UNIVERSIDAD NACIONAL DE LA PLATA

FACULTAD DE INFORMÁTICA

TESINA DE LICENCIATURA

Titulo: Clasificación de Subjetividad utilizando Técnicas de Aprendizaje Automático
Autores: Juan Manuel Coria
Director: Dra. Claudia Pons
Codirector: Dr. Waldo Hasperué
Carrera: Licenciatura en Informática

Resumen

La clasificación de subjetividad es un ámbito de la minería de texto poco estudiado en el idioma español, y sin embargo sus aplicaciones son extensas. Su estudio permite comprender mejor la semántica de un texto y la intención de su autor, sin mencionar las implicaciones de su uso en la inteligencia de negocios, para identificar las necesidades de los clientes y obtener métricas valiosas a partir de sus críticas. En este trabajo se intenta aplicar técnicas conocidas de análisis de subjetividad en inglés, adaptadas al español, construyendo en el proceso una base de datos y un sistema clasificador de oraciones.

Palabras Claves

Aprendizaje Automático, Máquinas de Vectores de Soporte, Redes Neuronales, Subjetividad, Objetividad, Minería de Texto.

Trabajos Realizados

Se construye una base de datos y un sistema de software capaz de clasificar oraciones como subjetivas u objetivas utilizando un clasificador de máquinas de vectores de soporte y un multiperceptrón. Se compara el rendimiento de los modelos entrenados y se simula la puesta en producción del sistema, utilizando como entrada la misma base de datos.

Conclusiones

Se obtuvieron resultados satisfactorios en ambos modelos, pero la máquina de vectores de soporte alcanzó un mayor rendimiento en general. Sin embargo, el sistema en producción probó ser dependiente de un estimador de subjetividad base, lo que indica que el método planteado puede resultar útil para incrementar la performance de otros clasificadores.

Trabajos Futuros

Se propone como posibles trabajos futuros desarrollar una base de datos con mayor cantidad de oraciones y con textos más actuales; entrenar, evaluar y comparar otros tipos de clasificadores; explorar otras características extraídas del texto; aplicar algún mecanismo de desambiguación semántica.

Fecha de la presentación: Noviembre 2017

Clasificación de Subjetividad utilizando Técnicas de Aprendizaje Automático

Tesina de Licenciatura

Alumnos
Juan Manuel Coria

Directores
Dra. Claudia Pons
Dr. Waldo Hasperué

2017

Facultad de Informática
Universidad Nacional de La Plata

Índice General

Índice de Figuras

Índice de Tablas

Introducción

La gran cantidad de datos generada cada día en el planeta ha aumentado con el paso de los años desde los inicios de Internet, y aún continúa en aumento. Este resulta ser un escenario que favorece el auge de las técnicas de aprendizaje automático en todas sus áreas, pero fundamentalmente, el campo del análisis de texto es uno de los que más se ve afectado, dado que con el crecimiento y la popularidad de sistemas de software a través de la web, el movimiento de datos multimedia y textuales ha alcanzado un volumen inmenso.

En el contexto de los datos en forma de texto crudo, la balanza se ha inclinado a favor de las metodologías estadísticas, cuya eficacia para analizar grandes volúmenes de información ha sido descrita por muchas publicaciones académicas, desde sus inicios en los años 80' y 90'.

Por otro lado, la disciplina del procesamiento del lenguaje natural no es nueva, y ha definido métodos y características de gran relevancia para el análisis que se emplea actualmente, como son el uso de la categoría gramatical de las palabras (part of speech), la Teoría de Estructura Retórica (Rhetorical Structure Theory), extracción de raíces (stemming), eliminado de palabras vacías (stopwords), el uso de conocimiento previo o diccionarios, n-gramas, entre otras.

De esta rama de investigación del lenguaje es de donde surge la disciplina de Minería de Texto, que utiliza algoritmos de preprocesado de texto plano como los mencionados anteriormente, en conjunto con aquellos de la rama del aprendizaje automático, para predecir, clasificar, agrupar y extraer información concisa. La ventaja del uso de técnicas de preprocesado es que reducen la dimensionalidad de los datos originales, permitiendo aprovechar las características de cada algoritmo y obtener resultados más precisos.

En este trabajo en particular se estudiará el uso de estas técnicas para la clasificación de subjetividad en oraciones, a fin de poder diseñar e implementar un sistema de software que permita efectuar este análisis para textos en español.

Los resultados obtenidos en esta experiencia prueban que el modelo planteado puede predecir con un alto grado de confianza la subjetividad de oraciones, aunque su rendimiento depende en cierta forma de un estimador base predefinido, que como se verá, permite también que el modelo sea utilizado para mejorar la performance de otros.

Finalmente, se detalla a continuación la estructura del presente trabajo, junto con una breve descripción de cada uno de los tópicos a cubrir:

Capítulo 1: Se definen conceptos y se especifican aspectos generales a tener en cuenta respecto del análisis de subjetividad.
Capítulo 2: Se introducen las principales y más populares técnicas de extracción de información y características del texto, haciendo hincapié en su funcionamiento, ventajas y desventajas, considerando su nivel de impacto en la subjetividad de una oración.
Capítulo 3: Se introducen los algoritmos de aprendizaje automático a utilizar en el trabajo, haciendo hincapié en su funcionamiento, ventajas y desventajas, considerando particularmente detalles de parametrización y costo computacional.
Capítulo 4: Se explora el estado del arte en clasificación de subjetividad, describiendo las experiencias realizadas en trabajos previos en el área, sin distinción de idiomas.
Capítulo 5: Se propone el diseño de un sistema de software clasificador de oraciones respecto de su subjetividad, y se evalúan consideraciones, ventajas y desventajas de la implementación propuesta. Se especifica asimismo el lenguaje de programación y bibliotecas de software a utilizar.
Capítulo 6: Se detalla el proceso de construcción y las decisiones de diseño tomadas en cuenta para la construcción de una base de datos capaz de entrenar y evaluar el modelo propuesto.
Capítulo 7: Se describen los resultados obtenidos, especificando métricas y comparando las soluciones planteadas.
Capítulo 8: Se finaliza con conclusiones sobre el trabajo realizado y en base a los resultados obtenidos. Se exploran posibles trabajos futuros sobre la temática abordada.

Por último, se citan las referencias bibliográficas.

Capítulo 1

Conceptos y Definiciones

En este capítulo, se introducen conceptos y definiciones a tener en cuenta en el análisis de subjetividad y se especifican las decisiones tomadas en este trabajo en función de los mismos.

Clasificación de Subjetividad

Existen varios tipos de análisis relacionados con la clasificación de subjetividad, por lo que en esta sección se intentará desambiguar conceptos y definir el ámbito de este trabajo.

El enfoque que se toma es el de la clasificación de grano fino, es decir, se intenta detectar la subjetividad a nivel de oraciones, donde el objetivo es clasificarlas de acuerdo a una de dos etiquetas: *subjetiva* u *objetiva*, por lo que la base de datos a construir y utilizar estará etiquetada acorde a esto.

Por otro lado, existen otros tipos de análisis de texto cuyos fundamentos son similares a la clasificación de subjetividad, pero que no deben confundirse con ella. Se trata del *Análisis de Sentimiento* y la *Minería de Opiniones*.

El análisis de sentimiento consiste en la detección de emociones, ya sea positivas o negativas, usualmente clasificadas en niveles de acuerdo a alguna escala preestablecida. Dentro de este ámbito se encuentra lo que se conoce como minería de opiniones, que busca opiniones del sujeto respecto a un objeto o conjunto de objetos en particular, con el fin de detectar sentimientos generalizados respecto de los mismos.

Subjetividad y Objetividad

Si bien estos conceptos resultan inmediatamente familiares, muchas veces prueban ser difíciles de definir, sobre todo en diferentes ámbitos académicos. Sin embargo, antes de abordar su estudio y análisis, será necesario especificar un criterio para decidir si una oración es subjetiva o no, y para ello se debe contar con una definición concreta del término.

En principio, el diccionario de la Real Academia Española define *subjetivo* como:

1. Perteneciente o relativo al sujeto, considerando en oposición al mundo externo.
2. Perteneciente o relativo al modo de pensar o de sentir del sujeto, y no al objeto en sí mismo.
3. (*Gramática*) Perteneciente o relativo al sujeto o al agente.

A pesar de que estas definiciones parecen ser muy generales, se acercan a la idea que se quiere representar cuando se habla de análisis de subjetividad en una oración. La definición (2) es la única que indica que un sujeto piensa y siente, es decir, que es humano, y que esto impacta en el concepto de subjetividad; por esta razón es que resultará de mayor utilidad, dado que (1) es demasiado general, y (3) se refiere al ámbito de la gramática del lenguaje.

Por otra parte, también existen definiciones armadas en el contexto del procesamiento del lenguaje natural, particularmente aquellas mencionadas en los trabajos de Wiebe [ref. 6] y Liu [ref. 9].

Liu define la subjetividad en el contexto de una oración y en contraste con la objetividad:

Original

> "An objective sentence expresses some factual information about the world, while a subjective sentence expresses some personal feelings or beliefs."

Traducción

> "Una oración objetiva expresa información factual acerca del mundo, mientras que una oración subjetiva expresa sentimientos personales o creencias."

Una vez más, aparece el concepto de sentimiento, y se le agregan las creencias del sujeto; sin embargo, ahora se obtiene además una definición explícita de lo que no es la subjetividad, es decir la objetividad, que expresa información factual, o dicho de otra manera, no se encuentra contaminada por el pensar y/o sentir del sujeto.

En contraste, Wiebe utiliza una definición teniendo en cuenta los puntos de vista y los estados privados del sujeto:

Original

> "[...] we shall call sentences that take a character's psychological point of view subjective, in contrast to sentences that objectively narrate events or describe the fictional world. Subjective sentences present private states of characters - states of an experiencer holding an attitude, optionally towards an object."

Traducción

> "[...] llamaremos a las oraciones que toman el punto de vista psicológico de un personaje subjetivas, en contraste a oraciones que objetivamente narran eventos o describen el mundo de ficción. Las oraciones subjetivas presentan estados privados de los personajes - estados de aquel que experiencia algo sosteniendo una actitud, opcionalmente en relación a un objeto."

En síntesis, una oración se considera subjetiva bajo esta definición si expresa un punto de vista psicológico de un sujeto, y presenta estados privados de sujetos.

Nótese que [ref. 6] define la subjetividad en el contexto del análisis de puntos de vista en textos narrativos, por lo que no todas las características en esta definición resultan de utilidad para el ámbito que concierne a este trabajo. Sin embargo, se puede destacar de ella la importancia del punto de vista y los estados privados del sujeto. Dichos estados se definen de acuerdo a tres categorías:

- Estados intelectuales, como creencias, dudas o conocimiento
- Estados emotivos, como odio, miedo o amor
- Estados perceptivos, como la acción de ver o escuchar algo

Partiendo de esta categorización, se observa fácilmente que estos estados privados representan sentimientos y pensamientos, con un nivel de detalle mayor que en otras definiciones, a lo que se agregan los puntos de vista del sujeto.

Finalmente, teniendo en cuenta las observaciones anteriores, se consideran para este trabajo las siguientes definiciones:

- Se definirá a una oración como *subjetiva* si expresa algún tipo de pensamiento, creencia, sentimiento o percepción relativa al sujeto.
- Se definirá a una oración como objetiva si expresa información factual acerca del mundo,

En este marco, una oración como la siguiente sería clasificada como subjetiva:

Un enorme oso atacó ferozmente al pobre e indefenso leñador mientras dormía

mientras que una como la siguiente sería clasificada como objetiva:

Un oso atacó al leñador mientras dormía

Resumen

El problema de detección de subjetividad, que no debe ser confundido con minería de opiniones o análisis de sentimiento, busca determinar si un fragmento de texto es subjetivo u objetivo. En este trabajo se trabaja a nivel de oraciones y se determina que una oración es subjetiva si expresa algún tipo de pensamiento, creencia, sentimiento o percepción relativa al sujeto, en oposición a una objetiva, que expresa información factual.

Capítulo 2

Extracción de Características

Considerando que la gran mayoría de los algoritmos de minería de texto necesitan recibir datos de entrada en forma de vectores de características numéricos, es necesario que las oraciones que se desean clasificar sean transformadas mediante algún método que permita realizar esta conversión. Aquí es donde se presentan los algoritmos de extracción de características o de preprocesado de texto, cuya tarea consiste en convertir texto crudo en datos numéricos interpretables por los clasificadores.
La mayoría de estos algoritmos se basan en teorías y técnicas conocidas del procesamiento de lenguaje natural, que indican la forma de obtener métricas precisas acerca de la información lingüística contenida en el texto, como por ejemplo la función gramatical de las palabras, o la relación entre las mismas.

En este capítulo se introducirán los algoritmos más utilizados para la extracción de características de textos y se analizarán en función de su relevancia para la detección de subjetividad. Las explicaciones de términos y conceptos presentes en este capítulo están basadas en el trabajo hecho en [ref. 7].

Tokenización

Previo a la obtención de características de un texto, debe obtenerse una estructura computacional adecuada del mismo para su posterior preprocesado; una de las técnicas más utilizadas para esto se denomina tokenización, y consiste en la subdivisión del texto en *tokens*. La forma de efectuar dicha subdivisión puede variar dependiendo del idioma en el que esté escrito el documento y la granularidad con la que se desee analizarlo. Por ejemplo, podría dividirse un documento en caracteres, palabras, o combinaciones de palabras (n-gramas).

En general, la forma más utilizada de tokenización es la de subdivisión en palabras, principalmente porque ellas son las componentes atómicas de significado en componentes más complejos, como oraciones, párrafos o documentos enteros, y permiten obtener características de grano fino de la información que se quiere comunicar. Aunque esta es una solución que funciona muy bien con idiomas como el español o el inglés, puede fallar o resultar inútil en idiomas como el japonés o chino, donde no existe o no se puede distinguir de forma sintáctica la separación de palabras. Incluso en idiomas como el inglés o el francés, pueden presentarse casos ambiguos, en los que la separación no está clara.

Existen contracciones en inglés que pueden ser divididas en palabras:

I'd like a coffee → [I, d (would), like, a, coffee]

pero dicha división no tiene sentido en los indicadores de posesión:

The professor's book → [The, professor's, book]

Además, muchas veces se utiliza una misma contracción para denotar significados diferentes:

I would like a coffee → I'd like a coffee
I had been there → I'd been there

En este caso, la primera contracción se deriva de *I* + *would*, mientras que la segunda se deriva de *I* + *had*, y a pesar de esto, la separación sería idéntica, por lo que se requeriría un estudio más detallado del funcionamiento de estas componentes utilizando mecanismos de desambiguación.
Por otro lado, idiomas como el francés suelen usar el guión para unir palabras donde una división puede tener sentido:

Quel temps fait-il? → [Quel, temps, fait, il]

Pero existen situaciones donde la separación sobre el guión tiene impacto sobre la semántica:

Est-ce que tu es professeur? → [Est-ce, que, tu, es, professeur]

La separación entre *est* y *ce* no tiene sentido, pues *est-ce* es un término idiomático utilizado en la formación de preguntas, *est* es la conjugación en tercera persona del verbo *Être* (ser/estar), y *ce* (eso/esto) es un artículo determinante masculino. De hecho podría tomarse *est-ce que* como un token entero, dado que su presencia no aporta nada en la semántica de la oración, sólo es utilizado como una fórmula que introduce una pregunta.

En el caso particular del español, no se utilizan caracteres especiales para construir contracciones, sino que éstas se consolidan como nuevas palabras (del, al); en consecuencia, es seguro utilizar palabras como tokens, separadas por el carácter " " (espacio).

Un último caso a considerar respecto de la tokenización, es el de los textos donde aparecen fragmentos en otro idioma. Una opción que se ha utilizado para resolver estas situaciones, principalmente en el área de recuperación de información, donde las búsquedas pueden

contener fragmentos en otros idiomas (por ejemplo en la búsqueda en español de una canción en inglés), es el de la utilización de clasificadores para reconocer el idioma principal de un texto para separarlo en tokens de forma acorde. Sin embargo, en este trabajo no se tomarán en cuenta estos casos.

Stopwords

Resulta evidente que ciertas palabras no aportan demasiada información a la idea que un texto expresa, palabras como artículos o preposiciones. En consecuencia, muchas veces estas son eliminadas en la etapa de preprocesado de los datos. Dichas palabras son denominadas *palabras vacías o stopwords*.

Si bien una ventaja inmediata que se puede apreciar en el uso de esta técnica es la reducción de dimensionalidad en la representación de los datos, a veces puede impactar negativamente en la performance del sistema.
Tal como sucede con la tokenización, se deben tener en cuenta múltiples factores para determinar si es conveniente o no eliminar stopwords del texto de entrada; en particular, se encontrarán casos donde el uso de estas palabras influye en la connotación de una frase u oración, o donde eliminarlas simplemente deja muy poca (o ninguna) información para analizar.

La efectividad del uso de listas de palabras vacías *o stop lists*, es un punto de debate en el área de minería de texto. Se han llevado a cabo trabajos de investigación con el fin de comparar la performance de algoritmos conocidos con y sin el uso de stop lists, o limitando su participación. En particular, se ha encontrado que el uso de listas preestablecidas suele dañar considerablemente la precisión del sistema, razón por la cual, de usarse una lista, esta suele ser generada automáticamente a partir de métricas obtenidas del texto, como la frecuencia de aparición de cada palabra.

Por otro lado, existen oraciones en español que contienen mucha información y usan palabras muy comunes, que pueden ser consideradas stopwords. Utilizando la *Tabla 2.1* como un fragmento de una posible stop list, considérense las siguientes oraciones:

lo que dijo ayer es muy bajo para él
(*tokenización*) → [lo, que, dijo, ayer, es, muy, bajo, para, él]
(*filtrado de stopwords*) → [dijo, ayer, él]

dijo ayer que él es muy bajo
(*tokenización*) → [dijo, ayer, que, él, es, muy, bajo]
(*filtrado de stopwords*) → [dijo, ayer, él]

Puede verse fácilmente como las palabras que antes parecían irrelevantes, hacen que la oración misma adquiera un significado distinto, puesto que mediante el mismo filtrado, diferentes oraciones terminan reduciéndose a [dijo, ayer, él]. Además, puede verse que la primera oración tiene una connotación subjetiva, porque expone un punto de vista del sujeto, mientras que la segunda expone información factual, por lo que es claramente objetiva; sin embargo, al ser reducidas a la misma estructura, ambas serían asignadas a la misma clase.

Estos problemas empeoran cuando frases enteras están compuestas únicamente por palabras vacías:

Lo que tengo es mío y tuyo
(*tokenización*) → [Lo, que, tengo, es, mío, y, tuyo]
(*filtrado de stopwords*) → []

En este caso particular, el filtro elimina completamente toda la información de la frase.

un	desde	cierto	algunos	vaya	arriba
una	conseguir	ciertos	algunas	ha	encima
unas	consigo	cierta	ser	tener	usar
unos	consigue	que	es	tengo	uso
uno	consigues	su	soy	tiene	usas
sobre	mio	intento	eres	cuando	usa
todo	tuyo	intenta	somos	muy	usamos
también	ir	donde	sois	tienen	valor
tras	voy	aquí	estoy	el	usan
otro	va	era	esta	la	eso
algún	vamos	y	bien	lo	empleo
alguno	vais	dos	estais	las	modo
alguna	van	bajo	estan	los	para

Tabla 2.1: Fragmento de una posible Stop List

Stemming y Lematización

Los conceptos de *stemming* y *lematización* suelen ser utilizados como sinónimos, sin embargo, a pesar de que ambas técnicas intentan resolver el mismo problema, existen características sutiles que marcan la diferencia entre ellas.

