ISSN 2395-8618

Análisis de similitud de documentos de texto

Claudio Isaac Soriano Burgos, Rafael Guzman-Cabrera*

***Resumen*—La clasificación automática de textos constituye una de las áreas más relevantes dentro del aprendizaje automático y la minería de datos, debido al crecimiento exponencial de la información digital generada y almacenada diariamente en forma de documentos electrónicos, publicaciones en redes sociales, correos electrónicos, noticias y otros recursos textuales. El objetivo de esta tarea es asignar automáticamente categorías o etiquetas a los documentos en función de su contenido, facilitando así la organización, recuperación y análisis de grandes volúmenes de información. Uno de los procesos fundamentales para lograr una clasificación eficiente es el preprocesamiento del texto. Esta etapa consiste en transformar los documentos originales en una representación adecuada para su análisis computacional, mediante técnicas como la eliminación de caracteres especiales, conversión a minúsculas, eliminación de palabras vacías (stopwords), tokenización, lematización o stemming, entre otras. Un adecuado preprocesamiento contribuye significativamente a mejorar la calidad de los datos y, por ende, el desempeño de los algoritmos de clasificación. En este contexto, el presente trabajo propone el desarrollo de un método (implementado en un script en Python) orientado al preprocesamiento automatizado de documentos de texto para su posterior clasificación mediante el software de aprendizaje automático Weka. La herramienta desarrollada permite preparar los documentos de manera eficiente, generando archivos compatibles con las estructuras de datos requeridas por dicho entorno de análisis. Adicionalmente, el sistema incorpora mecanismos para evaluar la similitud entre documentos mediante la aplicación de los coeficientes de Jaccard y Sorensen-Dice, presentando los resultados a través de representaciones gráficas que facilitan la interpretación visual de las relaciones existentes entre los textos analizados. De esta manera, la propuesta contribuye a optimizar las etapas iniciales del proceso de minería de textos y proporciona herramientas complementarias para el análisis exploratorio de colecciones documentales.**

***Palabras clave*—Similitud entre textos, Jaccard, Sorensen-Dice, preprocesamiento del texto.**

Text Document Similarity Analysis

***Abstract*—Automatic text classification is one of the most relevant areas within machine learning and data mining, due to the exponential growth of digital information generated and stored daily in the form of electronic documents, social media posts, emails, news articles, and other textual resources. The goal of this task is to automatically assign categories or tags to documents based on their content, thus facilitating the organization, retrieval, and analysis of large volumes of information. One of the fundamental processes for achieving efficient classification is text preprocessing. This stage consists of transforming the original documents into a representation suitable for computational analysis, using techniques such as the removal of special characters, conversion to lowercase, removal of stop words, tokenization, lemmatization, and stemming, among others. Proper preprocessing significantly contributes to improving data quality and, consequently, the performance of classification algorithms. In this context, this paper proposes the development of a method (implemented in a Python script) for the automated preprocessing of text documents for subsequent classification using the Weka machine learning software. The developed tool efficiently prepares documents, generating files compatible with the data structures required by this analytical environment. Additionally, the system incorporates mechanisms to evaluate document similarity using Jaccard and Sorensen-Dice coefficients, presenting the results through graphical representations that facilitate the visual interpretation of the relationships between the analyzed texts. Thus, the proposal contributes to optimizing the initial stages of the text mining process and provides complementary tools for the exploratory analysis of document collections.**

***Index Terms*—Text similarity, Jaccard, Sorensen-Dice, text preprocessing.**

I. Introducción

En los últimos años, el aprendizaje automático supervisado ha experimentado un crecimiento significativo debido al incremento exponencial de la información digital disponible en formato electrónico. Documentos académicos, noticias, publicaciones en redes sociales, correos electrónicos y otros recursos textuales son generados diariamente en grandes cantidades, lo que ha impulsado el desarrollo de técnicas capaces de analizar y organizar automáticamente dichos contenidos. En este contexto, la clasificación automática de textos se ha consolidado como una de las áreas más importantes dentro de la minería de datos y el procesamiento del lenguaje natural.

