Grado en Ingeniería en Tecnologías de Telecomunicación

Trabajo de fin de grado

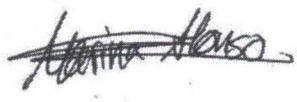
Análisis de denuncias de acoso mediante la aplicación de técnicas de procesamiento del lenguaje natural para detectar la intervención de testigos

Autora:
Marina Alonso Parra

Supervisado por:
Rafael Palacios
Cristina Puente

Madrid
Julio 2020

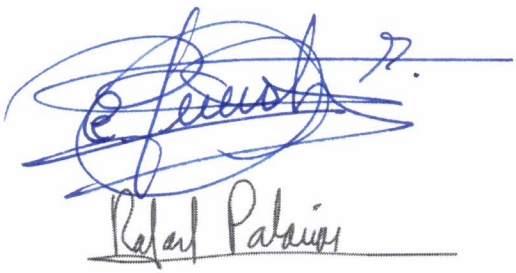
Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título: Análisis de denuncias de acoso mediante la aplicación de técnicas de procesamiento del lenguaje natural para detectar la intervención de testigos en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el curso académico 2019-2020 es de mi autoría, original e inédito y no ha sido presentado con anterioridad a otros efectos. El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.



Fdo.: Marina Alonso Parra. Fecha: 3/6/2020

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO



Fdo.: Cristina Puente Águeda Fecha: …3 / JUN / 2020

Rafael Palacios Hielscher

Grado en Ingeniería en Tecnologías de Telecomunicación

Trabajo de fin de grado

Análisis de denuncias de acoso mediante la aplicación de técnicas de procesamiento del lenguaje natural para detectar la intervención de testigos

Autora:
Marina Alonso Parra

Supervisado por:
Rafael Palacios
Cristina Puente

Madrid
Julio 2020

ANÁLISIS DE DENUNCIAS DE ACOSO MEDIANTE LA APLICACIÓN DE TÉCNICAS DE PROCESAMIENTO DEL LENGUAJE NATURAL PARA DETECTAR LA INTERVENCIÓN DE TESTIGOS

Autor: Alonso Parra, Marina.

Director: Palacios Hielscher, Rafael
Puente Agueda, Cristina

Entidad Colaboradora: SoGooData, Hollaback

RESUMEN DEL PROYECTO

En este proyecto se analizaron las descripciones de situaciones de acoso provenientes de la base de datos de Hollaback, a través de herramientas de procesamiento del lenguaje natural. Además, se elaboraron modelos de clasificación de texto con aprendizaje automático para separar las descripciones en aquellas que tienen presencia de testigos y las que no la tienen.

Palabras clave: Procesamiento de Lenguaje Natural, clasificación de textos, aprendizaje automático, denuncias de acoso, presencia de testigos, Python.

1. Introducción

Este proyecto está lanzado por la ONG de tratamiento de datos, SoGooData y por la ONG Hollaback, que tiene el fin de dar apoyo, visibilizar y conseguir eliminar las situaciones de acoso en lugares públicos. El proyecto consiste en el análisis, por medio de herramientas de procesamiento de lenguaje natural, de las descripciones de su base de datos para elaborar un modelo de clasificación de texto que diferencie las descripciones con presencia de testigo de las que no lo tienen.

El proyecto se encuentra enmarcado en un contexto en el que, según un estudio estadounidense realizado en 2019, el 81% de las mujeres y el 43% de los hombres afirman haber sufrido acoso sexual [1]. Estas experiencias tienen normalmente un impacto negativo en la víctima, generando sentimientos de miedo, vergüenza, enfado... Pudiendo hasta causar traumas e inseguridades para el resto de sus vidas. Sin embargo, se ha estudiado que hay algunos factores que alteran este impacto psicológico, entre ellos la presencia de testigos. Por tanto, por medio de este proyecto se quiso profundizar más en el efecto de este factor.

2. Definición del proyecto

El Proyecto se desarrolló en tres partes. La primera consistió en un análisis exploratorio de los datos de la base, para obtener estadísticas en la distribución temporal y espacial de las descripciones, los idiomas de las descripciones, el tipo de

acoso… En la segunda parte se realizó el procesamiento de las descripciones con herramientas de NLP (limpieza y normalización del texto, reducción de dimensionalidad, Word embeding…) y por último la tercera parte trató la elaboración de un modelo de clasificación con herramientas de aprendizaje supervisado.

3. Descripción del modelo

El modelo consistió en un algoritmo que importa los datos de la base de datos, realiza el procesado de las descripciones, y las utiliza finalmente para entrenar y hacer el test de distintos modelos de clasificación de texto. Algunas de las herramientas utilizadas durante el procesado son la limpieza y normalización del texto, tokenización, tratamiento del idioma, LSA, LDA, BoW, TF-IDF y word2vec. Los modelos utilizados son regresión logística, naive bayes, máquinas de vector de soporte, k vecinos más próximos y un clasificador SGD. Además, estos modelos se entrenaron y probaron con 4 combinaciones diferentes de datos: descripciones completas, descripciones simplificadas, descripciones simplificadas con test completo y descripciones mixtas; para comprobar los distintos resultados.

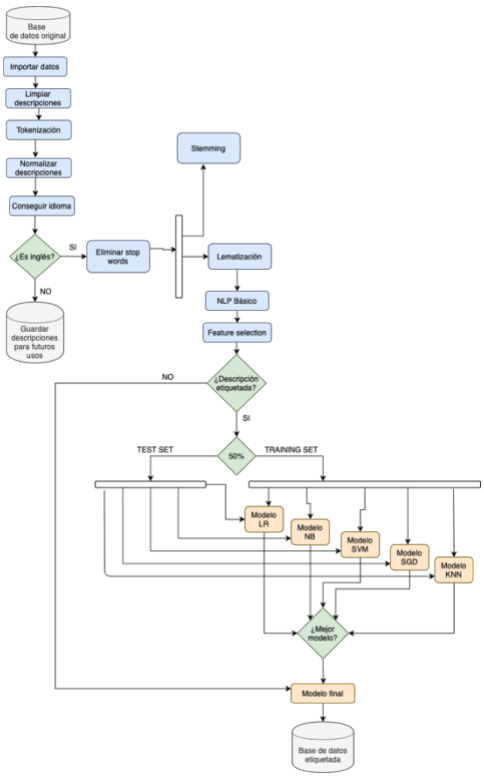


Ilustración 1 – Diagrama de flujo del algoritmo

4. Resultados

Durante el análisis exploratorio de datos se quiso ver el alcance mundial de la plataforma Hollaback. Se representaron las denuncias en un mapa y se comprobó que el número de incidentes reales y el de denuncias no estaban fuertemente correlacionados. Las denuncias se encontraban más concentradas en Norte América y Europa, lugares donde la plataforma es más conocida, y menos concentradas en el resto del mundo. Esta distribución se puede ver en la figura 2.

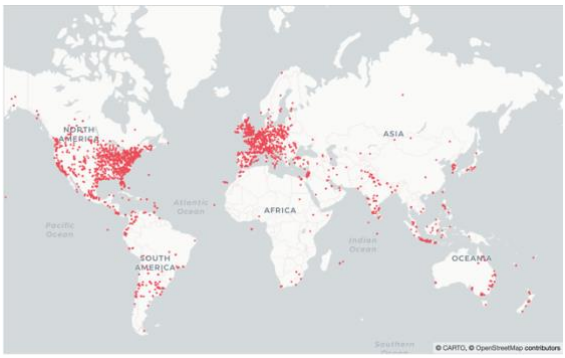


Ilustración 2– Distribución mundial de las denuncias

Durante la creación de los modelos de clasificación, tras probar los distintos algoritmos y los distintos conjuntos de datos, se llegó a una serie de resultados basados en las métricas. En los modelos entrenados con descripciones completas, se comprobó que el modelo más eficiente es el de la máquina de vector de soporte. Tuvo para la categoría "con testigo" una exactitud del 72%, una precisión del 61% y una sensibilidad del 47%, en comparación a otros modelos como la regresión logística que tuvo una precisión del 100% pero sólo 17% de sensibilidad.

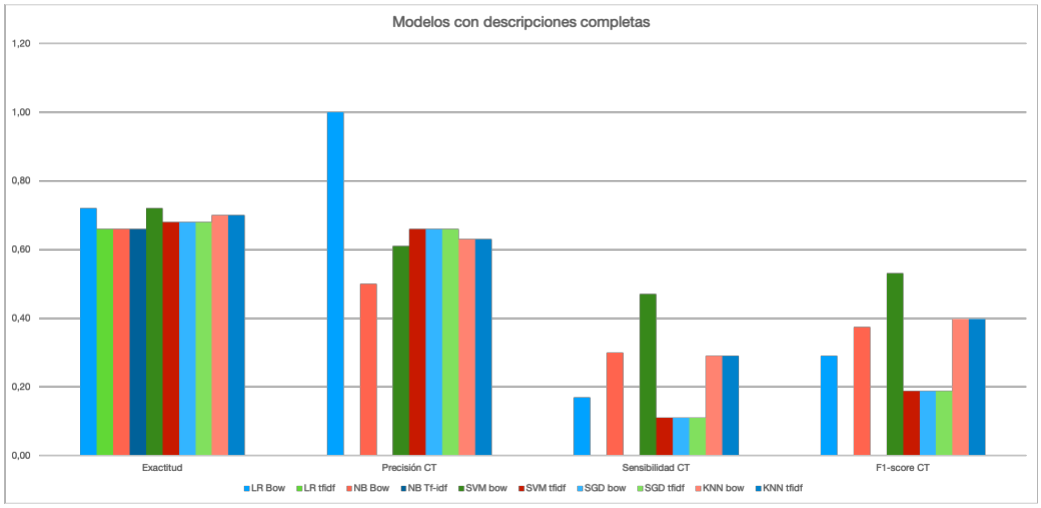


Ilustración 3 – Métricas de los modelos con descripciones completas

En los modelos entrenados con descripciones simplificadas, se pudo ver que, excepto el clasificador SDG, todos los modelos obtuvieron buenos resultados, con una exactitud del 69%, una precisión del 67% y una sensibilidad del 89%.

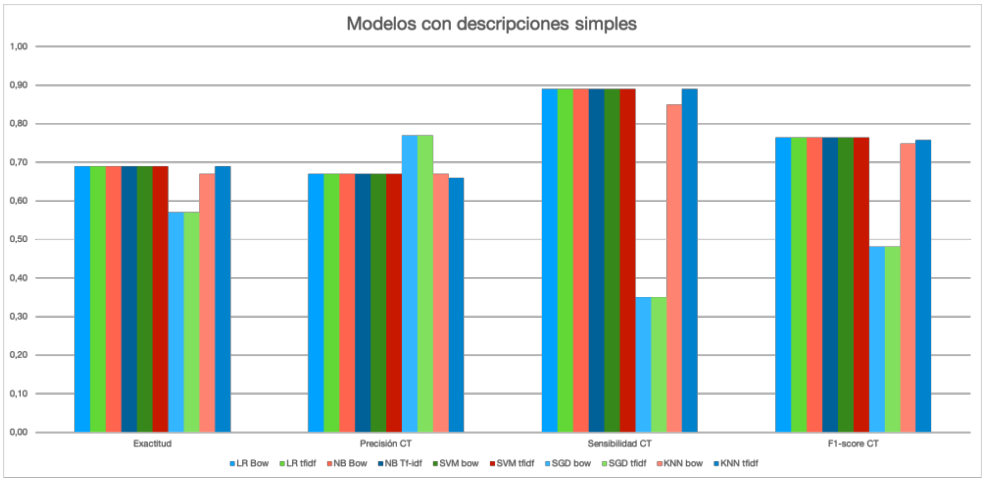


Ilustración 4 – Métricas de los modelos con descripciones simplificadas

Cuando se aplicó a los modelos entrenados con descripciones simplificadas y descripciones completas de test, todos los modelos aumentaron su sensibilidad al 95% y, sin embargo, la precisión disminuyó al 45%.

Finalmente, entrenando los modelos simultáneamente con descripciones simplificadas y descripciones completas, y probándolos con descripciones completas, se comprobó que tanto la precisión como la sensibilidad disminuyen en gran medida; por lo que no se considerarían buenos modelos.

5. Conclusiones

En conclusión, se pudo ver que para este proyecto el mejor clasificador es el SVM entrenado con descripciones completas, ya que fue el que de verdad permitió clasificar los datos de la base actual. Si se quisiese mejorar la precisión se podrían introducir más descripciones etiquetadas para el entrenamiento, pero realizar el etiquetado requiere tiempo y trabajo.

Los modelos con descripciones simplificadas dieron buenos resultados con descripciones simplificadas, pero no con completas; por tanto, no valdrían para solucionar la problemática actual. Haría falta encontrar un modo de relacionar estos dos tipos de descripciones para poder introducirlas en el modelo y que mantenga las métricas.

6. Referencias

[1] USCD center on gender equality and health. “A national study on sexual harassment.” 2019

[2] Bird, S; Klein, E; Loper, E. “Natural Language Processing in Python”, 2007

[3] Scikit learn. “API Reference” https://scikit-learn.org/stable/modules/ classes.html

[4] NLTK 3.5. “Documentation”. https://www.nltk.org

ANALYSIS OF HARASSMENT COMPLAINTS USING NATURAL LANGUAGE PROCESSING TO DETECT WITNESS' INTERVENTION

Author: Alonso Parra, Marina.

Supervisor: Palacios Hielscher, Rafael
Puente Agueda, Cristina

Collaborating entity: SoGooData, Hollaback

ABSTRACT

This project aimed to analyze the descriptions of harassment situations coming from the Hollaback database through Natural Language Processing tools. In addition, classification models with machine learning were developed to classify the descriptions into those that have bystander's presence and those that do not.

Key words: Natural Language Processing, text classification, machine learning, harassment complaints, bystander's presence, Python.

1. Introduction

This project was proposed by the data processing NGO, SoGooData, and the NGO Hollaback, that aims to support, make visible, and eliminate situations of harassment in public places. The project consists of the analysis, with Natural Language Processing tools, of the descriptions in its database; in order to develop a classification model that differentiates descriptions with the presence of a bystander from those without.

The project is placed in a context where, according to an American study conducted in 2019, 81% of women and 43% of men [1] claim to have suffered sexual harassment. These experiences usually have a negative impact on the victim, generating feelings of fear, shame and anger. It can even cause trauma and insecurity for the rest of their lives. However, it has been studied that there are some factors that modify this psychological impact, among them the presence of bystanders. Therefore, through this project it was aimed to go deeper into the effect of this factor.

2. Project definition

The Project was developed in three parts. The first part consisted of an exploratory analysis of the data in the database, to obtain statistics on the temporal and spatial distribution of the descriptions, the languages of the descriptions, the type of harassment... The second part took care of the processing of the descriptions with NLP tools (cleaning and standardization of the text, reduction of dimensionality,

Word embeding...) and finally, the third part consisted of the development of a classification model with supervised learning tools.

3. Model description

The model consisted of an algorithm that imports the data from the database, performs the processing of descriptions, and then uses them to train and test different text classification models. Some of the tools used during processing are text cleaning and normalization, tokenization, language treatment, LSA, LDA, BoW, TF-IDF and word2vec. The models used are logistic regression, naïve bayes, support vector machines, k nearest neighbors and an SGD classifier. In addition, these models were trained and tested with 4 different combinations of data: full descriptions, simplified descriptions, simplified descriptions with full descriptions for test and mixed descriptions; to check the different results.

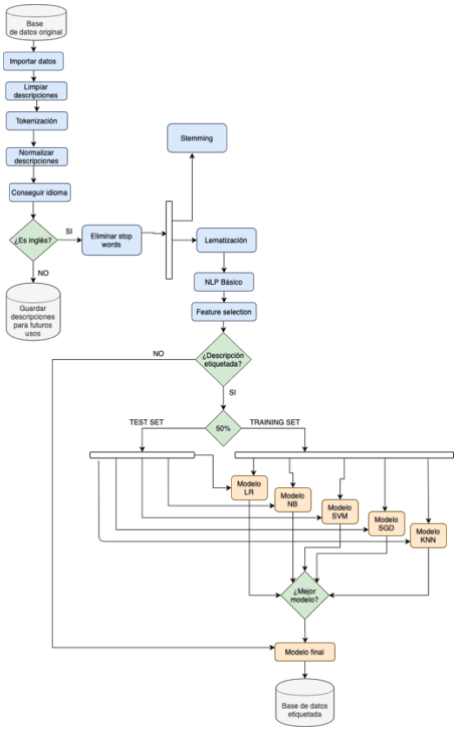


Figure 2 – Algorithm's flux diagram

4. Results

The exploratory data analysis was performed to see the global reach of the Hollaback platform. The complaints were represented on a map and it was found that the number of actual incidents and the number of complaints were not strongly related. The reports were more concentrated in North America and Europe, where the platform is best known, and less concentrated in the rest of the world. This distribution can be seen in figure 2.

Figure 2 – Distribution of complaints

During the creation of the classification models, after testing different algorithms for different data sets, a series of results could be seen based on the metrics. In the models trained with complete descriptions, it was seen that the most efficient model was the support vector machine. It had for the category "with bystander", 72% accuracy, 61% precision and 47% sensitivity, compared to other models such as the logistic regression which had 100% precision but only 17% sensitivity.

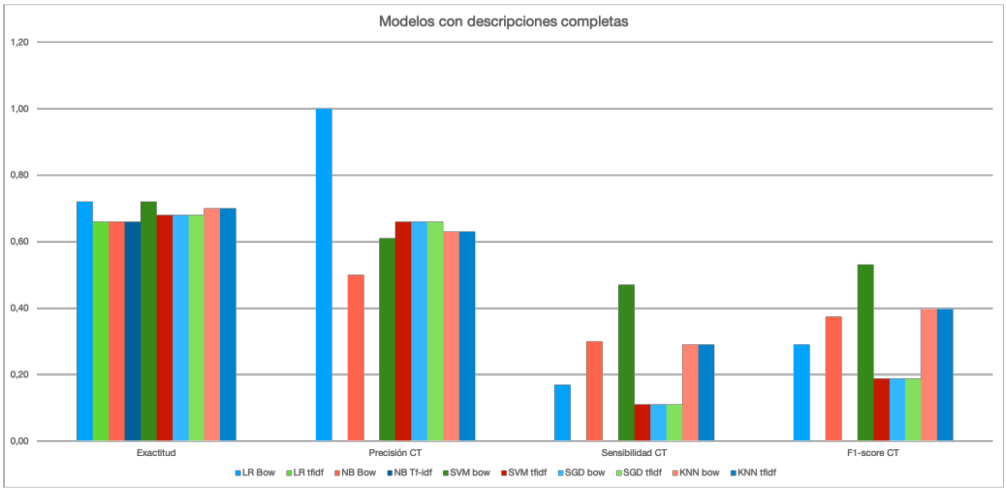


Figure 4 – Metrics for models built with whole descriptions

In the models trained with simplified descriptions, it could be seen that except for the SDG classifier, all the models obtained good results, with 69% accuracy, 67% precision and 89% sensitivity.

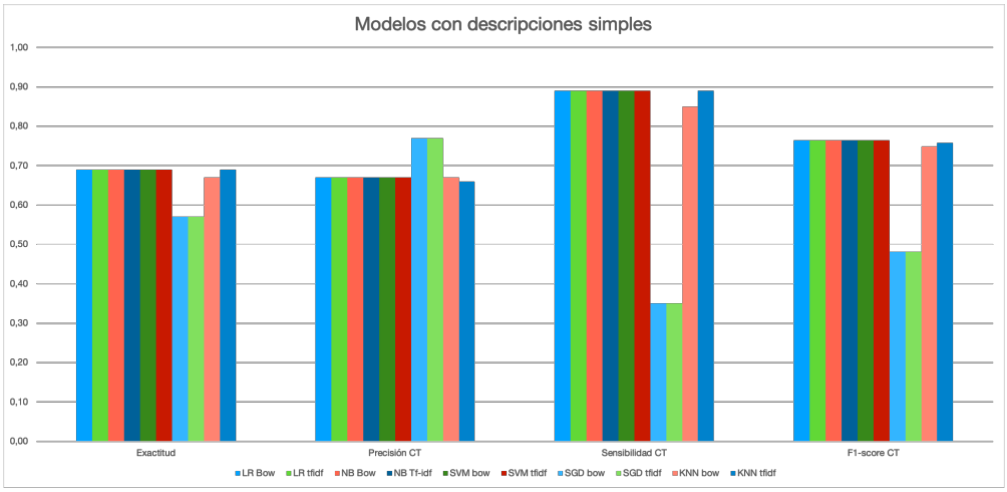


Figure 4 – Metrics for models built with simplified descriptions

When full descriptions were used to test models trained with simplified descriptions, all models increased their sensitivity to 95% and yet the accuracy decreased to 45%.

Finally, training models with simplified descriptions and full descriptions and testing them with full descriptions showed that both accuracy and sensitivity decrease greatly; therefore, they would not be considered good models.

5. Conclusions

In conclusion, it could be seen that for this project the best classifier is the SVM trained with complete descriptions, since it was the one that really allowed to classify the data of the current base. If precision wanted to be improved, it would be necessary to introduce more tagged descriptions for training, but doing the tagging requires time and work. The models with simplified descriptions gave good results with simplified descriptions, but not with complete ones, therefore they would not be valid for the current problem. It would be necessary to find a way to relate these two types of descriptions to be able to introduce them in the model and to maintain the metrics.

6. References

[1] USCD center on gender equality and health. "A national study on sexual harassment." 2019

[2] Bird, S; Klein, E; Loper, E. "Natural Language Processing in Python", 2007

[3] Scikit learn. "API Reference" https://scikit-learn.org/stable/modules/ classes.html

[4] NLTK 3.5. "Documentation". https://www.nltk.org

A mi familia, mis amigos y Jorge.

