



Procesamiento de Lenguaje Natural en Sistemas de Análisis de Sentimientos

Tesis de Grado de Ingeniería en Informática

Tesista: Luciana Dubiau
(ldubiau@fi.uba.ar)

Director: Dr. Juan M. Ale
(ale@acm.org)

Facultad de Ingeniería
Universidad de Buenos Aires

Octubre 2013

*“There’s no right,
there’s no wrong,
there’s only popular opinion.”*

Twelve Monkeys, 1995

Resumen

La tarea conocida como Análisis de Sentimientos, dentro del área de estudio del Procesamiento de Lenguaje Natural, consiste en la identificación y extracción de opiniones emitidas en textos con el objetivo de clasificarlos, mediante procesamiento computacional, según la polaridad de las emociones que expresan sobre determinados objetos, situaciones o personas.

La explotación de información subjetiva es una tarea que ha adquirido gran interés en el último tiempo por su potencialidad de aplicación y por la gran cantidad de opiniones no estructuradas que se encuentran disponibles en los distintos medios sociales y que no son analizadas automáticamente. Por ejemplo, resulta interesante para las empresas conocer la opinión de sus clientes sobre sus productos; para los clientes conocer la opinión de otros compradores sobre un producto que desean adquirir; para la sociedad conocer la opinión pública sobre situaciones de actualidad; etc. Esta información puede obtenerse aplicando técnicas de Análisis de Sentimientos y sin necesidad de analizar manualmente las grandes cantidades de resultados que retornan los motores de búsqueda.

En esta tesis se presenta la investigación, análisis, evaluación y comparación experimental de técnicas de Procesamiento de Lenguaje Natural para análisis de información subjetiva como opiniones, sentimientos y emociones en textos no estructurados.

Para validar la investigación presentada hemos desarrollado una herramienta de Análisis de Sentimientos para textos en idioma español que tiene como objetivo clasificar documentos automáticamente según polaridad de emociones (positivos o negativos) y provee las métricas necesarias para evaluar la performance de los distintos modelos de clasificación.

En la etapa de experimentación utilizamos la herramienta desarrollada y dos sitios de crítica online como casos de estudio y los clasificamos en función de parámetros de entrada como tamaño de corpus, tipos de atributos extraídos, preprocesamientos aplicados, algoritmo de clasificación, proporción de documentos de cada clase y conjunto de datos de entrenamiento. Luego, definimos un criterio de evaluación y analizamos los resultados obtenidos utilizando cada técnica estudiada en los distintos escenarios.

Palabras Clave: procesamiento de lenguaje natural, análisis de sentimientos, minería de opiniones, extracción de opiniones, análisis subjetivo, minería de sentimientos.

Abstract

The task known as Sentiment Analysis, as part of the study area of Natural Language Processing, consists of the identification and extraction of opinions issued in texts in order to classify them using computational processing by means of the polarity of emotions expressed about objects, situations or persons.

Subjective information mining is a task that has acquired much interest in recent times because of its application potential and the vast number of unstructured opinions available in social media which are not automatically analyzed. For example, companies find interesting to know their customers' opinions, as well as customers seek for other customers' experiences before buying a product, and even the society expects to know the public opinion on emerging issues. This information can be obtained using Sentiment Analysis techniques, avoiding manual analysis of large number of results from search engines.

This work presents the research, analysis, evaluation and experimental comparison of Natural Language Processing techniques applied to the analysis of subjective information such as opinions, sentiments and emotions expressed in unstructured texts.

In order to validate our research we have developed a Sentiment Analysis tool for Spanish language texts that aims to classify documents automatically, according to sentiment polarity (positive or negative). The tool also provides metrics to evaluate the performance of the classification models.

In the experimentation phase we used the mentioned Sentiment Analysis tool and two review sites as study cases and we classified them based on parameters such as corpus size, type of extracted features, preprocessing techniques, classification algorithms, proportions of documents of each class and training data set. Then we defined an evaluation criterion and we compared the results using each studied technique in different scenarios.

Keywords: natural language processing, sentiment analysis, opinion mining, opinion extraction, subjectivity analysis, sentiment mining.

Agradecimientos

Gracias infinitas a mis padres, Alicia y Federico y a mis hermanas Sil y Vero por confiar en mí siempre y por hacer esto posible. Gracias a mi amor, Alejandro, por su invaluable ayuda emocional y técnica. Gracias a toda mi familia y a mis amigas por estar siempre presentes. Gracias al director de esta Tesis, Dr. Juan M. Ale y a todos los profesores y gente maravillosa que conocí en mi paso por esta casa de estudios.

Índice General

Resumen	II
Agradecimientos	IV
Índice General	V
Índice de Figuras	VIII
Índice de Tablas	XII
1 Introducción	1
1.1 Objetivo	1
1.2 Definición del Problema	1
1.3 Interés del Problema	5
1.4 Contribuciones de esta Tesis	5
1.5 Organización de esta Tesis	6
1.6 Antecedentes del Autor	6
2 Estado del Arte	8
2.1 Definiciones	8
2.2 Modelado del Lenguaje con N-Gramas	9
2.2.1 Premisa de Markov	10
2.2.2 Modelos de N-Grama	10
2.2.3 Maximum Likelihood Estimate (MLE)	11
2.2.4 Smoothing	12
2.2.5 Google N-Grams Datasets	14
2.2.6 Modelado de Lenguaje en tareas de Análisis de Sentimientos	15
2.3 Clasificación de Textos	15
2.3.1 Clasificación basada en Reglas Ad-Hoc	16
2.3.2 Clasificación utilizando técnicas de Machine Learning . .	16
2.3.3 Selección de Features	17
2.3.4 Modelos Generativos vs Discriminativos	19
2.3.5 Clasificador de Naïve Bayes	20
2.3.6 Modelo de Máxima Entropía (MaxEnt Model)	26
2.3.7 Support Vector Machines (SVMs)	31
2.3.8 Decision Trees	32
2.3.9 Polaridad de Turney	34
2.3.10 Clasificación basada en Léxico de Opinión	36
2.3.11 Clasificación basada en Puntaje	38

2.3.12	Clasificación de Múltiples Aspectos	38
2.3.13	Combinación de Clasificadores	41
2.3.14	Evaluación de Clasificadores	42
2.4	Preparación de los datos	47
2.4.1	Preprocesamiento	47
3	Desarrollo de la Herramienta de Análisis de Sentimientos	51
3.1	Análisis del Problema	51
3.1.1	Proceso de Clasificación Supervisado	52
3.1.2	Proceso de Clasificación No Supervisado	53
3.1.3	Parámetros de Clasificación	55
3.1.4	Evaluación de Otros Modelos	56
3.2	Implementación	56
3.2.1	Proceso de Clasificación	59
3.2.2	Frameworks y Herramientas Externas	61
3.2.3	Adaptación del Algoritmo de Turney	62
4	Casos de Estudio y Experimentación	64
4.1	Casos de Estudio	64
4.1.1	Construcción de los Corpus de Datos	66
4.2	Experimentación	68
4.2.1	Selección de Atributos	68
4.2.2	Efectividad de Preprocesadores	69
4.2.3	Efectividad de Clasificadores	71
4.2.4	Cambio de Dominio	77
4.2.5	Corpus Balanceado vs Desbalanceado	79
4.2.6	Análisis de Resultados	83
4.2.7	Dificultades	85
5	Conclusiones y Trabajo Futuro	88
	Referencias	92
	Anexos	
A	Código Fuente, Instalación y Ejecución del Software Desarrollado	93
A.1	Código Fuente	93
A.2	Instalación de Software	93
A.3	Ejecución de la Aplicación	94
A.4	Formato de Resultados	96
B	Tablas Completas de Resultados de Clasificación	98
B.1	Clasificación Supervisada para Corpus Balanceado y un Único Dominio	98
B.1.1	Naïve Bayes	98
B.1.2	MaxEnt	101
B.1.3	SVM	103
B.1.4	Decision Trees	106
B.2	Clasificación No Supervisada para Corpus Balanceado	108

B.3	Clasificación Supervisada para Corpus Balanceado con Cambio de Dominio	109
B.4	Clasificación Supervisada para Corpus Desbalanceado	111
B.5	Clasificación No Supervisada para Corpus Desbalanceado	115
Índice Alfabético		116

Índice de Figuras

1.1	<i>Wheel of Emotions</i> de Plutchik	2
2.1	Source-Channel Model	9
2.2	Google Ngram Viewer	15
2.3	Ejemplo Linear Regression	28
2.4	Support Vector Machines	31
2.5	Ejemplo Árbol de Decisión	33
2.6	Predicción de la Orientación Semántica de Adjetivos: Clustering	38
2.7	Ejemplo Parsing	40
2.8	Combinación de Clasificadores utilizando SVM	41
2.9	10-Fold Cross-Validation	47
3.1	Proceso de Análisis de Sentimientos	52
3.2	Proceso de Clasificación Supervisado	53
3.3	Proceso de Clasificación No Supervisado o Semi-Supervisado	54
3.4	Jerarquía de Clasificadores	57
3.5	Jerarquía de Preprocesadores	58
3.6	Jerarquía de Feature Extractors	59
3.7	Clasificación Supervisada - Diagrama de Secuencia	60
3.8	Clasificación No Supervisada - Diagrama de Secuencia	61
4.1	Caso de Estudio: Guía Óleo	65
4.2	Caso de Estudio: Google Play	66
4.3	Efectividad de Preprocesadores para Clasificación Supervisada: Mejora de Accuracy (%) vs Preprocesos	71
4.4	Efectividad de Clasificadores Supervisados por Atributo: Accuracy vs Tamaño de Corpus	73
4.5	Efectividad de Clasificadores Supervisados por Algoritmo: Accuracy vs Tamaño de Corpus	74
4.6	Efectividad de Clasificadores No Supervisados: Accuracy vs Tamaño de Corpus	75
4.7	Comparación de Clasificadores Supervisados y No Supervisados por Feature: Algoritmo vs Accuracy	77
4.8	Efectividad de Clasificadores Supervisados para Cambio de Dominio: Accuracy vs Tamaño de Corpus	78
4.9	Efectividad de Clasificadores Supervisados por Clase para Cambio de Dominio y Máximo Tamaño de Corpus: Algoritmo vs Accuracy	79

4.10	Efectividad de Clasificadores No Supervisados para Corpus Balanceados y Desbalanceados: F1 vs Tamaño de Corpus	80
4.11	Efectividad de Clasificadores Supervisados para Corpus Balanceados y Desbalanceados: F1 vs Tamaño de Corpus	81
4.12	Efectividad de Clasificadores Supervisados y No Supervisados para Corpus Desbalanceados y Máximo Tamaño de Corpus: Algoritmo vs F1	83
A.1	Resultados de Ejecución	96

Índice de Tablas

1.1	Contraste de Emociones de Plutchik	3
1.2	Contraste de Sentimientos de Plutchik	3
2.1	Ejemplo Multinomial Naïve Bayes - Conjunto de Datos de Entrenamiento	22
2.2	Ejemplo Multinomial Naïve Bayes - Conjunto de Datos de Prueba	23
2.3	Ejemplo Binarized Multinomial Naïve Bayes - Conjunto de Datos de Entrenamiento	24
2.4	Ejemplo Binarized Multinomial Naïve Bayes - Conjunto de Datos de Prueba	25
2.5	Ejemplo Linear Regression - Conjunto de Datos: MPG	27
2.6	Algoritmo de Turney - Patrones	35
2.7	Evaluación de Clasificadores - Recuperación de Textos	42
2.8	Evaluación de Clasificadores - Análisis de Sentimientos	43
2.9	Evaluación de Clasificadores - Corpus de Ejemplo	44
2.10	Evaluación de Clasificadores - Ejemplo de Métricas	44
3.1	Patrones de Turney para el Idioma Español	62
4.1	Efectividad de Preprocesadores para Clasificación Supervisada: Parámetros de la Experiencia	69
4.2	Efectividad de Preprocesadores para Clasificación Supervisada: Valores de Accuracy	70
4.3	Efectividad de Clasificadores Supervisados: Parámetros de la Experiencia	72
4.4	Efectividad de Clasificadores No Supervisados: Parámetros de la Experiencia	75
4.5	Comparación de Clasificadores Supervisados y No Supervisados por Feature: Parámetros de la Experiencia	76
4.6	Comparación de Clasificadores Supervisados y No Supervisados por Feature: Valores de Accuracy	76
4.7	Efectividad de Clasificadores Supervisados para Cambio de Dominio: Parámetros de la Experiencia	78
4.8	Efectividad de Clasificadores para Corpus Balanceados y Desbalanceados: Parámetros de la Experiencia	80
4.9	Efectividad de Clasificadores Supervisados y No Supervisados para Corpus Desbalanceados y Máximo Tamaño de Corpus: Valores de Accuracy y F1	82
4.10	Ejemplos de Dificultades de Clasificación	86

A.1	Ejemplo de Ejecución 1	95
A.2	Ejemplo de Ejecución 2	96
A.3	Resultados de Ejecución: Métricas	97
B.1	Resultados de Clasificación utilizando el Algoritmo Naïve Bayes y Presencia de Unigramas como Features.	98
B.2	Resultados de Clasificación utilizando el Algoritmo Naïve Bayes y Frecuencia de Unigramas como Features.	99
B.3	Resultados de Clasificación utilizando el Algoritmo Naïve Bayes y Presencia de Bigramas como Features.	99
B.4	Resultados de Clasificación utilizando el Algoritmo Naïve Bayes y Presencia de Unigramas y Bigramas como Features.	100
B.5	Resultados de Clasificación utilizando el Algoritmo Naïve Bayes y Presencia de Adjetivos como Features.	100
B.6	Resultados de Clasificación utilizando el Algoritmo MaxEnt y Presencia de Unigramas como Features.	101
B.7	Resultados de Clasificación utilizando el Algoritmo MaxEnt y Frecuencia de Unigramas como Features.	101
B.8	Resultados de Clasificación utilizando el Algoritmo MaxEnt y Presencia de Bigramas como Features.	102
B.9	Resultados de Clasificación utilizando el Algoritmo MaxEnt y Presencia de Unigramas y Bigramas como Features.	102
B.10	Resultados de Clasificación utilizando el Algoritmo MaxEnt y Presencia de Adjetivos como Features.	103
B.11	Resultados de Clasificación utilizando el Algoritmo SVM y Presencia de Unigramas como Features.	103
B.12	Resultados de Clasificación utilizando el Algoritmo SVM y Frecuencia de Unigramas como Features.	104
B.13	Resultados de Clasificación utilizando el Algoritmo SVM y Presencia de Bigramas como Features.	104
B.14	Resultados de Clasificación utilizando el Algoritmo SVM y Presencia de Unigramas y Bigramas como Features.	105
B.15	Resultados de Clasificación utilizando el Algoritmo SVM y Presencia de Adjetivos como Features.	105
B.16	Resultados de Clasificación utilizando el Algoritmo DecisionTrees y Presencia de Unigramas como Features.	106
B.17	Resultados de Clasificación utilizando el Algoritmo DecisionTrees y Frecuencia de Unigramas como Features.	106
B.18	Resultados de Clasificación utilizando el Algoritmo DecisionTrees y Presencia de Bigramas como Features.	107
B.19	Resultados de Clasificación utilizando el Algoritmo DecisionTrees y Presencia de Unigramas y Bigramas como Features.	107
B.20	Resultados de Clasificación utilizando el Algoritmo DecisionTrees y Presencia de Adjetivos como Features.	108
B.21	Resultados de Clasificación utilizando el Algoritmo de Turney adaptado al español.	108
B.22	Resultados de Clasificación Supervisada cambiando de Dominio y utilizando el Algoritmo Naïve Bayes y Presencia de Unigramas como Features.	109

B.23 Resultados de Clasificación Supervisada cambiando de Dominio y utilizando el Algoritmo MaxEnt y Presencia de Unigramas como Features.	109
B.24 Resultados de Clasificación Supervisada cambiando de Dominio y utilizando el Algoritmo SVM y Presencia de Unigramas como Features.	110
B.25 Resultados de Clasificación Supervisada cambiando de Dominio y utilizando el Algoritmo DecisionTrees y Presencia de Unigramas como Features.	110
B.26 Resultados de Clasificación utilizando el Algoritmo Naïve Bayes, Presencia de Unigramas como Features y Corpus Desbalanceado: 80% positivos y 20% negativos.	111
B.27 Resultados de Clasificación utilizando el Algoritmo MaxEnt, Presencia de Unigramas como Features y Corpus Desbalanceado: 80% positivos y 20% negativos.	111
B.28 Resultados de Clasificación utilizando el Algoritmo SVM, Presencia de Unigramas como Features y Corpus Desbalanceado: 80% positivos y 20% negativos.	112
B.29 Resultados de Clasificación utilizando el Algoritmo DecisionTrees, Presencia de Unigramas como Features y Corpus Desbalanceado: 80% positivos y 20% negativos.	112
B.30 Resultados de Clasificación utilizando el Algoritmo Naïve Bayes, Presencia de Unigramas como Features y Corpus Desbalanceado: 20% positivos y 80% negativos.	113
B.31 Resultados de Clasificación utilizando el Algoritmo MaxEnt, Presencia de Unigramas como Features y Corpus Desbalanceado: 20% positivos y 80% negativos.	113
B.32 Resultados de Clasificación utilizando el Algoritmo SVM, Presencia de Unigramas como Features y Corpus Desbalanceado: 20% positivos y 80% negativos.	114
B.33 Resultados de Clasificación utilizando el Algoritmo DecisionTrees, Presencia de Unigramas como Features y Corpus Desbalanceado: 20% positivos y 80% negativos.	114
B.34 Resultados de Clasificación utilizando el Algoritmo Turney adaptado al español y Corpus Desbalanceado: 80% positivos y 20% negativos.	115
B.35 Resultados de Clasificación utilizando el Algoritmo Turney adaptado al español y Corpus Desbalanceado: 20% positivos y 80% negativos.	115

Capítulo 1

Introducción

En este capítulo describimos el objetivo de esta tesis, definimos y presentamos una introducción teórica sobre el problema de Análisis de Sentimientos y por último explicamos las motivaciones y contribuciones de este trabajo.

1.1 Objetivo

El objetivo de esta tesis es investigar, analizar y comparar técnicas de procesamiento de lenguaje natural para la clasificación de documentos a partir de la identificación y extracción de información subjetiva como opiniones, sentimientos y emociones.

La tarea de identificar y extraer información subjetiva de textos no estructurados, mediante procesamiento computacional, es conocida como Análisis de Sentimientos, Extracción de Opiniones, Minería de Opiniones, Minería de Sentimientos o Análisis Subjetivo y tiene como objetivo principal determinar la actitud de un escritor ante determinados objetos, situaciones o personas.

En esta tesis utilizaremos estos términos como sinónimos y nos referiremos a esta tarea como Análisis de Sentimientos (AS).

1.2 Definición del Problema

Las principales definiciones del problema de Análisis de Sentimientos son:

- “Sentiment analysis or opinion mining is the computational study of opinions, sentiments and emotions expressed in text.” (Bing Liu, 2010)
- “Sentiment analysis is the computational study of how opinions, attitudes, emotions, and perspectives are expressed in language.” (Christopher Potts, 2011)
- “Opinion mining and sentiment analysis deals with the computational treatment of opinion, sentiment, and subjectivity in text.” (Bo Pang and Lilian Lee, 2008)

Si bien el análisis computacional de sentimientos está aún lejos de poder utilizar las teorías cognitivas de la emoción o el afecto para tomar decisiones, éstas sí pueden ser muy útiles, por ejemplo, para identificar similitudes y contrastes entre sentimientos o entender etiquetas naturales de los textos (Christopher Potts, 2011), por lo que comenzaremos haciendo una breve introducción al concepto de estados afectivos y actitudes.

Los estados afectivos se dividen en: emoción, humor, posturas interpersonales, actitudes y rasgos de la personalidad.

Los sentimientos son un subtipo de actitudes definidas como: relativamente duraderas, creencias afectivas, preferencias y predisposiciones hacia objetos o personas (Klaus R. Scherer, 1984).

A continuación mostraremos el gráfico de relación de emociones de Plutchik que define ocho emociones básicas y ocho emociones complejas con su composición y contraste.

Esta definición de emociones resulta de gran utilidad en la identificación de opiniones similares y opuestas en sistemas de análisis de sentimientos (ver sección 2.3.10).

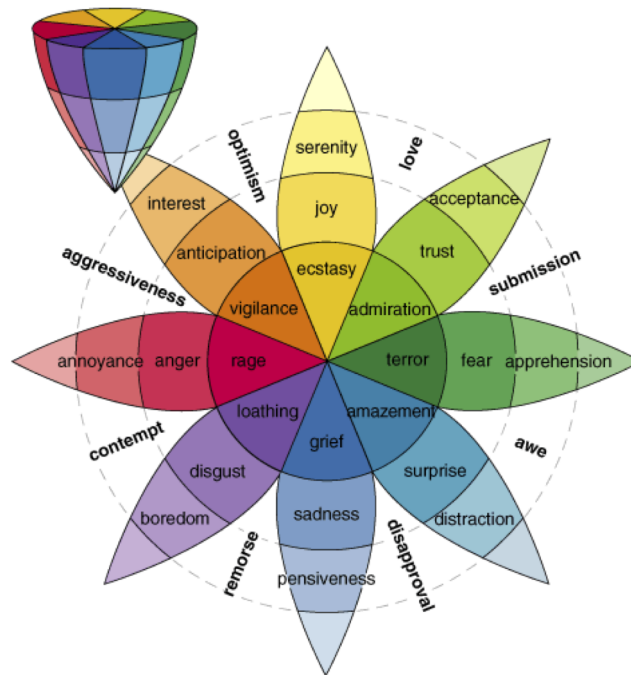


Figura 1.1: *Wheel of Emotions* de Plutchik
(Robert Plutchik, 2002)

El gráfico anterior puede interpretarse de la siguiente manera:

Emoción Básica	Emoción Opuesta
Alegría	Tristeza
Confianza	Disgusto
Miedo	Ira
Sorpresa	Anticipación

Tabla 1.1: Contraste de Emociones de Plutchik

Emociones Básicas	Sentimiento	Sentimiento Opuesto
Anticipación + Alegría	Optimismo	Desaprobación
Alegría + Confianza	Amor	Remordimiento
Confianza + Miedo	Sumisión	Desprecio
Miedo + Sorpresa	Temor	Agresividad
Sorpresa + Tristeza	Desaprobación	Optimismo
Tristeza + Disgusto	Remordimiento	Amor
Disgusto + Ira	Desprecio	Sumisión
Ira + Anticipación	Agresividad	Temor

Tabla 1.2: Contraste de Sentimientos de Plutchik

En su enfoque más general, el análisis computacional de sentimientos consiste en detectar estas emociones, quién las posee (holder); cuál es el aspecto que genera la emoción (target); cuál es el tipo de emoción (me gusta, me encanta, lo valoro, lo odio) o su polaridad (positiva, negativa, neutra); y cuáles son las sentencias que las contienen (Daniel Jurafsky and Christopher D. Manning, 2012).

El tipo de información que puede obtenerse utilizando sistemas de AS incluye:

- Polaridad de sentimientos en críticas sobre arte, productos o servicios
- Nivel de fidelización de clientes
- Opinión pública sobre representantes políticos o situaciones de interés social
- Predicciones sobre resultados de elecciones
- Tendencias de mercado

Existen distintas tareas que pueden realizarse en sistemas de AS: la tarea más simple es la clasificación binaria de la actitud de un texto, en positiva o negativa (también puede existir el neutro); una tarea un poco más compleja

es la clasificación de un texto en múltiples categorías según el grado de polaridad de la actitud dentro de una escala; y la tarea más avanzada es la identificación de los aspectos mencionados en un texto y sus sentimientos asociados.

Por ejemplo, consideremos la siguiente crítica de un restaurante extraída de una guía gastronómica online (www.guiaoleo.com):

Ejemplo 1.1. *Enfoques en Tareas de Análisis de Sentimientos*

“Me gusta mucho este restaurante para comer comida árabe. La comida es fresca y se nota que hay rotación continua. El ambiente es bastante ruidoso pero los mozos son muy amables y los precios muy accesibles.”

El resultado de realizar análisis de sentimientos sobre el texto anterior, puede ser alguno de los siguientes según los enfoques antes mencionados:

1. Clasificación Binaria: Positivo
2. Clasificación en Múltiples Categorías: Rating 4/5
3. Clasificación de Aspectos: {Comida: Positivo, Ambiente: Negativo, Servicio: Positivo, Precio: Positivo}

Algunas de las dificultades más importantes que se encuentran en sistemas de AS (resueltas o en vías de investigación) son:

- Detección de sarcasmo
“Este restaurante es genial! si querés terminar hospitalizado...”
- Neologismos
“Encontramos este lugar googleando”
- Lunfardo
“La cuenta un poco salada, gastamos dos gambas por cabeza”
- Errores gramaticales y ortográficos
“La cómda muy cara para lo que es! Un salteado de lomo mas parecía un salteado de papas.. Poca carne y un exagero de papas por 150 pesos!!!”
- Informalidad en textos
“Maliiiiisimo este lugar no volvemos ni locossssss!”
- Procesamiento de *hashtags* y nombres en textos extraídos de Twitter
“Comiendo #ceviche en @sipanrestaurante, buenísimo!”
- Procesamiento de emoticones
“Todo perfecto hasta que llegó la cuenta :(”
- Capitalizaciones
“Comimos en El Buen Gusto, el servicio es muy malo y la comida regular.”

1.3 Interés del Problema

El análisis de sentimientos es un área donde se han logrado grandes avances pero que aún no se encuentra completamente resuelta ni se utiliza con gran efectividad en aplicaciones comerciales (Daniel Jurafsky and Christopher D. Manning, 2012).

En la actualidad los motores de búsqueda son absolutamente efectivos cuando se trata de información objetiva, pero en cuanto a la información subjetiva aún es un tema pendiente la posibilidad de extraer la opinión pública cuando se busca un determinado tema.

La motivación principal para la realización de esta tesis es la de poder investigar los avances y las dificultades en el área de análisis y extracción de opiniones desestructuradas, por ser éste uno de los grandes temas a resolver en materia de recuperación de información.

Para comprender la potencialidad y el interés en la investigación de sistemas de AS mencionaremos algunos datos estadísticos sobre como las opiniones online de los consumidores impactan en el comportamiento de compra:

- 81% de los usuarios de Internet han hecho una investigación sobre un producto al menos una vez
- 20% lo hacen en un día típico
- sobre los lectores de críticas online de restaurantes, hoteles y servicios varios (agencias de viaje o doctores), entre el 73% y el 87% reportan que esas críticas han tenido una influencia significativa en sus compras
- consumidores reportan estar dispuestos a pagar desde un 20% a un 99% más por un producto clasificado como 5-estrellas que por uno clasificado como 4-estrellas (La variación se debe al tipo de artículo o servicio)
- 32% han clasificado un producto, servicio o persona a través de un sistema online de clasificaciones y un 30% han posteoado un comentario online o crítica sobre un producto o servicio.

Extraído de comScore/the Kelsey group. Online consumer-generated reviews have significant impact on offline purchase behavior, Press Release, 2007 (Bo Pang and Lilian Lee, 2008)

1.4 Contribuciones de esta Tesis

Las contribuciones más relevantes de esta tesis son:

1. Proveer un estudio sobre técnicas, tendencias y dificultades actuales en materia de clasificación subjetiva de textos y tratamiento de opiniones y sentimientos.
2. Elaborar una propuesta de método para análisis de opiniones en castellano.

3. Aportar resultados concretos basados en experiencias sobre la efectividad de los algoritmos de clasificación de textos.
4. Aportar métricas y comparaciones de los algoritmos implementados usando como entrada textos en inglés y castellano y aplicando distintas técnicas de preprocesamiento.
5. Construir una herramienta de análisis de sentimientos que provee distintos algoritmos de clasificación, selección de atributos y preprocesamientos de texto.
6. Aportar el corpus construido para este trabajo como recurso lingüístico en idioma español para otras experiencias.

1.5 Organización de esta Tesis

El resto de esta tesis está organizada como se indica a continuación.

El segundo capítulo consiste en una introducción teórica sobre el estado del arte en materia de modelado de lenguaje natural, clasificación de textos y preprocesamiento de datos. En este apartado se presentan los principales algoritmos, métodos y técnicas utilizadas para análisis subjetivo de textos y evaluación de clasificadores.

En el tercer capítulo se describe la solución de software de análisis de sentimientos desarrollada, se describen las herramientas externas utilizadas, los algoritmos e interfaces implementados y las adaptaciones realizadas para el idioma español.

En el cuarto capítulo presentamos un caso de estudio; definimos los criterios adoptados para la construcción del conjunto de datos; definimos y presentamos los resultados de realizar múltiples experiencias de análisis subjetivo de textos que se ejecutaron utilizando la herramienta desarrollada; presentamos, comparamos y analizamos los resultados obtenidos con cada método estudiado; proponemos mejoras para aumentar la precisión de los clasificadores; y describimos las dificultades actuales en tareas de análisis de sentimientos.

Por último, en el capítulo quinto de esta tesis presentamos las conclusiones finales y el trabajo futuro de esta investigación.

Así también, al final de este trabajo incluimos un índice alfabético de temas y un apéndice con los detalles de instalación, configuración y ejecución de la herramienta.

1.6 Antecedentes del Autor

Previo al desarrollo de esta Tesis el autor publicó el artículo introductorio “Análisis de Sentimientos sobre un Corpus en Español: Experimentación con un Caso de Estudio” (Dubiau and Ale, 2013) en el Simposio Argentino de

Inteligencia Artificial (ASAI) de las 42° Jornadas Argentinas de Informática (JAIIO) organizadas por la S.A.D.I.O.

Capítulo 2

Estado del Arte

En las primeras secciones de este capítulo definiremos algunos conceptos básicos de procesamiento de lenguaje natural que nos permitirán en secciones posteriores presentar las técnicas más utilizadas en la actualidad en tareas clasificación subjetiva de textos. Luego, explicaremos las técnicas de evaluación de los distintos modelos de clasificación presentados y los preprocesamientos de textos utilizados para mejorar los resultados de los clasificadores.

2.1 Definiciones

Documento

Llamamos documento a cada unidad de texto que conforma la colección de datos. En sistemas de AS, un documento es un texto no estructurado compuesto por un conjunto de sentencias o secuencia de palabras que expresan opiniones y emociones.

Corpus

El corpus de datos está compuesto por el conjunto de documentos que se utilizan como entrada para entrenar el sistema de AS (conjunto de datos de entrenamiento) y por el conjunto de documentos que serán clasificados utilizando el sistema de AS (conjunto de datos de prueba)¹.

Léxico de Opinión

Es el conjunto de palabras, o vocabulario, que se utiliza para expresar opinión en un idioma dado.

Feature

En este trabajo utilizaremos el término *feature* para referirnos a características o atributos de los documentos que nos permitan encontrar patrones para predecir la positividad o negatividad de las opiniones expresadas en un texto.

¹Como se verá en próximas secciones, esta distinción existirá si el método de clasificación es supervisado, de otra forma existirá exclusivamente el conjunto de datos a clasificar.

2.2 Modelado del Lenguaje con N-Gramas

En la mayoría de las tareas de procesamiento de lenguaje natural es necesario identificar las sentencias o secuencias de palabras que ocurren con mayor probabilidad dado un contexto.

El objetivo de un modelo de lenguaje es calcular la probabilidad de ocurrencia de una sentencia que es originada por una secuencia de palabras que pasa a través de lo que se conoce como *Noisy Channel*².

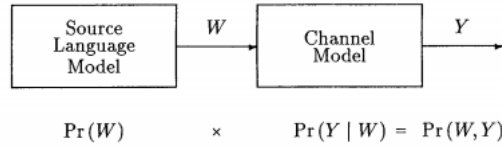


Figura 2.1: Source-Channel Model
(Peter E. Brown and Vincent J. Della Pietra, 1992)

La figura 2.1 muestra el modelo que es utilizado, por ejemplo, en las siguientes tareas: reconocimiento de discurso, donde y es la señal acústica producida; traducción automática de textos, donde y es la sentencia en otro idioma; y en tareas de corrección automática de textos, donde y es la secuencia de caracteres emitida por un escritor imperfecto. En los tres casos, dada una secuencia de salida y se busca predecir la sentencia w que la originó (Peter E. Brown and Vincent J. Della Pietra, 1992).

Se denomina modelo de lenguaje al cálculo computacional de la probabilidad de ocurrencia de una sentencia (Daniel Jurafsky and Christopher D. Manning, 2012).

Los algoritmos que asignan probabilidades a una sentencia pueden ser también utilizados para calcular la probabilidad de la siguiente palabra dada una sentencia. Esto resulta de gran utilidad en tareas de *part-of-speech-tagging*³ (Daniel Jurafsky and James H. Martin, 2009).

La probabilidad de una sentencia puede expresarse como:

$$\begin{aligned}
 P(w_1 w_2 \dots w_i) &= \prod_i P(w_i | w_1 w_2 \dots w_{i-1}) \\
 &= P(w_1) P(w_2 | w_1) P(w_3 | w_1 w_2) \dots P(w_i | w_1 w_2 \dots w_{i-1}) \quad (2.1)
 \end{aligned}$$

Por ejemplo,

²Canal que distorsiona una señal de entrada w transformándola en la señal de salida y .

³Tarea dentro del área de Procesamiento de Lenguaje Natural para asignar etiquetas a cada parte del discurso.

$$\begin{aligned}
 P(\text{"Lo esencial es invisible a los ojos"}) &= P(lo) * P(esencial | lo) * \\
 &P(es | lo esencial) * \\
 &P(invisible | lo esencial es) * \\
 &P(a | lo esencial es invisible) * \\
 &P(los | lo esencial es invisible a) * \\
 &P(ojos | lo esencial es invisible a los)
 \end{aligned}$$

2.2.1 Premisa de Markov

Para calcular las probabilidades mencionadas hasta ahora deberíamos tener un corpus de datos demasiado grande y representativo de todas las secuencias de palabras que son posibles en un idioma dado y que nos permitiera contar la cantidad de veces que aparece cada sentencia.

Como no existe una forma simple de obtener estas probabilidades se utiliza una simplificación que aproxima la probabilidad de una palabra a partir su contexto local, es decir, de las N palabras que la preceden. Esta aproximación se denomina premisa de **Markov** (Daniel Jurafsky and James H. Martin, 2009).

2.2.2 Modelos de N-Grama

Modelo de Unigrama

El modelo de Unigrama es el caso trivial de la premisa de Markov, es decir, aproxima la probabilidad de una palabra con la probabilidad de ocurrencia de esa palabra en el corpus de datos y sin tener en cuenta las palabras que la preceden, es decir, considerando $N = 0$.

$$\hat{P}(\text{ojos} | \text{lo esencial es invisible a los}) = P(\text{ojos}) \quad (2.2)$$

Modelo de Bigrama

El modelo de Bigrama utiliza la premisa de Markov para aproximar la probabilidad de una palabra con la probabilidad de ocurrencia de esa palabra dada la palabra anterior, es decir, considerando $N = 1$. Este modelo se conoce también como Modelo de Markov de primer orden.

$$\hat{P}(\text{ojos} | \text{lo esencial es invisible a los}) = P(\text{ojos} | \text{los}) \quad (2.3)$$

Modelo de Trigramas

El mismo modelo descripto para unigramas y bigramas puede utilizarse teniendo en cuenta las dos palabras anteriores, es decir, considerando $N = 2$. Este modelo se conoce también como Modelo de Trigramas o Modelo de Markov de segundo orden.

$$\hat{P}(\text{ojos} | \text{lo esencial es invisible a los}) = P(\text{ojos} | \text{a los}) \quad (2.4)$$

Modelo de N-grama

Generalizando los modelos anteriores, la probabilidad de una palabra puede aproximarse como la probabilidad de una palabra dadas las N palabras anteriores, lo que se conoce como modelo de N-Grama o Modelo de Markov de orden $N - 1$.

$$\hat{P}(w_i|w_1^{i-1}) = P(w_i|w_{i-N+1}^{i-1}) \quad (2.5)$$

En la ecuación 2.5, w_1^{i-1} representa la historia y w_i la predicción del modelo de lenguaje.

2.2.3 Maximum Likelihood Estimate (MLE)

Se denomina *Maximum Likelihood Estimate*⁴ a la estimación de probabilidades utilizando la frecuencia relativa de una secuencia de palabras en un corpus de entrenamiento (Christopher D. Manning and Hinrich Schütze, 1999).

$$P_{MLE}(w_1..w_i) = \frac{C(w_1..w_i)}{N} \quad (2.6)$$

$$P_{MLE}(w_i|w_1..w_{i-1}) = \frac{C(w_1..w_i)}{C(w_1..w_{i-1})} \quad (2.7)$$

Donde, $C(w_1..w_i)$ es la cantidad de ocurrencias de la sentencia en el corpus de entrenamiento; N es el número total de sentencias; y $C(w_1..w_{i-1})$ es la cantidad de ocurrencias de la secuencia de palabras precedentes.

Ejemplo 2.1. Estimación MLE

Consideremos la sentencia “*un viaje*” y supongamos que queremos estimar la probabilidad de que esta secuencia de palabras esté seguida de la palabra “*a*”, es decir, $P(a|un\ viaje)$.

Se observa en un corpus de entrenamiento que esta sentencia aparece diez veces, de las cuales seis están seguidas de la palabra “*a*”, tres de la palabra “*largo*” y una de la palabra “*desde*”.

La estimación basada en frecuencias relativas asignará las siguientes probabilidades de ocurrencia de la próxima palabra:

$$\begin{aligned} P(a|un\ viaje) &= 0.6 \\ P(largo|un\ viaje) &= 0.3 \\ P(desde|un\ viaje) &= 0.1 \end{aligned}$$

La estimación MLE asignará probabilidad cero a cualquier otro evento que no se encuentre en el corpus de entrenamiento y maximizará la probabilidad de los eventos observados.

