Repaso examen

Cargamos librerias

```
library(ggplot2)
library(dplyr)
```

```
Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
library(caret)
```

```
Loading required package: lattice
```

```
library(e1071)
library(ggstatsplot)
```

```
You can cite this package as:
     Patil, I. (2021). Visualizations with statistical details: The 'ggstatsplot' approach.
     Journal of Open Source Software, 6(61), 3167, doi:10.21105/joss.03167
```

Cargamos los datos

```
datos <- read.csv("./datos/diabetes.csv")
head(datos)
```

```
  Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI
1           6     148            72            35       0 33.6
2           1      85            66            29       0 26.6
3           8     183            64             0       0 23.3
4           1      89            66            23      94 28.1
5           0     137            40            35     168 43.1
6           5     116            74             0       0 25.6
  DiabetesPedigreeFunction Age Outcome
1                    0.627  50       1
2                    0.351  31       0
3                    0.672  32       1
4                    0.167  21       0
5                    2.288  33       1
6                    0.201  30       0
```

Miramos las clases de los datos

```
str(datos)
```

```
'data.frame':   768 obs. of  9 variables:
 $ Pregnancies             : int  6 1 8 1 0 5 3 10 2 8 ...
 $ Glucose                 : int  148 85 183 89 137 116 78 115 197 125 ...
 $ BloodPressure           : int  72 66 64 66 40 74 50 0 70 96 ...
 $ SkinThickness           : int  35 29 0 23 35 0 32 0 45 0 ...
 $ Insulin                 : int  0 0 0 94 168 0 88 0 543 0 ...
 $ BMI                     : num  33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 ...
 $ DiabetesPedigreeFunction: num  0.627 0.351 0.672 0.167 2.288 ...
 $ Age                     : int  50 31 32 21 33 30 26 29 53 54 ...
 $ Outcome                 : int  1 0 1 0 1 0 1 0 1 1 ...
```

Se cambia únicamen esta variable `Outcome` a factor. Donde 1 es diebetes, y 0 es no diabetes

```
datos$Outcome  <- as.factor(datos$Outcome)
```

Análisis estadístico preliminar

```
dim(datos)
```

```
[1] 768   9
```

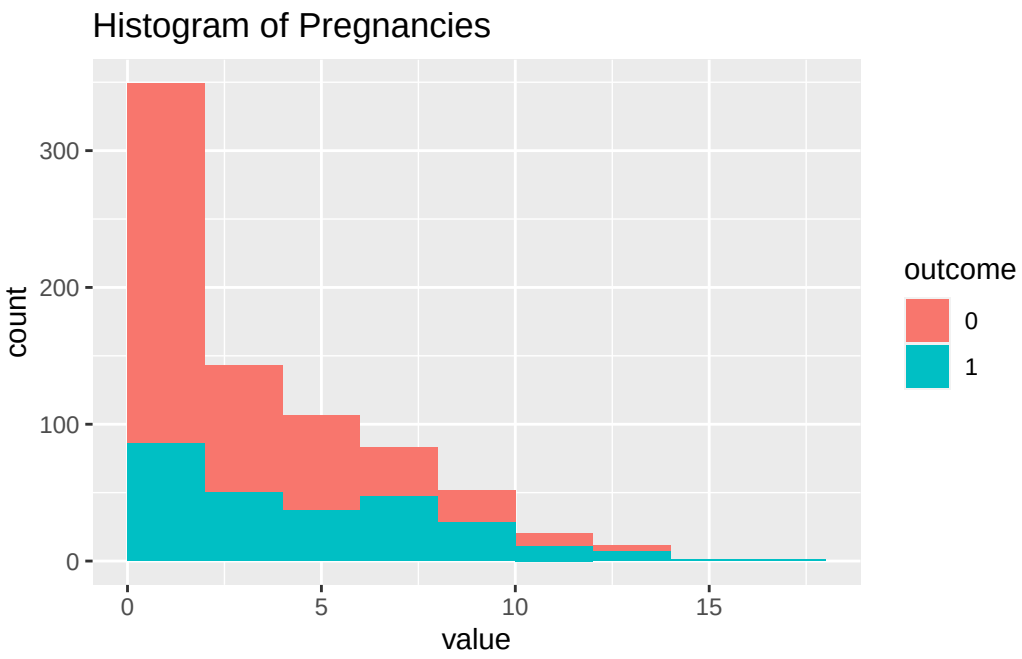
Se analiza primero dos a dos las variables una por una

Histogramas

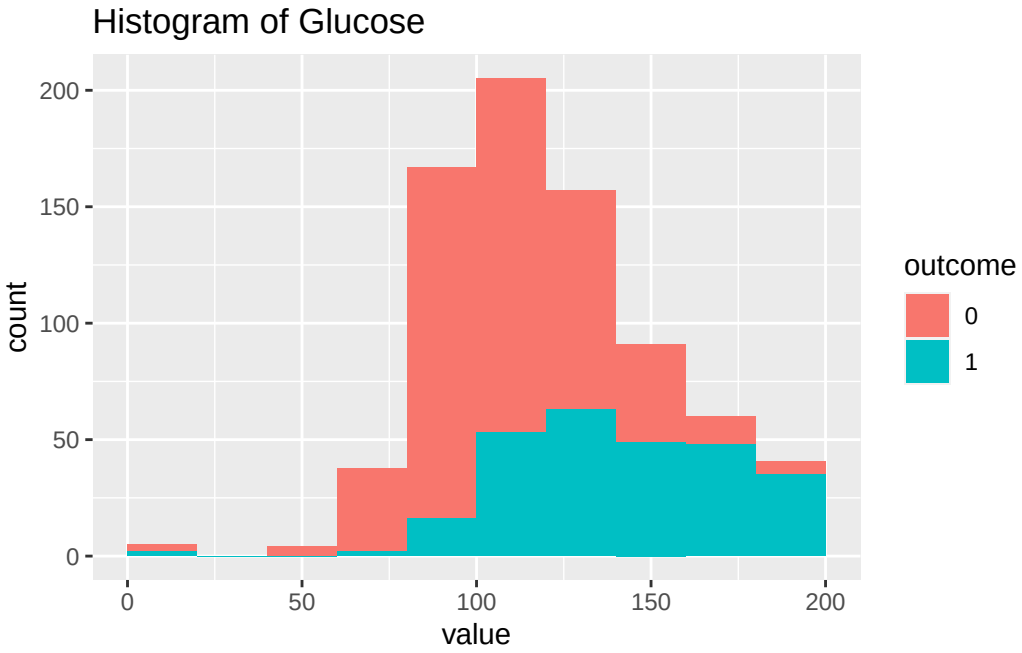
```
l.plots <- vector("list",length = ncol(datos)-1)
n1 <- ncol(datos) -1
for(j in 1:n1){

  h <-hist(datos[,j],plot = F)
  datos.tmp <- data.frame(value=datos[,j],outcome=datos$Outcome)
  p1 <- ggplot(datos.tmp,aes(value,fill=outcome))+geom_histogram(breaks=h$breaks) + ggtitl

  l.plots[[j]] <- p1
}
```

```
l.plots
```

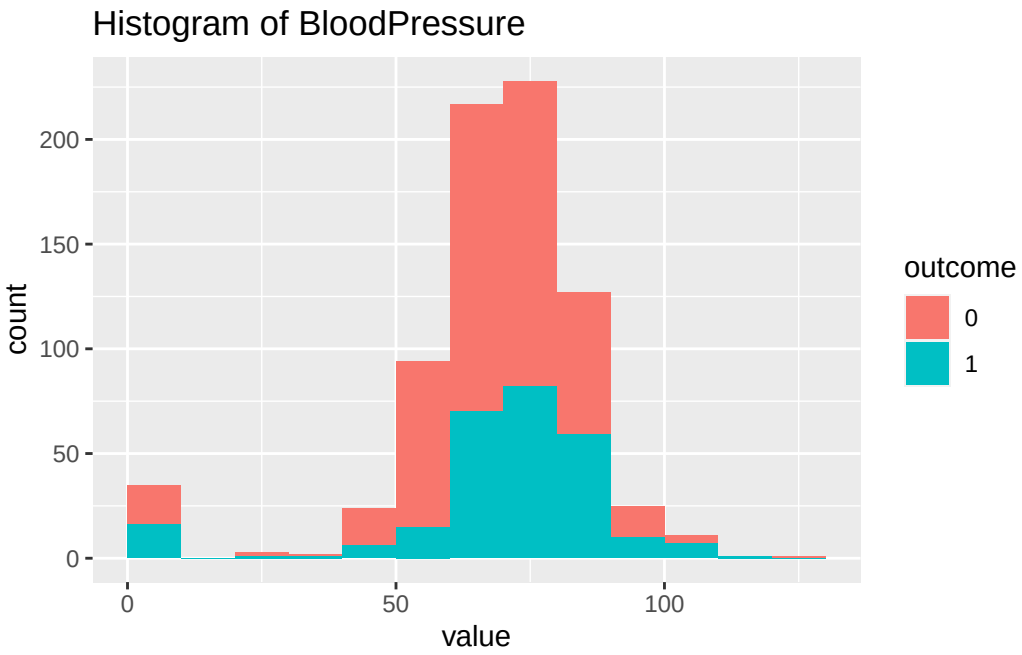
```
[[1]]
```


Histogram of Pregnancies
count
value
outcome
0
1
0
100
200
300
0
5
10
15

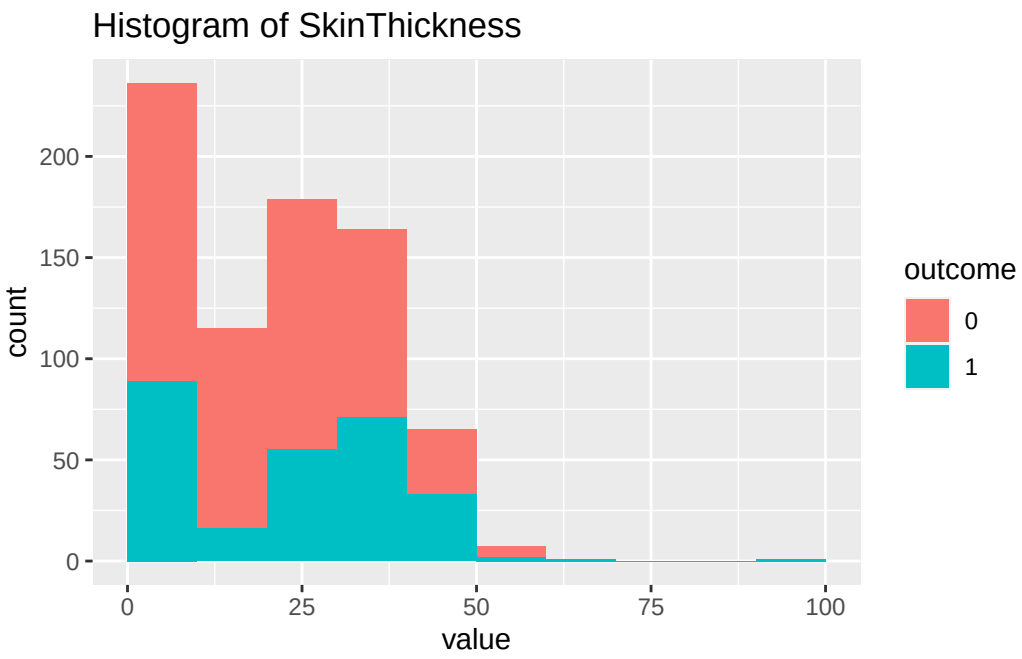
```
[[2]]
```


Histogram of Glucose
count
value
outcome
0
1
0
50
100
150
200
0
50
100
150
200

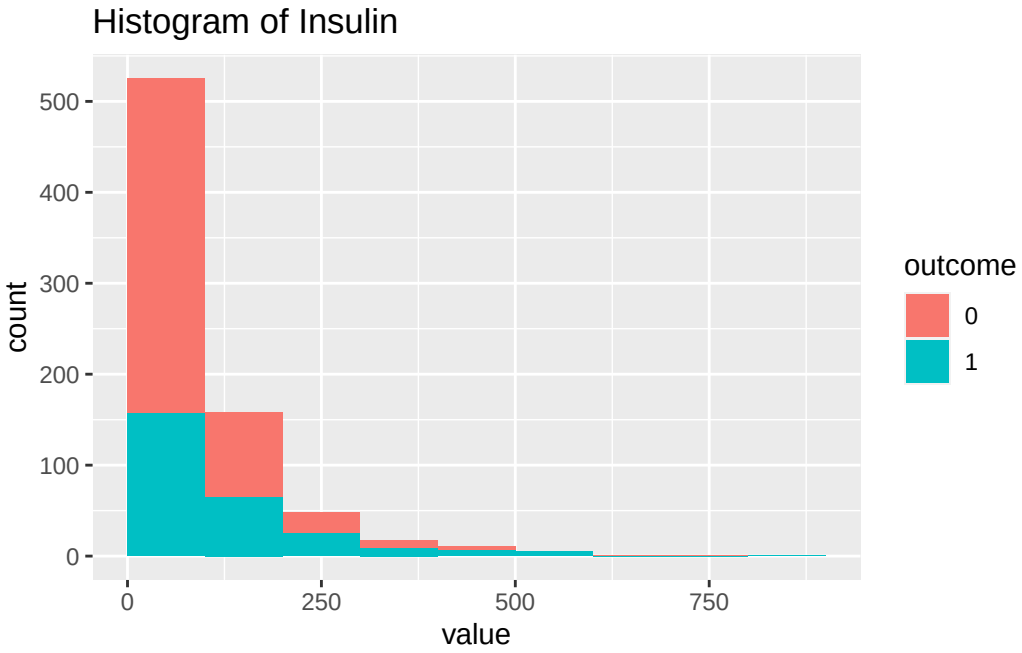
```
[[3]]
```


Histogram of BloodPressure
count
200
150
100
50
0
0
50
100
value
outcome
0
1

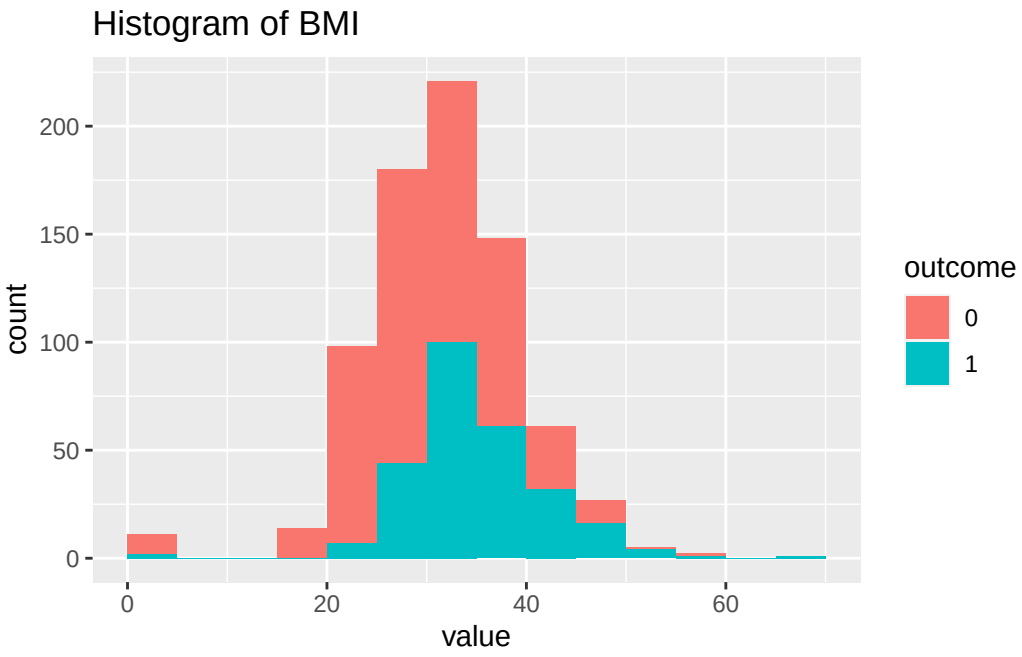
```
[[4]]
```


Histogram of SkinThickness
count
value
outcome
0
1
0
50
100
150
200
0
25
50
75
100

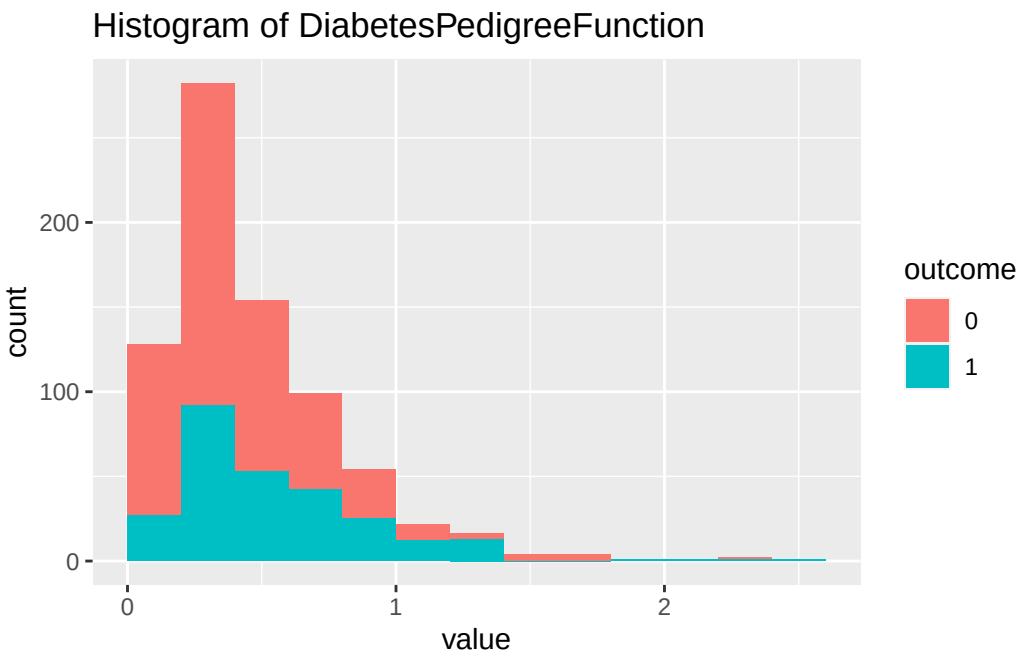
```
[[5]]
```


Histogram of Insulin
count
value
outcome
0
1
0
100
200
300
400
500
0
250
500
750

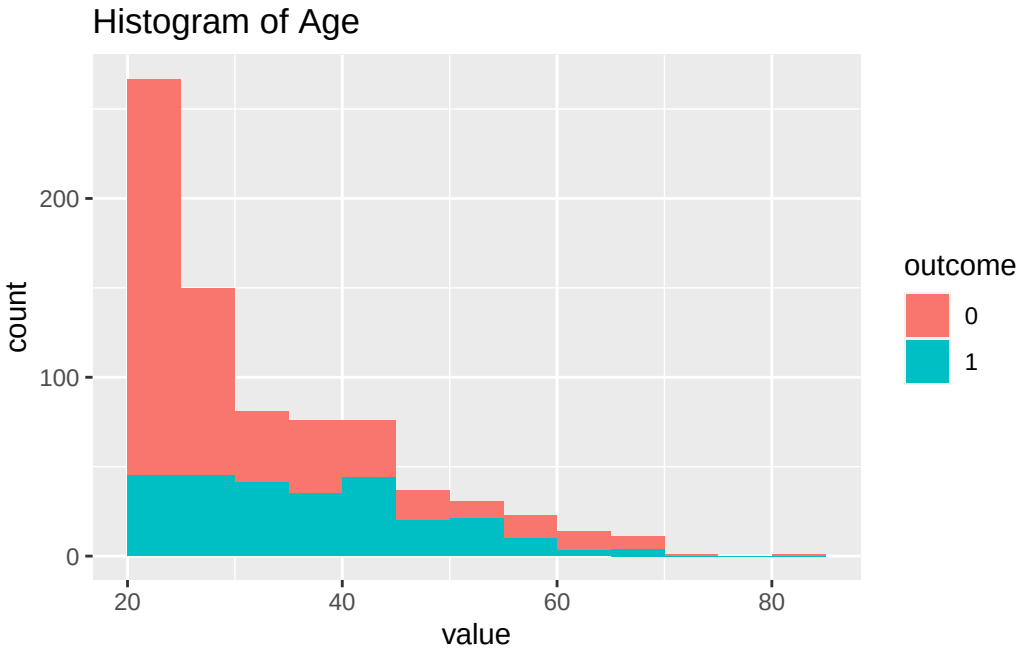
```
[[6]]
```


Histogram of BMI
count
200
150
100
50
0
0
20
40
60
value
outcome
0
1

```
[[7]]
```


