# 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

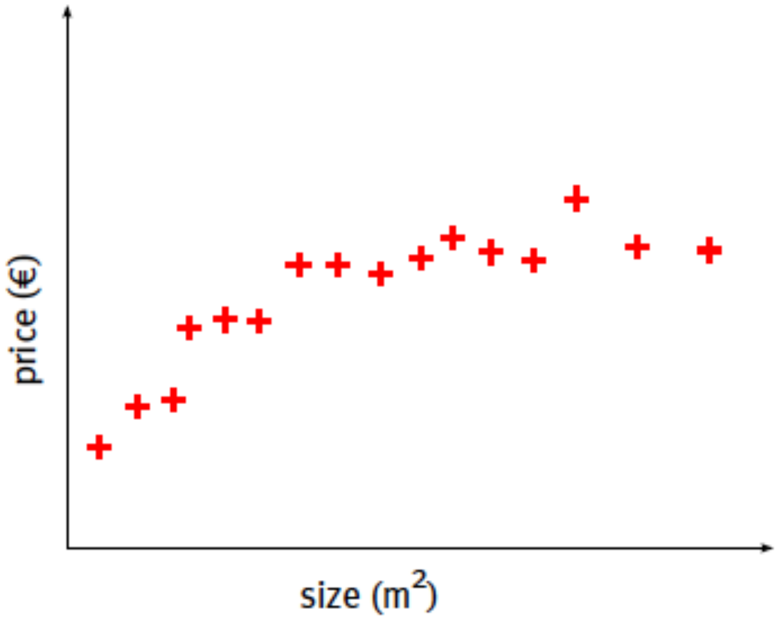
2. Clasificación Monotónica

# CLASIFICACIÓN ORDINAL Y MONOTÓNICA

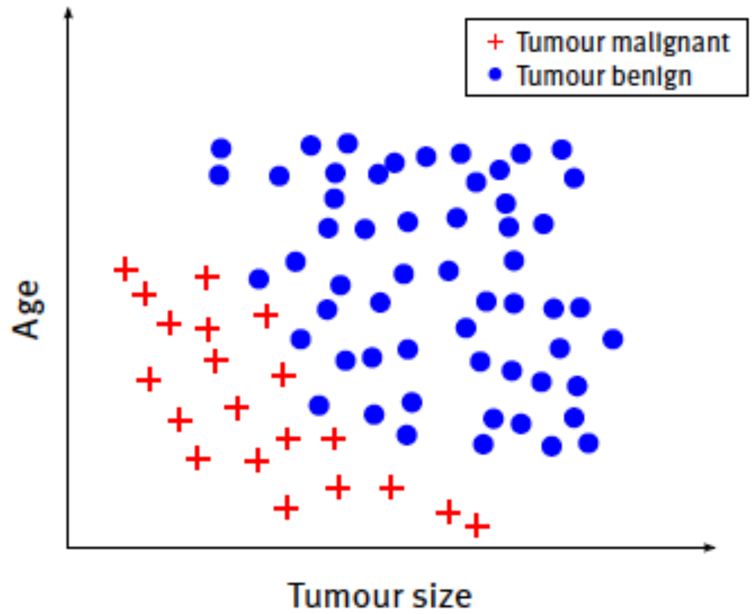
1. Clasificación Ordinal

2. Clasificación Monotónica

# APRENDIZAJE SUPERVISADO

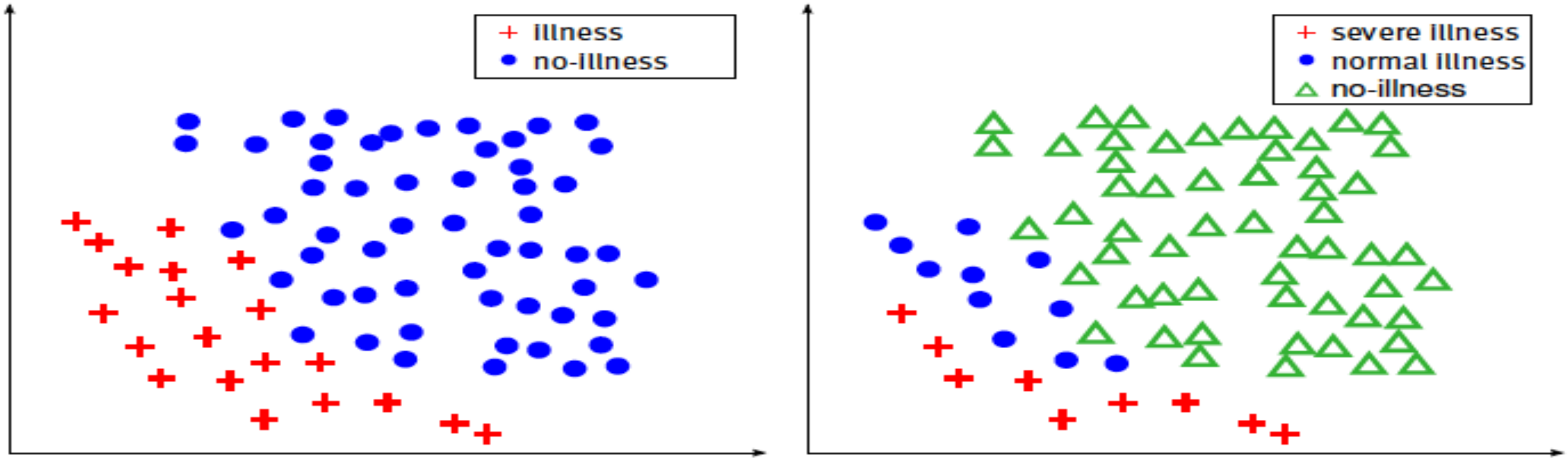



Problema de regresión "Dados estos datos, para una casa de 75m$^2$, ¿cuánto se espera obtener?"

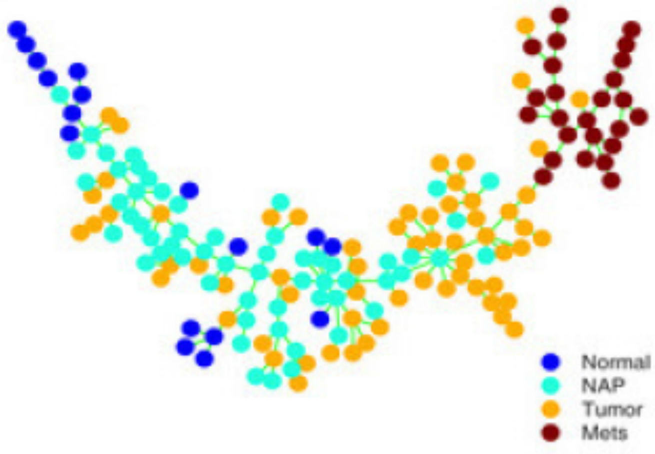



Problema de clasificación "¿Se puede obtener el tipo de tumor dada la edad y el tamaño del mismo?"

# CLASIFICACIÓN BINARIA VS. ORDINAL




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



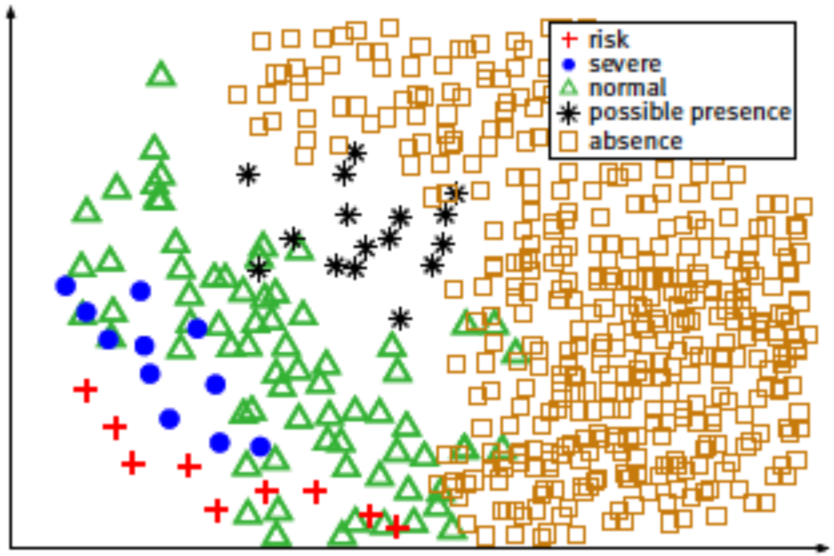
# REGRESIÓN/CLASIFICACIÓN ORDINAL

**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

$\{C_1$ = riesgo, $C_2$ = severa, $C_3$ = normal, $C_4$ = posible presencia, $C_5$ = 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 $\phi$ desde el espacio de entrada $X$ a un conjunto finito $C = \{C_1, C_2, \ldots, C_Q\}$ conteniendo $Q$ etiquetas, donde el conjunto de etiquetas tiene una relación de orden lineal $C_1 \prec C_2 \prec \ldots \prec C_Q$

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

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

Ordinal Regression commonly taken as standard multinomial classification:

- Ignore ordering information $\rightarrow$ Worse Classifiers.
- All mistakes are equally penalysed:
    - 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 sligthly different scenario, where the objective is to order all pairs of samples:

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

Circular ordinal regression: directional predictions.

# MEDIDAS DE RENDIMIENTO

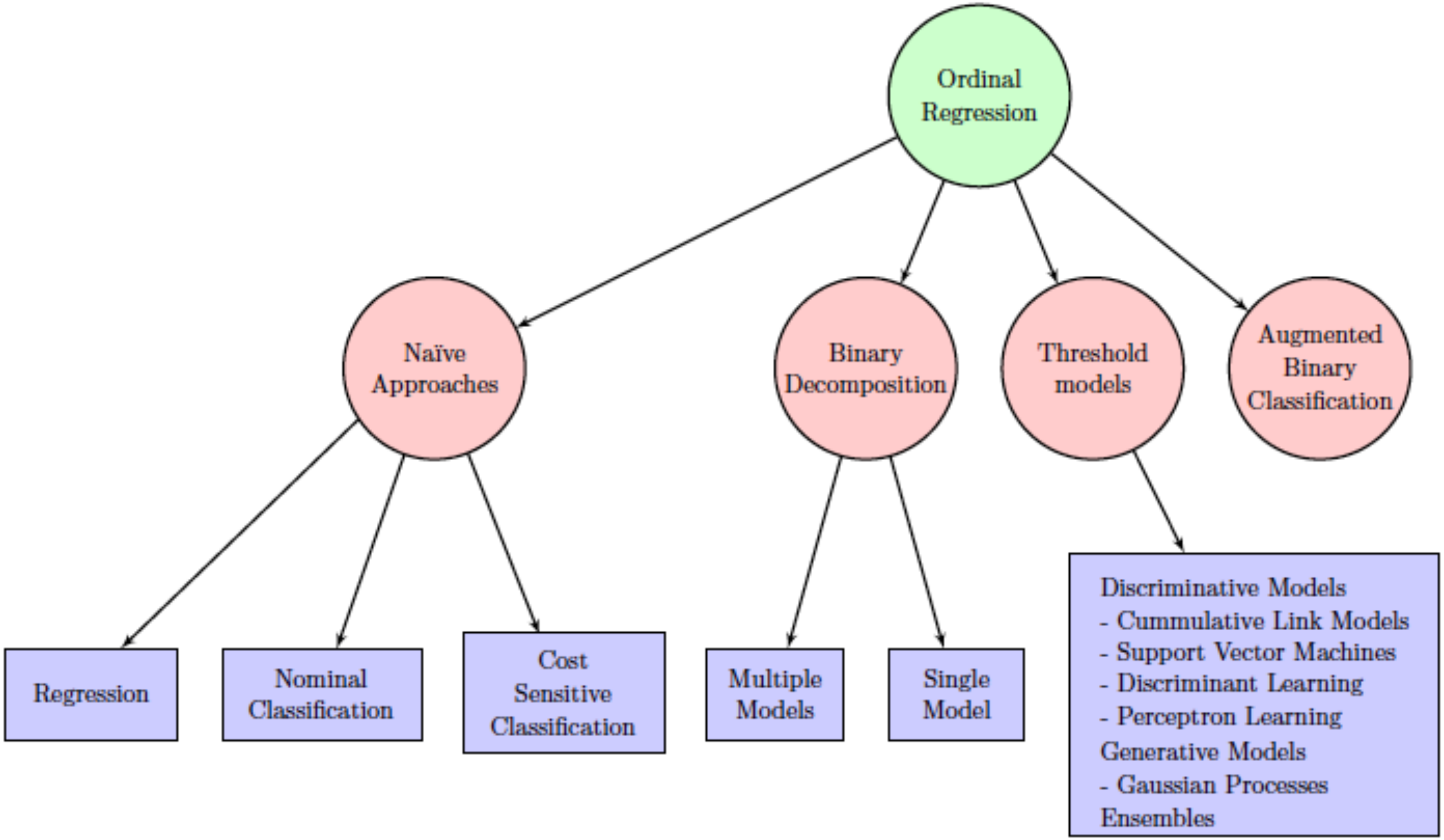
## Mean Zero One Error (MZE)

$$MZE = 1 - Acc = \frac{1}{N}\sum_{i=1}^{N} [\![ y_i^* \neq y_i ]\!]$$

## Mean Absolute Error (MAE)

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

# TAXONOMÍA


Ordinal Regression
Naïve Approaches
Binary Decomposition
Threshold models
Augmented Binary Classification
Regression
Nominal Classification
Cost Sensitive Classification
Multiple Models
Single Model
Discriminative Models
- Cummulative Link Models
- Support Vector Machines
- Discriminant Learning
- Perceptron Learning
Generative Models
- Gaussian Processes
Ensembles

