Clasificación automática de documentos basado en un enfoque de recuperación de información

Juan Camilo Forero Vélez

Director: PhD. Carlos Alberto Cobos Lozada

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Sistemas
Programa Ingeniería de Sistemas
Grupo de Investigación y Desarrollo en Tecnologías de la Información (GTI)
Línea Investigación: Gestión de la Información
Popayán, marzo de 2022

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

RESUMEN

Cuando una persona necesita realizar un trámite por lo general son requeridos documentos necesarios para que la empresa tramitadora pueda ejecutar la petición del usuario. Estos documentos son recibidos y verificados por la empresa, muchas veces por un funcionario de forma manual, en caso de que los documentos estén incompletos el trámite es rechazado y el usuario debe volver a presentar todo para volver a empezar. Este proceso es ineficiente para la empresa ya que debe dedicar tiempo a revisar la documentación, y por parte del cliente que quiere hacer el trámite porque pierde tiempo solo esperando una primera respuesta.

En el presente trabajo se logró plantear un método de clasificación automática de documentos basado en un enfoque de recuperación de información, que es capaz de operar con poca información de entrenamiento y que las clases de documentos pueden incrementar con el tiempo sin afectar mucho los resultados. Luego se evaluó con documentos reales, calificándolo con métricas de minería de datos. Basado en el método propuesto, se desarrolló también una aplicación web con arquitectura de microservicios, la cual permite realizar el entrenamiento y verificación de modelos utilizando poca información de documentos. Al final del desarrollo de la aplicación, esta fue evaluada por expertos en el tema de clasificación de documentos, de quienes se recibió retroalimentación sobre mejoras al modelo y a la aplicación web. Los resultados obtenidos del método junto a la aplicación web denotan que es una solución muy útil y que aún existe un gran potencial de mejora del método propuesto.

CAPÍTULO 1

1 INTRODUCCIÓN

1.1 PLANTEAMIENTO DEL PROBLEMA

En el día a día, la gente alrededor del mundo necesita realizar trámites (solicitudes, autorizaciones, reclamos u otros) en alguna organización pública, privada o mixta, que están relacionadas con diversas actividades ya sea en el área de la salud, educación, negocios, servicios públicos, servicios bancarios, entre otros. Generalmente para efectuarlos, las organizaciones cuentan con ventanillas únicas, cuyo concepto, entendido desde el punto de vista del ciudadano, es un lugar en el cual se puede presentar cualquier solicitud o trámite de manera virtual o presencial.

No obstante, cuando los trámites se realizan a través de una ventanilla única, es muy común que la verificación de las solicitudes se efectúe en un paso posterior al de su recepción. Si los documentos no están completos o correctamente diligenciados o con enmendaduras, la solicitud será rechazada en un paso posterior durante la ejecución del flujo del proceso [1][2][3], provocándose así un reproceso, ya que se debe devolver la solicitud y el usuario deberá iniciar nuevamente el trámite. Un estudio realizado en 2018 [4], muestra que en promedio una persona en Colombia tarda 7,4 horas en completar un trámite, y además, el 26% de los trámites requieren hasta 3 interacciones o más para su resolución. Así, además del retraso, el reproceso y la insatisfacción del cliente, se suma el costo mismo de la verificación de la documentación nuevamente entregada, costo monetario que asume la organización y costo en tiempo que se adiciona al proceso que busca resolver el trámite al usuario.

Por otro lado, la transformación digital de las organizaciones requiere de una continua innovación y compromiso, más aún cuando se tuvo el escenario de distanciamiento social a raíz de la emergencia sanitaria generada por el coronavirus en el año 2020, que aceleró los planes de transformación digital en las empresas tanto públicas como privadas al virtualizar el contacto con ciudadanos y clientes; al mismo tiempo que, todos los ciudadanos del común, se vieron en la imperiosa necesidad de realizar sus actividades desde casa a través de plataformas virtuales, aumentando significativamente la cantidad de estos trámites y solicitudes [5].

Para evitar estos reprocesos y demoras en los procesos, se buscó que el usuario cuente con una aplicación que le permita señalar el tipo de trámite a realizar, le

muestre los documentos requeridos e indique si los documentos anexados por él para realizar el trámite o solicitud están completos desde el momento en que inicia el proceso. Esto significa que la aplicación debe contar con una base de conocimientos (datos, modelos o ambos) que le señale cuáles son los documentos que se deben anexar para cada tramite o solicitud, cuáles son las características distintivas de los documentos que el usuario debe diligenciar, cuales serían estructurados (es decir, se basan en una plantilla donde se debe llenar solo algunos campos específicos), semiestructurados (que además de una plantilla, hay partes adicionales donde se puede escribir texto libre, como por ejemplo uno o varios párrafos o una sola frase), o desestructurados (que no tienen formato ni orden en los campos que se deben presentar), cuáles son los campos que el usuario debe diligenciar y las características de los datos (campos) que allí se deben incluir.

Para efectuar una aplicación como la descrita previamente se ha empleado un enfoque clásico de clasificación de documentos (document classification) [6][7][8], que consiste principalmente en entrenar modelos (clasificadores) de aprendizaje de máquina (machine learning) con documentos existentes previamente etiquetados (o clasificados), para que el clasificador identifique las características de cada clase y sea capaz de asignarle a un nuevo documento una “clase” a la cual pertenece ese documento. Sin embargo, varios de los modelos más utilizados de machine learning requieren una gran cantidad de información para poder ser entrenados [6] y cada cierto tiempo deben ser reentrenados (nuevos datos de entrenamiento, nuevas clases o tipos de documentos), y los tiempos de solución no se adaptan a las necesidades reales de un entorno empresarial. Las empresas necesitan que este tipo de soluciones se adapten rápidamente a la forma cambiante de los documentos, a la aparición constante de nuevas clases de documentos y con un nivel de calidad apropiado para los usuarios y las características propias de los procesos empresariales.

La recuperación de la información (Information Retrieval, IR) por su parte, es una subárea del procesamiento del lenguaje natural (Natural Language Processing, NLP) que tiene como objetivo recuperar u obtener recursos de un sistema de información (o de un repositorio) que son relevantes para la necesidad de información de un usuario. El proceso de recuperación de información empieza cuando el usuario introduce una consulta en el sistema, esta consulta puede traer varios documentos que estén relacionados en cierto grado y que son ordenados de acuerdo con el nivel de coincidencia con la consulta. Los documentos que son recuperados previamente debieron ser almacenados e indexados. Este proceso de IR puede ser adaptado al proceso de clasificación y validación de documentos, y tiene importantes ventajas, a saber: 1) los textos tipo (o clases) pueden crecer dinámicamente en el tiempo y sus características individuales ser identificadas en la medida que el repositorio de clases va creciendo; 2) Cuando llega un documento nuevo y se necesita definir su clase, se puede tratar el problema como una búsqueda por relevancia frente a las características específicas de los documentos

base (clases) registrados e indexados en la aplicación; 3) El proceso de reentrenamiento de un clasificador normal no se realiza, sino que se van agregando las nuevas clases con sus características de forma manual o con algún proceso automático; entre otras.

Por las razones anteriormente expresadas, surgió la siguiente pregunta de investigación ¿Cuáles serían los componentes principales de un servicio web basado en el enfoque de Recuperación de información (Information Retrieval) que fuese capaz de verificar que la documentación presentada por un usuario estuviese completa, y que consiguiera operar sin la necesidad de una gran cantidad de información previa para entrenamiento y que se pudiera utilizar en diversos tipos de trámites empresariales?

De acuerdo a lo anterior y buscando acortar los tiempos de verificación por parte de la entidad que ejecuta el trámite y darle al usuario la tranquilidad de que el trámite iniciará con los documentos completos, se presenta este método de clasificación automática con poca información, como alternativa de solución que podrá ser utilizado en diversos tipos de trámites, razón por la que cuenta con una implementación que permite personalizar el servicio, dependiendo de aspectos como, tipos de documentos que se vayan a verificar, el diseño del documento, entre otros, considerando también que, al ser un servicio accesible a cualquier usuario, este puede cargar los documentos a la plataforma en diferentes formatos, como imágenes o documentos en pdf [9] utilizando servicios de software ya existentes que realizan la labor de extracción de datos para su procesamiento. Para la construcción y evaluación del método se recolectaron documentos de diferentes clases y dicho método se evaluó usando métricas estándar de minería de datos en forma incremental, esto es, primero con dos clases, luego con tres, y así sucesivamente con el objetivo de evaluar su estabilidad.

