

# T5-IN.pdf



patrivc



Apuntes Variados



4º Grado en Ingeniería Informática



Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación  
Universidad de Granada

## Máster Online en Ciberseguridad

Nº1 en España según El Mundo



Hasta el 46%  
de beca



Mejor Máster  
según el  
Ranking de  
ELMUNDO

Para ser el mejor hay que aprender  
de los mejores.

IMEF

Smart Education

Deloitte

Infórmate



## Tema 5: Preprocesamiento de datos

**Preprocesamiento:** Tareas para disponer de datos de calidad previos al uso de algoritmos de extracción de conocimiento.

### 1. Introducción. Preprocesamiento

“El propósito fundamental de la preparación de los datos es la manipulación y transformación de los datos sin refinar para que la información contenida en el conjunto de datos pueda ser descubierta o estar accesible de forma más fácil.”

“El preprocesamiento de datos es un paso a menudo descuidado pero muy importante en el proceso de minería de datos”.

#### Importancia del preprocesamiento de datos

1. Los datos reales pueden ser impuros, pueden conducir a la extracción de patrones/reglas poco útiles. Esto se puede deber a: datos incompletos (falta de valores de atributos, etc), datos con ruido y datos inconsistentes (incluyendo discrepancias).
2. El preprocesamiento de datos puede generar un conjunto de datos más pequeño que el original, lo cual puede mejorar la eficiencia del proceso de Minería de Datos. Esta acción incluye:
  - Selección relevante de datos: eliminando registros duplicados, eliminando anomalías...
  - Reducción de datos: selección de características, muestreo o selección de instancias, discretización. Una reducción de datos produce una mayor eficiencia y eficacia.
3. El preprocesamiento de datos genera “datos de calidad”, los cuales pueden conducir a “patrones/reglas de calidad”. Por ejemplo, se puede: recuperar información incompleta, eliminar outliers, resolver conflictos, seleccionar variables relevantes...

Datos de baja calidad puede llevar a modelos de minería de datos de baja calidad. Decisiones de calidad deben ser basadas en datos de calidad. El preprocesamiento de datos (limpieza, transformación, reducción...) puede llevar la mayor parte del tiempo de trabajo en una aplicación de ciencia de datos (80%). El preprocesamiento de datos consume una parte muy importante del tiempo total de un proceso de ciencia de datos.

#### ¿Qué incluye el preprocesamiento de datos?

El preprocesamiento de datos engloba a todas aquellas técnicas de análisis de datos que permite mejorar la calidad de un conjunto de datos de modo que las técnicas de extracción de conocimiento/minería de datos puedan obtener mayor y mejor información (mejor porcentaje de clasificación, reglas con más completitud, etc.)

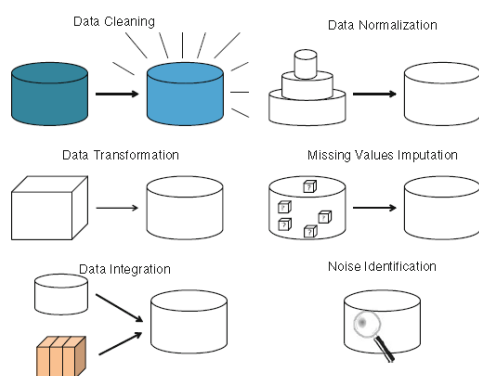
Nos referimos a la **preparación de datos** como el conjunto de técnicas que inicializan los datos adecuadamente para que sirvan de entrada a un determinado algoritmo de DM. La **reducción de datos** comprende el conjunto de técnicas que, de un modo u otro, obtienen una representación reducida de los datos originales.

El preprocesamiento de datos incluye la preparación de los datos, compuesta por la integración, la limpieza, la normalización y la transformación de los datos; y las tareas de reducción de datos, como la selección de características, la selección de instancias, la discretización, etc.

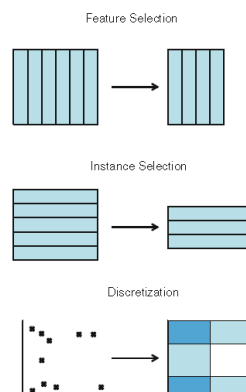


El resultado que se espera tras un encadenamiento fiable de las tareas de preprocesamiento de datos es un conjunto de datos final, que puede considerarse correcto y útil para otros algoritmos de minería de datos.

### Preparación de datos



### Reducción de datos



## 2. Integración, Limpieza y Transformación

### Preparación de datos

- ¿Cómo se limpian los datos? - Data Cleaning
- ¿Cómo se incorporan y ajustan los datos? — Data Integration
- ¿Cómo puedo proporcionar datos precisos? — Data Transformation
- ¿Cómo se unifican y escalan los datos? — Data Normalization

### Integración de datos

Obtiene los datos de diferentes fuentes de información, resuelve problemas de representación y codificación e Integra los datos desde diferentes tablas para crear información homogénea.

Ejemplos:

Diferentes escalas: salario en dólares VS peniques

Atributos derivados: salario mensual VS salario anual

Tenemos que tener en cuenta algunas consideraciones al realizar la integración de datos desde distintas fuentes:

**Integración del esquema.** ¿Cómo asegurar que entidades equivalentes se emparejan correctamente cuando se produce la fusión desde distintas fuentes? Por ejemplo: id-cliente y num-cliente. Esto se resuelve usando los metadatos que normalmente se almacenan en las BBDD y los DW.

### Detección de datos duplicados e inconsistencias

**Redundancia.** Un atributo es redundante si puede obtenerse a partir de otros. Una forma de detectar redundancia es mediante análisis de correlaciones.

