```
##
## 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

## Loading required package: lattice

## Loading required package: ggplot2

##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

##
## ---------------------
## Welcome to dendextend version 1.13.4
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues
## Or contact: <tal.galili@gmail.com>
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendextend))
## ---------------------

##
## Attaching package: 'dendextend'

## The following object is masked from 'package:stats':
##
##     cutree

## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa

##
## Attaching package: 'e1071'

## The following object is masked from 'package:pracma':
##
##     sigmoid
```

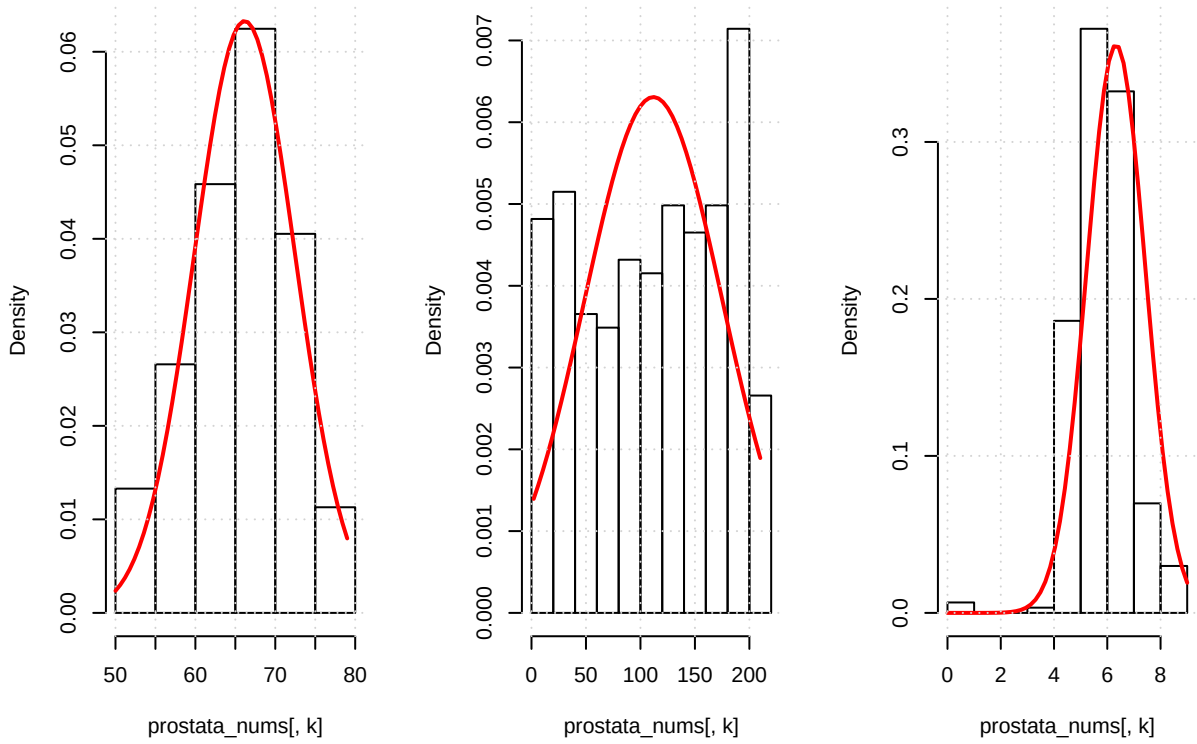
```r
# Defino funciones
sampleo <-function(dataset,split)
  {
    dni = 36637757
    n = round(split* nrow(dataset))
```

```
    set.seed(dni)
    mask = sample(1:nrow(dataset),size=n,replace=FALSE)
    dataset = dataset[mask,]
    return(dataset)
}

train_test <-function(dataset,split)
{
  dni = 36637757
  n = round(split* nrow(dataset))
  set.seed(dni)
  mask = sample(1:nrow(dataset),size=n,replace=FALSE)
  dataset_train = dataset[mask,]
  dataset_test = dataset[-mask,]
  return(list(dataset_train,dataset_test))
}

prostata_DataTypes <- function(dataset)
  {
    # elimino el id
    dataset = dataset[,-8]
    dataset = dataset[,-1]
    # Seteo como factores
    dataset$CAPSULE = as.factor(dataset$CAPSULE)
    dataset$RACE = as.factor(dataset$RACE)
    dataset$DPROS = as.factor(dataset$DPROS)
    dataset$DCAPS = as.factor(dataset$DCAPS)

    # Seteo como numeros
    dataset$PSA = as.numeric(dataset$PSA)

    # Elimino filas con NA en gleason dado que hay errores en otros datos
    dataset = dataset[!is.na(dataset$GLEASON),]
    return(dataset)
}

seguros_Datatypes <- function(dataset)
{
  dataset$edad = as.numeric(dataset$edad)
  dataset$BMI= as.numeric(dataset$BMI)
  dataset$hijos= as.numeric(dataset$hijos)
  dataset$fuma= as.numeric(dataset$fuma)
  dataset$cargos= as.numeric(dataset$cargos)
  dataset$primadelseguro= as.numeric(dataset$primadelseguro)

  dataset$sexo = as.factor(dataset$sexo)
  dataset$region = as.factor(dataset$region)

  return(dataset)

}
```

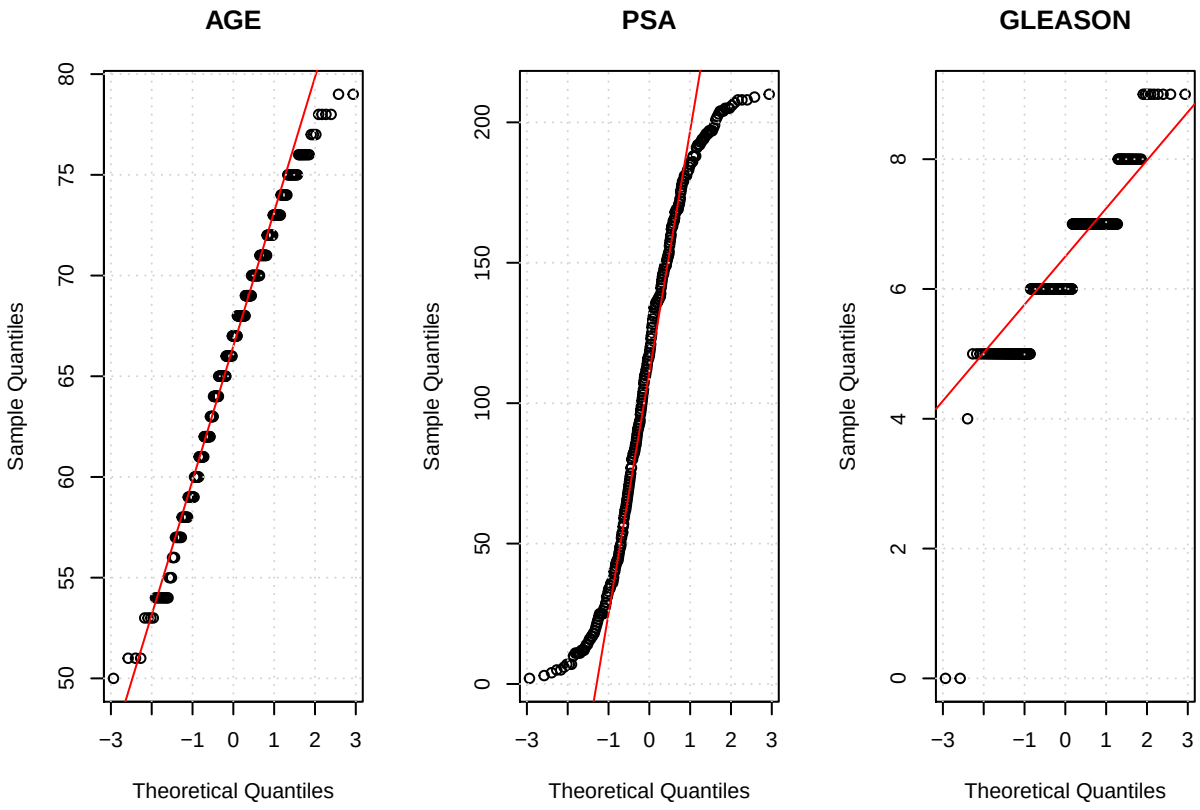
```
prostata = read.csv(file ="./Prostata.csv",dec = ",")
prostata = sampleo(prostata,0.8)
```

```
prostata <- prostata_DataTypes(prostata) # Corrijo tipos de datos, elimino nulos,etc

nums <- unlist(lapply(prostata, is.numeric))
prostata_nums = prostata[,nums]
prostata_factor = prostata[,!nums]

prostata.train_test = train_test(prostata,split = 0.8)  # Dejo los dataset para clasificar no ingenuame
```

```
par(mfcol = c(1,length(prostata_nums)))
for (k in 1:length(prostata_nums))
  {
    hist(prostata_nums[,k],proba = T,main = names(prostata_nums[,k]),10)
    x0 <- seq(min(prostata_nums[, k]), max(prostata_nums[, k]), le = 50)
    lines(x0, dnorm(x0, mean(prostata_nums[,k]), sd(prostata_nums[,k])), col = "red", lwd = 2)
    grid()
}
```




```
pval = list()
par(mfcol = c(1,length(prostata_nums)))
for (k in 1:length(prostata_nums)){
  qqnorm(prostata_nums[,k],main = names(prostata_nums[k]))
  qqline(prostata_nums[,k],col="red")
  pval[k] = ad.test(prostata_nums[,k])$p.value
  grid()
}
```



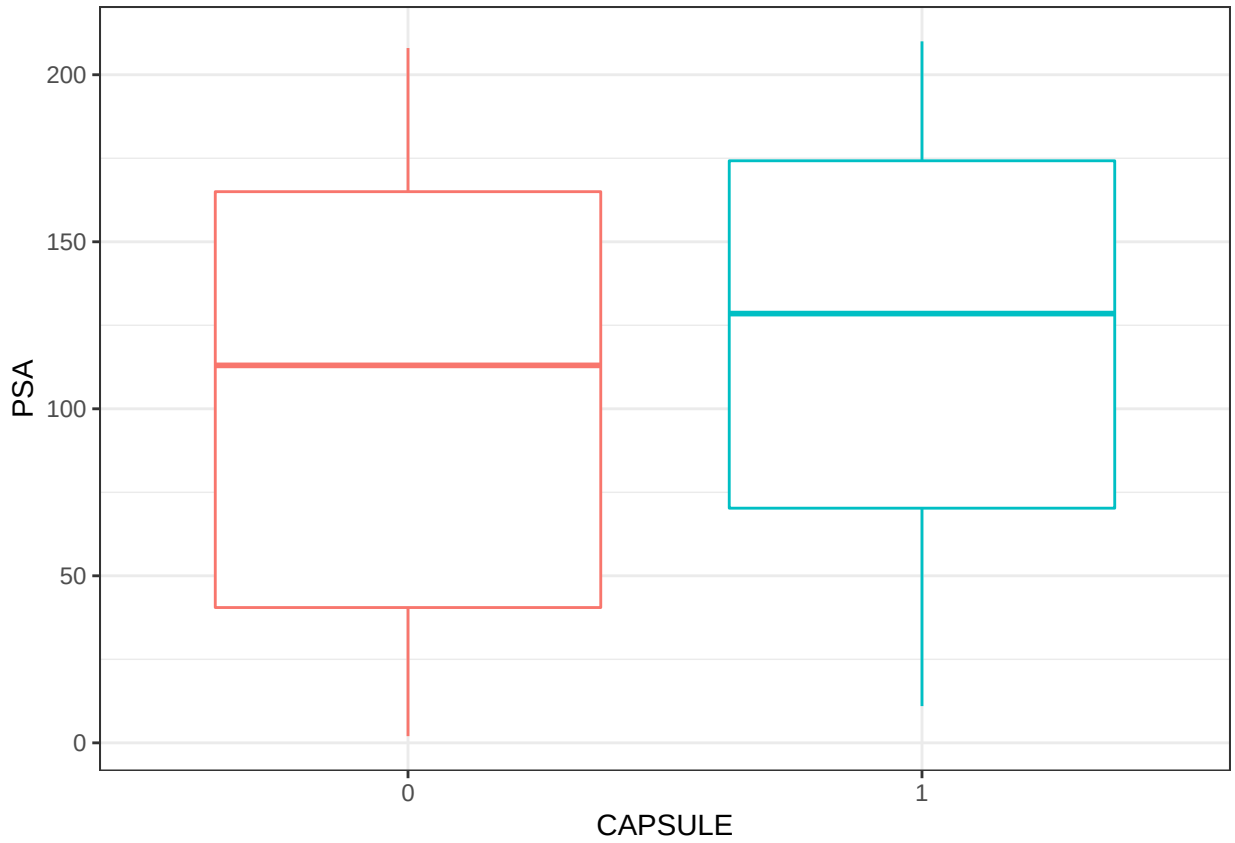

```
pval # Rechazo normalidad por en todas
```

```
## [[1]]
## [1] 0.001191781
##
## [[2]]
## [1] 1.185092e-13
##
## [[3]]
## [1] 3.7e-24
```

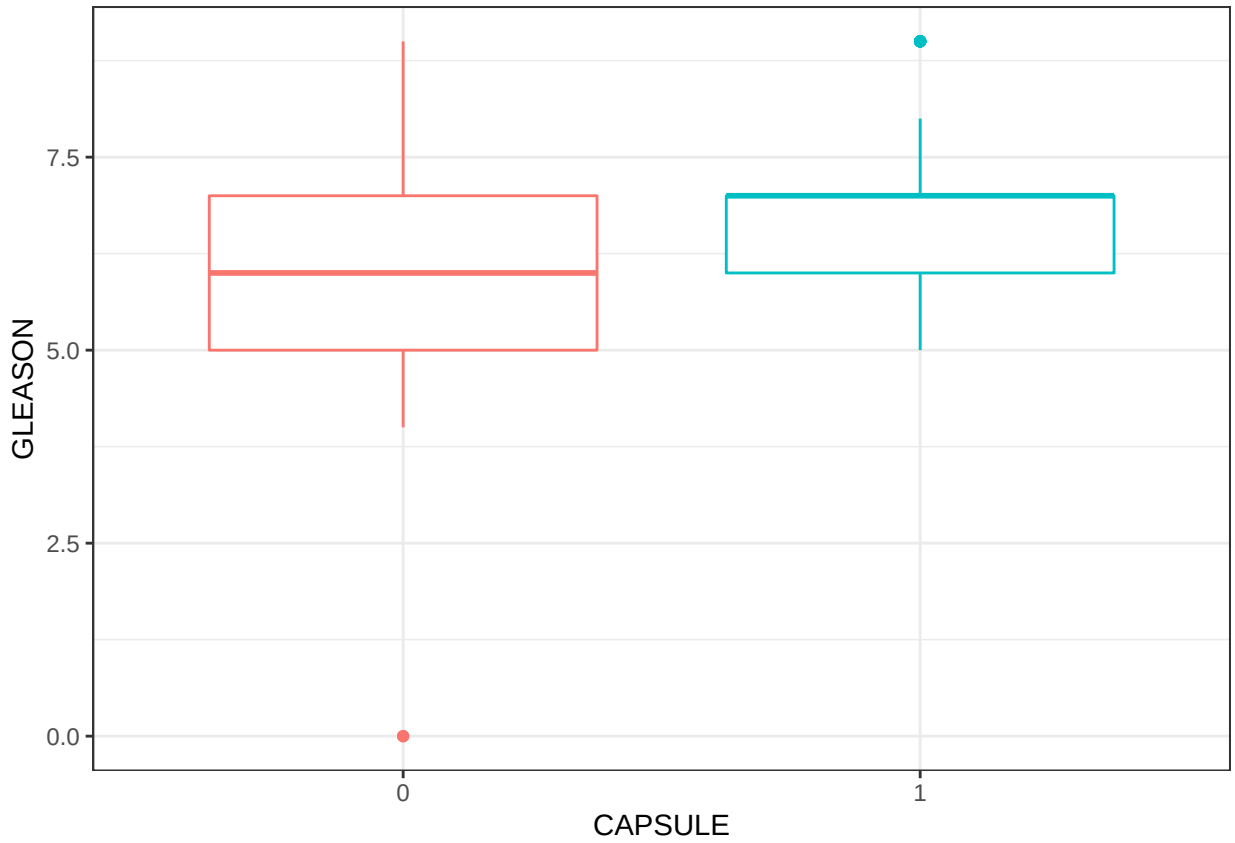
```
cor(prostata_nums)
```

```
##                AGE        PSA    GLEASON
## AGE     1.00000000 0.03082790 0.07080193
## PSA     0.03082790 1.00000000 0.02572054
## GLEASON 0.07080193 0.02572054 1.00000000
```

