Tema 9: Inducción de Reglas

Abdelmalik Moujahid, Iñaki Inza, Pedro Larrañaga

Departamento de Ciencias de la Computación e Inteligencia Artificial

Universidad del País Vasco

http://www.sc.ehu.es/isg/

Contenido

- Introducción
- El algoritmo IREP (Incremental Reduced Error Pruning) (Fürnkranz y Widner, 1994)
- El algoritmo RIPPER (Repeated Incremental Pruning Produce Error Reduction) (Cohen, 1995)

Introducción

Los sistemas de apendizaje de reglas representan un paradigma:

- Transparente
- Fácilmente comprensible y aplicable
- Mas genérico que los árboles de clasificación

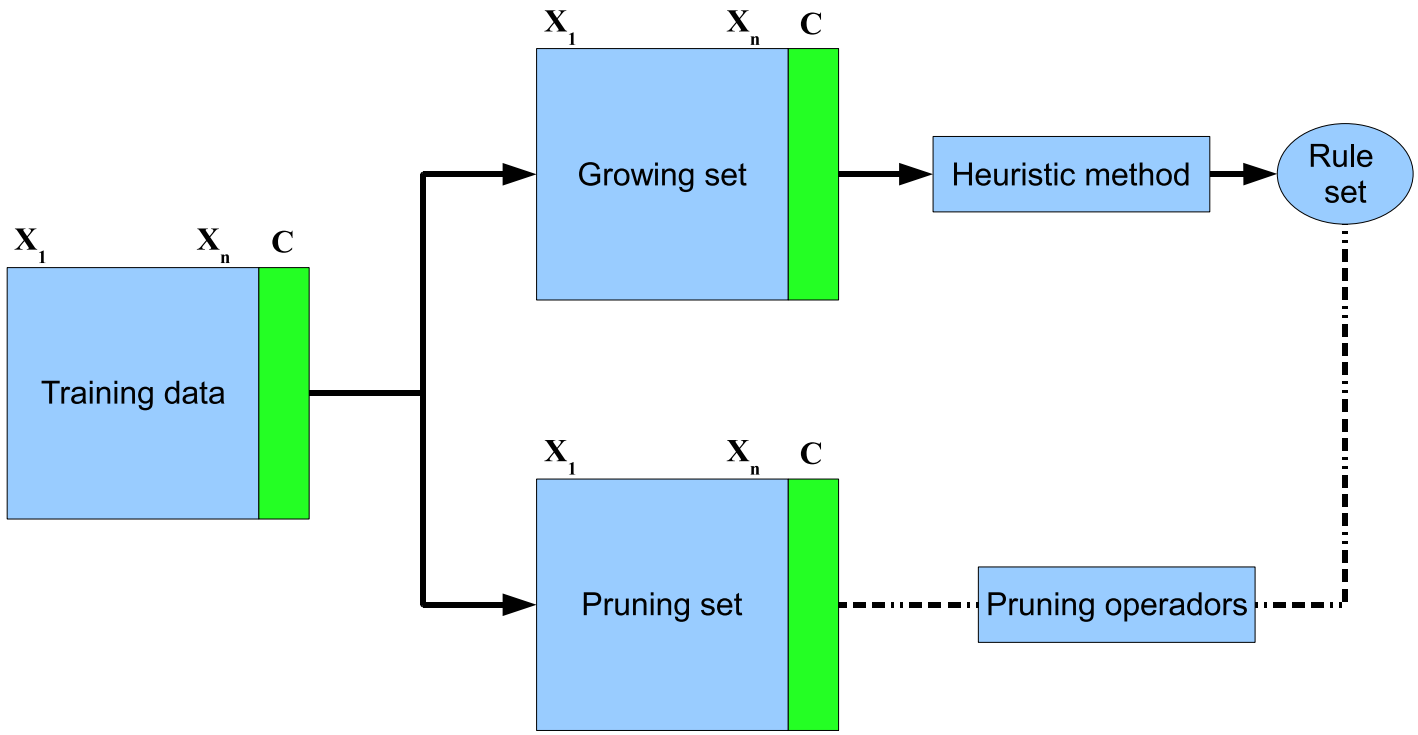
Introducción

Muchas de las técnicas utilizadas en los sistemas de aprendizaje de reglas fueron adaptadas del aprendizaje de árboles de decisión, el cual se basa en:

- la estrategia de aprendizaje conocida como overfit-and-simplify,
- la técnica de poda (o pruning) conocida como REP (reduced error pruning).

