PRA2

Luis Villazón Esteban, Jose Javier Marti Camarasa

12/21/2020

Descripción del dataset. ¿Por qué es importante y qué pregunta/problema pretende responder?

El dataset seleccionado contiene datos referentes a pacientes Indios que sufren de Hígado. El problema que se pretende responder con los datos facilitados es la clasificación de pacientes para saber si en función de sus múltiples atributos es un paciente que sufre de Hígado o no. Para ello el dataset nos ofrece 583 pacientes, de los cuales 416 se encuentran identificados como pacientes que sufren de Hígado y 167 como pacientes que no tienen problemas relacionados con el mismo.

El dataset contiene los siguientes atributos:

- **age**: Edad del paciente, todo aquel paciente cuya edad sea superior a 89 es marcado como 90.
- **gender** Sexo del paciente.
- **tot_bilirubin** Bilirubina Total.
- **direct_bilirubin** Bilirubina en sangre.
- **alkphos** Fosfatasa Alcalina(Niveles altos pueden indicar daño en el hígado).
- **sgpt** Test Alamina aminotransferasa: Test sanguineo para comprobar si hay daño en hígado.
- **sgot** Test Aspartato Aminotransferasa: Test sanguineo para comprobar si hay daño en hígado.
- **tot_proteins** roteinas totales.
- **albumin** Albumina.
- **ag_ratio** A/G Ratio Albumina y Globulina (Grupo de proteinas solubles en sangre)
- **is_patient** Selector usado para indicar si es paciente de higado. 1 significa si, 2 significa no.

Integración y selección de los datos de interés a analizar.

En primer lugar realizamos la carga del dataset en R y transformamos la variable dependiente a tipo factor, transformando el valor 1 a "si_padece" y el valor 2 a "no_padece".

```
ilpd_data <- read.csv("ilpd_data.csv",header = FALSE, col.names = c("edad","sexo","TB","DB","alk_phos",'

ilpd_data <- ilpd_data%>% mutate(
    Padece = as.factor(case_when(
      Padece == "1"  ~ "si_padece",
      Padece == "2" ~ "no_padece"
           ))
)
```

A continuación mostramos un resumen y una descripción de los valores del dataset.

```
summary(ilpd_data)
```

```
##       edad           sexo           TB               DB
##  Min.   : 4.00   Female:142   Min.   : 0.400   Min.   : 0.100
```

```
##  1st Qu.:33.00   Male  :441   1st Qu.: 0.800   1st Qu.: 0.200
##  Median :45.00                Median : 1.000   Median : 0.300
##  Mean   :44.75                Mean   : 3.299   Mean   : 1.486
##  3rd Qu.:58.00                3rd Qu.: 2.600   3rd Qu.: 1.300
##  Max.   :90.00                Max.   :75.000   Max.   :19.700
##     alk_phos         alamine          aspartate           TP
##  Min.   :  63.0   Min.   :  10.00   Min.   :  10.0   Min.   :2.700
##  1st Qu.: 175.5   1st Qu.:  23.00   1st Qu.:  25.0   1st Qu.:5.800
##  Median : 208.0   Median :  35.00   Median :  42.0   Median :6.600
##  Mean   : 290.6   Mean   :  80.71   Mean   : 109.9   Mean   :6.483
##  3rd Qu.: 298.0   3rd Qu.:  60.50   3rd Qu.:  87.0   3rd Qu.:7.200
##  Max.   :2110.0   Max.   :2000.00   Max.   :4929.0   Max.   :9.600
##     albumin           A.G              Padece
##  Min.   :0.900   Min.   :0.3000   no_padece:167
##  1st Qu.:2.600   1st Qu.:0.7000   si_padece:416
##  Median :3.100   Median :0.9200
##  Mean   :3.142   Mean   :0.9443
##  3rd Qu.:3.800   3rd Qu.:1.1000
##  Max.   :5.500   Max.   :2.8000
```

```
str(ilpd_data)
```

```
## 'data.frame':    583 obs. of  11 variables:
##  $ edad     : int  65 62 62 58 72 46 26 29 17 55 ...
##  $ sexo     : Factor w/ 2 levels "Female","Male": 1 2 2 2 2 2 1 1 2 2 ...
##  $ TB       : num  0.7 10.9 7.3 1 3.9 1.8 0.9 0.9 0.9 0.7 ...
##  $ DB       : num  0.1 5.5 4.1 0.4 2 0.7 0.2 0.3 0.3 0.2 ...
##  $ alk_phos : int  187 699 490 182 195 208 154 202 202 290 ...
##  $ alamine  : int  16 64 60 14 27 19 16 14 22 53 ...
##  $ aspartate: int  18 100 68 20 59 14 12 11 19 58 ...
##  $ TP       : num  6.8 7.5 7 6.8 7.3 7.6 7 6.7 7.4 6.8 ...
##  $ albumin  : num  3.3 3.2 3.3 3.4 2.4 4.4 3.5 3.6 4.1 3.4 ...
##  $ A.G      : num  0.9 0.74 0.89 1 0.4 1.3 1 1.1 1.2 1 ...
##  $ Padece   : Factor w/ 2 levels "no_padece","si_padece": 2 2 2 2 2 2 2 2 1 2 ...
```

Podemos comprobar como todos los valores son númericos continuos excepto las varibles **sexo** y **Padece** las cuales son categorícas y se han detectado correctamente.

