# COMP 6838 Data Mining

LECTURE 13:
Decision Trees

Dr. Edgar Acuna
Department of Mathematics

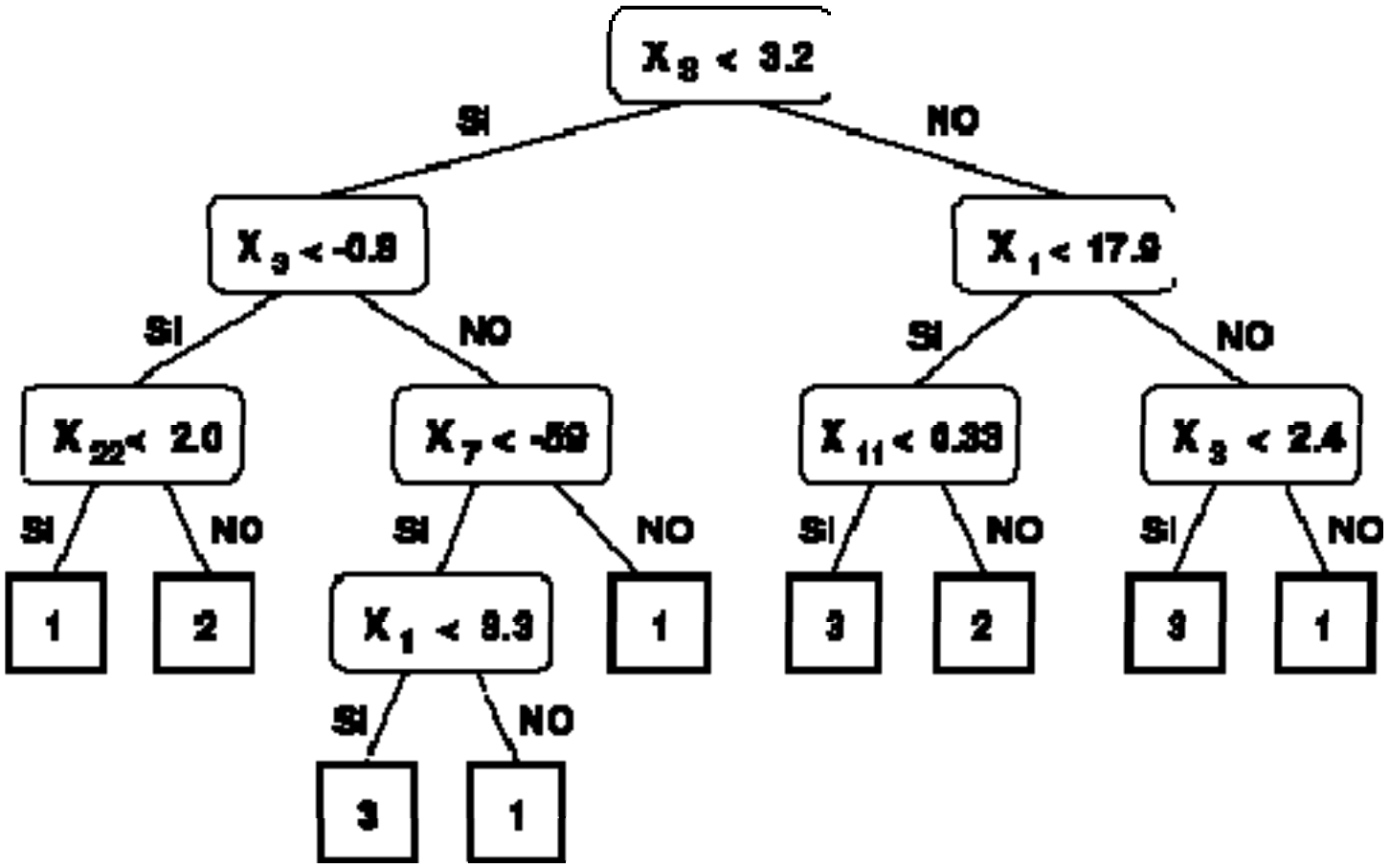
University of Puerto Rico- Mayaguez

math.uprm.edu/~edgar

The use of decision trees began in the social sciences with the work of Sonquist and Morgan (1964), and Morgan and Messenger (1979) carried out at the Survey Research Center of the Institute for Social Research, Universidad of Michigan. The program THAID (Theta Automatic Interaction Detection), created by Sonquist, Baker and Morgan (1971), was one of the first methdos to fit data based on decision trees.

