

Clasificación ordinal y monotónica

Minería de Datos: Aspectos Avanzados

Salvador García

salvagl@decsai.ugr.es

Objetivos

- Entender el problema de la clasificación ordinal y situarlo en el contexto de la predicción.
- Conocer las restricciones de monotonicidad que se pueden imponer a problemas ordinales.
- Estudiar diferentes medidas de evaluación técnicas de clasificación ordinal y monotónica.
- Presentar algunas propuestas clásicas en ambos enfoques.

CLASIFICACIÓN ORDINAL Y MONOTÓNICA

1. Clasificación Ordinal

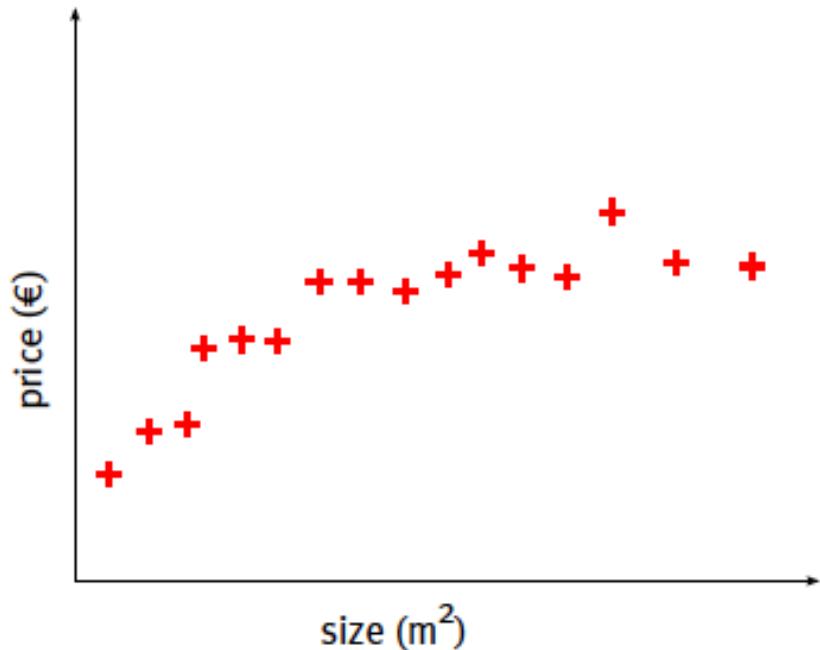
2. Clasificación Monotónica

CLASIFICACIÓN ORDINAL Y MONOTÓNICA

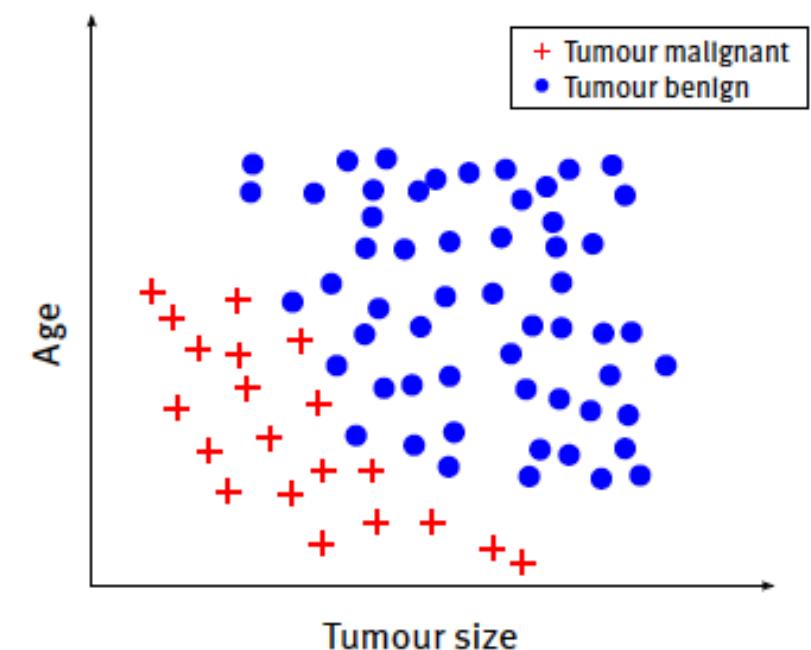
1. Clasificación Ordinal

2. Clasificación Monotónica

APRENDIZAJE SUPERVISADO

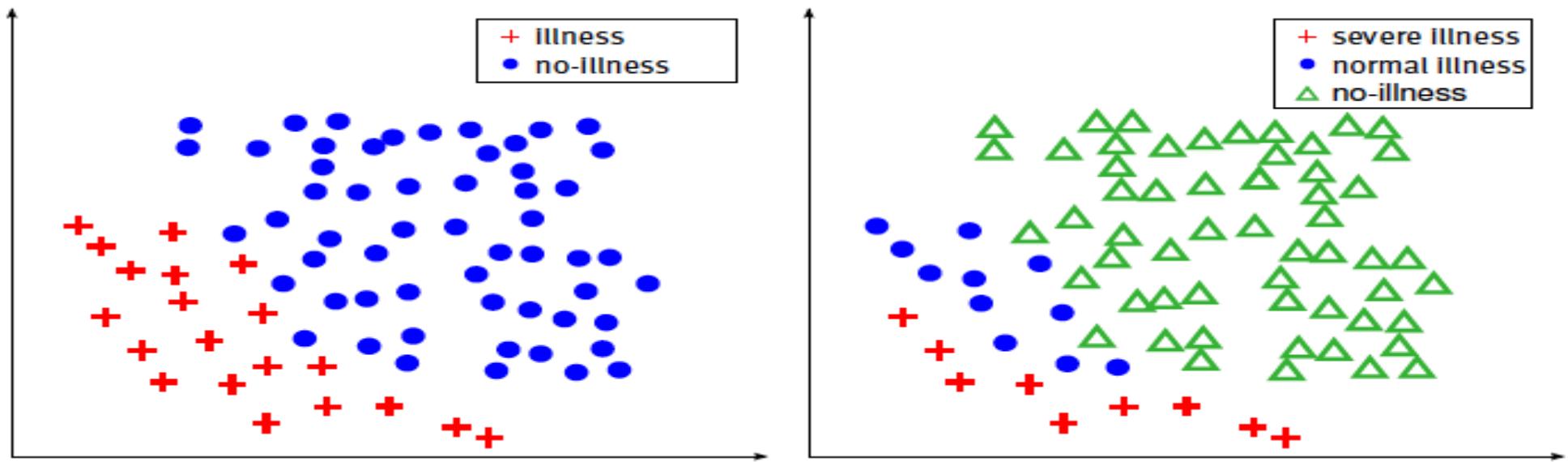


Problema de regresión
“Dados estos datos, para una casa de 75m², ¿cuánto se espera obtener?”

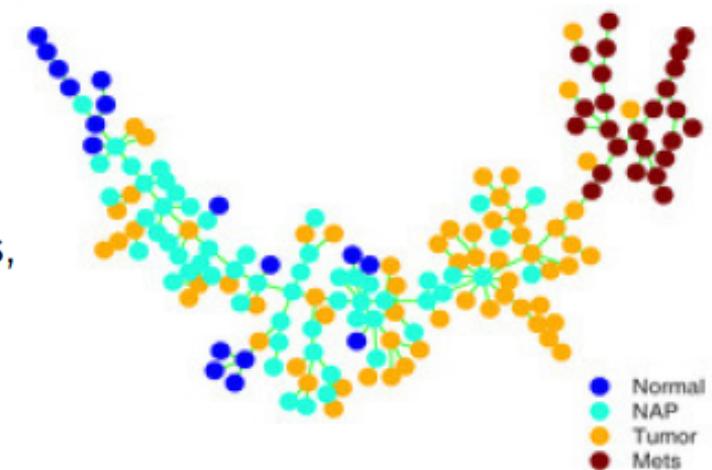


Problema de clasificación
“¿Se puede obtener el tipo de tumor dada la edad y el tamaño del mismo?”

CLASIFICACIÓN BINARIA VS. ORDINAL



Example: Normal cells, Near tumor cells, tumor cells, metastasis cells



REGRESIÓN/CLASIFICACIÓN ORDINAL

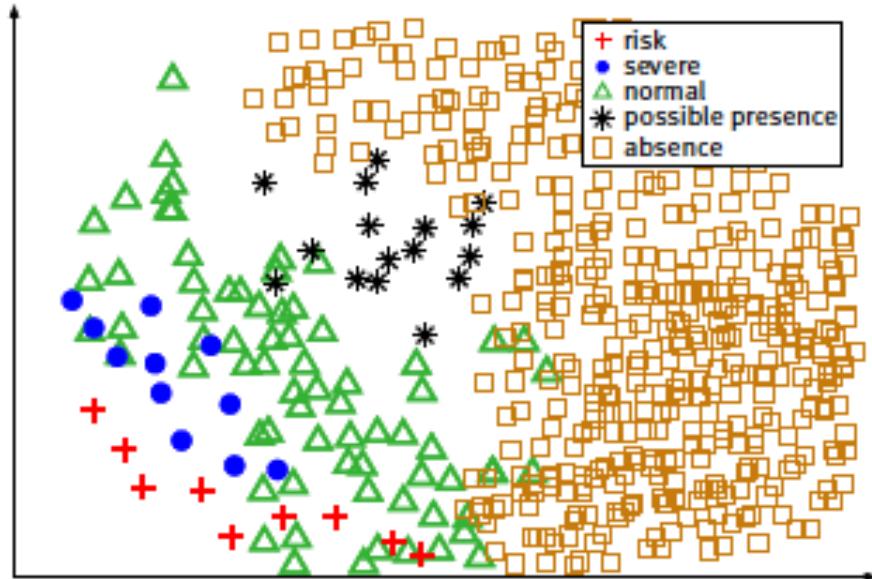
Definición: También llamada ranking/sorting, es un problema de aprendizaje supervisado para predecir categorías que tienen una disposición ordenada.

Objetivos:

Explotar las relaciones ordinales de los datos.
Minimizar errores que consideran el orden entre clases

Aplicaciones: Evaluaciones de personal, rating de seguros y créditos, predicciones de producción, ...

EJEMPLO: CLASIFICACIÓN DE DOLENCIAS



Dolencias basadas en una escala ordinal

{ $C_1 = \text{riesgo}$, $C_2 = \text{severa}$,
 $C_3 = \text{normal}$, $C_4 = \text{ posible presencia}$, $C_5 = \text{ausencia}$ }

Las etiquetas de clase se inspiran en información ordinal.
Los costes de clasificaciones erróneas no son los mismos para clases diferentes.
Las clases no balanceadas pueden ser muy comunes.

FORMULACIÓN DEL PROBLEMA

El propósito es aprender una función ϕ desde el espacio de entrada X a un conjunto finito $C = \{C_1, C_2, \dots, C_Q\}$ conteniendo Q etiquetas, donde el conjunto de etiquetas tiene una relación de orden lineal

$$C_1 \prec C_2 \prec \dots \prec C_Q$$

Cada ejemplo se representa por un vector K -dimensional $x \in X \subseteq \Re^K$ y una etiqueta de clase $y \in C$

El conjunto de entrenamiento T se compone de N ejemplos

$$T = \{(X, Y) = (x_i, y_i) : x_i \in X, y_i \in C (i = 1, \dots, N)\}$$

Ordinal Regression commonly taken as standard multinomial classification:

- Ignore ordering information → Worse Classifiers.
- All mistakes are equally penalised:
 - Is it the same classifying an *A* student as *E* than classifying a *D* student as *E*?

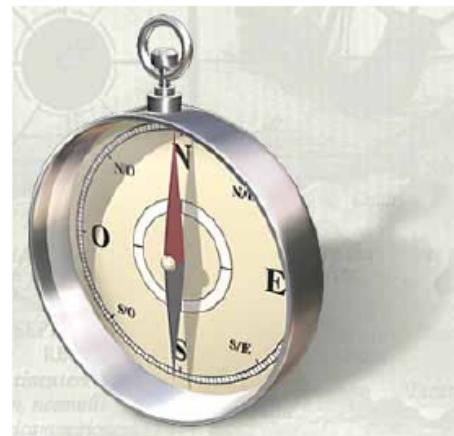
If we consider Ordinal Regression as standard regression:

- How do we assign a value to the labels?
 - Is there the same distance between *D* and *E* than between *A* and *B*?
 - Generally, it strongly depends on the problem considered.
 - Sometimes, we don't have information (subjective evaluations) and there is no way to decide the value of these labels.

Learning-to-rank problems: sometimes the word **ranking** is used to refer to a slightly different scenario, where the objective is to order all pairs of samples:

$$\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\} \Rightarrow \{\mathbf{x}_N, \mathbf{x}_1, \dots, \mathbf{x}_2\}$$

Circular ordinal regression: directional predictions.



MEDIDAS DE RENDIMIENTO

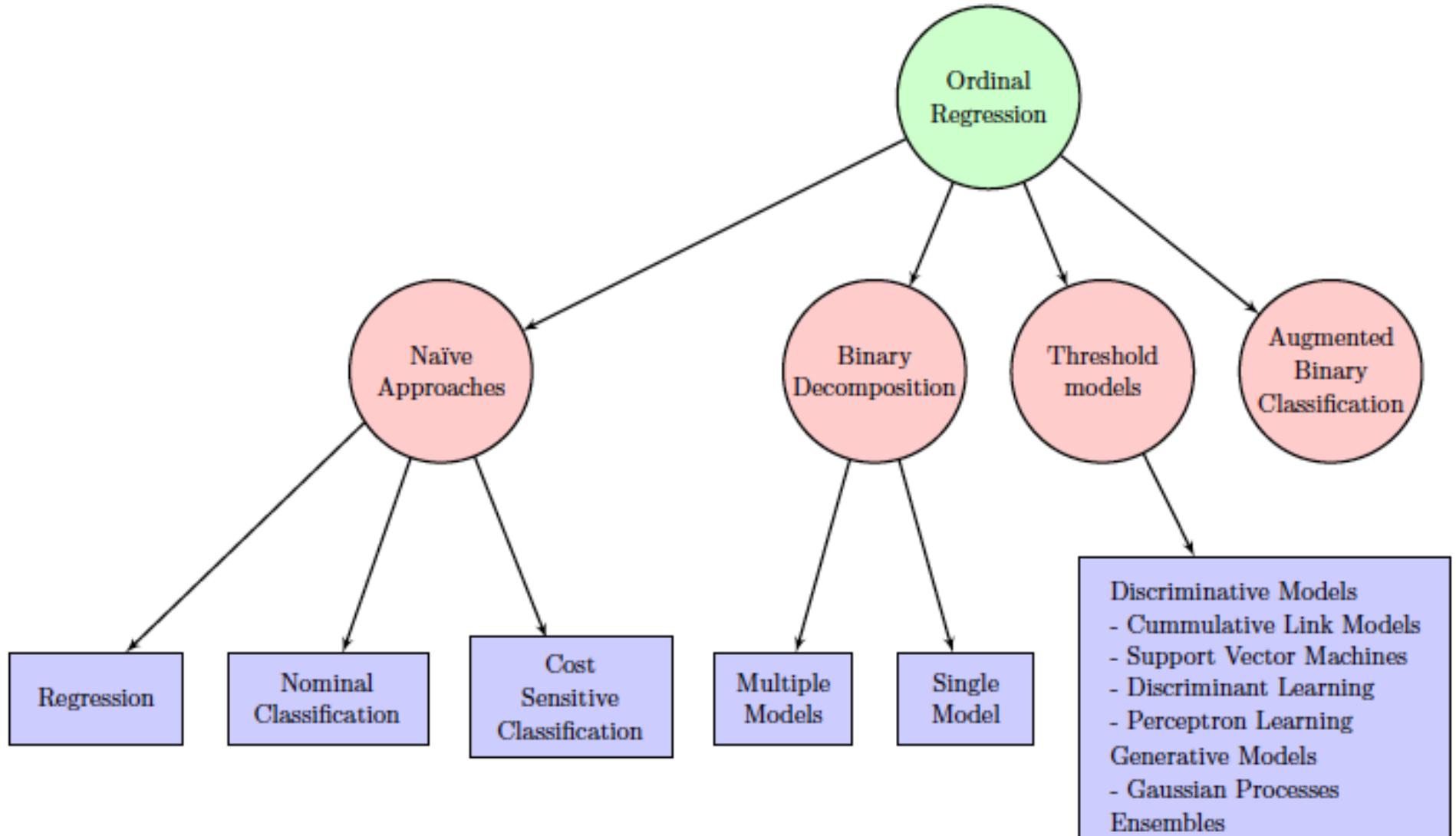
Mean Zero One Error (MZE)

$$MZE = 1 - Acc = \frac{1}{N} \sum_{i=1}^N \llbracket y_i^* \neq y_i \rrbracket$$

Mean Absolute Error (MAE)

$$MAE = \frac{1}{N} \sum_{i=1}^N |\mathcal{O}(y_i) - \mathcal{O}(y_i^*)|$$

TAXONOMÍA



Simplify to regression

- Kramer, S., Pfahringer, B., Widmer, G., Groeve, M.D.: Prediction of ordinal classes using regression trees. *Fundamenta Informaticae* 47, 1001-1013 (2001)
- Transform **each category into a numerical value**, learn a regression function, and round its result for assigning the class when predicting new samples (post-processing).
- Alternatively, the output of each leaf can be forced to be a valid integer value, by considering the median, the mode...
- A problem with this approach is that there might be no principled way of devising an appropriate mapping function since the true metric distances between the ordinal scales are unknown in most of the tasks.

Using cost matrixes

- S.B. Kotsiantis and P.E. Pintelas: A Cost Sensitive Technique for Ordinal Classification Problems. LNAI 3025, 220-229 (2004)
- The conditional risk for choosing a class \mathcal{C}_i is defined as:

$$R(\mathcal{C}_i|\mathbf{x}) = \sum_{j=1}^{J-1} C_{ij} P(\mathcal{C}_j|\mathbf{x}) \quad (6)$$

where C is the cost matrix.

- In this paper, C4.5, PART and 3-NN algorithms are shown to obtain better *MAE* values when the cost matrixes are used, without harming (and even improving) A .

Threshold methods

- Learn a real valued function $f : X \rightarrow \mathbb{R}$,
- together with a set of thresholds $b^1 \leq \dots \leq b^{J-1}$
- The rule for predicting new samples is:

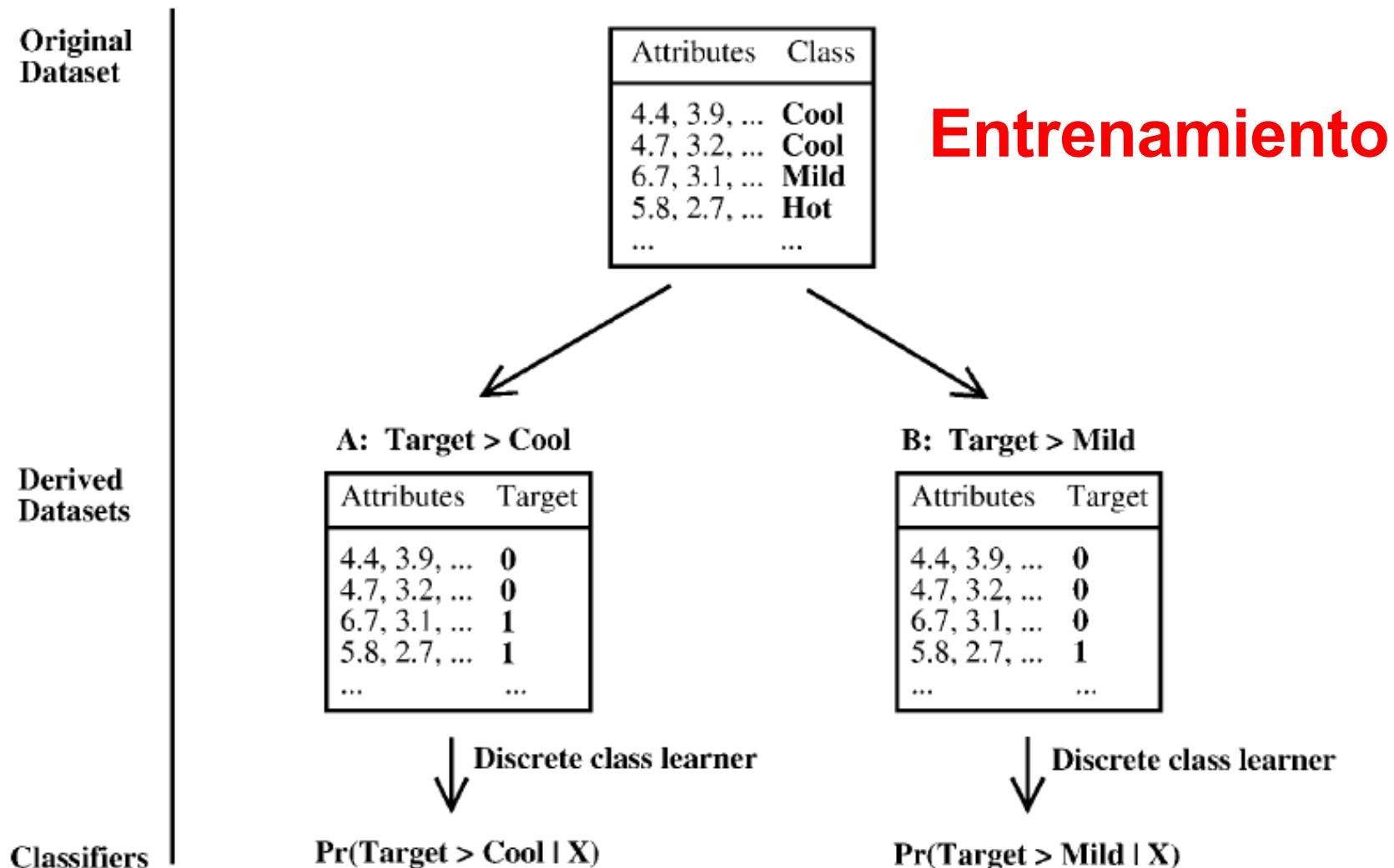
$$g(\mathbf{x}) = \begin{cases} c_1, & \text{if } f(\mathbf{x}) \leq b^1 \\ c_2, & \text{if } b^1 < f(\mathbf{x}) \leq b^2 \\ \dots \\ c_J, & \text{if } f(\mathbf{x}) > b^{J-1} \end{cases} \quad (7)$$

where $f(\mathbf{x})$ is a ranking function.

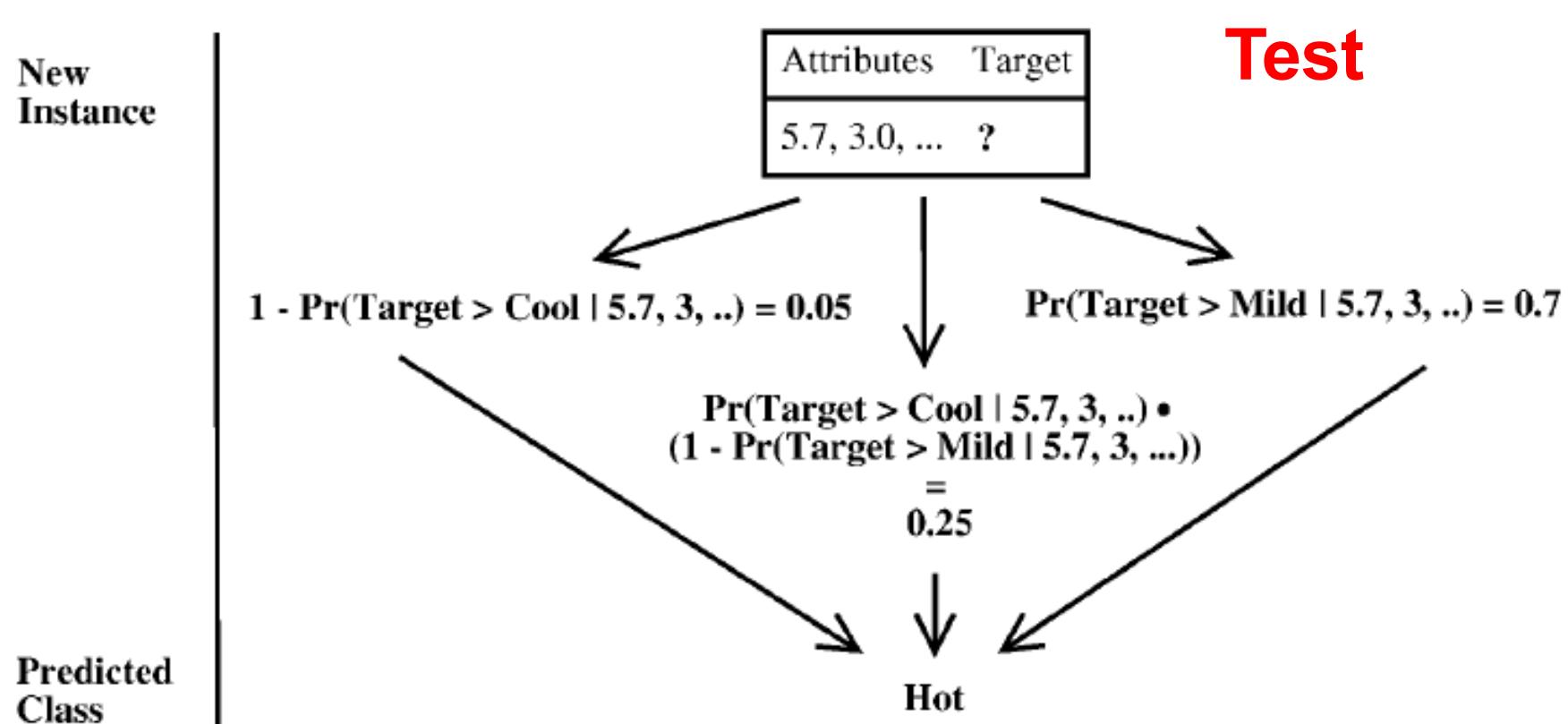
Multipe Model for Ordinal Classification

- Se trata de un método muy conocido para transformar cualquier algoritmo de clasificación estándar en un algoritmo de clasificación ordinal.