⁴En español es conocido como *Estimador de Máxima Verosimilitud*

Como vimos en el ejemplo anterior, el problema que surge de utilizar MLE es que se asignan probabilidades nulas a todos los eventos que no son observados en el corpus de entrenamiento y como los corpus son finitos, existen sentencias válidas que son ignoradas.

Para mejorar esta estimación de probabilidades se utiliza la técnica de Smoothing que describiremos a continuación.

2.2.4 Smoothing

El término *Smoothing*⁵ describe técnicas para ajustar el modelo de *Maximum Likelihood Estimate* para estimar probabilidades en forma más apropiada. Esta técnica tiende a unificar la distribución ajustando las bajas o nulas probabilidades hacia arriba y las altas hacia abajo (Stanley F. Chen and Joshua Goodman, 1998).

A continuación veremos las técnicas de smoothing más utilizadas.

Laplace Smoothing

El método de *Laplace*, también conocido como *Add-One Smoothing* es la técnica más simple y consiste en asumir que cada n-grama aparece una vez más de las que realmente aparece.

Por ejemplo, para modelos de bigramas:

$$P(w_i|w_{i-1}) = \frac{1 + C(w_{i-1}w_i)}{\sum_{w_i} [1 + C(w_{i-1}w_i)]} = \frac{1 + C(w_{i-1}w_i)}{|V| + \sum_{w_i} C(w_{i-1}w_i)} \quad (2.8)$$

Donde $|V|$ es el número de términos del vocabulario del conjunto de datos.

Este mismo método puede generalizarse asumiendo que cada n-grama ocurre δ veces más de las que ocurre realmente ⁶ y se conoce como *Additive Smoothing*.

$$P(w_i|w_{i-N+1}^{i-1}) = \frac{\delta + C(w_{i-N+1}^i)}{\delta|V| + \sum_{w_i} C(w_{i-N+1}^i)} \quad (2.9)$$

Los distintos trabajos de análisis de técnicas de Smoothing muestran que este es un método pobre de ajuste de probabilidades. No es utilizado en modelado de lenguaje porque existen técnicas mejores pero sí se utiliza en tareas de clasificación de textos o en dominios donde el número de ceros no es muy grande.

Linear Interpolation

Esta técnica combina modelos de mayor orden con modelos de menor orden, por ejemplo en modelos de trigramas, se combina con modelos de bigrama y

⁵En español, suavizar, alisar, unificar.

⁶Siendo $0 < \delta < 1$

unigrama que son menos afectados por la falta de ocurrencias en el corpus de entrenamiento.

Las probabilidades estimadas en cada modelo se combinan linealmente ponderadas según su contribución y dando como resultado una nueva probabilidad estimada.

Para el caso de trigramas, la estimación de probabilidad se calcula como:

$$P(w_i|w_{i-2}, w_{i-1}) = \lambda_1 P_1(w_i) + \lambda_2 P_2(w_i|w_{i-1}) + \lambda_3 P_3(w_i|w_{i-1}, w_{i-2}) \quad (2.10)$$

Donde, $0 < \lambda_i < 1$ y $\sum_i \lambda_i = 1$

Generalizando el modelo anterior,

$$P(w|h) = \sum_{i=1}^k \lambda_i(h) P_i(w|h) \quad (2.11)$$

Donde $\forall h, 0 \leq \lambda_i \leq 1$ y $\sum_i \lambda_i = 1$.

Los pesos λ_i pueden definirse manualmente maximizando la probabilidad estimada o utilizando algoritmos numéricos.

Backoff

Esta técnica también combina modelos de mayor orden con modelos de menor orden, utilizando el que corresponda según la evidencia que se tenga en el corpus de entrenamiento.

$$P_{bo}(w_i|w_{i-N+1}^{i-1}) = \begin{cases} \frac{C(w_{i-N+1}^i)}{C(w_{i-N+1}^{i-1})} & \text{if } C(w_{i-N+1}^i) > 0 \\ \alpha P_{bo}(w_i|w_{i-N+2}^{i-1}) & \text{otherwise} \end{cases} \quad (2.12)$$

$$P_{bo}(w_i) = \frac{C(w_i)}{N} \quad (2.13)$$

Esta técnica es utilizada generalmente en modelos de N-Grama muy grandes como la Web (Daniel Jurafsky and Christopher D. Manning, 2012)

Good-Turing Estimate

En la estimación de *Good Turing*, por cada N-Grama que ocurre r veces se asume que ocurre r^* veces.

$$r^* = (r + 1) \frac{n_{r+1}}{n_r} \quad (2.14)$$

Siendo n_r el número de N-gramas que ocurren exactamente r veces en el corpus de entrenamiento.

El primer paso en este método de estimación es obtener la tabla de frecuencias de las diferentes frecuencias y luego se estima la probabilidad como:

$$P_{GT} = r^*/N \quad (2.15)$$

Siendo N la cantidad total de términos del corpus de entrenamiento.

Witten-Bell Smoothing

Esta técnica se basa en el modelo de interpolación lineal. Se utiliza un modelo de mayor orden si el n-grama w_{i-N+1}^i está presente en el corpus de entrenamiento y se utilizan modelos de menor orden en caso contrario.

$$P_{WB}(w_i|w_{i-N+1}^{i-1}) = \lambda_{i-N+1}^{i-1} P_{MLE}(w_{i-N+1}^i) + (1 - \lambda_{i-N+1}^{i-1}) P_{WB}(w_i|w_{i-N+2}^{i-1}) \quad (2.16)$$

Siendo, λ_{i-N+1}^{i-1} la probabilidad de usar un modelo de mayor orden y $1 - \lambda_{i-N+1}^{i-1}$ la probabilidad de que el n-grama no aparezca en el corpus de entrenamiento y se deba utilizar un modelo de menor orden.

Para obtener el parámetro λ se utiliza el número de términos únicos que siguen al n-grama w_{i-N+1}^{i-1} (Stanley F. Chen and Joshua Goodman, 1998).

2.2.5 Google N-Grams Datasets

Cuando no se tiene un corpus de entrenamiento pueden utilizarse datasets disponibles en la web para calcular la probabilidad de ocurrencia de una sentencia dada.

Existen herramientas como Google N-Grams Dataset (Google, 2008) que proveen grandes colecciones de datos en distintos idiomas y permiten calcular estas probabilidades con muy buena precisión.

Además, Google provee una herramienta para visualizar gráficamente la probabilidad de ocurrencia de un conjunto de sentencias dado un idioma y un número de smoothing y utilizando grandes cantidades de libros como corpus de datos.

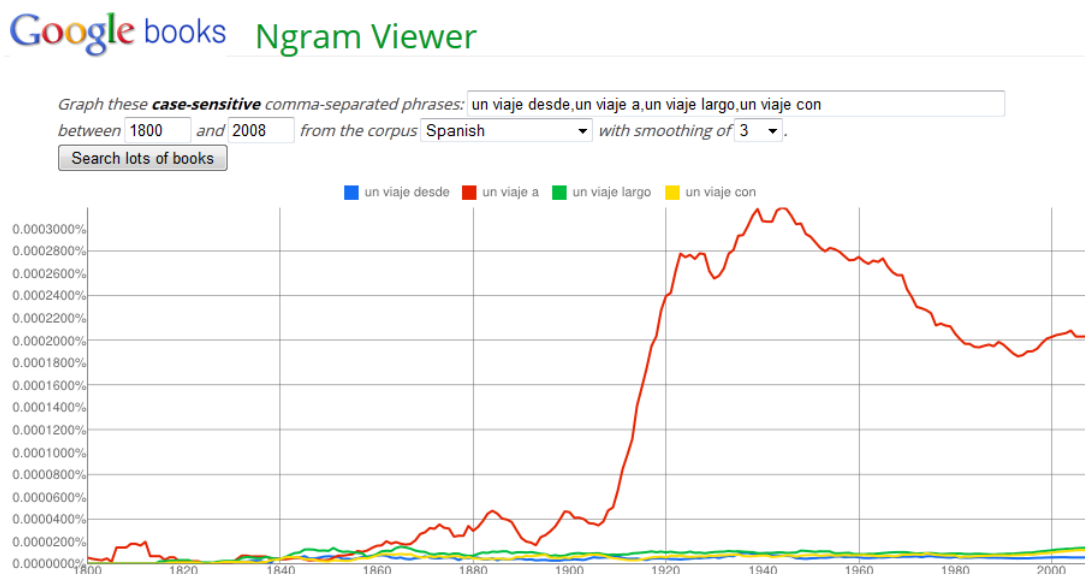


Figura 2.2: Google Ngram Viewer

2.2.6 Modelado de Lenguaje en tareas de Análisis de Sentimientos

En las secciones que siguen veremos cómo se aplican los conceptos de modelado de lenguaje estudiados en este capítulo en algunos de los métodos probabilísticos de clasificación de textos que se utilizan actualmente en tareas de análisis de sentimientos. Además, las técnicas de smoothing presentadas en este capítulo son utilizadas en los distintos métodos de clasificación para el tratamiento de casos que no ocurren en el conjunto de datos de entrenamiento.

2.3 Clasificación de Textos

La utilización de técnicas de procesamiento de lenguaje natural para clasificación de textos consiste principalmente en encontrar patrones y características del lenguaje que permitan asignar una categoría, etiqueta o clase a un documento.

La tarea más común es la clasificación de documentos por tópico a partir de la extracción de información objetiva, como se realiza en sistemas de Information Retrieval⁷ y Question Answering⁸. Otras tareas que pueden resolverse utilizando técnicas de clasificación de textos incluyen detección de spam, identificación del autor de un texto, identificación de idioma, etc. En las tareas mencionadas el objetivo consiste en predecir la clase de un documento, por ejemplo, spam o no spam; autor A o autor B; documento relevante o irrelevante para la búsqueda por tópico, etc.

⁷En español, Recuperación de Información.

⁸En español, Búsqueda de Respuestas

En sistemas de AS pueden utilizarse las mismas técnicas de clasificación que en el caso de información objetiva, pero se buscan características y patrones asociados a opiniones, emociones y sentimientos y el objetivo consiste en predecir la polaridad (positiva o negativa) de un documento, sentencia o aspectos determinados de un texto.

En las secciones que siguen presentaremos las técnicas más utilizadas de clasificación de textos y las distintas tareas de clasificación subjetiva que pueden realizarse sobre un documento.

2.3.1 Clasificación basada en Reglas Ad-Hoc

El método de clasificación más simple consiste en escribir manualmente un conjunto de reglas según las cuales se predice la clase de un texto.

Por ejemplo, en sistemas de AS, el caso trivial será seleccionar un conjunto de palabras, sentencias o patrones⁹ ad-hoc que sugieran opiniones positivas o negativas y realizar la clasificación basada en la presencia o ausencia de estas características en los documentos a analizar.

Si se refinan las reglas apropiadamente puede obtenerse buena precisión en la predicción, sin embargo, esta forma de clasificación requiere de un gran conocimiento del dominio, las reglas son muy difíciles de mantener y poco escalables (Daniel Jurafsky and Christopher D. Manning, 2012).

2.3.2 Clasificación utilizando técnicas de Machine Learning

En esta sección presentaremos una breve introducción al tema de *Machine Learning* que nos permita comprender cómo esta técnica es aplicada en algoritmos de clasificación de textos a partir de información subjetiva.

Se define *Machine Learning*, o *Aprendizaje Automático* en español, como el campo de estudio que le da a las computadoras la habilidad de aprender sin haber sido explícitamente programadas¹⁰ (Arthur Samuel, 1959). Esto significa que el programa debe aprender de la experiencia de realizar una determinada tarea repetidas veces midiendo los resultados.

En el trabajo de Samuel se estudia el entrenamiento de un programa para jugar ajedrez y se muestra como utilizando técnicas de Machine Learning el sistema “aprende” a jugar este juego mejor que su programador. A grandes rasgos, el programa ejecuta un gran número de partidas y registra qué movimientos producen mejores resultados, de esta forma, se dice que su comportamiento depende del aprendizaje obtenido de la experiencia y no de la programación explícita.

Este mismo concepto es utilizado en sistemas de AS, donde el entrenamiento se

⁹Estos patrones pueden representarse utilizando expresiones regulares.

¹⁰Definición original de Samuel: “Machine learning is the field of study that gives computers the ability to learn without being explicitly programmed.”

realiza a partir de un gran número de documentos (experiencias), atributos de esos documentos (features) y resultados conocidos (clases).

Existen distintas formas de aprendizaje que describiremos a continuación.

Aprendizaje Supervisado

Se dice que el aprendizaje es supervisado cuando el entrenamiento se realiza a partir de una gran colección de experiencias, atributos de esas experiencias y resultados correctos en base a los cuales el algoritmo de clasificación predice nuevos resultados correspondientes a experiencias desconocidas con una determinada probabilidad.

En sistemas de AS, el entrenamiento del algoritmo de clasificación se realiza a partir de un gran número de documentos previamente clasificados correctamente (como positivos o negativos) y se busca, a partir de determinados atributos de los documentos, predecir la clase de otros documentos desconocidos con alta probabilidad de acierto.

Aprendizaje No Supervisado

En algoritmos de aprendizaje no supervisado el entrenamiento se realiza a partir de una gran colección de experiencias y atributos de esas experiencias pero no se tienen datos sobre los resultados correctos, es decir, no se conoce la clase a la que pertenece cada experiencia. En este tipo de algoritmos el objetivo es encontrar similitudes, patrones o estructuras en los datos que permitan agruparlos sin conocer previamente los grupos existentes.

Por ejemplo, puede utilizarse aprendizaje no supervisado cuando se tiene una gran base de clientes con sus atributos relevantes y se los quiere agrupar en segmentos sin conocer previamente cuáles son los segmentos de mercado existentes. En este caso, será trabajo del algoritmo encontrar esos segmentos o *clusters*.

Aprendizaje Semi-Supervisado

Se considera que el aprendizaje es semi-supervisado cuando el entrenamiento se realiza a partir de resultados conocidos de algunos de los elementos del conjunto de datos pero no se tiene información del resto.

Como veremos en la sección 2.3.10, en sistemas de AS existen métodos que se consideran semi-supervisados aunque no se conozca la clase de ningún elemento del conjunto de datos sino que sólo se conoce la polaridad de determinados términos que indican opinión (léxico de opinión) que nos permiten obtener la polaridad de un texto.

2.3.3 Selección de Features

En los algoritmos de clasificación supervisados que analizaremos en esta sección, será una tarea clave la definición de los atributos o *features* que extraeremos de los documentos para entrenar el clasificador y luego para clasificar los

documentos cuya clase es desconocida.

Como ya hemos mencionado, un feature es una característica del texto que resulta relevante para una tarea de procesamiento particular. Existen algunos features que resultan de utilidad en la mayoría de las tareas de clasificación y otros que se seleccionan específicamente.

Por ejemplo, si la tarea de clasificación consistiera en identificar nombres propios dentro de un texto¹¹, un feature trivial que podríamos extraer sería “Comienza con mayúscula” y los posibles valores que este atributo tomará en cada documento serán “verdadero” o “falso”.

Cuantos más features que provean información relevante consigamos extraer del texto, mejor “entrenado” estará nuestro clasificador y más precisos serán los resultados, pero se debe tener en cuenta que a medida que se agregan features se podría incurrir en generalizaciones incorrectas y se incrementa el costo de entrenamiento del clasificador por lo que el objetivo será seleccionar aquellos features que provean información más relevante, por ejemplo, pueden refinarse los features dependiendo del dominio.

Así también, podemos mejorar los resultados del clasificador eliminando lo que se conoce como **noisy features**, que son aquellos que no proveen información y cuando se agregan al clasificador incrementan el error. Por ejemplo, si la palabra “queso” aparece siempre en la clase de comentarios positivos entonces nuestro clasificador asignará erróneamente la clase positivo siempre que aparezca el término “queso”. A este tipo de generalización incorrecta se le llama **overfitting**.

Existen tareas en las que no conocemos a priori qué valores de atributos proveen más información al clasificador, en estos casos trabajaremos con atributos genéricos que permitirán encontrar valores de features relevantes para la clasificación.

A continuación mencionaremos algunos ejemplos de features, basados en modelos de lenguaje, que luego utilizaremos en nuestro caso de estudio.

- Bag of Words: Se consideran features todos los términos del corpus de entrenamiento descartando la posición. Cuando se utiliza este feature puede considerarse frecuencia de aparición o no, por lo que podemos hacer la siguiente distinción:
 - Presencia de Unigramas: Por cada documento se evalúa la presencia o ausencia de cada unigrama descartando la información de frecuencia de aparición.
 - Frecuencia de Unigramas: Por cada documento se evalúa la frecuencia con la que aparece cada unigrama.
- Presencia de Bigramas: Se consideran features todos los conjuntos de dos términos consecutivos del corpus de entrenamiento y por cada documento

¹¹Esta tarea es conocida como *Name Entity Recognition (NER)*

se evalúa la presencia o ausencia de cada bigrama.

- **Presencia de Adjetivos:** Se consideran features todos los adjetivos del corpus de entrenamiento y por cada documento se evalúa presencia o ausencia. Este tipo de feature es utilizado en tareas de AS dado que los adjetivos aportan la información más relevante en cuanto a opiniones y sentimientos.

Cuando se tienen demasiados features puede ser necesario eliminar aquellos que aporten menos información al clasificador con el objetivo de mejorar la performance y reducir el costo de entrenamiento. Para esta tarea el método más utilizado consiste en seleccionar los *top N* features (generalmente en porcentaje) ordenados por frecuencia de aparición en el conjunto de datos de entrenamiento.

En aplicaciones reales de análisis de sentimientos resulta fundamental utilizar features y pesos específicos de dominio para mejorar la performance de los clasificadores.

2.3.4 Modelos Generativos vs Discriminativos

En las secciones que siguen presentaremos distintos modelos supervisados de clasificación de textos que pueden ser divididos en Generativos y Discriminativos (también llamados Condicionales).

Los clasificadores generativos aprenden el modelo a partir de la forma en que se generan las observaciones, esto es, a partir de la probabilidad conjunta $P(C, D)$ (donde D es el documento y C la clase) y realizan la predicción maximizando la probabilidad condicional $P(C|D)$ que es calculada utilizando el teorema de Bayes (ver ecuación 2.17). A diferencia de éstos, los clasificadores discriminativos modelan directamente la probabilidad condicional $P(C|D)$ para realizar la predicción, es decir, a partir de la estructura oculta de los datos sin tener en cuenta la forma en que se generan.

La desventaja de los modelos generativos es que resuelven un problema más general como paso intermedio, en cambio, en los modelos discriminativos se resuelve el problema de clasificación directamente (Ng and Jordan, 2001).

Ejemplo 2.2. *Ejemplo de Modelos Generativos vs Modelos Discriminativos*

Si tuviéramos que clasificar el idioma de un texto dado, el enfoque generativo se basaría en aprender el idioma a partir de las observaciones del corpus de entrenamiento para luego predecir en qué idioma está escrito el texto. En el enfoque condicional se buscarían directamente las diferencias lingüísticas de las observaciones para realizar la predicción.

La característica más importante de los modelos discriminativos es que asignan pesos a los features de forma que el modelo corresponda con las observaciones. Esto resuelve el problema de los modelos generativos donde los features son multiplicados aunque aporten la misma información (Daniel Jurafsky and Christopher D. Manning, 2012).

Los modelos generativos más importantes son los modelos de N-Gramas (ver sección 2.2) y Naïve Bayes y los modelos discriminativos más relevantes son Logistic Regression, Maximum Entropy Models, SVM, etc.

2.3.5 Clasificador de Naïve Bayes

Naïve Bayes es un método de clasificación supervisado y generativo, muy similar a los modelos de lenguaje estudiados en la sección 2.2, que se basa en el teorema de Bayes y en la premisa de independencia de los atributos dada una clase. Esta premisa es conocida como “*Naïve Assumption*” y se le llama “naïve” (o ingenua) considerando que en la práctica los atributos raramente son independientes, lo cual en la mayoría de los casos, como veremos en nuestra experiencia y como se observa en los trabajos actuales, no afecta los buenos resultados del método.

Considerando el teorema de Bayes,

$$P(C_i|D) = \frac{P(C_i)P(D|C_i)}{P(D)} \quad (2.17)$$

Donde D es un documento del conjunto de datos de entrenamiento, C_i es cada una de las posibles clases y $P(C_i|D)$ es la probabilidad de que el documento D pertenezca a la clase C_i . El clasificador seleccionará la clase que maximice esta probabilidad.

$P(D)$ puede ser ignorada ya que es la misma para todas las clases y no afecta los valores relativos de probabilidad.

Además, basados en la premisa de independencia de los atributos dada una clase podemos descomponer $P(D|C_i)$ como se ve en la ecuación que sigue:

$$P(C_i|D) \propto P(C_i) \prod_{k=1}^n P(f_k|C_i) \quad (2.18)$$

Donde f_k son los features del documento y $P(f_k|C_i)$ es la probabilidad de ocurrencia del feature en la clase dada.

Por lo tanto, la clase seleccionada por el clasificador sera la que maximice la probabilidad anterior.

$$C_{NB} = \arg \max_i P(C_i) \prod_{k=1}^n P(f_k|C_i) \quad (2.19)$$

Las distintas implementaciones del algoritmo de Naïve Bayes difieren principalmente en la aproximación de $P(f_k|C_i)$ y las técnicas de smoothing utilizadas para el tratamiento de probabilidades bajas o nulas (ver sección 2.2.4).

Multinomial Naïve Bayes

Consideremos el caso de utilizar como features todos los términos del vocabulario del corpus de entrenamiento teniendo en cuenta su frecuencia de aparición; add-one como técnica de smoothing; y MLE (ver sección 2.2.3) para estimar $P(f_k|C_i)$. Este caso es conocido como Naïve Bayes multinomial y la implementación será como sigue:

$$\hat{P}(w_i|C_i) = \frac{\text{count}(w_i, C_i) + 1}{|V| + \sum_w \text{count}(w, C_i)} \quad (2.20)$$

Donde, $\text{count}(w_i, C_i)$ será la cantidad de veces que el término w_i aparece en la clase C_i y $\sum_w \text{count}(w, C_i)$ será la sumatoria de frecuencias de aparición de cada término en la clase C_i .

A continuación veremos la implementación de este algoritmo en lenguaje Python.

Una consideración importante a tener en cuenta en la implementación de algoritmos que calculan probabilidades es que conviene realizar los cálculos en forma logarítmica porque las probabilidades suelen ser valores muy pequeños y al multiplicarlos podemos encontrarnos con un problema de precisión. Por otro lado, esta forma de cálculo tiene la ventaja de que computacionalmente la suma es menos costosa que la multiplicación.

Algoritmo 2.1: Multinomial Naïve Bayes

```
"""
    * Entrenamos el clasificador agregando informacion de
    * cada experiencia (documento),
    * con sus atributos (frecuencia de aparicion de unigramas)
    * y el resultado conocido (clase).
"""
def add_example(self, klass, words):
    self.total_docs += 1
    if klass not in self.classes:
        self.classes[klass] = dict()
        self.classes[klass]['words'] = collections.defaultdict(lambda: 0)
        self.classes[klass]['n_docs'] = 0
        self.classes[klass]['n_words'] = 0

    self.classes[klass]['n_docs'] += 1
    for w in words:
        self.words.add(w)
        self.classes[klass]['words'][w] += 1
        self.classes[klass]['n_words'] += 1

"""
    * Clasificamos un documento desconocido a partir
    * de sus atributos y
    * retornamos la clase que maximice la probabilidad P(C|D)
"""
```

```

def classify(self, words):
    max_prob = None
    predicted_klass = ''

    for klass in self.classes:
        # Calculamos la probabilidad de la clase: P(label)

        prob = math.log(self.classes[klass]['n_docs'])
        prob -= math.log(self.total_docs)

        for w in words:
            """
            * P(feature|label) = count(feature,label) + 1 /
            *                               count(features|label) + |V|
            * Siendo,
            * count(feature,label):
            *   cantidad de veces que aparece el feature en la clase
            * count(features|label):
            *   cantidad de features que tiene la clase
            * |V|:
            *   vocabulario del corpus de entrenamiento
            """

            prob += math.log(self.classes[klass]['words'][w] + 1)
            prob -= math.log(self.classes[klass]['n_words'] + len(self.words))

        if prob > max_prob or max_prob == None:
            max_prob = prob
            predicted_klass = klass

    return predicted_klass

```

Ejemplo 2.3. Clasificación utilizando Multinomial Naïve Bayes

A continuación desarrollaremos un ejemplo muy simplificado de clasificación utilizando Multinomial Naïve Bayes y considerando que todos los documentos son transformados a minúscula y se eliminan los signos de puntuación.

Consideremos el siguiente corpus de datos,

Documento	Texto	Clase
D_1	"La comida es excelente, es un muy buen lugar."	POS
D_2	"Excelente restaurante, excelente servicio, todo es muy recomendable."	POS
D_3	"Horrible lugar, la comida es mala."	NEG
D_4	"La comida es horrible, el servicio es regular"	NEG

Tabla 2.1: Ejemplo Multinomial Naïve Bayes - Conjunto de Datos de Entrenamiento

Documento	Texto	Clase
D_5	<i>“Es excelente la comida, es excelente todo, recomendable.”</i>	?

Tabla 2.2: Ejemplo Multinomial Naïve Bayes - Conjunto de Datos de Prueba

$$\begin{aligned}\hat{P}(C_{POS}|D_5) &= \hat{P}(C_{POS}) * \hat{P}(es|C_{POS}) * \hat{P}(excelente|C_{POS}) * \\ &\quad \hat{P}(la|C_{POS}) * \hat{P}(comida|C_{POS}) * \hat{P}(es|C_{POS}) * \\ &\quad \hat{P}(excelente|C_{POS}) * \hat{P}(todo|C_{POS}) * \\ &\quad \hat{P}(recomendable|C_{POS})\end{aligned}$$

$$\begin{aligned}\hat{P}(C_{NEG}|D_5) &= \hat{P}(C_{NEG}) * \hat{P}(es|C_{NEG}) * \hat{P}(excelente|C_{NEG}) * \\ &\quad \hat{P}(la|C_{NEG}) * \hat{P}(comida|C_{NEG}) * \hat{P}(es|C_{NEG}) * \\ &\quad \hat{P}(excelente|C_{NEG}) * \hat{P}(todo|C_{NEG}) * \\ &\quad \hat{P}(recomendable|C_{NEG})\end{aligned}$$

$$\begin{aligned}\hat{P}(C_{POS}) &= 0.5 \\ \hat{P}(C_{NEG}) &= 0.5\end{aligned}$$

Vocabulario = {la, comida, es, excelente, un, muy, buen, lugar, restaurante, servicio, todo, recomendable, horrible, mala, el, regular}
 $|V| = 16$

$$\begin{aligned}\sum_w count(w, C_{POS}) &= 17 \\ \sum_w count(w, C_{NEG}) &= 14\end{aligned}$$

$$\begin{aligned}\hat{P}(es|C_{POS}) &= \frac{3+1}{16+17} = 0.121 \\ \hat{P}(es|C_{NEG}) &= \frac{3+1}{16+14} = 0.133\end{aligned}$$

$$\begin{aligned}\hat{P}(excelente|C_{POS}) &= \frac{3+1}{16+17} = 0.121 \\ \hat{P}(excelente|C_{NEG}) &= \frac{0+1}{16+14} = 0.033\end{aligned}$$

$$\begin{aligned}\hat{P}(la|C_{POS}) &= \frac{1+1}{16+17} = 0.06 \\ \hat{P}(la|C_{NEG}) &= \frac{2+1}{16+14} = 0.1\end{aligned}$$

$$\begin{aligned}\hat{P}(comida|C_{POS}) &= \frac{1+1}{16+17} = 0.06 \\ \hat{P}(comida|C_{NEG}) &= \frac{2+1}{16+14} = 0.1\end{aligned}$$

$$\begin{aligned}\hat{P}(todo|C_{POS}) &= \frac{1+1}{16+17} = 0.06 \\ \hat{P}(todo|C_{NEG}) &= \frac{0+1}{16+14} = 0.033\end{aligned}$$

$$\hat{P}(recomendable|C_{POS}) = \frac{1+1}{16+17} = 0.06$$

$$\hat{P}(\text{recomendable}|C_{NEG}) = \frac{0+1}{16+14} = 0.033$$

$$\begin{aligned}\hat{P}(C_{POS}|D_5) &= 0.5 * 0.121 * 0.121 * 0.06 * 0.06 * 0.121 * 0.121 * 0.06 * 0.06 \\ &= 1.4e^{-9}\end{aligned}$$

$$\begin{aligned}\hat{P}(C_{NEG}|D_5) &= 0.5 * 0.133 * 0.033 * 0.1 * 0.1 * 0.133 * 0.033 * 0.033 * 0.033 \\ &= 1.05e^{-10}\end{aligned}$$

Dado que $\hat{P}(C_{POS}|D_5) > \hat{P}(C_{NEG}|D_5)$, entonces asignaremos la clase *POS* al comentario D_5 .

Como podemos observar, Naïve Bayes resulta muy simple de implementar y como mostraremos luego en nuestro caso de estudio, se pueden obtener muy buenos resultados. Otra ventaja de este método es que el entrenamiento no resulta tan costoso computacionalmente como ocurre con los métodos que veremos en las secciones que siguen.

Binarized Multinomial Naïve Bayes

En el modelo multinomial visto en la sección anterior, considerábamos la frecuencia de aparición de cada feature para la estimación de $P(f_k|C_i)$. A diferencia de este método, Naïve Bayes puede ser implementado considerando exclusivamente presencia o ausencia de features, es decir, asociándolos con un valor booleano cuando se evalúa un documento y descartando información de frecuencia y posición (Daniel Jurafsky and Christopher D. Manning, 2012).

La implementación de este método se realiza del mismo modo que en el caso de Naïve Bayes Multinomial pero se considera que cada término aparece una única vez en cada documento.

Ejemplo 2.4. Clasificación utilizando Binarized Multinomial Naïve Bayes

En el siguiente ejemplo aplicaremos Binarized Multinomial Naïve Bayes para clasificar el conjunto de datos del ejemplo 2.3. Para esto, eliminaremos los términos duplicados de cada documento.

Documento	Texto	Clase
D_1	“La comida es excelente, es un muy buen lugar.”	POS
D_2	“Excelente restaurante, excelente servicio, todo es muy recomendable.”	POS
D_3	“Horrible lugar, la comida es mala.”	NEG
D_4	“La comida es horrible, el servicio es regular”	NEG

Tabla 2.3: Ejemplo Binarized Multinomial Naïve Bayes - Conjunto de Datos de Entrenamiento

Documento	Texto	Clase
D_5	“Es excelente la comida, es excelente todo, ? recomendable.”	

Tabla 2.4: Ejemplo Binarized Multinomial Naïve Bayes - Conjunto de Datos de Prueba

La estimación de probabilidades será como se indica a continuación:

$$\begin{aligned}\hat{P}(C_{POS}|D_5) &= \hat{P}(C_{POS}) * \hat{P}(es|C_{POS}) * \hat{P}(excelente|C_{POS}) * \\ &\quad \hat{P}(la|C_{POS}) * \hat{P}(comida|C_{POS}) * \hat{P}(todo|C_{POS}) * \\ &\quad \hat{P}(recomendable|C_{POS})\end{aligned}$$

$$\begin{aligned}\hat{P}(C_{NEG}|D_5) &= \hat{P}(C_{NEG}) * \hat{P}(es|C_{NEG}) * \hat{P}(excelente|C_{NEG}) * \\ &\quad \hat{P}(la|C_{NEG}) * \hat{P}(comida|C_{NEG}) * \hat{P}(todo|C_{NEG}) * \\ &\quad \hat{P}(recomendable|C_{NEG})\end{aligned}$$

$$\begin{aligned}\hat{P}(C_{POS}) &= 0.5 \\ \hat{P}(C_{NEG}) &= 0.5\end{aligned}$$

Vocabulario = {la, comida, es, excelente, un, muy, buen, lugar, restaurante, servicio, todo, recomendable, horrible, mala, el, regular}
|V| = 16

$$\begin{aligned}\sum_w count(w, C_{POS}) &= \mathbf{15} \\ \sum_w count(w, C_{NEG}) &= \mathbf{13}\end{aligned}$$

$$\begin{aligned}\hat{P}(es|C_{POS}) &= \frac{2+1}{16+15} = 0.097 \\ \hat{P}(es|C_{NEG}) &= \frac{2+1}{16+13} = 0.103\end{aligned}$$

$$\begin{aligned}\hat{P}(excelente|C_{POS}) &= \frac{2+1}{16+15} = 0.097 \\ \hat{P}(excelente|C_{NEG}) &= \frac{0+1}{16+13} = 0.034\end{aligned}$$

$$\begin{aligned}\hat{P}(la|C_{POS}) &= \frac{1+1}{16+15} = 0.065 \\ \hat{P}(la|C_{NEG}) &= \frac{2+1}{16+13} = 0.103\end{aligned}$$

$$\begin{aligned}\hat{P}(comida|C_{POS}) &= \frac{1+1}{16+15} = 0.065 \\ \hat{P}(comida|C_{NEG}) &= \frac{2+1}{16+13} = 0.103\end{aligned}$$

$$\begin{aligned}\hat{P}(todo|C_{POS}) &= \frac{1+1}{16+15} = 0.065 \\ \hat{P}(todo|C_{NEG}) &= \frac{0+1}{16+13} = 0.034\end{aligned}$$

$$\begin{aligned}\hat{P}(recomendable|C_{POS}) &= \frac{1+1}{16+15} = 0.065 \\ \hat{P}(recomendable|C_{NEG}) &= \frac{0+1}{16+13} = 0.034\end{aligned}$$

$$\begin{aligned}
 \hat{P}(C_{POS}|D_5) &= 0.5 * 0.097 * 0.097 * 0.065 * 0.065 * 0.065 * 0.065 \\
 &= 8.4e^{-8} \\
 \hat{P}(C_{NEG}|D_5) &= 0.5 * 0.103 * 0.034 * 0.103 * 0.103 * 0.034 * 0.034 \\
 &= 2.1e^{-8}
 \end{aligned}$$

En este caso también se cumple que $\hat{P}(C_{POS}|D_5) > \hat{P}(C_{NEG}|D_5)$ por lo que asignaremos la clase *POS* al comentario D_5 .

Más adelante en este trabajo veremos como mejorar los resultados del clasificador eliminando algunos features que no aportan información relevante en tareas de AS.

Como ya se ha mostrado en trabajos previos (Bo Pang, Lillian Lee and Shivakumar Vaithyanathan, 2002) y como comprobaremos luego en nuestro caso de estudio, en tareas de AS, considerar presencia de términos arroja mejores resultados que considerar frecuencia, a diferencia de lo que ocurre en tareas de clasificación por tópico.

Otra forma de aplicar Naïve Bayes consiste en utilizar el modelo Multivariate Bernoulli, pero ya se ha visto que no funciona bien en tareas de AS (Daniel Jurafsky and Christopher D. Manning, 2012), por lo que no analizaremos este modelo.

2.3.6 Modelo de Máxima Entropía (MaxEnt Model)

El modelo de Máxima Entropía, conocido también como Multinomial Logistic Regression, es un método de clasificación discriminativo y de aprendizaje supervisado donde los documentos del conjunto de datos son descritos a partir de una lista de features, siendo cada uno de estos features una restricción del modelo. Este método se basa en seleccionar la distribución de probabilidad que satisfaga todas las restricciones del modelo y maximice la entropía. Esto apunta a preservar la incertidumbre tanto como sea posible (Christopher D. Manning and Hinrich Schütze, 1999).

Cada feature f_i es una función binaria que puede ser utilizada para caracterizar una propiedad del par (\vec{x}, c) donde \vec{x} es un vector que representa un documento y c , la clase:

$$f_i(\vec{x}_j, c) = \begin{cases} 1 & \text{si el feature } i \text{ esta presente en el documento } j \text{ y} \\ & j \text{ pertenece a la clase } c \\ 0 & \text{en otro caso} \end{cases} \quad (2.21)$$

Log-Linear Classifier

MaxEnt pertenece a la familia de clasificadores conocidos como exponenciales o *log-linear* y se basa en extraer features de las observaciones y combinarlos linealmente ponderados con un determinado peso como se indica en la ecuación que sigue (Daniel Jurafsky and James H. Martin, 2009):

$$P(c|x) = \frac{1}{Z} \exp\left(\sum_i w_i f_i\right) \quad (2.22)$$

Siendo, w_i el peso del feature f_i y Z un factor de normalización para que las probabilidades sumen 1.

Linear Regression

Los clasificadores *log-linear* utilizan el concepto de *Linear Regression*¹² para la combinación de features. Este método tiene el objetivo de predecir un valor, perteneciente a los números reales, a partir de la combinación lineal de los valores que toman determinados atributos en cada observación del modelo¹³.

Ejemplo 2.5. Linear Regression

Consideremos las siguientes observaciones de un conjunto de datos de consumo de combustible en millas por galón (MPG) a partir de los atributos de determinados autos¹⁴:

mpg	cylinders	horsepower	weight	acceleration	car name
18	8	307	3504	12	chevrolet chevelle malibu
15	8	350	3693	11.5	buick skylark 320
16	8	304	3433	12	amc rebel sst
17	8	302	3449	10.5	ford torino
15	8	429	4341	10	ford galaxie 500
14	8	455	3086	10	buick estate wagon (sw)
21	6	200	2587	16	ford maverick
25	4	104	2375	17.5	saab 99e
26	4	121	2234	12.5	bmw 2002
21	6	199	2648	15	amc gremlin

Tabla 2.5: Ejemplo Linear Regression - Conjunto de Datos: MPG

A continuación graficaremos el consumo de combustible en función del atributo “weight” y la línea de regresión que se ajusta mejor a las observaciones del conjunto de datos:

¹²En español, Regresión Lineal.