Histogram of DiabetesPedigreeFunction
count
200
100
0
0
1
2
value
outcome
0
1

```
[[8]]
```

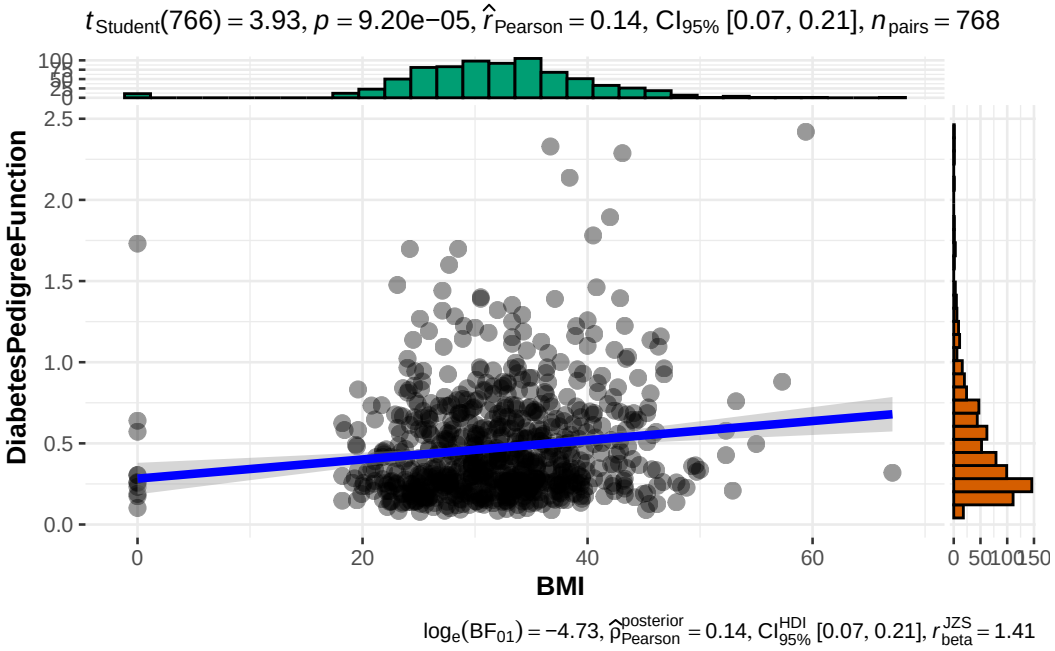

Histogram of Age
count
200
100
0
20
40
60
80
value
outcome
0
1

Se correlaciona todas con todas las variables

```
ggscatterstats(datos,BMI,DiabetesPedigreeFunction)
```

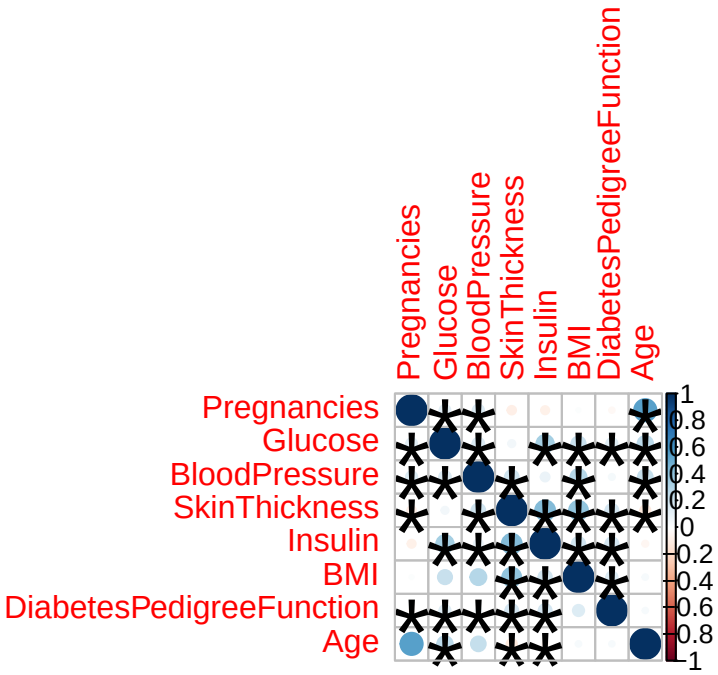
```
Registered S3 method overwritten by 'ggside':
  method from
  +.gg   ggplot2
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Se condensa todo para no hacer demasiadas gráficas

```
obj.cor <- psych::corr.test(datos[,1:n1])
p.values <- obj.cor$p
p.values[upper.tri(p.values)] <- obj.cor$p.adj
p.values[lower.tri(p.values)] <- obj.cor$p.adj
diag(p.values) <- 1
corrplot::corrplot(corr = obj.cor$r,p.mat = p.values,sig.level = 0.05,insig = "label_sig")
```



Ahora se puede continuar realizando un proceso similar, llevando a cabo una serie de comparaciones individuales entre las medias o medianas de cada variable y la variable de interés.

En primer lugar, procedemos a realizar una regresión lineal con la variable numérica como variable dependiente y la variable categórica como predictor. Esto se asemeja a un t-test, pero con el propósito de analizar los residuos y evaluar su normalidad.

```
p.norm <- apply(apply(datos[,1:n1],
                2,
                function(x) summary(lm(x~datos$Outcome))$residuals),
          2,
          shapiro.test)

p.norm
```

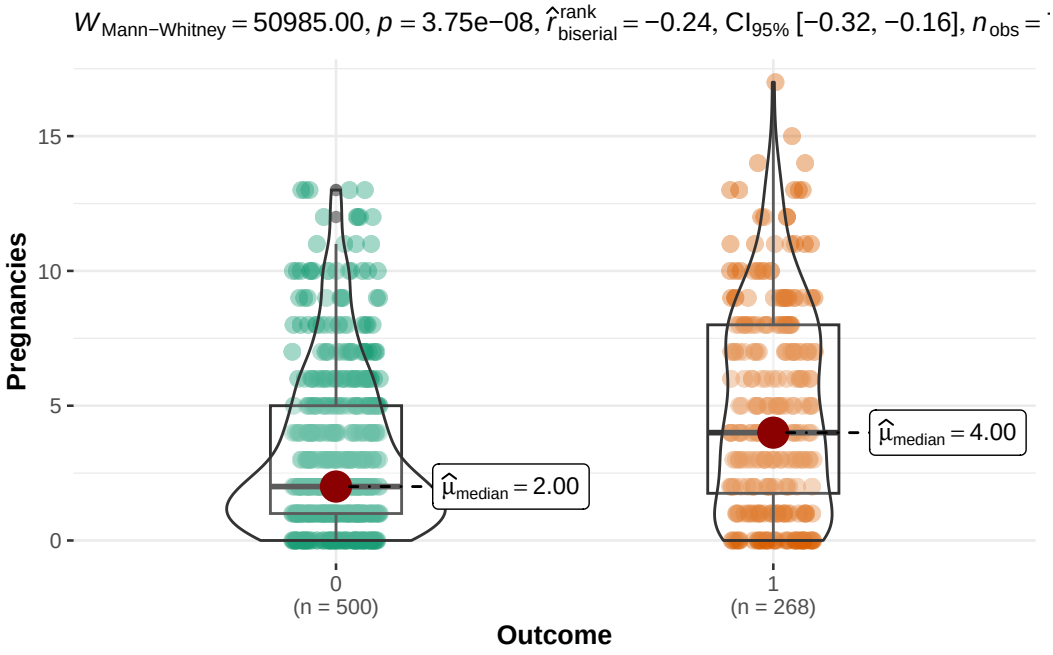
```
$Pregnancies

    Shapiro-Wilk normality test

data:  newX[, i]
W = 0.9389, p-value < 2.2e-16
```

```
$Glucose

    Shapiro-Wilk normality test

data:  newX[, i]
W = 0.97511, p-value = 3.726e-10


$BloodPressure

    Shapiro-Wilk normality test

data:  newX[, i]
W = 0.81468, p-value < 2.2e-16


$SkinThickness

    Shapiro-Wilk normality test

data:  newX[, i]
W = 0.92004, p-value < 2.2e-16


$Insulin

    Shapiro-Wilk normality test

data:  newX[, i]
W = 0.77776, p-value < 2.2e-16


$BMI

    Shapiro-Wilk normality test

data:  newX[, i]
W = 0.94359, p-value < 2.2e-16


$DiabetesPedigreeFunction
```

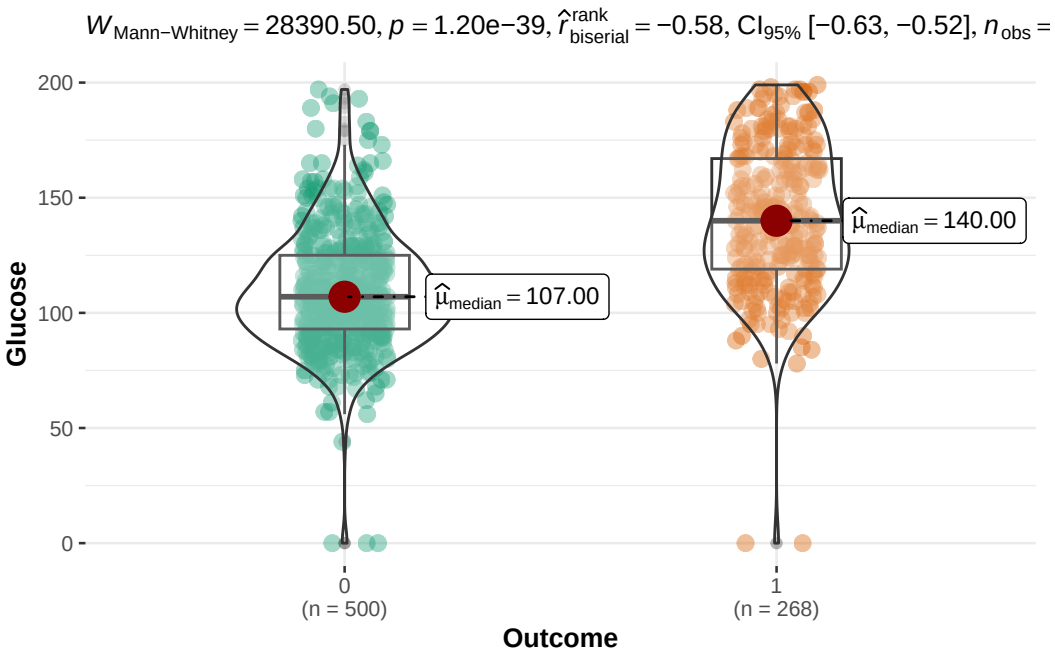
```
    Shapiro-Wilk normality test

data:  newX[, i]
W = 0.84939, p-value < 2.2e-16


$Age

    Shapiro-Wilk normality test

data:  newX[, i]
W = 0.88114, p-value < 2.2e-16
```

Se ve en los histogramas que todas las variables son normales

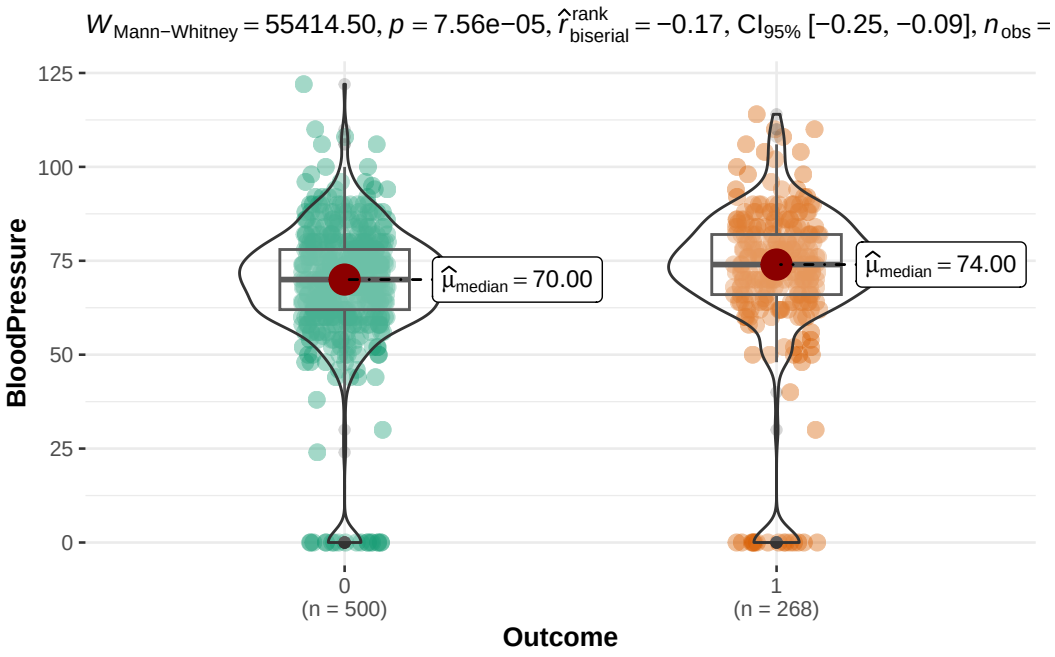
```
ggbetweenstats(datos,Outcome,Pregnancies,type = "nonparametric")
```



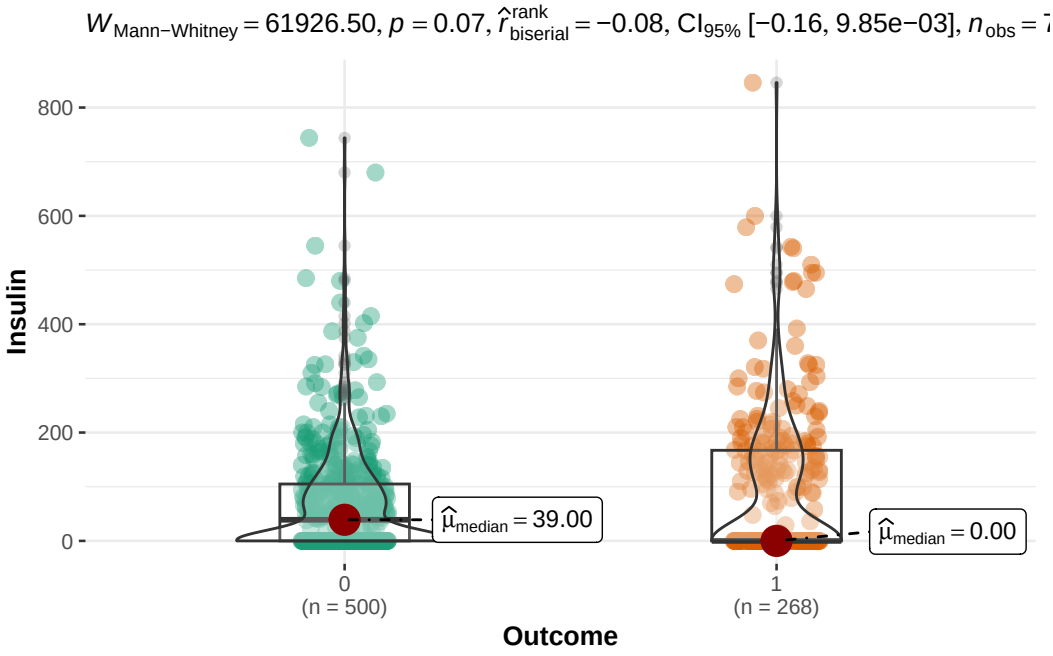

```
ggbetweenstats(datos,Outcome,Glucose,type = "nonparametric")
```



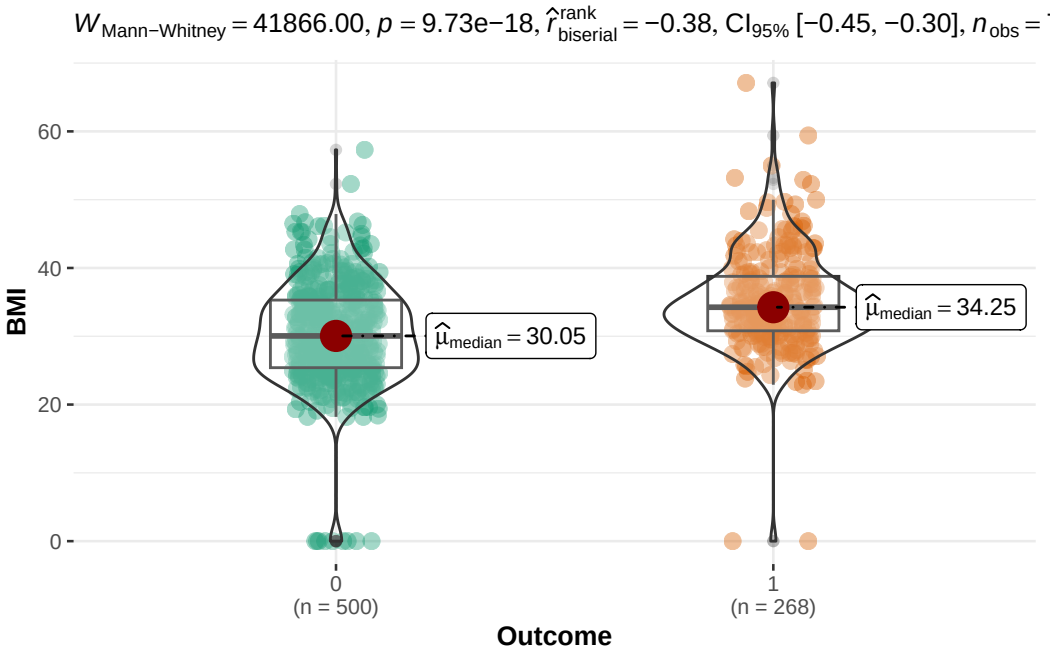