REP para sistemas de aprendizaje de reglas (Pagallo and Haussler, 1990)

Introducción



Partición del fichero de casos en el algoritmo REP

$IREP$ (*Incremental Reduced Error Pruning*)

El algoritmo de aprendizaje de reglas $IREP$ integra:

- el algoritmo REP (reduced error pruning)
- el algoritmo de aprendizaje de reglas separate-and-conquer

$IREP$ (*Incremental Reduced Error Pruning*)

$IREP$: Reglas en forma normal disyuntiva

- Una regla (*rule*)es una conjunción de literales

$$R_j \equiv \left(X_7 = x_7^1\right) \,\&\, \left(X_{14} = x_{14}^2\right) \,\&\, \left(X_{24} = x_{24}^1\right)$$

R_j constituida por la intersección de los 3 literales siguientes:

$$\left(X_7 = x_7^1\right), \left(X_{14} = x_{14}^2\right) \text{ y } \left(X_{24} = x_{24}^1\right)$$

- Un conjunto de reglas (*rule set*) está formado por una disyunción de reglas

$$R_1 \text{ or } R_2 \text{ or } \ldots \text{ or } R_k$$

- Una regla parcial (*partial rule*), R_j^{par} de una determinada regla R_j es la intersección de un subconjunto de los literales a partir de los cuales se forma R_j

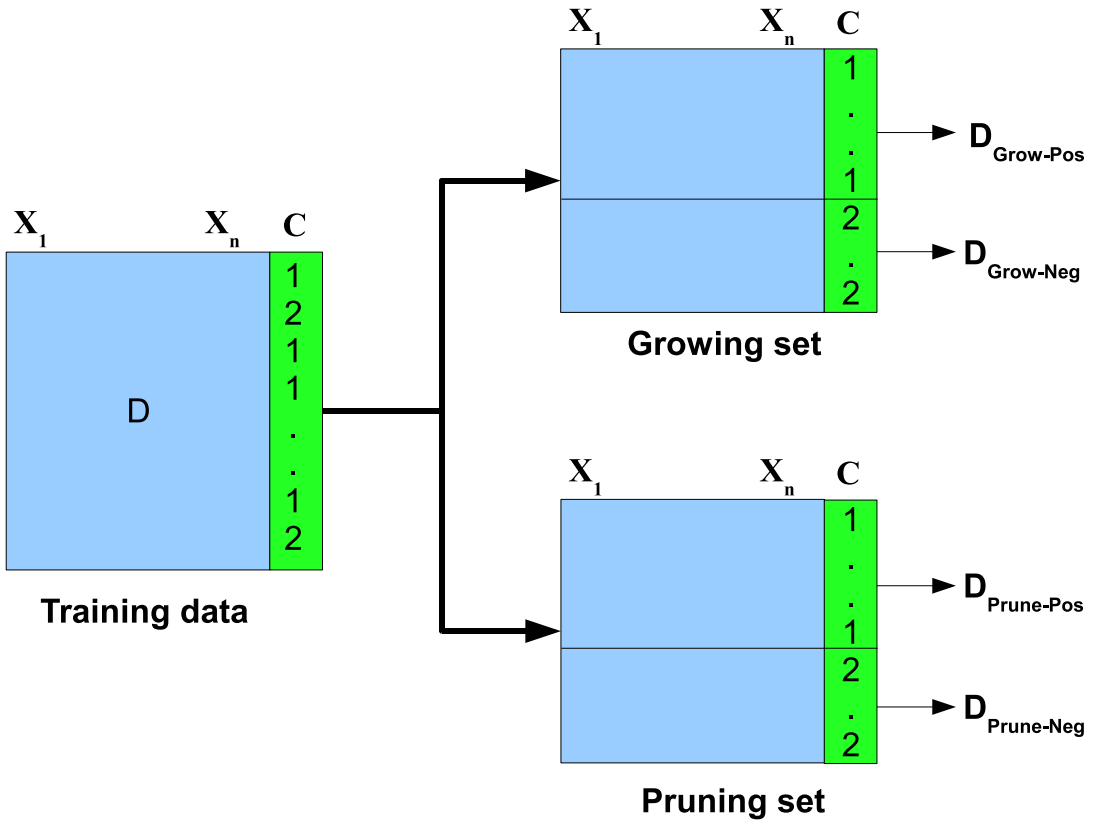
$$R_j^{par} \equiv \left(X_7 = x_7^1\right) \,\&\, \left(X_{14} = x_{14}^2\right)$$