¹³En estadística, el término *regression* indica que el resultado es un valor perteneciente a los números reales.

¹⁴Muestra extraída de <http://archive.ics.uci.edu/ml/datasets/Auto+MPG>

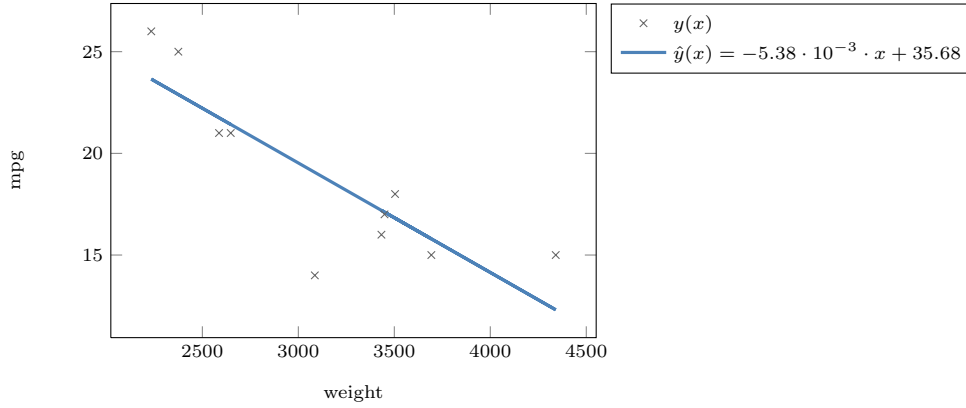


Figura 2.3: Ejemplo Linear Regression

La línea de regresión del gráfico 2.3 fue calculada a partir de la ecuación que define la recta $y = m \cdot x + b$ y las restricciones que imponen las observaciones del conjunto de datos considerando que \mathbf{x} es el atributo “weight”; \mathbf{m} la ponderación; e \mathbf{y} el valor de consumo de combustible. Por lo tanto, a partir de la ecuación que sigue podremos estimar el consumo de combustible para cualquier vehículo.

$$\hat{mpg} = 35.6809180528 - 0.0053846629 * valor_{weight}$$

En la práctica seleccionaremos una gran cantidad de features¹⁵ del conjunto de datos por lo que representaremos cada observación como una combinación de los valores que esos features seleccionados adquieren y ponderados por un peso determinado. Si en el ejemplo 2.5 seleccionamos todos los features del conjunto de datos excepto el nombre del auto que no aporta información relevante, la predicción del consumo será como sigue:

$$\begin{aligned} \hat{mpg} = & w_0 + w_1 * valor_{cylinders} + w_2 * valor_{horsepower} + \\ & w_3 * valor_{weight} + w_4 * valor_{acc} \end{aligned}$$

Más adelante en este capítulo mencionaremos algunos algoritmos que pueden utilizarse para el cálculo de los pesos que ponderan los features en modelos de máxima entropía.

Generalizando lo anterior, la ecuación del método de *linear regression* será:

$$y = \sum_{i=0}^N w_i \times f_i \quad (2.23)$$

Logistic Regression

Hasta ahora vimos el método de regresión lineal donde el valor a predecir pertenecía a los números reales. Sin embargo, en las tareas de clasificación de textos que estamos estudiando, la predicción toma un valor discreto, por

¹⁵Cuando usamos más de un feature el método se conoce como Multiple Linear Regression, o en español, Regresión Lineal Múltiple.

ejemplo, para indicar que el texto pertenece a determinada clase, la variable y tomará el valor 1 (verdadero) o 0 (falso) y este valor está dado con una probabilidad de acierto.

Por lo tanto, aplicando lo anterior a la ecuación de regresión lineal, obtendremos:

$$P(y = true|x) = \sum_{i=0}^N w_i \times f_i \quad (2.24)$$

Sin embargo, lo anterior no es un modelo probabilístico válido dado que no estamos asegurando que la probabilidad sea un valor entre 0 y 1. Para resolver esto se utiliza lo que se conoce como **logit-function**:

$$logit(p(x)) = \ln\left(\frac{p(x)}{1 - p(x)}\right) \quad (2.25)$$

Aplicando la función *logit* al modelo lineal anterior, obtendremos:

$$\ln\left(\frac{p(y = true|x)}{1 - p(y = true|x)}\right) = w \cdot f_i \quad (2.26)$$

Escrito de otra forma:

$$p(y = true|x) = \frac{\exp(\sum_{i=0}^N w_i f_i)}{1 + \exp(\sum_{i=0}^N w_i f_i)} \quad (2.27)$$

El modelo que utiliza una función lineal para estimar la probabilidad utilizando la función *logit* se conoce como **logistic regression** (Daniel Jurafsky and James H. Martin, 2009).

Clasificador utilizando un Modelo de Máxima Entropía

Generalizando el modelo de *logistic regression* obtenemos la siguiente ecuación para modelos de máxima entropía:

$$P(c|x) = \frac{\exp(\sum_{i=0}^N w_{ci} f_i)}{\sum_{c' \in C} \exp(\sum_{i=0}^N w_{c'i} f_i)} \quad (2.28)$$

Siendo el denominador el factor de normalización Z de la ecuación 2.22.

Será tarea del clasificador seleccionar la clase que maximice la probabilidad anterior.

$$C_{MAX_ENT} = \arg \max_{c \in C} \frac{\exp(\sum_{i=0}^N w_{ci} f_i)}{\sum_{c' \in C} \exp(\sum_{i=0}^N w_{c'i} f_i)} \quad (2.29)$$

Cálculo de Pesos en Modelos de Máxima Entropía

El cálculo de pesos en modelos de máxima entropía es un problema complejo conocido como **convex optimization** (Daniel Jurafsky and James H. Martin, 2009) que busca maximizar la verosimilitud del modelo. La resolución de este problema excede el alcance de esta tesis por lo que sólo mencionaremos algunos métodos de optimización numérica que luego utilizaremos en nuestro caso de estudio tales como:

- Gradient Descendent (GD)
- Generalized Iterative Scaling (GIS)
- Improved Iterative Scaling (IIS)
- Conjugate Gradient (CG)
- Métodos Quasi Newton (L-BFGS)
- Optimización de CG y LM-BFGS (Hal Daumé III, 2004)

Ejemplo 2.6. Modelo de Máxima Entropía

Consideremos el siguiente ejemplo donde se busca clasificar todos los términos de un documento dadas las siguientes clases: {PERSONA, LOCALIDAD, VERBO, OTRO} y el siguiente conjunto de features:

$$\begin{aligned} f_1(c, d) &= [c = PERSONA \ \& \ mayuscula(w)] \\ f_2(c, d) &= [c = LOCALIDAD \ \& \ mayuscula(w)] \\ f_3(c, d) &= [c = LOCALIDAD \ \& \ w_{i-1} = "en"] \\ f_4(c, d) &= [c = VERBO \ \& \ finaliza("ar", "er", "ir")] \end{aligned}$$

Siendo, los pesos de los features: λ_1 , λ_2 , λ_3 y λ_4 respectivamente.

Clasifiquemos ahora el término “Argentina” extraído del documento “en Argentina”:

$$P(PERSONA|en \ Argentina) = \frac{e^{\lambda_1}}{e^{\lambda_1} + e^{\lambda_2}e^{\lambda_3} + e^0 + e^0}$$

$$P(LOCALIDAD|en \ Argentina) = \frac{e^{\lambda_2}e^{\lambda_3}}{e^{\lambda_1} + e^{\lambda_2}e^{\lambda_3} + e^0 + e^0}$$

$$P(VERBO|en \ Argentina) = \frac{e^0}{e^{\lambda_1} + e^{\lambda_2}e^{\lambda_3} + e^0 + e^0}$$

$$P(OTRO|en \ Argentina) = \frac{e^0}{e^{\lambda_1} + e^{\lambda_2}e^{\lambda_3} + e^0 + e^0}$$

Luego, se obtendrán los pesos utilizando alguno de los métodos mencionados y se seleccionará aquella clase que maximice la probabilidad anterior.

En nuestra sección de experimentación utilizaremos la herramienta *megam* (Hal Daumé III, 2004) para el cálculo de pesos en algoritmos de máxima entropía en base a los buenos resultados obtenidos y velocidad de convergencia.

2.3.7 Support Vector Machines (SVMs)

Se denomina *Support Vector Machines* (o *Support Vector Networks*) al método supervisado de clasificación binaria que a diferencia de los métodos vistos hasta ahora no se basa en modelos probabilísticos sino que el entrenamiento consiste en encontrar un hiperplano¹⁶ que separe los vectores que representan los documentos del conjunto de datos (vectores de features) en dos grupos, siendo esta separación la más grande posible. Aquellos vectores que definen los márgenes de la máxima separación entre las clases se conocen como *support vectors* y pueden observarse recuadrados en la figura 2.4 (Cortes and Vapnik, 1995).

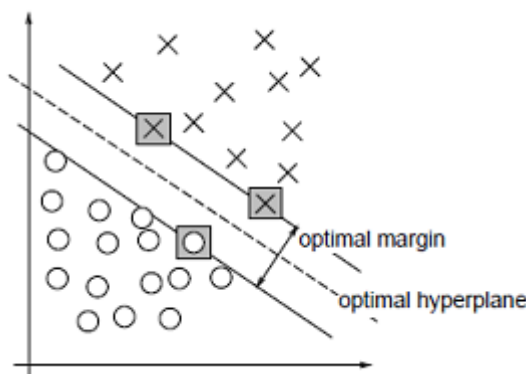


Figura 2.4: Support Vector Machines

A pesar de ser modelos totalmente distintos en cuanto a su concepción teórica, SVM comparte algunas características con los modelos de MaxEnt estudiados, como por ejemplo, la incorporación de información relevante a partir de las restricciones impuestas por los features del conjunto de datos de entrenamiento.

En el espacio de features representaremos el hiperplano óptimo según la siguiente ecuación:

$$\vec{w} \cdot \vec{x} + b = 0 \quad (2.30)$$

Siendo, \vec{w} el vector normal de pesos perpendicular al hiperplano; \vec{x} el vector de features; y b el término independiente que nos permitirá elegir entre todos los

¹⁶En geometría un hiperplano es una división del espacio en dos partes. En \mathbb{R}^1 , será un punto que divide una recta; en \mathbb{R}^2 será una recta que divide un plano; en \mathbb{R}^3 será un plano que divide el espacio. Esto mismo puede ser generalizado para espacios de más dimensiones.

vectores perpendiculares al vector normal.

Los pesos \vec{w} pueden representarse como una combinación lineal de los vectores de features del conjunto de datos de entrenamiento.

$$\vec{w} = \sum_{\text{feature vectors}} \alpha_i \vec{x}_i \quad (2.31)$$

Los valores de α se obtienen resolviendo un problema de optimización que puede ser consultado en el trabajo de (Cortes and Vapnik, 1995). Los vectores de features para los que α_i resulte un valor mayor que cero serán los *support vectors* que son aquellos que hacen un aporte al vector de pesos \vec{w} .

Por lo tanto, la ecuación anterior se puede redefinir como:

$$\vec{w} = \sum_{\text{support vectors}} \alpha_i \vec{x}_i \quad (2.32)$$

Si ahora consideramos un conjunto de datos de entrenamiento $D = \{(\vec{x}_i, y_i)\}$, donde \vec{x}_i es el vector de features del documento e y_i la clase y tenemos en cuenta que en SVM los valores de clases siempre serán $+1$ o -1 , la función de decisión del clasificador será como sigue:

$$f(\vec{x}) = \text{sign}(\vec{w} \cdot \vec{x} + b) \quad (2.33)$$

Siendo \vec{w} ,

$$\vec{w} = \sum_i \alpha_i \vec{x}_i \quad (2.34)$$

Por lo tanto,

$$f(\vec{x}) = \text{sign}\left(\sum_i \alpha_i \vec{x}_i \cdot \vec{x} + b\right) \quad (2.35)$$

En la ecuación anterior, un valor de -1 indicará que el documento pertenece a una clase y un valor de $+1$ a la otra, lo que representa de qué lado del hiperplano se encuentra el vector de features \vec{x} que se quiere clasificar.

Una de las herramientas más utilizadas para la clasificación de textos en base a modelos de SVM es SVM^{light} (Joachims, 1999). En nuestra sección de experimentación utilizaremos la implementación provista por el framework *Scikit-learn* (Pedregosa et al., 2011) por los buenos resultados obtenidos y la mejora en tiempos de procesamiento.

2.3.8 Decision Trees

Se conoce como *Decision Trees* al método de clasificación supervisado en el que el entrenamiento del modelo consiste en la construcción de un árbol de decisión a partir de los valores que toman los features en los documentos del conjunto de datos y la clasificación se realiza en base a reglas inferidas de este modelo.

Por ejemplo, consideremos el siguiente conjunto de features con el que entrenaremos un clasificador basado en árboles de decisión: {contiene('excelente'), contiene('malo'), contiene('comida'), contiene('fria')}

A partir de los valores que tomen estos features en el conjunto de datos de entrenamiento podríamos obtener un árbol de decisión como el que sigue:

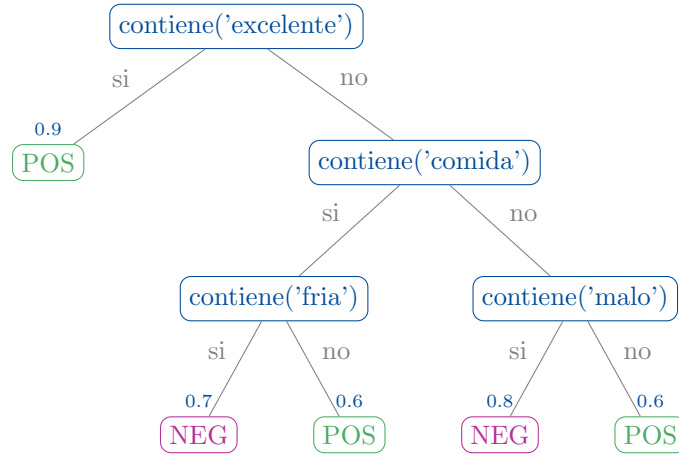


Figura 2.5: Ejemplo Árbol de Decisión

A partir de la figura 2.5 la predicción será:

- Los comentarios que contengan el término “excelente” serán calificados como “positivo” con un 90% de probabilidad de acierto.
- Los comentarios que no contengan el término “excelente” y contengan los términos “comida” y “fria” serán calificados como “negativo” con un 70% de probabilidad de acierto.
- Los comentarios que no contengan el término “excelente” y contengan el términos “comida” pero no el término “fria” serán calificados como “positivo” con un 60% de probabilidad de acierto.
- Los comentarios que no contengan los términos “excelente” y “comida” y contengan el término “malo” serán calificados como “negativo” con un 80% de probabilidad de acierto.
- Los comentarios que no contengan los términos “excelente”, “comida” y “malo” serán calificados como “positivo” con un 60% de probabilidad de acierto.

Para construir el árbol de decisión existen los siguientes algoritmos:

ID3¹⁷ es un método creado por Ross Quinlan (Quinlan, 1986) que construye un árbol de múltiples caminos buscando para cada nodo el feature discreto que

¹⁷Iterative Dichotomiser 3

provee la mayor ganancia de información para la clase. El árbol crece hasta su tamaño máximo y luego es acotado para mejorar su capacidad de generalización para los datos que no ocurren en el conjunto de entrenamiento.

C4.5 es una extensión de ID3 por lo que la construcción del árbol se realiza también en base a la ganancia de información que provee cada atributo. Una mejora significativa que introduce C4.5 es que puede manejar valores de atributos continuos y discretos¹⁸. Además, este algoritmo convierte el árbol de decisión en un conjunto de reglas *si-entonces* y evalúa la exactitud de cada regla para definir el orden en el que deben aplicarse. Para acotar el árbol se realiza una poda que elimina aquellas precondiciones que al removerlas producen una mejora en la exactitud de la regla, esta mejora respecto de ID3 se introduce para evitar el *overfitting* (ver sección 2.3.3).

La implementación más utilizada de este algoritmo se conoce como **J48**, está desarrollada en Java y puede encontrarse en el framework *Weka* (Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, Ian H. Witten, 2009).

C5.0 es la versión más actual del algoritmo de Ross Quinlan. Introduce mejoras en el manejo de memoria y construye un conjunto más pequeño de reglas que C4.5 obteniendo mejores resultados.

CART¹⁹ es un algoritmo no paramétrico de aprendizaje automático basado en árboles de decisión que permite construir árboles de clasificación o regresión dependiendo del tipo de atributos, discretos o continuos, respectivamente. La construcción del árbol también se realiza a partir de la ganancia de información que proveen los atributos, dividiendo cada nodo en dos recursivamente y hasta que no pueda obtenerse más ganancia o hasta que una precondición para acotar el árbol se cumpla. Como criterio para la división de nodos, este algoritmo utiliza el *Índice de Gini* como medida de la impureza que existe en los subconjuntos de entrenamiento generados al dividir el árbol a partir de un atributo.

El framework *Scikit-learn* (Pedregosa et al., 2011) implementa una versión optimizada de este algoritmo que utilizaremos en nuestra sección de experimentación.

2.3.9 Polaridad de Turney

Turney propone en su trabajo (Peter Turney, 2002) un método no-supervisado para la clasificación de textos según información subjetiva que se basa en predecir la polaridad de un documento en base a su orientación semántica. La orientación de un documento es calculada en base a la distancia a los términos del idioma inglés, “*excellent*” y “*poor*” que sugieren referencias positivas y negativas respectivamente.

El algoritmo propuesto consiste en los siguientes 3 pasos:

¹⁸Para atributos continuos se define dinámicamente un atributo discreto particionando el valor continuo del atributo en un conjunto discreto de intervalos.

¹⁹Classification And Regression Trees

1. Extraer las frases del documento que contienen adjetivos o adverbios utilizando una herramienta de *part-of-speech tagging* (esta técnica es descrita en la sección 2.4.1 de preprocesamiento de textos).
2. Estimar la orientación semántica de cada frase extraída utilizando el algoritmo PMI-IR.
3. Asignar las clases “recomendado” o “no recomendado” al documento.

En el primer paso se extraerán grupos de dos palabras cuyas etiquetas correspondan con los siguientes patrones:

Primera Palabra	Segunda Palabra	Tercera Palabra (no extraída)
JJ	NN, NNS	cualquier palabra
RB, RBR, RBS	JJ	palabra que no sea NN ni NNS
JJ	JJ	palabra que no sea NN ni NNS
NN, NNS	JJ	palabra que no sea NN ni NNS
RB, RBR, RBS	VB, VBD, VBN, VBG	cualquier palabra

Tabla 2.6: Algoritmo de Turney - Patrones

Siendo, JJ: Adjetivos; NN: Sustantivos en Singular; NNS: Sustantivos en Plural; RB: Adverbios; RBS: Adverbios Superlativos; RBR: Adverbios Comparativos; VB: Verbos; VBD: Verbos en Pasado; VBN: Verbos en Pasado Participio; VBG: Verbos Gerundios o Pasado Participio.

Algoritmo PMI-IR

Este algoritmo utiliza la técnica de **Pointwise Mutual Information** (PMI) para obtener una medida de la asociación semántica entre dos palabras según la ecuación que sigue:

$$PMI(word1, word2) = \log_2 \left[\frac{p(word1 \& word2)}{p(word1)p(word2)} \right] \quad (2.36)$$

Siendo $p(word1 \& word2)$ la probabilidad de que las palabras ocurran juntas.

Luego, la orientación semántica se calculará de la siguiente manera:

$$SO(phrase) = PMI(phrase, "excellent") - PMI(phrase, "poor") \quad (2.37)$$

El algoritmo PMI-IR estima el valor de PMI realizando consultas a un motor de búsqueda²⁰ a partir de la cantidad de resultados que devuelve la consulta y utilizando el operador *NEAR* que proveen algunos buscadores para estimar la ocurrencia conjunta de dos términos.

$$SO(phrase) = \log_2 \left[\frac{hits(phrase \text{ NEAR } "excellent")hits("poor")}{hits(phrase \text{ NEAR } "poor")hits("excellent")} \right] \quad (2.38)$$

²⁰Information Retrieval (IR) o en español, Recuperación de Información

Por último la clasificación de un documento como “recomendable” o “no recomendable” se realizará a partir del cálculo de la orientación semántica promedio de todas las frases extraídas del documento.

Ejemplo 2.7. *Algoritmo de Turney*

En este ejemplo aplicaremos el algoritmo de Turney con algunas adaptaciones para un corpus en español. Consideraremos los términos “excelente” como referencia de polaridad positiva y los términos “mal, malo, mala” como referencia negativa y estimaremos el operador *NEAR* a partir de la ocurrencia conjunta de palabras en la misma sentencia:

Consideremos las frases “buena comida” y “comida fria” para el cálculo de orientación semántica:

$$\begin{aligned} \text{count}(\text{“buena comida”}) &= 809 \\ \text{count}(\text{“excelente”}) &= 14749 \\ \text{count}(\text{“mal/o/a”}) &= 5847 \\ \text{count}(\text{“buena comida” NEAR “excelente”}) &= 264 \\ \text{count}(\text{“buena comida” NEAR “mal/o/a”}) &= 29 \end{aligned}$$

$$\begin{aligned} SO(\text{“buena comida”}) &= \log_2 \left[\frac{\text{hits}(\text{“buena comida” NEAR “excelente”}) \text{hits}(\text{“mal/o/a”})}{\text{hits}(\text{“buena comida” NEAR “mal/o/a”}) \text{hits}(\text{“excelente”})} \right] \\ &= \frac{264 \cdot 5847}{29 \cdot 14749} \\ &= 1.85 \end{aligned}$$

$$\begin{aligned} \text{count}(\text{“comida fria”}) &= 69 \\ \text{count}(\text{“excelente”}) &= 14749 \\ \text{count}(\text{“mal/o/a”}) &= 5847 \\ \text{count}(\text{“comida fria” NEAR “excelente”}) &= 4 \\ \text{count}(\text{“comida fria” NEAR “mal/o/a”}) &= 26 \end{aligned}$$

$$\begin{aligned} SO(\text{“comida fria”}) &= \log_2 \left[\frac{\text{hits}(\text{“comida fria” NEAR “excelente”}) \text{hits}(\text{“mal/o/a”})}{\text{hits}(\text{“comida fria” NEAR “mal/o/a”}) \text{hits}(\text{“excelente”})} \right] \\ &= \log_2 \left[\frac{4 \cdot 5847}{26 \cdot 14749} \right] \\ &= -4.03 \end{aligned}$$

A partir de las orientaciones semánticas obtenidas con el algoritmo de Turney, clasificaremos la frase “buena comida” como positiva y la frase “comida fria” como negativa.

En la sección de experimentación de esta tesis analizaremos otras adaptaciones que pueden realizarse a este algoritmo para mejorar los resultados para el idioma español.

2.3.10 Clasificación basada en Léxico de Opinión

Existen otros métodos de clasificación, considerados no supervisados o semi-supervisados, que se basan en utilizar un conjunto de términos “semilla”

cuya polaridad es conocida y la clasificación consiste en calcular la similitud o distancia de los términos que componen el documento a estos términos conocidos. En este tipo de métodos el léxico es utilizado como reemplazo de los features del modelo. El ejemplo más extremo de utilización de léxico de opinión es el algoritmo de Turney, que como hemos visto en la sección 2.3.9, utiliza la polaridad conocida de dos únicos términos (*‘excellent’* y *‘poor’*) para clasificar un texto a partir de la distancia de los términos del documento a ellos.

En el trabajo de (Sidorov et al., 2012) se propone un léxico de opinión de más de 2000 palabras en idioma español que incluye la distancia de cada uno de estos términos a alguna de las siguientes emociones básicas: *alegría, ira, miedo, tristeza, sorpresa y disgusto*. En base a esta información podría desarrollarse un algoritmo que infiera la polaridad de una sentencia a partir de calcular la distancia de los términos que la componen a las emociones básicas asociadas a sentimientos positivos o negativos.

En este tipo de clasificadores puede ser necesario asociar la polaridad de los términos del léxico a dominios u objetivos específicos. Por ejemplo, la palabra “barato” puede asociarse a sentimientos positivos cuando se refiere a comida y a sentimientos negativos cuando se refiere a vestimenta o decoración de ambientes; la palabra “añejo” tiene una connotación positiva cuando se refiere a vinos y altamente negativa si se refiere a comida; el término “frío” se asocia a emociones negativas si se habla de pizza y positivas si se habla de cervezas; etc.

Otro método semi-supervisado basado en léxico de opinión consiste en identificar la polaridad de los adjetivos de un texto a partir de la ocurrencia conjunta con términos cuya polaridad es conocida y teniendo en cuenta la conjunción que los relaciona, es decir, considerando que si los adjetivos están conectados por el término “y” tendrán la misma polaridad mientras que si aparecen conectados por el término “pero” tendrán polaridad opuesta. Un ejemplo de lo anterior es el método propuesto en el trabajo de (Hatzivassiloglou and McKeown, 1997) que consta de los siguientes pasos para la construcción de un léxico de polaridad:

1. Etiquetar un conjunto inicial de adjetivos semilla según su polaridad
2. Extender el conjunto anterior a partir de la ocurrencia conjunta de los adjetivos conocidos con otros adjetivos y teniendo en cuenta la conjunción que los relaciona para determinar si tienen la misma polaridad u opuesta.
3. Utilizar un clasificador supervisado para determinar el grado de similitud de polaridad de cada par de adjetivos generando como resultado un gráfico que relacione los adjetivos del conjunto de datos.
4. Clusterizar el gráfico anterior en dos grandes grupos de polaridad similar (positivos y negativos) como se observa en la figura 2.6.

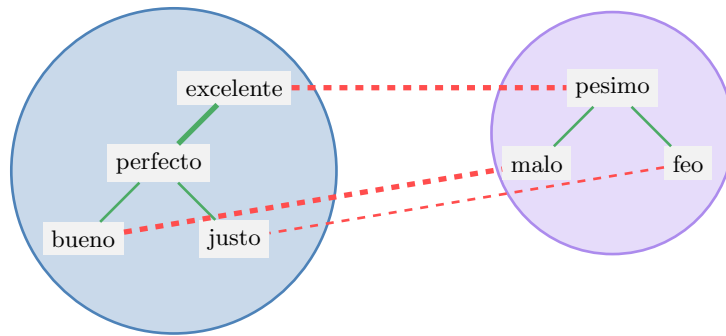


Figura 2.6: Predicción de la Orientación Semántica de Adjetivos: Clustering

2.3.11 Clasificación basada en Puntaje

En las secciones anteriores analizamos cómo clasificar un documento en forma binaria, es decir, asignando las clases “positivo” o “negativo”. Como ya hemos mencionado, existe otro enfoque para clasificación subjetiva de textos y consiste en asignar a cada documento un puntaje dentro de una escala dependiendo del grado de polaridad que expresa (por ejemplo, asignar estrellas del 1 al 5).

La solución más simple para esta tarea es clasificar el texto utilizando múltiples categorías como clases. Esta clasificación puede realizarse con los algoritmos de Naïve Bayes o MaxEnt vistos hasta ahora. En el caso de SVM, el algoritmo presentado en secciones anteriores se utiliza para clasificación binaria, para el caso de múltiples categorías existe el algoritmo llamado *One-vs-all* (Rifkin and Klautau, 2004) que se basa en crear un clasificador binario SVM para cada clase cuya salida nos permita saber si el documento pertenece o no a la clase en cuestión.

En el trabajo de (Bo Pang and Lillian Lee, 2005) se propone otra solución para resolver esta tarea considerándola un problema de *metric labeling*. En este tipo de problema la entrada es un conjunto de etiquetas (clases) y un grafo ponderado. Se define una función de distancia de las etiquetas y para cada etiqueta el costo de asignación a cada vértice, siendo el objetivo encontrar el menor costo de asignar una etiqueta a un vértice. Este enfoque combina los clasificadores SVM y *k-nearest neighbors* que es un método no paramétrico de clasificación basado en encontrar los elementos más cercanos del conjunto de datos de entrenamiento en el espacio de features.

Otro enfoque para la tarea de clasificación en múltiples categorías consiste en utilizar un método de *regresión* como los vistos en secciones anteriores y considerando que las categorías corresponden a la discretización de una función continua.

2.3.12 Clasificación de Múltiples Aspectos

Hasta ahora hemos presentado las tareas de análisis de sentimientos a nivel de documento, es decir, asignando una clase a un documento completo a partir de las emociones que expresa. Esta clasificación puede ser binaria (polaridad

positiva o negativa) o basada en múltiples categorías (utilizando puntajes o estrellas como clases) pero en ambos casos se clasifica el documento completo. En esta sección presentaremos la clasificación aislada de los múltiples objetivos, aspectos o *targets* que generan opinión dentro de un mismo documento. Esta tarea es conocida como *Multi-Aspect Sentiment Analysis*.

El objetivo de esta tarea consiste en la identificación de aspectos que generan opinión dentro de un mismo documento seleccionando aquellas características que son frecuentemente mencionadas y las emociones asociadas. Para identificar estas características o aspectos la solución más simple consiste en utilizar un parser²¹ que nos permita obtener las frases nominales del documento, teniendo en cuenta que la mayoría de los aspectos son sustantivos.

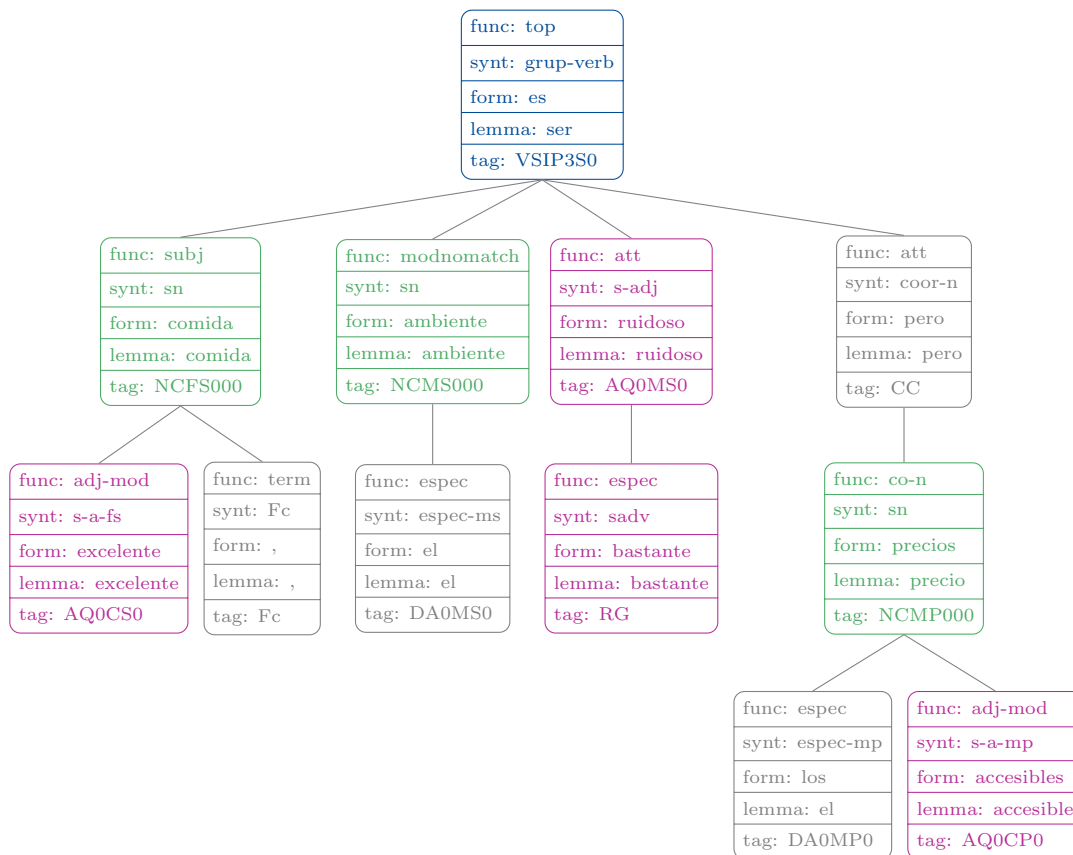
Por ejemplo, analicemos el siguiente documento:

“excelente comida, el ambiente es bastante ruidoso pero los precios accesibles”

El resultado de parsear²² esta sentencia para obtener su estructura sintáctica será el que se indica en la figura 2.7.

²¹Llamamos *parser* a una herramienta que utilizando técnicas de procesamiento de lenguaje natural nos permite obtener la estructura sintáctica de una sentencia.

²²La estructura sintáctica de esta sentencia se obtuvo con la herramienta Freeling3: <http://nlp.lsi.upc.edu/freeling/demo/demo.php>

**Figura 2.7:** Ejemplo Parsing

A partir de la clasificación anterior podremos extraer los sustantivos o frases nominales que nos permitirán identificar términos candidatos a aspectos y los adjetivos para identificar las emociones asociadas. Para obtener la polaridad de sentimientos utilizaremos los métodos de clasificación estudiados en secciones anteriores.

Luego, a partir de la polaridad de la emoción que genera cada aspecto debemos establecer un criterio que nos permita asignar pesos para obtener la polaridad general del documento. La forma más simple será basándonos en la frecuencia de aparición de cada aspecto en el corpus de datos.

Otro método simple para la identificación de *targets* que generan opinión consiste en definir aspectos ad-hoc para un determinado dominio, por ejemplo: {COMIDA, SERVICIO, AMBIENTE, OTROS} y luego ejecutar la clasificación de todos los términos del documento utilizando un método supervisado y un conjunto de datos de entrenamiento clasificado manualmente que nos permita obtener aquellos términos que pertenecen a alguna de las clases de aspectos relevantes generadores de opinión.

Identificación de Aspectos basada en Modelos de Tópicos

Existen otros métodos más generales de identificación de aspectos que se basan en modelos de reconocimiento de tópicos. El método más utilizado para identificación de tópicos se conoce como *Latent Dirichlet Allocation (LDA)* y es un modelo probabilístico generativo para colecciones de datos discretos, como corpus de textos, en el que cada ítem de la colección es modelado como una mezcla finita de un conjunto de tópicos subyacentes y cada tópico es a su vez modelado como una mezcla infinita de probabilidades de tópicos. En modelado de textos, las probabilidades de un tópico representan un documento (Blei et al., 2003).

El método anterior, sin embargo, tiende a encontrar tópicos globales en los datos por lo que resulta inadecuado para identificación de aspectos. Para resolver este problema surge el método *Multi-Grain Topic Model (MG-LDA)* que tiene como objetivo encontrar tópicos globales y locales, siendo estos últimos los que representan aspectos clasificables (Titov and McDonald, 2008).

En el trabajo de (Lu et al., 2011) se presenta un método débilmente supervisado basado en LDA para identificación de aspectos que consiste en utilizar palabras “semilla” y tópicos fijos para el dominio en cuestión. Éste enfoque guía el aprendizaje hacia tópicos específicos de los aspectos seleccionados por lo que se obtienen mejores resultados.

2.3.13 Combinación de Clasificadores

Otra técnica para mejorar la predicción consiste en utilizar *multi-clasificadores*, es decir, combinar distintos clasificadores y seleccionar la clase a partir de la votación de los modelos seleccionados. Esta votación puede realizarse en forma democrática (también conocida como *votación ingenua*); en forma calificada ponderando el voto de cada clasificador con un determinado peso en función de su eficiencia; o bien utilizando SVM como un clasificador múltiple que combine distintos modelos simples utilizando el resultado de cada modelo como feature en el entrenamiento del multclasificador como se indica en la figura 2.8 (Tsutsumi et al., 2007).

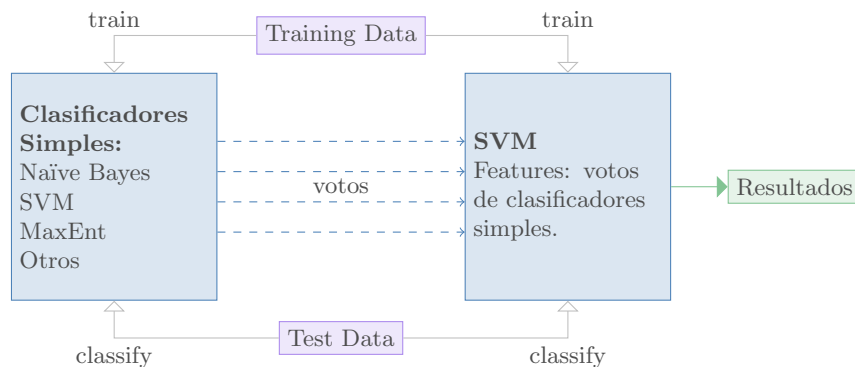


Figura 2.8: Combinación de Clasificadores utilizando SVM

2.3.14 Evaluación de Clasificadores

En esta sección analizaremos las distintas métricas que se utilizan para evaluar los resultados del clasificador y obtener una medida de su eficiencia. Además, presentaremos las distintas técnicas que se utilizan en métodos supervisados para evaluar los clasificadores de manera confiable en función de los datos que se utilizan para el entrenamiento y las métricas presentadas.

Métricas

Dependiendo de la tarea de procesamiento de lenguaje que se esté desarrollando podrá variar el significado o el sentido que se le da a cada medición pero siempre se considerarán los siguientes casos:

- Verdaderos Positivos, llamados *True Positives* (t_p): elementos a los que el clasificador asignó la clase relevante y ésta era correcta.
- Falsos Positivos, llamados *False Positives* (f_p): elementos a los que el clasificador asignó la clase relevante y ésta no era correcta.
- Falsos Negativos, llamados *False Negatives* (f_n): elementos a los que el clasificador asignó la clase no-relevante y ésta no era correcta.
- Verdaderos Negativos, llamados *True Negatives* (t_n): elementos a los que el clasificador asignó la clase no-relevante y ésta era correcta.

