



UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA
Instituto Universitario de Microelectrónica Aplicada
Sistemas de información y Comunicaciones

Máster en Tecnologías de Telecomunicación



Trabajo Fin de Máster

Clasificador basado en Análisis de Sentimiento

Autor: Leandro Aarón López Rodríguez
Tutor(es): José María Quinteiro González
Pablo Hernández Morera
Fecha: Septiembre 2011



t +34 928 451 086 | iuma@iuma.ulpgc.es
f +34 928 451 083 | www.iuma.ulpgc.es

Campus Universitario de Tafira
35017 Las Palmas de Gran Canaria



UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA
Instituto Universitario de Microelectrónica Aplicada
Sistemas de información y Comunicaciones

Máster en Tecnologías de Telecomunicación



Trabajo Fin de Máster Clasificador basado en Análisis de Sentimiento

HOJA DE FIRMAS

Alumno/a: Leandro Aarón López Rodríguez Fdo.:

Tutor/a: José María Quinteiro González Fdo.:

Tutor/a: Pablo Hernández Morera Fdo.:

Fecha: Septiembre 2011



t +34 928 451 086 iuma@iuma.ulpgc.es
f +34 928 451 083 www.iuma.ulpgc.es

Campus Universitario de Tafira
35017 Las Palmas de Gran Canaria



UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA
Instituto Universitario de Microelectrónica Aplicada
Sistemas de información y Comunicaciones

Máster en Tecnologías de Telecomunicación



Trabajo Fin de Máster

Clasificador basado en Análisis de Sentimiento

HOJA DE EVALUACIÓN

Calificación:

Presidente Francisco José Guerra Fdo.:

Secretario Pablo Hernández Morera Fdo.:

Vocal Fernando de la Puente Fdo.:

Fecha: Septiembre 2011



t +34 928 451 086 iuma@iuma.ulpgc.es
f +34 928 451 083 www.iuma.ulpgc.es

Campus Universitario de Tafira
35017 Las Palmas de Gran Canaria

1.Índice

1. Índice.....	7
2. Resumen.....	8
3. Abstract	10
4. Objetivos	12
5. Introducción	13
6. Diseño del Clasificador	16
6.1. Carga de Datos.....	16
6.2. Procesado de Datos	17
6.2.1. Stemmer	17
6.2.2. Stopword	19
6.2.3. Análisis Sintáctico	19
6.2.4. Bolsas de Palabras (Bag of Words)	23
6.2.5. N-Gramas.....	24
6.2.6. SentiWordNet.....	24
6.3. Entrenamiento.....	27
6.3.1. WEKA.....	27
7. Aproximaciones	28
7.1. Extracción de Atributos	28
7.2. Bolsas de Palabras con N-Gramas, Stemmer y Stopwords	29
8. Evaluación del Clasificador	34
8.1. Datos.....	34
8.1.1. Amazon Dataset	35
8.2. Algoritmos de Machine Learning	35
8.3. Resultados	36
8.3.1. Extracción de características	36
8.3.2. Bag of Word.....	37
9. Creación del Prototipo	52
10. Conclusiones y Trabajo Futuro	55
11. Referencias.....	56

2. Resumen

Las opiniones externas siempre han sido un elemento para la toma de decisiones, antes del boom de los blogs y de las redes sociales consultábamos a los amigos, mirábamos en revistas especializadas o preguntábamos al vendedor antes de comprarnos un coche. Ahora tenemos muchas más opciones. Los blogs, páginas de opinión y redes sociales han elevado el número de opciones de consulta, generando una gran cantidad de información generalmente desestructurada, y ahí es donde entra el análisis de sentimiento o minería de opinión.

El análisis de sentimiento trata de obtener la polaridad subyacente en texto libre, entendiendo por polaridad subyacente lo positivo o negativo que puede ser un texto. Este proceso también es conocido como minería de opinión ya que puede considerarse una rama de la minería de datos por la utilización que hace de sus herramientas y técnicas.

En este Trabajo de Fin de Master vamos a considerar la polaridad de un texto para clasificarlo como positivo o negativo. En trabajos posteriores podremos evolucionar hacia conceptos más complejos estableciendo rangos en el sistema de clasificación, un ejemplo de este tipo de sistemas pueden ser las valoraciones de una a cinco estrellas, o el establecimiento de porcentajes de polaridad.

Para crear los modelos a estudiar y poder compararlos se han seleccionado dos aproximaciones. En primer lugar vamos a considerar la aproximación más sencilla y sobre todo la más utilizada en la literatura científica, las bolsas de palabras y en segundo lugar utilizaremos la extracción de atributos.

En la creación de las *bolsas de palabras* de tamaño k , que de aquí en adelante las llamaremos n -gramas, existen distintas técnicas que se pueden utilizar como complemento: dos ejemplo clásicos son los *stemmer* (lematizadores) y las *stopwords* (palabras stop). El *stemming* de una palabra consiste en extraer la raíz de las palabras. Las *stopwords* son palabras que sirven en la lectura para unir frases pero cuyo valor semántico es despreciable.

La utilización de distintos valores de k , y la posibilidad de utilizar *stemmer* y/o *stopwords* nos otorga tres grados de libertad que nos permiten realizar una serie de experimentos en busca de la mejor combinación posible.

La segunda aproximación, la *extracción de atributos*, es más compleja en su diseño porque utiliza herramientas provenientes del análisis de texto para las

labores de preproceso de datos. Para el preproceso, primero dividiremos el texto en frases que individualmente serán analizadas sintácticamente. Posteriormente utilizaremos una herramienta de cálculo de polaridad que utiliza una palabra y su valor sintáctico para asignar la polaridad.

Una vez tenemos todo el texto marcado sintácticamente y con polaridades asignadas extraemos las características seleccionadas, en este caso obteniendo 21 atributos: frecuencia de nombre, verbos, adjetivos y adverbios, polaridad de nombres, verbos, adjetivos y adverbios, frecuencia de palabras positivas y negativas, número de frases, interrogantes y exclamaciones y número de negaciones.

Tanto para la primera aproximación como para la segunda se ha procedido a realizar el aprendizaje automático utilizando diferentes algoritmos. El resultado obtenido ha sido que la primera aproximación es mejor que la segunda incluso en sus peores combinaciones.

3. Abstract

The external opinions have always been a factor when taking a decision. Before the boom of blogs and social networks, we used to ask friends for their opinion, read magazines or ask the shop assistant before buying a car. Now we have more options. Blogs, opinion pages and social networks have increased the number of query options, generating a large amount of information, generally unstructured. And that is where the sentiment analysis and opinion mining.

Sentiment analysis is to obtain the underlying polarity of a free text. We understand as underlying polarity how positive or negative text can be. This process is also known as opinion mining and can be considered a branch of data mining by the use of its tools and techniques.

This Master Final Project considers the polarity of a text in order to classify it as positive or negative. In subsequent work, this classification can evolve into more complex concepts establishing ranges in the classification system. An example of such systems may be in stars, between one or five stars, or the establishment of polarity rates-

In order to create models and to compare them, two approaches have been selected. Firstly let us consider the simplest approach and the most common in the literature field, the bags of words; and secondly the feature extraction. There are different complementary techniques used when creating word bags K-size, from now on n-gramas. Two classic examples are the stemmer and stopwords. Stemming a word consists of removing the root words. The stopwords are words that are use to link sentences but whose semantic value is not significant.

Using different values of K, and the possibility of using stemmer and / or stopwords, gives us three degrees of freedom that allow us experiment to find the best possible combination.

The second approach, the extraction of feeatures, is more complex, because it uses tools from text analysis for data preprocessing tasks. In the preprocess phase, first the text is divided into individual sentences which are syntactically analyzed. Then we use a tool to calculate the polarity. It uses a word and its syntactic value to assign the polarity.

Once we have all the text syntactically marked and with its polarity assigned we extract the selected characteristic; in this case we obtain 21 attributes: frequency of nouns, verbs, adjectives and adverbs, polarity of nouns, verbs, adjectives and

adverbs, frequency of positive and negative words , number of sentences, questions and exclamations, and number of denials.

For both, the first approximation and the second, after the data preprocessing we have have done the automatic learning using different algorithms. The result showed that the first approach is better than the second even in the worst combinations.

4. Objetivos

Este Trabajo de Fin de Master pretende ser el punto de partida de una nueva vía de investigación, el análisis de sentimientos o minería de opinión. El análisis de sentimientos utiliza técnicas y herramientas de la minería de datos para características de texto libre.

En este Trabajo de Fin de Master se pretenden conseguir tres objetivos principales: crear una base de conocimientos, generar herramientas que agilicen trabajos posteriores y crear un prototipo.

La base del conocimiento se creará a partir de los datos obtenidos mediante la utilización de varias técnicas de clasificación, concretamente se utilizará Naive Bayesiana, Bayes multinomial, J48 y regresión logística simple.

Las herramientas generadas deberán tener una arquitectura consistente, ágil y flexible, que permita la realización de nuevos experimentos y la comparación de resultados de forma sencilla.

Además las librerías o aplicaciones externas que se utilicen deben ser integradas bajo una misma interfaz de forma que la inclusión de nuevas técnicas o la selección de las existentes resulte sencilla y eficiente.

Por ultimo tendremos que crear un prototipo que será utilizado en el proyecto RAUDOS 2¹, que utilizará el mejor de los sistemas de clasificación que sean propuestos y al cual se podrá acceder mediante un servicio web.

¹ RAUDOS 2: Red Interactiva Multiplataforma de Distribución de Contenidos Multimedia (TSI-020302-2010-67))

5.Introducción

Conocer el pensamiento de los demás siempre ha sido uno de los principales elementos a tener en cuenta cuando vamos a tomar una decisión, quién no ha preguntado a un amigo por una película, un restaurante o incluso a la hora de comprarse un coche. Con el aumento de los blogs, sistemas de valoración de las webs y las redes sociales, la cantidad de opiniones que una persona puede obtener se ha multiplicado y el número de personas que usa estos medios también, tal y como podemos observar en los resultados obtenidos de dos encuestas realizadas sobre más de 2000 personas [1,2]:

- El 81% de los usuarios de internet ha consultado al menos una vez por las opiniones de un producto
- El 20% lo hace habitualmente
- Entre el 73% y el 87% indica que las opiniones tienen gran influencia en su decisión
- Entre el 20% y el 99% pagaría más por un producto con 5 estrellas que por uno con 4 (dependiendo del tipo de producto)
- El 32% ha valorado un producto y el 30% ha escrito una opinión

De una encuesta realizada a 2.500 norteamericanos durante las elecciones de 2006 en Estados Unidos se puede observar que los usuarios ahora no solo consultan sino también opinan[3]:

- 28% buscaba opiniones entre miembros de su misma comunidad
- 34% buscaba opiniones entre personas externas a su comunidad
- 28% visitaba sitios con su mismo signo político
- 29% visitaba sitios de distinto signo político
- 8% publicaba su propia opinión

La cantidad de datos que esto genera hace necesaria la creación de sistemas de clasificación automáticos o semiautomáticos que permita convertir esta información en conocimiento.

El área de Análisis de Sentimientos y de la Minería de opinión ha disfrutado recientemente de un gran auge en la actividad de investigación aunque ha habido interés continuo desde 1979 desde que se publicó la tesis doctoral referenciada en [10]. Los trabajos posteriores se centraron principalmente en la interpretación de la metáfora, la narrativa, el punto de vista, el afecto y otras áreas relacionadas [11,12,13,14,15,16,17,18,19].

El año 2001 parece marcar el comienzo de la divulgación de los problemas y oportunidades de investigación en el análisis de sentimientos y de la minería de opinión[20,21,22,23,24,25,26,27,28,29,30,31,32].

Los factores principales detrás de este interés son:

- el aumento de los métodos de aprendizaje automático en el procesamiento de lenguaje natural y la recuperación de información
- la disponibilidad de bases de datos gracias a las webs de opinión
- el interés intelectual y comercial

El trabajo previo de análisis de sentimientos se centraba principalmente en determinar los términos que denotan subjetividad o términos que denotan la orientación positiva o negativa. Para esto primero es necesario encontrar palabras relevantes, frases o patrones que puedan ser utilizadas para expresar la subjetividad y determinar la polaridad.

Para determinar la orientación de palabras existen diferentes técnicas, como por ejemplo determinar el sentido de la palabra en la frase. Este aspecto se consigue mediante la categoría gramatical y la desambiguación[5].

Por otro lado, el análisis estadístico permite obtener información a partir de la correlación de determinadas palabras que tienden a aparecer juntas[8] y a partir de lo que se ha dado en llamar distribución de similitud, es decir, palabras con significados similares tienden a parecer en contextos similares [9].

Por último, si consideramos que los sinónimos tienen la misma subjetividad y que los antónimos tienen polaridades opuestas[33,34], podemos generar relaciones léxicas mediante la utilización de Wordnet[36].

Como muestra del interés que suscitan estas tecnologías en la literatura científica, se enumeran algunos de los artículos más referenciados en el campo del Análisis de Sentimiento:

- Opinión en Noticias [37,38,39,40,41,42]

- Opiniones sobre objetos [43,44,45,46,47,48]
- Perspectiva [49]
- pros y contras en revisiones [50]
- Estado de ánimo en los blogs [51,52,53]
- Felicidad [54]
- Política[55]
- Asignar reviews a las películas [56,58,60]
- Predecir los resultados de las elecciones[57]

Antes de pasar a la implementación nos gustaría hacer una pequeña digresión para hablar sobre los términos asociados a esta rama de estudio. El término minería de opinión apareció en un artículo [22] que fue publicado en la conferencia de la WWW (World Wide Web) de 2003, esta publicación puede explicar la popularidad del término en comunidades fuertemente asociadas con la búsqueda en la Web o de recuperación de información.

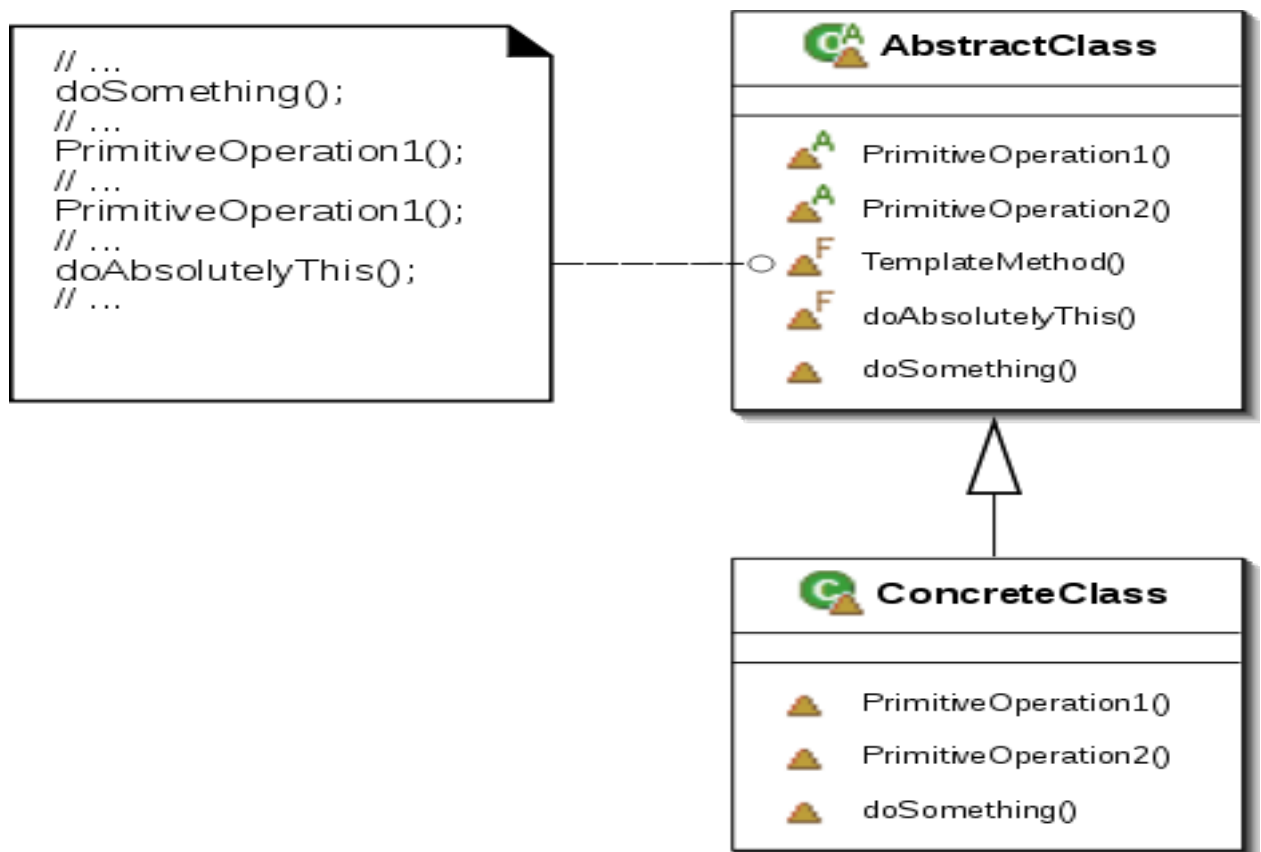
Según este artículo la herramienta ideal de minería de opinión debe procesar los resultados de una búsqueda realizada sobre un ítem dado, generando una lista de atributos y agregando opiniones acerca del ítem. Muchos de los artículos posteriores siguen esta descripción, sin embargo, el término ha sido ampliado a una evaluación más general de clasificación de textos[73].