1.2 APORTES DEL PROYECTO

Desde el punto de vista investigativo, el aporte se enfocó en la generación de un nuevo conocimiento en el uso de técnicas y conceptos de IR para realizar la tarea de clasificación de documentos con poca cantidad de información para entrenamiento (documentos estructurados y semi estructurados), ya que en la literatura (Scopus, Science Direct, IEEE, ACM y SpringerLink) se encuentran diversos trabajos previos pero elaborados con un gran volumen de información.

Desde el punto de vista de innovación, se desarrolló una aplicación que permite agilizar diferentes tipos de trámites (solicitudes, reclamos, autorizaciones, entre otros), ahorrar costos a las empresas que utilizan el servicio y aumentar la satisfacción de los clientes expresada como la disminución del tiempo para la realización del proceso (trámite).

Desde el punto de vista de desarrollo se adquirió (por parte del autor de la tesis) conocimiento y habilidades en el desarrollo de microservicios con Spring Boot que

se encuentran plasmados en este documento, los que se espera que sirvan de base para futuros trabajos de grado de la Facultad y la región.

1.3 OBJETIVOS

A continuación, se presentan los objetivos tal y como fueron aprobados por el Consejo de Facultad de la Facultad de Ingeniería Electrónica y Telecomunicaciones al inicio del proyecto y la aprobación de una modificación posterior al tercer objetivo específico.

1.3.1 OBJETIVO GENERAL

Proponer un método de clasificación automática de documentos basado en un enfoque de recuperación de información que sea usado por las empresas a través de un servicio web, para reducir los tiempos de verificación de la documentación entregada por un usuario al iniciar un trámite.

1.3.2 OBJETIVOS ESPECÍFICOS

- Definir un método de clasificación automática de documentos basado en un enfoque de recuperación de información usando poca información de entrenamiento y clases que incrementan dinámicamente en el tiempo, utilizando como guía el patrón de investigación iterativa (PII) propuesto por Pratt [30] y evaluando su calidad con métricas estándar de minería de datos (precisión, recuerdo, medida F1) a través de un proceso de validación cruzada, buscando agilizar la verificación de los documentos presentados para soportar un trámite en cualquier empresa.
- Desarrollar una aplicación web con base en el método previamente propuesto, con arquitectura basada en microservicios sobre Spring Boot, una interfaz web basada en Angular, Lucene como herramienta de Information Retrieval y SCRUM como metodología de desarrollo, para ofertar el servicio de verificación de documentos a las empresas.
- Definir el grado de satisfacción de la aplicación web con base en una muestra de usuarios de la empresa ATIX DIGITAL S.A.S, mediante la adaptación y aplicación de la encuesta “Satisfacción del cliente (producto)” proporcionada por encuestafacil.com usando el índice CSAT [10] conforme se expresa en el apartado 9.1.2 de la norma ISO 9001:2015 [11].

1.4 RESULTADOS OBTENIDOS

A continuación, se resumen los resultados principales del presente trabajo de grado:

1. **Monografía de trabajo de grado**: Se refiere al presente documento en el cual se presenta la motivación del problema, el enfoque de solución planteado en el anteproyecto y el estado del arte en el área de clasificación de documentos. Luego, muestra el método de clasificación automática de documentos basado

en un enfoque de IR usando poca información de entrenamiento y clases que incrementan dinámicamente en el tiempo, también presenta la aplicación web desarrollada, la que se basó en microservicios. Por último, muestra los resultados de la evaluación de satisfacción de una muestra de usuarios de ATIX DIGITAL S.A.S., las conclusiones del trabajo y el trabajo futuro que se espera desarrollar.

2. **Método de clasificación propuesto**: Se refiere al método de clasificación automática de documentos propuesto, el cual está basado en un enfoque de IR usando poca información de entrenamiento y clases que incrementan dinámicamente en el tiempo.

3. **Aplicación web**: que permite realizar la clasificación de documentos estructurados o semiestructurados, de la cual se destacan los siguientes productos principales, a saber:
   - **Código fuente**: Hace referencia al código fuente con el que se desarrollaron los componentes de la aplicación, entre los cuales se incluyen TypeScript para el desarrollo del Front-end con Angular, Java para el desarrollo del Back-end con Springboot, Python con el framework Flask para el desarrollo de un microservicio.
   - **Documentación del código de la aplicación**: Hace referencia a la documentación realizada sobre el código y los componentes de la aplicación desarrollada. También guías de instalación para los microservicios, Front-end, bases de datos e índices en Elasticsearch. La documentación de los microservicios fue generada con ayuda de la herramienta Doxygen para la generación automática de documentación.

4. **Artículo**: Un artículo con los resultados del trabajo de grado elaborado en formato IEEE que se espera enviar a evaluación a una revista o evento nacional o internacional indexado.

1.5 ESTRUCTURA DE LA MONOGRAFÍA

A continuación, se describe de manera general el contenido y organización de la presente monografía:

CAPITULO 1: INTRODUCCIÓN: Hace referencia al presente capitulo que introduce el tema de investigación, presenta la pregunta de investigación que originó el trabajo, los aportes realizados con el desarrollo del trabajo de grado, los objetivos (general y específicos) definidos para el proyecto, un breve resumen de los resultados obtenidos y finalmente la organización de la monografía.

CAPITULO 2: CONTEXTO TEÓRICO Y ESTADO DEL ARTE: En este capítulo se presentan los trabajos más recientes en el área de clasificación de documentos y la definición de las fases más comúnmente utilizadas para realizar esta tarea.

CAPITULO 3: MÉTODO DE CLASIFICACIÓN AUTOMÁTICA DE DOCUMENTOS: En este capítulo se presenta el método propuesto para realizar clasificación con pocos documentos y con la capacidad de soportar clases incrementales junto a las verificaciones para cuando se utilizan estas clases incrementales.

CAPITULO 4: APLICACIÓN WEB: En este capítulo se presenta el diseño detallado del sistema desarrollado describiendo la arquitectura de cada componente junto con los aspectos más importantes de su implementación.

CAPITULO 5: RESULTADOS DE EVALUACIÓN DE SATISFACCIÓN: En este capítulo se presenta la evaluación de la satisfacción de los usuarios de la aplicación web desarrollada.

CAPITULO 6: CONCLUSIONES Y TRABAJOS FUTUROS: En este capítulo se presentan las conclusiones obtenidas al finalizar el trabajo de grado e ideas que el grupo de investigación espera realizar como trabajo futuro.

CAPITULO 7: BIBLIOGRAFIA: Este último capítulo contiene las referencias bibliográficas de los artículos y libros consultados para la realización del proyecto.

CAPÍTULO 2

2 CONTEXTO TEÓRICO Y ESTADO DEL ARTE

En la literatura se encuentran trabajos relacionados con la clasificación de textos a partir de imágenes, por ejemplo, en 2019, Lu et al. [12] muestran como extraen texto de imágenes con la ayuda de un OCR, para luego realizar una clasificación de cada una a partir del diseño, color, textura, y otras características; después de realizar la clasificación y haber identificado el tipo de documento, se facilita la extracción de los datos a texto digital a través de un modelo basado en reglas.

También en 2019, Kowsari et al. [7] muestran avances importantes relacionados con la clasificación de textos en los últimos años, en este trabajo se mencionan diferentes enfoques que se pueden aplicar para la clasificación de textos dentro de un documento, es decir, si este se divide por niveles, la clasificación de textos se puede aplicar para obtener las categorías relevantes para los diferentes niveles definidos así: A nivel de un documento completo, de un solo párrafo o porción de documento, de una oración o porción de párrafo o a nivel de una subexpresión dentro de una oración. Kowsari et al. también menciona la forma cómo la mayoría de los sistemas de clasificación de texto y categorización de documentos están compuestos por cuatro fases principales: Extracción de características, reducción de la dimensionalidad, selección del clasificador y evaluación.