La clasificación de textos puede definirse como el proceso mediante el cual un documento es asignado automáticamente a una o varias categorías previamente establecidas, de acuerdo con las características y el contenido semántico que presenta. Esta tarea permite organizar grandes colecciones documentales y facilita procesos como la recuperación de información, el filtrado de contenidos, el análisis de opiniones, la detección de spam y la gestión inteligente del conocimiento. El objetivo principal de la clasificación de textos es transformar grandes volúmenes de información no estructurada en conocimiento útil que pueda ser aprovechado por usuarios y sistemas computacionales.

Para lograr resultados satisfactorios en los procesos de clasificación, es indispensable realizar una etapa previa conocida como preprocesamiento de datos. Esta fase tiene como finalidad preparar los documentos para su posterior

Manuscript received on 26/01/2025, accepted for publication on 11/03/2025. .Corresponding author is Rafael Guzman-Cabrera (guzmanc81@gmail.com).

Authors are with University of Guanajuato, Mexico.

ISSN 2395-8618

análisis mediante la aplicación de diversas técnicas, tales como la normalización de texto, la eliminación de caracteres especiales, la tokenización, la eliminación de palabras vacías (stopwords), la lematización o el stemming, entre otras. La calidad de esta etapa influye directamente en el desempeño de los algoritmos de aprendizaje automático, ya que permite reducir el ruido presente en los datos y resaltar la información relevante para el proceso de clasificación.

Debido a la importancia de esta fase, resulta conveniente contar con herramientas de software que automaticen las tareas de preprocesamiento y faciliten el trabajo de investigadores, docentes y estudiantes que desarrollan proyectos relacionados con la minería de textos. Funcionalidades como la tokenización automática, la generación de matrices de términos, el cálculo de medidas de similitud y la visualización gráfica de las relaciones existentes entre documentos permiten optimizar el análisis de grandes colecciones textuales y mejorar la comprensión de los datos antes de aplicar técnicas de clasificación.

Por ello, el presente trabajo se enfoca en el desarrollo de una herramienta basada en Python que permite realizar diversas tareas de preprocesamiento de documentos de texto y generar representaciones gráficas de similitud documental utilizando métricas ampliamente empleadas en la literatura, como los coeficientes de Jaccard y Sorensen-Dice. La herramienta propuesta busca simplificar las etapas iniciales del proceso de clasificación de textos y proporcionar un entorno de apoyo para la preparación de datos destinados a sistemas de aprendizaje automático, como Weka, contribuyendo así a mejorar la eficiencia y la calidad de los análisis realizados sobre colecciones documentales.

II. Estado del arte

Un trabajo en la medición de similitud es reportado en (Niwattanakul et al, 2013) en donde se propone una medición de similitud entre palabras mediante el uso del coeficiente de Jaccard, implementado en el lenguaje de programación Prolog y concluyen que el coeficiente de similitud de Jaccard es lo suficientemente adecuado para ser empleado en la medida de similitud de palabras, pero sin tener en cuenta la similitud entre las oraciones, lo que aumenta la proporción de similitud. Por otra parte, (Takale y Nandgaonkar, 2010) proponen 5 medidas de asociación en la recuperación de información de cualquier enciclopedia, como la Enciclopedia Británica o Wikipedia: Dice, Jaccard, overlap, coincidencia simple y coeficiente de coseno.

Referente a la clasificación de textos, (Kowsari et al, 2019) mencionan que los textos y documentos son conjuntos de datos no estructurados y que la extracción de características y el procesamiento previo son pasos cruciales para las aplicaciones de clasificación de texto. Como parte de este preprocesamiento mencionan que los datos deben de limpiarse para omitir caracteres y características innecesarias, además de realizar una tokenización y la eliminación de stop words. En un estudio de (Wang J. y Dong Y., 2020) indican que el proceso de medición en la similitud de textos se puede dividir en la distancia del texto y la representación del texto. La distancia del texto se puede dividir en distancia de longitud, distancia de distribución y distancia semántica; la representación de texto se divide en texto basado en cadenas, basado en corpus, texto semántico

Tabla I
Distribución de los archivos en el corpus.

Categoría	Positivos	Negativos	Total
books	1037	1463	2500
dvd	1394	1390	2784
electronics	1015	949	1964
kitchen	923	833	1756
			9004

Tabla II
Distribución de los archivos en clases.

