

# Módulo Text Mining

Autor: Luis Gascó Sánchez, Ph.D.

Curso 2020-2021

## Índice

1.	Introducción al Text Mining .....	3
1.1.	Contexto histórico.....	3
1.1.1.	Revolución Digital .....	3
1.1.2.	La era de la información.....	4
1.2.	Text Mining .....	5
1.3.	Librerías de programación para Text Mining.....	7
1.3.1.	NLTK (Natural Language Toolkit) .....	7
1.3.2.	Spacy .....	7
1.3.3.	Gensim .....	7
1.4.	Técnicas básicas de NLP en el Text Mining .....	8
1.4.1.	Corpora y Tokens .....	8
1.4.2.	Unigramas, Bigramas, Trigramas y n-gramas.....	9
1.4.3.	Lemas y Stems.....	10
1.4.4.	Etiquetas POS (Part-Of-Speech) .....	11
1.4.5.	Named Entity Recognition (NER) .....	11
1.4.6.	Estructura sintáctica de oraciones (parsing).....	12
1.4.7.	Sentido de las palabras y semántica. ....	13
1.4.8.	Word embeddings.....	14
1.5.	Representación numérica de documentos.....	14
1.5.1.	Bag-of-Words (BoW) .....	15
1.5.2.	Métricas de palabras en BoW .....	16
1.5.3.	Limitaciones de BoW.....	17
2.	Técnicas de Text Mining.....	18
2.1.	Flujo de los datos .....	18
2.1.1.	Adquisición y preparación.....	18
2.1.2.	Transformación: .....	19
2.1.3.	Entrenamiento y evaluación .....	19
2.2.	Clasificación.....	19
2.2.1.	Transformación .....	21
2.2.2.	Algoritmo de clasificación .....	21
2.2.3.	Evaluación .....	22

2.3.	Clustering .....	23
2.3.1.	Transformación .....	24
2.3.2.	Algoritmos de clustering .....	25
2.3.3.	Métricas de similitud.....	27
2.3.4.	Validación de agrupaciones .....	29
2.4.	Topic modeling.....	30
2.4.1.	Transformación y entrenamiento .....	31
2.4.2.	Latent Dirichlet Allocation .....	31
2.4.3.	Evaluación .....	31
3.	Caso de estudio: Análisis de sentimiento de Redes Sociales .....	33
3.1.	Aplicaciones de negocio del opinion mining.....	33
3.2.	Opinion mining y niveles de análisis .....	34
3.2.1.	Niveles de análisis .....	35
3.2.2.	Diccionarios de sentimiento .....	35
3.3.	Análisis de sentimientos de documentos cortos. ....	36
3.3.1.	Adquisición y preparación de los datos .....	37
3.3.2.	Transformación .....	37
3.3.2.	Entrenamiento y validación .....	39
4.	Futuro y lecturas recomendadas .....	40
	Referencias.....	41

## 1. Introducción al Text Mining

### 1.1. Contexto histórico

#### 1.1.1. Revolución Digital

Actualmente estamos inmersos en un mundo de datos, pero, ¿cómo hemos llegado hasta este punto? Todo comienza durante la segunda mitad del siglo XX, a mediados de la década de 1950, en el que un proceso tecnológico de transición conocido como la Revolución Digital, considerada por muchos como la Tercera Revolución Industrial, transformó el mundo tal y como se conocía en ese momento.

Durante ese periodo se realizaron importantes desarrollos tecnológicos tales como el transistor [1], el ordenador personal, la consola de videojuegos o los fundamentos de Internet (ARPANET [2]). A pesar de que en otras épocas históricas también se realizaron grandes avances tecnológicos, una de las principales características de la Revolución Digital es la reducción del **periodo de adopción tecnológica** especialmente marcado en los países desarrollados. Por ejemplo, en la Figura 1, se muestra la evolución de adopción de diferentes tecnologías en los Estados Unidos de América. Se observa que, en el caso del teléfono y el automóvil, hubo que esperar casi 60 años para que el 80% de la población dispusiera de este avance tecnológico. Sin embargo, a partir de la década de los 60, se advierte una mayor pendiente en las curvas de adopción que se incrementa con cada nueva tecnología. Por ejemplo, como comparativa a los automóviles, los teléfonos móviles sólo tardaron 15 años en ser utilizados por el 80% de la población [3].

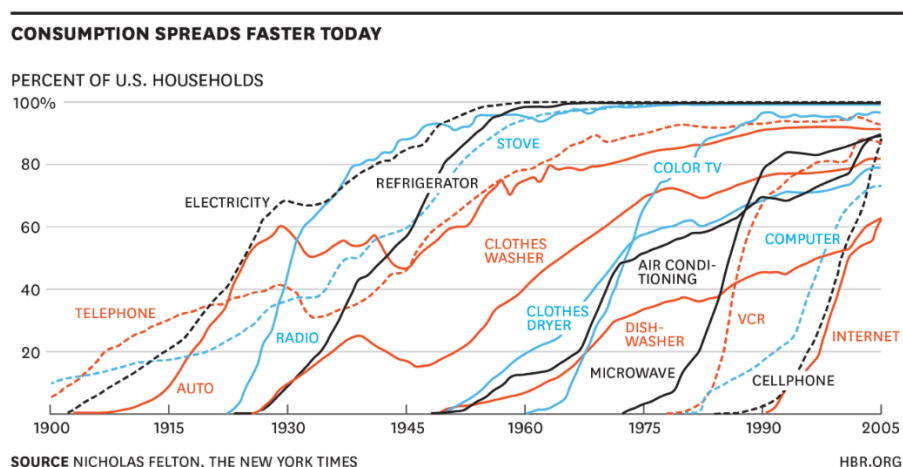


Figura 1. Curvas de adopción de tecnologías en los Estados de Unidos de América Source: The New York Times via [3].

Lo siguiente que habría que preguntarse es el porqué de esta aceleración en la adopción. La Revolución Digital no solo trajo consigo la invención de nuevos artefactos y tecnologías, sino una gran mejora en los costes de producción de dispositivos electrónicos, pudiendo incrementar los niveles de producción a un menor coste y además permitiendo incrementar la potencia de estos dispositivos de forma exponencial siguiendo leyes como la de Moore [4]. Este abaratamiento de costes, unido a un buen periodo de bonanza económica asociada a la ausencia de guerras en el territorio de los países desarrollados generó un entorno ideal para la adquisición de bienes y servicios tecnológicos que sin duda han ayudado a dar forma a la sociedad basada en datos tal y como la conocemos.

### 1.1.2. La era de la información

La Revolución Digital fue el prolegómeno de la Era de la Información o Era Digital. Esta era, en la que nos encontramos en la actualidad, está marcada por la importancia de las Tecnologías de la Información y Comunicaciones (TICs)<sup>1</sup> en la sociedad gracias al proceso de la miniaturización de la electrónica, que ha permitido modernizar el flujo y transporte de la información y mejorar los procesos de comunicación, facilitando que llegue a un mayor porcentaje de población y convirtiéndose en un motor de la evolución social [5].

La Era de la Información ha provocado grandes cambios a nivel social gracias a Internet y los dispositivos que permiten acceder a esta red. Al inicio de Internet y de la *World Wide Web*, se utilizaba como un método para acceder a la información en el que sólo las personas especializadas eran capaces de generar contenido estático. Con la llegada de la Web 2.0, centrada en el usuario, aparecieron nuevos servicios que permitieron generar contenido a cualquier persona mediante servicios como las Redes Sociales Online; pero también plataformas para acceder a la información de forma más sencilla, facilitando incrementar el impacto del contenido en la población. La importancia y el potencial del usuario dentro de la Web 2.0 tuvo tal impacto que en el año 2006 la revista *Time* nombró a “you”, todos los usuarios de Internet, como persona del año [6]. Esto denotó la importancia del usuario en el nuevo mundo conectado, donde Internet podía ser utilizado como una plataforma para compartir opiniones, sentimientos y crear redes virtuales, generando mucha información útil que podría ser útil para otros menesteres.

Las mejoras de Internet, junto a la gran adopción de otros productos como los teléfonos inteligentes, han favorecido la explosión de las Redes Sociales Online (RSO), importante fuente de generación de datos en la actualidad. La evolución tecnológica consecuencia de la Era de la Información ha permitido conectar diferentes tipos de sensores y dispositivos a la red, algo que se conoce como el Internet de las Cosas (*Internet Of Things [IoT]*) y derivó en el *Internet of Everything*. La combinación de estos factores junto a la modernización de las empresas, que cada vez están más digitalizadas, ha provocado un entorno en el que millones de datos diarios son generados a través de sensores en ciudades, información de tráfico, industria o contenidos de redes sociales.

El gran volumen de datos generados diariamente tiene la característica de ser muy variado, pudiendo clasificarlos a partir del nivel de estructuración en los que los encontramos cuando queremos enfrentarnos un proceso de análisis:

- **Datos estructurados:** Son datos que tienen un formato predefinido siguiendo un formato común. Son datos estructurados las hojas de cálculo de ventas de una empresa o sus bases de datos SQL con información sobre clientes.
- **Datos no estructurados:** Son datos cuya clasificación y análisis no es trivial por no tener un modelo predefinido de estructura. Por ejemplo, el contenido de los correos electrónicos, los informes de ventas en formato PDF de una consultora tecnológica o las publicaciones en medios sociales online incluyendo imágenes, videos y texto.

---

<sup>1</sup> Entre las TICs se encuentra cualquier producto, servicio o tecnología utilizada para almacenar, manipular, transmitir o recibir información. Dentro de esta definición se encuentran dispositivos como el ordenador personal, las televisiones inteligentes, la telefonía móvil, tecnologías como las redes 4G y servicios digitales como el correo electrónico u otras aplicaciones que funcionan en la nube.

Es importante señalar que cuando hablamos de datos estructurados nos referimos a que la información contenida no tiene estructura. Por ejemplo, una imagen está almacenada de forma estructurada con un conjunto de metadatos, pero no la información contenida en la imagen como puede ser la presencia y posición de personas presentes en esta.

El análisis de datos estructurados, aunque no es trivial, es a priori más sencillo que el de los datos no estructurados ya que tenemos una serie de características o variables predefinidas que permiten su análisis. Sin embargo, para el análisis de datos no estructurados se necesitan técnicas más avanzadas para extraer ese conjunto de características para analizarlas. Por ejemplo, en el caso de imágenes se utilizan técnicas de visión computacional, conocidas como técnicas de *Computer Vision*. En el caso de texto, que es el que nos ocupa en este módulo, se utilizan técnicas de *Text Mining*.

## 1.2. Text Mining

Debido a la diversidad de orígenes de documentos textuales y el fin de estos, el Text Mining es un área con un alto grado de multidisciplinariedad que a lo largo del tiempo han desarrollado estadísticos, lingüistas computacionales, ingenieros de Machine Learning o ingenieros informáticos. Estos profesionales han desarrollado técnicas para analizar el texto con diferentes objetivos como la recuperación de información o la clasificación de textos. Por ese motivo, y a diferencia de otros ámbitos de la ciencia de datos, la definición del *Text Mining* es bastante inespecífica, pudiendo describirse como:

**“El proceso o procesos de transformación de datos textuales no estructurados a un formato tabular que permita su análisis, identificar patrones y extraer conocimiento.”**

Ese proceso de estructuración de los datos puede aplicarse con seis fines diferentes:

- **Búsqueda y recuperación de información** (*Information Retrieval*): Consiste en el almacenado, búsqueda y obtención de documentos textuales a partir de búsquedas de palabras clave y estrategias de recuperación de la información. Por ejemplo, cuando se incorporan nuevos elementos en una base de datos bibliográfica o en Internet, es necesaria la indización automática de dichos documentos para que puedan ser recuperados con posterioridad. De hecho, cuando se realiza una búsqueda a partir de texto, se utilizan técnicas de Text Mining para procesarlo y buscar documentos relevantes que sean de interés para el usuario.
- **Agrupación de documentos** (*Clustering*): Consiste en la agrupación de textos o secciones de texto que tienen características comunes a partir de algoritmos de clustering. El Clustering se utiliza en aplicaciones en las que hay que encontrar relaciones entre muchos textos. Por ejemplo, se pueden utilizar este tipo de técnicas para detectar contenido duplicado o plagiado, para mejorar los sistemas de recomendación de noticias en internet, o para la organización natural de documentos a partir del contenido expresado en estos.
- **Clasificación de documentos**: Consiste en la categorización de textos a partir de algoritmos de clasificación que funcionan con datos previamente etiquetados. La clasificación automática de textos tiene un gran espectro de usos. Por ejemplo, en el mundo de las redes sociales como Twitter, existen multitud de links hacia noticias falsas o comentarios con incitación al odio. Gracias a sistemas de clasificación de documentos textuales se pueden entrenar sistemas de Inteligencia Artificial capaces de detectar este tipo de contenido a partir de un conjunto de datos previamente etiquetado indicando cual es el contenido no deseado.

- **Extracción de información (Information Extraction):** Consiste en la extracción de elementos de importancia dentro de documentos textuales y su relación con otros elementos dentro del documento. Por ejemplo, las empresas o instituciones tienen cientos de documentos almacenados de forma no estructurada (archivos Word, PDF, etc). Se pueden utilizar técnicas de Text Mining para analizar el contenido de esos textos y extraer información como número de teléfonos, facturaciones, datos de empleados, etc. [7]
- **Extracción de conceptos (Concept Extraction):** Consiste en la agrupación de palabras y frases de varios documentos en grupos semánticos similares. Gracias a este tipo de técnicas se pueden obtener los conceptos principales de textos de forma automática. Por ejemplo, a partir de un conjunto de 3000 reviews en Amazon, el vendedor del producto podría extraer los conceptos más relevantes en el conjunto de las opiniones sobre el producto e incluso con técnicas adicionales generar un texto resumen.
- **Procesado de Lenguaje Natural (Natural Language Processing NLP):** Consiste en la aplicación de técnicas de bajo nivel de procesamiento de lenguaje para una mejor comprensión del texto. Algunas de estas técnicas son la separación automática de un texto en frases o palabras.

Cada una de estas áreas utiliza tecnologías específicas, pero es importante mencionar que en muchos casos las herramientas desarrolladas en una rama se utilizan en el resto. Por ejemplo, se pueden utilizar técnicas de NLP para pre-procesar textos antes de su clasificación y mejorar el rendimiento del modelo.

Por último, para completar la visión de la “amalgama” del Text Mining y para facilitar su comprensión, los elementos anteriores se pueden agrupar también en las ramas de conocimiento mostradas en el diagrama de Venn de la Figura 2. De manera que, por ejemplo, las tecnologías de Procesado de Lenguaje Natural, además de formar parte del Text Mining, también son parte de la Lingüística Computacional y el Machine Learning. Por ese motivo en muchos casos se utilizan algunos términos indistintamente en la bibliografía y documentación online.

