

# Minería en Flujo de Datos

Jorge Casillas

Universidad de Granada

<http://decsai.ugr.es/~casillas>

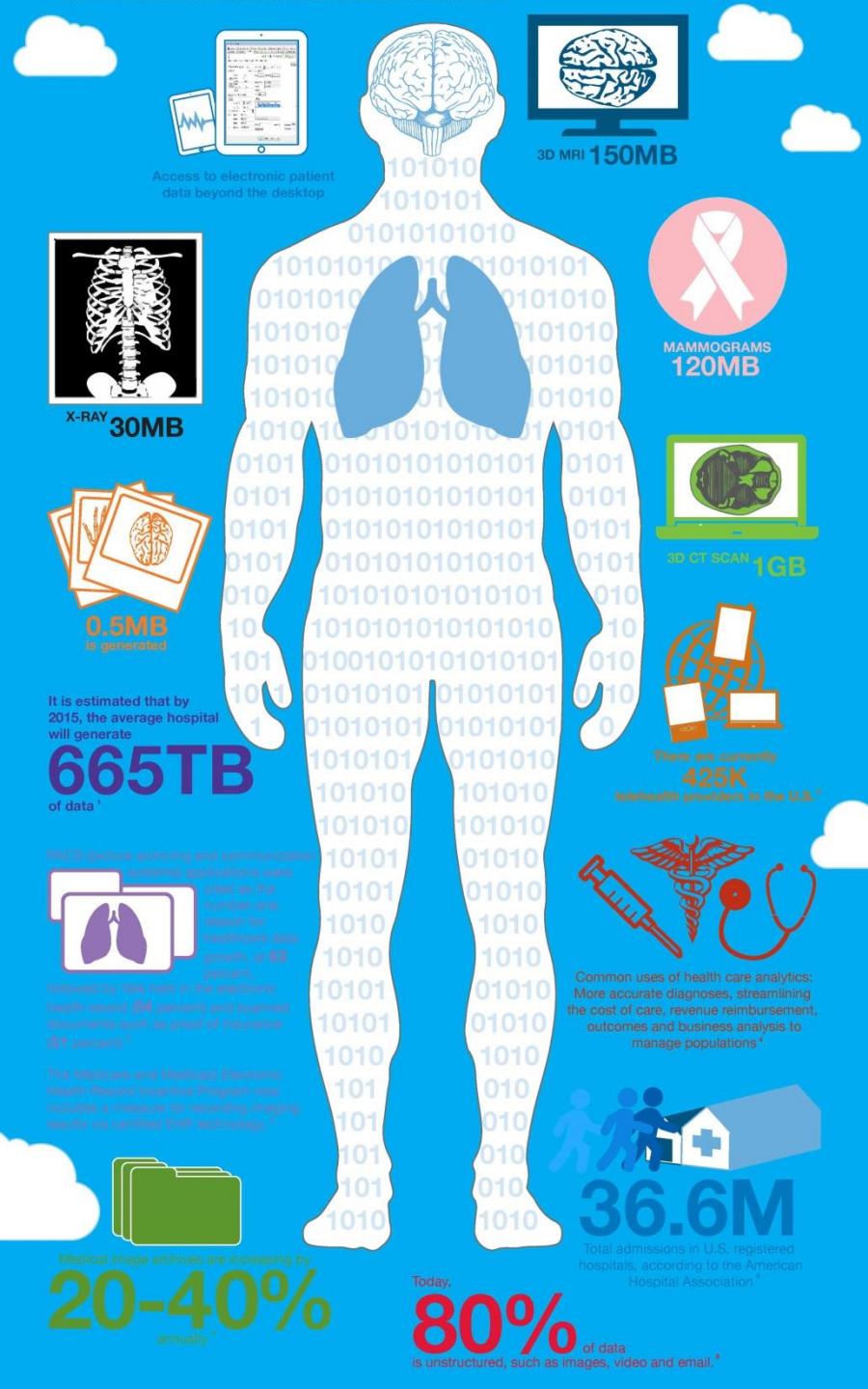




Cada **15 minutos**, la cantidad de información generada por los humanos crece **20 petabytes**, equivalente a todos los trabajos escritos en la **historia de la humanidad** en todas sus lenguas

# ¿Qué pasa en un minuto en internet en 2017?



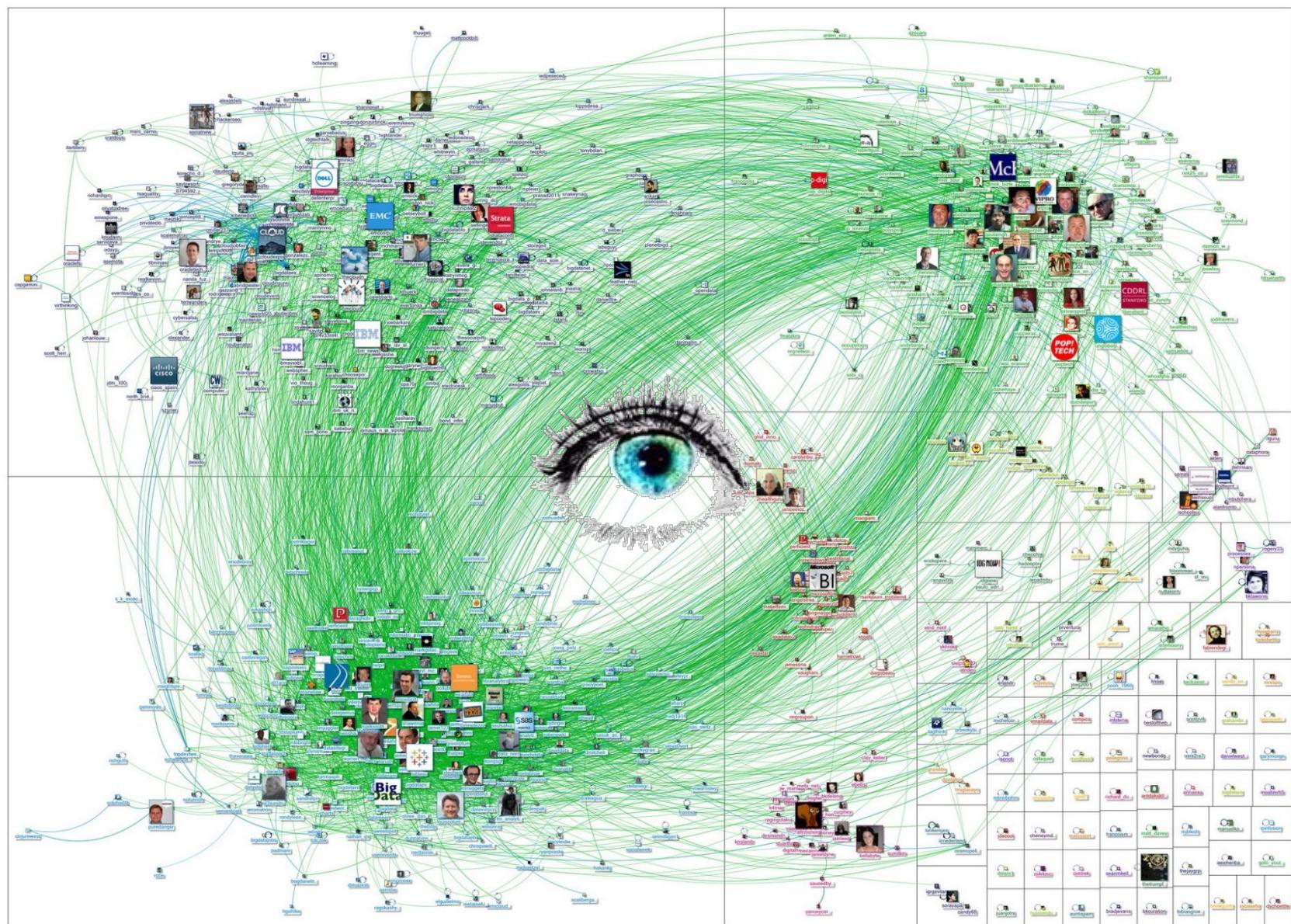


# El cuerpo humano es una

F <sub>4</sub> U <sub>1</sub> E <sub>1</sub> N <sub>1</sub> T <sub>1</sub> E <sub>1</sub>

I <sub>1</sub> L <sub>1</sub> I <sub>1</sub> M <sub>3</sub> I <sub>1</sub> T <sub>1</sub> A <sub>1</sub> D <sub>2</sub> A <sub>1</sub>

D <sub>2</sub> E <sub>1</sub> D <sub>2</sub> A <sub>1</sub> T <sub>1</sub> O <sub>1</sub> S <sub>1</sub>



Discernir información relevante, sintetizarla y extraer conocimiento de ella es, cada vez, un aspecto más crítico en la sociedad en que vivimos



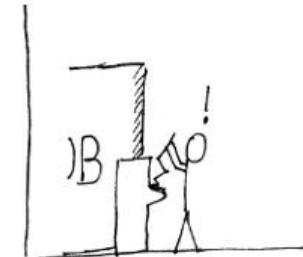
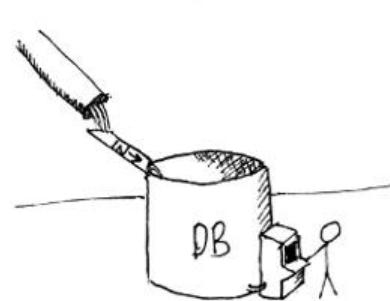
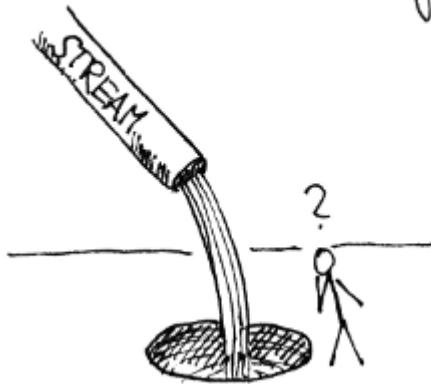
El crecimiento de datos supera la capacidad actual de almacenamiento y, sobre todo, de procesamiento



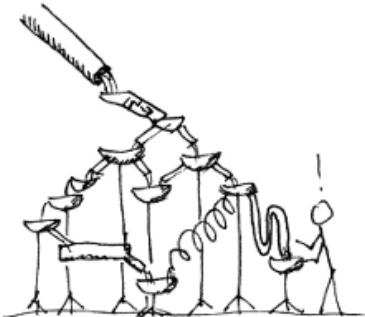
Además, cada vez hay mayor interés en analizar y comprender la información cuando se está produciendo

# ¿Qué hacer con tantos datos?

## Realtime Stream Analysis



## Stream Processing



## Stream Distiller



<http://blog.mikiobraun.de/2011/10/one-does-not-simply-scale-into-realtime-processing.html>

# Problemas Incrementales

- Muchos datos llegan como secuencia continua a lo largo del tiempo:

- Tráfico en redes
- Redes sociales
- Internet de las cosas
- Comunicaciones telefónicas
- Consumo de energía
- Señales fisiológicas
- Compras con tarjetas
- Mercados financieros
- Apuestas
- ...





# Aprendizaje Dinámico

- A diferencia del aprendizaje automático tradicional, no se asume la disponibilidad de suficientes datos de entrenamiento antes de empezar el proceso de aprendizaje
- Varias razones:

Aprendizaje incremental

1. Puede ser computacionalmente intratable manejar todos los datos al mismo tiempo
2. Cuando se dispone de nuevos datos, aprender desde cero es un malgasto de recursos. Puede ser más recomendable adaptar el conocimiento a la luz de los nuevos datos

Minería de flujo de datos

3. Si la generación de datos depende del tiempo, necesariamente deben procesarse conforme llegan
4. En entornos no estacionarios la dinámica puede variar y el aprendizaje debe adaptarse a la nueva realidad, que puede ser inconsistente con lo aprendido previamente

# Aprendizaje Incremental vs. Flujo de Datos

- **Aprendizaje incremental (por razones 1 y 2)**
  - Diseñado para datos estáticos
  - El modelo se construye conforme se analizan los datos. Se puede visitar cada dato varias veces
  - Su objetivo es mejorar la eficiencia al hacer la complejidad del algoritmo independiente del número de instancias
- **Minería de flujo de datos (por razones 3 y 4)**
  - Diseñado para problemas que generan datos continuamente
  - Robusto frente a variaciones en la dinámica de los datos
  - El dato se recibe, se procesa y se olvida
  - Muy eficiente en espacio y tiempo

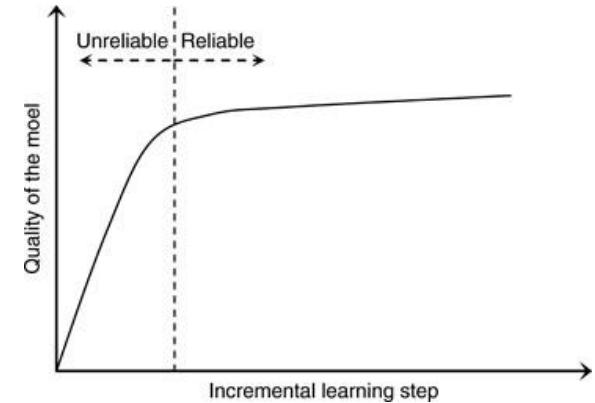
Aprendizaje Incremental  
(o aprendizaje de  
flujo de datos estacionario)

# Aprendizaje Incremental

- Al aprendizaje incremental es, en realidad, la forma habitual en la que los animales inteligentes adquieren conocimiento a lo largo del tiempo
- Las primeras propuestas datan de los años 80 (ID4 [[Shlimmer et al., 1986](#)]), aunque en realidad otras técnicas más antiguas como las RNA, k-nn o naïve bayes ya hacen, inherentemente, un aprendizaje incremental

# Criterios de Evaluación

- Además de los criterios habituales en aprendizaje automático (p. ej., precisión), se proponen otros:
  - Número de ejemplos necesarios para generar un conocimiento estable
  - Coste de procesar cada dato
  - Curva de aprendizaje
- Criterios a evaluar:
  - **Estabilidad**: la precisión no debería variar significativamente en cada paso del aprendizaje
  - **Mejora**: debería mejorarse el rendimiento conforme se van analizando los ejemplos de entrenamiento
  - **Recuperación**: deberían recuperarse de errores incluso ante una caída brusca en el rendimiento



# Modos de Aprendizaje Incremental

- Tareas de aprendizaje incremental de ejemplos
  - Se reciben nuevos datos después de haber entrenado el sistema de aprendizaje (p. ej., en análisis de mensajes en Twitter, llegan nuevos tuits)
- Tareas de aprendizaje incremental de clases
  - Surgen nuevas clases después de tener ya un modelo aprendido (p. ej., en el problema anterior, surgen nuevos *hashtags*)
- Tareas de aprendizaje incremental de atributos
  - Se proporcionan nuevos atributos de entrada después de haber realizado el aprendizaje (p. ej., en el problema anterior, se añaden nuevas propiedades de sentimientos en el texto)

# Aplicabilidad

- Se pueden resolver problemas estáticos con aprendizaje incremental
  - Se toman los datos de uno a uno con muestreo aleatorio. Se realizan varias pasadas a los datos
  - Gracias a su eficiencia, permite abordar problemas big data. Está muy de moda
- Se pueden resolver problemas de naturaleza incremental con aprendizaje no incremental
  - Se hace mediante fuerza bruta acumulando datos y aplicando una y otra vez algoritmos estáticos
  - Es poco eficiente e impráctico en muchos casos, pero sí es útil en algunos otros como, por ejemplo, predecir el precio de la energía en subastas cada 30 minutos

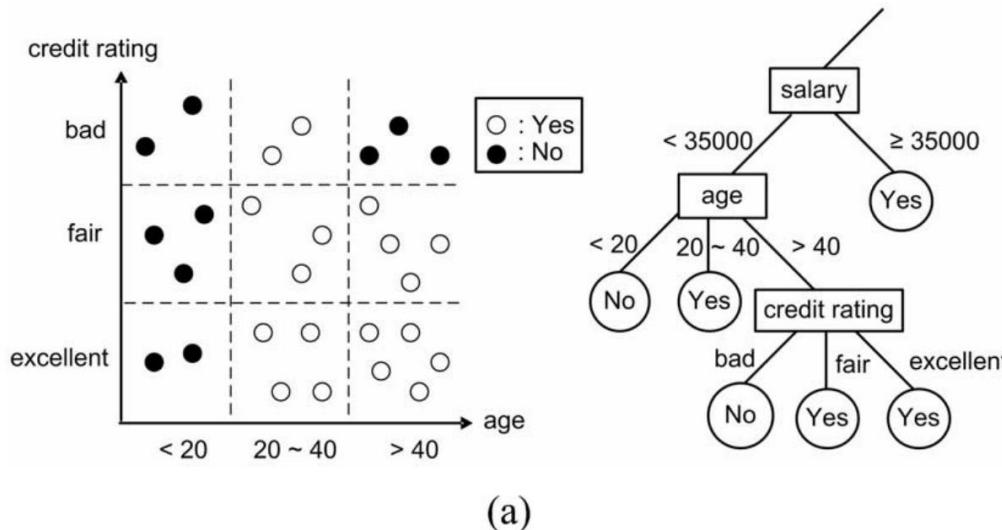
# Construcción Incremental de Árboles de Decisión

- Los árboles Hoeffding se presentan por primera vez en VFDT [Domingos, Hulten, 2000] y, con diferentes variaciones y mejoras, constituyen una familia de algoritmos capaces de construir árboles de decisión a partir de datos que llegan incrementalmente
- Se basan en la desigualdad de Hoeffding de Teoría de la Probabilidad, la cual proporciona una cota superior a la probabilidad de que la suma de variables aleatorias se desvíe una cierta cantidad de su valor esperado

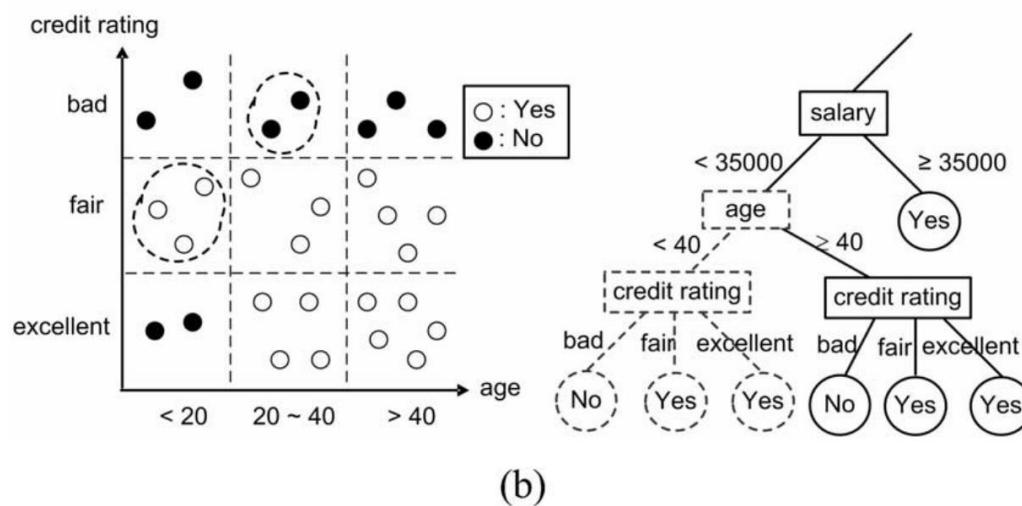
# Construcción Incremental de Árboles de Decisión

- Cada nodo interno de un árbol de decisión contiene una prueba que separa los ejemplos, enviándolos a una u otra rama según los valores de sus atributos
- La decisión crucial para construir el árbol es cuándo dividir un nodo y con qué prueba discriminante (si se usan variables de un único atributo, el número de pruebas posibles es el número de atributos)
- Un criterio habitual para seleccionar la prueba discriminante es la ganancia de información (ID3) o, su extensión, el ratio de ganancia (C4.5), que se basan en minimizar la entropía (es decir, la incertidumbre) de las probabilidades de las diferentes hojas generadas según cómo se separe el nodo

# Construcción Incremental de Árboles de Decisión



(a)



[Tsai, Lee, Yang, 2008]

# Construcción Incremental de Árboles de Decisión

- Con datos estáticos, la ganancia de información estimada es fácil de obtener pues se analizan todos los atributos y datos de entrenamiento disponibles
- Para datos incrementales, se puede usar la cota de Hoeffding que establece que, con una certeza  $1 - \delta$ , la media verdadera de una variable aleatoria de rango  $R$  no difiere de la media estimada después de  $n$  observaciones independientes por más que:

$$\epsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n}}$$

<https://www.otexts.org/1435>

# Esquema de un Árbol Hoeffding

```
1: Let  $HT$  be a tree with a single leaf (the root)
2: for all training examples do
3:   Sort example into leaf  $l$  using  $HT$ 
4:   Update sufficient statistics in  $l$ 
5:   Increment  $n_l$ , the number of examples seen at  $l$ 
6:   if  $n_l \bmod n_{\min} = 0$  and examples seen at  $l$  not all of same class then
7:     Compute  $\bar{G}_l(X_i)$  for each attribute
8:     Let  $X_a$  be attribute with highest  $\bar{G}_l$ 
9:     Let  $X_b$  be attribute with second-highest  $\bar{G}_l$ 
10:    Compute Hoeffding bound  $\epsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n_l}}$ 
11:    if  $X_a \neq X_\emptyset$  and  $(\bar{G}_l(X_a) - \bar{G}_l(X_b)) > \epsilon$  or  $\epsilon < \tau$  then
12:      Replace  $l$  with an internal node that splits on  $X_a$ 
13:      for all branches of the split do
14:        Add a new leaf with initialized sufficient statistics
15:      end for
16:    end if
17:  end if
18: end for
```

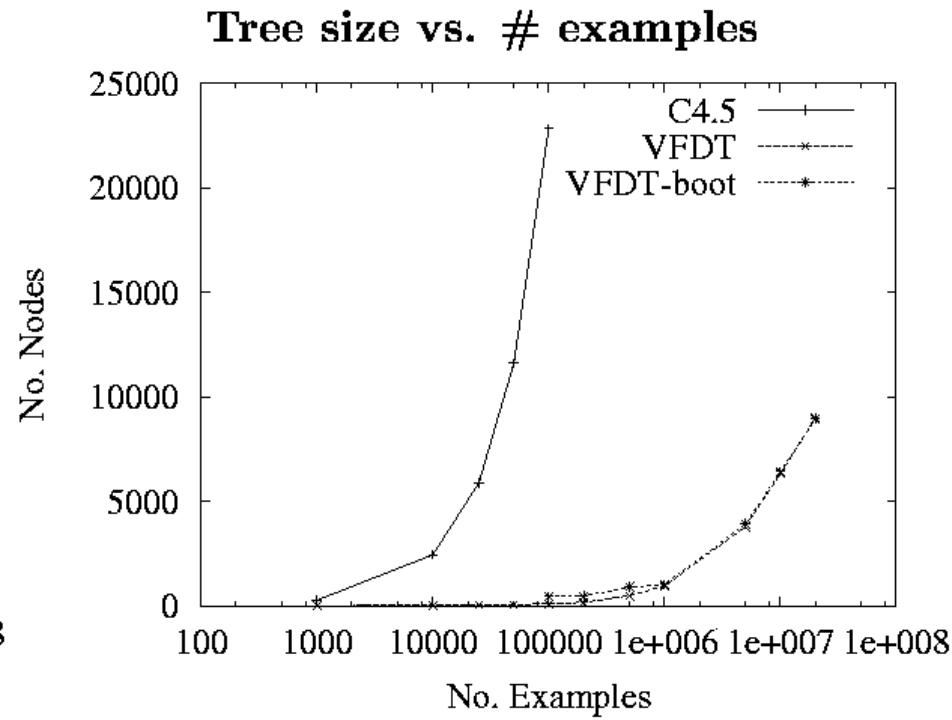
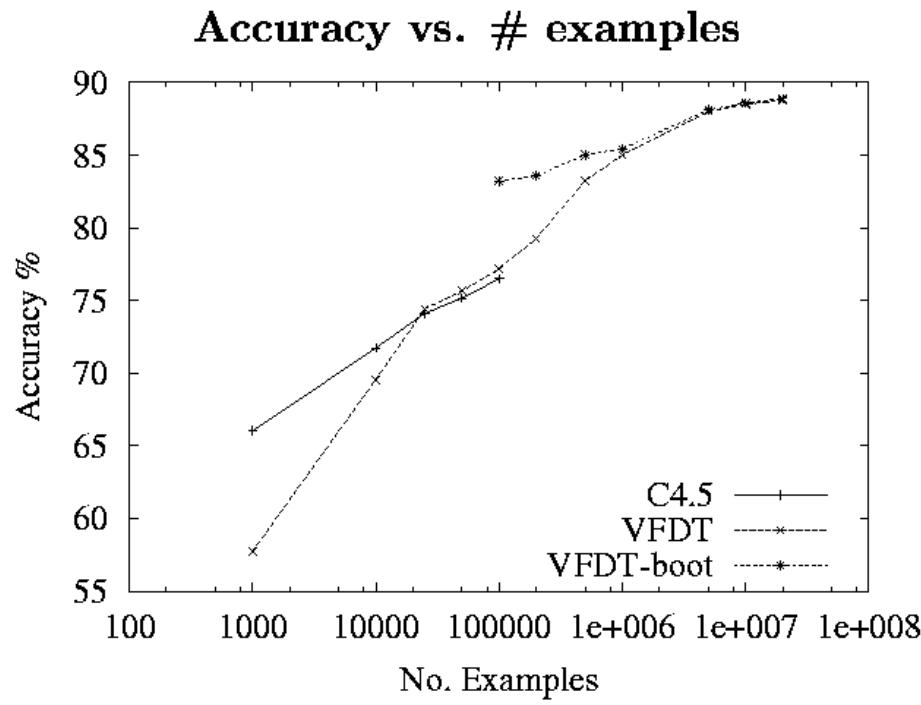
Si  $G$  es ganancia de información,  
 $R = \log c$ ,  
donde  $c$  es el número de clases

# Clasificación Incremental

- VFDT (Very Fast Decision Tree) [Domingos, Hulten, 2000]:
  - Mantiene estadísticas en los nodos hoja
  - No todos los ejemplos son procesados
  - Es incremental (los ejemplos son incorporados conforme van llegando) y el modelo está disponible en todo momento
  - No maneja atributos continuos
  - No contempla la capacidad de adaptarse a cambios en la dinámica de los datos
  - Procesa 1.6 millones de datos en 20 minutos ( $C4.5 = 24h$ )

# VFDT vs. C4.5

- VFDT:  $\delta = 10^{-7}$ ,  $\tau = 0,05$
- 2 clases, 100 atributos binarios



# Atributos Continuos

- VFML [Hulten, Domingos, 2003]
  - Resume la distribución numérica con un histograma compuesto de un número máximo de barras N. Los límites de las barras se determinan por los N primeros valores únicos vistos en el flujo. Es sensible al orden de los datos y a la elección de N
- VFDTc [Gama et al., 2003]:
  - Naïve Bayes en las hojas. Maneja atributos numéricos almacenando los puntos de corte continuos mediante *Exhaustive Binary Tree*
- Quantile Summaries [Greenwald, Khanna, 2001]
  - Mantiene valores de muestra (quantiles) más un rango de posibles puestos que las muestras pueden tomar (tuplas). Usa un número máximo de tuplas por resumen. Es muy eficiente en espacio

# Atributos Continuos

- Aproximación gaussiana

---

$weightSum = weight_{first}$

$mean = value_{first}$

$varianceSum = 0$

**for all** data points ( $value$ ,  $weight$ ) after first **do**

$weightSum = weightSum + weight$

$lastMean = mean$

$mean = mean + \frac{value - lastMean}{weightSum}$

$varianceSum = varianceSum + (value - lastMean) \times (value - mean)$

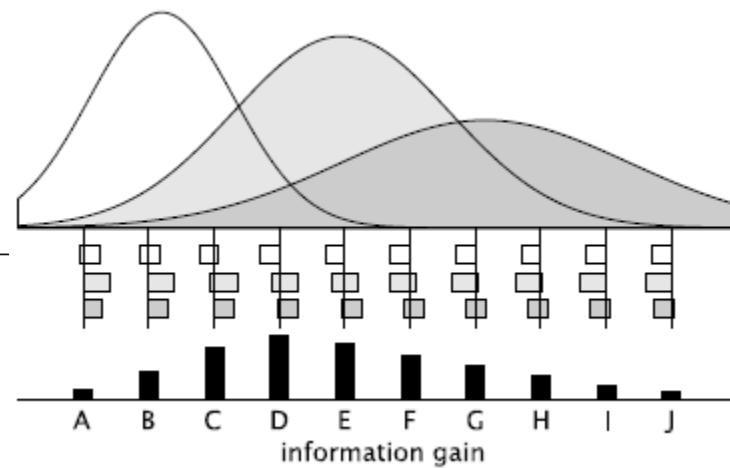
**end for**

**anytime output:**

**return**  $mean = mean$

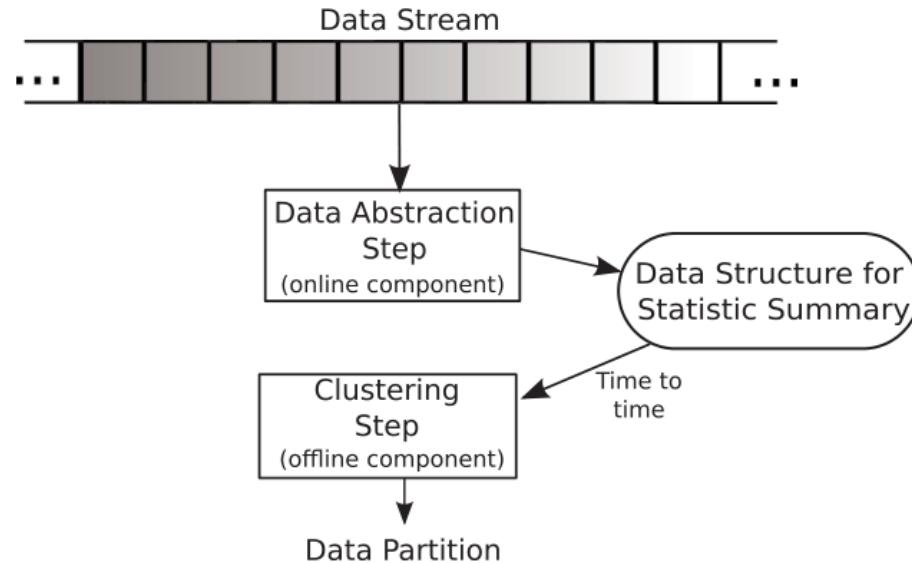
**return**  $variance = \frac{varianceSum}{weightSum - 1}$

---



# Clustering Incremental

- Los algoritmos de *clustering* incremental habitualmente dividen el proceso en dos pasos:
  1. Abstracción de datos (on-line): trata de resumir los datos que llegan conservando su esencia sin la necesidad de almacenar cada dato
  2. Agrupamiento (off-line): agrupa sobre esos resúmenes de datos. Se pueden aplicar técnicas clásicas de agrupamiento (como k-medias o DBSCAN). Al hacerse sobre los resúmenes y no sobre los datos, el proceso es eficiente



# Clustering Incremental

Algorithm	Data Structure	Window Models	Outlier Detection	Number of Parameters	Cluster Algorithm	Cluster Shape	Cluster Problem
(1) <i>BIRCH</i>	feature vector	landmark	density-based	5	<i>k</i> -means	hyper-sphere	object
(2) <i>CluStream</i>	feature vector	landmark	statistical-based	9	<i>k</i> -means	hyper-sphere	object
(3) <i>ClusTree</i>	feature vector	damped	—	3	<i>k</i> -means / <i>DBSCAN</i>	arbitrary	object
(4) <i>D-Stream</i>	grid	damped	density-based	5	<i>DBSCAN</i>	arbitrary	object
(5) <i>DenStream</i>	feature vector	damped	density-based	4	<i>DBSCAN</i>	arbitrary	object
(6) <i>DGClus</i>	grid	landmark	—	5	<i>k</i> -means	hyper-sphere	attribute
(7) <i>ODAC</i>	correlation matrix	landmark	—	3	hierarchical clustering	hyper-ellipsis	attribute
(8) <i>Scalable k-means</i>	feature vector	landmark	—	5	<i>k</i> -means	hyper-sphere	object
(9) <i>Single-pass k-means</i>	feature vector	landmark	—	2	<i>k</i> -means	hyper-sphere	object
(10) <i>Stream</i>	prototype array	landmark	—	3	<i>k</i> -median	hyper-sphere	object
(11) <i>Stream LSearch</i>	prototype array	landmark	—	2	<i>k</i> -median	hyper-sphere	object
(12) <i>StreamKM++</i>	coreset tree	landmark	—	3	<i>k</i> -means	hyper-sphere	object
(13) <i>SWClustering</i>	feature vector	landmark	—	5	<i>k</i> -means	hyper-sphere	object

- (1) *BIRCH* [Zhang et al. 1997];
- (2) *CluStream* [Aggarwal et al. 2003];
- (3) *ClusTree* [Kranen et al. 2011];
- (4) *D-Stream* [Chen and Tu 2007];
- (5) *DenStream* [Cao et al. 2006];
- (6) *DGClust* [Gama et al. 2011];
- (7) *ODAC* [Rodrigues et al. 2006; 2008];
- (8) *Scalable k-means* [Bradley et al. 1998];
- (9) *Single-pass k-means* [Farnstrom et al. 2000];
- (10) *Stream* [Guha et al. 2000];
- (11) *Stream LSearch* [O'Callaghan et al. 2002];
- (12) *StreamKM++* [Ackermann et al. 2012];
- (13) *SWClustering* [Zhou et al. 2008].