La **extracción de características** se divide en dos, a saber:

1. **Preprocesamiento y limpieza de texto**: En general los documentos a trabajar en el área de clasificación de textos se encuentran de manera desestructurada, así que este texto se debe transformar a una forma estructurada para poder utilizarlos como entrada a los clasificadores, además los textos pueden contener muchísimas palabras que tienen poca importancia (palabras vacías o stop words) y por lo tanto generarían “ruido” al momento de realizar una clasificación, así que primero los datos deben ser limpiados. Los métodos más utilizados son la tokenización, eliminación de palabras vacías, convertir a minúsculas o mayúsculas, manejo de palabras de un tipo de jerga, manejo de sinónimos o unificación de términos que representan conceptos similares, remoción de ruido, corrección de ortografía, stemming (reducir a stem) o lematización y ajuste a un vocabulario cerrado si se tiene un área de aplicación muy específica.
2. **Forma de extraer las características**: Una vez limpiados los datos, se busca un método que permita extraer las características más representativas de los

textos. Entre los principales enfoques están la representación de palabras sintácticas, palabras con peso o incrustaciones de palabras (Word embeddings). Dentro de estos métodos, los más comunes son: la técnica de los N-gramas, la bolsa de palabras (Bag-of-Words, BoW), Frecuencia de Término (Term Frequency, TF), Frecuencia de Término por Frecuencia Invertida de Documento (Term Frequency-Inverse Document Frequency, TF-IDF), Word2Vec, Doc2Vec y Global Vectors for Word Representation (GloVe). Aunque estos son los métodos más utilizados en el área de clasificación de textos, existen muchos más [13].

Reducción de la dimensionalidad: Un conjunto de documentos puede contener muchas palabras o tokens (dimensiones) que son únicas, y aunque el preprocesamiento y limpieza del documento ayuda a reducir significativamente ese número, se hace necesario usar técnicas de reducción de la dimensionalidad. Las técnicas más comunes incluyen el Análisis de Componentes Principales (Principal Component Analysis, PCA), el Análisis Discriminante Lineal (Linear Discriminant Analysis, LDA), la Factorización de Matrices No Negativas (Non-negative Matrix Factorization, NMF), la Descomposición en Valores Singulares (Singular Value Decomposition, SVD), la Proyección Aleatoria (Random Projection), los Autoencoders y la Incrustación Vecina Estocástica Distribuida en t (t-distributed Stochastic Neighbor Embedding, t SNE).

Selección de clasificador: Aunque todas las fases son importantes, esta se destaca por implicar un buen entendimiento del problema para hacer una apropiada elección del clasificador. En la literatura se reporta el uso de muchos clasificadores, este aspecto se detalla tres párrafos más adelante.

Evaluaciones: Es la última parte del proceso de clasificación de documentos y es la que permite identificar el desempeño del clasificador y el proceso seguido. El cálculo de la precisión es una de las medidas más utilizadas, pero este no se comporta bien cuando los datos se encuentran desbalanceados. También otras métricas muy usadas son la medida F1 ($F\beta$=1 score), el Coeficiente de Correlación de Matthews (Matthews Correlation Coefficient, MCC), la curva de la Característica Operativa del Receptor (Receiver Operating Characteristics curve, ROC) y el área Bajo la Curva ROC (AUC).

A continuación, se muestran los avances más recientes de clasificación de texto. Los principales métodos vienen desde el machine learning, aunque también existen aportes desde enfoques estadísticos, los cuales han sido utilizados principalmente en problemas de clasificación binaria [6]. En términos de machine learning, la tarea de clasificar textos se puede dividir en tres categorías: clasificación supervisada, no supervisada y semi supervisada.

Modelos supervisados: La clasificación supervisada viene del aprendizaje supervisado de la Inteligencia Artificial, donde un sistema es entrenado y probado

con una base de datos existente antes de realizar el proceso de clasificación real (en el ambiente de producción). Entre las tres clases, este se considera el más costoso debido a que requiere previa intervención humana para asignar las clases correspondientes a los documentos, es decir, etiquetarlos, que lo hace poco factible cuando los datos a entrenar incluyen grandes volúmenes. Entre los modelos más utilizados en el área de clasificación de textos están:

1. **Clasificación de Rocchio**: Modelo propuesto en el año 2017 [14], donde utilizando el algoritmo de Rocchio y los Bosques Aleatorios (Random Forest) logran realizar una categorización eficiente en comparación con otros modelos existentes en el área de clasificación multi clase de textos, a saber: ML-RBC, ML-FRC, ML-KNN, RankSVM y BoosTexter. Para los experimentos, se utilizaron 2 data sets distintos, uno con 800 documentos y 4 categorías distintas, y otro con 6,680 documentos y 7 categorías diferentes.

2. **Regresión logística y redes neuronales**: Propuesto en 2019 [15], es un modelo que combina regresión logística y una red neuronal convolucional, donde la red neuronal refina características extraídas de los documentos para que luego el modelo de regresión logística realice una mejor clasificación, obteniendo así un porcentaje de mejora entre 4-5% en comparación con modelos basados en K-nn, SVM, un híbrido entre K-nn y SVM y redes neuronales. Para los experimentos recolectaron 40,000 textos a través de una REST API.

3. **Naïve Bayes**: En 2018 [16] se propone un método de Bayes más eficiente para clasificación de textos en chino. Este método se soporta en la representación BoW con ponderación TF-IDF que ayuda al modelo de Naïve Bayes y así reduce la suposición natural que hace este clasificador, que consiste en asumir la independencia entre las características (dimensiones) y como resultado obtiene mejor desempeño que otros clasificadores basados en métodos de Bayes, a saber: Deep Feature Weighting Naïve Bayes (DFWNB) y Other Ordinary Feature Weighting Naïve Bayes (OFWNB) [17]. Para los experimentos utilizaron un conjunto de datos de 240,000 documentos.

4. **K Vecinos Más Cercanos (K-Nearest Neighbor, K-nn)**: Qin et al. en 2019 [18] proponen un modelo mejorado basado en K-nn para realizar la clasificación de textos relacionados con ofertas de trabajo. Para resolver el problema de alto tiempo de procesamiento del algoritmo, se realiza una reducción de características antes de realizar la clasificación, seleccionando términos técnicos (vocabulario restringido) que están presentes en una oferta. Los experimentos se llevaron a cabo con el modelo estándar y el modelo propuesto y se logró obtener un menor tiempo de procesamiento y en algunas configuraciones, mayor exactitud con el nuevo modelo. Se utilizaron 6,000 documentos extraídos de páginas de terceros y otras fuentes.

NOTA: Un modelo basado en K-nn se asemeja en cierta forma a una solución de IR ya que utilizan el mismo principio de medir la similitud entre datos almacenados previamente (datos de entrenamiento en K-nn o datos indexados en IR) y el documento a clasificar (o nueva entrada) para encontrar una solución. En K-nn se utiliza para encontrar los k vecinos más cercanos, mientras que en IR se listan los elementos en orden de similitud con la consulta (query).

5. **Máquinas de Soporte Vectorial (Support Vector Machine, SVM)**: En 2020 [19] realizan un modelo basado en SVM y la técnica del descenso del gradiente fraccional, que es una técnica de optimización para algoritmos de clasificación. Los resultados del modelo mostraron que se disminuye el tiempo de entrenamiento, pero a la vez se reduce la precisión en los resultados. Para los experimentos se utilizó un repositorio que contiene 4,143 documentos y 54,877 atributos.

6. **Árboles de decisión (Decision Trees)**: En 2019 [20] proponen un modelo basado en arboles de decisión para la detección de correo spam. Para esto utilizan una técnica híbrida entre árboles de decisión y algoritmos genéticos, donde los algoritmos genéticos se utilizan para mejorar el desempeño del árbol de decisión estándar. También, utilizando PCA para la reducción de las dimensiones del vector de características. Logran mejorar el rendimiento al compararlo con modelos basados en Naïve Bayes, SVM, K-nn y Árboles J-48. Para los experimentos utilizaron un repositorio de correo spam que contiene 4,601 mensajes.