## 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:

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

  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 $f : X \to \mathbb{R}$,
- together with a set of thresholds $b^1 \leq \cdots \leq b^{J-1}$
- The rule for predicting new samples is:

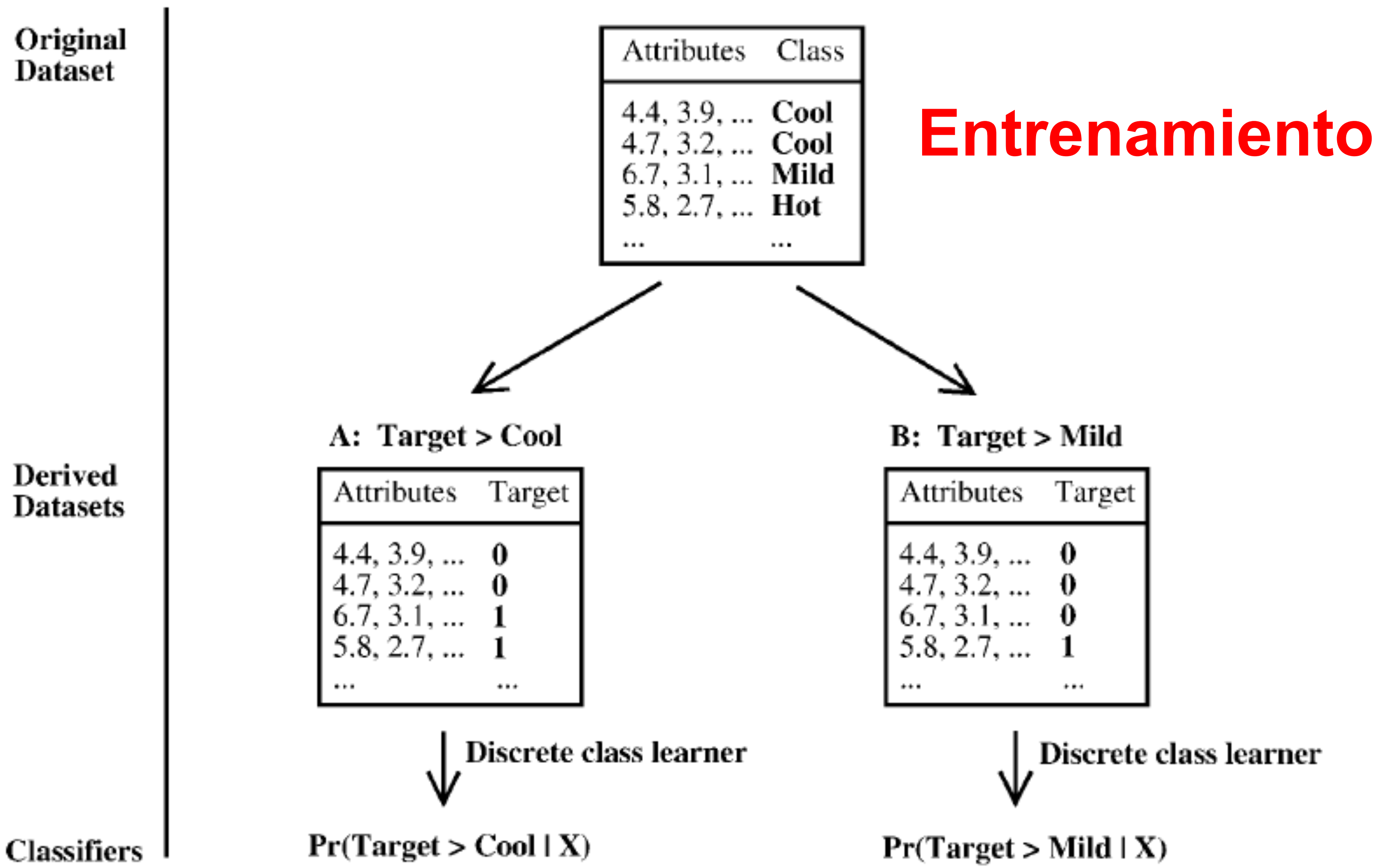
$$g(\mathbf{x}) = \begin{cases} c_1, & \text{if } f(\mathbf{x}) \leq b^1 \\ c_2, & \text{if } b^1 < f(\mathbf{x}) \leq b^2 \\ \dots & \\ c_J, & \text{if } f(\mathbf{x}) > b^{J-1} \end{cases} \tag{7}$$

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

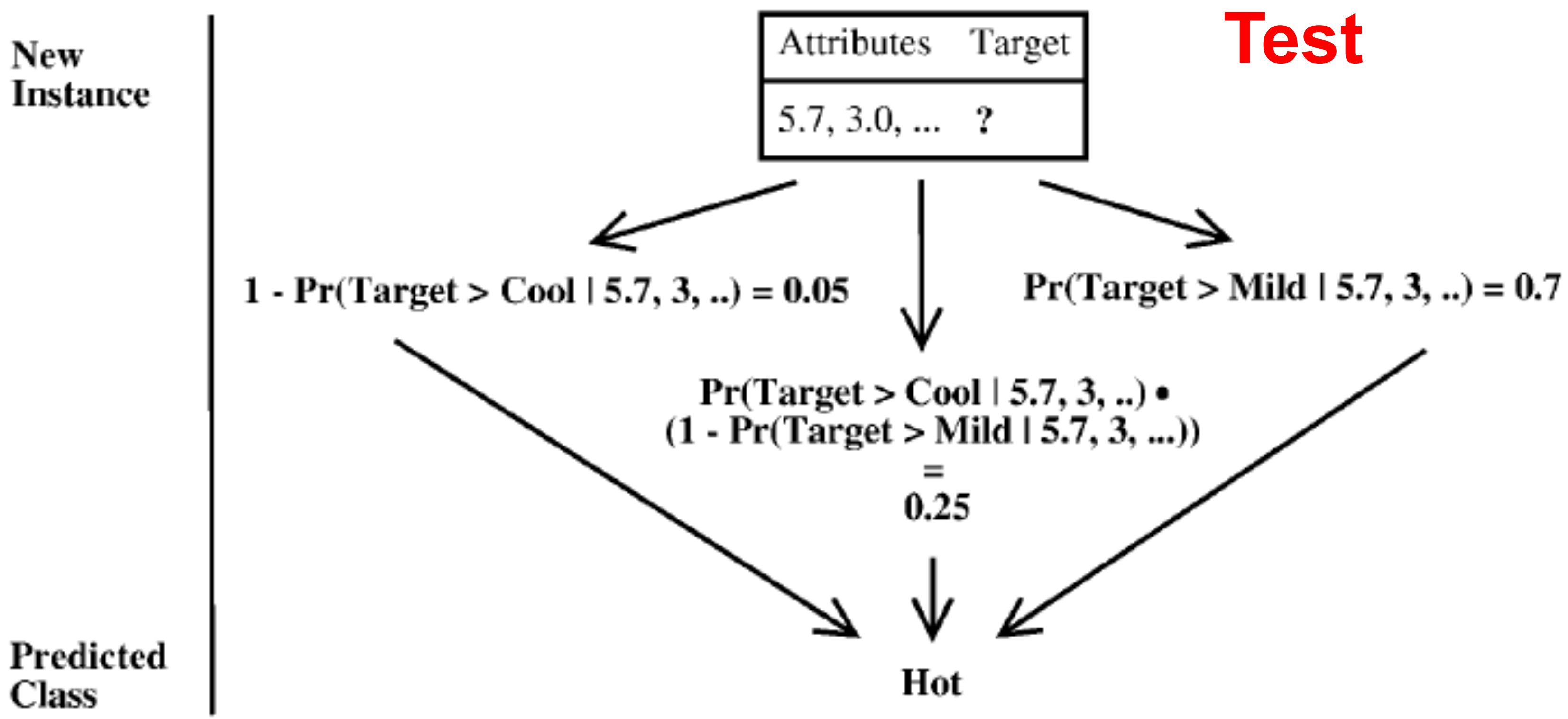
# Multipe Model for Ordinal Classification

- 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

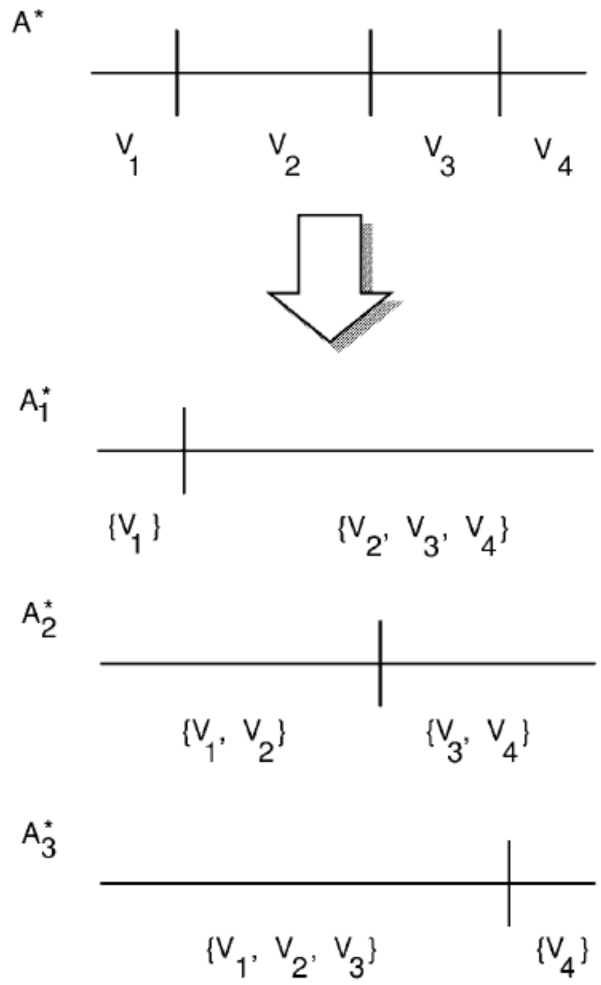


# C4.5 para clasificación Ordinal



# 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

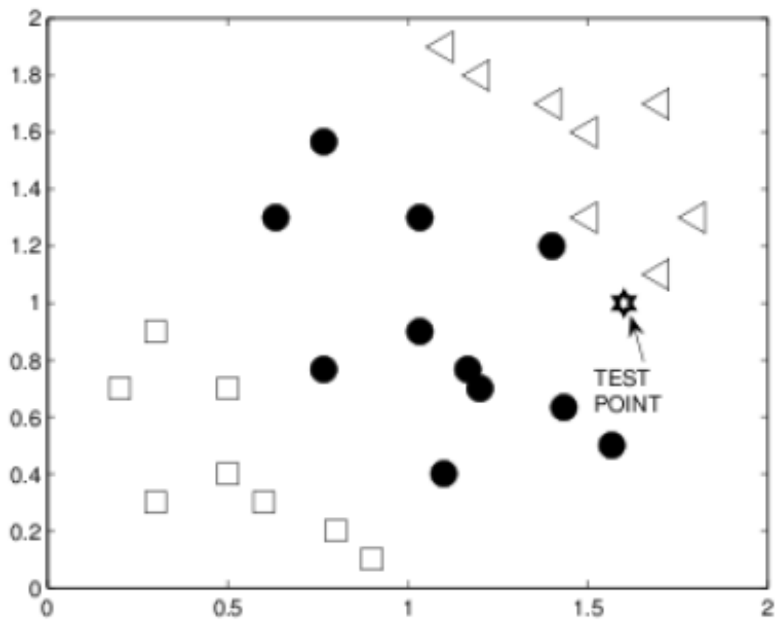
- El entrenamiento comienza generando los nuevos data sets, hasta *k – 1* problemas binarios. Cada uno contiene los mismos valores de atributos de entrada, pero la clase cambia a binaria dependiendo de la desigualdad que tiene cada data set asociada.
- Para predecir el valor clase de una instancia no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