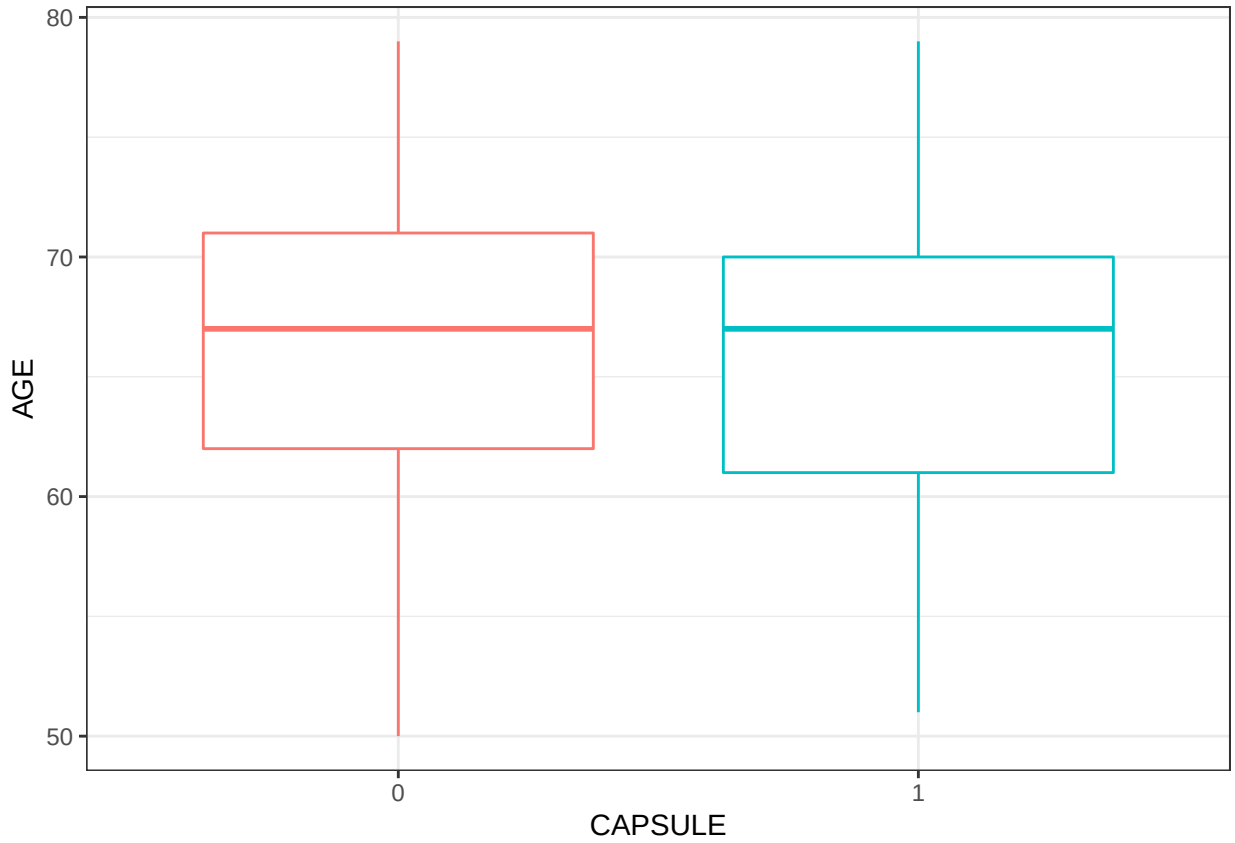
```
ggplot(data = prostata, mapping = aes(x = CAPSULE, y = PSA, colour = CAPSULE)) +
  geom_boxplot() + theme_bw() + theme(legend.position =  "none")
```




```
ggplot(data = prostata, mapping = aes(x = CAPSULE, y = GLEASON, colour = CAPSULE)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```




```
ggplot(data = prostata, mapping = aes(x = CAPSULE, y = AGE, colour = CAPSULE)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```




```
xtabs(~CAPSULE+RACE, data=prostata)
```

```
##         RACE
## CAPSULE   1   2   3   4
##       0 168  15   0   0
##       1 109   9   0   0
```

```
xtabs(~CAPSULE+DPROS, data=prostata)  # Examen prostatico ( 1 no hay, 2 unico izq, 3 unico der ,4  ambo
```

```
##         DPROS
## CAPSULE  1  2  3  4
##       0 67 63 39 14
##       1 15 34 39 30
```

```
xtabs(~CAPSULE+DCAPS, data=prostata)  # deteccion de envoltura ( 1 si 2 no)
```

```
##         DCAPS
## CAPSULE   1   2 3.8  33 67.1
##       0 175   8   0   0    0
##       1  93  25   0   0    0
```

```
prostata_glm = glm(data = prostata.train_test[[1]], CAPSULE~., family = "binomial" )
summary(prostata_glm)
```

```
##
## Call:
## glm(formula = CAPSULE ~ ., family = "binomial", data = prostata.train_test[[1]])
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -2.3798  -0.7602  -0.4513   0.8126   2.3840
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -8.887984   2.162108  -4.111 3.94e-05 ***
## AGE         -0.001273   0.025736  -0.049 0.960538
## RACE2       -0.680163   0.693367  -0.981 0.326614
## DPROS2       0.827009   0.453439   1.824 0.068174 .
## DPROS3       1.442971   0.463914   3.110 0.001868 **
## DPROS4       1.906669   0.556647   3.425 0.000614 ***
## DCAPS2       0.857878   0.575029   1.492 0.135729
## PSA          0.005274   0.002668   1.977 0.048055 *
## GLEASON      1.057477   0.209793   5.041 4.64e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 318.49  on 240  degrees of freedom
## Residual deviance: 238.87  on 232  degrees of freedom
## AIC: 256.87
##
## Number of Fisher Scoring iterations: 5
```

```
prostata.pred_test<-predict(prostata_glm,newdata=prostata.train_test[[2]], type="response")
prostata.pred_test = ifelse(test=prostata.pred_test > 0.5,yes=1,no=0)
confusion_matrix=confusionMatrix(prostata.train_test[[2]]$CAPSULE,as.factor(prostata.pred_test))
confusion_matrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 28  4
##          1 11 17
##
##                Accuracy : 0.75
##                  95% CI : (0.6214, 0.8528)
##     No Information Rate : 0.65
##     P-Value [Acc > NIR] : 0.06561
##
##                   Kappa : 0.4898
##
##  Mcnemar's Test P-Value : 0.12134
##
##             Sensitivity : 0.7179
##             Specificity : 0.8095
##          Pos Pred Value : 0.8750
##          Neg Pred Value : 0.6071
##              Prevalence : 0.6500
##          Detection Rate : 0.4667
##    Detection Prevalence : 0.5333
##       Balanced Accuracy : 0.7637
##
```

```
##        'Positive' Class : 0
##
```

```
# No deseable tener falsos negativos
prostata.pred_test<-predict(prostata_glm,newdata=prostata.train_test[[2]], type="response")
prostata.pred_test = ifelse(test=prostata.pred_test > 0.10,yes=1, no=0)
confusion_matrix=confusionMatrix(as.factor(prostata.pred_test),prostata.train_test[[2]]$CAPSULE)
confusion_matrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 11  1
##          1 21 27
##
##                Accuracy : 0.6333
##                  95% CI : (0.499, 0.7541)
##     No Information Rate : 0.5333
##     P-Value [Acc > NIR] : 0.0766
##
##                   Kappa : 0.2949
##
##  Mcnemar's Test P-Value : 5.104e-05
##
##             Sensitivity : 0.3438
##             Specificity : 0.9643
##          Pos Pred Value : 0.9167
##          Neg Pred Value : 0.5625
##              Prevalence : 0.5333
##          Detection Rate : 0.1833
##    Detection Prevalence : 0.2000
##       Balanced Accuracy : 0.6540
##
##        'Positive' Class : 0
##
```

```
# Este algoritmo permite encontrar una fórmula más simple sin perder mucha bondad de clasificación

prostata_stepwise_glm <- prostata_glm %>%  stepAIC(trace=FALSE)
summary(prostata_stepwise_glm)
```

```
##
## Call:
## glm(formula = CAPSULE ~ DPROS + DCAPS + PSA + GLEASON, family = "binomial",
##     data = prostata.train_test[[1]])
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3667  -0.7632  -0.4557   0.8304   2.4030
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -8.970648   1.453700  -6.171 6.79e-10 ***
## DPROS2       0.805258   0.448201   1.797 0.072392 .
## DPROS3       1.436978   0.458164   3.136 0.001710 **
```

```
## DPROS4       1.871990   0.550113   3.403 0.000667 ***
## DCAPS2       0.802444   0.566945   1.415 0.156956
## PSA          0.004967   0.002634   1.886 0.059362 .
## GLEASON      1.059278   0.206690   5.125 2.98e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 318.49  on 240  degrees of freedom
## Residual deviance: 239.88  on 234  degrees of freedom
## AIC: 253.88
##
## Number of Fisher Scoring iterations: 5
```

```
prostata.pred_test<-predict(prostata_stepwise_glm,newdata=prostata.train_test[[2]], type="response")
prostata.pred_test = ifelse(test=prostata.pred_test > 0.11,yes=1,no=0)
confusion_matrix=confusionMatrix(as.factor(prostata.pred_test),prostata.train_test[[2]]$CAPSULE)
confusion_matrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 13  1
##          1 19 27
##
##                Accuracy : 0.6667
##                  95% CI : (0.5331, 0.7831)
##     No Information Rate : 0.5333
##     P-Value [Acc > NIR] : 0.0251862
##
##                   Kappa : 0.3562
##
##  Mcnemar's Test P-Value : 0.0001439
##
##             Sensitivity : 0.4062
##             Specificity : 0.9643
##          Pos Pred Value : 0.9286
##          Neg Pred Value : 0.5870
##              Prevalence : 0.5333
##          Detection Rate : 0.2167
##    Detection Prevalence : 0.2333
##       Balanced Accuracy : 0.6853
##
##        'Positive' Class : 0
##
```

```
prostata_svm=svm(CAPSULE~ .,
              data=prostata.train_test[[1]],
              method="C-Classification",
              kernel="sigmoid",
              cost=100,
              gamma=1/nrow(prostata.train_test[[1]]))
svm_pred=predict(prostata_svm, newdata =  prostata.train_test[[2]])
```

```
confusion_matrix=confusionMatrix(as.factor(svm_pred),prostata.train_test[[2]]$CAPSULE)
confusion_matrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 28 12
##          1  4 16
##
##                Accuracy : 0.7333
##                  95% CI : (0.6034, 0.8393)
##     No Information Rate : 0.5333
##     P-Value [Acc > NIR] : 0.001201
##
##                   Kappa : 0.4545
##
##  Mcnemar's Test P-Value : 0.080118
##
##             Sensitivity : 0.8750
##             Specificity : 0.5714
##          Pos Pred Value : 0.7000
##          Neg Pred Value : 0.8000
##              Prevalence : 0.5333
##          Detection Rate : 0.4667
##    Detection Prevalence : 0.6667
##       Balanced Accuracy : 0.7232
##
##        'Positive' Class : 0
##
```

```
prostata_test.numeric= prostata.train_test[[2]][,nums]
prostata_test.PCA = prcomp(prostata_test.numeric,center = TRUE, scale. = TRUE)
ggbiplot(prostata_test.PCA,obs.scale = 1,var.scale = 1,alpha = 0.5, groups = interaction(prostata.pred_
```

```
## ------------------------------------------------------------------------------
```

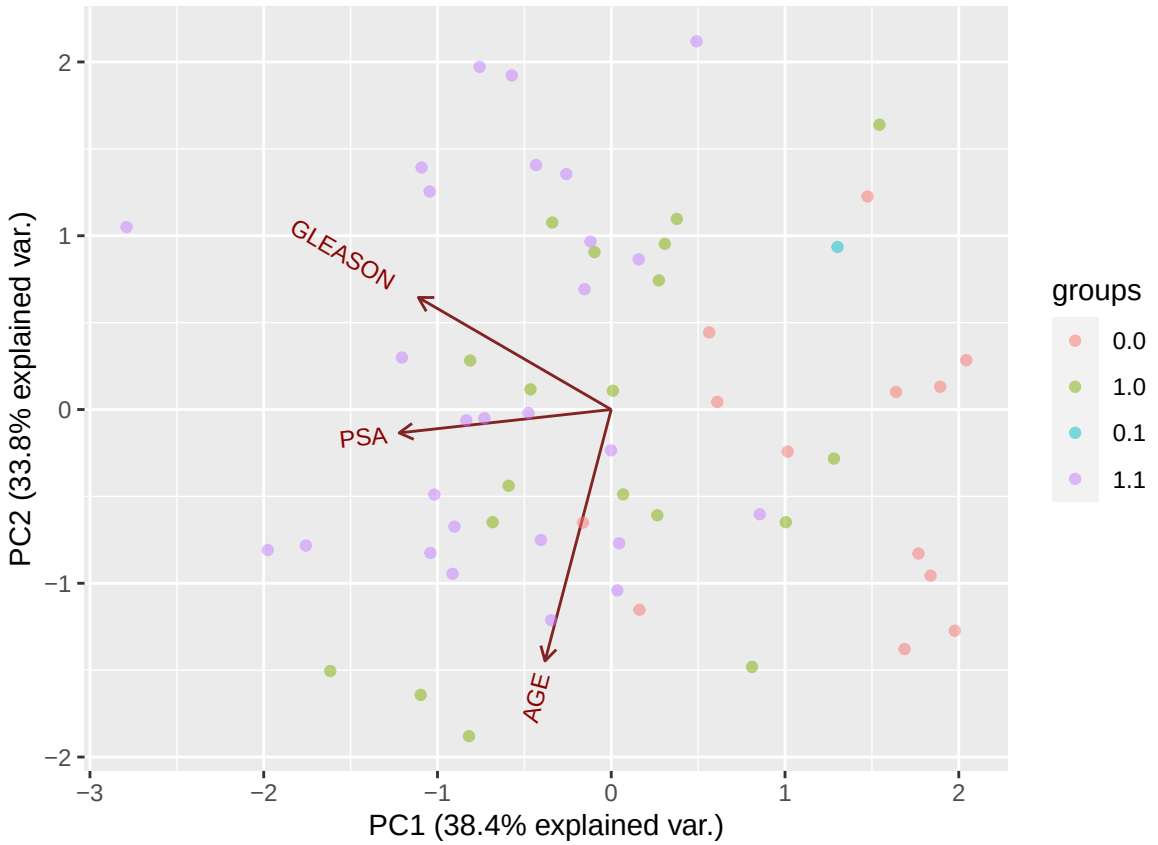
```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
```

```
## ------------------------------------------------------------------------------
```

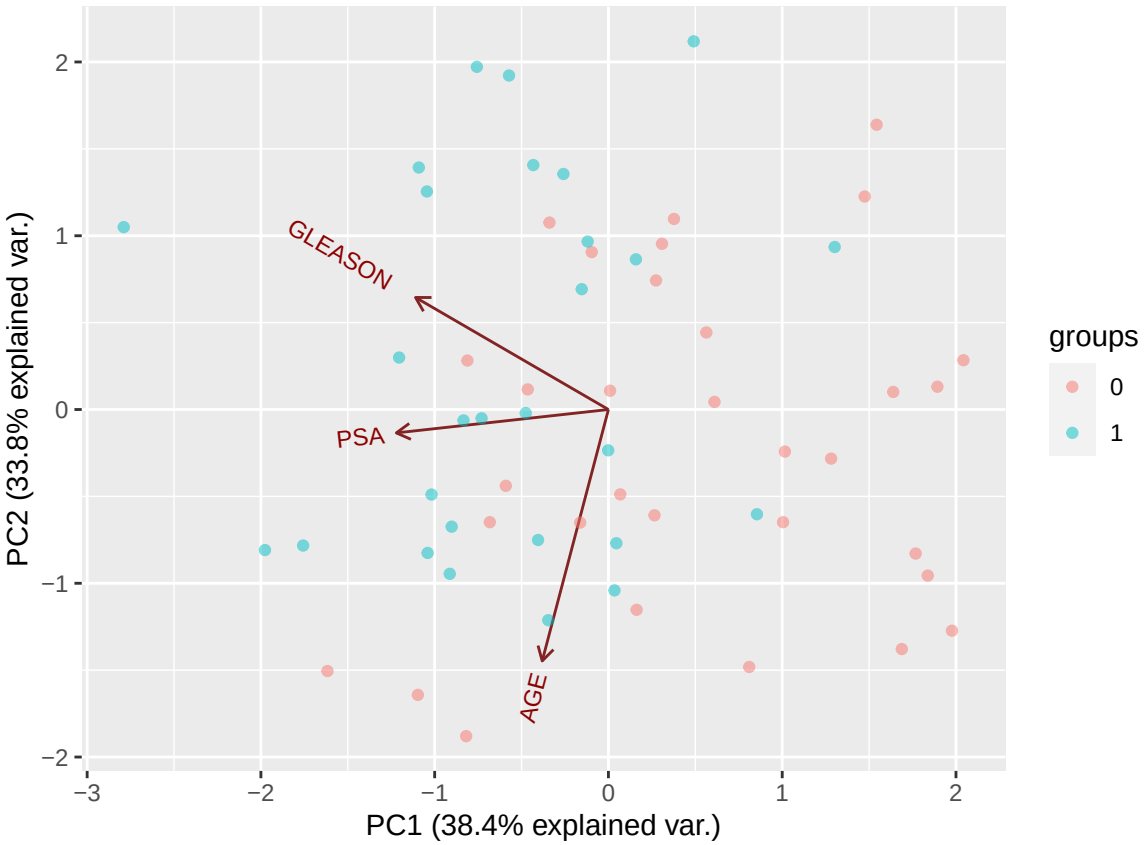
```
##
## Attaching package: 'plyr'
```

```
## The following object is masked from 'package:DMwR':
##
##     join
```

