Clasificación de los FRE mediante análisis por clusterización

Misión PRI 1901

true

08-10–2021

Abstract

A continuación, se presenta un análisis mediante agrupación por clústers de los FRE de acuerdo a variables de desempeño y complejidad de los mismos.

En este análisis se hace un preprocesamiento de los datos mediante estandarización Z.

```
require(readxl)
require(skimr)
require(PerformanceAnalytics)
require(factoextra)
require(patchwork)
require(clValid)
require(plotly)
require(tidymodels)
require(tidyverse); theme_set(theme_bw())
source(file.path('src', 'models', '900_funcionesAlmacenamientoGrafico.R'), encoding = 'UTF-8')
source(file.path('src', 'models', '803_funcionesggDendro.R'), encoding = 'UTF-8')
fig_path <- file.path('figures', '011_clasificacion')
```

```
data <-
  read_excel(file.path(
    'data',
    'raw',
    'ClasificacionFRE',
    'variablesClasificacionFRE.xlsx'
  ), na = '-') %>%
  mutate(Departamento...2 = str_to_title(Departamento...2))
```

```
## New names:
## * Departamento -> Departamento...1
## * Departamento -> Departamento...2
```

```
data %>%
  select(!contains('Departamento')) %>%
  chart.Correlation(., histogram = TRUE, pch = 19)
```

```
if (knitr::is_html_output()) {
  skimr::skim(data)
}
```

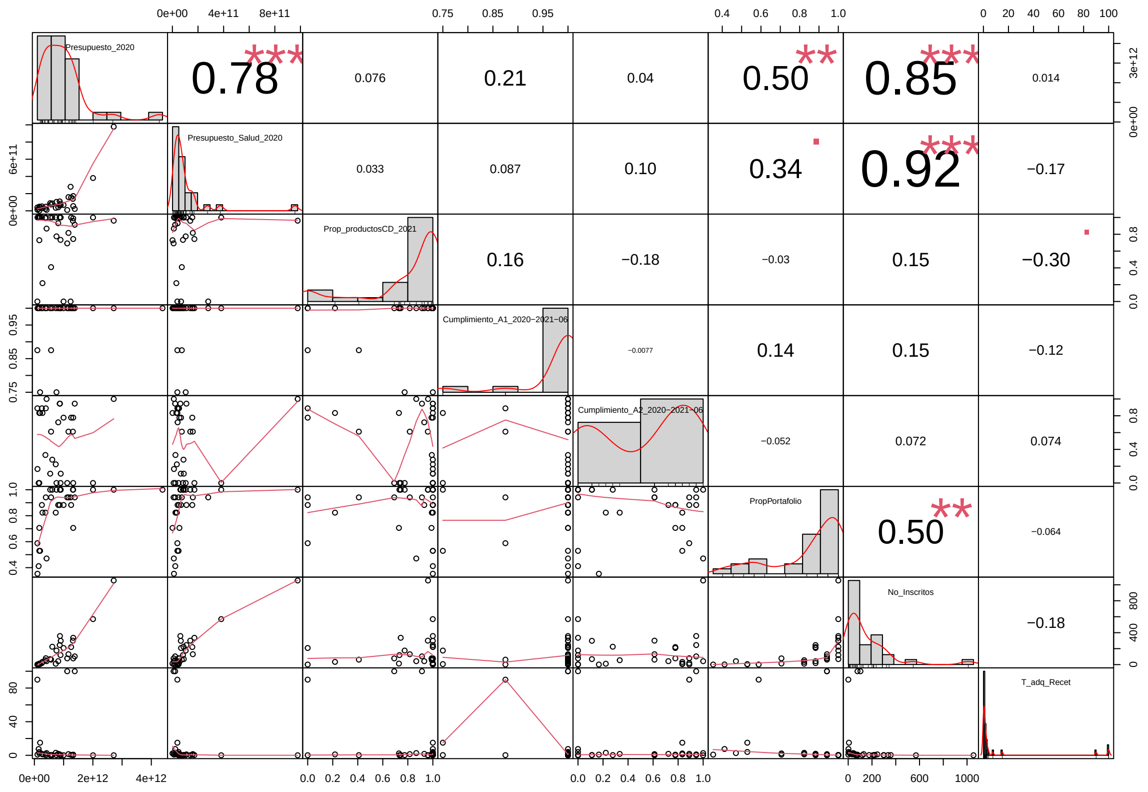

Figure 1: Correlación entre variables de PCA

1. Análisis por PCA

```
# 1. Análisis por PCA ----------------------------------------------------

pca1 <- data %>%
  drop_na() %>%
  column_to_rownames('Departamento...2') %>%
  select(!contains('Departamento')) %>%
  prcomp(., scale = TRUE, center = TRUE)

dimVar <- pca1$sdev %>% {.^2*100/sum(.^2)}
```

```
plot(pca1, main = 'Distribución de varianza')
```

```
gg1 <- fviz_pca_ind(pca1,
             col.ind = "cos2",
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE
) + labs(title = 'PCA de individuos')
```

```
gg1
```

```
guardarGGplot(gg1, '001_fig_ind', 8, 6, fig_path)

gg2 <- fviz_pca_var(
```

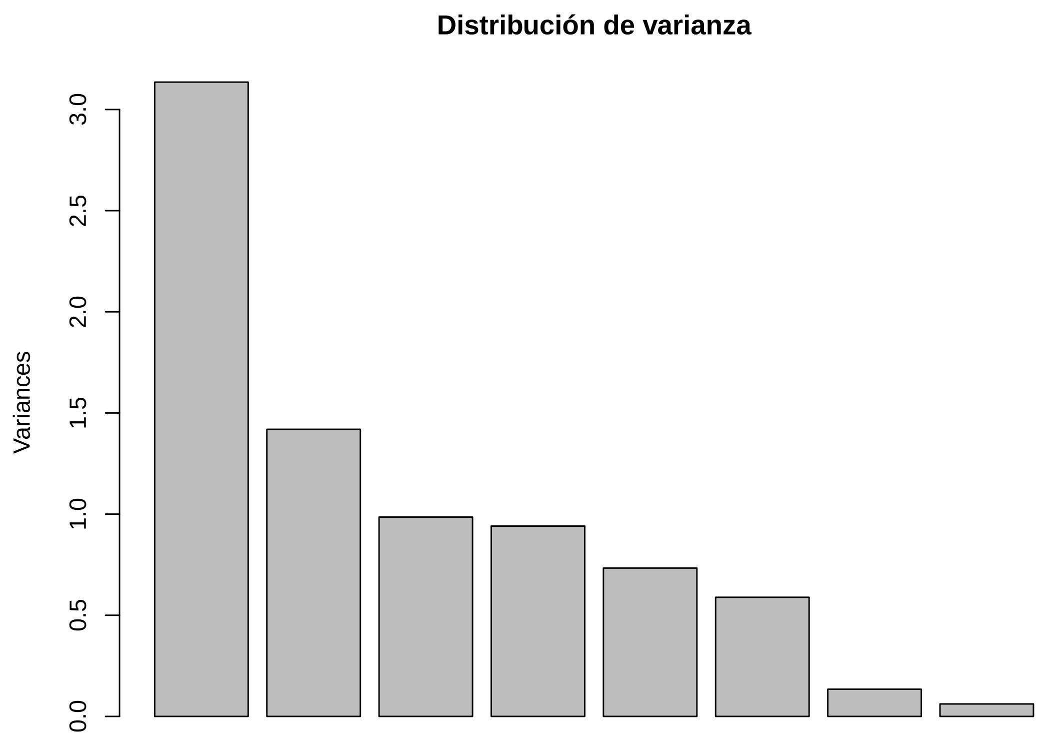

Figure 2: Distribución de las varianzas

```r
  pca1,
  col.var = "contrib",
  gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
  repel = TRUE
) + labs(title = 'PCA - Explicación de variables')
```

```r
gg2
```

```r
guardarGGplot(gg2, '002_fig_var', 8, 6, fig_path)
```

