KNN para el análisis de variables cuantitativas (regresión)

Contents

1 Predicción con un regresor **1**

2 Predicción con m regresores **3**

3 Normalización de los regresores **4**

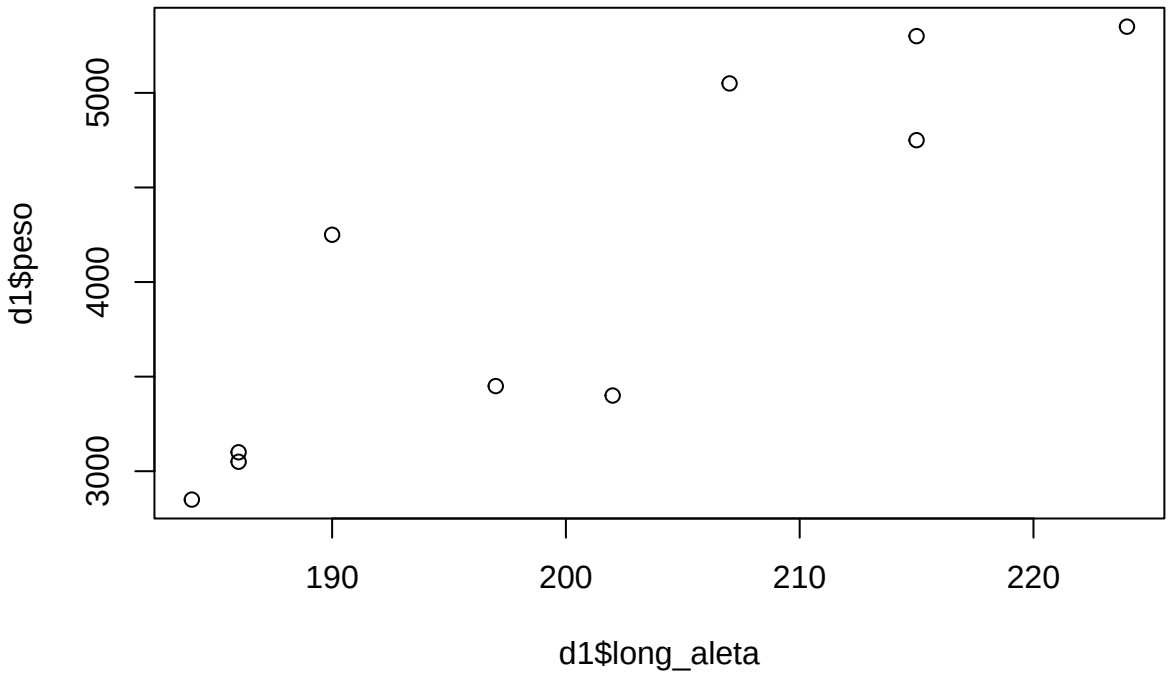
4 Regresores cualitativos **5**

5 Regresores cualitativos-2 **6**

1 Predicción con un regresor

El objetivo es predecir los valores que tomará una variable cuantitativa conocidos los valores de otra u otras variables. Por ejemplo, se quiere predecir el peso de los pingüinos si se conoce la longitud de su aleta. El peso es la **variable respuesta** y la longitud de la aleta es el **regresor** (también se conoce como variable independiente, cofactor, predictor,...). Cuando la variable respuesta es cuantitativa se suele denominar **problema de regresión**.

```
## se leen los datos
d = read.csv("datos/pinguinos.csv", sep = ";")
# por simplicidad se utilizan solo 10 datos seleccionados de manera aleatoria
set.seed(99)
pos = runif(10, min = 1, max = nrow(d))
# se seleccionan los 10 observaciones de la variable respuesta y el regresor
d1 = d[pos,c("peso","long_aleta")]
# gráfico de dispersion
plot(d1$long_aleta, d1$peso)
```



En este caso se quiere predecir el peso de un pingüino cuya aleta mide 210 mm. Para ello se va a utilizar el algoritmo **K-Nearest Neighbors (KNN)**. La idea es calcular la predicción como la media de los k-valores más cercanos a 210 mm. Primero se calcula la distancia al regresor que se va a predecir:

```
# se calcula la distancia y se guarda en d1
d1$dist = abs(d1$long_aleta - 210)
# se ordenan todas las observaciones de d1 en funcion de dist
(orden1 = sort(d1$dist, index.return = T))
```

```
## $x
##  [1]  3  5  5  8 13 14 20 24 24 26
##
## $ix
##  [1]  5  1  7  4  6  3  8  2  9 10
```