Dado que queremos conocer si un paciente padece o no de Hígado, vamos a comprobar cual es la correlación de la variable dependiente **Padece** con cada una de las variables independientes existentes en el Dataset.

```
p <- as.data.frame(model.matrix(~Padece, ilpd_data))
sexo <- as.data.frame(model.matrix(~sexo, ilpd_data))
ilpd_data['p'] <- p$Padecesi_padece
ilpd_data['s'] <- sexo$sexoMale
cor(ilpd_data[, c('edad', 'p', 's', 'TB', 'DB', 'alk_phos', 'alamine', 'alamine', 'TP', 'albumin', 'A.G
```

```
##                   edad           p            s           TB           DB
## edad       1.000000000  0.13735063  0.056560251  0.011762651  0.0075291381
## p          0.137350627  1.00000000  0.082415914  0.220207565  0.2460463416
## s          0.056560251  0.08241591  1.000000000  0.089290824  0.1004364357
## TB         0.011762651  0.22020756  0.089290824  1.000000000  0.8746179301
## DB         0.007529138  0.24604634  0.100436436  0.874617930  1.0000000000
## alk_phos   0.080424612  0.18486561 -0.027496175  0.206668795  0.2349387058
## alamine   -0.086882759  0.16341616  0.082332236  0.214064740  0.2338940545
## alamine.1 -0.086882759  0.16341616  0.082332236  0.214064740  0.2338940545
```

```
## TP         -0.187461261 -0.03500824 -0.089121043 -0.008099343 -0.0001387414
## albumin    -0.265924361 -0.16138782 -0.093799266 -0.222250406 -0.2285305729
## A.G        -0.212965093 -0.15858849  0.002950113 -0.201662048 -0.1952734704
##                alk_phos       alamine     alamine.1            TP     albumin
## edad        0.08042461 -0.0868827586 -0.0868827586 -0.1874612615 -0.26592436
## p           0.18486561  0.1634161567  0.1634161567 -0.0350082358 -0.16138782
## s          -0.02749618  0.0823322363  0.0823322363 -0.0891210427 -0.09379927
## TB          0.20666880  0.2140647402  0.2140647402 -0.0080993434 -0.22225041
## DB          0.23493871  0.2338940545  0.2338940545 -0.0001387414 -0.22853057
## alk_phos    1.00000000  0.1256799509  0.1256799509 -0.0285143556 -0.16545287
## alamine     0.12567995  1.0000000000  1.0000000000 -0.0425181903 -0.02974167
## alamine.1   0.12567995  1.0000000000  1.0000000000 -0.0425181903 -0.02974167
## TP         -0.02851436 -0.0425181903 -0.0425181903  1.0000000000  0.78405334
## albumin    -0.16545287 -0.0297416732 -0.0297416732  0.7840533354  1.00000000
## A.G        -0.22897021  0.0004487473  0.0004487473  0.2326917881  0.67894806
##                    A.G
## edad       -0.2129650935
## p          -0.1585884921
## s           0.0029501125
## TB         -0.2016620476
## DB         -0.1952734704
## alk_phos   -0.2289702051
## alamine     0.0004487473
## alamine.1   0.0004487473
## TP          0.2326917881
## albumin     0.6789480648
## A.G         1.0000000000
```

Limpieza de los datos.

3.1. ¿Los datos contienen ceros o elementos vacíos? ¿Cómo gestionarías cada uno de estos casos?

Según lo observado anteriormente con el uso del método **summary**, no tenemos ninguna variable con datos perdidos. Para tener una visión más clara sobre ello podemos mostrar el número de elementos nulos que existe en cada variable.

```
sort(colMeans(is.na(ilpd_data)), decreasing = TRUE)
```

```
##      edad      sexo        TB        DB  alk_phos   alamine aspartate        TP
##         0         0         0         0         0         0         0         0
##   albumin       A.G    Padece         p         s
##         0         0         0         0         0
```

Efectivamente no tenemos nigun variable con datos perdidos. La funcion colMeans,nos muestra qué proporción de datos no disponibles tenemos por columna.