En principio, la problemática que atacan estas técnicas es fundamentalmente la misma: cómo identificar conceptos semánticos similares a pesar de que estos sean denotados por representaciones sintácticas diferentes; es decir, dado un texto, cómo identificar conceptos similares expresados por palabras sintácticamente diferentes. Considérese la siguiente oración:

El cocinero pudo cocinar todo lo que la cocinera no había cocinado previamente

Puede notarse a simple vista que esta oración habla de la cocina, dado que la familia de términos relacionados con este concepto tiene una fuerte presencia en ella, en particular los términos *cocinero, cocinar, cocinera* y *cocinado*, que son variaciones sintácticas utilizadas en español para denotar el concepto semántico general de la cocina, aunque con pequeñas diferencias que hacen a la riqueza de expresión del lenguaje natural. Sin embargo, el análisis de subjetividad, entre otras ramas del análisis de texto, como la recuperación de información, la síntesis automática de documentos o el análisis de sentimiento, no precisa la presencia de estas diferencias sintácticas; más aún, estas hacen ruido en la información. En estos casos es donde se hace necesario aplicar técnicas de stemming o lematización.
Como se mostrará a continuación, estos algoritmos se encargan de agrupar palabras distintas que representen un mismo concepto y convertirlas en palabras idénticas, que pueden o no tener un significado para el idioma en el que se estén aplicando.[1]

Los métodos de stemming son algoritmos simples que eliminan las últimas letras de las palabras utilizando reglas predefinidas, esperando obtener resultados correctos la mayoría de las veces. En contraste, la lematización consiste en identificar inteligentemente conceptos en las palabras, basándose en estudios morfológicos y gramaticales del idioma para el que la implementación es diseñada. En consecuencia, los algoritmos de stemming son más eficientes, mientras que los lematizadores son más precisos; sin embargo, la precisión que se gana con estos últimos no es demasiada, por lo que la balanza se ha inclinado a favor del stemming.

Teniendo en cuenta la comparación anterior, se considera de mayor utilidad práctica para el análisis de subjetividad la técnica de stemming, de forma que se hará hincapié en los algoritmos basados en ella.

[1] Nótese que no puede aplicarse un mismo algoritmo de stemming o lematización en textos de distintos idiomas, puesto que estos son dependientes del mismo, y producirían resultados incoherentes o simplemente inútiles.

Existen distintos algoritmos de stemming, y algunos han sido adaptados e implementados para diferentes idiomas. Se presentarán aquellos más famosos a lo largo de la historia y con mayor aceptación de la comunidad académica, y posteriormente aquellos disponibles para el español:

Stemmer de Lovins (1968)

Diseñado para el inglés, fue el primer algoritmo de stemming publicado. Dada su extensa lista de sufijos y terminaciones, puede completar el proceso en sólo dos etapas, lo que hace que sea más rápido que otros stemmers, como el de Porter, que requiere ocho etapas. Este stemmer intercambia performance por espacio, dado que contempla 294 terminaciones, 29 condiciones y 35 reglas de transformación

Stemmer de Porter (1980)

Diseñado para el inglés, consta de ocho etapas que aplican sucesivamente distintas reglas basadas en las terminaciones de las palabras en relación a las consonantes y vocales presentes en ellas. Es el más utilizado actualmente, y ha sufrido varias modificaciones con el paso del tiempo, incluso se ha desarrollado una versión casi completamente nueva denominada Porter2.

Stemmer de Paice-Husk (1990)

Diseñado para el inglés, es el algoritmo más agresivo, y es capaz de generar términos casi irreconocibles respecto de los originales.

Algoritmo de stemming para español de Snowball

Snowball es un lenguaje creado para la definición e implementación de stemmers, y ha sido utilizado para definir una gran cantidad de los mismos en diferentes idiomas[2], entre ellos se encuentra el que se define en esta subsección para el español.

El algoritmo está compuesto por cuatro pasos:

- Paso 0: Elimina los sufijos con referencias pronominales: me, se, sela, selo, selas, selos, la, le, lo, las, les, los, nos.
- Paso 1: Elimina sufijos estándares.
- Paso 2: Se divide en dos pasos dependientes entre sí. El paso 2b se ejecuta sólo si el paso 2a termina sin eliminar sufijos.
- Paso 3: Elimina sufijos residuales.

[2] Se han implementado stemmers para español, francés, finlandés, noruego, sueco, ruso, italiano y portugués, entre otros.

En cada uno de los anteriores, el descarte de los sufijos está sujeto al cumplimiento de un conjunto de reglas predefinidas y de ciertas condiciones que varían en cada paso. Utilizando este algoritmo, la oración presentada anteriormente se reduce a:

El cociner pud cocin tod lo que la cociner no hab cocin previ

Es interesante destacar que *cocinero* y *cocinera* son reducidos a la raíz *cociner*, mientras que *cocinar* y *cocinado* son reducidos a *cocin*, lo que marca la sutil diferencia gramatical de que los primeros son sustantivos y los últimos son verbos.

Etiquetado Gramatical

El etiquetado gramatical, también conocido como *Part of Speech Tagging* o simplemente *POS Tagging* o *POST*, es una técnica de procesamiento de lenguaje natural que consiste en la construcción de una estructura computacional[3] con información de la categoría gramatical de cada palabra de un fragmento de texto. Considérese la siguiente oración:

Hemos intentado todo, pero los doctores no han podido salvar al paciente

Para aplicar POST a esta oración, se debe tokenizar por palabra, y posteriormente asignar una función gramatical a cada una de ellas, obteniendo finalmente una estructura que soporte dicha información. Entonces:

- *Sea* o *una oración arbitraria*
- *Sea* O *el conjunto de palabras en* o
- *Sea* C *el conjunto de categorías gramaticales disponibles en el idioma de* o
- *Sea* $\leftrightarrows$ *el operador tal que, si* $x \leftrightarrows z$ *entonces* x *'es un/a'* z*, en el sentido de pertenencia a una categoría gramatical, donde* x *es una palabra y* z *es una categoría gramatical*
- *Sea* OP *la lista de pares* $\langle x, c \rangle$*, tal que* $x \in O,\ c \in C \wedge x \leftrightarrows c$

Entonces OP es el resultado de una aplicación de etiquetado gramatical. Particularmente, en el ejemplo definido anteriormente sucede:

o = Hemos intentado todo, pero los doctores no han podido salvar al paciente
O = { Hemos, intentado, todo, pero, los, doctores, no, han, podido, salvar, al, paciente }
C = { Sustantivo, Adjetivo, Artículo, Pronombre, Verbo, Adverbio, Interjección, Preposición, Conjunción, Contracción }

[3] Por ejemplo, un diccionario o un conjunto o lista de pares.

OP = [«Hemos, Verbo», «intentado, Verbo», «todo, Adverbio», «pero, Conjunción», «los, Artículo», «doctores, Sustantivo», «no, Adverbio», «han, Verbo», «podido, Verbo», «salvar, Verbo», «al, Contracción», «paciente, Sustantivo»]

Las ventajas del uso de estos algoritmos en análisis de subjetividad es bastante clara, las oraciones que tienen una fuerte presencia de adjetivos o adverbios tienen mayor probabilidad de ser subjetivas, puesto que atribuyen características a la información que se comunica, y usualmente están ligadas a un pensamiento o creencia de quien narra. Sin embargo, también debe tenerse en cuenta que esta métrica no es suficiente para determinar subjetividad, dado que hay otros factores que influyen en ella, sin mencionar que algunas estructuras gramaticales hacen que la presencia de estos tipos de palabras no tengan ningún efecto en la subjetividad. Este caso se ilustra a continuación en el siguiente ejemplo; considérese la oración:

Caminaba de una forma extravagante

Puede observarse inmediatamente que se trata de una oración subjetiva, el narrador expone su punto de vista en relación a la forma de caminar de un tercero. Ahora obsérvese el efecto que produce la siguiente modificación:

Dijo que caminaba de una forma extravagante

Ahora es claro que la oración es objetiva, el punto de vista del narrador no está presente, sino que describe el de un tercero. Este efecto se produce siempre en la estructura del habla indirecta, donde se presenta información factual sobre la acción de un tercero, que tal vez haya expresado sus pensamientos o creencias, hablando subjetivamente.

Considerando esto, queda claro que un análisis estático de la cantidad de adjetivos y adverbios no es suficiente para determinar la subjetividad de una oración, aunque puede aportar información útil.

TF-IDF (Term Frequency - Inverse Document Frequency)

Esta métrica es un indicador de la relevancia de un término en un documento, en relación a todos los documentos de un corpus, por lo que es muy utilizada, especialmente en el área de recuperación de información, donde es preciso obtener resultados con un puntaje asociado.

Es una métrica que se calcula por cada término, y en relación a otros dos indicadores, la frecuencia con la que aparece el término en el documento (*term frequency*) y la frecuencia inversa con la que este aparece en todos los documentos (*inverse document frequency*).

Term Frequency: esta métrica representa la frecuencia relativa del término en el documento.

$$TF(x) = f_x / n$$

Donde f_x es la frecuencia absoluta del término x en el documento, y n la cantidad de términos en el documento.

Inverse Document Frequency: esta métrica representa el valor inverso a la cantidad de veces que el término aparece en todos los documentos, por lo tanto resalta naturalmente su relevancia.

$$IDF(x) = log_e(N / F_x)$$

Donde N es la cantidad total de documentos y F_x la cantidad de documentos donde aparece el término x.

Finalmente, el indicador TF-IDF se obtiene trivialmente a partir de la siguiente ecuación:

$$TFIDF(x) = TF(x) * IDF(x)$$

El indicador TF-IDF es algo restringido en cuanto al ámbito de su aplicación, dado que se limita a actuar sobre términos en documentos, sobre un corpus de documentos. Esto parecería indicar que no es un valor que pueda aplicarse en la clasificación de subjetividad que se desea realizar en este trabajo, que consta de una base de datos de oraciones; sin embargo, el concepto que identifica TF-IDF resulta interesante en el contexto de este análisis.

Un indicador de este estilo podría proveer al modelo información acerca del impacto de una palabra dada en la subjetividad de la oración, o funcionar como un filtro inteligente de palabras, sin necesidad de utilizar una stop list.

En vista de esto, se decidió utilizar una modificación de TF-IDF que a los efectos de este trabajo se denominará *Frecuencia de Palabras de Subjetividad - Frecuencia Inversa de Oraciones*, o para abreviar, SWF-ISF[4], que utiliza palabras en lugar de términos y pondera su frecuencia de aparición en contextos donde hay subjetividad junto con la aparición en la totalidad de la base de datos.

[4] Por su traducción al inglés: *Subjectivity Word Frequency - Inverse Sentence Frequency*

A continuación se detallan los indicadores utilizados y la forma en que se calculan y combinan.

Subjectivity Word Frequency: es la métrica que representa la frecuencia de aparición de una palabra x en contextos subjetivos.

$$SWF(x) = f_x^s / n^s$$

Donde f_x^s es la cantidad de oraciones subjetivas en las que aparece la palabra x, y n^s la cantidad total de oraciones subjetivas.

Inverse Sentence Frequency: es la métrica que representa la importancia de la palabra para la subjetividad.

$$ISF(x) = log_e(n / f_x)$$

Donde n es la cantidad total de oraciones en la base de datos, y f_x la cantidad de oraciones donde aparece la palabra x. Nótese que no son las mismas variables definidas para TF.

De la misma forma que antes, el nuevo indicador SWF-ISF se obtiene multiplicando ambos valores.

$$SWFISF(x) = SWF(x) * ISF(x)$$

Así, esta métrica será un indicador del impacto de cada palabra en la subjetividad, en relación a la totalidad de las palabras de la base de datos.

Para finalizar, cabe destacar que ISF meramente intercambia con IDF el concepto de documento por el de oración, siendo comparables en este sentido las variables N y n, como también F_x y f_x. En contraste, SWF introduce una pequeña modificación, sus variables son frecuencias de oraciones, en lugar de palabras. Se decidió definirlas de esta forma porque la longitud de las oraciones es demasiado corta en comparación a la de un documento completo, por lo que la frecuencia relativa de una palabra en una oración resultaría ser un valor poco predecible dado que las variaciones en las longitudes de cada oración impactarían fuertemente en la métrica. Para aclarar la problemática planteada, considérese el siguiente ejemplo:

1. *Su cabello era largo y hermoso*
2. *El Nilo es un río largo que se extiende desde el lago Victoria hasta el Mar Mediterráneo*

Es claro que (1) es subjetiva y (2) es objetiva. Ahora, asumiendo también que se dispone de una base de datos de 50 oraciones (25 subjetivas y 25 objetivas) y que estas dos oraciones son las únicas donde aparece la palabra *largo*, pareciera ser que la misma no es un indicador muy confiable de subjetividad, sin embargo, se obtienen los siguientes valores para ambas versiones del indicador:

A. SWF-IDF utilizando frecuencia relativa de la palabra en la oración, donde len_1 es la longitud de la oración (1):

$$len_1 = 6$$

$$SWF("largo") = 1 / len_1 \simeq 0.167$$

$$ISF("largo") = log_e(50 / 2) \simeq 3.219$$

$$SWFISF("largo") \simeq 0.167 * 3.219 \simeq 0.538$$

B. SWF-IDF utilizando frecuencia de oraciones subjetivas que contienen la palabra, sobre el total de oraciones subjetivas:

$$SWF("largo") = 1 / 25 = 0.04$$

$$ISF("largo") = log_e(50 / 2) \simeq 3.219$$

$$SWFISF("largo") \simeq 0.04 * 3.219 \simeq 0.129$$

Puede notarse fácilmente como a pesar de que la palabra elegida tiene la misma presencia tanto en las oraciones subjetivas como las objetivas, la fórmula (A) asigna un puntaje de relevancia considerablemente alto, particularmente más de 4 veces más alto que la fórmula que utiliza la cantidad de oraciones subjetivas. Asimismo, si la longitud de la oración (1) hubiese sido de 25 palabras, entonces los resultados habrían sido idénticos, y si la palabra *largo* hubiese aparecido más de una vez en la oración (1), esto habría elevado el SWF, y en consecuencia también el SWF-ISF final, lo que sólo habría empeorado aún más la situación. Por estas razones es que se considera que (A) es menos confiable, dado que es muy dependiente de la longitud de la oración en cuestión, mientras que (B) mantiene un factor fijo que sólo varía con la cantidad de oraciones subjetivas en la base de datos.

Resumen

Existen diferentes técnicas de preprocesamiento de datos que permiten transformar texto plano en información más precisa, tratable por algoritmos de aprendizaje automático. El concepto de tokenización permite dividir un texto en elementos atómicos de análisis, el

eliminado de stopwords quita elementos que en principio no agregan información, las técnicas de stemming y lematización agrupan conceptos similares que en un análisis exclusivamente sintáctico parecieran no estar relacionados, el etiquetado gramatical agrega información de la función de las palabras dentro de una oración y el TF-IDF asigna puntajes de relevancia a diferentes elementos o términos de acuerdo a los documentos en los que se encuentran. Por otro lado, se utiliza en el trabajo una variación del TF-IDF denominada SWF-ISF, que mide la subjetividad de acuerdo a la información presente en la base de datos.

Capítulo 3

Algoritmos de Aprendizaje Automático Utilizados

En este capítulo se detallan los algoritmos de los clasificadores utilizados en este trabajo: *Support Vector Machine* y *Perceptrón Multicapa*. Se realizará una introducción a su funcionamiento, parámetros disponibles, y costo computacional.

Support Vector Machine

Las máquinas de vectores de soporte son un algoritmo de aprendizaje automático que puede ser utilizado tanto para regresión como para clasificación. Particularmente, en este trabajo se lo utiliza para clasificación con dos clases.[5]
El algoritmo intenta hallar el mejor hiperplano que separe ambas clases. Se considera al mejor hiperplano aquel que satisface que la longitud entre este y los puntos más cercanos de cada clase, también denominados vectores de soporte, sea máxima.

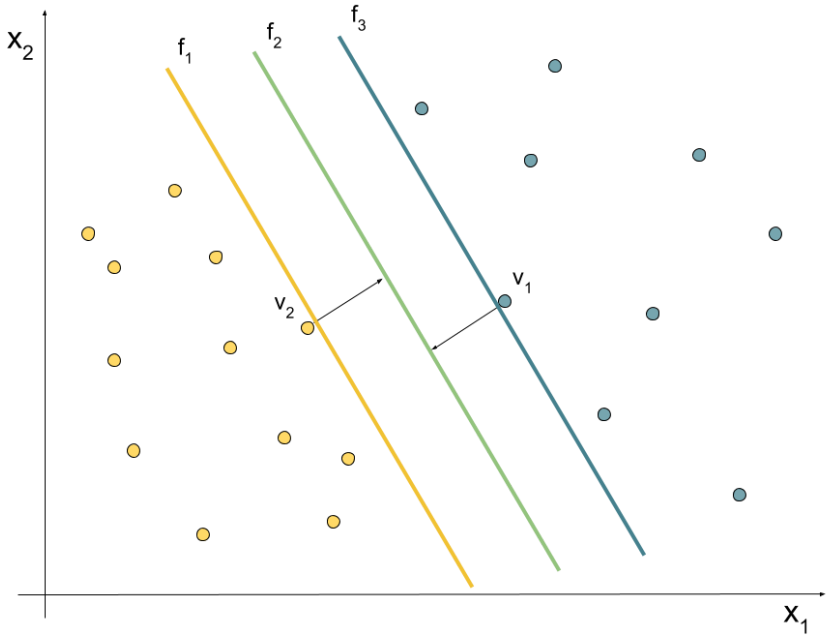

Figura 3.1: Ejemplo de márgenes de un SVM

[5] También puede usarse en problemas de más de dos clases, usando técnicas como *One vs Rest* y *One vs One*.

Como se observa en la *Figura 3.1*, el margen ideal es f_2, porque maximiza la distancia entre los vectores de soporte v_1 y v_2. Por otro lado, determinar dicho margen no es una tarea trivial. Se puede definir cada uno de estos márgenes a través de las siguientes ecuaciones:

$$f_1 : w \,.\, v + b = -1$$

$$f_2 : w \,.\, v + b = 0$$

$$f_3 : w \,.\, v + b = 1$$

Donde w es el vector con los parámetros del hiperplano, junto con b. De esta forma, la distancia que se quiere maximizar es aquella entre los hiperplanos f_1 y f_3, por lo tanto:

$$w \,.\, v_1 + b = 1$$
$$w \,.\, v_2 + b = -1$$

$$\Rightarrow w\,(v_1 - v_2) = 2$$

$$\Rightarrow \frac{w}{||w||}(v_1 - v_2) = \frac{2}{||w||}$$

Entonces, se debe maximizar $\frac{2}{||w||}$ con las restricciones del margen, es decir:

$$y_i(w \,.\, v_i + b) \geq 1 \quad \forall_i \, 1 \leq i \leq n$$

Lo que es igual a minimizar $\frac{1}{2}\,||w||^2$, que a su vez puede expandirse, por lo que equivale a minimizar:

$$W(\alpha) = \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j (v_i \,.\, v_j) - \sum_{i=1}^{n} \alpha_i$$

dado que:

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \;\wedge\; \alpha_i \geq 0 \;\forall_i \, 1 \leq i \leq n$$

De esta manera, visto que $(v_i \,.\, v_j)$ mide en cierta forma la similitud entre dos puntos, puede reemplazarse por una función de v_1 y v_2 para poder tratar problemas no linealmente separables. Esta función es lo que se conoce como *kernel*. Por otro lado, es importante destacar que el valor de α para un punto será más cercano a 0 cuanto más alejado del margen se encuentre dicho punto, lo que permite optimizar el algoritmo para recordar únicamente los vectores más cercanos al margen para cada clase.

Existen además otros parámetros que se utilizan para adecuar el funcionamiento de las máquinas de vectores de soporte, estos son C y $gamma$.

En vista de que el criterio para determinar el margen es demasiado rígido e intenta clasificar perfectamente todos los puntos, el parámetro C define el costo de clasificarlos erróneamente, obteniendo un margen más suave, que puede fallar en algunos ejemplos, pero que da mejores resultados en el conjunto de datos en general y en ejemplos no observados en la etapa de entrenamiento.

Por otro lado, $gamma$ es utilizado como parámetro del kernel RBF[6], e indica el grado de influencia de los vectores de soporte sobre otros, es decir, un valor bajo de este parámetro puede tomar dos puntos muy lejanos como similares, mientras que un valor alto requiere que estos estén muy cerca.

