CLASIFICACIÓN DE TWEETS MEDIANTE MODELOS DE APRENDIZAJE SUPERVISADO

Verónica Lucía Chamorro Alvarado

MÁSTER EN INGENIERÍA INFORMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

Trabajo Fin Máster en Ingeniería Informática

Convocatoria Septiembre de 2018
Calificación: 7 Notable

Directora:

Dra. Yolanda García Ruiz

Autorización de difusión

Autor
Verónica Lucía Chamorro Alvarado

Fecha
Septiembre 2018

El/la abajo firmante, matriculado/a en el Máster en Ingeniería Informática de la Facultad de Informática, autoriza a la Universidad Complutense de Madrid (UCM) a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a su autor el presente Trabajo Fin de Máster: "CLASIFICACIÓN DE TWEETS MEDIANTE MODELOS DE APRENDIZAJE SUPERVISADO", realizado durante el curso académico 2017-2018 bajo la dirección de Yolanda García Ruiz, y a la Biblioteca de la UCM a depositarlo en el Archivo Institucional E-Prints Complutense con el objeto de incrementar la difusión, uso e impacto del trabajo en Internet y garantizar su preservación y acceso a largo plazo.

Autor: Verónica Lucia Chamorro Alvarado

Resumen

En los últimos años las redes sociales han generado una gran cantidad de datos que se emplean como una fuente de información. En muchos casos el análisis de estas fuentes de información nos permite conocer casi al instante situaciones atípicas como por ejemplo accidentes de tráfico, congestión excesiva, el estado de las carreteras, etc. En este trabajo usamos Twitter como fuente datos y nos proponemos desarrollar un método para analizar el texto de un conjunto de tweets. Este método permite clasificar dichos tweets en 2 clases: "Tráfico", "Contaminación".

Empleamos varios algoritmos de clasificación supervisada, que fueron previamente entrenados. Se estudiaron los siguientes cuatro algoritmos, Bayes Naive multiclase, arboles de decisión, k-vecinos más cercanos y máquina de vectores de soporte, para obtener la exactitud de cada uno, y analizar cuál es el mejor algoritmo de clasificación para este caso de estudio. En los resultados obtenidos, con el algoritmo máquina de vectores de soporte, se logra un valor de exactitud de 85.22 % para la clasificación de eventos de tráfico y no tráfico. Además, fuimos capaces de realizar la clasificación multiclase, donde se obtuvo un valor de exactitud de 78.84 %.

Palabras clave

Detección de eventos de tráfico, detección de eventos de contaminación, SVM, algoritmos de clasificación

Abstract

In recent years social networks have generated a large amount of data, which is used as a source of information for several applications. In many cases analyzing these sources of information allow us to know almost instantly atypical situations, such as traffic accidents, traffic jumps, state of the roads, etc. In this work we use Twitter as source of information, and we proppose to develop a method to analize the text of a set of tweets. This method allows classifying those tweets into two classes: "Traffic" , and "Pollution".

We used several supervised classification algorithms, that were previously trained. The following four algorithms, Bayes Naive multiclase, decision trees, k-neighbors nearest and support vector machines, which were studied to obtain their accuracy, and to analyze the best classification algorithm for this case study. The results obtained, with the support vector machine algorithm, show an accuracy value of 85.22 % for the classification of traffic events and non-traffic. In addition, we were able to perform the multiclass classification, where the accuracy value is 78.84 %.

Keywords

Traffic event detection, pollution event detection, SVM, algorithms of classification.

Índice general

Lista de Figuras III

Lista de Tablas IV

Dedicatoria V

Agradecimientos VI

1. Introducción 1
- 1.1. Metodología . . . 2
- 1.2. Objetivos . . . 4
- 1.3. Estructura del documento . . . 4

2. Introduction 6
- 2.1. Methodology . . . 7
- 2.2. Objectives . . . 9
- 2.3. Document structure . . . 9

3. Estado del Arte 10
- 3.1. Aprendizaje automático . . . 13
- 3.2. Aprendizaje supervisado . . . 14
- 3.3. Aprendizaje no supervisado . . . 15
- 3.4. Extracción de Características . . . 16
  - 3.4.1. Modelo de espacio de vectores . . . 17
  - 3.4.2. Bolsa de palabras (Bag of Words) . . . 18
  - 3.4.3. TF-IDF (Frecuencia de término – frecuencia inversa de documento) . . . 18
- 3.5. Algoritmos de aprendizaje automático . . . 19
  - 3.5.1. Clasificador Bayes Naive (NB) . . . 19
  - 3.5.2. Árboles de decisión . . . 22
  - 3.5.3. kNN . . . 24
  - 3.5.4. SVM (Máquinas de vectores de soporte) . . . 25
- 3.6. Aplicaciones del Aprendizaje Automático . . . 27
- 3.7. Proceso del aprendizaje automático . . . 28
  - 3.7.1. Recolectar los datos . . . 28
  - 3.7.2. Pre-procesar los datos . . . 28
  - 3.7.3. Explorar los datos . . . 29
  - 3.7.4. Entrenar el algoritmo . . . 29
  - 3.7.5. Evaluar el algoritmo . . . 29

3.7.6. Utilizar el modelo . . . 29
3.8. Clasificación de textos . . . 30

4. Descripción del caso de estudio **32**
4.1. Arquitectura del Sistema de detección tráfico y contaminación. . . . 32
4.1.1. Obtención de tweets y preprocesamiento . . . 34
4.1.2. Procesamiento de tweets . . . 35
4.1.3. Clasificación de tweets . . . 39
4.1.4. Herramientas utilizadas . . . 40
4.2. Configuración del sistema . . . 42

5. Evaluación del sistema **45**
5.1. Descripción de los conjuntos de datos . . . 45
5.1.1. Conjunto de datos T (clase tráfico) y N (clase no tráfico) . . . 46
5.1.2. Conjunto de datos C (clase contaminación) y N (clase no contaminación) 48
5.1.3. Resultados experimentales . . . 50

6. Conclusiones y Trabajo Futuro **55**
6.1. Conclusiones . . . 55
6.2. Trabajo Futuro . . . 55

7. Conclusions and Future Work **57**
7.1. Conclusions . . . 57
7.2. Future Work . . . 57

Bibliografía **59**

Índice de figuras

3.1. Árbol de decisión 24
3.2. SVM de dos clases representando en el hiperplano y vectores de soporte . . . 26
3.3. Proceso de aprendizaje automático 28
3.4. Representación conceptual de alto nivel del proceso de clasificación de texto 31

4.1. Arquitectura del sistema para la detección de tráfico a partir del análisis de la información de Twitter 33
4.2. Procesamiento de tweets 35
4.3. Proceso de minería de texto aplicado a un tweet de la muestra 40
4.4. Proceso de entrenamiento del sistema de clasificación 42

5.1. Evaluación cruzada [56] 51

Índice de tablas

4.1. Tweets obtenidos 34

5.1. Tweets clasificados manualmente en las clases T (tráfico) y N (no tráfico) . . 47
5.2. Características significativas relacionadas con la clase T (tráfico) 48
5.3. Tweets clasificados manualmente en las clases C (contaminación) , CT (contaminación y tráfico) y N(ni contaminación ni tráfico) 49
5.4. Ecuaciones de Métricas estadísticas 52
5.5. Estadísticas clasificación de tweets en las clases T (tráfico) y N (no tráfico), los mejores valores en negrilla 53
5.6. Estadísticas clasificación de tweets en las clases C (contaminación) y CT (contaminación y tráfico), los mejores valores en negrilla 54

A mi madre Bertha Alvarado,
por apoyarme día a día a pesar de la distancia.
Verónica Chamorro

Agradecimientos

A mi tutora Dra. Yolanda García por su apoyo constante durante la elaboración de este proyecto.

A Richard por ayudarme en este proyecto y no dejar que me de por vencida, sin su ayuda y motivación no lo hubiese conseguido.

A mi madre por estar siempre pendiente de mí, por su cariño y apoyo en todo momento.

A mis hermanos en especial a Robert y Andrés que han sido mi apoyo desde muy pequeña, Fernando y Angelita que han estado pendientes de la realización de este trabajo.

A mis sobrinas, sobrinos y sobrinas nietas que me motivan a seguir adelante día a día. En especial a Mabel, Lizbeth, Pamela, Robert Alejandro, Jonathan, Karla, Katherine, Victoria, Zharick y Nicole los quiero muchísimo.

Igualmente quiero agradecer a mi cuñada Nuria, a mis tías Hermila, Mercedes, Gloria , Luz, Marinita, a mis primas y primos Silvana, Imelda, Albita, Fatima, Edgar, William y a toda mi familia.

A mis amigos de Ecuador en especial a Jorge, Juan, Daniela, Glenda, Rommel, Patricio, Geovanny, Dianita, Cleopatra, Mariana y Henry . A mis amigos de España en especial Yedmi, Jakeline, Roberto, Ricardo, Elizabeth, Estefanía, Francisco, José Luis y Patricio.

A todas las personas que de una u otra forma me han apoyado en este proyecto.

Capítulo 1

Introducción

En los últimos años la red social Twitter se ha vuelto cada vez más popular, hasta llegar a ser el sitio web de microblogs más utilizado con más de 330 millones de usuarios generando alrededor de 500 millones de mensajes al día, convirtiéndose así en una gran fuente de información. Los mensajes o *tweets* que se publican en Twitter pueden compartir información sobre acontecimientos actuales públicos, noticias de interés, noticias para grupos específicos, entre otros.

La particularidad de Twitter frente a otras redes sociales es el limitado tamaño de los mensajes que pueden enviar los usuarios, siendo este de únicamente 140 caracteres, con la finalidad de que los usuarios puedan expresar sus ideas con pocas palabras.

El contenido de los tweets en muchos casos puede expresar sentimientos sobre situaciones específicas, opiniones sobre productos o servicios. Estas opiniones se pueden analizar para responder de forma adecuada a los sentimientos que expresen los tweets; por ejemplo, para promocionar servicios o productos, o si se menciona algún acontecimiento en un lugar específico, como un desfile que está causando tráfico. Los consumidores de Twitter pueden estar informados de estas situaciones en el mismo instante en que están sucediendo para por ejemplo planificar sus rutas.

En este trabajo se propone un modelo para clasificar automáticamente tweets de tráfico y contaminación. Utilizando la información de los tweets, evaluamos cuatro algoritmos de aprendizaje automático supervisados para clasificación de texto y determinamos que el mejor

algoritmo para nuestro caso de estudio es el de máquina de vectores de soporte, pues es el que presenta un mayor valor de exactitud 85.22 % para la clasificación de eventos de tráfico y no tráfico. Además, fuimos capaces de realizar la clasificación multiclase, donde se obtuvo un valor de exactitud de 78.84 %. El código fuente ha compartido con la comunidad en GitHub[1].

1.1. Metodología

En esta sección describiremos la organización y estructura del trabajo a realizar durante el desarrollo del proyecto. La primera etapa del proyecto consistió en familiarizarnos con la fuente de datos, en este caso Twitter. Durante esta fase nos familiarizamos con la estructura de los tweets, su API y las restricciones de uso. Posteriormente se realizó una investigación en la literatura, en particular artículos de investigación existentes y se identificó los algoritmos más utilizados en la literatura, para la clasificación de textos. Después, comenzamos el desarrollo de la parte técnica del proyecto, relacionándonos con el entorno de trabajo, en el cual pre-procesamos la información e implementamos los algoritmos de clasificación. A continuación, se detallan los pasos seguidos.

- Instalación y configuración de la base de datos NoSQL MongoDB. Pruebas de su funcionamiento en local y creación de la base de datos.
- Instalación y configuración de Robomongo, para facilidad de uso al momento de realizar pruebas en la base de datos.
- Instalación y configuración de Jupyter notebook, donde realizaremos las tareas de programación requeridas.
- Instalación de Python, el lenguaje principal utilizado en los scripts del proyecto.
- Instalación de Pandas, librería que facilita y agiliza el análisis de datos con el lenguaje de programación Python.

[1] https://github.com/kinorev/TFM_UCM

- Instalación de NLTK, una plataforma para construir programas con el lenguaje Python para trabajar con datos de lenguaje humano [34]. Esta proporciona interfaces fáciles de usar con un conjunto librerías de procesamiento de texto para clasificación, división del texto en palabras, derivación, etiquetado, análisis y razonamiento semántico.

- Obtención de tweets proporcionados por la facultad de Geología de Universidad Complutense de Madrid.

El siguiente paso en el desarrollo del proyecto fue el análisis del contenido de los tweets y la aplicación de técnicas de minería de texto, para esto se realizó los siguientes pasos:

- Tomamos un conjunto de tweets y utilizando las librerías previamente instaladas, realizamos el procesamiento de los tweets. Esto incluye la tokenización, la eliminación de palabras vacías (palabras sin significado como artículos, pronombres, preposiciones, etc.), transformación de cada palabra en su raíz, filtrado de la raíz y la representación de características.

- Clasificamos manualmente las muestras de tweets utilizadas en el proyecto, con el propósito de tener un conjunto de entrenamiento clasificado correctamente.

- Mediante arboles de decisión se tomó las palabras relevantes, para utilizarlas en los diferentes algoritmos.

- El conjunto de datos entrenados se introdujo en el sistema y se realizó pruebas con nuevos conjuntos de datos, con el fin de clasificar nuevas muestras.

- Se calculó las métricas de precisión, exactitud y F-score, de los resultados obtenidos al aplicar los diferentes algoritmos, con el fin de determinar cuál es el mejor.

- Pruebas de funcionamiento del modelo de clasificación.

- Análisis y mejora de los resultados.

La tarea de redacción de la memoria se llevó a cabo una vez finalizado el desarrollo.

1.2. Objetivos

El principal objetivo de este trabajo es proponer un sistema para la detección de eventos relacionados con el tráfico y la contaminación a partir del análisis de datos de Twitter. Para esto, nuestro objetivo se divide en los siguientes objetivos secundarios.

- Analizar las técnicas actuales utilizadas para la detección de eventos en Twitter, para conocer y profundizar en los algoritmos de aprendizaje automático que nos pueden ayudar a diseñar e implementar el sistema de detección de eventos.
- Diseñar el sistema de detección de eventos utilizando varios paquetes de software, que faciliten la implementación de los algoritmos de clasificación de texto.
- Evaluar nuestro sistema propuesto con varios algoritmos y configuraciones, para seleccionar el algoritmo que presente mejores resultados de exactitud en la detección de eventos.

1.3. Estructura del documento

Este documento se estructura de la siguiente manera. En el Capítulo 3 se analiza los modelos de aprendizaje automático. Se describen los tipos de aprendizaje Automático como son el supervisado y el no supervisado. Se describen los pasos necesarios para realizar el aprendizaje automático. A continuación, se detallan algunos de los algoritmos del aprendizaje automático existentes y que se han utilizado en este trabajo como son el algoritmo Bayes Naive Multinomial, los arboles de decisión, kNN y Support Vector Machine (SVM). Se incluye una breve descripción de lo referente a la clasificación de textos.

En el Capítulo 4 se describe el caso de estudio analizado en este trabajo, y la arquitectura utilizada para el sistema de detección tráfico y contaminación. Luego, se detalla el proceso para la obtención de tweets y el pre-procesamiento de estos. Posteriormente, se sigue los pasos para la clasificación de textos, se realiza el procesamiento de tweets y se los clasifica.

También se especifican las herramientas utilizadas para lograr la clasificación de los tweets, y la configuración del sistema.

En el Capítulo 5 se describen los conjuntos de datos utilizados, y se analizan los resultados experimentales obtenidos. Finalmente en el Capítulo 6 se resume el trabajo presentando las conclusiones y posibles trabajos futuros.