#### Análisis de correlaciones

**Objetivo:** medir la fuerza con la que un atributo implica a otro, en función de los datos disponibles. La correlación entre dos atributos A y B puede medirse como:

$$r_{A,B} = \frac{\sum(A - \bar{A})(B - \bar{B})}{\sigma_A \sigma_B}$$

















$\bar{A}$ : media  
 $\sigma_A$ : desviación típica

- $0 < r_{A,B} \leq 1 \rightarrow A$  y  $B$  están correlacionados positivamente (ambas tienen comportamiento similar)
- $r_{A,B} = 0 \rightarrow A$  y  $B$  son independientes
- $-1 \leq r_{A,B} < 0 \rightarrow A$  y  $B$  están correlacionados negativamente (si un atributo crece, el otro decrece)

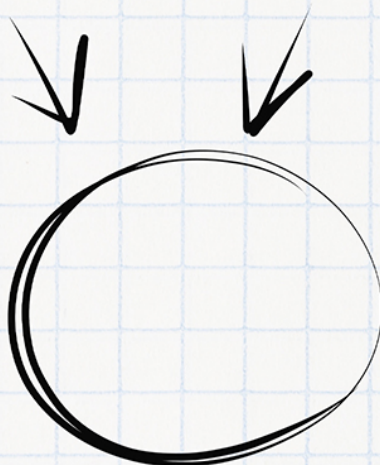


# Imagínate aprobando el examen

## Necesitas tiempo y concentración

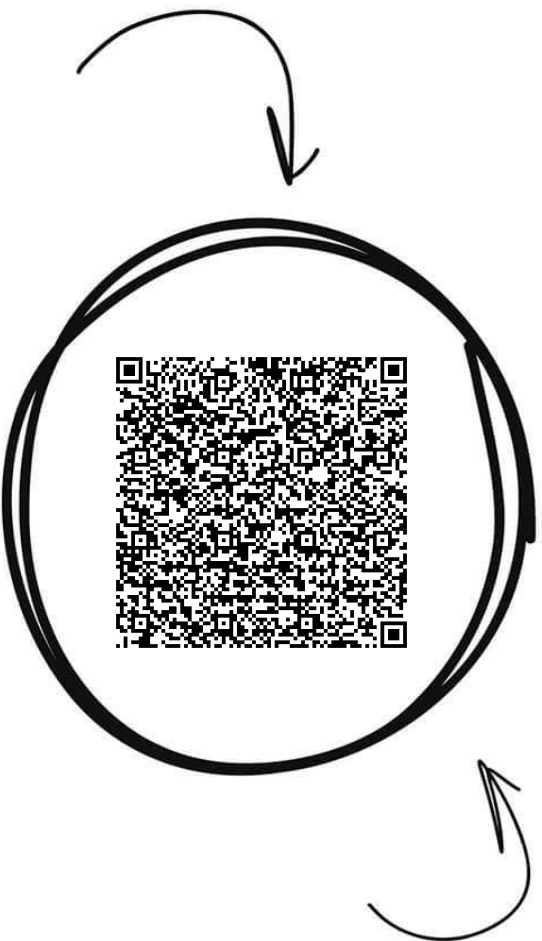
Planes	 PLAN TURBO	 PLAN PRO	 PLAN PRO+
 Descargas sin publi al mes	10 	40 	80 
 Elimina el video entre descargas			
 Descarga carpetas			
 Descarga archivos grandes			
 Visualiza apuntes online sin publi			
 Elimina toda la publi web			
 Precios <span>Anual <input type="checkbox"/></span>	0,99 € / mes	3,99 € / mes	7,99 € / mes

Ahora que puedes conseguirlo,  
¿Qué nota vas a sacar?



# WUOLAH

## Apuntes Variados



Banco de apuntes de la

WUOLAH



**Comparte estos flyers en tu clase y consigue más dinero y recompensas**

- 1** Imprime esta hoja
- 2** Recorta por la mitad
- 3** Coloca en un lugar visible para que tus compis puedan escanar y acceder a apuntes
- 4** Llévate dinero por cada descarga de los documentos descargados a través de tu QR





## Transformación: Normalización

**Objetivo:** pasar los valores de un atributo a un rango mejor. Útil/necesario para algunas técnicas como ANN o métodos basados en distancias (k-NN,...). Algunas técnicas de normalización son:

**Normalización min-max:** Realiza una transformación lineal de los datos originales. Las relaciones entre los datos originales se conservan.

$$[\min_A, \max_A] \rightarrow [\text{nuevo}_{\min_A}, \text{nuevo}_{\max_A}]$$
$$v' = \frac{v - \min_A}{\max_A - \min_A} (\text{nuevo}_{\max_A} - \text{nuevo}_{\min_A}) + \text{nuevo}_{\min_A}$$

**Normalización zero-mean (Z-score).** Se normaliza en función de la media y la desviación estándar. Útil cuando se desconocen los límites o cuando los datos anómalos pueden dominar la normalización min-max.

$$v' = \frac{v - \bar{A}}{\sigma_A}$$

**Normalización por escala decimal.** Normaliza moviendo el punto decimal de los valores del atributo. El número de puntos decimales movidos depende del valor absoluto máximo de A. Con  $j$  igual al menor entero tal que  $\max(|v'|) < 1$ .

$$v' = \frac{v}{10^j}$$

## 3. Datos Imperfectos

¿Cómo se manejan los datos imperfectos? Con la imputación, filtrado, reetiquetado...

Los datos imperfectos son: los datos normalizados, los datos con ruido y los valores perdidos.

### Valores perdidos

Se pueden utilizar las siguientes opciones, aunque algunas de ellas sesgan los datos:

- Ignorar la tupla. Suele usarse cuando la variable a clasificar no tiene valor
- Rellenar manualmente los datos. En general es impracticable
- Utilizar una constante global para la sustitución. P.e. “desconocido”, “?”,...
- Rellenar utilizando la media/desviación del resto de las tuplas
- Rellenar utilizando la media/desviación del resto de las tuplas pertenecientes a la misma clase
- Rellenar con el valor más probable. Para ello utilizar alguna técnica de inferencia, p.e. bayesiana o un árbol de decisión

Los valores perdidos no es que se han perdido por el camino, es que no se conocen. Por ejemplo, que no se han calculado.



Importante

Puedo eliminar la publi de este documento con 1 coin

¿Cómo consigo coins? → Plan Turbo: barato  
→ Planes pro: más coins

perdo espacio



- Estimar por el valor más probable (la moda)  
 $X = n \rightarrow \text{error}$

- Estimar por el valor más probable (la moda) dentro de la clase (+)  
 $X = a$  (prob. 0.5) o  $X = n$  (prob. 0.5)  
→ No resuelve nada

- Rellenar buscando relaciones entre variables. \* Si la variable a imputar es nominal, se aplicaría **clasificación**

Estimación mediante técnicas de aprendizaje automático:

- Imputación con k-NN (KNNI)
- Imputación con k-NN y pesos (WKNNI)
- Imputación basada en clustering (KMI)
- Imputación basada en algoritmos de SVM (SVMi)
- Otros: event covering (EC), singular value decomposition (SVDI), local least squares imputation (LLSI)

Algorithm 3 kNNI algorithm.