Clase	Positivos	Negativos	Archivos
C100	100	100	1 - 100
C200	200	200	101 - 200
C400	400	400	201 - 400
C800	800	800	401 - 800

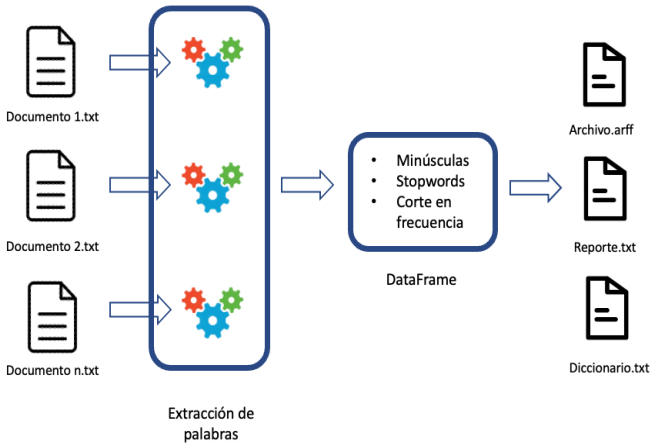

Fig. 1. Generación de archivos.

único, texto multisemántico y representación basada en estructura gráfica. Dentro de la similitud de texto basado en cadenas se encuentran los coeficientes Sorensen – Dice y de Jaccard. Concluyen que estos métodos tienen en cuenta el significado real del texto, sin embargo, no se pueden adaptar a diferentes dominios e idiomas.

El objetivo de las técnicas de aprendizaje automático supervisado para clasificación automática de texto (Kadhim A., 2019), es determinar si un documento determinado pertenece o no a la categoría dada mirando las palabras o términos de esa categoría. De las técnicas más utilizadas se encuentran Naive-Bayes (Mohammad et al, 2016), K-vecinos cercanos (Chen S. 2018), (Azam et al, 2018)

En el área de software especializado para realizar tareas de aprendizaje supervisado (Merlini & Rossini, 2021), se cuenta con WEKA (Entorno de Waikato para el análisis del conocimiento), disponible en https://www.cs.waikato.ac.nz/ml/weka/, es una colección avanzada de algoritmos de aprendizaje automático y técnicas de preprocesamiento que se ha diseñado para probar diferentes metodologías existentes como Naive Bayes, árboles de decisión, SVM, clústering y ábroles aleatorios, siendo una herramienta que permite realizar de manera efectiva tareas de aprendizaje supervisado.Los archivos ejecutables en WEKA son archivos con extensión ARFF y pertenecen a University of

ISSN 2395-8618

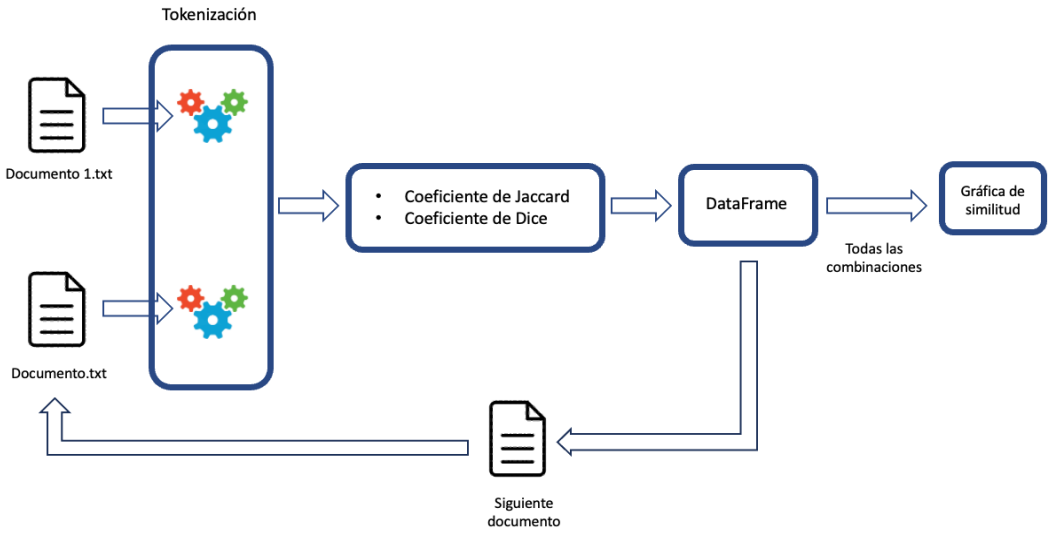

Fig. 2. Generación de gráficas de similitud.