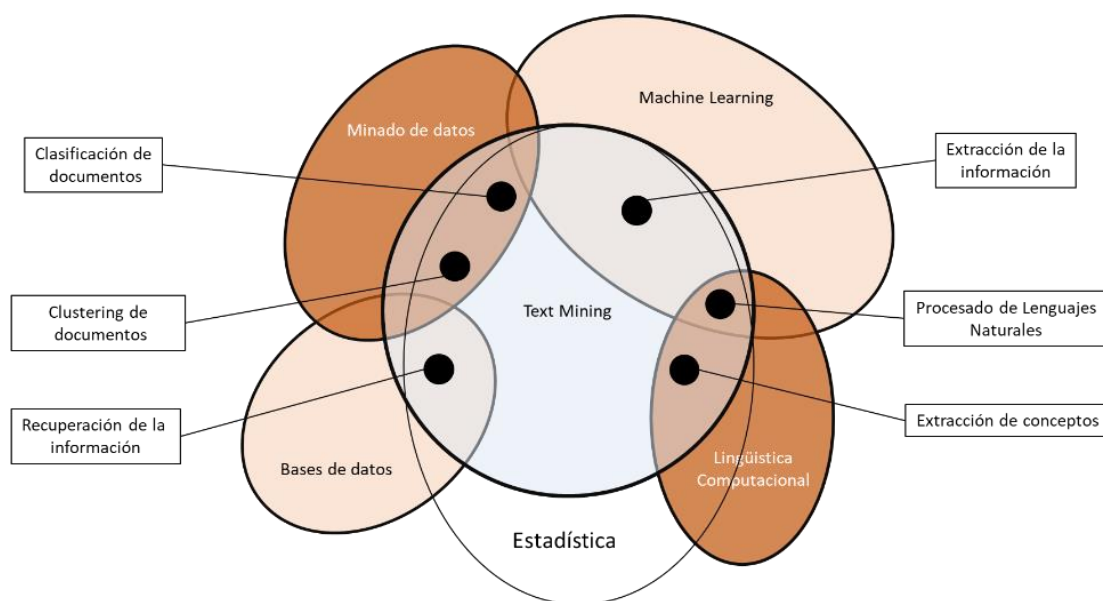


Figura 2. Diagrama de Venn de los campos de aplicación del Text Mining. Source: [8]

Debido al amplio espectro del Text Mining, tan amplio que se podría impartir un máster exclusivo en la temática y no tener tiempo para explicar todas las áreas de aplicación, y el reducido número de horas de este módulo. En este curso se introducirán las áreas más interesantes desde el punto de vista del científico de datos, que son las relacionadas con la clasificación de textos, la clusterización de documentos y la extracción de conceptos de documentos de textuales.

### 1.3. Librerías de programación para Text Mining

Gracias al rápido y eficaz desarrollo de librerías para Text Mining y NLP se están pudiendo implementar estas tecnologías en entornos de producción. El objetivo principal de estas librerías es facilitar el proceso de trabajo con datos textuales. Lo que ayuda a identificar una buena librería para desarrollar sistemas de Text Mining es lo siguiente:

- Facilitar la interoperabilidad de los datos procesados con la librería con otras librerías populares de Machine Learning o Deep Learning como Scikit-learn o Tensorflow.
- Disponer de una API fácil de usar, aprender y comprender.
- Actualización constante por parte de la comunidad de desarrolladores para que cuente con las últimas tecnologías.

Aunque existen multitud de librerías desarrolladas, aquí se mencionan las que se utilizarán a lo largo del módulo y que además tienen una gran adopción por la comunidad desarrolladora:

#### 1.3.1. NLTK (Natural Language Toolkit)

NLTK es una de las librerías principales para trabajar con textos libres que fue creada por la Universidad de Pennsylvania en el año 2001. Aunque su uso principal ha estado unido a entornos de investigación y educación, las facilidades en su uso y sus características la convierten en una de las librerías con un mayor número de recursos de aprendizaje como libros, foros o tutoriales. Contiene una gran cantidad de conjuntos de datos típicos para el aprendizaje de NLP y es muy utilizada en tareas para el preprocesado de texto antes de introducirlo en algoritmos de Inteligencia Artificial.

#### 1.3.2. Spacy

Es una librería para NLP desarrollada por Ines Montani y Matthew Honnibal. A diferencia de NLTK, que principalmente ha sido utilizada en entornos de investigación, *Spacy* se centra en proporcionar herramientas para poder incorporar sistemas de Text Mining en producción por facilidad. De hecho, su fácil interconexión con otras librerías del mundo de la ciencia de datos, junto a la incorporación de modelos pre-entrenados con técnicas de Deep Learning y su facilidad para trabajar con múltiples lenguajes de programación, la han convertido en una de las librerías más usadas, si no la que más, en la actualidad.

#### 1.3.3. Gensim

Es la librería por excelencia para realizar tareas de extracción de conceptos y de información como el *topic modeling*. Permite realizar tareas con grandes volúmenes de datos y tiene una sencilla interfaz para la realización de las tareas.



## 1.4. Técnicas básicas de NLP en el Text Mining

El Procesado de Lenguajes Naturales, *Natural Language Processing (NLP)* en inglés, es un campo de la lingüística y del aprendizaje automático, cuyo fin es definir un conjunto de técnicas para procesar lenguajes naturales y que sean comprensibles por sistemas computacionales. Pero, ¿qué es un lenguaje natural?

**“Un lenguaje natural es una forma de lenguaje humano con fines comunicativos que tiene asociado una serie de reglas sintácticas, conocidas como sintaxis”**

A diferencia de lo que ocurre en un **lenguaje formal**, como los lenguajes de programación, que han sido diseñados para ser utilizados en contextos determinados y con fines muy concretos, los lenguajes naturales ofrecen una gran versatilidad contextual que cubre las necesidades creativas del ser humano. Este hecho genera dificultades a la hora de analizar este tipo de lenguaje, ya que el significado de una frase puede diferir mucho en función de cómo, cuándo y dónde es utilizada. Esto quiere decir que, en los lenguajes naturales, además de la **sintaxis**, la **semántica** o significado específico de los componentes son claves para la comprensión del texto.

Por ejemplo, a la frase “Vamos al banco” se le pueden atribuir varios significados. El interlocutor podría referirse a un banco como un elemento del mobiliario para sentarse, o ir a un banco como institución bancaria para realizar operaciones financieras. Este es uno de los principales problemas de los lenguajes naturales, la importancia del **contexto** para comprender una oración, algo que no ocurre en los lenguajes de programación. Los seres humanos somos capaces de intuir el contexto con facilidad, algo que los sistemas informáticos no hacen tan bien y por eso se utilizan herramientas para que se pueda inferir el sentido de las oraciones en cada documento.

Con el paso de los años, los profesionales en el campo del NLP han desarrollado técnicas para analizar el lenguaje computacionalmente, muchas de estas técnicas se han convertido en esenciales para el preprocesado de textos en el Text Mining, por ello es importante definir algunos términos y procesos que se utilizan de forma generalizada en el flujo de trabajo de análisis textual.

### 1.4.1. Corpora y Tokens

Todo el proceso de análisis textual comienza con un dataset de documentos textuales, que generalmente se llama **corpus**, o corpora en plural. El corpus generalmente está compuesto de texto bruto con algunos metadatos asociados.

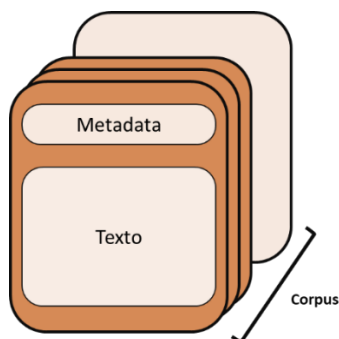


Figura 3. Representación de un corpus formado por documentos, a su vez compuestos por texto y metadatos.

El texto bruto está compuesto por una secuencia de caracteres. Antes de su análisis los textos son divididos en fragmentos más pequeños conocidos como **tokens**. Un token puede ser tanto una palabra, como un símbolo de puntuación, un número o un emoticono, en el caso de estar analizando datos de redes sociales.

El proceso de división del texto en tokens se llama **tokenización**. Como ejemplo, la frase “Francisco ha comido demasiadas patatas, ahora se encuentra mal” está compuesta por 10 tokens, 9 palabras y un signo de puntuación. El proceso de tokenización en algunos lenguajes puede ser muy complicado, pero en lenguajes como el español, inglés o francés es algo relativamente trivial y tecnológicamente superado en textos de propósito general. Sin embargo, debido a la diversidad de tipos de documentos (registros clínicos, tweets, artículos científicos, informes de ventas...) es un proceso que se debe realizar con cautela para su correcta ejecución.

También es importante la definición de **lexicón** o **vocabulario**, que es el conjunto de tokens únicos que están presentes en un corpus. Dentro de las palabras de un vocabulario se suelen quitar las palabras vacías, conocidas generalmente por su término en inglés **stopwords**. Las stopwords son palabras que tienen un uso principalmente gramatical pero que no aportan significado a la frase y complican su análisis, tales como artículos y preposiciones. De hecho, las investigaciones en la década de 1940 por George Kinsley Zipf, lingüista de la Universidad de Harvard, demostraron que en una lengua las frecuencias de uso de las palabras siguen una distribución inversa denominada Distribución de Zipf, dándose la situación que las palabras más frecuentes en los lenguajes son las stopwords [9]. Como estas partículas no aportan significado, en muchas ocasiones se puede optar por quitarlas para agilizar el proceso de entrenamiento de modelos de minado de textos.

#### 1.4.2. Unigramas, Bigramas, Trigramas y n-gramas

Los **n-gramas** son secuencias de n tokens consecutivos provenientes de un texto. La combinación de n-gramas puede proporcionar información sobre la temática de un texto. Generalmente se generan **unigramas**, que son iguales que los tokens del texto. Los **bigramas**, que son combinaciones pareadas de tokens y los **trigramas** que son triadas de tokens

En la Figura 4 se representa la frase utilizada en el ejemplo anterior señalando uno de los bigramas que la componen, uno de los trigramas y uno de los unigramas. Cualquier frase se puede descomponer en un conjunto de estas combinaciones de tokens, consiguiendo de este modo capturar el entorno de las diferentes palabras y aumentando generalmente las capacidades de los algoritmos.

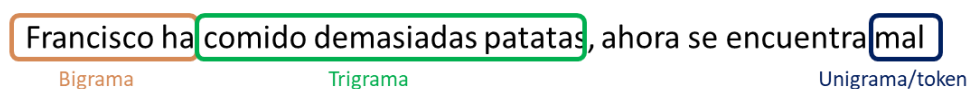


Figura 4. Ejemplo de Unigrama, Bigrama y Trigrama en una frase.

### 1.4.3. Lemas y Stems

Para este apartado es importante saber la composición de las palabras. Una palabra está compuesta por: una **raíz, lema o lexema**, que es la parte de la palabra que no varía y que indica su significado principal; y un **morfema**, que son partículas que se añaden a la raíz para la formación de nuevas palabras. Se dice que las palabras que tienen la misma raíz pertenecen a la misma **familia léxica**.

Los morfemas por otra parte pueden ser de tipo: **flexivo**, que son aquellos situados al final de las palabras y permiten modificar el tiempo verbal y el género o el número en el caso del español; y **derivativo**, que son partículas añadidas al final de los lexemas y general nuevas palabras. En este segundo grupo se incluirían los prefijos y sufijos, como por ejemplo los diminutivos.

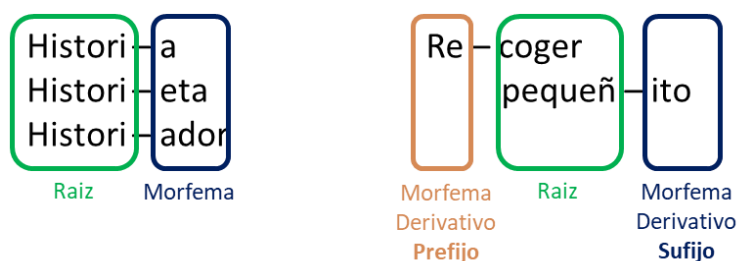


Figura 5. Ejemplos de raíces y tipos de morfemas.

Los lemas son las formas canónicas del léxico de un idioma. Debido a la diversidad de las lenguas naturales, en ocasiones es interesante utilizar técnicas para obtener el lema de las palabras de un documento y trabajar con éstos para conseguir así una reducción de la dimensionalidad en los modelos predictivos o para mejorar los resultados de búsqueda en un sistema de recuperación de la información ya que la uniformización de las palabras permitirá obtener un mayor número de resultados relevantes. Para la obtención de estas raíces existen dos procesos:

- **Stemming:** Los algoritmos más simples para la obtención de lemas<sup>2</sup> son los algoritmos de stemming, que consiste en utilizar reglas sintácticas para eliminar los morfemas de las palabras y reducirlas a una forma canónica denominada *stem*. Existen multitud de stemmers (algoritmos de stemming) desarrollados siendo los más populares los de Porter y el de Snowball:
  - *Algoritmo de Porter Stemming:* Algoritmo de stemming desarrollado para la lengua inglesa y que funciona con la mayoría de las palabras de este idioma. Consiste en eliminar la última letra de cada palabra junto a otras reglas consiguiendo normalizar los tokens.

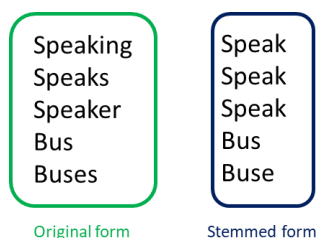


Figura 6. Resultado de aplicar el algoritmo de Porter a un conjunto de palabras.

<sup>2</sup> Cuando se utilizan algoritmos de stemming no se obtienen siempre lemas, por ese motivo en la literatura prefieren utilizar el término “stems”

- **Algoritmo de Snowball Stemmer:** Debido a la falta de algoritmos de stemming para lenguas diferentes a la inglesa, se desarrolló el algoritmo Snowball<sup>3</sup>, que es una versión mejorada del algoritmo de Porter Stemming con funcionamiento en multitud de lenguajes.

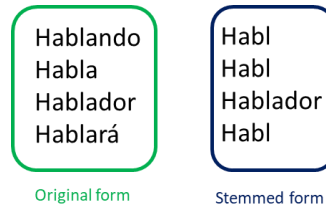


Figura 7. Resultado de aplicar el algoritmo de Snowball a un conjunto de palabras en español.

- **Lematización:** La lematización es un proceso similar al stemming con la diferencia de que no se produce un *stem* de la palabra, sino que se sustituyen los morfemas por un sufijo común, conocido como lema, para obtener una forma normalizada de la palabra. Por ejemplo, en las palabras “computes”, “computing”, “computed” la raíz es “comput”. Sin embargo, un algoritmo de lematización obtendría el infinitivo “compute” como palabra normalizada de esas tres.

#### 1.4.4. Etiquetas POS (Part-Of-Speech)

La categorización gramatical de palabras, **Part-Of-Speech Tagging**, ha sido un tema de estudio en lingüística desde hace tiempos inmemorables. La adición de categorías gramaticales a las palabras de una frase puede servir para múltiples propósitos en el Text Mining:

- **Information retrieval:** En el campo de la indexación y recuperación de textos la inclusión de información sobre POS puede ser beneficiosa ya que los nombres y los adjetivos son mejores candidatos para ser palabras clave que adverbios, verbos o pronombres
- **Clasificación:** En el análisis de sentimiento es importante extraer las palabras que se utilizan para extraer opinión, que suelen ser los adjetivos y adverbios.

