = c("SODIUM", "POTASS", "CA", "PHOS", "MG"),
```

```
  # Kidney Function
  kidney = c("CREA", "BUN"),

  # Liver Function
  liver = c("GBIL", "TP", "ALB", "AP", "ASAT", "ALAT", "GGT", "CHE"),

  # Cardiac & Metabolic
  cardiac_metabolic = c("HS", "LDH", "CK", "GLU", "TRIG", "CHOL"),

  # Pancreatic Enzymes
  pancreatic = c("AMY", "PAMY", "LIP"),

  # Inflammatory Markers
  inflammatory = c("CRP")
)

# Create category labels for each feature
feature_info <- data.frame(
  Variable = names(bacteremia_df),
  category = NA,
  stringsAsFactors = FALSE
)

for (cat_name in names(feature_categories)) {
  vars_in_cat <- feature_categories[[cat_name]]
  feature_info$category[feature_info$Variable %in% vars_in_cat] <- cat_name
}

# Combine with missing data info
feature_analysis <- missing_summary %>%
  left_join(feature_info, by = "Variable") %>%
  arrange(category, desc(Missing_Percent))

print(feature_analysis)
```

```
##        Variable Missing_Count Missing_Percent          category
## 1          TRIG          5061           34.45 cardiac_metabolic
## 2          CHOL          5045           34.34 cardiac_metabolic
## 3           GLU          4192           28.53 cardiac_metabolic
## 4            HS          3061           20.84 cardiac_metabolic
## 5            CK          2080           14.16 cardiac_metabolic
## 6           LDH          1714           11.67 cardiac_metabolic
## 7           FIB          2567           17.47       coagulation
## 8          APTT          2549           17.35       coagulation
## 9        POTASS          2008           13.67      electrolytes
## 10           MG          1869           12.72      electrolytes
## 11       SODIUM          1282            8.73      electrolytes
## 12           CA          1276            8.69      electrolytes
## 13         PHOS          1242            8.45      electrolytes
## 14          PDW          1102            7.50        hematology
## 15          MPV           702            4.78        hematology
## 16          RBC           461            3.14        hematology
## 17          WBC           462            3.14        hematology
```

```
## 18          RDW            56            0.38        hematology
## 19          MCV            42            0.29        hematology
## 20          HCT            42            0.29        hematology
## 21          PLT            42            0.29        hematology
## 22          MCH            42            0.29        hematology
## 23         MCHC            42            0.29        hematology
## 24          HGB            41            0.28        hematology
## 25          CRP           155            1.06      inflammatory
## 26          BUN           172            1.17            kidney
## 27         CREA           159            1.08            kidney
## 28          CHE          2447           16.66             liver
## 29          ALB          1676           11.41             liver
## 30           TP          1583           10.78             liver
## 31         GBIL          1441            9.81             liver
## 32           AP          1400            9.53             liver
## 33          GGT          1262            8.59             liver
## 34         ASAT          1154            7.86             liver
## 35         ALAT           987            6.72             liver
## 36         PAMY          7114           48.42        pancreatic
## 37          AMY          3913           26.64        pancreatic
## 38          LIP          3699           25.18        pancreatic
## 39           NT          2467           16.79  wbc_differential
## 40        BASOR           732            4.98  wbc_differential
## 41         EOSR           732            4.98  wbc_differential
## 42         LYMR           732            4.98  wbc_differential
## 43        MONOR           732            4.98  wbc_differential
## 44         NEUR           732            4.98  wbc_differential
## 45          NEU           728            4.96  wbc_differential
## 46          LYM           262            1.78  wbc_differential
## 47         MONO           246            1.67  wbc_differential
## 48         BASO           146            0.99  wbc_differential
## 49          EOS           135            0.92  wbc_differential
## 50           ID             0            0.00              <NA>
## 51          SEX             0            0.00              <NA>
## 52          AGE             0            0.00              <NA>
## 53 BloodCulture             0            0.00              <NA>
```

2.2.5.1 El algoritmo MICE La imputación múltiple mediante ecuaciones encadenadas es un método robusto e informativo para gestionar la falta de datos en conjuntos de datos. Este procedimiento rellena (imputa) los datos faltantes en un conjunto de datos mediante una serie iterativa de modelos predictivos. En cada iteración, cada variable especificada en el conjunto de datos se imputa utilizando las demás variables. Estas iteraciones deben ejecutarse hasta que se observe que se ha alcanzado la convergencia.

Fuga de datos:

MICE es especialmente útil si los valores faltantes están asociados con la variable objetivo de una forma que introduce fugas. Por ejemplo, supongamos que se desea modelar la retención de clientes en el momento del registro. Una variable se recopila en el momento del registro o un mes después. La ausencia de esa variable constituye una fuga de datos, ya que indica que el cliente no se retuvo durante un mes.

Análisis del embudo:

La información suele recopilarse en diferentes etapas de un embudo. MICE puede utilizarse para realizar estimaciones fundamentadas sobre las características de las entidades en diferentes puntos de un embudo.

Intervalos de confianza:

MICE puede utilizarse para imputar valores faltantes; sin embargo, es importante tener en cuenta que estos valores imputados constituyen una predicción. Crear múltiples conjuntos de datos con diferentes valores imputados permite realizar dos tipos de inferencia:

- Distribución de valores imputados: Se puede crear un perfil para cada valor imputado, lo que permite realizar afirmaciones sobre la distribución probable de dicho valor.
- Distribución de predicción del modelo: Con múltiples conjuntos de datos, se pueden crear múltiples modelos y una distribución de predicciones para cada muestra. Las muestras con valores imputados que no pudieron imputarse con mucha confianza presentarían una mayor varianza en sus predicciones.

Hacemos una copia del dataset y probamos solamente dos métodos.

```
bak_bacteremia_df <- bacteremia_df
```

El primer método es performar con MICE, tal y como explicado, sin embargo este método podría tardar muchas horas en ejecutarse.

Para ello comprobaremos más adelante el método de extracción de las características importantes o utilizaremos el método de reducción de dimensionalidad en SVD o PCA para poder extraer solamente las imporantes para realizar la imputación.

```
# Multiple imputation using MICE
# cat("\nPerforming Multiple Imputation...\n")
# imputed_data <- mice(bacteremia_df,
#                      m = 5,
#                      maxit = 50,
#                      method = 'pmm',
#                      seed = 42)
#
# # Extract first imputed dataset
# bacteremia_mice_imputed <- complete(imputed_data, 1)
#
# # Check imputation quality
# cat("\nMissing values after imputation:", sum(is.na(bacteremia_mice_imputed)), "\n")
```

2.3 Ejercicios de inferencia y simulación (1,5 puntos)

2.3.1 Basándoos en los conceptos trabajados en el LAB3, definid una función en R que realice algún tipo de cálculo de interés en el contexto del conjunto de datos.

Como hemos mencionado al inico, si la bacteremia es grave, puede provocar sepsis. En el contexto de la sepsis, la trombocitopenia (plaquetas bajas) y la anemia severa actúan como indicadores precoces de fallo orgánico y fragilidad hematológica. Debido a la importancia de estos dos marcadores en este contexto y al disponer de ellos en el conjunto de datos (variables HGB y PLT), se ha desarrollado la función *evaluar_riesgo*. Esta función permite identificar el riesgo de un paciente de sufrir anemia y trombocitopenia en función de los valores de estos indicadores:

```
evaluar_riesgo <- function(hgb, plt) {
  if(!is.numeric(hgb) | !is.numeric(plt)) {
    return("Error: Los parámetros deben ser numéricos")
  }
  if(hgb < 8 & plt < 100) {
    resultado <- "Riesgo Crítico: Anemia y Trombocitopenia severas"
  }
  else if(hgb < 8) {
    resultado <- "Riesgo Crítico: Anemia severa detectada"
  }
  else if(plt < 100) {
    resultado <- "Riesgo Crítico: Trombocitopenia severa detectada"
  }
  else if (hgb < 10 | plt < 150) {
    resultado <- "Riesgo Moderado: Monitorizar valores hematológicos"
  }
  else {
    resultado <- "Valores dentro de la normalidad"
  }
  return(resultado)
}
```

Para probar la consistencia de la función vamos a buscar en nuestro conjunto de datos valores de estos indicadores para cada uno de los casos contemplados:

```
fila_normal <- filter(bacteremia_df, HGB > 8 & PLT > 100)[1, c("HGB", "PLT")]
fila_mod <- filter(bacteremia_df,
              (HGB < 10 | PLT < 150)& HGB > 8 & PLT > 100)[1, c("HGB", "PLT")]
fila_anemia <- filter(bacteremia_df, HGB < 8 & PLT > 100)[1, c("HGB", "PLT")]
fila_tromb <- filter(bacteremia_df, HGB > 8 & PLT < 100)[1, c("HGB", "PLT")]
fila_ambos <- filter(bacteremia_df, HGB < 8 & PLT < 100)[1, c("HGB", "PLT")]

casos_criticos <- rbind(fila_normal,fila_mod,fila_anemia,fila_tromb,fila_ambos)
casos_criticos$Condicion_Riesgo <- mapply(evaluar_riesgo,
                                          casos_criticos$HGB,
                                          casos_criticos$PLT)

print(casos_criticos)
```

```
##    HGB PLT                                Condicion_Riesgo
## 1 11.5 307                 Valores dentro de la normalidad
```

```
## 2 15.7 144 Riesgo Moderado: Monitorizar valores hematológicos
## 3  7.3 188           Riesgo Crítico: Anemia severa detectada
## 4 10.8  38   Riesgo Crítico: Trombocitopenia severa detectada
## 5  7.4  64   Riesgo Crítico: Anemia y Trombocitopenia severas
```

Vemos si funciona el control de formato de valores de entrada:

```
evaluar_riesgo("12.7", 90)
```

```
## [1] "Error: Los parámetros deben ser numéricos"
```

```
evaluar_riesgo(12.7, 90)
```

```
## [1] "Riesgo Crítico: Trombocitopenia severa detectada"
```

Los umbrales de riesgo se han definido siguiendo los criterios de la OMS para la anemia grave (Hb < 8 g/dL) y los criterios SOFA para la evaluación del fallo orgánico en procesos infecciosos (Plaquetas < 100 G/L). Tambien, destacamos que aunque en el diccionario del dataset para la hemoglobina diga G/L, hemos visto en la sección de análisis exploratorio que los valores de la hemoglobina varían en un rango de 3 y 21 corresponden a g/dL.

2.3.2 Basándoos en los conceptos del LAB4 y la PEC2, plantead un mínimo de tres enunciados que respondan a una cuestión de probabilidad.

Para formular los enunciados de las siguientes cuestiones, primero vamos a obtener valores reales de nuestro conjunto de datos.

- Prevalencia de bacteriemia:

```
class_ratio <- prop.table(table(bacteremia_df$BloodCulture))
class_ratio
```

```
##
##         no        yes
## 0.91967871 0.08032129
```

La prevalencia de bacteramia es 8%.

- Proporción de hombres(1) y mujeres(2):

```
prop_sex <- prop.table(table(bacteremia_df$SEX))
prop_sex
```

```
##
##        1        2
## 0.581036 0.418964
```

- Proporciones de bacteramia en función del sexo:

```
sex_bact <- prop.table(table(bacteremia_df$BloodCulture, bacteremia_df$SEX))
sex_bact
```

```
##
##                1          2
##   no  0.53427268 0.38540603
##   yes 0.04676332 0.03355796
```

- Proporción MCV alto (mide el tamaño promedio de los glóbulos rojos):

```
df_MVC_alto<- filter(bacteremia_df, MCV>100)
proporcion <- nrow(df_MVC_alto)/nrow(bacteremia_df)
proporcion
```

```
## [1] 0.03852699
```

- Proporciones de bacteramia de los pacientes con MVC alto:

```
class_mvc <- prop.table(table(df_MVC_alto$BloodCulture))
class_mvc
```

```
##
##       no      yes
## 0.885159 0.114841
```

1. El 3.85% de los pacientes tienen un un volumen corpuscular medio (MCV) alto (>100). Si el 11.48% de los pacientes con MCV alto tienen bacteriemia ¿cuál es la probabilidad de que un paciente elegido al azar tenga ambos criterios?

Al tener:

- $P(MVC > 100) = 0.03852699$
- $P(Bacteriemia|MVC > 100) = 0.114841$

Aplicando la fórmula de la probabilidad condicionada obtenemos la probabilidad de la intersección de ambos sucesos:

$P(Bacteriemia MVC>100)= P(Bacteriemia|MVC>100) P(MVC>100) = 0.004424478 $

```
proporcion*class_mvc["yes"]
```

```
##         yes
## 0.004424478
```

Por tanto la probabilidad de tener volumen corpuscular medio alto y bacteriemia es del 0.44%.

2. Si la prevalencia de bacteriemia en el conjunto de pacientes es del 8,03% , si seleccionamos a 50 de ellos al azar, ¿cuál es la probabilidad de que al menos 5 tengan un cultivo positivo?

Al disponer de:

- n = 50 pacientes.
- p = 0.0803 probabilidad de tener bacteriemia (Éxito).
- k = 5 (al menos 5 éxitos).

Nos piden calcular $P(X \geq 5)$ siendo X una variable aleatoria X~Bin(n=250,p=0.0803).

```
pbinom(4, 20, class_ratio["yes"],lower.tail = FALSE)
```

```
## [1] 0.01863805
```

La probabilidad de que al menos 5 de esos 50 pacientes tengan bacteriemia es del 1.86%.

3. Sabemos que el 41.89% de los pacientes son mujeres y el 58.10% son hombres. Si la bacteriemia afecta al 4.68% de los hombres y al 3.36% de las mujeres, si seleccionamos un paciente con cultivo positivo, ¿cuál es la probabilidad de que sea mujer?

- $P(Hombre) = 0.581036$
- $P(Mujer) = 0.418964$
- $P(bacteriemia|Hombre) = 0.04676332$
- $P(bacteriemia|Mujer) = 0.03355796$

Como nos están pidiendo $P(Mujer|bacteriemia)$, por tanto aplicamos el teorema de Bayes:

$P(Mujer|bacteriemia) = \frac{P(bacteriemia|Mujer)\cdot P(Mujer)}{P(bacteriemia)} = 0.3409973$

Empelamos el teorema de la probabilidad total para calcular $P(bacteriemia)$:

$P(bacteriemia) = P(bacteriemia|Hombre)\cdot P(Hombre) + P(bacteriemia|Mujer)\cdot P(Mujer) = 0.04123075$

```
ph <- prop_sex[1]
pm <- prop_sex[2]
phb <- sex_bact[2]
phm <- sex_bact[4]
pbact <- ph*phb+pm*phm
cat("Probabilidad de tener bacteriemia:",pbact)
```

```
## Probabilidad de tener bacteriemia: 0.04123075
```

Finalmnete, tenemos que:

```
p_mujer_bact <- (pm*phm)/(pbact)
cat("Probabilidad de si un paciente tiene la enfermedad que sea mujer:",p_mujer_bact)
```

```
## Probabilidad de si un paciente tiene la enfermedad que sea mujer: 0.3409973
```

4. Un test rápido tiene una sensibilidad del 90% y la prevalencia es del 8.03%. ¿Cuál es la probabilidad de que un paciente esté realmente infectado si el test da positivo? (Asumiendo una especificidad del 95%).

Nota: los valores de la prevalencia y sensibilidad son inventados.

- $P(Infectado) = 0.0803$
- $P(Sano) = 1 - 0.0803 = 0.9197$
- Sensibilidad: $P(Positivo \mid Infectado) = 0.90$
- Especificidad: $P(Negativo \mid Sano) = 0.95 \Rightarrow P(Positivo \mid Sano) = 0.05$

Por tanto:

$P(Infectado \mid Positivo) = \frac{P(Positivo|Infectado) \cdot P(Infectado)}{P(Positivo)} =$

```
prev <- class_ratio["yes"]
sano <- 1-prev
sens <- 0.90
espf <- 0.95
fp <- 1-espf
prob <- (prev*sens)/(prev*sens+sano*fp)
cat("La probabilidad de estar infectado si el test da positivo es", prob)
```

```
## La probabilidad de estar infectado si el test da positivo es 0.6112054
```

2.3.3 Incluid un mínimo de un enunciado que corresponda a un breve modelo de simulación. Si vuestro conjunto de datos no facilita este tipo de enunciados, podéis generar una o varias distribuciones basándoos en parámetros determinados definidos por vosotros, afines al contexto del estudio.

Se observa que los niveles de hemoglobina (HGB) en pacientes con sospecha de bacteriemia siguen una distribución aproximadamente normal, una característica común en variables biológicas de grandes poblaciones. A partir de la media y la desviación típica obtenidas de nuestro conjunto de datos real, realizar una simulación de 10.000 casos utilizando una distribución normal. Estimar mediante simulacion la probabilidad de que un paciente presente un nivel de hemoglobina inferior a 10 g/dL (umbral de riesgo moderado/anemia) y comparar este resultado con la frecuencia observada en el dataset original para validar la precisión del modelo simulado.

En primer lugar, calculamos la media y la desviación típica de la variable HGB de nuestro dataset original:

```
media_hgb <- mean(bacteremia_df$HGB, na.rm = TRUE)
sd_hgb <- sd(bacteremia_df$HGB, na.rm = TRUE)

cat("Media:", media_hgb, "\n")
```

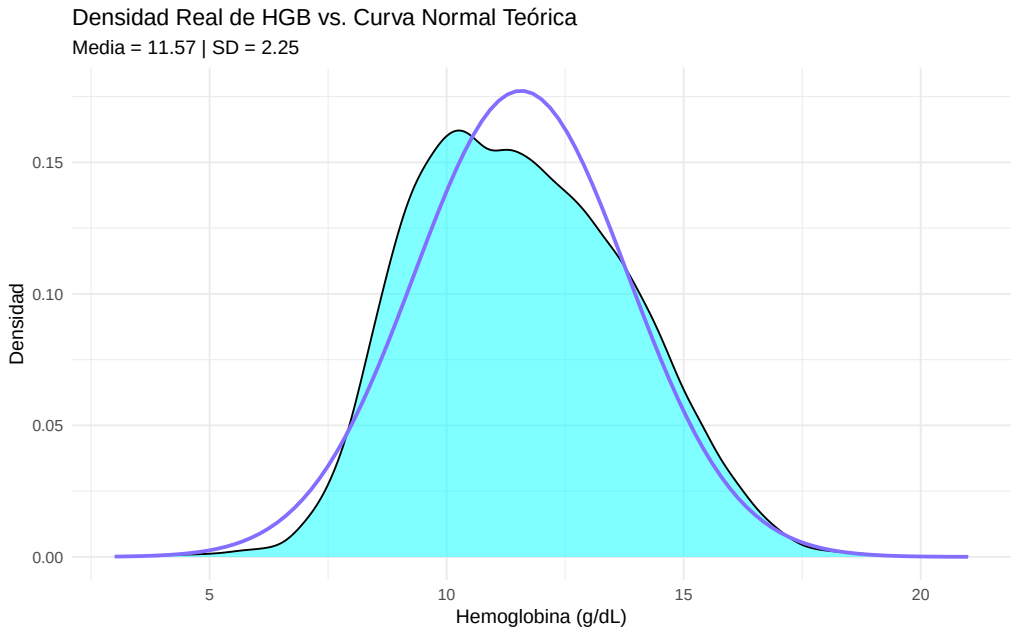
```
## Media: 11.56801
```

```
cat("Desviación Típica:", sd_hgb, "\n")
```

```
## Desviación Típica: 2.250767
```

Para verificar que esta variable sigue una distribución normal, graficamos su densidad frente a la curva de una normal con media y desviación típica ya calculadas de nuestra variable HGB:

```
library(ggplot2)

ggplot(bacteremia_df, aes(x = HGB)) +
  geom_density(fill = "cyan1", alpha = 0.5) +
  stat_function(fun = dnorm,
                args = list(mean = media_hgb, sd = sd_hgb),
                color = "slateblue1", size = 1) +
  labs(title = "Densidad Real de HGB vs. Curva Normal Teórica",
       subtitle = paste("Media =", round(media_hgb, 2), "| SD =", round(sd_hgb, 2)),
       x = "Hemoglobina (g/dL)", y = "Densidad") +
  theme_minimal()
```

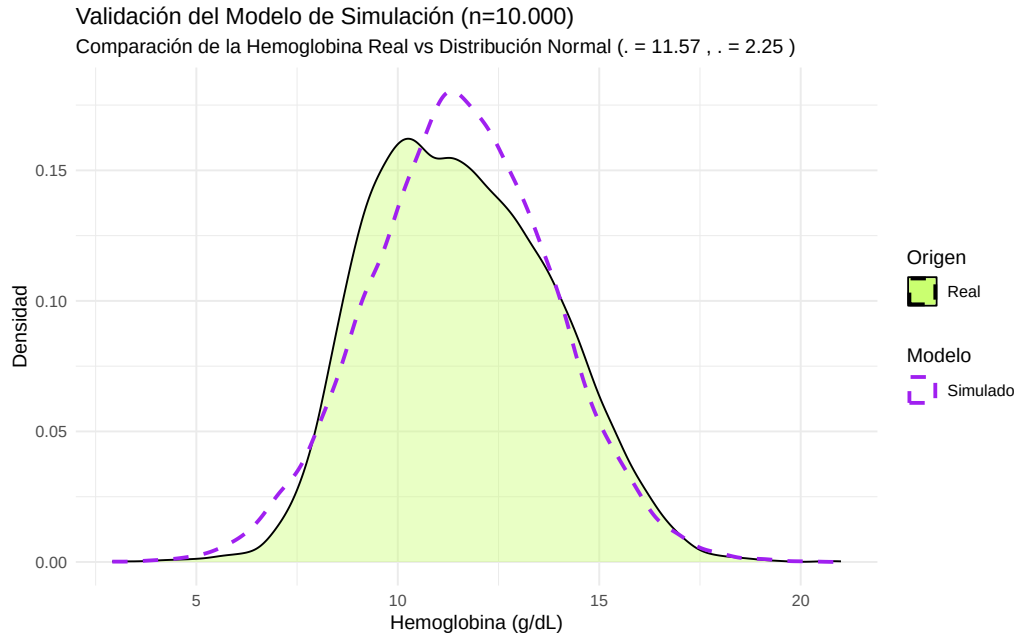


Efectivamente observamos que podemos asumir que la variable HGB se distribuye como una normal.

A continuación, simulamos una muestra de n=10.000 de una normal con media 11.56801 y desviación típica 2.250767. Mostramos gráficamente el resultado comparando la distribución de los valores reales frente a los obtenidos mediante simulación:

```
set.seed(123)
hgb_simulada <- rnorm(n = 10000, mean = media_hgb, sd = sd_hgb)
df_simulacion <- data.frame(HGB = hgb_simulada)

ggplot() +
  #Datos originales
  geom_density(data = bacteremia_df, aes(x = HGB, fill = "Real"), alpha = 0.4) +
  #Datos simulados
  geom_density(data = df_simulacion, aes(x = HGB, color = "Simulado"), size = 1, linetype = "dashed") +
```

```
  scale_fill_manual(name = "Origen", values = c("Real" = "darkolivegreen1")) +
  scale_color_manual(name = "Modelo", values = c("Simulado" = "purple")) +

  labs(title = "Validación del Modelo de Simulación (n=10.000)",
       subtitle = paste("Comparación de la Hemoglobina Real vs Distribución Normal (  =",
                        round(media_hgb, 2), ",   =", round(sd_hgb, 2), ")"),
       x = "Hemoglobina (g/dL)",
       y = "Densidad") +
  theme_minimal()
```



Podemos observar que salvo ligeras diferencias en la parte lateral izquierda y superior de la gráfica, los datos simulados se asemejan considerablemente a los valores reales.

Finalmente, calculamos la probabilidad de que un paciente presente un nivel de hemoglobina inferior a 10 g/dL tanto con nuestros datos reales como con los obtenidos mediante simulación:

```
prob_real <- sum(bacteremia_df$HGB < 10, na.rm = TRUE) / nrow(bacteremia_df)
prob_simulada <- sum(df_simulacion$HGB < 10) / 10000

cat("Frecuencia Real (HGB < 10):", round(prob_real, 4), "\n")
```

```
## Frecuencia Real (HGB < 10): 0.2645
```

```
cat("Probabilidad Simulada (HGB < 10):", round(prob_simulada, 4), "\n")
```

```
## Probabilidad Simulada (HGB < 10): 0.2418
```

Vemos que la diferencia entre el valor real y el valor simulado es de 0.02. Al ratarse de una diferencia mínima, podemos concluir que resultados obtenidos mediante simulación confirman la normalidad de la hemoglobina en este escenario.

---

3 Modelos de aprendizaje automático (2,5 puntos)

Resumen

Esta sección la dividiremos en dos secciones en las que se ha procedido de manera diferente.

1. Primer escenario:

- Se considerará el conjunto de datos eliminando las observaciones que contengan valores nulos.
- Las variables que se emplearán para entrenar los modelos han sido las seleccionadas tras el análisis de correlaciones de la sección 3.3.2.
- En esta sección hemos entrenado los modelos de Regresión logística, SMV y XGBoost para posteriormente comparar los resultados obtenidos.

2. Segundo escenario.

- Se considerará el conjunto de datos con todas las observacines.
- Se realizara un estudio de las características más relevantes a traves de un análisis de componentes principales (PCA) y extracción de features importance a traveés de randomforest.
- Una vez extraídas, se imputarán los valores nulos.
- En esta sección hemos entrenado el modelo de Regresión logística para posteriormente comparar los resultados obtenidos obtenidos en el primer escenario.

Finalmente, analizaremos los resultados obtenidos en ambos escenarios y extraeremos una conclusión final en la sección 6.

Consideraciones generales

Para ambos escenarios, debido a la naturaleza de la variable binaria `BloodCulture` hemos considerado adecuado un modelo de regresión logística.

A diferencia de la regresión lineal, este modelo no requiere normalidad ni homocedasticidad, pero para garantizar su validez en la predicción de bacteriemia se han verificado las siguientes condiciones fundamentales:

- **Respuesta binaria**: como ya hemos mencionado, la variable dependiente utilizada es BloodCulture_bin (obtenida a partir de BloodCulture), la cual clasifica a los pacientes de forma dicotómica en “0” (sin bacteriemia) y “1” (con bacteriemia).
- **Independencia de las observaciones**: como se menciona en la descripción del conjunto de datos, cada registro corresponde a observaciones individuales de pacientes distintos, garantizando que no exista dependencia entre las mediciones clínicas analizadas.
- **Multicolinealidad controlada**: se ha realizado un exhaustivo análisis de correlación mediante matrices de calor (heatmaps) por categorías. Para satisfacer esta condición, se han eliminado variables con coeficientes de correlación superiores a 0.30, evitando así la redundancia de información entre los predictores.

3.0.1 Primer escenario

3.0.1.1 Preparación modelo de datos Para poder preparar el baseline, por simplicidad eliminamos los registros que presentan valores nulos. Posteriormente se utilizarán alguna técnica de imputación de los valores nulos tal como indicamos en el punto 3.2.

```
bacteremia_cleaned <- na.omit(bacteremia_df)
head(bacteremia_cleaned)
```

```
##    ID SEX AGE   MCV  HGB  HCT PLT  MCH MCHC  RDW  MPV LYM MONO EOS BASO  NT
## 1   1   2  62  99.3 11.5 35.9 307 31.5 31.8 19.5 10.8 0.4  1.7 0.0  0.1  86
## 4   7   1  84  91.3 10.3 31.1 309 30.4 33.3 13.8  8.5 1.3  0.8 0.0  0.0  67
## 6  10   1  68 104.5 15.7 46.9 144 34.8 33.5 13.9 10.9 2.2  0.9 0.1  0.0  61
## 10 19   2  52  83.0 10.3 30.1 105 28.6 34.1 13.2 11.3 0.9  0.9 0.3  0.1  69
## 11 21   1  47  86.5  9.1 28.0 216 28.3 32.9 15.7 10.9 0.7  0.6 0.0  0.1 108
## 12 22   1  29  86.7  7.3 22.2 188 29.1 33.7 13.2 10.6 1.0  0.5 0.0  0.0  86
##    APTT FIB SODIUM POTASS   CA PHOS   MG CREA  BUN  HS GBIL   TP  ALB AMY PAMY
## 1  28.8 578    137   3.88 2.29 1.20 0.66 0.65  5.7 5.3 0.59 67.0 36.7  30   16
## 4  38.2 487    141   4.71 2.05 2.17 0.83 2.78 47.5 9.7 0.35 61.2 33.2  92   28
## 6  41.8 400    141   4.41 2.08 0.99 0.56 0.82 15.3 5.5 2.40 57.5 30.1  95   57
## 10 28.1 407    136   3.80 1.92 0.72 0.53 0.58 14.0 4.0 2.46 53.8 24.8  35   35
## 11 28.5 604    131   5.28 2.04 1.28 0.82 1.02 18.6 4.0 3.21 60.2 26.2  79   63
## 12 38.3 476    138   3.88 2.12 1.37 1.46 6.75 46.3 4.1 0.63 63.2 36.1  16   14
##    LIP  CHE  AP ASAT ALAT GGT  LDH   CK GLU TRIG CHOL  CRP     BASOR EOSR
## 1   10 5.12  85   22   14  48  284   23 107  105  175 3.94 0.4132231    0
## 4   18 4.10  94  774   72  23 1787 2422 105  134  141 3.78 0.0000000    0
## 6   25 6.79  68   32   11  68  263   75  89   85  144 5.89 0.0000000    1
## 10  38 2.64 146   97  156 192  277   87 104  207  123 3.80 1.6666667    5
## 11  52 2.61 180   24   63 266  221   30 104  292  194 3.04 1.0416667    0
## 12  19 2.85  64   13   23  19  299  118 102  221  151 6.79 0.0000000    0
##         LYMR     MONOR  NEU     NEUR  PDW RBC   WBC BloodCulture
## 1   1.652893  7.024793 22.0 90.90909 10.6 3.7 24.10           no
## 4  11.016949  6.779661  9.7 82.20339  8.7 3.5 11.58           no
## 6  22.000000  9.000000  6.8 68.00000 12.9 4.3  9.94           no
## 10 15.000000 15.000000  3.8 63.33333 13.2 3.5  5.98           no
## 11  7.291667  6.250000  8.2 85.41667 12.0 3.3  9.45           no
## 12 18.867925  9.433962  3.8 71.69811 11.9 2.5  5.26           no
##    BloodCulture_bin
## 1                 0
## 4                 0
## 6                 0
## 10                0
## 11                0
## 12                0
```

Una vez realizada la limpieza también comprobaremos como han quedado los valores respecto a la variable predictora y su proporción a diferencia del dataset original:

```
# Datos originales
prop.table(table(bacteremia_df$BloodCulture, bacteremia_df$SEX))*100
```

```
##
##               1         2
##   no  53.427268 38.540603
##   yes  4.676332  3.355796
```

```
# Sin valores nulos
prop.table(table(bacteremia_cleaned$BloodCulture, bacteremia_cleaned$SEX))*100
```

```
##
##                1         2
##   no  56.245288 35.687359
##   yes  4.548882  3.518472
```

Se observa una ligera disminución para el género femenino para los casos negativos, pero muy residual la diferencia.

Finalmente se analizan las diferencias de las observaciones iniciales respecto a los valores eliminados.

```
complete_data <- bacteremia_df %>% na.omit()
cat("Complete cases:", nrow(complete_data), "of", nrow(bacteremia_df),
    "(", round(100*nrow(complete_data)/nrow(bacteremia_df), 2), "%)\n")
```

```
## Complete cases: 3979 of 14691 ( 27.08 %)
```

Sin los datos nulos trabajaríamos solamente con una muestra del ~27% es decir unos casi 4.000 respecto a casi 15.000 observaciones iniciales.

3.0.1.2 Modelo con Regresión logística A continuación, usamos las variables seleccionadas en el apartado 3.1.2 del Análisis exploratorio de los datos:

- HGB, PLT, MCV, LYM, EOS, BASO, NT, SODIUM, POTASS, MG, BUN, GBIL, ALAT, AMY, CRP
- También incluimos las variables SEX y AGE ya que aportan infromación sobre el perfil del paciente. Esto evita que el modelo cometa errores al comparar, por ejemplo, los niveles de una persona de 20 años con los de una de 80. Vemos que además los valores de muchos de los indicadores se definen en función de la edad y el sexo.

```
# 1. Para manipulación de datos (mutate, select, across, %>% )
library(dplyr)
library(caTools)
library(caret)
# Lista de variables
vars_regresion <- c(
  "BloodCulture_bin", # Variable objetivo
  "AGE", "SEX",
  "HGB", "PLT", "MCV", "LYM", "EOS", "BASO", "NT",
  "SODIUM", "POTASS", "MG", "BUN", "GBIL", "ALAT", "AMY", "CRP"
)

# Selección y Escalado
#Escalamos las numéricas
df_modelo_final <- bacteremia_cleaned %>%
  select(all_of(vars_regresion)) %>%
  mutate(across(where(is.numeric) & !one_of("BloodCulture_bin"), scale))

# Dividimos 70 train y 30 test  para validar
set.seed(123)
split <- sample.split(df_modelo_final$BloodCulture_bin, SplitRatio = 0.7)
train_set <- df_modelo_final[split, ]
test_set  <- df_modelo_final[!split, ]
```

```r
# Modelo logístico
modelo_ajustado <- glm(
  BloodCulture_bin ~ .,
  data = train_set,
  family = binomial(link = "logit")
)

# Resumen modelo
summary(modelo_ajustado)
```

```
##
## Call:
## glm(formula = BloodCulture_bin ~ ., family = binomial(link = "logit"),
##     data = train_set)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.906005   0.104692 -27.758  < 2e-16 ***
## AGE          0.250082   0.082296   3.039 0.002375 **
## SEX          0.182529   0.074036   2.465 0.013686 *
## HGB          0.152189   0.080285   1.896 0.058013 .
## PLT         -0.032887   0.084512  -0.389 0.697170
## MCV          0.159785   0.079693   2.005 0.044962 *
## LYM         -1.289521   0.302851  -4.258 2.06e-05 ***
## EOS         -0.440942   0.147603  -2.987 0.002814 **
## BASO         0.053215   0.089709   0.593 0.553050
## NT           0.047713   0.072943   0.654 0.513038
## SODIUM      -0.155466   0.072958  -2.131 0.033098 *
## POTASS      -0.040588   0.071129  -0.571 0.568252
## MG          -0.248709   0.073729  -3.373 0.000743 ***
## BUN          0.305720   0.068250   4.479 7.48e-06 ***
## GBIL         0.224914   0.051727   4.348 1.37e-05 ***
## ALAT        -0.007713   0.072769  -0.106 0.915587
## AMY         -1.726733   1.073433  -1.609 0.107702
## CRP          0.300499   0.065630   4.579 4.68e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1563.6  on 2785  degrees of freedom
## Residual deviance: 1384.2  on 2768  degrees of freedom
## AIC: 1420.2
##
## Number of Fisher Scoring iterations: 8
```

```r
# Matriz de Confusión
probabilidades <- predict(modelo_ajustado, newdata = test_set, type = "response")
predicciones <- ifelse(probabilidades > 0.5, 1, 0)
confusionMatrix(as.factor(predicciones), as.factor(test_set$BloodCulture_bin))
```

```
## Confusion Matrix and Statistics
##
```

```
##           Reference
## Prediction    0    1
##          0 1094   95
##          1    3    1
##
##                Accuracy : 0.9179
##                  95% CI : (0.9008, 0.9328)
##     No Information Rate : 0.9195
##     P-Value [Acc > NIR] : 0.6102
##
##                   Kappa : 0.0137
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.99727
##             Specificity : 0.01042
##          Pos Pred Value : 0.92010
##          Neg Pred Value : 0.25000
##              Prevalence : 0.91953
##          Detection Rate : 0.91702
##    Detection Prevalence : 0.99665
##       Balanced Accuracy : 0.50384
##
##        'Positive' Class : 0
##
```

Se destacan los siguientes aspectos del modelo:

- Los factores más significativos (p-valor inferior a 0.05) son: AGE, SEX, HGB, MCV, LYM, EOS, SODIUM, MG, BUN, GBIL y CRP. El BUN (Riñón), la GBIL (Hígado) y la CRP (Inflamación) son los predictores más fuertes de bacteriemia. Por cada aumento en sus niveles, el riesgo de infección sube significativamente. Se destaca el papel de los Linfocitos (LYM) y el Magnesio (MG). Valores bajos de estos marcadores se asocian fuertemente con la presencia de bacterias en sangre.
- El modelo tiene una Accuracy del 91.79%, es decir, clasifica correctamente el 91.79% de los casos, sin embargo:
- El No Information Rate (0.9195): esto muestra si nuestro clasificador funciona mejor que la asignación aleatoria. Vemos que es muy elevado. Esto significa que si el modelo dijera siempre "NO" sin mirar los datos, acertaría el 91.95% de las veces.
- Sensibilidad (99.72%): es el número de predicciones positivas correctas dividido por el número total de positivos. en nuestro caso, como 'Positive' Class : 0, tenemos que tiene un porcentaje de acierto muy alto en el caso de que el diagnóstico real sea no tener bacteriemia. En la matriz de confusión observamos que se han clasificado correctamente en esta clase 1094 casos.
- Especificidad (1.04%): Muy baja. Observando la matriz de confusión, de 96 pacientes con bacteriemia real en el grupo de test, el modelo solo detectó a 1. A los otros 95 los marcó como sanos (Falsos Negativos). Este resultado es muy grave, ya que el coste de un Falso Negativo es demasiado alto ya que implicaría que el paciente no recibiría tratamiento cuando realmente si lo necesita, por lo que estamos poniendo en riesgo su salud.

Siguiendo las mismas directrices con el baseline, para analizar los primeros resultados de los algoritmos del método supervisado de clasificación, realizamos un modelo SMV y otro con XGBoost para comparar brevemente los resultados obtenidos con la regresión logística:

Teniendo en cuenta que las clases están desbalanceadas, tenemos un 8% de los casos positivos respecto a un 92% de los casos negativos, volveremos a realizar un ajuste al modelo aplicando los pesos.

```
### Aplicando WEIGHTS a la clase minoritaria, los casos positivos.

# Lista de variables
vars_regresion <- c(
  "AGE", "SEX",
  "HGB", "PLT", "MCV", "LYM", "EOS", "BASO", "NT",
  "SODIUM", "POTASS", "MG", "BUN", "GBIL", "ALAT", "AMY", "CRP"
)

# Creamos la variable target a factor
bacteremia_cleaned <- bacteremia_cleaned %>%
    mutate(BloodCulture = factor(BloodCulture, levels = c("no", "yes")))

# Selección y Escalado
#Escalamos las numéricas
df_modelo_final <- bacteremia_cleaned %>%
  select(all_of(vars_regresion), BloodCulture) %>%
  mutate(across(where(is.numeric), scale))

# Dividimos 70 train y 30 test  para validar
set.seed(123)
split <- createDataPartition(
  y = df_modelo_final$BloodCulture,
  p = 0.7,
  list = FALSE
)

train_set <- df_modelo_final[split, ]
test_set  <- df_modelo_final[-split, ]

p_yes <- mean(train_set$BloodCulture == "yes")
p_no  <- mean(train_set$BloodCulture == "no")

weights <- ifelse(
  train_set$BloodCulture == "yes",
  1 / p_yes,
  1 / p_no
)


# Modelo logístico
modelo_ajustado <- glm(
  BloodCulture ~ .,
  data = train_set,
  family = binomial(link = "logit"),
  weights = weights
)

# Resumen modelo
summary(modelo_ajustado)
```

```
##
## Call:
## glm(formula = BloodCulture ~ ., family = binomial(link = "logit"),
##     data = train_set, weights = weights)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.334036   0.032545 -10.264  < 2e-16 ***
## AGE          0.268077   0.032244   8.314  < 2e-16 ***
## SEX          0.091267   0.029865   3.056 0.002243 **
## HGB          0.012270   0.032035   0.383 0.701700
## PLT         -0.131654   0.033644  -3.913 9.11e-05 ***
## MCV          0.072100   0.031089   2.319 0.020388 *
## LYM         -0.364496   0.074811  -4.872 1.10e-06 ***
## EOS         -0.299544   0.039993  -7.490 6.89e-14 ***
## BASO        -0.105121   0.035667  -2.947 0.003205 **
## NT           0.097294   0.031184   3.120 0.001809 **
## SODIUM      -0.216216   0.030982  -6.979 2.98e-12 ***
## POTASS      -0.105958   0.030632  -3.459 0.000542 ***
## MG          -0.292075   0.032136  -9.089  < 2e-16 ***
## BUN          0.342475   0.034482   9.932  < 2e-16 ***
## GBIL         0.298624   0.034042   8.772  < 2e-16 ***
## ALAT        -0.005752   0.034497  -0.167 0.867578
## AMY         -1.034499   0.294774  -3.509 0.000449 ***
## CRP          0.232735   0.030012   7.755 8.86e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 7724.4  on 2785  degrees of freedom
## Residual deviance: 6803.2  on 2768  degrees of freedom
## AIC: 6459
##
## Number of Fisher Scoring iterations: 6
```

```r
# Matriz de Confusión
probabilidades <- predict(modelo_ajustado, newdata = test_set, type = "response")
#predicciones <- ifelse(probabilidades > 0.5, "yes", "no")

threshold <- 0.35   # try 0.25-0.45
predicciones <- factor(
  ifelse(probabilidades > threshold, "yes", "no"),
  levels = c("yes", "no")
)

confusionMatrix(
  predicciones,
  test_set$BloodCulture,
  positive = "yes"
)
```

```
## Confusion Matrix and Statistics
##
```

```
##          Reference
## Prediction  no yes
##        no  418  13
##        yes 679  83
##
##               Accuracy : 0.4199
##                 95% CI : (0.3918, 0.4485)
##    No Information Rate : 0.9195
##    P-Value [Acc > NIR] : 1
##
##                  Kappa : 0.059
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.86458
##             Specificity : 0.38104
##          Pos Pred Value : 0.10892
##          Neg Pred Value : 0.96984
##              Prevalence : 0.08047
##          Detection Rate : 0.06957
##    Detection Prevalence : 0.63873
##       Balanced Accuracy : 0.62281
##
##        'Positive' Class : yes
##
```

Finalmente verificando los resultados y la curva de AUC tenemos el siguiente resultado:

```
library(pROC)
roc_obj <- roc(test_set$BloodCulture, probabilidades, levels = c("no", "yes"))
auc(roc_obj)
```

```
## Area under the curve: 0.714
```

Un 71.4% de probabilidad de clasificar correctamente los casos positivos.

De hecho analizando los resultados con los pesos aplicados tenemos el siguiente escenario:

- TP (correct yes) = 83
- FN (missed yes) = 13
- FP (false alarm) = 679
- TN (correct no) = 418

Con una sensivitad del 86%, es decir los casos positivos son correctamente clasificados y solamente 13 han sido falsos negativos.

Para la especificidad clasifica un 38%, es decir los casos que no son y han sido clasificados, los falsos positivos.

En general un accuracy del 62% en clasificación como punto de partida está bastante bien, sin considerar otros factores y aspectos.