Waikato. Son archivos de texto ASCII y están divididos en 2 secciones: una cabecera que contiene los datos del nombre de las relaciones, las listas de atributos y sus tipos, cada una señalada con @ATTRIBUTE; y una sección de datos, en donde se especifican las instancias del conjunto de datos, señalado con @DATA.

III. Metodología

A. Conjunto de datos

Con la finalidad de generar los archivos ARFF y calcular los coeficientes de similitud de Jaccard y Dice, se formaron clases con la misma cantidad de archivos de cada categoría. Partiendo de un orden alfabético, la clase C100 está formada por los archivos 1 al 100 de la categoría positivos y por los archivos 1 al 100 de la categoría negativos; la clase C200 está formada por los archivos 101 al 200 de la categoría positivos y así sucesivamente.

El conjunto de datos utilizado consiste en un corpus de archivos de texto con los comentarios de usuarios de la tienda en línea Amazon, referentes a sus experiencias en la compra de libros, dvd's, electrónicos y artículos de cocina, separados en 2 categorías (positivos y negativos) de acuerdo si la experiencia en la compra fue positiva o negativa. El corpus consiste en 9004 archivos de texto separados de la siguiente forma:

B. Generación de archivos

Se implementó un script en Python con la finalidad de generar 3 tipos de archivos (Fig. 1): un archivo con extensión arff; un archivo de texto llamado Reporte, en donde se indican las rutas de las categorías y las palabras con mayor frecuencia que se extrajeron de los archivos de texto; y un archivo de texto llamado Diccionario, en donde se registran todas las palabras extraídas de los archivos de texto junto con sus frecuencias.

El algoritmo para la generación de archivos ARFF (Fig. 1) consiste en tomar cada archivo localizado en la clase. Se realiza la extracción de las palabras eliminando espacios vacíos y caracteres especiales, además de pasar todas las palabras a minúsculas para después ser almacenadas en una lista temporal. Posteriormente, cuando se han capturado todas las palabras de los archivos, se encuentran los valores únicos y sus frecuencias, posteriormente, estos datos se almacenan en un dataframe en donde se continuará el proceso de eliminación de las stopwords y de corte en frecuencia (ambas opciones si el usuario lo requiere). El dataframe resultante se utiliza para la generación de los archivos ARFF, Reporte y Diccionario.

```
RELATION /Ruta/nombre_archivo.arff
@ATTRIBUTE book numeric
@ATTRIBUTE also numeric
@ATTRIBUTE like numeric
@ATTRIBUTE oil numeric
@ATTRIBUTE states numeric
@ATTRIBUTE time numeric
@ATTRIBUTE kirk numeric
@ATTRIBUTE found numeric
@ATTRIBUTE one numeric
@ATTRIBUTE even numeric
@ATTRIBUTE class {positive, negative}
@DATA
0, 0, 0, 0, 0, 0, 0, 1, 0, positive
0, 0, 0, 0, 0, 0, 0, 0, 0, positive
0, 1, 1, 0, 0, 0, 0, 1, 0, positive
0, 0, 0, 0, 0, 0, 0, 0, 0, positive
0, 0, 0, 0, 0, 1, 0, 0, 0, positive
0, 0, 0, 0, 0, 0, 0, 0, 0, positive
0, 0, 0, 0, 0, 1, 0, 0, 1, positive
0, 0, 0, 0, 0, 0, 0, 0, 0, positive
0, 0, 0, 0, 0, 1, 0, 0, 0, positive
0, 0, 1, 0, 0, 0, 0, 0, 1, positive
0, 0, 1, 0, 0, 0, 0, 0, 0, negative
0, 0, 0, 0, 0, 0, 0, 0, 0, negative
0, 0, 1, 0, 0, 0, 0, 0, 0, negative
0, 0, 1, 0, 0, 0, 0, 0, 1, negative
0, 0, 1, 0, 0, 0, 0, 1, 0, negative
0, 0, 0, 0, 0, 0, 0, 0, 1, negative
0, 0, 1, 0, 0, 0, 0, 0, 0, negative
0, 0, 0, 0, 0, 0, 0, 1, 0, negative
0, 0, 0, 0, 0, 0, 0, 0, 1, negative
0, 0, 0, 0, 0, 0, 0, 0, 0, negative
```

