UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Revista Científica

http://revistas.udistrital.edu.co/ojs/index.php/revcie/index

CIDC

Artículo de investigación científica y tecnológica

Servicio de clasificación documental multi cliente basado en técnicas de aprendizaje de máquina y Elasticsearch

Multi-Client Document Classification Service Based on Machine Learning Techniques and Elasticsearch

Serviço de classificação documentária multi-cliente baseado em técnicas de aprendizagem de máquina e Elasticsearch

David-Santiago García-Chicangana[1]
Carlos-Alberto Cobos-Lozada[2]
Martha-Eliana Mendoza-Becerra[3]
Miguel-Ángel Niño-Zambrano[4]
James-Mauricio Martínez-Figueroa[5]

Recibido: agosto de 2021 **Aceptado:** octubre de 2021

Para citar este artículo: García-Chicangana, D. S., Cobos-Lozada, C. A., Mendoza-Becerra, M. E., Niño-Zambrano, M. A. y Martínez-Figueroa, J. M. (2022). Servicio de clasificación documental multi cliente basado en técnicas de aprendizaje de máquina y Elasticsearch. *Revista Científica*, *43*(1), 64-79. https://doi.org/10.14483/23448350.18352

Resumen

Este artículo presenta un servicio de clasificación documental que permite a los sistemas de gestión documental de múltiples clientes brindar una mayor confianza y credibilidad sobre los tipos documentales asignados a los documentos que cargan los usuarios. La investigación fue realizada a través de las fases de CRISP-DM en las que se evaluaron dos modelos de representación de documentos, bolsas de palabras con n-gramas acumulativos y BERT (propuesto recientemente por Google), y cinco técnicas de aprendizaje de máquina, perceptrón multicapa, bosques aleatorios, k vecinos más cercanos, árboles de decisión y un clasificador bayesiano ingenuo. Los experimentos se realizaron con datos de dos organizaciones y los mejores resultados fueron los obtenidos por el perceptrón multicapa, los bosques aleatorios y los k vecinos más cercanos, con resultados muy similares de exactitud general y recuerdo por clase para los tres algoritmos. Los resultados no son concluyentes para ofertar el servicio a múltiples clientes con un solo modelo, ya que esto depende de los documentos y tipos documentales de cada uno de ellos. Por lo anterior, se ofrece un servicio basado en una arquitectura de microservicios que permite a cada organización la creación de su propio modelo, el monitoreo de su rendimiento en producción y su actualización cuando el rendimiento no sea adecuado.

1. Nexura S.A.S. Cali, Colombia. dsgarcia@nexura.com.
2. Ph. D. Universidad del Cauca. Popayán, Colombia. ccobos@unicauca.edu.co.
3. Ph. D. Universidad del Cauca. Popayán, Colombia. mmendoza@unicauca.edu.co.
4. Ph. D. Universidad del Cauca. Popayán, Colombia. manzamb@unicauca.edu.co.
5. M. Sc. Nexura S.A.S. Cali, Colombia jmartinez@nexura.com.

Palabras clave: analítica de datos; bosques aleatorios; CRISP-DM; k vecinos más cercanos; perceptrón multicapa; sistema de gestión documental; trigramas.

Abstract
This paper presents a document classification service that allows multiple client (multi-tenant) document management systems to provide greater confidence and credibility regarding the document types assigned to documents uploaded by users. The research was carried out through the phases of CRISP-DM, where two document representation models were evaluated (bags of words with cumulative n-grams and BERT, which was recently proposed by Google) and five machine learning techniques (multilayer perceptron, random forests, k-nearest neighbors, decision trees, and naïve bayes). The experiments were carried out with data from two organizations, and the best results were obtained by multilayer perceptron, random forests, and k-nearest neighbors, which showed very similar results regarding general accuracy and recall by class. The results are not conclusive with respect to the ability to offer the service to multiple clients with a single model, since this also depends on their documents and document types. Therefore, a service is offered which is based on a microservices architecture that allows each organization to create its own model, monitor its performance in production, and update it when performance is not adequate.
Keywords: CRISP-DM; data analytics; document management system; k-nearest neighbors; multilayer perceptron; random forests; trigrams.

Resumo
Este artigo apresenta um serviço de classificação de documentos que permite que sistemas de gerenciamento de documentos de múltiplos clientes (multilocatário) forneçam maior confiança e credibilidade nos tipos de documentos atribuídos aos documentos carregados pelos usuários. A pesquisa foi realizada através das fases do CRISP-DM onde foram avaliados dois modelos de representação de documentos, sacos de palavras com n-gramas cumulativos e BERT (recentemente proposto pelo Google) e cinco técnicas de aprendizado de máquina, perceptron multicamadas, florestas aleatórias, k mais próximo vizinhos, árvores de decisão e bayes ingênuos. Os experimentos foram realizados com dados de duas organizações e os melhores resultados foram obtidos pelo perceptron multicamadas, as florestas aleatórias e os k vizinhos mais próximos, com resultados muito semelhantes de precisão geral e recuperação por classe para esses três algoritmos. Os resultados não são conclusivos para oferecer o serviço a vários clientes com um único modelo, pois isso depende também dos documentos e tipos de documentos de cada um deles. Portanto, um serviço é oferecido com base em uma arquitetura de microsserviços que permite a cada organização criar seu próprio modelo, monitorar seu desempenho na produção e atualizá-lo quando o desempenho não for adequado.
Palavras-chaves: análise de dados; CRISP-DM; florestas aleatórias; k-vizinhos mais próximos; perceptron multicamadas; sistema de gerenciamento de documentos; trigramas.

Introducción

Hoy en día las organizaciones que toman ventaja de sus datos y documentos son más competitivas. Es así como los sistemas de gestión documental (SGD) se han convertido en una herramienta fundamental para las organizaciones, ya que permiten un mejor acceso y organización de los documentos que se manejan en los diferentes procesos de negocio (Ismael y Okumus, 2017). Debido a la creciente demanda de búsquedas rápidas, precisas y oportunas de información en grandes volúmenes de datos y documentos, en los últimos años ha crecido la necesidad de incorporar estos sistemas en las organizaciones sin importar su sector económico (Rangel Palencia, 2017). Estos datos y documentos apropiadamente organizados y almacenados han empezado a ser objeto de análisis inteligente para facilitar los procesos en las organizaciones y para apoyar la toma de decisiones (Lacunza, 2020). En resumen, los SGD resultan de vital importancia para estos dos objetivos: facilitar procesos y apoyar la

toma de decisiones (Rodríguez, Castellanos y Ramírez, 2016).

Dado que los procesos de negocio de una organización pueden apoyarse con diferentes tipos documentales y que estos tipos pueden estar representados en documentos estructurados, semiestructurados y totalmente desestructurados, existe el riesgo de que los usuarios carguen documentos con un tipo documental erróneo. Esto afecta la calidad de la información que se recupera en el SGD e involucra demoras en el desarrollo de los procesos empresariales. Es por esto que se hace necesario realizar un proceso de clasificación automática de documentos que permita validar que el documento concuerda con el tipo documental que el usuario ha seleccionado, y en caso de que no corresponda, le advierta inmediatamente al usuario para que tome los correctivos necesarios. Con esta validación se puede aumentar la calidad de los documentos almacenados en el SGD, se aumenta la precisión de la información que se retorna cuando se hace una consulta, se disminuyen los retrasos en el desarrollo de los procesos de negocio y se tiene una fuente de datos de mayor calidad para el desarrollo de futuros trabajos de analítica de datos.