Capítulo 2

Introduction

In recent years the social media Twitter has become increasingly popular, to become the most used microblog website with more than 330 million users generating about 500 million messages a day, thus becoming an excellent source of information.

The messages or tweets published on Twitter may share information about current public events, news of interest, news for specific groups, among others. The particularity of Twitter compared to other social networks is the limited size of the messages that users can send, this being only 140 characters, with the purpose that users can express their ideas with few words.

The content of the tweets in many cases can express feelings about specific situations, opinions about products or services. These opinions can be analyzed to respond appropriately to the feelings expressed by the tweets; for instance, to promote services or products, or whether an event is mentioned in a specific place, such as a parade that is causing traffic. Twitter consumers can be informed of these situations at the same moment they are happening, which may be helpful to plan their routes.

In this work, we propose a model to classify traffic tweets and pollution automatically. Using the information in the tweets, we evaluate four machine learning supervised algorithms for text classification, and we determine that the best algorithm for our case study is support vector machine, since it has the highest accuracy value 85.22 % for the classification of traffic events and not traffic. Besides, we were able to perform the multiclass classification, where

we obtain an accuracy of 78.84 %.

2.1. Methodology

In this section, we describe the organization and structure of the work performed during the development of this project. The first stage of the project was to become familiar with the source of data, in this case, Twitter. During this phase, we become familiar with the structure of the tweets, their API, and usage restrictions. Subsequently, an investigation was carried out in the literature, in particular, we identified the existing research articles and the algorithms most used for classification of texts. Then, we started to develop the technical part of the project, testing the work environment, in which we pre-process the information and implement the classification algorithms. Next, we detail the followed steps.

- Installation and configuration of the NoSQL MongoDB database. Test its functioning in local and creation of the database.
- Installation and configuration of Robomongo, for ease of use at the time of testing the database.
- Installation and configuration of Jupyter notebook, where we will perform the required programming tasks.
- Python installation, the language used in project scripts.
- Installation of Pandas, a library that facilitates and improves the performance analysis of data in programs coded with the Python programming language.
- Installation of NLTK, a platform to build programs with the Python language to work with human language data [34]. It provides easy-to-use interfaces with a set of text processing libraries for classification, tokenization, steaming, tagging, labeling, analysis and semantic reasoning.

- Obtaining tweets provided by the Facultad de Geología from Universidad Complutense de Madrid.

The next step in the development of the project was the analysis of the content of the tweets, and the application of text mining techniques, for this we perform the following steps:

- We take a set of tweets and using the previously installed libraries. Then we perform the processing of tweets. It includes tokenization, stop-words filtering (removal of words without meaning such as articles, pronouns, prepositions, etc.), steaming, steam filtering and feature representation.
- We manually classify the samples of tweets used in the project, with the purpose of having a correctly classified training set.
- Using decision trees the relevant words were taken, to be used with the machine learning algorithms.
- We submit the trained dataset into the system and perform several tests with new datasets in order to classify new samples.
- We calculated the precision, the accuracy and F-score metrics of the results, applying the different algorithms, in order to determine which provide better results.
- Functioning tests of the classification model.
- Analysis and improvement of results.

The writing of this work was performed once the development phase was completed.

2.2. Objectives

The main objective of this work is to propose a system for the detection of related events with traffic and pollution from the analysis of Twitter data. Our main objective is divided into the following secondary objectives.

- Analyze the current techniques used for the detection of events on Twitter, to know and deepen the machine learning algorithms that can help us design and implement the event detection system.
- Design the event detection system, using several software packages, which facilitate the implementation of text classification algorithms.
- Evaluate our proposed system with several algorithms and configurations, to select the algorithm that presents the best results of accuracy in the detection of events.

2.3. Document structure

This document is structured as follows. Chapter 3 analyzes the machine learning models. The types of machine learning are described, such as supervised and unsupervised. The necessary steps to perform the machine learning are described. Next, it describes some machine learning algorithms that have been used in this work, such as multinomial Bayes Naive algorithm, decision trees, kNN and support vector machine (SVM). It Also presents, a brief description of the classification of texts is included.

Chapter 4 describes the case study analyzed in this work, and the architecture used for the traffic and pollution detection system. It details the process for obtaining tweets and preprocessing them. Then it describes the steps followed for the classification of texts, the processing of tweets and the classification of those. It also specifies the tools used to achieve the classification of tweets and the configuration of the system.

Chapter 5 describes the used datasets, and analyzes the experimental obtained results. Finally Chapter 7 summarizes this work by concluding and presenting possible future works.

Capítulo 3

Estado del Arte

Existen numerosos trabajos previos que analizan la clasificación o agrupación de la información publicada en Twitter. Usualmente el objetivo de estos trabajos es detectar y clasificar eventos en tiempo real. Para realizar la clasificación se recurre a diferentes enfoques, que se cubren ampliamente en la literatura estudiada.

Estos enfoques en alto nivel incluyen el aprendizaje automático y la minería de datos [32, 21], el procesamiento del lenguaje natural [27, 24], la extracción de información [22], la recuperación de información [6] y la minería de texto [22, 2]. Existen otros estudios más específicos que se relacionan con los objetivos de este trabajo los cuales se describen brevemente a continuación.

Tieman et al., [44] usan el muestreo de Gibbs para describir un modelo generativo para colecciones de documentos, el modelo de autor-tema, que modela simultáneamente el contenido de los documentos y los intereses de los autores.

Risch [42] evalúa LDA (Asignación latente de Dirichlet) para determinar si este método es adecuado para identificar temas en un medio con mensajes muy cortos como Twitter. Esto se realiza mediante la evaluación de la capacidad de LDA para modelar los mensajes entrantes y su capacidad para clasificar los mensajes nuevos que contienen temas conocidos. La evaluación muestra que el modelo puede ser adecuado para ciertas aplicaciones en la detección de temas cuando el tamaño del mensaje es lo suficientemente pequeño.

Se creía que el vocabulario y la forma de escribir de los tweets bajaría el rendimiento del

algoritmo LDA, pero no fue así, por otro lado, dado que Twitter contiene numerosos temas, el rendimiento de una versión de LDA que intenta modelar cada tema podría ser demasiado baja.

Gao et al., [18] empleo LDA para extraer regiones funcionales urbanas de puntos de interés y actividades humanas en redes sociales basadas en la ubicación, extrayendo regiones funcionales urbanas mediante uno de los métodos más conocidos K-means, para reducir brechas en la información obtenida se utilizó la validación cruzada.

Weber et al., [28] emplearon LDA para clasificar la información obtenida en Twitter relacionada con la obesidad, ellos concluyen que a partir de la ubicación geográfica se puede ayudar a los organismos de salud a determinar si existe correlación entre, si el lugar o 'en dónde vivimos' afecta nuestras elecciones, nuestro comportamiento saludable o nuestro acceso a la atención médica.

Ramage et al., [38] implementan LDA incorporando la supervisión para la concesión de créditos con múltiples etiquetas. Además, usa Bayes Naive multinomial mediante la incorporación de un modelo de mezcla. Con estos modelos demuestra que este modelo supervisado supera en gran medida las máquinas de vectores de soporte.

Lansley y Longley [26] aplican LDA a una muestra suficientemente grande de tweets geoetiquetados del interior de Londres. Los autores utilizan robustas técnicas de limpieza de texto, que demuestran que LDA puede clasificar los temas de Twitter de Londres basándose en la geolocalización. Además, sugieren que se puede aplicar LDA a cualquier muestra de Tweets. Indican también que se debe tener en cuenta un número óptimo de clases, dado que es un proceso subjetivo y es posible que los datos demográficos deban obtenerse de ubicaciones alternativas.

Zou y Song [62] utilizan un enfoque de dos pasos para el análisis de datos de Twitter. El objetivo es generar un modelo de tema y el segundo para agrupar los tweets en categorías basadas en temas, para esto también emplean LDA. Usaron la precisión para evaluar los métodos. Para identificar la precisión, determinaron manualmente a qué tema pertenece un

tweet como resultado del modelo de tema. La tasa de precisión la definieron como el número de tweets identificados correctamente por el modelo para un tema específico dividido por el número total de tweets clasificados para este tema. Los autores utilizaron los mensajes de Twitter de "Twitter Top 1000 seguidores" usando la API de Twitter con tipo de resultado como "popular" y clústeres usando modelos de tema no supervisados para identificar el tema y las palabras temáticas, finalmente utilizaron palabras temáticas combinadas con la influencia de tweet para visualizar gráficamente los temas más candentes durante el tiempo de muestreo.

Zhou et al., [61] implementan LDA y varias técnicas de limpieza de texto para la extracción de eventos de ciudades del mundo real a partir de datos de Twitter. Aplicaron su enfoque para identificar eventos específicamente en la ciudad de Londres, a partir de una muestra de más de 30000 tweets en un día determinado, los eventos detectados en sus experimentos oscilaron entre 44 y 51, siendo los eventos culturales los más mencionados en Twitter. El enfoque propuesto por los autores es un modelo no supervisado, dado que indican es el más adecuado para detectar eventos del mundo real que pueden informar e influir en la toma de decisiones y la planificación de las autoridades municipales y ciudadanas, ya que los enfoques que se concentran solo en tipos de eventos específicos pueden no ser factibles para proporcionar un contexto urbano.

Steiger et al. [50] emplean LDA y diversas técnicas de limpieza de texto con el fin de investigar patrones latentes de movilidad humana intraurbana durante eventos masivos. Los autores en su caso de estudio utilizaron tweets georreferenciados del área de Greater Boston y buscaron temas relacionados con la serie mundial de béisbol, el subconjunto utilizado para el análisis fue de 251.771 tweets. Una limitación clara de la metodología actual para extraer información es la dependencia de los datos de Twitter; suponiendo que los tweets se escriben en el sitio, haciendo referencia a un evento en un lugar y hora en que se han publicado. Las señales temporales y semánticas derivadas de Twitter también pueden no ser lo suficientemente significativas como para servir como un indicador indirecto de la

caracterización del comportamiento de movilidad complejo.

La mayoría de los artículos recientes utilizan LDA como modelo de clasificación de tweets, el inconveniente es que los tweets son muy cortos, el lenguaje utilizado es demasiado informal, no siguen la gramática, y como resultado, el sustantivo, el adjetivo y los verbos no proporcionan información muy útil para los modelos de tema.

En este capítulo, se describen los principales enfoques para la detección de eventos de Twitter. Los enfoques que se estudian son los del aprendizaje automático y la minería de textos. Sobre el primero, se estudian el enfoque de *aprendizaje supervisado* y el aprendizaje no supervisado, y sobre el segundo se describen los modelos LDA y el TF-IDF (del inglés Term frequency – Inverse document frequency).

3.1. Aprendizaje automático

La enorme cantidad de datos que se generan continuamente ha llevado a que se designe al tiempo que estamos viviendo como la era de los datos [36]. Estos grandes conjuntos de datos son utilizados en numerosos dominios como la ingeniería, las redes sociales, comercio, multimedia, investigación molecular, de seguridad informática, también del internet de las cosas con el incremento de dispositivos conectados a las redes de información. Constantemente escuchamos que la cantidad de datos que se generan por año se incrementa rápidamente [17], la información digital ha crecido nueve veces en volumen en sólo 5 años y se estima que su cantidad en el mundo llegará a 35 billones de gigabytes para el 2020 [40] .

Para poder analizar esta inmensa cantidad de información se utiliza el aprendizaje automático. Los analistas de datos en el gobierno, la academia, la industria, el comercio usan las herramientas del aprendizaje automático para la toma de decisiones, planeamiento de acciones, tratando de ofrecer servicios más personalizados y así obtener una ventaja competitiva sobre sus rivales [47].

El aprendizaje automático estudia técnicas, que permiten encontrar patrones de los datos, mediante procesos automáticos o semiautomáticos [57], con el objetivo de generalizar,

o generar reglas desconocidas a partir de ejemplos donde esa regla es aplicada.

3.2. Aprendizaje supervisado

El aprendizaje supervisado entrena el algoritmo a partir de datos que han sido previamente etiquetados de manera manual. Etiquetados quiere decir que se le asigna una clase manual en base a la experiencia. Las clases se utilizan para representar entidades (i.e., representación de un objeto o concepto del mundo real) por ejemplo los sustantivos en el lenguaje. Cada clase es un modelo que define un conjunto de variables o datos [53].

Una entidad es la representación de un objeto o concepto del mundo real, por ejemplo, la entidad "Alumno" podría tener los atributos: nombre, apellido, año de nacimiento, etc. La entidad "Noticia" podría tener los atributos: titular, descripción, texto de la noticia, fecha, etc [4].

Mientras más grande es el conjunto de datos entrenados, mayor es la eficacia del algoritmo de aprendizaje automático.

Una vez finalizado el proceso de entrenamiento, utilizamos los algoritmos de aprendizaje automático con una muestra de tweets sin clasificar, con la finalidad de clasificar los nuevos tweets.

Esto es similar al método de aprendizaje que se utiliza en las escuelas, donde se nos enseñan problemas y las formas de resolverlos, para que luego podamos aplicar los mismos métodos en planteamientos similares.

Aunque etiquetar manualmente una gran cantidad de mensajes de Twitter es una tarea que requiere mucha mano de obra y consume mucho tiempo, es más factible para eventos específicos que para los no especificados. Cuando se conocen algunas descripciones de eventos, las técnicas de filtrado pueden usarse para reducir la cantidad de mensajes irrelevantes y facilitar que un experto humano anote un conjunto de datos de tamaño "razonable".

Además, el filtrado según las descripciones de eventos específicos, como palabras clave, ubicación o tiempo, también reduciría la cantidad de mensajes de Twitter que se deben

procesar durante el funcionamiento del sistema y permitirá que el algoritmo de detección se centre en un conjunto restringido de tweets. Varios algoritmos de clasificación supervisados han sido propuestos para eventos específicos, por ejemplo, NB [8] y SVM [9, 45].Estos clasificadores suelen ser entrenados con un pequeño conjunto de mensajes de Twitter recolectados durante algunas semanas o meses y luego filtrados y etiquetados de acuerdo con el evento, por ejemplo, en [45], clasificó los tweets cuando había un terremoto o sin terremoto.

El procedimiento de etiquetado generalmente implica dos expertos en el tema a etiquetar. Cuando los tweets son ambiguos dado que los expertos no logran ponerse de acuerdo en cómo etiquetarlos, estos se descartan. Además de filtrar los mensajes irrelevantes, cuando la tarea de detección involucra eventos específicos, se pueden incluir características adicionales (aparte de la palabra o la frecuencia de hashtag), en el algoritmo de detección para una precisión mejorada del sistema. Estas características pueden variar ampliamente dependiendo del evento objetivo y su descripción. Por ejemplo, además de la frecuencia de las palabras, [45] consideraron las palabras clave especiales que mencionan un "terremoto", su variante o palabras relacionadas (por ejemplo, "temblor"), así como la información contextual que rodea estas palabras clave.