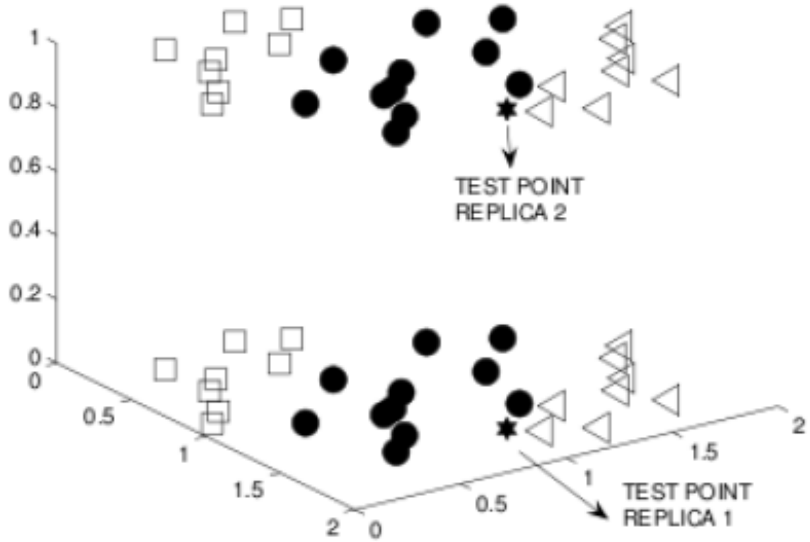
$$Pr(V_1) = 1 - Pr(Target > V_1)$$
$$Pr(V_i) = Pr(Target > V_{i-1}) \times (1 - Pr(Target > V_i)) \quad ,1 < i < k$$
$$Pr(V_k) = Pr(Targe > V_{k-1})$$

- 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

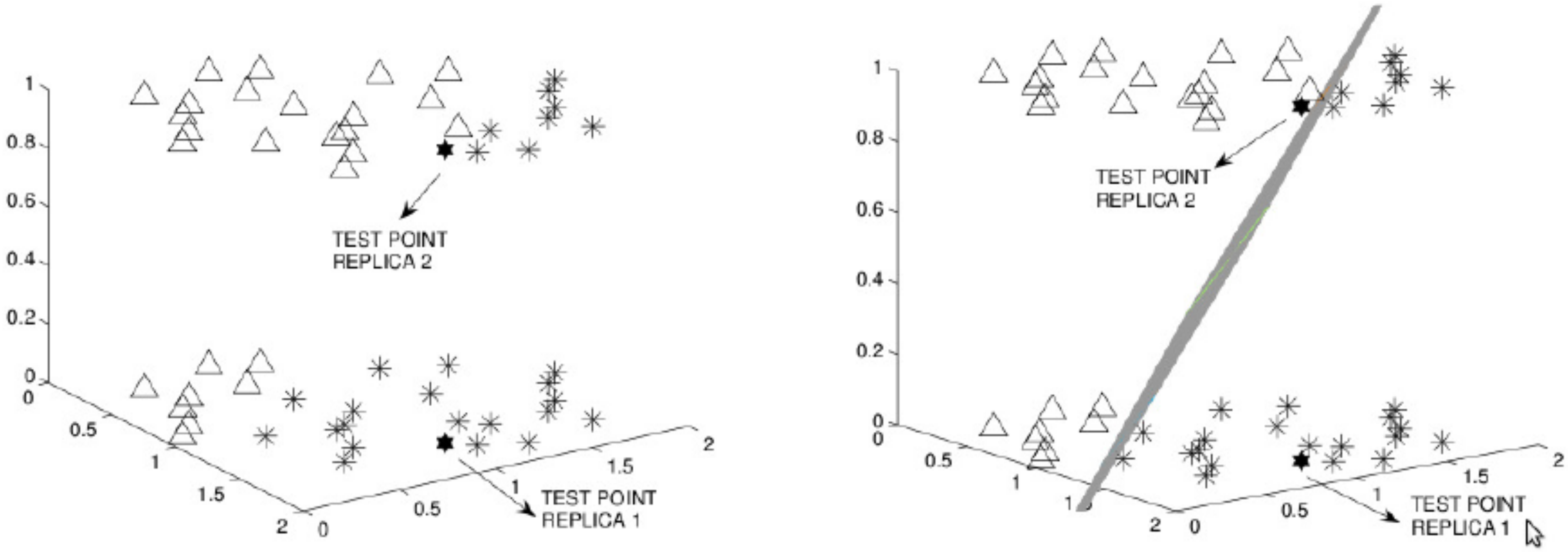
Cardoso & da Costa: Add additional dimensions and replicate your data points:



# 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

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

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

$$X \geq Y \quad si \;\; x_i > y_i \quad || \;\; x_i = y_i \quad \forall i = 1, 2, \dots, n$$

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

$$X \leq Y \quad si \;\; x_i < y_i \quad || \;\; x_i = y_i \quad \forall i = 1, 2, \dots, n$$

# Definición de monotonía (II)

Sea $X = (X, C_x)$ e $Y = (Y, C_x)$ 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:

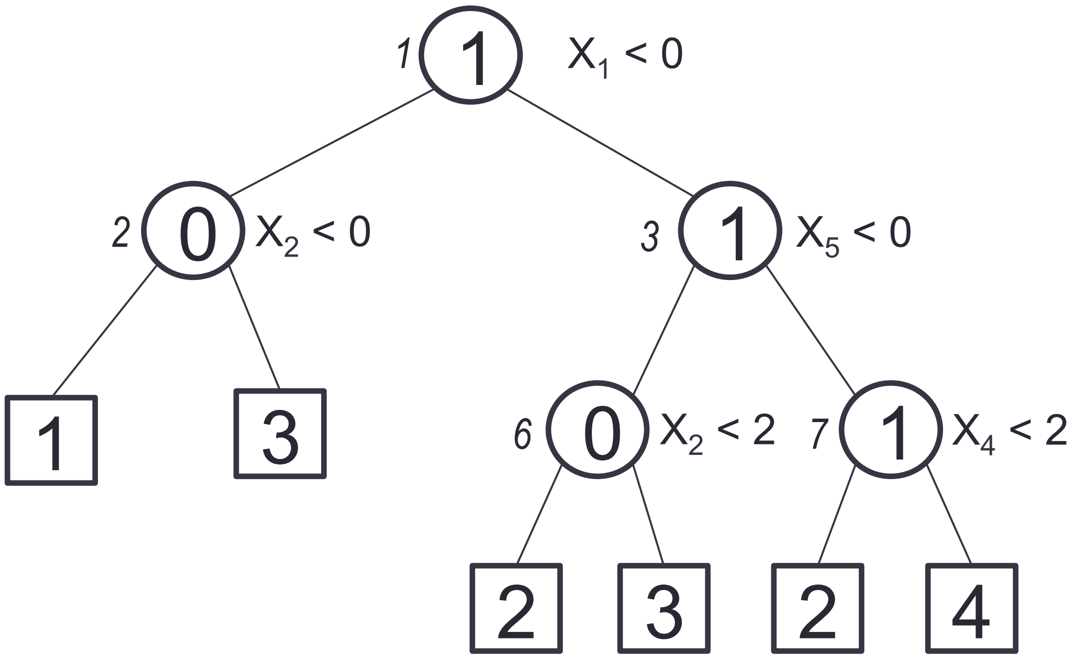
$$X < Y \quad \wedge \quad C_x > C_y \quad ||$$

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

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

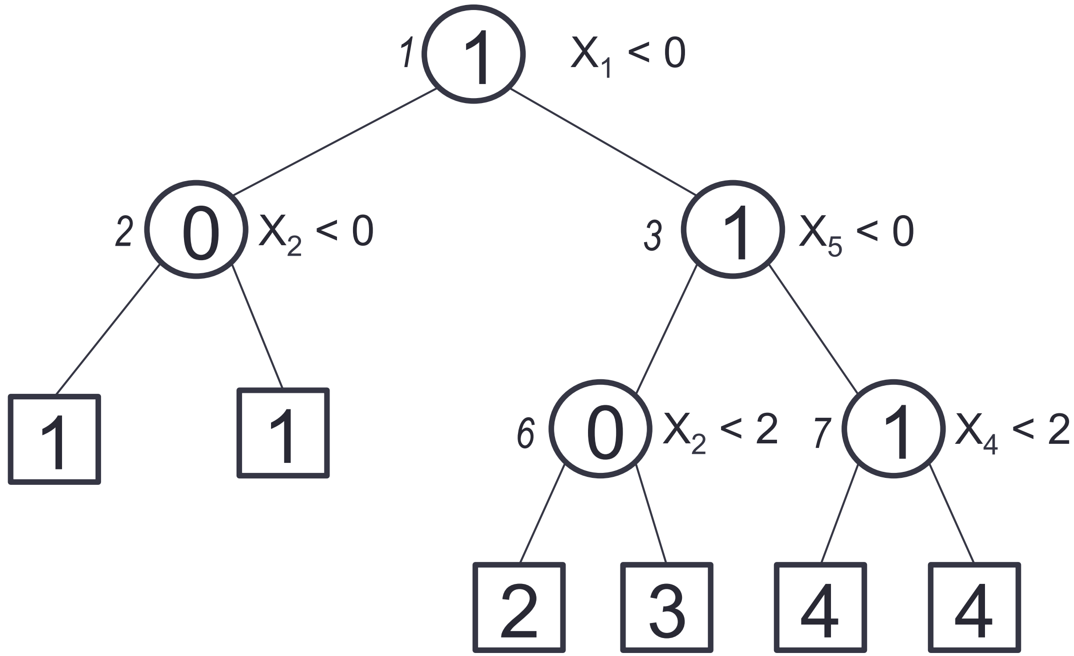
**Dos parejas atributos-clase = $(X, C_x)$ e $(Y, C_x)$ son monotónicas con respecto a la otra si no cumple cualquiera de las condiciones de la definición anterior.**

