Trabajo Fin de Máster: Título del Trabajo

Trabajo Fin de Máster

Nombre del Alumn@

Trabajo de investigación para el

Máster en Lenguajes y Sistemas Informáticos

Universidad Nacional de Educación a Distancia

Dirigido por el

Prof. Dr. D. Nombre Direct@r

Octubre 2018

Agradecimientos

Agradecimientos si procede.

Resumen

En el presente trabajo se propone una arquitectura para la detección del machismo en la red de microblogging Twitter. En la actualidad, el abuso online se ha convertido en un gran problema, especialmente por el anonimato y la interactividad de la web que facilita el incremento y permanencia de este tipo de abusos. Se trata de un campo en el que ha aumentado la producción científica enormemente durante este mismo año y donde se han desarrollado competiciones con gran participación por parte de la comunidad científica. A lo largo del trabajo, se presenta el ciclo completo para la recolección de datos, preprocesamiento y construcción del sistema de clasificación. Se desarrolla un sistema... Se evalua... Los resultados demuestran ... Finalmente, se identifican algunos problemas y líneas de trabajo futuras.

Completar cuando se avance en el trabajo

Abstract

Breve resumen del trabajo realizado y de los objetivos conseguidos en inglés.

Índice general

Índice de Figuras

Índice de Tablas

Capítulo 1

Introducción

1.1. Motivación

Con el rápido crecimiento de las redes sociales, la comunicación entre personas de diferentes culturas en todo el mundo se ha convertido mucho más directa y sencilla. Esto provoca un gran aumento de los "ciber" conflictos entre las personas que utilizan con frecuencia este tipo de plataformas. Con millones de contribuciones e información generada diariamente por los usuarios de este tipo de herramientas, resulta impracticable y poco escalable realizar una política manual para detectar el abuso y el machismo. Pese a esto, empresas como Facebook han anunciado planes para contratar varios miles de empleados encargados de moderar el contenido de la plataforma (Quartz, 2017). Pese a dedicar muchos esfuerzos y recursos, las grandes compañías como Twitter encuentran gran cantidad de dificultades para afrontar el problema (Atlantic, 2016) debido a la gran cantidad de posts que no pueden ser mediados por sus moderadores. Además, han impulsado fuertes iniciativas para responder a las críticas recibidas por no atajar el problema con la suficiente contundencia. Twitter, por ejemplo, ha aplicado politicas para prohibir el uso de sus plataformas para atacar a personas o grupos sociales (Twitter: (Twitter, 2018)). La importancia de este problema junto con la gran cantidad de información generada por los usuarios hace necesaria la creación de sistemas y herramientas que puedan automáticamente detectar el contenido inapropiado en redes sociales.

Un nuevo estudio realizado en EEUU (Duggan, 2017) sostiene que el 41 % de personas encuestadas había sufrido personalmente algún tipo de discriminación o acoso online, de las cuales el 18 % había sufrido algún tipo

de acoso grave, por ejemplo debido a su género (8 %). De igual modo, las mujeres tienen más de el doble de probabilidades de sufrir acoso debido a su género. (ESTE PARRAFO PROVOCA QUE LA SEPARACION ENTRE PARRAFOS CAMBIE)

La importancia del problema radica en la posibilidad de que un abuso verbal en redes sociales acarree eventualmente un acto de violencia física. De hecho, no es inusual que contenidos machistas hacia las mujeres sean trasladados a acciones violentas. Por ejemplo, algunos estudios sociales como (Fulper y Rowe, 2014) demuestra la existencia de una correlación entre el número de violaciones y el número de tweets machistas por estado en USA. Esto sugiere que las redes sociales pueden ser utilizadas como detector de violencia machista.

Amnistía internacional, publicó recientemente un estudio donde denuncia este hecho (International, 2017). En el reporte, se explica cómo para muchas mujeres Twitter es una plataforma donde la violencia y al abuso contra ellas florece, en la mayoría de los casos sin ninguna consecuencia. Según este informe, Twitter está fallando como empresa a la hora de respetar los derechos de la mujer en línea. En lugar de reforzar las voces de las mujeres, la violencia y el abuso que experimentan en la plataforma hace que las mujeres se autocensuren a la hora de postear, limiten sus interacciones e incluso les hace abandonar Twitter por completo. De este modo, la violencia y el abuso que muchas mujeres experimentan en Twitter tiene un efecto perjudicial en su derecho a expresarse en igualdad, libremente y sin miedo.

Todo lo expuesto justifica sin duda la realización de este trabajo, en el que se propone una arquitectura para la detección automática del machismo. Todos los estudios listados justifican la necesidad de detectar y filtrar de un modo automatizado el contenido que incita o promueve el machismo. En concreto, el lenguaje machista o sexista, ocupa gran parte de este discurso en sitios webs como Twitter. Mientras que en la mayoría de las plataformas el uso de este tipo de lenguaje está prohibido, el tamaño de estas redes hace imposible controlar todo el contenido que generan. Para la realización de este proyecto, ...

Completar cuando se avance en el trabajo

1.2. Propuesta y objetivos

Que se ha realizado en el trabajo de fin de master, cuales eran los objetivos y breve resumen de los resultados obtenidos. Texto de prueba 3.

1.3. Estructura del documento

Breve descripción de los capítulos del trabajo.

Capítulo 1. Introducción. Este capítulo introduce los principales motivos que han llevado a la realización de este trabajo, así como la problemática y el estado actual de la disciplina. Por último, se presentan las diferentes contribuciones del trabajo realizado.

Capítulo 2. Estado del arte. Este capítulo describe en mayor detalle la disciplina que nos ocupa, presentando su origen y su historia hasta el presente. Se muestran las técnicas actuales más utilizadas para resolver las tareas más relevantes del tema abordado, así como sus debilidades.

Capítulo 3. Sistema/Método/Caso de Estudio propuesto. En este capítulo se describe en profundidad el sistema/método o caso de estudio propuesto.

Capítulo 4. Evaluación. Este capítulo describe la metodología utilizada para evaluar la propuesta realizada, a la vez que presenta los resultados obtenidos al evaluar el método propuesto en diferentes tareas y sobre colecciones de evaluación de distintos dominios.

Capítulo 5. Discusión. Este capítulo analiza y discute en profundidad los resultados obtenidos en la evaluación presentada en el capítulo anterior.

Capítulo 6. Conclusiones y trabajo futuro. Este capítulo recopila las diferentes conclusiones extraídas del trabajo realizado, y propone algunas líneas de trabajo futuro.

Capítulo 2

Estado del arte

El presente capítulo tiene como objetivo presentar al lector la detección del lenguaje machista en redes sociales. Para ello, se realizará una revisión de los trabajos más relevantes en la tarea de detección de lenguaje abusivo y machista, en los que se analizarán los orígenes de esta tarea, las soluciones técnicas y las aportaciones más relevantes.

2.1. Clasificación de textos

El procesamiento del lenguaje natural (*"Natural Language Processing"*, NLP) tiene como objetivo fundamental el desarrollo de métodos que permitan a los computadores realizar tareas relacionadas con el lenguaje humano, como la comunicación o el procesamiento de textos.

La principal diferencia del NLP con el resto de líneas de investigación relacionadas con el análisis de datos o la inteligencia artificial es la necesidad de un conocimiento del lenguaje en todas sus aplicaciones. Elementos clave del lenguaje como la fonética, la fonología, la morfología, la sintaxis, la semántica, la pragmática y la discursiva son esenciales en cualquier técnica de procesamiento del lenguaje.

Una de las áreas más importantes de investigación relacionadas con el NLP es la clasificación de textos o documentos. De un modo general, se conoce como clasificación automática a la tarea de asignar una o varias categorías predefinidas sobre una colección de instancias a clasificar. Del mismo modo, la clasificación de textos se puede entender como aquella tarea en la que un documento o texto es etiquetado como perteneciente a un determinado conjunto. Este tipo de técnicas se utilizan para un gran número

de aplicaciones:

- Indexación para sistemas de recuperación de información
- Detección de *spam*
- Identificación del lenguaje
- Análisis de sentimientos
- Organización de documentos
- Desambigüación del sentido de las palabras
- Filtrado de textos

Formalmente, el problema se define como un texto o documento d que puede pertenecer a un conjunto fijo de clases $C = \{c_1, c_2, ..., c_i\}$. La salida del sistema como predicción la clase $c \in C$.