Los SVM suelen funcionar muy bien gracias a la utilización de kernels, que les permite alcanzar un buen rendimiento en la clasificación y una alta performance en problemas de alta dimensionalidad. Aunque por otra parte, no hay un método establecido para elegir kernels ni valores de parámetros de forma confiable, y su utilización en problemas de clasificación con más de dos clases no está tan desarrollado.

Perceptrón Multicapa

El perceptrón multicapa es un tipo de red neuronal particular con una arquitectura *feed forward*, donde existe una capa de neuronas de entrada, una capa de salida y una o múltiples capas intermedias, denominadas capas ocultas. En este tipo de redes, la información se propaga desde la capa de entrada hasta la de salida, en donde se observan los resultados.

Las redes neuronales en general son un algoritmo de aprendizaje automático inspirado en la forma en que se conectan las neuronas del cerebro biológico. En este modelo, una neurona artificial[7] puede interpretarse como se muestra en la *Figura 3.2*. Cada neurona tiene una función de activación f, un conjunto de valores de entrada x_i, un conjunto de pesos w_i y una salida y. Normalmente estas incluyen un valor constante 1 como parte del conjunto x, denominado *bias*, y un peso asociado en el conjunto w, que representan el umbral de activación de la función de la neurona.

[6] Radial Basis Function, o Función de Base Radial
[7] También conocida como *Perceptrón*

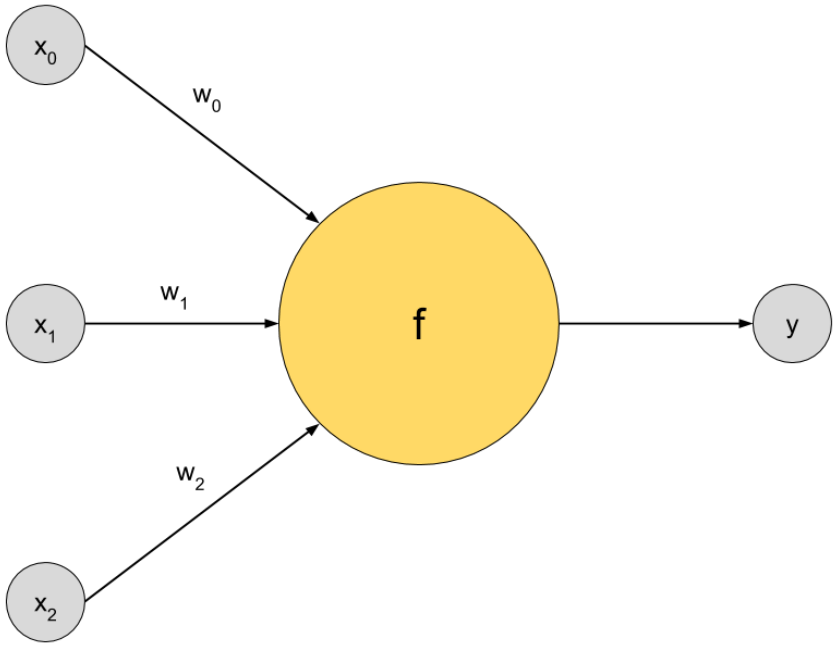

Figura 3.2: Representación visual de un perceptrón

El valor de salida y puede determinarse mediante la siguiente función:

$$y = f(\sum_{i=0}^{n} w_i x_i)$$

A su vez, las funciones de activación suelen ser funciones cuya imagen es [0, 1] cuando la red se utiliza para clasificación, a excepción de la función ReLU (Rectified Linear Unit), que está dada por la siguiente fórmula:

$$f(x) = max(0, x)$$

Así, un *multiperceptrón*, o *perceptrón multicapa* es simplemente un conjunto de perceptrones dispuestos de acuerdo a una arquitectura feed forward preestablecida, donde todas las neuronas de una capa reciben todas las salidas de la capa anterior y producen una salida en base a las mismas. En la etapa de entrenamiento de una red neuronal de este tipo, todos los pesos se inicializan de manera aleatoria, luego se calcula el valor de la salida asociada para cada ejemplo del conjunto de datos de entrenamiento, y posteriormente se calcula su grado de error, minimizando una función de costo definida. Para minimizar esta función se utiliza el algoritmo *gradient descent*, y para encontrar las derivadas parciales de las funciones se utiliza el algoritmo *backpropagation*.[8] Así, se calculan los errores para cada

[8] En aplicaciones concretas suelen utilizarse también variantes de estos algoritmos, a fin de satisfacer necesidades particulares de cada problema, o de mejorar la performance.

neurona de cada capa y se ajustan los pesos de la red para acercar el valor de salida al valor deseado originalmente.

Es evidente entonces que la función de activación elegida para las neuronas, el algoritmo de optimización de la función de costo y la arquitectura de la red, es decir, cuántas capas ocultas y de qué tamaño, son parámetros de estos modelos. Sin embargo, existe otro parámetro fundamental, llamado α, que controla la velocidad de aprendizaje. Este es en realidad un parámetro de la función de optimización, dado que es el valor por el cual se multiplican los pesos para acomodarse de acuerdo a los ejemplos provistos. Es interesante resaltar que un valor de α alto puede reducir el tiempo de entrenamiento, pero también llevar a situaciones de no convergencia, debido a que los pesos se alejan demasiado de los valores ideales. Por el contrario, un valor bajo de α facilita la convergencia pero puede elevar considerablemente los tiempos de entrenamiento.

Uno de los enfoques existentes para solucionar esta problemática es el de utilizar un valor de α dinámico, al cual se asigna un valor alto al inicio del entrenamiento y se disminuye a medida que el modelo se acerca a la convergencia.

La gran ventaja de las redes neuronales es su flexibilidad y performance, pueden utilizarse para problemas de regresión, clasificación e incluso clustering, utilizando redes competitivas. Además, existen diferentes arquitecturas que permiten resolver problemas basados en series o con retroalimentación. Por otro lado, se ha desarrollado una variante de este modelo, denominada *deep neural networks* (redes neuronales profundas), que funcionan extremadamente bien en áreas como visión por computadora, reconocimiento de audio, procesamiento de lenguaje natural, entre otras.

Resumen

El algoritmo de un SVM consiste en encontrar una división óptima entre dos clases, utilizando un método que maximiza la distancia entre los vectores de ambas clases más cercanos. Su parámetro C define la tolerancia del modelo a equivocarse en una predicción, mientras que el parámetro $gamma$ del kernel RBF define qué tan cercanos deben estar dos vectores para considerarse cercanos.

Los perceptrones multicapa son redes neuronales feed forward que tienen una capa de neuronas de entrada, una de salida, y cero o más capas ocultas. Las salidas de una capa sirven como entrada de la siguiente. El algoritmo backpropagation y gradient descent permiten que el modelo pueda aprender a clasificar entradas correctamente, modificando en una medida α los pesos asociados a las entradas de cada neurona. La función de optimización, la arquitectura de la red y la función de activación de las neuronas son a su vez parámetros de este algoritmo.

Capítulo 4

Estado del Arte

En el presente capítulo se introducen trabajos previos en el área de detección de subjetividad. Estos utilizan diferentes métodos y algoritmos, y conforman la base sobre la cual se desarrolla este trabajo.
Para cada uno de ellos se estudian los sistemas propuestos y resultados obtenidos.

Gabriel Murray, Giuseppe Carenini (2010)[ref. 4]

Subjectivity Detection in Spoken and Written Conversations

Este trabajo propone por un lado un análisis de subjetividad supervisado en el contexto de correos electrónicos y transcripciones de reuniones grabadas, y por otro, uno no supervisado, utilizando texto extraído de blogs. Su método toma en cuenta tres posibles clases de subjetividad: subjetivo-positivo, subjetivo-negativo y objetivo.
A su vez, se hace uso de una tokenización por trigramas con distintos niveles de abstracción, donde un elemento de uno de ellos puede ser una palabra específica o bien su etiqueta POS, que identifica su función en la oración, y por lo tanto también en el trigrama. Esto permite encontrar patrones en la combinación de dichas funciones más fácilmente.

Palabra 1	Palabra 2	Palabra 3
really	great	idea
really	great	SUSTANTIVO
really	ADJETIVO	idea
ADVERBIO	great	idea
really	ADJETIVO	SUSTANTIVO
ADVERBIO	great	SUSTANTIVO
ADVERBIO	ADJETIVO	idea
ADVERBIO	ADJETIVO	SUSTANTIVO

Tabla 4.1: Ejemplo de combinaciones de un trigrama

El método seguido en ambos enfoques consiste en clasificar y ordenar los trigramas en base a su probabilidad condicional $P(relevancia / trigrama)$, y de acuerdo a su cantidad de apariciones en textos relevantes e irrelevantes. Por ejemplo, para la obtención de trigramas subjetivos-positivos, el texto relevante sería aquel que es considerado subjetivo-positivo, mientras que el resto es considerado irrelevante.

Tipo de Trigramas a obtener	Relevante	Irrelevante
subjetivo	subjetivo-positivo y subjetivo-negativo	objetivo
subjetivo-positivo	subjetivo-positivo	subjetivo-negativo y objetivo
subjetivo-negativo	subjetivo-negativo	subjetivo-positivo y objetivo
objetivo	objetivo	subjetivo-positivo y subjetivo-negativo

Tabla 4.2: Datos relevantes e irrelevantes para cada tipo de trigrama

Enfoque Supervisado

Utilizando datos previamente anotados, provenientes de correos electrónicos y transcripciones de reuniones, se obtienen patrones de trigramas subjetivos-positivos y subjetivos-negativos, contrastando los documentos de acuerdo al método general descrito anteriormente.

Los datos anotados son una gran ventaja de este método, gracias a su alta confianza en el etiquetado.

Enfoque No Supervisado

Este enfoque presenta una dificultad fundamental, que es la ausencia de datos anotados. Por esta razón se utiliza texto proveniente de blogs, considerado subjetivo por su naturaleza, y texto proveniente de artículos de noticias, considerado naturalmente objetivo. Así, los patrones de trigramas que se obtienen poseen cierto grado de ruido, pero aquellos mejor posicionados en la clasificación obtenida, es decir, aquellos cuya probabilidad condicional es más alta, representan patrones acertados para la clase que se quiere identificar.

Asimismo, se utiliza un conjunto de características crudas y conversacionales para entrenar el modelo. Entre las primeras se destacan algunas como la presencia de los trigramas de mejor clasificación extraídos en la etapa previa, pares de palabras que ocurren en la misma oración, pares de etiquetas POS que ocurren en una misma oración, trigramas de caracteres extraídos del corpus, entre otras. Por otro lado, entre las características conversacionales se incluyen dos tipos de longitudes de oración, dos tipos de posiciones de

la oración en los datos de entrenamiento, tiempo desde el inicio de la conversación hasta que ocurre la oración, indicador de la dominación del autor de la oración en la conversación completa, si el autor inició la conversación, entre otros.

Respecto del modelo, se entrena un clasificador de máxima entropía (MaxEnt Classifier), y se evalúan los resultados utilizando las métricas de precisión, recall, f-score y la curva ROC (Receiver Operator Characteristic).

La parte experimental del trabajo se divide en cuatro etapas:

1. **Detección de Oraciones Subjetivas:** Se intenta discernir entre oraciones puramente subjetivas (positivas o negativas) del resto.
2. **Detección de Oraciones y Preguntas Subjetivas:** Se intenta identificar todas las oraciones y preguntas subjetivas, a fin de comparar los resultados obtenidos con el trabajo realizado en *Raaijmakers (2008)*[ref. 14].
3. **Detección de Oraciones Subjetivas-Positivas:** Se intenta detectar todas las oraciones subjetivas-positivas.
4. **Detección de Oraciones Subjetivas-Negativas:** Se intenta detectar todas las oraciones subjetivas-negativas.

En conclusión, los mejores resultados fueron encontrados en la etapa 2, y decrecieron en cierta medida para las etapas 1, 3 y 4, demostrando entonces que es más fácil detectar oraciones y preguntas subjetivas como un todo, que diferenciandolas en base a características más específicas, como la polaridad.

Bing Liu (2010)[ref. 9]

Sentiment Analysis and Subjectivity

Destinado a aparecer en el libro *Handbook of Natural Language Processing*, este trabajo es esencialmente una recopilación del estado del arte en análisis de subjetividad y sentimiento, basado en los trabajos existentes en la fecha en que fue escrito. Además, utiliza un lenguaje familiar y presenta los conceptos de una forma más comprensible para quienes quieran introducirse en el área.

El documento se encuentra estructurado en seis partes, que introducen ámbitos de aplicación diferentes sobre el análisis de subjetividad y sentimiento. Por tanto, esta sección se estructura de la misma forma.

El Problema del Análisis de Sentimiento

Se definen conceptos y se formaliza el problema, indicando modelos y el marco de trabajo.

Objeto: es una entidad que se representa mediante un par (T, A). T es un árbol de subcomponentes del objeto, cuya raíz es el componente que representa al objeto mismo. A es un conjunto de atributos del objeto. Por ejemplo, el objeto *iPhone* tiene los subcomponentes *batería* y *pantalla*, y los atributos *tamaño, peso,* entre otros.
En la práctica, esta representación se simplifica usando el término de *característica* para hablar tanto de componentes como de atributos, con el fin de facilitar su manipulación e interpretación.[9]

Documento con Opiniones: es un texto que evalúa un conjunto de objetos y que, en el caso más general, está compuesto por una secuencia de oraciones.

Pasaje de Opiniones sobre una Característica: se define como un grupo de oraciones consecutivas en un documento, que sostienen una opinión positiva o negativa sobre una característica de un objeto.

Características Implícitas y Explícitas: una característica es denominada explícita cuando ella misma o uno de sus sinónimos aparece en la oración que se analiza. Por contrario, si ni ella misma ni ninguno de sus sinónimos está presente, pero igualmente se está hablando de esa característica, entonces se la denomina implícita. Por ejemplo, en la oración *la batería dura demasiado poco*, la batería es una característica explícita, y en la oración *el teléfono es demasiado grande*, el tamaño es una característica implícita.

Portador de Opinión: es aquella persona o entidad que expresa su opinión sobre un objeto.

Opinión: es un punto de vista, actitud, emoción o apreciación, ya sea positiva o negativa, sobre una característica de un objeto, y formulada por un portador de opinión.

Orientación o Polaridad de la Opinión: indica si la opinión sobre una característica de un objeto es positiva, negativa o neutral.

En base a estas definiciones, el autor especifica entonces el modelo de un objeto y de un documento con opiniones.
El **modelo de un objeto** o consiste en un conjunto finito de características $F = \{f_1, f_2, \ldots, f_n\}$, que incluye al objeto mismo como una de ellas. Cada una de estas puede ser expresada por un conjunto de palabras o frases $W_i = \{w_{i1}, w_{i2}, \ldots, w_{im}\}$

[9] El término *característica* no es el mismo que se utiliza para denotar las características del *Capítulo 2*, que están íntimamente relacionadas con los algoritmos de aprendizaje automático.

denominados sinónimos de la característica, o indicada por alguno de sus indicadores $I_i = \{ I_{i1}, I_{i2}, \ldots, I_{ir} \}$.
El **modelo de un documento con opiniones** es un conjunto de opiniones $\{ o_1, o_2, \ldots, o_q \}$ de un conjunto de portadores de opinión $\{ h_1, h_2, \ldots, h_p \}$, donde las opiniones sobre cada objeto o_j son expresadas por un subconjunto F_j de sus características. Además, una opinión puede ser de uno de los siguientes tipos:

- Directa: es una tupla $(o_j, f_{jk}, oo_{ijkl}, h_i, t_l)$ donde o_j es un objeto, f_{jk} es una característica de o_j, oo_{ijkl} es la polaridad de la opinión, h_i es el portador de la opinión y t_l es el momento en el que se expresó la opinión.
- Comparativa: la opinión se expresa en términos de las similitudes y/o diferencias entre dos objetos, que poseen características similares.

Finalmente, son definidos los siguientes conceptos que marcan el límite entre la clasificación de subjetividad y el análisis de sentimiento.

Subjetividad de una Oración: una oración objetiva expresa información factual del mundo, mientras que una subjetiva expresa sentimientos personales o creencias. También se resalta en esta definición que una oración subjetiva no necesariamente contiene una opinión, de la misma forma que una oración objetiva tampoco está obligada a no contener opiniones. Por ejemplo, la oración *quería un teléfono con buena calidad de sonido* es subjetiva, y no presenta ninguna opinión, mientras que la oración *los auriculares se rompieron en dos días* es objetiva y contiene una opinión negativa acerca de la calidad de los auriculares.

Opiniones Implícitas y Explícitas: una opinión expresada explícitamente en una oración subjetiva se denomina explícita, mientras que una presente pero no explícita se considera implícita.

Oración con Opiniones: es una oración subjetiva u objetiva que expresa explícita o implícitamente opiniones positivas o negativas.

Clasificación de Sentimiento y Subjetividad
Se analizan los problemas de clasificación de sentimiento a nivel documento, y clasificación de subjetividad y sentimiento a nivel oración, mostrando los diferentes enfoques existentes.

En primer lugar, se realiza un análisis del problema de clasificación de sentimiento en un documento con opiniones. En esta tarea se asume que el documento dado expresa opiniones sobre un único objeto y de un único portador de opinión.
Se estudia entonces esta tarea desde dos puntos de vista, mediante algoritmos de aprendizaje supervisado y no supervisado.

- Enfoque Supervisado: implica detectar la polaridad de la opinión contenida en un documento a través del uso de algoritmos de aprendizaje supervisado, como modelos bayesianos, SVMs, redes neuronales, entre otros. Es importante notar que una parte crucial de este enfoque es construir el conjunto adecuado de características para que el aprendizaje sea óptimo; por ejemplo, se ha encontrado que los adjetivos y los adverbios juegan un rol fundamental en la expresión de opiniones. Algunas de las más utilizadas son los términos y su frecuencia (TF-IDF), etiquetas POS, palabras de opinión y frases base, dependencia sintáctica y negación.

- Enfoque No Supervisado: se detalla el método propuesto en [ref. 15], el cual consta de tres pasos:

    1. Se extraen del texto todas aquellas oraciones que incluyan adjetivos o adverbios, y luego se extraen todas aquellas secuencias de palabras que coincidan con uno de los patrones definidos en la *Tabla 4.3*.
    2. Se estima la polaridad de las frases extraídas usando el indicador de *información mutua puntual* (PMI):

    $$PMI(t_1,\ t_2)\ =\ log_2(\frac{P(t_1 \wedge t_2)}{P(t_1)P(t_2)})$$

    De esta forma, la polaridad de una frase x es calculada de la siguiente manera:

    $$oo(x)\ =\ PMI(x,\ "excellent")\ -\ PMI(x,\ "poor")$$

    3. Se calcula la polaridad promedio de todas las frases en el documento y se lo clasifica de acuerdo al valor obtenido.

Palabra 1	Palabra 2	Palabra 3 (No extraída)
Adjetivo	Sustantivo	-
Adverbio	Adjetivo	No Sustantivo
Adjetivo	Adjetivo	No Sustantivo
Sustantivo	Adjetivo	No Sustantivo
Adverbio	Verbo	-

Tabla 4.3: Patrones de palabras a extraer según el método de Peter D. Turney

Posteriormente, se examina el problema de clasificación de subjetividad y sentimiento por oraciones, es decir, dada una oración s, determinar primero si s es subjetiva, y luego, de ser así, si s expresa una opinión positiva o negativa.

La mayoría de los métodos que siguen este enfoque utiliza algoritmos de aprendizaje supervisado, a veces combinados con métodos no supervisados de extracción de patrones.
En este caso, se asume que una oración expresa una única opinión de un único portador de opinión

Por último, se describen dos formas de obtener palabras y frases de base para encontrar opiniones en textos.
El enfoque basado en diccionarios consiste simplemente en utilizar un pequeño conjunto de palabras conocidas y luego extender dicho conjunto buscando sus sinónimos y antónimos iterativamente en un diccionario dado.
Por otra parte, el enfoque basado en corpus consiste en analizar textos a partir de un conjunto inicial de palabras, y extraer otras a partir de su relación sintáctica con las mismas. Esta técnica se basa en que términos relacionados mediante conectores como *y* tienden a tener la misma polaridad, mientras que aquellos relacionados con otros como *pero* tienden a tener la polaridad opuesta.