3.3. Aprendizaje no supervisado

En los problemas de aprendizaje no supervisado el algoritmo utiliza un conjunto de datos que no tiene ninguna etiqueta; en este caso, nunca se le dice al algoritmo lo que representan los datos. La idea es que el algoritmo pueda encontrar por si solo patrones que ayuden a entender el conjunto de datos. El aprendizaje no supervisado es similar al método que utilizamos para aprender a hablar cuando somos bebes, en un principio escuchamos hablar a nuestros padres y no entendemos nada; pero a medida que vamos escuchando miles de conversaciones, nuestro cerebro comenzará a formar un modelo sobre cómo funciona el lenguaje y comenzaremos a reconocer patrones y a esperar ciertos sonidos. Existen varios artículos que utilizan algoritmos de aprendizaje no supervisado, uno de estos es [50]

LDA (Asignación latente de Dirichlet) LDA de sus siglas en ingles Latent Dirichlet Allocation, es un modelo generativo probabilístico para textos. LDA ha recibido bastante atención en los últimos años [62, 61, 39, 50]. El modelo permite que conjuntos de observaciones puedan ser explicados por grupos no observados que explican por qué algunas partes de los datos son similares. Por ejemplo, si las observaciones son palabras en documentos, presupone que cada documento es una mezcla de un pequeño número de categorías (también denominados tópicos) y la aparición de cada palabra en un documento se debe a una de las categorías a las que el documento pertenece. LDA es un ejemplo de modelo de categorías y fue presentado como un modelo en grafo para descubrir categorías [11, 54]. LDA es un algoritmo completamente no supervisado que modela cada documento como una mezcla de temas. El modelo genera resúmenes automáticos de temas en términos de una distribución de probabilidad discreta sobre palabras para cada tema, y además infiere distribuciones discretas por documento sobre temas. Lo que es más importante, LDA asume explícitamente que cada palabra se genera a partir de un tema subyacente. Aunque LDA es lo suficientemente expresivo para modelar múltiples temas por documento, no es apropiado para textos multi-etiquetados porque, como modelo no supervisado, no ofrece una forma obvia de incorporar un conjunto de etiquetas supervisadas en su procedimiento de aprendizaje [38].

3.4. Extracción de Características

Los datos que se usan para los estudios en el aprendizaje automático son representados por un número acordado de características que pueden ser binarias, categóricas o continuas [20]. Una característica es sinónimo de una variable de entrada o un atributo, únicas, atributos medibles o propiedades de un experimento o un punto de dato dentro de un conjunto. Encontrar una buena representación de los datos con estas características depende mucho del dominio de los datos, por ejemplo, si el dominio del estudio fuese la medicina las características podrían ser síntomas, como un conjunto de variables que representarían el estado de salud de un paciente (e.g., fiebre, nivel de glucosa, peso, presión arterial, etc.).

Estos algoritmos generalmente esperan características en forma de vectores numéricos, porque cada algoritmo es en esencia una operación matemática de optimización y minimiza la pérdida y el error, cuando intenta aprender patrones a partir de puntos de datos y observaciones. Entonces, con los datos textuales existe el desafío adicional de descubrir, cómo transformar los datos textuales y extraer las características numéricas de ellos.

Existen varias técnicas para la extracción de características que pueden ser aplicadas en la clasificación de textos, estas técnicas usualmente forman parte de los algoritmos del aprendizaje automático para el aprendizaje de patrones, que pueden ser aplicados sobre futuros nuevos puntos de datos para obtener información. Estos algoritmos a menudo reciben características en forma de vectores numéricos porque cada algoritmo es en esencia una operación matemática de optimización y minimiza la pérdida y el error, cuando intenta aprender patrones a partir de puntos de datos y observaciones. Entonces, con datos textuales existe el desafío de descubrir cómo transformar estos datos textuales y extraer las características numéricas de ellos.

A continuación, se presentan algunos conceptos relevantes que se involucran en la extracción de características, estos incluyen el modelo de espacio de vectores, la bolsa de palabras y el modelo TF-IDF.

3.4.1. Modelo de espacio de vectores

El modelo de espacio de vectores (en ingles Term Vector Model) es un concepto y modelo útil en caso de que tratemos con datos textuales y es muy popular en la recuperación de información y la clasificación de documentos. El Modelo de espacio de vectores, se define como un modelo matemático y algebraico para transformar y representar documentos de texto como vectores numéricos de términos específicos que forman las dimensiones del vector. Matemáticamente esto se puede definir de la siguiente manera. Digamos que tenemos un documento D en un espacio vectorial de documento VS. El número de dimensiones o columnas para cada documento será el número total de términos o palabras distintos para todos

los documentos en el espacio vectorial. Entonces, el espacio vectorial puede ser denotado por $VS = W_1, W_2, ..., W_n$, donde hay n palabras distintas en todos los documentos. Ahora podemos representar el documento D en este espacio vectorial como $D = W_{D1}, W_{D2}, ..., W_{Dn}$ donde W_{Dn} denota el peso para la palabra n en el documento D. Este peso es un valor numérico y puede representar cualquier cosa, desde la frecuencia de esa palabra en el documento, hasta la frecuencia promedio de aparición, o incluso hasta el peso TF-IDF.

3.4.2. Bolsa de palabras (Bag of Words)

El modelo de bolsa de palabras es una técnica simple para extraer características de documentos. En particular este modelo convierte documentos de texto en vectores, tal que cada documento es convertido en un vector que representa la frecuencia de todas las distintas palabras que están presentes en el espacio del vector del documento, para este documento en específico. Por lo tanto, teniendo en cuenta nuestro vector de muestra de la notación matemática anterior para D, el peso de cada palabra es igual a su frecuencia de aparición en ese documento.

3.4.3. TF-IDF (Frecuencia de término – frecuencia inversa de documento)

TF-IDF (del inglés Term frequency – Inverse document frequency), es la frecuencia de ocurrencia del término en la colección de documentos, es una medida numérica que expresa cuán relevante es una palabra para un documento en una colección. Esta medida se utiliza a menudo como un factor de ponderación en la recuperación de información y la minería de texto. El valor TF-IDF aumenta proporcionalmente al número de veces que una palabra aparece en el documento, pero es compensada por la frecuencia de la palabra en la colección de documentos, lo que permite manejar el hecho de que algunas palabras son generalmente más comunes que otras.

TF-IDF puede utilizarse exitosamente para el filtrado de las denominadas palabras vacías (palabras que suelen usarse en casi todos los documentos), en diferentes campos como la

clasificación y resumen de texto [55].

3.5. Algoritmos de aprendizaje automático

A continuación, se detallan brevemente los algoritmos utilizados para realizar la clasificación de tweets en este trabajo.

3.5.1. Clasificador Bayes Naive (NB)

El clasificador NB es un algoritmo de aprendizaje bayesiano muy práctico [29]. En algunos campos se ha demostrado su rendimiento, siendo comparable al de redes neuronales y al aprendizaje del árbol de decisiones. NB es un clasificador de textos supervisado.

El clasificador NB es un algoritmo de clasificación probabilística basado en la aplicación del teorema de Bayes, y se caracteriza por un modelo de probabilidad que asume independencia entre las entidades de entrada [48]. En otras palabras, el modelo supone que la presencia de una característica particular no está relacionada con la presencia de cualquier otra. Este algoritmo se utiliza específicamente para tareas de predicción y clasificación donde se tiene más de dos clases.

Matemáticamente podemos formular esto como se muestra a continuación [48]: Dada una respuesta de clase variable y y un conjunto de n características en forma de vector $\{x_1, x_2, \ldots, x_n\}$, usando el teorema de Bayes podemos denotar la probabilidad de la ocurrencia de y dadas las características como se muestra en la Ecuación 3.1

$$P(y|x_1, x_2, \ldots, x_n) = \frac{P(y) \times P(x_1, x_2, \ldots, x_n|y)}{P(x_1, x_1, \ldots, x_1)} \tag{3.1}$$

Bajo el supuesto que se indica en la Ecuación 3.2.

$$P(x_i|y, x_1, x_2, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = P(x_i|y) \tag{3.2}$$

Todos los i podemos representar en la Ecuación 3.3.

$$P(y|x_1, x_2, \ldots, x_n) = \frac{P(y) \times \prod_{i=1}^{n} P(x_i|y)}{P(x_1, x_2, \ldots, x_n)} \tag{3.3}$$

Donde i varía de 1 a n. En términos simples, esto puede escribirse como muestra la Ecuación 3.4.

$$posterior = \frac{anterior \times probabilidad}{evidencia} \tag{3.4}$$

Desde $P(y|x_1, x_2, \ldots, x_1)$ es constante, el modelo se expresa en la Ecuación 3.5.

$$P(y|x_1, x_2, \ldots, x_n)P(y) \times \prod_{i=1}^{n} P(x_i|y) \tag{3.5}$$

Bajo las suposiciones anteriores de independencia entre las características donde cada una es condicionalmente independiente de cualquier otra, la distribución condicional sobre la variable de clase que se va a predecir, y puede ser representada usando la Ecuación 3.6.

$$P(y|x_1, x_2, \ldots, x_n) = \frac{1}{z}P(y) \times \prod_{i=1}^{n} P(x_i|y) \tag{3.6}$$

Donde la medida de evidencia, $Z = p(x)$ es un factor de escala constante dependiente de las variables de entidad. A partir de esta ecuación, podemos construir el clasificador NB, combinándolo con una regla conocida como la regla de decisión MAP, que significa máximo a posteriori. Pasar a los detalles estadísticos sería imposible en el ámbito actual, pero al utilizarlo, el clasificador puede representarse como una función matemática que puede asignar una etiqueta de clase predicha $\hat{y} = C_k$ para algún k usando la Ecuación 3.7.

$$\hat{y} = argmax_{k \in \{1,2,\ldots,K\}} P(Ck) \times \prod_{i=1}^{n} P(x_i|y) \tag{3.7}$$

A menudo se dice que este clasificador es simple, muy evidentemente por su nombre y también por varias suposiciones que hacemos acerca de nuestros datos y características que podrían no serlo en el mundo real. Sin embargo, este algoritmo sigue funcionando muy bien en muchos casos de uso relacionados con la clasificación, incluyendo la clasificación de documentos de varias clases, el filtrado de spam, entre otros. Este algoritmo se puede entrenar

muy rápido en comparación con otros clasificadores y también funcionan bien incluso cuando no se tienen suficientes datos de entrenamiento. Los modelos a menudo no funcionan bien cuando tienen muchas características, y este fenómeno se conoce como la maldición de la dimensionalidad. NB se ocupa de este problema desacoplando las distribuciones de características condicionales relacionadas con la variable de clase, llevando a que cada distribución sea estimada independientemente como una distribución de una sola dimensión.

NB Multinomial es una extensión del algoritmo anterior para predecir y clasificar datos, se utiliza cuando se tiene más de dos clases. En este caso los vectores de características se suelen suponer que son recuentos de palabras del modelo bolsa de palabras, pero los pesos basados en TF-IDF también funcionan. Una limitación es que las características negativas basadas en el peso no se pueden alimentar en este algoritmo. Esta distribución se puede representar como $py = \{p_{y1}, p_{y2}, \ldots, p_{yn}\}$ para cada clase y, y el número total de características es n, que podría representarse como el vocabulario total de palabras o términos distintos en la analítica de texto. A partir de la ecuación precedente, $p_{yi} = P(x_i|y)$ representa la probabilidad de la característica i en cualquier muestra de observación que tenga un resultado o clase. El parámetro p_y puede ser estimado con una versión suavizada de la estimación de máxima verosimilitud (con frecuencia relativa de ocurrencias), y representado en la Ecuación 3.8.

$$\hat{p_{yi}} = \frac{F_{yi} + \alpha}{F_y + \alpha n} \tag{3.8}$$

F_{yi} en la ecuación 3.9 es la frecuencia de ocurrencia de la característica i en una muestra para la etiqueta de clase.

$$F_{yi} = \sum_{x \in TD} x_i \tag{3.9}$$

F_y en la ecuación 3.10 es la frecuencia total de todas las características de la etiqueta de clase y de nuestro conjunto de datos de entrenamiento.

$$F_y = \sum_{i=1}^{|TD|} F_{yi} \tag{3.10}$$

Hay una cierta cantidad de suavizado con la ayuda de los previos $\alpha \geq 0$, que tiene en cuenta las características que no están presentes en los puntos de datos de aprendizaje y ayudan a deshacerse de problemas de probabilidad cero.

Algunas configuraciones específicas para este parámetro se utilizan con bastante frecuencia. El valor de $\alpha = 1$ se conoce como suavizado de Laplace, y $\alpha < 1$ se conoce como suavizado de Lidstone. La librería scikit-learn proporciona una excelente implementación para NB Multinomial y la utilizamos cuando construimos nuestro clasificador de textos.

3.5.2. Árboles de decisión

El algoritmo de árboles de decisión según [37] genera un árbol de decisión de forma recursiva al dividir los datos de entrenamiento, de acuerdo con los valores de las características. Este algoritmo está categorizado como aprendizaje basado en similitudes. Los nodos no terminales del árbol de decisión representan pruebas en una o más características, mientras que los nodos terminales representan la salida predicha, al saber la clase. En el árbol de decisión resultante cada trayectoria (de la raíz a una hoja), identifica una combinación de valores de características asociadas con una clasificación particular. En cada nivel del árbol, el algoritmo elige la función que divide con mayor eficacia los datos, de acuerdo con la ganancia más alta de información.

Por ejemplo, una mañana de sábado en particular podría describirse como: pronóstico: nublado; temperatura: fresco; humedad: normal y ventoso: falso.

Cada objeto en el universo pertenece a uno de un conjunto de clases mutuamente excluyentes. Para simplificar el siguiente tratamiento, asumiremos que solo hay dos clases denotadas P y N, aunque la extensión a cualquier número de clases no es difícil. En las tareas de inducción de dos clases, los objetos de clase P y N a veces se refieren como instancias positivas e instancias negativas, respectivamente, del concepto siendo aprendido. El

otro ingrediente principal es un conjunto de objetos de entrenamiento cuya clase es conocida. La tarea de inducción es desarrollar una regla de clasificación que puede determinar la clase de cualquier objeto a partir de los valores de sus atributos. La pregunta inmediata es si el o los atributos proporcionan suficiente información para hacer esto. En particular, si el conjunto de entrenamiento contiene dos objetos que tienen valores idénticos para cada atributo y aún así pertenecen a diferentes clases, es claramente imposible diferenciar a qué clase pertenece. En tal caso, los atributos se calificarán de forma inadecuada para el conjunto de entrenamiento y, por lo tanto, para la tarea de inducción.

Como se mencionó anteriormente, una regla de clasificación se expresará como un árbol de decisión. Se utiliza un pequeño conjunto de entrenamiento que utiliza los atributos de "sábado por la mañana". Cada objeto se muestra el valor de cada atributo, junto con la clase del objeto (aquí, la clase P identifica las mañanas son adecuadas para alguna actividad no especificada). Un árbol de decisión que correctamente clasifica cada objeto en el conjunto de entrenamiento se da en la Fig. 3.1.

Las hojas de un árbol de decisión son nombres de clase, otros nodos representan pruebas basadas en atributos con una rama para cada posible resultado. Para clasificar un objeto, comenzamos en la raíz del árbol, evalúe la prueba y tome la rama apropiada para el resultado. El proceso continúa hasta que se encuentre una hoja, en cuyo momento se afirma que el objeto pertenece a la clase nombrada por la hoja. Tomando el árbol de decisión de la Fig. 3.1, este proceso concluye que el objeto que apareció como un ejemplo al comienzo de esta sección, y que no es miembro del conjunto de entrenamiento, debe pertenecer a la clase P. Tenga en cuenta que solo un subconjunto de los atributos se puede encontrar en una ruta particular desde la raíz del árbol de decisión a una hoja; en este caso, solo el atributo pronóstico se prueba antes determinando la clase. Si los atributos son adecuados, siempre es posible construir un árbol de decisión que clasifica correctamente cada objeto en el conjunto de entrenamiento, y generalmente hay muchos árboles de decisión. La esencia de la inducción es ir más allá del conjunto de entrenamiento, es decir, para construir un árbol

de decisión que clasifique correctamente no solo los objetos del conjunto de entrenamiento, pero también otros objetos (no vistos). Para hacer esto, el árbol de decisión debe capturar alguna relación significativa entre la clase de un objeto y los valores de sus atributos. Dada la elección entre dos árboles de decisión, cada uno de los cuales es correcto sobre el conjunto de entrenamiento, parece sensato preferir el más simple sobre la base de que es más probable que capture la estructura inherente al problema. El árbol más simple, por lo tanto, se espera que clasifique correctamente más objetos fuera del conjunto de entrenamiento.