El término análisis de sentimientos ha sufrido un proceso parecido, apareció en 2001 [74,75] en artículos relacionados con el marketing. En 2002 [30, 27] se publicaron artículos en *Association for Computational Linguistics* (ACL) y la *Empirical Methods in Natural Language* (EMNLP). El lugar de la publicación de estos artículos explica la popularidad del término análisis de sentimientos entre las comunidades de procesamiento del lenguaje. Nosotros nos situamos más cerca del procesamiento del lenguaje por los trabajos que se han realizado en el grupo anteriormente, por tanto se ha decidido utilizar el término análisis de sentimiento, aunque se podría utilizar minería de opinión ya que ambos hacen referencia al mismo campo de estudio.

6. Diseño del Clasificador

Uno de los objetivos de diseño del clasificador es la creación de una arquitectura que apoyándose en patrones de diseño sea fácilmente ampliable y que permita la rápida creación de experimentos.

La arquitectura se basa principalmente en el patrón plantilla (Template Method). Este patrón de diseño define una estructura de herencia de manera que la superclase sirve de plantilla para los métodos en las subclases, permitiendo a las subclases redefinir ciertos pasos de un algoritmo sin cambiar su estructura.



Este proceso se ha dividido en 3 etapas, las cuales coinciden con las etapas clásicas del aprendizaje automático: carga de datos, procesamiento de datos y entrenamiento.

6.1. Carga de Datos

Los conjuntos de datos existentes tienen distintos formatos y en general no se adaptan a un estándar. Suelen utilizar sistemas similares a xml, pero con estructura propia, provocando que los adaptadores genéricos no funcionen. Este problema nos ha llevado a tener que crear una serie de clases que lean los

conjuntos de datos y los transforma a formato csv (separado por comas). Se ha optado por el formato csv porque es el formato que usan las herramientas de minería de datos que utilizamos.

Este archivo csv tiene actualmente dos campos: *review* y *polaridad*. El campo *review* es el texto con la opinión y no debe ser nunca nulo. El campo *polaridad* se identifica con *pos* , *neg* o *?* en caso de que sea desconocido.

En el campo *polaridad* se ha optado por utilizar valores de texto en vez de valores numéricos, porque los algoritmos de clasificación a usar trabajan sobre valores nominales por lo que los valores numéricos tendrían que ser convertidos a valores nominales previamente.

Aunque el formato de los archivos .csv propuesto es muy sencillo, dos campos, es fácilmente ampliable y sus ampliaciones no necesitan de cambios en el código que en ese momento esté en funcionamiento.

6.2. Procesado de Datos

En el procesado de datos se han integrado distintas técnicas. Para realizar una integración ampliable se han introducido interfaces abstractas que agrupan técnicas del mismo tipo como por ejemplo los analizadores sintácticos. Al no tener todas las técnicas un conjunto de APIS, ha sido necesario crear puentes (patrón de diseño *bridge*), para incorporarlos a la arquitectura. De esta forma se facilita el mantenimiento porque ante cualquier cambio en las técnicas solo se ha de cambiar la implementación del puente.

A continuación se detallan las técnicas y las necesidades que han ido surgiendo tanto en su incorporación como en su implementación.

6.2.1. Stemmer

El *stemming* consiste en extraer la raíz de una palabra, por ejemplo tanto *bibliotecas* como *bibliotecario*, poseen como raíz *bibliotec*. Con este enfoque las palabras de raíz común pueden considerarse como un solo término. Este proceso es necesario en las búsquedas y clasificaciones de textos para no penalizar la frecuencia de las palabras.

Los algoritmos de stemming o lematización más conocidos son Lovins (1968) y Porter (1980)[77]. Ambos siguen un proceso similar y se centran en eliminar los sufijos de las palabras de forma iterativa, sin conocer a priori todas las posibles

terminaciones, es decir no utilizan un diccionario como base. No obstante los resultados entre los dos algoritmos de lematización varían.

La raíz lematizada no tiene el mismo sentido que la raíz lingüística. Si volvemos al ejemplo anterior, el lema de biblioteca sería *bibliotec*, en cambio la raíz lingüística del latín sería *bibliothēca*. Además su sentido tampoco es compartido ya que la lematización tiene como objetivo mejorar la frecuencia y no obtener el origen de la palabra.

Los algoritmos de stemming o lematización podrían extraer los prefijos y los sufijos, pero los métodos mencionados anteriormente solo eliminan los sufijos. Diferenciar un prefijo de una raíz es una labor compleja como podemos observar en las palabras *incorrecto* e *introducción*. *Indispensable* tiene como prefijo *in*, en cambio si eliminamos el sufijo *in-* de introducción se extraería el sufijo de forma incorrecta. Eliminando el sentido de la palabra y dando por tanto información errónea. Gracias al ejemplo anterior también podemos observar que los métodos de lematización son dependientes del idioma.

La eliminación de sufijos también presenta el problema de la eliminación del significado de la palabra, pero en menor medida, ya que en los sufijos podemos encontrar plurales, declinaciones, tiempos verbales, etc..

Para llevar a cabo una eliminación de sufijos correcta tenemos que identificar los sufijos frecuentes, identificar las ocurrencias de sufijos conjuntas, establecer el orden de los sufijos.

Una vez conocido su funcionamiento genérico vamos conocer los dos algoritmos de lematización que hemos utilizado.

1.1.1.1. Porter

El algoritmo de Porter lee los caracteres, divide las palabras, elimina los números y finalmente obtiene el lema.

El lematizador se compone de conjuntos de reglas por los cuales van pasando las palabras. Cada conjunto esta formado por n reglas y cada regla está constituida por un identificador, un sufijo, un texto de reemplazo, un tamaño del sufijo, un tamaño del texto reemplazado, un tamaño mínimo de la raíz después del reemplazo y una función de validación. Este proceso es iterativo, y se ejecuta hasta que no se active ninguna regla

Un ejemplo de regla es el siguiente:

*ied+ ies**

replace by i if preceded by more than one letter, otherwise by ie (so ties -> tie, cries -> cri)

Actualmente se encuentra en estado de *frozen* (congelado), invitándonos su propio autor a utilizar algoritmos derivados y que lo mejoran como Snowball. En la página oficial podemos encontrar librerías en C y Java, siendo esta última la que se ha elegido. Al ser Snowball tan popular se ha producido una comunidad de usuarios que han extendido Snowball a otros idiomas como el español.

1.1.1.2. Lovins

El algoritmo de Lovins sólo utiliza una pasada para eliminar los sufijos. Por esta razón necesita tantas reglas como posibilidades de sufijos existan, de las que aplica aquella que extrae el sufijo mayor tomando como límite un lema de 3 caracteres, independientemente del idioma. Fue el primer lematizador y en su momento una innovación importante.

Existe una variación llamada Iterator LovinsStemmer, cuyo funcionamiento es más semejante a Porter.

6.2.2. Stopword

Los ficheros de stopwords son ficheros que contienen términos que pueden eliminarse de un texto ya que no aportan información importante. Es un proceso muy utilizado en los clasificadores de textos, pero de discutida aceptación dentro de la minería de opinión ya que esa información eliminada puede llegar a ser importante. Por ejemplo las negaciones suelen ser eliminadas en las listas de stopwords clásicas pero en el caso del análisis de sentimientos aportan una información muy valiosa.

No existe un fichero estándar de stopwords y es mejor generarlos de acuerdo con las necesidades de la aplicación. Su estructura clásica es una palabra por línea. Y su implementación clásica es una búsqueda en el fichero y, si se localiza, la eliminación es directa.

6.2.3. Análisis Sintáctico

El análisis sintáctico descompone el texto en categorías gramaticales que son dependientes del idioma. Existen numerosas APIs y programas que generan análisis sintáctico. Dada su facilidad de uso y la amplia gama de idiomas se ha seleccionado

Tree-Tagger para esta tarea. Se ha escogido su conjunto de etiquetas sintácticas para crear las tablas que todos los analizadores sintácticos deben proporcionar como salida. Por tanto la introducción de un nuevo analizador llevaría consigo implícita la conversión de su salida estándar a las tablas de categorías (adjetivos, adverbios, verbos, sustantivos y conjunciones, tal y como se indica más adelante). Aunque pueda parecer trabajo extra, esta conversión de la salida propia de los analizadores a las tablas propuestas permite cambiar a los clasificadores de analizador sintáctico incluso de forma dinámica.

Adjetivos (JJ): Además de los adjetivos propiamente dichos, incluye composiciones de palabras con guión.

Ejemplos: happy-go-lucky

 One-of-kind

 Run-of-the-mill

Elemento Sintáctico	Tag
Adjetivo Comparativo	JJR
Adjetivo Superlativo	JJS
Número ordinal	JJ

Adverbios (RB): En esta categoría se incluyen la mayoría de palabras que finalizan en *-ly* , palabras que denotan cantidad o negación.

Elemento Sintáctico	Tag
Adverbio Comparativo	RBR
Adverbio Superlativo	RBS
Negación	RB

Determinantes (DT)

Elemento Sintáctico	Tag
Artículo	DT
Predeterminante	PDT

Nombres (NN)

Elemento Sintáctico	Tag
Nombre Plural	NNS

Pronombre Personal (PP)

Elemento Sintáctico	Tag
Pronombre Posesivo	PP\$
Nombre Propio Plural	NPS
Nombre Propio Singular	NP

Verbos (VB)

Elemento Sintáctico	Tag
Pasado Participio	VBN
Presente Participio, Gerundio	VBG
Pasado	VBD
Presente	VBP
Presente 3ª Persona Singular	VBZ

Pronombre (WP)

Elemento Sintáctico	Tag
Wh-Pronombre posesivo	WP\$
Wh-Adverbio	WRB
Wh-Determinante	WDT

--	--

Otros

Elemento Sintáctico	Tag
Conjunciones	CC
Preposición	IN
Palabra extranjera	FW
Intersección	UH
Lista de elementos	LS
Verbos Modales	MD
Partícula	RP
Símbolo	SYM
Existencial	EX
To	TO
Número Cardinales	CD
Conjunciones	CD
Posesivos	POS

Todos los Analizadores Sintácticos heredan de la clase `AnalizadorSintáctico`. Esta clase solo tiene una función `execute()`, que permite que el analizador descomponga la frase introducida en tokens sintácticos, cada uno de los cuales se compone de tres partes lexema, lema y forma gramatical. A continuación vamos a ver con detalle como funciona Tree Tagger.

1.1.1.3. Tree Tagger

Es un analizador sintáctico que utiliza marcadores probabilísticos para determinar la información antes indicada de los tokens. Para ello se emplea un árbol de decisión. Se ha demostrado que el etiquetado empleando un árbol de decisión tiene mayor precisión que un etiquetado tri-grama estándar (conjunto de palabra, raíz y categoría sintáctica). Además se ha demostrado la robustez del tree-tagger en lo que respecta al tamaño del conjunto de entrenamiento. La

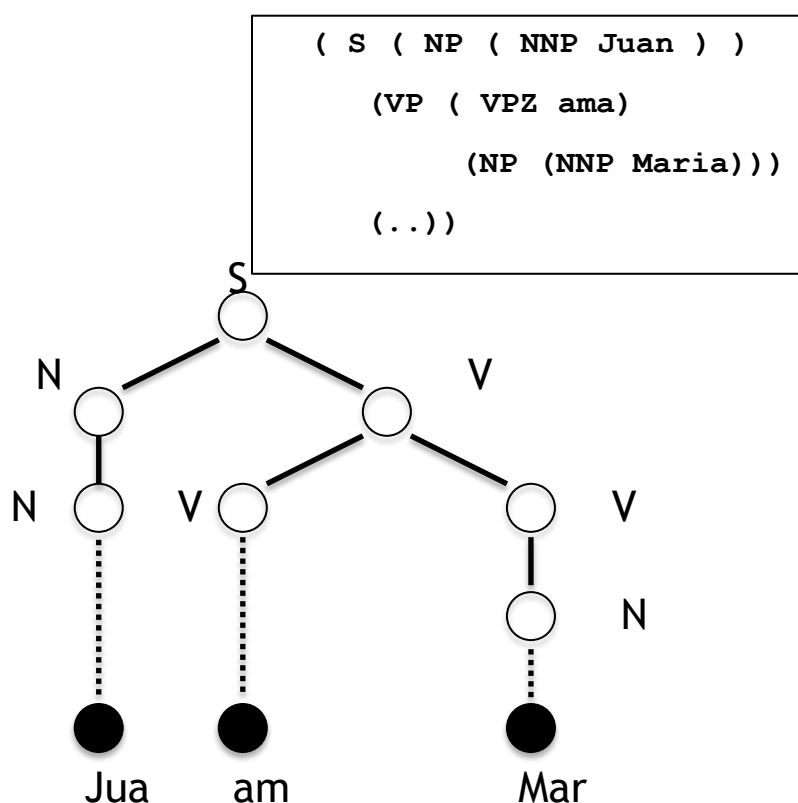
utilización de un conjunto de entrenamiento pequeño no supone una disminución importante en la exactitud del resultado.

Es posible mejorar la exactitud del etiquetador si se emplea un cuatri-grama dentro del contexto en vez de utilizar un tri-grama. Gracias a la eficacia de la aplicación, el etiquetador es capaz de etiquetar hasta 10.000 muestras por segundo.

El tree-tagger es adaptable a otros idiomas si se cuenta con un léxico y un conjunto de entrenamiento etiquetado manualmente.

Podemos ver un ejemplo de la construcción del árbol con la siguiente frase:

“Juan ama María”



Para su integración en el código se ha necesitado crear un puente ya que solo se permite su uso mediante línea de comandos. En este caso se ha añadido la versión inglesa y la española, aunque solo utilizaremos la inglesa en este Trabajo de Fin de Master.

6.2.4. Bolsas de Palabras (Bag of Words)

Las bolsas de palabras son una aproximación a la clasificación de textos. Su uso es muy sencillo y sus resultados muy buenos por lo que es una de las principales aproximaciones al análisis de sentimientos.

Esta técnica consiste en separar tokens de un texto, rompiendo cualquier orden o relación gramatical, y después aplicar algún algoritmo de clasificación como la inferencia Bayesiana, para obtener las predicciones.

6.2.5. N-Gramas

Los N-gramas son subsecuencias de N elementos. Las subsecuencias de dos elementos se denominan bigramas o digramas, y las de tres elementos trigramas, de cuatro en adelante se denominan N-Gramas o modelos de Markov de orden ($N-1$).

En el caso concreto del análisis de sentimiento los N-Gramas son conjuntos de palabras divididas por delimitadores seleccionados. Por ejemplo en la frase *Los amigos de Pedro van al cine*, podemos generar unigramas:

[Los, amigos, de, Pedro, van, al, cine]

bigramas:

[Los amigos, amigos de, de Pedro, Pedro van, van al, al cine]

trigramas:

[los amigos de, amigos de Pedro, de Pedro van, Pedro van al, van al cine]

O también podemos combinarlos

Hasta $N=3$

[Los, amigos, de, Pedro, van, al, cine, Los amigos, amigos de, de Pedro, Pedro van, van al, al cine, los amigos de, amigos de Pedro, de Pedro van, Pedro van al, van al cine]

En este ejemplo podemos observar que uno de los problemas de los N-gramas es la generación de muchas subsecuencias con poca información o sin ella. Es el caso de *Los, al*. También podemos observar una de las grandes ventajas de los N-gramas en la expresión *amigos de Pedro*, ya que es una expresión rica en información relevante.

Se han utilizado los filtros incluidos en weka herramienta en la que hablaremos en el apartado 6.3.1, por sencillez, aunque es posible que en versiones posteriores se creen filtros propios.

6.2.6. SentiWordNet

Es un recurso léxico creado para apoyar la clasificación de sentimientos y las aplicaciones de minería de opinión.

SentiWordNet[35] se trata de un recurso empleado por más de 300 grupos con gran variedad de proyectos de investigación.

Esta herramienta se encarga de clasificar las anotaciones de WordNet según el grado de positividad, negatividad o neutralidad. En el algoritmo empleado para clasificar automáticamente WordNet se incluye un paso previo de aprendizaje supervisado.

Se ha llevado a cabo una comparación entre SentiWordNet 3.0 frente a un fragmento de WordNet 3.0 clasificado manualmente, gracias a esta comparación los creadores llegaron a la conclusión de que SentiWordNet 3.0 supone una mejora de un 20% con respecto a SentiWordNet 1.0.