Además, conocer las categorías gramaticales nos permite establecer reglas para detectar automáticamente estructuras gramaticales que se usan de forma sistemática para expresar algo. Por ejemplo, si buscamos frases que expresen opinión sobre algo podríamos buscar un conjunto de frases que sigan la estructura Pronombre + Verbo + Sustantivo → como “Yo amo el rock-and-roll”

Existen multitud de modelos pre-entrenados para extraer el tipo de palabras en distintos idiomas que en la actualidad son entrenados a partir de arquitecturas de redes neuronales profundas y que son los que utilizaremos de forma práctica en los ejercicios.

#### 1.4.5. Named Entity Recognition (NER)

La clasificación y reconocimiento de entidades nominales, conocido comúnmente como **Named-Entity Recognition (NER)**, es una tarea de análisis de textos que consiste en reconocer y asignar una etiqueta a los nombres propios de un texto. El reconocimiento de estas entidades es una parte esencial de varias

<sup>3</sup> <http://snowball.tartarus.org/texts/introduction.html>

ramas del Text Mining como el resumen automático de textos, la búsqueda y la recuperación de información, la web semántica o el *topic modeling* [10].

Hoy en día las librerías de análisis de textos proporcionan modelos pre-entrenados para el reconocimiento de estas entidades. Algunos de esos sistemas están basados en reglas gramaticales y otros en modelos entrenados con arquitecturas de redes neuronales profundas, en ambos casos son capaces de identificar bastantes de las entidades más comunes en textos de propósito general como nombre de personas, localizaciones, organizaciones o estructuras temporales. Cuando no son capaces de reconocer las entidades de interés, es necesario entrenar modelos NER específicos para identificar términos importantes en los textos. Esto es lo común en ramas como la medicina, en el que existen sistemas específicos para identificar síntomas de enfermedades o medicamentos [11].

La Figura 8 muestra las entidades reconocidas en una frase escrita en inglés. El sistema pre-entrenado es capaz de identificar que *John* es una persona nacida en la ciudad de *Chicken* del estado de *Alaska* y que además, estudia en la organización *Cranberry Lemon University*. Un simple sistema de identificación de nombres basado en un diccionario no hubiera sido capaz de identificar que *Chicken* en este caso es una ciudad y no un animal.



Figura 8. Ejemplo de entidades reconocidas por un Sistema NER.

El uso de este tipo de sistemas nos permite extraer información específica sobre un individuo concreto. En la figura se expande el texto anterior y se habla sobre los gustos de John. Los sistemas NER permiten hacer entender a un ordenador que a John le gusta ir a Starbucks, siendo John una persona y Starbucks una organización. De un modo general se puede ver como que los modelos NER permiten extraer el “quien”, “donde” y “como” de una frase, facilitando la comprensión computacional de esta.

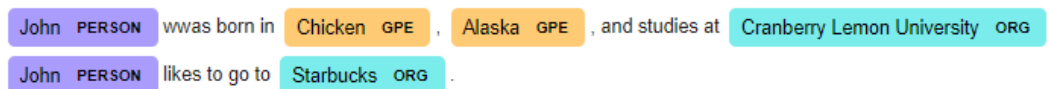


Figura 9. Ejemplo de más entidades reconocidas por un Sistema NER.

#### 1.4.6. Estructura sintáctica de oraciones (parsing)

Conocer la estructura sintáctica de una frase, o **parsing** en inglés, permite hacer análisis más exhaustivos que sirvan para identificar relaciones entre los componentes de un texto. Por ejemplo, en la Figura 10 se muestra el resultado de hacer *parsing* a una frase en inglés con un sistema pre-entrenado. Los resultados nos recordarán, y de hecho son los mismos, que los que solíamos obtener en las clases de Lengua y Literatura durante la Educación Secundaria Obligatoria. En la imagen se observa el sujeto de la frase, el predicado, los complementos verbales y las relaciones entre esos componentes mediante flechas que los unen.

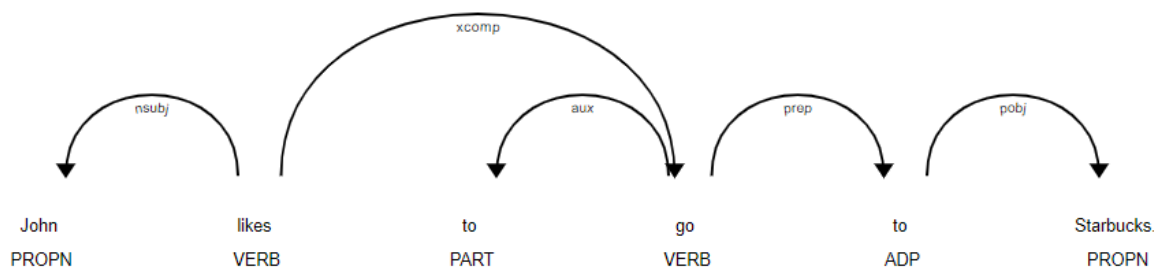


Figura 10. Ejemplo de Parsing en una frase en inglés.

Gracias a las técnicas de parsing se puede incorporar contexto gramatical a los análisis de textos. Permite conocer quién es la persona que realiza una acción y sobre quién ejerce dicha acción, algo que nos permitirá extraer relaciones útiles entre las entidades de las oraciones.

Estas técnicas son especialmente útiles en problemas de la lingüística computacional como la traducción automática, los chatbots o la predicción del lenguaje. No se utilizará demasiado en este módulo, pero es interesante saber sus posibilidades de uso, de ahí su mención en este documento.

#### 1.4.7. Sentido de las palabras y semántica.

Las palabras tienen significados, en ocasiones más de uno, que se conocen como acepciones. Uno de los principales problemas que existen a la hora de analizar textos es el sentido o acepción que tienen las palabras que lo componen. Como se mencionó anteriormente, cuando alguien utiliza la palabra “banco”, puede referirse a una institución financiera o a un elemento del mobiliario público para sentarse o descansar. Dentro del NLP se conoce como **Word Sense Disambiguation** al conjunto de técnicas y herramientas para resolver el problema de la ambigüedad semántica de palabras y frases. Existen diferentes técnicas para desambiguar o conocer el sentido de algunas palabras en un texto, como por ejemplo el estudio del contexto de la frase o la distribución de palabras utilizadas en un documento completo.

Además, existen recursos como **WordNet 3.0**, un diccionario jerárquico desarrollado por la Universidad de Princeton, que categoriza las acepciones de todas las palabras del inglés en relaciones semánticas con otras. Las dos principales relaciones son las siguientes:

- **Hiperónimos (Hypenyms):** La palabra Y es un hiperónimo de X si todos los X forman parte de Y. Por ejemplo “animal” es un hiperónimo de “perro”
- **Hipónimos (Hyponyms):** La palabra Y es un hipónimo de X si todas las Y forman parte de X. Por ejemplo “perro” es un hipónimo de “animal”.

Es importante mencionar que cada acepción de una palabra en WordNet es conocida como *synset*. Además gracias a esta estructura jerárquica se puede conocer el grado de similitud semántica de palabras en función de lo lejos que se encuentran dentro de la estructura jerárquica. El uso de este tipo de herramientas permite extraer una información más rica a nivel semántico, que además se ve complementado con otras herramientas como los embeddings y los modelos de lenguaje.

#### 1.4.8. Word embeddings.

Los **Word Embeddings** son una técnica de modelización de lenguaje y generación de características utilizadas en NLP. En un *embedding* cada palabra del vocabulario tiene asignada un vector en un espacio n-dimensional. Resulta que en este espacio multidimensional es más sencillo apreciar la relación semántica entre palabras individuales. Por ejemplo, en la Figura 11 se muestra un conjunto de 4 palabras representadas por sus respectivos vectores: *King*, *Queen*, *Woman* y *Man*.

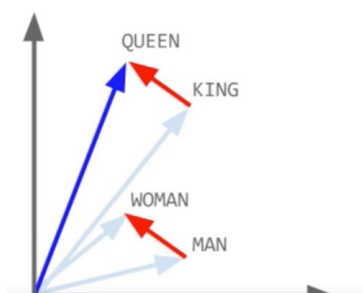


Figura 11. Representación gráfica de palabras en un embedding

La distancia existente entre los pares de palabras es la misma, ¿por qué? Porque gracias a la gran cantidad de documentos con las que ha sido entrenado el embedding, éste es capaz de capturar las relaciones semánticas y sabe que *Queen* es una mujer y que *King* es un hombre porque la distancia entre esos términos es la misma que la existente entre *Man* y *Woman*. De hecho, se podrían realizar operaciones matemáticas sobre las palabras y obtener lo siguiente:

$$\text{King} - \text{man} + \text{woman} = \text{Queen}$$

Las relaciones semánticas extraíbles de los embeddings son muy útiles para caracterizar palabras específicas dentro de un texto y de este modo extraer relaciones que de otro modo no podrían haberse extraído. De hecho estas técnicas son comúnmente utilizadas en los sistemas de detección automática de sentimiento en textos [12].

Existen muchos modelos pre-entrenados de Word Embeddings disponibles para su uso. Estos modelos han sido entrenados con millones de datos textuales provenientes de Wikipedia o Twitter, entre otros. Es conveniente utilizar un embedding que haya sido entrenado con los datos más parecidos a los que estás analizando para que se capturen las relaciones de forma correcta, ya que la forma de expresión es distinta en Wikipedia que en Twitter. Algunos populares son los de Glove[13] o Gensim.

#### 1.5. Representación numérica de documentos

Hasta el momento hemos visto algunos términos y técnicas comúnmente utilizados en Text Mining. Sin embargo, en muchas de las técnicas que se utilizan es necesario representar los textos en forma de vectores numéricos para que puedan ser procesados por modelos estadísticos o modelos de aprendizaje automático.

Como hemos visto, los textos están formados por un conjunto de elementos llamados tokens. Estos tokens están compuestos a su vez por un conjunto de caracteres que no pueden ser procesados por modelos de aprendizaje automático. Por ese motivo se aplica una técnica de vectorización o de representación numérica de textos conocida como Bag-Of-Words.

### 1.5.1. Bag-of-Words (BoW)

**Bag-of-Words** es una representación del texto basado en mostrarlo como la concurrencia de palabras dentro de un documento y está compuesto por dos elementos: 1. Un vocabulario de palabras conocidas y 2. Una métrica de las palabras que aparecen en el texto. Para comprender el proceso de construcción, seguiremos los siguientes pasos:

1. **Obtención de un corpus:** En primer lugar, debemos generar un conjunto de documentos que queremos representar con este método. Por ejemplo:

"Yo quiero agua"
"Yo quiero coca cola"
"Yo quiero agua y un agua"
"Yo no quiero vino"
"Yo quiero un entrecot"

2. **Generación del vocabulario:** Es necesario contar todas las palabras únicas que se utilizan en el corpus generado. Es recomendable transformar en minúsculas todos los términos para un mejor resultado. En nuestro caso, tenemos un vocabulario de 8 palabras distintas que se utilizan en nuestras 5 frases:

"agua", "coca cola", "entrecot", "no", "quiero", "un", "vino", "yo"
---

3. **Creación de vector de documentos:** Para ello generamos vectores en los que cada posición corresponderá a la presencia o no de una palabra del vocabulario. Como nuestro vocabulario tiene 8 palabras esa será la longitud de cada vector representando a los diferentes documentos (el orden de las palabras dentro del vector es el mostrado en el paso 2). En este caso utilizaremos como métrica de representación las palabras la suma de las ocasiones que ocurre una palabra en el documento:

"Yo quiero agua"	=	[1 0 0 1 0 0 1]
"Yo quiero coca cola"	=	[0 1 0 0 1 0 1]
"Yo quiero agua agua"	=	[2 0 0 1 1 0 1]
"Yo no quiero vino"	=	[0 0 0 1 1 0 1]
"Yo quiero un entrecot"	=	[0 0 1 0 1 1 0]

Se observa que con este método el orden de las palabras se pierde, por eso se denomina bolsa de palabras, pero es la manera más sencilla de extraer características de los documentos de nuestro corpus y se puede utilizar para construir modelos complejos..

También es importante destacar que si se introdujeran nuevos documentos cuyas palabras no están presentes en el vocabulario también se podrían codificar omitiendo las palabras desconocidas en la codificación.



Uno de los problemas del método BoW es que según va creciendo el tamaño del corpus, lo hace el tamaño del vocabulario y puede generar dificultades para entrenar un modelo por obtener un conjunto de vectores muy dispersa (*sparsity matrix*). Por ello es común utilizar técnicas para reducir el número de palabras del vocabulario como:

- Transformar todas las palabras a minúscula.
- Ignorar los signos de puntuación.
- Ignorar las stopwords, ya que apenas incorporan información a texto.
- Arreglar o corregir palabras con errores de escritura
- Reducir las palabras a su stem o lemas utilizando lematización y stemming.

También se podría seguir la misma metodología de BoW con bigramas, trigramas o algún tipo de n-grama. De manera que al final se tuviera una representación de un documento por la aparición de palabras individuales, pares de palabras o n-palabras.

### 1.5.2. Métricas de palabras en BoW

En el ejemplo anterior hemos utilizado como métrica de palabra la suma de la presencia de palabras dentro de un documento. También se puede realizar la división de cada uno de esos términos por el número total de tokens en esa frase para obtener la métrica TF (**Term Frequency, TF**). Sin embargo, existe otra métrica más eficiente que se conoce como **TF-IDF** (*Term Frequency Inverse-Document-Frequency*).

Esta métrica surge bajo la premisa de que cuando tienes un corpus procedente de un mismo tema, como por ejemplo textos de patentes, va a haber palabras como “sistema”, “método”, “proceso” que aparecen en todos los documentos. Cuando se utiliza una métrica de conteo de palabras estándar, aparecerán como palabras más importantes las que aparecen en todos los textos, pero que sin embargo no aportan un valor diferencial a cada uno de los documentos ya que aparecen en todos ellos. En cambio, palabras más específicas que ocurren menos frecuentemente tendrán un peso menor en los vectores, pero que sin embargo podrían indicar la tipología del documento. Por ese motivo surge IDF, que penaliza los tokens que aparecen de forma más común en todos los documentos y premia a los token más raros dentro de la representación vectorial. El cálculo del valor IDF para un token  $w$  se calcula como:

$$IDF(w) = \ln \frac{N}{n_w}$$

donde  $n_w$  es el número de documentos que contiene el token  $w$  y  $N$  es el número total de documentos. El resultado final de TF-IDF es el producto  $TF(w) * IDF(w)$ .

Si un token apareciera en todos los documentos, es decir que  $n_w = N$  el valor de  $IDF(w)$  sería 0 y por lo tanto el de TF-IDF también, penalizando completamente el término por no incorporar información útil a la representación del corpus.

Por ejemplo, si realizamos el cálculo para el corpus anterior obtendríamos una representación TF-IDF como la mostrada abajo. En los que se observa que las palabras que no salen en todos los documentos tienen un valor mayor que las que salen en todos ellos.

"Yo quiero agua"	=	[1.92	0	0	0	1	0	0	1]
"Yo quiero cocacola"	=	[0	2.61	0	0	1	0	0	1]
"Yo quiero agua y agua"	=	[3.83	0	0	0	1	1.92	0	1]
"Yo no quiero vino"	=	[0	0	0	2.61	1	0	2.61	1]
"Yo quiero un entrecot"	=	[0	0	2.61	0	1	1.92	0	1]

Hay que indicar que en las funciones de Python para el cálculo de estos vectores hay multitud de opciones adicionales, así que hay que estar muy atento de cuáles son los valores por defecto para evitar problemas. También indicar que este tipo de codificación (TF-IDF) no suele ser usada en las técnicas de Deep Learning (no cubiertas en este curso), pero no por ello dejan de ser importantes.