Por ejemplo, en tareas de recuperación de textos, las medidas anteriores tendrán el siguiente significado: documentos que fueron seleccionados por el clasificador y eran relevantes a la búsqueda (t_p); documentos que fueron seleccionados y no eran relevantes a la búsqueda (f_p); documentos que no fueron seleccionados y sí eran relevantes a la búsqueda (f_n); y documentos que no fueron seleccionados y no eran relevantes a la búsqueda (t_n).

Esto es,

	Relevantes	No Relevantes
Seleccionados	t_p	f_p
No Seleccionados	f_n	t_n

Tabla 2.7: Evaluación de Clasificadores - Recuperación de Textos

En tareas de AS las clases involucradas serán: “Documentos Positivos” y “Documentos Negativos” por lo que el significado que le daremos a la medición de resultados será: Documentos clasificados como positivos que eran efectivamente positivos (t_p); documentos clasificados como positivos que eran negativos (f_p); documentos clasificados como negativos que eran positivos (f_n); y documentos clasificados como negativos que eran efectivamente negativos (t_n).

Esto es,

	Positivo	Negativo
Clasificados como Positivos	t_p	f_p
Clasificados como Negativos	f_n	t_n

Tabla 2.8: Evaluación de Clasificadores - Análisis de Sentimientos

No se debe confundir el hecho de que los nombres de nuestras clases sean “Positivo” y “Negativo” con aquellos casos que resultan positivos (clase relevante) o negativos (clase no-relevante) para el clasificador.

A continuación veremos algunas métricas que resultan de utilidad para evaluar clasificadores:

Accuracy representa la porción de documentos que son clasificados correctamente sobre el total de casos.

$$Accuracy_{TOTAL} = \frac{t_p + t_n}{t_p + t_n + f_p + f_n} \quad (2.39)$$

Precision representa la porción de documentos que son clasificados correctamente para la clase A sobre el total de casos clasificados como clase A.

$$Precision_A = \frac{t_p}{t_p + f_p} \quad (2.40)$$

Recall representa la porción de documentos de clase A que son clasificados correctamente.

$$Recall_A = \frac{t_p}{t_p + f_n} \quad (2.41)$$

Ejemplo 2.8. *Evaluación de Clasificadores*

Consideremos el siguiente conjunto de datos:

Documento	Clase	Predicción del Clasificador
D1	POS	POS
D2	POS	POS
D3	POS	POS
D4	POS	POS
D5	POS	POS
D6	NEG	POS
D7	NEG	NEG
D8	POS	POS
D9	POS	NEG
D10	POS	POS
D11	NEG	POS
D12	NEG	POS
D13	NEG	POS
D14	NEG	POS
D15	NEG	NEG
D16	POS	POS
D17	POS	POS
D18	POS	POS
D19	POS	POS
D20	POS	POS

Tabla 2.9: Evaluación de Clasificadores - Corpus de Ejemplo

	Positivo	Negativo
Clasificados como Positivos	$t_p = 12$	$f_p = 5$
Clasificados como Negativos	$f_n = 1$	$t_n = 2$

Tabla 2.10: Evaluación de Clasificadores - Ejemplo de Métricas

Para los datos anteriores obtendremos las siguientes métricas:

$$Accuracy_{TOTAL} = 14/20 = 0.7$$

$$Precision_{POS} = 12/17 = 0.706$$

$$Recall_{POS} = 12/13 = 0.923$$

$$Precision_{NEG} = 2/3 = 0.66$$

$$Recall_{NEG} = 2/7 = 0.286$$

Esto significa que:

- El clasificador acierta el 70% de las veces ($Accuracy_{TOTAL}$)
- Los comentarios positivos se predicen con una probabilidad de acierto del 70% ($Precision_{POS}$)

- Del total de comentarios positivos, el 92% será clasificado como tal ($Recall_{POS}$)
- Los comentarios negativos se predicen con una probabilidad de acierto del 66% ($Precision_{NEG}$)
- Del total de comentarios negativos, sólo el 28% será clasificado como tal ($Recall_{NEG}$)

Si sólo miráramos los valores de accuracy o precisión del ejemplo anterior podríamos incurrir en el error de decir que el clasificador obtiene buenos resultados (basados en el 70% de acierto). Sin embargo, si miramos la recall de los comentarios negativos vemos que sólo el 28% de los comentarios de esta clase se clasifican como tal, lo cual podría significar (dependiendo el caso) un resultado inaceptable.

Para encontrar una métrica que represente la efectividad de un clasificador, teniendo en cuenta tanto la precision como la recall, de modo que nos permita tener una idea más general de cuan buenos son los resultados, utilizaremos lo que se conoce como *F-Measure*.

F-Measure combina las medidas de precisión y recall a partir de la media armónica ponderada²³ de estos dos valores.

$$F = \frac{1}{\alpha \frac{1}{P} + (1 - \alpha) \frac{1}{R}} = \frac{(\beta^2 + 1)PR}{\beta^2 P + R} \quad (2.42)$$

Por lo general se utiliza una forma balanceada de la métrica anterior, es decir, para $\beta = 1$ conocida como **F1-Measure**.

$$F_1 = \frac{2PR}{P + R} \quad (2.43)$$

En el ejemplo anterior,

$$F_{1POS} = \frac{2 \cdot 0.706 \cdot 0.923}{0.706 + 0.923} \approx 0.8$$

$$F_{1NEG} = \frac{2 \cdot 0.66 \cdot 0.286}{0.66 + 0.286} \approx 0.4$$

Accuracy vs. F-Measure

En el siguiente ejemplo analizaremos cuándo conviene usar cada una de estas métricas:

Ejemplo 2.9. Accuracy vs. F-Measure

Consideremos la tarea de clasificar textos como “spam” o “no-spam” y un conjunto de datos donde el 99% de los textos no son spam. Supongamos además, que se utiliza un clasificador que siempre clasifica como “no-spam” cualquiera sea el dato de entrada. En este caso la accuracy será del 99% por lo que podríamos incurrir en el error de considerar que el clasificador tiene un 99% de efectividad y quedarnos únicamente con esa información cuando es claro que

²³La media armónica es un tipo de promedio muy conservativo.

ese clasificador no es adecuado.

En casos como el anterior, donde el corpus está fuertemente desbalanceado, resulta de mucha utilidad considerar tanto los valores de precisión como de recall utilizando la métrica F_1 para medir la efectividad de un clasificador.

En casos donde el corpus esté balanceado (cada clase es equiprobable), la accuracy será una medida suficiente de la efectividad del clasificador dado que podemos estar seguros de no estar ignorando el comportamiento del clasificador para alguna de las clases.

Macro vs. Micro Averaging

Cuando se tiene más de una clase de relevante existen distintos enfoques para obtener la efectividad promedio de un clasificador:

Macroaveraging: Se calcula la performance de cada clase por separado y luego se obtiene el promedio.

Microaveraging: Se obtienen las mediciones para cada clase y se calcula la efectividad del clasificador en base a la suma de los valores obtenidos para cada clase.

Métodos de Evaluación de Clasificadores Supervisados

En los métodos supervisados la clasificación depende del entrenamiento que se realice sobre el clasificador por lo que su efectividad dependerá en gran medida de qué datos del corpus se utilicen para entrenamiento (training set) y cuáles para evaluarlo (test set). El criterio de selección de estos conjuntos de datos y la cantidad de veces que se evalúe al clasificador utilizando distintos datos definirá la confiabilidad de la evaluación. A continuación presentaremos los métodos más utilizados.

Hold-out Es el método más sencillo de evaluación y consiste en dividir el conjunto de datos en dos partes, conjunto de entrenamiento y conjunto de prueba (generalmente se utilizan dos tercios para entrenamiento y un tercio para prueba). Utilizando este método se obtienen resultados para el conjunto de datos de prueba elegido pero se ignora el comportamiento del clasificador para otros conjuntos de prueba, por lo que podría ocurrir que el clasificador funcione muy bien para los documentos de prueba elegidos pero no para otros casos. Incluso si usamos este mismo conjunto de prueba para implementar el clasificador podríamos correr el riesgo de refinarlo específicamente para los documentos de prueba y que luego no funcione bien para otros documentos desconocidos.

k-fold Cross-Validation Este método es una mejora del método *hold-out*. En este método el conjunto de datos se divide en k partes, cada una denominada **fold** y se repite el método holdout k veces utilizando un fold como conjunto de prueba y el resto ($k - 1$ folds) como conjunto de entrenamiento como se indica en la figura 2.9 para el caso de $k=10$. Luego, las métricas del clasificador se

calcularán como el promedio de las mediciones obtenidas en cada ejecución.

1	TEST	TRAIN		
2	TRAIN	TEST	TRAIN	
3	TRAIN		TEST	TRAIN
4	TRAIN		TEST	TRAIN
5	TRAIN		TEST	TRAIN
6	TRAIN		TEST	TRAIN
7	TRAIN		TEST	TRAIN
8	TRAIN		TEST	TRAIN
9	TRAIN		TEST	TRAIN
10	TRAIN			TEST

Figura 2.9: 10-Fold Cross-Validation

El valor de k más utilizado suele ser 10, pero puede seleccionarse $k=3,5,10$, etc.

La desventaja de este método es que ejecutar la clasificación k veces puede resultar muy costoso computacionalmente.

Leave-One-Out Cross-Validation Este método es una variante de k -cross validation llevada al extremo, donde $k = N$ siendo N el número total de documentos en el conjunto de datos. Esto significa que la clasificación se ejecuta N veces utilizando como entrenamiento todo el conjunto de datos excepto el elemento que se está evaluando por lo que en cada ejecución se predice la clase de un único documento.

.632 Bootstrap Este método se basa en el muestreo con repetición, es decir, en lugar de analizar repetidamente subconjuntos de datos como en el método de *cross-validation*, la repetición se encuentra en las muestras. Consideremos un conjunto de datos de n muestras uniformemente distribuidas, donde la probabilidad de una muestra de ser elegida es $1/n$ y la probabilidad de no ser elegida es $(1 - 1/n)$. Como la selección debe realizarse n veces, la probabilidad de que una muestra no sea elegida esas n veces será $(1 - 1/n)^n \approx e^{-1} \approx 0.368$. A partir de la ecuación anterior se define que el 36.8% de las muestras originales formarán parte del conjunto de prueba mientras que el 63.2% restante formará parte del conjunto de entrenamiento. El resto de las muestras del conjunto de datos de entrenamiento estará conformado por repeticiones de las muestras originales.

2.4 Preparación de los datos

2.4.1 Preprocesamiento

Con el objetivo de mejorar los resultados del clasificador y dependiendo de la tarea de procesamiento de texto que se esté realizando y el idioma en el que esté escrito el texto, pueden requerirse algunas transformaciones en el corpus de

datos de entrada antes de su procesamiento.

Estas transformaciones pueden incluir algunas de las técnicas que veremos a continuación.

Normalización

Consiste en unificar términos que representan la misma información y pueden ser escritos en distintos formatos. Por ejemplo, “*restaurante*”, “*restaurant*”, “*restorán*”, “*restó*”.

Tokenización

Separación de sentencias y palabras de un documento a partir de *tokens*, o caracteres especiales, que indican el fin de una sentencia o palabra y el comienzo de la que sigue.

Stemming

Existen algoritmos llamados de Stemming (o Stemmers) que permiten obtener la raíz o *stem* de una palabra eliminando terminaciones, con el objetivo de unificar aquellos términos que aportan la misma información al clasificador. Por ejemplo, los términos “recomendable, recomendamos, recomendar, recomendación” son reemplazados por su stem, “recomend”. El stem de una palabra no necesariamente será un término válido del vocabulario.

Reemplazar los términos del corpus con sus formas “stemizadas” suele ser de utilidad para mejorar los resultados en tareas de recuperación de textos. En nuestro caso de estudio veremos que para tareas de AS éste preprocesamiento no representa una mejora.

Los algoritmos de stemming son fuertemente dependientes del idioma. En inglés, la implementación más utilizada es el stemmer de Porter y en español se utiliza más frecuentemente el Snowball stemmer²⁴.

Lematización

Otra forma de unificar los términos que aportan la misma información al clasificador es reemplazando cada palabra por su *lema*. El lema de una palabra es un término válido del vocabulario (a diferencia de lo que ocurría con el stem) que por convención es la representación de todas las formas flexionadas de la palabra, es decir, para hallar el lema de un término se eliminan todas las flexiones (conjugaciones, grado, persona, género, número, etc).

Por ejemplo,

lema(“pésimo”) = malo
lema(“empieza”) = empezar

²⁴Snowball es un lenguaje para la construcción de Stemmers en distintos idiomas. <http://snowball.tartarus.org/algorithms/spanish/stemmer.html>

lema(“primeras”) = primero
 lema(“estas”) = este

Freeling (Padró and Stanilovsky, 2012) es la herramienta de lematización más utilizada para textos en español.

Tratamiento de Negaciones

Con el objetivo de distinguir aquellos términos que siguen a una negación se debe indicar que la palabra aparece negada.

Por ejemplo, consideremos los siguientes documentos a clasificar analizando sentimientos:

D_1 = “*Este restaurante es muy bueno*”

D_2 = “*Este restaurante es horrible, el servicio no es bueno, la comida muy mala.*”

D_1 es claramente positivo y D_2 negativo, sin embargo en ambos casos aparece la palabra “bueno” como término relevante, aunque en el segundo caso viene después de una negación.

Para distinguir estos dos casos se agrega un prefijo a los términos que siguen a la negación hasta el siguiente signo de puntuación.

Esto es,

D_1 = “*Este restaurante es muy bueno*”

D_2 = “*Este restaurante es horrible, el servicio no NOT_es NOT_bueno, la comida muy mala.*”

Realizando esta distinción evitamos incurrir en el error de considerar el término “bueno” como relevante en la clase de comentarios negativos y agregamos como relevante el término “NOT_bueno” en esta clase con el objetivo de mejorar los resultados del clasificador.

Los términos que se utilizan para indicar negación dependen del idioma del corpus de datos.

Part Of Speech (POS) Tagging

Dentro del área de Procesamiento de Lenguaje Natural se conoce como *Part Of Speech Tagging* (o *POS Tagging*) a la tarea de asignar a cada término del discurso una etiqueta que nos indique si la palabra actúa como sustantivo, verbo, adjetivo, artículo, adverbio, etc, dependiendo del contexto.

En clasificación de textos y con el objetivo de desambiguar el sentido de una palabra²⁵ puede ser de utilidad agregar a cada término un sufijo con la etiqueta asignada realizando POS tagging de modo de diferenciar aquellos términos idénticos que expresan significados distintos dependiendo del contexto de la oración. Por ejemplo, consideremos los siguientes documentos:

²⁵Tarea conocida en inglés como *Word Sense Desambiguation (WSD)*.

D_1 = “En este restaurante el vino es muy malo.”

D_2 = “Todo salió perfecto, vino el chef y lo felicitamos por sus excelentes platos.”

En este caso puede interesarnos diferenciar el término “vino” cuando éste actúa como verbo, donde no aporta información relevante, de su presencia actuando como sustantivo donde representa un target de opinión.

Luego de realizar POS tagging sobre los documentos mencionados, el término “vino” será reemplazado por *vino_SUST* y *vino_VERB* según corresponda.

Más adelante analizaremos cómo esta técnica puede ser de gran utilidad en sistemas de AS para identificar y extraer los adjetivos de un documento, donde se encuentra la información más relevante en términos de opinión.

Comienzo y finalización de Sentencias

En modelos de bigrama (ver sección 2.2.2), para evitar que la probabilidad de la palabra que inicia la sentencia dependa del final de la sentencia anterior se puede agregar la etiqueta $< S >$ al comienzo de cada oración. Siguiendo el mismo criterio para la última palabra de la sentencia y la palabra que comienza la sentencia que sigue, se puede agregar la etiqueta $< /S >$ al final de cada oración.

Otros Preprocesamientos

Otras transformaciones que pueden representar una mejora en la efectividad del clasificador incluyen eliminar términos o secuencias de palabras que no aporten información a la tarea de procesamiento que se está realizando. Por ejemplo, signos de puntuación, fechas, números, capitalizaciones²⁶, caracteres especiales, emoticones²⁷, caracteres repetidos más de dos veces, stopwords²⁸, palabras de menos de N letras, acentos, etc.

También puede ser de utilidad corregir errores ortográficos para mejorar los resultados del clasificador.

En el capítulo 4.2.2 y en base a nuestro caso de estudio, analizaremos qué transformaciones conviene realizar en tareas de AS sobre textos en español.

²⁶En algunas tareas de clasificación mantener las capitalizaciones puede resultar útil para identificar nombres propios.

²⁷En tareas de AS los emoticones pueden aportar información y tenerlos en cuenta puede mejorar los resultados del clasificador.

²⁸Se consideran stopwords aquellas palabras que no agregan información al texto que se está procesando. Por ejemplo, en español: de, la, el, que, a, etc.

Capítulo 3

Desarrollo de la Herramienta de Análisis de Sentimientos

En este capítulo presentamos la aplicación de todos los conceptos estudiados en este trabajo para el desarrollo de una herramienta de análisis binario de sentimientos que luego, en la etapa de experimentación, nos permitirá validar la investigación y comparar la efectividad de las distintas técnicas de clasificación y preprocesamiento de textos estudiadas.

En la primera sección de este capítulo definiremos el alcance de la aplicación desarrollada, analizaremos el comportamiento de la herramienta y los distintos procesos y flujos que ocurren en la clasificación. En la sección de implementación, describiremos cómo se construyó la aplicación, especificaremos el diseño, los criterios adoptados, las interfaces desarrolladas y los frameworks y herramientas externas que se utilizaron en la implementación.

3.1 Análisis del Problema

Como hemos estudiado en la etapa de investigación, una herramienta de análisis binario de sentimientos debe ser capaz de analizar textos en base a información subjetiva clasificándolos según la polaridad de sentimientos que expresan sobre productos, situaciones, organizaciones o personas. El sistema deberá procesar un corpus de textos, realizar las transformaciones necesarias e informar la clase a la que pertenece cada documento (positivo o negativo). Como hemos visto, existen otras tareas que puede realizar un sistema de análisis de sentimientos, como informar el grado de polaridad de las emociones o incluso identificar los distintos aspectos que se mencionan en un documento y clasificar la emoción asociada a cada aspecto particular. En esta implementación nos enfocaremos en la clasificación binaria y el proceso principal será como se indica en la figura 3.1.

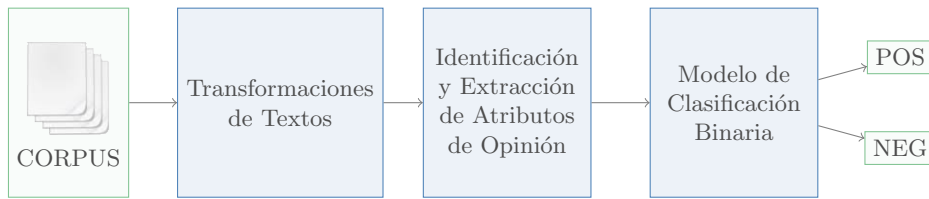


Figura 3.1: Proceso de Análisis de Sentimientos

El proceso anterior se ejecuta utilizando distintos modelos de clasificación y en función de una variedad de parámetros de entrada que nos permitirán realizar una comparación experimental de la efectividad de cada método. Para realizar la evaluación, además de predecir la clase, la aplicación desarrollada informa las métricas obtenidas sobre los resultados que incluyen: accuracy, precision, recall y F1 para ambas clases relevantes del modelo.

3.1.1 Proceso de Clasificación Supervisado

En el caso de los modelos supervisados el proceso de clasificación se ejecuta como se indica en la figura 3.2 y fue implementado en base a lo explicado en el estado del arte de este trabajo, es decir, entrenando los distintos modelos en base a información conocida sobre documentos etiquetados correctamente y a partir de la extracción de atributos del texto.

El flujo de procesamiento del corpus de datos se implementó como se indica a continuación:

1. Se toma como entrada un corpus etiquetado y se aplican las transformaciones correspondientes.
2. Se divide el corpus en un conjunto de datos de entrenamiento y un conjunto de datos de prueba y se extraen los features relevantes.
3. Se entrena el modelo a partir de los features y las clases conocidas del conjunto de datos de entrenamiento.
4. Se ejecuta la clasificación de los documentos del conjunto de prueba.
5. Se evalúan los resultados obtenidos comparándolos con las clases esperadas del conjunto de datos de prueba.
6. Se informan las métricas de la clasificación.

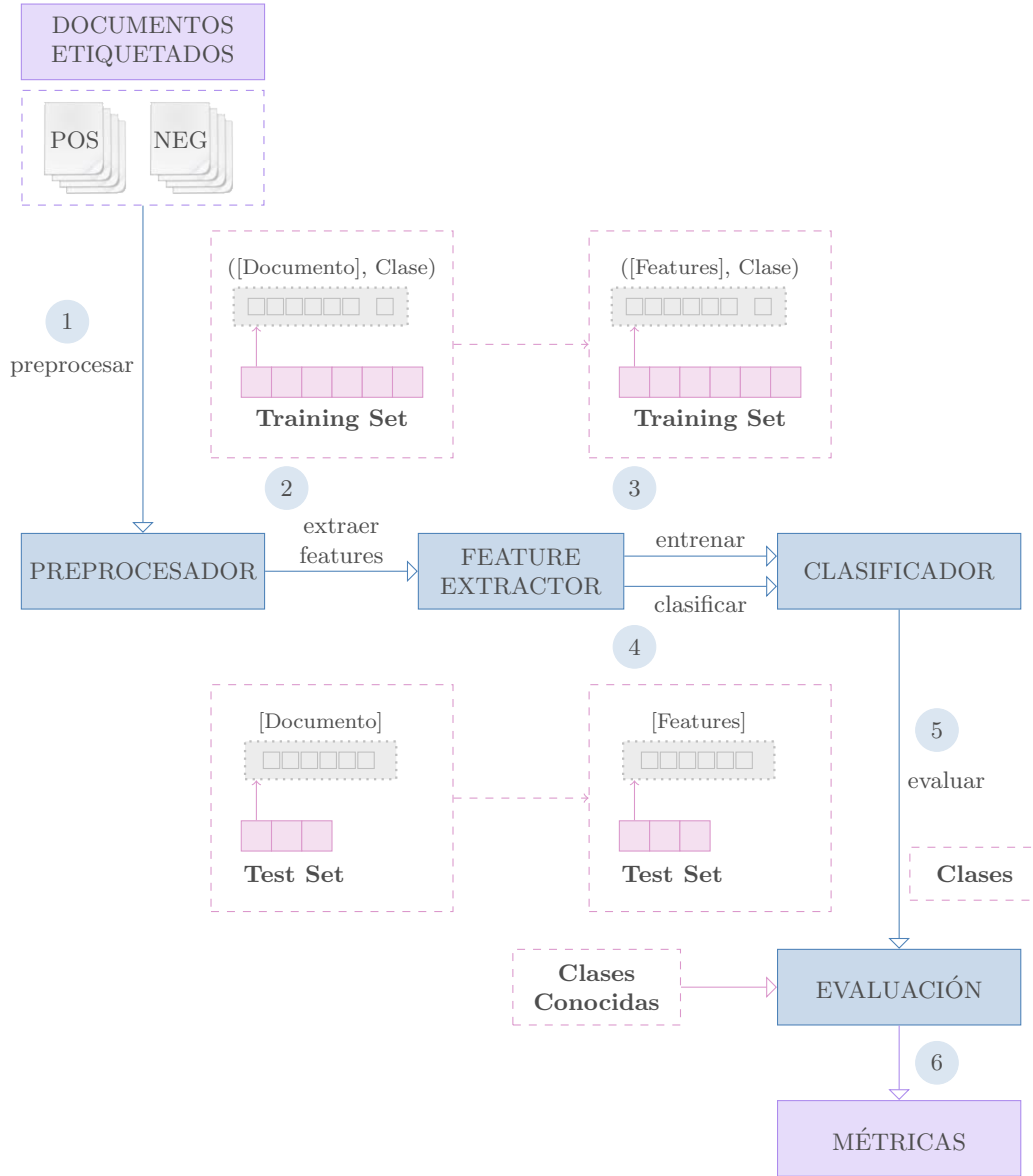


Figura 3.2: Proceso de Clasificación Supervisado

3.1.2 Proceso de Clasificación No Supervisado

El proceso no supervisado se ejecuta como se indica en la figura 3.3 y como hemos visto en el estado del arte de este trabajo, la clasificación se ejecuta en base al cálculo de orientación semántica de los términos de cada documento. Este método se considera no supervisado teniendo en cuenta que para predecir la polaridad de un texto no se utiliza información conocida sobre documentos ya clasificados del conjunto de datos. Sin embargo, hay quienes consideran este método como semi-supervisado por utilizar información conocida sobre la polaridad de los términos que componen el léxico.

El proceso de clasificación se ejecuta según los siguientes pasos:

1. Se toma como entrada un corpus no etiquetado y se aplican las transformaciones correspondientes.
2. Se seleccionan los elementos del conjunto de datos que serán clasificados.
3. Se clasifican los elementos del conjunto de prueba a partir del cálculo de orientación semántica.
4. Se evalúan los resultados obtenidos comparándolos con las clases esperadas del conjunto de datos de prueba.
5. Se informan las métricas de la clasificación.

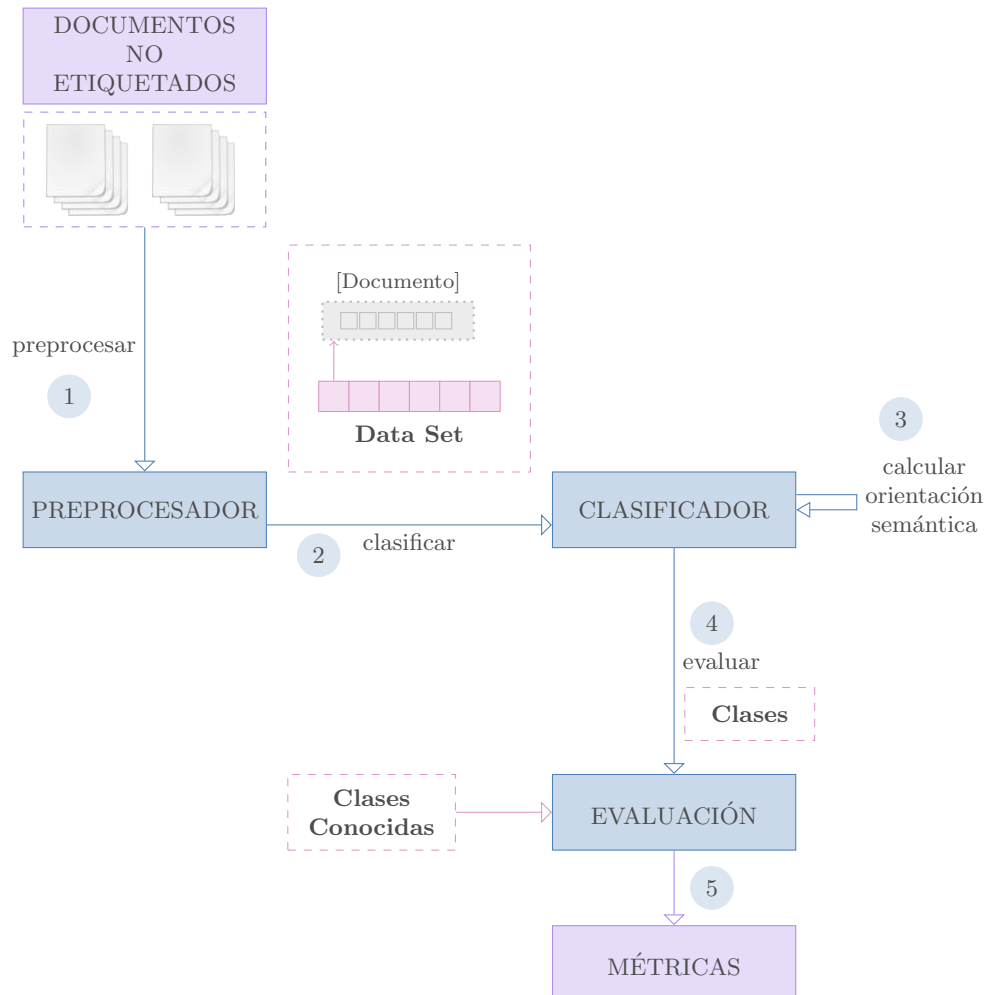


Figura 3.3: Proceso de Clasificación No Supervisado o Semi-Supervisado

3.1.3 Parámetros de Clasificación

A continuación describiremos los distintos parámetros con los que puede ejecutarse la clasificación para analizar la performance y sensibilidad de los modelos de clasificación ante distintos escenarios.

La herramienta provee los siguientes preprocesadores de textos y puede ejecutarse con cualquier combinación de ellos que se especifique por parámetro:

- Eliminación de stopwords
- Filtrado de palabras con un mínimo de longitud parametrizable
- Eliminación de caracteres repetidos más de dos veces
- Procesamiento de negaciones
- Stemming
- Lematización
- Transformación a minúscula
- Eliminación de signos de puntuación
- Eliminación de caracteres especiales

Los modelos binarios de clasificación provistos por la herramienta desarrollada incluyen:

- Clasificadores Supervisados
 - Naïve Bayes
 - Modelo de Máxima Entropía
 - Support Vector Machines (SVM)
 - Decision Trees
- Clasificadores No Supervisados
 - Algoritmo de Turney adaptado al idioma español

En cuanto a la extracción de características, los siguientes *feature extractors* fueron implementados y la aplicación puede ejecutarse con cualquier combinación de ellos:

- Presencia de unigramas
- Frecuencia de unigramas
- Presencia de bigramas
- Presencia de adjetivos

El método de evaluación del modelo también es configurable y se implementaron los siguientes algoritmos en función de lo explicado en la sección 2.3.14:

- k-fold cross validation: La cantidad de folds y el tamaño de cada fold es configurable y se utilizó *macro-averaging* para el cálculo de métricas generales a partir de los resultados obtenidos para cada fold.
- hold-out

Además de los parámetros anteriores, la aplicación provee la posibilidad de ejecutar la clasificación en los siguientes escenarios:

- Out-of-domain testing: Para clasificación supervisada, en la etapa de construcción de los conjuntos de datos de entrenamiento y prueba se utiliza un corpus de datos para entrenamiento y otro, perteneciente a un dominio distinto, para test. Como veremos en el capítulo de experimentación, este tipo de pruebas es fundamental para evaluar la performance de un modelo.
- Desbalanceo de corpus: La aplicación puede ser ejecutada con un parámetro que indique la proporción de documentos pertenecientes a cada clase (positivos o negativos) que serán utilizados en la clasificación con el objetivo de analizar la performance de los modelos para corpus de datos fuertemente desbalanceados.

3.1.4 Evaluación de Otros Modelos

La herramienta desarrollada fue implementada ad-hoc para este trabajo en el que se desea evaluar y comparar experimentalmente los distintos modelos de clasificación de textos mencionados. Sin embargo, el software desarrollado fue implementado en forma genérica y puede utilizarse como base para evaluar cualquier tipo de algoritmo de clasificación subjetiva que se implemente dado que permite medir sensibilidad en la efectividad de los modelos en base al tamaño de corpus, tipo de atributos extraídos, proporción de comentarios positivos y negativos para el entrenamiento, comportamiento ante el cambio de dominio y preprocesamientos aplicados. Además, la herramienta provee distintas técnicas de validación e informa, como resultado de la ejecución, todas las métricas necesarias para la evaluación de un modelo.

3.2 Implementación

En esta sección explicaremos los detalles técnicos de implementación del sistema de análisis de sentimientos así como las decisiones de diseño adoptadas, algoritmos e interfaces desarrolladas.

La solución fue desarrollada en lenguaje Python y los principales componentes del proceso de análisis de sentimientos se modelaron como se indica en los diagramas de clases de las figuras 3.4, 3.6 y 3.5 donde se indican las principales jerarquías de clasificadores, preprocesadores, extractores de features e interfaces con frameworks y herramientas externas de cada componente.

Como hemos mencionado en secciones anteriores, uno de los objetivos principales de esta aplicación es proveer un marco para la evaluación de la performance de modelos de clasificación subjetiva de textos en función de distintos escenarios, por lo que el diseño de la arquitectura se basó en la extensibilidad y simplicidad de la herramienta para la implementación de nuevos modelos de clasificación, extracción de features, preprocesamiento de textos y técnicas de validación y así abstraerse de la implementación del proceso de clasificación y cálculos de performance que son provistos por el marco de trabajo.

Como se observa en la figura 3.4 si se desea implementar un nuevo modelo de clasificación bastará con heredar la clase *SupervisedClassifier* y redefinir los métodos *train()* y *classify(comment)* para modelos supervisados o heredar de la clase *Classifier* y redefinir el método *classify(corpus)* para modelos no-supervisados. Luego, será necesario agregar un parámetro de ejecución de la herramienta para el nuevo modelo y esto será suficiente para ejecutar la clasificación y obtener todas las métricas necesarias para la evaluación de la performance del nuevo clasificador en todos los escenarios en estudio.

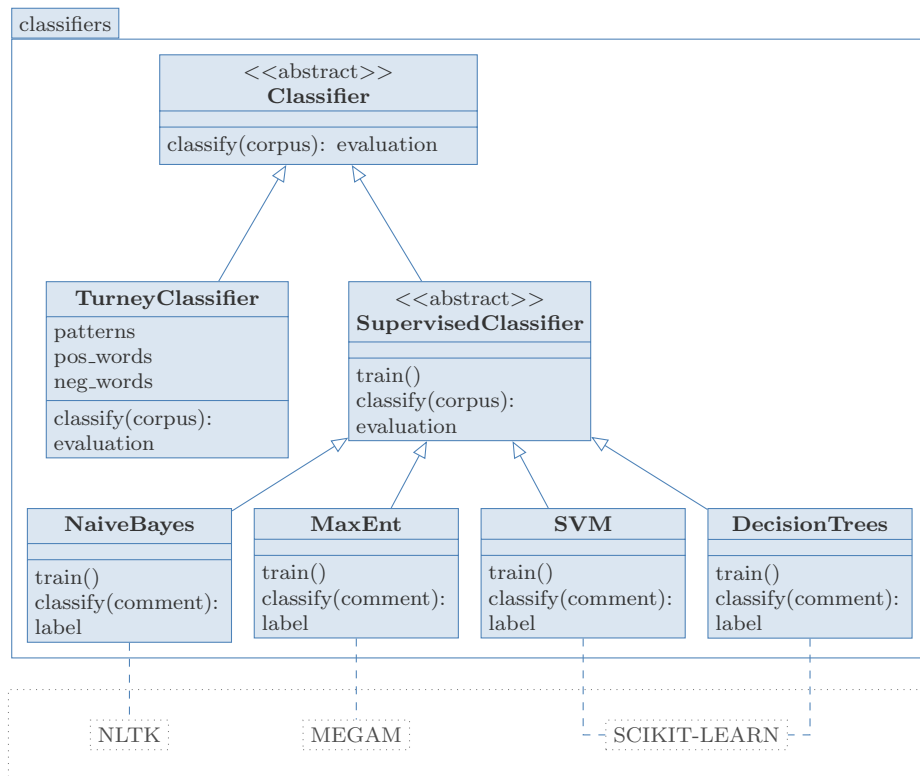


Figura 3.4: Jerarquía de Clasificadores

Del mismo modo que para la jerarquía de clasificadores, también se observa en las figuras 3.5 y 3.6 como la implementación de un nuevo extractor de

features o preprocesador de textos consiste en incluir el nuevo modelo en la jerarquía, redefinir los métodos *extract()* y *process()* respectivamente y agregar el parámetro de ejecución a la herramienta.