Por ejemplo dada la palabra *Estimable* con SentiWordNet 1.0 se pueden obtener diferentes resultados según la frase en la que se encuentre la palabra:

“Puede ser calculada o estimada” -> Subjetivo=1.0, Positivo=0.0, Negativo=0.0

“Merecen respeto y estima” -> Subjetivo=0.25, Positivo= 0.75, Negativo 0.0

Se debe indicar que cada uno de los tres rangos debe estar en el intervalo [0.0, 1.0] y sumar entre los tres 1.0 para cada palabra.

SentiWordNet cuenta con 4 versiones de las cuales tan sólo dos han visto la luz para fines de investigación:

1. SentiWordNet 1.0: presentado en 2006 por Esuli y Sebastiani y puesto a disposición del público para fines de investigación.

2. SentiWordNet 1.1: en 2007 se discute en un informe técnico pero nunca llega a publicarse.

3. SentiWordNet 2.0: se discute en la tesis doctoral de Esuli en 2008 y tampoco llega a ser publicado.

4. SentiWordNet 3.0: versión actual de SentiWordNet puesto a disposición del público para su uso en el ámbito de la investigación.

A continuación indicamos cual es el proceso que lleva a cabo SentiWordNet 3.0 para llevar a cabo la marcación automática:

1. Etapa de aprendizaje semi-supervisado: Esta etapa es similar a la empleada en SentiWordNet 1.0, se divide a su vez en 4 pasos:

a. Selección de datos (1): Se emplean dos pequeños conjuntos de datos (uno integrado por un conjunto de 7 términos paradigmáticamente positivos y otro con 7 términos paradigmáticamente negativos), creándose una serie de relaciones binarias pudiendo conservar o modificar la propiedad negativa o positiva, es decir, se pueden conectar dos sinónimos de la misma polaridad o de polaridad opuesta.

b. Entrenamiento de clasificador (2): En este paso se emplean los dos conjuntos generados en la etapa anterior y se agrega un tercer conjunto con los sinónimos que tienen la propiedad Neutra como conjunto de entrenamiento para llevar a cabo el desarrollo de un clasificador ternario. Se da un valor al conjunto de sinónimos en vez de al sinónimo en sí mismo. Por lo tanto podemos decir que SentiWordNet es una “bolsa de conjuntos de sinónimos”.

c. Clasificación de sinónimos (3): Se clasifican los sinónimos como pertenecientes a cualquiera de las clases (positivas, negativas o neutras) a través del clasificador generado en el paso anterior.

d. Combinación de clasificadores (4): En el paso (2) se pueden emplear diferentes valores para los parámetros del radio, nivel de profundidad, así como diferentes tecnologías de aprendizaje supervisado, por ello la clasificación es más precisa si en vez de emplear un clasificador ternario se emplea un conjunto de clasificadores ternarios siendo cada uno generado a partir de una combinación de las diferentes opciones para los dos parámetros anteriormente indicados (radio y tecnología de aprendizaje).

Existen por lo tanto un conjunto de 8 clasificadores como resultado de la combinación de 4 opciones de radio ($K \in \{0,2,4,6\}$) y dos tecnologías diferentes de aprendizaje (Rocchio y SVMs). En este último paso la polaridad final es generada como un promedio de todos los clasificadores empleados.

2. Paso aleatorio: Este paso radica en considerar WordNet como un grafo sobre el que se ejecuta un proceso iterativo en el que los valores positivo, negativo y neutro son asignados utilizando la información del paso anterior, este paso aleatorio finaliza cuando el proceso iterativo converge.

SentiWordNet solo ofrece un archivo de texto con todas las palabras incluidas en líneas, así que se ha necesitado crear un wrapper que permita de forma sencilla su acceso

6.3. Entrenamiento

Para esta etapa se ha optado por utilizar los algoritmos de WEKA, por sus múltiples virtudes que veremos a continuación.

6.3.1. WEKA

Weka[69] es una colección de algoritmos de aprendizaje automático (*machine learning*) en tareas de minería de datos. Y que pueden ser ejecutados desde un GUI (Graphical User Interface) de usuario, desde línea de comandos o desde código java directamente.

Weka ha sido desarrollado por la Universidad de Waikato (Nueva Zelanda) y existen numerosas publicaciones que la utilizan como herramienta principal. Además de ser incluida en números productos derivados de tesis doctorales de la misma universidad como Maui Indexer, WikiMiner o KEA o incluso en otros productos como NLP.

Weka además de los algoritmos de machine learning incluye una gran cantidad de filtros que nos permiten modificar los atributos de los datos de entrada.

7. Aproximaciones

Para la creación del clasificador se ha optado por dos aproximaciones diferentes. Por un lado vamos a realizar una extracción de atributos y por otro se va a utilizar una bolsa de palabras (Bag of Word, BOW).

7.1. Extracción de Atributos

En esta aproximación se ha decidido obtener una serie de características del texto partiendo de lo propuesto por Kertin Denecke[71]. En ese artículo se proponían 13 atributos:

- Frecuencia de nombre, verbos y adjetivos (3 atributos)
- Polaridad de nombres, verbos y adjetivos (9 atributos)
- Frecuencia de palabras positivas y negativas (2 atributos)
- Número de frases, interrogantes y exclamaciones (3 atributos)

Para mejorar los resultados se han añadido nuevos atributos:

- Frecuencia de adverbios (1 atributo)
- Polaridad de adverbios (3 atributos)
- Número de Negaciones (1 atributos)

Con lo cuál obtenemos 21 atributos en total.

Para determinar la polaridad de las palabras vamos a utilizar SentiWordNet y para desambiguar se ha utilizado el analizador sintáctico de Tree-Tagger. Como se ha mostrado anteriormente, SentiWordNet posee múltiples entradas para una misma palabra dentro de una misma categoría sintáctica. Por ejemplo: *good* como adverbio tiene dos entradas, la primera que indica calidad o habilidad *the baby can walk pretty good*. (positivo:0.375, subjetivo:0.625, negativo:0) y otra entrada que indica absoluto o completo *we beat him good* (positivo:negativo:0 y subjetivo:1). En este caso se aplicará la media y el resultado de *good* como adverbio sería:

- Positivo = 0.1875
- Negativo=0
- Subjetividad=0.8125

7.2. Bolsas de Palabras con N-Gramas, Stemmer y Stopwords

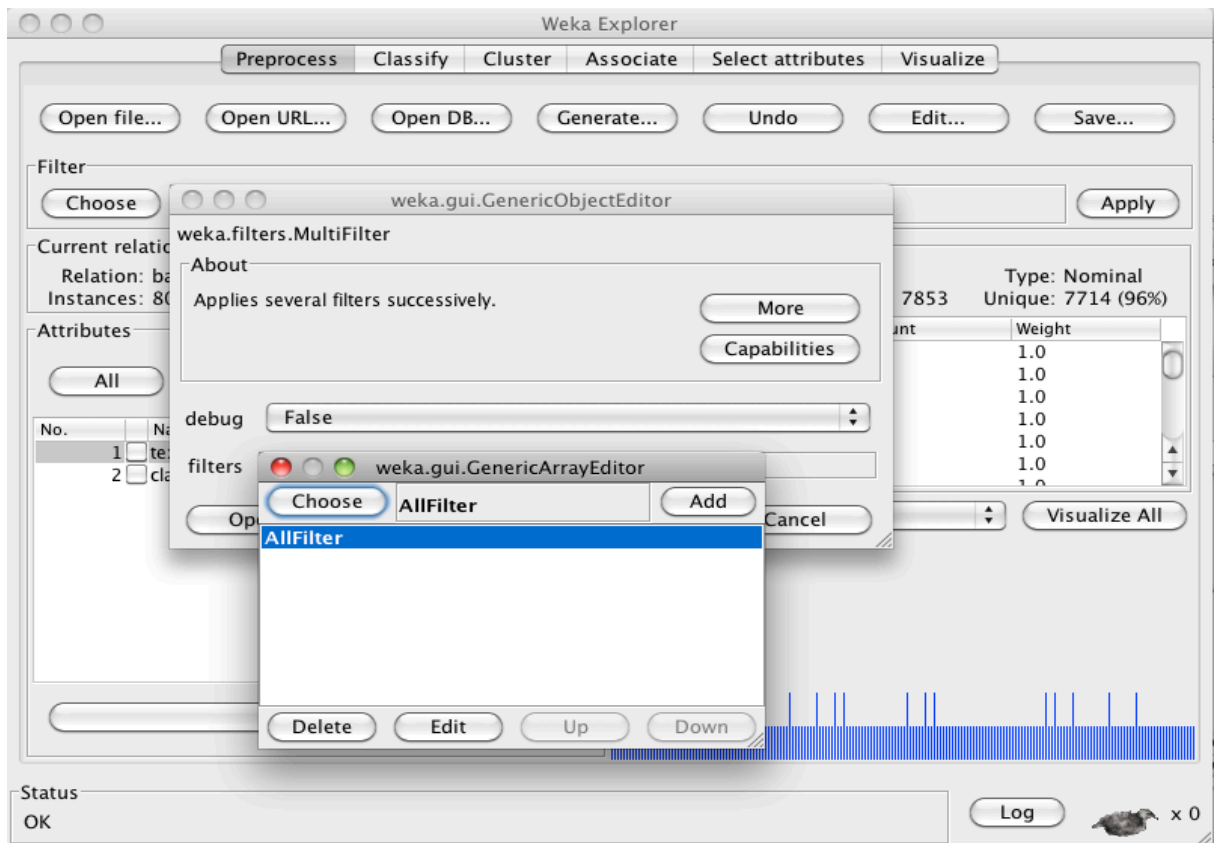
En esta segunda aproximación se ha optado por un método clásico de clasificación de documentos, las bolsas de palabras. En este caso vamos a sustituir las palabras con N-Gramas de 1 hasta 4 palabras, lo cual nos permitirá conocer la evolución de los N-Gramas en los diferentes contextos.

Además vamos a utilizar Stemmer y Stopwords para mejorar la frecuencia de aparición de las palabras y para eliminar las palabras que solo añaden ruido.

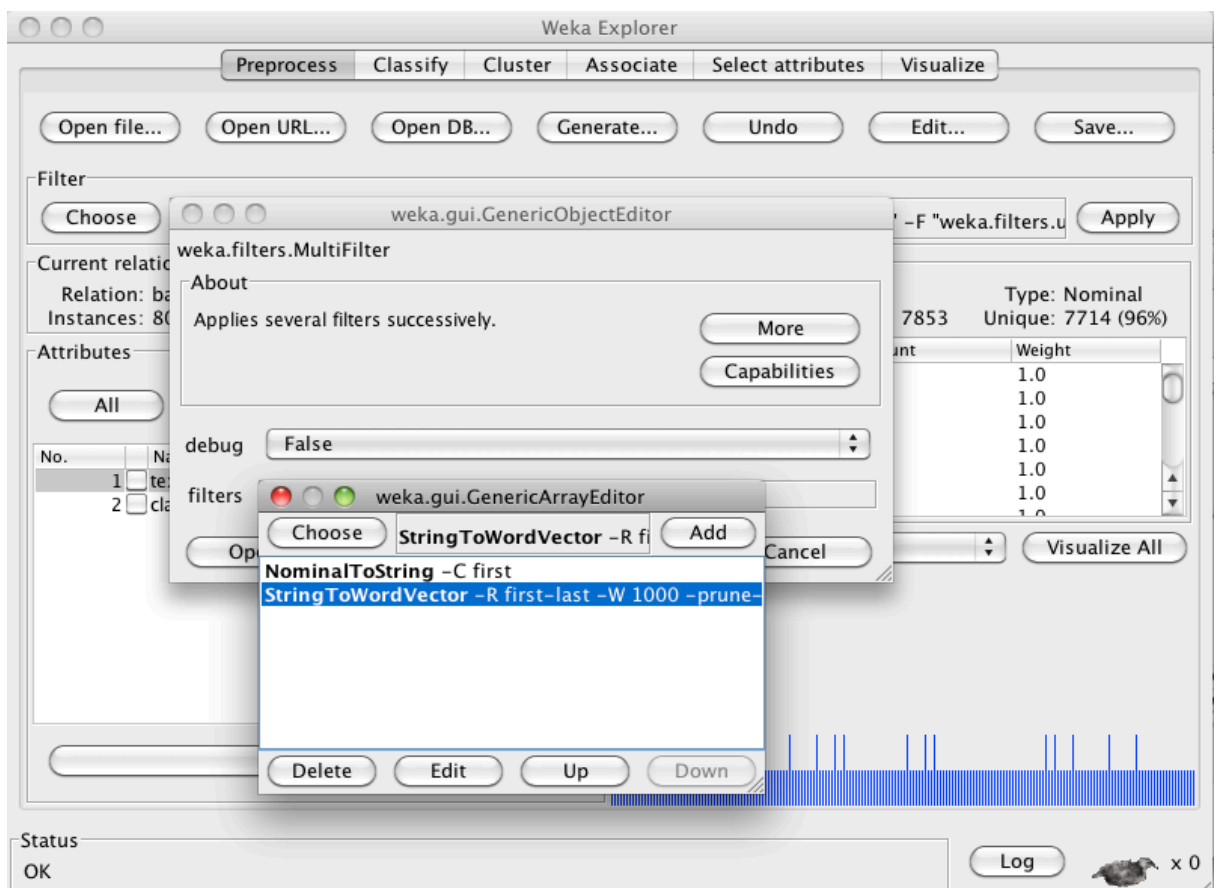
En esta aproximación se ha optado por utilizar diferentes combinaciones con los tres componentes N-Gramas, Stemmer y Stopwords, permitiéndonos de esta manera elegir la mejor combinación a posteriori.

Ahora vamos a ver un ejemplo de cómo crear de forma rápida una bolsa de palabras con la interfaz gráfica de WEKA, de esta manera podemos tener una idea de cómo funcionará aunque evidentemente para una aplicación en el mundo real necesitaremos crear el proceso de manera programada.

Primero realizamos la carga de datos. En este caso vamos a utilizar el mismo formato que utilizaremos en nuestra versión programada, un archivo csv, cuyo primer parámetro es la review y el segundo parámetro es la polaridad mediante las palabras pos y neg (positivo y negativo). aplicamos dos filtros, para ello cargamos el filtro *multifilter*.

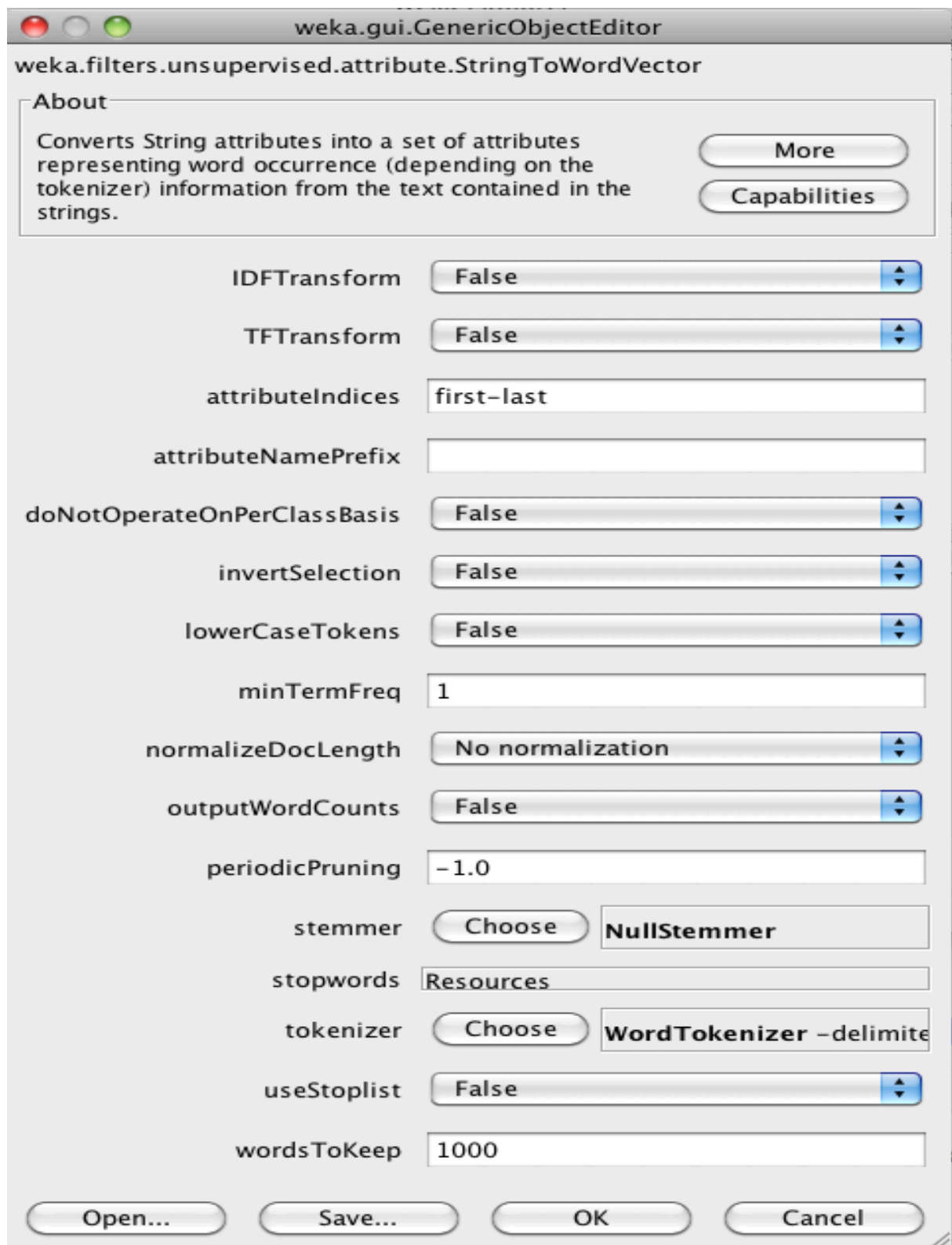


A continuación seleccionamos los filtros *NominalToString* y *StringToWordVector*. El primero convierte los valores en ristra de caracteres para que el segundo filtro pueda convertir las instancias en vectores de caracteres.