La clasificación de documentos es un proceso que involucra varias etapas, y las decisiones tomadas en cada una de estas etapas junto con los documentos que se manejan define la calidad del resultado. Las etapas son: (i) ***preprocesamiento*** y limpieza del texto, que incluye tokenización, conversión a minúsculas, eliminación de palabras vacías, stemming o lematización, entre otras; (ii) ***extracción de características*** que representan los documentos basados en tokens, palabras, n-gramas, incrustaciones, frases, párrafos u otras y cálculo de los pesos de estas características en los documentos; (iii) ***reducción de la dimensionalidad*** con técnicas como análisis de componentes principales, análisis discriminante lineal, factorización de matrices no negativas, descomposición en valores singulares, wrappers y filtors, entre otras (Villegas *et al.*, 2018; Dorado *et al.*, 2019), que buscan reducir el tamaño de la representación y eliminar ruido de la colección de documentos; (iv) ***definición de los modelos*** con base en clasificadores como K-NN (Qin *et al.*, 2019), arboles de decisión (Taloba y Ismail, 2019), bosques aleatorios (Lagrari, Ziyati y El Kettani, 2019), clasificadores bayesianos ingenuos (Qu *et al.*, 2018), redes bayesianas, máquinas de soporte vectorial (Hapsari, Utoyo y Purnami, 2020), regresión logística (Gitanjali y Lakhwani, 2019), algoritmo de Rocchio (Selvi *et al.*, 2017), campos aleatorios condicionales (Chen *et al.*, 2017) y redes neuronales como perceptrón multicapa (Yang *et al.*, 2016), redes de aprendizaje profundo o deep learning (Kowsari *et al.*, 2017; Chen, Yan y Wong, 2020; Jiang *et al.*, 2018), o combinaciones de estos; (v) ***evaluación y comparación de los modelos*** obtenidos para seleccionar el mejor y definir si es apropiado para su implementación y uso en producción; y (vi) ***despliegue*** en el ambiente de producción, que implica, además ***monitoreo*** del rendimiento del modelo durante su uso para realizar la actualización de este cuando sea indicado.

Por otro lado, el actual escenario tecnológico en el que los servicios en la nube se han convertido en una alternativa muy ventajosa por su relación costo beneficio, ha permitido a muchas compañías contar con la última tecnología al menor costo posible. Es así como el diseño de los SGD está migrando a arquitecturas multicliente (multi-tenant) en la nube, facilitando que una instancia del sistema almacene información de muchas compañías o clientes, eso sí, con el apropiado aislamiento y la seguridad que permiten a cada cliente tener sus propios procesos y, dentro de cada uno de sus espacios, tener sus propios tipos documentales, que en algunos casos pueden ser similares a los de los otros clientes.

Con base en lo anterior, el presente trabajo propone un modelo de aprendizaje de máquina que basado en el motor de indexación y búsqueda de ElasticSearch, apoye los procesos de clasificación documental en un SGD multi-tenant, permitiendo a las organizaciones realizar el proceso de clasificación con el modelo (Perceptrón Multicapa,

Random Forest, K-NN, Naïve Bayes, o Decision Tree) más apropiado a los documentos que esperan clasificar. Además, presenta los resultados experimentales de dos clientes y una comparación con una adaptación al español del modelo de representación más reciente propuesto por Google, denominado BERT (BETO para español).

A continuación, en la sección siguiente, se presentan algunos trabajos previos de interés para la investigación y luego se resume la metodología seguida para el desarrollo de esta. Después se presentan los resultados obtenidos en cada una de las fases de la metodología y finalmente se presentan las conclusiones de la investigación y el trabajo futuro que se espera realizar en el corto plazo.

Trabajos relacionados

En relación con el proceso de clasificación, además de las referencias presentadas en la sección anterior de introducción, se precisa resumir las siguientes: en Vijayan *et al.* (2017) se presentan las ventajas y desventajas de varios algoritmos al momento de clasificar textos, lo que es muy importante conocer para elegir el más apropiado en un contexto especifico. Aliwy y Ameer (2017) presentan cinco algoritmos que consideran esenciales para el proceso de clasificación, a saber: árboles de decisión, k vecinos más cercanos (K-NN), Naïve Bayes (NB), máquinas de soporte vectorial (SVM) y un modelo oculto de Markov; sus resultados evidenciaron que la manera más práctica para mejorar el proceso de clasificación es mediante la reducción de características, lo que hace que el proceso sea más rápido y eficiente. Y en Rasjid y Setiawan (2017) se comparan dos métodos de clasificación: K-NN y NB, concluyendo que K-NN se desempeñó de una mejor manera, a pesar de que el proceso de clasificación puede ser más lento si se tiene un gran volumen de documentos (este clasificador es perezoso, no establece un modelo sino que usa los datos de entrenamiento para definir la clase de una nueva instancia o registro).

Por otra parte, el procesamiento de lenguaje natural (NLP) también ha hecho grandes aportes a la clasificación de textos o documentos. Además de los diferentes esquemas de representación de documento basados en conteos (modelo binario, modelo de bolsa de palabras, modelos probabilísticos, entre otros), los modelos de lenguaje pre-entrenados (modelos que capturan la sintaxis y la semántica del lenguaje) han demostrado una gran utilidad para la representación de los documentos, generando buenos resultados en varias tareas de entendimiento del lenguaje natural (Cao *et al.*, 2019). Uno de ellos es el modelo BERT, un transformador bidireccional multicapa propuesto por investigadores de Google en 2019 (Devlin *et al.*, 2019) que se puede ajustar para crear modelos de vanguardia para una amplia gama de tareas como la respuesta a preguntas y la inferencia de lenguaje. Aunque este modelo fue entrenado inicialmente con datos en inglés y chino, se han desarrollo propuestas para otros idiomas como el español, el cual es abordado en Cañete *et al.* (2020) y se denomina BETO (presentado en 2020).

Metología

El presente trabajo fue realizado utilizando CRISP-DM (Cross Industry Standard Process for Data Mining) (Wirth, 2000), debido al impacto que ha tenido tanto en la industria como en el contexto investigativo en donde se ha empleado para el desarrollo de diversos proyectos de minería y ciencia de datos (Schröer, Kruse y Gómez, 2021). CRISP-DM es un proceso iterativo e incremental que se desarrolla con las siguientes fases: primero, ***entendimiento de las necesidades del negocio*** que permiten definir los objetivos del proyecto de minería de datos (o big data) que se espera desarrollar. Seguidamente se realiza un proceso de ***entendimiento de los datos*** y su calidad. A continuación, se ***procesan los datos*** para obtener datasets que puedan ser usados por los algoritmos que crean modelos o extraen patrones o tendencias en la fase de ***modelamiento***. Con los modelos generados y la

experiencia ganada en el proceso, se realiza una ***evaluación*** de la calidad y desempeño del modelo, de las cosas que se hicieron bien y de las que se deben mejorar, así como el cumplimiento de los objetivos establecidos en la primera fase. Finalmente, de ser aceptado el modelo, se realiza la planeación y la ejecución de su ***despliegue*** en el entorno de producción. Este modelo no es lineal, algunas fases se pueden realizar en paralelo o en ciclos, como cuando se está entendiendo el negocio que en muchos casos requiere en paralelo entender los datos.