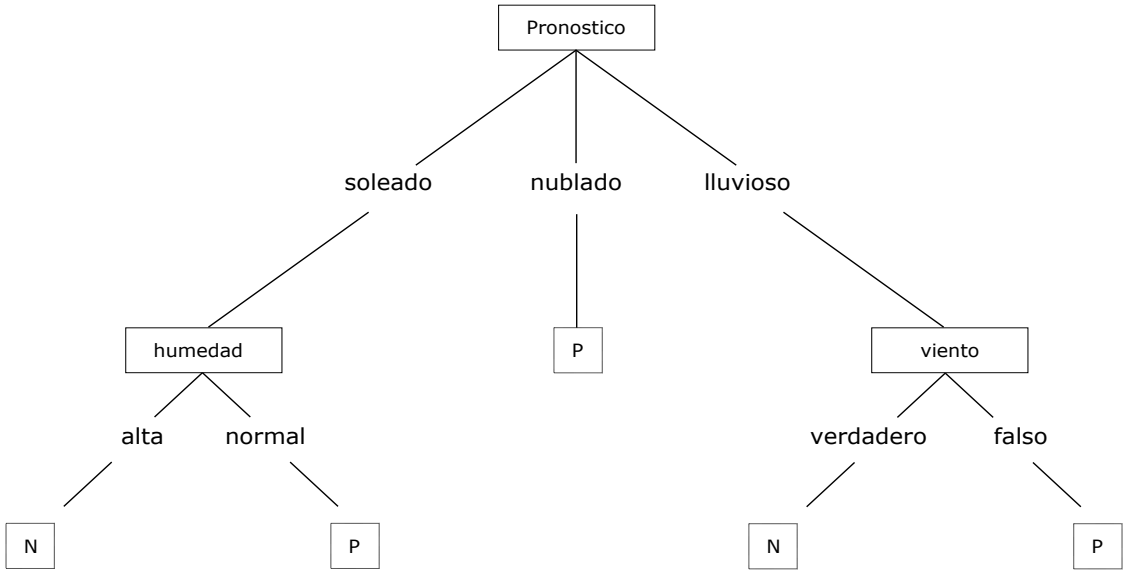

Figura 3.1: *Árbol de decisión*

3.5.3. kNN

El algoritmo kNN según [15] pertenece a la familia de algoritmos de clasificación "vagos". El principio básico de funcionamiento es el siguiente: cada muestra invisible se compara con un número de muestras de entrenamiento preclasificados, y se evalúa su similitud de acuerdo con una medida sencilla que es la distancia, (por ejemplo, la distancia euclídea normalizada), con el fin de encontrar la clase de salida asociada. El parámetro k permite especificar el número de vecinos, es decir, la formación de muestras a tomar en cuenta para la clasificación. En este trabajo nos centramos en tres modelos kNN con k igual a 1, 2, y 5.

El kNN clasificador empleado en este trabajo sigue la aplicación se describe en [3] .

3.5.4. SVM (Máquinas de vectores de soporte)

Las SVM por sus siglas en inglés (support vector machine) fueron introducidas en la literatura por primera vez en [7], son algoritmos de clasificación discriminativos basados en un hiper-plano de separación, según el cual, las nuevas muestras se pueden clasificar. El mejor hiper-plano es el que tiene el margen máximo, es decir, la distancia mínima más grande, que se obtiene a partir de las muestras de entrenamiento y se calcula basándose en los vectores de soporte (es decir, las muestras del conjunto de entrenamiento). El clasificador SVM empleado en este trabajo es la implementación se describe en [13].

SVM es un algoritmo de aprendizaje supervisado utilizado para la clasificación, regresión y detección de anomalías. SVM utiliza una clasificación binaria, dado un conjunto de datos de entrenamiento clasificados previamente asignamos una clase específica, el algoritmo SVM puede ser entrenado en base a estos datos, de tal manera que puede asignar una de las dos clases para datos futuros.

Este algoritmo representa las muestras de datos de entrenamiento como puntos en el espacio de tal manera que los puntos pertenecientes a cualquier clase pueden ser separados por una gran brecha entre ellos, llamados hiperplano, y los nuevos puntos de datos a predecir se asignan a las clases basándose en qué lado de este hiperplano caen. Este proceso es para una clasificación lineal. Sin embargo, SVM también puede realizar la clasificación no lineal por un enfoque interesante conocido como un truco kernel, donde las funciones kernel se utilizan para operar en espacios de características de alta dimensional que no son separables linealmente.

El algoritmo SVM toma un conjunto de puntos de datos de entrenamiento y construye un hiperplano de una colección de hiperplanos, para un espacio de características de alta dimensionalidad. Cuanto más grandes sean los márgenes, mejor será la separación, por lo que se pueden dar errores de generalización más bajos del clasificador. A continuación, se

presenta la representación formal y matemática del SVM.

Teniendo un conjunto de datos de entrenamiento de n puntos de datos $(\vec{x_1}, y_1), \ldots, (\vec{x_n}, y_n)$ tales que la clase variable $y_i \in \{-1, 1\}$, donde cada valor indica la clase correspondiente al punto $\vec{x_1}$. Cada punto de datos $\vec{x_1}$ es un vector de características. El objetivo del algoritmo SVM, es encontrar en el hiperplano, el máximo margen que separa el conjunto de puntos de datos que tienen el rótulo de clase de $y_i = 1$ del conjunto de puntos de datos que tienen el rótulo de clase $y_i = -1$ de tal manera que la distancia entre el hiperplano y la muestra de los puntos de datos de cualquier clase más cercana a ella se maximiza. Estos puntos de datos de muestra se conocen como vectores de soporte. La Fig. 3.2, muestra cómo se ve el espacio vectorial en el hiperplano.

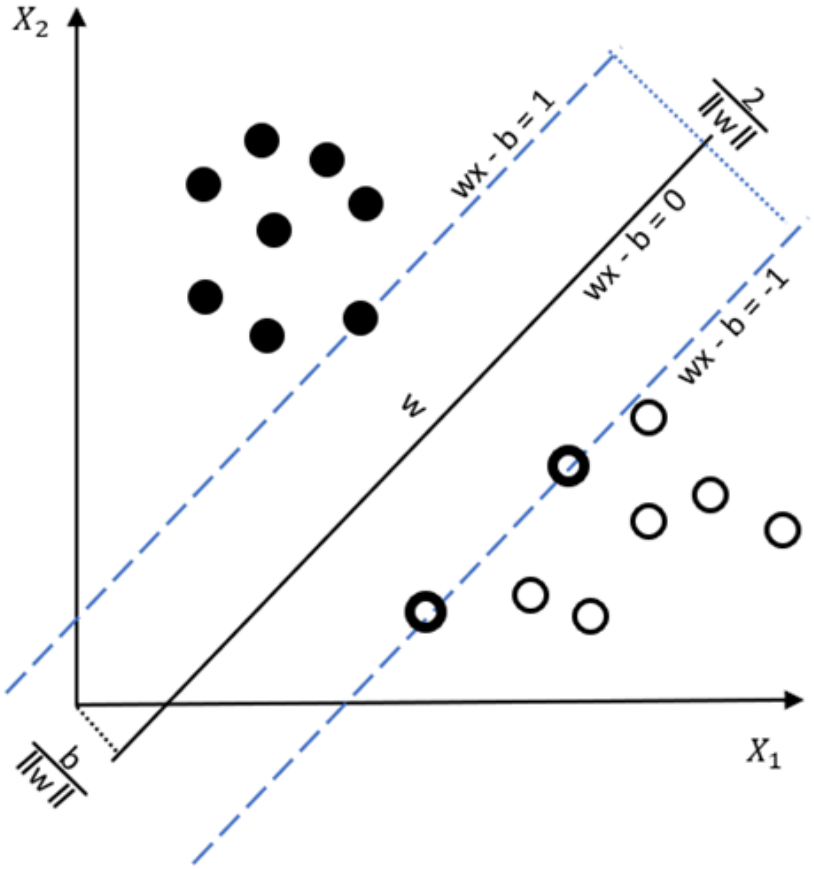

Figura 3.2: *SVM de dos clases representando en el hiperplano y vectores de soporte*

3.6. Aplicaciones del Aprendizaje Automático

El Aprendizaje automático tiene una amplia gama de aplicaciones, sus algoritmos han sido implementados en diversas áreas del conocimiento, por ejemplo: análisis del mercado de valores, haciendo predicciones de los precios de acciones; clasificación de secuencias de ADN, para determinar los genes responsables de alguna patología; reconocimiento del habla y del lenguaje escrito, vídeo juegos, robótica entre otros [41, 19, 31]. A continuación, se describen de forma breve otras aplicaciones del aprendizaje automático.

Búsquedas en línea Un ejemplo muy conocido del uso de técnicas de aprendizaje automático se puede ver en los motores de búsqueda. Estos predicen los resultados de las búsquedas que se realizan, a la vez que mejoran los resultados en base al comportamiento de los usuarios [49].

Reconocimiento de imágenes Se utilizan diversos algoritmos de aprendizaje automático para poder encontrar algo en particular o agrupar zonas de una imagen. Un ejemplo común de esto es el reconocimiento de caracteres, que analiza una imagen para encontrar letras, agruparlas y descifrar textos. Otro ejemplo es la detección y reconocimiento de personas, presencia humana en cámaras de vídeo vigilancia, o en los teléfonos inteligentes para el reconocimiento facial [12].

Segmentación de audiencia El aprendizaje no supervisado se puede usar para crear y descubrir patrones no conocidos en el comportamiento de los clientes de una empresa, aplicación o de una página web. Estos algoritmos también permiten descubrir grupos de clientes que no se conocía, o descubrir características en común de grupos de clientes. Esto permite que las empresas puedan realizar campañas dirigidas a audiencias específicas [52].

Seguridad informática En el ámbito de la seguridad informática también se utiliza el aprendizaje automático para combatir el software malicioso (malware). Los sistemas antivirus utilizan algoritmos que aprenden las características del software malicioso ya conocido, para clasificarlos en familias. Esto reduce en gran medida la complejidad de la detección de

nuevas variantes de software malicioso de familias conocidas [43].

Detección de fraudes Otra aplicación es la detección de fraude en el uso de tarjetas de crédito [58]. Las técnicas de aprendizaje automático se pueden emplear para predecir las transacciones sospechosas y no sospechosas mediante el uso de clasificadores. De forma similar trabajan las aplicaciones o servicios de anti-spam filtrando los correos no deseados.

3.7. Proceso del aprendizaje automático

Construir un modelo de aprendizaje automático, no se reduce solo a utilizar un algoritmo de aprendizaje automático; sino que es todo un proceso que suele involucrar los pasos que se muestran en la Fig. 3.3:

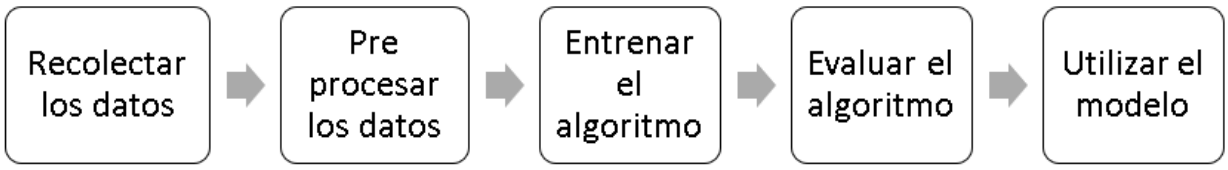

Figura 3.3: *Proceso de aprendizaje automático*

3.7.1. Recolectar los datos

Recolectar los datos consiste en juntar, agrupar o conseguir los diferentes datos a utilizar, podemos recolectar los datos desde diversas fuentes, por ejemplo, datos de un sitio web, utilizando una API o desde una base de datos. Podemos también utilizar otros dispositivos que recolectan los datos por nosotros; o utilizar datos que son de dominio público. El número de opciones que tenemos para recolectar datos es sumamente diverso. Esta parte del proceso parece obvia, pero es uno de los más complicados y conlleva mucho tiempo.

3.7.2. Pre-procesar los datos

Pre-procesar los datos, consiste en que una vez que tenemos los datos, los datos obtenidos tengan el formato correcto para alimentar el algoritmo de aprendizaje. Es prácticamente

inevitable tener que realizar varias tareas de preprocesamiento antes de utilizar los datos. Igualmente, este punto suele ser mucho más sencillo que el paso anterior.

3.7.3. Explorar los datos

Finalizados los pasos anteriores, podemos realizar un preanálisis para corregir los casos de valores faltantes o intentar encontrar a simple vista algún patrón en los mismos que nos facilite la construcción del modelo. En esta etapa suelen ser de mucha utilidad las medidas estadísticas y los gráficos en 2 y 3 dimensiones para tener una idea visual de cómo se comportan nuestros datos. En este punto podemos detectar valores atípicos que debamos descartar; o encontrar las características que poseen mayor influencia para realizar una predicción.

3.7.4. Entrenar el algoritmo

Entrenar el algoritmo requiere utilizar las técnicas de aprendizaje automático, en esta etapa alimentamos o introducimos al o los algoritmos de aprendizaje los datos que hemos procesado en las etapas anteriores. Los algoritmos deben ser capaces de extraer información útil de los datos preprocesados, para luego realizar predicciones de forma eficiente.

3.7.5. Evaluar el algoritmo

Evaluar el algoritmo consiste en poner a prueba la información o conocimiento que el algoritmo obtuvo del entrenamiento del paso anterior. Evaluamos que tan preciso es el algoritmo en sus predicciones y si no se obtiene el rendimiento esperado, podemos volver a la etapa anterior y continuar entrenando el algoritmo cambiando algunos parámetros hasta lograr un rendimiento aceptable.

3.7.6. Utilizar el modelo

Utilizar el modelo consiste en poner a prueba el modelo seleccionado, utilizando los nuevos datos, con el fin de etiquetarlos de forma correcta. Se evalúa el rendimiento del

modelo y en caso de no obtener el resultado esperado se regresa a revisar todos los pasos anteriores, hasta obtener buenos resultados.

3.8. Clasificación de textos

Para entender la clasificación de texto, es necesario entender el alcance de los datos textuales y lo que realmente queremos decir con la clasificación. Los datos textuales utilizados para la clasificación pueden ser cualquier texto que van desde una frase, oración, o un documento completo con párrafos de texto, que se puede obtener de blogs, o cualquier parte de la Web.

La clasificación de texto es también llamada clasificación de documentos, para cubrir todas las formas de contenido textual. Los documentos podrían ser definidos como una forma de representación concreta de pensamientos o eventos que podrían estar en forma escrita, discurso grabado, dibujos o presentaciones.

Un sistema de clasificación de texto puede clasificar con éxito cada documento a su o sus clases correctas, basándose en las propiedades inherentes del documento.

Matemáticamente, podemos definir: dada cierta descripción y atributos d para un documento D , donde $d \in D$, y tenemos un conjunto de clases o categorías predefinidas como se muestra en la ecuación 3.11:

$$C = \{c_1, c_2, c_3, \ldots, c_n\} \tag{3.11}$$

El documento real D puede tener muchas propiedades inherentes y atributos, que lo conducen a ser una entidad en un espacio de alta dimensión. Usando un subconjunto de ese espacio con un conjunto de descripciones y características descritas por d, deberíamos ser capaces de asignar el documento original D a su clase C_x correcta usando un sistema de clasificación de texto T.

Esto puede ser representado por la ecuación 3.12:

$$T : D \rightarrow C_x \tag{3.12}$$

La Fig. 3.4 muestra de manera conceptual la clasificación de textos. Los algoritmos de clasificación de texto empleados en este trabajo se detallan en el capítulo 3.