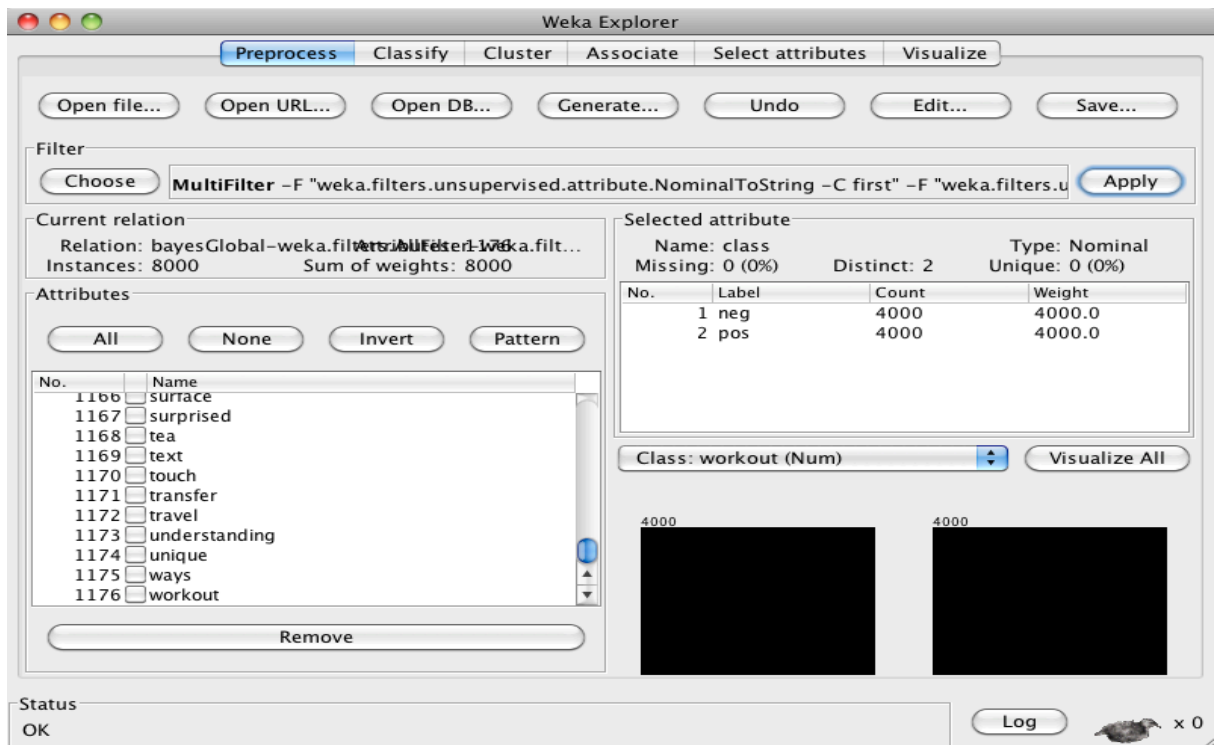


En este caso concreto tenemos que variar el primer filtro indicando que solo el primer campo se convierta a ristra de caracteres, es decir el campo review.

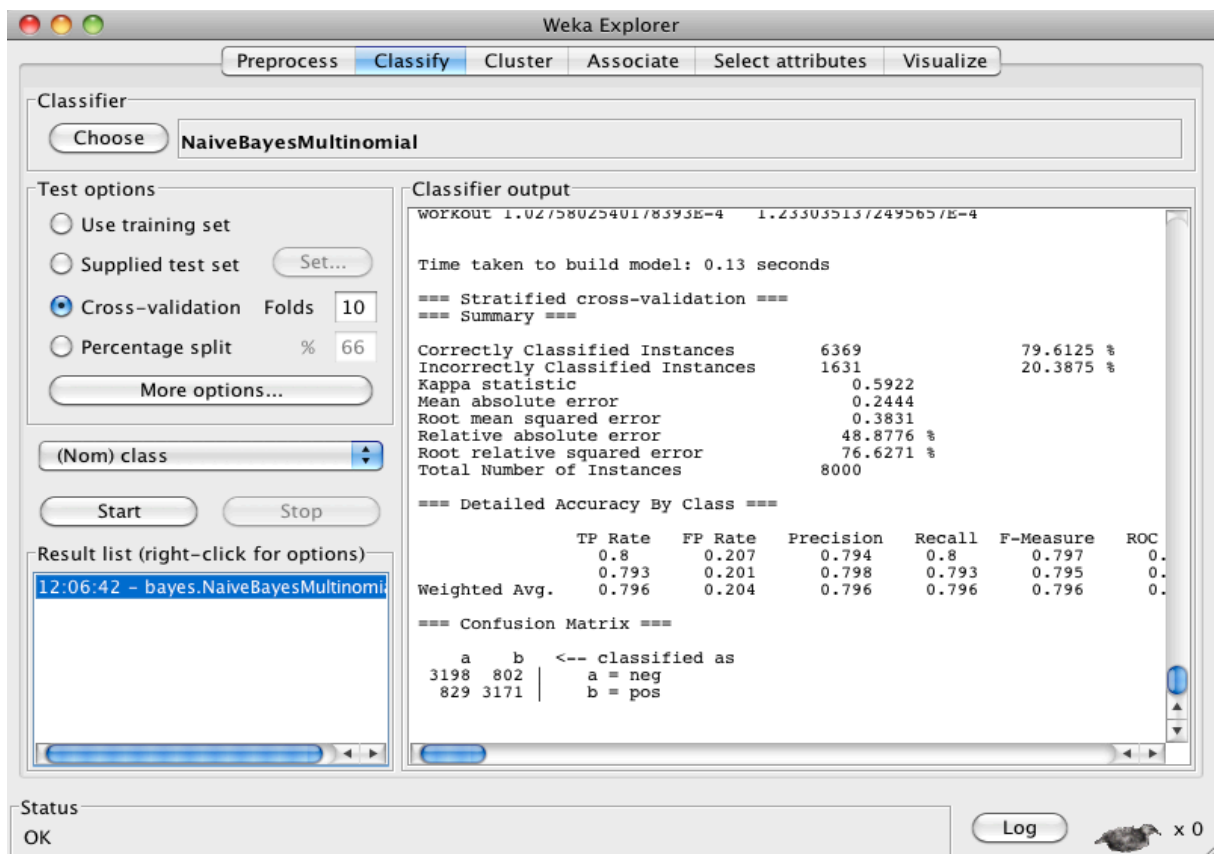
El segundo filtro es mas complejo ya que podemos decidir múltiples opciones tales como aplicar TF IDF, convertir todas las palabras a minúsculas, eliminar los términos con frecuencias inferiores a un umbral, aplicar frecuencias normalizadas con respecto al tamaño de los documentos, la utilización de stemmer y cual utilizar, eliminar las palabras de la lista de stopword, numero máximo de palabras que debe mantener y la opción que mas nos interesa la tokenización, sobre la cuál podemos elegir las opciones para combinación de n-gramas que vayamos a utilizar.



Una vez aplicados los filtros podemos ver como el número de atributos aumenta, debido a la creación de la bolsa de palabras.



El siguiente paso será seleccionar un algoritmo de clasificación que trabaje con valores nominales como por ejemplo Naive Bayes Multinomial.



En la polaridad podríamos introducir valores nulos los cuales serían descubiertos con el entrenamiento. La forma más sencilla es añadir el valor '?' ya que es el valor por defecto que WEKA asigna a los nulos.

8. Evaluación del Clasificador

El método mas sencillo para la validación es la división simple de los datos en datos de entrenamiento y datos de test. Desgraciadamente este modelo suele sobreestimar, por tanto se ha optado por utilizar Cross Validation, el cual divide en K conjuntos la muestra. En cada iteración se construye y evalúa un modelo utilizando como test el conjunto y el resto como entrenamiento. Los resultados obtenidos son la media aritmética de los resultados de los K conjuntos. En nuestro caso particular se ha utilizado k=10 por ser un valor usual.

Para la comparación entre experimentos se ha decidido utilizar tres factores relacionados entre sí, la precisión, el recall y la medida-f.

La *precisión* en la clasificación de textos varía de la definición clásica y se puede definir utilizando la siguiente ecuación:

$$\frac{\text{documentos relevantes} \cap \text{documentos recuperados}}{\text{documentos recuperados}}$$

en otras palabras la precisión indica la exactitud del clasificador, una precisión alta indica pocos falsos positivos.

El *recall* indica la sensibilidad del clasificador, un recall alto indica menos falsos negativos.

$$\frac{\text{documentos relevantes} \cap \text{documentos recuperados}}{\text{documentos relevantes}}$$

Por tanto, si mejora el recall es probable que empeore la precisión ya que es más difícil ser preciso en un entorno más acotado.

La *medida F* (F-measure) es la media armónica entre recall y precisión, y es la medida que tomaremos como parámetro clave de decisión.

8.1. Datos

Existen varios métodos para obtener o crear el conjunto de datos de test, en este caso se ha optado por utilizar un dataset creado a partir de datos de las

opiniones vertidas en la web de Amazon y cuya creación esta explicada en un artículo de investigación[72].

8.1.1. Amazon Dataset

Para la evaluación se ha utilizado el conjunto de datos multidominio de Amazon, algunos dominios tienen cientos de miles de reviews, y otros solo unos cientos. Cada review está evaluada con valores entre 1 y 5, lo cuál no es óptimo para nuestra aproximación por lo cual vamos a utilizar el filtrado realizado por Mansour[68], que convierte los datos en datos positivos y negativos.

El fichero está estructurado por carpetas, dentro de las cuales podemos encontrar 5 archivos:

- all.review Todas las reviews en el formato original
- positive.review
- negative.review
- unlabeled.review
- processed.review Reviews polarizadas
- processed.review.balanced Reviews polarizadas y con el número de positivas y negativas balanceadas (mismo número de frases positivas y negativas)

Todas las reviews contiene un identificador (id), pero éste no se utiliza porque se ha destruido en el preproceso y ha dejado de ser único.

En este caso particular se han usado cuatro dominios: dvd, books, electronics y kitchen. Para nuestra evaluación se ha utilizado las reviews polarizadas y balanceados. Lo que nos deja con 2000 instancias por dominio.

Finalmente se ha creado un dominio único uniendo todas las reviews obteniendo un mega dominio de propósito mas general con 8000 instancias.

8.2. Algoritmos de Machine Learning

Dentro de las aproximaciones elegidas se han utilizado cuatro algoritmos de aprendizaje de tres ramas diferentes, bayesiana (Naive Bayes, Bayes Multinomial), arboles de decisión (J48) y funciones de regresión (Regresión Lógica).

- *Naïve Bayes* [61] considera la probabilidad de aparición de cada término dada la clase de forma binaria, es decir el termino aparece o no y entonces su probabilidad condicional dada la clase es o no considerada.
- *Bayes Multinomial* es igual que *Naïve Bayes* pero considera el número de apariciones del término para evaluar la contribución de la probabilidad condicional dada la clase con lo que el modelado de cada documento se ajusta mejor a la clase a la que pertenece.
- *J48* es una implementación de C4.5, genera un árbol de decisión a partir de los datos mediante particiones realizadas recursivamente. El árbol se construye mediante la estrategia de profundidad-primero (depth-first). Se consideran todas las pruebas posibles que pueden dividir el conjunto de datos y selecciona la prueba que resulta en la mayor ganancia de información. Para cada atributo discreto, se considera una prueba con n resultados, siendo n el número de valores posibles que puede tomar el atributo. Para cada atributo continuo, se realiza una prueba binaria sobre cada uno de los valores que toma el atributo en los datos. En cada nodo, el sistema debe decidir qué prueba escoge para dividir los datos. Se ha elegido este algoritmo por ser el utilizado por los creadores de WEKA en sus analizadores de texto [64, 65, 66] y uno de los más utilizado en la literatura científica para clasificar texto.
- *Simple Logistic*, la regresión logística[62] se utiliza cuando el objetivo es describir la relación entre una clase categórica y un conjunto de variables asociadas, que pueden ser categorías o cuantitativas. En este caso son todas cuantitativas.

8.3. Resultados

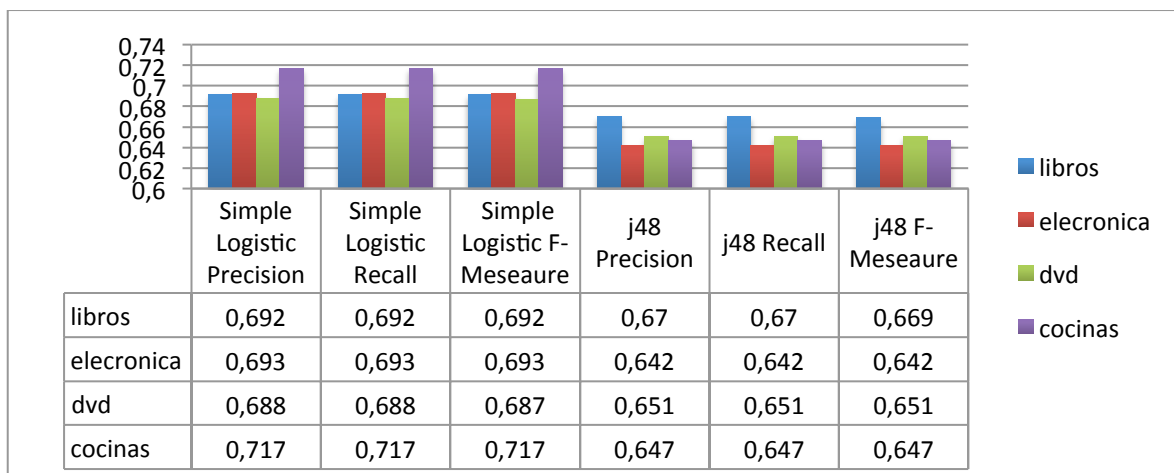
Se ha dividido la sección en las dos aproximaciones. Dentro de la aproximación de *bolsa de palabras* ha sido necesario crear subsecciones que representan los distintos dominios del conjunto de datos.

Aunque se han probado todos los algoritmos en ambas aproximaciones solo mostramos los datos de los dos mejores algoritmos.

8.3.1. Extracción de características

En este gráfico podemos observar que se trata de una aproximación muy pobre ya que los resultados están lejos de la barrera del 80% que se suele fijar como

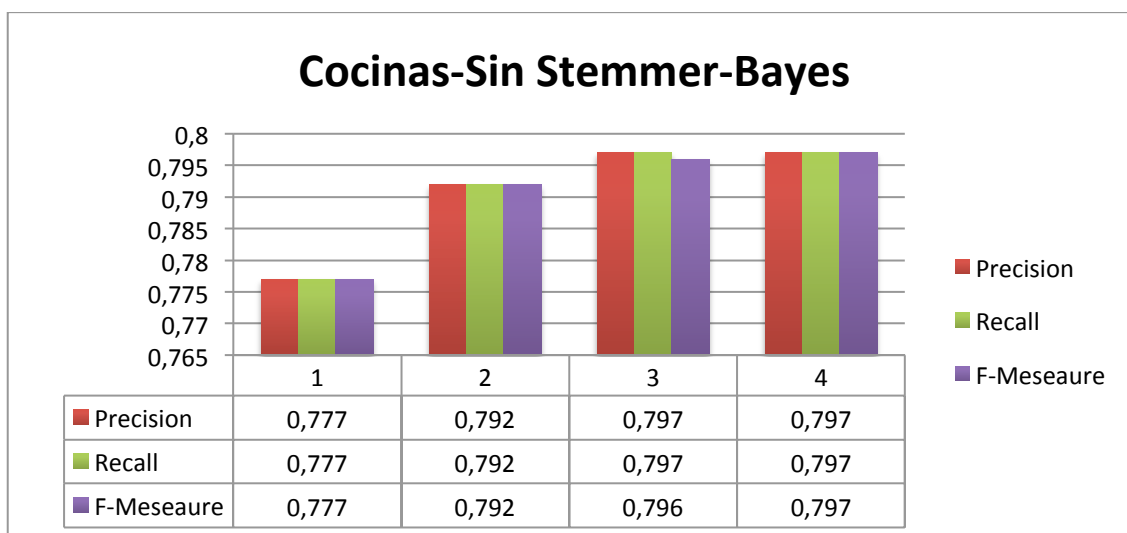
meta. También podemos observar que la regresión logística es mejor en todos los casos que el árbol de decisión J48.