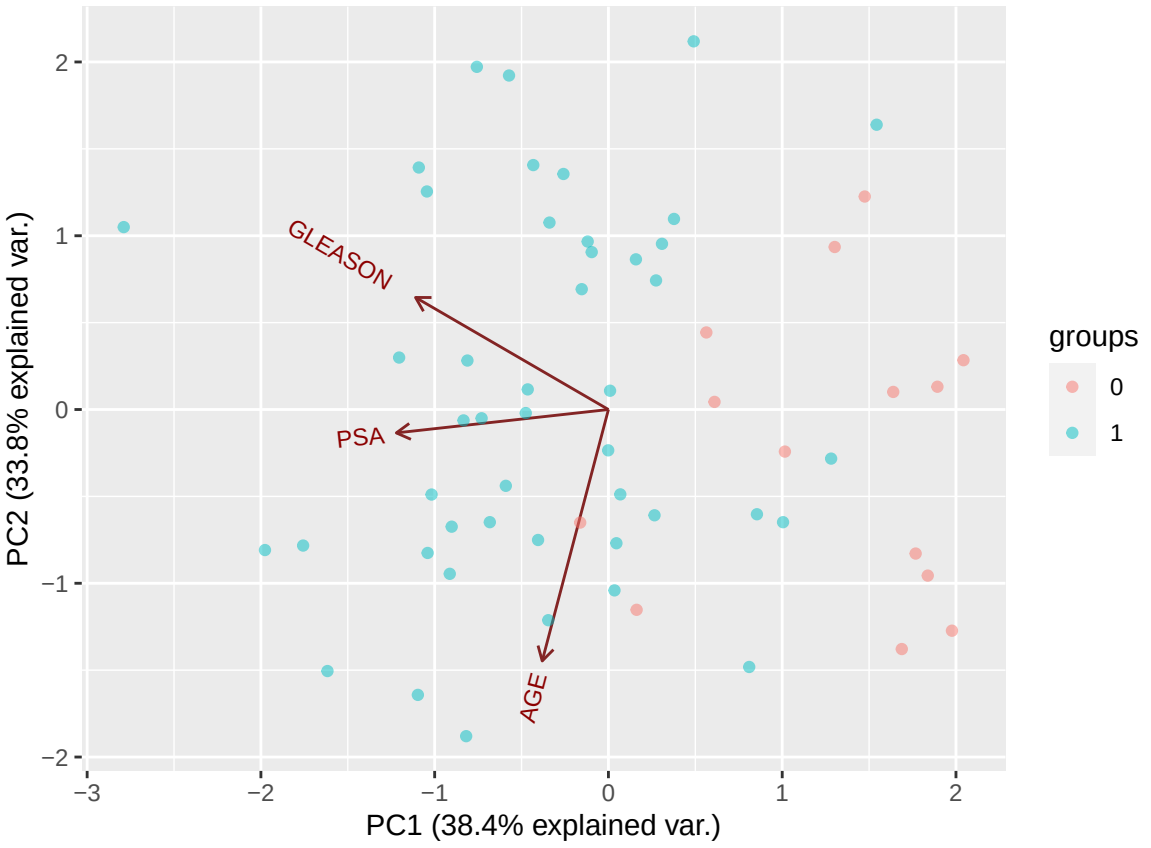
```
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```



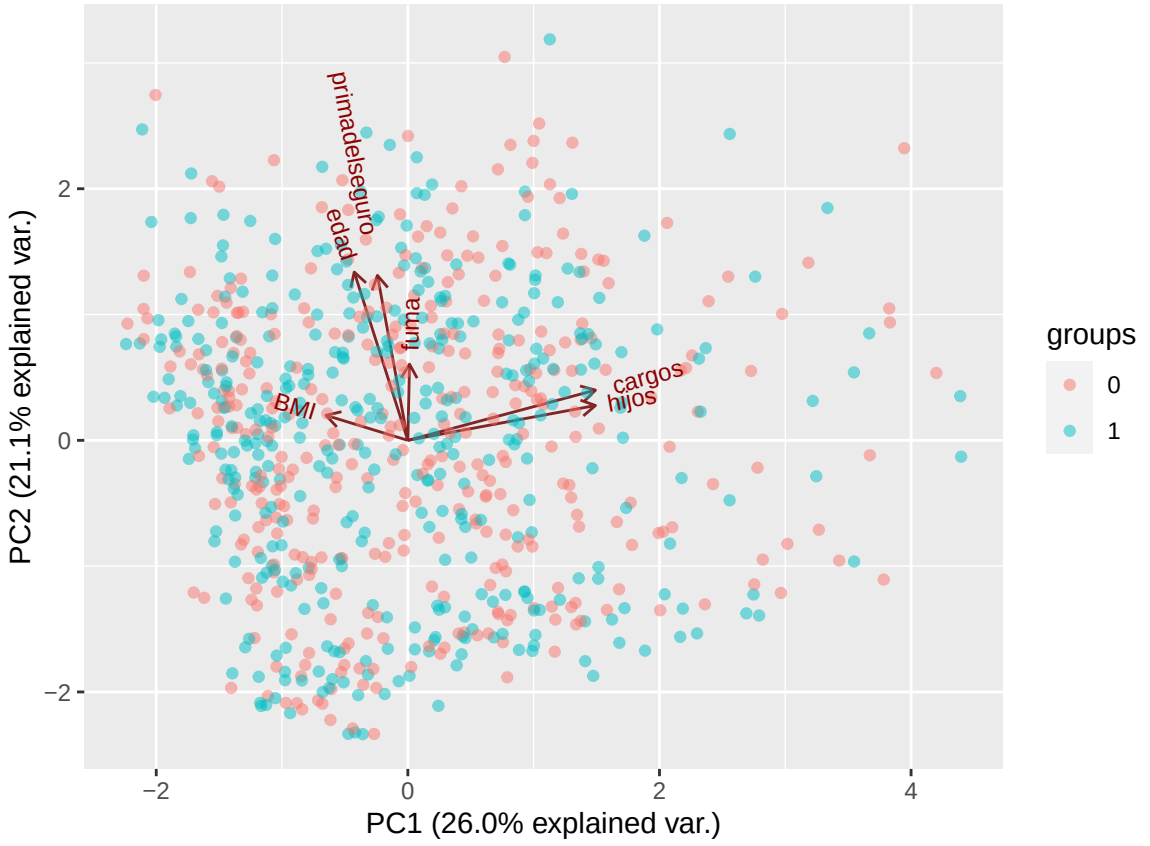

```
ggbiplot(prostata_test.PCA,obs.scale = 1,var.scale = 1,alpha = 0.5, groups = interaction(prostata.train
```



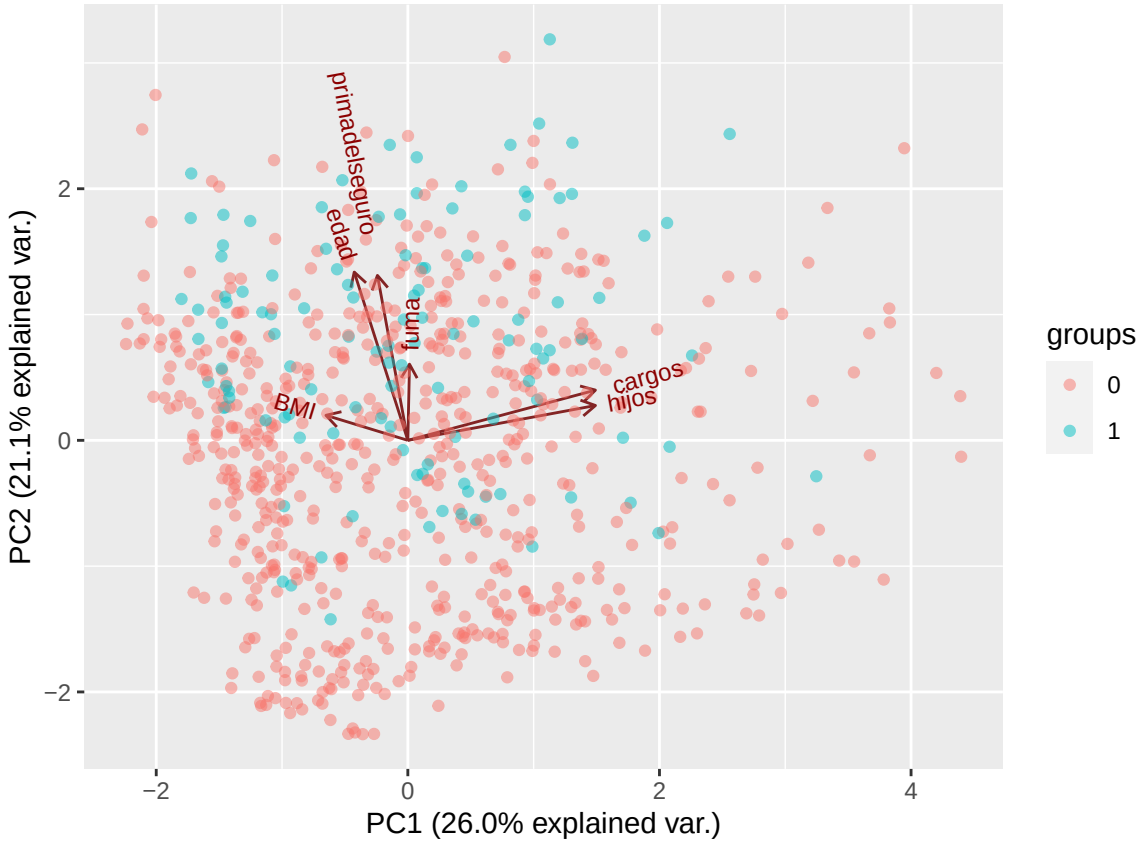

```
ggbiplot(prostata_test.PCA,obs.scale = 1,var.scale = 1,alpha = 0.5, groups = interaction(prostata.pred_
```



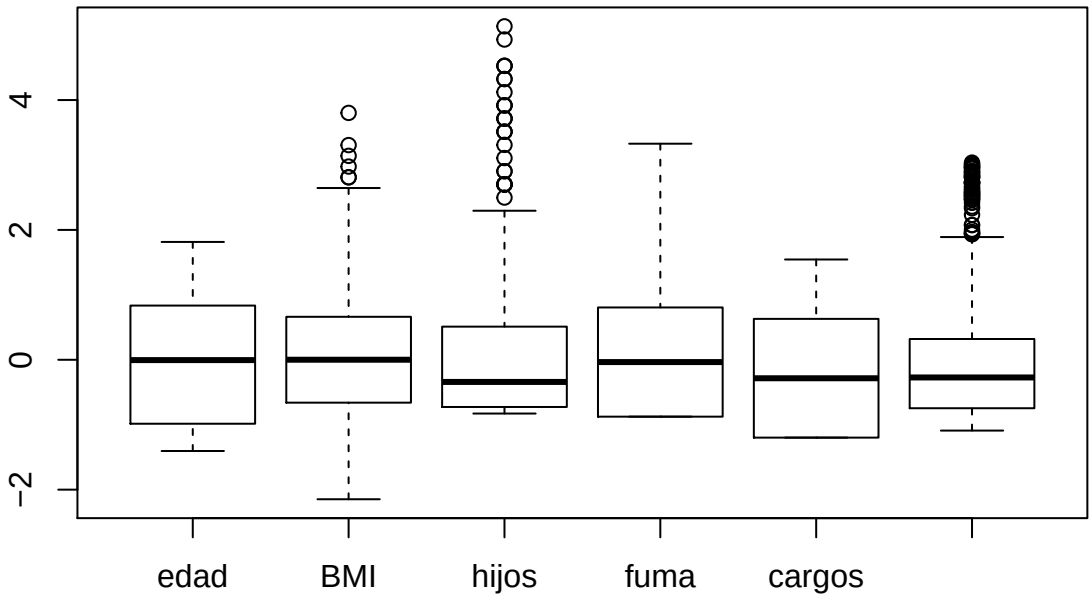

```
seguros = read.csv(file = "./Seguros.csv", dec = ",")
seguros = sampleo(seguros,0.75)
seguros = seguros_Datatypes(seguros)
seguros = seguros[seguros$hijos < 300 & seguros$primadelseguro < 39000,]
nums <- unlist(lapply(seguros, is.numeric))
seguros.numeric = seguros[,nums]
seguros.factor = seguros[,!nums]
seguros.pca = prcomp(seguros.numeric,scale = TRUE)
ggbiplot(seguros.pca, obs.scale=1 ,var.scale=1, alpha=0.5,groups = as.factor(seguros$sexo) )
```



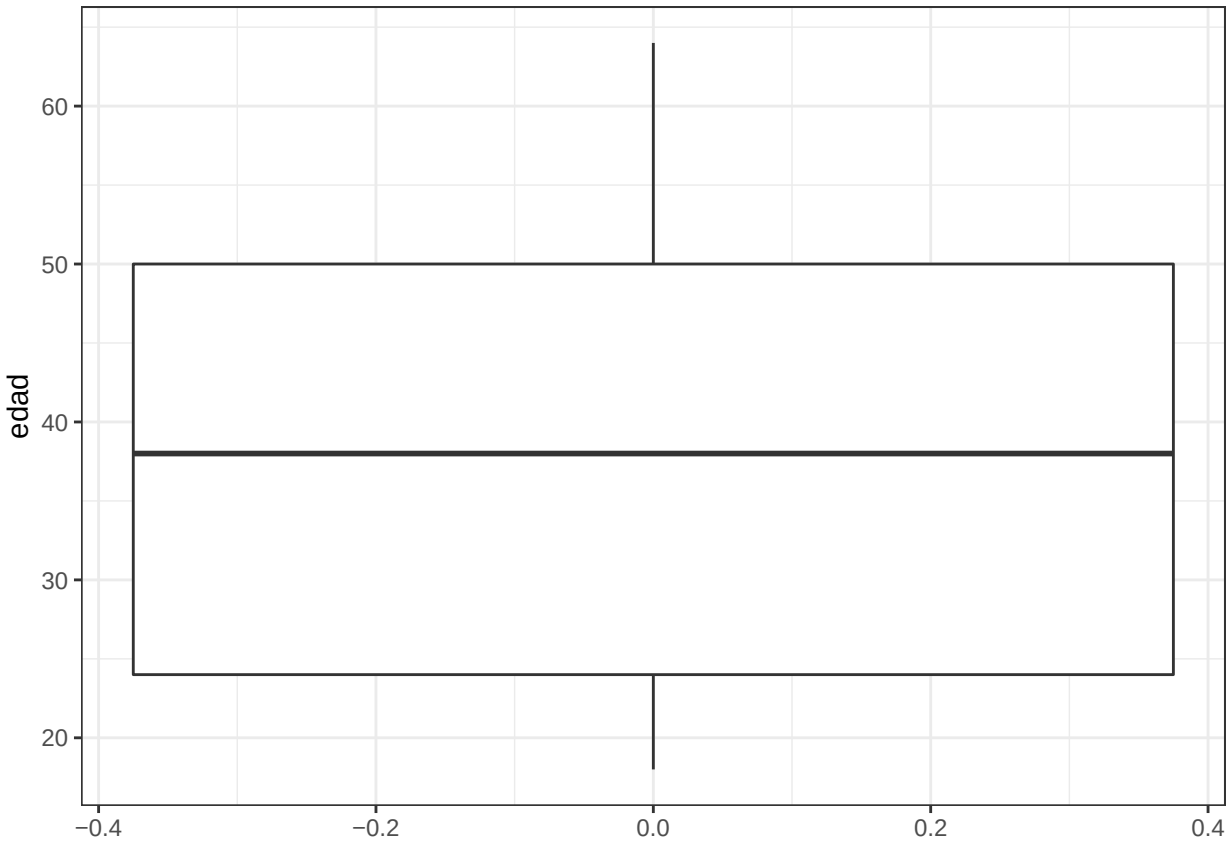

```
ggbiplot(seguros.pca, obs.scale=1 ,var.scale=1, alpha=0.5,groups = as.factor(seguros$region) )
```



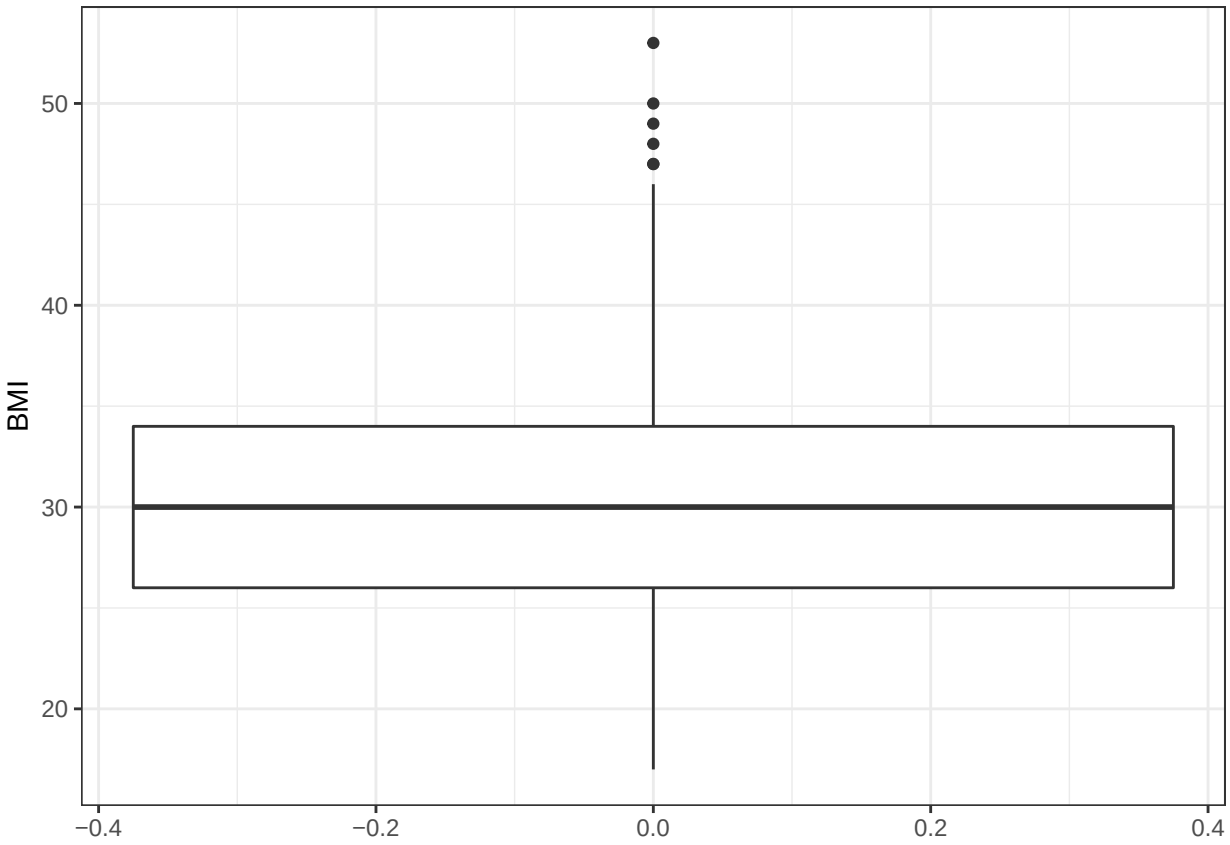