7. **Bosques Aleatorios (Random Forest)**: En 2019 [21] proponen un modelo de clasificación de textos utilizando como clasificador principal a Random Forest. Para realizar la selección de características utilizan algoritmos genéticos, luego utilizan tres modelos combinados en uno basado en Random Forest, Gradient Boosting Machines y Recursive Partitioning, esto con el objetivo de obtener nuevas variables. Luego con estas nuevas variables realizan una nueva clasificación solo con Random Forest y así obtienen mejores resultados. El modelo fue comparado con otros métodos: VSC_SIMCA, extracción de características usando PLSA, y K-nn. Los experimentos fueron realizados con un data set de 1,080 documentos de 9 clases distintas.

8. **Campo Aleatorio Condicional (Conditional Random Field, CRF)**: En 2017 [22] realizan una categorización a nivel de oración para definir su polaridad en análisis de sentimientos. Utilizando un tipo de red neuronal recurrente denominada LSTM (Long Short-Term Memory) y con una capa adicional de CRF, logran obtener un rendimiento similar a modelos que se encuentran en el estado del arte; para esto, se realizaron experimentos comparando el modelo propuesto con 14 modelos distintos, entre ellos, modelos basados en redes

neuronales, naïve bayes, árboles de decisión, entre otros. Los autores utilizaron un grupo de datos que contiene 14,492 oraciones para realizar el entrenamiento del modelo, y para probarlo utilizaron diferentes data sets que contenían diferentes tipos de oraciones para realizar la clasificación, que en total suman 35,901 oraciones.

9. **Deep Learning**: Entre los modelos más recientemente desarrollados para clasificación de textos se encuentra la arquitectura de Hierarchical Attention Network (HAN) (2016) [23], la cual está centrada en la clasificación a nivel de documento, pero además también se enfoca en el nivel de oración y de palabra. Los resultados experimentales mostraron que esta arquitectura mejora los resultados en comparación con 4 trabajos diferentes que aplican métodos basados en SVM, redes neuronales y usan BoW para la representación de los documentos. Para realizar las pruebas se utilizaron en total 8,478,154 de documentos divididos en 6 data sets de distintos tipos.

   En 2017 [24], se propuso una nueva arquitectura jerárquica denominada HDLTex partiendo de que un clasificador básico puede funcionar bien para un número limitado de clases, pero que su rendimiento cae cuando el número de clases a predecir aumenta. Ese problema se resuelve creando sub-arquitecturas que especializan modelos de Deep learning para su nivel en la jerarquía del documento. Los resultados muestran que esta arquitectura tiene un mejor desempeño comparada con la línea base para los experimentos, que son modelos basados en redes neuronales convolucionales y recurrentes, y SVM. Se utilizaron en total 46,985 documentos de 7 clases diferentes.

   En 2018 [25] se propone un modelo basado en redes neuronales convolucionales para la detección de comentarios de Twitter con objeto de agresión verbal, siendo esta una tarea de análisis de sentimientos para textos cortos. Se realizaron experimentos junto con modelos basados en SVM, regresión logística y redes neuronales LSTM, de los cuales el modelo propuesto tuvo un mejor rendimiento. Los datos para el entrenamiento y prueba fueron reutilizados de un trabajo hecho anteriormente por los mismos autores y contaban con documentos con aproximadamente 2,000 características extraídas después de realizar el preprocesamiento de los textos.

   Jang et al. en 2018 [26], proponen un modelo basado en una Red de Creencias Profundas (Deep Belief Network, DBN) y Softmax Regression para la clasificación de textos. El modelo DBN está basado en diversas Máquinas de Boltzmann restringidas. Se utiliza el modelo DBN para resolver el problema de alta dimensionalidad en los datos extraídos en la selección de características y Softmax Regression se emplea para realizar la clasificación. Los experimentos también se llevaron a cabo con SVM y K-nn. Se utilizaron 2 data sets distintos,

uno con 21,578 ejemplos con 135 clases y otro con 18,845 ejemplos con 20 clases diferentes.

Modelos no supervisados: La clasificación no supervisada viene del aprendizaje no supervisado y se da cuando no es posible acceder a datos de entrenamiento, datos con una etiqueta o clase previamente definida, es decir, no se cuenta con documentos con una clase específica en los datos de entrenamiento, así que se usa la inferencia como forma de clasificación, el método más común es el clustering o de agrupación de datos [6]. Un ejemplo reciente se muestra en 2020 [27], donde a pesar de que los documentos utilizados no tienen clases, se aprovecha la ayuda de expertos en el tema para obtener mejores resultados a la hora de clasificar. Se realizaron los experimentos junto con un modelo basado en Naïve Bayes y otros 4 enfoques no supervisados, de los cuales el modelo propuesto tuvo un mejor rendimiento en 4 de los 5 data sets. Los 5 data sets utilizados cuentan con un total de 1,623,166 documentos.

Modelos semi supervisados: La clasificación semi supervisada se presenta cuando hay una pequeña parte de los datos de entrenamiento etiquetados y otra gran parte que no lo están, es una combinación entre técnicas de modelos supervisados y no supervisados, así que, al entrenar el modelo, la gran cantidad de datos sin clase se suple de información extraída de los pocos datos que ya están previamente clasificados [6]. En 2020 [28], se muestran avances de una variación sobre el proceso de entrenamiento para modelos semi supervisados. Principalmente se basa en realizar primero un entrenamiento con los datos que están clasificados, para luego predecir las clases de los datos que no tienen clasificación, y luego se seleccionan los mejores datos que se clasificaron para pasarlos al conjunto de entrenamiento del modelo final junto con los datos previamente clasificados. El método propuesto es comparado con otros 2 que se encuentran en el estado del arte, mostrando que este es robusto contra la acumulación de error y mejora el rendimiento en algunos casos. El modelo utilizado para realizar las pruebas fue TextCNN el cual se basa en redes neuronales convolucionales y los datos utilizados vienen de 4 data sets diferentes, que sumados tienen un total de 1,034,898 documentos, de los cuales solo se tomó el 1% de los datos para hacer el entrenamiento inicial.

Propuestas que incorporan Recuperación de Información: Harish et al. en 2012 [29] proponen un método de clasificación de documentos con el objetivo de preservar la secuencia de ocurrencia de términos en un documento, esto se logra con la ayuda de una estructura de datos propuesta llamada ‘Status Matrix’. Además, con el objetivo de evitar el pareo secuencial (sequential matching) durante la clasificación, indexan las características en un Árbol-B. Para probar su desempeño, el método propuesto fue ejecutado junto con modelos basados en Naïve Bayes, K-nn, SVM y un clasificador basado en votación, de los cuales el método propuesto tuvo mejor precisión que el resto. Los experimentos fueron llevados a cabo con 5 data sets diferentes, uno con 20,000 documentos de 20 clases distintas, otro con

2,000 documentos y 20 grupos, otro con 1,000 documentos de 10 clases, otro también con 1,000 documentos de 10 clases diferentes, y otro en el que no se especifica la cantidad de documentos pero que tiene 4 clases diferentes.

En 2018 [30] se muestra una manera de optimizar las búsquedas a la hora de realizar la clasificación. En este trabajo se indexan las características extraídas de los documentos junto con los documentos, esto se hace para asociar documentos a diferentes términos de búsqueda. Para probar el rendimiento de este enfoque de indexación se utilizaron modelos basados en Naïve Bayes, K-nn, un clasificador basado en centroides y SVM, de los cuales en la clasificación con SVM se obtuvo una mejora en el desempeño. Para realizar los experimentos se utilizaron 3 data sets distintos con un total de 42,018 documentos.