8.3.2. Bag of Word

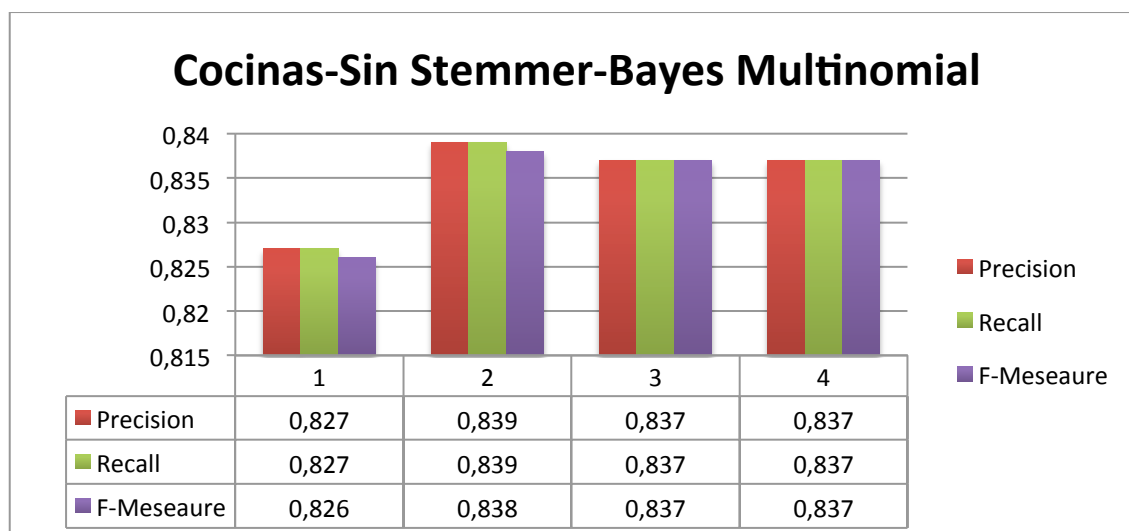
Se han realizado pruebas para cada uno de los dominios, cocinas, electrónica, dvd y libros, aunque incluimos las gráficas de todos los dominios, utilizamos la misma explicación ya que los resultados son muy similares.

En esta primera gráfica observamos como en el caso de no utilizar stemmer ni stopwords, los N-gramas aumentan la precisión y la medida-f pero desde la utilización de $N=2$ hasta $N=4$ la mejoría es mínima.

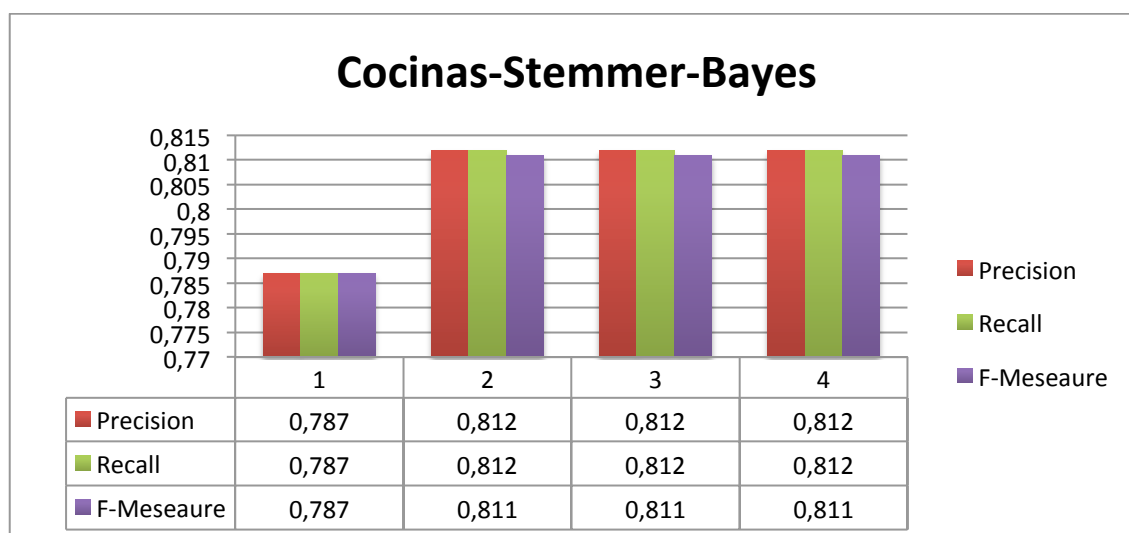


En esta segunda gráfica utilizando Bayes Multinomial vemos algo que será pauta en todos los datos $N=2$ es la mejor opción o como mínimo igual de buena que la

utilización de N de mayor tamaño. También podemos observar que, para todos los casos, el algoritmo Bayes Multinomial es mejor que el Naive Bayes.

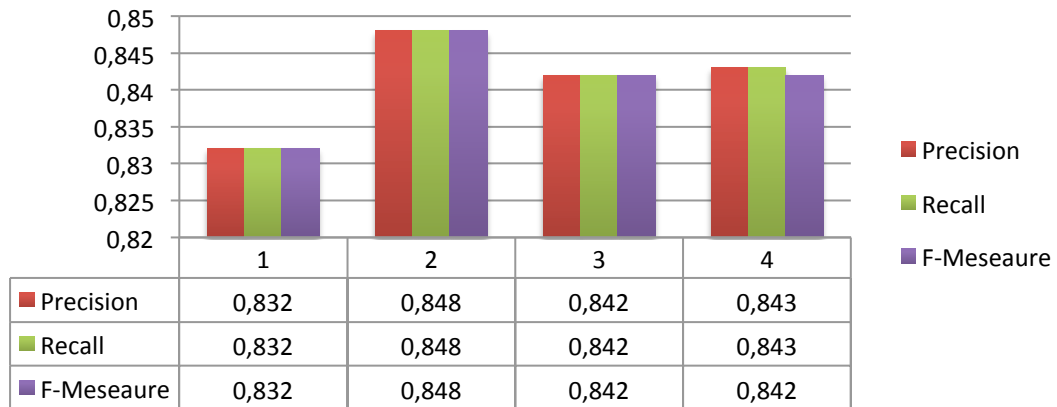


La utilización de Stemmer en Naive Bayes, mejora a los resultados y sitúa a $N=2$ al mismo nivel que $N=3$ y $N=4$.



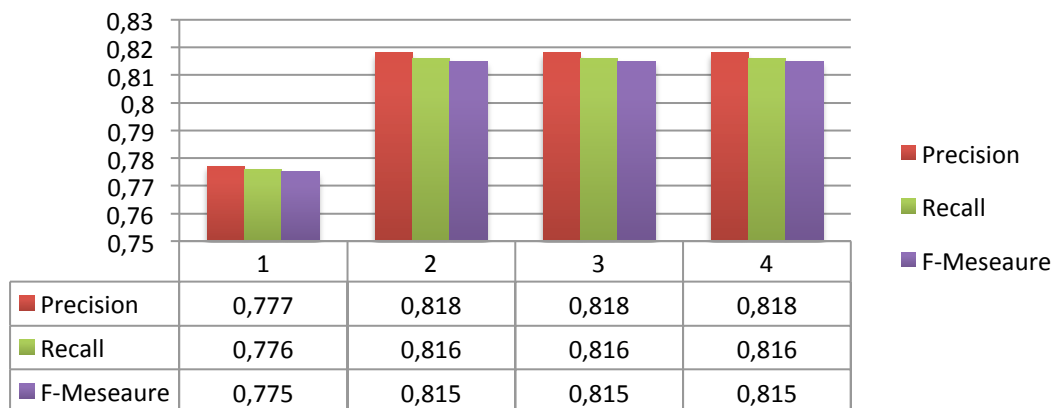
En el caso de Bayes Multinomial obtenemos una mejora en todos los casos y $N=2$ mejora más que el resto de N .

Cocinas-Stemmer-Bayes Multinomial



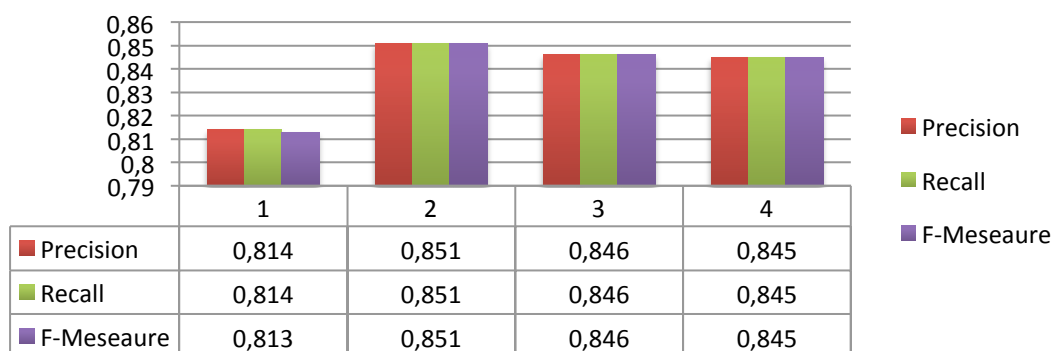
Si utilizamos stopwords con Naive Bayes mantiene la situación aunque aumentan los N-Gramas con $N > 1$ y disminuye con $N = 1$.

Cocinas-Stemmer+StopWords-Bayes



En el caso de Bayes Multinomial sucede exactamente lo mismo que en el caso anterior con Naive Bayes.

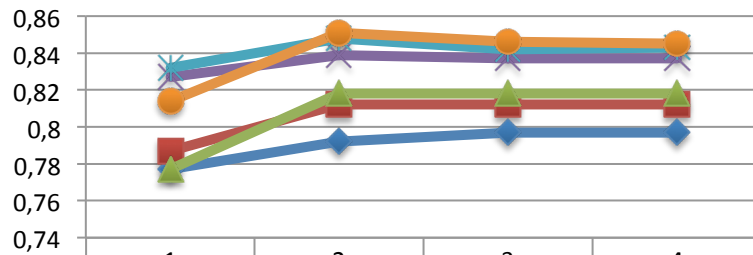
Cocinas-Stemmer+StopWord-Bayes Multinomial



Se ha dividido las gráficas siguientes en tres para su mejor visualización. Podemos observar que Bayes Multinomial es mejor que Naive Bayes para todas las configuraciones, se ve aprecia claramente como las pruebas con Bayes Multinomial se agrupan sobre las pruebas con Naive Bayes como . También podemos observar claramente la evolución de los N-Gramas, para $N=2$ mejora en todos los caso, pero para N mayores no lo hace.

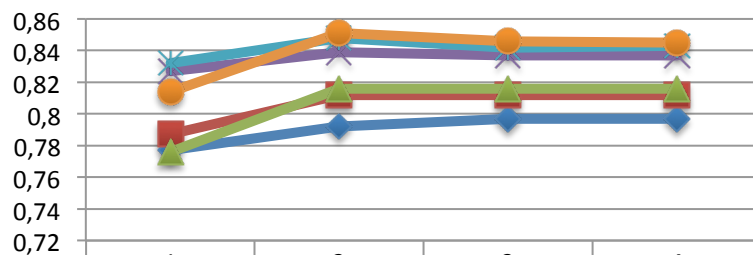
De todos los datos podemos obtener que la mejor configuración es utilizar N-Gramas con $N=2$,stopword y stemmer junto con el algoritmo de clasificación Bayes Multinomial

Cocinas-Presición



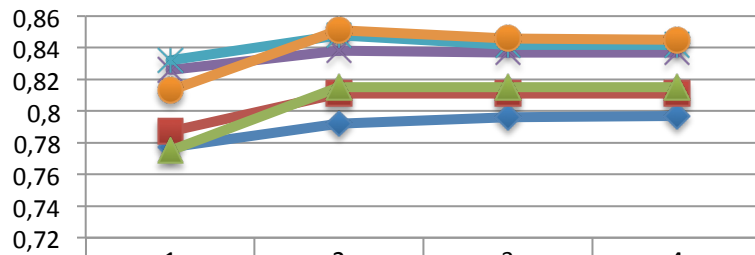
	1	2	3	4
NBayes-Sin Stemmer	0,777	0,792	0,797	0,797
NBayes-Con Stemmer	0,787	0,812	0,812	0,812
NBayes-Con Stemmer+SW	0,777	0,818	0,818	0,818
BayesM-Sin Stemmer	0,827	0,839	0,837	0,837
BayesM-Con Stemmer	0,832	0,848	0,842	0,843
BayesM-Con Stemmer+SW	0,814	0,851	0,846	0,845

Cocinas-Recall



	1	2	3	4
NBayes-Sin Stemmer	0,777	0,792	0,797	0,797
NBayes-Con Stemmer	0,787	0,812	0,812	0,812
NBayes-Con Stemmer+SW	0,776	0,816	0,816	0,816
BayesM-Sin Stemmer	0,827	0,839	0,837	0,837
BayesM-Con Stemmer	0,832	0,848	0,842	0,843
BayesM-Con Stemmer+SW	0,814	0,851	0,846	0,845

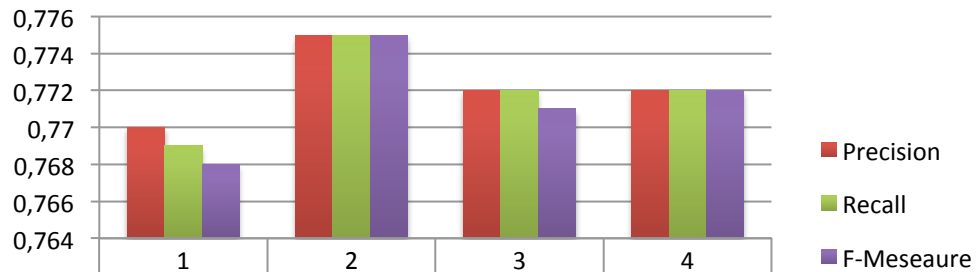
Cocinas-FMeasure



	1	2	3	4
NBayes-Sin Stemmer	0,777	0,792	0,796	0,797
NBayes-Con Stemmer	0,787	0,811	0,811	0,811
NBayes-Con Stemmer+SW	0,775	0,815	0,815	0,815
BayesM-Sin Stemmer	0,826	0,838	0,837	0,837
BayesM-Con Stemmer	0,832	0,848	0,842	0,842
BayesM-Con Stemmer+SW	0,813	0,851	0,846	0,845

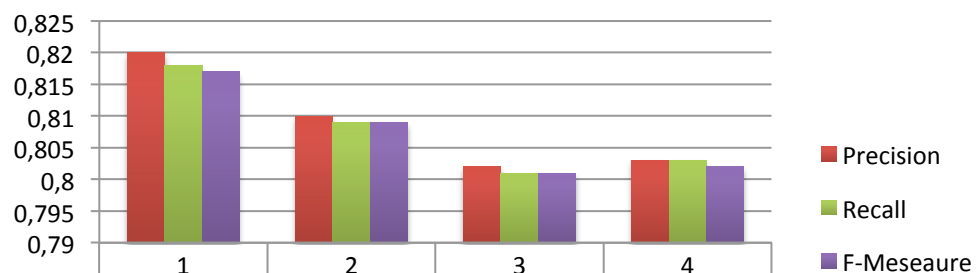
1.1.1.4. DVD

DVD's-Sin Stemmer-Bayes



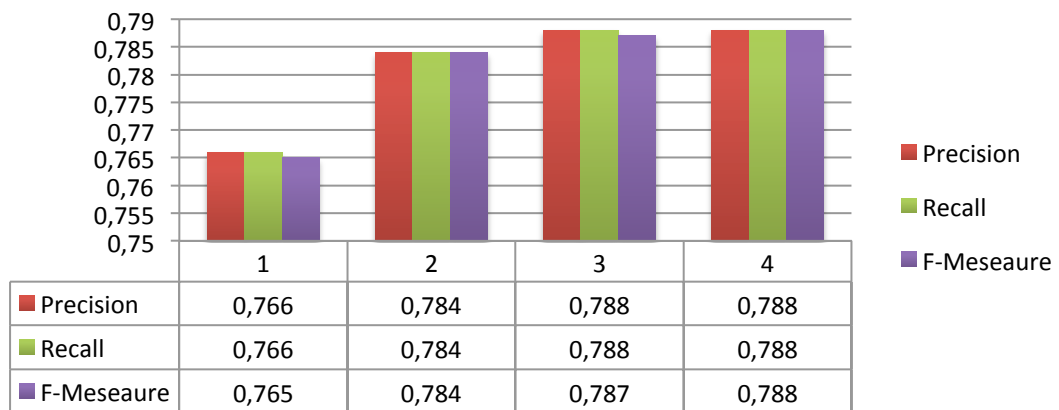
	1	2	3	4
Precision	0,77	0,775	0,772	0,772
Recall	0,769	0,775	0,772	0,772
F-Meseaure	0,768	0,775	0,771	0,772