### 1.5.3. Limitaciones de BoW

Aunque ya se han mostrado anteriormente, BoW tiene algunas limitaciones que hay que tener en cuenta cuando representamos los textos con esta herramienta:

- **Significado:** Dado que el orden de las palabras se ignora, el contexto de un documento se pierde, y en muchas ocasiones se pierde la semántica de las palabras por este motivo. Como se ha visto antes, el contexto es esencial en muchas de las tareas del NLP y es un problema que hay que tener en cuenta cuando se utiliza este tipo de técnica.
- **Vocabulario:** Según se trabaja con documentos largos, y con un gran número de estos, el tamaño del vocabulario puede crecer mucho, lo que podría afectar al rendimiento en la computación de los modelos posteriores.
- **Sparsity:** Especialmente cuando el vocabulario es muy numeroso, existirán muchos ceros en la matriz de n-grams, algo que empeorará los tiempos de computación.

## 2. Técnicas de Text Mining

Una vez introducidos la terminología y los conceptos que se suelen utilizar en el Text Mining, en este apartado se tratarán algunas de las técnicas que más utilizan los *Data Scientist* cuando se enfrentan a datos textuales: la clasificación de documentos, el clustering y el topic modeling.

### 2.1. Flujo de los datos

De forma general, cuando se van a aplicar cualquiera de esas técnicas a datos textuales se sigue un flujo de trabajo común que se puede estructurar en los tres pasos mostrados en la Figura 12: adquisición y preparación, transformación y entrenamiento/validación.

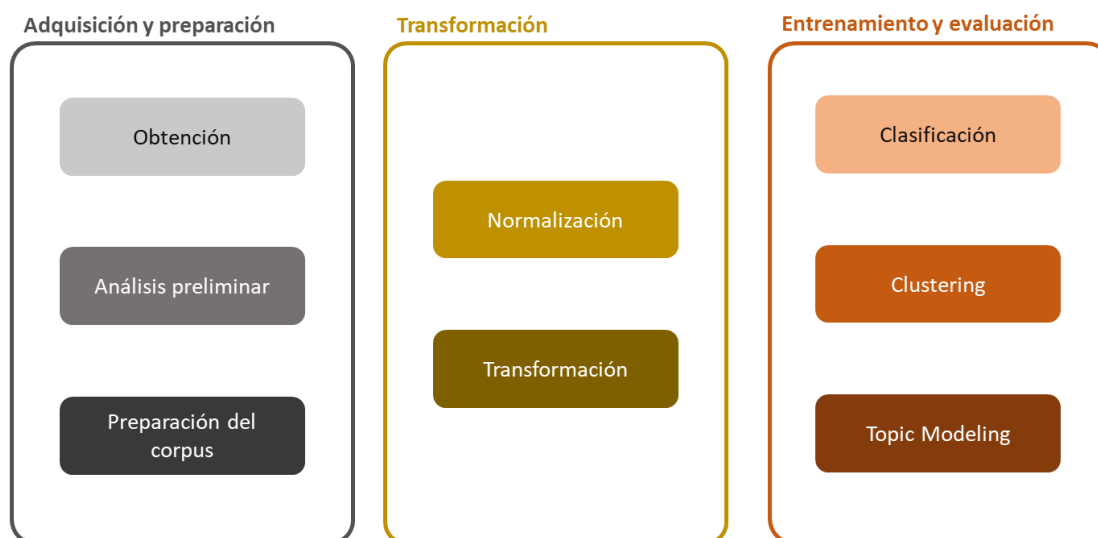


Figura 12. Estructura del flujo de trabajo en tareas de Text Mining

#### 2.1.1. Adquisición y preparación

Este bloque está compuesto a su vez de tres tareas específicas:

1. **Obtención de datos:** Consiste en definir una estrategia para la obtención y la limpieza de datos textuales en bruto para su posterior análisis. Los datos pueden obtenerse mediante técnicas de **scrapping** y de bases de datos para aprendizaje automático. Si se opta por la primera opción habrá que prestar especial atención a la limpieza de los datos para eliminar del corpus metadatos o etiquetas que suelen estar presentes cuando se descargan datos de páginas web. Si por el contrario se opta a utilizar un conjunto de datos existente, también es necesario observar que el corpus es apto para el análisis y que los creadores de éste realizaron las tareas de limpieza pertinentes.
2. **Análisis preliminar:** Como se haría en cualquier flujo de trabajo de ciencia de datos, una vez disponemos del conjunto de datos limpio es aconsejable hacer un análisis preliminar: analizar la longitud media de los documentos, ver la distribución de clases si se requiere...
3. **Preparación del corpus:** En función del tipo de técnica que se quiera implementar, será necesario dividir el corpus en un subconjunto de datos de entrenamiento, desarrollo y testeo, o incluso en un número mayor de subconjuntos para hacer validación cruzada

### 2.1.2. Transformación:

En los problemas de Text Mining es necesario transformar los documentos textuales en un conjunto de características numéricas comprensibles por los algoritmos. Este proceso se divide en dos etapas:

1. **Normalización:** Los textos en lenguaje natural presentan tal diversidad en sus formas escritas que en ocasiones es mejor normalizar los documentos antes de generar los vectores de características. Este proceso trata de uniformizar las palabras existentes en el corpus con la intención de reducir la dimensionalidad de los datos, algo que puede favorecer la velocidad de construcción de los modelos y en algunas ocasiones mejorar su eficiencia.  
Los procesos de normalización más comunes son el stemming, lematización, eliminación de signos de puntuación, transformación a minúsculas y exclusión de las palabras vacías, aunque en función de la aplicación podría no ser necesario aplicar todos esos tipos de normalización. En algunas ocasiones es recomendable aplicar algunas más como por ejemplo el remplazo de contracciones en el idioma inglés, ya que se ha demostrado que el remplazo de estas formas lingüísticas consigue mejorar el rendimiento de los modelos [14].  
En todo caso, no existe una fórmula ideal para el proceso de normalización. Lo recomendable es buscar bibliografía especializada sobre el tipo de análisis que se va a realizar y ver el tipo de procesos de normalización utilizados en ese campo.
2. **Generación de vectores de características:** Es el proceso de representar el texto en forma numérica. Se pueden utilizar las técnicas anteriormente explicadas como Bag Of Words o Bag of N-grams. En función de la aplicación será recomendable utilizar TFIDF, conteo de palabras o *embeddings*. En los últimos años se ha popularizado la representación del texto con modelos de lenguaje más avanzados como BERT, pero es importante reseñar que para algunas tareas las técnicas de representación tradicional pueden conseguir resultados óptimos sin tanto coste computacional.

### 2.1.3. Entrenamiento y evaluación

En este paso se aplicarán las estrategias de entrenamiento y evaluación específicas del modelo que se esté construyendo que serán distintas para la clasificación, clustering y topic modeling.

## 2.2. Clasificación

La **categorización o clasificación** de la información es una de las ramas del Text Mining más utilizadas en multitud de campos y aplicaciones diferentes. La premisa de las técnicas de clasificación es sencilla: a partir de un conjunto de datos con una categoría o etiqueta asignada, el objetivo es construir un sistema que sea capaz de identificar los patrones existentes en los documentos que ayuden a determinar su clase de forma automática. Dado que hay unas clases pre-asignadas, se dice que la clasificación es una tarea de Machine Learning supervisado, y el fin durante la construcción del modelo de inteligencia artificial será minimizar el error existente entre las categorías predichas por el sistema y las reales asignadas previamente.

Algunos ejemplos de aplicación en los que se utilice la clasificación de textos son:

- **Detección de spam:** Desde que se extendió el uso del correo electrónico ha existido la presencia de mensajes no deseados que recibimos en nuestras bandejas de entrada. El acceso y eliminación de estos mensajes suele costar tiempo laboral al trabajador, algo que al final tiene repercusiones en la eficiencia de las empresas. Por ese motivo, existen sistemas de detección de SPAM en el que a partir del contenido de estos mensajes pueden detectarse los correos no deseados y enviarse a la bandeja de SPAM que tienen casi todos los sistemas de correo actuales.
- **Detección del sentimiento:** Con la creciente importancia de los canales de venta online ofreciendo bienes y servicios se ha convertido en algo habitual que muchas personas den su opinión sobre la calidad de los productos recibidos. Para las empresas es importante saber cuál es el nivel de satisfacción de sus clientes, por ese motivo utilizan técnicas de clasificación de textos para analizar las reviews y saber cuáles de ellas tienen un carácter positivo y cuales negativo para analizarlas con mayor profundidad y conocer que funcionalidades de sus productos deben en el futuro.
- **Detección de discurso de odio:** Con el incremento de la polarización política en los últimos años, es normal encontrar multitud de mensajes de odio en plataformas sociales como Twitter y Facebook. Estas empresas han desarrollado sistemas de clasificación de textos para identificar el contenido que podría ser considerado de odio para limitar su difusión o incluso borrarlo.
- **Detección de noticias falsas:** Del mismo modo que ha pasado con el discurso del odio, con el paso de los años han proliferado las noticias falsas en internet. Este es un problema importante porque alienta la desinformación de la población y favorece conductas inapropiadas respecto a problemas sociales como por ejemplo el COVID o la inmigración. Existen empresas como [Maldita](#), que son expertos en detectar este tipo de noticias falsas con distintas herramientas entre las que se encuentra los clasificadores de textos.

Los problemas de clasificación pueden ser: binarios, en los que se construirá un modelo capaz de identificar si un documento pertenece o no a una clase; multiclase, en el que existen multitud de clases diferentes y hay que asignar una clase a cada documento; o multietiqueta, en el que cada documento puede pertenecer a más de una clase de forma simultánea.

Es importante reseñar la dificultad del proceso del etiquetado de los datos, que generalmente se realiza mediante un proceso manual con expertos en el campo de aplicación del modelo que se quiere generar. Para realizar correctamente el proceso hay que comprobar que las clases definidas son las correctas y además intentar que un subconjunto de documentos sea etiquetado al menos por dos personas diferentes y comprobar el grado de concordancia entre éstos, ya que nos permitirá ver la calidad de las clases definidas. Ese proceso de concordancia se conoce como **agreement**, y hay métricas de **Inter Annotation Agreement (IAA)** definidas con este objetivo. Afortunadamente, con la expansión del uso de la inteligencia artificial en el mundo empresarial, han ido apareciendo herramientas como [Prodigy](#), que permiten aligerar el proceso de anotación y obtener datasets válidos para entrenar modelos en el menor tiempo posible.

El proceso para entrenar un modelo de clasificación de textos sigue una estructura similar al proceso que se sigue para entrenar clasificadores de datos estructurados. El primer paso es el más simple, dividir nuestro corpus anotado en subconjuntos de training, development y testing. Podemos optar por una subdivisión del corpus tradicional o utilizar algoritmos de validación cruzada. A continuación, transformaríamos los textos en vectores de características que puedan ser introducidos en el clasificador. Por último, entrenaríamos el clasificador con esos vectores de características y buscaríamos ajustar los hiperparámetros para conseguir mejor rendimiento.

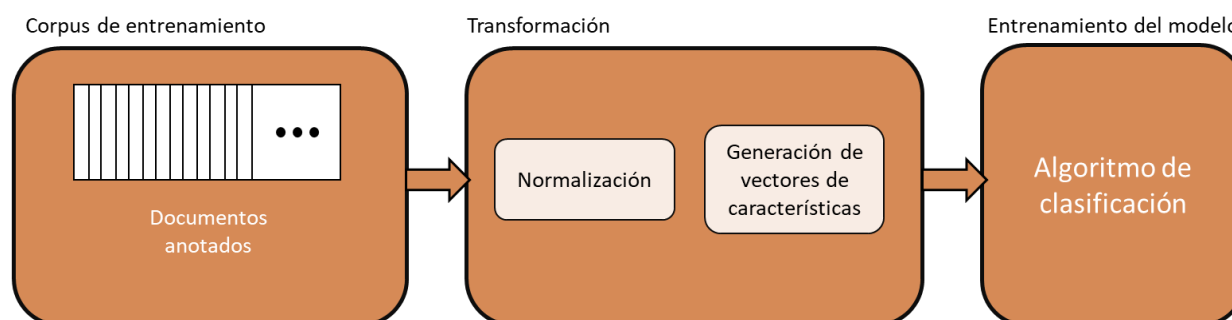


Figura 13. Flujo de procesos en el entrenamiento de un clasificador de textos.

### 2.2.1. Transformación

El proceso de transformación en un problema de clasificación está compuesto por las dos etapas definidas anteriormente: Normalización y generación de vectores de características.

- **Normalización:** En las tareas de clasificación de textos generalmente se aplican algoritmos de stemming y lematización para reducir la dimensionalidad de las palabras. En ocasiones se utilizan clasificadores cuyo coste computacional es muy elevado y reducir el número de características uniformizando palabras ayuda a obtener resultados en menos tiempo. También se aplica la eliminación de palabras vacías y de forma específica se pueden remplazar las contracciones cuando se trabaja en inglés [14].
- **Generación de vectores de carecterísticas:** El método más utilizado para generar los vectores de características es la técnica de TFIDF combinando unigramas y otros n-gramas. En algunas ocasiones es interesante incorporar otras características como número de adjetivos o sustantivos en el documento, especialmente si es un texto corto; y representación de las frases mediante embeddings.

Con la popularidad de las redes neuronales profundas se está imponiendo el uso de modelos de lenguaje como BERT, que no utilizaremos durante el curso para acelerar el proceso de obtención de características.

### 2.2.2. Algoritmo de clasificación

Una vez se ha conseguido una representación numérica de los documentos, comenzaríamos el proceso de entrenamiento utilizando un algoritmo de clasificación. Existen multitud de algoritmos para la clasificación de textos. Algunos de ellos son:

- **Clasificador Naïve Bayes:** Este modelo probabilístico es simple pero sorprendentemente efectivo para clasificar textos. Se basa en la regla de Bayes, y se denomina *naïve* porque asume la independencia de cada una de las variables en el modelo. Es decir, que la probabilidad de que una palabra vaya a continuación de otra es aleatoria, algo irreal por la correlación existente entre palabras en un lenguaje.
- **Modelo de regresión logística:** El modelo de regresión logística sirve para resolver problemas de clasificación binaria. El término regresión se refiere al modo que se tiene de calcular los parámetros del modelo logístico. Este tipo de clasificador utiliza una combinación de las características de entrada ponderadas en una función sigmoide para saber la probabilidad de pertenencia a una de las clases.
- **SVMs:** Las máquinas de Vector Soporte (SVMs) son unos algoritmos con buen funcionamiento para calcular las clases del texto. En los SVM se buscan los hiperplanos que mejor separen las características para cada una de las clases en un espacio n-dimensional. Tienen el inconveniente (gran inconveniente) de que los tiempos de entrenamientos de SVM suelen crecer cuadráticamente con el número de muestras, lo que los convierten en poco prácticos para usarlos de forma habitual, especialmente con grandes cantidades de datos.