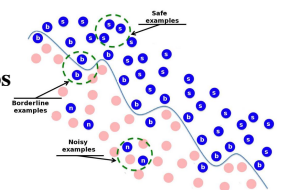
```
function kNNI( $T$  - dataset with MVs,  $k$  - number of neighbors per instance to be chosen,  
 $D(x, y)$  - a distance or dissimilarity function of  $x$  and  $y$ ,  $S$  - the imputed version of  $T$ )  
  initialize:  $S = \{\}$   
  for each instance  $y_i$  in  $T$  do  
     $\hat{y}_i \leftarrow y_i$   
    if  $y_i$  contains any missing value then  
      Find set  $I_{K_i}$  with the  $k$  nearest instances to  $y_i$  from  $T$  using  $D$   
      for each missing value in attribute  $h$  of  $y_i$  do  
        if  $h$  is numerical then  
           $\hat{y}_{ih} = (\sum_{j \in I_{K_i}} y_{jh}) / (|I_{K_i}|)$   
        else  
           $\hat{y}_{ih} = \text{mode}(I_{K_ih})$   
        end if  
      end for  
    end if  
     $S \leftarrow \hat{y}_i$   
  end for  
  return  $S$   
end function
```

## Tipos de ruido

Clases con ruido: son contradictorias o no están etiquetadas.

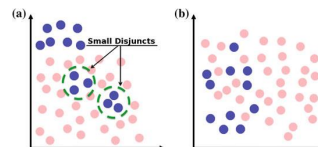
Atributos con ruido: valores erróneos, que faltan o valores que no importan

Tipos de ejemplos: Los tres tipos de ejemplos considerados en este libro: **ejemplos seguros** (etiquetados como s), **ejemplos límite** (etiquetados como b) y **ejemplos ruidosos** (etiquetados como n). La línea continua muestra el límite de decisión entre las dos clases



Ejemplos de interacción entre clases:

a) pequeñas disyuntivas y b) solapamiento entre clases



## Limpeza de datos con ruido

Uso de técnicas de filtrado de ruido en clasificación: Los tres filtros de ruido que se mencionan a continuación, los más conocidos, utilizan un esquema de votación para determinar qué casos eliminar del conjunto de entrenamiento:

- **Ensemble Filter (EF):** Se utilizan **diferentes algoritmos de aprendizaje** para crear clasificadores en varios subconjuntos de los datos de entrenamiento que sirven como filtros de ruido para los conjuntos de entrenamiento. Hay dos pasos principales:



1. Para cada algoritmo de aprendizaje (C4.5, 1-NN y LDA), se utiliza una validación cruzada k-fold para etiquetar cada ejemplo de validación como correcto (predicción = etiqueta de los datos de entrenamiento) o mal etiquetado (predicción  $\neq$  etiqueta de los datos de entrenamiento)
2. Se utiliza un esquema de votación para identificar el conjunto final de ejemplos ruidosos
  - **Votación por consenso:** elimina un ejemplo si es mal clasificado por todos los clasificadores
  - **Votación por mayoría:** elimina un ejemplo si es mal clasificado por más de la mitad de los clasificadores

---

**Algorithm 4 EF algorithm.**

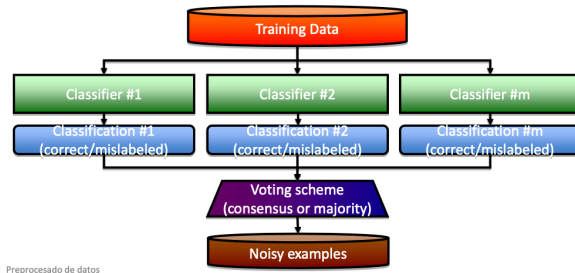

---

```

function EF( $T$  - dataset with MVs,  $\Gamma$  - number of subsets,  $\mu$  - number of filters to be
used,  $F$  - set of classifiers)
  Split the training data set  $T$  into  $T_i, i = 1 \dots \Gamma$  equal sized subsets
  for each filter  $F_{x_i}, x = 1$  to  $\mu$  do
    for each subset  $T_i$  do
      Use  $\{T_i, j \neq i\}$  to train  $F_{x_i}$  resulting in  $F_{x_i}^n$ 
      for each instance  $t$  in  $T_i$  do
        Classify  $t$  with every  $F_{x_i}^n$ 
      end for
    end for
  end for
  for each instance  $t$  in  $T$  do
    Use a voting scheme to include  $t$  in  $T_N$  according to the classifications made by
    each filter  $F_{x_i}$ 
  end for
  return  $T - T_N$ 
end function

```

---



- **Cross-Validated Committees Filter (CVCF):** CVCF es similar a EF, pero con dos diferencias principales:

1. El mismo algoritmo de aprendizaje (C4.5) se utiliza para crear clasificadores en varios subconjuntos de los datos de entrenamiento. Los autores del CVCF hacen especial hincapié en el uso de conjuntos de árboles de decisión, como el C4.5, porque funcionan bien como filtro para datos ruidosos.
2. Cada clasificador construido con la validación cruzada k-fold se utiliza para etiquetar TODOS los ejemplos de entrenamiento (no sólo el conjunto de validación) como correctos (predicción = etiqueta de datos de entrenamiento) o mal etiquetados (predicción  $\neq$  etiqueta de datos de entrenamiento)

- **Iterative-Partitioning Filter (IPF):** La IPF elimina los datos ruidosos en múltiples iteraciones utilizando CVCF hasta que se alcanza un criterio de parada. El proceso iterativo se detiene si, durante un número de iteraciones consecutivas, el número de ejemplos ruidosos en cada iteración es inferior a un porcentaje del tamaño del conjunto de datos de entrenamiento

## Detección de datos anómalos

Outliers: Son objetos/datos con características que son considerablemente diferentes de la mayoría de los otros datos/objetos del conjunto.