$IREP$ (*Incremental Reduced Error Pruning*)

D conjunto de casos o patrones etiquetados se particiona en dos subconjuntos:

- D_{pos} conjunto de patrones positivos; D_{neg} conjunto de patrones negativos
- Cada uno de los subconjuntos anteriores se subdivide en otros dos subconjuntos:
  - $D_{grow-pos}$ y $D_{prune-pos}$ subconjuntos relacionados respectivamente con la construcción y el podado de las reglas
  - Análogamente $D_{grow-neg}$ y $D_{prune-neg}$
- $D = D_{pos} \cup D_{neg} = (D_{grow-pos} \cup D_{prune-pos}) \cup (D_{grow-neg} \cup D_{prune-neg})$
- $D_{pos} \cap D_{neg} = D_{grow-pos} \cap D_{prune-pos} = D_{grow-neg} \cap D_{prune-neg} = \emptyset$

IREP (*Incremental Reduced Error Pruning*)



Partición del fichero de casos etiquetados de partida D

$IREP$ (*Incremental Reduced Error Pruning*)

- $IREP$ induce –basándose en $D_{grow-pos}$ y $D_{grow-neg}$– el conjunto de reglas (*rule set*) de manera voraz, escogiéndose en cada paso añadir el mejor literal a la regla parcial *partial rule* en construcción
- *GrowRule* añade de forma repetida a la regla parcial (*partial rule*) R^{par} el literal que da origen a la regla parcial R'^{par} con mayor valor del criterio:

$$v\left(R^{par}, R'^{par}, D_{grow-pos}, D_{grow-neg}\right)$$

$$= cu\left[-log_2\left(\frac{pos}{pos+neg}\right) + log_2\left(\frac{pos'}{pos'+neg'}\right)\right]$$

done cu es el porcentaje de ejemplos en $D_{grow-pos}$ que siendo cubiertos por R^{par} están también cubiertos por R'^{par} (regla parcial más específica que R^{par}). pos (respectivamente neg) es el número de ejemplos en $D_{grow-pos}$ ($D_{grow-neg}$) cubiertos por la regla R^{par}. Y pos' (respectivamente neg') es el número de ejemplos en $D_{grow-pos}$ ($D_{grow-neg}$) cubiertos por la regla R'^{par}.

$IREP$ (*Incremental Reduced Error Pruning*)

Supongamos que la regla parcial, R^{par}, cubre 90 ejemplos de los cuales 70 son positivos y 20 negativos

- $R^{m,par}$ especialización mala de R^{par}, con la cual se van a cubrir 50 ejemplos positivos y 20 ejemplos negativos
- $R^{b,par}$ especialización buena de R^{par}, la cual cubre 70 ejemplos positivos y 10 ejemplos negativos
- $v\left(R^{par}, R^{m,par}, D_{grow-pos}, D_{grow-neg}\right)$ $= \frac{50}{70}\left[-log_2\left(\frac{70}{90}\right) + log_2\left(\frac{50}{70}\right)\right]$
- $v\left(R^{par}, R^{b,par}, D_{grow-pos}, D_{grow-neg}\right)$ $= \frac{70}{70}\left[-log_2\left(\frac{70}{90}\right) + log_2\left(\frac{70}{80}\right)\right]$
- El valor del criterio para $R^{b,par}$ es superior al correspondiente al $R^{m,par}$

