Análisis Exploratorio

Introducción

El objetivo de este trabajo es llevar a cabo un análisis exploratorio de los datos contenidos en el archivo `pacientes_cancer.csv`. Para ello, será necesario realizar las siguientes tareas:

1. Entender qué representa cada variable y **eliminar las columnas innecesarias**.
2. **Distinguir entre columnas según el tipo** de variable (numérica, categórica, etc.)
3. Buscar **valores nulos** y **registros duplicados**.
4. Calcular **estadísticos descriptivos**:
    - Variables numéricas: *media*, *mediana*, *moda*, *cuartiles* y valores *máximo* y *mínimo*.
    - Variables categóricas: número de *categorías diferentes*, el valor de las *más y menos frecuentes*, y sus *frecuencias asociadas*.
5. Representar la **distribución** de los datos de la variable (*histograma* y *diagramas de cajas*).

Para realizar todas estas tareas, vamos a necesitar los siguientes paquetes:

```
library(tidyverse)
library(skimr)
library(reshape2)
```

Análisis Exploratorio

Primero, cargamos los datos y observamos las primeras filas de las primeras cinco columnas:

```
df <- read.csv("cancer_gene_expression_dataset.csv")
df |> select(1:6) |> head()
```

```
##   Patient_ID     TP53    BRCA1    BRCA2     EGFR     KRAS
## 1       P001 2.621229 2.294844 2.285151 3.162488 2.602772
## 2       P002 2.644670 2.086924 2.105762 2.326020 3.698626
## 3       P003 1.942043 1.781996 2.277527 1.855014 2.667775
## 4       P004 2.533690 2.987298 2.059414 2.213807 2.869632
## 5       P005 2.380817 2.821170 3.066866 2.706125 2.098054
## 6       P006 2.178189 3.109436 3.635576 3.039668 1.860386
```

Cada variable representa el nivel de expresión de un gen concreto. Podemos prescindir de la columna `Patient_ID`, ya que no aporta información relevante (es el índice + 1).

```
df <- df %>% select(-Patient_ID)

df |> select(1:5) |> head()
```

```
##       TP53    BRCA1    BRCA2     EGFR     KRAS
## 1 2.621229 2.294844 2.285151 3.162488 2.602772
## 2 2.644670 2.086924 2.105762 2.326020 3.698626
## 3 1.942043 1.781996 2.277527 1.855014 2.667775
## 4 2.533690 2.987298 2.059414 2.213807 2.869632
## 5 2.380817 2.821170 3.066866 2.706125 2.098054
## 6 2.178189 3.109436 3.635576 3.039668 1.860386
```

Estructura

A continuación, comprobamos el tipo de los datos de cada variable:

```
df |> skim() |> summary()
```

Table 1: Data summary

Name	df
Number of rows	100
Number of columns	50
Column type frequency:	
numeric	50
Group variables	None

Tenemos 100 registros con 50 variables numéricas. Veamos algunos estadísticos descriptivos de las primeras columnas:

```
df |> skim() |> yank("numeric") |> select(-c(complete_rate, hist))
```

Variable type: numeric

skim_variable	n_missing	mean	sd	p0	p25	p50	p75	p100
TP53	0	0.56	1.77	-2.85	-0.71	0.29	2.30	3.80
BRCA1	0	0.62	1.77	-2.81	-0.73	0.40	2.34	3.66
BRCA2	0	0.65	1.67	-2.57	-0.56	0.22	2.27	3.64
EGFR	0	0.65	1.74	-2.33	-0.66	0.18	2.30	3.92
KRAS	0	0.65	1.77	-2.18	-0.82	0.20	2.52	4.14
MYC	0	0.61	1.67	-2.61	-0.82	0.32	2.18	4.14
PTEN	0	0.59	1.77	-2.58	-0.65	0.20	2.38	3.60
RB1	0	0.65	1.84	-3.07	-0.81	0.18	2.33	4.65
APC	0	0.62	1.69	-2.69	-0.86	0.43	2.20	3.50
VHL	0	0.77	1.78	-2.21	-0.69	0.31	2.44	4.21
CDKN2A	0	0.97	1.40	-0.80	-0.26	0.38	2.27	3.93
PIK3CA	0	1.01	1.39	-1.01	-0.15	0.45	2.47	3.76
BRAF	0	1.05	1.37	-1.09	-0.07	0.44	2.46	3.61
NRAS	0	0.90	1.41	-1.25	-0.28	0.38	2.30	3.75
ERBB2	0	1.05	1.30	-0.76	-0.03	0.41	2.46	3.48
ALK	0	1.02	1.34	-1.05	-0.08	0.43	2.28	4.21
RET	0	1.03	1.42	-1.22	-0.19	0.42	2.39	3.82