Análisis de Sentimiento Basado en Características
Se intenta identificar el objeto sobre el cual una persona da una opinión, y luego se decide si dicha opinión es positiva o negativa.

Muchas veces, resulta necesario conocer una opinión o un sentimiento con una granularidad más fina, que no puede conseguirse a nivel documento o a nivel oración, dado que estos pueden contener diferentes opiniones sobre diferentes objetos, e incluso con diferente polaridad. Para realizar esta tarea, se centra el estudio en las críticas de internet, donde se obtienen varias ventajas inmediatas, como información sobre el portador de opinión, la fecha en la que se escribió una crítica, entre otras. A continuación se describen entonces los métodos utilizados en esta técnica.

Extracción de Características[10]

Puede aplicarse utilizando textos escritos en diferentes formatos. Por un lado, existe el formato de texto libre, donde no se requiere que el autor escriba de ninguna forma en particular, y por otro, el formato de pros y contras, donde el autor de cada crítica debe escribir los objetos a los que evalúa como positivos o negativos, y además escribir un texto libre.

[10] Nótese que el término no es equivalente al de extracción de características tal como se ha utilizado en este trabajo para referirse a los algoritmos de preprocesamiento.

El formato de pros y contras se aplica extrayendo del texto pequeñas secuencias separadas por determinados caracteres como "," o "-", que consisten en pares de palabras con su etiqueta POS. Luego estas secuencias son transformadas en reglas reemplazando los sustantivos que representan características con tokens especiales.

En contraste, el formato de texto libre implica tratar con un mayor grado de ruido en el texto, por lo que es necesario aplicar técnicas más complejas. En este contexto se describe un método de dos pasos, donde primero se buscan las características principales, identificando los sustantivos simples y compuestos con mayor presencia en el texto, y luego, a partir de los resultados obtenidos en la etapa anterior, se extraen características secundarias que siguen los mismos patrones gramaticales que las primarias. Por ejemplo, asumiendo que *pictures* es una característica primaria encontrada en el primer paso:

the pictures are amazing

Dado que *amazing* es un adjetivo con polaridad positiva, se puede extraer el patrón:

<ARTÍCULO> <CARACTERÍSTICA> <VERBO> amazing

De esta forma se podría determinar en la siguiente oración que el sustantivo *software* también es una característica, y la polaridad de la opinión expresada es positiva:

the software is amazing

Una vez terminadas estas etapas, se plantean a su vez algunos problemas posteriores a resolver, como por ejemplo la identificación de características sinónimas, es decir aquellas características que son mencionadas usando diferentes sustantivos, pero que en esencia se refieren a la misma. Otra problemática que se presenta es la de hallar características implícitas en las críticas, por ejemplo, cuando un autor habla de algo *hermoso* o *bello*, está haciendo referencia a la *apariencia*, o cuando habla de algo *grande* o *chico* se refiere a su *tamaño*.

Identificación de la Orientación de las Opiniones

Se describe el algoritmo léxico definido en [ref. 19], con el que se identifica la polaridad de una oración. Este algoritmo se ejecuta en cuatro pasos:

1. Se identifican palabras y frases de opinión, asignando puntajes de polaridad iniciales a cada una de ellas. Se asigna el valor 1 a opiniones positivas, -1 a las negativas y 0 a aquellas dependientes del contexto.

2. Se invierten los valores asignados en la etapa anterior en la presencia de negaciones.
3. Se actualizan una vez más los valores de la etapa anterior teniendo en cuenta cláusulas de conjunción de oposición, introducidas por palabras o frases como *pero*, *sin embargo*, *no obstante*, entre otras. Así, se determina que dichas cláusulas deben tener una polaridad opuesta.
4. Se combinan todos los valores para determinar la polaridad total, utilizando una función de agregación.

Por otro lado, también se describe un conjunto de reglas de opinión que ayudan a determinar la polaridad en ciertas expresiones, por ejemplo: si se habla del decrecimiento de algo negativo, entonces la opinión es positiva, si se niega algo positivo, entonces la opinión es negativa, si se habla de que una característica alcanzó un grado deseado de alguna propiedad entonces la opinión es positiva, entre varias otras.

Análisis de Sentimiento de Oraciones Comparativas
Determina la opinión cuando las oraciones simplemente comparan un objeto o una característica de un objeto con otro.

Este análisis consiste en identificar aquellas oraciones comparativas, donde se expresa que una característica de un objeto es superlativa, o es mejor o peor que otra. En esta sección se hace referencia a algunos métodos existentes sobre esta temática [refs. 20, 21].

El método consiste en detectar adjetivos y adverbios comparativos y superlativos, con el fin de clasificar las oraciones en tres tipos: *graduable no igual*, *igual*, y *superlativo*. Posteriormente se identifican los diferentes objetos y sus características, para lo cual pueden utilizarse modelos como campos aleatorios condicionales o modelos ocultos de Markov, entre otros.

Finalmente, es necesario determinar aquellos objetos que son preferidos en este tipo de oraciones. En este contexto existen dos escenarios que pueden presentarse, y que deben tratarse de distinta manera. Por un lado existen los adjetivos y adverbios comparativos no dependientes del contexto, como *mejor*, *peor*, etc., para los cuales es sencillo determinar la preferencia. Pero por otra parte también podemos encontrar comparativos del tipo *más*, *menos*, etc., para los cuales deben aplicarse las siguientes reglas:

1. <Comparativo Creciente> Negativo → Opinión Comparativa Negativa
2. <Comparativo Creciente> Positivo → Opinión Comparativa Positiva
3. <Comparativo Decreciente> Negativo → Opinión Comparativa Positiva
4. <Comparativo Decreciente> Positivo → Opinión Comparativa Negativa

Sin embargo, también debe tenerse en cuenta el dominio del problema, dado que los comparativos acerca de ciertas características no siempre resultan obvios, principalmente debido a la naturaleza ambigua de dichas características. Por ejemplo:

la cámara x tiene mayor apertura que la cámara y

En este contexto, se debe establecer de antemano si una mayor apertura en el lente de una cámara representa un rasgo deseable de la misma, en caso contrario la orientación de la opinión en esta comparación no puede ser determinada de forma fehaciente.

Búsqueda y Recuperación de Opiniones
Se ataca al problema desde el punto de vista de la búsqueda, es decir, cómo encontrar opiniones sobre un objeto particular dado.

En este enfoque, se consideran dos tipos de búsqueda de opiniones: sobre objetos o características de objetos, como una cámara de fotos o un teléfono celular, y de personas u organizaciones sobre objetos o características de objetos, como por ejemplo la opinión de Obama sobre el aborto.

De esta manera, una búsqueda de opiniones va a constar de dos etapas: búsqueda de documentos y oraciones relevantes, y búsqueda de opiniones en los textos encontrados, determinando su polaridad. La primera etapa se alinea con la búsqueda y recuperación de información tradicional, pero la segunda es propia del análisis de subjetividad y sentimiento.
Asimismo, es importante remarcar que los resultados que se presentan al usuario no pueden estar organizados de la forma tradicional, sino que deben balancearse las opiniones positivas y negativas para reflejar la proporción natural de dichas opiniones. Por otro lado también resulta interesante considerar una organización por características del objeto, aunque puede resultar una tarea muy compleja en la práctica.

Finalmente se describe el funcionamiento del sistema planteado en [ref. 18], que consta de dos componentes:

- **Componente de Recuperación:** además de ejecutar la tarea tradicional de recuperación de documentos, identifica y desambigua conceptos en la búsqueda del usuario, luego amplía la lista de conceptos a buscar con sus sinónimos, y finalmente calcula la relevancia de los documentos en base a la aparición de dichos conceptos y palabras clave.
- **Componente de Clasificación de Opiniones:** separa los documentos en dos clases, dependiendo si contienen opiniones o no, y luego clasifica aquellos con opiniones según su polaridad. Para esto se utiliza un método supervisado con clasificadores SVM, que se entrenan con textos de críticas extraídos de diferentes sitios de internet.

Utilidad y Spam de Opiniones

Se intenta detectar opiniones falsas o fraudulentas, y medir la utilidad de las opiniones identificadas.

Existen otros tipos de problemáticas relacionadas a la detección de opiniones. En primer lugar, visto que la relevancia de las críticas de productos ha cobrado una gran importancia en internet, existen críticas creadas de forma fraudulenta, con el fin de aumentar la popularidad de un producto, o disminuir la del de un competidor. Por esta razón es que se precisan herramientas de identificación de spam de opiniones.

Según los trabajos [refs. 22, 23], existen diferentes tipos de spam de opiniones: el de *tipo 1*, que comprende las críticas con opiniones falsas, ya sea positivas o negativas; el de *tipo 2*, que hace referencia a opiniones acerca de marcas en lugar de productos; y finalmente el de *tipo 3*, que abarca aquellas críticas que en realidad no contienen opiniones, como preguntas o publicidad. En estos trabajos también se define un conjunto de características que pueden ser utilizadas, que se encuentran divididas según a quién o qué hacen referencia: a la crítica en sí, al autor o al producto. Por último, los resultados muestran algunas tendencias interesantes, como por ejemplo que los usuarios con mejor clasificación en los sistemas de críticas tienden a ser autores de spam, o que los productos con menores ventas son los más afectados por este.

Por otra parte, la utilidad de las opiniones también es un factor interesante a tener en cuenta, y generalmente tiende a plantearse como un problema de regresión. Se han realizado varias investigaciones sobre este área, como [refs. 24, 25], que hacen uso de características como TF-IDF de unigramas y bigramas, cantidad de palabras y oraciones, etiquetas POS, entre otros. Particularmente, en [ref. 24] también se analiza la subjetividad de la crítica como una medida de utilidad.

Ahmad Kamal (2013)[ref. 2]

Subjectivity Classification using Machine Learning Techniques for Mining Feature-Opinion Pairs from Web Opinion Sources

El método propuesto en este trabajo consta de dos fases. La primera fase utiliza un clasificador supervisado para determinar si las críticas de clientes a un producto dado son subjetivas u objetivas, mientras que en la segunda fase se analizan aquellas críticas subjetivas de forma semántica y lingüística a través de un método de reglas, a fin de obtener pares *(característica, opinión)* para cada producto.
Dado que el presente trabajo tiene su principal foco en la clasificación de subjetividad, se hará mayor énfasis en la primera fase del método propuesto.

La arquitectura del sistema propuesto se divide en cinco componentes funcionales: un data crawler, un preprocesador de documentos, un analizador de subjetividad/objetividad, un componente de aprendizaje de características y opiniones, y un analizador de viabilidad.

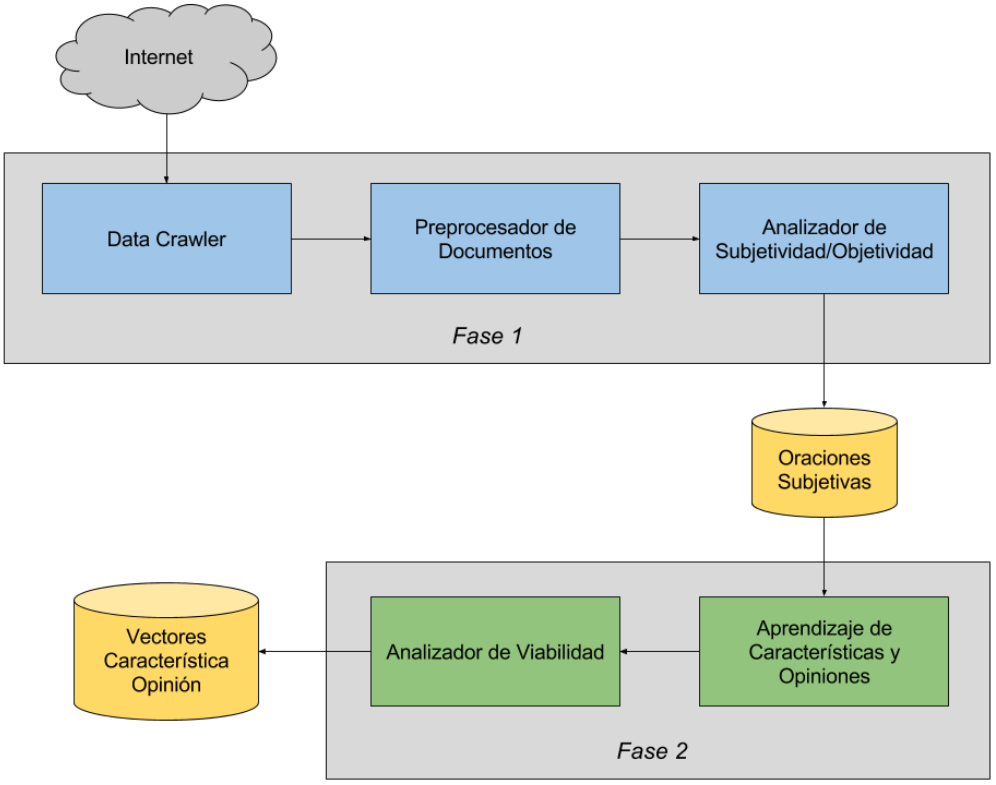

Figura 4.1: Arquitectura simplificada del sistema propuesto por Kamal

Data Crawler y Preprocesador de Documentos

Se extrae del sitio destino el texto asociado a las críticas y se divide en porciones manejables, filtrándolas mediante un análisis semántico basado en POS Tagging

Analizador de Subjetividad/Objetividad

Se construye un vector de características por cada unigrama de una lista, clasificados en subjetivos y objetivos. Luego se los utiliza para entrenar un clasificador Naïve Bayes, que se aplica posteriormente para la clasificación de cada oración, siguiendo un modelo de unigramas.

Las características que se toman en cuenta para construir cada vector son las siguientes:

- *TF-IDF*
- *Posición en la oración*: -1 para el principio, 0 para el medio y 1 para el final
- *POS Tag*

- *Indicador de Opinión*: se indica con 0 o 1 si el unigrama se encuentra presente en una lista de indicadores de opinión preestablecida
- *Negación*: 1 si el unigrama presenta signos de negación o 0 en caso contrario
- *Presencia de Modificador*: 1 si el unigrama es un modificador o 0 en caso contrario (usualmente adverbios)

Componente de Aprendizaje de Características y Opiniones

Se utilizan dos reglas definidas, junto con información extraída de un análisis léxico realizado por el POS Tagger desarrollado en Stanford, para poder determinar en cada crítica los componentes de las tuplas que identifican características y opiniones.

El formato que utiliza el autor para presentar esta información es (f, m, o), donde f es una característica (ej. batería del Samsung Galaxy S4), o es una opinión sobre f (ej. duradera), y m es un modificador de la expresividad de o (ej. muy).

Analizador de Viabilidad

Dado que existe la posibilidad de obtener información ruidosa en el POS Tagging, se calculan métricas para evaluar la confiabilidad de las tuplas obtenidas en la etapa anterior.

Finalmente, la evaluación del clasificador revela que los mejores valores de precisión, recall y f-score promedio sobre los datos de prueba son obtenidos utilizando Naïve Bayes, por lo que se considera a este el clasificador más adecuado en el experimento.

José M. Chenlo, David E. Losada (2013)[ref. 3]

Machine Learning approach for Subjectivity Classification based on Positional and Discourse Features

En este trabajo, se propone un análisis del texto a nivel oración, incluyendo información retórica sobre el mismo, basándose en la teoría de estructura retórica. A su vez, se proponen un conjunto de características que luego son utilizadas en dos modelos, un clasificador SVM y uno de regresión logística. Las características utilizadas para cada oración son:

- Presencia de **unigramas y bigramas** con más de cuatro ocurrencias en todo el documento.
- **Características léxicas de sentimiento**, es decir, la frecuencia y porcentaje de términos con opiniones, interrogaciones, y exclamaciones. Dichos términos fueron extraídos utilizando la herramienta *OpinionFinder*.
- **Características retóricas**. Basándose en la teoría de estructura retórica, se determina la presencia de ciertas relaciones predefinidas entre segmentos de la oración, por ejemplo, las relaciones de causa, explicación, elaboración, entre otras.

El análisis retórico fue implementado mediante el uso de la herramienta SPADE (Sentence-level Parsing of Discourse).

- **Características de longitud**, incluyendo la cantidad de palabras presentes en los diferentes segmentos obtenidos en el análisis retórico.
- **Características posicionales**, incluyendo la posición absoluta de la oración en el documento, y la cantidad de oraciones en el documento del que proviene.

Se decidió trabajar con la versión en inglés de la base de datos NTCIR-7, que contiene noticias de diferentes fuentes y sobre diferentes temáticas. Usando estos datos, se entrenaron los modelos SVM y regresión logística de la biblioteca *liblinear*, obteniendo los mejores clasificadores aplicando validación cruzada con $k = 5$. Por otro lado, teniendo en consideración que la base de datos elegida contiene alrededor de tres veces más oraciones subjetivas que objetivas, se decidió penalizar la clasificación de una oración subjetiva como objetiva, y se diseñó el entrenamiento para maximizar el f-score en relación a la clase subjetiva. Posteriormente se evaluaron dichos modelos contra el clasificador de subjetividad de *OpinionFinder*, que está compuesto por clasificadores bayesianos entrenados con patrones lingüísticos correlacionados con la subjetividad.

En conclusión, los modelos demostraron superar aquellos definidos en *OpinionFinder*, y se encontró que las características retóricas no jugaron un papel esencial por sí mismas en la detección de subjetividad, pero sí probaron ser de mucha utilidad para respaldar y dar fuerza a otras características, mejorando en gran medida los resultados finales.

Reynier Ortega Bueno (2014)[ref. 5]

Método No Supervisado para la Detección de Subjetividad

Este trabajo propone un enfoque no supervisado, orientado a oraciones y para el idioma inglés, que intenta mitigar precisamente la dificultad de los métodos supervisados para detectar el significado de las palabras en el contexto en que se usan, es decir, determinar cuándo una palabra está siendo usada con una connotación subjetiva u objetiva. Para esto, propone un método donde se utilizan varias herramientas externas para desambiguar conceptos e identificar sentidos y subjetividad.

La arquitectura del sistema propuesto, como se observa en la *Figura 4.2*, es simplemente un pipeline de ejecución, que en cada etapa extrae más información acerca de las oraciones, hasta finalmente poder clasificar los textos de entrada en subjetivos u objetivos. A continuación se describe en qué consiste cada una de las fases planteadas.

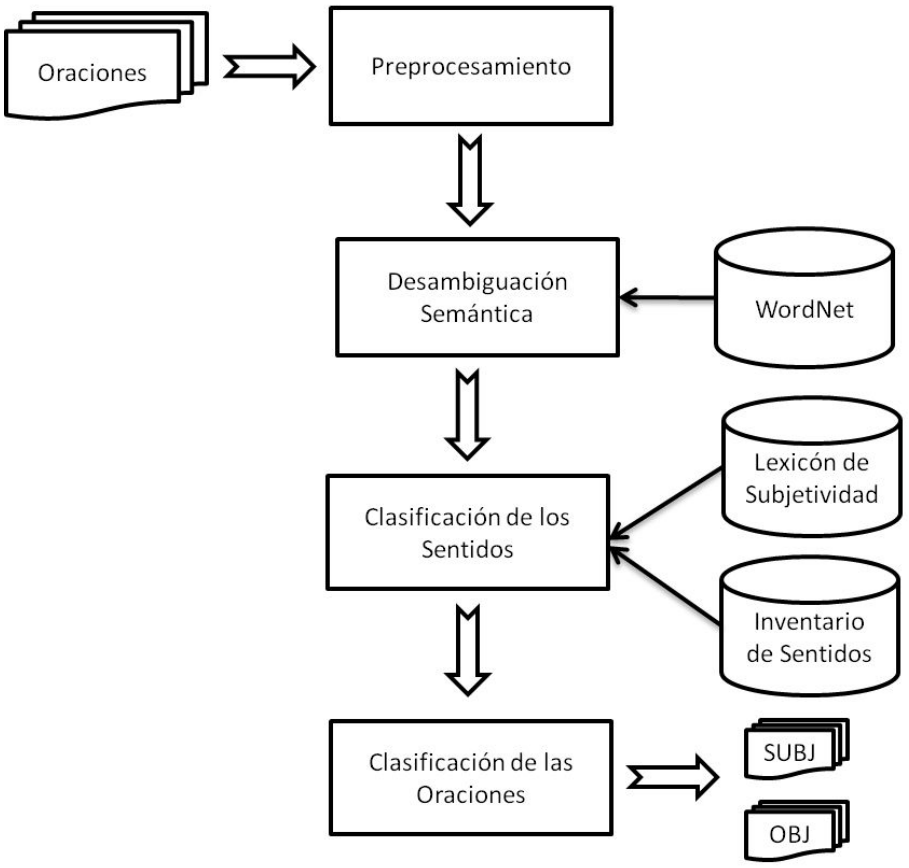

Figura 4.2: Arquitectura general del sistema propuesto por Ortega Bueno

Preprocesamiento

Se utiliza la herramienta *FreeLing* para dividir las oraciones de entrada en palabras, y para cada una de ellas identificar su lema y categoría gramatical. Por último, se eliminan de las oraciones todas las palabras vacías.