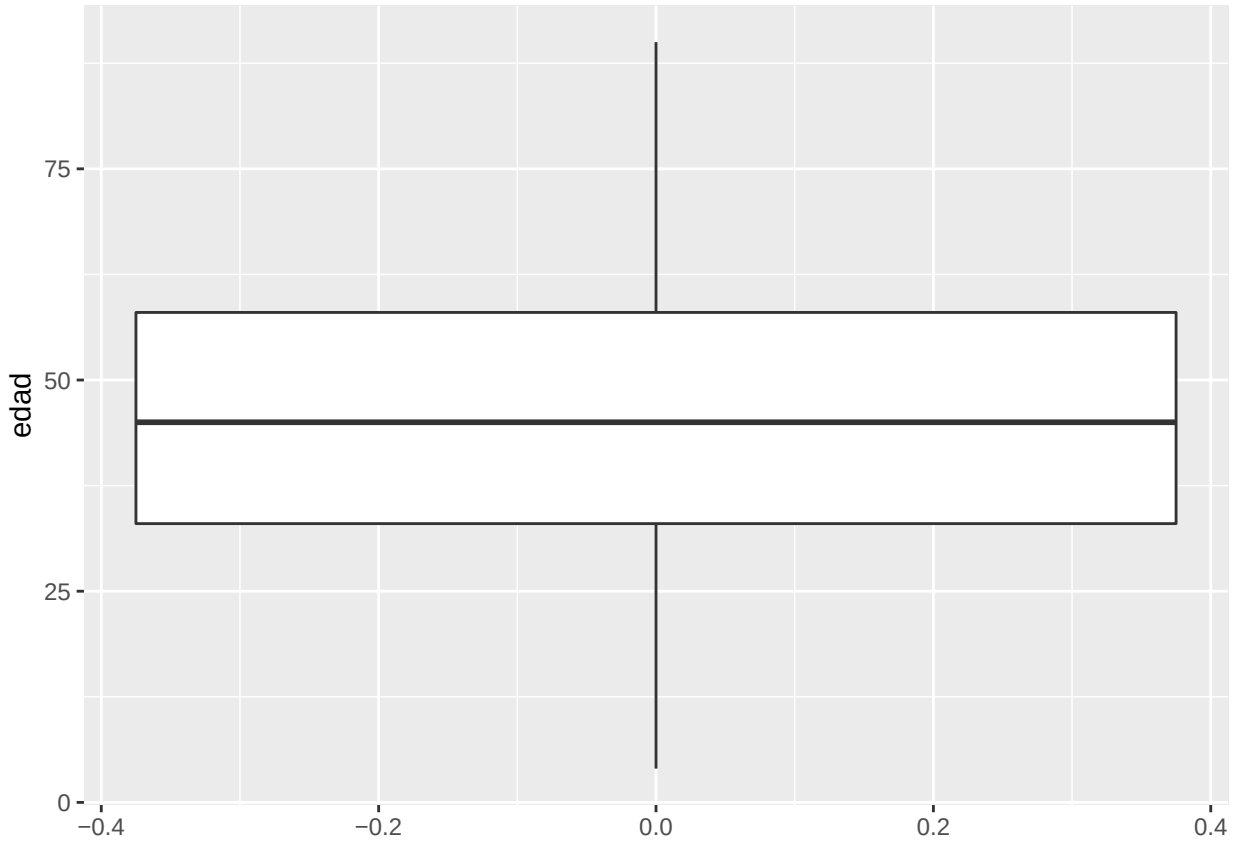
Disponemos por tanto de un data frame formado por 2 variables categóricas y 8 variables exceptuando la variable objetivo, sin valores nulos