Por otro lado, los enfoques basados en reglas se caracterizan por necesitar intervención humana para crear el modelo de clasificación, pero el proceso de clasificación puede ser mostrado y es más entendible que los modelos de aprendizaje de maquina supervisados previamente mencionados. Recientemente Aubaid et al. [31] en 2020 propusieron un modelo de clasificación de documentos basado en reglas, donde además del enfoque basado en reglas, se utiliza la técnica Doc2Vec de incrustaciones de palabras para extraer las características de los documentos, y así logran obtener una mejor precisión y exactitud en comparación con los algoritmos de machine learning basados en reglas JRip, OneR y ZeroR. Los experimentos fueron realizados con 2 data sets distintos, uno con 21,578 documentos de 5 clases distintas y otro con 20,000 documentos de 20 clases diferentes.

Un área más específica de la clasificación de documentos y que fue importante resaltar e investigar, es la clasificación de flujo de documentos: esta área se caracteriza en que las páginas de los documentos pueden venir en un orden aleatorio y que pueden pertenecer a diferentes clases, con ello entonces, se requiere que la solución pueda clasificar estas páginas y su vez definir su orden correctamente, separándolas en grupos definidos. Los trabajos en esta área se pueden separar en tres subcategorías: características textuales, características visuales, y combinación entre características visuales y textuales. Es preciso señalar que el presente trabajo se centra en características textuales.

Características textuales: Existen múltiples trabajos que utilizan el enfoque de extracción del texto como herramienta principal y realizar un preprocesamiento antes de utilizar un clasificador de aprendizaje de máquina [32]. En [33] utilizan el concepto de continuidad y ruptura para identificar el flujo correcto de los documentos, en conjunto con expresiones regulares para extraer estos descriptores de los documentos. En 2018 [34] proponen un método basado en Doc2Vec y es comparado con un sistema basado en reglas en el que se concluye que el método basado en Doc2Vec mejora la exactitud en problemas de clasificación con múltiples páginas. En [35] hacen uso de Doc2Vec para extraer las características principales

del texto, luego utilizan un regresor logístico para realizar la tarea de clasificación. En [36] utilizan el concepto de descriptores de diseño y de contexto para poder clasificar el flujo de los documentos de manera correcta, esta solución se limita a que es estricta debido a que los descriptores que ya están definidos, y que un descriptor puede estar en mitad de un documento y esté señalándolo como el fin de este. También existen trabajos basados en modelos no supervisados [32] donde se utiliza el método de clustering por K-means, y aunque en los resultados de exactitud no supera a un método supervisado, llega a estar muy cerca. Cabe resaltar que para el desarrollo del método propuesto en este trabajo de grado se utilizaron solo características textuales.

Características visuales: Toman como fuente imágenes y debido a la naturaleza del problema y que las imágenes pueden estar borrosas, un enfoque basado en procesamiento de imágenes es mucho más efectivo que uno basado en texto. Por ejemplo, en [37] utilizan distintos data sets los cuales se caracterizan por tener imágenes con cierto nivel de pérdida de resolución. También en [38] y [39] en se usa una técnica que utiliza Bag-of-Words para realizar clasificación tratando los descriptores de una imagen como palabras. Y por último en 2016 [40] utilizan una red neuronal convolucional y una red neuronal profunda en la que usan como entrada la imagen de la página de un documento.

Combinación de características textuales y visuales:, G. Wiedemann et al. en 2017 [41] y luego en 2019 [42] presentaron un método que combina ambos tipos de características para lograr mejores resultados, utilizando una red convolucional para el texto y otra red convolucional para las imágenes, y luego fusionando los resultados lograron obtener valores de exactitud hasta del 93%. En 2017 [43] con la ayuda de un método de segmentación de pixeles y una red neuronal convolucional los autores proponen un método que mejora resultados que se encontraban en el estado del arte hasta entonces. En [44] al tener datos que tienen texto escrito a mano, un modelo basado en características textuales no funciona muy bien, por ello emplean el enfoque visual y textual a la vez. En 2019 [45] T. Dauphinne et al. utilizaron un modelo basado en una red neuronal convolucional para clasificar imágenes y con una técnica de BoW extraen las características principales del texto, y luego con la ayuda de un meta-clasificador basado en XGBoost que toma los resultados de los dos modelos anteriores, obtiene un mejor resultado. En [46]–[49] hacen uso de esta misma técnica para obtener mejores resultados.

CAPÍTULO 3

3 MÉTODO DE CLASIFICACIÓN AUTOMÁTICA DE DOCUMENTOS

3.1 EL MÉTODO PROPUESTO

Como se presentó en el capítulo anterior, los trabajos realizados alrededor del área de clasificación de documentos usan algoritmos de aprendizaje de máquina. Una de las mayores desventajas que se presentan al utilizar algoritmos de este tipo es que necesitan gran cantidad de información para poder ser entrenados. También, si se requiere agregar información al modelo se necesita reentrenar este desde cero, lo cual no es ideal en un sistema donde los documentos pueden cambiar cada cierto tiempo.

Al utilizar un enfoque de IR se ataca el problema de tener muchos documentos ya que se puede tener un solo documento base para indexar y posteriormente realizar la búsqueda sobre este, y se puede agregar nueva información al índice en cualquier momento.

Para diseñar el método se tuvieron en cuenta las siguientes condiciones:

- **Poca información**: Se necesita tener un documento de ejemplo por clase para realizar el entrenamiento y luego poder clasificar nuevos documentos
- **Estructurado o semiestructurado**: El documento puede ser estructurado o semiestructurado.
- **Opcionales**: Hay documentos que pueden ser opcionales o no dentro de un trámite
- **Sin orden**: Los documentos pueden estar en diferente orden, no solo de clases de documentos, sino de páginas entre documentos.

Para poder realizar un método que funcione para diferentes tipos de trámite, se debe tener en cuenta la información más importante sobre la estructura de un archivo (con uno o múltiples clases de documentos), y como esta puede variar, para cada documento en un trámite, en la **Figura 1** se presenta un ejemplo de documentos necesarios para un trámite. En la vida real, se encontraron dos posibles casos:

- El documento es de una sola página y solo puede tener una página: En este caso se incluyen documentos que tienen formato cerrado (proforma), donde la persona diligencia exactamente la información solicitada.

- El documento puede tener una o muchas páginas: En este caso se incluyen documentos que pueden ser más libres de formato (semiestructurados o no estructurados) para que el usuario agregue tanta información como considere necesario.

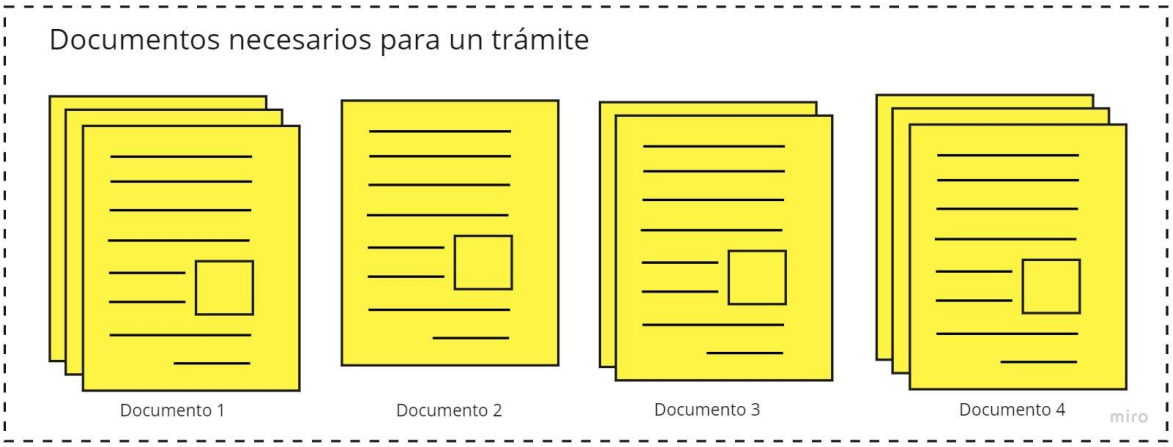

Figura 1 *Documentos necesarios para realizar un trámite en un sólo archivo*

- **Secciones:**

  Para generalizar este tipo de archivos, las clases de documentos se manejan como secciones que pueden ser de un formato cerrado (proforma) o de múltiples páginas con texto abierto (libre). Además, una clase de documento puede ser o no obligatorio dentro un trámite.