Valores anómalos, atípicos o extremos (*outliers*): son correctos aunque sean anómalos estadísticamente. Pueden ser un inconveniente para métodos basados en ajuste de pesos (p.e. ANN)

Técnicas de **detección**:

- Definir una distancia y ver los individuos con mayor distancia media al resto de individuos
- Clustering parcial: los datos se agrupan en clusters y los datos que queden fuera pueden considerarse outliers

Tratamiento de valores anómalos:

- Ignorar. Algunos algoritmos son robustos a datos anómalos
- Filtrar (eliminar o reemplazar) la columna. Solución extrema, conveniente si existe una columna (atributo) dependiente con datos de mayor calidad

- Filtrar (eliminar o reemplazar) la fila. A veces puede sesgar los datos porque las causas de un dato anómalo están relacionadas con casos o tipos especiales
- Reemplazar el valor. Por valor nulo si el algoritmo de DM trabaja bien con datos nulos, con el máximo o mínimo o la media
- Discretizar: Si transformamos un valor continuo en discreto (muy alto, ..., muy bajo), los datos anómalos caen en la categoría muy alto o muy bajo y se tratan sin problemas

## 4. Reducción de datos

Selecciona/extrae datos relevantes para la tarea de la minería de datos/extracción de información.

¿Cuáles son las cuestiones básicas que deben resolverse en la reducción de datos? Proporcionamos una serie de preguntas acompañadas de las respuestas correctas que involucran cada tipo de proceso que pertenece a la técnica de reducción de datos.

- ¿Cómo reduzco la dimensionalidad de los datos? Selección de características
- ¿Cómo elimino ejemplos redundantes y/o conflictivos? Selección de instancia (selección de prototipos vs selección de conjuntos de entrenamiento)
- ¿Cómo simplifico el dominio de un atributo? Discretización. Se divide el rango de atributos numéricos (continuos o no) en intervalos. Almacenamos las etiquetas de los intervalos. Es crucial para las reglas de asociación y algunas clasificaciones, algoritmos que solo aceptan datos discretos.
- ¿Cómo completo los vacíos en los datos? Extracción de funciones y/o generación de instancias

Hay diferentes vías:

**Selección de características:** El problema de la selección de características (SC) o variables (*Feature Subset Selection, FSS*) consiste en encontrar un subconjunto de las variables del problema que

optimice la probabilidad de clasificar correctamente. ¿Por qué es necesaria la selección de variables?

- Más atributos no significa más éxito en la clasificación
- Trabajar con menos variables reduce la complejidad del problema y disminuye el tiempo de ejecución
- Con menos variables la capacidad de generalización aumenta
- Los valores para ciertos atributos pueden ser costosos de obtener

El resultado de la SC sería:

- Menos datos → los algoritmos pueden aprender más rápidamente
- Mayor exactitud → el clasificador generaliza mejor
- Resultados más simples → más fácil de entender

SC tiene su extensión en la transformación (extracción y construcción de atributos). La SC se puede considerar como un problema de búsqueda.

En un algoritmo de selección de características se distinguen dos componentes principales

- Una estrategia de búsqueda para seleccionar subconjuntos candidatos
- Una función objetivo que evalúe esos subconjuntos

Estrategia de búsqueda

- Dadas  $N$  variables, explorar todos los subconjuntos posibles supone  $2^N$  (p.e.  $2^{20}=1048576$ )

- Si queremos exactamente subconjuntos de M variables ( $M \leq N$ ) entonces supone  $(N^M)$ . P.e. explorar subconjuntos de 10 variables de 20 posibles, daría 184756
- Una búsqueda exhaustiva no es aceptable

Función objetivo: evaluar la bondad del subconjunto seleccionado. Se distinguen dos enfoques distintos:

**Filtro (filter).** La función objetivo evalúa los subconjuntos basándose en la información que contienen. Se utiliza como función objetivo medidas de separabilidad de clases, de dependencias estadísticas, basadas en teoría de la información,...

**Envoltente (wrapper).** La función objetivo consiste en aplicar la técnica de aprendizaje que se utilizará finalmente sobre la proyección de los datos al conjunto de variables candidato. El valor devuelto suele ser el porcentaje de acierto del clasificador construido

**Algorithm 1** Sequential forward feature set generation - SFG.

```
function SFG(F - full set, U - measure)
  initialize: S = {}
  repeat
    f = FINDNEXT(F)
    S = S ∪ {f}
    F = F - {f}
  until S satisfies U or F = {}
  return S
end function
```

▷ S stores the selected features

**Algorithm 2** Sequential backward feature set generation - SBG.

```
function SBG(F - full set, U - measure)
  initialize: S = {}
  repeat
    f = GETNEXT(F)
    F = F - {f}
    S = S ∪ {f}
  until S does not satisfy U or F = {}
  return F ∪ {f}
end function
```

▷ S holds the removed features

## Medidas filtro

**Medidas de separabilidad.** Miden la separabilidad entre clases: euclídeas, Mahalanobis,...

- P. ej. Para un problema con 2 clases, un proceso de SC basado en medidas de este tipo determina que X es mejor que Y si X induce una diferencia mayor que Y entre las dos probabilidades condicionales de las clases

**Correlaciones.** Serán buenos subconjuntos los que estén muy correlacionados con la clase

$$f(X_1, \dots, X_M) = \frac{\sum_{i=1}^M \rho_{ic}}{\sum_{i=1}^M \sum_{j=i+1}^M \rho_{ij}}$$

donde  $\rho_{ic}$  es el coeficiente de correlación entre la variable  $X_i$  y la etiqueta  $c$  de la clase (C) y  $\rho_{ij}$  es el coeficiente de correlación entre  $X_i$  y  $X_j$

## Medidas basadas en teoría de información

- La correlación sólo puede medir dependencias lineales. Un método bastante más potente es la información mutua  $I(X_1, \dots, X_M; C)$  donde H representa la entropía y  $\omega_c$  la c-ésima etiqueta de la clase C

$$f(X_{1 \dots M}) = I(X_{1 \dots M}; C) = H(C) - H(C|X_{1 \dots M}) = \sum_{c=1}^{|C|} \int_{X_{1 \dots M}} P(X_{1 \dots M}, \omega_c) \log \frac{P(X_{1 \dots M}, \omega_c)}{P(X_{1 \dots M})P(\omega_c)} dx$$