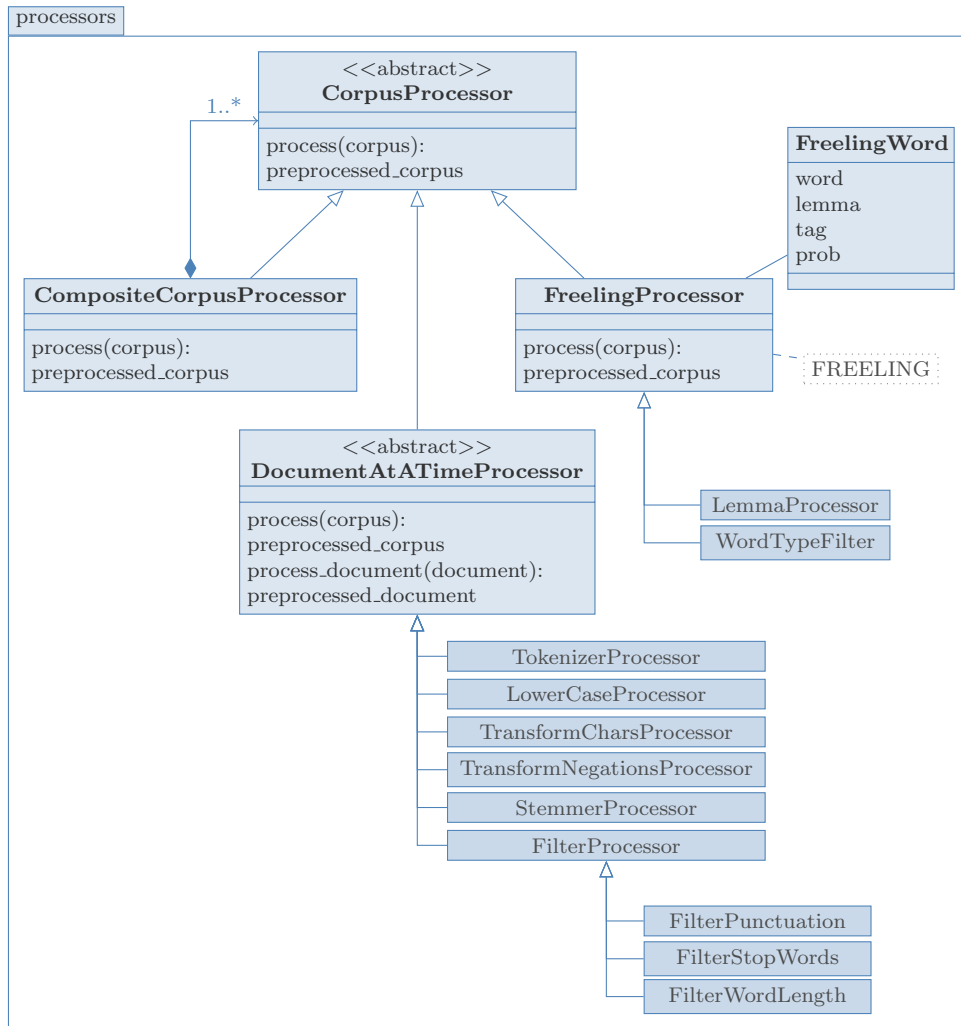


Figura 3.5: Jerarquía de Preprocesadores

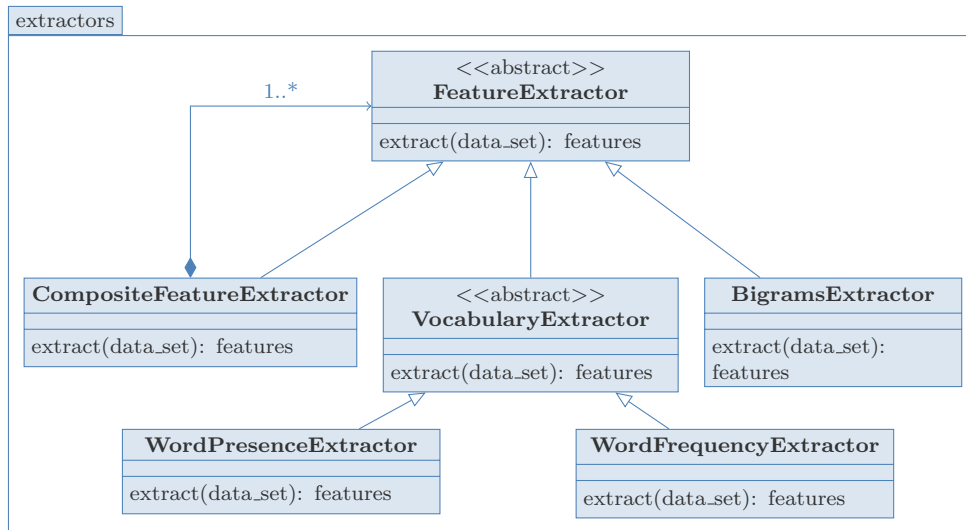


Figura 3.6: Jerarquía de Feature Extractors

La extracción de adjetivos como features se implementó aplicando un preproceso de filtro por tipo de palabra a partir de realizar POS tagging sobre el corpus de datos y luego se utilizó presencia de unigramas como feature extractor.

3.2.1 Proceso de Clasificación

El proceso de clasificación se implementó como se indica en los diagramas de secuencia de las figuras 3.7 y 3.8. En estos diagramas se observan las interacciones de los distintos componentes de la aplicación y las transformaciones y tareas de procesamiento que se realizan sobre los documentos del corpus de datos.

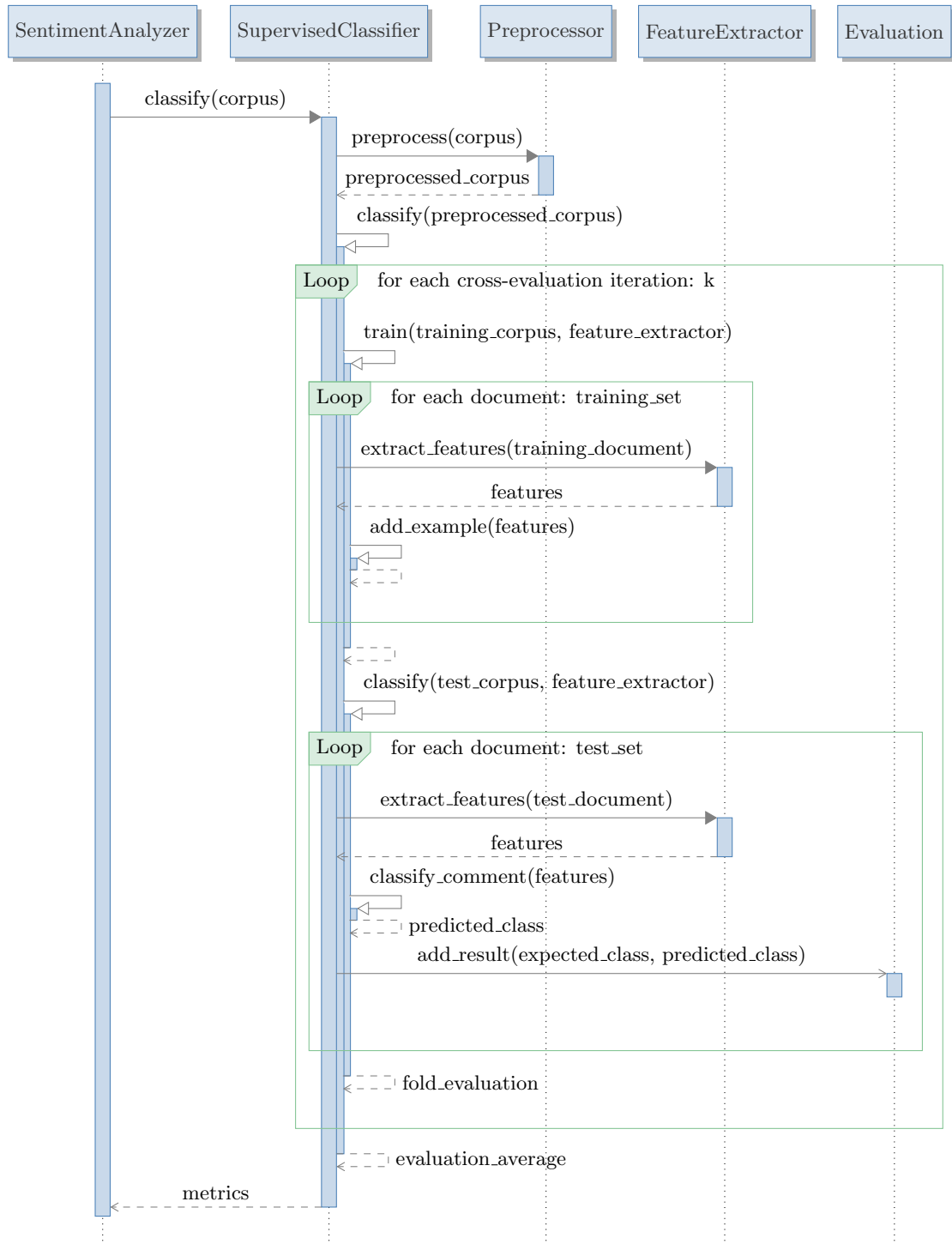


Figura 3.7: Clasificación Supervisada - Diagrama de Secuencia

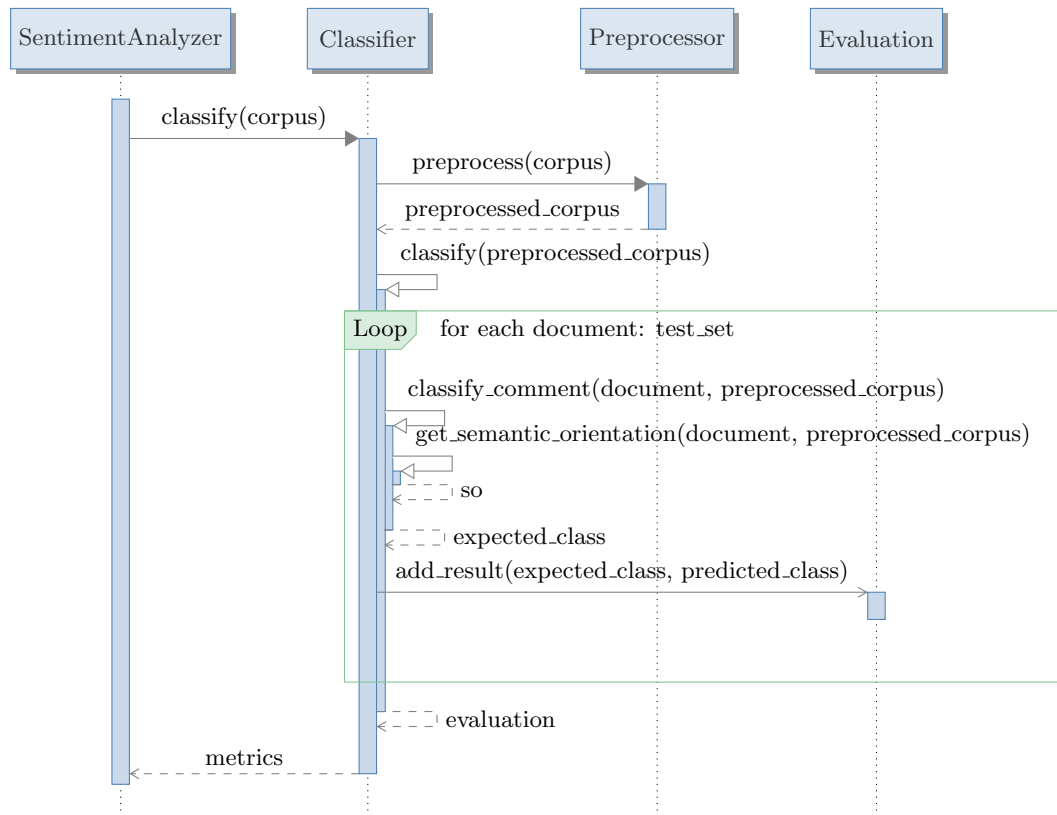


Figura 3.8: Clasificación No Supervisada - Diagrama de Secuencia

3.2.2 Frameworks y Herramientas Externas

A continuación describiremos las principales herramientas y frameworks utilizados en el desarrollo del software de clasificación subjetiva de textos para resolver distintas funcionalidades de la aplicación. Basamos la elección de estas herramientas en la precisión de los resultados, la simplicidad de implementación, la buena documentación y los tiempos de procesamiento.

NLTK (Steven Bird, Edward Loper and Ewan Klein, 2009) es la plataforma más utilizada para trabajar con lenguaje natural en Python por su simplicidad y excelente documentación. En este trabajo se utilizó la versión 2.0 para la implementación del algoritmo de *Naïve Bayes*, tareas de *tokenización* de sentencias y filtrado de stopwords¹.

Megam (Hal Daumé III, 2004) es una herramienta para el cálculo de pesos de features en modelos de máxima entropía. En este trabajo se desarrolló una interfaz con esta herramienta para la implementación del algoritmo de *Max Ent* resultando de gran utilidad por su efectividad y velocidad de convergencia.

¹NLTK provee una lista de stopwords en español.

Sci-Kit Learn (Pedregosa et al., 2011) es un framework que provee una colección de algoritmos basados en aprendizaje supervisado para tareas de clasificación. En este trabajo se desarrolló una interfaz con esta herramienta para utilizar las implementaciones de los algoritmos *SVM* y *Decision Trees*.

Freeling (Padró and Stanilovsky, 2012) es un software que provee una colección de herramientas para tareas de análisis de textos en español. En este trabajo se desarrolló una interfaz con este software para tareas de *POS tagging*, *Stemming* y *Lematización*.

Además, se desarrollaron interfaces con las herramientas Weka (Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, Ian H. Witten, 2009) y *SVM^{Light}* (Joachims, 1999) que luego no se utilizaron en etapa de experimentación por haber obtenido mejores resultados con las herramientas antes mencionadas pero de todas formas la aplicación puede ser configurada para ejecutar la clasificación utilizando los modelos provistos por estos frameworks.

3.2.3 Adaptación del Algoritmo de Turney

Para adaptar el clasificador de Turney (ver sección 2.3.9) al idioma español se tuvieron las siguientes consideraciones:

- El operador *NEAR* se definió como la ocurrencia conjunta de los términos en la misma sentencia.
- Para la extracción de bigramas se utilizaron los siguientes patrones de opinión definidos en trabajos previos para el idioma español (Cruz et al., 2008):

Primera Palabra	Segunda Palabra	Tercera Palabra (No Extraída)
Adjetivo	Nombre	Cualquiera
Nombre	Adjetivo	No Nombre
Adverbio	Adjetivo	No Nombre
Adverbio	Verbo	Cualquiera
Verbo	Adverbio	Cualquiera

Tabla 3.1: Patrones de Turney para el Idioma Español

- Los términos utilizados para representar polaridad fueron:
 - Positivos: *excelente, excelentes, bueno/a, buenos/as, buenísimo/a, buenísimos/as, rico/a, ricos/as, espectacular, genial.*
 - Negativos: *mal, malo/a, malos/as, feo/a, feos/as, horrible, horribles, pesimo/a, pesimos/as, desastre, mediocre, NOT-bueno,*

*NOT_buenos, NOT_buena, NOT_buenas, NOT_rico, NOT_ricos,
NOT_rica, NOT_ricas.*

- Como técnica de smoothing para el tratamiento de aquellos términos que no aparecen en forma conjunta con los términos que indican polaridad se sumó 0.01 al resultado de la operación *NEAR*.

Capítulo 4

Casos de Estudio y Experimentación

En este capítulo presentaremos los casos de estudio elegidos y explicaremos las técnicas y criterios adoptados para la construcción de los corpus de datos con los que trabajaremos en la etapa de experimentación de este trabajo. Luego, describiremos las experiencias realizadas y presentaremos los resultados obtenidos a partir de la clasificación subjetiva, binaria y a nivel de documento de los conjuntos de datos construidos y analizaremos el impacto de la variación de parámetros de entrada en la efectividad de los clasificadores implementados.


4.1 Casos de Estudio


Para la evaluación de los distintos modelos en estudio construimos un conjunto de datos principal con el que realizaremos la mayoría de las experiencias de análisis de sentimientos y otro conjunto de datos que nos servirá para realizar pruebas conocidas como *out-of-domain testing*, es decir, entrenando con un conjunto de datos y clasificando otro de un dominio distinto. Esta experiencia es muy importante en modelos supervisados para analizar la efectividad de los clasificadores de forma objetiva como luego veremos en detalle.

El caso de estudio principal será el sitio de crítica gastronómica online www.guiaoleo.com. En este sitio los usuarios emiten opiniones sobre restaurantes de la ciudad de Buenos Aires y proveen una calificación en las categorías: comida, ambiente y servicio. Cada una de estas categorías puede ser calificada como mala/regular, buena, muy buena o excelente asignado un puntaje como se observa en la figura 4.1.




1180 Comentarios Ordenados por fecha o por [experiencia de usuario](#)

Ver votos: Todos, [excelentes](#), [buenos/muy buenos](#), [regulares/malos](#)




chucky68  ★★★★★

13 Jul 2013 Festejamos el cumpleaños de un amigo. El lugar lleno total es poco. Habia gente esperando afuera cuando llegamos , y habia gente esperando cuando salimos. Teniamos reserva, pero la demora promedio es de 1 hora (dicho por el de la puerta). Comimos re bien, variado: falafel, keppe, ensalada belen, y mas. El postre, copa Sarkis, una bomba riquisima. La relacion calidad/precio es correcta.




Comida:  Servicio:  Ambiente: 

[Muy útil](#) • [Responder](#) • [Reportar indebido](#)





lucia66 ★★★★★

13 Jul 2013 Excelente lugar!!! Hemos ido con la familia, ahora volvimos con mi pareja. Comimos muy bien! Bien asesorados en cuanto a cuanto pedir, platos abundantes (y para nada caros), muy ricos y poco comunes en Bs. As. Volveremos!!




Comida:  Servicio:  Ambiente: 

[Muy útil](#) • [Responder](#) • [Reportar indebido](#)




PupiSoy  ★★★★★

6 Jul 2013 Muy buen lugar, muy buena atención. Los platos son variados y muy bien elaborados. Las opciones vegetarianas son sumamente recomendables.




Comida:  Servicio:  Ambiente: 

[Muy útil](#) • [Responder](#) • [Reportar indebido](#)





alchemy ★★★★★

5 Jul 2013 Deliciosa comida árabe. Lo único que hay que esperar bastante...




Comida:  Servicio:  Ambiente: 

[Muy útil](#) • [Responder](#) • [Reportar indebido](#)



JJAJ  ★★★★★

26 Jun 2013 Fuimos 6 a cenar, nos costo una hora conseguir mesa, pero valió la pena, comimos muy muy bien y con un precio muy adecuado, ya hemos ido muchas veces y siempre nos dan ganas de volver.

Comida:  Servicio:  Ambiente: 

[Muy útil](#) 1 voto • [Responder](#) • [Reportar indebido](#)

[Ver más comentarios](#)

Figura 4.1: Caso de Estudio: Guía Óleo

Otro caso de estudio con el que trabajaremos será el sitio play.google.com. En este sitio se presentan las opiniones emitidas por los usuarios sobre aplicaciones para dispositivos móviles cuyo sistema operativo es Android como podemos observar en la figura 4.2. Los usuarios emiten opiniones y califican las aplicaciones asignando un puntaje general de 1 a 5 estrellas, siendo 1 estrella la

peor calificación y 5 la mejor.

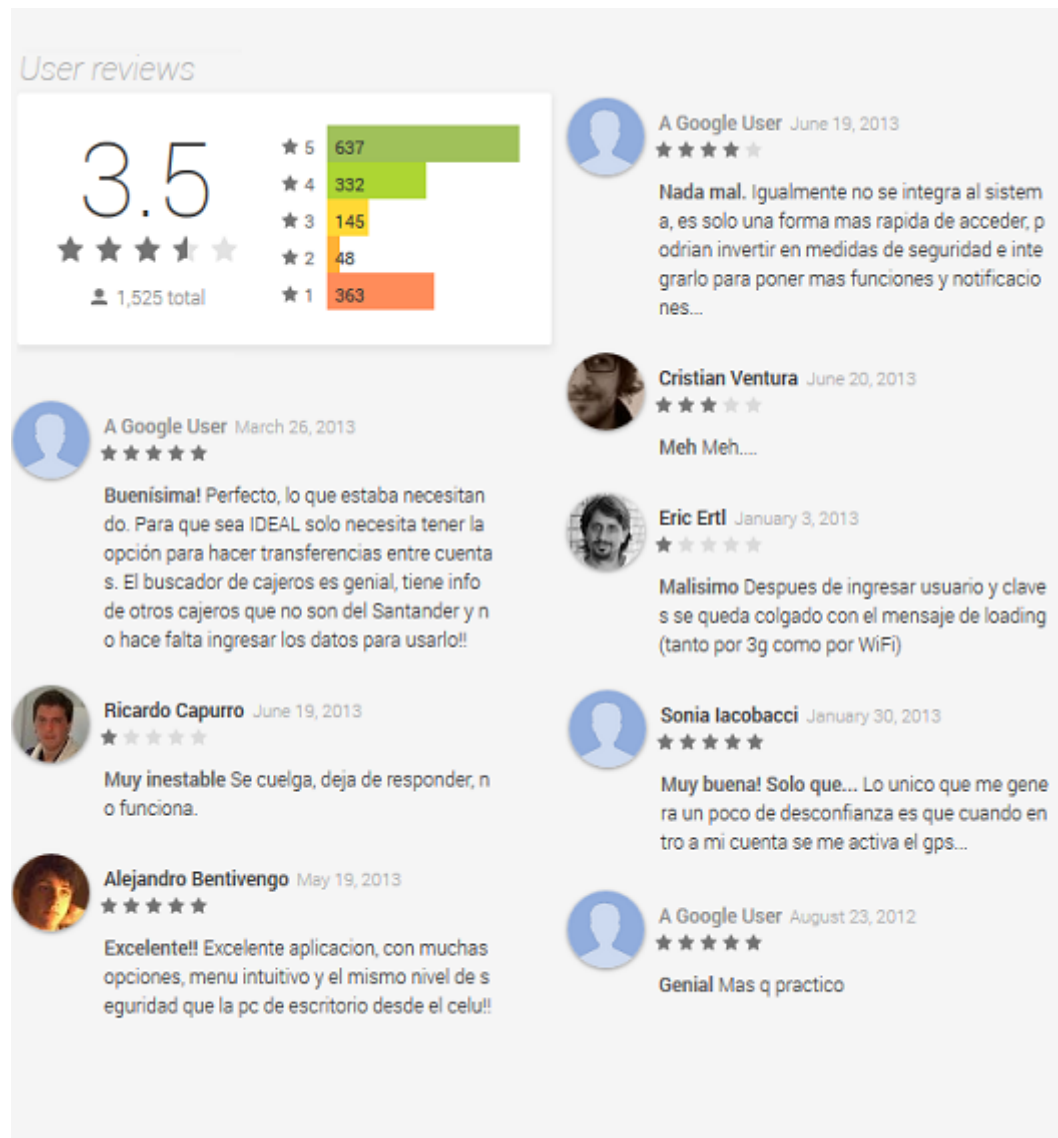


Figura 4.2: Caso de Estudio: Google Play

4.1.1 Construcción de los Corpus de Datos

Como hemos visto en la sección anterior, los sitios que utilizaremos para construir los corpus de datos utilizan un sistema de calificación basado en puntajes. Este sistema resulta de mucha utilidad en algoritmos de aprendizaje supervisado ya que permite definir un criterio para la asignación de una etiqueta a cada comentario del conjunto de datos (positivo, negativo o neutro) y en base a esta información entrenar el clasificador evitando la clasificación manual.

Además, utilizaremos esta categorización para comparar los resultados de los clasificadores con las clases esperadas y evaluar la performance de cada modelo.

Para la construcción de los corpus de datos correspondientes a los casos de estudio presentados se desarrolló un script en lenguaje Python que recorre automáticamente¹ todas las páginas de críticas de ambos sitios y extrae los comentarios y puntajes asignados por los usuarios. Luego, se implementó un segundo procesamiento para la asignación de etiquetas a los comentarios a partir de la calificación de los usuarios siguiendo los criterios definidos a continuación para cada caso.

Corpus de Críticas sobre Restaurantes: Guía Óleo

Para la asignación de etiquetas en la construcción de este corpus de datos consideramos las siguientes categorías y sistema de puntajes de la guía:

Categorías: {COMIDA, AMBIENTE, SERVICIO}

Puntajes:

- 1 = malo/regular
- 2 = bueno
- 3 = muy bueno
- 4 = excelente

El criterio de asignación de clases (o etiquetas) a los comentarios fue el siguiente:

1. Se asignó la etiqueta “POSITIVO” a aquellos comentarios cuya suma de puntos en las categorías mencionadas sea 10 o superior. Esto significa que extraeremos aquellos comentarios que tengan 4 puntos (excelente) en al menos una categoría y 3 puntos (muy bueno) o más en las restantes.
2. Se asignó la etiqueta “NEGATIVO” a aquellos comentarios cuya puntaje en la categoría “Comida” (identificada como la más relevante para el dominio estudiado) sea 1 (malo/regular) o bien sea 2 (bueno) y que el resto de las categorías tengan ambas puntaje 1 (malo/regular).
3. El resto de los comentarios fueron considerados “NEUTRO” y no se incluyeron en el corpus de datos.

El conjunto de datos final se encuentra disponible en la siguiente url: <http://web.fi.uba.ar/~ldubiau/datasets/restaurante-review-dataset.zip> e incluye un total de 34808 comentarios positivos y 17633 negativos agrupados dentro de los directorios “pos” y “neg” respectivamente. Cada directorio contiene un archivo en formato json² por cada restaurante con los comentarios correspondientes a la clase.

Corpus de Críticas sobre Aplicaciones para Android: Google Play

En el caso de el corpus de datos de Google Play y considerando que los usuarios asignan un único puntaje general a la aplicación, el criterio adoptado fue asignar

¹La técnica de extraer información de sitios web automáticamente es conocida como Web Crawling.

²<http://www.json.org/>

la etiqueta "POSITIVO" a aquellos comentarios calificados por el usuario con 5 estrellas (excelente) y la etiqueta "NEGATIVO" a los comentarios cuya calificación fue de 1 estrella (mala). Además, se tuvieron en cuenta en este segundo procesamiento sólo aquellos comentarios de más de 5 palabras.

El conjunto de datos final se encuentra disponible en la siguiente url: <http://web.fi.uba.ar/~ldubiau/datasets/androidapps-review-dataset.zip> e incluye un total de 5700 comentarios positivos y 2581 negativos siguiendo los mismos criterios de formato definidos en la sección anterior para el corpus de Guía Óleo.

4.2 Experimentación

En esta sección presentamos los resultados obtenidos a partir de la clasificación de los corpus de datos descritos en la sección 4.1 en función de los preprocesos aplicados, el modelo de clasificación, los distintos tamaños de corpus y los atributos extraídos.

Para medir la efectividad de los clasificadores en las experiencias con corpus balanceados nos basamos en la medida de *accuracy* que se obtiene a partir de la relación entre la cantidad de casos clasificados correctamente y el total de casos mientras que en experiencias con corpus desbalanceados utilizamos la medida F1 como se explica en la sección 2.3.14.

Todas las experiencias presentadas en esta sección fueron ejecutadas para tamaños de corpus entre 500 y 22000 documentos del conjunto de datos. Este tamaño máximo fue seleccionado en base al costo de entrenamiento de los métodos supervisados y teniendo en cuenta que la mejora obtenida para corpus mayores no se consideró representativa.

Las tablas completas de resultados obtenidos en las experiencias de este capítulo pueden consultarse en el Anexo B.

4.2.1 Selección de Atributos

Como hemos mencionado, los atributos o *features* con los que experimentaremos en este capítulo para métodos supervisados incluyen: presencia y frecuencia de unigramas, presencia de bigramas, presencia de adjetivos y combinaciones de ellos.

Teniendo en cuenta el costo de entrenamiento de los métodos supervisados y con el objetivo de evitar el *overfitting* se adoptó el siguiente criterio para la selección de atributos: para unigramas se extrajeron aquellos cuya frecuencia de aparición es mayor a 10; en el caso de adjetivos y bigramas la frecuencia mínima requerida fue de 4 y en todos los casos se utilizó un máximo de 3000 atributos. Esta decisión está basada en experiencias que se realizaron sobre los datos y utilizando el método *cut-off* que se menciona en el trabajo de (Christopher Potts, 2011).

4.2.2 Efectividad de Preprocesadores

Para definir los preprocesamientos más efectivos en esta tarea específica de análisis de sentimientos y para el dominio en estudio, ejecutamos la clasificación para el tamaño máximo de corpus y utilizando cada preprocesamiento de forma aislada y en conjunto para analizar el impacto en los resultados. Esta experiencia se ejecutó utilizando un método supervisado con los parámetros que se indican en la tabla 4.1.

Parámetro	Valor
Corpus	Guía Óleo
Algoritmo de Clasificación	Naïve Bayes
Algoritmo de Validación	5-fold Cross Validation
Tamaño de Corpus	22000 documentos
Corpus Balanceado	Si
Features	Presencia de Unigramas
Preprocesamientos	Variable

Tabla 4.1: Efectividad de Preprocesadores para Clasificación Supervisada: Parámetros de la Experiencia

En la tabla 4.2 se observan los valores de accuracy y porcentajes de mejora obtenidos aplicando cada preprocesamiento y combinaciones de ellos. Para las experiencias de clasificación supervisadas que siguen en este trabajo utilizaremos la combinación de preprocesadores que representa el porcentaje de mejora más alto según se observa en la figura 4.3, es decir, eliminación de stopwords, tratamiento de negaciones, filtrado por tamaño de palabra, eliminación de signos de puntuación, reemplazo de caracteres especiales y transformación a minúsculas.

Preproceso	Accuracy	Mejora (%)
NP (Sin Preproceso)	0.868	
SW (Eliminación de stopwords)	0.895	3.11%
NEG (Tratamiento de negaciones)	0.875	0.81%
WL (Filtrado de palabras de menos de 3 caracteres)	0.883	1.73%
DC (Eliminación de caracteres repetidos más de 2 veces)	0.867	-0.12%
STEM (Stemming)	0.861	-0.81%
LC (Transformación de capitalizaciones)	0.868	0.00%
PUNCT (Eliminación de signos de puntuación)	0.871	0.35%
SC (Transformación de caracteres especiales)	0.869	0.12%
LEMMA (Lematización)	0.867	-0.12%
Combinación 1: SW + NEG	0.905	4.26%
Combinación 2: SW + NEG + WL	0.911	4.95%
Combinación 3: SW + NEG + WL + PUNCT	0.914	5.3%
Combinación 4: SW + NEG + WL + PUNCT + SC	0.918	5.76%
Combinación 5: SW + NEG + WL + PUNCT + SC + LC	0.924	6.45%
Combinación 6: SW + NEG + WL + PUNCT + SC + LC + LEMMA	0.92	5.99%
Combinación 7: SW + NEG + WL + PUNCT + SC + LC + DC	0.924	6.45%
Combinación 8: SW + NEG + WL + PUNCT + SC + LC + STEM	0.918	5.76%

Tabla 4.2: Efectividad de Preprocesadores para Clasificación Supervisada: Valores de Accuracy

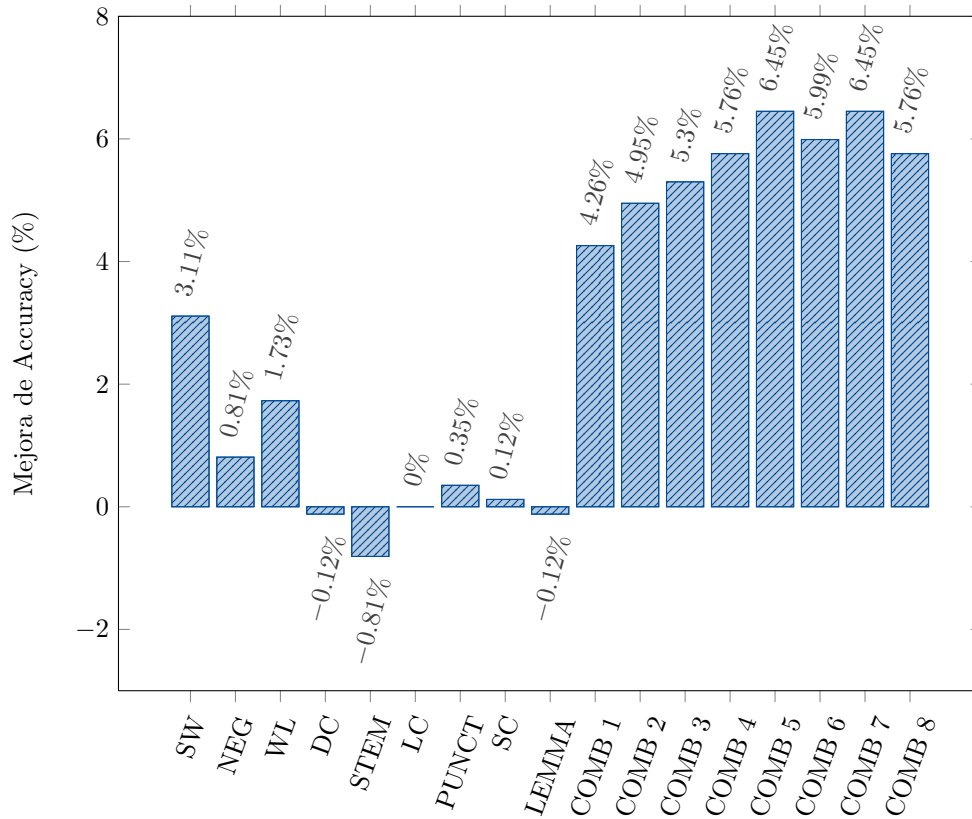


Figura 4.3: Efectividad de Preprocesadores para Clasificación Supervisada: Mejora de Accuracy (%) vs Preprocesos

En el caso de los modelos no supervisados, como ocurre en el clasificador de Turney, no utilizamos preprocesamientos como lematización o stemming dado que la clasificación se basa en la ocurrencia conjunta de los términos del documento con los términos que indican polaridad y éstos son fijos. En el caso de tratamiento de negaciones se incluyeron las palabras negadas más representativas de cada clase al léxico de opinión para poder utilizar este preprocesamiento. Por otro lado hemos hallado que la eliminación de stopwords y el filtro de palabras por mínimo de longitud no representa una mejora en este tipo de clasificadores por lo que el criterio adoptado para métodos no supervisados fue el de utilizar como preprocesamientos el tratamiento de negaciones, transformación a minúscula y reemplazo de caracteres especiales y signos de puntuación.

4.2.3 Efectividad de Clasificadores

En esta sección presentaremos todos los resultados de performance obtenidos en la clasificación del corpus de datos principal, utilizando los preprocesamientos seleccionados y los distintos modelos de clasificación, en función de las variantes de features, tamaño de corpus, balanceo del conjunto de datos y utilizando distintas técnicas de validación.

Clasificadores Supervisados

En las figuras 4.4 y 4.5 observamos los valores de accuracy obtenidos para cada clasificador supervisado, en función del tamaño de corpus y atributos extraídos, presentados en gráficas por atributo y por algoritmo respectivamente. Los parámetros con los que se ejecutó esta experiencia son los indicados en la tabla 4.3.