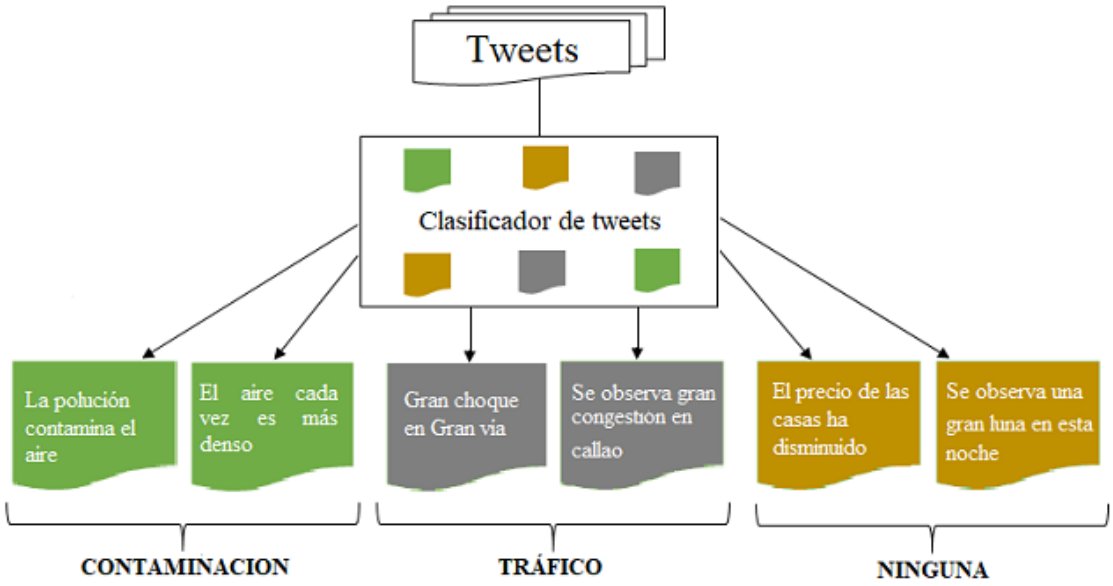

Figura 3.4: *Representación conceptual de alto nivel del proceso de clasificación de texto*

Capítulo 4

Descripción del caso de estudio

En este capítulo se describe el caso de estudio realizado, empezando con la arquitectura del sistema de detección de tráfico y contaminación, después se describe la configuración del ambiente de desarrollo para el sistema, a continuación, se detallan los procesos que realiza el sistema y, por último, se presenta la evaluación del sistema con varios modelos de clasificación de texto.

4.1. Arquitectura del Sistema de detección tráfico y contaminación.

La arquitectura del sistema de clasificación utilizada en este trabajo está compuesta de 3 módulos:

- Obtención de tweets y preprocesamiento.
- Procesamiento de tweets.
- Clasificación de tweets.

El propósito del sistema propuesto es obtener tweets desde la red social Twitter, para procesarlos aplicando minería de texto, con el fin de clasificarlos asignando etiquetas de clases definidas, en nuestro caso tráfico, contaminación, tráfico y contaminación o ninguna de las anteriores. Por último, como se muestra en la Fig. 4.1, mediante el análisis, preprocesamiento

y clasificación de los tweets, el sistema será capaz de clasificarlos en las clases anteriormente indicadas.

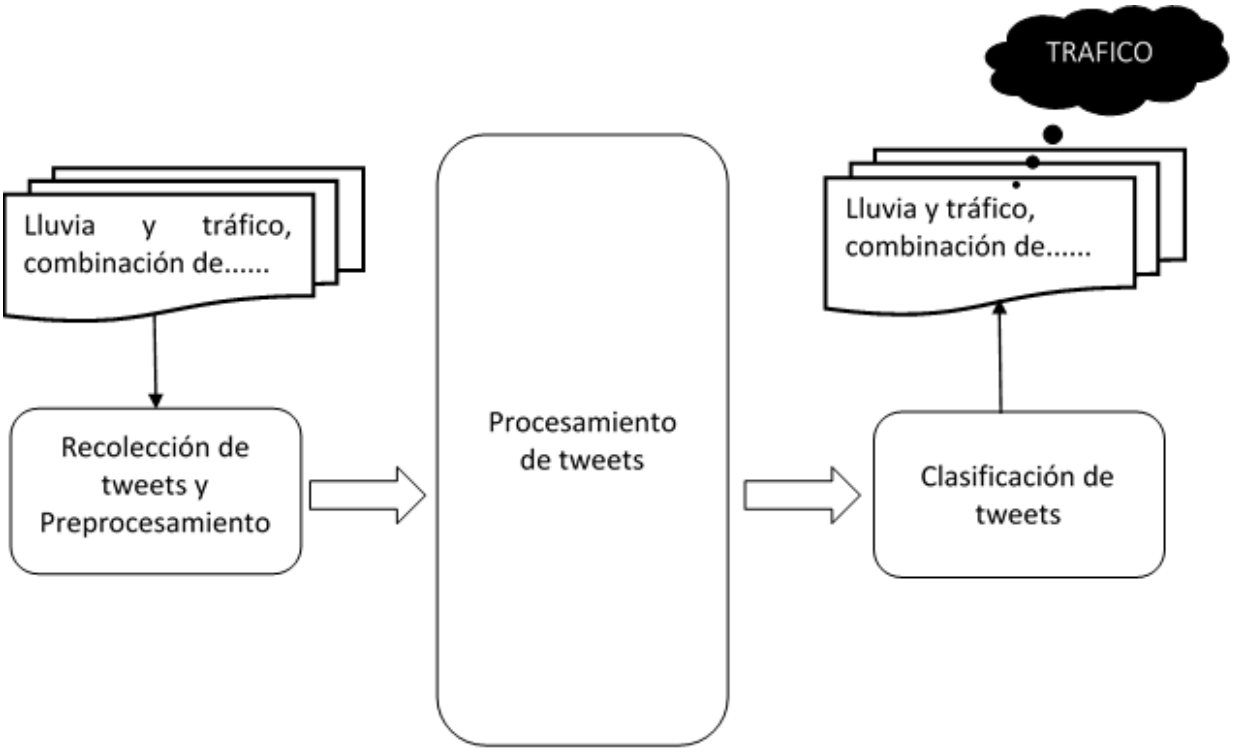

Figura 4.1: *Arquitectura del sistema para la detección de tráfico a partir del análisis de la información de Twitter*

Las principales herramientas que hemos utilizado son mongo y Python, que hemos empleado principalmente para los datos preprocesamiento y elaboración de minería de texto. Los Tweets fueron proporcionados por la facultad de Geología de la Universidad Complutense de Madrid.

La "obtención de tweets y preprocesamiento" y la "Clasificación de tweets" son módulos que requieren el establecimiento de valores óptimos con parámetros específicos, utilizando el aprendizaje supervisado. Con este objetivo, hemos utilizado un conjunto de entrenamiento compuesto por un conjunto de tweets previamente preprocesados y clasificados de forma manual. Más adelante se especificará con mayor detalle cómo se establecen los parámetros concretos de cada módulo durante la etapa de aprendizaje supervisado.

4.1.1. Obtención de tweets y preprocesamiento

El primer módulo es la "Obtención de tweets y preprocesamiento", para esto la facultad de Geología de la UCM extrae los tweets publicados, en base a uno o más criterios de búsqueda (por ejemplo, coordenadas geográficas), para este caso específico tweets publicados en Madrid. Cada uno de los tweets contienen: ID del Twitter, ID de usuario, la fecha y hora, las coordenadas geográficas, y el texto del tweet. El texto puede contener información adicional, como hashtags, enlaces, menciones, y caracteres especiales. En este trabajo, tomamos sólo los tweets de lengua española en cuenta. Sin embargo, el sistema se puede adaptar fácilmente para hacer frente a diferentes idiomas, para esto basta con obtener los tweets en otros idiomas.

Después los tweets son pre-procesados con el fin de extraer el texto de cada tweet y eliminar todos los metadatos asociados a este, utilizamos Python para obtener únicamente la fecha, hora creación y el texto del Tweet. Adicional creamos una columna "palabras" que contiene el texto del Tweet cambiando todos los caracteres a minúsculas y separándolos en palabras. Al final de esta elaboración, cada tweet aparece como una cadena, es decir, una secuencia de palabras. Denotamos al jth tweet pre-procesado por el primer módulo como $tweet_j$, con $j = 1, \ldots, N$, N es el número total de los tweets obtenidos, por ejemplo, en la Tabla 4.1 el valor de N es 3.

Tabla 4.1: *Tweets obtenidos*

Fecha hora creación	texto	palabras
2016-12-29 09:15:25	Sin prohibiciones ,libre circulación ,sin protocolos ,marca España #photo #car #day #free... https://t.co/9cqmlb9uj1	sin prohibiciones ,libre circulación ,sin protocolos...
2016-12-30 10:41:06	Preguntan en Cuatro a la concejala de circulación por qué no ponen transporte gratuito en días de restricción. Eso a Cifuentes, chati.	preguntan en cuatro a la concejala de circulación...
2016-04-27 17:11:47	Benditos filtros que me quitan la cara de congestión. Alergias a mí, ahora que empieza lo bueno... https://t.co/97DwahtoeS	benditos filtros que me quitan la...

4.1.2. Procesamiento de tweets

El segundo módulo, "Procesamiento de tweets", está dedicado a transformar el conjunto de los tweets pre-procesados, es decir, un conjunto de cadenas, en un conjunto de vectores numéricos para ser procesados por el módulo de "Clasificación de los tweets". Para este objetivo se utilizan algunas técnicas de minería de texto a cada uno de los tweets pre-procesados. En la Fig. 4.2 se indican los pasos a seguir para el procesamiento de tweets.

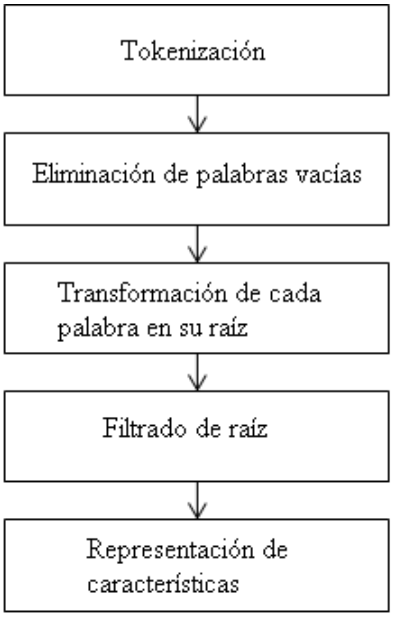

Figura 4.2: *Procesamiento de tweets*

A continuación, se describen de forma detallada los pasos realizados de minería de textos pertinentes para la Procesamiento de tweets:

Tokenización es el primer paso del proceso de minería de texto consiste en la transformación de palabras. Durante este paso realizamos la limpieza de Tweet, por ejemplo, acentos, apóstrofes, guiones, tabulaciones y espacios. En el sistema propuesto, este proceso elimina todos los signos de puntuación y divide cada tweet en tokens. Los tokens corresponden a palabras del tweet; esto se denotará como bolsa de palabras. Al final de este paso, cada $tweet_j$ se representa como la secuencia de palabras contenidas en él. Denotamos la bolsa de

palabras como se indica en la Ecuación 4.1

$$tweet_j^T = \{t_{j1}^T, \ldots, t_{jh}^T, \ldots, t_{jH1}^T\} \tag{4.1}$$

Dónde t_{jH}^T es cada token y H_j es el número total de tokens en $tweet_j^T$

Tomando el siguiente tweet como ejemplo:

"Lluvia y tráfico, combinación ideal para hacer un día de mier... coles con congestión (@ Sonae SR - Worten HQ - @wortenes) https://t.co/0QzdHDysRs "

El resultado después de realizar el proceso de tokenización será:
<lluvia>, <y>, <tráfico>, <combinación>, <ideal>, <para>, <hacer >, <un>, <día>, <de>, <mier>, <coles>, <con>, <congestion>, <sonae>, <sr>, <worten >, <hq>, <wortenes>

Eliminación de palabras vacías (en inglés Stop-word filtering) consiste en la eliminación de palabras que proporcionan poca o ninguna información para el análisis de textos. Usualmente las palabras vacías son artículos, conjunciones, preposiciones, pronombres, etc. Otras palabras favoritas son las que no tienen significación estadística, es decir, que normalmente aparecen con mucha frecuencia en las oraciones de la lengua considerada (palabras vacías específicas del idioma), o en el conjunto de textos que se analiza (palabras vacías específicas de dominio), y por lo tanto se puede considerar como ruido [37]. Los autores en [15] han demostrado que las 10 palabras más frecuentes en textos y documentos del idioma inglés son aproximadamente del 20-30 % de los tokens en un documento dado. En el sistema propuesto, la lista de palabras vacías para el idioma español fue descargado del sitio Web Snowball Tartarus [51] y ampliada con otras palabras vacías definidas por nosotros por ejemplo rayes, serio, suda, piro,etc.

Al final de este paso, cada *tweet* se reduce así a una secuencia de tokens relevantes. Denotaremos la j_{th} palabra vacía filtrada del tweet como se muestra en la Ecuación 4.2:

$$tweet_j^{SW} = \{t_{j1}^{SW}, \ldots, t_{jk}^{SW}, \ldots, t_{jK_j}^{SW}\} \tag{4.2}$$

Donde t_{jk}^{SW} es el kth token relevante y K_j es el número total de tokens relevantes en

$tweet_j^{SW}$,con $K_j \leq H_j$. Recordemos que un token relevante es un token que no pertenece al conjunto de palabras prohibidas.

A partir del ejemplo de tokenización:
<lluvia>, <y>, <tráfico>, <combinación>, <ideal>, <para>, <hacer >, <un>, <día>, <de>, <mier>, <coles>, <con>, <congestion>, <sonae>, <sr>, <worten >, <hq>, <wortenes>

Al realizar el proceso de eliminación de palabras vacías, el conjunto de tokens resultantes será:
<lluvia>, <tráfico>, <combinación>, <ideal>, <para>, <hacer >, <día>, <congestion>, <sonae>, <sr>, <worten >, <hq>, <wortenes>

Transformación de cada palabra en su raíz es el proceso de reducción de cada palabra (i.e., token) a su tallo o forma raíz (stem), mediante la eliminación de su sufijo. El propósito de este paso es agrupar palabras con el mismo tema teniendo la semántica estrechamente relacionados. En el sistema propuesto, el stemmer aprovecha el Snowball Tartarus Stemmer [51] para el idioma español, basado en el algoritmo de Porter [33]. Por lo tanto, al final de este paso cada *tweet* se representa como una secuencia de stems o raíces extraídas de las fichas contenidas en él. Denotamos la jth stemmed (derivado) *tweet* como se muestra en la Ecuación 4.3:

$$tweet_j^S = \{t_{j1}^S, \ldots, t_{jl}^S, \ldots, t_{jL_j}^S\} \tag{4.3}$$

Dónde t_{jl}^S es la raíz lth y L_j, con $L_j \leq K_j$, es el número total de raíces en $tweet_j^S$.

A partir del ejemplo de eliminación de palabras vacías, el conjunto de tokens resultantes será el siguiente:
<lluvia>, <tráfico>, <combinación>, <ideal>, <para>, <hacer >, <día>, <congestion>, <sonae>, <sr>, <worten >, <hq>, <wortenes>

Al realizar el proceso de transformación de cada palabra a su raíz y agrupación de términos se obtendrán las siguientes raíces:

<lluvi>, <trafic>, <combin>, <ideal>, <hac>, <dia>, <congestion>, <sona>, <sr>, <wort>, <hq>, <worten>

Filtrado de raíz consiste en reducir el número de raíces de cada tweet. En particular, cada tweet es filtrado por remoción del conjunto de raíces que no pertenecen al conjunto de raíces relevantes. Para realizar este paso aplicamos TF-IDF para eliminar palabras que aparecen en la mayoría de los tweets y que por tanto no permiten distinguir unos tweets de otros. El conjunto de F raíces relevantes $RS = \{\widehat{S}_1, ..., \widehat{S}_f, ..., \widehat{S}_F\}$ se identifican durante la etapa supervisada de aprendizaje que será detallado en la sección 4.2.