```
boxplot(scale(seguros.numeric))
```



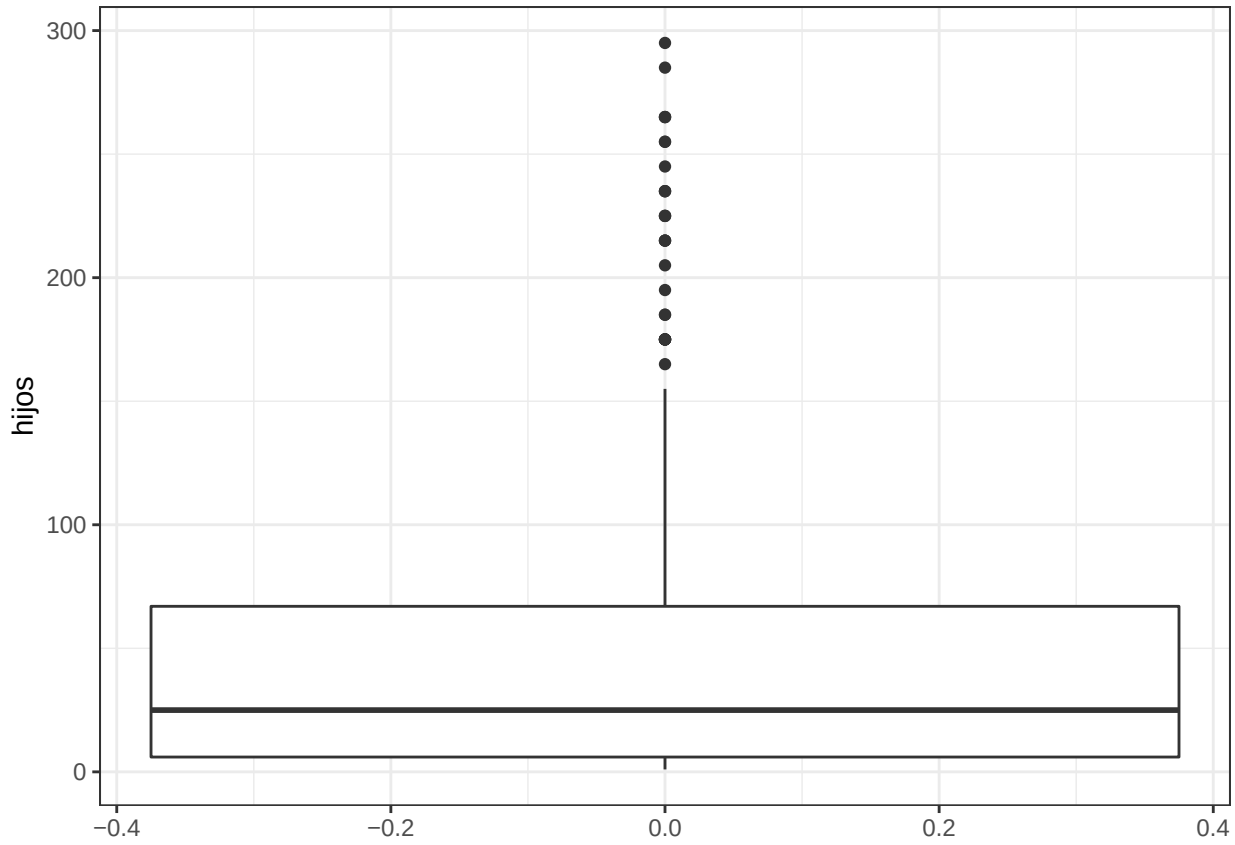

```
ggplot(data = seguros, mapping = aes( y = edad)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```



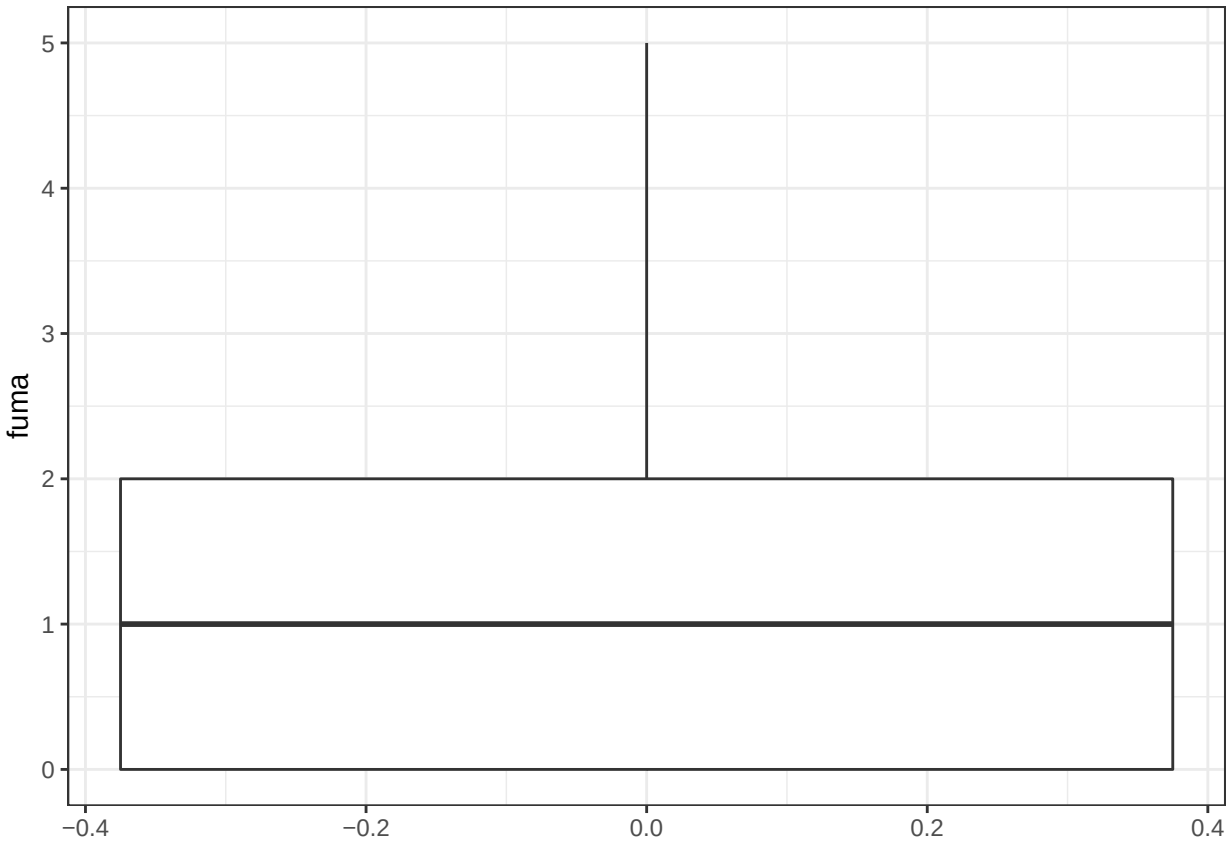

```
ggplot(data = seguros, mapping = aes( y = BMI)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```



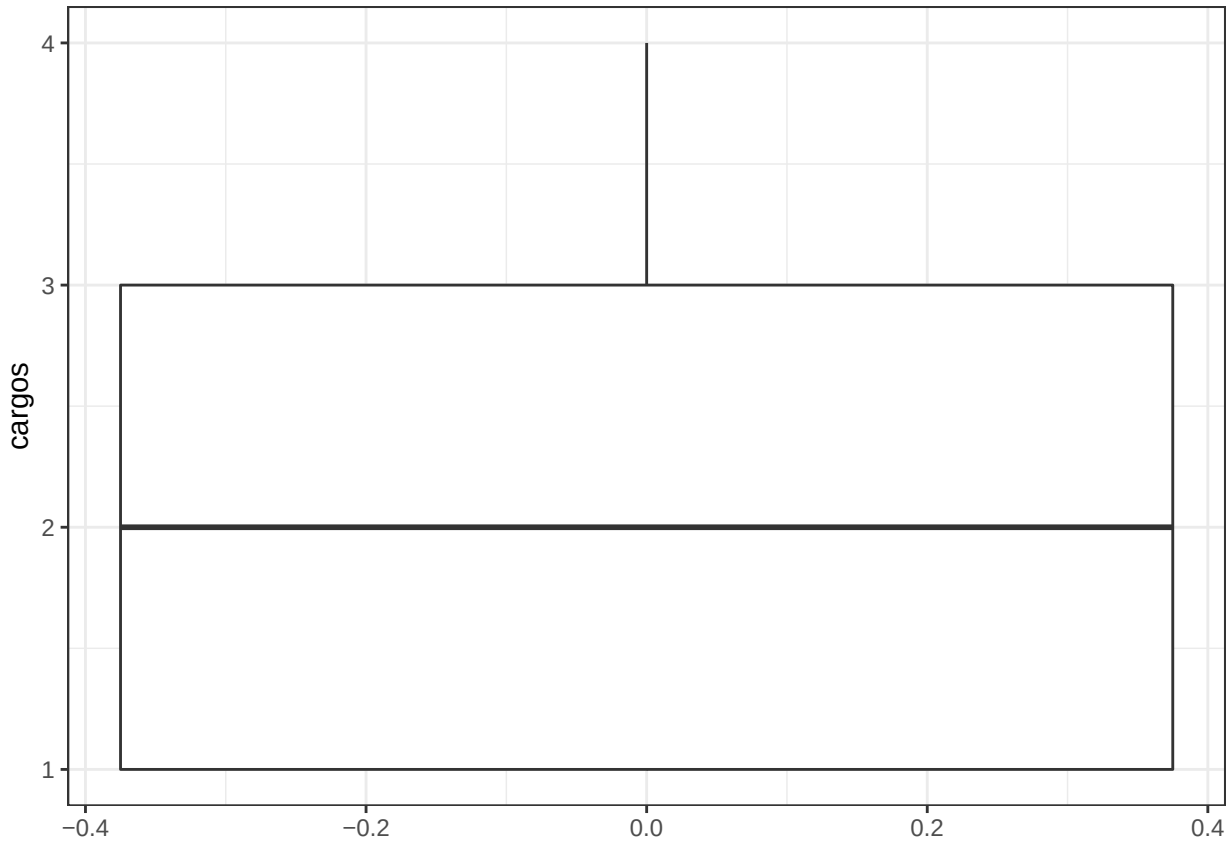

```
ggplot(data = seguros, mapping = aes( y = hijos)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```



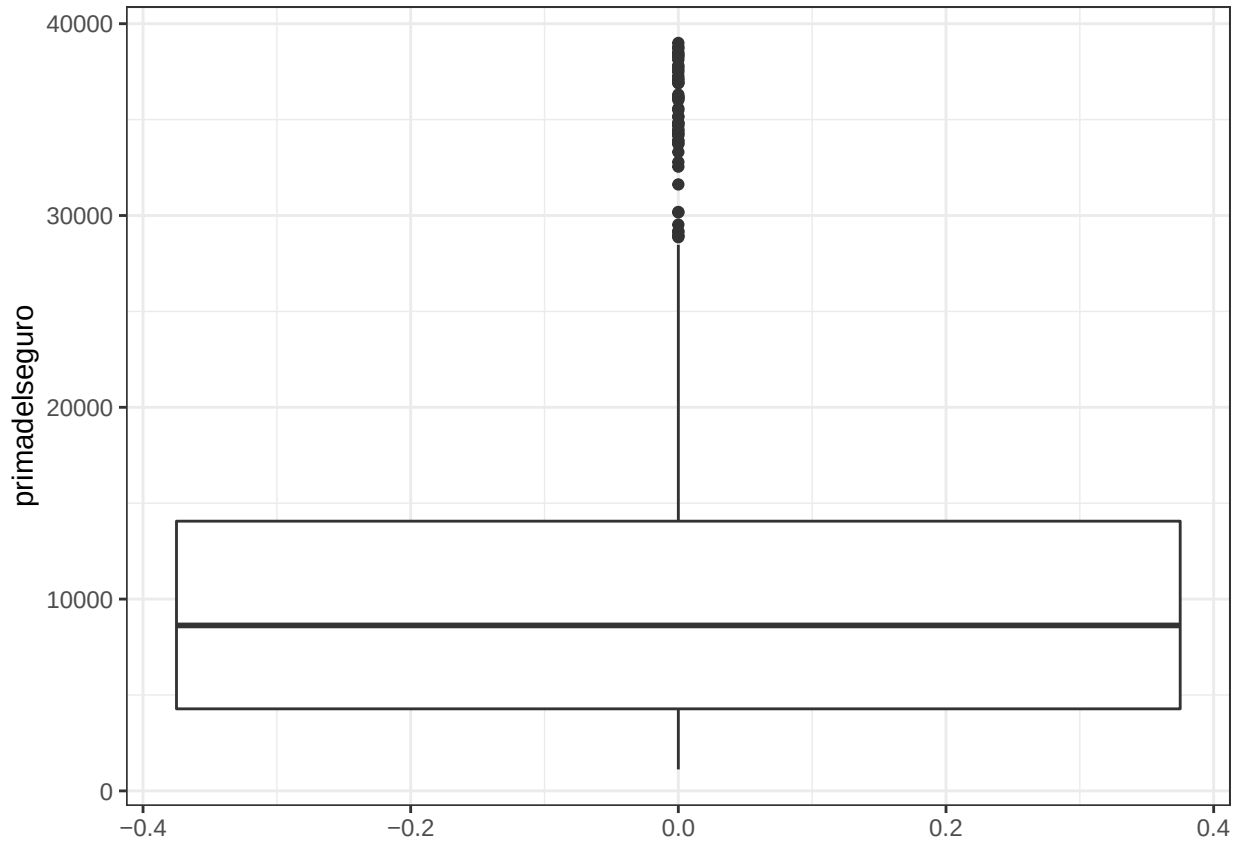

```
ggplot(data = seguros, mapping = aes( y = fuma)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```



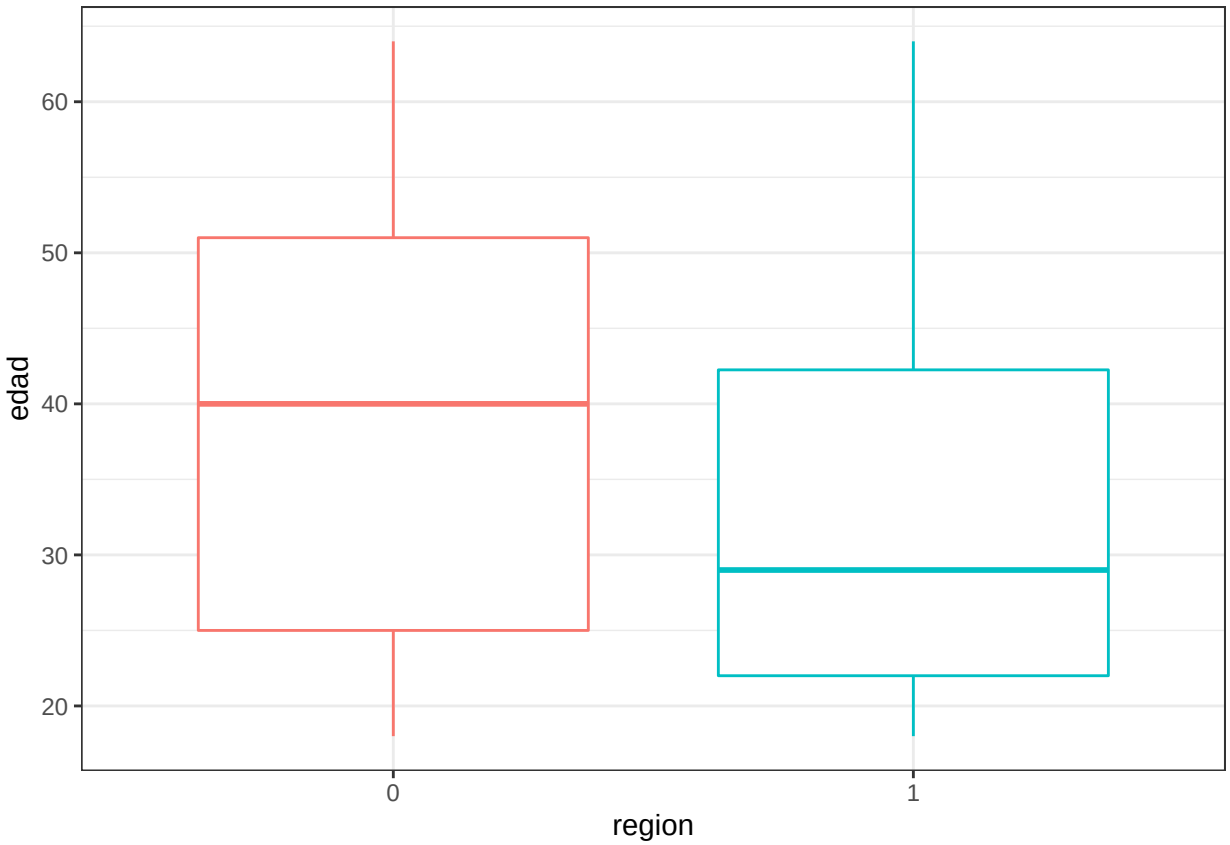