# Árboles de Decisión Monotónicos




ÁRBOL DE DECISIÓN **NO MONOTÓNICO**

# Árboles de Decisión Monotónicos




ÁRBOL DE DECISIÓN **MONOTÓNICO**

# CLASIFICACIÓN MONOTÓNICA: Datasets

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

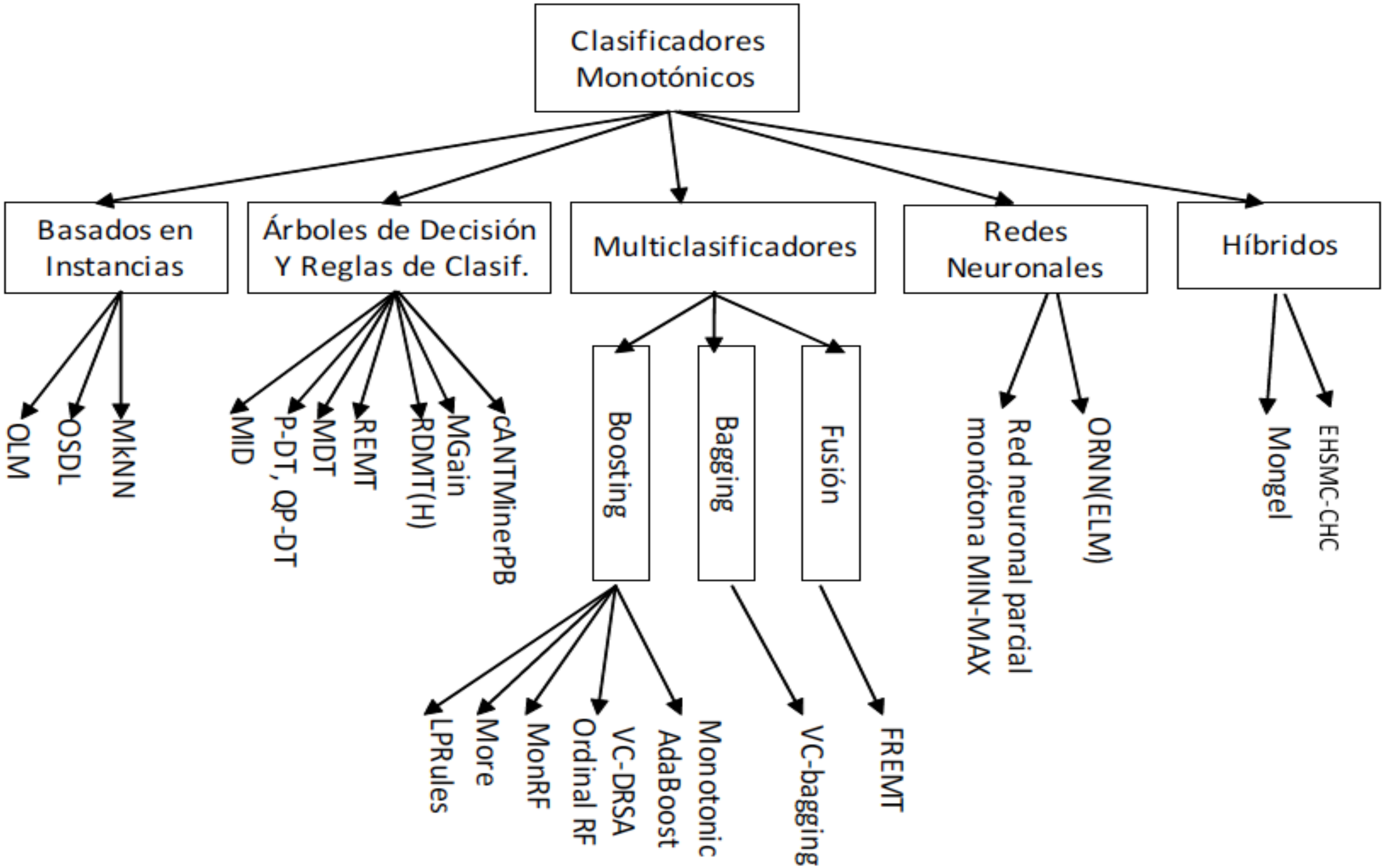
| Attribute | Type | Sign |
|---|---|---|
| mpg | continuous | target |
| cylinders | multi-valued discrete | − |
| displacement | continuous | − |
| horsepower | continuous | − |
| weight | continuous | − |
| acceleration | continuous | + |
| model year | multi-valued discrete | + |
| origin | multi-valued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

# CLASIFICACIÓN MONOTÓNICA: Datasets Clásicos

| Dataset | #Instancias | #Atributos | #Clases |
|---|---|---|---|
| Employee Rejection Acceptance (ERA) | 1000 | 4 | 9 |
| Employee Selection (ESL) | 488 | 4 | 9 |
| Lectures Evalution (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.

# CLASIFICACIÓN MONOTÓNICA: Taxonomía



# 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.

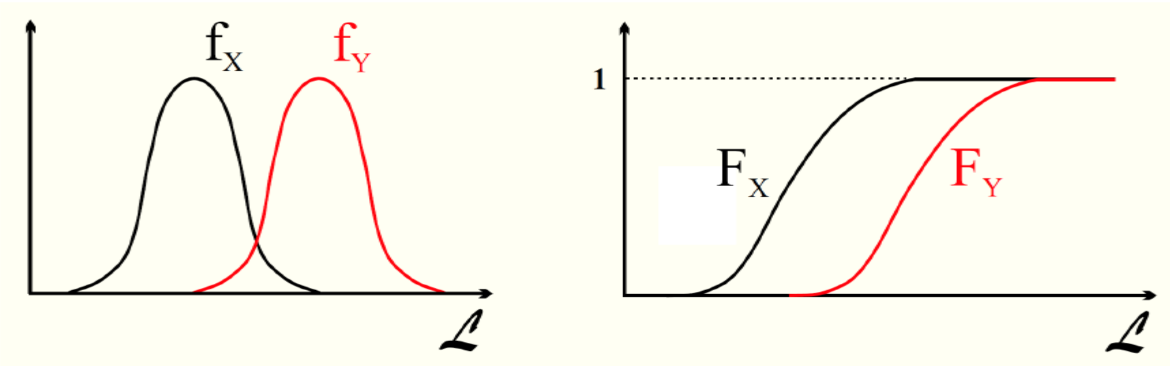
# Ordinal Learning Model (OLM)

- 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 Stochastic Dominance Learner (OSDL)

- Ordinal Stochastic Dominance Learner (OSDL) es un clasificador basado en instancias que asegura la monotonía entrenado 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.