Para saber más:

[J.A. Silva, E.R. Faria, R.C. Barros, E.R. Hruschka,  
A. De Carvalho, J. Gama, Data stream clustering: a  
survey, ACM Computing Surveys 46:1 (2013) 13]  
DOI: [10.1145/2522968.2522981](https://doi.org/10.1145/2522968.2522981)

# Clustering Incremental

- BIRCH [Zhang et al., 1996]: mantiene características de los *clusters* ( $CF=\{N, LS, SS\}$ ) para resumir los objetos que van llegando

(1) *Incrementality*. A new object  $\mathbf{x}^j$  can be easily inserted into CF vector by updating its statistic summaries as follows.

$$\begin{aligned} LS &\leftarrow LS + \mathbf{x}^j \\ SS &\leftarrow SS + (\mathbf{x}^j)^2 \\ N &\leftarrow N + 1 \end{aligned}$$

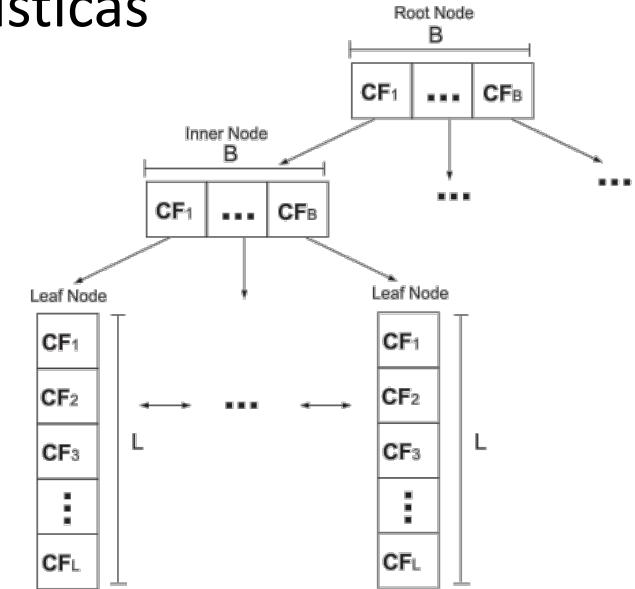
(2) *Additivity*. Two disjoint vectors  $CF_1$  and  $CF_2$  can be easily merged into  $CF_3$  by summing up their components.

$$\begin{aligned} N_3 &= N_1 + N_2 \\ LS_3 &= LS_1 + LS_2 \\ SS_3 &= SS_1 + SS_2 \end{aligned}$$

$$centroid = \frac{LS}{N}$$

$$radius = \sqrt{\left(\frac{SS}{N} - \left(\frac{LS}{N}\right)^2\right)}$$

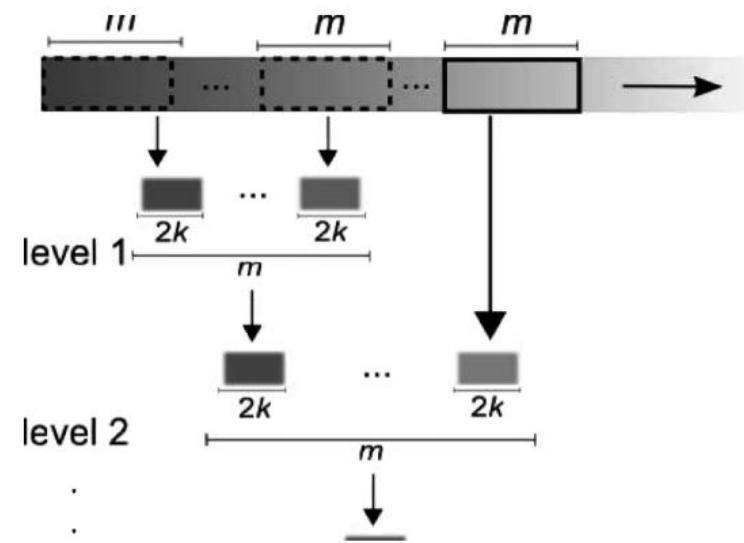
$$diameter = \sqrt{\left(\frac{2N * SS - 2 * LS^2}{N(N - 1)}\right)}$$



Cada vez que llega un objeto, desciende por el árbol escogiéndose en cada nodo el CF más cercano. Cuando llega a una hoja, si puede ser absorbido por algún CF existente (porque esté en su radio menor que  $T$ ) se agrega, si no, se crea un CF nuevo si hay menos de  $L$ . Si se alcanza el máximo  $L$ , se divide la hoja en dos con el par de CF más lejanos de la hoja anterior.

# Clustering Incremental

- En Stream [Guha et al., 2000] se propone un enfoque basado en divide y vencerás para poder abordar con eficiencia grandes cantidades de datos de forma incremental mediante una adaptación de k-medias
- El algoritmo emplea un esquema jerárquico donde se aplica *clustering* por segmentos de datos para luego agrupar los centroides (que pasan a considerarse como puntos ponderados por el tamaño del *cluster* que representan) en un segundo nivel y así hasta obtener el número de *clusters* deseado



# *Clustering Incremental*

- Stream no es eficiente pues obliga a almacenar todo en memoria, de modo que se propone otro enfoque (Stream LSearch) donde se determina el tamaño de la muestra y se aplica búsqueda local si el tamaño supera un umbral. El proceso se repite para cada segmento de datos
- En cualquier caso, estos enfoques no consideran la evolución de los datos ni la granularidad del tiempo. Datos obsoletos pueden dominar sobre datos recientes
- En problemas reales, la naturaleza de los *clusters* puede variar según el momento en el que se calculan (*concept drift*) así como sobre el horizonte de tiempo en el que se miden. En la siguiente sección veremos otras propuestas que sí tienen en cuenta esto

# Minería de Flujo de Datos (no estacionario)

# Flujo de Datos

- Un flujo de datos es una secuencia ordenada de información que no se almacena y, por tanto, se lee una sola vez (o un pequeño número de veces) usando capacidades de cómputo y almacenamiento limitados
- Algunas características de estos datos son: flujo **continuo** y, en principio, infinito; **ordenado**; tasa de **llegada muy alta**; **cambiente** a lo largo del tiempo, es decir, con una distribución no estacionaria; volumen **inmenso**; de **solo lectura**; con requerimientos de **respuesta inmediata**...
- La **minería de flujo de datos** se encarga de manejar esta información a través del proceso de extraer conocimiento dinámicamente a partir de los datos conforme éstos se van recibiendo

# Flujo de Datos

- Un flujo de datos es una secuencia ordenada de información que no se almacena y, por tanto, se lee una sola vez (o un pequeño número de veces) usando capacidades de cómputo y almacenamiento limitados.



Gödel Prize 2005 for  
their foundational contribution to  
streaming algorithms

- Algunos de los primeros resultados fundamentales en el campo de los algoritmos para flujo de datos fueron obtenidos por Noga Alon, Yossi Matias, y Mario Szegedy, estos autores presentaron el resultado titulado "The space complexity of approximating the frequency moments, which required space proportional to  $\sqrt{m}$ " en la revista Journal of Computer and System Sciences 58 (1999) 137-147.

- La **minería de flujo de datos** se encarga de manejar esta información a través del proceso de extraer conocimiento dinámicamente a partir de los datos conforme éstos se van recibiendo.

# Flujo de Datos

- Podemos distinguir dos modos de caracterizar este campo:
  - Según los modelos y técnicas computacionales para gestionar este tipo de datos: con independencia del problema concreto a resolver, se recurre a diferentes soluciones para almacenar, mantener y tratar el flujo de datos
  - *Data stream mining* (minería de flujo de datos): Según el problema de aprendizaje que se resuelve. Pretende abordar este problema mediante procesos de extracción de conocimiento a partir de registros de datos recibidos de forma continua y rápida

# Modelos Estadísticos y Computacionales para Flujo de Datos

- Basados en datos:
  - *Sampling*: decisión basada en probabilidades de procesar un dato o no
  - *Load shedding*: proceso de ignorar una secuencia de datos
  - *Sketching*: proyección de subconjuntos de características (especialmente para comparar secuencias de datos)
  - *Synopsis*: técnicas de integración capaces de resumir el flujo de datos para su posterior análisis (histogramas, frecuencias, cuantiles...)
  - Agregación: cálculos estadísticos como medias y varianzas para resumir los datos

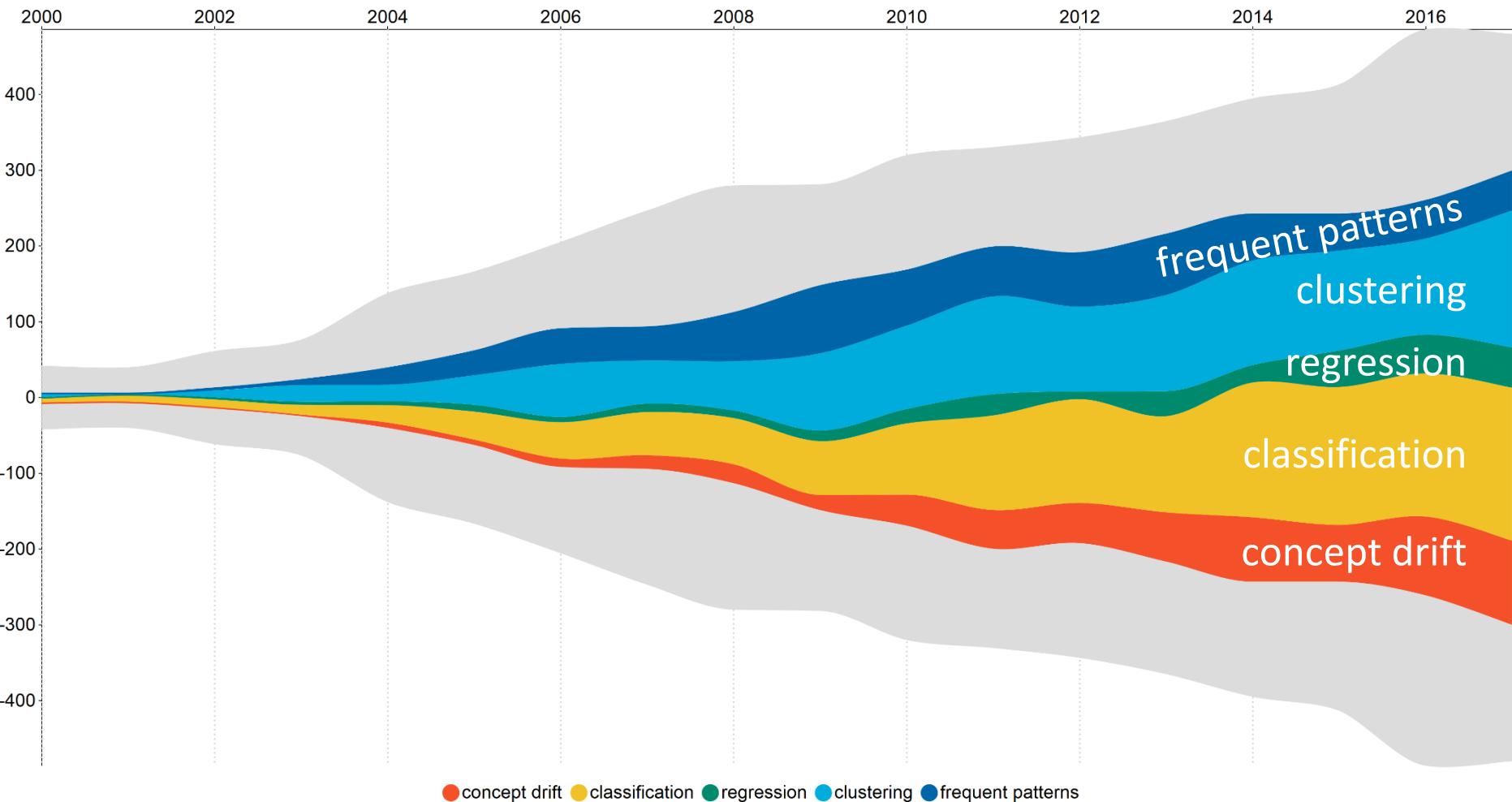
# Modelos Estadísticos y Computacionales para Flujo de Datos

- Basados en tareas:
  - Algoritmos aproximados: basados en soluciones aproximadas con o sin límites de error
  - Ventana deslizante: centra el interés en los datos más recientes, haciendo un análisis detallado sobre estos datos y apoyándose en versiones resumidas de los datos anteriores
  - *Algorithm output granularity* (AOG): análisis previo de los datos en función de las restricciones de memoria y tiempo de procesado (minería, adaptación y fusión de conocimiento)

# Minería de Flujo de Datos

- Principales tareas en minería de flujo de datos:
  - **Clasificación**: aprendizaje supervisado; se asume la existencia de un flujo de datos etiquetados con una clase y los algoritmos tratan de maximizar la precisión de predicción al tiempo que reaccionan y se adaptan a situaciones cambiantes
  - **Agrupamiento**: es más difícil adaptar los *clusters* a flujo de datos debido a las limitaciones de analizar cada dato solo una vez, pero existen algoritmos de *clustering* incremental incluso antes de este campo de investigación en flujo de datos
  - **Patrones Frecuentes**: se mantienen conjuntos de elementos frecuentes de modo incremental conforme se recibe el flujo de datos; también pueden mantenerse reglas de asociación
- Los modelos de representación son similares a los de datos estáticos (reglas, árboles de decisión, grupos, patrones...)

# Evolución del Número de Publicaciones de Algunas categorías de Minería de Flujo de Datos



Fuente: <http://www.scopus.com>

1. ***Concept drift*** en flujo de datos
2. Clasificación en flujo de datos
3. *Clustering* en flujo de datos
4. Patrones frecuentes en flujo de datos

# *Concept Drift (Desvío de Concepto)*

- Una de las principales dificultades que se abordan en flujo de datos es el desvío de concepto o *concept drift*
- La ley subyacente en los datos cambia gradual o abruptamente. Es decir, el modelo construido en el pasado ya no es consistente con los datos que se reciben en el presente
- Un tratamiento adecuado de este efecto significa que datos recientes que entren en conflicto con asunciones pasadas deben priorizar la construcción del modelo. Pero, al mismo tiempo, asunciones basadas en datos antiguos que aún sean consistentes deben conservarse

# Concept Drift (Desvío de Concepto)

- Este cambio en la dinámica de los datos puede deberse a diferentes razones tales como ruido, influencia del entorno, variación en características no contempladas en el modelo, alteraciones estacionales...
- Al mismo tiempo, los cambios pueden ser graduales, abruptos, recurrentes...

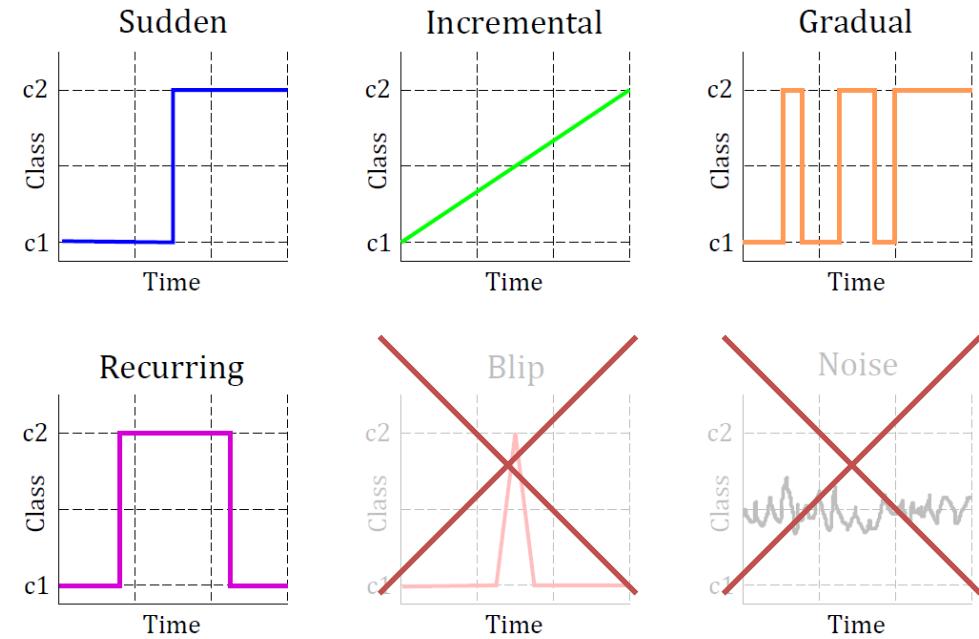
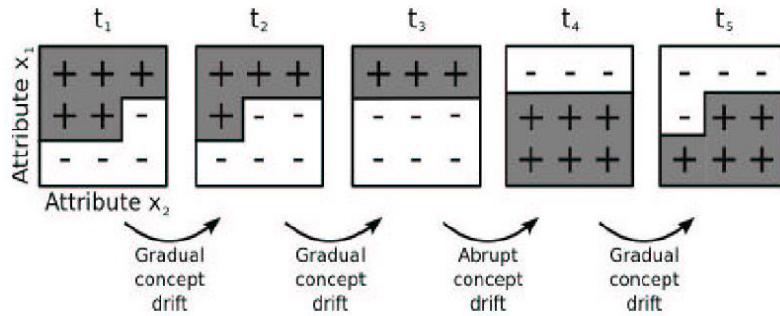
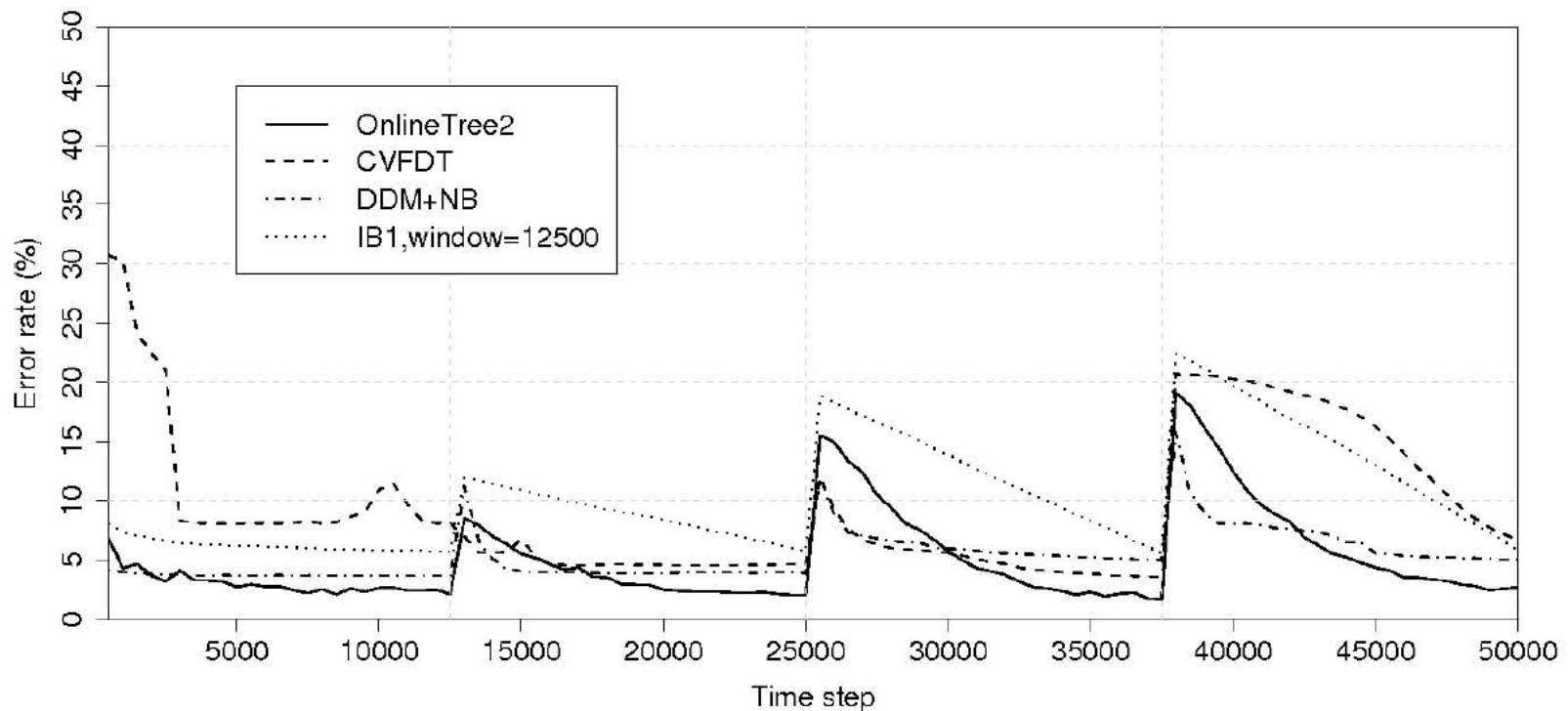


Figura tomada del TFM de D. Brzeziński (2010)

# *Concept Drift (Desvío de Concepto)*

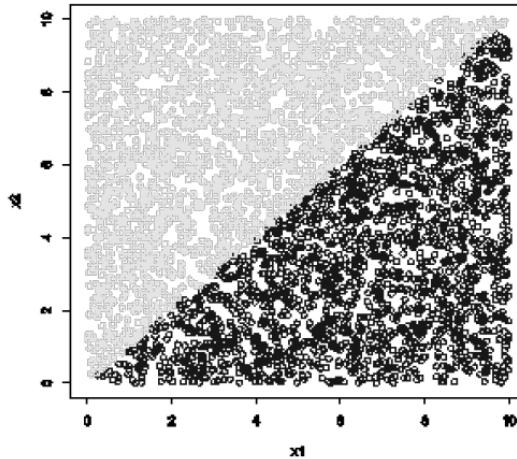
- Ejemplo de resultados en SEA, un conjunto de datos sintético muy popular en este campo



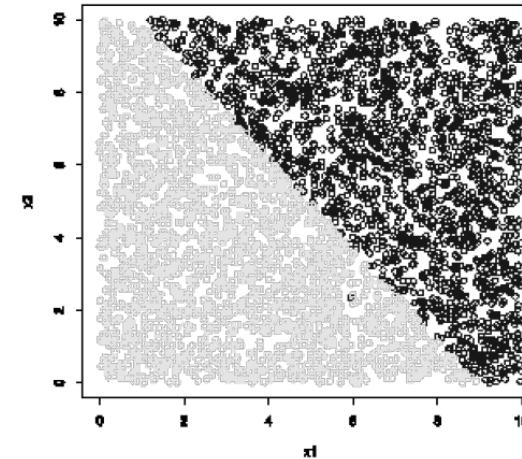
[Nuñez et al., 2007]