Para resolver el problema de la clasificación de textos existen dos enfoques principales: uno basado en reglas y otro mediante algoritmo de clasificación supervisado.

Los sistemas basados en reglas utilizan patrones predefinidos por un experto para crear un conjunto de pautas mediante la combinación de palabras u otros atributos. En este tipo de arquitecturas, la precisión puede ser alta siempre que estas reglas estén cuidadosamente seleccionadas por un experto. Sin embargo, dichos sistemas resultan muy costosos de construir y mantener.

El aprendizaje supervisado se construye sobre un conocimiento a priori. Se debe disponer de un conjunto de documentos de ejemplo para cada una de las categorías consideradas. Después de una etapa de entrenamiento, el sistema queda ajustado de modo que, ante nuevos ejemplos, el algoritmo es capaz de clasificarlos en alguna de las clases existentes. Para este tipo de sistemas se utilizan distintos modelos de clasificadores: *Naive Bayes*, *Regresión logística*, *SVM*, *redes neuronales*, etc.

Para construir cualquier clasificador de textos o documentos es necesario seguir los siguientes pasos:

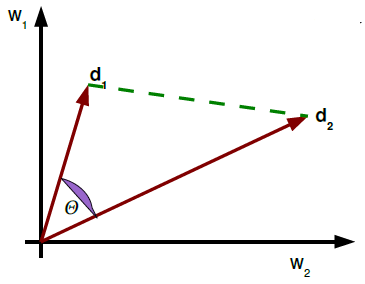
- Extraer los atributos o *features* necesarias para realizar una representación fiel del texto y que permita la utilización de un algoritmo de clasificación
- Desarrollar procedimientos por los cuales los documentos puedan ser clasificados automáticamente dentro de categorías.
- Evaluar la calidad de la clasificación en relación a algún criterio.

2.1.1. Representación textual

La representación del texto es un paso fundamental para el procesamiento automático de textos. Una representación fiel al contenido del documento, que incluya la información necesaria para extraer conocimiento útil, será clave para el desarrollo de una arquitectura con un rendimiento adecuado. En este proceso, se han de tener en cuenta las especificaciones de los algoritmos que se empleen a continuación.

En esta fase, se definen todos los atributos utilizados en el paso posterior por el algoritmo de clasificación. Los atributos seleccionados o generados a partir de los originales serán los que marquen el éxito de la arquitectura completa. La elección del algoritmo de clasificación para los pasos posteriores influirá de un modo mucho menos significativo. Por ejemplo, en (Nina-Alcocer, 2018) y (Endang Wahyu Pamungkas y Patti, 2018) se utiliza el mismo algoritmo de clasificación pero los resultados son muy diferentes debido a los atributos utilizados.

Un modelo de representación muy utilizado se conoce como modelo de representación vectorial. Mediante esta representación, los documentos se modelan como vectores dentro de un espacio euclídeo. De este modo, se pueden aplicar operaciones de distancia entre vectores, como indicador de su cercanía según el contenido textual. En la siguiente imagen se muestra un ejemplo en dos dimensiones:



En este caso, se tendría un vocabulario con únicamente dos rasgos w_1 y w_2 que conforman el espacio en el que se encuentran los documentos o textos d_1 y d_2. De este modo, se pueden emplear medidas de distancia, como la distancia euclídea o la distancia coseno, para comparar ambos documentos.

Utilizando este modelo, un texto quedará representado como una combinación lineal de vectores, donde cada coeficiente representa la relevancia de

cada rasgo en el contenido del texto, calculado con una función de pesado. Para un texto d, un vocabulario de tamaño n: $\vec{d} = t_1 j\vec{t_1} + ... + t_n j\vec{t_n}$. Para el cálculo de la relevancia de cada rasgo $t_n j$, se utilizará una función de pesado. Una de las más utilizadas se conoce como TF-IDF (frecuencia del termino x frecuencia inversa del documento) y se calcularía del siguiente modo:

$$TF - IDF(\vec{t_i}, \vec{d_j}) = f_{ij} \log(\frac{N}{d_f(\vec{t_i})})$$

donde N es la dimensión del corpus (en este caso número de tweets), f_{ij} la frecuencia del término en el documento y $d_f(\vec{t_i})$ el número de documentos (en este caso el tweet) en los que aparece el término.

2.1.2. Clasificación

Como ya se introdujo en apartados anteriores, la clasificación automática de documentos se puede entender como aquella tarea en la que un documento, o una parte del mismo, es etiquetado como perteneciente a un determinado conjunto, grupo o categoría predeterminada.

Los métodos de clasificación supervisados utilizan un conjunto de documentos de ejemplo para cada una de las categorías que presenta la variable objetivo (a clasificar). Estos algoritmos, realizan una etapa de entrenamiento donde se presentan los patrones de ejemplo de modo que ante futuros patrones, el algoritmo será capaz de clasificar en alguna de las clases contenidas en el conjunto de ejemplo. Dentro de este proceso, existen muchas variables que influirán en los resultados del sistema como el tamaño del conjunto de ejemplo, la elección del algoritmo de clasificación o los parámetros de inicialización del mismo.

Se puede ampliar explicación de cada método

Existen numerosos tipos de algoritmos de clasificación, a continuación se indican los más importantes para clasificación textual:

- Naive Bayes: Está basado en la teoría de la decisión de Bayes: la teoría de las probabilidades condicionadas. Por tanto, el problema de la clasificación se reduce al cálculo de las probabilidades a posteriori de una clase dado un documento.
- Arboles de decisión: Se trata de un método que a través de un proceso recursivo de las los atributos de entrada, realiza una representación para clasificar el conjunto de datos presentado.

- Máquinas de vectores de soporte ("Support Vector Machine", SVM): Estos algoritmos pretenden encontrar una hipersuperficie de separación entre clases dentro del espacio de representación.
- Redes Neuronales: Son un modelo computacional compuesto por elementos ("neuronas") interconectados entre sí que aplican una transformación a los datos para producir una salida. Es posible entrenar una red neuronal para que dada una entrada determinada (un vector de representación) produzca una salida deseada (la categoría a la que corresponde ese documento).
- KNN (K-Nearest Neighbour): Este algoritmo se basa en la aplicación de una métrica que establezca la similitud entre un documento que se quiere clasificar y cada uno de los documentos de entrenamiento. La clase o categoría que se asigna al documento sería la categoría del documento más cercano según la métrica establecida.

2.1.3. Métodos de evaluación

En la última fase de un sistema de clasificación textual, el modelo se evalúa con un conjunto de datos de prueba en el que se conocen las clases a las que pertenecen sus documentos.

Para la evaluación de los resultados se utiliza comúnmente la matriz de confusión. Se trata de una una herramienta que representa en cada columna el número de predicciones de cada clase, mientras que cada fila representa a las instancias en la clase real. En la siguiente imagen se presenta un esquema de la matriz de confusión:

		Predicted class	
		P	*N*
Actual Class	*P*	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

Esta tabla está formada por verdaderos positivos, verdaderos negativos,

falsos positivos y falsos negativos. Utilizando estos cuatro componentes se calculas las medidas principales para evaluar los resultados:

- Precisión: representa la fracción de asignaciones correctas frente al total de asignaciones positivas realizadas para esa clase.

$$Precision = \frac{TP}{TP + FP}$$

- Cobertura: representa la fracción de asignaciones positivas respecto al conjunto real de elementos pertenecientes a la clase.

$$Cobertura = \frac{TP}{TP + FN}$$

- Medida-F: combina las dos medidas anteriores.

$$Medida - F = \frac{2xprecisionxcobertura}{precision + cobertura}$$

2.2. Detección de lenguaje o discurso del odio (*hate speech detection*)

La detección del lenguaje machista o sexista está muy relacionada con la detección del lenguaje o discurso del odio en redes sociales. Existen numerosos trabajos donde se intenta detectar distintos tipos de lenguaje del odio, entre ellos el sexismo (HAJIME WATANABE, 2018; Zeerak Waseem, 2016; Georgios K. Pitsilis y Langseth, 2018; Pinkesh Badjatiya, 2017; Steven Zimmerman, 2018; Park y Fung, 2017; Waseem, 2016). El lenguaje del odio se refiere al uso de lenguaje agresivo, violento u ofensivo hacia un grupo específico de personas que comparten una propiedad en común, sea esta propiedad su género, su raza, sus creencias o su religión (Thomas Davidson, 2017). Atendiendo a esta definición, se puede considerar la detección del machismo como un caso particular del discurso del odio. Por ello, es muy interesante realizar una evaluación de los trabajos realizados en esta línea de investigación.