Además de esos modelos, se pueden utilizar cualquier otro del ámbito de la ciencia de datos. Especialmente populares se han vuelto los modelos de boosting y las nuevas arquitecturas de Deep Learning como las Redes de Neuronas Recurrentes. Sin embargo, por el carácter introductorio de este módulo se ha decidido no tratarlas en estos apuntes.

### 2.2.3. Evaluación

Como ocurre en cualquier problema de clasificación, la evaluación del modelo es esencial y de hecho es una de las preguntas que hay que hacerse es, ¿cuándo consideramos que nuestro modelo es suficientemente bueno como para dejar de intentar mejorarlo?

En el campo de la clasificación de textos las medidas de evaluación no son absolutas, ya que dependen de la tarea de clasificación específica: no es lo mismo clasificar textos médicos que clasificar si una review es positiva o negativa. Lo más normal es buscar en la bibliografía los *baselines* para tareas similares y comparar con ellos si estamos consiguiendo resultados aceptables.

Tal y como ocurre con una tarea de clasificación tradicional, las métricas más utilizadas son:

- **Precisión:** Es la proporción de documentos clasificados correctamente sobre el total para los que el modelo ha predicho la clase c.

$$\text{precision} = \frac{\# \text{ true positives}}{\# \text{ true positives} + \# \text{ false positives}}$$

- **Recall:** Es la proporción de documentos correctamente clasificados entre todos los documentos del conjunto de entrenamiento con la clase c.

$$\text{recall} = \frac{\# \text{ true positives}}{\# \text{ true positives} + \# \text{ false negatives}}$$

- **F1-Score:** En términos generales un buen clasificador deberá tener un balance entre precisión y recall. Para ello existe la métrica F1-score, que considera en su cálculo ambos parámetros y penalizará el valor total si alguno de los dos es demasiado bajo.

$$F1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Además de lo anterior, para llegar a un clasificador de calidad nos tenemos que mover entre la línea del **overfitting** y el **underfitting**. Un modelo **underfitted** tiene baja varianza, que significa que siempre que se introduce el mismo dato a su entrada se obtiene la misma predicción, pero tiene un gran sesgo (bias), que se traduce en que su predicción está demasiado alejada de la realidad. Este fenómeno ocurre cuando no existen (o no se disponen) de suficientes datos de entrenamiento como para que el modelo sea capaz de encontrar los patrones existentes en los datos. Por otra parte, un modelo **overfitted** se puede ver como un sistema que tiene tan buena memoria que es capaz de predecir muy bien los datos que ya conoce, pero que sin embargo funciona mal con datos nuevos.

La manera de conseguir el punto óptimo de trabajo es evaluar el modelo con datos con los que no ha sido entrenado y nunca ha visto, por ese motivo en ocasiones es recomendable subdividir el corpus en múltiples subconjuntos para utilizar técnicas de validación cruzada (**cross-validation**). Estas técnicas consisten en ordenar los datos de forma aleatoria para posteriormente dividirlos en  $k$  subconjuntos. De forma que el entrenamiento se realice con  $\frac{k-1}{k}$  subconjuntos y la evaluación con el  $\frac{1}{k}$  restante. De este modo el modelo se entrenará  $k$  veces, con todos los datos y viendo siempre su funcionamiento cuando hay datos con los que no se ha entrenado. Generalmente se utiliza un valor de  $k$  de 10 o 12, aunque dependerá del volumen de datos disponibles.

La metodología de cross-validation es muy común con el ML tradicional, pero no lo es con técnicas computacionalmente más costosas como el Deep Learning, en los que se opta por dividir el corpus en los tres subconjuntos tradicionales para evitar el sobrecoste computacional de la validación cruzada.

## 2.3. Clustering

En el apartado anterior se han visto las nociones sobre cómo construir un clasificador a partir de un conjunto de datos con una serie de clases o etiquetas asignadas. Sin embargo, en un entorno real lo común es encontrar datos en bruto, en los que no ha habido un trabajo previo de anotación y que sin embargo es necesario analizar para extraer conocimiento de ellos. Cuando esto ocurre es necesario utilizar técnicas no supervisadas para el análisis como los algoritmos de **clustering** o **agrupamiento**.

La diferencia existente entre la agrupación y la clasificación de textos al principio puede ser difícil de entender, ya que en ambos casos lo que se busca es dividir un conjunto de documentos en grupos. Sin embargo, mientras que en la clasificación el objetivo es entrenar un sistema capaz de replicar una categorización de clases realizada por un humano previamente de forma manual, en el clustering se busca agrupar documentos similares a partir de la información intrínseca que poseen, por lo que se puede decir que se busca agrupar los documentos por su estructura natural.



En el clustering en muchas ocasiones se puede terminar generando grupos en los que no seamos capaces de identificar propiedades comunes de los documentos que los forman. Por ese motivo, uno de los mayores retos en este tipo de tarea de aprendizaje automático es la generación de grupos explicables que nos ayuden a comprender los datos y que permitan su reutilización en otros procesos de análisis textual.

El clustering es una herramienta muy útil en muchos campos de la industria y la ciencia. Sirve como herramienta para análisis exploratorio con el fin de:

- **Extraer conocimiento sobre el contenido de un corpus:** Mediante la agrupación de documentos podemos identificar los documentos típicos dentro del corpus y realizar análisis más exhaustivos sobre ellos. Por ejemplo, podemos agrupar quejas sobre un producto para posteriormente analizar más exhaustivamente el contenido de las quejas con otros métodos de Text Mining.
- **Relacionar documentos similares:** Es muy común contar con textos duplicados en los corpus. Las técnicas utilizadas en el clustering pueden ayudar a identificar estos elementos para extraerlos del corpus. De hecho, esto se utiliza para mejorar la eficiencia de los procesos de anotación: el tiempo de trabajo de un anotador es muy caro, por ese motivo es importante proporcionarle documentos representativos sobre la temática a anotar y lo más importante, eliminar los textos duplicados o muy similares entre sí.
- **Crear una estructura natural** de los textos para generar características para otras tareas, por ejemplo, la clasificación de textos.

Los algoritmos de clustering se originaron en el campo de la estadística y de Data Mining, donde son utilizados con conjuntos de datos numéricos. En el caso del Text Mining, al igual de lo ocurrido en el Data Mining generalista, se necesitan dos componentes principales para utilizar estos algoritmos: un método para comparar los documentos, una métrica para calcular la similitud entre ítems, y una manera de evaluar los resultados. Adicionalmente habrá que transformar los datos para que puedan ser utilizados por los algoritmos.

### 2.3.1. Transformación

En el clustering el proceso de transformación del texto es exactamente el mismo que el llevado a cabo en la clasificación:

- **Normalización:** Se suele aplicar stemming, eliminación de stopwords y la transformación a minúsculas de las palabras.
- **Generación de vectores de carecterísticas:** El método más utilizado para generar los vectores de características es la técnica de TFIDF utilizando exclusivamente unigramas. Aunque sea el método más común, es esencial probar diferentes combinaciones con otros n-gramas.

### 2.3.2. Algoritmos de clustering

Existen multitud de métodos para el clustering de documentos. En la agrupación de textos tradicionalmente los que más se utilizan son el **clustering jerárquico aglomerativo** (*Hierarchical Agglomerative Clustering*) y el **K-means**, un método de agrupación particional.

#### Clustering Jerárquico Aglomerativo:

Este tipo de algoritmo de clustering considera que al inicio del proceso existen tantos clusters como documentos existentes en el corpus. A partir de ese momento, se empiezan a comparar los grupos de forma iterativa con una **función de similitud** para agruparlos y generar grupos más numerosos. La idea es que en cada iteración se generen grupos con más documentos hasta que se termine el proceso con un único grupo final, que es el corpus completo. El resultado del clustering suele visualizarse a través de un *dendograma* en el que se puede visualizar la estructura de las agrupaciones generadas. A partir de ese dendograma podemos decidir el número de clusters que queremos utilizar, estableciendo un límite en las ramas existentes del gráfico.

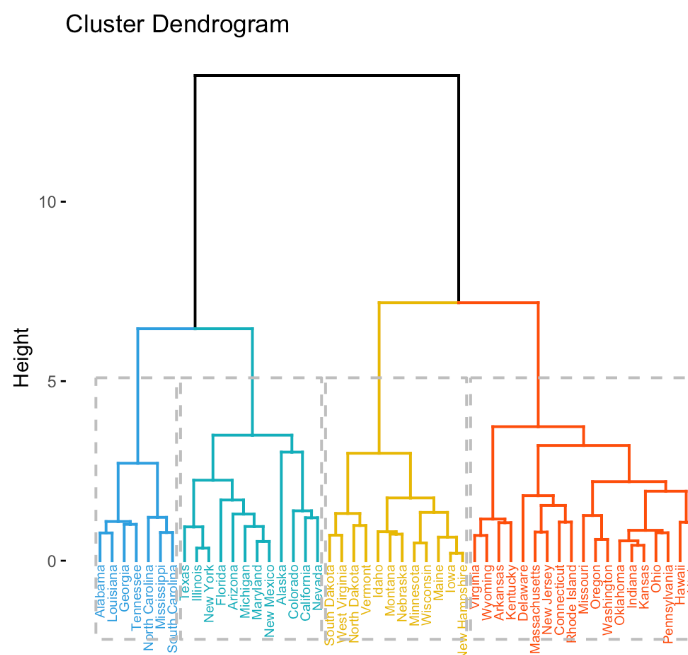


Figura 14. Ejemplo de Dendograma después de realizar cluster jerárquico sobre un conjunto de datos. Source: Datanovia

Pero, ¿cómo calcular la similitud entre los grupos que se van creando? Existen tres métodos distintos:

- **Single-Link:** El método del single link compara la similitud entre los elementos de cada grupo más cercanos entre ellos en el espacio vectorial. Este es el método más optimista desde el punto de vista de similitud, ya que desde este punto de vista los grupos estarán muy unidos, debido a que la similitud del grupo quedará definida a partir del par de datos más cercanos entre ellos, es decir que tendrán una mayor similitud. Es sensible frente a outliers, ya que la decisión de agrupación se toma por un único elemento, y proporciona clusters muy separados.

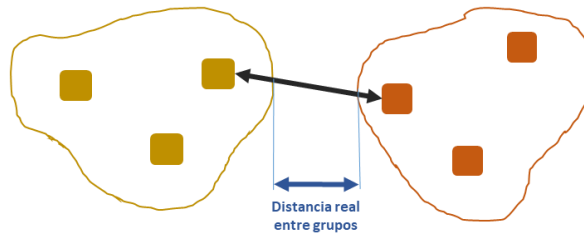


Figura 15. Representación gráfica del single-link.

- **Complete-Link:** Este método compara la similitud entre los documentos más alejados de cada grupo en el espacio vectorial. Este método es el más pesimista, al tomar la similitud del par más lejano como la similitud de los dos grupos nos aseguramos de que si los dos grupos tienen alta similitud serán muy parecidos. Es sensible frente a outliers, ya que la decisión de agrupación se toma por un único elemento, y proporciona clusters muy juntos.

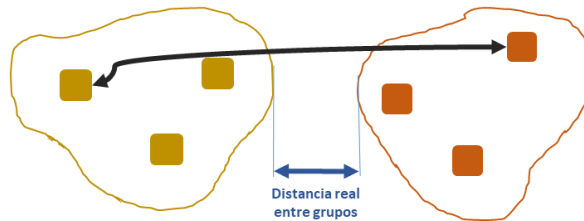


Figura 16. Representación gráfica del complete-link.

- **Average-Link:** Calcula el valor medio de cada uno de los grupos y los compara entre ellos. Es un método intermedio entre los anteriores. Es robusto frente a outliers, ya que la decisión de agrupar se toma mediante el conjunto de todos los elementos pertenecientes a un grupo.

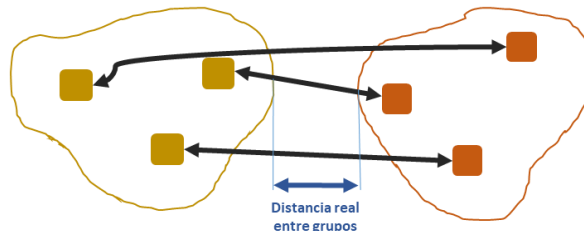


Figura 17. Representación gráfica del average-link.

Es importante reseñar que es recomendable probar los tres métodos cuando se utiliza este tipo de algoritmo ya que suelen proporcionar resultados distintos. Ninguno de los métodos es mejor que el otro, ya que los resultados dependerán de la distribución de los datos del dataset.

### *K-means clustering*

El método de **K-medias** tiene la peculiaridad de que es antes de comenzar el entrenamiento del modelo es necesario predefinir un número de  $k$  clusters. Una vez seleccionado, el proceso de asignar a los documentos un grupo es el siguiente:

1. Se seleccionan de forma aleatoria  $K$  **centroides** en el espacio vectorial. que corresponderán a cada uno de los clusters que se buscan.
2. Se recorre la lista de documentos, asignado a cada uno de ellos el cluster cuyo centroide tenga mayor similitud (cuya distancia en el espacio vectorial sea menor).
3. Una vez finalizado se re-calcula un nuevo valor de los centroides basados en los documentos que componen el grupo, y se realiza el mismo proceso hasta llegar a un punto de convergencia.

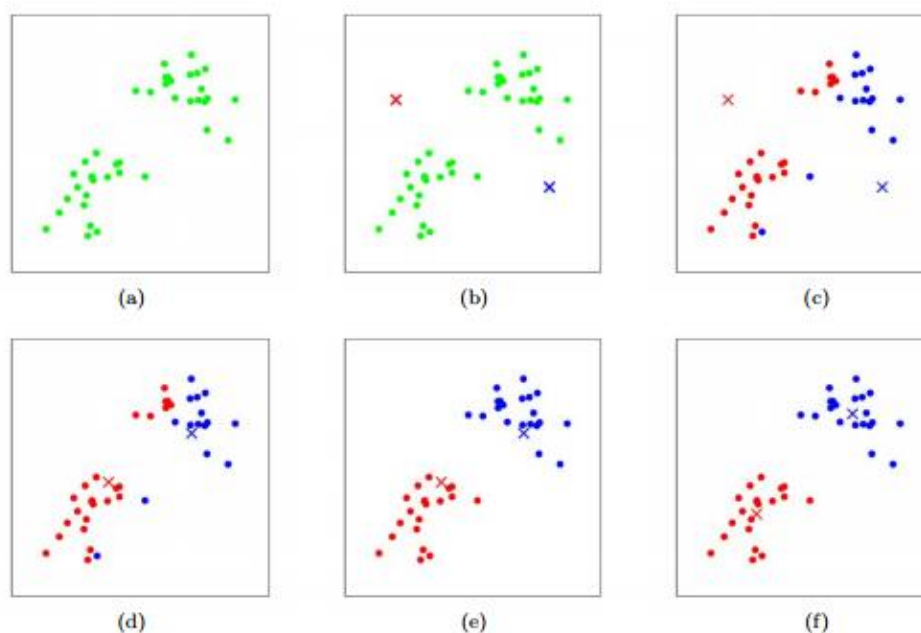


Figura 18. Representación gráfica del funcionamiento del algoritmo *k-means*. Source: [Stanford University](https://stanford.edu/)

### 2.3.3. Métricas de similitud

Ya se han visto los algoritmos tradicionales de clustering en Text Mining. Se ha visto que ambos basan todas sus decisiones en la similitud existente en los documentos utilizados pero, ¿cómo se puede medir la similitud?