$IREP$ (*Incremental Reduced Error Pruning*)

- El proceso de crecimiento, `GrowRule`, finaliza cuando no se encuentra ningún literal cuya inclusión en la regla parcial permita que la regla especializada mejore el criterio $v\left(R^{par}, R'^{par}, D_{grow-pos}, D_{grow-neg}\right)$)
- Entonces comienza con el proceso de podado, `PruneRule`, de dicha regla
- `PruneRule` plantea el borrado, de manera secuencial, y empezando por el último literal introducido a la regla en su fase de crecimiento
- Se van a ir borrando (podando) literales mientras se mejore el criterio $v\left(Rule, D_{prune-pos}, D_{prune-neg}\right)$, siendo

$$v\left(Rule, D_{prune-pos}, D_{prune-neg}\right) = \frac{pos + (Neg - neg)}{Pos + Neg}$$

donde pos (respectivamente neg) es el número de ejemplos en $D_{prune-pos}$ ($D_{prune-neg}$) cubiertos por la regla, y Pos (respectivamente Neg) es el número de ejemplos en $D_{prune-pos}$ ($D_{prune-neg}$).

$IREP$ (*Incremental Reduced Error Pruning*)

Procedure *IREP*
Begin
Ruleset= $\emptyset$
while $D_{Pos} = D_{Grow-Pos} \cup D_{Prune-Pos} \neq \emptyset$ **do**
/* Construir y podar una nueva regla */
Dividir D en $(D_{Grow-Pos} \cup D_{Grow-Neg}) \cup (D_{Prune-Pos} \cup D_{Prune-Neg})$
Rule:=GrowRule($D_{Grow-Pos} \cup D_{Grow-Neg}$)
Rule:=PruneRule(Rule, $D_{Prune-Pos}, D_{Prune-Neg}$)
if la tasa de error de Rule en $(D_{Prune-Pos} \cup D_{Prune-Neg}) > 50\,\%$
then
return RuleSet
else
Añadir Rule a RuleSet
Borrar ejemplos cubiertos por Rule de D
endif
end while
return RuleSet
End

$RIPPER$ (*Repeated Incremental Pruning Produce Error Reduction*)

$RIPPER$ (Cohen (1995)) mejora de $IREP$ en tres aspectos:

1. *Métrica alternativa para la fase de poda*

- R_1 cubre 2000 ejemplos positivos en $D_{prune-pos}$ y 1000 ejemplos negativos en $D_{prune-neg}$. R_2 cubre 1000 ejemplos positivos en $D_{prune-pos}$ y 1 ejemplo negativo en $D_{prune-neg}$
- $IREP$ va a preferir R_1 a R_2, ya que

$$\frac{2000 + (Neg - 1000)}{Pos + Neg} > \frac{1000 + (Neg - 1)}{Pos + Neg}$$

  y sin embargo es intuitivo que la R_2 es preferible a R_1
- $RIPPER$ basa su poda en el criterio siguiente:

$$v\left(Rule, D_{prune-pos}, D_{prune-neg}\right) = \frac{pos - neg}{pos + neg}$$

$$\frac{2000 - 1000}{3000} < \frac{1000 - 1}{1001}$$

y R_2 se seleccionaría frente a R_1

$RIPPER$ (*Repeated Incremental Pruning Produce Error Reduction*)

2. Incorporación de un *heurístico* para determinar cuándo parar el proceso de añadir reglas

3. $RIPPER$ –posteriormente a todo el proceso visto para $IREP$– efectúa una *búsqueda local para optimizar el conjunto de reglas* (*rule set*) de dos maneras diferentes:

- Reemplazando una regla R_i que forma parte del *rule set* { $R_1, \ldots, R_{i-1}, R_i, R_{i+1}, \ldots, R_k$ } por R'_i, siempre y cuando el *rule set* correspondiente tenga un menor error en la clasificación en $D_{prune-pos} \cup D_{prune-neg}$
- Revisar una determinada regla R_i añadiendo literales para que así se consiga un menor error en $D_{prune-pos} \cup D_{prune-neg}$