La detección del lenguaje del odio es una linea de investigación muy actual, datando el primer estudio evaluado en el año 2012 (Guang Xiang, 2012). En este articulo se emplea un modelo de detección de temas o categorías (*topic modelling*) que explota la concurrencia de palabras para la

creación de atributos o *features* que alimentarán un algoritmo de clasificación de aprendizaje de máquina o *machine learning*. En la mayoría de trabajos previos se empleaban soluciones basadas en patrones para la clasificación de tweets. Utilizando estos métodos, el uso de expresiones coloquiales y soeces en redes sociales hace más complicado establecer las fronteras entre el uso de lenguaje ofensivo que no tiene como objetivo despreciar a ningún grupo de personas y el lenguaje del odio (Thomas Davidson, 2017).De este modo, este artículo supone un paso muy importante hacia la automatización y a los sistemas basados en algoritmos de *machine learning*.

Durante los últimos tres años, se han sucedido diferentes artículos en la temática aumentando considerablemente la producción científica en este campo. En (Zeerak Waseem, 2016) se aporta el primer corpus de referencia anotado que se utilizará posteriormente en (Waseem, 2016; Georgios K. Pitsilis y Langseth, 2018; Pinkesh Badjatiya, 2017; Steven Zimmerman, 2018; Park y Fung, 2017). Está compuesto por 16.000 *tweets* etiquetados en mensajes sexistas, racistas o sin contenido ofensivo. En este primer trabajo, se sientan las bases de las soluciones aplicadas en el resto de artículos, se utilizan atributos como los *unigramas, bigramas, trigramas* y *cuatri-gramas* y un algoritmo de regresión logística para la clasificación.

En el artículo desarrollado por el mismo autor (Waseem, 2016) se propone una solución similar pero se amplía el corpus en 4033 *tweets* y se utiliza una plataforma de *crowdsourcing* para anotar los mensajes, lo que introduce más diversidad en los criterios del etiquetado. Según los autores, el empeoramiento de los resultados puede deberse al posible sesgo que se produce en (Zeerak Waseem, 2016) ya que los *tweets* solo fueron etiquetados por los autores únicamente.

En el resto de artículos que evalúan su propuesta utilizando el corpus desarrollado por (Waseem, 2016), se utilizan redes neuronales en la etapa de clasificación y, en algunos, en la etapa de preprocesamiento. En la solución propuesta por (Steven Zimmerman, 2018) se aplican redes neuronales convolucionales (*CNN, Convolutional Neural Network*) para codificar el texto y extraer los atributos que se utilizarán para el clasificador final, basado también en CNNs. Esta técnica permite tener en cuenta la posición de la palabra (su contexto) para extraer los atributos de cada *tweet*. Esta misma idea junto con el uso de redes neuronales recurrentes (*RNN, Recurrent Neural Network*) se utiliza en (Pinkesh Badjatiya, 2017) para obtener los

atributos en la etapa de procesamiento. En ambos artículos se consiguen mejorar los resultados alcanzados por (Waseem, 2016) lo que afianza el uso de técnicas basadas en redes neuronales en el procesado del lenguaje.

Una idea interesante es el uso de atributos como la tendencia al racismo o al sexismo sirviéndose del historial de los usuarios. En (Georgios K. Pitsilis y Langseth, 2018) se demuestra como el uso de este tipo de atributos mejora notablemente los resultados. Esta misma idea se utiliza en (Despoina Chatzakouy, 2017) donde se detectan cuentas agresivas estudiando al usuario y su red de seguidores.

En todos los artículos revisados anteriormente, se trata el problema como una clasificación múltiple donde el texto se puede clasificar según las etiquetas racismo, sexismo o ninguno. Sin embargo, se podría resolver el problema con un doble clasificador, el primero detecta si el texto contiene lenguaje abusivo o no y el segundo realizaría la tarea de clasificar en contenido sexista o racista (Park y Fung, 2017).

Un desafío importante en la detección del lenguaje del odio en redes sociales es la separación entre el lenguaje ofensivo y el lenguaje que incita o promueve el odio. Davidson (Thomas Davidson, 2017) aporta un corpus etiquetado de 25.000 *tweets* para diferenciar entre estos 2 tipos de lenguaje. En su trabajo, se propone un modelo similar a (Waseem, 2016) donde se ponen de manifiesto las dificultades de esta solución para considerar el contexto de las palabras. De este modo, si se utilizan palabras que pueden expresar odio (por ejemplo, "*gay*") en un contexto positivo, hay muchas probabilidades de que el sistema detecte odio en el texto. Los resultados serán mejorados posteriormente en (HAJIME WATANABE, 2018) donde se ampliará el número de*features* y se utilizará un algoritmo basado en árboles de decisión para la tarea de clasificación.

2.3. Detección de la misoginia

La misoginia se define según la RAE como *"Aversión a las mujeres"* (RAE, b). El machismo, sin embargo, se define como "Actitud de prepotencia de los varones respecto de las mujeres" o "forma de sexismo caracterizada por la prevalencia del varón" (RAE, a). Si bien estos dos términos tienen matices distintos, tienen como denominador común la discriminación de las mujeres debido a su sexo. De hecho, existen trabajos donde se expone que

la misoginia se manifiesta lingüísticamente mediante la exclusión, discriminación, hostilidad, trato de violencia objetificación o cosificación sexual (Maria Anzovino y Rosso, 2018; E. Fersini y Anzovino, 2018a). Muchas de estas señales textuales de misoginia serían aplicables del mismo modo al machismo (Aranbarri, 2014; Giraldo, 1972).

Durante este último año, se ha llevado a cabo la competición IberEval 2018 donde una de las tareas era la detección automática de la misoginia (https://amiibereval2018.wordpress.com/, 2018) (AMI, *"Automatic Misogyny Identification"*). En esta tarea se propone la labor de identificar la misoginia en *tweets* en español e inglés. En total, participaron once equipos de cinco países distintos para la detección en inglés, mientras que para la detección en castellano participaron un total de ocho equipos (E. Fersini y Anzovino, 2018b). Los artículos publicados para esta tarea en castellano resultan de gran interés, pues guarda una relación importante con el presente trabajo.

Para la tarea de clasificación, la mayoría de los equipos utilizaron Máquinas de Vectores de Soporte (SVM, *Suppor Vector Machines*) y métodos combinados de aprendizaje (EoC, *Ensemble of Classifiers*). Las técnicas basadas en SVMs fueron utilizadas por (Canós, 2018; Endang Wahyu Pamungkas y Patti, 2018; Nina-Alcocer, 2018). Los equipos (Resham Ahluwalia y Cock, 2018; Elena Shushkevich, 2018; Simona Frenda y y Gomez, 2018; Han Liu y Cocea, 2018) aplicaron técnicas EoC, mientras que en (Goenaga y Perez, 2018) se exploraron soluciones basadas en redes neuronales.

Las soluciones aportadas por (Canós, 2018; Endang Wahyu Pamungkas y Patti, 2018) obtuvieron la mejor tasa de acierto para la detección de la misoginia en castellano. El modelo propuesto por (Canós, 2018) utiliza *features* basadas en la vectorización de cada tweet, utilizando la medida tf-idf (*term frequency - Inverse document frequency*). Posteriormente, se emplea un modelo SVM con núcleo lineal para la etapa de clasificación. Esta solución tan sencilla alcanza los mejores resultados para *tweets* en castellano, pero empeora considerablemente para *tweets* en inglés.

