Escuela Politécnica Superior de Jaén

Máster en Ingeniería Informática

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Jaén

Trabajo Fin de Máster

ALGORITMOS EVOLUTIVOS DE MINERÍA DE DATOS DESCRIPTIVA PARA FLUJOS CONTINUOS DE DATOS

Alumno: Ángel Miguel García Vico

Tutor: Prof. D. Cristóbal José Carmona del Jesus
Dpto: Informática

Tutor: Prof. D. Pedro González García
Dpto: Informática

Septiembre, 2019

Universidad de Jaén
Escuela Politécnica Superior de Jaén
Departamento de Informática

Don CRISTÓBAL JOSÉ CARMONA DEL JESUS y PEDRO GONZÁLEZ GARCÍA, tutores del trabajo fin de máster titulado: ALGORITMOS EVOLUTIVOS DE MINERÍA DE DATOS DESCRIPTIVA PARA FLUJOS CONTINUOS DE DATOS, que presenta ÁNGEL MIGUEL GARCÍA VICO, autoriza su presentación para defensa y evaluación en la Escuela Politécnica Superior de Jaén.

Jaén, SEPTIEMBRE de 2019

El alumno:

GARCIA VICO ANGEL MIGUEL - 26256679V

Firmado digitalmente por GARCIA VICO ANGEL MIGUEL - 26256679V
Fecha: 2019.09.05 19:52:09 +02'00'

ÁNGEL MIGUEL GARCÍA VICO

Los tutores:

CARMONA DEL JESUS CRISTOBAL JOSE - 75102603J

Firmado digitalmente por CARMONA DEL JESUS CRISTOBAL JOSE - 75102603J
Fecha: 2019.09.06 09:06:27 +02'00'

CRISTÓBAL JOSÉ CARMONA DEL JESUS

GONZALEZ GARCIA PEDRO - 26210879X

Firmado digitalmente por GONZALEZ GARCIA PEDRO - 26210879X
Fecha: 2019.09.05 20:19:22 +02'00'

PEDRO GONZÁLEZ GARCÍA

Agradecimientos

A mi familia, por todo el esfuerzo y sacrificio empleado. Aquí está el fruto de vuestro empeño.

A todos mis amigos, por esos ratos inolvidables de desconexión que tan bien han venido.

A mis tutores, Cristóbal y Pedro, por sus ideas, sugerencias y revisiones las cuales sin ellas este trabajo no hubiera salido adelante.

A todos vosotros, muchas gracias.

Índice general

Introducción **1**

1. INTRODUCCIÓN **1**

1.1. Objetivos 4

1.2. Estructura de la memoria 5

2. Revisión del estado del arte **6**

2.1. Minería de flujo de datos 6

2.1.1. Definición del problema 7

2.1.2. Flujos de datos dinámicos y cambio de concepto 8

2.1.3. Diseño de métodos con manejo del cambio de concepto 12

2.1.3.1. Procesamiento de datos 12

2.1.3.2. Proceso de aprendizaje 15

2.1.3.3. Proceso de monitorización 15

2.1.3.4. Proceso de adaptación 16

2.1.3.5. Proceso de evaluación 17

2.2. Minería de patrones emergentes 18

2.2.1. Medidas de calidad en EPM 20

2.2.2. Tipos de patrones emergentes 23

2.2.3. Algoritmos de minería de patrones emergentes 26

3. Algoritmo para la extracción de patrones emergentes difusos en flujos continuos de datos 32

3.1. FEPDS: Fuzzy Emerging Patterns from Data Streams 33

3.1.1. Componente de memoria 34

3.1.2. Proceso de adaptación del aprendizaje al cambio de concepto . 35

3.1.3. Enfoque de aprendizaje evolutivo 37

3.1.3.1. Representación del conocimiento 38

3.1.3.2. Operadores genéticos 39

3.1.3.3. Proceso de recompensa en token competition 40

3.1.4. Esquema operacional de FEPDS 42

3.2. Estudio experimental 45

3.2.1. Marco experimental 45

3.2.2. Análisis de la adaptación al cambio de concepto 47

3.2.3. Análisis de la calidad del conocimiento extraído 51

4. Análisis de patrones para la caracterización de perfiles de clientes en los *yellow taxis* de Nueva York 54

4.1. Marco experimental 55

4.2. Extracción de patrones emergentes por FEPDS en el flujo de datos de los *yellow taxis* de Nueva York 55

5. Conclusiones y publicaciones relacionadas 60

5.1. Conclusiones 60

5.2. Publicaciones relacionadas 62

5.3. Trabajos futuros 63

APÉNDICES 65

A. Publicación sometida a IEEE Transactions on Fuzzy Systems relativa al algoritmo FEPDS 65

B. Tabla de resultados relativa al estudio experimental 76

Bibliografía 80

Índice de figuras

2.1. Tipos principales de cambio de concepto 10
2.2. Relación entre los diferentes tipos de patrones emergentes. 23

3.1. Proceso de recolección de datos y transformación en bloques del algoritmo FEPDS. 34
3.2. Esquema general de funcionamiento del algoritmo FEPDS. 36
3.3. Representación de una variable numérica mediante cinco etiquetas lingüísticas triangulares uniformes. 38
3.4. Representación de un patrón difuso en forma normal disyuntiva con variables categóricas y continuas en FEPDS. 39
3.5. Confianza y tiempo de ejecución del algoritmo FEPDS 48
3.5. Confianza y tiempo de ejecución del algoritmo FEPDS 49
3.5. Confianza y tiempo de ejecución del algoritmo FEPDS 50

Índice de tablas

2.1. Tabla de contingencia de un patrón. 21
2.2. Algoritmos para minería de patrones emergentes 28

3.1. Resultados promedio de FEPDS. 51

4.1. Descripción de las variables utilizadas en el conjunto de datos de los taxis de Nueva York. 56
4.2. Resultados promedio de FEPDS en el flujo de datos de los taxis de Nueva York. 56
4.3. Patrones más repetidos para los taxis de Nueva York 58

CAPÍTULO 1

INTRODUCCIÓN

Actualmente vivimos inmersos en la época de la llamada “revolución de los datos”. La llamamos así porque se generan cantidades ingentes de datos de forma continua. Este hecho se debe principalmente al abaratamiento de sensores, dispositivos de almacenamiento y al conjunto de técnicas informáticas desarrolladas que permiten un almacenamiento mucho más rápido y eficiente. Estos datos contienen implícito conocimiento útil que permitirían mejorar la calidad de vida de las personas, procedimientos en empresas o incluso servicios cotidianos. El problema que subyace bajo esta gran cantidad de datos generados es que se hace imposible una extracción manual de este conocimiento implícito. Actualmente, esta extracción es llevada a cabo de manera eficaz, eficiente y automática mediante el proceso de extracción de conocimiento en grandes bases de datos, conocido como *Knowledge Discovery in Databases* (KDD) (Fayyad et al., 1996b).

KDD puede definirse como el proceso no trivial de identificar patrones en los datos que sean: válidos, novedosos, útiles y comprensibles. El proceso de KDD es un conjunto de pasos interactivos e iterativos, entre los que se incluye el preprocesamiento de los datos para corregir posibles datos erróneos, incompletos o inconsistentes, la reducción del número de registros o características encontrando los más representativos, la búsqueda de patrones de interés con una representación particular y la interpretación de estos patrones incluso de una forma visual.

La metodología KDD se puede dividir en distintas etapas de entre las que destaca la minería de datos. La minería de datos trata de extraer conocimiento que no sea trivial, normalmente desconocido y potencialmente útil para el usuario a partir de grandes cantidades de datos (Fayyad et al., 1996a). Esta extracción de conocimiento se ha llevado a cabo tradicionalmente mediante dos enfoques claramente diferenciados:

- Un enfoque predictivo, en el que se pretende predecir el valor de una clase prefijada en nuevas instancias. Para este tipo de enfoque se utiliza normalmente un aprendizaje supervisado, en donde se entrena un modelo con datos previamente etiquetados por expertos.

- Un enfoque descriptivo en el que se busca realizar resúmenes de los datos o bien buscar relaciones entre los ejemplos de un conjunto de datos. Al contrario que el enfoque predictivo, este tipo de minería emplea por regla general aprendizaje no supervisado, pues no necesita datos etiquetados para encontrar estas relaciones.

Actualmente estamos rodeados de dispositivos que están continuamente generando datos. A este tipo de datos se les conoce como flujo de datos (Gama, 2010). Ejemplos representativos pueden ser redes de sensores, la gestión de una planta de generación de energía eléctrica o las redes de comunicaciones, entre otros (Žliobaitė et al., 2016). En este tipo de aplicaciones, una de las principales características que nos encontramos es que los datos pierden relevancia a medida que pasa el tiempo. Por lo tanto, un proceso clásico de KDD no sería adecuado ya que trataría a todos los datos con el mismo peso. Para solucionar esto, un análisis continuo de los datos a medida que estos llegan al sistema es más interesante que un enfoque tradicional.

Sin embargo, la minería de flujos de datos nos impone una serie de restricciones que son un desafío a la hora de desarrollar nuevos métodos. Entre otras restricciones, se destaca la continua adaptación y actualización del modelo aprendido conforme van llegando los datos para evitar la obsolescencia del mismo. Además, la velocidad de llegada de los datos se supone que es muy rápida, por lo tanto, esta actualización debe realizarse lo más rápido posible.

La minería de patrones emergentes (EPM) (Dong y Li, 1999; García-Vico et al., 2018) es una técnica de minería de datos englobada dentro del marco denominado "descubrimiento de reglas descriptivas mediante aprendizaje supervisado" (SDRD) (Kralj-Novak et al., 2009). EPM extrae conocimiento que permite la descripción de características discriminatorias entre los diferentes valores de una variable objetivo o la descripción de tendencias emergentes en los datos. Este tipo de técnica de minería de datos ha sido aplicada con éxito en diferentes campos como por ejemplo en administración (Li et al., 2015) o detección de enfermedades (Yu et al., 2014), entre otros. De hecho, varios enfoques basados en sistemas difusos evolutivos (EFSs) (Herrera, 2008) han sido propuestos en (García-Vico et al., 2018, 2017) en donde el balance entre la generalidad de las descripciones y la fiabilidad de las mismas ha sido mejorada con respecto a otros enfoques desarrollados (García-Vico et al., 2018), produciendo unos modelos más sencillos y fiables. Esta simplicidad, unida a una mayor fiabilidad en las descripciones, hace que EPM sea una técnica muy relevante dentro de la minería de flujo de datos ya que es necesario una descripción sencilla y precisa de lo que está ocurriendo en todo momento con el objetivo de tomar decisiones rápidamente. Sin embargo, a día de hoy la extracción de patrones emergentes (EPs) en flujos de datos no ha sido una tarea muy explotada en la literatura científica.

1.1. Objetivos

Actualmente existe una gran variedad de algoritmos y aplicaciones abordados con técnicas dentro de SDRD, más específicamente con patrones emergentes. Sin embargo, el principal problema es la extracción de EPs descriptivos de alta calidad que sean capaces de cumplir con las restricciones que nos encontramos en la tarea de minería de flujos de datos. Esto se debe a un espacio de búsqueda no convexo que crece de manera de exponencial respecto al número de variables (Wang et al., 2004a).

El objetivo principal de este trabajo es el desarrollo de un nuevo algoritmo que sea capaz de obtener un modelo simple y fiable del flujo de datos. Para ello, el algoritmo deberá ser capaz de adaptarse a los datos y a las tendencias cambiantes en los mismos.

Para alcanzar este objetivo principal, se definen los siguientes objetivos particulares:

1. Realizar una revisión del estado del arte de la literatura dentro del ámbito de la minería de patrones emergentes y de la minería de flujo de datos en general. Este análisis sentará las bases de la temática, la dificultad de la misma y la manera de abordar el problema por los investigadores.

2. Realizar un estudio exhaustivo de los modelos existentes para la minería de patrones emergentes. Este estudio servirá para determinar las características y estrategias a utilizar para el desarrollo del nuevo modelo.

3. Desarrollar un nuevo algoritmo capaz de extraer continuamente patrones emergentes de calidad adaptándose a los datos que van llegando a lo largo del tiempo.

4. Analizar el comportamiento de la nueva propuesta respecto a los diferentes tipos de cambios de concepto en los flujos de datos para determinar su capacidad de adaptación. Asimismo, se realizará un estudio para determinar la calidad del conocimiento extraído por la propuesta.

5. Aplicar minería de patrones emergentes sobre un flujo de datos real para demostrar las bondades descriptivas de los modelos de minería de patrones emergentes en el contexto de minería de flujo de datos.

1.2. Estructura de la memoria

La estructura de esta memoria de trabajo estará dividida en los siguientes capítulos:

- En el capítulo 2 se muestra una revisión del estado del arte de los conceptos a trabajar en este proyecto: minería de flujo de datos y minería de patrones emergentes.
- En el capítulo 3 se muestran en profundidad los detalles y características del nuevo modelo desarrollado. Además, se presenta un estudio experimental en donde se determina la adaptabilidad de la propuesta a diferentes tipos de cambios de concepto que se pueden dar en los flujos de datos para determinar la adaptabilidad del mismo. Asimismo, en este capítulo se determina la calidad del conocimiento extraído por la propuesta.
- En el capítulo 4 se expone una aplicación del método propuesto en un flujo de datos real, en donde se muestran y analizan los resultados obtenidos.
- Finalmente, en el capítulo 5 se exponen las conclusiones de esta memoria, así como las líneas de trabajo futuro a desarrollar. También se exponen diferentes trabajos publicados en los que se describen parte de los resultados del trabajo realizado en esta memoria.

CAPÍTULO 2

Revisión del estado del arte

En este capítulo se presentará una revisión de los conceptos fundamentales que se trabajarán a lo largo de este trabajo y que nos proporcionarán una base sólida de conocimiento del problema. En un primer lugar, se definirá el problema de la minería de flujo de datos, junto un estudio en profundidad de las diferentes estrategias utilizadas para abordarla. A continuación, se definirá el problema de la minería de patrones emergentes. En este apartado se expondrán las diferentes medidas de calidad utilizadas para validar el conocimiento extraído junto a una revisión de la literatura.

2.1. Minería de flujo de datos

Antiguamente la única entrada de datos que se podía realizar en un equipo informático era mediante un empleado, el cual introducía uno por uno los diferentes datos de interés. A día de hoy, los equipos informáticos no solo son capaces de registrar datos de muy diversa índole, sino que también son capaces de transmitirlos y almacenarlos directamente en otro equipo sin ninguna intervención humana. De hecho, la cantidad de dispositivos existentes provoca que a día de hoy se generen continuamente cantidades inmensas de información. Por ejemplo, en solo un segundo se publican más de ocho mil *tweets*, más de mil fotos son subidas a *Instagram*, se realizan más de

setenta y cinco mil búsquedas en *Google* o se envían casi 3 millones de *e-mails*, entre otros[1]. Toda esta ingente cantidad de información contiene conocimiento que es útil. No obstante, la naturaleza de estos datos suele ser efímera, por lo que rápidamente pierden interés. Es por esta razón que dichos datos deberían ser continuamente analizados. A toda esta información que se genera continuamente se le denomina flujo de datos (Gama, 2010). En este apartado se define que es un flujo de datos, que tipos de flujos de datos se pueden encontrar y, finalmente, la descripción del problema del cambio de concepto y cómo abordar flujos de datos dinámicos.

2.1.1 Definición del problema

Un flujo de datos se define como una secuencia potencialmente infinita y ordenada de datos que llegan al sistema a lo largo del tiempo (Gama, 2010). El tiempo de llegada entre un dato y el siguiente no es fijo y puede variar a lo largo del tiempo. Asimismo, estos datos pueden ser simples pares atributo-valor, como una tupla, o estructuras más complejas, como por ejemplo un grafo.

A pesar de la definición tan simple de flujo de datos, ésta esconde problemas complejos de resolver desde el punto de vista de la minería de datos tradicional. En concreto, las diferencias más relevantes entre la minería de datos tradicional y la minería de flujo de datos son (Bifet, 2009; Gama, 2010):

- No hay control sobre el orden en el que los datos llegan y se procesan en el sistema. Además, los datos pueden llegar en cualquier momento, por lo que el sistema debe ser capaz de reaccionar instantáneamente.
- Al ser una secuencia de datos potencialmente infinita, se asume que es imposible el almacenamiento de todo el flujo de datos en memoria.
- Unido al punto anterior, normalmente se permite una única visualización del dato que llega para su procesamiento. Una vez se procesa, éste se descarta o se almacena de manera temporal.
- Se asume que la velocidad de llegada de los datos es alta con respecto a la capacidad de procesamiento del sistema. Por lo tanto, se debe dar una respuesta rápida con el fin de evitar encolamientos de datos.

[1] https://www.internetlivestats.com/one-second

- Los flujos de datos son susceptibles a cambios en la distribución subyacente que los genera. Esto provoca que el modelo actual, aprendido sobre una distribución diferente, falle. Por lo tanto, es necesario el desarrollo de estrategias que permitan una rápida adaptación del modelo cuando se produzcan estos cambios.

Este tipo de restricciones impide que la mayoría de algoritmos de minería de datos tradicional sean capaces de afrontar el problema de la minería de flujo de datos ya que una de sus principales asunciones es que el tamaño del problema es finito. Asimismo, no son capaces de cumplir otras restricciones como cantidad de memoria, rápida respuesta o escaneo único de las instancias (Brzeziński, 2015). No obstante, es importante destacar que existen métodos de minería de datos tradicional, como Naïve Bayes o las redes neuronales, que tienen una naturaleza incremental. A pesar de esta clara ventaja, este tipo de algoritmos tampoco es capaz de cumplir con las restricciones computacionales de los flujos de datos ni tampoco tienen en cuenta la naturaleza cambiante de las fuentes de datos (Gama, 2010).

Finalmente, los ejemplos (o instancias) del flujo de datos pueden llegar al sistema de dos maneras diferentes:

- Online. Los ejemplos van llegando al sistema uno a uno, y estos son procesados conforme llegan.
- Bloques o *chunks*. Los datos se encuentran disponibles en conjuntos de varios ejemplos. Normalmente el tamaño de estos bloques es relativamente grande y no suele variar su tamaño. Todo el proceso de construcción, actualización o evaluación del modelo se lleva a cabo cuando todos los ejemplos de un bloque están disponibles.

2.1.2 Flujos de datos dinámicos y cambio de concepto

Actualmente existen dos modelos básicos de flujo de datos: un modelo estacionario, donde los datos son generados siguiendo una distribución de probabilidad normalmente desconocida; y modelos dinámicos o no-estacionarios en donde esta distribución de probabilidad cambia a lo largo del tiempo. Esto implica que, dado dos puntos

de tiempo i y j, la distribución subyacente que los generaba puede ser distinta, es decir, $D_i \neq D_j$ (Khamassi et al., 2018). Este cambio supone un auténtico reto en el aprendizaje de modelos de minería de datos ya que estos generalizan dicha distribución D. Si esa distribución cambia, el modelo falla por completo.

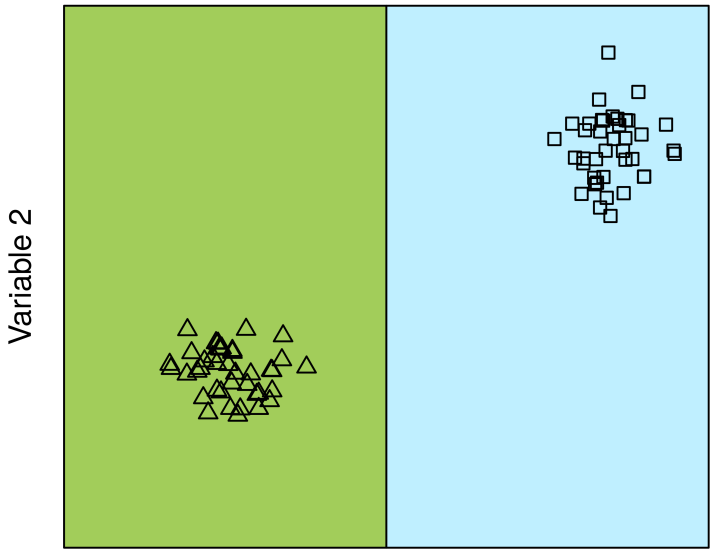
Este tipo de flujos de datos dinámicos son muy comunes en aplicaciones reales. Ejemplos de ello podrían ser la detección de spam y fraude, posicionamiento de objetos, robótica, etc. (Žliobaitė et al., 2016).

Formalmente (Gama, 2010; Webb et al., 2016), un ejemplo o instancia es generado por la fuente de datos en un tiempo t siguiendo una probabilidad conjunta $P^t(X, y)$, donde X son los valores que forman el ejemplo e y es la clase. Un concepto es estacionario si todos los ejemplos son generados con la misma distribución de probabilidad. Sin embargo, si para dos puntos de tiempo t y $t + \delta$ existe un X tal que $P^t(X, y) \neq P^{t+\delta}(X, y)$, entonces se ha producido un cambio de concepto.

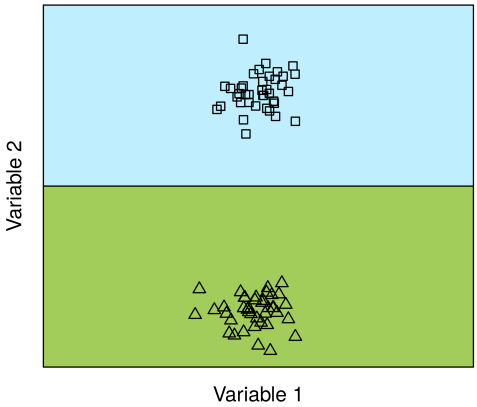
Cuando un cambio de concepto se produce, varios componentes de $P^t(X, y)$ pueden cambiar. En concreto se pueden dar cambios en la probabilidad a priori de las clases, es decir, $P(y)$ o en la probabilidad condicionada de la clase $P(X|y)$. Con esta clasificación se pueden producir dos tipos de cambios de concepto: un cambio de concepto real, y un cambio de concepto virtual (Gama et al., 2014), presentados gráficamente en la Figura 2.1:

- Cambio de concepto real. Este tipo de cambio se debe a cambios producidos en la probabilidad a posteriori $P(y|X)$, lo que significa que la clase para una instancia con los mismos valores cambia. Por lo tanto, este tipo de cambio afecta directamente a las fronteras de decisión del modelo de aprendizaje, por lo que éste decrementa su rendimiento. El tratamiento de este tipo de cambio de concepto ha sido el más estudiado a lo largo de la literatura. Un ejemplo de como sería este tipo de cambio se presenta en la Figura 2.1b.

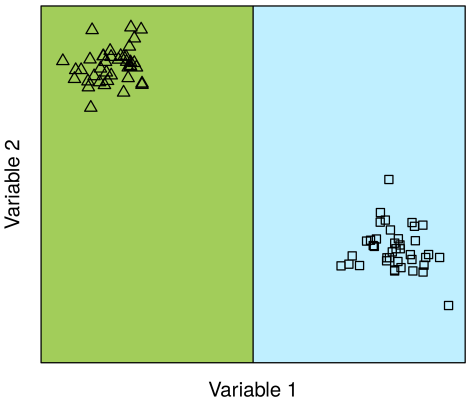
- Cambio de concepto virtual. Este tipo de cambio de concepto es más sutil que el anterior y se debe a cambios en la probabilidad condicionada de la clase $P(X|y)$ sin afectar a la probabilidad a posteriori $P(y|X)$. En este sentido, la distribución que genera los datos ha cambiado sin afectar a las fronteras de decisión, por lo



(a) Datos Originales



(b) Cambio real



(c) Cambio virtual

Figura 2.1: Tipos principales de cambio de concepto en función de la influencia en las fronteras de decisión del modelo.

que los modelos no pierden rendimiento. Sin embargo, es interesante desde un punto de vista descriptivo detectar este tipo de cambios para informar y describir dicha tendencia de cambio. Un ejemplo de como sería este tipo de cambio se presenta en la Figura 2.1c.

Cada uno de estos tipos se pueden dividir a su vez en diferentes subcategorías en función de la velocidad a la que se produce el cambio, la severidad y la recurrencia del mismo (Khamassi et al., 2018):

- Velocidad del cambio. Es la duración del cambio de concepto en función del número de pasos temporales en los que un nuevo concepto reemplaza a otro. En este aspecto, se puedan dar los siguientes tipos de cambio:
  - Cambio abrupto. Ocurre cuando la nueva distribución reemplaza inmediatamente a la antigua.
  - Cambio gradual. Ocurre cuando la transición de un cambio a otro es más lenta. Se pueden dar dos posibilidades de cambio: un cambio gradual probabilístico y un cambio gradual continuo. En el primero, existen dos o más distribuciones que tienen probabilidad de producir ejemplos. Estas probabilidades cambian de modo que conforme pasa el tiempo hay mayor probabilidad de obtener instancias de la nueva distribución. En el segundo, el concepto va cambiando mediante pequeñas modificaciones que sólo son notables en un largo período de tiempo.
- Severidad. Se refiere a la cantidad de instancias afectadas por el cambio. Este tipo de cambios se clasifican en cambios locales y globales. Los cambios locales son los que se producen en algunas regiones del espacio, por lo que sólo un subconjunto de instancias son afectadas. En contraposición, en un cambio global todo el espacio de búsqueda se ve afectado.
- Recurrencia. Se refiere a la capacidad de reaparición de nuevos conceptos a lo largo del tiempo. Este tipo de cambios se clasifican en cambios recurrentes o acíclicos. Un cambio recurrente se produce con respecto una tendencia estacional o cada cierto tiempo, mientras que para un acíclico no se conoce con exactitud cuando el concepto puede volver a reaparecer.

2.1.3 Diseño de métodos con manejo del cambio de concepto

Los flujos de datos dinámicos son los más comunes en problemas reales. Por esta razón, es interesante el desarrollo de métodos que sean capaces de extraer conocimiento de estos flujos de datos pudiendo adaptarse a las características dinámicas de los mismos. Con todas estas características, las preguntas más comunes que se deben responder para el correcto desarrollo de métodos de aprendizaje para minería de flujo de datos son las siguientes (Khamassi et al., 2018):

1. ¿Cómo se deben procesar los datos en la aplicación?
2. ¿Cómo se lleva a cabo la tarea de aprendizaje?
3. ¿Cómo se monitoriza el cambio de concepto?
4. ¿Cómo adaptar el aprendizaje al cambio?
5. ¿Cómo evaluar la calidad del modelo?