- La información mutua mide la cantidad de incertidumbre que disminuye en la clase C cuando se conocen los valores del vector  $X_1 \dots X_M$
- Por la complejidad de cálculo de I se suelen utilizar reglas heurísticas

$$f(X_{1 \dots M}) = \sum_{i=1}^M I(X_i; C) - \beta \sum_{i=1}^M \sum_{j=i+1}^M I(X_i; X_j)$$

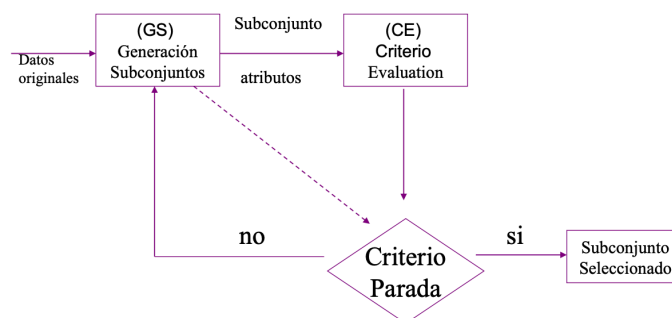
# Consigue Empleo o Prácticas

Matricúlate en IMF y accede sin coste a nuestro servicio de Desarrollo Profesional con más de 7.000 ofertas de empleo y prácticas al mes.

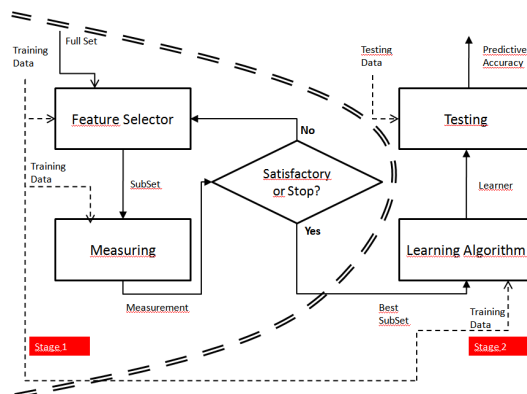


## Medidas de consistencia

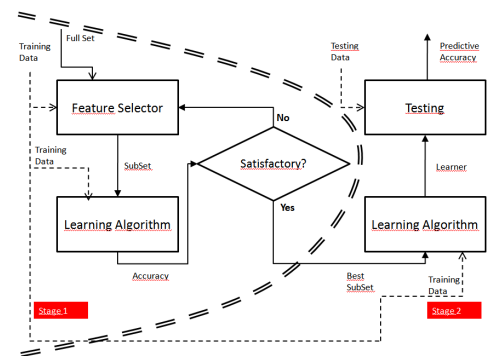
- Los tres grupos de medidas anteriores intentan encontrar las características que puedan, de forma maximal, predecir una clase frente al resto
- Este enfoque no puede distinguir entre dos variables igualmente adecuadas, no detecta variables redundantes
- Las medidas de consistencia tratan de encontrar el mínimo número de características que puedan separar las clases de la misma forma que lo hace el conjunto completo de variables



Modelo de filtro



Modelo de envoltura (wrapper)



## Ventajas

### ■ Envoltentes:

- Exactitud: generalmente son más exactos que los de filtro, debido a la fuerte interacción entre el clasificador y la selección de características
- Regular el sobreajuste: al incorporar técnicas de validación, pueden controlar el sobreajuste

### ■ Filtro:

- Rápidos: Suelen limitarse a cálculos de frecuencias, mucho más rápido que entrenar un clasificador
- Generalidad: Al evaluar propiedades intrínsecas de los datos y no su interacción con un clasificador, sus resultados pueden ser utilizados por cualquier clasificador





## Inconvenientes

### ■ Envolventes:

- Muy costosos: para cada evaluación hay que aprender un modelo y validarlo. No es factible para clasificadores costosos
- Pérdida de generalidad: la solución está sesgada hacia el clasificador utilizado y hay menos garantías de que la selección siga funcionando bien en otros clasificadores

### ■ Filtro:

- Tendencia a incluir muchas variables: Normalmente se debe a las características monótonas de la función objetivo utilizada. El usuario deberá seleccionar un umbral

## Tenemos distintas clasificaciones:

### 1. Según la evaluación:

filter  
wrapper

### 2. Disponibilidad de la clase:

Supervisados  
No supervisado

### 3. Según la búsqueda:

Completa  $O(2^N)$   
Heurística  $O(N^2)$   
Aleatoria ¿?

### 4. Según la salida del algoritmo:

Ranking  
Subconjunto de atributos

**Algoritmos Subconjunto de Atributos:** Devuelven un subconjunto de atributos optimizado según algún criterio de evaluación

```
Entrada: x atributos - U criterio evaluación
Subconjunto = {}
Repetir
     $S_k = \text{generarSubconjunto}(x)$ 
    si existeMejora( $S, S_k, U$ )
        Subconjunto =  $S_k$ 
Hasta CriterioParada()
Salida: Lista, atts más relevantes al principio
```

**Algoritmos de Ranking:** Devuelven una lista de atributos ordenados según algún criterio de evaluación.

```
Entrada: x atributos - U criterio evaluación
Lista = {}
Para cada Atributo  $x_i, i \in \{1, \dots, N\}$ 
     $v_i = \text{calcular}(x_i, U)$ 
    situar  $x_i$  dentro de Lista conforme  $v_i$ 
Salida: Lista, atts más relevantes al principio
```

**Selección hacia delante:** La selección *forward* comienza con el conjunto vacío y de forma secuencial añade al subconjunto actual  $S$  el atributo  $X_i$  que maximiza  $f(S, X_i)$

1. Comenzar con  $S = \emptyset$
2. Seleccionar la variable  $X^+ = \arg \max_{X \in U - S} f(S \cup X)$
3.  $S = S \cup \{X^+\}$
4. Ir al paso 2

Funciona mejor cuando el subconjunto óptimo tiene pocas variables. Es incapaz de eliminar variables.