Fig. 3. Archivo ARFF generado.

	x	y	dice
0	1	1	1.000000
1	1	2	0.020833
2	1	3	0.037383
3	1	4	0.029851
4	1	5	0.050314
...	...	...	...
159995	400	396	0.014706
159996	400	397	0.075472
159997	400	398	0.000000
159998	400	399	0.044118
159999	400	400	1.000000

	x	y	jaccard
0	1	1	1.000000
1	1	2	0.010526
2	1	3	0.019048
3	1	4	0.015152
4	1	5	0.025806
...	...	...	...
159995	400	396	0.007407
159996	400	397	0.039216
159997	400	398	0.000000
159998	400	399	0.022556
159999	400	400	1.000000

Fig. 4. Cálculo de los coeficientes de Jaccard y Dice para la clase C400.

ISSN 2395-8618

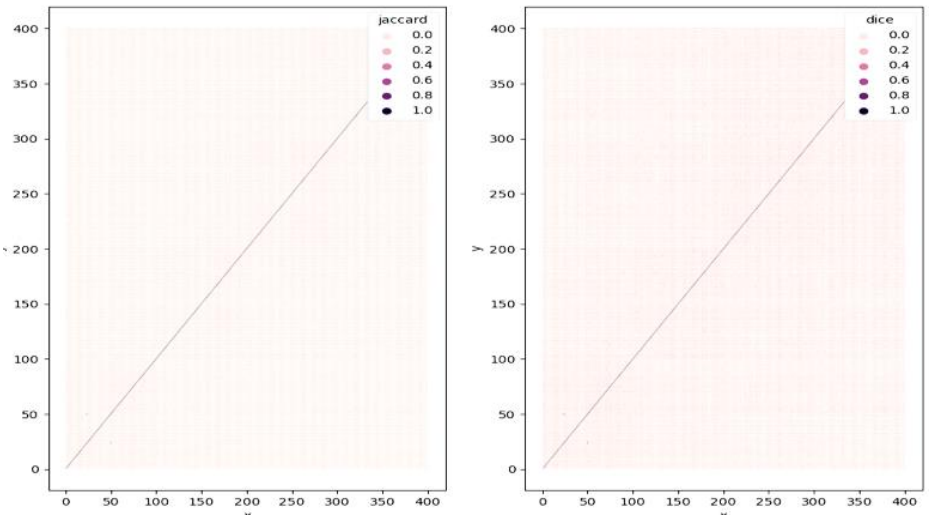

Fig. 5. Gráficas de similitud de la clase C100, sin stopwords.

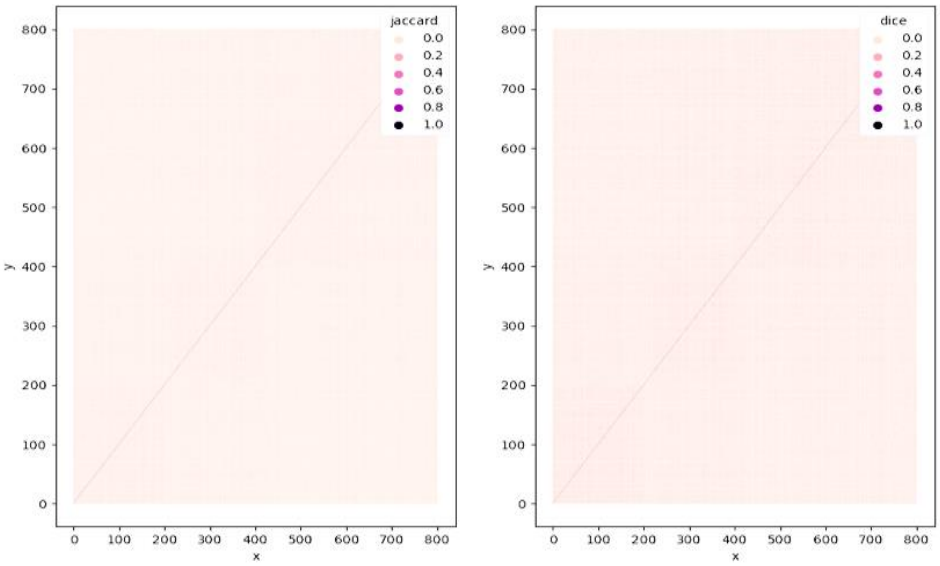

Fig. 6. Gráficas de similitud de la clase C200, sin stopwords.