Agradecimientos

Primero, quiero agradecer a mis padres la oportunidad que me han dado de estudiar tanto en ICAI como en CentraleSupélec, y por apoyarme siempre con todas mis decisiones y animarme a conseguir todo lo que me hiciese feliz. También agradezco a ICAI la oportunidad que me ha dado de hacer un doble diploma en CentraleSupélec y de seguir desarrollándome tanto profesional como personalmente.

En segundo lugar, quiero darle las gracias a Jorge por haber estado siempre conmigo, en los buenos y malos momentos. Has sabido apoyarme, motivarme y demostrarme que juntos todo es más fácil. Francia no hubiese sido lo mismo sin ti y no puedo esperar a ver cual será nuestra siguiente aventura.

También a mis amigos de Madrid porque a pesar de la distancia, siempre he podido contar con ellos; y a mis amigos de Francia por haber hecho de estos dos años una experiencia increíble.

Finalmente, quiero darle las gracias a mis directores de proyecto Rafael y Cristina, y a Ana y Priscilla, por haberme dado la oportunidad de hacer este proyecto y haberme guiado durante estos meses.

Índice general

Índice de figuras

1. Introducción

En la actualidad, la mayor parte de mujeres y muchos hombres sufren alguna experiencia de acoso sexual a lo largo de sus vidas. Durante muchos años estas situaciones no han sido denunciadas, pero en 2018 hubo un gran movimiento mediático en el que salieron a la luz escándalos de acoso sexual por parte de directores de cine, grandes ejecutivos, fotógrafos... Este hecho desencadenó un importante movimiento en las redes sociales llamado *#MeToo*, donde se manifestaron multitud de personas que habían sufrido acoso en sus vidas.

La magnitud de este movimiento hizo que en 2019 se realizase un estudio estadounidense con el mismo nombre, que desveló que el 81 % de las mujeres afirma haber sufrido acoso sexual, cifra que desciende al 43 % en el caso de los hombres.[1].

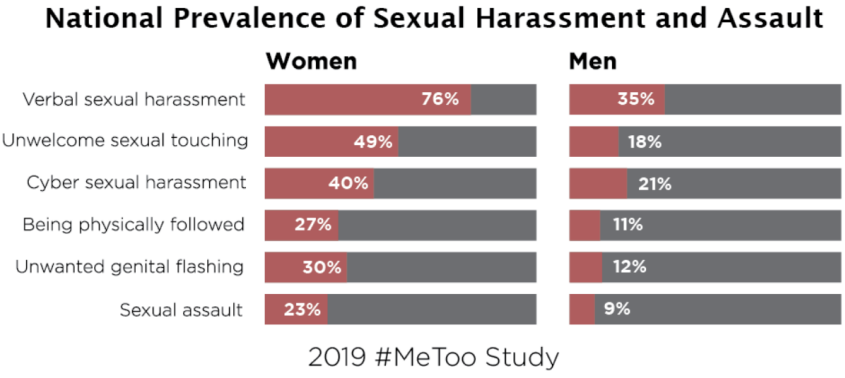


Figura 1.1: National prevalence of sexual harassment and assault[2]

Estas experiencias tienen normalmente un impacto negativo en la víctima, generando sentimientos de miedo, vergüenza, enfado... Pudiendo hasta causar traumas e inseguridades para el resto de sus vidas. Por ello este tipo de conductas pueden llegar a considerarse como violencia de género y por tanto violaciones de los derechos humanos. Al mismo tiempo están en conflicto directo con los Objetivos de

Desarrollo Sostenible de la ONU donde se defiende la igualdad de genero (objetivo 5) y las sociedades justas y seguras (objetivo 16).[3]

Debido al gran impacto que tienen estas situaciones en la sociedad, en los últimos años han ganado protagonismo múltiples asociaciones que buscan combatirlo. Entre ellas se encuentra *Hollaback!* que es una ONG fundada en 2005 en Nueva York y que tiene el fin de dar apoyo, visibilizar y conseguir eliminar las situaciones de acoso en lugares públicos. Consiste en una plataforma donde las víctimas o testigos de situaciones de acoso, pueden registrarse y publicar sus propias experiencias. Los testimonios están acompañados de la geolocalización del incidente, la fecha y la hora, el tipo de acoso, una descripción sobre lo sucedido y pueden incluso subir una foto o descripción del agresor. Luego estas experiencias se publican en la página web, así otras víctimas pueden leerlas y sentirse acompañadas, reforzadas por gente que les plantó cara o estar atentas a un incidente en la misma zona; en resumen, se sienten dentro de una comunidad [4].

Durante los últimos años se ha estado analizando el efecto psicológico de estos incidentes en las víctimas y viendo los numerosos factores que pueden modificarlo. En el caso del acoso callejero, se ha comprobado que la intervención de un testigo es un factor de gran importancia a la hora de disminuir o incrementar el efecto psicológico en la víctima. Esto ha llevado a algunas ONGs a lanzar distintas iniciativas para concienciar sobre los comportamientos que deberían tener los testigos en caso de presenciar una escena de acoso. Un buen ejemplo sería la guía de actuación del testigo, desarrollada por CUP y Hollaback. [5]

Todavía hay mucha investigación que hacer en el campo de la intervención de los testigos, y gracias a las nuevas tecnologías se puede obtener una perspectiva diferente. Este proyecto lanzado por Sogoodata, ONG de análisis de datos, y Hollaback; pretende ayudar a comprender del impacto que tienen los testigos través del análisis de las descripciones la base de datos de Hollaback.

Para ello se hará en un primer paso un análisis exploratorio de toda la base y luego se aplicarán herramientas de procesamiento de lenguaje natural para tratar las distintas descripciones hechas por la víctima. El objetivo es primeramente hacer un modelo que permita una clasificación de las descripciones entre las que tienen intervención de un testigo y las que no. Y en un segundo paso, que queda fuera del alcance de este TFG, aplicar técnicas de análisis de sentimiento sobre las descripciones etiquetadas como testigo. De este modo se obtendrán estadísticas sobre que actitudes del testigo son las más beneficiosas y cuáles las más perjudiciales; para que en un futuro todos sepamos como actuar si presenciamos una situación de acoso.

2. Estado del Arte

Este capítulo se utilizará para poner en contexto el procesamiento del lenguaje natural como parte de la Inteligencia Artificial. Por tanto, se hablará de lo que es, de sus niveles, sus usos generales y la importancia que tiene en la industria. También se tratará más concretamente la clasificación de textos, los sistemas de clasificación y sus usos.

2.1. Procesamiento del lenguaje natural

El procesamiento de lenguaje natural, a partir de ahora NLP por sus siglas en inglés (Natural Language Processing), es una rama de la Inteligencia Artificial que permite la interpretación, comprensión y manipulación del lenguaje natural. [6] Es un área que relaciona directamente la lingüística computacional y la informática. Ya que, por un lado, la informática aporta el lenguaje de programación, los algoritmos de optimización y la implementación de los sistemas. Y por otro lado la lingüística aporta las reglas fonéticas, morfológicas y gramaticales que permiten la comprensión del texto.

La principal diferencia entre el NLP y otro tipo de ramas de la informática es que las entradas en vez de ser en lenguaje máquina, son en lenguaje natural. Este es tratado en base a unas reglas léxicas y gramaticales y produce unas salidas también en lenguaje natural. [7]

2.1.1. Historia

El NLP empezó entre los años 40 y 60, con la intención de desarrollar herramientas de traducción automática. Estas primeras herramientas lograron una traducción palabra por palabra, sin embargo, se perdía el componente sintáctico

y a menudo había ambigüedades. Por ello en los años 60 se trabajó en técnicas de análisis sintáctico para conseguir un mayor contexto. En los setenta se logró desarrollar distintas herramientas de preguntas-respuestas, el mejor ejemplo es Eliza que recogía respuestas de clientes y elaboraba preguntas en modo entrevista. Entre los años setenta y ochenta aumenta la investigación en programación lógica y se desarrollan proyectos de traducción plurilingüe como Eurotra. Finalmente, en los últimos años, las investigaciones se han centrado en el tratamiento del lenguaje de modo estadístico y con técnicas de inteligencia artificial; así como en sistemas de recuperación de información para textos complejos. [8]

Figura 2.1: Chatbot Eliza [9]

2.1.2. Niveles del estudio del lenguaje

El procesamiento del lenguaje natural se estructura en diferentes niveles: [8]

Nivel fonético: este nivel solo se usa en NLP en aquellos casos en los que la entrada es por voz. En él se hace un análisis de los sonidos y sus representaciones escritas (fonemas). De este modo se convierte una frase de voz en un texto que pueda ser tratado.

Nivel morfológico: está estrechamente relacionado con el léxico de las palabras. El léxico es la información que el ordenador utiliza para procesar cada término y contiene detalles sobre la morfología y categoría gramatical de cada palabra. Gracias a este nivel se pueden obtener las etiquetas morfológicas y realizar la lematización de las palabras.

Nivel sintáctico: tiene como función analizar los componentes de la oración y cómo las palabras se combinan para generar sintagmas. Por ejemplo, una palabra etiquetada como determinante al lado de otra etiquetada como sustantivo, serían analizadas como un sintagma nominal.

Nivel semántico: este tipo de análisis tiene como fin obtener el significado de una frase, sin tener en cuenta el contexto. Se puede hacer copiando la estructura de los sintagmas de forma simétrica, o aplicando una serie de transformaciones a la estructura sintáctica y luego obteniendo su representación semántica.

Nivel pragmático: aporta significación adicional a la frase ya que tiene en cuenta elementos del contexto, como la información presupuesta. Marca la diferencia entre la referencia que es la relación directa entre una palabra y su significado; y la inferencia que es la relación probable entre la palabra y su significado, y está inducida por el emisor. Es el nivel más complejo de análisis. [7]

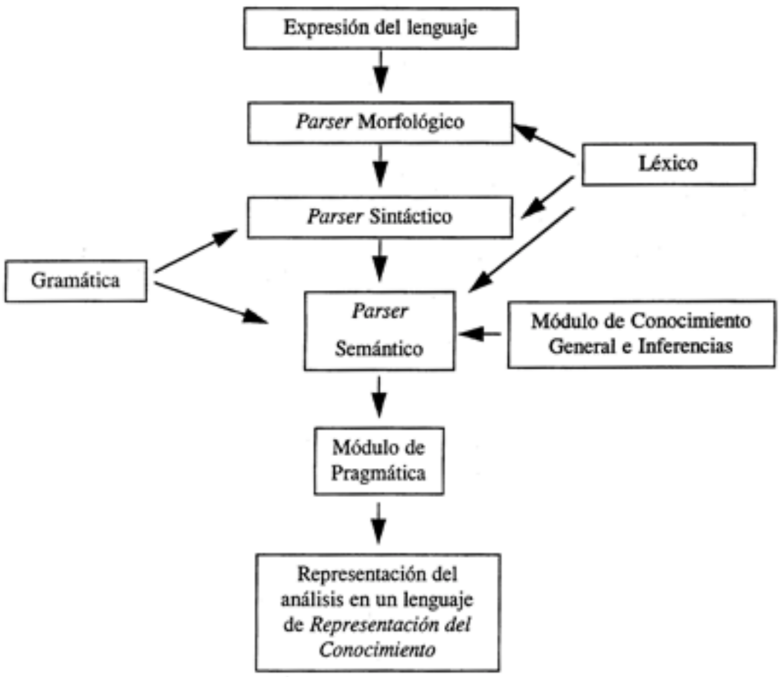


Figura 2.2: Niveles del procesamiento del lenguaje natural[8]

2.1.3. Usos generales del procesamiento de lenguaje natural

El procesamiento de lenguaje natural tiene múltiples usos en la actualidad, aquí se recogen algunos de los más importantes.

Traducción automática

Consiste en que un ordenador haga la traducción de un texto de un idioma a otro, el más claro ejemplo es Google Translator. Este proceso de traducción es complicado ya que no se basa en la traducción palabra por palabra, sino en traducción estadística donde la máquina trata de hacer un paralelismo de estructuras y contexto entre los dos idiomas para encontrar la traducción más exacta.

Reconocimiento del habla

Consiste en detectar el discurso de una persona y convertir los fonemas en texto que pueda ser interpretado por una máquina. De este modo la información y ordenes que antes se podían pasar gracias al teclado o el ratón, ahora se pueden hacer a través de comandos verbales. Esta es la base de sistemas como Alexa, Siri,Cortana...

Análisis de sentimiento

Gracias al NLP se puede identificar dentro de los textos información con alta carga subjetiva para entender la emoción de la persona que lo estaba escribiendo. Esta herramienta es fundamental, por ejemplo, para los comercios, ya que frecuentemente la gente no esta dispuesta a rellenar cuestionarios de satisfacción. Sin embargo, sí dejan comentarios y gracias a esta herramienta se puede extraer su opinión de ellos.

Preguntas y respuestas

Como ya se ha dicho, el NLP sirve para el reconocimiento del habla, como peticiones o preguntas; la acción complementaria a ese proceso es la generación de respuestas en lenguaje natural. Por ejemplo, si se le dice a Siri, ¿Qué hora es? no enseñará un reloj, sino que responderá en la misma lengua diciendo la hora.

Resumen de textos y extracción de información importante

Gracias al NLP se pueden obtener las palabras más repetidas o frases más importantes de un texto largo para hacerse una idea del tema sin leerlo entero. También se puede localizar palabras o información clave, proceso que es de gran importancia en las empresas, por ejemplo, en el departamento de recursos humanos.

Chatbots

Son una herramienta utilizada en gran medida por comercios online, permiten responder a dudas del cliente basándose en palabras claves y el contexto. Así pueden ofrecer la información más precisa acorde a la pregunta.

Correctores de gramática y ortografía

El NLP permite corregir un texto, no solo ortográficamente; sino también sintácticamente (orden en la oración, concordancias de género y número), también evita repeticiones, propone sinónimos... El mejor ejemplo de este sistema es la aplicación Grammarly.

2.1.4. NLP en la industria

El procesamiento del lenguaje natural causará cambios dramáticos en múltiples sectores, esto se debe a su capacidad para sintetizar, clasificar y detectar patrones en textos. [10]

Sector legal

A menudo los abogados realizan tareas monótonas y que consumen gran cantidad de tiempo. El NLP permite resumir textos, extraer información importante y localizar palabras claves que relacionen diferentes documentos. Así, el trabajo de lectura de un abogado queda reducido en gran medida, lo que a largo plazo permite reducir los costes.

Sector financiera

El machine learning ha ganado mucha importancia recientemente para el análisis de los mercados. En el caso del NLP, podría ser una herramienta de gran utilidad leyendo noticias e identificando hechos que el permitiesen decidir si comprar o vender. Por ejemplo, si saliese una noticia de que se ha descubierto un escándalo en las nuevas tecnologías de Apple; lo lógico sería vender porque esas acciones caerán. Gracias al NLP esto se hará de manera automática lo que evitará muchas horas de trabajo y de lectura a los trabajadores de esta industria.

Sector sanitaria

Los médicos y enfermeros manejan una gran cantidad de documentos y expedientes. Gracias al NLP se podrían encontrar nuevas correlaciones entre síntomas y enfermedades; y se podría dar más fácilmente un diagnóstico en base a todos los datos procesados previamente. Además, cada hospital o seguro médico, podría tener un chatbot que permitiese al paciente introducir sus síntomas y orientarle al mejor médico o darle un diagnóstico previo en caso de que el problema se pudiese solucionar con facilidad.

Industria comercial

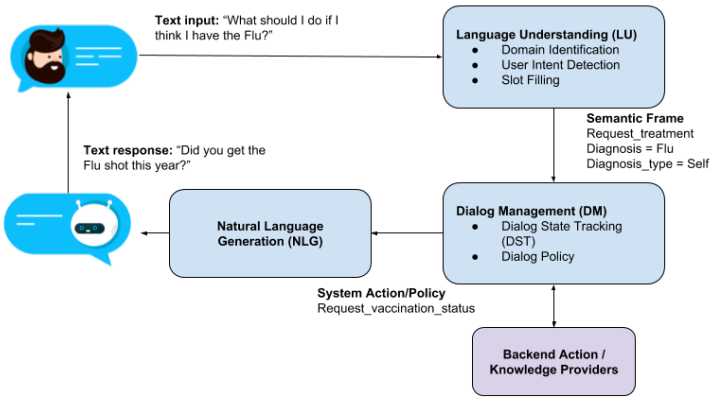


Figura 2.3: Chatbot médico [11]

EL NLP, al igual que el machine learning, tiene mucha importancia en el ámbito de los negocios. Por un lado, se puede utilizar para mejorar la calidad del servicio al consumidor, analizando el sentimiento de las críticas y modificando aquellas cosas negativas o implementando chatbots que ayuden al cliente. Por otro lado, se pueden realizar estudios de posicionamiento en el mercado; ya que gracias al NLP se pueden leer automáticamente blogs, páginas web, artículos... y ver qué se dice de la empresa. Al mismo tiempo se puede analizar la posición de la competencia y actuar en consecuencia. Finalmente, también se puede hacer publicidad dirigida a cada consumidor en función de sus búsquedas.

2.1.5. NLP para la clasificación de textos

La clasificación de textos es uno de los usos fundamentales del NLP, esta herramienta nos permite dado un texto con datos desestructurados, etiquetarlo en categorías determinadas.

2.1.5.1. Técnicas de clasificación

Sistemas basados en reglas

Estos sistemas están creados manualmente, cada predicción se basa en una serie de antecedentes o patrones. Cuando se trata de desarrollar estos sistemas, se debe crear una lista de palabras relacionadas con la categoría y ponderadas en función de su importancia. Durante el proceso de clasificación se buscarán estas palabras en el texto y se le dará la categoría que haya tenido más puntuación.

Este tipo de sistemas tienen la ventaja de ser fáciles de comprender y se pueden ir mejorando con el tiempo. Sin embargo, tienen la desventaja de que requieren un gran conocimiento de la materia y la dedicación mucho tiempo a la creación de los diccionarios, generación de reglas y a probar y analizar los resultados.

Sistemas basados en aprendizaje automático

En este tipo de sistemas en vez de tener que crear manualmente el modelo de clasificación, este se crea a través de un algoritmo de aprendizaje automático, más conocido como "machine learning".

Para crear el modelo hay una primera fase que es el aprendizaje, en el que se le introducen al algoritmo los textos vectorizados y sus respectivas categorías para que genere el modelo. En una segunda fase llamada predicción, se aplica este modelo al texto que queremos clasificar, para obtener su respectiva etiqueta.

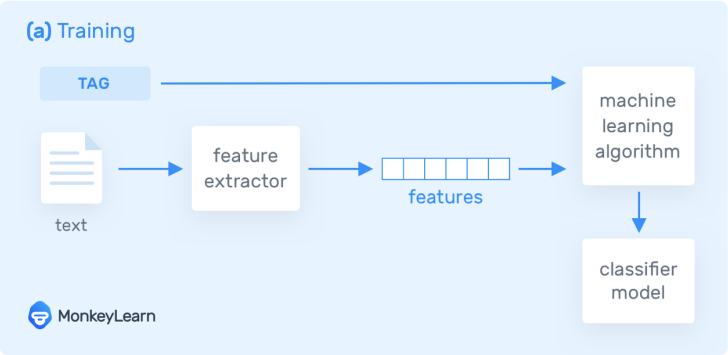


Figura 2.4: Modelo de aprendizaje machine learning[12]

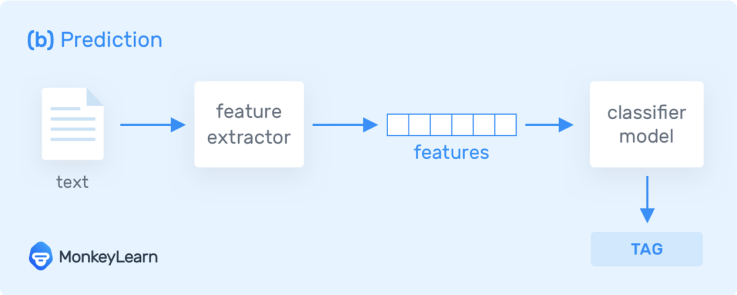


Figura 2.5: Modelo de predicción de categorías[12]

2.1.5.2. Usos de la clasificación de textos

- **Clasificación de emails:** la clasificación de emails es una herramienta implementada por todas las bandejas de email, ahorra mucho tiempo a los usuarios y les protege de peligros como virus. Cuando un email llega a la bandeja de entrada es clasificado entre spam, importante o promociones. Esto se hace en función de algunas palabras clave como pueden ser: "importante", "reunión", "urgente-para emails importantes o "gana", "promoción", "descuento- para spam.

- **Filtro de mensajes y comentarios abusivos:** la mayor parte de los foros tienen implementado este sistema para asegurar la seguridad y el respeto entre los usuarios. Cuando un comentario se clasifica como "no apto" es bloqueado; esta categorización se hace con la detección de insultos, palabras denigrantes o de discriminación hacia otros usuarios.

- **Publicidad personalizada:** la información subida a las redes sociales se puede clasificar en muchos tipos dependiendo de los gustos y estilos de vida de los usuarios. Esta clasificación puede ser usada para ofrecer publicidad personalizada. Por ejemplo, una madre podría publicar un texto en Facebook sobre la importancia de la familia y recibirá una publicidad diferente a un joven que hable de viajes y deporte.

- **Atención al cliente:** cuando un cliente hace una pregunta en el servicio de atención al cliente de una empresa, puede ser por distintos motivos (problemas con el software, con el hardware, con envios...). Este problema puede ser clasificado automáticamente para ser atendido directamente por personas especializadas, o para orientar al cliente sobre donde buscar la información deseada. Por ejemplo, si tienen un problema con la facturación de un servicio, la petición será detectada y transmitida directamente al departamento

de contabilidad.