2.1.3.1. Procesamiento de datos

Debido a la naturaleza cambiante de los flujos de datos, es fundamental procesarlos de una manera que nos permita una mejor adaptación al cambio. Para ello, la estrategia que más se ha seguido en la literatura es considerar que los datos más recientes son los más representativos y, por tanto, tienen un mayor peso a la hora de realizar el aprendizaje en el instante actual. Estos datos se pueden procesar de manera secuencial, es decir, uno a uno, o bien mediante el uso de ventanas temporales, es decir, bloques de múltiples instancias.

En el procesamiento de datos secuencial los datos son procesados uno a uno por el método y una vez usados son descartados. Se basa en el proceso estadístico de detección de anomalías (Wald, 1973). Básicamente, este tipo de procesamiento intenta detectar si hay un cambio significativo entre dos distribuciones D_0 y D_1 en el momento de tiempo t. Para determinar si un cambio es significativo, se comprueba si una medida de desigualdad $Diss_t(D_0, D_1)$ entre ambas distribuciones supera cierto umbral τ.

Como se puede observar, dos parámetros son fundamentales para el procesamiento secuencial: la medida de desigualdad, y el umbral τ. En cuanto a medidas de desigualdad, se han usado en la literatura muchas medidas que trabajan directamente sobre los datos, como por ejemplo la distancia Euclídea (Tran, 2019), la distancia de Mahalanobis (Gonçalves et al., 2014; Toubakh y Sayed-Mouchaweh, 2015) o la de Hellinger (Ditzler y Polikar, 2011), entre otras.

Por otro lado, el procesamiento basado en ventanas temporales ha sido ampliamente utilizado en la literatura debido a su flexibilidad y bajo consumo de memoria. Una ventana temporal se define como una memoria que almacena datos o resúmenes de los mismos que corresponden con el concepto actual. Estas ventanas pueden ser utilizadas posteriormente por el algoritmo de aprendizaje. Las características de estas ventanas se definen de acuerdo a cuatro dimensiones:

- **Ámbito**. Las ventanas temporales pueden ser utilizadas dentro del algoritmo de aprendizaje (Hulten et al., 2001), o fuera del mismo como un elemento genérico para cualquier algoritmo de aprendizaje (Khamassi et al., 2015).
- **Naturaleza**. El tamaño de las ventanas temporales viene definido o bien por el número de ejemplos (Babcock et al., 2002) o bien se define por una duración temporal (Babcock et al., 2002)
- **Tamaño**. El tamaño de una ventana temporal puede ser fijo (Widmer y Kubat, 1996) a lo largo del tiempo, o variable.
- **Estrategia de posicionamiento**. Se refiere a como evoluciona la posición de la ventana temporal durante el seguimiento del cambio y se han utilizado en la literatura una o dos ventanas:
  - **Una ventana**. Dentro de este grupo tenemos los siguientes tipos:
    - ***Sliding window***. Los datos más recientes son almacenados en una cola *first-in-first-out* (FIFO) y el sistema va aprendiendo con los datos que existen en ella (Hulten et al., 2001).

- ***Landmark window***. En esta estructura se almacenan los datos desde un periodo determinado hasta que ocurre cierto evento. Por ejemplo, se almacenan datos hasta que se detecta un cambio de concepto (Gama y Castillo, 2006).

- **Dos ventanas**. La idea principal del uso de dos ventanas es mantener una de ellas como ventana de referencia y otra que contenga los datos actuales. Dentro de este grupo tenemos los siguientes tipos:
  - **Separadas**. En este tipo, la ventana de referencia y la actual divergen a lo largo del tiempo. Esta estrategia es interesante para el uso de métodos *off-line*, en donde el algoritmo aprende inicialmente con la ventana de referencia y no vuelve a ser ejecutado hasta que se detecte un cambio de concepto entre la ventana actual y la de referencia (Bach y Maloof, 2008).
  - **Adyacentes**. Ambas ventanas se encuentran unidas durante el procesamiento. Generalmente, en este tipo de ventanas la comparación entre ellas se realiza de manera secuencial, por lo que sólo se pueden aplicar a modelos de aprendizaje secuenciales. Se pueden dar tres subtipos en función de como evoluciona el tamaño de estas ventanas a lo largo del tiempo: fijo/fijo (Kifer et al., 2004), variable/fijo (Khamassi y Sayed-Mouchaweh, 2014) y variable/variable (Bifet y Gavalda, 2007).
  - **Solapadas**. La ventana de referencia y la actual comparten datos. Esto puede ser interesante cuando existen pocos datos o el flujo de datos está desbalanceado. También es interesante su uso en *ensembles* al tener intensificada una zona en varios métodos (Tran, 2019).
  - **Discontinuas**. Sólo se usa un subconjunto de la ventana de referencia para comparar con la ventana actual. Estos subconjuntos se escogen en función de ciertos criterios y son útiles a la hora de detectar cambios locales (Khamassi et al., 2015).

2.1.3.2. Proceso de aprendizaje

Una vez establecido como se van a procesar los datos, es necesario establecer el método de aprendizaje por el que se va a extraer conocimiento, pues depende del tipo de procesamiento de datos usado en el apartado anterior. En este sentido, la clasificación se establece en el uso de un único algoritmo de aprendizaje o en el uso de varios formando *ensembles*.

El uso de un único algoritmo de aprendizaje es idóneo en aquellos escenarios donde se tiene un método incremental y se requiere un tiempo de respuesta rápido, pues son método simples. No obstante, se han desarrollado extensiones de métodos incrementales que incorporan mecanismos de olvido con el objetivo de descartar conceptos obsoletos (Cauwenberghs y Poggio, 2001). La idea es reducir el peso de aquellas instancias más antiguas ya que es más probable que contengan información obsoleta.

Por otro lado, el uso de *ensembles* es más adecuado cuando tenemos entornos muy cambiantes ya que tienen unas mayores capacidades de generalización que un único algoritmo. Sin embargo, la complejidad del sistema aumenta considerablemente ya que se tienen que tener en cuenta conceptos como la diversidad de los métodos de aprendizaje y cómo éstos se adaptan a los cambios (Krawczyk et al., 2017).

2.1.3.3. Proceso de monitorización

Un método de monitorización del cambio de concepto nos va a permitir identificar cuando y dónde ha ocurrido un cambio. Sin embargo, este tipo de proceso está muy influenciado por la disponibilidad de la etiqueta de clase. Si ésta está disponible inmediatamente, se pueden usar métodos basados en indicadores supervisados; sino, otras técnicas como indicadores no supervisados pueden ser utilizadas:

- **Métodos basados en indicadores supervisados**. Se utilizan cuando se tiene *feedback* inmediato y se basan en el mantenimiento del rendimiento en el algoritmo de aprendizaje. Por lo tanto, son buenos adaptándose a cambios de concepto reales. El principal indicador que se utiliza en este caso es la precisión del modelo (Bifet y Gavalda, 2007; Gama y Castillo, 2006). Sin embargo, otros indicadores como el *recall*, la confianza o la sensitividad y la especificidad pueden ser interesantes en otros escenarios, como por ejemplo en datos desbalanceados.
- **Métodos basados en indicadores no supervisados**. Este tipo de métodos es más común en aplicaciones reales, pues normalmente no se tiene *feedback* inmediato. Además, pueden ser interesantes para detectar cambios de concepto virtuales. Estos métodos se basan o bien en buscar similitudes en los datos tanto temporales (Shaker y Lughofer, 2014) como espaciales (Toubakh y Sayed-Mouchaweh, 2015), o bien en la monitorización de la complejidad de algunos modelos (Cauwenberghs y Poggio, 2001).

2.1.3.4. Proceso de adaptación

El objetivo de la detección y aplicación de mecanismos en el cambio de concepto es para determinar como el algoritmo de aprendizaje se adapta a estos cambios. Para adaptarse al cambio de concepto, en la literatura se han utilizado principalmente dos enfoques. Un enfoque ciego y un enfoque informado:

- **Enfoque ciego**. Este tipo de enfoque adapta el método de aprendizaje al concepto actual en intervalos regulares sin utilizar para ello ningún método de detección de cambio. Estos métodos pueden ser interesantes cuando se producen cambios de concepto graduales. Algunas de las estrategias más utilizadas para una adaptación ciega son el uso de ventanas deslizantes de tamaño fijo (Muthukrishnan et al., 2007), la ponderación de instancias (Krawczyk y Woźniak, 2015) o incluso el empleo de *ensembles* en donde el peor método es remplazado por uno nuevo (Kuncheva, 2004). Este tipo de estrategias es adecuada para la adaptación a los cambios de tipo gradual, pues los ligeros cambios que se producen son fácilmente adaptados por el modelo al estar constantemente adaptándose. Sin embargo, la re-adaptación en un momento donde no es necesario implica un coste computacional evitable.

- **Enfoque informado**. Este tipo de enfoque utiliza un método de monitorización para comprobar el estado del modelo y/o del flujo de datos para determinar si ha habido cambios de concepto. En caso de que se haya producido, se lanza una señal de alerta en donde el método es adaptado a los nuevos datos. Estos enfoques son interesantes cuando es necesario identificar dónde se ha producido el cambio.

2.1.3.5. Proceso de evaluación

Una vez se ha determinado la estructura del modelo, es necesario el desarrollo de una estrategia de evaluación con el objetivo de determinar la calidad del modelo. En la minería de datos tradicional, esta calidad se ha determinado mediante el empleo de diversos mecanismos como la validación cruzada, *hold-out*, entre otros (Japkowicz y Shah, 2011). Sin embargo, estos mecanismos requieren de un conjunto de datos finito ya que es necesario la utilización de particiones de entrenamiento y prueba. Como se ha comentado anteriormente, los flujos de datos son potencialmente infinitos, por lo que la aplicación de este tipo de técnicas tradicionales no es posible. Es por eso que es necesario evaluar el modelo tan pronto como los datos sean procesados. Una de las técnicas más populares para la evaluación de la calidad de los modelos es el método *test-then-train* (Wald, 1973).

Este método primero realiza una predicción del nuevo dato que llega al sistema empleando el modelo actual. Después, y una vez se conoce el verdadero valor de esta instancia, el modelo es evaluado con la medida de calidad correspondiente. Una vez ocurre esto, la instancia que ya ha sido utilizada para evaluar el modelo pasa a actualizarlo, es decir, pasa a ser un dato de entrenamiento. Por medio de esta técnica de evaluación es posible estar continuamente evaluando el modelo a la vez que se va actualizando conforme los datos llegan al sistema.

Asimismo, es interesante monitorizar otros factores que determinan el rendimiento del sistema, como por ejemplo el tiempo de ejecución en la actualización del modelo o el consumo de memoria del método a lo largo del tiempo.

2.2. Minería de patrones emergentes

La minería de patrones emergentes (EPM) (Dong y Li, 1999; García-Vico et al., 2018) es una tarea de minería de datos que pertenece al marco SDRD (Kralj-Novak et al., 2009). El objetivo de EPM es la búsqueda de patrones que contienen un cambio significativo en el soporte entre dos conjuntos de datos, o de una clase dada al resto de clases del problema en un único conjunto de datos.

Formalmente, un patrón se define como: sea $V = \{v_1, v_2, ..., v_n\}$ las variables de un problema. Estas pueden ser de tipo numérico o categórico. Una de estas variables será la variable de interés del problema o clase, y será denominada como v_c. En concreto para este trabajo, se asume que v_c es siempre de tipo categórico. Sea $X_i = \{x_{i1}, x_{i2}, ...x_{in}\}$ el conjunto de los diferentes valores o categorías para una variable categórica V_i, el dominio si V_i es numérica o un conjunto de etiquetas lingüísticas (LLs) (Zadeh, 1975) para la representación de variables numéricas mediante lógica difusa. Un ejemplo es definido como $E = \{(v_1, x_{1j}), (v_2, x_{2j}), ..., (v_n, x_{nj})\}$. Un dataset D es definido como un conjunto de ejemplos. Un selector (Michalski y Stepp, 1982) es definido como una tupla (v_i, x_{ij}) que conecta una variable con sus valores. Por lo tanto $v_i \in V$ y $x_{ij} \in X_i$.

Sea $I = \{i_1, i_2, ..., i_n\}$ el conjunto de todos los posibles selectores del problema dado. Un patrón P es un subconjunto de I. Habitualmente, los selectores pertenecientes a P están unidos mediante conjunciones o en forma normal disyuntiva. Para facilitar la comprensión del trabajo, se indica que un patrón P cubre a un ejemplo E, o que un ejemplo E contiene a un patrón P si y solo si todas los elementos de P son satisfechos por E. Finalmente, P se denominará como patrón emergente si su índice de crecimiento (GR) es mayor que un umbral dado $\rho > 1$. El GR es definido en la Ecuación 2.1.

$$GR(X) = \begin{cases} 0, & \text{Si } Sop_1(X) = Sop_2(X) = 0, \\ \infty, & \text{Si } Sop_1(X) = 0 \wedge Sop_2(X) \neq 0, \\ \frac{Sop_2(x)}{Sop_1(x)}, & \text{en otro caso} \end{cases} \tag{2.1}$$

donde $Sop_i(X) = \frac{count(X)}{|D_i|}$ es el soporte de dicho patrón en el conjunto de datos i, $count(X)$ es el número de instancias que verifican el patrón y $|D_i|$ es el número de instancias total del conjunto de datos D_i.

Los principales objetivos de la EPM son (Dong y Li, 1999):

- Detectar tendencias emergentes.
- Detectar diferencias características entre clases en un conjunto de datos.
- Detectar diferencias entre múltiples variables.

Un ejemplo de patrones emergentes puede ser el obtenido en el conjunto de datos *Mushroom* del repositorio UCI (Dheeru y Karra˜Taniskidou, 2017), el cual contiene dos clases (*edible* y *poisonous*). Este conjunto de datos corresponde a la identificación de setas comestibles o venenosas. En este conjunto de datos se encontraron gran cantidad de patrones emergentes. Dos de estos patrones fueron los siguientes (Dong y Li, 1999):

$$X = \{(ODOR = none), (GILL\_SIZE = broad), (RING\_NUMBER = one)\}$$

$$Y = \{(BRUISES = no), (GILL\_SPACING = close), (VEIL\_COLOR = white)\}$$

En concreto, el patrón X corresponde con aquellas setas que no tienen olor, con un tamaño de lámina ancho y un número de anillos igual a uno. Por otro lado, el patrón Y corresponde con las setas cuyo micelo no cambia de color tras arrancarse la seta del suelo, con una separación entre láminas estrecha y con un color de velo blanco. Estos patrones obtuvieron los siguientes índices de crecimiento:

EP	***poisonous***	***edible***	***GR***
X	0 %	63.9 %	∞
Y	81.4 %	3.8 %	21.4

En este ejemplo, el patrón Y sería un patrón emergente de *edible* a *poisonous* con índice de crecimiento de 21.4 y el patrón X sería un patrón emergente de *poisonous* a *edible* con índice de crecimiento infinito. Esto quiere decir que, las setas cuyas características concuerdan con las del patrón Y, por cada seta comestible que concuerde

con Y, habrá 21.4 setas venenosas que tengan las mismas características. Por lo tanto la probabilidad de comer una seta venenosa es muy elevada. Por el contrario, todas las setas que concuerden con las características del patrón X serán comestibles Es importante destacar el gran poder discriminativo que poseen estos patrones, así como su facilidad de comprensión al poseer un bajo número de variables.

Un punto importante a destacar en la dificultad para la extracción eficiente de estos patrones, es que los patrones obtenidos no cumplen la propiedad Apriori (Agrawal et al., 1996) que tanto éxito ha tenido en la minería de patrones frecuentes, haciendo que el problema sea NP-Duro para un gran número de variables (Wang et al., 2004b). Siguiendo con el ejemplo anterior de (Dong y Li, 1999), el subpatrón $X = \{(ODOR = none)\}$ no es un patrón emergente en ese conjunto de datos, incumpliendo así la propiedad Apriori. Esto se debe a que el índice de crecimiento se basa en la proporción de cambio que hay en los soportes y no en el soporte actual de un patrón. Por lo que un patrón más específico (menor soporte) puede tener una proporción de cambio mayor que un patrón más general (mayor soporte).

Para facilitar la comprensión de estos patrones, en la literatura se suelen representar de la siguiente manera:

$$P : Cond \rightarrow Target_{value} \tag{2.2}$$

donde $Cond$ es un conjunto de selectores en conjunción o en forma normal disyuntiva y $Target_{value}$ es el valor de la variable objetivo, la cual será contrastada con el resto de valores de la variable objetivo para medir su índice de crecimiento.

2.2.1 Medidas de calidad en EPM

Para la obtención de un modelo de calidad en EPM es importante encontrar una buena relación entre la capacidad discriminativa de los EPs extraídos y su capacidad para explicar fácilmente el fenómeno subyacente en los datos a los expertos. Por lo tanto, EPM puede ser considerado como un problema multiobjetivo en donde varias medidas de calidad deben ser utilizadas para determinar los diferentes aspectos clave.

Para la determinación de estos objetivos, se emplean las medidas de calidad clásicas que se encuentran incluidas en el marco SDRD. Estas medidas pueden ser fácilmente calculadas por medio de una tabla de contingencias en donde se muestra el número de ejemplos correctamente/incorrectamente cubiertos/no cubiertos para cada patrón. Este tipo de tabla se presenta en la Tabla 2.1.

Tabla 2.1: Tabla de contingencia de un patrón.

	Clase	No Clase
Cubierto	tp	fp
No cubierto	fn	tn

En esta tabla se define tp como el número de verdaderos positivos, es decir, el número de ejemplos de la clase marcada por el patrón correctamente clasificados como clase positiva. fp es el número de falsos positivos, es decir, ejemplos incorrectamente cubiertos por el patrón. fn indica el número de falsos negativos, los cuales son ejemplos incorrectamente no cubiertos. Finalmente, tn es el número de verdaderos negativos, o ejemplos correctamente no cubiertos.

Las medidas de calidad más importantes en EPM son (García-Borroto et al., 2017):

- **Índice de crecimiento (GR).** Es la medida que define la EPM. Calcula el ratio de soporte entre dos clases o conjuntos de datos. Es una medida que nos permite calcular la fiabilidad o el poder discriminatorio de un patrón y se calcula como (Dong y Li, 1999):

$$GR(P) = \frac{tp\,(fp + tn)}{fp\,(tp + fn)} \tag{2.3}$$

- **Confianza (Conf).** Esta medida calcula la precisión del patrón con respecto a sus ejemplos cubiertos. Por lo tanto, es una medida que nos permite determinar la fiabilidad y precisión del patrón (Fayyad et al., 1996a):

$$Conf(P) = \frac{tp}{tp + fp} \tag{2.4}$$

- **Atipicidad.** Es una medida que estima el balance entre la confianza de la regla con respecto a su cobertura. Es decir, con esta medida se intenta buscar patrones precisos que cubran muchos ejemplos del problema. Por lo tanto, es una medida híbrida que nos permite detectar novedad, fiabilidad y cobertura (Carmona et al., 2018):

$$WRAcc(P) = \frac{tp + fp}{tp + fp + fn + tn}\left(\frac{tp}{tp + fp} - \frac{tp + fn}{tp + fn + fp + tn}\right) \tag{2.5}$$

  El dominio de este valor depende del porcentaje de ejemplos positivos de la clase. Por lo tanto, para realizar comparativas entre diferentes conjuntos de datos, es necesario realizar una normalización de esta medida (Carmona et al., 2018):

$$WRAcc_{Norm}(P) = \frac{WRAcc(P) - (1 - \%Pos)\,(0 - \%Pos)}{\%Pos\,(1 - \%Pos) - (1 - \%Pos)\,(0 - \%Pos)} \tag{2.6}$$

  donde $\%Pos = \frac{tp+fn}{tp+fp+tn+fn}$ es el porcentaje de ejemplos positivos del problema.

- **Tasa de verdaderos positivos (TPr).** Esta medida cuantifica el porcentaje de ejemplos correctamente cubiertos por el patrón con respecto al total de ejemplos positivos del problema. Es una medida que nos permite determinar la cobertura del patrón y se calcula como (Kloesgen, 1996):

$$TPr(P) = \frac{tp}{tp + fn} \tag{2.7}$$

- **Tasa de falsos positivos (FPr).** Calcula el porcentaje de ejemplos incorrectamente cubiertos con respecto al total de ejemplos negativos del problema. Esta medida nos permite calcular la fiabilidad del patrón y su capacidad discriminativa. Sin embargo, esta medida debe ser minimizada. Se calcula como (Gamberger y Lavrac, 2002):

$$FPr(P) = \frac{fp}{fp + tn} \tag{2.8}$$

- **Diferencia de soporte (SopDif).** Mide la diferencia de soporte entre la clase positiva y negativa de una regla. Esta medida, al igual que el GR, nos indica cómo de discriminitiva puede ser una regla teniendo en cuenta, además, la capacidad de cobertura de la misma (Bay y Pazzani, 1999).

$$SopDif(P) = \frac{tp}{tp + fn} - \frac{fp}{fp + tn} \tag{2.9}$$

2.2.2 Tipos de patrones emergentes

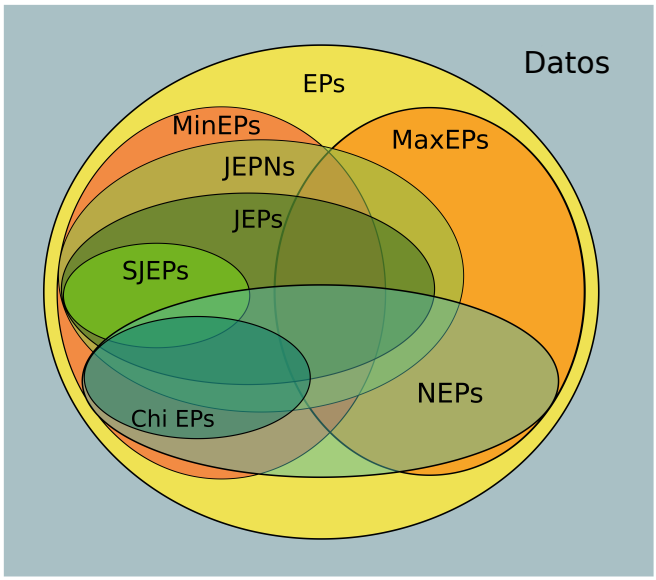


Figura 2.2: Relación entre los diferentes tipos de patrones emergentes.

La extracción de todos los posibles patrones emergentes de un problema dado es considerado un problema NP-Duro con respecto al número de variables (Wang et al., 2004a). Por esta razón, a lo largo de la literatura, los investigadores han intentado encontrar maneras de reducir el conjunto de patrones emergentes extraído con el objetivo de ser más eficiente y mejorar en calidad. En este sentido, la búsqueda de los investigadores se orientan a la extracción de aquellos patrones cuyas características describan ciertas relaciones específicas. De este modo, un conjunto mucho más simple de patrones emergentes es extraído de manera eficiente mejorando además la calidad de los modelos debido a la eliminación de ruido.

En la figura 2.2 se muestra gráficamente el conjunto de los tipos de patrones emergentes más importantes a lo largo de la literatura. A continuación, se presenta una breve definición de los mismos (García-Vico et al., 2018):

- ***Emerging Patterns* (EPs).** Son los patrones emergentes clásicos y engloban al resto de subconjuntos que se mencionarán. Se definen como los patrones con un índice de crecimiento (Ecuación 2.1) mayor a un umbral ρ (Dong y Li, 1999).

$$EPs = \{P | GR(P) \geq \rho\} \tag{2.10}$$

- ***Jumping Emerging Patterns* (JEPs).** Son patrones emergentes cuyo índice de crecimiento es igual a infinito. Esto quiere decir que los patrones obtenidos definen únicamente a una clase, por lo que poseen un gran poder discriminativo entre clases, permitiendo la obtención de potentes clasificadores (Dong et al., 1999a).

$$JEPs = \{P | GR(P) = \infty\} \tag{2.11}$$

- ***Essential Jumping Emerging Patterns* (eJEPs).** También conocidos como *Strong Jumping Emerging Patterns* (SJEPs). Este tipo de patrón emergente debe de cumplir dos características principales: (Fan y Ramamohanarao, 2002, 2006)

$$eJEPs = \{P | GR(P) = \infty \wedge \nexists S \subset P \text{ t.q. } GR(S) = \infty\} \tag{2.12}$$

- **EPs minimales (MinEPs).** Un MinEP se define como un EP cuyos sub-patrones ya no son EPs. Esto quiere decir que los patrones minimales son los que contienen la mayor capacidad de generalidad de todos, ya que su número de variables suele ser, por lo general, bajo (Fan y Ramamohanarao, 2002, 2006).

$$MinEPs = \{P | GR(P) \geq \rho \wedge \nexists S \subset P \text{ t.q. } GR(S) \geq \rho\} \tag{2.13}$$

- ***Maximal Emerging Patterns*** **(MaxEPs).** Al contrario que los patrones minimales, los patrones maximales son patrones en el que ningún superconjunto de P es un EP. Esto reduce al mínimo la posibilidad de provocar duplicidades en la clasificación, sin embargo, al ser muy específicos, es difícil poder clasificar una nueva instancia porque muy pocos patrones coincidirán con ella (Wang et al., 2004c).

$$MaxEPs = \{P | GR(P) \geq \rho \wedge \nexists Q \supset P \text{ t.q. } GR(Q) \geq \rho\} \tag{2.14}$$

- **EPs tolerantes a ruido (NEP)**, también llamados *constrained emerging patterns* (CEPs). Los NEP relajan un poco la definición de JEP, permitiendo así capturar patrones que no se podrían obtener si hubiera ruido de clases. Más formalmente, un NEP de D_1 a D_2 debe cumplir (Bailey et al., 2003; Fan y Ramamohanarao, 2006):

$$NEPs = \{P | Sop_1(P) \leq \delta_1 \wedge Sop_2(P) \geq \delta_2\} \tag{2.15}$$

  En donde normalmente $\delta_2 \gg \delta_1$ con el objetivo de obtener patrones con un alto GR.