La investigación se realizó sobre el SGD multi-tenant de la empresa Nexura Internacional S.A.S. denominado GFiles. La empresa había determinado que era necesario definir una forma de asegurar que los documentos que se subían como soporte a los procesos correspondían con lo requerido, buscando con esto evitar reprocesos, disminuir el tiempo de ejecución de los procesos y mejorar la satisfacción de los usuarios.

En referencia a las herramientas utilizadas para el almacenamiento y el análisis de la información, se destacan soluciones como Elaticsearch, Hadoop y Spark (Shah, Willick y Mago, 2018). Respecto a la primera, entre sus ventajas están las funcionalidades de búsqueda en tiempo real, sus capacidades para escalar y su facilidad para ser configurada (Zamfir *et al.*, 2019). A pesar de que su velocidad de indexación y búsqueda es ligeramente menor en comparación con otras herramientas como Sphinx, actualmente es un sistema muy completo con varias capacidades para dividir los datos entre varias máquinas logrando con esto realizar tareas con alto desempeño, poco uso de memoria e indexación incremental rápida cuando se trabaja con varios documentos a la vez (Voit *et al.*, 2017). Las anteriores características lo convierten en una gran opción para utilizarlo en procesos de analítica, especialmente en aquellos contextos en los que se requiera analizar datos en tiempo real y por eso fue la herramienta que se usó en GFiles para el almacenamiento y la búsqueda de los documentos, trabajo que se realizó como una fase "0" o previa al proceso de CRISP-DM y de la cual no se dan muchos detalles, ya que corresponde a actividades más de ingeniería y desarrollo que de investigación.

Resultados obtenidos por cada fase de CRISP-DM

A continuación se presentan los resultados obtenidos en el proceso de construcción del modelo de clasificación documental.

Entendimiento del negocio

Para llevar a cabo esta fase, se realizó un análisis de las necesidades y requerimientos de GFiles desde el punto de vista de Nexura como su empresa desarrolladora, la cual además tiene contacto directo con los usuarios de las diferentes organizaciones cliente. Aunque desde la perspectiva de minería de datos se observaron diferentes necesidades, se decidió priorizar la validación de los documentos que se cargan en el SGD en relación con su tipo documental, dado que con esto se mejora la calidad de los datos almacenados, se mejoran los tiempos de respuesta de los procesos y se mejora el nivel de satisfacción de los usuarios finales y las empresas clientes. Por otro lado, como GFiles tiene una interfaz web y se estaba implementado una interfaz móvil, se consideró importante resaltar que el modelo de validación de los documentos se implementara como un microservicio independiente del resto de la aplicación, facilitando así su administración y uso desde diferentes interfaces. De igual forma, implementar el motor de indexación y búsqueda basado en ElasticSearch como un microservicio que sirva de base para la validación de los documentos como para otras tareas propias de GFiles.

Entendimiento de los datos

Originalmente, los datos de GFiles se encontraban almacenados en un sistema de archivos e

indexados por ciertos campos en una base de datos relacional; con la inclusión del microservicio de ElasticSearch, los documentos se indexaron en índices específicos de cada organización cliente facilitando al usuario su búsqueda a texto casi completo (máximo 10 páginas por documento) y mejorando los tiempos de respuesta a los usuarios. El texto de los documentos (generalmente en formato PDF) se extrae con el servicio de OCR de Google y luego se organiza la información en ElasticSearch. La primera empresa que se denominará índice_1 cuenta con 6.096 documentos indexados a texto completo y la segunda, índice_2, cuenta con 2.167 para un total de 8.263. En este punto, y por diferentes razones fuera del alcance de la investigación, no se logró contar con la totalidad de los documentos de estas dos empresas, el cual asciende a 178.971 documentos. Después de revisar la calidad de los documentos indexados se consolidó un total de 3.670 documentos en los dos índices.

Como se esperaba, los tipos documentales de las dos empresas tuvieron similitudes y diferencias, también se observó que de algunos tipos documentales había más registros que de otros (problema de clases desbalanceadas), pero algo que llamó mucho la atención fue que en una misma empresa se contaba con varios tipos documentales muy relacionados entre sí. Por ejemplo, el índice_1 contaba con varios tipos documentales para Sugerencias, además en algunos tipos documentales se encontraba un número muy bajo de registros (Respuestas y Cartas con solo tres documentos cada una) por esto se removieron y al final quedaron 3.664 documentos en el proceso. Teniendo esto en mente, se construyó una vista minable que agrega los tipos documentales relacionados en una misma clase conforme se aprecia en la Tabla 1.

Preparación de datos

Con los datos identificados, se realizó el proceso de limpieza y procesamiento. Inicialmente, la construcción de los datasets y la creación de los modelos se realizaron con RapidMiner, debido a las ventajas que ofrece para prototipar y evaluar de una manera ágil y práctica proyectos de minería de datos. Las tareas aplicadas a cada documento fueron: tokenización, filtrado de tokens por tamaño, transformación de tokens a minúscula, remoción de palabras vacías (stopwords), transformación de tokens a su raíz léxica (stemming), generación de n-gramas (unigramas + bigramas + trigramas). Con los tokens procesados, se creó la matriz de términos por documento con ponderación TF-IDF y en conjunto con la clase respectiva para cada documento se generó la vista minable, la cual quedó conformada por 3.664 registros (documentos) y 14.004 atributos.

Tabla 1. Clases documentales propuestas para la construcción de los modelos

#	Clases	Cantidad	Tipos documentales
1	Comunicaciones	75	183-Comunicaciones recibidas, 182-Comunicaciones enviadas, 2023-Comunicaciones internas, 4736-Comunicaciones externas, 4314-Comunicaciones externas
2	Sugerencias	1.917	4261-Sugerencias, 2094-Sugerencias, 4744-Sugerencias, 4087-Sugerencias
3	Solicitudes	261	4252-Solicitud de concepto, 640-Solicitud
4	Transferencias	46	633-Transferencia electrónica
5	Inspecciones	599	38-Inspección del vehículo, 41-Inspección del vehículo
6	Revisiones	443	39-Revisión Preventiva de Seguridad - RPS
7	Peticiones	26	97-Petición
8	Actas	297	2-Acta Desintegración Vehículos de Transporte Público Colectivo e Individual y Particular

Modelamiento

La fase de modelamiento se realizó en dos pasos importantes, en el primero se utilizó RapidMiner como apoyo al prototipado y la evaluación rápida de modelos, y luego en el segundo paso se utilizó Python con las librerías NLTK y Scikit-Learn, buscando obtener los modelos definitivos que pudieran ir a despliegue.

