Árboles de clasificación/decisión mediante R

Paquete

Hay dos paquetes de R que cubren este asunto: `tree` y `rpart`. Trabajaremos con el segundo porque

1. Está incluido en la distribución básica de R.
2. La complejidad algorítmica no es mucho mayor que la del primero.
3. El autor de `tree` recomienda `rpart`.

El artículo de referencia del paquete se puede encontrar en
`http://cran.r-project.org/web/packages/rpart/vignettes/longintro.pdf`

Datos

Como ejemplo usaremos el conjunto de datos `kyphosis`, incluido en `rpart`, recogidos a partir de niños operados de la columna vertebral (un análisis detallado se encuentra en la sección 6.3 del artículo de referencia). Contiene las siguientes variables:

- `Kyphosis`: factor con niveles `absent` y `present` indicando si tras la operación aparecía cifosis (un tipo de deformación de la columna vertebral).
- `Age`: edad en meses.
- `Number`: número de vértebras involucradas.
- `Start`: número de la primera vértebra (la más alta) operada.

Poda

Sea T un árbol. Sea |T| el número de nodos terminales de T. Sea R(T) el riesgo del árbol (en problemas de clasificación, la probabilidad de clasificación incorrecta). Para podar el árbol se define su costo como $R_\alpha(T) = R(T) + \alpha\,|T|$. Mediante validación cruzada se puede encontrar el valor óptimo de α. En las tablas, la idea del coeficiente α se presenta a través del costo por complejidad, cp, de modo que $R_{cp}(T) = R(T) + cp\,|T|\,R(T_1)$, donde T_1 es el árbol sin divisiones.