```
ggplot(data = seguros, mapping = aes( y = cargos)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```



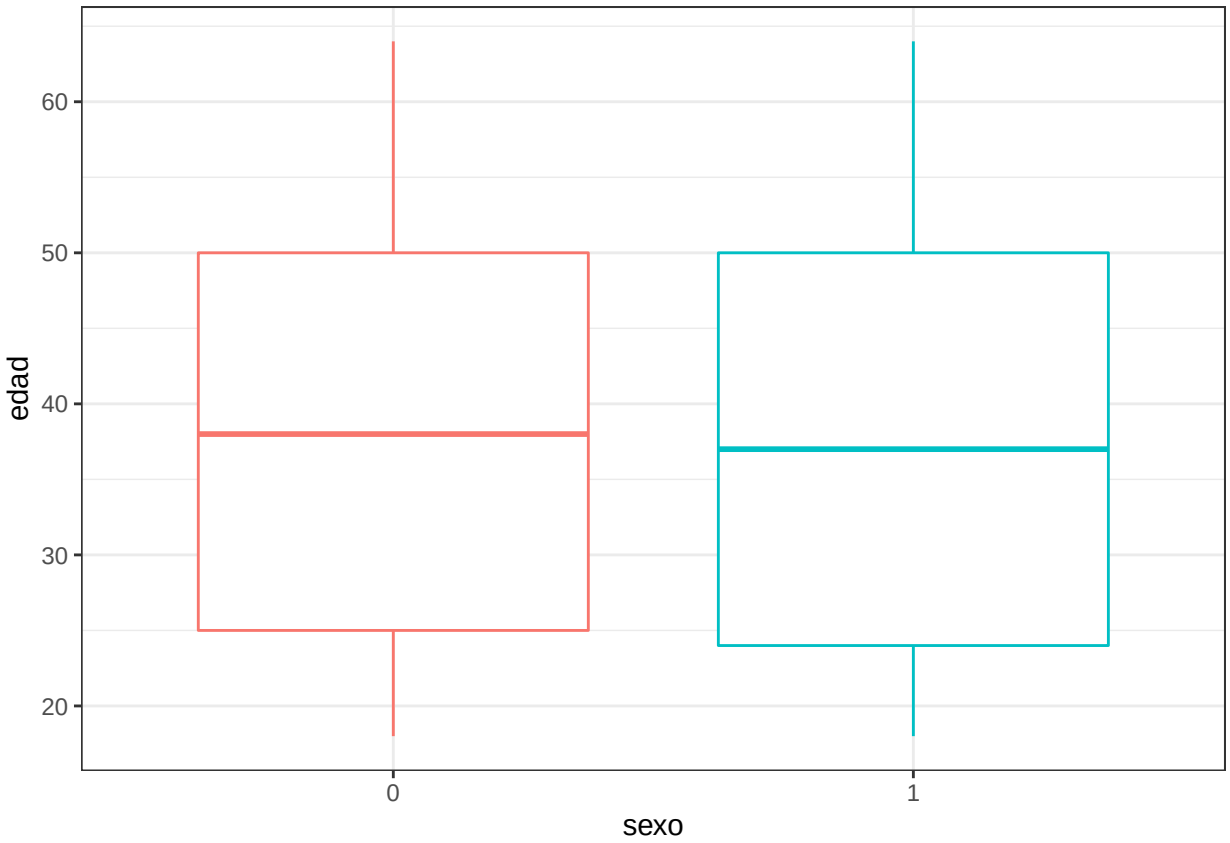

```
ggplot(data = seguros, mapping = aes( y = primadelseguro)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```



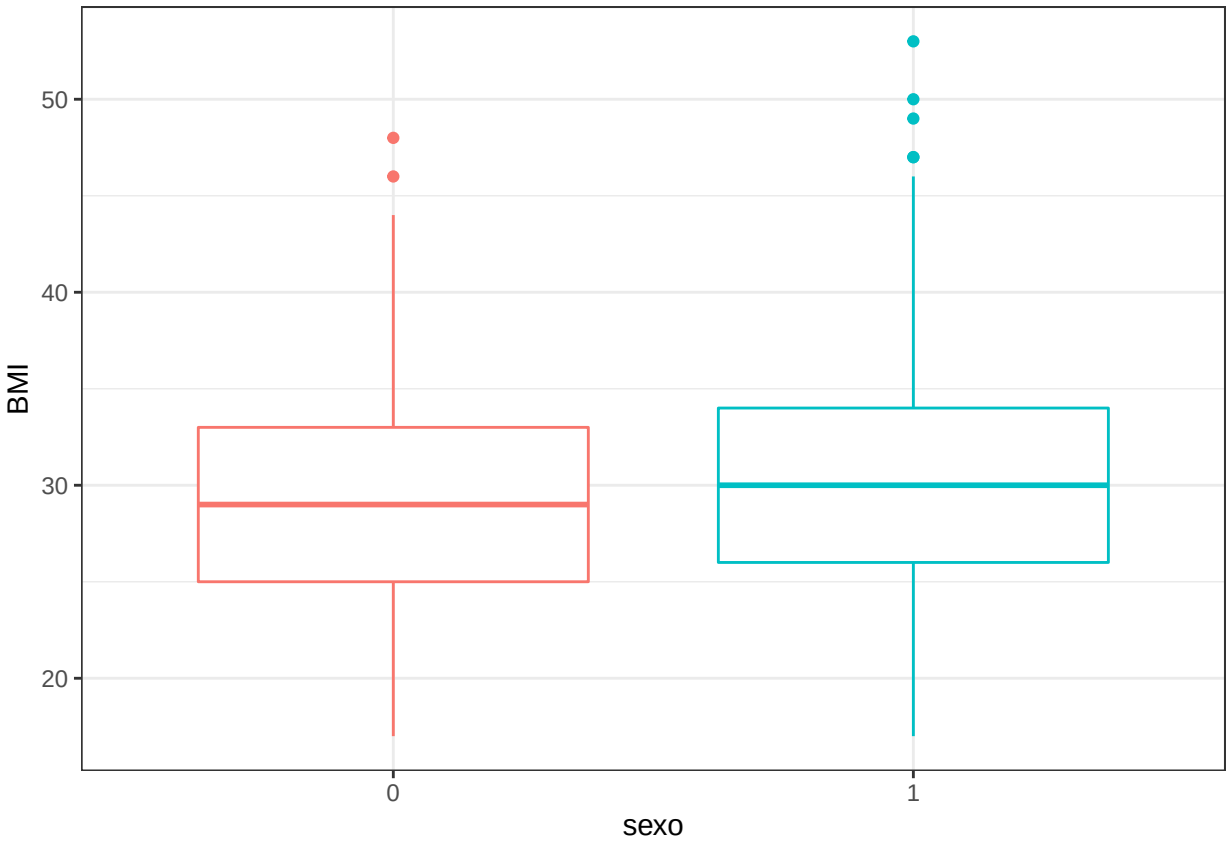

```
ggplot(data = seguros, mapping = aes(x = region, y = edad, colour = region)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```



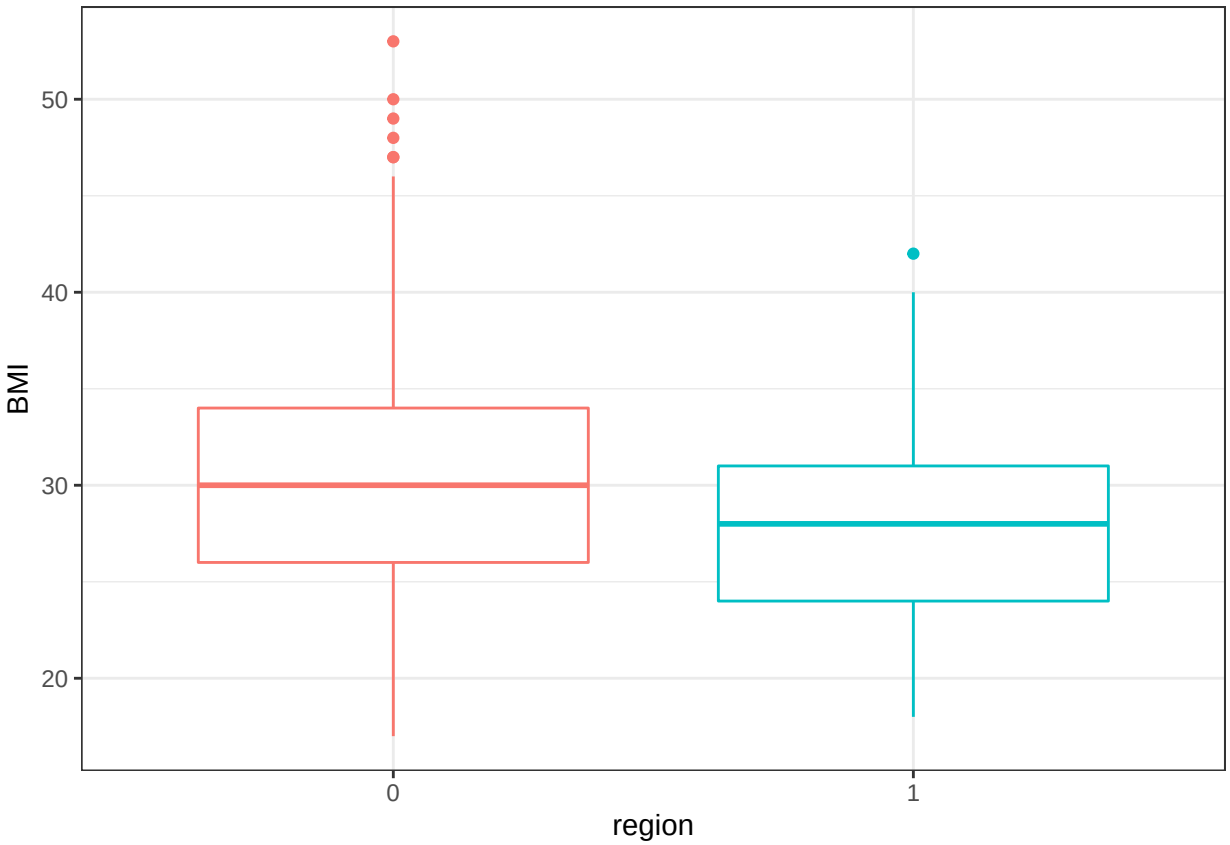

```
ggplot(data = seguros, mapping = aes(x = sexo, y = edad, colour = sexo)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```



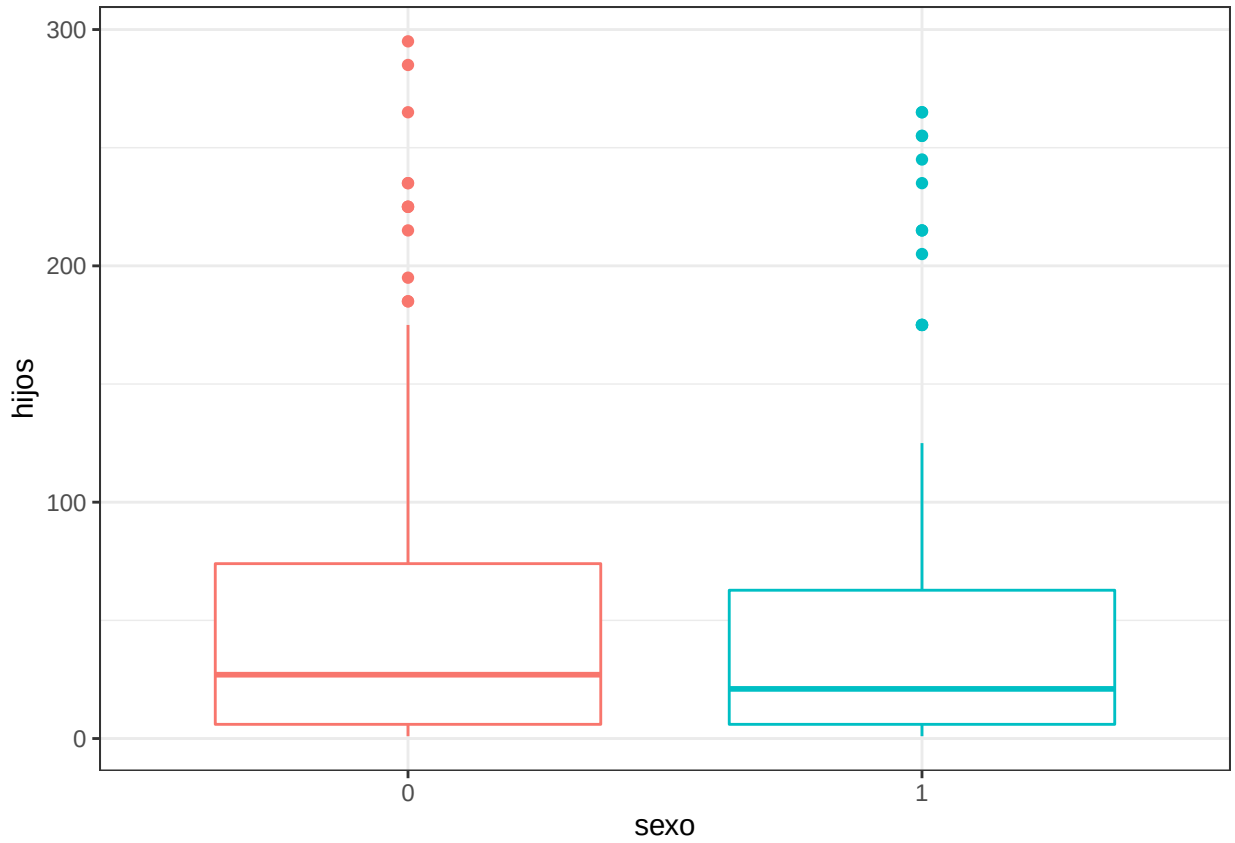

```
ggplot(data = seguros, mapping = aes(x = sexo, y = BMI, colour = sexo)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```