skim_variable	n_missing	mean	sd	p0	p25	p50	p75	p100
MET	0	0.96	1.41	-1.42	-0.12	0.34	2.36	3.76
FGFR1	0	0.95	1.18	-1.28	-0.02	0.50	2.14	3.17
FGFR2	0	1.09	1.38	-1.11	-0.05	0.51	2.52	4.05
CDK4	0	1.08	1.42	-1.20	-0.01	0.54	2.49	4.24
CDK6	0	0.99	1.54	-1.47	-0.20	0.30	2.53	4.47
CCND1	0	0.94	1.40	-0.93	-0.13	0.33	2.35	3.76
CCNE1	0	1.02	1.55	-1.04	-0.18	0.38	2.39	4.38
MDM2	0	0.94	1.40	-1.31	-0.22	0.46	2.28	4.02
BCL2	0	0.98	1.52	-1.59	-0.16	0.41	2.48	3.96
BAX	0	0.92	1.51	-1.27	-0.30	0.21	2.47	3.85
CASP3	0	1.09	1.40	-1.21	0.05	0.46	2.66	3.93
CASP9	0	1.02	1.48	-1.04	-0.15	0.46	2.50	4.17
FAS	0	0.99	1.41	-1.59	-0.15	0.42	2.49	3.72
VEGFA	0	1.02	1.49	-0.89	-0.08	0.34	2.42	4.72
VEGFR2	0	0.88	1.44	-1.12	-0.27	0.29	2.36	3.75
HIF1A	0	0.94	1.41	-1.06	-0.24	0.44	2.25	4.14
STAT3	0	0.94	1.42	-1.42	-0.22	0.37	2.34	3.89
JAK2	0	0.97	1.41	-0.96	-0.18	0.30	2.45	3.96
NOTCH1	0	0.98	1.48	-1.46	-0.16	0.35	2.43	4.39
WNT1	0	1.00	1.45	-1.10	-0.17	0.42	2.41	4.35
CTNNB1	0	0.90	1.40	-0.98	-0.18	0.22	2.36	3.58
SMAD4	0	0.97	1.38	-1.05	-0.11	0.44	2.48	3.92
TGFB1	0	0.98	1.50	-1.44	-0.16	0.35	2.63	3.93
ATM	0	0.55	1.02	-0.94	-0.07	0.20	0.88	3.00
CHEK2	0	0.61	1.14	-1.46	-0.14	0.28	1.32	3.31
MLH1	0	0.58	1.04	-1.03	-0.11	0.22	1.08	3.03
MSH2	0	0.49	0.99	-0.78	-0.17	0.14	0.74	3.37
PALB2	0	0.52	1.11	-1.26	-0.26	0.19	1.05	3.26
RAD51	0	0.53	1.10	-1.12	-0.27	0.22	0.98	3.58
NF1	0	0.47	1.09	-1.35	-0.24	0.16	0.93	3.25
TSC1	0	0.49	1.18	-1.50	-0.41	0.11	1.23	3.94
TSC2	0	0.58	0.92	-0.78	-0.06	0.32	0.95	3.14
STK11	0	0.55	1.09	-0.93	-0.20	0.17	1.26	3.07

Podemos ver la media, la desviación estándar, el mínimo, el máximo y los cuartiles, incluyendo la mediana. Comprobemos si hay registros duplicados:

```
duplic <- duplicated(df) |> sum()
paste0("Registros duplicados: ", duplic) |> print()
```