In Statistics, Kass (1980) introduced a recursive non-binary algorithm, called CHAID (Chi-square automatic interaction detection). Later, Breiman, Friedman, Olshen and Stone (1984) introduced a new algorithm for construction of trees and applied it to regression and classification problems. The method is known as CART (Classification and regression trees). At the same time the Machine Learning comunity began to use the induction process through decision trees (Michalski, (1973), Quinlan (1983)). In the field of Pattern Recognition , Henrichon and Fu (1969), wrote a paper about nonparametric partitioning that is close related to decision trees, but it uses hyperplanes that no parallel to the axis of coordinates.

# Example of classification using trees



# Algorithms for decision trees

**C4.5.** Introduced by Quinlan (1993) among the Machine Learning comunity. It a descendent of ID3 (Quinlan, 1986).

**CHAID (**Chi-square automatic interaction detection). It was introduced by Kass (1980) and it derivates from THAID: "A sequential search program for the analysis of nominal scale dependent variables" (Morgan and Messenger, 1973). The criterion to partition is based on the $\chi 2$ statistic.

**NewId.** (Boswell, 1990). It is also a descendent of ID3 (Quinlan, 1986)

**CART.** Introduced by Breiman et al. (1984), properly is an algorithm for binary decision trees. There is a similar version called IndCART and it is available in the package IND distribuited by NASA. RPART (Recursive PARTicioning), a version of CART is available in R.

**Bayesian Trees**: It is based on application of bayesian methods to decision trees, Buntine (1992). It is available in the package IND distributed by NASA.

**CN2. I**ntroduced by Clark and Niblett (1989).

# Construction of decision trees

A decision tree partitions the space of predictors variables in a set of hyper-rectangles and in each of them it fits a simple model, usually a constant. That is, $y=c$, where y is the response variable.

The construction of a decision tree is based on four elements:

a) A set Q of binary questions of the form $\{ x \in A? \}$ where A is a subset of the sample space.

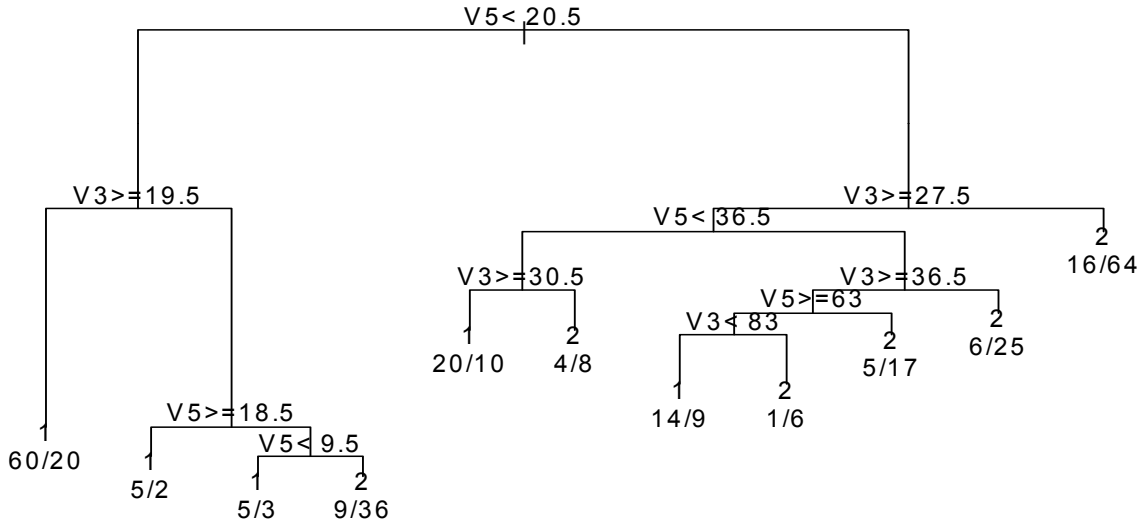
b) The method used to partition the nodes.

c)The strategy required to stop the growth of the tree.