En el primer paso, haciendo optimización de parámetros, se crearon varios modelos con un árbol de decisión (Decision Tree, DT) (Kowsari *et al.*, 2019) y con el algoritmo K-NN (Rasjid y Setiawan, 2017) a través de en un flujo de trabajo en RapidMiner, recibiendo como entrada la matriz de términos por documentos (basada en TF-IDF), creada previamente también con un proceso de RapidMiner. La evaluación de los modelos se realizó con validación cruzada de cinco folders. Los mejores resultados obtenidos se presentan al lado izquierdo de la Tabla 2, donde se observa en la parte superior el valor de exactitud (accuracy) para cada algoritmo y luego el valor de recuerdo (recall) para cada clase, esto último con el objetivo de analizar el efecto del desbalanceo de las clases sobre cada clasificador. En rojo se muestran los valores de recuerdo más bajos por clase y en naranja los valores mayores o iguales a 60% y menores a 90%. A pesar de que K-NN es más lento en la fase de predicción, sus resultados de exactitud a nivel global y de recuerdo por clase son muy competitivos y en la clase "Solicitudes" son mejores, aunque igualmente desfavorables. Es preciso comentar que los resultados obtenidos con K-NN concuerdan con la literatura en el hecho de que la similitud de cosenos (cosine similarity) es una de las mejores opciones para comparar documentos de texto.

Teniendo en cuenta que con el DT se pudieron determinar los atributos más significativos del dataset de entrenamiento, con estos atributos se definió un vocabulario controlado y se incluyó en el proceso de creación de la matriz de términos por documento. La nueva matriz denominada vista minable "reducida" tan solo conservó 427 atributos de los 14.004 originales. Luego, se realizó nuevamente la optimización de parámetros de DT y K-NN usando la vista minable reducida y validación cruzada de cinco folders y los resultados obtenidos se pueden observar al lado derecho de la Tabla 2. En general los valores de exactitud y recuerdo por clase disminuyeron, hecho que se atribuye al cambio de los pesos en el momento del cálculo de la matriz TF-IDF, por lo cual se desechó la realización de otros procesos de selección de atributos, ya que se presentaría la misma situación. Por otro lado, realizar el proceso de selección de atributos incluyendo el proceso de creación de la matriz TF-IDF resulta muy costoso computacionalmente e inconveniente para el servicio que se ofrecería a los usuarios del SGD GFiles.

Para manejar el desbalance de clases se identificaron tres alternativas, a saber: 1) creación de datos de las clases minoritarias, buscando contar con un número similar de instancias (datos o ejemplos) al que presentan las clases mayoritarias; 2) utilizar un enfoque de aprendizaje basado en costos (metaclasificador) en el que se establece una matriz de costos basada en la relación que existe de equivocarse prediciendo una instancia de una clase mientras que realmente es de otra; y 3) sacar los representantes (prototipos) de las clases que poseen un mayor número de ejemplos (clases mayoritarias). De esta forma se reduce el número de instancias y se equiparan con las clases minoritarias.

La primera opción (creación de datos con SMOTE (Fernández-Navarro, Hervás-Martínez y Gutiérrez, 2011) o similares) no se usó debido a que se pueden crear sesgos o acentuar algunos rasgos de los pocos datos que ya se tienen. La segunda opción (definición de costos manualmente, o automáticamente con SPIDER3 (Wojciechowski, Wilk y Stefanowski, 2018) o similares) también fue descartada dado que el número de clases en el problema que se va a modelar crece dinámicamente con el tiempo. Tampoco se consideró apropiado definir los costos basados en las frecuencias de los documentos por clase y la posible asignación errónea

a otras clases. Por esto, se eligió la tercera opción, seleccionar de las clases mayoritarias los ejemplos más representativos (prototipos), logrando así reducir su número y disminuir la diferencia entre las cantidades que poseen las clases documentales. Para realizar este proceso se usaron los algoritmos CNN (Condensed Nearest Neighbor) (Gowda y Krishna, 1979) y RMHC (Random Mutation Hill Climbing) (Skalak, 1994) con los mismos clasificadores previamente evaluados (DT y K-NN). Los resultados del anterior proceso se observan en la Tabla 3. Como se puede ver, los porcentajes de exactitud general obtenidos por los modelos son altos (entre 87,42% y 92,03%). Sin embargo, al revisar el valor de recuerdo por clase, se puede observar que la clase "Solicitudes" sigue presentando valores muy bajos, en las 4 pruebas realizadas no ha superado el 30% y en varias ocasiones no se identificó ninguno de los documentos de esta clase (0,00%). Otras evaluaciones con un nuevo selector de prototipos, ELH (Encoding Length Heuristic) (Cameron-Jones, 1995), y la optimización de parámetros de CNN, mostraron resultados muy similares a los presentados en esta tabla.

Tabla 2. Resultados de los algoritmos Decision Tree (DT) y K-NN en RapidMiner

	Resultados con vista minable original		**Resultados con vista minable reducida**	
Algoritmo	DT	K-NN	DT	K-NN
Parámetros	Gain Ratio Maximal Depth = 10	k = 5 Cosine Similarity	Gain Ratio Maximal Depth = 10	k = 5 Cosine Similarity
Exactitud	91,70%	92,03%	91,18%	87,20%
Valor de recuerdo por clase				
Comunicaciones	72,00%	97,33%	60,00%	86,67%
Solicitudes	0,00%	27,97%	0,00%	31,80%
Actas	100,00%	100,00%	100,00%	100,00%
Revisiones	99,77%	99,77%	99,55%	99,77%
Sugerencias	99,48%	95,31%	99,48%	86,38%
Inspecciones	99,17%	99,17%	99,17%	99,17%
Peticiones	92,31%	100,00%	65,38%	76,92%
Transferencias	91,30%	86,96%	86,96%	82,61%

Tabla 3. Resultados usando selectores de instancias con DT y K-NN en la vista minable original

	Selector CNN		**Selector RMHC**	
Algoritmo	DT	K-NN	DT	K-NN
Parámetros	Gain Ratio Maximal Depth = 10	k = 5 Cosine Similarity	Gain Ratio Maximal Depth = 10	k = 5 Cosine Similarity
Exactitud	91,70%	92,03%	91,18%	87,20%
Valor de recuerdo por clase				
Comunicaciones	77,33%	97,33%	89.33%	96,00%
Solicitudes	8,05%	29,89%	0,00%	26,82%
Actas	100,00%	100,00%	100,00%	100,00%
Revisiones	99,77%	99,87%	60,05%	99,77%
Sugerencias	98,07%	95,04%	99,74%	95,04%
Inspecciones	99,17%	99,17%	99,67%	99,17%
Peticiones	84,62%	100,00%	92,31%	100,00%
Transferencias	86,96%	86,96%	86,96%	86,96%

Con base en los resultados obtenidos hasta el momento, se evidencia que la exactitud general es una buena medida del comportamiento del clasificador, pero no debe ser la única medida que se debe tener en cuenta, ya que al contar con clases bajamente representadas (pocos registros) que tienen un muy bajo valor de recuerdo, este valor bajo se oculta en los altos valores reportados por la exactitud general. Este hecho debe ser tenido en cuenta al momento de hacer el despliegue de la solución en la plataforma GFiles, en dado caso que no se logre obtener niveles aceptables en el valor de recuerdo para algunas de las clases. Esta situación también planteó otra inquietud al equipo sobre cómo manejar documentos que pertenecen a clases muy generalistas, que manejan muchos temas trasversales al negocio, es decir, clases como "Solicitudes", en las que se puede solicitar atender una radicación, o solicitar que se acate una sugerencia, u otras similares, tema que sigue abierto a investigación.