- **JEPs con negación (JEPNs).** Son una extensión del subconjunto de los JEPs. Este tipo de patrón permite la representación de valores negados. Un valor negado representa la idea de que dicho valor no aparece en un ejemplo, permitiendo el resto. Por lo tanto, siendo I el conjunto inicial de selectores, y $\bar{I} = \{\bar{i}\}_{i \in I}$ el conjunto de selectores con valores negados, el nuevo conjunto de los selectores para los JEPNs es definido como $\mathcal{I} = \{p \subseteq I \cup \bar{I} | \forall_{i \in I} i \in p \rightarrow \bar{i} \notin p\}$. Con este nuevo espacio de búsqueda, un JEPN es similar a un JEP, es decir:

$$JEPNs = \{P | GR(P) = \infty\} \tag{2.16}$$

  donde $P = \{x_1, x_2, ..., x_n | x_i \in \mathcal{I}\}$. Un ejemplo de este tipo de patrones podría ser el siguiente:

$$P = \{(Odor = \overline{none}), (G.Size = broad), (Ring.Number = one)\}$$

Este patrón representa aquellos valores sin el valor $none$ en la variable $Odor$, pero sí cualquier otro, y contiene *G.Size = broad* y *Ring.Number = one*.

- ***Chi Emerging Patterns*** **(Chi EPs).** Son patrones similares a los NEPs, sin embargo en este tipo de patrones se incluye información sobre la capacidad discriminativa de cada selector en el patrón mediante un test χ^2, este test deberá dar como resultado que todos los selectores del patrón contribuyen de manera significativa a la capacidad discriminativa del mismo. Formalmente, un patrón P será Chi EP si (Fan y Ramamohanarao, 2004):

  1. $Sop(P) \geq \xi$, con ξ como un umbral mínimo de soporte.
  2. $GR(P) \geq \rho$, con ρ como umbral de mínimo índice de crecimiento.
  3. $\nexists S \subset P$ t. q. $(Sop(S) \geq \xi) \wedge (GR(S) \geq \rho) \wedge (Strength(S) \geq Strength(P))$ donde $Strength(P) = \frac{GR(P)}{GR(P)+1} Sop_2(R)$ (Ramamohanarao y Fan, 2007).
  4. $|P| = 1 \vee |P| > 1 \wedge \forall S \, (S \subset P \wedge |S| = |P| - 1) \Rightarrow chi(P,S) \geq \eta$ donde $\eta = 3{,}84$ es un umbral mínimo para chi-cuadrado y la función $chi(P,S)$ se calcula mediante una tabla de contingencias.

- ***Fuzzy Emerging Patterns*** **(FEP).** Este tipo de patrón utiliza lógica difusa para representar variables de tipo numérico, usando para ello selectores difusos del tipo $[Atributo \in Cjto.Difuso]$. Usando este tipo de patrones se obtienen ventajas en interpretabilidad, ya que son más simples de entender por el experto, y en flexibilidad, ya que los patrones cubren a las instancias con un determinado grado de pertenencia (García-Borroto et al., 2011).

2.2.3 Algoritmos de minería de patrones emergentes

A lo largo de la literatura especializada se han desarrollado una gran cantidad de métodos de extracción de los diferentes tipos de EPs, analizados en la sección anterior. Sin embargo, hasta donde nosotros sabemos, no existe a día de hoy una aportación relevante en la literatura enfocada a la extracción eficiente de algún tipo de EP en entornos de flujo de datos que sean capaces de afrontarlos de manera eficiente a lo largo del tiempo.

Por esta razón, en esta sección se presentará brevemente los detalles de los enfoques más relevantes utilizados para la extracción de EPs a lo largo de la literatura. El objetivo es presentar una visión global del desarrollo de métodos en EPM para poder establecer un punto de partida para el desarrollo de una estrategia de búsqueda.

Actualmente, los desarrollos realizados a lo largo de la literatura pueden ser agrupados según el tipo de estructura de datos o algoritmo de generación de patrones que se utiliza para obtener dicho patrones. En concreto, se pueden encontrar cuatro grupos de algoritmos de EPM (García-Vico et al., 2018): Basados en límites, basados en árboles, basados en árboles de decisión y basados en sistemas difusos evolutivos. A modo de resumen y consulta para el lector, en la Tabla 2.2 se presentan los algoritmos más destacados de cada uno de los diferentes grupos con su respectiva referencia.

A continuación se detallarán las características comunes que comparten los distintos algoritmos de cada grupo, de modo que se tenga una visión global de los desarrollos realizados en la literatura (García-Vico et al., 2018):

- **Algoritmos basados en límites.** El concepto de límite fue introducido junto con el propio concepto de EPM por Dong y Li en (Dong y Li, 1999) como método para la extracción eficiente de patrones emergentes. El concepto de límite permite representar de manera compacta grandes cantidades de patrones emergentes. Un límite es un par $< \mathcal{L}, \mathcal{R} >$ en el que $\mathcal{L}$ (límite izquierdo) es un conjunto de patrones minimales y $\mathcal{R}$ (límite derecho) es un conjunto de patrones maximales que deben cumplir:

  1. $\mathcal{L}$ y $\mathcal{R}$ son *anti-cadenas*. Un conjunto de patrones S es una *anti-cadena* si $\forall X, Y \in S, X \not\subseteq Y \wedge Y \not\subseteq X$
  2. Cada elemento de $\mathcal{L}$ es un subconjunto de algún elemento de $\mathcal{R}$ y cada elemento de $\mathcal{R}$ es un superconjunto de algún elemento de $\mathcal{L}$.

  Estas condiciones permiten al límite representar un espacio convexo dentro del espacio de los EPs donde aquellos patrones entre $\mathcal{L}$ y $\mathcal{R}$, es decir, aquellos S tal que $X \subseteq S \subseteq Y$ donde $X \in \mathcal{L}$ e $Y \in \mathcal{R}$ son EPs. Muchos de los algoritmo de este enfoquen utilizan el algoritmo Border-Diff (Bayardo y Roberto, 1998) en su núcleo para incluir todos los EPs cuyo GR es mayor que un umbral dado.

Tabla 2.2: Clasificación de los algoritmos desarrollados para minería de patrones emergentes.

Algoritmos basados en límites
MBD-LLBorders (Dong et al., 1999b)
JEPProducer (Zhang et al., 2000)
DeEPS (Li et al., 2001; Ramamohanarao y Fan, 2007)
Algoritmos basados en representación mediante árboles
Tree-based JEP-C (Bailey et al., 2002)
BCEP (Fan y Ramamohanarao, 2003a)
iEPMiner (Fan y Ramamohanarao, 2003b)
StrongJEP-C (Fan y Ramamohanarao, 2006)
Top-k minimal JEPs (Terlecki y Walczak, 2008)
DGCP (Liu et al., 2014)
Algoritmos basados en árboles de decisión
LCMine (García-Borroto et al., 2010a)
CEPMine (García-Borroto et al., 2010b)
EP Random Forest (Wang et al., 2010)
FEPM (García-Borroto et al., 2011)
Algoritmos basados en sistemas evolutivos difusos
EvAEP (García-Vico et al., 2016)
MOEA-EFEP (García-Vico et al., 2018)
BD-EFEP (García-Vico et al., 2019)

De estos algoritmos se observa que emplean una búsqueda eficiente gracias al concepto de límites. Aunque el número de patrones sigue siendo aún excesivamente elevado y, en el peor de los casos, la complejidad computacional de estos algoritmos es exponencial en función del número de variables. Por esta razón muchos de ellos emplean JEPs para reducir el espacio de búsqueda. Estos, a pesar de ser muy discriminativos, son muy sensibles a ruido. A pesar de todos estos inconvenientes, ofrecen resultados competentes de clasificación. Sin embargo, debido al gran número de patrones obtenidos y la representación de los mismos mediante límites hacen que los resultados obtenidos no sean lo suficientemente interpretables y se pierden muchas de las capacidades descriptivas que poseen estos patrones.

Dentro de este grupo se pueden encontrar los algoritmos iniciales creados para la tarea. De entre todos ellos, destacan el algoritmo DeEPS (Li et al., 2001).

- **Algoritmos basados en representación mediante árboles.** La anterior estrategia tiene una complejidad exponencial en el peor de los casos. Para solventar esto, los algoritmos en este grupo representan los datos de entrenamiento en una estructura de árbol para extraer eficientemente los EPs. En general, los nodos de estos árboles consisten en un selector y un contador que indica la ocurrencia del patrón formado por el camino existente entre la raíz y el nodo actual. Estos selectores son primeramente ordenados por una medida de calidad, habitualmente GR, de modo que los selectores más discriminativos se encuentran más cerca del nodo raíz, haciendo que la minería de los mismos sea más eficiente al haber menos profundidad en el árbol. Finalmente, una vez se tiene el árbol construido, los diferentes algoritmos desarrollados en la literatura normalmente se basan en cómo se va a podar este árbol para realizar una extracción eficiente de los diferentes tipos de EPs.

  Es importante destacar que, en la mayoría de los algoritmos que se encuentran en este grupo, utilizan una heurística de poda basada en un umbral de mínimo soporte. De hecho, la mayoría de métodos intenta obtener SJEPs con un soporte muy alto. La idea detrás de este tipo de poda es que los patrones con muy poco

soporte pueden ser considerados como ruido en los datos, por lo que se suaviza la alta sensibilidad que tienen los JEPs a los datos con ruido. Asimismo, utilizando esta heurística se reduce el espacio de búsqueda, obteniendo un menor tiempo de ejecución que los algoritmos basados en límites.

Los algoritmos más destacados en la literatura que utilizan este esquema son iEPMiner (Fan y Ramamohanarao, 2003b) y StrongJEP-C (Fan y Ramamohanarao, 2006).

- **Algoritmos basados en árboles de decisión.** Los algoritmos encuadrados en esta técnica se basan en la utilización de bosques de árboles de decisión inducidos de los datos de entrenamiento y, a partir de esos árboles, obtener conjuntos de EPs para clasificar. Estos árboles de decisión son modificados para que busquen más candidatos en el espacio de búsqueda, es decir, estos árboles no solo utilizan relaciones de igualdad, sino que emplean todo tipo de relaciones de igual/desigualdad, pertenencia/no pertenencia, etc. El uso de estos operadores les permite el tratamiento de variables numéricas sin necesidad de realizar una discretización previa de los datos. Asimismo, los árboles generados se pueden limitar en profundidad con el objetivo de acelerar el proceso de extracción de EPs.

  La extracción de EPs se realiza mediante el recorrido completo de todos los caminos existentes en los diferentes árboles, o bien mediante operaciones adicionales en las regiones del espacio de búsqueda que delimitan cada uno de los árboles.

  Finalmente, cabe destacar que este tipo de algoritmos permite una mayor interpretabilidad de resultados al tener que utilizarse una mayor combinación de operadores, junto a la posibilidad de tratar operadores numéricos de forma directa o mediante el uso de lógica difusa. Asimismo, los algoritmos que se encuadran en esta técnica no permiten obtener la totalidad de los EPs, como ocurre en los métodos de las secciones anteriores. Sin embargo, son capaces de obtener un conjunto de patrones de gran calidad que permiten obtener resultados competitivos con menores tiempos de ejecución.

De entre los algoritmos existentes que emplean esta técnica, destaca e algoritmo FEPM (García-Borroto et al., 2011).

- **Algoritmos basados en sistemas difusos evolutivos.** Los algoritmos encuadrados en esta técnica utilizan un concepto similar al grupo anterior, es decir, no se intentan obtener todos los EPs posibles, sino un subconjunto de gran calidad. De hecho, el problema de la obtención de patrones emergentes puede ser visto como un problema de optimización, en donde se busca optimizar el conjunto de patrones emergentes extraído para que las medidas de calidad utilizadas sean maximizadas o minimizadas, según sea conveniente. Por esta razón, este grupo de algoritmos utilizan algoritmos evolutivos (Goldberg, 1989) ya que permiten la obtención de una solución cercana a la óptima (la óptima no se garantiza), en un tiempo aceptable. Asimismo, estos algoritmos permiten la optimización de múltiples objetivos, lo cual es clave en la tarea. Finalmente, para la mejora de la interpretabilidad de los resultados (Hüllermeier, 2011) y aumentar su robustez frente a ruido (Fernández et al., 2017) hacen uso de lógica difusa mediante etiquetas lingüísticas en aquellas variables de tipo numérico. En concreto, la mayoría de estos métodos se basan en representar los posibles EPs como individuos a optimizar por el proceso evolutivo. Después, estos EPs irán modificándose de acuerdo a los diferentes operadores genéticos del problema.

  Es importante destacar que este tipo de algoritmos tienen una mayor facilidad de trabajo que los grupos anteriores en entornos de grandes volúmenes de datos o Big Data en donde un trabajo distribuido es necesario. En este aspecto, existen varios desarrollos en la literatura los cuales se basan en la realización de manera distribuida del proceso de evaluación de cada uno de los individuos.

  De entre los algoritmos existentes en este grupo, destaca el algoritmo MOEA-EFEP (García-Vico et al., 2018) y el algoritmo BD-EFEP (García-Vico et al., 2019) como algoritmo de EPM enfocado a entornos Big Data.

CAPÍTULO 3

Algoritmo para la extracción de patrones emergentes difusos en flujos continuos de datos

En este capítulo se describe una nueva propuesta para la extracción de EPs difusos en entornos de flujos continuos de datos, llamada FEPDS (*Fuzzy Emerging Patterns from Data Streams*) (García-Vico et al., Submitted). Este algoritmo permite la extracción de un conjunto de EPs de alta calidad e interpretabilidad el cual se irá actualizando a lo largo del tiempo. El objetivo es que este conjunto de patrones esté constantemente actualizado con respecto al flujo de datos.

En primer lugar se presentan las principales propiedades y componentes del algoritmo FEPDS para comprender su funcionamiento. A continuación, se presenta un estudio experimental para determinar la calidad del método propuesto y su capacidad de adaptación a diferentes tipos de cambio de concepto con el objetivo de determinar la viabilidad de la propuesta en entornos de minería de flujos de datos.

Finalmente, es importante destacar que fruto de la realización de este capítulo, se ha realizado un artículo científico que actualmente se encuentra bajo revisión en la revista *IEEE Transactions on Fuzzy Systems*. Dicho trabajo se adjunta en esta memoria en el Apéndice A.

3.1. FEPDS: Fuzzy Emerging Patterns from Data Streams

En esta sección se presentan los principales detalles del algoritmo denominado FEPDS para la extracción de patrones emergentes difusos en entornos de flujos continuos de datos.

A modo de resumen, los principales componentes del algoritmo son:

- Se considera que las instancias provenientes del flujo de datos son recogidas mediante el empleo de bloques de instancias de tamaño fijo.
- El algoritmo de aprendizaje utilizado para la extracción de patrones emergentes se basa en un sistema difuso evolutivo (Herrera, 2008) multiobjetivo. Esto nos permitirá obtener patrones con un buen balance entre su fiabilidad, generalidad e interpretabilidad. Además, el uso de lógica difusa dentro del proceso evolutivo nos permite obtener robustez frente a pequeños cambios, por lo que se puede obtener una mejor adaptación a cambios graduales.
- Dentro de este método de aprendizaje, se hace uso de una memoria específica para sesgar el aprendizaje en función de los datos históricos. Esta memoria, en concreto, se basa en una ventana deslizante en la que se permite el almacenamiento de los modelos extraídos previamente.
- Se hace uso de una estrategia ciega para actualizar el modelo. Por lo tanto, el algoritmo de aprendizaje es ejecutado cada vez que se obtiene un nuevo bloque de datos. Con esta estrategia se pretende obtener una buena adaptación a cambios graduales ya que el método es ejecutado regularmente.
- Para finalizar, la evaluación del modelo actual se basa en la metodología *test-then-train*, la cual nos permite una evaluación continua del modelo.

A continuación se presentan detalladamente los diferentes elementos de FEPDS. En primer lugar, se presenta el componente de memoria de FEPDS. Luego se presentan los principales conceptos de adaptación del modelo de aprendizaje al flujo de datos. Después, se presenta el algoritmo de aprendizaje basado en un sistema difuso evolutivo multiobjetivo para la extracción de patrones emergentes junto con sus principales componentes. Finalmente, la conexión entre los diferentes componentes presentados se muestra en un esquema operacional.

3.1.1 Componente de memoria

En primer lugar, se asume que los datos llegan al sistema en forma de bloques de datos de tamaño fijo. En el caso de que el flujo de datos no se adapte a esta característica, se lanza un proceso en el algoritmo que consiste en la recolección de los datos provenientes del flujo y formar bloques de tamaño fijo, como se puede observar en la Figura 3.1.

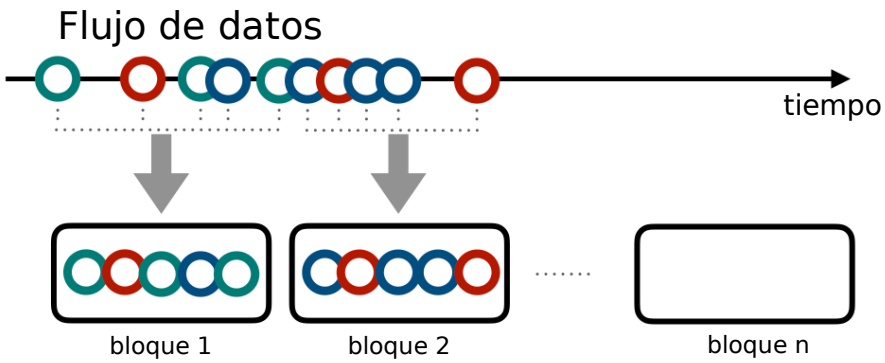


Figura 3.1: Proceso de recolección de datos y transformación en bloques del algoritmo FEPDS.

FEPDS utiliza una estructura de memoria con el objetivo de almacenar cada uno de los conjuntos de EPs difusos extraídos por el algoritmo de aprendizaje cuando procesa los diferentes bloques de datos. Dicha memoria se basa en una ventana deslizante (VD) de tamaño fijo. Asimismo, esta estructura de memoria utiliza una política FIFO cuando la capacidad está completa. En concreto, el tamaño de la VD es inicializado a un valor m fijo, que será el número de máximo de conjuntos de EPs que será capaz de almacenar. Por lo tanto, se asume que el conocimiento extraído hace más de m bloques de datos es lo suficientemente obsoleto como para ser descartado y olvidado por completo por el algoritmo evolutivo de aprendizaje.

Con esta estructura se le otorga al algoritmo la capacidad de olvidar conocimiento antiguo mientras que es capaz de recordar el más reciente para ser utilizado en el proceso de aprendizaje actual. De hecho, es importante destacar que dentro de cada conjunto de EPs, se almacenan los EPs extraídos así como información relativa a su calidad en el instante en el que fueron extraídos. De este modo, el algoritmo de aprendizaje es capaz de utilizar dicha información para intentar ajustarse lo mejor posible al concepto actual.

3.1.2 Proceso de adaptación del aprendizaje al cambio de concepto

Como se ha comentado, el proceso de adaptación al cambio de concepto por parte de FEPDS se basa en una estrategia ciega con un reentrenamiento completo. Este proceso de aprendizaje es, por tanto, ejecutado en cada bloque de datos y, por cada ejecución, un nuevo conjunto de EPs es extraído, sustituyendo al anterior. Esta estrategia de adaptación va a permitir a FEPDS adaptarse mejor a cambios de tipo gradual ya que se está continuamente adaptando el modelo de patrones a los datos más recientes del conjunto de datos.

El proceso de adaptación, representado gráficamente en la Figura 3.2, funciona de la siguiente manera:

1. Se inicializa la estructura de VD presentada anteriormente como vacía.

2. Una vez el algoritmo dispone de un bloque de datos que procesar, el proceso de recolección de datos continúa funcionando de manera asíncrona a todo el proceso recolectando más datos para la siguiente ejecución.

3. Si hubiera un conjunto de EPs (cEP) o modelo sin evaluar, siguiendo la estrategia *test-then-train* este sería evaluado con el nuevo bloque de datos que hay disponible.

4. Tras la evaluación, el proceso de aprendizaje evolutivo obtiene conocimiento basado en el bloque de datos actual y los m anteriores almacenados en la VD, si los hubiera.

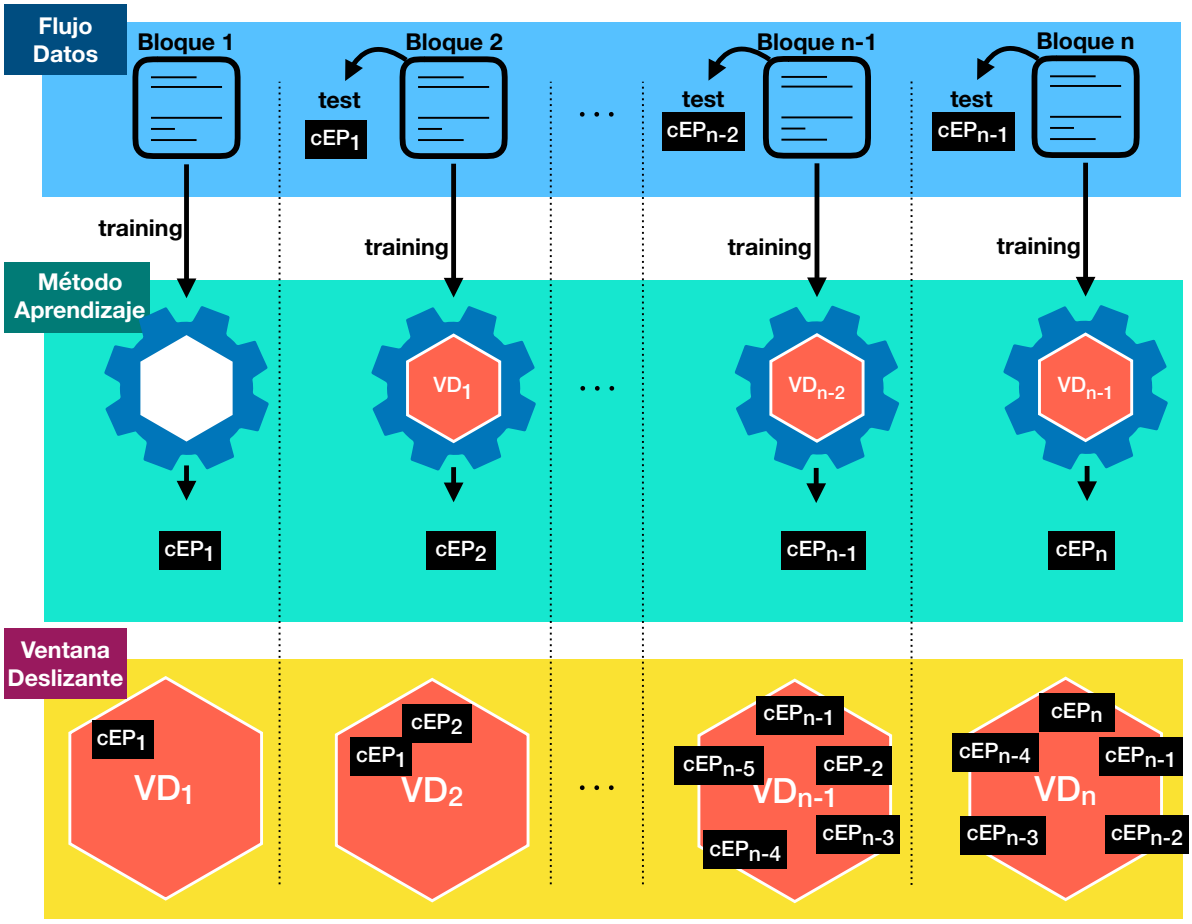


Figura 3.2: Esquema general de funcionamiento del algoritmo FEPDS.

5. Finalmente, el conjunto de EPs extraído en el bloque de datos actual es incorporado en la VD. Así, la estructura de VD se actualiza incrementalmente por medio de la inclusión del conocimiento previo en el proceso evolutivo de aprendizaje actual.

La razón de este proceso se debe a que este conocimiento extraído recientemente contiene un resumen muy valioso del flujo de datos. Por lo tanto, aquellos EPs que han ido apareciendo de manera frecuente en bloques anteriores deben de tener mucho más peso en el proceso de aprendizaje evolutivo que aquellos que no. De hecho, aquellos que no aparecen deberían ser gradualmente olvidados ya que se asume que describen conceptos antiguos. Con este proceso se evita dar relevancia a aquella información sobre el estado actual del flujo de datos que no es relevante dentro del proceso de aprendizaje.

3.1.3 Enfoque de aprendizaje evolutivo

El algoritmo de aprendizaje de FEPDS es un sistema difuso evolutivo multiobjetivo para la extracción de EPs difusos. Para guiar correctamente el proceso de búsqueda, es necesario determinar la calidad de los EPs. Esta se determina mediante diferentes objetivos, los cuales son las diferentes medidas de calidad presentadas anteriormente en la sección 2.2.1. No obstante, el rendimiento de un algoritmo multiobjetivo decae cuando el número de objetivos a optimizar es mayor que dos (Ishibuchi et al., 2008). Por esta razón, en este trabajo los objetivos a ser optimizados serán la atipicidad y la diferencia de soporte. Estas medidas nos proporcionan en su conjunto un buen balance entre la generalidad de las reglas extraídas y su fiabilidad. Por otro lado, este proceso evolutivo hace uso de una población élite con el objetivo de almacenar el mejor conjunto de patrones encontrado para el bloque de datos actual. Sin embargo, la inclusión de información acerca de la evolución del flujo de datos es clave para ajustar el conocimiento extraído al fenómeno subyacente que genera el flujo de datos. De este modo, esta población élite se actualiza mediante un proceso de *token competition* (Leung et al., 1992) en el que se emplea la información de la VD para recompensar aquellos patrones más representativos del flujo de datos a lo largo del tiempo. A continuación, se detallan los elementos más importantes de este algoritmo de aprendizaje.

En concreto, se presenta en primer lugar la representación de los EPs utilizada. A continuación se detallan los operadores genéticos empleados y, finalmente, se presenta en detalle el proceso de actualización de la población élite basado en *token competition* con recompensa basándose en la información que se encuentra en la VD.

3.1.3.1. Representación del conocimiento

El sistema difuso evolutivo hace uso de una representación del conocimiento basada en el esquema "cromosoma = patrón" (Wong y Leung, 2000) en donde un individuo o cromosoma de la población representa una patrón potencial. Por lo tanto, el resultado final devuelto por el algoritmo será un conjunto de estos individuos. En FEPDS se hace uso de lógica difusa para la representación de variables numéricas a través del uso de etiquetas lingüísticas. La forma de estas etiquetas lingüísticas puede ser determinada por parte del experto o bien se definen mediante funciones de pertenencia triangulares uniformes, las cuales han tenido bastante éxito en aplicaciones reales tanto en EPM como en tareas similares como el descubrimiento de subgrupos. Un ejemplo de representación de una variable numérica mediante el empleo de cinco etiquetas lingüísticas triangulares uniformes es mostrado en la Fig. 3.3.