```
ggbetweenstats(datos,Outcome,BloodPressure,type = "nonparametric")
```



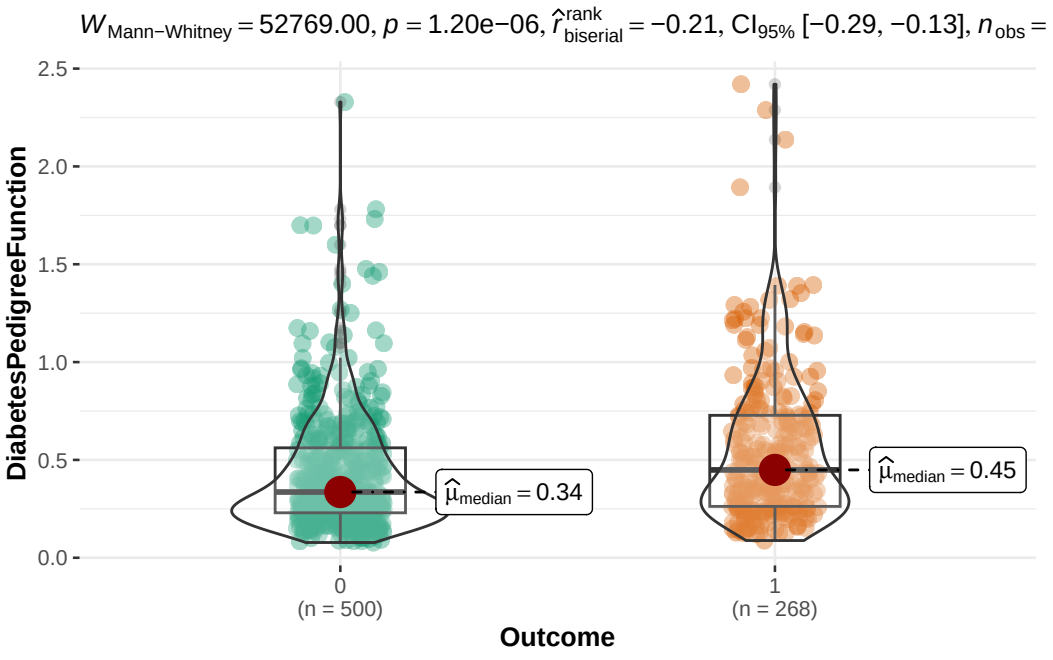
```
ggbetweenstats(datos,Outcome,Insulin,type = "nonparametric")
```



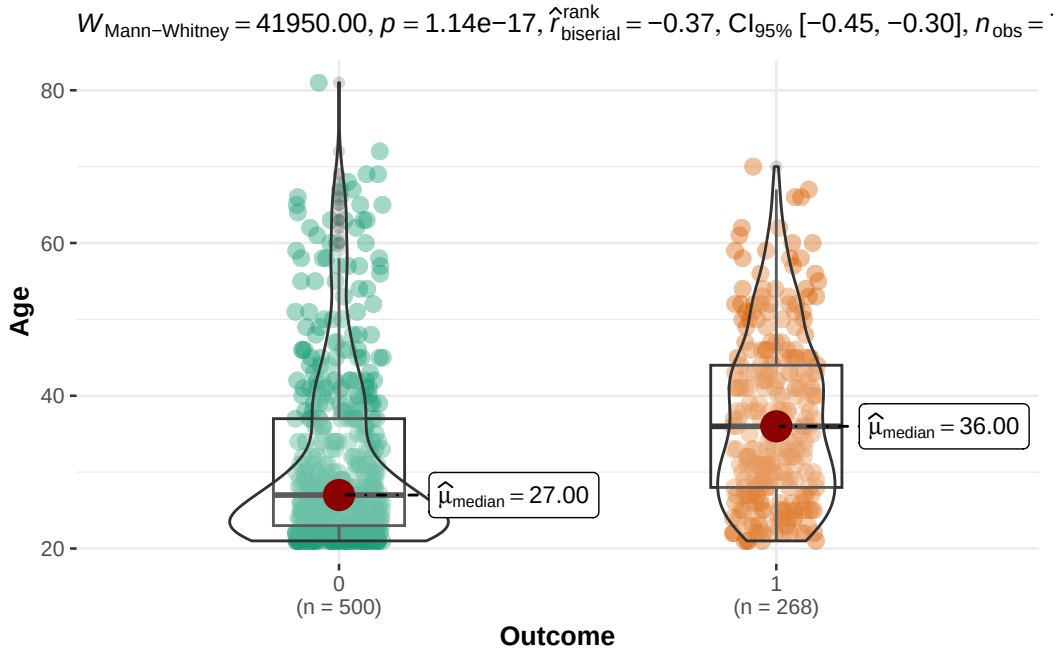

```
ggbetweenstats(datos,Outcome,BMI,type = "nonparametric")
```




```
ggbetweenstats(datos,Outcome,DiabetesPedigreeFunction,type = "nonparametric")
```



```
ggbetweenstats(datos,Outcome,Age,type = "nonparametric")
```



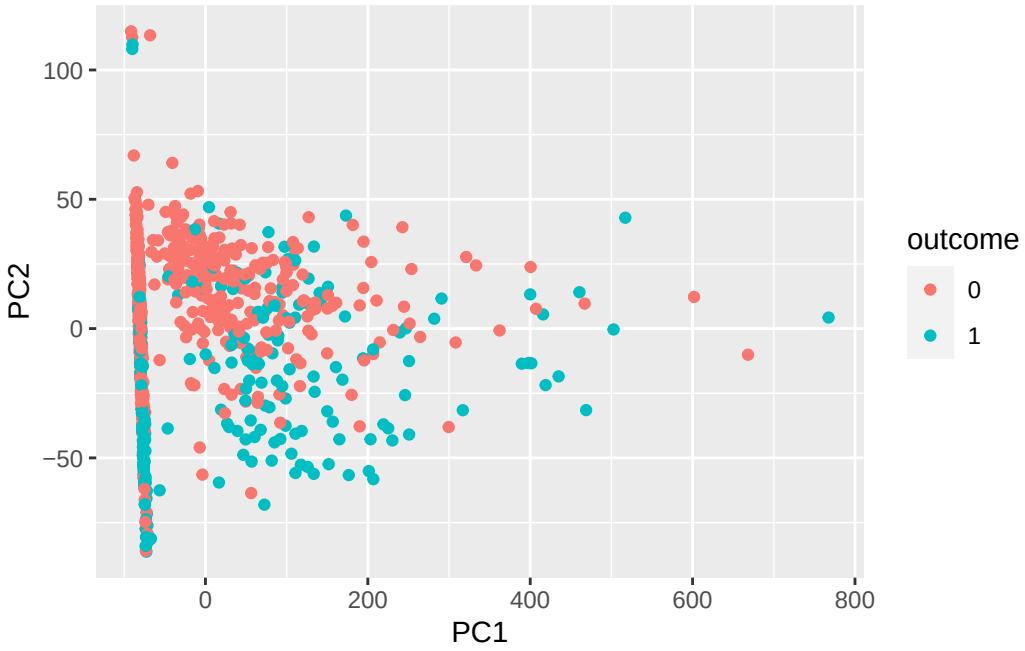
PCA

```
summary(datos)
```

```
  Pregnancies        Glucose      BloodPressure    SkinThickness
Min.   : 0.000   Min.   :  0.0   Min.   :  0.00   Min.   : 0.00
1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 62.00   1st Qu.: 0.00
Median : 3.000   Median :117.0   Median : 72.00   Median :23.00
Mean   : 3.845   Mean   :120.9   Mean   : 69.11   Mean   :20.54
3rd Qu.: 6.000   3rd Qu.:140.2   3rd Qu.: 80.00   3rd Qu.:32.00
Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00
    Insulin          BMI        DiabetesPedigreeFunction      Age
Min.   :  0.0   Min.   : 0.00   Min.   :0.0780           Min.   :21.00
1st Qu.:  0.0   1st Qu.:27.30   1st Qu.:0.2437           1st Qu.:24.00
Median : 30.5   Median :32.00   Median :0.3725           Median :29.00
Mean   : 79.8   Mean   :31.99   Mean   :0.4719           Mean   :33.24
3rd Qu.:127.2   3rd Qu.:36.60   3rd Qu.:0.6262           3rd Qu.:41.00
Max.   :846.0   Max.   :67.10   Max.   :2.4200           Max.   :81.00
```

```
Outcome
0:500
1:268
```

```
pcx <- prcomp(datos[,1:n1],scale. = F) ## escalamos por la variablidad de los datos

plotpca <- bind_cols(pcx$x,outcome=datos$Outcome)
ggplot(plotpca,aes(PC1,PC2,color=outcome))+geom_point()
```



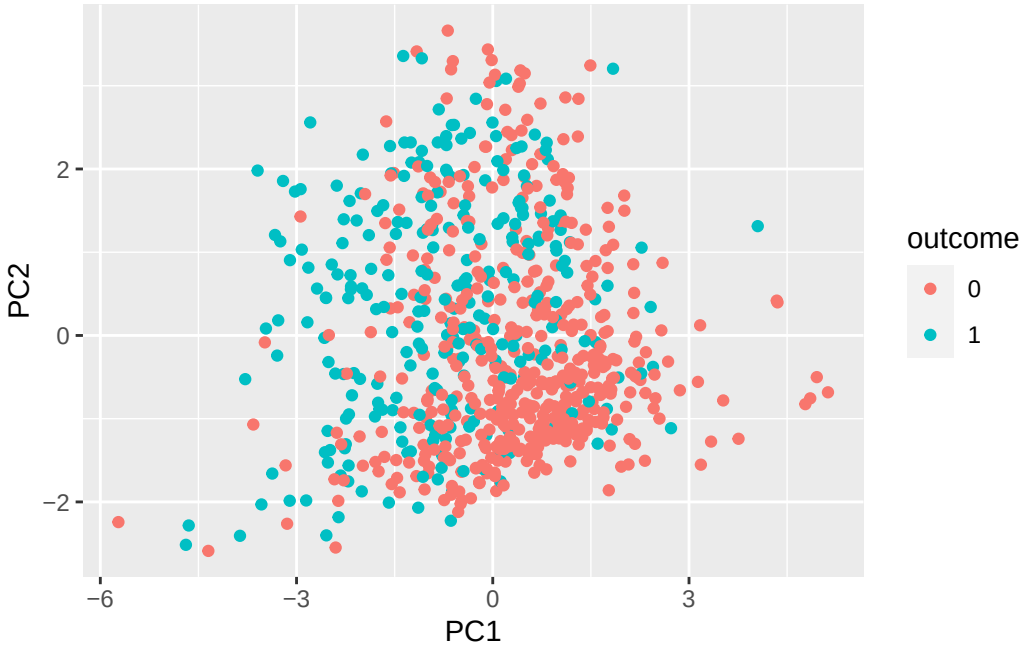
Ahora exploraremos si realizar algunas transformaciones puede afectar los resultados. Sin embargo, antes de hacerlo, debemos examinar las variables que nos generan sospechas.

Además, también podemos considerar la posibilidad de aplicar una escala para ver si se produce algún cambio en los resultados.

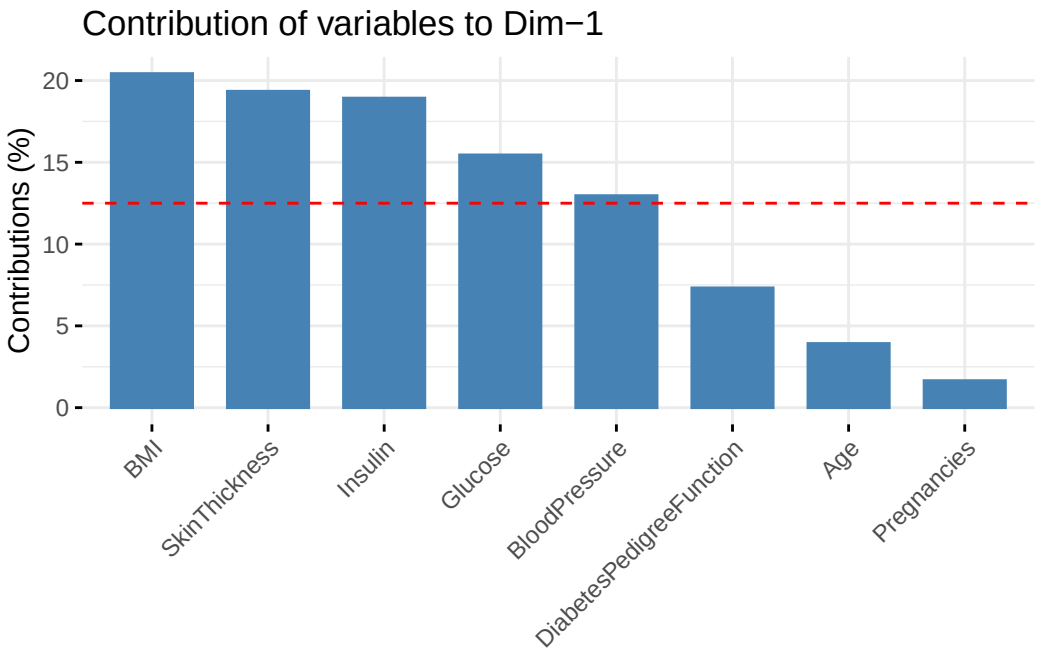
```
summary(datos)
```

```
 Pregnancies        Glucose      BloodPressure    SkinThickness
Min.   : 0.000   Min.   :  0.0   Min.   :  0.00   Min.   : 0.00
1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 62.00   1st Qu.: 0.00
Median : 3.000   Median :117.0   Median : 72.00   Median :23.00
Mean   : 3.845   Mean   :120.9   Mean   : 69.11   Mean   :20.54
3rd Qu.: 6.000   3rd Qu.:140.2   3rd Qu.: 80.00   3rd Qu.:32.00
Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00
   Insulin           BMI        DiabetesPedigreeFunction      Age
Min.   :  0.0   Min.   : 0.00   Min.   :0.0780           Min.   :21.00
1st Qu.:  0.0   1st Qu.:27.30   1st Qu.:0.2437           1st Qu.:24.00
Median : 30.5   Median :32.00   Median :0.3725           Median :29.00
Mean   : 79.8   Mean   :31.99   Mean   :0.4719           Mean   :33.24
3rd Qu.:127.2   3rd Qu.:36.60   3rd Qu.:0.6262           3rd Qu.:41.00
Max.   :846.0   Max.   :67.10   Max.   :2.4200           Max.   :81.00
Outcome
0:500
1:268
```

```r
pcx <- prcomp(datos[,1:n1],scale. = T) ## escalamos por la variablidad de los datos

plotpca <- bind_cols(pcx$x,outcome=datos$Outcome)
ggplot(plotpca,aes(PC1,PC2,color=outcome))+geom_point()
```

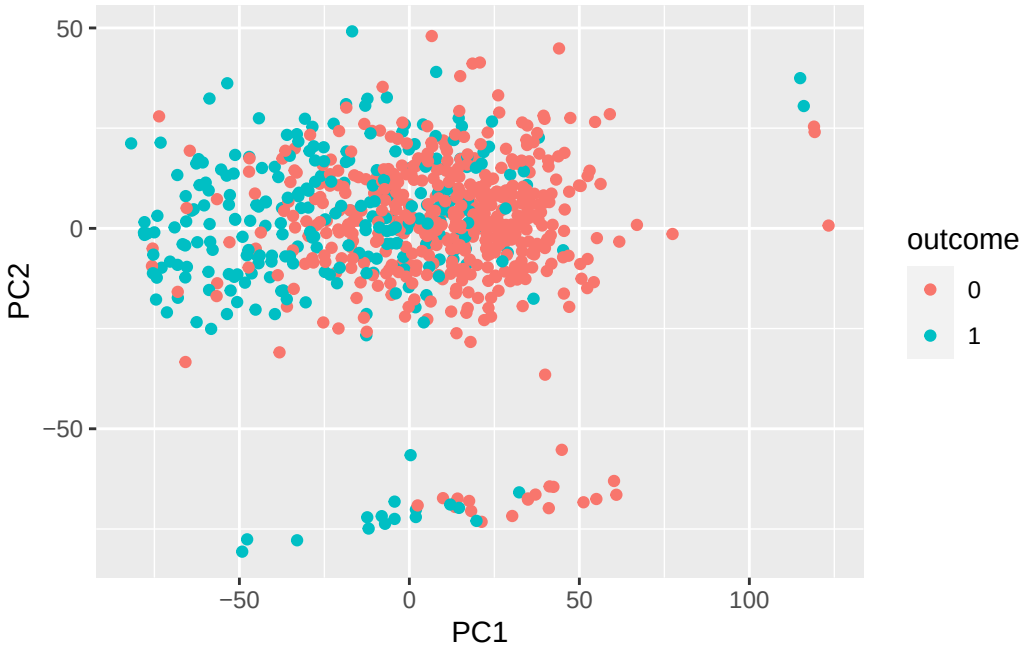



```
factoextra::fviz_contrib(pcx,"var")
```



La insulina da problemas

```
## indices a quitar
w <- c(grep("insulin",ignore.case = T,colnames(datos)),ncol(datos))
pcx <- prcomp(datos[,-w],scale. = F) ## escalamos por la variablidad de los datos

plotpca <- bind_cols(pcx$x,outcome=datos$Outcome)
ggplot(plotpca,aes(PC1,PC2,color=outcome))+geom_point()
```



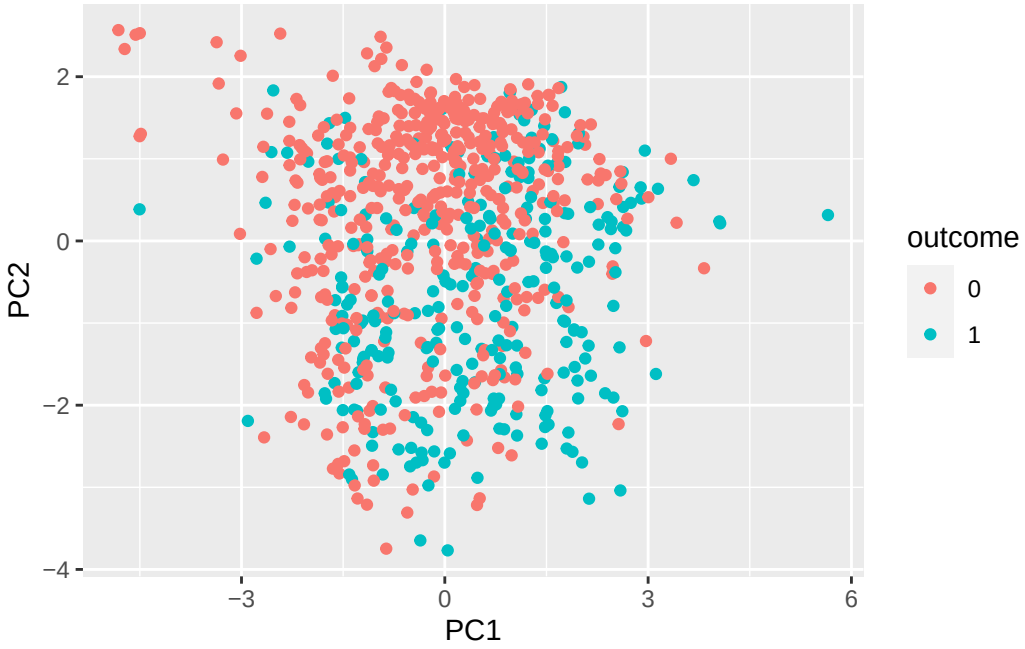
Se transforma la variable de la insulina

```
datos$Insulin  <- log(datos$Insulin+0.05)

summary(datos)
```

```
  Pregnancies        Glucose      BloodPressure    SkinThickness
Min.   : 0.000   Min.   :  0.0   Min.   :  0.00   Min.   : 0.00
1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 62.00   1st Qu.: 0.00
Median : 3.000   Median :117.0   Median : 72.00   Median :23.00
Mean   : 3.845   Mean   :120.9   Mean   : 69.11   Mean   :20.54
3rd Qu.: 6.000   3rd Qu.:140.2   3rd Qu.: 80.00   3rd Qu.:32.00
```