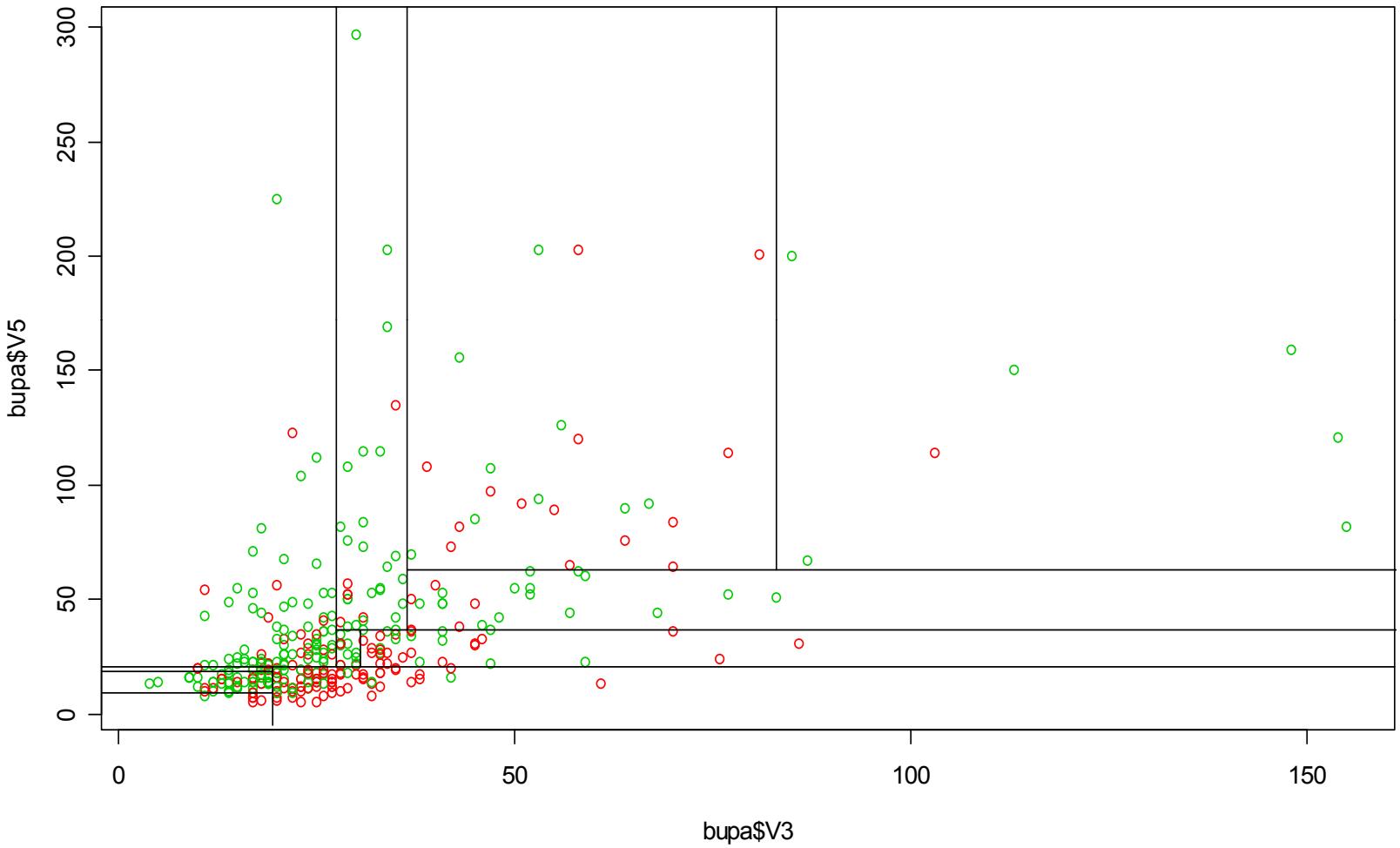
d)The assignment of a value of the response variable (in regression) or a class (in classification) to each terminal node.

The main differences among the decision trees algorithms are in the rules to partition the nodes, the strategy to prune the trees, and the treatment of missing values.

# Example: classification of bupa with attributes V3 y V5




```
> plot(arbolbupa,margin=.25)
> text(arbolbupa,use.n=T)
```




Partition of the sample space according to the decision tree.