2. Preprocesamiento

```r
# 2. Preprocesamiento ----------------------------------------

norm_trans <- data %>%
  recipe(~ Presupuesto_2020 + Presupuesto_Salud_2020 + Prop_productosCD_2021 +
           `Cumplimiento_A1_2020-2021-06` + `Cumplimiento_A2_2020-2021-06` +
           PropPortafolio + No_Inscritos + T_adq_Recet + Departamento...2) %>%
  update_role(Departamento...2, new_role = 'ID') %>%
  step_normalize(all_numeric_predictors()) %>%
  step_naomit(all_numeric_predictors())
```

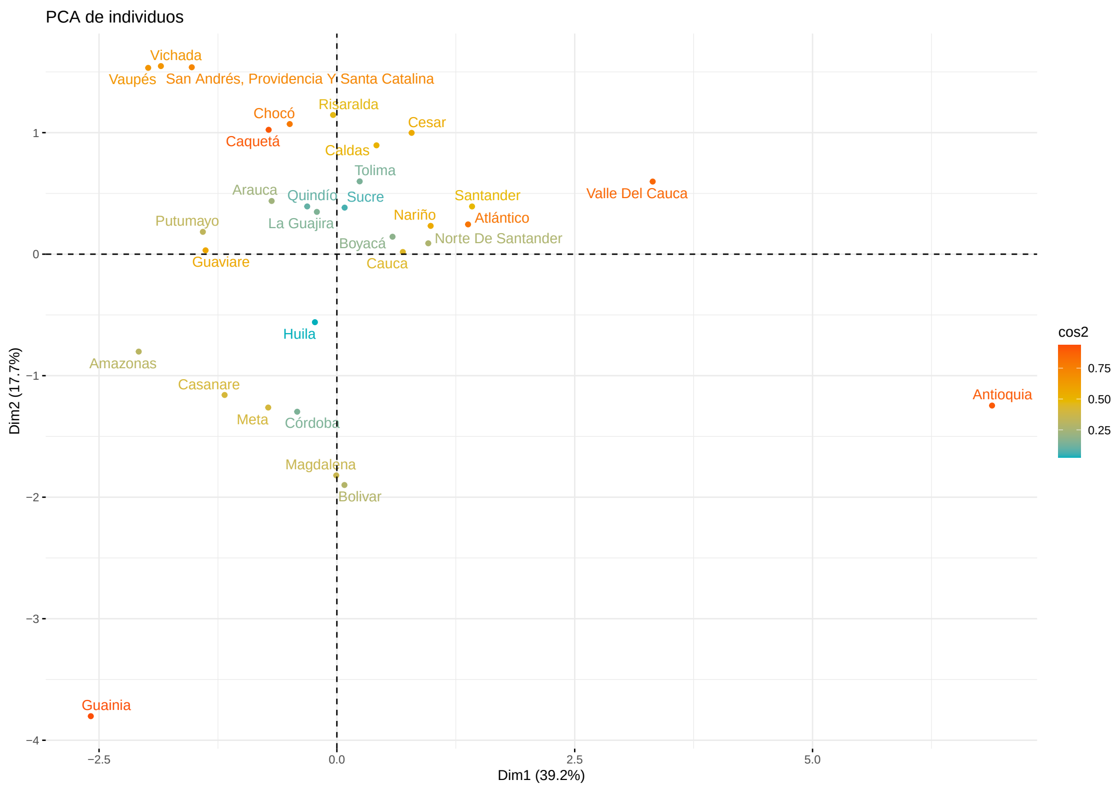

Figure 3: Individuos representados en PCA

```r
norm_obj <- norm_trans %>%
  prep(training = data)

clean_data <- bake(norm_obj, data) %>%
  column_to_rownames('Departamento...2')
```

3. Análisis de Clústers por Kmeans

```r
# 3. Análisis de Clústers por Kmeans ---------------------------------------

kmValid <- clValid(clean_data, 2:30, clMethods = 'kmeans',
                   validation = c('internal', 'stability'))


kmValid %>% optimalScores()
```

```
##                    Score Method Clusters
## APN          0.008064516 kmeans        2
## AD           0.052982849 kmeans       30
## ADM          0.047163094 kmeans       30
## FOM          0.686961591 kmeans       28
## Connectivity 5.000793651 kmeans        2
## Dunn         1.228225121 kmeans       30
## Silhouette   0.452524710 kmeans        2
```

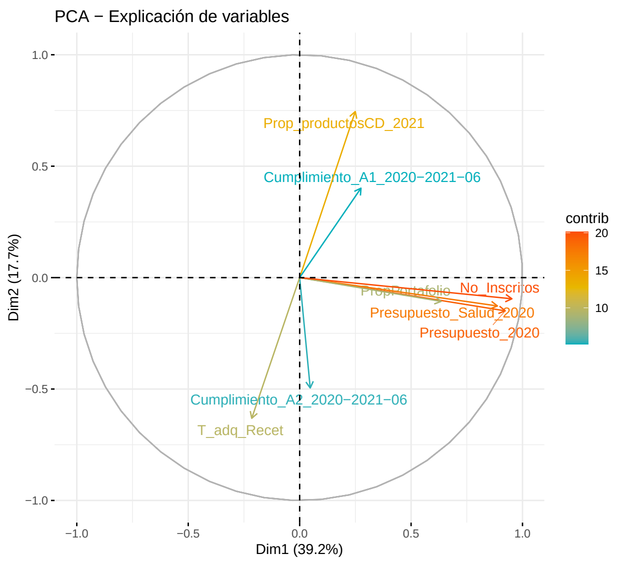

Figure 4: Explicación de variables en PCA

```
kmValidS <- as.data.frame(measures(kmValid)) %>%
  {rownames_to_column(as_tibble(t(.)), var = 'k')} %>%
  mutate(k = as.double(k) + 1,
         tot_withins_ss = map_dbl(k, ~ kmeans(clean_data, .x)$tot.withinss)) %>%
  pivot_longer(cols = !matches('k'))  %>%
  # Selección manual de óptimo de índices de validez
  mutate(koptim = case_when(
    name == 'tot_withins_ss' ~ 9,
    name == 'AD' ~ 5,
    name == 'APN' ~ 2,
    name == 'Connectivity' ~ 2,
    name == 'FOM' ~ 5,
    name == 'Dunn' ~ 6,
    name == 'Silhouette' ~ 6,
    TRUE ~ 3
  ))

gg3 <- kmValidS  %>%
  ggplot(aes(x = k, y = value)) +
  geom_point() + geom_line() +
  geom_vline(aes(xintercept = koptim), col = 'blue4', lty = 'dashed') +
  ggrepel::geom_label_repel(data = subset(kmValidS, k == koptim),
          aes(label = koptim)) +
  ylab('Valor') +
  scale_color_manual(values = c('Sí' = 'red', 'No' = NA)) +
  facet_wrap(vars(name), scales = 'free_y')
```

```
gg3
```

```
guardarGGplot(gg3, '003_elbowlKmeans', 8, 6, fig_path)

kmValidS %>%
  distinct(name, koptim) %>%
```