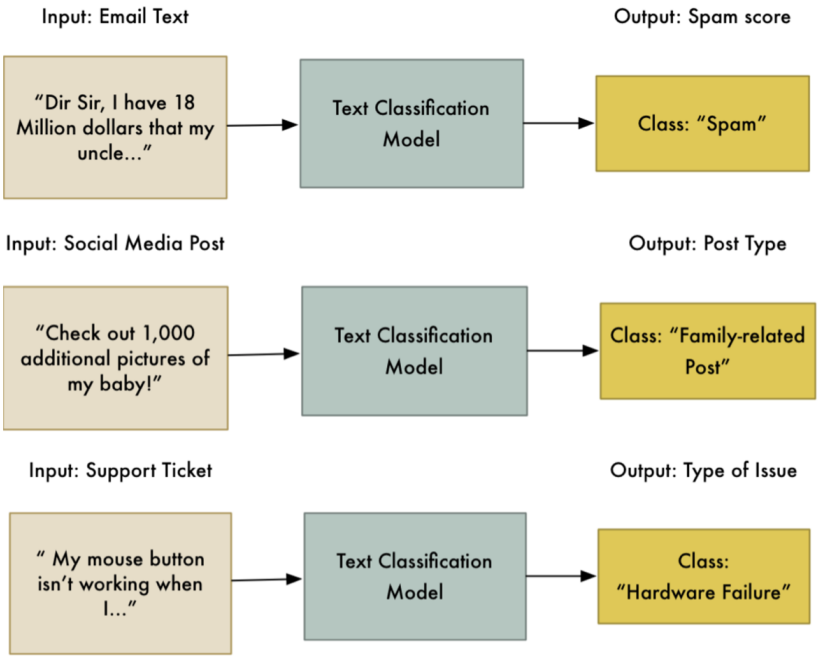


Figura 2.6: Usos clasificación de textos automática[13]

2.1.6. Retos NLP

Recuperación de la información

Este es un proceso que en la actualidad no es complicado de realizar y aun así es una parte crítica del NLP. Un algoritmo tiene que ser lo suficientemente inteligente como para recuperar la información siguiendo las estructuras básicas del texto como la separación de párrafos en frases. La principal dificultad está en recuperar la información estructurada de datos desestructurados, por ejemplo, leer la información de una tabla o un gráfico que se encuentren incluidos en el texto de una página web.

POS y grafos de dependencia

Una parte fundamental del NLP es detectar, entender y aplicar lo que se llaman POS (parts of speech) que son los sintagmas y estructuras dentro de la frase. Un

programa tiene que ser capaz de generar grafos que le permitan saber el orden de los elementos en el sintagma y la frase, la conjugación del verbo, la concordancia en género y número...

Construir vocabularios

Los algoritmos tienen que ser capaces de leer un texto y generar un vocabulario específico acorde con él. En la actualidad esto todavía genera problemas en vocabularios que utilizan términos complejos, donde sea importante el conjunto de palabras y no las palabras por separado. Por ejemplo, si estamos hablando de finanzas, los términos serían gestión de riesgos, capital social, línea de crédito... y no las palabras por separado.

Unión de componentes del vocabulario y extracción de significados semánticos

En el NLP es fundamental que un modelo entienda la relación entre los componentes de la frase y en función de ello sea capaz de extraer su significado. Esto es complejo ya que en muchas ocasiones simplemente cambiando una palabra o una coma o el orden de las palabras, puede modificarse por completo el significado de la oración. Por ejemplo: No, tengo hambre y no tengo hambre. Por ello es un gran reto extraer el significado semántico y relacionar unas palabras con otras.

Establecer el contexto

Uno de los mayores retos del NLP es detectar las ambigüedades generadas por el contexto. Este es el caso de los homónimos, por ejemplo, no sería lo mismo estoy en la planta tres de mi edificio, que la hormiga está en la planta de mi terraza. Pero también se vería ambigüedad en el uso de sarcasmo, o en frases que hacen referencia a elementos mencionados previamente en el texto.

Extraer nombres de entidades

Finalmente es indispensable para un algoritmo NLP ser capaz de diferenciar dentro de un texto los nombres de entidades del texto normal. Por ejemplo, si se está utilizando una herramienta de finanzas y se habla del Banco Santander, saber que hace referencia a una entidad y no a una ciudad. Es lo que se llama en inglés NER (Named Entity Recognition) y en la actualidad ha sido solucionado por asociaciones como Stanford CoreNL.[14]

3. Descripción de las tecnologías

Este capítulo trata de las tecnologías más importantes para el procesamiento del lenguaje natural. En concreto se hablará de las bibliotecas de python útiles para el NLP, de los sistemas de procesado del texto, de la representación de documentos, de las técnicas de selección de características y de reducción de dimensionalidad y de los algoritmos de aprendizaje automático para el NLP y sus métricas.

3.1. Bibliotecas Python para NLP

3.1.1. NLTK

La biblioteca NLTK (Natural Language Toolkit) [15] fue desarrollada como un proyecto para el curso de lingüística computacional de la universidad de Pensilvania en 2001. Desde ese momento han colaborado para su desarrollo docenas de personas y universidades alrededor del mundo. El paquete nltk fue diseñado teniendo cuatro objetivos en mente:

- **Modularidad:** los componentes de la biblioteca pueden ser usados independientemente sin conocer los otros.
- **Consistencia:** ofrece un marco operativo con interfaces y métodos uniformes y con nombres fáciles de adivinar.
- **Extensibilidad:** ofrece un marco operativo con facilidad para añadir nuevos módulos y nuevas implementaciones.
- **Simplicidad:** ofrece un marco operativo fácil de usar, donde los usuarios pueden manejarse sin tener un amplio conocimiento del NLP.

Existen diversos módulos para cubrir las necesidades del NLP y se encuentran recogidos en la siguiente tabla:

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

Figura 3.1: Módulos del paquete NLTK[15]

3.1.2. Scikit-learn

Scikit-learn es una biblioteca de python especializada en aprendizaje automático. El proyecto empezó en 2007 como un trabajo para el "Google Summer of Code", y fue elaborado por David Cournapeau. [16] Más adelante Matthieu Brucher continuó con el proyecto como parte de su tesis y finalmente en 2010 Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort y Vincent Michel tomaron el control del proyecto e hicieron la revelación pública del la biblioteca.

Esta librería cuenta con distintos paquetes:

- **Aprendizaje supervisado:** Naive Bayes, árboles de decisión, k vecinos más cercanos, SVM...
- **Aprendizaje sin supervisar**: clustering, estimación de covarianzas, modelo de mezclas gausianas...
- **Selección y evaluación de modelos:** validación cruzada, métricas, curvas de validación...
- **Inspección y visualización**
- **Transformaciones de datos:** selección de características, preprocesamiento, reducción de dimensionalidades...

3.1.3. Gensim

Gensim cuyo nombre viene de los términos "generate similar" [17], es una librería que nace en 2008, como una colección de archivos python para la biblioteca de matemáticas digital checa. Tenían el objetivo de, dado un artículo, ofrecer los artículos más parecidos a él y se quería implementar una biblioteca clara, eficiente y escalable. En la actualidad se ha convertido en una de las librerías más robustas y eficientes utilizadas para el aprendizaje automático sin supervisar en textos.

Esta librería cuenta con distintos paquetes:

- **Corpora:** wikicorpus, dictionary...
- **Models:** Tfidf model, word2vec, LDA...
- **Summarization:** keywords, text cleaning, summarizer...
- **Scripts:** make wikiCorpus, glove2word2vec...
- **Topic coherence:** text analysis, segmentation, aggregation...

3.2. Procesado del texto

Para los seres humanos es fácil leer un texto desestructurado y extraer los elementos más importantes de él; sin embargo, para las máquinas este proceso puede resultar complejo sobre todo en textos largos. Por ello resulta muy importante la división de estos textos en componentes más pequeños y la reducción de la cantidad de palabras almacenadas.

3.2.1. Tokenización

El primer paso para el procesado del texto consiste en convertirlo en una lista de palabras que puedan ser tratadas como componentes independientes. Este proceso se conoce como tokenización y la separación suele realizarse por los espacios entre las palabras. Una vez tokenizadas las palabras, lo habitual es limpiarlas y normalizarlas para poder analizarlas del mismo modo y con mayor comodidad.

3.2.2. Eliminación de stopwords

Las stopwords son aquellas palabras que aparecen en más del 80 % de los textos. Son en su mayor parte palabras que sirven de nexo en el lenguaje pero que no aportan información adicional como conjunciones, preposiciones, artículos y determinantes. En ciertas ocasiones algunos adjetivos y verbos también pueden considerarse stopwords.

Las ventajas de eliminarlas son:

- Reducción del tamaño de la frase y la memoria necesaria.
- Mayor velocidad de indexación.
- Mejor selección de palabras claves y por tanto mejor eficiencia.

Las desventajas son:

- Pérdida de información que puede cambiar el sentido de la frase, como negaciones.

3.2.3. Stemming

Es un proceso por el cual cada palabra es transformada en su raíz, es decir en su unidad fundamental. De este modo se consigue reducir el número de términos diferentes y por tanto se facilita la reducción de dimensionalidad y se utiliza menos memoria para tratar los textos. Existen diferentes tipos de stemmers en la biblioteca NLTK, el más importante es el Porter's; pero también existen otros como Snowball y Lancaster.

3.2.4. Lematización

Al igual que el stemming, la lematización es un proceso por el cual a una palabra se le eliminan los prefijos y sufijos dejando únicamente el lema. La principal diferencia entre el lema y la raíz, es que el lema es una palabra existente. Nuevamente, gracias a este método se puede reducir el número de palabras de un texto. El principal lematizador de la biblioteca nltk es WordNetLemmatizer, para optimizar su funcionamiento requiere como parámetro la categoría morfológica de la palabra.

Figura 3.2: Ejemplo stemming y lematización[18]

3.3. Representación de documentos

Al igual que pasa con el cambio de analógico a digital, los ordenadores necesitan convertir el lenguaje natural en números para poder entenderlo y procesarlo. Este proceso se realiza creando un modelo de lenguaje y es lo que se conoce como representación de documentos. Existen distintos métodos para realizar la representación de documentos: [19]

3.3.1. 1-hot encoding

Es una técnica por la que a cada palabra se le asigna un vector binario, de dimensión el número de palabras del texto y donde casi todas las componentes son 0 menos la que representa la propia palabra que es 1. Este es un método que no tiene en cuenta la relación entre palabras dentro de un texto y además resulta bastante inefectivo para textos grandes debido a la gran dimensionalidad de los vectores.

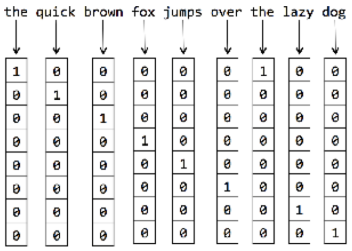


Figura 3.3: Ejemplo 1-hot encoding[19]

3.3.2. Frecuencia en el documento

La frecuencia de palabras en el documento, más conocida por su término en inglés Bag of Words, relaciona los términos presentes en un texto con su frecuencia de aparición. Es una técnica que resulta muy interesante a la hora de clasificar textos o de analizar sentimientos ya que establece patrones con palabras que toman más importancia en una categoría o en otra.

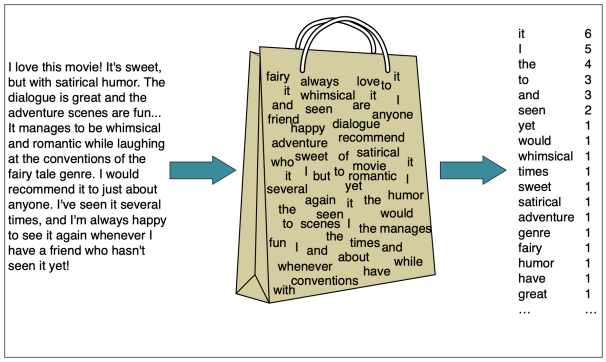


Figura 3.4: Ejemplo Bag of Words[20]

3.3.3. TF-IDF

En ciertas ocasiones ocurre que la frecuencia de palabras no es el mejor modo de representar un documento. Esto se debe a que, si una palabra se encuentra repetida muchas veces en todos los documentos, saldrá como una palabra importante y sin embargo, no aportará mucha información ni será muy discriminatoria. Para solucionar este problema se introduce el concepto de TF-IDF de una palabra t en un documento d. Este término se utiliza para dar importancia a aquellas palabras que salen mucho en un documento y poco en otros y que por tanto, son muy significativas del primero. [20] El TF-IDF se calcula de la siguiente manera:

$$TFIDF_{t,d} = TF_{t,d} * IDF_t \tag{3.1}$$

En él existe un primer término TF correspondiente a la frecuencia de una palabra en un documento:

$$TF_{t,d} = contar(t\ en\ d) \tag{3.2}$$

El segundo término corresponde al inverso de la frecuencia de la palabra en textos.

$$IDF_t = log(\frac{n^o documentos}{n^o documentos\ contienen\ t}) \tag{3.3}$$

Por tanto, se quiere ver la diferencia de un texto con respecto a otros, se dará importancia a aquellas palabras con un valor alto en el TF-IDF.

3.3.4. N-gramas

Durante el procesamiento de lenguaje natural hay ocasiones en las que el análisis palabra por palabra no es exacto o se pierde información. Esto ocurre con conceptos formados por más de una palabra, por ejemplo, "parque de atracciones". Para poder tratar estas situaciones se implementa el modelo n-gramas (bigramas, trigramas...) que establece la probabilidad que tiene una palabra de aparecer, dada una secuencia. Este modelo es muy útil y necesario en herramientas de reconocimiento del habla, donde en una frase alguna palabra puede haber llegado con ruido y el algoritmo tiene que intuir que palabra era. O por ejemplo, en herramientas de sugerencia de palabras y corrección de errores, ya que si un usuario escribe "Aquí ahí dos estuches", gracias al modelo n-gramas un ordenador puede saber que probablemente al lado de "aquí" venga "hay" y no "ahí", y lo corrija.

Para calcular estas probabilidades, el modelo n-gramas se basa en los textos introducidos para el aprendizaje. La probabilidad de que una palabra aparezca, dada una frase se calcula por la regla de Bayes como: casos en los que ha aparecido la frase completa con la palabra entre casos en los que ha aparecido la frase (con y sin palabra)

$$P(estás/que\ tal) = \frac{C(que\ tal\ estás)}{C(que\ tal)} \tag{3.4}$$

El modelo N-gramas presenta algunos problemas, el primero es que en el lenguaje a menudo las frases son largas y complejas, y por tanto en ciertas ocasiones palabras totalmente relacionadas están demasiado separadas como para que el algoritmo las tenga en cuenta. Por ejemplo: "Se me ha roto el ratón, no sé como voy a poder trabajar hoy con el ordenador", aquí ratón y ordenador están totalmente relacionadas y sin embargo, existe una gran distancia entre ellas. Otro problema

es que el modelo N-gramas se basa en corpus anteriores para calcular las probabilidades, estos en muchas ocasiones no están lo suficientemente completos como para poder calcular bien las probabilidades.

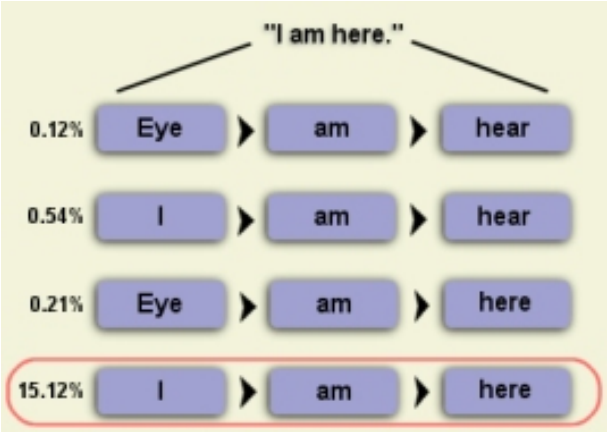


Figura 3.5: Ejemplo N-gramas[21]

3.3.5. Word embedding

Word embedding es una técnica de aprendizaje automático que consiste en que las palabras con significados parecidos tienen representaciones similares. Las palabras están representadas por vectores con valores reales y cientos de dimensiones (a diferencia de OHE que eran miles de dimensiones). Los valores asociados al vector de la palabra se obtienen por medio de redes neuronales, lo que lo relaciona directamente con el aprendizaje profundo (deep learning).[22]

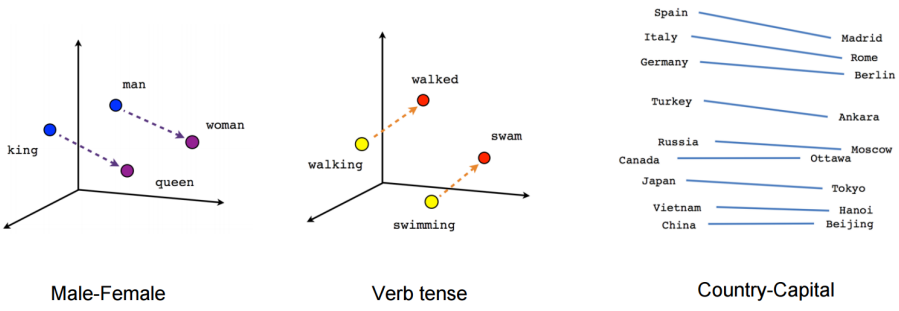


Figura 3.6: Word embedding[23]

Existen distintas técnicas para hacer word embedding, algunas de las más importantes son las siguientes.

3.3.5.1. Word2Vec

Esta técnica se basa en el uso de una red con dos capas, la primera recibe los inputs y tiene dimensión de CxV siendo C el numero de palabras y V la dimensión de las palabras. La salida del nodo oculto j es una combinación lineal de las componentes del input i y el peso de la conexión entre i y j.

$$salida\ del\ nodo\ j = u_j = \sum_{i=1}^{V} w_i j x_i \tag{3.5}$$

La capa de salida tiene dimensión V, un valor por cada palabra del input y el valor es la función softmax.

$$valor\ salida\ en\ el\ nodo\ k = O_k = \frac{exp(u_k)}{\sum exp(u_q)} \tag{3.6}$$

$$u_k = \sum_{i=1}^{N} w_j k h_j \tag{3.7}$$

Los valores de w se actualizan a cada iteración utilizando el algoritmo del descenso de gradiente estocástico, hasta conseguir los valores óptimos.

Existen dos formas de aplicación del Word2Vec, la primera es CBOW y consiste en dado un contexto predecir la palabra que falta. La segunda es skip-gram y consiste en adivinar el contexto dado una palabra. [24]

3.3.5.2. GloVe

Su nombre viene de "Global vectors for word representation" y al igual que Word2Vec es un algoritmo no supervisado para obtener la representación vectorial de palabras.[25]

Este modelo se entrena con matrices de ocurrencia, donde a cada palabra se le asigna la frecuencia con la que aparece con otra palabra en un texto. El modelo es se basa en un sistema log-bilineal con la función de los mínimos cuadrados como objetivo y relaciona que las palabras con mayor frecuencia tienen algún tipo de significado común. El objetivo de GloVe es obtener vectores cuyo producto escalar sea igual al logaritmo de la coincidencia de palabras.

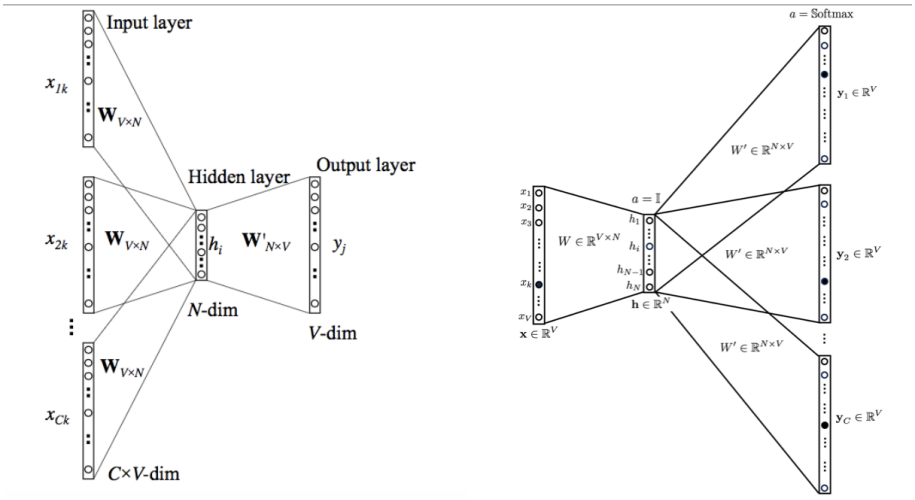


Figura 3.7: Modelos CBOW y Skip-gram[24]

3.3.5.3. FastText

Es una mejora del Word2Vec, pero en vez de utilizar las palabras completas para el aprendizaje, las representa como grupos de N letras. Por ejemplo, con trigramas caballo sería cab, aba, bal, all, llo. De este modo se capta el significado de palabras más pequeñas, el significado de los prefijos y sufijos y consigue tratar con palabras raras para las cuales no había sido entrenado dividiéndolas en N-gramas.[26]

3.4. Selección de características

Más comúnmente conocido por su término en inglés "feature selection"; es el proceso por el cual se elige un subgrupo de características dentro del "training set" para realizar la clasificación del texto únicamente en base a ellas. Este proceso tiene dos ventajas, la primera es que al reducir la cantidad de vocabulario se hace el proceso más rápido y eficiente. La segunda ventaja es que con la selección de características se eligen aquellas palabras más relevantes para la clasificación, por tanto, se mejora la precisión del algoritmo.[27]. Existen distintos métodos para hacer la selección de características:

3.4.1. Métodos de filtro

Utilizan distintos criterios como la distancia entre palabras, la dependencia, la consistencia y distintos estadísticos para hacer una clasificación de la importancia de las variables y quedarse con las más pertinentes. Posteriormente estas variables son introducidas en el algoritmo de machine learning y se calcula su rendimiento. [28]

- **Chi-cuadrado**: es un estadístico ampliamente utilizado para medir la dependencia entre dos variables. Este estadístico nos dice cuanto se desvían dos elementos, el previsto y el observado. En el caso de "feature selection" queremos ver la dependencia entre dos características (O y E) y que esta dependencia sea máxima. Esto se medirá con valores altos de χ^2 ya que cuanto más independientes sean dos características más similares serán sus valores esperado y observado.

$$\chi_c^2 = \sum \frac{(O_i - E_i)^2}{E_i} \tag{3.8}$$

- **ANOVA:** Su nombre viene de análisis de varianza y fue desarrollada por R.A.Ficher. Es un estadístico que permite analizar el efecto de uno o más factores sobre la media global de una población, por tanto, permite dividir las características en conjuntos.