Una idea interesante, explorada en (Endang Wahyu Pamungkas y Patti, 2018), es el uso de un léxico auxiliar que contenga palabras que se encuentren con frecuencia en textos sexistas. Este léxico fue desarrollado en un trabajo italiano (De Mauro, 2016). En dicho estudio, se utiliza como clasificador un modelo basado en SVM con núcleo lineal para el castellano y núcleo radial

para el inglés. En este caso, se alcanza la máxima tasa de acierto en inglés y en español.

2.3.1. Corpus disponibles

A continuación se citan algunos corpus que pueden ser utilizados para la detección de lenguaje del odio en textos:

- IberEval 2018 Automatic Misogyny Identification (E. Fersini y Anzovino, 2018b): Se trata de un corpus etiquetado que contiene campos que denotan si el texto contenido en un tweet tiene un componente sexista. Fue recogido entre el 20-07-2018 y 30-11-2017 donde se recogieron 83 millones de tweets en inglés y 72 millones en castellano. Para el proceso de etiquetado se utilizadon dos pasos: en el primero dos anotadores etiquetaban el conjunto y en el segundo se utilizó una plataforma de crowdsourcing. Finalmente, se etiquetaron 3521 tweets en inglés y 3307 en español para la fase de entrenamiento. En cuanto al conjunto de test, se compartieron 831 tweets en español y 726 en inglés.
- Corpus etiquetado (Zeerak Waseem, 2016): Está compuesto por 16.000 *tweets* etiquetados para mensajes sexistas, racistas o sin contenido ofensivo.

2.3.2. Ejemplo subsección

Capítulo 3

Herramientas utilizadas

En este capítulo se describen en profundidad las distintas herramientas evaluadas para la creación del sistema propuesto. Además, se exponen los motivos por los que se han elegido frente a otras alternativas disponibles.

3.1. Crawler

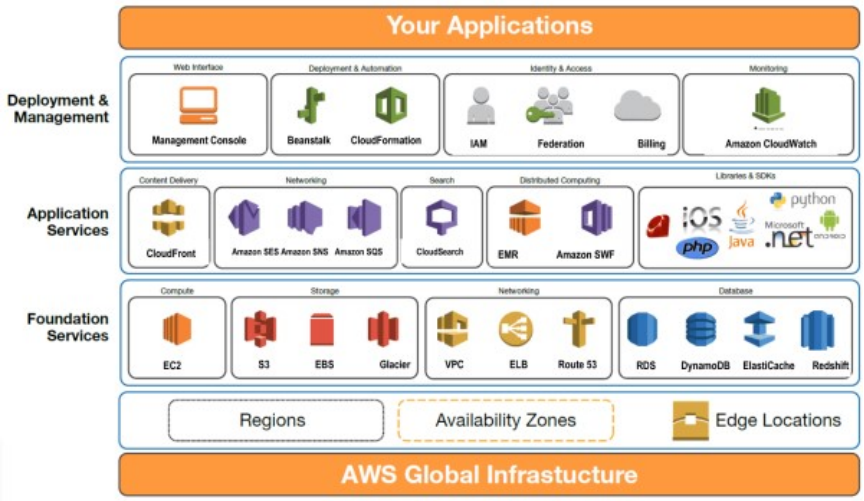
3.1.1. Amazon Web Services

AWS es una creciente unidad dentro la compañía Amazon.com que ofrece una importante variedad de soluciones de Cloud Computing a empresas tanto PYMES como grandes organizaciones a través de su infraestructura interna siendo la marca más utilizada actualmente en el mercado de la nube con casi un 40 % de cuota de mercado (Research, 2018). Amazon ofrece unos servicios en la nube pública mediante una tarificación de precios en función del tiempo de uso, anchos de banda consumidos, etc. Por lo tanto, su gran ventaja competitiva es ofrecer unos recursos de infraestructura y plataforma poco asumibles a la mayoría de empresas para el periodo que se requiera.

Los clientes de AWS tan sólo deben pagar lo que usen del servicio, de esta manera, obtener unos potentes servidores con una plataforma determinada, un espacio de almacenamiento o una gran base de datos supone la adquisición de un hardware que no se aproveche todo el tiempo, que tan sólo interese para un periodo determinado y satisfacer una necesidad puntual, prescindiendo de importantes inversiones en infraestructura. Orientado a empresas, se adapta con total flexibilidad y escalabilidad a las necesidades de Cloud que tenga el cliente, mediante un acuerdo de nivel de servicio,

se especifica el nivel de compromiso del servicio, disponibilidad y ofrece un punto de confianza que otros proveedores de nube pública no proporcionan, dato que le da ventaja frente a sus competidores.

Dado que ha sido pionero en el sector y posee una gran cantidad de desarrolladores que trabajan para mejorar el servicio, desde su publicación en 2006, ha sido líder en el sector por delante de Google App Engine, Azure de Microsoft, Alibaba, etc (Research, 2018). Siempre ha ido un paso por delante y le ha permitido innovar en el sector y ofrecer unos precios muy competitivos, soluciones para todos los gustos e importantes acuerdos con Microsoft, IBM y HP como estrategias de marketing para ofrecer software y plataformas propietarias (además de software libre que fue lo primero que se ofrecía con plataformas Linux) en sus imágenes de máquinas virtuales. En la siguiente figura se puede ver un resumen de los servicios de AWS:



Los diferentes servicios de AWS se incrementan con el paso del tiempo, siendo EC2, S3 y Lambda los que más peso tienen en el presente proyecto:

- Amazon Elastic Compute Cloud (EC2): proporciona servidores virtuales escalables. Proporciona las capacidades de Cloud Computing a sus clientes de manera que permite una configuración y administración de las capacidades de máquinas virtuales que se solicitan a la nube, pudiendo pagar tan sólo el tiempo de computación. Actualmente existen numerosos tipos de instancias con características hardware distintas según los requisitos del usuario.

- Amazon Simple Storage Service (S3): proporciona un Web Service basado en el almacenamiento online para aplicaciones. Este almacenamiento en Internet proporciona una simple interfaz web, como su nombre indica, que puede ser usada para almacenar grandes cantidades de datos en cualquier momento desde cualquier sitio, dando acceso confiable y seguro con SLA, altamente escalable, rápido y barato en la infraestructura de Amazon. Físicamente, los datos están distribuidos por los Data Center de Amazon, pero es algo que permanece ajeno al cliente y de lo que no debe preocuparse (escalabilidad). Su integración con EC2 es esencial para que las imágenes de máquinas virtuales puedan trabajar con datos y objetos almacenados en S3 y tener un espacio donde los desarrolladores puedan trabajar cómodamente incluso poder solicitar más espacio temporal para las máquinas o disponer de varios ?buckets? donde compartir datos entre instancias.
- AWS Lambda: se trata de un servicio de computación sin servidor. Este servicio permite ejecutar código sin aprovisionar ni administrar servidores, pagando únicamente por el tiempo de cómputo que se consuma. De este modo, este servicio permite que se AWS quien se encargue de la administración de las máquinas y el usuario únicamente trabaje en el código que se ejecuta.

La segunda plataforma de cloud computing más importante a nivel mundial es Azure, propiedad de la empresa Microsoft. En este caso, no se ha elegido Microsoft Azure porque ya se contaba con un conocimiento previo en el uso de los servicios de AWS. Además, AWS cuenta con servicios de computación serverless, como AWS Lambda, muy útiles para la realización del crawler.

3.1.2. Twitter API y rtweet

Twitter proporciona múltiples APIs para facilitar el acceso a los datos de su plataforma. De todas ellas, la necesaria para crear el corpus objetivo sería el API REST de Twitter. En concreto, es necesario utilizar la funcionalidad Tweet Search que permite realizar búsquedas de los tweets generados en la plataforma según distintos parámetros de búsqueda.