Es preciso destacar que en relación con la clase "Solicitudes", los resultados de K-NN tuvieron un valor de recuerdo alrededor del 30% a diferencia de aquellas que utilizaron DT. Además, a nivel general, K-NN obtuvo porcentajes levemente mayores o competitivos de recuerdo en todas las clases en comparación con los obtenidos por DT.

En el segundo paso y teniendo en cuenta que el despliegue de la solución (de obtenerse buenos resultados) se realizaría en Python a manera de un microservicio que se pudiera llamar desde GFiles, se procedió a implementar todo el proceso (captura, procesamiento y generación de modelos) previamente realizado en RapidMiner en este lenguaje. Como era de esperarse, los resultados presentaron variaciones, debido a que las implementaciones de los algoritmos varían de una librería a otra. En la fase de procesamiento de los documentos para la construcción de la vista minable se usó la clase TfidfVectorizer de Scikit-Learn y se incluyó un límite para el número máximo y mínimo de caracteres permitidos en los tokens con el objetivo de obtener resultados más similares a los obtenidos con RapidMiner.

En Python, además de DT y K-NN, se incluyeron los Bosques Aleatorios (Random Forest, RT), perceptrón multicapa (MLP) y Naïve Bayes basados en la clase GaussianNB de la librería de Scikit-Learn. Los resultados de la ejecución de los cinco clasificadores en la vista minable original se presentan en la Tabla 4. La exactitud de los clasificadores está entre 88,26% (DT) y 91,18% (MLP). Los valores de recuerdo para los tres mejores clasificadores (MLP, RF y K-NN) son superiores al 80% en todas las clases, excepto "Solicitudes", que está entre 21,84% y 35,63%, lo que contrasta con los resultados de los clasificadores DT y NB que para esta clase tienen resultados entre un 39,08% y 40,99%, castigando el recuerdo en otras clases. Con base en los resultados generados hasta el momento, se puede evidenciar que el mejor algoritmo es el MLP. Además, la clase "Solicitudes" presenta valores bajos de recuerdo en todos los modelos (similar a lo obtenido en RapidMiner). En este paso también se realizó un proceso de selección de atributos y los resultados fueron similares o de menor calidad en algunos clasificadores por lo que se siguió trabajando con la vista minable original y se descartó la reducida.

Evaluación

Los resultados obtenidos en la anterior etapa evidenciaron que los modelos construidos obtuvieron valores similares de exactitud entre el 88% y el 91% con la vista minable original y similares problemas con el recuerdo para una clase, excepto Naïve Bayes, que tuvo problemas con el recuerdo en dos clases. En este punto fue difícil seleccionar un único algoritmo como el claro vencedor en el proceso de clasificación de documentos y teniendo en cuenta que cada cliente (organización o empresa) puede tener documentos con características diferentes, se decidió ofertar el servicio con la posibilidad de que sea el mismo cliente quien selecciona el mejor modelo para su contexto.

En este sentido, se tomaron los datos de las dos organizaciones (índice_1 e índice_2) por separado, se crearon las vistas minables de cada una y se ejecutaron los modelos utilizados. Los resultados se pueden observar en la Tabla 5. Para la primera organización, los modelos reportan una exactitud que oscila entre 82% y 86%, y se escoge como mejor modelo el MLP, que no tiene la mejor exactitud (muy cercana a la mejor obtenida por RF) pero el recuerdo por clase es en general mejor, con lo que se logra un mejor balance entre exactitud y recuerdo. Por otro lado, el hecho de que el problema de recuerdo persista en la clase "Solicitudes" involucra que el modelo al pasar a producción debe establecer un proceso para advertir al usuario y recolectar información que le pueda servir en un entrenamiento futuro para mejorar su rendimiento, esto último se incluye como una historia de usuario en el desarrollo del servicio.

Los resultados de la segunda organización evidencian un alto valor de exactitud y de recuerdo por clase. Los valores son superiores a 90% en todos los casos y, a pesar de que los tres mejores son MLP, RF y K-NN, sería apropiado seleccionar RF o MLP en lugar de K-NN, buscando tener una mejor velocidad de respuesta. Por otro lado, dado que los modelos de RF y MLP se pueden volver obsoletos, se define una historia de usuario en el desarrollo del servicio para monitorear el rendimiento y hacer el registro de datos para entrenamiento. Estos resultados permiten afirmar que el cliente puede pasar a producción uno de estos modelos con entera confianza.

Finalmente y buscado evaluar y comparar los resultados obtenidos con uno de los modelos de representación de documento más actuales y de más reconocimiento en la actualidad, BERT de Google, se aplicó una versión pre-entrenada en español denominada BETO (Cañete *et al.*, 2020) para representar los documentos de las dos organizaciones y armar la vista minable. Los documentos fueron tratados de la siguiente manera: el texto extraído fue convertido en minúscula, se removieron los signos de puntuación y, además, se omitieron las palabras vacías. Teniendo en cuenta que BETO procesa textos con una longitud máxima de 512 tokens y que el tokenizador que acompaña al modelo puede generar tokens adicionales en el proceso, se procedió a dividir cada documento en fragmentos que no superaran dicho límite. Para lo anterior, cada documento fue divido en partes de 150 palabras como máximo y, una vez procesado cada documento, se procedió a escoger el primer fragmento de cada documento para posteriormente ejecutarlo en BETO y obtener su representación.

Tabla 4. Resultados de todos los clasificadores en Python con la vista minable original

Algoritmo	**MLP**	**RF**		**K-NN**		**NB**	**DT**
Parámetros	N/A	Criterion gini	Criterion entropy	k = 5, Euclidian distance weights=distance	k = 5, Manhattan distance weights=distance	N/A	Criterion entropy
Exactitud	91,18%	91,07%	90,97%	90,56%	89,74%	88,51%	88,26%
Valor de recuerdo por clase							
Comunicaciones	93,33%	92%	92%	92,00%	94,67%	45,33%	89,33%
Solicitudes	35,63%	22,60%	21,84%	27,20%	30,65%	40,99%	39,08%
Actas	99,66%	99,66%	99,66%	99,66%	99,66%	99,66%	98,65%
Revisiones	99,55%	99,55%	99,55%	99,55%	99,55%	99,55%	97,52%
Sugerencias	93,01%	94,73%	94,63%	93,22%	90,72%	89,51%	87,85%
Inspecciones	98,99%	98,99%	98,99%	99,00%	99,00%	99,16%	98,83%
Peticiones	100%	100%	100%	92,31%	100,00%	84,61%	92,31%
Transferencias	84,78%	80,43%	80,43%	80,44%	91,30%	71,74%	86,96%

La Tabla 6 muestra los resultados obtenidos con BETO. Se puede observar que para la primera organización los resultados de exactitud son similares a los obtenidos con la vista minable original para MLP, RF y K-NN, pero decaen notoriamente para NB y DT. También se observa un menor recuerdo en todos los algoritmos con esta representación. Para la segunda organización se evidencian valores comparables, aunque un poco más bajos en exactitud en relación con la vista minable original. Los valores de recuerdo también fueron altos y mayores al 90%, a excepción de los obtenidos por DT con las clases documentales "Inspecciones II" y "Transferencias", y con Random Forest con la clase "Transferencias". En resumen, el servicio pasa a despliegue con los algoritmos evaluados y debe permitir que cada cliente (organización) defina su propio modelo. Se deben definir esquemas para monitorear el rendimiento de los modelos y capturar datos que permitan enriquecer futuros procesos de entrenamiento.