- **Correlación de Pearson** Es una medida de la correlación lineal entre dos variables aleatorias. Se buscará características con coeficientes de correlación muy cercanos al 1, es decir, muy dependientes.

$$\rho_{x,y} = \frac{\sigma_{x,y}}{\sigma_x \sigma_y} \tag{3.9}$$

3.4.2. Métodos de envoltura

Estos métodos de selección de características comienzan utilizando un subconjunto de ellas, calculan el modelo y el rendimiento de éste. Luego repiten este proceso con otras agrupaciones, introduciendo o eliminando diferentes características hasta generar el conjunto óptimo. Este proceso es muy costoso en términos de recursos ya que requiere hacer el cálculo para numerosos grupos de características diferentes.[29] Existen distintos algoritmos para los métodos de envoltura:

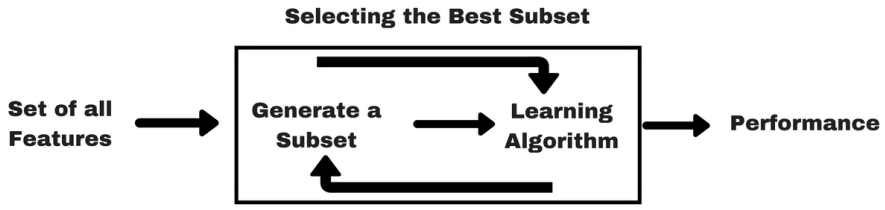


Figura 3.8: Método de envoltura[28]

- **Selección hacia delante**: el proceso empieza con un conjunto vacío, de todas las características se obtiene la mejor en función del p-valor y se introduce en el conjunto. Luego se prueba con el conjunto generado y otra característica de las restantes y nuevamente se introduce la que da mejor p-valor. Así sucesivamente se van añadiendo una por una las características.
- **Selección hacia atrás:** el proceso es al revés que el previo, se empieza con el conjunto lleno y se va eliminando a cada iteración la característica con el p-valor mayor, es decir, la más insignificante.
- **Selección recursiva:** es una combinación de las dos anteriores. Empieza como selección hacia delante añadiendo al conjunto la característica que de el mejor p-valor. La diferencia es que una vez introducida esa característica, se comprueban todas las demás pertenecientes al conjunto y se eliminan las que den un p-valor mayor, es decir, se hace selección hacia atrás.

3.4.3. Métodos integrados

Estos métodos combinan filtrado y envoltura; en ellos la selección de características se hace durante la propia construcción y entrenamiento del modelo. Los algoritmos más destacables son LASSO que utiliza el valor absoluto de los coeficientes como penalización y RIDGE que utiliza cuadrado.

3.5. Reducción de dimensionalidad

Como su propio nombre indica, las técnicas de reducción de dimensionalidad reducen las dimensiones de los datos, manteniendo la mayor cantidad de información posible. Este proceso se hace con distintos objetivos: para reducir el tiempo

y almacenamiento necesario, para facilitar la visualización de los datos y para mejorar el rendimiento del algoritmo.

3.5.1. LSA (Análisis semántico latente)

Es una técnica de reducción de dimensionalidad que comienza utilizando una matriz de frecuencia de términos-documentos (términos en las filas y documentos en las columnas). A esta matriz se le aplica la descomposición en valores singulares que consiste en representar una matriz como la multiplicación de otras tres.

$$M = U\Sigma V^T \tag{3.10}$$

Cada una de estas matrices tiene una utilidad, U es singular por la izquierda y corresponde con la matriz que relaciona los términos y los temas; Σ es una matriz diagonal con términos no negativos y refleja la importancia del tema. Finalmente, V es singular por la derecha y refleja el tema de los documentos.

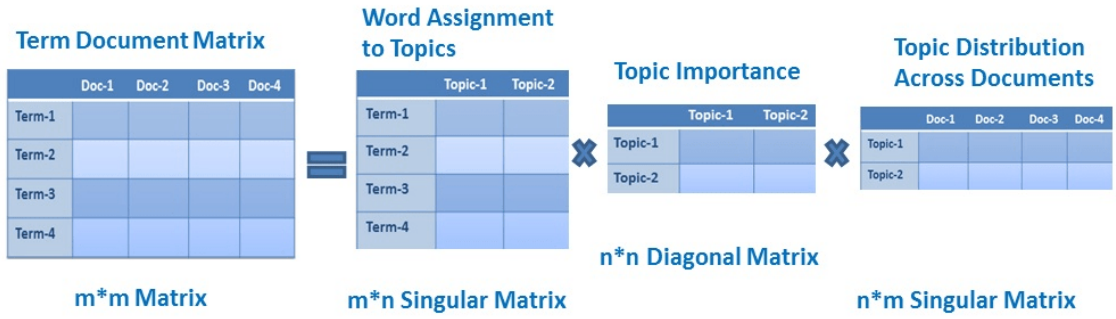


Figura 3.9: LSA[30]

3.5.2. LDA(Análisis discriminante lineal)

Cuando se quiere reducir la dimensionalidad de un conjunto, una de las soluciones posibles es proyectar los elementos en los ejes. Para ello, es muy importante la elección del eje de proyección ya que, si se limita el uso a los ejes X o Y, se puede perder mucha información debido a características que se solapen. Este problema se resuelve gracias al LDA, una técnica estrechamente ligada con ANOVA y PCA,

donde la idea es generar un nuevo eje basado en la información de las características. Este eje se situará de tal forma que la varianza entre las características sea mínima y maximice la distancia entre las medias. Más adelante se proyectarán las características en este nuevo eje y de este modo se reducirá la dimensión.[31]

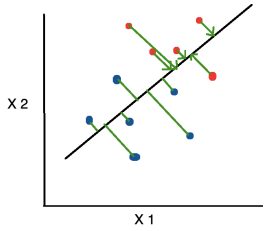


Figura 3.10: Análisis discriminante lineal[31]

3.5.3. LDA(Latent Dirichlet Allocation)

Es una técnica de aprendizaje no supervisado que sirve para dividir las características en temas. La división se hace por medio de una bolsa de palabras donde los textos con las mismas palabras estarán en el mismo tema. El modelo es de probabilidad generativa y la idea básica es que los documentos se reparten en los temas latentes y cada tema tiene una distribución diferente de probabilidad. Para asignar palabras a temas el algoritmo empieza asignando de manera aleatoria un tema a cada palabra de un documento. Luego para cada palabra en cada documento se calcula la probabilidad de que se de un tema dado un documento y la probabilidad de que se de la palabra dado un tema. Finalmente se actualiza que la probabilidad de que w pertenezca al tema es la multiplicación de las anteriores probabilidades. [32]

De este modo se pueden separar las palabras por temas y posteriormente los documentos por temas.

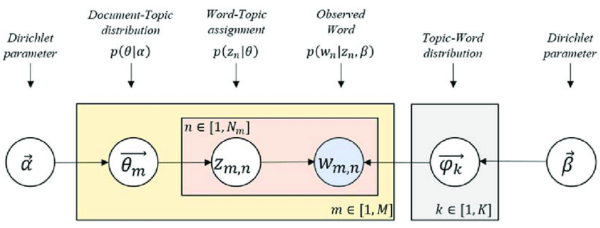


Figura 3.11: LDA[33]

3.6. Modelos de clasificación

3.6.1. Aprendizaje supervisado

El aprendizaje supervisado es aquel en el que para entrenar al modelo se introducen una serie de datos etiquetados. Este aprende de estos datos y luego los utiliza para predecir la salida dado un nuevo valor de entrada. Es el método más utilizado y eficiente para hacer la clasificación de textos.

Existen dos tipos de algoritmos de clasificación, los discriminatorios y los generativos. En el caso de los generativos el objetivo del algoritmo es aprender a crear la clase desde cero; en ellos la clasificación se hace partiendo de un input y prediciendo que clase terminaría generando un objeto de ese tipo. Por el contrario, en el caso de la clasificación discriminatoria el modelo aprende a clasificar en categorías, es decir, qué característica es más representativa de un grupo, pero sin aprender como se generaría ese grupo.

3.6.1.1. Naive bayes

Naive bayes es un algoritmo de clasificación generativa que se basa en una categorización por probabilidades. Es decir, para comprobar si un texto t pertenece a una categoría â, calculará para todas las categorías la probabilidad de c dado t y cogerá el valor máximo:

$$\hat{a} = argmax(P(c|t)) \ \forall \ c \ \epsilon \ Categorías \quad (3.11)$$

La probabilidad de c dado t se calcula por medio de la regla de Bayes, introducida en 1763 por el matemático con el mismo nombre. Esta probabilidad se puede calcular como:

$$P(c|t) = \frac{P(t|c) * P(c)}{P(t)} \quad (3.12)$$

Un texto está compuesto por distintas características por tanto se puede expresar:

$$P(t|c) = P(f1, f2, f3...|c) \quad (3.13)$$

Para simplificar esa probabilidad se asume que todas las características son independientes entre sí, de ahí el nombre Naive Bayes. Teniendo en cuenta esta premisa

la probabilidad condicionada se puede expresar por medio de la regla de la cadena y se puede descomponer como:

$$P(f1, f2, f3...|c) = P(f1|c) * P(f2|c) * P(f3|c)... * P(fn|c) \quad (3.14)$$

Aplicando las ecuaciones 3.11, 3.12 y 3.14 se puede llegar al resultado del clasificador:

$$\hat{a} = argmax(P(c) * \prod_{f \epsilon F} P(f|c)) \ \forall \ c \ \epsilon \ Categorías \ (3.15)$$

Este clasificador es tratado en espacio logarítmico para aumentar la velocidad y convertirlo en lineal:

$$\hat{a} = argmax(log(P(c)) + \sum_{f \epsilon F} log(P(f|c))) \ \forall \ c \ \epsilon \ Categorías \ (3.16)$$

Finalmente, gracias a esta fórmula el algoritmo puede decidir si un texto pertenece a una categoría o a otra.

Para poder aplicar el clasificador también es necesario calcular las probabilidades de cada característica en un texto. Este proceso se hace durante el entrenamiento del algoritmo como:

$$P(fi|c) = \frac{count(fi, c)}{\sum_{f \epsilon F} count(f, c)} \quad (3.17)$$

Este dato es suavizado a través de Laplace para evitar probabilidades igual a 0 que anulen el resultado de la probabilidad condicionada.[20]

3.6.1.2. Regresión logística

Al contrario que Naive Bayes que es un algoritmo de clasificación generativo, la regresión logística es discriminatorio. Ambos algoritmos tienen en común que se basan en probabilidades para la discriminación. En la regresión logística lo primero que hace el algoritmo, durante el entrenamiento, es asignar a cada característica un peso en relación a la categoría en la que esté; cuanto más peso más importante será esa característica en esa categoría, y luego le añade un término de sesgo. Por tanto, la representación de una categoría se hace como una combinación linear de las características y sus pesos:

$$z = (\sum_{i=1}^{n} wi * xi) + b \quad (3.18)$$

Este valor es pasado a la función logística o sigmoid, que toma valores entre 0 y 1 y tiene la forma:

$$\sigma(z) = \frac{1}{1 + exp(-z)} \tag{3.19}$$

Por tanto cuando se quiere clasificar un texto en una categoría, se introduce de la forma z a la función sigmoide y si el resultado es mayor de 0.5 (o otra barrera que se establezca) pertenece a la categoría original; si el valor es menor pertenece a la otra categoría.[20]

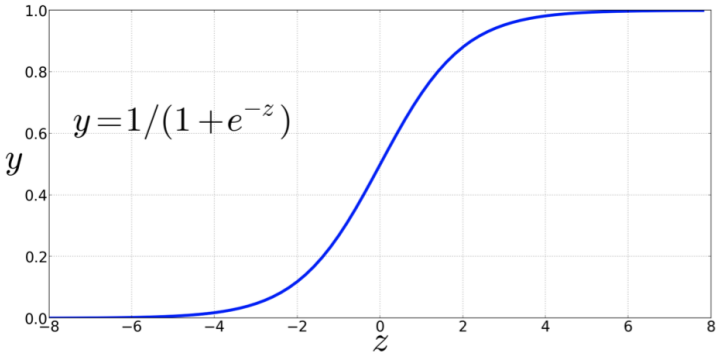


Figura 3.12: Función sigmoide[20]

3.6.1.3. Árbol de decisión

Los árboles de decisión son algoritmos que se basan en ir dividiendo la información en subconjuntos en función de sus características. Cada árbol empieza con la característica más importante seleccionada por la entropía, la relación de ganancia o el índice gini; y divide los datos en dos subconjuntos. Este proceso se repite recursivamente hasta que no queden datos o atributos.

3.6.1.4. Bosques aleatorios

Este algoritmo es más conocido por su nombre en inglés "Random forests". Está constituido por una serie de arboles de decisión, cada uno da una clasificación y al final se elige la clasificación más votada. Se basa en el concepto de que varios modelos no correlacionados tendrán un rendimiento mejor que un solo modelo. Por tanto, para que este tipo de algoritmos funcione, es fundamental que las predicciones hechas por los individuos tengan una baja correlación entre ellas.

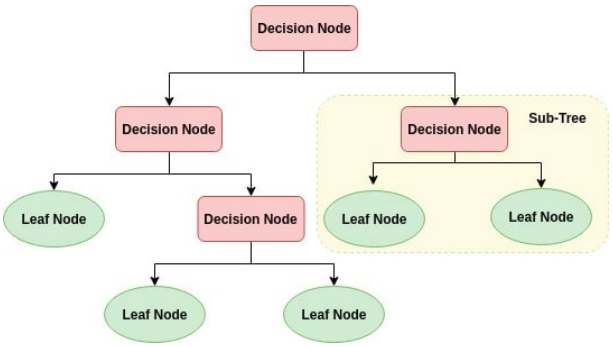


Figura 3.13: Árbol de decisión[34]

3.6.1.5. K vecinos más próximos

El algoritmo se basa en el concepto de que los elementos más próximos en distancia suelen pertenecer mismo grupo . Los elementos a clasificar son vectores de N dimensiones y pueden ser teóricamente representados en el espacio euclídeo. El algoritmo selecciona los K vecinos más próximos al elemento que queremos clasificar. La proximidad se hace en función de la distancia euclídea:

$$d(x,y) = \sqrt{\sum_{i=1}^{N}(x_i - y_i)^2} \tag{3.20}$$

Una vez seleccionados los vecinos, establece que la categoría del elemento es aquella que más se repita entre los vecinos.

Para la elección de K el mejor método es probar con diferentes K y quedarse con aquella que tenga el error más bajo. Algunas cosas a tener en cuenta son que cuanto más grande es la K más se reduce el ruido, pero también puede llegar un punto donde se pierda el límite entre clases y por tanto haya más errores. También es importante tener en cuenta que para una clasificación binaria resulta interesante elegir una K impar para que siempre haya una mayoría.

La ventaja de este algoritmo es que tiene una fácil implementación y sirve para clasificaciones, regresiones y búsquedas; sin embargo, cuando el número de atributos es muy grande, se vuelve muy lento.[35]

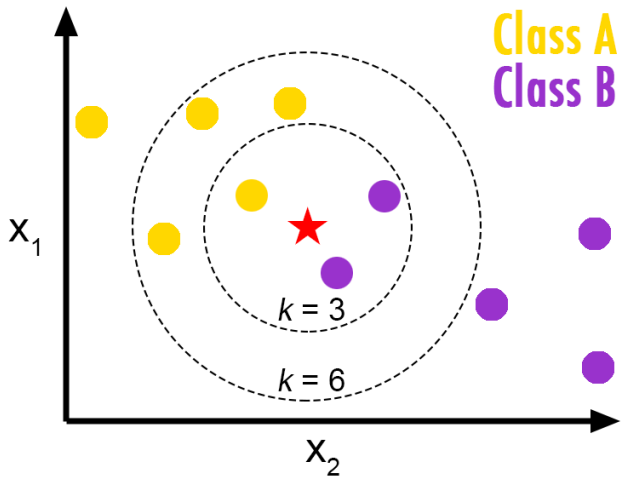


Figura 3.14: k vecinos más próximos[36]

3.6.1.6. Máquinas de vectores de soporte (SVM)

Este algoritmo está basado en el uso de hiperplanos como separadores binarios de categorías. Un hiperplano está definido como $a_1x_1 + a_2x_2 + ... + a_nx_n = c$ por tanto si el valor es superior al hiperplano estará en una de las categorías y si por el contrario es menor, estará en la otra categoría. Existen diferentes tipos de hiperplanos, pero para optimizar el algoritmo se busca el plano óptimo de separación, es decir el que más alejado está de todos los elementos. Las observaciones que definen este plano son las que se conocen como vectores de soporte.

Sin embargo, el plano óptimo de separación es muy sensible a cambios de datos y al ajustarse perfectamente a los datos del entrenamiento suele producir overfitting. Para evitar esto se utilizan los llamados "clasificadores de vector de soporte". Estos planos no separan tan perfectamente las clases, pero son más robustos y menos sensibles al overfitting.

Los clasificadores de vector de soporte son de gran utilidad para problemas lineales; sin embargo, para datos separados no linealmente fallan. En esos casos se utilizan las máquinas de vector de soporte que añaden una dimensión a los datos para que sea posible separar los datos en dos categorías por medio de un hiperplano. Para hacer este aumento de dimensión se utilizan kernels y los hay lineares, polinómicos, gaussianos...[37]

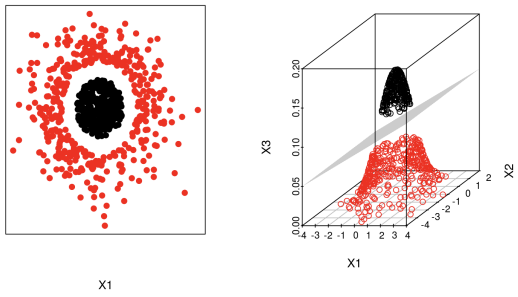


Figura 3.15: SVM[37]

3.6.2. Aprendizaje no supervisado

En el aprendizaje no supervisado, no se introducen unos datos etiquetados para entrenar el algoritmo, sino que todos están sin etiquetar. En contraste con los algoritmos de aprendizaje supervisados que buscan una función que les permita hacer predicciones, en el aprendizaje no supervisado el objetivo es encontrar la estructura básica de los datos que nos permita obtener más información sobre ellos. En general estos algoritmos sirven para hacer agrupaciones o para reducir la dimensionalidad. [38]

3.6.2.1. K-medias

Es un algoritmo que sirve para hacer agrupamiento de datos y al igual que el algoritmo K vecinos más próximos, se basa en la distancia euclídea entre los elementos. El algoritmo comienza eligiendo aleatoriamente k centroides, luego a cada punto se le asignara el centroide más cercano. Finalmente se calculará el nuevo centroide de cada clúster como el punto en el que el error cuadrático a los elementos del clúster es mínimo. Este proceso se repetirá hasta que los centroides no cambien más o hasta que se llegue al número máximo de iteraciones definidas.