  Entonces una sección de un archivo puede ser de un documento de una página o de múltiples páginas y además puede ser o no obligatoria dentro del trámite. En el ejemplo que se muestra en la **Figura 2** cada página de un documento se ha categorizado en si es obligatoria o no, y si es una página proforma o de texto más libre de formato en diferentes páginas.

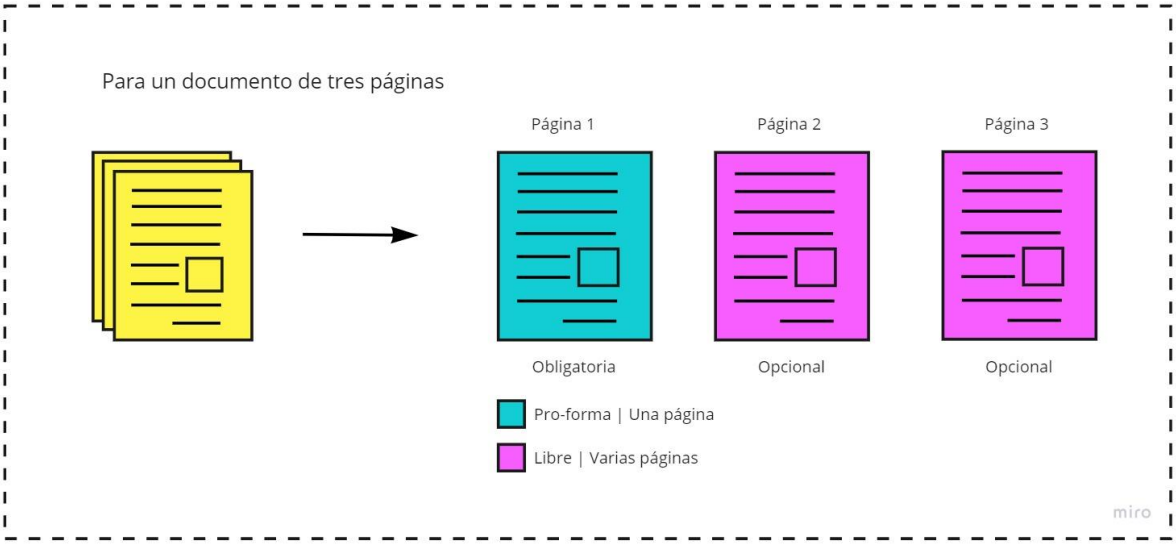

Figura 2 *Ejemplo de división en secciones de un documento de tres páginas*

De esta forma, se logró generalizar documentos que contienen páginas que cambian de longitud y cuando estas puedan ser opcionales.

- **Extracción de texto**

La siguiente tarea que se abordó en el método propuesto fue como extraer el contenido de cada página, ya que el documento puede presentarse en diferentes formatos, como Word, PDF o incluso imágenes. Para hacer esta tarea se utilizó la herramienta para extraer texto a partir de imágenes OCR de Google Cloud Vision [9]. Con la ayuda del OCR se extrae el texto que detecta cuando el formato de la página es una imagen o un PDF.

- **Filtro de texto**

El texto extraído por la herramienta es cualquier texto que se detecte en la imagen, incluso, si es legible, puede detectar texto escrito a mano, pero también puede detectar texto no deseado o innecesario, como por ejemplo textos en logos de las páginas o en los pies de páginas que pueden no ser relevantes para clasificar un documento. Por lo tanto, es necesario realizar el filtrado del texto que no es necesario o irrelevante. Esta tarea se deja inicialmente como manual y el encargado de realizar este filtro debe ser experto en el trámite o la clase de documentos que se manejan, ya que puede detectar cuales son las oraciones o frases determinantes para clasificar un documento o página como un tipo específico. Terminado este proceso, ya se cuenta con el texto guía de cada página de un documento para un trámite.

- **Indexación**

Superados los pasos anteriores se procede a realizar el proceso de clasificación. Para utilizar el enfoque de IR, originalmente se había planteado utilizar Lucene

como librería para manejar índices, pero en la primera fase del proyecto se encontró la herramienta **Elasticsearch**, que está construido sobre Lucene y provee las funcionalidades de esta a través de llamados a API REST, lo que facilita su uso desde microservicios, también provee herramientas de escalabilidad, administración y trazabilidad sobre los índices. Por lo anterior y otras características adicionales de Elasticsearch, el trabajo de grado continuo el proceso con esta herramienta.

El enfoque de IR en términos generales tiene como objetivo encontrar material (generalmente documentos) de naturaleza no estructurada (generalmente texto libre) que satisface una necesidad de información dentro de grandes colecciones almacenadas en una red de computadoras, como internet [50]. En el caso de interés del presente trabajo de grado, los textos de las páginas de entrenamiento definen las características de su clase y con estas se busca clasificar un nuevo documento. En cuanto al concepto de clases incrementales, es preciso comentar que al utilizar el índice de Elasticsearch se pueden indexar documentos en cualquier momento sin necesidad de recrear el índice completo.

Para realizar el proceso de indexación se toma un documento como base para realizar la extracción del texto, luego se separa por páginas automáticamente para que posteriormente el experto realice el filtro del texto y correcta definición de las secciones, y finalmente se indexa cada sección (ver **Figura 3**).

Figura 3 *Resumen del proceso de indexación*

- **Configuración de Elasticsearch**

Debido a que el método propuesto y su implementación debe soportar el proceso de clasificación para múltiples empresas, y tener la capacidad para que cada empresa gestione modelos, usuarios, índice, clasificadores y documentos de acuerdo con sus propios requisitos, se hizo necesario aplicar el concepto del manejo de múltiples inquilinos (multitenancy). Multitenancy se define como un grupo de usuarios que comparten el uso de una sola aplicación, es decir, una sola aplicación funciona de manera flexible y opera como si fuera exclusivo para un usuario. En el caso de esta aplicación, un inquilino (tenant) es una empresa. Por esto, se planteó la forma de utilizar un solo índice de Elasticsearch para que pueda ser utilizado por todas las empresas registradas en el sistema.

Elasticsearch posee las características necesarias para realizar esta separación. El concepto de fragmentos (shards). En esencia, un fragmento (shard) contiene un índice de Lucene en su interior (se puede tener como mínimo un fragmento dentro de un índice de Elasticsearch), pero tener un solo shard limita el sistema ya que este deberá gestionar todas las peticiones, y para el caso de la aplicación que manejan múltiples inquilinos al mismo tiempo no es lo ideal. Por esto se puede configurar el número de shards a tener dentro de un índice y si se aloja este nuevo shard en un nuevo nodo, significaría tener el doble de capacidad computacional para un índice. Es claro que esto no está libre de costos, ya que cada consulta para realizar una búsqueda debe realizarse en todos los shards, esto requiere tener la capacidad computacional para soportar el manejo de todos estos shards. En un principio se puede pensar en configurar un shard por cada empresa, desafortunadamente esto es muy costoso. Teniendo en cuenta lo anterior se optó por tener 10 shards en el índice. También por cada shard se pueden crear réplicas que actúan como balanceadores de carga cuando el shard principal esta sobrecargado. Configurar una réplica conlleva a tener una réplica por cada shard principal, para el caso propuesto en el trabajo de grado, se configuraron 10 shards principales y dos réplicas, es decir un total de 30 shards para todo el índice 10 principales y 20 copias (ver **Figura 4**). Es preciso comentar que esto se puede afinar en la medida que la propuesta y su implementación se usa y el número de empresas, archivos y documentos crece.