Despliegue

A partir de los resultados obtenidos, se estableció un servicio de clasificación a utilizar en conjunto con el SGD GFiles. La Figura 1 muestra la interacción de los componentes del servicio. En este sentido, cada

Tabla 5. Resultados obtenidos con las dos organizaciones usando las vistas minables creadas

Organización	Algoritmo	MLP	RF	K-NN	NB	DT
Índice_1	Exactitud	85,96%	86,58%	86,19%	83,84%	82,32%
	Valor de recuerdo por clase					
	Comunicaciones	100%	100%	100%	97,36%	77,01%
	Solicitudes	39,24%	38,75%	38,65%	37,16%	33,23%
	Sugerencias	91,15%	89,91%	90,35%	90,19%	91,43%
Índice_2	Exactitud	99,36%	99,36%	99,36%	99,14%	97,94%
	Valor de recuerdo por clase					
	Actas	100%	100%	100%	99%	99,66%
	Revisiones	98,87%	98,65%	98,65%	98,88%	98,15%
	Inspecciones I	99,55%	99,55%	99,55%	99,77%	96,32%
	Inspecciones II	99,31%	99,31%	99,31%	97,95%	99,31%
	Peticiones	100%	100%	100%	100%	100%
	Transferencias	96,29%	100%	100%	100%	92,85%

Tabla 6. Resultados obtenidos con las dos organizaciones usando la vista minable basada en BETO

Organización	Algoritmo	MLP	RF	K-NN	NB	DT
Índice_1	Exactitud	85,30%	85,87%	84,72%	70,73%	76,68%
	Valor de recuerdo por clase					
	Comunicaciones	60%	21,33%	56%	62,66%	33,33%
	Solicitudes	30,38%	6,92%	33,84%	65%	30,38%
	Sugerencias	93,74%	99,11%	92,74%	71,83%	84,66%
Índice_2	Exactitud	99,07%	98,65%	98,93%	95,81%	94,04%
	Valor de recuerdo por clase					
	Actas	99,66%	99,66%	100,00%	94,27%	96,29%
	Revisiones	99,54%	99,54%	99,54%	96,83%	93,45%
	Inspecciones I	98,68%	98,46%	98,46%	95,60%	97,58%
	Inspecciones II	99,30%	99,30%	99,30%	95,83%	84,02%
	Peticiones	92,30%	92,30%	92,30%	92,30%	92,30%
	Transferencias	97,82%	86,95%	93,47%	100,00%	82,60%

cliente de GFiles inicia el proceso de creación del modelo de clasificación con la recuperación de los documentos que están indexados en ElasticSearch, especialmente el texto extraído de los documentos junto con el tipo documental con el que se registró. Después, se realiza el proceso de preparación (preprocesamiento) de los datos implementado durante el modelado. Una vez preparados los datos, se procede a construir la matriz de términos, utilizando la técnica TF-IDF que junto con la clase documental (definida a partir de la agrupación de los tipos documentales relacionados) se usa para generar la vista minable de entrenamiento, la validación de los modelos y la selección del mejor modelo. Para realizar este proceso se usa la librería Scikit-Learn de Python. Teniendo en cuenta que este proceso puede requerir mucho tiempo de ejecución, se utiliza la librería Celery y el bróker de mensajes RabbitMQ para ejecutarlo en segundo plano. Los resultados generados en los anteriores procesos se registran en la base de datos para poder realizar un seguimiento a las acciones ejecutadas. Al comparar el rendimiento de los modelos y seleccionar el mejor se marca para que este pase a producción por la empresa cliente en GFiles.

Los pasos anteriormente mencionados pertenecen a funcionalidades de un API REST basada en microservicios, que permiten acoplarse adecuadamente con GFiles sin generar cambios a nivel de la arquitectura de este u otro SGD. A continuación, se resume cada uno de los microservicios desarrollados:

1. ***Registrar organización***: crea una estructura de directorios para la organización donde se almacenarán los documentos, datasets y modelos generados. Ya que GFiles trabaja con múltiples clientes, los microservicios se diseñaron de la misma forma.
2. ***Subir archivos***: carga los documentos a utilizar para la creación de la vista minable sobre la que se realiza el entrenamiento y validación de los modelos.
3. ***Crear matriz***: crea la representación TF-IDF de los documentos cargados que, junto con la clase documental, se convierte en la vista minable.
4. ***Obtener clasificadores***: lista los algoritmos clasificadores disponibles para el entrenamiento del modelo.
5. ***Entrenar modelo***: crea un modelo de datos utilizando la vista minable y el algoritmo clasificador seleccionado por el usuario. Este microservicio usa validación cruzada.
6. ***Obtener resultado***: retorna el resultado de las métricas que sirven para decidir cuál modelo desplegar: exactitud a nivel general, y precisión, recuerdo y medida F1 por cada clase, y la matriz de confusión.
7. ***Seleccionar modelo***: establece (marca) el modelo que será utilizado por el sistema de gestión documental para la clasificación de los documentos de la organización cliente.
8. ***Predecir archivo***: utiliza el modelo desplegado para predecir la clase documental de un documento.

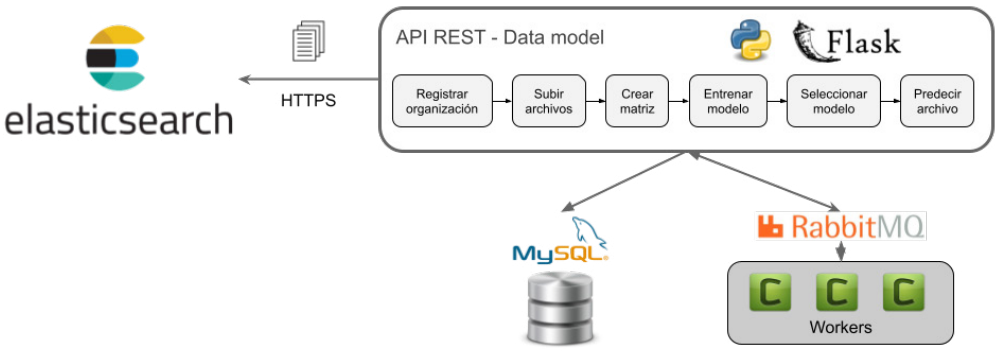

Figura 1. Modelo de clasificación documental

Es importante mencionar que todos los microservicios a excepción de "Predecir archivo" serán ejecutados principalmente por un administrador de analítica de la organización cliente (o asesor de GFiles provisto por Nexura), debido a que el proceso de entrenamiento y evaluación de los algoritmos requiere de conocimientos en esta área para ejecutarlos apropiadamente.

