

Master en Ciencia de Datos e Ingeniería de Computadores



Trabajo Final de Máster

Análisis de asociaciones en tiempo real en Twitter

Autor:

Pedro Jesús López Abenza
48454588-M

Tutor:

Jorge Casillas

Universidad de Granada
Curso 2018/2019

Resumen

En esta era en la que las cantidades de información generadas a diario son enormes y crecen exponencialmente, la *Minería de flujo* de datos se presenta como un ámbito emergente de la ciencia de datos. Sin embargo, este campo requiere de la creación de técnicas propias a causa de la naturaleza de los datos de estudio, los cuales se presentan como un flujo de datos y destacan por los retos que presentan. Estos retos se centran en lo relativo a la llegada de los datos, la cual se produce de forma continua, a elevadas velocidades y con evoluciones en el tiempo, lo cual se traduce en *cambios de concepto*. Una de las tareas que más interés ha generado en este ámbito es la *extracción de reglas de asociación en tiempo real*, en la cual se estudia el problema de la extracción de reglas de asociación interesantes entre los atributos de los datos de forma dinámica, en un entorno en el que no es posible asumir una estructura de datos *a priori* y en el que existe una evolución temporal de los datos. En este proyecto se pretende realizar una prueba de concepto enfocada en el ámbito de la política. Así, se hace uso de un conjunto de mensajes extraídos de la red social Twitter durante un cierto periodo de tiempo. Así pues, se pretende realizar un cálculo dinámico de las reglas de asociación entre los términos de interés presentes en los mensajes a lo largo del tiempo.

Abstract

Nowadays, the quantity of information daily-generated is enormous and is exponentially increasing day after day. In this situation, the data stream mining shows up as an emergent field of *Data Science*. Nevertheless, this field needs the development of its own techniques, due to the analyzed data's nature. These data are called *streams* and are outlined by the challenges that arise from their features, which need to be overcome: Continue, high-speed arrivals with time evolutions that mean *concept drifts*. One of the most prominent tasks in *data stream mining* is *association stream mining*, which focus on the problem of dynamical extraction of interesting association rules between the data features in a situation where it is not possible to assume an *a priori* data structure and there is a evolution of these data features on time. This project aims to carry a proof of concept focused on a political thematic out, by using *tweets* extracted from the social network, Twitter during a certain period of time. Thereby, a dynamical analysis of association rules between interesting terms with presence in the *tweets* is carried out.

Palabras clave

data stream mining; frequent pattern mining; Reglas de Asociación; Análisis de sentimientos; Machine Learning; Twitter; Tweets políticos, Opiniones políticas; Python; MOA; IncMine; Fuzzy-CSarAFP; SpaCy; NLKT; R

Índice

1. Objetivos	1
2. Introducción	2
3. Marco teórico	4
3.1. <i>Data Stream Mining</i>	4
3.1.1. Definición	4
3.1.2. Modelos de <i>Data Stream Mining</i>	6
3.1.3. MOA	7
3.1.4. Aplicaciones	8
3.2. <i>Association Stream Mining</i>	9
3.2.1. Definición	9
3.2.2. Medidas de calidad	10
3.2.3. Generación de reglas de asociación de interés	14
3.2.4. Reglas de asociación en <i>Data Stream Mining</i>	16
3.2.5. <i>Fuzzy-CSarAFP</i>	17
3.2.6. Aplicaciones	20
3.3. <i>Frequent Pattern Mining</i>	21
3.3.1. Definición	21
3.3.2. <i>Frequent Pattern Mining</i> en <i>data streams</i>	24
3.3.3. MOA-IncMine	26
3.4. <i>Social Media Analysis</i> y Twitter	30
3.4.1. Twitter como red social por excelencia	30
3.4.2. Twitter y la minería de flujo de datos	31
3.4.3. Twitter como fuente de datos políticos	32
3.5. Minería de texto	34
3.5.1. Definición	34
3.5.2. Preprocesamiento	35
3.6. Análisis de sentimientos	40
3.6.1. Definición	40
3.6.2. Enfoques para el análisis de sentimientos	40
3.6.3. Aplicaciones	41
3.6.4. Análisis de Sentimientos en castellano	42
3.6.5. Análisis de Sentimientos en Twitter	44

4. Análisis de resultados	46
4.1. Aplicación en las Elecciones Políticas de 2016 en USA	47
4.1.1. Introducción al conjunto de datos. Análisis exploratorio	47
4.1.2. Procesamiento del Lenguaje Natural	49
4.1.3. <i>Association Stream Mining</i> : Extracción dinámica de reglas de asociación	55
4.1.4. Estudio de las frecuencias de aparición en reglas de interés	63
4.2. Aplicación en el Proceso de Investidura de 2019 en España	64
4.2.1. Introducción al conjunto de datos. Análisis exploratorio	64
4.2.2. Procesamiento del Lenguaje Natural	67
4.2.3. <i>Association Stream Mining</i> : Extracción dinámica de reglas de asociación	72
4.3. Análisis del tiempo promedio	80
5. Conclusiones	81
A. Preparación del entorno de la API de Twitter en <i>Python</i>	82
B. Consultas con la API de Twitter en <i>Python</i>	85
C. Análisis de sentimientos con <i>SentiStrength</i>	90
D. Reglas de asociación obtenidas para el problema de USA	92
E. Reglas de asociación obtenidas para el problema de España	94
Bibliografía	98

1. Objetivos

El objetivo principal de este trabajo es la creación de un método automatizado para la obtención asociaciones entre términos mediante el análisis en tiempo real de flujos de datos consistentes en textos extraídos de la red social de *microblogging* Twitter sobre un determinado tema. Para ello, se seguirá un estudio fundamentado en los pilares teóricos de las ramas de la ciencia de datos aplicadas a lo largo del proyecto, así como en las investigaciones más importantes realizadas durante los últimos años en dichos campos. Por tanto, los objetivos generales del proyecto se pueden desglosar en los siguientes:

- Obtener mensajes propios de la red social Twitter acerca de un cierto tema de interés, mediante un adecuado empleo de las herramientas disponibles para ello.
- Realizar un procesamiento adecuado de los datos textuales previamente obtenidos, con la intención de pasar de datos no-estructurados a un contexto en el que los datos presentan una estructura real, que permite la adecuada extracción de conocimiento mediante técnicas de Minería de datos.
- Obtener la evolución de las asociaciones entre los términos de interés mediante el análisis dinámico de los flujos de datos formados por dichos textos estructurados.
- Realizar un gráfico adecuado, que permita una visualización correcta y eficiente de las asociaciones obtenidas anteriormente y de la evolución de las mismas a lo largo del tiempo.
- Exponer los conceptos más significativos y relevantes acerca del estudio realizado, explicando el estado del arte y la aplicabilidad de cada uno de los ámbitos de la ciencia de datos utilizados, así como los problemas abiertos y limitaciones de los mismos.

Así pues, tras la realización del proyecto se pretende ofrecer una serie de reflexiones con las conclusiones pertinentes acerca del proyecto realizado, así como de los campos tratados a lo largo del proyecto, teniendo en cuenta los conocimientos adquiridos durante el desarrollo del mismo.

2. Introducción

En la actualidad, la sociedad se encuentra en la conocida como era tecnológica. Una época en la que se ha producido la generalización del empleo de aparatos tecnológicos tanto a nivel de negocios y empresas como a nivel personal. Una época en la que todo es automatizado mediante el uso de herramientas tecnológicas y donde todos están conectados con todos, gracias a la generalización del uso de Internet, con un acceso mucho más sencillo en todo todo el mundo. Todo esto se traduce un enorme aumento de la cantidad de datos tecnológicos generados, el cual presenta un carácter exponencial. Aplicaciones móviles, sistemas de telecomunicaciones, sistemas de monitoreo de tráfico de red, flujos de paquetes de Internet, registros de correo electrónico, datos generados en redes sociales... Son algunas de las numerosas fuentes de datos, en las cuales se producen enormes cantidades de datos en muy poco tiempo.

Estos datos producidos se generan de una forma secuencial y ordenada en el tiempo, a velocidades muy elevadas y sin limitaciones en cuanto a las cantidades a producir. Así pues, estas secuencias de datos ordenadas en el tiempo dan lugar a lo que se conoce como flujo de datos. Otra de las implicaciones existentes en la naturaleza de los flujos de datos es que la composición de estos se produce en tiempo real, a medida que se reciben los datos, lo cual se traduce en que en el momento de realizar el análisis no se conocen todos los datos. De este modo, el análisis se debe realizar únicamente en base a los datos conocidos en ese momento. Además, estos flujos de datos pueden representar información perteneciente a situaciones muy separadas en el tiempo, de modo que las realidades expresadas por los datos pueden verse alteradas como resultado de otros factores. Así, una propiedad muy destacada de los flujos de datos es su capacidad a evolucionar en el tiempo, dando lugar al fenómeno conocido como *concept drift* o cambio de concepto.

Estas singularidades hacen que sea necesario recurrir a la aplicación de técnicas propias y características, en las cuales se hace necesario utilizar un enfoque diferente. Surge así un nuevo campo en la ciencia de datos, conocido como minería de flujo de datos o *data stream mining*. Este ámbito se presenta como la solución adecuada para enfrentar problemas en los que se presentan propiedades y situaciones muy características, en las que los datos del problema deban de ser estudiados de una forma distinta a la propuesta por la minería de datos tradicional. Es el caso de las aproximaciones basadas en el uso de ventanas deslizantes.

Una de las ramas de la minería de flujo de datos más destacadas durante los últimos años es la extracción de reglas de asociación en tiempo real o *association stream mining*. Esta rama sigue los conceptos propios del ámbito de reglas de asociación propio de la minería de datos tradicional aunque, como es obvio, requiere de un enfoque capaz de lidiar con la naturaleza de los flujos de datos.

Entre las numerosas fuentes de datos posibles, en este caso se ha decidido analizar un problema de minería de flujo de datos basado en el análisis en tiempo real de los datos generados en la red social Twitter con las opiniones de los usuarios acerca de una determinada temática. En concreto, la problemática elegida para este estudio ha sido la política, presentando dos conjuntos de datos para el análisis. Por un lado, a modo de estudio inicial, se estudiará un conjunto de datos relativo a las elecciones a la presidencia de los Estados Unidos de 2016. Por otro lado, analizará la política española y el proceso de investidura de 2019. De tal modo, se han analizado los mensajes sobre dicho tema, publicados por los usuarios durante Julio y Agosto de 2019.

La gran importancia de Twitter se debe a la amplia difusión de esta red social, la cual presenta la capacidad de generar un debate en el que todas las personas pueden discutir y expresar su opinión en cualquier momento y desde cualquier parte del mundo. Así, aunque existen diversas temáticas con gran tirón dentro de esta plataforma, la política y los temas relacionados con ella se ha convertido en el centro en el eje central sobre el que gira esta plataforma. De este modo, los propios partidos políticos han pasado a utilizar esta plataforma para expresar sus ideas e incluso como parte de sus campañas electorales: Twitter y política son términos totalmente vinculados.

El objetivo principal que se pretende alcanzar en este proyecto es el estudio y monitorización de la evolución de las asociaciones existentes entre términos de interés en política y el sentimiento presente en los mensajes publicados por los usuarios. De este modo, se pretende analizar la presencia de aprobación o repulsa hacia términos o partidos políticos concretos. No obstante, aunque el objetivo principal de este proyecto gira en torno la búsqueda de las asociaciones frecuentes, todo ello requiere de una simplificación y procesamiento del texto presente en los mensajes publicados en Twitter. En este punto las técnicas propias de Minería de Texto y procesamiento del lenguaje natural (NLP) se presentan como herramientas muy útiles, capaces de simplificar y estandarizar el texto presente en los *tweets*, dotándolos de una estructura a partir de la que la extracción de la información sea mucho más fácil y precisa. De tal modo, se recurrirá a la aplicación de algunas de las técnicas existentes en este ámbito.

Así pues, esta combinación de la minería de flujo de datos (y, en concreto, la extracción de reglas de asociación en tiempo real) con estas otras ramas permite alcanzar un análisis completo de la evolución en el tiempo de las relaciones entre términos y sentimientos. De este modo, se presenta como una herramienta capaz de detectar eventos en la línea temporal en los que se produce una llamada de los usuarios o surge una reacción de los mismos ante un cierto suceso. Todo ello podría tener aplicaciones en la preparación y en el enfoque de campañas políticas futuras, así como mejorar la comprensión sobre los votantes utilizando una muestra de la población basada en los usuarios de Twitter.

3. Marco teórico

El proyecto llevado a cabo es totalmente transversal, de modo que en él se han realizado tareas propias de diversos campos de la ciencia de datos. Así, encontramos una combinación de técnicas propias tanto del ámbito de la minería de flujo de datos como del procesamiento del lenguaje natural (NLP). A continuación, se pretende dar una descripción puramente teórica en la que se introduzcan dichos campos aplicados, así como los conceptos más primordiales de los mismos.

3.1. *Data Stream Mining*

3.1.1. Definición

La **minería de flujo de datos** o *data stream mining* constituye un ámbito de la ciencia de datos, separado de la Minería de datos tradicional. La principal y diferencial característica de los problemas propios de dicho ámbito se debe a la forma en la que se dispone de los datos a analizar. Así pues, mientras que en la Minería de datos común se cuenta con los datos antes de iniciar el proceso de aprendizaje, la minería de flujo de datos destaca por el hecho de que **los datos van llegando de forma continua a lo largo del tiempo**, en forma de secuencia con un orden temporal, en grandes cantidades y con una notable velocidad.

Además, en el caso de los flujos de datos, estos se pueden clasificar en dos tipos: *online* y *offline* [1]. Se dice que un flujo de datos es *offline* cuando la llegada de datos se produce de forma regular [2]. Un posible ejemplo real de este tipo de flujo de datos es en el análisis de las secuencias de registro web, pues normalmente se realiza en base a datos de registro en un cierto período de tiempo. Por otro lado, se dice que un flujo de datos es *online* cuando tienen lugar llegadas de datos de uno en uno y en tiempo real. Un ejemplo real de este tipo de flujo de datos sería el análisis de datos recogidos por sensores, pues deben procesarse a medida que llegan y descartarse justo después de ser procesados.

De este modo, la disponibilidad de los datos en la minería de flujo de datos es completamente diferente a la minería de datos tradicional, haciendo que la forma de tratar estos datos sea diferente. Sin embargo, todo ello supone la aparición de diversos retos, debidos a las limitaciones de los recursos computacionales (memoria, procesadores, etc.), los cuales hay que enfrentar. A continuación, se detallan los principales:

- La llegada de los datos se produce a altas velocidades, lo cual dificulta el análisis en tiempo real.
- Los datos, además de ser recibidos de forma continua, suelen obtenerse en cantidades muy elevadas y no limitadas. De tal modo, se adquieren en bloques de gran tamaño,

lo que dificulta su almacenamiento. Por tanto, se requiere un análisis en tiempo real en el que los datos no sean almacenados sino que, tras estudiarse, se descartan.

- En entornos no estacionarios sucede que los datos entrantes pueden tener distintas características. En tal caso, el aprendizaje realizado por el clasificador hasta el momento puede dejar de ser útil y **necesitar adaptarse a la nueva realidad**. En estas situaciones puede que aprender desde cero puede ser ineficiente, haciendo que quizás sea más recomendable adaptar el conocimiento los nuevos datos.

Por tanto, la **minería de flujo de datos** requiere de métodos capaces de dar una rápida respuesta (o al menos mayor que la velocidad con la que se produce la llegada de los datos) y que no requiera del almacenamiento de los mismos. Además, deben de ser capaces de manejar cambios en la distribución de los datos. De tal modo, estos métodos suelen recurrir a soluciones de carácter aproximado, con el objetivo de reducir costes tanto a nivel de memoria como de tiempo.

Por todo ello, los problemas dentro de este ámbito son un caso especial, en el cual se requiere un **aprendizaje en tiempo real** y capaz de clasificar correctamente nuevas instancias con características diferentes a las analizadas anteriormente. Además, se trata de un **aprendizaje dinámico** en el que, a diferencia del aprendizaje automático tradicional, no se cuenta con suficientes datos para el entrenamiento en el momento de comenzar el proceso de aprendizaje. Así, se trata de **algoritmos muy eficientes en espacio y tiempo**, en los que el dato recibido, es procesado y utilizado para actualizar el modelo y, tras ello, olvidado. Este comportamiento queda resumido en la siguiente imagen.

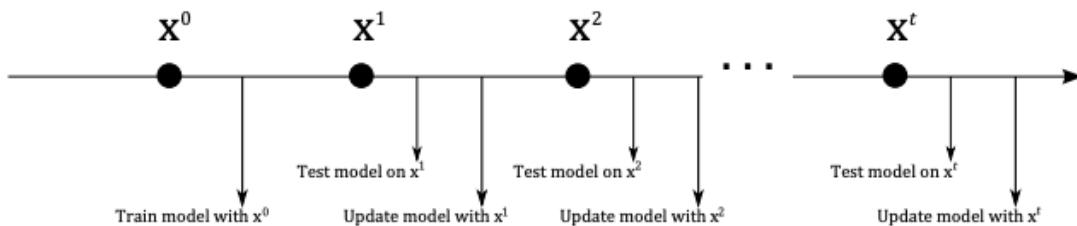


Figura 1. Esquema con el comportamiento característico en los modelos de minería de flujo de datos.

Los principales problemas estudiados en la Minería de flujos de datos se sitúan en los ámbitos de clasificación, *clustering* y *frequent patterns*, *Gam2010*. En el caso del problema de clasificación no sólo se busca mejorar la precisión de la clasificación sino también mejorar la capacidad de reacción (velocidad) y de adaptación a cambios de concepto [3, 4]. Sin embargo, la aplicabilidad de estas ideas en problemas reales es reducida, debido a la dificultad

de conseguir datos etiquetados. Por su parte, los problemas de aprendizaje no-supervisado sí que tienen una mayor aplicación en problemas reales, por lo que se han realizado importantes desarrollos en técnicas de *clustering* incremental y *frequent patterns mining*. Así pues, también se están desarrollando trabajos dentro del ámbito de búsquedas de reglas de asociación o *association rules mining*.

3.1.2. Modelos de *Data Stream Mining*

En general, las posibles variaciones de las características de los datos entrantes en los procesos de minería de flujo de datos hace que normalmente sea más adecuado realizar un estudio en base a los datos propios del pasado más reciente en lugar de utilizar todos los datos a lo largo del tiempo, lo cual además supondría un costo computacional mucho mayor a la hora de realizar cálculos estadísticos. En esta situación se suele recurrir a la definición de una ventana deslizante (*sliding window*) para la selección de los datos a estudiar mediante el modelo de minería de flujo de datos. Así pues, dichas ventanas permiten tomar sólamente aquellos datos que son más *recientes*, reduciéndose el número reducidos de instancias a analizar.

Aunque existen diversos tipos de ventanas deslizantes, podemos considerar principalmente dos tipos de ventanas, en función de la característica utilizada para definir dicha ventana [5]:

- **Sequence-based:** El tamaño de la ventana está definido en base al número de observaciones a considerar. Entre este tipo de ventanas deslizantes aparecen varios modelos:
 - En primer lugar, la **ventana deslizante de tamaño fijo** cuenta con un tamaño de ventana previamente definido, de modo que en ella existe un número fijo de observaciones a considerar. Conforme se van analizando las observaciones se añaden a la ventana deslizante, mientras que las más antiguas se van olvidando.
 - Por otro lado, en el caso de la **ventana del tipo landmark** se considera una ventana *en base a hitos*. En ella se considera toda la historia del flujo de datos desde un cierto punto en el tiempo determinado por un hito o evento, mientras que la longitud de la ventana no es fija, sino que aumenta a medida que llegan nuevos datos. Sin embargo, puede surgir un nuevo hito o evento de interés, de modo que este pasaría a ser el nuevo punto de inicio.
 - Por último, la **ventana amortiguada** (*damped window*) es similar a la anterior, pues considera todo el histórico de datos, aunque en ella se define un factor de amortiguamiento, el cual hace que el peso de los datos sea menor cuanto mayor es su antigüedad. De este modo, los datos más antiguos tienen un menor peso en

el análisis de los datos actuales, siendo los más recientes los que presentan una mayor influencia.

- **Timestamp-based:** La ventana no presenta un tamaño fijo sino que se adecua a un rango de duración o *timestamp*, de manera que sólo se consideran aquellos datos con una antigüedad no mayor a dicho intervalo. Así, por ejemplo, sea el caso de una ventana diaria con avance diario, cuando se avanza a la siguiente hora se utilizan todos los datos comprendidos en dicha hora, mientras que todos los pertenecientes a la hora más antigua son olvidados.

3.1.3. MOA

Massive Online Analysis (MOA) [6] es un *software open-source* desarrollado por la universidad de Waikato y enfocado a la aplicación de experimentos de minería de datos en el contexto de flujos de datos dinámicos. **MOA** está escrito en Java y sigue la idea de la minería de datos en *batch* propia de *Weka*. Además, existen librerías *open-source* que utilizan **MOA** para analizar flujos de datos, como es el caso de *Meka*, *ADAMS* y *OpenML*. Además, existe una plataforma que permite el análisis de flujos de datos en entornos distribuidos conocida como *Apache SAMOA*, la cual combina **MOA** y *Apache Hadoop*.

MOA cuenta con una serie de modelos de aprendizaje, generadores de flujos y evaluadores, que pueden ser usados tanto desde la línea de comandos como con una interfaz gráfica (GUI), que se aplican en un contexto de realización de tareas. Además, cuenta con algoritmos capaces de realizar distintos cálculos en flujos de datos, como pueden ser clasificación, regresión, detección de anomalías, sistemas de recomendación y *frequent pattern mining*. Para realizar algunas de estas tareas se requiere de paquetes adicionales extensiones, como es el caso de las extensiones *MOA-IncMine* y *MOA-Moment*, necesarias para estudiar el problema de *frequent-closed pattern mining*, o *IBLStreams*, que permite analizar problemas de regresión.

La ventaja principal de **MOA** reside en el hecho de que puede aprender y realizar búsquedas en enormes flujos o bases de datos mediante un único análisis sobre los datos de forma notablemente rápida. Ello se debe al hecho de que tras el análisis de cada dato, el algoritmo almacena resúmenes estadísticos en lugar del propio dato, lo que permite que el tiempo de cálculo sea reducido y el consumo de memoria bajo. Además, esto permite una mayor capacidad para enfrentar problemas con cambios de concepto.

3.1.4. Aplicaciones

Existen numerosos escenarios en los cuales está presente la minería de flujo de datos, debido a la necesidad de un análisis o monitorización en tiempo real de la información generada. Encontramos diversas aplicaciones, como es el caso de redes de sensores, control de tráfico, análisis de redes sociales... Algunas de estas aplicaciones se presentan a continuación [7]:

- **IoT y sensores:** Los sensores son utilizados en ámbitos industriales para monitorizar procesos y mejorar la calidad de los mismos pero también en las ciudades con el objetivo de monitorizar la situación de la ciudad y poder hacerla accesible a sus ciudadanos, facilitando sus vidas. Estas enormes redes de sensores requieren de un análisis rápido y eficiente, capaz de informar de la situación en tiempo real.
- **Telecomunicaciones:** Las llamadas telefónicas y los datos generados por el elevado número de teléfonos móviles existentes son enormes, de modo que se trata de una cantidad ingente de datos a procesar y que, normalmente, suele requerirse de respuestas en tiempo real.
- **Medios sociales:** La producción de datos por parte de los usuarios de las principales plataformas sociales en la web (*Facebook*, Twitter, *LinkedIn*...) es constante y a gran velocidad. Estos datos producidos por los usuarios contienen información con un valor total en el momento de su publicación, por lo que para obtener de ellos el mayor rédito posible es necesario realizar un análisis en tiempo real. Ejemplos de estos análisis son la detección de comunidades, de temas de interés o incluso el análisis de sentimientos.
- **Seguridad:** Los sistemas informáticos deben ser protegidos de posibles problemas de funcionamiento e intrusiones maliciosas. Por tanto, se requiere un análisis en tiempo real para la detección de amenazas internas e intrusos.
- **Salud:** Al igual que los signos vitales (presión sanguínea, ritmo cardíaco, temperatura) de los numerosos pacientes ingresados en los hospitales son monitorizados, la telemedicina que se está desarrollando trata de monitorizar estas constantes vitales cuando estén en casa, los cuales quizás necesiten de un análisis en tiempo real para poder advertir a tiempo de problemas de salud del paciente.
- **Marketing y e-commerce:** Las ventas realizadas mediante internet por negocios suponen una enorme cantidad de información sobre transacciones, las cuales pueden ser analizadas en tiempo real.

3.2. Association Stream Mining

3.2.1. Definición

Las **reglas de asociación** constituyen una de las técnicas de minería de datos más utilizada para extraer conocimiento interesante a partir de bases de datos grandes. Así pues, se presentan como una forma para detectar patrones o dependencias entre elementos (*items*) de una base de datos de transacciones. Puesto que generalmente no se conocen la clase de los datos, esta técnica se engloba dentro del ámbito del Aprendizaje no-supervisado.

Inicialmente, este tipo de técnicas fue aplicadas en el análisis del comportamiento del consumidor a la hora de realizar compras en supermercados, de modo que se analiza la presencia de diferentes elementos en las compras realizadas, denominadas transacciones, para buscar regularidades o patrones en los que se encuentren apariciones conjuntas de productos [8]. Así pues, sea un cierto conjunto de atributos que serán denominados *items*, $\mathcal{I} = \{i_1, \dots, i_N\}$, y sea un conjunto de transacciones, $\mathcal{T} = \{t_1, \dots, t_K\}$, que será nuestra base de datos, cada transacción, t_i cuenta con un subconjunto de K elementos del *itemset* \mathcal{I} , denominado *k-itemset*.

Es posible definir una regla de asociación como una implicación del tipo $A \rightarrow C$, donde A y C son conjuntos de items o *itemset* contenidos en el conjunto de atributos ($A, C \subseteq \mathcal{I}$) tales que son disjuntos, de modo que no tienen elementos en común ($A \cap C = \emptyset$). En esta regla definida, el conjunto A es conocido como el *itemset* antecedente de la regla mientras que C es el *itemset* en el consecuente de la regla. Así pues, la interpretación intuitiva de la citada regla de asociación es que cuando A aparece, ello implica que C también tiende a aparecer.

Estas técnicas de búsqueda de reglas de asociación se centran en la búsqueda de relaciones entre los valores presentes en la base de datos y no entre las variables de la misma. Así, se buscan posibles relaciones existentes entre valores de las variables, de manera que se buscan casos en los que la aparición de un valor en una variable suponga el valor de otra variable. Este tipo de asociaciones pueden ser muy poderosas en casos donde las variables pueden tomar más de dos valores. Las técnicas estadísticas pueden tener dificultades para detectar una asociación en este tipo de situaciones, especialmente a la hora de buscar relaciones no-triviales [9].

Además, las técnicas de búsqueda de asociaciones entre términos presenta otras ventajas frente a otro tipo de enfoques de carácter estadístico, los cuales suelen presentar una complejidad del tipo exponencial con respecto al número de variables. Por su parte, las técnicas de búsqueda de asociaciones son escalables a datos de alta-dimensionalidad, siendo capaces de trabajar en bases de datos con miles de variables.

3.2.2. Medidas de calidad

Toda regla de asociación puede evaluarse con una medida de calidad, la cual se encarga de indicar el grado de éxito alcanzado por la regla de asociación en cuestión. Las dos medidas de calidad tradicionalmente utilizadas son el soporte y la confianza, pero presentan ciertos inconvenientes que han supuesto que a lo largo de los años se desarrollen otras medidas nuevas y capaces de representar mejor el éxito de la regla en ciertos contextos. A continuación, se definen las citadas medidas de calidad tradicionales, así como las utilizadas en este proyecto como referencia, detallando sus diferentes ventajas e inconvenientes y justificando así el uso de cada una de ellas:

Soporte

Introducido por [8], esta magnitud es la medida clásica de importancia y mide la proporción de instancias de la base de datos que contienen un *itemset*, X , o que cumplen un cierta regla, $X \rightarrow Y$: Es decir, mide la probabilidad o frecuencia de que un *itemset* o regla aparezcan en el conjunto de transacciones de la base de datos, \mathcal{T} .

$$\text{Support}(X \rightarrow Y) = \frac{\sigma(X \rightarrow Y)}{N_t} = \mathbb{P}(X)$$

donde N_t es el número total de instancias en la base de datos y σ es la función de recuento, la cual recoge el número de ocurrencias de una regla en la base de datos ($\sigma(X \rightarrow Y) = \{t \in N_t; [X \rightarrow Y] \subseteq t\}$). Esta magnitud se utiliza para medir la importancia o validez de un *itemset* o regla de asociación en la base de datos.

El rango de valores posibles del soporte se sitúa entre 0 y 1 ($[0, 1]$) de modo que, de acuerdo a su definición, un soporte nulo indica que la frecuencia de aparición de la regla es nula y un soporte de 1 significa que la regla aparece a lo largo de la totalidad de la base de datos. Decimos que un cierto *itemset* constituye un *itemset* frecuente cuando presenta un soporte superior a un cierto umbral, σ_{Sup} .

Una de las principales características del soporte es que cumple la propiedad anti-monótona, la cual implica que todos los subconjuntos de un *itemset* frecuente son también frecuentes. Esto implica que es imposible que un superconjunto de *itemset* no-frecuente pueda ser frecuente. Esta propiedad es explotada por algoritmos como *Apriori* y su formulación matemática es la siguiente:

$$\forall X, Y \quad \text{t.q.} \quad (X \subseteq Y) \rightarrow \text{Support}(X) \geq \text{Support}(Y)$$

El principal problema que presenta esta medida es el problema de *item-raro* (*rare item problem*), el cual surge por el hecho de que, en general, las reglas más interesantes y que permiten extraer información de calidad no son aquellas que presentan soportes altos, las cuales son poco útiles y ofrecen informaciones obvias. Sin embargo, el uso del soporte como medida de calidad supone dar un mayor peso a esas reglas poco útiles a la vez que eliminar reglas con elementos poco frecuentes, pero que pueden dar lugar a reglas de asociación interesantes y potencialmente valiosas.

Confianza

La confianza, también introducida por [8], es una medida de cumplimiento, la cual mide la probabilidad de que se cumpla una cierta regla definida sobre el total de transacciones de la base de datos que cuenten con el antecedente. Así pues, esta magnitud mide el ratio entre el número de apariciones de la regla frente al número de apariciones del antecedente de la misma. Esto puede ser visto como el cociente entre los soportes de la regla de asociación definida y el antecedente de la misma:

$$\text{Confidence}(X \rightarrow Y) = \frac{\text{Support}(X \rightarrow Y)}{\text{Support}(X)} = \mathbb{P}(Y|X)$$

Como se puede esperar de su definición, la confianza presenta valores diferentes para las reglas $X \rightarrow Y$ y $Y \rightarrow X$. El rango de valores posibles de la confianza para una regla de asociación, $A \rightarrow C$, es entre 0 y 1 ($[0, 1]$) de modo que una confianza de 1 indica que la aparición de A implica la aparición de C. Por su parte, una confianza nula indica que cuando ocurre A, no ocurre C. Decimos que un cierta regla, $A \rightarrow C$, es una regla de asociación cuando presenta una confianza superior a un cierto umbral, σ_{Conf} .

Al contrario que sucedía con el soporte, la confianza no cumple la propiedad anti-monótona anteriormente citada. La principal desventaja de la confianza surge por el hecho de que esta medida es insensible a la frecuencia de aparición del consecuente de la regla definida en la base de datos. De tal modo, la definición de reglas con consecuentes con un alto soporte producirá grandes confianzas independientemente de que no existan asociaciones entre los *itemsets*. Para ilustrar esto, dada una regla $A \rightarrow C$ donde C es muy frecuente (tiene un alto soporte) en la base de datos, la regla citada será definida con una alta confianza, a pesar de que en realidad la existencia de A no implique la aparición de C. Es decir, C es muy frecuente por su propia naturaleza y no por la presencia de A, de modo que la regla de asociación es errónea.

LIFT

Esta magnitud fue introducida por [10] y fue originalmente denominado como **interés**. Se utiliza para medir cuánto mayor es el soporte de una regla respecto al soporte teórico del mismo dado el supuesto de independencia entre los elementos que la componen.

$$\text{LIFT}(X \rightarrow Y) = \frac{\text{Support}(X \rightarrow Y)}{\text{Support}(X)\text{Support}(Y)} = \frac{\text{Confidence}(X \rightarrow Y)}{\text{Support}(Y)}$$

Esta magnitud indica cuando una regla es mejor prediciendo el resultado que asumiéndolo de forma aleatoria. Así pues, el *LIFT* presenta el ratio entre el soporte de la regla $X \rightarrow Y$ y el producto de los soportes de X e Y si fuesen estadísticamente independientes. El rango de valores posibles del *LIFT* se sitúa entre 0 e infinito ($[0, \infty]$) y presenta 3 comportamientos de interés:

- Cuando valor del *LIFT* igual a 1 indica que X e Y son estadísticamente independientes, pues el conjunto reproduce resultados acordes a lo esperado bajo condiciones de independencia entre los elementos de la regla.
- Si el valor del *LIFT* es mayor a 1 implica que la regla aparece con una frecuencia mucho mayor a lo esperado bajo condiciones de independencia entre sus elementos. Por tanto, se intuye que existe una relación de dependencia entre estos elementos que implica su aparición en conjunto con una frecuencia mayor de lo normal. Cuando esto ocurre, implica que la regla definida se ve favorecida respecto a la frecuencia teórica de los elementos bajo el supuesto de independencia.
- Si el valor del *LIFT* es menor a 1 indica que la regla aparece con una frecuencia mucho menor a lo esperado bajo condiciones de independencia entre sus elementos. De tal modo, al igual que antes, se aprecia que existe una relación de dependencia entre estos elementos que implica su aparición en conjunto con una frecuencia menor de lo normal.

La ventaja de usar *LIFT* como medida de calidad reside en que, al estar definido en base a un cociente, éste es capaz de procesar mejor la calidad de reglas con bajo soporte. Esto hace que esta medida sea la ideal para el problema estudiado en el marco práctico, donde es común encontrar soportes bajos.

En la siguiente imagen se puede apreciar el comportamiento del valor del *LIFT* según cuál es la relación entre el soporte del *itemset* y el caso de independencia entre los elementos que componen dicho *itemset*:

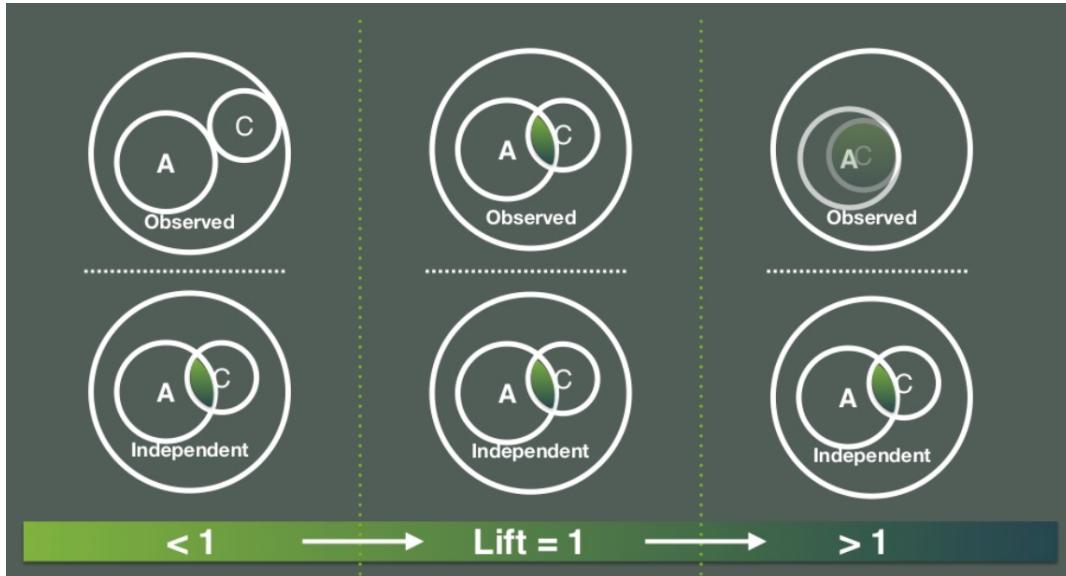


Figura 2. Descripción del valor del *LIFT* según el ratio entre el soporte del *itemset* y el caso de independencia entre los elementos que componen dicho *itemset*. Extraídas de [9].

La ventaja del *LIFT* es que no se ve afectado por la presencia del problema de *itemset* raro (*rare item problem*). Sin embargo, esta medida es susceptible a la presencia de ruido en bases de datos pequeñas. Así pues, la presencia de *itemsets* raros con baja probabilidad pueden producir enormes valores del *LIFT* si, por casualidad, apareciesen alguna vez en las transacciones.