3.2. Identificación y tratamiento de valores extremos.

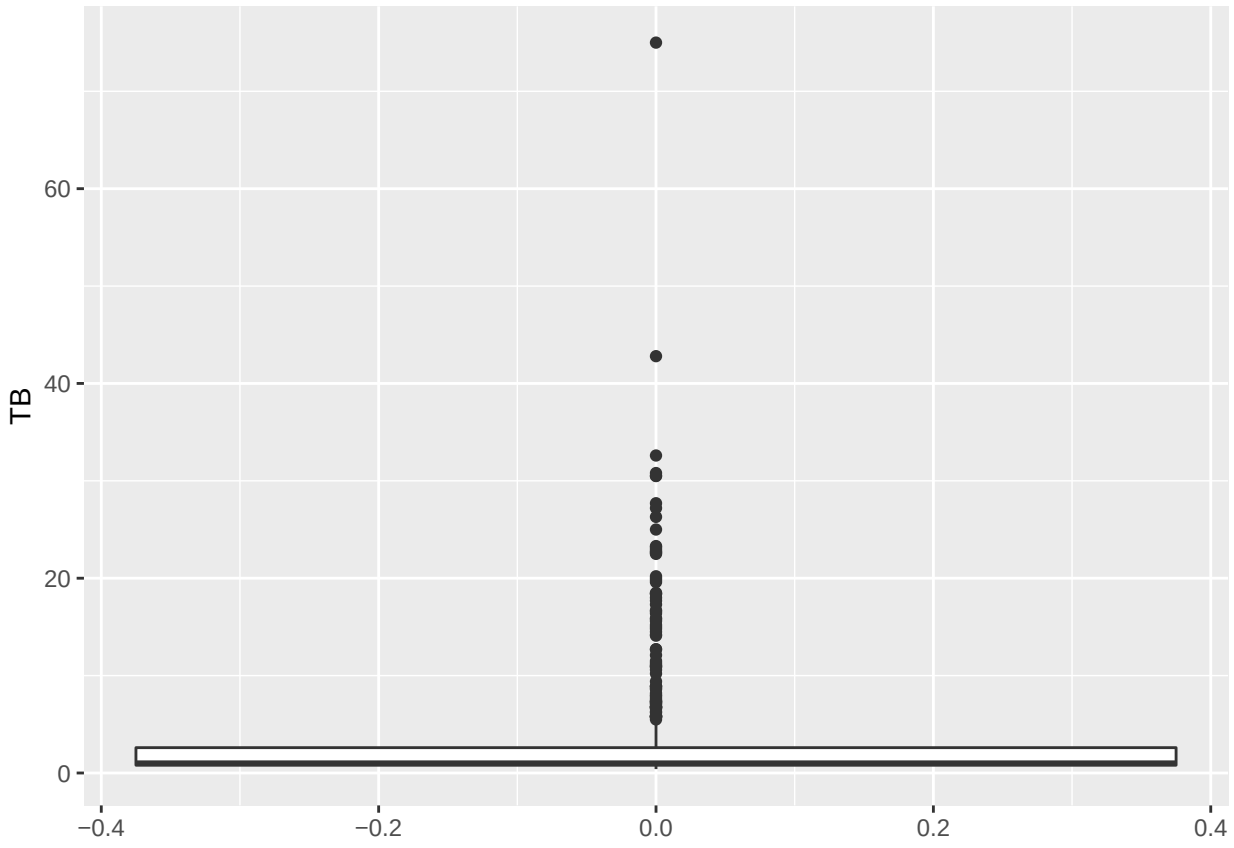
Para identificar los valores extremos vamos a utilizar diagramas de cajas.

```
library("ggplot2")
```

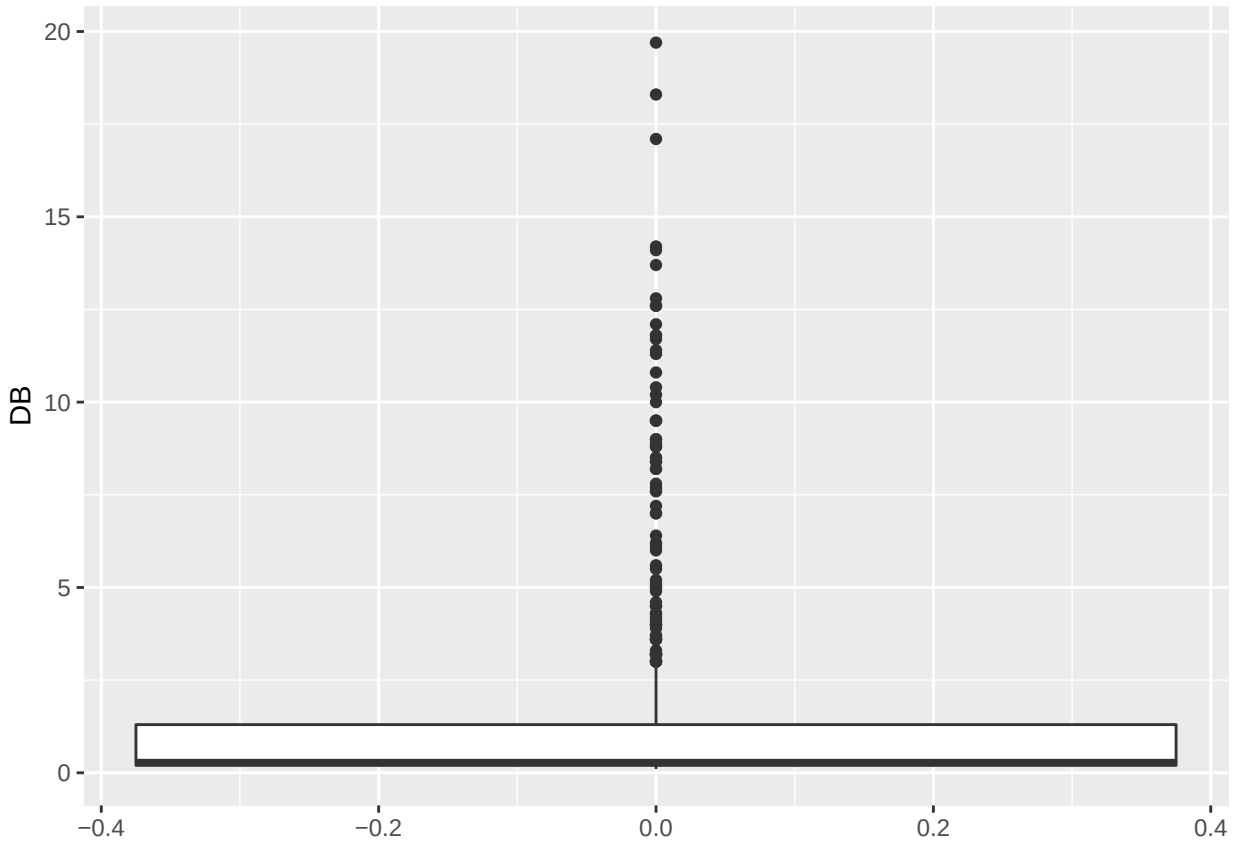
```
ggplot(ilpd_data, aes(y=edad)) + geom_boxplot()
```