3.6.3. Redes neuronales

Las redes neuronales son estructuras que simulan la conexión de las neuronas del cerebro. Están compuestas por varias capas, a cada neurona le entra un impulso correspondiente a la entrada, multiplicada por el peso de la conexión. Estos impulsos se suman en la neurona y se pasan a una función de activación que da

la salida final de la neurona. La primera capa de neuronas recibe la información inicial, y la última capa devuelve la salida final del algoritmo, todas las demás capas se llaman capas ocultas.[39]

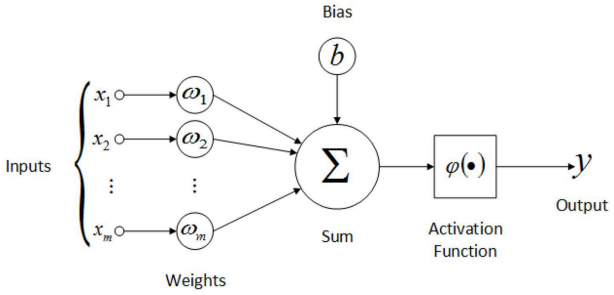


Figura 3.16: Neurona[40]

El entrenamiento de un modelo de aprendizaje profundo con redes neuronales se basa en ajustar los pesos de las conexiones de las neuronas para conseguir la salida esperada. Esto se realiza por un método llamado propagación hacia atrás y consiste en introducir datos a la red y se van ajustando los pesos en función de los errores hasta conseguir la salida esperada. [41]

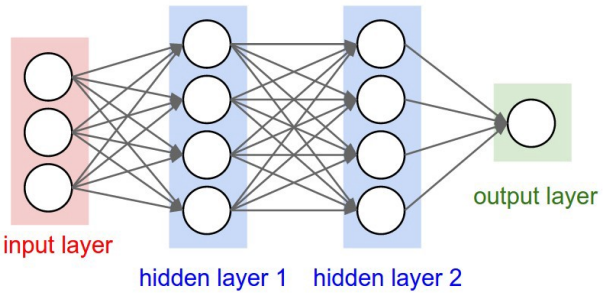


Figura 3.17: Red neuronal[40]

3.7. Métricas del aprendizaje supervisado

Aquí se presentan algunas métricas del aprendizaje supervisado cuando está enfocado a la clasificación, estas métricas son los valores que nos permiten ver la eficiencia del algoritmo.[41]

Matriz de confusión

Las matrices de confusión están formadas por filas que representan los valores reales de los datos y columnas que representan los valores predichos. En cada celda se encuentra una cifra que corresponde al número de datos de cada categoría observada y predicha. De ella podemos sacar distintos componentes:

- VP: son los verdaderos positivos, es decir la cantidad de valores que son positivos y se identifican como positivos
- VN: son los verdaderos negativos, al igual que antes son aquellos valores que son negativos y se identifican como negativos
- FP: son los falsos positivos, es decir aquellos valores que son negativos y se identifican como positivos
- FN: son los falsos negativos, aquellos valores que son positivos y se identifican como negativos

		Predicción	
		Positivos	**Negativos**
Observación	**Positivos**	Verdaderos Positivos (VP)	Falsos Negativos (FN)
	Negativos	Falsos Positivos (FP)	Verdaderos Negativos (VN)

Figura 3.18: Matriz de confusión[41]

Exactitud

Es el porcentaje de elementos clasificados correctamente, es decir lo cerca que está el algoritmo de conseguir los valores reales.

$$Exactitud = \frac{VP + VN}{Total} \tag{3.21}$$

Sensibilidad

Es la tasa de verdaderos positivos, es decir, el porcentaje de positivos identificados de entre todos los datos positivos. Un algoritmo muy sensible detecta muy bien la clase para la que se ha medido la sensibilidad.

$$Sensibilidad = \frac{VP}{VP + FN} \tag{3.22}$$

Precisión

Es la tasa de verdaderos positivos, es decir porcentaje de positivos reales de entre todos los datos identificados como positivos; representa como de cerca están los valores entre ellos. Un algoritmo preciso indica que si sacamos un elemento de entre los clasificados como positivos, el elemento probablemente esté bien clasificado.

$$Precisión = \frac{VP}{VP + FP} \tag{3.23}$$

Especificidad

Es la tasa de falsos negativos, es decir porcentaje de negativos identificados de entre todos los negativos. Un algoritmo con mucha especificidad indica que si sacamos un elemento de entre los clasificados como negativos, el elemento probablemente esté bien clasificado.

$$Especificidad = \frac{VN}{VN + FN} \tag{3.24}$$

F1-score

Es una medida que relaciona precisión y sensibilidad; es muy útil cuando las categorías son diferentes de positivo o negativo.

$$F1 = \frac{2 * sensibilidad * precisión}{sensibilidad + precisión} \tag{3.25}$$

4. Definición del trabajo

En este capítulo se hablará del objetivo principal y objetivos secundarios del proyecto, se describirá la base de datos utilizada, lo que se entiende por presencia de testigos y se hablará del funcionamiento del algoritmo creado.

4.1. Objetivos del proyecto

Este trabajo tiene como objetivo principal elaborar un modelo de clasificación de las descripciones de la base de datos de Hollaback. La clasificación se hará en base a dos categorías, aquellas en las que hay presencia de testigo, y aquellas en las que no.

Para lograr el objetivo principal, se establecen cuatro objetivos secundarios:

1. **Análisis de datos exploratorio:** para obtener estadísticas relevantes sobre la base de datos, como las ubicaciones de las denuncias, el tipo de descripciones, las franjas horarias en las que ocurrieron...

2. **Aplicación de NLP básico a las descripciones:** como una primera herramienta para tratar el texto y observar sus características principales.

3. **Detección de los elementos que permiten al lector identificar la presencia de un testigo dentro de una descripción y etiquetado de descripciones.**

4. **Elaboración e implementación de un modelo NLP para diferenciar las descripciones con testigo.**

4.2. Descripción de la base de datos

La base de datos utilizada es propiedad de la ONG Hollaback, y es cedida a SoGooData con el objetivo de poder elaborar este proyecto sin fines lucrativos. Esta base de datos contiene aproximadamente 12000 denuncias diferentes. Las denuncias están hechas por víctimas o testigos de situaciones de acoso en lugares públicos y está compuesta por distintos campos:

- **ID:** es una cadena de caracteres, única en la base de datos, asignada automáticamente por la página al hacer la denuncia y que permite identificarla.
- **ReportedAt:** es un valor con formato fecha/hora (YYYY-MM-DD T hh:mm:ss) y recoge la fecha y hora del incidente.
- **Lng:** es un valor con formato coma flotante (float) y recoge la longitud del lugar donde ocurrió el incidente.
- **Lat:** también es un valor con formato coma flotante (float) y recoge la latitud del lugar donde ocurrió el incidente.
- **Categories:** es una cadena de caracteres que recoge la categoría o categorías del acoso sufrido. Las diferentes categorías pueden ser: abuso verbal, gestos sexuales, tocar inapropiadamente, ser seguido, homofobia, transfobia, racismo, colorismo, discriminación contra las personas con discapacidad o discriminación por el tamaño de alguien (sizeism).
- **ReportedByBystander:** es un valor booleano que es verdadero si la historia está narrada por un testigo y falso si está narrada por la víctima.
- **Title:** es una cadena de caracteres con el título de la descripción.
- **Description:** es una cadena de caracteres con la descripción del incidente. Este campo se utilizará para hacer la clasificación.

ID	Reported At	Lng	Lat	Categories	ReportedByBystander	Title	Description
[redacted]	2010-10-04T15:43:58+00:00	-122.67611111111	45.523333333333	["verbal"]	false	Portland Potty-Mouth	Portland, OR is a place where I actually encountered "Not to be disrespectful, but that is a very pretty bac

Figura 4.1: Ejemplo extraído de la base de datos

4.3. Definición de presencia de testigos

Durante este proyecto se considerará que hay un testigo cuando una persona externa a la situación la ve. Sin embargo, debido a la gran variedad de descripciones y a las diferentes formas de contar las experiencias, en ocasiones puede resultar compleja la diferencia entre presencia o no. A continuación, se detallan las consideraciones hechas para la clasificación.

- Se considerará presencia de testigo siempre que haya alguien que pueda haber visto el incidente, intervenga o no. Esto se debe a que en el momento que hay una persona, aunque no actúe, ya cambia el sentimiento de la víctima.
- Aunque las víctimas o acosadores sean más de una persona, no se considerarán como testigos, sino como múltiples víctimas o múltiples acosadores.
- Cuando se habla de un lugar público, se considerará como testigo cuando se diga explícitamente que había gente o se haga referencia a alguna persona externa; sino pese a ser público se considerará que no había testigos.

4.4. Desarrollo de la herramienta de clasificación

El programa de procesado de las descripciones se realiza en diversos pasos para asegurar mayor eficiencia. El diagrama de flujo completo se puede encontrar en la figura 4.2 y los pasos se encontrarán desarrollados durante el análisis de resultados. En líneas generales se realizará:

1. Importación de datos.
2. Limpieza de descripciones.
3. Tokenización del texto.
4. Normalización de las descripciones.
5. Tratamiento del idioma.
6. Eliminación de stopwords.
7. Stemming y lematización.

8. Eliminación de stopwords.
9. NLP básico: representación del documento y reducción de dimensionalidad.
10. Selección de características y entrenamiento y prueba de los modelos de clasificación. Los modelos utilizados serán: regresión logística, naive bayes, máquinas de vector de soporte, clasificador SGD y k vecinos más cercanos.
11. Selección del mejor modelo.
12. Clasificación de descripciones completas.

Cabe destacar que el proceso de entrenamiento y prueba de modelos se hará con conjuntos de datos diferentes para ver como varía la eficiencia. Las combinaciones serán:

- Modelo entrenado con descripciones completas y probado con descripciones completas.
- Modelo entrenado con descripciones simplificadas y probado con descripciones simplificadas.
- Modelo creado con descripciones simplificadas y probado con descripciones completas.
- Modelo creado con descripciones completas y simplificadas y probado con descripciones completas.

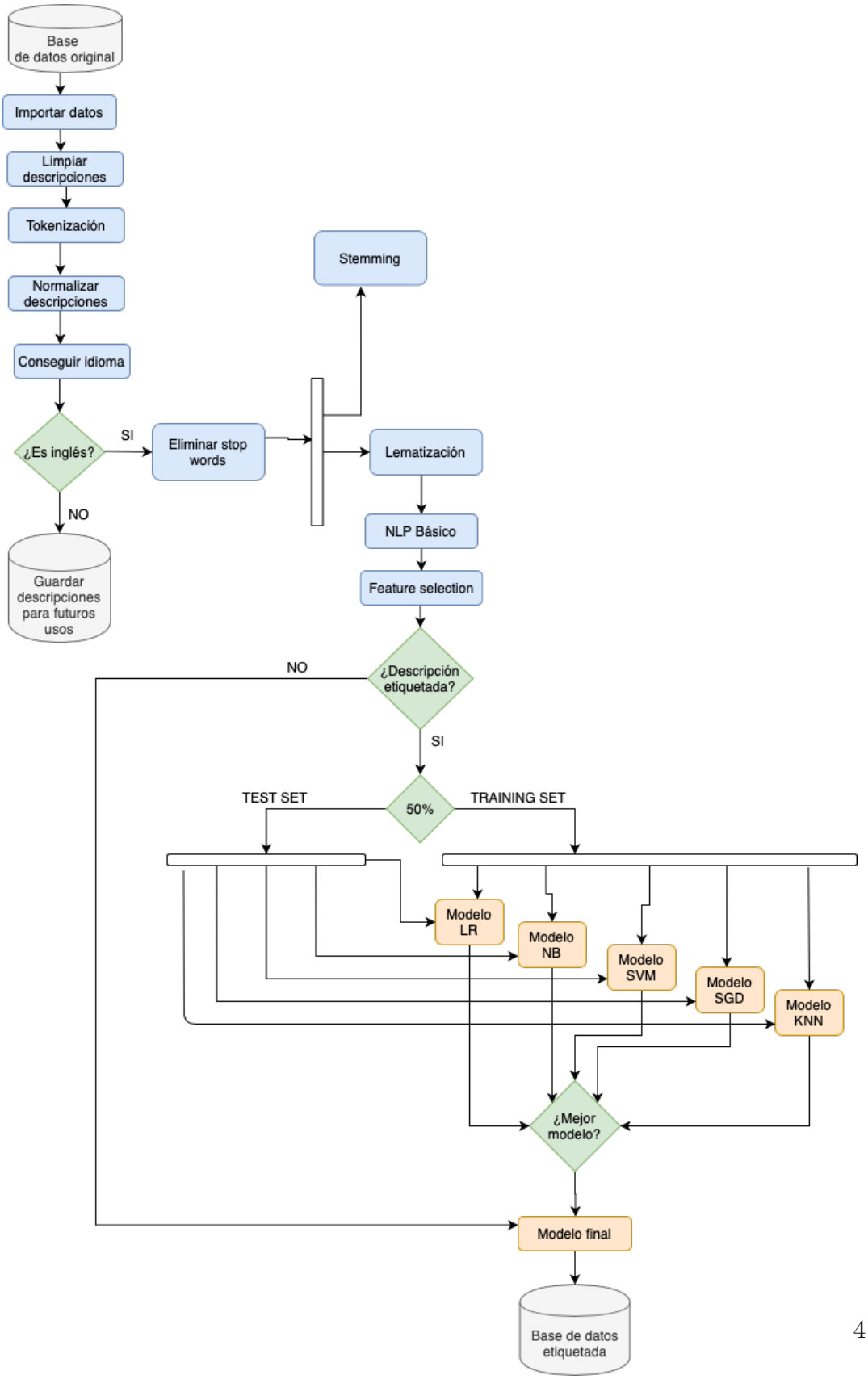


Figura 4.2: Diagrama de flujo del algoritmo

5. Análisis de resultados

5.1. Resultados del análisis preliminar

En esta sección se hizo un análisis previo de la base de datos para identificar algunas características e información destacable.

Para interpretar bien los resultados es importante tener en cuenta que los eventos de acoso reales y el número de denuncias no están fuertemente correlacionados. Existen numerosos factores sociales y culturales, como puede ser el conocimiento de la plataforma Hollaback, que pueden hacer favorecer que se realicen más o menos denuncias dependiendo de las condiciones. Algunos de los factores que se piensa que pueden afectar son la localización geográfica, la hora en la que ocurrió, el idioma de las denuncias o el tipo de acoso.

5.1.1. Distribución de las denuncias

Gracias a la herramienta de uso libre llamada Carto [42] se pueden representar diferentes capas de datos en un mapa dinámico. En este caso se hizo una representación de las denuncias en un mapa mundial utilizando la latitud y longitud proporcionadas. De este modo se pudo observar la distribución general de las denuncias, que refleja en gran medida el alcance mundial del movimiento Hollaback.

En esta imagen se puede ver que las denuncias se concentran principalmente en Europa y Estados Unidos. En estos lugares el movimiento Hollaback es más fuerte, y se debe a distintos factores. Uno de los más importantes es porque son los primeros lugares donde se lanzó la plataforma y por tanto el origen del movimiento, concretamente en Nueva York. Esta imagen también refleja las zonas del mundo donde el movimiento todavía no ha llegado como África o está empezando a coger fuerza como la India, Sudeste asiático y Sudamérica.

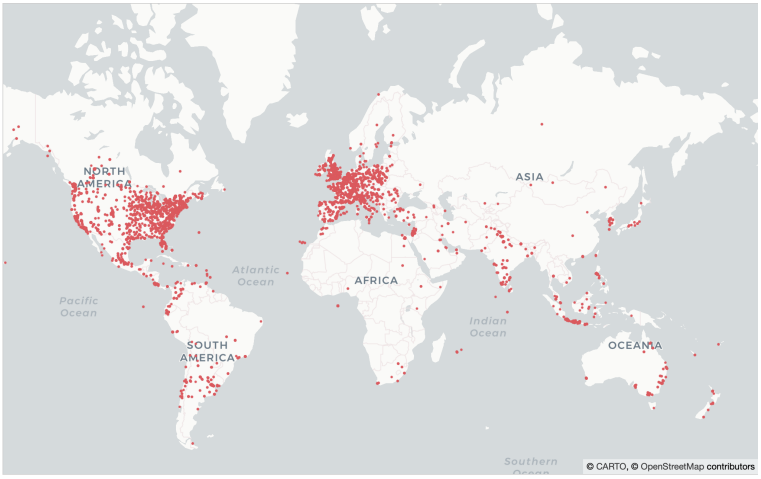


Figura 5.1: Distribución mundial de denuncias

Otro dato de interés es la distribución del número de denuncias a lo largo de los años, para ver como ha ido variando el éxito y alcance de Hollaback desde su creación. Se puede observar que desde que empezó en 2005 el número de usuarios fue creciendo hasta 2014 donde tuvo su punto más alto, desde entonces el número de denuncias ha ido disminuyendo con los años. Se podría pensar que la caída se debe a un descenso en el número de situaciones de acoso; sin embargo, resulta poco probable que el decrecimiento sea tan grande sobre todo si se compara con estadísticas generales. Por tanto, lo más probable es que la disminución esté causada por la aparición de nuevas plataformas similares que han hecho que se repartan los usuarios.

5.1.2. Idioma de las denuncias

Otro dato que se analizó fue el idioma de las denuncias. En el caso del NLP es necesario construir un modelo para cada idioma, por tanto esta información tiene relevancia para elegir el idioma más común.

En este caso se comprobó que el idioma más usado era el inglés, siendo el 70 % de las descripciones en este idioma. Eso hizo que se decidiese utilizar el inglés como idioma base para el desarrollo del programa. Los siguientes idiomas mayoritarios son español y francés; por tanto serían otras lenguas interesantes a analizar para posteriores estudios.

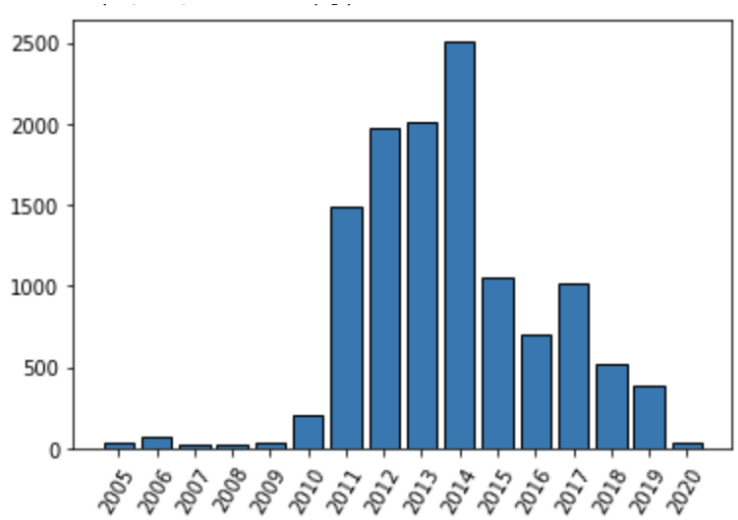


Figura 5.2: Distribución de las denuncias por años

Figura 5.3: Idiomas en denuncias

5.1.3. Distribución temporal de las denuncias

Posteriormente, se estudió la distribución de las denuncias en función de cuatro intervalos horarios: madrugada (1 am a 6 am), mañana (7 am a 12 am), tarde (1 pm a 6pm) y noche (7pm a 12pm). El objetivo era ver en qué intervalos es más frecuente sufrir una experiencia de acoso y también ver como se distribuían en una ciudad.

En este caso se escogió una ciudad conocida para poder entender mejor los resultados. En la imagen 5.4 se ve claramente que el menor número de incidentes ocurren de madrugada, pero que están más concentrados en el centro. En el resto

Figura 5.4: Distribución de las denuncias por intervalos en Madrid

de horarios la concentración de denuncias sigue siendo en el sector más céntrico, sin embargo, se puede ver que por las mañanas tiende a estar desplazado al norte y por las tardes al sur-este. La concentración mayoritaria en el centro se debe a que Hollaback es más conocido entre extranjeros que se mueven principalmente por esa zona. Esto se puede comprobar viendo el idioma de las descripciones, inglés casi siempre en el centro y español en los exteriores. Por tanto, no se puede deducir que haya más acoso en una zona o en otra, solo que está más denunciado.

En la figura 5.4 también se puede ver una gran diferencia entre las horas de la madrugada y las de el resto del día. Para comprobar si esto ocurría en el resto del mundo se hizo un histograma por intervalos horarios de todas las denuncias.

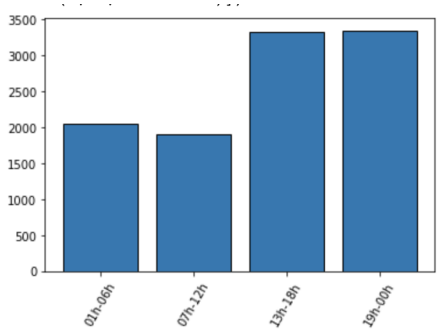


Figura 5.5: Histograma de denuncias por intervalos horarios

En esta imagen se puede ver que la madrugada y la mañana son los intervalos

horarios con menos denuncias. Resulta interesante el hecho de que a pesar de que de madrugada hay un tercio de afluencia en las calles, hay más denuncias que por la mañana. Esto implica que cruzándose a alguien, hay mayor probabilidad que esta persona vaya a acosar de madrugada que por la mañana y que por tanto sea una franja horaria mucho más peligrosa. Otro dato interesante del gráfico es que por la tarde y por la noche hay el mismo número de denuncias. Por tanto, al igual que en el caso anterior, al haber menos gente por la noche, hay mayor probabilidad de sufrir un acoso.

5.1.4. Tipo de acoso

Otro de los datos interesantes que se recoge en la base es el tipo de acoso sufrido por las víctimas. Estos datos se pueden comparar con los presentados en la introducción y elaborados en el #MeToo [1], que tenía en cuenta todos los tipos de acoso y no solo aquellos en lugares públicos. Se puede comprobar que como en el estudio, el grupo más frecuente son aquellos acosos de tipo verbal; según el estudio estas representaban el 76 % de los casos en mujeres. En en el caso de la base de datos la cifra está en torno al 60 % pero habría que tener en cuenta todas aquellas descripciones puestas como "otros". En el estudio la segunda categoría es tocamientos indeseados, esto también ocurre con las descripciones de la base de datos siendo "groping" la segunda más frecuente. Por tanto, se puede ver que la base de datos pese a no abarcar tantos ámbitos como el estudio #MeToo, tiene en general los mismos resultados cualitativos.

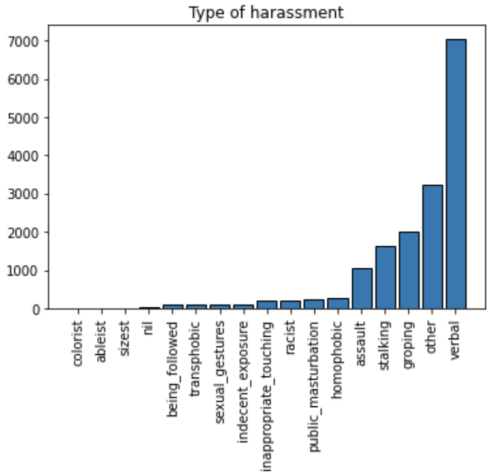


Figura 5.6: Frecuencia del tipo acoso

5.1.5. Denunciado por testigos

Finalmente, en la base de datos existe una categoría para indicar si la descripción ha sido subida o no por un testigo que presenció el incidente. Este dato es un buen indicador de si el testigo ha tomado un rol activo durante el acoso, ya que si denuncian el incidente es probable que hayan intervenido testigos. Podemos ver que apenas un 5 % de las descripciones están subidas por ellos, lo que podría implicar que la mayoría de testigos no toman acción directa.

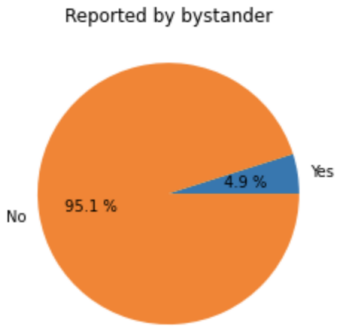


Figura 5.7: Denunciado por testigos

Sin embargo, también es importante ver la relación entre el número de denuncias hechas en un año y el número de denuncias hechas por testigos ese mismo año, de este modo se puede saber si han aumentado en proporción. En la figura 5.8 se puede ver que a lo largo de los años, si descartamos el 2020 por estar todavía incompleto, ha aumentado ligeramente el número de denuncias hechas por los testigos en relación al número de denuncias hechas (figura 5.2). Lo que puede sugerir que la gente está más concienciada.