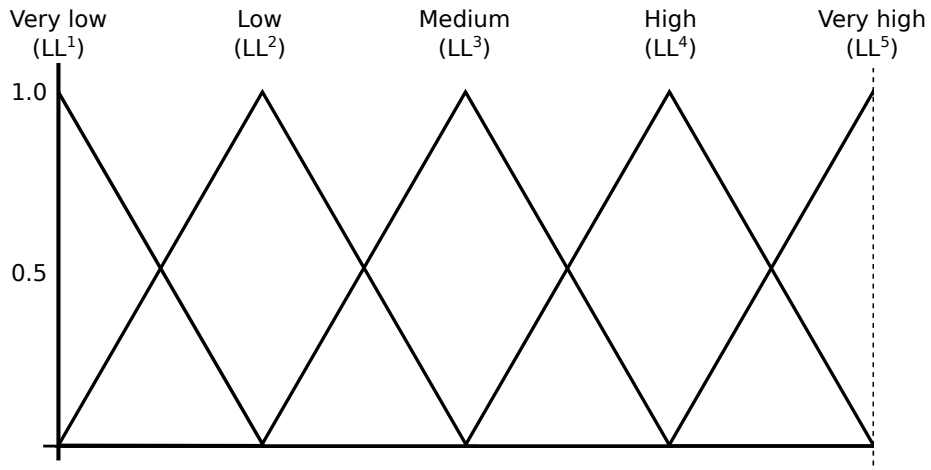


Figura 3.3: Representación de una variable numérica mediante cinco etiquetas lingüísticas triangulares uniformes.

Este sistema difuso evolutivo extrae patrones representados mediante forma normal disyuntiva (DNF), ya que es una buena representación para EPs descriptivos (García-Vico et al., 2018). Los patrones DNF son codificados en el algoritmo evolutivo mediante el empleo de un vector de bits de longitud igual al número de selectores. Además, la parte del consecuente se representa en estos patrones mediante un valor entero que permite la extracción de patrones para todas las clases del problema en una única ejecución, acelerando el proceso de búsqueda. Un ejemplo de EP difuso y su representación en el algoritmo de aprendizaje de FEPDS se muestra en la Fig. 3.4.

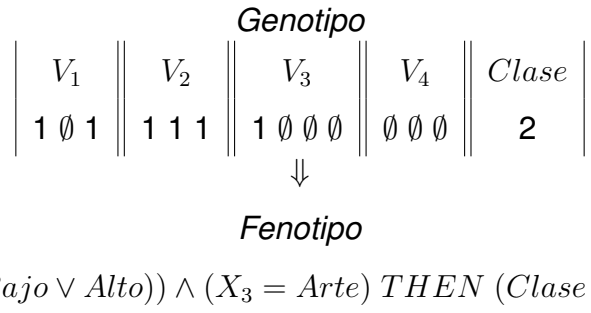


Figura 3.4: Representación de un patrón difuso en forma normal disyuntiva con variables categóricas y continuas en FEPDS.

3.1.3.2. Operadores genéticos

A pesar de que el proceso de adaptación del aprendizaje está basado en un esquema de reentranamiento del modelo, el algoritmo de aprendizaje evolutivo de FEPDS inicializa su población de individuos introduciendo en primer lugar aquellos EPs extraídos para el bloque de datos previo, si hubiera disponibles. El objetivo de este proceso es introducir conocimiento sobre el estado actual del flujo de datos para poder guiar el proceso de búsqueda en el bloque de datos actual. A continuación, los individuos restantes hasta completar el tamaño completo de la población se inicializan aleatoriamente.

Los operadores genéticos empleados en el proceso evolutivo son:

- Un operador de selección por torneo binario (Miller y Goldberg, 1995). Con este operador mantenemos una fuerte presión selectiva para obtener una convergencia rápida del algoritmo genético.

- Un operador de cruce en dos puntos (Holland, 1975). Este operador nos permitirá intensificar en aquellas zonas prometedoras del espacio de búsqueda.
- Un operador de mutación clásico en el que se modifica el valor de un gen escogido al azar de manera aleatoria, lo que nos permitirá diversificar la búsqueda.
- Un operador de mutación orientada en donde se elimina una variable al completo de un patrón. Con este proceso se permite además de diversificar la búsqueda, reducir la complejidad de los posibles EPs.

Finalmente, se hace uso de un operador de reinicialización con el objetivo de actualizar la población élite, ya que estamos ante un máximo (local o global), para a continuación moverse a otra zona del espacio de búsqueda prometedora con el objetivo de encontrar otro máximo mejor. Este proceso es inicializado cuando el frente de Pareto no evoluciona durante al menos un 25 % del total de evaluaciones y al final del proceso evolutivo para actualizar la población élite, la cual contiene el conjunto de patrones emergentes a devolver al experto. Se dice que la población no evoluciona cuando no es capaz de cubrir nuevos ejemplos no cubiertos anteriormente. Este proceso de reinicialización se basa en el proceso de recompensa en *token competition*, detallado a continuación.

3.1.3.3. Proceso de recompensa en token competition

La recompensa en el proceso de *token competition* se aplica en el proceso de reinicialización del proceso evolutivo, que es cuando se actualiza la población élite. La idea de este proceso es recompensar aquellos patrones extraídos en bloques anteriores ya que no solo contiene conocimiento relevante sobre el flujo de datos, sino también tiene la capacidad de simplificar el modelo de patrones ya que se necesitan menos para cubrir el espacio de ejemplos. Concretamente, estos patrones que han sido extraídos anteriormente provienen de la VD, descrita anteriormente. Con esta estructura, se utiliza una función $SW(P_i, j)$ que devuelve un valor de uno si el patrón P_i se encuentra en el conjunto de patrones emergentes devuelto para el bloque de datos j o cero en caso contrario.

El proceso de reinicialización utilizando la recompensa en *token competition* funciona de la siguiente manera: en primer lugar, para cada uno de los individuos P_i de la población actual Pob_g del proceso evolutivo se calcula un valor Div_t, calculado con la función utilizada en la ecuación 3.1.

$$Div_t(P_i) = WRAcc_t(P_i) + \sum_{j=t-n}^{t} SW(P_i, j) \cdot 2^{-(t-j)} \cdot WRAcc_j(P_i) \tag{3.1}$$

donde $WRAcc_t(P_i)$ es el valor de atipicidad para el bloque de datos actual t para el patrón P_i, y $WRAcc_j(P_i)$ es el valor de atipicidad para un bloque de datos anterior j para el mismo patrón P_i. Por lo tanto, se puede observar que la recompensa que se añade actualmente es un valor proporcional a la calidad de dicho patrón en el instante de tiempo en el que fue obtenido. En este sentido, la recompensa permite al algoritmo una mejor adaptación a los posibles cambios de concepto que se puedan producir.

A continuación, se lanza el proceso de *token competition*. En este proceso, cada una de las instancias del bloque de datos actual posee un *token*. A continuación, se ordena la población Pob_g en orden descendiente por el valor Div. Después, cada individuo P_i recogerá aquellos *tokens* que no hayan sido obtenidos previamente por individuos con un valor Div mayor. Es importante destacar que un individuo P puede adquirir un *token* si P_i cubre a dicha instancia. Una vez procesados todos los individuos, aquellos que no hayan podido obtener algún *token* son eliminados. Finalmente, la población élite es actualizada con el resultado de este proceso si y solo si su valor de atipicidad medio es superior al de la población élite actual.

Con este proceso se intenta conseguir un doble objetivo. Por un lado, este tipo de operación nos permitirá obtener un conjunto de patrones emergentes con un bajo número de patrones y con un nivel de solapamiento reducido pues estos patrones describen zonas diferentes del espacio de búsqueda. Por otro lado, la política de actualización de la población élite basada en atipicidad media nos permite obtener aquel conjunto de patrones emergentes con una mejor relación entre cobertura y fiabilidad.

Tras finalizar la actualización de la población élite, es necesario generar la población de la siguiente generación Pob_{g+1} con el objetivo de escapar del óptimo actual. Los nuevos individuos de Pob_{g+1} son generados mediante un proceso de reinicialización guiada (García-Vico et al., 2018). En este proceso, se elige al azar una instancia que no haya sido cubierta aún por un individuo generado anteriormente. A continua-

ción, se crea un individuo con un máximo de un 25 % de sus variables inicializadas y que sea capaz de cubrir esta instancia. Con este proceso de inicialización se promueve la diversificación de la población en diferentes zonas del espacio de búsqueda con el objetivo de encontrar otro óptimo.

3.1.4 Esquema operacional de FEPDS

En esta sección se presenta el esquema operacional del algoritmo FEPDS, donde se muestra la interacción de todos los elementos presentados anteriormente. Para facilitar la comprensión del funcionamiento del método, en Algoritmo 1 se presenta el pseudocódigo de FEPDS.

FEPDS hace uso de una estrategia *test-then-train* para evaluar el modelo obtenido por el procedimiento de aprendizaje de manera constante. Este esquema de evaluación se muestra en las líneas 6-8 en donde se puede observar que el modelo de patrones extraído, representado como cEP (conjunto de EPs), en el bloque de datos anterior es testeado en el bloque de datos actual si lo hubiese, y los resultados son presentados al experto.

A continuación, siguiendo una estrategia de adaptación de aprendizaje ciega el algoritmo se ejecuta para cada bloque de datos. Una vez este bloque de datos se completa (líneas 3-5), y el conjunto de patrones emergentes actual ha sido evaluado contra este nuevo bloque de datos (línea 7), se lanza el proceso de aprendizaje evolutivo para la extracción de un conjunto de patrones emergentes basado en un enfoque de reentrenamiento (líneas 9-24). El proceso evolutivo empieza inicializaciando su población inicial P_0 mediante el operador de inicialización presentado anteriormente en donde se incluye el conjunto de patrones emergentes previo (línea 10). A continuación se evalúa P_0 para determinar el valor de calidad de los objetivos a optimizar por parte del proceso evolutivo (línea 11). Después el proceso evolutivo da comienzo (líneas 13-24) en donde la condición de parada es la llegada a un número máximo de generaciones o bien hay un nuevo bloque de datos disponible.

Dentro del proceso evolutivo, la población actual P_g crea una población de descendientes Off_g mediante la aplicación de los diferentes operadores genéticos presentados anteriormente (línea 14). Tras esto, Off_g es evaluada para determinar la calidad de los descendientes en función de los objetivos a optimizar (línea 15). Una

Algoritmo 1 Esquema del algoritmo FEPDS.

$t \leftarrow 0$
repeat
  while Tamaño de bloque es menor que n **do**
    Recoger datos del flujo de datos.
  end while
  if $cEP_t \neq \emptyset$ **then**
    Test de cEP_t contra el bloque actual.
  end if
  $g \leftarrow 0$
  $P_g \leftarrow$ Inicialización(cEP_t)
  Evaluar(P_g)
  $t \leftarrow t + 1$
  while $g < maxGens$ & no hay un nuevo bloque disponible **do**
    $Off_g \leftarrow$ OperadoresGenéticos(P_g)
    Evaluar(Off_g)
    $R_g \leftarrow P_g \cup Off_g$
    $F \leftarrow$ OrdenaciónDominancia(R_g)
    if FrentePareto(F) no evoluciona **then**
      $P_{g+1} \leftarrow$ RecompensaEnTokenCompetition(F, VD)
    else
      $P_{g+1} \leftarrow$ RellenarPorFrentes(F)
    end if
    $g \leftarrow g + 1$
  end while
  $cEP_t \leftarrow$ RecompensaEnTokenCompetition(F,VD)
  Añadir cEP_t a VD
  return cEP_t
until El final del flujo de datos.

vez terminado el proceso de evaluación de la población de descendientes, ambas poblaciones se unen y se aplica el operador de ordenación por dominancia (Deb et al., 2002) (líneas 16-17). Se dice que un individuo X domina a otro Y si todos los valores objetivo de X son mejores que los de Y. El proceso de ordenación por dominancia nos devuelve los individuos de la población agrupados en diferentes frentes F_i donde i indica el número de individuos que son dominados por los individuos de F_i. En nuestro caso, el frente más interesante es F_0 pues este frente contiene a los individuos no dominados o frente de Pareto. Tras la finalización del proceso de evaluación, se comprueba la aplicación del proceso de reinicialización y actualización de la población élite mediante el proceso de recompensa en *token competition*. Si la población se estanca, es decir, no evoluciona durante un 25 % del total de evaluaciones máximas, se aplica el proceso de reinicialización y recompensa en *token competition*, el cual hace uso de la VD (línea 19). En caso contrario, P_{g+1} se rellena introduciendo los primeros k frentes enteros o los primeros k individuos de un frente si no quedara espacio para introducir el frente completo (línea 21).

Tras finalizar el proceso evolutivo, el proceso de recompensa en *token competition* es aplicado una vez más para filtrar patrones solapados, mejorando aquellos que son capaces de mantenerse en el tiempo (línea 25). Finalmente, la población élite resultante será el nuevo conjunto de patrones emergentes extraído y que será devuelto al experto tras su inclusión en la estructura de VD, la cual sigue una política FIFO (líneas 26-27).

Es importante destacar que todo este proceso será ejecutado de manera continua hasta la finalización del flujo de datos.

3.2. Estudio experimental

En esta sección se presentan los detalles del estudio experimental llevado a cabo para determinar la calidad del método propuesto así como la adaptabilidad del mismo al cambio de concepto. En primer lugar, se presenta el marco experimental en donde se describen los detalles de la experimentación llevada a cabo. Luego se presenta el análisis de la adaptabilidad del método a diferentes cambios de concepto. Finalmente, se presenta y se analiza la calidad del conocimiento extraído por el método propuesto en este trabajo.

3.2.1 Marco experimental

En este apartado se muestran los detalles para la reproducción de los estudios experimentales llevados a cabo. El primer estudio realizado está relacionado con la adaptabilidad del método al cambio de concepto. El segundo estudio se refiere a la determinación de la calidad del conocimiento extraído. Los detalles del estudio llevado a cabo son los siguientes:

- Datasets. En el primer estudio, los conjuntos de datos utilizados han sido generados artificialmente utilizando la herramienta MOA (Bifet et al., 2010) que nos permite generar flujos de datos en donde se crean cambios de concepto de diferentes tipos con control del lugar en donde se produce el mismo. Dos conjuntos de tres flujos de datos diferentes cada uno fueron generados: uno contiene un cambio de concepto abrupto, mientras que el otro contiene un cambio gradual. En ambos casos, los cambios se producen cada 50.000 instancias. Para el cambio gradual, se establece una ventana de cambio de 10.000 instancias. Todos los conjuntos de datos fueron generados con un total de 500.000 instancias, por lo que se producen exactamente 9 cambios a lo largo del progreso de los flujos de datos.

Para el segundo estudio, se ha utilizado un conjunto de 94 conjuntos de datos artificiales generados con MOA utilizando diferentes generadores de flujo de datos. Cada uno de estos conjuntos de datos contiene un millón de instancias. Para cada uno de estos conjuntos, se han generado cinco flujos de datos con diferentes semillas con el objetivo de promediar el resultado para evitar sesgos.

- Parámetros. En ambos estudios se han empleado los mismos parámetros. La selección de los valores empleados ha sido ampliamente utilizada a lo largo de la literatura en algoritmos evolutivos para EPM. En concreto, el algoritmo de aprendizaje de FEPDS se ejecutará como máximo durante 60 generaciones. Se utilizarán tres etiquetas lingüísticas para la representación de variables de tipo numérico. El tamaño de la población para cada una de las clases del problema es 50. Las probabilidades de cruce y mutación son 0.8 y 0.1 respectivamente. Finalmente, el tamaño de la estructura de VD es 5 y el tamaño del bloque es de 2500 instancias.

- Medidas de calidad. La confianza es la probabilidad condicional de la clase de un patrón, dado el antecedente del mismo (García-Borroto et al., 2017). Utilizando esta idea, cuando se produce un cambio de concepto la confianza media del conjunto de patrones emergentes extraído se puede ver afectada significativamente debido a que esta probabilidad condicionada cambia. Por lo tanto, la confianza se puede utilizar como un indicador para el análisis del comportamiento del algoritmo propuesto respecto al cambio del concepto en el primero estudio. Asimismo, es interesante determinar el tiempo de ejecución para comprobar si la propuesta es capaz de responder en tiempo real.

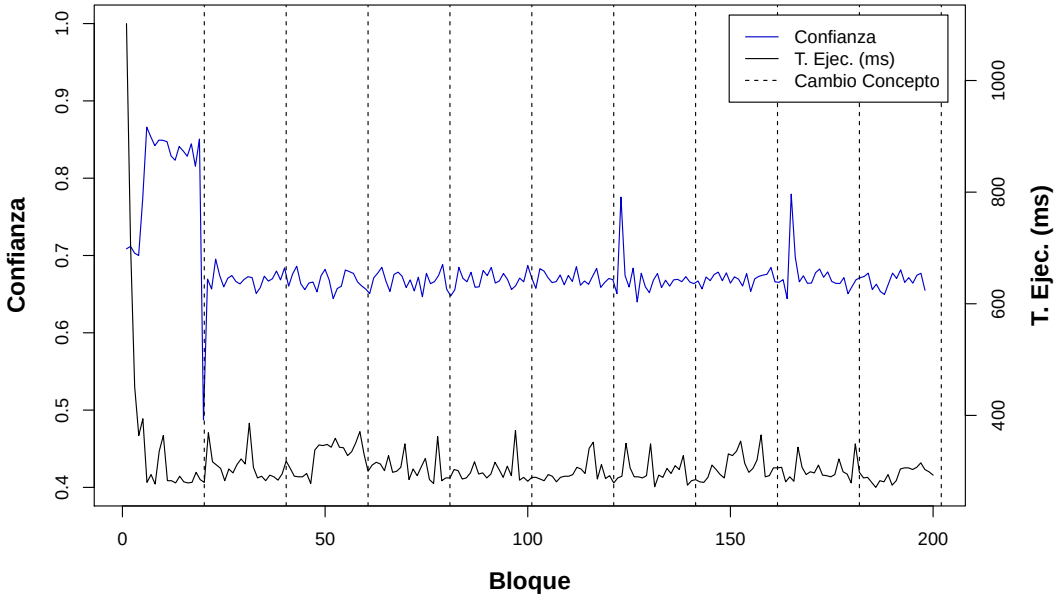
  Para el segundo estudio, las medidas de calidad que se muestran son las relativas a la descripción de las principales características que se analizan en EPM (García-Vico et al., 2018): la complejidad del modelo mediante el número de patrones (n_r) y el número medio de variables (n_v), WRAcc para medir novedad y fiabilidad. El GR promedio, confianza y FPR se utiliza para determinar la fiabilidad. Finalmente, se hace uso del TPR para determinar la generalidad.

3.2.2 Análisis de la adaptación al cambio de concepto

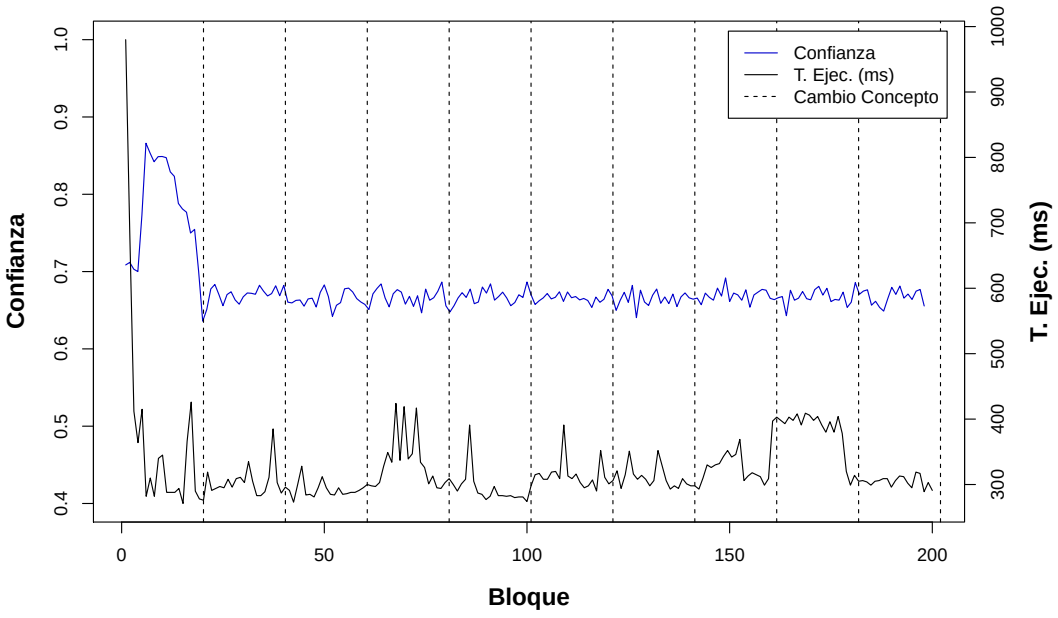
En la Figura 3.5 se muestra el rendimiento de FEPDS respecto a diferentes cambios de concepto analizados. En concreto, las Figuras 3.5a, 3.5c, 3.5e contienen un cambio abrupto mientras que las Figuras 3.5b, 3.5d, 3.5f contienen un cambio gradual. Es importante destacar que cada cambio que se produce en los flujos de datos es remarcado con una línea vertical y punteada en cada gráfico.

En todos los conjuntos de datos analizados el rendimiento tras el primer cambio de concepto cambia de manera significativa. Después de este punto se puede observar que el algoritmo se recupera rápidamente del cambio de concepto producido gracias al proceso de adaptación de aprendizaje ciego. En general, tras este primer cambio, el algoritmo permanece estable independientemente de que se produzca cambio de concepto. Este hecho puede ser producido gracias al empleo de lógica difusa en aquellas variables de tipo numérico. El uso de lógica difusa permite manejar cierto nivel de incertidumbre que es bastante útil cuando el cambio de concepto se produce, ya que nos proporciona un nivel de tolerancia en cambios pequeños, que pueden ser tratados como ruido, y en cambios graduales sin afectar al rendimiento del método. Finalmente se puede observar que, a pesar del enfoque de reentrenamiento, el tiempo de ejecución del método no se ve afectado significativamente con respecto al cambio de concepto. De hecho, en promedio el tiempo de ejecución es cercano a los 300 milisegundos por bloque, el cual es lo suficientemente rápido para proporcionar una respuesta en tiempo real.

Por lo tanto, se puede observar que el método es capaz de responder de manera eficaz y eficiente a diferentes tipos de cambio de concepto.

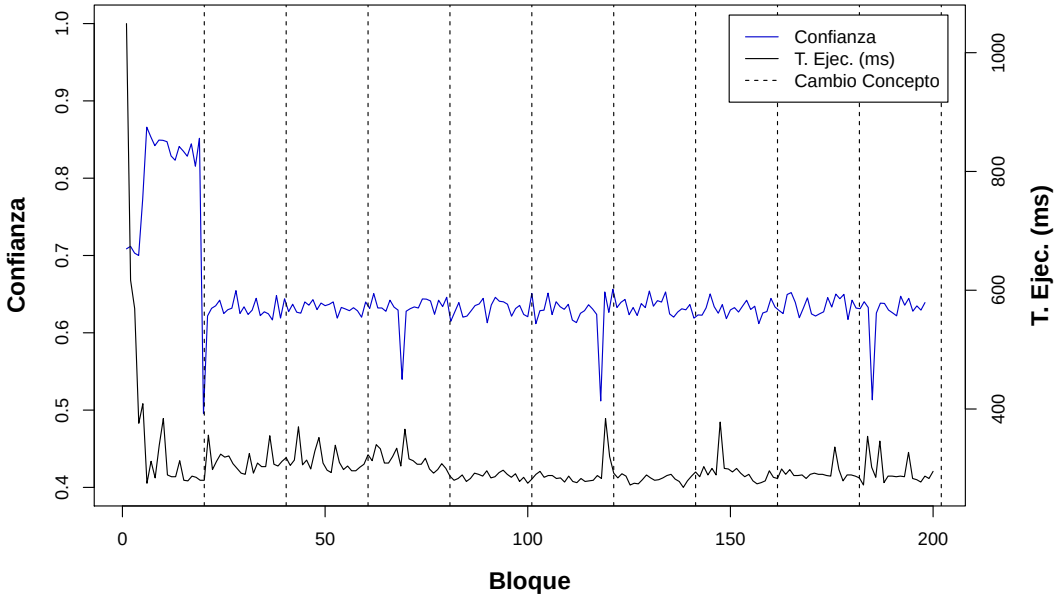


(a)

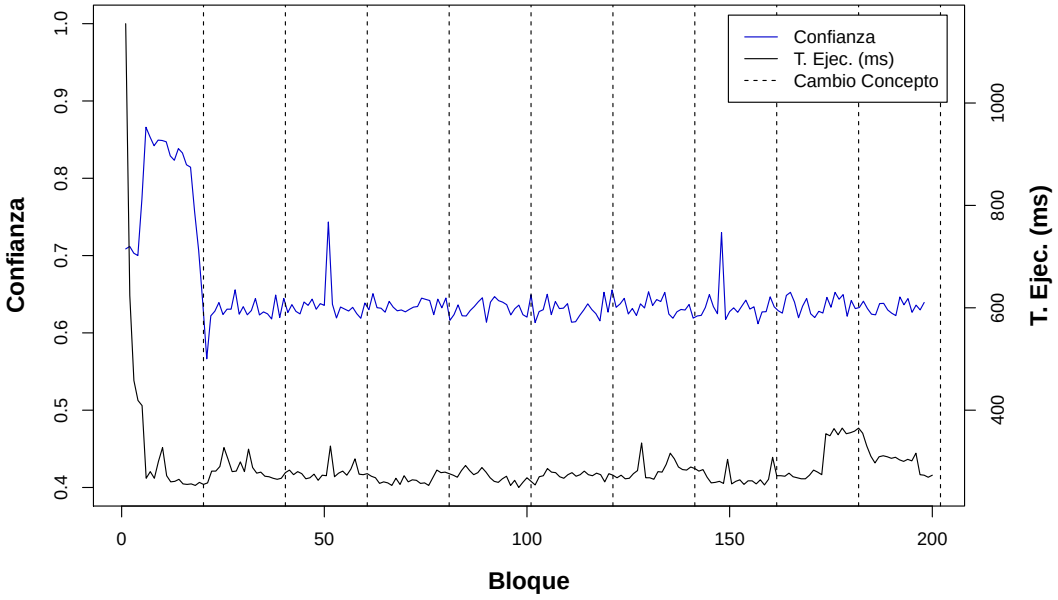


(b)

Figura 3.5: Confianza y tiempo de ejecución del algoritmo FEPDS con diferentes flujos de datos con cambio de concepto. (a), (c) y (e) contienen un cambio de concepto abrupto, mientras que (b), (d) y (f) son flujos con cambio de concepto gradual.