Leverage

Esta magnitud, introducida en [11], mide la diferencia entre el soporte de un *itemset* respecto al soporte teórico del mismo dado el supuesto de independencia entre los elementos que lo componen. Así, mide la diferencia entre la frecuencia de la aparición conjunta de la regla y lo que se esperaría a partir de sus apariciones por separado de los elementos de la misma si fueran independientes.

$$\text{Leverage}(X \rightarrow Y) = \text{Support}(X \rightarrow Y) - (\text{Support}(X)\text{Support}(Y))$$

Así pues, el *leverage* presenta la diferencia entre el soporte de la regla $X \rightarrow Y$ y el producto de los soportes de X e Y si fuesen estadísticamente independientes. El rango de valores posibles se sitúa entre -1 y 1 ($[-1, 1]$), donde el valor de 0 indica un comportamiento independiente, de modo que el conjunto reproduce resultados acordes a lo esperado bajo condiciones de independencia entre los elementos de la regla.

En la siguiente imagen se puede apreciar el comportamiento del valor del *leverage* según cuál es la relación entre el soporte del *itemset* y el caso de independencia entre los elementos que componen dicho *itemset*:

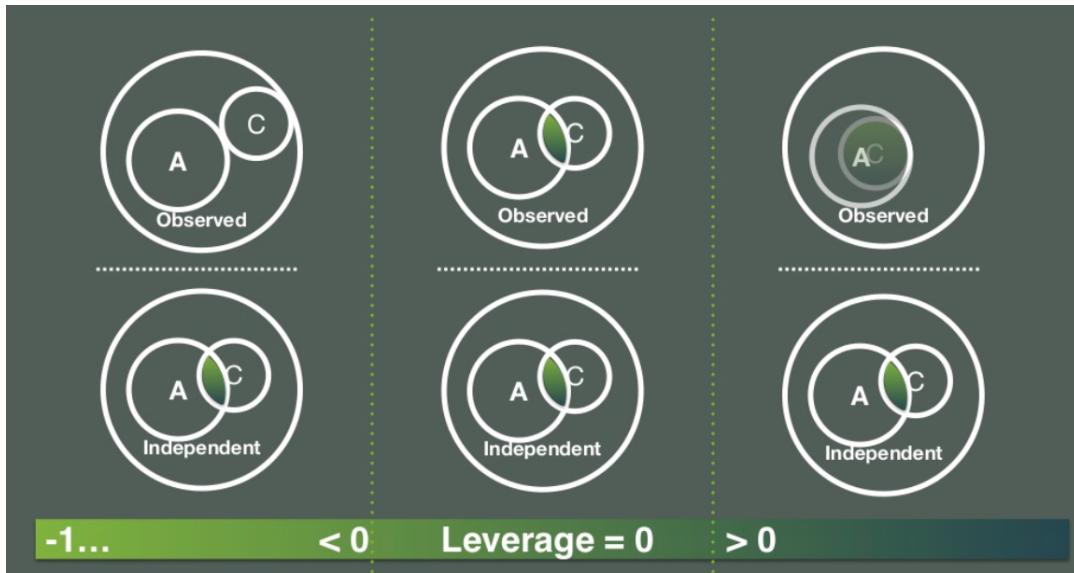


Figura 3. Descripción del valor del *leverage* según la diferencia entre el soporte del *itemset* y el caso de independencia entre los elementos que componen dicho *itemset*. Extraídas de [9].

El establecimiento de un umbral mínimo de *leverage* permite incorporar una restricción de frecuencia implícita. Así, inicialmente es posible aplicar un algoritmo para encontrar todos los conjuntos de elementos que alcanzan un soporte igual al umbral mínimo establecido y, tras ello, realice un filtrado, quedándose con los conjuntos de elementos encontrados utilizando dicho umbral mínimo de *leverage*. No obstante, presenta la desventaja de que el *leverage* también puede sufrir el problema de *item-raro* (*rare item problem*).

3.2.3. Generación de reglas de asociación de interés

La búsqueda de asociaciones entre términos dentro de una base de datos requiere de dos etapas principales, las cuales se describen a continuación:

- **Búsqueda de los conjuntos de *items* frecuentes (*frequent itemsets*)**. En primer lugar, se produce la búsqueda de todos aquellos *itemsets* con un soporte mayor a un ciertos umbral.
- **Generación de reglas de asociación formadas los *frequent itemsets***. Tras ello, se buscan aquellas reglas de asociación que poseen un valor de soporte y confianza mayores a los umbrales definidos.

En general, en la búsqueda de reglas existen dos parámetros: El soporte umbral, el cual controla el número mínimo de casos que la regla debe cubrir, y la confianza umbral, la cual controla la fuerza de predicción de la regla [12]. Históricamente la investigación en descubrimiento de reglas de asociación se ha centrado en el desarrollo de técnicas eficiente para encontrar conjuntos de elementos frecuentes (*frequent itemsets*), prestando poca atención a la extracción de reglas de asociación [9]. La investigación en esta primera etapa de la obtención de reglas de asociación es extensa, siendo el algoritmo de *Apriori* el más conocido, propuesto por [13]. Destaca por su sencillez y su fácil implementación, aunque no es el más adecuado a la hora de analizar bases de datos muy grandes, debido a su lentitud. Otro algoritmo muy popular es el *FP-Growth*, propuesto por [14], destaca por ser muy práctico y recomendable para bases de datos grandes y textuales [15]. En él se crea una representación comprimida de la base de datos utilizando una estructura de árbol *FP-Tree*.

No obstante, recientemente ha habido un crecimiento en el estudio de reglas de asociación raras o de interés [16]. Este tipo de reglas de asociación son aquellas capaces de extraer conocimiento que pueda ser interesante y no fácilmente visible. En general, estas reglas de asociación de interés no son las reglas de asociación más comunes, sino que destacan por presentar un bajo soporte a la vez que tienen una alta confianza. Por tanto, el hecho de definir un bajo soporte umbral en este tipo de análisis supone la generación de un elevado número de reglas. En este tipo de problemas, puede no ser suficiente con algoritmos como *Apriori* y *FP-Growth*, de manera que se han propuesto variaciones de estos algoritmos, adaptándolos a esta tarea. Es el caso de los algoritmos *Apriori-Rare* o *Apriori-Inverse* [17, 18].

Sin embargo, la principal dificultad en este tipo de problemas reside en la definición del soporte umbral, pues un umbral alto generará pocos *itemsets* frecuentes, de modo que puede perderse alguno interesante, mientras que uno bajo puede considerar *itemsets* que no sean interesantes. Para solucionar esto, algunos han considerado que es necesario estudiar las frecuencias de aparición cada elemento de la base de datos. Así, se propone que cada elemento cuenta con un soporte propio, llamado *MIS* (*Minimum Item Support*), dependiente de su frecuencia de aparición en la base de datos [19]. Así, un *itemset* será frecuente si su soporte es mayor que el *MIS* más pequeño de los elementos que lo componen. En esta línea se han presentado variaciones de los algoritmos *Apriori* y *FP-Growth*, utilizando la aproximación *MIS*, como los algoritmos *MSapriori* y *CFP-Growth, Kiran2010*.

Por otro lado, otros como [20] han aplicado este tipo de técnicas al aprendizaje de ontologías, buscando asociaciones entre conceptos lexicalizados dentro del dominio de un cierto *corpus*. Por su parte, en [21] se propone un sistema capaz de extraer reglas de asociación de un documento web basado en características de ciertas palabras clave. Una idea similar es propuesta en [22], donde aplican técnicas de minería de texto, utiliza el algoritmo *GARW* para la generación de reglas de asociación.

3.2.4. Reglas de asociación en *Data Stream Mining*

En el contexto de minería de flujo de datos, donde se producen llegadas de datos dinámicos de forma continua y a altas velocidades, se necesita un modelo capaz de obtener reglas de asociaciones entre los atributos de los datos, sin asumir *a priori* ninguna estructura y con la habilidad de adaptarse de forma *online* ante una dinámica cambiante en los datos que van llegando. De cara a la extracción de reglas de asociación, se presentan tres estrategias diferentes:

Procedimientos de *generar-y-testear*

Se trata de un tipo de algoritmo tremadamente simple, en el que se busca la solución mediante múltiples análisis completos de la base de datos, por lo que no es el más veloz y no es el más adecuado para grandes bases de datos. Este algoritmo persigue la búsqueda de los *itemsets* frecuentes mediante la explotación de la propiedad anti-monótona, ya anteriormente introducida. Esta propiedad afirma que si un *itemset* es frecuente, todos sus subconjuntos también lo serán. Así, una vez se encuentran los *itemsets* frecuentes, el algoritmo realiza combinaciones de estos de cara a generar las reglas de interés.

Técnicas de *divide-y-vencerás*

Esta técnica, comúnmente aplicada en modelos de minería de datos, mejorar la generación de *itemsets* frecuentes y evitar la generación innecesaria de candidatos. Para ello, se realiza la construcción de una estructura auxiliar capaz de recoger las estadísticas de los distintos *itemsets*. De este modo, el algoritmo es capaz de evitar realizar múltiples análisis de la base de datos. Así, tras encontrar los *itemsets* frecuentes, el algoritmo los combina para producir las reglas deseadas. Este tipo de técnicas han sido aplicadas en algoritmos de búsqueda de *itemsets* frecuentes tan conocidos como el *FP-Growth*, en el cual se sigue una estructura de árboles de patrones.

Métodos metaheurísticos

El uso de algoritmos evolutivos (EAs) se presenta como una opción atractiva, pues estos pueden evolucionar hacia reglas de asociación interesantes directamente a partir de los datos. Este es el caso de los algoritmos *Learning Classifier System* (LCSs), basados en sistemas de aprendizaje evolutivo y que se pueden clasificar en dos ramas:

- *Michigan-style LCSs*: Consistente en un sistema *online* y de inspiración cognitiva, en el que se combinan algoritmos evolutivos con un algoritmo de distribución de crédito. Cada uno de ellos produce sus propias reglas, las cuales son evaluadas de forma *online* por el sistema, cuya búsqueda es impulsada por un algoritmo evolutivo.

- *Pittsburgh-style LCSs*: Consistente en un sistema *offline* de algoritmos evolutivos, donde cada uno de ellos produce un conjunto de reglas como solución completa al problema y en el que la búsqueda es impulsada por un algoritmo genético generacional.

Además, recientemente han surgido otras nuevas estrategias, inspiradas en los LCSs. Este es el caso de los *Iterative Rule Learning* (IRL), el cual ha sido ya utilizado para el proceso de *association rule mining*, *Martinez-Ballesteros2011*. Este algoritmo se inspira en los *Michigan-LCSs* en el hecho de que cada algoritmo individual produce sus propias reglas y en el *Pittsburgh-LCSs* en el hecho de que la búsqueda es impulsada por un algoritmo genético generacional. Sin embargo, este método cuenta con dos pasos en lugar de uno: En el primero se realiza el aprendizaje de la regla de asociación mediante el mecanismo de descubrimiento genético, mientras que en el segundo se eliminan los ejemplos cubiertos por el conjunto de entrenamiento.

Otro ejemplo es el *Genetic Cooperative-Competitive Learning* (GCCL), el cual es otra combinación de los métodos LCSs y que ha sido también utilizado para el proceso de *association rule mining* en el campo de comportamiento del consumidor [23]. En este caso, se inspira en los *Pittsburgh-LCSs* en el hecho de que el procesamiento de las reglas se realiza de forma *offline*, mientras que sigue a los *Michigan-LCSs* en el hecho de que las reglas son producidas por cada individuo y posteriormente combinadas.

3.2.5. Fuzzy-CSarAPP

Entre los algoritmos disponibles para este propósito destaca el algoritmo *Fuzzy-CSarAPP*, el cual ha sido propuesto en [24]. Este algoritmo se presenta como un método válido para la extracción de reglas de asociación de interés, mediante el uso de un sistema de carácter genético, difuso y *online*, de manera que es capaz de adaptar el conocimiento adquirido rápidamente ante situaciones de cambios de concepto (*concept drift*).

Fuzzy-CSarAPP se presenta como un algoritmo tremadamente ventajoso y competitivo de cara a la extracción de conocimiento a partir de grandes volúmenes de información. Este algoritmo se presenta como una extensión del método presente en el Sistema de Clasificación Difusa para Reglas de Asociación (*Fuzzy-CSar*), el cual consiste en un clasificador de aprendizaje no-supervisado especializado en la extracción de reglas de asociación difusas y de carácter *online*. Este método se basa en un modelo metaheurístico del tipo *Michigan-LCSs*, el cual utiliza un algoritmo genético para el desarrollo de conjuntos de reglas de asociación. Así pues, la generación del conjunto de reglas de asociación se realiza mediante una evolución directa y muy efectiva, sin recurrir a ninguna clase de análisis o estudio de los conjuntos de *itemsets* frecuentes.

Fuzzy-CSar es capaz de trabajar tanto con atributos categóricos como continuos. El algoritmo realiza una evolución incremental con una población, compuesta por las reglas de asociación difusas. Así, dada una población de individuos, [P], donde cada uno de ellos consiste en una regla de asociación difusa y un conjunto de parámetros con la calidad de la regla, entre los que aparecen magnitudes como soporte, confianza, LIFT, precisión, *fitness*, etc. *Fuzzy-CSar* realiza un aprendizaje incremental del flujo de datos, de manera que en cada iteración se actualizan incrementalmente los parámetros asociados a los individuos de la población de reglas.

Para ello, se crea así un conjunto, [M], donde se cuenta con todas las reglas de la población que cumplen (tanto en antecedente como en consecuente) las relaciones presentes base de datos en un grado mayor a 0. Si el número de individuos considerados es menor a un cierto parámetro de configuración, se lanza un *covering operator*, encargado de aumentar dicha población hasta dicho umbral mediante la definición como antecedente de una variable de la base de datos de forma aleatoria. Tras ello, los individuos son agrupados en conjuntos de candidatos en base a los antecedentes presentes en ellos, estableciéndose una competición entre ellos, en la que los mejores individuos dominan cada grupo. Puesto que cada grupo está definido por los antecedentes de las reglas individuo, hay tantos grupos como reglas en [M] con diferentes atributos en el antecedente. Cada uno de estos grupos puede ser elegido con una probabilidad proporcional a su confianza media. El grupo elegido es sometido a un proceso de generalización, el cual permite que se reduzca el número de reglas que expresan asociaciones similares. Tras ello, se actualizan los parámetros de todos los individuos de la población [M] y se aplica un algoritmo genético (no-generacional) sobre aquellos grupos cuyo tiempo transcurrido desde la última aplicación del algoritmo genético es mayor a un cierto parámetro umbral.

Dicho algoritmo *Fuzzy-CSar* ha sido estudiado bajo distintas situaciones de interés en minería de flujo de datos en [24]. Se destaca la gran capacidad de respuesta en entornos *online* con una rápida adaptación a cambios de concepto, además de ser capaz de extraer información de interés del flujo de datos en forma de reglas de asociación. Asimismo, también se ha estudiado dicho algoritmo en situaciones estáticas, comparándolo con otros como el *FP-Growth* o el *Fuzzy-Apriori*, mostrando una notable superioridad en lo que a tiempos de análisis se refiere, con una gran escalabilidad y siendo capaz de extraer reglas de mejor calidad.

Sin embargo, existe una versión avanzada de dicho algoritmo *Fuzzy-CSar*: *Fuzzy-CSarAFP*, con dos mejoras principales:

- La utilización de diferentes granularidades a través de la definición de nuevos operadores genéticos y representaciones.

- Un mecanismo para actualizar el rango de cada atributo de forma *online*, de manera que no es necesario conocer el dominio de los atributos de la base de datos.

Este nuevo algoritmo, que será el aplicado en la práctica del proyecto, ha sido aplicado en problemas reales en el contexto de búsqueda de relaciones entre las señales obtenidas de la grabación de la actividad cerebral mediante la colocación de electrodos en varios puntos de la cabeza de sujetos con distintas edades y rutinas de actividad física [24].

Como se deduce de la descripción dada sobre el algoritmo, existen numerosos parámetros con diversas tareas. Asimismo, también es necesario determinar qué atributos han sido elegidos para el antecedente de las reglas y cuáles para el consecuente de dichas reglas. A continuación, se citan los distintos grupos de parámetros que se han de definir a la hora de configurar el algoritmo, realizando anotaciones en aquellos que han resultado interesantes:

- **Parámetros del problema:** Se indica el archivo de la base de datos, así como los atributos elegidos como antecedente y consecuente de las reglas, además de definir una semilla y el número de explotaciones a realizar.
- **Parámetros de inicialización**
- **Parámetros del clasificador:** Se indican, entre otros, los parámetros umbral comentados anteriormente, así como el tipo de asociación o si se quiere realizar la generalización de las reglas para evitar redundancias. Cabe destacar el parámetro θ_{mna} , cuyo valor determina el número mínimo de *matchings* entre dato entrante y el conjunto de reglas. De tal modo, si un dato no cumple este mínimo, se añadirán al conjunto de reglas tantas reglas como sea necesarias para que se cumpla. También destaca el parámetro ν , el cual ejerce como exponente del cálculo del *fitness* para medir la validez de una regla. Esta medida se presenta como el producto del soporte de la regla por el *LIFT* de la misma elevado al parámetro ν .
- **Parámetros fuzzy**
- **Parámetros del algoritmo genético:** Se indican las características del algoritmo genético, como el tipo de selección entre individuos y las probabilidades de cada tipo de mutación y *crossover* a realizar. Destacan los parámetros p_C (mutación mediante adicción o eliminación de variables en el antecedente), p_M (mutación por cambio de valores de las variables) y p_S (mutación por cambios de variable en el consecuente).
- **Parámetros de representación**
- **Parámetros estadísticos:** Se indica si realizar cálculos estadísticos, así como el tamaño de la ventana a considerar a la hora de realizarlos.

- **Parámetros de *test*:** Se indica si realizar un *test* con las reglas generadas y si dicho *test* será secuencial, así como el archivo sobre el que aplicarlo y el tamaño de la ventana de datos a considerar para dicho *test*.

3.2.6. Aplicaciones

Las reglas de asociación constituyen un método que permite obtener el conocimiento oculto en los datos, capaz de mejorar la comprensión de los procesos a estudiar y ayudar a la toma de decisiones. Además, también pueden ser aplicadas con el objetivo de realizar tareas de predicción, en el sentido de que es posible determinar un valor de un atributo en base al valor de otro. Para ilustrar esto, por ejemplo, si se sabe que el salario es alto, es posible deducir que los estudios son superiores. Entre los distintos ámbitos donde las reglas de asociación aparecen como una técnica con gran utilidad se encuentran:

- Extracción de información a partir de datos bancarios. Detección de usos fraudulentos de **servicios bancarios**.
- Extracción de información de datos recopilados por **sistemas monitorizados**. Detección de fallos y comportamientos anómalos en redes o sistemas de sensores.
- **Minería de texto y de medios sociales**: Búsqueda de asociaciones de términos en documentos o en informaciones propias de redes sociales y mecanismos de comunicación.
- **Minería web**: Búsqueda de asociaciones entre las características de los internautas y el acceso a páginas, los tiempos de acceso, etc.
- **Minería de patrones secuenciales**: Capaz de proporcionar secuencias de visitas a páginas web que se repiten con frecuencia.

3.3. Frequent Pattern Mining

Aunque el algoritmo presentado anteriormente realiza un análisis de las reglas de asociación en minería de flujo de datos mediante la aplicación de un algoritmo metaheurístico, la forma más común de realizar la extracción de las reglas de asociación ha sido tradicionalmente a la extracción de patrones frecuentes. Esta tarea ha sido tan importante en esta búsqueda que ha dado pie a un ámbito con nombre propio, conocido como *frequent pattern mining*. A continuación, se presenta más detalladamente esta rama, así como sus posibilidades en el campo de la minería de flujo de datos.

3.3.1. Definición

frequent pattern mining se centra en la búsqueda y descubrimiento de patrones frecuentes que aparecen en bases de datos, como por ejemplo registros de transacciones, conjuntos de datos textuales, etc. Esta búsqueda de patrones frecuentes se presenta como la tarea principal dentro de la investigación en reglas de asociación realizada durante las últimas décadas. Se entiende como *patrones* a entidades cuya presencia o ausencia supone un comportamiento concreto y alejado de la aleatoriedad en alguna otra característica.

Estos patrones pueden ser *itemsets*, secuencias, subárboles o subgrafos según la tarea de interés y de la base de datos utilizada. Su objetivo, por tanto, es buscar correlaciones entre atributos, de modo que se trata de una tarea estrechamente ligada a la búsqueda de reglas de asociación. Se trata de una tarea de carácter no supervisado, que puede ser usada en un entorno de análisis exploratorio de datos, con el objetivo de poder discriminar características de cara a aplicar modelos de clasificación o *clustering*.

Sea una base de datos formada por una colección de objetos, $\mathcal{D} = \{o_1, \dots, o_{|D|}\}$, y sea P un conjunto de los posibles patrones que tienen lugar en D , es posible definir una función de conteo tal que $g : P \times O \rightarrow N$, donde O se define como un conjunto de objetos y N se define como un conjunto de enteros no-negativos. Así, dados los parámetros $p \in P$ y $o \in O$, la función $g(p, o)$ devuelve el número de veces que p tiene lugar en o . Por su parte, el soporte del patrón $p \in P$ en la base de datos \mathcal{D} queda definido como:

$$\text{Support}(p) = \sum_{j=0}^{|D|} I(g(p, o_j))$$

donde I es una función, denominada como función indicador, la cual será igual a 1 si $g(p, o_j) > 0$ o será igual a 0 en cualquier otro caso. De tal modo, se considera que un patrón es frecuente si éste presenta un soporte superior a un cierto soporte umbral, definido previamente.

El ejemplo primero y más común es la búsqueda de *itemsets* frecuentes en transacciones o secuencias de elementos, tal y como se propone en [8]. Volviendo a la definición dada dentro del estudio de las reglas de asociación, un *itemset* se define como un conjunto de elementos o atributos que serán denominados *items*, $\mathcal{I} = \{i_1, \dots, i_N\}$. Así, sea un subconjunto de K elementos, se dice que este es un *k-itemset*. Así pues, sea un conjunto de transacciones, $\mathcal{T} = \{t_1, \dots, t_K\}$, que será nuestra base de datos, es posible buscar dentro de ella aquellos *k-itemsets* que aparecen con un soporte superior a un cierto soporte umbral.

Frequent Pattern Mining: Algoritmos de tipo batch

La búsqueda de patrones frecuentes se basa en la idea de analizar todo el conjunto de datos y obtener la calidad de todos los posibles patrones encontrados mediante su soporte, de modo que se determina si un cierto *itemset* es frecuente mediante el recuento del número de transacciones en las que dicho *itemset* aparece. Tras la realización de dicho análisis, se consideran sólo aquellos patrones cuyo soporte es mayor que el soporte umbral definido. Esto es tremadamente costoso cuando se refiere al estudio de *itemsets* pequeños debido al alto número de combinaciones posibles, que crecen exponencialmente con el número de elementos a considerar. De igual modo, otro problema a tener en cuenta es que el número de patrones a considerar crece demasiado rápido con el tamaño del conjunto de datos. Todo ello complica la viabilidad de esta tarea, de modo que se requiere de aproximaciones más eficientes.

Los algoritmos de tipo *batch* se presentan como un tipo de método de *frequent pattern mining* capaz para enfrentar este problema en la búsqueda de patrones frecuentes. Una solución posible radica en la explotación de la propiedad anti-monótona, ya previamente comentada, según la cual se pueden definir enfoques como el del algoritmo *Apriori* [13]. En este enfoque se utiliza el conjunto de patrones de un cierto tamaño $k - 1$ para así generar el conjunto de candidatos a patrones frecuentes de tamaño k . Para ello, utiliza dicha propiedad anti-monótona, de modo que sólamente son considerados como candidatos aquellos patrones de tamaño k que tienen subpatrones de tamaño $k - 1$ frecuentes. Así, se realiza un análisis para calcular el soporte de cada uno de los patrones candidatos, de modo que aquellos que superan un cierto soporte umbral pasan a ser patrones frecuentes. Así, se utiliza la propiedad anti-monótona para considerar todos los posibles candidatos a patrones frecuentes. Además, el número de patrones frecuentes decrece rápidamente con el tamaño de los patrones candidatos considerados, por lo que estos algoritmos se hacen progresivamente más rápidos.

Otro enfoque de tipo *batch* interesante son aquellos que utilizan una estructura de datos adicional para ganar eficiencia y requerir de un menor número de análisis de la base de datos. Es el caso del algoritmo *FP-Growth*, *Han2000*, el cual utiliza una estructura de datos llamada *FP-trees*, la cual puede almacenar el conjunto de datos de forma eficiente y desde la que se pueden recuperar los patrones frecuentes de forma directa y sin generar candidatos. Por

su parte, el algoritmo *Eclat* [25], el cual asocia a cada patrón observado una lista formada por los identificadores de las transacciones en las que se cumple, tras lo cual combina dichos conjuntos para así obtener patrones más grandes.

Sin embargo, existe otro problema más complejo y relativo a la enorme cantidad de patrones generados por los algoritmos, a niveles que se hacen imposibles de analizar o utilizar, de modo que se debe reducir el gasto de memoria. Además, el proceso de extracción de conocimiento se ve a menudo complicado por el hecho de que estos patrones suelen ser redundantes. Un ejemplo de redundancia se presenta cuando hay dos patrones frecuentes tales que uno está contenido en otro y ambos presentan soportes casi iguales. Ante esto, surgen las nociones de patrones cerrados y maximales, los cuales se presentan como representaciones capaces de terminar con estas redundancias y reducir el gasto de memoria y tiempo.

Se define como *patrón cerrado* de una cierta base de datos a aquel patrón frecuente que no tiene superpatrones que sean frecuentes y con un soporte igual al suyo (es obvio que mayor es imposible), mientras que se entiende como *patrón maximal* a aquel patrón que no tiene superpatrones inmediatos que sean frecuentes y presenten un soporte igual al suyo. Todo patrón maximal es cerrado, pero no a la inversa, de modo que los patrones cerrados son más numerosos que los maximales. Sin embargo, aunque ambas representaciones permiten obtener fácilmente todos los patrones frecuentes de la base de datos, estos últimos no aportan información acerca de la frecuencia de los patrones frecuentes, de modo que sería necesario volver a calcular el soporte de los mismo.

Por tanto, los patrones cerrados parecen una buena representación para ahorrar en costes de cálculo. Así, los algoritmos utilizarán los patrones cerrados para realizar los cálculos, aumentando su eficiencia a la vez que cuenta con toda información original. Son varios los algoritmos de tipo-*batch* que pretenden explotar esta representación en el caso de la búsqueda de *itemsets* frecuentes en bases de datos de transacciones. Entre ellos, destacan los métodos *CLOSET* [26], *CLOSET+* [27] y *CHARM* [28].

CHARM se presenta como un algoritmo para la extracción de los patrones frecuentes cerrados de forma eficiente, para lo cual explora el espacio de *itemsets* y el espacio de identificación de transacciones o *tidset*. Así pues, aplica una unión sobre estos dos conjuntos de elementos unidos y, tras ello, una intersección con el conjunto de datos, lo cual permite realizar una búsqueda novedosa en la que se omiten niveles y se identifican rápidamente los *itemsets* frecuentes cerrados. Para ello, utiliza una estrategia de *pruning* de *dos puntas*, de manera que la poda de los candidatos no se realiza únicamente en base a un bajo soporte, sino que también se aplica una poda en función de la propiedad de no-cierre. Así, cualquier *itemset* no-cerrado es podado, evitando así analizar las extensiones de *itemsets* no-frecuentes. Además, hace uso de estructuras de datos internas como *hash-trees* o *tries*.

3.3.2. *Frequent Pattern Mining en data streams*

Aunque la búsqueda de patrones frecuentes es una tarea ampliamente investigada en las últimas décadas, su extensión a la minería de flujo de datos es algo tremadamente interesante. Ello se debe a que la búsqueda de patrones frecuentes choca totalmente con la naturaleza de la minería de flujo de datos, pues está fundamentada en cálculos que no pueden ser completados hasta que se tengan en cuenta todas las transacciones de la base de datos en todo el historial del tiempo: No solamente vale con los datos más recientes, sino que se requieren todos los datos propios del pasado, así como los futuros. Algoritmos como *Apriori* no pueden ser utilizados en este entornos de flujo de datos, pues en ellos se realizan operaciones de unión, las cuales no pueden llevarse a cabo sobre flujos de datos dado que no se dispone de todos los datos, sino que se cuenta con una minoría actualizada de los mismos a partir de ventanas de tamaño limitado [5].

En cuanto al resto de algoritmos propuestos, la mayoría de ellos no diferencian entre la información reciente y la más antigua: Son métodos estáticos, incapaces de diferenciar datos según su antigüedad. No obstante, en un entorno de minería de flujo de datos esto no es válido, pues se requiere de una búsqueda actualizada de los patrones frecuentes, teniendo en cuenta el carácter dinámico de los datos [29]. Ante esto, se ha recurrido al uso de algoritmos *online*, basados en la generación de los *itemsets* frecuentes dentro de una ventana deslizante, la cual es utilizada para definir qué datos se han de considerar en cada momento. Se trata, por tanto, de un problema donde la base de datos no es fija, sino que va cambiando en el tiempo, en base a una limitación temporal (*time-sensitive*) o transaccional (*transaction-sensitive*).

En esta situación, surge el concepto de *itemset* frecuente de flujo, el cual consiste en un *itemset* que presenta un soporte momentáneo (en una cierta ventana temporal de cierto tamaño) superior al soporte umbral establecido. Además, un factor que aumenta el costo computacional de cara al cálculo de los *itemsets* frecuentes en este tipo de problemas es que se debe almacenar la información tanto de los conjuntos frecuentes como de los no-frecuentes, pues estos podrían pasar a ser frecuentes en un instante futuro. Durante los últimos años se han realizado distintas propuestas, de modo que la mayoría de ellas se pueden clasificar principalmente en dos ramas, las cuales se introducen a continuación:

- **Métodos exactos:** En ellos se extraen los conjuntos frecuentes de una base de datos. Para realizar dicho cálculo se requiere del seguimiento de todos en la ventana deslizante y sus soportes, puesto que un *itemset* no-frecuente podría pasar a ser fecuente en cualquier momento, debido al carácter evolutivo de los datos. Esto se vuelve complejo para los casos donde la llegada de los datos se produce a altas velocidades o con ventanas deslizantes de tamaño elevado, de forma que en estos casos el coste computacional de esa exactitud se hace elevado.

- **Métodos aproximados:** En ellos se extraen solamente los *itemsets* cerrados o máximales de una base de datos. A pesar de ser aproximados, estos métodos se presentan como la mejor solución en los escenarios más complejos, pues normalmente son capaces de ofrecer una solución suficientemente buena. No obstante, en ocasiones se presentan resultados erróneos, de manera que se producen falsos positivos (se incluye algún *itemset* pero en realidad no lo es) o falsos negativos (se considera que un *itemset* no es frecuente cuando en realidad sí lo es).

Entre las propuestas para la extracción de patrones frecuentes en minería de flujo de datos destacan las de [2], donde se presentan dos algoritmos, en los cuales se almacenan las frecuencias aproximadas y el margen de error máximo de las mismas: Por un lado, el algoritmo *Sticky sampling*, el cual busca los patrones frecuentes usando un método de sampleado, y, por otro lado, el algoritmo *Losing counting*, el cual se extendió para extraer dichos patrones frecuentes utilizando un modelo de ventana deslizante tipo *landmark*. Por su parte, en [30] se utiliza una combinación de la ventana deslizante común y la ventana amortiguada para generar un modelo de *ensemble* de ventanas deslizantes ponderadas, en el cual el usuario puede definir el número y tamaño de las ventanas, así como sus pesos. En [31] se propone un método aproximado para la extracción de patrones frecuentes, llamado *Can(T)*, capaz de extraer de un flujo de datos los K patrones más frecuentes. Por otro lado, en [32] se introduce el concepto de ventana deslizante de tamaño variable en este tipo de problemas, con el objetivo de manejar flujos de datos con diferencias de tráfico en el tiempo.

En cuanto a los métodos exactos, destaca el propuesto en [33], donde se presenta un algoritmo incremental para la extracción de *itemsets* frecuentes, conocido como *MOMENT* y que utiliza una ventana deslizante y una estructura de datos en forma de árbol denominada como *CET* o *Closed Enumeration Tree*, la cual permite almacenar los *itemsets* cerrados de cada ventana deslizante, así como la información de aquellos nodos de *itemsets* no-frecuentes, pues podrían pasar a ser frecuentes en el futuro. En este algoritmo se realiza una búsqueda inicial en dicha estructura y tras ello va realizando actualizaciones sobre el tipo de cada nodo. Su principal desventaja es que almacena internamente todas las transacciones en una estructura *FP-tree* modificada. Una variación de este algoritmo es *NEWMOMENT*, el cual fue propuesto en [34]. En él se mantiene la ventana deslizante sensible a transacciones y el carácter exacto, pero se utiliza una representación basada en secuencia de *bits* para mejorar la eficiencia. Además, utiliza una nueva estructura de datos, *NewCET*, en la que solamente se almacenan los *itemsets* cerrados en la ventana deslizante.

Otra variación es la propuesta en [35], donde se propone el algoritmo *Subset-lattice*. En este se incorporan las ideas del *NEWMOMENT* pero se emplea una estructura reticular para extraer todos los *itemsets* frecuentes en una ventana deslizante. Además, estos datos

se actualizan gradualmente con el desplazamiento de la ventana sin necesidad de reconstruir la estructura de la red. Por último, otro de los algoritmos exactos más destacados es el *CLOSTREAM*, propuesto en [36], en el que se busca el conjunto de *itemsets* cerrados con una ventana deslizante sensible a transacciones. Se utilizan dos estructuras de datos para mantener información acerca de los *itemsets* cerrados en la ventana deslizante, conocidas como *Closed Table* y *Cid List*. La actualización de la información se realiza por transacción, mediante dos procedimientos: *CloStream+* y *CloStream-*. Así, mientras que el primero se encarga de lidiar con la llegada de transacciones a la ventana deslizante, el segundo se encarga de las salidas de la misma.

Sin embargo, otra rama que está ganando ventaja son los métodos aproximados. Entre ellos, destaca el método presentado en [37], denominado *CLAIM*. En él se utiliza una ventana deslizante sensible a transacciones y se presentan los conceptos de intervalo de relajación y *itemset* cerrado *relajado*. Así pues, se utilizan grafos bipartitos para generar una representación de doble enlace para administrar los *itemsets* en cada intervalo relajado, con el objetivo de reducir el coste en las actualizaciones. Para ello, estos grafos utilizan un *Hash-based Relaxed closed itemset Tree (HR-Tree)*, el cual combina las características de un *prefix-tree* y una matriz asociativa. Sin embargo, uno de los que mejores resultados ofrece es el método *IncMine*, propuesto en [38], el cual utiliza un soporte umbral de relajación para mantener los *itemsets* que podrían ser frecuentes en un futuro y una indexación invertida para facilitar la actualización. Este método será el utilizado en el análisis realizado, de modo que será descrito de una forma más minuciosa a continuación.

3.3.3. MOA-IncMine

IncMine es una extensión de **MOA**, basada en el algoritmo presentado en [38], el cual es utilizado para la extracción y actualización incremental de *itemsets* cerrados frecuentes (FCIs) sobre flujos de datos de alta velocidad. Las características que hacen de este algoritmo tan útil se citan a continuación:

- Ofrece soluciones aproximadas, para lo cual hace uso de un soporte umbral de relajación, mediante el cual se mantiene un conjunto adicional con *itemsets* no-frecuentes pero que en un futuro podrían pasar a convertirse en frecuentes.
- Define la magnitud de soporte aproximado de un *itemset* en un cierto intervalo de tiempo, la cual es descrita como el soporte del *itemset* en dicho intervalo de tiempo, a excepción de cuando este valor es inferior a un cierto límite, ϵ :

$$\hat{sup}(X, t) = \begin{cases} 0 & \text{if } sup(X, t) < \epsilon \\ sup(X, t) & \text{otherwise} \end{cases}$$

- Introduce el concepto de *semi-FCIs*, de modo que los *itemsets* son mantenidos en el análisis por más tiempo, con un soporte umbral que aumenta progresivamente para ellos. Así, aquellos *itemsets* cerrados que en algún momento fueron definidos como frecuentes deben demostrar que mantienen su alta frecuencia para seguir siendo considerados como tal. Para realizar las actualizaciones eficientemente, se crea y mantiene una estructura de índices invertidos de los *semi-FCIs*.
- Utiliza una aproximación sensible al tiempo a través de una ventana deslizante en la que se toman aquellos elementos que han llegado en los últimos N pasos, pudiendo ser estos elementos muchos o ninguno. Esta ventana, junto con la presencia de los FCIs, permite manejar los cambios de concepto.
- Realiza una actualización del tipo-*batch*, lo cual se vuelve crucial de cara a mejorar la eficiencia de cálculo. Además, permite asumir la estacionariedad en segmentos moderadamente largos del flujo, así como evitar problemas ante pequeñas inexactitudes durante periodos transitorios relativamente cortos. Así pues, los conjuntos de elementos de los segmentos de datos son extraídos y analizados, de modo que el resultado obtenido se utiliza para actualizar una estructura de datos global.

