

A Simple Approach to Ordinal Classification

Ciencia de datos 21/22

27/1/2022

Procederemos a ilustrar la funcionalidad del algoritmo con el ejemplo base Iris

```
summary(iris)
```

```
##      Sepal.Length      Sepal.Width      Petal.Length      Petal.Width
## Min.       :4.300    Min.       :2.000    Min.       :1.000    Min.       :0.100
## 1st Qu.:5.100    1st Qu.:2.800    1st Qu.:1.600    1st Qu.:0.300
## Median :5.800    Median :3.000    Median :4.350    Median :1.300
## Mean   :5.843    Mean   :3.057    Mean   :3.758    Mean   :1.199
## 3rd Qu.:6.400    3rd Qu.:3.300    3rd Qu.:5.100    3rd Qu.:1.800
## Max.    :7.900    Max.    :4.400    Max.    :6.900    Max.    :2.500
##      Species
## setosa      :50
## versicolor:50
## virginica   :50
##
##
##
```

Para mayor comodidad , se codificará la clase a enteros haciendo uso de la función revalue de la librería plyr

```
library(plyr)
iri<-iris
iri$Species <- revalue(iri$Species, c("setosa"="1", "versicolor"="2", "virginica"="3"))
iri
```

```
##      Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1           5.1         3.5         1.4         0.2         1
## 2           4.9         3.0         1.4         0.2         1
## 3           4.7         3.2         1.3         0.2         1
## 4           4.6         3.1         1.5         0.2         1
## 5           5.0         3.6         1.4         0.2         1
## 6           5.4         3.9         1.7         0.4         1
## 7           4.6         3.4         1.4         0.3         1
## 8           5.0         3.4         1.5         0.2         1
## 9           4.4         2.9         1.4         0.2         1
## 10          4.9         3.1         1.5         0.1         1
## 11          5.4         3.7         1.5         0.2         1
## 12          4.8         3.4         1.6         0.2         1
## 13          4.8         3.0         1.4         0.1         1
## 14          4.3         3.0         1.1         0.1         1
## 15          5.8         4.0         1.2         0.2         1
## 16          5.7         4.4         1.5         0.4         1
## 17          5.4         3.9         1.3         0.4         1
## 18          5.1         3.5         1.4         0.3         1
## 19          5.7         3.8         1.7         0.3         1
```

## 20	5.1	3.8	1.5	0.3	1
## 21	5.4	3.4	1.7	0.2	1
## 22	5.1	3.7	1.5	0.4	1
## 23	4.6	3.6	1.0	0.2	1
## 24	5.1	3.3	1.7	0.5	1
## 25	4.8	3.4	1.9	0.2	1
## 26	5.0	3.0	1.6	0.2	1
## 27	5.0	3.4	1.6	0.4	1
## 28	5.2	3.5	1.5	0.2	1
## 29	5.2	3.4	1.4	0.2	1
## 30	4.7	3.2	1.6	0.2	1
## 31	4.8	3.1	1.6	0.2	1
## 32	5.4	3.4	1.5	0.4	1
## 33	5.2	4.1	1.5	0.1	1
## 34	5.5	4.2	1.4	0.2	1
## 35	4.9	3.1	1.5	0.2	1
## 36	5.0	3.2	1.2	0.2	1
## 37	5.5	3.5	1.3	0.2	1
## 38	4.9	3.6	1.4	0.1	1
## 39	4.4	3.0	1.3	0.2	1
## 40	5.1	3.4	1.5	0.2	1
## 41	5.0	3.5	1.3	0.3	1
## 42	4.5	2.3	1.3	0.3	1
## 43	4.4	3.2	1.3	0.2	1
## 44	5.0	3.5	1.6	0.6	1
## 45	5.1	3.8	1.9	0.4	1
## 46	4.8	3.0	1.4	0.3	1
## 47	5.1	3.8	1.6	0.2	1
## 48	4.6	3.2	1.4	0.2	1
## 49	5.3	3.7	1.5	0.2	1
## 50	5.0	3.3	1.4	0.2	1
## 51	7.0	3.2	4.7	1.4	2
## 52	6.4	3.2	4.5	1.5	2
## 53	6.9	3.1	4.9	1.5	2
## 54	5.5	2.3	4.0	1.3	2
## 55	6.5	2.8	4.6	1.5	2
## 56	5.7	2.8	4.5	1.3	2
## 57	6.3	3.3	4.7	1.6	2
## 58	4.9	2.4	3.3	1.0	2
## 59	6.6	2.9	4.6	1.3	2
## 60	5.2	2.7	3.9	1.4	2
## 61	5.0	2.0	3.5	1.0	2
## 62	5.9	3.0	4.2	1.5	2
## 63	6.0	2.2	4.0	1.0	2
## 64	6.1	2.9	4.7	1.4	2
## 65	5.6	2.9	3.6	1.3	2
## 66	6.7	3.1	4.4	1.4	2
## 67	5.6	3.0	4.5	1.5	2
## 68	5.8	2.7	4.1	1.0	2
## 69	6.2	2.2	4.5	1.5	2
## 70	5.6	2.5	3.9	1.1	2
## 71	5.9	3.2	4.8	1.8	2
## 72	6.1	2.8	4.0	1.3	2
## 73	6.3	2.5	4.9	1.5	2