```
Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00
   Insulin             BMI        DiabetesPedigreeFunction      Age
Min.   :-2.996   Min.   : 0.00   Min.   :0.0780           Min.   :21.00
1st Qu.:-2.996   1st Qu.:27.30   1st Qu.:0.2437           1st Qu.:24.00
Median : 3.418   Median :32.00   Median :0.3725           Median :29.00
Mean   : 1.008   Mean   :31.99   Mean   :0.4719           Mean   :33.24
3rd Qu.: 4.847   3rd Qu.:36.60   3rd Qu.:0.6262           3rd Qu.:41.00
Max.   : 6.741   Max.   :67.10   Max.   :2.4200           Max.   :81.00
Outcome
0:500
1:268
```

```r
pcx <- prcomp(datos[,1:n1],scale. = T) ## escalamos por la variablidad de los datos

plotpca <- bind_cols(pcx$x,outcome=datos$Outcome)
ggplot(plotpca,aes(PC1,PC2,color=outcome))+geom_point()
```



Hay un cambio notorio, esto indica que no hemos eliminado la información de la insulina, sino que simplemente la hemos transformado.

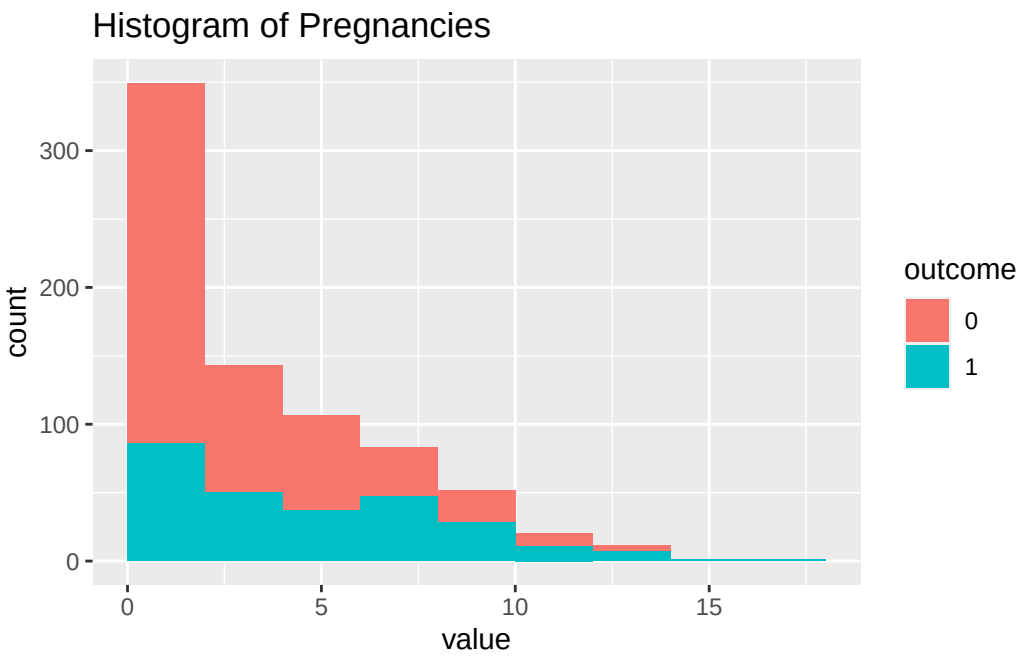
En otras palabras, si transformamos los datos, se produce un cambio. A partir de esto, podemos llevar a cabo pruebas de diferencia de medianas nuevamente, pero esta vez veremos los resultados de forma resumida.

```
datos <- read.csv("./datos/diabetes.csv")
datos$Outcome <- as.factor(datos$Outcome)
datsc <- scale(datos[,-ncol(datos)])
```

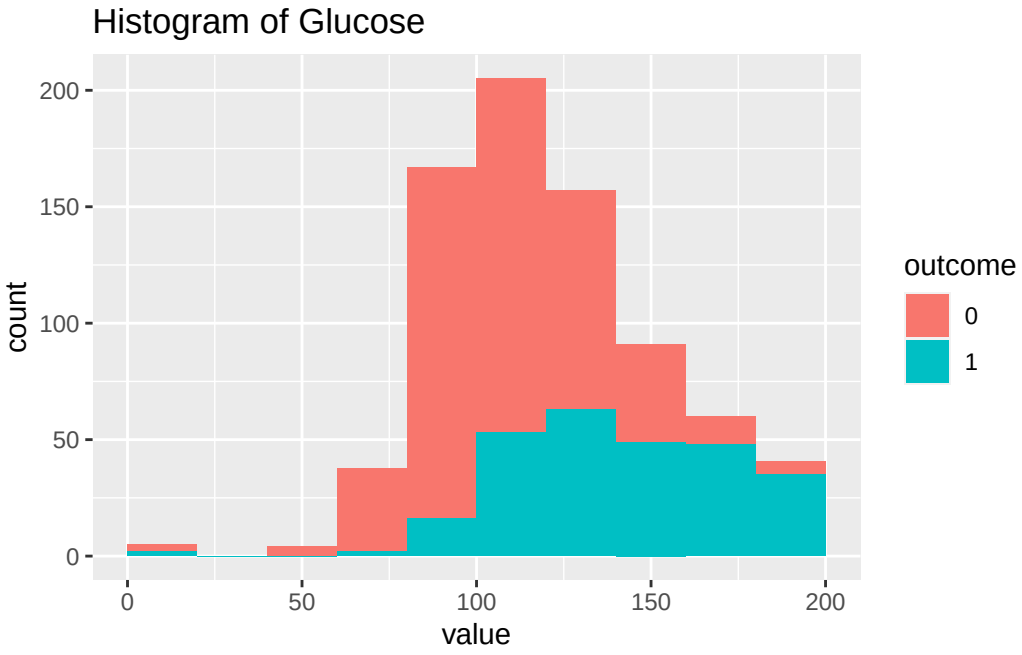
Distribuciones de nuevo

```
l.plots <- vector("list",length = ncol(datos)-1)
n1 <- ncol(datos) -1
for(j in 1:n1){

  h <-hist(datos[,j],plot = F)
  datos.tmp <- data.frame(value=datos[,j],outcome=datos$Outcome)
  p1 <- ggplot(datos.tmp,aes(value,fill=outcome))+geom_histogram(breaks=h$breaks) + ggtitl

  l.plots[[j]] <- p1
}
l.plots
```

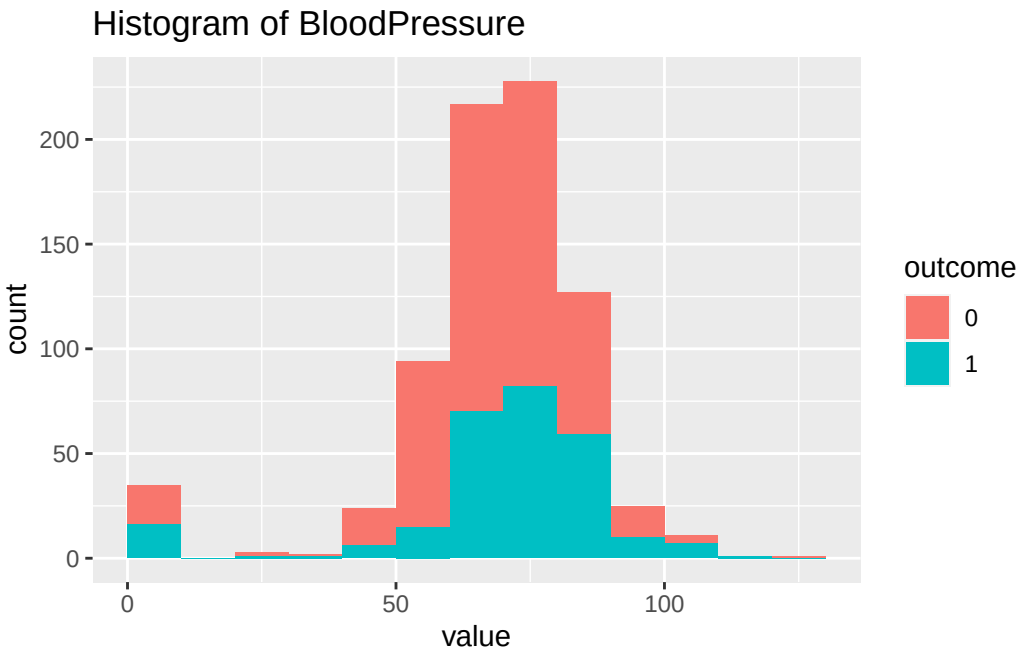
```
[[1]]
```


Histogram of Pregnancies
count
value
outcome
0
1
0
100
200
300
0
5
10
15

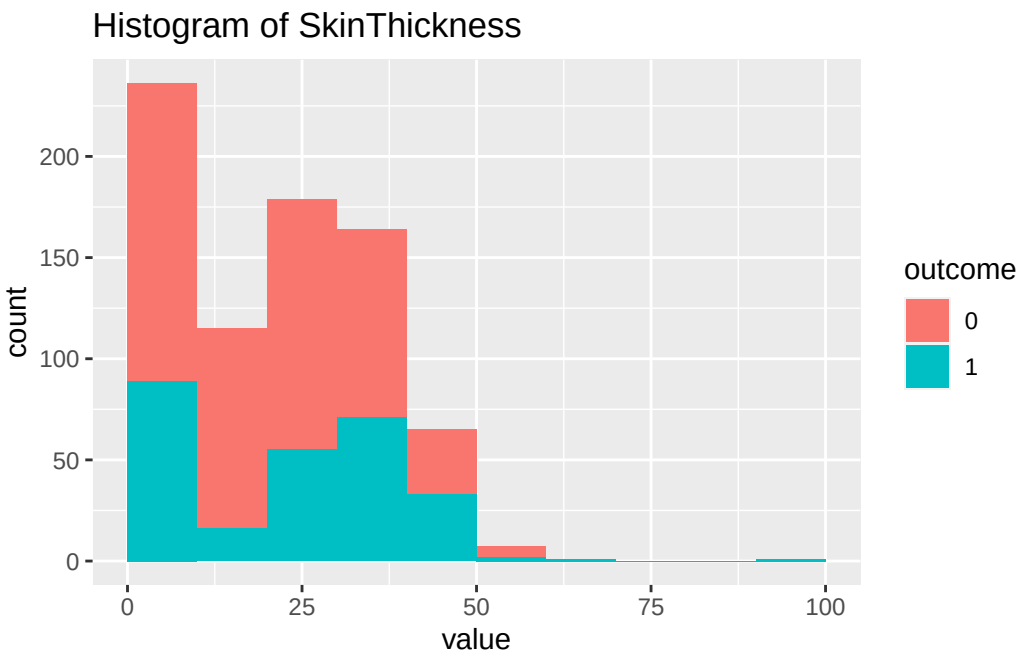
```
[[2]]
```


Histogram of Glucose
count
value
outcome
0
1
0
50
100
150
200
0
50
100
150
200

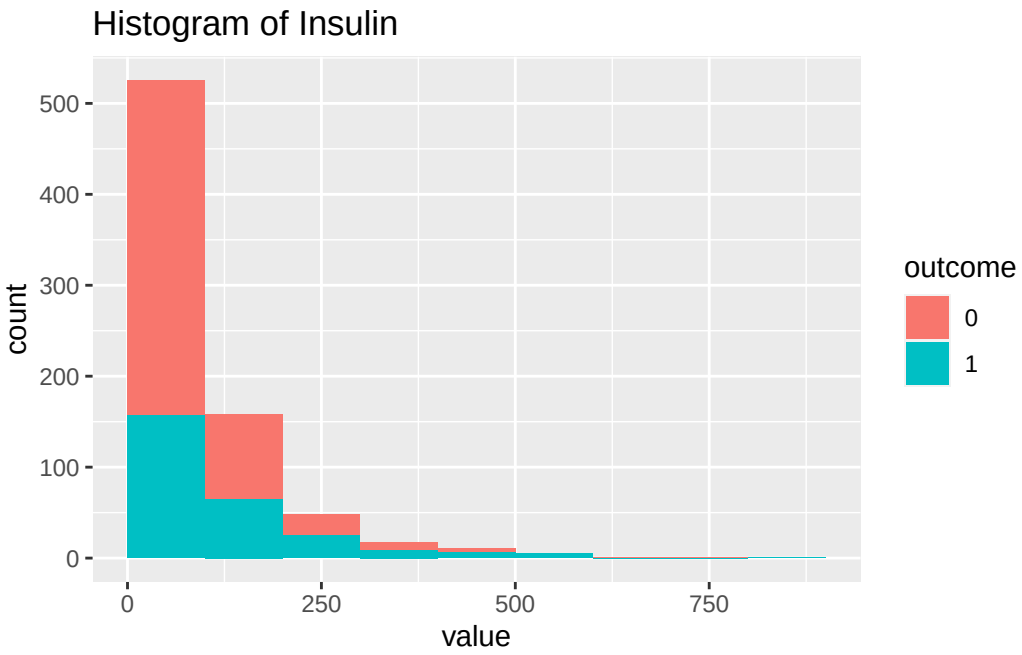
```
[[3]]
```


Histogram of BloodPressure
count
200
150
100
50
0
0
50
100
value
outcome
0
1

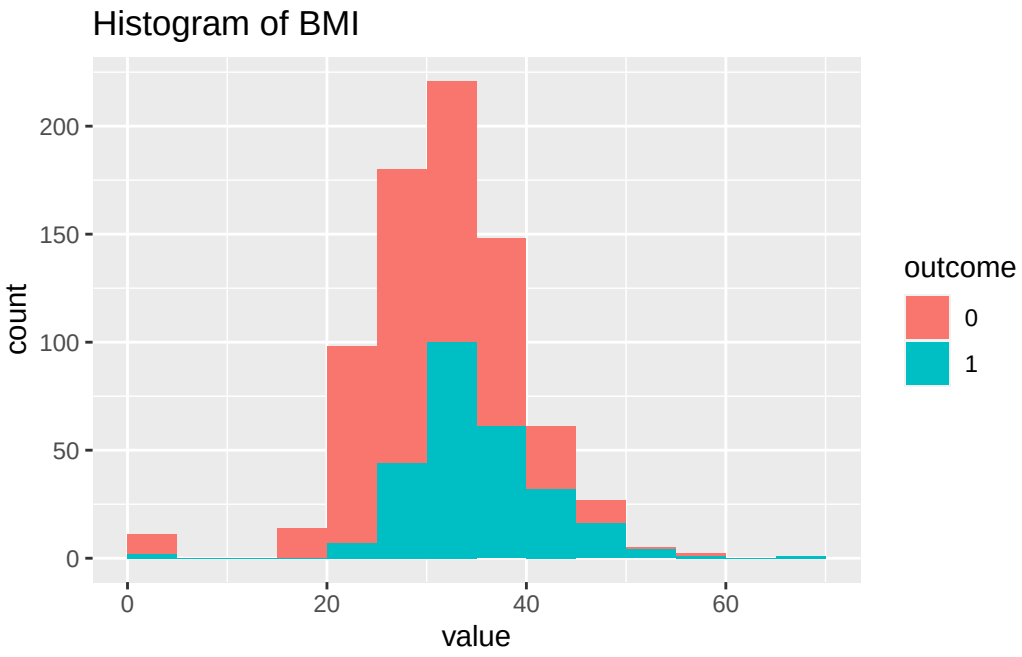
```
[[4]]
```


Histogram of SkinThickness
count
value
outcome
0
1
0
50
100
150
200
0
25
50
75
100

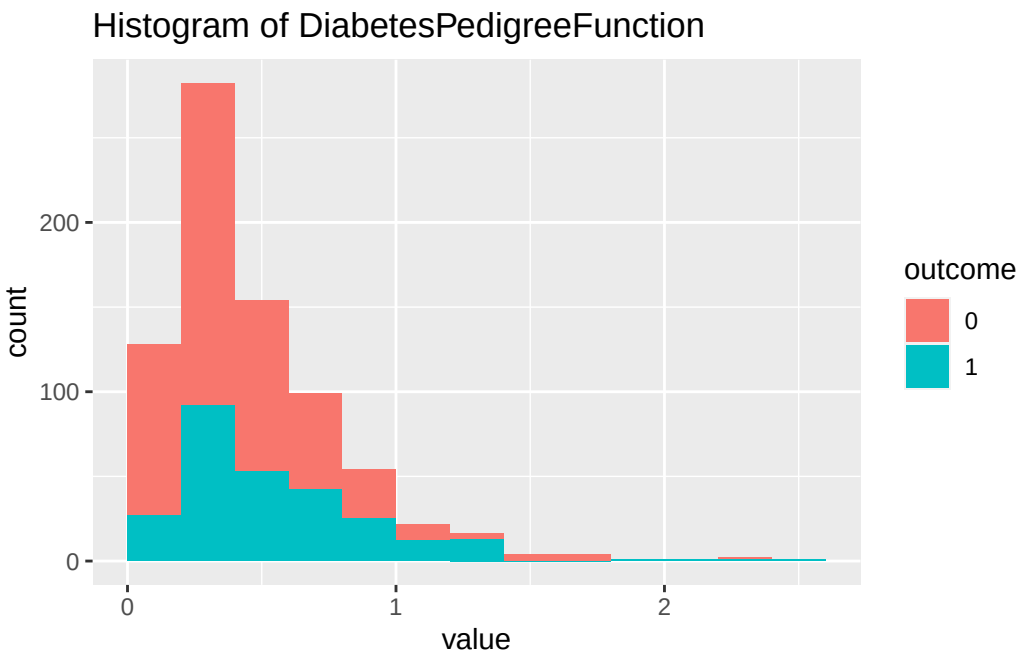
```
[[5]]
```


Histogram of Insulin
count
value
outcome
0
1
0
100
200
300
400
500
0
250
500
750

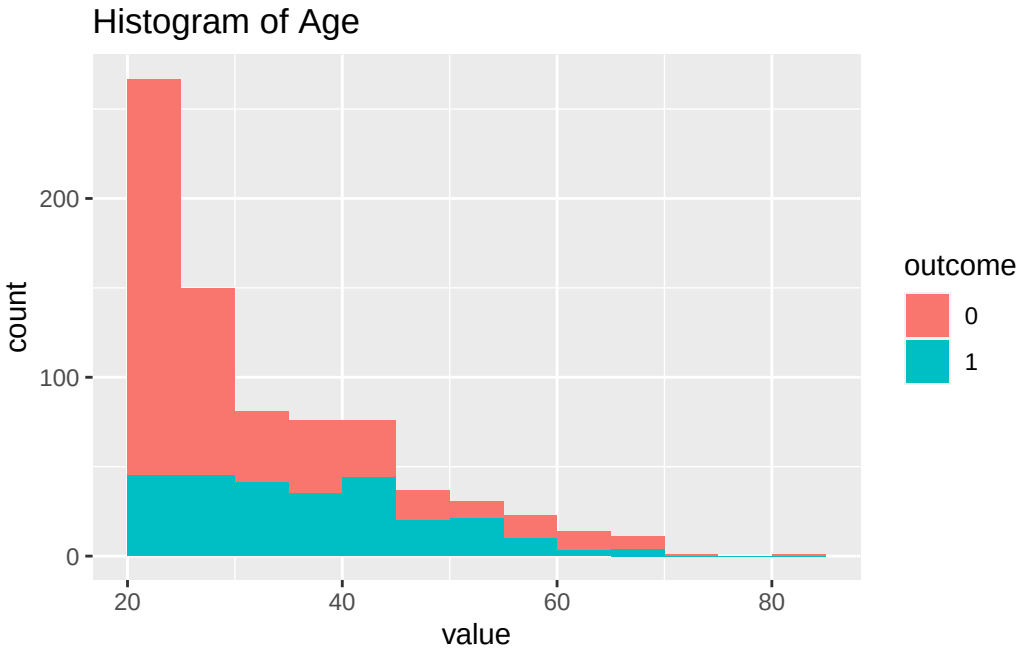
```
[[6]]
```


Histogram of BMI
count
200
150
100
50
0
0
20
40
60
value
outcome
0
1