Ahora se ordenan los datos en función de la distancia al regresor predicho:

```
(d1s = d1[orden1$ix,])
```

```
##     peso long_aleta dist
## 178 5050        207    3
## 195 5300        215    5
## 223 4750        215    5
## 330 3400        202    8
## 321 3450        197   13
## 228 5350        224   14
## 98  4250        190   20
## 38  3100        186   24
## 119 3050        186   24
## 59  2850        184   26
```

Ya se puede calcular la predicción:

- Si K = 1 se utiliza el valor más cercano. Luego la predicción para el peso es 5050 g.
- Si K = 2 la predicción es la media entre 5050 y 5300 = 5175 g.

- Si K = 3 la predicción es la media entre 5050, 5300 y 4750 = 5033.3333333 g.
- Y así para otros valores de K.

2 Predicción con m regresores

Se quiere predecir el peso de un pingüino con long_pico = 40 mm, prof_pico = 18 mm y long_aleta = 210 mm, es decir, se tiene la información de tres regresores. En el caso de que se tengan dos o más regresores el algoritmo sigue siendo el mismo: la predicción es la media de los k valores más cercanos. En general se utiliza la distancia euclídea para calcular la distancia entre cada observación y los datos a predecir.

$$x = (x_1, x_2, ..., x_m), \quad y = (y_1, y_2, ..., y_m)$$

$$d(x, y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_m - y_m)^2}$$

```
knn_dist = function(x,y){
  # funcion que calcula la distancia euclidea entre los vectores x e y
  dist = sqrt(sum((x-y)^2))
  return(dist)
}
```

Con la función anterior se calcula la distancia entre los regresores y el dato a predecir:

```
# se seleccionan los regresores correspondientes
d2 = d[pos,c("peso","long_pico","prof_pico","long_aleta")]
# regresor a predecir
xp2 = c(40,18,210)
# se calculan las distancias
d2$dist = 0
for (ii in 1:10){
  xii = d2[ii,2:4]
  d2$dist[ii] = knn_dist(xii,xp2)
}
```

Ahora se ordenan los datos en función de la distancia al regresor predicho:

```
orden2 = sort(d2$dist, index.return = T)
(d2s = d2[orden2$ix,])
```

```
##     peso long_pico prof_pico long_aleta      dist
## 178 5050      45.1      14.5        207  6.874591
## 195 5300      45.2      15.8        215  7.541883
## 223 4750      45.2      13.8        215  8.347455
## 330 3400      43.5      18.1        202  8.732697
## 228 5350      50.0      15.9        224 17.332340
## 321 3450      52.2      18.8        197 17.846008
## 98  4250      37.8      20.0        190 20.219792
## 38  3100      36.0      18.5        186 24.336187
## 119 3050      35.2      15.9        186 24.565219
## 59  2850      36.4      17.1        184 26.263473
```

- Si K = 1 la predicción para el peso es 5050 g.
- Si K = 2 la predicción es la media entre 5050 y 5300 = 5175 g.
- Si K = 3 la predicción es la media entre 5050, 5300 y 4750 = 5033.3333333 g.
- Y así sucesivamente.

3 Normalización de los regresores

En el ejemplo anterios se han utilizado 3 variables para calcular la distancia:

- long_pico, que toma valores en torno a 40 mm;
- prof_pico, que toma valores alrededor de 20 mm;
- long_aleta, con valores alrededor de 200 mm.

Por tanto, la longitud de la aleta va a tener más influencia en la magnitud de la distancia que las otras dos variables. En cierta manera se desperdicia la información aportada por long_pico y prof_pico. Por ello se suelen normalizar los regresores, de manera que todos ellos aporten información en las mismas condiciones. La manera tradicional de reescalar variables en KNN es la normalización min-max. Este proceso transforma una variable cualquiera de manera que pasa a tomar valores en el rango 0-1. La fórmula que consigue dicha transformación es:

$$x = (x_1, x_2, ..., x_m) \Rightarrow x^* = (x_1^*, x_2^*, ..., x_m^*)$$

$$x_i^* = \frac{x_i - min(x)}{max(x) - min(x)}$$

Con esta transformación se consique que $min(x^*) = 0$ y $max(x^*) = 1$.