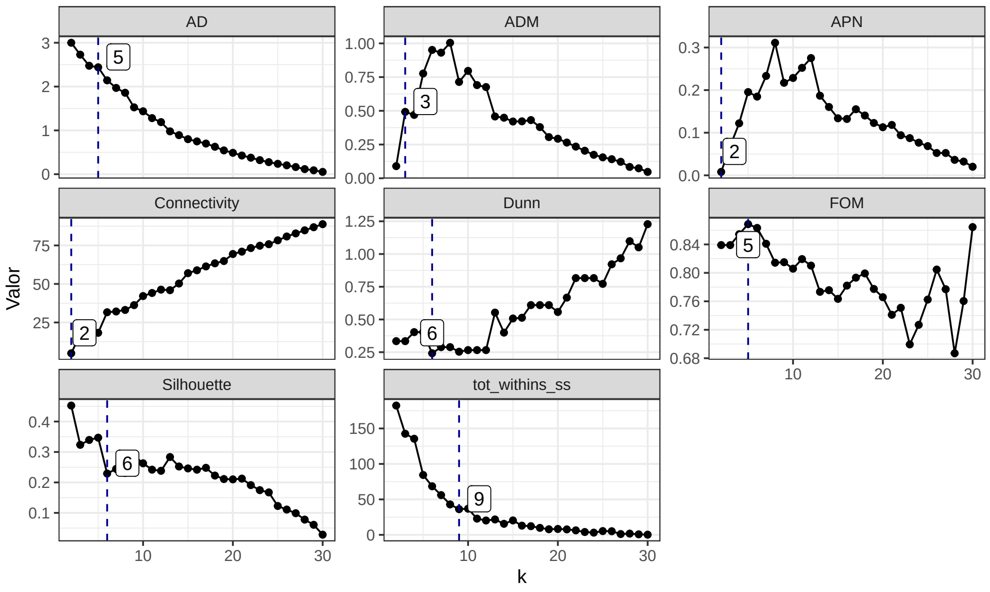

Figure 5: Criterio de codo para clústers por Kmeans

```
arrange(koptim)
```

```
## # A tibble: 8 x 2
##   name           koptim
##   <chr>           <dbl>
## 1 APN                 2
## 2 Connectivity        2
## 3 ADM                 3
## 4 AD                  5
## 5 FOM                 5
## 6 Dunn                6
## 7 Silhouette          6
## 8 tot_withins_ss      9
```

```
g1 <- kmeans(clean_data, 8) %>%
  fviz_cluster(., data = clean_data, stand = T,
               show.clust.cent = F, repel = T, max.overlaps=Inf) +
  scale_fill_brewer(palette = 'Dark2') +
  scale_colour_brewer(palette = 'Dark2') +
  theme_bw() + labs(title = NULL)
```

```
g1
```

```
## Warning: ggrepel: 2 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

```
guardarGGplot(g1, '004_clusterKmeans', 8, 5, fig_path)
```

```
## Warning: ggrepel: 2 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

```
g2 <- kmeans(clean_data, 5) %>%
  fviz_cluster(., data = clean_data, axes = c(1, 2)) +
```

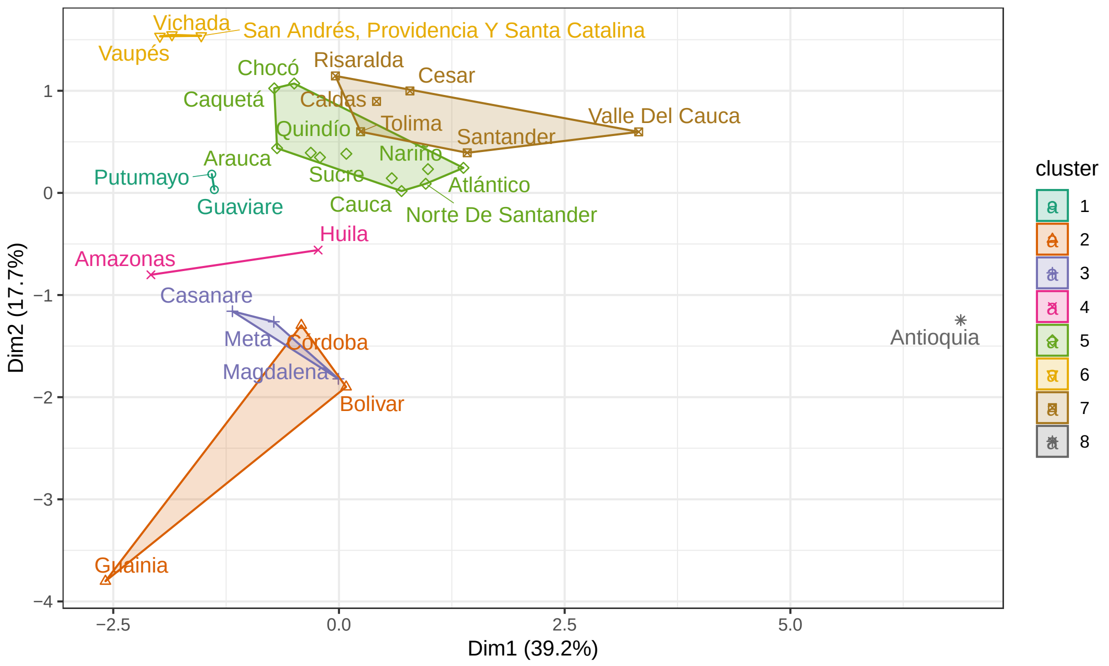

Figure 6: Clústers Kmeans visualizados en primeros dos PC

```
  theme_bw() + labs(title = NULL)
g3 <- kmeans(clean_data, 5) %>%
  fviz_cluster(., data = clean_data, axes = c(1, 3)) +
  theme_bw() + labs(title = NULL)
g4 <- kmeans(clean_data, 5) %>%
  fviz_cluster(., data = clean_data, axes = c(2, 3)) +
  theme_bw() + labs(title = NULL)
g5 <- kmeans(clean_data, 5) %>%
  fviz_cluster(., data = clean_data, axes = c(1, 4)) +
  theme_bw() + labs(title = NULL)

ggt <- wrap_plots(g2, g3, g4, g5)
```

```
ggt
```

```
guardarGGplot(ggt, '005_cluz_group', 12, 8, fig_path)
```

3. Clúster jerárquicos

```
funClusters_2 <- function(data, k) {
  t1 <- data %>%
    dist(method = 'euclidean') %>%
    hclust(method = 'complete')

  t2 <- cutree(t1, k)
  return(list(clust = t1, tree = t2))
}
```