Una forma de cuantificar la similitud entre documentos es medir la distancia que existe entre la representación vectorial de los documentos en el espacio  $n$ -dimensional. Estas métricas permiten conocer el grado de cercanía o de separación entre los documentos, que se relaciona con que las características que los componen son los más o menos similares. Estas características dependen de los datos o del problema, por lo que no es posible definir una métrica de similitud que funcione bien en todos los casos.

Algunas de las métricas más utilizadas son:

- **Distancia Euclídea:** Este tipo de métrica es la utilizada de forma generalizada en problemas geométricos. La distancia euclídea entre dos puntos en un espacio cartesiano es la longitud de la recta que une dichos puntos. Esta métrica se utiliza de forma muy extendida en problemas de agrupación, incluyendo la agrupación de textos, y en muchos casos es la métrica utilizada por defecto en los algoritmos de clustering. La representación matemática es:

$$d(P1, P2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

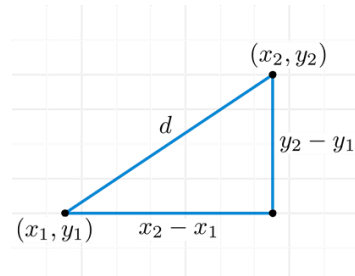


Figura 19. Representación de la distancia euclídea. Source: Wikipedia

- **Cosine Similarity:** Cuando representamos documentos como vectores, la similitud de dos documentos se puede interpretar como el coseno del ángulo que forman ambos vectores respecto al origen de coordenadas. El valor de esta métrica puede oscilar entre -1 y 1, siendo 1 cuando los dos documentos son iguales. Este método es utilizado en aplicaciones de recuperación de información, pero también en clustering.

$$s(I_1, I_2) = \cos(\theta) = \frac{I_1 \cdot I_2}{\|I_1\| \|I_2\|}$$

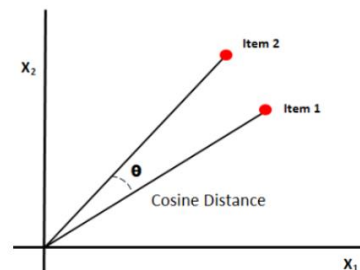


Figura 20. Representación gráfica del cosine similarity Source: O'Really.

- **Coefficiente de Jaccard:** Este coeficiente mide la similitud como la intersección de los elementos de dos documentos divididos por su unión. Esta métrica es común para la similitud de textos, igual que la similitud coseno, pero es mucho más costosa computacionalmente.

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

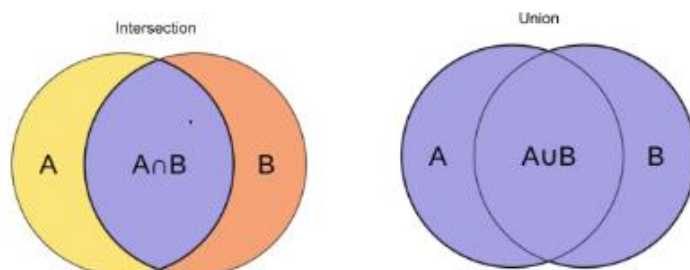


Figura 21. Representación gráfica del cosine similarity Source: O'Really.

#### 2.3.4. Validación de agrupaciones

El proceso de validar el resultado de un algoritmo de clustering no es trivial. Al final del proceso se obtendrán grupos similares. Pero hay que tener en cuenta que son similares desde un punto de vista de distancias en un espacio vectorial, pero ¿lo son para una perspectiva humana? El Data Scientist debe definir de forma clara cuál es su idea de similitud, algo conocido como **clustering bias**. Esta perspectiva es esencial para evaluar un algoritmo de clustering. Por ejemplo, observando la Figura 22 ¿cuál de las dos agrupaciones es mejor? La respuesta dependerá de nuestra consideración de similitud, si buscamos similitud en base a las formas de los objetos será mejor el resultado 1, si es en base al color será mejor el resultado 2.

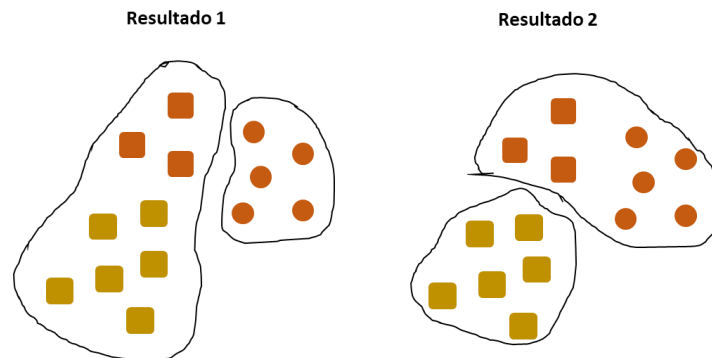


Figura 21. Ejemplos de resultados del clustering en el mismo dataset.

A pesar de esa consideración subjetiva de evaluación, también se pueden considerar dos formas de evaluar los clusters de documentos:

- **Forma directa:** Con esta forma buscamos comparar los resultados del clustering automático con los grupos ideales. Para ello la evaluación constaría de 4 pasos:
  - Proporcionar el conjunto de datos de test a un anotador para que cree un resultado de clustering ideal, conocido como *gold standard*, que nos servirá como referencia
  - Utilizar el algoritmo para producir resultados con el mismo test set
  - Cuantificar la similitud entre los clusters generados por el sistema y el gold standard. Esa similitud puede ser medida desde múltiples perspectivas: F-measure, información mutua normalizada, pureza...
- **Forma indirecta:** Con esta forma se busca evaluar lo útiles que son los resultados del clustering para una aplicación concreta. En este caso el sesgo, clustering bias, está impuesto por la aplicación final del sistema. El proceso de evaluación consistiría en:
  - Crear un conjunto de test con documentos representativos de la aplicación del sistema final para cuantificar el rendimiento.
  - Identificar el sistema *baseline* de la aplicación en la que queremos incorporar el clustering. Una vez identificado aplicar nuestros datos a dicho sistema.

- A continuación, incorporaríamos nuestro de algoritmo de clustering al sistema anterior y lo aplicaríamos a nuestros datos.
- Por último, comprobaríamos si nuestro sistema obtiene mejores resultados que el *baseline*.

Esta manera se llama indirecta porque no se comprueba directamente la calidad de los clusters, sino como el clustering contribuye a mejorar un sistema específico.

## 2.4. Topic modeling

Diariamente se generan medio millón de textos cortos en la red social Twitter, supongamos que los hubiéramos conseguido almacenar para su análisis. ¿Cómo averiguaríamos cuales son los temas más populares a partir de los tweets de ese día? Existen procedimientos *naïve* para esto, como analizar la frecuencia de uso de las palabras en el conjunto de tweets. Sin embargo, existen técnicas más sofisticadas para conocer los temas tratados en documentos textuales conocidas como las técnicas de **Topic Modeling**, que como se ha dicho buscan identificar, sin ayuda de recursos externos, los temas principales que encierran un conjunto de textos (o un texto muy largo).

El topic modeling es un tipo de estrategia estadística que permite conocer los temas que se tratan en un conjunto relativamente grande de documentos de forma no supervisada. A diferencia de las técnicas de clustering que agrupan un corpus de documentos en grupos similares, en el topic modeling se busca encontrar y extraer la temática de los textos que se están analizando.

La representación de la temática de un documento refleja la información sobre su contenido. Por lo que un documento podría ser descompuesto en el conjunto de los temas que lo componen. Esto es útil para muchas aplicaciones como la recuperación de la información, la clasificación de textos y el resumen de documentos. Pero, ¿cuál es la definición formal de *topic*?

***“Un topic o tema es un conjunto de palabras que es probable que aparezcan en el mismo contexto”***

El topic modeling busca a analizar las estructuras del texto para determinar que palabras son las que más probabilidad tienen de aparecer dentro de un corpus y contrariamente a lo que pudiera parecer, no depende tanto de la sintaxis de los textos, sino de la semántica de estos. Existen múltiples métodos para realizar topic modeling, siendo el principal de ellos el **Latent Dirichlet Allocation (LDA)**, existen otros con mejor rendimiento como el **Dirichlet Multinomial Regression (DMR)** que no se tratarán en este módulo.

Para entender cuál sería el resultado de aplicar un algoritmo de este tipo a un conjunto de textos. Consideremos un conjunto de textos de ejemplo como el siguiente:

**Doc1:** *A María le encantan los animales. Disfruta mucho paseando a sus perros y montando a caballo. En ocasiones, cuida de los gatos de sus amigos. Suele alimentarles con las sobras de sus comidas siempre que sean proteínas como pollo y ternera. Nunca les da pescado.*

**Doc2:** *Él es experto gastronómico para la Guía Michelin. Ha asistido a los mejores restaurantes del mundo y ha probado la mejor carne de ternera y el mejor sushi del mundo en Japón*

**Doc3:** *El perro de Juan murió hace 6 meses. Él todavía no lo ha superado, por eso está planteando adoptar un gato y comprar un conejo para no sentirse tan solo.*

Un algoritmo como LDA nos devolvería un conjunto de **topics**, compuestos por un conjunto de palabras que desde un punto de vista teórico serían capaz de reconstruir semánticamente cada uno de los textos del corpus.

**Topic 1:** [perros, caballo, gatos, conejo]

**Topic 2:** [pollo, ternera, pescado, sushi]

A partir de esos topics, se podría obtener la composición de los textos según esos temas como por ejemplo:

**Doc1:** Topic 1 (70%) y Topic 2 (30%)

**Doc2:** Topic 2 (100%)

**Doc3:** Topic 1 (100%)

#### 2.4.1. Transformación y entrenamiento

Cuando se aplica **LDA** a textos, al algoritmo hay que proporcionarle los textos transformados con el método Bag of Words y el número de temas que queremos que el algoritmo trate de extraer de estos. En ocasiones es recomendable utilizar sólo términos de una categoría gramatical específica como nombres y/o adjetivos.

#### 2.4.2. Latent Dirichlet Allocation

En este módulo nos centraremos en el LDA. El **método de Latent Dirichlet Allocation** fue introducido por David Blei, Andrew Ng y Michael Jordan en el año 2003. Es un algoritmo probabilístico en el que los temas de un texto se representan como la probabilidad de que un conjunto de términos aparezca en estos.

El algoritmo asume que los documentos tienen una estructura semántica implícita (*los topics*) que se puede inferir a través de las concurrencias de palabra-documento. Se puede decir que parte de la idea de que las palabras están relacionadas con los topics, y los topics con los documentos. De hecho, en LDA los documentos se consideran como un conjunto de topics.

La característica principal de estos modelos es que no se requiere que los temas sean exclusivos, es decir que ciertas palabras pueden aparecer en múltiples temas, permitiendo de este modo obtener temas más flexibles que se ajustan a la diversidad del lenguaje.

#### 2.4.3. Evaluación

Debido a la naturaleza no supervisada del topic modeling, existe cierto debate sobre como evaluar el resultado de los algoritmos. De hecho, dado que al algoritmo hay que proporcionarle el número de temas a calcular, ¿cómo saber si el número elegido es el correcto?



Tal y como ocurría con la evaluación de los algoritmos de clustering se puede realizar validaciones de carácter cualitativo mediante herramientas de visualización como *pyLDavis*<sup>4</sup>, que ayuda a interpretar los temas generados con el algoritmo LDA.

También se pueden utilizar métricas como la perplejidad y la coherencia de los topics:

- La **perplejidad** es una medida de evaluación usada comúnmente en los modelos lingüísticos que intenta medir como el modelo reacciona con datos que no había visto con anterioridad y se realiza con un subconjunto de datos de test. Generalmente se ha dicho que cuando un modelo LDA presentaba baja perplejidad era de mayor calidad. Sin embargo, investigaciones recientes han demostrado que a menudo esta métrica esta anti correlacionada con la valoración humana de los resultados. Es decir, que aunque el modelo consiguiese una baja perplejidad, los topics eran no interpretables por un humano [15].
- Las métricas de **coherencia de los topics** permiten evaluar el grado de similitud semántica entre las palabras de un mismo tema. Si los temas tienen alta similitud el grado de coherencia será mayor y podremos considerar que el resultado es correcto. Para medir la coherencia se suele utilizar la **métrica  $C_v$** . No vamos a entrar a explicar su proceso de cálculo, pero se puede decir que un mayor valor de esa métrica se traduce en una mayor coherencia y por lo tanto los topics serán de mejor calidad. Por ejemplo, en la Figura 23 se observa que el modelo con mejores resultados desde el punto de vista de similitud terminológica en los temas se consigue cuando se consideran 8 y 9 en el proceso de entrenamiento del modelo.

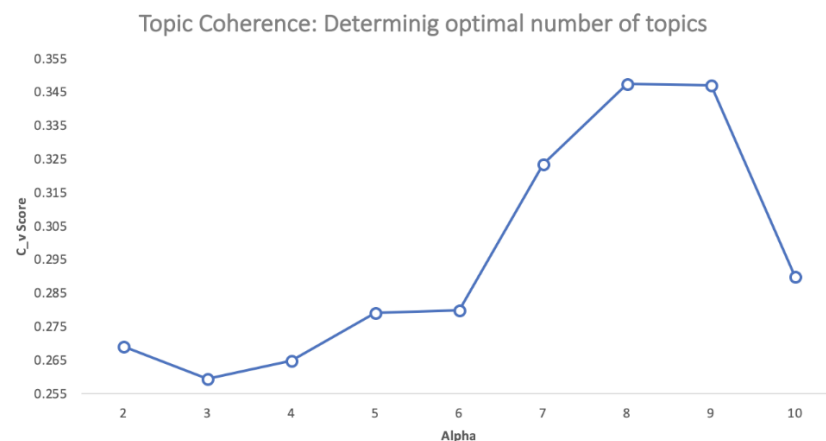


Figura 22. Gráfica de medida de coherencia para distinto número de topics en LDA. Source: Medium

<sup>4</sup> <https://github.com/bmabey/pyLDavis>

### 3. Caso de estudio: Análisis de sentimiento de Redes Sociales

Como se comentó en el capítulo de la introducción, en los últimos años, con la entrada de la sociedad en la Era de la Información se ha experimentado una transición de Internet como una herramienta para consumir información al Internet social. El Internet social ha generado cambios en los comportamientos de las personas, ya que ahora pueden interactuar entre ellas mediante plataformas digitales entre las que se encuentran los canales de ventas digitales o los medios sociales.

Específicamente, las **Redes Sociales Online (RSO, Online Social Networks)** se han convertido en las plataformas preferidas para llevar a cabo esas interacciones. Los usuarios de RSO observan, analizan, crean y difunden información sobre sus opiniones e impresiones respecto a temas diversos como política, deportes, música o productos. Esta información es de gran utilidad para las empresas, ya que pueden extraer información fidedigna de sus clientes actuales y futuros, pero al ser información desestructurada en forma de texto es necesario utilizar técnicas de Text Mining para analizar los datos.