Se va a repetir el ejemplo anterior pero con las variables normalizadas entre 0 y 1. Primero se define la función de normalización:

```
knn_normaliza  = function(x, min_x = min(x), max_x = max(x)){
  # knn_normaliza(x,min_x,max_x): se normaliza x utilizando min_x y max_x
  # knn_normaliza(x): se normaliza x utilizando el min y el max de x
  x1 = (x - min_x)/(max_x - min_x)
  return(x1)
}
```

Se normaliza:

```
# se crea la base de datos
d3 = d[pos,c("peso","long_pico","prof_pico","long_aleta")]
# se incluyen los regresores normalizados
d3$long_pico1 = knn_normaliza(d3$long_pico)
d3$prof_pico1 = knn_normaliza(d3$prof_pico)
d3$long_aleta1 = knn_normaliza(d3$long_aleta)
```

También se tiene que normalizar el dato que se quiere predecir:

```
xp3 = c(0,0,0)
xp3[1] = knn_normaliza(40, min(d3$long_pico),max(d3$long_pico))
xp3[2] = knn_normaliza(18, min(d3$prof_pico), max(d3$prof_pico))
xp3[3] = knn_normaliza(210, min(d3$long_aleta), max(d3$long_aleta))
xp3
```

```
## [1] 0.2823529 0.6774194 0.6500000
```

Y ahora se calculan las distancias y se ordena:

```
d3$dist = 0
for (ii in 1:10){
  xii = d3[ii,5:7]
  d3$dist[ii] = knn_dist(xii,xp3)
}
```

```r
orden3 = sort(d3$dist, index.return = T)
(d3s = d3[orden3$ix,])
```

```
##     peso long_pico prof_pico long_aleta long_pico1 prof_pico1 long_aleta1
## 330 3400      43.5      18.1        202 0.48823529  0.6935484       0.450
## 195 5300      45.2      15.8        215 0.58823529  0.3225806       0.775
## 98  4250      37.8      20.0        190 0.15294118  1.0000000       0.150
## 178 5050      45.1      14.5        207 0.58235294  0.1129032       0.575
## 38  3100      36.0      18.5        186 0.04705882  0.7580645       0.050
## 59  2850      36.4      17.1        184 0.07058824  0.5322581       0.000
## 119 3050      35.2      15.9        186 0.00000000  0.3387097       0.050
## 223 4750      45.2      13.8        215 0.58823529  0.0000000       0.775
## 228 5350      50.0      15.9        224 0.87058824  0.3387097       1.000
## 321 3450      52.2      18.8        197 1.00000000  0.8064516       0.325
##          dist
## 330 0.2874851
## 195 0.4848706
## 98  0.6089382
## 178 0.6436641
## 38  0.6495129
## 59  0.6988677
## 119 0.7446123
## 223 0.7537148
## 228 0.7637048
## 321 0.7983055
```

- Si K = 1 la predicción para el peso es 3400 g.
- Si K = 2 la predicción es la media entre 3400 y 5300 = 4350 g.
- Si K = 3 la predicción es la media entre 3400, 5300 y 4250 = 4316.6666667 g.

4 Regresores cualitativos

La variable respuesta obligatoriamente tiene que ser cuantitativa ya que estamos en un problema de regresión. Hasta el momento sólo hemos utilizado regresores cuantitativos pero también pueden ser cualitativos. Por ejemplo, se va a introducir en el análisis el regresor *especie*. En este caso se quiere predecir el peso de un pingüino con long_pico = 40 mm, prof_pico = 18 mm, long_aleta = 210 mm y especie = Gentoo. Para poder aplicar el algoritmo KNN se necesita calcular la distancia entre los datos: ¿como se calcula la distancia entre Adelie, Gentoo y Chinstrap? La solución consiste en introducir variables auxiliares 0-1. Se van a crear tres variables auxiliares, una por nivel de la variable factor: Adelie1, Gentoo1 y Chinstrap1.

- Adelie1 = 1 si especie = Adelie, y Adelie1 = 0 si especie $\neq$ Adelie.
- Chinstrap1 = 1 si especie = Chinstrap, y Chinstrap1 = 0 si especie $\neq$ Chinstrap.
- Gentoo1 = 1 si especie = Gentoo, y Gentoo1 = 0 si especie $\neq$ Gentoo.