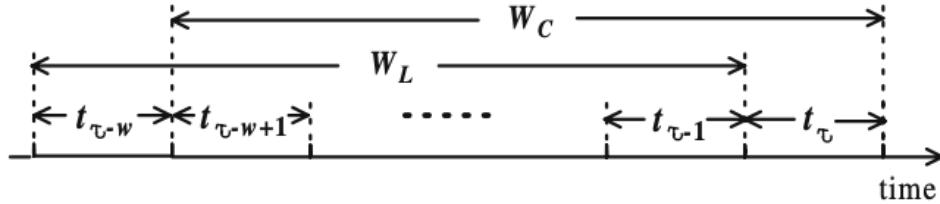


Figura 4. Ventana deslizante propia del algoritmo *IncMine*. extraída de [38].

IncMine hace uso de una ventana deslizante sensible al tiempo, de manera que busca obtener los *itemsets* cerrados frecuentes en dicha ventana. En ellas, se realiza la búsqueda de aquellos *itemsets* frecuentes o candidatos a frecuentes en el futuro. Sin embargo, en el caso de que un *itemset* no-frecuente en el pasado se convierta en el futuro en frecuente, no se conoce el soporte de dicho *itemset* antes de ser frecuente. Para ello, se utiliza la idea propuesta en [2], en la que se propone un parámetro de error, $\epsilon \in [0, \sigma]$ (donde σ es el soporte conocido). Así, en la siguiente ventana se considerarán aquellos *itemsets* que presenten un soporte mayor a ϵN en la ventana actual, donde N es el número de transacciones. Dado que un valor pequeño de ϵ supone muchos *itemsets* candidatos, se define un ϵ basado en una función MST(*Minimum-Threshold-Support*) relajada, del tipo $\epsilon = r\sigma$, donde $r \in [0, 1]$ es el ratio de relajación. Esta función es utilizada para extraer a lo largo del tiempo las FCIs, los cuales serán aquellos *itemsets* que presentan un soporte aproximado no-nulo.

Así pues, sea un conjunto de FCIs, L, extraídos en una cierta ventana temporal, dicho conjunto es actualizado en cada paso temporal en base a las nuevas transacciones recibidas, de modo que debe reflejar estas nuevas transacciones y olvidar los efectos de las más antiguas. Para ello, *IncMine* extrae en primer lugar el conjunto de FCIs en las nuevas transacciones, C, y, tras ello, actualiza el conjunto L con el contenido de C. Sin embargo, la implementación directa de esta idea es costosa, pues el número de *itemsets* a almacenar puede ser muy grande, ya que todos los conjuntos (hasta los menos frecuentes en dicha ventana) deben mantenerse en L por si vuelven a ser frecuentes en próximas ventanas temporales [39].

La implementación disponible en la extensión de **MOA** cuenta con ciertas diferencias con respecto al algoritmo descrito en [38], de modo que su funcionamiento es ligeramente diferente. En primer lugar, la ventana deslizante implementada es sensible a transacciones en lugar de ser sensible al tiempo. Asimismo, al ser un método aproximado puede presentar falsos negativos (en este algoritmo en concreto no hay falsos positivos), de modo que se eliminan *itemsets* que pueden ser frecuentes en realidad. Cuenta con un método *batch* para la extracción de *itemsets* frecuentes, basado en una versión mejorada del método *CHARM*, el cual utiliza datos de *bits* para representar las transacciones. Por último, como se comentó anteriormente, *IncMine* utiliza una estructura de índices invertidos para manejar los *semi-FCIs* almacenados en una ventana deslizante. La implementación de **MOA** realiza una indexación invertida muy sofisticada, basada en un *array* (*FCI-array*) y un índice invertido (*Inverted FCI Index* o *IFI*). En el primero se almacenan todos los *semi-FCIs*, junto con una etiqueta asociada al mismo (y que se corresponde con la posición en el *array*) y su soporte aproximado. Cada *array* tiene asociado una *garbage-queue*, de modo que cuando un *semi-FCI* es eliminado del *FCI-array* su etiqueta se incluye en dicha cola. Si vuelve a ser introducido en el *FCI-array*, se elimina la etiqueta de dicha cola.

ID	Size-1	Size-2	Size-3	Size-4
0	x	xy	xyz	bxyz
1	y	xz	bxy	abcd
2	b	bx	bxz	
3	g	by	abd	
4		bc		
5		bd		

Figura 5. Ejemplo de la estructura de *FCI-arrays*, propia de la indexación invertida implementada en *IncMine*. Extraídas de [40].

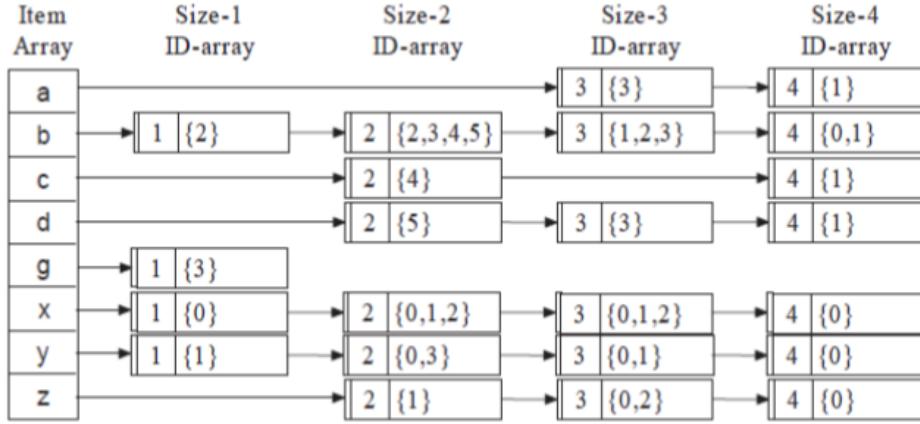


Figura 6. Ejemplo de estructura de *IFI*, propia de la indexación invertida implementada en *IncMine*. Extraídas de [40].

Dada esta nueva estructura, dado un cierto *itemset* es posible obtener eficientemente su posición en el *FCI-array*, seleccionar el menor superconjunto de *semi-FCIs* (lo cual se denomina como SFS) y considerarlo o borrarlo. La eficiencia de la indexación invertida se debe a que simplemente hay que unir dos *arrays* ya ordenados, lo cual es simple y rápido. Sin embargo, el orden de unión de los *arrays* puede variar la eficiencia del cálculo [38]. Este problema ha sido estudiado en [41], mostrando diversas soluciones. De ellas, la implementación en **MOA** sigue una aproximación *small vs. small*, el cual realiza la intersección procediendo desde la lista más pequeña a la más grande.

Cabe destacar que, aunque el algoritmo *MOMENT* [33] también cuenta con su propia extensión para el *software* de **MOA**, los experimentos realizados muestran la conveniencia de hacer uso de *IncMine* pues, a pesar de ser un método aproximado, es capaz de realizar cálculos con muy buenos resultados y, sobretodo, con una eficiencia mucho mayor [39].

3.4. *Social Media Analysis* y Twitter

3.4.1. Twitter como red social por excelencia

Uno de los ámbitos en el cual es cada vez más utilizada la minería de flujo de datos es el **análisis de medios sociales**. Durante los últimos años, estos medios sociales se han erigido como uno de las grandes fuentes de información y de opinión. La aplicación de técnicas de minería de datos sobre los datos provenientes de redes sociales puede revelar patrones sobre los individuos inmersos en el ambiente compartido y producir conocimiento que antes no era factible encontrar debido a la variedad y complejidad de la información. Así, estos medios sociales se han convertido en un elemento fundamental a tener en cuenta por parte de las grandes empresas a la hora de analizar la calidad de sus productos, definir sus estrategias de *marketing* e incluso a la hora de la toma de decisiones [42].

En concreto, dentro de los distintos medios sociales aparecidos en los últimos años, destaca el caso de Twitter, una red social de *micro-blogging* que permite conectarse con amigos y obtener en tiempo real información de eventos y temas de interés. Desde su aparición en 2006, se ha erigido como una red social masiva. Contando en la actualidad con más de 140 millones de usuarios activos y 400 millones de *tweets* diarios, la facilidad y la rapidez que ofrece al usuario a la hora de compartir sus opiniones y sentimientos hace de Twitter un medio de comunicación con un rol cada vez mayor en ámbitos de carácter socio-político.

Los usuarios de esta red social pueden interactuar directamente entre ellos, para lo cual existen una serie de convenios que permiten a los usuarios dar una estructura a sus mensajes [42]. Así, para mencionar o contestar a otros usuarios se debe utilizar el símbolo @ antes del *username* del usuario en cuestión, estableciendo un vínculo entre usuarios que permite las conversaciones entre ellos. Además, entre estas posibles interacciones surge la acción del *retweet*, la cual permite volver a publicar el mensaje de otro usuario. Por otro lado, destaca que los *tweets* suelen ir acompañados de etiquetas, denominadas como *hashtag* y que utiliza el símbolo #, que permiten categorizar una publicación o crear una relación con un tema de conversación que es tendencia. Además, también es posible añadir *links* a páginas webs externas utilizando un servicio de acortamiento de URL propio.

En resumen, Twitter se erige como la red social más destacada de cara a obtener información concisa y precisa a lo largo del tiempo. Además, la estructura de *micro-blogging* aporta un fácil mecanismo para la comunicación y las discusiones entre usuarios, así como herramienta dentro de las estrategias de *Marketing* por parte de las empresas. El análisis de su uso deja al intercambio de información, las conversaciones y los informes de noticias como principales tareas, mientras que los principales tipos de usuarios son fuente de información, amigos y buscadores de información [43].

3.4.2. Twitter y la minería de flujo de datos

Twitter sigue un modelo de flujo de datos en el que nuevos datos surgen a una gran velocidad y en cantidades ilimitadas. Además, estos datos suelen contener información en la que los usuarios expresan opiniones o información acerca de sus opiniones, gustos e intereses. Dicha información puede cambiar o evolucionar en el tiempo en base a diversas circunstancias personales o globales, de modo que estos datos tienden a perder valor a medida que pasa el tiempo. En este escenario, la minería de flujo de datos se presenta como una herramienta necesaria de cara a disponer de una capacidad adecuada para trabajar con la información generada en Twitter de la forma más rápida posible y adecuada al contexto temporal.

Twitter presenta su propia herramienta API (*Application Programming Interface*) para permitir a empresas e investigadores obtener, analizar y trabajar con los datos generados por sus usuarios. No obstante, cabe destacar que, con el objetivo de proteger la información de sus usuarios, esta API presenta en su versión *Standard* una fuerte restricción en cuanto a número de *tweets* a obtener y antigüedad máxima de los mismos. A pesar de ello, existen versiones *Premium* (gratuitas aunque con ciertas limitaciones) para aumentar las capacidades de búsqueda (Aunque lamentablemente seguían sin ser suficientes para el propósito del proyecto, que requería del mayor número de datos posible). Así pues, en el **Apéndice A** queda detallado todo lo relacionado con la definición de dicha API y sus credenciales, así como los distintos para realizar un uso adecuado para nuestro propósito. Por su parte, el uso de la API de Twitter requiere de la realización de una búsqueda en base a la solicitud de una cierta consulta, en la cual se puede hacer uso de diversas opciones o parámetros. Estos parámetros, así como los operadores lógicos aceptados por el sistema para la consulta quedan resumidos dentro del **Apéndice B**.

La información generada en los medios sociales y, en especial, en Twitter es tremenda mente abundante, de modo que estos medios se imponen como la mejor forma para recolectar datos en tiempo real. Sin embargo, lamentablemente esta información no es información de calidad. Ello se debe a que los datos generados en estos medios se presentan en forma de texto libre, de modo que no cuenta con una estructura adecuada para hacer de ellos información de calidad. Las distintas organizaciones y empresas, interesadas en extraer información de estas redes sociales para entender cómo piensan los consumidores y cómo aumentar sus beneficios, requieren de un análisis capaz de dotar a dichos datos de un carácter adecuado y que permita extraer de ellos la máxima información útil posible.

En este punto, la Minería de Texto surge como una herramienta necesaria para realizar un procesamiento del texto libre, con el objetivo de hacer de éste información de calidad y estructurada. A continuación, se describirá este ámbito de la ciencia de datos, de manera que se explorarán y se detallarán las principales técnicas empleadas en él.

3.4.3. Twitter como fuente de datos políticos

Los medios sociales y, en especial, Twitter tienen cada día un papel más importante en las sociedades modernas. Uno de los temas a los que Twitter está cada vez más ligado es la política, lo cual quedó demostrado tras la exitosa campaña de Barack Obama en Twitter de cara a las elecciones presidenciales a la Casa Blanca de 2008 [44]. Por ello, durante las últimas décadas se han llevado a cabo investigaciones enfocadas a analizar el uso de Twitter como medio para promover ideas o debatir temas políticos [45]. De tal modo, la mayoría de las investigaciones con *tweets* se ha centrado en la predicción de los resultados electorales [46, 47], así como la monitorización de campañas electorales, principalmente en Estados Unidos [48, 49].

Más allá de América, también en Europa se han realizado estudios interesantes sobre la relación entre Twitter y los resultados electorales. Es el caso de las elecciones federales alemanas de 2009, donde se analizaron 104003 *tweets* en los que se mencionaban a los principales candidatos políticos, traduciéndolos directamente al inglés y realizando un análisis LIWC [50]. Se descubrió que las menciones de partidos políticos era un reflejo de las preferencias del electorado. Por su parte, en el caso de las elecciones generales de Italia del año 2013, el análisis de Twitter mostró un buen indicador de los resultados finales. Así, el análisis de los *tweets* permitió detectar una fuerte presencia del inesperado partido ganador, aunque no era del todo correcto a la hora de predecir los resultados de los partidos más pequeños [51]. En otros casos, como las elecciones nigerianas de 2011 y las elecciones presidenciales paraguayas, venezolanas y ecuatorianas, se mostró como Twitter ejercía como un buen indicador de los resultados finales, especialmente cuando se analizaban las cantidades de mensajes mencionando a los candidatos [52, 53].

Sin embargo, los resultados ofrecidos por los análisis de Twitter no siempre fueron positivos. Es el caso de las elecciones al senado holandés de 2011 [54]. Asimismo, en el caso de las elecciones irlandesas de 2011, se realizó un análisis con resultados no del todo positivos, pues la precisión en la clasificación efectiva obtenida era del 65 % [55]. Posiblemente ello se debía a que los *tweets* estudiados no recogían los resultados obtenidos debido posiblemente a las diferencias de representaciones de los partidos políticos. Además, algunos defienden que las predicciones electorales mediante Twitter es como lanzar una moneda al aire, por lo que utilizarlo para predecir es más que controvertido [56, 57], de modo que para una predicción fiables es necesario realizar análisis más sofisticados.

Por otro lado, también se han realizado numerosos estudios enfocados a identificar patrones de comportamiento y no predecir resultados electorales. Así, en ellos se ha analizado el lenguaje de los mensajes de los distintos usuarios con el de los principales partidos políticos, mostrando que los usuarios más activos son menos propensos a cambiar sus preferencias [58].

Por otro lado, Twitter ha sido también utilizado para encontrar comunidades o fraccionamientos en el electorado. Un ejemplo es el caso de las elecciones presidenciales de Pakistán, donde se encontraron diferencias entre zonas urbanas y rurales, así como entre expatriados y personas continuaban viviendo en el país [59].

En el caso de la política española, Twitter se convierte en una herramienta muy utilizada durante los períodos electorales, como se muestra en [60], donde se analizan 370000 mensajes de 100000 usuarios durante las elecciones generales de 2011. Sin embargo, el análisis mostró que la mitad de los *tweets* fueron enviados el 7 % de los participantes, mientras que el 78 % de las menciones iban dirigidas a políticos [61]. Asimismo, existe una correlación entre el número de menciones de partidos políticos con los resultados electorales, aunque solamente en el caso de partidos con más de un 1 % de los votos.

También se han realizado estudios (aunque básicos) enfocados a elecciones regionales, como es el caso de las elecciones catalanas y las andaluzas [62,63]. En la primera se analizaron 84387 mensajes, descubriendo que los propios usuarios se agrupaban por afinidad política: Existe un agrupamiento ideológico. Sin embargo, a pesar de ello se hacía difícil predecir la preferencia política del usuario en base a sus seguidores y seguidos. Por su parte, en el caso de las elecciones andaluzas de 2012 se realizó un recuento del número de seguidores de las cuentas de Twitter de los dos principales partidos políticos de la época para predecir los resultados electorales. Los resultados fueron positivos, aunque en el caso de los partidos políticos más pequeños y sin tanta actividad en Twitter era totalmente ineficiente.

3.5. Minería de texto

Aunque hoy en día los datos basados en números se presentan como la principal forma para expresar resultados debido en gran parte a la facilidad que ofrecen a la hora de realizar una cuantificación de procesos o informaciones, existen otros tipos de datos menos estructurados pero que permiten generar una información mucho más completa, como es el caso de los datos textuales. El principal caso donde vemos esto es en las redes sociales, cuyos datos generados son primordialmente textuales. En ellas vemos como el uso de texto permite enfatizar o matizar opiniones, jugar con ironías, etc. Además, la propia red es una fuente inacabable de información textual, donde la utilización de técnicas de *web scraping* ofrecen una herramienta para obtener toda esa información. Todo esto hace que el 80 % de la información de interés empresarial sea encontrada en forma de texto libre y no-estructurado, tal y como se indica en [64].

3.5.1. Definición

En este contexto, la Minería de Texto surge dentro de un nuevo paradigma, denominado KDT (*Knowledge Discovery in Textual-Databases*), el cual fue introducido en [65] y es distinto al ya existente KDD (*Knowledge Discovery in Databases*), el cual estaba enfocado en la búsqueda de patrones dentro de datos completamente estructurados. Así, en este nuevo enfoque se pretende analizar y extraer información útil y no-explicita de datos textuales, carentes de estructura y homogeneidad, en los cuales los atributos son desconocidos, para lo cual se hace necesaria la utilización de algoritmos de *machine learning* sobre datos textuales [66].

Así, la Minería de Texto constituye una subcategoría de la Minería de datos, consistente en la obtención de información y descubrimiento e identificación de patrones, entidades y relaciones en datos de texto no-estructurados. Esta subcategoría es relativamente nueva y está aumentando notablemente su interés y utilidad, debido a las ingentes cantidades de información que día a día se generan en ellas, las cuales son principalmente de carácter textual. Además, constituye un área altamente interdisciplinar, donde se aúnan otras ramas destacadas como el procesamiento del lenguaje natural, la recuperación de información y la extracción de información, así como el uso de técnicas propias de *machine learning*.

La estructuración de los datos textuales mediante las técnicas de Minería de Texto permite dotar al texto de una forma capaz de ser estructurada y, por ende, estudiada mediante las técnicas de *machine learning* común. Así pues, las dos principales tareas de minería de datos utilizadas son la clasificación y *clustering*. En la primera de ella, se pretende asignar una categoría a cada texto o documento según sus características, mientras que en la segunda pretende generar grupos de documentos o textos similares en base a sus características [67].

3.5.2. Preprocesamiento

El preprocesamiento de texto se presenta como una de las fases esenciales en toda tarea propia de la Minería de Texto. La comunicación textual ofrece al usuario un amplio abanico de posibilidades y una gran libertad a la hora de expresarse, lo cual hace de los textos datos con una mayor cantidad de información intrínseca. Sin embargo, esta libertad supone que el preprocesamiento de textos no sea algo común, sino existan diversas posibilidades y que cada tipo de texto requiera de un análisis adecuado [68]. Por ejemplo, en el caso de datos generados en Twitter es común encontrar emoticonos con un sentimiento asociado.

Así pues, la Minería de Texto cuenta con técnicas de preprocesamiento propias, no aplicables en otras ramas, y tremadamente características [69]. Es el caso de técnicas como la tokenización, la lematización, la generación de N-gramas... Todas ellas suponen un intento de realizar una estandarización del texto, de generar una estructura común en textos no-estructurados, con el objetivo de conseguir que la extracción de la información sea mucho más sencilla y precisa.

Cabe destacar que, dentro de las opciones en *Python* de cara a la aplicación de técnicas de Minería de Texto y procesamiento del lenguaje natural, destacan dos: *NLTK* y *SpaCy*, *nltk2*, *spacy2*. Aunque la primera es la más conocida y presenta unos tiempos de cálculo mucho menores, la segunda ha resultado ser más interesante a la hora de identificar entidades y, sobretodo, a la hora de trabajar con textos en idiomas distintos al inglés.

A continuación, se detallarán todas estas técnicas destinadas a la estandarización del texto:

Tokenización

Como se indica en [70], la tokenización se presenta como una técnica que permite dividir toda oración en palabras, haciendo de estas como la entidad mínima con sentido. Esta entidad mínima es conocida como *token*. Aunque en el caso del español se hace muy sencillo, pues simplemente hay que tener en cuenta la separación entre palabras como espacios, en otros idiomas se hace mucho más complejo. Es el caso del inglés, donde se presentan contracciones en ciertas palabras, o del alemán, donde surgen palabras (verbos) separables que alcanzan un significado conjunto. A pesar de que el español es un caso sencillo, la utilización de **SpaCy** como herramienta para el análisis de texto nos permitirá no inmiscuirnos en esta técnica, siendo la propia librería la que lo realizará al analizar el texto.

Transformación a minúsculas

La transformación de los textos a minúsculas es una de las técnicas de preprocesamiento más comunes. En ella, todas las letras de cada palabras del texto son convertidas a su versión en minúsculas. Esta técnica permite estandarizar muchas palabras, que pasan a ser iguales, de manera que supone la reducción de la dimensionalidad del problema.

Presencia de caracteres repetidos

Tal y como se detalla en [71], existe una tendencia en los usuarios a buscar un énfasis mediante la repetición (de más de dos veces) de una misma letra dentro de una palabra de forma sucesiva. Es el caso de ejemplos como *Pleeeease* en inglés o *Bastaaaa!* en español. Así, los usuarios buscan dar una mayor intensidad al mensaje y alcanzar con su mensaje un impacto lo más grande posible. Este factor ha sido tenido en cuenta en estudios [72] y [42], de modo que su corrección se traduce en resultados tremadamente positivos. Por tanto, en este proyecto se corregirá este defecto, transformando las palabras que presenten dicho fenómeno en una versión reducida de la misma. Así, el término *Pleeeease* se reducirá a *Please* y el término *Bastaaaa!* a *Basta!*.

Presencia de *stopwords*

En todos los idiomas existe una serie de palabras que ejercen como conectores, cuantificadores u otras funciones, simplemente aclaratorias, pero que prácticamente no aportan información adicional al contenido del texto en lo que a nivel semántico se refiere. Este grupo de palabras es conocidas como *stopwords* y en él encontramos tipos de palabras como pronombres, preposiciones, etc.

En lo que a análisis de sentimientos se refiere, el estudio realizado por [72] concluye que, en el caso de los textos generados en Twitter, la eliminación de las *stopwords* apenas tiene relevancia, debido a que *tweets* son textos cortos. No obstante, hay opiniones contrarias a esta, pues en [42] se afirma que la eliminación de las *stopwords* de la bolsa de palabras es beneficiosa, pues supone una reducción de la dimensión de los términos.

Sin embargo, para el caso del análisis de asociaciones entre términos, la presencia de las *stopwords* sólo supondría la aparición de asociaciones inútiles y sin interés. Además, la presencia de estas palabras sólo se traduce en una disminución de la velocidad del análisis de texto, pues la cantidad de elementos a procesar por cada texto se hace notablemente mayor. Esto, especialmente en el caso de problemas de minería de flujo de datos, se hace tremadamente problemático, pues se requieren análisis que sean lo más rápido posible. Por tanto, una de las medidas comunes a realizar y, por supuesto, aplicadas en este proyecto es el filtrado de dichos *stopwords*.

Presencia de números y puntuaciones

La presencia de números en los textos siempre es común. Sin embargo, ella solo se utiliza para apoyar un mensaje con cifras. No parece que sea un elemento a tener en cuenta de cara al análisis de asociaciones y, por supuesto, no es interesante para el análisis de opiniones, pues no contienen ningún sentimiento. Aunque algunos destacan su utilidad para este último apartado [73] en el caso de textos de cierto tamaño, en el caso de textos cortos como los *tweets* parece suponer una mejora [74].

Por su parte, la eliminación de signos de puntuación es una técnica comúnmente aplicada durante el preprocesamiento. Ello se debe a que supone una reducción notable del problema, aunque en algunas ocasiones la presencia de signos de puntuación puede interpretarse como intensificadores [74].

Presencia de *links*

Uno de los elementos más comunes dentro de los *tweets* es la presencia de *links*, los cuales dirigen al usuario a una nueva página con más información. Aunque existe la posibilidad de acceder a la página, obtener el texto contenido en ella y asociarlo al *tweet* en cuestión, su realización sólo supondría una complicación de la búsqueda de asociaciones. Además, supondría acabar con el espíritu de analizar texto cortos y que daba a Twitter aún más encanto.

Lematización

Esta técnica permite dar una forma común a palabras con diferente derivaciones morfológicas (género y número) y distintas formas verbales. Esta técnica es aplicada con el objetivo de obviar la información morfológica de la palabra y centrar el estudio en la información semántica de la misma. Tal y como se indica en [71], la lematización es un método costoso, ya que requiere de un profundo conocimiento de la lingüística del idioma en cuestión. Sin embargo, su utilidad es notable, pues supone una reducción sustancial del ruido, mejorando los resultados del *Stemming*, aunque tomando un mayor tiempo de ejecución.

Stemming

El *Stemming* se presenta como otra técnica capaz de transformar una palabra a una forma común, aunque en este caso se consigue mediante la reducción de la palabra a su base o forma raíz. Así, nuevamente se pasa a destacar el apartado semántico de la palabra, olvidando su morfología, permitiendo reducir la dimensionalidad del problema.

Named-Entity-Recognition

El reconocimiento de entidades o *Named-Entity-Recognition* permite la extracción de entidades propias en el texto, como pueden ser nombres de personas, organizaciones, ubicaciones, expresiones de tiempos, cantidades, valores monetarios, porcentajes, etc. De este modo, dicha tarea permite agregar una gran cantidad de conocimiento semántico, así como comprender rápidamente el tema de cualquier texto.

POS (*Part-Of-Speech*) Tagger

El *Part-Of-Speech Tagger* o etiquetado de parte del discurso es una técnica utilizada para indicar la categoría gramatical de cada una de las palabras dentro de la oración. Así, esta técnica se utiliza para enfrentar a palabras con varios significados posibles, de manera que analiza la posición de la palabra en las oraciones para definir el significado real de la palabra, comprobando su situación y comparando con secuencias gramaticales frecuentes en los distintos idiomas.

Existen numerosos artículos donde se comenta que la utilidad de este tipo de técnicas para textos propios de *micro-blogs* (caso de Twitter) es prácticamente nula, debido principalmente a la brevedad del texto [75, 76]. Se ha tratado de aplicar en el proyecto en cuestión, de manera que, además, se han considerado dos nuevos tipos de entidades: **USER** y **HASHTAG**, asociados a la citación de usuarios o *hashtags* en los *tweets*, las cuales quedan indicadas por los signos *y #*, respectivamente. Así pues, para el trabajo a realizar se ha intentado aplicar el etiquetado con el objetivo de eliminar palabras que no sean verbos, adjetivos, nombres, entidades, usuarios o *hashtags*, reduciendo así la dimensionalidad del problema.

Otra tarea asociada al etiquetado de parte del discurso es la fragmentación de texto o *chunking*. Esta técnica permite mejorar la estructuración de las oraciones, dando lugar a la división del texto en fragmentos, de manera que se cuenta con elementos basados en agrupaciones de palabras. Para realizar dichas agrupaciones, esta técnica hace uso de la categoría gramatical de las palabras, de manera que analiza si las secuencias de categoría gramatical de las palabras presentes en el texto pueden dar pie a la definición de un grupo de palabras. Por tanto, la técnica de *chunking* es muy similar al reconocimiento de entidades o *Named-Entity-Recognition*, pues permite identificar entidades mediante las secuencias de categorías gramaticales. Además, cabe destacar que esta técnica presenta la ventaja de que permite realizar análisis superficiales más precisos.

Identificación de N-gramas

Por último, otro de los factores a tener en cuenta en el análisis de textos es la presencia de N-gramas. Definimos como N-gramas a grupos de palabras que pueden agruparse juntos, ya que tienen un significado de interés que hace que su aparición en la base de datos sea más elevada de lo normal. En el caso del análisis de asociaciones, esta técnica se convierte en un punto fundamental puesto que, si algún N-grama no es identificado, éste podría dar pie a la aparición de una asociación entre los términos que compondrían el N-grama no identificado. Ello podría suponer ocultar otras asociaciones de interés, por lo que este método se aplicará en el proyecto.

En resumen, la aplicación de este tipo de técnicas permite realizar un análisis del texto que permite pasar desde un elemento de texto sin ninguna clase de estructura a un documento completamente depurado y estructurado. Por tanto, como ya se ha comentado anteriormente, estas técnicas se hacen fundamentales de cara a la extracción de conocimiento desde documentos de texto.

3.6. Análisis de sentimientos

3.6.1. Definición

El **análisis de sentimientos** o **minería de opiniones** es un área de investigación enmarcada dentro del campo del procesamiento del lenguaje natural (NPL). Constituye una disciplina cuyo principal objetivo es la detección automática de opiniones, sentimientos y subjetividad dentro de un texto. De tal modo, persigue el reconocimiento de las emociones existente detrás de los textos analizados, buscando determinar la polaridad del mismo y la fuerza de la misma.

Así, se aplican distintos tipos de técnicas con el objetivo de representar la subjetividad u opinión existente en un texto mediante un valor cuantitativo en una cierta escala, lo cual permite clasificar dicho texto en un cierto tipo de sentimiento u opinión. En este contexto, una opinión es una valoración positiva o negativa acerca de un producto, servicio, organización, persona o cualquier otro tipo de ente sobre la que se expresa un texto determinado. En la actualidad, la popularización de las redes sociales de *micro-blogging* como Twitter, han aumentado el interés en este ámbito, de modo que se trata de conseguir una monitorización en tiempo real de las opiniones de miles de personas.

3.6.2. Enfoques para el análisis de sentimientos

Como se ha comentado, la minería de opiniones pretende identificar el sentimiento predominante en un cierto texto. Esta tarea constituye una labor computacional muy compleja, por lo que una correcta automatización del proceso requiere de un laborioso desarrollo y perfeccionamiento de forma continua. Así pues, se presentan dos enfoques principales:

- **Aproximación semántica:** Siguen una idea de aprendizaje no-supervisado, donde se utiliza un enfoque semántico basado en el uso de diccionarios de sentimientos (*lexicons*) con orientación de polaridad u opinión. De tal modo, se analiza el texto comprobando la aparición de los términos del *lexicons* para asignar su polaridad mediante la suma de los valores de polaridad ponderada de los términos. Así, mientras que algunos como [77–79] emplean diccionarios de palabras y frases de sentimiento con sus orientaciones e intensidades y trabajan la clasificación a nivel de frase y aspecto, otros como [80] incorporan una intensificación y el concepto de negación para realizar el cálculo final.
- **Enfoque Machine Learning:** Se cuenta con una serie de textos de entrenamiento cuyo sentimiento ha sido etiquetado, los cuales ejercerán como conjunto de entrenamiento, de modo que con ellos se puede entrenar un modelo de *machine learning* para posteriormente analizar otros textos sin etiquetar [81]. Para entrenar estos modelos se presta atención a distintas características como categorías gramaticales, palabras y

frases de sentimiento, reglas de opinión, modificadores de sentimiento, dependencias sintácticas... Entre los distintos métodos empleados destacan Naive-Bayes y SVM.

Además, se distinguen tres niveles de estudio de un texto a la hora de realizar el análisis de sentimientos, definidos en base a granularidad, profundidad y detalle requeridos. Dichos niveles son:

- **Análisis a nivel de documento:** Se analiza el sentimiento global de un documento como un todo indivisible. Se asume que dicho documento expresa una valoración sobre una única entidad, de modo que no es aplicable en casos en los que se hable simultáneamente sobre varias entidades.
- **Análisis a nivel de oración:** Se divide el documento en oraciones individuales para extraer posteriormente la opinión que contiene cada una de ellas.
- **Análisis a nivel de aspecto:** Se considera que una entidad está formada por distintos elementos o aspectos y sobre cada uno de ellos se expresa una opinión cuya polaridad puede ser distinta en cada caso.

Sin embargo, la brevedad que se presenta por lo general en los *tweets* (la limitación inicial a 140 caracteres hace que se tiendan a escribir mensajes cortos) supone que en el problema tratado se considere un análisis a nivel de documento, asumiendo que cada mensaje expresa la opinión del usuario, tomando que sólo existe una única entidad.

3.6.3. Aplicaciones

La minería de opiniones se presenta como una herramienta muy útil y valiosa tanto para empresas y organizaciones como gobiernos, pues permite saber en cada momento qué piensa la gente sobre productos, medidas o acciones. Asimismo, la monitorización de las redes sociales y el análisis de las opiniones sobre determinados temas, puede ayudar a detectar la gestación de determinados acontecimientos sociales como huelgas. En concreto, a continuación se detallan algunas de las aplicaciones del análisis de sentimientos:

- **Valoración de opinión de productos y servicios:** El análisis de sentimientos de las opiniones en *blogs* y redes sociales permite a las empresas conocer la opinión de los usuarios acerca de sus productos sin necesidad de llevar a cabo estudios como las tradicionales encuestas de satisfacción. Así, en caso negativo, las empresas pueden replantear estrategias en el menor tiempo posible otorgando así ventajas competitivas.
- **Análisis para valoración y recomendación:** El análisis de sentimientos podría analizar las palabras las opiniones de usuarios en sus reseñas acerca de productos,

siendo capaz de analizar un posible error en la puntuación determinada manualmente por el usuario. Además, en base a ello se podrían priorizar las recomendaciones de un producto u otro.

- **Reputación política:** La minería de opiniones permite conocer la opinión de la gente a lo largo del tiempo acerca de un determinado partido político o un cierto candidato, así como la acogida de determinadas políticas propuestas. En este ámbito destaca notablemente el uso como fuente de datos de redes sociales como Twitter.
- **Análisis del mercado financiero:** El análisis de la información en la web y en redes sociales sobre una empresa permite prever su evolución en el mercado financiero a partir del valor agregado de la polaridad de todas las opiniones encontradas.

3.6.4. Análisis de Sentimientos en castellano

Aunque la cantidad de herramientas disponibles para la clasificación de sentimientos en inglés es notable y hay algunas bastante precisas, para el caso del otros idiomas no todo está tan avanzado. Por tanto, a la hora de enfrentar el análisis de sentimientos para el caso de *tweets* sobre política española aparece la dificultad de realizar una determinación adecuada del sentimiento resultante de los mensajes. Por ello, el análisis de este problema para otros lenguajes se ha convertido en un tema de interés en la investigación. En el caso del castellano, durante la última década aparecen novedosos análisis como es el caso de [72, 82, 83].

Una primera opción posible es la utilización de *Lexicons* de palabras etiquetadas como positivas y negativas. Como ejemplos de este tipo de *lexicons* en español aparecen algunos como el *ElhPolar dictionary* [84], *iSOL* [85] o *ML-SentiCON Sentiment Spanish Lexicon* [86]. Además, es posible jugar con la realización de la traducción del texto y clasificar utilizando algún *Lexicon* disponible en inglés, donde se cuenta con más posibilidades. Sin embargo, a pesar de que este tipo de métodos se presenta como una de las opciones más explotadas para el análisis de sentimientos en español [87], la sencillez de todos estos métodos juega en su contra, pues los resultados obtenidos a la hora de realizar la clasificación de sentimientos no son del todo positivo. Por tanto, se ha desecharo utilizar esta opción.

Otra de las opciones posibles consiste en utilizar un *corpus* para crear un *ensemble learner* capaz de clasificar *tweets*. Una de las opciones posibles es utilizar el *dataset* del *TASS*, el cual está compuesto por 68000 mensajes de Twitter en español y escritos por 150 personalidades destacadas en ámbitos como la política, la economía, la comunicación, la cultura... De ellos, sólamente unos 2500 son relacionados con política, por lo que quizás la presencia de *tweets* propios de otros ámbitos reduzca la capacidad de clasificación de sentimientos.