C4.5 para clasificación Ordinal

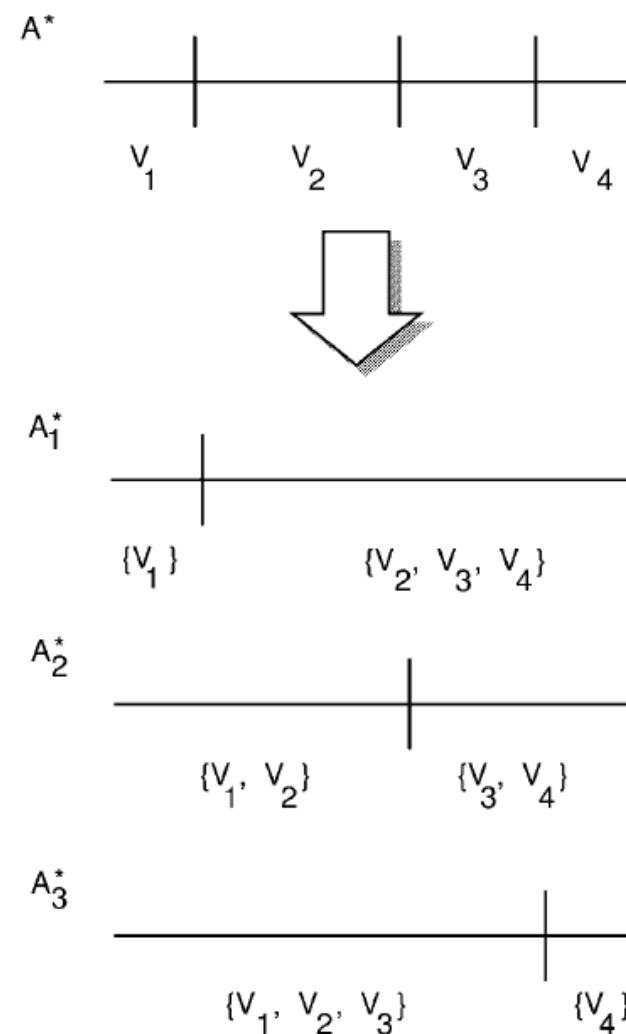


C4.5 para clasificación Ordinal



C4.5 para clasificación Ordinal

- El proceso es sencillo.
- Los datos se transforman de un problema de k clases ordinal a un problema de $k - 1$ clases binarias.
- El atributo i -ésimo representa el test $A^* > V_i$.



C4.5 para clasificación Ordinal

- El entrenamiento comienza generando los nuevos data sets, hasta $k - 1$ problemas binarios. Cada uno contiene los mismos valores de atributos de entrada, pero la clase cambia a binaria dependiendo de la desigualdad que tiene cada data set asociada.
- Para predecir el valor clase de una instancia no vista, necesitamos estimar las probabilidades de las k clases ordinales a partir de nuestros $k - 1$ modelos. En general, para los valores de clase V_i :

$$Pr(V_1) = 1 - Pr(\text{Target} > V_1)$$

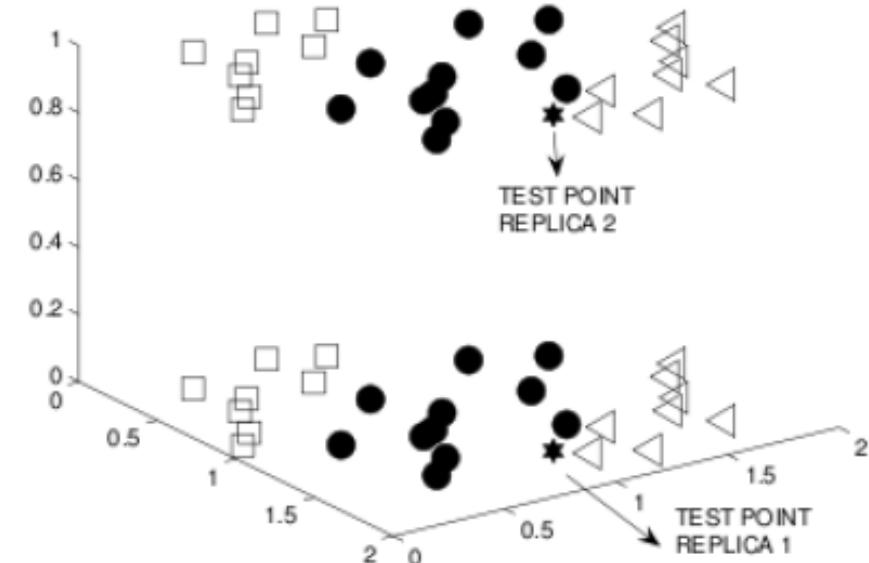
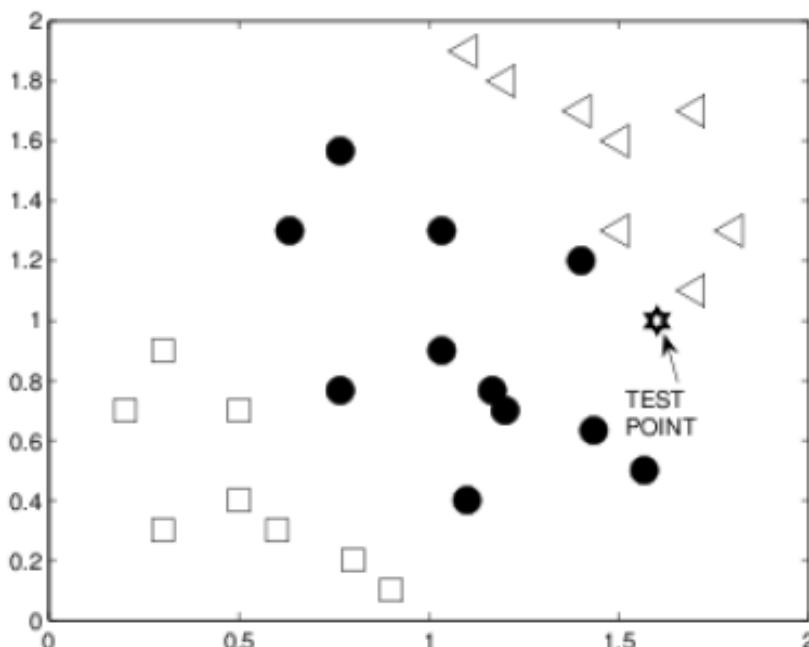
$$Pr(V_i) = Pr(\text{Target} > V_{i-1}) \times (1 - Pr(\text{Target} > V_i)) , 1 < i < k$$

$$Pr(V_k) = Pr(\text{Target} > V_{k-1})$$

- A la hora de predecir, una instancia con clase desconocida se procesa por cada uno de los $k - 1$ clasificadores, calculando la probabilidad como se indica arriba. La clase con máxima probabilidad se asigna a la instancia.

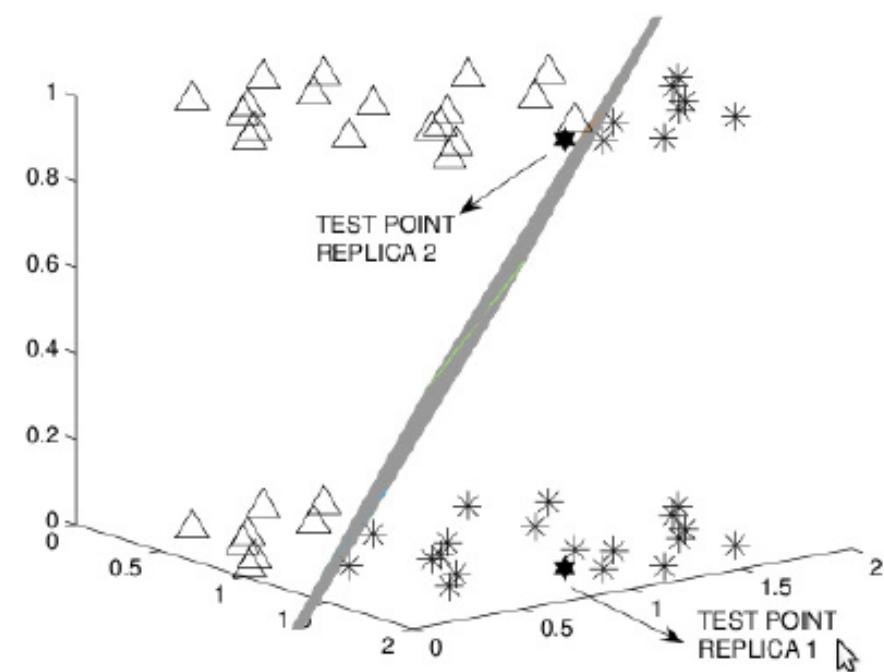
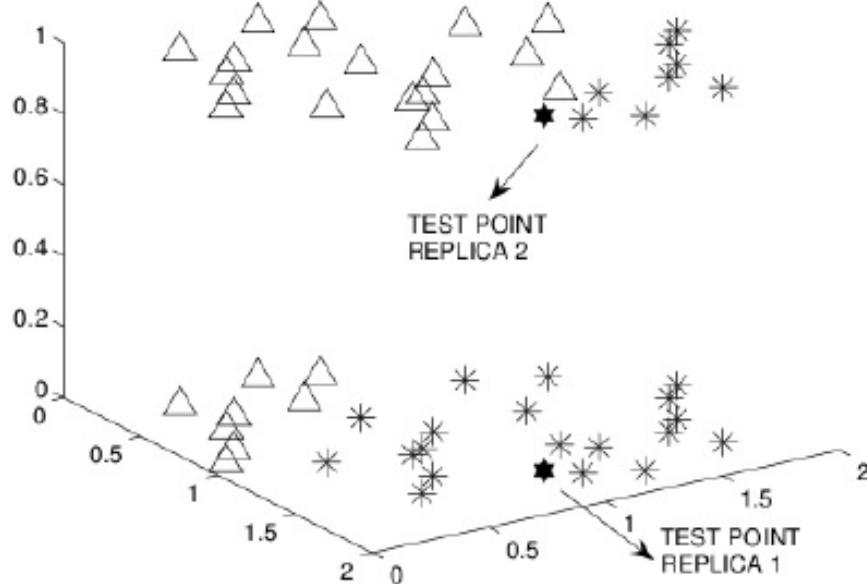
Clasificación Binaria Aumentada

Cardoso & da Costa: Add additional dimensions and replicate your data points:



Clasificación Binaria Aumentada

Cardoso & da Costa: Assign a binary label and learn a binary classifiers over the new space:



Survey y Software de Clasificación Ordinal

IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING

Ordinal regression methods: survey and experimental study

Pedro Antonio Gutiérrez, *Senior Member, IEEE*, María Pérez-Ortiz, Javier Sánchez-Monedero, Francisco Fernández-Navarro, and César Hervás-Martínez, *Senior Member, IEEE*

Abstract—Ordinal regression problems are those machine learning problems where the objective is to classify patterns using a categorical scale which shows a natural order between the labels. Many real-world applications present this labelling structure and that has increased the number of methods and algorithms developed over the last years in this field. Although ordinal regression can be faced using standard nominal classification techniques, there are several algorithms which can specifically benefit from the ordering information. Therefore, this paper is aimed at reviewing the state of the art on these techniques and proposing a taxonomy based on how the models are constructed to take the order into account. Furthermore, a thorough experimental study is proposed to check if the use of the order information improves the performance of the models obtained, considering some of the approaches within the taxonomy. The results confirm that ordering information benefits ordinal models improving their accuracy and the closeness of the predictions to actual targets in the ordinal scale.

CLASIFICACIÓN ORDINAL Y MONOTÓNICA

1. Clasificación Ordinal

2. Clasificación Monotónica

CLASIFICACIÓN MONOTÓNICA

- La clasificación con restricciones monotónicas, también llamada clasificación monotónica, es un problema de clasificación ordinal donde una restricción monotónica es clara: un valor más alto de un atributo en un ejemplo, fijando el resto de valores, no debería reducir el valor de la asignación de clase.
- Un clasificador monotónico es aquel que tiene como objetivo no violar las restricciones monotónicas en los modelos aprendidos.

Definición de monotonía (I)

Sea $X = (x_1, x_2, \dots, x_n)$ e $Y = (y_1, y_2, \dots, y_n)$ dos instancias del mismo problema con n atributos

$$X = Y \quad \text{si } x_i = y_i \quad \forall i = 1, 2, \dots, n$$

$$X > Y \quad \text{si } x_i > y_i \quad \forall i = 1, 2, \dots, n$$

$$X \geq Y \quad \text{si } x_i > y_i \quad \parallel x_i = y_i \quad \forall i = 1, 2, \dots, n$$

$$X < Y \quad \text{si } x_i < y_i \quad \forall i = 1, 2, \dots, n$$

$$X \leq Y \quad \text{si } x_i < y_i \quad \parallel x_i = y_i \quad \forall i = 1, 2, \dots, n$$

Definición de monotonía (II)

Sea $X = (X, C_x)$ e $Y = (Y, C_y)$ representan dos parejas atributos-clase. Las clases de X e Y se denotan por C_x y C_y respectivamente. (X, C_x) y (Y, C_x) son no-monotónicos con respecto a cada uno si:

$$X < Y \quad \wedge \quad C_x > C_y \quad \parallel$$

$$X > Y \quad \wedge \quad C_x < C_y \quad \parallel$$

$$X = Y \quad \wedge \quad C_x \neq C_y$$

Dos parejas atributos-clase = (X, C_x) e (Y, C_y) son monotónicas con respecto a la otra si no cumple cualquiera de las condiciones de la definición anterior.

Órdenes Parciales

- Para aplicar la monotonicidad a vectores (instancias), el orden natural (total) debe extenderse para permitir órdenes de vectores en \mathbb{R}^p
- A éstas relaciones se les llama órdenes parciales

$$\mathbf{x} \preceq \mathbf{x}' \iff \forall i = 1..p, x_i \leq x'_i$$

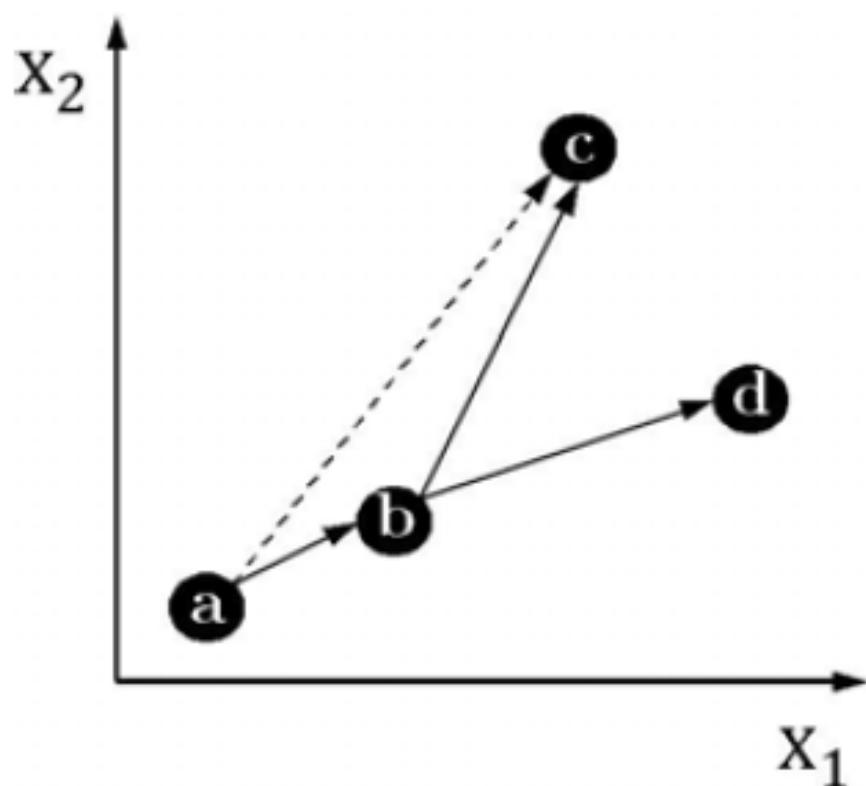
- Función monotónica completa:

$$\mathbf{x} \preceq \mathbf{x}' \Rightarrow f(\mathbf{x}) \leq f(\mathbf{x}'), \quad \forall \mathbf{x}, \mathbf{x}' \in \mathbb{R}^p$$

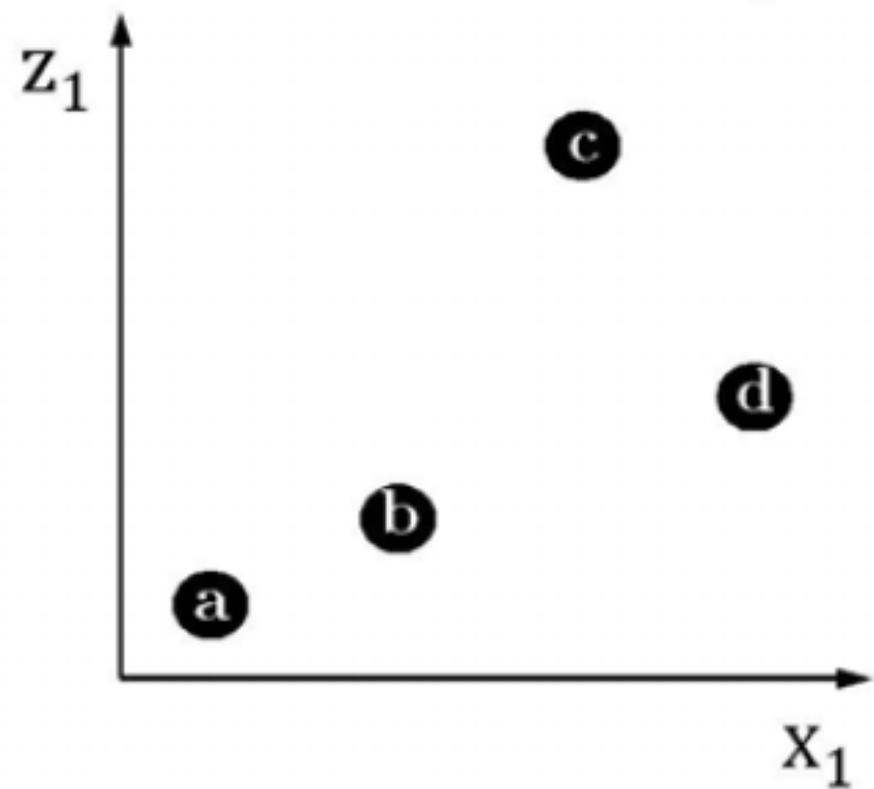
- Propiedades:
 1. *Reflexivity:* $\mathbf{a} \preceq \mathbf{a}$
 2. *Transitivity:* $\mathbf{a} \preceq \mathbf{b} \wedge \mathbf{b} \preceq \mathbf{c} \Rightarrow \mathbf{a} \preceq \mathbf{c}$
 3. *Antisymmetry:* $\mathbf{a} \preceq \mathbf{b} \wedge \mathbf{b} \preceq \mathbf{a} \Rightarrow \mathbf{a} = \mathbf{b}$

Comparabilidad

Monotone in x_1, x_2



Monotone in x_1 ,
Nonmonotone in z_1



\mathbf{x} comparable with $\mathbf{x}' \Leftrightarrow \mathbf{x} \preceq \mathbf{x}' \text{ or } \mathbf{x}' \preceq \mathbf{x}$

Funciones Parcialmente Monótonas

- Puede haber atributos monótonos y no monótonos.
- El espacio de entrada se puede dividir en X (atributos monótonos) y Z (atributos no monótonos).
- El orden parcial se define ahora como:

$$(\mathbf{x}, \mathbf{z}) \preceq_{\mathcal{X}} (\mathbf{x}', \mathbf{z}') \Leftrightarrow \mathbf{x} \preceq \mathbf{x}' \wedge \mathbf{z} = \mathbf{z}'$$

- La función monótona se define como:

$$\mathbf{x} \preceq \mathbf{x}' \Rightarrow f(\mathbf{x}, \mathbf{z}) \leq f(\mathbf{x}', \mathbf{z}), \quad \forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}, \mathbf{z} \in \mathcal{Z}$$

Conjuntos de Datos Monótonos

- Conjunto totalmente monótono:

$$\mathbf{x}_i \preceq \mathbf{x}_j \Rightarrow y_i \leq y_j \quad \forall (\mathbf{x}_i, y_i), (\mathbf{x}_j, y_j) \in \mathcal{D}$$

- Conjunto parcialmente monótono:

$$(\mathbf{x}_i, \mathbf{z}_i) \preceq_{\mathcal{X}} (\mathbf{x}_j, \mathbf{z}_j) \Rightarrow y_i \leq y_j \quad \forall ((\mathbf{x}_i, \mathbf{z}_i), y_i), ((\mathbf{x}_j, \mathbf{z}_j), y_j) \in \mathcal{D}$$

- Problemas de comparabilidad.
- Real o no?