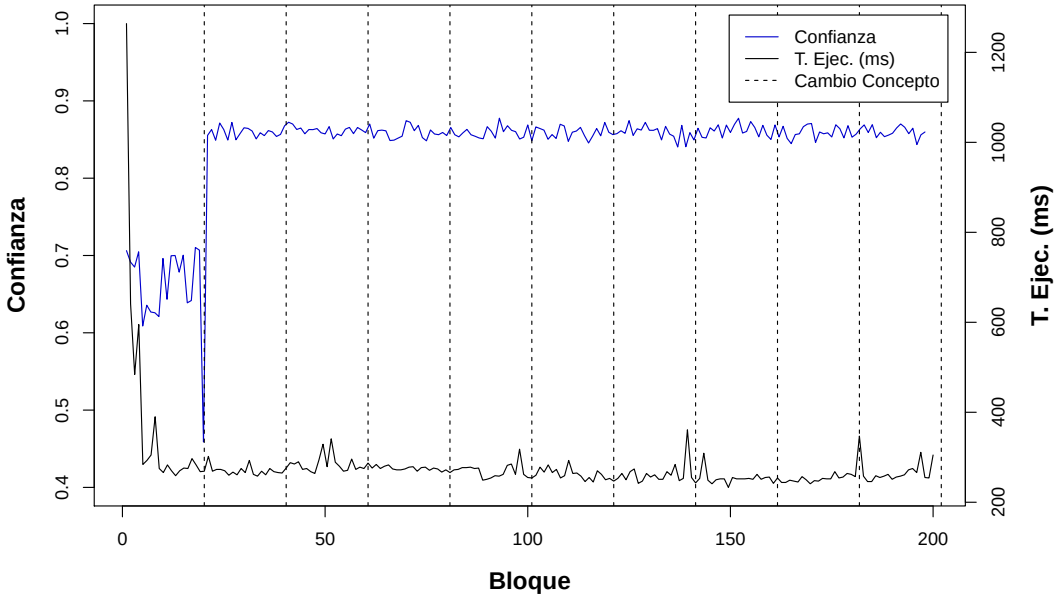


(c)

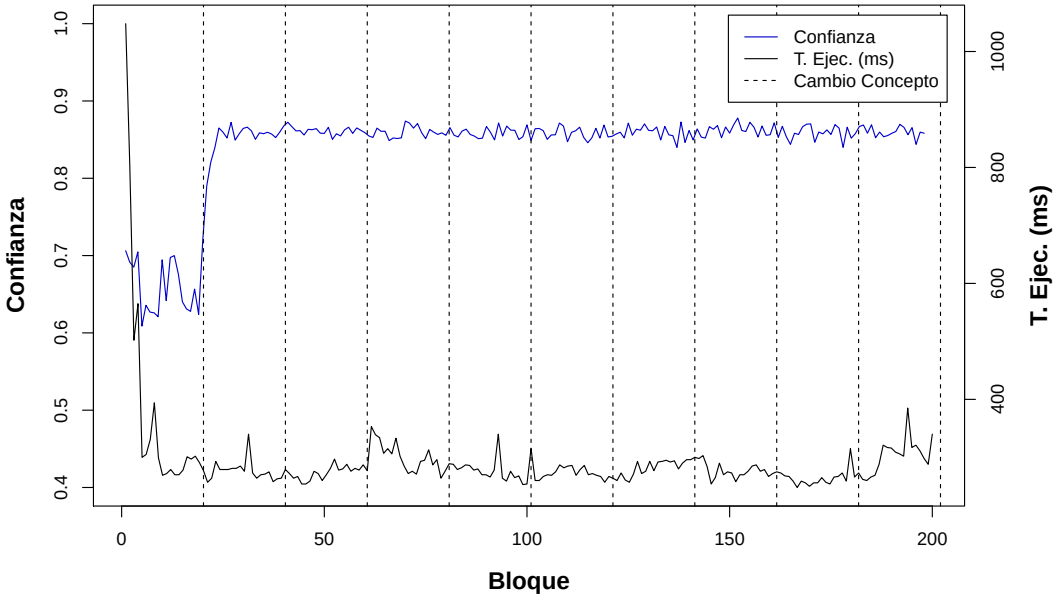


(d)

Figura 3.5: Confianza y tiempo de ejecución del algoritmo FEPDS con diferentes flujos de datos con cambio de concepto. (a), (c) y (e) contienen un cambio de concepto abrupto, mientras que (b), (d) y (f) son flujos con cambio de concepto gradual.



(e)



(f)

Figura 3.5: Confianza y tiempo de ejecución del algoritmo FEPDS con diferentes flujos de datos con cambio de concepto. (a), (c) y (e) contienen un cambio de concepto abrupto, mientras que (b), (d) y (f) son flujos con cambio de concepto gradual.

3.2.3 Análisis de la calidad del conocimiento extraído

En este apartado se muestra la calidad del conocimiento extraído por FEPDS a lo largo del flujo de datos. En la tabla 3.1 se muestra la calidad media de un EP extraído por FEPDS en los diferentes flujos de datos analizados. La tabla de resultados completos para cada conjunto de datos analizado se presenta en el Apéndice B, debido a su extensión. Un análisis de los resultados respecto a los diferentes aspectos en EPM se muestra a continuación:

Tabla 3.1: Resultados promedio de FEPDS.

n_r	n_v	WRACC	CONF	GR	TPR	FPR
2.38	2.19	0.75	0.75	0.99	0.71	0.21

- Complejidad del modelo. Se mide mediante n_r y n_v. En general, el conjunto de patrones emergentes que se extrae es simple. De hecho, se puede observar que en muchos casos se extrae un único EP por clase. Por otro lado, el número de variables se mantiene en un valor lo suficientemente bajo como para ser rápidamente comprendido por el experto. Por lo tanto, el conocimiento extraído por FEPDS en cada bloque de datos, de media, es fácil de comprender gracias a su simplicidad.

- Interés. Se mide utilizando WRAcc. Los resultados obtenidos son de gran interés para los expertos, ya que existe una alta relación entre generalidad, en términos de cobertura de instancias por parte de un EP, y su fiabilidad. Esto quiere decir que los EPs extraídos no solo cubren una gran cantidad de instancias, sino que además la mayoría de estas están correctamente cubiertas. Por lo tanto, este tipo de patrones pueden ser muy interesantes para los expertos ya que son una buena aproximación a la descripción del fenómeno que subyace en los datos.

- Generalidad. Determinada mediante TPR, se puede observar que, de media, los EPs extraídos por FEPDS contienen una gran capacidad de generalidad, ya que cada uno de ellos cubre, de media, un 71 % de las instancias de su clase. Esto quiere decir que las medidas tomadas de acuerdo a los resultados extraídos afectarán a un gran número de instancias, lo cual puede ser muy relevante para los expertos.

- Fiabilidad. Se determina mediante diferentes medidas como CONF, GR y FPR. Este elemento es el factor clave en EPM. Como se puede observar, la confianza de los EPs extraídos es muy elevada, de hecho, de media un 75 % de los ejemplos cubiertos por un EP están correctamente cubiertos. Esto, unido a una alta generalidad, como se ha visto en el punto anterior, hace que estos patrones sean de gran interés para los expertos. Por otro lado, el alto porcentaje de GR nos indica que prácticamente todos los patrones extraídos son EPs en los datos de test, indicando el correcto funcionamiento de FEPDS en la extracción de este tipo de patrón. Finalmente, es importante destacar que, a pesar de que el FPR extraído de media es alto, el ratio entre TPR y FPR extraído es lo suficientemente elevado para determinar que estos EPs poseen una gran capacidad discriminativa. Por lo tanto, la conclusión es que los EPs extraídos son fiables y los expertos pueden confiar en ellos.

En EPM es necesario la búsqueda de un buen balance entre tres objetivos: simplicidad, generalidad y fiabilidad, pues buscamos una descripción fácil de comprender del fenómeno subyacente en los datos. De acuerdo a los resultados extraídos y el análisis realizado, la conclusión que se obtiene de este estudio es que, de media, el EP extraído por el algoritmo FEPDS en cada uno de los bloques de un flujo de datos es lo suficientemente simple como para realizar un análisis rápido de los resultados y que es un elemento clave en minería de flujo de datos. Por otro lado, a pesar de la simplicidad de este modelo, no se sacrifica en gran medida la fiabilidad del mismo, ya que los EPs extraídos contiene un gran poder discriminativo unido a una gran confianza que les permite ser muy fiables. Asimismo, los patrones extraídos tienen una

gran capacidad de generalidad, permitiendo cubrir a un gran número de instancias. Todas estas capacidades hacen que los EPs extraídos por el algoritmo FEPDS sean capaces de obtener descripciones interesantes, fiables y eficientes sobre el fenómeno subyacente a lo largo de un flujo continuo de datos.

CAPÍTULO 4

Análisis de patrones para la caracterización de perfiles de clientes en los *yellow taxis* de Nueva York

En este capítulo se presenta un estudio en donde se emplea EPM aplicado a un flujo de datos real. Una de las imágenes más populares de la ciudad de Nueva York son los *yellow taxis*, es decir, sus taxis de color amarillo. Una de las peculiaridades de estos taxis es que no es necesario realizar una reserva previa del servicio. Es decir, para ir en un *yellow taxi* lo único que se necesita es levantar la mano al conductor en la calle y que este esté libre. La flota de *yellow taxis* está regulada por la *New York City Taxi & Limousine Commision* (NYC-TLC), la cual está formada por 13587 taxis. La NYC-TLC recopila información de cada uno de los viajes realizados por cada uno de estos taxis desde el año 2009.

En particular, el problema abordado consiste en la determinación del perfil de cliente a lo largo del tiempo de modo que se optimice el servicio en tiempo real y, a largo o medio plazo, mejorar el servicio de estos taxis. En este sentido, se establecen dos objetivos para este estudio: primero, la determinación de un perfil en tiempo real de los pasajeros y analizar sus resultados; en segundo lugar se establecerá un perfil más general de clientes de modo que se pueda mejorar el servicio a medio y largo plazo.

Para ello, en este capítulo se describirá en primer lugar el problema y el marco experimental del mismo para a continuación realizar un análisis de los resultados obtenidos.

4.1. Marco experimental

Los datos a analizar consisten en el conjunto de viajes realizados a lo largo del año 2017 por parte de los *yellow taxis*[1]. Para realizar una correcta extracción de conocimiento, ha sido necesaria la realización de tareas básicas de preprocesamiento de datos como por ejemplo la eliminación de inconsistencias y *outliers*, variables no relevantes por tener un alto porcentaje de valores idénticos, etc. En el caso de que se quiera reproducir este estudio, se adjunta en esta memoria el script utilizado para llevar a cabo este preprocesamiento.

Tras este preprocesamiento, el conjunto de datos final se compone de 100 millones de instancias con 10 variables cada una. La variable objetivo de este problema es el coste total del viaje. Como se puede observar, esta variable es de tipo numérico, por lo que este valor tiene que ser discretizado previamente para que pueda ser utilizado por FEPDS. El proceso para discretizar esta variable ha sido el criterio de un experto el cual ha determinado los intervalos [0,15), [15,30), [30, ∞], los cuales se corresponden con tarifas bajas, medias y altas, respectivamente. Finalmente, una breve descripción de cada variable del conjunto de datos analizado se presenta en la Tabla 4.1.

4.2. Extracción de patrones emergentes por FEPDS en el flujo de datos de los *yellow taxis* de Nueva York

En esta sección se presenta en la Tabla 4.2 el resultado promedio de la calidad de los patrones emergentes extraídos en cada bloque del flujo de datos. Un análisis de los resultados obtenidos se presenta a continuación:

[1] https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page

Tabla 4.1: Descripción de las variables utilizadas en el conjunto de datos de los taxis de Nueva York.

Nombre	Tipo	Descripción
passenger_count	numérico	Número de pasajeros en el viaje
RateCodeID	nominal	La tarifa en efecto durante el viaje
store_and_fwd_flag	nominal	Conexión con el servidor
payment_type	nominal	El tipo de pago efectuado
extra	nominal	Suplementos de tarifa (Hora punta & Nocturna)
tip_amount	numérico	Cantidad de dinero dado como propina
tolls_amount	numérico	Cantidad de dinero gastado en peaje
PUBorough	nominal	Zona de recogida de pasajeros
DOBorough	nominal	Zona de bajada de pasajeros
total_amount	nominal	Coste total del viaje (clase)

Tabla 4.2: Resultados promedio de FEPDS en el flujo de datos de los taxis de Nueva York.

n_r	n_v	WRACC	CONF	GR	TPR	FPR
3.00	3.63	0.78	0.62	1.00	0.72	0.16

- Complejidad del modelo. Se puede observar que FEPDS devuelve constantemente tres EPs. Esto significa que el método propuesto devuelve únicamente una regla por clase, mientras que el número de variables medio es lo suficientemente bajo para hacer un análisis rápido y sencillo del conocimiento extraído.

- Interés. Los resultados extraídos muestran un excelente WRAcc medio de 0.78. Esto significa que los patrones extraídos contienen un gran balance entre la cobertura de las instancias objetivo y su fiabilidad. De hecho, se puede observar en los resultados que este interés es debido a su alto valor de TPR unido a un valor de confianza bueno. Esto se puede deber principalmente al proceso de recompensa en *token competition* que permite mantener aquellos patrones relevantes encontrados durante el proceso mientras que se intenta buscar otro conocimiento interesante con el objetivo de adaptarse a los cambios. Por lo tanto, los patrones extraídos por FEPDS son interesantes para los expertos ya que presentan un alto balance entre la generalidad y la fiabilidad de los mismos.

- Generalidad. De media, el 71 % de las instancias objetivo están cubiertas por cada patrón, tal y como se puede observar con el valor de TPR. Este valor es muy interesante para los expertos ya que les proporciona una visión muy amplia del comportamiento general de sus clientes, por lo que se pueden tomar mejores decisiones.

- Fiabilidad. En primer lugar, se puede observar que todos los patrones extraídos son EPs porque su valor de GR es 1. Además, el ratio entre TPR y FPR de los EPs extraídos es elevado, por lo que estos patrones tienen un alto poder discriminativo. Finalmente, el nivel de confianza de los EPs extraídos es lo suficientemente alto para confiar en el conocimiento extraído, máxime cuando estos patrones contienen un nivel de generalidad tan elevado. Por lo tanto, los patrones extraídos contienen descripciones fiables que permiten al experto extraer conclusiones robustas de los mismos.

Complementando al estudio previo, se han obtenido aquellos EPs más repetidos a lo largo del flujo de datos junto con su valor de calidad promedio. El objetivo de este estudio es la creación de perfiles de cliente más generales de modo que se puedan crear soluciones a medio y largo plazo.

Tabla 4.3: Patrones más repetidos para cada clase y su valor de calidad promedio a lo largo del flujo de datos de los taxis de Nueva York.

Patrón	Apariciones	n_v	WRACC	CONF	GR	TPR	FPR
R_1: SI DOBorough = (Downtown, Midtown, Central Park, Uppertown) ENTONCES BAJO [0,15)	4633	1	0.76	0.86	2.29	0.93	0.42
R_2: SI DOBorough = (Downtown, Midtown, Central Park) ENTONCES BAJO [0,15)	3340	1	0.78	0.88	2.97	0.88	0.32
R_3: SI DOBorough = (Uppertown, Other, Central Park) ENTONCES MEDIO [15,30)	448	1	0.71	0.59	4.03	0.59	0.16
R_4: SI payment_type = credit_card & tip_amount = LL_3 ENTONCES MEDIO [15,30)	199	2	0.71	0.84	15.10	0.45	0.03
R_5: SI PUBorough = (Central Park, Midtown, Downtown, Other) ENTONCES ALTO [30, ∞)	630	1	0.85	0.23	16.72	0.78	0.08
R_6: SI payment_type = credit_card & tip_amount = (LL_4, LL_6) ENTONCES ALTO [30, ∞)	341	2	0.87	0.38	33.47	0.76	0.02

En la Tabla 4.3 se presentan los dos reglas más repetidas de cada clase a lo largo del flujo de datos. En general la calidad de los patrones es alta, destacándose los altos niveles de GR y WRAcc. Esto significa que los patrones extraídos son altamente discriminativos, de modo que se puede confiar en la fiabilidad de los patrones extraídos.

De estos resultados se pueden extraer diversas conclusiones. Por ejemplo, se puede observar que muchos de los viajes más baratos se realizan dentro del área de Manhattan dedicada a los negocios. Para aquellas tarifas de valor medio, el destino usual de los clientes son las zonas residenciales o de ocio de Manhattan. Por otro lado, para los viajes más caros, los clientes son recogidos en las zonas donde la gente trabaja. Por lo tanto, de este resultado se puede observar claramente el día a día de las personas, donde van al trabajo y luego vuelven a casa. Finalmente, es importante destacar la presencia del pago con tarjeta en aquellos viajes con tarifas medias o altas.

CAPÍTULO 5

Conclusiones y publicaciones relacionadas

En este capítulo se resumen los resultados obtenidos en esta memoria y se extraen las conclusiones del trabajo realizado. Asimismo, se comentan aspectos relativos a trabajos futuros relacionados con la línea de este trabajo. Adicionalmente, se presentan algunas publicaciones relacionadas con el desarrollo de esta memoria.

5.1. Conclusiones

El desarrollo de este algoritmo es el fruto de la amplia revisión bibliográfica realizada en el capítulo 2 sobre minería de flujo de datos y EPM la cual nos ha proporcionado los diferentes enfoques realizados por la comunidad científica y un amplio conocimiento para poder aplicar nuevos conceptos de manera adecuada.

En el capítulo 3 se presenta un nuevo modelo para la extracción de patrones emergentes bajo el contexto de la minería de flujo de datos con el que se pretende extraer conocimiento de manera continuada. Asimismo, se presenta un estudio experimental que demuestra la adaptabilidad del método a los cambios de concepto y la calidad de los patrones obtenidos.

FEPDS procesa el flujo de datos por medio de bloques de datos en los cuales las instancias son recogidas. Una vez existe un bloque de datos terminado, el algoritmo de aprendizaje extrae patrones emergentes que describen el fenómeno que subyace en ese bloque. En concreto, el algoritmo propuesto se basa en una estrategia de adaptación al cambio ciega con reentrenamiento en el que un nuevo modelo se extrae desde cero para cada uno de los bloques de datos que llegan al sistema. FEPDS utiliza como algoritmo de aprendizaje un algoritmo evolutivo multi-objetivo capaz de extraer conjuntos de patrones con una buen balance entre la simplicidad del conjunto y su fiabilidad. Este método de aprendizaje hace uso del conocimiento extraído para bloques nuevos por medio de una estructura de memoria basada en una ventana deslizante con el objetivo de adaptarse mejor a los cambios. Finalmente, FEPDS hace uso de una estrategia de evaluación *test-then-train* donde el modelo de conocimiento está continuamente siendo evaluado con el objetivo de determinar su calidad.

El algoritmo propuesto ha sido probado en un amplio estudio experimental en donde se muestra que el método es capaz de adaptarse correctamente a cambios de concepto de tipo abrupto y gradual sin decrementar su rendimiento en términos de tiempo de ejecución y calidad. Además, la calidad del conocimiento extraído es alta, destacando su alta cobertura y su alta fiabilidad.

Por lo tanto, se concluye que la propuesta presentada es una buena alternativa para la extracción eficaz y eficiente de patrones emergentes en entornos de minería de flujos de datos.

En el capítulo 4 se presenta un estudio con un flujo de datos real con el objetivo de analizar en tiempo real el perfil de los clientes que utilizan los taxis amarillos de la ciudad de Nueva York durante el año 2017.

En general, los resultados extraídos por el algoritmo FEPDS son de gran calidad. Destacando una gran capacidad de generalidad, donde se cubren un 78 % de los ejemplos con altos niveles de confianza, lo que permite extraer conocimiento muy interesante a los expertos, sobre todo en aquellas tarifas con un coste medio o bajo. Asimismo, un estudio de aquellos patrones más recurrentes se ha llevado a cabo donde se destacan comportamientos ligeramente recurrentes como el pago con tarjeta en tarifas altas o medias, o los viajes diarios al trabajo de los personas que viven en la ciudad de Nueva York.

5.2. Publicaciones relacionadas

Fruto del desarrollo de este trabajo, en esta sección se presentan diferentes trabajos relacionados que han sido publicados en congresos o revistas internacionales indexadas en *Journal Citation Reports*.

- GARCÍA-VICO, A. M.; CARMONA, C. J.; GONZÁLEZ, P. y DEL JESUS, M. J.: «MOEA-EFEP: Multi-Objective Evolutionary Algorithm for Extracting Fuzzy Emerging Patterns». *IEEE Transactions on Fuzzy Systems*, 2018, **26(5)**, pp. 2861 – 2872.

  En este trabajo se presenta el algoritmo MOEA-EFEP, el cual es un algoritmo evolutivo multi-objetivo para la extracción de patrones emergentes. Este trabajo ha sido fruto del profundo trabajo de revisión sobre minería de patrones emergentes.

- GARCÍA-VICO, A.; CARMONA, C. J.; GONZÁLEZ, P. y DEL JESUS, M. J.: «Una primera aproximación para la extracción de patrones emergentes en flujos continuos de datos». En: *Proc. of the XVIII Conferencia de la Asociación Española para la Inteligencia Artificial (XVIII CAEPIA)*, pp. 1093–1098, 2018.

  En este trabajo se presenta una versión preliminar del algoritmo FEPDS, en donde se comprueban diferentes funciones de evaluación y medidas objetivo, así como el tamaño óptimo de bloque. Este trabajo recibió además un premio *ex-aequo* al mejor trabajo del II Workshop en Big Data y Analisis de datos Escalable (BigDADE) en el XVIII Congreso de la Asociación Española para la Inteligencia Artificial (CAEPIA).

- GARCÍA-VICO, A.; CARMONA, C. J.; GONZÁLEZ, P. y DEL JESUS, M. J.: «FEPDS: A proposal for the Extraction of Fuzzy Emerging Patterns in Data Streams». *IEEE Transactions on Fuzzy Systems*, Submitted.

  En este trabajo se presenta el algoritmo FEPDS desarrollado en este trabajo. Actualmente, el trabajo ha sido sometido para su revisión.

5.3. Trabajos futuros

Los trabajos futuros que se abren tras la realización de este proyecto son:

- Realización de nuevas propuestas algorítmicas para la extracción de patrones emergentes en flujos de datos de manera más eficiente y/o con mejor calidad. Para ello, se pueden abordar diferentes casuísticas: modificando el esquema de adaptación del aprendizaje a un enfoque supervisado u *online*, utilizando nuevos algoritmos de aprendizaje basados en otras metaheurísticas o incluso mediante *ensembles*, entre otros.

- Especial importancia en este tipo de minería es la presencia de desbalanceo de clases o los llamados *small disjuncts*, los cuales pasan totalmente desapercibidos por ser tratados como ruido, por lo que no se obtienen resultados óptimos. A pesar de que existen patrones como los SJEPs que nos permitirían obtener estos patrones, estos no son lo suficientemente potentes ya que añaden una gran cantidad de ruido.

- Otro punto interesante es el tratamiento de otras problemáticas intrínsecas a la minería de flujo de datos, como por ejemplo la aparación o desaparación de clases y/o variables a lo largo del tiempo que hacen que el rendimiento decaiga. Estos aspectos, hasta donde sabemos, no han sido analizados para minería de patrones emergentes.

- Uno de los principales objetivos de la minería de patrones emergentes es la búsqueda de tendencias emergentes. A esto se le conoce en la literatura como minería de cambio (*change mining*). A pesar del posible interés que puede suscitar esta técnica, los desarrollos realizados en la literatura son escasos y con un gran margen de mejora.

- Finalmente, a día de hoy la cantidad de datos que se genera es tan grande que en algunos casos es necesaria la realización de procesamiento distribuido de flujos de datos con el objetivo de poder abordarlos eficientemente y ser escalables con el paso del tiempo. A pesar de que actualmente existen herramientas como Apache Spark o Apache Flink que nos permiten este tipo de procesamiento, en minería de patrones emergentes todavía no existen desarrollos de este tipo.

APÉNDICE A

Publicación sometida a IEEE Transactions on Fuzzy Systems relativa al algoritmo FEPDS

FEPDS: A Proposal for the Extraction of Fuzzy Emerging Patterns in Data Streams

Ángel M. García-Vico, Cristóbal J. Carmona, Pedro González, Huseyin Seker and María J. del Jesus

***Abstract*—Nowadays, most data is generated by devices that produce data continuously. These kinds of data can be categorised as data streams and valuable insights can be extracted from them. In particular, the insights extracted by emerging patterns are interesting in a data stream context as easy, fast, reliable decisions can be made. However, their extraction is a challenge due to the necessary response time, memory and continuous model updates.**

In this paper, an approach for the extraction of emerging patterns in data streams is presented. It processes the instances by means of batches following a retraining approach. The learning algorithm is an evolutionary fuzzy system where previous knowledge is employed in order to adapt to concept drift. A wide experimental study has been performed in order to show both the suitability of the approach in combating concept drift and the quality of the knowledge extracted. Finally, the proposal is applied to a case study related to the continuous determination of the profiles of New York City cab customers according to their fare amount, in order to show its potential.

***Index Terms*—Emerging pattern mining, Data stream mining, Evolutionary fuzzy systems, Multi-objective evolutionary algorithms.**

I. Introduction

THE data continuously sent by the devices surrounding us can be categorised as a data stream [1]. An analysis of such data could provide valuable insights into our activities and environments. Representative examples of these kinds of data are sensor networks, energy management and telecommunications [2]. In this kind of application one of the main characteristics is that data is less relevant as it becomes older. Therefore, a continuous analysis of data is better than a traditional static one. However, the characteristics of data streams pose several challenges for the development of methods. Among others, the most relevant constraints in data stream mining are the continuous updating and adaptation of the learning model as data arrive as underlying distributions can change over time. This is known in the literature as concept drift [1]. Moreover, the speed at which data arrives could be very high so the update of the model must be as fast as possible.