Las otras opciones analizadas consiste en la utilización de herramientas o *software* capaz de realizar la clasificación de sentimientos de un texto. Para ello, en general, dichas herramientas han sido entrenadas utilizando *corpus* de mensajes o agrupaciones de *lexicons* que hacen que permiten contar con una riqueza capaz de realizar una clasificación bastante adecuada. A continuación, se detallan los analizados, con sus ventajas e inconvenientes:

- **senti-py:** Este clasificador de sentimientos en español [88] ha sido pre-entrenado utilizando fuentes como *TripAdvisor*, *PedidosYa*, *Apestan*, *QuejasOnline*, *MercadoLibre*, *SensaCine*, *OpenCine*, *TASS* o Twitter. En él se presenta la polaridad del texto con un valor entre 0 y 1, donde 1 indica totalmente positivo y 0 totalmente negativo. Aunque esto tiene la ventaja de que permite dividir la polaridad en los grupos que se deseé, las fuentes no son las más adecuada para la tarea que se pretende realizar (pues no son fuentes de contenido político). Además, su funcionamiento no es el mejor, de modo que en las comprobaciones realizadas se muestran resultados confusos.
- **MeaningCloudClassifier:** Esta API [89] disponible en *Python* es capaz de reconocer *SENTIMENTO*, *SUBJECTIVIDAD* e *IRONIA*. Asimismo, también permite reconocer entidades en el texto, como por ejemplo partidos políticos e incluso doctrinas políticas. Aunque tiene un funcionamiento relativamente correcto, presenta la limitación de que la polaridad siempre se define como *AGREEMENT* y *DISAGREEMENT*, sin considerar un término neutro. Parece que ha sido entrenado con los *tweets* propios de la base de datos del *TASS*.
- **AutoCop:** Se trata de un proyecto de la USAL, el cual cuenta con una versión disponible en *Python*, *autocop*, *Arcila-Calderon2017*. Este modelo ha sido entrenado con 8000 tweets políticos y utilizando un ensemble de clasificadores con una precisión del 77% en clasificación. Este clasificador ofrece un *sentiment_value* en el que se indica una polaridad positiva (*POS*) o negativa (*NEG*), así como un valor *confidence*, el cual indica el grado de seguridad de la polaridad definida. Gracias a dicho parámetro *confidence* sería posible realizar una división de la polaridad en positivo, negativo y neutro.
- **SentiStrength:** Este *software* [90] es utilizado para calcular la intensidad del sentimiento presente en textos propios de medios sociales y ha sido utilizado en numerosas investigaciones acerca de comunicaciones políticas [91, 92]. Para ello, asigna tres valores a cada texto: Uno para la intensidad de sentimientos positivos, otro para los sentimientos y otro para los sentimientos negativos, de modo que todos ellos presentan valores entre 1 y 5, siendo este último el valor de máxima intensidad. Cuenta con una versión española validada en [45] en el contexto de las elecciones generales de 2015, mostrando una precisión del 81% para el sentimiento positivo y un 84.8% para el sentimiento negativo.

- **Lingmotif (SentiText)**: Este *software*, propio de la UMA, permite analizar textos tanto generales como especializados, utilizando léxicos generados por el usuario [93]. Es capaz de realizar el análisis de sentimiento tanto en español como inglés, utilizando tres fuentes de datos: un léxico de palabras individuales, un léxico de frases y una serie de reglas de contexto. Sin embargo, la utilización de este *software* requiere de acceso autorizado que nunca se recibió.
- **apicultur**: Este *software* [94] ha sido utilizado como clasificador de sentimientos en estudios como el de las elecciones colombianas de 2015 [95], donde muestra un funcionamiento bastante correcto. Para ello, devuelve un sentimiento (que puede ser *Positivo*, *Negativo*, y *Neutral*) y un valor de intensidad entre 0 y 5.

Así pues, se ha realizado una comprobación del funcionamiento de los distintos clasificadores de sentimientos comentados, de manera que se ha podido analizar su capacidad, así como sus ventajas e inconvenientes y las posibilidades que ofrece cada uno de ellos. Tras dicho estudio se ha decidido hacer la clasificación de sentimientos mediante el *software SentiStrength*, el cual se describe en el **Apéndice C**. Esta decisión ha quedado también respaldada por los resultados de diversos estudios teóricos, como los realizados en [92] o [96], donde se presenta a este *software* como una herramienta destacada en dicha tarea.

3.6.5. Análisis de Sentimientos en Twitter

La Minería de texto y el Análisis de Sentimientos se presentan como dos herramientas de gran utilidad para analizar, en especial, textos o documentos muy largos. No obstante, en el caso del análisis de redes sociales se ofrece una buena herramienta para la extracción de conocimiento, ya sea para clasificación o *clustering*, e incluso a la hora de realizar predicciones. Durante las últimas décadas se han realizado numerosas investigaciones en el ámbito de análisis de sentimientos aplicados a Twitter.

Inicialmente fue tomado como una clasificación binaria, positivo-negativo, tal y como se propuso en [97], donde se asignaba un sentimiento utilizando una colección de *tweets* con anotaciones en base a emoticonos, tras lo cual se aplicaron técnicas de minería de texto, como localización de N-gramas e identificación de categorías gramaticales, y de *machine learning*, como *Naive-Bayes*. En [98] se comparan la utilización de los modelos de *Naive-Bayes* y de máxima entropía para la clasificación de *tweets*, mostrando como el primero funciona notablemente mejor. En [99] se propone un detector basado en dos fases, en cada una de las cuales es capaz de clasificar *tweets* como objetivos o subjetivos y como positivos o negativos. En él se utiliza información como *hashtags*, *retweets*, signos de puntuación e identificación de categorías gramaticales.

Por su parte, en [100] se utilizan modelos de Máquinas de Soporte Vectorial (SVM) con diversas características, como N-gramas, identificación de categorías gramaticales, *hashtags*, emoticonos y negaciones, así como un preprocesamiento en el que eliminar menciones, *links* y palabras alargadas más tokenización. Asimismo, en [101] se presenta un clasificador basado en el algoritmo de regresión logística para predecir el sentimiento del *tweet*. Para ello, se utiliza un preprocesamiento similar al de [100], además de un corrector de escritura y un *desambiguador* del significado de las palabras. Por otro lado [102] utiliza un *ensemble* de clasificadores, combinando los modelos *Naive-Bayes*, máxima entropía y SVM. Mientras que en [75] se utilizan y comparan diversos algoritmos para determinar la polaridad de los *tweets*, otros como [103] se utiliza un *corpus* donde los mensajes son clasificados en base a los emoticonos presentes en ellos.

En el ámbito del aprendizaje no supervisado destaca [104], quien realiza un análisis de sentimientos basado en un *lexico*, de manera que cuenta con una bolsa de palabras (BOW). Así, cada mensaje pasa a ser representado sólamente por aquellas palabras que aparecen en dicha lista de interés, las cuales tienen asociado un cierto grado de polaridad. De tal modo, el sentimiento de un mensaje será el resultado de la suma total de las polaridades de las palabras que representan el mensaje, pudiendo ser éste positivo, negativo o neutro. Este método presenta una elevada simplicidad, lo que hace que no sea capaz de detectar sarcasmos y frases hechas, además de no ser capaz de lidiar con palabras con varias acepciones.

En lo que a análisis en tiempo real se refiere, en [6] se aplican técnicas de clasificación para análisis de sentimientos sobre flujos de datos de Twitter obtenidos de *Firehouse API*. Utilizaron técnicas como *Naive-Bayes* multinomial, *Hoeffding tree* y descenso de gradiente estocástico (SDG), siendo este último el que mejores resultados ofrecía. Por su parte [105] diseño un modelo capaz de detectar sentimientos en mensaje de *microblogging* de forma automática. Por su parte, en [106] se estudia el análisis de opinión *online* para mensajes de redes sociales utilizando el algoritmo *Winnow* para el aprendizaje de un clasificador lineal a partir de ejemplos pre-etiquetados. Este algoritmo también ha sido integrado en MOA.

Por su parte, en cuanto al análisis de sentimientos enfocado a política y Twitter, destaca el trabajo realizado por [49], quien desarrolló un modelo pionero capaz de realizar el análisis de sentimientos en tiempo real en las elecciones presidenciales estadounidenses de 2012. Asimismo, en [45] se realiza un análisis en tiempo real de *tweets* políticos en España, analizándose la evolución de los sentimientos existentes sobre los principales partidos y personajes políticos mes a mes y utilizando una versión para español del *SentiStrength* como clasificador.

4. Análisis de resultados

A continuación, se procede a detallar el estudio experimental realizado. Se ha tratado de seguir una estructura de análisis fija, comenzando con un procesamiento de texto y análisis de sentimientos (si lo requiere), para posteriormente dar paso al análisis de flujo de datos en busca de las reglas de asociación. Para este último paso existen dos vías posibles. Todos los pasos realizados quedan recogidos en la figura 7:

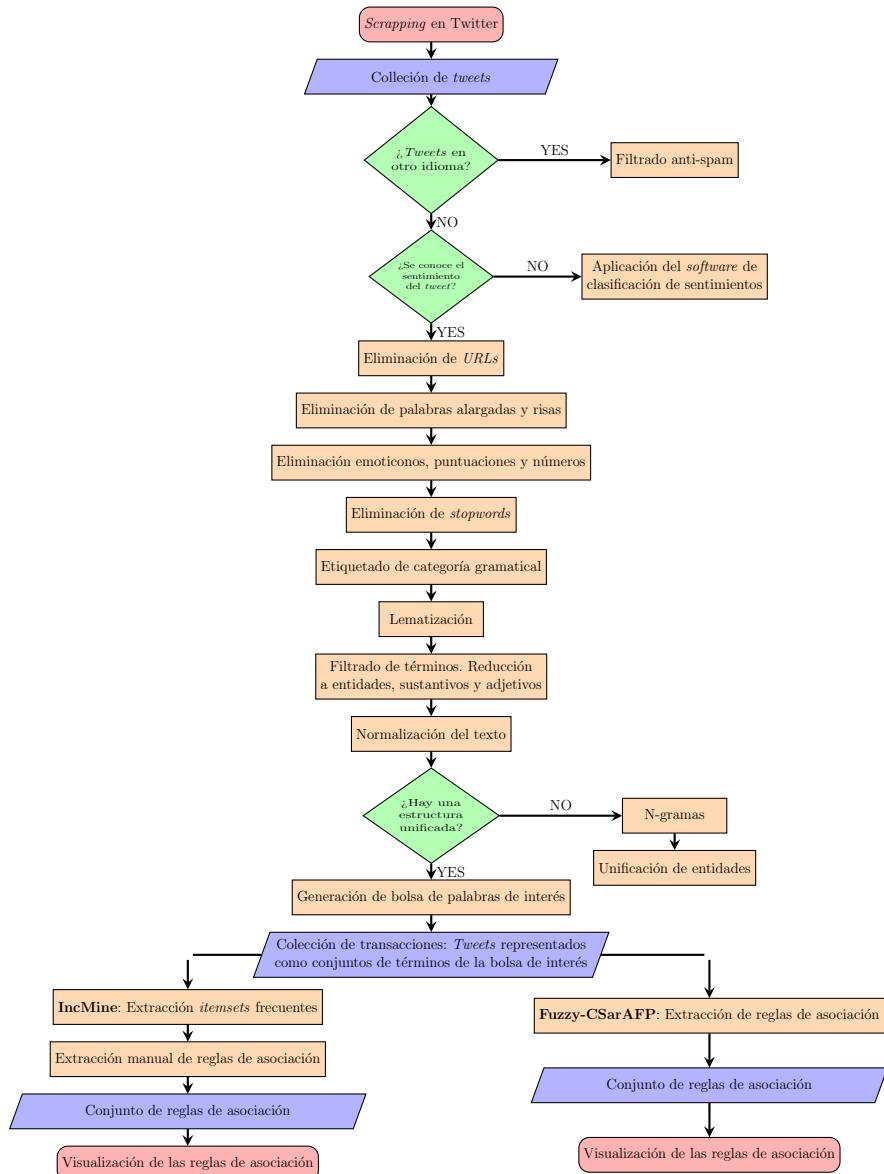


Figura 7. Diagrama de flujo del esquema seguido en el estudio realizado.

El estudio realizado ha sido aplicado sobre dos conjuntos de datos, ambos con carácter políticos: Uno enfocado en las elecciones políticas de Estados Unidos en el año 2016 y otro relativo al proceso de investidura de España durante el verano de 2019. El primero es un *dataset* más pequeño y ya etiquetado, de modo que ha sido utilizado para el estudio inicial. El segundo presenta un *dataset* con datos reales, extraídos de Twitter durante un mes. Así pues, se pasa a describir el estudio realizado, junto con las distintas consideraciones tomadas. Tras ello, se muestran los correspondientes resultados obtenidos con ellos.

4.1. Aplicación en las Elecciones Políticas de 2016 en USA

4.1.1. Introducción al conjunto de datos. Análisis exploratorio

En primer lugar, como ejercicio introductorio, se ha decidido utilizar un *dataset* ya etiquetado en lo que a clasificación de sentimientos se refiere. De tal modo, se ha elegido el dataset *The SemEval-2016 Stance Dataset*, relativo a las elecciones políticas de Estados Unidos en el año 2016 y disponible en [107]. A continuación, se realizará una descripción de los distintos atributos asociados a los distintos *tweets* que aparecen en dicho conjunto de datos:

- ***ID***: Número identificativo del *tweet*.
- ***Target***: Indica la entidad de interés en la que se engloba el *tweet*
- ***Tweet***: Mensaje contenido en el *tweet*
- ***Stance***: Indica la actitud del usuario que ha escrito el *tweet*. Hay tres posibles posturas:
 1. *FAVOR*: El usuario apoya (directa o indirectamente) al objetivo.
 2. *AGAINST*: El usuario está en contra del objetivo (directa o indirectamente).
 3. *NONE*: Ninguna de los anteriores.
- ***Opinion_towards***: Es el objetivo de la opinión expresada en el *tweet*. Hay tres posibles objetivos:
 1. *TARGET*: El mensaje expresa una opinión explícita sobre la entidad de interés o un aspecto del mismo.
 2. *OTHER*: El mensaje expresa una opinión sobre algo distinto a la entidad de interés.
 3. *NO ONE*: El mensaje no va dirigido a opinar. Puede ser una información.

- **Sentiment:** Indica el sentimiento presente en el *tweet*. Hay tres posibles sentimientos:

1. *POSITIVE*: Muestra un lenguaje positivo, basado en la presencia de señales de admiración y actitudes positivas.
2. *NEGATIVE*: Muestra un lenguaje negativo, basado en la presencia de críticas y actitudes negativas o de dudas.
3. *NEITHER*: Ninguna de los anteriores.

Así pues, se ha realizado un análisis de las distribuciones de algunos atributos de interés, como es el caso del sentimiento presente en los *tweets* o el objetivo de los mismos. A continuación, se muestra dichas distribuciones en la figura 8:

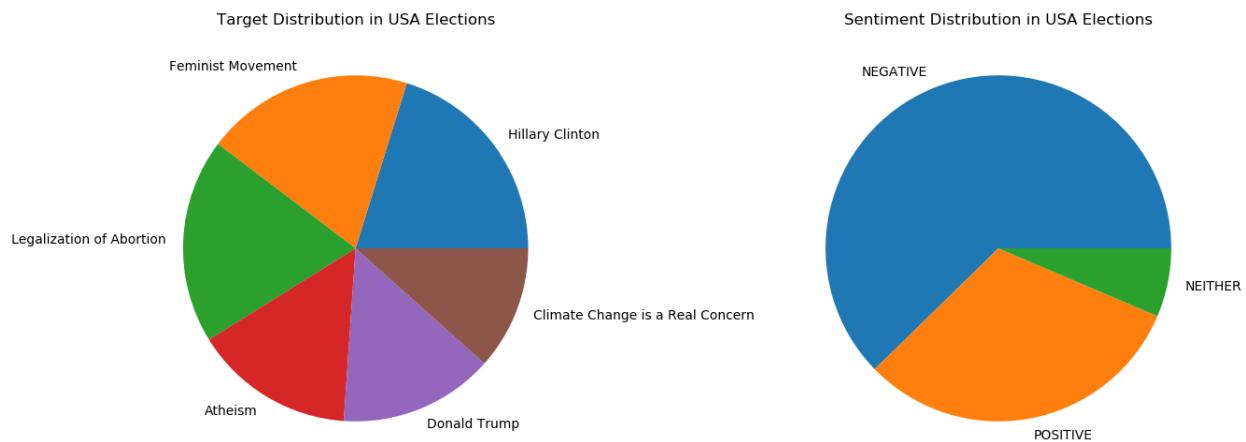


Figura 8. Gráficos con las distribuciones del objetivo(izquierda) y sentimiento(derecha) presentes en los *tweets* a estudiar.

Se aprecia como existe una predilección a publicar mensajes de carácter negativos entre los mensajes a estudiar. Además, solamente una pequeña proporción de los mensajes no expresan ningún sentimiento claro. Por otro lado, la distribución de los objetivos de los mensajes de la base de datos de estudio muestra una división bastante pareja.

Sin embargo, la distribución de los temas de interés no es constante a lo largo del tiempo en el *dataset* estudiado. Existen variaciones en el interés mostrado en los distintos temas por parte de los usuarios de la red social, los cuales pueden estar influidos por posibles noticias publicadas por los medios o declaraciones realizadas por personajes con cierta repercusión. Por tanto, se ha comprobado la evolución de la frecuencia de aparición de estos objetivos de interés, la cual se muestra en la figura 9:

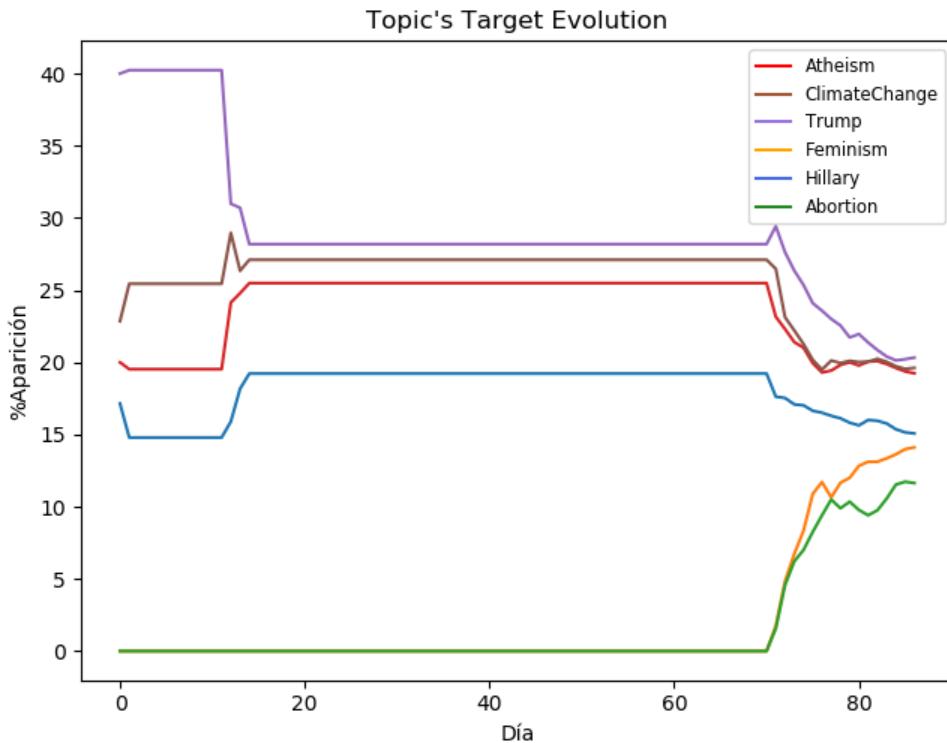


Figura 9. Gráfico con la evolución temporal de la frecuencia de publicación sobre los temas de interés presentes en los *tweets* del problema de USA.

Se aprecia como el reparto de los temas de interés en los mensajes del conjunto de datos evoluciona a lo largo del tiempo desde una situación donde sólo unos pocos temas absorben todo el interés hasta un punto en el que existe un reparto más equitativo. Además, se ha realizado un gráfico de barras dinámico con esta evolución, el cual es posible encontrar en <https://drive.google.com/open?id=1JCbLdEVojz9B-xSwuJwEfItQDRNNKTjt>.

4.1.2. Procesamiento del Lenguaje Natural

En primer lugar, ha sido necesario realizar una transformación de los *ID* de los mensajes de Twitter, compuestos por una estructura de 64-*bits*, generados con *snowflake*. Así, de los 64 *bits* con los que cuenta cada *ID*, 41 *bits* son dedicados a la fecha con precisión de milisegundos, más un identificador de máquina o trabajador de 10 *bits* y un número secuencial de 12 *bits* basado en un contador local. Por tanto, se ha utilizado la herramienta disponible en [108], capaz de decodificar la fecha de envío de cada *tweet*. Así, se ha aprovechado esta para estudiar la parte del día en la que se publicó el mensaje (considerando los intervalos de mañana, tarde y noche).

Tras ello, se ha aplicado sobre los *tweets* un análisis de texto, similar al presentado en el marco teórico. Así pues, se ha comenzado eliminando *links*, números, puntuaciones (a excepción de los # y @ que hacen referencia a *hashtags* y usuarios, respectivamente), emoticonos y risas de los mensajes. También se ha eliminado la aparición de la misma letra de forma múltiple (más de dos) y consecutiva, reduciendo estas a simplemente dos apariciones consecutivas.

A continuación, se ha utilizado **SpaCy** [109] como herramienta para el análisis de texto. Por tanto, se ha comenzado eliminando las *stopwords* y palabras cortas (de longitud menor a 3). Asimismo, se ha aplicado la técnica de *POS-Tagger*, realizando un etiquetado gramatical sobre las palabras. Además, cabe destacar que para el caso de los *hashtags* y usuarios se ha utilizado dos nuevas etiquetas gramaticales, definiendo así las etiquetas *HASHTAG* y *USER*, respectivamente. Posteriormente, se ha aplicado la lematización sobre el texto, a excepción de aquellos términos que hacen referencia a *hashtags* y usuarios, pues estos son considerados como entidades invariables.

Tras ello, se ha optado por la simplificación del texto, mediante la eliminación de las palabras en base a su categoría gramatical. Mientras que los verbos suelen cuantificar la actividad-pasividad en el contenido del mensaje, los adjetivos tienden a ser útiles a la hora de identificar el sentimiento presente en el texto. Por tanto, *a priori* no son elementos de interés en un texto. Puesto que en este problema se pretende encontrar relaciones entre términos, sólamente son importantes los sustantivos y entidades, además de los usuarios y *hashtags*. Sin embargo, dado que en ocasiones los adjetivos pueden ser útiles a la hora de discernir conceptos, también han sido finalmente considerados.

A continuación, se ha realizado el análisis de *N-gramas* en el texto, buscando bigramas y trigramas. Tras analizar los trigramas obtenidos, el único término de interés es *anti_choice_law*, el cual hace referencia a la ley contra el libre aborto. En el caso de los bigramas, se han encontrado una serie de conceptos, los cuales se muestran en la tabla 1:

Bigrama	Concepto
(donald, trump)	Candidato republicano.
(climate, change)	Cambio climático.
(hillary, clinton)	Candidata demócrata.
(woman, right)	Derechos de las mujeres.
(pro, life)	Pro-vida. Detractores del libre aborto.
(equal, right)	Igualdad de derechos.
(white, house)	Casa blanca. Gobierno de los EEUU.
(pro, choice)	Pro-elección. Defensores del libre aborto.
(human, right)	Derechos humanos.
(global, warming)	Calentamiento global.
(unborn, child)	Niños no-natos. Fetos.

Bigrama	Concepto
(gay, marriage)	Matrimonio homosexual.
(supreme, court)	Tribunal Supremo de los EEUU.
(united, states)	Estados Unidos.
(birth, control)	Control de natalidad.
(marriage, equality)	Igualdad de matrimonio.
(confederate, flag)	Bandera de EEUU
(sea, level)	Nivel del mar.
(david, attenborough)	Científico y divulgador británico.
(human, being)	Seres humanos.
(social, medium)	Medios sociales.
(mother, teresa)	Madre Teresa de Calcuta.
(death, penalty)	Peña de muerte.
(gender, equality)	Igualdad de género.
(preborn, child)	Niños no-natos. Fetos.

Cuadro 1. Tabla con los distintos bigramas encontrados en el caso de USA y su definición.

Tras ello, se ha procedido a la unificación de entidades, pues se ha podido comprobar que hay distintas formas de citar o hacer referencia a entidades de interés en el *dataset* bajo distintos términos. En la tabla 2 se muestran las entidades consideradas, así como los términos englobados bajo dicha entidad:

Entidad	Términos asociados
donald_trump	@realdonaldtrump, donald_trump, mr_trump, trump, #trump, donaldtrump, #donaldtrump #hillaryclinton, hillaryclinton, hillary, clinton, hilary, #stophillary, #stophillarypac, hillary_clinton, #readyforhillary, #nohillary, #killary
hillary_clinton	
barack_obama	@barackobama, barackobama, obama, #obamas
democrat	@thedemocrats, @vademocrats, democrat, #democrats, #democrat
republican	republican, #republican, #republicans, #republicanvalues, republicans
USA	unitedstates, united_states, usa, @unitedstates, #unitedstates
SCOTUS	scotus, #supremecourt, supremecourt, #scotus, supreme_court
david_attenboroug	david_attenboroug, @sir_attenboroug, davidattenboroug, attenborough

Cuadro 2. Tabla con las distintas entidades consideradas y los términos englobados en ellas.

Además, aunque en un principio se pensó en no aplicar ningún tipo de lematización sobre los *hashtags*, considerando estos como entidades invariantes e inamovibles, se ha comprobado como en ocasiones estos *hashtags* son incluidos en el texto como parte del mensaje, más allá de su función de etiqueta original. Por tanto, resulta lógico pensar que, para incluirlos más fácilmente en la estructura del texto a través de la bolsa de palabras, estos sean lematizados. Una vez lematizados estos *hashtags*, se ha vuelto a proceder a la unificación de términos de interés de la base de datos, con el objetivo de enriquecer la aparición de los términos de interés en los *tweets* disponibles. Así pues, en la tabla 3 se muestran los distintos términos unificados bajo cada concepto:

Concepto	Términos asociados
prolife	<i>prolife, pro_life, prolifeyouth, prolifegen, alllivesmatter</i>
prochoice	<i>prochoice, pro_choice, prowomanchoice</i>
feminism	<i>feminist, feminism, genderequality, gender_equality</i>
antifeminism	<i>antifeminism, antifeminist, feminazi, notafeminist, meninist, spankafeminist</i>
misogyny	<i>misogyny, misogynism, misogynst, misogynist, misogynistic, misogynyisugly</i>
immigrant	<i>immigrant, immigration, latino, hispanic</i>
child	<i>child, kid, baby</i>
unbornchild	<i>unborn_child, preborn_child, unborn, preborn, fetus</i>
pregnant	<i>pregnant, pregnancy</i>
equality	<i>equality, equalityforall, equal_right, equalright, eaquality</i>
marriageequality	<i>marriageequality, marriageequality, marriage_equality</i>
gaymarriage	<i>scotusmarriage, gaymarriage, gay_marriage</i>
woman_right	<i>woman_right, womensright</i>
lgbt	<i>lgbt, gay, homosex, homosexual, homosexuality, lesbian</i>
rape	<i>rape, rapeculture, rapist, maritalrape, maritalrapedebate</i>
sexism	<i>sexism, sexist</i>
racism	<i>racism, racist</i>
climate_change	<i>climate_change, global_warming, climate, climatehope, climatenexus, emission, climatechangeisreal, mychangeforclimate, ecologyaction</i>
freedom	<i>freedom, liberty</i>
science	<i>science, sciencerule, scientist</i>
anti_choice_law	<i>anti_choice_law, antichoice</i>
catholic	<i>catholic, romancatholic, church, christian, christ, christianity, bible, jesus, teamjesus</i>
atheism	<i>atheism, agnostic, atheist, atheistq</i>
islamic	<i>islamic, islam, isis, islamicstate</i>
man	<i>man, male</i>
woman	<i>woman, girl, female, yesallwoman, yesallwomen</i>
mexicanpeople	<i>mexicanpeople, wearemexico, mexico, mexican</i>

Cuadro 3. Tabla con los distintos términos de interés unificados bajo un mismo término.

Una vez unificados los términos y elementos de los mensajes de la base de datos, se ha procedido a crear una bolsa de elementos de interés, en la cual se definen los términos interesantes dentro de la base de datos en base al concepto al que hacen referencia y a su frecuencia de aparición. Esta ha sido creada de forma manual por el autor, de modo que ha elegido los términos que se veía con cierta relevancia en los debates realizados en los mensajes de Twitter del *dataset*. La lista de palabras incluidas dentro de la bolsa de interés quedan recogidas en la figura 4:

<i>hillary_clinton</i>	<i>woman</i>	<i>donald_trump</i>	<i>feminism</i>	<i>man</i>
<i>catholic</i>	<i>child</i>	<i>abortion</i>	<i>equality</i>	<i>climate_change</i>
<i>prolife</i>	<i>barack_obama</i>	<i>rape</i>	<i>unbornchild</i>	<i>religion</i>
<i>islam</i>	<i>pregnant</i>	<i>freedom</i>	<i>marriage</i>	<i>liberal</i>
<i>democrat</i>	<i>black</i>	<i>antifeminism</i>	<i>woman_right</i>	<i>republican</i>
<i>family</i>	<i>racism</i>	<i>sexism</i>	<i>immigrant</i>	<i>murder</i>
<i>mexicanpeople</i>	<i>science</i>	<i>religious</i>	<i>gaymarriage</i>	<i>prochoice</i>
<i>atheism</i>	<i>justice</i>	<i>human_right</i>	<i>marriageequality</i>	<i>crime</i>
<i>obamacare</i>	<i>misogyny</i>	<i>patriarchy</i>	<i>violence</i>	<i>muslim</i>
<i>anti_choice_law</i>	<i>conception</i>	<i>humanist</i>	<i>lgbt</i>	<i>birth_control</i>

Cuadro 4. Conjunto de elementos de interés considerados para la base de datos de USA.

Además, en la figura 10 es posible apreciar dichos dentro de una nube de palabras (*word cloud*):

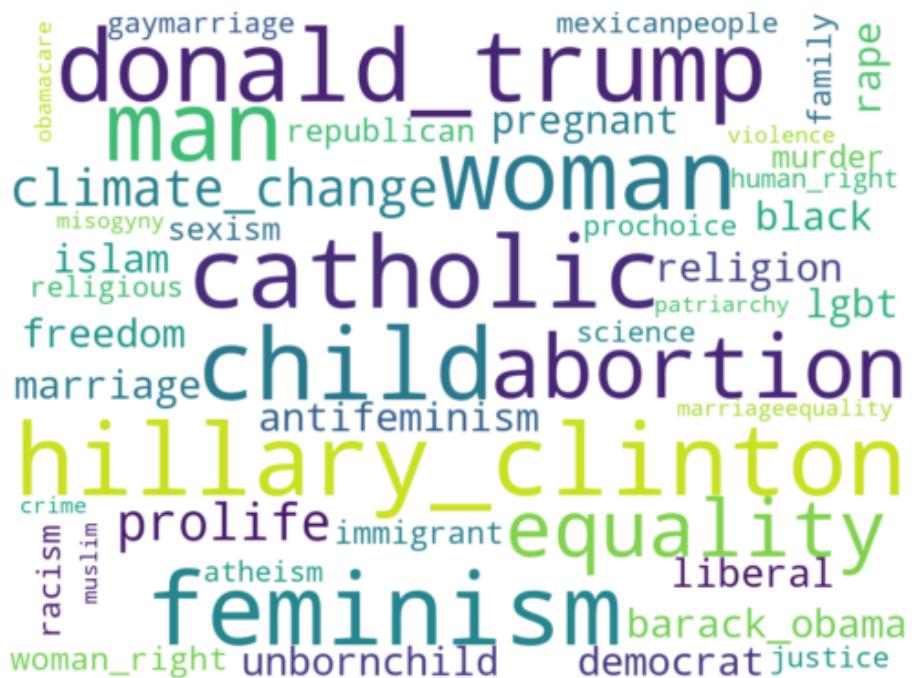


Figura 10. *Word cloud* con las palabras elegidas para la bolsa de interés del problema de las elecciones de USA.

Asimismo, es posible analizar la frecuencia de aparición de dichos términos, de manera que es posible comprobar como los candidatos a la presidencia, junto con los temas más candentes y cuyas políticas estaban a debate en la sociedad estadounidense durante dicho año, aparecen con una frecuencia mucho mayor, mientras que otros términos más periféricos aparecen con menor frecuencia. Esto queda recogido en la figura 11:

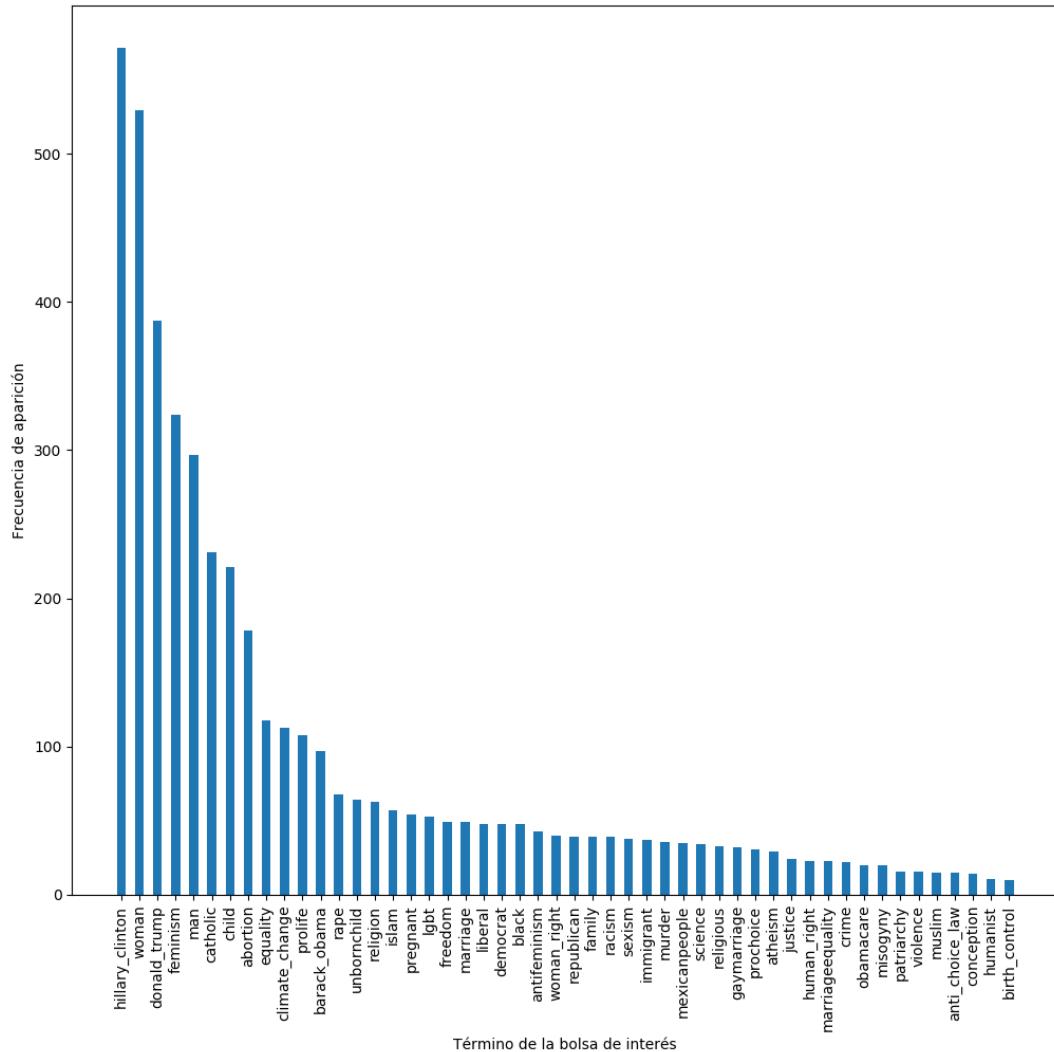


Figura 11. Histograma con la frecuencia de aparición de cada una de las palabras elegidas para la bolsa de interés en el problema de las elecciones de USA.

Tras ello, cada *tweet* de la base de datos queda representado mediante una secuencia de palabras de las presentes en la bolsa de interés. Así pues, en este punto cada *tweet* puede ser visto como una transacción, mientras que cada una de las palabras de la bolsa de interés es vista como un *item*, los cuales pueden aparecer en conjuntos, formando *itemsets*.

Así pues, es posible trabajar sobre este conjunto de transacciones con los términos de interés en busca de los *itemsets* frecuentes y las reglas de asociación existentes entre los distintos términos, lo cual se realizará a continuación.

4.1.3. *Association Stream Mining*: Extracción dinámica de reglas de asociación

Una vez se ha conseguido estructurar los datos textuales y generar una base de datos transaccional en base a la representación de los *tweets* mediante los términos de la bolsa de interés definida, ya es posible pasar a la etapa definitiva. En esta etapa se extraerán las reglas de asociación entre términos, realizando un aprendizaje en tiempo real mediante la aplicación algoritmos propios de minería de flujo de datos. En este punto, tal y como se indica en la figura 7, existen dos vías de acción:

- Por un lado, es posible aplicar un enfoque basado en la extracción y actualización incremental de *itemsets* cerrados frecuentes (FCIs) sobre flujos de datos, mediante el uso de *IncMine* [38]. Tras ello, es posible realizar la extracción de las reglas de asociación de forma *offline* mediante el análisis de estos *itemsets* frecuentes y de todas las reglas posibles.
- Por otro lado, es posible realizar la extracción directa de las reglas de asociación presentes mediante la aplicación de una técnica difusa y completamente *online*, como es el caso del algoritmo *Fuzzy-CSarAFP* [24].

Tras la recolección de las reglas, se estudiarán para buscar vínculos entre la presencia de términos y sentimientos. De tal modo, se hará un recuento del número de reglas generadas para cada par antecedente más consecuente, buscando así realizar un análisis la existencia de relaciones término-consecuente (donde el consecuente es el sentimiento en este caso). Ello requiere de un filtrado de las reglas a sólamente aquellas de calidad, para lo cual se ha hecho uso del *LIFT* como medida de calidad indicadora de la validez de las reglas: Se buscarán reglas con *LIFT* mayor que 1.2.