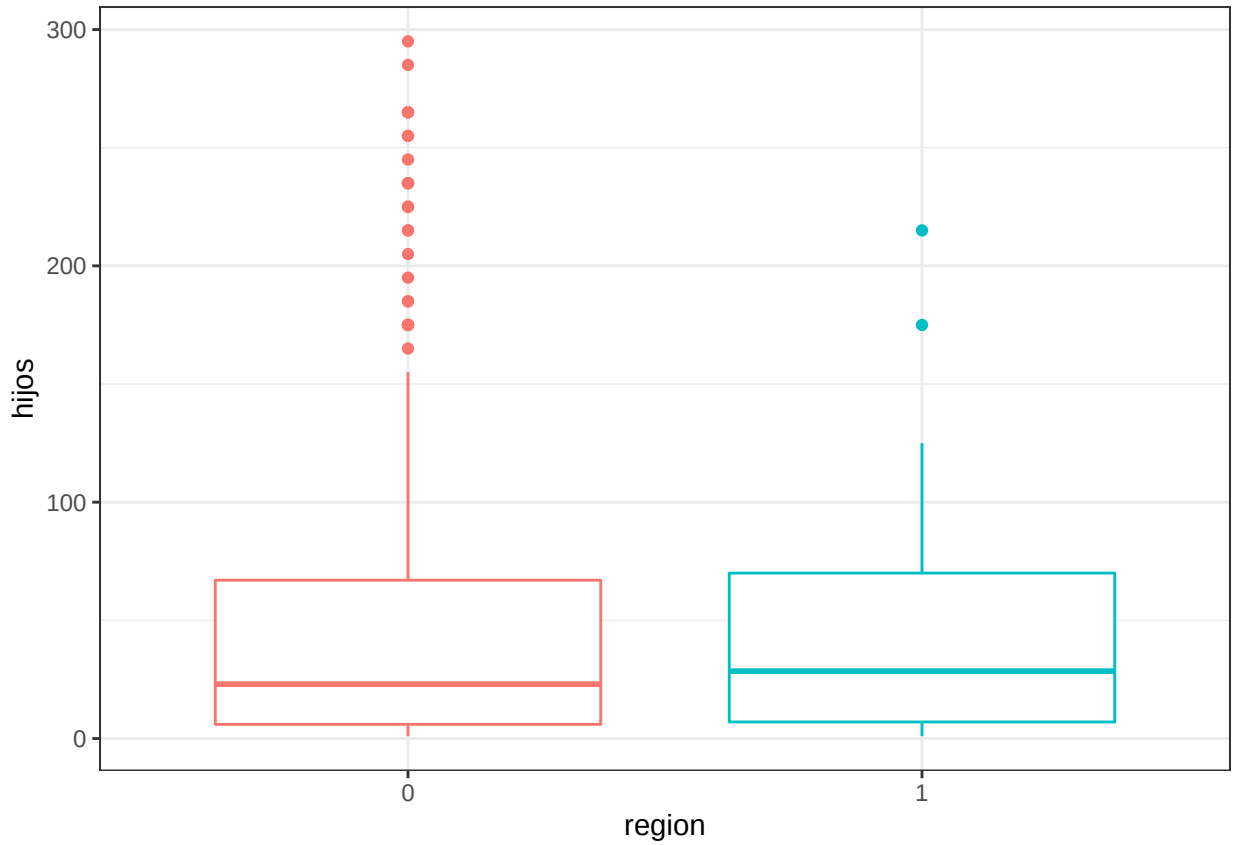

```
ggplot(data = seguros, mapping = aes(x = region, y = BMI, colour = region)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```



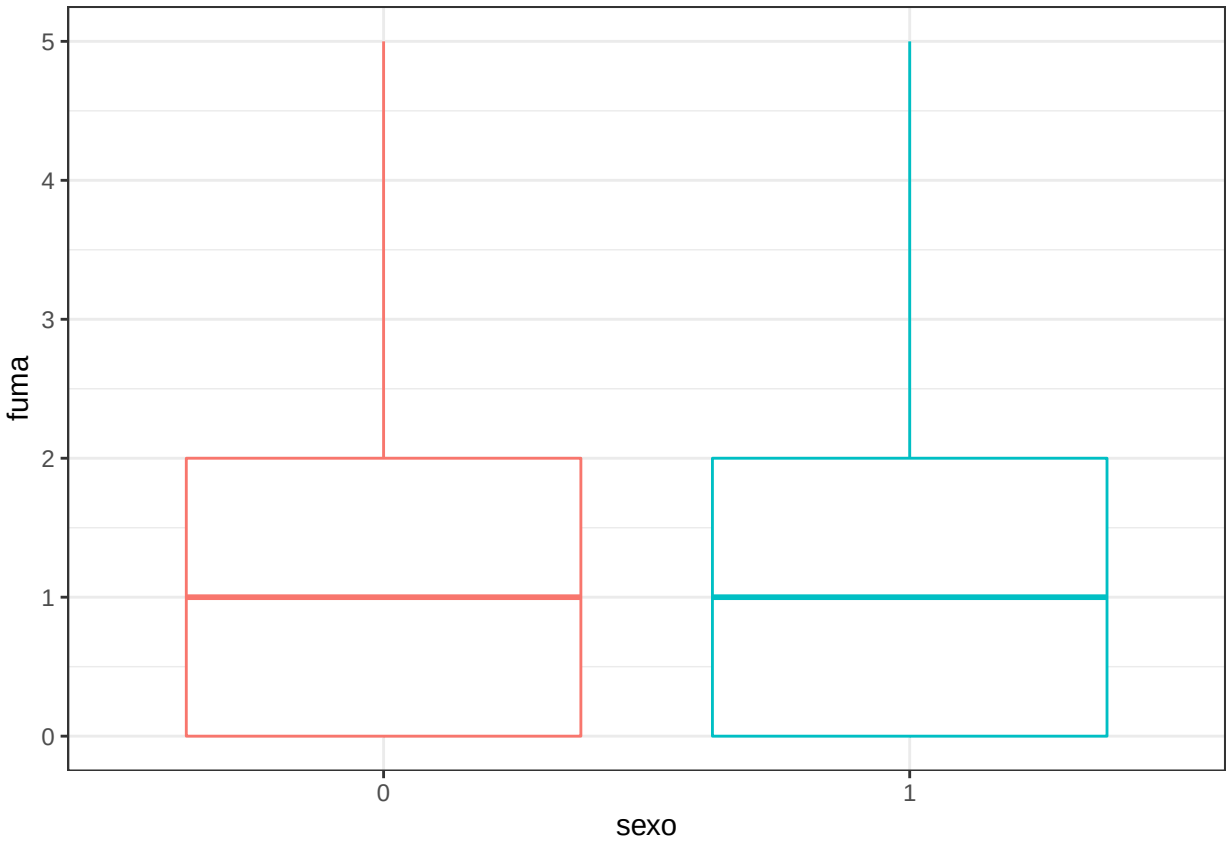

```
ggplot(data = seguros, mapping = aes(x = sexo, y = hijos, colour = sexo)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```



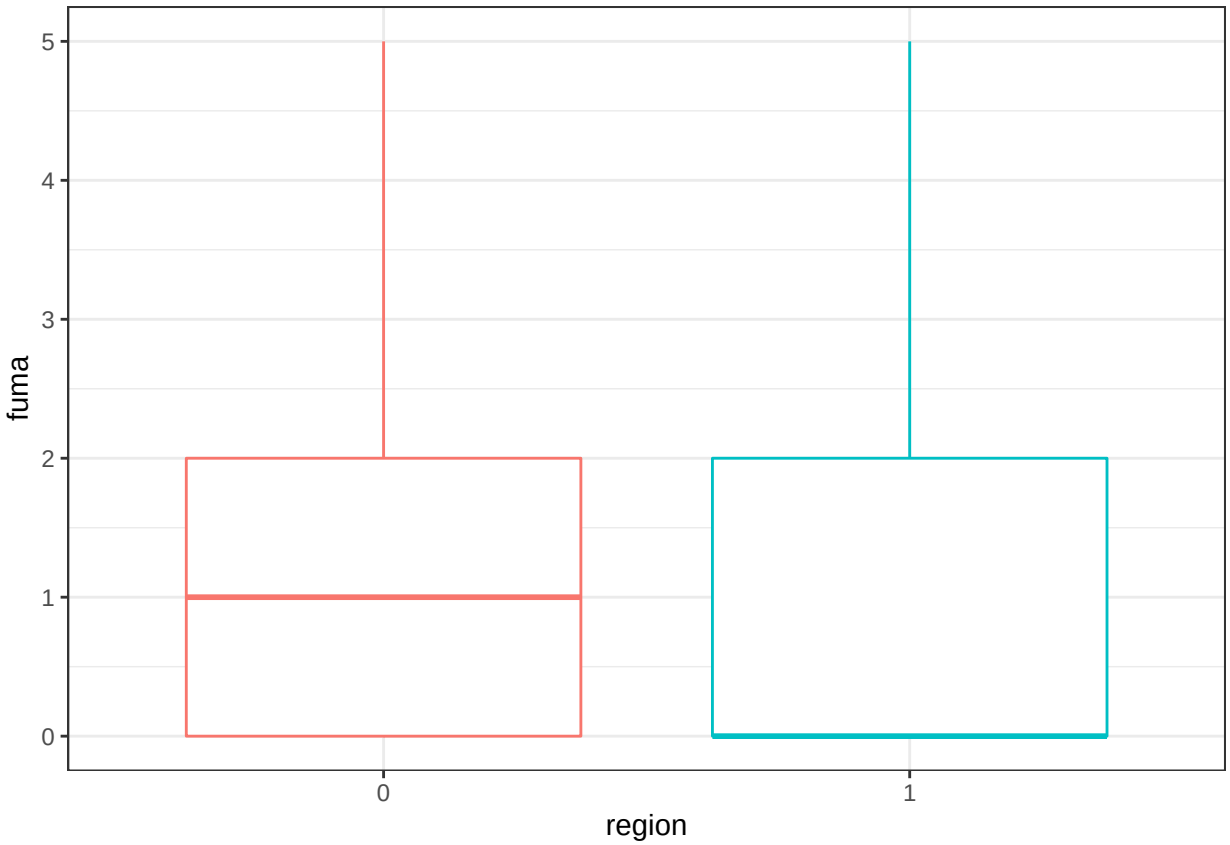

```
ggplot(data = seguros, mapping = aes(x = region, y = hijos, colour = region)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```



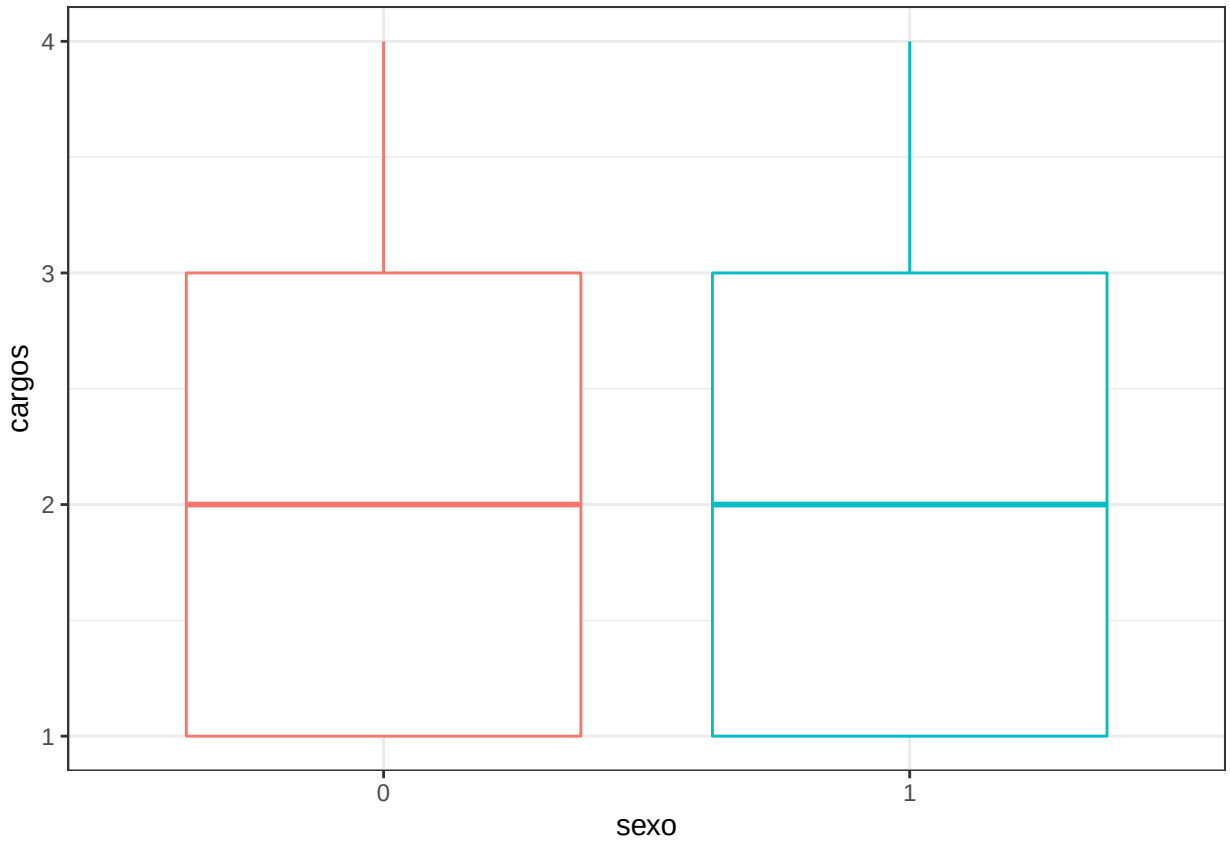

```
ggplot(data = seguros, mapping = aes(x = sexo, y = fuma, colour = sexo)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```



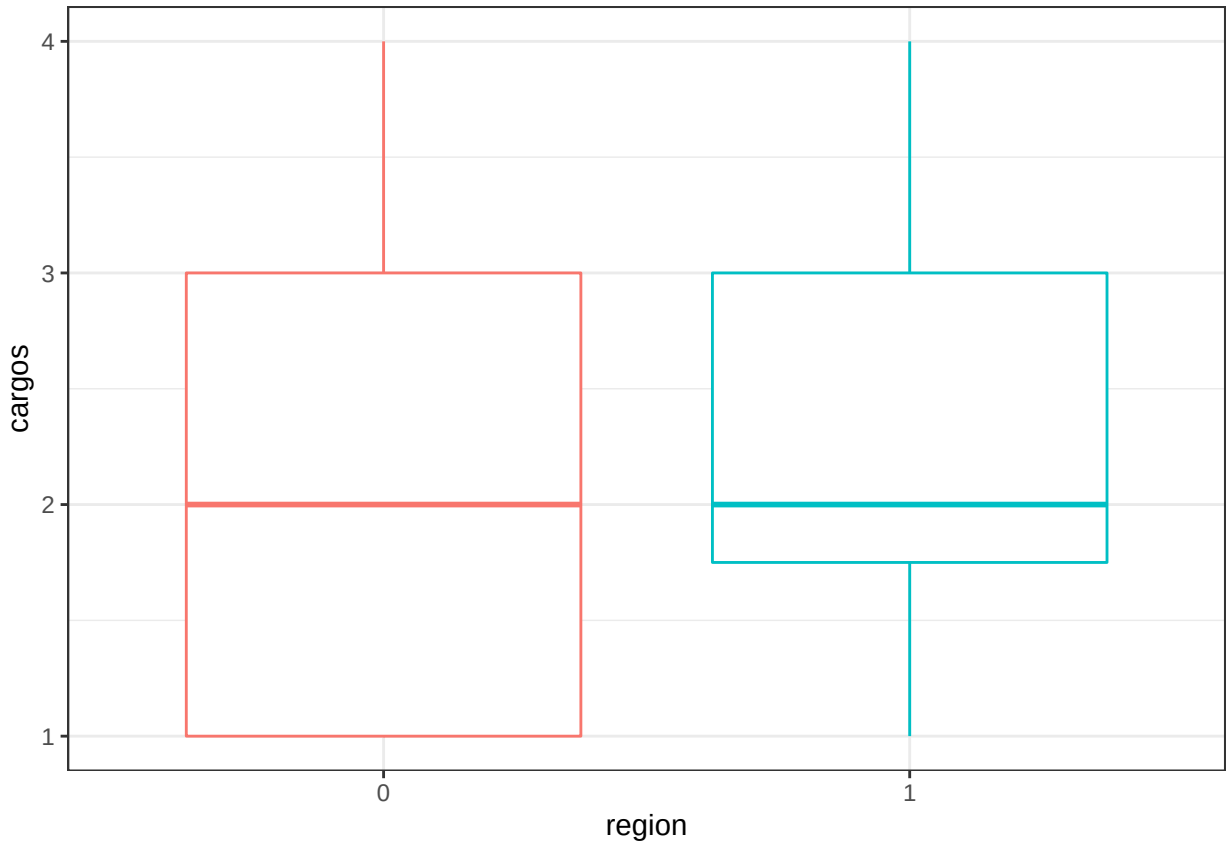

```
ggplot(data = seguros, mapping = aes(x = region, y = fuma, colour = region)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```



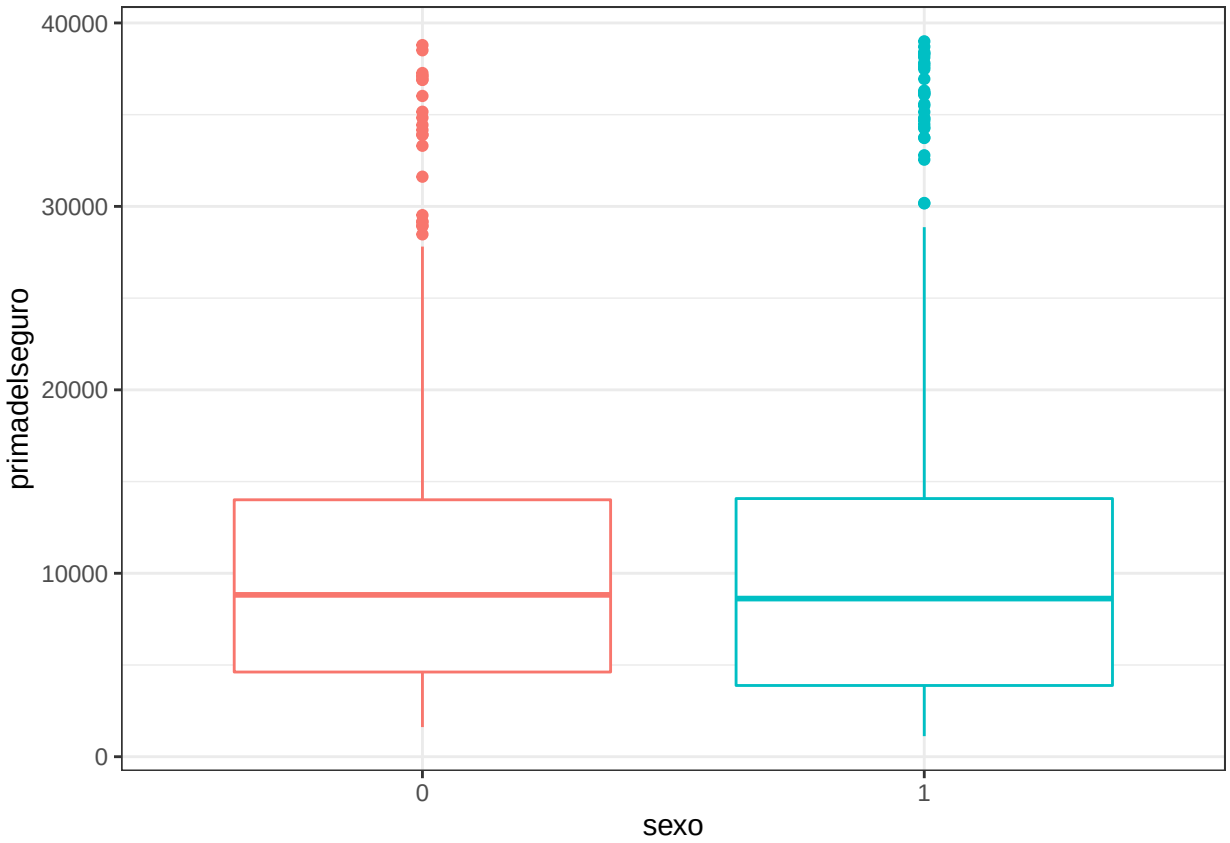