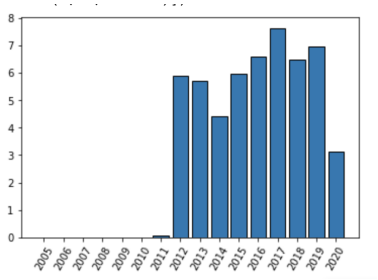


Figura 5.8: Denunciado por testigos

En este proyecto se tratarán las descripciones hechas por víctimas, para poder

detectar en ellas la presencia de testigos. Esto permitirá etiquetar mejor los datos para futuros trabajos con el fin de concienciar a los testigos sobre la importancia de su intervención.

5.2. Resultados del algoritmo de procesamiento del texto

Para poder crear los modelos de clasificación con las descripciones de la base de datos, es importante que el texto sea lo más homogéneo posible. El procesamiento de texto se realiza en una serie de etapas detalladas a continuación, que tienen el objetivo de reducir la ambigüedad sin perder información importante.

5.2.1. Importación de datos

Como se ha dicho previamente el primer paso del algoritmo es la importación del csv al programa. Esto se hace a través del módulo csv [43] y por medio de un diccionario con clave el número de descripción y atributos otro diccionario con la información importante. El resultado son elementos con esta estructura:

```
hour_and_date: 2010-10-04T15:43:58+00:00
longitude: -122.67611111111
latitude: 45.523333333333
categories: ["verbal"]
reported_by_bystander: false
description: Portland, OR is a place where
```

Figura 5.9: Estructura inicial de datos

Cabe destacar que al comienzo del algoritmo se importan todos los datos para poder usarlos en el análisis preliminar. Tras el análisis preliminar se utilizarán únicamente las descripciones para el resto del algoritmo.

5.2.2. Limpieza de datos

Para realizar esta tarea se utilizan dos funciones con dos módulos diferentes. El primero es bs4 de BeautifulSoup [44] y elimina las etiquetas html que pueda

haber en las descripciones. El segundo módulo es Contractions [45] para separar las posibles contracciones del texto y tratarlas como dos palabras. Este proceso tiene el efecto esperado y consigue eliminar hasta las contracciones más complicadas como “gonna” y “wanna”; sin embargo no trata con los genitivos (’s).

5.2.3. Tokenización

Como se ha explicado previamente en las tecnologías, la tokenización es el proceso por el cual se divide un texto en una lista de palabras. Para realizar esta función se usó el módulo wordtokenize de la biblioteca nltk[46]. Al haber eliminado previamente las contracciones se evita que la tokenización separe en función de los apóstrofos y tenga en cuenta las palabras completas, a excepción del ’s que se utiliza en inglés.

5.2.4. Normalización

El proceso de normalización tiene como objetivo hacer todos los tokens iguales y fáciles de procesar. Para ello realiza tres pasos, el primero es convertir la palabra a minúsculas, para que “HOLA” y “hola” no sean consideradas dos palabras diferentes. El segundo consiste en eliminar los genitivos (’s) ya que se comprobó que eran separados durante la tokenización y generaban errores durante el procesado si no eran eliminados. El tercero elimina la puntuación y otros caracteres extraños que hayan podido quedar tras la tokenización; sin embargo, mantiene tildes y letras de otros idiomas por si el procesamiento se hace en otra lengua. En la figura 5.10 se puede un ejemplo del texto tras el tratamiento.

```
Frase de ejemplo:
I wasn't <p> gonna </p> <main> go</main>. However, my father's friend Marco convinced me to go & I went.


Frase tras limpieza:
I was not  going to   go. However, my father's friend Marco convinced me to go & I went.


Frase tras tokinización:
['I', 'was', 'not', 'going', 'to', 'go', '.', 'However', ',', 'my', 'father', "'s", 'friend', 'Marco', 'convinced', 'me', 'to', 'go', '&', 'I', 'went', '.']


Frase tras normalización:
['i', 'was', 'not', 'going', 'to', 'go', 'however', 'my', 'father', 'friend', 'marco', 'convinced', 'me', 'to', 'go', 'i', 'went']
```

Figura 5.10: Resultados del preprocesado

5.2.5. Detección de idioma

Durante la lectura de las descripciones se comprobó que existían en diversos idiomas diferentes, por tanto era imperativo conocer el idioma de cada una para poder tratarlas. Los idiomas más comunes y por tanto los detectados son: inglés, español, francés, italiano y alemán; el resto de las descripciones se guardan como "otros idiomas".

Para hacer esta detección se hizo uso de las stop words de cada diccionario que, como se ha dicho previamente, son las palabras más frecuentes en un idioma. Para cada lengua se contó cuantas stop words había en el texto y se escogió el idioma en el que este número era máximo.

Para tratar con los "otros idiomas" se estableció un recuento mínimo de stop words, si no llegaban pertenecían a la categoría de "otros". Este mínimo empezó siendo 5, pero posteriormente se vio que no funcionaba para textos largos ya que palabras como "a" salen en todos los idiomas. Por tanto, se acabó filtrando en función del tamaño y asignando que mínimo el 10 % de las palabras tenían que ser stop words para pertenecer a un idioma.

Este proceso todavía da algunos problemas con idiomas parecidos entre ellos como el español y el portugués, como se puede comprobar en la imagen 5.11. Si se quisiese más precisión se podría solucionar comparando el diccionario entero en vez de las stop words; sin embargo, sería un proceso muy lento y consumiría muchos recursos.

```
La chica iba corriendo por la calle cuando se le acercó se hacercó a hablar con ella un deconocido.
spanish

The girl was running down the street when she was approached by a stranger who spoke to her.
english

La jeune fille courait dans la rue lorsqu'elle a été approchée par un inconnu qui lui a parlé.
french

La ragazza stava correndo per strada quando fu avvicinata da uno sconosciuto che le parlava.
italian

Das Mädchen rannte die Straße entlang, als sie von einem Fremden angesprochen wurde, der mit ihr sprach.
german

Devushka bezhala po ulitse, kogda k ney podoshel neznakomets, kotoryy govoril s ney.
other

A garota estava correndo pela rua quando foi abordada por um estranho que falou com ela.
spanish
```

Figura 5.11: Resultado de la detección de idiomas

Para este proyecto se utilizan solo las descripciones en inglés, que son las más numerosas en la base de datos; el resto de descripciones son almacenadas para otros usos futuros. Como se ha dicho antes, a veces existen algunos idiomas mal etiquetados debido a las stop words; esto ocurre en el caso del inglés. Debido a que estas palabras sesgarían mucho el algoritmo de clasificación si se introdujesen, es imprescindible eliminarlas. Para ello se introduce una función que comprueba palabra por palabra todas las descripciones etiquetadas como inglés y elimina aquellas palabras que no pertenezcan al diccionario inglés. Esto se hace obteniendo el diccionario completo gracias al modulo words de la biblioteca nltk.corpus. [46]

5.2.6. Eliminación stopwords

Como se ha dicho previamente una fase de gran importancia en el procesado del texto es la eliminación de stop words. De este modo se reduce memoria y además se evita el uso de palabras que salen en todos los textos y no aportan significado. Para ello se eliminan las palabras de las descripciones que coincidan con la lista de stopwords. Esta lista se obtiene a través del módulo stopwords de la biblioteca nltk.corpus,[46].

5.2.7. Lematización y stemming

Este paso se realiza para reducir el número de palabras en memoria, dejarlas con la misma raíz y poder compararlas como palabras iguales. Para realizar este proceso se utiliza el LancasterStemmer y el WordNetLemmatizer de la biblioteca nltk.stem.[46]

Durante el proceso se comprobó que el stemmer funcionaba bien; sin embargo, el lematizador necesitaba información sobre el tipo de palabra a lematizar para tener más exactitud. Eso hizo que se implementase una función para detectar el tipo de palabra gracias al modulo postag de nltk.[46]. En la figura 5.12 se puede ver el resultado de la lematización y stemming.

```
Ejemplo: I was walking with my best friend to go pick up my little brother from elementary school. I saw a car pull up r
Ejemplo stemming: ['walk', 'best', 'friend', 'go', 'pick', 'littl', 'broth', 'el', 'school', 'saw', 'car', 'pul', 'real'
Ejemplo lematización: ['walk', 'best', 'friend', 'go', 'pick', 'little', 'brother', 'elementary', 'school', 'saw', 'car'
```

Figura 5.12: Resultado de stemming y lematización

5.2.8. Bolsa de palabras

Como se dijo previamente uno de los modos de representación de documentos es el Bag of Words. En este caso se aplicó para poder tener una idea de cuales eran las palabras más frecuentes en los textos. Para ello se utilizaron dos herramientas: wordcloud [47] que nos permite una visión más agradable de las palabras más importantes y FreqDist de nltk [46] que permite saber exactamente el recuento.

En la figura 5.13 se puede ver que verbos como decir y andar son las palabras más frecuentes, con alrededor de 10000 veces en los 8000 documentos en inglés, lo que sugiere que a la mayoría de las personas les ocurrió algo mientras andaban o que alguien les habló. En el quinto puesto se encuentra la palabra hombre, lo que indica que en la mayor parte de los casos un hombre desconocido hizo algo. Esto tiene sentido tratándose de una base de datos de denuncias de acoso donde la mayoría están hechas por mujeres.

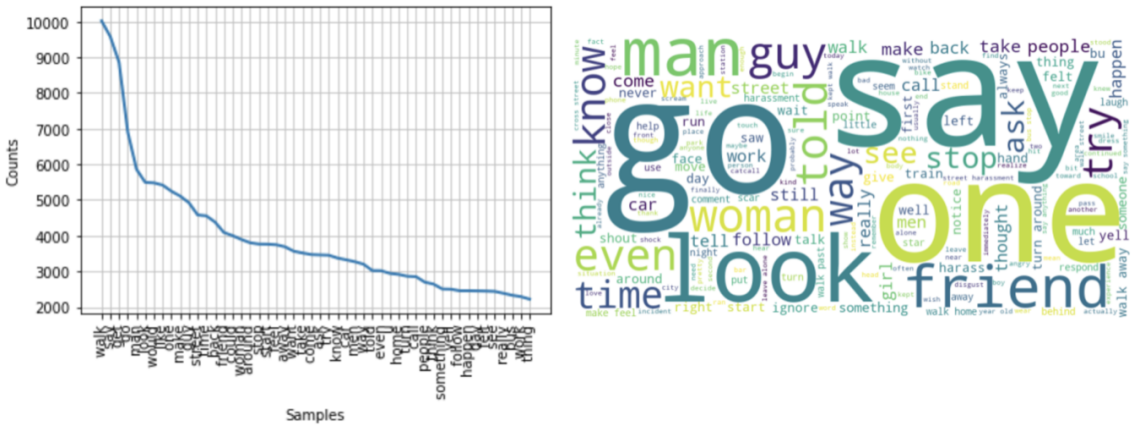


Figura 5.13: Resultado de la bolsa de palabras

5.2.9. TF-IDF

Otra forma de representar el documento, fue a través de la técnica TF-IDF que, como se dijo antes, nos permite ver que palabras son más destacables en un documento, en relación a otros. Al igual que la técnica de bolsa de palabras, esta representación se utilizará más adelante como input para los algoritmos. Se puede ver su uso en la figura 5.14, donde se muestran las palabras más representativas de la descripción en términos de tf-idf.

```
['place', 'actually', 'encounter', 'little', 'sexual', 'harassment', 'street', 'grateful', 'able', 'exist',
'public', 'without', 'visual', 'appraisal', 'week', 'stop', 'expect', 'one', 'day', 'walk', 'bike', 'fairly',
'unpopulated', 'sidewalk', 'middle', 'afternoon', 'man', 'walk', 'behind', 'caught', 'say', 'disrespectful',
'pretty', 'backside', 'ugh', 'bike', 'way', 'block', 'yell', 'back', 'disrespectful', 'could', 'hear', 'murmur',
'well', 'still', 'pretty', 'submit']

disrespectful
appraisal
unpopulated
visual
bike
```

Figura 5.14: Resultado de tf-idf

5.2.10. Latent semantic análysis

Como se ha dicho previamente, LSA es una técnica de reducción de dimensionalidad para NLP que se basa en el uso del singular value descomposition, más concretamente la reducción se hace a través del SVD truncado. En este proyecto se aplica el LSA para ver como quedaría el texto dividido en 2 temas generales en función a las palabras contenidas. Esta reducción se hace con el modelo TruncatedSVD de la parte de descomposición de sklearn [48] y con TF-IDF como representación de textos.

Tras su aplicación se obtienen los siguientes resultados: las palabras más importantes y características del tema 0 son: walk, say, man, guy, look, like, street; y las del tema 1 son: walk, car, yell, street, drove, hey, shout. Y se puede ver que los textos quedan divididos en dos categorías, siendo la 0 más mayoritaria que la 1. En la imagen 5.15 se muestra un ejemplo donde la primera frase pertenecería en un 17 % a la categoría 0 y en un 0.5 % a la categoría 1

	0	1
place actually encounter little sexual harassment street grateful able exist public without visual appraisal week stop expect one day walk bike fairly unpopulated sidewalk middle afternoon man walk behind caught say disrespectful pretty backside ugh bike way block yell back disrespectful could hear murmur well still pretty submit	0.173634	0.067673
walk get coffee man corner make sound walk back home start monologue look good kept go shout behind shut hand full coffee muffin key would take picture ridiculous submit	0.170807	0.055620

Figura 5.15: Resultado de LSA

Esta descomposición se puede representar en el espacio de dos dimensiones, como se aprecia en la figura 5.16. Solo se representan 100 descripciones para no complicar la visualización en el gráfico.

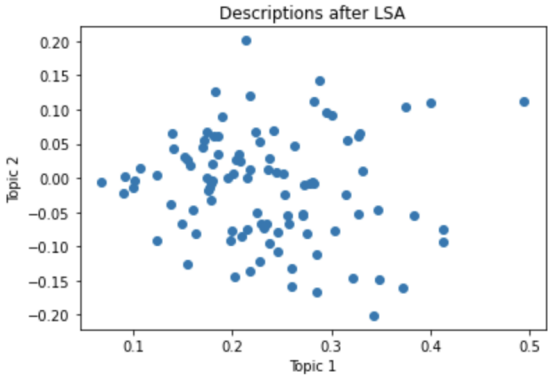


Figura 5.16: Resultado de la representación de las 100 primeras descripciones

5.2.11. LDA

Otra técnica de reducción de dimensionalidad basada en machine learning es LDA, que se implementa a través de la biblioteca gensim [17] y usando el modelo BoW. En este caso nuevamente se intenta separar los textos en dos temas gracias a palabras con pesos ponderados, el resultado es: **tema 0: 0.020*walk + 0.015*say + 0.014*get + 0.011*man + 0.011*go, y tema 1: 0.014*say + 0.013*walk + 0.013*get + 0.012*go + 0.009*make**

Se puede ver que los temas de la categoría 0 coinciden en gran medida con los del LSA. La categoría 1 difiere y tiende a repetir palabras de la categoría 0. Esto puede estar causado por el uso del BoW en vez del TF-IDF.

Finalmente, es importante destacar que la división natural en dos temas no se hace en función del criterio deseado que es que haya testigo o no. Por tanto se tendrán que implementar algoritmos de clasificación con aprendizaje supervisado para poder realizarla.

5.2.12. Word2Vec

Como se dijo previamente, el word embeding es una técnica que consiste en representar con vectores y forma similar, palabras con significados parecidos. La técnica más conocida de word embeding es Word2Vec y durante el proyecto se aplicó a la base de datos para ver la representación de estas palabras. Para ello se utilizó la biblioteca models de gensim.[17]

El resultado es que cada palabra tiene una representación vectorial de 100

dimensiones, pasadas como parámetro al algoritmo. Se puede ver un ejemplo para la palabra "beautiful" en la figura 5.17

```
[-0.5916291  -1.3135731   0.07636797 -0.878296    0.69156015  0.49097985
  0.8363047  -0.2005922  -1.1527596   0.6799757  -0.19309519  0.555235
 -0.9533119   0.41384482  0.61732936  0.08811411 -0.29886287  0.19298612
  0.16975226  0.6413581  -0.12439906  1.2338562  -0.7686942  -0.17625698
  0.35185164  0.59442985 -0.3326274  -1.0014902   1.3514202   1.1579859
 -0.7668243   1.0110258  -0.42515978  0.53915787 -0.51393723 -0.47680923
  0.27265772  1.0580481  -0.5305022  -0.12030201 -1.015835    0.71569896
  1.1594025  -0.9429615   0.52370507 -0.17292602 -1.2233396   0.4141339
  0.4797607   0.4084016   0.3161362  -0.14923307 -0.3570145  -0.30425775
 -0.40437946 -0.5428885   0.11020811 -0.29307142 -0.6897252   0.5548541
  0.8369494   0.41655448 -0.7151269   0.5964238  -0.88060755 -0.9102045
 -0.5311429   0.21459942  0.05726774  1.562365   -1.8381716   0.21185818
 -0.00430986 -0.22943443  0.24835272 -0.5961513   0.79111296  0.47132623
  0.13766564 -0.22203478  0.7546793  -0.78878546 -0.13025725  0.16434067
  0.6145185  -1.1796758   0.22600156 -0.00587703 -0.34899253  1.1723247
 -0.70265406 -0.55084205 -1.2238903   1.9388906  -0.13553776  0.24971437
 -0.20714605  0.5145178   1.3294437   0.14213131]
```

Figura 5.17: Representación en 100 dimensiones de la palabra "beautiful"

La técnica word2vec resulta muy útil si se quiere encontrar palabras similares a una dada, y por tanto palabras que se encontrarán en mismos contextos. Por ejemplo, si se buscase las palabras más similares a "beautiful" se obtendría el resultado de la figura 5.18

```
[('gorgeous', 0.9526709318161011), ('baby', 0.9347470998764038),
('hello', 0.916921854019165), ('nice', 0.9122025966644287),
('sexy', 0.90898597240448), ('damn', 0.8988128900527954), ('hey',
0.8908101320266724), ('hi', 0.8842440247535706), ('babe',
0.879196047782898), ('cute', 0.8746659755706787)]
```

Figura 5.18: Palabras más similares a "beautiful"

Se puede ver que el algoritmo acierta en gran medida con sinónimos como "gorgeous","nice","sexy","cute". Pero que sin embargo, incluye algunas palabras que suelen estar directamente relacionadas con la palabra pero no son similares a ella. Por ejemplo, "hi" muy usada en "hi beautiful" o "babe" por frases como "beautiful babe". Estas expresiones se pueden encontrar a menudo en las descripciones y por tanto pese a no ser sinónimos el algoritmo las considera similares.

5.3. Resultados de los modelos de clasificación

Se han creado modelos de clasificación para tres escenarios distintos: descripciones completas, descripciones simplificadas y ambas descripciones. Para cada escenario se han desarrollado cinco modelos de clasificación: regresión logística, naive bayes, máquinas de vector de soporte, clasificador SGD (descenso de gradiente estocástico) y k vecinos más próximos.

5.3.1. Modelos de clasificación con descripciones completas

Al tratarse de modelos de aprendizaje supervisado, resultaba imprescindible tener un número de descripciones etiquetadas para construir el modelo. Por tanto se leyeron y etiquetaron 99 descripciones de la base de datos, de ellas 33 tenían presencia de testigo y 66 no la tenían. Estás se dividieron de forma equilibrada en dos conjuntos, el de entrenamiento y el de prueba. A las descripciones se les aplicaron dos tipos de extracción de características: tf-idf y BoW, del paquete feature-extraction de sklearn [48] y a las etiquetas se les aplicó un modelo binario con el módulo LabelBinarizer de la biblioteca preprocessing de sklearn. Tras ajustar y calcular varias veces los modelos se comprobó que los parámetros más eficientes para la extracción de características eran el mínimo y máximo df a 0 y 1 y el número de n-gramas limitado a 1.

Posteriormente se construyeron distintos modelos de clasificación para comprobar cual es el más eficiente para la base de datos dada. Estos modelos son: regresión logística, multinomial naive bayes, máquinas de vector de soporte (SVM), K vecinos más próximos y un clasificadorSGD. La evaluación de los modelos se basó en las métricas relacionadas con la categoría "con testigo". Además se priorizó que los modelos tengan precisión (los elementos detectados en una categoría realmente pertenecen a ella) que sensibilidad (que detecte la mayoría de casos), ya que las descripciones se usarán posteriormente para evaluar el sentimiento y es preferible que haya menos descripciones pero que estas estén bien clasificadas y no haya datos falsos.

5.3.1.1. Regresión logística

El primer modelo que se utilizó fue el de regresión logística, para ello se utilizó el módulo LogisticRegression perteneciente a la biblioteca linear models de sklearn[48]. Se estableció un número máximo de 500 iteraciones y una penalización l2 y se aplicó el modelo a los datos en BoW y TF-IDF. Tras ello se calcula la matriz de confusión:

$$\begin{pmatrix} 33 & 0 \\ 14 & 3 \end{pmatrix} \tag{5.1}$$

Esta matriz representa en las filas los valores reales y en las columnas los predichos, el primer valor corresponde con la categoría "sin testigo" y el segundo la categoría "con testigo". Por ejemplo, en este caso habría 14 descripciones etiquetadas como "sin testigo" que en realidad eran "con testigo".