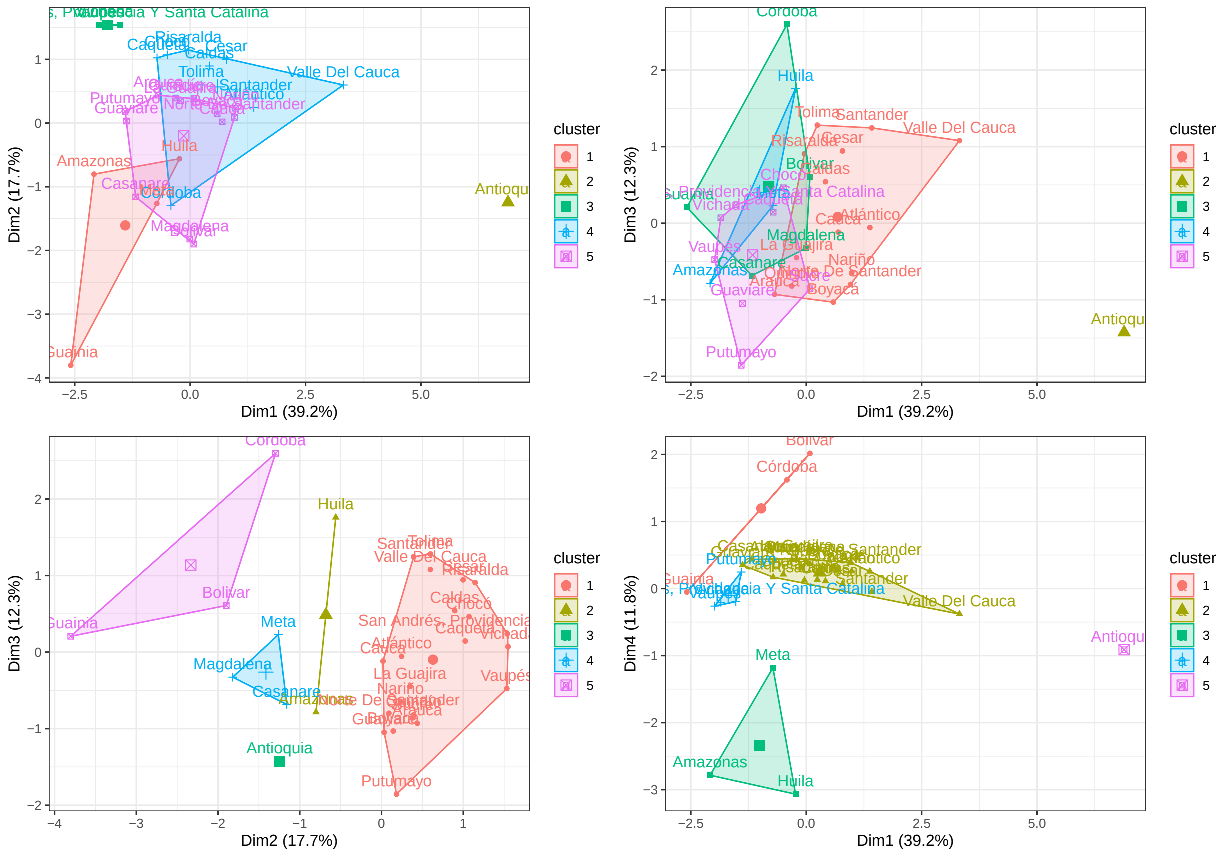

Figure 7: Clústers Kmeans visualizados en varios componentes

```
# 3. Análisis de Clústers por Kmeans ---------------------------------------

p1 <- plot(funClusters_2(clean_data, 3)$clust, cex = 0.6, hang = -1, ylab = 'Altura',
     main = 'Dendrograma de clúster', xlab = NULL)

pdf(file.path(fig_path, '010_dendrograma.pdf'), width = 12, height = 8)
plot(funClusters_2(clean_data, 3)$clust, cex = 0.6, hang = -1, ylab = 'Altura',
     main = 'Dendrograma de clúster', xlab = NULL)
dev.off()

## pdf
##   2

saveRDS(p1, file.path(fig_path, '010_dendrograma.rds'))


ddata <- funClusters_2(clean_data, 8)$clust %>%
  {dendro_data_k(., 8)}

p1b <- plot_ggdendro(
  ddata,
  direction   = "lr",
  expand.y    = 0.5,
```

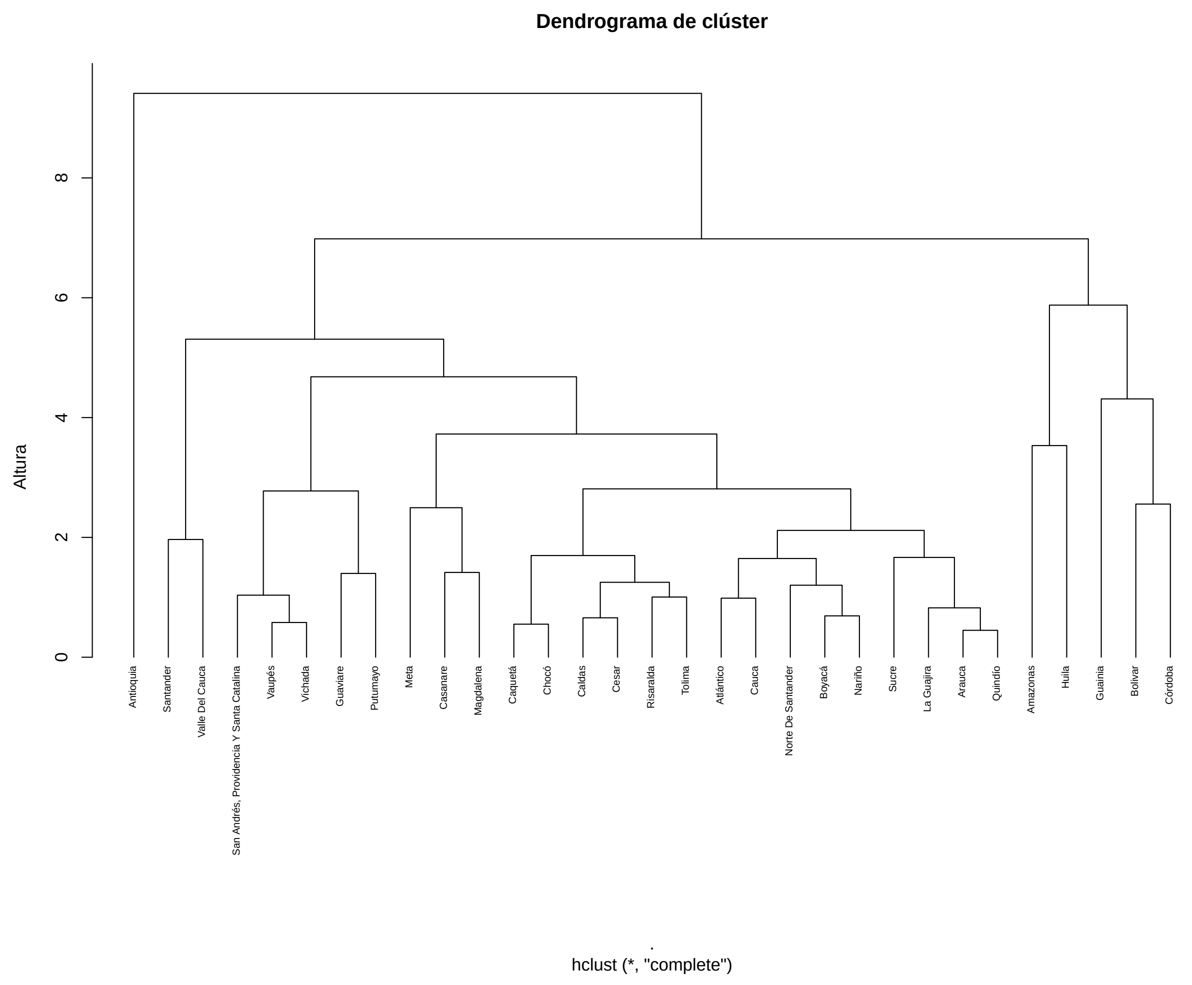

Figure 8: Dendrograma de análisis por clústers

```
  scale.color = RColorBrewer::brewer.pal(8 + 1, "Paired")
) +
  theme(axis.title = element_blank(),
        panel.grid = element_blank())

guardarGGplot(p1b, '010b_dendrograma', 8, 6, fig_path)

gg1 <- funClusters_2(clean_data, 3)$clust$height %>%
  as.tibble() %>%
  add_column(groups = length(funClusters_2(clean_data, 3)$clust$height):1) %>%
  rename(height = value) %>%
  ggplot(aes(x=groups, y = height)) +
  geom_point() + geom_line() +
  # coord_cartesian(xlim=c(0, 15)) +
  geom_vline(xintercept = 8, lty = 'dashed', col = 'blue3') +
  ylab('Altura') +
  xlab('N.° de Clusters (k)')
```

```
## Warning: `as.tibble()` was deprecated in tibble 2.0.0.
```