## 74	6.1	2.8	4.7	1.2	2
## 75	6.4	2.9	4.3	1.3	2
## 76	6.6	3.0	4.4	1.4	2
## 77	6.8	2.8	4.8	1.4	2
## 78	6.7	3.0	5.0	1.7	2
## 79	6.0	2.9	4.5	1.5	2
## 80	5.7	2.6	3.5	1.0	2
## 81	5.5	2.4	3.8	1.1	2
## 82	5.5	2.4	3.7	1.0	2
## 83	5.8	2.7	3.9	1.2	2
## 84	6.0	2.7	5.1	1.6	2
## 85	5.4	3.0	4.5	1.5	2
## 86	6.0	3.4	4.5	1.6	2
## 87	6.7	3.1	4.7	1.5	2
## 88	6.3	2.3	4.4	1.3	2
## 89	5.6	3.0	4.1	1.3	2
## 90	5.5	2.5	4.0	1.3	2
## 91	5.5	2.6	4.4	1.2	2
## 92	6.1	3.0	4.6	1.4	2
## 93	5.8	2.6	4.0	1.2	2
## 94	5.0	2.3	3.3	1.0	2
## 95	5.6	2.7	4.2	1.3	2
## 96	5.7	3.0	4.2	1.2	2
## 97	5.7	2.9	4.2	1.3	2
## 98	6.2	2.9	4.3	1.3	2
## 99	5.1	2.5	3.0	1.1	2
## 100	5.7	2.8	4.1	1.3	2
## 101	6.3	3.3	6.0	2.5	3
## 102	5.8	2.7	5.1	1.9	3
## 103	7.1	3.0	5.9	2.1	3
## 104	6.3	2.9	5.6	1.8	3
## 105	6.5	3.0	5.8	2.2	3
## 106	7.6	3.0	6.6	2.1	3
## 107	4.9	2.5	4.5	1.7	3
## 108	7.3	2.9	6.3	1.8	3
## 109	6.7	2.5	5.8	1.8	3
## 110	7.2	3.6	6.1	2.5	3
## 111	6.5	3.2	5.1	2.0	3
## 112	6.4	2.7	5.3	1.9	3
## 113	6.8	3.0	5.5	2.1	3
## 114	5.7	2.5	5.0	2.0	3
## 115	5.8	2.8	5.1	2.4	3
## 116	6.4	3.2	5.3	2.3	3
## 117	6.5	3.0	5.5	1.8	3
## 118	7.7	3.8	6.7	2.2	3
## 119	7.7	2.6	6.9	2.3	3
## 120	6.0	2.2	5.0	1.5	3
## 121	6.9	3.2	5.7	2.3	3
## 122	5.6	2.8	4.9	2.0	3
## 123	7.7	2.8	6.7	2.0	3
## 124	6.3	2.7	4.9	1.8	3
## 125	6.7	3.3	5.7	2.1	3
## 126	7.2	3.2	6.0	1.8	3
## 127	6.2	2.8	4.8	1.8	3

