Analisis de conjunto de datos

Daniela Cuesta

1. Se ha cargado los datos correctamente y se ha seleccionado las variables numéricas adecuadas. Además, se ha convertido la variable V9 en un factor y se ha asignado a la variable clase

```
datos <- read.table("./ecoli.data",header = F)[,-1]
head(datos)
```

```
    V2   V3   V4  V5   V6   V7   V8 V9
1 0.49 0.29 0.48 0.5 0.56 0.24 0.35 cp
2 0.07 0.40 0.48 0.5 0.54 0.35 0.44 cp
3 0.56 0.40 0.48 0.5 0.49 0.37 0.46 cp
4 0.59 0.49 0.48 0.5 0.52 0.45 0.36 cp
5 0.23 0.32 0.48 0.5 0.55 0.25 0.35 cp
6 0.67 0.39 0.48 0.5 0.36 0.38 0.46 cp
```

```
datos.numericos <- datos[, sapply(datos, is.numeric)]

# Convertir la variable "V9" a factor y asignarla a "clase"
clase <- datos$V9 <- as.factor(datos$V9)
```

2. Se aplica un `summary(datos)`, esto generará un resumen estadístico de todas las variables en el conjunto de datos, específicamente `datos`

```
summary(datos)
```

```
       V2                V3              V4                V5        
 Min.   :0.0000   Min.   :0.16   Min.   :0.4800   Min.   :0.5000  
 1st Qu.:0.3400   1st Qu.:0.40   1st Qu.:0.4800   1st Qu.:0.5000  
 Median :0.5000   Median :0.47   Median :0.4800   Median :0.5000  
```

```
Mean   :0.5001   Mean   :0.50   Mean   :0.4955   Mean   :0.5015
3rd Qu.:0.6625   3rd Qu.:0.57   3rd Qu.:0.4800   3rd Qu.:0.5000
Max.   :0.8900   Max.   :1.00   Max.   :1.0000   Max.   :1.0000

      V6               V7               V8               V9
Min.   :0.000   Min.   :0.0300   Min.   :0.0000   cp     :143
1st Qu.:0.420   1st Qu.:0.3300   1st Qu.:0.3500   im     : 77
Median :0.495   Median :0.4550   Median :0.4300   pp     : 52
Mean   :0.500   Mean   :0.5002   Mean   :0.4997   imU    : 35
3rd Qu.:0.570   3rd Qu.:0.7100   3rd Qu.:0.7100   om     : 20
Max.   :0.880   Max.   :1.0000   Max.   :0.9900   omL    :  5
                                                  (Other):  4
```

Inferencia univariante

Se crea una lista **resultados_shapiro** donde almacenaremos los resultados de la prueba de normalidad para cada variable. Utilizamos un bucle **for** para iterar sobre cada columna en **datos.numericos**. En cada iteración, aplicamos **shapiro.test()** a la variable correspondiente y guardamos el resultado en la lista **resultados_shapiro**, utilizando el nombre de la variable como etiqueta.