Desambiguación Semántica

Existen tres tipos de métodos para realizar esta tarea, los métodos supervisados, no supervisados, y basados en conocimiento, por ejemplo mediante el uso de diccionarios u ontologías. Aunque los enfoques supervisados han alcanzado mejores resultados, estos corren con la desventaja de la dependencia y limitación que conlleva el uso de un corpus anotado, esto es, la necesidad de intervención humana para anotar las oraciones, la dependencia de la fuente, idioma y otras características del corpus utilizado, entre otras.

En este trabajo, se evaluaron tres diferentes métodos existentes de desambiguación semántica:

- El método propuesto por [ref. 26], que consiste en un algoritmo de ranking sobre grafos, donde los vértices son conceptos, y las aristas relaciones entre ellos.
- El método propuesto por [ref. 27], que se basa en la técnica de clustering, agrupando todos los posibles sentidos para las palabras (extraídos de *WordNet*), para finalmente filtrar los grupos obtenidos según cómo se adecúen al contexto.
- El método propuesto por [ref. 28], que es una modificación del algoritmo de Lesk, propuesto en 1986. Este método utiliza la palabra que se quiere desambiguar, y sus n palabras más cercanas para extraer sus significados, y elegir un sentido correcto en base a los mismos.

Clasificación de los Sentidos

En esta etapa se hace uso de diferentes recursos externos para construir un inventario de sentidos subjetivos, en el que a cada sentido se le asigna un valor de subjetividad en concordancia con el rango definido en el trabajo; estos rangos son *fuertemente subjetivo*, *subjetivo*, y *objetivo*.

Para construir este inventario, se define un conjunto de reglas que transforman la información que proveen las herramientas externas en algún valor de subjetividad. Las herramientas utilizadas para llevar a cabo esta tarea fueron: *SentiWordNet*, *MicroWN-Op*, *Q-WordNet*, *WordNet-Affect*, *SentiSense*. Cabe aclarar que todas estas herramientas se basan en el análisis de sentimiento y detección de emociones, por lo que las reglas utilizadas son simplemente funciones que dado un valor de polaridad o token de emoción, eligen un valor de subjetividad adecuado.

Finalmente, para clasificar cada sentido se le asigna un puntaje de acuerdo a una función que toma en cuenta la categoría asignada en el inventario y la presencia del sentido en el lexicón de subjetividad, dando valores más altos cuanta más evidencia existe de que el sentido es subjetivo.

Clasificación de las Oraciones

Por último, en esta fase del pipeline se determina la subjetividad de las oraciones. Para esto se utiliza una función que clasifica a una oración como subjetiva cuando la sumatoria de los puntajes de subjetividad para cada palabra supera un determinado umbral λ, que fue estimado en 4.0 de manera empírica, sobre la base de datos *SemCor_MT*.

Resultados

En la etapa de evaluación del método, se encontró que el uso de una técnica de desambiguación semántica mejora en gran medida los resultados, detectando una mayor cantidad de oraciones subjetivas sin sacrificar precisión, y reduciendo los errores en la

clasificación de oraciones objetivas. Por otro lado, el uso de inventarios de subjetividad y de un lexicón de subjetividad probó también ser de mayor utilidad en la clasificación, impactando directamente en el f-score final.

Por último, al ser comparado con otros métodos populares como *OpinionFinder*, los resultados son prometedores, superando o igualándolo en cada uno de los bancos de datos utilizados.

Resumen

Todos los trabajos previos estudiados en este capítulo se basan en el idioma inglés. Algunos autores utilizan técnicas de aprendizaje supervisado [ref. 2, 3], otros aquellas no supervisadas [ref. 5], y otros combinan ambos métodos [ref. 4]. Por otro lado, es importante remarcar que las características que se utilizan suelen estar relacionadas con la categoría gramatical, la longitud del texto, la posición de las palabras, o patrones de bigramas y trigramas construidos según dichas categorías gramaticales. A su vez, existen también trabajos como [ref. 9], que resumen mucha de la información existente en relación a la detección de subjetividad y análisis de sentimiento, dado que estos suelen estar estrechamente relacionados, explorando tópicos más alejados del objetivo de este trabajo pero aún así interesantes, como búsqueda o utilidad de opiniones.

Capítulo 5

Diseño del Clasificador y Tecnologías Utilizadas

En este capítulo se presenta el diseño del clasificador de subjetividad propuesto y los algoritmos implementados, así como también una breve descripción de las tecnologías utilizadas y el porqué de su elección.

Arquitectura Propuesta

El sistema propuesto en este trabajo consta de varias etapas definidas que contienen distintos componentes funcionales. Dichos componentes adaptan los datos de entrada para ser interpretados más eficientemente, o bien entrenan al modelo elegido.

A grandes rasgos, la arquitectura completa consta de las cuatro etapas fundamentales que rigen a todos los sistemas de aprendizaje automático estadístico: preprocesado, entrenamiento, evaluación y predicción. Sin embargo, en esta sección se detallarán todas las funciones y subcomponentes característicos del sistema, así como también las decisiones de diseño e implementación tomadas en cada uno de ellos.

Representación de los Datos

Dado que el formato de texto crudo no es interpretable por los algoritmos de aprendizaje automático, es necesario recurrir a una forma alternativa de representación, de forma que pueda operarse con ellos.

En este marco se definen entonces dos representaciones, una intermedia en forma de matriz, y una final en forma de vector, con datos extraídos de su matriz correspondiente.

La representación matricial de una oración consiste simplemente en un conjunto de características de cada una de las palabras que la componen, luego de haber aplicado todas las técnicas de preprocesamiento de texto:

- **SWF-ISF:** tal como se describió en el *Capítulo 2*, este valor indica el nivel de impacto de una palabra en la subjetividad.
- **Frecuencia Relativa Subjetiva:** se define la FRS de x como

$$f_{xs} = \frac{n_{xs}}{n_s}$$

donde n_{xs} es la cantidad de ocurrencias de la palabra x en oraciones subjetivas, y n_s la cantidad total de palabras en oraciones subjetivas.

- **Frecuencia Relativa Objetiva:** se define la FRO de x como

$$f_{xo} = \frac{n_{xo}}{n_o}$$

donde n_{xo} es la cantidad de ocurrencias de la palabra x en oraciones objetivas, y n_o la cantidad total de palabras en oraciones objetivas.

- **Modificador:** esta característica es 1 cuando la palabra es un adjetivo o un adverbio, y un 0 en caso contrario.

De esta forma, una entrada de la base de datos como la siguiente:

S@El aguileño príncipe gozaba de una reputación hiperbólica

Se transforma, a través del método propuesto, en una matriz con la forma de la *Tabla 5.1*.

Palabra Original	SWF-ISF	FRS	FRO	Modificador
el	0.16716	0.00678	0.00805	0
aguileño	0.01381	0.0002	0	1
príncipe	0.013	0.0002	0.00008	0
gozaba	0.013	0.0002	0.00008	0
reputación	0.0076	0.0001	0	0
hiperbólica	0.0076	0.0001	0	1

Tabla 5.1: Ejemplo de representación matricial de una oración simple

Sin embargo, como se ha mencionado previamente, esta es sólo una representación intermedia, que facilita la obtención de métricas detalladas de las oraciones. Es interesante destacar que estas matrices no permiten una comparación numérica directa de las oraciones, puesto que las filas de matrices de dos oraciones distintas pueden representar palabras diferentes. Además las oraciones tienen longitudes variables, lo que dificulta aún más la tarea de comparación.

Una posible solución a ambos problemas es utilizar filas específicas para representar cada palabra de la base de datos, de esta forma las matrices se vuelven comparables y de

longitud fija. No obstante, esto traería consigo un conjunto de implicaciones que pueden ser aún más desfavorables para el modelo. El tamaño de las matrices sería muy grande, la mayoría de las filas debería ser rellenada con ceros y los datos estarían muy dispersos. Todo esto hace que se introduzca ruido en el modelo, y que los tiempos de entrenamiento y predicción suban considerablemente.

En consecuencia, esta no resulta ser una buena representación final para las oraciones, y por esta razón, las matrices generadas con el método descrito anteriormente sirven de base para el cálculo de características a nivel de oración, reduciendo la dimensionalidad de los datos, obteniendo así *vectores oración*.

Un vector oración es entonces la representación final de una oración para el modelo, y contiene las siguientes características:

- **Media de SWF-ISF Máximos:** la media de los n mayores SWF-ISF de la oración.[11]
- **Media de FRS:** la media de las FRS de la oración.
- **Media de FRO:** la media de las FRO de la oración.
- **Frecuencia Relativa de FRS sobre FRO:** se define a la $FR_{FRS/FRO}$ como la frecuencia relativa de las palabras cuya FRS es mayor a su FRO.
- **Frecuencia Relativa de Modificadores:** se define a la FRM como la frecuencia relativa de las palabras que son adjetivos o adverbios.
- **Frecuencia Relativa de Patrones de Bigramas Subjetivos:** se define a la métrica PABS como la frecuencia relativa de los bigramas que coinciden con alguno de los patrones de bigramas subjetivos preestablecidos en el modelo. A continuación se describen dichos patrones:

    1. **Sustantivo Modificado 1 (Patrón B1):** aquellos bigramas donde un adjetivo precede a un sustantivo.

        $$(ADJ,\ SUST)$$

        Ejemplos: (“*aguileño*”, “*príncipe*”), ("*enorme*", "*oso*")

    2. **Sustantivo Modificado 2 (Patrón B2):** aquellos bigramas donde un adjetivo sucede a un sustantivo.

        $$(SUST,\ ADJ)$$

        Ejemplos: (“*reputación*”, “*hiperbólica*”), ("*calidad*", "*aceptable*")

    3. **Verbo Modificado (Patrón B3):** aquellos bigramas donde un adverbio sucede a un verbo.

[11] En este trabajo se eligió utilizar un valor de n = 3.

$$(VERB, ADV)$$

Ejemplos: (“*funciona*”, “*bien*”), ("*superará*", "*ampliamente*")

- **Frecuencia Relativa de Patrones de Trigramas Subjetivos:** se define a la métrica PATS como la frecuencia relativa de los trigramas que coinciden con alguno de los patrones de trigramas subjetivos preestablecidos en el modelo. A continuación se describen dichos patrones.

    1. **Sustantivo y Adjetivo Modificados 1 (Patrón T1):** aquellos trigramas donde un adverbio y un adjetivo preceden a un sustantivo.

    $$(ADV, ADJ, SUST)$$

    Ejemplos: (“*muy*”, "*buena*", “*cámara*”)

    2. **Sustantivo y Adjetivo Modificados 2 (Patrón T2):** aquellos trigramas donde un sustantivo precede a un adverbio y un adjetivo.

    $$(SUST, ADV, ADJ)$$

    Ejemplos: (“*animal*”, "*bastante*", “*grande*”), (“*hombre*”, "*demasiado*", “*tímido*”)

    3. **Verbo Modificado 1 (Patrón T3):** aquellos trigramas donde un verbo precede a dos adverbios.

    $$(VERB, ADV, ADV)$$

    Ejemplos: (“*voló*”, "*muy*", “*alto*”), (“*sanará*”, "*suficientemente*", “*rápido*”)

    4. **Verbo Modificado 2 (Patrón T4):** aquellos trigramas donde dos adverbios preceden a un verbo.

    $$(ADV, ADV, VERB)$$

    Ejemplos: (“*muy*”, "*calmado*", “*dijo*”), (“*bien*”, "*arriba*", “*estaba*”)

De esta manera, la versión vectorizada de la matriz de la *Tabla 5.1* tendría la forma que se muestra en la *Tabla 5.2*.

Media de SWF-ISF Máximos	Media de FRS	Media de FRO	$FR_{FRS/FRO}$	FRM	PABS	PATS
0.06465	0.00126	0.00136	0.83333	0.33333	0.28571	0

Tabla 5.2: Ejemplo de representación vectorizada de una oración simple

Finalmente, se define la representación de las etiquetas subjetiva y objetiva como valores numéricos, donde 1 representa la etiqueta *subjetiva* y 0 la etiqueta *objetiva*.
Así, el resultado de ejecutar el pipeline de preprocesamiento es una lista de vectores similares a la que se muestra en la *Tabla 5.2*, y una lista de etiquetas asociadas a cada uno de ellos, que indican si corresponde a una oración subjetiva u objetiva.

Preprocesamiento de los Datos

En base a la representación definida anteriormente para cada oración, se define ahora el método concreto, las herramientas y la arquitectura que rige al sistema de preprocesamiento. El objetivo de este subsistema es obtener oraciones etiquetadas de la base de datos y transformarlas en sus matrices y vectores correspondientes de acuerdo al formato mostrado en la sección anterior. Para esto se hace uso de una selección y combinación de las técnicas estudiadas previamente.[12]

A grandes rasgos, el subsistema de preprocesamiento de datos está compuesto por tres fases, la fase de formato, la fase de numerización y la fase de vectorización.

- **Fase de Formato:** En esta etapa se llevan a cabo todos aquellos procesos que agregan metadatos y filtran información irrelevante. Está conformado por los siguientes componentes:

    1. *Filtro de Caracteres*: Elimina de la oración de entrada todos los caracteres de puntuación, signos de exclamación y dígitos.
    2. *Tokenizador*: Utiliza el tokenizador de NLTK en español para transformar la oración en una lista de palabras.
    3. *POS Tagger*: Se utiliza el POS Tagger de Stanford para español, a fin de obtener una lista de pares $(palabra, tag)$ a partir de la lista de palabras generada en el paso anterior. La desventaja que presenta la elección de esta herramienta es que su código fuente está escrito en Java, por lo cual el método que lo utiliza desde Python simplemente ejecuta el archivo *jar* del tagger cada vez que se desea etiquetar una oración y luego interpretar los resultados, lo que reduce considerablemente la performance del preprocesador en una primera implementación. Sin embargo, la herramienta también dispone de un modo de ejecución *batch* en el que se utiliza una sóla instancia del tagger para etiquetar tantas oraciones como se le indiquen. Al pasar a este último modo, los tiempos de ejecución se reducen en gran medida. Por lo tanto, este componente etiqueta todas las oraciones en una sóla llamada.

[12] Ver *Capítulo 2: Extracción de Características*

4. *Extracción de Bigramas y Trigramas*: Construye una lista de bigramas y una de trigramas para cada oración utilizando la biblioteca NLTK.
5. *Filtro de Stopwords*: Elimina de los pares $(palabra, tag)$ generados en la etapa de POS tagging aquellos que coinciden con una palabra presente en la lista de stopwords proporcionada por NLTK para el idioma español.
6. *Stemmer*: Ejecuta el algoritmo de stemming de Snowball para español para cada palabra restante en la estructura.

- **Fase de Numerización:** Se recolecta toda la información adquirida en la fase de formato, se calculan las características deseadas y se construye la matriz asociada a cada oración.

- **Fase de Vectorización:** En esta última etapa se toman las matrices generadas en la fase de numerización, se calculan las características deseadas y se construyen los vectores oración asociados.

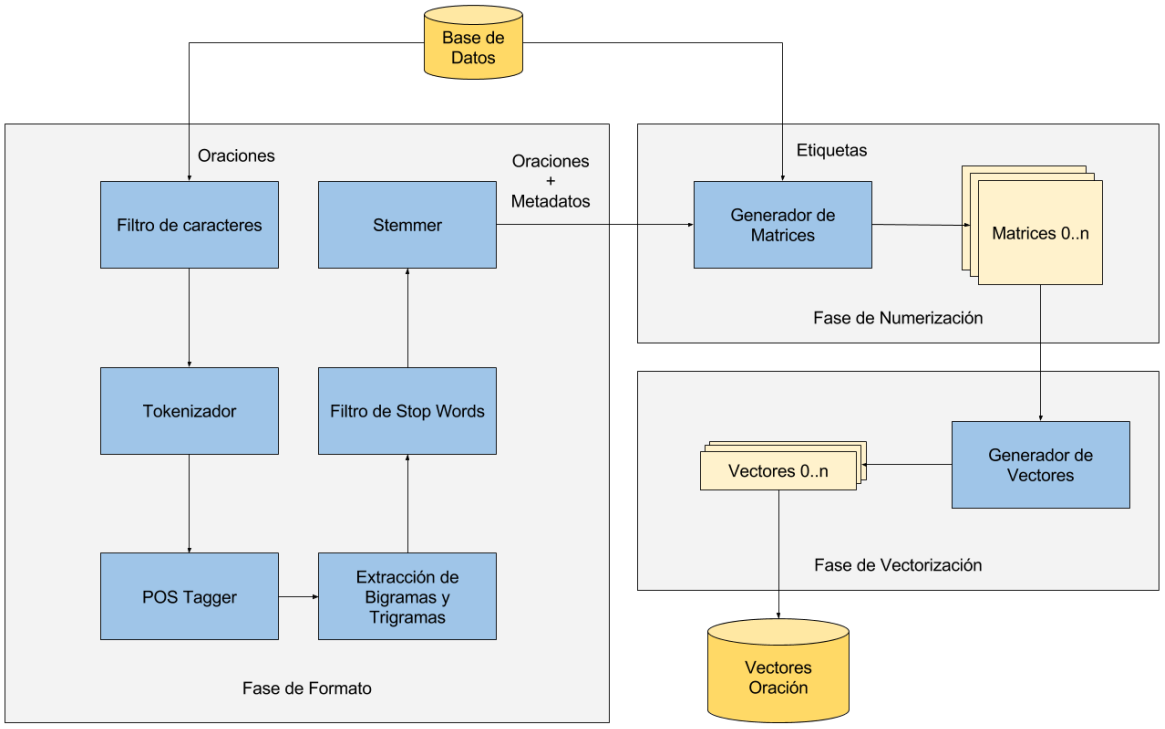

Figura 5.1: Arquitectura del Preprocesador

Finalmente, la división de los datos en datos de entrenamiento y evaluación sigue una relación de 80% y 20% de la base de datos original, respectivamente. Para esto, se utilizaron las funciones predefinidas de la biblioteca *scikit-learn* de manera de obtener particiones estratificadas, es decir, cuya relación entre oraciones subjetivas y objetivas sea la misma en ambas particiones, a fin de obtener datos de evaluación que reflejen fielmente la relación entre las clases en la etapa de entrenamiento.

Entrenamiento del Modelo

Para la etapa de entrenamiento se eligieron dos algoritmos de aprendizaje automático, y se exploró su configuración a fin de obtener los resultados óptimos. A continuación se describen los pasos seguidos para ajustar los parámetros de cada uno de ellos.

Support Vector Machine

Se procedió con el entrenamiento del clasificador con 100 particiones de datos diferentes. Para cada partición se entrenaron 3 modelos con kernels distintos: lineal, RBF y sigmoidal, y con una combinación de valores de los parámetros C y $gamma$ en un rango preestablecido. Para cada uno de ellos, se evaluó precisión, recall, f-score y f-score macro promedio de los resultados de una validación cruzada con $k = 5$, obteniendo resultados con la forma del extracto que puede verse en la *Tabla 5.4*.
Nótese que los valores de $gamma$ no fueron modificados para el kernel lineal, puesto que dicho parámetro no tiene efecto en el mismo.
Los rangos de valores utilizados para los parámetros del clasificador fueron:

$$R_c = (0.01,\ 0.1,\ 1,\ 10,\ 100,\ 1000)$$

$$R_{gamma} = (0.001,\ 0.01,\ 0.1,\ 1,\ 10,\ 100,\ 1000)$$

Por último, se buscó aquella combinación de parámetros que produjera el máximo f-score medio en la validación cruzada, y se almacenó la partición utilizada en ese caso. Los parámetros óptimos encontrados pueden observarse en la *Tabla 5.3*.