DVD's-Sin Stemmer-Bayes Multinomial



	1	2	3	4
Precision	0,82	0,81	0,802	0,803
Recall	0,818	0,809	0,801	0,803
F-Meseaure	0,817	0,809	0,801	0,802

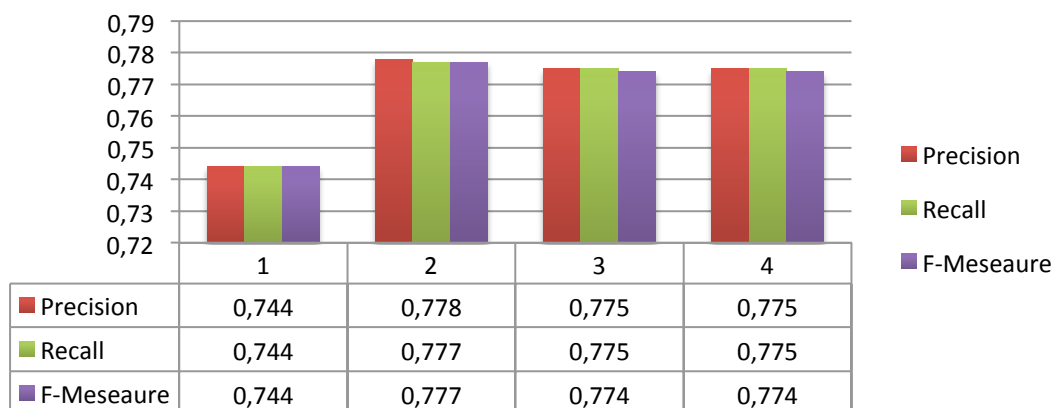
DVD's-Con Stemmer-Bayes



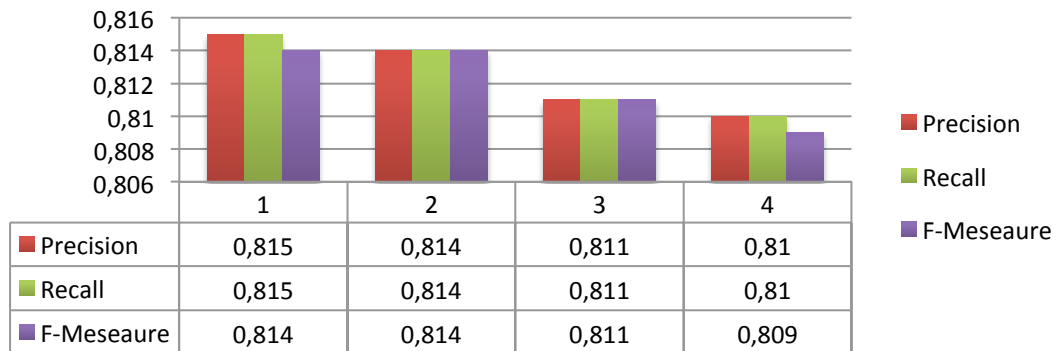
DVD's-Con Stemmer-Bayes Multinomial



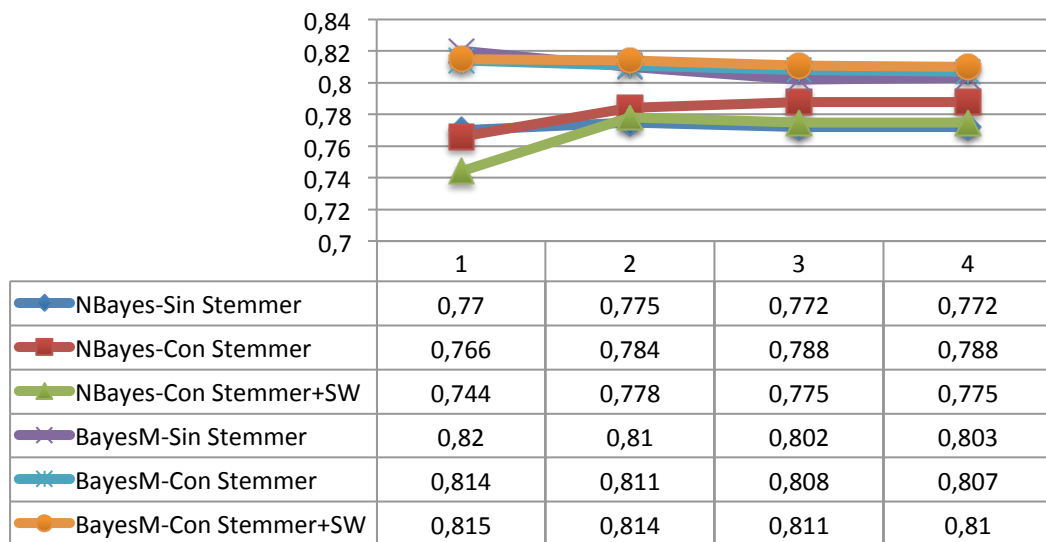
DVD's-Stemmer+StopWords-Bayes



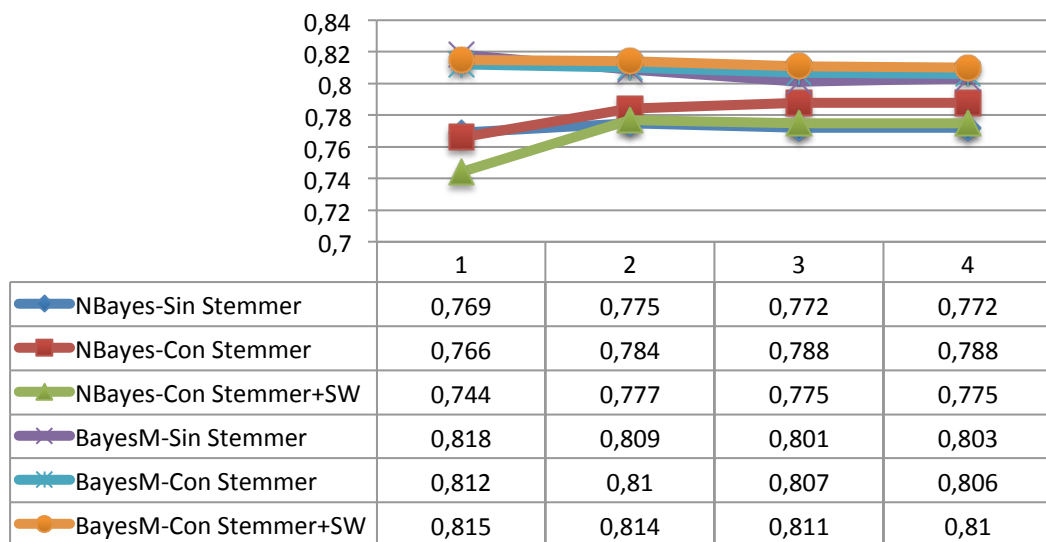
DVD's-Stemmer+StopWords-Bayes Multinomial



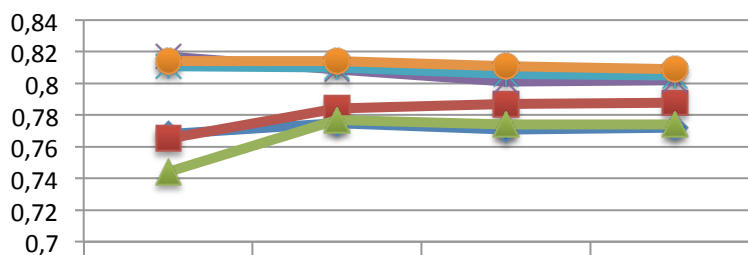
DVD's-Precisión



DVD's-Recall



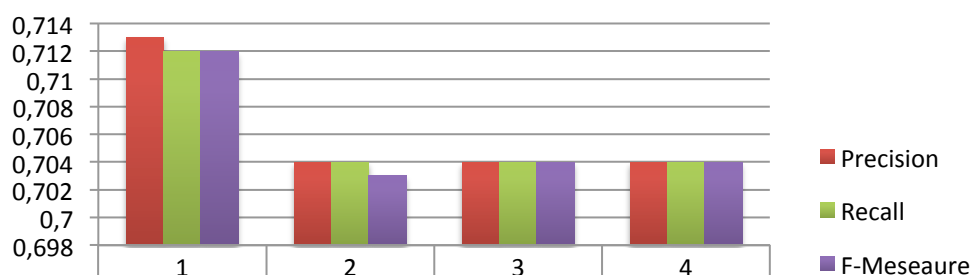
DVD's-FMeasure



	1	2	3	4
NBayes-Sin Stemmer	0,768	0,775	0,771	0,772
NBayes-Con Stemmer	0,765	0,784	0,787	0,788
NBayes-Con Stemmer+SW	0,744	0,777	0,774	0,774
BayesM-Sin Stemmer	0,817	0,809	0,801	0,802
BayesM-Con Stemmer	0,811	0,81	0,806	0,805
BayesM-Con Stemmer+SW	0,814	0,814	0,811	0,809

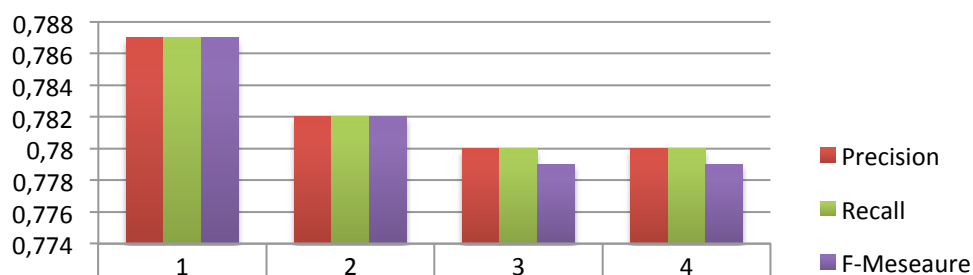
1.1.1.5. Libros

Libros-Sin Stemmer-Bayes



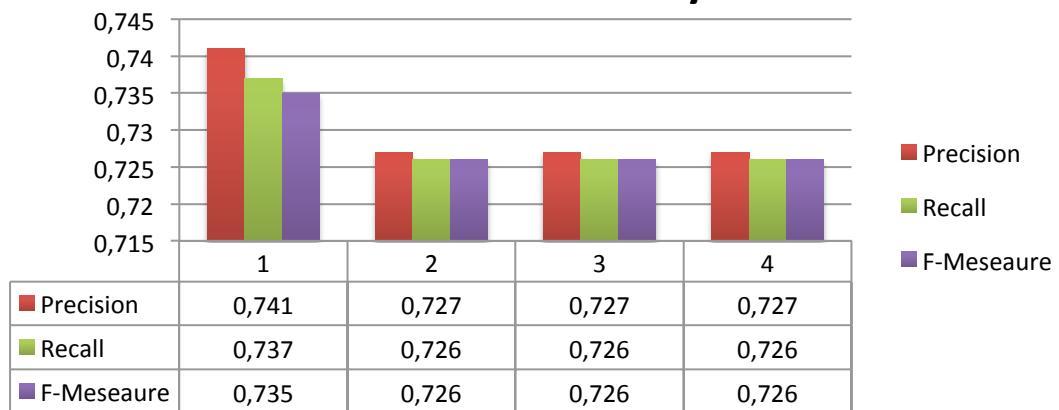
	1	2	3	4
Precision	0,713	0,704	0,704	0,704
Recall	0,712	0,704	0,704	0,704
F-Meseaure	0,712	0,703	0,704	0,704

Libros-Sin Stemmer- Bayes Multinomial

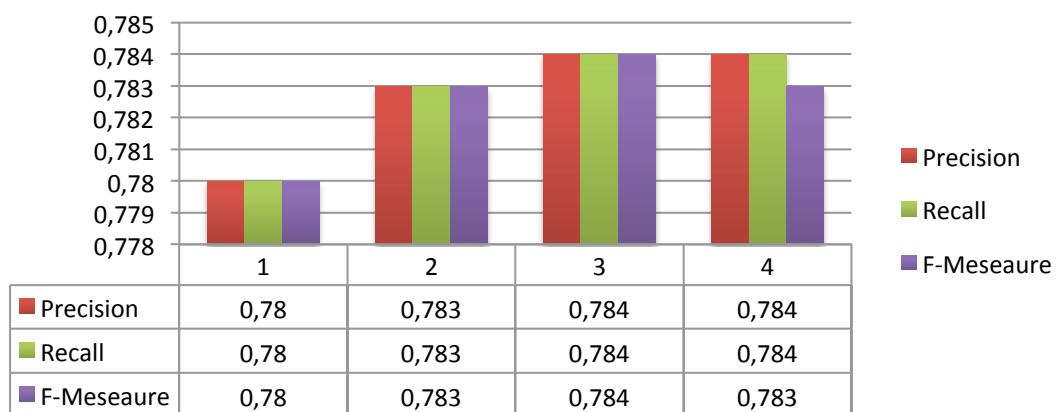


	1	2	3	4
Precision	0,787	0,782	0,78	0,78
Recall	0,787	0,782	0,78	0,78
F-Meseaure	0,787	0,782	0,779	0,779

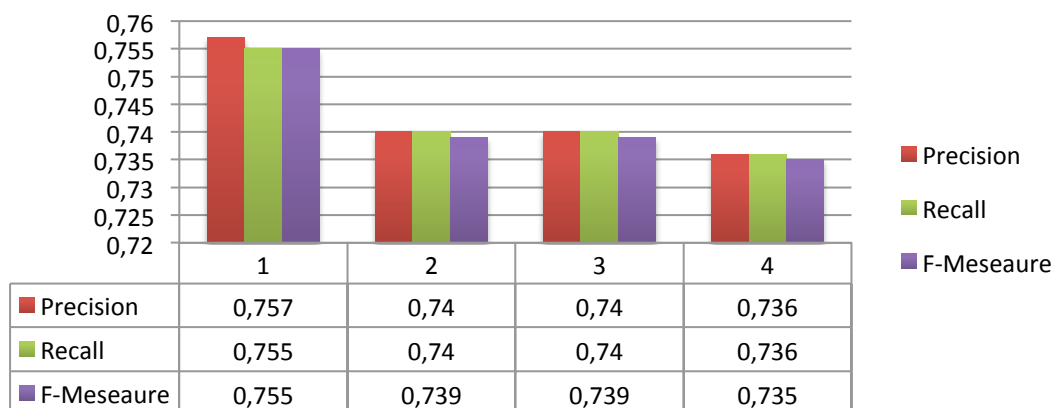
Libros-Con Stemmer- Bayes



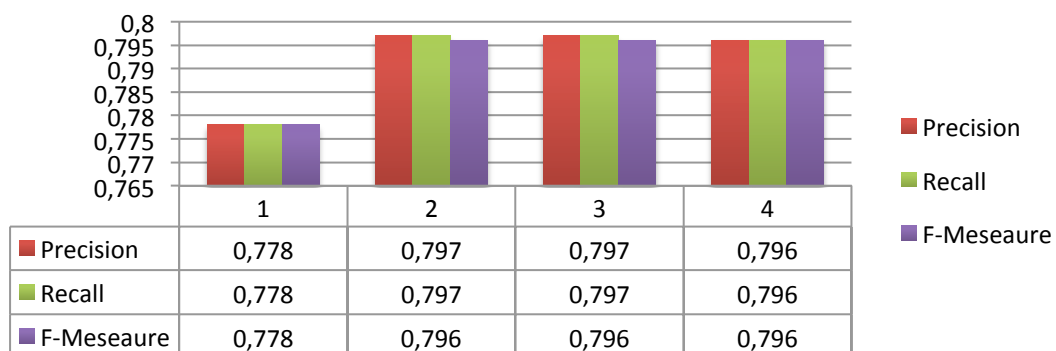
Libros-Con Stemmer-Bayes Multinomial



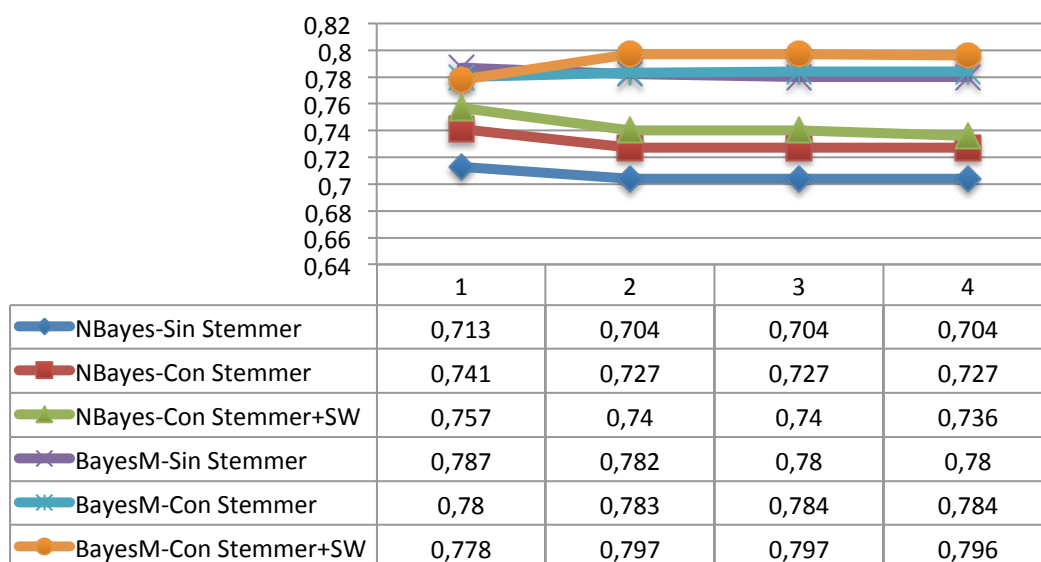
Libros-Stemmer+StopWords-Bayes



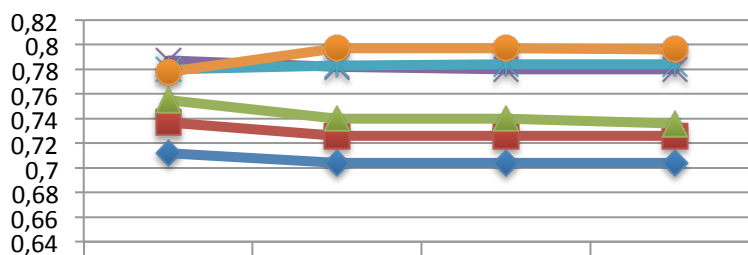
Libros-Stemmer+StopWords-Bayes Multinomial