Emerging Pattern Mining (EPM) [3], [4] is a data mining technique framed within supervised descriptive rule discovery (SDRD) [5]. This task is half-way between descriptive and predictive inductions and attempts to describe relationships in data with respect to a given property of interest. In particular, EPM tries to describe discriminative relationships between different values of a property of interest or the description of emerging behaviour in data. EPM has been successfully applied in several fields such as management [6], disease detection [7], bioinformatics [8] and others [9]. Recently, several approaches based on evolutionary fuzzy systems (EFSs) have been proposed [10], [11] whose trade-off between the generality and reliability of the patterns extracted is improved with respect to other approaches developed for EPM [4]. In this way, a better description of the underlying phenomena in data is provided. This is quite relevant for data stream mining, as an easy, accurate description of what is going on within data is necessary in order to make decisions as fast as possible. However, to the best of our knowledge the extraction of emerging patterns (EPs) in data streams environments has not been tackled so far.

This paper presents an approach for the extraction of Fuzzy EPs in Data Streams (FEPDS), where its main components and working scheme are shown. The quality of the patterns extracted by FEPDS is analysed in a wide experimental study with a set of 94 synthetic datasets together with an analysis of the adaptation to the concept drift. In addition, the usefulness of FEPDS is demonstrated against a real dataset. In particular, the aim for that problem is the continuous description of the characteristics of New York City yellow cab customers with respect to the amount of money spent on their journeys, in order to improve the service.

This paper is organised as follows: Firstly, the main concepts related to concept drift, data streams and EPM are briefly described in Section II. Next, Section III presents the FEPDS algorithm, its components and its characteristics. Section IV presents the experimental framework and the analysis of the results. Section V presents the case study and its results. Finally, the conclusions are shown in Section VI.

II. Related work

In this section, the main related concepts in this paper are described such as concept drift (Section II-A), data streams with concept drift (Section II-B), and finally, fuzzy EPM (Section II-C).

A.M. García-Vico, C.J. Carmona, P. González and M.J. del Jesus are with the Andalusian Research Institute on Data Science and Computational Intelligence, University of Jaén, 23071 Jaén (Spain) (e-mail: {agvico|ccarmona|pglez|mjjesus}@ujaen.es)

Huseyin Seker is with the Department of Computer and Information Sciences, University of Northumbria, NE1 8ST, Newcastle (United Kingdom) (e-mail: huseyin.seker@notrhumbria.ac.uk)

This work was supported by the Spanish Ministry of Economy and Competitiveness under the project TIN2015-68454-R (FEDER Founds) and FPI 2016 Scholarship reference BES-2016-077738 (FEDER Founds). This paper was written in the Department of Computer and Information Sciences, Faculty of Engineering and Environment of The Northumbria University at Newcastle, Newcastle-Upon-Tyne, during the research visit of A.M. García-Vico. Thanks to Prof. H. Seker for the invitation.

A. Concept drift

Concept drift occurs when a non-stationary target concept is modified in the environment. Formally, a real drift is defined by a change in time t with respect to a time $t + \delta$ on the conditional probability of the classes given an instance x, i.e., $P(C|x)^t \neq P(C|x)^{t+\delta}$ [12]. Specifically, two types of drift are defined [13] considering both cause and effect of the probability distribution associated to the data (instances and classes):

- *Real drift*. It is considered when the probability distribution is modified with respect to the class but the incoming data probability could not be modified. The change could be visible or not from the data distribution.
- *Virtual drift*. It can be considered as incomplete data representation or changes in data distribution occurring without affecting the concept, amongst other factors.

The technology and their properties associated to the generation of stream data cause changes in data distribution over time. These changes can be reviewed in different ways. They can be classified as: abrupt, where a sudden drift in data occur instantly; incremental, where data changes gradually over time; gradual, which is a type of incremental drift with changes in class distribution too; and recurrent where concepts occur in a specific order again [14].

In a general way, it is important to remark that data stream mining should be able to work from evolving streams, regardless of classification or concept drift type. Therefore, concept drift must be analysed and taken into account for the design and development of data streaming algorithms.

B. Data streams with concept drift

A data stream is an unbounded, ordered sequence of instances that arrive at the system throughout time at a variable speed [15]. This simple definition of a data stream together with drifting concepts produces a huge amount of differences with respect to classical static datasets in data mining that must be taken into account. For example, their structures should be adapted with respect to the new incoming data in order to provide the best performance and a quick response [13]. Similarly, constraints arise and must be taken into consideration [16], [1]. For instance, there are space constraints because it is impossible to store the whole data stream in memory [17]. Finally, concept drift produces a significant decrease in the performance of the learning algorithms [18]. Therefore, mechanisms for adapting the learner to these changes must be developed.

These aspects lead to online adaptive learning whereby data arrive in the system and a prediction value is obtained with the current model. In a later stage, the estimation of this prediction is analysed and the model can be modified or confirmed. In this way, data stream mining algorithms should determine how the instances will be processed, how they will learn from these instances in order to handle the concept drift and how the quality of the model will be determined. Thus, four important elements are defined in [12], [19]:

1) *Memory*. This is the element in charge of processing the new information and forgetting the old one. For processing new data these elements can be found: *sequential* or *online*, where the instances arrive one by one and are processed by the learning algorithm as soon as they are available (this is known as Online active learning [20]); and *Windowing* where informative data or summaries concerning model behaviour or data distribution are stored in a short memory. Usually, it is considered that the most recent instances are the most representative. These windows can be specific to the applied learner or for general purposes, i.e. for any learning algorithm. In addition, its size can be fixed or variable, and its position with respect to the stream can be based on the sliding window model [21] or on the landmark model [22].
2) *Learning process*. The learning process employed can be based on online learners, or based on a single adaptive learner. The latter usually incorporates forgetting mechanisms in order to discard old data. The idea is to mimic the way online learners work. These kinds of methods are interesting for controlling the complexity of the system in order to provide a fast response. In addition, ensembles can be employed which are able to handle several learners containing different concepts. Nevertheless, its complexity is higher.
3) *Change monitoring*. In this component all techniques and mechanisms which are able to detect concept drift or change within the distribution are considered. These methods can be based on monitoring supervised indicators such as accuracy, recall or sensitivity and specificity when feedback is immediately available. These methods are interesting for the detection of real drift. When feedback is delayed, it is interesting to use methods based on unsupervised indicators such as similarity in time and space of data or based on measuring the model complexity. These elements are also interesting for the detection of virtual drift.
4) *Learning adaptation*. This component is relevant in evolving data environments because it contains the modules and operators for analysing and updating the structures in order to maintain the quality of the knowledge extracted. Two main types of adaptations in the learner can be found: *Retraining* where a new model is built from scratch with respect to data collected recently; and *Incremental* where the structures and model are adapted using the data within a window. Moreover, it is important to remark that learning models can use different strategies for the adaptation of their structures, such as *Blind*, where the learner is triggered at regular intervals, so no change monitoring is needed; or reactive strategies where the learner is triggered when a change monitoring method throws an alarm signal, known as *Informed*.

Finally, the development of an evaluation strategy in order to determine the quality of the model is necessary. In traditional data mining, this can be determined by several mechanisms such as cross-validation and hold-out, amongst others [23]. However, these mechanisms require a finite dataset in order to split the data into the training and test partitions. Data

streams are unbounded so the use of these kinds of techniques is impossible. One of the most popular estimation techniques for the evaluation of data stream models is the well-known interleaved test-then-train method [24]. With this method, the model performs a prediction or an evaluation of the model using the new incoming instance or chunk. After that, once the real value of the class for the instance or chunk is known, the model is updated with it, i.e. it now belongs to the training data. By means of this estimation technique we are able to continuously evaluate the model. In addition, other factors such as performance or memory consumption over time should be taken into consideration.

C. Fuzzy emerging pattern mining

The EPM was defined by Dong and Li [3], [4] as the search for patterns whose support increases significantly from one dataset (D_1) to another (D_2). In particular, D_1 can be the dataset formed by examples that belongs to one class, and D_2 contains the examples that belong to the remaining classes. A pattern is considered as emerging if and only if its growth rate (GR) is greater than a given threshold ρ. The GR measure is defined as in Eq. 1

$$GR(x) = \begin{cases} 0, & IF\ Sup_{D_1}(x) = Sup_{D_2}(x) = 0, \\ \infty, & IF\ Sup_{D_1}(x) \neq 0 \wedge Sup_{D_2}(x) = 0, \\ \frac{Sup_{D_1}(x)}{Sup_{D_2}(x)}, & another\ case \end{cases} \tag{1}$$

where $Sup_{D_i}(x)$ is the support of the pattern x on dataset i (D_i). EPs are represented by means of conjunctions of attribute-value pairs, or attribute-value pairs in disjunctive normal form, which represents the discriminative characteristics they want to describe. For the determination of D_1 and D_2, these patterns are usually labelled with the class or the dataset they try to describe.

The main purpose of EPM is the extraction of the differentiating characteristics among classes or the description of emerging trends with respect to a property of interest.In recent years the use of fuzzy logic within SDRD is gaining special relevance with the development of different algorithms such as SDIGA [25], FEPM [26], NMEEFSD [27] and MOEA-EFEP [11], amongst others. The use of fuzzy logic allows us to handle numeric variables by means of fuzzy linguistic labels (LLs), avoiding the loss of information produced by a discretised representation and a knowledge description closer to human reasoning [28]. In addition, the majority of proposals are combined with evolutionary algorithms (EAs) [29], well known throughout the literature as EFSs [30]. These kinds of systems have been applied in several real-world applications [31], [32], [33], [34], [35]. For EPM, a detailed review where the most important EPM algorithms are classified according to the approach used to mine EPs is presented in [4].

In EPM, the determination of the descriptive characteristics of a pattern is based on the number of examples covered or not covered by the patterns which belong or do not belong to the class of the pattern. Thus, several quality measures can be used from the SDRD framework for the determination of a wide range of aspects. The most widely used quality measures in EPM are outlined in Table I [4] where p are examples correctly covered, n are incorrectly covered examples, P are examples of the class and N are examples that do not belong to the class.

TABLE I
QUALITY MEASURES USED IN EPM FOR THE DETERMINATION OF THE QUALITY OF A PATTERN

Name	Abbreviation	Formula
Confidence [36]	Conf	$\frac{p}{p+n}$
Unusualness [37]	WRAcc	$\frac{p+n}{P+N}\left(\frac{p}{p+n} - \frac{P}{P+N}\right)$
Growth Rate [3]	GR	$\frac{p \cdot N}{P \cdot n}$
Support Difference [37]	SuppDiff	$\frac{p}{P} - \frac{n}{N}$
True Positive Rate [38]	TPR	$\frac{p}{P}$
False Positive Rate [39]	FPR	$\frac{n}{N}$

III. FEPDS: Fuzzy Emerging Patterns Extraction from Data Stream

In this section the algorithm for the extraction of fuzzy EPs (fEPs) in data stream mining, called FEPDS, and its main characteristics are presented.

Breifly, the main ideas of the algorithm are: it is considered for the learning algorithm that instances are collected from the stream by means of batches of a predetermined size. The algorithm employs as learning method a multi-objective EFS for the extraction of fEPs. This learning method allows the extraction of high-quality fEPs with a good trade-off between generality and reliability. In addition, the use of fuzzy logic within the learning process produces robustness against slight changes, so a good adaptation to gradual drift can be achieved. FEPDS follows a blind strategy with retraining to update the model. Following this strategy, the learning method is triggered for each batch of data. This learning method also employs a specific memory structure in order to use previously extracted knowledge for biasing the current learning process based on the history. This memory is based on a sliding window which allows the storage of the previous models extracted. In this way, a good adaptation to gradual drift can be achieved as the method is regularly executed. Finally, the evaluation of the model follows a test-then-train approach which allows the continuous evaluation of the current model.

Below, the main elements of FEPDS are depicted. Firstly, the memory component of FEPDS is presented in Section III-A. Next, in Section III-B the main concepts of the learning adaptation process for the algorithm are described. After that, the evolutionary learning approach of the FEPDS algorithm for extracting fEPs and its main characteristics are shown in Section III-C. Finally, the connections between the different components are presented in the operational scheme of FEPDS in Section III-D.

A. Memory component

In FEPDS it is assumed that data arrives in the system in the form of batches of data of a fixed length. For this purpose, a process is carried out in the algorithm for collecting these batches of data, as can be observed in Fig. 1.

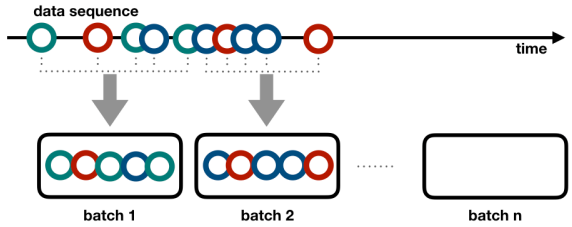


Fig. 1. Online phase of the FEPDS algorithm where data are collected in batch with a fixed length.

FEPDS uses a memory structure in order to store each set of fuzzy patterns (fPS) extracted by the learning algorithm when processing the different batches of data. It is a sliding window (SW) of a fixed size. In this way, the memory structure employs a first-in-first-out (FIFO) policy when the capacity of the window is full. In particular, the size of the SW is initialised for the storage of m fPSs, and it is assumed that knowledge extracted more than m data batches ago is old enough to be discarded for the complete evolutionary process. Therefore, the algorithm forgets old knowledge while it is able to remember the most recent for its use in the current learning process. It is important to remark that the patterns of the fPS included in (SW_t), together with the associated quality information for each fEP are queued as a single element in SW.

The information in SW is employed within the learning process. In particular, it is employed within the reward on the token-competition-based procedure which is detailed in Section III-C3. In this way, the procedure is able to bias the current learning in order to provide a good adaptation to the stream as it is influenced by good previously extracted patterns.

B. Learning adaptation process

The adaptation to the drift in FEPDS is based on a blind strategy with retraining where a new model is obtained for each batch. The employment of a blind scheme allows a good adaptation to gradual changes as it is continuously adapting to the changes. The algorithm initialises fPS and the SW structure as empty. Once the first batch is completed, data are continuously received for use in the following stages. Meanwhile, the evolutionary learning method obtains knowledge which is based on the current batch together with the previous insights stored in SW, if available, and this new fPS is incorporated into the SW structure. In this way, the SW is incrementally updated by means of adding the previous knowledge in the current evolutionary process. This is because these insights contain a valuable summary of the stream. In this way patterns that appeared frequently will have more weight in the evolutionary process, and patterns that do not appear should be gradually forgotten. The application of these mechanisms to previous knowledge avoids the inclusion of non-relevant information within the current evolutionary process.

Finally, it is very important to highlight that despite the fact that the algorithm employs adaptive learning without the detection of changes (blind strategy), the use of the SW brings the algorithm some insights about past knowledge and the behaviour of FEPDS with respect to the stream.

C. Multi-objective evolutionary learning approach

The learning algorithm within FEPDS is a multi-objective EFS (MOEA) for obtaining fEPs. The determination of the quality of the fEPs is necessary to properly guide the search process. This is determined by several objectives, calculated using quality measures such as those presented in Table I. In particular, the objectives employed in the evolutionary process to be improved are WRAcc and SuppDiff as they provide a good trade-off between generality and reliability. In addition, an elite population is employed in order to store the best fPS found so far for the current batch of data. However, the addition of information about the evolution of the stream is key to fitting the knowledge to the underlying phenomena which generate the data in order to handle issues related to the concept drift. In this way, when the elite population is being updated, a token-competition-based procedure is applied [40]. This procedure employs the information stored in SW for taking into consideration the evolution of the stream. In the following subsection, the main elements of the evolutionary learning algorithm of FEPDS are presented.

1) Pattern representation: The EFS uses a "chromosome = pattern" approach [41] whereby each individual in the population represents a potential pattern. In FEPDS, fuzzy logic is used for the representation of numeric variables by means of linguistic labels (LLs). The shape of these LLs is predetermined by means of triangular membership functions which have been successful in real-world applications in both EPM [42] and similar tasks such as subgroup discovery [43], [44], as can be observed in Fig. 2.

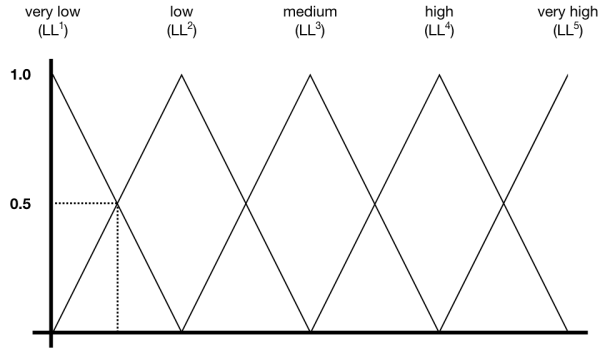


Fig. 2. Representation of a continuous variable with five linguistic labels.

The EFS extracts patterns following a disjunctive normal form (DNF) representation because it is a good knowledge representation for descriptive EPs [11]. DNF patterns are codified in the evolutionary algorithm by means of a bit-vector genotype whose length is equal to the total number of elements. The number of elements is determined by the

number of possible categories for nominal variables, while for numerical variables it is the number of LLs used. In addition, the consequent part is represented by an integer value which allows the extraction of patterns for all classes in a single execution. A fEP and its representation in FEPDS can be observed in Fig. 3.

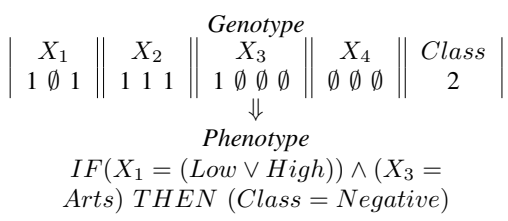


Fig. 3. Representation of a fuzzy DNF pattern with continuous and categorical variables in FEPDS.

2) Genetic operators: Although the evolutionary process is based on a retraining approach, the evolutionary learning algorithm employs an initialisation procedure which incorporates the fEPs that belong only to the previous fPS in the initial population. The aim is to incorporate knowledge about the current state of the stream in order to properly guide the search process in the current batch. After that, the remaining individuals are randomly initialised until the population is filled.

The genetic operators employed are the classical binary tournament selection [45], the multi-point crossover operator [46] and an oriented mutation operator. This mutation operator removes a variable of a pattern or randomly changes a gene with the same probability.

Finally, a reinitialisation operator is employed in order to avoid falling into a local maxima. In this procedure, the memory SW is employed when the Pareto front does not evolve, i.e. it is not able to cover new examples, during at least 25% of the total evaluations; and during the final stage it is used to obtain the final fPS. With this procedure the elite population is updated by means of the reward following a token-competition-based procedure which is described below.

3) Reward in the token-competition-based procedure: FEPDS introduces a function in order to take into consideration the underlying phenomena in the stream. These underlying phenomena are known by the patterns extracted in previous batches of the stream. Therefore, the idea of FEPDS is to reward the knowledge that represents the underlying phenomena of the stream as it provides relevant information about its evolution. It is important to remark that usually the older the pattern, the lesser the accuracy in the description of its current state. In this way, if a pattern was good a few moments ago, the probability of its being a good pattern again is very high. Nevertheless, this probability is lesser as time goes by. Therefore, less reward should be given to older patterns. Reward in the token-competition-based procedure is applied for the complete main population (P_g) when the token-competition-based procedure is applied to the reinitialisation process, when the elite population is updated. The main idea is to reward the patterns extracted in previous stages, as they contain not only relevant insights but the capacity to simplify the model as fewer patterns are needed to cover the examples space. Specifically, the operator incorporates SW with the function $SW\,(fP_i, j)$ that returns a value of one if the pattern fP_i is in the set of patterns fPS_j, or zero otherwise. A value is obtained for each individual as follows:

$$Div_t(fP_i) = WRAcc_t(fP_i) + \sum_{j=t-n}^{t} SW(fP_i, j) \cdot 2^{-(t-j)} \cdot WRAcc_j(fP_i) \quad (2)$$

where $WRAcc_t(fP_i)$ is the value of the unusualness measure in the current batch t for the pattern i, and $WRAcc_j(fP_i)$ is the value of unusualness for the batch j for the same pattern. As we have mentioned previously, the main idea is to add unusualness values for the fP which appears in previous stages within of the SW.

The complete population is ordered with the Div_t values (from high to low) and the token competition operator is applied afterwards [31]. In this way, individuals with the highest diversity values will exploit its niches by seizing as many tokens as they can. The operator allows us to obtain a fPS where all patterns cover at least one example of the dataset not yet covered by other stronger patterns. In order to complete the population of the next generation (P_{g+1}) the new individuals are generated through the guided reinitialization procedure [11], with the objective of searching into other promising areas of space so that the algorithm can cover all the examples.

The reward component incorporated into the algorithm allows an adaptation for the possible drift changes.

D. Operational scheme of FEPDS

In Fig. 4 a graphical representation of the working scheme of FEPDS is presented. In addition, Alg. 1 presents the pseudo-code of FEPDS in order to ease its understanding. Firstly, FEPDS uses a test-then-train evaluation approach (lines 6-8). The evolutionary learning algorithm extracts a fPS for each batch of data while the test-then-train approach allows us the evaluation of the previous insights gained against the current batch of data. Therefore, fPS_1 is evaluated with the second batch, fPS_2 with the third, and so on.

After that, following the blind adaptive learning strategy, the algorithm is executed for each batch of data. Once the batch is completed (lines 3-5), and the current fPS has been tested against this batch (line 7), the evolutionary learning process is triggered for the extraction of a new fPS following the retraining approach (lines 9-27). The evolutionary learning algorithm starts the initial population P_0 using the initialisation operator which makes use of the previous fPS (line 10). Next, P_0 is evaluated to determine its quality (line 11). Then the evolutionary process begins (lines 13-24), until a maximum number of generations is reached or a new batch is available. Within this process, P_g creates an offspring population Off_g by means of the application of the genetic operators (line 14), while Off_g is evaluated afterwards (line 15). These populations are joined together and the fast non-dominated

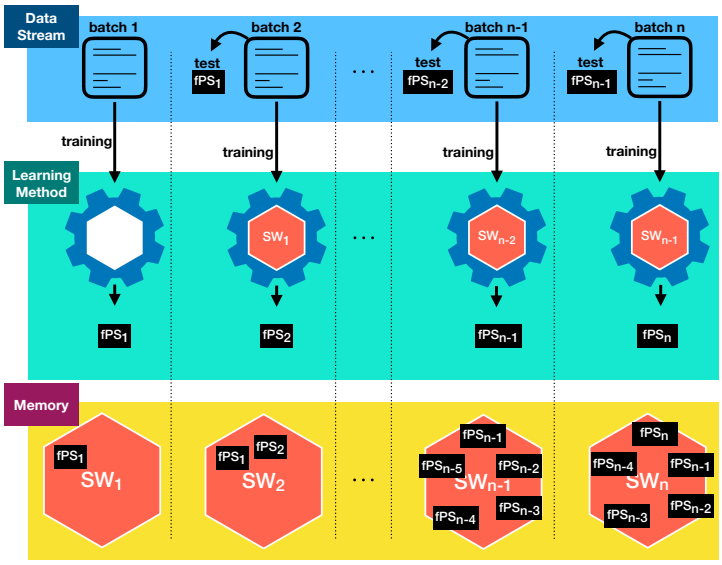


Fig. 4. General working schema of the FEPDS algorithm.

sorting algorithm [47] is applied (lines 16-17). Finally, the application of the reinitialisation is checked. If the population is stagnated, P_{g+1} is filled by means of the reward in the token-competition-based procedure (line 19), which makes use of SW. Otherwise, P_{g+1} is filled by introducing the first k whole fronts or the first k individuals of a front if there is no room for the whole front (line 21). At the end of the evolutionary process, the reward in the token-competition-based procedure is applied once again to filter overlapped patterns, boosting those ones which are able to remain in time (line 25). Finally, the final fPS extracted is stored in SW which follows a first-in-first-out (FIFO) strategy and fPS is returned (lines 26-27). It is important to remark that the whole process is repeated until the end of the data stream.

Algorithm 1 Operational scheme of the FEPDS algorithm.

1: $t \leftarrow 0$
2: **repeat**
3: **while** Batch size is less than n **do**
4: Collect data from the stream.
5: **end while**
6: **if** $fPS_t \neq \emptyset$ **then**
7: Test fPS_t against current batch.
8: **end if**
9: $g \leftarrow 0$
10: $P_g \leftarrow$ Initialisation(fPS_t)
11: Evaluate(P_g)
12: $t \leftarrow t + 1$
13: **while** $g < maxGens$ **and** no new batch is available **do**
14: $Off_g \leftarrow$ GeneticOperators(P_g)
15: Evaluate(Off_g)
16: $R_g \leftarrow P_g \cup Off_g$
17: $F \leftarrow$ DominanceSorting(R_g)
18: **if** ParetoFront(F) does not evolve **then**
19: $P_{g+1} \leftarrow$ RewardInTokenCompetition(F, SW)
20: **else**
21: $P_{g+1} \leftarrow$ FillByFronts(F)
22: **end if**
23: $g \leftarrow g + 1$
24: **end while**
25: $fPS_t \leftarrow$ RewardInTokenCompetition(F,SW)
26: Add fPS_t in SW
27: **return** fPS_t
28: **until** The end of the stream

IV. Experimental study

A complete experimental study is carried out in order to determine the quality of the proposed algorithm. The aim of this study is twofold: firstly, the adaptation of FEPDS to concept drift is analysed. After that, a second study where the quality of the knowledge is analysed is carried out. These studies are shown in this section. Firstly, the experimental framework is presented in Subsection IV-A. Then the method is analysed for concept drift adaptation in Subsection IV-B. Finally, the quality of the insights extracted is analysed in Subsection IV-C.

A. Experimental framework

In this section, the details of the experimental framework are shown in order to ease the reproducibility of this study. First, the datasets employed are presented. Next, the parameters employed by FEPDS are shown. Finally, the quality measures employed for determining the quality of the insights are

presented. For further information and tools to reproduce this study, please refer to our GitHub repository[1].

a) Datasets: For the first study, datasets were artificially generated using the MOA software [48]. Two sets of three datasets each one were generated, one set containing an abrupt drift and the other a gradual drift. In both cases, drifts occur every 50.000 instances. For gradual drift, the size of the window of change is 10.000. All datasets were generated using 500.000 instances, so exactly 9 changes were produced.