CLASIFICACIÓN MONOTÓNICA: Datasets

- Antes de empezar, hay que estudiar el sentido de los datos. Ejemplo en conjunto Auto-Mpg:

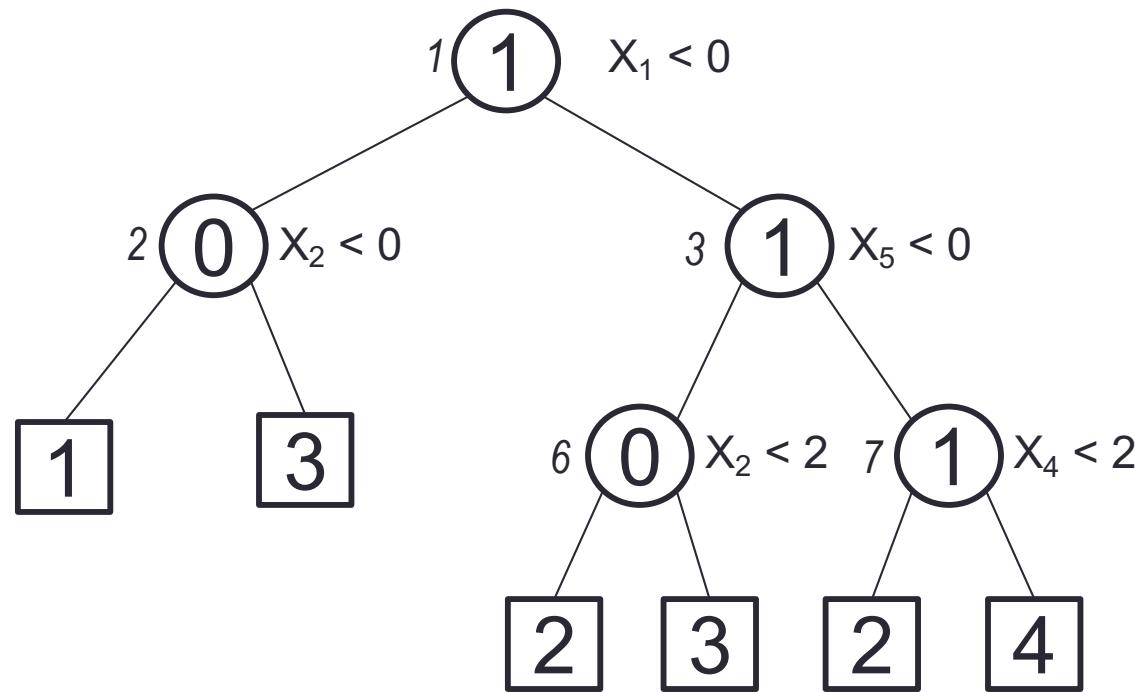
Attribute	Type	Sign
mpg	continuous	target
cylinders	multi-valued discrete	—
displacement	continuous	—
horsepower	continuous	—
weight	continuous	—
acceleration	continuous	+
model year	multi-valued discrete	+
origin	multi-valued discrete	+

CLASIFICACIÓN MONOTÓNICA: Datasets Clásicos

Dataset	#Instancias	#Atributos	#Clases
Employee Rejection Acceptance (ERA)	1000	4	9
Employee Selection (ESL)	488	4	9
Lectures Evaluation (LEV)	1000	4	5
Social Workers Decision (SWD)	1000	10	4

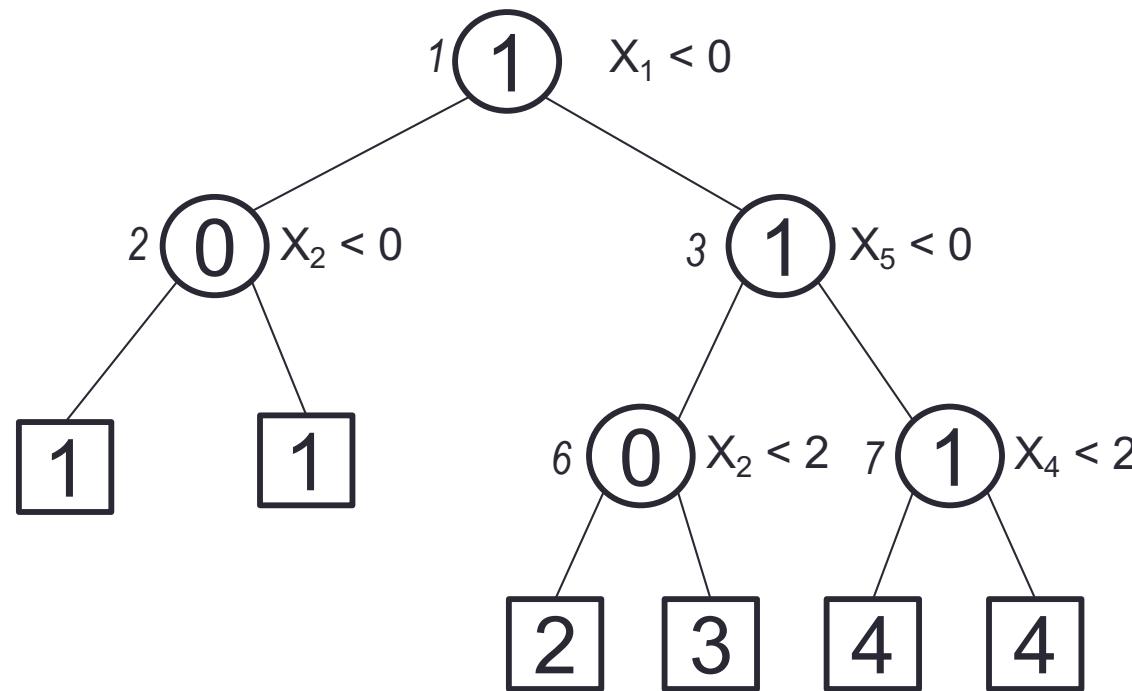
- **ERA.** Conjunto de datos que incluye los atributos de los candidatos hipotéticos para un trabajo, y las evaluaciones de los estudiantes de Administración de Empresas con respecto a sus calificaciones.
- **ESL.** Conjunto de datos de solicitantes en un proceso industrial abierto y valoraciones de expertos sobre sus calificaciones
- **LEV.** Conjunto de datos de los conferenciantes hipotéticos, y opiniones de estudiantes de Administración de Empresas sobre sus calificaciones de enseñanza.
- **SWD.** Contiene evaluaciones del mundo real de trabajadores sociales que califican el riesgo que enfrentan los niños si se quedaban con sus familias en el hogar.

Árboles de Decisión Monotónicos



ÁRBOL DE DECISIÓN NO
MONOTÓNICO

Árboles de Decisión Monotónicos



ÁRBOL DE DECISIÓN MONOTÓNICO

Medidas de Rendimiento (I)

$$\text{Accuracy } p_a(\mathcal{Y}, \hat{\mathcal{Y}}) = \frac{1}{N} \sum_{i=1}^N 1[\hat{y}_i = y_i]$$

$$\text{Mean Absolute Error } MAE(\mathcal{Y}, \hat{\mathcal{Y}}) = \frac{1}{N} \sum_{i=1}^N |\hat{y}_i - y_i|$$

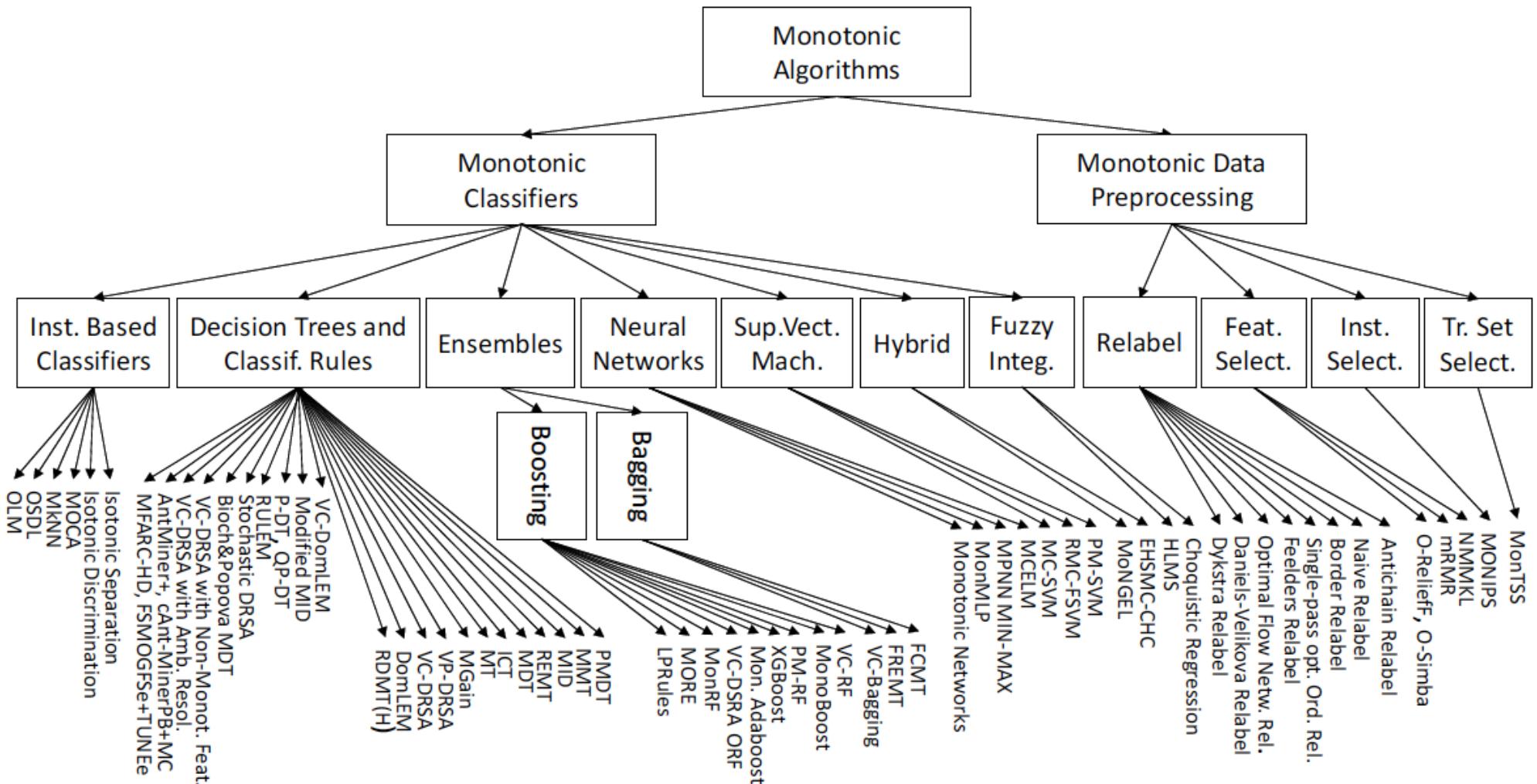
Medidas de Rendimiento (II)

$$NMI1 = \frac{1}{N^2 - N} \sum_{i=1}^N \sum_{j=1}^N \mathbf{1}[(\mathbf{x}_i \preceq \mathbf{x}_j \wedge y_i > y_j) \vee (\mathbf{x}_i \succeq \mathbf{x}_j \wedge y_i < y_j)]$$

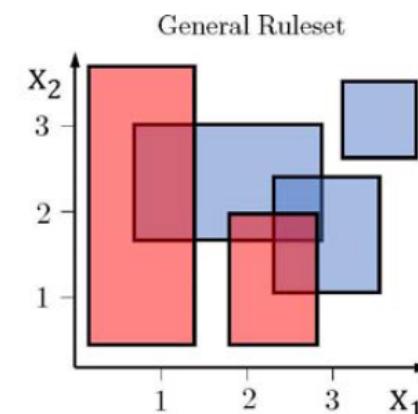
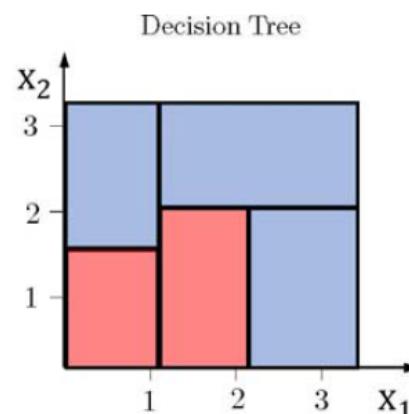
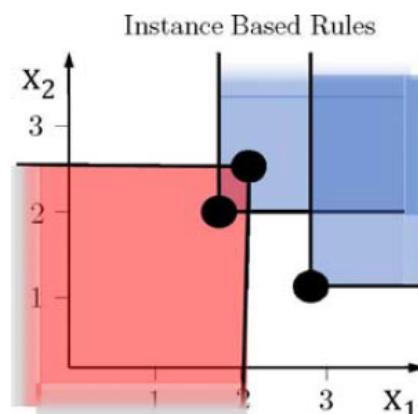
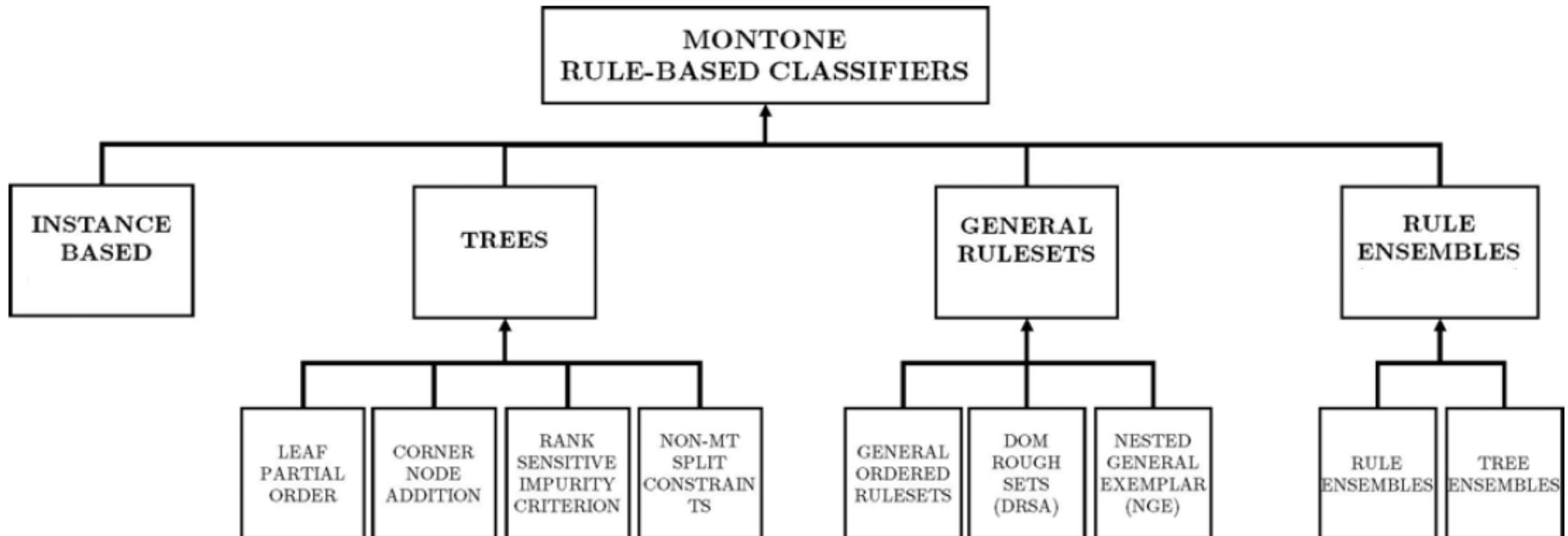
$$NMI2 = \frac{1}{NumComparablePairs} \sum_{i=1}^N \sum_{j=1}^N \mathbf{1}[(\mathbf{x}_i \preceq \mathbf{x}_j \wedge y_i > y_j) \vee (\mathbf{x}_i \succeq \mathbf{x}_j \wedge y_i < y_j)]$$

$$NMI3 = \frac{1}{N} \sum_{i=1}^N \mathbf{1}[\exists \mathbf{x}_j \text{ s.t. } (\mathbf{x}_i \preceq \mathbf{x}_j \wedge y_i > y_j) \vee (\mathbf{x}_i \succeq \mathbf{x}_j \wedge y_i < y_j)]$$

CLASIFICACIÓN MONOTÓNICA: Taxonomía



CLASIFICACIÓN MONOTÓNICA: Algoritmos basados en reglas e instancias

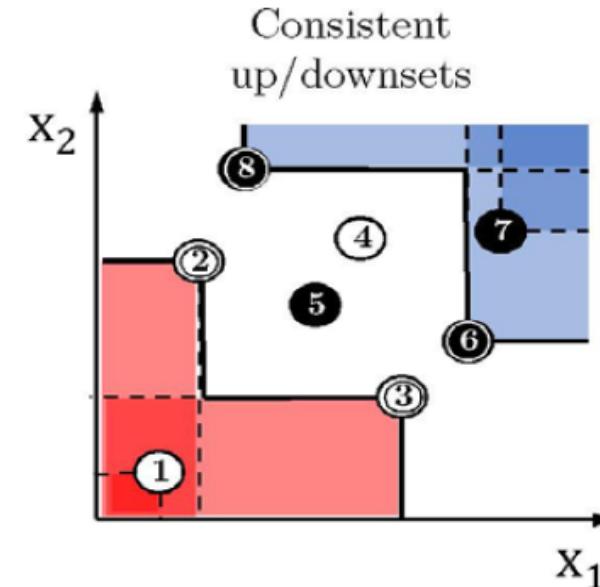
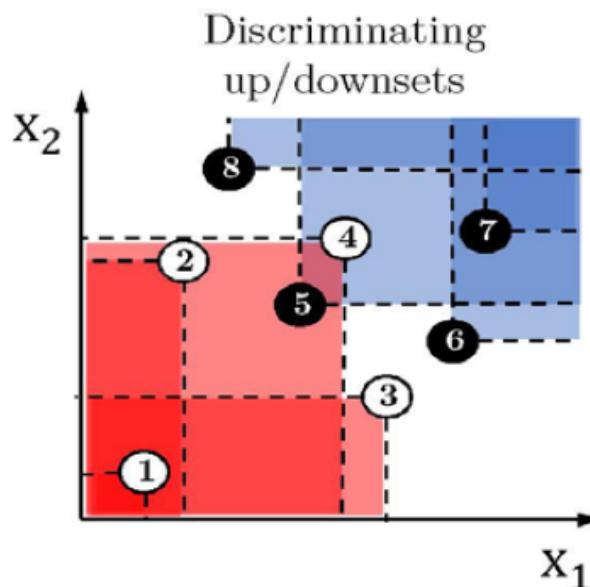


Métodos basados en Instancias

- Definimos el upset $[x]^\uparrow$ y downset $[x]_\downarrow$ (conjuntos dominantes/dominados)

$$[x]^\uparrow = \{\mathbf{x}_i \mid \mathbf{x} \preceq \mathbf{x}_i, i = 1..N\}$$

$$[x]_\downarrow = \{\mathbf{x}_i \mid \mathbf{x}_i \preceq \mathbf{x}, i = 1..N\}$$



- Dos estrategias para convertir en regla de clasificación:
 - Usar los propios puntos y el orden parcial
 - DRSA: Dominance Rough Sets Approach

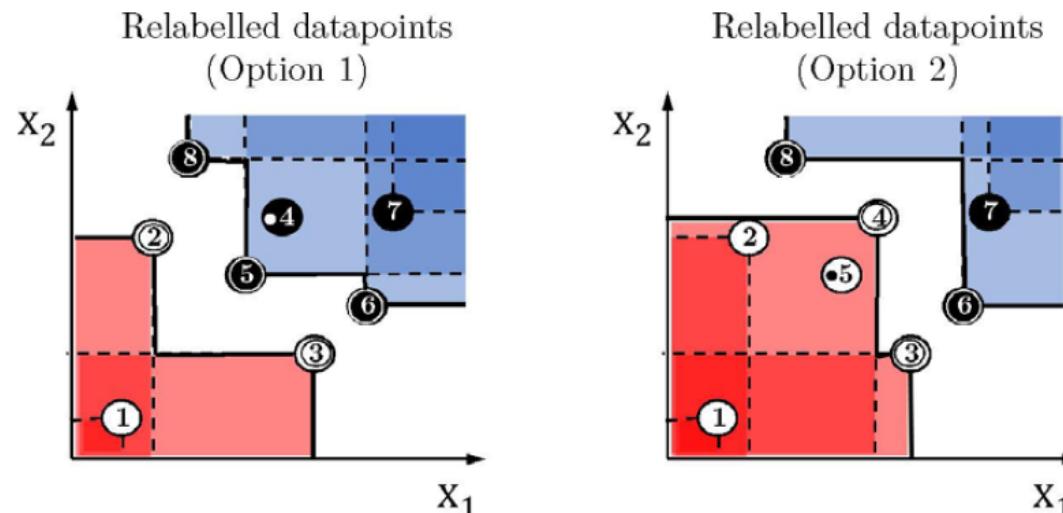
Reetiquetado

- Siempre habrá zonas ambiguas en conjuntos no monótonos.
- La consistencia monótona se puede conseguir o mejorar con el reetiquetado. Problema optimización:

$$\min_{y_i^*} \sum_{i=1}^N L(y_i^*, y_i)$$

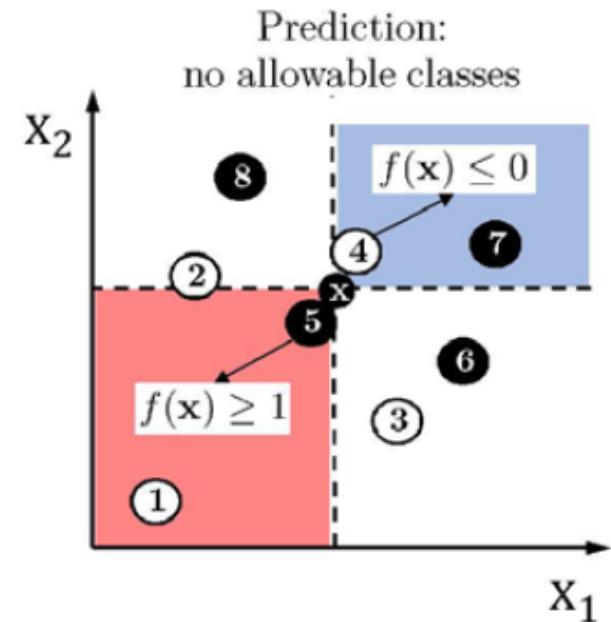
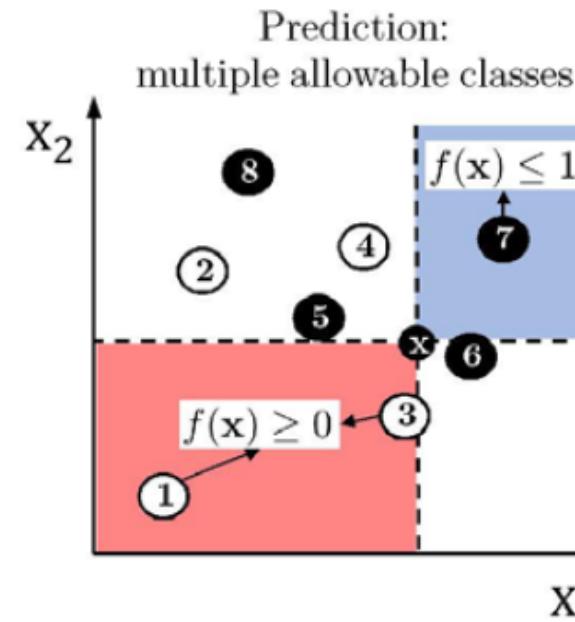
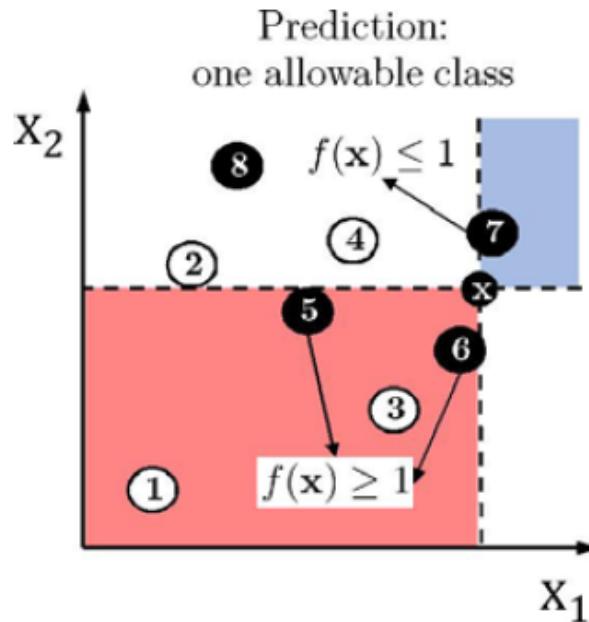
subject to: $\mathbf{x}_i \preceq \mathbf{x}_j \implies y_i^* \leq y_j^*, \quad i, j \in 1..N$

- Funciones de pérdida L comunes: 0/1 loss, L1 (absoluta), L2 (cuadrada)



Métodos basados en instancias: Escenarios

- Predicción no ambigua – Predicción ambigua – Predicción imposible



Ordinal Learning Model (OLM)
Ben-David 1992

Downset

Ordinal Stochastic Dominance Learner
(OSDL)
Cao-Van 2003, Lievens et al. 2008

Upset, Downset

Monotone kNN
Duivesteijn & Feelders 2008

Upset, Downset

Ordinal Learning Model (OLM)

- Algoritmo simple que aprende conceptos ordinales eliminando inconsistencias pareadas entre ejemplos.
- Los conceptos que genera se pueden ver como reglas.
- Durante la fase de aprendizaje, cada ejemplo se verifica con cada regla de la base de reglas, que comienza vacía inicialmente.
- Si un ejemplo supera el test de inconsistencia con todas las reglas de la base, se añade como regla.

Ordinal Learning Model (OLM)

- La base de reglas se mantiene monotónica en todo momento.
- La clasificación se hace de forma conservativa:
 - Todas las reglas se comprueban en orden decreciente de los valores de clases contra el vector de atributos del ejemplo, y el ejemplo se clasifica con la clase de la primera regla que lo cubre.
 - Si dicha regla no existe, el ejemplo se asigna a la clase más baja posible.