Parámetro	Valor
Corpus	Guía Óleo
Algoritmo de Clasificación	Variable
Algoritmo de Validación	5-fold Cross Validation
Tamaño de Corpus	Variable
Corpus Balanceado	Si
Features	Variable
Preprocesamientos	Seleccionados (ver sección 4.2.2)

Tabla 4.3: Efectividad de Clasificadores Supervisados: Parámetros de la Experiencia

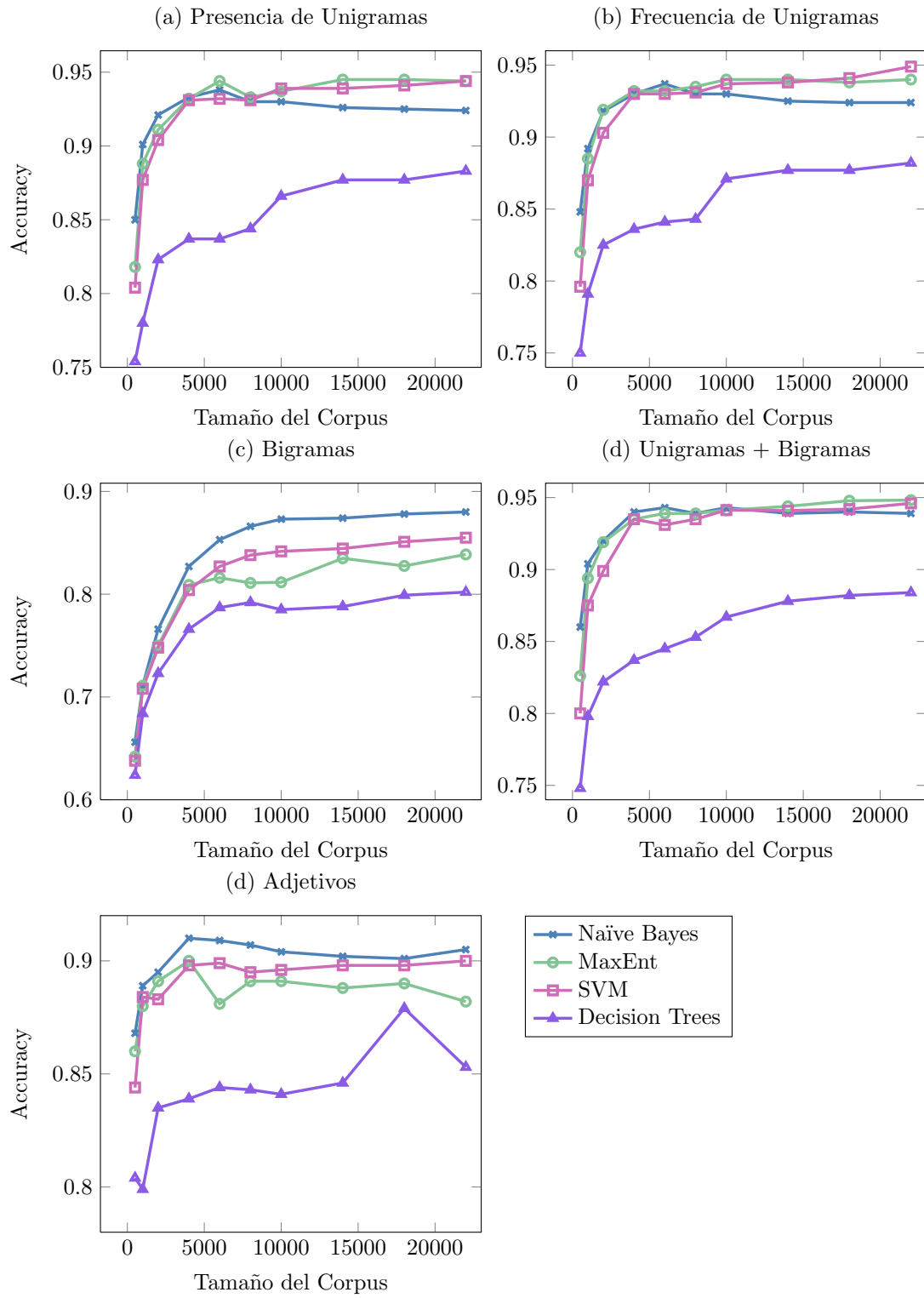


Figura 4.4: Efectividad de Clasificadores Supervisados por Atributo: Accuracy vs Tamaño de Corpus

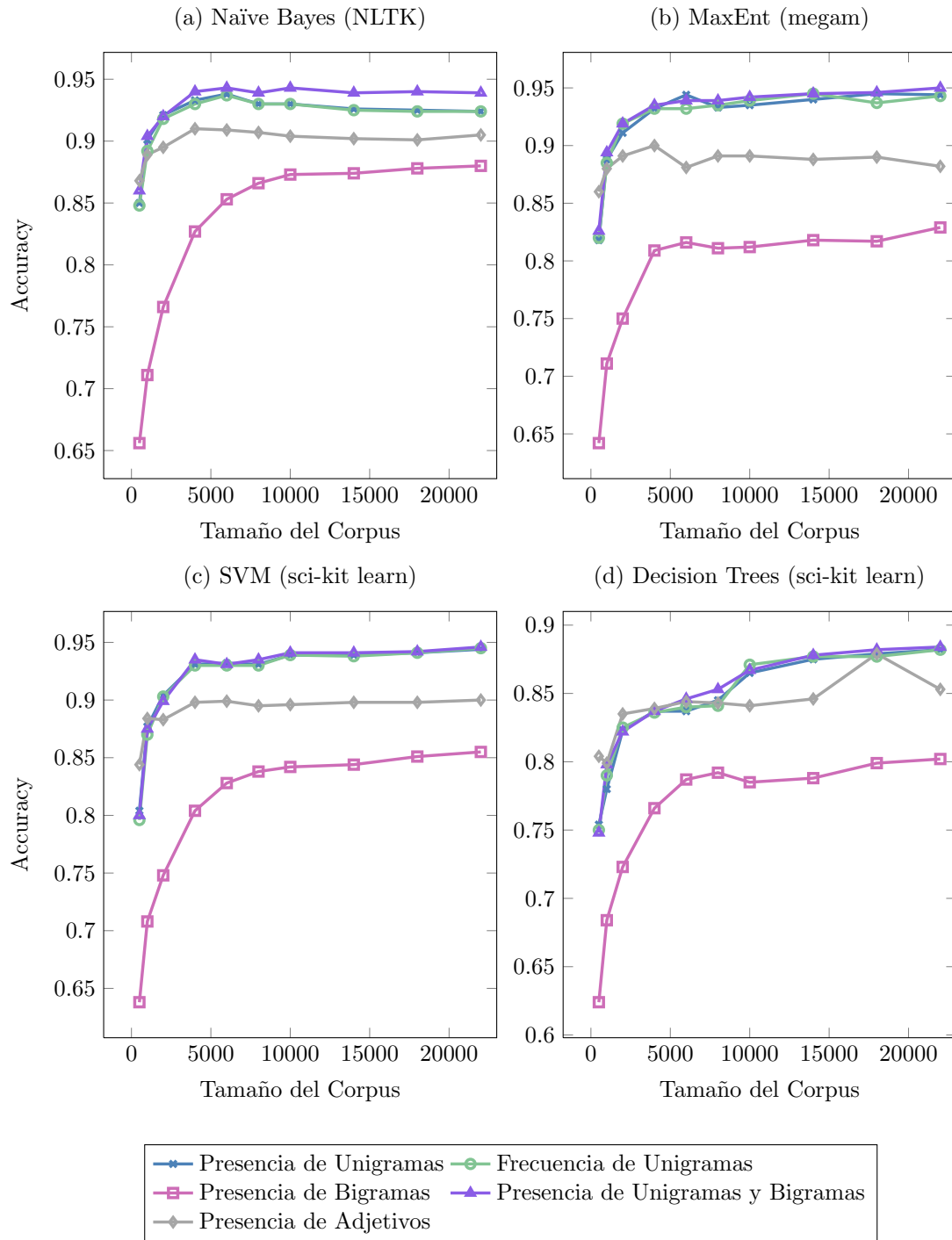


Figura 4.5: Efectividad de Clasificadores Supervisados por Algoritmo: Accuracy vs Tamaño de Corpus

Clasificadores No Supervisados

En la figura 4.6 presentamos los valores de accuracy obtenidos para el algoritmo no supervisado de Turney en función del tamaño de corpus. En cuanto a los preprocesamientos de texto luego de una variedad de pruebas hemos comprobado que eliminar stopwords y filtrar palabras por tamaño mínimo de longitud no producía mejoras en los resultados de ese tipo de clasificadores. En el caso de las negaciones, como hemos mencionado en la sección 3.2.3, en la implementación del algoritmo se agregaron las palabras negadas más relevantes al conjunto de términos con polaridad negativa y se utilizó el preprocesamiento de tratamiento de negaciones obteniendo una mejora considerable en los resultados.

Los parámetros con los que ejecutamos esta experiencia se presentan en la tabla 4.4.

Parámetro	Valor
Corpus	Guía Óleo
Algoritmo de Clasificación	Turney
Algoritmo de Validación	Hold-out
Tamaño de Corpus	Variable
Corpus Balanceado	Si
Preprocesamientos	Seleccionados (ver sección 4.2.2)

Tabla 4.4: Efectividad de Clasificadores No Supervisados: Parámetros de la Experiencia

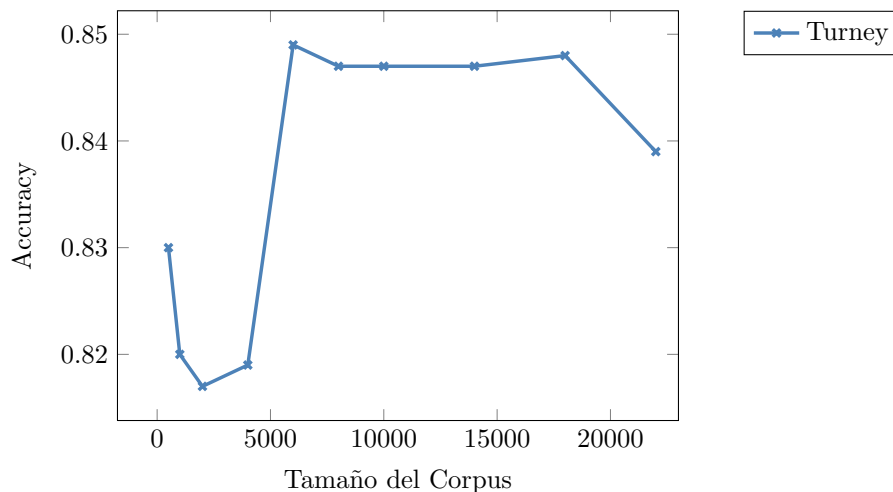


Figura 4.6: Efectividad de Clasificadores No Supervisados: Accuracy vs. Tamaño de Corpus

Clasificadores Supervisados vs No Supervisados

En esta sección realizaremos la comparación de efectividad de todos los modelos en estudio para máximo tamaño de corpus y en el caso de los métodos supervisados presentaremos los resultados en función de los features extraídos. En la tabla 4.6 pueden observarse los valores máximos de accuracy obtenidos en esta experiencia y luego en la figura 4.7 se observa la comparación gráfica de los resultados. Los parámetros con los que se ejecutó esta experiencia son los indicados en la tabla 4.5.

Parámetro	Valor
Corpus	Guía Óleo
Algoritmo de Clasificación	Variable
Algoritmo de Validación	5-fold Cross Validation para métodos supervisados, Hold-out para métodos no supervisados
Tamaño de Corpus	22000 documentos
Corpus Balanceado	Si
Features	Variable
Preprocesamientos	Seleccionados (ver sección 4.2.2)

Tabla 4.5: Comparación de Clasificadores Supervisados y No Supervisados por Feature: Parámetros de la Experiencia

Feature	NB	MaxEnt	SVM	DT
Presencia de Unigramas	0.924	0.944	0.944	0.882
Frecuencia de Bigramas	0.924	0.943	0.945	0.882
Presencia de Bigramas	0.88	0.829	0.855	0.802
Presencia de Unigramas y Bigramas	0.939	0.950	0.946	0.884
Presencia de Adjetivos	0.905	0.882	0.900	0.853
Turney				
Patrones de Opinión	0.839			

Tabla 4.6: Comparación de Clasificadores Supervisados y No Supervisados por Feature: Valores de Accuracy

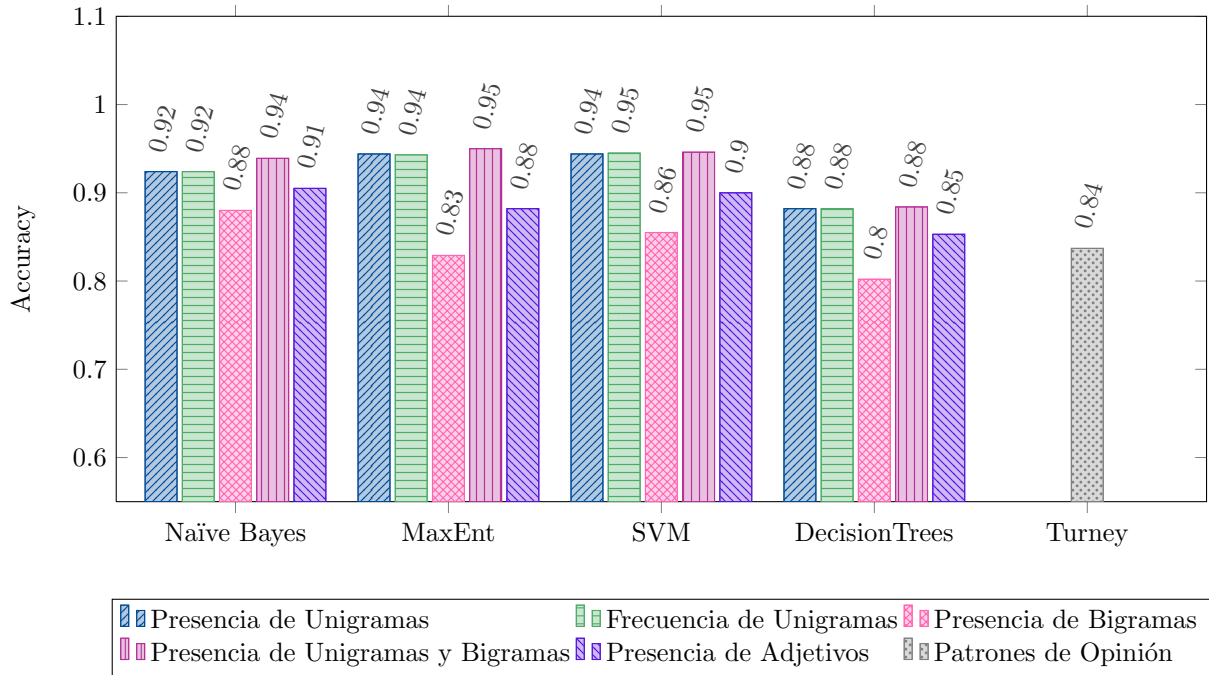


Figura 4.7: Comparación de Clasificadores Supervisados y No Supervisados por Feature: Algoritmo vs Accuracy

4.2.4 Cambio de Dominio

En el caso de los clasificadores supervisados una de las pruebas más importantes que debemos realizar para medir la efectividad de un modelo es lo que se conoce como *out-of-domain testing* (Christopher Potts, 2011), esto significa que entrenamos el modelo con un corpus de datos perteneciente a un dominio específico y clasificamos otro conjunto de datos correspondiente a un dominio distinto.

Este tipo de pruebas nos permiten conocer el comportamiento del modelo cuando no ha sido refinado específicamente para un dominio en particular y su efectividad en cuanto a generalización de información desconocida. La importancia de esta experiencia radica en que por lo general cuando se utilizan los clasificadores para predecir la polaridad de un conjunto de documentos no se tiene información conocida para entrenar el clasificador en el dominio específico y se deben utilizar datos de entrenamiento pertenecientes a otros dominios para los que se tenga información de las clases.

En esta sección presentaremos experiencias en las que el entrenamiento de los distintos modelos supervisados se realiza con el corpus de Guía Óleo y luego la clasificación se realiza sobre el corpus de Google Play (ver sección 4.1). Esta experiencia se ejecutó con los parámetros indicados en la figura 4.7

Parámetro	Valor
Training Corpus	Guía Óleo
Test Corpus	Google Play
Algoritmo de Clasificación	Variable
Algoritmo de Validación	5-fold Cross Validation
Tamaño de Corpus	Variable
Corpus Balanceado	Si
Features	Presencia de Unigramas
Preprocesamientos	Seleccionados (ver sección 4.2.2)

Tabla 4.7: Efectividad de Clasificadores Supervisados para Cambio de Dominio: Parámetros de la Experiencia

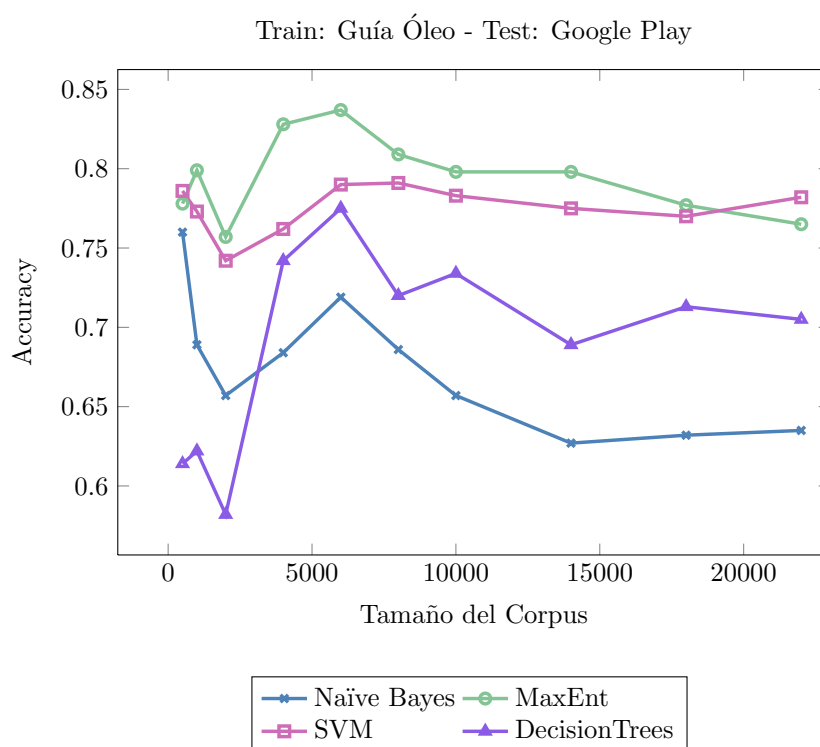


Figura 4.8: Efectividad de Clasificadores Supervisados para Cambio de Dominio: Accuracy vs Tamaño de Corpus

A continuación veremos los resultados de performance obtenidos para clase cuando se experimenta cambiando de dominio utilizando el tamaño máximo de corpus.

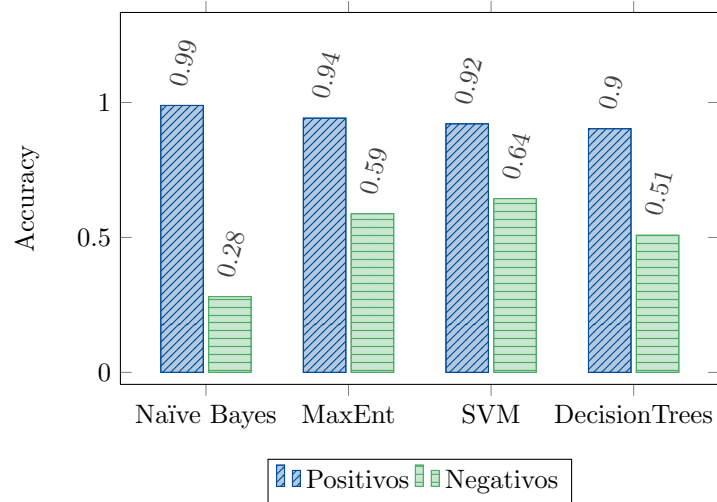


Figura 4.9: Efectividad de Clasificadores Supervisados por Clase para Cambio de Dominio y Máximo Tamaño de Corpus: Algoritmo vs Accuracy

4.2.5 Corpus Balanceado vs Desbalanceado

En esta experiencia analizaremos el comportamiento de los métodos en estudio cuando el conjunto de datos se encuentra fuertemente desbalanceado y compararemos los resultados con los obtenidos para corpus balanceados.

Como hemos explicado en la sección 2.3.14 cuando el corpus está fuertemente desbalanceado la medida de accuracy puede no ser suficiente para medir la efectividad de un método debido a que podríamos obtener muy buenos resultados para la clase mayoritaria y no hacerlo para la otra clase y de todos modos obtendríamos un alto nivel de accuracy. Por esto, en este caso utilizaremos la medida F1 para medir efectividad y analizar el comportamiento de los distintos modelos.

En las figuras 4.10 y 4.11 observamos los valores de F1 obtenidos para los distintos algoritmos en función de la variación del tamaño de corpus. Los parámetros con los que se ejecutaron estas experiencias son los que se indican en la tabla 4.8.

Parámetro	Valor
Corpus	Guía Óleo
Algoritmo de Clasificación	Variable
Algoritmo de Validación	5-fold Cross Validation para métodos supervisados, Hold-out para métodos no supervisados
Tamaño de Corpus	Variable
Corpus Balanceado	Variable
Features	Presencia de Unigramas
Preprocesamientos	Seleccionados (ver sección 4.2.2)

Tabla 4.8: Efectividad de Clasificadores para Corpus Balanceados y Desbalanceados: Parámetros de la Experiencia

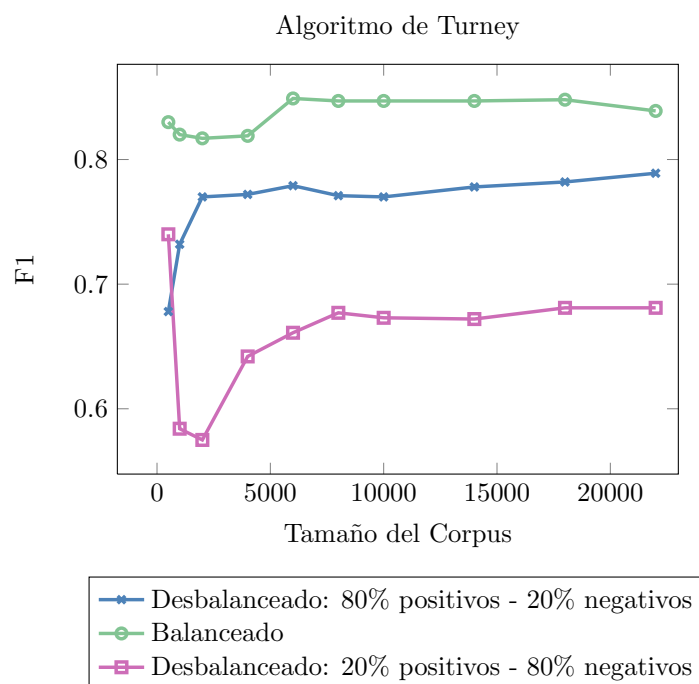


Figura 4.10: Efectividad de Clasificadores No Supervisados para Corpus Balanceados y Desbalanceados: F1 vs Tamaño de Corpus

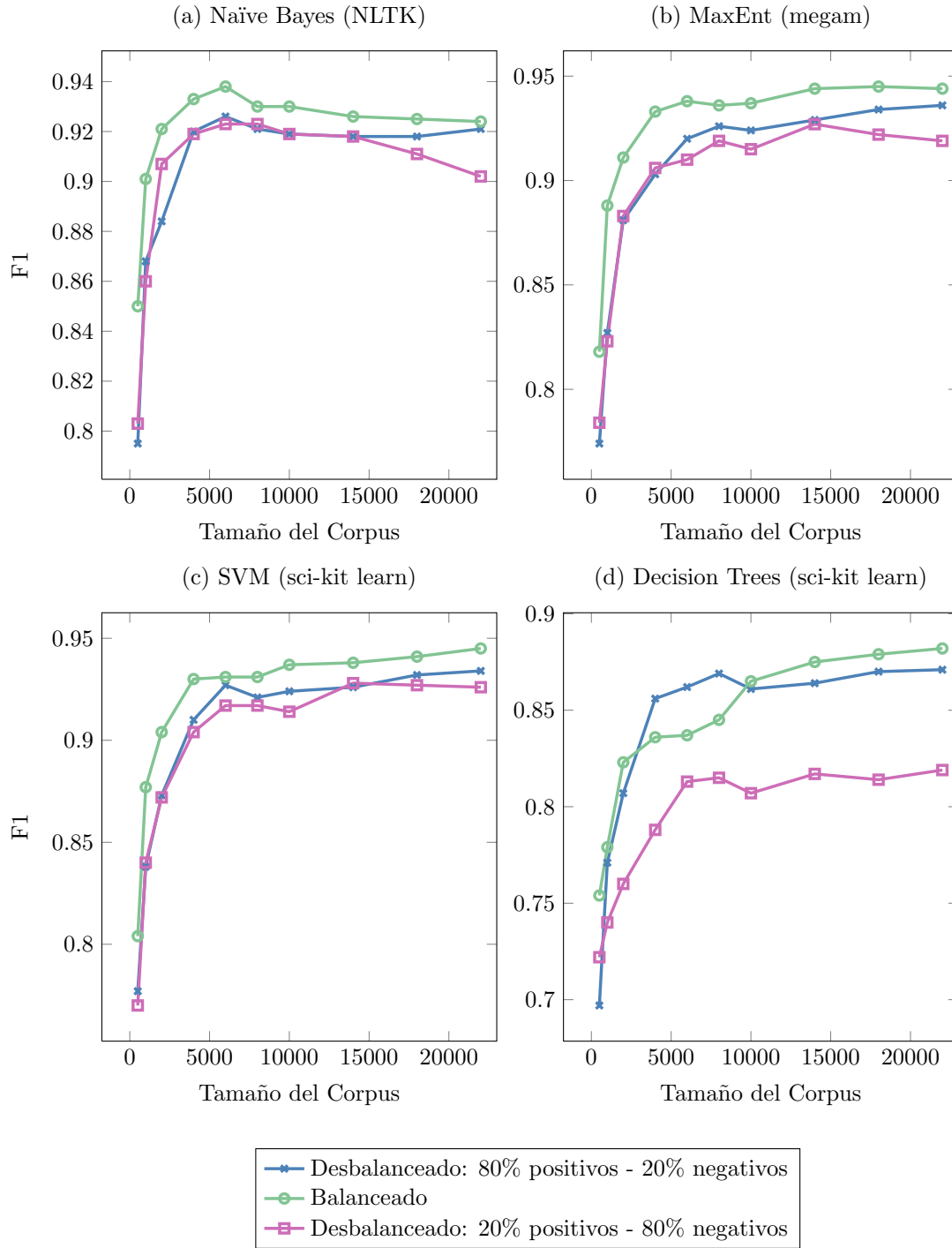


Figura 4.11: Efectividad de Clasificadores Supervisados para Corpus Balanceados y Desbalanceados: F1 vs Tamaño de Corpus

Clasificadores Supervisados vs No Supervisados para Corpus Desbalanceados

En esta sección realizaremos una comparación de todos los algoritmos en estudio en función de la proporción de desbalanceo de clases para el tamaño máximo de corpus de datos. En la tabla 4.9 se presentan los valores de accuracy y F1 obtenidos y luego estos valores se comparan gráficamente en la figura 4.12.

Balanceo de Corpus	NB		MaxEnt		SVM		DT		Turney	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Desbalanceado										
80% POS - 20% NEG	0.915	0.921	0.925	0.936	0.930	0.934	0.874	0.871	0.850	0.789
Balanceado	0.924	0.924	0.944	0.944	0.944	0.944	0.882	0.882	0.839	0.839
Desbalanceado										
20% POS - 80% NEG	0.932	0.902	0.907	0.918	0.928	0.926	0.832	0.817	0.793	0.681

Tabla 4.9: Efectividad de Clasificadores Supervisados y No Supervisados para Corpus Desbalanceados y Máximo Tamaño de Corpus: Valores de Accuracy y F1

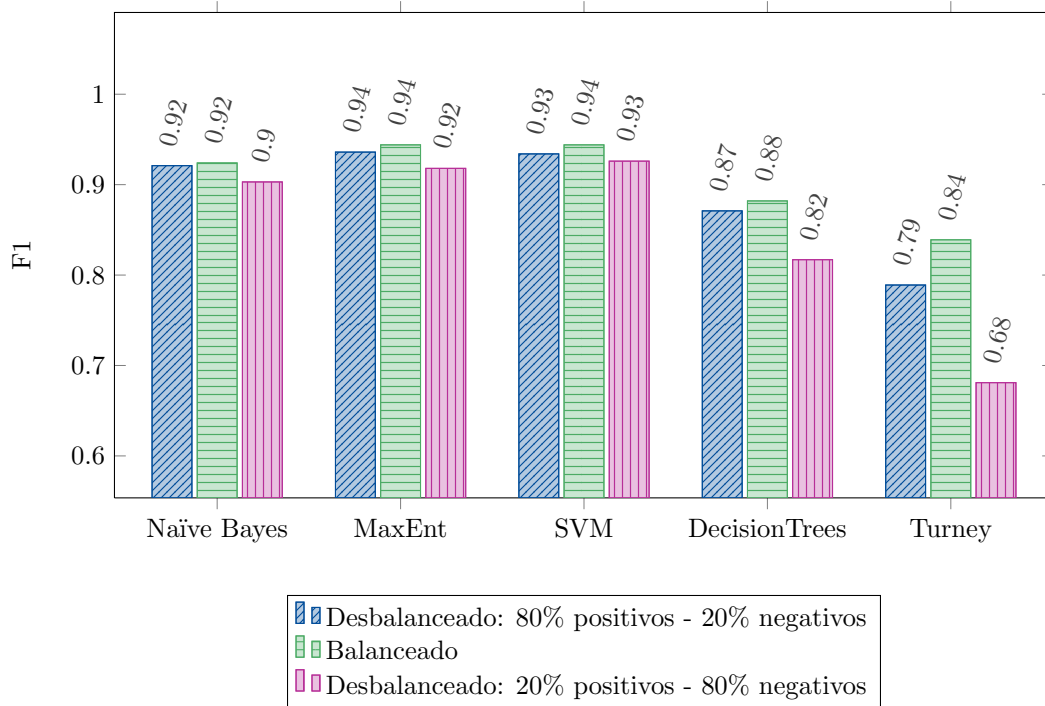


Figura 4.12: Efectividad de Clasificadores Supervisados y No Supervisados para Corpus Desbalanceados y Máximo Tamaño de Corpus: Algoritmo vs F1

4.2.6 Análisis de Resultados

A partir de las experiencias con los distintos preprocesamientos de texto aplicados y en función de las mejoras de performance obtenidas realizamos las siguientes observaciones que surgen de la figura 4.3:

- Hay preprocesamientos que aplicados en forma aislada no representan una mejora pero sí lo hacen cuando son aplicados en combinación con otros como se observa para el caso de transformación a minúscula.
- Contrario a lo que ocurre en tareas de IR (Recuperación de Información), realizar Stemming o Lematización sobre el texto empeora los resultados.
- La mejora que representa la combinación de preprocesos seleccionada resulta mayor que la suma de las mejoras provistas por cada preprocesamiento aplicado en forma aislada.

Si bien los preprocesamientos dependen del corpus con el que se trabaje, en esta experiencia obtuvimos que utilizando preprocesadores genéricos adaptados al idioma español puede mejorarse considerablemente la performance del clasificador.

En cuanto a la efectividad de los algoritmos de clasificación realizamos las siguientes observaciones que surgen de la figura 4.4:

- Naïve Bayes arroja los mejores resultados para corpus pequeños pero su performance decrece levemente para los tamaños de corpus más grandes.
- MaxEnt y SVM mejoran su performance a medida que crece el tamaño de corpus y se obtiene la máxima performance alcanzada por la experiencia.
- Si se utilizan adjetivos o bigramas como atributos Naïve Bayes presenta los mejores resultados.
- La performance del clasificador basado en árboles de decisión es notablemente peor que la obtenida con los otros modelos supervisados analizados, por lo que podemos concluir que este clasificador no resulta adecuado para este tipo de tarea.
- Para todos los modelos se observa que a partir de un tamaño de corpus de 8000 documentos, el valor de performance se vuelve relativamente estable.

En la figura 4.5 realizamos las siguientes observaciones respecto de la performance de los distintos modelos supervisados en función de los tipos de features extraídos:

- Para todos los clasificadores supervisados los mejores resultados se obtienen utilizando como atributos la combinación de presencia de unigramas y bigramas aún cuando para todos los clasificadores la performance al utilizar sólo bigramas no resulta favorable.
- Como ya se ha mostrado para el inglés en trabajos anteriores (Bo Pang, Lillian Lee and Shivakumar Vaithyanathan, 2002), comprobamos que en términos de información subjetiva considerar frecuencia de unigramas como atributos no representa una mejora notable con respecto a presencia.
- Utilizar adjetivos como atributos arroja resultados considerablemente peores que utilizando todos los unigramas, contrario a lo que podíamos esperar considerando que la mayor ganancia de información en cuanto a opiniones es provista por los adjetivos.

En la figura 4.6 se observa que la performance del algoritmo de Turney mejora levemente a medida que crece el tamaño de corpus pero este crecimiento resulta inestable. Luego, en la figura 4.7 podemos observar la comparación con métodos supervisados, si bien el algoritmo de Turney no alcanza la efectividad máxima de éstos para presencia de unigramas, se obtienen muy buenos resultados comparables con los obtenidos utilizando bigramas como atributos en métodos supervisados, con la ventaja de que no requiere un corpus de entrenamiento.

En cuanto a las experiencias realizadas utilizando un corpus para entrenamiento y otro de un dominio distinto para clasificación supervisada, observamos en la figura 4.8 que la performance decrece considerablemente en comparación con la clasificación del mismo dominio pero sin embargo MaxEnt y SVM son los modelos que responden mejor al cambio. A partir de estos resultados y comparando con los obtenidos con el algoritmo de Turney, podemos concluir que en casos donde no se tenga un conjunto de datos de entrenamiento propio del dominio conviene utilizar un algoritmo no supervisado o semi-supervisado (por ejemplo, basado en léxico de opinión) en lugar de un método supervisado

con un conjunto de entrenamiento de otro dominio.

Además, en la figura 4.9 observamos que la clase de comentarios negativos es la que se predice peor cuando se realiza un cambio de dominio, esto puede estar relacionado con que los términos utilizados para expresar polaridad negativa son más específicos de dominio en los casos de estudio analizados que los términos de polaridad positiva donde la performance de la experiencia resulta mayor al 90% para todos los modelos.

En las figuras 4.10, 4.11 y 4.12 observamos el comportamiento de los distintos modelos ante la variación de la proporción de comentarios positivos y negativos en el conjunto de datos, es decir, realizando el entrenamiento con corpus desbalanceados. En el caso del algoritmo de Turney se observa que la performance decrece en gran medida cuando la clase de comentarios positivos es poco representativa mientras que la performance alcanzada cuando la clase de comentarios negativos es poco representativa es muy cercana al caso de corpus balanceados. Al igual que ocurría en la experiencia de cambio de dominio notamos que en los casos de estudio analizados, los términos más polares de la clase de comentarios positivos son en gran medida más representativos que los términos negativos.

En el caso de los métodos supervisados se observa tanto para Naïve Bayes como MaxEnt y SVM que en los dos casos de presencia de cada clase la performance alcanzada es muy cercana a la obtenida en experiencias balanceadas por lo que el desbalanceo no representa un problema mayor para este tipo de modelos. En el caso de árboles de decisión, la performance sí se ve afectada considerablemente cuando la presencia de comentarios positivos es escasa.

4.2.7 Dificultades

Las principales dificultades en tareas de AS con las que nos hemos encontrado en la etapa de experimentación de este trabajo incluyen:

- Informalidad del texto: errores gramaticales y ortográficos.
- Textos demasiado cortos.
- Negaciones no tratadas: por ejemplo *“es caro y tampoco es excelente”*.
- Comparaciones no tratadas: por ejemplo *“En la Farola de Belgrano se come mejor.”*.
- Uso de términos específicos de dominio para indicar polaridad: por ejemplo en el corpus de Google Play donde el término *“funciona”* y sus formas flexionadas se utilizan en la mayoría de los comentarios mientras que este término no es utilizado en un corpus como el de Guía Óleo.
- Comentarios en otros idiomas.
- Comentarios irónicos.

CAPÍTULO 4. CASOS DE ESTUDIO Y EXPERIMENTACIÓN

Documento	Clase	Clase Asignada				
		NB	ME	SVM	DT	Turney
1 “Es muy caro para lo poco que sirven y tampoco es de tan buena calidad. La atencion es buena, pero no profesional”	NEG	POS	NEG	NEG	NEG	POS
2 “Comida impresentable”	NEG	POS	POS	POS	POS	NEG
3 “Todo impecable, lindo, fashion y glamuroso. La comida? En la Farola de Belgrano se come mejor. Creo que para dar de comer tan mal hay que poner ganas y esfuerzo”	NEG	POS	NEG	NEG	POS	NEG
4 “We sit down and 30 MINUTES later a kid comes over and gives us menus. 10 minutes later, a girl comes over no hello or welcome...”	NEG	POS	POS	POS	POS	NEG
5 “Sin lugar a dudas la pero atención de Buenos Aires!!!!”	NEG	POS	POS	POS	POS	POS
6 “Excelente la ubicacion frente a la plaza, la picada de \$40 incluia 4 platitos con aceituna, queso, salame y jamon (que no provocaban ser comidos) lo mejor el volcan de papas fritas. El servilletero y los condimentos no se podian tocar”	NEG	POS	POS	POS	POS	POS
7 “Excelentísima propuesta !! Viajamos mucho y este restaurante logro sorprendernos. Diseño, luz, comida, música? pasamos una noche realmente encantadora.”	POS	POS	NEG	NEG	NEG	NEG

Tabla 4.10: Ejemplos de Dificultades de Clasificación

Analicemos los documentos de ejemplo de la tabla 4.10 que han sido clasificados incorrectamente:

- En el primer documento se observa que en el tratamiento de negaciones deberíamos tener en cuenta otros términos que indiquen negación como por ejemplo: “nunca”, “tampoco”, “nada”.
- En comentarios demasiado cortos si los términos no son representativos no hay suficiente información para la clasificación. Por ejemplo, en el segundo documento, Turney clasifica correctamente porque si bien el término “impresentable” no aporta suficiente información, aparece en el corpus junto con términos que sí son representativos de la clase de comentarios negativos.
- El manejo de ironías o sarcasmo es una dificultad que aún no ha sido resuelta en tareas de procesamiento de lenguaje natural por lo que este tema queda pendiente de investigación.

- Para clasificar comentarios en otros idiomas podría realizarse previamente una tarea de procesamiento que asigne el idioma a los comentarios y luego entrenar el clasificador utilizando un conjunto de datos apropiado. En el caso del corpus que estamos analizando la cantidad de comentarios en otros idiomas no resulta representativa por lo que esta tarea no fue necesaria.
- En el ejemplo número cinco el único término representativo de la clase de comentarios negativos está mal escrito (“*pero*” en lugar de “*peor*”) por lo que la clasificación es incorrecta en todos los modelos.
- En el comentario número seis hay términos muy polares de la clase positivo (“*excelente*”, “*mejor*”) y los sentimientos negativos se expresan indirectamente, es decir, se deducen a partir del conocimiento popular en el contexto de gastronomía y servicios y no utilizando términos representativos de la clase de comentarios negativos.
- En el comentario número siete vemos que al no utilizar preprocesamientos de lematización o stemming, el término “*excelentísima*” (forma flexionada de la palabra “*excelente*”) no resulta representativo de la clase de comentarios positivos.

Capítulo 5

Conclusiones y Trabajo Futuro

En este trabajo investigamos las técnicas que se utilizan actualmente en tareas de análisis de sentimientos sobre textos no estructurados y en idioma español; presentamos una introducción al concepto de modelado de lenguaje que nos permitió comprender las bases del procesamiento de lenguaje natural; estudiamos las distintas técnicas de preprocesamiento y clasificación de textos en base a información subjetiva; y presentamos métricas y distintos métodos de evaluación para analizar la performance de los modelos en estudio.

Para validar nuestra investigación implementamos una herramienta de clasificación subjetiva de textos que provee todas las técnicas estudiadas adaptadas al idioma español, luego diseñamos una variedad de experiencias que nos permitieron evaluar el comportamiento de los distintos modelos ante la variación de determinados parámetros y realizamos un análisis de performance de todos los modelos presentados a partir de la ejecución de estas experiencias en distintos escenarios y en base a las métricas provistas por la aplicación desarrollada para la evaluación de performance.

Las experiencias presentadas muestran el impacto en la performance de los clasificadores ante la variación de parámetros de entrada como transformaciones de textos, tipos de atributos extraídos, conjunto de datos de entrenamiento, conjunto de datos de prueba, tamaño de corpus utilizado y proporción de documentos de cada clase.

Hallamos que los modelos supervisados de Naïve Bayes, MaxEnt y SVM resultan muy apropiados para clasificar información subjetiva en idioma español alcanzando hasta un 95% de accuracy para los tamaños de corpus más grandes mientras que técnicas basadas en árboles de decisión no proveen buenos resultados. Además, adaptamos la técnica de clasificación no supervisada de Turney al idioma español y obtuvimos excelentes resultados considerando que no se requiere información previa ni un conjunto de datos de entrenamiento.