## 128	6.1	3.0	4.9	1.8	3
## 129	6.4	2.8	5.6	2.1	3
## 130	7.2	3.0	5.8	1.6	3
## 131	7.4	2.8	6.1	1.9	3
## 132	7.9	3.8	6.4	2.0	3
## 133	6.4	2.8	5.6	2.2	3
## 134	6.3	2.8	5.1	1.5	3
## 135	6.1	2.6	5.6	1.4	3
## 136	7.7	3.0	6.1	2.3	3
## 137	6.3	3.4	5.6	2.4	3
## 138	6.4	3.1	5.5	1.8	3
## 139	6.0	3.0	4.8	1.8	3
## 140	6.9	3.1	5.4	2.1	3
## 141	6.7	3.1	5.6	2.4	3
## 142	6.9	3.1	5.1	2.3	3
## 143	5.8	2.7	5.1	1.9	3
## 144	6.8	3.2	5.9	2.3	3
## 145	6.7	3.3	5.7	2.5	3
## 146	6.7	3.0	5.2	2.3	3
## 147	6.3	2.5	5.0	1.9	3
## 148	6.5	3.0	5.2	2.0	3
## 149	6.2	3.4	5.4	2.3	3
## 150	5.9	3.0	5.1	1.8	3

El siguiente paso consistirá en la creación de n° clases -1 (en este caso $3-1 = 2$) dataframes con clase binaria descomponiendo el problema inicial. Considerando el orden de aparición de las clases

```
clases=as.integer(unique(iri$Species))
clases
```

```
## [1] 1 2 3
```

Seleccionaremos los índices de éstas:

```
indices<-which(iri$Species==clases[1])
indices
```

```
## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
## [24] 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46
## [47] 47 48 49 50
```

Guardamos la variable clase en un vector auxiliar

```
y = as.integer(iri$Species)
```

Cambiamos los valores de estas clases a 0 y el resto a 1

```
y[indices]<-0
y = ifelse(y==0,0,1)
```

Con esto ya tenemos casi listo el primer data frame derivado, nos queda por juntar el resto del dataset con la nueva clase binaria. Más adelante se procederá a clasificar dicho conjunto de datos , por lo que es conveniente pasar la variable a factor

```
data1 = cbind(iri[,1:4],target1=as.factor(y))
sapply(data1,class)
```

```
## Sepal.Length Sepal.Width Petal.Length Petal.Width target1
## "numeric" "numeric" "numeric" "numeric" "factor"
```

```
data1
```

```
## Sepal.Length Sepal.Width Petal.Length Petal.Width target1
## 1 5.1 3.5 1.4 0.2 0
## 2 4.9 3.0 1.4 0.2 0
## 3 4.7 3.2 1.3 0.2 0
## 4 4.6 3.1 1.5 0.2 0
## 5 5.0 3.6 1.4 0.2 0
## 6 5.4 3.9 1.7 0.4 0
## 7 4.6 3.4 1.4 0.3 0
## 8 5.0 3.4 1.5 0.2 0
## 9 4.4 2.9 1.4 0.2 0
## 10 4.9 3.1 1.5 0.1 0
## 11 5.4 3.7 1.5 0.2 0
## 12 4.8 3.4 1.6 0.2 0
## 13 4.8 3.0 1.4 0.1 0
## 14 4.3 3.0 1.1 0.1 0
## 15 5.8 4.0 1.2 0.2 0
## 16 5.7 4.4 1.5 0.4 0
## 17 5.4 3.9 1.3 0.4 0
## 18 5.1 3.5 1.4 0.3 0
## 19 5.7 3.8 1.7 0.3 0
## 20 5.1 3.8 1.5 0.3 0
## 21 5.4 3.4 1.7 0.2 0
## 22 5.1 3.7 1.5 0.4 0
## 23 4.6 3.6 1.0 0.2 0
## 24 5.1 3.3 1.7 0.5 0
## 25 4.8 3.4 1.9 0.2 0
## 26 5.0 3.0 1.6 0.2 0
## 27 5.0 3.4 1.6 0.4 0
## 28 5.2 3.5 1.5 0.2 0
## 29 5.2 3.4 1.4 0.2 0
## 30 4.7 3.2 1.6 0.2 0
## 31 4.8 3.1 1.6 0.2 0
## 32 5.4 3.4 1.5 0.4 0
## 33 5.2 4.1 1.5 0.1 0
## 34 5.5 4.2 1.4 0.2 0
## 35 4.9 3.1 1.5 0.2 0
## 36 5.0 3.2 1.2 0.2 0
## 37 5.5 3.5 1.3 0.2 0
## 38 4.9 3.6 1.4 0.1 0
## 39 4.4 3.0 1.3 0.2 0
## 40 5.1 3.4 1.5 0.2 0
## 41 5.0 3.5 1.3 0.3 0
## 42 4.5 2.3 1.3 0.3 0
## 43 4.4 3.2 1.3 0.2 0
## 44 5.0 3.5 1.6 0.6 0
```