Dentro del Text Mining existe un subcampo específico con este fin: el campo del **análisis del sentimiento o opinion mining**. Las herramientas aglutinadas en este campo se utilizan para detectar, extraer y analizar la opinión expresada por el autor de un texto. En este campo el concepto de opinión aglutina otros como los sentimientos, valoraciones, actitudes y emociones que un sujeto presenta hacia una persona, objeto, institución o evento.

El campo del análisis del sentimiento ha crecido exponencialmente en los últimos años y se ha convertido en uno de los campos más activos de investigación en áreas como el NLP y el Machine Learning. Su desarrollo hace años que escapó del cerco de la informática y se ha incorporado como herramienta esencial en otras ramas del saber cómo las ciencias sociales. Además, gracias a que es imprescindible conocer la opinión de la gente para tomar decisiones empresariales basadas en datos y que el negocio prospere, estas herramientas han sido adoptadas por empresas para mejorar su eficiencia, impacto y calidad de sus bienes y servicios.

La última sección de estos apuntes estará centrada en las técnicas de análisis de sentimiento. En primer lugar, se hablará sobre las aplicaciones del opinion mining en entornos industriales y sus futuros usos gracias a los últimos avances en ramas de investigación aplicada. A continuación, definiremos algunos aspectos básicos sobre el análisis de sentimiento, para terminar con una guía que explique cómo captar la subjetividad y el sentimiento de los textos.

#### 3.1. Aplicaciones de negocio del opinion mining.

Una de las características de los seres humanos es su capacidad de socialización. El hecho de vivir en sociedad hace que nos relacionemos con personas que tienen diferente forma de ser, ideología o principios. Esta vida en comunidad hace que nuestras personalidades se moldeen y tengamos opiniones sobre diferentes aspectos que nos rodean, entre las que se incluyen partidos políticos, empresas o productos.

La opinión es un aspecto primordial en la sociedad de hoy en día: nos gusta saber lo que piensa la gente sobre productos tecnológicos antes de comprarlos o conocer la opinión política de nuestro círculo cercano para tomar decisiones a la hora de votar. En el pasado obteníamos esta opinión mediante círculos sociales cercanos como amigos y familiares, hoy en día recurrimos a Internet.

La presencia de opiniones en Internet es algo que no ha pasado desapercibido para las empresas. Tradicionalmente las compañías realizaban encuestas para conocer la percepción del público sobre su imagen corporativa y sus productos. Hoy en día gracias a la explosión de los medios sociales complementan esta información analizando el contenido textual presente en Internet. Por ejemplo, las grandes corporaciones de comida rápida utilizan datos de RSO para conocer la opinión de la gente sobre sus nuevos productos y cuál es su popularidad respecto a los de la competencia [16]. También se ha utilizado en otros sectores como el de la automoción para conocer la percepción de los clientes sobre una marca específica en cada uno de los mercados internacionales en los que operan para mejorar sus campañas de marketing [17].

Aunque esos son solo algunos ejemplos, el análisis de sentimiento se ha expandido a casi todos los dominios, con el fin de conocer la opinión de la gente sobre productos, servicios, salud, finanzas, eventos sociales o política. Empresas como Microsoft, Google y SAP han construido sus propios sistemas para realizar análisis de opinión sobre datos textuales, pero también han surgido start-ups, como la española *Graphtext*, que centran su modelo de negocio a la información existente en redes sociales y a extraer conocimiento tanto de la estructura de la red como del contenido que comparte la gente que la compone.

En el futuro su aplicación se extenderá a muchos ámbitos de la vida real tal y como muestran las investigaciones que se están realizando. En los últimos años se han realizado estudios de investigación en redes sociales en los que se utilizan técnicas de análisis de sentimiento para mejorar el conocimiento de los efectos secundarios de medicamentos [18] y para mejorar los sistemas de vigilancia pública de drogodependencia [19]. También se han propuesto sistemas para predecir las ventas de productos a partir del sentimiento de los clientes [20], o incluso para establecer relaciones entre las apuestas deportivas y las opiniones del público mostradas en twitter y blogs [21].

También en investigación, recientemente se ha vuelto popular el uso de la extracción de las opiniones para mejorar los sistemas de gestión ambiental de ciudades ideando sistemas capaces de pronosticar los índices de contaminación atmosférica a partir de la actividad en redes sociales [22] o predecir la molestia acústica de grandes eventos a partir de datos similares [23]. Además, también ha sido utilizado para monitorizar los sistemas de gestión en situaciones de emergencia asociadas a desastres naturales como terremotos e inundaciones [24]–[26].

Por último, las últimas investigaciones se han centrado en el uso del análisis de sentimiento para conocer la opinión sobre la pandemia de la COVID19 expresada por la gente en redes sociales e incluso saber los efectos del confinamiento sobre su salud física y mental[27].

### 3.2. Opinion mining y niveles de análisis

La popularidad del análisis de sentimiento no debe engañarnos y hacernos pensar que es una tarea sencilla. La dificultad del análisis del sentimiento dependerá del nivel de detalle que queramos extraer de los textos y de la fuente de estos. Por ejemplo, cuando se analizan textos provenientes de medios sociales hay que utilizar técnicas específicas para detectar el sarcasmo, una de las tareas más complicadas del NLP, o detectar las opiniones generadas de forma irregular comúnmente conocidas como spam. Además, en esta rama se mantienen las problemáticas existentes en el resto del Text Mining como la detección de la negación o las comparaciones.

### 3.2.1. Niveles de análisis

Bing Liu, uno de los investigadores más influyentes en el campo del análisis el sentimiento, especifica que desde un punto de vista general existen tres niveles de granularidad en la extracción de opiniones sobre un texto [28]:

- **Nivel de documento:** El objetivo del análisis e sentimiento a este nivel consiste en determinar cuándo un documento completo expresa una opinión positiva o negativa. Por ejemplo, en el caso de que se analizaran *reviews* de una plataforma de venta de productos como Amazon, habría que categorizar que documentos expresan de forma global una opinión positiva o negativa asumiendo que solo se habla sobre un único producto o una característica de este.
- **Nivel de frase:** El análisis de sentimiento a nivel de frase analiza las oracuibes individuales de un documento y las clasifica en positivas, negativas o neutrales. Para calcular el sentimiento del documento final se realizaría una operación aritmética para decidir si el documento en su cómputo global es positivo, negativo o neutral.
- **Nivel de entidad:** Este nivel es conocido en inglés como *aspect-based sentiment analysis*. Cuando se analiza el texto con este nivel de agregación se buscar conocer el sentimiento sobre aspectos específicos de un producto. Por ejemplo, se podría evaluar la opinión sobre el precio, el peso y el rendimiento de una review de una cámara de fotos.

### 3.2.2. Dicionarios de sentimiento

En las frases que expresan opiniones existen ciertas palabras que ayudan a determinar si esa opinión es positiva o negativa. Esas palabras, que suelen ser adjetivos y adverbios, se llaman **partículas de sentimiento**. Algunos ejemplos de *partículas de sentimiento* positivas son “bien”, “bonito”, “agradable”, “feliz”; algunos ejemplos de partículas negativas son sus antónimos “mal”, “feo”, “desagradable”, “triste”.

Existen diccionarios contruidos por grupos de investigación que agrupan las palabras de sentimiento según sean positivas, negativas o neutras, e incluso incorporan una serie de emociones a esas palabras como furia, sorpresa o confianza. Estos diccionarios generalmente se construyen mediante un proceso de anotación manual siendo algunos de los más populares los de SentiWordNet<sup>5</sup> [29], WordNet-Affect<sup>6</sup>[30], los generados por Bing Liu<sup>7</sup> o los generados en las shared-task EmoLex<sup>8</sup> . También existen diccionarios específicos para trabajar con datos de redes sociales. Los datos de redes sociales como Twitter contienen emojis que pueden ayudar a entender el sentimiento de la frase, por eso se desarrolló el lexicón *Emoji Sentiment Ranking*<sup>9</sup> [31], que asocia un sentimiento de positividad, negatividad y neutralidad a cada emoji existente.

<sup>5</sup> <https://github.com/aesuli/SentiWordNet>

<sup>6</sup> <https://wndomains.fbk.eu/wnaffect.html>

<sup>7</sup> <http://www.cs.uic.edu/~liub/FBS/opinion-lexicon-English.rar>

<sup>8</sup> <https://saifmohammad.com/WebPages/lexicons.html>

<sup>9</sup> [http://kt.ijs.si/data/Emoji\\_sentiment\\_ranking/](http://kt.ijs.si/data/Emoji_sentiment_ranking/)




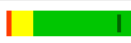







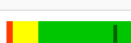


Char	Image [twemoji]	Unicode codepoint	Occurrences [5...max]	Position [0...1]	Neg [0...1]	Neut [0...1]	Pos [0...1]	Sentiment score [-1...+1]	Sentiment bar (c.i. 95%)
😂		0x1f602	14622	0.805	0.247	0.285	0.468	0.221	
❤️		0x2764	8050	0.747	0.044	0.166	0.790	0.746	
♥️		0x2665	7144	0.754	0.035	0.272	0.693	0.657	
😍		0x1f60d	6359	0.765	0.052	0.219	0.729	0.678	
😭		0x1f62d	5526	0.803	0.436	0.220	0.343	-0.093	
😘		0x1f618	3648	0.854	0.053	0.193	0.754	0.701	
😊		0x1f60a	3186	0.813	0.060	0.237	0.704	0.644	

Figura 23. Ejemplo del sentimiento de emojis del Emoji Sentiment Ranking. Source: [Emoji Sentiment Ranking](#)

### 3.3. Análisis de sentimientos de documentos cortos.

En esta sección trataremos la forma más básica del análisis de sentimiento, que es la categorización de documentos según expresen opiniones negativas o positivas. Como se ha comentado con anterioridad, el análisis de sentimientos de un texto completo se puede ver como un problema de clasificación binaria. Al enfrentarse al problema del análisis del sentimiento como una clasificación textual estamos asumiendo que el autor trata un único tema en el documento.

Generalmente la clasificación del sentimiento se aplica a documentos de tipo *review* de productos online y con documentos de redes sociales de microblogging. Como cualquier proceso de clasificación de textos, es necesario adquirir y preparar el corpus, transformarlo a características numéricas y por último entrenar y validar el modelo de clasificación.

En este caso se explicará el proceso de clasificación de textos cortos provenientes de una red social de microblogging como Twitter. En Twitter, los usuarios se comunican mediante textos cortos expresando su opinión sobre diversidad de temas como política, deportes o sociedad. Además, tienden a expresar su opinión de forma directa por la limitación de caracteres de cada entrada. Esto es algo positivo ya que podemos enfrentarnos al problema de la extracción del sentimiento como un problema de clasificación, bajo la presunción de que en un documento corto no se puede elaborar una opinión sobre más de un tema. Sin embargo, la limitación de caracteres genera retos en la clasificación, especialmente a nivel de preprocesado ya que los usuarios tienden a utilizar jerga y diminutivos que requieren algunos pasos de normalización antes de realizar cualquier análisis [32], [33].

### 3.3.1. Adquisición y preparación de los datos

El proceso de adquisición de datos en Twitter se puede realizar mediante su **API (Application Programming Interface)**. Al utilizar un script en un servidor que haga llamadas a este servicio web se pueden obtener documentos que contengan palabras clave específicas.

Cuando se trabaja con Twitter hay que tener en cuenta que existen distintos tipos de elementos textuales dentro de los documentos:

- **Hastags:** Es el nombre asignado a las palabras que los usuarios utilizan para etiquetar los tweets en una categoría específica. Estas etiquetas se identifican por tener el símbolo “#” delante de la palabra en cuestión.
- **Menciones y respuestas:** Los usuarios pueden interactuar entre ellos mediante menciones entre sí. Estas menciones se detectan porque se utiliza el símbolo “@” delante del nombre del usuario al que se quiere hacer referencia.
- **Emoticonos:** Los emoticonos o emojis son combinaciones de caracteres que los usuarios utilizan para expresar emociones.
- **Retweets:** Es el nombre dado al contenido re-publicado por un usuario que ha sido escrito por otra persona.

Al adquirir los datos de la API, es común desechar los retweets, porque muestran la opinión de otra persona sobre un tema específico y el retweet no tiene por qué alinearse al 100% con la opinión de la persona. Además, especialmente cuando se utilizan técnicas tradicionales, es mejor no utilizar las respuestas a otros tweets: como se ha dicho, los tweets tienen una longitud muy limitada, cuando se responde a un tweet se hace referencia a una información no conocida por el clasificador, algo que podría decrementar el rendimiento global del sistema de aprendizaje automático [34]

### 3.3.2. Transformación

Una vez extraídos y seleccionados los tweets será necesario etiquetarlos en las categorías positivo o negativo en función del sentimiento que expresan. Este proceso tiene las peculiaridades explicadas en el capítulo 2.

#### Normalización

Los textos provenientes de Twitter tienen un estilo de escritura compleja en la que abundan la jerga de internet y la presencia de errores ortográficos. Se pueden utilizar distintos flujos de normalización de los datos, pero en este caso se propone el que se muestra en la Figura 25.

En ese flujo de preprocesado el primer paso es la tokenización de tweets. Se pueden utilizar sistemas de tokenización generalistas u otros específicos para Twitter, como el creado por la *Carnegie Mellon University*<sup>10</sup> [35]. Lo bueno de los tokenizadores específicos es que son capaces de trabajar con la sintaxis de Twitter y son capaces de detectar y tokenizar correctamente los emojis, hastags y URLs..

<sup>10</sup> <http://www.cs.cmu.edu/~ark/TweetNLP/>

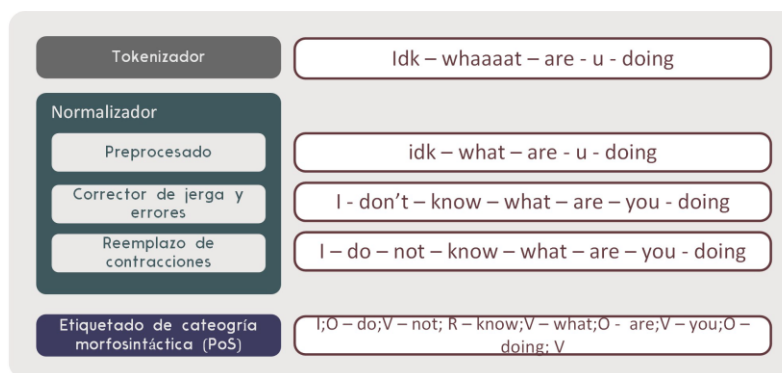


Figura 24. Flujo de normalización de texto aplicado a tweets. Source: [Luis Gascó Doctoral Dissertation](#)

Después de la tokenización, es necesario llevar a cabo un proceso de normalización de los textos:

1. **Preprocesado:** Es una buena práctica transformar los tokens a minúsculas y eliminar los caracteres repetidos de las palabras para que puedan ser detectados por los diccionarios de sentimiento ("bieeeeen" debería ser transformado a "bien"). A diferencia de otros campos, no se ha demostrado que la lematización ayude a mejorar el rendimiento de los clasificadores de sentimiento por lo que no es necesario aplicarla.
2. **Corregir jerga y errores:** Existen diccionarios de palabras de jerga utilizada en Twitter con su versión extendida. Es útil reemplazar palabras como "LOL" por "Lot of Laughs" para incrementar la longitud de los tweets y extender la información que contienen.
3. **Reemplazar contracciones:** Como se ha dicho antes el reemplazo de las contracciones y negaciones en inglés incrementa la calidad de los clasificadores, por lo que es recomendable aplicarla [36].

