K-vecinos más cercanos (kNN) - iris

Karina Itzel Rodríguez Conde

2022-05-13

Introducción

El método de K vecinos más cercanos es también conocido como **kNN** y consiste en un método de clasificación no paramétrico que se basa en buscar para una observación, sus k vecinos más cercanos, es decir, aquellas observaciones que están más cercanas a una determinada distancia de dicha observación.

Matriz de datos

Para esta práctica, se trabajó con la matriz **iris**, la cual fue extraída del paquete *datos* que se encuentra precargada en R y muestra a tres diferentes especies de flor: setosa, virginica y versicolor.

1.- Paquetería y librería a utilizar

```
install.packages("MASS")
library(MASS)
```

2.- Cargar los datos iris

```
Z <- as.data.frame(iris)
```

Exploración de la matriz

1.- Dimensión

```
dim(Z)
```

```
## [1] 150   5
```

La base de datos cuenta con 150 observaciones y 5 variables.

2.- Nombre de las variables

```
colnames(Z)
```

```
## [1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"  "Species"
```

3.- Tipo de variables

```
str(Z)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

4.- Saber si existen datos nulos

```
anyNA(Z)
```

```
## [1] FALSE
```

Esta base de datos no contiene datos nulos.

Tratamiento de la matriz

1.- Definir la matriz de datos y la variable respuesta con las clasificaciones

```
x <- Z[,1:4]
y <- Z[,5]
```

2.- Se definen las variables y observaciones

```
n<-nrow(x)
p<-ncol(x)
```

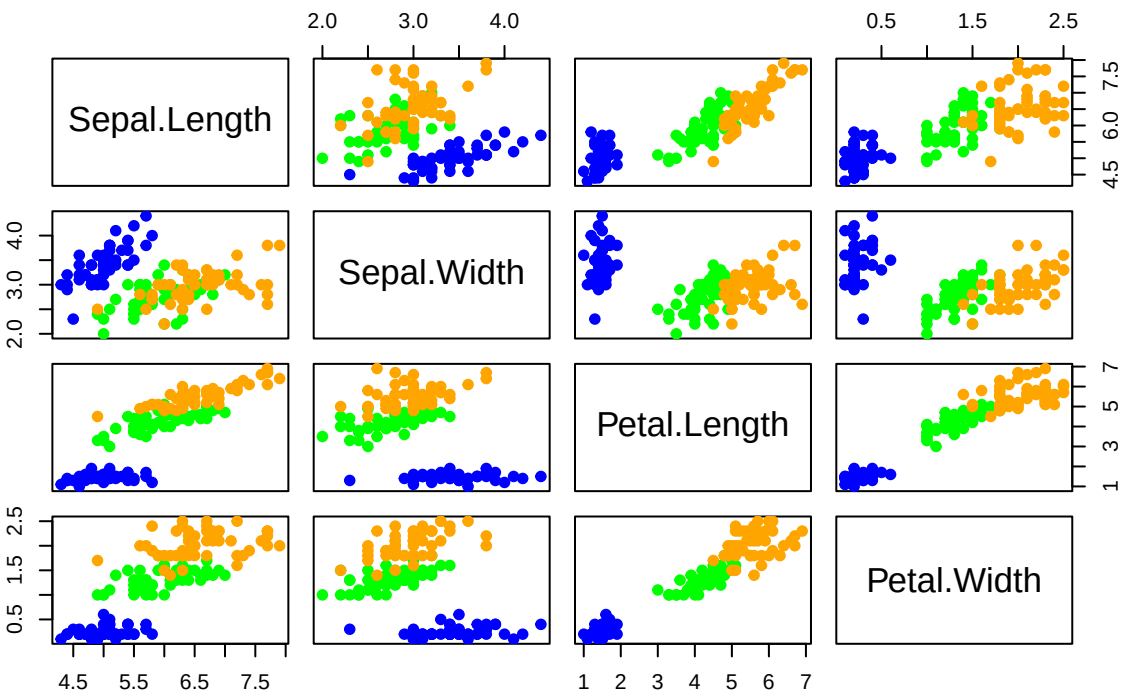
3.- Gráfico scatter plot

Creación de un vector de colores