Análisis

En verde están las órdenes introducidas por nosotros. En negro, la salida del ordenador. En azul, comentarios que hemos incrustado en este código. Se llama nodo a cada uno de los subconjuntos de la muestra original.

```
library(rpart)
```

```
árbol <- rpart (Kyphosis ~ . , kyphosis)
árbol
n= 81

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 81 17 absent (0.7901235 0.2098765)
   2) Start>=8.5 62  6 absent (0.9032258 0.0967742)
     4) Start>=14.5 29  0 absent (1.0000000 0.0000000) *
     5) Start< 14.5 33  6 absent (0.8181818 0.1818182)
      10) Age< 55 12  0 absent (1.0000000 0.0000000) *
      11) Age>=55 21  6 absent (0.7142857 0.2857143)
        22) Age>=111 14  2 absent (0.8571429 0.1428571) *
        23) Age< 111 7  3 present (0.4285714 0.5714286) *
   3) Start< 8.5 19  8 present (0.4210526 0.5789474) *

# En cada renglón de la salida anterior tenemos:
# node  – número identificador del nodo
#         (los hijos del nodo x son 2x y 2x+1, por comodidad)
# split – regla que determina la inclusión de un individuo en el nodo
# n     – número de individuos en ese nodo
# loss  – número de individuos mal clasificados en ese nodo
# yval  – categoría donde se clasificarían los individuos del nodo
# yprob – frecuencias relativas de las categorías en ese nodo
#   Así, por ejemplo, en el nodo raíz hay 81 observaciones; como
# la mayoría son absent, el yval vale absent; loss=17 porque hay
# 17 present en la muestra original.

ls.str(árbol) # estructura del árbol

summary (árbol)

Call:
rpart(formula = Kyphosis ~ ., data = kyphosis)
  n= 81

          CP nsplit rel error    xerror      xstd
1 0.17647059      0 1.0000000 1.0000000 0.2155872
2 0.01960784      1 0.8235294 0.9411765 0.2107780
3 0.01000000      4 0.7647059 0.9411765 0.2107780

# La tabla anterior indica
# que valores de CP > 0'17... no generan divisiones;
# que CP > 0'0196 genera 1 división (dos nodos terminales);
# que CP > 0'01 (límite por omisión) genera 5 nodos terminales;
# Nótese que en la tabla las columnas “rel error”, “xerror” y “xstd”
# están rescaladas de modo que el primer “rel error” vale 1.  Para
# hallar las frecuencias absolutas correspondientes habría que
# multiplicar las tres columnas por 17.  (La “x” se refiere a la
# validación cruzada.)

Node number 1: 81 observations,    complexity param=0.1764706
  predicted class=absent   expected loss=0.2098765
    class counts:    64    17
   probabilities: 0.790 0.210
```

```
  left son=2 (62 obs) right son=3 (19 obs)
  Primary splits:
      Start  < 8.5  to the right, improve=6.762330, (0 missing)
# La mejora es n por el índice de impureza, calculado mediante la
# aplicación de la función de información (-p.log(p)) o la de Gini
# (2.p.(1-p)) a las frecuencias de los nodos:
# Δi(t) = i(t) − pR i(tR) − pL i(tL)

# por omisión, se usa la de Gini; por ejemplo, la mejora de 6'762330
# de arriba viene de multiplicar 81 por Δi(t):
# 81 . (gini(17/81) - (19/81).gini(8/19) – (62/81).gini(56/62))
      Number < 5.5  to the left,  improve=2.866795, (0 missing)
      Age    < 39.5 to the left,  improve=2.250212, (0 missing)
  Surrogate splits:
      Number < 6.5  to the left,  agree=0.802, adj=0.158, (0 split)


Node number 2: 62 observations,    complexity param=0.01960784
  predicted class=absent   expected loss=0.09677419
    class counts:    56     6
   probabilities: 0.903 0.097
  left son=4 (29 obs) right son=5 (33 obs)
  Primary splits:
      Start  < 14.5 to the right, improve=1.0205280, (0 missing)
      Age    < 55   to the left,  improve=0.6848635, (0 missing)
      Number < 4.5  to the left,  improve=0.2975332, (0 missing)
  Surrogate splits:
      Number < 3.5  to the left,  agree=0.645, adj=0.241, (0 split)
      Age    < 16   to the left,  agree=0.597, adj=0.138, (0 split)

Node number 3: 19 observations
  predicted class=present  expected loss=0.4210526
    class counts:     8    11
   probabilities: 0.421 0.579

Node number 4: 29 observations
  predicted class=absent   expected loss=0
    class counts:    29     0
   probabilities: 1.000 0.000

Node number 5: 33 observations,    complexity param=0.01960784
  predicted class=absent   expected loss=0.1818182
    class counts:    27     6
   probabilities: 0.818 0.182
  left son=10 (12 obs) right son=11 (21 obs)
  Primary splits:
      Age    < 55   to the left,  improve=1.2467530, (0 missing)
      Start  < 12.5 to the right, improve=0.2887701, (0 missing)
      Number < 3.5  to the right, improve=0.1753247, (0 missing)
  Surrogate splits:
      Start  < 9.5  to the left,  agree=0.758, adj=0.333, (0 split)
      Number < 5.5  to the right, agree=0.697, adj=0.167, (0 split)

Node number 10: 12 observations
  predicted class=absent   expected loss=0
    class counts:    12     0
```