Kernel	Sigmoidal
C	0.01
Gamma	0.001

Tabla 5.3: Resultados del SVM con parámetros óptimos

Part.	Kernel	Prec. O	Prec. S	Recall O	Recall S	F-Score O	F-Score S	F-Score Macro	C	Gamma
0	linear	0.95	0.876	0.865	0.955	0.905	0.913	0.84	0.1	auto
0	sigmoid	0.945	0.875	0.865	0.95	0.903	0.911	0.879	0.01	0.1
1	rbf	0.922	0.802	0.77	0.935	0.839	0.863	0.861	0.1	0.1
1	linear	0.916	0.84	0.825	0.925	0.868	0.88	0.791	10	auto
2	sigmoid	0.908	0.855	0.845	0.915	0.875	0.884	0.814	1	1
2	rbf	0.933	0.858	0.845	0.94	0.887	0.897	0.884	0.01	0.1
3	linear	0.926	0.885	0.88	0.93	0.902	0.907	0.873	10	auto

Tabla 5.4: Fragmento de resultados de SVMs con diferentes configuraciones

Perceptrón Multicapa

Para este clasificador se consideraron 10 particiones de datos diferentes, dada la cantidad de tiempo que toma el entrenamiento y la cantidad de parámetros a ajustar, que alcanzan a sumar un total de más de 56000 configuraciones diferentes para cada partición. En cada una de ellas se entrenaron modelos variando los valores de un subconjunto de sus diferentes parámetros. De manera similar al procedimiento seguido para el SVM, se evaluó precisión, recall, f-score y f-score macro promedio de los resultados de una validación cruzada con $k = 5$ para cada modelo entrenado, obteniendo resultados como los que se muestran en la *Tabla 5.6*.

Los optimizadores utilizados fueron *lbfgs* y *adam*, mientras que las funciones de activación fueron la función logística, tangente hiperbólica y ReLU. El rango de valores utilizado para el parámetro alfa fue:

$$R_{\alpha} = (0.0001,\ 0.001,\ 0.01,\ 0.03,\ 0.1,\ 0.2)$$

Por otro lado, también se entrenaron redes con diferentes arquitecturas, modificando el tamaño y cantidad de capas ocultas, para lo cual se tuvieron en cuenta las siguientes restricciones:

- La cantidad mínima de neuronas en una capa oculta es de 2
- La cantidad máxima de neuronas en una capa oculta es de 6
- Debe existir al menos una capa oculta, por limitaciones de la implementación utilizada
- La cantidad máxima de capas ocultas es de 3

Por último, se buscó aquella combinación de parámetros que produjera el máximo f-score medio en la validación cruzada, y se almacenó la partición utilizada en ese caso. Los parámetros óptimos encontrados pueden observarse en la *Tabla 5.5*.

Optimizador	adam
Activación	relu
Alfa	0.01
Capas Ocultas	(3)

Tabla 5.5: Resultados de la red neuronal con parámetros óptimos

Así, la arquitectura elegida para la red consta de 7 neuronas en la capa de entrada, 1 en la de salida, y 1 capa oculta con 3 neuronas, tal como puede apreciarse en la *Figura 5.2*.

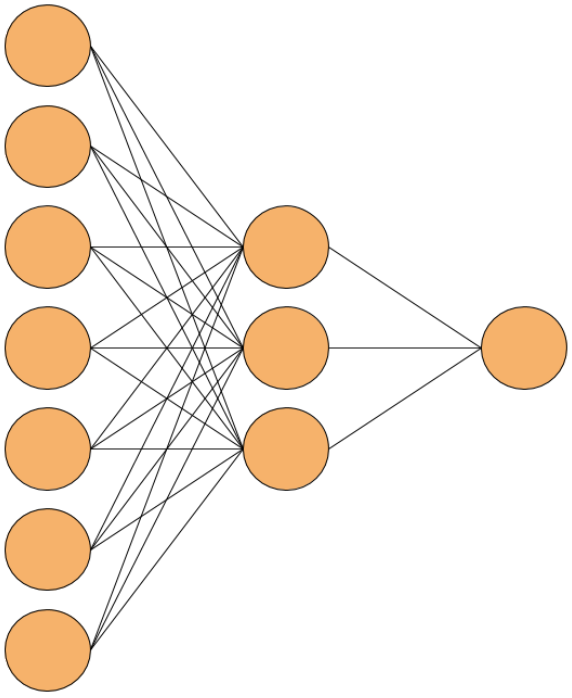

Figura 5.2: Arquitectura de la red neuronal óptima encontrada

Part.	Prec. O	Prec. S	Recall O	Recall S	F-Score O	F-Score S	F-Score Macro	Optim	Activ	Alfa	Capas Ocultas
0	0.912	0.938	0.94	0.91	0.926	0.923	0.871	lbfgs	logistic	0.001	(2)
0	0.945	0.871	0.86	0.95	0.9	0.909	0.82	adam	logistic	0.2	(2)
1	0.914	0.863	0.855	0.92	0.883	0.891	0.784	lbfgs	relu	0.01	(5, 5, 5)
1	0.885	0.889	0.89	0.885	0.887	0.887	0.901	lbfgs	relu	0.01	(5, 5, 6)
2	0.929	0.869	0.86	0.935	0.893	0.9	0.831	adam	logistic	0.001	(3, 5)
2	0.934	0.870	0.86	0.94	0.895	0.903	0.896	adam	logistic	0.01	(2, 5)
3	0.916	0.928	0.93	0.915	0.923	0.921	0.798	adam	tanh	0.03	(5, 2, 3)

Tabla 5.6: Fragmento de resultados de redes neuronales con diferentes configuraciones

Resumen

El texto originario de la base de datos es preprocesado a través de un pipeline que utiliza las técnicas tratadas en el *Capítulo 2* para convertir cada oración en una matriz de características, donde las filas son las palabras de la oración, y las columnas son métricas calculadas durante dicha etapa de preprocesado. Posteriormente, dado que el formato matricial no es apropiado para los algoritmos de aprendizaje automático, se calculan nuevas métricas en base a las matrices, generando vectores oración.

Con esta información, se realizaron búsquedas de modelos SVM y perceptrón multicapa óptimos, variando sus parámetros y las particiones de datos hasta encontrar aquellos cuyas predicciones fueran de mejor calidad. Dichos modelos fueron evaluados con una validación cruzada con $k = 5$.

Capítulo 6

Construcción de la Base de Datos

En este capítulo se describe la forma de la base de datos construida y utilizada para diseñar el sistema descrito anteriormente. A su vez, se detallan las consideraciones de diseño para la obtención de las dos mil oraciones clasificadas que conforman la base de datos obtenida.

Consideraciones

Como ya se ha mencionado anteriormente, la cantidad de datos y herramientas disponibles para el procesamiento de lenguaje natural en español no es tan extensa como la que puede encontrarse para el inglés, de forma que fue imposible contar con una base de datos preexistente de oraciones subjetivas y objetivas. En vista de esto, fue necesario construir una nueva a partir de bases de datos de texto plano que pueden encontrarse en la web de forma gratuita. Estos *corpus* contienen miles y miles de oraciones en sus textos, por lo cual resultaron ser una herramienta fundamental en el desarrollo.

En particular, se han intentado incluir oraciones de textos de diferentes ámbitos y disciplinas, a fin de ofrecer un grado aceptable de variación de subjetividad en la recopilación de oraciones. Así, la relación obtenida entre oraciones subjetivas y objetivas es de 50%-50%. Considerando esto, se han elegido una variedad de categorías de textos, entre los que se encuentran textos históricos, novelas y científicos.

Los textos científicos fueron extraídos de una lista de papers del CACIC (Congreso Argentino de Ciencias de la Computación) publicados entre los años 2005 y 2013. Las novelas y textos históricos fueron extraídos del corpus del Proyecto Gutenberg, que al día de la fecha ofrece alrededor de 53.000 libros de forma gratuita y en diferentes formatos, de los cuales existe un amplio subconjunto escrito en español.

Sin embargo, la disposición de dichos textos es sólo el primer paso en la construcción de la base de datos, puesto que es necesario preprocesar esa información cruda, de manera que pueda ser etiquetada y estructurada para utilizarse con el fin de entrenar el modelo.

Estructura

Una vez obtenidos los textos crudos, se procedió con la división de los textos en oraciones. Aunque esta tarea puede parecer sencilla en un principio, su implementación no es trivial, dado que no existe una única forma de separar un texto en oraciones.

Inicialmente, con el objetivo de simplificar el problema, se podría definir la estructura de un texto a través del siguiente conjunto de reglas:

1. El elemento atómico es la letra, que pertenece al abecedario español
2. Las letras conforman palabras, delimitadas por espacios (‘ ’)
3. Las palabras conforman oraciones, delimitadas por puntos (‘.’)
4. Las oraciones conforman párrafos, delimitados por fines de línea (‘\n’ o ‘\r\n’)
5. Los párrafos conforman la totalidad del documento

Si bien estas reglas parecen describir completamente la estructura de un documento, tienen graves fallas que provienen de la asunción de ciertas construcciones que en general no suelen ser respetadas.

Desde cierto punto de vista, *la regla 1* es acertada, dado que no hay componente más elemental en un texto que una letra perteneciente al abecedario español; sin embargo, muchos caracteres de los que se encuentran en los textos seleccionados son caracteres especiales o signos de puntuación, lo cual introduce nuevas problemáticas como por ejemplo la forma de tratar signos de exclamación y admiración, los caracteres que surgen de la conversión de otros formatos a texto plano[13], o la codificación de cada documento[14], que puede introducir caracteres desconocidos si se lo decodifica con un formato incorrecto.

La regla 2 asume que toda separación entre palabras se produce por un espacio en blanco, cuando en realidad puede ser por signos de puntuación, guiones que indiquen la presencia de diálogo, corchetes que indiquen una referencia bibliográfica, comillas que indiquen una cita, signos de exclamación o admiración, o simplemente un carácter de fin de línea. Si esto no se soluciona en la fase de tokenización, podría perderse mucha información, por ejemplo al interpretar los términos “aparecía,” y “aparecía” como palabras totalmente distintas.

La regla 3 asume que las oraciones se delimitan por puntos, lo cual, como se ha mencionado antes, puede ser falso. Existen casos como “Sr. Méndez” o “5.000 pesos” donde la presencia del punto no indica la separación de dos oraciones. Además, una oración que introduce un diálogo con el carácter ‘:’ o diferentes líneas de un diálogo separadas por un carácter de fin de línea sí suelen marcar este límite. Por este último caso

[13] Un ejemplo de esto son palabras en itálica que al convertirse a texto plano pierden el formato, y aparecen encerradas entre guiones bajos (‘_’): *despertar* -> _despertar_
[14] UTF-8, ISO/IEC 8859-1, UTF-16, US-ASCII, etc.

es que *la regla 4* tampoco es completamente válida, dado que si dos oraciones pueden separarse por un carácter de fin de línea, entonces no puede asumirse que dos porciones de texto separadas por dicho carácter sean párrafos y no oraciones; sin embargo esto no resulta un problema en la práctica, debido a que en este trabajo el grado de separación que se desea alcanzar es a nivel de oración, por lo que toda pieza de texto que componga estas unidades será últimamente fragmentada.

Por último, *la regla 5* es acertada. Aunque podría descomponerse el texto en distintas unidades además de párrafos (por ejemplo diálogos o citas), esta separación no tiene un efecto negativo considerable sobre lo que se quiere conseguir en este trabajo, por lo que se toma a este tipo de fragmentos textuales como oraciones comunes y corrientes.

Considerando lo mencionado anteriormente, se decidió que la estructura de la base de datos debe tener las siguientes características:

- Los datos son texto plano, codificados según el estándar Unicode, utilizando su representación de longitud variable, denominada UTF-8 (Unicode Transformation Format)
- Cada línea de texto está compuesta por un indicador de clase y una oración, separados por el carácter '@', y en ese orden. Los indicadores de clase posibles son dos: S (oración subjetiva) y O (oración objetiva)
- Las oraciones aparecen en la base de datos tal como aparecen en los textos fuente, sin modificación de su contenido, y cualquier procesado de los datos para posterior análisis se delega a la etapa de preprocesamiento del modelo.

Clasificación

Ya divididos los textos en oraciones, es necesario obtener un etiquetado de cada una de ellas, a fin de formar la estructura que se describe en la sección anterior. Para llevar a cabo esta tarea existen diferentes opciones; a continuación se describen aquellas que fueron consideradas.

Etiquetado Manual

Este es el enfoque más sencillo de todos, pero también aquel que más tiempo consume. Consiste en la revisión manual de las oraciones, decidiendo para cada una si es subjetiva u objetiva.

Si bien, como se ha dicho, es el enfoque más sencillo, también es el más preciso, dado que depende íntegramente de la concepción de subjetividad de la persona que realiza el etiquetado, que puede ser fácilmente guiada para concordar con las definiciones planteadas en el trabajo.

Por otra parte, el tiempo necesario para realizar un etiquetado manual puede reducirse hasta cierto punto mediante el uso de un software que guíe la tarea y sea el encargado de producir datos en el formato necesario, evitando así posibles errores humanos.

Etiquetado Automático

Este método consiste en delegar completamente la tarea de etiquetado a un sistema de software. Su principal desventaja es el alto grado de imprecisión, que depende en gran medida del algoritmo que utilice el software elegido para etiquetar cada oración. En contraste, el tiempo que requiere es muy poco, superando ampliamente a los métodos manuales.

Existen varias técnicas que pueden utilizarse en el etiquetado automático; a continuación se listan algunas de ellas:

- Según la cantidad de adjetivos y adverbios presentes en la oración.
- Según la aparición de un conjunto de términos o construcciones indicadoras de subjetividad u objetividad. Por ejemplo, los verbos *preferir, querer, pensar, creer*, entre otros, suelen ser buenos indicadores de subjetividad, mientras que verbos utilizados para describir, como *decir, armar, ir* o *correr*, suelen indicar objetividad.
- Utilizando algoritmos conocidos que miden características similares a la subjetividad, como clasificadores de sentimientos positivos, negativos y neutrales; en este contexto, podrían considerarse emociones positivas o negativas como indicadores de subjetividad, mientras que aquellas más neutrales podrían ser consideradas objetivas.

Etiquetado Asistido

Este método se define como la combinación entre etiquetado manual y automático, de manera que cada oración se clasifica de forma manual, pero con una sugerencia por parte de un algoritmo automático, logrando reducir aún más los tiempos de etiquetado manual y los errores humanos, pero manteniendo la precisión. Aún así, los tiempos de etiquetado suelen ser muy altos con esta técnica.

Conclusiones

En base al análisis presentado anteriormente, se intentó utilizar una combinación de las técnicas de etiquetado manual y automático, con un algoritmo de clasificación de sentimiento. Sin embargo, el enfoque fue reemplazado por un etiquetado únicamente manual, dado que, cómo fue demostrado en [ref. 16], los clasificadores de sentimiento

existentes en la actualidad para español, no tienen la madurez suficiente como para determinar fehacientemente la polaridad de una oración.[15]

A través de este método se obtuvo una base de datos cuya distribución y cantidad de entradas se ve representada en la *Figura 6.1*.

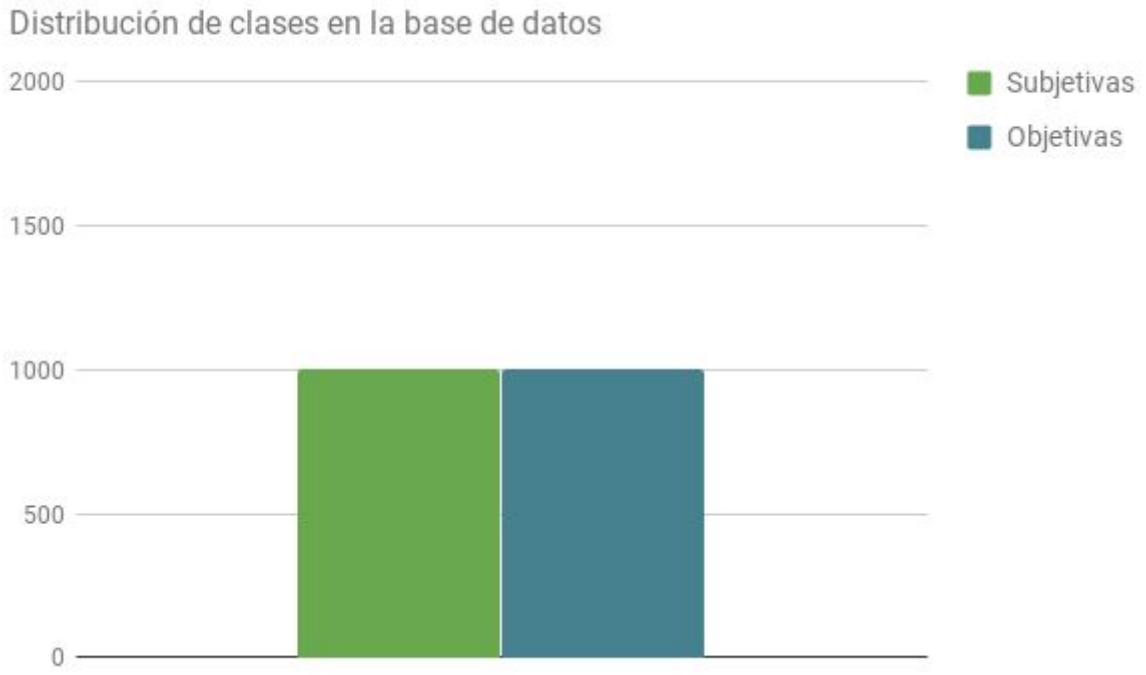

Figura 6.1: Distribución de clases en la base de datos

De esta manera, se obtuvo un total de 2000 oraciones etiquetadas, cuya precisión es muy alta, gracias a la clasificación manual empleada.

Resumen

La base de datos de este trabajo fue construída utilizando textos académicos actuales y textos literarios del proyecto Gutenberg, escritos en español. De todas las opciones de etiquetado disponibles (manual, automática y asistida), se eligió trabajar con un etiquetado manual, de forma de asegurar la precisión en las clasificaciones, y la concordancia de las mismas con las definiciones del *Capítulo 1*. Finalmente, la base de datos obtenida consta de dos mil oraciones, de las cuales una mitad es objetiva, y la otra subjetiva.

[15] En [ref. 16], los clasificadores comparados produjeron resultados inconsistentes entre sí, sobre un mismo conjunto de datos

Capítulo 7

Resultados Obtenidos

Se detallan los resultados obtenidos luego del entrenamiento y evaluación de los clasificadores, resaltando aspectos interesantes como las características más útiles en cada modelo y la correlación entre las variables de los vectores y la subjetividad.

Una vez encontradas las mejores particiones de datos y la mejor configuración de los modelos, pueden extraerse ciertas métricas que indican cómo estos se comportan, y qué relaciones existen entre los datos que se obtuvieron de las oraciones. Asimismo también puede explorarse la distribución de estas variables en la base de datos, y la forma en que impactan individualmente en la subjetividad.

Métricas de Evaluación

Dado un conjunto de etiquetas y predicciones, es muy sencillo obtener tres métricas extremadamente útiles en la evaluación de clasificadores: precisión, recall (o cobertura) y f-score. Sin embargo, antes de describir cada una de ellas es necesario definir un conjunto de términos asociados a las predicciones y su calidad.