En el algoritmo se ha implementado también la función de realizar la carga del archivo Reporte.txt, con la finalidad de ser archivo fuente para la modificación o generación de archivos ARFF, permitiendo recortar la cantidad de palabras y de realizar nuevas búsquedas de palabras en los archivos que conforman el corpus en otras rutas.

IV. Medidas de similitud

A. Coeficiente de Jaccard

El coeficiente de Jaccard es una de las técnicas que se utilizan para medir la similitud léxica. Es un valor numérico entre 0 y 1, que mide la similitud entre dos documentos de texto (1 = completamente iguales; 0 = completamente desiguales) considerando las palabras que tengan en común ambos documentos. Este coeficiente considera que todos los elementos o atributos son igualmente importantes. Puede ser visto como una medida de similitud sobre conjuntos:

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

B. Coeficiente de Dice

El coeficiente de Sorensen-Dice es un coeficiente de similitud parecido al coeficiente de Jaccard que, de igual manera, su valor se encuentra entre 0 y 1, pero a diferencia de Jaccard, este coeficiente considera de mayor importancia a los elementos en común de un par de conjuntos:

$$s(A,B) = \frac{2|A \cap B|}{|A| + |B|}.$$

V. Representación de similitud

A. Coeficiente de Jaccard

Para la generación de las gráficas de similitud se consideraron el coeficiente de Jaccard y el coeficiente de Dice (a elegir por el usuario). El algoritmo implementado consiste en

ISSN 2395-8618

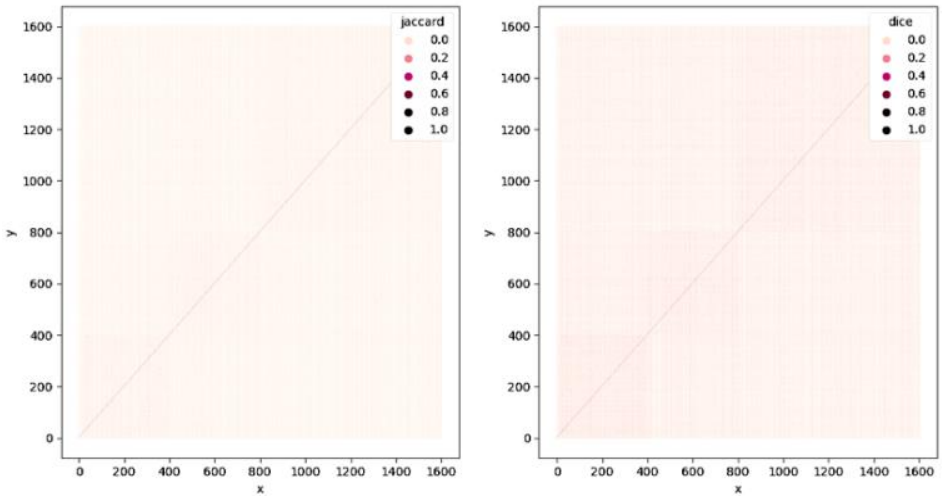

Fig. 7. Gráficas de similitud de la clase C400, sin stopwords.