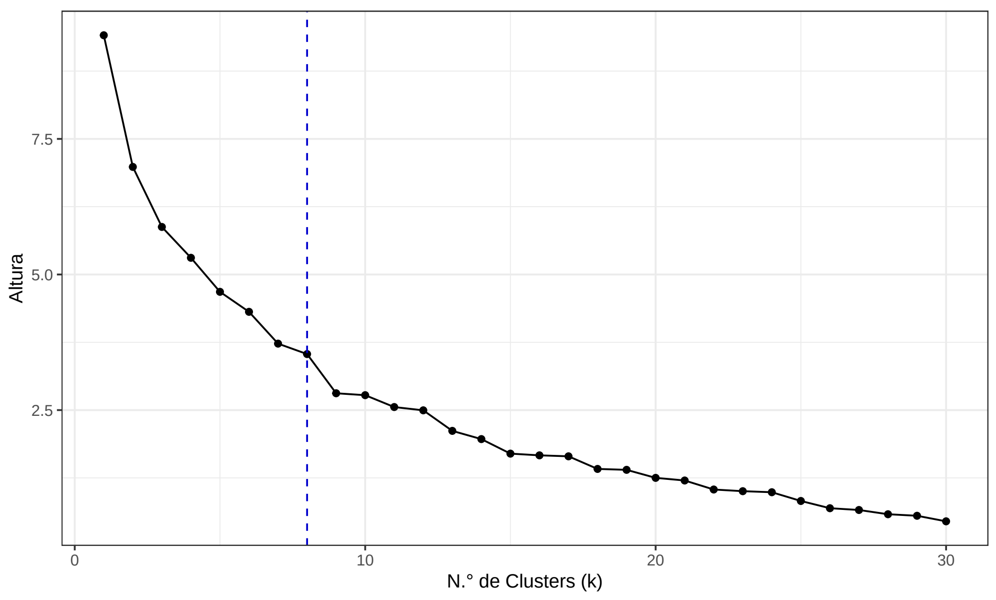

Figure 9: Criterio de codo para clústers jerárquicos

```
## Please use `as_tibble()` instead.
## The signature and semantics have changed, see `?as_tibble`.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```

```
gg1
```

```
guardarGGplot(gg1, '012_elbowlWard', 6, 4, fig_path)

# Índices de validez de clúster para algoritmo jerárquico
```

```
gg1b
```

```
guardarGGplot(gg1b, '003b_elbowlKmeans', 8, 6, fig_path)

gg1c <- kmValidS %>%
  filter(name == 'Silhouette') %>%
  ggplot(aes(x = k, y = value)) +
  geom_line() +
  geom_vline(aes(xintercept = koptim), col = 'blue4', lty = 'dashed') +
  ggrepel::geom_label_repel(data = subset(kmValidS, k == koptim & (name == 'Silhouette')),
                            aes(label = koptim)) +
  ylab('Valor') + xlab('N.° de clústers (k)')

gg1c
```

```
guardarGGplot(gg1c, '003c_elbowlWard', 8, 6, fig_path)

gg2 <- clean_data %>%
  {fviz_cluster(list(data = ., cluster = funClusters_2(., 8)$tree),
               repel = T, max.overlaps = Inf)} +
  theme_bw() + labs(title = NULL) +
  scale_color_brewer(palette = 'Dark2', name = 'Clúster') +
```

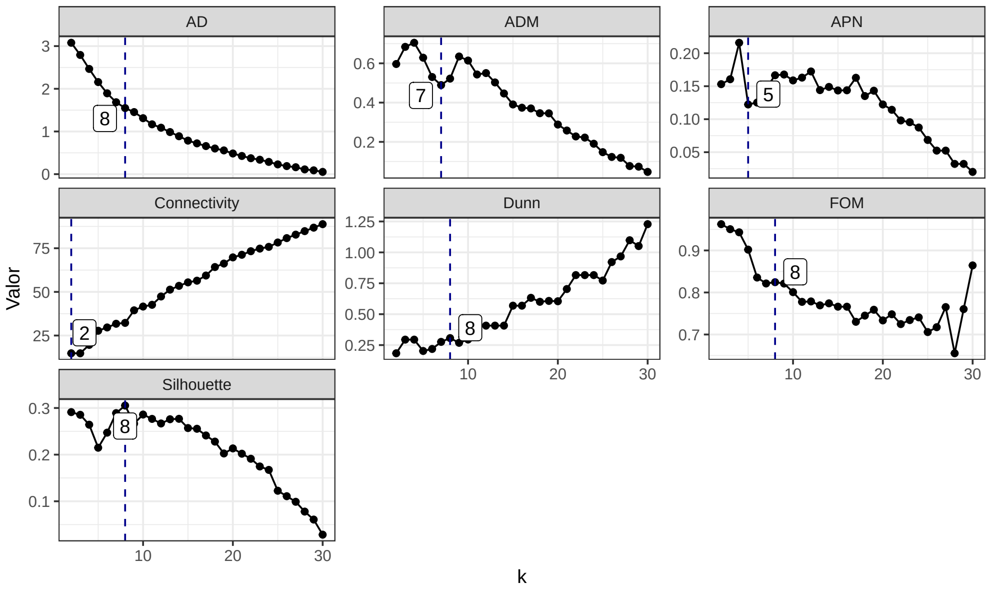

Figure 10: Criterios de codo para clústers por Jerárquico

```
  scale_fill_brewer(palette = 'Dark2', name = 'Clúster') +
  scale_shape_discrete(name = 'Clúster')
```

```
## Scale for 'shape' is already present. Adding another scale for 'shape', which
## will replace the existing scale.
```

```
gg2
```

```
## Warning: The shape palette can deal with a maximum of 6 discrete values because
## more than 6 becomes difficult to discriminate; you have 8. Consider
## specifying shapes manually if you must have them.