For the second study, a set of 94 artificial datasets with one million instances was generated using several streams generators from MOA. For each dataset five different streams were generated changing its random seed. In this way, results can be averaged in order to avoid biases.

b) Parameters: The same parameters were employed for both studies. The selected values have been widely used in the EAs for EPM. In this way, FEPDS runs for 60 generations. It uses 3 LLs for numerical variables. The population size per class is 50. The crossover and mutation probabilities are 0.8 and 0.1 respectively. The size of the sliding windows is 5 and the data chunk size is 2500.

c) Quality measures: The confidence is the conditional probability of the class, given the pattern [49]. When a concept drift occurs, the average confidence of the extracted fPS could be significantly affected. Therefore, the confidence is analysed in order to determine the behaviour of the algorithm with respect to concept drift in the first study. In addition, it is interesting to determine the execution time.

For the second study, the quality measures shown in the results tables are the ones related to the description of the main characteristics of descriptive patterns in EPM [4], i.e. the average number of patterns (n_r) and the average number of variables (n_v) needed to measure the conciseness of the model extracted; the average WRAcc needed to compute the novelty and reliability; the average GR, Conf and FPR needed to estimate the reliability; and the average TPR needed to calculate the generality. It is important to remark that the results shown are the average of five executions using different seeds in the stream generators.

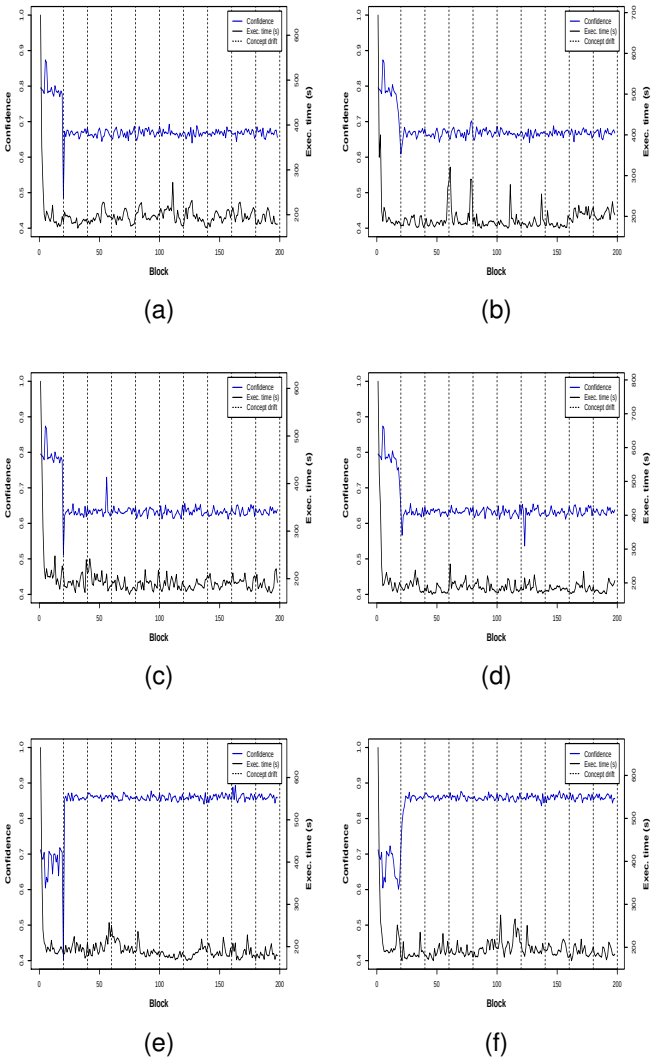


Fig. 5. Confidence and execution time performance of FEPDS on different datasets with concept drift. (a), (c), (e) are datasets with an abrupt drift, while (b), (d), (f) are datasets with a gradual drift.

B. Concept drift adaptation analysis

Fig 5 shows the performance of FEPDS against different concept drifts. In fact, Figs. 5a, 5c and 5e contain an abrupt drift while Figs. 5b, 5d and 5f present a gradual one. It is important to remark that each drift is marked with a dashed, vertical line on each plot.

In all the datasets analysed the performance after the first drift changes significantly. After this point the algorithm is able to recover rapidly from the drift. In general, the performance of the algorithm remains stable regardless of the concept drift. This fact can be produced by the employment of fuzzy logic on numerical variables. The use of fuzzy logic allows us to handle some uncertainty which is useful when the concept changes as it provides a tolerance level for small or gradual changes without affecting performance. In addition, the evolutionary process includes a learning stage that is able to provide a fast adaptation to the change. This tolerance level, together with the evolutionary process, allows the extraction of patterns with a good trade-off between their quality and their adaptability to changes.

Finally, it can be observed that the retraining approach is not significantly affected by concept drifts in terms of execution time. In fact, the execution time is in general close to 200ms, which is fast enough to provide a real-time response.

C. Quality analysis

Table II presents the average fEP quality across the data stream, together with the average complexity of the extracted fPS. From this table, an analysis of the different aspects of EPM is depicted below:

- Complexity of the model. This is measured by n_r and n_v. In general, the fPS extracted is simple. In fact, it

[1] https://github.com/SIMIDAT/FEPDS

TABLE II
AVERAGE RESULTS OF FEPDS.

n_r	n_v	WRACC	CONF	GR	TPR	FPR
2.38	3.62	0.75	0.78	0.99	0.69	0.18

is composed of almost one fEP per class, with a low number of variables. This simplicity allows a fast analysis of the insights extracted, which is key in the data stream mining context.

- Interest. This is determined by WRAcc. The results extracted are of great interest. This is due to the high trade-off between the generality, in terms of covered instances, and the reliability, according to the amount of correctly covered instances. Therefore, the fEPs could be interesting for the experts as they are a good approximation of the underlying phenomena in the data.
- Generality. This is determined by TPR. In general the fEPs extracted contains a high generalisation capacity as each one is able to cover 68% of all positive instances. Therefore, the insights extracted by the proposed algorithm concern a high number of target instances.
- Reliability. This aspect is determined by CONF, GR and FPR. 77% of instances covered by each fEP are covered correctly. In addition, the high GR percentage indicates that almost all the patterns extracted are EPs on test data. In addition, although the amount of FPR is high, its ratio to TPR determines that the average fEP extracted by the proposed method contains a high discriminative power and is trustworthy.

In EPM it is necessary to search for a good trade-off between three objectives: simplicity, generality and reliability. The average fPS extracted by FEPDS is simple enough to perform a fast analysis which is relevant in the data stream mining context. Although the model is simple, it does not concern its reliability as the patterns extracted contains a high discriminative power together with a high confidence. In addition, the fPS extracted is able to cover a large amount of target instances. Therefore, using the insights extracted experts are able to extract interesting, efficient conclusions about the underlying phenomena in data.

V. A CASE STUDY: DETERMINATION OF PROFILES OF NEW YORK CITY CAB CUSTOMERS ACCORDING TO THEIR FARE AMOUNT

One of the most popular images of New York City (NYC) is its yellow cab. The fleet is regulated by the NYC Taxi & Limousine Commission (NYC-TLC), which is formed of 13587 medallion cabs. One of the characteristics of these vehicles is that to be picked-up by one of them the passenger must hail the driver in the street. Therefore, no previous booking is needed before travelling.

In this case study the characteristics of passengers with respect to their fare amount is analysed. The idea is to continuously analyse the behaviour of people in order to improve the service.

A. Characteristics of data

The data analysed consist of all the records in 2017 of the journeys made by the yellow cabs[2]. Moreover, basic data formatting and data cleaning procedures have been carried out such as removing outliers and inconsistencies, unnecessary variables, and so on. To reproduce this preprocessing, please refer to our repository.

The final dataset is composed of almost 100 million instances with 10 variables each. It contains a class value that represents the amount of money spent on each journey. As can be observed, the amount of money spent is a numerical (num) variable, therefore it must be discretised in order to be handled by FEPDS. This class was discretised in order to be handled by FEPDS. This process was carried out by means of an expert criterion, determining the following intervals: [0, 15), [15, 30), [30, ∞], which correspond to low, medium and high fares. Finally, a brief description of each nominal (nom) and numerical (num) variable in the dataset is presented in Table III.

TABLE III
DESCRIPTION OF VARIABLES USED IN NYC-TLC TAXIS DATA.

Name	Type	Description
passenger_count	num	Passengers in the journey
RateCodeID	nom	The rate in effect during the journey
store_and_fwd_flag	nom	Connection with the server
payment_type	nom	The type of payment made
extra	nom	Extra surcharges (Rush & overnight)
tip_amount	num	The amount of money spend on tips
tolls_amount	num	The amount of money spend on tolls
PUBorough	nom	The pick-up zone
DOBorough	nom	The drop-off zone
total_amount	nom	The total fare of the journey

B. Extraction of emerging patterns in NYC-TLC data stream by FEPDS

TABLE IV
AVERAGE RESULTS OF FEPDS ON THE NYC TAXIS DATASET.

n_r	n_v	WRACC	CONF	GR	TPR	FPR
3.00	3.63	0.78	0.62	1.00	0.72	0.16

Table IV presents the average quality results for each individual fEP for each data chunk analysed, except for n_r and n_v which correspond the whole set of patterns extracted for that data chunk. An analysis of these results is presented below:

- Complexity of the models. FEPDS constantly returns three fEPs, which means that the method extracts only one fEP for each class. On the other hand, the average number of variables of each pattern is low. Therefore, experts are able to perform a fast, easy analysis of the knowledge extracted.
- Interest. The results extracted show an excellent average WRAcc value of 0.77. This means that the patterns

[2]https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page

extracted contain an excellent trade-off between the coverage of the target instances and their reliability. In fact, this interest can be observed in the high TPR value together with an acceptable confidence value. This fact is due to the use of the reward in the TC-based procedure evaluation function making it possible to keep those relevant patterns while the evolutionary process searches for other interesting knowledge in order to adapt to potential changes. Therefore, the patterns extracted by FEPDS are very interesting for the expert because they present an excellent trade-off between generality and reliability.

- Generality. On average, 71% of the target instances are covered for each pattern, as can be observed from the TPR. This is quite relevant for the experts as the knowledge extracted is able to provide a wide view of the general behaviour of customers, so better decisions can be carried out.
- Reliability. Firstly, it can be observed that all the patterns extracted are EPs because the GR value is 1. In addition, the ratio between the TPR and the FPR of the extracted fEPs is high, so they have a great discriminative power. On the other hand, the confidence level is high enough to trust on the knowledge extracted, especially when the level of generality is very high. Therefore, the patterns extracted present reliable descriptions which allows the experts to extract robust conclusions from them.

Additionally, the most repeated patterns extracted throughout the stream are presented together with their average quality measures. The idea is to determine recurrent patterns in the stream analysed so a general profile can be extracted.

Table V presents the two most repeated patterns throughout the stream for each class. In general the quality of these patterns is very good, highlighting the high levels of GR and WRACC, which means that the patterns are highly discriminative. In this way, we can trust in the reliability of the patterns extracted.

Several conclusions can be extracted from this table. For example, it can be observed that many of the cheapest journeys are produced within the working area of Manhattan. For medium fares, it can be observed that the destinations are usually residential or leisure areas of Manhattan. On the other hand, for the highest fares it can be observed that users are usually picked up in those areas where people usually work. Therefore, from these results the day-to-day lives of people can be observed, where first they go to their workplace and then they return home. Finally, it is important to remark on the presence of the payment by card on those journeys with medium and high fares.

VI. Concluding remarks

This paper presents an algorithm for extracting fEPS in data stream environments. FEPDS processes stream by means of a batch strategy where instances are collected. Once a batch is completed, the learning method of the algorithm extracts fPS associated it. Specifically, the algorithm is based on a blind strategy with retraining whereby a new model is obtained for each batch. In addition, FEPDS uses a learning method based on a multi-objective evolutionary algorithm able to extract pattern models with a very good trade-off between the simplicity of the model and its reliability. This learning method employs the fPSs obtained in previous stages by means of a memory structure based on a sliding window (SW) in order to reward patterns that sometimes appear in the window. Finally, the algorithm uses a test-then-train evaluation where the model is continuously evaluated against unseen instances in order to determine the quality of the knowledge extracted.

FEPDS has been tested in an experimental study. Firstly, the adaptability of the proposed algorithm to different kinds of concept drifts was analysed. The conclusion is that FEPDS is able to adapt the model properly with respect to abrupt and gradual drift without decreasing performance in terms of execution time. In addition, the quality of the insights extracted by FEPDS contains an interesting ratio between the coverage of target instances and their reliability, together with a fPS that can be analysed easily.

Finally, a case study was carried out. In particular, the profile of the customers of the New York City taxis according to their fare amount was analysed. In general, the results extracted by FEPD are of a high quality. In particular, the generality of each individual pattern is highlighted. Moreover, an analysis of recurrent patterns was carried out, from which slightly recurrent behaviours such as payment by card on medium or high fares and the daily travel to work of new yorkers have been extracted.

References

[1] J. Gama, *Knowledge discovery from data streams*. CRC Press, 2010.

[2] I. Žliobaitė, M. Pechenizkiy, and J. Gama, "An overview of concept drift applications," in *Big Data Analysis: New Algorithms for a New Society*. Springer, 2016, pp. 91–114.

[3] G. Dong and J. Li, "Efficient mining of emerging patterns: Discovering trends and differences," in *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 1999, pp. 43–52.

[4] A. M. García-Vico, C. J. Carmona, D. Martín, M. García-Borroto, and M. J. del Jesus, "An overview of emerging pattern mining in supervised descriptive rule discovery: Taxonomy, empirical study, trends and prospects," *WIREs: Data Mining and Knowledge Discovery*, vol. 8, no. 1, 2018.

[5] P. Kralj-Novak, N. Lavrac, and G. I. Webb, "Supervised Descriptive Rule Discovery: A Unifying Survey of Constrast Set, Emerging Pattern and Subgroup Mining," *Journal of Machine Learning Research*, vol. 10, pp. 377–403, 2009.

[6] G. Li, R. Law, H. Q. Vu, J. Rong, and X. R. Zhao, "Identifying emerging hotel preferences using emerging pattern mining technique," *Tourism management*, vol. 46, pp. 311–321, 2015.

[7] Y. Yu, K. Yan, X. Zhu, and G. Wang, "Detecting of PIU Behaviors Based on Discovered Generators and Emerging Patterns from Computer-Mediated Interaction Events," in *Proc. of the 15th International Conference on Web-Age Information Management*, vol. 8485, 2014, pp. 277–293.

[8] J.-P. Métivier, A. Lepailleur, A. Buzmakov, G. Poezevara, B. Crémilleux, S. O. Kuznetsov, J. L. Goff, A. Napoli, R. Bureau, and B. Cuissart, "Discovering structural alerts for mutagenicity using stable emerging molecular patterns," *Journal of chemical information and modeling*, vol. 55, no. 5, pp. 925–940, 2015.

[9] C.-H. Weng and C.-K. H. Tony, "Observation of sales trends by mining emerging patterns in dynamic markets," *Applied Intelligence*, pp. 1–15, 2018.

[10] A. M. García-Vico, P. González, M. J. del Jesus, and C. J. Carmona, "A first approach to handle emergining patterns mining on big data problems: The evaefp-spark algorithm," in *IEEE International Conference on Fuzzy Systems*, 2017, pp. 1–6.

TABLE V
MOST REPEATED PATTERNS FOR EACH CLASS AND THEIR AVERAGE QUALITY THROUGHOUT THE NYC-TLC DATA STREAM.

Pattern	Repetitions	n_v	WRACC	CONF	GR	TPR	FPR
R_1: IF DOBorough = (Downtown, Midtown, Central Park, Uppertown) THEN LOW [0,15)	4633	1	0.76	0.86	2.29	0.93	0.42
R_2: IF DOBorough = (Downtown, Midtown, Central Park) THEN LOW [0,15)	3340	1	0.78	0.88	2.97	0.88	0.32
R_3: IF DOBorough = (Uppertown, Other, Central Park) THEN MEDIUM [15,30)	448	1	0.71	0.59	4.03	0.59	0.16
R_4: IF payment_type = credit_card & tip_amount = LL_3 THEN MEDIUM [15,30)	199	2	0.71	0.84	15.10	0.45	0.03
R_5: IF PUBorough = (Central Park, Midtown, Downtown, Other) THEN HIGH [30, ∞)	630	1	0.85	0.23	16.72	0.78	0.08
R_6: IF payment_type = credit_card & tip_amount = (LL_4, LL_6) THEN HIGH [30, ∞)	341	2	0.87	0.38	33.47	0.76	0.02

[11] A. M. García-Vico, C. J. Carmona, P. González, and M. J. del Jesus, "Moea-efep: Multi-objective evolutionary algorithm for extracting fuzzy emerging patterns," *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 5, pp. 2861 – 2872, 2018.

[12] J. a. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Comput. Surv.*, vol. 46, no. 4, pp. 44:1–44:37, 2014.

[13] B. Krawczyk, L. L. Minku, J. a. Gama, J. Stefanowski, and M. Woźniak, "Ensemble learning for data stream analysis: A survey," *Information Fusion*, vol. 37, pp. 132–156, 2017.

[14] A. Sayuri-Iwashita and J. Papa, "An overview on concept dripft learning," *IEEE Access*, vol. 7, pp. 1532–1547, 2019.

[15] M. M. Gaber, "Advances in data stream mining," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 2, no. 1, pp. 79–85, 2012.

[16] A. Bifet, "Adaptive learning and mining for data streams and frequent patterns," Ph.D. dissertation, Universitat Politècnica de Catalunya, 2009.

[17] S. Ramírez-Gallego, B. Krawczyk, S. García, M. Wozniak, and F. Herrera, "A survey on data preprocessing for data stream mining: Current status and future directions," *Neurocomputing*, vol. 239, pp. 39–57, 2017.

[18] G. I. Webb, L. K. Lee, B. Goethals, and F. Petitjean, "Analyzing concept drift and shift from sample data," *Data Mining and Knowledge Discovery*, vol. 32, no. 5, pp. 1179–1199, 2018.

[19] I. Khamassi, M. Sayed Mouchaweh, M. Hammami, and K. Ghédira, "Discussion and review on evolving data streams and concept drift adapting," *Evolving Systems*, vol. 9, no. 1, pp. 1–23, 2018.

[20] E. Lughofer, "On-line active learning: A new paradigm to improve practical useability of data stream modeling methods," *Information Sciences*, pp. 356–376, 2017.

[21] G. Hulten, L. Spencer, and P. Domingos, "Mining time-changing data streams," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2001, pp. 97–106.

[22] J. Gama and G. Castillo, "Learning with local drift detection," in *Proceedings of the Second International Conference in Advanced Data Mining and Applications, ADMA, Xi'an, China, August 14-16*, 2006, pp. 42–55.

[23] N. Japkowicz and M. Shah, *Evaluating learning algorithms: a classification perspective*. Cambridge University Press, 2011.

[24] A. Wald, *Sequential analysis*. Courier Corporation, 1973.

[25] M. J. del Jesus, P. González, F. Herrera, and M. Mesonero, "Evolutionary Fuzzy Rule Induction Process for Subgroup Discovery: A case study in marketing," *IEEE Transactions on Fuzzy Systems*, vol. 15, no. 4, pp. 578–592, 2007.

[26] M. García-Borroto, J. Martínez-Trinidad, and J. Carrasco-Ochoa, "Fuzzy emerging patterns for classifying hard domains." *Knowledge and Information Systems*, vol. 28, no. 2, pp. 473–489, 2011.

[27] C. J. Carmona, P. González, M. J. del Jesus, and F. Herrera, "NMEEF-SD: Non-dominated Multi-objective Evolutionary algorithm for Extracting Fuzzy rules in Subgroup Discovery," *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 5, pp. 958–970, 2010.

[28] E. Hüllermeier, "Fuzzy methods in machine learning and data mining: Status and prospects," *Fuzzy Sets and Systems*, vol. 156, no. 3, pp. 387–406, 2005.

[29] D. E. Goldberg, *Genetic Algorithms in search, optimization and machine learning*. Addison-Wesley Longman Publishing Co., Inc., 1989.

[30] F. Herrera, "Genetic fuzzy systems: taxomony, current research trends and prospects," *Evolutionary Intelligence*, vol. 1, pp. 27–46, 2008.

[31] C. J. Carmona, V. Ruiz-Rodado, M. J. del Jesus, A. Weber, M. Grootveld, P. González, and D. Elizondo, "A fuzzy genetic programming-based algorithm for subgroup discovery and the application to one problem of pathogenesis of acute sore throat conditions in humans," *Information Sciences*, vol. 298, pp. 180–197, 2015.

[32] A. Starkey, H. Hagras, S. Shakya, and G. Owusu, "A multi-objective genetic type-2 fuzzy logic based system for mobile field workforce area optimization," *Information Sciences*, vol. 329, pp. 390–411, 2016.

[33] O. Castillo, L. Cervantes, J. Soria, M. Sánchez, and J. Castro, "A generalized type-2 fuzzy granular approach with applications to aerospace," *Information Sciences*, vol. 354, pp. 165–177, 2016.

[34] M. Antonelli, D. Bernardo, H. Hagras, and F. Marcelloni, "Multiobjective evolutionary optimization of type-2 fuzzy rule-based systems for financial data classification," *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 2, pp. 249–264, 2017.

[35] M. Babaei and M. Sheidaii, "Desirability-based design of space structures using genetic algorithm and fuzzy logic," *International Journal of Civil Engineering*, vol. 15, no. 2, pp. 231–245, 2017.

[36] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery: an overview," in *Advances in knowledge discovery and data mining*. Menlo Park, CA, USA: AAAI/MIT Press, 1996, pp. 1–34.

[37] C. J. Carmona, M. J. del Jesus, and F. Herrera, "A Unifying Analysis for the Supervised Descriptive Rule Discovery via the Weighted Relative Accuracy," *Knowledge-Based Systems*, vol. 139, pp. 89–100, 2018.

[38] W. Kloesgen, "Explora: A Multipattern and Multistrategy Discovery Assistant," in *Advances in Knowledge Discovery and Data Mining*. Menlo Park, CA, USA: American Association for Artificial Intelligence, 1996, pp. 249–271.

[39] D. Gamberger and N. Lavrac, "Expert-Guided Subgroup Discovery: Methodology and Application," *Journal Artificial Intelligence Research*, vol. 17, pp. 501–527, 2002.

[40] K. S. Leung, Y. Leung, L. So, and K. F. Yam, "Rule Learning in Expert Systems Using Genetic Algorithm: 1, Concepts," in *Proc. of the 2nd International Conference on Fuzzy Logic and Neural Networks*, K. Jizuka, Ed., 1992, pp. 201–204.

[41] M. L. Wong and K. S. Leung, *Data Mining using Grammar Based Genetic Programming and Applications*, 1st ed. Norwell, MA, USA: Kluwer Academics Publishers, 2000.

[42] A. M. García-Vico, J. Montes, J. Aguilera, C. J. Carmona, and M. J. del Jesus, "Analysing Concentrating Photovoltaics Technology through the use of Emerging Pattern Mining," in *Proc. of the 11th International Conference on Soft Computing Models in Industrial and Environmental Applications*. San Sebastián, Spain: Springer, 2016, pp. 1–8.

[43] C. J. Carmona, C. Chrysostomou, H. Seker, and M. J. del Jesus, "Fuzzy Rules for Describing Subgroups from Influenza A Virus Using a Multi-objective Evolutionary Algorithm," *Applied Soft Computing*, vol. 13, no. 8, pp. 3439–3448, 2013.

[44] C. J. Carmona, M. J. del Jesus, and S. García, *Foundations and Applications of Intelligent Systems*. Springer, 2014, ch. Applying Subgroup Discovery Based on Evolutionary Fuzzy Systems for Web Usage Mining in E-Commerce: A Case Study on OrOliveSur.com, pp. 591–601.

[45] B. L. Miller and D. E. Goldberg, "Genetic Algorithms, Tournament Selection, and the Effects of Noise," *Complex System*, vol. 9, pp. 193–212, 1995.

[46] J. H. Holland, "Adaptation in natural and artificial systems," *University of Michigan Press*, 1975.

[47] K. Deb, A. Pratap, S. Agrawal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[48] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "MOA: massive online analysis," *Journal of Machine Learning Research*, vol. 11, pp. 1601–1604, 2010. [Online]. Available: https://moa.cms.waikato.ac.nz/

[49] M. García-Borroto, O. Loyola-González, J. F. Martínez-Trinidad, and J. A. Carrasco-Ochoa, "Evaluation of quality measures for contrast patterns by using unseen objects," *Expert Systems with Applications*, vol. 83, pp. 104 – 113, 2017.

APÉNDICE B

Tabla de resultados relativa al estudio experimental

En este apéndice se muestra una tabla de resultados para cada uno de los conjuntos de datos analizados en la Sección 3.2.3. Se destaca que el tiempo de ejecución es relativo a cada bloque de datos. El resultado mostrado en cada conjunto de datos es la media de cinco ejecuciones con diferentes semillas aleatorias.