De forma previa, en la figura 12 se muestran el número de reglas generadas en las distintas etapas del aprendizaje realizado por cada uno de los algoritmos mediante la aplicación de un filtrado de reglas de asociación de acuerdo a alguna de las medidas de calidad definidas. Cabe destacar que posteriormente se indicarán los parámetros utilizados en cada algoritmo para obtener tal resultado. Se aprecia disparidad entre los algoritmos.

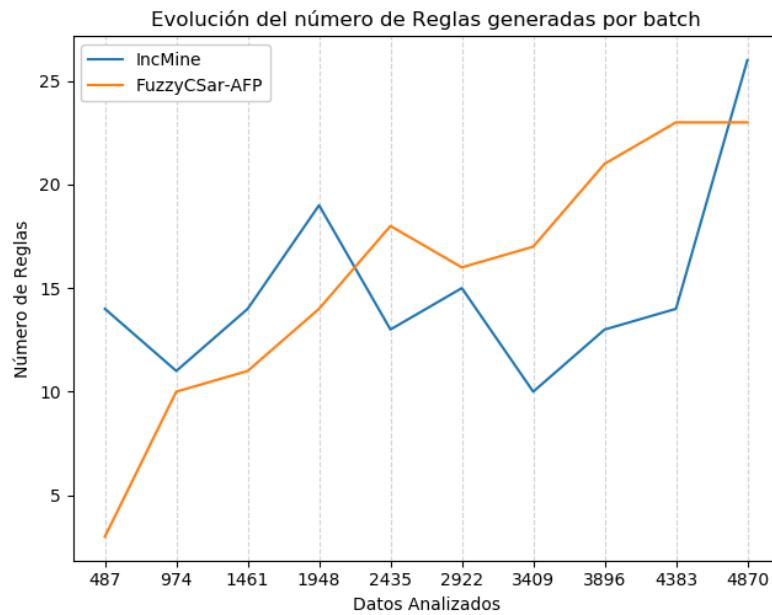


Figura 12. Evolución del número de reglas de asociación adecuadas generadas por cada algoritmo para el problema de USA.

A continuación, se pasa a analizar los resultados obtenidos mediante cada una de las dos técnicas presentadas, para lo cual se recurrirá a una visualización del tipo *Sankey*, el cual permite resumir la presencia de relaciones entre pares término-sentimiento mediante la conexión de estos mediante flujos o enlaces proporcionales al número de reglas que confirman dicho vínculo.

Extracción dinámica de reglas de asociación con *Fuzzy-CSarAFP*

La primera opción se centra en el algoritmo *Fuzzy-CSarAFP*, el cual permite realizar una extracción directa de las reglas de asociación mediante un algoritmo genético y difuso. Así, presenta una población de reglas candidatas con sus respectivas medidas de confianza en base a los datos entrantes, de manera que realiza un aprendizaje *online* conforme son recibidos los componentes del flujo de datos. Así pues, esta se presenta mucho más sencilla y de mayores garantías, con un funcionamiento muy superior al de otros algoritmos, tanto en aprendizaje estático como en el entorno de minería de flujo de datos, tal y como se destaca en [24].

Para la aplicación de algoritmo sobre el *dataset* de estudio ha sido necesario indicar las variables que ejercen como antecedentes (que en este caso serían la parte del día y las variables de los términos de la bolsa de interés) y el consecuente (que se corresponde con el sentimiento del mensaje). Además, se ha definido que la escritura de las reglas obtenidas se

realice cada 487 elementos (lo que se corresponde con el 10 % de los datos). Por su parte, el parámetro θ_{mna} , el cual permite definir el número mínimo de *matchings* que debe existir entre un nuevo dato y el conjunto de reglas existentes (pues, tal y como se explica en la descripción teórica, sino el algoritmo generará tantas como sea necesaria para cumplir dicho mínimo), de manera que se ha definido como $\theta_{mna} = 3$. Por otro lado, los parámetros de mutación p_C y p_M se han definido como 0.4, mientras que p_S es nulo, dado su definición (en este caso no es posible realizar cambios en el consecuente).

Tras ello, se ha realizado el pertinente análisis de las reglas generadas, buscando en ellas la presencia en ellas de vínculos entre términos de interés como antecedente y sentimiento como consecuente. En las figuras 13 y 14 se muestran las dichas relaciones mediante el uso de una visualización del tipo *Sankey* en base a las reglas generadas con el algoritmo *Fuzzy-CSarAFP*. Cada una de ellas se corresponde con distintos instantes del entrenamiento *online*, de modo que equivalen a las reglas generadas tras analizar una cierta cantidad de datos del total del flujo. Se aprecia que, a mayor cantidad de datos estudiados, aparecen más relaciones término-sentimiento. Destaca la figura 14, en la cual se presentan las reglas finales, las cuales pueden verse en las tablas disponibles en el **Apéndice D**.

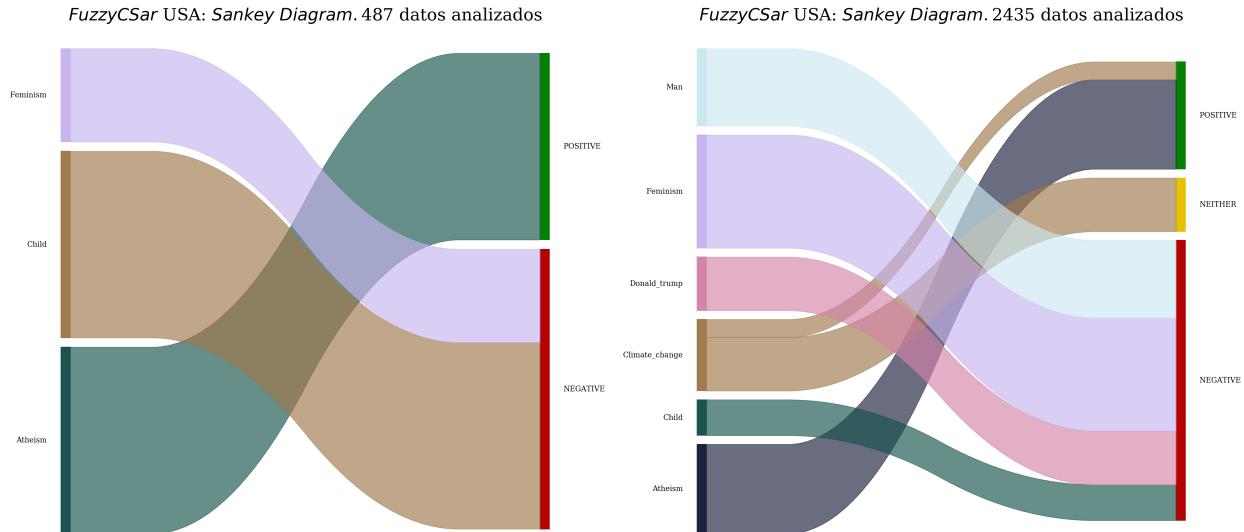


Figura 13. Diagrama *Sankey* para el problema de USA. Conjunto de vínculos término-sentimiento obtenidos por *Fuzzy-CSarAFP* hasta los *batchs* 1 y 5.

FuzzyCSar USA: Sankey Diagram. 4870 datos analizados



Figura 14. Diagrama *Sankey* para el problema de USA. Conjunto de vínculos término-sentimiento obtenidos por *Fuzzy-CSarAFP* tras analizar todos los datos.

A modo de ejemplo, se realizará un análisis del diagrama presente en la figura 14, donde se aprecia como se presentan los vínculos entre los términos de interés del antecedente y el consecuente de sentimiento, cuya validez queda recogida por la amplitud el enlace existente entre dichos elementos, la cual se corresponde con la suma de los pesos del término en las reglas que confirman dicho enlace. Así, si se compara con la tabla 13 presente en el **Apéndice D**, se comprueba como dichos enlaces son proporcionales al número de reglas que confirman dicho vínculo entre término y sentimiento.

Se aprecia como la mayoría de relaciones entre términos existentes en las reglas van referidas a un sentimiento en concreto, a excepción del tema *CLIMATE_CHANGE*, que presenta una cierta cantidad de reglas con mensajes neutros. Prestando atención a este término, se puede apreciar una gran proporción de reglas en las que se asocia dicha termino con sentimientos neutros y positivos, posiblemente debido a disconformidades de los usuarios con dicho tema.

Además, como se puede ver en la primera imagen de la figura 13, *Fuzzy-CSarAFP* presenta un estado inicial muy deficiente en lo que a reglas de asociación se refiere. Se confirma que el algoritmo necesita muchos más datos para aplicar correctamente su entorno de minería de flujo de datos, lo cual es comprensible, pues el *dataset* estudiado no da mucho juego. Es normal que con tan pocos datos el algoritmo fallen, pues no tiene información suficiente para hacer el aprendizaje correctamente. Poco a poco va produciendo una mejora del aprendizaje, incrementando las reglas generadas a medida que se analizan datos, tal y como se puede ver en la segunda imagen de la figura 13. Esto continúa hasta llegar a la situación de la figura 14, confirmándose la tendencia creciente.

En cuanto al análisis del comportamiento dinámico del sistema, se aprecia como el entorno *Fuzzy-CSar* va evolucionando y mejorando el conjunto de reglas, dando pie a más relaciones entre términos y consecuente. Todo esto puede verse mucho mejor en el video correspondiente en la carpeta compartida indicada al final de la sección. Se aprecian comportamientos similares en el tiempo, de modo que la llegada de nuevos datos permite reforzar las poblaciones de reglas en las que el término de interés pasa de un asociado sentimiento a otro distinto. Así, relaciones como la de *CHILD* y *FEMINISM* con un sentimiento negativo se mantienen en el tiempo. Feminismo y aborto se presentan como temas que crean mucha controversia por su situación actual en la sociedad, de modo que ello explica el motivo de esta asociación con opiniones negativas a lo largo del tiempo.

No se aprecian cambios drásticos en la dinámica del sistema, de modo que no hay cambios de concepto en los que se pase de un vínculo positivo a uno negativo. Sin embargo, el término cambio *CLIMATE_CHANGE* anteriormente mencionado sí que presenta una ramificación en el tiempo, que hace que pase de un vínculo con el sentimiento neutro a también presentar una población de reglas del tipo *CLIMATE_CHANGE* → *POSITIVE*.

Extracción dinámica de reglas de asociación con *IncMine*

En cuanto a la segunda opción, se basa en la utilización del *software MOA* y su extensión *IncMine*, la cual contiene el algoritmo con igual nombre que se introdujo en el marco teórico y que realizará la extracción *itemsets* frecuentes (FCIs). Para este problema se hace necesario definir algunos parámetros comunes como el soporte mínimo (necesario siempre) pero también otros característicos del algoritmo, como el ratio de relajación, introducido en el marco teórico. Así pues, se ha tomado un soporte umbral de 0.05 (5 %) y un *relaxation rate* de 0.3. El parámetro *windowSize* permite definir el número de puntos en los que realizar la escritura de soportes, de modo que se ha definido como 10, para así tener la escritura en 10 instantes y que coincida con los puntos definidos en el algoritmo *Fuzzy-CSarAFP*. También se ha especificado el tamaño de la ventana deslizante, la cual se ha definido como 90 datos.

En la extracción de las reglas de asociación de forma *offline* se estudiarán todas las reglas contenidas en los FCIs de cada *batch* de datos. y, mediante el análisis de su valor del *LIFT*, se determinará las reglas que son más interesantes. Sobre el conjunto resultante se buscarán indicios de relaciones entre términos de interés y sentimientos. De nuevo se muestran tres visualizaciones del tipo *Sankey* en el que se resumen los patrones de vínculos en las reglas de asociación obtenidas mediante el algoritmo *IncMine*, las cuales aparecen en las figuras 4.1.3 y 15. Cada una de las imágenes se corresponde con un punto del aprendizaje *offline* sobre distintos *batchs* de datos. Destaca la figura 4.1.3, en la cual se presentan las reglas de asociación finales, las cuales han sido obtenidas tras analizar la totalidad de los datos. Estas reglas se pueden encontrar en la tabla 12, disponible en el **Apéndice D**.

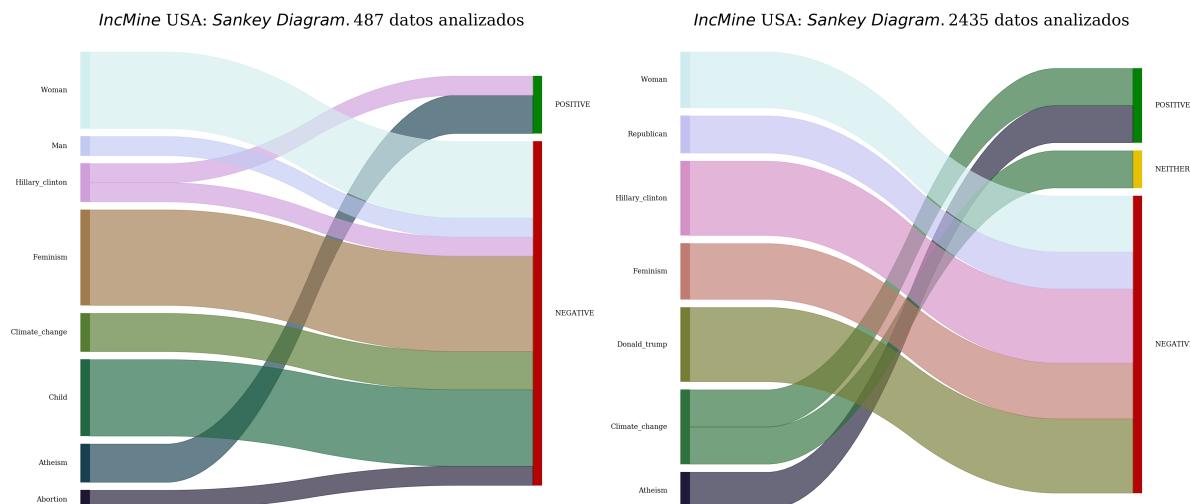


Figura 15. Diagrama *Sankey* para el problema de USA. Conjunto de vínculos término-sentimiento obtenidos por *IncMine-AFP* en los *batch* 1 y 5.

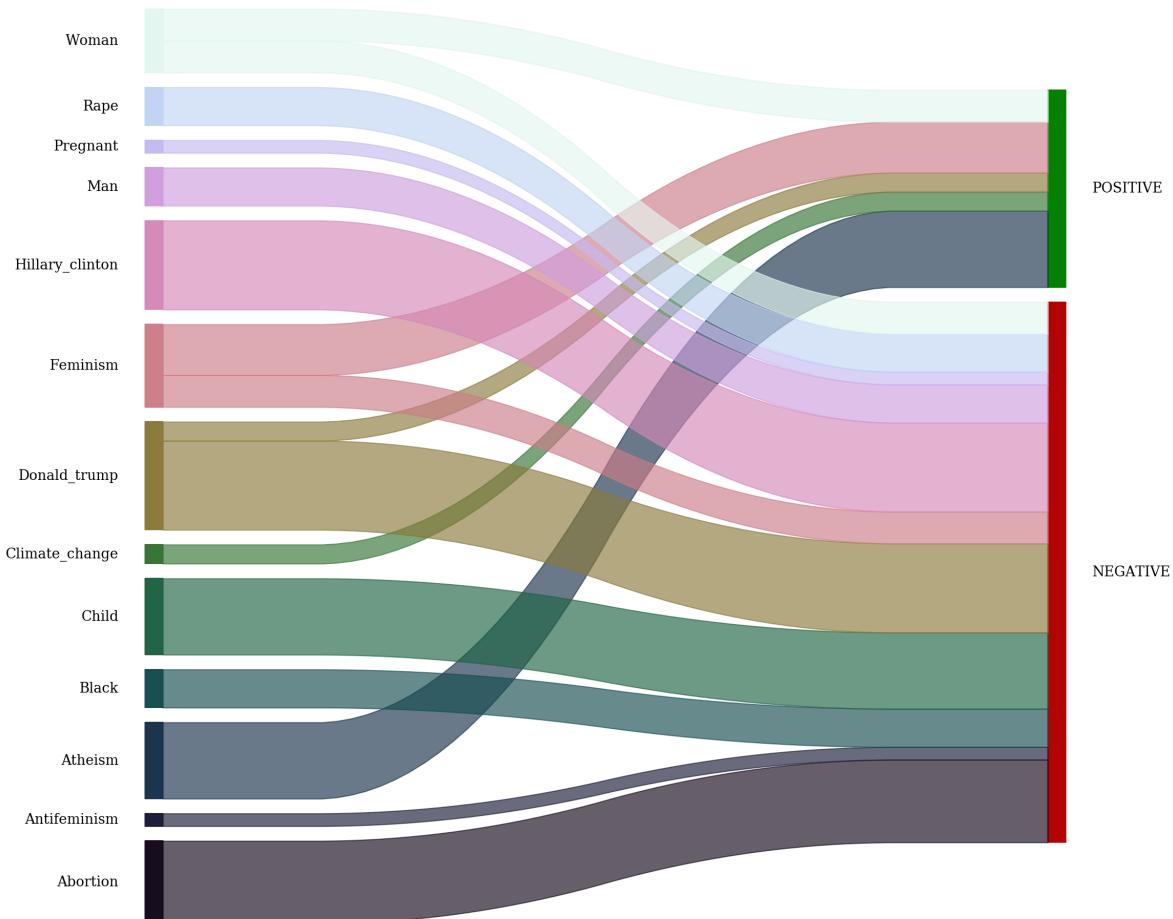
IncMine USA: Sankey Diagram. 4870 datos analizados

Figura 16. Diagrama *Sankey* para el problema de USA. Conjunto de vínculos término-sentimiento obtenidos por *IncMine-AFP* en el último *batch*.

En cuanto al análisis del diagrama presente en la figura , de nuevo se aprecian los vínculos entre los términos de interés y el consecuente de sentimiento, cuya aparición en el conjunto de reglas queda recogida por la amplitud el enlace existente entre dichos elementos. Se aprecia que los vínculos creados en base a las reglas obtenidas se dirigen a un sentimiento en concreto y casi nunca neutro. Esto encaja con el concepto en el que se enmarca Twitter, una red social utilizada por los usuarios para reivindicar sus opiniones y donde suele existir una tendencia de dominio por aquellos usuarios o actitudes extremas (Lo cual se multiplica en temas candentes como la política).

Se aprecia una gran variedad de relaciones entre términos, de modo que hay muchos términos asociados a sentimientos y con pesos similares. Quizá el más destacado es el término *DONALD_TRUMP*, quien presenta una fuerte tendencia a mensajes negativos en esta etapa final. Sin embargo, también presenta un cierto vínculo con mensajes neutros. Otros ejemplos de reglas con gran cantidad de apariciones son aquellas que contienen relaciones del tipo *HILLARY_CLINTON* → NEGATIVE. Por tanto, en esta etapa los candidatos han recibido muchos mensajes negativos, que dan lugar a poblaciones de reglas de este tipo. Las tabla presentes en el **Apéndice D**, permiten comprobar lo resumido en el gráfico, con varias reglas que relacionan dicho término y el correspondiente sentimiento.

En cuanto al comportamiento global del sistema en este entorno de minería de flujo de datos, es posible realizar un estudio del mismo a partir de la evolución dinámica del sistema (Véase video disponible en la carpeta compartida indicada al final de la sección). En este se puede apreciar que existe una evolución de las reglas generadas de modo que, aunque la mayoría parecen mantenerse, hay algunos cambios. A modo de ejemplo, de nuevo se cita el **cambio de concepto** presente en los dos candidatos: Como se comentaba anteriormente, ambos presentan vínculos finales con sentimientos negativos. Sin embargo, en algunas etapas presentan relaciones con sentimientos positivos, de manera que en estas etapas el flujo de datos mostraban mensajes con sentimientos de carácter positivo y que han dado lugar a reglas de dicha índole.

Otro caso a estudiar es el del término *CLIMATE_CHANGE*, de modo que inicialmente existe una población de reglas que respalda la relación *CLIMATE_CHANGE* → NEGATIVE pero que pronto evoluciona a las otras relaciones posibles, con sentimientos positivo y neutro. Esto se mantiene durante unas cuantas etapas más, hasta que posteriormente vuelve a una relación negativa. Sin embargo, finalmente vuelve a cambiar hasta volver a presentar alguna regla con sentimiento positivo.

Enlace a video con comportamiento dinámico del algoritmo

Cabe destacar que se ha generado un gráfico dinámico con la evolución de las reglas de asociación encontradas por los distintos algoritmos durante los distintos instantes del aprendizaje, de manera que se hace posible apreciar cómo se actualizan estas reglas a lo largo del tiempo. Dichas visualizaciones dinámicas se encuentran disponibles, tanto para *IncMine* como para *Fuzzy-CSarAFP* en la siguiente carpeta compartida:

<https://drive.google.com/open?id=1hMd-XRMQhGeCxWdPU7jhorKHJuNJ-L6t>

4.1.4. Estudio de las frecuencias de aparición en reglas de interés

Otro de los posibles ejercicios a realizar para comprobar la calidad de las reglas generadas por los algoritmos consiste en realizar un análisis de las frecuencias de aparición en reglas interesantes. De tal modo, se ha elegido alguna de las reglas simples (es decir, formadas por un sólo antecedente y un sólo consecuente) encontradas para cada método, analizando la evolución de las frecuencias de los términos tanto por separado como en conjunto para cada uno de los distintos *batch* de datos definidos.

Es posible estudiar la evoluciones temporales de los términos de alguna de reglas, como, por ejemplo, la regla $\text{ABORTION} \rightarrow \text{CHILD}$. En la figura 17 se muestra dicha evolución. En ella aparecen instantes de tiempo (a partir del tercio de los datos analizados) donde la aparición del término *ABORTION* se ve notablemente reducida sin afectar a la frecuencia de la regla $\text{ABORTION} \rightarrow \text{CHILD}$, que prácticamente se mantiene. Se deduce que existe una tendencia a que los términos *ABORTION* y *CHILD* aparezcan juntos. Así, a pesar de que la aparición individual del término *ABORTION* se reduce, la frecuencia de aparición de la regla $\text{ABORTION} \rightarrow \text{CHILD}$ no se ve reducida, debido a que las apariciones de los términos *ABORTION* y *CHILD* tienden a estar relacionadas, por lo que la frecuencia de aparición de la regla no se ve reducida. Este comportamiento se aprecia especialmente en el octavo *batch*.

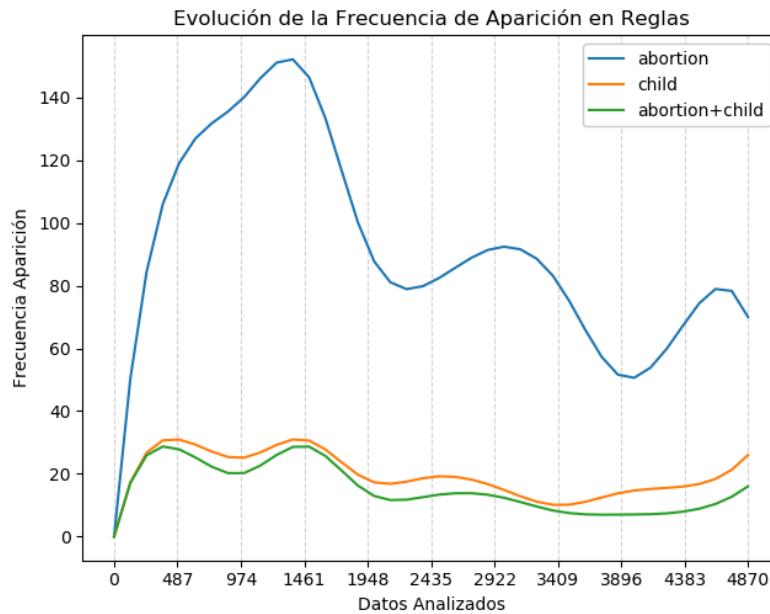


Figura 17. Evolución de la frecuencia de aparición de los términos de la regla de *IncMine* $\text{ABORTION} \rightarrow \text{CHILD}$, por separado y en conjunto, en el problema de USA.

4.2. Aplicación en el Proceso de Investidura de 2019 en España

4.2.1. Introducción al conjunto de datos. Análisis exploratorio

Tras la realización del estudio inicial con el *dataset* que anteriormente se presentaba, se ha decidido aplicar el sistema creado sobre una serie de datos reales, extraídos de Twitter mediante la utilización de su API. Así pues, se han recogido los *tweets* relativos a un tema tan de moda en redes sociales como es la política. En concreto, se ha decidido extraer los mensajes relacionados con el Proceso de Investidura de 2019 que hayan sido publicados entre el 15 de Julio de 2019 y el 29 de Agosto de 2019.

En total, se han extraído **261080 tweets**. Para ello, se ha realizado una tarea de *scraping* utilizando una serie de *hashtags* de referencia para las búsquedas. Los *hashtags* utilizados han sido: #PSOE, #PP, #UnidasPodemos, #CiudadanosCs, #VOX, #InvestiduraCongreso19 y #InvestiDudaARV.

La herramienta preparada por Twitter de extracción de *tweets* presenta numerosas opciones, tal y como se detallan en el **Apéndice B**. Aprovechando esto, se han decidido obtener varios de los atributos de los metadatos para describir cada mensaje. A continuación, se realizará una descripción de los distintos atributos asociados a los distintos *tweets* que aparecen en dicho conjunto de datos:

- ***Created_at***: Fecha de creación del *tweet*.
- ***ID***: Número identificativo del *tweet*.
- ***Text***: Mensaje contenido en el *tweet*.
- ***User_ID***: Número identificativo del usuario que publica el *tweet*.
- ***User_Name***: Nombre del usuario que publica el *tweet*.
- ***Entities_Hashtags***: Contiene los distintos *hashtags* presentes en el *tweet*.
- ***RT***: Indica si el mensaje publicado es un *retweet* o no.
- ***RT_Count***: Recuento del número de *retweets* del *tweet*.
- ***Favorite_Count***: Recuento del número de *favs* del *tweet*.

Además, se han realizado ciertas transformaciones para obtener otras variables que puedan resultar útiles para la extracción de conocimiento. Estas variables han sido generadas a partir de la fecha de publicación y son:

- **Weekday:** Indica si el mensaje se publicó entre semana (*WEEKDAY*) o en fin de semana (*WEEKEND*).
- **PartDay:** Parte del día en la que se publicó el mensaje. Existen tres posibles valores:
 1. *MAÑANA*: El mensaje se ha publicado entre las 7.00 y las 14.00
 2. *TARDE*: El mensaje se ha publicado entre las 14.00 y las 21.00
 3. *NOCHE*: El mensaje se ha publicado entre las 21.00 y las 7.00

Además, el sentimiento presente en los mensajes se presenta como uno de los atributos de mayor interés, puesto que se pretende extraer reglas con dicha variable como consecuente. Con este objetivo, se ha aplicado el clasificador *SentiStrength* anteriormente introducido, el cual permitirá extraer el sentimiento presente en el texto y lo agrupará en el atributo ***Sentiment***. Asimismo, se ha podido comprobar como alguno de los *hashtags* utilizados daba lugar a mensajes con opiniones políticas en otros idiomas. Ante esto, se ha decidido aplicar un analizador del idioma, de modo que si el mensaje está, con una probabilidad lo suficientemente alta, en un idioma distinto al español, lo clasifica como extranjero. Esto se recoge en la variable ***Language***, la cual permitirá realizar un filtrado posterior.

La aplicación del analizador del idioma es totalmente positiva, permitiendo reducir el total de mensajes a **250152**, de modo que más de **10000 tweets** han sido descartados por tener un idioma distinto al español. Asimismo, nuevamente se han analizado las distribuciones de algunos atributos de interés. Es el caso de atributos como la parte del día o de la semana en la que se realiza la publicación del mensaje, las cuales se recogen en la figura 18:

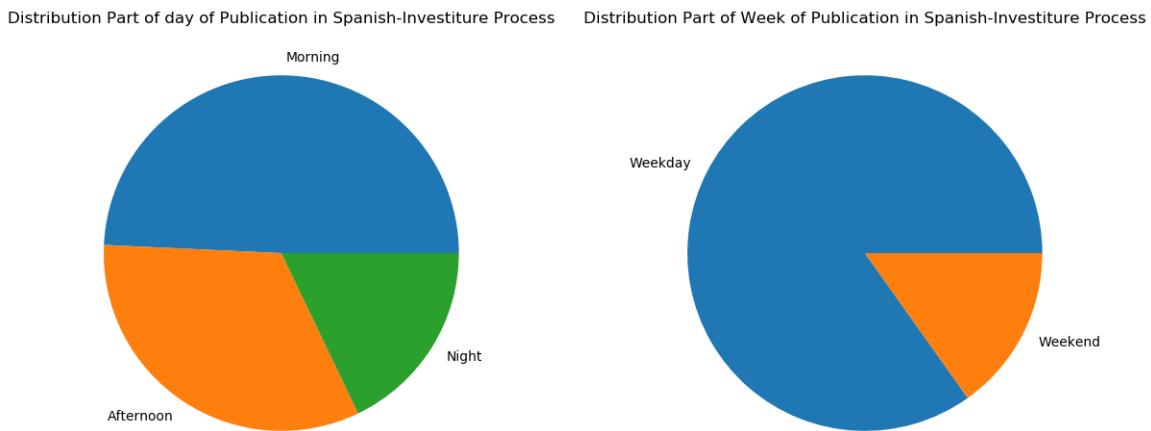


Figura 18. Gráficos con las distribuciones de la parte del día(izquierda) y parte de la semana(derecha) en las que se realizó la publicación de los *tweets* a estudiar.

Se aprecia lo esperado: La mayoría de mensajes se publican por la mañana, que es la parte del día en la que normalmente se realizan los debates y se publican las noticias de interés en los medios de comunicación y, en su gran mayoría, durante la semana, de modo que la gente utiliza los fines de semana para desconectar. Sin embargo, la variable cuya distribución presenta *a priori* mayor interés es el sentimiento presente en los mensajes. Esta se recoge en la figura 19, donde aprecia que existe un reparto equitativo entre el sentimiento de los mensajes, aunque quizá existe una cierta predominancia de mensajes neutros (lo cual quizá se deba a una abundancia de mensajes de carácter informativo, con sentimiento neutro):

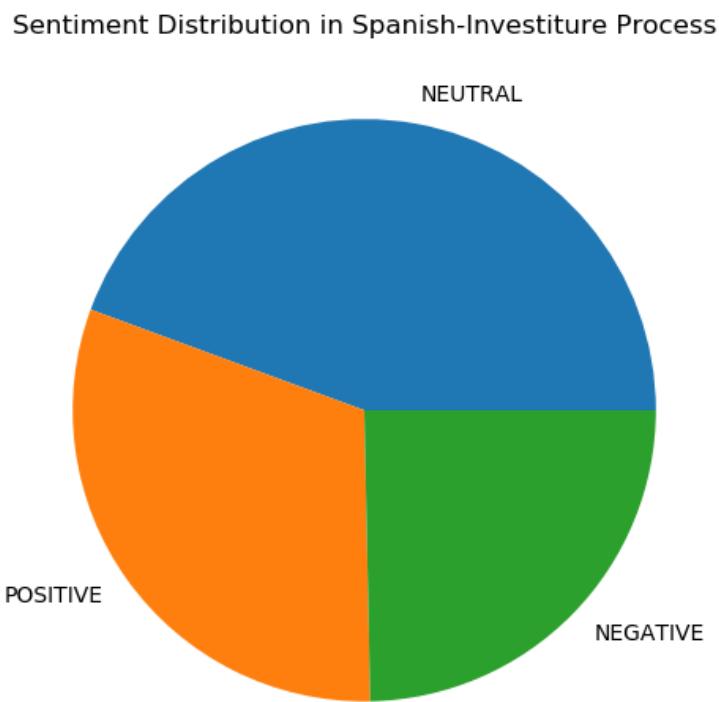


Figura 19. Gráficos con la distribución del sentimiento presente en los *tweets* a estudiar.

Además, se ha analizado la influencia de los grupos políticos en base a las menciones a los propios partidos y sus integrantes en los *tweets* publicados en la línea temporal. Estos personajes han sido elegidos en base a su relevancia en el histórico de mensajes. Así, mientras que para el PSOE se han considerado personajes como Pedro Sánchez(líder), Carmen Calvo y Adriana Lastra, en el caso de PP aparecen personajes como Pablo Casado(líder) o Isabel Díaz Ayuso. En el caso de Ciudadanos se han destacado a Inés Arrimadas y Albert Rivera(líder), mientras que para VOX se han tenido en cuenta a Santiago Abascal(líder) y Ortega Smith. Por último, en Unidas Podemos se ha considerado a Pablo Iglesias(líder), Irene Montero y Pablo Echenique. De tal modo, la evolución final queda recogida en la figura 20.

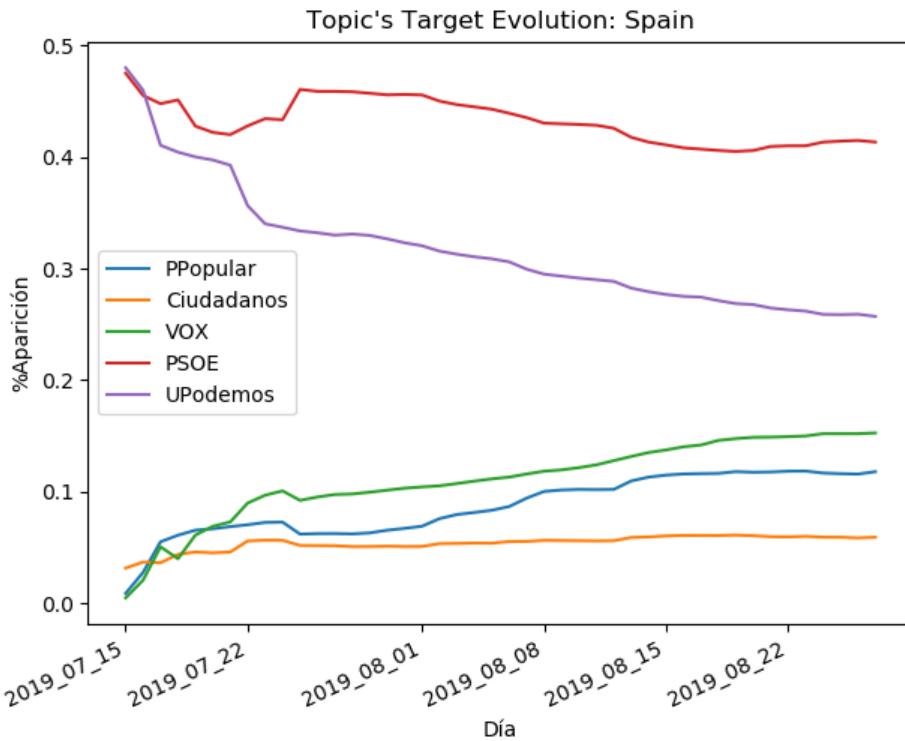


Figura 20. Gráfico con la evolución temporal de la frecuencia de publicación sobre los temas de interés presentes en los *tweets* del problema de España.

Se puede apreciar como partidos como PSOE y Unidas Podemos presentan inicialmente una relevancia prácticamente total, a raíz de las discusiones por el proceso de investidura, mientras que el resto de partidos no eran objeto de debate. Sin embargo, a medida que avanzan los días se aprecia cómo las menciones a los partidos se van equilibrando, a excepción del caso del PSOE, el cual siguió siendo el centro de debate por diversos motivos.

4.2.2. Procesamiento del Lenguaje Natural

Se ha llevado a cabo la aplicación de un análisis de texto muy similar al comentado para el *dataset* de las elecciones de los Estados Unidos. Así pues, de nuevo se han eliminado contenidos como *links*, números, puntuaciones (a excepción de los # y @ que hacen referencia a *hashtags* y usuarios, respectivamente), emoticonos y risas de los mensajes, así como posibles repeticiones de letras consecutivas. También se ha aplicado las técnicas de minería de texto comunes, comenzando por la supresión de *stopwords* y palabras cortas (de longitud menor a 3) y realizando el etiquetado gramatical (volviendo a considerar que *hashtags* y usuarios tienen sus propias etiquetas gramaticales, denominadas como *HASHTAG* y *USER*, respectivamente) y la técnica de lematización para estandarizar el texto.

De nuevo, se ha utilizado **SpaCy** [109] como herramienta para el análisis de texto pues, aunque su velocidad de análisis no es la mejor, presenta una mejor librería para el análisis de textos en español. Además, se ha vuelto a realizar una simplificación del texto según la categoría gramatical de las palabras. Se ha seguido una estrategia similar a la aplicada en la base de datos de USA, con la salvedad de que esta vez sí se han considerado verbos, pues se ha comprobado como el etiquetador de **SpaCy** presenta pequeños fallos y en algunas ocasiones identifica palabras que son sustantivos como verbos.