## Warning: Removed 7 rows containing missing values (geom_point).

## Warning: Removed 2 rows containing missing values (geom_point).

## Warning: ggrepel: 2 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

```
guardarGGplot(gg2, '013_cluz_group', 8, 5, fig_path)
```

```
## Warning: The shape palette can deal with a maximum of 6 discrete values because
## more than 6 becomes difficult to discriminate; you have 8. Consider
## specifying shapes manually if you must have them.

## Warning: Removed 7 rows containing missing values (geom_point).

## Warning: Removed 2 rows containing missing values (geom_point).

## Warning: ggrepel: 2 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

```
funClusters_3 <- function(axes) {
  clean_data %>%
    {fviz_cluster(list(data = ., cluster = funClusters_2(., 8)$tree),
                  axes = axes, ellipse = T)} +
```

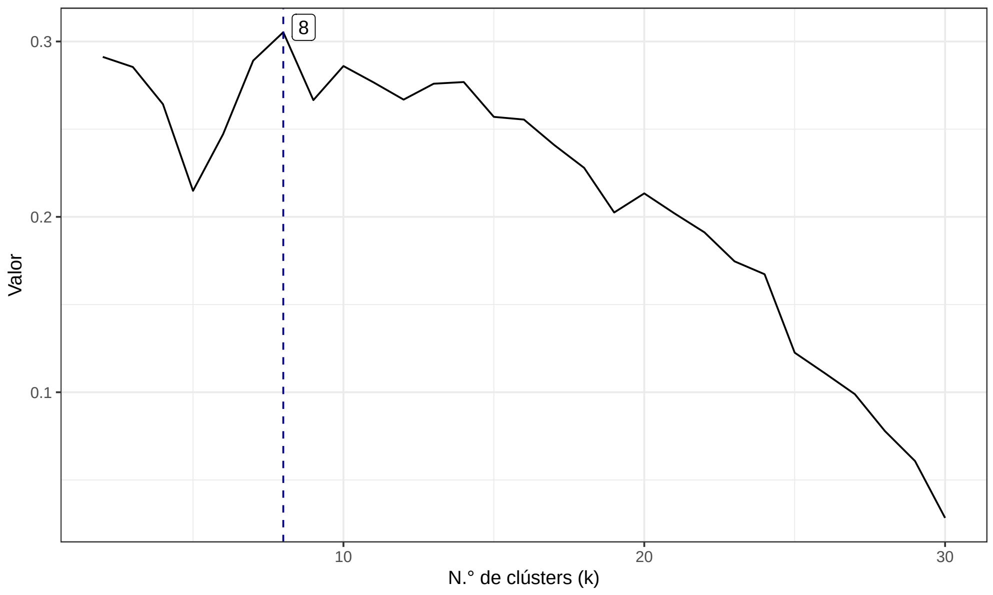

Figure 11: Criterios de codo para clústers por Jerárquico

```
    theme_bw() + labs(title = NULL) +
    scale_color_discrete(name = 'Clúster') +
    scale_shape_discrete(name = 'Clúster') +
    scale_fill_discrete(name = 'Clúster')
}

g2 <- funClusters_3(c(1,2))
```

```
## Scale for 'shape' is already present. Adding another scale for 'shape', which
## will replace the existing scale.
```

```
g3 <- funClusters_3(c(1,3))
```

```
## Scale for 'shape' is already present. Adding another scale for 'shape', which
## will replace the existing scale.
```

```
g4 <- funClusters_3(c(2,3))
```

```
## Scale for 'shape' is already present. Adding another scale for 'shape', which
## will replace the existing scale.
```

```
g5 <- funClusters_3(c(1,4))
```

```
## Scale for 'shape' is already present. Adding another scale for 'shape', which
## will replace the existing scale.
```

```
ggt <- wrap_plots(g2, g3, g4, g5)
```

```
ggt
```

```
## Warning: The shape palette can deal with a maximum of 6 discrete values because
## more than 6 becomes difficult to discriminate; you have 8. Consider
## specifying shapes manually if you must have them.
```

```
## Warning: Removed 7 rows containing missing values (geom_point).
```

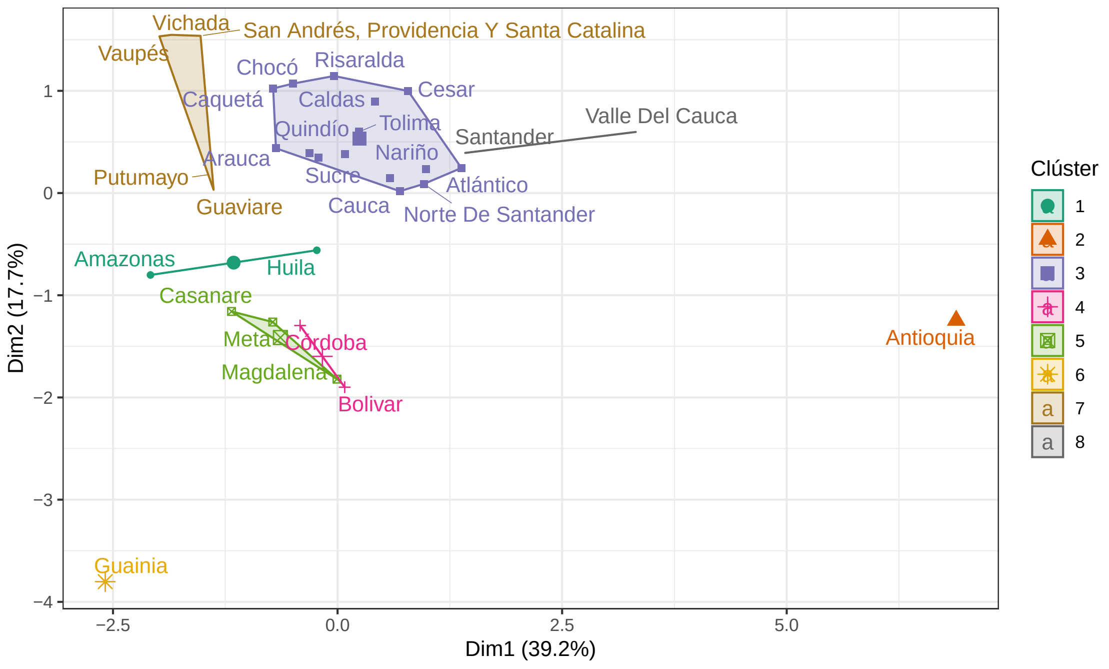

Figure 12: Clústers Jerárquicos visualizados en primeros dos componentes

```
## Warning: Removed 2 rows containing missing values (geom_point).

## Warning: The shape palette can deal with a maximum of 6 discrete values because
## more than 6 becomes difficult to discriminate; you have 8. Consider
## specifying shapes manually if you must have them.

## Warning: Removed 7 rows containing missing values (geom_point).

## Warning: Removed 2 rows containing missing values (geom_point).

## Warning: The shape palette can deal with a maximum of 6 discrete values because
## more than 6 becomes difficult to discriminate; you have 8. Consider
## specifying shapes manually if you must have them.

## Warning: Removed 7 rows containing missing values (geom_point).

## Warning: Removed 2 rows containing missing values (geom_point).

## Warning: The shape palette can deal with a maximum of 6 discrete values because
## more than 6 becomes difficult to discriminate; you have 8. Consider
## specifying shapes manually if you must have them.

## Warning: Removed 7 rows containing missing values (geom_point).

## Warning: Removed 2 rows containing missing values (geom_point).
```

```
guardarGGplot(ggt, '014_cluz_group2', 12, 10, fig_path)
```

```
## Warning: The shape palette can deal with a maximum of 6 discrete values because
## more than 6 becomes difficult to discriminate; you have 8. Consider
## specifying shapes manually if you must have them.