# *Concept Drift (Desvío de Concepto)*

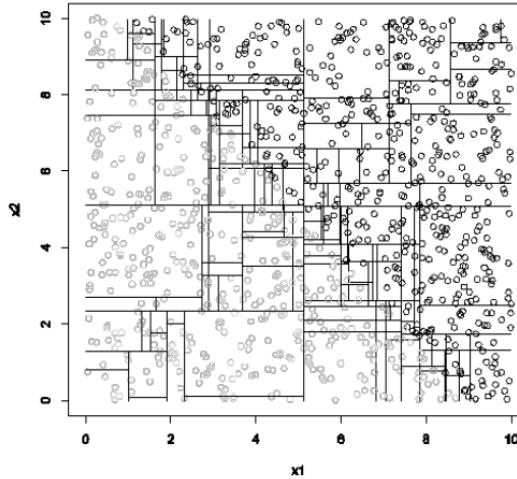
Concept 1



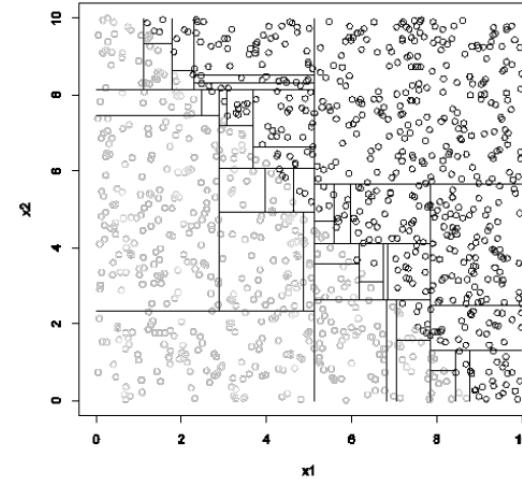
Concept 2



Sin detección del CD ni adaptación



Con detección y adaptación al CD

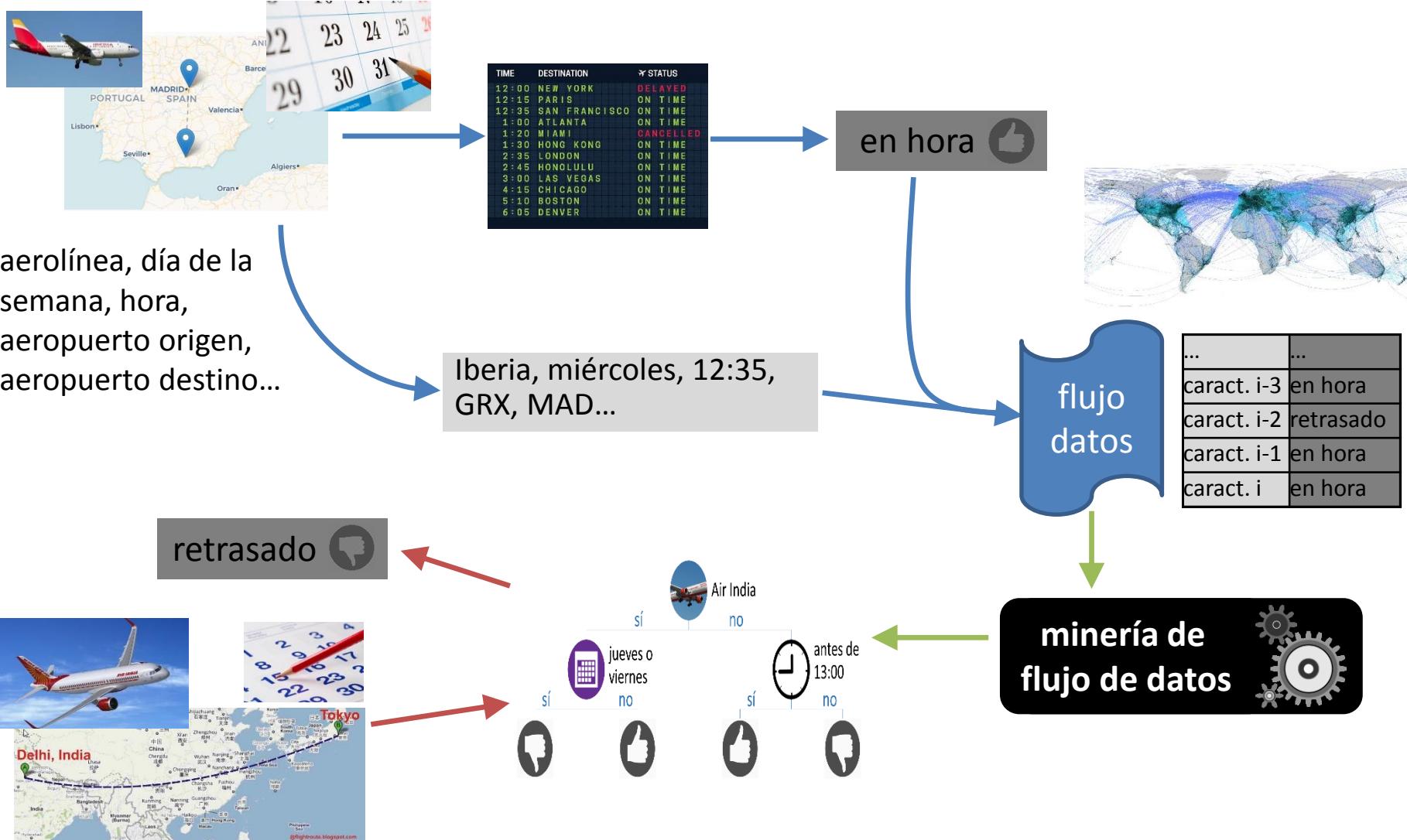


1. *Concept drift* en flujo de datos
2. **Clasificación en flujo de datos**
3. *Clustering* en flujo de datos
4. Patrones frecuentes en flujo de datos

# Clasificación en Flujo de Datos

- Actualmente, es quizás el problema más ampliamente estudiado en el contexto de la minería de flujo de datos
- Las primeras aproximaciones se basan en algoritmos de aprendizaje incremental (online, secuencial) donde no se dispone de todos los datos previos, no todos residen en memoria o el aprendizaje no finaliza al obtener el modelo sino que se adapta continuamente
- Sin embargo, solo se consideran de flujo de datos en sentido estricto cuando se dan a la vez todas esas características, y rigurosamente cuando cada dato se analiza una sola vez y en el momento en el que se recibe

# Aprendizaje Supervisado en Flujo de Datos



# *Concept Drift (CD) en Clasificación*

- Existen algunas primeras aproximaciones que manejaban desvíos de concepto (STAGGER, IB3, FLORA...)
- Se ha desarrollado mucha investigación en este caso; podemos dividir estos algoritmos principalmente en:
  1. Aprendizaje online
  2. Soluciones basadas en instancias (ventana)
  3. Aprendizaje de múltiples modelos (*ensemble*)
  4. Algoritmos de detección del desvío

# CD mediante Aprendizaje Online

- Son una familia de algoritmos que continuamente actualizan los parámetros del clasificador mientras procesan los datos que van llegando
- No todos los clasificadores pueden actuar como aprendizaje online, tienen que cumplir una serie de requisitos básicos:
  - Cada objeto debe procesarse solo una vez durante el entrenamiento
  - El sistema debe consumir una memoria y procesamiento limitados con independencia de la cantidad de datos procesada
  - El aprendizaje se puede pausar en cualquier momento y su precisión no debería ser peor que la de un clasificador entrenado offline sobre los datos vistos hasta el momento

# CD mediante Aprendizaje Online

- CVFDT (Concept-adapting VFDT)  
[Hulten et al., 2001]:

- Mantiene el árbol consistente con una ventana deslizante
- Mantiene estadísticas en todos los nodos del árbol para las distintas divisiones candidatas
- Comienza a construir ramas del árbol alternativas cuando otras divisiones son mejores
- Periódicamente evalúa la validez del modelo con nuevos ejemplos y sustituye un nodo por la rama alternativa si mejora
- Tarda entre 5 y 6 veces más que VFDT
- No maneja atributos continuos

CVFDT(*Stream*,  $\delta$ )

- 1 Let HT be a tree with a single leaf(root)
- 2 Init counts  $n_{ijk}$  at root
- 3 for each example  $(x, y)$  in Stream
  - 4 do Add, Remove and Forget Examples
  - 5 CVFDTGROW( $(x, y), HT, \delta$ )
  - 6 CHECKSPLITVALIDITY( $HT, n, \delta$ )

CVFDTGROW( $(x, y), HT, \delta$ )

- 1 Sort  $(x, y)$  to leaf  $l$  using  $HT$
- 2 Update counts  $n_{ijk}$  at leaf  $l$  and nodes traversed in the sort
- 3 if examples seen so far at  $l$  are not all of the same class
  - 4 then Compute  $G$  for each attribute
  - 5 if  $G(\text{Best Attr.}) - G(\text{2nd best}) > \sqrt{\frac{R^2 \ln 1/\delta}{2n}}$
  - 6 then Split leaf on best attribute
    - 7 for each branch
      - 8 do Start new leaf and initialize counts
      - 9 Create alternate subtree

CHECKSPLITVALIDITY( $HT, n, \delta$ )

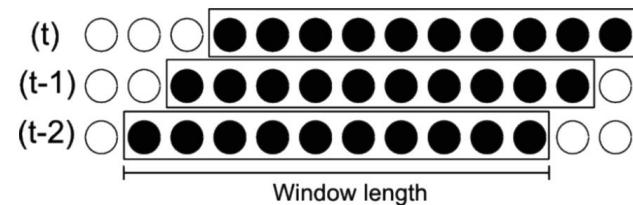
- 1 for each node  $l$  in  $HT$  that it is not a leaf
  - 2 do for each tree  $T_{alt}$  in  $ALT(l)$ 
    - 3 do CHECKSPLITVALIDITY( $T_{alt}, n, \delta$ )
    - 4 if exists a new promising attributes at node  $l$ 
      - 5 do Start an alternate subtree

# CD mediante Ventana

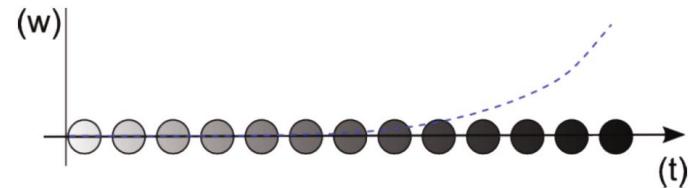
- Son algoritmos que incorporan mecanismos para olvidar datos antiguos
- Se basan en asumir que los datos llegados recientemente son más relevantes porque contienen características del contexto actual y su relevancia disminuye con el paso del tiempo
- Por tanto, ajustar el rango de datos a aquellos que han llegado más recientemente puede ayudar a formar un conjunto de datos que representa al contexto actual

# CD mediante Ventana

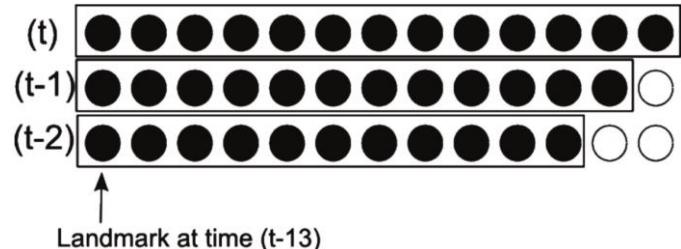
1. Seleccionar instancias por medio de una ventana deslizante que descarta datos antiguos [Widmer, 1996] (dificultad para fijar el tamaño de la ventana)



2. Ponderar los datos según su relevancia o antigüedad (*damped window, facing/decay factor*)



3. Determinar las ventanas según hitos (*landmarks*) en los datos (p.ej., del día actual, semana actual, etc.)



# CD mediante *Ensemble*

- *Ensemble* son algoritmos que incorporan conjuntos de clasificadores elementales (combinación de modelos)
- Se ha demostrado que una decisión colectiva puede incrementar la precisión porque el conocimiento distribuido entre los clasificadores puede ser más exhaustivo
- Esta premisa es cierta si se dispone de un conjunto de clasificadores suficientemente diverso [Shipp, 2002]
- En entornos estáticos, la diversidad se puede referir al modelo de clasificador, el conjunto de características o las instancias usadas en el entrenamiento
- En un entorno cambiante, la diversidad puede referirse también al contexto

# CD mediante *Ensemble*

- Entre los enfoques *ensemble* para flujo de datos más populares, podemos destacar:
  - Streaming Ensemble Algorithm (SEA) [Street, 2001]
  - Accuracy Weighted Ensemble (AWE) [Wang, 2003]
- Ambos algoritmos mantienen un número fijo de clasificadores. Los datos que van llegando se recogen en fragmentos que se usan para entrenar nuevos clasificadores
- Los clasificadores se evalúan según su precisión y el peor es reemplazado por uno nuevo si este último es más preciso
- SEA usa voto por mayoría, mientras que AWE usa un voto ponderado más sofisticado

# CD mediante Ensemble

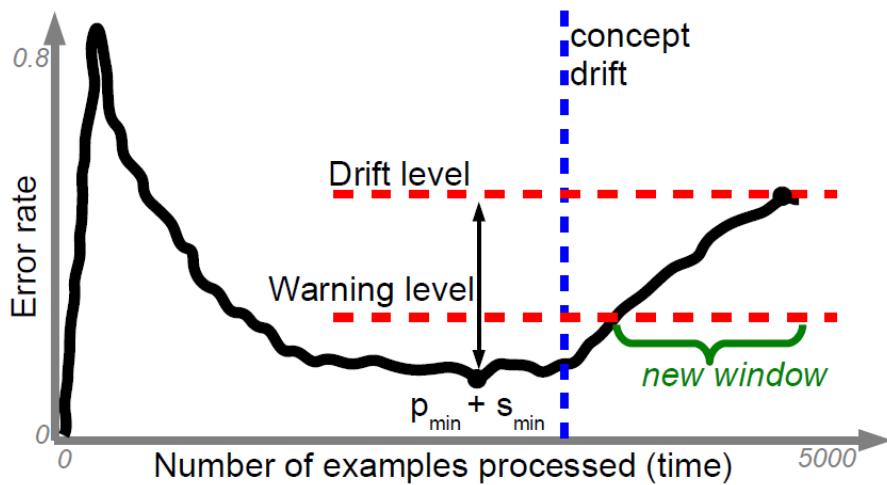
- Dynamic Weighted Majority (DWM) [Kolter, 2003] pondera los clasificadores en función de sus decisiones incorrectas. El número de clasificadores es variable; clasificadores con un peso por debajo de un umbral se eliminan, mientras que se pueden añadir clasificadores cuando el conjunto toma una decisión errónea
- En [Jackowski, 2013] se usa una bolsa de clasificadores que se actualiza cuando se detecta CD pero no se eliminan clasificadores, de forma que el conocimiento del pasado se puede conservar para el futuro
- Para saber más: [B. Krawczyk, L.L. Minku, J. Gama, J. Stefanowski, M. Wozniak, Ensemble learning for data stream analysis: A survey, *Information Fusion* 37 (2017) 132-156] DOI: [10.1016/j.inffus.2017.02.004](https://doi.org/10.1016/j.inffus.2017.02.004)

# Detección de CD

- En lugar de dotar de adaptabilidad al proceso de aprendizaje, otra alternativa es detectar cuándo se produce el CD y actuar para paliar sus efectos (por ejemplo, reseteando total o parcialmente el conocimiento adquirido por el algoritmo)
- El objetivo de cualquier detector de CD es reconocer los cambios en flujos de datos no estacionarios de manera precisa y oportuna
- Para ello, la mayoría de las investigaciones analizan la disminución en el rendimiento predictivo como un signo de cambio en los datos
- Para saber más: [J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, A survey on concept drift adaptation, ACM Computing Surveys 46:4 (2014) 44] DOI: [10.1145/2523813](https://doi.org/10.1145/2523813)

# Detección de CD: DDM

- Asumen la posibilidad de conocer el rendimiento del modelo que se está aprendiendo
- Detectan el desvío de concepto en función de la precisión del clasificador o mediante un análisis de distribución de la clase [Gama et al. 2006; Baena-García et al., 2006; Alippi, Roveri, 2008]



Drift detection method (DDM)  
[Gama et al. 2004]

probability of missclassifying ( $p_i$ )

$$s_i = \sqrt{p_i(1 - p_i)/i}$$

*Warning level:*  $p_i + s_i \geq p_{min} + 2 \cdot s_{min}$   
Se empieza a almacenar ejemplos

*Drift level:*  $p_i + s_i \geq p_{min} + 3 \cdot s_{min}$   
Se construye un nuevo modelo a partir de los datos almacenados. Se reinician  $p_{min}$  y  $s_{min}$

- Estos algoritmos pueden ser ineficientes y, por tanto, incapaces de asumir una tasa de llegada de datos elevada

# Detección de CD: ADWIN

- Existen propuestas para detectar CD y adaptar la longitud de la ventana como ADWIN (ADaptive sliding WINdow) [Bifet, 2009]
- Dada la ventana  $W$ , si existen dos subventanas  $W_0$  y  $W_1$  suficientemente grandes y con medias suficientemente distintas, podemos concluir que los valores esperados son diferentes y se puede eliminar la parte antigua de  $W$

1 Initialize Window  $W$   
2 for each  $t > 0$

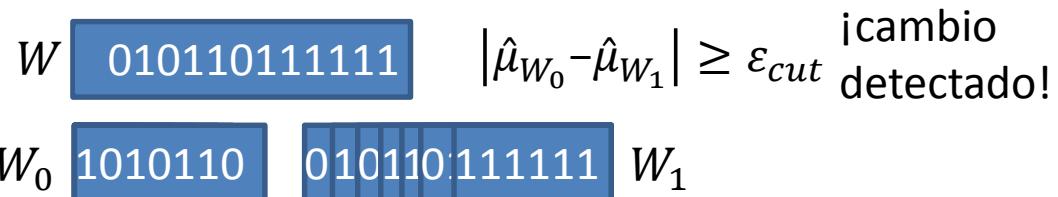
3 do  $W \leftarrow W \cup \{x_t\}$  (i.e., add  $x_t$  to the head of  $W$ )

4 repeat Drop elements from the tail of  $W$

5 until  $|\hat{\mu}_{W_0} - \hat{\mu}_{W_1}| < \epsilon_{cut}$  holds

6 for every split of  $W$  into  $W = W_0 \cdot W_1$

7 output  $\hat{\mu}_W$



$$\epsilon_{cut} = \sqrt{\frac{2\lambda}{m} \ln \frac{2}{\delta}}$$

$$1/m = 1/n_0 + 1/n_1$$

$\lambda$  media de los datos de la ventana

# Detección de CD

- Las propuestas convencionales de detección del CD vincula la detección con el rendimiento del algoritmo de aprendizaje, pudiéndose dar la situación paradójica de que los algoritmos capaces de adaptarse autónomamente a CD pueden no ser útiles para detectar CD débiles que no afectan radicalmente la capacidad de predicción
- Por supuesto, el objetivo principal de detección de CD es mejorar el rendimiento de predicción, pero tener el conocimiento sobre cuándo ocurren los CD también es una información valiosa para la toma de decisiones en problemas reales
- Además, una detección efectiva de estos CD ayudaría a los algoritmos adaptativos a reaccionar aún más rápido y mejor
- Por último, pero no menos importante, un inconveniente importante de la detección convencional de CD es la eficiencia. Necesar conocer el rendimiento del algoritmo para detectar CD ralentiza significativamente el proceso, lo que puede hacer que estos métodos sean incapaces de abordar problemas del mundo real con un gran número de muestras por segundo y donde se exige una respuesta rápida (*high-speed data stream*)

Igual que la serpiente detecta a la iguana solo cuando se mueve, el CD se puede detectar cuando varían las fronteras de decisión

<https://youtu.be/Rv9hn4IGofM>



# Algoritmo HSC de Detección de CD

J. Casillas, S. Wang, X. Yao, Concept Drift Detection in Histogram-Based Straightforward Data Stream Classification, 2017

Proponemos una solución que basa la detección exclusivamente en las muestras, sin necesidad de monitorizar al algoritmo, con las siguientes ventajas:

- Complejidad de tiempo y espacio  $O(n \cdot m)$  –siendo  $n$  el número de características y  $m$  el número de clases– para procesar cada muestra: es extremadamente eficiente
- Implementación e integración ágil, lo que facilita la creación de prototipos y el análisis exploratorio de nuevos problemas de flujo de datos. Su diseño es simple, por lo que es fácil de entender su comportamiento y depurar el proceso
- La verdadera detección de CD, incluso cuando los algoritmos eficaces no se degradan significativamente, puede mejorar el proceso de predicción y obtener una mayor comprensión sobre la naturaleza del problema
- En último término, HSC es compatible con el uso de detectores de CD tradicionales basados en el rendimiento del algoritmo, por lo que se podría aprovechar la información provista por este método alternativo para corroborar detecciones o adaptarse dinámicamente a las características del problema

# Algoritmo HSC de Detección de CD

Pero si cae aquí, el poder discriminante será alto y si es el mayor entre todas las características, el clasificador dirá que es de la clase  $c_1$



si el valor cae aquí, esta variable tendrá poco poder discriminante ( $v_{ij}$  bajo)

Debido al uso de histogramas univariantes, el clasificador puede funcionar mal si las fronteras de decisión son ortogonales a los ejes. Pero eso no importa, solo se utiliza para detectar variaciones de esas fronteras con independencia de que sea para mejorar o empeorar la capacidad de predicción (G)

## Mantenimiento de histogramas

$$\mathbf{H}_{ij}^t = \alpha(t) \cdot \mathbf{H}_{ij}^{t-1} + \mathbf{I}_{ij}^t, \quad \mathbf{H}_{ij}^0 = 0 \quad (1a)$$

$$\mathbf{I}_{ij}^t = \left( K_{A_1}(x_i^t, y^t, c_j), \dots, K_{A_{b_i}}(x_i^t, y^t, c_j) \right) \quad (1b)$$

$$K_A(x, y, c) = \begin{cases} 1, & y = c \text{ and } x \in A \\ 0, & \text{otherwise} \end{cases} \quad (1c)$$

$$A_k^i = \begin{cases} [a_{k-1}^i, a_k^i], & \forall k \in \{1, \dots, b_i - 1\} \\ [a_{k-1}^i, a_k^i], & k = b_i \end{cases} \quad (1d)$$

$$a_k^i = \min_i + k \cdot s_i, \quad s_i = \frac{\max_i - \min_i}{b_i} \quad (1e)$$

$$\sum_k H_{ijk}^t = \frac{1}{1 - \alpha(t)} = \delta \quad (2)$$

$$\alpha(t) = \begin{cases} 1, & 1 \leq t \leq \delta \\ \frac{\delta - 1}{\delta}, & t > \delta \end{cases} \quad (3)$$

## Clasificador elemental basado en el histograma

$$\phi_i = k \text{ s.t. } x_i \in A_k^i \quad (4a)$$

$$\varphi_i = \begin{cases} k - 1, & x_i \in A_k^i, k > 1, x_i < \frac{a_{k-1}^i + a_k^i}{2} \\ k + 1, & x_i \in A_k^i, k < b_i, x_i > \frac{a_{k-1}^i + a_k^i}{2} \\ k, & \text{otherwise} \end{cases} \quad (4b)$$

$$w_i = \begin{cases} 0.5 + \frac{x_i - a_{k-1}^i}{\max_i - \min_i}, & \varphi_i = k - 1 \\ 1.5 + \frac{a_k^i - x_i}{\max_i - \min_i}, & \varphi_i = k + 1 \\ 1, & \text{otherwise} \end{cases} \quad (4c)$$

$$h_{ij} = \frac{w_i H_{ij}\phi_i + (1 - w_i)H_{ij}\varphi_i}{\sum_k H_{ijk}}, \quad \forall j \in \{1, \dots, m\} \quad (4d)$$

$$\nu_{ij} = \frac{h_{ij}}{\sum_k h_{ik}}, \quad \vartheta_j = \max_i \nu_{ij} \quad (5)$$

$$\hat{y} = c_k \text{ s.t. } k = \arg \max_j \vartheta_j \quad (6)$$

## Cálculo de rendimiento del clasificador elemental

$$G^t = \sqrt{\frac{M_{tp}^t}{M_{tp}^t + M_{fn}^t} \cdot \frac{M_{tn}^t}{M_{tn}^t + M_{fp}^t}} \quad (7a)$$

$$\mathbf{M}^t = (M_{tp}^t, M_{fn}^t, M_{tn}^t, M_{fp}^t), \quad \mathbf{M}^0 = 0 \quad (7b)$$

$$M_{tp}^t = \alpha \cdot M_{tp}^{t-1} + tp^t, \quad M_{fn}^t = \alpha \cdot M_{fn}^{t-1} + fn^t, \quad (7c)$$

$$M_{tn}^t = \alpha \cdot M_{tn}^{t-1} + tn^t, \quad M_{fp}^t = \alpha \cdot M_{fp}^{t-1} + fp^t \quad (7d)$$

$$tp^t = \begin{cases} 1, & y^t = \hat{y}^t = c_1 \\ 0, & \text{otherwise} \end{cases}, \quad fn^t = \begin{cases} 1, & y^t = c_1, \hat{y}^t = c_2 \\ 0, & \text{otherwise} \end{cases} \quad (7e)$$

$$tn^t = \begin{cases} 1, & y^t = \hat{y}^t = c_2 \\ 0, & \text{otherwise} \end{cases}, \quad fp^t = \begin{cases} 1, & y^t = c_2, \hat{y}^t = c_1 \\ 0, & \text{otherwise} \end{cases} \quad (7f)$$

# Algoritmo HSC de Detección de CD

**Parameters:**  $\delta, \alpha = \frac{\delta-1}{\delta}, b_i, \epsilon = 0.01$   
**Static variables:**  $t \leftarrow 1, t_{CD}, p_i, \sigma_i, p_{min}, \sigma_{min}, p_{max}, \sigma_{max}$ ,  
**Data:**  $\mathbf{H}^{t-2}, \mathbf{M}^{t-2}, \mathbf{x}^{t-1}$   
**Input:**  $\mathbf{x}^t, y^{t-1}$   
**Output:**  $(t, \{\text{true, false}\})$

```

1 begin
2   concept_drift ← false
3   if  $t > 1$  then
4     Obtain  $\hat{y}^{t-1}$  from  $\mathbf{H}^{t-2}$  and  $\mathbf{x}^{t-1}$            // eqs. (4-6)
5     Update  $\mathbf{M}^{t-1}$  from  $\mathbf{M}^{t-2}, \hat{y}^{t-1}$ , and  $y^{t-1}$  // eqs. (7b-7f)
6     if  $t_{CD} > 0.25 \cdot \delta$  then
7       Compute  $G^{t-1}$  from  $\mathbf{M}^{t-1}$                 // eq. (7a)
8        $(p_i, \sigma_i) \leftarrow \text{IMSD}(1 - G^{t-1}, t_{CD}, \alpha)$  // Algorithm 1
9     else
10       $p_i \leftarrow 0.25, \sigma_i \leftarrow 0$ 
11    if ( $t_{CD} > 0.5 \cdot \delta$ ) then
12      if  $p_i + \sigma_i < p_{min} + \sigma_{min}$  then
13         $p_{min} \leftarrow p_i, \sigma_{min} \leftarrow \sigma_i$ 
14      if  $p_i - \sigma_i > p_{max} - \sigma_{max}$  then
15         $p_{max} \leftarrow p_i, \sigma_{max} \leftarrow \sigma_i$ 
16        if  $t_{CD} \leq 0.5 \cdot \delta + 1$  then
17           $\text{first\_max} \leftarrow p_{max}$ 
18      increase ←  $(p_i + \sigma_i \geq p_{min} + 3\sigma_{min} + \epsilon \text{ and } t_{CD} > \delta)$ 
19      decrease ←  $(p_i - \sigma_i \leq p_{max} - 3\sigma_{max} - \epsilon \text{ and } t_{CD} > \delta)$ 
20      if decrease and (increase or
21         $((p_{min} - p_i > 2\epsilon) \text{ and } (p_{max} - \text{first\_max} < 2\epsilon)))$  then
22        concept_drift ← true
23    if  $t = 1$  or concept_drift then
24       $p_i \leftarrow 0, \sigma_i \leftarrow 0$ 
25       $p_{min} \leftarrow 1, \sigma_{min} \leftarrow 0$ 
26       $p_{max} \leftarrow 0, \sigma_{max} \leftarrow 0$ 
27      first_max ← 0
28       $\mathbf{M}^{t-1} \leftarrow 0$ 
29       $\mathbf{H}^{t-1} \leftarrow 0$ 
30       $t_{CD} \leftarrow 0$ 
31      Reset static variables in Algorithm 1
32    else
33      Update  $\mathbf{H}^{t-1}$  from  $\mathbf{H}^{t-2}$  and  $(\mathbf{x}^{t-1}, y^{t-1})$ 
34       $t \leftarrow t + 1$ 
35       $t_{CD} \leftarrow t_{CD} + 1$ 
return  $(t - 1, \text{concept\_drift})$ 
```

El cálculo incremental de la media y desviación típica se hace con factor de decaimiento para una reacción más rápida y mejor detección de variaciones

Los cálculos de error medio y desviación típica comienzan cuando el histograma ha acumulado suficientes datos

También se monitorizan los valores máximos pues estamos interesados en fluctuaciones del clasificador incluso si es para mejorar

El CD se produce cuando ha habido un descenso y ascenso del error (es decir, se ha formado un valle o pico)

No obstante, si el error decrece mucho, también se considera un CD pues puede deberse a un cambio de la frontera que favorece al clasificador

HSC tiene una complejidad de tiempo  $O(n \cdot m)$  debido a las líneas 4 y 32, siendo  $n$  el número de características y  $m$  el número de clases. La complejidad de espacio también es  $O(n \cdot m)$ , que es lo que ocupa el histograma  $\mathbf{H}$

**Algorithm 1:** Incremental computation of mean and standard deviation with decay factor

```

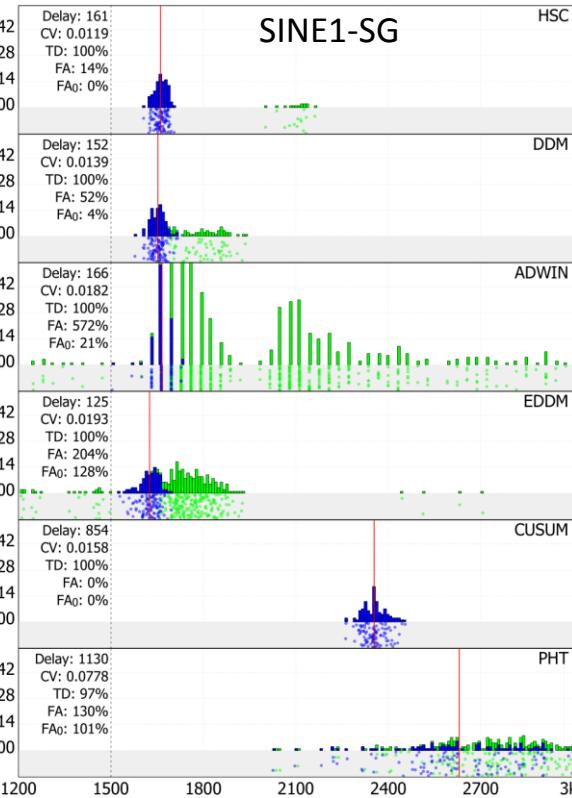
1 Function IMSD( $x, n, \alpha$ ):
2    $\rho \leftarrow x - \bar{x}$ 
3   temp ← sum ·  $\alpha + 1$ 
4    $R \leftarrow \rho / \text{temp}$ 
5    $\bar{x} \leftarrow \bar{x} + R$ 
6    $M_2 \leftarrow M_2 \cdot \alpha + (\text{sum} \cdot \alpha \cdot \rho \cdot R)$ 
7   sum ← temp
8    $\sigma \leftarrow \sqrt{(M_2 \cdot n) / (\text{sum} \cdot (n - 1))}$ 
9   return  $(\bar{x}, \sigma)$ 
```

No se actualizan los valores mínimos y máximos hasta que la media no se haya estimado con suficientes datos