# 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**

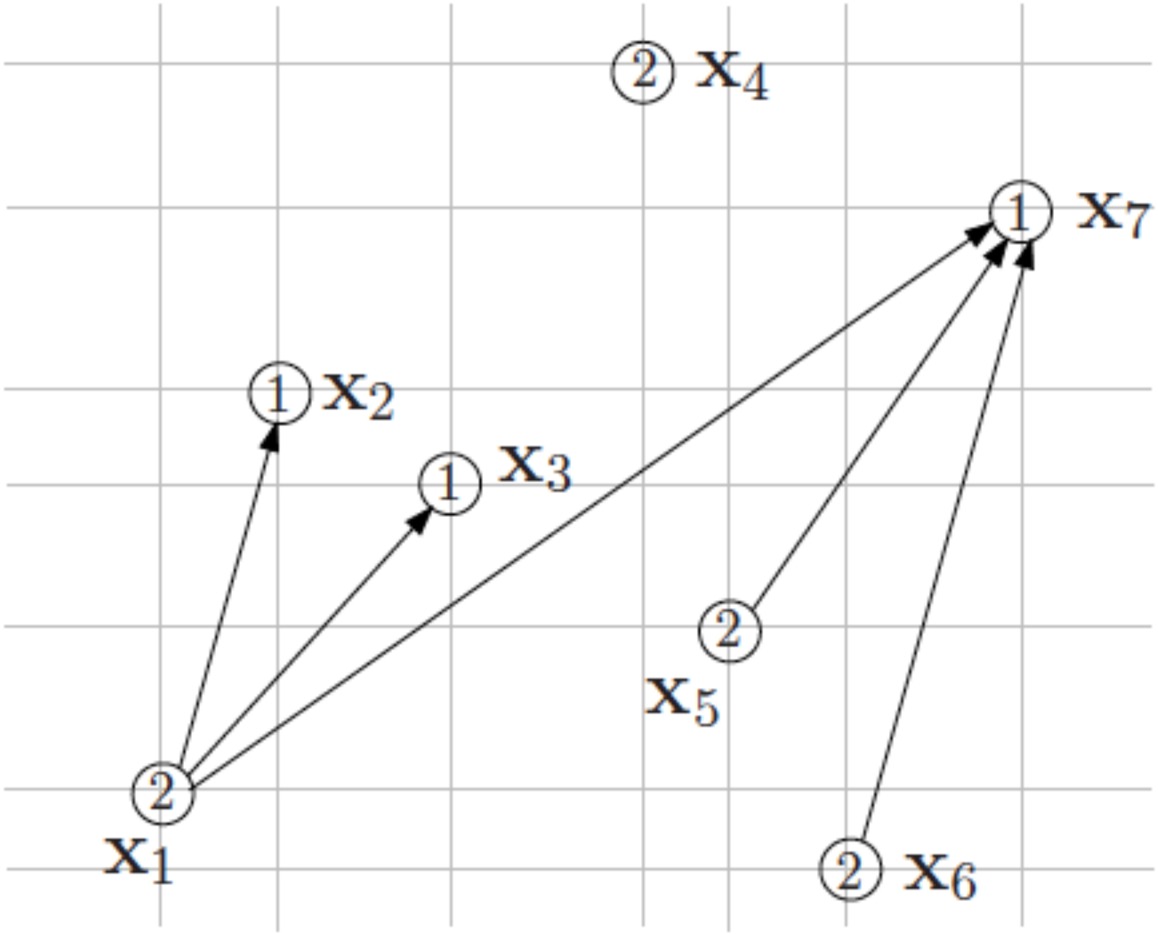



**Fig. 1.** Example Monotonicity Violation Graph

# Monotonic k-Nearest Neighbors

## Primera Etapa

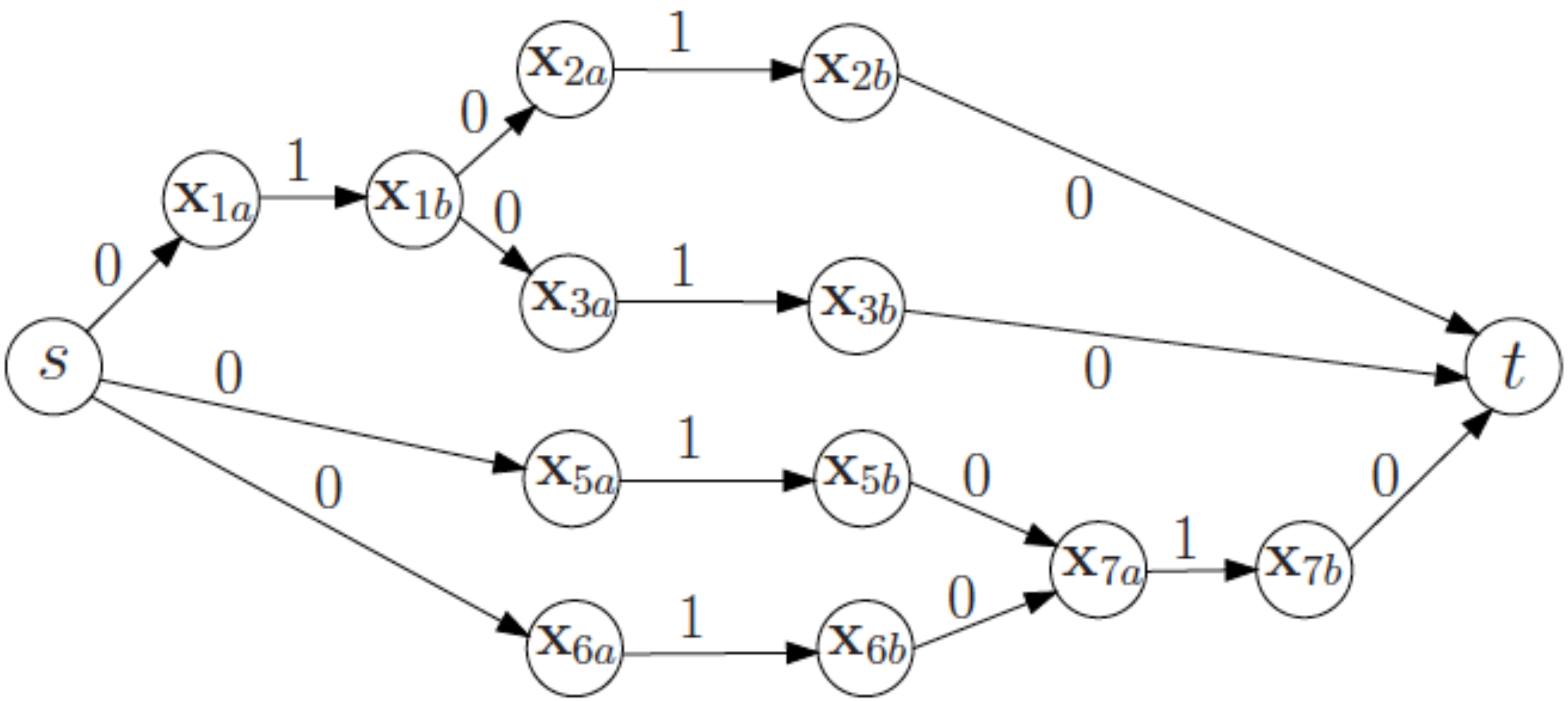



**Fig. 2.** Transportation network based on the Monotonicity Violation Graph in Figure 1

# 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

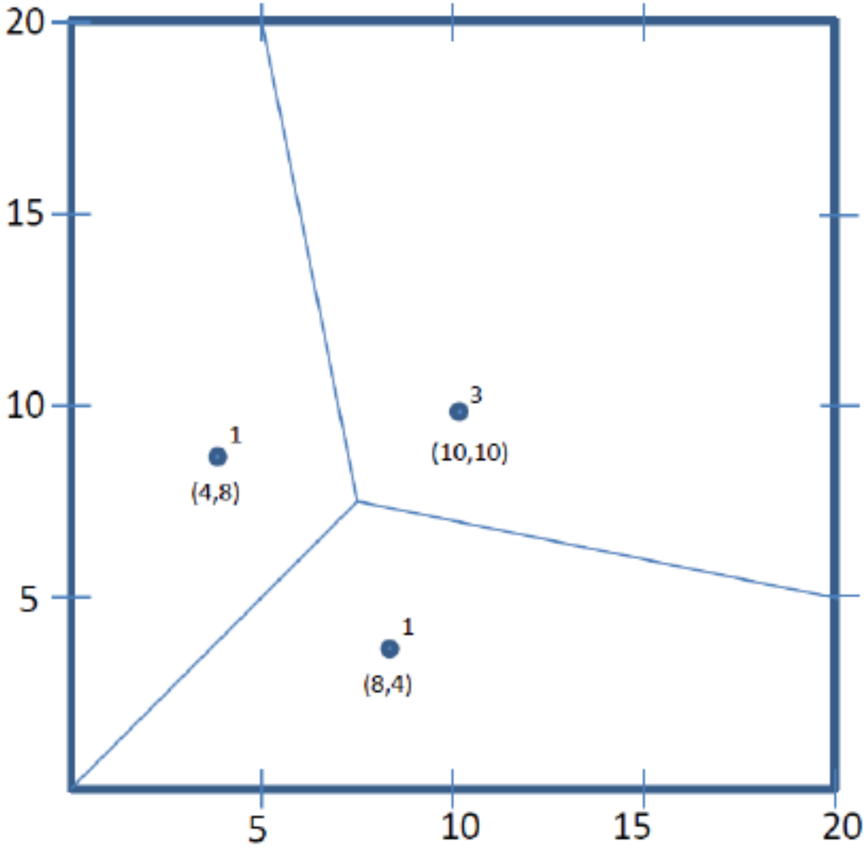
**Segunda Etapa, Variantes:**

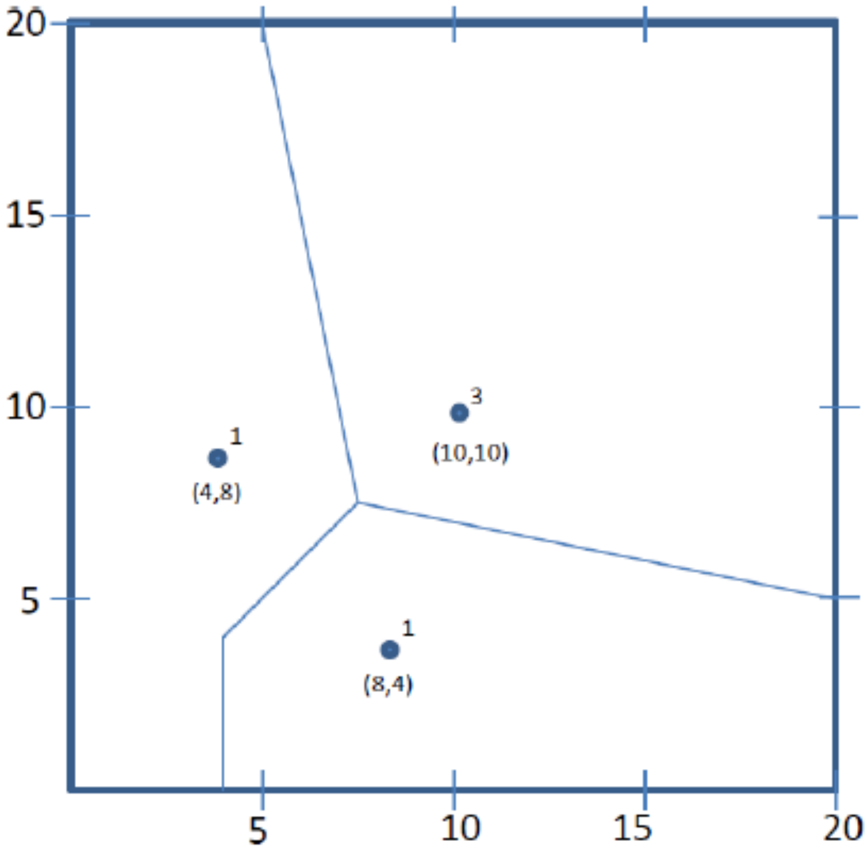
- Coger los *k* vecinos cercanos de $x_0$ de *D* y predecir la etiqueta desde [$y_{min}$, $y_{max}$] que más frecuente aparece entre esos *k* vecinos. Si ninguna de las *k* etiquetas se permite, escoger una al azar entre [$y_{min}$, $y_{max}$].

- 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:**



# 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

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

## 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.**

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

# Monotonic Induction of Decision Tree (MID)

- **DEFINICIÓN 1**: Sea $X = (x_1, x_2, \ldots, x_n)$ e $Y = (y_1, y_2, \ldots, y_n)$ 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:

$$
\begin{array}{llll}
X = Y & \text{if } x_i = y_i & \forall i = 1, 2, \ldots, n & \\
X > Y & \text{if } x_i > y_i & \forall i = 1, 2, \ldots, n & \\
X \geq Y & \text{if } x_i > y_i & \text{OR} \quad x_i = y_i & \forall = 1, 2, \ldots, n \\
X < Y & \text{if } x_i < y_i & \forall i = 1, 2, \ldots, n & \\
X \leq Y & \text{if } x_i < y_i & \text{OR} \quad x_i = y_i & \forall i = 1, 2, \ldots, n
\end{array}
$$

# 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:

$$
\begin{aligned}
&X \leq Y \wedge C_x > C_y \quad \text{OR} \\
&X \geq Y \wedge C_x < C_y \quad \text{OR} \\
&X = Y \wedge C_x \neq C_y
\end{aligned}
$$

- **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(income) = 0.667 (2/3*1 + 1/3*0)
  - E(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-monotonicas con respecto a cada otra en el mismo árbol.
- Para encontrar este índice en un árbol de *k* ramas, construimos una matrix *M*, *k x 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 $a_1$, $a_2$, …, $a_v$, se define como:

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

# 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.

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

- **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.

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

# 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.

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

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

# Monotonic Induction of Decision Tree (MID)

## Construyendo árboles monotónicos precisos

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

$$
\begin{aligned}
E_{\text{income}} &= 0.667 \text{ bits} \\
I_{\text{income}} &= A_{\text{income}} = 0 \\
T_{\text{income}} &= 0.667\ (0.667 + 2 * 0)
\end{aligned}
$$

$$
\begin{aligned}
E_{\text{assets}} &= 0 \text{ bits} \\
I_{\text{assets}} &= 0.333\ (2/(3^2 - 3)) \\
A_{\text{assets}} &= 0.630\ (-\log_2 0.333)^{-1} \\
T_{\text{assets}} &= 1.260\ (0 + 2 * 0.630)
\end{aligned}
$$

$$
\begin{aligned}
E_{\text{credit history}} &= 0.667 \text{ bits} \\
I_{\text{credit history}} &= A_{\text{credit history}} = 0 \\
T_{\text{credit history}} &= 0.667.
\end{aligned}
$$

# Monotonic Induction of Decision Tree (MID)

## Construyendo árboles monotónicos precisos

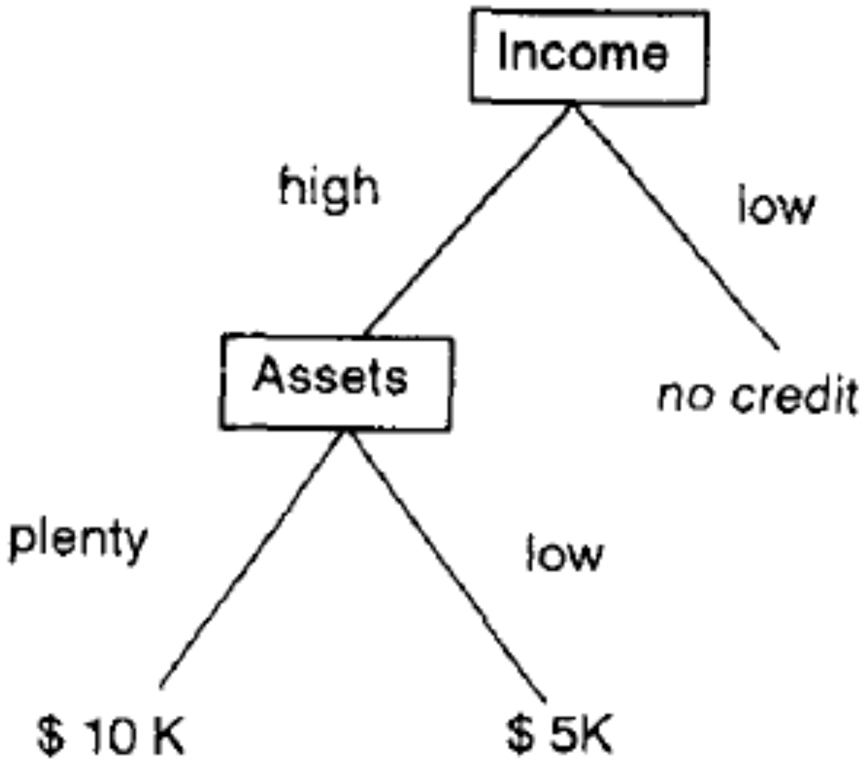
- Los scores de ambigüedad total para *income* y *credit history* son los más bajos, y asumimos que el atributo income se selecciona en la primera partición. El score de ambigüedad total de *income* + *assets* se comprueba:

$$E_{\text{income+assets}} = 0 \text{ bits}$$
$$I_{\text{income+assets}} = A_{\text{income+assets}} = 0$$
$$T_{\text{income+assets}} = 0.$$

- Ahora, el árbol obtenido es monotónico.

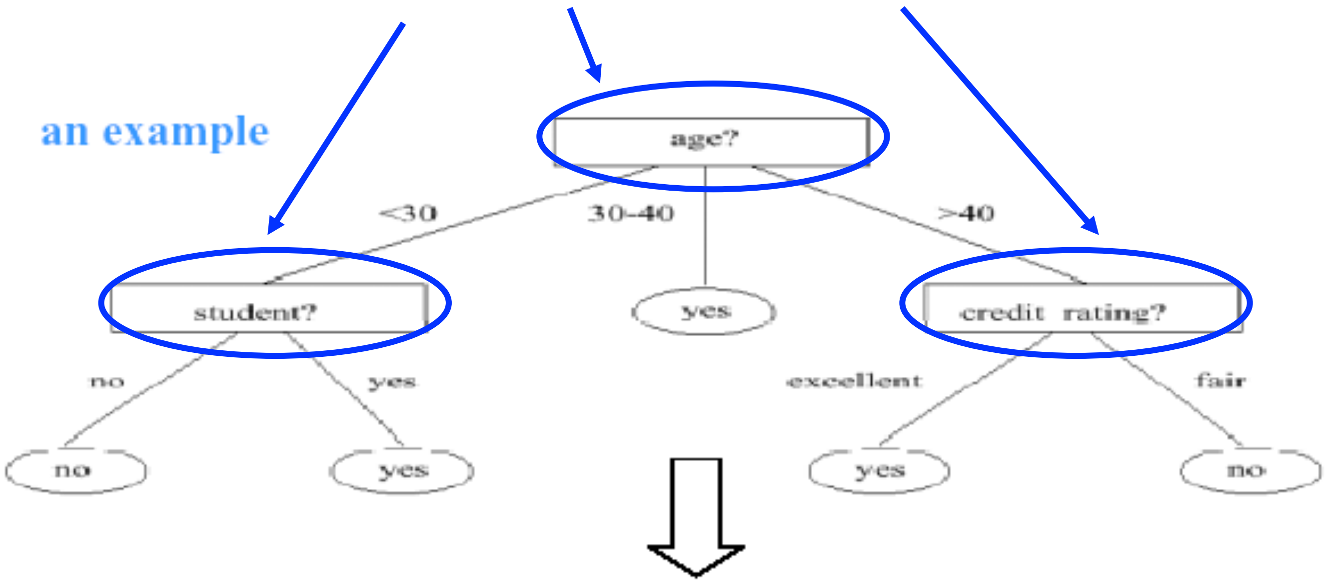
# Monotonic Induction of Decision Tree (MID)