## 45	5.1	3.8	1.9	0.4	0
## 46	4.8	3.0	1.4	0.3	0
## 47	5.1	3.8	1.6	0.2	0
## 48	4.6	3.2	1.4	0.2	0
## 49	5.3	3.7	1.5	0.2	0
## 50	5.0	3.3	1.4	0.2	0
## 51	7.0	3.2	4.7	1.4	1
## 52	6.4	3.2	4.5	1.5	1
## 53	6.9	3.1	4.9	1.5	1
## 54	5.5	2.3	4.0	1.3	1
## 55	6.5	2.8	4.6	1.5	1
## 56	5.7	2.8	4.5	1.3	1
## 57	6.3	3.3	4.7	1.6	1
## 58	4.9	2.4	3.3	1.0	1
## 59	6.6	2.9	4.6	1.3	1
## 60	5.2	2.7	3.9	1.4	1
## 61	5.0	2.0	3.5	1.0	1
## 62	5.9	3.0	4.2	1.5	1
## 63	6.0	2.2	4.0	1.0	1
## 64	6.1	2.9	4.7	1.4	1
## 65	5.6	2.9	3.6	1.3	1
## 66	6.7	3.1	4.4	1.4	1
## 67	5.6	3.0	4.5	1.5	1
## 68	5.8	2.7	4.1	1.0	1
## 69	6.2	2.2	4.5	1.5	1
## 70	5.6	2.5	3.9	1.1	1
## 71	5.9	3.2	4.8	1.8	1
## 72	6.1	2.8	4.0	1.3	1
## 73	6.3	2.5	4.9	1.5	1
## 74	6.1	2.8	4.7	1.2	1
## 75	6.4	2.9	4.3	1.3	1
## 76	6.6	3.0	4.4	1.4	1
## 77	6.8	2.8	4.8	1.4	1
## 78	6.7	3.0	5.0	1.7	1
## 79	6.0	2.9	4.5	1.5	1
## 80	5.7	2.6	3.5	1.0	1
## 81	5.5	2.4	3.8	1.1	1
## 82	5.5	2.4	3.7	1.0	1
## 83	5.8	2.7	3.9	1.2	1
## 84	6.0	2.7	5.1	1.6	1
## 85	5.4	3.0	4.5	1.5	1
## 86	6.0	3.4	4.5	1.6	1
## 87	6.7	3.1	4.7	1.5	1
## 88	6.3	2.3	4.4	1.3	1
## 89	5.6	3.0	4.1	1.3	1
## 90	5.5	2.5	4.0	1.3	1
## 91	5.5	2.6	4.4	1.2	1
## 92	6.1	3.0	4.6	1.4	1
## 93	5.8	2.6	4.0	1.2	1
## 94	5.0	2.3	3.3	1.0	1
## 95	5.6	2.7	4.2	1.3	1
## 96	5.7	3.0	4.2	1.2	1
## 97	5.7	2.9	4.2	1.3	1
## 98	6.2	2.9	4.3	1.3	1