```
ggplot(data = seguros, mapping = aes(x = sexo, y = cargos, colour = sexo)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```



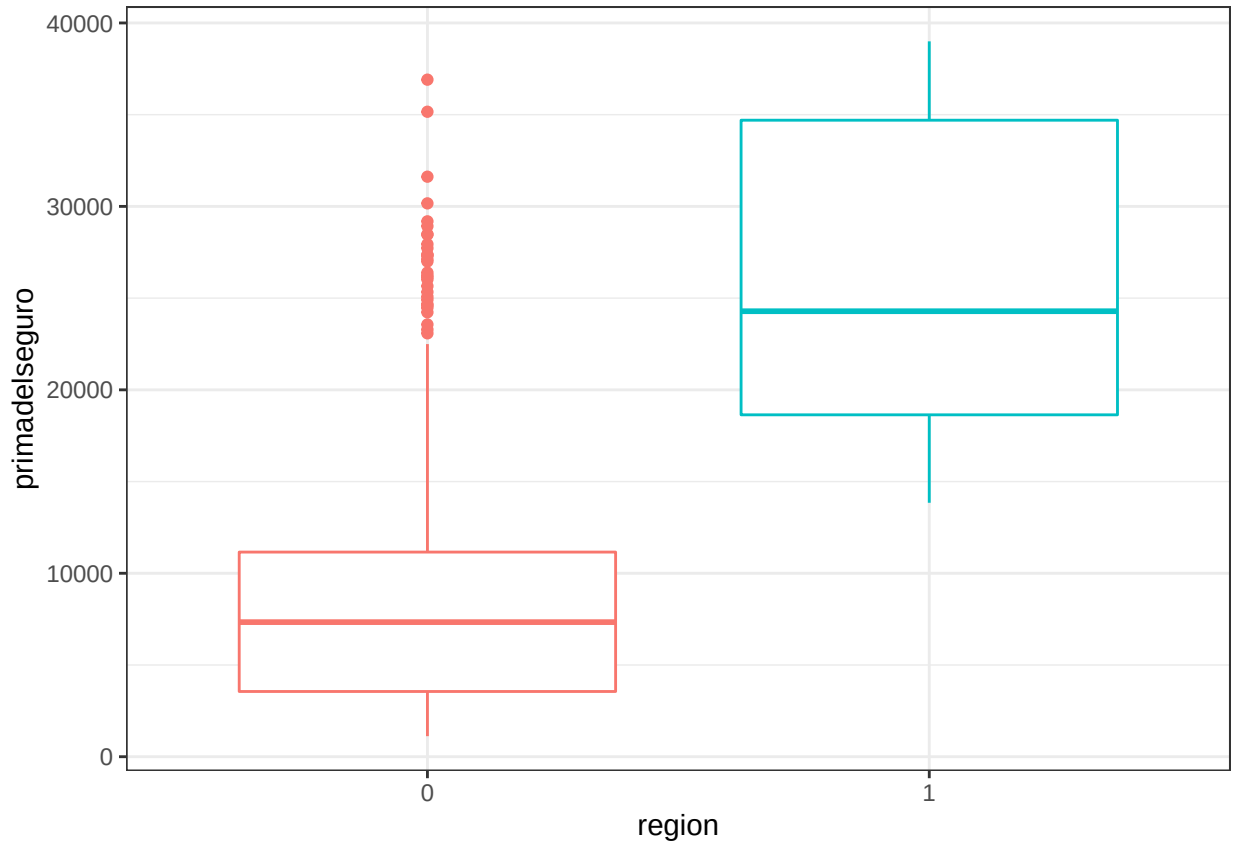

```
ggplot(data = seguros, mapping = aes(x = region, y = cargos, colour = region)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```




```
ggplot(data = seguros, mapping = aes(x = sexo, y = primadelseguro, colour = sexo)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```




```
ggplot(data = seguros, mapping = aes(x = region, y = primadelseguro, colour = region)) +
  geom_boxplot() + theme_bw() + theme(legend.position = "none")
```




```
# Correlacion en las variables numricas
round(cor(seguros.numeric),2)
```

```
##                 edad   BMI hijos  fuma cargos primadelseguro
## edad            1.00  0.08 -0.06  0.08  -0.02           0.24
## BMI             0.08  1.00 -0.08  0.01  -0.13           0.02
## hijos          -0.06 -0.08  1.00  0.05   0.50          -0.04
## fuma            0.08  0.01  0.05  1.00  -0.01           0.03
## cargos         -0.02 -0.13  0.50 -0.01   1.00           0.03
## primadelseguro  0.24  0.02 -0.04  0.03   0.03           1.00
```

```
kmax = 10
seguros.pca = prcomp(seguros.numeric,scale = TRUE)
```

```
a= fviz_nbclust(scale(seguros.numeric),hcut,method = c("silhouette"),nboot = 50,diss = dist(scale(segur
max(a$data$y)
```

```
## [1] 0.1672606
```

```
b= fviz_nbclust(scale(seguros.numeric),hcut,method = c("silhouette"),nboot = 50,diss = dist(scale(segur
max(b$data$y)
```

```
## [1] 0.1835167
```

```
c= fviz_nbclust(scale(seguros.numeric),hcut,method = c("silhouette"),nboot = 50,diss = dist(scale(segur
max(c$data$y)
```

```
## [1] 0.1427675
```

```
d=fviz_nbclust(scale(seguros.numeric),hcut,method = c("silhouette"),nboot = 50,diss = dist(scale(seguros
max(d$data$y)
```

```
## [1] 0.1672606
```

```
a= fviz_nbclust(scale(seguros.numeric),kmeans,method = c("silhouette"),nboot = 50,diss = dist(scale(segu
max(a$data$y)
```

```
## [1] 0.2035724
```

```
b= fviz_nbclust(scale(seguros.numeric),kmeans,method = c("silhouette"),nboot = 50,diss = dist(scale(segu
max(b$data$y)
```

```
## [1] 0.214996
```

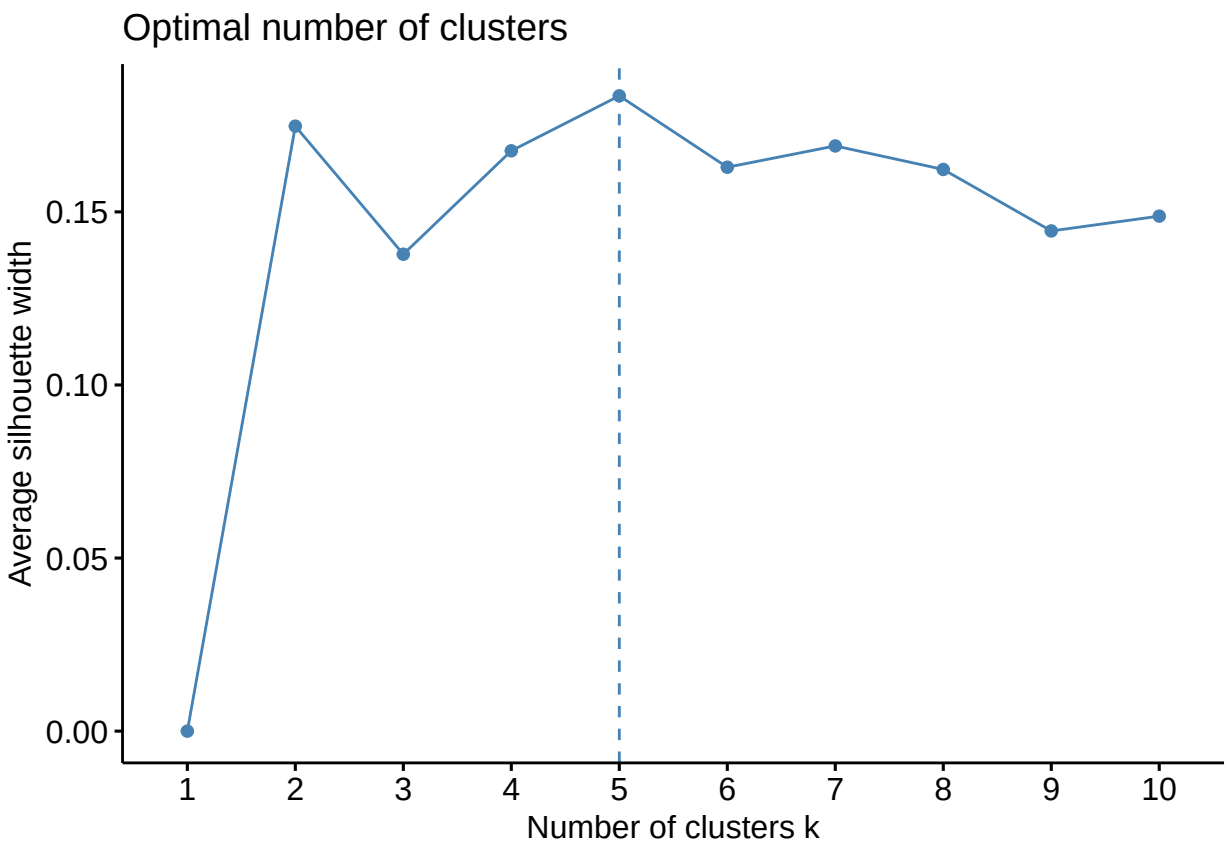
```
c= fviz_nbclust(scale(seguros.numeric),kmeans,method = c("silhouette"),nboot = 50,diss = dist(scale(segu
max(c$data$y)
```

```
## [1] 0.1962031
```

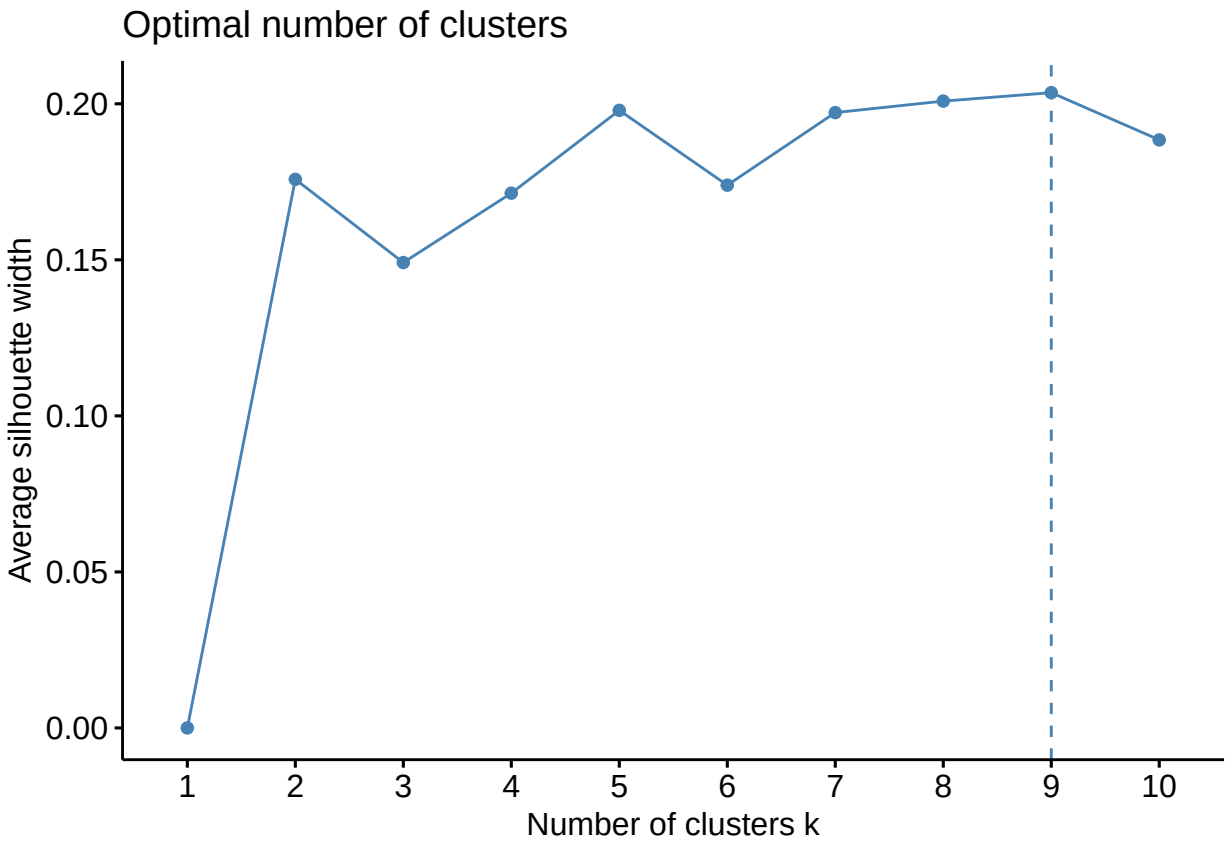
```
d=fviz_nbclust(scale(seguros.numeric),kmeans,method = c("silhouette"),nboot = 50,diss = dist(scale(segui
max(d$data$y)
```

```
## [1] 0.2035724
```

```
fviz_nbclust(scale(seguros.numeric),hcut,method = c("silhouette"),nboot = 50,diss = dist(scale(seguros.n
```

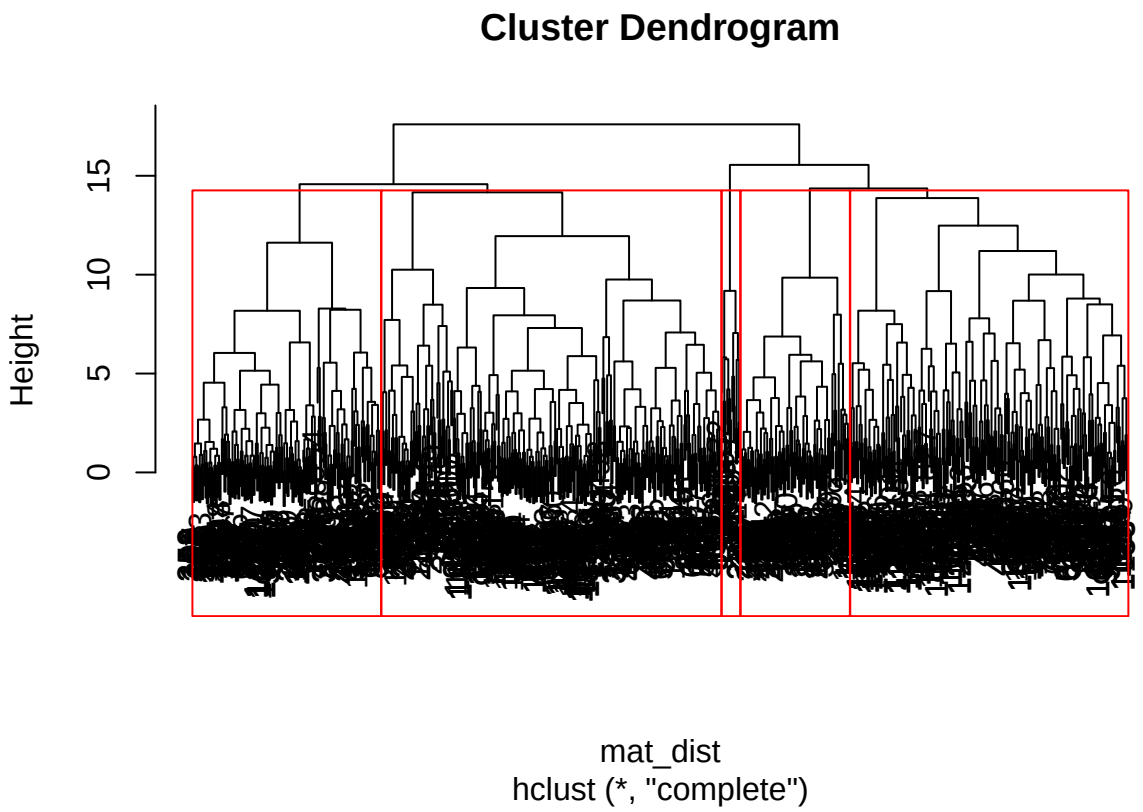



```
fviz_nbclust(scale(seguros.numeric),kmeans,method = c("silhouette"),nboot = 50,diss = dist(scale(seguros
```