Mostramos que la aplicación de preprocesamientos influye considerablemente

en los resultados del clasificador y presentamos las mejoras de performance obtenidas en cada caso. En cuanto al tamaño de corpus obtuvimos que la máxima precisión se obtiene utilizando MaxEnt y SVM para corpus grandes y Naïve Bayes para corpus más pequeños. Así también, mostramos que si se entrena con un dominio muy distinto al de prueba o si la proporción de documentos de cada clase resulta muy desbalanceada, la performance se degrada considerablemente. En el caso de las pruebas *cross domain* hemos visto que en determinados casos conviene utilizar una técnica no supervisada de clasificación refinada para el dominio en cuestión en lugar de entrenar con un dominio distinto.

Por último, la investigación presentada y la herramienta desarrollada para esta tesis constituyen una base para la construcción de sistemas de análisis de sentimientos en idioma español sobre textos no estructurados permitiendo seleccionar la técnica más apropiada para el escenario en cuestión. Además, los corpus de datos contruidos para estas experiencias resultan un aporte como recurso lingüístico para ser utilizado en futuras investigaciones.

Para trabajos futuros proponemos investigar soluciones para resolver las dificultades de clasificación que mostramos en la etapa de experimentación de este trabajo; evaluar otros preprocesamientos de texto (por ejemplo, corrección ortográfica) y otros tipos de atributos específicos de opinión que permitan mejorar los resultados y generalizar los clasificadores; desarrollar un modelo basado en léxico de opinión utilizando como recurso el conocimiento adquirido de los conjuntos de datos etiquetados que se construyeron para esta tesis; y experimentar con otros tipos de tareas de análisis de sentimientos como análisis del grado de polaridad de emociones y clasificación individual de los distintos aspectos que generan opinión en un mismo documento.

Referencias

- Python 2.7.
<http://www.python.org/>.
- SciPy 0.10.0 - Open Source Library of Scientific Tools For Python.
<http://www.scipy.org/>.
- Alexander Clark, Chris Fox and Shalom Lappin. *The handbook of computational linguistics and natural language processing*. Wiley-Blackwell, 2010.
- Andrew Ng. *Machine Learning Course*. Stanford University, 2012.
- Arthur Samuel. *Some Studies in Machine Learning Using the Game of Checkers*. IBM Journal of Research and Development 3(3), 211–229, 1959.
- Bing Liu. *Sentiment Analysis and Subjectivity*. Department of Computer Science, University of Illinois at Chicago, 2010.
- David M. Blei, Andrew Y. Ng, Michael I. Jordan, and John Lafferty. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:2003, 2003.
- Bo Pang and Lilian Lee. *Opinion Mining and Sentiment Analysis*. Department of Computer Science, Cornell University, 2008.
- Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. 2005.
- Bo Pang, Lillian Lee and Shivakumar Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. 2002.
- Brendan O'Connor, Ramnath Balasubramanyan, Bryan R. Routledge and Noah A. Smith. *From Tweets to Polls: Linking Text Sentiment to Public Opinion Time Series*. School of Computer Science, Carnegie Mellon University, 2010.
- Samuel Brody and Noemie Elhadad. An unsupervised aspect-sentiment model for online reviews. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 804–812. Association for Computational Linguistics, 2010.
- Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, 1999.
- Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

- Christopher Potts. *Sentiment Analysis Symposium, San Francisco*. 2011.
- C. Cortes and V. Vapnik. Support-vector network. *Machine Learning*, 20: 273–297, 1995.
- Fermín L. Cruz, Jose A. Troyano, Fernando Enriquez, and Javier Ortega. Clasificación de documentos basada en la opinión: experimentos con un corpus de críticas de cine en español. *Procesamiento del Lenguaje Natural*, 41, 2008.
- Daniel Jurafsky and Christopher D. Manning. *Natural Language Processing Course*. Stanford University, 2012.
- Daniel Jurafsky and James H. Martin. *Speech And Language Processing. An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. 2nd Edition. Prentice Hall, 2009.
- Luciana Dubiau and Juan M Ale. Análisis de sentimientos sobre un corpus en español: Experimentación con un caso de estudio. 42° *JAIIO ASAI*, 2013. ISSN 1850-2776.
- Google. Google n-grams dataset, 2008.
<http://books.google.com/ngrams>.
- Hal Daumé III. Notes on CG and LM-BFGS Optimization of Logistic Regression. 2004.
<http://www.umiacs.umd.edu/~hal/megam/>.
- Harry Zhang. *The Optimality of Naive Bayes*. University of New Brunswick, 2004.
- Vasileios Hatzivassiloglou and Kathleen R. McKeown. Predicting the semantic orientation of adjectives. pages 174–181, 1997.
- Irina Rish. *An empirical study of the naive Bayes classifier*. IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence, 2001.
- Jacob Perkins. *Python Text Processing with NLTK 2.0 Cookbook*. Packt Publishing Ltd, 2010.
- T. Joachims. *Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning*. B. Schölkopf and C. Burges and A. Smola, MIT-Press, 1999.
- Klaus R. Scherer. *Emotion as a Multicomponent Process: A model and some cross-cultural data*. 1984.
- Bin Lu, Myle Ott, Claire Cardie, and Benjamin K. Tsou. Multiaspect sentiment analysis with topic models. IEEE Computer Society, 2011.
- Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, Ian H. Witten. *The WEKA Data Mining Software*. SIGKDD Explorations, 2009.
<http://www.cs.waikato.ac.nz/ml/weka/>.

- Andrew Y. Ng and Michael I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes, 2001.
- Lluís Padró and Evgeny Stanilovsky. Freeling 3.0: Towards wider multilinguality. In *Proceedings of the Language Resources and Evaluation Conference (LREC 2012)*, Istanbul, Turkey, 2012. ELRA.
<http://nlp.lsi.upc.edu/freeling>.
- F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python . *Journal of Machine Learning Research*, 12: 2825–2830, 2011.
<http://scikit-learn.org/>.
- Pedro Domingos, Michael Pazzani. *On the Optimality of the Simple Bayesian Classifier under Zero-One Loss*, *Machine Learning*. Kluwer Academic Publishers, 1997.
- Peter E. Brown and Vincent J. Della Pietra. *Class-Based n-gram Models of Natural Language*. IBM Thomas J. Watson Research Center, 1992.
- Peter Turney. *Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews*. Institute for Information Technology, National Research Council of Canada, 2002.
- Ross Quinlan. Induction of decision trees. *Machine Learning*, 1986.
- Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5:101–141, 2004.
- Robert Plutchik. *Emotions and Life*. American Psychological Association, 2002.
- Grigori Sidorov, Sabino Miranda-Jiménez, Francisco Viveros-Jiménez, Alexander Gelbukh, Noé Castro-Sánchez, Francisco Velásquez, Ismael Díaz-Rangel, Sergio Suárez-Guerra, Alejandro Treviño, and Juan Gordon. Empirical Study of Opinion Mining in Spanish Tweets. *LNAI*, pages 7629–7630, 2012.
- Benjamin Snyder and Regina Barzilay. Multiple aspect ranking using the good grief algorithm. 2007.
- Stanley F. Chen and Joshua Goodman. *An Empirical Study of Smoothing Techniques for Language Modeling*. Center of Research in Computing Technology, Harvard University, Cambridge, Massachusetts, 1998.
- Steven Bird, Edward Loper and Ewan Klein. *Natural Language Processing with Python*. O’Reilly Media Inc, 2009.
<http://nltk.org>.
- Ivan Titov and Ryan McDonald. Modeling online reviews with multi-grain topic models. In *Proceedings of the 17th international conference on World Wide Web*, pages 111–120. ACM, 2008. ISBN 978-1-60558-085-2.
- Kimitaka Tsutsumi, Kazutaka Shimada, and Tsutomu Endo. Movie review classification based on a multiple classifier. 2007.

Anexo A

Código Fuente, Instalación y Ejecución del Software Desarrollado

En este anexo describiremos como ejecutar el clasificador de sentimientos desarrollado en esta tesis, veremos cómo parametrizar la aplicación y explicaremos los resultados.

A.1 Código Fuente

El código fuente está disponible para su descarga en el siguiente repositorio:

https://github.com/ldubiau/sentiment_classifier

A.2 Instalación de Software

La aplicación requiere Python 2.7 y como se especifica en el archivo *setup.py* del código fuente, es necesario instalar las siguientes dependencias:

```
httplib2 (0.7.6)
beautifulsoup4 (4.1.3)
nltk (2.0.3)
numpy (1.6.2)
PrettyTable (0.6.1)
scipy (0.10.0)
scikit-learn (0.12.1)
svmlight (0.4)
```

Y las siguientes herramientas externas:

```
megam.opt (0.91)
freeling (3.0)
weka.jar (3.6.9)
svmlight (6.02)
```

A.3 Ejecución de la Aplicación

La aplicación desarrollada se ejecuta según la siguiente especificación:

```
> python sentiment_classifier.py -h

usage: sentiment_classifier.py [-h]
                               (-nb |
                                -weka {maxent,svm,nb,tree} |
                                -megam |
                                -svmlight |
                                -sklearn {maxent,svm,nb,tree} |
                                -turney)
                               [-f F]
                               [-fn FN]
                               [-s S]
                               [-od]
                               [-pp PP]
                               [-u]
                               [-wf]
                               [-bi]
                               [-adj]
                               [-sw]
                               [-wl WL]
                               [-dc]
                               [-neg]
                               [-stem]
                               [-lc]
                               [-punct]
                               [-acc]
                               [-lemma]
                               [-allprepro]
```

Sentiment Classification Tool

optional arguments:

-h, --help	show this help message and exit
-nb	Naive-bayes classification
-weka {maxent,svm,nb,tree}	Classification using WEKA API
-megam	MaxEnt classification using MEGAM algorithm
-svmlight	SVM classification using SVMLight
-sklearn {maxent,svm,nb,tree}	Classification using sci-kit learn API
-turney	Classification using Turney algorithm
-f F	Number of folds for supervised algorithms using k-fold cross validation. If this parameter is not provided then hold-out validation is performed.
-fn FN	Fold number for supervised algorithms using k-fold

ANEXO A. CÓDIGO FUENTE, INSTALACIÓN Y EJECUCIÓN DEL
SOFTWARE DESARROLLADO

	cross validation
-s S	Corpus size
-od	Out of domain testing
-pp PP	Proportion of positive comments for unbalanced experiences
-u	Use top training unigrams as feature extractor
-wf	Use top training unigrams frequency as feature extractor
-bi	Use top training bigrams as feature extractor
-adj	Filter words to use just adjectives
-sw	Remove stop words
-wl WL	Filter words by minimum length
-dc	Remove duplicated characters
-neg	Preprocess negations
-stem	Use stemmed words
-lc	Transform chars to lower case
-punct	Remove punctuation marks
-acc	Remove spanish accents
-lemma	Use lemmatized words
-allprepro	Apply all preprocessors

Ejemplos de Ejecución

```
> sentiment_classifier.py -nb -f 5 -s 2000 -u -bi -sw -wl 3 -neg -punct -acc -lc
```

Parámetro	Valor
Algoritmo de Clasificación	Naïve Bayes
Algoritmo de Validación	5-fold Cross Validation
Tamaño de Corpus	2000 documentos
Features	Presencia de Unigramas + Presencia de Bigramas
Preprocesamientos	Eliminación de stopwords, filtrado de palabras de menos de 3 letras tratamiento de negaciones eliminación de signos de puntuación reemplazo de acentos, transformación a minúscula.

Tabla A.1: Ejemplo de Ejecución 1

```
> sentiment_classifier.py -megam -s 600 -u -adj -allprepro
```

Parámetro	Valor
Algoritmo de Clasificación	MaxEnt + Megam para cálculo de pesos
Algoritmo de Validación	Hold-out
Tamaño de Corpus	600
Features	Presencia de Adjetivos
Preprocesamientos	Todos

Tabla A.2: Ejemplo de Ejecución 2

A.4 Formato de Resultados

En la figura A.1 se observan los resultados de ejecución de la herramienta de análisis de sentimientos desarrollada.

Fold	tp(p)	cases(p)	prec(p)	rec(p)	acc(p)	f1(p)
0	768	800	0.932	0.960	0.960	0.946
1	767	800	0.916	0.959	0.959	0.937
2	769	800	0.890	0.961	0.961	0.924
3	766	800	0.894	0.958	0.958	0.925
4	764	800	0.906	0.955	0.955	0.930
Total	3834	4000	0.907	0.959	0.959	0.932

Fold	tp(n)	cases(n)	prec(n)	rec(n)	acc(n)	f1(n)
0	744	800	0.959	0.930	0.930	0.944
1	730	800	0.957	0.912	0.912	0.934
2	705	800	0.958	0.881	0.881	0.918
3	709	800	0.954	0.886	0.886	0.919
4	721	800	0.952	0.901	0.901	0.926
Total	3609	4000	0.956	0.902	0.902	0.928

Fold	acc(avg)	f1(avg)
0	0.945	0.945
1	0.936	0.936
2	0.921	0.921
3	0.922	0.922
4	0.928	0.928
Total	0.930	0.930

Figura A.1: Resultados de Ejecución

Siendo,

Métrica	Descripción
$fold$	número de fold
$t_p(p)$	cantidad de casos verdaderos positivos de la clase “positivo” (comentarios positivos que fueron clasificados correctamente).
$cases(p)$	cantidad de casos totales de la clase “positivo”.
$prec(p)$	precisión para la la clase “positivo”.
$rec(p)$	recall para la la clase “positivo”.
$acc(p)$	accuracy para la la clase “positivo”.
$F1(p)$	F1-measure para la la clase “positivo”.
$t_p(n)$	cantidad de casos verdaderos positivos de la clase “negativo” (comentarios negativos que fueron clasificados correctamente).
$cases(n)$	cantidad de casos totales de la clase “negativo”.
$prec(n)$	precisión para la la clase “negativo”.
$rec(n)$	recall para la la clase “negativo”.
$acc(n)$	accuracy para la la clase “negativo”.
$F1(n)$	F1-measure para la la clase “negativo”.
$acc(avg)$	accuracy promedio utilizando macro-averaging.
$F1(avg)$	F1-measure promedio utilizando macro-averaging.

Tabla A.3: Resultados de Ejecución: Métricas

Además, en el caso de clasificadores supervisados, la herramienta imprime los features más informativos del conjunto de datos de entrenamiento.

Anexo B

Tablas Completas de Resultados de Clasificación

En este anexo se exponen las tablas de resultados completas de todas las experiencias realizadas en esta tesis a partir de los cuales obtuvimos las gráficas y realizamos el correspondiente análisis en la sección 4.2. Las métricas utilizadas en este capítulo se basan en lo explicado en la sección 2.3.14.

B.1 Clasificación Supervisada para Corpus Balanceado y un Único Dominio

B.1.1 Naïve Bayes

algoritmo: Naïve Bayes

features: Presencia de Unigramas

corpus: Guía Óleo (balanceado)

método de validación: 5-fold cross validation

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	223	250	0.823	0.892	0.892	0.856	202	250	0.882	0.808	0.808	0.843	0.850	0.850
1000	477	500	0.863	0.954	0.954	0.906	424	500	0.949	0.848	0.848	0.895	0.901	0.901
2000	955	1000	0.895	0.955	0.955	0.924	888	1000	0.952	0.888	0.888	0.919	0.921	0.921
4000	1918	2000	0.912	0.959	0.959	0.935	1815	2000	0.957	0.907	0.907	0.931	0.933	0.933
6000	2883	3000	0.919	0.961	0.961	0.940	2747	3000	0.959	0.916	0.916	0.937	0.938	0.938
8000	3826	4000	0.909	0.957	0.957	0.932	3615	4000	0.954	0.904	0.904	0.928	0.930	0.930
10000	4783	5000	0.908	0.957	0.957	0.932	4517	5000	0.954	0.903	0.903	0.928	0.930	0.930
14000	6674	7000	0.904	0.953	0.953	0.928	6294	7000	0.951	0.899	0.899	0.924	0.926	0.926
18000	8592	9000	0.901	0.955	0.955	0.927	8054	9000	0.952	0.895	0.895	0.922	0.925	0.925
22000	10490	11000	0.901	0.954	0.954	0.926	9844	11000	0.951	0.895	0.895	0.922	0.924	0.924

Tabla B.1: Resultados de Clasificación utilizando el Algoritmo Naïve Bayes y Presencia de Unigramas como Features.

ANEXO B. TABLAS COMPLETAS DE RESULTADOS DE
CLASIFICACIÓN

algoritmo: Naïve Bayes
features: Frecuencia de Unigramas
corpus: Guía Óleo (balanceado)
método de validación: 5-fold cross validation
preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	225	250	0.815	0.900	0.900	0.856	199	250	0.888	0.796	0.796	0.840	0.848	0.848
1000	473	500	0.854	0.946	0.946	0.898	419	500	0.939	0.838	0.838	0.886	0.892	0.892
2000	957	1000	0.888	0.957	0.957	0.921	879	1000	0.953	0.879	0.879	0.915	0.918	0.918
4000	1919	2000	0.907	0.960	0.960	0.932	1803	2000	0.957	0.901	0.901	0.928	0.930	0.930
6000	2887	3000	0.915	0.962	0.962	0.938	2733	3000	0.960	0.911	0.911	0.935	0.937	0.937
8000	3834	4000	0.907	0.959	0.959	0.932	3609	4000	0.956	0.902	0.902	0.928	0.930	0.930
10000	4792	5000	0.907	0.958	0.958	0.932	4511	5000	0.956	0.902	0.902	0.928	0.930	0.930
14000	6681	7000	0.902	0.954	0.954	0.928	6276	7000	0.952	0.897	0.897	0.923	0.925	0.925
18000	8604	9000	0.898	0.956	0.956	0.926	8025	9000	0.953	0.892	0.892	0.921	0.924	0.924
22000	10507	11000	0.900	0.955	0.955	0.927	9830	11000	0.952	0.894	0.894	0.922	0.924	0.924

Tabla B.2: Resultados de Clasificación utilizando el Algoritmo Naïve Bayes y Frecuencia de Unigramas como Features.

algoritmo: Naïve Bayes
features: Presencia de Bigramas
corpus: Guía Óleo (balanceado)
método de validación: 5-fold cross validation
preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	102	250	0.810	0.408	0.408	0.543	226	250	0.604	0.904	0.904	0.724	0.656	0.633
1000	283	500	0.797	0.566	0.566	0.662	428	500	0.664	0.856	0.856	0.748	0.711	0.705
2000	684	1000	0.817	0.684	0.684	0.745	847	1000	0.728	0.847	0.847	0.783	0.766	0.764
4000	1570	2000	0.857	0.785	0.785	0.820	1739	2000	0.802	0.870	0.870	0.834	0.827	0.827
6000	2459	3000	0.877	0.820	0.820	0.847	2656	3000	0.831	0.885	0.885	0.857	0.853	0.852
8000	3365	4000	0.885	0.841	0.841	0.862	3562	4000	0.849	0.890	0.890	0.869	0.866	0.866
10000	4271	5000	0.887	0.854	0.854	0.870	4454	5000	0.859	0.891	0.891	0.875	0.873	0.872
14000	6090	7000	0.878	0.870	0.870	0.874	6151	7000	0.871	0.879	0.879	0.875	0.874	0.874
18000	7692	9000	0.896	0.855	0.855	0.875	8104	9000	0.861	0.900	0.900	0.880	0.878	0.877
22000	9417	11000	0.899	0.856	0.856	0.877	9939	11000	0.863	0.904	0.904	0.883	0.880	0.880

Tabla B.3: Resultados de Clasificación utilizando el Algoritmo Naïve Bayes y Presencia de Bigramas como Features.

ANEXO B. TABLAS COMPLETAS DE RESULTADOS DE
CLASIFICACIÓN

algoritmo: Naïve Bayes

features: Presencia de Unigramas y Bigramas

corpus: Guía Óleo (balanceado)

método de validación: 5-fold cross validation

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	222	250	0.841	0.888	0.888	0.864	208	250	0.881	0.832	0.832	0.856	0.860	0.860
1000	473	500	0.873	0.946	0.946	0.908	431	500	0.941	0.862	0.862	0.900	0.904	0.904
2000	948	1000	0.898	0.948	0.948	0.922	892	1000	0.945	0.892	0.892	0.918	0.920	0.920
4000	1925	2000	0.921	0.963	0.963	0.942	1836	2000	0.961	0.918	0.918	0.939	0.940	0.940
6000	2879	3000	0.928	0.960	0.960	0.944	2778	3000	0.958	0.926	0.926	0.942	0.943	0.943
8000	3839	4000	0.921	0.960	0.960	0.940	3671	4000	0.958	0.918	0.918	0.937	0.939	0.939
10000	4823	5000	0.924	0.965	0.965	0.944	4602	5000	0.963	0.920	0.920	0.941	0.943	0.942
14000	6745	7000	0.919	0.964	0.964	0.941	6403	7000	0.962	0.915	0.915	0.938	0.939	0.939
18000	8678	9000	0.920	0.964	0.964	0.941	8242	9000	0.962	0.916	0.916	0.939	0.940	0.940
22000	10595	11000	0.919	0.963	0.963	0.940	10063	11000	0.961	0.915	0.915	0.937	0.939	0.939

Tabla B.4: Resultados de Clasificación utilizando el Algoritmo Naïve Bayes y Presencia de Unigramas y Bigramas como Features.

algoritmo: Naïve Bayes

features: Presencia de Adjetivos

corpus: Guía Óleo (balanceado)

método de validación: 5-fold cross validation

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	221	250	0.857	0.884	0.884	0.870	213	250	0.880	0.852	0.852	0.866	0.868	0.868
1000	453	500	0.876	0.906	0.906	0.891	436	500	0.903	0.872	0.872	0.887	0.889	0.889
2000	912	1000	0.882	0.912	0.912	0.897	878	1000	0.909	0.878	0.878	0.893	0.895	0.895
4000	1843	2000	0.902	0.921	0.921	0.911	1799	2000	0.920	0.899	0.899	0.910	0.910	0.910
6000	2756	3000	0.901	0.919	0.919	0.910	2696	3000	0.917	0.899	0.899	0.908	0.909	0.909
8000	3688	4000	0.896	0.922	0.922	0.909	3570	4000	0.920	0.892	0.892	0.906	0.907	0.907
10000	4622	5000	0.888	0.924	0.924	0.906	4419	5000	0.921	0.884	0.884	0.902	0.904	0.904
14000	6590	7000	0.873	0.941	0.941	0.906	6037	7000	0.936	0.862	0.862	0.898	0.902	0.902
18000	8520	9000	0.867	0.947	0.947	0.905	7693	9000	0.941	0.855	0.855	0.896	0.901	0.901
22000	10438	11000	0.872	0.949	0.949	0.909	9469	11000	0.944	0.861	0.861	0.900	0.905	0.905

Tabla B.5: Resultados de Clasificación utilizando el Algoritmo Naïve Bayes y Presencia de Adjetivos como Features.

ANEXO B. TABLAS COMPLETAS DE RESULTADOS DE
CLASIFICACIÓN

B.1.2 MaxEnt

algoritmo: MaxEnt

features: Presencia de Unigramas

corpus: Guía Óleo (balanceado)

método de validación: 5-fold cross validation

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	208	250	0.809	0.832	0.832	0.821	201	250	0.827	0.804	0.804	0.815	0.818	0.818
1000	458	500	0.867	0.916	0.916	0.891	430	500	0.911	0.860	0.860	0.885	0.888	0.888
2000	906	1000	0.916	0.906	0.906	0.911	917	1000	0.907	0.917	0.917	0.912	0.911	0.911
4000	1863	2000	0.932	0.931	0.931	0.932	1864	2000	0.932	0.932	0.932	0.932	0.932	0.932
6000	2854	3000	0.938	0.951	0.951	0.945	2811	3000	0.951	0.937	0.937	0.944	0.944	0.944
8000	3767	4000	0.925	0.942	0.942	0.933	3694	4000	0.941	0.923	0.923	0.932	0.933	0.933
10000	4679	5000	0.934	0.936	0.936	0.935	4672	5000	0.936	0.934	0.934	0.935	0.935	0.935
14000	6621	7000	0.935	0.946	0.946	0.940	6538	7000	0.945	0.934	0.934	0.940	0.940	0.940
18000	8597	9000	0.936	0.955	0.955	0.946	8416	9000	0.954	0.935	0.935	0.945	0.945	0.945
22000	10500	11000	0.936	0.955	0.955	0.945	10278	11000	0.954	0.934	0.934	0.944	0.944	0.944

Tabla B.6: Resultados de Clasificación utilizando el Algoritmo MaxEnt y Presencia de Unigramas como Features.

algoritmo: MaxEnt

features: Frecuencia de Unigramas

corpus: Guía Óleo (balanceado)

método de validación: 5-fold cross validation

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	208	250	0.812	0.832	0.832	0.822	202	250	0.828	0.808	0.808	0.818	0.820	0.820
1000	454	500	0.868	0.908	0.908	0.888	431	500	0.904	0.862	0.862	0.882	0.885	0.885
2000	924	1000	0.915	0.924	0.924	0.919	914	1000	0.923	0.914	0.914	0.919	0.919	0.919
4000	1877	2000	0.927	0.939	0.939	0.933	1852	2000	0.938	0.926	0.926	0.932	0.932	0.932
6000	2795	3000	0.932	0.932	0.932	0.932	2797	3000	0.932	0.932	0.932	0.932	0.932	0.932
8000	3802	4000	0.921	0.951	0.951	0.936	3675	4000	0.949	0.919	0.919	0.934	0.935	0.935
10000	4742	5000	0.931	0.948	0.948	0.940	4648	5000	0.947	0.930	0.930	0.938	0.939	0.939
14000	6713	7000	0.932	0.959	0.959	0.945	6511	7000	0.958	0.930	0.930	0.944	0.945	0.945
18000	8509	9000	0.930	0.945	0.945	0.938	8358	9000	0.945	0.929	0.929	0.937	0.937	0.937
22000	10407	11000	0.940	0.946	0.946	0.943	10334	11000	0.946	0.939	0.939	0.943	0.943	0.943

Tabla B.7: Resultados de Clasificación utilizando el Algoritmo MaxEnt y Frecuencia de Unigramas como Features.

ANEXO B. TABLAS COMPLETAS DE RESULTADOS DE
CLASIFICACIÓN

algoritmo: MaxEnt
features: Presencia de Bigramas
corpus: Guía Óleo (balanceado)
método de validación: 5-fold cross validation
preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	97	250	0.789	0.388	0.388	0.520	224	250	0.594	0.896	0.896	0.715	0.642	0.617
1000	278	500	0.806	0.556	0.556	0.658	433	500	0.661	0.866	0.866	0.750	0.711	0.704
2000	648	1000	0.814	0.648	0.648	0.722	852	1000	0.708	0.852	0.852	0.773	0.750	0.747
4000	1500	2000	0.851	0.750	0.750	0.797	1737	2000	0.776	0.869	0.869	0.820	0.809	0.809
6000	2272	3000	0.858	0.757	0.757	0.805	2625	3000	0.783	0.875	0.875	0.826	0.816	0.816
8000	2935	4000	0.868	0.734	0.734	0.795	3554	4000	0.769	0.888	0.888	0.825	0.811	0.810
10000	3629	5000	0.877	0.726	0.726	0.794	4493	5000	0.766	0.899	0.899	0.827	0.812	0.811
14000	5417	7000	0.848	0.774	0.774	0.809	6031	7000	0.792	0.862	0.862	0.825	0.818	0.817
18000	6820	9000	0.859	0.758	0.758	0.805	7884	9000	0.783	0.876	0.876	0.827	0.817	0.816
22000	8423	11000	0.877	0.766	0.766	0.818	9823	11000	0.792	0.893	0.893	0.840	0.829	0.829h

Tabla B.8: Resultados de Clasificación utilizando el Algoritmo MaxEnt y Presencia de Bigramas como Features.

algoritmo: MaxEnt
features: Presencia de Unigramas y Bigramas
corpus: Guía Óleo (balanceado)
método de validación: 5-fold cross validation
preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	209	250	0.820	0.836	0.836	0.828	204	250	0.833	0.816	0.816	0.824	0.826	0.826
1000	459	500	0.876	0.918	0.918	0.896	435	500	0.914	0.870	0.870	0.891	0.894	0.894
2000	919	1000	0.918	0.919	0.919	0.919	918	1000	0.919	0.918	0.918	0.918	0.919	0.918
4000	1875	2000	0.933	0.938	0.938	0.935	1866	2000	0.937	0.933	0.933	0.935	0.935	0.935
6000	2833	3000	0.934	0.944	0.944	0.939	2800	3000	0.944	0.933	0.933	0.938	0.939	0.939
8000	3786	4000	0.932	0.947	0.947	0.939	3722	4000	0.946	0.930	0.930	0.938	0.939	0.938
10000	4748	5000	0.936	0.950	0.950	0.943	4673	5000	0.949	0.935	0.935	0.942	0.942	0.942
14000	6631	7000	0.942	0.947	0.947	0.945	6595	7000	0.947	0.942	0.942	0.945	0.945	0.945
18000	8600	9000	0.938	0.956	0.956	0.947	8428	9000	0.955	0.936	0.936	0.945	0.946	0.946
22000	10553	11000	0.941	0.959	0.959	0.950	10343	11000	0.959	0.940	0.940	0.949	0.950	0.950

Tabla B.9: Resultados de Clasificación utilizando el Algoritmo MaxEnt y Presencia de Unigramas y Bigramas como Features.

ANEXO B. TABLAS COMPLETAS DE RESULTADOS DE
CLASIFICACIÓN

algoritmo: MaxEnt
features: Presencia de Adjetivos
corpus: Guía Óleo (balanceado)
método de validación: 5-fold cross validation
preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	214	250	0.863	0.856	0.856	0.859	216	250	0.857	0.864	0.864	0.861	0.860	0.860
1000	439	500	0.882	0.878	0.878	0.880	441	500	0.878	0.882	0.882	0.880	0.880	0.880
2000	891	1000	0.892	0.891	0.891	0.891	892	1000	0.891	0.892	0.892	0.892	0.891	0.891
4000	1809	2000	0.896	0.904	0.904	0.900	1791	2000	0.904	0.895	0.895	0.900	0.900	0.900
6000	2601	3000	0.893	0.867	0.867	0.880	2687	3000	0.871	0.896	0.896	0.883	0.881	0.881
8000	3541	4000	0.896	0.885	0.885	0.890	3588	4000	0.887	0.897	0.897	0.892	0.891	0.891
10000	4450	5000	0.892	0.890	0.890	0.891	4464	5000	0.890	0.893	0.893	0.892	0.891	0.891
14000	6304	7000	0.878	0.901	0.901	0.889	6127	7000	0.898	0.875	0.875	0.886	0.888	0.888
18000	8040	9000	0.887	0.893	0.893	0.890	7978	9000	0.893	0.886	0.886	0.890	0.890	0.890
22000	9671	11000	0.884	0.879	0.879	0.881	9728	11000	0.880	0.884	0.884	0.882	0.882	0.882

Tabla B.10: Resultados de Clasificación utilizando el Algoritmo MaxEnt y Presencia de Adjetivos como Features.

B.1.3 SVM

algoritmo: SVM
features: Presencia de Unigramas
corpus: Guía Óleo (balanceado)
método de validación: 5-fold cross validation
preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	210	250	0.784	0.840	0.840	0.811	192	250	0.828	0.768	0.768	0.797	0.804	0.804
1000	453	500	0.856	0.906	0.906	0.880	424	500	0.900	0.848	0.848	0.873	0.877	0.877
2000	912	1000	0.898	0.912	0.912	0.905	896	1000	0.911	0.896	0.896	0.903	0.904	0.904
4000	1874	2000	0.926	0.937	0.937	0.932	1851	2000	0.936	0.925	0.925	0.931	0.931	0.931
6000	2830	3000	0.922	0.943	0.943	0.932	2760	3000	0.942	0.920	0.920	0.931	0.932	0.932
8000	3757	4000	0.924	0.939	0.939	0.932	3692	4000	0.938	0.923	0.923	0.931	0.931	0.931
10000	4742	5000	0.931	0.948	0.948	0.939	4647	5000	0.947	0.929	0.929	0.938	0.939	0.939
14000	6615	7000	0.934	0.945	0.945	0.940	6534	7000	0.944	0.933	0.933	0.939	0.939	0.939
18000	8537	9000	0.934	0.949	0.949	0.941	8400	9000	0.948	0.933	0.933	0.940	0.941	0.941
22000	10453	11000	0.938	0.950	0.950	0.944	10304	11000	0.950	0.937	0.937	0.943	0.944	0.943

Tabla B.11: Resultados de Clasificación utilizando el Algoritmo SVM y Presencia de Unigramas como Features.

ANEXO B. TABLAS COMPLETAS DE RESULTADOS DE
CLASIFICACIÓN

algoritmo: SVM
features: Frecuencia de Unigramas
corpus: Guía Óleo (balanceado)
método de validación: 5-fold cross validation
preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	205	250	0.782	0.820	0.820	0.801	193	250	0.811	0.772	0.772	0.791	0.796	0.796
1000	447	500	0.853	0.894	0.894	0.873	423	500	0.889	0.846	0.846	0.867	0.870	0.870
2000	907	1000	0.899	0.907	0.907	0.903	898	1000	0.906	0.898	0.898	0.902	0.903	0.902
4000	1877	2000	0.923	0.939	0.939	0.931	1843	2000	0.937	0.921	0.921	0.929	0.930	0.930
6000	2826	3000	0.920	0.942	0.942	0.931	2754	3000	0.941	0.918	0.918	0.929	0.930	0.930
8000	3753	4000	0.924	0.938	0.938	0.931	3691	4000	0.937	0.923	0.923	0.930	0.930	0.930
10000	4736	5000	0.931	0.947	0.947	0.939	4651	5000	0.946	0.930	0.930	0.938	0.939	0.939
14000	6619	7000	0.931	0.946	0.946	0.938	6512	7000	0.945	0.930	0.930	0.937	0.938	0.938
18000	8537	9000	0.934	0.949	0.949	0.941	8400	9000	0.948	0.933	0.933	0.940	0.941	0.941
22000	10467	11000	0.940	0.952	0.952	0.946	10328	11000	0.951	0.939	0.939	0.945	0.945	0.945

Tabla B.12: Resultados de Clasificación utilizando el Algoritmo SVM y Frecuencia de Unigramas como Features.

algoritmo: SVM
features: Presencia de Bigramas
corpus: Guía Óleo (balanceado)
método de validación: 5-fold cross validation
preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	96	250	0.780	0.384	0.384	0.515	223	250	0.592	0.892	0.892	0.711	0.638	0.613
1000	274	500	0.806	0.548	0.548	0.652	434	500	0.658	0.868	0.868	0.748	0.708	0.700
2000	648	1000	0.809	0.648	0.648	0.720	847	1000	0.706	0.847	0.847	0.770	0.748	0.745
4000	1499	2000	0.842	0.750	0.750	0.793	1718	2000	0.774	0.859	0.859	0.814	0.804	0.804
6000	2377	3000	0.850	0.792	0.792	0.820	2582	3000	0.806	0.861	0.861	0.832	0.827	0.826
8000	3237	4000	0.858	0.809	0.809	0.833	3466	4000	0.820	0.867	0.867	0.842	0.838	0.838
10000	4072	5000	0.862	0.814	0.814	0.837	4346	5000	0.824	0.869	0.869	0.846	0.842	0.842
14000	5808	7000	0.854	0.830	0.830	0.842	6006	7000	0.834	0.858	0.858	0.846	0.844	0.844
18000	7525	9000	0.860	0.836	0.836	0.848	7778	9000	0.841	0.864	0.864	0.852	0.850	0.850
22000	9378	11000	0.856	0.853	0.853	0.854	9427	11000	0.853	0.857	0.857	0.855	0.855	0.855

Tabla B.13: Resultados de Clasificación utilizando el Algoritmo SVM y Presencia de Bigramas como Features.

ANEXO B. TABLAS COMPLETAS DE RESULTADOS DE
CLASIFICACIÓN

algoritmo: SVM

features: Presencia de Unigramas y Bigramas

corpus: Guía Óleo (balanceado)

método de validación: 5-fold cross validation

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	201	250	0.798	0.804	0.804	0.801	199	250	0.802	0.796	0.796	0.799	0.800	0.800
1000	447	500	0.861	0.894	0.894	0.877	428	500	0.890	0.856	0.856	0.873	0.875	0.875
2000	903	1000	0.896	0.903	0.903	0.899	895	1000	0.902	0.895	0.895	0.899	0.899	0.899
4000	1882	2000	0.929	0.941	0.941	0.935	1857	2000	0.940	0.928	0.928	0.934	0.935	0.935
6000	2823	3000	0.922	0.941	0.941	0.931	2761	3000	0.940	0.920	0.920	0.930	0.931	0.931
8000	3767	4000	0.929	0.942	0.942	0.935	3710	4000	0.941	0.927	0.927	0.934	0.935	0.935
10000	4741	5000	0.937	0.948	0.948	0.942	4680	5000	0.948	0.936	0.936	0.942	0.942	0.942
14000	6644	7000	0.935	0.949	0.949	0.942	6539	7000	0.948	0.934	0.934	0.941	0.942	0.942
18000	8526	9000	0.937	0.947	0.947	0.942	8425	9000	0.947	0.936	0.936	0.941	0.942	0.942
22000	10475	11000	0.941	0.952	0.952	0.947	10345	11000	0.952	0.940	0.940	0.946	0.946	0.946

Tabla B.14: Resultados de Clasificación utilizando el Algoritmo SVM y Presencia de Unigramas y Bigramas como Features.

algoritmo: SVM

features: Presencia de Adjetivos

corpus: Guía Óleo (balanceado)

método de validación: 5-fold cross validation

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	210	250	0.847	0.840	0.840	0.843	212	250	0.841	0.848	0.848	0.845	0.844	0.844
1000	436	500	0.893	0.872	0.872	0.883	448	500	0.875	0.896	0.896	0.885	0.884	0.884
2000	888	1000	0.879	0.888	0.888	0.884	878	1000	0.887	0.878	0.878	0.882	0.883	0.883
4000	1801	2000	0.896	0.900	0.900	0.898	1791	2000	0.900	0.895	0.895	0.898	0.898	0.898
6000	2697	3000	0.898	0.899	0.899	0.899	2695	3000	0.899	0.898	0.898	0.899	0.899	0.899
8000	3599	4000	0.891	0.900	0.900	0.895	3561	4000	0.899	0.890	0.890	0.894	0.895	0.895
10000	4494	5000	0.893	0.899	0.899	0.896	4462	5000	0.898	0.892	0.892	0.895	0.896	0.896
14000	6325	7000	0.893	0.904	0.904	0.898	6244	7000	0.902	0.892	0.892	0.897	0.898	0.898
18000	8140	9000	0.893	0.904	0.904	0.899	8024	9000	0.903	0.892	0.892	0.897	0.898	0.898
22000	9968	11000	0.895	0.906	0.906	0.901	9836	11000	0.905	0.894	0.894	0.900	0.900	0.900

Tabla B.15: Resultados de Clasificación utilizando el Algoritmo SVM y Presencia de Adjetivos como Features.