Gracias a la matriz se pueden calcular las métricas y se comprueba que en el modelo con BoW la exactitud es alta (72 %). Sin embargo, tiene una sensibilidad muy baja, del 17 % para las descripciones con testigo (solo 17 de cada 100 descripciones "con testigo" son clasificadas como tal). También tiene una precisión total (100 %) para la categoría "con testigo", osea que si el algoritmo incluye la descripción en esa categoría, estará bien clasificada. El modelo con tf-idf tiene menor exactitud 68 % y el mismo problema de sensibilidad, con una matriz de confusión:

$$\begin{pmatrix} 33 & 0 \\ 17 & 0 \end{pmatrix} \tag{5.2}$$

Se puede comprobar que la regresión logística no es muy buen clasificador para los datos de origen.

5.3.1.2. Naive Bayes

El segundo modelo utilizado es naive bayes también perteneciente a la biblioteca sklearn. En este caso el modelo BoW tiene una exactitud del 66 % y una matriz

$$\begin{pmatrix} 28 & 5 \\ 12 & 5 \end{pmatrix} \tag{5.3}$$

Este modelo pese a tener peor exactitud y precisión (50 %), tiene una sensibilidad mayor (30 %) para la categoría "con testigo" que el modelo LR.

En el caso del TF-IDF se obtienen los mismos resultados que en LR, es decir, una exactitud del 66 % y una matriz:

$$\begin{pmatrix} 33 & 0 \\ 17 & 0 \end{pmatrix} \tag{5.4}$$

Por tanto el modelo NB para bow tiene una eficiencia parecida al LR, mejorando la sensibilidad pero empeorando la precisión. Para tfidf sigue siendo un mal clasificador.

5.3.1.3. Máquinas de vector de soporte

El tercer modelo creado fue el SVM y también se implementó con la biblioteca sklearn. Para este modelo en el caso del BoW, la exactitud aumenta al 72 % y la matriz de confusión es:

$$\begin{pmatrix} 28 & 5 \\ 9 & 8 \end{pmatrix} \tag{5.5}$$

Se puede ver que la precisión aumenta al 61 % y la sensibilidad al 47 %. por ello es un modelo mejor que los anteriores.

En el caso del tfidf la matriz es:

$$\begin{pmatrix} 32 & 1 \\ 15 & 2 \end{pmatrix} \tag{5.6}$$

Por lo tanto la precisión es 66 % y la sensibilidad 11 %, al ser mucho menores que en el caso anterior es un modelo peor.

5.3.1.4. Clasificador con descenso de gradiente estocástico

El cuarto modelo es el llamado SGDClassifier. En realidad el SGD es simplemente una técnica de optimización que se aplica dentro del modelo a un clasificador. Este clasificador se puede cambiar en los parámetros, en este caso se utiliza la opción loss='hinge', es decir, será un SVM. Por tanto, se trata de un modelo como el anterior pero con un tipo de optimización diferente. Para elaborar este clasificador se utiliza el modelo SGDClassifier de sklearn. En este caso la matrices de confusión tanto del bow y el tfidf son similares a la del tfidf del modelo SVM. Por tanto también tienen poca precisión y sensibilidad y no es un buen clasificador.

$$\begin{pmatrix} 32 & 1 \\ 15 & 2 \end{pmatrix} \tag{5.7}$$

5.3.1.5. K vecinos más próximos

El último modelo usado es el de K vecinos más próximos y es implementado con el módulo KNeighborsClassifier de sklearn. En este caso tanto para el modelo bow como para el modelo tfidf se obtiene una matriz de confusión:

$$\begin{pmatrix} 30 & 3 \\ 12 & 5 \end{pmatrix} \tag{5.8}$$

Podemos ver que tiene más precisión (63 %) que el modelo SVM, pero menos exactitud (70 %) y menos sensibilidad (29 %) por lo que sería un modelo peor que el SVM pero mejor que el clasificador SGD, LR y naive bayes.

5.3.1.6. Resumen

Para poder hacer una comparación más precisa, se han recogido en la tabla 5.19 y en el gráfico 5.20 todas las métricas de los modelos expuestos. Estas métricas están calculadas en función de las matrices de confusión de cada modelo. Las siglas ST y CT hacen referencia a las categorías "con testigo" o "sin testigo". A pesar de que la tabla recoge información de ambas categorías, la que se tendrá en cuenta es aquella "con testigo". Los modelos evaluados son la Regresión logística (LR), multinomial naive bayes (NB), support vector machine (SVM), stochastic gradient descendant (SGD) y K nearest neighbours (KNN).

Modelo	Exactitud	Precisión ST	Precisión CT	Sensibilidad ST	Sensibilidad CT	F1-score ST	F1-score CT
LR Bow	0,72	0,70	1,00	1,00	0,17	0,82	0,29
LR tfidf	0,66	0,66	-	1,00	0,00	0,80	-
NB Bow	0,66	0,70	0,50	0,85	0,30	0,77	0,38
NB Tf-idf	0,66	0,66	-	1,00	0,00	0,80	-
SVM bow	0,72	0,75	0,61	0,85	0,47	0,80	0,53
SVM tfidf	0,68	0,68	0,66	0,97	0,11	0,80	0,19
SGD bow	0,68	0,68	0,66	0,97	0,11	0,80	0,19
SGD tfidf	0,68	0,68	0,66	0,97	0,11	0,80	0,19
KNN bow	0,70	0,71	0,63	0,90	0,29	0,79	0,40
NKK tfidf	0,70	0,71	0,63	0,90	0,29	0,79	0,40

Figura 5.19: Métricas de modelos con descripciones completas

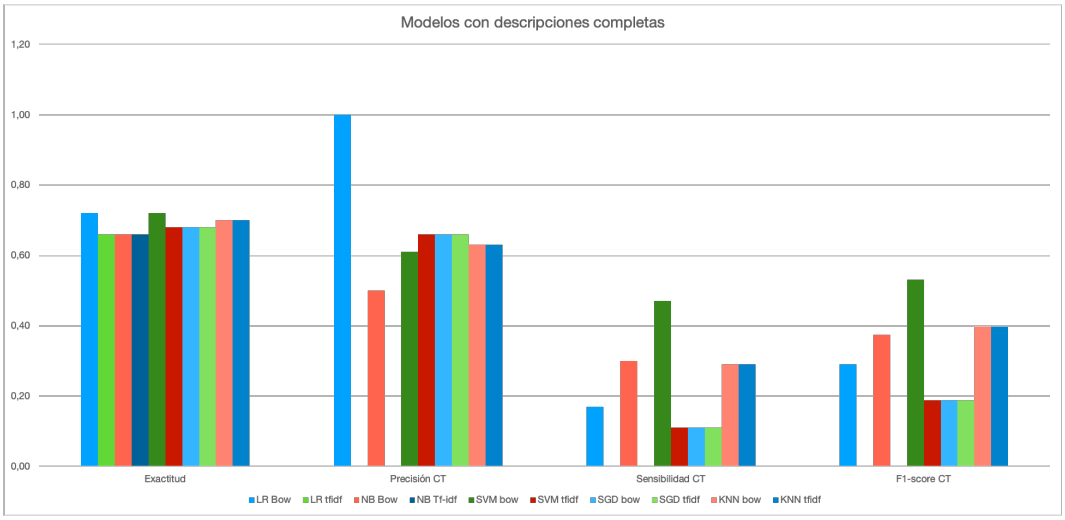


Figura 5.20: Métricas de modelos con descripciones completas

Se puede ver que todos los modelos tienen una exactitud alrededor del 70 %. Además, salvo la regresión logística y naive bayes, también tienen una precisión

en torno el 63 % para el modelo con testigo. Esto quiere decir que el 63 % de las veces un elemento sacado de la clase con testigo estará bien clasificado. También se puede ver que la mayoría de modelos tienen baja sensibilidad para detectar testigos, siendo el más alto el SVM para el BoW con un 47 %. Esto querría decir que teniendo una descripción testigo, el algoritmo tiene un 47 % de posibilidades de categorizarlo como tal. Otro estimador que se tendrá en cuenta en gran medida es el F1-score que permite relacionar precisión y sensibilidad, su fórmula se encuentra en la ecuación 3.25. Fijándose en este parámetro para los modelos, se puede comprobar que el mejor sería el SVM con la representación BoW.

5.3.1.7. Complejidad de las descripciones

Se puede ver que aunque la mayoría de algoritmos tienen una precisión y exactitud aceptable, tienen poca sensibilidad. Se piensa que esto puede ser debido a la complejidad y gran extensión de las descripciones, en las que apenas dos palabras de 100 sirven para indicar la presencia de testigos y por tanto resulta difícil para un algoritmo clasificarlas.

Para entender mejor esa complejidad, en la figura 5.21 se muestra una descripción real, bien clasificada. En rojo están marcadas las palabras que indican la presencia de un testigo. Se puede ver que estas suponen una pequeña parte de la descripción, aun así el modelo es capaz de clasificarlo porque palabras como "full" o "everyone" son palabras muy distintivas. Sin embargo, la mayoría de descripciones no cuentan con palabras con tanto peso y por tanto el modelo tiene problemas para identificarlas como "con testigo".

Algunos de los problemas de precisión también están asociados a la complejidad de las descripciones; por ejemplo, cuando aparece la palabra "friend", es difícil saber si el amigo era testigo o era víctima o agresor. Además palabras como "boyfriend" o "girlfriend", quedan reducidas a "friend" durante la lematización lo que genera aún más ambigüedad. Este tipo de descripciones suele clasificarse como positiva, dando positivos reales como en el ejemplo de la imagen 5.22, o falsos positivos, como en la imagen 5.23. Esos falsos positivos provocan una reducción en la precisión.

Por tanto se puede ver que existe una gran complejidad en las descripciones que provoca falsos positivos y falsos negativos y hace que la clasificación no sea totalmente eficiente. Para tratar de solucionarlo se decidió crear modelos con descripciones simplificadas que permitiesen entrenar los algoritmos únicamente con palabras claves.

Like every other woman, I have been harassed countless times, but this particular instance really creeped me out. I was on a **fairly full** L train headed to Brooklyn when I noticed this old guy sitting across staring very intently at me. There's a lot of creepy guys out there, but this one, I kid you not, looked like he would be a creep from a movie. I have never seen a more disturbing looking individual with burning, glazed over, angry-drunken eyes in my life.

So as the train pulls into the next stop, he gets up and sits down right next to me. This would be the day that I actually don't have my iPod with me. Then, he says "You're very pretty." I ignore him, and he escalates it louder, "I SAID, you're VERY PRETTY." Again, I ignore him. At this point, he is leering right up in my face and follows with, "What do you SAY WHEN SOMEONE GIVES YOU A COMPLIMENT? HUH? ANSWER ME!"

To stop his banter I finally say "Thank you." I'm looking around the train and **everyone is just staring**, and watching this all take place without, of course, doing anything. The guy is getting more enraged by the minute, and I am trying to hide my physical shaking. He continues bantering me with "I GAVE YOU A COMPLIMENT! YOU COULDN'T EVEN SAY THANK YOU, YOU FUCKING BITCH! YOU GONNA TALK TO ME? YOU GONNA TALK TO ME?" This man is shouting in my face. I was torn as to whether it was safe for me to even get up and move to the next car, since he was getting increasingly irate but I just couldn't take it and got up a ran off the train when it pulled to the next stop, and as I look back he's still screaming at me and running to try and follow me, but luckily the doors had just shut.

Like I said, I've been harassed plenty of times before, and I've been chased for a short distance, but never have I been so frightened like I was in this incident.

Figura 5.21: Ejemplo de descripción completa

I was just trying to throw out my trash. it was a bit late around 9pm. we live on a main street and there are always people going by. While walking to the trash I hear someone call "hey, hey girl." I turn around and this guy on a bike is riding up my driveway. I know I looked terrified and begun to start walking towards the door. He keeps coming at me saying " let me talk to you come here." I'm like "nah" and get up on the porch thats when my **boyfriend come outside**. The dude looks shocked and finally starts to back up to the street again. my boyfriend shouts" hey homie WTF do you want." the guy just says " just trying to sell some tools." and heads off. Cant even be on your own property these days

Figura 5.22: Ejemplo de descripción completa con positivo real

11/27/2010, around 1pm. I was walking with my **friend** V. on Broadway in uptown Manhattan (just past Nagle Avenue, along the wall of Fort Tryon Park) when we encountered four young men, around 13 or 14, perched on the wall and benches hanging out. One kid yelled out at us, saying "your girlfriend's cute!" The others laughed and chimed in, yelling at V., and started asking "is she your sister or girlfriend?"

Feeling a bit defensive and reckless (I honestly wanted to answer "both" just to see their reactions) I lied and said yes, and they all started hooting "tap that ass!" and "kiss, kiss, kiss." They shouted at us down the rest of the block; it was awkward, but we both took it in good humor. I would hope that they're just being reckless young kids and they'll grow out of that kind of behavior soon.

Figura 5.23: Ejemplo de descripción completa con falso positivo

5.3.2. Modelos de clasificación con descripciones simplificadas

Como se ha dicho, tras elaborar los modelos con las descripciones completas, se comprobó que muchos no eran muy sensibles a las descripciones con testigo.

Se pensó que esto podía deberse a que las descripciones completas eran muy largas y complejas. Por ello se decidió crear una base de datos alternativa, cogiendo las expresiones más representativas que utilizan las víctimas para indicar la presencia de testigos y otras inventadas; y expresiones que pudiesen sugerir que no había testigos. La base de datos creada manualmente tiene solo dos campos, las descripciones y las etiquetas, un ejemplo se muestra en la figura 5.24.

Description	Intervention
Because there were other people in the elevator	1
Because there weren't any other people	0
There was no one	0
There was someone	1
Two ladies were next to him	1
It involved bystanders	1

Figura 5.24: Base de datos simplificada

Esta base de datos está compuesta por 97 descripciones etiquetadas de las cuales 55 indican presencia de testigos y 42 indican la no presencia. Al igual que con las descripciones completas, estas son representadas como bag of words y tfidf y luego introducidas a todos los modelos usados previamente.

Para evitar detallar caso por caso como se ha hecho antes y debido a la similitud del proceso, se exponen en las figuras 5.25 y 5.26 la tabla y gráfico con los resultados obtenidos a través de las matrices de confusión.

Modelo	Exactitud	Precisión ST	Precisión CT	Sensibilidad ST	Sensibilidad CT	F1-score ST	F1-score CT
LR Bow	0,69	0,75	0,67	0,43	0,89	0,55	0,76
LR tfidf	0,69	0,75	0,67	0,43	0,89	0,55	0,76
NB Bow	0,69	0,75	0,67	0,43	0,89	0,55	0,76
NB Tf-idf	0,69	0,75	0,67	0,43	0,89	0,55	0,76
SVM bow	0,69	0,75	0,67	0,43	0,89	0,55	0,76
SVM tfidf	0,69	0,75	0,67	0,43	0,89	0,55	0,76
SGD bow	0,57	0,50	0,77	0,85	0,35	0,63	0,48
SGD tfidf	0,57	0,50	0,77	0,85	0,35	0,63	0,48
KNN bow	0,67	0,69	0,67	0,43	0,85	0,53	0,75
KNN tfidf	0,69	0,75	0,66	0,43	0,89	0,55	0,76

Figura 5.25: Métricas de modelos con descripciones simplificadas

Se puede ver que menos el SGDClassifier que da resultados peores, todos los modelos tienen las mismas métricas. Tienen una exactitud del 69 %, una precisión

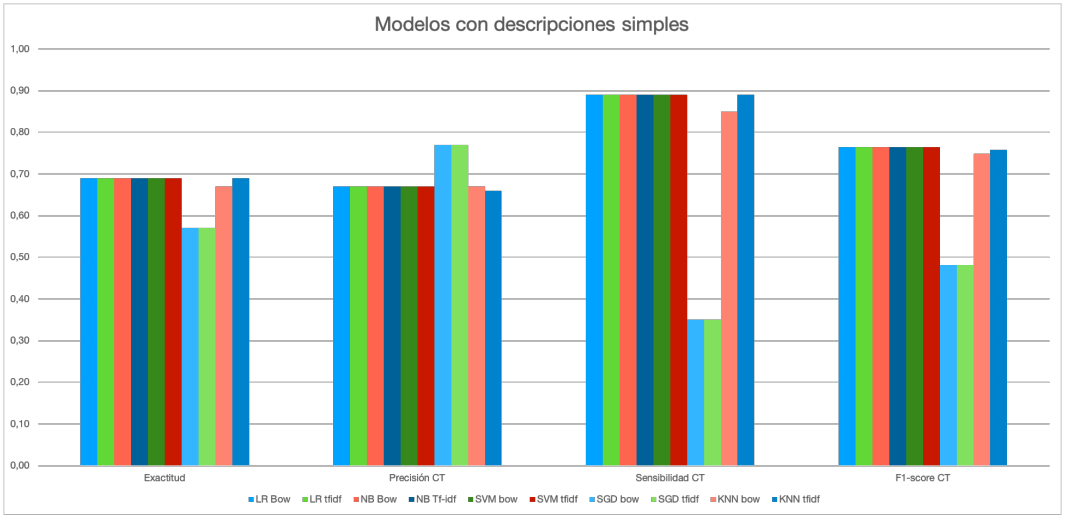


Figura 5.26: Métricas de modelos con descripciones simplificadas

un poco más alta que los modelos con descripciones completas (67 %) y una sensibilidad mucho mayor (89 %). Esto quiere decir que para descripciones cortas, el 89 % de las veces si la descripción tiene testigos, la clasificará bien y el 67 % de las veces si se saca una descripción de las clasificadas como con testigo, estará bien clasificada. Combinando estas dos métricas podemos ver que tienen también un F1-score mayor (76 %) que el 53 % que daba el mejor modelo con las descripciones completas.

Como se ha visto las descripciones simplificadas dan mejor resultado que las descripciones completas a la hora de entrenar y probar los modelos. Sin embargo, el objetivo del proyecto es clasificar las descripciones existentes en la base de datos. Por ello el siguiente paso fue probar a aplicar los modelos simplificados y aparentemente más eficientes, a las descripciones completas.

5.3.3. Modelos de clasificación con descripciones simplificadas aplicados a descripciones completas

En este apartado se analizará la efectividad de los modelos elaborados con descripciones simplificadas para predecir la categoría de las descripciones completas. Nuevamente para evitar entrar en detalles sobre cada modelo, se presentarán y analizarán la tabla de métricas y gráfico obtenidos por medio de las matrices de confusión.

Modelo	Exactitud	Precisión ST	Precisión CT	Sensibilidad ST	Sensibilidad CT	F1-score ST	F1-score CT
LR Bow	0,51	0,95	0,40	0,28	0,96	0,43	0,56
LR tfidf	0,40	1,00	0,36	0,12	1,00	0,21	0,53
NB Bow	0,62	0,79	0,46	0,59	0,69	0,68	0,55
NB Tf-idf	0,60	0,96	0,45	0,42	0,96	0,58	0,61
SVM bow	0,60	0,96	0,45	0,42	0,96	0,58	0,61
SVM tfidf	0,51	0,85	0,40	0,29	0,96	0,43	0,56
SGD bow	0,42	1,00	0,36	0,13	1,00	0,23	0,53
SGD tfidf	0,50	1,00	0,40	0,26	1,00	0,41	0,57
KNN bow	0,66	0,76	0,49	0,71	0,55	0,73	0,52
KNN tfidf	0,42	1,00	0,37	0,14	1,00	0,25	0,54

Figura 5.27: Métricas de modelos simples aplicados a descripciones completas

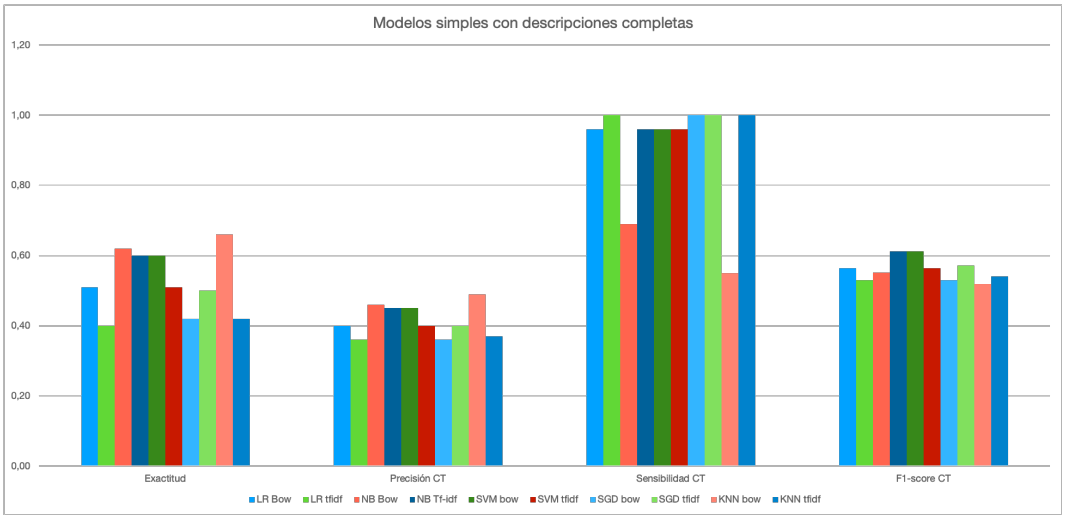


Figura 5.28: Métricas de modelos simples aplicados a descripciones completas

En este caso se puede comprobar que los modelos han mejorado mucho su puntuación F1 con respecto a los modelos con descripciones completas, debido a un gran aumento de la sensibilidad. Sin embargo, la exactitud y precisión han bajado en gran medida teniendo una precisión para la categoría con testigo de aproximadamente 40 %. Esto implicaría que solo el 40 % de los elementos de la categoría con testigo estarían bien clasificados. Estos modelos no podrían ser utilizados para el proyecto ya que habría muchos datos falsos, como se ha dicho antes, resulta preferible un modelo poco sensible pero que los datos que cojan si sean precisos.

Viendo que unos modelos tenían precisión y otros sensibilidad, se se intentó buscar una solución mixta.