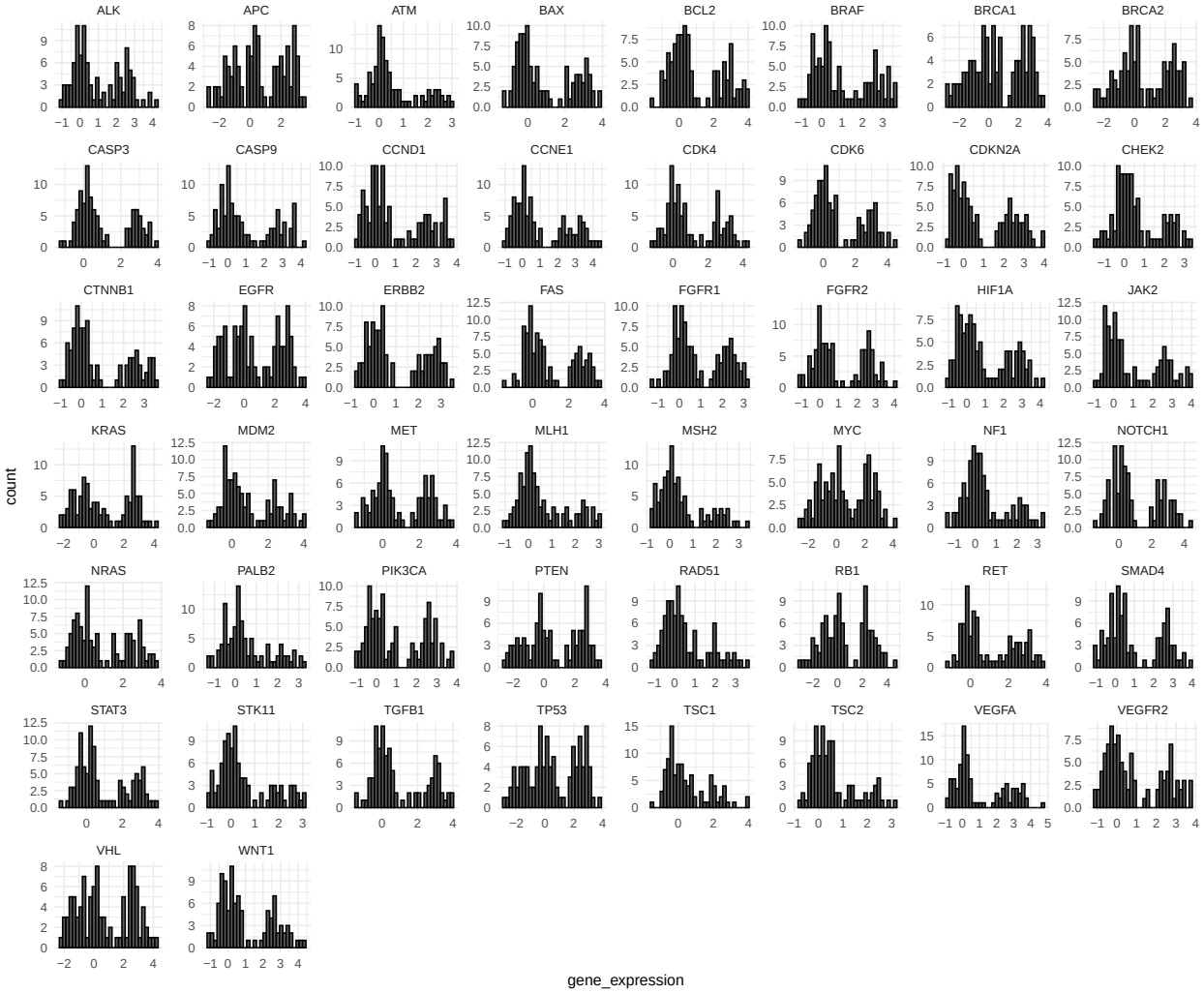
```
## [1] "Registros duplicados: 0"
```

Afortunadamente, no hay valores nulos ni registros duplicados. Además, todos los datos están en formato numérico, así que no es necesario limpiarlos.

Distribución

Podemos visualizar la distribución de los datos de cada variable con histogramas:

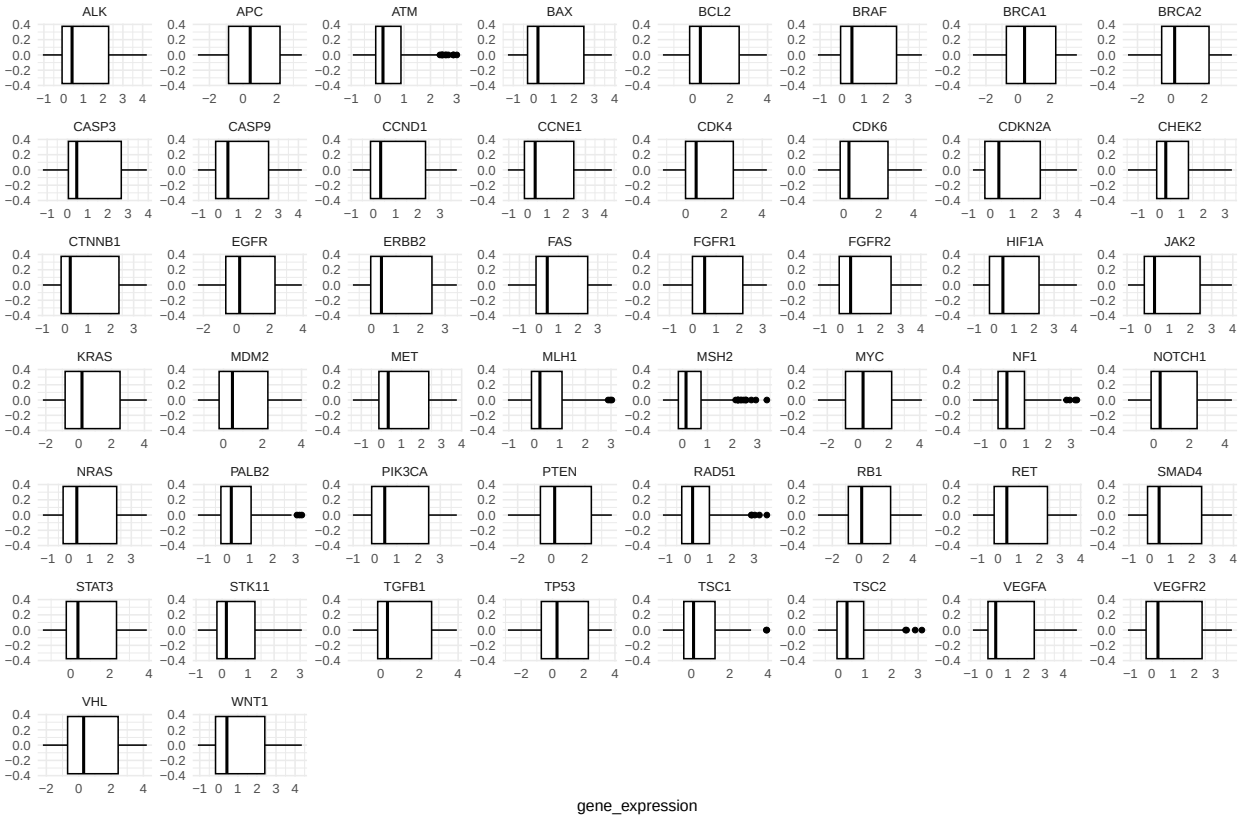
```
df |>
    pivot_longer(
        cols = everything(),
        values_to = "gene_expression",
        names_to = "gene"
        ) |>
    ggplot(aes(x = gene_expression))+
    geom_histogram(color = "black")+
    facet_wrap(~gene, scales = "free")+
    theme_minimal()
```



También podemos ver los diagramas de cajas correspondientes para inspeccionar visualmente los cuartiles y el rango. Así, podemos detectar la presencia de valores anómalos:

```
df |>
    pivot_longer(
        cols = everything(),
        values_to = "gene_expression",
        names_to = "gene"
        ) |>
```

```
    ggplot(aes(x = gene_expression))+
    geom_boxplot(color = "black", outliers = TRUE, orientation = "y")+
    facet_wrap(~gene, scales = "free")+
    theme_minimal()
```



Se observa una distribución similar para todas las variables, aproximadamente bimodal. Unas pocas contienen valores anómalos. Por ello, antes de implementar el algoritmo de agrupación correspondiente, será necesario escalar los datos utilizando un estimador que sea robusto con los valores anómalos.

Guardamos los datos:

```
write.csv(df, "cancer_gene_expression_dataset_clean.csv", row.names = FALSE)
```

Análisis de Componentes Principales

Como el objetivo de este trabajo es hacer una agrupación de los pacientes por la expresión de distintos genes, conviene realizar un análisis de componentes principales (PCA). Esta técnica reduce la dimensionalidad de los datos, haciendo posible juzgar visualmente si los datos se agrupan siguiendo algún patrón. Para llevar a cabo este análisis, cargamos los siguientes paquetes:

```
library(factoextra)
library(broom)
```