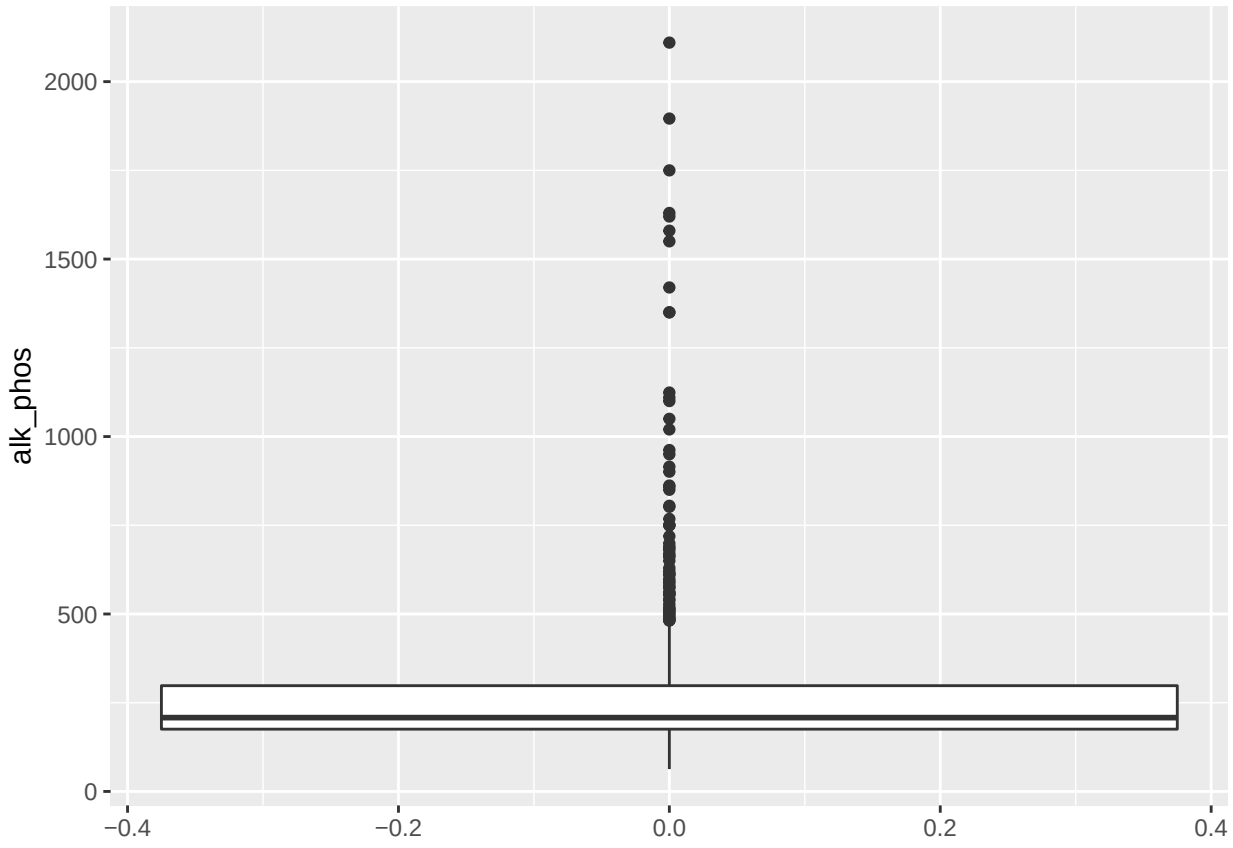

```
ggplot(ilpd_data, aes(y=TB)) + geom_boxplot()
```