```
[[7]]
```


Histogram of DiabetesPedigreeFunction
count
200
100
0
0
1
2
value
outcome
0
1

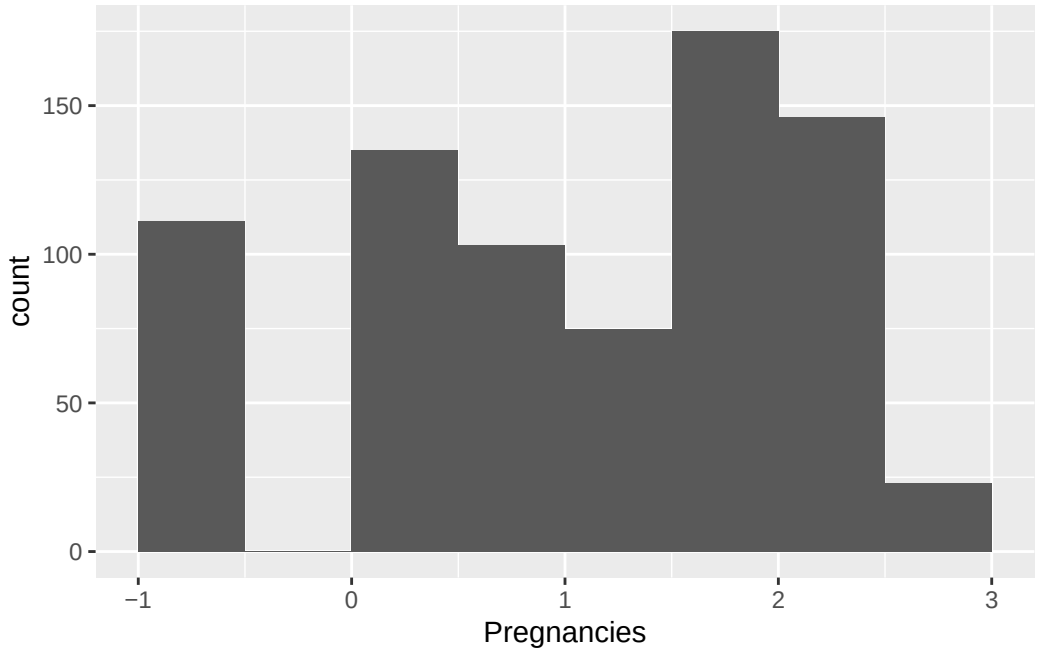
```
[[8]]
```


Histogram of Age
count
200
100
0
20
40
60
80
value
outcome
0
1

Es interesante observar que los valores de la insulina han cambiado debido a la transformación en valor, pero no ha cambiado la distribución. Ahora procederemos a realizar algunos ajustes adicionales.

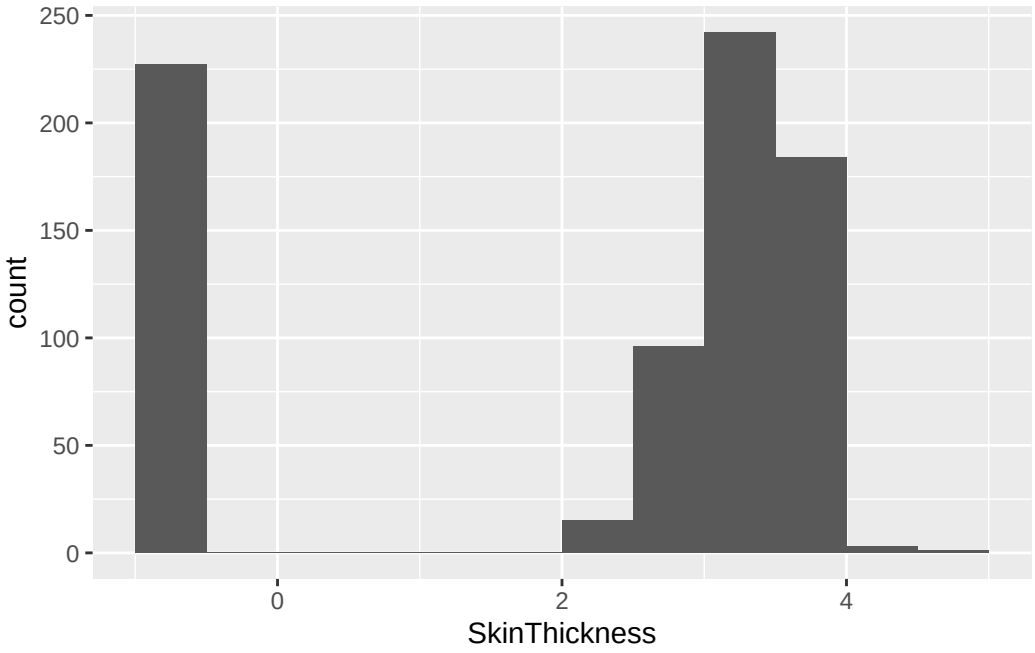
Además, parece que la variable de preñanza está relacionada con una escala logarítmica de base 2. Esto es algo diferente y merece una atención especial.

```
datos <- read.csv("./datos/diabetes.csv")
datos$Outcome <- as.factor(datos$Outcome)
datos$Pregnancies  <- log(datos$Pregnancies+0.5)
ggplot(datos,aes(Pregnancies))+geom_histogram(breaks = hist(datos$Pregnancies,plot=F)$brea
```



Se realiza lo mismo con la grosura de la piel

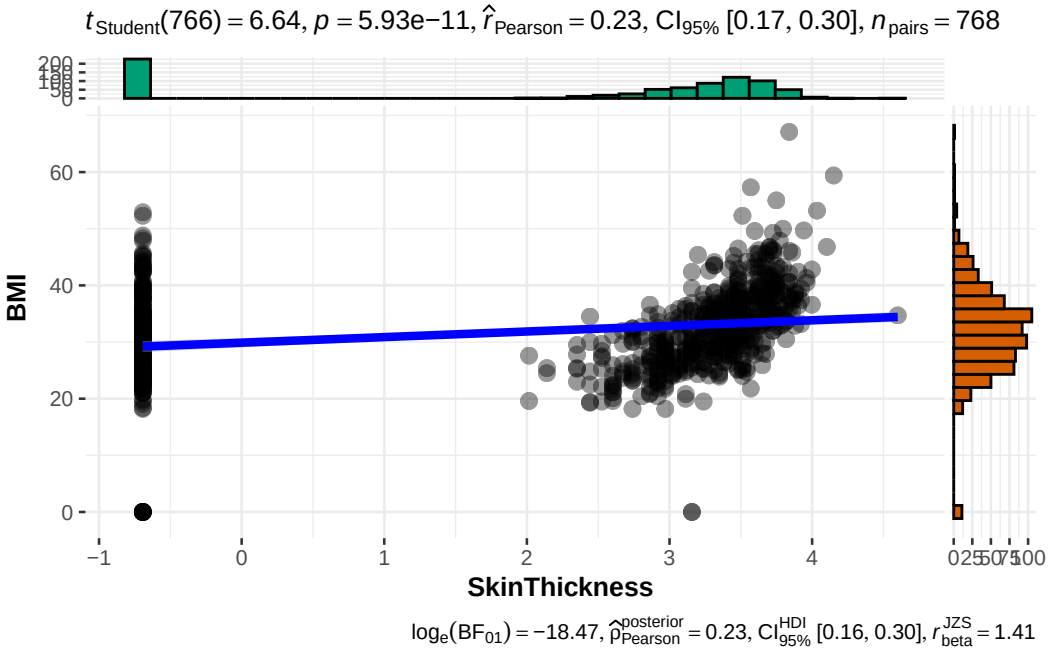
```
datos <- read.csv("./datos/diabetes.csv")
datos$Outcome <- as.factor(datos$Outcome)
datos$SkinThickness  <- log(datos$SkinThickness+0.5)
ggplot(datos,aes(SkinThickness))+geom_histogram(breaks = hist(datos$SkinThickness,plot=F)$
```



Lo raro está dado por lo obesidad

```
ggscatterstats(datos,SkinThickness,BMI)
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Al parecer, los datos contienen valores nulos que se encuentran únicamente en las variables distintas a "pregnancies". Procederemos a eliminar esos valores nulos para continuar con el análisis.

```
datos <- read.csv("./datos/diabetes.csv")
datos[,-c(1,9)] <- apply(datos[,-c(1,9)],2,function(x) ifelse(x==0,NA,x))

datos$Outcome <- as.factor(datos$Outcome)
```

Quitamos estos valores

```
datos <- datos[complete.cases(datos),]
```

El data set está reducido a 392 observaciones

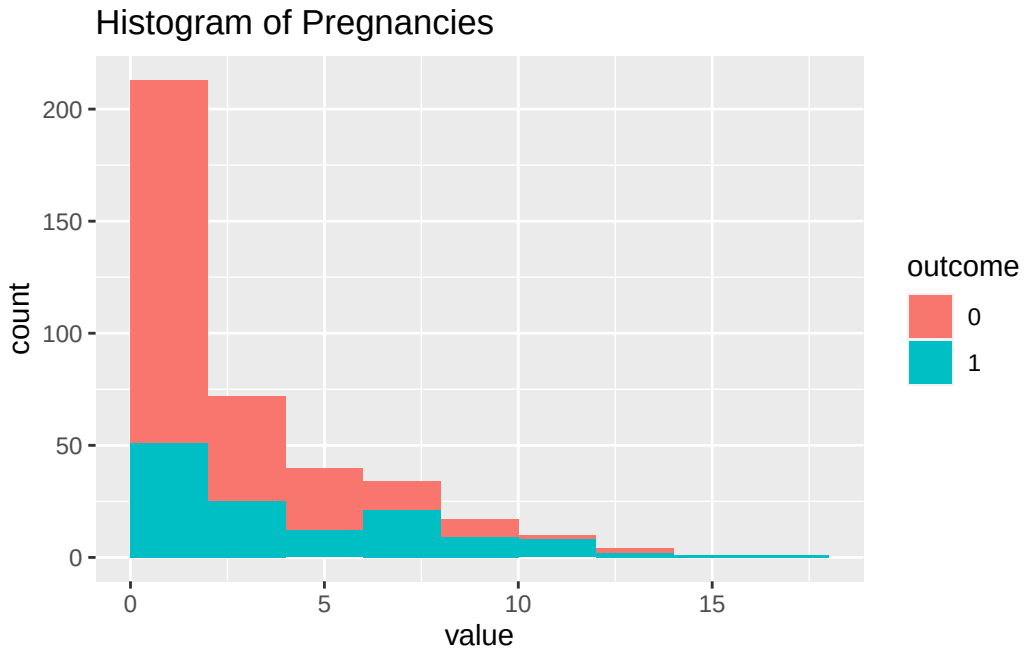
```
table(datos$Outcome)
```

```
0   1
```

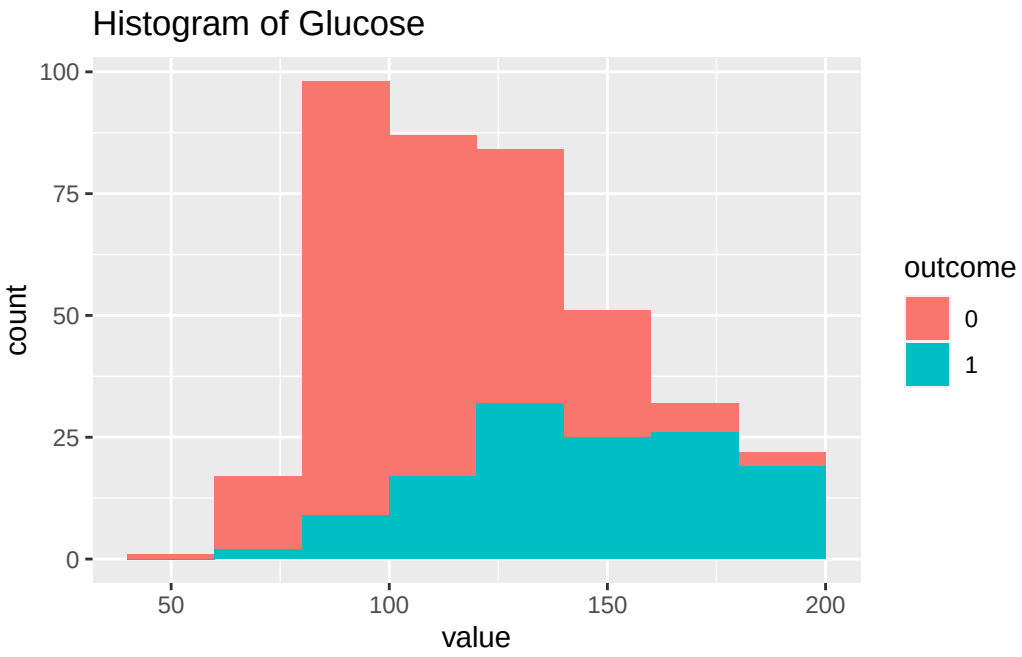
```
262 130
```

```
l.plots <- vector("list",length = ncol(datos)-1)
n1 <- ncol(datos) -1
for(j in 1:n1){

  h <-hist(datos[,j],plot = F)
  datos.tmp <- data.frame(value=datos[,j],outcome=datos$Outcome)
  p1 <- ggplot(datos.tmp,aes(value,fill=outcome))+geom_histogram(breaks=h$breaks) + ggtitl

  l.plots[[j]] <- p1
}
l.plots
```