Ordinal Learning Model (OLM)

Phase 1: Order the examples in descending order of output;

WHILE there are unchecked examples

{

Mark the new example, E ;

Mark each example which has similar input;

Replace the marked examples with one example which has the same input
and the average output of the marked examples;

}

Phase 2: WHILE there are unchecked examples

{

Get a new example, E ;

WHILE there are unchecked rules in the rule-base

{

IF E is either redundant or inconsistent with a rule, R , in the rule-base

IF $E_I < R_I$ and E is consistent and irredundant with respect to the rest
of the rules currently in the rule-base

Reject R ;

Add E to the rule-base;

ELSE

Reject E ;

}

IF E has not been rejected so far

Add E to the rule-base;

}

Ordinal Learning Model (OLM)

WHILE no classification took place yet

{

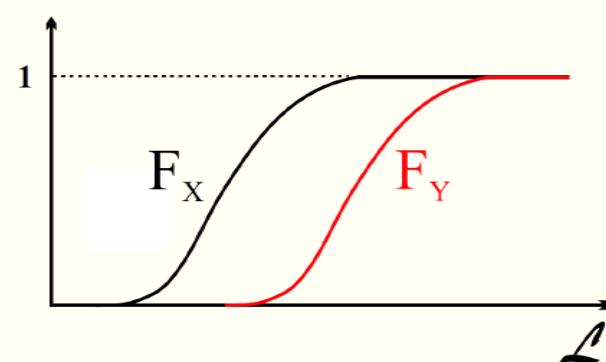
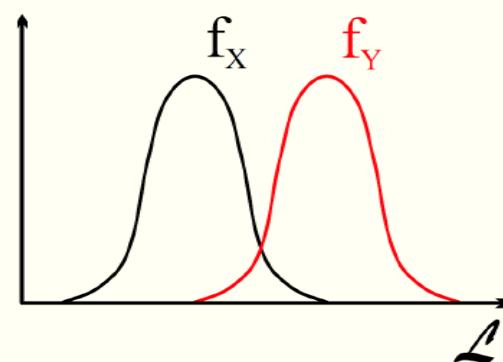
 Compare the object to be classified, F , with the current rule, R , in the CISE;
 IF $F_I > R_I$, classify F as R_O ;

}

OTHERWISE classify F as the average output of the closest input profiles;

Ordinal Stochastic Dominance Learner (OSDL)

- Ordinal Stochastic Dominance Learner (OSDL) es un clasificador basado en instancias que asegura la monotonía entrenando con conjuntos monotónicos.
 - Basado en Funciones de distribución acumulada (CDF) para cada instancia i .
 - Manejan violaciones por igualdad que llaman dudas.
 - Traslada el concepto de monotonía en clases a dominancia en CDFs.
 - Parecido a Mon-kNN, restringe la clasificación de i en un rango de clases validas que preservan la monotonía, pero en CDFs.
 - CDF para nuevas instancias se estiman con interpolación dentro del rango valido y su clase como la mediana del CDF.



Ordinal Stochastic Dominance Learner (OSDL)

Estimación de CDF cada instancia de entrenamiento

- Para la estimación de CDF, usa las ‘dudas’ en el conjunto, instancias repetidas con clases diferentes. Por ejemplo, para la instancia i:

Clases	Repeticiones
L ₁	0
L ₂	2
L ₃	5
L ₄	3

$$\text{PMF}_i = (0 ; 0.2 ; 0.5 ; 0.3) \\ \text{CDF}_i = (0 ; 0.2 ; 0.7 ; 1)$$

Una instancia j sin repeticiones de la clase 2 tendrá una
 $\text{CDF}_j = (0 ; 1.0 ; 1.0 ; 1.0)$

- El valor de CDF para la instancia x y la clase l corresponde al número de instancias iguales cuya clase es menor o igual que l dividido entre el total de instancias iguales a x .

$$\hat{F}^*(\mathbf{x}, \ell) = \frac{|\{a \in \mathcal{S} \mid \mathbf{a} = \mathbf{x} \wedge d(a) \leq_{\mathcal{L}} \ell\}|}{|\{a \in \mathcal{S} \mid \mathbf{a} = \mathbf{x}\}|}$$

Ordinal Stochastic Dominance Learner (OSDL)

Monotonía por Dominancia Estocástica 1ord.

- Para satisfacer las restricciones de monotonidad tenemos:
 $x \leq y \Rightarrow C(x) \leq C(y)$
- Para satisfacerlas con CDFs, x tiene que ser débilmente dominado (FSD) por y :

$$x \leq y \Rightarrow f_x \leq_{FSD} f_y$$

- x es dominado por y si todo valor de CDF_x es mayor/igual que de CDF_y:
 $f_x \leq_{FSD} f_y \Rightarrow (\forall l)(F_x(l) \geq F_y(l))$

- Si y domina a x , la mediana de y será mayor o igual que la de x :
 $f_x \leq_{FSD} f_y \Rightarrow med(f_x) \leq med(f_y)$

- Esto demuestra la validez de los CDFs para la monotonía.

Ordinal Stochastic Dominance Learner (OSDL)

De PMF/CDF a clase real mediante la mediana

- ¿Si seleccionamos la probabilidad máxima como clase?

$$f_x = (0.1 ; 0.3 ; 0.4 ; 0.2)$$

$x \leq y \Rightarrow f_x \leq_{FSD} f_y$; y domina a x , sin embargo, eligiendo la clase con la probabilidad máxima tenemos:

$$f_y = (0.0 ; 0.4 ; 0.3 ; 0.3)$$

$$\text{argmax}(f_x) = l_3 > l_2 = \text{argmax}(f_y)$$

El máximo no es válido.

- Medidas de centralidad:

- La media necesita de una escala numérica.
- La mediana se define como el valor ν que tiene una probabilidad acumulada mayor/igual que $\frac{1}{2}$ de ser menor/igual que ν y de ser mayor/igual que ν . En discreto, es un intervalo $[l_m, l_M]$.

$$\text{med}(f_x) = l_3 = l_3 = \text{med}(f_y)$$

$$\ell_m = \inf\{\ell \in \mathcal{L} \mid \text{Prob}\{X \leq \ell\} \geq 1/2\}$$

$$f_z = (0.2 ; 0.3 ; 0 ; 0.3 ; 0.2)$$
$$\text{med}(f_z) = [2,4] = 3$$

$$\ell_M = \sup\{\ell \in \mathcal{L} \mid \text{Prob}\{X \geq \ell\} \geq 1/2\}.$$

Ordinal Stochastic Dominance Learner (OSDL)

CDF para nuevas instancias I

- Restringe la clasificación con CDFs mínimas F_{min} y máximas F_{max} llamados extensiones. Cualquier valor entre los intervalos definidos por dichas extensiones, respeta la monotonía.
- $F_{min}(x, \cdot)$ se obtiene con los valores mínimos para cada clase l de los F de las instancias y menores/iguales que x . Análogamente, $F_{max}(x, \cdot)$, con los máximos $F(y, l)$ de $y \geq x$.

$$F_{min}(x, \cdot) = \min\{F(y, l) | y \in S \wedge y \leq x\}$$

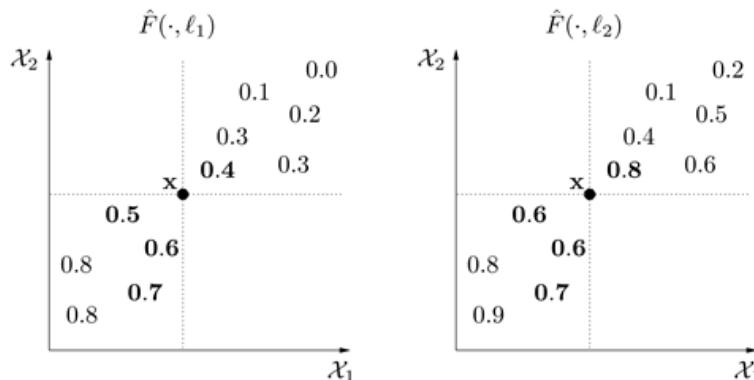
$$F_{max}(x, \cdot) = \max\{F(y, l) | y \in S \wedge y \geq x\}$$

- Si $\{y \in S \wedge y \leq x\} = \emptyset$, se asigna 1 a la clase más pequeña. Para $F_{max}(x, \cdot)$, se asigna 1 a la mayor en el caso análogo.

Ordinal Stochastic Dominance Learner (OSDL)

CDF para nuevas instancias II

- Si el conjunto es monotónico tenemos:
 - $F(y, \cdot) \leq_{FSD} F_{min}(x, \cdot) \mid y \leq x$
 - $F_{max}(x, \cdot) \leq_{FSD} F(y, l) \mid y \geq x$
 - Toda extensión monotónica $F(x, \cdot)$ cumple $F_{min}(x, \cdot) \leq_{FSD} F(x, \cdot) \leq_{FSD} F_{max}(x, \cdot)$
- Con esta situación, cualquier interpolación entre $F_{min}(x, \cdot)$ y $F_{max}(x, \cdot)$ es válida y monótona como clasificación de la nueva instancia x .
- Si el conjunto no es monotónico con una violación entre x e y , cada instancia z entre x e y tendrá al menos un l , tal que $F_{min}(z, l) < F_{max}(z, l)$.



$$\begin{aligned}
 \hat{F}(\cdot, \ell_1) & \quad \hat{F}(\cdot, \ell_2) & F_{min}(x, \ell_1) &= \min\{0.8, 0.8, 0.7, 0.6, 0.5\} = 0.5 & F_{min}(x, \cdot) &= \{0.5, 0.6\} \\
 & & F_{min}(x, \ell_2) &= \min\{0.9, 0.8, 0.7, 0.6, 0.6\} = 0.6 & & \\
 F_{max}(x, \ell_1) &= \max\{0.4, 0.3, 0.3, 0.2, 0.1, 0.0\} = 0.4 & & & F_{max}(x, \cdot) &= \{0.4, 0.8\} \\
 F_{max}(x, \ell_2) &= \max\{0.8, 0.6, 0.5, 0.4, 0.2, 0.1\} = 0.8 & & & &
 \end{aligned}$$

Ordinal Stochastic Dominance Learner (OSDL)

Esquemas de interpolación: OSDL

- La interpolación más básica es dar un punto entre los intervalos válidos con ayuda de un valor prefijado S de $[0,1]$.

$$F_{OSDL}(x, l) = S * F_{min}(x, l) + (1 - S) * F_{max}(x, l)$$

- Por defecto, se recomienda $S = 0.5$, sin embargo, podemos aumentarlo o disminuirlo, según cómo de pesimista o optimista quedamos que sea dicha predicción.
- Esta interpolación preserva la monotonía, siempre que el conjunto lo sea.

Monotonic k-Nearest Neighbors

- Dos etapas:
 - Se monotoniza el conjunto de entrenamiento mediante una técnica óptima de reetiquetado, realizando el menor número de reetiquetados posible.
 - Se modifica la regla de predicción de etiquetas de clase de los nuevos ejemplos para que no ocurran violaciones de la restricción de monotonicidad.

Monotonic k-Nearest Neighbors

Primera Etapa

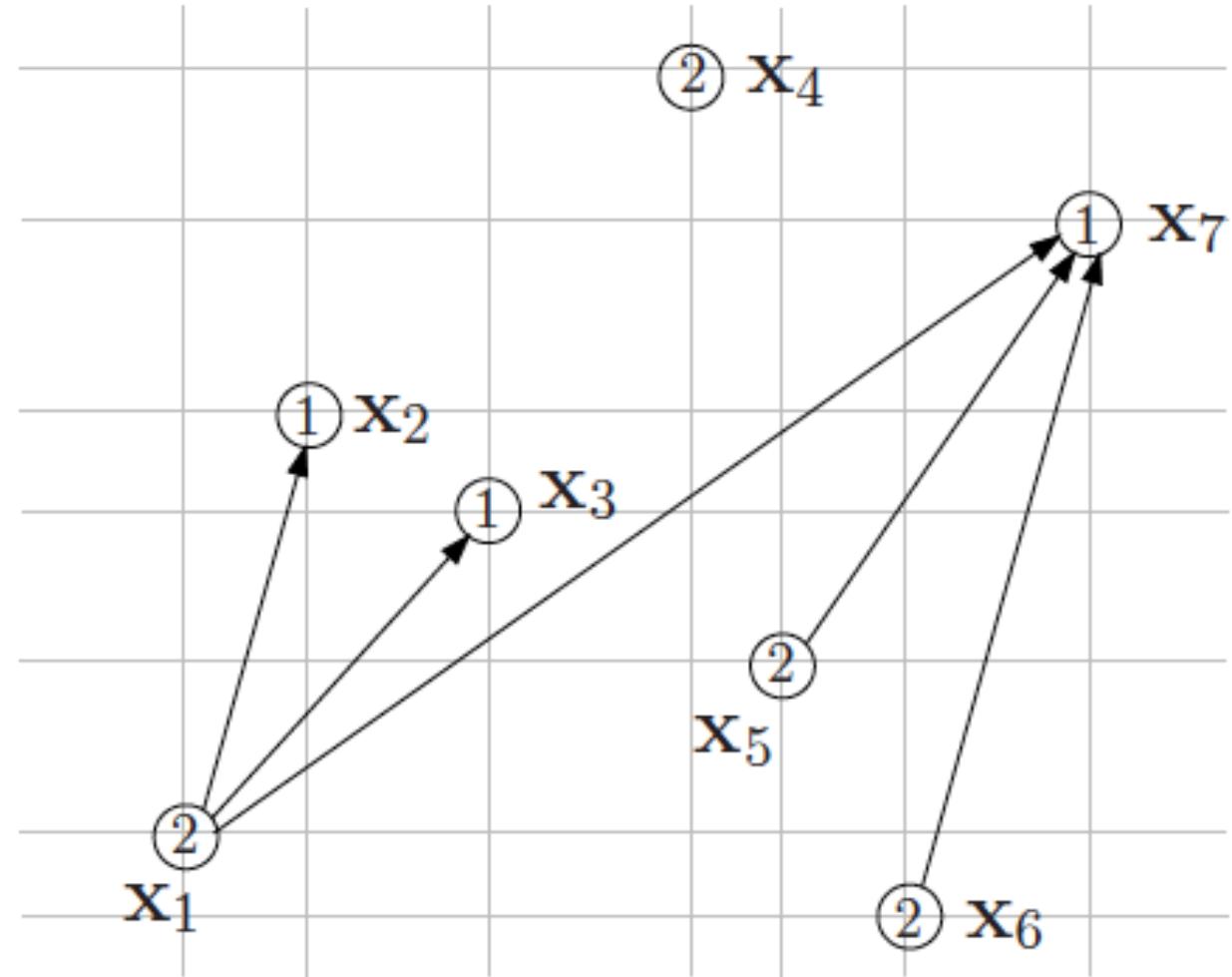


Fig. 1. Example Monotonicity Violation Graph

Monotonic k-Nearest Neighbors

Primera Etapa

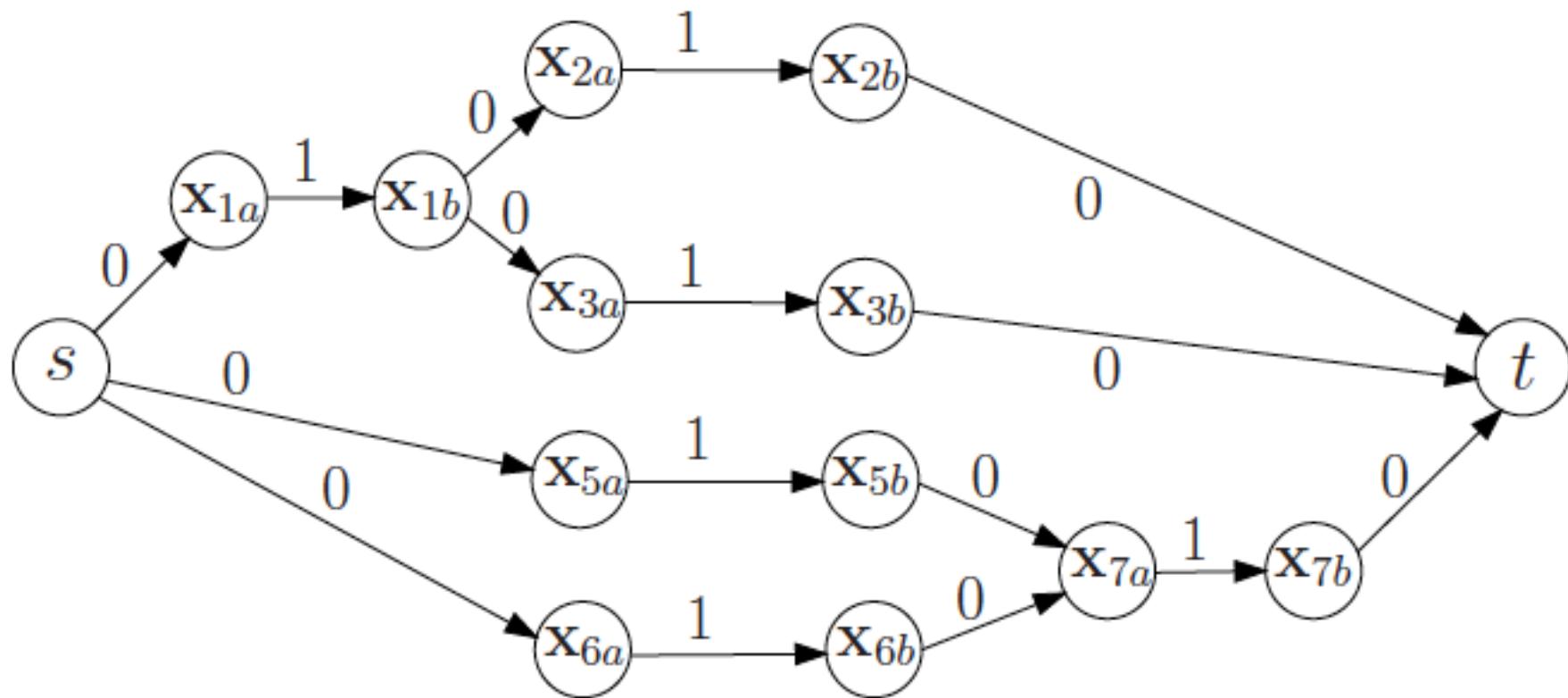


Fig. 2. Transportation network based on the Monotonicity Violation Graph in Figure 1

Monotonic k-Nearest Neighbors

Segunda Etapa:

Para satisfacer las restricciones monotónicas, el etiqueta de clase de un nuevo ejemplo x_0 se restringe a caer en el intervalo $[y_{\min}, y_{\max}]$, donde

$$y_{\min} = \max\{y | (\mathbf{x}, y) \in D \wedge \mathbf{x} \leq \mathbf{x}_0\},$$

$$y_{\max} = \min\{y | (\mathbf{x}, y) \in D \wedge \mathbf{x}_0 \leq \mathbf{x}\},$$

D es un conjunto de entrenamiento reetiquetado.

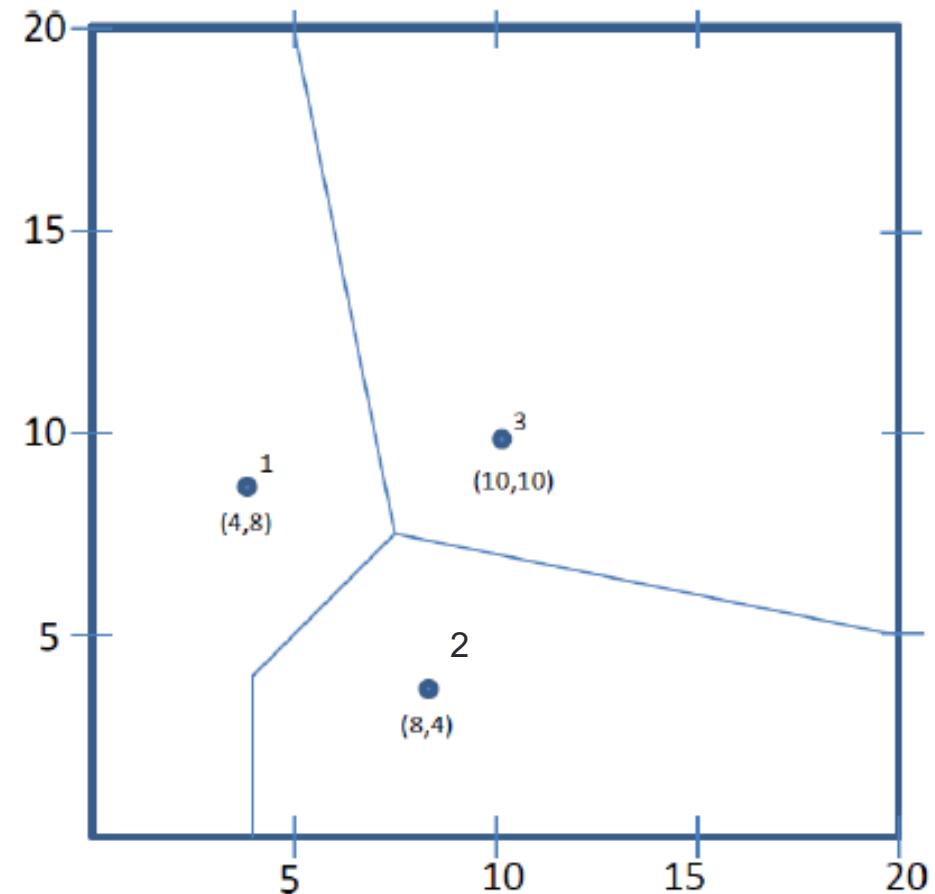
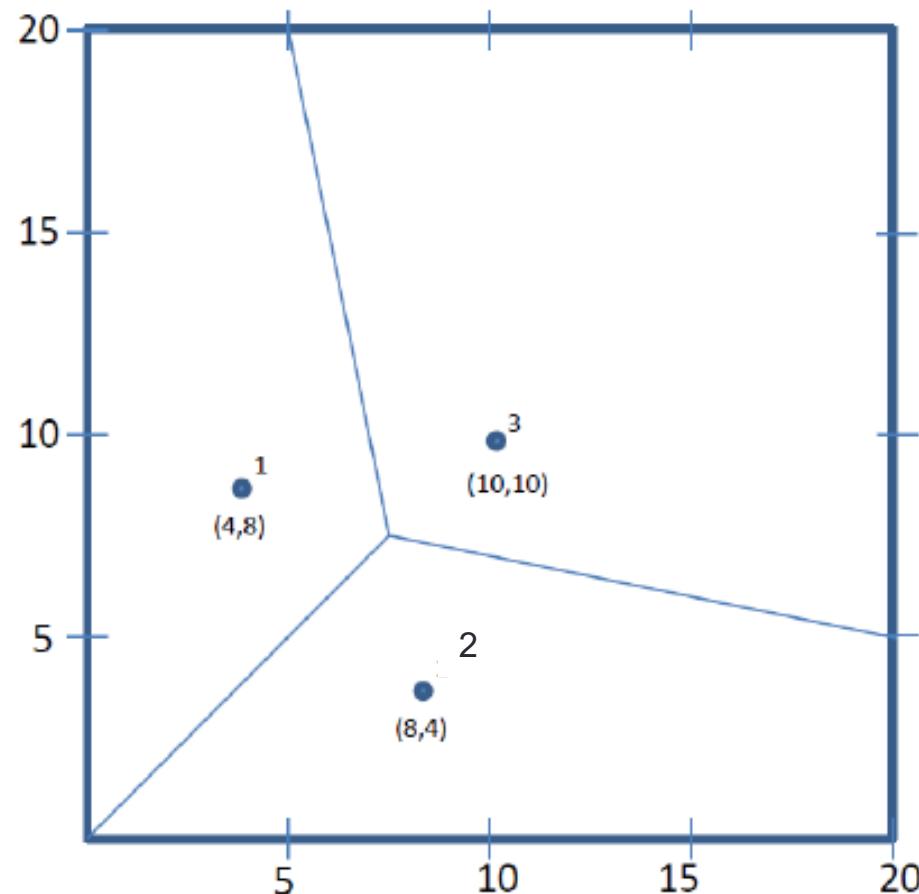
Monotonic k-Nearest Neighbors

Segunda Etapa, Variantes:

- Coger los k vecinos cercanos de x_0 de D y predecir la etiqueta desde $[y_{\min}, y_{\max}]$ que más frecuente aparece entre esos k vecinos. Si ninguna de las k etiquetas se permite, escoger una al azar entre $[y_{\min}, y_{\max}]$.
- Coger los k vecinos cercanos de x_0 de D y con etiquetas desde $[y_{\min}, y_{\max}]$ y predecir la clase con voto mayoritario.