## Construyendo árboles monotónicos precisos



# Rank Entropy Monotonic Tree (REMT)

**La questión 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* THEN *buys_computer* = *no*
IF *age* = "<30" AND *student* = *yes* THEN *buys_computer* = *yes*
IF *age* = "30-40" THEN *buys_computer* = *yes*
IF *age* = ">40" AND *credit_rating* = *excellent* THEN *buys_computer* = *yes*
IF *age* = ">40" AND *credit_rating* = *fair* THEN *buys_computer* = *no*

# Rank Entropy Monotonic Tree (REMT)

## Métrica RMI

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

$a_j$ $c$

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 |

$[x_i]^{\leq}_{a_j}$

$[x_i]^{\leq}_{c}$

# Rank Entropy Monotonic Tree: Otras métricas

## Métrica RSD

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

$a_j$ $c$

**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 |

$[x_i]_{a_j}^{\leq}$

$[x_i]_c^{\leq}$

# Rank Entropy Monotonic Tree: Otras métricas

## Métrica RGD

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

$a_j$ $c$

| 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 |

$[x_i]^{\leq}_{a_j}$

$[x_i]^{\leq}_{c}$

# 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 $DT = < U, C, D >$, $B \subseteq C$, 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 ejemplo 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 $U = \{x_1, x_2, \cdots, x_n\}$ 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

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

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

**Número de elementos**

**La entropía de Shannon se define como**

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

# Rank Entropy Monotonic Tree (REMT)

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

$$RH^{\leq}_{B \cup C}(U) = -\frac{1}{n}\sum_{i=1}^{n} \log \frac{|[x_i]^{\leq}_{B} \cap [x_i]^{\leq}_{C}|}{n}, \qquad (3)$$

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

$$RH^{\geq}_{B \cup C}(U) = -\frac{1}{n}\sum_{i=1}^{n} \log \frac{|[x_i]^{\geq}_{B} \cap [x_i]^{\geq}_{C}|}{n}. \qquad (4)$$

# 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

$$RH^{\leq}_{B|C}(U) = -\frac{1}{n}\sum_{i=1}^{n}\log\frac{|[x_i]^{\leq}_{B} \cap [x_i]^{\leq}_{C}|}{|[x_i]^{\geq}_{C}|}, \tag{5}$$

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

$$RH^{\geq}_{B|C}(U) = -\frac{1}{n}\sum_{i=1}^{n}\log\frac{|[x_i]^{\geq}_{B} \cap [x_i]^{\geq}_{C}|}{|[x_i]^{\geq}_{C}|}. \tag{6}$$

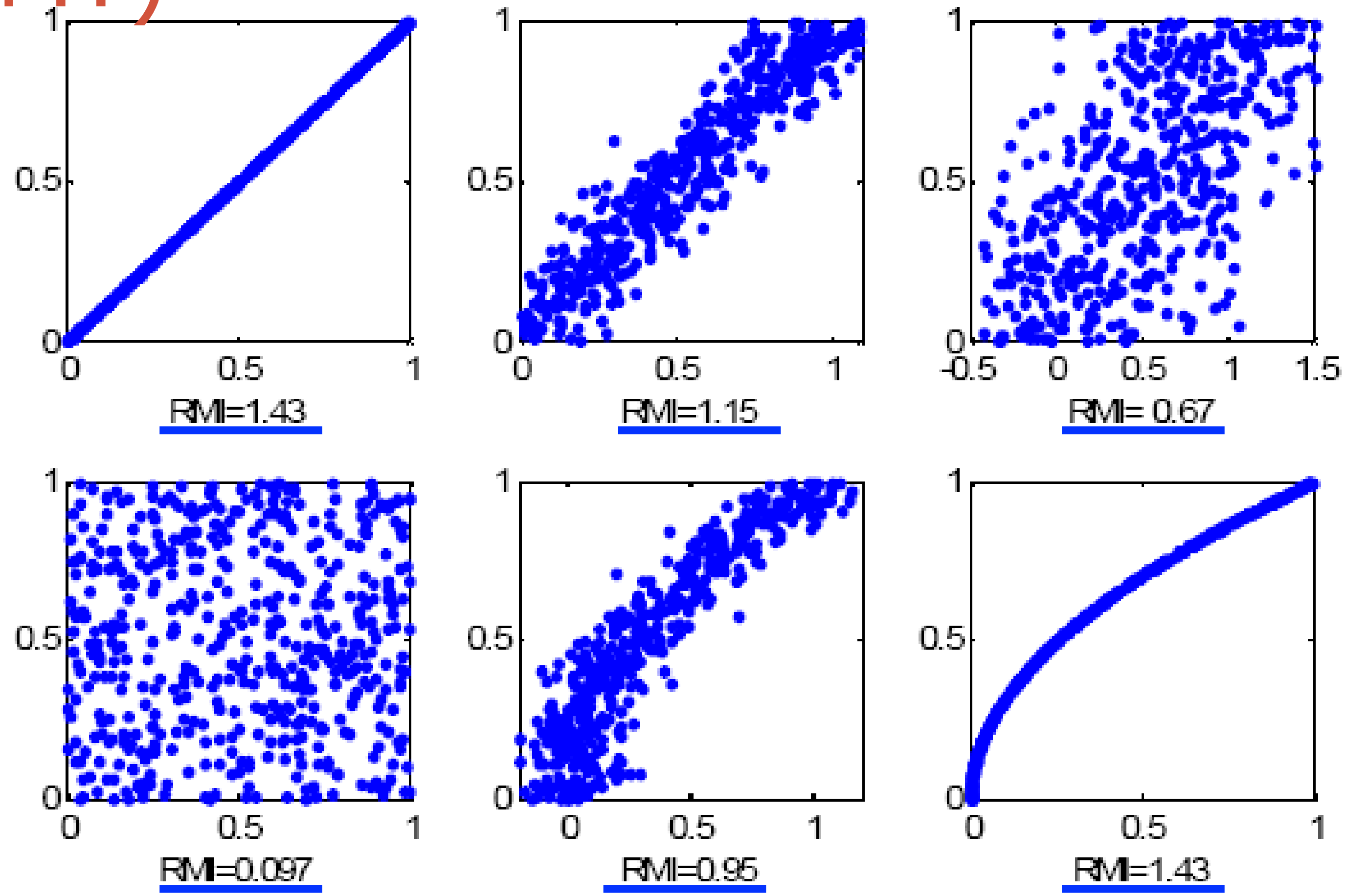
# Rank Entropy Monotonic Tree (REMT)

**Definition 8.** Given $DT = <U, A, D>$, $B \subseteq A$, $C \subseteq A$. The ascending rank mutual information (ARMI) of the set $U$ regarding $B$ and $C$ is defined as

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

**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.**

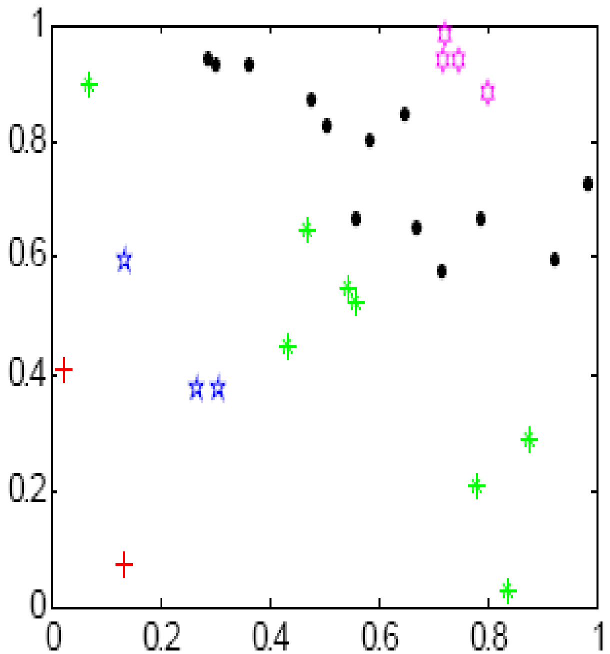
# Rank Entropy Monotonic Tree (REMT)




**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.**

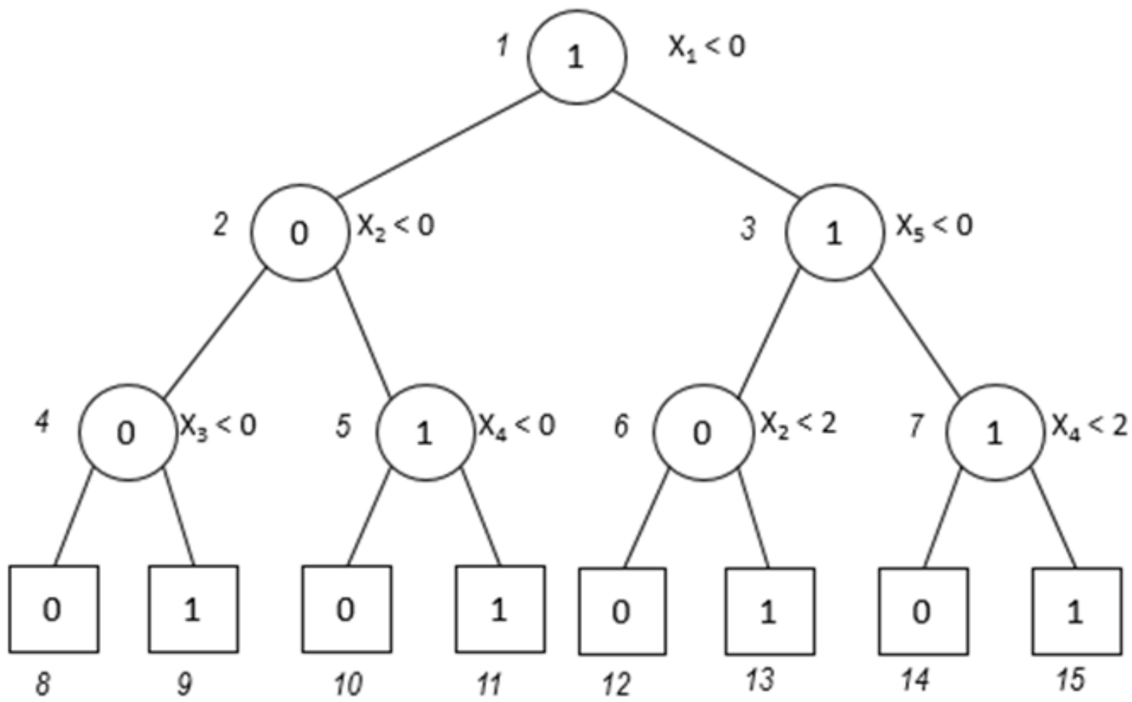
**2. Seleccionar el atributo con la máxima información mutua y dividirlo de acuerdo a sus valores.**

**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 l node sea puro.**

# Poda Monotónica

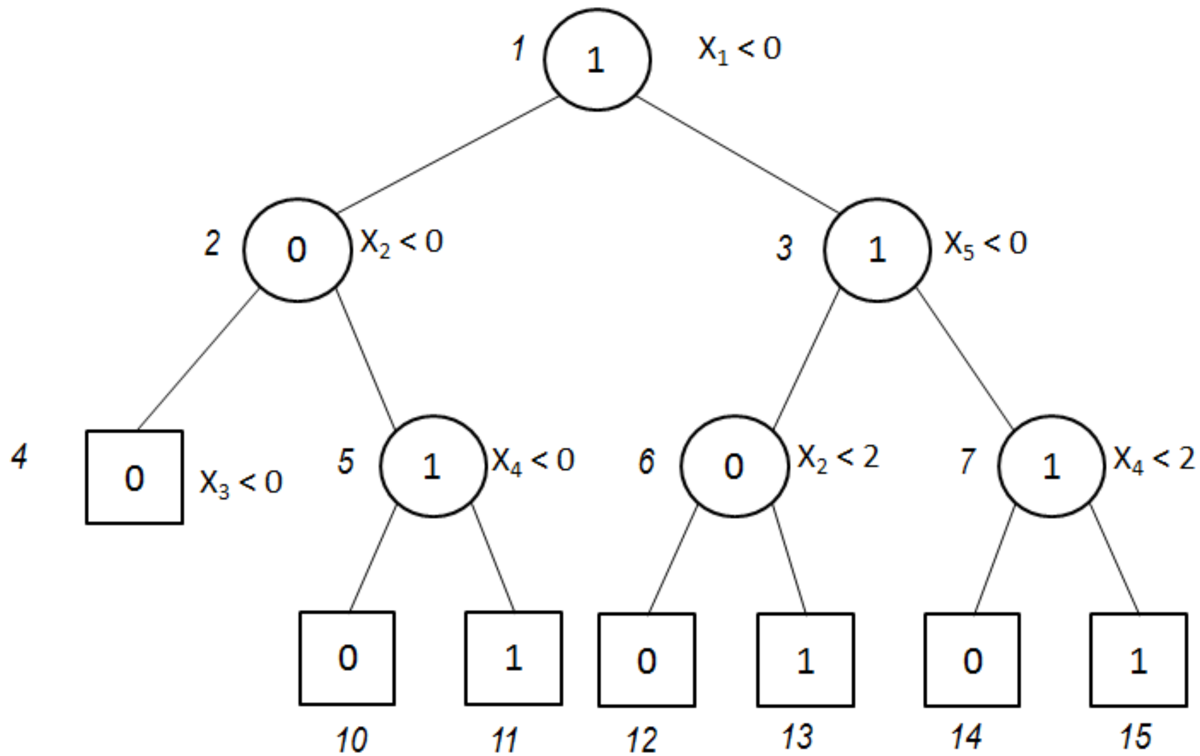
- Hacer Poda es necesario para paliar los efectos del sobreaprendizaje
- No se pueden aplicar las técnicas de poda de árboles de decisión clásicos
- Métricas monotónicas no aseguran árboles monotónicos
- Se han propuesto dos algoritmos de poda:
    - Mayor Padre no monotónico (MNP)
    - Mejor arreglo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