```r
library(e1071)  # Para el modelo SVM
library(caTools)
library(caret)

vars_seleccionadas <- c(
  "BloodCulture_bin", "AGE", "SEX", "HGB", "PLT", "MCV",
  "LYM", "EOS", "BASO", "NT", "SODIUM", "POTASS", "MG",
  "BUN", "GBIL", "ALAT", "AMY", "CRP"
)

df_svm <- bacteremia_cleaned %>%
  select(all_of(vars_seleccionadas)) %>%
  # Convertimos la respuesta a factor para que SVM haga CLASIFICACIÓN
  mutate(BloodCulture_bin = as.factor(BloodCulture_bin)) %>%
  mutate(across(where(is.numeric), scale))

set.seed(123)
split <- sample.split(df_svm$BloodCulture_bin, SplitRatio = 0.7)
train_svm <- df_svm[split, ]
test_svm  <- df_svm[!split, ]

modelo_svm <- svm(
  BloodCulture_bin ~ .,
  data = train_svm,
  kernel = "radial",
  probability = TRUE
)

pred_svm <- predict(modelo_svm, newdata = test_svm)

confusionMatrix(pred_svm, test_svm$BloodCulture_bin)
```

3.0.1.3 Modelo SMV

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1097   96
##          1    0    0
##
##                Accuracy : 0.9195
##                  95% CI : (0.9026, 0.9343)
##     No Information Rate : 0.9195
##     P-Value [Acc > NIR] : 0.5271
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 1.0000
##             Specificity : 0.0000
```

```
##          Pos Pred Value : 0.9195
##          Neg Pred Value :    NaN
##              Prevalence : 0.9195
##          Detection Rate : 0.9195
##    Detection Prevalence : 1.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : 0
##
```

Observamos que con este modelo, ni siquiera se clasifica correctamente ni un solo caso de tener bacteriemia. Presenta una especificidad de 0.00.

Nuevamente con el Soporte de máquinas de vectores sin utilizar el parámetro de `weights` no clasifica los casos positivos de forma correcta.

3.0.1.4 Modelo XGBoost Vemos que los resultados han mejorado considerablemente, sin embargo, en el contexto de detección de esta enfermedad sigue sin ser suficiente ya que de 96 casos solo se han clasificado correctamente 14.

3.0.2 Segundo escenario

En este segundo esceneario, conociendo que podemos adoptar un peso para los casos positivos de la variable dependiente, donde recordamos tenemos solo un 8% de los casos pero muy pocas muestras, seguiremos con un escenario del estudio de las features más importantes a través reducción de dimensionalidad y finalmente imputar las características.

```
# A FALTA DE REALIZAR LAS IMPUTACIONES
bacteremia_imputed <- bacteremia_cleaned

# Preparamos el dataset para el modelo
model_data <- bacteremia_imputed %>%
  select(-ID) %>%
  select(where(is.numeric), BloodCulture)

# Comparamos los datos según la variable target
table(model_data$BloodCulture)
```

3.0.2.1 Preparación modelo de datos

```
##
##   no  yes
## 3658  321
```

3.0.2.2 Estudio de las características - Feature importance Como vimos al apartado 3.1.3 seleccionaremos las features más importantes que tengan un score superior al 0.80 y comparando con aquellos que mínimo tienen un 0.65.

```
# Calcularemos la matriz de correlaciones
cor_matrix <- cor(bacteremia_cleaned %>%
                  select(where(is.numeric), -ID),
                  use = "complete.obs")

# Mostraremos la matriz y la guardaremos en formato PDF
pdf("correlation_matrix.pdf", width = 14, height = 12)
corrplot(cor_matrix, method = "color", type = "upper",
         tl.cex = 0.7, tl.col = "black",
         title = "Feature Correlation Matrix",
         mar = c(0,0,1,0))
dev.off()
```

```
## pdf
##   2
```

```
# Buscamos las features importantes superior al 0.8
high_cor <- findCorrelation(cor_matrix, cutoff = 0.8, names = TRUE)
cat("\nHighly correlated features (>0.8):\n")
```

```
##
## Highly correlated features (>0.8):
```

```
print(high_cor)
```

```
## [1] "HGB"  "RBC"  "NEU"  "NEUR" "ASAT" "MPV"  "MCV"  "EOS"  "LIP"
```

```
# Buscamos las features importantes superior al 0.65
med_cor <- findCorrelation(cor_matrix, cutoff = 0.65, names = TRUE)
cat("\nMedium correlated features (>0.65):\n")
```

```
##
## Medium correlated features (>0.65):
```

```
print(med_cor)
```

```
##  [1] "ALB"  "HGB"  "RBC"  "BUN"  "NEU"  "NEUR" "CRP"  "ASAT" "BASO" "MPV"
## [11] "MCH"  "EOS"  "LIP"  "PAMY"
```

Las `features importance` por correlación y con un valor superior a 0.80 son estas 9 características:

- "HGB" "RBC" "NEU" "NEUR" "ASAT" "MPV" "MCV" "EOS" "LIP"

Si observamos con un mínimo de 0.65, tendríamos 14 características.

- "ALB" "HGB" "RBC" "BUN" "NEU" "NEUR" "CRP" "ASAT" "BASO" "MPV" "MCH" "EOS" "LIP" "PAMY"

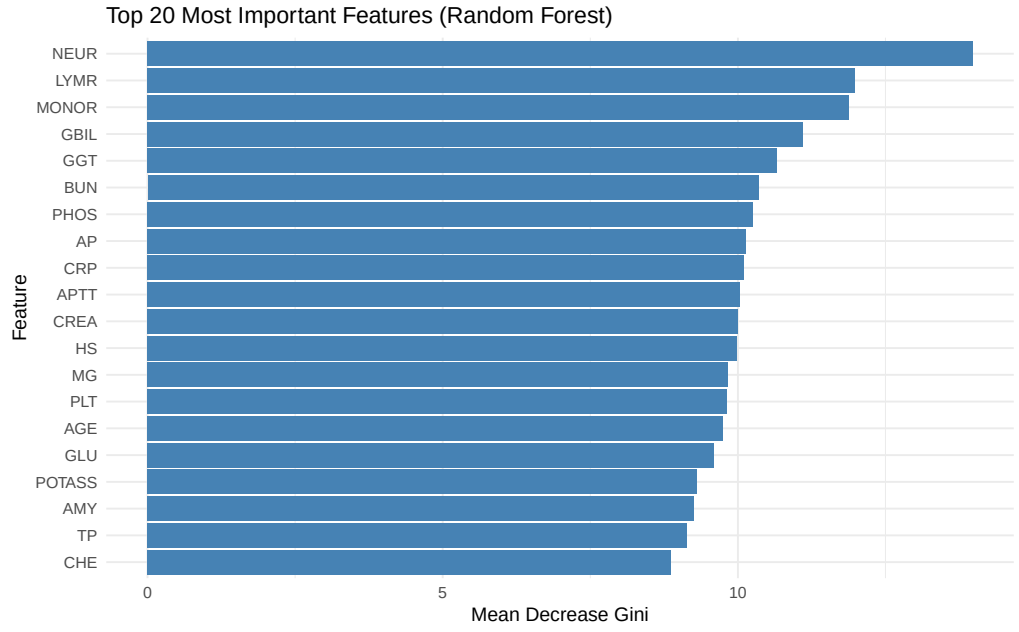
Para poder realizar un análisis exhaustivo de las características, optaremos para modelar con el algoritmo `Random Forest` y extraer las *features importance* y también a tráves del `análisis de componenentes principales (PCA)`

Extracción de features importance a través de randomforest

```
model_data <- model_data %>%
  select(-BloodCulture_bin) %>%
  mutate(
    BloodCulture = as.factor(BloodCulture),
    across(where(is.character), as.factor)
  )

# 2. DESPUÉS hacemos la partición (ahora train_data heredará el formato factor)
set.seed(123)
train_idx <- createDataPartition(model_data$BloodCulture, p = 0.7, list = FALSE)
train_data <- model_data[train_idx, ]
test_data  <- model_data[-train_idx, ]

# Entrenamos con el modelo Random Forest
cat("\nTraining Random Forest for feature importance...\n")
```

```
##
## Training Random Forest for feature importance...
```

```r
set.seed(123)
rf_model <- randomForest(
  BloodCulture ~ .,
  data = train_data,
  ntree = 500,
  importance = TRUE,
  na.action = na.omit
)

# Extraeremos las features importantes
importance_df <- as.data.frame(importance(rf_model)) %>%
  rownames_to_column("Feature") %>%
  arrange(desc(MeanDecreaseGini))

# Visualizaremos las 20 features más importantes
top_features <- importance_df %>%
  slice_head(n = 20)

ggplot(top_features, aes(x = reorder(Feature, MeanDecreaseGini),
                         y = MeanDecreaseGini)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  coord_flip() +
  labs(title = "Top 20 Most Important Features (Random Forest)",
       x = "Feature", y = "Mean Decrease Gini") +
  theme_minimal()
```




```r
ggsave("feature_importance_rf.pdf", width = 10, height = 8)
```

A través de las `TOP Features` seleccionaremos las características con una puntuación de `MeanDecreaseGini` que tienen un valor superior a 10.

```r
top_features %>%
  select(Feature, MeanDecreaseGini) %>%
  filter(MeanDecreaseGini > 10)
```

```
##    Feature MeanDecreaseGini
## 1     NEUR         13.97442
## 2     LYMR         11.98047
## 3    MONOR         11.88990
## 4     GBIL         11.11065
## 5      GGT         10.66087
## 6      BUN         10.34916
## 7     PHOS         10.25692
## 8       AP         10.13713
## 9      CRP         10.09736
## 10    APTT         10.04062
```

Para poder explicar exactamente los valores extraidos del `MeanDecreaseGini` de nuestro estudio, el **Gini** es una medida de "impureza" o "desorden" en un nodo del árbol de decisión. Básicamente mide cuánto ayuda cada variable a "purificar" o separar mejor las clases, que en nuestro caso es BloodCulture "yes" vs "no".

De los resultados obtenidos tenemos:

cuando una variable, como el caso de `NEUR` divide bien este grupo:

Grupo A: 90% negativos, 10% positivos → más "puro" Grupo B: 20% negativos, 80% positivos → más "puro"

El `Mean Decrease Gini` = cuánto disminuye la impureza cuando usamos esa variable para dividir.

De los 4 resltados superior al 10:

- NEUR (Neutrófilos %):

Cada vez que Random Forest usa NEUR para dividir los datos, reduce la impureza en promedio 13.97 unidades, separando muy bien pacientes con bacteremia positiva vs negativa. Esto tiene sentido clínico ya que los neutrófilos aumentan en infecciones bacterianas

- LYMR (Linfocitos %):

Los linfocitos también son indicadores inmunológicos clave

- MONOR (Monocitos %):

Parte del recuento diferencial de glóbulos blancos

- GBIL (Bilirrubina) y GGT (Gamma-glutamil transferasa):

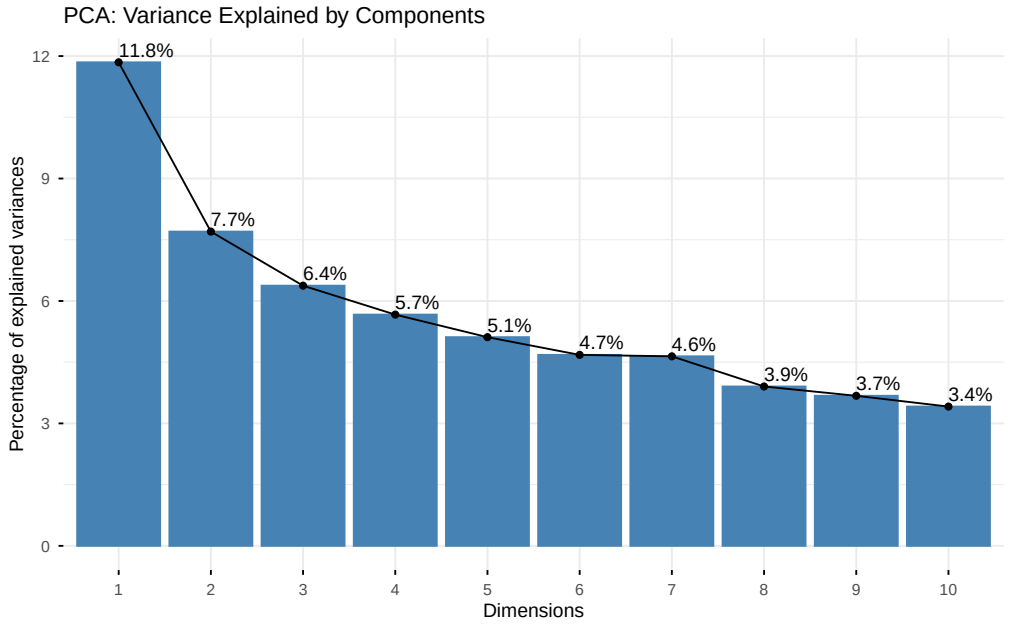
Marcadores hepáticos que pueden alterarse en sepsis

3.0.2.3 Análisis de componentes principales (PCA) Tanto el análisis de componentes principales, principal component analysis (PCA) en inglés, como la descomposición de valores singulares, singular value decomposition (SVD) en inglés, son técnicas que nos permitan trabajar con nuevas características llamadas componentes, que ciertamente son independientes entre sí. En realidad, estas dos técnicas nos permiten representar el juego de datos en un nuevo sistema de coordenadas que denominamos componentes principales. Este sistema está mejor adaptado a la distribución del juego de datos, de forma que recoge mejor su variabilidad.

Aplicamos el análisis de componentes principales al dataset. Empezamos ejecutando la función `PCA` y `prcomp()`.

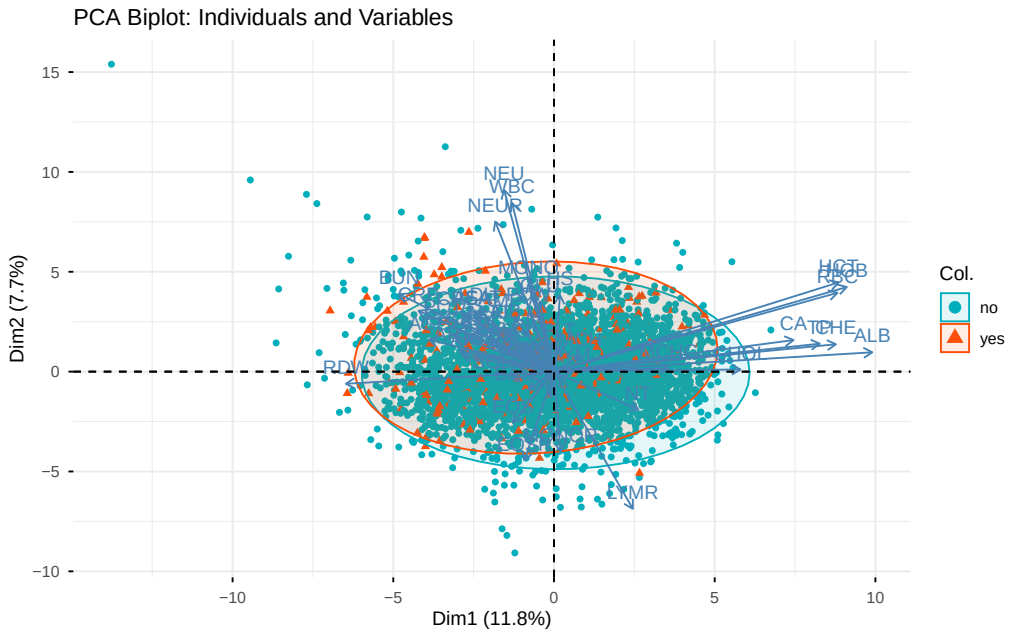
```
# Función de normalización Min-Max
nor <-function(x) { (x -min(x))/(max(x)-min(x))}

# Aplicando función Z-Score
z_score <- function(x) {(x - mean(x)) / sd(x)}

# Aplicamos la normalización a nuestro modelo de datos
temp_df <- model_data %>%
  select(-BloodCulture)
model_data_minmax_norm <- as.data.frame(lapply(temp_df, nor))
model_data_z_norm <- as.data.frame(lapply(temp_df, z_score))
```

```
# Perform PCA
pca_result <- PCA(model_data_z_norm, scale.unit = FALSE, graph = FALSE)

# Scree plot - variance explained
fviz_eig(pca_result, addlabels = TRUE,
         main = "PCA: Variance Explained by Components")
```

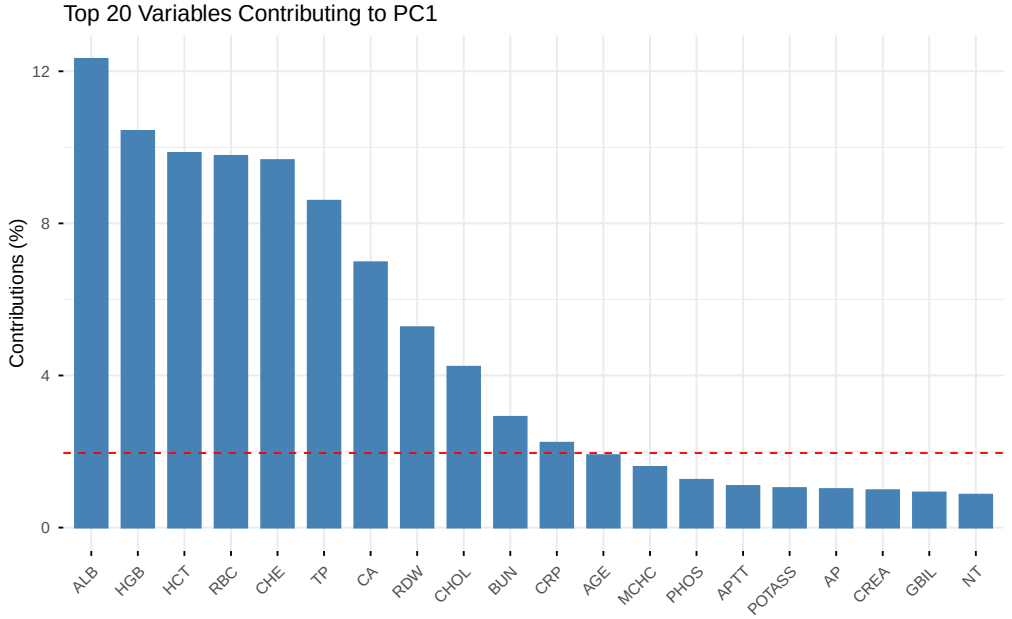



```
ggsave("pca_scree_plot.pdf", width = 10, height = 6)

# Biplot - first two components
```