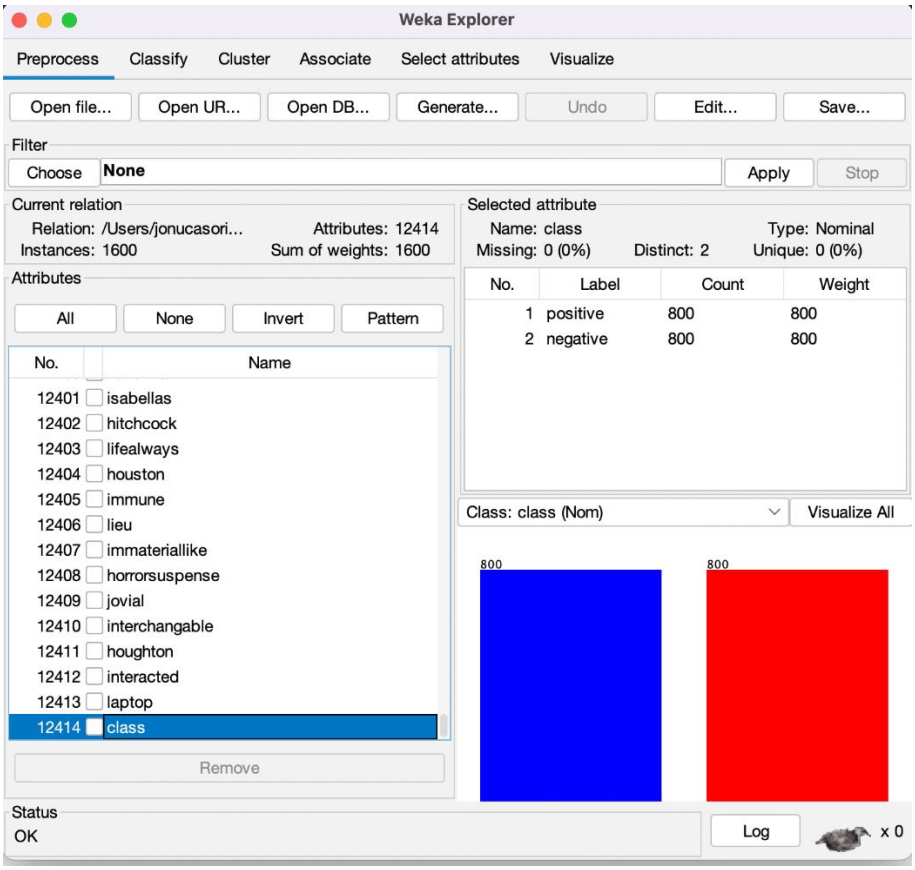

Fig. 8. Ejecución del archivo ARFF para la clase C400 en Weka.

tomar cada archivo por separado y realizar el proceso de tokenización, en donde se separan las palabras que forman el texto, se eliminan los espacios vacíos y los caracteres especiales y se convierten a minúsculas.

Posteriormente, estas palabras son almacenadas en dos listas para calcular el coeficiente de similitud indicado y se almacena en un dataframe.

Cuando un archivo ha sido comparado con el resto, se tomará un segundo archivo de la clase y se repetirá el proceso hasta completar todas las combinaciones.

Al finalizar de realizar las comparaciones, se utilizan los datos del dataframe para realizar una gráfica dispersión que será la gráfica de similitud de todos los archivos de la clase seleccionada.

VI. Resultados

Como resultado de la ejecución del script, se han generado los archivos ARFF, Reporte.txt y Diccionario.txt con los datos que se requieren para la ejecución en Weka y para la carga de un nuevo proceso en el script.

ISSN 2395-8618

Para la generación de las gráficas de similitud, se calcularon los coeficientes de Jaccard y Dice, de cada par de archivos contenidos en las clases C100, C200, C400 y C800. Los valores de estos coeficientes se encuentran en el rango de 0 a 1, en donde el valor 0 corresponde a una completa desigualdad entre los dos archivos, mientras que el valor igual a 1 corresponde al caso en el que se compara un archivo consigo mismo. De acuerdo con las gráficas de similitud generadas, se puede observar un valor más alto en el coeficiente Dice, debido a que este coeficiente les da una mayor importancia a los elementos en común, a diferencia del coeficiente de Jaccard, donde todos los elementos tienen la misma importancia.

El archivo ARFF generado se pudo ejecutar en el software Weka, el cual nos permite realizar tareas de aprendizaje automático aprovechando las funciones con las que cuenta el software.

La incorporación de métricas de similitud documental, específicamente los coeficientes de Jaccard y Sorensen-Dice, permitió analizar las relaciones existentes entre los documentos procesados. La representación gráfica de estas medidas proporciona una forma intuitiva de identificar grupos de documentos con características similares, facilitando la exploración preliminar de los datos y apoyando la toma de decisiones antes de aplicar algoritmos de clasificación supervisada.