## 99	5.1	2.5	3.0	1.1	1
## 100	5.7	2.8	4.1	1.3	1
## 101	6.3	3.3	6.0	2.5	1
## 102	5.8	2.7	5.1	1.9	1
## 103	7.1	3.0	5.9	2.1	1
## 104	6.3	2.9	5.6	1.8	1
## 105	6.5	3.0	5.8	2.2	1
## 106	7.6	3.0	6.6	2.1	1
## 107	4.9	2.5	4.5	1.7	1
## 108	7.3	2.9	6.3	1.8	1
## 109	6.7	2.5	5.8	1.8	1
## 110	7.2	3.6	6.1	2.5	1
## 111	6.5	3.2	5.1	2.0	1
## 112	6.4	2.7	5.3	1.9	1
## 113	6.8	3.0	5.5	2.1	1
## 114	5.7	2.5	5.0	2.0	1
## 115	5.8	2.8	5.1	2.4	1
## 116	6.4	3.2	5.3	2.3	1
## 117	6.5	3.0	5.5	1.8	1
## 118	7.7	3.8	6.7	2.2	1
## 119	7.7	2.6	6.9	2.3	1
## 120	6.0	2.2	5.0	1.5	1
## 121	6.9	3.2	5.7	2.3	1
## 122	5.6	2.8	4.9	2.0	1
## 123	7.7	2.8	6.7	2.0	1
## 124	6.3	2.7	4.9	1.8	1
## 125	6.7	3.3	5.7	2.1	1
## 126	7.2	3.2	6.0	1.8	1
## 127	6.2	2.8	4.8	1.8	1
## 128	6.1	3.0	4.9	1.8	1
## 129	6.4	2.8	5.6	2.1	1
## 130	7.2	3.0	5.8	1.6	1
## 131	7.4	2.8	6.1	1.9	1
## 132	7.9	3.8	6.4	2.0	1
## 133	6.4	2.8	5.6	2.2	1
## 134	6.3	2.8	5.1	1.5	1
## 135	6.1	2.6	5.6	1.4	1
## 136	7.7	3.0	6.1	2.3	1
## 137	6.3	3.4	5.6	2.4	1
## 138	6.4	3.1	5.5	1.8	1
## 139	6.0	3.0	4.8	1.8	1
## 140	6.9	3.1	5.4	2.1	1
## 141	6.7	3.1	5.6	2.4	1
## 142	6.9	3.1	5.1	2.3	1
## 143	5.8	2.7	5.1	1.9	1
## 144	6.8	3.2	5.9	2.3	1
## 145	6.7	3.3	5.7	2.5	1
## 146	6.7	3.0	5.2	2.3	1
## 147	6.3	2.5	5.0	1.9	1
## 148	6.5	3.0	5.2	2.0	1
## 149	6.2	3.4	5.4	2.3	1
## 150	5.9	3.0	5.1	1.8	1

Procederemos a repetir los pasos anteriores, teniendo en cuenta las clases ya convertidas en el paso anterior,

por lo que las añadiremos al vector de índices previamente creado.

```
indices<-c(indices,which(iri$Species==clases[2]) )
indices
```

```
##   [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17
##  [18] 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34
##  [35] 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51
##  [52] 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68
##  [69] 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85
##  [86] 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
```

Volvemos a crear de nuevo la variable auxiliar

```
y = as.integer(iri$Species)
```

Cambiamos los valores de estas clases a 0 y el resto a 1 considerando las anteriores

```
y[indices]=0
y = ifelse(y==0,0,1)
```