Dentro del API existen 3 tipos de cuenta según la cantidad de información disponible para consulta: Standard Search, Premium Search y Enter-

prise Search. De todas ellas, solamente la primera es gratuita por lo que será la utilizada durante el proceso de generación del corpus. Es importante señalar que este tipo de búsqueda presenta algunas limitaciones. Las dos más importantes serían la existencia de una ventana temporal de consulta limitada a 7 días anteriores y, por otra parte, la limitación de descarga de tweets a 18.000 cada 15 minutos.

Para recopilar la información de Twitter, se ha utilizado la herramienta rtweet (Kearney, 2018). Se trata de un cliente del lenguaje de programación R para acceder al API de Twitter. Este paquete facilita mucho las tareas habituales como la búsqueda de tweets.

Existen varias alternativas a rtweet como tweetpy para el lenguaje de programación Python o twitteR. Se ha optado por rtweet porque ambas alternativas están más desactualizadas y son proyectos mucho más inactivos

3.2. Preprocesado y tokenización

En la etapa inicial para la clasificación de textos, se aplican distintas técnicas que permitan Extraer los atributos o *features* necesarias para realizar una representación fiel del texto y que permita la utilización de un algoritmo de clasificación. Existen multitud de procedimientos aplicables en esta etapa como la tokenización, el reconocimiento de entidades nombradas, el etiquetado sintáctico y morfológico:

- Tokenización: Permite separar cada palabra o símbolo del corpus en unidades independientes (como palabras) que pueden ser almacenadas para su posterior procesado.
- Reconocimiento de entidades nombradas: Tarea que permite clasificar en categorías predefinidas, como personas, organizaciones, lugares, expresiones de tiempo y cantidades.
- Etiquetado sintáctico: Proceso en el que se busca sobre el espacio de todas las posibles combinaciones de las reglas gramaticales definidas para encontrar la estructura de una oración.
- Etiquetado morfológico: En este proceso se le asigna a cada palabra su función dentro del corpus utilizado. Normalmente, se utilizan 8 etiquetas distintas en la mayoría de los idiomas utilizados en Europa: nombre, verbo, pronombre, preposición, adverbio, conjunción, partícula y

articulo.

Para aplicar este tipo de técnicas, existen gran cantidad de proyectos o librerías de computación disponibles. Algunas de las más utilizadas son las siguientes:

- Freeling: Es una librería que soporta el lenguaje español y se utiliza en (Simona Frenda y y Gomez, 2018). Pese a que tiene mucha de las características que se necesitan, tiene una menor comunidad y está menos extendido que algunas del resto de las herramientas.
- Stanford Parser: Se trata de una librería desarrollada por el grupo de trabajo de NLP de la universidad de Stanford.
- TweetNLP: Librería desarrollada específicamente para le procesado de tweets. Su uso no está muy extendido.
- Spacy: Se utiliza en (Waseem, 2016) y permite aplicar las técnicas de procesado de un modo eficiente.
- NLTK: Se trata de la librería más extendida para el preprocesamiento, se utiliza en (Steven Zimmerman, 2018; Thomas Davidson, 2017; Simona Frenda y y Gomez, 2018).

De todas las herramientas listadas, se ha optado por la librería NLTK. Se trata de una librería muy extendida que cuenta con una gran comunidad y permite un desarrollo muy ágil. Algunas librerías como Freeling o Stanford Parser requieren varias dependencias para poder ser utilizadas.

La mejor alternativa a NLTK considerada sería Spacy. Su uso está aumentando y su funcionamiento es muy similar ya que ambas están desarrolladas en Python. Se ha optado por NLTK porque aún sigue siendo más utilizada.

3.2.1. NLTK: Natural Language Toolkit

NLTK es una librería que define una infraestructura en la que crear programas para el procesado del lenguaje natural (NLP, "Natural language processing") en "Python". Provee la estructura básica para representar datos relevantes para el procesado del lenguaje natural, interfaces para realizar tareas como el etiquetado del discurso (POS, "part-of-speech tagging"), etiquetado sintáctico y clasificación de texto (NLTK, 2018).

Esta librería fue desarrollada originalmente en el año 2001 como parte de un curso de lingüística computacional en la universidad de Pennsylvania. Desde entonces, ha sido desarrollado y mejorado por distintos contribuidores al tratarse de un proyecto libre. Actualmente, NLTK es utilizado en gran cantidad de investigaciones y supone un estándar muy importante para realizar tareas relacionadas con NLP. Está compuesto por una cantidad importante de módulos que pueden ser invocados desde un programa escrito en Python. En la siguiente figura se recogen los más importantes (Steven Bird y Loper, 2009):

Explicar con más detalle los módulos que utilice en el trabajo

(LA IMAGEN PROVOCA QUE SE DESCUADRE EL DOCUMENTO)

Language processing task	NLTK modules	Functionality
Accessing corpora	nltk.corpus	standardized interfaces to corpora and lexicons
String processing	nltk.tokenize, nltk.stem	tokenizers, sentence tokenizers, stemmers
Collocation discovery	nltk.collocations	t-test, chi-squared, point-wise mutual information
Part-of-speech tagging	nltk.tag	n-gram, backoff, Brill, HMM, TnT
Classification	nltk.classify, nltk.cluster	decision tree, maximum entropy, naïve Bayes, EM, k-means
Chunking regular	nltk.chunk	expression, n-gram, namedentity
Parsing	nltk.parse	chart, feature-based, unification, probabilistic, dependency
Semantic interpretation	nltk.sem, nltk.inference	lambda calculus, first-order logic, model checking
Evaluation metrics	nltk.metrics	precision, recall, agreement coefficients
Probability and estimation	nltk.probability	frequency distributions, smoothed probability distributions
Applications	nltk.app, nltk.chat	graphical concordancer, parsers, WordNet browser, chatbots
Linguistic fieldwork	nltk.toolbox	manipulate data in SIL Toolbox format

3.3. Scikit-learn

Tal y como se ha ido introduciendo, el objetivo final del presente trabajo es la detección del lenguaje machista en las redes sociales. Este problema se puede modelar como una clasificación de documentos en los que en este caso se asignará una categoría a cada uno de los mensajes que conforman el corpus.

En la actualidad, la mayoría de trabajos llevan a cabo esta tarea mediante algoritmos o ténicas de aprendizaje supervisado (Steven Zimmerman, 2018; Thomas Davidson, 2017; E. Fersini y Anzovino, 2018b). Para el desarrollo de este tipo de métodos se utilizará la librería "scikit-learn" muy utilizada en proyectos de clasificación de documentos (Resham Ahluwalia y Cock, 2018).

Scikit-learn (F. Pedregosa, 2011) es un proyecto que provee una librería de aprendizaje de máquina para el entorno de programación "Python". El objetivo principal del proyecto es establecer un conjunto de herramientas dentro de un entorno de programación que sea accesible a usuarios no expertos. Esta librería incluye algoritmos clásicos de aprendizaje de máquina, herramientas para la selección, evaluación de modelos y preprocesado.

Todo los modelos de aprendizaje supervisados o funciones auxiliares relacionadas con el procesado de datos utilizados en el presente trabajo están implementados o han sido desarrollados con ayuda de funciones disponibles en la librería scikit-learn.

Todos los objetos dentro de la librería comparten un API básica compuesta por 3 interfaces complementarias: "estimators" que permiten construir y ajustar modelos, "predictors" para realizar predicciones y "transformers" que permiten realizar conversiones a los datos.

3.3.1. "Estimators"

La interfaz "estimator" define objetos y provee de un método "fit" para ajustar un modelo a los datos de entrenamiento. Todos los algoritmos supervisados y no supervisados implementados en la librería son tratados como objetos implementando esta interfaz. Otro tipo de tareas relacionadas con el aprendizaje de máquina como la selección de atributos o métodos para la reducción de la dimensionalidad también utilizan el interfaz "estimator".

La inicialización de un "estimator" y el ajuste de un modelo a los datos de entrenamiento están diferenciados en la librería. Un "estimator" se puede inicializar un con conjunto de parámetros de entrada (por ejemplo, el parámetro C para SVM) y, posteriormente, se utiliza el método "fit" para realizar el proceso de ajuste a los datos de entrenamiento. En el siguiente código se ilustra esta funcionalidad:

```
from sklearn.ensemble import RandomForestClassifier
```

```
rf = RandomForestClassifier(n_estimators=250)
rf.fit(X_train, y_train)
```

En el código anterior, primero se inicializa un "estimator" estableciendo el argumento "n_estimator". Tras esto, se realiza una llamada al método "fit" para realizar el ajuste utilizando los datos de entrenamiento.