# Algoritmo HSC de Detección de CD

**Strong drift**

ds	method	delay	CV	TD	FA	FA <sub>0</sub>	MTFA	MTR	rank
SINE1-SS	HSC	39.52	0.0035	100	0	0	300000.0	7591.09	1
	DDM	19.81	0.0019	100	17	4	13636.4	688.36	2
	ADWIN	35.68	0.0021	100	155	21	1694.9	47.50	4
	EDDM	32.26	0.0772	100	146	128	1090.9	33.82	5
	CUSUM	574.50	0.0153	100	0	0	300000.0	522.19	3
SEA-SS	PHT	827.88	0.0873	99	153	101	1176.5	1.41	6
	HSC	38.58	0.0042	100	0	0	300000.0	7776.05	1
	DDM	19.76	0.0023	100	19	0	15000.0	759.11	2
	ADWIN	36.00	0.0000	100	141	2	2083.3	57.87	5
	EDDM	19.49	0.0059	100	112	85	1515.2	77.74	4
SINE1-SG	CUSUM	541.54	0.0102	100	0	0	300000.0	559.98	3
	PHT	757.91	0.0796	100	158	108	1123.6	1.48	6
	HSC	161.06	0.0119	100	14	0	20000.0	124.18	2
	DDM	152.34	0.0139	100	52	4	5263.2	34.55	3
	ADWIN	165.92	0.0182	100	572	21	505.1	3.04	5
SEA-SG	EDDM	125.36	0.0193	100	204	128	900.9	7.19	4
	CUSUM	854.36	0.0158	100	0	0	300000.0	351.14	1
	PHT	1129.97	0.0778	97	130	101	1293.1	1.11	6
	HSC	159.47	0.0130	100	7	0	37500.0	235.15	2
	DDM	151.68	0.0122	100	60	0	4918.0	32.42	3
SINE1-WS	ADWIN	166.24	0.0164	100	536	2	556.6	3.35	5
	EDDM	120.56	0.0233	100	204	85	1034.5	8.58	4
	CUSUM	825.28	0.0115	100	0	0	300000.0	363.51	1
	PHT	1071.43	0.0778	100	154	108	1140.7	1.06	6
	HSC	84.02	0.0115	100	0	0	300000.0	3570.58	1
SEA-WS	DDM	52.44	0.0078	100	49	4	5555.6	105.94	2
	ADWIN	64.12	0.0173	99	143	21	1818.2	28.07	3
	EDDM	51.88	0.0189	100	146	128	1090.9	21.03	4
	CUSUM	1207.00	0.0477	2	0	0	300000.0	4.97	5
	PHT	1035.41	0.1257	17	9	101	2702.7	0.44	6
SINE1-WG	HSC	110.45	0.0180	100	0	0	300000.0	2716.16	1
	DDM	72.79	0.0148	100	36	0	8108.1	111.39	2
	ADWIN	82.08	0.0165	100	154	2	1910.8	23.28	5
	EDDM	57.58	0.0183	100	116	85	1485.1	25.79	4
	CUSUM	1402.87	0.0223	15	0	0	300000.0	32.08	3
SEA-WG	PHT	1239.30	0.0977	23	14	108	2439.0	0.45	6
	HSC	219.46	0.0271	100	18	0	15789.5	71.95	1
	DDM	186.35	0.0248	100	147	4	1973.7	10.59	2
	ADWIN	234.46	0.0497	99	179	21	1492.5	6.30	3
	EDDM	148.80	0.0330	100	215	128	872.1	5.86	5
SINE1-WG	CUSUM	1443.00	0.0132	3	0	0	300000.0	6.24	4
	PHT	1095.38	0.0737	8	8	101	2727.3	0.20	6
	HSC	362.79	0.0275	100	0	0	300000.0	826.92	1
	DDM	259.03	0.0223	100	47	0	6250.0	24.13	2
	ADWIN	291.36	0.0320	100	252	2	1176.5	4.04	5
SEA-WG	EDDM	192.90	0.0363	100	139	85	1333.3	6.91	3
	CUSUM	1462.00	0.0128	2	0	0	300000.0	4.10	4
	PHT	1248.18	0.0912	11	2	108	2702.7	0.24	6



En azul las verdaderas detecciones (TD) y en verde las falsas alarmas, es decir, aquellas detecciones antes de que se produzca el CD ( $FA_0$ ) o después de la genuina detección (FA)

Delay indica el retraso medio en detectar el CD desde que se produce (*Sudden change*) o comienza (*Gradual change*) en 1500. CV indica el coeficiente de variación

	Delay	CV	TD	FA+FA <sub>0</sub>	MTR	overall	rank
	reactivity	robustness	effectiveness	reliability			
<b>HSC</b>	0.96104	0.98155	1.00000	0.99178	2864.01	1.250	
<b>DDM</b>	0.98489	1.00000	1.00000	0.90662	220.81	2.250	
<b>ADWIN</b>	0.97013	0.94253	0.99745	0.53120	21.68	4.375	
<b>EDDM</b>	1.00000	0.85408	1.00000	0.55017	23.36	4.125	
<b>CUSUM</b>	0.30925	0.94594	0.51786	1.00000	229.78	3.000	
<b>PHT</b>	0.30058	0.32522	0.55995	0.69140	0.80	6.000	

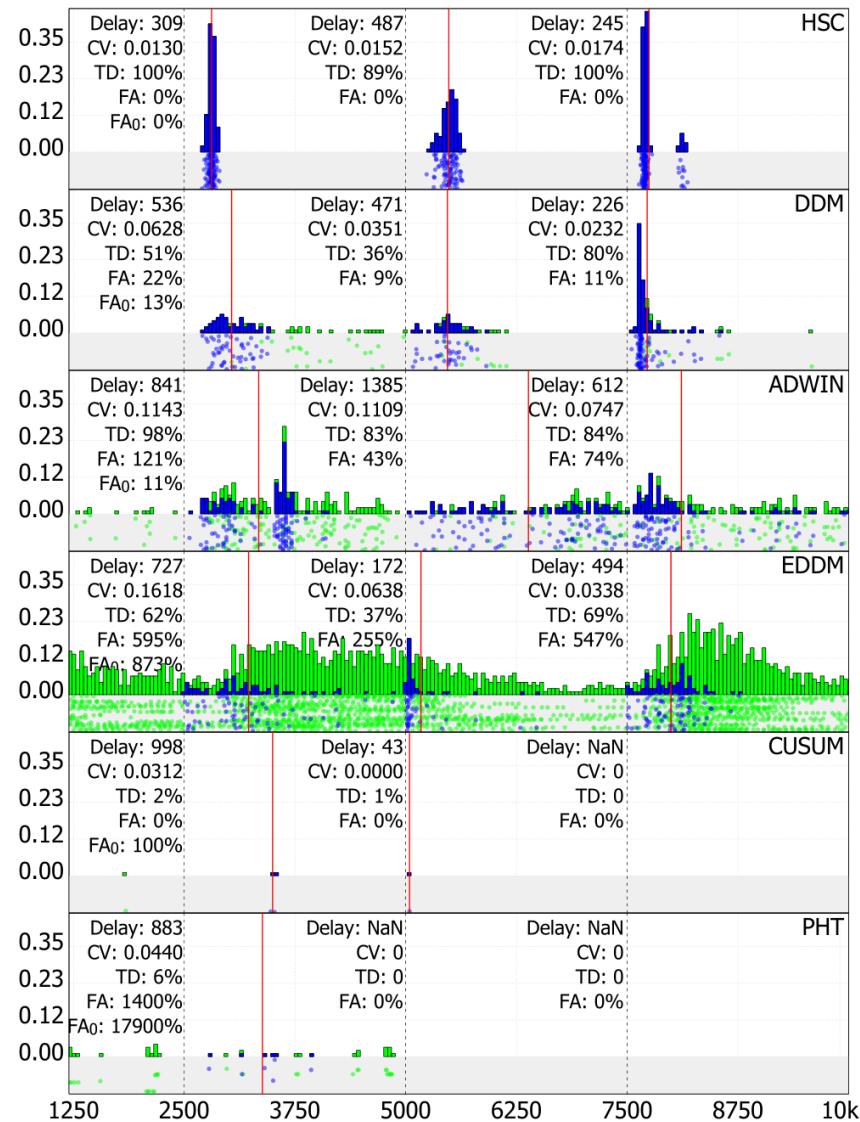
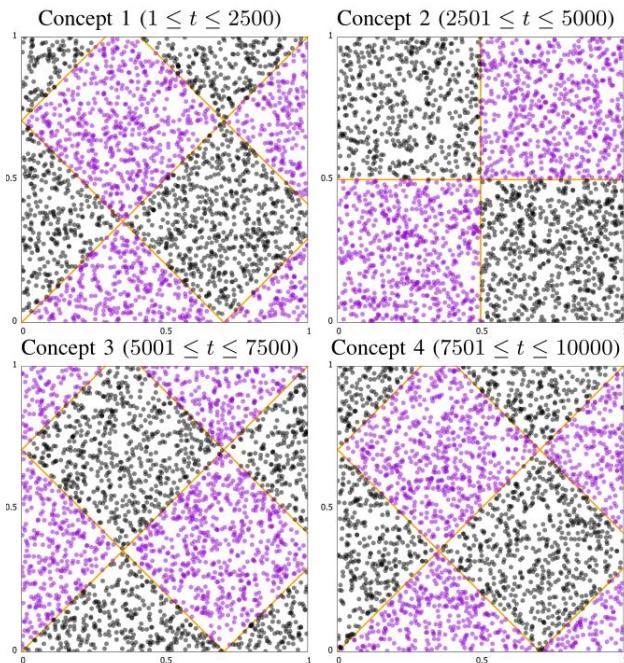
# Algoritmo HSC de Detección de CD

$$s(\mathbf{x}, \theta) = \left\lceil \frac{x_1 \cdot \cos \theta - x_2 \cdot \sin \theta}{0.5} \right\rceil + \left\lfloor \frac{x_1 \cdot \cos \theta + x_2 \cdot \sin \theta}{0.5} \right\rfloor$$

$$y(\mathbf{x}, \theta) = 2 \cdot (|s(\mathbf{x}, \theta)| \% 2) - 1 \quad (17)$$

$$y_{CB}(\theta) = [\mathbf{x}, y(\mathbf{x}, \theta)], \mathbf{x} \sim U([0, 1])^2$$

$$\text{CB-VS}(t) = \begin{cases} y_{CB}\left(\frac{3\pi}{4}\right), & 1 \leq t \leq 2500 \\ y_{CB}\left(\frac{\pi}{2}\right), & 2501 \leq t \leq 5000 \\ y_{CB}\left(\frac{5\pi}{4}\right), & 5001 \leq t \leq 7500 \\ y_{CB}\left(\frac{7\pi}{4}\right), & 7501 \leq t \leq 10000 \end{cases} \quad (18)$$



# Algoritmo HSC de Detección de CD

Data stream				HSC					DDM	
name	class labels	features	samples	bins	time/ samples	time/ (samples·bins)	time	Hz	time	Hz
<b>SINE1-SS</b>	2	2	3 000	80	0.02466667	0.00030833	74	40 541	988	3 036
<b>SINE1-SG</b>	2	2	3 000	80	0.02466667	0.00030833	74	40 541	1 008	2 976
<b>SINE1-WS</b>	2	2	3 000	80	0.02466667	0.00030833	74	40 541	1 077	2 786
<b>SINE1-WG</b>	2	2	3 000	80	0.02466667	0.00030833	74	40 541	973	3 083
<b>SEA-SS</b>	2	3	3 000	120	0.02933333	0.00024444	88	34 091	1 040	2 885
<b>SEA-SG</b>	2	3	3 000	120	0.02933333	0.00024444	88	34 091	967	3 102
<b>SEA-WS</b>	2	3	3 000	120	0.02933333	0.00024444	88	34 091	987	3 040
<b>SEA-WG</b>	2	3	3 000	120	0.02933333	0.00024444	88	34 091	1 155	2 597
<b>CB-VS</b>	2	2	10 000	80	0.02520000	0.00031500	252	39 683	15 146	660
<b>Electricity</b>	2	7	45 312	310	0.05863789	0.00018915	2 657	17 054	596 081	76
<b>Airlines</b>	2	7	539 378	250	0.05240295	0.00020961	28 265	19 083	53 244 877	10

En DDM se usa como clasificador el *bagging* incremental de Oza y Russell (2001) con 10 árboles.

HSC es 10 veces más rápido que DDM en SINE1 y SEA, 60 veces más en CB, 200 veces más en Electricity y 1800 veces más en Airlines. Es capaz de procesar hasta 40.000 muestras por segundo.



# ¿Tiene Sentido la Clasificación en Flujo de Datos?

- En aprendizaje supervisado, no podemos reemplazar al oráculo/experto, lo necesitamos siempre porque cada muestra que viene del flujo de datos necesita ser etiquetada
- Si el flujo de datos es una fuente potencialmente infinita de datos, necesitamos al oráculo para siempre
- Por lo tanto, cuando llega un nuevo dato, ¿por qué pedir al modelo que conozca la clase cuando podemos preguntar al oráculo, ya que está permanentemente disponible?
- Si lo que queremos es predecir la clase de los próximos datos, ¿por qué no esperar a recibirlos?
- Etiquetar datos es costoso, pero al mismo tiempo se espera una tasa de llegada de datos muy elevada. ¿Cómo casan estos dos requerimientos contradictorios?

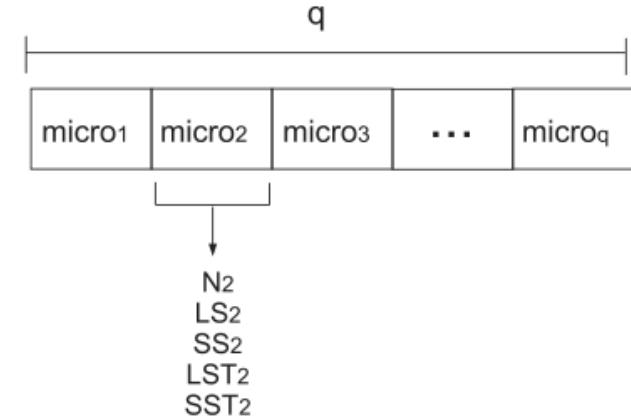
# Razones para Usar Aprendizaje Supervisado en Flujo de Datos

- El modelo **evoluciona continuamente** a medida que el experto está disponible y puede usarse para predecir casos adicionales cuando no lo esté
  - Esto no es flujo de datos «real» pues el algoritmo dejaría de aprender cuando dejen de llegar datos etiquetados
- Es útil tratar **big data** estático como aprendizaje incremental ya que se procesa de manera eficiente una gran cantidad de instancias
  - Bueno, sin duda es atractivo y prometedor, pero sigue sin ser flujo de datos «real» pues no se considera CD, respuesta inmediata, etc.
- El modelo solo se usa para **monitorizar** el sistema, no para predecir la clase. De hecho, el modelo se adapta a los datos para explicar lo que está sucediendo en todo momento
  - Sí, esto es muy útil, pero ¿por qué la mayoría de la investigación actual en este campo se centra en la precisión e ignora la interpretabilidad?

1. *Concept drift* en flujo de datos
2. Clasificación en flujo de datos
3. ***Clustering*** en flujo de datos
4. Patrones frecuentes en flujo de datos

# Clustering en Flujo de Datos

- CluStream [Aggarwal et al., 2003] divide el proceso en dos componentes:
  1. *Online*: mantiene modelos resumidos de los datos basados en estadísticos (*microclusters*) similares a los CF (número de datos así como suma de valores y sus cuadrados) usados en BIRCH pero añadiendo también información sobre el tiempo (suma lineal y cuadrática del tiempo en que se recibe cada objeto). Cuando llega un dato, se puede agregar a un *microcluster* (si cae dentro de su frontera) o formar uno nuevo, en cuyo caso, para hacer espacio, se elimina el más antiguo o bien se funden dos en uno

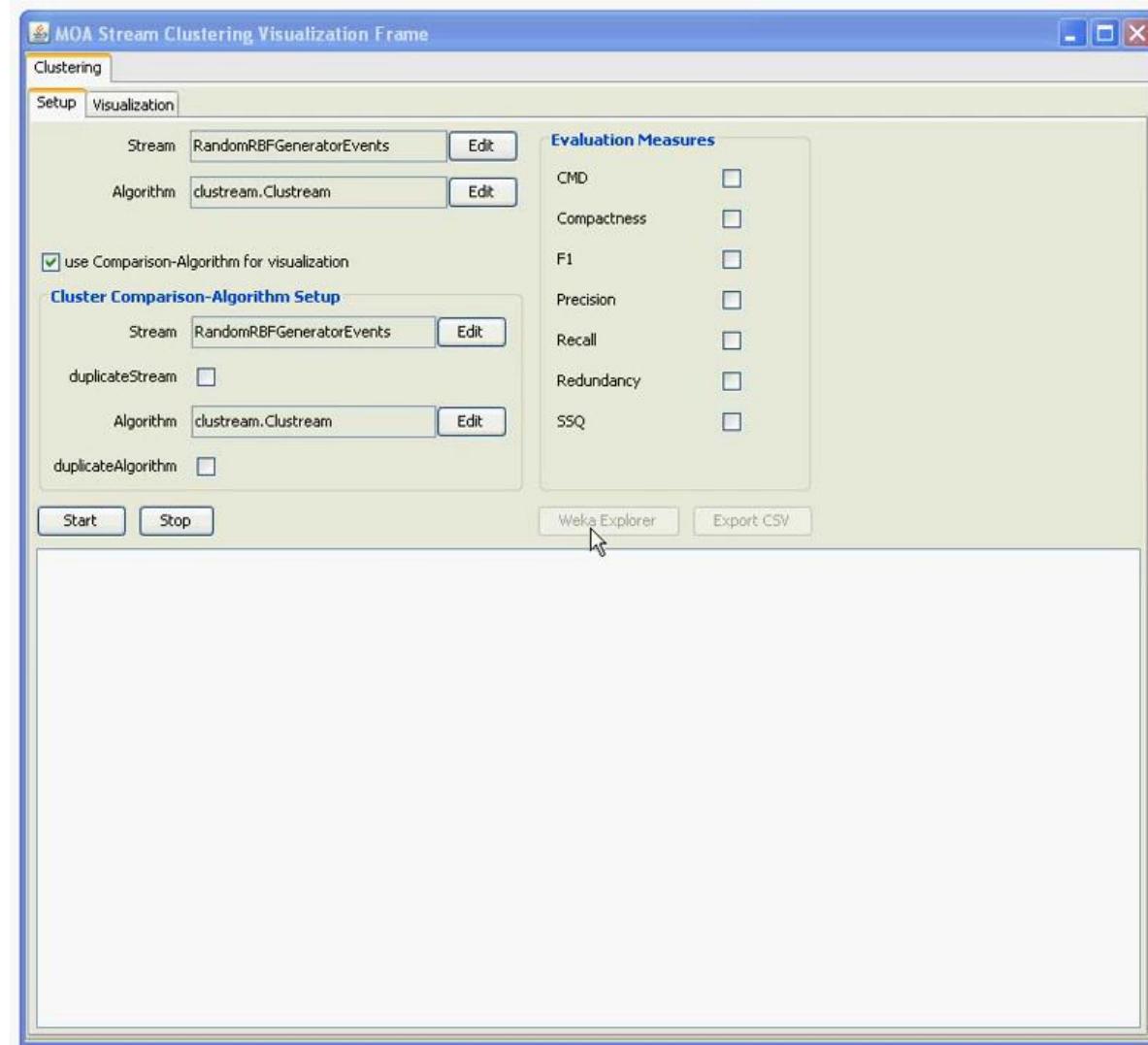


# *Clustering* en Flujo de Datos

- 2. *Offline*: que aplica agrupamiento sobre estos resúmenes de los datos según preferencias del usuario como horizonte de tiempo y número de *clusters*
- CluStream es más eficiente que el algoritmo Stream y ofrece un marco basado en ventana de tiempo piramidal que permite abordar verdaderos flujos de datos incluso con *concept drift*, a pesar de que no detecta los desvíos y no puede trabajar con tasas de consultas muy elevadas
- Otra versión de los mismos autores (HPStream) aplica proyección para datos de alta dimensión

# Clustering en Flujo de Datos

Ejemplo de CluStream  
en MOA



# Clustering en Flujo de Datos

- DenStream [Cao et al., 2006] también usa vectores de características como BIRCH. En la fase online, mantiene dos estructuras ( $p$ -microclusters con posibles clusters y  $o$ -microclusters para detección de outliers)
- Se les asocia un peso que indica su importancia basada en el tiempo. Se calcula el radio ponderado según esos pesos
- Cuando llega un nuevo objeto, se intenta insertar en el  $p$ -microcluster más cercano si al hacerlo su radio está dentro de la frontera predefinida. Si no, se intenta insertar en el  $o$ -microcluster más cercano o bien se crea uno nuevo
- Cuando el peso del  $o$ -microcluster crece, se convierte en un  $p$ -microcluster

$$w = \sum_{j \in p\text{-micro-cluster}} f(T - t^j),$$

$$f(t) = 2^{-\lambda t}$$

$$WLS = \sum_{j \in p\text{-micro-cluster}} f(T - t^j) \mathbf{x}^j$$

$$WSS = \sum_{j \in p\text{-micro-cluster}} f(T - t^j) \mathbf{x}^{j^2}$$

$$r = \sqrt{\sum_{j=1}^n \left( \frac{WSS^j}{w} - \left( \frac{WLS^j}{w} \right)^2 \right)}$$

# Clustering en Flujo de Datos

- ClusTree [Kranen et al., 2011] también usa vectores de características de *clusters* ponderados que se almacenan en un árbol jerárquico
- Permite que pueda ser interrumpido el proceso de inserción de nuevos objetos en cualquier momento. En tal caso, los nuevos objetos se almacenan en un árbol que actúa de *buffer* y que puede incorporarse al modelo cuando se permita
- No hace asunciones previas sobre el tamaño del modelo, se ajusta automáticamente
- Se adapta también a diferentes tasas de llegada de objetos (cuando es elevada, se agregan objetos similares para una inserción más rápida; cuando es baja, se aprovecha ese tiempo para mejorar la calidad del agrupamiento)

1. *Concept drift* en flujo de datos
2. Clasificación en flujo de datos
3. *Clustering* en flujo de datos
4. **Patrones frecuentes en flujo de datos**

# Patrones Frecuentes y Reglas de Asociación



Frequent Itemsets	Association Rule	Support Count of Antecedent	Support Count of Rule	Support of Rule	Confidence of Rule
{Beer, Bread, Milk}	{Bread, Milk} → Beer	3	2	2/5 = 0.4	2/3 = 0.67
{Beer, Bread, Milk}	{Beer, Milk} → Bread	2	2	2/5 = 0.4	2/2 = 1.00
{Beer, Bread, Milk}	{Beer, Bread} → Milk	3	2	2/5 = 0.4	2/3 = 0.67
{Bread, Milk}	Bread → Milk	4	3	3/5 = 0.6	3/4 = 0.75
{Bread, Milk}	Milk → Bread	3	3	3/5 = 0.6	3/3 = 1.00
{Beer, Diaper}	Diaper → Beer	3	3	3/5 = 0.6	3/3 = 1.00
{Bread, Coke, Milk}	{Coke, Milk} → Bread	1	1	1/5 = 0.2	1/1 = 1.00

# Patrones Frecuentes en Flujo de Datos

- *Heavy hitters*, es decir, encontrar elementos que superan un umbral de frecuencia, se puede considerar como el enfoque más básico de *frequent items* (sólo un objeto)
- Existen muchos algoritmos de este tipo, destacando el algoritmo SpaceSaving [Metwally et al. 2005] que usa distintas técnicas de muestreo, bocetos...
- El caso de *frequent itemsets* en general se suele resolver con ventana deslizante de granularidad de tiempo fija (como estWin [Chang y Lee, 2005] o Moment [Chi et al., 2004]) o múltiple (como FP-tree [Giannella et al., 2004])
- La mayoría de enfoques son para variables nominales

# Patrones Frecuentes en Flujo de Datos

Reference	Category	Type	Data	Approach	Rules	Experiments
(Karp et al., 2003)	heavy hitter	counting	categorical	—	no	—
(Chang and Lee, 2003)	frequent pattern mining	counting	categorical	decay factor	no	synthetic
(Chi et al., 2004)	closed itemsets mining	tree	categorical	sliding	no	mixed
(Chang and Lee, 2005a)	frequent pattern mining	tree	categorical	sliding	no	mixed
(Chang and Lee, 2005b)	frequent pattern mining	counting	categorical	sliding	no	synthetic
(Cheng et al., 2006)	frequent pattern mining	tree	categorical	sliding	no	synthetic
(Wong and Fu, 2006)	top-k frequent itemsets	counting	categorical	sliding	no	synthetic
(Marascu and Masseglia, 2006)	frequent pattern mining	tree	categorical	sliding	no	mixed
(Raïssi and Poncelet, 2007)	sequential pattern mining	counting	categorical	sliding	no	mixed
(Chen and Li, 2007)	closed itemsets mining	tree	categorical	sliding	no	synthetic
(Yang et al., 2007)	frequent pattern mining	tree	categorical	sliding	no	synthetic
(Cormode et al., 2008)	heavy hitter	counting	categorical	—	no	mixed
(Cheng et al., 2008)	closed itemsets mining	tree	categorical	sliding	no	synthetic
(Fan et al., 2009)	ratio rules	counting	quantitative*	sliding	yes	mixed
(Yen et al., 2009)	frequent pattern mining	hashing	categorical	sliding	no	synthetic
(Leung and Boyu, 2009)	frequent pattern mining	tree	uncertain	sliding	no	synthetic
(Tu et al., 2010)	closed itemsets mining	tree	categorical	sliding	no	mixed
(Tan et al., 2010)	frequent pattern mining	tree	categorical	sliding	yes	mixed
(Wang and Chen, 2011)	frequent pattern mining	hashing	categorical	sliding	no	mixed
(Chen et al., 2010)	fuzzy association rules	tree	fuzzy	sliding	yes	mixed
(Memar et al., 2011)	frequent pattern mining	queue	categorical	sliding	no	synthetic
(Bifet et al., 2011)	frequent closed graphs mining	graph	categorical	sliding	no	mixed
(Huang et al., 2012)	rare itemsets mining	tree	categorical	sliding	no	mixed
(Manku and Motwani, 2012)	frequent pattern mining	counting	categorical	landmark	no	mixed
(Patnaik et al., 2013)	frequent pattern mining	counting	categorical	sliding	no	mixed
(Zhang et al., 2013)	frequent pattern mining	tree	categorical	sliding	no	real-world
(HewaNadungodage et al., 2013)	frequent pattern mining	hyper-structure	uncertain	sliding	no	mixed
(Akbarinia and Masseglia, 2013)	probabilistic pattern mining	counting	quantitative	sliding	no	mixed
(Chen, 2014)	top-k frequent itemsets	tree	categorical	sliding	no	synthetic

# Patrones Frecuentes en Flujo de Datos

- Category

- heavy hitter: find the singleton items with a support greater than the given threshold
- top-k frequent itemsets: find the top k frequent items in a stream
- frequent pattern mining: find all the frequent itemsets
- closed itemsets discoverer: find those itemsets that have no frequent supersets with the same frequency, thus eliminating redundancy
- rare itemsets mining: itemsets that do not occur frequently
- ratio rules: find the quantitative knowledge between distinct itemsets inside a rule
- frequent closed graphs mining: find those graphs that have no frequent supersets with the same frequency
- probabilistic pattern mining: find those itemsets with a probability greater than a minimum threshold
- association rules: find all the frequent (quantitative) rules

# Patrones Frecuentes en Flujo de Datos

- Type:

- Counting based: an iterative counting algorithm
- Tree based: a tree structure is deployed for pattern identification and extraction
- Queue based: a queue structure is used for pattern discovery
- Hashing based: hash tables are exploited for frequent itemset discovery
- Graph based: a graph structure is utilized for pattern discovery
- Hyper-structure based: a set of dynamic structures are combined for pattern identification

- Approach:

- the method does not distinguish recent items from older ones, learning from a static batch of instances (landmark window)
- the algorithm gives more importance to recent transactions, learning from a dynamic batch of instances (sliding window or decay factor)

# Reglas de Asociación en Flujo de Datos

- Las **reglas de asociación** son mucho más descriptivas y útiles que los patrones frecuentes ya que muestran relaciones de causalidad. Sin embargo, aprender reglas de asociación en flujo de datos es mucho más complicado que en datos estáticos
- La mayoría de propuestas aplican el proceso en dos pasos para minería de reglas donde primero se obtienen patrones frecuentes online ( $\{Beer, Bread, Milk\}$ ) y luego se extraen las reglas offline ( $\{Beer, Milk\} \rightarrow Bread$ ). Esto implica poder acceder a todos los datos, lo cual no es realmente flujo de datos
- Las pocas soluciones que mantienen reglas de asociación online son muy ineficientes e impracticables en problemas con una tasa de llegada de datos alta
- Además, lo habitual es no tratar concept drift y manejar variables categóricas