Al final de este paso, cada tweet se representa como una secuencia de raíces relevantes. Nosotros denotaremos los j_{th} tweets filtrados como se muestra en la Ecuación 4.4

$$tweet_j^{SF} = \{t_{j1}^{SF}, \ldots, t_{jp}^{SF}, \ldots, t_{jP_j}^{SF}\} \tag{4.4}$$

Donde $t_{jp}^{SF} \in RS$, pth es la raíz relevante y P_j es el número total de raíces relevantes en $tweet_j^{SF}$, con $P_j \leq F$

A partir del ejemplo de transformación de cada termino a su raíz y agrupación de términos:

<lluvi>, <trafic>, <combin>, <ideal>, <hac>, <dia>, <congestion>, <sona>, <sr>, <wort>, <hq>, <worten>

Al realizar el proceso de filtrado de raíz se obtendrá el vector F con las raíces relevantes seleccionadas en la fase de entrenamiento:

$F = [congestion, atasc, lluvi, retencion, \ldots, trafic, metrom, vial]$

Representación de características consiste en la construcción del vector con características numéricas para cada tweet. Con el fin de clasificar los tweets de forma correcta, se debe representar cada tweet con el mismo número de características.

En particular, nosotros consideramos el F-dimensional conjunto de características

$$X = \{X_1, \ldots, X_f, \ldots, X_F\}$$

correspondiente al conjunto de raíces relevantes.

Para cada $tweet_j^{SF}$ definiremos el vector:

$$X = \{X_{j1}, \ldots, X_{jf}, \ldots, X_{jF}\}$$

Donde se establece cada elemento como se muestra en la Ecuación 4.5

$$X_{if} = \begin{cases} \omega_f & \text{si la raíz } \widehat{S}_f \in tweet_j^{SF} \\ 0 & \text{si no pertenece} \end{cases} \tag{4.5}$$

En la ecuación 4.5, ω_f es el peso numérico asociado a la raíz relevante $\widehat{S}_f$, se detalla cómo se calcula ω_f en la sección 4.2.

A partir del ejemplo de vector F:

$$F = [congestion, atasc, lluvi, retencion, \ldots, trafic, metrom, vial] \tag{4.6}$$

Al realizar el proceso de representación de características se obtendrá el vector ω con los pesos equivalentes a cada token relevante.

$$\omega F = [0, \omega_{atasc}, \omega_{lluvi}, 0, \ldots, \omega_{trafic}, 0, 0] \tag{4.7}$$

En la Fig. 4.3, se resume el proceso de minería aplicados a un tweet del módulo de "Procesamiento de tweets".

4.1.3. Clasificación de tweets

El tercer módulo, "Clasificación de los tweets", asigna a cada tweet elaborado una etiqueta de clase, relacionada con eventos de tráfico o contaminación. La salida de este módulo es una colección de N tweets etiquetados. Para lograr el objetivo de etiquetar cada tweet, se emplea un modelo de clasificación. Los parámetros del modelo de clasificación han sido identificados durante la etapa de aprendizaje supervisado. En el capítulo 3 se detallan los diferentes modelos de clasificación que han sido considerados y se realiza la comparación de estos.

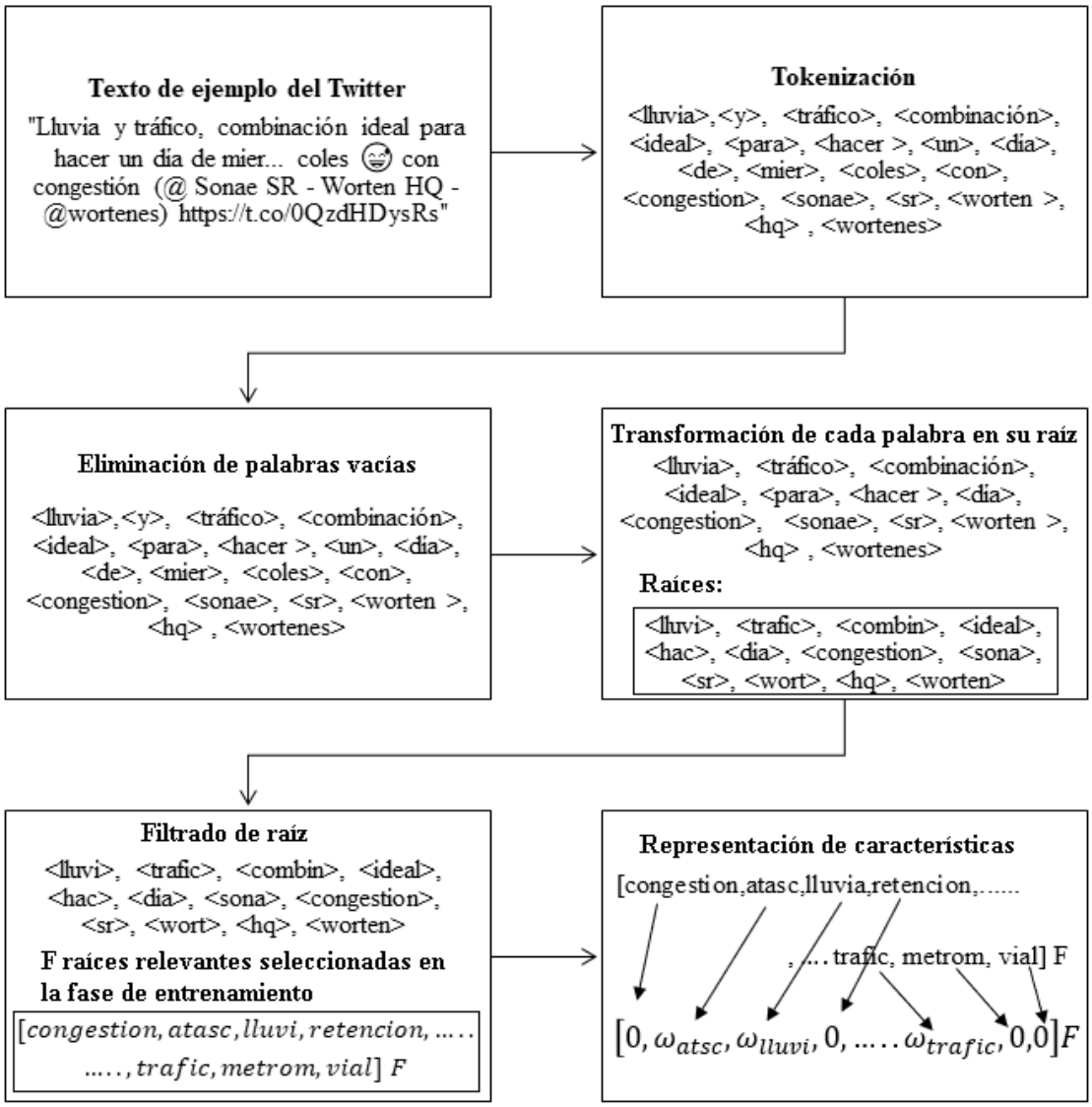

Figura 4.3: *Proceso de minería de texto aplicado a un tweet de la muestra*

4.1.4. Herramientas utilizadas

Para este propósito se utilizan las siguientes herramientas de software libre y gratuito: **Anaconda** es una distribución de Python que recopila muchas de las bibliotecas necesarias en el ámbito de la computación científica. Esta distribución incluye multitud de utilidades que nos facilitarán el trabajo y es ideal para empezar [59].

Jupyter notebook es una aplicación web de código abierto que permite crear y compartir documentos que contienen código vivo, ecuaciones, y texto explicativo [35]. Esta aplicación se puede ejecutar en los diferentes navegadores Web ya se Google Chrome, Explorer, Opera, Firefox entre otros, la ventaja de este es que en nuestro caso nos permite la ejecución en vivo del lenguaje de programación Python, con lo cual podemos ir escribiendo y probando el código de forma más rápida y podremos detectar errores de manera oportuna.

Mongodb es una base de datos NoSQL de código abierto, almacena los datos en colecciones que son estructuras de tipo JSON llamados documentos y tienen una estructura clave-valor. La estructura de almacenamiento es tan flexible que uno de los hechos importantes que se comparten al introducir esta base de datos es que; distintos documentos en la misma colección no deben tener obligatoriamente los mismos campos o estructura. Inclusive documentos con campos en común no tienen necesariamente que tener el mismo tipo de dato [30].

Robomongo es una Interfaz gráfica de usuario, que permite interactuar como admin de Mongo, bajo el mismo motor JavaScript que usa la Shell de Mongo, por lo tanto, todo lo que se escribe en la Shell de Mongo funciona en Robomongo. También ofrece la opción de autocompletado.

Pandas es una biblioteca de código abierto, que proporciona estructuras de datos de alto rendimiento y dispone de herramientas que facilitan y agilizan el análisis de datos con el lenguaje de programación Python. Fue construida sobre la librería NumPy, es útil para trabajar con series, matrices, operaciones aritméticas y permite realizar operaciones del álgebra relacional, lo cual nos ayuda al momento de realizar el preprocesamiento de los tweets. Proporciona estructuras de datos como son las Series y DataFrame. La estructura DataFrame permite trabajar con los datos que son representados en forma de tabla, por lo cual se podrían ver como tabla de SQL o una hoja de cálculo. Las filas están indexadas, lo que minimiza el tiempo de búsqueda. Las columnas están etiquetadas y cada etiqueta corresponde al nombre de la columna. Un objeto del tipo Series es semejante a un DataFrame pero con una única columna. Se trata de un objeto de tipo array de una dimensión que tiene

asociado un array de índices. El acceso y manejo de los DataFrames y Series es sumamente sencillo y rápido lo cual agiliza el proceso de tratamiento de datos [1].

NLTK kit de herramientas de lenguaje natural NLTK es la plataforma principal para la creación de programas Python para trabajar con datos de lenguaje humano. Proporciona interfaces fáciles de usar y recursos, junto con un conjunto de bibliotecas de procesamiento de texto para clasificación, tokenización, derivación, etiquetado, análisis y razonamiento semántico. Es importante saber que NLTK es un proyecto libre, de código abierto y comunitario, además es multiplataforma [10].

Scikit-learn

Scikit-learn es una librería de aprendizaje automático usada en Python. Dispone de herramientas simples y eficientes para la minería de datos y el análisis de datos, presenta varios algoritmos de clasificación , regresión y agrupación como SVM, random forests, y k-means [14].

4.2. Configuración del sistema

Como se dijo anteriormente, se requiere una etapa de aprendizaje supervisado para llevar a cabo la configuración del sistema, el esquema del proceso se detalla en Fig. 4.4.

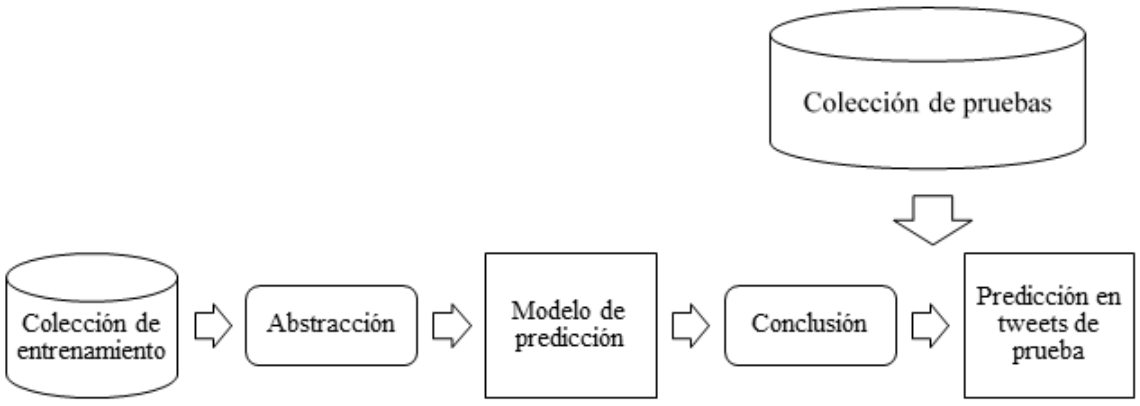

Figura 4.4: *Proceso de entrenamiento del sistema de clasificación*

En particular, necesitamos identificar el conjunto de raíces relevantes, los pesos asociados con cada uno de ellas, y los parámetros que describen los modelos de clasificación. Empleamos una colección de 1500 tweets etiquetados como conjunto de entrenamiento. Durante la etapa de aprendizaje, cada tweet se transforma aplicando la tokenización, filtrado utilizando palabras vacías, transformación de cada palabra a su raíz, filtrado de raíz y representación de características.

El conjunto completo de las raíces se construye como se muestra en la Ecuación 4.8:

$$CS = (U_{j=1}^{Ntr} tweet_j^S) = \{S_1, \ldots, S_q, \ldots, S_Q\} \tag{4.8}$$

CS es la unión de todas las raíces extraídas de los N_{tr} tweets del conjunto de entrenamiento. $tweet_j^S$ es el conjunto de raíces que describe la jth tweet después del paso de eliminación de palabras vacías y transformación de cada palabra a su raíz en el conjunto de entrenamiento.

Posteriormente se calcula el peso de cada raíz en CS, lo que nos permite establecer la importancia de cada raíz s_q en la colección de tweets del conjunto de entrenamiento, mediante el uso de la frecuencia inversa de documento (IDF) como índice:

$$wi = IDF_i = ln\frac{N_t r}{N_i} \tag{4.9}$$

Dónde $N_t r$ es el número de tweets del conjunto de entrenamiento y N_i es el número de tweets del conjunto de entrenamiento con el término i [46]. El índice de IDF es una versión simplificada de la TF-IDF, donde TF considera la frecuencia de una raíz específica dentro de cada tweet. En los experimentos realizados en nuestra muestra de tweets rara vez aparece la misma raíz más de una vez. Para los experimentos utilizamos el peso de las palabras relevantes.

Con el fin de seleccionar el conjunto de raíces relevantes, se aplicaron árboles de decisión. Los tweets son descritos por un conjunto S1,. . . , Si,. . . , SI de I características, donde cada característica Si corresponde a la raíz si. Los posibles valores de característica S_i son i y 0. Donde i es el peso calculado previamente.

Por último, identificamos el conjunto de raíces relevantes RS, seleccionamos las raíces con peso positivo mayor que a 0.5 y valores relevantes basándonos en arboles de decisión.

La última parte de la etapa de aprendizaje supervisado se refiere a la identificación de los modelos de clasificación más adecuados y la configuración de sus parámetros estructurales. Se tuvo en cuenta varios algoritmos de clasificación ampliamente utilizados en la literatura para tareas de clasificación de textos [5] , a saber, i) SVM [60], ii) NB [23], iii) los árboles de decisión[37], iv) k-Vecinos más cercanos (kNN) [3]. Los algoritmos de aprendizaje utilizados para construir los clasificadores mencionados fueron explicados en el capítulo anterior.

Capítulo 5

Evaluación del sistema

En este capítulo, se presentan la evaluación del sistema propuesto. Se realizaron varios experimentos utilizando cuatro conjuntos de datos diferentes. Para cada conjunto de datos se construyó y se compararon modelos de clasificación diferentes: SVM, NB, y kNN (con k igual a 1, 2 y 5). A continuación, describimos cómo generamos los conjuntos de datos para completar la configuración del sistema. Posteriormente presentamos los resultados obtenidos y las métricas estadísticas utilizadas para evaluar el desempeño de los clasificadores. Por último, proporcionamos una comparación con algunos resultados extraídos de otras obras de la literatura.