Durante el uso del servicio es preciso manejar ciertas situaciones. La primera de ellas se presenta cuando se añade una nueva clase en el proceso de entrenamiento y validación. Para este caso, se establece un esquema de actualización que consiste en entrenar un nuevo modelo con los datos actualizados y, posteriormente, decidir cuál modelo se debe desplegar. Durante este proceso se debe tener en cuenta que los nuevos datos pueden venir con ruido o estar mal clasificados, lo cual afecta la calidad del nuevo modelo. Para evitar esta situación, el microservicio de entrenamiento permite construir modelos en un directorio diferente al utilizado por el modelo desplegado, evitando así que se afecte su funcionamiento. Además, los microservicios "Obtener resultado" y "Seleccionar modelo" permiten al usuario evaluar los resultados del nuevo modelo antes de decidir cuál modelo emplear en el SGD. Una vez se selecciona un nuevo modelo, se elimina el que previamente se había desplegado.

Una segunda y muy importante situación se relaciona con el monitoreo del modelo para decidir el momento en que el modelo debe actualizarse debido a la pérdida de calidad en sus resultados. Para lo anterior, se registra la retroalimentación que dan los usuarios y se calculan estadísticas de su rendimiento en producción. Cada vez que el usuario carga un documento, el modelo predice la clase documental y este valor con el tipo seleccionado por el usuario se almacena para conocer aciertos y posibles errores en predicción. Con base en los anteriores datos, el administrador de la organización cliente podrá saber qué porcentaje de los registros ha fallado en la predicción de cada clase documental y con esto decidir si se requiere o no la actualización del modelo. Cada cliente decide la forma de seleccionar el modelo, aunque se recomienda usar en forma conjunta la exactitud general del modelo y el recuerdo reportado por cada clase.

Conclusiones

En este trabajo se realizó un proceso de clasificación documental basado en diferentes modelos de aprendizaje de máquina, perceptrón multicapa, bosques aleatorios, k vecinos más cercanos, arboles de decisión y clasificadores bayesianos ingenuos con datos de dos organizaciones cliente del sistema de gestión documental GFiles. El esquema de representación usado se basó en una bolsa de palabras con n-gramas acumulativos (unigramas, bigramas y trigramas).

El perceptrón multicapa, los bosques aleatorios y los k vecinos más cercanos obtienen los mejores resultados medidos en exactitud general (alrededor del 90%) del modelo y recuerdo por cada clase. Estos resultados son muy similares y hace difícil seleccionar un modelo sobre otro. Se recomienda el uso de los dos primeros, ya que son más rápidos en producción que el modelo de los k vecinos más cercanos, en especial cuando los datos de entrenamiento tienen un alto volumen.

Los resultados de la evaluación y la comparación con BETO (red neuronal profunda BERT pre-entrenada para español) muestran que la representación por bolsa de palabras basada en n-gramas acumulativos obtiene mejores resultados con los clasificadores evaluados. Los proponentes de BERT deben eliminar la limitación que este tiene para representar totalmente el documento, la actual cantidad limitada de tokens que representan un documento hace que el modelo pierda información valiosa.

El proceso de selección de prototipos, realizado con CNN, RMHC y ELH, no sirvió como estrategia de mejora en el recuerdo de la clase "Solicitudes", que reportó problemas en los experimentos originales, además su alto costo computacional limitó su uso en la fase de despliegue.

En relación con el despliegue, no se define un modelo único para todos los clientes de GFiles, sino que se ofrece a cada cliente la funcionalidad requerida para cargar sus documentos, tipos documentales, homologación de clases documentales, creación de la vista minable, entrenamiento, validación y comparación de varios modelos, selección del mejor y uso de este por parte de sus usuarios específicos.

La implementación desarrollada del servicio (basada en una arquitectura de microservicios) permitió que fuera integrada en GFiles sin realizar modificaciones importantes en el sistema gestor documental, lo que además lo habilita para su uso en otros SGD. Como trabajo futuro se espera mejorar el proceso de entrenamiento mediante la inclusión de otros clasificadores y técnicas para reducir el número de atributos de manera que permita aumentar la velocidad en la creación del modelo sin perder la calidad en sus resultados.

Agradecimientos

El presente trabajo fue parcialmente financiado por el Ministerio de Ciencia Tecnología e Innovación - MinCiencias (Colombia) a través del proyecto 72870 realizado por la empresa Nexura Internacional S.A.S. y la Universidad del Cauca.

Referencias

Aliwy, A. H., Ameer, E. A. (2017). Comparative study of five text classification algorithms with their improvements. *International Journal of Applied Engineering Research*, *12*(14), 4309-4319

Cameron-Jones, R. M. (1995). Instance selection by encoding length heuristic with random mutation hill climbing. *Eighth Australian Joint Conference on Artificial Intelligence, Canberra*, 99-106

Cañete, J., Chaperon, G., Fuentes, R., Ho, J.-H., Kang, H., Pérez, J. (2020). Spanish pre-trained BERT model and evaluation data. En *PML4DC, ICLR*, 1-10

Cao, Z., Zhou, Y., Yang, A., Fu, J. (2019). Contextualized Word Representations with Effective Attention for Aspect-Based Sentiment Analysis. En M. Sun, X. Huang, H. Ji, Z. Liu, & Y. Liu (Eds.), *Chinese Computational Linguistics* (pp. 467-478). Springer. https://doi.org/10.1007/978-3-030-32381-3_38

Chen, J., Yan, S., Wong, K.-C. (2020). Verbal aggression detection on Twitter comments: Convolutional neural network for short-text sentiment analysis. *Neural Computing and Applications*, *32*(15), 10809-10818. https://doi.org/10.1007/s00521-018-3442-0

Chen, T., Xu, R., He, Y., Wang, X. (2017). Improving sentiment analysis via sentence type classification using BiLSTM-CRF and CNN. *Expert Systems with Applications*, *72*, 221-230. https://doi.org/10.1016/j.eswa.2016.10.065

Devlin, J., Chang, M.-W., Lee, K., Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. En *Proceedings of the 2019 Conference of the North {A}merican Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 4171-4186. https://doi.org/10.18653/v1/N19-1423

Dorado, H., Cobos, C., Torres-Jimenez, J., Burra, D. D., Mendoza, M., Jimenez, D. (2019). Wrapper for building classification models using covering arrays. *IEEE Access*, *7*, 148297-148312. https://doi.org/10.1109/ACCESS.2019.2944641

Fernández-Navarro, F., Hervás-Martínez, C., Gutiérrez, P. A. (2011). A dynamic over-sampling procedure based on sensitivity for multi-class problems. *Pattern Recognition*, *44*(8), 1821-1833. https://doi.org/10.1016/j.patcog.2011.02.019