**Selección hacia atrás:** La selección *backward* comienza con el conjunto completo  $U$  y de forma secuencial elimina del subconjunto actual  $S$  el atributo  $X$  que decremente menos  $f(S - X)$

1. Comenzar con  $S = U$
2. Seleccionar la variable  $X^- = \arg \max_{X \in S} f(S - X)$
3.  $S = S - \{X^-\}$
4. Ir al paso 2

Funciona mejor cuando el subconjunto óptimo tiene muchas variables. El principal inconveniente es el de reevaluar la utilidad de algunos atributos previamente descartados.

**Selección l-más r-menos:** Es una generalización de *forward* y *backward*

1. Si  $l > r$  entonces  $S = \emptyset$   
si no,  $S = U$  e ir al paso 3
2. Repetir  $l$  veces  
 $X^+ = \arg \max_{X \in U - S} f(S \cup X)$   
 $S = S \cup \{X^+\}$
3. Repetir  $r$  veces  
 $X^- = \arg \max_{X \in S} f(S - X)$   
 $S = S - \{X^-\}$
4. Ir al paso 2

**Selección bidireccional:** Es una implementación paralela de *forward* y *backward*. Hay que asegurar que los atributos eliminados por *backward* no son introducidos por *forward* (y viceversa)

1. Comenzar *forward* con  $S_F = \emptyset$
2. Comenzar *backward* con  $S_B = U$
3. Seleccionar  
 $X^+ = \arg \max_{X \in S_B - S_F} f(S_F \cup X)$   
 $S_F = S_F \cup \{X^+\}$
4. Seleccionar  
 $X^- = \arg \max_{X \in S_B - S_F} f(S_B - X)$   
 $S_B = S_B - \{X^-\}$
5. Ir al paso 3

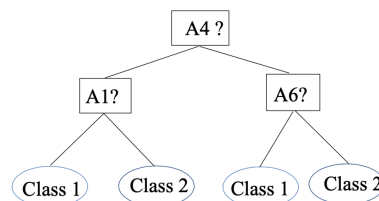
**Selección flotante:** Extensión de l-más r-menos para evitar fijar el  $l$  y  $r$  a priori. Hay dos métodos: uno comienza por el conjunto vacío y otro por el total

1. Comenzar con  $S = \emptyset$
2. Seleccionar  
 $X^+ = \arg \max_{X \in U - S} f(S \cup X)$   
 $S = S \cup \{X^+\}$
3. Seleccionar  
 $X^- = \arg \max_{X \in S} f(S - X)$
4. Si  $f(S - X^-) > f(S)$   
entonces  $S = S - \{X^-\}$  e ir al paso 3  
si-no ir al paso 2

## Selección de características con árboles de decisión

Conjunto inicial de atributos: {A1, A2, A3, A4, A5, A6}

Características seleccionadas: {A1, A4, A6}



### Algunos algoritmos relevantes:

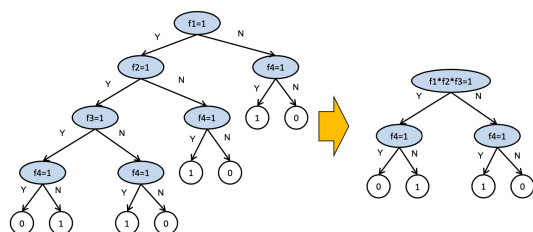
**Algoritmos exhaustivos.** Garantizan el óptimo (son eficaces)

pero el número de subconjuntos evaluados aumenta exponencialmente con la dimensionalidad del espacio de búsqueda (Branch and bound, berma search)

**Algoritmos heurísticos.** Añaden o eliminan variables al subconjunto candidato de forma secuencial. Suelen quedarse en óptimos locales pero son eficientes (Selección hacia delante, selección hacia atrás, selección l-más r-menos, búsqueda bidireccional, selección secuencial flotante)

**Algoritmos estocásticos.** Utilizan aleatoriedad para escapar de óptimos locales. Buen compromiso entre eficacia y eficiencia (Ascensión de colinas con reinicios, enfriamiento estocástico, algoritmos genéticos, enfriamiento simulado)

## Extracción de características



El efecto de utilizar el producto de características en el modelado de árboles de decisión

**Selección de instancias:** La SI pretende elegir los ejemplos que sean relevantes para una aplicación y lograr el máximo rendimiento. El resultado de la SI sería:

- Menos datos → los algoritmos pueden aprender más rápidamente
- Mayor exactitud → el clasificador generaliza mejor
- Modelos generados más simples → más fácil de entender

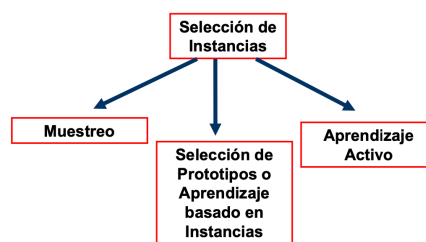
SI y Transformación (compactación/agrupamiento).

### Selección de prototipos:

Propiedades:

- Dirección de la búsqueda: Incremental, decremental, por lotes, mezclada y fijada.
- Tipo de selección: Condensación, Edición, Híbrido.
- Tipo de evaluación: Filtrada o envolvente.

(Taxonomía de selección de prototipos y mapa de selección de prototipos)



Importante

Puedo eliminar la publi de este documento con 1 coin

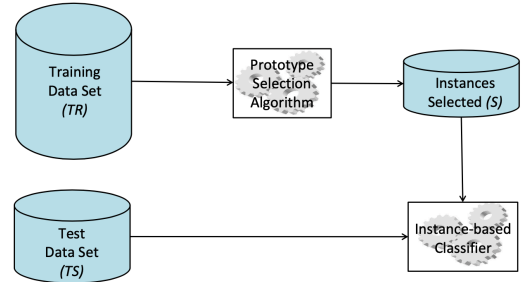
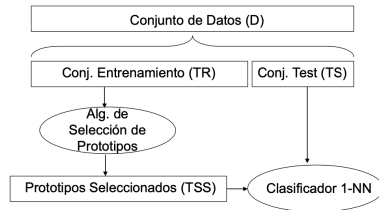
¿Cómo consigo coins? → Plan Turbo: barato  
→ Planes pro: más coins

pierdo espacio



Selección de prototipos

Selección de Prototipos para Clasificación con 1-NN



Formas de evaluar un algoritmo de Selección de instancias en k-NN:

- Reducción del espacio de almacenamiento
- Tolerancia al ruido
- Precisión en la generalización del aprendizaje
- Requerimientos de cómputo

Un par de algoritmos clásicos:

**Algoritmo clásico de Condensación: Condensed Nearest Neighbor (CNN):** es incremental. Inserta solo las instancias mal clasificadas a partir de una selección aleatoria de una instancia de cada clase. Dependiente del orden de presentación. Tiende a retener puntos pertenecientes al borde

```
Algorithm 10 CNN algorithm.
function CNN(T - training data)
  initialize:  $S = \emptyset$ 
  repeat
    for all  $x \in T$  (in random order) do
      Find  $x' \in S$  s.t.  $\|x - x'\| = \min_{x' \in S} \|x - x'\|$ 
      if  $class(x) \neq class(x')$  then
         $S = S \cup \{x\}$ 
      end if
    end for
  until  $S$  does not change
  return  $S$ 
end function
```

**Algoritmo clásico de Edición: Edited Nearest Neighbor (ENN):** vale por lotes. Se eliminan aquellas instancias que se clasifican incorrectamente usando sus k vecinos más cercanos ( $k = 3, 5, 7, 9 \dots$ ). Suaviza las fronteras, pero retiene el resto de puntos (muchos redundantes)

```
Algorithm 11 ENN algorithm.
function ENN(T - training data, k - number of nearest neighbor)
  initialize:  $S = T$ 
  for all  $x \in S$  do
     $X' = \emptyset$ 
    for  $i = 1$  to  $k$  do
      Find  $x'_i \in T$  s.t.  $x \neq x'_i$  and  $\|x - x'_i\| = \min_{x' \in (T \setminus X')} \|x - x'\|$ 
       $X' = X' \cup \{x'_i\}$ 
    end for
    if  $class(x) \neq majorityClass(X')$  then
       $S = S \setminus \{x\}$ 
    end if
  end for
  return  $S$ 
end function
```

**Selección de Instancias. Eficiencia**

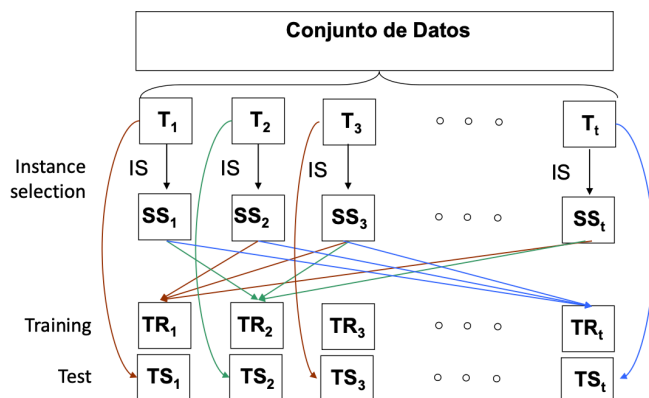
El orden de los algoritmos es superior a  $O(n^2)$  y suele estar en orden  $O(n^3)$



Las principales dificultades que deben afrontar los algoritmos de Selección de Prototipos son: Eficiencia, recursos, generalización, representación.

¿Cómo realizar la selección de instancias con grandes bases de datos? Combinamos una estrategia de estratificación con los algoritmos de selección de instancias.

### Grandes Bases de Datos. Estrategia de Estratificación.



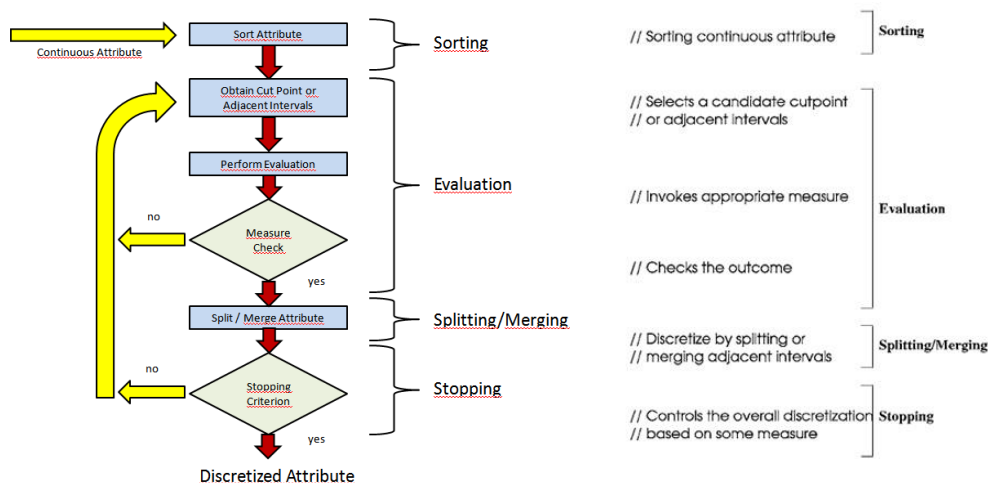
### Conjuntos de datos no balanceados

Algunos problemas tienen una presencia de las clases desigual (diagnóstico médico: 90% sin-enfermedad, 10% enfermedad). La situación es similar con múltiples clases. La mayoría de los clasificadores obtienen un 97% de clasificación correcta, pero no son útiles.

¿Cómo se procesan las clases no balanceadas? Utilizando técnicas de reducción de datos para balancear las clases reduciendo las clases mayoritarias y realizando sobremuestreo para balancear aumentar el tamaño de las clases minoritarias.

**Discretización:** los valores discretos son muy útiles en Minería de Datos. Representan información más concisa, son más fáciles de entender, más cercanos a la representación a nivel de conocimiento. La discretización busca transformar los valores continuos/discretos que se encuentran ordenados en valores nominales que no están ordenados. Proceso de cuantificación de atributos numéricos. Los valores nominales tienen un dominio finito, por lo que también se considera una técnica de reducción de datos. La discretización puede hacerse antes de la obtención de conocimiento o durante la etapa de obtención de conocimiento. Es útil para reglas de asociación y clasificación, pues algunos algoritmos solo aceptan datos discretos.