3.3.2. "Predictors"

La interfaz "predictor" extiende la funcionalidad del "estimator" añadiendo el método "predict". Este método devuelve un vector de predicciones tomando como entrada una matriz con los datos de testeo. Ampliando el ejemplo anterior:

```
y_pred = rf.predict(X_test)
```

3.3.3. "Transformers"

Antes de aplicar un método de clasificación supervisada, suele ser habitual realizar filtrados o modificaciones en los datos, para ello, "scikit-learn" implementa la interfaz "transformer" para llevar a cabo este tipo de tareas.

Esta interfaz define el método "transform" que toma como entrada una matriz de datos y devuelve como salida una versión transformada de estos datos. Algunas de las transformaciones más comunes pueden ser la selección de atributos, preprocesado o métodos de reducción de dimensionalidad. Un ejemplo de preprocesado podría ser el estandarizado de un conjunto de datos:

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(X_train)
X_train = scaler.transform(X_train)
```

3.3.4. "Pipelines y selección de modelos"

"Scikit-learn" permite componer nuevos "estimators" utilizando otros lo que permite crear flujos de trabajo en un único objeto. Este tipo de tarea se puede realizar de dos modos: mediante "Pipeline" utilizando un modelo secuencial o mediante "FeatureUnion".

Los objetos "Pipeline" encadenan "estimators" en un único objeto. Esto permite crear flujos de trabajo siguiendo un número fijo de pasos, por ejemplo: extracción de atributos, reducción de dimensionalidad, ajuste de un modelo y realización de de predicciones.

Los objetos "FeatureUnion" combinan multiples "transformers" en uno único y concatena los resultados. De este modo, este tipo de objeto es capaz de realizar transformaciones distintas sobre el mismo conjunto de datos o sobre una parte del mismo.

Ambos objetos pueden ser combinados para crear flujos de trabajo más complejos. Por ejemplo, en el siguiente código se combinan dos "Pipeline" utilizando "FeatureUnion" y se añade un último paso "clf" que añade un clasificador.