# The set Q of binary questions

- Supongamos que el vector de variables predictoras es de la forma $\mathbf{x}=(x_1,\ldots..x_p)$, donde algunas de las variables xi son discretas y otras son continuas. Entonces, el conjunto Q de preguntas binarias en los nodos debe tener las siguientes características
  - a) cada división de los nodos depende del valor de una sola variable predictora
  - b) Si la variable $x_k$ es continua entonces Q incluye todas las preguntas de la forma {Es $x_k \leq c$}, donde c es cualquier número real. Usualmente c es el punto medio entre dos valores consecutivos de un atributo.

c) Si la variable $x_k$ es categórica tomando valores en $\{b_1,b_2,\ldots\ldots b_m\}$ entonces Q incluye todas las preguntas de la forma $\{ x_k \in A?\}$ donde A es un subconjunto cualquiera de $\{b_1,b_2,\ldots\ldots b_m\}$ . En total se puedeen considerar $2^{m-1}-1$

- Por ejemplo si x2, x3 y x4 son variables predictoras continuas y x1 es categórica con valores 0, 1 y 2, entonces Q incluye preguntas de la siguiente forma:

Es x3≤ 4.5?

Es x4≤ -1.4?

Es x1= 0 ó 1?

- También se puede usar divisiones en mas de dos nodos, pero no se recomienda porque el conjunto de datos se dividiría muy rápido dejando muy pocos datos para las subsiguientes divisiones.

# Procedimiento para particionar los nodos

La idea fundamental es que los nodos hijos sean más puros que los nodos padres. La partición de un nodo t del árbol T se hace de acuerdo a un criterio que es diseñado para producir nodos hijos que produzcan una suma de cuadrados de errores menor que la del nodo padre en el caso de regresión o que separen mejor las clases que el nodos padre en el caso de clasificación.

En árboles de clasificación sean $p(s)=\{\#\ i\leq N: Xi\in s\}/N$ la proporción de observaciones en el nodo s, y

$p(j/s)=\{\#\ i\leq N: Xi \in s \text{ y } Yi=j\}/\{\#\ i\leq N: Xi \in s\}$

la proporción de observaciones en el nodo s que pertenecen a la clase j (j=1,….,J), donde J es el número de clases.

El indice de la impureza del nodo s como $i(s)=\varphi(p(1/s),p(2/s),\ldots.p(J/s))$ donde $\varphi$ es una función de impureza, la cual debe satisfacer ciertas propiedades.

Entonces la regla para particionar el nodo t es como sigue:

Formar el nodo hijo derecho $t_R$ y el nodo hijo izquierdo $t_L$ tal que la disminución de la impureza dada por

$\Delta i(t)= i(t)-\{p(t_L)i(t_L)+p(t_R)i(t_R)\}$

sea máxima.

# Medidas de Impureza

Para árboles de clasificación se pueden usar las siguientes medidas de impureza.

a) El ***Coeficiente de Gini***. *Para el nodo t* ***se*** *define por*

$$i_G(t) = \sum_j \sum_k p(j/t)\,p(k/t) = \sum_{j=1}^{J} p(j/t)(1 - p(j/t))$$

Si hay dos clases (J=2) presentes entonces

$i_G(t)=2p(1-p)$

donde p es la proporción de observaciones en la primera clase.

El coeficiente de Gini se puede interpretar como uno en donde se clasifica cada observación de un nodo a una clase j con probabilidad p(j/t) en lugar de clasificar todas las observaciones a la clase con mayor número de observaciones.

b) La ***Entropia*** **Cruzada o Devianza o Impureza de Información** definida por