Dataset	n_r	n_v	WRAcc	CONF	GR	TPR	FPR	T. Ejec (ms)
AG-c10	2,000	1,117	0,637	0,638	1,000	0,637	0,362	89
AG-c11	1,999	1,367	0,626	0,628	1,000	0,611	0,359	92
AG-c12	2,011	1,340	0,626	0,634	1,000	0,609	0,356	89
AG-c13	1,998	1,103	0,859	0,859	1,000	0,859	0,141	89
AG-c14	2,001	1,129	0,859	0,859	1,000	0,859	0,141	88
AG-c15	1,999	2,095	0,825	0,548	1,000	0,749	0,099	103
AG-c1	1,999	1,779	0,522	0,516	0,925	0,638	0,595	109
AG-c16	2,001	2,051	0,828	0,923	1,000	0,746	0,091	89
AG-c17	1,999	1,096	0,849	0,849	1,000	0,849	0,151	85
AG-c18	1,996	1,136	0,849	0,850	1,000	0,849	0,151	85
AG-c19	2,000	2,049	0,777	0,508	0,948	0,707	0,150	100
AG-c20	2,001	2,529	0,872	0,888	1,000	0,862	0,118	99
AG-c2	2,016	1,729	0,523	0,519	0,946	0,662	0,617	95
AG-c3	2,000	1,576	0,614	0,609	1,000	0,609	0,381	88
AG-c4	2,001	1,729	0,615	0,617	1,000	0,608	0,379	89

Continua en la siguiente página

Dataset	n_r	n_v	WRAcc	CONF	GR	TPR	FPR	T. Ejec (ms)
AG-c5	1,999	2,009	0,686	0,787	1,000	0,530	0,157	92
AG-c6	2,002	1,921	0,680	0,761	1,000	0,556	0,197	94
AG-c7	2,000	1,272	0,674	0,665	1,000	0,673	0,325	95
AG-c8	2,001	1,137	0,674	0,676	1,000	0,673	0,325	92
AG-c9	1,999	1,246	0,637	0,632	1,000	0,637	0,363	90
ANG-c1	1,999	1,484	0,884	0,945	1,000	0,821	0,053	83
ANG-c2	1,997	1,995	0,961	0,995	1,000	0,927	0,005	93
ANG-c3	2,001	1,510	0,905	0,940	1,000	0,878	0,069	95
ANG-c4	2,000	1,477	0,926	0,962	1,000	0,894	0,041	90
ANG-c5	1,998	1,943	0,958	0,992	1,000	0,924	0,008	93
HPG-c1	2,001	1,495	0,719	0,722	1,000	0,716	0,277	75
HPG-c2	2,000	1,382	0,814	0,815	1,000	0,814	0,185	73
HPG-c3	2,007	3,137	0,607	0,615	1,000	0,587	0,373	110
HPG-c4	2,004	2,615	0,702	0,703	1,000	0,700	0,297	108
HPG-c5	2,008	3,791	0,577	0,586	1,000	0,545	0,392	163
HPG-c6	2,004	3,045	0,663	0,665	1,000	0,660	0,333	154
LEDG-c1	10,000	2,544	0,964	0,722	1,000	1,000	0,072	739
LEDG-c2	10,000	2,118	0,962	0,707	1,000	1,000	0,076	593
LEDGD-c1	10,000	2,418	0,965	0,742	1,000	1,000	0,070	665
LEDGD-c2	10,000	2,216	0,963	0,733	1,000	1,000	0,074	666
MG-c1	1,999	1,574	0,757	0,870	1,000	0,651	0,136	65
MG-c2	2,011	1,551	0,758	0,864	1,000	0,649	0,133	66
MG-c3	1,998	1,521	0,757	0,858	1,000	0,661	0,148	68
MG-c4	2,009	1,574	0,758	0,858	1,000	0,654	0,139	67
RRBFG-c1	2,001	2,144	0,579	0,598	1,000	0,524	0,366	70
RRBFG-c2	1,999	2,391	0,568	0,604	0,999	0,480	0,344	67
RRBFG-c3	1,997	2,190	0,551	0,563	0,999	0,501	0,399	69
RRBFG-c4	2,001	4,413	0,654	0,734	1,000	0,472	0,164	76
RRBFG-c5	2,004	3,305	0,622	0,672	1,000	0,532	0,288	96
RRBFG-c6	2,003	2,991	0,597	0,635	1,000	0,510	0,316	82
RRBFG-c7	2,004	4,482	0,739	0,798	1,000	0,646	0,168	111
RRBFG-c8	2,006	4,238	0,674	0,726	1,000	0,581	0,233	125
RRBFG-c9	2,001	4,553	0,620	0,685	1,000	0,469	0,230	131
RRBFGD-c1	2,002	3,256	0,565	0,609	0,994	0,418	0,287	85
RRBFGD-c2	2,000	3,181	0,569	0,617	1,000	0,439	0,301	87
RRBFGD-c3	2,002	3,613	0,540	0,597	0,990	0,322	0,243	91
RRBFGD-c4	2,009	4,056	0,656	0,685	1,000	0,613	0,301	139

Continua en la siguiente página

Dataset	n_r	n_v	WRAcc	CONF	GR	TPR	FPR	T. Ejec (ms)
RRBFGD-c5	2,011	4,079	0,642	0,662	1,000	0,608	0,324	128
RRBFGD-c6	2,005	3,984	0,621	0,648	1,000	0,567	0,324	126
RRBFGD-c7	2,007	5,981	0,626	0,649	0,990	0,583	0,331	150
RRBFGD-c8	2,013	5,554	0,641	0,659	0,999	0,605	0,324	153
RRBFGD-c9	2,010	5,900	0,596	0,623	0,999	0,529	0,338	151
RTG-c1	2,000	1,490	0,510	0,515	0,806	0,400	0,381	90
RTG-c2	2,002	2,728	0,620	0,602	1,000	0,615	0,374	118
RTG-c3	2,001	2,243	0,562	0,574	0,985	0,571	0,446	249
SEAG-c1	1,998	1,636	0,804	0,912	1,000	0,694	0,087	73
SEAG-c2	2,006	1,630	0,804	0,891	1,000	0,695	0,088	72
SEAG-c3	1,997	1,696	0,776	0,909	1,000	0,655	0,103	74
SEAG-c4	2,004	1,779	0,776	0,884	1,000	0,656	0,103	71
SEAG-c5	1,998	1,614	0,831	0,886	1,000	0,737	0,074	76
SEAG-c6	2,006	1,614	0,832	0,909	1,000	0,736	0,071	76
SEAG-c7	1,998	1,698	0,763	0,893	1,000	0,641	0,116	78
SEAG-c8	2,010	1,813	0,763	0,878	1,000	0,640	0,115	71
SG-c1	2,012	1,498	0,937	0,999	1,000	0,875	0,000	94
SG-c2	2,015	1,498	0,937	1,000	1,000	0,874	0,000	87
SG-c3	2,006	1,500	0,900	1,000	1,000	0,800	0,000	102
SG-c4	2,009	1,500	0,900	1,000	1,000	0,800	0,000	96
SG-c5	2,000	1,000	1,000	1,000	1,000	1,000	0,000	106
SG-c6	2,000	1,000	1,000	1,000	1,000	1,000	0,000	103
SING-c10	2,000	1,176	0,814	0,816	0,999	0,814	0,186	74
SING-c11	1,997	1,001	0,814	0,809	1,000	0,814	0,186	72
SING-c12	1,994	1,007	0,814	0,816	1,000	0,814	0,186	81
SING-c13	1,998	1,152	0,814	0,809	1,000	0,814	0,186	78
SING-c14	1,996	1,163	0,814	0,816	1,000	0,814	0,186	75
SING-c15	1,994	1,049	0,814	0,809	1,000	0,814	0,186	76
SING-c1	1,995	1,632	0,787	0,785	1,000	0,787	0,213	80
SING-c16	1,994	1,002	0,814	0,816	0,999	0,814	0,186	72
SING-c2	2,001	1,401	0,787	0,788	1,000	0,787	0,213	80
SING-c3	1,991	1,212	0,787	0,785	1,000	0,786	0,213	82
SING-c4	1,998	1,365	0,787	0,788	1,000	0,786	0,213	81
SING-c5	1,999	1,572	0,787	0,786	1,000	0,786	0,212	80
SING-c6	2,002	1,459	0,787	0,788	1,000	0,786	0,212	78
SING-c7	1,996	1,341	0,787	0,786	1,000	0,785	0,212	78
SING-c8	1,995	1,403	0,787	0,788	1,000	0,786	0,212	77

Continua en la siguiente página

Dataset	n_r	n_v	WRAcc	CONF	GR	TPR	FPR	T. Ejec (ms)
SING-c9	1,996	1,217	0,814	0,809	1,000	0,814	0,186	77
WG-c1	3,003	3,280	0,806	0,615	1,000	0,900	0,287	160
WG-c2	3,005	4,492	0,806	0,618	1,000	0,896	0,283	217
WGD-c1	3,004	4,444	0,805	0,615	1,000	0,898	0,287	193
WGD-c2	3,006	4,688	0,806	0,617	1,000	0,896	0,283	194
MEDIA	**2,385**	**2,193**	**0,751**	**0,757**	**0,996**	**0,715**	**0,212**	**122**

Bibliografía

AGRAWAL, R.; MANNILA, H.; SRIKANT, R.; TOIVONEN, H. y VERKAMO, A. I.: «Fast Discovery of Association Rules.» *Advances in knowledge discovery and data mining*, 1996, **12(1)**, pp. 307–328.

BABCOCK, B.; BABU, S.; DATAR, M.; MOTWANI, R. y WIDOM, J.: «Models and Issues in Data Stream Systems». En: *Proceedings of the Twenty-first ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 3-5, Madison, Wisconsin, USA*, pp. 1–16, 2002.

BACH, S. H. y MALOOF, M. A.: «Paired learners for concept drift». En: *2008 Eighth IEEE International Conference on Data Mining*, pp. 23–32, 2008.

BAILEY, J.; MANOUKIAN, T. y RAMAMOHANARAO, K.: «A fast algorithm for computing hypergraph transversals and its application in mining emerging patterns». En: *Proc. of the 3th International Conference on Data Mining*, pp. 485–488. IEEE, 2003.

BAILEY, J.; MANOUKIAN, T. y RAMAMOHANARAO, K.: «Fast algorithms for mining emerging patterns». En: *Principles of Data Mining and Knowledge Discovery*, pp. 39–50. Springer, 2002.

BAY, S. D. y PAZZANI, M. J.: «Detecting change in categorical data: Mining contrast sets». En: *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 302–306. ACM, 1999.

BAYARDO, J. y ROBERTO, J.: «Efficiently Mining Long Patterns from Databases». *SIGMOD Rec*, 1998, **27(2)**, pp. 85–93.

BIFET, A.; HOLMES, G.; KIRKBY, R. y PFAHRINGER, B.: «MOA: Massive Online Analysis». *Journal of Machine Learning Research*, 2010, **11**, pp. 1601–1604.
`https://moa.cms.waikato.ac.nz/`

BIFET, A.: *Adaptive learning and mining for data streams and frequent patterns*. Tesis doctoral, Universitat Politècnica de Catalunya, 2009.

BIFET, A. y GAVALDA, R.: «Learning from time-changing data with adaptive windowing». En: *Proceedings of the 2007 SIAM international conference on data mining*, pp. 443–448, 2007.

BRZEZIŃSKI, D.: *Block-based and online ensembles for concept-drifting data streams*. Tesis doctoral, Poznan University of Technology, 2015.

CARMONA, C. J.; DEL JESUS, M. J. y HERRERA, F.: «A Unifying Analysis for the Supervised Descriptive Rule Discovery via the Weighted Relative Accuracy». *Knowledge-Based Systems*, 2018, **139**, pp. 89–100.

CAUWENBERGHS, G. y POGGIO, T.: «Incremental and decremental support vector machine learning». En: *Advances in neural information processing systems*, pp. 409–415, 2001.

DEB, K.; PRATAP, A.; AGRAWAL, S. y MEYARIVAN, T.: «A fast and elitist multiobjective genetic algorithm: NSGA-II». *IEEE Transactions Evolutionary Computation*, 2002, **6(2)**, pp. 182–197.

DHEERU, D. y KARRA TANISKIDOU, E.: «UCI Machine Learning Repository», 2017.
`http://archive.ics.uci.edu/ml`

DITZLER, G. y POLIKAR, R.: «Hellinger distance based drift detection for nonstationary environments». En: *2011 IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments (CIDUE)*, pp. 41–48. IEEE, 2011.

DONG, G.; LI, J. y ZHANG, X.: «Discovering jumping emerging patterns and experiments on real datasets». En: *Proceedings of th 9th International Database Conference (IDC'99), Hong Kong*, pp. 155–168, 1999a.

DONG, G.; ZHANG, X.; WONG, L. y LI, J.: «CAEP: Classification by aggregating emerging patterns». En: *Proceeding of Discovery Science'99*, pp. 30–42. Springer, 1999b.

DONG, G. y LI, J.: «Efficient Mining of Emerging Patterns: Discovering Trends and Differences». En: *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 43–52. ACM, New York, NY, USA. ISBN 1-58113-143-7, 1999.

FAN, H. y RAMAMOHANARAO, K.: «An efficient single-scan algorithm for mining essential jumping emerging patterns for classification». En: *Proc. of the 6th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 456–462, 2002.

FAN, H. y RAMAMOHANARAO, K.: «A bayesian approach to use emerging patterns for classification». En: *Proc. of the 14th Australasian database conference*, volumen 17, pp. 39–48. Australian Computer Society, Inc., Adelaide, Australia, 2003a.

FAN, H. y RAMAMOHANARAO, K.: «Fast discovery and the generalization of strong jumping emerging patterns for building compact and accurate classifiers». *IEEE Transactions on Knowledge and Data Engineering*, 2006, **18(6)**, pp. 721–737.

FAN, H. y RAMAMOHANARAO, K.: «Efficiently mining interesting emerging patterns». En: *Advances in Web-Age Information Management*, pp. 189–201. Springer, 2003b.

FAN, H. y RAMAMOHANARAO, K.: «Noise tolerant classification by chi emerging patterns». En: *Advances in Knowledge Discovery and Data Mining*, pp. 201–206. Springer, 2004.

FAYYAD, U. M.; PIATETSKY-SHAPIRO, G. y SMYTH, P.: «From data mining to knowledge discovery: an overview». En: *Advances in knowledge discovery and data mining*, pp. 1–34. AAAI/MIT Press, Menlo Park, CA, USA, 1996a.

FAYYAD, U.; PIATETSKY-SHAPIRO, G. y SMYTH, P.: «The KDD process for extracting useful knowledge from volumes of data». *Communications of the ACM*, 1996b, **39(11)**, pp. 27–34.

FERNÁNDEZ, A.; ALTALHI, A.; ALSHOMRANI, S. y HERRERA, F.: «Why Linguistic Fuzzy Rule Based Classification Systems perform well in Big Data Applications?» *International Journal of Computational Intelligence Systems*, 2017, **10(1)**, pp. 1211–1225.

GAMA, J. A.; ŽLIOBAITĖ, I.; BIFET, A.; PECHENIZKIY, M. y BOUCHACHIA, A.: «A Survey on Concept Drift Adaptation». *ACM Comput. Surv.*, 2014, **46(4)**, pp. 44:1–44:37.

GAMA, J.: *Knowledge discovery from data streams*. CRC Press, 2010.

GAMA, J. y CASTILLO, G.: «Learning with local drift detection». En: *International Conference on Advanced Data Mining and Applications*, pp. 42–55. Springer, 2006.

GAMBERGER, D. y LAVRAC, N.: «Expert-Guided Subgroup Discovery: Methodology and Application». *Journal Artificial Intelligence Research*, 2002, **17**, pp. 501–527.

GARCÍA-BORROTO, M.; MARTÍNEZ-TRINIDAD, J. y CARRASCO-OCHOA, J.: «Fuzzy emerging patterns for classifying hard domains.» *Knowledge and Information Systems*, 2011, **28(2)**, pp. 473–489.

GARCÍA-BORROTO, M.; LOYOLA-GONZÁLEZ, O.; MARTÍNEZ-TRINIDAD, J. F. y CARRASCO-OCHOA, J. A.: «Evaluation of quality measures for contrast patterns by using unseen objects». *Expert Systems with Applications*, 2017, **83**, pp. 104 – 113.

GARCÍA-BORROTO, M.; MARTÍNEZ-TRINIDAD, J. F. y CARRASCO-OCHOA, J. A.: «Fuzzy emerging patterns for classifying hard domains». *Knowledge and information systems*, 2011, **28(2)**, pp. 473–489.

GARCÍA-BORROTO, M.; MARTÍNEZ-TRINIDAD, J. F.; CARRASCO-OCHOA, J. A.; MEDINA-PÉREZ, M. A. y RUIZ-SHULCLOPER, J.: «LCMine: An efficient algorithm for mining discriminative regularities and its application in supervised classification». *Pattern Recognition*, 2010a, **43(9)**, pp. 3025–3034.

GARCÍA-BORROTO, M.; MARTÍNEZ-TRINIDAD, J. F. y CARRASCO-OCHOA, J. A.: «A New emerging pattern mining algorithm and its application in supervised classification». En: *Advances in Knowledge Discovery and Data Mining*, pp. 150–157. Springer, 2010b.

GARCÍA-VICO, A. M.; CARMONA, C. J.; GONZÁLEZ, P. y DEL JESUS, M. J.: «MOEA-EFEP: Multi-Objective Evolutionary Algorithm for Extracting Fuzzy Emerging Patterns». *IEEE Transactions on Fuzzy Systems*, 2018, **26(5)**, pp. 2861 – 2872.

GARCÍA-VICO, A. M.; CARMONA, C. J.; GONZÁLEZ, P. y DEL JESUS, M. J.: «A Big Data Approach for Extracting Fuzzy Emerging Patterns». *Cognitive Computation*, 2019, **11(3)**, pp. 400 – 417.

GARCÍA-VICO, A. M.; CARMONA, C. J.; MARTÍN, D.; GARCÍA-BORROTO, M. y DEL JESUS, M. J.: «An Overview of Emerging Pattern Mining in Supervised Descriptive Rule Discovery: Taxonomy, Empirical Study, Trends and Prospects». *WIREs: Data Mining and Knowledge Discovery*, 2018, **8(1)**.

GARCÍA-VICO, A. M.; GONZÁLEZ, P.; DEL JESUS, M. J. y CARMONA, C. J.: «A First Approach to Handle Emergining Patterns Mining on Big Data Problems: The EvAEFP-Spark Algorithm». En: *IEEE International Conference on Fuzzy Systems*, pp. 1–6, 2017.

GARCÍA-VICO, A.; CARMONA, C. J.; GONZÁLEZ, P. y DEL JESUS, M. J.: «Una primera aproximación para la extracción de patrones emergentes en flujos continuos de datos». En: *Proc. of the XVIII Conferencia de la Asociación Española para la Inteligencia Artificial (XVIII CAEPIA)*, pp. 1093–1098, 2018.

GARCÍA-VICO, A.; CARMONA, C. J.; GONZÁLEZ, P. y DEL JESUS, M. J.: «FEPDS: A proposal for the Extraction of Fuzzy Emerging Patterns in Data Streams». *IEEE Transactions on Fuzzy Systems*, Submitted.

GARCÍA-VICO, A. M.; MONTES, J.; AGUILERA, J.; CARMONA, C. J. y DEL JESUS, M. J.: «Analysing Concentrating Photovoltaics Technology through the use of Emerging Pattern Mining». En: *11th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO), San Sebastián, España*, , 2016.

GOLDBERG, D. E.: *Genetic Algorithms in search, optimization and machine learning*. Addison-Wesley Longman Publishing Co., Inc., 1989.

GONÇALVES, P. M.; DE CARVALHO SANTOS, S. G.; BARROS, R. S. y VIEIRA, D. C.: «A comparative study on concept drift detectors». *Expert Systems with Applications*, 2014, **41(18)**, pp. 8144–8156.

HERRERA, F.: «Genetic fuzzy systems: taxomony, current research trends and prospects». *Evolutionary Intelligence*, 2008, **1**, pp. 27–46.

HOLLAND, J. H.: «Adaptation in natural and artificial systems». *University of Michigan Press*, 1975.

HÜLLERMEIER, E.: «Fuzzy sets in machine learning and data mining». *Applied Soft Computing*, 2011, **11(2)**, pp. 1493–1505.

HULTEN, G.; SPENCER, L. y DOMINGOS, P.: «Mining time-changing data streams». En: *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 97–106. ACM, 2001.

ISHIBUCHI, H.; TSUKAMOTO, N.; HITOTSUYANAGI, Y. y NOJIMA, Y.: «Effectiveness of Scalability Improvement Attempts on the Performance of NSGA-II for Many-objective Problems». En: *Proc. of the 10th Annual Conference on Genetic and Evolutionary Computation*, pp. 649–656, 2008.

JAPKOWICZ, N. y SHAH, M.: *Evaluating learning algorithms: a classification perspective*. Cambridge University Press, 2011.

KHAMASSI, I. y SAYED-MOUCHAWEH, M.: «Drift detection and monitoring in non-stationary environments». En: *2014 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, pp. 1–6, 2014.

KHAMASSI, I.; SAYED MOUCHAWEH, M.; HAMMAMI, M. y GHÉDIRA, K.: «Self-Adaptive Windowing Approach for Handling Complex Concept Drift». *Cognitive Computation*, 2015, **7(6)**, pp. 772–790.

KHAMASSI, I.; SAYED MOUCHAWEH, M.; HAMMAMI, M. y GHÉDIRA, K.: «Discussion and review on evolving data streams and concept drift adapting». *Evolving Systems*, 2018, **9(1)**, pp. 1–23.

KIFER, D.; BEN-DAVID, S. y GEHRKE, J.: «Detecting change in data streams». En: *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pp. 180–191, 2004.

KLOESGEN, W.: «Explora: A Multipattern and Multistrategy Discovery Assistant». En: *Advances in Knowledge Discovery and Data Mining*, pp. 249–271. American Association for Artificial Intelligence, Menlo Park, CA, USA, 1996.

KRALJ-NOVAK, P.; LAVRAC, N. y WEBB, G. I.: «Supervised Descriptive Rule Discovery: A Unifying Survey of Constrast Set, Emerging Pattern and Subgroup Mining». *Journal of Machine Learning Research*, 2009, **10**, pp. 377–403.

KRAWCZYK, B.; MINKU, L. L.; GAMA, J. A.; STEFANOWSKI, J. y WOŹNIAK, M.: «Ensemble learning for data stream analysis: A survey». *Information Fusion*, 2017, **37**, pp. 132–156.

KRAWCZYK, B. y WOŹNIAK, M.: «One-class classifiers with incremental learning and forgetting for data streams with concept drift». *Soft Computing*, 2015, **19(12)**, pp. 3387–3400.

KUNCHEVA, L. I.: «Classifier ensembles for changing environments». En: *International Workshop on Multiple Classifier Systems*, pp. 1–15. Springer, 2004.

LEUNG, K. S.; LEUNG, Y.; SO, L. y YAM, K. F.: «Rule Learning in Expert Systems Using Genetic Algorithm: 1, Concepts». En: K. Jizuka (Ed.), *Proc. of the 2nd International Conference on Fuzzy Logic and Neural Networks*, pp. 201–204, 1992.

LI, G.; LAW, R.; VU, H. Q.; RONG, J. y ZHAO, X. R.: «Identifying emerging hotel preferences using Emerging Pattern Mining technique». *Tourism management*, 2015, **46**, pp. 311–321.

LI, J.; DONG, G.; RAMAMOHANARAO, K. y WONG, L.: «Deeps: A new instance-based lazy discovery and classification system». *Machine Learning*, 2001, **54(2)**, pp. 99–124.

LIU, Q.; SHI, P.; HU, Z. y ZHANG, Y.: «A novel approach of mining strong jumping emerging patterns based on BSC-tree». *International Journal of Systems Science*, 2014, **45(3)**, pp. 598–615.

MICHALSKI, R. S. y STEPP, R.: «Revealing conceptual structure in data by inductive inference». *Machine Intelligence*, 1982, **10**, pp. 173–196.

MILLER, B. L. y GOLDBERG, D. E.: «Genetic Algorithms, Tournament Selection, and the Effects of Noise». *Complex System*, 1995, **9**, pp. 193–212.

MUTHUKRISHNAN, S.; VAN DEN BERG, E. y WU, Y.: «Sequential change detection on data streams». En: *Seventh IEEE International Conference on Data Mining Workshops (ICDMW 2007)*, pp. 551–550, 2007.

RAMAMOHANARAO, K. y FAN, H.: «Patterns based classifiers». *World Wide Web*, 2007, **10(1)**, pp. 71–83.

SHAKER, A. y LUGHOFER, E.: «Self-adaptive and local strategies for a smooth treatment of drifts in data streams». *Evolving Systems*, 2014, **5(4)**, pp. 239–257.

TERLECKI, P. y WALCZAK, K.: «Efficient discovery of top-k minimal jumping emerging patterns». *Rough Sets and Current Trends in Computing*, 2008, pp. 438–447.

TOUBAKH, H. y SAYED-MOUCHAWEH, M.: «Hybrid dynamic data-driven approach for drift-like fault detection in wind turbines». *Evolving Systems*, 2015, **6(2)**, pp. 115–129.

TRAN, D.-H.: «Automated change detection and reactive clustering in multivariate streaming data». En: *2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF)*, pp. 1–6. IEEE, 2019.

WALD, A.: *Sequential analysis*. Courier Corporation, 1973.

WANG, L.; ZHAO, H.; DONG, G. y LI, J.: «On the complexity of finding emerging patterns». En: *Proc. of the 28th Annual International Computer Software and Applications Conference*, volumen 2, pp. 126–129, 2004a.

WANG, L.; WANG, Y. y ZHAO, D.: «Building emerging pattern (EP) random forest for recognition». En: *17th IEEE International Conference on Image Processing (ICIP)*, pp. 1457–1460, 2010.

WANG, L.; ZHAO, H.; DONG, G. y LI, J.: «On the complexity of finding emerging patterns». En: *Proceedings of the 28th Annual International Computer Software and Applications Conference (COMPSAC)*, volumen 2, pp. 126–129. IEEE, 2004b.

WANG, Z.; FAN, H. y RAMAMOHANARAO, K.: «Exploiting maximal emerging patterns for classification». En: *AI 2004: Advances in Artificial Intelligence*, pp. 1062–1068. Springer, 2004c.

WEBB, G. I.; HYDE, R.; CAO, H.; NGUYEN, H. L. y PETITJEAN, F.: «Characterizing concept drift». *Data Mining and Knowledge Discovery*, 2016, **30(4)**, pp. 964–994.

WIDMER, G. y KUBAT, M.: «Learning in the presence of concept drift and hidden contexts». *Machine learning*, 1996, **23(1)**, pp. 69–101.

WONG, M. L. y LEUNG, K. S.: *Data Mining using Grammar Based Genetic Programming and Applications*. Kluwer Academics Publishers, Norwell, MA, USA, 2000.

YU, Y.; YAN, K.; ZHU, X. y WANG, G.: «Detecting of PIU Behaviors Based on Discovered Generators and Emerging Patterns from Computer-Mediated Interaction Events». En: *Proc. of the 15th International Conference on Web-Age Information Management*, volumen 8485, pp. 277–293, 2014.

ZADEH, L. A.: «The concept of a linguistic variable and its applications to approximate reasoning. Parts I, II, III». *Information Science*, 1975, **8-9**, pp. 199–249,301–357,43–80.

ZHANG, X.; DONG, G. y RAMAMOHANARAO, K.: «Exploring constraints to efficiently mine emerging patterns from large high-dimensional datasets». En: *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 310–314. ACM, 2000.

ŽLIOBAITĖ, I.; PECHENIZKIY, M. y GAMA, J.: «An overview of concept drift applications». En: *Big Data Analysis: New Algorithms for a New Society*, pp. 91–114. Springer, 2016.