# Poda monotónica (BF)




PAREJAS NO MONOTÓNICAS: **[9, 10], [9, 12], [9, 14], [11, 12], [11, 14], [13,14]**

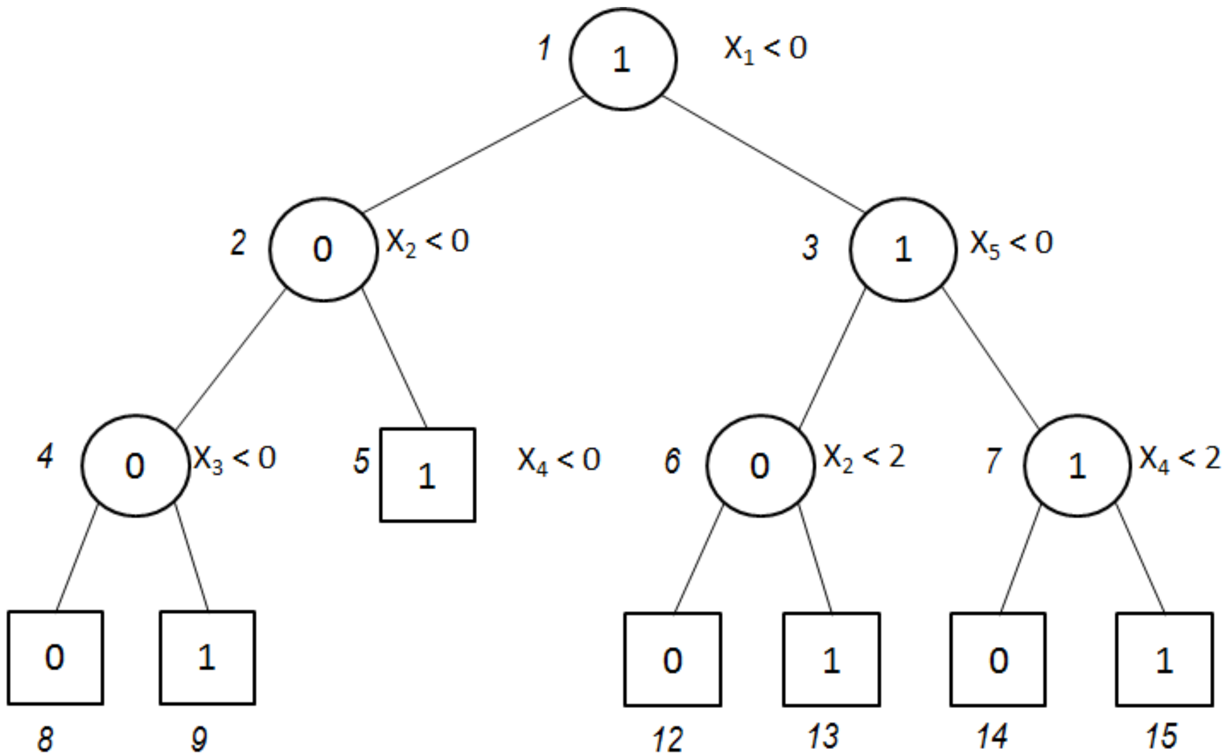
# Poda monotónica (BF)




PAREJAS NO MONOTÓNICAS : **[11, 12], [11, 14], [13,14]**

Podando NODO 4 quedan **3 parejas no monotónicas**

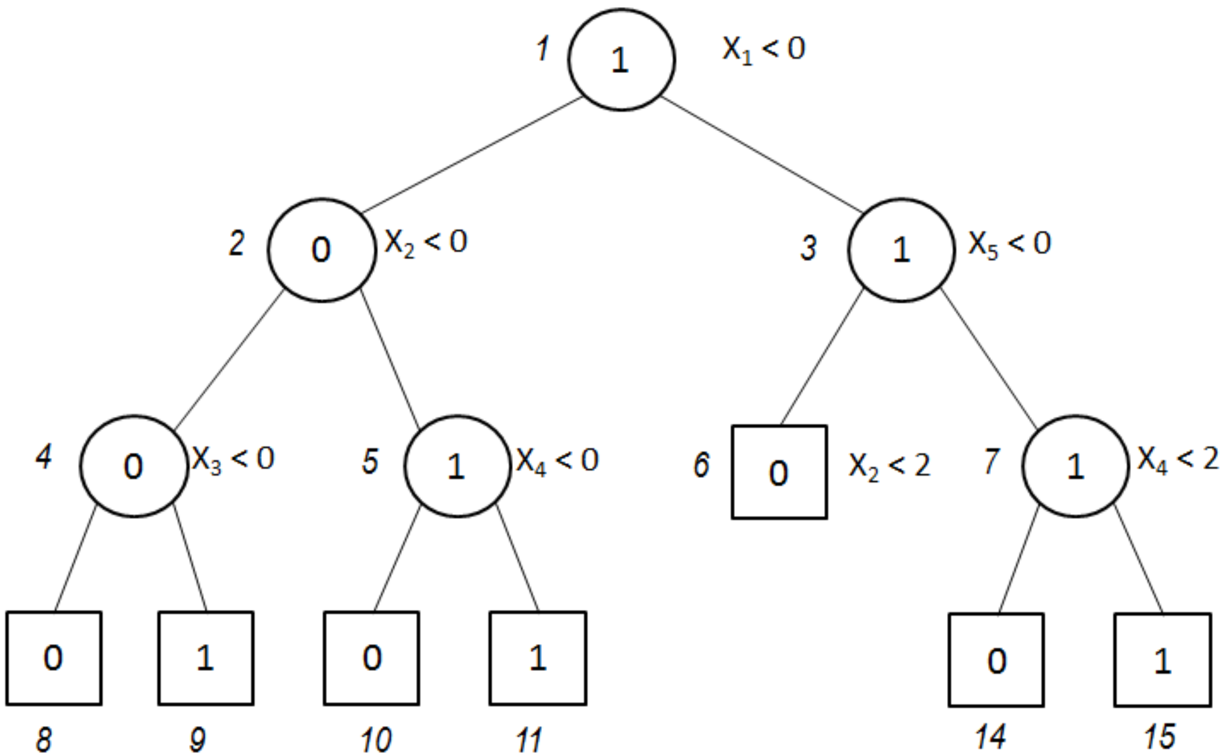
# Poda monotónica (BF)




PAREJAS NO MONOTÓNICAS **: [9, 12], [9, 14], [5, 12], [5, 14], [13,14]**

Podando NODO 5 quedan **5 parejas no monotónicas**

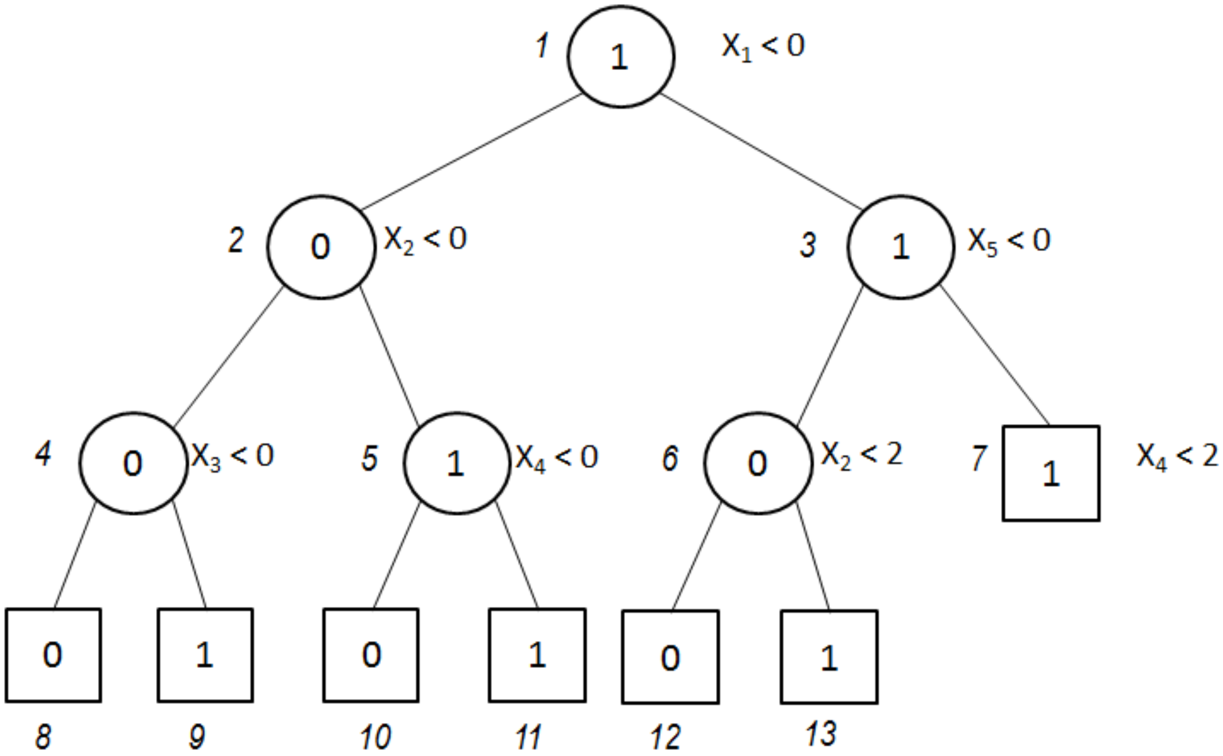
# Poda monotónica (BF)




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).

# 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 procedure, $S$ - 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.