Tras ello, se ha vuelto a realizar un análisis de *N-gramas*, buscando bigramas, trigramas e incluso cuatrigramas en el texto. Tras analizar los cuatrigramas obtenidos, el único término de interés es `partido_socialista_obra_español`, el cual hace referencia al partido político **PSOE**. Por su parte, en el caso de los trigramas se han encontrado un total de 6 que pueden resultar interesantes. De ellos, cinco hacen referencias a personalidades políticas (`isabel_diaz_ayuso`, `cayetana_alvarez_toledo`, `pedro_sanchez_castejon`, `javier_ortega_smith`, `miguel_angel_blanco`) mientras que, por su parte, los trigramas de `ley_patrimonio_natural` y `ley_violencia_genero` hacen referencia a leyes. En concreto, el primero habla de una ley que defiende la conservación del patrimonio natural y de la biodiversidad de los ecosistemas (lo cual se deberá probablemente a opiniones relativas al incendio de Gran Canaria) mientras que el segundo hace referencia a una ley de 2004 en favor de los derecho de las maltratadas (probablemente citada ante las noticias diarias de violaciones o ante las reivindicaciones realizadas por VOX en temas de feminismo e igualdad).

Por su parte, en el caso de los bigramas se han encontrado numerosos términos de interés. Como se puede comprobar, la mayoría de bigramas hacen referencia a personalidades políticas o comunidades autónomas. También aparecen grupos o instituciones públicas (como `guardia_civil`, `sanidad_publica`, `seguridad_social` o `audiencia_nacional`) y programas o medidas (como `politica_social`, `servicio_publico` o `ley_electoral`), además de conceptos interesantes de la política (como `preso_politico`, `violencia_genero`, `libertad_expcion`, `preso_politico` o `agresion_sexual`). Los bigramas encontrados quedan recogidos en la tabla 5:

(pedro, sanchez)	(pablo, iglesias)	(unidas, podemos)	(carmen, calvo)	(violencia, genero)
(mayoria, absoluto)	(adriana, lastra)	(albert, rivera)	(partir, politico)	(susana, diaz)
(pablo, casado)	(mucion, censura)	(open, arms)	(santiago, abascal)	(guardia, civil)
(comunidad, madrid)	(reformar, laboral)	(irene, montero)	(repetir, eleccion)	(castilla, leon)
(diaz, ayuso)	(consejo, ministro)	(aitor, esteban)	(felipe, gonzalez)	(ivan, redondo)
(señor, sanchez)	(repeticion, electoral)	(campaña, electoral)	(politico, social)	(ley, mordaza)
(gabriel, rufian)	(congreso, diputados)	(preso, politico)	(partido, popular)	(grupo, parlamentario)
(antonio, martinez)	(rocio, monasterio)	(esperanza, aguirre)	(señor, iglesias)	(violencia, machista)
(ana, oramas)	(repeticion, eleccion)	(ortega, smith)	(alvarez, toledo)	(reforma, laboral)
(seguridad, social)	(pais, vasco)	(ortega, lara)	(ana, botella)	(guerra, civil)
(isabel, #diazayuso)	(señor, rivera)	(servicio, publico)	(señora, calvo)	(mariano, rajoy)
(javier, maroto)	(junta, andalucia)	(audiencia, nacional)	(alberto, garzon)	(lopez, miras)
(raquel, romero)	(sanchez, castejon)	(iñigo, errejon)	(region, murcia)	(libertad, expresion)
(santi, abascal)	(cristina, cifuentes)	(partido, socialista)	(ines, arrimadas)	(pablo, montesinos)
(pablo, echenique)	(julio, anguita)	(sra, calvo)	(maria, claver)	(agresion, sexual)
(sanidad, publico)	(laura, borras)	(ley, electoral)	(noelia, vera)	(ana, beltran)
(señor, pedro)	(coalicion, canaria)	(diaz, #ayuso)	(policia, nacional)	(yolanda, diaz)

Cuadro 5. Tabla con los distintos bigramas encontrados para el caso de España y su definición.

Sin embargo, de nuevo ha sido necesario realizar una unificación de entidades, pues existe una gran variedad de términos para hacer referencia a personalidades o grupos políticos y entidades de interés. En la tabla 6 se muestran las entidades consideradas junto con los términos que pasan a englobarse bajo cada una de ellas:

Entidad	Términos asociados
PSOE	partido_socialista_obra..._español, psoe, #psoe, #partidosocialistaobreroespañol, @psoe, partido_socialista
pedro_sanchez	sanchez_castejon, @sanchezcastejon, #sanchez, #pedrosanchez, pedro_sanchez_castejon, #pedrosanchezcastejon, #sanchezcastejon, pedro_sanchez, señor_pedro, #pedrosanchezenlaser, #sanchezsi, #siapedro, #pedronoseatreve, #pedronoquiere, #sanchezdimision
carmen_calvo	carmen_calvo, señor_calvo, sra_calvo, #carmencalvo, @carmencalvo, @carmencalvo_voz, @voxes,@vox, #vox, #sentidocomunvox, #voxutil, #voxverdaderaoposicion, #yovolvereavotarvox, #voxextremanece...idad
VOX	adriana_lastra, adriana_lastra, #adriana_lastra, @adrilastra, lastra
adriana_lastra	santiago_abascal, @santiabascal, @santi_abascal, santi_abascal, abascal, #abascal, señor_abascal, #santiabascal
santiago_abascal	#abascal, señor_abascal, #santiabascal
ortega_smith	ortega_smith, javier_ortega_smith, @ortegasmith, @ortega_smith, #ortegasmith
unidas_podemos	unidas_podemos, podemos, #unidaspodemos, @unidas_podemos, @unidaspodemos, @ahorapodemos, #podemos, #ahorapodemos
pablo_iglesias	pablo_iglesias, señor_iglesias, @pabloiglesias, iglesias, @pablo_iglesias, #pabloiglesias, #iglesias
irene_montero	irene_montero, @irenemontero, #irenemontero, @irene_montero, #montero, irenemontero
pablo_echenique	pablo_echenique, echenique, @pnique, #pabloechenique, #echenique
iñigo_errejon	iñigo_errejon, errejon, #errejon, #iñigoerrejon, @ierrejon
alberto_garzon	alberto_garzon, #garzon, @garzon, #albertogarzon, @agarzon
ciudadanos	#ciudadanoscs, @ciudadanoscs, #cs, @ciudadanos, #ciudadanos
ines_arrimadas	ines_arrimadas, #inesarrimadas, #arrimadas, arrimadas, @inesarrimadas
albert_rivera	albert_rivera, señor_rivera, #albertrivera, albertrivera, #rivera, @albert_rivera, @albertrivera

Entidad	Términos asociados
gabriel_rufian	gabriel_rufian, señor_rufian, #rufian, rufian, @rufian, @gabrielrufian, #gabrielrufian
partido_popular	partido_popular, #partidopopular, partidopopular, @partidopopular, #pp, @ppopular, #ppopular, @populares
pablo_casado	pablo_casado, señor_casado, #casado, casado, @pablocasado, @pablocasado, #pablocasado
isabel_diaz_ayuso	diaz_ayuso, @idiazayuso, #isabeldiazayuso, #diazayuso, #ayuso, isabel_diaz_ayuso, isabeldiazayuso, @isabeldiazayuso
cayetana_alvarez_toledo	cayetana_alvarez_toledo, #cayetanaalvarez, @cayetanaat, alvarez_toledo, #cayetanaalvareztoledo
investidura	#investidurapublico, #mvtinvestidura, #investiduraarv, #debatedeinvestidura, #mvtinvestidura, investidura, #investidurave, #investidura, #investiduracongreso #investidurafallida, #debateinvestiduraespu, #investiduravej, #sesiondeinvestidura
elecciones	elecciones, #elecciones, electoral, eleccion, #eleccionesespana
derecha	derecha, #derecha, ultraderechista, ultraderecha, #ultraderecha, #trifachito, trifachito
izquierda	izquierda, izquierdo, progresista, #izquierda, #progresista, #socialista, #sociolista, #sociolista
independentismo	independentismo, independentista, #independentista, #independencia, independencia
abstencion	abstencion, #abstencion, abstenerse, abstenido, abstener, #abstener
repeticion_electoral	repeticion_electoral, repeticion_elección, repetición_elecciones, repetir_elecciones, repetir_elección, #repetitionelectoral, #repetitionelecciones, #repetitionelección dialogo, #dialogo, dialogar, negociar, #negociacion, #negociacion
negociacion	pacto, #pacto, pactar
pacto	gobierno, #gobierno, gobernar, #construirgobierno
gobierno	feminismo, #feminismo, feminista, #feminista, feminazi
feminismo	open_arms, #openarms, @openarms
open_arms	democracia, democratico, democracia
democracia	open_arms, #openarms, @openarms, inmigracion, #inmigracion, inmigrante
inmigracion	#violenciamachista, violencia_genero, agresion_sexual, #violenciadegenero, #misoginia
violencia_genero	corrupto, #corrupcion, corrupcion
corrupcion	

Cuadro 6. Tabla con los distintas entidades consideradas y los términos englobados en ellas.

Una vez hecho esto, los textos ya están lo más estandarizados posible, de modo que se procede a la creación de una bolsa de términos de interés de la base de datos, los cuales pueden ser interesantes en base al concepto al que hacen referencia y a su frecuencia de aparición. De nuevo se ha creado de forma manual, eligiendo los términos que aparecen dentro de la tabla 7, compuesta esta vez por 40 palabras.

investidura	PSOE	pedro_sanchez	pablo_iglesias	gobierno
unidas_podemos	VOX	partido_popular	izquierda	izquierda
ciudadanos	elecciones	coalicion	negociacion	pacto
albert_rivera	carmen_calvo	santiago_abascal	isabel_diaz_ayuso	democracia
pablo_casado	gabriel_rufian	abstencion	adriana_lastra	irene_montero
reforma_laboral	feminismo	violencia_genero	iñigo_errejon	corrupcion
pablo_echenique	independentismo	ortega_smith	bipartidismo	alberto_garzon
inmigracion	mayoria_absoluta	repeticion_electoral	regeneracion	ines_arrimadas

Cuadro 7. Conjunto de elementos de interés considerados para la base de datos de España.

Así, todo el contenido de los mensajes quedará reducido a aquellos términos contenidos en la bolsa de interés definida. De tal modo, aquellos mensajes que no contengan ninguno de los términos de interés serán directamente descartados. No obstante, puesto que la búsqueda se ha realizado en base a *hashtags* relacionados con estos términos de interés, en todos ellos hay al menos una palabra de la bolsa de palabras definida.

En la figura 21 es posible apreciar dichos dentro de una nube de palabras (*word cloud*):



Figura 21. *Word cloud* con las palabras elegidas para la bolsa de interés del problema de España.

Asimismo, de nuevo es posible analizar la frecuencia de aparición de dichos términos, comprobando así cuáles aparecen con una frecuencia mayor y, por tanto, gozan de mayor importancia. Esto queda recogido en la figura 22:

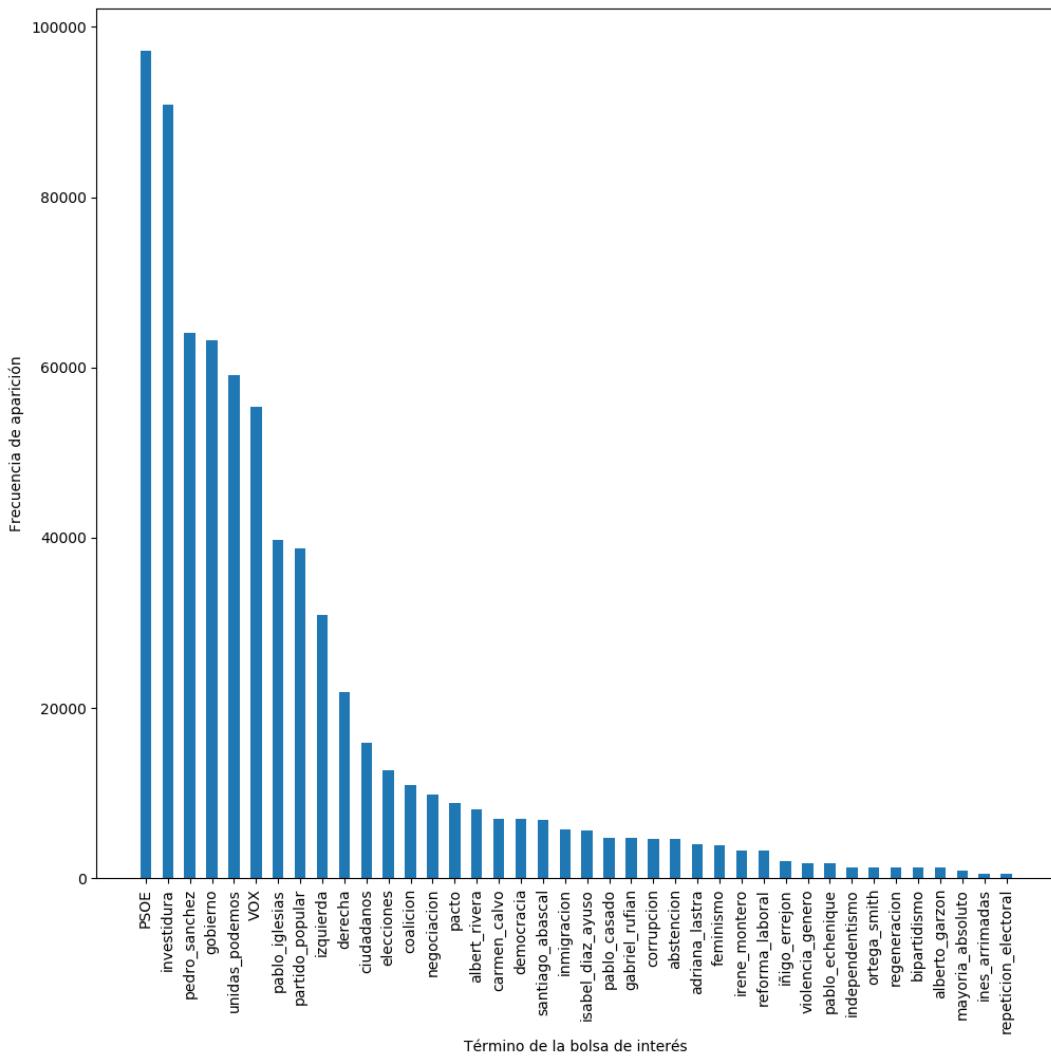


Figura 22. Histograma con la frecuencia de aparición de cada una de las palabras elegidas para la bolsa de interés en el problema de España.

4.2.3. *Association Stream Mining*: Extracción dinámica de reglas de asociación

De nuevo, una vez generada la base de datos de transacciones de términos de interés se procede a la obtención de las reglas de asociación entre dichos términos y a la extracción de la presencia vínculos entre términos y sentimientos. De igual modo que en el caso anterior, se volverá a realizar un estudio siguiendo las dos aproximaciones consideradas: Un aprendizaje incremental de flujos de datos para extraer los FCIs mediante el uso de *IncMine* [38], seguido de un aprendizaje de reglas *offline*; y una extracción directa de las reglas de asociación presentes mediante la aplicación de una técnica difusa y *online* como es *Fuzzy-CSarAFP* [24].

Tras ello, se analizarán estas reglas para buscar la presencia de relaciones entre términos y sentimientos en ellas. Nuevamente, tras obtener todas reglas se ha de realizar el estudio pertinente para buscar relaciones del tipo término de interés-sentimiento, para lo cual se volverá a realizar un recuento de las reglas generadas que presentan un par antecedente-sentimiento. Es necesario prestar atención sólamente a aquellas que sean lo suficientemente válidas, para lo cual se vuelve a emplear el *LIFT* como medida de calidad indicadora de la calidad de las reglas: Se vuelve a buscar reglas con *LIFT* mayor que 1.2.

Aunque posteriormente se indicarán los parámetros utilizados en cada algoritmo para obtener tal resultado, se ha vuelto a estudiar el número de reglas generadas en las distintas etapas del aprendizaje realizado por cada uno de los algoritmos mediante la aplicación de un filtrado de reglas de asociación comentado. Este queda recogido en la figura 4.2.3.

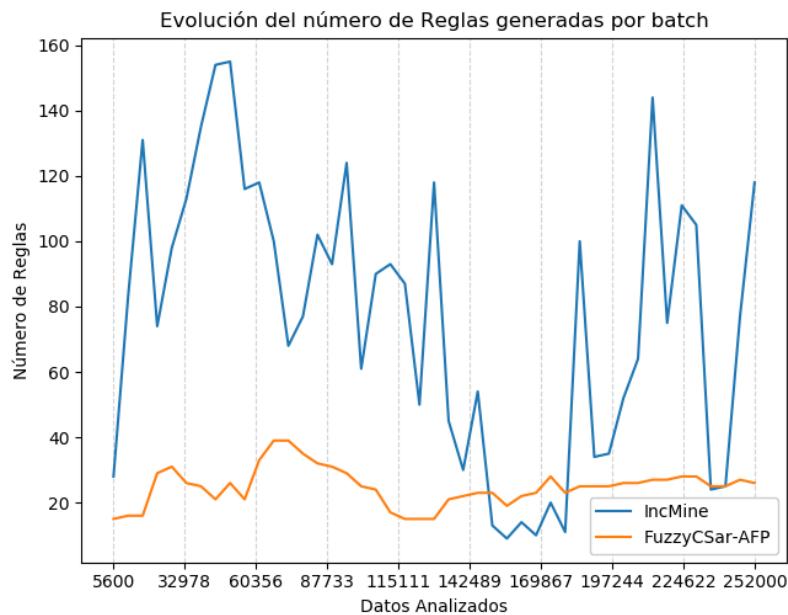


Figura 23. Evolución del número de reglas de asociación adecuadas generadas por cada algoritmo para el problema de España.

Se aprecia que en este caso la cantidad de reglas propuestas por *IncMine* es notablemente mayor. A continuación, de nuevo se procede a analizar los resultados obtenidos mediante cada una de las dos técnicas presentadas, para lo cual se vuelve a utilizar un diagrama de *Sankey* para resumir la presencia de relaciones entre pares término-sentimiento mediante la conexión de estos mediante flujos o enlaces proporcionales al número de reglas que confirman dicho vínculo.

Extracción dinámica de reglas de asociación con *Fuzzy-CSarAFP*

En este punto, se ha realizado la aplicación del aprendizaje *online* a través *Fuzzy-CSarAFP* procediendo de forma similar al caso anterior: Se ha indicado que la variable de sentimiento es la que ejerce como consecuente, mientras que el resto (parte del día, parte de la semana, indicación sobre si es *retweet* y las variables de los 40 términos de la bolsa de interés) ejercen como posible antecedentes. Asimismo, se ha definido que la escritura de las reglas obtenidas se realice cada 5600 elementos, lo que se corresponde con el promedio de *tweets* por día. Por otro lado, ante el aumento del número de variables a considerar, se ha decidido definir el parámetro $\theta_{mna} = 6$, de modo que se establece un mínimo de *matching* mayor. Por su parte, cabe destacar que en este caso se han definido los parámetros de mutación p_C y p_M como 0.1, mientras que, dada su definición, p_S se ha dejado como 0, pues en este caso sólo hay un elemento como consecuente.

De nuevo se han analizado las reglas resultantes en busca de vínculos entre los términos de interés definidos y sentimientos. En las visualizaciones presentes en las figuras 24 y 25 se muestran tres ejemplos de los diagramas de *Sankey* obtenidos con dichas relaciones en base a las reglas generadas con el entrenamiento *online* de *Fuzzy-CSarAFP*. Destaca la figura 25, en la cual se presentan las reglas finales, las cuales pueden verse en la tabla 15, disponible en el **Apéndice E**.

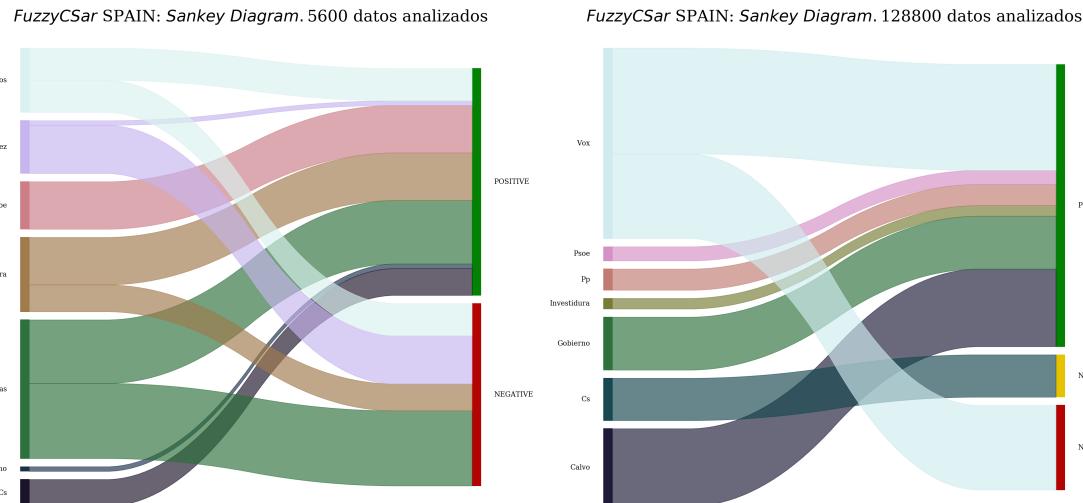


Figura 24. Diagrama *Sankey* para el problema de España. Conjunto de vínculos término-sentimiento obtenidos por *Fuzzy-CSarAFP* tras analizar el primer grupo de datos y a mitad del análisis.

FuzzyCSar SPAIN: Sankey Diagram. 250147 datos analizados

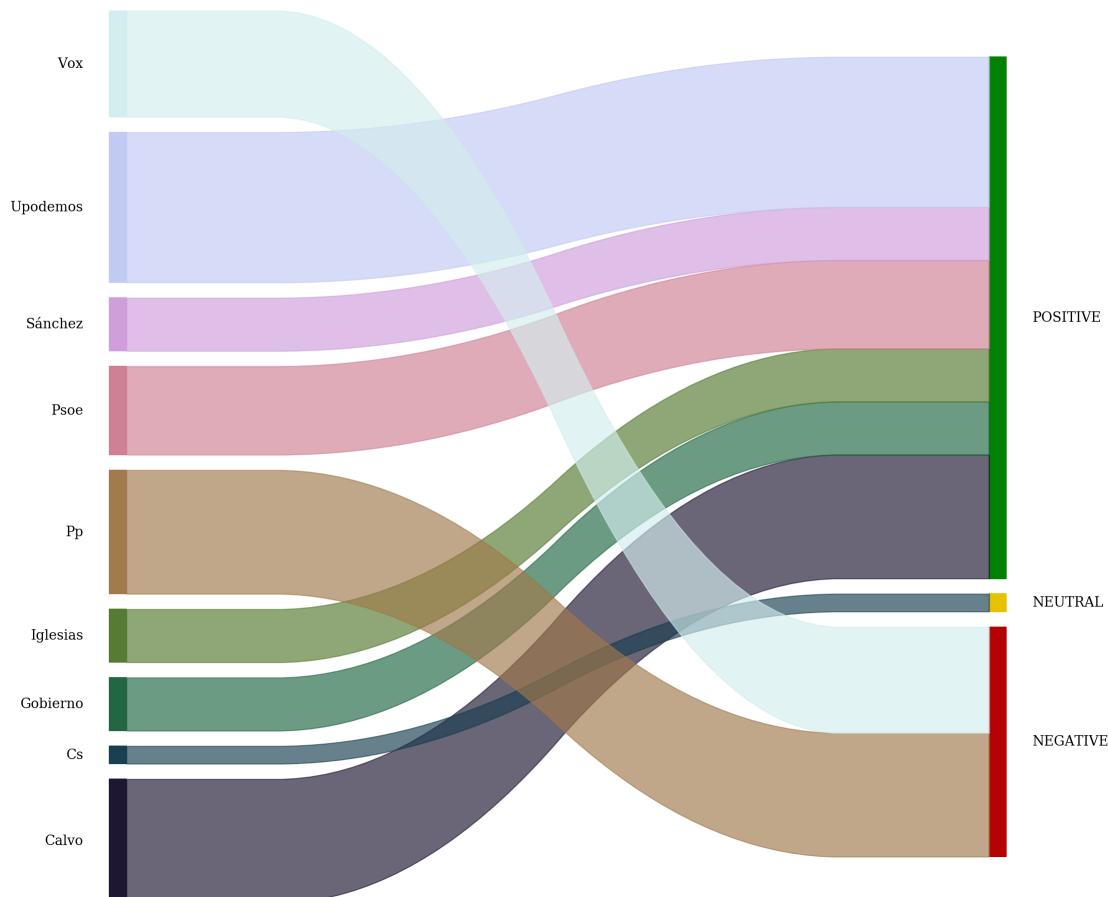


Figura 25. Diagrama *Sankey* para el problema de España. Conjunto de vínculos término-sentimiento obtenidos por *Fuzzy-CSarAFP* tras analizar todos los datos.

Se aprecia un comportamiento variable en el número de reglas y las relaciones presentes a lo largo del tiempo. Existen por tanto numerosos fenómenos de *concept drift*, como era de esperar en un conjunto de datos real como el estudiado, de modo que existen numerosas situaciones en las que se produce una llegada de datos del flujo que presentan temáticas o sentimientos totalmente diferentes a los de los instantes anteriores. Este cambio del objetivo de las conversaciones de Twitter recogidas en los mensajes estudiados da lugar a la generación de reglas de asociación en las que se resumen dichos mensajes y, por ende, en los cambios que se aprecian en el diagrama de *Sankey* correspondiente.

El análisis de la figura 25 nos muestra una situación muy interesante, en la cual todos los partidos políticos aparecen asociados a un sentimiento, dándonos un resumen del sentimiento generado por los distintos grupos políticos en la totalidad de datos. Así, mientras que PSOE, Podemos y VOX presentan una elevada cantidad de reglas con sentimiento positivo, PP obtiene un vínculo con el negativo. Por otro lado, Ciudadanos está relacionado con reglas de sentimiento neutro.

Comparando los dos gráficos presentados en la figura 24, se aprecia como los conjuntos de relaciones obtenidos no se mantienen en el tiempo, como resultado de los cambios de concepto comentados. A pesar de estas variaciones, se aprecian como las relaciones entre mensajes positivos y el PSOE, aunque el peso de este tipo de reglas se han visto reducidas. EN cambio, se aprecian cambios notable, como el de Ciudadanos, que pasa de presentar una relación con un sentimiento positivo (aunque muy débil) a una relación notable con sentimiento neutro. Se recomienda encarecidamente visualizar el video disponible en la carpeta compartida para entender mejor el comportamiento dinámico del sistema y poder ir analizando paso a paso qué cambios de concepto y en qué línea se producen.

A modo de ejemplo, el análisis en los primeros instantes del término *PP* presenta una variación que sugiere un cambio de concepto: En los primeros instantes en los que aparece está relacionado con reglas con sentimientos positivos y neutros. Sin embargo, en la cuarta etapa se aprecia un cambio drástico, en el que pasan a darse reglas del tipo *PP* → NEUTRO. No suficiente con esto, un par de etapas después muestra una ramificación en la que el vínculo con reglas de sentimiento positivo coge gran peso. Esto se mantiene unas ciertas etapas hasta que el término *PP* deja de tener peso, a consecuencia de que deja de aparecer en los mensajes y ya no se generan reglas que lo impliquen.

Otro caso interesante es el de *VOX*, quien inicialmente no aparece en las reglas pero que poco a poco va cogiendo peso en los mensajes y apareciendo en las reglas generadas. Destaca que en el punto medio presenta vínculos con los sentimientos positivo y negativo con una importancia elevadísima, pues la combinación de los pesos de ambos vínculos es prácticamente la mitad del total. Este crecimiento en el punto medio se puede apreciar en la segunda imagen de la figura 24.

Extracción dinámica de reglas de asociación con *IncMine*

De nuevo, se ha aplicado el algoritmo *IncMine* disponible en **MOA**. Aunque se ha mantenido el valor del soporte umbral de 0.05 (5 %), en este caso se ha definido un ratio de relajación distinto, con un valor de 0.3. Por su parte, se ha definido un tamaño de ventana de 45 (tantas como número de días) mientras que el tamaño de la ventana deslizante se ha fijado en 950 datos. Tras la obtención de los FCIs y la posterior extracción *offline* de reglas de asociación a partir de dichos *itemsets*, se ha vuelto a analizar la presencia de relaciones entre términos de interés y sentimientos en base a las reglas existentes en el conjunto de reglas de asociación filtrado en base al valor del *LIFT* umbral establecido.

Asimismo, se ha vuelto a hacer uso del diagrama tipo *Sankey* para mostrar una visualización que contenga toda la información acerca de vínculos encontrados. En las figuras 26 y 27 se presentan los patrones de vínculos presentes en las reglas de asociación obtenidas mediante el algoritmo *IncMine* para el problema de España. Cada una de las imágenes se corresponde con un punto del aprendizaje *offline* sobre distintos *batchs* de datos. Destaca la figura 4.1.3, en la cual se presentan las reglas de asociación finales, las cuales han sido obtenidas tras analizar la totalidad de los datos. Estas reglas se pueden encontrar en la tabla 14, disponible en el **Apéndice E**.

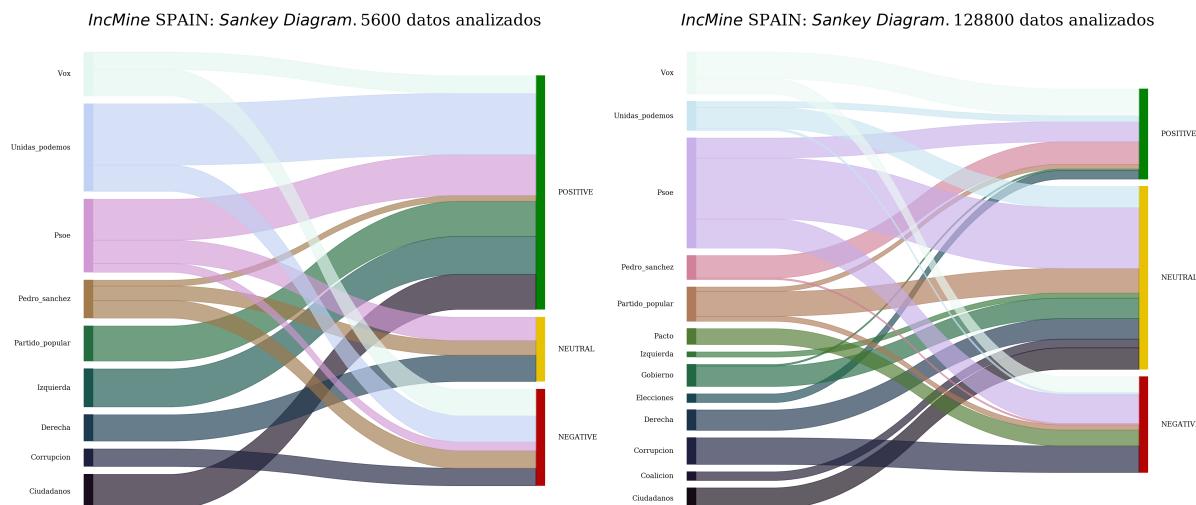


Figura 26. Diagrama *Sankey* para el problema de España. Conjunto de vínculos término-sentimiento obtenidos por *IncMine* tras analizar el primer grupo de datos y a mitad del análisis.

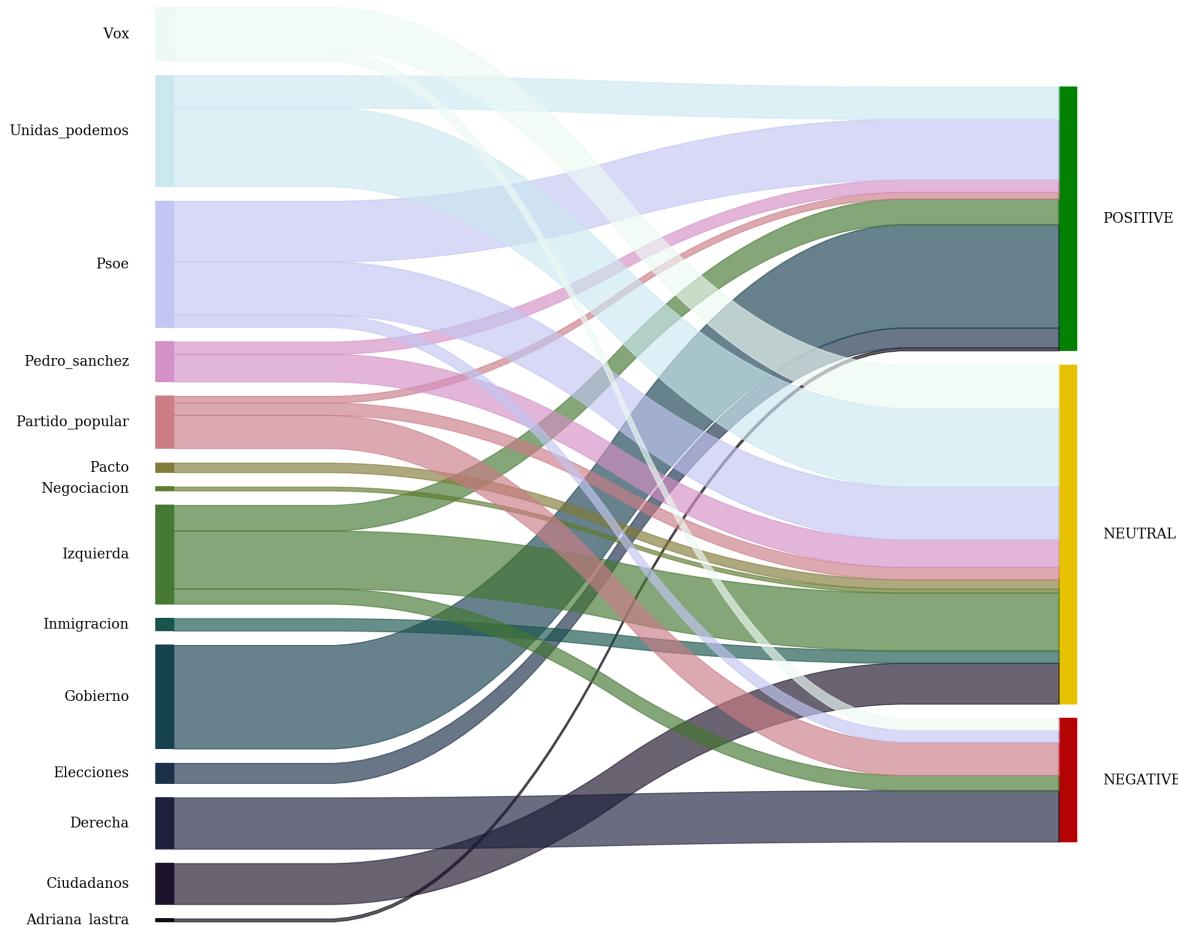
IncMine SPAIN: Sankey Diagram. 250147 datos analizados

Figura 27. Diagrama Sankey para el problema de España. Conjunto de vínculos término-sentimiento obtenidos por *IncMine* tras analizar todos los datos.

Se aprecia como la cantidad de reglas obtenidas es dispar, de modo que el número y el tipo de las relaciones entre términos van variando en el tiempo, como se puede apreciar en la figura anteriormente comentada. El análisis de la figura 27 nos muestra una situación muy similar a la obtenida para *Fuzzy-CSarAFP*, de modo que nuevamente todos los partidos políticos aparecen asociados a un sentimiento concreto. Sin embargo, los vínculos creados son diferentes a los presentados en dicho caso, aunque cabe recordar que en el caso de *IncMine* esto sólamente se refiere a los resultados en la última ventana de 950 datos definida.

El análisis de las primeras etapas y la comparación sucesiva entre ellas muestra muchas relaciones que se mantienen constantes en el tiempo, como el vínculo CORRUPCION → NEGATIVO. Sin embargo, también variaciones, de manera que se presenta nuevamente el fenómeno de *concept drift*. Un ejemplo es el caso del término *PP*, el cual aparece inicialmente vinculado a un sentimiento positivo pero que pronto cambia a una relación del tipo *PP* → NEUTRAL y que en instantes centrales de la evolución presenta también relaciones variantes entre los tres sentimientos. Así, a lo largo de la línea el tiempo presenta vinculaciones con sentimientos de toda clase, por lo que los mensajes recibidos en el flujo de datos muestran cambios en las opiniones de los usuarios acerca de dicho concepto, los cuales dan lugar a reglas que implican consecuentes distintos.

Cabe destacar que la **comparación** de ambos algoritmos es confusa. Aunque *Fuzzy-CSarAFP* no está originalmente pensado para este tipo de problemas, se esperaba que fuese capaz de permitir obtener una mayor cantidad de reglas. No ha sido competencia para *Inc-Mine*, quien en este caso real es capaz de ofrecer una población de reglas mucho mejor. No obstante, cabe la posibilidad de que sea posible realizar una adecuación más exhaustiva del código (si bien se cree que *a priori* está optimizado). En cualquier caso, ambas herramientas parecen una buena complementación, de modo que mientras que *Fuzzy-CSarAFP* permite realizar un estudio *online* de los datos a medida que estos van llegando al sistema, *IncMine* nos permite analizar los *itemsets* frecuentes por cada *batch*. De tal modo, la información obtenida por los dos métodos es igualmente válida, pudiendo ser dos complementos interesantes para un posible análisis más exhaustivo.