Por tanto, el dato que se quiere predecir es:

```r
# se crea el dato a predecir. Los valores corresponden a:
# long_pico, prof_pico, long_aleta, Adelie1, Chinstrap1, Gentoo1
(xp4 = c(xp3,0,0,1))
```

```
## [1] 0.2823529 0.6774194 0.6500000 0.0000000 0.0000000 1.0000000
```

Se crea la base de datos:

```r
# tomamos las variables que ya están normalizadas
d4 = d3[,c("peso","long_pico1","prof_pico1","long_aleta1")]
```

```r
# y se añade el factor especie
d4$especie = factor(d$especie[pos])
```

```r
# se añaden las variables auxiliares:
d4$Adelie1 = ifelse(d4$especie == "Adelie", 1,0)
d4$Chinstrap1 = ifelse(d4$especie == "Chinstrap", 1,0)
d4$Gentoo1 = ifelse(d4$especie == "Gentoo", 1,0)
```

No es necesario escalar estas variables porque ya están en el rango 0-1. Se calculan las distancias y se ordena:

```r
d4$dist = 0
for (ii in 1:10){
  xii = d4[ii,c(2,3,4,6,7,8)]
  d4$dist[ii] = knn_dist(xii,xp4)
}
orden4 = sort(d4$dist, index.return = T)
(d4s = d4[orden4$ix,])
```

```
##     peso long_pico1 prof_pico1 long_aleta1   especie Adelie1 Chinstrap1 Gentoo1
## 195 5300 0.58823529  0.3225806       0.775    Gentoo       0          0       1
## 178 5050 0.58235294  0.1129032       0.575    Gentoo       0          0       1
## 223 4750 0.58823529  0.0000000       0.775    Gentoo       0          0       1
## 228 5350 0.87058824  0.3387097       1.000    Gentoo       0          0       1
## 330 3400 0.48823529  0.6935484       0.450 Chinstrap       0          1       0
## 98  4250 0.15294118  1.0000000       0.150    Adelie       1          0       0
## 38  3100 0.04705882  0.7580645       0.050    Adelie       1          0       0
## 59  2850 0.07058824  0.5322581       0.000    Adelie       1          0       0
## 119 3050 0.00000000  0.3387097       0.050    Adelie       1          0       0
## 321 3450 1.00000000  0.8064516       0.325 Chinstrap       0          1       0
##          dist
## 195 0.4848706
## 178 0.6436641
## 223 0.7537148
## 228 0.7637048
## 330 1.4431381
## 98  1.5397421
## 38  1.5562349
## 59  1.5774714
## 119 1.5982639
## 321 1.6239740
```

- Si K = 1 la predicción para el peso es 5300 g.
- Si K = 2 la predicción es la media entre 5300 y 5050 = 5175 g.
- Si K = 3 la predicción es la media entre 5300, 5050 y 4750 = 5033.3333333 g.

5 Regresores cualitativos-2

En realidad no es necesario utilizar las tres variables auxiliares, con dos de ellas es suficiente. Por ejemplo, se va a utilizar Adelie1 y Chinstrap1:

- Los pinguinos Adelie se indican como (Adelie1=1, Chinstrap1=0);
- Los pinguinos Chinstrap como (Adelie1=0, Chinstrap1=1);
- Y los pinguinos Gentoo como (Adelie1=0, Chinstrap1=0);

Para comprobarlo se va a repetir el apartado anterior utilizando solo esas dos variables auxiliares:

```r
# se crea el dato a predecir:
(xp5 = c(xp3,0,0))
```

```
## [1] 0.2823529 0.6774194 0.6500000 0.0000000 0.0000000
```

```r
# Se crea la base de datos:
d5 = d4[,c("peso","long_pico1","prof_pico1","long_aleta1","Adelie1","Chinstrap1")]
```

```r
# Se calculan las distancias y se ordena:
d5$dist = 0
for (ii in 1:10){
  xii = d5[ii,c(2:6)]
  d5$dist[ii] = knn_dist(xii,xp5)
}
orden5 = sort(d5$dist, index.return = T)
(d5s = d5[orden5$ix,])
```

```
##     peso long_pico1 prof_pico1 long_aleta1 Adelie1 Chinstrap1      dist
## 195 5300 0.58823529  0.3225806       0.775       0          0 0.4848706
## 178 5050 0.58235294  0.1129032       0.575       0          0 0.6436641
## 223 4750 0.58823529  0.0000000       0.775       0          0 0.7537148
## 228 5350 0.87058824  0.3387097       1.000       0          0 0.7637048
## 330 3400 0.48823529  0.6935484       0.450       0          1 1.0405036
## 98  4250 0.15294118  1.0000000       0.150       1          0 1.1708141
## 38  3100 0.04705882  0.7580645       0.050       1          0 1.1924206
## 59  2850 0.07058824  0.5322581       0.000       1          0 1.2200066
## 119 3050 0.00000000  0.3387097       0.050       1          0 1.2467748
## 321 3450 1.00000000  0.8064516       0.325       0          1 1.2795670
```