Creamos así el segundo dataset derivado

```
data2 = cbind(iri[,1:4],target2=as.factor(y))
sapply(data2,class)
```

```
## Sepal.Length Sepal.Width Petal.Length Petal.Width target2
##      "numeric"      "numeric"      "numeric"      "numeric"      "factor"
```

```
data2
```

```
##      Sepal.Length Sepal.Width Petal.Length Petal.Width target2
## 1           5.1           3.5           1.4           0.2         0
## 2           4.9           3.0           1.4           0.2         0
## 3           4.7           3.2           1.3           0.2         0
## 4           4.6           3.1           1.5           0.2         0
## 5           5.0           3.6           1.4           0.2         0
## 6           5.4           3.9           1.7           0.4         0
## 7           4.6           3.4           1.4           0.3         0
## 8           5.0           3.4           1.5           0.2         0
## 9           4.4           2.9           1.4           0.2         0
## 10          4.9           3.1           1.5           0.1         0
## 11          5.4           3.7           1.5           0.2         0
## 12          4.8           3.4           1.6           0.2         0
## 13          4.8           3.0           1.4           0.1         0
## 14          4.3           3.0           1.1           0.1         0
## 15          5.8           4.0           1.2           0.2         0
## 16          5.7           4.4           1.5           0.4         0
## 17          5.4           3.9           1.3           0.4         0
## 18          5.1           3.5           1.4           0.3         0
## 19          5.7           3.8           1.7           0.3         0
```


## 20	5.1	3.8	1.5	0.3	0
## 21	5.4	3.4	1.7	0.2	0
## 22	5.1	3.7	1.5	0.4	0
## 23	4.6	3.6	1.0	0.2	0
## 24	5.1	3.3	1.7	0.5	0
## 25	4.8	3.4	1.9	0.2	0
## 26	5.0	3.0	1.6	0.2	0
## 27	5.0	3.4	1.6	0.4	0
## 28	5.2	3.5	1.5	0.2	0
## 29	5.2	3.4	1.4	0.2	0
## 30	4.7	3.2	1.6	0.2	0
## 31	4.8	3.1	1.6	0.2	0
## 32	5.4	3.4	1.5	0.4	0
## 33	5.2	4.1	1.5	0.1	0
## 34	5.5	4.2	1.4	0.2	0
## 35	4.9	3.1	1.5	0.2	0
## 36	5.0	3.2	1.2	0.2	0
## 37	5.5	3.5	1.3	0.2	0
## 38	4.9	3.6	1.4	0.1	0
## 39	4.4	3.0	1.3	0.2	0
## 40	5.1	3.4	1.5	0.2	0
## 41	5.0	3.5	1.3	0.3	0
## 42	4.5	2.3	1.3	0.3	0
## 43	4.4	3.2	1.3	0.2	0
## 44	5.0	3.5	1.6	0.6	0
## 45	5.1	3.8	1.9	0.4	0
## 46	4.8	3.0	1.4	0.3	0
## 47	5.1	3.8	1.6	0.2	0
## 48	4.6	3.2	1.4	0.2	0
## 49	5.3	3.7	1.5	0.2	0
## 50	5.0	3.3	1.4	0.2	0
## 51	7.0	3.2	4.7	1.4	0
## 52	6.4	3.2	4.5	1.5	0
## 53	6.9	3.1	4.9	1.5	0
## 54	5.5	2.3	4.0	1.3	0
## 55	6.5	2.8	4.6	1.5	0
## 56	5.7	2.8	4.5	1.3	0
## 57	6.3	3.3	4.7	1.6	0
## 58	4.9	2.4	3.3	1.0	0
## 59	6.6	2.9	4.6	1.3	0
## 60	5.2	2.7	3.9	1.4	0
## 61	5.0	2.0	3.5	1.0	0
## 62	5.9	3.0	4.2	1.5	0
## 63	6.0	2.2	4.0	1.0	0
## 64	6.1	2.9	4.7	1.4	0
## 65	5.6	2.9	3.6	1.3	0
## 66	6.7	3.1	4.4	1.4	0
## 67	5.6	3.0	4.5	1.5	0
## 68	5.8	2.7	4.1	1.0	0
## 69	6.2	2.2	4.5	1.5	0
## 70	5.6	2.5	3.9	1.1	0
## 71	5.9	3.2	4.8	1.8	0
## 72	6.1	2.8	4.0	1.3	0
## 73	6.3	2.5	4.9	1.5	0