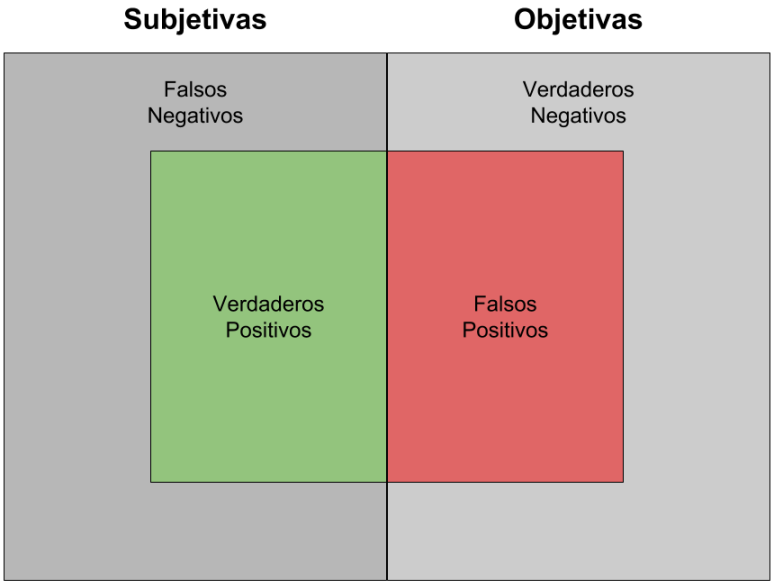

Figura 7.1: Relación entre etiquetas y predicciones del clasificador

En la *Figura 7.1* se puede observar cómo se caracteriza al conjunto de oraciones, el recuadro interno representa todas las oraciones que fueron clasificadas como subjetivas, dejando al resto de las oraciones del recuadro externo clasificadas como objetivas. En este marco se presentan los siguientes conceptos.

Verdaderos Positivos: Aquellas oraciones subjetivas clasificadas como subjetivas.
Falsos Positivos: Aquellas oraciones objetivas clasificadas como subjetivas.
Verdaderos Negativos: Aquellas oraciones objetivas clasificadas como objetivas.
Falsos Negativos: Aquellas oraciones subjetivas clasificadas como objetivas.

Es importante aclarar que estas definiciones están hechas desde el punto de vista de la clase subjetiva, por lo que su significado se invierte cuando se evalúa la clase objetiva. Basándose en estos conceptos, se definen entonces las métricas que se utilizaron en la evaluación de los clasificadores:

Precisión: Representa la fracción de elementos de una clase dentro del total de elementos asignados a esa clase.

$$P(c) = \frac{VP}{VP + FP}$$

Donde c es una clase, es decir, subjetiva u objetiva, VP son los verdaderos positivos y FP son los falsos positivos.

Recall: Representa la fracción de elementos asignados a una clase dentro del total de elementos de esa clase.

$$R(c) = \frac{VP}{VP + FN}$$

Donde c es una clase, es decir, subjetiva u objetiva, VP son los verdaderos positivos y FN son los falsos negativos.

F-Score: Dado que ni precisión ni recall pueden determinar individualmente el verdadero rendimiento del clasificador, esta medida representa lo que se denomina la media armónica entre ambas.

$$FS(c) = 2\left(\frac{P(c)R(c)}{P(c) + R(c)}\right)$$

Donde c es una clase, es decir, subjetiva u objetiva, y P y R son las funciones definidas previamente para precisión y recall, respectivamente.

Características Individuales

En particular, se encontró que las variables por sí mismas no son grandes indicadores de subjetividad, como puede apreciarse en las *Figuras 7.2* a la *7.6*, la mayoría de las variables tiene una ligera tendencia a incrementar o disminuir con una clase particular, pero ninguna lo suficientemente marcada como para poder afirmar que existe una correlación entre ambas.[16]

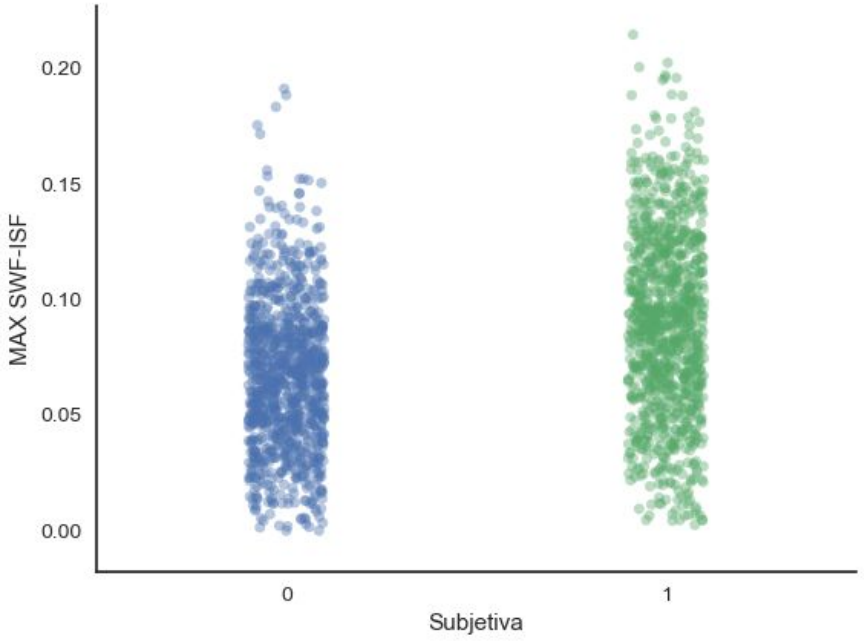

Figura 7.2: Relación entre promedio de SWF-ISFs máximos y subjetividad

[16] Notar que todos los gráficos incluyen la misma cantidad de oraciones subjetivas que objetivas

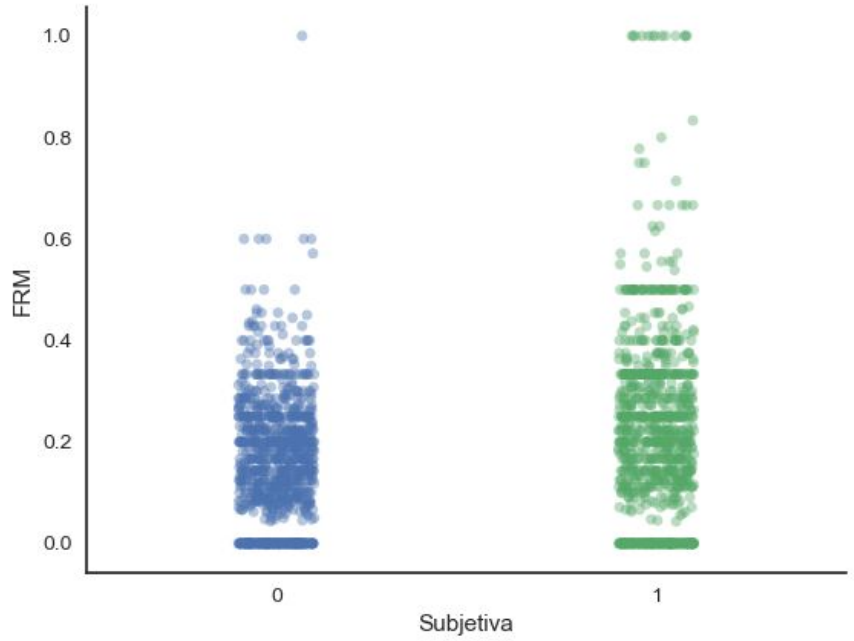

Figura 7.3: Relación entre frecuencia relativa de modificadores y subjetividad

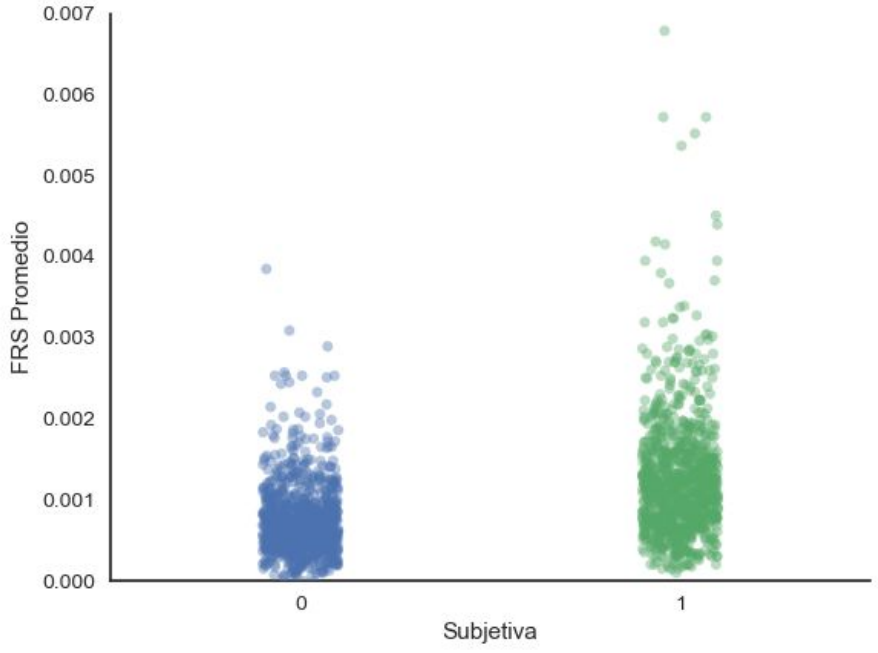

Figura 7.4: Relación entre FRS promedio y subjetividad

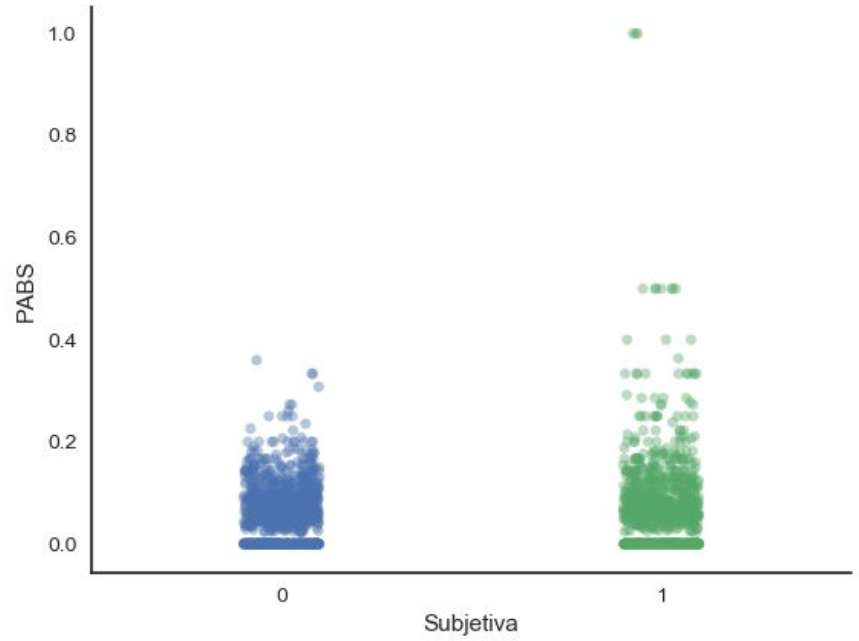

Figura 7.5: Relación entre PABS y subjetividad

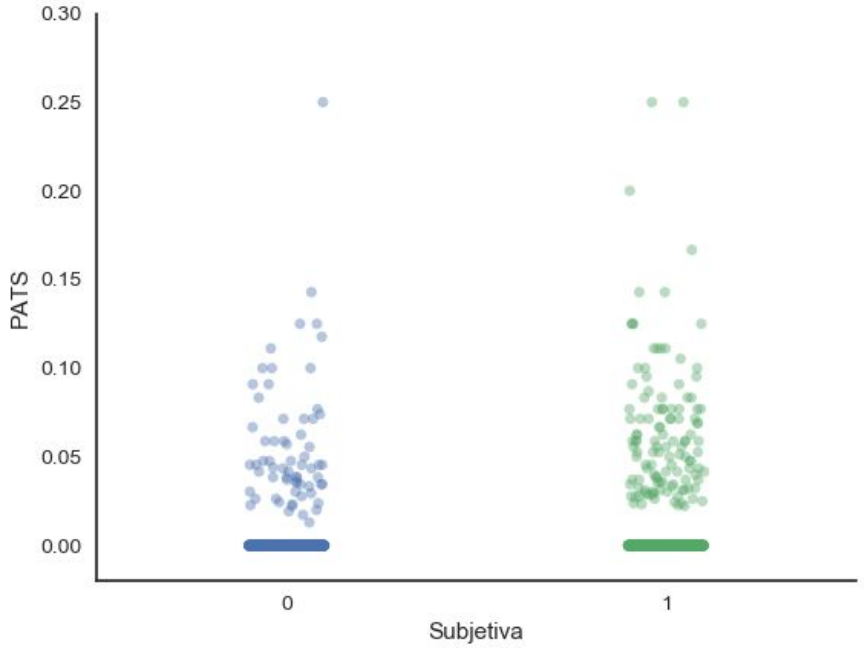

Figura 7.6: Relación entre PATS y subjetividad

Por otro lado, algunas variables dejan ver una correlación más fuerte, que da indicios de su posible utilidad para los modelos, como la frecuencia relativa de FRS sobre FRO (*Figura 7.7*), y el FRO promedio (*Figura 7.8*).

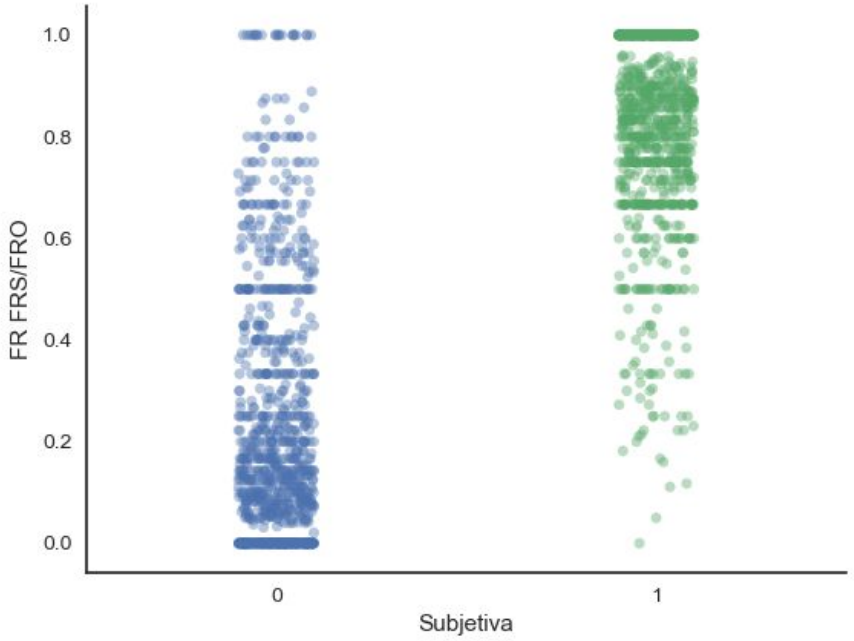

Figura 7.7: Relación entre frecuencia relativa de FRS/FRO y subjetividad

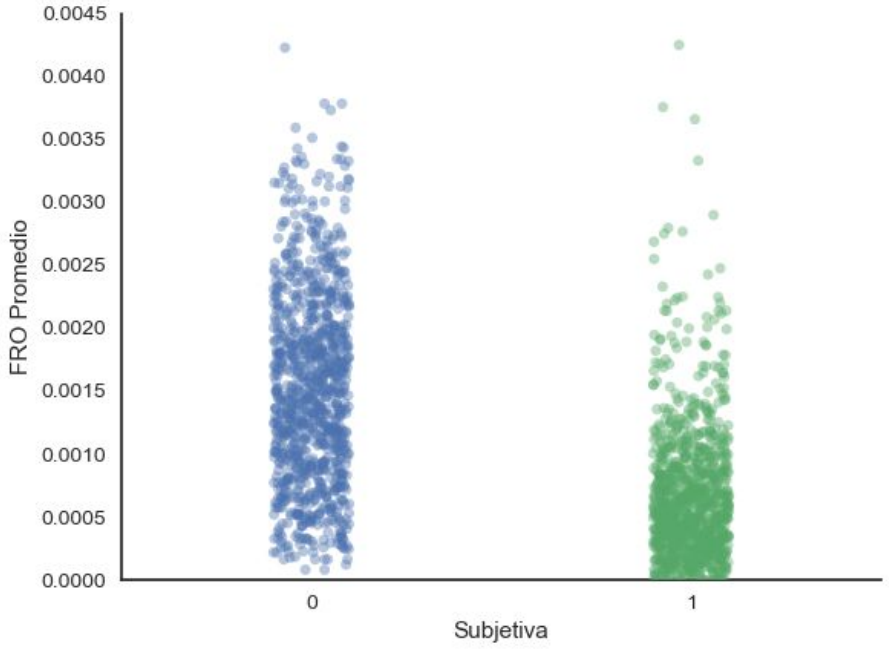

Figura 7.8: Relación entre FRO promedio y subjetividad

Support Vector Machine

Dado que el SVM óptimo encontrado utiliza un kernel lineal, es posible extraer del modelo los coeficientes asignados a cada característica de los vectores, determinando así cuál es el grado de relevancia y qué tanta información se obtiene de cada característica.

De esta forma, puede verse en la *Figura 7.9* la importancia que el modelo asigna a cada variable en su etapa de entrenamiento. Tal como se esperaba a partir del análisis realizado sobre las variables individuales, se observa que la característica más relevante resulta ser la frecuencia relativa de FRS sobre FRO, seguido por la frecuencia relativa de modificadores y el promedio de los SWF-ISFs máximos. Estos resultados son coherentes, dado que la relación entre palabras subjetivas y objetivas, la presencia de modificadores, y el índice SWF-ISF son claros indicadores de subjetividad.

Por otro lado, es sorprendente no encontrar tanta relevancia en la presencia de bigramas y trigramas subjetivos, siendo su utilidad casi la mitad de la del SWF-ISF, y más de cuatro veces menos que la de la frecuencia relativa de FRS sobre FRO. Sin embargo, analizándolo en retrospectiva, tal vez los n-gramas elegidos son característicos de textos más modernos como blogs y críticas en internet; en contraste con la base de datos utilizada, que fue construida extrayendo oraciones de textos de fines del siglo XIX y principios del siglo XX. Por esta razón, sería interesante ver los resultados de este mismo modelo utilizando una base de datos más actual, de esta forma tal vez los bigramas y trigramas tomen un rol más fuerte en la clasificación.

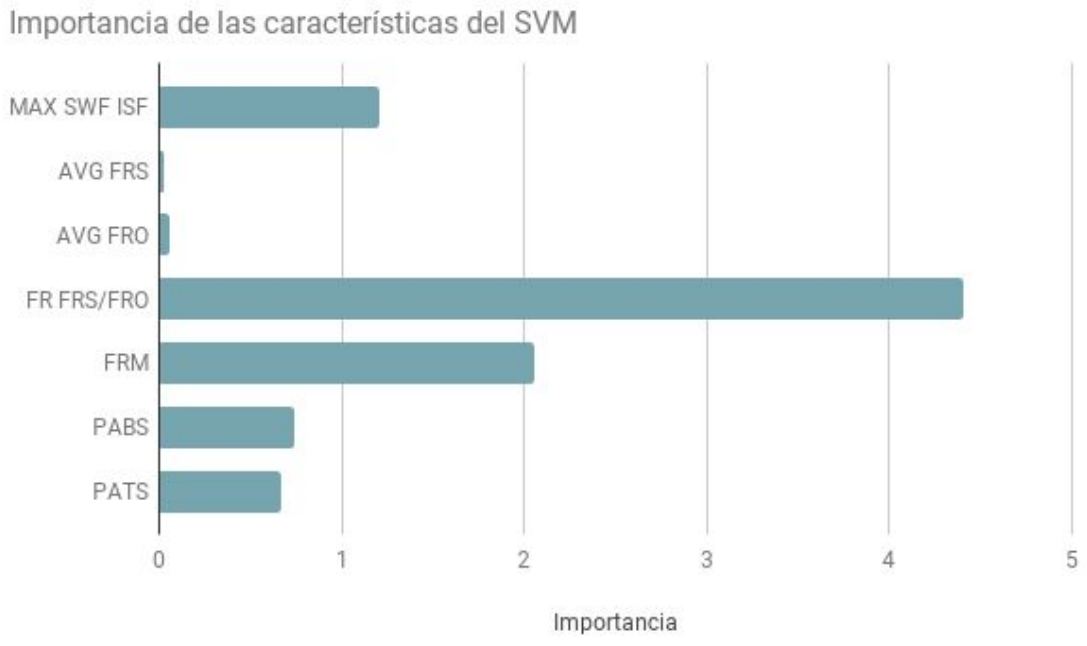

Figura 7.9: Importancia de las características del SVM

	Objetiva	Subjetiva
Precisión	0.90575	0.87081
Recall	0.865	0.91
F-Score	0.88491	0.88997
F-Score Macro	0.89126	

Tabla 7.1: Resultados del SVM óptimo hallado

Precisión Promedio	0.86936
Recall Promedio	0.92729
F-Score Macro Promedio Total	0.90161
Desviación Estándar de F-Score Macro Promedio	0.0003

Tabla 7.2: Resultados de la evaluación del SVM con validación cruzada

Perceptrón Multicapa

Debido al funcionamiento de este algoritmo y a su cualidad de caja negra, es muy difícil determinar fehacientemente la relevancia de las características de acuerdo a los pesos asignados a cada una de ellas. No obstante, en algunos trabajos se emplea un análisis de sensibilidad sobre la red para determinar cómo las fluctuaciones en los valores de cada variable afectan a la salida.