$i_E(t) = -\Sigma_j\, p(j/t)\log[p(j/t)]$

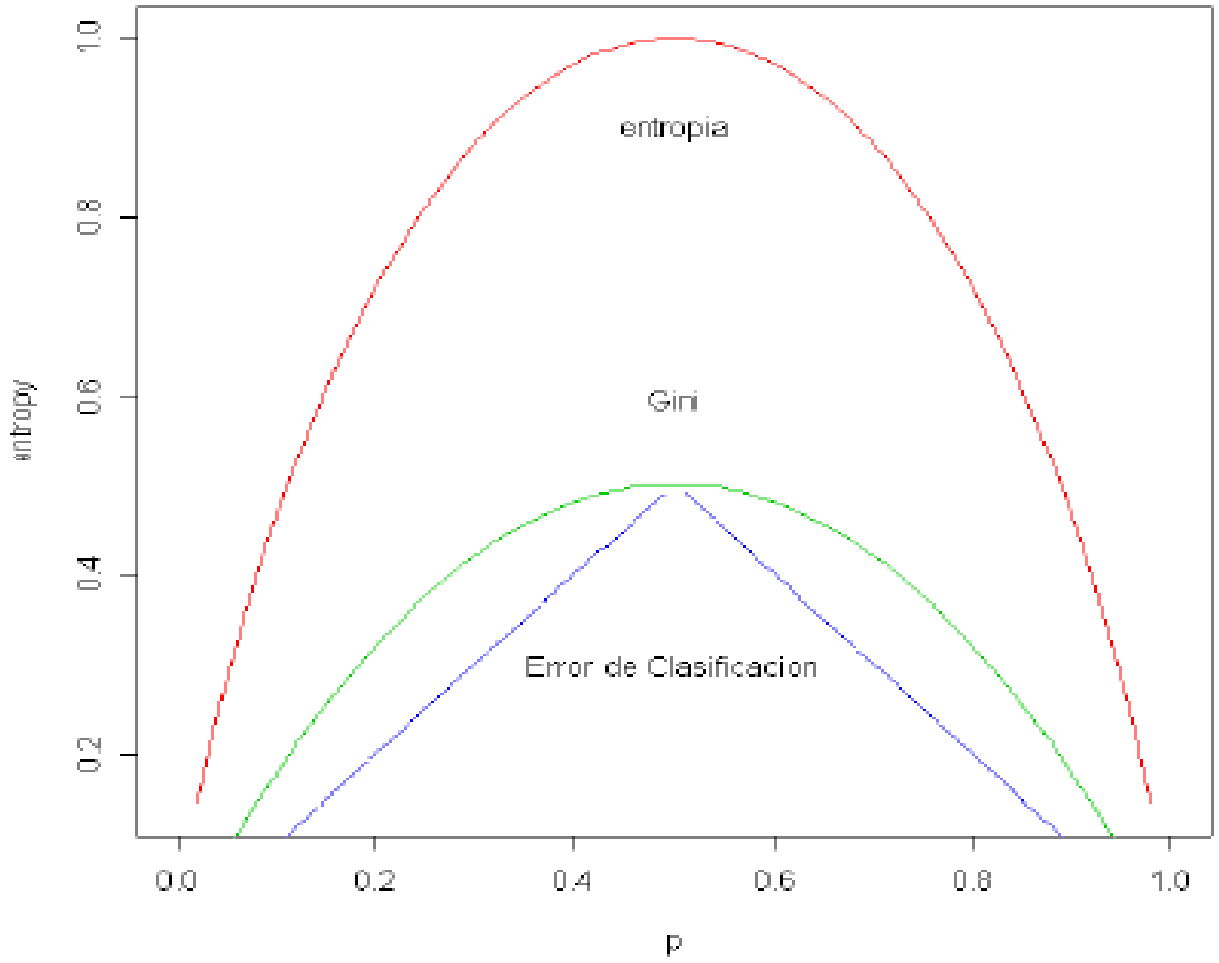
El logaritmo es tomado en base 2. Cuando se aplica entropía a distribuciones continuas se aplica logaritmo natural. Para dos clases se tiene

$i_E(t) = -2p*\log(p) - 2(1-p)*\log(1-p)$

En regresion, La Devianza es equivalente a la suma de cuadrados de los errores y está dada por el negativo del doble del logaritmo de la función de verosimilitud.

Rpart usa por default la impureza Gini, pero tiene la opción de usar la impureza de información.

- c) La ***tasa de Mala clasificación***, definida por $i_{MC}(t)=1-\max_j p(j/t)$ Para dos clases sería: iMC(t)=1-max(p,1-p)



Para árboles de regresión donde la variable de respuesta y es continua, se pueden usar las siguientes medidas de impureza

i) La **Varianza**, definida por

$$i(t) = \frac{\sum_{j \in t} (y_j - \bar{y}_t)^2}{n_t}$$

donde $\bar{y}_t$ es la media de la variable de respuesta en el nodo t.

ii) La **Desviación absoluta mediana**, definida por

$$i(t) = \frac{\sum_{j \in t} | y_j - med(y_t) |}{n_t}$$

# Criterios para parar el crecimiento del árbol.

La función **rpart** de R tiene varios criterios de parada que son aplicados simultáneamente y son controlados con la función **rpart.control** .