5.3.4. Modelos de clasificación con descripciones mixtas

En este apartado se buscó analizar el efecto de entrenar el modelo con los datos simplificados y con una parte de los datos completos. La otra parte de las descripciones completas se guardará para hacer el test de los modelos. Las métricas de cada uno de los modelos están resumidas en la siguiente tabla y gráfico:

Modelo	Exactitud	Precisión ST	Precisión CT	Sensibilidad ST	Sensibilidad CT	F1-score ST	F1-score CT
LR Bow	0,58	0,67	0,35	0,72	0,29	0,69	0,32
LR tfidf	0,58	0,65	0,33	0,75	0,23	0,70	0,27
NB Bow	0,60	0,74	0,43	0,60	0,59	0,66	0,50
NB Tf-idf	0,62	0,73	0,45	0,67	0,52	0,70	0,48
SVM bow	0,64	0,73	0,47	0,73	0,47	0,73	0,47
SVM tfidf	0,58	0,66	0,33	0,76	0,24	0,71	0,28
SGD bow	0,56	0,69	0,38	0,60	0,47	0,64	0,42
SGD tfidf	0,56	0,69	0,38	0,60	0,47	0,64	0,42
KNN bow	0,34	-	0,34	0,00	1,00	-	0,51
KNN tfidf	0,34	-	0,34	0,00	1,00	-	0,51

Figura 5.29: Métricas modelos mixtos aplicados a descripciones completas

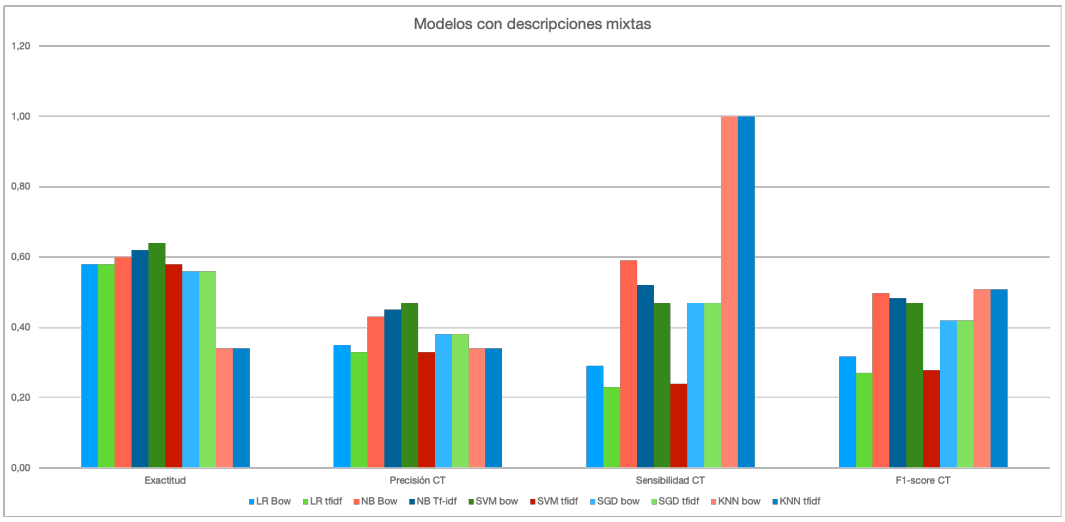


Figura 5.30: Métricas modelos mixtos aplicados a descripciones completas

En este caso se puede comprobar que tanto la precisión como la sensibilidad han disminuido en gran medida, por lo que este modelo pese a haber mezclado dos bases de datos que funcionaban bien, es el menos eficiente de todos.

6. Conclusiones

6.1. Conclusiones del proyecto

Del análisis exploratorio de datos se pueden sacar distintas conclusiones que resultan de gran importancia para el proyecto. La primera es que el **idioma predominante es el inglés** y que por tanto el análisis de texto tiene que realizarse en este idioma para poder sacar el máximo provecho a la base de datos. La segunda es la **baja tasa de denuncias hechas por testigos**, esto demuestra que sí existe la necesidad de concienciar a la gente sobre la importancia de intervenir como testigo y se reafirma la importancia del desarrollo de este proyecto. Otras conclusiones que se pueden extraer es que hay un **reparto desigual de las denuncias en el mundo**, lo que puede sugerir la necesidad de expandir el alcance de la plataforma a lugares donde no se conoce y podría ser necesaria. Además, las garantías de anonimato deberían ayudar a romper las barreras culturales y que se realicen más denuncias en todo el mundo.

De la parte de procesamiento del texto se puede concluir que el algoritmo tiene el comportamiento deseado y que además es suficientemente genérico como para poder soportar cualquier tipo de descripción o texto en general. Por tanto puede ser reutilizable para otros proyectos.

De la parte de NLP básico se puede concluir que la división natural del texto al estar hecha por algoritmos de aprendizaje sin supervisar, se basa en la proximidad semántica y por tanto genera unas categorías totalmente diferentes a las deseadas que son "con testigo" o "sin testigo". Este hecho demuestra la **necesidad de etiquetar las descripciones** y de incluir algoritmos de clasificación binaria basados en aprendizaje supervisado para poder fijar las categorías deseadas.

De los algoritmos de clasificación se pueden extraer múltiples conclusiones. La primera es que **el modelo más eficiente para las descripciones completas**

es el de SVM con una representación BoW del documento. Tiene un 72 % de exactitud, un 61 % de precisión y un 47 % de sensibilidad para la categoría de descripciones con testigo. Este modelo podría ser usado como clasificador general de las descripciones.

Los segundos modelos son aquellos creados con descripciones simplificadas con el fin de mejorar los primeros modelos. En ellos se puede concluir que, menos el clasificador SGD, **todos los modelos son mejores que el de descripciones completas**. Esto se debe a que tienen una exactitud del 68 %, una precisión del 67 % y una sensibilidad del 89 %. Sin embargo, cuando se usan estos modelos para clasificar las descripciones completas se observa que aunque aumenta la sensibilidad casi al 100 %, la precisión disminuye al 45 %. Por ello se puede concluir que no son buenos modelos para la clasificación de descripciones, ya que se prioriza mayor precisión y menor sensibilidad.

Finalmente, en el modelo entrenado con ambas descripciones, disminuye tanto la precisión como la sensibilidad y por tanto se concluye que sería el peor de los modelos.

6.2. Futuros avances

Continuación del proyecto

Como se ha dicho a lo largo de toda la memoria, este proyecto forma parte de uno más grande de tipo sociológico cuya intención es comprobar el efecto de los testigos sobre las víctimas de acosos. Por tanto la continuación natural de este proyecto es aplicarlo para discriminar las descripciones con testigo, y sobre esas descripciones aplicar análisis de sentimientos. Una vez hecho esto se podrían realizar estadísticas con los comportamientos más beneficiosos y más nocivos, y sacar una guía del testigo actualizada que recoja los nuevos descubrimientos hechos a través del análisis de texto.

Mejora de los modelos con descripciones completas

Los modelos creados con descripciones completas se podrían mejorar de diversos modos. El primero sería entrenándolo con más datos, es decir, más descripciones etiquetadas. El problema de este método es que se requeriría etiquetar alrededor de 1000 descripciones. Esto supone una carga de trabajo importante, pero mejoraría significativamente el rendimiento del algoritmo.

Otra solución para evitar tener que hacer el etiquetado de las descripciones, sería incluir en la plataforma web una casilla donde se marcase si ha habido un testigo. De este modo las nuevas descripciones estarían directamente etiquetadas y solo habría que pasarlas por el algoritmo de procesamiento de texto e introducirlas en los modelos para entrenarlos. Esto tiene el problema de que se tardaría en recuperar nuevas descripciones y que en ciertas ocasiones sería complicado para la víctima saber si considerarlo testigo o no.

Por lo que se concluye que la solución más sencilla con los datos actuales es aumentar el número de descripciones utilizadas para el entrenamiento y requiere dedicarle tiempo.

Mejora de los modelos con descripciones parciales

Para mejorar los modelos con descripciones parciales también se podría aumentar el número de descripciones utilizadas para entrenar el modelo. Para ello habría que extraer más expresiones de las descripciones existentes, encontrar otras descripciones en el lenguaje y desarrollar un modo de indicar de forma sencilla que no había testigo. Este proceso será largo y complejo si se quiere hacer bien, y probablemente requiera de alguien nativo y con un amplio vocabulario y conocimiento de la materia.

Otra solución sería hacer más parecidas estructuralmente las descripciones completas y las parciales, probablemente a través de una reducción de dimensionalidad o el preprocesamiento del texto. El objetivo sería conseguir que las descripciones completas puedan ser procesadas de forma efectiva con los modelos de descripciones simples.

6.3. Comentario final

En conclusión, se puede ver que las tecnologías de aprendizaje automático y procesamiento del lenguaje natural están avanzando en gran medida en los últimos años. Cada vez tendrán más exactitud y pronto permitirán establecer correlaciones entre datos que a día de hoy pueden pasar desapercibidas para las personas. Este proyecto está enfocado con esta mentalidad y aunque todavía le faltan horas de procesamiento para hacerlo una herramienta exacta, es un primer paso para una clasificación fiable en un problema que inicialmente parecía imposible de automatizar. Por lo tanto este proyecto supone un avance muy relevante para poder obtener datos sobre la importancia de la intervención de los testigos. Gracias a ello, si se sigue investigando en este tema, se puede conseguir reducir en gran medida

el impacto psicológico en las víctimas, reducir las situaciones de acoso y hacer de la sociedad un lugar un poco más seguro.

A. Anexo A: Objetivos de desarrollo sostenible

A.1. Introducción a los ODS

Los objetivos de desarrollo sostenible son 17 metas fijadas por los miembros de la ONU en 2015.[3] En ellas se reconocen y fijan los 169 objetivos, divididos en 17 categorías, a cumplir antes del año 2030. Se centran en 5 áreas: personas (reducción de la pobreza, las desigualdades...), planeta(asegurar el consumo responsable, evitar el cambio climático...), prosperidad(asegurar que todos los seres humanos puedan tener vidas completas económica, tecnológica y socialmente), paz (sociedades inclusivas y libres de violencia) y asociación (solidaridad global y alineación de las naciones para cumplir estos objetivos).

Figura A.1: Objetivos de desarrollo sostenible[3]

Este proyecto tiene un carácter claramente social. Se trata de una colaboración con una ONG donde el objetivo a corto plazo es minimizar el impacto psicológico que causan las situaciones de acoso en las víctimas y el objetivo a largo plazo es eliminar todas las formas de acoso. El proyecto está alineado con varios de los objetivos de desarrollo sostenible de la ONU.

A.2. ODSs primarios

5. Lograr la igualdad de género y empoderar a todas las niñas y mujeres.

Este ODS se encuentra enmarcado en la dimensión social de los objetivos de desarrollo sostenible. Tiene como finalidad acabar con todas las formas de discriminación y violencia contra las mujeres, asegurar la igualdad de oportunidades en puestos de liderazgo, políticos y asegurar el derecho a sanidad sexual.

Como se ha dicho previamente el 80 % de las mujeres han sufrido alguna experiencia de acoso sexual en su vida, lo que está considerado como una forma de violencia con grandes impactos psicológicos. Este proyecto trata de reducir el número de situaciones de acoso y su impacto, por lo tanto está alineado con el SGD 5 teniendo el objetivo final de eliminar la violencia en este ámbito. Además, las situaciones de acoso están en un 88 %[49] producidas por hombres, normalmente debido a un sentimiento de superioridad. Reduciendo estos hechos, se reduciría igualmente el sentimiento de superioridad y por tanto se alcanzaría una mayor igualdad de género

10. Reducir las desigualdades dentro y entre los países

Este objetivo, alineado también con la dimensión social de los ODSs y tiene como finalidad dar más voz en la política mundial a los países en vías de desarrollo, implementar políticas fiscales y económicas que lleven a la igualdad y promover la seguridad e inclusión de las personas sin importar su etnia, raza, sexo, edad u orientación sexual.

Además de los acosos sexuales a las mujeres, la otra gran parte de los acosos son aquellos producidos por razones racistas u homófobas. Este proyecto trata de reducir todos los tipos de acoso, y por tanto promover la inclusión y seguridad de las personas en la calle, sin importar su etnia, orientación sexual, raza...

11. Convertir las ciudades y demás asentamientos humanos en lugares seguros, resilientes y sostenibles

El ODS 11 tiene como objetivos asegurar el acceso a una vivienda segura, a sistemas de transporte; asegurar el desarrollo de edificios y comunidades sostenibles

y ofrecer áreas públicas y seguras especialmente para niños, mujeres y personas mayores.

Este proyecto está alineado con el ODS 11 ya que reduciendo el acoso callejero se fomenta la seguridad de las mujeres en lugares públicos y también la libertad de usarlos sin miedo de ser agredida.

A.3. ODSs secundarios

4. Asegurar una educación inclusiva y equitativa y promover oportunidades vitalicias para todos

El ODS 4 tiene diversos objetivos entre los que destacan: ofrecer una educación de calidad, gratis y completa a todos los niños y niñas; eliminar las diferencias de género en la educación y educar a los niños y niñas en estilos de vida sostenibles, igualdad de género, cultura...

Este proyecto está alineado de forma secundaria con los ODS. Con él se trata de educar a los hombres y mujeres en la necesidad de intervenir cuando son testigos de una situación de acoso y de como hacerlo para minimizar el impacto psicológico de la víctima. Al mismo tiempo que se les educa en esto, se les está transmitiendo valores en igualdad de género, de raza, el respeto...

16. Promover sociedades pacíficas e inclusivas para el desarrollo sostenible, proporcionar acceso a la justicia para todos y construir instituciones efectivas, inclusivas y responsables

EL ODS 16 tiene como objetivos promover el acceso igualitario a la justicia para todos, reducir la corrupción, reducir el tráfico de armas y acabar con la explotación y el abuso de personas. Además hace hincapié en eliminar todas las formas de violencia.

Este proyecto está alineado con el ODS 16 ya que como se ha dicho previamente, pretende reducir el acoso y la violencia en

la calle y por tanto forma parte del objetivo de eliminar todas las formas de violencia.

Bibliografía

[1] UCSD center on gender equality y health. "A national study on sexual harassment and assault". En: (2019).

[2] Stop Street Harassment. http://www.stopstreetharassment.org/resources/statistics.

[3] ONU. *Sustainable development goals*. https://sustainabledevelopment.un.org/?menu=1300.

[4] Hollaback! https://www.ihollaback.org.

[5] CUP y Hollaback! "Show Up. Your guide to bystander intervention". En: ().

[6] SAS. *Natural Language Processing. What it is and why it matters*. https://www.sas.com/en_us/insights/analytics/what-is-natural-language-processing-nlp.html.

[7] Mario Alberich. "Procesamiento del Lenguaje Natural. Guía Introductoria". En: (jun. de 2007).

[8] El profesional de la información. *Procesamiento del lenguaje natural: revisión del estado actual, bases teóricas y aplicaciones*. http://www.elprofesionaldelainformacion.com/contenidos/1997/enero/procesamiento_del_lenguaje_natural_revisin_del_estado_actual_bases_tericas_y_aplicaciones_parte_i.html.

[9] Chatbot Pack. *How Eliza Chatbot works?* https://www.chatbotpack.com/how-eliza-chatbot-works/.

[10] IT Chronicles. *3 Industries That Will Be Disrupted by Natural Language Processing*. https://itchronicles.com/artificial-intelligence/3-industries-that-will-be-disrupted-by-natural-language-processing/.

[11] Xavier Amatriain. *NLP Healthcare: Understanding the Language of Medicine*. https://medium.com/curai-tech/nlp-healthcare-understanding-the-language-of-medicine-e9917bbf49e7.

[12] Monkey Learn. *Text Classification*. https://monkeylearn.com/text-classification/.

[13] Adam Geitgey. *Text Classification is Your New Secret Weapon*. https://medium.com/@ageitgey/text-classification-is-your-new-secret-weapon-7ca4fad15788.

[14] Muhammad Ishaq. *What are some of the challenges we face in NLP today?* https://medium.com/datadriveninvestor/what-are-some-of-the-challenges-we-face-in-nlp-today-2e9d94da1f63.

[15] Edward Loper Steven Bird Ewan Klein. *Natural Language Processing in Python*. 2007.

[16] Scikit learn. *About us*. https://scikit-learn.org/stable/about.html.

[17] Gensim. *About us*. https://radimrehurek.com/gensim/.

[18] Cristian Blanco. *CS50's Introduction to Artificial Intelligence with Python*. http://crismablanco.com.

[19] Michel Kana. *Representing text in natural language processing*. https://towardsdatascience.com/representing-text-in-natural-language-processing-1eead30e57d8.

[20] James H. Martin Daniel Jurafsky. *An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. 2019.

[21] Argaldo blog. *Google y sus N-gramas*. https://argaldo.wordpress.com/2006/08/10/google-y-los-n-gramas/.

[22] Jason Brownlee. *What Are Word Embeddings for Text?* https://machinelearningmastery.com/what-are-word-embeddings/.

[23] Hunter Heidenreich. *Introduction to Word Embeddings*. http://hunterheidenreich.com/blog/intro-to-word-embeddings/.

[24] Stokastik. *Understanding Word Vectors and Word2Vec*. http://www.stokastik.in/understanding-word-vectors-and-word2vec/.

[25] Jeffrey Pennington, Richard Socher y Christopher D. Manning. "GloVe: Global Vectors for Word Representation". En: *Empirical Methods in Natural Language Processing (EMNLP)*. 2014, págs. 1532-1543. URL: http://www.aclweb.org/anthology/D14-1162.

[26] SAP Conversational AI. *GloVe and fastText — Two Popular Word Vector Models in NLP*. https://cai.tools.sap/blog/glove-and-fasttext-two-popular-word-vector-models-in-nlp/.

[27] Stanford. *Feature selection*. https://nlp.stanford.edu/IR-book/html/htmledition/feature-selection-1.html.

[28] Saurav Kaushik. *Introduction to Feature Selection methods*. https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/.

[29] Vikashraj Luhaniwal. *Feature selection using Wrapper methods in Python*. https://towardsdatascience.com/feature-selection-using-wrapper-methods-in-python-f0d352b346f.

[30] Avinash Navlani. *Latent Semantic Analysis using Python*. https://www.datacamp.com/community/tutorials/discovering-hidden-topics-python.

[31] Cory Maklin. *Linear Discriminant Analysis In Python*. https://towardsdatascience.com/linear-discriminant-analysis-in-python-76b8b17817c2.

[32] Thushan Ganegedara. *Intuitive Guide to Latent Dirichlet Allocation*. https://towardsdatascience.com/light-on-math-machine-learning-intuitive-guide-to-latent-dirichlet-allocation-437c81220158.

[33] Junseok Lee y col. "Ensemble Modeling for Sustainable Technology Transfer". En: *Sustainability* 10 (jul. de 2018), pág. 2278. DOI: 10.3390/su10072278.

[34] Avinash Navlani. *Decision Tree Classification in Python*. https://www.datacamp.com/community/tutorials/decision-tree-classification-python.

[35] Onel Harrison. *Machine Learning Basics with the K-Nearest Neighbors Algorithm*. https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761.

[36] Shubham Panchal. *K Nearest Neighbours Classifier (KNN)-Machine Learning Algorithms*. https://medium.com/@equipintelligence/k-nearest-neighbor-classifier-knn-machine-learning-algorithms-ed62feb86582.

[37] Joaquín Amat Rodrigo. *Máquinas de Vector Soporte*. https://www.cienciadedatos.net/documentos/34_maquinas_de_vector_soporte_support_vector_machinesm\unhbox\voidb@x\bgroup\let\unhbox\voidb@x\setbox\@tempboxa\hbox{a\global\mathchardef\accent@spacefactor\spacefactor}\accent19a\egroup\spacefactor\accent@spacefactor\futurelet\@let@token\protect\penalty\@M\hskip\z@skipquinas_de_vector_soporte.

[38] Victor Roman. *Aprendizaje No Supervisado en Machine Learning: Agrupación*. https://medium.com/datos-y-ciencia/aprendizaje-no-supervisado-en-machine-learning-agrupaci\unhbox\voidb@x\bgroup\let\unhbox\voidb@x\setbox\@tempboxa\hbox{o\global\mathchardef\accent@spacefactor\spacefactor}\accent19o\egroup\spacefactor\accent@spacefactor\futurelet\@let@token\protect\penalty\@M\hskip\z@skipn-bb8f25813edc.

[39] *Qué son las redes neuronales y sus funciones*. https://www.atriainnovation.com/que-son-las-redes-neuronales-y-sus-funciones/.

[40] Ricardo Mendoza. *APLICACIÓN DEL GRADIENTE EN REDES NEURONALES*. https://medium.com/@ricardojmv85/aplicaci\unhbox\voidb@x\bgroup\let\unhbox\voidb@x\setbox\@tempboxa\hbox{o\global\mathchardef\accent@spacefactor\spacefactor}\accent19o\egroup\spacefactor\accent@spacefactor\futurelet\@let@token\protect\penalty\@M\hskip\z@skipn-del-gradiente-en-redes-neuronales-78bff0d802d5.

[41] Juan Ignacio Barrios Arce. *La matriz de confusión y sus métricas*. https://www.juanbarrios.com/matriz-de-confusion-y-sus-metricas/.

[42] *Carto maps*. https://carto.com.

[43] *CSV File Reading and Writing*. https://docs.python.org/3/library/csv.html.

[44] *Beautiful Soup Documentation*. https://www.crummy.com/software/BeautifulSoup/bs4/doc/.

[45] *Contractions 0.0.25*. https://pypi.org/project/contractions/.

[46] *NLTK 3.5 documentation*. https://www.nltk.org.

[47] *Wordcloud 1.7.0*. https://pypi.org/project/wordcloud/.

[48] *Scikit learn, API Reference*. https://scikit-learn.org/stable/modules/classes.html.

[49] *Estudio sobre acoso sexual, acoso sexista, acoso por orientación sexual y acoso por identidad y expresión de género en la Universidad Complutense de Madrid*. Inf. téc. Universidad Complutense de Madrid, 2018.