Figura 4 *Configuración de shards para índice principal de Elasticsearch*

Ya contando con la capacidad de soportar una cantidad amplia de usuarios (empresas) en la aplicación, la forma de hacer una redirección para cada usuario a los archivos que le pertenecen a este (y para no tener que realizar

consultas sobre todos los shards), se utiliza el concepto de ***enrutamiento de documentos***, el cual permite que al indexar un documento, este se almacene relacionado con un shard específico, y al utilizar siempre el mismo enrutador se logra que siempre se almacene en un mismo shard, para lograr esto, se configura como el valor de enrutador al identificador de la empresa, así cada vez que se agregue un documento este será almacenado en un solo shard; pero esto no significa que en un solo shard se vaya a almacenar información de una sola empresa, puede seguirse almacenando información de muchas empresas. Para resolver esta situación, se configura un filtro de campo, el cual es el identificador de la empresa, así las búsquedas se realizan sobre un shard específico, y se filtra solo para hacer consultas sobre la información de una empresa específica. Por último, se configuraron muchos alias del índice, cada alias puede ser creado con un valor de enrutamiento y filtro de campo, y así cada vez que se haga una petición de búsqueda, se toma como nombre de índice el alias, así se simula que cada usuario (empresa) tiene su propio índice. Un ejemplo de petición en formato JSON para crear un alias se muestra en la **Figura 5**. En este código se trabaja sobre el índice llamado "*índice*", al cual se le va a crear un alias con nombre "*índice-company-001*" y se configura con enrutamiento al identificador de la empresa "*1" y* con un filtro de campo al campo "*tenant_id*" con valor al identificador de la empresa "*1*".

```
PUT /índice/_alias/índice-company-001
{
    "routing": 1,
    "filter": {
        "term": {
            "tenant_id": 1
        }
    }
}
```

Figura 5 *Ejemplo creación de alias de índice*

Otra pregunta que se debió resolver con la configuración propuesta para Elasticsearch es ¿qué sucedería si llegase a existir una empresa con un volumen de documentos demasiado grande? En esta situación se puede llegar a encontrar que un shard se encuentre sobrecargado, para este caso se hace necesario migrar al usuario a un nuevo índice, y para esto, el primer paso es crear el nuevo índice para ese usuario (empresa), luego migrar los datos del shard al nuevo índice y por último realizar la migración del alias, es decir se elimina el alias existente de la empresa y se crea uno nuevo, pero apuntando al nuevo índice. De esta manera se realiza una migración y expansión sin que la empresa se dé cuenta o sin siquiera cambiar datos en código fuente, todo se puede hacer por peticiones REST al servidor de

Elasticsearch. Esta estrategia de escalabilidad y manejo de múltiples tenants fue adaptada de [51].

Una última pregunta que se debió resolver con la configuración propuesta para Elasticsearch es ¿cómo una empresa puede gestionar diferentes versiones del proceso de clasificación? En este caso se tomó la decisión de crear dos índices de Elasticsearch distintos, uno que maneje los índices principales y se utilice cuando el modelo entrenado está listo para ser utilizado en despliegue, y otro que se usa para realizar pruebas, para así tener la posibilidad de que pueda modificar el índice libremente sin preocuparse de que haya posibles daños al modelo cuando se cambie o agregue un archivo de documento(s). Después la empresa tiene la oportunidad de migrar su configuración e información del índice de pruebas al índice principal. El índice de pruebas fue configurado al igual que el índice principal con 10 shards principales y 1 de copias, es decir 20 shards en total (ver **Figura 6**).

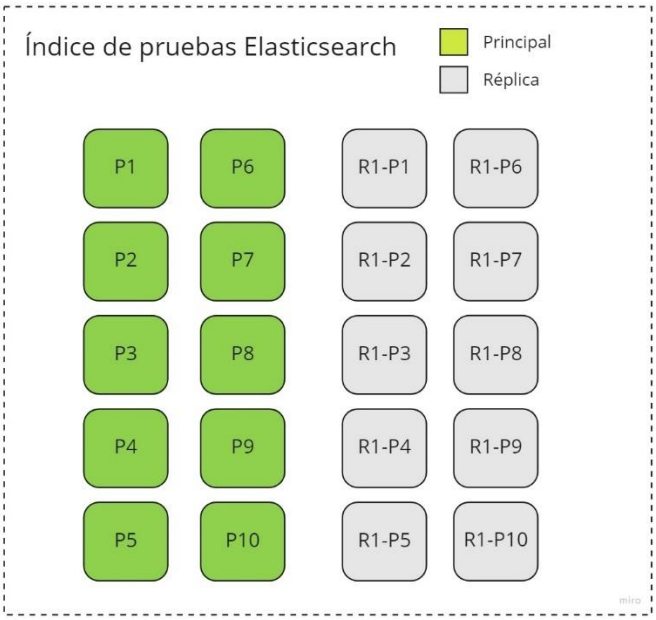

Figura 6 *Configuración de shards para el índice de pruebas de Elasticsearch*

Ya que se ha configurado el método con todas las herramientas necesarias, y se hayan indexado los documentos que se necesitan para el trámite, se da paso a la tarea de realizar la clasificación de los documentos.

- **Búsqueda**

  Al ya haberse indexado documentos en Elasticsearch, se pueden realizar búsquedas sobre este. Las búsquedas sobre un índice, al igual que en la indexación, se debe realizar con el texto del contenido del documento, es decir, que primero se debe realizar la extracción del texto del archivo a clasificar. Para esto, se apoya nuevamente en el servicio OCR de Google

Cloud Vision. Ya que la tarea de clasificación debe ser automática, no se requiere revisar los datos extraídos por el OCR, por lo tanto, el texto pasa directamente a la tarea búsqueda en el índice.

Las búsquedas sobre el índice de Elasticsearch se realizan sobre un campo de documento, este campo debe ser el mismo sobre el cual se guardó la información del texto cuando se indexó este, este campo es el *contenido*. En la **Figura 7** se define un ejemplo de búsqueda a través de consulta por API a Elasticsearch, donde se realiza la búsqueda por el campo *content* y en las respuestas solo se incluirá el campo *clase*

```
GET /índice-company-001/_search
{
  "query": {
    "match": {
      "content": "start ... end of text "
    }
  },
  "_source": {
    "includes": "clase"
  }
}
```

Figura 7 *Consulta para búsqueda sobre índice elasticsearch*

La consulta al índice retornará los documentos más similares ordenados de mayor a menor puntaje (score), donde el mayor puntaje indica que es la mejor coincidencia. Siempre se tomará como clase predicha a la clase del documento con mayor puntaje. Un ejemplo de resultados de búsqueda en formato JSON se puede ver en la **Figura 8** donde el array *hits* corresponde a los documentos encontrados, y ya que está ordenado por score, el primer *hit* es la mejor coincidencia, entonces se toma el nombre de la clase como clase predicha del documento, en este ejemplo **póliza-salud.**

```
"hits" : [
    {
        "_index" : "test-index",
        "_type" : "_doc",
        "_id" : "HE7F4n8BflV7_Abvgb6L",
        "_score" : 2298.1406,
        "_routing" : "9",
        "_source" : {
          "clase" : "poliza-salud"
        }
    },
    {
        "_index" : "test-index",
        "_type" : "_doc",
        "_id" : "JU7F4n8BflV7_Abvgr4h",
        "_score" : 410.29163,
        "_routing" : "9",
        "_source" : {
          "clase" : "poliza-copropiedades"
        }
    },
    { ... }
]
```


Figura 8 *Ejemplo de resultados de búsqueda sobre índice elasticsearch*

Para realizar el proceso de búsqueda sobre el índice se toma el documento que se quiere clasificar, se extrae el texto de cada página para luego realizar la búsqueda del texto de estas, y finalmente el índice retorna los resultados (ver **Figura 9**).

Figura 9 *Resumen del proceso de búsqueda sobre un índice*

Con esto se finaliza el proceso de clasificación para un solo documento. En caso de que el modelo contenga más documentos y se requieran verificar todos, se debe hacer el mismo proceso por cada uno.

3.2 EVALUACIÓN DE CALIDAD DEL MÉTODO

Para realizar la evaluación de la calidad del método se utilizaron documentos facilitados por la empresa ATIX Digital S.A.S para poder entrenar y evaluar el método propuesto. Esta base de datos contiene documentos de seguros de diferentes empresas. Cada empresa ofrece distintos tipos de seguro, pero algunas ofrecen seguros similares, por ejemplo, seguros de salud, seguros de vehículos y seguros de responsabilidad civil. Con estos datos fue posible definir un protocolo para evaluar cómo se comporta el método cuando hay dos tipos de documentos muy similares.