## Warning: Removed 7 rows containing missing values (geom_point).

## Warning: Removed 2 rows containing missing values (geom_point).
```

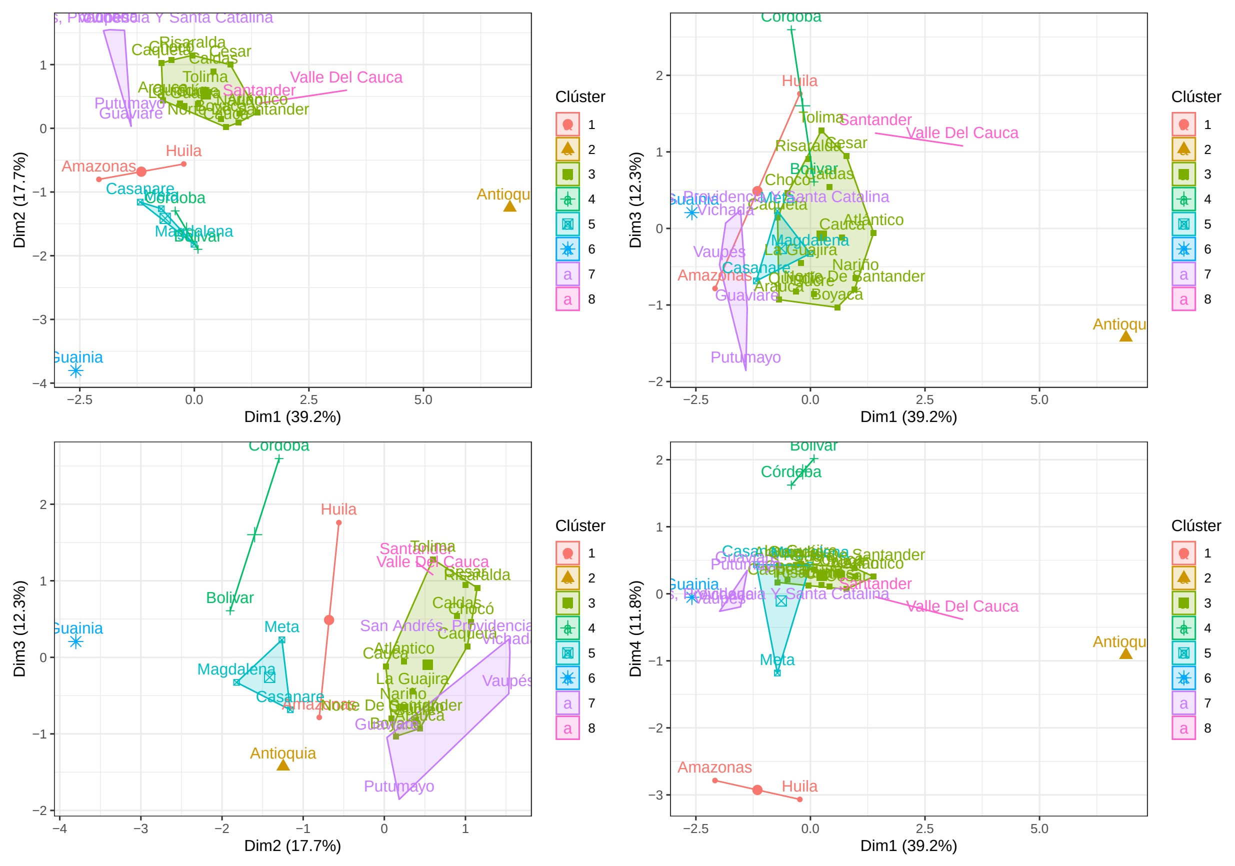

Figure 13: Clústers jerarquizados visualizados en varios componentes

```
## Warning: The shape palette can deal with a maximum of 6 discrete values because
## more than 6 becomes difficult to discriminate; you have 8. Consider
## specifying shapes manually if you must have them.

## Warning: Removed 7 rows containing missing values (geom_point).

## Warning: Removed 2 rows containing missing values (geom_point).

## Warning: The shape palette can deal with a maximum of 6 discrete values because
## more than 6 becomes difficult to discriminate; you have 8. Consider
## specifying shapes manually if you must have them.

## Warning: Removed 7 rows containing missing values (geom_point).

## Warning: Removed 2 rows containing missing values (geom_point).

## Warning: The shape palette can deal with a maximum of 6 discrete values because
## more than 6 becomes difficult to discriminate; you have 8. Consider
## specifying shapes manually if you must have them.

## Warning: Removed 7 rows containing missing values (geom_point).

## Warning: Removed 2 rows containing missing values (geom_point).
```

```
colors <- c(`1` = '#a705b3', `2` = '#7081ff', `3` = '#ff0000',
            `4` = '#8cffb7', `5` = '#00ff5e')
```

```r
# Creación de hover
hov_data <- data %>%
  mutate(
    Prec2020 = round(Presupuesto_2020/1e9, 2),
    PrecSalud2020 = round(Presupuesto_Salud_2020/1e9, 2),
    Hover = glue::glue(
      "<b>{Departamento...2}</b> <br>",
      "<i>Complejidad</i> <br>",
      "Presupuesto (miles de millones): {Prec2020} <br>",
      "Presupuesto Salud (miles de millones): {PrecSalud2020} <br>",
      "No. de inscritos: {No_Inscritos} <br>",
      "Prop. uso portafolio: {round(PropPortafolio,2)} <br>",
      "<i>Indicadores</i> <br>",
      "Prop. ventas por FRE vs CD: {round(Prop_productosCD_2021,2)} <br>",
      "Prop. cumplimiento A1: {round(`Cumplimiento_A1_2020-2021-06`,2)} <br>",
      "Prop. cumplimiento A2: {round(`Cumplimiento_A2_2020-2021-06`,2)} <br>",
      "Raz. tiempo de adq rec vs no. rec: {round(T_adq_Recet, 2)}"
    )
  ) %>%
  select(Departamentos = Departamento...2, Hover)



trans_data <- as_tibble(pca1$x, rownames = 'Departamentos') %>%
  left_join(funClusters_2(clean_data, 8)$tree %>%
              as_tibble(rownames = 'Departamentos'), by = 'Departamentos') %>%
  rename(Grupo_hclus = value) %>%
  mutate(colores = colors[Grupo_hclus],
         Grupo_hclus = factor(Grupo_hclus))


fig <- trans_data %>%
  left_join(hov_data, by = 'Departamentos') %>%
  plot_ly(name = ~Grupo_hclus) %>%
  add_trace(x = ~PC1, y = ~PC2, z = ~PC3,
            customdata = ~Hover,
            hovertemplate = "%{customdata}",
            mode = 'markers',
            type = 'scatter3d',
            color = ~Grupo_hclus,
            colors = "Paired"
            # marker = list(color = ~colores, size=6)
            ) %>%
  layout(scene = list(
    xaxis = list(title = paste0('PC1 (', round(dimVar[1], 1), '%)'), range = c(-7,+7)),
    yaxis = list(title = paste0('PC2 (', round(dimVar[2], 1), '%)'), range = c(-4,+4)),
    zaxis = list(title = paste0('PC3 (', round(dimVar[3], 1), '%)'), range = c(-3,+3))
  ))
```