```
col.iris<-c("blue","green","orange")[y]
col.iris
```

```
##   [1] "blue"   "blue"   "blue"   "blue"   "blue"   "blue"   "blue"   "blue"
##   [9] "blue"   "blue"   "blue"   "blue"   "blue"   "blue"   "blue"   "blue"
##  [17] "blue"   "blue"   "blue"   "blue"   "blue"   "blue"   "blue"   "blue"
##  [25] "blue"   "blue"   "blue"   "blue"   "blue"   "blue"   "blue"   "blue"
##  [33] "blue"   "blue"   "blue"   "blue"   "blue"   "blue"   "blue"   "blue"
##  [41] "blue"   "blue"   "blue"   "blue"   "blue"   "blue"   "blue"   "blue"
##  [49] "blue"   "blue"   "green"  "green"  "green"  "green"  "green"  "green"
##  [57] "green"  "green"  "green"  "green"  "green"  "green"  "green"  "green"
##  [65] "green"  "green"  "green"  "green"  "green"  "green"  "green"  "green"
##  [73] "green"  "green"  "green"  "green"  "green"  "green"  "green"  "green"
##  [81] "green"  "green"  "green"  "green"  "green"  "green"  "green"  "green"
##  [89] "green"  "green"  "green"  "green"  "green"  "green"  "green"  "green"
##  [97] "green"  "green"  "green"  "green"  "orange" "orange" "orange" "orange"
## [105] "orange" "orange" "orange" "orange" "orange" "orange" "orange" "orange"
## [113] "orange" "orange" "orange" "orange" "orange" "orange" "orange" "orange"
## [121] "orange" "orange" "orange" "orange" "orange" "orange" "orange" "orange"
## [129] "orange" "orange" "orange" "orange" "orange" "orange" "orange" "orange"
## [137] "orange" "orange" "orange" "orange" "orange" "orange" "orange" "orange"
## [145] "orange" "orange" "orange" "orange" "orange" "orange"
```

```
pairs(x, main="Data set Iris", pch=19,col=col.iris)
```



Aplicación del kNN

1.- Paquetería y librería a utilizar

```
install.packages("class")
library(class)
```

2.- Se fija una "semilla" para tener valores iguales

```
set.seed(1000)
```

3.- Creación de los ciclos para k=1 hasta k=20.

Se selecciona el valor de k que tenga el error más bajo.

3.1.- Inicialización de una lista vacía de tamaño 20

```
knn.class<-vector(mode="list",length=20)
knn.tables<-vector(mode="list", length=20)
```

3.2.- Clasificaciones erróneas

```
knn.mis<-matrix(NA, nrow=20, ncol=1)
knn.mis
```

```
##       [,1]
##  [1,]   NA
##  [2,]   NA
##  [3,]   NA
##  [4,]   NA
##  [5,]   NA
##  [6,]   NA
##  [7,]   NA
##  [8,]   NA
##  [9,]   NA
## [10,]   NA
## [11,]   NA
## [12,]   NA
## [13,]   NA
## [14,]   NA
## [15,]   NA
## [16,]   NA
## [17,]   NA
## [18,]   NA
## [19,]   NA
## [20,]   NA
```

```
for(k in 1:20){
  knn.class[[k]]<-knn.cv(x,y,k=k)
  knn.tables[[k]]<-table(y,knn.class[[k]])
  # la suma de las clasificaciones menos las correctas
  knn.mis[k]<- n-sum(y==knn.class[[k]])
}

knn.mis
```

```
##       [,1]
##  [1,]    6
##  [2,]    7
##  [3,]    6
##  [4,]    6
##  [5,]    5
##  [6,]    4
##  [7,]    5
##  [8,]    5
##  [9,]    4
## [10,]    5
## [11,]    4
## [12,]    6
## [13,]    5
## [14,]    3
## [15,]    4
## [16,]    5
## [17,]    4
## [18,]    3
## [19,]    3
## [20,]    4
```

4.- Número óptimo de k-vecinos

```
which(knn.mis==min(knn.mis))
```

```
## [1] 14 18 19
```

5.- Visualización de los números óptimos de k-vecinos

```
knn.tables[[14]]
```

```
##
## y            setosa versicolor virginica
##   setosa         50          0         0
##   versicolor      0         48         2
##   virginica       0          1        49
```

```
knn.tables[[18]]
```

```
##
## y            setosa versicolor virginica
##   setosa         50          0         0
##   versicolor      0         48         2
##   virginica       0          1        49
```

```
knn.tables[[19]]
```

```
##
## y            setosa versicolor virginica
##   setosa         50          0         0
##   versicolor      0         48         2
##   virginica       0          1        49
```

El más eficiente es k=14

6.- Se señala el k más eficiente

```
k.opt<-14
knn.cv.opt<-knn.class[[k.opt]]
knn.cv.opt
```

```
##   [1] setosa     setosa     setosa     setosa     setosa     setosa
##   [7] setosa     setosa     setosa     setosa     setosa     setosa
##  [13] setosa     setosa     setosa     setosa     setosa     setosa
##  [19] setosa     setosa     setosa     setosa     setosa     setosa
##  [25] setosa     setosa     setosa     setosa     setosa     setosa
##  [31] setosa     setosa     setosa     setosa     setosa     setosa
##  [37] setosa     setosa     setosa     setosa     setosa     setosa
##  [43] setosa     setosa     setosa     setosa     setosa     setosa
##  [49] setosa     setosa     versicolor versicolor versicolor versicolor
##  [55] versicolor versicolor versicolor versicolor versicolor versicolor
##  [61] versicolor versicolor versicolor versicolor versicolor versicolor
##  [67] versicolor versicolor versicolor versicolor virginica  versicolor
##  [73] versicolor versicolor versicolor versicolor versicolor versicolor
##  [79] versicolor versicolor versicolor versicolor versicolor virginica
##  [85] versicolor versicolor versicolor versicolor versicolor versicolor
##  [91] versicolor versicolor versicolor versicolor versicolor versicolor
```