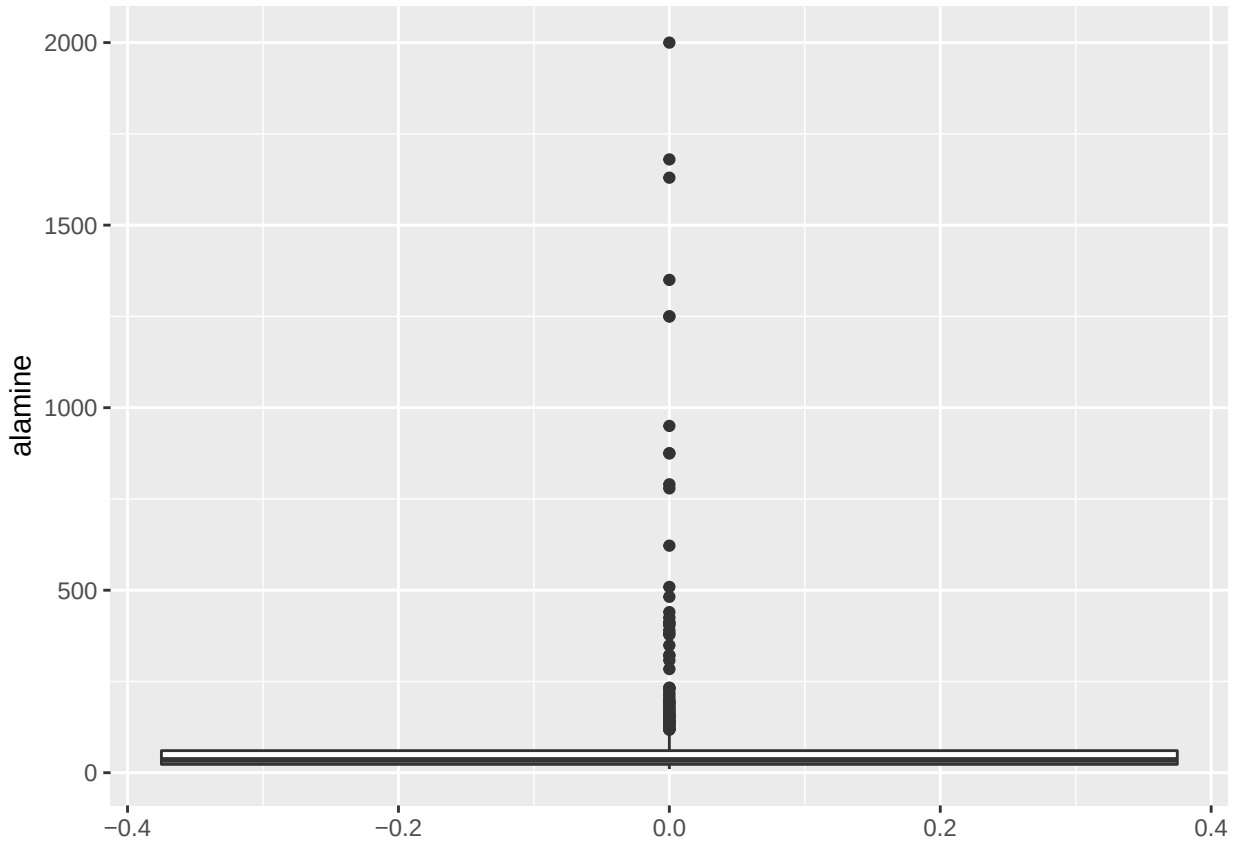

```
ggplot(ilpd_data, aes(y=DB)) + geom_boxplot()
```



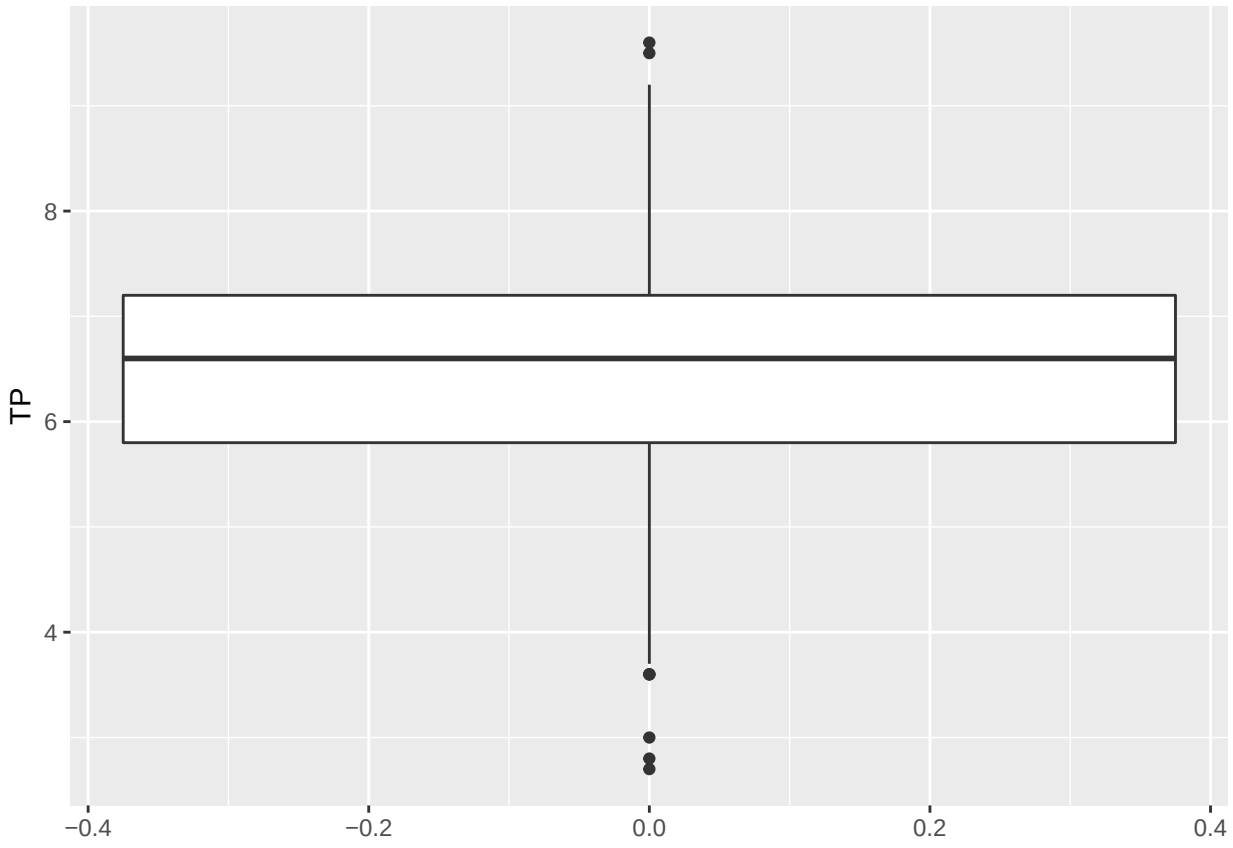

```
ggplot(ilpd_data, aes(y=alk_phos)) + geom_boxplot()
```