5.1. Descripción de los conjuntos de datos

Construimos cuatro conjuntos de datos diferentes, es decir, un conjunto de datos T (clase tráfico), N (clase no tráfico ni contaminación, C (clase contaminación) y CT (clase contaminación y tráfico). Para cada conjunto de datos, se recogieron los tweets en el idioma español de la ciudad de Madrid mediante el establecimiento de criterios de búsqueda en este caso las coordenadas geográficas de Madrid. Entonces, los tweets fueron etiquetados de forma manual, mediante la asignación de la etiqueta de la clase correcta.

5.1.1. Conjunto de datos T (clase tráfico) y N (clase no tráfico)

El primer conjunto de datos consta de tweets que pertenecen a dos clases posibles: 1. Tweets relacionados con el tráfico (T) 2. Tweets no relacionados con el tráfico (N). Los tweets se obtuvieron de diferentes fechas del año 2016 en Madrid. Para esto en colaboración con la facultad de Geología, se obtuvo los tweets generados en la provincia de Madrid, parametrizando las coordenadas geográficas, esta información fue subida a la base de datos mongo, en la cual se realizaron varias consultas con términos relacionados con tráfico definidos por expertos de la facultad de Geología de la UCM, se obtuvieron varios archivos Excel en base a las consultas realizadas.

Los mismos posteriormente fueron leídos y clasificados manualmente, los tweets se marcaron manualmente con una de las dos etiquetas posibles, es decir, en relación con evento de tráfico en carretera (T), por ejemplo, accidentes, atascos, congestión, circulación o no relacionadas (N). Más en detalle, primero se leyó cada uno de los tweets obtenidos de las consultas previas, se interpretó y se etiquetó asignando una etiqueta de la clase de tráfico (T) a cada tweet de tráfico candidato. Entre todos los tweets candidatos de la clase tráfico, cerca del 90 % de los tweets de clase de tráfico candidato no fueron etiquetados con la etiqueta de clase de tráfico. Con el objetivo de formar correctamente el sistema, añadimos estos tweets a la clase no-tráfico (N). De hecho, hemos recogido también un número de tweets que contengan las palabras clave relacionadas con el tráfico como por ejemplo congestión, atasco, atasco sin fin, retención, lesscars, circulación, trafico, estrés, transporte público, metro, bus, bici, Madrid, autopista, autovía, centro, alerta, obras, rojo, amarillo, verde, blanco, negro, Pero en realidad no referente a las incidencias del tráfico por carretera. Tales tweets están relacionados con, por ejemplo, el comercio ilegal de drogas, el tráfico de red, o tráfico de órganos. Vale la pena señalar que, la palabra“tráfico” es empleada en varios contextos, lo que podría ocasionar una confusión al momento de la clasificación, claro ejemplo de esto es el siguiente tweet: "Stigmabase | ES - Acabar con la siniestra espiral del **tráfico** y consumo de drogas - La drogadicción produce... https://t.co/EXkqkrkr9QDetenidos tres guardias

civiles en una operación contra el tráfico de drogas...

Luego, con el fin de obtener un conjunto equilibrado de datos, se seleccionaron de forma aleatoria los tweets candidatos de la clase N (no tráfico) hasta llegar a 750 tweets de clase N, y se verifico manualmente que los tweets seleccionados no pertenecían a la clase T (clase tráfico). Por lo tanto, el último conjunto de datos tiene 1500 tweets y está equilibrado, es decir, contiene 750 tweets de cada clase.

La Tabla 5.1 presenta el texto de una muestra de los tweets obtenidos por el sistema, que fueron manualmente etiquetados, las palabras clave están en negrilla dentro del texto de cada tweet. Los tweets primero, segundo, tercero y cuarto son ejemplos de tweets de la clase tráfico (T), los tweets quinto y sexto son ejemplos de tweets que contienen palabras clave relacionadas con el tráfico, pero el contexto del tweet no se refiere al tráfico, por lo que son asignados a la clase no-tráfico (N), el séptimo y el octavo son ejemplos de tweets de la clase no tráfico (N).

Tabla 5.1: *Tweets clasificados manualmente en las clases T (tráfico) y N (no tráfico)*

Tweet	Clase
#Madrid París, Milán o Roma, otras ciudades que recurrieron antes a la circulación alterna https://t.co/1d8AM6GiSR https://t.co/Xqp9ABGsby	T
Sin prohibiciones ,libre circulación ,sin protocolos ,marca España #photo #car #day #free... https://t.co/9cqmlb9uj1	T
Preguntan en Cuatro a la concejala de circulación por qué no ponen transporte gratuito en días de restricción. Eso a Cifuentes, chati.	T
#Madrid Interrumpida la circulación en la incorporación de la M50 a la A6 tras volcar dos camiones... https://t.co/lITBXQ93eh	T
#Madrid Dispositivo especial de circulación, #movilidad y transporte por el Mapoma 2017 https://t.co/5Yu9NDf2yR https://t.co/xtuIXfmVnj	T
Benditos filtros que me quitan la cara de congestión. Alergias a mí, ahora que empieza lo bueno... https://t.co/97DwahtoeS	N
Así nos vemos actualmente en reposo sin congestión ni nada, eso sí tirando de las luces de la... https://t.co/CQL9VmEwyE	N
Lectura de los #resultados de la #votación de la #elección de los #órganos #federales de #fesmc... https://t.co/bayRYfg8Ob	N
#Madrid Llamamiento a los madrileños a donar órganos https://t.co/b4AfQj8a86 https://t.co/bDfoiCUWs6	N

La Tabla 5.2 muestra algunas de las características textuales más importantes (es decir, tallos) y su significado, relacionado con los tweets de clase de tráfico, identificado por el sistema para este conjunto de datos.

Tabla 5.2: *Características significativas relacionadas con la clase T (tráfico)*

Característica (raíz)	Palabra fuente
Congestio	Congestión
Atasc	Atasco
Señal	Señal
Choqu	Choque
Circulación	Circulación
Trafic	Trafico
Autop	Autopista
Autovi	Autovía
Alert	Alerta
Obras	Obras
Roj	Rojo
Amarill	Amarillo
Verd	Verde

5.1.2. Conjunto de datos C (clase contaminación) y N (clase no contaminación)

La otra etiqueta o clase que asignaremos a los tweets es C (clase contaminación), para ello tomaremos una muestra de tweets como hicimos en el proceso anterior con palabras clave como contaminación, anticontaminación, sin contaminación, alerta contaminaciones, cielo, nube, niebla, boina, laboina, hongo, aire, limpiar, oxigeno, ozono, troposférico, protocolo, carmena, franja negra. Estos tweets se leyeron y clasificaron manualmente, se obtuvieron 400 tweets de esta clase, como era de esperarse surgieron inconvenientes semejantes a los del momento de etiquetar la clase C (clase contaminación), dado que por ejemplo las palabras nube o circulación se puede utilizar en diferentes contextos, por ejemplo:

"No se pueden poner barreras a la luz ni al aire ni a las nubes, cuanto menos a las personas.*nPor una circulación de ciudadanos por el mundo"

Adicional a esto hay que tener en cuenta que existen tweets que pueden hablar de contaminación y tráfico CT (clase contaminación y tráfico), por ejemplo: "#Madrid La restricción de aparcamiento y circulación por protocolo de contaminación se anunciará a las 12 del..... https://t.co/T40HZmUa2Z" Por último, los tweets fueron etiquetados de forma manual con cuatro posibles etiquetas de clase.

Para tener un equilibrio se seleccionó 400 tweets de cada clase, en primer lugar, los tweets candidatos etiquetados T (tweets relacionados con el tráfico), C (tweets relacionados con la contaminación), CT (tweets relacionados tanto con contaminación como con tráfico) y N (tweets no relacionados con contaminación ni tráfico).

La Tabla 5.3 muestra una selección de tweets obtenidos por el sistema para el conjunto de datos de la clase C (contaminación), de la clase CT (contaminación y tráfico) y de la clase N (ni contaminación ni tráfico), con la respectiva etiqueta añadida manualmente.

Tabla 5.3: *Tweets clasificados manualmente en las clases C (contaminación) , CT (contaminación y tráfico) y N(ni contaminación ni tráfico)*

Tweet	Clase
La contaminación paralisa la Biodiversidad	C
Si respirar #contaminación en exceso doliese #Aranjuez este pasado fin de semana se habría... https://t.co/EQ2WYYFjuA	C
#Madrid La red de vigilancia de la contaminación se renueva cuatro años https://t.co/tNUJSOUyi8 https://t.co/TU3xPZnPMC	C
#Madrid Nuevo protocolo regional para episodios de contaminación por NO2, aún en borrador https://t.co/fDot59Hh7q https://t.co/iZrHrolVKZ	C
#Madrid En estudio el aumento de la contaminación por ozono O3 troposférico https://t.co/b8WUmD4vLQ https://t.co/KdRGyxXOri	C
Madrid! Sí, ya septiembre, sus atascos, cabreos, polución , pero es Madrid, y encima... https://t.co/GGniQqPvRl	CT
#Madrid Alcanzado el escenario 1 del protocolo por contaminación. Posibles restricciones al tráfico el viernes 28 https://t.co/6DhO7oJTpF	CT
#Madrid Madrid reduce la velocidad en la M-30 a 70 km/h este domingo por la elevada contaminación... https://t.co/hPSQmoLv6J	CT
Intentando cazar alguna Perséida desde la ciudad y su contaminación lumínica. Al final ha... https://t.co/OR9uHIdJX6	N
Respirar para aliviar el dolor. No hay mejor fuente de oxígeno. @... https://t.co/bsB9bKeY0n	N

En la Tabla 5.3, los tweets primero, segundo, tercero, cuarto y quinto son ejemplos de tweets que pertenecen a la clase C, los tweets sexto, séptimo y octavo son ejemplos de tweets que pertenecen a la clase CT; y los tweets noveno y décimo pertenecen a la clase N . Las palabras marcadas en negrilla dentro de cada tweet representan las palabras claves utilizadas para la clasificación.

5.1.3. Resultados experimentales

Al finalizar el proceso de validación manual de los tweets, se procede al entrenamiento del modelo, en este capítulo se presentan los resultados de la clasificación obtenidos mediante la aplicación de los clasificadores mencionados en el capítulo 3, a los cuatro conjuntos de datos descritos anteriormente. Dado que nuestra muestra de documentos clasificados manualmente es pequeña, se utilizará la evaluación cruzada en n iteraciones. Esta técnica consiste en dividir la muestra de tweets en n subconjuntos, se recomienda que el número de elementos sea semejante en los diferentes subconjuntos. Posteriormente se toma un subconjunto como dato para realizar la evaluación cruzada y los demás subconjuntos (n -1) como datos de entrenamiento. Se repite el proceso n veces con cada uno de los posibles subconjuntos de datos de prueba. Al finalizar cada una de las n iteraciones, se procede a calcular la media aritmética con el fin de obtener un resultado final. Esta técnica es muy precisa dado que evaluamos a partir de n combinaciones de datos de prueba y de entrenamiento, el número de n dependerá de la cantidad de datos que contenga el conjunto de tweets. Se utiliza n = 10 dado que es la medida de n más común según varios autores [56].

Entonces la exactitud de la estimación de validación cruzada [25] se representa como se muestra en la Ecuación 5.1:

$$acc_{cv} = \frac{1}{n} \sum_{(v_i y_i) \in D} \delta(I(DD_i, v_i), y_i) \tag{5.1}$$

Donde:

D es el conjunto de datos, n es el número de subconjunto de D, v_i es el subconjunto de

datos de prueba, y_i es el subconjunto de datos de entrenamiento.

Para cada clasificador se realizaron pruebas utilizando n = 10. Se divide la muestra como se muestra en la Fig. 5.1, en n subconjuntos y las clases en cada subconjunto se representan en la misma proporción. Posteriormente el modelo de clasificación es entrenado en (n - 1) subconjuntos, y el subconjunto restante es utilizado para probar el modelo. Se realiza la validación cruzada n veces, en este caso 10, utilizando como datos de prueba cada uno de los n subconjuntos una única vez.

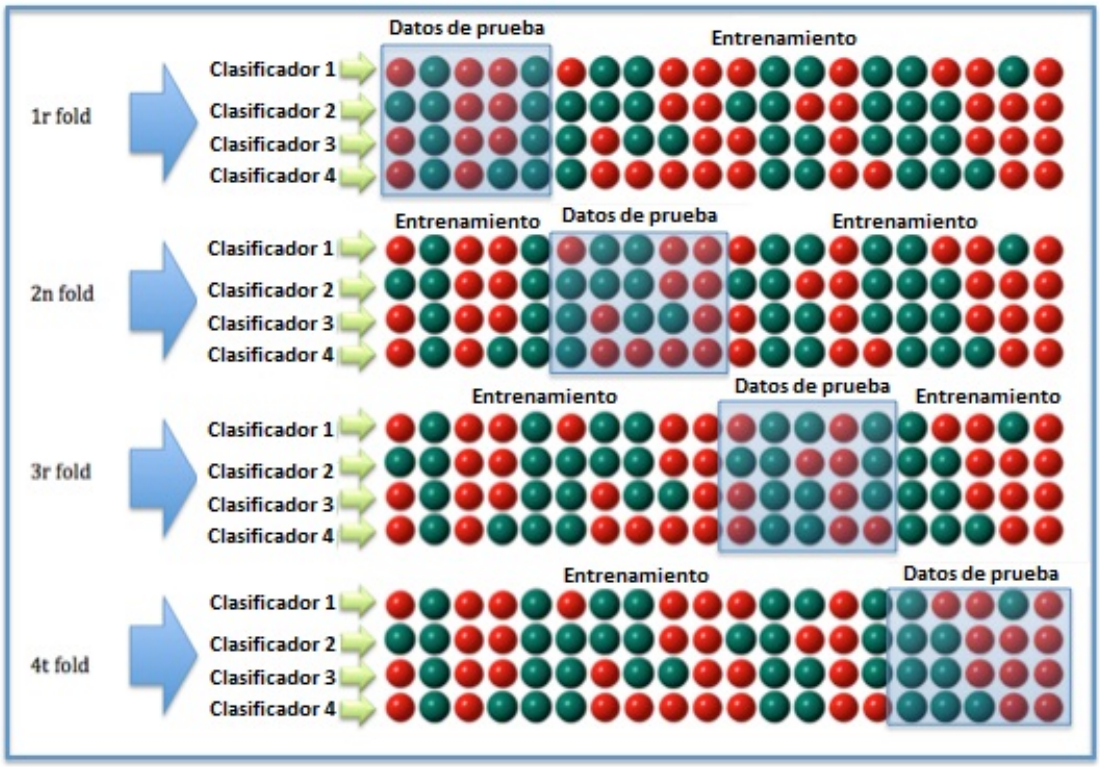

Figura 5.1: *Evaluación cruzada [56]*

Los resultados de la evaluación cruzada al final son promediados para obtener la estimación total. Repetimos n-veces la validación cruzada. Posteriormente para cada subconjunto, consideraremos un conjunto de entrenamiento específico que se utiliza en la etapa de aprendizaje supervisado para aprender tanto el preprocesamiento (es decir, el conjunto de tallos relevante y sus pesos) y los parámetros del modelo de clasificación. Para validar cual es el

mejor clasificador se emplean métricas de evaluación ampliamente conocidas como son la exactitud, precisión, recall (exhaustividad) y F-score.