```
##  [97] versicolor versicolor versicolor versicolor virginica  virginica
## [103] virginica  virginica  virginica  virginica  versicolor virginica
## [109] virginica  virginica  virginica  virginica  virginica  virginica
## [115] virginica  virginica  virginica  virginica  virginica  virginica
## [121] virginica  virginica  virginica  virginica  virginica  virginica
## [127] virginica  virginica  virginica  virginica  virginica  virginica
## [133] virginica  virginica  virginica  virginica  virginica  virginica
## [139] virginica  virginica  virginica  virginica  virginica  virginica
## [145] virginica  virginica  virginica  virginica  virginica  virginica
## Levels: setosa versicolor virginica
```

7.- Tabla de contingencia con las clasificaciones buenas y malas

```
knn.tables[[k.opt]]
```

```
##
## y            setosa versicolor virginica
##   setosa         50          0         0
##   versicolor      0         48         2
##   virginica       0          1        49
```

8.- Cantidad de observaciones mal clasificadas

```
knn.mis[k.opt]
```

```
## [1] 3
```

Se puede observar que hay 3 malas clasificaciones, puesto que están fuera de la diagonal.

9.- Error de clasificación (MR)

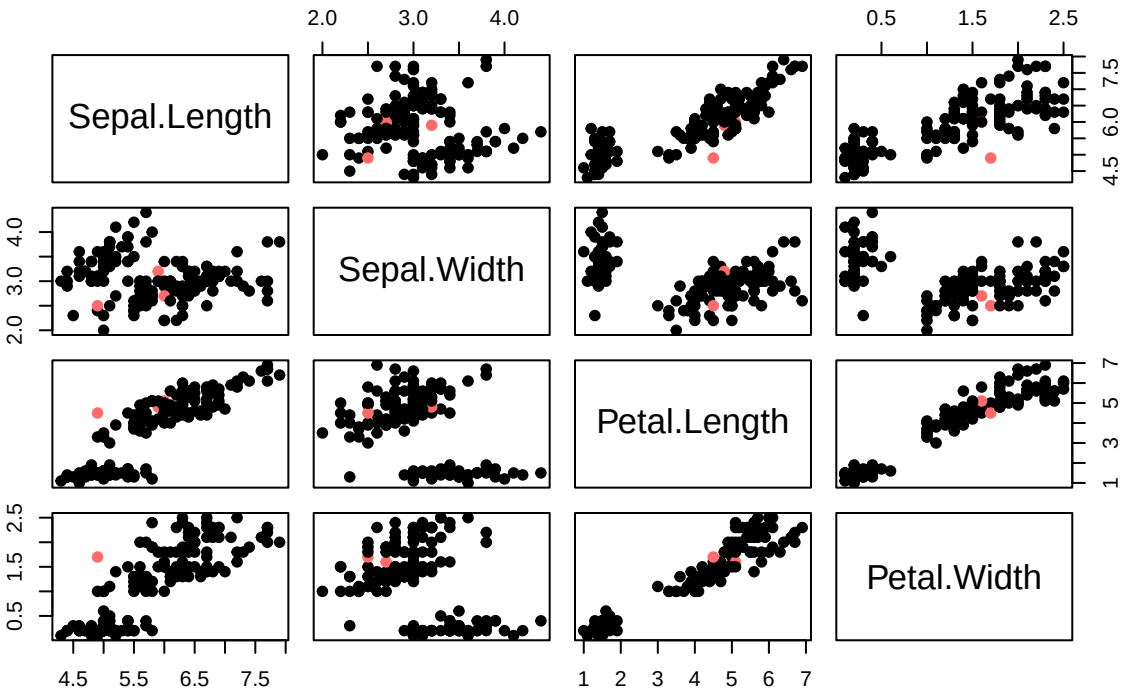
```
knn.mis[k.opt]/n
```

```
## [1] 0.02
```

El error de clasificación es de un 0.02%

10.- Gráfico de clasificaciones correctas y erróneas

```
col.knn.iris<-c("indianred1","black")[1*(y==knn.cv.opt)+1]
pairs(x, main="Clasificación kNN de Iris",
      pch=19, col=col.knn.iris)
```



Aquellas observaciones marcadas en rojo son las mal clasificadas.