# 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.

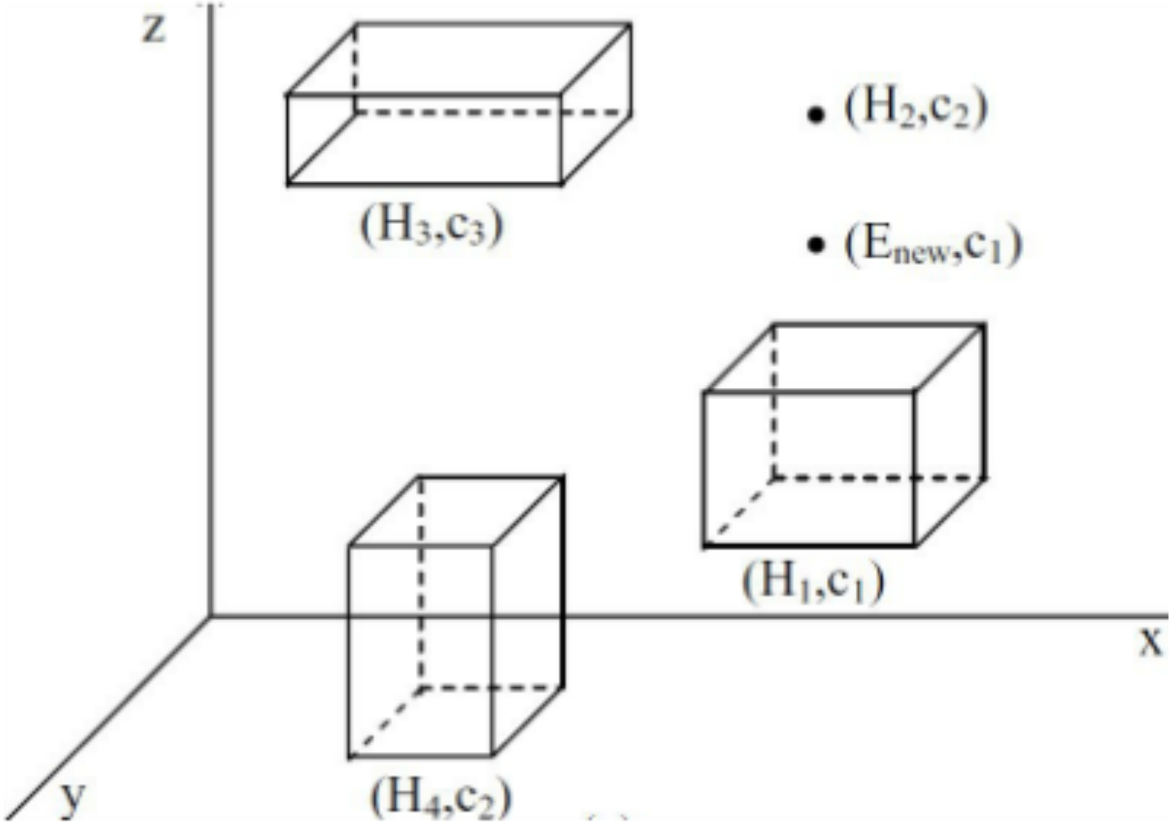
El pilar central de NGE es el proceso de correspondencia, donde se calcula una puntuación de correspondencia dada por:

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

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

# Nested Generalized Examples (NGE)

## GENERALIZACIÓN DE HIPERRECTÁNGULOS



# MonGEL

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

Etapa de Inicialización del conjunto de Reglas

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
                $R''_i = Generalize(R_i, R'_i)$.
                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.

Etapa de Generalización del conjunto de Reglas

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 Eliminación de las reglas no monotónicas

# MonGEL

## ELIMINACIÓN DE REGLAS NO MONOTÓNICAS

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

$$V_1=(2\ \ 3\ \ 4\ \ 2\ \ 3)$$

37

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

$$V_2=(1\ \ 2\ \ 1\ \ 2)$$

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

$$V_3=(1\ \ 0\ \ 0)$$

38

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

39

$$V_4=(0\ \ 0)$$

# 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**

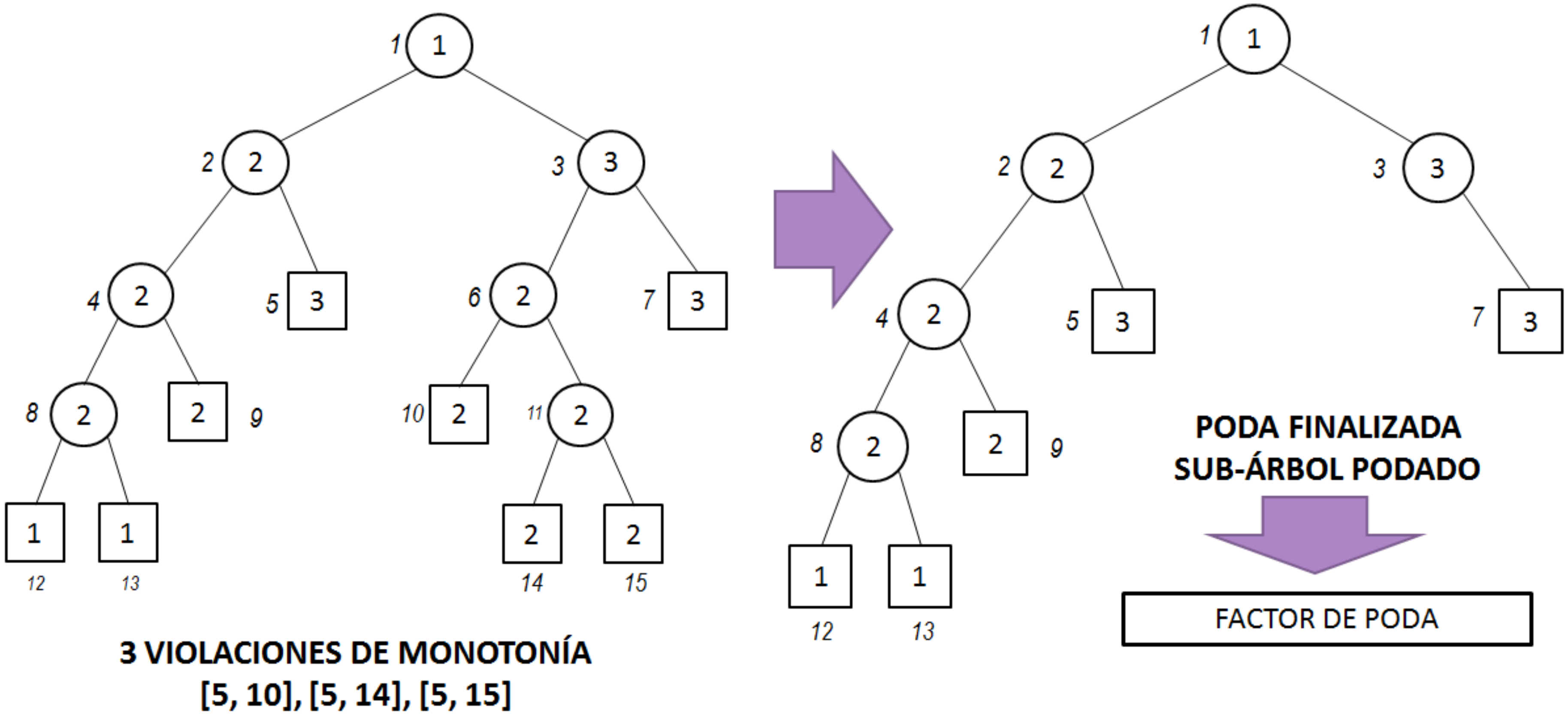
↓

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% |
| | Accu: 19.4% | Accu: 46.9% | Accu: 44.3% | Accu: 46.6% |
| RSD | Hojas: 2 | Hojas: 20 | Hojas: 3 | Hojas: 5 |
| | INM: 0% | INM: 0% | INM: 0% | INM: 0% |
| | Accu: 12.4% | Accu: 50.4% | Accu: 30% | Accu: 48% |
| RMI | Hojas: 6 | Hojas: 10 | Hojas: 63 | Hojas: 18 |
| | INM: 0% | INM: 0% | INM: 0% | INM: 0% |
| | Accu: 21.4% | Accu: 36.8% | Accu: 57.9% | Accu: 53.2% |
| MID | 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.**

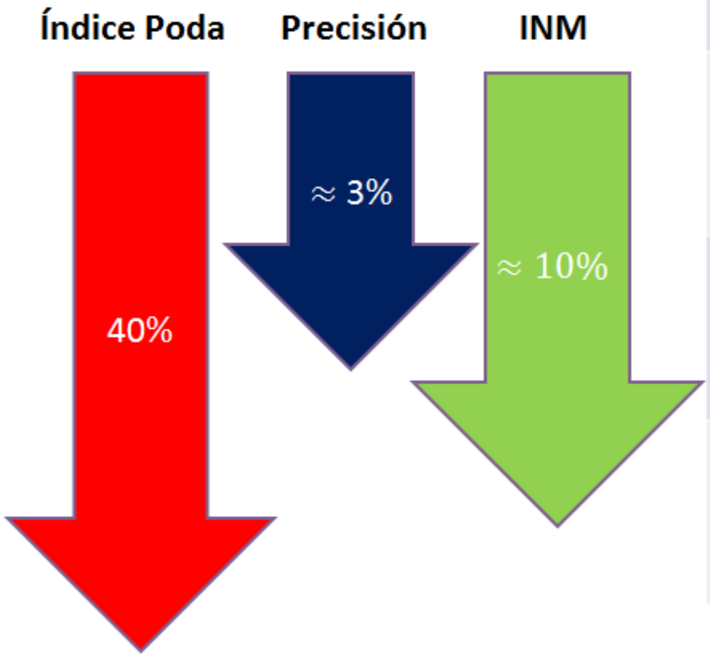
**Se marcan los resultados donde INM es más elevado.**

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

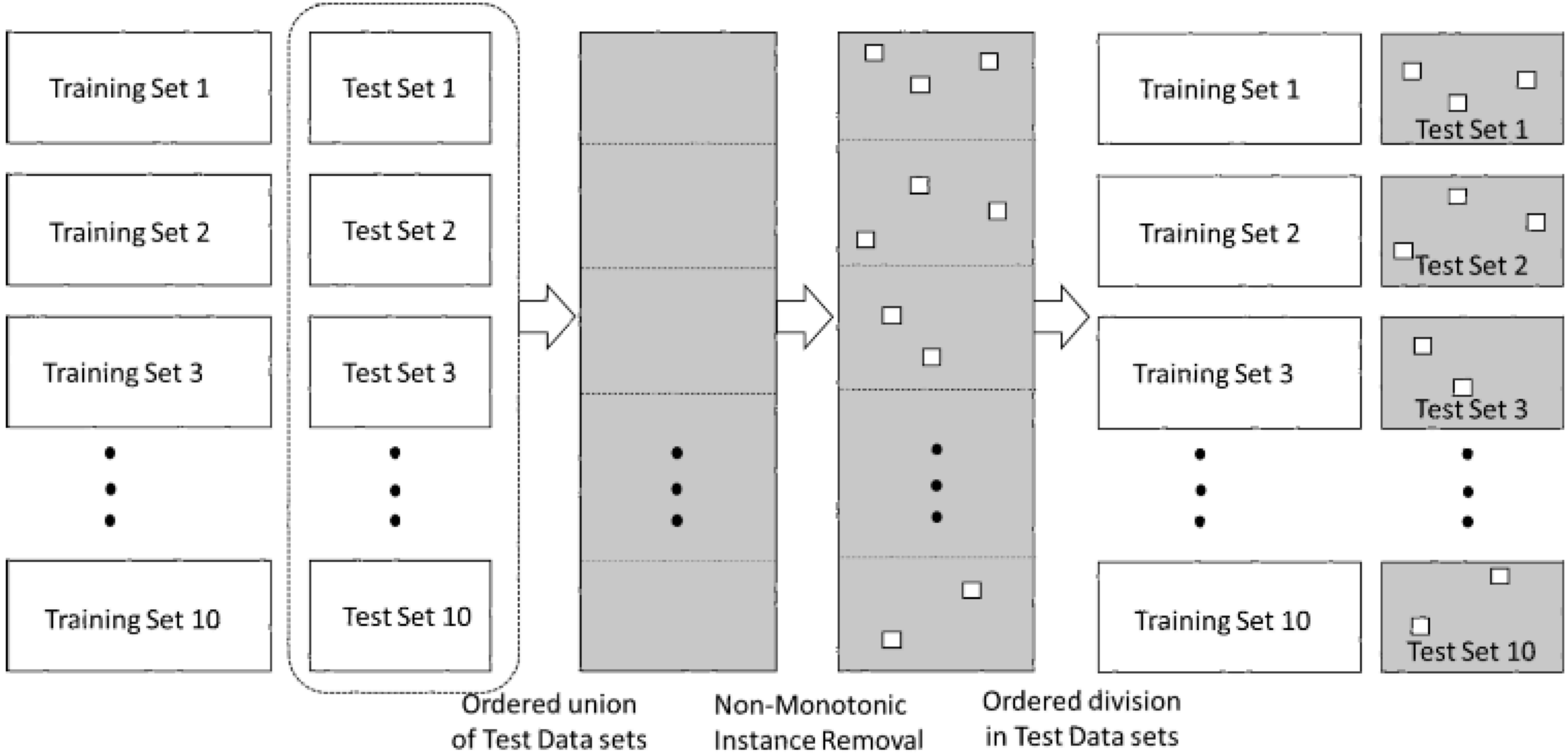
**RMI mejor** que resto de métricas

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

## MEDIDAS MONOTÓNICAS: MMAE y MACC



# THE FUTURE

- **Monotonic Imbalanced Classification:**
- Ordinal classification is frequently associated to imbalanced classes.
- This hybridization makes sense.
- Applications in bank loans, lecturer evaluation, etc.
- **1 result found:** CCCS: A top-down associative classifier for imbalanced class distribution. ACM SIGKDD 2006. (Nitesh)

# 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, radium, 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, Aggresive Behavior, Sleep Problems, etc…

- My view is that it makes sense (many applications).