La clasificación en el mejor de los casos es binaria, es decir, la clase positiva frente a la clase negativa. Pero en ocasiones pueden darse falsos positivos y falsos negativos. La corrección de una clasificación puede ser evaluado de acuerdo con cuatro valores:

1. verdaderos positivos (VP): El número de muestras positivas reales clasificados correctamente como positivos;

2. verdaderos negativos (VN): el número de muestras negativas reales clasificados correctamente como negativas;

3. falsos positivos (FP): el número de muestras negativas reales incorrectamente clasificados como positivo;

4. falsos negativos (FN): el número de muestras positivas reales incorrectamente clasificados como negativos.

Tabla 5.4: *Ecuaciones de Métricas estadísticas*

	Ecuación
Exactitud	$Ex = \frac{VP+VN}{VP+FP+FN+VN}$
Precisión	$Pr = \frac{VP}{VP+FP}$
Recall	$Re = \frac{VP}{VP+FN}$
FScore	$F = (1+\beta^2)\frac{Precision \times Recall}{(\beta^2.Precision)+Recall}$

Basado en las definiciones anteriores, ahora podemos definir formalmente las métricas estadísticas utilizadas y proporcionar, en la Tabla 5.4, las ecuaciones correspondientes. La precisión representa la eficacia general del clasificador y se corresponde con el número de

muestras clasificadas correctamente por el número total de muestras. La precisión es el número de muestras correctamente clasificadas de una clase, es decir, clase positivo, sobre el número de muestras clasificadas como pertenecientes a esa clase. Recall es el número de muestras correctamente clasificados de una clase, es decir, clase positivo, sobre el número de muestras de dicha clase; que representa la eficacia del clasificador para identificar muestras positivas. El F-score utilizado de manera frecuente con $\beta = 1$ para conjuntos de datos de clase equilibrados, es la media armónica ponderada de precisión y el recall, se utiliza para comparar diferentes clasificadores [16]. En el primer experimento, se realizó una clasificación de tweets utilizando el conjunto de datos de la clase tráfico T y de la clase no tráfico N que consta de 1500 tweets, detallados en la Descripción de los conjuntos de datos. El objetivo es asignar una etiqueta de clase T (clase tráfico) o N (clase no tráfico) a cada tweet.

La Tabla 5.5 muestra los resultados promedios obtenidos por los clasificadores SVM, KNN (donde k toma los valores 1,2 y 5) Y NB, aplicados al conjunto de tweets utilizando las clases T (tweets relacionados con tráfico) y N (tweets no relacionados con tráfico). Más en detalle, la Tabla 5.5 muestra para cada clasificador, la exactitud, y el valor por cada clase de recall, precisión, y F-score. Todos los valores se promedian a través de los 20 valores obtenidos repitiendo dos veces la validación cruzada. El mejor clasificador resultó ser la SVM con una exactitud de 85.22 %.

Tabla 5.5: *Estadísticas clasificación de tweets en las clases T (tráfico) y N (no tráfico), los mejores valores en negrilla*

Clasificador	Exactitud (%)	Precisión (%)		Recall (%)		F-Score (%)	
		T	N	T	N	T	N
SVM	**85.22**	**84.82**	**85.71**	**85.89**	**84.55**	**85.35**	**85.12**
1NN	80.85	82.02	80.26	79.99	82.10	80.99	81.17
3NN	80.69	82.10	79.46	79.11	82.28	80.58	80.85
5NN	80.62	82.46	79.11	78.58	82.63	80.47	80.83
NB	79.69	82.02	77.79	77.18	82.19	79.52	79.93

La Tabla 5.6, muestra los resultados promedios obtenidos por los clasificadores SVM, KNN (donde k toma los valores 1,2 y 5) Y NB, aplicados al conjunto de tweets (para este

caso de estudio 750 tweets) utilizando las clases C (tweets relacionados con contaminación) y CT (tweets relacionados con tráfico y contaminación). Más en detalle, la Tabla 5.6 muestra para cada clasificador, la exactitud, y el valor por cada clase de recall, precisión, y F-score. El mejor clasificador resultó ser de nuevo SVM con una exactitud de 78.84 %.

Tabla 5.6: *Estadísticas clasificación de tweets en las clases C (contaminación) y CT (contaminación y tráfico), los mejores valores en negrilla*

Clasificador	Exactitud (%)	Precision (%)		Recall (%)		F-Score (%)	
		C	CT	C	CT	C	CT
SVM	**78.84**	**78.47**	**79.29**	**79.46**	**78.22**	**78.96**	**78.75**
1NN	74.14	75.21	73.6	73.36	75.29	74.27	74.44
3NN	73.99	75.29	72.87	72.55	75.45	73.9	74.14
5NN	73.93	75.62	72.55	72.07	75.78	73.8	74.13
NB	73.08	75.21	71.34	70.77	75.37	72.93	73.3

Capítulo 6

Conclusiones y Trabajo Futuro

6.1. Conclusiones

En este trabajo, hemos propuesto un sistema para la detección de eventos relacionados con el tráfico y contaminación a partir del análisis de flujo de datos de Twitter. El sistema propuesto es capaz de analizar muestras de tweets, para clasificarlos como eventos relacionados con tráfico, contaminación, o tráfico y contaminación o ninguna estas.

Hemos utilizados varias librerías de software disponibles y diferentes técnicas de aprendizaje automático y minería de texto, para el análisis y clasificación de tweets.

Estas librerías y técnicas se han analizado, afinado, adaptado e integrado, con el fin de construir el sistema global para la clasificación de tweets.

Hemos demostrado la superioridad del algoritmo SVM, que ha logrado una exactitud de 85.22 % para la clasificación de eventos de tráfico y no tráfico, para los eventos de contaminación y eventos relacionados con tráfico y contaminación obtuvimos una exactitud de 78.84 %.

6.2. Trabajo Futuro

Los resultados obtenidos en este trabajo de fin máster abren varias direcciones para futuras investigaciones. A continuación, presentamos algunos trabajos futuros que pueden desarrollarse como resultado de esta investigación.

- Contrastar los resultados obtenidos en este trabajo, contra la información del ayuntamiento de Madrid a fin de mejorar las medidas a tomar en caso de tráfico o contaminación.

- Realizar un sistema para la detección de eventos como tráfico utilizando los tweets georeferenciados del Ecuador, con la finalidad de proporcionar soluciones a los diferentes municipios, para contrarrestar el tráfico en mi país.

- Proponer un sistema que sugiera a los conductores rutas alternativas, cuando exista un número determinado tweets sobre un evento de tráfico en algún sitio específico.

Capítulo 7

Conclusions and Future Work

7.1. Conclusions

In this work, we have proposed a system for the detection of events related to traffic and pollution, using the analysis of Twitter data flow. The system proposed is able to analyze datasets of tweets in order to classify them as events related to traffic, pollution, or traffic and pollution or none of these.

We have used several available software libraries and various machine learning and text mining techniques for the analysis and classification of tweets. These libraries and techniques have been analyzed, tuned, adapted and integrated, in order to build a global system for the classification of tweets. We have demonstrated the superiority of the SVM algorithm, which has achieved an accuracy of 85.22 % for the classification of traffic and no traffic events, for pollution events and events related to traffic and pollution it obtained an accuracy of 78.84 %.

7.2. Future Work

The results obtained in this work open several directions for further research. Next, we present some future works that can be developed as a result of this investigation.

- Compare the results obtained in this work, against the information of the city of Madrid in order to improve the actions to be taken in case of traffic or pollution.

- Make a system for the detection of events such as traffic using the geo-referenced tweets of Ecuador, in order to provide solutions to the different municipalities, to improve the traffic in my country.
- Propose a system that suggests alternative routes to drivers, when there is a certain number of tweets about a traffic event in a specific place.

Bibliografía

[1] Python data analysis library. http://pandas.pydata.org/.

[2] C. C. Aggarwal. An Introduction to social network data analytics. *Social Network Data Analytics*, 2011.

[3] D. W. Aha, D. Kibler, and M. K. Albert. Instance-based learning algorithms. *Machine learning*, 6(1):37–66, 1991.

[4] L. Alegsa. Definición de entidad, 1998 - 2018. http://www.alegsa.com.ar/Dic/entidad.php.

[5] Y. Aphinyanaphongs, L. D. Fu, Z. Li, E. R. Peskin, E. Efstathiadis, C. F. Aliferis, and A. Statnikov. A comprehensive empirical comparison of modern supervised classification and feature selection methods for text categorization. *Journal of the Association for Information Science and Technology*, 65(10):1964–1987, 2014.

[6] R. Baeza-Yates and B. Ribeiro-Neto. Modern Information Retrieval: The Concepts and Technology behind Search. *Information Retrieval*, 2011.

[7] G. Bagallo and D. Haussler. Boolean feature discovery in empirical learning. *Machine learning*, 5(1):71–99, 1990.

[8] B. Becker, R. Kohavi, and D. Sommerfield. Visualizing the simple Bayesian classifier. *Information visualization in data mining and knowledge discovery*, 2001.

[9] H. Becker, M. Naaman, and L. Gravano. Beyond trending topics: Real-world event identification on Twitter. *Icwsm*, 2011.

[10] S. Bird, E. Klein, and E. Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. .°'Reilly Media, Inc.", 2009.

[11] D. M. Blei, B. B. Edu, A. Y. Ng, A. S. Edu, M. I. Jordan, and J. B. Edu. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 2003.

[12] W. A. Castrillon, D. A. Alvarez, and A. F. López. Técnicas de extracción de características en imágenes para el reconocimiento de expresiones faciales. *Scientia et technica*, 14(38):7–12, 2008.

[13] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[14] D. Cournapeau. scikit-learn: machine learning in python; scikit-learn 0.19.2 documentation, 2018. "http://scikit-learn.org/stable/".

[15] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967.

[16] E. D'Andrea, P. Ducange, B. Lazzerini, and F. Marcelloni. Real-time detection of traffic from twitter stream analysis. *IEEE transactions on intelligent transportation systems*, 16(4):2269–2283, 2015.

[17] J. Gantz and D. Reinsel. Extracting Value from Chaos State of the Universe: An Executive Summary. *IDC iView*, 2011.

[18] S. Gao, K. Janowicz, and H. Couclelis. Extracting urban functional regions from points of interest and human activities on location-based social networks. *Transactions in GIS*, 21(3):446–467, 2017.

[19] A. Gelbukh. Procesamiento de lenguaje natural y sus aplicaciones. *Komputer Sapiens*, 1:6–11, 2010.

[20] I. Guyon and A. Elisseeff. An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research (JMLR)*, 2003.

[21] J. Hastie, Trevor, Tibshirani, Robert, Friedman. *The Elements of Statistical Learning The Elements of Statistical LearningData Mining, Inference, and Prediction, Second Edition.* 2009.

[22] F. Hogenboom, F. Frasincar, U. Kaymak, and F. De Jong. An overview of event extraction from text. *Workshop on Detection, Representation, and Exploitation of Events in the Semantic Web (DeRiVE 2011) at Tenth International Semantic Web Conference (ISWC)*, 2011.

[23] G. H. John and P. Langley. Estimating continuous distributions in bayesian classifiers. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pages 338–345. Morgan Kaufmann Publishers Inc., 1995.

[24] D. Jurafsky and J. H. Martin. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. *Speech and Language Processing An Introduction to Natural Language Processing Computational Linguistics and Speech Recognition*, 2009.

[25] R. Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145. Montreal, Canada, 1995.

[26] G. Lansley and P. A. Longley. The geography of Twitter topics in London. *Computers, Environment and Urban Systems*, 58:85–96, 2016.

[27] C. D. Manning and H. Schütze. *Foundations of Natural Language Processing.* 1999.

[28] A. Mason. NIH Public Access. 19(2):389–399, 2009.

[29] T. Mitchell. Chapther 06. *Machine Learning*, 1997.

[30] MongoDB. Mongodb for giant ideas. "https://www.mongodb.com/es".

[31] A. Moreno. Aprendizaje automático, 1994.

[32] K. P. Murphy. *Machine learning: a probabilistic perspective (adaptive computation and machine learning series)*. 2012.

[33] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[34] N. Project. Natural language toolkit nltk 3.3 documentations, 2018. "https://www.nltk.org/".

[35] Project Jupyter. Jupyter Notebook. *Jupyter website*, 2017.

[36] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng. A survey of machine learning for big data processing. *EURASIP Journal on Advances in Signal Processing*, 2016(1):67, May 2016.

[37] J. R. Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

[38] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, (August):248–256, 2009.

[39] D. Ramage, C. D. Manning, and S. Dumais. Partially labeled topic models for interpretable text mining. *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '11*, page 457, 2011.

[40] G. Reinsel and J. Gantz. The digital universe decade-are you ready? *IDC White Paper*, 2010.

[41] J. C. Riquelme Santos, R. Ruiz, and K. Gilbert. Minería de datos: Conceptos y tendencias. *Inteligencia artificial: Revista Iberoamericana de Inteligencia Artificial*, 10(29):11–18, 2006.

[42] J. Risch. Detecting Twitter topics using Latent Dirichlet Allocation. 2016.

[43] R. Rivera. *Análisis de características estáticas de ficheros ejecutables para la clasificación de Malware*. PhD thesis, Universidad Politécnica de Madrid, 2014.

[44] M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth. The author-topic model for authors and documents. *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 487–494, 2004.

[45] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake Shakes Twitter Users: Real-time Event Detection by Social Sensors. *Proceedings of the 19th International Conference on World Wide Web*, 2010.

[46] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

[47] A. Sandryhaila and J. M. Moura. Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure. *IEEE Signal Processing Magazine*, 31(5):80–90, 2014.

[48] D. Sarkar. *Text Analytics with Python*. Springer.

[49] G. Sierra. Extracción de contextos definitorios en textos de especialidad a partir del reconocimiento de patrones lingüísticos. *Linguamática*, 1(2):13–37, 2009.

[50] E. Steiger, T. Ellersiek, B. Resch, and A. Zipf. Uncovering latent mobility patterns from Twitter during mass events. *Journal for Geographic Information Science*, (Zheng):525–534, 2011.

[51] S. Tartarus. Spanish stemming algorithm. http://snowball.tartarus.org/algorithms/spanish/stemmer.html.

[52] R. Weber. Data mining en la empresa y en las finanzas utilizando tecnologías inteligentes. *Revista Ingeniería de Sistemas*, 14(1):61–78, 2000.

[53] Wikipedia. Class computer programming. https://en.wikipedia.org/wiki/Class(computer_programming).

[54] Wikipedia. Latent dirichlet allocation. https://es.wikipedia.org/wiki/Latent_Dirichlet_Allocation.

[55] Wikipedia. "tf idf". "https://es.wikipedia.org/wiki/Tf-idf".

[56] Wikipedia. Validacion cruzada, May 2018. https://es.wikipedia.org/wiki/Validacion_cruzada.

[57] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2016.

[58] O. S. Yee, S. Sagadevan, and N. H. A. H. Malim. Credit card fraud detection using machine learning as data mining technique. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, 10(1-4):23–27, 2018.

[59] A. S. C. YOLANDA GARCIA RUIZ. Instalación de anaconda. https://cv4.ucm.es/moodle/pluginfile.php/4014194/mod_resource/content/7/Tema_1/t1_instalacion_Anaconda.html.

[60] Z.-Q. Zeng, H.-B. Yu, H.-R. Xu, Y.-Q. Xie, and J. Gao. Fast training support vector machines using parallel sequential minimal optimization. In *Intelligent System and Knowledge Engineering, 2008. ISKE 2008. 3rd International Conference on*, volume 1, pages 997–1001. IEEE, 2008.

[61] Y. Zhou, S. De, and K. Moessner. Real World City Event Extraction from Twitter Data Streams. *Procedia Computer Science*, 58(DaMIS):443–448, 2016.

[62] L. Zou and W. W. Song. LDA-TM: A two-step approach to Twitter topic data clustering. *Proceedings of 2016 IEEE International Conference on Cloud Computing and Big Data Analysis, ICCCBDA 2016*, (July 2016):342–347, 2016.