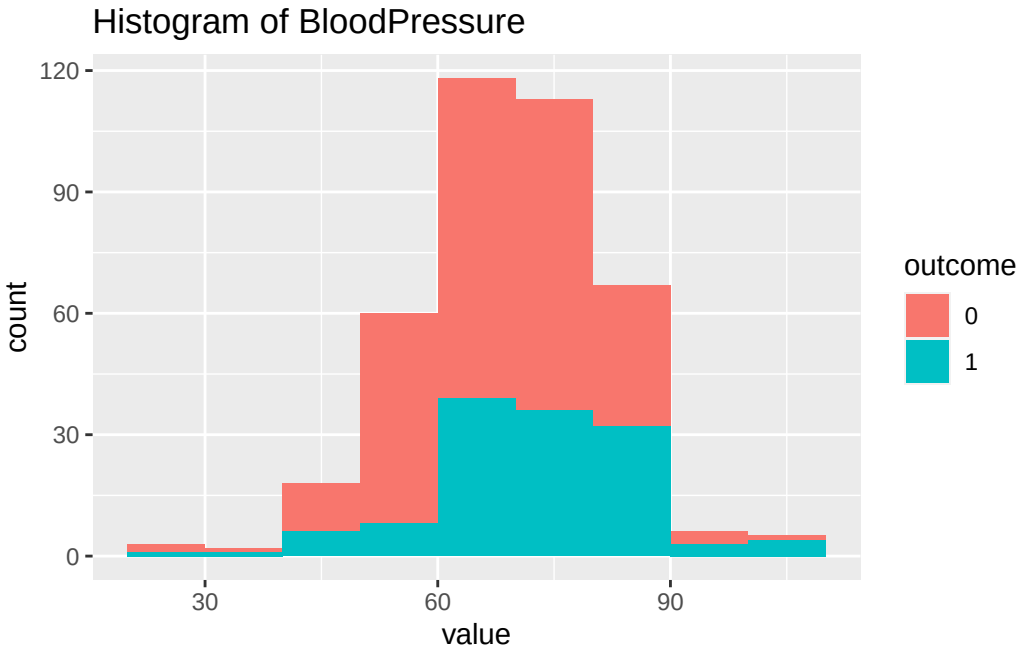
```
[[1]]
```



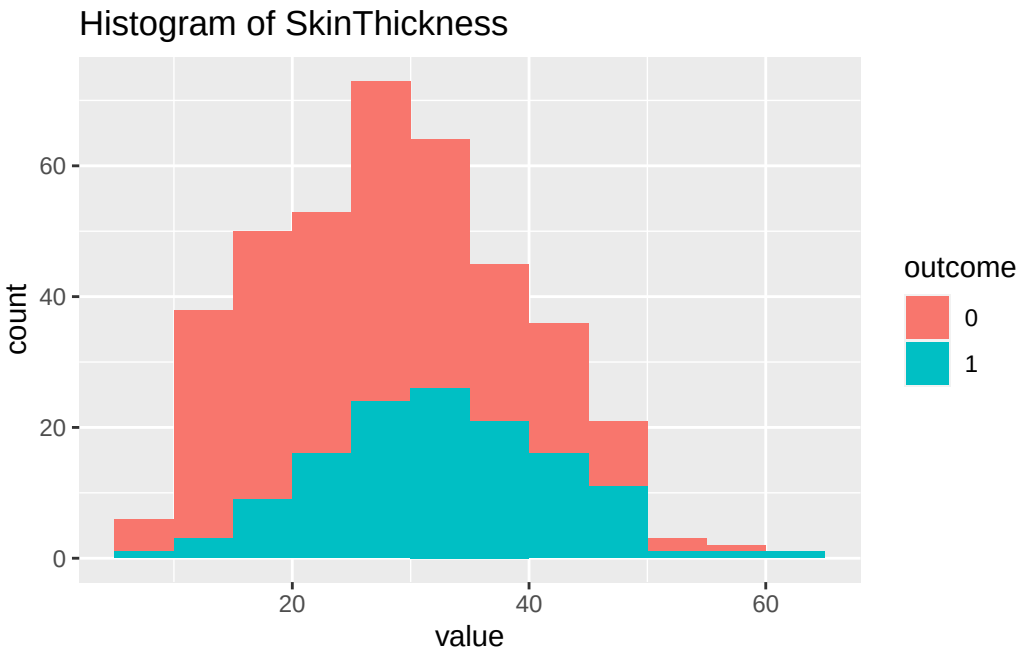

```
[[2]]
```


Histogram of Glucose
100
75
50
25
0
count
50
100
150
200
value
outcome
0
1

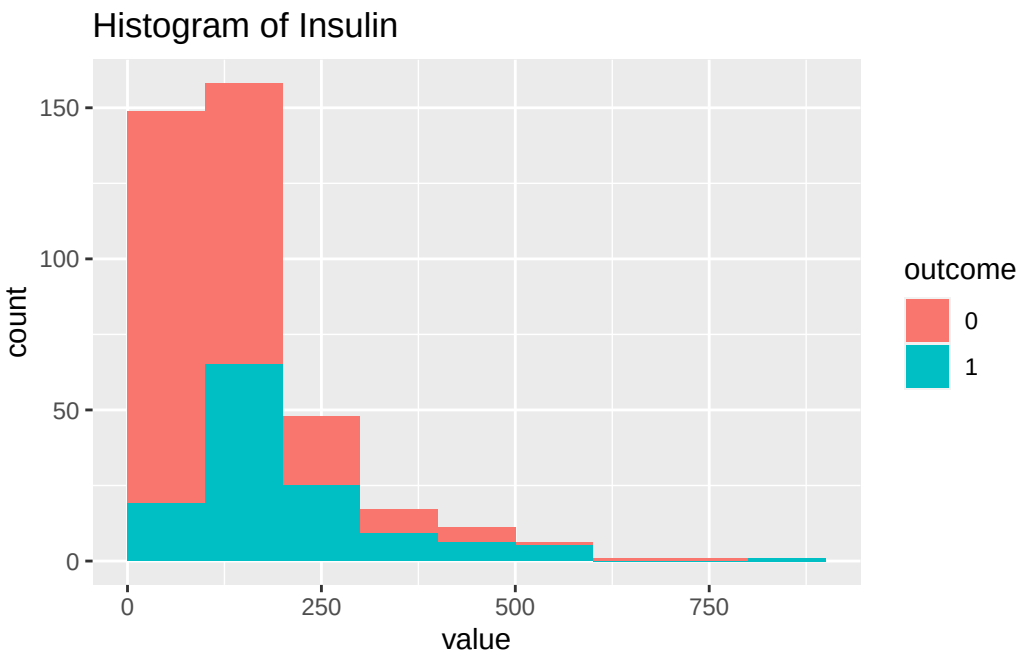
```
[[3]]
```


Histogram of BloodPressure
120
90
60
30
0
count
30
60
90
value
outcome
0
1

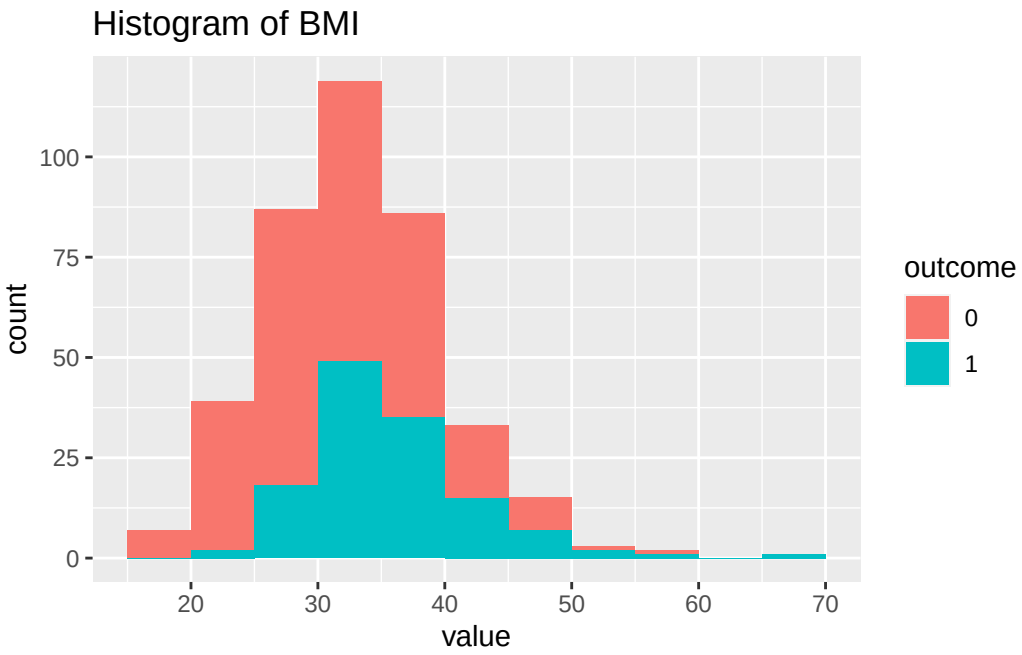
```
[[4]]
```


Histogram of SkinThickness
count
60
40
20
0
20
40
60
value
outcome
0
1

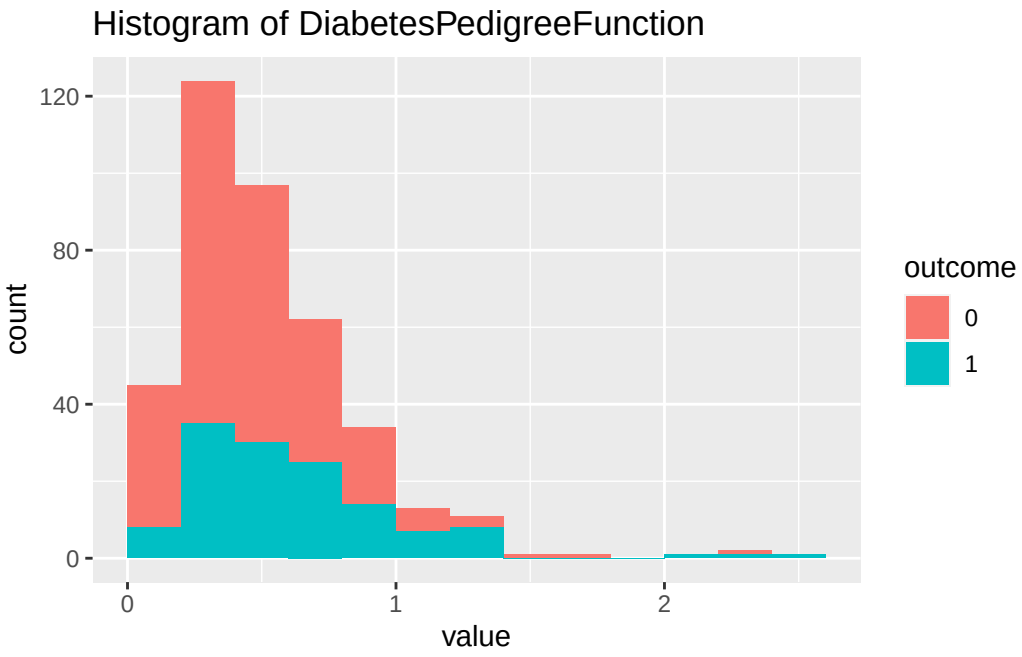
```
[[5]]
```


Histogram of Insulin
150
100
50
0
count
0
250
500
750
value
outcome
0
1

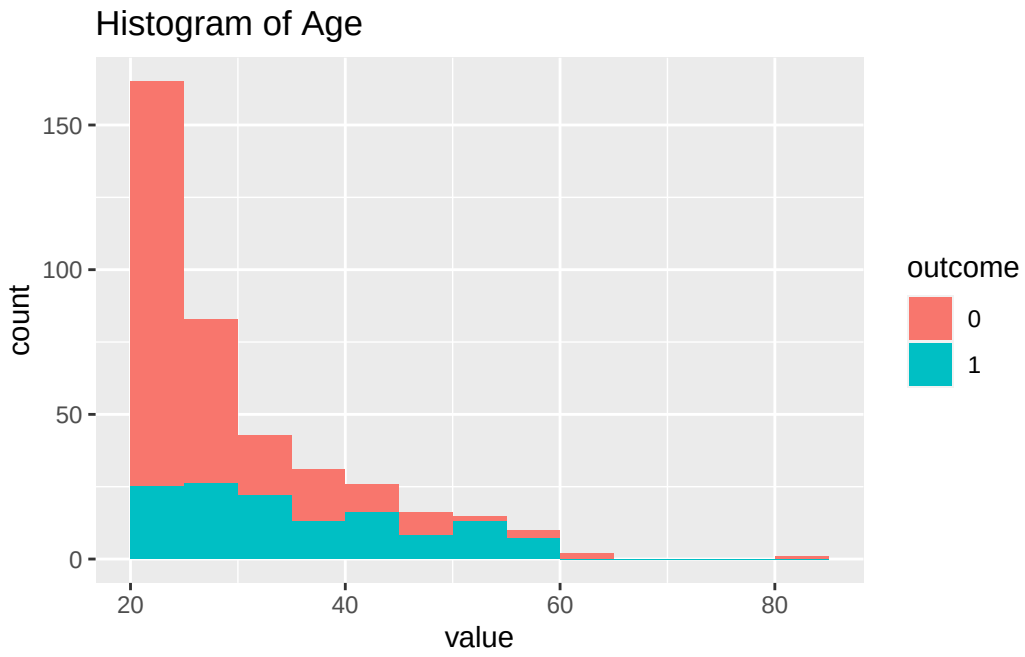
```
[[6]]
```


Histogram of BMI
100
75
50
25
0
count
20
30
40
50
60
70
value
outcome
0
1

[[7]]


Histogram of DiabetesPedigreeFunction
count
120
80
40
0
0
1
2
value
outcome
0
1

[[8]]

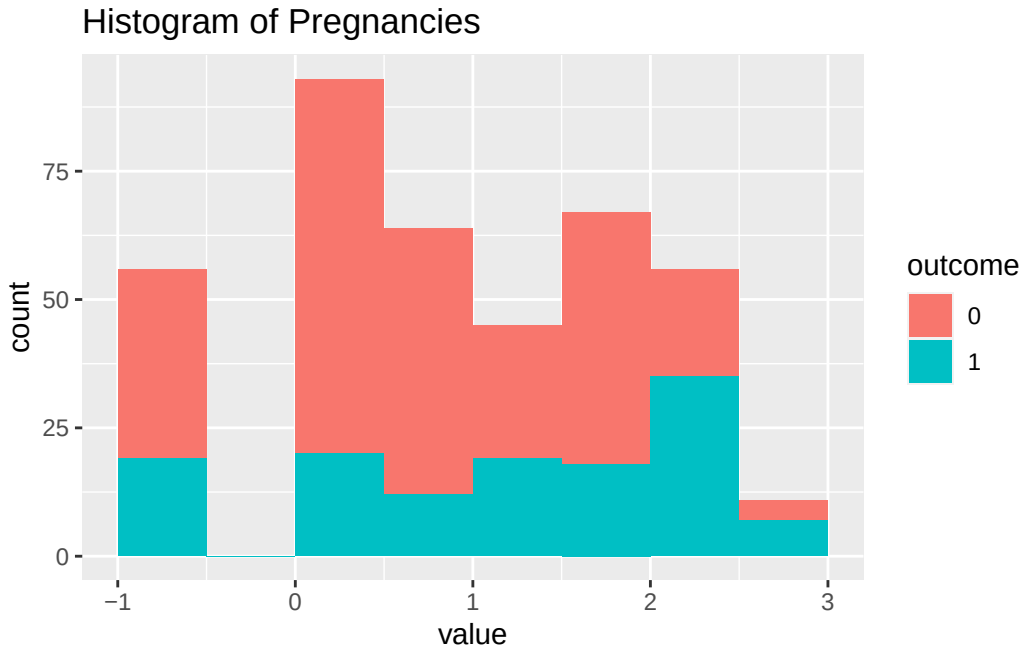


Ahora se realiza las transformaciones

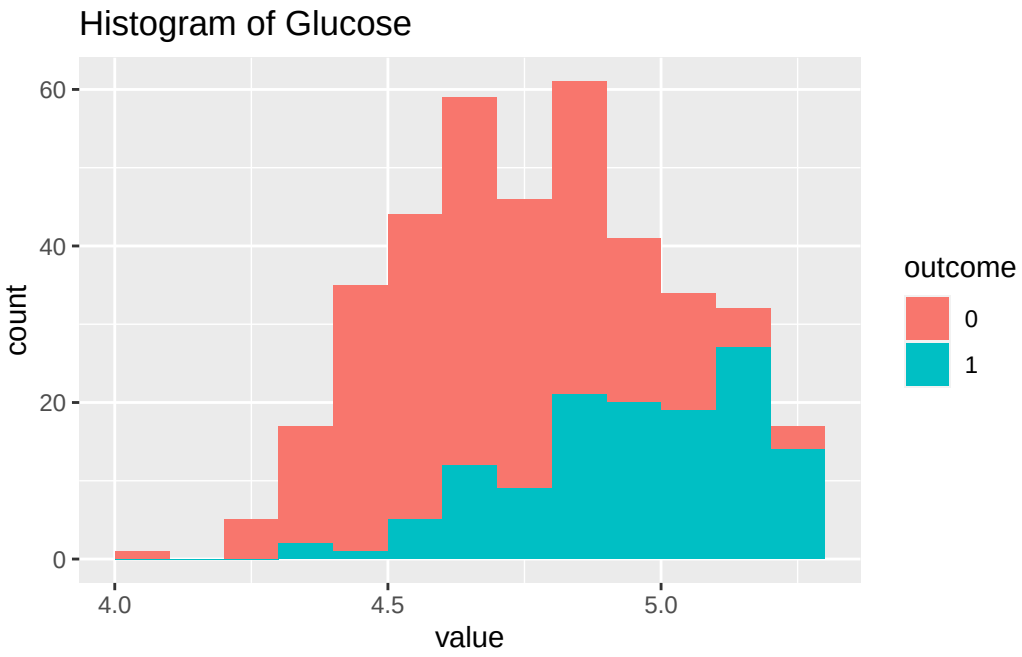
```
datos <- read.csv("./datos/diabetes.csv")
datos[,-c(1,9)] <- apply(datos[,-c(1,9)],2,function(x) ifelse(x==0,NA,x))
datos <- datos[complete.cases(datos),]

datos$Outcome <- as.factor(datos$Outcome)
datos$Insulin <- log(datos$Insulin)
datos$Pregnancies <- log(datos$Pregnancies+0.5)
datos$DiabetesPedigreeFunction <- log(datos$DiabetesPedigreeFunction)

datos$SkinThickness <- sqrt((datos$SkinThickness))
datos$Glucose <- log(datos$Glucose)
datos$Age <-log2(datos$Age)
l.plots <- vector("list",length = ncol(datos)-1)
n1 <- ncol(datos) -1
for(j in 1:n1){

  h <-hist(datos[,j],plot = F)
  datos.tmp <- data.frame(value=datos[,j],outcome=datos$Outcome)
  p1 <- ggplot(datos.tmp,aes(value,fill=outcome))+geom_histogram(breaks=h$breaks) + ggtitl
```

```
  l.plots[[j]] <- p1
}
l.plots
```