	Objetiva	Subjetiva
Precisión	0.94972	0.86425
Recall	0.85	0.955
F-Score	0.89709	0.90736
F-Score Macro	0.91228	

Tabla 7.3: Resultados del perceptrón multicapa óptimo hallado

Precisión Promedio	0.81686
Recall Promedio	0.9047
F-Score Macro Promedio Total	0.88412
Desviación Estándar de F-Score Macro Promedio	0.05217

Tabla 7.4: Resultados de la evaluación del perceptrón multicapa con validación cruzada

Comparación de los Modelos

Como se observa en la *Tabla 7.1 y 7.3*, al buscar el f-score macro máximo para cada modelo en la etapa de búsqueda del modelo óptimo, se garantiza que los clasificadores ponderan la precisión y la cobertura de ambas clases, y que su performance es medida en una variedad de datos de entrenamiento y prueba, gracias a la validación cruzada. A su vez, cada clasificador fue evaluado una vez más repitiendo la validación cruzada con $k = 5$ 30 veces, obteniendo los resultados de la *Figura 7.10* y *7.11*, que determinan los valores mostrados en la *Tabla 7.2* y *7.4*.

Esta evaluación final de los modelos revela detalles interesantes: el SVM tiende a mantener su calidad de predicción en diferentes conjuntos de datos, mientras que la de la red neuronal es mucho más variable e impredecible. Esto indica que es probable que haya un cierto nivel de sobreajuste de la red a la partición de datos utilizada (entrenamiento y prueba).

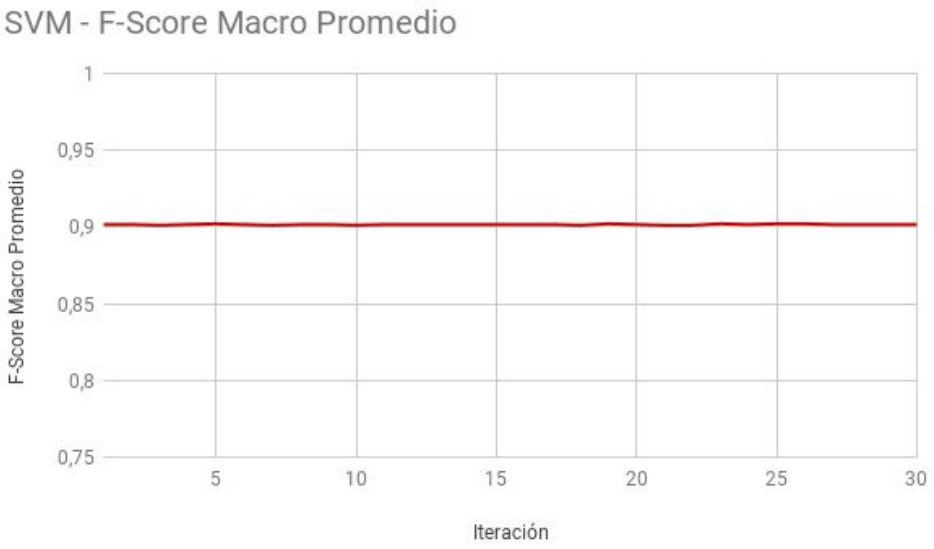

Figura 7.10: F-Scores Macro promedio del SVM

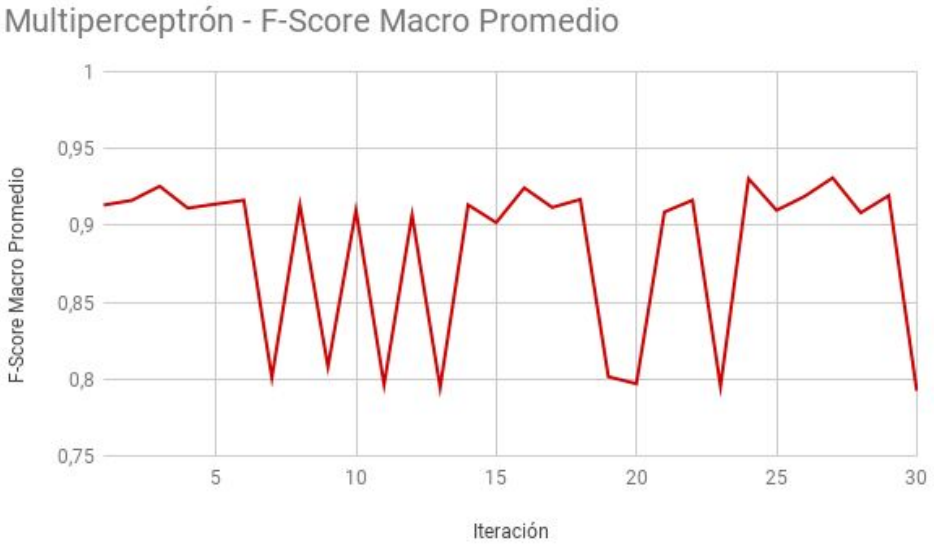

Figura 7.11: F-Scores Macro promedio del multiperceptrón

Por otro lado, Ambos modelos comparten la característica de que la cobertura de la clase subjetiva supera en gran medida a su precisión. También se observa que dicha característica se invierte completamente para la clase objetiva. Esto es importante, dado que a partir de las fórmulas de estas dos métricas se puede determinar que hay una mayor presencia de falsos positivos, lo que significa principalmente que ambos modelos tienden a clasificar a las oraciones como subjetivas, con una precisión no tan alta.

A su vez, cabe destacar que fueron entrenados modelos cuyos resultados alcanzaban valores considerablemente más altos que los valores que presentan los modelos finales en la evaluación sobre los datos de prueba de la partición elegida, pero su performance en la validación cruzada era muy pobre.

Finalmente, si bien el SVM obtuvo una performance menor en la validación cruzada de la etapa de búsqueda del modelo óptimo, se lo considera un clasificador más fuerte respecto a la red neuronal, por el hecho de mantener su performance en diferentes particiones de datos.

Simulación de Puesta en Producción

Para efectuar la puesta en producción del modelo completo, es necesario contar previamente con un estimador de subjetividad, a fin de tener una base para calcular el SWF-ISF, FRO y FRS. Así, se definen dos alternativas para obtener esta información:

1. Utilizar un estimador basado en la base de datos de entrenamiento, donde por ejemplo se pueda estimar si una oración es subjetiva mediante el valor máximo de la siguiente fórmula aplicada a cada palabra de la oración:

$$\overline{S(x)} = \frac{f_s(x)}{f_o(x)} \quad si\, f_o(x) \neq 0$$

$$\overline{S(x)} = 2 \quad si\, f_o(x) = 0\, y\, f_s(x) \neq 0$$

$$\overline{S(x)} = 0 \quad si\, f_o(x) = 0\, y\, f_s(x) = 0$$

   Donde $f_s(x)$ es la frecuencia absoluta de la palabra x en oraciones subjetivas de la base de datos, y $f_o(x)$ es la frecuencia absoluta de la palabra x en oraciones objetivas de la base de datos. Así, se obtiene el máximo $\overline{S}$ de la oración, y se determina la subjetividad total de la siguiente manera:

$$\overline{s_{max}} > 1 \rightarrow Subjetiva$$

$$\overline{s_{max}} \leq 1 \rightarrow Objetiva$$

   Una desventaja inmediata del uso de este estimador es que es muy dependiente del vocabulario de la base de datos con la que se haya entrenado el modelo. Si se entrenó con textos científicos y se predice en producción con blogs de internet, es probable que no se obtengan buenos resultados. Por otro lado, el estimador resulta muy débil en la práctica, considerando que en principio se desconoce completamente la orientación subjetiva de las oraciones que se quieren clasificar

2. Utilizar textos donde ya se sepa de antemano la tendencia de la subjetividad, por ejemplo si las fuentes de los textos a predecir en producción provienen de noticias o de textos académicos, se sabe que las oraciones tienen una mayor tendencia a ser objetivas, mientras que si se quiere predecir la subjetividad sobre artículos de blogs o novelas, se sabe que es más probable que sean subjetivas.

Una vez definido el estimador base del SWF-ISF, se utiliza el modelo óptimo hallado para predecir una por una las oraciones de los documentos en producción. Nótese que el método de separación en oraciones de los documentos de entrada queda a elección del usuario.

En este marco, se simuló una puesta en producción utilizando como entrada la misma base de datos, y como estimador base de SWF-ISF la *opción 1*. Los resultados obtenidos, como se muestra en la *Tabla 7.6*, fueron desalentadores respecto de la performance que se esperaría del modelo construído, sin embargo, es importante destacar que, como se muestra en la *Tabla 7.5*, el estimador utilizado es débil, apenas superando una clasificación aleatoria, dado que se simuló una situación en la que se desconoce completamente la subjetividad de las oraciones.

	Objetiva	**Subjetiva**
Precisión	1	0.56433
Recall	0.228	1
F-Score	0.37133	0.7215
Predicciones Correctas	61.4%	

Tabla 7.5: Resultados individuales del estimador base

	Objetiva	**Subjetiva**
Precisión	0.92939	0.72817
Recall	0.645	0.951
F-Score	0.76151	0.8248
Predicciones Correctas	79.8%	

Tabla 7.6: Resultados de la simulación de puesta en producción con la base de datos

En conclusión, se observa que la performance del modelo propuesto depende en gran medida del estimador de subjetividad base que utilice, lo cual representa una gran desventaja. Sin embargo, si se tiene algo de información previa acerca de la subjetividad del texto de origen, la performance puede incrementar. Estos factores demuestran que el clasificador puede ser de utilidad para elevar la performance de otros clasificadores de subjetividad, usándolos como estimador base.

Resumen

Se utilizaron las métricas de precisión, recall y f-score para evaluar cada uno de los clasificadores encontrados de forma individual con su partición de datos óptima, y con una validación cruzada con $k = 5$ repetida treinta veces. Por un lado las variables por sí mismas no demostraron ser grandes indicadores de subjetividad, a excepción de la frecuencia relativa de FRS/FRO y el FRO promedio, aunque un análisis de la utilidad de las características en el SVM entrenado revela que las que más impactan en la subjetividad son la frecuencia relativa de FRS/FRO, FRM y el promedio de los tres SWF-ISF máximos. Por otra parte, la performance del perceptrón multicapa decrece en la validación cruzada repetida, mientras que la del SVM se mantiene.

Finalmente, se realiza una simulación de puesta en producción del modelo utilizando el SVM y un estimador de subjetividad base muy simple para calcular las métricas SWF-ISF, FRS y FRO. Los resultados son considerablemente más bajos que los esperados, lo que muestra la dependencia del modelo respecto del estimador base elegido. Esto da indicios de que el modelo puede ser utilizado para mejorar la performance de otros.

Capítulo 8

Conclusiones y Trabajos Futuros

En este último capítulo, se detallan las conclusiones del trabajo y se exploran posibles mejoras y desarrollos futuros que continúen o completen el trabajo realizado. Se exploran temáticas que abordan el uso de diferentes bases de datos y diferentes algoritmos de preprocesamiento y aprendizaje automático.

Conclusiones

En primer lugar, los resultados obtenidos en las experiencias realizadas y el desarrollo del sistema planteado muestran que las características elegidas tienen un impacto considerable en la subjetividad de las oraciones, sobre todo la relación entre FRS y FRO, aunque fue sorprendente haber encontrado que los n-gramas elegidos no fueran de tanta utilidad como se esperaba.

Por otro lado, ambos clasificadores entrenados dieron buenos resultados en su evaluación, aunque se esperaba que su performance fuera aún más alta. Probablemente sería una buena idea continuar explorando distintas características relacionadas a la subjetividad, descartando aquellas que no dieron buenos resultados en este trabajo.

Finalmente, si bien la performance de estos clasificadores es buena, la puesta en simulación del sistema completo probó que es necesario contar con un estimador o clasificador de base para poder obtener predicciones interesantes. Esto lleva a la conclusión de que este desarrollo puede ser utilizado para mejorar el rendimiento de otros modelos existentes, incluso si su performance es baja.

Trabajos Futuros

Se describen a continuación una serie de tópicos que se consideran interesantes para continuar explorando el funcionamiento del modelo planteado en el trabajo, teniendo en cuenta los resultados obtenidos.

- Explorar otras características diferentes a las que se construyeron, o mejorar las que se plantearon. Sería interesante observar cómo se comporta el pipeline al utilizar características que no dependen de ningún estimador base. Tal vez este nuevo modelo y el de este trabajo puedan combinarse para alcanzar un alto grado de precisión.
- Construir una base de datos más grande, con fuentes más variadas, y sobre todo más actualizadas. Los datos utilizados en este trabajo fueron extraídos principalmente de papers académicos y textos literarios antiguos, por lo que hay mucho vocabulario específico presente, y muchas expresiones quizá no se correspondan con las que se utilizan hoy en día. Un buen experimento sería probar el mismo modelo utilizando una base de datos varias veces más grande, y con textos extraídos de blogs y artículos de noticias de los últimos 5 o 10 años. Es muy probable que las predicciones sean mejores si se basan en datos más sólidos como estos.
- Probar el mismo modelo en simulación de producción usando diferentes estimadores o incluso clasificadores de subjetividad existentes, y observar posteriormente cuál es el impacto en el rendimiento total.
- Agregar algún mecanismo de desambiguación semántica en las oraciones para poder determinar más en detalle cuándo un n-grama específico realmente corresponde a una expresión subjetiva.
- Realizar un estudio detallado de n-gramas indicadores de subjetividad en español, a fin de incluirlos en las características de las oraciones y observar cómo impacta en la performance final.
- Entrenar otros algoritmos de aprendizaje automático con las mismas características para ver si pueden obtener mejores resultados que los que fueron entrenados en este trabajo. Podría utilizarse un modelo bayesiano, árboles de decisión, random forests, ada boost, entre otros.
- Llevar a cabo pruebas con algoritmos de deep learning.

Referencias Bibliográficas

1. M. Rajman, R. Besançon, Artificial Intelligence Laboratory, Computer Science Department, Swiss Federal Institute of Technology, Suiza (1997): **Text Mining: Natural Language techniques and Text Mining applications.**
2. Ahmad Kamal, Department of Mathematics, Jamia Millia Islamia – A Central University, India (2013): **Subjectivity Classification using Machine Learning Techniques for Mining Feature-Opinion Pairs from Web Opinion Sources.**
3. José M. Chenlo, David E. Losada, Centro de Investigación en Tecnologías de la Información (CITIUS), Universidad de Santiago de Compostela, España (2013): **A Machine Learning approach for Subjectivity Classification based on Positional and Discourse Features.**
4. Gabriel Murray, Giuseppe Carenini, Department of Computer Science, University of British Columbia, Canadá (2010): **Subjectivity Detection in Spoken and Written Conversations.**
5. Reynier Ortega Bueno, Facultad de Matemática y Computación, Universidad de Oriente, Cuba (2014): **Método No Supervisado para la Detección de Subjetividad.**
6. Janyce M. Wiebe, Department of Computer Science, New Mexico State University, Estados Unidos (1994): **Tracking Point of View in Narrative.**
7. Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze (2008): **An Introduction to Information Retrieval.**
8. Christopher D. Manning, Hinrich Schütze (1999): **Foundations of Statistical Natural Language Processing.**
9. Bing Liu, Department of Computer Science, University of Illinois, Estados Unidos (2010): **Sentiment Analysis and Subjectivity.**
10. Hassan Saif, Miriam Fernández, Yulan He, Harith Alani, Knowledge Media Institute, The Open University, Reino Unido (2014): **On Stopwords, Filtering and Data Sparsity for Sentiment Analysis of Twitter.**
11. Documentación de los stemmers de Snowball: http://snowballstem.org/
12. Zhang Yun-tao, Gong Ling, Wang Yong-cheng, Network and Information Center, School of Electronic and Information Technology, Shanghai Jiaotong University, China (2003): **An Improved TF-IDF Approach for Text Classification.**
13. George Forman, Hewlett-Packard Labs, Estados Unidos (2008): **BNS Feature Scaling: An Improved Representation over TF-IDF for SVM Text Classification.**
14. Stephan Raaijmakers, Khiet Truong, Theresa Wilson (2008): **Multimodal Subjectivity Analysis of Multiparty Conversation.**

15. Peter D. Turney, Institute for Information Technology, Canadá (2002): **Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews.**
16. Ignacio Saporiti, Juan Agustín Tibaldo, Facultad de Informática de la Universidad Nacional de La Plata, Argentina (2017): **Minería de opiniones y visualización de datos aplicables a estudios de mercado.**
17. William C. Mann, Sandra A. Thompson, University of Southern California's Information Sciences Institute, Estados Unidos (1988): **Rhetorical structure theory: Toward a functional theory of text organization.**
18. Wei Zhang, Clement Yu, University of Illinois, Estados Unidos (2007): **UIC at TREC 2007 Blog Track.**
19. Xiaowen Ding, Bing Liu, Philip S. Yu, University of Illinois, Estados Unidos (2008): **A Holistic Lexicon-Based Approach to Opinion Mining.**
20. Nitin Jindal, Bing Liu, University of Illinois, Estados Unidos (2006): **Mining Comparative Sentences and Relations.**
21. Nitin Jindal, Bing Liu, University of Illinois, Estados Unidos (2006): **Identifying Comparative Sentences in Text Documents.**
22. Nitin Jindal, Bing Liu, University of Illinois, Estados Unidos (2007): **Review Spam Detection.**
23. Nitin Jindal, Bing Liu, University of Illinois, Estados Unidos (2008): **Opinion Spam and Analysis.**
24. Anindya Ghose, Panagiotis G. Ipeirotis, Department of Information, Operations, and Management Sciences, New York University (2007): **Designing Novel Review Ranking Systems: Predicting Usefulness and Impact of Reviews.**
25. Soo-Min Kim, Patrick Pantel, Tim Chklovski, Marco Pennacchiotti, University of Southern California, Estados Unidos, University of Rome, Italia (2006): **Automatically Assessing Review Helpfulness.**
26. Eneko Agirre, Aitor Soroa, University of the Basque Country, España (2009): **Personalizing PageRank for Word Sense Disambiguation.**
27. Henry Anaya-Sanchez, Aurora Pons-Porrata, Rafael Berlanga-Llavori, Center of Pattern Recognition and Data Mining, Universidad de Oriente, Cuba (2007): **TKB-UO: Using Sense Clustering for WSD.**
28. Siddharth Patwardhan, Satanjeev Banerjee, Ted Pedersen, University of Utah, Carnegie Mellon University, University of Minnesota, Estados Unidos (2005): **SenseRelate::TargetWord – A Generalized Framework for Word Sense Disambiguation.**
29. Nello Cristianini, John Shawe-Taylor, Cambridge University Press New York, Estados Unidos (1999): **An introduction to support Vector Machines: and other kernel-based learning methods.**
30. Documentación de Sci-Kit Learn: http://scikit-learn.org/stable/user_guide.html
31. José Hernández Orallo, María José Ramírez Quintana, César Ferri Ramírez, Editorial Alhambra S. A. (2004): **Introducción a la minería de datos.**