Gitanjali, Lakhwani, K. (2019). A novel approach of sensitive data classification using convolution neural network and logistic regression. *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, *8*(8), 2883-2886

Gowda, K., Krishna, G. (1979). The condensed nearest neighbor rule using the concept of mutual nearest neighborhood. *IEEE Transactions on Information Theory*, *25*(4), 488-490. https://doi.org/10.1109/TIT.1979.1056066

Hapsari, D. P., Utoyo, I., Purnami, S. W. (2020). Text categorization with fractional gradient descent

support vector machine. *Journal of Physics: Conference Series*, *1477*, e 022038. https://doi.org/10.1088/1742-6596/1477/2/022038

Ismael, A., Okumus, I. (2017). Design and implementation of an electronic document management system. *Journal of Applied Sciences of Mehmet Akif Ersoy University*, *1*(1), 9-17. https://doi.org/10.31200/makuubd.321093

Jiang, M., Liang, Y., Feng, X., Fan, X., Pei, Z., Xue, Y., Guan, R. (2018). Text classification based on deep belief network and softmax regression. *Neural Computing and Applications*, *29*(1), 61-70. https://doi.org/10.1007/s00521-016-2401-x

Kowsari, K., Brown, D. E., Heidarysafa, M., Jafari Meimandi, K., Gerber, M. S., Barnes, L. E. (2017). HDLTex: Hierarchical deep learning for text classification. En *16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 364-371. https://doi.org/10.1109/ICMLA.2017.0-134

Kowsari, K., Meimandi, K. J., Heidarysafa, M., Mendu, S., Barnes, L., Brown, D. (2019). Text classification algorithms: A survey. *Information*, *10*(4), e150. https://doi.org/10.3390/info10040150

Lacunza, A. C. (2020). *Implementación de un Sistema de Gestión Documental Electrónico en la Universidad Nacional de la Plata: El camino hacia el expediente electrónico* [Tesis de Maestría]. Universidad Nacional de la Plata, Argentina. http://sedici.unlp.edu.ar/bitstream/handle/10915/115287/Documento_completo.pdf

Lagrari, F.-E., Ziyati, H., El Kettani, Y. (2019). An efficient model of text categorization based on feature selection and random forests: Case for Business documents. En M. Ezziyyani (Ed.), *Advanced Intelligent Systems for Sustainable Development (AI2SD'2018)* (pp. 465-476). Springer. https://doi.org/10.1007/978-3-030-11928-7_42

Qin, W., Guo, W., Liu, X., Zhao, H. (2019). A novel scheme for recruitment text categorization based on KNN algorithm. En M. Qiu (Ed.), *SmartCom 2019: Smart Computing and Communication* (pp. 376-386). Springer. https://doi.org/10.1007/978-3-030-34139-8_38

Qu, Z., Song, X., Zheng, S., Wang, X., Song, X., Li, Z. (2018). Improved bayes method based on TF-IDF feature and grade factor feature for Chinese information classification. En: *IEEE International Conference on Big Data and Smart Computing (BigComp)*, 677-680. https://doi.org/10.1109/BigComp.2018.00124

Rangel Palencia, E. L. (2017). *Guía de Implementación de Un Sistema de Gestión de Documentos Electrónicos de Archivo - SGDEA*. Archivo General de la Nación de Colombia. https://www.archivogeneral.gov.co/caja_de_herramientas/docs/12._herramientas/DT - IMPLEMENTACION DE UN SGDEA.pdf

Rasjid, Z. E., Setiawan, R. (2017). Performance comparison and optimization of text document classification using k-NN and naïve bayes classification techniques. *Procedia Computer Science*, *116*, 107-112. https://doi.org/10.1016/j.procs.2017.10.017

Rodríguez Cruz, Y., Castellanos Crespo, A., Ramírez Peña, Z. (2016). Gestión documental, de información, del conocimiento e inteligencia organizacional: particularidades y convergencia para la toma de decisiones estratégicas. *Revista Cubana de Información en Ciencias de la Salud*, *27*(2), e206224

Schröer, C., Kruse, F., Gómez, J. M. (2021). A systematic literature review on applying CRISP-DM process model. *Procedia Computer Science*, *181*, 526-534. https://doi.org/10.1016/j.procs.2021.01.199

Selvi, S. T., Karthikeyan, P., Vincent, A., Abinaya, V., Neeraja, G., Deepika, R. (2017). Text categorization using Rocchio algorithm and random forest algorithm. En *Eighth International Conference on Advanced Computing (ICoAC)*, 7-12. https://doi.org/10.1109/ICoAC.2017.7951736

Shah, N., Willick, D., Mago, V. (2018). A framework for social media data analytics using Elasticsearch and Kibana. *Wireless Networks, 2018*. https://doi.org/10.1007/s11276-018-01896-2

Skalak, D. B. (1994). Prototype and feature selection by sampling and random mutation hill climbing algorithms. En W. W. Cohen, & H. Hirsh (Eds.), *Machine Learning: Proceedings of the Eleventh International Conference*. Morgan Kaufmann. https://doi.org/10.1016/b978-1-55860-335-6.50043-x

Taloba, A. I., Ismail, S. S. I. (2019). An intelligent hybrid technique of decision tree and genetic algorithm for e-mail spam detection. En *Ninth International Conference on Intelligent Computing and Information Systems (ICICIS)*, 99-104. https://doi.org/10.1109/ICICIS46948.2019.9014756

Vijayan, V. K., Bindu, K. R., Parameswaran, L. (2017). A comprehensive study of text classification algorithms. En *International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 1109-1113. https://doi.org/10.1109/ICACCI.2017.8125990

Villegas, J., Cobos, C., Mendoza, M. E., Herrera-Viedma, E. (2018). Feature selection using sampling with replacement, covering arrays and rule-induction techniques to aid polarity detection in twitter sentiment analysis. *Lecture Notes in Computer Science*, *11238*, 467-480. https://doi.org/10.1007/978-3-030-03928-8_38

Voit, A., Stankus, A., Magomedov, S., Ivanova, I. (2017). Big data processing for full-text search and visualization with Elasticsearch. *International Journal of Advanced Computer Science and Applications*, *8*(12), e11. https://doi.org/10.14569/IJACSA.2017.081211

Wirth, R., Hipp, J. (2000). CRISP-DM: Towards a standard process model for data mining. En *Proceedings of the Fourth International Conference on the Practical Application of Knowledge Discovery and Data Mining*, 29-39

Wojciechowski, S., Wilk, S., Stefanowski, J. (2018). An algorithm for selective preprocessing of multi-class imbalanced data. En *Proceedings of the 10th International Conference on Computer Recognition Systems CORES 2017*. In M. Kurzynski, M. Wozniak, & R. Burduk (Eds.) (pp. 238-247). Springer.

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., Hovy, E. (2016). Hierarchical attention networks for document classification. En *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1480-1489. https://doi.org/10.18653/v1/N16-1174

Zamfir, V.-A., Carabas, M., Carabas, C., Tapus, N. (2019). Systems monitoring and big data analysis using the elasticsearch system. En *22nd International Conference on Control Systems and Computer Science (CSCS)*, 188-193. https://doi.org/10.1109/CSCS.2019.00039