Monotonic k-Nearest Neighbors

Diferencia entre versión monotónica y no monotónica:



Monotone Support Vector Machine

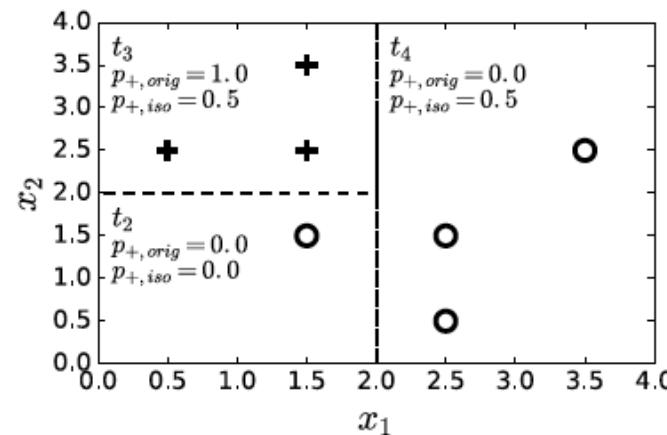
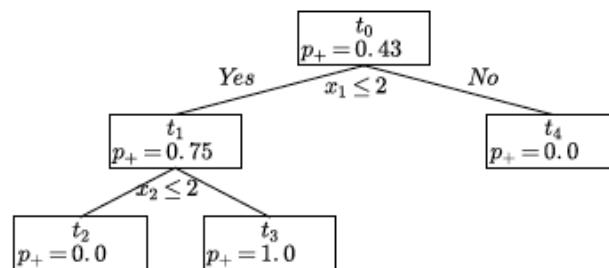
- Las SVM existentes añaden restricciones al problema original de optimización para requerir que la solución sea monótona para un conjunto seleccionado de puntos.

$$\mathbf{w}^T \psi(\tilde{\mathbf{x}_k}) \geq \mathbf{w}^T \psi(\tilde{\mathbf{x}}_k), \quad k = 1, \dots, K$$

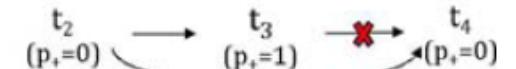
- Las K restricciones se añaden mediante dos procedimientos:
 - CJ1: Seleccionando aleatoriamente $m=5$ puntos de entrenamiento para cada K .
 - CJ2: Particionando cada atributo monótono en K particiones entre su valor mínimo y máximo del training set. Usar un k-medias por partición para seleccionar puntos.

CLASIFICACIÓN MONOTÓNICA: Árboles de Decisión

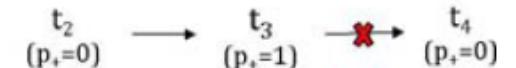
- Cuatro familias:
 - **Leaf Partial Orders:** El orden parcial entre hojas se entiende cuando una hoja t_j contiene subregiones que dominan puntos en subregiones de la hoja t_i . $(t_i \preceq_L t_j)$



Transitive closure:



Transitive reduction:



- Ejemplos de árboles: MID (derivados), ICT con relabeling ($p_{+,iso}$) y poda.

Monotonic Induction of Decision Tree (MID)

Define una métrica que considera tanto la precisión como la monotonía del árbol.

PASO 1

Índice de monotonía

$$W = \sum_{i=1}^k \sum_{j=1}^k m_{ij}$$

Valor máximo de W es $(k^2 - k)$.

Así el Índice de monotonía se calcula:

$$I_{árbol} = \frac{W_{árbol}}{k_{árbol}^2 - k_{árbol}}$$

PASO 2 Orden de ambigüedad

$$A_{árbol} = \begin{cases} 0 & \text{si } I_{árbol} = 0 \\ -(\log_2 I_{árbol})^{-1} & \text{en otro caso} \end{cases}$$

PASO 3 Ambigüedad total

$$T_{árbol} = E_{árbol} + A_{árbol}$$

Donde $E_{árbol}$ representa la **ganancia de información**. En este paquete se ha cambiando por **ratio de ganancia**.

$$T_{árbol} = E_{árbol} + R \times A_{árbol}$$

Monotonic Induction of Decision Tree (MID)

- **DEFINICIÓN 1:** Sea $X = (x_1, x_2, \dots, x_n)$ e $Y = (y_1, y_2, \dots, y_n)$ dos instancias del mismo problema, con n atributos. Todos los valores de los atributos son ordinales o numéricos. Un orden entre X e Y se define como:

$$X = Y \quad \text{if } x_i = y_i \quad \forall i = 1, 2, \dots, n$$

$$X > Y \quad \text{if } x_i > y_i \quad \forall i = 1, 2, \dots, n$$

$$X \geq Y \quad \text{if } x_i > y_i \quad \text{OR} \quad x_i = y_i \quad \forall i = 1, 2, \dots, n$$

$$X < Y \quad \text{if } x_i < y_i \quad \forall i = 1, 2, \dots, n$$

$$X \leq Y \quad \text{if } x_i < y_i \quad \text{OR} \quad x_i = y_i \quad \forall i = 1, 2, \dots, n$$

Monotonic Induction of Decision Tree (MID)

- **DEFINICIÓN 2:** Sea $X = (X, C_x)$ e $Y = (Y, C_y)$ representando dos parejas clase-atributos. Las clases de X e Y se denotan por C_x y C_y respectivamente. (X, C_x) y (Y, C_y) son no-monotónicos con respecto a cada uno si:

$$X \leq Y \wedge C_x > C_y \quad \text{OR}$$

$$X \geq Y \wedge C_x < C_y \quad \text{OR}$$

$$X = Y \wedge C_x \neq C_y$$

- **DEFINICIÓN 3:** Dos parejas clase-atributos (X, C_x) e (Y, C_y) son monotónicas con respecto al otro si no cumple cualquiera de las condiciones de la definición anterior.

Monotonic Induction of Decision Tree (MID)

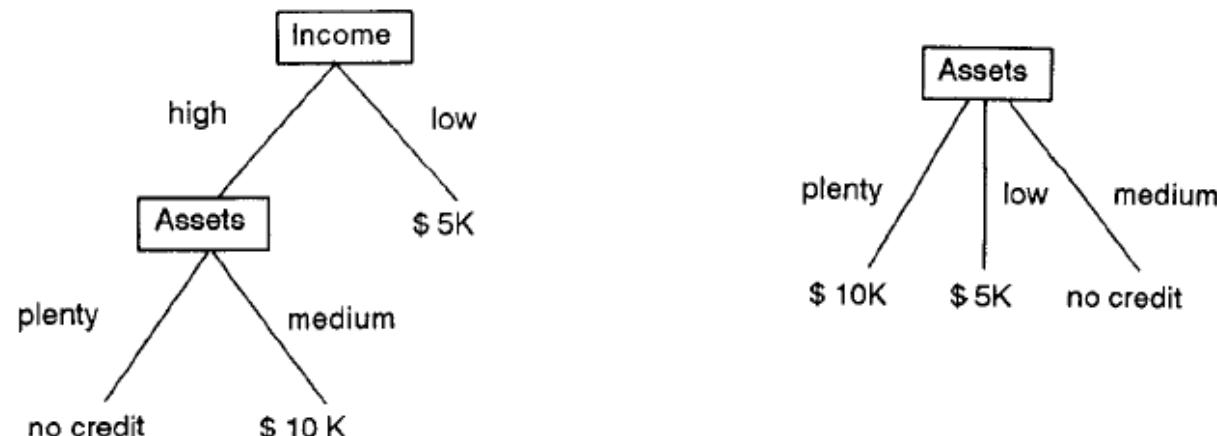
- **DEFINICIÓN 4:** Sean (P, C_p) y (Q, C_q) dos caminos atributo-test/respuesta-nodo en el mismo árbol de decisión, donde P y Q son atributo-test y C_p y C_q son respuesta-nodos. Ambos caminos son monotónicos con respecto al otro si no cumple cualquiera de las condiciones de la definición 2.
- **DEFINICIÓN 5:** Un árbol de decisión es *monotónico* si todas las parejas atributo-test/respuesta-nodo son monotónicas con respecto a cada otra.

Monotonic Induction of Decision Tree (MID)

- Conjunto monotónico

Example	Income	Assets	Credit history	Class
#1	high	plenty	good	\$ 10K
#2	high	low	bad	\$ 5K
#3	low	medium	bad	no credit

- Si aplicamos ID3:
 - $E(\text{income}) = 0.667 (2/3 * 1 + 1/3 * 0)$
 - $E(\text{assets}) = 0$



Monotonic Induction of Decision Tree (MID)

Construyendo árboles monotónicos precisos

- **DEFINICIÓN 6:** Un índice de no-monotonicidad es el ratio entre el número real de parejas de ramas del árbol de decisión, y el máximo número de ramas que podrían ser no-monotónicas con respecto a cada otra en el mismo árbol.
- Para encontrar este índice en un árbol de k ramas, construimos una matrix M , $k \times k$ de no-monotonicidad, simétrica, donde el elemento m_{ij} vale 1 si la rama i es no-monotónica con respecto a la rama j , y 0 si viceversa. La suma de las M 's entradas se denota por W .

$$W = \sum_{i=1}^k \sum_{j=1}^k m_{ij}$$

Monotonic Induction of Decision Tree (MID)

Construyendo árboles monotónicos precisos

- Como máximo, $(k^2 - k)$ entradas de M pueden ser etiquetadas como no-monotónicas. El índice de no-monotonicidad de un árbol de decisión con los atributos test a_1, a_2, \dots, a_v , se define como:

$$I_{a_1, a_2, \dots, a_v} = \frac{W_{a_1, a_2, \dots, a_v}}{k_{a_1, a_2, \dots, a_v}^2 - k_{a_1, a_2, \dots, a_v}}$$

Monotonic Induction of Decision Tree (MID)

Construyendo árboles monotónicos precisos

- **DEFINICIÓN 7:** El *score de ambigüedad de orden* de un árbol de decisión se define en términos del índice de no-monotonicidad.

$$A_{a_1, a_2, \dots, a_v} = \begin{cases} 0 & \text{if } I_{a_1, a_2, \dots, a_v} = 0 \\ -(\log_2 I_{a_1, a_2, \dots, a_v})^{-1} & \text{otherwise} \end{cases}$$

- **DEFINICIÓN 8:** El *score total de ambigüedad* es la suma de la entropía, usada por ID3 o C4.5, y el score de ambigüedad de orden.

$$T_{a_1, a_2, \dots, a_v} = E_{a_1, a_2, \dots, a_v} + A_{a_1, a_2, \dots, a_v}$$

Monotonic Induction of Decision Tree (MID)

Construyendo árboles monotónicos precisos

- Una forma efectiva de expresar equilibrios entre la entropía y la monotonidad se puede conseguir introduciendo un parámetro adicional al score de ambigüedad total.

$$T_{a_1, a_2, \dots, a_v} = E_{a_1, a_2, \dots, a_v} + R A_{a_1, a_2, \dots, a_v}$$

- El parámetro R expresa la relativa importancia de la monotonidad relativa al acierto inductivo en un problema dado. Este parámetro podría ajustarse mediante varias iteraciones del algoritmo sobre los mismos datos.

Monotonic Induction of Decision Tree (MID)

Construyendo árboles monotónicos precisos

- Para ilustrar cómo funciona, aplicamos ID3 a los datos anteriores. Usamos $R = 2$:

$$E_{\text{income}} = 0.667 \text{ bits}$$

$$I_{\text{income}} = A_{\text{income}} = 0$$

$$T_{\text{income}} = 0.667 (0.667 + 2 * 0)$$

$$E_{\text{assets}} = 0 \text{ bits}$$

$$I_{\text{assets}} = 0.333 (2/(3^2 - 3))$$

$$A_{\text{assets}} = 0.630 (-\log_2 0.333)^{-1}$$

$$T_{\text{assets}} = 1.260 (0 + 2 * 0.630)$$

$$E_{\text{credit history}} = 0.667 \text{ bits}$$

$$I_{\text{credit history}} = A_{\text{credit history}} = 0$$

$$T_{\text{credit history}} = 0.667.$$

Monotonic Induction of Decision Tree (MID)

Construyendo árboles monotónicos precisos

- Los scores de ambigüedad total para *income* y *credit history* son los más bajos, y asumimos que el atributo *income* se selecciona en la primera partición. El score de ambigüedad total de *income + assets* se comprueba:

$$E_{\text{income+assets}} = 0 \text{ bits}$$

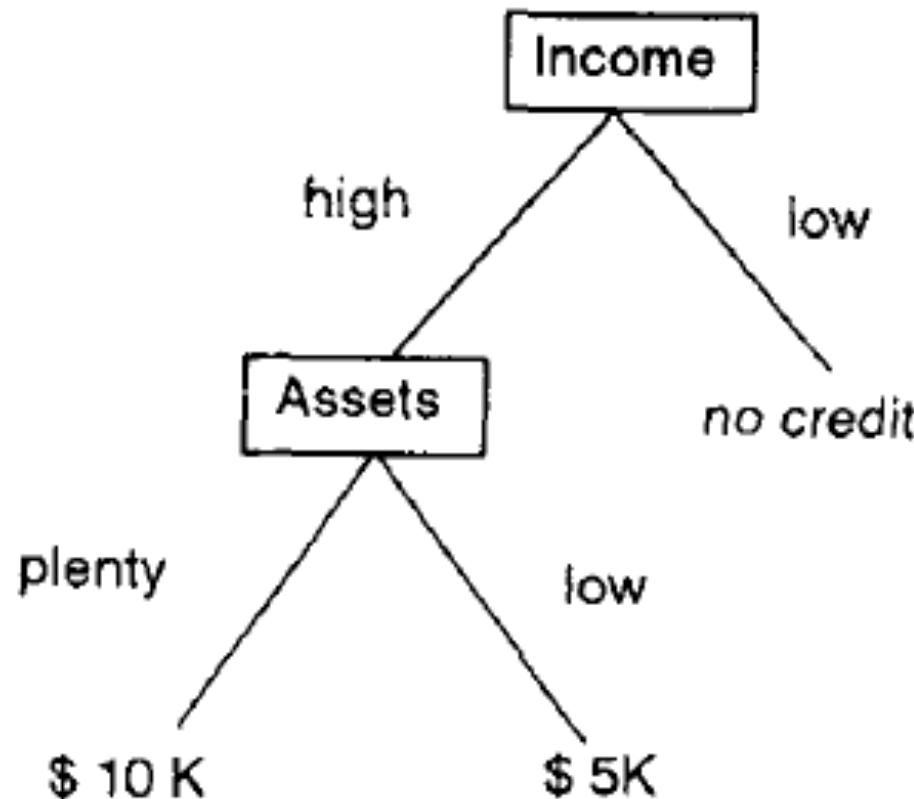
$$I_{\text{income+assets}} = A_{\text{income+assets}} = 0$$

$$T_{\text{income+assets}} = 0.$$

- Ahora, el árbol obtenido es monotónico.

Monotonic Induction of Decision Tree (MID)

Construyendo árboles monotónicos precisos



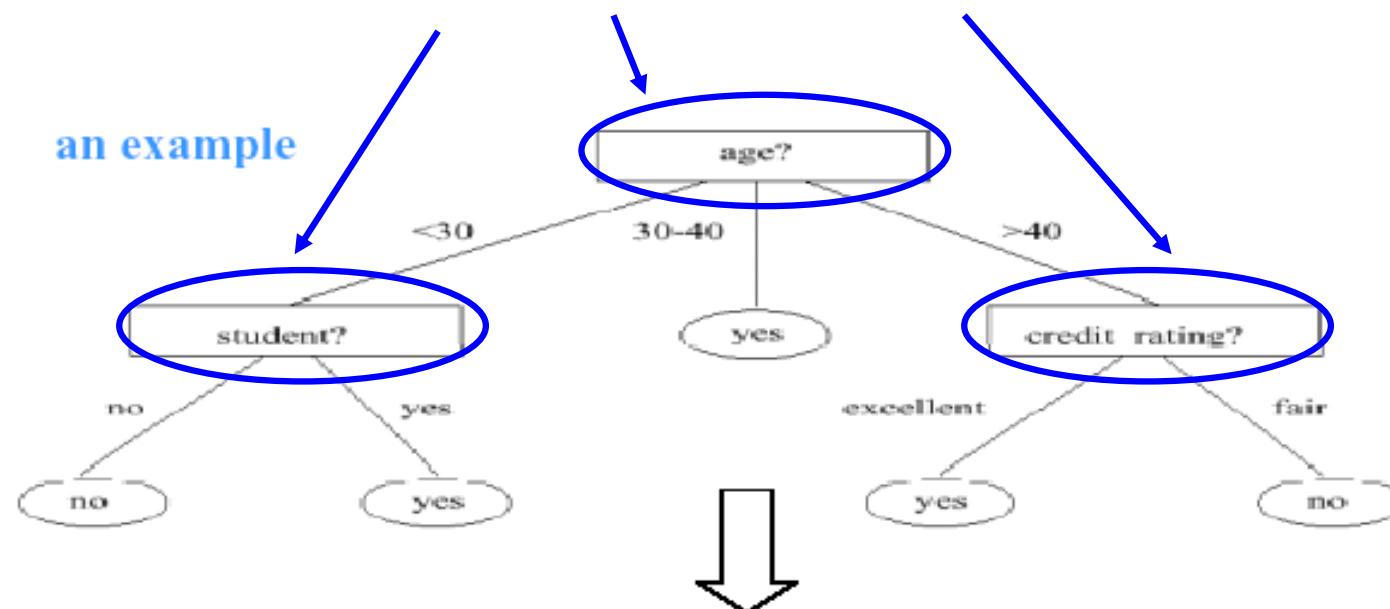
CLASIFICACIÓN MONOTÓNICA: Árboles de Decisión

- **Corner Point Addition:**

- Después de decidir un split, cada nodo hoja tiene dos nuevos puntos añadidos, el upper corner **a** (máximo) y el lower corner **b** (mínimo).
- Las clases de estos puntos artificiales se escogen usando el conjunto de entrenamiento, usando instance-based learning.
- Si el conjunto es monótono consistente, el resultado es monótono global.
- El ruido se trata con relabeling.
- No permite monotonicidad parcial y necesita añadir nuevos puntos.
- Positive Decision Tree ó REMT.

Rank Entropy Monotonic Tree (REMT)

La pregunta más importante en la construcción de árboles de decisión es diseñar una medida para calcular la calidad de los atributos, y seleccionar el mejor para hacer las particiones.



IF *age* = “<30” AND *student* = no

IF *age* = “<30” AND *student* = yes

IF *age* = “30-40”

IF *age* = “>40” AND *credit_rating* = excellent

IF *age* = “>40” AND *credit_rating* = fair

THEN *buys_computer* = no

THEN *buys_computer* = yes

THEN *buys_computer* = yes

THEN *buys_computer* = yes

THEN *buys_computer* = no

Rank Entropy Monotonic Tree (REMT)

Métrica RMI

$$RMI^{\leq} = -\frac{1}{|N|} \sum_{i=1}^{|N|} \left(\log \frac{|[x_i]_{a_j}^{\leq} \times [x_i]_c^{\leq}|}{N \times |[x_i]_{a_j}^{\leq} \cap [x_i]_c^{\leq}|} \right)$$

DATASET LEV				
IN1	IN2	IN3	IN4	OUT
6	5	6	6	6
5	4	5	5	5
6	4	6	7	4

Rank Entropy Monotonic Tree: Otras métricas

Métrica RSD

$$H_S(c|a_j) = \sum_{i=1}^{|N|} \left[-\log_2 \left(\frac{|[x_i]_{a_j}^{\leq} \cap [x_i]_c^{\leq}|}{|[x_i]_{a_j}^{\leq}|} \right) \right]$$

The diagram shows a dataset labeled "DATASET LEV" with columns IN1, IN2, IN3, IN4, and OUT. The OUT column is highlighted with a blue border. Two thresholds are defined: a_j (applied to IN2) and c (applied to OUT). The regions $[x_i]_{a_j}^{\leq}$ and $[x_i]_c^{\leq}$ are indicated by shaded boxes around specific values in the IN2 and OUT columns respectively.

DATASET LEV				
IN1	IN2	IN3	IN4	OUT
6	5	6	6	6
5	4	5	5	5
5	3	4	5	4
6	4	6	7	6

Annotations:

- a_j points to the value 4 in the IN2 column.
- c points to the value 5 in the OUT column.
- $[x_i]_{a_j}^{\leq}$ highlights the row where IN2 ≤ 4.
- $[x_i]_c^{\leq}$ highlights the row where OUT ≤ 5.

Rank Entropy Monotonic Tree: Otras métricas

Métrica RGD

$$H_G(c|a_j) = \sum_{i=1}^{|N|} \left(1 - \frac{|[x_i]_{a_j}^{\leq} \cap [x_i]_c^{\leq}|}{|[x_i]_{a_j}^{\leq}|} \right)$$

DATASET LEV				
IN1	IN2	IN3	IN4	OUT
6	5	6	6	6
5	4	5	5	5
5	3	4	5	4
6	4	6	7	6

Rank Entropy Monotonic Tree (REMT)

- Información ordinal, **Q. Hu, et al. 2012 IEEE TKDE**

Definition 1 Let $DT = \langle U, C, D \rangle$ be a decision table, $B \subseteq C$. We say DT is consistent in terms of B if for $\forall a \in B$, $x_i, x_j \in U$, $v(x_i, a) = v(x_j, a)$, we have $v(x_i, D) = v(x_j, D)$.

Definition 2 Let $DT = \langle U, C, D \rangle$ be a decision table, $B \subseteq C$. We say DT is ordinally consistent in terms of B if for $\forall a \in B$, $x_i, x_j \in U$, $v(x_i, a) \leq v(x_j, a)$, we have $v(x_i, D) \leq v(x_j, D)$.

Rank Entropy Monotonic Tree (REMT)

Definition 3 Given $DT = \langle U, C, D \rangle$, $B \subseteq C$, we define the following set

$$1) [x_i]_B^{\leq} = \{x_j \in U \mid x_i \leq_B x_j\};$$

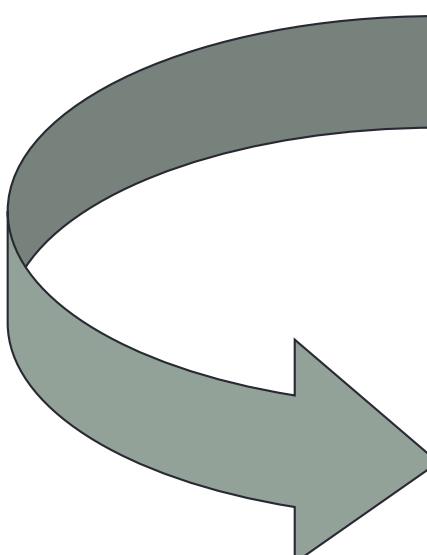
$$2) [x_i]_D^{\leq} = \{x_j \in U \mid x_i \leq_D x_j\}.$$

El subconjunto de ejemplos cuyos valores de atributos son menores que x_i en términos de atributos B .

El subconjunto de ejemplos cuyas decisiones son menores que x_i .