Por último, se podrían extraer las categorías morfosintácticas de los tweets (PoS) ya que pueden ser utilizadas posteriormente para extraer características adicionales que permitan mejorar la calidad del clasificador de sentimientos.

### Extracción de características

A diferencia de los clasificadores de documentos estándar, en los que en multitud de ocasiones no es necesario extraer más características más allá de n-gramas con ponderación TFIDF. En la clasificación de sentimientos de Twitter es necesario incorporar más características artificiales para proveer de más información a los clasificadores para funcionar.

Los tweets tienen una longitud muy limitada, por lo que extender información sobre ellos de forma artificial es una buena práctica. Algunas de esas características son:

- **Generar características a partir de PoS:** Se pueden calcular el número y el porcentaje de cada PoS en cada documento. Por ejemplo, el número de sustantivos, adjetivos y emoticonos podría ayudar al clasificador a identificar patrones para categorizar los documentos correctamente.

- **Características de sentimientos:** Es muy recomendable calcular características relacionadas con el sentimiento a partir de diccionarios de sentimiento como *SentiWordNet* y *Emoji Sentiment Ranking*.
- **Glove Embeddings:** En el capítulo introductorio se explicó que un *Word embedding* es un modelo que representa una palabra en un espacio vectorial con multitud de dimensiones en el que las relaciones entre palabras pueden ser encontradas con mayor facilidad. Existen embeddings contruidos con tweets, como por ejemplo el de la Universidad de Stanford, que fue construido utilizando 2000 millones de Tweets [37]. Se pueden utilizar estos modelos preentrenados para buscar los vectores de los sustantivos y adjetivos de los documentos y calcular su valor medio, la suma y la desviación típica para obtener una representación matemática multidimensional del tweet y que el clasificador pueda identificar en ese espacio las diferencias entre los textos positivos y negativos.

### 3.3.2. Entrenamiento y validación

Dado que es un problema de clasificación de texto, cualquier método de machine learning supervisado podrá ser aplicado a un dataset de tweets. Una vez evaluado el sistema y generado el modelo, podríamos aplicarlo a nuevos datos y extraer más información como por ejemplo extraer los temas de los que hablan los tweets positivos y negativos mediante técnicas de topic modeling, generar visualizaciones como *wordclouds*<sup>11</sup> o mostrar la evolución de tweets positivos y negativos a lo largo del tiempo.



Figura 25. Visualización Wordcloud realizada con la librería wordcloud de Python.

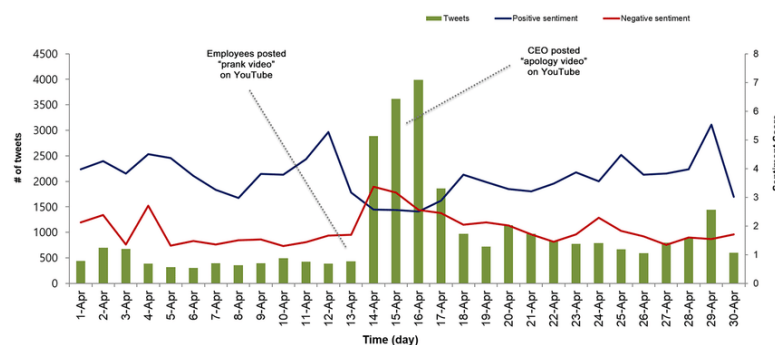


Figura 26. Ejemplo de evolución temporal del sentimiento de comentarios de Youtube. Source: [This paper](#)

<sup>11</sup> [https://amueller.github.io/word\\_cloud/](https://amueller.github.io/word_cloud/)



## 4. Futuro y lecturas recomendadas

En este módulo se han tratado los conceptos básicos y las técnicas principales del Text Mining. Los algoritmos utilizados no son los que mejor resultados ofrecen en la actualidad, pero son la base para empezar a utilizar sistemas más sofisticados.

En los últimos años ha habido una revolución en las técnicas utilizadas para el análisis de texto. Se han pasado de utilizar herramientas de Machine Learning tradicional a sistemas mucho más complejos basado en redes neuronales profundas, conocidas comúnmente como **Deep Learning**. Aunque los resultados con arquitecturas de Deep Learning son en general significativamente mejores que los obtenidos con técnicas tradicionales, se necesita una capacidad computacional muy alta para poder obtener los resultados en un tiempo asequible. De todos modos, con el abaratamiento de los centros de cálculo y de las tarjetas **GPUs** cada vez es más común utilizar estas tecnologías en ámbitos complejos como la traducción automática (DeepL) o la extracción de información de textos médicos.

En el Deep Learning aplicado a texto hay que destacar la importancia de las arquitecturas de **redes neuronales recurrentes (RNN)**. Debido a los problemas de convergencia que presentaban se diseñaron las arquitecturas con unidades **LSTM (Long Short-Term Memory)**, ya que permitían incorporar una mayor cantidad de contexto cuando trabajaban sin tener problemas de convergencia.

En los últimos años han aparecido tecnologías que intentan modelar el lenguaje a partir de modelos entrenados con millones de documentos y parámetros. Entre estos modelos destacan **GPT-2**, **GPT-3** y los modelos basados en sistemas de atención como **BERT (Bidirectional Encoder Representations from Transformers)**, siendo este último el más popular en la actualidad y que de hecho es utilizado por el motor de búsqueda de Google para comprender mejor la información que necesitamos.

Debido al carácter introductorio del módulo no se han podido ver con profundidad cada una de las ramas del Text Mining, pero no deberíais tener problema para empezar a enfrentaros a problemas de este tipo y poder ampliar información de forma autónoma entendiendo sin tanta dificultad la bibliografía. La rápida actualización de las técnicas de procesamiento de lenguaje hace **esencial** la actualización constante. Por ese motivo, a continuación recomiendo una bibliografía básica sobre el NLP y el Text Mining, y recomiendo que si queréis estar al día leáis blogs sobre la temática y consultéis las últimas publicaciones científicas:

1. **Foundations of Statistical Natural Language Processing** de Christopher Manning y Hinrich Schütze
2. **Text Mining: Predictive methods for Analyzing Unstructured Information** de Sholom M. Weiss et al.
3. **The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data** de Feldman, R y James Sanger, J.
4. **Neural Networks Methods in Natural Language Processing** de Yoav Goldberg y Graeme Hirst.
5. **Sentiment Analysis and Opinion Mining** de Bing Liu
6. **Practical Text Mining and Statistical Analysis for Non/structured Text Data Applications.**
7. **Handbook of Natural Language Processing** de Nitin INdurkhya y Fred J. Damerau
8. **Applied Text Analysis with Python** de Benjamin Bengfort, Rebecca Bilbro y Tony Ojeda.

## Referencias

- [1] Nobel Prize, “The Nobel Prize in Physics 1956.” <https://www.nobelprize.org/prizes/physics/1956/summary/> (accessed Feb. 14, 2021).
- [2] Sistemas Website, “Definición de ARPANET.” <https://sistemas.com/arpamet.php> (accessed Feb. 14, 2021).
- [3] R. Gunther McGrath, “The Pace of Technology Adoption is Speeding Up,” *Harvard Business Review*, Nov. 2013.
- [4] R. R. Schaller, “Moore’s law: past, present and future,” *IEEE Spectr.*, vol. 34, no. 6, pp. 52–59, 1997, doi: 10.1109/6.591665.
- [5] R. Kluver, “Globalization, informatization, and intercultural communication.” na, 2000, [Online]. Available: <http://ac-journal.org/journal/vol3/iss3/spec1/kluver.htm>.
- [6] A. P. Sheth, “Citizen Sensing , Social Signals , and Enriching Human Experience,” *IEEE Internet Comput.*, vol. 13, no. August, pp. 87–92, 2009.
- [7] “Chapter 4: Case-studies for practical Information Extraction,” .
- [8] G. Miner, J. Elder IV, A. Fast, T. Hill, R. Nisbet, and D. Delen, *Practical text mining and statistical analysis for non-structured text data applications*. Academic Press, 2012.
- [9] M. A. Montemurro, “Beyond the Zipf–Mandelbrot law in quantitative linguistics,” *Phys. A Stat. Mech. its Appl.*, vol. 300, no. 3–4, pp. 567–578, 2001.
- [10] I. Augenstein, L. Derczynski, and K. Bontcheva, “Generalisation in named entity recognition: A quantitative analysis,” *Comput. Speech Lang.*, vol. 44, pp. 61–83, 2017, doi: 10.1016/j.csl.2017.01.012.
- [11] A. G. Agirre, M. Marimon, A. Intxaurreondo, O. Rabal, M. Villegas, and M. Krallinger, “PharmaCoNER: Pharmacological Substances, Compounds and proteins Named Entity Recognition track,” pp. 1–10, 2019, doi: 10.18653/v1/d19-5701.
- [12] R. Socher *et al.*, “Recursive deep models for semantic compositionality over a sentiment treebank,” *EMNLP 2013 - 2013 Conf. Empir. Methods Nat. Lang. Process. Proc. Conf.*, pp. 1631–1642, 2013.
- [13] J. Pennington, R. Socher, and C. D. Manning, “GloVe: Global Vectors for Word Representation,” 2020. <https://nlp.stanford.edu/projects/glove/>.
- [14] F. Liu, F. Weng, B. Wang, and Y. Liu, “Insertion , Deletion , or Substitution ? Normalizing Text Messages without Pre-categorization nor Supervision,” in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics:shortpapers*, 2011, pp. 71–76.
- [15] J. Chang, J. Boyd-Graber, C. Wang, S. Gerrish, and D. M. Blei, “Reading tea leaves: How humans interpret topic models,” in *Neural information processing systems*, 2009, vol. 22, pp. 288–296.
- [16] NETBASE, “Netbase Arbys Onlinse Social Networks sentiment analysis case study,” 2018.
- [17] NETBASE, “Netbase Chevrolet customer insights case study,” 2016. [Online]. Available: [https://www.netbase.com/wp-content/uploads/NetBase\\_CS\\_Chevrolet\\_2016.pdf](https://www.netbase.com/wp-content/uploads/NetBase_CS_Chevrolet_2016.pdf).

- [18] A. Sarker *et al.*, “Utilizing social media data for pharmacovigilance : A review,” *J. Biomed. Inform.*, vol. 54, pp. 202–212, 2015, doi: 10.1016/j.jbi.2015.02.004.
- [19] A. Sarker *et al.*, “Social Media Mining for Toxicovigilance : Automatic Monitoring of Prescription Medication Abuse from Twitter,” *Drug Saf.*, vol. 39, no. 3, pp. 231–240, 2016, doi: 10.1007/s40264-015-0379-4.
- [20] J. Liu, Y. Cao, C.-Y. Lin, Y. Huang, and M. Zhou, “Low-quality product review detection in opinion summarization,” in *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, 2007, pp. 334–342.
- [21] Y. Hong and S. Skiena, “The wisdom of bookies? sentiment analysis versus. the nfl point spread,” in *Proceedings of the International AAAI Conference on Web and Social Media*, 2010, vol. 4, no. 1.
- [22] W. Jiang, Y. Wang, M. H. Tsou, and X. Fu, “Using social media to detect outdoor air pollution and monitor air quality index (AQI): A geo-targeted spatiotemporal analysis framework with sina weibo (Chinese twitter),” *PLoS One*, vol. 10, no. 10, pp. 1–18, 2015, doi: 10.1371/journal.pone.0141185.
- [23] L. Gasco, C. Clavel, C. Asensio, and G. de Arcas, “Beyond sound level monitoring: Exploitation of social media to gather citizens subjective response to noise,” *Sci. Total Environ.*, vol. 658, pp. 69–79, doi: 10.1016/j.scitotenv.2018.12.071.
- [24] A. Alfarrarjeh, S. Agrawal, S. H. Kim, and C. Shahabi, “Geo-Spatial Multimedia Sentiment Analysis in Disasters,” in *2017 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, Oct. 2017, pp. 193–202, doi: 10.1109/DSAA.2017.77.
- [25] D. Buscaldi and F.- Villetaneuse, “Sentiment Analysis on Microblogs for Natural Disasters Management : A Study on the 2014 Genoa Floodings,” in *Proceedings of the 24th International Conference on World Wide Web*, 2015, pp. 1185–1188, doi: 10.1145/2740908.2741727.
- [26] M. V Sangameswar, M. Nagabhushana Rao, and S. Satyanarayana, “An algorithm for identification of natural disaster affected area,” *J. Big Data*, vol. 4, no. 1, p. 39, Nov. 2017, doi: 10.1186/s40537-017-0096-1.
- [27] W. So, E. P. Bogucka, S. Scepanovic, S. Joglekar, K. Zhou, and D. Quercia, “Humane Visual AI: Telling the Stories Behind a Medical Condition,” *IEEE Trans. Vis. Comput. Graph.*, 2020.
- [28] B. Liu, *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers, 2012.
- [29] A. Esuli and F. Sebastiani, “SentiWordNet : A Publicly Available Lexical Resource for Opinion Mining,” in *In Proceedings of the 5th Conference on Language Resources and Evaluation (LREC’06)*, 2006, pp. 417–422.
- [30] A. Valitutti, C. Strapparava, and O. Stock, “Developing affective lexical resources,” *PsychNology J.*, vol. 2, no. 1, pp. 61–83, 2004.
- [31] P. K. Novak, J. Smailovi, B. Sluban, and I. Mozeti, “Sentiment of Emojis,” no. March, pp. 1–22, 2015, doi: 10.1371/journal.pone.0144296.
- [32] B. Han, P. Cook, and T. Baldwin, “Automatically Constructing a Normalisation Dictionary for Microblogs,” in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural*

- Language Processing and Computational Natural Language Learning*, 2012, pp. 421–432, [Online]. Available: <http://dl.acm.org/citation.cfm?id=2390948.2391000>.
- [33] B. Han, P. Cook, and T. Baldwin, “Lexical Normalization for Social Media Text,” *ACM Trans. Intell. Syst. Technol.*, vol. 4, no. 1, pp. 5:1–5:27, 2013, doi: 10.1145/2414425.2414430.
- [34] L. Derczynski *et al.*, “Analysis of named entity recognition and linking for tweets,” *Inf. Process. Manag.*, vol. 51, no. 2, pp. 32–49, 2015, doi: <https://doi.org/10.1016/j.ipm.2014.10.006>.
- [35] O. Owoputi, B. O. Connor, C. Dyer, K. Gimpel, N. Schneider, and N. A. Smith, “Improved Part-of-Speech Tagging for Online Conversational Text with Word Clusters,” in *Proceedings of NAACL-HLT 2013*, 2013, pp. 380–390.
- [36] Z. Jianqiang, “Pre-processing Boosting Twitter Sentiment Analysis?,” in *IEEE International Conference on Smart City/SocialCom/SustainCom*, 2015, pp. 748–753, doi: 10.1109/SmartCity.2015.158.
- [37] J. Pennington, R. Socher, and C. D. Manning, “GloVe: Global Vectors for Word Representation,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543, [Online]. Available: <http://www.aclweb.org/anthology/D14-1162>.