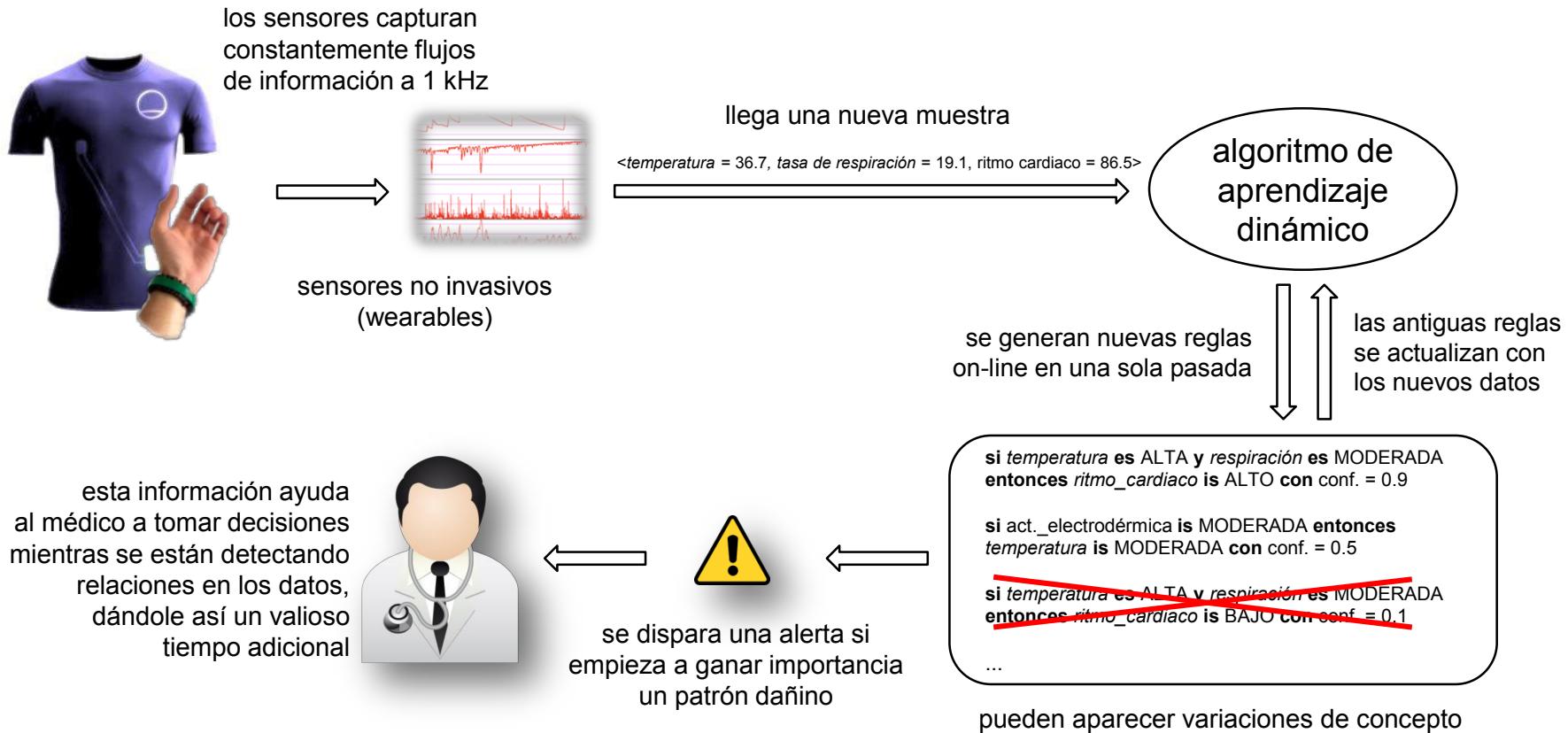
# *Association Stream Mining*

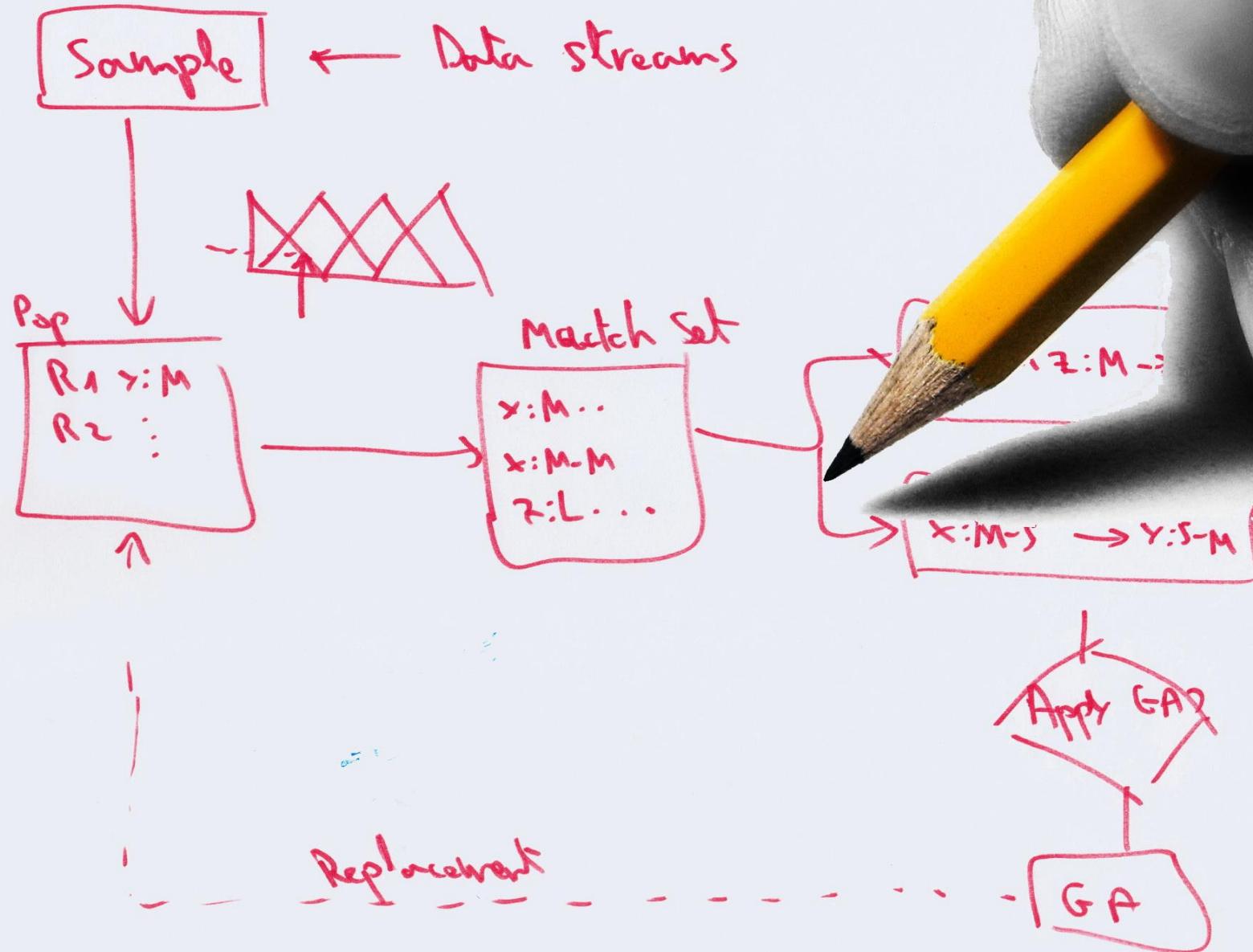
El objetivo es extraer asociaciones legibles entre variables en flujos de información a alta velocidad y con cambios a lo largo del tiempo así como adaptarse a cambios de concepto de una manera puramente en línea

- Las principales características de este campo son las siguientes:
  - Flujo continuo de datos que debe procesarse en un solo paso
  - Restricciones en el uso de la memoria
  - El concepto a aprender puede cambiar con el tiempo
  - No se puede suponer una estructura subyacente en los datos
  - No se puede suponer una distribución de datos fija
  - Se desea un buen grado de interpretabilidad

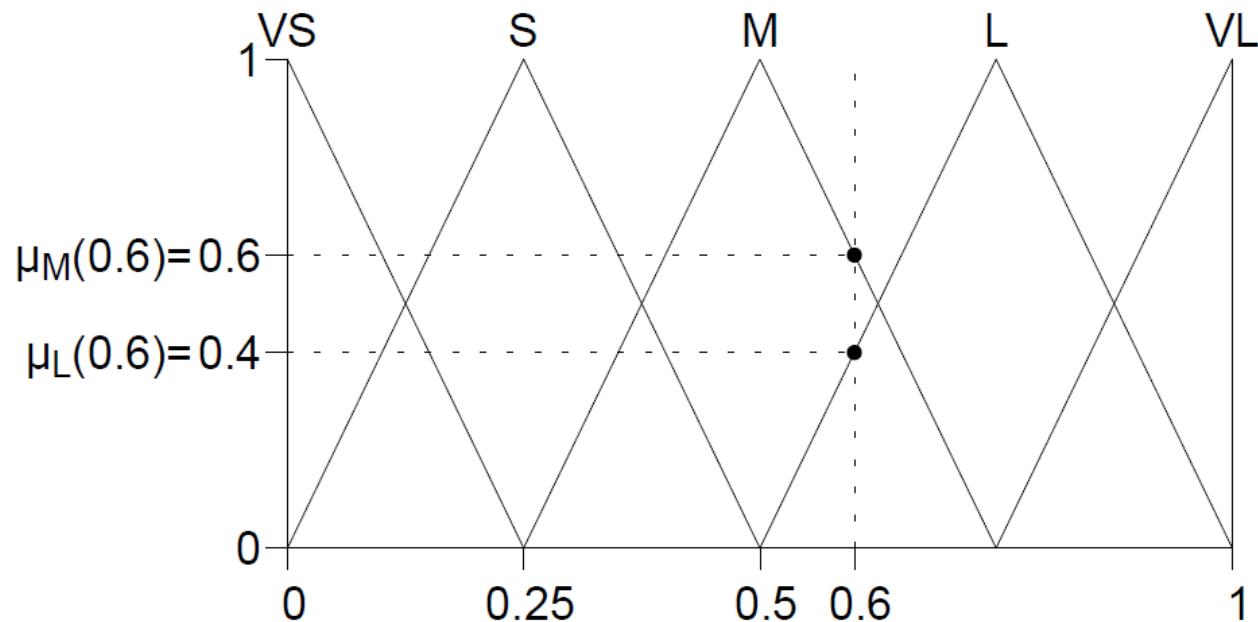
A. Sancho-Asensio, A. Orriols-Puig, J. Casillas,  
Evolving Association Streams,  
Information Sciences 334-335 (2016) 250-272  
DOI: [10.1016/j.ins.2015.11.043](https://doi.org/10.1016/j.ins.2015.11.043)

# Ilustración de una Potencial Aplicación de Association Stream Mining





# Regla de Asociación Difusa



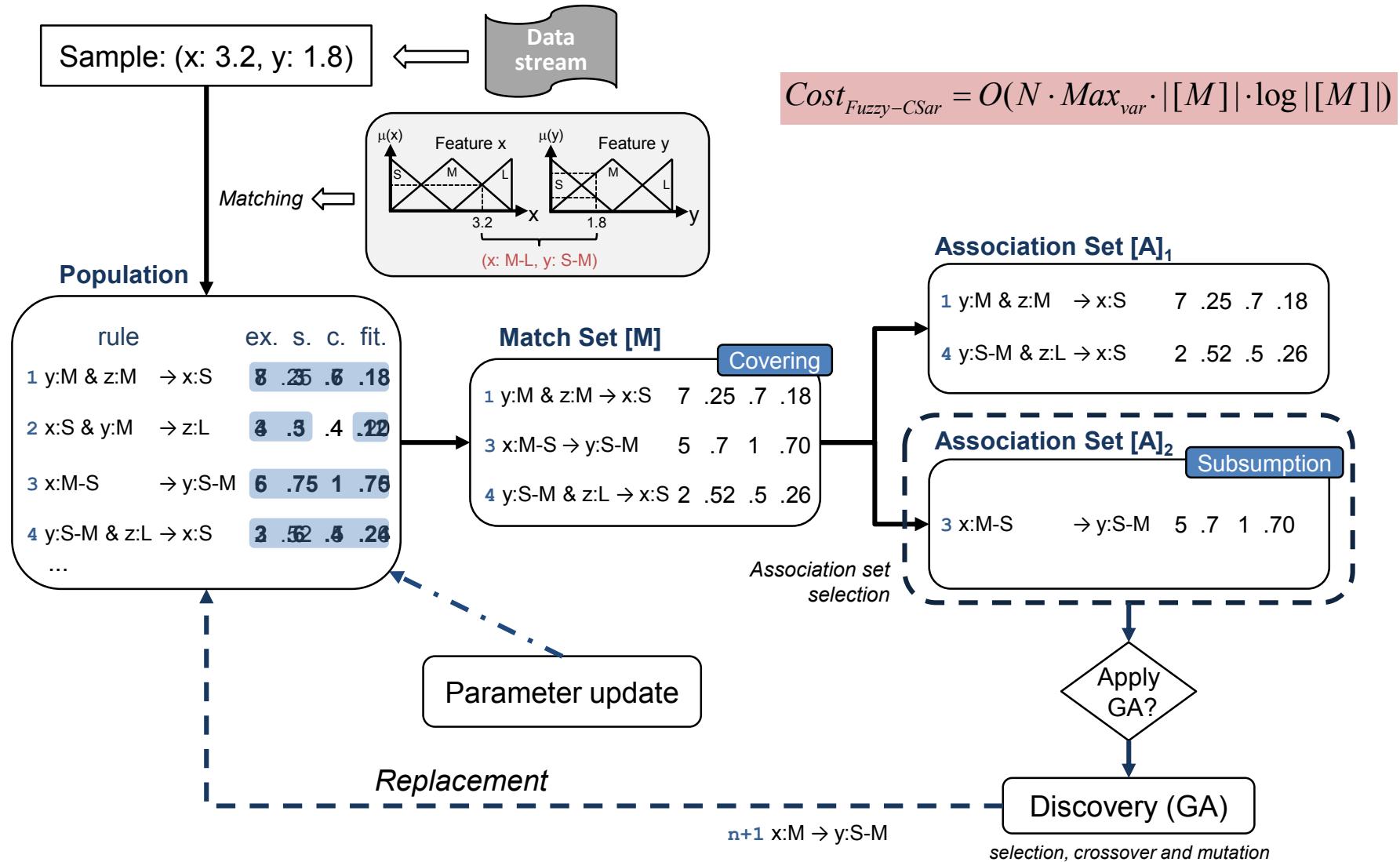
**IF**  $x_1$  is  $A_1$  and ... and  $x_n$  is  $A_n$  **THEN**  $y_1$  is  $C_1$  and ... and  $y_m$  is  $C_m$

$$\text{supp}(X \cup Y) = \prod \mu_{A_i}(x_i) \prod \mu_{C_j}(y_j)$$

$$\text{conf}(X \Rightarrow Y) = \prod \mu_{A_i}(x_i) \cdot \max \left\{ 1 - \prod \mu_{A_i}(x_i), \prod \mu_{C_j}(y_j) \right\}$$

# Fuzzy-Csar

A. Sancho-Asensio, A. Orriols-Puig, J. Casillas, Evolving Association Streams, Information Sciences 334-335 (2016) 250-272



# Fuzzy-CSar

- Association set candidates:
  - The purpose of generating niches is to group rules that express similar associations to establish a competition among them. This strategy lets the best individuals take over their niche, hence evolving highly
  - Fuzzy-CSar considers that two rules are similar if they have exactly the same variables in their antecedent, regardless of their corresponding linguistic terms. The underlying principle is that rules with the same antecedent variables may express similar knowledge
  - Afterwards, the final set  $[A]$  is selected following a roulette-wheel strategy, where each  $[A]_i$  has a probability of being selected proportional to its accumulated confidence:

$$p_{sel}^k \leftarrow \frac{\sum_{i \in [A]_k} w_i \cdot conf_i}{\sum_{j \in [M]} w_j \cdot conf_j}, \quad w_i = \begin{cases} 1 & \text{if } exp_i > \theta_{exp} \\ 0.1 & \text{otherwise} \end{cases}$$

- Association Set Subsumption

- It is applied to reduce the number of different rules that express the same knowledge
- Each rule in [A] is checked for subsumption with each other rule of the set. A rule  $r_i$  is a candidate subsumer of  $r_j$  if two conditions are satisfied:

1.  $r_i$  has a *similar* confidence than  $r_j$  and it is experienced *enough*
2.  $r_i$  is more general than  $r_j$   
(i.e.,  $r_i$  contains at least all the variables and terms of  $r_j$ )

Ej:  $r_i: y:S\text{-}M \rightarrow x:M\text{-}L$

$r_j: z:L \& y:M \rightarrow x:M$

- Every time a rule  $r_i$  subsumes another  $r_j$ ,  $r_i$  increases its number of copies and the subsumed one  $r_j$  is deleted from population

# Fuzzy-CSar

- Parameter Update (on the whole population)

$$experience_{t+1} \leftarrow experience_t + 1$$

$$supp_{t+1} \leftarrow supp_t + \frac{\prod \mu_{A_i}(x_i) \prod \mu_{C_j}(y_j) - supp_t}{experience}$$

$$imp_{t+1} \leftarrow imp_t + \prod \mu_{A_i}(x_i) \cdot \max \{1 - \prod \mu_{A_i}(x_i), \prod \mu_{C_i}(y_i)\}$$

$$suppX_{t+1} \leftarrow suppX_t + \frac{\prod \mu_{A_i}(x_i) - suppX_t}{experience}$$

$$conf_{t+1} \leftarrow \frac{imp_{t+1}}{suppX_{t+1}}$$

$$fitness_{t+1} \leftarrow (supp_{t+1} \cdot conf_{t+1})^\nu$$

(other options for fitness, e.g. based on *lift* and *accuracy*, are also possible)

# Fuzzy-CSar

- Discovery component (steady-state, niche-based, genetic algorithm)
  - When?: discovery component is triggered when the average time from its last application upon the individuals in [A] exceeds a threshold
  - Selection: two rules are selected as parents from [A] using tournament selection on fitness
  - Crossover: applied with a specific probability; it exchanges variables and linguistic terms among these two parent rules
  - Mutation: applied with a specific probability; it chooses between different actions, as switching an antecedent variable with the consequent, adding or removing variables in the antecedent, or expanding/contracting/shifting linguistic terms in a variable

# Fuzzy-CSar

- Replacement mechanism

- The new offspring are introduced into the population, but first each individual is checked for subsumption
- If the population is full, exceeding individuals are deleted from  $[P]$  with probability directly proportional to their association set size estimation and inversely proportional to its fitness

$$p^k \leftarrow \frac{d^k}{\sum_{\forall j \in [P]} d^j}, \quad d^k \leftarrow \begin{cases} \frac{as \cdot num \cdot F_{[P]}}{F^k} & \text{if } experience^k > \theta_{del} \text{ and } F^k < \delta F_{[P]} \\ as \cdot num & \text{otherwise} \end{cases}$$

- Thus, the deletion algorithm balances the individual's allocation in the different  $[A]$  by pushing toward the deletion of rules belonging to large association sets. At the same time, it favors the search towards highly fit individuals, since the deletion probability of rules whose fitness is much smaller than the average fitness is increased

# Fuzzy-CSar

- Characteristics of Fuzzy-CSar:

- It deals with both **categorical and numeric** variables
- It evolves **association rules directly**, which explain causality relationships among variables (not as frequent itemsets)
- The association rules are **linguistically legible**
- **Without** the need of using **sliding window** or decay factor, it gives more importance to new associations on data, but only forget old associations if they lay in contradiction with the new ones
- It quickly **reacts to concept drifts**

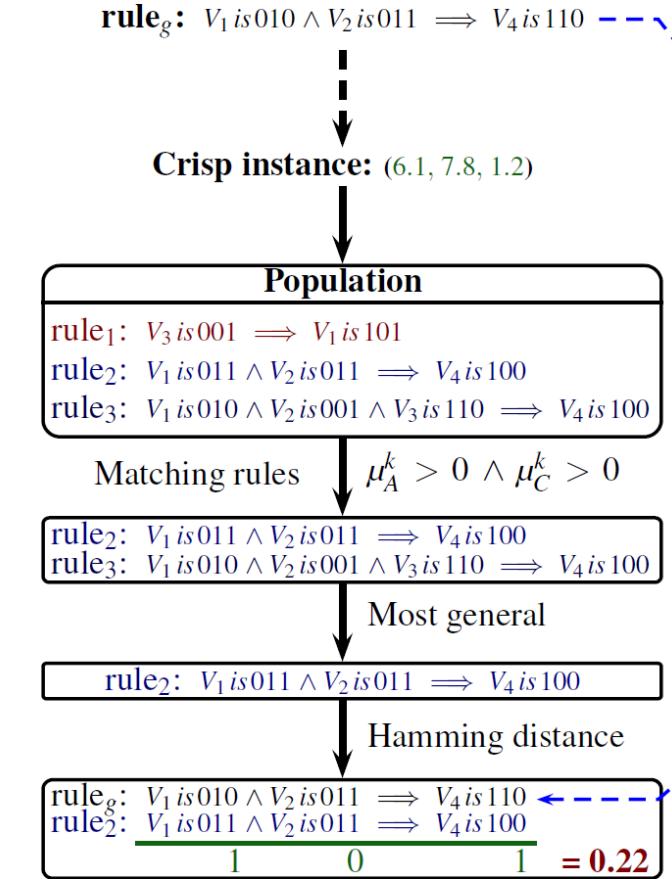


A green rectangular sign with a white border and a decorative pattern of small circles along the top and bottom edges. The word "Results" is written in large, bold, white, sans-serif capital letters. The sign is mounted on a vertical wooden post.

Results

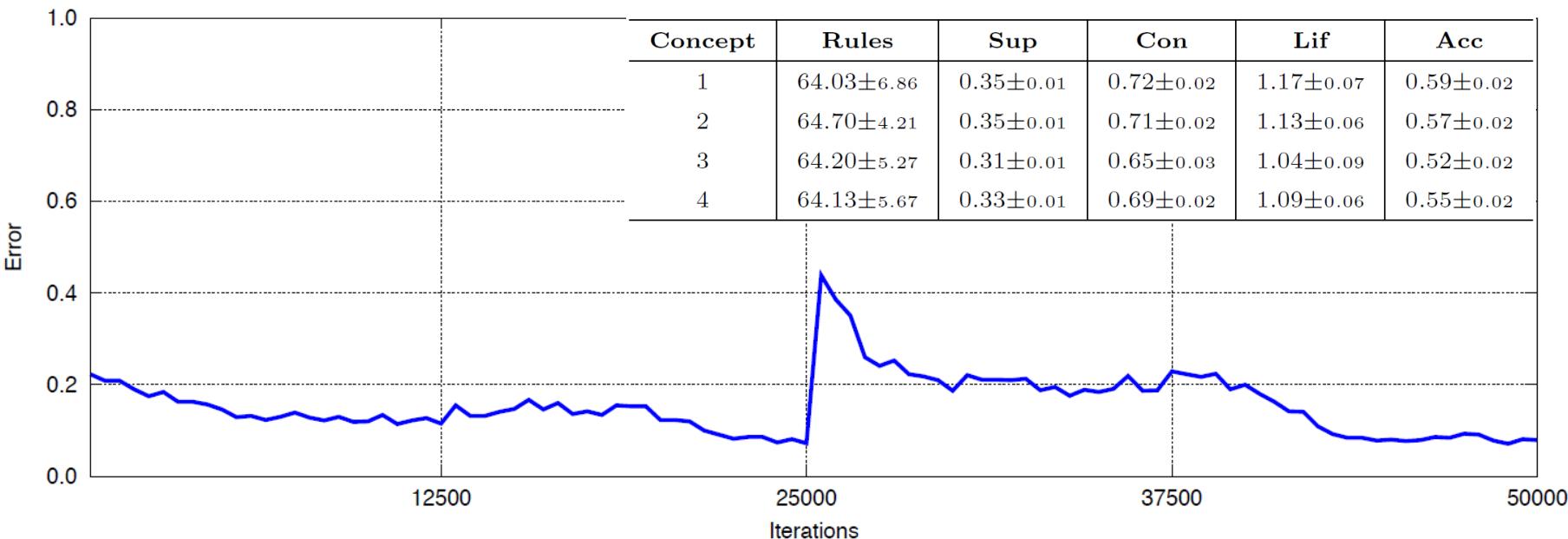
# Methodology in Synthetic Association Stream problems

- Difficult to evaluate quantitatively
- We had to define a testing environment
  - Given a set of fuzzy rules, it generates the crisp data
  - These data are accessed online, simulating a real data stream
  - Fuzzy-CSar has to discover the predefined rules out of data
- We used a holdout strategy
  - Separate train and test sets
- Procedure
  - First, we find the rule with maximum matching degree among the solutions provided by Fuzzy-CSar
  - Next, we compare this rule with the predefined one
- Comparisons by means of the accumulated Hamming distance
- Measures of interestingness and the number of rules per concept are also considered



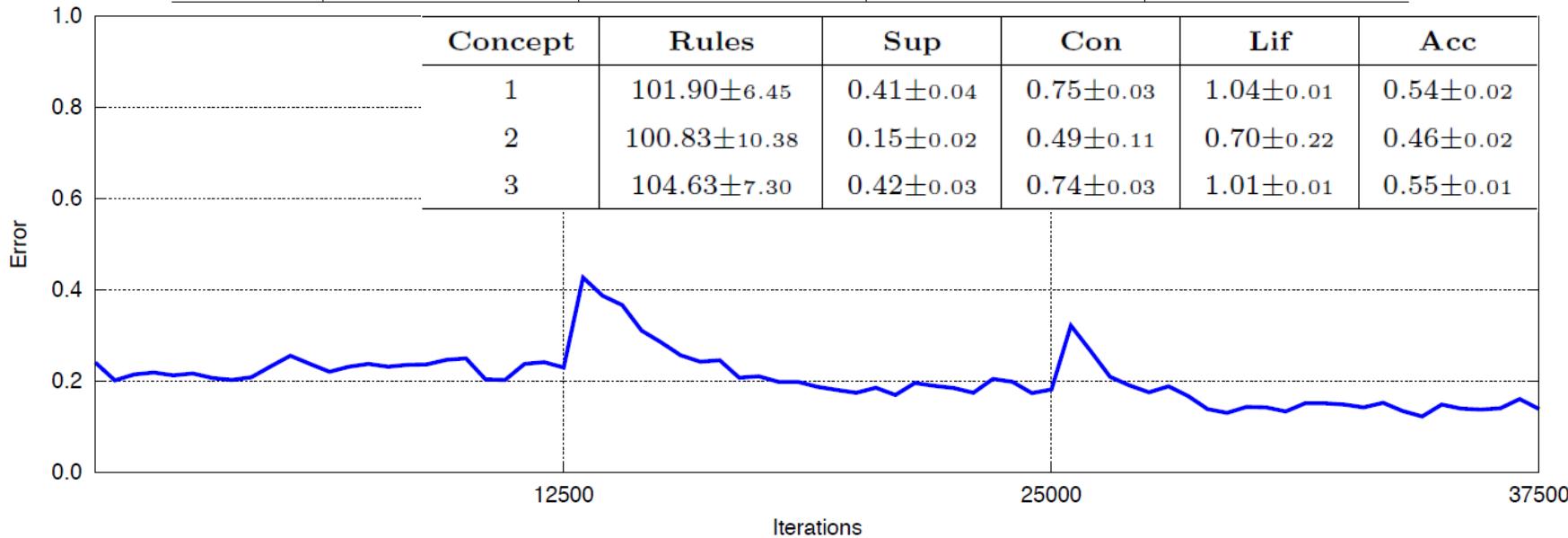
# Análisis de los Resultados en Entornos Sintéticos

Concept 1	$x_1$					$x_2$					Concept 3	$x_3$					$x_1$				
	VS	S	M	L	VL	VS	S	M	L	VL		VS	S	M	L	VL	VS	S	M	L	VL
$R_1$	X X X					X X X					$R_1$	X X					X X				
$R_2$	X X					X X					$R_2$						X X				
Concept 2	$x_2$					$x_1$					Concept 4	$x_1$					$x_2$				
	VS	S	M	L	VL	VS	S	M	L	VL		VS	S	M	L	VL	VS	S	M	L	VL
$R_1$	X X X					X X X					$R_1$	X X X					X X X				
$R_2$	X X					X X					$R_2$	X X					X X				



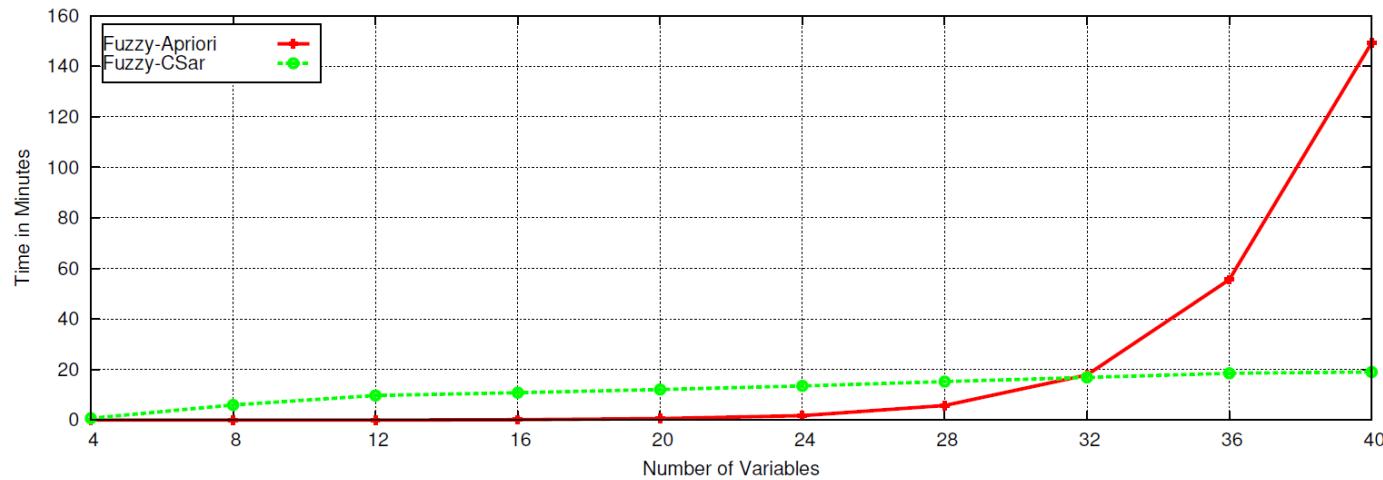
# Análisis de los Resultados en Entornos Sintéticos