Rank Entropy Monotonic Tree (REMT)

Definition 5. Let $U = \{x_1, x_2, \dots, x_n\}$ be a set of samples described with a set of attributes A , $B \subseteq A$. The ascending rank entropy and descending rank entropy of the set U with respect to B are defined as


$$RH_B^<(U) = -\frac{1}{n} \sum_{i=1}^n \log \frac{|[x_i]_B^<|}{n}, \quad (1)$$

$$RH_B^>(U) = -\frac{1}{n} \sum_{i=1}^n \log \frac{|[x_i]_B^>|}{n} \quad (2)$$

Número de elementos

La entropía de Shannon se define como

$$RH_B^=(U) = -\frac{1}{n} \sum_{i=1}^n \log \frac{|[x_i]_B^=|}{n},$$

Rank Entropy Monotonic Tree (REMT)

Definition 6. Given $DT = \langle U, A, D \rangle$, $B \subseteq A$, $C \subseteq A$.
The ascending rank joint entropy of the set U with respect to B and C is defined as

$$RH_{B \cup C}^{\leq}(U) = -\frac{1}{n} \sum_{i=1}^n \log \frac{|[x_i]_B^{\leq} \cap [x_i]_C^{\leq}|}{n}, \quad (3)$$

and descending rank joint entropy of the set U with respect to B and C is defined as

$$RH_{B \cup C}^{\geq}(U) = -\frac{1}{n} \sum_{i=1}^n \log \frac{|[x_i]_B^{\geq} \cap [x_i]_C^{\geq}|}{n}. \quad (4)$$

Rank Entropy Monotonic Tree (REMT)

Definition 7. Given $DT = \langle U, A, D \rangle$, $B \subseteq A$, $C \subseteq A$. If C is known, the ascending rank conditional entropy of the set U with respect to B is defined as

$$RH_{B|C}^{\leq}(U) = -\frac{1}{n} \sum_{i=1}^n \log \frac{|[x_i]_B^{\leq} \cap [x_i]_C^{\leq}|}{|[x_i]_C^{\leq}|}, \quad (5)$$

and descending rank conditional entropy of the set U with respect to B is defined as

$$RH_{B|C}^{>}(U) = -\frac{1}{n} \sum_{i=1}^n \log \frac{|[x_i]_B^{>} \cap [x_i]_C^{>}|}{|[x_i]_C^{>}|}. \quad (6)$$

Rank Entropy Monotonic Tree (REMT)

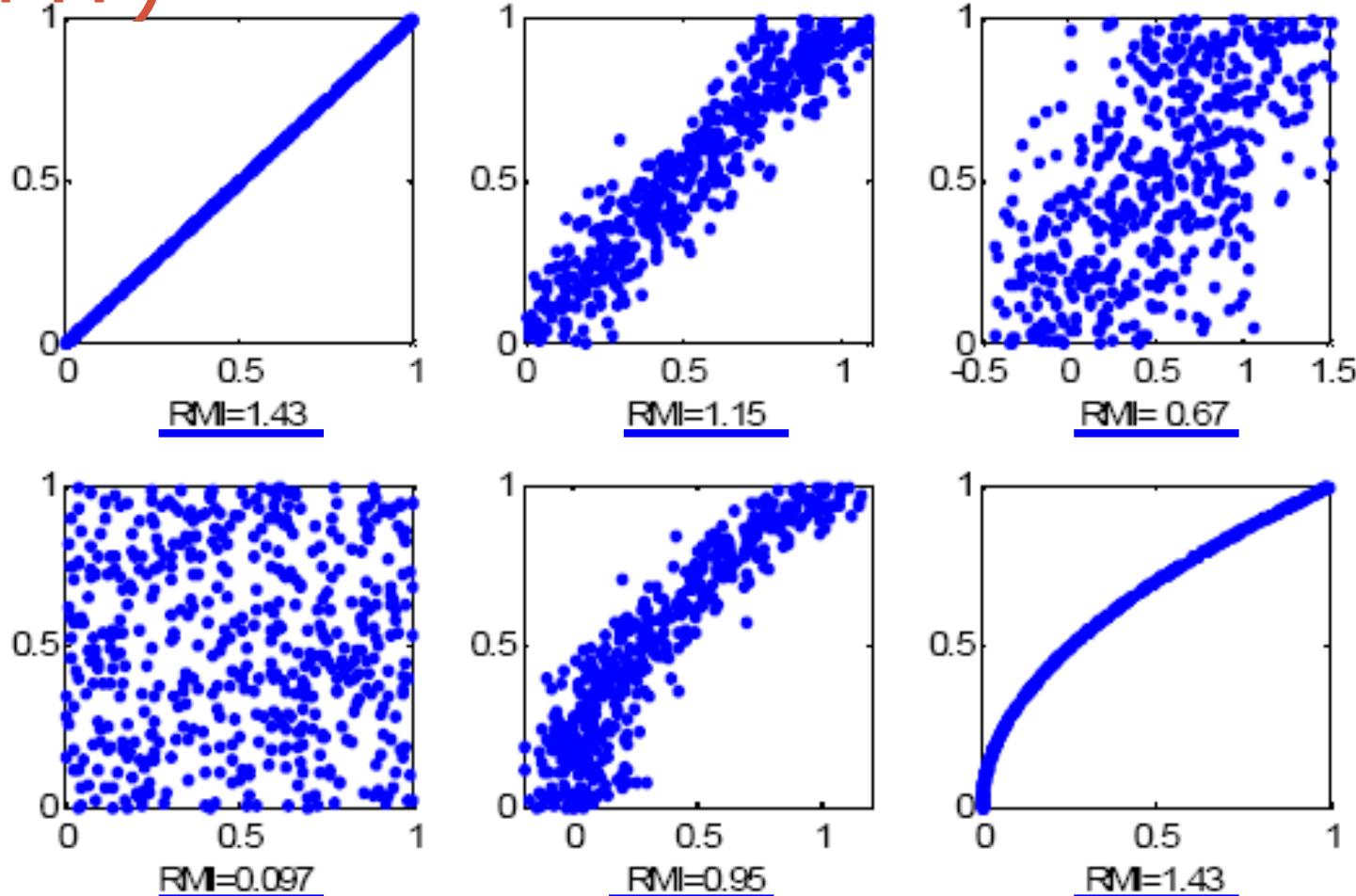
Definition 8. Given $DT = \langle U, A, D \rangle$, $B \subseteq A$, $C \subseteq A$.

The ascending rank mutual information (ARMI) of the set U regarding B and C is defined as

$$RMI^{\leq}(B, C) = -\frac{1}{n} \sum_{i=1}^n \log \frac{|[x_i]_B^{\leq}| \times |[x_i]_C^{\leq}|}{n \times |[x_i]_B^{\leq} \cap [x_i]_C^{\leq}|}, \quad (7)$$

Si B es un conjunto de atributos y C es una decisión, entonces RMI se puede ver como un coeficiente de relevancia ordinal entre B y C , así refleja la capacidad de B para predecir C .

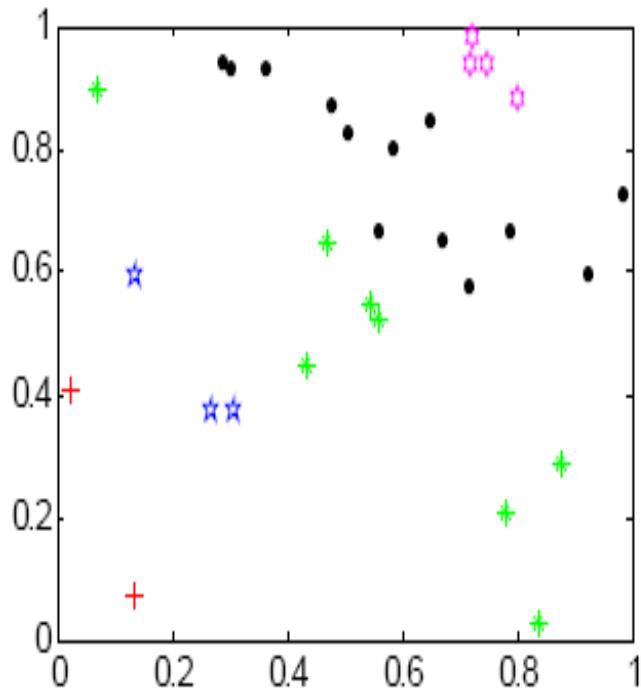
Rank Entropy Monotonic Tree (REMT)



El RMI ascendente entre X e Y. Si consideramos que x es un atributo, y es una decisión, entonces RMI refleja consistencia ordinal o monotonicidad.

Rank Entropy Monotonic Tree (REMT)

- Dado un conjunto de datos de entrenamiento, ¿cómo inducir un modelo de decisión sobre él
(REMT)

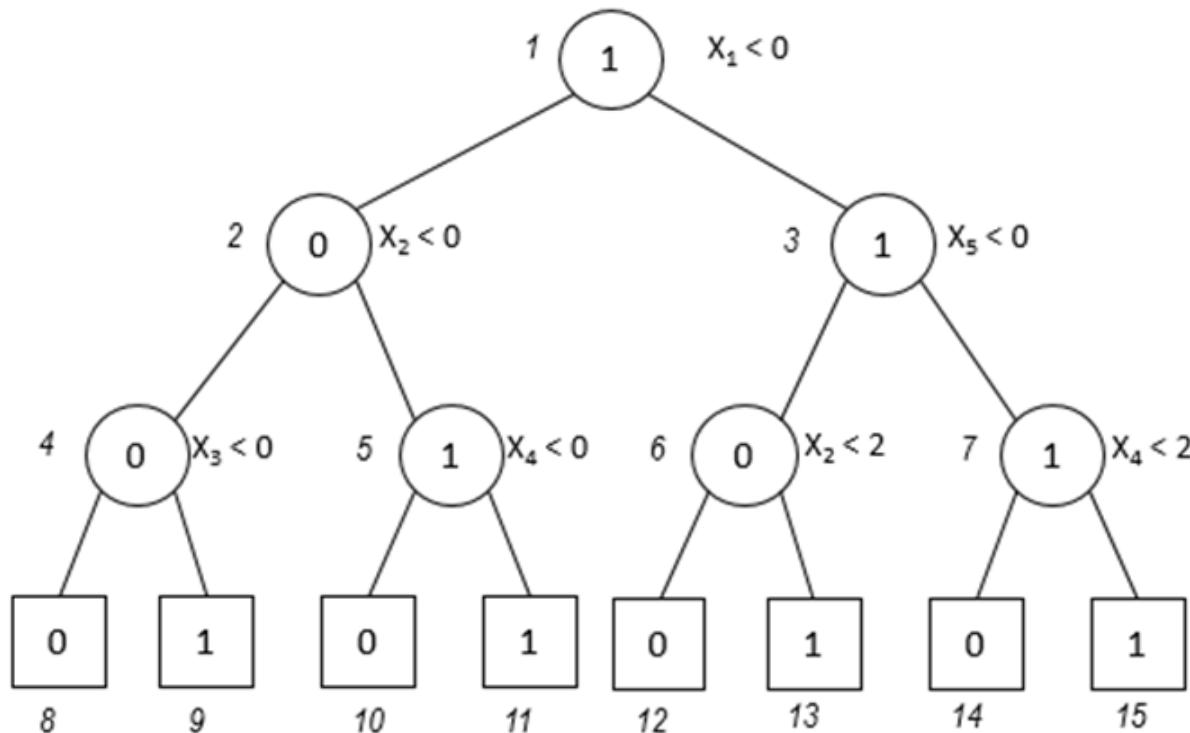


- 1. Calcula el RMI entre cada atributo y la decisión basada en la muestra de puntos de la raíz.**
- 2. Seleccionar el atributo con la máxima información mutua y dividirlo de acuerdo a sus valores.**
- 3. Calcula el RMI entre cada atributo y la decisión basada en la muestra de ese nodo y seleccionar el mejor atributo hasta el que el nodo sea puro.**

Poda Monotónica

- Hacer Poda es necesario para paliar los efectos del sobreaprendizaje
- No se pueden aplicar las técnicas de poda de árboles de decisión clásicos
- Métricas monotónicas no aseguran árboles monotónicos
- Se han propuesto dos algoritmos de poda:
 - Mayor Padre no monotónico (MNP)
 - Mejor arreglo (BF)

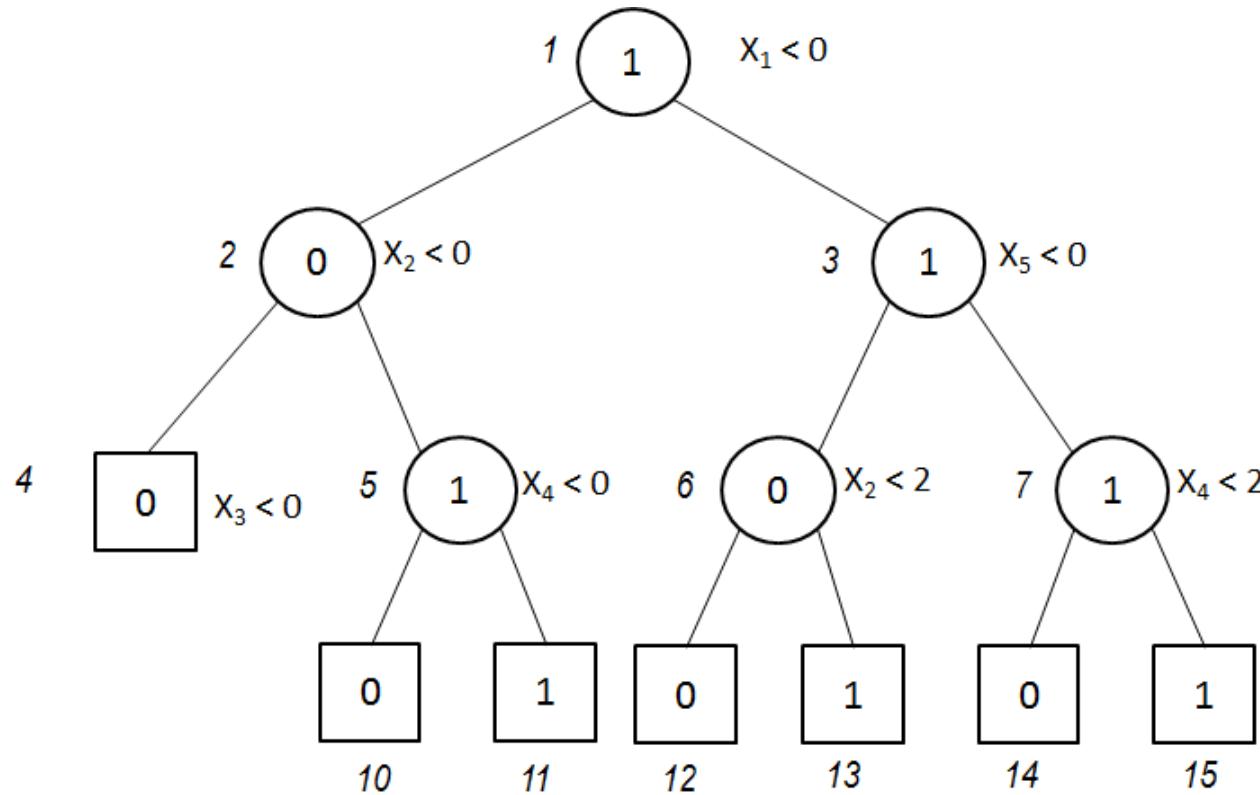
Poda monotónica (BF)



PAREJAS NO
MONOTÓNICAS

: [9, 10], [9, 12], [9, 14], [11, 12], [11, 14], [13, 14]

Poda monotónica (BF)

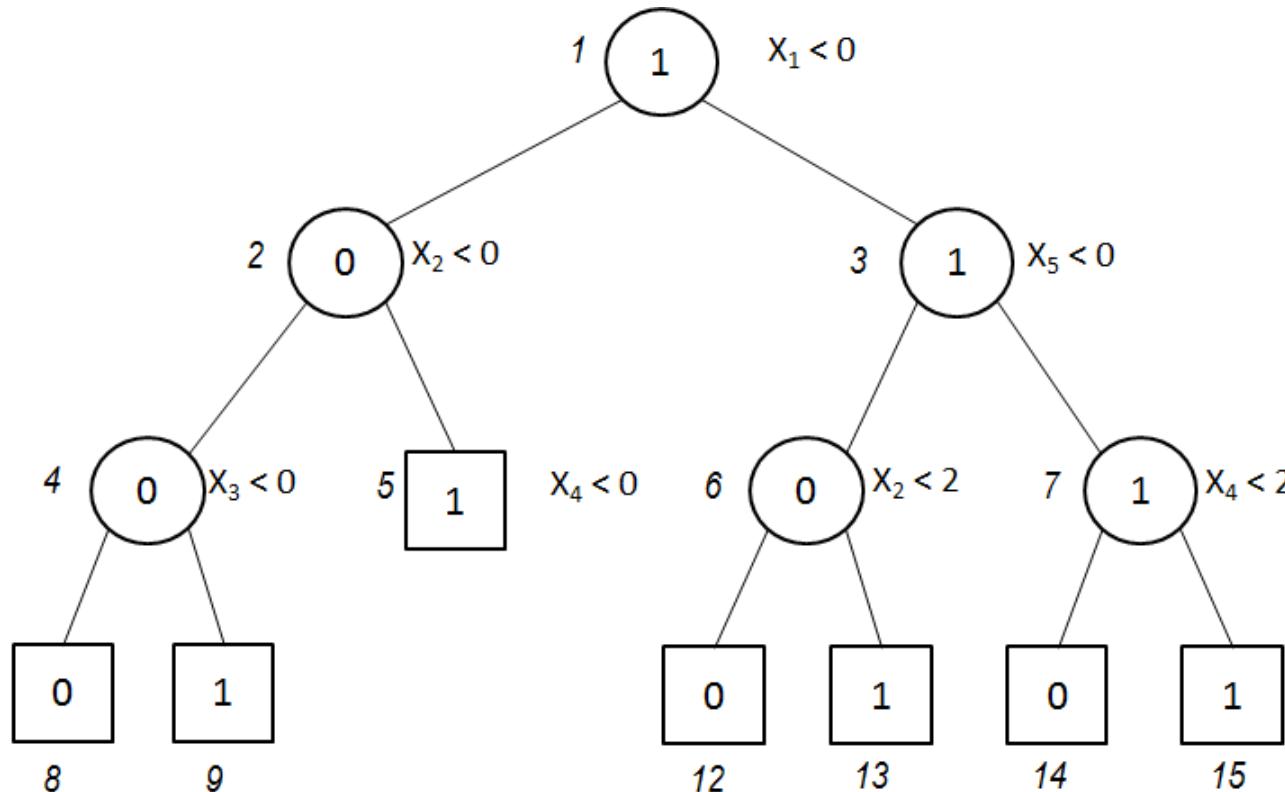


PAREJAS NO
MONOTÓNICAS

: [11, 12], [11, 14], [13, 14]

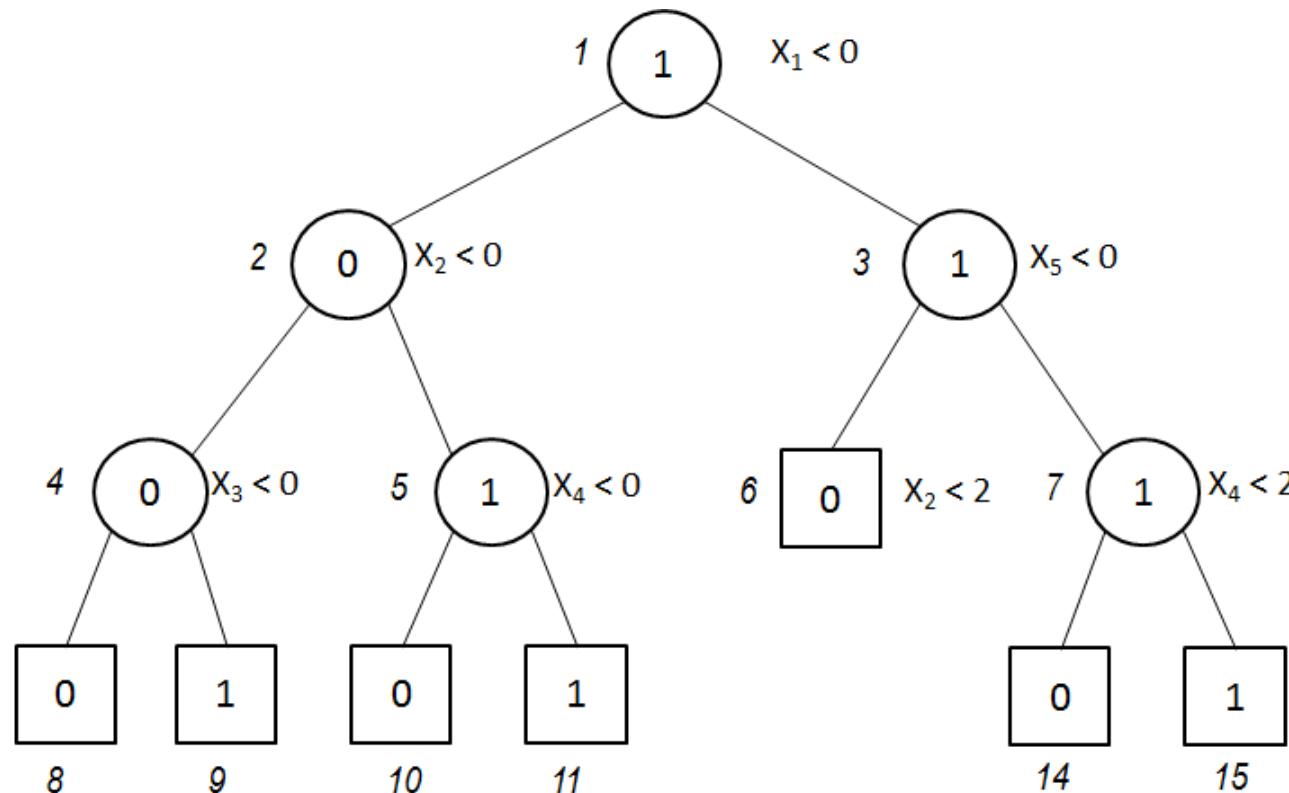
Podando NODO 4 quedan 3 parejas no monotónicas

Poda monotónica (BF)



PAREJAS NO MONOTÓNICAS : [9, 12], [9, 14], [5, 12], [5, 14], [13,14]
Podando NODO 5 quedan 5 parejas no monotónicas

Poda monotónica (BF)

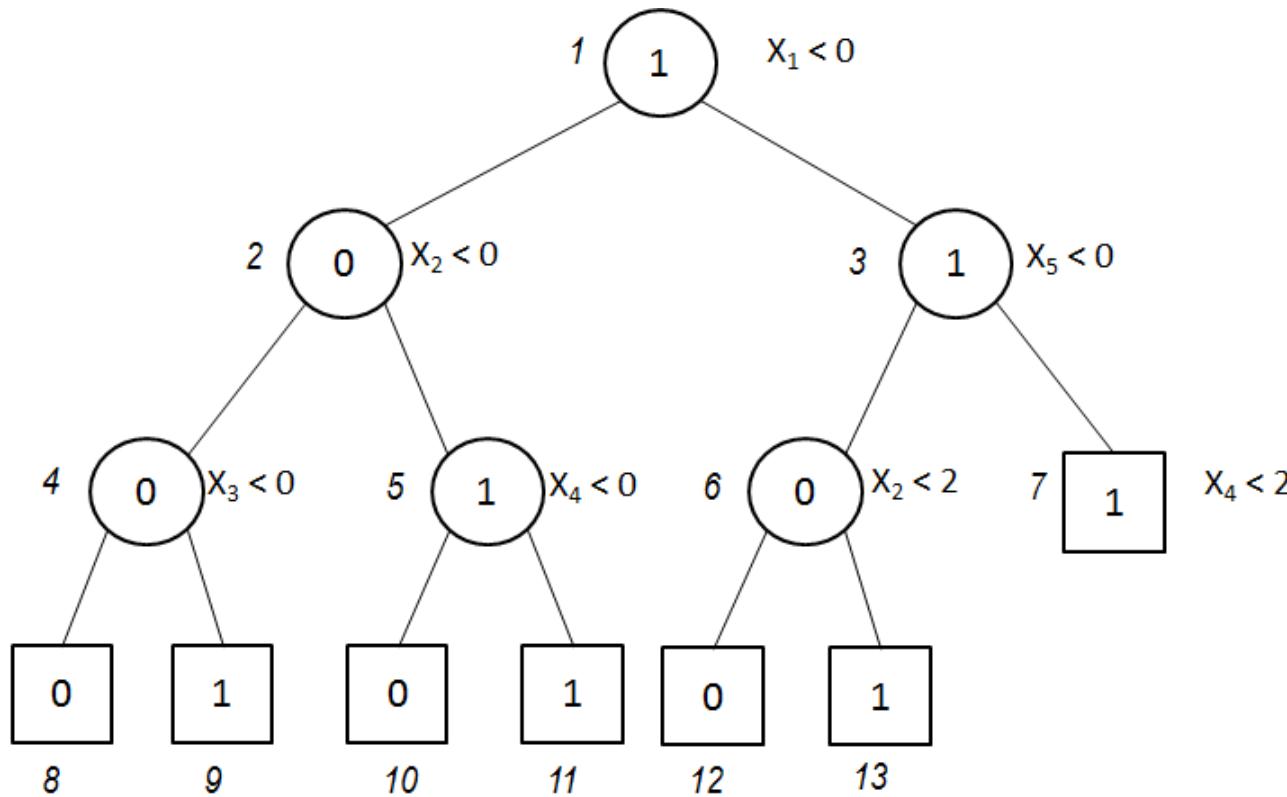


PAREJAS NO
MONOTÓNICAS

: [9, 10], [9, 6], [9, 14], [11, 6], [11, 14],

Podando NODO 6 quedan 5 parejas no monotónicas

Poda monotónica (BF)



PAREJAS NO

MONOTÓNICAS

: [9, 10], [9, 12], [11, 12]

Podando NODO 7 quedan 3 parejas no monotónicas

Poda monotónica (BF)

Hay 2 Nodos que si son podados dejan el mismo número de parejas monotónicas (Nodo 4 y 7). Para desempatar se emplean dos factores:

- 1.Aquel que tenga menor número de descendientes (BF).** En este caso hay el mismo número de descendientes.
- 2.Aquel que tenga menos instancias (LNO):** El nodo 4 tiene menos instancias que el 7, podándose así el nodo 4 (en el gráfico no aparece la cuenta de ejemplos que caen en cada hoja).