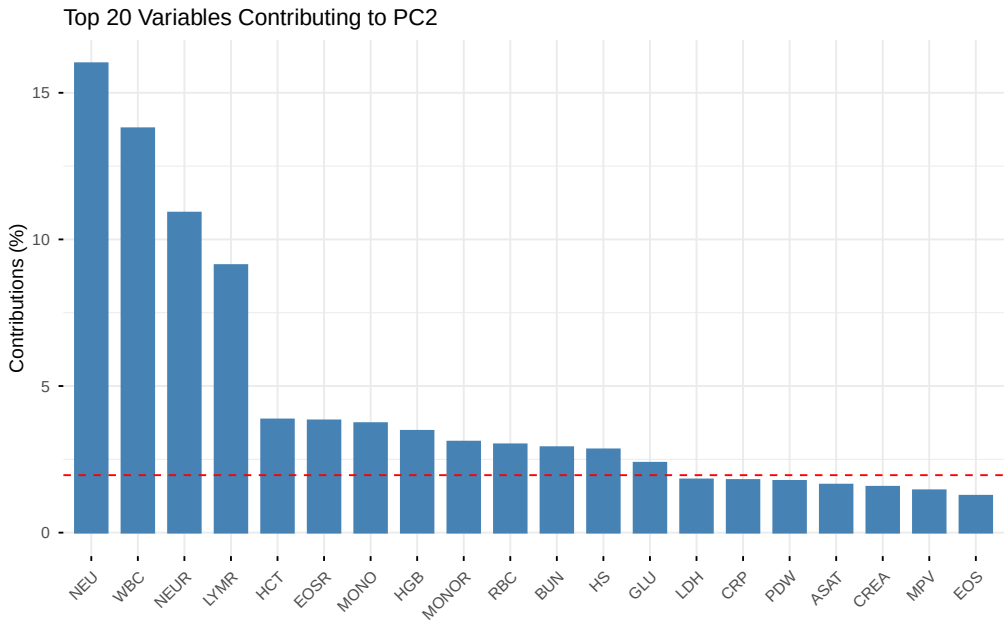
```r
fviz_pca_biplot(pca_result,
                label = "var",
                col.ind = model_data$BloodCulture,
                palette = c("#00AFBB", "#FC4E07"),
                addEllipses = TRUE,
                title = "PCA Biplot: Individuals and Variables")
```




```r
ggsave("pca_biplot.pdf", width = 12, height = 10)

# Variable contributions to PC1 and PC2
fviz_contrib(pca_result, choice = "var", axes = 1, top = 20,
             title = "Top 20 Variables Contributing to PC1")
```



```
ggsave("pca_contrib_pc1.pdf", width = 10, height = 8)

fviz_contrib(pca_result, choice = "var", axes = 2, top = 20,
             title = "Top 20 Variables Contributing to PC2")
```




```
ggsave("pca_contrib_pc2.pdf", width = 10, height = 8)

# Summary of PCA
summary_pca <- summary(pca_result)
```

```
##
## Call:
## PCA(X = model_data_z_norm, scale.unit = FALSE, graph = FALSE)
##
##
## Eigenvalues
##                        Dim.1   Dim.2   Dim.3   Dim.4   Dim.5   Dim.6   Dim.7
## Variance               6.040   3.926   3.250   2.889   2.607   2.386   2.368
## % of var.             11.847   7.699   6.374   5.666   5.113   4.679   4.643
## Cumulative % of var.  11.847  19.546  25.920  31.586  36.699  41.378  46.022
##                        Dim.8   Dim.9  Dim.10  Dim.11  Dim.12  Dim.13  Dim.14
## Variance               1.991   1.874   1.740   1.560   1.482   1.353   1.232
## % of var.              3.904   3.676   3.413   3.059   2.906   2.653   2.416
## Cumulative % of var.  49.926  53.602  57.015  60.074  62.980  65.633  68.049
##                       Dim.15  Dim.16  Dim.17  Dim.18  Dim.19  Dim.20  Dim.21
## Variance               1.175   1.102   1.081   0.926   0.908   0.867   0.833
## % of var.              2.305   2.161   2.120   1.816   1.782   1.701   1.634
## Cumulative % of var.  70.354  72.515  74.634  76.450  78.232  79.932  81.566
##                       Dim.22  Dim.23  Dim.24  Dim.25  Dim.26  Dim.27  Dim.28
## Variance               0.777   0.734   0.688   0.636   0.595   0.569   0.528
## % of var.              1.524   1.439   1.350   1.247   1.167   1.116   1.036
## Cumulative % of var.  83.090  84.529  85.878  87.126  88.292  89.409  90.445
```

```
##                        Dim.29  Dim.30  Dim.31  Dim.32  Dim.33  Dim.34  Dim.35
## Variance                0.482   0.476   0.461   0.422   0.398   0.370   0.359
## % of var.               0.945   0.934   0.905   0.828   0.781   0.726   0.705
## Cumulative % of var.   91.390  92.325  93.229  94.058  94.839  95.565  96.270
##                        Dim.36  Dim.37  Dim.38  Dim.39  Dim.40  Dim.41  Dim.42
## Variance                0.347   0.273   0.227   0.207   0.194   0.176   0.132
## % of var.               0.680   0.536   0.446   0.407   0.380   0.345   0.260
## Cumulative % of var.   96.950  97.486  97.931  98.338  98.718  99.063  99.323
##                        Dim.43  Dim.44  Dim.45  Dim.46  Dim.47  Dim.48  Dim.49
## Variance                0.115   0.108   0.056   0.043   0.016   0.003   0.002
## % of var.               0.226   0.212   0.111   0.085   0.032   0.006   0.004
## Cumulative % of var.   99.549  99.761  99.872  99.957  99.989  99.995  99.999
##                        Dim.50  Dim.51
## Variance                0.001   0.000
## % of var.               0.001   0.000
## Cumulative % of var. 100.000 100.000
##
## Individuals (the 10 first)
##            Dist    Dim.1    ctr   cos2    Dim.2    ctr   cos2    Dim.3    ctr
## 1      |  6.125 | -0.553  0.001  0.008 |  2.202  0.031  0.129 | -1.167  0.011
## 2      |  7.543 | -2.599  0.028  0.119 |  1.533  0.015  0.041 |  1.857  0.027
## 3      |  5.597 |  1.030  0.004  0.034 | -0.123  0.000  0.000 |  2.382  0.044
## 4      |  7.089 | -1.953  0.016  0.076 | -3.212  0.066  0.205 |  1.802  0.025
## 5      |  6.118 | -2.577  0.028  0.177 | -0.469  0.001  0.006 |  0.467  0.002
## 6      |  8.606 | -2.756  0.032  0.103 | -1.437  0.013  0.028 |  1.318  0.013
## 7      |  8.961 | -2.043  0.017  0.052 | -3.701  0.088  0.171 |  4.353  0.147
## 8      |  5.947 | -1.197  0.006  0.041 |  2.218  0.032  0.139 |  0.322  0.001
## 9      |  4.461 | -0.978  0.004  0.048 | -1.994  0.025  0.200 | -1.439  0.016
## 10     |  4.664 |  1.766  0.013  0.143 | -0.824  0.004  0.031 | -0.555  0.002
##          cos2
## 1       0.036 |
## 2       0.061 |
## 3       0.181 |
## 4       0.065 |
## 5       0.006 |
## 6       0.023 |
## 7       0.236 |
## 8       0.003 |
## 9       0.104 |
## 10      0.014 |
##
## Variables (the 10 first)
##           Dim.1    ctr   cos2    Dim.2    ctr   cos2    Dim.3    ctr   cos2
## SEX    | -0.010  0.002  0.000 | -0.112  0.321  0.013 | -0.193  1.151  0.037 |
## AGE    | -0.339  1.905  0.115 |  0.140  0.501  0.020 |  0.026  0.021  0.001 |
## MCV    | -0.097  0.155  0.009 |  0.075  0.142  0.006 |  0.387  4.613  0.150 |
## HGB    |  0.794 10.432  0.630 |  0.369  3.471  0.136 |  0.221  1.503  0.049 |
## HCT    |  0.771  9.852  0.595 |  0.389  3.858  0.151 |  0.201  1.243  0.040 |
## PLT    |  0.013  0.003  0.000 |  0.107  0.289  0.011 | -0.385  4.557  0.148 |
## MCH    |  0.071  0.082  0.005 |  0.068  0.117  0.005 |  0.403  4.998  0.162 |
## MCHC   |  0.310  1.596  0.096 |  0.015  0.006  0.000 |  0.166  0.848  0.028 |
## RDW    | -0.564  5.269  0.318 | -0.053  0.072  0.003 | -0.041  0.051  0.002 |
## MPV    | -0.108  0.194  0.012 |  0.238  1.443  0.057 |  0.435  5.813  0.189 |
```

```r
cat("\nPCA Summary - Variance Explained:\n")
```

```
##
## PCA Summary - Variance Explained:
```

```r
print(summary_pca)
```

```
## NULL
```

```r
pca_result
```

```
## **Results for the Principal Component Analysis (PCA)**
## The analysis was performed on 3979 individuals, described by 51 variables
## *The results are available in the following objects:
##
##    name               description
## 1  "$eig"             "eigenvalues"
## 2  "$var"             "results for the variables"
## 3  "$var$coord"       "coord. for the variables"
## 4  "$var$cor"         "correlations variables - dimensions"
## 5  "$var$cos2"        "cos2 for the variables"
## 6  "$var$contrib"     "contributions of the variables"
## 7  "$ind"             "results for the individuals"
## 8  "$ind$coord"       "coord. for the individuals"
## 9  "$ind$cos2"        "cos2 for the individuals"
## 10 "$ind$contrib"     "contributions of the individuals"
## 11 "$call"            "summary statistics"
## 12 "$call$centre"     "mean of the variables"
## 13 "$call$ecart.type" "standard error of the variables"
## 14 "$call$row.w"      "weights for the individuals"
## 15 "$call$col.w"      "weights for the variables"
```

```r
# Method 1: Extract from eigenvalues matrix
eigenvalues <- pca_result$eig
print(eigenvalues)
```

```
##           eigenvalue percentage of variance cumulative percentage of variance
## comp 1  6.040383e+00           1.184687e+01                           11.84687
## comp 2  3.925721e+00           7.699428e+00                           19.54629
## comp 3  3.250013e+00           6.374177e+00                           25.92047
## comp 4  2.888735e+00           5.665610e+00                           31.58608
## comp 5  2.607096e+00           5.113238e+00                           36.69932
## comp 6  2.385586e+00           4.678795e+00                           41.37811
## comp 7  2.367568e+00           4.643457e+00                           46.02157
## comp 8  1.990769e+00           3.904449e+00                           49.92602
## comp 9  1.874492e+00           3.676399e+00                           53.60242
## comp 10 1.739953e+00           3.412529e+00                           57.01495
## comp 11 1.559709e+00           3.059022e+00                           60.07397
## comp 12 1.481528e+00           2.905687e+00                           62.97966
## comp 13 1.352656e+00           2.652933e+00                           65.63259
## comp 14 1.232028e+00           2.416348e+00                           68.04894
```

```
## comp 15 1.175106e+00          2.304710e+00                 70.35365
## comp 16 1.101771e+00          2.160878e+00                 72.51453
## comp 17 1.080855e+00          2.119856e+00                 74.63438
## comp 18 9.258556e-01          1.815860e+00                 76.45024
## comp 19 9.083682e-01          1.781562e+00                 78.23180
## comp 20 8.671261e-01          1.700675e+00                 79.93248
## comp 21 8.329680e-01          1.633681e+00                 81.56616
## comp 22 7.768736e-01          1.523664e+00                 83.08982
## comp 23 7.336678e-01          1.438926e+00                 84.52875
## comp 24 6.881535e-01          1.349660e+00                 85.87841
## comp 25 6.359432e-01          1.247261e+00                 87.12567
## comp 26 5.949084e-01          1.166780e+00                 88.29245
## comp 27 5.692505e-01          1.116458e+00                 89.40891
## comp 28 5.281920e-01          1.035931e+00                 90.44484
## comp 29 4.819969e-01          9.453296e-01                 91.39017
## comp 30 4.764369e-01          9.344248e-01                 92.32459
## comp 31 4.613856e-01          9.049051e-01                 93.22950
## comp 32 4.222717e-01          8.281918e-01                 94.05769
## comp 33 3.982133e-01          7.810068e-01                 94.83870
## comp 34 3.703450e-01          7.263493e-01                 95.56505
## comp 35 3.593625e-01          7.048095e-01                 96.26986
## comp 36 3.467491e-01          6.800712e-01                 96.94993
## comp 37 2.731152e-01          5.356546e-01                 97.48558
## comp 38 2.273288e-01          4.458547e-01                 97.93144
## comp 39 2.074105e-01          4.067895e-01                 98.33823
## comp 40 1.938379e-01          3.801699e-01                 98.71840
## comp 41 1.757996e-01          3.447918e-01                 99.06319
## comp 42 1.324637e-01          2.597979e-01                 99.32299
## comp 43 1.152168e-01          2.259721e-01                 99.54896
## comp 44 1.082963e-01          2.123991e-01                 99.76136
## comp 45 5.638004e-02          1.105769e-01                 99.87193
## comp 46 4.334335e-02          8.500833e-02                 99.95694
## comp 47 1.634231e-02          3.205181e-02                 99.98899
## comp 48 3.010141e-03          5.903721e-03                 99.99490
## comp 49 1.942786e-03          3.810341e-03                 99.99871
## comp 50 6.587547e-04          1.292001e-03                100.00000
## comp 51 5.311509e-30          1.041734e-29                100.00000
```

```r
# The eigenvalues matrix contains:
# - Column 1: Eigenvalue (variance) for each component
# - Column 2: Percentage of variance for each component
# - Column 3: Cumulative percentage of variance

# Method 2: Get cumulative variance directly
cumulative_variance <- eigenvalues[, 3]  # Third column
cat("\nCumulative Variance Explained:\n")
```

```
##
## Cumulative Variance Explained:
```

```r
print(cumulative_variance)
```

```
##    comp 1    comp 2    comp 3    comp 4    comp 5    comp 6    comp 7    comp 8
```

```
##  11.84687  19.54629  25.92047  31.58608  36.69932  41.37811  46.02157  49.92602
##    comp 9   comp 10   comp 11   comp 12   comp 13   comp 14   comp 15   comp 16
##  53.60242  57.01495  60.07397  62.97966  65.63259  68.04894  70.35365  72.51453
##   comp 17   comp 18   comp 19   comp 20   comp 21   comp 22   comp 23   comp 24
##  74.63438  76.45024  78.23180  79.93248  81.56616  83.08982  84.52875  85.87841
##   comp 25   comp 26   comp 27   comp 28   comp 29   comp 30   comp 31   comp 32
##  87.12567  88.29245  89.40891  90.44484  91.39017  92.32459  93.22950  94.05769
##   comp 33   comp 34   comp 35   comp 36   comp 37   comp 38   comp 39   comp 40
##  94.83870  95.56505  96.26986  96.94993  97.48558  97.93144  98.33823  98.71840
##   comp 41   comp 42   comp 43   comp 44   comp 45   comp 46   comp 47   comp 48
##  99.06319  99.32299  99.54896  99.76136  99.87193  99.95694  99.98899  99.99490
##   comp 49   comp 50   comp 51
##  99.99871 100.00000 100.00000
```

```r
# Method 3: Create a clean data frame
variance_df <- data.frame(
  Component = paste0("PC", 1:nrow(eigenvalues)),
  Eigenvalue = eigenvalues[, 1],
  Variance_Percent = eigenvalues[, 2],
  Cumulative_Percent = eigenvalues[, 3]
)

print(variance_df)
```

```
##         Component   Eigenvalue Variance_Percent Cumulative_Percent
## comp 1        PC1 6.040383e+00     1.184687e+01           11.84687
## comp 2        PC2 3.925721e+00     7.699428e+00           19.54629
## comp 3        PC3 3.250013e+00     6.374177e+00           25.92047
## comp 4        PC4 2.888735e+00     5.665610e+00           31.58608
## comp 5        PC5 2.607096e+00     5.113238e+00           36.69932
## comp 6        PC6 2.385586e+00     4.678795e+00           41.37811
## comp 7        PC7 2.367568e+00     4.643457e+00           46.02157
## comp 8        PC8 1.990769e+00     3.904449e+00           49.92602
## comp 9        PC9 1.874492e+00     3.676399e+00           53.60242
## comp 10      PC10 1.739953e+00     3.412529e+00           57.01495
## comp 11      PC11 1.559709e+00     3.059022e+00           60.07397
## comp 12      PC12 1.481528e+00     2.905687e+00           62.97966
## comp 13      PC13 1.352656e+00     2.652933e+00           65.63259
## comp 14      PC14 1.232028e+00     2.416348e+00           68.04894
## comp 15      PC15 1.175106e+00     2.304710e+00           70.35365
## comp 16      PC16 1.101771e+00     2.160878e+00           72.51453
## comp 17      PC17 1.080855e+00     2.119856e+00           74.63438
## comp 18      PC18 9.258556e-01     1.815860e+00           76.45024
## comp 19      PC19 9.083682e-01     1.781562e+00           78.23180
## comp 20      PC20 8.671261e-01     1.700675e+00           79.93248
## comp 21      PC21 8.329680e-01     1.633681e+00           81.56616
## comp 22      PC22 7.768736e-01     1.523664e+00           83.08982
## comp 23      PC23 7.336678e-01     1.438926e+00           84.52875
## comp 24      PC24 6.881535e-01     1.349660e+00           85.87841
## comp 25      PC25 6.359432e-01     1.247261e+00           87.12567
## comp 26      PC26 5.949084e-01     1.166780e+00           88.29245
## comp 27      PC27 5.692505e-01     1.116458e+00           89.40891
## comp 28      PC28 5.281920e-01     1.035931e+00           90.44484
## comp 29      PC29 4.819969e-01     9.453296e-01           91.39017
```

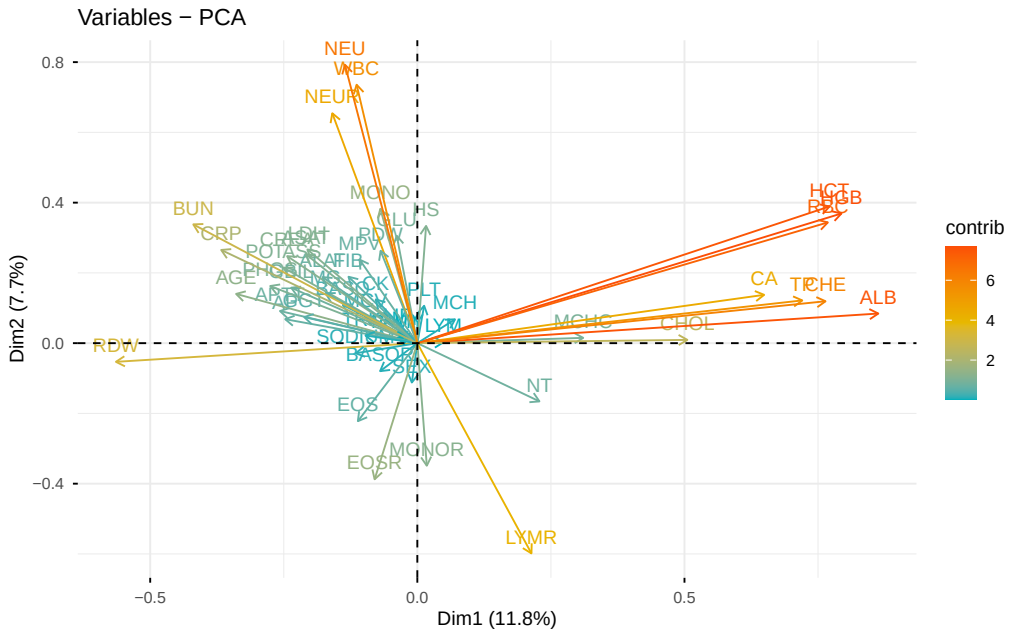
```
## comp 30      PC30 4.764369e-01     9.344248e-01           92.32459
## comp 31      PC31 4.613856e-01     9.049051e-01           93.22950
## comp 32      PC32 4.222717e-01     8.281918e-01           94.05769
## comp 33      PC33 3.982133e-01     7.810068e-01           94.83870
## comp 34      PC34 3.703450e-01     7.263493e-01           95.56505
## comp 35      PC35 3.593625e-01     7.048095e-01           96.26986
## comp 36      PC36 3.467491e-01     6.800712e-01           96.94993
## comp 37      PC37 2.731152e-01     5.356546e-01           97.48558
## comp 38      PC38 2.273288e-01     4.458547e-01           97.93144
## comp 39      PC39 2.074105e-01     4.067895e-01           98.33823
## comp 40      PC40 1.938379e-01     3.801699e-01           98.71840
## comp 41      PC41 1.757996e-01     3.447918e-01           99.06319
## comp 42      PC42 1.324637e-01     2.597979e-01           99.32299
## comp 43      PC43 1.152168e-01     2.259721e-01           99.54896
## comp 44      PC44 1.082963e-01     2.123991e-01           99.76136
## comp 45      PC45 5.638004e-02     1.105769e-01           99.87193
## comp 46      PC46 4.334335e-02     8.500833e-02           99.95694
## comp 47      PC47 1.634231e-02     3.205181e-02           99.98899
## comp 48      PC48 3.010141e-03     5.903721e-03           99.99490
## comp 49      PC49 1.942786e-03     3.810341e-03           99.99871
## comp 50      PC50 6.587547e-04     1.292001e-03          100.00000
## comp 51      PC51 5.311509e-30     1.041734e-29          100.00000
```