## 74	6.1	2.8	4.7	1.2	0
## 75	6.4	2.9	4.3	1.3	0
## 76	6.6	3.0	4.4	1.4	0
## 77	6.8	2.8	4.8	1.4	0
## 78	6.7	3.0	5.0	1.7	0
## 79	6.0	2.9	4.5	1.5	0
## 80	5.7	2.6	3.5	1.0	0
## 81	5.5	2.4	3.8	1.1	0
## 82	5.5	2.4	3.7	1.0	0
## 83	5.8	2.7	3.9	1.2	0
## 84	6.0	2.7	5.1	1.6	0
## 85	5.4	3.0	4.5	1.5	0
## 86	6.0	3.4	4.5	1.6	0
## 87	6.7	3.1	4.7	1.5	0
## 88	6.3	2.3	4.4	1.3	0
## 89	5.6	3.0	4.1	1.3	0
## 90	5.5	2.5	4.0	1.3	0
## 91	5.5	2.6	4.4	1.2	0
## 92	6.1	3.0	4.6	1.4	0
## 93	5.8	2.6	4.0	1.2	0
## 94	5.0	2.3	3.3	1.0	0
## 95	5.6	2.7	4.2	1.3	0
## 96	5.7	3.0	4.2	1.2	0
## 97	5.7	2.9	4.2	1.3	0
## 98	6.2	2.9	4.3	1.3	0
## 99	5.1	2.5	3.0	1.1	0
## 100	5.7	2.8	4.1	1.3	0
## 101	6.3	3.3	6.0	2.5	1
## 102	5.8	2.7	5.1	1.9	1
## 103	7.1	3.0	5.9	2.1	1
## 104	6.3	2.9	5.6	1.8	1
## 105	6.5	3.0	5.8	2.2	1
## 106	7.6	3.0	6.6	2.1	1
## 107	4.9	2.5	4.5	1.7	1
## 108	7.3	2.9	6.3	1.8	1
## 109	6.7	2.5	5.8	1.8	1
## 110	7.2	3.6	6.1	2.5	1
## 111	6.5	3.2	5.1	2.0	1
## 112	6.4	2.7	5.3	1.9	1
## 113	6.8	3.0	5.5	2.1	1
## 114	5.7	2.5	5.0	2.0	1
## 115	5.8	2.8	5.1	2.4	1
## 116	6.4	3.2	5.3	2.3	1
## 117	6.5	3.0	5.5	1.8	1
## 118	7.7	3.8	6.7	2.2	1
## 119	7.7	2.6	6.9	2.3	1
## 120	6.0	2.2	5.0	1.5	1
## 121	6.9	3.2	5.7	2.3	1
## 122	5.6	2.8	4.9	2.0	1
## 123	7.7	2.8	6.7	2.0	1
## 124	6.3	2.7	4.9	1.8	1
## 125	6.7	3.3	5.7	2.1	1
## 126	7.2	3.2	6.0	1.8	1
## 127	6.2	2.8	4.8	1.8	1

## 128	6.1	3.0	4.9	1.8	1
## 129	6.4	2.8	5.6	2.1	1
## 130	7.2	3.0	5.8	1.6	1
## 131	7.4	2.8	6.1	1.9	1
## 132	7.9	3.8	6.4	2.0	1
## 133	6.4	2.8	5.6	2.2	1
## 134	6.3	2.8	5.1	1.5	1
## 135	6.1	2.6	5.6	1.4	1
## 136	7.7	3.0	6.1	2.3	1
## 137	6.3	3.4	5.6	2.4	1
## 138	6.4	3.1	5.5	1.8	1
## 139	6.0	3.0	4.8	1.8	1
## 140	6.9	3.1	5.4	2.1	1
## 141	6.7	3.1	5.6	2.4	1
## 142	6.9	3.1	5.1	2.3	1
## 143	5.8	2.7	5.1	1.9	1
## 144	6.8	3.2	5.9	2.3	1
## 145	6.7	3.3	5.7	2.5	1
## 146	6.7	3.0	5.2	2.3	1
## 147	6.3	2.5	5.0	1.9	1
## 148	6.5	3.0	5.2	2.0	1
## 149	6.2	3.4	5.4	2.3	1
## 150	5.9	3.0	5.1	1.8	1