La opción **minsplit** fija el número mínimo de observaciones en un nodo para para que este sea dividido. Esta opción por defecto es 20.

La opción **minbucket** indica el número mínimo de observaciones en cualquier nodo terminal. Por defecto esta opción es el valor redondeado de minsplit/3.

La opción cp; parámetro de complejidad. Indica que si el criterio de impureza no es reducido en mas de cp*100% entonces se para..Por defecto cp=.01. Es decir, la reducción en la Devianza del nodo terminal debe ser menor del 1% de la devianza inicial.

Tambien hay un parámetro **maxdepth** que condiciona la profundidad máxima del arbol. Por defecto está establecida como 30.

# Recortando ("prunning") un árbol.

- Hacer crecer un árbol demasiado grande puede crear problemas de "sobreajuste" , es decir el modelo puede seguir más al ruido que a la señal.

Para cualquier árbol T y cualquier $\alpha \geq 0$ ( $\alpha$ es llamado el parámetro de complejidad), una medida del mérito del árbol T (o medida de costo-complejidad) está dada por:

$$R_\alpha(T) = Resub(T) + \alpha|T|$$

donde Resub(T) es estimado por resubsitución de la tasa de clasificación errada de T, |T| es el número de nodos terminales de T. Cuando $\alpha=0$ se obtiene el árbol más grande y cuando $\alpha=\propto$ se obtiene un árbol con un solo nodo. El árbol óptimo $T\alpha$ es el árbol mas pequeño que minimiza $R\alpha(T)$ y el parámetro $\alpha$ es encontrado usando validación cruzada, en particular la validación cruzada de 10

- La función **prune** de la librería **rpart** ejecuta recorte de un árbol. La opción cp de prune es llamado el parámetro de complejidad. El cp indica que se descartará cualquier partición que no disminuye la impureza por un factor igual a cp.

# Estimación del Error de Clasificación

- Breiman, et al (1984) recomiendan usar validación cruzada 10 para estimar el error de clasificación. Ellos no recomiendan "bootstrapping" porque han mostrado que el sesgo se reduce en menos del 40% del verdadero sesgo. Aunque, para el caso de muestras pequeñas el "boostrapping" podria ser usado.
- La función **xpred.rpart** da las predicciones de la variable de respuesta usando validación cruzada para valores dados del parámetro de complejidad.
- Arbolbupa=rpart(V7~.,data=bupa,method="class")
- > cvpred=xpred.rpart(arbolbupa,xval=10,cp=.1)
- > error=mean(cvpred !=bupa[,7])

# Tratamiento de valores perdidos en clasificación por árboles

El procedimiento para tratar los casos con observaciones perdidas es como sigue:

Se determina la mejor partición del nodo basado en una variable digamos xk con los datos que se tiene disponible. Si cuando se desea clasificar una observación del conjunto de entrenamiento o de prueba no hay el valor correspondiente de la variable xk entonces se usa la “partición sustituta“ y si no hubiera un valor

observado de la variable envuelta en la variable sustituta entonces se usa una segunda partición sustituta y asi sucesivamente. Esto es algo similar a cuando en un modelo lineal se reemplaza el valor perdido de una variable predictora por la regresión con la variable predictora restante que esta más altamente correlacionada con ella.

# Ventajas y desventajas de clasificación por arboles

**Ventajas:**

- Puede ser aplicado a cualquier tipo de variables predictoras: continuas y categóricas
- Los resultados son fáciles de entender e interpretar.
- No tiene problema de trabajar con datos perdidos.
- Hace automáticamente selección de variables.
- Es invariante a transformaciones de las variables predictoras.
- Es robusto a la presencia de "outliers".
- Es un clasificador noparamétrico, es decir que no requiere suposiciones.
- Toma en cuenta las interacciones que puede existir entre las variables predictoras.
- Es rápido de calcular.

**Desventajas:**

- El proceso de selección de variables es sesgado hacia las variables con mas valores diferentes.
- Dificultad para elegir el árbol óptimo
- La superficie de predicción no es muy suave, ya que son conjuntos de planos.
- Requiere un gran numero de datos para asegurarse que la cantidad de observaciones en los nodos terminales es significativa.
- Ausencia de una función global de las variables y como consecuencia pérdida de la representación geométrica.