Enlace a video con comportamiento dinámico del algoritmo

Cabe destacar que se ha generado un gráfico dinámico con la evolución de las reglas de asociación encontradas por los distintos algoritmos durante los distintos instantes del aprendizaje, de manera que se hace posible apreciar cómo se actualizan estas reglas a lo largo del tiempo. Dichas visualizaciones dinámicas se encuentran disponibles, tanto para *IncMine* como para *Fuzzy-CSarAFP* en la siguiente carpeta compartida:

<https://drive.google.com/open?id=1hMd-XRMQhGeCxWdPU7jhorKHJuNJ-L6t>

4.3. Análisis del tiempo promedio

Uno de los factores más importantes a tener en cuenta en cualquier análisis en el ámbito de la minería de flujo de datos es la capacidad del sistema para trabajar o procesar los datos con una rapidez suficiente. Las características y retos propios de la minería de flujo de datos hacen que sea necesario una velocidad de procesamiento rápida, capaz de ofrecer un análisis en tiempo real. Por tanto, resulta interesante conocer la capacidad de análisis del sistema creado, para así poder validar su capacidad para trabajar en una aplicación totalmente real, así como poder identificar aquellas tareas en las que el procesamiento de los datos no está totalmente optimizada.

La capacidad de dicho cálculo queda determinada por las especificaciones de la computadora utilizada. En este caso se trata de un *MacBook Air* con un procesador *Intel Core i7* de 2.2 GHz y una RAM de 8GB. En la figura 8 se recogen los valores promedio de tiempo invertido por dato para la aplicación de cada una de las etapas del proceso.

Tarea aplicada	Promedio del tiempo de análisis por dato (segundos)
Análisis de Sentimientos	0.3395
Comprobación de Idioma	0.0250
Análisis de Texto	0.1899
Estandarización del Texto	0.0001
Extracción Reglas (<i>Fuzzy-CSarAFP</i>)	0.0005
Extracción FCIs (<i>IncMine</i>)	0.0001
Extracción Reglas (<i>IncMine</i>)	0.0001

Cuadro 8. Resultados promedio del tiempo invertido por dato durante la aplicación de cada una de las etapas del proceso.

Se comprueba como el uso de *IncMine* es bastante más rápido que *Fuzzy-CSarAFP*. En general, se presenta un tiempo de cómputo levemente superior a 0.5 segundos por mensaje. Quizá este tiempo de estudio por *tweet* no es el más adecuado si se quiere realizar un sistema de minería de flujo de datos real, pues el tiempo de análisis es alto para esta tarea. Sin embargo, hay que tener en cuenta las circunstancias, pues se ha realizado con un ordenador común, sin prestaciones elevadas, por lo que en un sistema de mayor capacidad sería posible reducir esto. Además, ha habido que lidiar con un gran hándicap como es el hecho de aplicar un análisis de minería de texto para mensajes en español. Como se comentó anteriormente, aunque *spaCy* se presenta como una solución más lenta que el paquete *NLTK*, su capacidad para el análisis de textos en español (que no está tan desarrollado como el de inglés) es mucho mayor en comparación al *NLTK*. Este aspecto se presentaba como un factor fundamental para extraer la mejor información posible de los mensajes, de modo que, a pesar de buscar también la rapidez que todo sistema de minería de flujo de datos, se decidió elegir dicha opción.

5. Conclusiones

En conclusión, este proyecto se presenta como un ejercicio muy completo e interdisciplinar, en el cual se combina la minería de flujo de datos (y, en concreto, la extracción de reglas de asociación en tiempo real) con otras ramas de la ciencia de datos como la minería de texto, el análisis de sentimientos. La unión de todas estas disciplinas ha permitido generar un sistema capaz de extraer información a partir de una base de datos formada por elementos no estructurados, como es el caso de un histórico de mensajes de Twitter publicados durante un intervalo (lo suficientemente grande) de tiempo.

En concreto, este análisis se ha centrado en la minería de asociaciones en un entorno de aprendizaje de minería de flujo de datos, presentando un método muy interesante de cara a la búsqueda y extracción relaciones entre términos de forma *online* en un flujo de datos compuesto por la línea temporal de los *tweets* analizados. Estos mensajes pueden ser vistos como transacciones cuyos *items* se corresponden con las palabras del mensaje, de modo que es posible extraer relaciones o vínculos entre estos *items* (términos) de interés y el sentimiento de los mensajes. Para este proyecto, se ha elegido realizar dicho análisis sobre un tema de notable interés y con gran repercusión en la sociedad, como es la política, donde la búsqueda de asociaciones entre términos y sentimientos puede ser muy útil e interesante, pues puede permitir extraer conclusiones acerca de lo que sucede y se opina en la sociedad actual.

Tras la realización de un estudio en profundidad, finalmente se han obtenido las distintas reglas de asociación presentes en distintos espacios de tiempo del histórico de datos estudiado, considerando siempre el sentimiento del mensaje como consecuente. A partir de la población de reglas generadas mediante los distintos algoritmos presentados, se han estudiado la presencia de vínculos entre términos y sentimiento, cuantificados mediante la cantidad de reglas presentes. Para una adecuada visualización de estas relaciones entre términos se ha recurrido a gráficos especializados, como es el caso del diagrama *Sankey*, el cual recoge estos vínculos y los muestra a partir de enlaces proporcionales a dicha cantidad de reglas. Además, para presentar los resultados de una forma más fácilmente visible y comprensible por el usuario, se ha decidido implementar una visualización dinámica, en la que se muestre la evolución de las asociaciones entre términos en distintos intervalos temporales a través de una combinación en secuencia de los diagramas *Sankey* generados.

Así pues, aunque se han podido detectar pequeñas debilidades tanto en las capacidades de los algoritmos empleados como en los tiempos de ejecución del sistema, todas estas son comprensibles dada la situación presente ante el tipo de datos a estudiar y las herramientas disponibles. Por tanto, mejorar estos factores sigue siendo una posible tarea a realizar en una futura continuación del proyecto para una implementación completamente real.

A. Preparación del entorno de la API de Twitter en *Python*

Una de las primeras etapas a realizar dentro de este proyecto ha sido la adquisición de aquellos *tweets* de contenido político. Para poder obtener toda esta información ha sido necesario hacer uso de la API de Twitter para *Python*. Mediante el acceso a Twitter mediante API es posible acceder a dos tipos de datos, diferenciados según su diseño y método de acceso:

- **REST APIs:** Siguen una estrategia de recopilación de la información solicitada explícitamente por el usuario, mediante una consulta. Así, devuelve todos los datos publicados con anterioridad a la consulta y dentro de un cierto intervalo de tiempo.
- **Streaming APIs:** En las cuales se realiza inicialmente una consulta de información y se obtiene un flujo continuo de datos a medida que se van realizando publicaciones relacionadas con la consulta.

Aunque la estrategia *Streaming* sería la más adecuada debido a la naturaleza de este trabajo, ello requería de un dispositivo, *listener*, conectado a lo largo del tiempo para ir recibiendo la información. De tal modo, se ha optado por realizar una acumulación de datos mediante una estrategia *REST*. Así, una vez conseguidos todos los datos, se realiza el análisis de los mismos simulando su llegada en el tiempo.

Para poder trabajar con dicha API y poder solicitar información ha sido necesario crear una cuenta de desarrollador dentro de Twitter. Tras ello, se ha creado una *app* para la búsqueda de *tweets*. A continuación, se ha generado las credenciales (*keys* y *tokens*) propias de dicha *app*, las cuales serán nuestros identificadores a la hora de realizar consultas mediante la utilización de la API de Twitter para *Python*.

En este punto, las distintas herramientas y paquetes disponibles para hacer uso de la *REST-API* de Twitter en *Python* ofrecen dos vías principales de trabajo. La más sencilla consiste en utilización de las **búsquedas estándar** (*Standard Search*), la cual permite obtener todos los *tweets* dentro de los últimos 7 días. Sin embargo, puesto que también se requerían *tweets* de una antigüedad mayor a esos 7 días, se hacía imposible su obtención mediante la búsqueda estándar. De tal modo, se ha obtenido una suscripción *Premium*, dentro de su apartado gratuito, conocido como *Free Sandbox*, la cual permite al usuario acceder a una mayor cantidad de datos de Twitter para análisis más profundos como el que ocupa este proyecto.

Este tipo de cuenta nos permite realizar **búsquedas premium**, en las cuales se puede obtener la información de *tweets* dentro de todo el archivo de Twitter desde su apertura

en 2006 (opción *full-archive search*) o simplemente dentro del último mes (opción *30-days seach*). Una vez elegida una suscripción con *Premium*, es necesario acceder a la opción *Dev enviroments* dentro de la cuenta de *twitter-developer* y definir un entorno para su utilización en la búsqueda. Tras ello, se asociará este nuevo entorno a la *app* creada anteriormente, de manera que se podrán realizar búsquedas premium con dicha *app* de acuerdo a la suscripción deseada.

Una vez realizado esto, es necesario utilizar la siguiente sentencia en la terminal para obtener el *bearer token*, el cual será una de las credenciales necesarias para poder realizar búsquedas:

```
curl -u 'API key:API secret key' --data 'grant_type=client_credentials'
'https://api.twitter.com/oauth2/token'
```

donde **API key** y **API secret key** son dos claves alfanuméricas asociadas a la *app* que se creó inicialmente. En este punto, ya contamos con la identificación para realizar búsquedas de Twitter con *Python*. Así, para la búsqueda estándar se hará uso del paquete *TwitterAPI* [110], mientras que para las búsquedas premium se utilizará el paquete *searchtweets* [111]. En la figura 28 se señala esta división a la hora de realizar consultas, además de destacar los pros y contras de cada una de ellas:

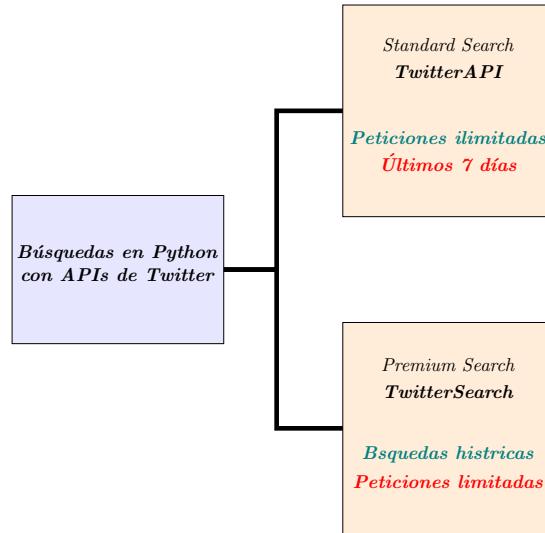


Figura 28. APIs de Twitter para realizar búsquedas en *Python*

En el caso del paquete *TwitterAPI* es posible comenzar a realizar peticiones simplemente tras definir los *keys* y *tokens* y sus respectivas claves, tal y como se muestra a continuación:

```
from TwitterAPI import TwitterAPI
api = TwitterAPI(CONSUMER_KEY, CONSUMER_SECRET, ACCESS_TOKEN, ACCESS_SECRET)
```

Sin embargo, en el caso del paquete *searchtweets* queda un paso más para poder realizar búsquedas premium: Es necesario crear un archivo *YAML* en el cual aparecen las credenciales que a continuación se muestran:

```
search_tweets_fullarchive_dev:
    account_type: premium
    endpoint: <FULL_URL_OF_ENDPOINT>
    bearer_token: <TOKEN>
```

Así pues, en este archivo aparece el tipo de cuenta utilizada (en este caso sería *premium*), así como el *bearer token* anteriormente obtenido y el *endpoint*, consistente en la URL del entorno que responderá a la petición realizada. Este *endpoint* tendrá una estructura del siguiente tipo:

https://api.twitter.com/1.1/tweets/search/name_product/name_enviroment.json

donde *name_enviroment* es el nombre del entorno que anteriormente se había creado y *name_product* es el tipo de producto elegido dentro de la suscripción, de modo que en este caso sería necesario definirlo como *fullarchive*. Tras todo esto, ya es posible cargar las credenciales tal y como se muestra a continuación:

```
from searchtweets import load_credentials, gen_rule_payload, ResultStream
premium_search_args = load_credentials(filename="premium_cred.yaml",
                                         yaml_key="search_tweets_premium")
```

De este modo, una vez hecho todo esto ya será posible realizar búsqueda de *tweets*, tanto estándar como premium, mediante las APIs en *Python* citadas: *TwitterAPI* y *searchtweets*.

B. Consultas con la API de Twitter en *Python*

Realización de las consultas con la API de Twitter

El uso de la API de Twitter requiere de la realización de una petición de búsqueda, la cual queda definida en base a una cierta consulta. Para un mejor resultado, resulta muy útil tener en cuenta diversos parámetros a definir dentro de dicha consulta (Si bien hay que tener en cuenta que la búsqueda *Premium* presenta ciertas limitaciones la definición de estos parámetros frente a la búsqueda *Standard*). Estos posibles parámetros quedan recogidos en la tabla 9:

Parámetro	¿Opcional?	Descripción	Ejemplo
<code>q</code>	NO	Texto para la consulta de máximo 500 caracteres y en formato UTF-8.	@UGR
<code>geocode</code>	SÍ	Restringue la búsqueda por la geoposición del usuario y un radio	37.1881700 -3.6066700 1km
<code>lang</code>	SÍ	Restringue la búsqueda a <i>tweets</i> de un cierto lenguaje.	es
<code>locale</code>	SÍ	Especificación del idioma de la consulta. (ja es la única opción disponible)	ja
<code>result_type</code>	SÍ	Especifica qué tipo de resultados de búsqueda prefiere recibir	popular mixed recent
<code>count</code>	SÍ	Número de <i>tweets</i> a devolver en página (15 por defecto).	100
<code>since</code>	SÍ	Restringue la búsqueda a los <i>tweets</i> publicados después de la fecha definida.	2019-07-27
<code>until</code>	SÍ	Restringue la búsqueda a los <i>tweets</i> publicados antes de la fecha definida.	2019-07-27
<code>max_id</code>	SÍ	Restringue la búsqueda a los <i>tweets</i> con una ID menor o igual a la ID especificada	54321
<code>include_entities</code>	SÍ	Parámetro booleano en el que se elige si se quieren incluir entidades	False/True.

Cuadro 9. Parámetros disponibles para la realización de consultas en Twitter.

Además, es posible realizar una descripción especial acerca del parámetro `q` anteriormente comentado y su utilización, pues mediante su especificación se realiza la definición de la consulta a realizar en el buscador de Twitter. De tal modo, cabe destacar las distintas posibilidades ofrecidas por Twitter al usuario de cara a la agilización y acotación de las búsquedas de mensajes en la red social. Así, es posible hacer uso de diferentes operadores de búsqueda, los cuales quedan descritos de forma básica en la tabla 10:

Operador	Funcionamiento
<i>A B</i>	Busca <i>tweets</i> que contengan <i>A</i> y <i>B</i> .
<i>' A B '</i>	Busca <i>tweets</i> que contengan exactamente <i>A B</i> .
<i>A OR B</i>	Busca <i>tweets</i> que contengan <i>A</i> o <i>B</i> .
<i>A -B</i>	Busca <i>tweets</i> que contengan <i>A</i> pero no <i>B</i> .
<i>from: C</i>	Busca <i>tweets</i> enviados por la cuenta <i>C</i> .
<i>to: C</i>	Busca <i>tweets</i> que responden a <i>tweets</i> enviados por la cuenta <i>C</i> .
<i>@C</i>	Busca <i>tweets</i> que mencionan a la cuenta <i>C</i> .
<i>list: C/lista</i>	Busca <i>tweets</i> enviados por una cuenta en la lista <i>lista</i> de <i>C</i> .
<i>A -filter:retweets</i>	Busca <i>tweets</i> que contengan <i>A</i> pero que no sean <i>retweets</i> .
<i>A filter:safe</i>	Busca <i>tweets</i> que contengan <i>A</i> pero que no sean potencialmente sensibles o peligrosos.
<i>A filter:images</i>	Busca <i>tweets</i> que contengan <i>A</i> y que adjunten imágenes.
<i>A filter:native_video</i>	Busca <i>tweets</i> que contengan <i>A</i> y que adjunten un video común, de <i>periscope</i> o de <i>vine</i> .
<i>A filter:media</i>	Busca <i>tweets</i> que contengan <i>A</i> y que adjunten archivos multimedia.
<i>A url:B</i>	Busca <i>tweets</i> que contengan <i>A</i> y que incluyan una URL que contenga <i>B</i> .
<i>A filter:links</i>	Busca <i>tweets</i> que contengan <i>A</i> y que contengan enlaces.
<i>A since:YYYY-MM-DD</i>	Busca <i>tweets</i> que contengan <i>A</i> , enviados después de la fecha <i>YYYY-MM-DD</i> .
<i>A until:YYYY-MM-DD</i>	Busca <i>tweets</i> que contengan <i>A</i> , enviados antes de la fecha <i>YYYY-MM-DD</i> .
<i>A :)</i>	Busca <i>tweets</i> que contengan <i>A</i> y que sean positivos.
<i>A :(</i>	Busca <i>tweets</i> que contengan <i>A</i> y que sean negativos.
<i>A ?</i>	Busca <i>tweets</i> que contengan <i>A</i> y que hagan una pregunta.

Cuadro 10. Operadores disponibles para la realización de consultas en Twitter.

Resultado de las consultas con la API de Twitter

Así pues, un factor común en todas las APIs de Twitter es que todas devuelven los *tweets* con una estructura JSON (*JavaScript Object Notation*), la cual se basa en pares clave-valor para describir los diferentes atributos del objeto. Además de la información general como autor, mensaje, ID, *timestamp*, se presentan datos adicionales como los metadatos del usuario, así como las entidades presentes en el *tweet* (*hashtags*, menciones, *links*, multimedia...). Estos JSON deben obtenerse en formato UTF-8 y el *parser* creado debe ser tolerante a variaciones en el orden de los campos, así como a valores perdidos, pues no todos los campos aparecen en todos los contextos.

Estructura para un *tweet* común

Por tanto, los *tweets* presentan una estructura básica común, de manera que todos ellos presentan, al menos, información acerca de campos básicos propios de la información general del *tweet*. La estructura básica de los JSON generados en los *tweets* se muestra a continuación:

```
{
  "created_at": ,
  "id_str": ,
  "text": ,
  "user": {
  },
  "place": {
  },
  "entities": {
  },
  "extended_entities": {
  }
}
```

Estructura para un *extended tweet* común

Sin embargo, la reciente actualización de Twitter permite escribir textos superiores a los famosos 140 caracteres iniciales, lo cual supuso una alteración en esta estructura puesto que el atributo `text` pasaba a quedar truncado. Así pues, a partir de dicha actualización, los *tweets* que superan dicho límite pasaban a incluir un campo adicional, denominado como `extended_tweet`, en el cual se muestra el mensaje al completo y otros metadatos. La estructura de los JSON generados en este tipo de *tweets* se muestra a continuación:

```
{
  "created_at": ,
  "id_str": ,
  "text": ,
  "display_text_range": ,
  "truncated": ,
  "user": {
    "id_str": ,
    "screen_name": ,
  },
  "extended_tweet": {
    "full_text": ,
    "display_text_range": ,
    "entities": {
      "hashtags": [
        {
          "text": ,
          "indices": [
          ]
        }
      ],
      "entities": {
        "hashtags": []
      }
    }
  }
}
```

Estructura para un *retweet* o un *tweet* citado

Además, la presencia de *retweets* y *tweet* citado (*quoted*) de *tweets*, la estructura JSON se ve modificada, de modo que esta pasará a contar con varios objetos, cada uno de los cuales contendrá sus respectivos metadatos. Así, la estructura JSON contará con un nuevo campo que, dependiendo de la acción realizada, se denominará como `retweeted_status` o `quoted_status`. A continuación, se muestra la estructura propia de los JSON generados cuando se trata de un *retweet*:

```
{  
    "tweet": {  
        "text": "RT @author original message"  
        "user": {  
            "screen_name": "Retweeter"  
        },  
        "retweeted_status": {  
            "text": "original message".  
            "user": {  
                "screen_name": "OriginalTweeter"  
            },  
            "place": {  
            },  
            "entities": {  
            },  
            "extended_entities": {  
            }  
        },  
        "entities": {  
        },  
        "extended_entities": {  
        }  
    }  
}
```

Un ejemplo real

A modo de ejemplo, se ha elegido un mensaje para el análisis y comprobar su correspondiente estructura en formato JSON. El *tweet* elegido es el siguiente:



Figura 29. Tweet ejemplo para mostrar la estructura JSON generada por la API de Twitter.

Y a continuación se presenta como ejemplo la estructura JSON asociada al mismo:

```
{
  "created_at": "Thu Apr 06 15:24:15 +0000 2017",
  "id_str": "850006245121695744",
  "text": "1/ Today we're sharing our vision for the future of the Twitter API platform! https://t.co/XweGngmxlP",
  "user": {
    "id": 2244994945,
    "name": "Twitter Dev",
    "screen_name": "TwitterDev",
    "location": "Internet",
    "url": "https://dev.twitter.com",
    "description": "Your official source for Twitter Platform news, updates & events. Need technical help? Visit https://twittercommunity.com \u2328 \ufe0f #TapIntoTwitter"
  },
  "place": {},
  "entities": {
    "hashtags": [
      {}
    ],
    "urls": [
      {
        "url": "https://t.co/XweGngmxlP",
        "unwound": {
          "url": "https://cards.twitter.com/cards/18ce53wgo4h3xo1c",
          "title": "Building the Future of the Twitter API Platform"
        }
      }
    ],
    "user_mentions": []
  }
}
```

C. Análisis de sentimientos con *SentiStrength*

Como se comentó anteriormente, la clasificación del sentimiento en los mensajes del conjunto de mensajes extraídos de *Twittter* ha sido realizado utilizando la herramienta de *SentiStrength*, desarrollada por [90]. Este *software* está construido en *Java* y hace uso de una serie de diccionarios definidos en base a la categoría gramatical de las palabras, siguiendo la idea propuesta y descrita en el proyecto de [80].

De tal modo, estos diccionarios están formados por palabras, emoticonos e incluso expresiones o frases hechas. Los distintos términos incluidos en estos diccionarios son aquellos con un sentimiento de positividad o negatividad asociado, de forma que dicho sentimiento se presenta de forma cuantizada, con valores entre +1 y +4 (o -1 y -4). En el caso de los emoticonos se utilizados asociando un sentimiento positivo(+1), negativo(-1) o neutro(0). Además, cabe destacar que también son utilizados para la detección de ironías.

Para su utilización, simplemente es necesario ejecutar el paquete de java con el *software*, indicando la carpeta en la que se sitúan los diccionarios de interés. Tras esto, se indicará el texto a analizar, utilizando una concatenación entre palabras mediante el símbolo +. De tal modo, un ejemplo para su ejecución es el siguiente:

```
java -jar SentiStrength.jar sentidata SentiStrength_DataFolder/ text texto+para+analizar
```

Así pues, ofrece como resultado dos valores: El de positividad y el de negatividad del texto, los cuales se corresponden respectivamente con el valor del término más positivo y negativo encontrado en dicho texto. De tal modo, el sentimiento presente en el texto ha sido obtenido sacando la diferencia entre estos valores de positividad y negatividad:

- Si el valor de la diferencia es mayor que 1, el sentimiento del texto es **POSITIVO**.
- Si el valor de la diferencia es menor que -1, el sentimiento del texto es **NEGATIVO**.
- Si el valor de la diferencia está entre -1 y 1, el sentimiento del texto es **NEUTRO**.

Además de clasificar textos únicos, *SentiStrength* también permite realizar clasificaciones de múltiples textos, los cuales pueden estar presentes en forma de línea o como columna dentro de un cierto archivo. Sin embargo, puesto que en este caso se ha trabajado con este *software* sobre un *dataset* en *Python*, no ha sido necesario utilizar estas características. Sí que ha sido importante utilizar la función `stdin`, la cual permite integrar de forma eficiente este *software* en otros programas distintos a Java. Los resultados obtenidos serán obtenidos en una salida estándar, *STDout*.

Por otro lado, al utilizar unos diccionarios claramente separados y estructurados, este *software* permite mejorar las capacidades de clasificación en función de las características de los textos a clasificar. Así, es tremadamente fácil corregir tendencias hacia un cierto sentimiento en base a la eliminación de aquellos términos que en dichos casos no presentan relevancia en lo que a sentimientos se refieren. Esto se puede realizar fácilmente mediante la edición de los archivos *EmotionLookupTable.txt*. Asimismo, es posible añadir nuevas frases hechas y expresiones mediante su adición en el archivo *IdiomLookupTable.txt*.

Además, esta herramienta permite incluir distintos parámetros con el objetivo de mejorar el funcionamiento de la clasificación y adaptarla a las necesidades del usuario. En la figura 11 se presenta una lista con los distintos parámetros disponibles en *SentiStrength*:

Parámetro	Descripción
<code>explain</code>	Devuelve una explicación del análisis realizado para el cálculo de los valores de positividad y negatividad.
<code>sentidata</code>	Indica la localización de la carpeta en la que se sitúan los diccionarios lingüísticos.
<code>keywords</code>	Permite incluir una lista de palabras separadas por comas, de modo que clasificará el sentimiento sólo con las partes del texto cercanas a las palabras de dicha lista.
<code>scale</code>	Permite cambiar la escala de los valores de positividad y negatividad resultantes
<code>trinary</code>	Añade un tercer parámetro en la salida con el sentimiento final según una división positivo(1)-neutro(0)-negativo(-1)
<code>binary</code>	Añade un tercer parámetro con el sentimiento final según una división positivo(1)-negativo(-1)
<code>resultsexension</code>	Indica la extensión del archivo en la salida.
<code>outputFolder</code>	Indica la carpeta en la que guardar la salida.
<code>alwaysSplitWordsAtApostrophes</code>	Permite dividir palabras cuando contienen apóstrofes.
<code>noBoosters</code>	Permite ignorar intensificadores de sentimiento (muy, poco, ...)
<code>noNegatingPositiveFlipsEmotion</code>	Permite ignorar los inversores de sentimiento.
<code>noNegatingNegativeNeutralisesEmotion</code>	Evita utilizar inversores para neutralizar sentimientos.
<code>maxWordsBeforeSentimentToNegate</code>	Número de palabras máximas a admitir entre el inversor y la palabra con sentimiento (Por defecto, 0).
<code>negatedWordStrengthMultiplier</code>	Aumenta la fuerza de los términos negados.
<code>noIdioms</code>	Permite ignorar la lista de frases hechas.
<code>noEmoticons</code>	Permite ignorar la lista de emoticonos.
<code>questionsReduceNeg</code>	Permite reducir el valor del sentimiento cuando se trata de preguntas.
<code>exclamations2</code>	Permite intensificar un sentimiento en caso de existan exclamaciones.
<code>noDictionary</code>	Evita intentar corregir palabras erróneas o desconocidas mediante una comparación con el diccionario.
<code>mood</code>	Permite definir el valor neutral de sentimiento (mediante [-1,0,1]).
<code>noMultiplePosWords</code>	Evita que el sentimiento positivo aumente ante múltiples palabras positivas.
<code>noMultipleNegWords</code>	Evita que el sentimiento negativo aumente ante múltiples palabras negativas.
<code>noIgnoreBoosterWordsAfterNEGATIVES</code>	Evita ignorar intensificadores después de palabras negativas.
<code>noDeleteExtraDuplicateLetters</code>	Evita eliminar letras repetidas consecutivamente en palabras.
<code>illegalDoubleLettersInWordMiddle</code>	Evita que una serie de letras aparezcan duplicados en el interior de una palabra.
<code>illegalDoubleLettersAtWordEnd</code>	Evita que una serie de letras aparezcan duplicados al final de una palabra.
<code>noMultipleLetters</code>	Evita utilizar la presencia de letras consecutivas en una palabra como intensificador.

Cuadro 11. Parámetros disponibles para el análisis de sentimientos con *SentiStrength*.

De tal modo, todas estas opciones hacen de este *software* una opción bastante correcta en la clasificación de sentimientos y flexible a las necesidades del usuario según el conjunto de datos de estudio. Además, esta herramienta está diseñada para su aplicación en textos cortos, por lo que encaja con la filosofía inicial de Twitter, de modo que es ideal para la tarea realizada.

D. Reglas de asociación obtenidas para el problema de USA

En las tablas 13 y 12 se muestra el conjunto de las reglas de asociación (con sentimiento como consecuente) que se han obtenido por los distintos métodos (*IncMine* y *Fuzzy-CSarAFP*) tras analizar todos los datos disponibles en la base de datos de USA. Además, se incluyen los valores de las distintas medidas de calidad. Se presenta ordenada por valor de *LIFT*.

IncMine

Rule	Supp_A	Supp_C	Support	Confidence	Leverage	LIFT
Woman+Feminism+Night→POSITIVE	0.033333	0.355556	0.033333	1.000000	0.021481	2.812500
Atheism+Night→POSITIVE	0.055556	0.355556	0.044444	0.800000	0.024691	2.250000
Atheism→POSITIVE	0.144444	0.355556	0.111111	0.769231	0.059753	2.163462
Woman+Feminism→POSITIVE	0.044444	0.355556	0.033333	0.750000	0.017531	2.109375
Atheism+Morning→POSITIVE	0.066667	0.355556	0.044444	0.666667	0.020741	1.875000
Hillary_Clinton+Donald_Trump+Night→NEGATIVE	0.033333	0.566667	0.033333	1.000000	0.014444	1.764706
Hillary_Clinton+Donald_Trump→NEGATIVE	0.033333	0.566667	0.033333	1.000000	0.014444	1.764706
Child→NEGATIVE	0.033333	0.566667	0.033333	1.000000	0.014444	1.764706
Pregnant+Abortion+Night→NEGATIVE	0.022222	0.566667	0.022222	1.000000	0.009630	1.764706
Woman+Abortion+Night→NEGATIVE	0.022222	0.566667	0.022222	1.000000	0.009630	1.764706
Antifeminism+Feminism+Morning→NEGATIVE	0.022222	0.566667	0.022222	1.000000	0.009630	1.764706
Abortion+child→NEGATIVE	0.022222	0.566667	0.022222	1.000000	0.009630	1.764706
Child+Night→NEGATIVE	0.022222	0.566667	0.022222	1.000000	0.009630	1.764706
Woman+Abortion→NEGATIVE	0.022222	0.566667	0.022222	1.000000	0.009630	1.764706
Man→NEGATIVE	0.022222	0.566667	0.022222	1.000000	0.009630	1.764706
Black→NEGATIVE	0.022222	0.566667	0.022222	1.000000	0.009630	1.764706
Rape→NEGATIVE	0.022222	0.566667	0.022222	1.000000	0.009630	1.764706
Feminism+Night→POSITIVE	0.077778	0.355556	0.044444	0.571429	0.016790	1.607143
Abortion+Night→NEGATIVE	0.055556	0.566667	0.044444	0.800000	0.012963	1.411765
Feminism+Morning→NEGATIVE	0.055556	0.566667	0.044444	0.800000	0.012963	1.411765
Donald_Trump+Morning→POSITIVE	0.088889	0.355556	0.044444	0.500000	0.012840	1.406250
Climate_Change+Morning→POSITIVE	0.044444	0.355556	0.022222	0.500000	0.006420	1.406250
Donald_Trump+Night→NEGATIVE	0.244444	0.566667	0.177778	0.727273	0.039259	1.283422
Hillary_Clinton→NEGATIVE	0.155556	0.566667	0.111111	0.714286	0.022963	1.260504
Hillary_Clinton+Night→NEGATIVE	0.144444	0.566667	0.100000	0.692308	0.018148	1.221719
Donald_Trump→NEGATIVE	0.355556	0.566667	0.244444	0.687500	0.042963	1.213235

Cuadro 12. Reglas de asociación obtenidas para el problema de USA con *IncMine*, utilizando los parámetros de $minSup = 0.05$, $\sigma_{relax}=0.3$ y ventana deslizante de 90 datos.

Fuzzy-CSarAFP

Rule	Support	Confidence	LIFT
Morning+Climate_Change→NEITHER	0.010	0.224	3.348
Climate_Change→NEITHER	0.027	0.188	2.792
Night+Climate_Change→NEITHER	0.010	0.182	2.705
Afternoon+Climate_Change→NEITHER	0.008	0.174	2.588
Afternoon+Atheism→POSITIVE	0.031	0.620	1.981
Night+Atheism→POSITIVE	0.038	0.609	1.944
Catholic→POSITIVE	0.019	0.565	1.933
Atheism→POSITIVE	0.090	0.597	1.908
Morning+Atheism→POSITIVE	0.023	0.547	1.741
Afternoon+Feminism+Man→NEGATIVE	0.011	0.883	1.418
Night+Feminism+Man→NEGATIVE	0.013	0.861	1.383
Afternoon+Man→NEGATIVE	0.016	0.837	1.343
Child→NEGATIVE	0.033	0.777	1.246
Morning→NEITHER	0.016	0.082	1.266
Morning+Feminism→NEGATIVE	0.039	0.786	1.265
Man→NEGATIVE	0.041	0.781	1.263
Hillary_Clinton+Woman→POSITIVE	0.005	0.390	1.245
Feminism→NEGATIVE	0.152	0.768	1.232
Night+Feminism→NEGATIVE	0.067	0.767	1.232
Morning+Climate_Change→POSITIVE	0.016	0.370	1.230
Religion→NEGATIVE	0.010	0.767	1.230
Afternoon+Feminism→NEGATIVE	0.048	0.759	1.217
Afternoon+Woman→NEGATIVE	0.024	0.757	1.217

Cuadro 13. Reglas de asociación obtenidas para el problema de USA con *Fuzzy-CSarAFP*, utilizando la variable sentimiento como consecuente y el resto como antecedente, además de $\theta_{mna} = 3$, $\nu=1$, $p_C = p_M = 0.4$

E. Reglas de asociación obtenidas para el problema de España

En las tablas 15 y 14 se muestra el conjunto de las reglas de asociación (con sentimiento como consecuente) que se han obtenido por los distintos métodos (*IncMine* y *Fuzzy-CSarAFP*) tras analizar todos los datos disponibles en la base de datos de USA. Además, se incluyen los valores de las distintas medidas de calidad. Se presenta ordenada por valor de *LIFT*.