Necesitaríamos casi 28 PC para obtener un 90% de la varianza explicada. Solamente con una componente, tendríamos casi el 12%.

```
head(pca_result$var$coord,10)
```

```
##            Dim.1       Dim.2       Dim.3        Dim.4       Dim.5
## SEX  -0.01028197 -0.11217029 -0.19339908  0.043868313  0.05420174
## AGE  -0.33924426  0.14023302  0.02586324 -0.002654145 -0.24627379
## MCV  -0.09667631  0.07473725  0.38719967 -0.232935472  0.06896624
## HGB   0.79380222  0.36911875  0.22104301 -0.094716820  0.03073408
## HCT   0.77144105  0.38917486  0.20103206 -0.019741982 -0.01896616
## PLT   0.01273048  0.10654305 -0.38483975  0.596887484  0.14727672
## MCH   0.07050830  0.06786202  0.40305136 -0.386073064  0.17296927
## MCHC  0.31048705  0.01509701  0.16605009 -0.391301050  0.23513770
## RDW  -0.56414887 -0.05320022 -0.04079186  0.214326343 -0.12448143
## MPV  -0.10813143  0.23800245  0.43464033 -0.335714242 -0.05729973
```

```
fviz_pca_var(pca_result, col.var = "contrib", gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"))
```



Utilizaremos la función `prcomp` para obtener el resto de datos.

```
pca.acc <- prcomp(model_data_z_norm, scale. = FALSE)
summary(pca.acc)
```

```
## Importance of components:
##                            PC1     PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     2.4580 1.98159 1.80301 1.69984 1.61485 1.54473 1.53888
## Proportion of Variance 0.1185 0.07699 0.06374 0.05666 0.05113 0.04679 0.04643
## Cumulative Proportion  0.1185 0.19546 0.25920 0.31586 0.36699 0.41378 0.46022
##                            PC8     PC9     PC10    PC11    PC12    PC13    PC14
## Standard deviation     1.41112 1.36929 1.31924 1.24904 1.21733 1.16318 1.11011
## Proportion of Variance 0.03904 0.03676 0.03413 0.03059 0.02906 0.02653 0.02416
## Cumulative Proportion  0.49926 0.53602 0.57015 0.60074 0.62980 0.65633 0.68049
##                           PC15    PC16   PC17    PC18    PC19    PC20    PC21
## Standard deviation     1.08416 1.04978 1.0398 0.96233 0.95320 0.93131 0.91279
## Proportion of Variance 0.02305 0.02161 0.0212 0.01816 0.01782 0.01701 0.01634
## Cumulative Proportion  0.70354 0.72515 0.7463 0.76450 0.78232 0.79932 0.81566
##                           PC22    PC23   PC24    PC25    PC26    PC27    PC28
## Standard deviation     0.88152 0.85665 0.8297 0.79756 0.77140 0.75458 0.72686
## Proportion of Variance 0.01524 0.01439 0.0135 0.01247 0.01167 0.01116 0.01036
## Cumulative Proportion  0.83090 0.84529 0.8588 0.87126 0.88292 0.89409 0.90445
##                           PC29    PC30    PC31    PC32    PC33    PC34    PC35
## Standard deviation     0.69435 0.69033 0.67934 0.64991 0.63112 0.60864 0.59954
## Proportion of Variance 0.00945 0.00934 0.00905 0.00828 0.00781 0.00726 0.00705
## Cumulative Proportion  0.91390 0.92325 0.93229 0.94058 0.94839 0.95565 0.96270
##                           PC36    PC37    PC38    PC39   PC40    PC41   PC42
## Standard deviation     0.5889 0.52267 0.47685 0.45548 0.4403 0.41934 0.3640
## Proportion of Variance 0.0068 0.00536 0.00446 0.00407 0.0038 0.00345 0.0026
## Cumulative Proportion  0.9695 0.97486 0.97931 0.98338 0.9872 0.99063 0.9932
##                           PC43    PC44    PC45    PC46    PC47    PC48    PC49
## Standard deviation     0.33948 0.32913 0.23747 0.20822 0.12785 0.05487 0.04408
## Proportion of Variance 0.00226 0.00212 0.00111 0.00085 0.00032 0.00006 0.00004
```

```
## Cumulative Proportion  0.99549 0.99761 0.99872 0.99957 0.99989 0.99995 0.99999
##                            PC50      PC51
## Standard deviation     0.02567 3.373e-15
## Proportion of Variance 0.00001 0.000e+00
## Cumulative Proportion  1.00000 1.000e+00
```

Como se puede observar la función `summary`, nos devuelve la proporción de varianza aplicada al conjunto total de cada atributo. Gracias a esto, el atributo 1 explica el 12% de variabilidad del total de datos; en cambio, el atributo 8 explica solo casi el 4%. Si tenemos que ver el acumulado, para poder llegar al 90% de proporción de la varianza, necesitaríamos 28 componentes principales.

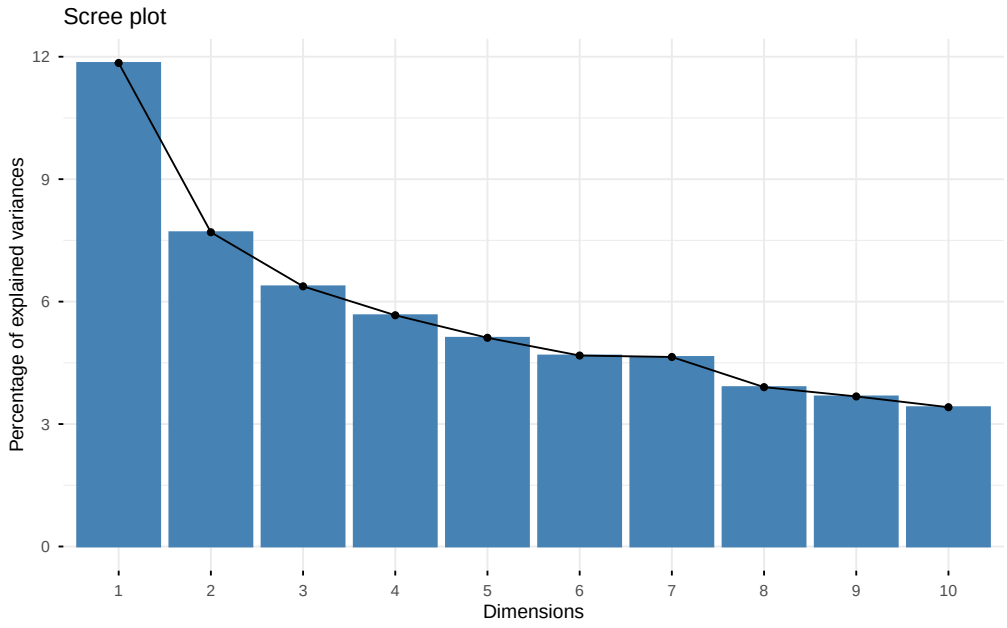
A continuación, se muestra un histograma para ver el peso de cada atributo sobre el conjunto total de datos:

```
#Los valores propios corresponden a la cantidad de variación explicada por cada componente principal (P
ev= get_eig(pca.acc)
ev
```

```
##          eigenvalue variance.percent cumulative.variance.percent
## Dim.1  6.041901e+00     1.184687e+01                     11.84687
## Dim.2  3.926708e+00     7.699428e+00                     19.54629
## Dim.3  3.250830e+00     6.374177e+00                     25.92047
## Dim.4  2.889461e+00     5.665610e+00                     31.58608
## Dim.5  2.607751e+00     5.113238e+00                     36.69932
## Dim.6  2.386185e+00     4.678795e+00                     41.37811
## Dim.7  2.368163e+00     4.643457e+00                     46.02157
## Dim.8  1.991269e+00     3.904449e+00                     49.92602
## Dim.9  1.874963e+00     3.676399e+00                     53.60242
## Dim.10 1.740390e+00     3.412529e+00                     57.01495
## Dim.11 1.560101e+00     3.059022e+00                     60.07397
## Dim.12 1.481901e+00     2.905687e+00                     62.97966
## Dim.13 1.352996e+00     2.652933e+00                     65.63259
## Dim.14 1.232337e+00     2.416348e+00                     68.04894
## Dim.15 1.175402e+00     2.304710e+00                     70.35365
## Dim.16 1.102048e+00     2.160878e+00                     72.51453
## Dim.17 1.081126e+00     2.119856e+00                     74.63438
## Dim.18 9.260884e-01     1.815860e+00                     76.45024
## Dim.19 9.085965e-01     1.781562e+00                     78.23180
## Dim.20 8.673441e-01     1.700675e+00                     79.93248
## Dim.21 8.331774e-01     1.633681e+00                     81.56616
## Dim.22 7.770689e-01     1.523664e+00                     83.08982
## Dim.23 7.338522e-01     1.438926e+00                     84.52875
## Dim.24 6.883265e-01     1.349660e+00                     85.87841
## Dim.25 6.361031e-01     1.247261e+00                     87.12567
## Dim.26 5.950579e-01     1.166780e+00                     88.29245
## Dim.27 5.693936e-01     1.116458e+00                     89.40891
## Dim.28 5.283247e-01     1.035931e+00                     90.44484
## Dim.29 4.821181e-01     9.453296e-01                     91.39017
## Dim.30 4.765567e-01     9.344248e-01                     92.32459
## Dim.31 4.615016e-01     9.049051e-01                     93.22950
## Dim.32 4.223778e-01     8.281918e-01                     94.05769
## Dim.33 3.983134e-01     7.810068e-01                     94.83870
## Dim.34 3.704381e-01     7.263493e-01                     95.56505
## Dim.35 3.594528e-01     7.048095e-01                     96.26986
## Dim.36 3.468363e-01     6.800712e-01                     96.94993
```

```
## Dim.37 2.731838e-01     5.356546e-01                97.48558
## Dim.38 2.273859e-01     4.458547e-01                97.93144
## Dim.39 2.074626e-01     4.067895e-01                98.33823
## Dim.40 1.938867e-01     3.801699e-01                98.71840
## Dim.41 1.758438e-01     3.447918e-01                99.06319
## Dim.42 1.324970e-01     2.597979e-01                99.32299
## Dim.43 1.152458e-01     2.259721e-01                99.54896
## Dim.44 1.083235e-01     2.123991e-01                99.76136
## Dim.45 5.639421e-02     1.105769e-01                99.87193
## Dim.46 4.335425e-02     8.500833e-02                99.95694
## Dim.47 1.634642e-02     3.205181e-02                99.98899
## Dim.48 3.010898e-03     5.903721e-03                99.99490
## Dim.49 1.943274e-03     3.810341e-03                99.99871
## Dim.50 6.589203e-04     1.292001e-03               100.00000
## Dim.51 1.137448e-29     2.230290e-29               100.00000
```

```
fviz_eig(pca.acc)
```



Los valores propios se pueden utilizar para determinar el número de componentes principales a retener después de la PCA (Kaiser 1961):

Un valor propio > 1 indica que los PCs representan más varianza de la que representa una de las variables originales de los datos estandarizados. Esto se utiliza habitualmente como punto de corte para el cual se conservan los PCs. Esto solo es cierto cuando los datos están estandarizados.

También podemos limitar el número de componentes a este número que representa una determinada fracción de la varianza total. Por ejemplo, si estamos satisfecho con el 80% de la varianza total explicada, usamos el número de componentes para conseguirlo que son los 4 componentes principales vistos antes.

Continuamos con el análisis de los componentes principales. Después de aplicar el método Káiser se han seleccionado los 4 componentes principales.

```
var <- get_pca_var(pca.acc)
var
```

```
## Principal Component Analysis Results for variables
##  ===================================================
##   Name       Description
## 1 "$coord"   "Coordinates for the variables"
## 2 "$cor"     "Correlations between variables and dimensions"
## 3 "$cos2"    "Cos2 for the variables"
## 4 "$contrib" "contributions of the variables"
```

Los componentes de `get_pca_var()` se pueden utilizar en el diagrama de variables de la siguiente manera:

- var$coord: coordenadas de variables para crear un diagrama de dispersión.
- var$cos2: representa la calidad de representación de las variables al mapa de factores. Se calcula como las coordenadas al cuadrado: var.cos2 = var.coord * var.coord.
- var$contrib: contiene las contribuciones (en porcentaje) de las variables a los componentes principales. La contribución de una variable (var) a un determinado componente principal es (en porcentaje): (var.cos2 * 100) / (cos2 total del componente).

```
#Utilizamos los 4 componentes principales encontrados antes
head(var$coord[,1:4],11)
```

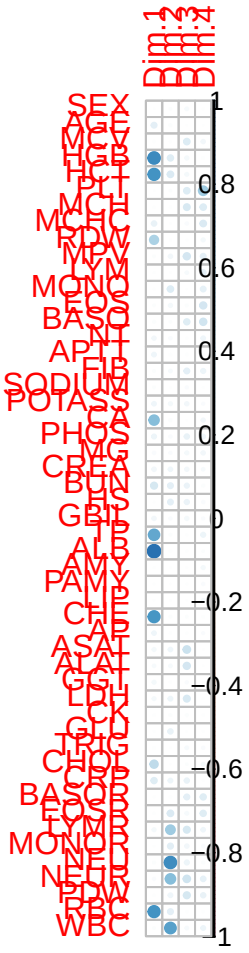
```
##            Dim.1        Dim.2       Dim.3        Dim.4
## SEX  -0.01028326 -0.112184391  0.19342338 -0.043873827
## AGE  -0.33928690  0.140250640 -0.02586649  0.002654478
## MCV  -0.09668846  0.074746643 -0.38724833  0.232964748
## HGB   0.79390199  0.369165143 -0.22107079  0.094728725
## HCT   0.77153800  0.389223771 -0.20105733  0.019744464
## PLT   0.01273208  0.106556445  0.38488812 -0.596962503
## MCH   0.07051716  0.067870546 -0.40310202  0.386121587
## MCHC  0.31052607  0.015098903 -0.16607096  0.391350230
## RDW  -0.56421977 -0.053206911  0.04079699 -0.214353280
## MPV  -0.10814502  0.238032364 -0.43469496  0.335756436
## LYM   0.04887307  0.002499193 -0.17867568 -0.341257788
```

La calidad de representación de las variables en el mapa de factores se denomina cos2 (coseno cuadrado, coordenadas cuadradas). Podemos acceder al cos2 de la siguiente manera:

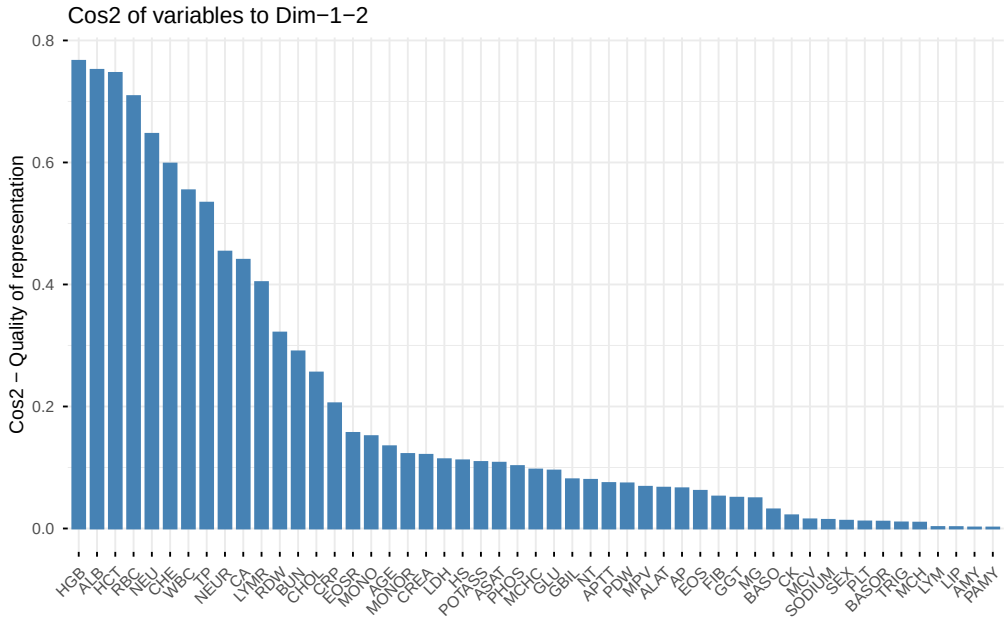
```
head(var$cos2[,1:4],11)
```

```
##             Dim.1        Dim.2        Dim.3        Dim.4
## SEX  0.0001057454 1.258534e-02 0.0374126053 1.924913e-03
## AGE  0.1151155972 1.967024e-02 0.0006690755 7.046254e-06
## MCV  0.0093486575 5.587061e-03 0.1499612705 5.427257e-02
## HGB  0.6302803714 1.362829e-01 0.0488722935 8.973531e-03
## HCT  0.5952708916 1.514951e-01 0.0404240482 3.898438e-04
## PLT  0.0001621059 1.135428e-02 0.1481388669 3.563642e-01
## MCH  0.0049726701 4.606411e-03 0.1624912398 1.490899e-01
## MCHC 0.0964264427 2.279769e-04 0.0275795628 1.531550e-01
## RDW  0.3183439504 2.830975e-03 0.0016643942 4.594733e-02
## MPV  0.0116953448 5.665941e-02 0.1889597057 1.127324e-01
## LYM  0.0023885773 6.245967e-06 0.0319250003 1.164569e-01
```

```
corrplot(var$cos2[,1:4], is.corre=FALSE)
```




```
fviz_cos2(pca.acc, choice = "var", axes = 1:2)
```



- Un cos2 elevado indica una buena representación de la variable en el componente principal. En este caso, la variable se coloca cerca de la circunferencia del círculo de correlación.
- Un cos2 bajo indica que la variable no está perfectamente representada por los PC. En este caso, la variable está cerca del centro del círculo.

Para una variable dada, la suma del cos2 de todos los componentes principales es igual a uno.