```r
if (knitr::is_html_output()) {
  fig
}

guardarPlotly(fig, '020_cluster_1', ruta = fig_path, libdir = 'plotly')
```

```
trans_data1 <- data %>%
  left_join(funClusters_2(clean_data, 8)$tree %>%
              as_tibble(rownames = 'Departamentos'),
            by = c('Departamento...2' = 'Departamentos')) %>%
  rename(Grupo_hclus = value, Departamentos = Departamento...2) %>%
  mutate(Grupo_hclus = as.integer(Grupo_hclus))

tr_df1 <- trans_data1 %>%
  group_by(Grupo_hclus) %>%
  summarise(across(!matches('Departamento'), mean))

tr_df1
```

```
## # A tibble: 9 x 9
##   Grupo_hclus Presupuesto_2020 Presupuesto_Salud_2020 Prop_productosCD_2021
##         <int>            <dbl>                  <dbl>                 <dbl>
## 1           1          4.76e11            71447478785                 0.887
## 2           2          2.71e12           978388352690                 0.962
## 3           3          8.22e11            80272733312.                0.961
## 4           4          1.25e12            15524588562.                0.802
## 5           5          6.15e11            57380750808.                0.209
## 6           6          1.02e11            38902520000                 0
## 7           7          2.01e11            19011901713.                0.919
## 8           8          1.67e12           276624306576                 0.871
## 9          NA          2.81e12                     NA                NA
## # ... with 5 more variables: Cumplimiento_A1_2020-2021-06 <dbl>,
## #   Cumplimiento_A2_2020-2021-06 <dbl>, PropPortafolio <dbl>,
## #   No_Inscritos <dbl>, T_adq_Recet <dbl>
```

```
tr_df1 %>%
  write_csv(file.path('references', 'clusterRead.csv'))




trans_data1 %>%
  mutate(across(!matches('Departamento|Grupo'),
                function(x) x >= mean(x, na.rm = T))) %>%
  group_by(Grupo_hclus) %>%
  summarise(across(!matches('Departamento'),
                   function(x) sum(x)/n())) %>%
  drop_na(Grupo_hclus) %>%
  column_to_rownames('Grupo_hclus') %>%
  t()
```

```
##                                1 2         3   4         5 6   7   8
## Presupuesto_2020             0.0 1 0.2666667 1.0 0.3333333 0 0.0 1.0
## Presupuesto_Salud_2020       0.0 1 0.2666667 0.0 0.0000000 0 0.0 1.0
## Prop_productosCD_2021        0.5 1 0.9333333 0.5 0.0000000 0 0.8 0.5
## Cumplimiento_A1_2020-2021-06 0.0 1 1.0000000 1.0 0.6666667 0 1.0 1.0
## Cumplimiento_A2_2020-2021-06 0.5 1 0.6000000 0.5 1.0000000 1 0.4 0.0
## PropPortafolio               0.5 1 0.8000000 1.0 0.6666667 0 0.0 1.0
## No_Inscritos                 0.5 1 0.4000000 0.0 0.3333333 0 0.0 1.0
## T_adq_Recet                  0.5 0 0.0000000 1.0 0.0000000 1 0.0 0.0
```

```r
aplicCuartiles <- function(rango) {

  quant <- quantile(rango, probs = seq(0, 1, by = 0.20), na.rm=TRUE) %>%
    unique()

  quant_labels <- paste0('G', seq(1, length(quant)-1))

  # return(quant_labels)

  cut(rango,
      breaks= quant,
      labels = quant_labels,
      include.lowest=TRUE)
}

trans_data1 %>%
  filter(!is.na(Grupo_hclus)) %>%
  mutate(across(!matches('Departamento|Grupo'),
                ~aplicCuartiles(.x))) %>%
  group_by(Grupo_hclus) %>%
  summarise(across(!matches('Departamento'),
                   function(x) paste0(unique(x), collapse = ','))) %>%
  drop_na(Grupo_hclus) %>%
  column_to_rownames('Grupo_hclus') %>%
  t()
```

```
##                               1       2    3                 4       5
## Presupuesto_2020              "G1,G3" "G5" "G1,G5,G3,G2,G4" "G5,G4" "G2,G4"
## Presupuesto_Salud_2020        "G2,G4" "G5" "G3,G5,G4,G1,G2" "G1"    "G2,G3,G4"
## Prop_productosCD_2021         "G3,G2" "G3" "G3,G2,G1"       "G2,G1" "G1"
## Cumplimiento_A1_2020-2021-06 "G1"    "G1" "G1"             "G1"    "G1"
## Cumplimiento_A2_2020-2021-06 "G3,G1" "G4" "G3,G2,G4,G1"    "G4,G1" "G3,G2"
## PropPortafolio                "G1,G4" "G4" "G2,G4,G3,G1"    "G3"    "G2,G3"
## No_Inscritos                  "G1,G4" "G5" "G2,G5,G4,G1,G3" "G3"    "G2,G4"
## T_adq_Recet                   "G5,G2" "G1" "G4,G1,G3,G2"    "G5"    "G3,G1"
##                               6    7          8
## Presupuesto_2020              "G1" "G1,G2"    "G5"
## Presupuesto_Salud_2020        "G2" "G1,G3"    "G5"
## Prop_productosCD_2021         "G1" "G1,G2,G3" "G2,G3"
## Cumplimiento_A1_2020-2021-06 "G1" "G1"       "G1"
## Cumplimiento_A2_2020-2021-06 "G3" "G3,G4,G1" "G1"
## PropPortafolio                "G1" "G1"       "G4"
## No_Inscritos                  "G1" "G1,G2"    "G5"
## T_adq_Recet                   "G5" "G4,G5,G3" "G2,G1"
```

```r
trans_data1 %>%
  mutate(across(!matches('Departamento'), ~mean(.x, na.rm = T),
                .names = "{.col}_mn"))
```

```
## # A tibble: 33 x 20
##    Departamento...1 Departamentos Presupuesto_2020 Presupuesto_Salud_2020
##               <dbl> <chr>                    <dbl>                  <dbl>
##  1               22 Amazonas          195000000000            38926678786
##  2                3 Antioquia        2714000000000           978388352690
##  3               25 Arauca            257000000000            51748020084
## 4                  2 Atlántico         1317000000000           139536459191
## 5                 33 Bogotá D.c.       4383000000000                     NA
## 6                  5 Bolivar           1381000000000            21531831266
## 7                 11 Boyacá             866000000000           109472194637
## 8                 29 Caldas             612000000000            86482421170
## 9                 21 Caquetá            377000000000            23916156000
## 10                 8 Casanare           277000000000            30786544250
## # ... with 23 more rows, and 16 more variables: Prop_productosCD_2021 <dbl>,
## #   Cumplimiento_A1_2020-2021-06 <dbl>, Cumplimiento_A2_2020-2021-06 <dbl>,
## #   PropPortafolio <dbl>, No_Inscritos <dbl>, T_adq_Recet <dbl>,
## #   Grupo_hclus <int>, Presupuesto_2020_mn <dbl>,
## #   Presupuesto_Salud_2020_mn <dbl>, Prop_productosCD_2021_mn <dbl>,
## #   Cumplimiento_A1_2020-2021-06_mn <dbl>,
## #   Cumplimiento_A2_2020-2021-06_mn <dbl>, PropPortafolio_mn <dbl>, ...
```

```
# group_by(Grupo_hclus) %>%
#   summarise(across(!matches('Departamento'), function(x) {
#     sum(ifelse(x > mean(x), 1, 0)) / n()
#   }))

fig1 <- trans_data1 %>%
  plot_ly(x = ~No_Inscritos, y = ~`Cumplimiento_A2_2020-2021-06`,
          z = ~PropPortafolio, split = ~Grupo_hclus,
          colors = colors, name = ~Grupo_hclus, text = ~Departamentos,
          hovertemplate = "%{text}<br>N.° inscritos: %{x}<br>Cumplimiento A2: %{y}<br>Prop. portafolio:
```

```
if (knitr::is_html_output()) {
  fig1 %>% add_markers()
}
```