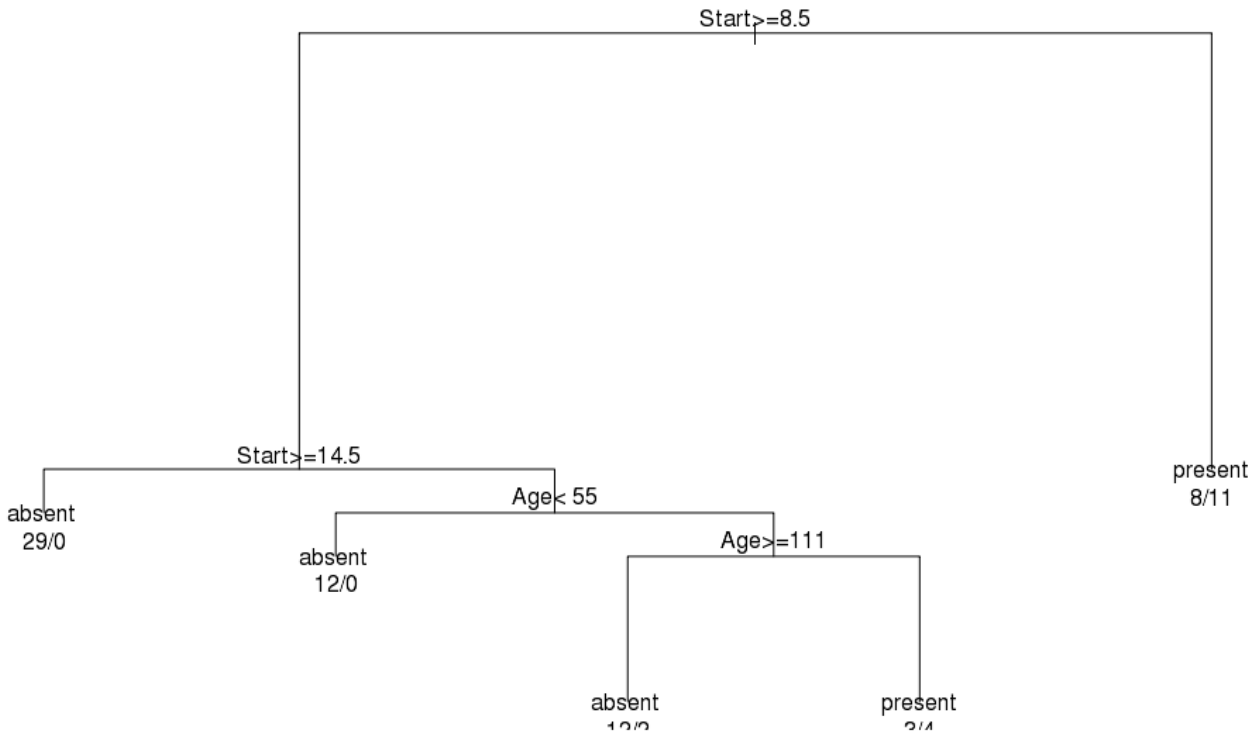
```
   probabilities: 1.000 0.000

Node number 11: 21 observations,    complexity param=0.01960784
  predicted class=absent   expected loss=0.2857143
    class counts:    15     6
   probabilities: 0.714 0.286
  left son=22 (14 obs) right son=23 (7 obs)
  Primary splits:
      Age    < 111  to the right, improve=1.71428600, (0 missing)
      Start  < 12.5 to the right, improve=0.79365080, (0 missing)
      Number < 3.5  to the right, improve=0.07142857, (0 missing)

Node number 22: 14 observations
  predicted class=absent   expected loss=0.1428571
    class counts:    12     2
   probabilities: 0.857 0.143

Node number 23: 7 observations
  predicted class=present  expected loss=0.4285714
    class counts:     3     4
   probabilities: 0.429 0.571
```

```
plot (árbol)             # representa las ramas del árbol
text (árbol, use.n=TRUE) # añade etiquetas al árbol
```



Cada nodo del árbol está etiquetado con una condición. Si se cumple, se sigue por la rama de la izquierda; si no, por la derecha. Al haber usado `use.n=TRUE` se muestran en los nodos terminales las frecuencias absolutas de las categorías `absent/present`, separadas por una barra.

Podemos recurrir a otra representación posible en este caso, ya que la clasificación se realiza mediante sólo dos variables (Start y Age), utilizando la nube de puntos correspondiente:

```
# Gráfico aclaratorio de la clasificación
# dispersión de Start frente a Age
plot (Start ~ Age, kyphosis, col=kyphosis$Kyphosis)
# la opción col=kyphosis$Kyphosis asigna colores a las categorías
abline (h =  8.5) # dibuja la línea horizontal Start=8.5
abline (h = 14.5) # dibuja la línea horizontal Start=14.5
lines (c(55,  55 ), c(8.5, 14.5)) # recta de (55;8.5) a (55;14.5)
lines (c(111, 111), c(8.5, 14.5))
```