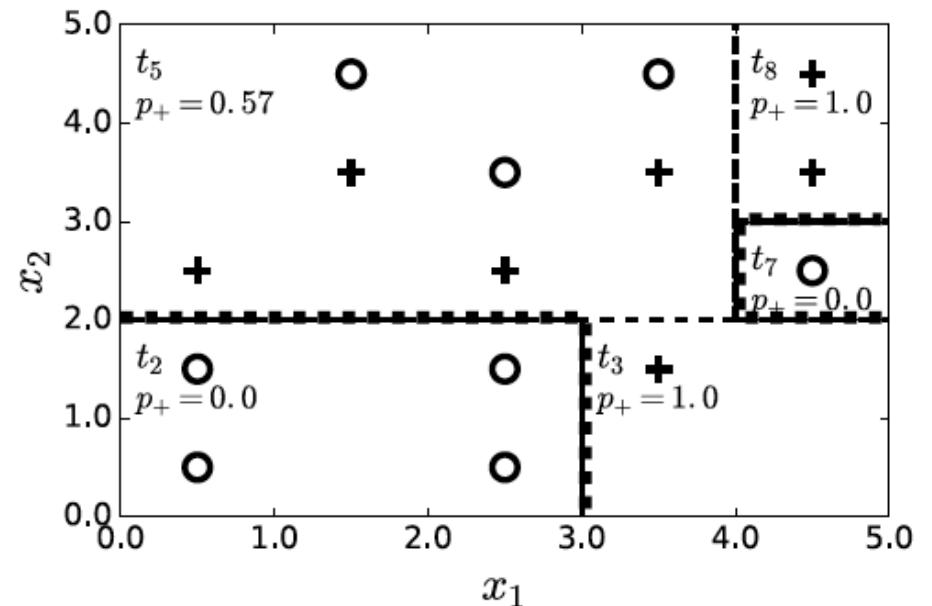
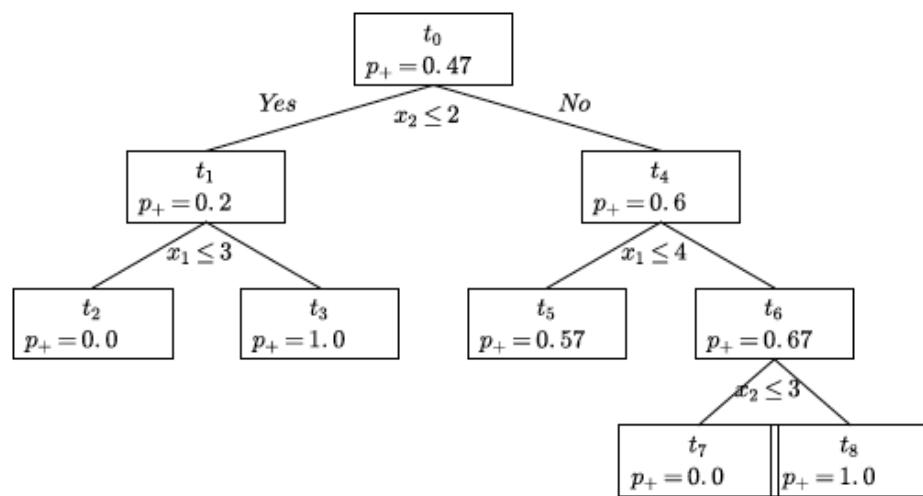
CLASIFICACIÓN MONOTÓNICA: Árboles de Decisión

- **Nonmonotone Split Constraint:**

- La solución más simple para monotonizar un árbol es restringir cada rama a generar splits no-monotónicos.
- En regresión, esto es igual a asegurar que las medias de los nodos hoja propuestos tienen un orden correspondiente al orden de hoja parcial.
- Para clasificación binaria, la restricción consiste en aplicar la $\text{Pr}(y=+1)$.
- Técnica implementada en los paquetes `arborist` y `GMB` de R.
- Para multiclase, se puede usar First Stochastic Dominance (FSD) que es orden parcial sobre probabilidad de distribuciones (cdf).
- No suponen coste computacional, pero no produce modelos totalmente monotónicos.

CLASIFICACIÓN MONOTÓNICA: Árboles de Decisión

- **Nonmonotone Split Constraint:**



- Este árbol no es monotónico global a pesar de que cada rama satisface las restricciones en ambos hijos.

Nested Generalized Examples (NGE)

NGE es una modificación del aprendizaje basados en ejemplos. Almacena ejemplos generalizados: hiperrectángulos. Similitudes con el aprendizaje basado en instancias y el basado en inducción de reglas.

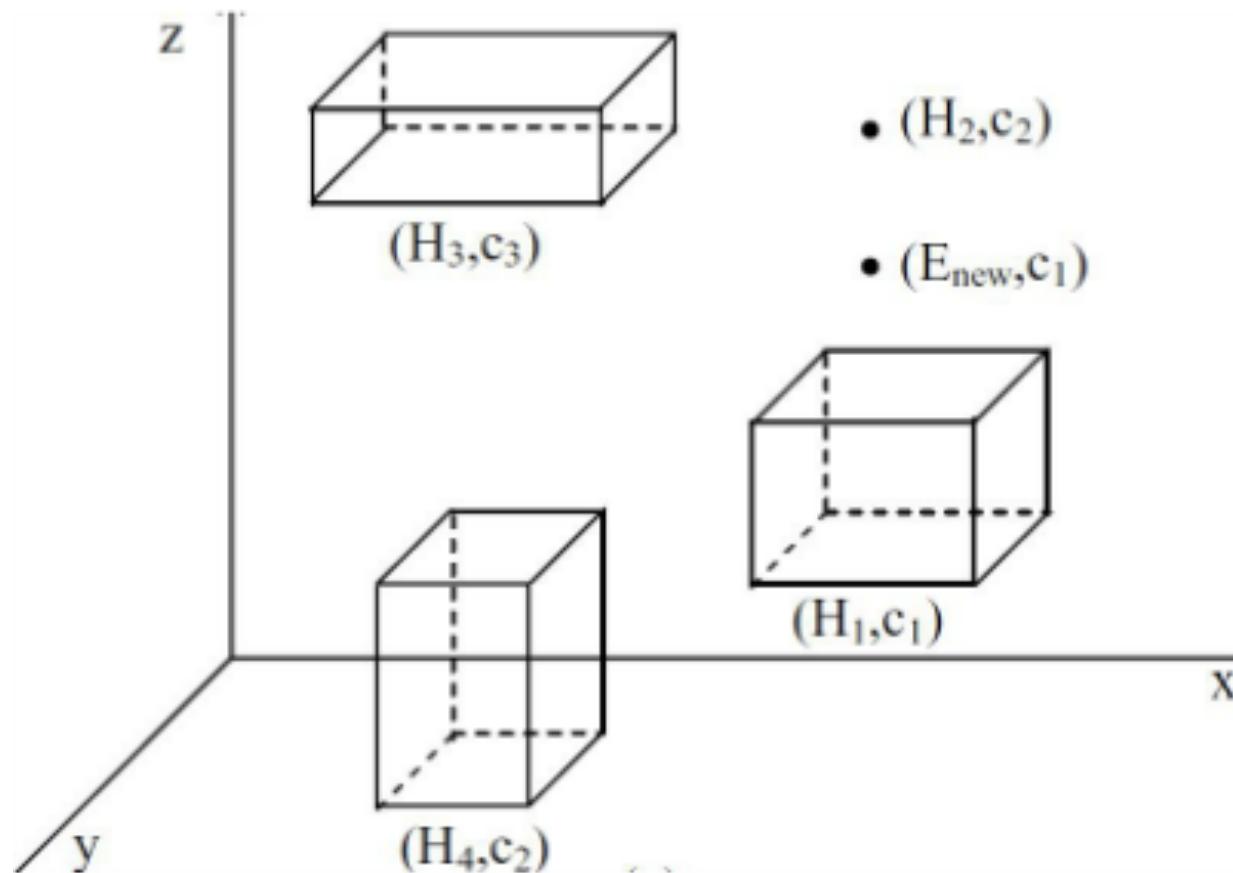
El pilar central de NGE es el proceso de correspondencia, donde se calcula una puntuación de correspondencia dada por:

$$D_{EH} = \sqrt{\sum_{i=1}^m \left(\frac{dif_i}{max_i - min_i} \right)^2}$$

$$dif_i = \begin{cases} E_{f_i} - H_{upper} & \text{cuando } E_{f_i} > H_{upper} \\ H_{lower} - E_{f_i} & \text{cuando } E_{f_i} < H_{lower} \\ 0 & \text{en otro caso} \end{cases}$$

Nested Generalized Examples (NGE)

GENERALIZACIÓN DE HIPERRECTÁNGULOS



MonGEL

DISTANCIAS	Númerico	Nominal
$D_{X_i R} = \sqrt{\sum_{j=1}^M \left(\frac{dis_j}{Range} \right)^2}$	$\text{Range} = max_j - min_j$ $dis_j = \begin{cases} X_{ij} - j_{max} & \text{if } X_{ij} > j_{max} \\ j_{min} - X_{ij} & \text{if } X_{ij} < j_{min} \\ 0 & \text{en otro caso} \end{cases}$	$\text{Range} = \text{Número de los posibles valores del atributo j}$ $dis_j = \begin{cases} 0 & X_{ij} \in A_j \\ 1 & \text{en otro caso} \end{cases}$
$D_{RR'} = \sqrt{\sum_{j=1}^M \left(\frac{dis_j}{Range} \right)^2}$	$dis_j = \left \frac{j_{\max} + j_{\min}}{2} - \frac{j'_{\max} + j'_{\min}}{2} \right $ $\text{Range} = \max_j - \min_j,$	$dis_j = \#(A_j \cap A'_j) / Range$ $Range = \text{Número de posibles Valores del atributo j}$

MonGEL

S , Set of Rules.

$S = \emptyset$.

For each instance \mathbf{x}_i of the training set

\mathbf{x}_i is transformed to a rule R_i of dimension 0.
 $S = S \cup R_i$.

Sort the rules in S by its class value in descending order, obtaining the subset S_C for each class.

Remove the repeated rules in all S_C .

Repeat

 For each class C

 Randomize the order of presentation of rules for class

 For each rule $R_i \in S_C$

 Find the nearest rule $R'_i \in S_C$ such that it is comparable with R_i .

 If R'_i exists

$R''_i = \text{Generalize}(R_i, R'_i)$.

 If for all $R_j \in S_C$, R_j and R''_i are disjoints

 Delete R'_i .

 Replace R_i by R''_i .

Until no generalization is done.

Construct the monotonicity violation matrix N , and the list V .

While i exists such that $v_i > 0$

 Obtain the row having the greatest number of monotonicity violations in V .

 In case of a tie, locate that row representing the rule which covers the smallest number of instances: p . If there are more than one row, choose one randomly.

 Remove the row p and column p in N .

 Update the list V according to N .

Etapa de Inicialización del conjunto de Reglas

Etapa de Generalización del conjunto de Reglas

Etapa de Eliminación de las reglas no monotónicas

MonGEL

ELIMINACIÓN DE REGLAS NO MONOTÓNICAS

$$N_1 = \begin{bmatrix} 0 & 1 & \textcolor{red}{1} & 0 & 0 \\ 1 & 0 & \textcolor{red}{1} & 0 & 1 \\ \textcolor{red}{1} & \textcolor{red}{1} & 0 & \textcolor{red}{1} & 1 \\ 0 & 0 & \textcolor{red}{1} & 0 & 1 \\ 0 & 1 & \textcolor{red}{1} & 1 & 0 \end{bmatrix}$$

$$V_1 = (2 \ 3 \ \textcolor{red}{4} \ 2 \ 3)$$

37

$$N_2 = \begin{bmatrix} 0 & 1 & 0 & \textcolor{red}{0} \\ 1 & 0 & 0 & \textcolor{red}{1} \\ 0 & 0 & 0 & \textcolor{red}{1} \\ \textcolor{red}{0} & 1 & 1 & 0 \end{bmatrix}$$

$$V_2 = (1 \ 2 \ 1 \ \textcolor{red}{2})$$

$$N_4 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$V_4 = (0 \ 0)$$

39

$$N_3 = \begin{bmatrix} 0 & 1 & 0 \\ \textcolor{red}{1} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$V_3 = (\textcolor{red}{1} \ 0 \ 0)$$

38

Monotonic Random Forest

Algorithm 1 Monotonic RF algorithm.

```
function MONRF( $D$  - dataset,  $nTrees$  - number of random trees built,  $R_{limit}$  -  
importance factor for monotonic constrains,  $T$  - Threshold used in the pruning pro-  
cedure,  $S$  - the predicted version of  $D$ )  
    initialize:  $S = \{\}$ ,  $Trees[1..nTrees]$ ,  $D_{bootstraps}[1..nTrees]$ ,  $NMIs[1..nTrees]$   
    for i in  $[1, nTrees]$  do  
         $D_{bootstraps}[i] = Bootstrap\_Sampler(nTrees, D)$   
         $rand = Random(1, R_{limit})$   
         $Trees[i] = Build\_Tree(D_{bootstraps}[i], rand)$   
         $NMIs[i] = Compute\_NMI(Trees[i])$   
    end for  
     $Trees = Sort(Trees, NMIs)$   
    for i in  $[1, \lceil nTrees * T \rceil]$  do  
         $\widehat{Trees} \leftarrow Trees[i]$   
    end for  
    for d in  $D$  do  
         $S \leftarrow Predict\_Majority\_Voting(\widehat{Trees}, d)$   
    end for  
    return  $S$   
end function
```

Monotonic Random Forest

- Se utiliza el factor R (MID) como una manera extra de aleatorizar y diversificar los diferentes árboles construidos en el RF.
- Al mismo tiempo, forzamos al proceso de creación de árboles a estar dominado por las consideraciones monotónicas.
- Para ello, cada árbol se construye desde el principio con un factor R diferente, escogido con un número aleatorio entre 1 y R_{limit} .

Monotonic Random Forest

Se utiliza un mecanismo de poda basado en un umbral de índice de monotonicidad de cada árbol resultante para la combinación final de predicción.

En lugar de utilizar todos los árboles, se seleccionan los mejores árboles de acuerdo a las violaciones de monotonicidad que generar de acuerdo a un determinado umbral.

Usando el criterio NMI, y ordenando los árboles en orden ascendente, el método selecciona los primeros t árboles, siendo t una tasa en el rango $(0,1]$.

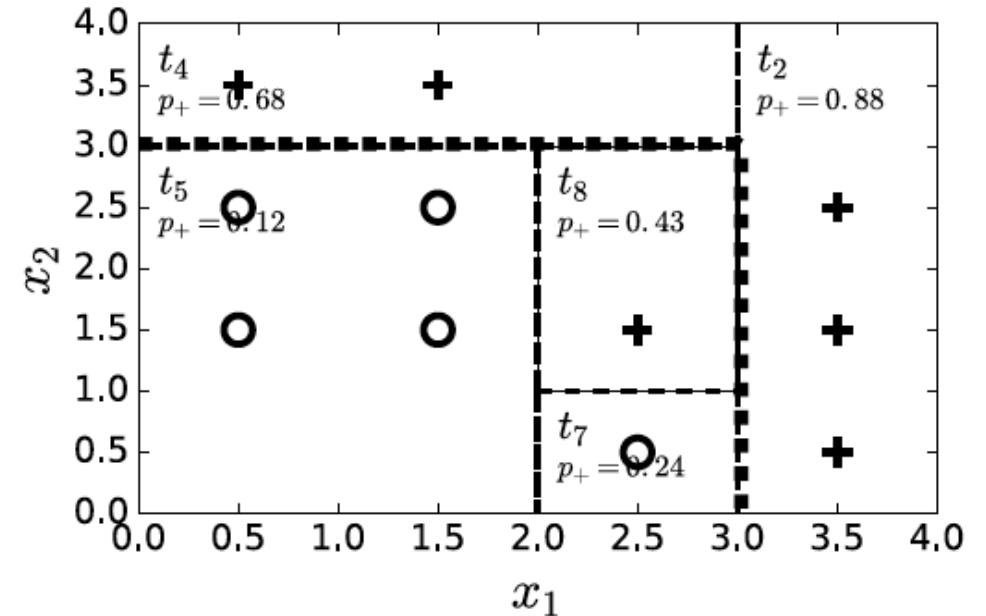
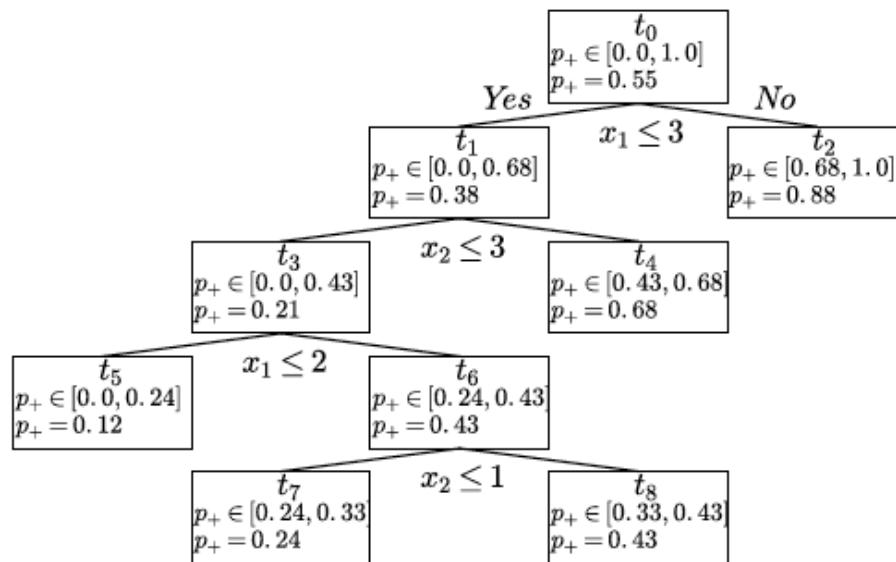
Experimentos muestran como un $t=0.5$ funciona adecuadamente.

CLASIFICACIÓN MONOTÓNICA: XGBoost

- **Monotone Split Constraint:**

- La popular biblioteca de gradient boosting utiliza esta aproximación.
- La restricción de split se hace añadiendo límites de coeficientes heredados de las hojas.
- En cada split, la media de los coeficientes se pasa como límite superior sobre los futuros coeficientes a la hoja de la izquierda, y un límite inferior a la hoja de la derecha.
- Estas restricciones heredadas se respetan al encontrar nuevos cortes y garantizan monotonía global, a pesar de no suponer carga computacional.
- El problema es que deteriora mucho la precisión, alejando cada árbol de los datos de entrenamiento.
- XGBoost solo trabaja con clases binarias.
- Hay más ensambles de árboles: FREMT, FCMT, FSD-RF, etc.

CLASIFICACIÓN MONOTÓNICA: XGBoost



- Desde t_3 , es imposible predecir la clase $y=+1$.
- Esto produce un fallo en t_8 , a pesar de que podría predecir $p_+=0.68$ sin violar monotonía.

CLASIFICACIÓN MONOTÓNICA: Descomposición Multi-clase

- Para SVM o XGBoost hace falta utilizar un OVA / OVO.
- La propuesta OVA es simple:
 - Variante de Frank y Hall.
 - Preserva la monotonicidad y limita la L1.
 - $h_c(\mathbf{x}) = \mathbf{1}[y \geq c]$, clasificador binario que distingue clase menor que c de clase mayor igual que c , donde $\mathbf{1}[\text{condición}] = 1$ si condición, sino 0.
 - El clasificador multi-clase viene dado por:

$$h(\mathbf{x}) = 1 + \sum_{c=2}^C h_c(\mathbf{x})$$

- **No existe aún una idea OVO en este campo.**

CLASIFICACIÓN MONOTÓNICA:

Algunos experimentos

Comparativa entre
C4.5 y métricas

- **Baja precisión** de todos los modelos.



PEQUEÑO CONJUNTO DE
DATOS

- **MID mejor balance**
Monotonía vs Precisión
- **Modelos con pocas hojas**

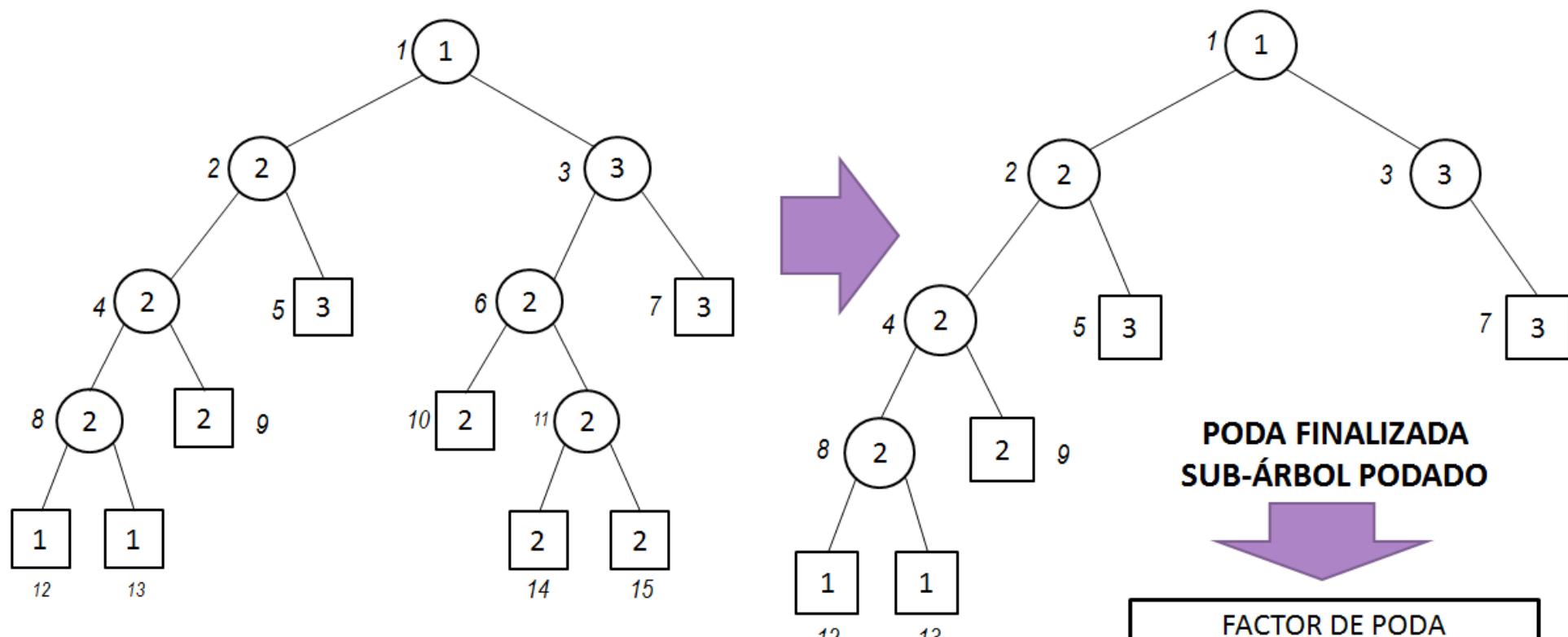


PODA AGRESIVA

	ESL	ESL	SWD	LEV
C4.5	Hojas: 16	Hojas: 39	Hojas 25	Hojas: 26
	INM: 8.33%	INM: 4.84%	INM: 10%	INM: 8.92%
	Accu: 33.3%	Accu: 76.22%	Accu: 63.8%	Accu: 65.2%
RGD	Hojas: 3	Hojas: 10	Hojas: 14	Hojas: 9
	INM: 0%	INM: 0%	INM: 0%	INM: 0%
RSD	Accu: 19.4%	Accu: 46.9%	Accu: 44.3%	Accu: 46.6%
	Hojas: 2	Hojas: 20	Hojas: 3	Hojas: 5
	INM: 0%	INM: 0%	INM: 0%	INM: 0%
RMI	Accu: 12.4%	Accu: 50.4%	Accu: 30%	Accu: 48%
	Hojas: 6	Hojas: 10	Hojas: 63	Hojas: 18
	INM: 0%	INM: 0%	INM: 0%	INM: 0%
MID	Accu: 21.4%	Accu: 36.8%	Accu: 57.9%	Accu: 53.2%
	Hojas: 16	Hojas: 40	Hojas: 35	Hojas: 27
	INM: 0%	INM: 0%	INM: 0%	INM: 0%
	Accu: 27%	Accu: 61.4%	Accu: 56.4%	Accu: 53.2%

CLASIFICACIÓN MONOTÓNICA: Algunos experimentos

Poda Agresiva



CLASIFICACIÓN MONOTÓNICA: Algunos experimentos

Análisis de efectos de poda

Se analizarán los resultados de hacer una poda menos agresiva (índice poda= 0.6) dejando un **60% de violaciones de monotonía.**



Se marcan los resultados donde INM es más elevado.