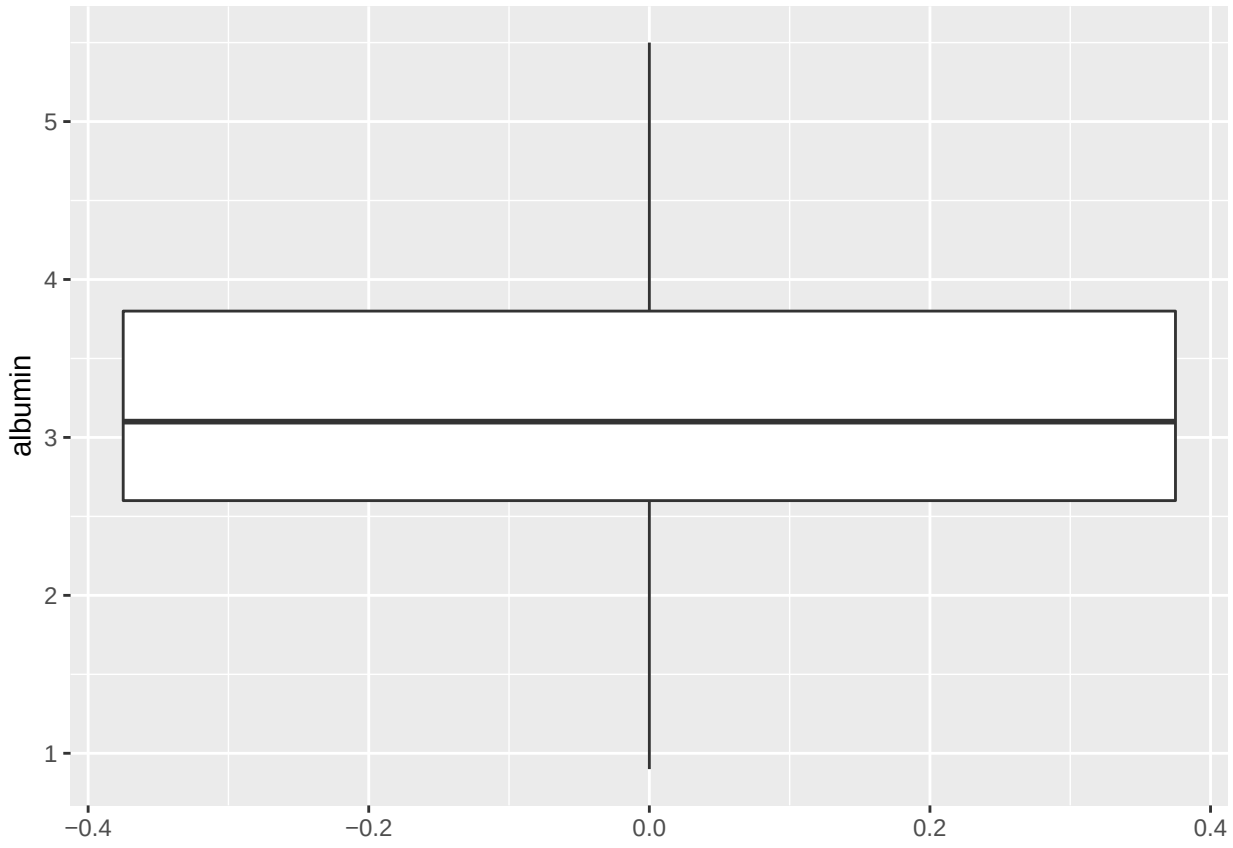

```
ggplot(ilpd_data, aes(y=alamine)) + geom_boxplot()
```