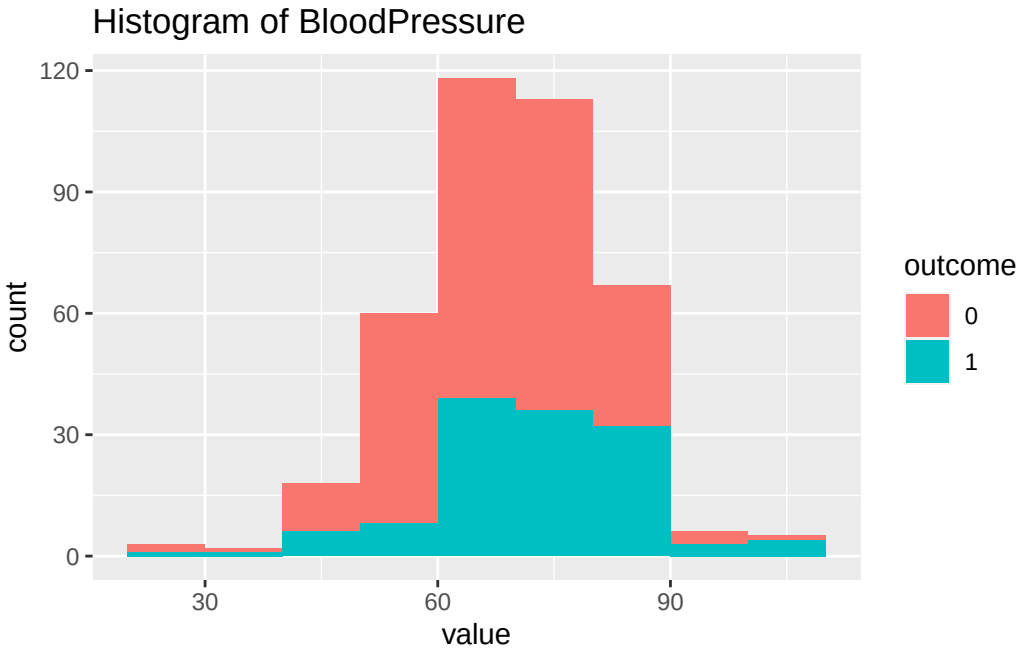
```
[[1]]
```



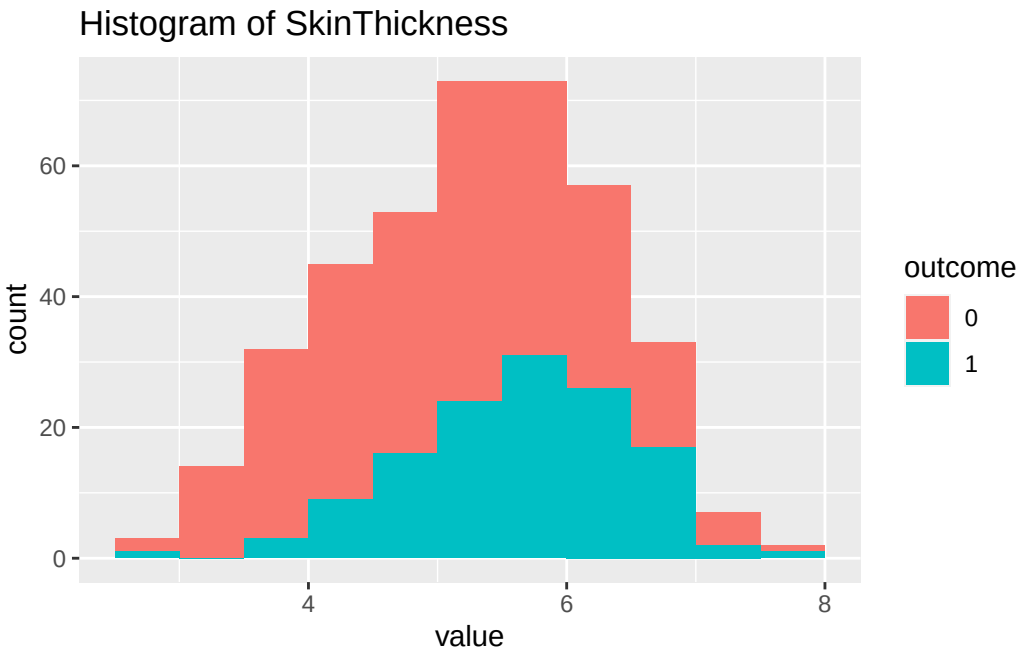

```
[[2]]
```


Histogram of Glucose
count
value
outcome
0
1
60
40
20
0
4.0
4.5
5.0

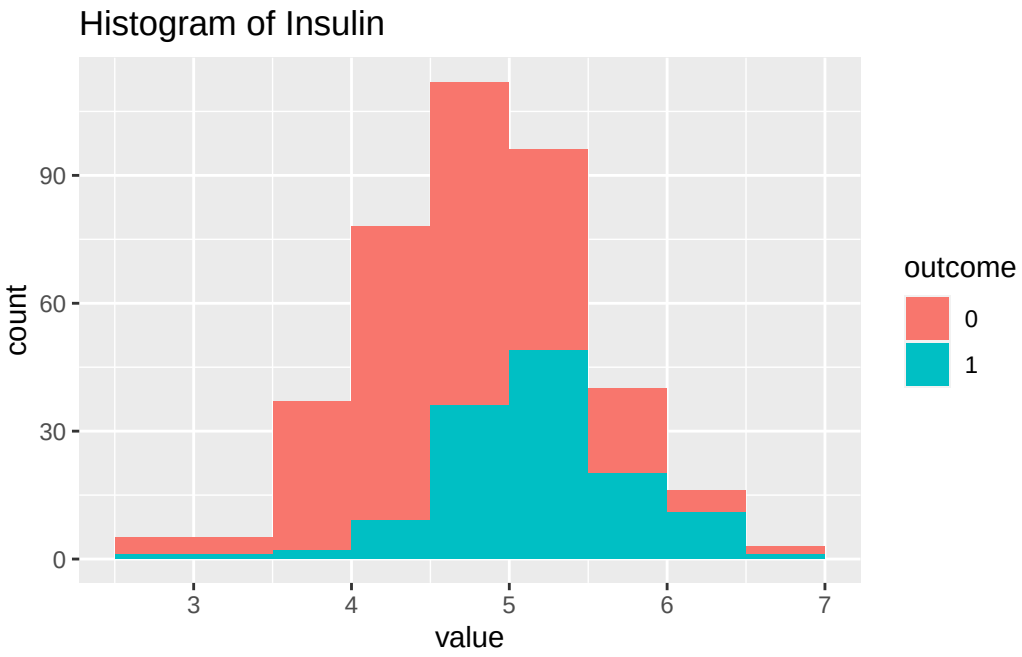
```
[[3]]
```


Histogram of BloodPressure
count
value
outcome
0
1
120
90
60
30
0
30
60
90

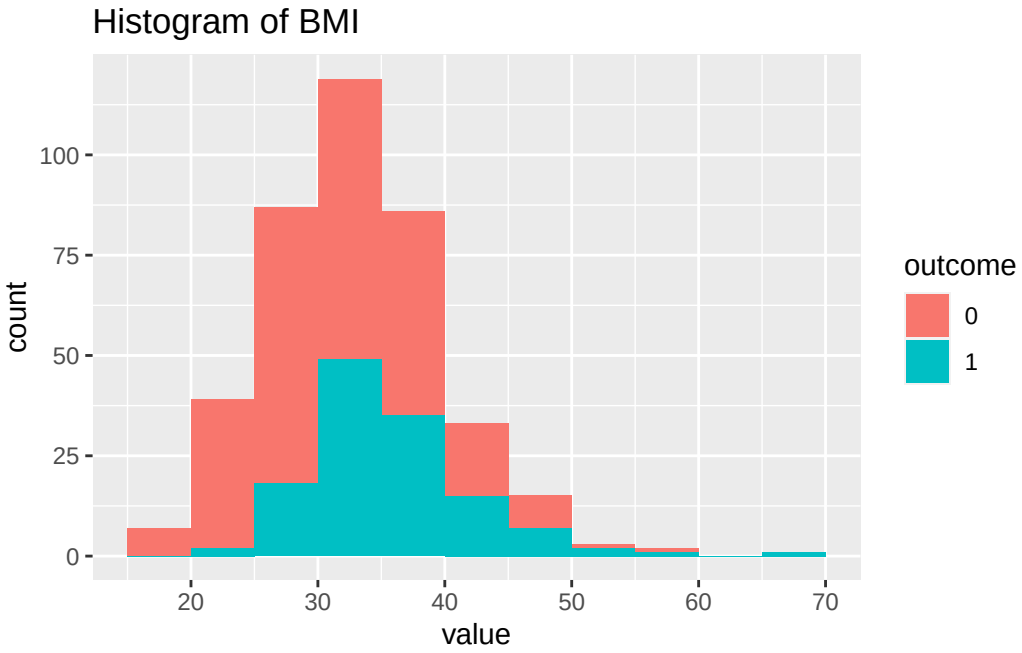
```
[[4]]
```


Histogram of SkinThickness
count
60
40
20
0
4
6
8
value
outcome
0
1

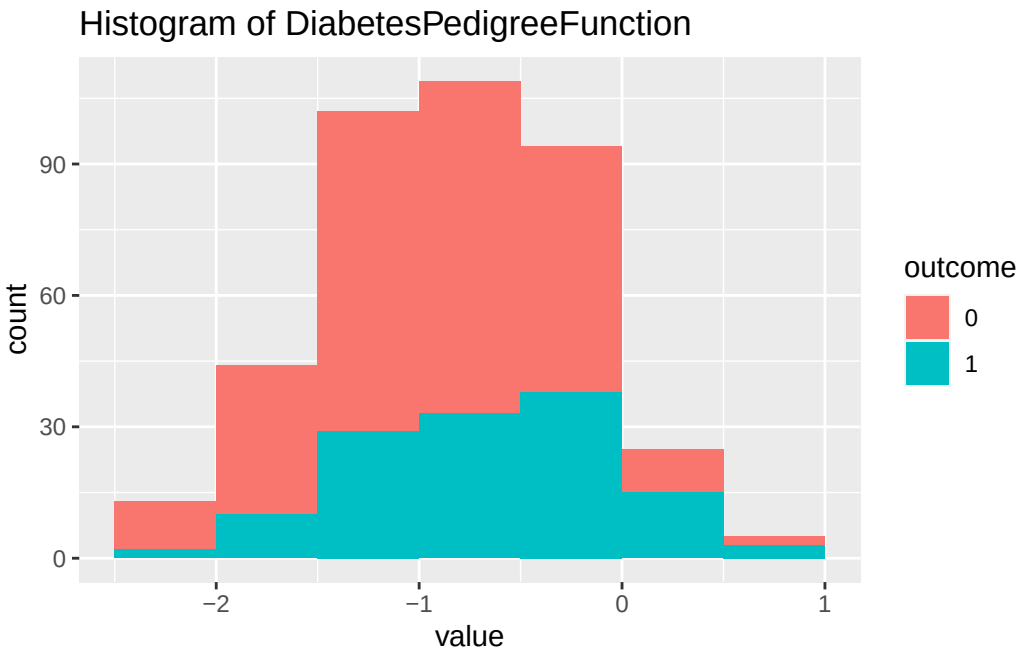
```
[[5]]
```


Histogram of Insulin
count
90
60
30
0
3
4
5
6
7
value
outcome
0
1

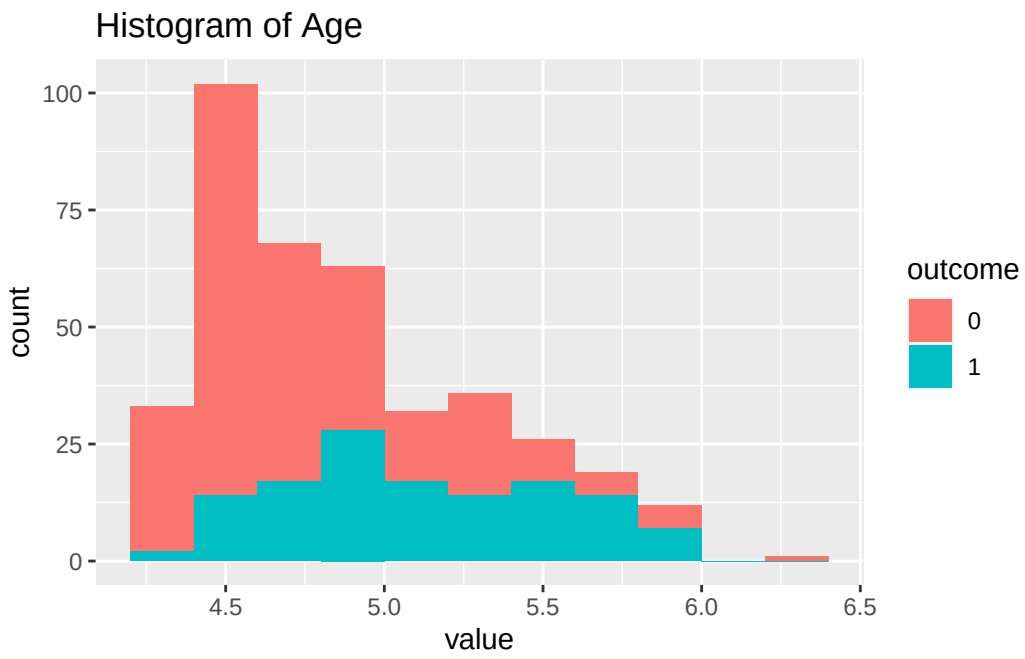
```
[[6]]
```


Histogram of BMI
count
100
75
50
25
0
20
30
40
50
60
70
value
outcome
0
1

```
[[7]]
```


Histogram of DiabetesPedigreeFunction
count
90
60
30
0
−2
−1
0
1
value
outcome
0
1

```
[[8]]
```

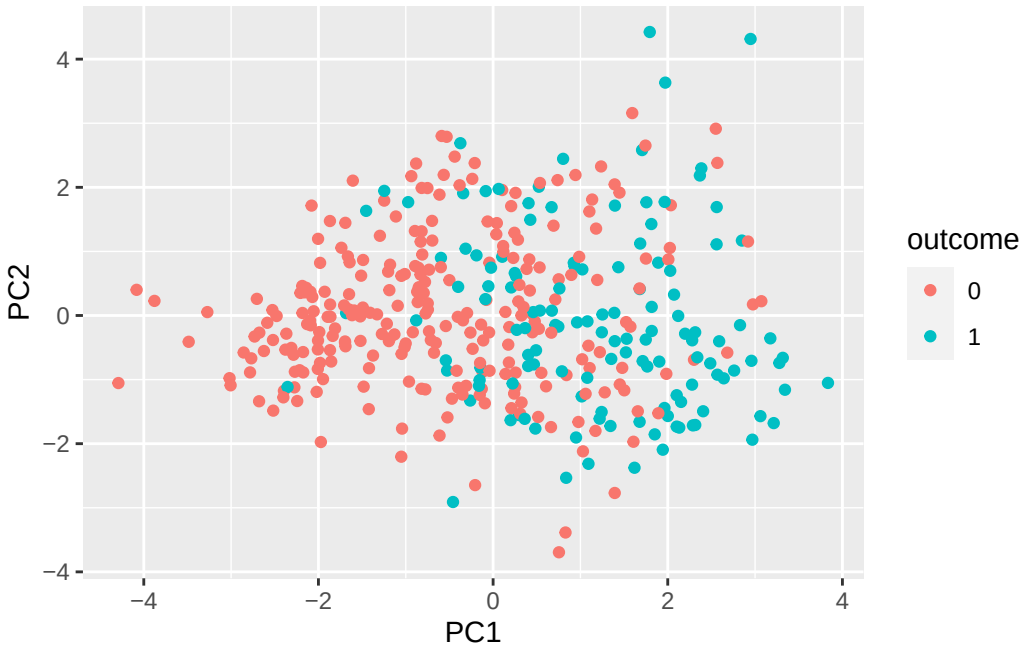


Ahora podemos realizar el PCA otra vez

```
summary(datos)
```

```
 Pregnancies          Glucose      BloodPressure    SkinThickness
Min.   :-0.6931   Min.   :4.025   Min.   : 24.00   Min.   :2.646
1st Qu.: 0.4055   1st Qu.:4.595   1st Qu.: 62.00   1st Qu.:4.583
Median : 0.9163   Median :4.779   Median : 70.00   Median :5.385
Mean   : 0.9590   Mean   :4.778   Mean   : 70.66   Mean   :5.305
3rd Qu.: 1.7047   3rd Qu.:4.963   3rd Qu.: 78.00   3rd Qu.:6.083
Max.   : 2.8622   Max.   :5.288   Max.   :110.00   Max.   :7.937
    Insulin           BMI         DiabetesPedigreeFunction      Age
Min.   :2.639   Min.   :18.20   Min.   :-2.4651          Min.   :4.392
1st Qu.:4.341   1st Qu.:28.40   1st Qu.:-1.3103          1st Qu.:4.524
Median :4.832   Median :33.20   Median :-0.7996          Median :4.755
Mean   :4.813   Mean   :33.09   Mean   :-0.8391          Mean   :4.882
3rd Qu.:5.247   3rd Qu.:37.10   3rd Qu.:-0.3754          3rd Qu.:5.170
Max.   :6.741   Max.   :67.10   Max.   : 0.8838          Max.   :6.340
Outcome
0:262
1:130
```

```r
pcx <- prcomp(datos[,1:n1],scale. = T) ## escalamos por la variablidad de los datos

plotpca <- bind_cols(pcx$x,outcome=datos$Outcome)
ggplot(plotpca,aes(PC1,PC2,color=outcome))+geom_point()
```



Se hace pruebas de medianas

```r
p.norm <- apply(apply(scale(datos[,1:n1]),
            2,
            function(x) summary(lm(x~datos$Outcome))$residuals),
      2,
      shapiro.test)

p.norm
```

```
$Pregnancies
```

```
    Shapiro-Wilk normality test

data:  newX[, i]
W = 0.95146, p-value = 4.684e-10


$Glucose

    Shapiro-Wilk normality test

data:  newX[, i]
W = 0.9958, p-value = 0.3813


$BloodPressure

    Shapiro-Wilk normality test

data:  newX[, i]
W = 0.99011, p-value = 0.009686


$SkinThickness

    Shapiro-Wilk normality test

data:  newX[, i]
W = 0.99384, p-value = 0.1123


$Insulin

    Shapiro-Wilk normality test

data:  newX[, i]
W = 0.99054, p-value = 0.0128


$BMI

    Shapiro-Wilk normality test
```

```
data:  newX[, i]
W = 0.97122, p-value = 5.374e-07


$DiabetesPedigreeFunction

    Shapiro-Wilk normality test

data:  newX[, i]
W = 0.99456, p-value = 0.1796


$Age

    Shapiro-Wilk normality test

data:  newX[, i]
W = 0.93053, p-value = 1.561e-12
```

Se ha logrado alcanzar la normalidad en tan solo dos variables. En caso de que hubiera más variables, procederíamos con t test, pero dado que no es el caso, utilizaremos pruebas de Wilcoxon para el análisis.

```
p.norm <- apply(scale(datos[,1:n1]),
            2,
            function(x) wilcox.test(x~datos$Outcome)$p.value)
```

Notamos que inicialmente todas las variables muestran diferencias significativas, lo cual es algo que debemos corregir.

```
p.adj <- p.adjust(p.norm,"BH")
```

Todas las variables siguen siendo estadísticamente significativas. Ahora procederemos a analizar cuáles de ellas aumentan o disminuyen en comparación con las otras.

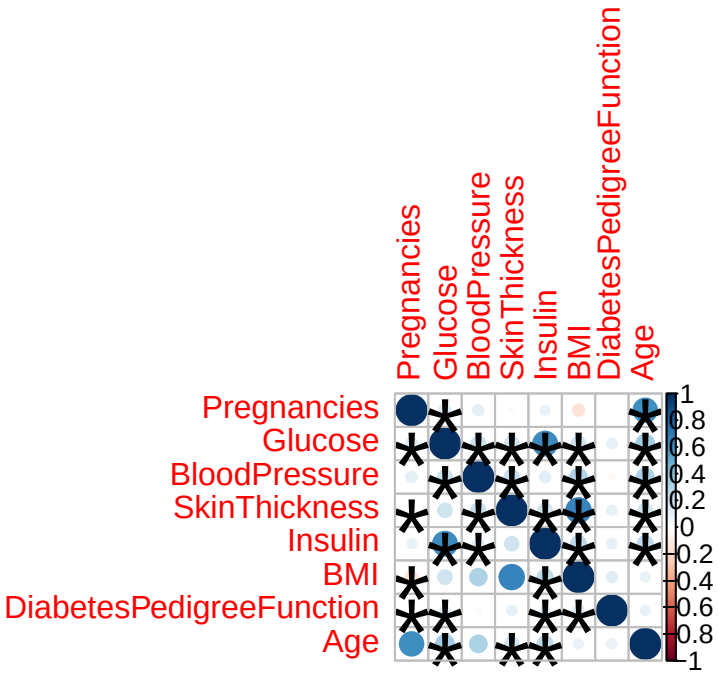
```
datos.split <- split(datos,datos$Outcome)

datos.median <- lapply(datos.split, function(x) apply(x[,-ncol(x)],2,median))
```

```r
toplot <- data.frame(medianas=Reduce("-",datos.median)
,p.values=p.adj)

toplot
```

```
                           medianas     p.values
Pregnancies               -0.3364722 8.957407e-05
Glucose                   -0.2957935 4.902429e-22
BloodPressure             -4.0000000 8.957407e-05
SkinThickness             -0.5484102 4.309442e-07
Insulin                   -0.4788534 3.241934e-13
BMI                       -3.3500000 2.574728e-07
DiabetesPedigreeFunction -0.2779529 8.957407e-05
Age                       -0.4005379 1.577456e-14
```

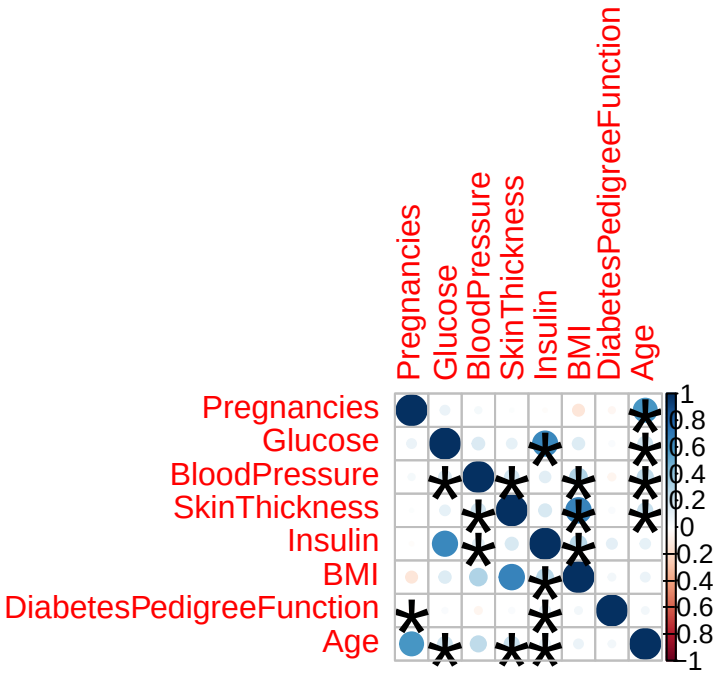
La mayoría de valores significativos con la obesidad

```r
obj.cor <- psych::corr.test(datos[,1:n1])
p.values <- obj.cor$p
p.values[upper.tri(p.values)] <- obj.cor$p.adj
p.values[lower.tri(p.values)] <- obj.cor$p.adj
diag(p.values) <- 1
corrplot::corrplot(corr = obj.cor$r,p.mat = p.values,sig.level = 0.05,insig = "label_sig")
```

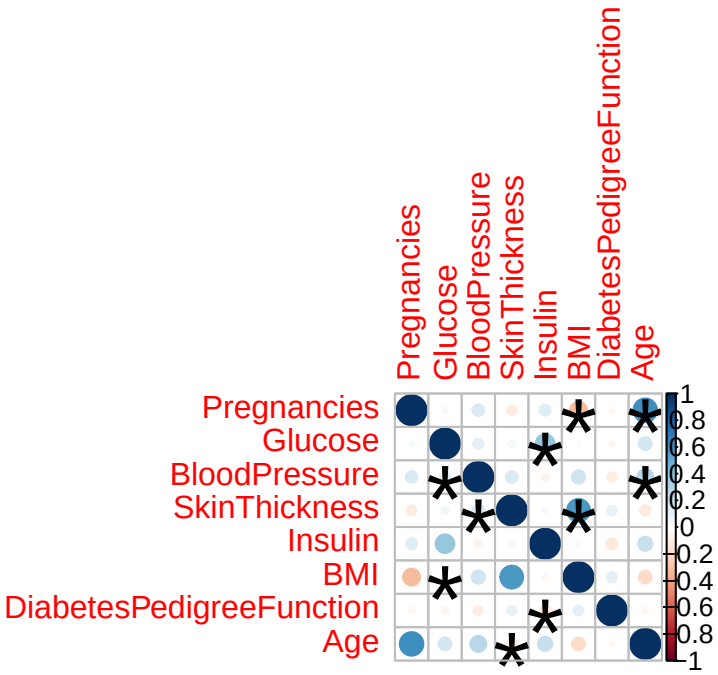


Se puede observar como cambian las relaciones segun la diabetes

```
obj.cor <- psych::corr.test(datos[datos$Outcome==0,1:n1])
p.values <- obj.cor$p
p.values[upper.tri(p.values)] <- obj.cor$p.adj
p.values[lower.tri(p.values)] <- obj.cor$p.adj
diag(p.values) <- 1
corrplot::corrplot(corr = obj.cor$r,p.mat = p.values,sig.level = 0.05,insig = "label_sig")
```




```
obj.cor <- psych::corr.test(datos[datos$Outcome==1,1:n1])
p.values <- obj.cor$p
p.values[upper.tri(p.values)] <- obj.cor$p.adj
p.values[lower.tri(p.values)] <- obj.cor$p.adj
diag(p.values) <- 1
corrplot::corrplot(corr = obj.cor$r,p.mat = p.values,sig.level = 0.05,insig = "label_sig")
```



En otras palabras, hay correlaciones específicas entre la obesidad y la no obesidad, así como otras correlaciones que se deben a factores distintos.