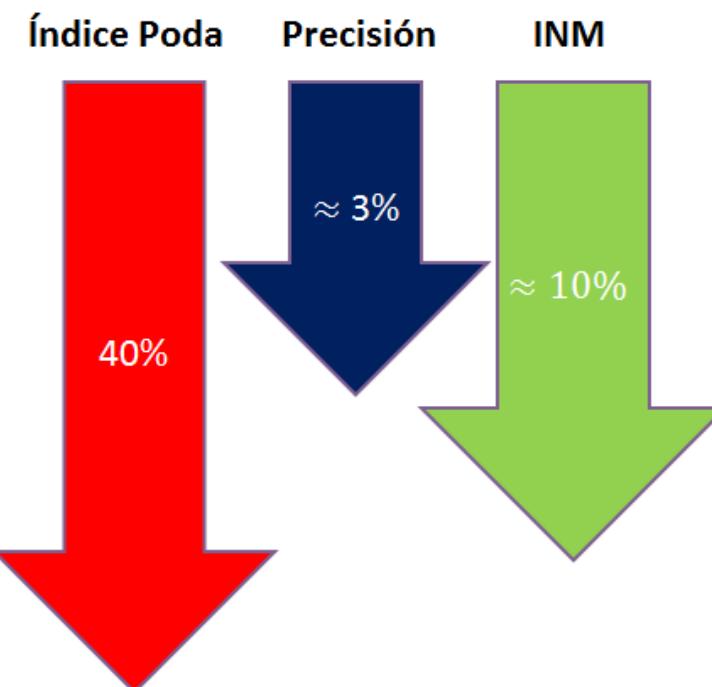
	ESL	ESL	SWD	LEV
MID	Hojas: 16 INM: 0% Accu: 27%	Hojas: 55 INM: 2.96% Accu: 68%	Hojas: 73 INM: 1.86% Accu: 60.8%	Hojas: 30 INM: 1.6% Accu: 54.5%
RSD	Hojas: 37 INM: 27.7% Accu: 33.3%	Hojas: 64 INM: 4.16% Accu: 67%	Hojas: 95 INM: 17.2% Accu: 60.7%	Hojas: 70 INM: 7.49% Accu: 62.3%
RGD	Hojas: 38 INM: 28.3% Accu: 32.6%	Hojas: 67 INM: 4.02% Accu: 66.1%	Hojas: 103 INM: 10.6% Accu: 62.6%	Hojas: 77 INM: 13.02% Accu: 64.4%
RMI	Hojas: 38 INM: 11.3% Accu: 33.4%	Hojas: 108 INM: 1.97% Accu: 77.8%	Hojas: 110 INM: 3.15% Accu: 63.6%	Hojas: 80 INM: 6.29% Accu: 66%

CLASIFICACIÓN MONOTÓNICA:

Algunos experimentos

Análisis de efectos de poda

Poda más agresiva, **sin llegar a ser 100% monotónica** (índice poda = 0.2)



	ESL	ESL	SWD	LEV
MID	Hojas: 16	Hojas: 48	Hojas: 55	Hojas: 28
	INM: 0%	INM: 1.24%	INM: 1.14%	INM: 0.26%
	Accu: 27%	Accu: 65.3%	Accu: 58.3%	Accu: 53.5%
RSD	Hojas: 27	Hojas: 54	Hojas: 65	Hojas: 59
	INM: 18.23%	INM: 2.02%	INM: 12.3%	INM: 3.5%
	Accu: 30%	Accu: 60.4%	Accu: 55.5%	Accu: 58.7%
RGD	Hojas: 26	Hojas: 58	Hojas: 78	Hojas: 62
	INM: 18.76%	INM: 1.69%	INM: 6.16%	INM: 6.61%
	Accu: 28.7%	Accu: 62.3%	Accu: 59%	Accu: 59.1%
RMI	Hojas: 30	Hojas: 98	Hojas: 103	Hojas: 61
	INM: 5.05%	INM: 0.86%	INM: 1.18%	INM: 3.82%
	Accu: 31.7%	Accu: 75.4%	Accu: 60.9%	Accu: 64.2%

CLASIFICACIÓN MONOTÓNICA: Algunos experimentos

Análisis de métricas (Eficiencia)

Número de podas necesarias para hacer un árbol de decisión totalmente monotónico

	ERA	ESL	SWD	LEV	MEDIA
RGD	45	70	105	82	75.5
RSD	46	54	113	74	71.75
RMI	45	110	56	69	70
MID	0	19	41	5	16.25

- La poda consume un tiempo significativo dentro de la obtención del modelo.
Múltiples recorridos del árbol completo.

MENOR ITERACIONES



MENOR TIEMPO EJECUCIÓN

CLASIFICACIÓN MONOTÓNICA: Algunos experimentos

Análisis de métricas (Estadísticos)

ÍNDICE DE MONOTONÍA
MID mejor que resto de métricas
PRECISIÓN
RMI mejor que resto de métricas

	ESL	ESL	SWD	LEV
MID	INM: 0%	INM: 4.7%	INM: 2.98%	INM: 3.22%
RSD	INM: 35.6%	INM: 6.52%	INM: 22.4%	INM: 11.6%
RGD	INM: 37.6%	INM: 5.94%	INM: 15.4%	INM: 18.6%
RMI	INM: 15.6%	INM: 3.29%	INM: 5.3%	INM: 9.94%
	Accu: 27%	Accu: 68.4%	Accu: 61%	Accu: 55.3%
	Accu: 34.2%	Accu: 69.2%	Accu: 63.7%	Accu: 64.3%
	Accu: 34.2%	Accu: 68.4%	Accu: 64.1%	Accu: 66.3%
	Accu: 34.2%	Accu: 78.6%	Accu: 65.2%	Accu: 66.9%

NOTA: Los malos resultados en RSD y RGD debido a que conjunto de datos de entrada no es monotónico.

CLASIFICACIÓN MONOTÓNICA: Selección de Instancias

Data & Knowledge Engineering 112 (2017) 94–105

■ Probabilistic Collision Removal

- Before the TSS, we apply an ordered probabilistic removal of harmful instances.
- The process is stochastic to avoid falling in local optima.
- Two parameters are introduced
 - **Candidates**, which points out the rate of the best candidates to be selected.
 - **CollisionsAllowed**, which represents the minimal rate of collisions permitted to stop the removal process.

CLASIFICACIÓN MONOTÓNICA: Selección de Instancias

Data & Knowledge Engineering 112 (2017) 94–105

■ Probabilistic Collision Removal

```

1: function ProbColRemoval(T - training data, Candidates - candidate rate, CollisionsAllowed - minimal rate of collisions
allowed)
2:   initialize: totalCollisions=0, S=T
3:   [Col[], ColMatrix[][]], totalCollisions]=Calculate_Collisions(S)
4:   maxCandidates=# S.Candidates
5:   minCol = totalCollisions.CollisionsAllowed
6:   newCol = totalCollisions
7:   repeat
8:     S=sort(S,Col[]])
9:     candSelected=Rand(1, maxCandidates)
10:    S=S \ {xcandSelected}
11:    newCol=newCol - Col[candSelected]
12:    Col[candSelected] = 0
13:    for all xi ∈ T do
14:      if (ColMatrix[j][candSelected] = true ) then
15:        ColMatrix[j][candSelected] = false
16:        Col[j] = Col[j] - 1
17:      end if
18:    end for
19:    until newCol < minCol
20:    return S
21: end function

22: function Calculate_Collisions(T - training data)
23:   for all xi ∈ T do
24:     Col[i]=0
25:     for all xj ∈ T do
26:       ColMatrix[i][j]=0
27:     end for
28:   end for
29:   for all xi ∈ T do
30:     for all xj ∈ T do
31:       if (i ≠ j ) then
32:         if (xi≤xj and Y(xi) > Y(xj) or (xi = xj and Y(xi) ≠ Y(xj)) then
33:           Col[i]=Col[i] + 1
34:           Col[j]=Col[j] + 1
35:           ColMatrix[i][j] = true
36:           ColMatrix[j][i] = true
37:         end if
38:       end if
39:     end for
40:   end for
41:   totalCollisions = 0
42:   for all xi ∈ T do
43:     totalCollisions = totalCollisions + Col[i]
44:   end for
45:   return [Col[], ColMatrix[][]], totalCollisions
46: end function

```

CLASIFICACIÓN MONOTÓNICA: Selección de Instancias

Data & Knowledge Engineering 112 (2017) 94–105

■ Quality Metrics Selection

■ Delimitation (Del)

$$\text{Del}(x_i) = \frac{|\text{Dom}(x_i) - \text{NoDom}(x_i)|}{\text{Dom}(x_i) + \text{NoDom}(x_i)},$$

$$\text{Dom}(x_i) = \#X', x' \in X' \Leftrightarrow x_i < x' \wedge Y(x_i) = Y(x'),$$

■ Influence (Infl)

$$\text{NoDom}(x_i) = \#Z', x' \in Z' \Leftrightarrow x_i \geq x' \wedge Y(x_i) = Y(x'),$$

kNN_{x_i} = k nearest neighbors of x_i

$$\text{nWeight}(x_j) = \frac{\sum_{l=1}^k \text{Distance}(x_i, x_l) - \text{Distance}(x_i, x_j)}{\sum_{l=1}^k \text{Distance}(x_i, x_l)}, \quad \forall x_j \in kNN_{x_i},$$

$$\text{influenceWeight}(x_j) = \frac{\text{nWeight}(x_j)}{\sum_{l=1}^k \text{nWeight}(x_l)}, \quad \forall x_j \in kNN_{x_i},$$

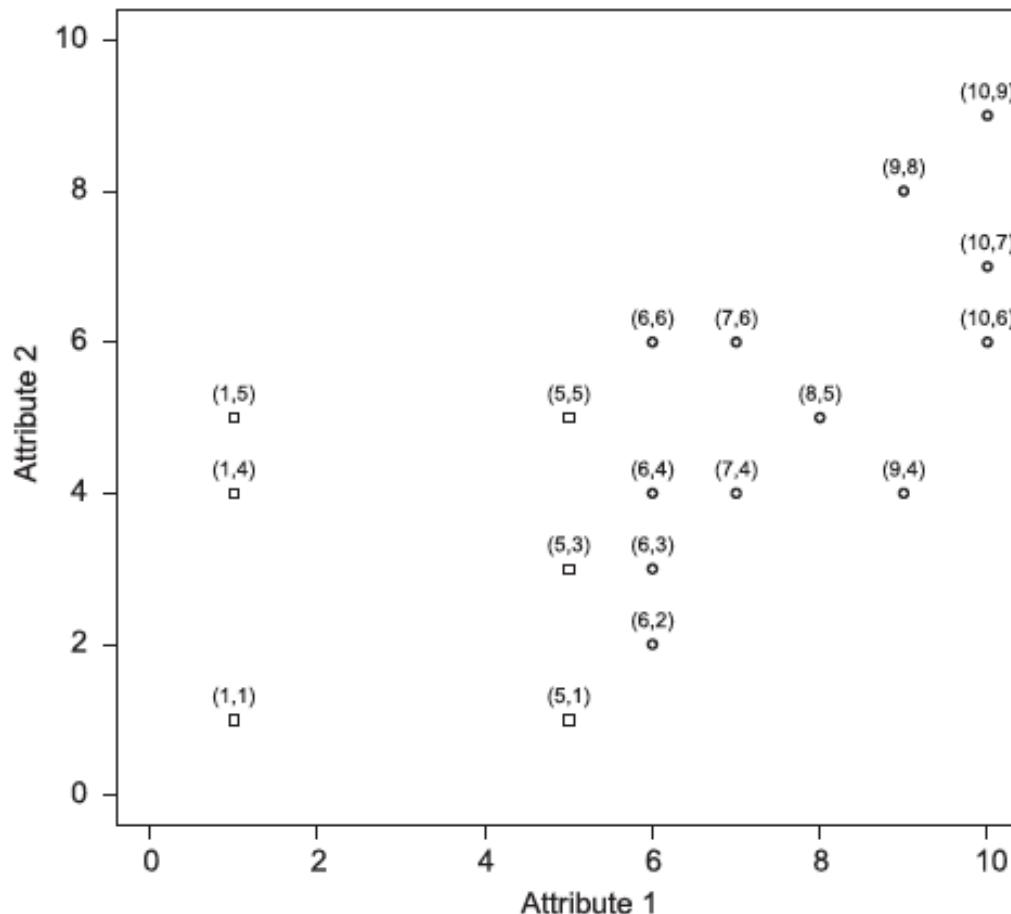
$$\text{Infl}(x_i) = \sum_{j=1}^k \text{influenceWeight}(x_j),$$

where $Y(x_i) \neq Y(x_j) \wedge x_j \in kNN_{x_i}$,

CLASIFICACIÓN MONOTÓNICA: Selección de Instancias

Data & Knowledge Engineering 112 (2017) 94–105

■ Quality Metrics Selection



$$(1,1) \Rightarrow \text{Dom } ((1,1))=5, \text{ NoDom } ((1,1))=0 \Rightarrow \text{Del}((1,1))=\frac{|5 - 0|}{5}=\frac{5}{5}=1$$

$$(1,4) \Rightarrow \text{Dom } ((1,4))=2, \text{ NoDom } ((1,4))=1 \Rightarrow \text{Del}((1,4))=\frac{|2 - 1|}{3}=\frac{1}{3}=0.333$$

$$(1,5) \Rightarrow \text{Dom } ((1,5))=1, \text{ NoDom } ((1,5))=2 \Rightarrow \text{Del}((1,5))=\frac{|1 - 2|}{3}=\frac{1}{3}=0.333$$

$$(5,1) \Rightarrow \text{Dom } ((5,1))=2, \text{ NoDom } ((5,1))=1 \Rightarrow \text{Del}((5,1))=\frac{|2 - 1|}{3}=\frac{1}{3}=0.333$$

$$(5,3) \Rightarrow \text{Dom } ((5,3))=1, \text{ NoDom } ((5,3))=2 \Rightarrow \text{Del}((5,3))=\frac{|1 - 2|}{3}=\frac{1}{3}=0.333$$

$$(5,5) \Rightarrow \text{Dom } ((5,5))=0, \text{ NoDom } ((5,5))=5 \Rightarrow \text{Del}((5,5))=\frac{|0 - 5|}{5}=\frac{5}{5}=1$$

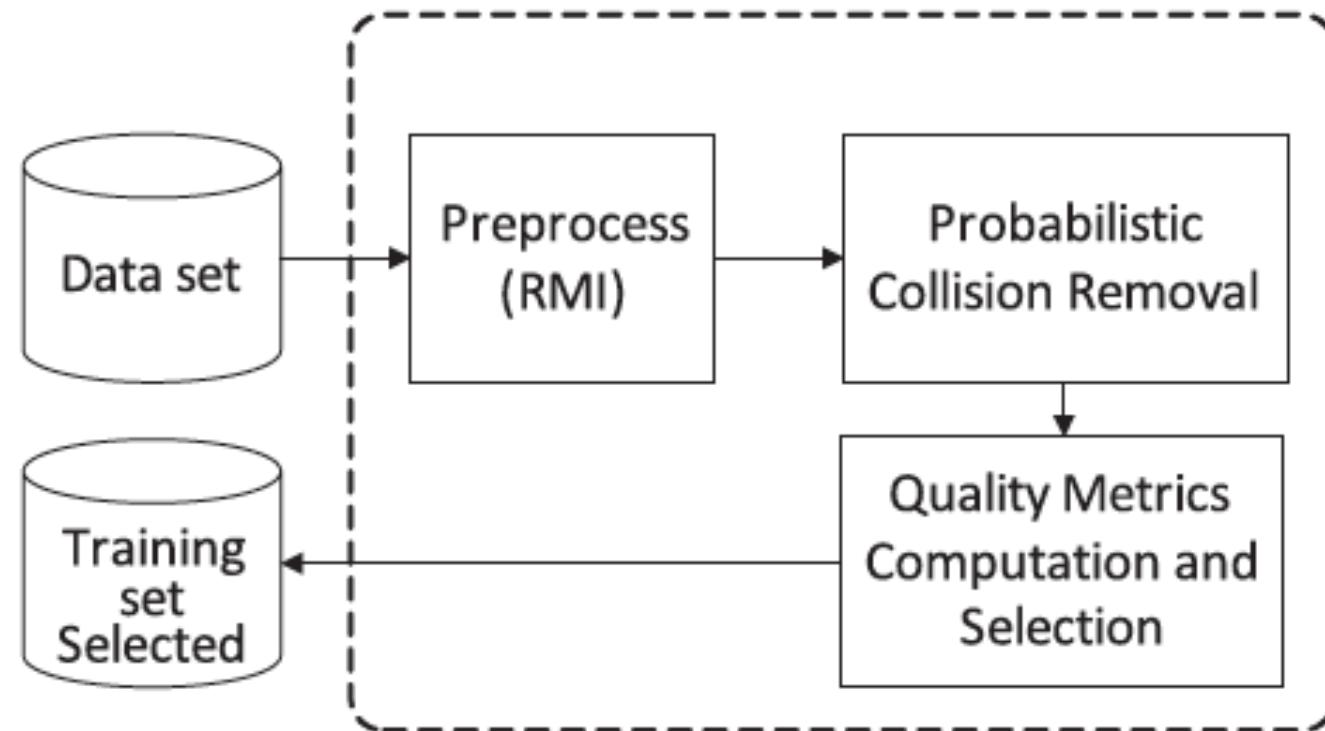
Example of calculation of *Influence* metric for point (5,3).

Considering $k=5$, $kNN_{(5,3)}=\{(5,5), (5,1), (6,4), (6,3), (6,2)\}$
 $nWeight((5,5))=0.744 \Rightarrow influenceWeight((5,5))=0.186$
 $nWeight((5,1))=0.744 \Rightarrow influenceWeight((5,1))=0.186$
 $nWeight((6,4))=0.819 \Rightarrow influenceWeight((6,4))=0.204$
 $nWeight((6,3))=0.872 \Rightarrow influenceWeight((6,3))=0.218$
 $nWeight((6,2))=0.819 \Rightarrow influenceWeight((6,2))=0.204$
 $Infl(5,3)=influenceWeight((6,4)) + influenceWeight((6,3)) + influenceWeight((6,2))=0.626$

CLASIFICACIÓN MONOTÓNICA: Selección de Instancias

Data & Knowledge Engineering 112 (2017) 94–105

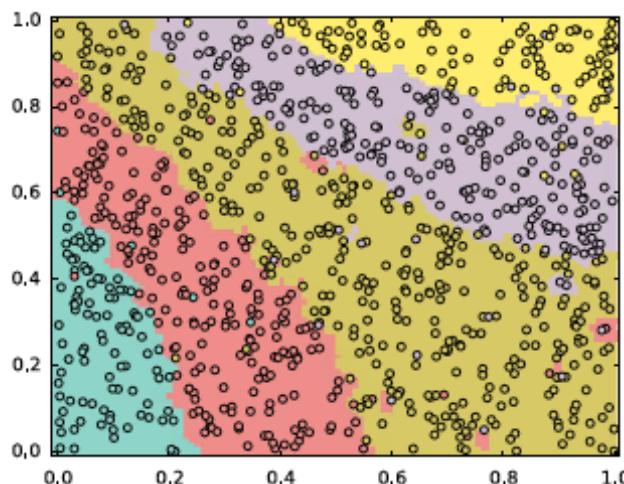
■ Algorithm



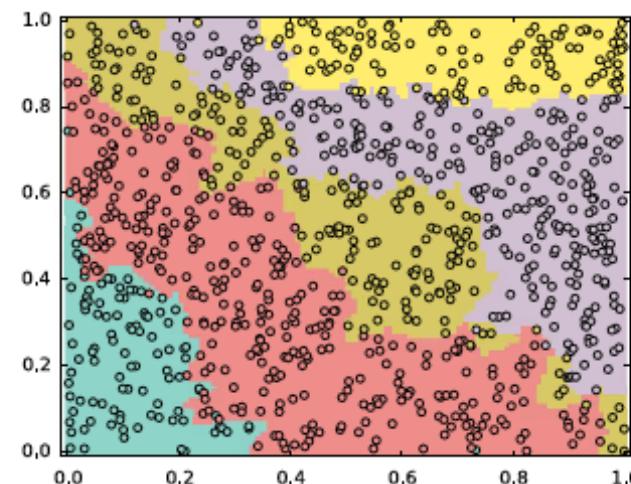
$$\text{Select } x_i = \begin{cases} \text{true} & \text{if } \text{Del}(x_i) < \text{Infl}(x_i) \\ & \text{or } \text{Del}(x_i) \geq 0.9 \\ \text{false} & \text{otherwise.} \end{cases}$$

CLASIFICACIÓN MONOTÓNICA: Selección de Instancias

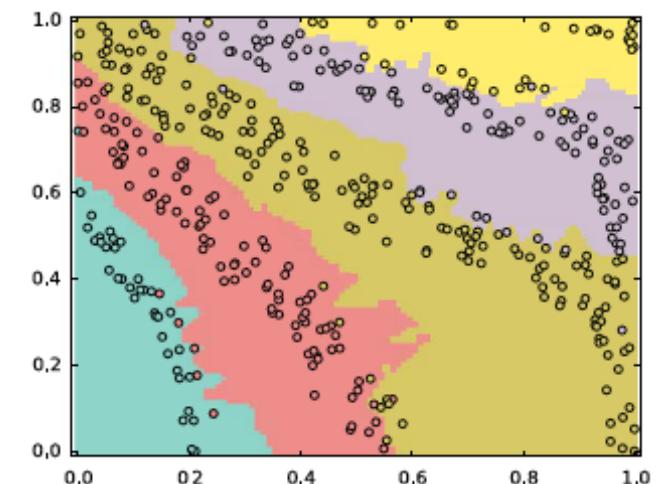
Data & Knowledge Engineering 112 (2017) 94–105



(a) Artiset Original



(b) Artiset Reableled.



(c) Artiset MonTSS.

Fig. 3. Artificial data set preprocessed by Relabeling and MonTSS with the borders calculated by MkNN with 3 neighbors.

THE FUTURE

- **Monotonic Imbalanced Classification:**
- Ordinal classification is frequently associated to imbalanced classes.

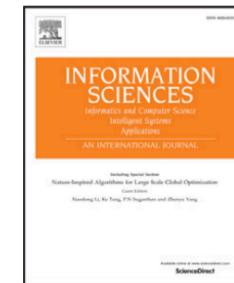
Information Sciences 474 (2019) 187–204



Contents lists available at [ScienceDirect](#)

Information Sciences

journal homepage: www.elsevier.com/locate/ins



Chain based sampling for monotonic imbalanced classification

Sergio González^{a,*}, Salvador García^a, Sheng-Tun Li^b, Francisco Herrera^{a,c}

^a Department of Computer Science and Artificial Intelligence, University of Granada, Granada 18071, Spain

^b Department of Industrial and Information Management, National Cheng Kung University, Tainan 701, Taiwan ROC

^c Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia



THE FUTURE

- **Monotonic Data Streams Classification:**
- Does it make sense?
- **1 result found:** BT* - An advanced algorithm for anytime classification. LNCS 2012.

THE FUTURE

- **Monotonic Multi-Label/Domain Classification:**
- Imagine the prediction of breast cancer:
 - **Inputs** based on a biopsy of a cell or a lymph node: size, radius, perimeter, symmetry, concavity, etc...
 - **Outputs:** STAGE (0 – I – IIA – IIB – III - IV) and Proliferation Degree (0 – 1 – 2 – 3).
- My view is that it makes sense (very reduced applications).

THE FUTURE

- **Monotonic Subgroup Discovery / Association Rules:**
- Example from the project NUTRIMENTHE:
 - **Inputs:** SE_F_stat_work=full time >34h/w, SE_M_career=employee, FamilyStatus=Living together, etc.
 - **Outputs:** Scale from 0 to 10: Attention Problems, Anxious Depressed, Aggressive Behavior, Sleep Problems, etc...
- My view is that it makes sense (many applications).