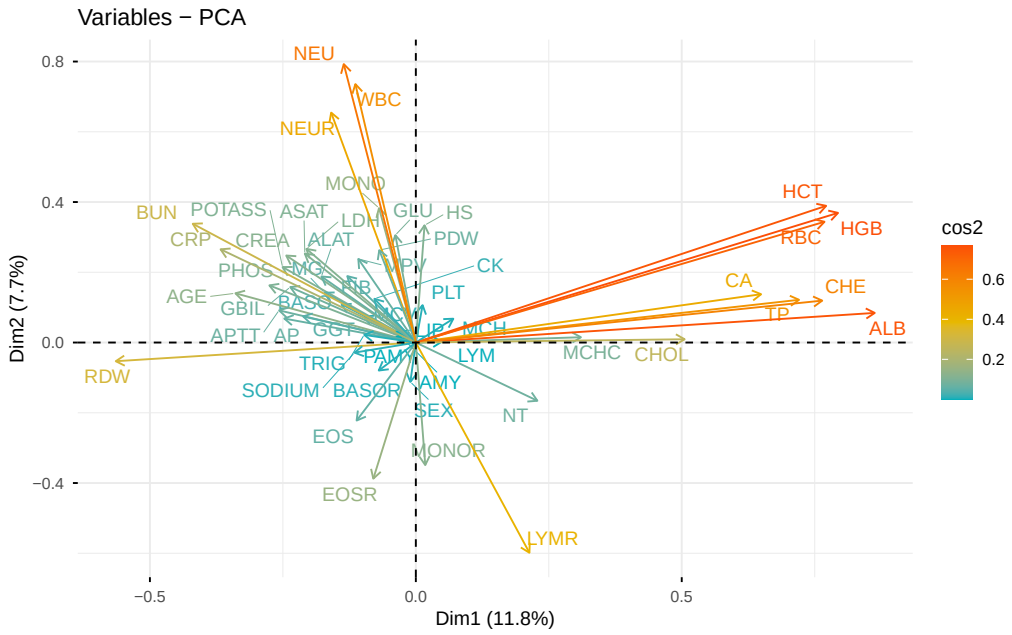
Si una variable está perfectamente representada por solo dos componentes principales (Dim.1 y Dim.2), la suma del cos2 en estos dos PCs es igual a uno. En este caso las variables se colocarán en el círculo de correlaciones.

Para algunas de las variables, pueden ser necesarios más de 2 componentes para representar perfectamente los datos. En este caso las variables se sitúan dentro del círculo de correlaciones.

En resumen:

Los valores de cos2 se utilizan para estimar la calidad de la representación Cuanto más próxima esté una variable al círculo de correlaciones, mejor será su representación en el mapa de factores (y más importante es interpretar estos componentes) Las variables que están próximas en el centro de la trama son menos importantes para los primeros componentes.

```
fviz_pca_var(pca.acc,
col.var = "cos2",
gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
repel = TRUE
)
```



3.0.2.4 Contribución de las características Las contribuciones de las variables en la contabilización de la variabilidad de un determinado componente principal se expresan en porcentaje.

Las variables que están correlacionadas con PC1 (es decir, Dim.1) y PC2 (es decir, Dim.2) son las más importantes para explicar la variabilidad en el conjunto de datos.

Las variables que no están correlacionadas con ningún PC o con las últimas dimensiones son variables con una contribución baja y se pueden eliminar para simplificar el análisis global.

La contribución de las variables se puede extraer de la siguiente manera:

```
head(var$contrib[,1:4],11)
```

```
##             Dim.1        Dim.2      Dim.3        Dim.4
## SEX   0.001750201 0.3205060634 1.15086306 6.661839e-02
## AGE   1.905287587 0.5009346651 0.02058168 2.438605e-04
## MCV   0.154730388 0.1422835738 4.61301438 1.878294e+00
## HGB  10.431821551 3.4706655021 1.50337878 3.105607e-01
## HCT   9.852376813 3.8580699165 1.24349917 1.349192e-02
## PLT   0.002683028 0.2891550838 4.55695474 1.233324e+01
## MCH   0.082303066 0.1173097377 4.99845342 5.159781e+00
## MCHC  1.595961875 0.0058058014 0.84838518 5.300469e+00
## RDW   5.268936545 0.0720953871 0.05119905 1.590169e+00
## MPV   0.193570601 1.4429238203 5.81265973 3.901502e+00
## LYM   0.039533536 0.0001590637 0.98205680 4.030401e+00
```

Cuando más grande sea el valor de la contribución, más contribución habrá al componente.

```
sort(abs(pca.acc$rotation[, 1]), decreasing = TRUE)
```

```
##         ALB         HGB         HCT         RBC         CHE          TP
## 0.351076335 0.322983305 0.313884960 0.312642220 0.310878977 0.293168584
##          CA         RDW        CHOL         BUN         CRP         AGE
## 0.264139711 0.229541642 0.205659654 0.170641508 0.149374720 0.138032155
##        MCHC        PHOS        APTT      POTASS          AP        CREA
## 0.126331385 0.112061435 0.104630123 0.101959581 0.100666390 0.099144812
##        GBIL          NT        LYMR         GGT        ASAT         LDH
## 0.096061361 0.092997149 0.086905352 0.086225562 0.084953252 0.083745864
##        ALAT          MG        NEUR        BASO         NEU         FIB
## 0.072234520 0.069827339 0.064772103 0.056773205 0.055126825 0.052437979
##      SODIUM         WBC         EOS         MPV        TRIG         MCV
## 0.046918943 0.046240787 0.045299875 0.043996659 0.039526874 0.039335784
##        EOSR          CK         MCH       BASOR        MONO         PDW
## 0.032527318 0.031696496 0.028688511 0.028365791 0.028173185 0.027975427
##         LYM         GLU        PAMY         LIP         AMY       MONOR
## 0.019883042 0.015673792 0.014995016 0.014788375 0.013312378 0.007221296
##          HS         PLT         SEX
## 0.006659716 0.005179795 0.004183541
```

Seleccionaremos las 10 más importantes de la PC1

```
top_pc1 <- sort(abs(pca.acc$rotation[, 1]), decreasing = TRUE)
top_pc1 <- names(top_pc1[1:10])
top_pc1
```

```
##  [1] "ALB"  "HGB"  "HCT"  "RBC"  "CHE"  "TP"   "CA"   "RDW"  "CHOL" "BUN"
```

Lo mismo para la PC2, en este caso serían 4

```
sort(abs(pca.acc$rotation[, 2]), decreasing = TRUE)
```

```
##         NEU         WBC        NEUR        LYMR         HCT        EOSR
## 0.400081003 0.371332511 0.330342859 0.302052466 0.196419702 0.195617881
##        MONO         HGB       MONOR         RBC         BUN          HS
## 0.193200050 0.186297222 0.176118303 0.173507133 0.170648628 0.168463981
##         GLU         LDH         CRP         PDW        ASAT        CREA
## 0.154313847 0.134622018 0.133930952 0.132756598 0.127885361 0.124995531
##         MPV         EOS      POTASS         FIB        ALAT          NT
## 0.120121764 0.112089253 0.108445539 0.095417986 0.094921505 0.083689926
##        PHOS        GBIL          MG         AGE          CA          CK
## 0.082090363 0.079657425 0.071597302 0.070776738 0.069409165 0.062903638
##          TP         CHE         SEX        BASO         PLT        APTT
## 0.061383422 0.060215442 0.056613255 0.054871132 0.053773142 0.046203068
##         ALB       BASOR         MCV         GGT          AP         MCH
## 0.042546250 0.040231651 0.037720495 0.037508181 0.034459248 0.034250509
##         RDW         LIP      SODIUM         AMY        TRIG        PAMY
## 0.026850584 0.015642606 0.014490819 0.012058660 0.011020031 0.007736739
##        MCHC        CHOL         LYM
## 0.007619581 0.004703282 0.001261205
```

```
top_pc2 <- sort(abs(pca.acc$rotation[, 2]), decreasing = TRUE)
top_pc2 <- names(top_pc2[1:4])
top_pc2
```

```
## [1] "NEU"  "WBC"  "NEUR" "LYMR"
```

3.0.3 Aplicamos una imputación a través de MICE

```
feat_variables <- c("ALB","HGB","HCT","RBC","CHE","TP", "CA",
                    "RDW","CHOL","BUN","NEU", "WBC","NEUR", "LYMR")

bacteremia_imputed <- bacteremia_cleaned %>%
  select(feat_variables, BloodCulture)

# Multiple imputation using MICE
cat("\nPerforming Multiple Imputation...\n")
```

```
##
## Performing Multiple Imputation...
```

```
imputed_data <- mice(bacteremia_imputed,
                     m = 5,
                     maxit = 50,
                     method = 'pmm',
                     seed = 42,
                     printFlag = FALSE)

# # Extract first imputed dataset
bacteremia_mice_imputed <- complete(imputed_data, 1)
```

```r
# Check imputation quality
cat("\nMissing values after imputation:", sum(is.na(bacteremia_mice_imputed)), "\n")
```

```
##
## Missing values after imputation: 0
```

3.0.3.1 Modelo final con Regresión Logística Entrenamos el modelo de regresión logística con las variables obtenidas en las secciones 4.2.2 y 4.2.3.

```r
feat_variables <- c("ALB","HGB","HCT","RBC","CHE","TP", "CA","RDW","CHOL","BUN" ,
                    "NEU", "WBC","NEUR", "LYMR")

# Selección y Escalado
model_data_glm <- bacteremia_mice_imputed %>%
  select(all_of(feat_variables), BloodCulture) %>%
  mutate(across(all_of(feat_variables), scale)) %>%
    mutate(BloodCulture = factor(BloodCulture, levels = c("no", "yes")))


# Dividimos 70 train y 30 test  para validar
set.seed(123)
split <- createDataPartition(
  y = model_data_glm$BloodCulture,
  p = 0.7,
  list = FALSE
)

train_set <- model_data_glm[split, ]
test_set  <- model_data_glm[-split, ]

p_yes <- mean(train_set$BloodCulture == "yes")
p_no  <- mean(train_set$BloodCulture == "no")

weights <- ifelse(
  train_set$BloodCulture == "yes",
  1 / p_yes,
  1 / p_no
)


# Modelo logístico
modelo_ajustado <- glm(
  BloodCulture ~ .,
  data = train_set,
  family = binomial(link = "logit"),
  weights = weights
)

# Resumen modelo
summary(modelo_ajustado)
```

```
##
```

```
## Call:
## glm(formula = BloodCulture ~ ., family = binomial(link = "logit"),
##     data = train_set, weights = weights)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.323449   0.031759 -10.185  < 2e-16 ***
## ALB         -0.284428   0.063540  -4.476 7.59e-06 ***
## HGB          0.342626   0.159029   2.154   0.0312 *
## HCT          0.005494   0.173873   0.032   0.9748
## RBC         -0.408859   0.076792  -5.324 1.01e-07 ***
## CHE         -0.216764   0.046036  -4.709 2.49e-06 ***
## TP           0.237545   0.049274   4.821 1.43e-06 ***
## CA           0.290748   0.042323   6.870 6.43e-12 ***
## RDW          0.024495   0.038131   0.642   0.5206
## CHOL        -0.025960   0.035951  -0.722   0.4702
## BUN          0.276365   0.029730   9.296  < 2e-16 ***
## NEU          1.069166   0.234173   4.566 4.98e-06 ***
## WBC         -1.159533   0.231698  -5.005 5.60e-07 ***
## NEUR         1.396278   0.109766  12.721  < 2e-16 ***
## LYMR         0.894518   0.087074  10.273  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 7724.4  on 2785  degrees of freedom
## Residual deviance: 6833.0  on 2771  degrees of freedom
## AIC: 6479.6
##
## Number of Fisher Scoring iterations: 5
```

```r
# Matriz de Confusión
probabilidades <- predict(modelo_ajustado, newdata = test_set, type = "response")

threshold <- 0.35   # probando 0.25-0.45
predicciones <- factor(
  ifelse(probabilidades > threshold, "yes", "no"),
  levels = c("yes", "no")
)

confusionMatrix(
  predicciones,
  test_set$BloodCulture,
  positive = "yes"
)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  421  20
##        yes 676  76
##
```

```
##                Accuracy : 0.4166
##                  95% CI : (0.3884, 0.4452)
##     No Information Rate : 0.9195
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.0426
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.79167
##             Specificity : 0.38377
##          Pos Pred Value : 0.10106
##          Neg Pred Value : 0.95465
##              Prevalence : 0.08047
##          Detection Rate : 0.06370
##    Detection Prevalence : 0.63034
##       Balanced Accuracy : 0.58772
##
##        'Positive' Class : yes
##
```

```r
library(pROC)
roc_obj <- roc(test_set$BloodCulture, probabilidades, levels = c("no", "yes"))
auc(roc_obj)
```

```
## Area under the curve: 0.6724
```

```r
# Odds Ratios e intervalos de confianza al 95%
or_coef <- exp(coef(modelo_ajustado))
or_ci <- exp(confint(modelo_ajustado))
or_table <- cbind(OR = or_coef, or_ci)
print(or_table)
```

```
##                     OR     2.5 %    97.5 %
## (Intercept) 0.7236489 0.6798035 0.7699425
## ALB         0.7524446 0.6640413 0.8519071
## HGB         1.4086419 1.0314636 1.9241851
## HCT         1.0055094 0.7150940 1.4139550
## RBC         0.6644076 0.5713488 0.7720919
## CHE         0.8051199 0.7355603 0.8810727
## TP          1.2681319 1.1518345 1.3973015
## CA          1.3374274 1.2319572 1.4545285
## RDW         1.0247974 0.9509876 1.1043690
## CHOL        0.9743739 0.9080853 1.0456150
## BUN         1.3183286 1.2442201 1.3980966
## NEU         2.9129504 1.8513353 4.6341739
## WBC         0.3136325 0.1980433 0.4908419
## NEUR        4.0401351 3.2617894 5.0152745
## LYMR        2.4461565 2.0625500 2.9017166
```

```r
# Creamos la matriz de confusión
matriz_conf <- table(Predicho = predicciones,
```

```
                     Real = test_set$BloodCulture)
print(matriz_conf)
```

```
##         Real
## Predicho  no yes
##      yes 676  76
##      no  421  20
```

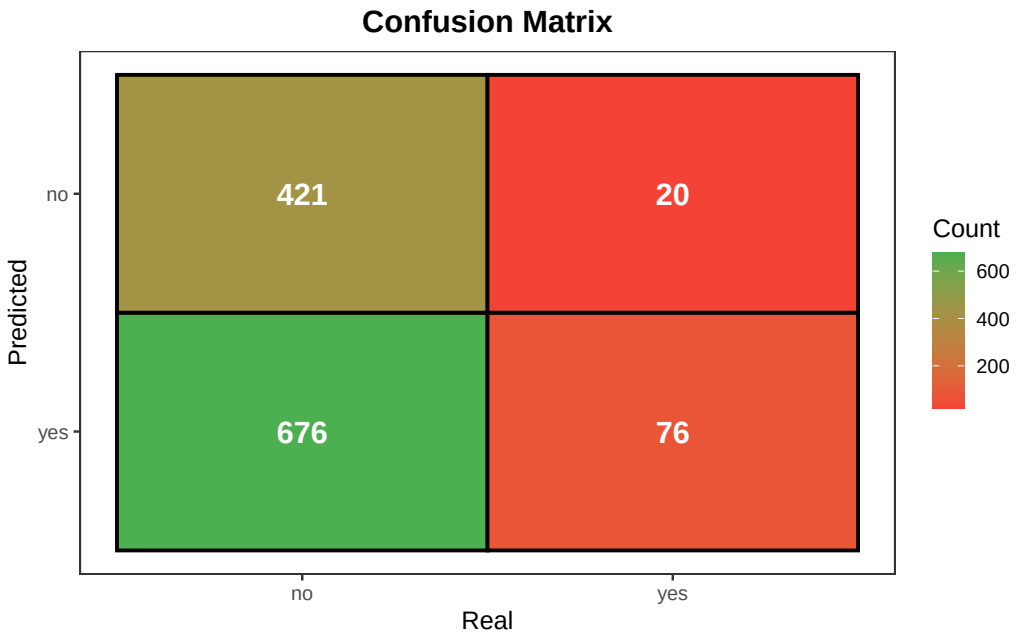
```
# Se calculan las métricas
VP <- matriz_conf[2, 2] # Verdaderos Positivos
VN <- matriz_conf[1, 1] # Verdaderos Negativos
FP <- matriz_conf[2, 1] # Falsos Positivos
FN <- matriz_conf[1, 2] # Falsos Negativos

sensibilidad <- VP / (VP + FN)
especificidad <- VN / (VN + FP)
precision <- VP / (VP + FP)
accuracy <- (VP + VN) / sum(matriz_conf)
f1_score <- 2 * (precision * sensibilidad) / (precision + sensibilidad)


# Visualizaremos las métricas
cat(sprintf("Sensibilidad (Recall): %.3f\n", sensibilidad))
```

```
## Sensibilidad (Recall): 0.208
```

```
matriz_conf[1, 1]
```

```
## [1] 676
```

```
matriz_df <- as.data.frame(matriz_conf)
colnames(matriz_df) <- c("Predicho", "Real", "Freq")

ggplot(matriz_df, aes(x = Real, y = Predicho, fill = Freq)) +
  geom_tile(color = "black", linewidth = 1) +
  geom_text(aes(label = Freq), size = 6, color = "white", fontface = "bold") +
  scale_fill_gradient(
    low = "#F44336",
    high = "#4CAF50"
  ) +
  labs(
    title = "Confusion Matrix",
    x = "Real",
    y = "Predicted",
    fill = "Count"
  ) +
  theme_bw(base_size = 14) +
  theme(
    plot.title = element_text(hjust = 0.5, face = "bold"),
    panel.grid = element_blank()
  )
```


Confusion Matrix
421
20
676
76
no
yes
Predicted
no
yes
Real
Count
600
400
200

4 Visualización (1,5 puntos)

Aplicación Shiny con alguna solución aportada / análisis

La aplicación Shiny desarrollada incluye todo los siguientes elementos:

- **Carga de archivos flexible**: CSV, TSV, TXT con parámetros configurables
- **Vista previa de datos**: Verificación inmediata del formato
- **Filtrado dinámico**: Eliminación de NAs y selección de genes variables
- **Análisis exploratorio**: Gráficos de varianza génica
- **Heatmap de correlación**: Visualización clara de co-expresión
- **Interfaz intuitiva**: Diseño con pestañas y controles organizados
- **Descarga de resultados**: Export del heatmap en alta resolución

```
library(shiny)
library(shinydashboard)
library(ggplot2)
library(dplyr)
library(tidyr)
library(DT)
library(plotly)
library(corrplot)
library(randomForest)
library(caret)
library(xgboost)

# UI Definition
ui <- dashboardPage(
  dashboardHeader(title = "Análisis de Bacteremia"),
  dashboardSidebar(
    sidebarMenu(
      menuItem("Autores y Objetivo", tabName = "autores", icon = icon("users")),
      menuItem("Resumen", tabName = "resumen", icon = icon("home")),
      menuItem("Calidad de Datos", tabName = "calidad", icon = icon("database")),
      menuItem("Análisis de Variables", tabName = "analisis", icon = icon("chart-bar")),
      menuItem("Modelos", tabName = "modelos", icon = icon("brain"))
    )
  ),
  dashboardBody(
    tags$head(
      tags$style(HTML("
        .small-box {height: 100px;}
        .info-box {min-height: 100px;}
        .content-wrapper {background-color: #f4f6f9;}
        .author-box {background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
                     color: white; padding: 20px; border-radius: 10px;}
        .aim-box {background: linear-gradient(135deg, #905896 0%, #f5576c 100%);
                  color: white; padding: 20px; border-radius: 10px;}
        .highlight {background-color: #b2aa8f; padding: 10px; border-left: 4px solid #ffc107;}
      "))
    ),
    tabItems(
      # Autores y Objetivo Tab
      tabItem(tabName = "autores",
```

```
            fluidRow(
              column(6,
                box(title = "Autores del Proyecto", status = "primary", solidHeader = TRUE,
                    width = NULL,
                    div(class = "author-box",
                        h3("Marco Russo"),
                        tags$ul(
                          tags$li(icon("envelope"), " Email: mrusorb@uoc.edu"),
                          tags$li(icon("github"), " GitHub: https://github.com/marcusRB/uoc-ub-scienti
                          tags$li(icon("linkedin"), " LinkedIn: https://www.linkedin.com/in/marcusrb/"
                          tags$li(icon("calendar"), " Fecha: January 13, 2026")
                        ),
                        br(),
                        h3("Silvia Gamundi Sumando"),
                        tags$ul(
                          tags$li(icon("envelope"), " Email: sgamundis@uoc.edu"),
                          tags$li(icon("github"), " GitHub: https://github.com/"),
                          tags$li(icon("linkedin"), " LinkedIn: https://www.linkedin.com/in/"),
                          tags$li(icon("calendar"), " Fecha: January 13, 2026")
                        )
                    )
                )
              ),
              column(6,
                box(title = "Objetivo del Estudio", status = "warning", solidHeader = TRUE,
                    width = NULL,
                    div(class = "aim-box",
                        h3("Contexto y Objetivo del Estudio"),
                        p("El dataset elegido es 'Bacteremia' del autor Heinze, G. (2023), disponible
                        tags$blockquote(
                          "Heinze, G. (2023). Bacteremia [Data set]. In PLoS One (Version S2, Vol. 9,
                          tags$a(href = "https://doi.org/10.5281/zenodo.7554815",
                                 "https://doi.org/10.5281/zenodo.7554815",
                                 target = "_blank")
                        ),
                        br(),
                        h4("Características del Dataset:"),
                        tags$ul(
                          tags$li("14,691 observaciones de pacientes con sospecha clínica de bacteremi
                          tags$li("Recogidas en el Vienna General Hospital, Austria (2006-2010)"),
                          tags$li("51 predictores potenciales de bacteremia"),
                          tags$li("Variable objetivo: Resultado del hemocultivo (no/yes)")
                        ),
                        br(),
                        h4("Objetivos Principales:"),
                        tags$ol(
                          tags$li(strong("Análisis Exploratorio:"), " Comprender la estructura y calid
                          tags$li(strong("Preprocesamiento:"), " Manejo de valores faltantes y selecci
                          tags$li(strong("Modelización:"), " Desarrollar modelos predictivos para dete
                          tags$li(strong("Validación:"), " Evaluar el rendimiento de los modelos en es
                        ),
                        br(),
                        div(class = "highlight",
```

```
                        h5("Contexto Clínico:"),
                        p("La bacteremia es la presencia de bacterias en la sangre. Aunque muchas
                    )
                )
            )
          )
        ),
        fluidRow(
          box(title = "Información Técnica del Proyecto", status = "success", solidHeader = TRUE,
              width = 12,
              fluidRow(
                valueBox("R", "Lenguaje Principal", icon = icon("r-project"), color = "blue", widt
                valueBox("Shiny", "Dashboard Interactivo", icon = icon("dashboard"), color = "oliv
                valueBox("Git", "Control de Versiones", icon = icon("code-branch"), color = "yello
                valueBox("Markdown", "Documentación", icon = icon("file-alt"), color = "red", widt
              ),
              br(),
              p("Este proyecto forma parte de la evaluación final del curso de Programación Cienti
              p("Todos los análisis, visualizaciones y modelos se han desarrollado utilizando meto
          )
        )
    ),

    # Resumen Tab
    tabItem(tabName = "resumen",
        fluidRow(
          valueBox("14,691", "Observaciones", icon = icon("database"), color = "blue", width = 3),
          valueBox("53", "Variables", icon = icon("table"), color = "green", width = 3),
          valueBox("8.03%", "Tasa de Bacteremia", icon = icon("percentage"), color = "red", width
          valueBox("27%", "Casos Completos", icon = icon("check-circle"), color = "yellow", width
        ),
        fluidRow(
          box(title = "Distribución de Edad por Grupo", status = "primary", solidHeader = TRUE,
              plotlyOutput("edad_plot"), width = 8),
          box(title = "Estadísticas por Grupo", status = "info", solidHeader = TRUE,
              tableOutput("edad_stats"), width = 4)
        ),
        fluidRow(
          box(title = "Proporción de Bacteremia", status = "success", solidHeader = TRUE,
              plotOutput("proporcion_plot"), width = 6),
          box(title = "Distribución por Sexo", status = "warning", solidHeader = TRUE,
              plotOutput("sexo_plot"), width = 6)
        )
    ),

    # Calidad de Datos Tab
    tabItem(tabName = "calidad",
        fluidRow(
          box(title = "Valores Faltantes por Variable", status = "primary", solidHeader = TRUE,
              plotlyOutput("missing_plot"), width = 8, height = 600),
          box(title = "Categorías de Correlación", status = "info", solidHeader = TRUE,
              DTOutput("correlation_table"), width = 4)
        ),
```

```
            fluidRow(
              infoBox("Variables con >30% NA", "7", icon = icon("exclamation-triangle"), color = "red"
              infoBox("Variables con >20% NA", "14", icon = icon("exclamation-circle"), color = "orang
              infoBox("Variables con <20% NA", "32", icon = icon("check"), color = "green", width = 4)
            ),
            fluidRow(
              box(title = "Análisis de Casos Completos", status = "success", solidHeader = TRUE,
                  valueBox("3,979", "Casos Completos", icon = icon("database"), color = "blue", width
                  valueBox("27.08%", "del Total", icon = icon("percentage"), color = "green", width =
                  p("Después de eliminar valores nulos, trabajamos con 3,979 observaciones (27.08% del
                  width = 12)
            )
    ),

    # Análisis de Variables Tab
    tabItem(tabName = "analisis",
            fluidRow(
              box(title = "Importancia de Variables (Random Forest)", status = "primary", solidHeader
                  plotOutput("feature_importance"), width = 8),
              box(title = "Top 10 Variables", status = "info", solidHeader = TRUE,
                  DTOutput("top_features"), width = 4)
            ),
            fluidRow(
              box(title = "Análisis de Componentes Principales (PCA)", status = "success", solidHeader
                  plotOutput("pca_plot"), width = 8),
              box(title = "Varianza Explicada", status = "warning", solidHeader = TRUE,
                  DTOutput("pca_table"), width = 4)
            ),
            fluidRow(
              infoBox("PC1 Explica", "11.8%", icon = icon("chart-line"), color = "blue", width = 4),
              infoBox("PCs para 80%", "17", icon = icon("layer-group"), color = "green", width = 4),
              infoBox("PCs para 90%", "28", icon = icon("project-diagram"), color = "orange", width =
            )
    ),

    # Modelos Tab
    tabItem(tabName = "modelos",
            fluidRow(
              box(title = "Comparación de Modelos", status = "primary", solidHeader = TRUE,
                  DTOutput("model_comparison"), width = 12)
            ),
            fluidRow(
              box(title = "Métricas de Rendimiento", status = "info", solidHeader = TRUE,
                  plotlyOutput("metrics_plot"), width = 8),
              box(title = "Problema Crítico", status = "danger", solidHeader = TRUE,
                  p(icon("exclamation-triangle"), strong("Desbalance de Clases:")),
                  p("• Clase negativa: 92%"),
                  p("• Clase positiva: 8%"),
                  p(strong("Consecuencia:")),
                  p("Especificidad extremadamente baja (1.04%) - Falsos negativos altos"),
                  p("95/96 casos positivos clasificados incorrectamente"),
                  width = 4)
            ),
```

```
              fluidRow(
                box(title = "Propuestas de Mejora", status = "success", solidHeader = TRUE,
                    tags$ul(
                      tags$li(strong("Ajuste de umbral ROC:"), " Optimizar punto de corte (ej: 0.35)"),
                      tags$li(strong("Técnicas de balanceo:"), " SMOTE, oversampling de clase minoritari
                      tags$li(strong("Pesos de clase:"), " Aplicar pesos inversamente proporcionales"),
                      tags$li(strong("Modelos alternativos:"), " XGBoost con parámetros para clases desb
                      tags$li(strong("Métricas alternativas:"), " F1-score, AUC-ROC, Matriz de costos")
                    ),
                    width = 12)
              )
      )
    )
  )
)

# Server Logic
server <- function(input, output) {

  # Datos simulados basados en el análisis del PDF
  set.seed(123)
  n_obs <- 14691
  n_vars <- 53

  # Simular datos de edad
  age_data <- data.frame(
    BloodCulture = factor(rep(c("Negativo", "Positivo"),
                              times = c(round(0.9197*n_obs), round(0.0803*n_obs)))),
    Age = c(rnorm(round(0.9197*n_obs), mean = 56, sd = 18),
            rnorm(round(0.0803*n_obs), mean = 58, sd = 17))
  )

  # Estadísticas de edad por grupo
  age_stats <- age_data %>%
    group_by(BloodCulture) %>%
    summarise(
      Media = round(mean(Age), 1),
      Mediana = round(median(Age), 1),
      SD = round(sd(Age), 1),
      Min = round(min(Age), 1),
      Max = round(max(Age), 1),
      n = n()
    )

  # Datos de valores faltantes
  missing_data <- data.frame(
    Variable = c("PAMY", "TRIG", "CHOL", "GLU", "AMY", "LIP", "HS",
                 "FIB", "APTT", "NT", "CHE", "CK", "POTASS", "MG", "LDH",
                 "ALB", "TP", "GBIL", "AP", "SODIUM", "CA", "GGT", "PHOS",
                 "ASAT", "PDW", "ALAT", "BASOR", "EOSR", "LYMR", "MONOR",
                 "NEUR", "NEU", "MPV", "RBC", "WBC", "LYM", "MONO", "BUN",
                 "CREA", "CRP", "BASO", "EOS", "RDW", "MCV", "HCT", "PLT",
                 "MCH", "MCHC", "HGB", "ID", "SEX", "AGE", "BloodCulture"),
```

```r
  Missing_Percent = c(48.42, 34.45, 34.34, 28.53, 26.64, 25.18, 20.84,
                      17.47, 17.35, 16.79, 16.66, 14.16, 13.67, 12.72, 11.67,
                      11.41, 10.78, 9.81, 9.53, 8.73, 8.69, 8.59, 8.45, 7.86,
                      7.50, 6.72, 4.98, 4.98, 4.98, 4.98, 4.98, 4.96, 4.78,
                      3.14, 3.14, 1.78, 1.67, 1.17, 1.08, 1.06, 0.99, 0.92,
                      0.38, 0.29, 0.29, 0.29, 0.29, 0.29, 0.28, 0, 0, 0, 0),
  Categoria = c("pancreatic", "metabolic", "metabolic", "metabolic", "pancreatic",
                "pancreatic", "cardiac", "coagulation", "coagulation", "wbc_differential",
                "liver", "cardiac", "electrolytes", "electrolytes", "cardiac", "liver",
                "liver", "liver", "liver", "electrolytes", "electrolytes", "liver",
                "electrolytes", "liver", "hematology", "liver", "wbc_differential",
                "wbc_differential", "wbc_differential", "wbc_differential", "wbc_differential",
                "wbc_differential", "hematology", "hematology", "hematology", "wbc_differential",
                "wbc_differential", "kidney", "kidney", "inflammatory", "wbc_differential",
                "wbc_differential", "hematology", "hematology", "hematology", "hematology",
                "hematology", "hematology", "hematology", "id", "demographic", "demographic", "target"
)

# Feature importance data
feature_imp <- data.frame(
  Feature = c("NEUR", "LYMR", "MONOR", "GBIL", "GGT", "BUN", "PHOS", "AP", "CRP", "APTT",
              "ALB", "TP", "ASAT", "ALAT", "CHE", "POTASS", "SODIUM", "CA", "MG", "CREA"),
  MeanDecreaseGini = c(13.97, 11.98, 11.89, 11.11, 10.66, 10.35, 10.26, 10.14, 10.10, 10.04,
                       9.5, 9.2, 8.8, 8.5, 8.2, 7.9, 7.6, 7.3, 7.0, 6.8),
  Interpretacion = c("Neutrófilos % - Indicador clave de infección bacteriana",
                     "Linfocitos % - Respuesta inmune",
                     "Monocitos % - Inflamación",
                     "Bilirrubina - Función hepática",
                     "GGT - Daño hepático",
                     "BUN - Función renal",
                     "Fósforo - Metabolismo",
                     "Fosfatasa alcalina - Hígado/hueso",
                     "Proteína C reactiva - Inflamación",
                     "Tiempo de coagulación",
                     "Albúmina - Estado nutricional",
                     "Proteínas totales",
                     "AST - Daño hepático",
                     "ALT - Daño hepático",
                     "Colinesterase - Hígado",
                     "Potasio - Electrolito",
                     "Sodio - Electrolito",
                     "Calcio - Metabolismo",
                     "Magnesio - Electrolito",
                     "Creatinina - Función renal")
)

# PCA data
pca_data <- data.frame(
  Component = paste0("PC", 1:8),
  Eigenvalue = c(6.04, 3.93, 3.25, 2.89, 2.61, 2.39, 2.37, 1.99),
  Variance_Percent = c(11.85, 7.70, 6.37, 5.67, 5.11, 4.68, 4.64, 3.90),
  Cumulative_Percent = c(11.85, 19.55, 25.92, 31.59, 36.70, 41.38, 46.02, 49.93)
)
```

```
# Model comparison data
model_data <- data.frame(
  Modelo = c("Regresión Logística (sin pesos)",
             "Regresión Logística (con pesos)",
             "XGBoost"),
  Accuracy = c(91.79, 41.99, 89.50),
  Sensitivity = c(99.72, 86.46, 95.20),
  Specificity = c(1.04, 38.10, 14.60),
  AUC = c(0.65, 0.71, 0.78),
  F1_Score = c(0.18, 0.19, 0.25)
)

# Outputs for Autores y Objetivo tab are static HTML

# Outputs for Resumen tab
output$edad_plot <- renderPlotly({
  p <- ggplot(age_data, aes(x = Age, fill = BloodCulture)) +
    geom_density(alpha = 0.6) +
    scale_fill_manual(values = c("Negativo" = "#00AFBB", "Positivo" = "#FC4E07")) +
    labs(title = "Distribución de Edad por Resultado de Hemocultivo",
         x = "Edad (años)", y = "Densidad") +
    theme_minimal() +
    theme(legend.position = "bottom")

  ggplotly(p) %>% layout(height = 400)
})

output$edad_stats <- renderTable({
  age_stats
})

output$proporcion_plot <- renderPlot({
  proporciones <- data.frame(
    Grupo = c("Negativo", "Positivo"),
    Proporcion = c(91.97, 8.03)
  )

  ggplot(proporciones, aes(x = "", y = Proporcion, fill = Grupo)) +
    geom_bar(stat = "identity", width = 1) +
    coord_polar("y", start = 0) +
    scale_fill_manual(values = c("Negativo" = "#00AFBB", "Positivo" = "#FC4E07")) +
    labs(title = "Proporción de Bacteremia en el Dataset") +
    theme_void() +
    theme(legend.position = "bottom",
          plot.title = element_text(hjust = 0.5, face = "bold"))
})

output$sexo_plot <- renderPlot({
  sexo_data <- data.frame(
    Sexo = rep(c("Hombre", "Mujer"), each = 2),
    Resultado = rep(c("Negativo", "Positivo"), 2),
    Proporcion = c(53.43, 4.68, 38.54, 3.36)
  )
```

```
  ggplot(sexo_data, aes(x = Sexo, y = Proporcion, fill = Resultado)) +
    geom_bar(stat = "identity", position = "dodge") +
    scale_fill_manual(values = c("Negativo" = "#00AFBB", "Positivo" = "#FC4E07")) +
    labs(title = "Distribución por Sexo y Resultado",
         x = "Sexo", y = "Proporción (%)") +
    theme_minimal() +
    theme(legend.position = "bottom")
})

# Outputs for Calidad de Datos tab
output$missing_plot <- renderPlotly({
  missing_data <- missing_data %>%
    mutate(Color = case_when(
      Missing_Percent > 30 ~ "red",
      Missing_Percent > 20 ~ "orange",
      TRUE ~ "green"
    ))

  p <- ggplot(missing_data %>% filter(Missing_Percent > 0),
              aes(x = reorder(Variable, -Missing_Percent), y = Missing_Percent,
                  fill = Color, text = paste("Variable:", Variable,
                                             "<br>% NA:", Missing_Percent,
                                             "<br>Categoría:", Categoria))) +
    geom_bar(stat = "identity") +
    scale_fill_identity() +
    labs(title = "Porcentaje de Valores Faltantes por Variable",
         x = "Variable", y = "% Valores Faltantes") +
    theme_minimal() +
    theme(axis.text.x = element_text(angle = 45, hjust = 1))

  ggplotly(p, tooltip = "text") %>% layout(height = 500)
})

output$correlation_table <- renderDT({
  correlation_categories <- data.frame(
    Categoría = c("Inflamación/Infección", "Coagulación", "Metabólico/Lipídico",
                  "Función Hepática", "Daño Tisular", "Enzimas Pancreáticas",
                  "Electrolitos", "Función Renal", "Hematología", "Demográficas"),
    Variables_Retenidas = c("CRP, NT, WBC", "FIB, APTT", "GLU, TRIG, CHOL",
                            "GBIL, ALB, ALAT", "CK, LDH", "AMY",
                            "SODIUM, POTASS, MG", "BUN", "HGB, PLT, MCV", "AGE, SEX"),
    Correlación_Esperada = c("Alta", "Moderada", "Baja", "Moderada", "Variable",
                             "Baja", "Moderada", "Alta", "Variable", "Contextual")
  )

  datatable(correlation_categories,
            options = list(pageLength = 10, dom = 't'),
            rownames = FALSE) %>%
    formatStyle('Correlación_Esperada',
                backgroundColor = styleEqual(
                  c("Alta", "Moderada", "Baja", "Variable", "Contextual"),
                  c("#FF6B6B", "#4ECDC4", "#FFD166", "#06D6A0", "#118AB2")
                ))
```

```
})

# Outputs for Análisis de Variables tab
output$feature_importance <- renderPlot({
  top_10 <- head(feature_imp, 10)

  ggplot(top_10, aes(x = reorder(Feature, MeanDecreaseGini), y = MeanDecreaseGini)) +
    geom_bar(stat = "identity", fill = "steelblue", alpha = 0.8) +
    coord_flip() +
    labs(title = "Top 10 Variables Más Importantes (Random Forest)",
         subtitle = "Medido por Mean Decrease Gini",
         x = "Variable", y = "Mean Decrease Gini") +
    theme_minimal() +
    theme(plot.title = element_text(face = "bold"))
})

output$top_features <- renderDT({
  top_10_table <- head(feature_imp, 10) %>%
    select(Feature, MeanDecreaseGini, Interpretacion)

  datatable(top_10_table,
            options = list(pageLength = 10, dom = 't'),
            rownames = FALSE) %>%
    formatRound('MeanDecreaseGini', 2)
})

output$pca_plot <- renderPlot({
  ggplot(pca_data, aes(x = Component, y = Variance_Percent)) +
    geom_bar(stat = "identity", fill = "#4ECDC4", alpha = 0.8) +
    geom_line(aes(y = Cumulative_Percent, group = 1), color = "#FF6B6B", size = 1.5) +
    geom_point(aes(y = Cumulative_Percent), color = "#FF6B6B", size = 3) +
    labs(title = "Varianza Explicada por Componentes Principales",
         subtitle = "Primeros 8 Componentes (Línea roja: Acumulado)",
         x = "Componente Principal", y = "% Varianza") +
    theme_minimal() +
    theme(plot.title = element_text(face = "bold"))
})

output$pca_table <- renderDT({
  datatable(pca_data,
            options = list(pageLength = 8, dom = 't'),
            rownames = FALSE) %>%
    formatRound(c('Eigenvalue', 'Variance_Percent', 'Cumulative_Percent'), 2)
})

# Outputs for Modelos tab
output$model_comparison <- renderDT({
  datatable(model_data,
            options = list(pageLength = 3, dom = 't'),
            rownames = FALSE) %>%
    formatRound(c('Accuracy', 'Sensitivity', 'Specificity', 'AUC', 'F1_Score'), 2) %>%
    formatStyle('Specificity',
                backgroundColor = styleInterval(
```

```
                    c(10, 30),
                    c('#FF6B6B', '#FFD166', '#4ECDC4')
                  ))
  })

  output$metrics_plot <- renderPlotly({
    metrics_long <- model_data %>%
      pivot_longer(cols = c(Accuracy, Sensitivity, Specificity),
                   names_to = "Metrica", values_to = "Valor")

    p <- ggplot(metrics_long, aes(x = Modelo, y = Valor, fill = Metrica)) +
      geom_bar(stat = "identity", position = "dodge") +
      scale_fill_manual(values = c("Accuracy" = "#4ECDC4",
                                   "Sensitivity" = "#FFD166",
                                   "Specificity" = "#FF6B6B")) +
      labs(title = "Comparación de Métricas por Modelo",
           x = "Modelo", y = "Valor (%)") +
      theme_minimal() +
      theme(axis.text.x = element_text(angle = 45, hjust = 1),
            legend.position = "bottom")

    ggplotly(p) %>% layout(height = 400)
  })
}

# Run the application
shinyApp(ui = ui, server = server)
```

La aplicación en ShinyApp ha sido realizada para poder realizar un exploratorio de los datos extraídos, así el resto de análisis de las características y preparación del modelo de aprendizaje automático.