El siguiente paso sería crear un modelo de clasificación para conjunto de datos. Usaremos el c4.5 , implementado en el paquete de Rweka como j48

```
library(Rweka)
m1 <- J48(target1 ~ ., data = data1)
m1
```

```
## J48 pruned tree
## -----
##
## Petal.Width <= 0.6: 0 (50.0)
## Petal.Width > 0.6: 1 (100.0)
##
## Number of Leaves : 2
##
## Size of the tree : 3
```

```
m2 <- J48(target2 ~ ., data = data2)
m2
```

```
## J48 pruned tree
## -----
##
## Petal.Width <= 1.7
## |   Petal.Length <= 4.9: 0 (98.0/1.0)
## |   Petal.Length > 4.9
## | |   Petal.Width <= 1.5: 1 (3.0)
## | |   Petal.Width > 1.5: 0 (3.0/1.0)
## Petal.Width > 1.7: 1 (46.0/1.0)
```

```
##
## Number of Leaves : 4
##
## Size of the tree : 7
```

Podemos hacer un estudio más detallado de los modelos , haciendo uso de la siguiente función

```
eval_m1 <- evaluate_Weka_classifier(m1, numFolds = 10, complexity = FALSE, class = TRUE)
eval_m1
```

```
## === 10 Fold Cross Validation ===
```

```
##
```

```
## === Summary ===
```

```
##
```

```
## Correctly Classified Instances      149          99.3333 %
## Incorrectly Classified Instances    1           0.6667 %
## Kappa statistic                     0.9849
## Mean absolute error                 0.0067
## Root mean squared error             0.0816
## Relative absolute error             1.4973 %
## Root relative squared error         17.3203 %
## Coverage of cases (0.95 level)     99.3333 %
## Mean rel. region size (0.95 level)  50          %
## Total Number of Instances          150
```

```
##
```

```
## === Detailed Accuracy By Class ===
```

```
##
```

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,980	0,000	1,000	0,980	0,990	0,985	0,990	0,987	0
	1,000	0,020	0,990	1,000	0,995	0,985	0,990	0,990	1
## Weighted Avg.	0,993	0,013	0,993	0,993	0,993	0,985	0,990	0,989	

```
##
```

```
## === Confusion Matrix ===
```

```
##
```

```
##      a      b      <-- classified as
```

```
##      49      1 |      a = 0
```

```
##      0 100 |      b = 1
```

```
eval_m2 <- evaluate_Weka_classifier(m2, numFolds = 10, complexity = FALSE, class = TRUE)
eval_m2
```

```
## === 10 Fold Cross Validation ===
```

```
##
```

```
## === Summary ===
```

```
##
```

```
## Correctly Classified Instances      141          94          %
## Incorrectly Classified Instances    9           6          %
## Kappa statistic                     0.8643
## Mean absolute error                 0.0744
## Root mean squared error             0.2422
## Relative absolute error             16.7021 %
## Root relative squared error         51.378  %
## Coverage of cases (0.95 level)     94.6667 %
```

```
## Mean rel. region size (0.95 level)      52      %
## Total Number of Instances              150
##
## === Detailed Accuracy By Class ===
##
##           TP Rate  FP Rate  Precision  Recall  F-Measure  MCC      ROC Area  PRC Area  Class
##           0,960   0,100   0,950     0,960   0,955     0,864   0,925   0,947     0
##           0,900   0,040   0,918     0,900   0,909     0,864   0,925   0,827     1
## Weighted Avg.   0,940   0,080   0,940     0,940   0,940     0,864   0,925   0,907
##
## === Confusion Matrix ===
##
##   a  b   <-- classified as
##  96  4 |   a = 0
##   5 45 |   b = 1
```

Necesitamos conocer las probabilidades generadas por nuestros modelos, para ello probaremos a predecir la instancia numero 130 de nuestro dataset, sabiendo de por si que pertenece a la clase 3

```
pred1<-predict(m1,iri[130,1:4],type="probability")
pred1
```

```
##      0 1
## 130 0 1
```

```
pred2<-predict(m2,iri[130,1:4],type="probability")
pred2
```

```
##           0           1
## 130 0.6666667 0.3333333
```