Libros-Precisión

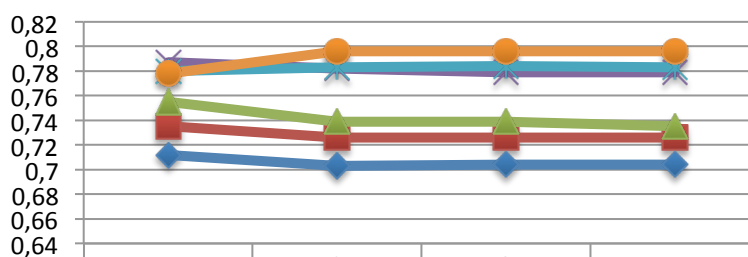


Libros-Recall



	1	2	3	4
NBayes-Sin Stemmer	0,712	0,704	0,704	0,704
NBayes-Con Stemmer	0,737	0,726	0,726	0,726
NBayes-Con Stemmer+SW	0,755	0,74	0,74	0,736
BayesM-Sin Stemmer	0,787	0,782	0,78	0,78
BayesM-Con Stemmer	0,78	0,783	0,784	0,784
BayesM-Con Stemmer+SW	0,778	0,797	0,797	0,796

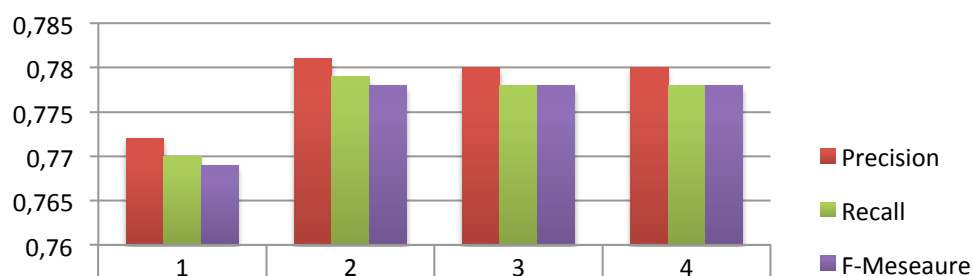
Libros-FMeasure



	1	2	3	4
NBayes-Sin Stemmer	0,712	0,703	0,704	0,704
NBayes-Con Stemmer	0,735	0,726	0,726	0,726
NBayes-Con Stemmer+SW	0,755	0,739	0,739	0,735
BayesM-Sin Stemmer	0,787	0,782	0,779	0,779
BayesM-Con Stemmer	0,78	0,783	0,784	0,783
BayesM-Con Stemmer+SW	0,778	0,796	0,796	0,796

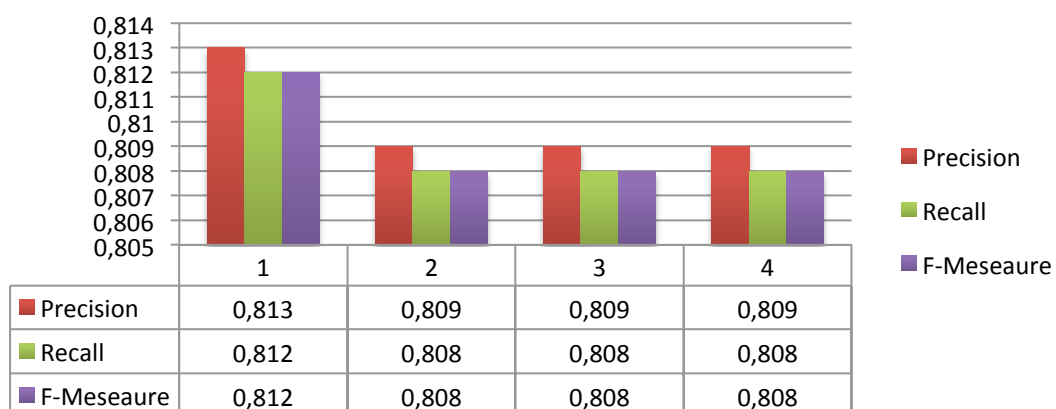
1.1.1.6. Electrónica

Electrónica-Sin Stemmer-Bayes

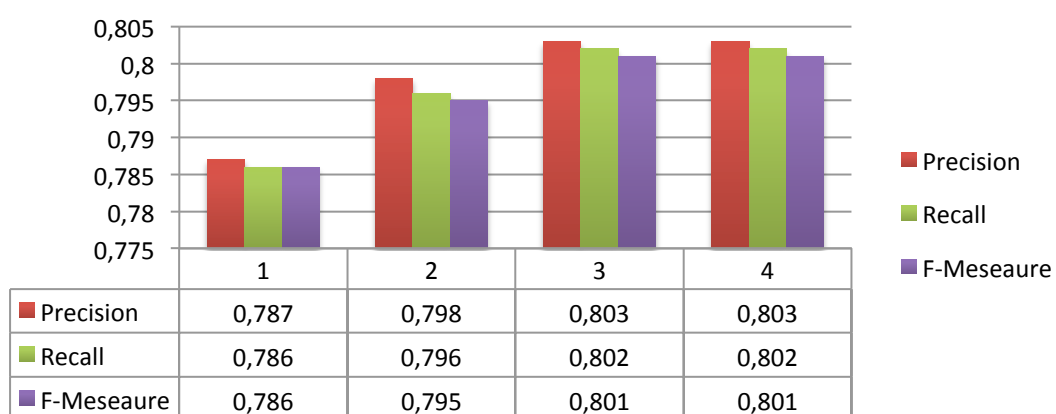


	1	2	3	4
Precision	0,772	0,781	0,78	0,78
Recall	0,77	0,779	0,778	0,778
F-Measure	0,769	0,778	0,778	0,778

Electrónica-Sin Stemmer-Bayes Multinomial



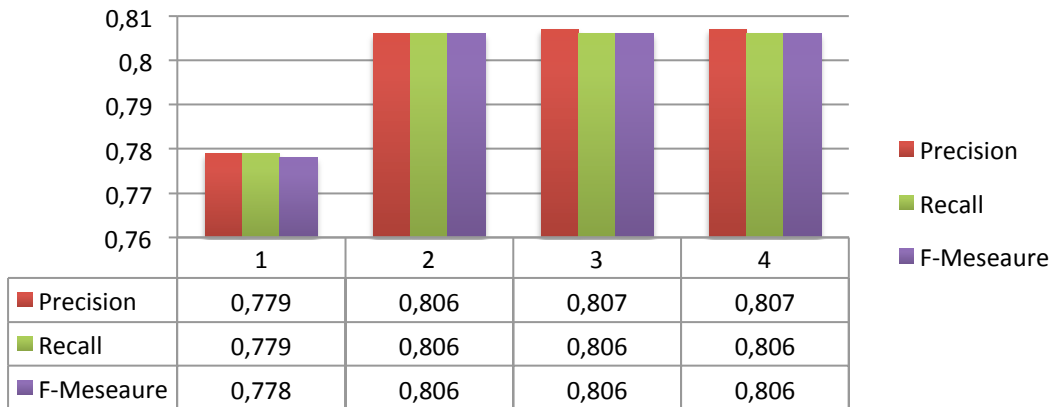
Electrónica-Stemmer-Bayes



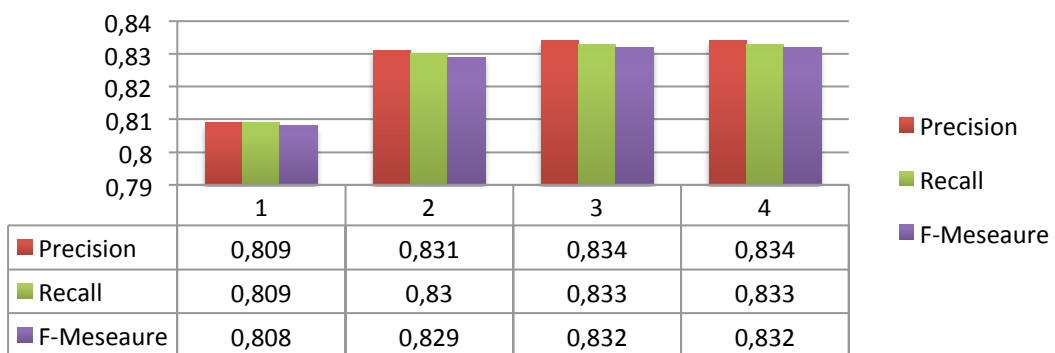
Electrónica-Stemmer-Bayes Multinomial



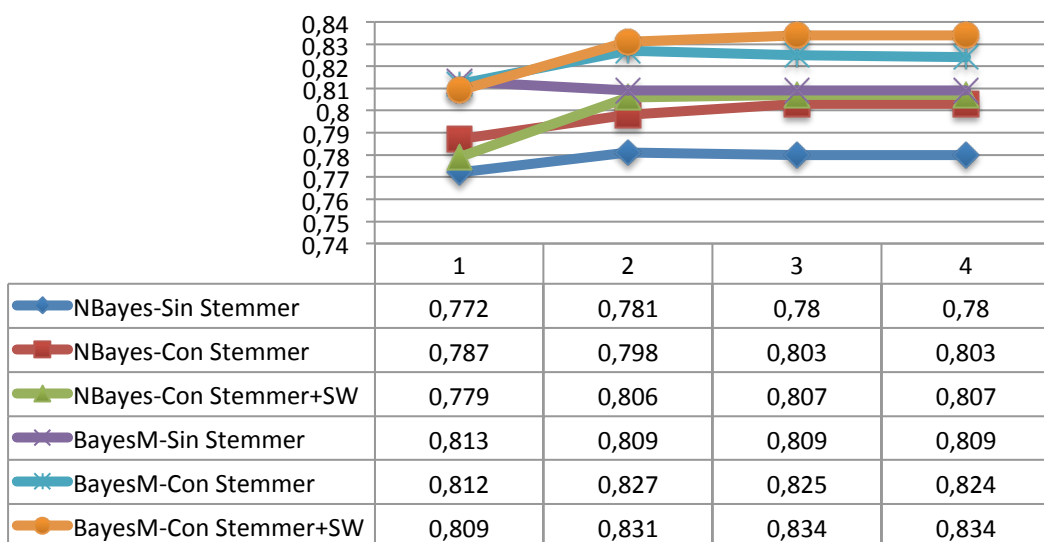
Electrónica-Stemmer+StopWords-Bayes



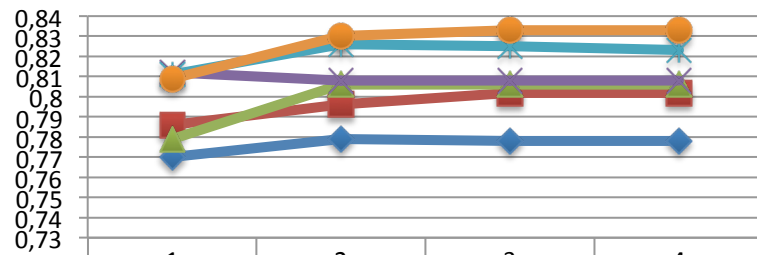
Electrónica-Stemmer+StopWords-Bayes Multinomial



Electrónica-Precisión

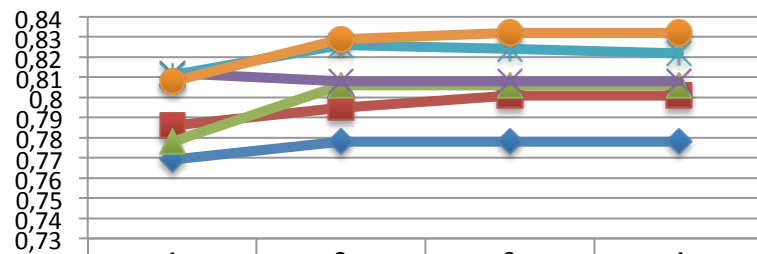


Electrónica-Recall



	1	2	3	4
NBayes-Sin Stemmer	0,77	0,779	0,778	0,778
NBayes-Con Stemmer	0,786	0,796	0,802	0,802
NBayes-Con Stemmer+SW	0,779	0,806	0,806	0,806
BayesM-Sin Stemmer	0,812	0,808	0,808	0,808
BayesM-Con Stemmer	0,811	0,826	0,825	0,823
BayesM-Con Stemmer+SW	0,809	0,83	0,833	0,833

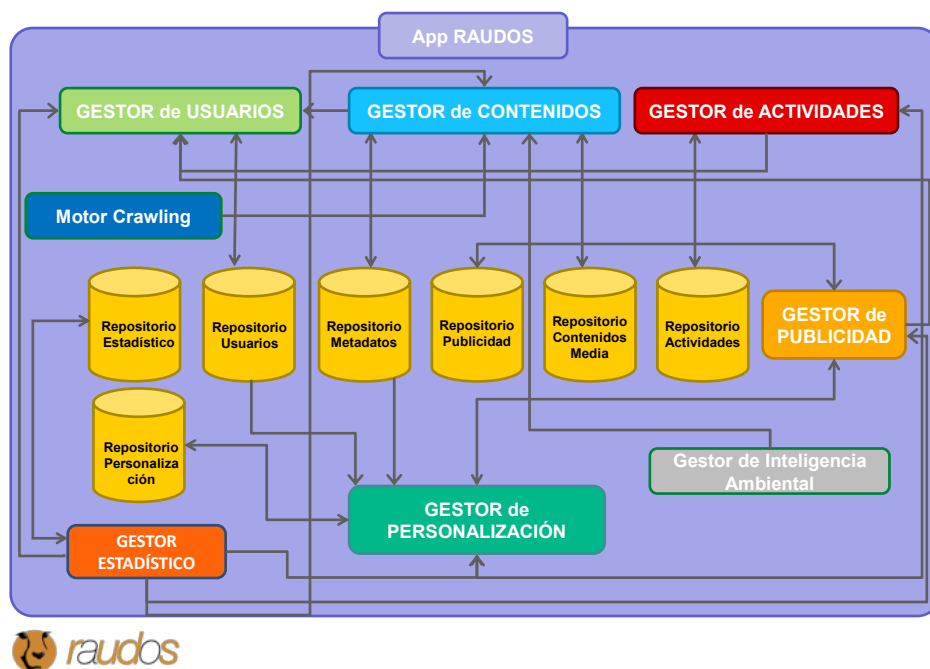
Electrónica-FMeasure



	1	2	3	4
NBayes-Sin Stemmer	0,769	0,778	0,778	0,778
NBayes-Con Stemmer	0,786	0,795	0,801	0,801
NBayes-Con Stemmer+SW	0,778	0,806	0,806	0,806
BayesM-Sin Stemmer	0,812	0,808	0,808	0,808
BayesM-Con Stemmer	0,811	0,826	0,824	0,822
BayesM-Con Stemmer+SW	0,808	0,829	0,832	0,832

9.Creación del Prototipo

El clasificador generado como resultado de las pruebas será incluido en el proyecto RAUDOS 2, más específicamente dentro del Gestor de Publicidad, en el siguiente gráfico podemos ver una diagrama general del proyecto:



La relación del Gestor de Publicidad entre el resto de gestores es la siguiente:



- Gestor de Usuarios. Aumentar del número de tokens.
- Gestor de Personalización. Obtener de criterios de personalización.
- Gestor de Estadístico. Registrar el consumo de publicidad.

- Gestor de Crawling. Proporciona un resumen de los nuevos productos, y periódicamente los actualiza con comentarios y ratings de diversas fuentes.