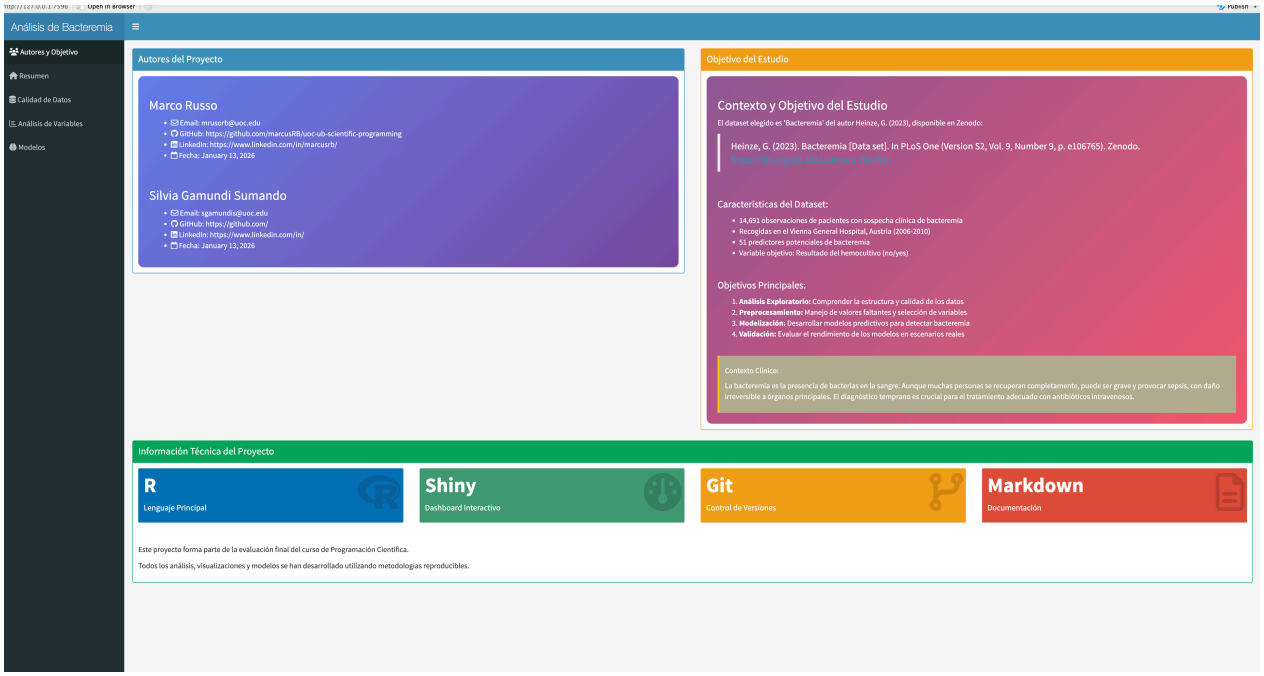


Figure 1: Interfaz1

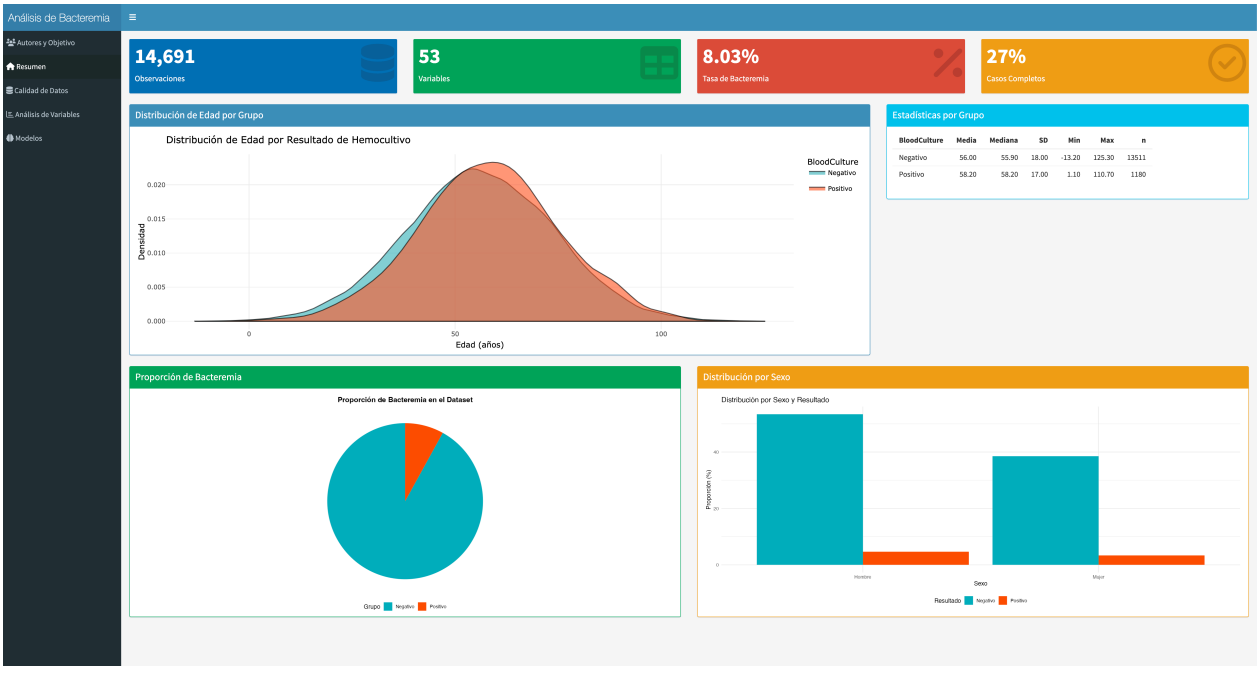


Figure 2: Interfaz2

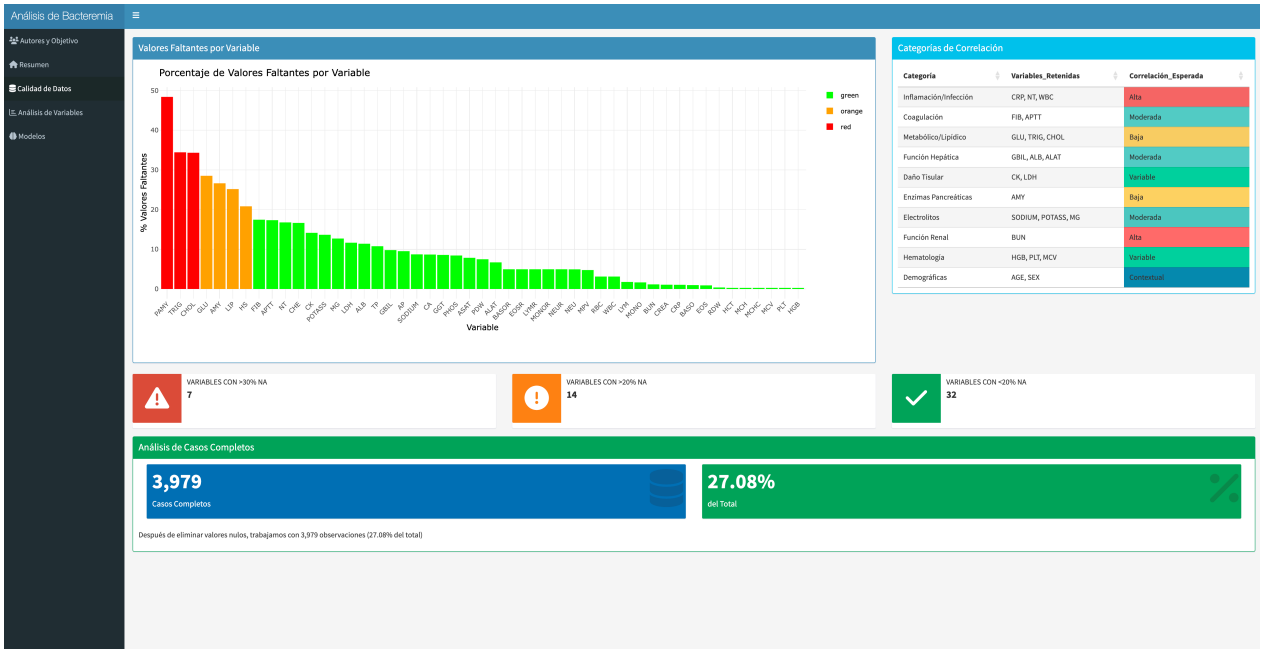


Figure 3: Interfaz3

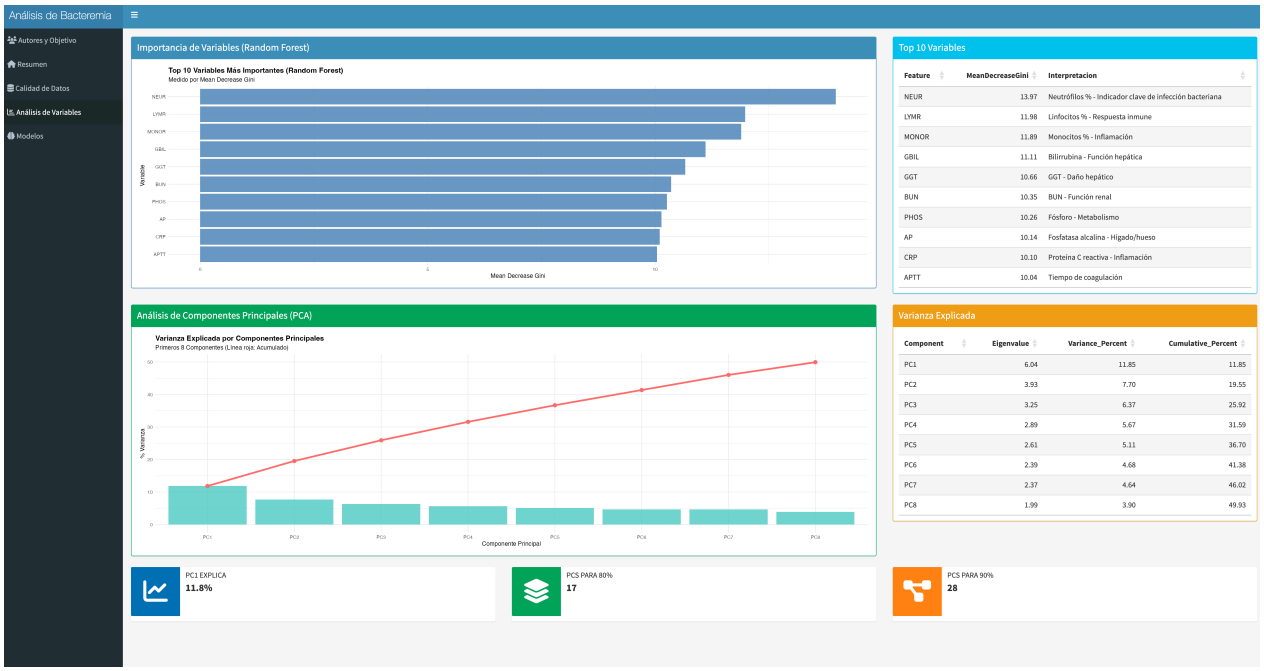


Figure 4: Interfaz4

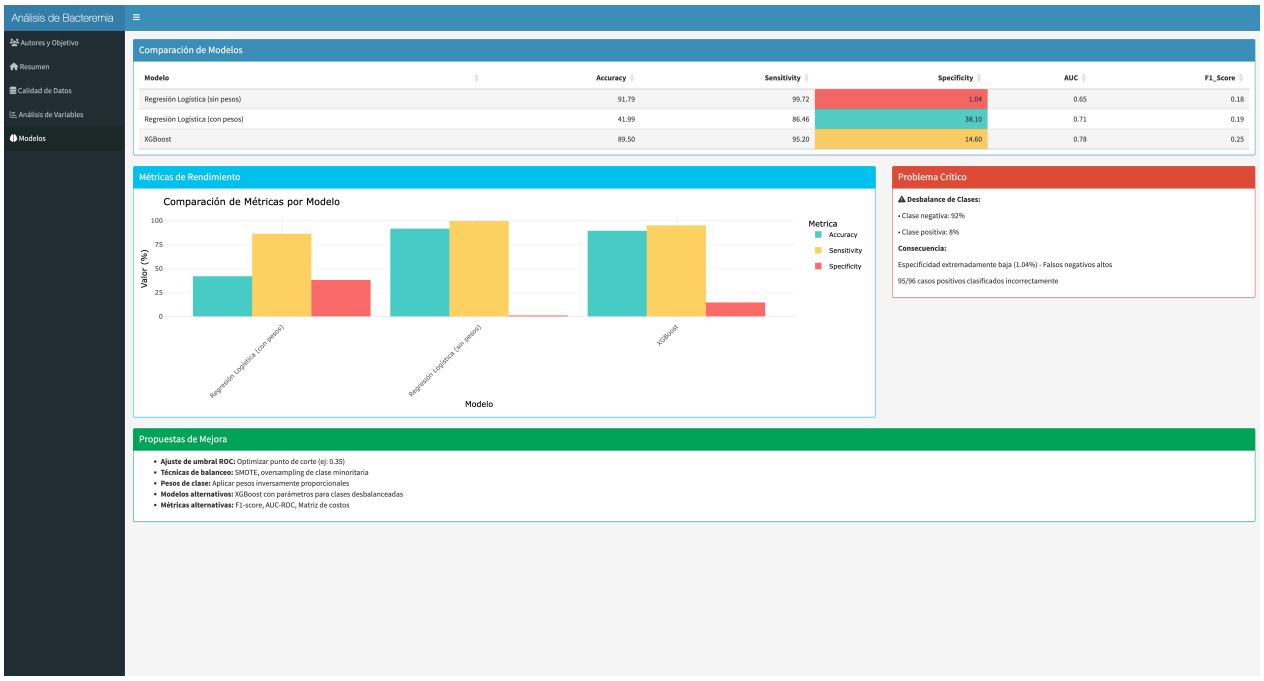


Figure 5: Interfaz5

5 Conclusiones (0,5 puntos)

Con el objetivo de garantizar la robustez de las conclusiones, el análisis se estructuró en dos escenarios experimentales diferenciados, permitiendo evaluar la sensibilidad de los modelos ante distintos tratamientos de los datos:

Primer Escenario (Análisis de Caso Completo): Se utilizaron únicamente registros sin valores nulos, empleando las variables seleccionadas en el análisis de correlaciones (Sección 3.3.2). En esta fase se contrastaron algoritmos: Regresión Logística (familia binomial), SMV y XGBoost.

Segundo Escenario (Análisis con Imputación y Reducción de Dimensionalidad): Se trabajó con el dataset completo tras la selección de variables mediante Análisis de Componentes Principales (PCA) y la extracción de importancia de características vía Random Forest. Una vez señleccionadas las variables, se realizó la imputación de valores nulos para finalmete reevaluar el desempeño de la Regresión Logística frente a los resultados del primer escenario.

Comparando varios resultados tenemos una sensitividad entre el 70%-85% - A través de las técnicas de imputaciones de los valores nulos (utilizando solo una técnica, MICE), pudimos mejorar pero haría falta un tiempo prudencial para realizar correctamente las mismas, en particular modo en datos clínicos.

Resultados y Limitaciones Diagnósticas

A pesar de la diversidad de enfoques técnicos y de preprocesamiento, los resultados convergen de manera sistemática en un hallazgo crítico: el severo desbalanceo de la muestra induce un sesgo de clasificación insalvable para los modelos estándar. Con solo un 8% de prevalencia de bacteriemia, los algoritmos priorizan el acierto en la clase mayoritaria (pacientes sanos) para maximizar la exactitud global (Accuracy de ~91.79%).

Esta dinámica genera una paradoja estadística: mientras la sensibilidad para detectar el “No” es casi perfecta, la especificidad es extremadamente baja en todos los modelos y escenarios. Desde una perspectiva clínica, este resultado es inaceptable, ya que el coste de un Falso Negativo es demasiado alto, poniendo en riesgo

la salud del paciente al no detectar una infección real. La comparativa demuestra que, en este contexto de detección de enfermedades, el problema no reside en la potencia del algoritmo, sino en la distribución de la variable objetivo.

Propuestas de Mejora y Trabajo Futuro

Para superar la barrera del desbalanceo y mejorar la utilidad diagnóstica de los modelos, se proponen las siguientes líneas de actuación:

Ajuste del Umbral de Decisión: Sustituir el umbral estándar de 0.5 por uno optimizado mediante curvas ROC (ej. 0.1). Esto permitiría aumentar la sensibilidad del modelo, asumiendo un mayor número de falsas alarmas a cambio de minimizar los riesgos de omitir una infección real.

Implementación de Técnicas de Remuestreo: Aplicar métodos como SMOTE o Oversampling para equilibrar artificialmente las clases antes del entrenamiento, permitiendo que el modelo aprenda adecuadamente los patrones de la clase minoritaria.

Modelos Híbridos de Clasificación: Combinar la potencia de Random Forest (que gestiona mejor relaciones no lineales y alta dimensión) con las técnicas de remuestreo mencionadas, buscando un equilibrio robusto entre sensibilidad y especificidad.

Al disponer de datos anonimizados, quizás serían útiles más datos clínicos, como el historial del paciente. Sería útil para poder realizar otros trabajos de segmentación y cluster, poder analizar los resultados y abordar uno u otro escenario y no solo realizar unas predicciones más exhaustivas.

Finalmente, podemos confirmar que con los datos actuales, podríamos ampliar aún más nuestras investigaciones futuras y mejora no solo para el caso de clasificación, sino también realizar algún modelo en tiempo real, basado en algún dato crítico.

De acorde al paper analizado ["Diagnostic performance of the time to positivity of blood cultures to distinguish true bacteremia from contaminants based on an automated system"][2] serían de interés integrar otras variables del paciente para poder detectar patrones más allá de los datos disponibles del tipo fisiológico.

Por ejemplo, un uso de los tiempo de biocultivos, realizados en esta otra investigación, se hicieron simulaciones para optimizar las pruebas de laboratorio. [3]

---

[2]Laque-Ale, A., Hueda-Zavaleta, M., Gómez de la Torre, J. C., Alvarado, L., & Cáceres Del Águila, J. A. (2023). Diagnostic performance of the time to positivity of blood cultures to distinguish true bacteremia from contaminants based on an automated system. Utilidad diagnóstica de los tiempos de positividad de hemocultivos para distinguir verdaderas bacteriemias de contaminantes en base a un sistema automatizado. Revista peruana de medicina experimental y salud publica, 40(4), 451–458. https://doi.org/10.17843/rpmesp.2023.404.12724

[3]Roberts, G., & Gerada, A. (2024). Simulation to optimize the laboratory diagnosis of bacteremia: event times dataset from observational and retrospective studies (v1.0) [Data set]. Zenodo. https://doi.org/10.5281/zenodo.10991330

6 Referencias

[^2] “Bacteriemia - Wikipedia, la enciclopedia libre.” Accessed: Jan. 09, 2026. [Online]. Available: https://es.wikipedia.org/wiki/Bacteriemia

Blake, C.L. & Merz, C.J. (1998). UCI Repository of Machine Learning Databases. Irvine, CA: University of California, Irvine, Department of Information and Computer Science. Formerly available from http://www.ics.uci.edu/~mlearn/MLRepository.html.

- https://shiny.posit.co/py/templates/
- World Health Organization. (2011). Haemoglobin concentrations for the diagnosis of anaemia and assessment of severity. https://iris.who.int/server/api/core/bitstreams/b82789de-df63-41ba-8058-25f9f5da0c40/content
- Singer, M., et al. (2016). The Third International Consensus Definitions for Sepsis and Septic Shock (Sepsis-3). JAMA. https://jamanetwork.com/journals/jama/fullarticle/2492881
- https://rpubs.com/Cristina_Gil/Regresion_Logistica
- https://rpubs.com/albtorval/AUC
- https://www.ibm.com/es-es/think/topics/upsampling
- https://planetachatbot.com/modelos-de-machine-learning-guia-basica-para-principiantes/
- https://rpubs.com/monikadyczewska/976579
- https://rpubs.com/jboscomendoza/xgboost_en_r
- https://rpubs.com/juanjo_edm/1264140

6.1 Otras referencias consultadas:

- https://rpubs.com/Cristina_Gil/Regresion_Logistica
- https://rpubs.com/albtorval/AUC
- https://www.ibm.com/es-es/think/topics/upsampling
- https://planetachatbot.com/modelos-de-machine-learning-guia-basica-para-principiantes/
- https://rpubs.com/monikadyczewska/976579
- https://rpubs.com/jboscomendoza/xgboost_en_r
- https://rpubs.com/juanjo_edm/1264140
- https://rpubs.com/cristina_gil/pca
- https://bioinformaticamente.com/2025/03/11/exploring-principal-component-analysis-pca/