ANEXO B. TABLAS COMPLETAS DE RESULTADOS DE
CLASIFICACIÓN

B.1.4 Decision Trees

algoritmo: DecisionTrees
features: Presencia de Unigramas
corpus: Guía Óleo (balanceado)
método de validación: 5-fold cross validation
preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	192	250	0.747	0.768	0.768	0.757	185	250	0.761	0.740	0.740	0.751	0.754	0.754
1000	420	500	0.750	0.840	0.840	0.792	360	500	0.818	0.720	0.720	0.766	0.780	0.779
2000	866	1000	0.797	0.866	0.866	0.830	780	1000	0.853	0.780	0.780	0.815	0.823	0.823
4000	1743	2000	0.814	0.872	0.872	0.842	1603	2000	0.862	0.801	0.801	0.831	0.837	0.836
6000	2603	3000	0.817	0.868	0.868	0.842	2418	3000	0.859	0.806	0.806	0.832	0.837	0.837
8000	3492	4000	0.827	0.873	0.873	0.849	3268	4000	0.865	0.817	0.817	0.841	0.845	0.845
10000	4434	5000	0.850	0.887	0.887	0.868	4220	5000	0.882	0.844	0.844	0.862	0.865	0.865
14000	6211	7000	0.866	0.887	0.887	0.877	6041	7000	0.884	0.863	0.863	0.874	0.875	0.875
18000	8004	9000	0.871	0.889	0.889	0.880	7811	9000	0.887	0.868	0.868	0.877	0.879	0.879
22000	9830	11000	0.873	0.894	0.894	0.883	9564	11000	0.891	0.869	0.869	0.880	0.882	0.882

Tabla B.16: Resultados de Clasificación utilizando el Algoritmo DecisionTrees y Presencia de Unigramas como Features.

algoritmo: DecisionTrees
features: Frecuencia de Unigramas
corpus: Guía Óleo (balanceado)
método de validación: 5-fold cross validation
preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	193	250	0.739	0.772	0.772	0.755	182	250	0.762	0.728	0.728	0.744	0.750	0.750
1000	426	500	0.759	0.852	0.852	0.803	365	500	0.831	0.730	0.730	0.777	0.791	0.790
2000	868	1000	0.799	0.868	0.868	0.832	782	1000	0.856	0.782	0.782	0.817	0.825	0.825
4000	1738	2000	0.815	0.869	0.869	0.841	1605	2000	0.860	0.802	0.802	0.830	0.836	0.836
6000	2593	3000	0.825	0.864	0.864	0.844	2450	3000	0.858	0.817	0.817	0.837	0.841	0.840
8000	3501	4000	0.819	0.875	0.875	0.846	3228	4000	0.866	0.807	0.807	0.836	0.841	0.841
10000	4435	5000	0.856	0.887	0.887	0.871	4254	5000	0.883	0.851	0.851	0.866	0.869	0.869
14000	6218	7000	0.864	0.888	0.888	0.876	6024	7000	0.885	0.861	0.861	0.873	0.874	0.874
18000	8016	9000	0.868	0.891	0.891	0.879	7778	9000	0.888	0.864	0.864	0.876	0.877	0.877
22000	9822	11000	0.871	0.893	0.893	0.882	9550	11000	0.890	0.868	0.868	0.879	0.881	0.881

Tabla B.17: Resultados de Clasificación utilizando el Algoritmo DecisionTrees y Frecuencia de Unigramas como Features.

ANEXO B. TABLAS COMPLETAS DE RESULTADOS DE
CLASIFICACIÓN

algoritmo: DecisionTrees
features: Presencia de Bigramas
corpus: Guía Óleo (balanceado)
método de validación: 5-fold cross validation
preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	96	250	0.738	0.384	0.384	0.505	216	250	0.584	0.864	0.864	0.697	0.624	0.601
1000	277	500	0.749	0.554	0.554	0.637	407	500	0.646	0.814	0.814	0.720	0.684	0.679
2000	635	1000	0.771	0.635	0.635	0.696	811	1000	0.690	0.811	0.811	0.745	0.723	0.721
4000	1468	2000	0.784	0.734	0.734	0.758	1596	2000	0.750	0.798	0.798	0.773	0.766	0.766
6000	2351	3000	0.789	0.784	0.784	0.787	2373	3000	0.785	0.791	0.791	0.788	0.787	0.787
8000	3166	4000	0.791	0.791	0.791	0.791	3165	4000	0.791	0.791	0.791	0.791	0.791	0.791
10000	4345	5000	0.744	0.869	0.869	0.802	3504	5000	0.843	0.701	0.701	0.765	0.785	0.783
14000	6335	7000	0.733	0.905	0.905	0.810	4698	7000	0.876	0.671	0.671	0.760	0.788	0.785
18000	8101	9000	0.749	0.900	0.900	0.818	6285	9000	0.875	0.698	0.698	0.777	0.799	0.797
22000	9885	11000	0.755	0.899	0.899	0.820	7784	11000	0.875	0.708	0.708	0.782	0.803	0.801

Tabla B.18: Resultados de Clasificación utilizando el Algoritmo DecisionTrees y Presencia de Bigramas como Features.

algoritmo: DecisionTrees
features: Presencia de Unigramas y Bigramas
corpus: Guía Óleo (balanceado)
método de validación: 5-fold cross validation
preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	194	250	0.735	0.776	0.776	0.755	180	250	0.763	0.720	0.720	0.741	0.748	0.748
1000	417	500	0.778	0.834	0.834	0.805	381	500	0.821	0.762	0.762	0.790	0.798	0.798
2000	862	1000	0.798	0.862	0.862	0.829	782	1000	0.850	0.782	0.782	0.815	0.822	0.822
4000	1742	2000	0.816	0.871	0.871	0.843	1607	2000	0.862	0.803	0.803	0.832	0.837	0.837
6000	2613	3000	0.828	0.871	0.871	0.849	2458	3000	0.864	0.819	0.819	0.841	0.845	0.845
8000	3513	4000	0.836	0.878	0.878	0.857	3311	4000	0.872	0.828	0.828	0.849	0.853	0.853
10000	4450	5000	0.853	0.890	0.890	0.871	4233	5000	0.885	0.847	0.847	0.865	0.868	0.868
14000	6225	7000	0.866	0.889	0.889	0.878	6037	7000	0.886	0.862	0.862	0.874	0.876	0.876
18000	8023	9000	0.871	0.891	0.891	0.881	7814	9000	0.889	0.868	0.868	0.878	0.880	0.880
22000	9843	11000	0.875	0.895	0.895	0.885	9596	11000	0.892	0.872	0.872	0.882	0.884	0.884

Tabla B.19: Resultados de Clasificación utilizando el Algoritmo DecisionTrees y Presencia de Unigramas y Bigramas como Features.

ANEXO B. TABLAS COMPLETAS DE RESULTADOS DE CLASIFICACIÓN

algoritmo: DecisionTrees
features: Presencia de Adjetivos
corpus: Guía Óleo (balanceado)
método de validación: 5-fold cross validation
preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	212	250	0.779	0.848	0.848	0.812	190	250	0.833	0.760	0.760	0.795	0.804	0.804
1000	415	500	0.782	0.830	0.830	0.805	384	500	0.819	0.768	0.768	0.793	0.799	0.799
2000	878	1000	0.808	0.878	0.878	0.842	792	1000	0.867	0.792	0.792	0.828	0.835	0.835
4000	1720	2000	0.825	0.860	0.860	0.842	1636	2000	0.854	0.818	0.818	0.836	0.839	0.839
6000	2602	3000	0.829	0.867	0.867	0.848	2463	3000	0.861	0.821	0.821	0.840	0.844	0.844
8000	3467	4000	0.827	0.867	0.867	0.846	3273	4000	0.860	0.818	0.818	0.839	0.843	0.842
10000	4315	5000	0.827	0.863	0.863	0.844	4095	5000	0.857	0.819	0.819	0.837	0.841	0.841
14000	6052	7000	0.833	0.865	0.865	0.849	5789	7000	0.859	0.827	0.827	0.843	0.846	0.84
18000	7785	9000	0.835	0.865	0.865	0.850	7462	9000	0.860	0.829	0.829	0.844	0.847	0.847
22000	9584	11000	0.840	0.871	0.871	0.855	9178	11000	0.866	0.834	0.834	0.850	0.853	0.853

Tabla B.20: Resultados de Clasificación utilizando el Algoritmo DecisionTrees y Presencia de Adjetivos como Features.

B.2 Clasificación No Supervisada para Corpus Balanceado

algoritmo: Turney adaptado al español
corpus: Guía Óleo (balanceado)
método de validación: hold-out
preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	42	50	0.824	0.840	0.840	0.832	41	50	0.837	0.820	0.820	0.828	0.830	0.830
1000	81	100	0.827	0.810	0.810	0.818	83	100	0.814	0.830	0.830	0.822	0.820	0.820
2000	163	200	0.819	0.815	0.815	0.817	164	200	0.816	0.820	0.820	0.818	0.817	0.817
4000	337	400	0.804	0.843	0.843	0.823	318	400	0.835	0.795	0.795	0.814	0.819	0.819
6000	520	600	0.837	0.867	0.867	0.852	499	600	0.862	0.832	0.832	0.846	0.849	0.849
8000	693	800	0.834	0.866	0.866	0.850	662	800	0.861	0.828	0.828	0.844	0.847	0.847
10000	864	1000	0.836	0.864	0.864	0.850	830	1000	0.859	0.830	0.830	0.844	0.847	0.847
14000	1203	1400	0.838	0.859	0.859	0.849	1168	1400	0.856	0.834	0.834	0.845	0.847	0.847
18000	1565	1800	0.834	0.869	0.869	0.851	1489	1800	0.864	0.827	0.827	0.845	0.848	0.848
22000	1891	2200	0.825	0.860	0.860	0.842	1799	2200	0.853	0.818	0.818	0.835	0.839	0.839

Tabla B.21: Resultados de Clasificación utilizando el Algoritmo de Turney adaptado al español.

B.3 Clasificación Supervisada para Corpus Balanceado con Cambio de Dominio

algoritmo: Naïve Bayes

features: Presencia de Unigramas

training set: Guía Óleo (balanceado), **test set:** Google Play

método de validación: 5-fold cross validation

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	232	250	0.695	0.928	0.928	0.795	148	250	0.892	0.592	0.592	0.712	0.760	0.753
1000	489	500	0.620	0.978	0.978	0.759	200	500	0.948	0.400	0.400	0.563	0.689	0.661
2000	981	1000	0.595	0.981	0.981	0.741	333	1000	0.946	0.333	0.333	0.493	0.657	0.617
4000	1953	2000	0.616	0.977	0.977	0.755	782	2000	0.943	0.391	0.391	0.553	0.684	0.654
6000	2950	3000	0.643	0.983	0.983	0.778	1364	3000	0.965	0.455	0.455	0.618	0.719	0.698
8000	3947	4000	0.616	0.987	0.987	0.759	1542	4000	0.967	0.386	0.386	0.551	0.686	0.655
10000	4936	5000	0.594	0.987	0.987	0.742	1630	5000	0.962	0.326	0.326	0.487	0.657	0.614
14000	6939	7000	0.573	0.991	0.991	0.727	1839	7000	0.968	0.263	0.263	0.413	0.627	0.570
18000	8897	9000	0.577	0.989	0.989	0.729	2477	9000	0.960	0.275	0.275	0.428	0.632	0.578
22000	10883	11000	0.579	0.989	0.989	0.731	3090	11000	0.964	0.281	0.281	0.435	0.635	0.583

Tabla B.22: Resultados de Clasificación Supervisada cambiando de Dominio y utilizando el Algoritmo Naïve Bayes y Presencia de Unigramas como Features.

algoritmo: MaxEnt

features: Presencia de Unigramas

training set: Guía Óleo (balanceado), **test set:** Google Play

método de validación: 5-fold cross validation

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	220	250	0.731	0.880	0.880	0.799	169	250	0.849	0.676	0.676	0.753	0.778	0.776
1000	448	500	0.750	0.896	0.896	0.817	351	500	0.871	0.702	0.702	0.777	0.799	0.797
2000	918	1000	0.694	0.918	0.918	0.791	596	1000	0.879	0.596	0.596	0.710	0.757	0.751
4000	1768	2000	0.795	0.884	0.884	0.837	1543	2000	0.869	0.771	0.771	0.817	0.828	0.827
6000	2728	3000	0.795	0.909	0.909	0.848	2295	3000	0.894	0.765	0.765	0.825	0.837	0.836
8000	3675	4000	0.754	0.919	0.919	0.828	2800	4000	0.896	0.700	0.700	0.786	0.809	0.807
10000	4709	5000	0.731	0.942	0.942	0.823	3269	5000	0.918	0.654	0.654	0.764	0.798	0.794
14000	6529	7000	0.734	0.933	0.933	0.822	4638	7000	0.908	0.663	0.663	0.766	0.798	0.794
18000	8512	9000	0.708	0.946	0.946	0.810	5482	9000	0.918	0.609	0.609	0.732	0.777	0.771
22000	10367	11000	0.696	0.942	0.942	0.801	6473	11000	0.911	0.588	0.588	0.715	0.765	0.758

Tabla B.23: Resultados de Clasificación Supervisada cambiando de Dominio y utilizando el Algoritmo MaxEnt y Presencia de Unigramas como Features.

ANEXO B. TABLAS COMPLETAS DE RESULTADOS DE
CLASIFICACIÓN

algoritmo: SVM

features: Presencia de Unigramas

training set: Guía Óleo (balanceado), **test set:** Google Play

método de validación: 5-fold cross validation

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	225	250	0.733	0.900	0.900	0.808	168	250	0.870	0.672	0.672	0.758	0.786	0.783
1000	450	500	0.718	0.900	0.900	0.799	323	500	0.866	0.646	0.646	0.740	0.773	0.769
2000	941	1000	0.673	0.941	0.941	0.784	542	1000	0.902	0.542	0.542	0.677	0.742	0.731
4000	1839	2000	0.700	0.919	0.919	0.795	1210	2000	0.883	0.605	0.605	0.718	0.762	0.756
6000	2768	3000	0.730	0.923	0.923	0.815	1975	3000	0.895	0.658	0.658	0.759	0.790	0.787
8000	3665	4000	0.733	0.916	0.916	0.815	2667	4000	0.888	0.667	0.667	0.762	0.791	0.788
10000	4623	5000	0.721	0.925	0.925	0.810	3209	5000	0.895	0.642	0.642	0.747	0.783	0.779
14000	6540	7000	0.709	0.934	0.934	0.806	4317	7000	0.904	0.617	0.617	0.733	0.775	0.770
18000	8345	9000	0.705	0.927	0.927	0.801	5510	9000	0.894	0.612	0.612	0.727	0.770	0.764
22000	10127	11000	0.721	0.921	0.921	0.809	7088	11000	0.890	0.644	0.644	0.748	0.782	0.778

Tabla B.24: Resultados de Clasificación Supervisada cambiando de Dominio y utilizando el Algoritmo SVM y Presencia de Unigramas como Features.

algoritmo: DecisionTrees

features: Presencia de Unigramas

training set: Guía Óleo (balanceado), **test set:** Google Play

método de validación: 5-fold cross validation

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	215	250	0.592	0.860	0.860	0.701	102	250	0.745	0.408	0.408	0.527	0.634	0.614
1000	449	500	0.579	0.898	0.898	0.704	173	500	0.772	0.346	0.346	0.478	0.622	0.591
2000	915	1000	0.549	0.915	0.915	0.686	249	1000	0.746	0.249	0.249	0.373	0.582	0.530
4000	1610	2000	0.715	0.805	0.805	0.757	1358	2000	0.777	0.679	0.679	0.725	0.742	0.741
6000	2522	3000	0.743	0.841	0.841	0.789	2128	3000	0.817	0.709	0.709	0.759	0.775	0.774
8000	3513	4000	0.668	0.878	0.878	0.759	2251	4000	0.822	0.563	0.563	0.668	0.720	0.713
10000	4307	5000	0.686	0.861	0.861	0.764	3029	5000	0.814	0.606	0.606	0.695	0.734	0.729
14000	6368	7000	0.631	0.910	0.910	0.745	3280	7000	0.838	0.469	0.469	0.601	0.689	0.673
18000	8217	9000	0.652	0.913	0.913	0.761	4616	9000	0.855	0.513	0.513	0.641	0.713	0.701
22000	9932	11000	0.647	0.903	0.903	0.754	5588	11000	0.840	0.508	0.508	0.633	0.705	0.694

Tabla B.25: Resultados de Clasificación Supervisada cambiando de Dominio y utilizando el Algoritmo DecisionTrees y Presencia de Unigramas como Features.

B.4 Clasificación Supervisada para Corpus Desbalanceado

algoritmo: Naïve Bayes
features: Presencia de Unigramas
corpus: Guía Óleo Desbalanceado - 80% positivos y 20% negativos
método de validación: 5-fold cross validation
preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	388	400	0.896	0.970	0.970	0.932	55	100	0.821	0.550	0.550	0.659	0.760	0.795
1000	784	800	0.928	0.980	0.980	0.953	139	200	0.897	0.695	0.695	0.783	0.837	0.868
2000	1568	1600	0.937	0.980	0.980	0.958	294	400	0.902	0.735	0.735	0.810	0.857	0.884
4000	3138	3200	0.958	0.981	0.981	0.969	664	800	0.915	0.830	0.830	0.870	0.905	0.920
6000	4695	4800	0.965	0.978	0.978	0.971	1028	1200	0.907	0.857	0.857	0.881	0.917	0.926
8000	6238	6400	0.963	0.975	0.975	0.969	1363	1600	0.894	0.852	0.852	0.872	0.913	0.921
10000	7789	8000	0.963	0.974	0.974	0.968	1702	2000	0.890	0.851	0.851	0.870	0.912	0.919
14000	10918	11200	0.961	0.975	0.975	0.968	2361	2800	0.893	0.843	0.843	0.868	0.909	0.918
18000	14013	14400	0.963	0.973	0.973	0.968	3061	3600	0.888	0.850	0.850	0.869	0.912	0.918
22000	17130	17600	0.964	0.973	0.973	0.969	3769	4400	0.889	0.857	0.857	0.873	0.915	0.921

Tabla B.26: Resultados de Clasificación utilizando el Algoritmo Naïve Bayes, Presencia de Unigramas como Features y Corpus Desbalanceado: 80% positivos y 20% negativos.

algoritmo: MaxEnt
features: Presencia de Unigramas
corpus: Guía Óleo Desbalanceado - 80% positivos y 20% negativos
método de validación: 5-fold cross validation
preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	378	400	0.896	0.945	0.945	0.920	56	100	0.718	0.560	0.560	0.629	0.752	0.774
1000	755	800	0.924	0.944	0.944	0.934	138	200	0.754	0.690	0.690	0.721	0.817	0.827
2000	1554	1600	0.940	0.971	0.971	0.955	301	400	0.867	0.752	0.752	0.806	0.862	0.881
4000	3135	3200	0.949	0.980	0.980	0.964	630	800	0.906	0.787	0.787	0.843	0.884	0.903
6000	4698	4800	0.960	0.979	0.979	0.969	1003	1200	0.908	0.836	0.836	0.870	0.907	0.920
8000	6322	6400	0.957	0.988	0.988	0.972	1318	1600	0.944	0.824	0.824	0.880	0.906	0.926
10000	7903	8000	0.956	0.988	0.988	0.971	1633	2000	0.944	0.817	0.817	0.876	0.902	0.924
14000	11059	11200	0.959	0.987	0.987	0.973	2331	2800	0.943	0.833	0.833	0.884	0.910	0.929
18000	14191	14400	0.964	0.985	0.985	0.975	3073	3600	0.936	0.854	0.854	0.893	0.920	0.934
22000	17309	17600	0.967	0.983	0.983	0.975	3816	4400	0.929	0.867	0.867	0.897	0.925	0.936

Tabla B.27: Resultados de Clasificación utilizando el Algoritmo MaxEnt, Presencia de Unigramas como Features y Corpus Desbalanceado: 80% positivos y 20% negativos.

ANEXO B. TABLAS COMPLETAS DE RESULTADOS DE
CLASIFICACIÓN

algoritmo: SVM

features: Presencia de Unigramas

corpus: Guía Óleo Desbalanceado - 80% positivos y 20% negativos

método de validación: 5-fold cross validation

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	379	400	0.896	0.948	0.948	0.921	56	100	0.727	0.560	0.560	0.633	0.754	0.777
1000	756	800	0.930	0.945	0.945	0.937	143	200	0.765	0.715	0.715	0.739	0.830	0.838
2000	1546	1600	0.939	0.966	0.966	0.952	299	400	0.847	0.748	0.748	0.794	0.857	0.873
4000	3117	3200	0.956	0.974	0.974	0.965	658	800	0.888	0.823	0.823	0.854	0.898	0.910
6000	4703	4800	0.964	0.980	0.980	0.972	1022	1200	0.913	0.852	0.852	0.881	0.916	0.927
8000	6251	6400	0.962	0.977	0.977	0.969	1352	1600	0.901	0.845	0.845	0.872	0.911	0.921
10000	7811	8000	0.965	0.976	0.976	0.970	1713	2000	0.901	0.857	0.857	0.878	0.916	0.924
14000	10937	11200	0.966	0.977	0.977	0.971	2412	2800	0.902	0.861	0.861	0.881	0.919	0.926
18000	14101	14400	0.968	0.979	0.979	0.973	3130	3600	0.913	0.869	0.869	0.891	0.924	0.932
22000	17198	17600	0.971	0.977	0.977	0.974	3883	4400	0.906	0.882	0.882	0.894	0.930	0.934

Tabla B.28: Resultados de Clasificación utilizando el Algoritmo SVM, Presencia de Unigramas como Features y Corpus Desbalanceado: 80% positivos y 20% negativos.

algoritmo: DecisionTrees

features: Presencia de Unigramas

corpus: Guía Óleo Desbalanceado - 80% positivos y 20% negativos

método de validación: 5-fold cross validation

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	344	400	0.884	0.860	0.860	0.872	55	100	0.495	0.550	0.550	0.521	0.705	0.697
1000	723	800	0.911	0.904	0.904	0.907	129	200	0.626	0.645	0.645	0.635	0.774	0.771
2000	1452	1600	0.931	0.907	0.907	0.919	292	400	0.664	0.730	0.730	0.695	0.819	0.807
4000	2994	3200	0.946	0.936	0.936	0.941	630	800	0.754	0.787	0.787	0.770	0.862	0.856
6000	4529	4800	0.946	0.944	0.944	0.945	941	1200	0.776	0.784	0.784	0.780	0.864	0.862
8000	6059	6400	0.948	0.947	0.947	0.947	1267	1600	0.788	0.792	0.792	0.790	0.869	0.869
10000	7549	8000	0.945	0.944	0.944	0.944	1561	2000	0.776	0.780	0.780	0.778	0.862	0.861
14000	10569	11200	0.947	0.944	0.944	0.945	2205	2800	0.778	0.787	0.787	0.782	0.866	0.864
18000	13641	14400	0.949	0.947	0.947	0.948	2862	3600	0.790	0.795	0.795	0.793	0.871	0.870
22000	16646	17600	0.950	0.946	0.946	0.948	3529	4400	0.787	0.802	0.802	0.795	0.874	0.871

Tabla B.29: Resultados de Clasificación utilizando el Algoritmo DecisionTrees, Presencia de Unigramas como Features y Corpus Desbalanceado: 80% positivos y 20% negativos.

ANEXO B. TABLAS COMPLETAS DE RESULTADOS DE
CLASIFICACIÓN

algoritmo: Naïve Bayes

features: Presencia de Unigramas

corpus: Guía Óleo Desbalanceado - 20% positivos y 80% negativos

método de validación: 5-fold cross validation

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	65	100	0.714	0.650	0.650	0.681	374	400	0.914	0.935	0.935	0.925	0.792	0.803
1000	166	200	0.735	0.830	0.830	0.779	740	800	0.956	0.925	0.925	0.940	0.877	0.860
2000	354	400	0.821	0.885	0.885	0.852	1523	1600	0.971	0.952	0.952	0.961	0.918	0.907
4000	740	800	0.824	0.925	0.925	0.872	3042	3200	0.981	0.951	0.951	0.965	0.938	0.919
6000	1113	1200	0.834	0.927	0.927	0.878	4579	4800	0.981	0.954	0.954	0.967	0.941	0.923
8000	1485	1600	0.834	0.928	0.928	0.878	6104	6400	0.982	0.954	0.954	0.967	0.941	0.923
10000	1850	2000	0.826	0.925	0.925	0.872	7609	8000	0.981	0.951	0.951	0.966	0.938	0.919
14000	2614	2800	0.818	0.934	0.934	0.872	10617	11200	0.983	0.948	0.948	0.965	0.941	0.918
18000	3357	3600	0.799	0.932	0.932	0.861	13555	14400	0.982	0.941	0.941	0.961	0.937	0.911
22000	4097	4400	0.777	0.931	0.931	0.847	16421	17600	0.982	0.933	0.933	0.957	0.932	0.902

Tabla B.30: Resultados de Clasificación utilizando el Algoritmo Naïve Bayes, Presencia de Unigramas como Features y Corpus Desbalanceado: 20% positivos y 80% negativos.

algoritmo: MaxEnt

features: Presencia de Unigramas

corpus: Guía Óleo Desbalanceado - 20% positivos y 80% negativos

método de validación: 5-fold cross validation

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	61	100	0.693	0.610	0.610	0.649	373	400	0.905	0.932	0.932	0.919	0.771	0.784
1000	134	200	0.761	0.670	0.670	0.713	758	800	0.920	0.948	0.948	0.933	0.809	0.823
2000	313	400	0.841	0.782	0.782	0.811	1541	1600	0.947	0.963	0.963	0.955	0.873	0.883
4000	647	800	0.890	0.809	0.809	0.847	3120	3200	0.953	0.975	0.975	0.964	0.892	0.906
6000	967	1200	0.908	0.806	0.806	0.854	4702	4800	0.953	0.980	0.980	0.966	0.893	0.910
8000	1335	1600	0.906	0.834	0.834	0.869	6262	6400	0.959	0.978	0.978	0.969	0.906	0.919
10000	1627	2000	0.916	0.814	0.814	0.862	7851	8000	0.955	0.981	0.981	0.968	0.897	0.915
14000	2383	2800	0.914	0.851	0.851	0.882	10977	11200	0.963	0.980	0.980	0.972	0.916	0.92
18000	3001	3600	0.917	0.834	0.834	0.874	14130	14400	0.959	0.981	0.981	0.970	0.907	0.922
22000	3677	4400	0.907	0.836	0.836	0.870	17223	17600	0.960	0.979	0.979	0.969	0.907	0.919

Tabla B.31: Resultados de Clasificación utilizando el Algoritmo MaxEnt, Presencia de Unigramas como Features y Corpus Desbalanceado: 20% positivos y 80% negativos.

ANEXO B. TABLAS COMPLETAS DE RESULTADOS DE
CLASIFICACIÓN

algoritmo: SVM

features: Presencia de Unigramas

corpus: Guía Óleo Desbalanceado - 20% positivos y 80% negativos

método de validación: 5-fold cross validation

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	65	100	0.619	0.650	0.650	0.634	360	400	0.911	0.900	0.900	0.906	0.775	0.770
1000	146	200	0.756	0.730	0.730	0.743	753	800	0.933	0.941	0.941	0.937	0.836	0.840
2000	313	400	0.807	0.782	0.782	0.794	1525	1600	0.946	0.953	0.953	0.950	0.868	0.872
4000	669	800	0.855	0.836	0.836	0.846	3087	3200	0.959	0.965	0.965	0.962	0.900	0.904
6000	1042	1200	0.867	0.868	0.868	0.868	4640	4800	0.967	0.967	0.967	0.967	0.917	0.917
8000	1389	1600	0.867	0.868	0.868	0.867	6186	6400	0.967	0.967	0.967	0.967	0.917	0.917
10000	1734	2000	0.857	0.867	0.867	0.862	7711	8000	0.967	0.964	0.964	0.965	0.915	0.914
14000	2497	2800	0.879	0.892	0.892	0.885	10857	11200	0.973	0.969	0.969	0.971	0.931	0.928
18000	3211	3600	0.874	0.892	0.892	0.883	13939	14400	0.973	0.968	0.968	0.970	0.930	0.927
22000	3907	4400	0.875	0.888	0.888	0.881	17041	17600	0.972	0.968	0.968	0.970	0.928	0.926

Tabla B.32: Resultados de Clasificación utilizando el Algoritmo SVM, Presencia de Unigramas como Features y Corpus Desbalanceado: 20% positivos y 80% negativos.

algoritmo: DecisionTrees

features: Presencia de Unigramas

corpus: Guía Óleo Desbalanceado - 20% positivos y 80% negativos

método de validación: 5-fold cross validation

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	71	100	0.490	0.710	0.710	0.580	326	400	0.918	0.815	0.815	0.864	0.762	0.722
1000	136	200	0.533	0.680	0.680	0.598	681	800	0.914	0.851	0.851	0.882	0.766	0.740
2000	265	400	0.586	0.662	0.662	0.622	1413	1600	0.913	0.883	0.883	0.898	0.773	0.760
4000	558	800	0.636	0.698	0.698	0.665	2880	3200	0.922	0.900	0.900	0.911	0.799	0.788
6000	897	1200	0.668	0.748	0.748	0.705	4354	4800	0.935	0.907	0.907	0.921	0.827	0.813
8000	1195	1600	0.674	0.747	0.747	0.709	5822	6400	0.935	0.910	0.910	0.922	0.828	0.815
10000	1458	2000	0.665	0.729	0.729	0.696	7266	8000	0.931	0.908	0.908	0.919	0.819	0.807
14000	2121	2800	0.671	0.757	0.757	0.711	10158	11200	0.937	0.907	0.907	0.922	0.832	0.817
18000	2704	3600	0.668	0.751	0.751	0.707	13056	14400	0.936	0.907	0.907	0.921	0.829	0.814
22000	3345	4400	0.676	0.760	0.760	0.716	15997	17600	0.938	0.909	0.909	0.923	0.835	0.819

Tabla B.33: Resultados de Clasificación utilizando el Algoritmo DecisionTrees, Presencia de Unigramas como Features y Corpus Desbalanceado: 20% positivos y 80% negativos.

B.5 Clasificación No Supervisada para Corpus Desbalanceado

algoritmo: Turney adaptado al español

corpus: Guía Óleo Desbalanceado - 80% positivos y 20% negativos

método de validación: hold-out

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	59	80	0.922	0.738	0.738	0.819	15	20	0.417	0.750	0.750	0.536	0.744	0.678
1000	128	160	0.934	0.800	0.800	0.862	31	40	0.492	0.775	0.775	0.602	0.788	0.732
2000	263	320	0.953	0.822	0.822	0.883	67	80	0.540	0.838	0.838	0.657	0.830	0.770
4000	525	640	0.956	0.820	0.820	0.883	136	160	0.542	0.850	0.850	0.662	0.835	0.772
6000	800	960	0.954	0.833	0.833	0.889	201	240	0.557	0.838	0.838	0.669	0.835	0.779
8000	1070	1280	0.945	0.836	0.836	0.887	258	320	0.551	0.806	0.806	0.655	0.821	0.771
10000	1334	1600	0.946	0.834	0.834	0.886	324	400	0.549	0.810	0.810	0.655	0.822	0.770
14000	1859	2240	0.955	0.830	0.830	0.888	472	560	0.553	0.843	0.843	0.668	0.836	0.778
18000	2385	2880	0.959	0.828	0.828	0.889	618	720	0.555	0.858	0.858	0.674	0.843	0.782
22000	2939	3520	0.961	0.835	0.835	0.894	761	880	0.567	0.865	0.865	0.685	0.850	0.789

Tabla B.34: Resultados de Clasificación utilizando el Algoritmo Turney adaptado al español y Corpus Desbalanceado: 80% positivos y 20% negativos.

algoritmo: Turney adaptado al español

corpus: Guía Óleo Desbalanceado - 20% positivos y 80% negativos

método de validación: hold-out

preprocesamientos: seleccionados

tamaño de corpus	clase positivos						clase negativos						avg.	
	t_p	cases	prec.	rec.	acc.	F1	t_p	cases	prec.	rec.	acc.	F1	acc.	F1
500	16	20	0.500	0.800	0.800	0.615	64	80	0.941	0.800	0.800	0.865	0.800	0.740
1000	34	40	0.324	0.850	0.850	0.469	89	160	0.937	0.556	0.556	0.698	0.703	0.584
2000	70	80	0.320	0.875	0.875	0.468	171	320	0.945	0.534	0.534	0.683	0.705	0.575
4000	143	160	0.373	0.894	0.894	0.527	400	640	0.959	0.625	0.625	0.757	0.759	0.642
6000	214	240	0.392	0.892	0.892	0.545	628	960	0.960	0.654	0.654	0.778	0.773	0.661
8000	286	320	0.407	0.894	0.894	0.560	864	1280	0.962	0.675	0.675	0.793	0.784	0.677
10000	360	400	0.404	0.900	0.900	0.558	1069	1600	0.964	0.668	0.668	0.789	0.784	0.673
14000	509	560	0.402	0.909	0.909	0.558	1484	2240	0.967	0.662	0.662	0.786	0.786	0.672
18000	653	720	0.411	0.907	0.907	0.566	1945	2880	0.967	0.675	0.675	0.795	0.791	0.681
22000	801	880	0.412	0.910	0.910	0.567	2376	3520	0.968	0.675	0.675	0.795	0.793	0.681

Tabla B.35: Resultados de Clasificación utilizando el Algoritmo Turney adaptado al español y Corpus Desbalanceado: 20% positivos y 80% negativos.

Índice Alfabético

- accuracy, 43
- análisis de sentimientos, 1
- análisis subjetivo, *véase* análisis de sentimientos
- atributos de opinión, *véase* features
- bootstrap, 47
- clasificación de textos, 15, 52, 59
 - combinación de clasificadores, 41
 - léxico de opinión, *véase* léxico de opinión
 - múltiples aspectos, 38
 - puntajes, 38
 - reglas ad-hoc, 16
 - técnicas de machine learning, *véase* machine learning
- corpus, 8, 64
- cross validation, *véase* k-fold cross validation
- Decision Trees, 32
- evaluación de clasificadores, 42, 71
 - algoritmos supervisados vs no supervisados, 77
 - cambio de dominio, 77
 - corpus desbalanceado, 45, 79
 - evaluación por algoritmo, 74, 75
 - evaluación por atributo, 73
 - métodos de evaluación, 46
 - métricas, 42
- extracción de opiniones, *véase* análisis de sentimientos
- F-measure, 45
 - F1-measure, 45
- features, 8, 17, 68
- freeling, 39, 62
- Google N-Grams Dataset, 14
- hold-out, 46
- k-fold cross validation, 46
- léxico de opinión, 8, 36
- leave-one-out cross validation, 47
- machine learning, 16
 - aprendizaje no supervisado, 17, 53
 - aprendizaje semi-supervisado, 17
 - aprendizaje supervisado, 17, 52
- macro averaging, 46
- markov model, 10
- Maximum Likelihood Estimate, 11
- megam, 61
- micro averaging, 46
- minería de opiniones, *véase* análisis de sentimientos
- minería de sentimientos, *véase* análisis de sentimientos
- Modelo de Máxima Entropía, 26
- modelos de n-grama, 9
 - bigramas, 10
 - n-gramas, 11
 - trigramas, 10
 - unigramas, 10
- modelos discriminativos, 19
- modelos generativos, 19
- multi-aspect sentiment analysis, 38
- Naïve Bayes, 20
 - Binarized Multinomial Naïve Bayes, 24
 - Multinomial Naïve Bayes, 21
- NLTK, 61
- normalización, 48
- opinion extraction, *véase* análisis de sentimientos
- opinion mining, *véase* análisis de sentimientos

out of domain validation, 77

parsing, 39

part-of-speech tagging, 49

precision, 43

preprocesamiento, 47, 69

recall, 43

scikit-learn, 62

stemming, 48

Support Vector Machines, 31

técnicas de smoothing

- Add-one, *véase* Laplace
- Backoff, 13
- Good Turing, 13
- Kneser-Ney, 14
- Laplace, 12
- linear interpolation, 12
- Witten-Bell, 14

tokenización, 48

Turney, 34, 62