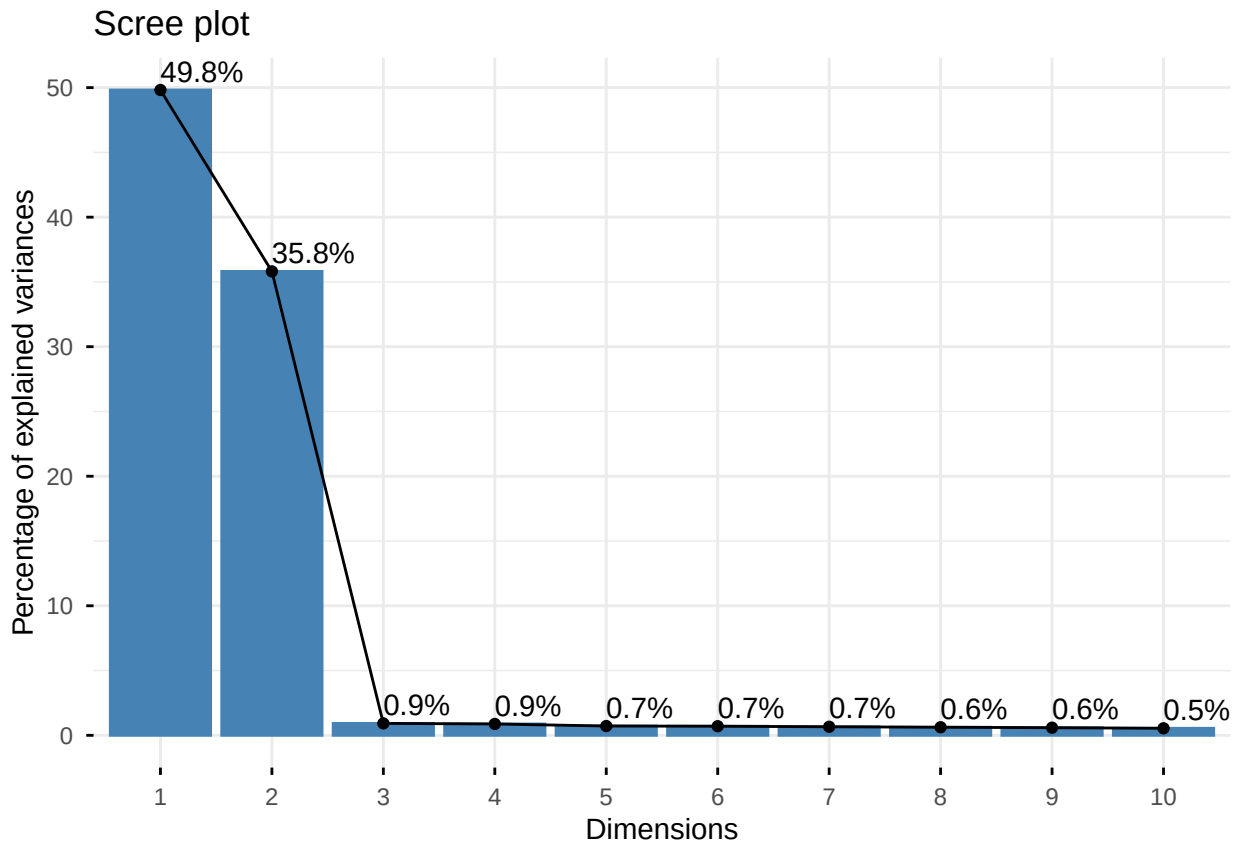
A continuación, implementamos el PCA y mostramos el valor de las componentes principales (PC) correspondientes a los primeros registros:

```r
pca <- df |> prcomp(scale. = TRUE) # PCA

pca_tidy <- pca |> tidy() |> # datos del PCA en forma de tabla
    pivot_wider(
        names_from = PC,
        names_prefix = "PC",
        values_from = value
    ) |>
    select(-row)

pca_tidy |> select(1:5) |> head()
```

```
## # A tibble: 6 x 5
##     PC1   PC2     PC3    PC4    PC5
##   <dbl> <dbl>   <dbl>  <dbl>  <dbl>
## 1 -5.72  1.45 -0.0565 -0.444 -0.848
## 2 -6.09  1.83 -0.749  -0.631 -0.792
## 3 -5.87  1.46 -0.107   0.553  0.162
## 4 -6.03  1.99  0.788   0.261 -0.437
## 5 -6.07  1.76 -0.972  -0.284  0.128
## 6 -5.35  1.37 -1.45   -0.183 -0.772
```