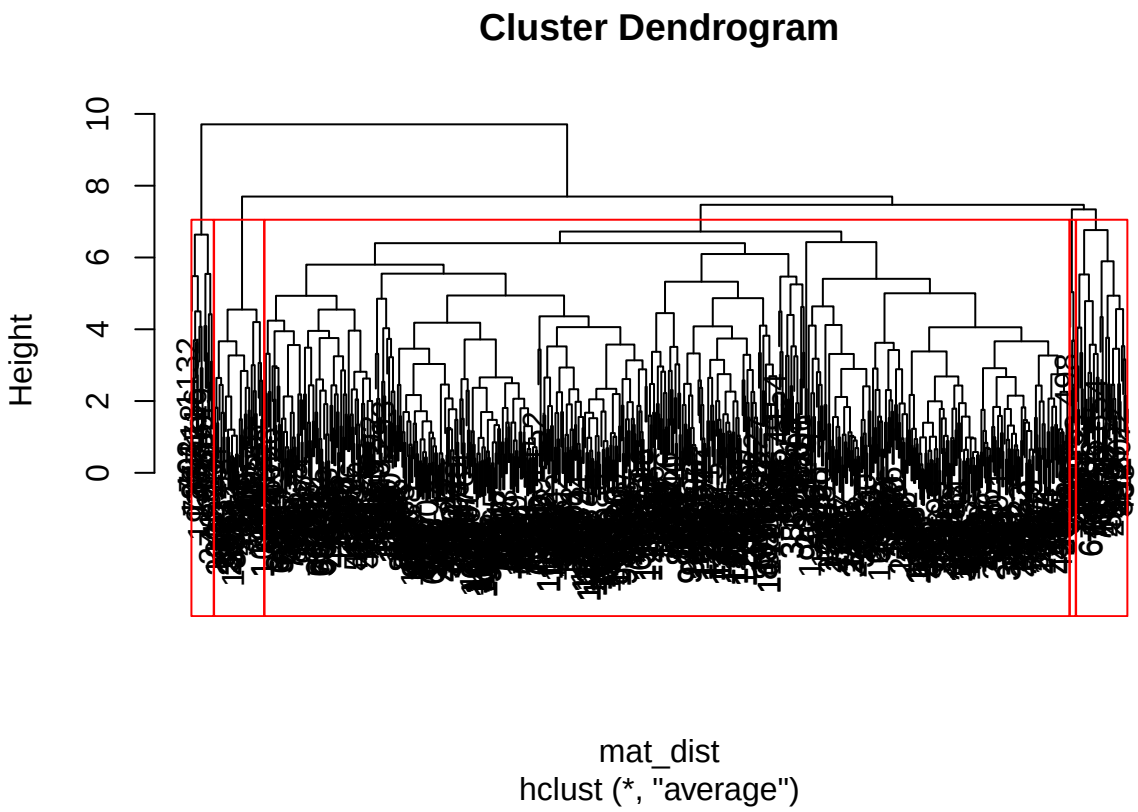



```
mat_dist <- dist(x = scale(seguros.numeric), method = "manhattan")
hc_complete <- hclust(d = mat_dist, method = "complete")
hc_average  <- hclust(d = mat_dist, method = "average")
hc_single   <- hclust(d = mat_dist, method = "single")
hc_ward     <- hclust(d = mat_dist, method = "ward.D2")
```

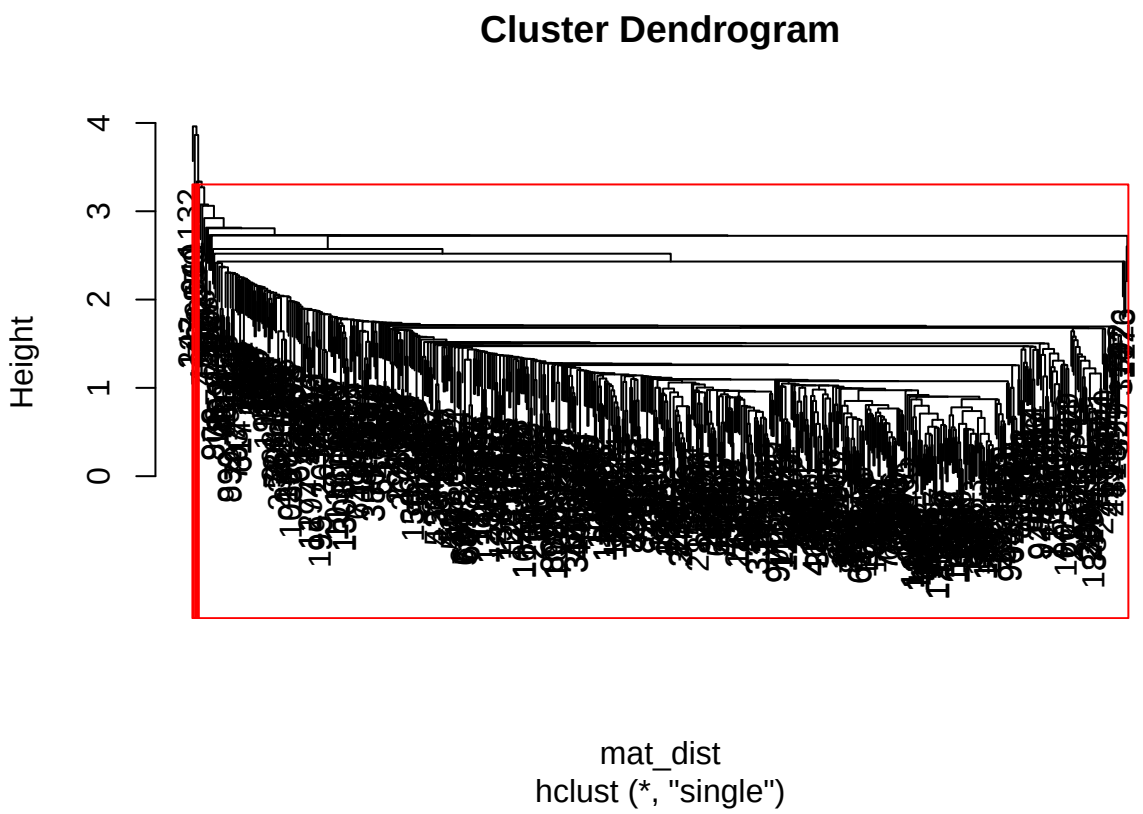
```
plot(hc_complete )
rect.hclust(hc_complete, k=5, border="red")
```



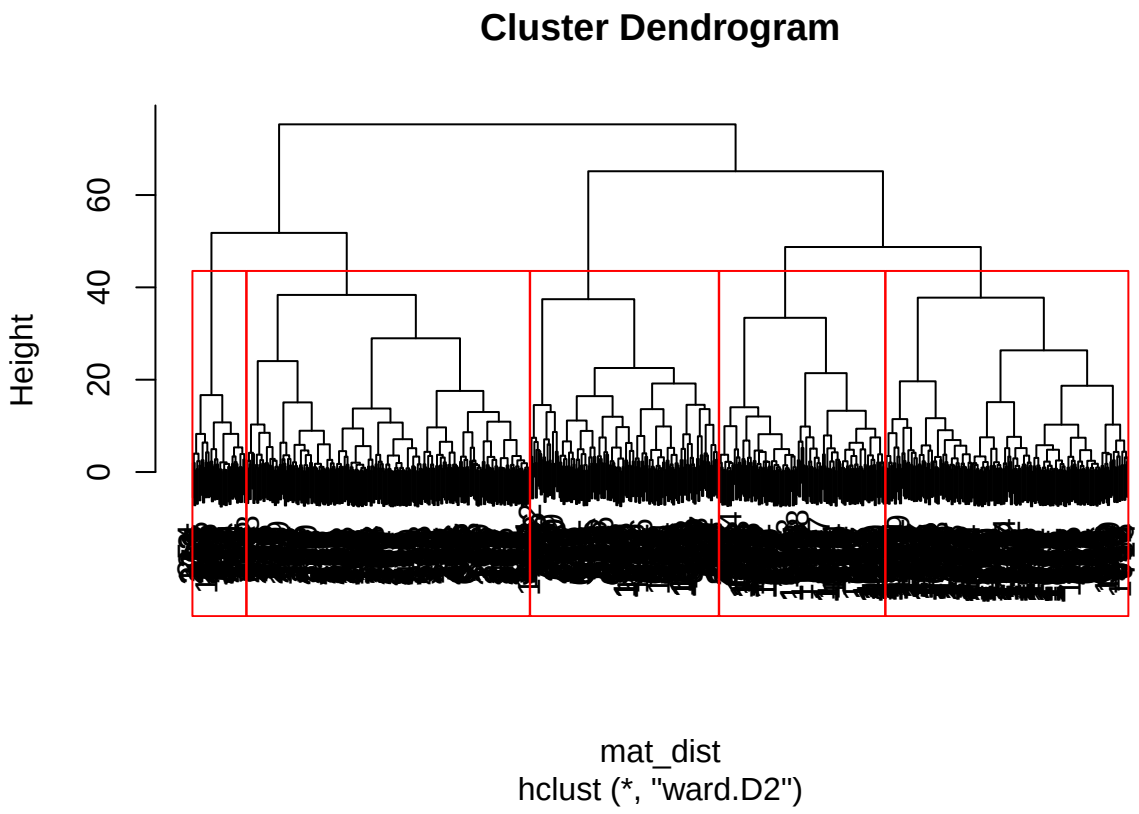

```
plot(hc_average )
rect.hclust(hc_average, k=5, border="red")
```



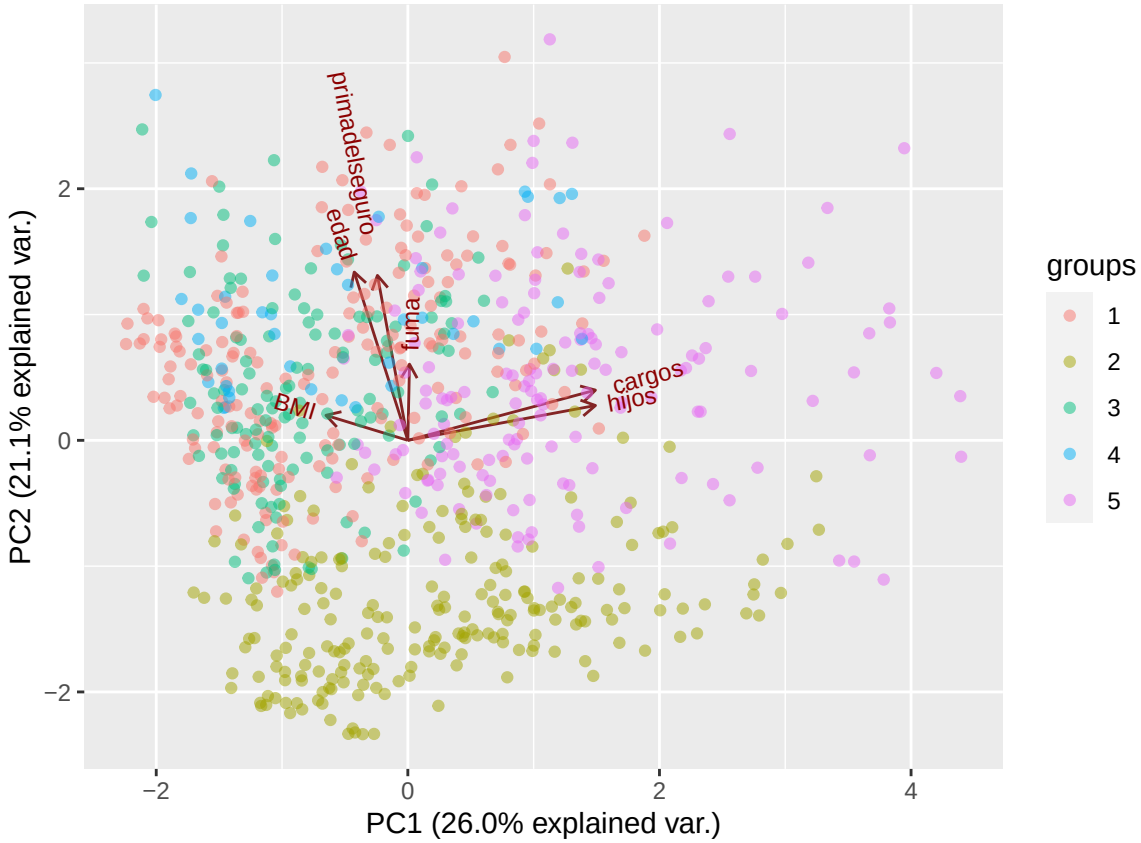

```
plot(hc_single )
rect.hclust(hc_single, k=5, border="red")
```




```
plot(hc_ward )
rect.hclust(hc_ward, k=5, border="red")
```



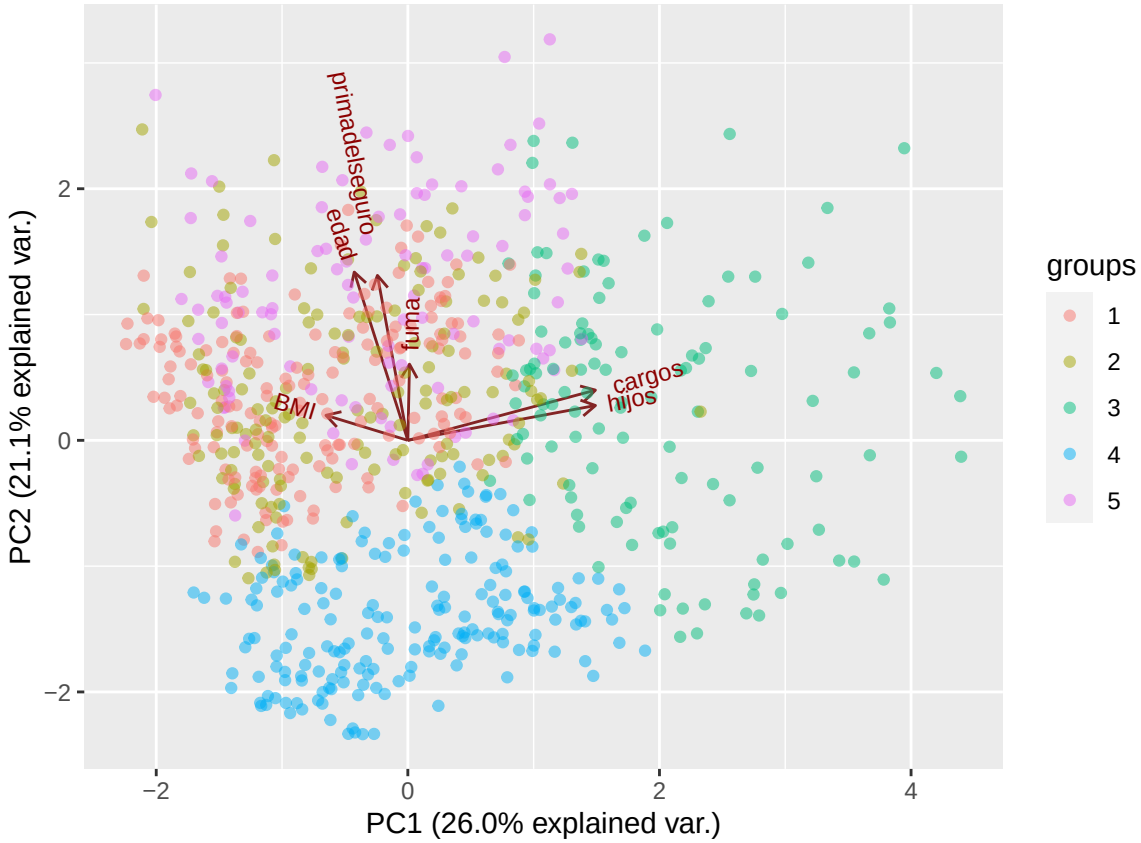

```
seguros$hcluster<-cutree(hc_ward,k=5)#con 5 grupos
ggbiplot(seguros.pca, obs.scale=1 ,var.scale=1, alpha=0.5,groups = as.factor(seguros$hcluster) )
```




```
seguros %>%  mutate(region = as.numeric(seguros$region) -1) %>%  mutate(sexo = as.numeric(seguros$sexo)-
  group_by(hcluster) %>%
  summarise_all(mean)
```

```
## # A tibble: 5 x 9
##   hcluster  edad  sexo   BMI hijos  fuma region cargos primadelseguro
##      <int> <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl>  <dbl>          <dbl>
## 1        1  52.5 0.492  29.6  25.4 0.337 0.135    2.05         14000.
## 2        2  24.1 0.498  29.1  38.7 0.32  0.124    2.32          5438.
## 3        3  44.0 0.515  33.7  19.2 2.18  0.0530   1.85         10195.
## 4        4  26.0 0.651  33.1  29.8 0.674 0.930    2.09         35629.
## 5        5  38.8 0.5    27.7  91.0 2.13  0.1      3.11          9761.
```

```
seguros.kmeans = kmeans(scale(seguros.numeric),nstart=50,centers = 5)
seguros$kmeanscluster = as.factor(seguros.kmeans$cluster)

ggbiplot(seguros.pca, obs.scale=1 ,var.scale=1, alpha=0.5,groups = as.factor(seguros$kmeansclust) )
```




```
seguros %>%  mutate(region = as.numeric(seguros$region) -1) %>%  mutate(sexo = as.numeric(seguros$sexo)-
  group_by(kmeanscluster) %>%
  summarise_all(mean)
```

```
## # A tibble: 5 x 10
##   kmeanscluster  edad  sexo   BMI hijos  fuma region cargos primadelseguro
##   <fct>         <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl>  <dbl>          <dbl>
## 1 1              52.5 0.538  32.1  21.6 0.434 0        1.96         10525.
## 2 2              40.1 0.506  31.1  27.7 2.72  0.0962   1.98          9903.
## 3 3              37.9 0.443  26.8 117.  1.11  0.104    3.66          9072.
## 4 4              23.9 0.505  29.2  29.9 0.376 0.0773   2.07          4089.
## 5 5              37.6 0.543  29.7  36.0 0.705 0.705    2.35         29193.
## # ... with 1 more variable: hcluster <dbl>
```