Concept 1	$x_1$					$x_2$					$x_3$					
	VS	S	M	L	VL	VS	S	M	L	VL	VS	S	M	L	VL	
$R_1$	X	X	X			X	X	X								
$R_2$	X	X	X								X	X	X			
Concept 2	$x_1$					$x_2$					$x_3$					
	VS	S	M	L	VL	VS	S	M	L	VL	VS	S	M	L	VL	
$R_1$	X	X		X			X	X	X							
$R_2$	X	X		X							X	X	X			
Concept 3	$x_1$					$x_4$					$x_2$					
	VS	S	M	L	VL	VS	S	M	L	VL	VS	S	M	L	VL	
$R_1$	X	X					X				X	X				
$R_2$	X	X					X					X	X			

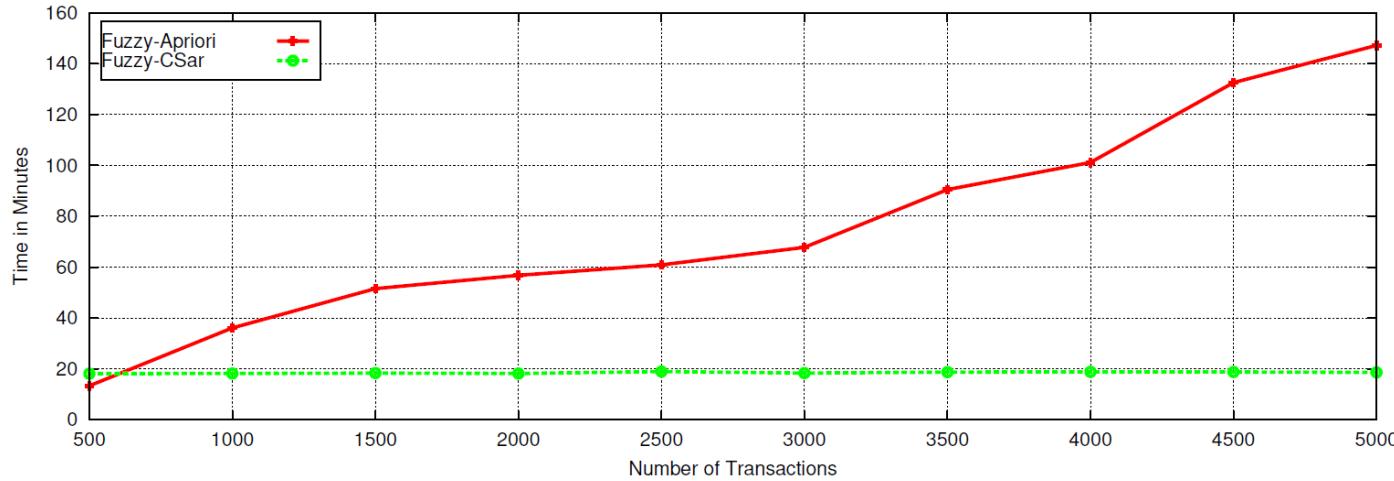


# Análisis de Escalabilidad

Runtime vs.  
no. variables



Runtime vs.  
no. transactions



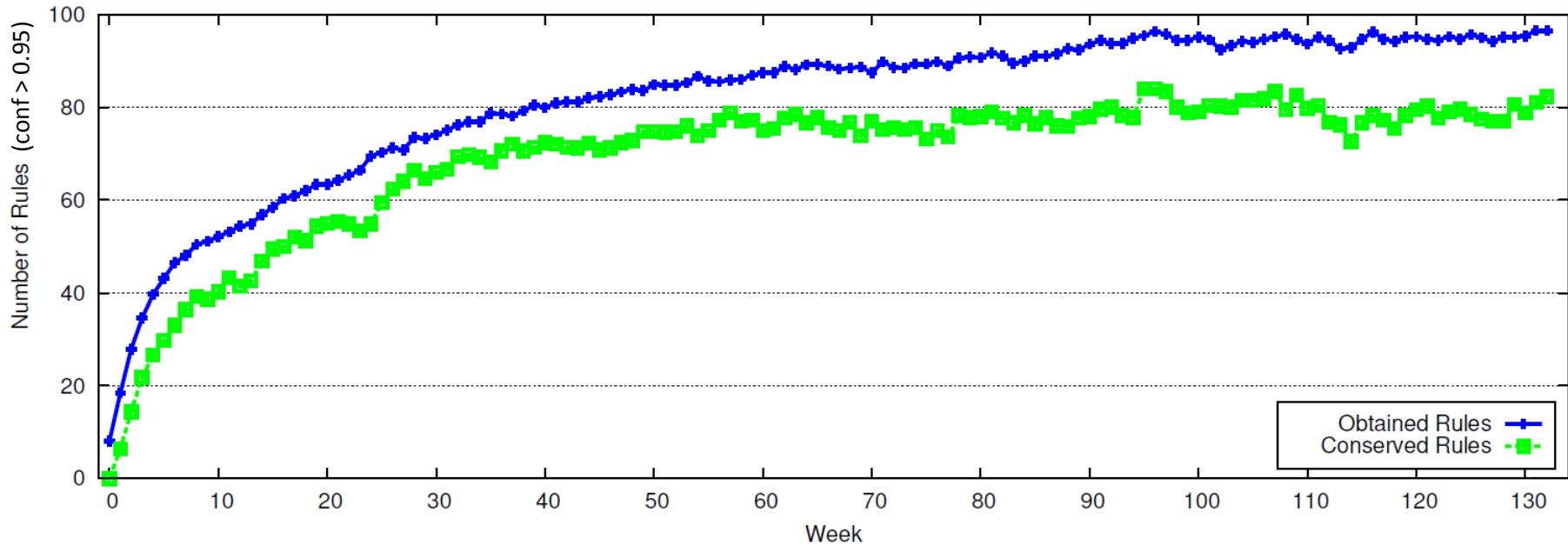
wav	2 132.300 $\pm$ 41.075	4.489 $\pm$ 4.404	0.415 $\pm$ 0.054	0.891 $\pm$ 0.005	545.313 $\pm$ 8.845
	1 262 758	3.372 $\pm$ 0.457	0.066 $\pm$ 0.000	0.761 $\pm$ 0.002	9 417.744 $\pm$ 621.681



# Análisis de Resultados en un Problema Real

- Problema del mercado de electricidad de Nueva Gales del Sur en Australia
- En este mercado, los precios se ven afectados por la demanda y la oferta del mercado; se establecen cada cinco minutos
- No estacionario e impredeciblemente cambiante
- 8 funciones continuas
- 45.312 ejemplos, cada uno muestrado cada 30 minutos
- Probado cada 336 ejemplos (una semana)
- Se muestra el número de reglas con una confianza mayor a 0.95 en la semana actual y esas reglas igual a la semana anterior

# Análisis de Resultados en un Problema Real



**IF NSWPrice is {XS,S,M} and NSWDemand is {S,M,L}**  
**THEN transfer is {S,M, L} [s: 0.6, c: 1.0]**

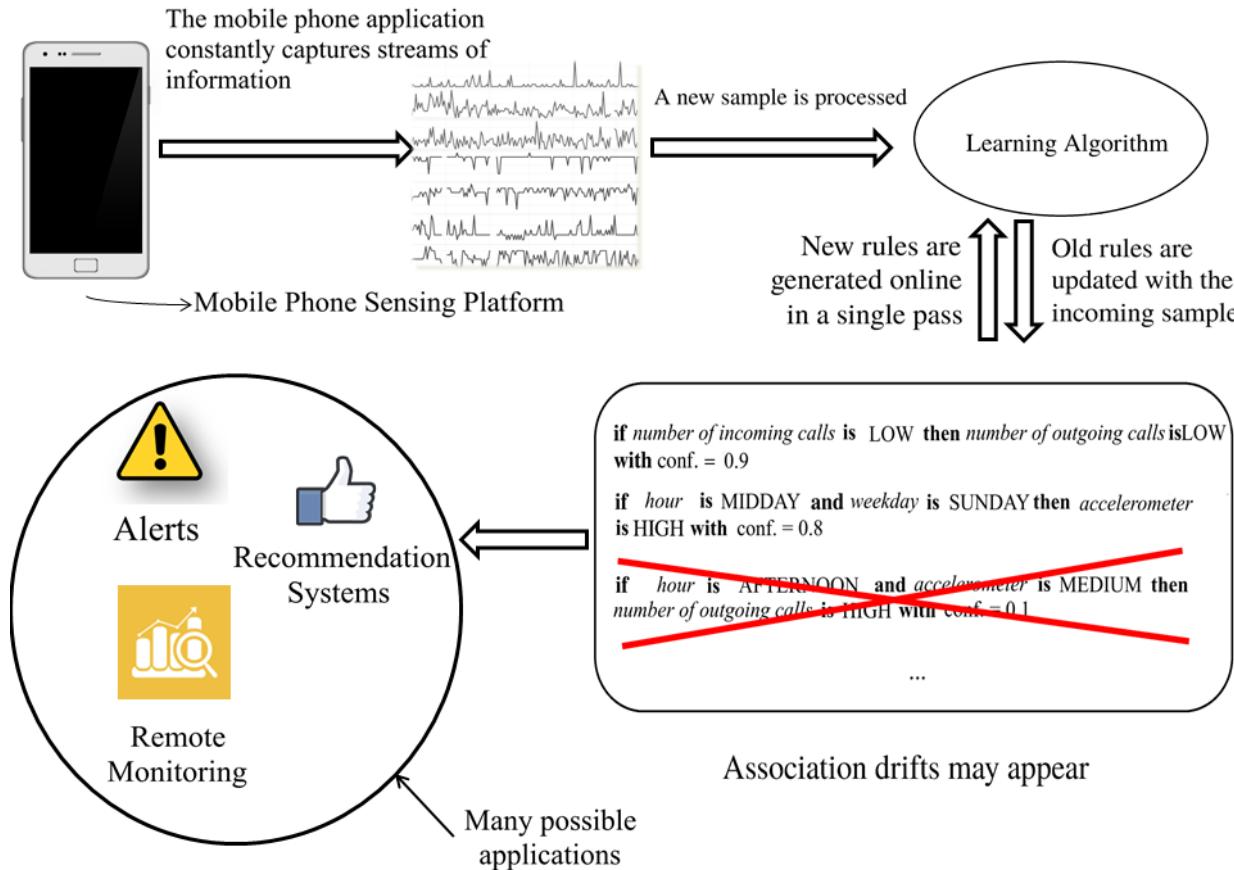
**IF NSWDemand is {S,M,L} and VICPrice is XS and transfer is {XS,S,M}**  
**THEN NSWPrice is {XS,S,M} [s: 0.78, c: 1.0]**



# Monitorización de Actividades en Telefonía Móvil mediante *Association Stream Mining*

- *Friends and Family Study* es una investigación realizada por el Media Lab (MIT) durante 2010 y 2011 [Aharony et al., 2011]
- Durante casi un año, casi todos los comportamientos, las comunicaciones y los detalles sociales de las vidas de los miembros de una comunidad se registran mientras siguen su vida cotidiana de manera normal. Un total de 130 sujetos participaron en este estudio
- Se acumuló una gran cantidad de datos. Este conjunto de datos incluye una gran colección de señales en teléfonos móviles que incluyen ubicación, actividades de comunicación (llamadas y SMS), aplicaciones instaladas, aplicaciones en ejecución, información del acelerómetro, información sobre dispositivos cercanos detectados por Bluetooth...
- Además, los participantes responden a cuestionarios (tanto por Internet como por teléfono) a intervalos regulares sobre la felicidad, el estrés, el sueño, la autopercepción de las relaciones...

# Monitorización de Actividades en Telefonía Móvil mediante Association Stream Mining



[E. Ruiz, J. Casillas, Monitoring Mobile Phone Activities by Association Stream Mining]

# Monitorización de Actividades en Telefonía Móvil mediante *Association Stream Mining*

- Los conjuntos de datos originales contienen una gran cantidad de archivos (7 GB en total) distribuidos en varios campos generados a partir de diferentes fuentes y con diferentes formatos
- Para el teléfono móvil, preparamos el flujo de datos siguiendo estos pasos: (1) filtrado de datos por fecha, (2) partición de datos por participante, (3) eliminación de datos duplicados y clasificación cronológica, e (4) integración de datos
- Para las encuestas, aplicamos estos pasos: (1) estudio preliminar de los datos de la encuesta y filtrado temporal, (2) clasificación de los datos por participante, e (3) integración de los datos de cada participante
- Aplicamos Fuzzy-CSar para extraer asociaciones del flujo de datos

# Monitorización de Actividades en Telefonía Móvil mediante Association Stream Mining

INFORMATION	SAMPLING FREQUENCY
Scanning of nearby Bluetooth devices	every 5 minutes
Call log	every time a call is sent or received
SMS log	every time a SMS is sent or received
Location	every 30 minutes
Apps in the phone	every 10 minutes
Running apps	every 30 seconds
Acceleration of the phone	one record every 20 minutes approx.
Battery status of the phone	variable

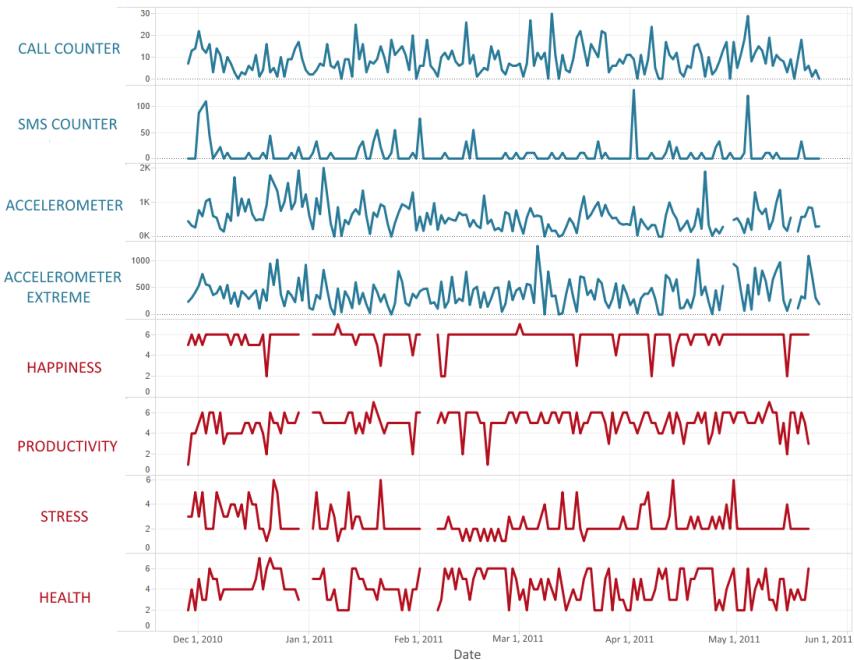
Datos originales

Flujo de datos



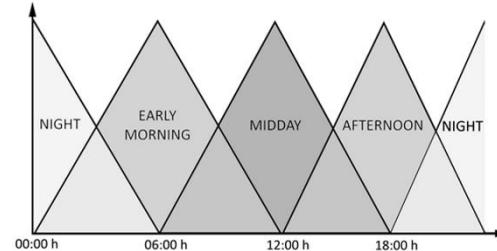
ATTRIBUTE
Day of the week (Monday-Sunday)
Time in minutes (from 0 to 1439)
Battery level in percentage (0-100)
SMS counter in the last 10 minutes
Incoming SMS counter in the last 10 minutes
Outgoing SMS counter in the last 10 minutes
Call type (incoming, outgoing, missed)
Call counter in the last 10 minutes
Incoming call counter in the last 10 minutes
Outgoing call counter in the last 10 minutes
Missed call counter in the last 10 minutes
Interval count accelerometer accumulated
Interval count extreme accelerometer accumulated
Interval duration accelerometer accumulated
Any application desinstalled
Running applications (x6)

minutely sampling



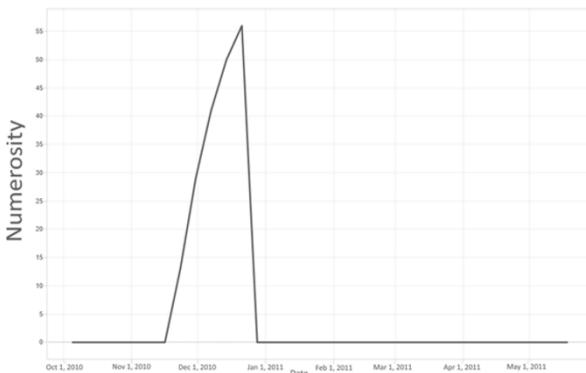
# Análisis de Reglas de Asociación Extraídas

Partición difusa  
de “hour”

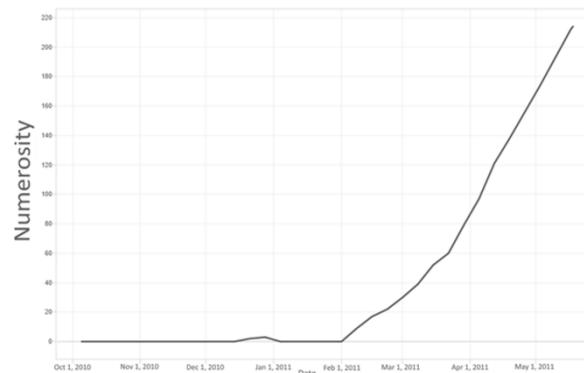


ID	Fuzzy Association IF-THEN Rule	Timestamp	Supp.	Conf.
Rule1	<b>IF</b> hour is MIDDAY or AFTERNOON <b>THEN</b> accelerometer is MEDIUM-HIGH	Nov 18, 2010	0.488	0.987
Rule2	<b>IF</b> number of missed calls is LOW and accelerometer is MEDIUM-HIGH <b>THEN</b> number of outgoing calls is LOW	Feb 02, 2011	0.620	0.980
Rule3	<b>IF</b> hour is NIGHT or EARLY MORNING and number of input SMS is LOW <b>THEN</b> number of outgoing calls is LOW	Oct 07, 2010	0.496	0.990

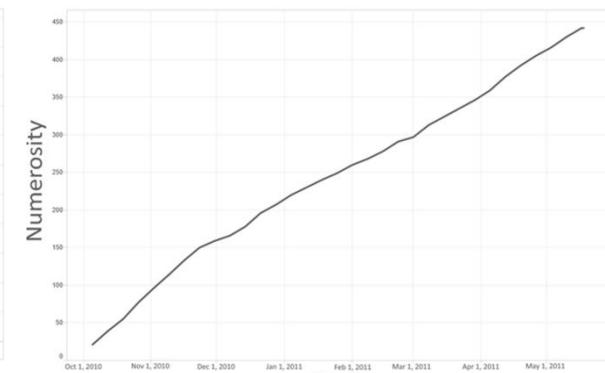
Rule 1



Rule 2

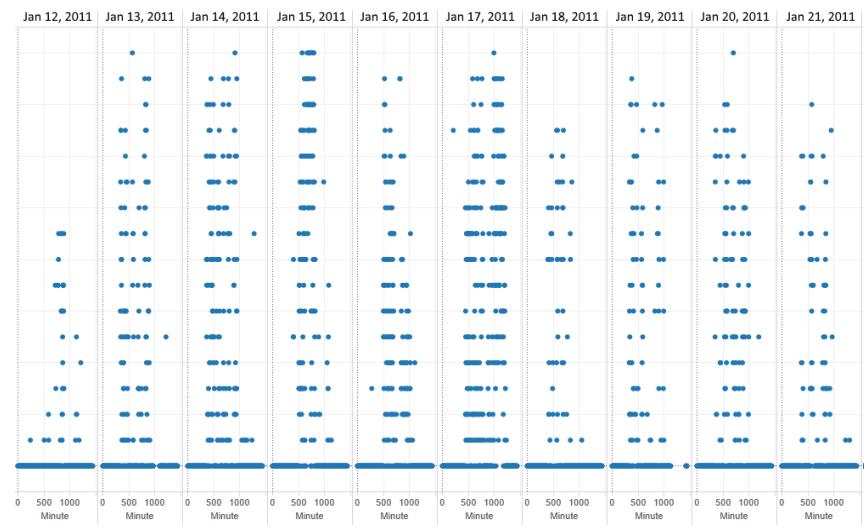
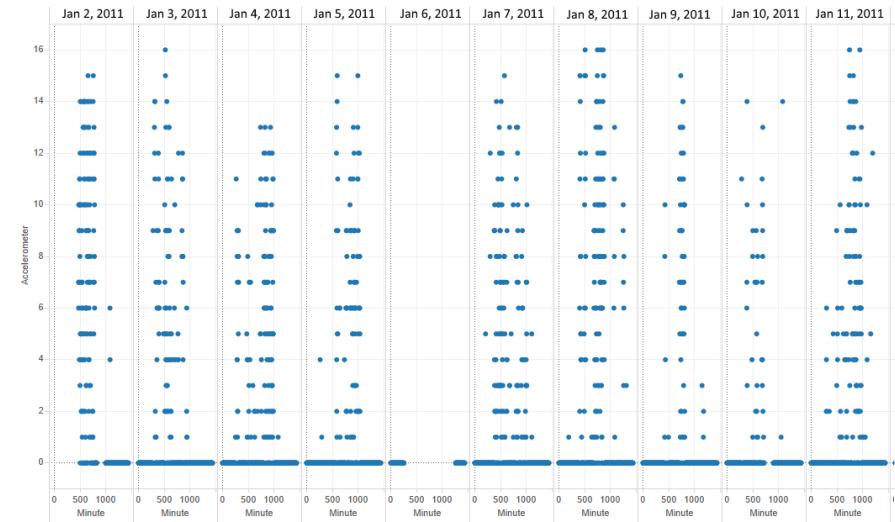
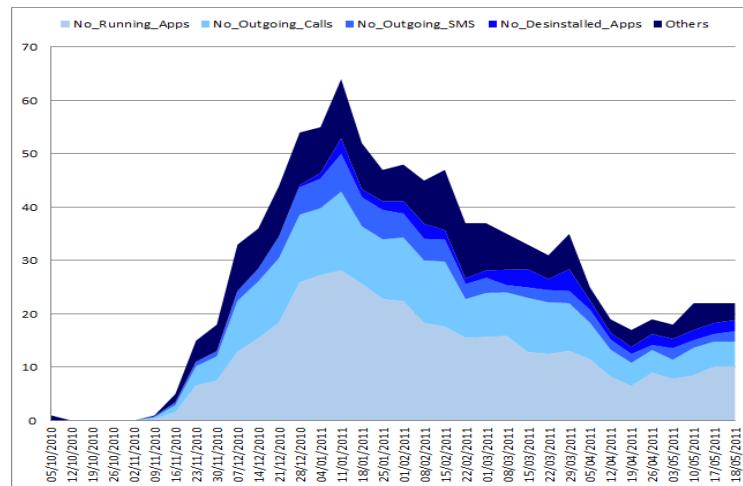
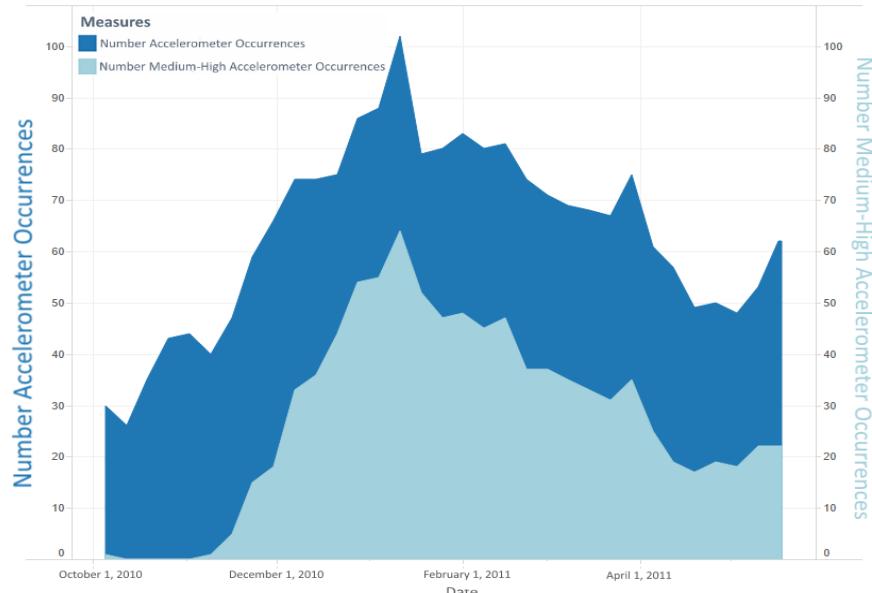


Rule 3



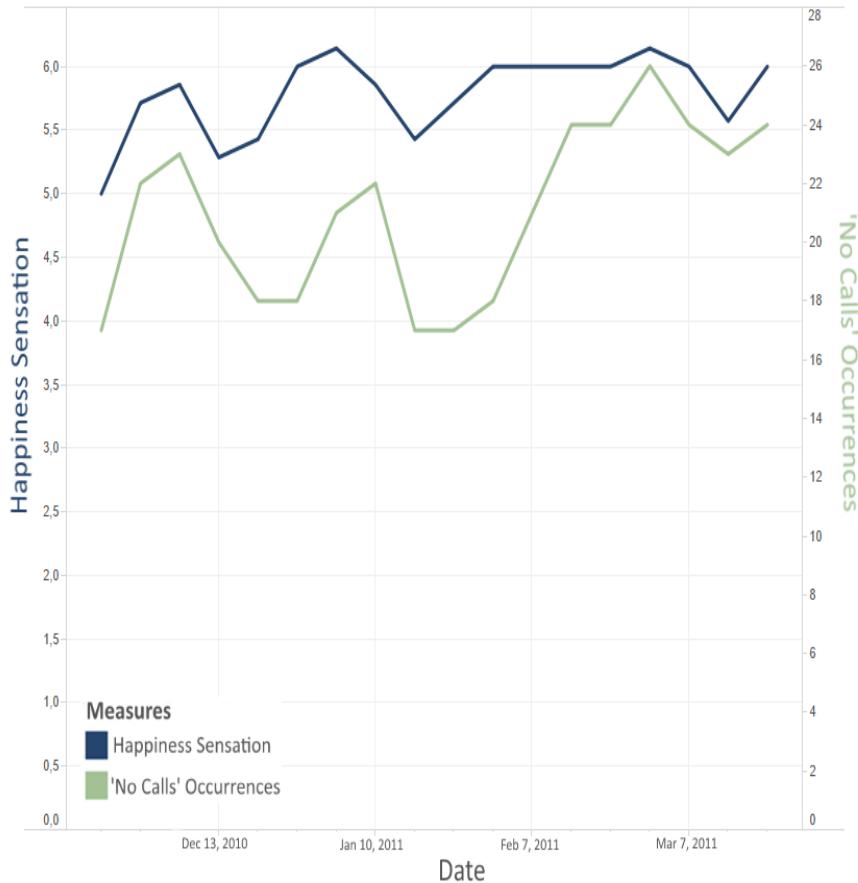
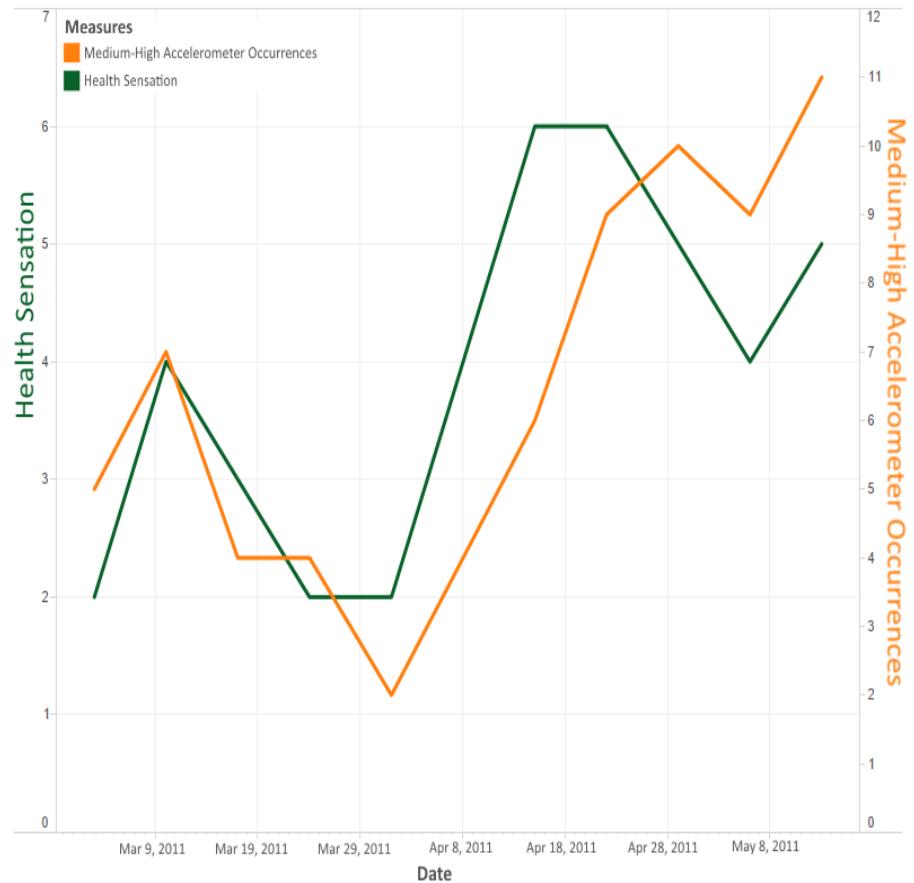
# Análisis de Reglas de Asociación Extraídas