### Etapas en el proceso de discretización

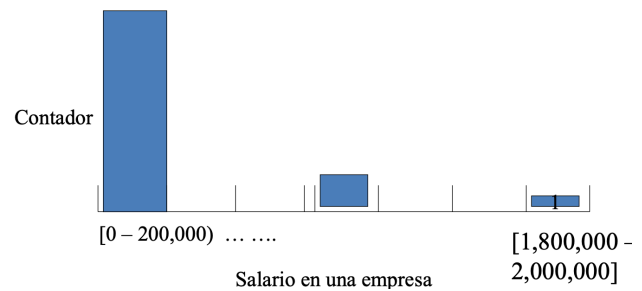
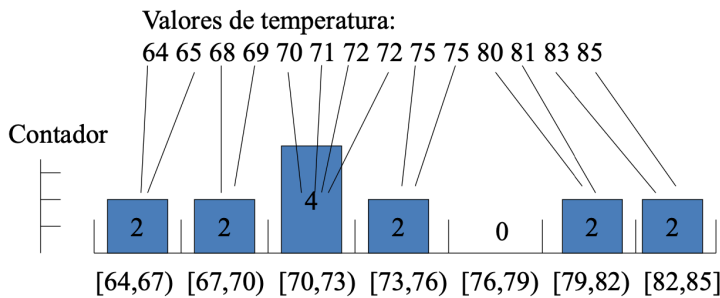


La discretización se ha desarrollado a lo largo de diferentes líneas según las necesidades:

- **Supervisados vs. No supervisados:** Consideran o no el atributo objetivo
- **Dinámicos vs. estáticos:** Mientras se construye o no el modelo
- **Locales vs. Globales:** Centrados en una subregión del espacio de instancias o considerando todas ellas
- **Top-down vs. Bottom-up:** Empiezan con una lista vacía o llena de puntos de corte
- **Directos vs. Incrementales:** Usan o no un proceso de optimización posterior

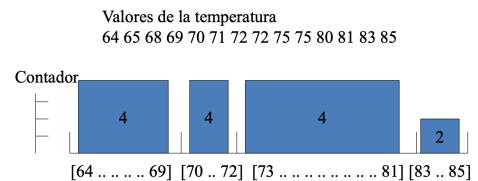
Ejemplo discretización: igual amplitud (no supervisado)

(Puede producir desequilibrios)



Ejemplo discretización: igual amplitud (no supervisado)

- **Ventajas:**
  - Generalmente es preferible porque evita desequilibrios en el balanceo entre valores
  - En la práctica permite obtener puntos de corte más intuitivos
- **Consideraciones adicionales:**
  - Se deben crear cajas para valores especiales
  - Se deben tener puntos de corte interpretables



Algoritmos no supervisados: intervalos de igual amplitud, de igual frecuencia, clustering...

Algoritmos supervisados: basados en entropía, métodos chi-square...

### Discretizador Entropy MDLP (Fayyad)

Comienza por ordenar las instancias según el atributo a discretizar para obtener los puntos de corte entre ejemplos de diferentes clases:

11	14	15	18	19	20	21	22	23	25	30	31	33	35	36
R	C	C	R	C	R	C	R	C	C	R	C	R	C	R

Minimum Description Length Principle (MDLP), basado en entropía, se utiliza para escoger los puntos de corte útiles entre los anteriores. Concretamente, MDLP se usa para decidir cuándo parar la discretización.

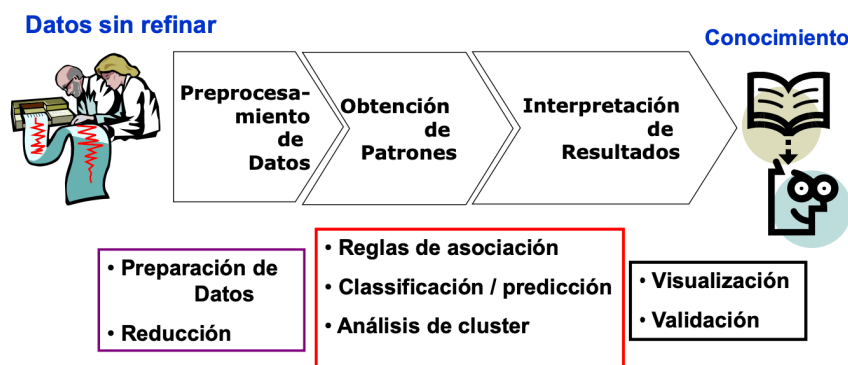
MDLP se formula como el problema de encontrar el coste de comunicación entre un emisor y un receptor. Se asume que el emisor tiene el conjunto de instancias completas mientras que el receptor tiene las instancias sin las etiquetas de clase

Una partición inducida por un punto de corte es aceptada si y solo si el coste del mensaje requerido para enviar antes de particionar es mayor que el requerido después de particionar.

**¿Qué discretizador será mejor?** Como siempre, dependerá de la aplicación, necesidades del usuario, etc. Algunas formas de evaluación son: número total de intervalos, número de inconsistencias causadas, tasa de acierto predictivo y tamaño del modelo generado.

## 5. Comentarios Finales

El preprocesamiento de datos es una necesidad cuando se trabaja con una aplicación real, con datos obtenidos directamente del problema.



**Una ventaja:** El preprocesamiento de datos permite aplicar los modelos de Aprendizaje/Minería de Datos de forma más rápida y sencilla, obteniendo modelos/patrones de más calidad: precisión e/o interpretabilidad.

**Un inconveniente:** El preprocesamiento de datos no es un área totalmente estructurada con una metodología concreta de actuación para todos los problemas.

Cada problema puede requerir una actuación diferente, utilizando diferentes herramientas de preprocesamiento. El diseño de procesos automáticos de uso de las diferentes etapas/técnicas en minería de datos es uno de los nuevos retos existentes.

Las Técnicas de Reducción de Datos pueden permitir mejorar la precisión/interpretabilidad de los métodos de extracción de conocimiento, además de reducir el tamaño de la BD y el tiempo de los algoritmos de aprendizaje.

Para cada método de aprendizaje/problema puede ser necesario diseñar un mecanismo de reducción de datos que sea cooperativo con el propio método de aprendizaje.

*“Good data preparation is key to produce valid and reliable models”*

El software de minería de datos KEEL incluye un módulo de preparación de datos de datos (selección de características, imputación de valores perdidos, selección de instancias, discretización...)