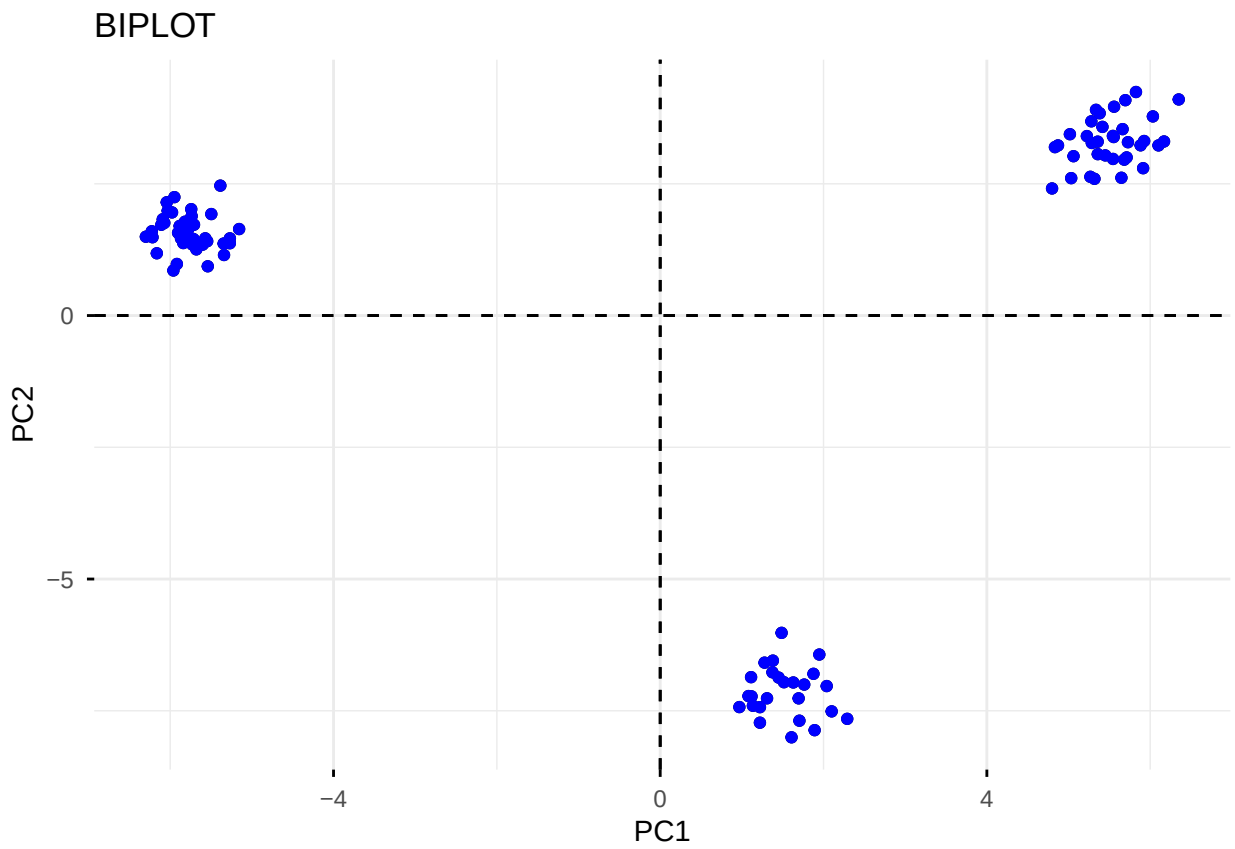
Antes de juzgar visualmente si los datos se agrupan siguiendo algún patrón, es necesario saber si las dos primeras PCs son suficientes. Para ello, hacemos un biplot:

```r
pca |> fviz_screeplot(addlabels = TRUE)
```



Las dos primeras PCs explican más del 80 de la varianza de los datos, por lo que podemos considerarlas suficiente. Veamos ahora el biplot:

```
pca |> fviz_pca_biplot(geom.ind = "point", geom.var = 0)+
    labs(
        x = "PC1",
        y = "PC2",
        title = "BIPLOT"
    )+
    geom_point(color = "blue")
```



Atendiendo a este gráfico, podemos ver cómo los datos parecen agruparse en tres clusters diferentes. Para profundizar en ello, a continuación vamos a agrupar los datos implementando el algoritmo *K-means*.