Partición de datos

```
datos[,1:n1] <- as.data.frame(scale(datos[,-ncol(datos)]))
levels(datos$Outcome) <- c("D","N")
train <- sample(nrow(datos),size = nrow(datos)*0.7)

dat.train <- datos[train,]
dat.test <- datos[-train,]
```

Modelado

```
datos[,1:n1] <- as.data.frame(scale(datos[,-ncol(datos)]))

glm.mod <- glm(Outcome ~.,data=dat.train,family = "binomial")
```

```r
prediccion <- as.factor(ifelse(predict(glm.mod,dat.test,type="response")>=0.5,"N","D"))

caret::confusionMatrix(prediccion,dat.test$Outcome)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  D  N
         D 72 11
         N  7 28

               Accuracy : 0.8475
                 95% CI : (0.7697, 0.907)
    No Information Rate : 0.6695
    P-Value [Acc > NIR] : 1.001e-05

                  Kappa : 0.6461

 Mcnemar's Test P-Value : 0.4795

            Sensitivity : 0.9114
            Specificity : 0.7179
         Pos Pred Value : 0.8675
         Neg Pred Value : 0.8000
             Prevalence : 0.6695
         Detection Rate : 0.6102
   Detection Prevalence : 0.7034
      Balanced Accuracy : 0.8147

       'Positive' Class : D
```

Lasso

```r
tuneGrid=expand.grid(
               .alpha=0,
               .lambda=seq(0, 1, by = 0.001))
trainControl <- trainControl(method = "repeatedcv",
                         number = 10,
                         repeats = 3,
                         # prSummary needs calculated class,
```

```
                        classProbs = T)

  model <- train(Outcome ~ ., data = dat.train, method = "glmnet", trControl = trainControl,
                                          metric="Accuracy"
  )

  confusionMatrix(predict(model,dat.test[,-ncol(dat.test)]),dat.test$Outcome)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  D  N
         D 75 15
         N  4 24

               Accuracy : 0.839
                 95% CI : (0.76, 0.9002)
    No Information Rate : 0.6695
    P-Value [Acc > NIR] : 2.695e-05

                  Kappa : 0.6082

 Mcnemar's Test P-Value : 0.02178

            Sensitivity : 0.9494
            Specificity : 0.6154
         Pos Pred Value : 0.8333
         Neg Pred Value : 0.8571
             Prevalence : 0.6695
         Detection Rate : 0.6356
   Detection Prevalence : 0.7627
      Balanced Accuracy : 0.7824

       'Positive' Class : D
```

```
  tuneGrid=expand.grid(
                .alpha=1,
                .lambda=seq(0, 1, by = 0.0001))
  trainControl <- trainControl(method = "repeatedcv",
```

```
                        number = 10,
                        repeats = 3,
                        # prSummary needs calculated class,
                        classProbs = T)

model <- train(Outcome ~ ., data = dat.train, method = "glmnet", trControl = trainControl,
                                        metric="Accuracy"
)

confusionMatrix(predict(model,dat.test[,-ncol(dat.test)]),dat.test$Outcome)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  D  N
         D 72 11
         N  7 28

               Accuracy : 0.8475
                 95% CI : (0.7697, 0.907)
    No Information Rate : 0.6695
    P-Value [Acc > NIR] : 1.001e-05

                  Kappa : 0.6461

 Mcnemar's Test P-Value : 0.4795

            Sensitivity : 0.9114
            Specificity : 0.7179
         Pos Pred Value : 0.8675
         Neg Pred Value : 0.8000
             Prevalence : 0.6695
         Detection Rate : 0.6102
   Detection Prevalence : 0.7034
      Balanced Accuracy : 0.8147

       'Positive' Class : D
```

```r
datos[,1:n1] <- as.data.frame(scale(datos[,-ncol(datos)]))
levels(datos$Outcome) <- c("D","N")
train <- sample(nrow(datos),size = nrow(datos)*0.7)

dat.train <- datos[train,]
dat.test <- datos[-train,]
mdl <- naiveBayes(Outcome ~ .,data=dat.train,laplace = 0)
prediccion <-predict(mdl,dat.test[,-ncol(dat.test)])
confusionMatrix(prediccion,dat.test$Outcome)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  D  N
         D 65 13
         N 10 30

               Accuracy : 0.8051
                 95% CI : (0.722, 0.8722)
    No Information Rate : 0.6356
    P-Value [Acc > NIR] : 4.866e-05

                  Kappa : 0.5729

 Mcnemar's Test P-Value : 0.6767

            Sensitivity : 0.8667
            Specificity : 0.6977
         Pos Pred Value : 0.8333
         Neg Pred Value : 0.7500
             Prevalence : 0.6356
         Detection Rate : 0.5508
   Detection Prevalence : 0.6610
      Balanced Accuracy : 0.7822

       'Positive' Class : D
```

```r
lambda_use <- min(model$finalModel$lambda[model$finalModel$lambda >= model$bestTune$lambda
position <- which(model$finalModel$lambda == lambda_use)
```

```r
featsele <- data.frame(coef(model$finalModel)[, position])
```

```r
rownames(featsele)[featsele$coef.model.finalModel....position.!=0]
```

```
[1] "(Intercept)"              "Glucose"
[3] "SkinThickness"            "Insulin"
[5] "BMI"                      "DiabetesPedigreeFunction"
[7] "Age"
```

```r
mdl.sel <-naiveBayes(Outcome ~ Insulin+Glucose+DiabetesPedigreeFunction+Age,data = dat.tra
```

```r
prediccion <- predict(mdl.sel,dat.test[,-ncol(dat.test)])
```

```r
confusionMatrix(prediccion,dat.test$Outcome)
```

```
Confusion Matrix and Statistics

          Reference
Prediction  D  N
         D 66 16
         N  9 27

               Accuracy : 0.7881
                 95% CI : (0.7033, 0.858)
    No Information Rate : 0.6356
    P-Value [Acc > NIR] : 0.0002564

                  Kappa : 0.5262

 Mcnemar's Test P-Value : 0.2301393

            Sensitivity : 0.8800
            Specificity : 0.6279
         Pos Pred Value : 0.8049
         Neg Pred Value : 0.7500
             Prevalence : 0.6356
         Detection Rate : 0.5593
   Detection Prevalence : 0.6949
      Balanced Accuracy : 0.7540
```

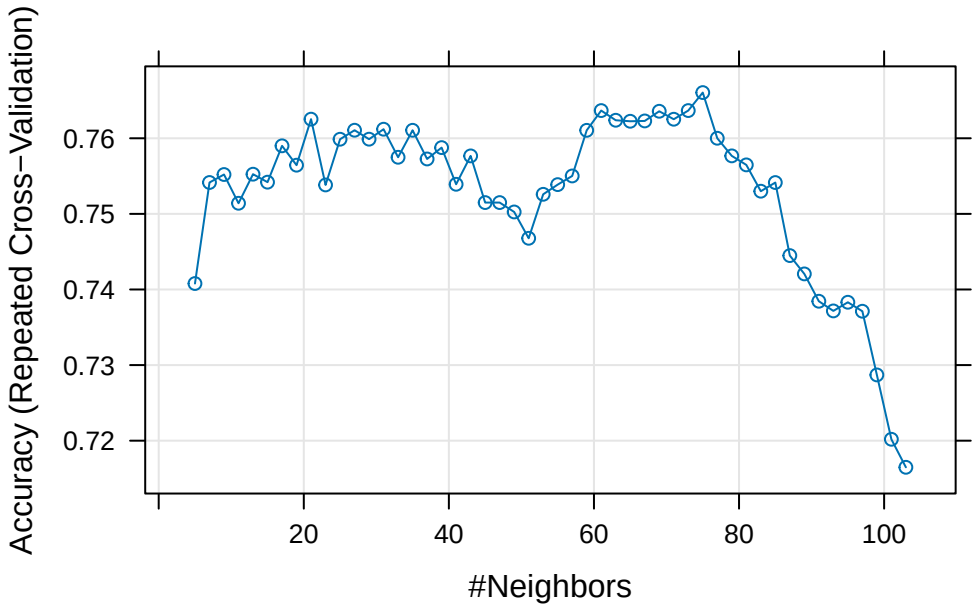
```
      'Positive' Class : D
```

```r
library(ISLR)
library(caret)
set.seed(400)
ctrl <- trainControl(method="repeatedcv",repeats = 3) #,classProbs=TRUE,summaryFunction =
knnFit <- train(Outcome ~ ., data = dat.train, method = "knn", trControl = ctrl, preProces

#Output of kNN fit
knnFit
```

```
k-Nearest Neighbors

274 samples
  8 predictor
  2 classes: 'D', 'N'

Pre-processing: centered (8), scaled (8)
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 247, 248, 246, 246, 248, 246, ...
Resampling results across tuning parameters:

  k    Accuracy   Kappa
    5  0.7407882  0.3677020
    7  0.7541514  0.3895086
    9  0.7552062  0.3911773
   11  0.7514042  0.3692961
   13  0.7552503  0.3808030
   15  0.7541955  0.3764246
   17  0.7590015  0.3799037
   19  0.7564408  0.3721568
   21  0.7625288  0.3842463
   23  0.7538292  0.3624573
   25  0.7598732  0.3808860
   27  0.7610670  0.3851855
   29  0.7598799  0.3798491
   31  0.7611993  0.3841737
   33  0.7574888  0.3729076
   35  0.7610602  0.3820950
```

```
  37  0.7572650  0.3710281
  39  0.7587675  0.3755547
  41  0.7539174  0.3583843
  43  0.7576686  0.3662317
  45  0.7514957  0.3508114
  47  0.7514957  0.3486822
  49  0.7502544  0.3462943
  51  0.7467745  0.3296410
  53  0.7525912  0.3417677
  55  0.7538699  0.3454670
  57  0.7550163  0.3445749
  59  0.7610535  0.3590659
  61  0.7636650  0.3665274
  63  0.7623864  0.3601235
  65  0.7622507  0.3569641
  67  0.7622948  0.3556200
  69  0.7635735  0.3603590
  71  0.7625187  0.3562611
  73  0.7636650  0.3560092
  75  0.7660460  0.3624019
  77  0.7600054  0.3393928
  79  0.7576720  0.3317154
  81  0.7564815  0.3278284
  83  0.7529982  0.3153590
  85  0.7541446  0.3188825
  87  0.7444851  0.2851928
  89  0.7420601  0.2770902
  91  0.7384446  0.2640745
  93  0.7371659  0.2565293
  95  0.7383123  0.2583348
  97  0.7371218  0.2537169
  99  0.7287003  0.2182288
 101  0.7201872  0.1895384
 103  0.7164801  0.1724485

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 75.
```

```r
plot(knnFit)
```




```
knnPredict <- predict(knnFit,newdata = dat.test[,-ncol(dat.test)] )
#Get the confusion matrix to see accuracy value and other parameter values
confusionMatrix(knnPredict, dat.test$Outcome )
```

```
Confusion Matrix and Statistics

          Reference
Prediction  D  N
         D 71 28
         N  4 15

               Accuracy : 0.7288
                 95% CI : (0.6392, 0.8065)
    No Information Rate : 0.6356
    P-Value [Acc > NIR] : 0.02062

                  Kappa : 0.3354

 Mcnemar's Test P-Value : 4.785e-05

            Sensitivity : 0.9467
```

```
            Specificity : 0.3488
         Pos Pred Value : 0.7172
         Neg Pred Value : 0.7895
             Prevalence : 0.6356
         Detection Rate : 0.6017
   Detection Prevalence : 0.8390
      Balanced Accuracy : 0.6478

       'Positive' Class : D
```

```
library(caret)
datos <- read.csv("./datos/diabetes.csv")
datos$Outcome <-as.factor(datos$Outcome)
datos[,1:n1] <- as.data.frame(scale(datos[,-ncol(datos)]))
levels(datos$Outcome) <- c("D","N")
train <- sample(nrow(datos),size = nrow(datos)*0.7)

dat.train <- datos[train,]
dat.test <- datos[-train,]
set.seed(1001)
ctrl<-trainControl(method="repeatedcv",number=10,classProbs = TRUE,summaryFunction = twoCl
plsda<-train(x=dat.train[,-ncol(datos)], # spectral data
             y=dat.train$Outcome, # factor vector
             method="pls", # pls-da algorithm
             tuneLength=10, # number of components
             trControl=ctrl, # ctrl contained cross-validation option
             preProc=c("center","scale"), # the data are centered and scaled
             metric="ROC") # metric is ROC for 2 classes
plsda
```

```
Partial Least Squares

537 samples
  8 predictor
  2 classes: 'D', 'N'

Pre-processing: centered (8), scaled (8)
Resampling: Cross-Validated (10 fold, repeated 1 times)
Summary of sample sizes: 483, 484, 483, 483, 483, 483, ...
Resampling results across tuning parameters:
```

```
  ncomp  ROC        Sens       Spec
  1      0.8183485  0.8468067  0.5657895
  2      0.8348713  0.8667227  0.6181579
  3      0.8346068  0.8814286  0.6023684
  4      0.8342848  0.8756303  0.6076316
  5      0.8338425  0.8784874  0.6023684
  6      0.8336922  0.8784874  0.6023684
  7      0.8336922  0.8784874  0.6023684

ROC was used to select the optimal model using the largest value.
The final value used for the model was ncomp = 2.
```

```r
prediccion <- predict(plsda,newdata = dat.test[,-ncol(datos)])

confusionMatrix(prediccion,dat.test$Outcome)
```

```
Confusion Matrix and Statistics

          Reference
Prediction   D   N
         D 135  40
         N  19  37

               Accuracy : 0.7446
                 95% CI : (0.6833, 0.7995)
    No Information Rate : 0.6667
    P-Value [Acc > NIR] : 0.006419

                  Kappa : 0.3833

 Mcnemar's Test P-Value : 0.009220

            Sensitivity : 0.8766
            Specificity : 0.4805
         Pos Pred Value : 0.7714
         Neg Pred Value : 0.6607
             Prevalence : 0.6667
         Detection Rate : 0.5844
   Detection Prevalence : 0.7576
      Balanced Accuracy : 0.6786
```

```
      'Positive' Class : D
```

Si modificamos lambda

```
datos <- read.csv("./datos/diabetes.csv")
datos$Outcome <-as.factor(datos$Outcome)
levels(datos$Outcome) <- c("D","N")
train <- sample(nrow(datos),size = nrow(datos)*0.7)

dat.train <- datos[train,]
dat.test <- datos[-train,]
lambda <- seq(0,50,0.1)

  modelo <- naiveBayes(dat.train[,-ncol(datos)],dat.train$Outcome)

  predicciones <- predict(modelo,dat.test[,-ncol(datos)])

confusionMatrix(predicciones,dat.test$Outcome)$overall[1]
```

```
 Accuracy
0.7705628
```

```
datos <- read.csv("./datos/diabetes.csv")
datos$Outcome <-as.factor(datos$Outcome)
datos[,1:n1] <- as.data.frame(scale(datos[,-ncol(datos)]))
levels(datos$Outcome) <- c("D","N")
train <- sample(nrow(datos),size = nrow(datos)*0.7)

dat.train <- datos[train,]
dat.test <- datos[-train,]
library(caret)
set.seed(1001)
ctrl<-trainControl(method="repeatedcv",number=10,classProbs = TRUE,summaryFunction = twoCl
plsda<-train(x=dat.train[,c(2,5,7,8)], # spectral data
             y=dat.train$Outcome, # factor vector
             method="pls", # pls-da algorithm
             tuneLength=10, # number of components
             trControl=ctrl, # ctrl contained cross-validation option
             preProc=c("center","scale"), # the data are centered and scaled
```

```
              metric="ROC") # metric is ROC for 2 classes

prediccion <- predict(plsda,dat.test[,c(2,5,7,8)])
confusionMatrix(prediccion,dat.test$Outcome)
```

```
Confusion Matrix and Statistics

          Reference
Prediction   D   N
         D 136  42
         N   9  44

               Accuracy : 0.7792
                 95% CI : (0.7201, 0.831)
    No Information Rate : 0.6277
    P-Value [Acc > NIR] : 5.532e-07

                  Kappa : 0.4876

 Mcnemar's Test P-Value : 7.433e-06

            Sensitivity : 0.9379
            Specificity : 0.5116
         Pos Pred Value : 0.7640
         Neg Pred Value : 0.8302
             Prevalence : 0.6277
         Detection Rate : 0.5887
   Detection Prevalence : 0.7706
      Balanced Accuracy : 0.7248

       'Positive' Class : D
```

Se puede hacer un análisis de la varianza multivariante

```
library(vegan)
```

```
Loading required package: permute

This is vegan 2.6-4
```

```
Attaching package: 'vegan'

The following object is masked from 'package:caret':

    tolerance
```

```r
adonis2(datos[,-ncol(datos)] ~datos$Outcome,method = "euclidean")
```

```
Permutation test for adonis under reduced model
Terms added sequentially (first to last)
Permutation: free
Number of permutations: 999

adonis2(formula = datos[, -ncol(datos)] ~ datos$Outcome, method = "euclidean")
               Df SumOfSqs      R2      F Pr(>F)
datos$Outcome   1    357.8 0.05831 47.434  0.001 ***
Residual      766   5778.2 0.94169
Total         767   6136.0 1.00000
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

En resumen, aunque las variables por sí solas no pueden detectar la diabetes como variables independientes, si las consideramos como variables dependientes de la diabetes, encontramos correlaciones significativas. Esto implica que la diabetes influye en los parámetros analizados, pero es menos probable que la diabetes sea la causa de estas alteraciones, con una precisión del 77 por ciento.

Es importante tener en cuenta que las variables explican solo el 77 por ciento de la diabetes, mientras que la propia condición de la diabetes tiene un mayor impacto en la media global.

Para investigar más a fondo, se podría realizar un análisis de correlación parcial teniendo en cuenta la diabetes, con el objetivo de identificar las variables específicamente relacionadas con esta condición.