```
Pipeline([('feature-union',
        FeatureUnion([('text-features', text_pipeline),
        ('other-features', preprocess_pipeline)])),
        ('clf', LogisticRegression(penalty = ''L2'')])
```

En "scikit-learn" es posible realizar selección de modelos mediante el meta-estimador "GridSearchCV". Este método toma como entrada un "estimator" cuyos parámetros de entrada deben de ser optimizados. Para ello, se definen todos los valores que se deben de tener en cuenta en el proceso para cada parámetro de entrada.

Capítulo 4

Sistema

En el presente capítulo se describe el sistema automático de detección del machismo en redes sociales. El sistema está basado en aprendizaje supervisado y emplea distintos atributos unificados para, posteriormente, hacer uso de un algoritmo de aprendizaje de máquina que clasificará los registros de entrada en 3 categorías. De este modo, las expresiones textuales de entrada se clasificarán según el grado de machismo que presenten.

El sistema ha sido desarrollado teniendo en cuenta la naturaleza del problema tratado. En este caso, como ya se introdujo, se trata de texto en español por lo que método de clasificación estará preparado para trabajar con este idioma. No obstante, la arquitectura del sistema y las técnicas aplicadas pueden ser adaptadas perfectamente para otros idiomas.

El sistema propuesto se divide en 3 fases principales, tal y como se puede observar en la siguiente figura.

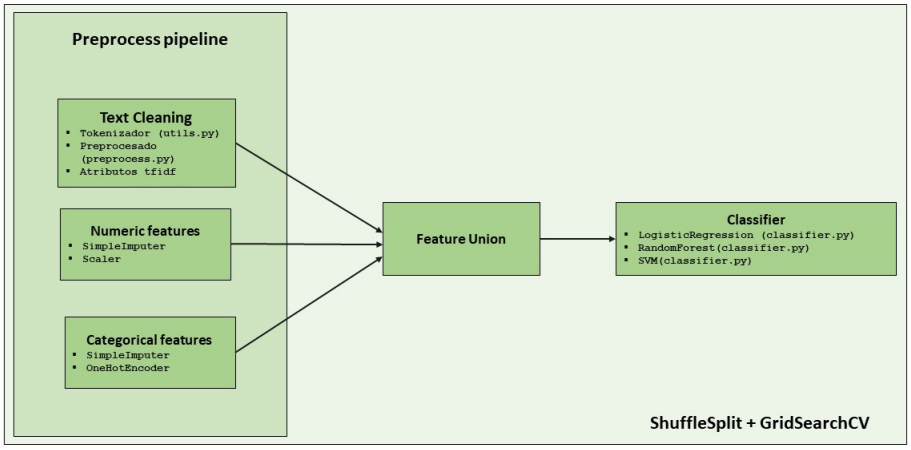


Figura 4.1: Arquitectura clasificador

En la primera etapa, se realiza un preprocesado diferenciado según el tipo de atributo. En este sistema, se consideran 3 tipos de atributos distintos: variables categóricas, numéricas y texto. Para cada atributo, se aplican diferentes métodos como la tokenización, el escalado o la sustitución de emoticonos. Tras esto, se unifican los distintos tipos de atributos procesados en un conjunto de datos común que será la entrada de la última fase. En la última etapa, se emplean algoritmos de clasificación supervisada con la intención de obtener un modelo predictivo capaz de detectar textos machistas. Se emplean 3 métodos distintos de aprendizaje automático en este último paso: Regresión logística, Random Forest y Máquinas de vectores de soporter (SVM, "Support Vector Machine"). Para evaluar los resultados obtenidos con estos algoritmos, se realiza una búsqueda de parámetros de entrada (GridSearchCV) y, posteriormente, se utilizan estos parámetros para realizar una validación cruzada en el conjunto de testeo.

4.1. Preprocesado

La primera fase del sistema de clasificación lleva a cabo diferentes acciones relacionadas con el preprocesado. Se realizan tareas tan importantes como la división del texto en tokens que permitirá que la información lingüística sea tratada por las sucesivas etapas del sistema de clasificación. Como se

introdujo, el tipo de preprocesado depende del tipo de atributo considerado.

4.1.1. Texto

La variable que contiene el texto contenido en el tweet es la más importante para la realización del sistema de detección del machismo. Idealmente, se debería de encontrar en este atributo todas las señales textuales que indican si el mensaje es machista o no. Para este atributo, se aplican 3 métodos distintos: preprocesado, tokenizador y creación de atributos tf-idf.

En el preprocesado de texto se llevan a cabo las siguientes acciones:

- Reemplazo de emojis: Se reemplazan los emojis por una descripción de lo que representa el dibujo. En este caso, resulta útil poder identificar el emoji que se está utilizando ya que puede modificar el significado de la frase. Por ejemplo, en la frase de la siguiente imagen () se puede observar como el emoji permite identificar como machista la expresión:

Incluir emojis en latex

Figura 4.2: Uso de emoji en contexto machista

- Filtrado de URLs:Se reemplazan las URLs por la palabra "*twurl*".
- Reemplazo de acentos: Se eliminan los acentos propios del castellano.
- Filtrado de usuarios: Se reemplazan las menciones de los usuarios (las palabras que comienzan por "@") por la palabra "*twuser*". De este modo, se identifica cuando se utilizan las menciones omitiendo el usuario concreto.
- Convertidor de hastags: Se realiza una conversión para los *hastags* que utilizan las mayúsculas como separador. Por ejemplo, la frase "*#FelizDía*"se convertiría a "*Feliz Día*".
- Filtrado de hastags: Se reemplazan los *hastags* (las palabras que comienzan por "#") por la palabra "*twhastag*".
- Convertidor a minúsculas: Se convierten todos los caracteres a minúscula.
- Reemplazo de exclamaciones: Se reemplazan los signos de exclamación por la palabra "*twexclamation*".

- Reemplazo de interrogaciones: Se reemplazan los signos de interrogación por la palabra "*twinterrogation*".
- Reemplazo de signos de puntuación: Se eliminan los signos de puntuación.

Para ilustrar el funcionamiento de la etapa de preprocesado, supongamos el siguiente mensaje:

Figura 4.3: Ejemplo de preprocesado

Utilizando el preprocesado se obtendría la siguiente oración transformada: "*esta es la reina de las feministas de verdad o no twinterrogation twuser thumbs_up twurl*". Como se puede observar, se han convertido los caracteres a minúscula, se ha reemplazado el caracter de interrogación, se ha sustituido la mención del usuario, se reemplaza el emoji por una descripción y se reemplaza la URL.

A partir del texto preprocesado, se realiza la tokenización de cada mensaje. Para llevar a cabo este proceso se utiliza la clase "TweetTokenizer" disponible en la librería "NLTK". Se trata de un tokenizador desarrollado específicamente para el texto generado en Twitter. Tras la tokenización, se obtiene para cada mensaje una lista de unidades independientes que, mayoritariamente, representarán palabras, emoticonos y signos de puntuación.

Una vez realizada la tokenización del mensaje, se llevan a cabo tres procesos: filtrado de stopwords, reemplazo de abreviaturas y stemming. Las palabras vacías o stopwords constituyen el grupo de palabras si un significado concreto, por ejemplo, artículos, preposiciones o conjunciones. Este tipo de elementos no aportan información adicional al contexto del mensaje y, por tanto, se eliminan antes del proceso de clasificación.

La tarea llevada a cabo en este trabajo presenta una dificultad añadida por el entorno en el que se utiliza el lenguaje. En las redes sociales, y en Twitter en concreto, se publican mensajes cortos y, frecuentemente, no siguen las reglas convencionales del idioma, de este modo, el uso de abreviaturas o emoticones está muy extendido en este tipo de plataformas. Es por ello, que realizar diccionarios específicos para cada idioma que permitan normalizar este tipo de contenido es muy importante previo a la tarea de clasificación. En este trabajo se utiliza el diccionario en castellano realizado en

(Helena Gómez-Adorno, 2016) que compila diccionarios de palabras "slang", abreviaturas, contracciones y emoticones que ayudan al preprocesamiento de textos publicados en redes sociales. Un ejemplo del tipo de expresiones que se reemplazan en este proceso se podría ilustrar del siguiente modo: "*Esta es aki la reuna de las feministas de verdad o no?*". En el ejemplo anterior, se encuentra la palabra coloquial .ªki"que normalizada al español sería "aquí".

Por último, se aplica un método de stemming para reducir las palabras a su raíz. En este caso, se utiliza el algoritmo de Porter (Porter, 1980) desarrollado en la década de los ochenta por Martin Porter y basado en un conjunto de reglas aplicadas en cascada para obtener la raíz de las palabras.

Para ilustrar el proceso anterior, se supone el ejemplo de la figura 4.3. Aplicando el proceso de tokenización se obtendría la siguiente lista de tokens: ['reina', 'feminista', 'verdad', 'twinterrog', 'twuser', 'thumbs_up', 'twurl']. Como se puede observar, se obtiene una lista de palabras, emoticonos y signos de interrogación entre los cuales no se encuentran las palabras vacías.

A partir del texto tokenizado, es necesario representar la información contenida en el texto de un modo interpretable por las etapas posteriores del sistema de clasificación. Para el presente trabajo, se realiza esta representación utilizando vectores de términos *tf-idf* mediante los unigramas de los tokens obtenidos del proceso anterior.

4.1.2. Atributos numéricos

Otro tipo de atributo que se utiliza en el presente sistemas son los numéricos. En este caso particular se consideran los siguientes atributos numéricos:

- display_text_width: número de caracteres del tweet.
- favorite_count: número de veces que el tweet ha sido marcado como favorito.
- retweet_count: número de veces que el tweet ha sido retwiteado.
- followers_count: número de seguidores del usuario que publica el tweet.
- friends_count: número de personas seguidas por el usuario que publica el tweet.
- listed_count: número de listas en las que está inscrito el usuario que publica el tweet.

- statuses_count: número de tweets publicados por el usuario que publicó el tweet.
- favourites_count: número de tweets que el usuario que publicó el tweet marcó como favorito.

El uso de este tipo de atributos permite tener en cuenta distintos aspectos del contexto en el que se genera el tweet. Por ejemplo, existe un grupo de atributos como favorite_count o retweet_count que mide la popularidad del tweet. Por otra parte, existen campos como followers_count que miden la popularidad del usuario que publica el tweet. Además, otros campos como display_text_width demuestran tener una relevancia especial para los tweets etiquetados con categoria "DUDOSO".

En los atributos numéricos se realizan únicamente 2 procesos: imputación de valores nulos y escalado. En este caso, se imputan los valores nulos sustituyéndolos por 0. Para el escalado, se realiza una estandarización para que en los valores numéricos se consiga una media nula y una desviación estándar de uno.

4.1.3. Atributos categóricos

El último tipo de atributo que se emplea son los atributos categóricos. Para este sistema se consideran los siguientes atributos categóricos:

- source: tipo de dispositivo con el que se publica el tweet.
- respuesta: indica si el tweet es una respuesta a otro.
- respuesta_screen_name: nombre del usuario al que se responde.
- hastag_presence: indica la presencia de "hastags" en el tweet.
- url_presence: indica la presencia de URLs en el tweet.
- media_type: indica si el tweet contiene imagenes o videos.
- mentions_presence: indica la presencia de la mención a algún usuario en el tweet.
- verified: indica si el usuario que publica el tweet es verificado por Twitter.

Al igual que en el caso de los atributos numéricos, el uso de algunos de estos atributos categóricos intentan recoger el contexto en el que se publica

el mensaje, por ejemplo, el campo “verified” mide la influencia del usuario y los atributos “url_presence” o “media_type” indica si el mensaje comparte otro contenido distinto a su texto.

En los atributos categóricos únicamente se aplica una transformación para convertir esta información a tipo numérico. En este caso se emplea la codificación “one-hot” que crea un nuevo atributo por cada valor del atributo categórico asignando 1 ó 0 según la existencia o no de ese valor para cada registro.

4.2. Unión de atributos

La primera fase del sistema de clasificación lleva a cabo diferentes acciones relacionadas con el preprocesado. Se realizan tareas tan importantes como la división del texto en tokens que permitirá que la información lingüística sea tratada por las sucesivas etapas del sistema de clasificación. Como se introdujo, el tipo de preprocesado depende del tipo de atributo considerado.

4.3. Clasificación

La primera fase del sistema de clasificación lleva a cabo diferentes acciones relacionadas con el preprocesado. Se realizan tareas tan importantes como la división del texto en tokens que permitirá que la información lingüística sea tratada por las sucesivas etapas del sistema de clasificación. Como se introdujo, el tipo de preprocesado depende del tipo de atributo considerado.

GridSearch para 3 métodos de clasificación + baseline

Capítulo 5

Evaluación

Explicar proceso de evaluación con digramas: ShuffleSplit + Crossvalidation

Este capítulo describe la metodología utilizada para evaluar el sistema/método o caso de estudio propuesto, a la vez que presenta los resultados obtenidos en la evaluación de las diferentes tareas y sobre colecciones de evaluación de distintos dominios. Algunos ejemplos de secciones pueden ser estos:

5.1. Metodología de evaluación

5.2. Métricas de evaluación

5.3. Colecciones de evaluación

5.4. Resultados

Capítulo 6

Discusión

Este capítulo analiza y discute en profundidad los resultados obtenidos en la evaluación presentada en el capítulo anterior. La estructura de este capítulo dependerá del tema del trabajo y de la estructura del capítulo anterior.

6.1. Ejemplo sección

6.1.1. Ejemplo subsección

Capítulo 7

Conclusiones y trabajo futuro

Este capítulo recopila las diferentes conclusiones extraídas del trabajo realizado, y propone algunas líneas de trabajo futuro. Las siguientes secciones son las que suelen contener este tipo de capítulos, aunque pueden variar dependiendo del tema del trabajo.

7.1. Conclusiones

7.2. Trabajo futuro

Bibliografía

Bibliografía

[Aranbarri2014] Aranbarri, Garazi Urdangarin. 2014. Cosificación de las adolescentes en las redes sociales digitales, pág 43. Master's thesis, Universidad del País Vasco.

[Atlantic2016] Atlantic, The. 2016. https://www.theatlantic.com/technology/archive/2016/07/twitter-swings-the-mighty-ban-hammer/492209/.

[Canós2018] Canós, Jose Sebastián. 2018. Misogyny identification through svm at ibereval 2018. En *IberEval 2018*.

[De Mauro2016] De Mauro, T. 2016. Le parole per ferire. En *Internazionale (2016)*.

[Despoina Chatzakouy2017] Despoina Chatzakouy, Nicolas Kourtellisz, Jeremy Blackburnz Emiliano De Cristofaro Gianluca Stringhini Athena Vakaliy. 2017. Mean birds: Detecting aggression and bullying on twitter. En *WebSci*.

[Duggan2017] Duggan, M. 2017. Online harassment 2017. En *Pew Research Center, July 2017*.

[E. Fersini y Anzovino2018a] E. Fersini, P. Rosso y M. Anzovino. 2018a. Overview of the task on automatic misogyny identication at ibereval 2018. En *Proceedings of the Third Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2018)*.

[E. Fersini y Anzovino2018b] E. Fersini, P. Rosso y M. Anzovino. 2018b. Overview of the task on automatic misogyny identification at ibereval 2018. En *IberEval 2018*.

[Elena Shushkevich2018] Elena Shushkevich, John Cardiff. 2018. Classifying misogynistic tweets using a blended model: The ami shared task in ibereval 2018. En *IberEval 2018*.

[Endang Wahyu Pamungkas y Patti2018] Endang Wahyu Pamungkas, Alessandra Teresa Cignarella, Valerio Basile y Viviana Patti. 2018. Exploiting lexical knowledge for detecting misogyny in english and spanish tweets. En *IberEval2018*.

[F. Pedregosa2011] F. Pedregosa, G. Varoquaux, A. Gramfort. 2011. Scikit-learn: Machine learning in python. En *JMLR*.

[Fulper y Rowe2014] Fulper, Rachael, Giovanni Luca Ciampaglia Emilio Ferrara Y. Ahn Alessandro Flammini Filippo Menczer Bryce Lewis y Kehontas Rowe. 2014. Misogynistic language on twitter and sexual violence. En *In Proceedings of the ACM Web Science Workshop on Computational Approaches to Social Modeling (ChASM)*.

[Georgios K. Pitsilis y Langseth2018] Georgios K. Pitsilis, Heri Ramampiaro y Helge Langseth. 2018. Detecting oensive language in tweets using deep learning. En *Department of Computer Science Norwegian University of Science and Technology*.

[Giraldo1972] Giraldo, Octavio. 1972. El machismo como fenómeno psicocultural. En *Revista Latinoamericana de Psicología*.

[Goenaga y Perez2018] Goenaga, A. Atutxa, K. Gojenola A. Casillas A. Daz de Ilarraza N. Ezeiza M. Oronoz A. Perez y O. Perez. 2018. Automatic misogyny identification using neural networks. En *IberEval 2018*.

[Guang Xiang2012] Guang Xiang, Bin Fan, Ling Wang Jason I. Hong Carolyn P. Rose. 2012. Detecting offensive tweets via topical feature discovery over a large scale twitter corpus. En *CIKM, 2012, Maui, HI, USA*.

[HAJIME WATANABE2018] HAJIME WATANABE, MONDHER BOUAZIZI, TOMOAKI OHTSUKI. 2018. Hate speech on twitter: A pragmatic approach to collect hateful and offensive expressions and perform hate speech detection. En *2018 IEEE*.

[Han Liu y Cocea2018] Han Liu, Fatima Chiroma y Mihaela Cocea. 2018. Identification and classication of misogynoustweets using multi-classier fusion. En *IberEval 2018*.

[Helena Gómez-Adorno2016] Helena Gómez-Adorno, Ilia Markov, Grigori Sidorov. 2016. Compilación de un lexicón de redes sociales para la identificación de perfiles de autor. En *Centro de Investigación en Computación, México*.

[https://amiibereval2018.wordpress.com/2018] https://amiibereval2018.wordpress.com/. 2018. Automatic misogyny identification, ibereval 2018.

[International2017] International, Amnesty. 2017. Toxic twitter - a toxic place for women. En *Amnesty International Research*.

[Kearney2018] Kearney, Michael W., 2018. *rtweet: Collecting Twitter Data*. R package version 0.6.7.

[Maria Anzovino y Rosso2018] Maria Anzovino, Elisabetta Fersini y Paolo Rosso. 2018. Automatic identification and classification of misogynistic language on twitter. En *Springer International Publishing AG, part of Springer Nature 2018*.

[Nina-Alcocer2018] Nina-Alcocer, Victor. 2018. Ami at ibereval2018 automatic misogyny identification in spanish and english tweets. En *IberEval 2018*.

[NLTK2018] NLTK. 2018. https://www.nltk.org/.

[Park y Fung2017] Park, Ji Ho y Pascale Fung. 2017. One-step and two-step classification for abusive language detection on twitter. En *Proceedings of the First Workshop on Abusive Language Online*.

[Pinkesh Badjatiya2017] Pinkesh Badjatiya, Shashank Gupta, Manish Gupta Vasudeva Varma. 2017. Deep learning for hate speech detection in tweets. En *2017 International World Wide Web Conference Committee (IW3C2)*.

[Porter1980] Porter, M. 1980. An algorithm for suffix stripping.

[Quartz2017] Quartz. 2017. https://qz.com/1101455/facebook-fb-is-hiring-more-people-to-moderate-content-than-twitter-twtr-has-at-its-entire-company/.

[RAEa] RAE, Definición Machismo. http://dle.rae.es/srv/search?m=30&w=machismo.

[RAEb] RAE, Definición Misoginia. http://lema.rae.es/dpd/srv/search?key=misoginia.

[Research2018] Research, Synergy. 2018. https://www.srgresearch.com/articles/aws-leading-public-cloud-market-all-major-regions.

[Resham Ahluwalia y Cock2018] Resham Ahluwalia, Evgeniia Shcherbinina, Edward Callow Anderson Nascimento1 y Martine De Cock. 2018. Detecting misogynous tweets. En *IberEval 2018*.

[Simona Frenda y y Gomez2018] Simona Frenda, Bilal Ghanem y Manuel Montes y Gomez. 2018. Exploration of misogyny in spanish and english tweets. En *IberEval 2018*.

[Steven Bird y Loper2009] Steven Bird, Ewan Klein y Edward Loper. 2009. *Natural Language Processing with Python*. O'REILLY.

[Steven Zimmerman2018] Steven Zimmerman, Chris Fox, Udo Kruschwitz. 2018. Improving hate speech detection with deep learning ensembles. En *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

[Thomas Davidson2017] Thomas Davidson, Dana Warmsley, Michael Macy Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. En *ICWSM 2017*.

[Twitter2018] Twitter. 2018. https://help.twitter.com/en/rules-and-policies/hateful-conduct-policy.

[Waseem2016] Waseem, Zeerak. 2016. Are you a racist or am i seeing things? annotator influence on hate speech detection on twitter. En *Proceedings of 2016 EMNLP Workshop on Natural Language Processing and Computational Science*.

[Zeerak Waseem2016] Zeerak Waseem, Dirk Hovy. 2016. Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter. En *Proceedings of NAACL-HLT 2016*, páginas 33–41.

Apéndice A

Publicaciones

Publicaciones derivadas del trabajo realizado.

Todo list

v, Completar cuando se avance en el trabajo

2, Completar cuando se avance en el trabajo

8, Se puede ampliar explicación de cada método

20, Explicar con más detalle los módulos que utilice en el trabajo

27, Incluir emojis en latex

33, Explicar proceso de evaluación con digramas: ShuffleSplit + Crossvalidation