Evolución a lo largo del tiempo de la cantidad de reglas sobre actividad física



# Análisis de Reglas de Asociación Extraídas

## actividad física y llamadas vs. condiciones de salud y felicidad





# Aplicación en Psicofisiología

- Se desea encontrar relaciones entre diferentes bioseñales
- A pesar del conocimiento de que el organismo humano es una red integrada de sistemas fisiológicos, probar interacciones entre ellos sigue siendo una tarea desalentadora que requiere métodos más sofisticados de análisis multivariante
- Aquí, usamos minería de flujo de asociación para explorar las relaciones entre una serie de variables fisiológicas durante los estados de reposo y después de la exposición a estímulos estresantes
- La comprensión de estas relaciones y las diferencias entre estados emocionales puede ayudar a controlar a los pacientes y prevenir reacciones psicosomáticas o trastornos psicológicos

# Aplicación en Psicofisiología

- EDA: actividad electrodermal



- resp: respiración



- EMG: electromiograma

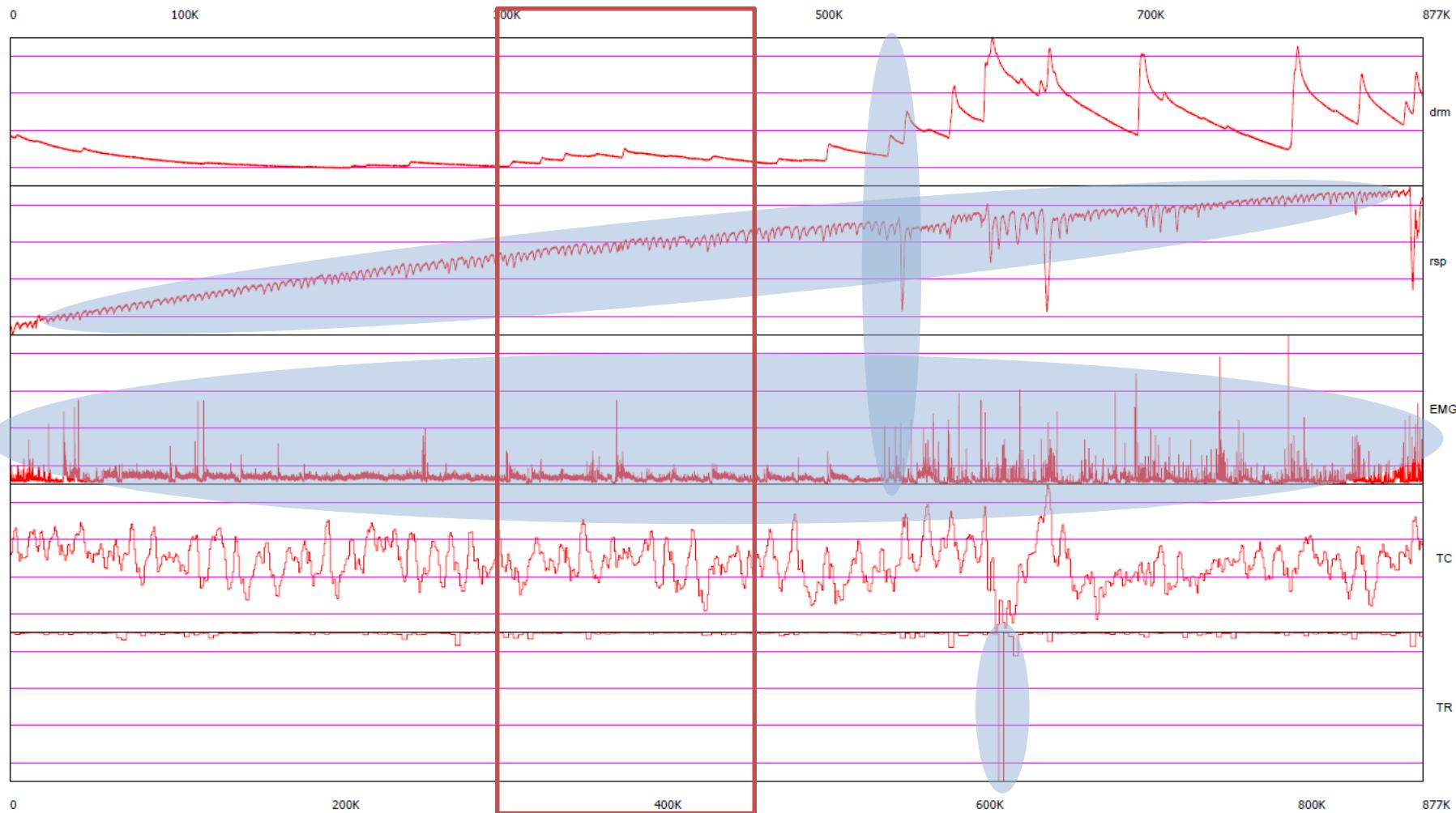


- HR: ritmo cardiaco

Frecuencia de  
muestreo: 1000Hz

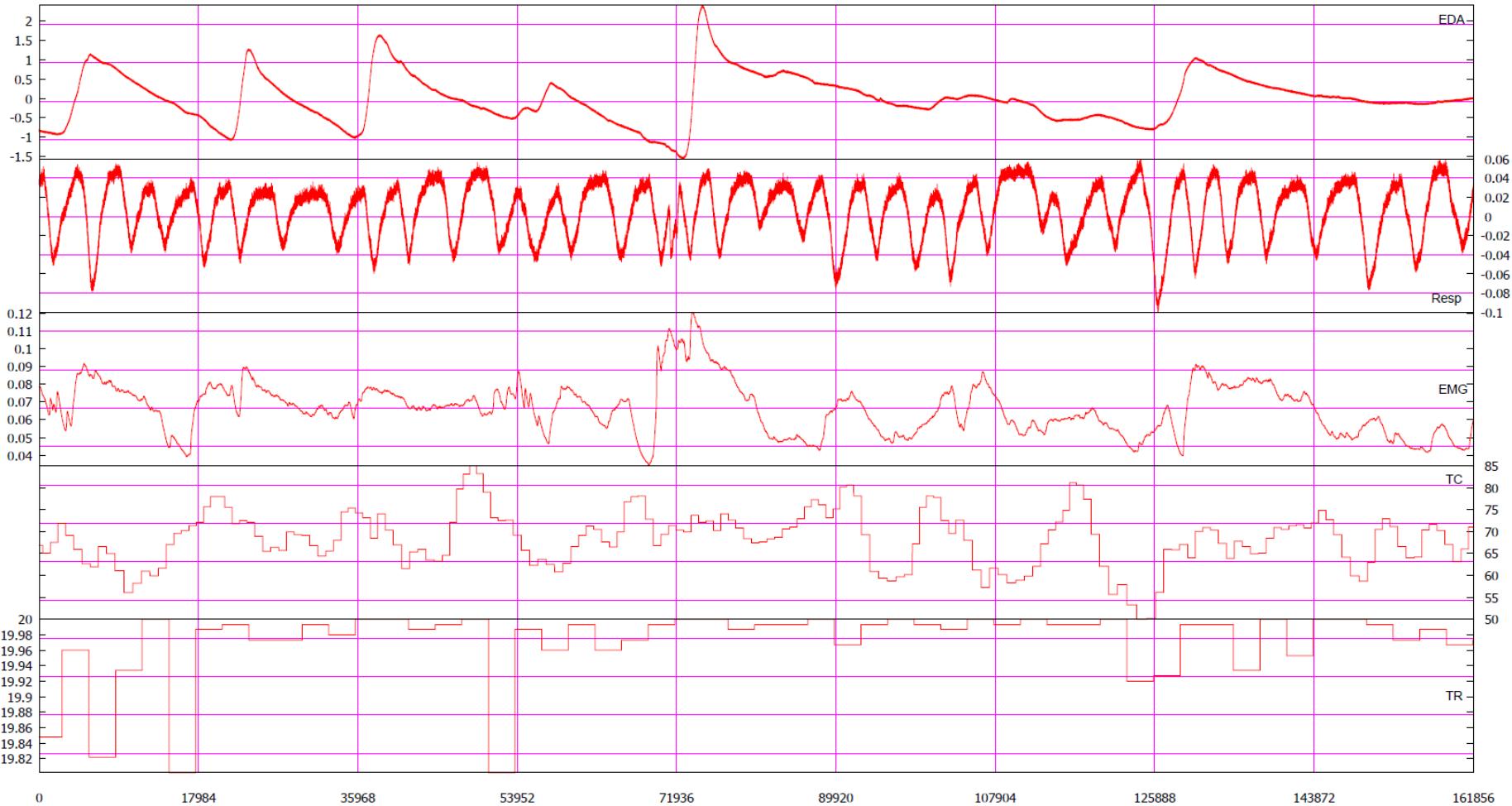
- RR: ritmo respiratorio

# Aplicación en Psicofisiología



Raw biosignals must be preprocessed: smooth curves, remove artifacts, correct artificial trends, synchronizing time responses, ...

# Aplicación en Psicofisiología



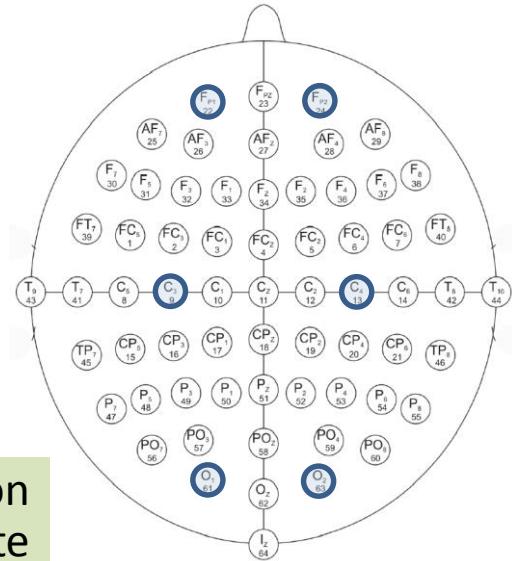
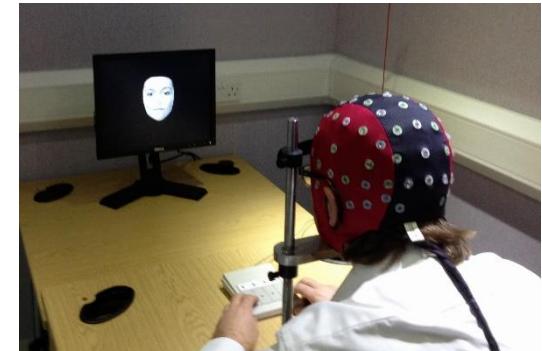
# Aplicación en Psicofisiología

- **IF** *resp* is L and *HR* is M and *RR* is L **THEN** *EDA* is M  
[s: 0.111, c: 0.819, l: 1.154, a: 0.423]  
tstamp: 12192 (12s), exp: 149664 (2m30s)
- **IF** *EDA* is M and *resp* is M **THEN** *EMG* is M  
[s: 0.129, c: 0.829, l: 1.224, a: 0.448]  
tstamp: 132635, exp: 29221 (29s)
- **IF** *resp* is XL **THEN** *HR* is M  
[s: 0.122, c: 0.785, l: 1.085, a: 0.391]  
tstamp: 8519, exp: 153337 (2m33s)
- **IF** *EDA* is L **THEN** *RR* is XL  
[s: 0.107, c: 0.862, l: 1.718, a: 0.613]  
tstamp: 68630, exp: 93226 (1m33s)



# Análisis de Electroencefalograma (EEG)

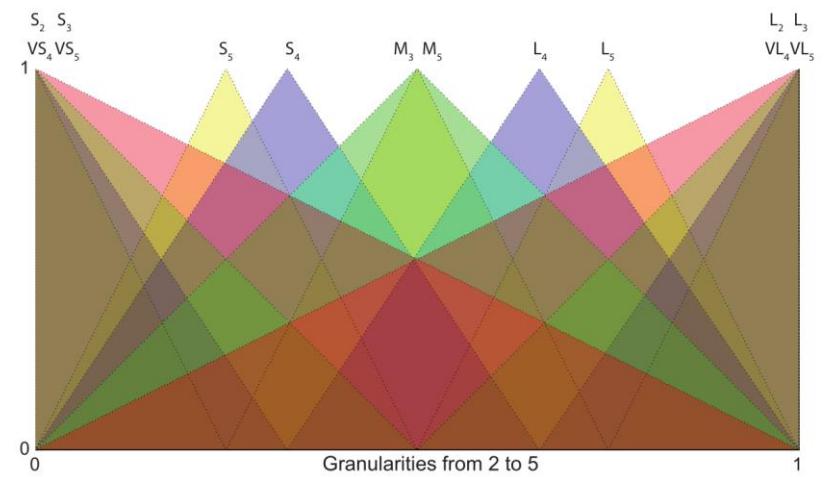
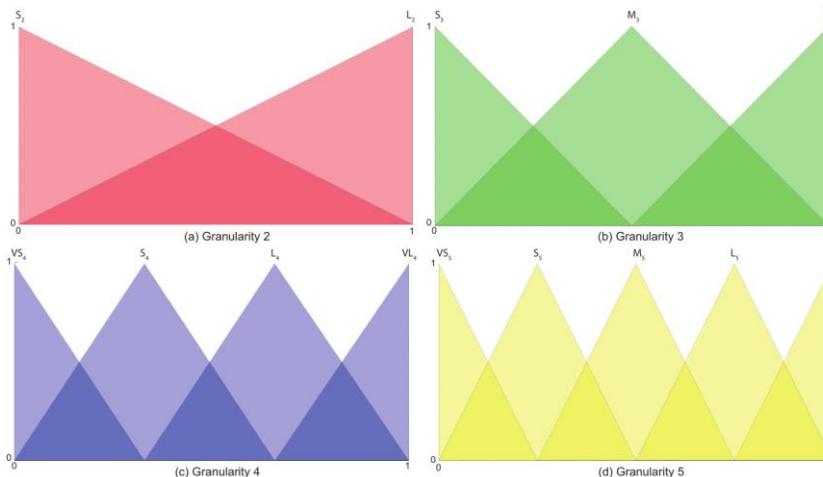
- Se analizan sujetos sedentarios y otros entrenados (deportistas de triatlón) entre 22-25 años
- Durante una hora, se captura el EEG de sujetos mientras reaccionan a avisos en pantalla con el objetivo de analizar su concentración
- Muestra a 256 Hz. Después de una hora se genera casi un millón de datos
- Se han seleccionado 6 electrodos:  
Frontales: F<sub>p1</sub>, F<sub>p2</sub>  
Centrales: C<sub>3</sub>, C<sub>4</sub>  
Occipitales: O<sub>1</sub>, O<sub>2</sub>



E. Ruiz, J. Casillas, Adaptive fuzzy partitions for evolving association rules in big data stream, International Journal of Approximate Reasoning 93 (2018) 463-486, DOI: [10.1016/j.ijar.2017.11.014](https://doi.org/10.1016/j.ijar.2017.11.014)

# Mejoras en Fuzzy-CSar: Adaptive Fuzzy Partitions (AFP)

- La representación y operadores genéticos se rediseñan para tratar particiones difusas de diferentes granularidades
- A diferencia de la clasificación, aquí el objetivo es proporcionar al experto **patrones independientes** que expliquen las relaciones de asociación individuales entre los atributos. Por lo tanto, el uso de diferentes granularidades en diferentes reglas es compatible con la comprensión de los expertos



# Mejoras en Fuzzy-CSar: Actualización online del dominio de los atributos

```
procedure Streaming-Mean-Variance( sum at time t, mean at
time t,  $M_2$  at time t, n at time t,  $\alpha$ ,  $\beta$ , x )
```

**Data:**  $n$  is the number of samples processed at time  $t$   
 $x$  is the value of a new data for a certain attribute  
 $\mu$  and  $\sigma$  are the stream mean and standard deviation  
 $sum, M_2, \alpha$  and  $\beta$  are real values

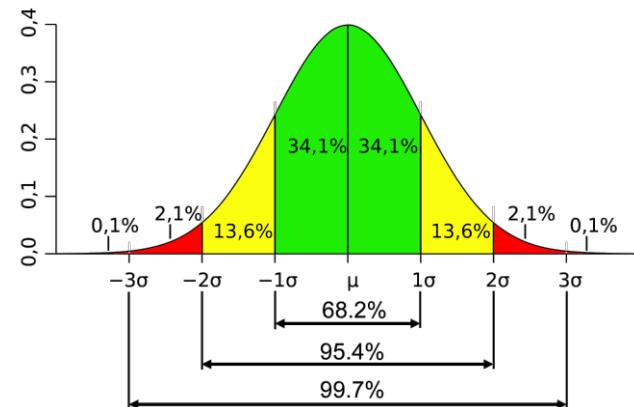
**Result:**  $min$  and  $max$  at time  $t + 1$

**begin**

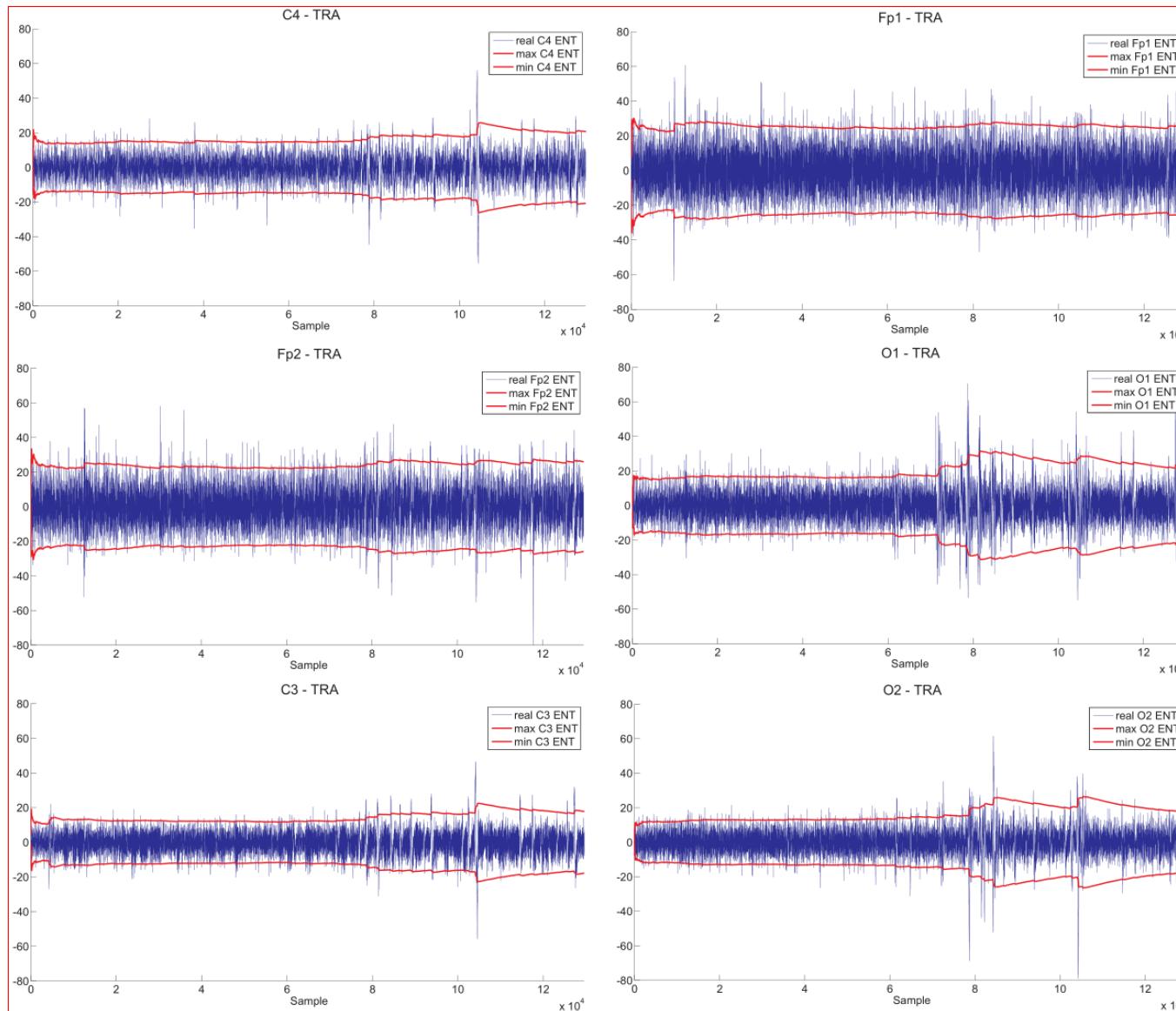
```
temp ← sum ·  $\alpha$  + 1
diff ← x -  $\mu$ 
R ← diff / temp
 $\mu$  ←  $\mu$  + R
 $M_2$  ←  $M_2$  ·  $\alpha$  + (sum ·  $\alpha$  · diff · R)
sum ← temp
 $\sigma^2$  ← ( $M_2$  · n) / (sum · (n - 1))
(min, max) ← ( $\mu$  -  $\beta$  ·  $\sigma^2$ ,  $\mu$  +  $\beta$  ·  $\sigma^2$ )
```

**end**

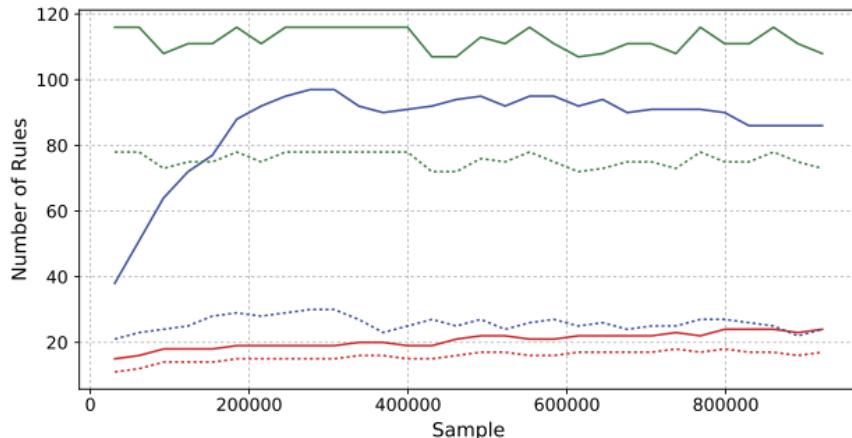
$$\beta = 2.5$$



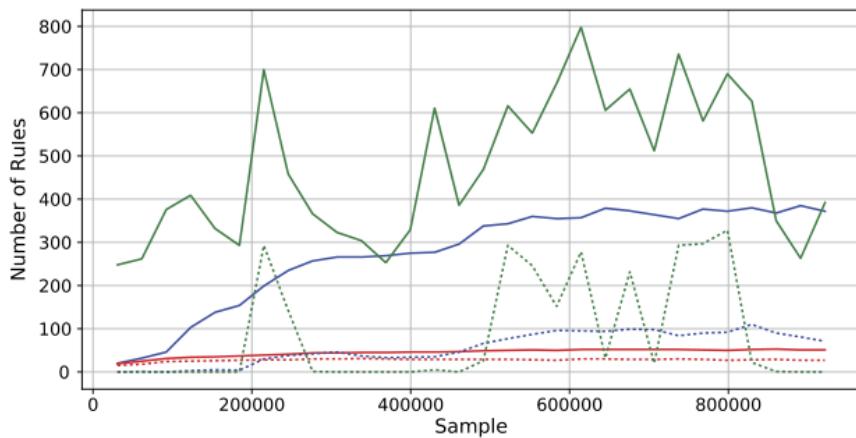
# El Dominio de las Variables se Asume Desconocido y Evoluciona con los Datos



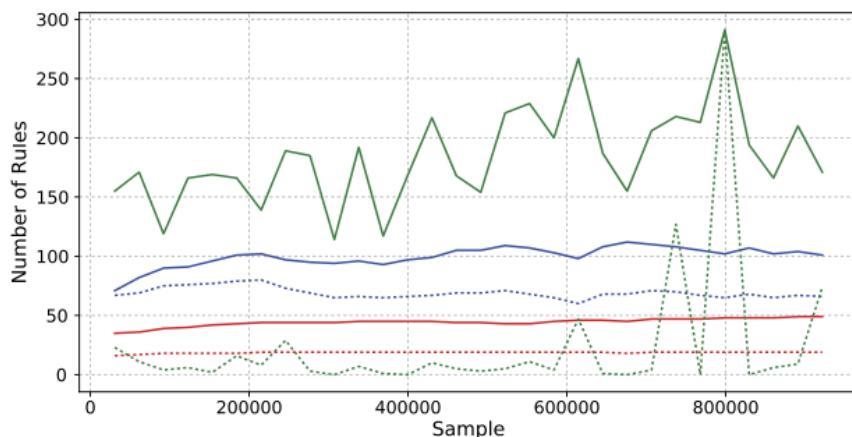
# AFP Genera más Reglas



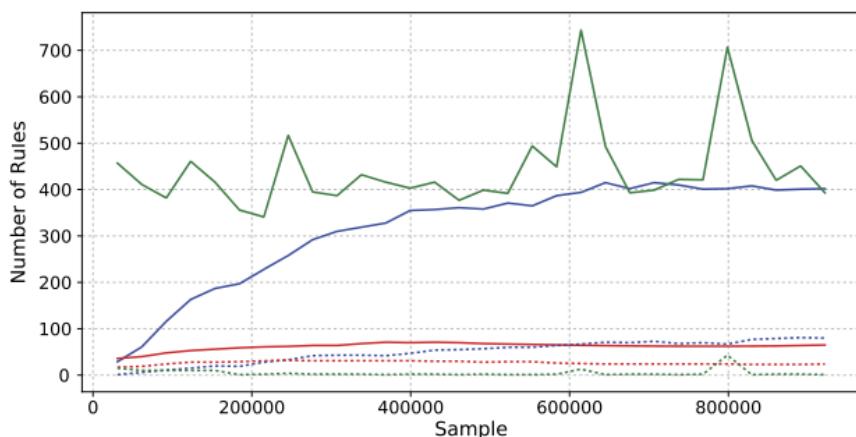
(a) TRA, absolute domains,  $supp \geq 0.25$



(b) TRA, evolving domains,  $supp \geq 0.02$

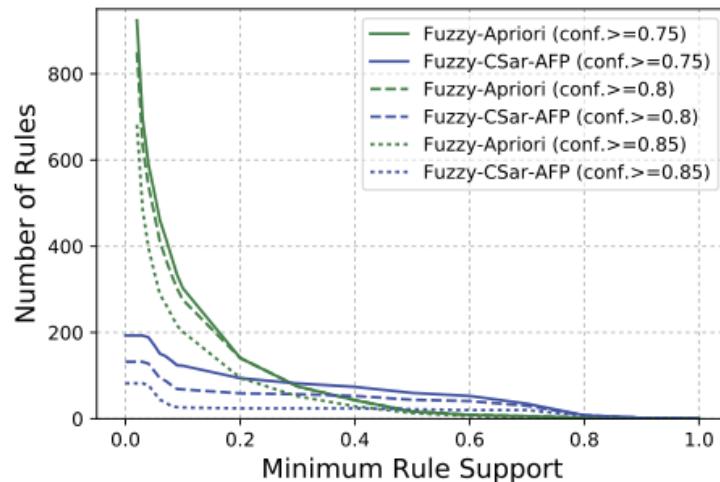


(c) SED, absolute domains,  $supp \geq 0.10$

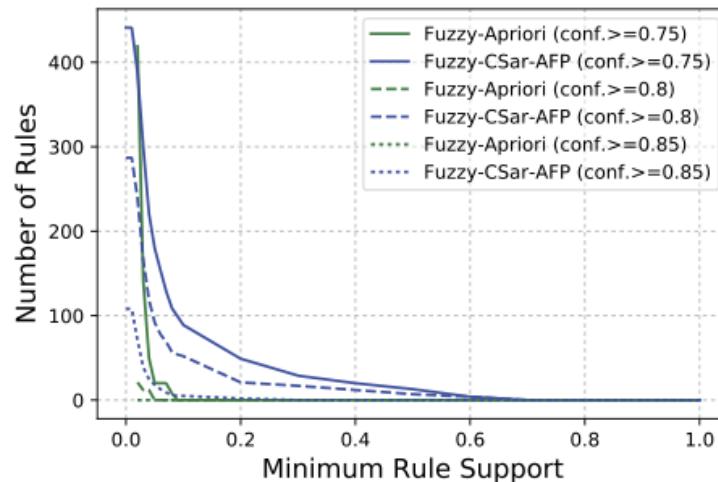


(d) SED, evolving domains,  $supp \geq 0.02$

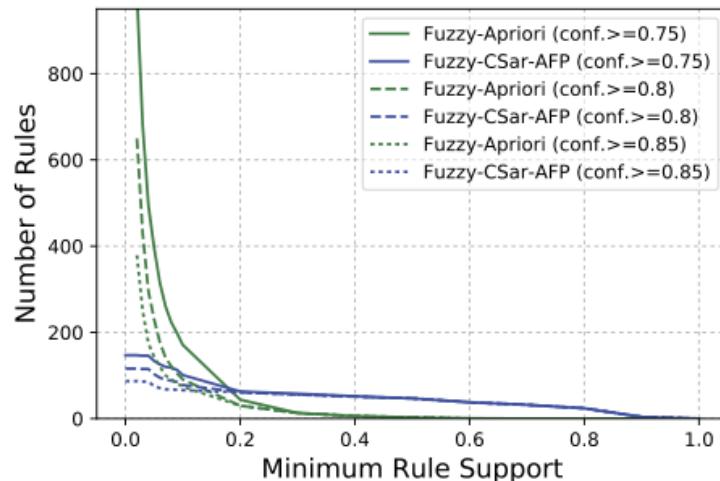
# Estas Reglas son más Precisas...