Los resultados obtenidos demuestran que la automatización de las tareas de preprocesamiento y análisis de similitud puede mejorar significativamente la eficiencia de los procesos de minería de textos. Además, la herramienta desarrollada ofrece una alternativa accesible y flexible para investigadores, docentes y estudiantes interesados en el análisis de documentos electrónicos, integrando funcionalidades que tradicionalmente requieren el uso de múltiples aplicaciones independientes.

VII. Conclusiones

La clasificación automática de textos constituye una de las aplicaciones más relevantes del aprendizaje automático debido al crecimiento constante de la información digital disponible en formato textual. Sin embargo, el desempeño de los algoritmos de clasificación depende en gran medida de la calidad de los datos de entrada, por lo que la etapa de preprocesamiento desempeña un papel fundamental dentro del proceso de minería de textos. Una adecuada preparación de los documentos permite reducir el ruido presente en los datos, extraer características significativas y mejorar la capacidad de los modelos para identificar patrones relevantes.

En este trabajo se desarrolló una herramienta basada en Python que automatiza diversas tareas de preprocesamiento textual, facilitando la preparación de documentos para su posterior análisis y clasificación mediante el software Weka. La implementación de procesos como la tokenización y la generación de estructuras compatibles con herramientas de aprendizaje automático contribuye a reducir el tiempo y esfuerzo requeridos durante las etapas iniciales del análisis de datos textuales.

Como trabajo futuro, se propone incorporar nuevas técnicas de procesamiento del lenguaje natural, tales como lematización, eliminación automática de términos irrelevantes, extracción de características mediante TF-IDF y representaciones vectoriales basadas en embeddings.

Referencias

[1] S. Takale and S. Nandgaonkar, “Measuring semantic similarity between words using web documents,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 1, no. 4, Oct. 2010.

[2] S. Niwattanakul, J. Singthongchai, E. Naenudorn, and S. Wanapu, “Using Jaccard coefficient for keywords similarity,” in *Proc. Int. MultiConf. Eng. Comput. Sci. (IMECS)*, vol. I, Mar. 2013.

[3] K. Kowsari, K. Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, “Text classification algorithms: A survey,” *Information*, vol. 10, p. 15, 2019.

[4] V. Saquiela, F. Baculima, G. Orellana, N. Piedra, M. Orellana, and M. Espinoza, “Similarity detection among academic contents through semantic technologies and text mining,” in *CEUR Workshop Proc.*, vol. 2096, 2018.

[5] S. Canuto, T. Salles, T. Rosa, and M. Gonçalves, “Similarity-based synthetic document representations for meta-feature generation in text classification,” in *Proc. 42nd Int. ACM SIGIR Conf. Res. Dev. Inf. Retr. (SIGIR)*, Jul. 2019.

[6] P. Rosso, S. Correa, and D. Buscaldi, “Passage retrieval in legal texts,” *J. Log. Algebraic Program.*, vol. 80, no. 3–5, pp. 139–153, 2011.

[7] S. Chen, “K-nearest neighbor algorithm optimization in text categorization,” *IOP Conf. Ser.: Earth Environ. Sci.*, vol. 108, 2018.

[8] M. Azam, T. Ahmed, F. Sabah, and M. Hussain, “Feature extraction based text classification using K-nearest neighbor algorithm,” *Int. J. Comput. Sci. Netw. Secur.*, vol. 18, no. 12, 2018.

[9] Kadhim, “Survey on supervised machine learning techniques for automatic text classification,” *Artif. Intell. Rev.*, 2019.

[10] Mohammad, T. Alwanda, and O. Al-Momani, “Arabic text categorization using support vector machine, naïve Bayes and neural network,” *GSTF J. Comput.*, vol. 5, no. 1, pp. 108–115, 2016.

[11] X. Luo, “Efficient English text classification using selected machine learning techniques,” Hunan Univ. Technol. Bus., China, Tech. Rep., 2020.

[12] D. Merlini and M. Rossini, “Text categorization with WEKA: A survey,” unpublished, 2021.