El prototipo utilizando la información proporcionada por el Gestor de Crawling realiza un aprendizaje evolutivo LA dos funciones creadas para la ingesta de datos tienen la siguiente interfaz:

1. **addReview** esta función permite al Crawler introducir valoraciones de un producto obtenidas de internet. Los parámetros que tiene asignados son:

- **idProduct** : la id del producto que previamente nosotros demos de alta en el crawler para ser buscado.
- **reviewText**: texto de la opinión encontrada
- **idUser**: id del usuario que haya dado su opinión. Es el id referido a la web donde se encuentra su opinión NO a el idUser de raudos2
- **rating**
- **source**: fuente donde se ha encontrado la información
- **timestamp**: fecha en formato timestamp

2. **addSummary** esta función permite al Crawler introducir descripciones del producto obtenidas de internet. Los parámetros que tiene asignados son:

- **idProduct**: idem anterior
- **summary**: texto descriptivo del producto
- **source**: idem anterior

Ambas funciones tienen un primer parámetro *pSessionID* herencia de raudos 1.

Mediante una función que acepta como parámetros de entrada dos cadenas de caracteres, la primera es la cadena con la información que quiere ser evaluada, y la segunda indica mediante código el idioma del texto, como resultado devolverá un entero con valor 1 para positiva y -1 para negativa. Será usada para mejorar las recomendaciones generadas por el gestor de personalización, sobre todo aquellas relacionadas con la publicidad. Actualmente solo existe la versión en inglés.

Internamente este clasificador utilizará como dataset de entrenamiento el dataset global creado en amazon, para tratar de abarcar más cantidad de

dominios, y utilizará una implementación de la segunda aproximación, es decir incluirá bolsas de palabras con stemmer y stopwords. Utilizaremos bigramas porque se ha comprobado que es la combinación que mejores resultados proporciona en general.

10. Conclusiones y Trabajo Futuro

Este trabajo de fin de master ha cumplido con el objetivo de ser una iniciación a una nueva vía de investigación, aportando las bases y herramientas. Se han explorado dos vías de aproximación dejando de lado otras igual de válidas.

Se ha comprobado que un número elevado de N-Gramas no mejora el clasificador, y que las técnicas clásicas de clasificación de datos, stemmer y stopwords sí lo mejoran.

Ha permitido crear una arquitectura que puede ser reutilizada en el futuro, y que incluye procesos comunes al análisis de sentimientos, que nos ahorrarán muchas horas de trabajo en siguientes investigaciones.

También ha permitido incluir diferentes APIS, bajo una misma interfaz, lo que nos permitirá incluir nuevas herramientas y compararlas de forma rápida y sencilla.

Por último se ha creado un clasificador prototipo, que aunque mejorable tiene unos resultados en un rango humano como se ha visto en la sección de resultados.

Los siguientes pasos que debemos tomar es la mejora de la arquitectura, incluyendo herramientas para la selección automática de atributos, así como el filtrado de los N-Gramas más relevantes. Más herramientas de Análisis Sintáctico como Freeling[63] y nuevas APIs para la entrada de datos recogidos de rss y redes sociales como twitter podrían ser añadidas.

11. Referencias

[1] comScore/the Kelsey group. Online consumer-generated reviews have significant impact on offline purchase behavior. Press Release, November 2007. <http://www.comscore.com/press/release.asp?press=1928>.

[2] John A. Horrigan. Online shopping. Pew Internet & American Life Project Report, 2008.

[3] Lee Rainie and John Horrigan. Election 2006 online. Pew Internet & American Life Project Report, January 2007.

[4] Real Academia Española (2011, Junio). <http://www.rae.es/rae.html>

[5] Vasileios Hatzivassiloglou and Kathleen McKeown. Predicting the semantic orientation of adjectives. In *Proceedings of the Joint ACL/EACL Conference*, pages 174-181, 1997.

[6] Anthony Aue and Michael Gamon. Automatic identification of sentiment vocabulary: Exploiting low association with known sentiment terms. In *Proceedings of the ACL Workshop on Feature Engineering for Machine Learning in Natural Language Processing*, 2005.

[7] Jaap Kamps, Maarten Marx, Robert J. Mokken, and Maarten de Rijke. Using WordNet to measure semantic orientation of adjectives. In *LREC*, 2004.

[8] Peter D. Turney and Michael L. Littman. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems (TOIS)*, 21(4):315-346, 2003.

[9] Lin, D. and Zhao, S. and Qin, L. and Zhou, M. Identifying synonyms among distributionally similar words. INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE. Volumen 18 pages 1492-1493, 2003

[10] Jaime Carbonell. *Subjective Understanding: Computer Models of Belief Systems*. PhD thesis, Yale, 1979.

[11] Marti Hearst. Direction-based text interpretation as an information access refinement. In Paul Jacobs, editor, *Text-Based Intelligent Systems*, pages 257-274. Lawrence Erlbaum Associates, 1992.

[12] Alison Huettner and Pero Subasic. Fuzzy typing for document management. In *ACL 2000 Companion Volume: Tutorial Abstracts and Demonstration Notes*, pages 26-27, 2000.

[13] Mark Kantrowitz. Method and apparatus for analyzing affect and emotion in text. U.S. Patent 6622140, 2003. Patent filed in November 2000

[14] Warren Sack. On the computation of point of view. In *Proceedings of AAAI*, page 1488, 1994. Student abstract

[15] Janyce Wiebe and Rebecca Bruce. Probabilistic classifiers for tracking point of view. In *Proceedings of the AAAI Spring Symposium on Empirical Methods in Discourse Interpretation and Generation*, pages 181-187, 1995.

[16] Janyce M. Wiebe. Identifying subjective characters in narrative. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, pages 401-408, 1990.

[17] Janyce M. Wiebe. Tracking point of view in narrative. *Computational Linguistics*, 20(2):233-287, 1994.

[18] Janyce M. Wiebe, Rebecca F. Bruce, and Thomas P. O'Hara. Development and use of a gold standard data set for subjectivity classifications. In *Proceedings of the Association for Computational Linguistics (ACL)*, pages 246-253, 1999.

[19] Janyce M. Wiebe and William J. Rapaport. A computational theory of perspective and reference in narrative. In *Proceedings of the Association for Computational Linguistics (ACL)*, pages 131-138, 1988.

[20] Claire Cardie, Janyce Wiebe, Theresa Wilson, and Diane Litman. Combining low-level and summary representations of opinions for multi-perspective question answering. In *Proceedings of the AAAI Spring Symposium on New Directions in Question Answering*, pages 20-27, 2003.

[21] Sanjiv Das and Mike Chen. Yahoo! for Amazon: Extracting market sentiment from stock message boards. In *Proceedings of the Asia Pacific Finance Association Annual Conference (APFA)*, 2001.

[22] Kushal Dave, Steve Lawrence, and David M. Pennock. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proceedings of WWW*, pages 519-528, 2003.

[23] Luca Dini and Giampaolo Mazzini. Opinion classification through information extraction. In *Proceedings of the Conference on Data Mining Methods and Databases for Engineering, Finance and Other Fields (Data Mining)*, pages 299-310, 2002.

[24] Hugo Liu, Henry Lieberman, and Ted Selker. A model of textual affect sensing using real-world knowledge. In *Proceedings of Intelligent User Interfaces (IUI)*, pages 125-132, 2003.

[25] Satoshi Morinaga, Kenji Yamanishi, Kenji Tateishi, and Toshikazu Fukushima. Mining product reputations on the web. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 341-349, 2002. Industry track

[26] Tetsuya Nasukawa and Jeonghee Yi. Sentiment analysis: Capturing favorability using natural language processing. In *Proceedings of the Conference on Knowledge Capture (K-CAP)*, 2003.

[27] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 79-86, 2002.

[28] Kenji Tateishi, Yoshihide Ishiguro, and Toshikazu Fukushima. Opinion information retrieval from the Internet. *Information Processing Society of Japan (IPSJ) SIG Notes*, 2001(69(20010716)):75-82, 2001. Also cited as “A reputation search engine that gathers people’s opinions from the Internet”, IPSJ Technical Report NL-14411. In Japanese.

[29] Richard M. Tong. An operational system for detecting and tracking opinions in on-line discussion. In *Proceedings of the Workshop on Operational Text Classification (OTC)*, 2001.

[30] Peter Turney. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the Association for Computational Linguistics (ACL)*, pages 417-424, 2002.

[31] Janyce Wiebe, Eric Breck, Christopher Buckley, Claire Cardie, Paul Davis, Bruce Fraser, Diane Litman, David Pierce, Ellen Riloff, Theresa Wilson, David Day, and Mark Maybury. Recognizing and organizing opinions expressed in the world press. In *Proceedings of the AAIL Spring Symposium on New Directions in Question Answering*, 2003.

[32] Hong Yu and Vasileios Hatzivassiloglou. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2003.

[33] Andrea Esuli and Fabrizio Sebastiani. Determining term subjectivity and term orientation for opinion mining. In *Proceedings of the European Chapter of the Association for Computational Linguistics (EACL)*, 2006.

[34] Andrea Esuli and Fabrizio Sebastiani. Determining the semantic orientation of terms through gloss analysis. In *Proceedings of the ACM SIGIR Conference on Information and Knowledge Management (CIKM)*, 2005.

[35] Andrea Esuli and Fabrizio Sebastiani. SentiWordNet : A publicly available lexical resource for opinion mining. In *Proceedings of Language Resources and Evaluation (LREC)*, 2006.

[36] Andrea Esuli and Fabrizio Sebastiani. Pageranking wordnet synsets: An application to opinion mining. In *Proceedings of the Association for Computational Linguistics (ACL)*, 2007.

[37] Soo-Min Kim and Eduard Hovy. Determining the sentiment of opinions. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, 2004.

[38] Soo-Min Kim and Eduard Hovy. Automatic detection of opinion bearing words and sentences. In *Companion Volume to the Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP)*, 2005.

[39] Soo-Min Kim and Eduard Hovy. Automatic identification of pro and con reasons in online reviews. In *Proceedings of the COLING/ACL Main Conference Poster Sessions*, pages 483-490, 2006.

[40] Yejin Choi, Claire Cardie, Ellen Riloff, and Siddharth Patwardhan. Identifying sources of opinions with conditional random fields and extraction patterns. In *Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, 2005.

[41] Steven Bethard, Hong Yu, Ashley Thornton, Vasileios Hatzivassiloglou, and Dan Jurafsky. Automatic extraction of opinion propositions and their holders. In

Proceedings of the AAAI Spring Symposium on Exploring Attitude and Affect in Text, 2004.

[42] Nozomi Kobayashi, Kentaro Inui, and Yuji Matsumoto. Extracting Aspect-Evaluation and Aspect-of Relations in Opinion Mining. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP-CoNLL-2007)*, 2007.

[43] Kamal Nigam and Matthew Hurst. Towards a Robust Metric of Polarity. In Shanahan, J., Qu, J. & Wiebe, J. (Eds.). *Computing Attitude and Affect in Text: Theory and Applications*. Springer, Dordrecht, The Netherlands. 2006.

[44] Michael Gamon, Anthony Aue, Simon Corston-Oliver, and Eric Ringger. Pulse: Mining customer opinions from free text. In *Proceedings of the International Symposium on Intelligent Data Analysis (IDA)*, number 3646 in Lecture Notes in Computer Science, pages 121-132, 2005.

[45] Bing Liu, Mingqing Hu, and Junsheng Cheng. Opinion observer: Analyzing and comparing opinions on the web. In *Proceedings of WWW*, 2005.

[46] Ana Maria Popescu and Oren Etzioni. Extracting product features and opinions from reviews. In *Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, 2005.

[47] Soo-Min Kim and Eduard Hovy. Identifying and analyzing judgment opinions. In *Proceedings of the Joint Human Language Technology/North American Chapter of the ACL Conference (HLT-NAACL)*, 2006.

[48] Nozomi Kobayashi, Kentaro Inui, and Yuji Matsumoto. Opinion Mining from Web documents: Extraction and Structurization. *Journal of Japanese society for artificial intelligence*, Vol.22 No.2, special issue on data mining and statistical science, pages 227--238, 2007.

[49] Wei-Hao Lin and Alexander Hauptmann. Are these documents written from different perspectives? A test of different perspectives based on statistical distribution divergence. In *Proceedings of the International Conference on Computational Linguistics (COLING)/Proceedings of the Association for Computational Linguistics (ACL)*, pages 1057-1064, Sydney, Australia, July 2006. Association for Computational Linguistics.

[50] Soo-Min Kim and Eduard Hovy. Identifying and analyzing judgment opinions. In *Proceedings of the Joint Human Language Technology/North American Chapter of the ACL Conference (HLT-NAACL)*, 2006.

[51] Gilad Mishne and Maarten de Rijke. Capturing global mood levels using blog posts. In *AAAI Symposium on Computational Approaches to Analysing Weblogs (AAAI-CAAW)*, pages 145-152, 2006.

[52] Gilad Mishne and Maarten de Rijke. Moodviews: Tools for blog mood analysis. In *AAAI Symposium on Computational Approaches to Analysing Weblogs (AAAI-CAAW)*, pages 153-154, 2006.

[53] Gilly Leshed and Joseph 'Jofish' Kaye.. Understanding how bloggers feel: recognizing affect in blog posts. In *CHI '06 extended abstracts on Human factors in computing systems (CHI EA '06)*. ACM, New York, NY, USA, 1019-1024. DOI=10.1145/1125451.1125646 <http://doi.acm.org/10.1145/1125451.1125646>, 2006

[54] Rada Mihalcea and Hugo Liu, A corpus-based approach to finding happiness, in the AAAI Spring Symposium on Computational Approaches to Weblogs, March 2006

[55] Lada A. Adamic and Natalie Glance. The political blogosphere and the 2004 U.S. election: Divided they blog. In *Proceedings of LinkKDD*, 2005.

[56] Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the Association for Computational Linguistics (ACL)*, pages 115-124, 2005

[57] Soo-Min Kim and Eduard Hovy. Crystal: Analyzing predictive opinions on the web. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2007.

[58] Gilad Mishne and Natalie Glance. Predicting movie sales from blogger sentiment. In *AAAI Symposium on Computational Approaches to Analysing Weblogs (AAAI-CAAW)*, pages 155-158, 2006.

[59] Michael Gamon. Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In *Proceedings of the International Conference on Computational Linguistics (COLING)*, 2004.

- [60] Alistair Kennedy and Diana Inkpen. Sentiment classification of movie reviews using contextual valence shifters. *Computational Intelligence*, 22(2, Special Issue on Sentiment Analysis):110-125, 2006.
- [61] LANGLEY, P. IBA, W. y THOMPSON, K. An analysis of Bayesian classifiers. En: AAAI-92, (1992).
- [62] Landwehr, N., Hall, M., & Frank, E. (2003). Logistic model trees. In Proc 14th European Conference on Machine Learning (pp. 241-252). Springer-Verlag
- [63] FreeLing 2.2 (2011, Junio) <http://nlp.lsi.upc.edu/freeling/>
- [64] Milne, D. and Witten, I.H. An open-source toolkit for mining Wikipedia. Proc. New Zealand Computer Science Research Student Conf. Vol 9. 2009
- [65] KEA (2011, Mayo) <http://www.nzdl.org/Kea/download.html>
- [66] Witten, I.H. and Medelyan, O. and Milne, D. Finding documents and reading them: Semantic metadata extraction, topic browsing and realistic books. 2008
- [67] John Blitzer, Mark Dredze, Fernando Pereira. Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. Association of Computational Linguistics (ACL), 2007.
- [68] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain Adaptation with Multiple Sources. Neural Information Processing Systems (NIPS), 2009
- [69] Hall, M. and Frank, E. and Holmes, G. and Pfahringer, B. and Reutemann, P. and Witten, I.H. The WEKA data mining software: an update. ACM SIGKDD Explorations Newsletter. Vol 11. Number 1. Pages 10–18. 2009
- [70] Porter, M.F. Snowball: A language for stemming algorithms. 2001
- [71] Denecke, K. Are SentiWordNet scores suited for multi-domain sentiment classification?. Digital Information Management, 2009. ICDIM 2009. Fourth International Conference on pag 1-6. IEEE
- [72] Amazon (Enero, 2011) <http://www.amazon.com/>
- [73] Bing Liu. *Web data mining; Exploring hyperlinks, contents, and usage data*, chapter 11: Opinion Mining. Springer, 2006.
- [74] Sanjiv Das and Mike Chen. Yahoo! for Amazon: Extracting market sentiment from stock message boards. In *Proceedings of the Asia Pacific Finance Association Annual Conference (APFA)*, 2001.

[75] Richard M. Tong. An operational system for detecting and tracking opinions in on-line discussion. In *Proceedings of the Workshop on Operational Text Classification (OTC)*, 2001.

[76] RAUDOS 2:Red Interactiva Multiplataforma de Distribución de Contenidos Multimedia (TSI-020302-2010-67))

[77] Hull, D.A. Stemming algorithms: A case study for detailed evaluation. *Journal of the American Society for Information Science*. Vol 47. Number 1. Pages 70-84, 1996