Finalmente, utilizamos otro bucle **for** para mostrar los resultados de la prueba de normalidad para cada variable, imprimiendo el nombre de la variable y el resultado correspondiente.

```r
# Obtener las variables numéricas del conjunto de datos
datos.numericos <- datos[, sapply(datos, is.numeric)]

# Aplicar la prueba de normalidad de Shapiro-Wilk a cada variable
resultados_shapiro <- list()

for (i in 1:ncol(datos.numericos )) {
  variable <- datos.numericos [, i]
  resultado <- shapiro.test(variable)
  resultados_shapiro[[colnames(datos.numericos )[i]]] <- resultado
}

# Mostrar los resultados de la prueba de normalidad
for (i in 1:length(resultados_shapiro)) {
  variable <- names(resultados_shapiro[i])
  resultado <- resultados_shapiro[[i]]
  print(paste("Variable:", variable))
```

```
    print(resultado)
    cat("\n")
  }
```

```
[1] "Variable: V2"

    Shapiro-Wilk normality test

data:  variable
W = 0.97366, p-value = 8.231e-06


[1] "Variable: V3"

    Shapiro-Wilk normality test

data:  variable
W = 0.95098, p-value = 3.863e-09


[1] "Variable: V4"

    Shapiro-Wilk normality test

data:  variable
W = 0.15841, p-value < 2.2e-16


[1] "Variable: V5"

    Shapiro-Wilk normality test

data:  variable
W = 0.02915, p-value < 2.2e-16


[1] "Variable: V6"

    Shapiro-Wilk normality test

data:  variable
```

```
W = 0.98139, p-value = 0.0002423
```

```
[1] "Variable: V7"

    Shapiro-Wilk normality test

data:  variable
W = 0.95543, p-value = 1.431e-08
```

```
[1] "Variable: V8"

    Shapiro-Wilk normality test

data:  variable
W = 0.93293, p-value = 3.656e-11
```

También se puede aplicar para cada variables

```
shapiro.test(datos$V2)
```

```
    Shapiro-Wilk normality test

data:  datos$V2
W = 0.97366, p-value = 8.231e-06
```

Ninguno de los valores p obtenidos indica que alguna de las variables siga una distribución normal, ya que los valores obtenidos son extremadamente pequeños

Inferencia bivariante

Se realiza una prueba de correlación de Pearson entre las variables V2 y V3. **`datos$V2`** y **`datos$V3`** son las columnas correspondientes a esas variables en el conjunto de datos.

```
cor.test(datos$V2, datos$V3)
```

```
    Pearson's product-moment correlation

data:  datos$V2 and datos$V3
t = 9.333, df = 334, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.3656018 0.5357301
sample estimates:
      cor
0.4548053
```

```
cor.test(datos$V3, datos$V4)
```

```
    Pearson's product-moment correlation

data:  datos$V3 and datos$V4
t = 0.80133, df = 334, p-value = 0.4235
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.06348734  0.15009526
sample estimates:
       cor
0.04380447
```

```
cor.test(datos$V4, datos$V5)
```

```
    Pearson's product-moment correlation

data:  datos$V4 and datos$V5
t = 6.0006, df = 334, p-value = 5.118e-09
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.2120338 0.4054138
sample estimates:
     cor
0.311951
```

```r
cor.test(datos$V5, datos$V6)
```

```
    Pearson's product-moment correlation

data:  datos$V5 and datos$V6
t = -0.81821, df = 334, p-value = 0.4138
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.1509975  0.0625678
sample estimates:
        cor
-0.04472587
```

Según el valor p obtenido se puede observar si hay una correlación significativa entre las variables V2 y V3 y así con todas las variables

Inferencia multivariante: PCA

Se utiliza la función **`scale()`** para estandarizar las variables numéricas en el conjunto de datos. La estandarización asegura que todas las variables tengan media cero y desviación estándar uno.

Se utiliza la función **`prcomp()`** para realizar el análisis de Componentes Principales.

La función **`summary()`**. Proporciona información sobre los componentes principales

```r
#
# Seleccionar las variables numéricas para el PCA
#datos_numericos <- datos[, sapply(datos, is.numeric)]

# Estandarizar las variables
datos_estandarizados <- scale(datos.numericos)

# Realizar el PCA
pca <- prcomp(datos_estandarizados)

# Resumen del PCA
summary(pca)
```

```
Importance of components:
                          PC1    PC2    PC3    PC4     PC5     PC6     PC7
Standard deviation     1.4851 1.2088 1.0961 0.9258 0.81819 0.69185 0.35556
Proportion of Variance 0.3151 0.2087 0.1716 0.1225 0.09563 0.06838 0.01806
Cumulative Proportion  0.3151 0.5238 0.6955 0.8179 0.91356 0.98194 1.00000
```