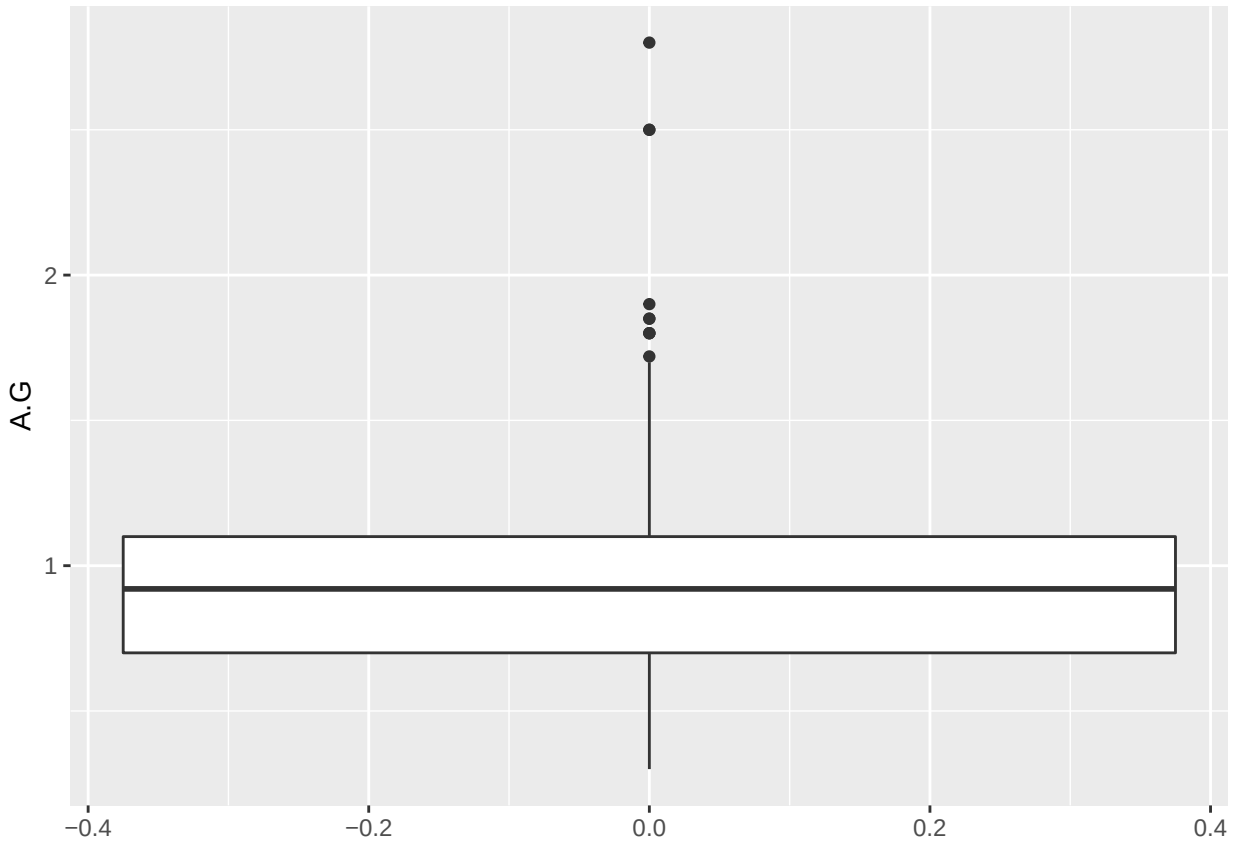

```
ggplot(ilpd_data, aes(y=TP)) + geom_boxplot()
```




```
ggplot(ilpd_data, aes(y=albumin)) + geom_boxplot()
```




```
ggplot(ilpd_data, aes(y=A.G)) + geom_boxplot()
```



Se pueden adivinar posibles outliers o valores extremos. Con un conocimiento del dominio, se podría ver si son susceptibles de quitar o no. Para este análisis los dejaremos por desconocimiento de del dominio.

4. Análisis de los datos.

4.1. Selección de los grupos de datos que se quieren analizar/comparar (planificación de los análisis a aplicar).

4.2. Comprobación de la normalidad y homogeneidad de la varianza.

4.3. Aplicación de pruebas estadísticas para comparar los grupos de datos. En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes.

5. Representación de los resultados a partir de tablas y gráficas.

6. Resolución del problema. A partir de los resultados obtenidos, ¿cuáles son las conclusiones? ¿Los resultados permiten responder al problema?