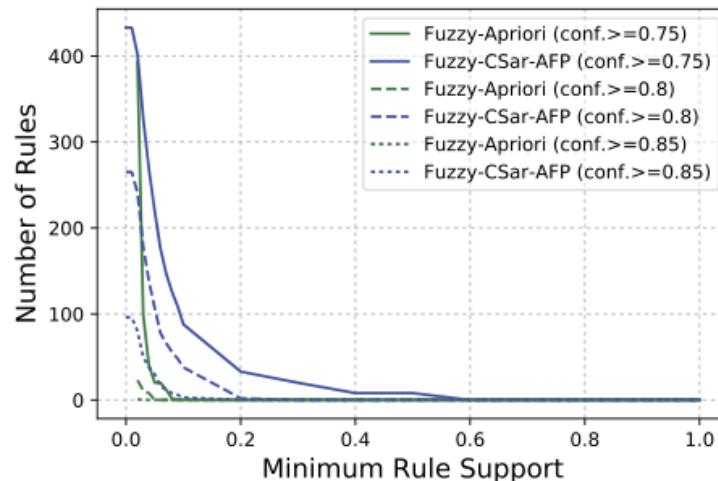
(a) *TRA* subject, absolute domains



(b) *TRA* subject, evolving domains

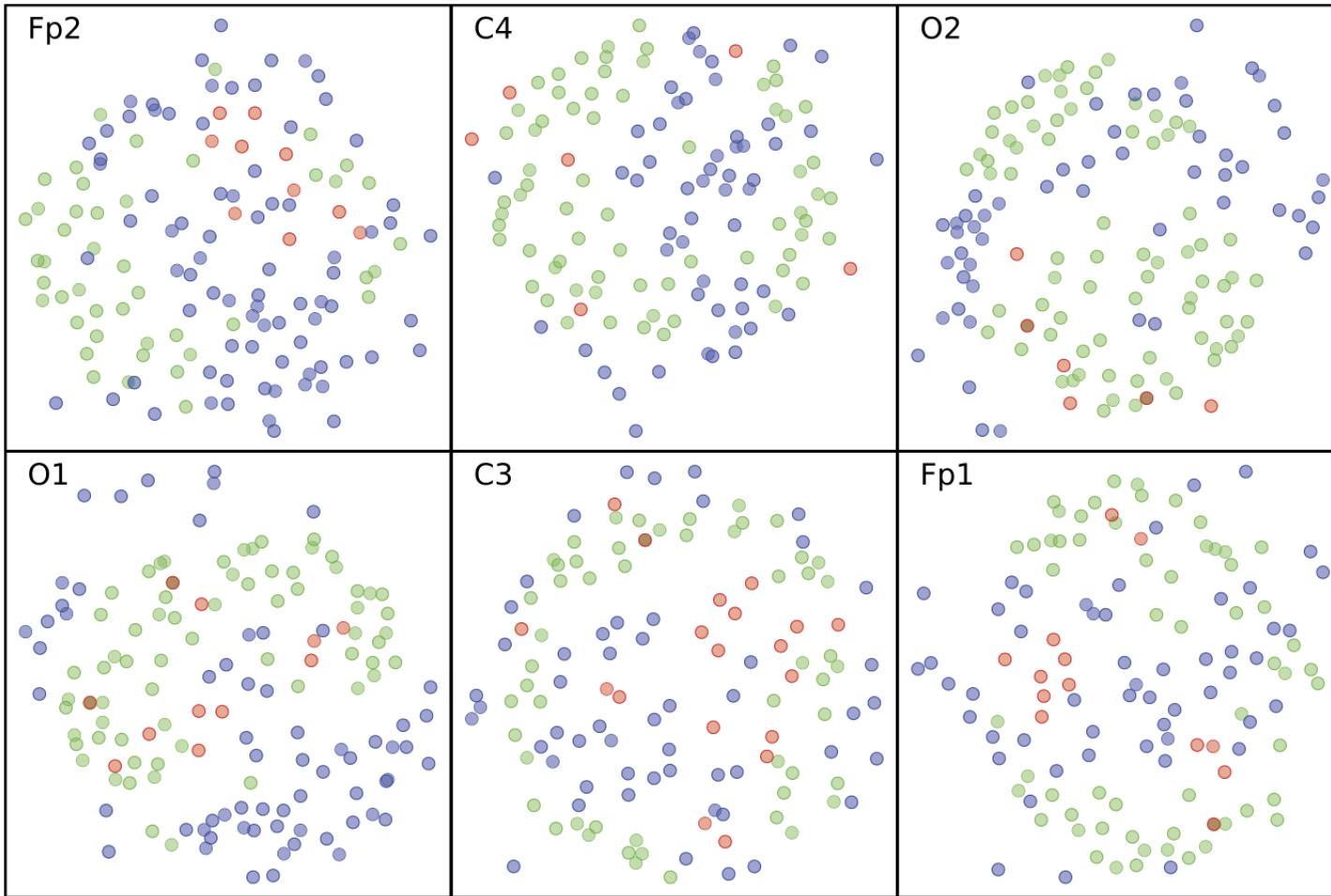


(c) *SED* subject, absolute domains



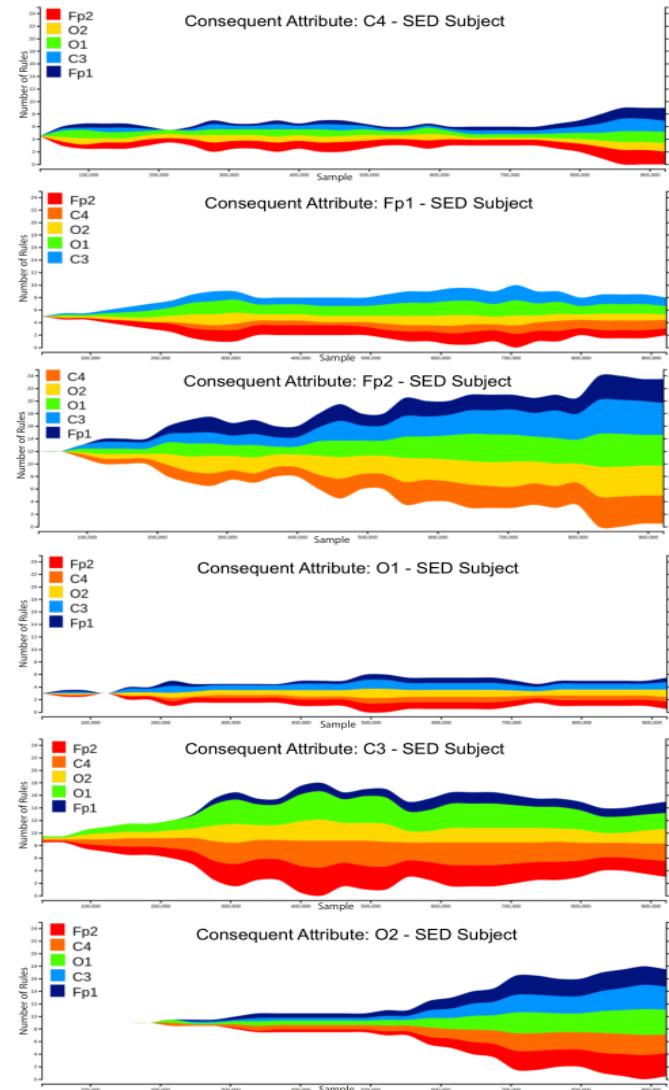
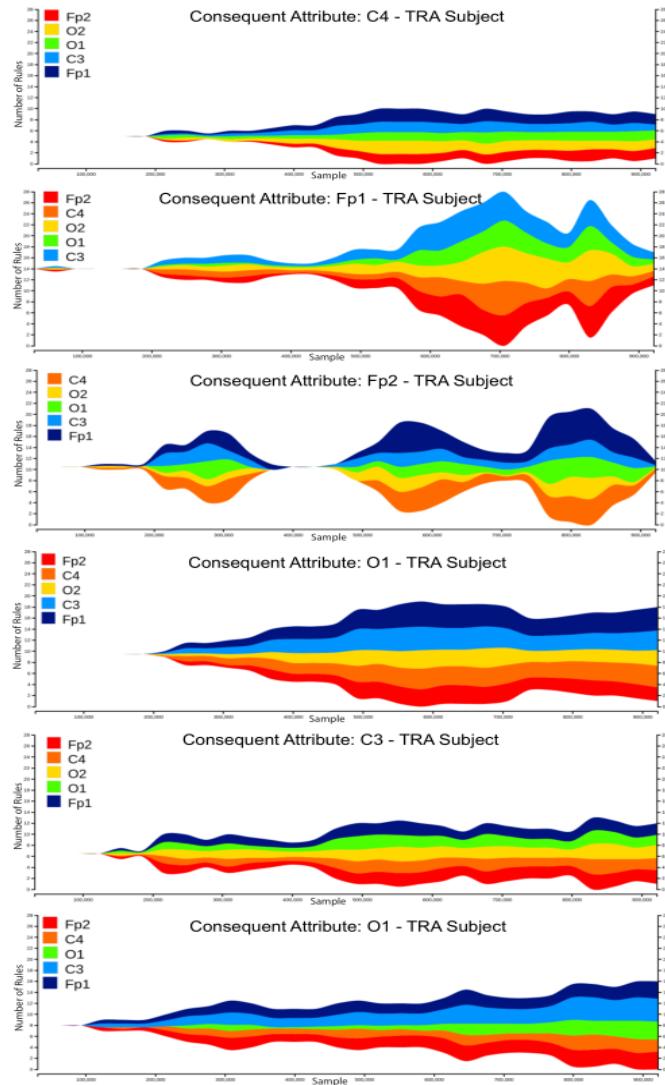
(d) *SED* subject, evolving domains

# ...y más Diversas



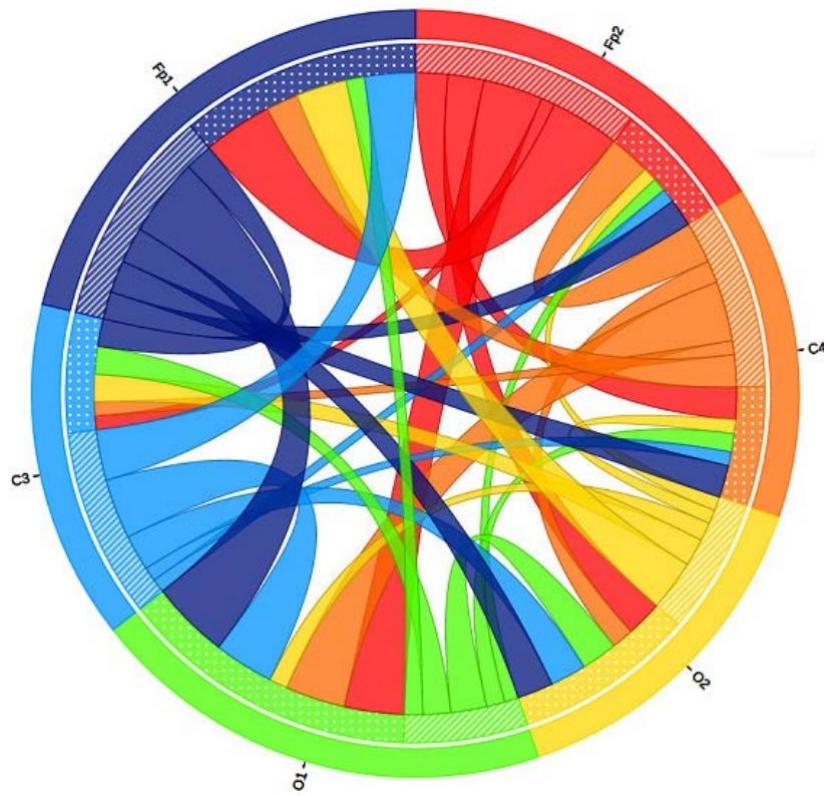
We use Multidimensional Scaling (MDS) with *ad hoc* distance metric to compare diversity of fuzzy association rules.  
(blue = Fuzzy-CSar-APP, red = Fuzzy-CSar, green = Fuzzy-Apriori)

# Genera y Mantiene un Buen Número de Reglas de Asociación con Confianza > 0,8

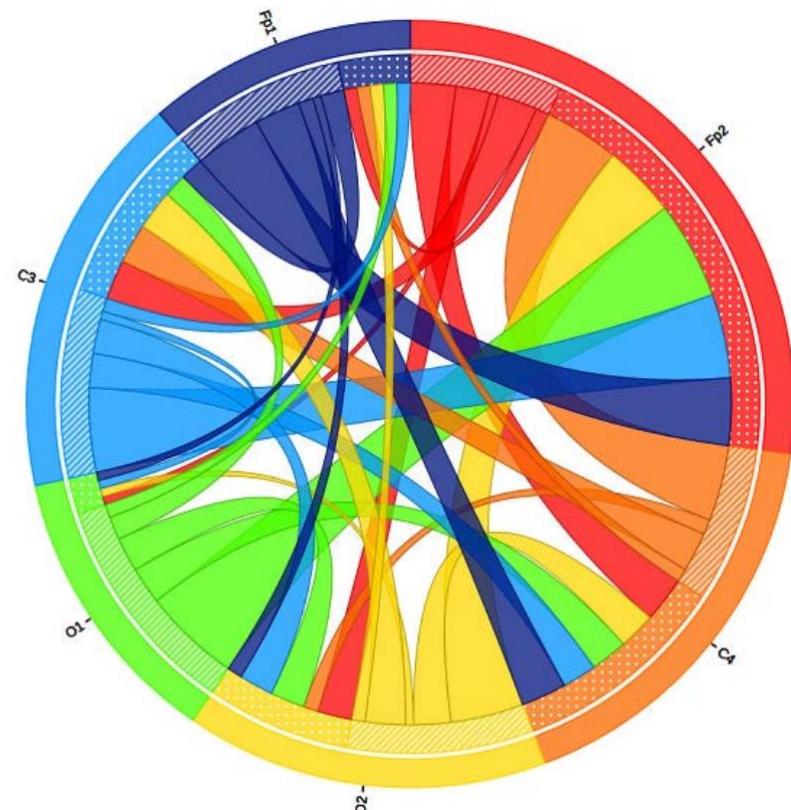


# Influencia de las Variables en las Reglas

Sujeto entrenado



Sujeto sedentario



sop.  $\geq 0.05$ , conf.  $\geq 0.85$  (rayado: antecedente, punteado: consecuente)

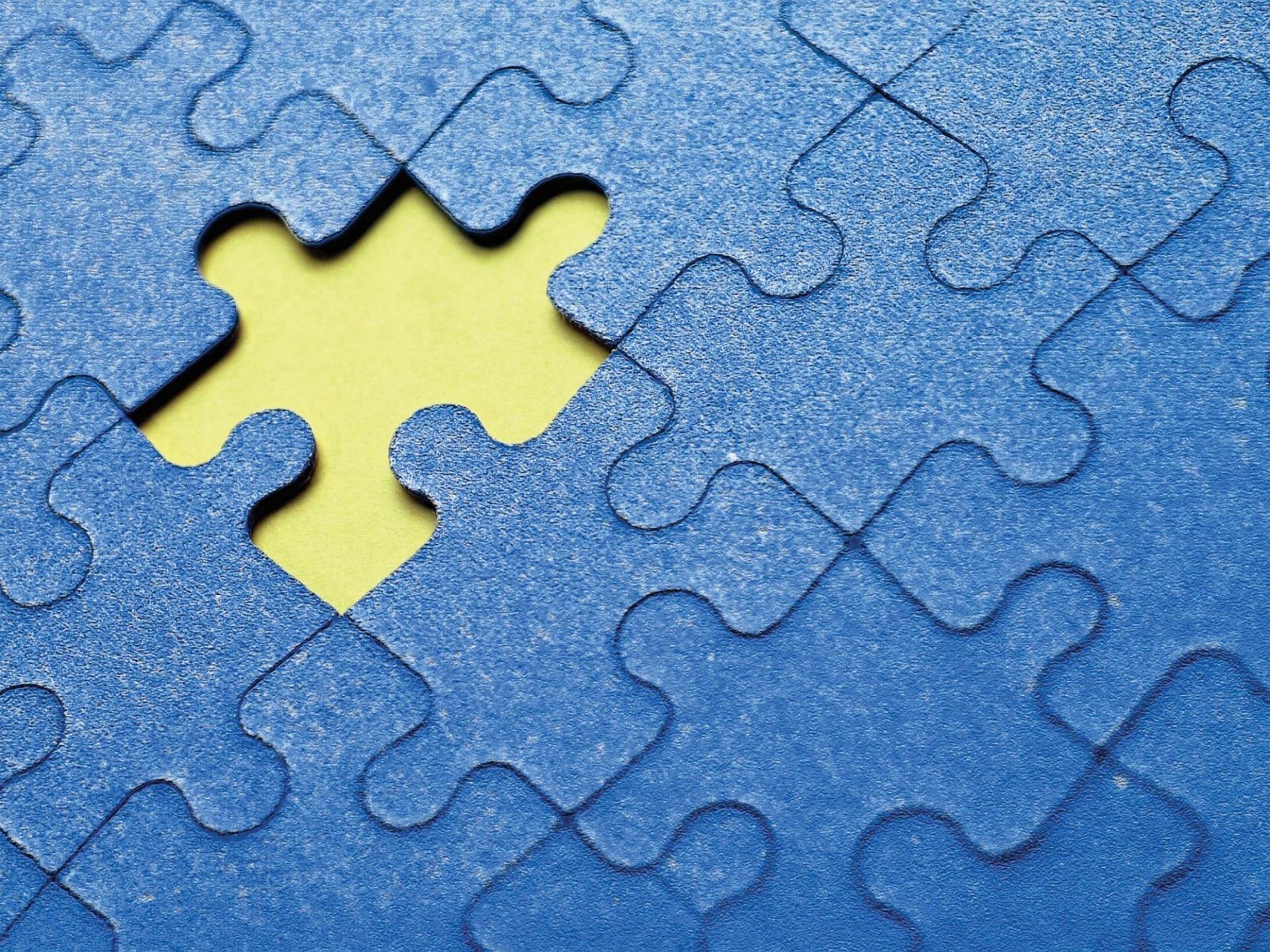
*Software*

# Software

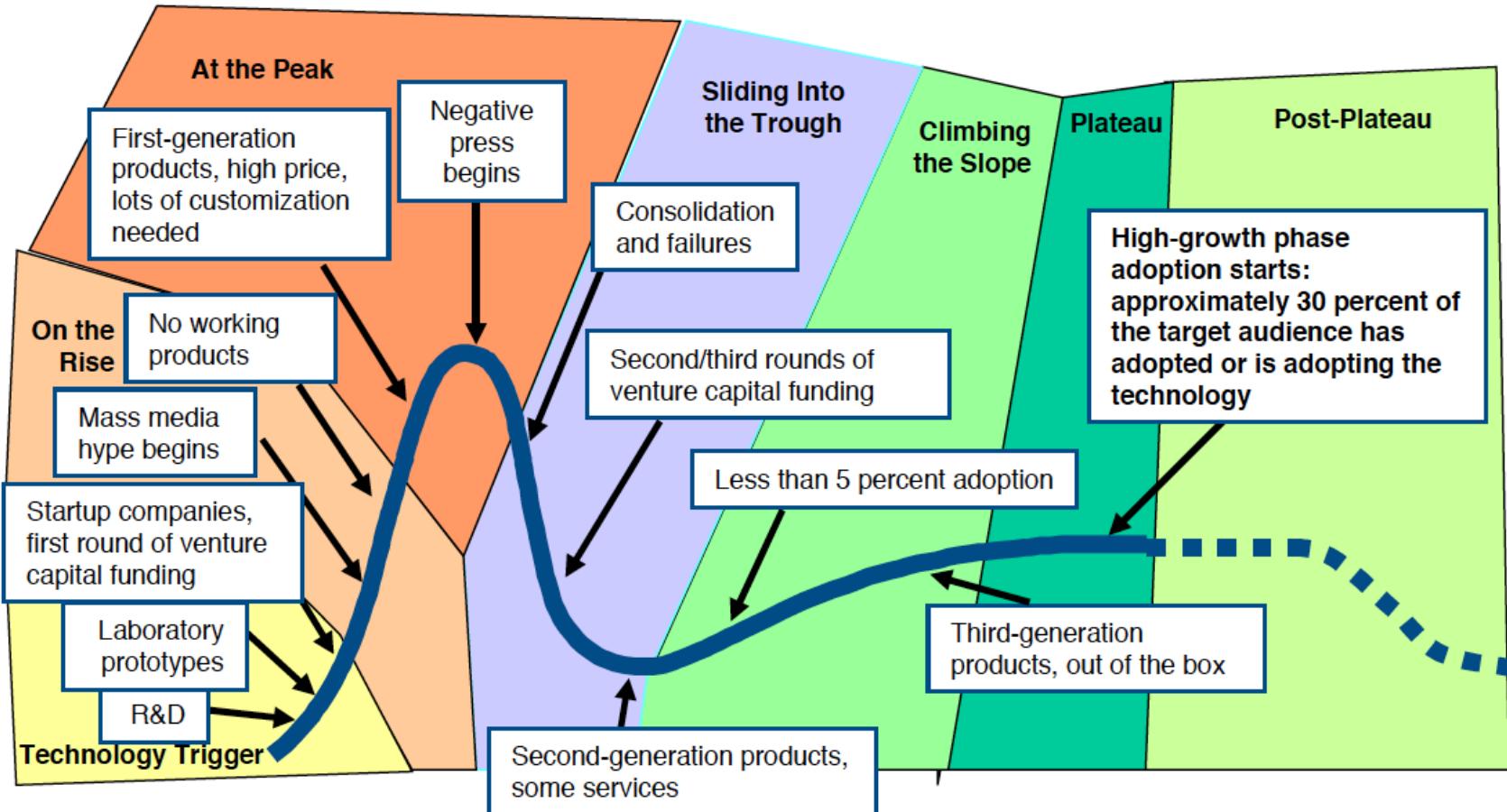
- RapidMiner - <http://www.rapidminer.com/>:
  - Software de fuente abierta para descubrimiento de conocimiento y minería de datos que también puede abordar problemas de flujo de datos y *concept drift*. Para ello, es necesario combinarlo con un *plugin* para *data stream*
- Massive Online Analysis (MOA) - <http://moa.cms.waikato.ac.nz/>:
  - Software de fuente abierta en Java específico para minería de flujo de datos. Contiene métodos de *concept drift*, *clustering*, visualización, generadores de flujo de datos sintéticos tipo SEA, etc. Soporta interacción bidireccional con Weka

# *Software*

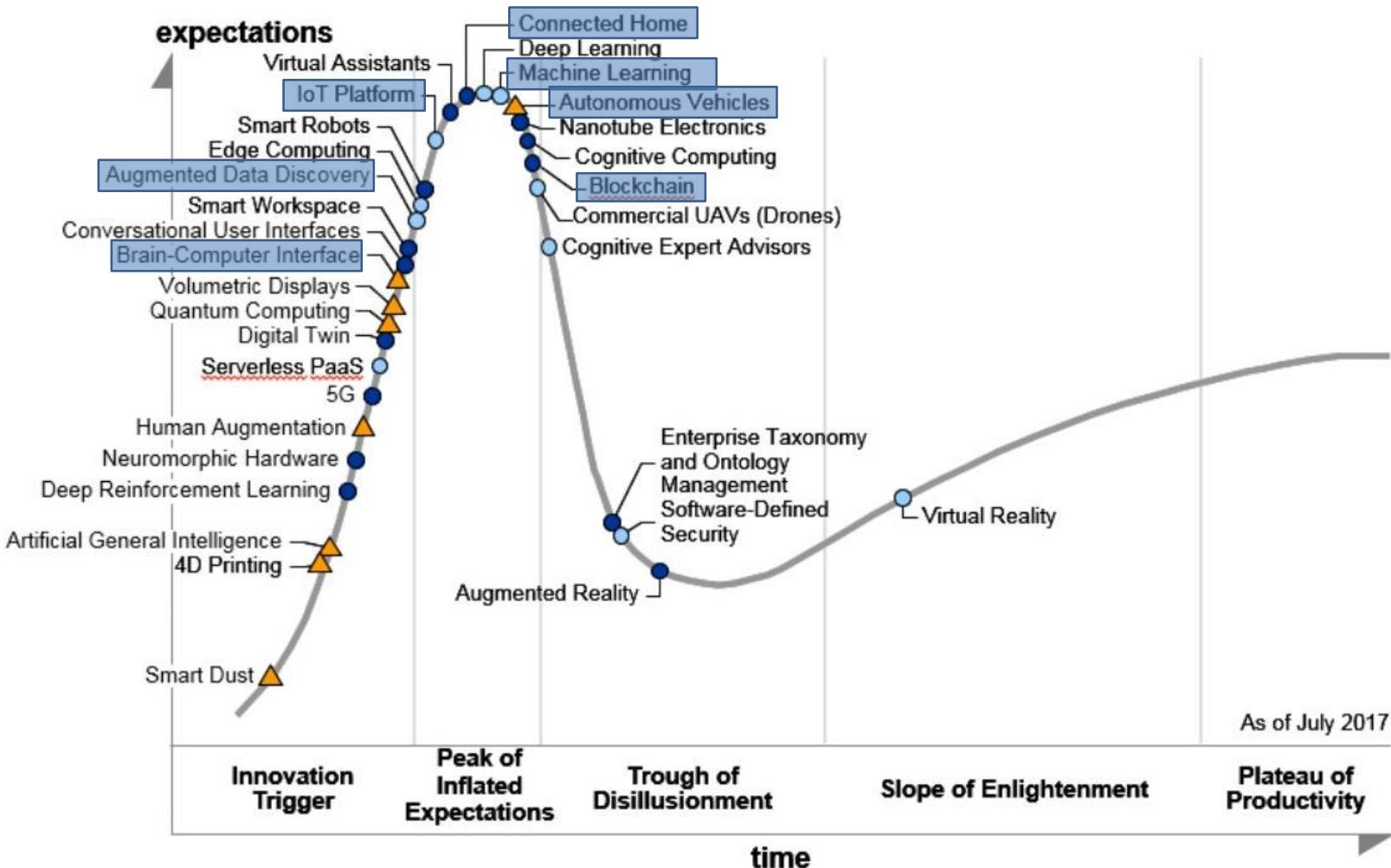
- No distribuidos
  - VFML (C) - <http://www.cs.washington.edu/dm/vfml/>
  - Vowpal Wabbit (C++) -  
[https://github.com/JohnLangford/vowpal\\_wabbit/](https://github.com/JohnLangford/vowpal_wabbit/)
  - Sofia-ML (C++)
- Distribuidos
  - Streams (Java)
  - Jubatus (C++) - <http://jubat.us/en/>
  - Apache Flink (Java, Scala, Python) - <https://flink.apache.org/>
- streamDM (C++ y Spark)  
<http://huawei-noah.github.io/streamDM/>



# Gartner Hype Cycle



# Oportunidades de *Data Stream* en Tecnologías Emergentes



Gartner

Years to mainstream adoption:

○ less than 2 years    ○ 2 to 5 years    ● 5 to 10 years    ▲ more than 10 years    ⊗ obsolete  
⊗ before plateau

# Líneas Abiertas de Investigación

- Buenos repositorios de datos reales
- Preprocesado dinámico de flujo de datos
- Manejo de muchas variables
- Reglas de asociación para flujo de datos «puro»
- Aprendizaje semi-supervisado
- Visualización
- Interactividad con el usuario
- Aplicación en Big Data e integración en computación distribuida MapReduce y Spark

# Bibliografía

- João Gama, **Knowledge Discovery from Data Streams**, Chapman and Hall/CRC, 2010
- Albert Bifet, Geoff Holmes, Richard Kirkby, Bernhard Pfahringer, **Data Stream Mining – A Practical Approach**, Centre for Open Software Innovation, 2011
- Moamar Sayed-Mouchaweh, Edwin Lughofer (Eds.), **Learning in Non-Stationary Environments: Methods and Applications**, Springer, 2012
- Albert Bifet, Ricard Gavaldà, Geoff Holmes, Bernhard Pfahringer, **Machine Learning for Data Streams**, The MIT Press, 2018



# ¿Quieres saber si has aprendido los conceptos principales sobre minería de flujo de datos?

Acepta el reto y completa esta prueba de 12 preguntas tipo test

(Su resultado no influye en la calificación del curso, es solo para ponerte a prueba y saber qué has entendido bien o en qué tienes alguna carencia que merece ser revisada)

**casillas\_ugr has challenged you to a game of kahoot!**



**Minería de Flujo de Datos**  
Created 8 hours ago  
casillas\_ugr

To play this challenge, you need the Kahoot! app.

[Download on the App Store](#) [GET IT ON Google Play](#)

Find out more at [kahoot.com](https://kahoot.com)

# Kahoot!

<https://kahoot.it/challenge/0144395>