IncMine

Rule	Supp_A	Supp_C	Support	Confidence	Leverage	LIFT
VOX+Izquierda+Derecha+Weekday→NEGATIVE	0.023158	0.264211	0.023158	1.000000	0.017039	3.784861
Gobierno+Unidas_Podemos+Izquierda+Weekday→POSITIVE	0.044211	0.295789	0.044211	1.000000	0.031134	3.380783
Gobierno+Unidas_Podemos+Izquierda→POSITIVE	0.044211	0.295789	0.044211	1.000000	0.031134	3.380783
PSOE+Adriana_Lastra+Weekday+Afternoon→POSITIVE	0.018947	0.295789	0.018947	1.000000	0.013343	3.380783
PSOE+Elecciones+siRT+Weekday→POSITIVE	0.014737	0.295789	0.013684	0.928571	0.009325	3.139298
PSOE+Elecciones+siRT→POSITIVE	0.014737	0.295789	0.013684	0.928571	0.009325	3.139298
PSOE+Elecciones+Weekday+Afternoon→POSITIVE	0.017895	0.295789	0.015789	0.882353	0.010496	2.983044
PSOE+Elecciones+Weekday→POSITIVE	0.017895	0.295789	0.015789	0.882353	0.010496	2.983044
Izquierda+Derecha+siRT+Weekday→NEGATIVE	0.031579	0.264211	0.024211	0.766667	0.015867	2.901726
VOX+Izquierda+Weekday+Afternoon→NEGATIVE	0.030526	0.264211	0.023158	0.758621	0.015093	2.871274
Izquierda+Derecha+Weekday+Afternoon→NEGATIVE	0.034737	0.264211	0.026316	0.757576	0.017138	2.867319
Izquierda+Derecha+Weekday→NEGATIVE	0.034737	0.264211	0.026316	0.757576	0.017138	2.867319
Gobierno+Izquierda+siRT+Weekday→POSITIVE	0.057895	0.295789	0.046316	0.800000	0.029191	2.704626
Gobierno+Izquierda+siRT→POSITIVE	0.057895	0.295789	0.046316	0.800000	0.029191	2.704626
Gobierno+Unidas_Podemos+siRT+Weekday→POSITIVE	0.075789	0.295789	0.060000	0.791667	0.037582	2.676453
Gobierno+Unidas_Podemos+siRT→POSITIVE	0.075789	0.295789	0.060000	0.791667	0.037582	2.676453
Gobierno+Izquierda+Weekday+Afternoon→POSITIVE	0.058947	0.295789	0.046316	0.785714	0.028880	2.656329
Gobierno+Izquierda+Weekday→POSITIVE	0.058947	0.295789	0.046316	0.785714	0.028880	2.656329
Gobierno+Izquierda→POSITIVE	0.058947	0.295789	0.046316	0.785714	0.028880	2.656329
Gobierno+siRT+Weekday+Afternoon→POSITIVE	0.176842	0.295789	0.137895	0.779762	0.085587	2.636206
Gobierno+siRT+Weekday→POSITIVE	0.176842	0.295789	0.137895	0.779762	0.085587	2.636206
Gobierno+siRT→POSITIVE	0.176842	0.295789	0.137895	0.779762	0.085587	2.636206
Gobierno+Unidas_Podemos+Weekday+Afternoon→POSITIVE	0.081053	0.295789	0.063158	0.779221	0.039183	2.634376
Gobierno+Unidas_Podemos+Weekday→POSITIVE	0.081053	0.295789	0.063158	0.779221	0.039183	2.634376
Gobierno+Unidas_Podemos→POSITIVE	0.081053	0.295789	0.063158	0.779221	0.039183	2.634376
Gobierno+Weekday+Afternoon→POSITIVE	0.190526	0.295789	0.142105	0.745856	0.085750	2.521578
Gobierno+Weekday→POSITIVE	0.190526	0.295789	0.142105	0.745856	0.085750	2.521578
Gobierno→POSITIVE	0.190526	0.295789	0.142105	0.745856	0.085750	2.521578
PSOE+gobierno+siRT+Weekday→POSITIVE	0.122105	0.295789	0.090526	0.741379	0.054409	2.506443
PSOE+gobierno+siRT→POSITIVE	0.122105	0.295789	0.090526	0.741379	0.054409	2.506443
VOX+Derecha+siRT+Weekday→NEGATIVE	0.033684	0.264211	0.022105	0.656250	0.013206	2.483815
VOX+Derecha+Weekday+Afternoon→NEGATIVE	0.037895	0.264211	0.024211	0.638889	0.014198	2.418105
PSOE+gobierno+Weekday+Afternoon→POSITIVE	0.131579	0.295789	0.093684	0.712000	0.054765	2.407117
PSOE+gobierno+Weekday→POSITIVE	0.131579	0.295789	0.093684	0.712000	0.054765	2.407117
PSOE+gobierno→POSITIVE	0.131579	0.295789	0.093684	0.712000	0.054765	2.407117
PSOE+Unidas_Podemos+Izquierda+Weekday→NEUTRAL	0.129474	0.440000	0.127368	0.983740	0.070400	2.235772
PSOE+Izquierda+siRT+Weekday→NEUTRAL	0.143158	0.440000	0.138947	0.970588	0.075958	2.205882

Rule	Supp_A	Supp_C	Support	Confidence	Leverage	LIFT
PSOE+Izquierda+siRT→NEUTRAL	0.143158	0.440000	0.138947	0.970588	0.075958	2.205882
PSOE+Unidas_Podemos+pacto+Weekday→NEUTRAL	0.131579	0.440000	0.127368	0.968000	0.069474	2.200000
PSOE+Izquierda+Weekday+Afternoon→NEUTRAL	0.144211	0.440000	0.138947	0.963504	0.075495	2.189781
PSOE+Izquierda+Weekday→NEUTRAL	0.144211	0.440000	0.138947	0.963504	0.075495	2.189781
PSOE+Izquierda→NEUTRAL	0.144211	0.440000	0.138947	0.963504	0.075495	2.189781
PSOE+Partido_Popular+Weekday+Afternoon→POSITIVE	0.035789	0.295789	0.023158	0.647059	0.012572	2.187565
PSOE+Partido_Popular+Weekday→POSITIVE	0.035789	0.295789	0.023158	0.647059	0.012572	2.187565
Unidas_Podemos+pacto+Weekday+Afternoon→NEUTRAL	0.147368	0.440000	0.131579	0.892857	0.066737	2.029221
PSOE+gobierno+Unidas_Podemos+Weekday→POSITIVE	0.035789	0.295789	0.018947	0.529412	0.008361	1.789826
PSOE+gobierno+Unidas_Podemos→POSITIVE	0.035789	0.295789	0.018947	0.529412	0.008361	1.789826
PSOE+Unidas_Podemos+siRT+Weekday→NEUTRAL	0.190526	0.440000	0.148421	0.779006	0.064589	1.770467
PSOE+Unidas_Podemos+sirt→NEUTRAL	0.190526	0.440000	0.148421	0.779006	0.064589	1.770467
PSOE+Unidas_Podemos+Weekday+Afternoon→NEUTRAL	0.200000	0.440000	0.150526	0.752632	0.062526	1.710526
PSOE+Unidas_Podemos+Weekday→NEUTRAL	0.200000	0.440000	0.150526	0.752632	0.062526	1.710526
PSOE+Unidas_Podemos→NEUTRAL	0.200000	0.440000	0.150526	0.752632	0.062526	1.710526
elecciones+Weekday+Afternoon→POSITIVE	0.031579	0.295789	0.015789	0.500000	0.006449	1.690391
Pedro_Sanchez+Unidas_Podemos+siRT+Weekday→NEUTRAL	0.027368	0.440000	0.020000	0.730769	0.007958	1.660839
Pacto+siRT+Weekday→NEUTRAL	0.187368	0.440000	0.135789	0.724719	0.053347	1.647089
Elecciones+siRT+Weekday+Afternoon→POSITIVE	0.028421	0.295789	0.013684	0.481481	0.005278	1.627784
Derecha+siRT+Weekday+Afternoon→NEGATIVE	0.060000	0.264211	0.025263	0.421053	0.009411	1.593625
Derecha+siRT+Weekday→NEGATIVE	0.060000	0.264211	0.025263	0.421053	0.009411	1.593625
Derecha+siRT→NEGATIVE	0.060000	0.264211	0.025263	0.421053	0.009411	1.593625
Derecha+Weekday+Afternoon→NEGATIVE	0.070526	0.264211	0.029474	0.417910	0.010840	1.581733
Derecha+Weekday→NEGATIVE	0.070526	0.264211	0.029474	0.417910	0.010840	1.581733
Derecha→NEGATIVE	0.070526	0.264211	0.029474	0.417910	0.010840	1.581733
Unidas_Podemos+Izquierda+Weekday+Afternoon→NEUTRAL	0.184211	0.440000	0.127368	0.691429	0.046316	1.571429
VOX+Ciudadanos+siRT+Weekday→NEUTRAL	0.138947	0.440000	0.093684	0.674242	0.032547	1.532369
Partido_Popular+Ciudadanos+siRT+Weekday→NEUTRAL	0.140000	0.440000	0.093684	0.669173	0.032084	1.520848
Pedro_Sanchez+Unidas_Podemos+Weekday+Afternoon→NEUTRAL	0.031579	0.440000	0.021053	0.666667	0.007158	1.515152
Pedro_Sanchez+Unidas_Podemos+Weekday→NEUTRAL	0.031579	0.440000	0.021053	0.666667	0.007158	1.515152
Pedro_Sanchez+Unidas_Podemos→NEUTRAL	0.031579	0.440000	0.021053	0.666667	0.007158	1.515152
Ciudadanos+siRT+Weekday+Afternoon→NEUTRAL	0.141053	0.440000	0.093684	0.664179	0.031621	1.509498
Ciudadanos+siRT+Weekday→NEUTRAL	0.141053	0.440000	0.093684	0.664179	0.031621	1.509498
Ciudadanos+siRT→NEUTRAL	0.141053	0.440000	0.093684	0.664179	0.031621	1.509498
VOX+Partido_Popular+Ciudadanos+Weekday→NEUTRAL	0.152632	0.440000	0.098947	0.648276	0.031789	1.473354
VOX+Ciudadanos+Weekday→NEUTRAL	0.152632	0.440000	0.098947	0.648276	0.031789	1.473354
Izquierda+siRT+Weekday+Afternoon→NEUTRAL	0.224211	0.440000	0.145263	0.647887	0.046611	1.472471
Izquierda+siRT+Weekday→NEUTRAL	0.224211	0.440000	0.145263	0.647887	0.046611	1.472471
Izquierda+siRT→NEUTRAL	0.224211	0.440000	0.145263	0.647887	0.046611	1.472471
Partido_Popular+Ciudadanos+Weekday+Afternoon→NEUTRAL	0.153684	0.440000	0.098947	0.643836	0.031326	1.463263
Partido_Popular+Ciudadanos+Weekday→NEUTRAL	0.153684	0.440000	0.098947	0.643836	0.031326	1.463263
PSOE+Inmigracion+siRT+Weekday→NEUTRAL	0.029474	0.440000	0.018947	0.642857	0.005979	1.461039
PSOE+noRT+Weekday+Afternoon→NEGATIVE	0.041053	0.264211	0.015789	0.384615	0.004943	1.455716
PSOE+noRT+Weekday→NEGATIVE	0.041053	0.264211	0.015789	0.384615	0.004943	1.455716
PSOE+noRT→NEGATIVE	0.041053	0.264211	0.015789	0.384615	0.004943	1.455716
Ciudadanos+Weekday+Afternoon→NEUTRAL	0.154737	0.440000	0.098947	0.639456	0.030863	1.453309
Ciudadanos+Weekday→NEUTRAL	0.154737	0.440000	0.098947	0.639456	0.030863	1.453309
Izquierda+Weekday+Afternoon→NEUTRAL	0.228421	0.440000	0.145263	0.635945	0.044758	1.445329
Izquierda+Weekday→NEUTRAL	0.228421	0.440000	0.145263	0.635945	0.044758	1.445329
Izquierda→NEUTRAL	0.228421	0.440000	0.145263	0.635945	0.044758	1.445329
Unidas_Podemos+siRT+Weekday+Afternoon→NEUTRAL	0.271579	0.440000	0.172632	0.635659	0.053137	1.444679
Unidas_Podemos+siRT+Weekday→NEUTRAL	0.271579	0.440000	0.172632	0.635659	0.053137	1.444679
Unidas_Podemos+siRT→NEUTRAL	0.271579	0.440000	0.172632	0.635659	0.053137	1.444679
Inmigracion+siRT+Weekday+Afternoon→NEUTRAL	0.030526	0.440000	0.018947	0.620690	0.005516	1.410658

<i>Rule</i>	<i>Supp_A</i>	<i>Supp_C</i>	<i>Support</i>	<i>Confidence</i>	<i>Leverage</i>	<i>LIFT</i>
PSOE+Inmigracion+Weekday+Afternoon→NEUTRAL	0.030526	0.440000	0.018947	0.620690	0.005516	1.410658
PSOE+Inmigracion+Weekday→NEUTRAL	0.030526	0.440000	0.018947	0.620690	0.005516	1.410658
Unidas_Podemos+Weekday+Afternoon→NEUTRAL	0.290526	0.440000	0.176842	0.608696	0.049011	1.383399
Unidas_Podemos+Weekday→NEUTRAL	0.290526	0.440000	0.176842	0.608696	0.049011	1.383399
Unidas_Podemos→NEUTRAL	0.290526	0.440000	0.176842	0.608696	0.049011	1.383399
Partido_Popular+siRT+Weekday+Afternoon→NEGATIVE	0.265263	0.264211	0.096842	0.365079	0.026757	1.381774
Partido_Popular+siRT+Weekday→NEGATIVE	0.265263	0.264211	0.096842	0.365079	0.026757	1.381774
Partido_Popular+siRT→NEGATIVE	0.265263	0.264211	0.096842	0.365079	0.026757	1.381774
PSOE+Pedro_Sanchez+Weekday+Afternoon→POSITIVE	0.044211	0.295789	0.017895	0.404762	0.004818	1.368412
PSOE+Pedro_Sanchez+Weekday→POSITIVE	0.044211	0.295789	0.017895	0.404762	0.004818	1.368412
PSOE+Pedro_Sanchez→POSITIVE	0.044211	0.295789	0.017895	0.404762	0.004818	1.368412
Partido_Popular+Weekday+Afternoon→NEGATIVE	0.290526	0.264211	0.103158	0.355072	0.026398	1.343900
Partido_Popular+Weekday→NEGATIVE	0.290526	0.264211	0.103158	0.355072	0.026398	1.343900
Partido_Popular→NEGATIVE	0.290526	0.264211	0.103158	0.355072	0.026398	1.343900
VOX+siRT+Weekday+Afternoon→NEUTRAL	0.268421	0.440000	0.153684	0.572549	0.035579	1.301248
VOX+siRT+Weekday→NEUTRAL	0.268421	0.440000	0.153684	0.572549	0.035579	1.301248
VOX+siRT→NEUTRAL	0.268421	0.440000	0.153684	0.572549	0.035579	1.301248
noRT+Weekday+Afternoon→NEGATIVE	0.105263	0.264211	0.035789	0.340000	0.007978	1.286853
noRT+Weekday→NEGATIVE	0.105263	0.264211	0.035789	0.340000	0.007978	1.286853
noRT→NEGATIVE	0.105263	0.264211	0.035789	0.340000	0.007978	1.286853
VOX+Weekday+Afternoon→NEUTRAL	0.326316	0.440000	0.184211	0.564516	0.040632	1.282991
VOX+Weekday→NEUTRAL	0.326316	0.440000	0.184211	0.564516	0.040632	1.282991
VOX→NEUTRAL	0.326316	0.440000	0.184211	0.564516	0.040632	1.282991
negociacion+Weekday+Afternoon→NEUTRAL	0.021053	0.440000	0.011579	0.550000	0.002316	1.250000
Pedro_Sanchez+siRT+Weekday+Afternoon→NEUTRAL	0.050526	0.440000	0.027368	0.541667	0.005137	1.231061
Pedro_Sanchez+siRT+Weekday→NEUTRAL	0.050526	0.440000	0.027368	0.541667	0.005137	1.231061
Pedro_Sanchez+siRT→NEUTRAL	0.050526	0.440000	0.027368	0.541667	0.005137	1.231061

Cuadro 14. Reglas de asociación obtenidas para el problema de España con *IncMine*, utilizando los parámetros de $minSup = 0.05$, $\sigma_{relax}=0.3$ y ventana deslizante de 950 datos.

Fuzzy-CSarAFP

<i>Rule</i>	<i>Support</i>	<i>Confidence</i>	<i>LIFT</i>
PSOE+Calvo→POSITIVE	0.013	0.698	2.412
siRT+PSOE+Calvo→POSITIVE	0.009	0.651	2.089
siRT+Calvo→POSITIVE	0.012	0.584	1.934
Calvo→POSITIVE	0.013	0.540	1.797
Weekend+siRT+UPodemos→POSITIVE	0.029	0.466	1.666
Weekend+UPodemos→POSITIVE	0.035	0.441	1.508
Weekend+PSOE→POSITIVE	0.054	0.406	1.456
siRT+UPodemos→POSITIVE	0.075	0.411	1.499
Afternoon+Weekend+siRT→POSITIVE	0.036	0.380	1.361
Weekend+siRT→POSITIVE	0.089	0.376	1.342
PP→NEGATIVE	0.053	0.318	1.337
Morning+VOX→NEGATIVE	0.035	0.322	1.326
Weekday+siRT+PP→NEGATIVE	0.035	0.326	1.325
siRT+PP→NEGATIVE	0.041	0.322	1.310
Afternoon+Weekend→POSITIVE	0.044	0.367	1.306
Weekday+VOX→NEGATIVE	0.057	0.310	1.287
Weekday+PP→NEGATIVE	0.042	0.315	1.277
gobierno→POSITIVE	0.100	0.399	1.276
UPodemos→POSITIVE	0.094	0.395	1.274
VOX→NEGATIVE	0.073	0.305	1.264
Iglesias→POSITIVE	0.023	0.438	1.255
Morning+Weekday+CS→NEUTRAL	0.016	0.557	1.250
Weekday+UPodemos→POSITIVE	0.077	0.389	1.247
Afternoon+Sánchez→POSITIVE	0.034	0.384	1.224
siRT+Sánchez→POSITIVE	0.047	0.338	1.214
Morning+siRT+PSOE→POSITIVE	0.054	0.352	1.201

Cuadro 15. Reglas de asociación obtenidas para el problema de España con *Fuzzy-CSarAFP*, utilizando la variable sentimiento como consecuente y el resto como antecedente, además de $\theta_{mna} = 6$, $\nu=1$, $p_C = p_M = 0.1$

Referencias

- [1] Nan Jiang and Le Gruenwald. Research issues in data stream association rule mining. *ACM Sigmod Record*, 35(1):14–19, 2006.
- [2] Gurmeet Singh Manku and Rajeev Motwani. Approximate frequency counts over data streams. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases*, pages 346–357. Elsevier, 2002.
- [3] Albert Orriols-Puig and Jorge Casillas. Fuzzy knowledge representation study for incremental learning in data streams and classification problems. *Soft Computing*, 15(12):2389–2414, 2011.
- [4] Martin Scholz and Ralf Klinkenberg. Boosting classifiers for drifting concepts. *Intelligent Data Analysis*, 11(1):3–28, 2007.
- [5] Brian Babcock, Shivnath Babu, Mayur Datar, Rajeev Motwani, and Jennifer Widom. Models and issues in data stream systems. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 1–16. ACM, 2002.
- [6] Albert Bifet and Eibe Frank. Sentiment knowledge discovery in twitter streaming data. In *International conference on discovery science*, pages 1–15. Springer, 2010.
- [7] Albert Bifet, Ricard Gavaldà, Geoff Holmes, and Bernhard Pfahringer. *Machine learning for data streams: with practical examples in MOA*. MIT Press, 2018.
- [8] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. *SIGMOD Rec.*, 22(2):207–216, 1993.
- [9] The BigML Team. Association discovery with the bigml dashboard. <https://bigml.com>.
- [10] Sergey Brin, Rajeev Motwani, Jeffrey D Ullman, and Shalom Tsur. Dynamic itemset counting and implication rules for market basket data. *Acm Sigmod Record*, 26(2):255–264, 1997.
- [11] Gregory Piatetsky-Shapiro. Discovery, analysis, and presentation of strong rules. *Knowledge discovery in databases*, pages 229–238, 1991.
- [12] Bing Liu, Wynne Hsu, and Yiming Ma. Mining association rules with multiple minimum supports. In *Proceedings of the fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 337–341. ACM, 1999.

- [13] Rakesh Agrawal, Ramakrishnan Srikant, et al. Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, volume 1215, pages 487–499, 1994.
- [14] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. *ACM sigmod record*, 29(2):1–12, 2000.
- [15] Sajid Mahmood, Muhammad Shahbaz, and Aziz Guergachi. Negative and positive association rules mining from text using frequent and infrequent itemsets. *The Scientific World Journal*, 2014, 2014.
- [16] Sunitha Vanamala, L Padma Sree, and S Durga Bhavani. Rare association rule mining for data stream. In *International Conference on Computing and Communication Technologies*, pages 1–6. IEEE, 2014.
- [17] Laszlo Szathmary, Petko Valtchev, and Amedeo Napoli. Generating rare association rules using the minimal rare itemsets family. *International Journal of Software and Informatics*, 4:219–238, 2010.
- [18] Yun Sing Koh and Nathan Rountree. Finding sporadic rules using apriori-inverse. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 97–106. Springer, 2005.
- [19] N Hoque, B Nath, and DK Bhattacharyya. A new approach on rare association rule mining. *International Journal of Computer Applications*, 53(3):1–6, 2012.
- [20] Alexander Maedche and Steffen Staab. Discovering conceptual relations from text. In *Proceedings of the 14th European Conference on Artificial Intelligence, ECAI’00*, pages 321–325. IOS Press, 2000.
- [21] Vaishali Bhujade and NJ Janwe. Knowledge discovery in text mining technique using association rules extraction. In *2011 International Conference on Computational Intelligence and Communication Networks*, pages 498–502. IEEE, 2011.
- [22] Manasi Kulkarni and Sagar Kulkarni. Knowledge discovery in text mining using association rule extraction. *International Journal of Computer Applications*, 143(12):30–35, 2016.
- [23] Jorge Casillas and Francisco J Martínez-López. Mining uncertain data with multiobjective genetic fuzzy systems to be applied in consumer behaviour modelling. *Expert Systems with Applications*, 36(2):1645–1659, 2009.
- [24] Andreu Sancho-Asensio, Albert Orriols-Puig, and Jorge Casillas. Evolving association streams. *Information Sciences*, 334:250–272, 2016.

- [25] Mohammed J Zaki, Srinivasan Parthasarathy, Mitsunori Ogihara, and Wei Li. Parallel algorithms for discovery of association rules. *Data mining and knowledge discovery*, 1(4):343–373, 1997.
- [26] Jian Pei, Jiawei Han, Runying Mao, et al. Closet: An efficient algorithm for mining frequent closed itemsets. In *ACM SIGMOD workshop on research issues in data mining and knowledge discovery*, volume 4, pages 21–30, 2000.
- [27] Jianyong Wang, Jiawei Han, and Jian Pei. Closet+: Searching for the best strategies for mining frequent closed itemsets. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 236–245. ACM, 2003.
- [28] Mohammed J Zaki and Ching-Jui Hsiao. Charm: An efficient algorithm for closed itemset mining. In *Proceedings of the 2002 SIAM international conference on data mining*, pages 457–473. SIAM, 2002.
- [29] Chris Giannella, Jiawei Han, Jian Pei, Xifeng Yan, and Philip S Yu. Mining frequent patterns in data streams at multiple time granularities. *Next generation data mining*, 212:191–212, 2003.
- [30] Pauray SM Tsai. Mining frequent itemsets in data streams using the weighted sliding window model. *Expert Systems with Applications*, 36(9):11617–11625, 2009.
- [31] Raymond Chi-Wing Wong and Ada Wai-Chee Fu. Mining top-k frequent itemsets from data streams. *Data Mining and Knowledge Discovery*, 13(2):193–217, 2006.
- [32] Mahmood Deypir, Mohammad Hadi Sadreddini, and Sattar Hashemi. Towards a variable size sliding window model for frequent itemset mining over data streams. *Computer & Industrial Engineering*, 63(1):161–172, 2012.
- [33] Yun Chi, Haixun Wang, Philip S Yu, and Richard R Muntz. Moment: Maintaining closed frequent itemsets over a stream sliding window. In *Fourth IEEE International Conference on Data Mining (ICDM'04)*, pages 59–66. IEEE, 2004.
- [34] Hua-Fu Li, Chin-Chuan Ho, and Suh-Yin Lee. Incremental updates of closed frequent itemsets over continuous data streams. *Expert Systems with Applications*, 36(2):2451–2458, 2009.
- [35] Ye-In Chang, Chia-En Li, and Wei-Hau Peng. An efficient subset-lattice algorithm for mining closed frequent itemsets in data streams. In *2012 Conference on Technologies and Applications of Artificial Intelligence*, pages 21–26. IEEE, 2012.

- [36] Show-Jane Yen, Cheng-Wei Wu, Yue-Shi Lee, Vincent S Tseng, and Chaur-Heh Hsieh. A fast algorithm for mining frequent closed itemsets over stream sliding window. In *2011 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011)*, pages 996–1002. IEEE, 2011.
- [37] Guojie Song, Dongqing Yang, Bin Cui, Baihua Zheng, Yunfeng Liu, and Kunqing Xie. Claim: An efficient method for relaxed frequent closed itemsets mining over stream data. In *International Conference on Database Systems for Advanced Applications*, pages 664–675. Springer, 2007.
- [38] James Cheng, Yiping Ke, and Wilfred Ng. Maintaining frequent closed itemsets over a sliding window. *Journal of Intelligent Information Systems*, 31(3):191–215, 2008.
- [39] Massimo Quadrana, Albert Bifet, and Ricard Gavaldà. An efficient closed frequent itemset miner for the moa stream mining system. *AI Communications*, 28(1):143–158, 2015.
- [40] Massimo Quadrana. Methods for frequent pattern mining in data streams within the moa system. *Proyecto final de Carrera. Universitat Politècnica de Catalunya*, 2012.
- [41] J Shane Culpepper and Alistair Moffat. Efficient set intersection for inverted indexing. *ACM Transactions on Information Systems (TOIS)*, 29(1):1, 2010.
- [42] Wilas Chamlertwat, Pattarasinee Bhattacharayya, Tippakorn Rungkasiri, and Choochart Haruechaiyasak. Discovering consumer insight from twitter via sentiment analysis. *Journal of Universal Computer Science*, 18(8):973–992, 2012.
- [43] Akshay Java, Xiaodan Song, Tim Finin, and Belle Tseng. Why we twitter: understanding microblogging usage and communities. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis*, pages 56–65. ACM, 2007.
- [44] Christine Williams and Girish Gulati. What is a social network worth? facebook and vote share in the 2008 presidential primaries. In *Annual Meeting of the American Political Science Association*, volume 54, page 21. Citeseer, 2008.
- [45] David Vilares, Mike Thelwall, and Miguel A Alonso. The megaphone of the people? spanish sentistrength for real-time analysis of political tweets. *Journal of Information Science*, 41(6):799–813, 2015.
- [46] Andrea Ceron, Luigi Curini, and Stefano Iacus. Using social media to forecast electoral results. a meta-analysis. *UNIMI-Research Papers in Economics, Business, and Statistics*, page 62, 2014.

- [47] Jennifer Golbeck and Derek Hansen. Computing political preference among twitter followers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1105–1108. ACM, 2011.
- [48] Michael J Jensen and Nick Anstead. Psephological investigations: Tweets, votes, and unknown unknowns in the republican nomination process. *Policy & Internet*, 5(2):161–182, 2013.
- [49] Hao Wang, Dogan Can, Abe Kazemzadeh, François Bar, and Shrikanth Narayanan. A system for real-time twitter sentiment analysis of 2012 us presidential election cycle. In *Proceedings of the ACL 2012 System Demonstrations*, pages 115–120. Association for Computational Linguistics, 2012.
- [50] Andranik Tumasjan, Timm O Sprenger, Philipp G Sandner, and Isabell M Welpe. Predicting elections with twitter: What 140 characters reveal about political sentiment. In *Fourth International AAAI Conference on Weblogs and Social Media*, 2010.
- [51] Guido Caldarelli, Alessandro Chessa, Fabio Pammolli, Gabriele Pompa, Michelangelo Puliga, Massimo Riccaboni, and Gianni Riotta. A multi-level geographical study of italian political elections from twitter data. *PloS one*, 9(5):e95809, 2014.
- [52] Clay Fink, Nathan Bos, Alexander Perrone, Edwina Liu, and Jonathon Kopecky. Twitter, public opinion, and the 2011 nigerian presidential election. In *2013 International Conference on Social Computing*, pages 311–320. IEEE, 2013.
- [53] Manish Gaurav, Amit Srivastava, Anoop Kumar, and Scott Miller. Leveraging candidate popularity on twitter to predict election outcome. In *Proceedings of the 7th Workshop on Social Network Mining and Analysis*, page 7. ACM, 2013.
- [54] Erik Tjong Kim Sang and Johan Bos. Predicting the 2011 dutch senate election results with twitter. In *Proceedings of the Workshop on Semantic Analysis in Social Media*, pages 53–60, 2012.
- [55] Adam Bermingham and Alan Smeaton. On using twitter to monitor political sentiment and predict election results. In *Proceedings of the Workshop on Sentiment Analysis where AI meets Psychology (SAAIP 2011)*, pages 2–10, 2011.
- [56] Panagiotis T Metaxas, Eni Mustafaraj, and Dani Gayo-Avello. How (not) to predict elections. In *2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing*, pages 165–171. IEEE, 2011.

- [57] Mark Edward Huberty. Multi-cycle forecasting of congressional elections with social media. In *Proceedings of the 2nd Workshop on Politics, Elections and Data*, pages 23–30. ACM, 2013.
- [58] Aibek Makazhanov, Davood Rafiei, and Muhammad Waqar. Predicting political preference of twitter users. *Social Network Analysis and Mining*, 4(1):193, 2014.
- [59] Muhammad Asif Razzaq, Ali Mustafa Qamar, and Hafiz Syed Muhammad Bilal. Prediction and analysis of pakistan election 2013 based on sentiment analysis. In *Proceedings of the 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 700–703. IEEE Press, 2014.
- [60] J Ignacio Criado, Guadalupe Martínez-Fuentes, and Aitor Silván. Twitter en campaña: las elecciones municipales españolas de 2011. *RIPS. Revista de Investigaciones Políticas y Sociológicas*, 12(1):93–113, 2013.
- [61] Javier Borondo, AJ Morales, Juan Carlos Losada, and Rosa M Benito. Characterizing and modeling an electoral campaign in the context of twitter: 2011 spanish presidential election as a case study. *CHAOS: An Interdisciplinary Journal of Nonlinear Science*, 22(2):023138, 2012.
- [62] Mariluz Congosto, M Fernández, and Esteban Moro. Twitter y política: Información, opinión y ¿predicción? *Cuadernos de Comunicación Evoca*, 4:11–15, 2011.
- [63] Luis Deltell, Florencia Claes, and José Miguel Osteso. Predicción de tendencia política por twitter: Elecciones andaluzas 2012. *Ámbitos. Revista Internacional de Comunicación*, 22, 2013.
- [64] Behrooz Bayat. Deep text: Using text analytics to conquer information overload, get real value from social media, and add big (er) text to big data. *The Electronic Library*, 2017.
- [65] Ronen Feldman and Ido Dagan. Knowledge discovery in textual databases (kdt). In *Knowledge Discovery in Databases*, volume 95, pages 112–117, 1995.
- [66] Stephan Bloehdorn, Sebastian Blohm, Philipp Cimiano, Eugenie Giesbrecht, Andreas Hotho, Uta Lösch, Alexander Mädche, Eddie Mönch, Philipp Sorg, Steffen Staab, et al. Combining data-driven and semantic approaches for text mining. In *Foundations for the Web of Information and Services*, pages 115–142. Springer, 2011.
- [67] Andreas Hotho, Andreas Nürnberger, and Gerhard Paaß. A brief survey of text mining. In *LDV Forum*, volume 20, pages 19–62. Citeseer, 2005.

- [68] Chandrasekhar Rangu, Shuvrojit Chatterjee, and Srinivasa Rao Valluru. Text mining approach for product quality enhancement:(improving product quality through machine learning). In *2017 IEEE 7th International Advance Computing Conference (IACC)*, pages 456–460. IEEE, 2017.
- [69] Antonio Moreno and Teófilo Redondo. Text analytics: the convergence of big data and artificial intelligence. *International Journal of Interactive Multimedia and Artificial Intelligence*, 3(6):57–64, 2016.
- [70] Markus Hofmann and Andrew Chisholm. *Text mining and visualization: case studies using open-source tools*, volume 40. CRC Press, 2016.
- [71] Eduardo Campillos García. Análisis sentimental de la polaridad de tweets en tiempo real aplicando algoritmos de clasificación. *Trabajo de Fin de Grado. Universidad Politécnica de Madrid*, 2017.
- [72] Eugenio Martínez-Cámara, M Teresa Martín-Valdivia, L Alfonso Ureña-López, and Ruslan Mitkov. Polarity classification for spanish tweets using the cost corpus. *Journal of Information Science*, 41(3):263–272, 2015.
- [73] Chenghua Lin and Yulan He. Joint sentiment/topic model for sentiment analysis. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, pages 375–384. ACM, 2009.
- [74] Dimitrios Effrosynidis, Symeon Symeonidis, and Avi Arampatzis. A comparison of pre-processing techniques for twitter sentiment analysis. In *International Conference on Theory and Practice of Digital Libraries*, pages 394–406. Springer, 2017.
- [75] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1(12):2009, 2009.
- [76] Efthymios Kouloumpis, Theresa Wilson, and Johanna Moore. Twitter sentiment analysis: The good the bad and the omg! In *Fifth International AAAI Conference on Weblogs and Social Media*, 2011.
- [77] Soo-Min Kim and Eduard Hovy. Determining the sentiment of opinions. In *Proceedings of the 20th international conference on Computational Linguistics*, page 1367. Association for Computational Linguistics, 2004.
- [78] Xiaowen Ding, Bing Liu, and Philip S Yu. A holistic lexicon-based approach to opinion mining. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, pages 231–240. ACM, 2008.

- [79] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM, 2004.
- [80] Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. Lexicon-based methods for sentiment analysis. *Computational Linguistics*, 37(2):267–307, 2011.
- [81] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical Methods in Natural Language Processing- Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.
- [82] Eugenio Martínez Cámará y M. Teresa Martín Valdivia y José M. Perea Ortega y L. Alfonso Ureña López. Técnicas de clasificación de opiniones aplicadas a un corpus en español. *Procesamiento del Lenguaje Natural*, 47(0):163–170, 2011.
- [83] Julian Brooke, Milan Tofiloski, and Maite Taboada. Cross-linguistic sentiment analysis: From english to spanish. In *Proceedings of the International Conference RANLP-2009*, pages 50–54, 2009.
- [84] Julio Villena-Román and Janine García-Morera. Workshop on sentiment analysis at sepln 2013: An overview. In *Taller de Análisis de Sentimientos en la SEPLN / Workshop on Sentiment Analysis at SEPLN*, pages 112–125, 2013.
- [85] Eugenio Martínez-Cámará, M Teresa Martín-Valdivia, M Dolores Molina-González, and L Alfonso Ureña-López. Bilingual experiments on an opinion comparable corpus. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 87–93, 2013.
- [86] Fermín L Cruz, José A Troyano, Beatriz Pontes, and F Javier Ortega. Building layered, multilingual sentiment lexicons at synset and lemma levels. *Expert Systems with Applications*, 41(13):5984–5994, 2014.
- [87] Carlos Henríquez Miranda and Jaime Guzman. A review of sentiment analysis in spanish. *Tecciencia*, 12(22):35–48, 2017.
- [88] Elliot Hofman. senti-py. <https://github.com/aylliote/senti-py>, 2018.
- [89] MeaningCloud. meaningcloud-python. <https://github.com/meaningCloud/meaningcloud-python>, 2018.

- [90] Jonathan Culpeper, Alison Findlay, Beth Cortese, and Mike Thelwall. Measuring emotional temperatures in shakespeare’s drama. *English Text Construction*, 11(1):10–37, 2018.
- [91] Raquel Alvarez, David Garcia, Yamir Moreno, and Frank Schweitzer. Sentiment cascades in the 15m movement. *EPJ Data Science*, 4(1):6, 2015.
- [92] M ThelWall, K Buckley, G Paltoglou, D Cai, and A Kappas. Sentiment in short strength detection informal text (vol 61, pg 2544, 2010). *Journal of the American Society for Information Science and Technology*, 62(2):419–419, 2011.
- [93] Antonio Moreno Ortiz and Chantal Pérez Hernández. Lexicon-based sentiment analysis of twitter messages in spanish. *Procesamiento del Lenguaje Natural*, 50, 2013.
- [94] Javier G. Sogo. apicultur-python. <https://github.com/jgsogo/apicultur-python>, 2015.
- [95] Sonia Ordoñez Salinas, Juan Manuel Pérez Trujillo, and Romario Albeiro Sánchez Montero. Election analysis in colombia and venezuela 2015 through sentiment analysis and twitter. *Sistemas y Telemática*, 14(39):57–70, dec 2016.
- [96] Pollyanna Gonçalves, Matheus Araújo, Fabrício Benevenuto, and Meeyoung Cha. Comparing and combining sentiment analysis methods. In *Proceedings of the first ACM conference on Online social networks*, pages 27–38. ACM, 2013.
- [97] Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *Language Resources and Evaluation Conference*, volume 10, pages 1320–1326, 2010.
- [98] Ravi Parikh and Matin Movassate. Sentiment analysis of user-generated twitter updates using various classification techniques. *CS224N Final Report*, 118, 2009.
- [99] Luciano Barbosa and Junlan Feng. Robust sentiment detection on twitter from biased and noisy data. In *Proceedings of the 23rd international conference on computational linguistics: posters*, pages 36–44. Association for Computational Linguistics, 2010.
- [100] Saif M Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. In *SEM 2013 - 2nd Joint Conference on Lexical and Computational Semantics*, 2013.
- [101] Yasuhide Miura, Shigeyuki Sakaki, Keigo Hattori, and Tomoko Ohkuma. Teamx: A sentiment analyzer with enhanced lexicon mapping and weighting scheme for unbalanced data. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 628–632, 2014.

- [102] Rui Xia, Chengqing Zong, and Shoushan Li. Ensemble of feature sets and classification algorithms for sentiment classification. *Information Sciences*, 181(6):1138–1152, 2011.
- [103] Jonathon Read. Using emoticons to reduce dependency in machine learning techniques for sentiment classification. In *Proceedings of the ACL Student Research Workshop*, pages 43–48. Association for Computational Linguistics, 2005.
- [104] Peter D Turney. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 417–424. Association for Computational Linguistics, 2002.
- [105] Po-Wei Liang and Bi-Ru Dai. Opinion mining on social media data. In *2013 IEEE 14th International Conference on Mobile Data Management*, volume 2, pages 91–96. IEEE, 2013.
- [106] Nathan Aston, Timothy Munson, Jacob Liddle, Garrett Hartshaw, Dane Livingston, and Wei Hu. Sentiment analysis on the social networks using stream algorithms. *Journal of Data Analysis and Information Processing*, 2(02):60, 2014.
- [107] Saif M Mohammad, Parinaz Sobhani, and Svetlana Kiritchenko. Stance and sentiment in tweets. *ACM Transactions on Internet Technology (TOIT)*, 17(3):26, 2017.
- [108] Falcon Dai. python-snowflake. <https://github.com/falcondai/python-snowflake>, 2017.
- [109] Matthew Honnibal and Ines Montani. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. *To appear*, 7, 2017.
- [110] geduldig. Twitterapi. <https://github.com/geduldig/TwitterAPI>, 2019.
- [111] @TwitterDev. Python twitter search api. <https://github.com/twitterdev/search-tweets-python>, 2019.