Las pruebas se ejecutaron de forma incremental, es decir, para evaluar la estabilidad del método se ejecutan pruebas con documentos de dos clases, en la segunda iteración se ejecutan pruebas con documentos de tres clases distintas, y así sucesivamente hasta llegar a tener 5 clases diferentes, luego se realizan más iteraciones, pero esta vez incrementando 5 clases al tiempo, hasta tener 30 clases.

Para evaluar cada iteración de las pruebas se tomó un enfoque de evaluación generalmente aceptado en minería de datos y aprendizaje de máquina, en específico, la validación cruzada con 3 folders y a partir de los datos arrojados se calculó el valor de precisión, recuerdo y medida F (F1).

Sobre cada clase se escogieron entre 4 y 5 documentos que se usaron para realizar el entrenamiento.

Como prerrequisitos para realizar la indexación de documentos en el índice de Elasticsearch, se debe extraer el texto de los archivos, así que desde un principio se seleccionan cuáles conjuntos de documentos van a pertenecer a cada iteración y se realiza un proceso de extracción de texto con la ayuda de la herramienta OCR de Google.

Primera Iteración: Se tomó como conjunto de entrenamiento documentos con dos clases distintas, (ver **Tabla 1**). Las clases de documentos que se usan para entrenar son *allianz-salud* y *allianz-vehículos*.

Tabla 1 *Documentos para realizar evaluación en la primera iteración*

Nombre de documento	Clase
200522471-salud	allianz-salud
200643074 - salud	allianz-salud
200777994-salud	allianz-salud
200856194 - salud	allianz-salud
200610829 - veh	allianz-vehiculos
200640916 - veh	allianz-vehiculos
200690523 - veh	allianz-vehiculos
200723797	allianz-vehiculos
200746553- veh	allianz-vehiculos

Entonces se ejecutó un script que está descrito en el **Anexo D**, para decidir los registros que conformarían los folders para validación cruzada. Debido a que el número de folders (de la validación cruzada) es de 3, el proceso se repite la misma cantidad de veces. En la **Tabla 2**, **Tabla 3** y **Tabla 4** se pueden ver los resultados de selección de los folders.

Tabla 2 *Selección de documentos a utilizar para el primer folder*

Nombre de documento	Clase	Selección
200522471-salud	allianz-salud	entrenamiento
200643074 - salud	allianz-salud	entrenamiento
200777994-salud	allianz-salud	validación
200856194 - salud	allianz-salud	entrenamiento
200610829 - veh	allianz-vehiculos	entrenamiento
200640916 - veh	allianz-vehiculos	entrenamiento
200690523 - veh	allianz-vehiculos	validación
200723797	allianz-vehiculos	entrenamiento
200746553- veh	allianz-vehiculos	validación

Tabla 3 *Selección de documentos a utilizar para el segundo folder*

Nombre de documento	Clase	Selección
200522471-salud	allianz-salud	validación
200643074 - salud	allianz-salud	validación
200777994-salud	allianz-salud	entrenamiento
200856194 - salud	allianz-salud	entrenamiento
200610829 - veh	allianz-vehiculos	entrenamiento
200640916 - veh	allianz-vehiculos	entrenamiento
200690523 - veh	allianz-vehiculos	entrenamiento
200723797	allianz-vehiculos	validación
200746553- veh	allianz-vehiculos	entrenamiento

Tabla 4 *Selección de documentos a utilizar para el tercer folder*

Nombre de documento	Clase	Selección
200522471-salud	allianz-salud	entrenamiento
200643074 - salud	allianz-salud	entrenamiento
200777994-salud	allianz-salud	entrenamiento
200856194 - salud	allianz-salud	validación
200610829 - veh	allianz-vehiculos	validación
200640916 - veh	allianz-vehiculos	validación
200690523 - veh	allianz-vehiculos	entrenamiento
200723797	allianz-vehiculos	entrenamiento
200746553- veh	allianz-vehiculos	entrenamiento

Por cada folder, se indexan los documentos que fueron seleccionados para entrenamiento, se indexa el contenido del documento y el nombre de la clase. Luego, por cada documento seleccionado para validación se realiza una búsqueda por el contenido de este y Elasticsearch arroja como resultado cuál de los documentos indexados tiene mayor similitud. Si la clase encontrada es la misma que la clase verdadera significa que el método tuvo éxito en clasificar este documento. Al final de la iteración, cuando se tiene información de todos los resultados para cada folder, se recopila todo en una matriz de confusión (ver **Tabla 5**).

Tabla 5 *Matriz de confusión para Iteración 1*

	Predicted: allianz-salud-pred	**Predicted: allianz-vehiculos-pred**
True: allianz-salud-real	4	0
True: allianz-vehiculos-real	0	5

Donde las filas representan las clases verdaderas y las columnas las clases predichas. Los valores numéricos en cada casilla corresponden a la cantidad de veces que un documento se verificó y se predijo como su clase real u otra. Con los resultados de la matriz de confusión se calcularon las medidas de evaluación (precisión, recuerdo y medida F). Para el caso de la primera Iteración se obtuvieron los siguientes resultados: Precisión de 100%, Recuerdo de 100% y Medida F (F1) de 100%.

Concluida la evaluación de la primera iteración, se realiza este proceso 4 veces más, pero en cada iteración agregando una clase de documentos nueva.

Los resultados de la segunda iteración con la clase adicional *allianz-copropiedades* tuvo los mismos resultados en las métricas de precisión, recuerdo y medida F que en la primera iteración.

Los resultados de la tercera iteración con clase adicional *allianz-responsabilidad-civil* cambió un poco (ver **Tabla 6**). Las métricas de calidad reportaron: Precisión de 90%, Recuerdo de 92,8% y Medida F de 91,4%.

Tabla 6 *Matriz de confusión para iteración 3*

	allianz-salud-pred	**allianz-vehiculos-pred**	**allianz-copropiedades-pred**	**allianz-responsabilidad-civil-pred**
allianz-salud-real	4	0	0	0
allianz-vehiculos-real	0	3	2	0
allianz-copropiedades-real	0	0	5	0

allianz-responsabilidad-civil-real	0	0	0	5

En la cuarta iteración se agregó la clase *mapfre-vehiculos* que es el mismo tipo de seguros utilizados por la clase *allianz-vehiculos*. Se reflejaron algunos cambios en las validaciones y métricas de precisión (92%), recuerdo (94,3%) y medida F (93,1%) (ver **Tabla 7**).

Tabla 7 *Matriz de confusión para iteración 4*

	allianz-salud-pred	**allianz-vehiculos-pred**	**allianz-copropiedades-pred**	**allianz-responsabilidad-civil-pred**	**mapfre-vehiculos-pred**
allianz-salud-real	4	0	0	0	0
allianz-vehiculos-real	0	3	2	0	0
allianz-copropiedades-real	0	0	5	0	0
allianz-responsabilidad-civil-real	0	0	0	5	0
mapfre-vehiculos-pred	0	0	0	0	5

Luego de estas iteraciones, se realizaron 5 iteraciones adicionales pero esta vez agregando 5 clases de documentos nuevas al mismo tiempo, es decir, para la quinta iteración se agregaron 5 clases a las 5 ya registradas anteriormente, para un total de 10. Las métricas obtenidas para todas las iteraciones realizadas se muestran en la **Tabla 8**.

Tabla 8 *Resultados de medidas para iteraciones con clases incrementales*

Iteración	**Número de clases**	**Precisión**	**Recuerdo**	**Medida F1**
1	2	100%	100%	100%
2	3	100%	100%	100%
3	4	90%	92,85%	91,41%
4	5	92%	94,28%	93,13%
5	10	96%	96,67%	96,33%
6	15	97,33%	97,77%	97,55%
7	20	92,75%	94,64%	93,68%
8	25	91,80%	93,10%	92,44%

9	30	85,83%	85,53%	85,68%

En la **Figura 10** se muestra un gráfico comparativo de las métricas de precisión, recuerdo y medida F1 desde la iteración 4 hasta la 9.

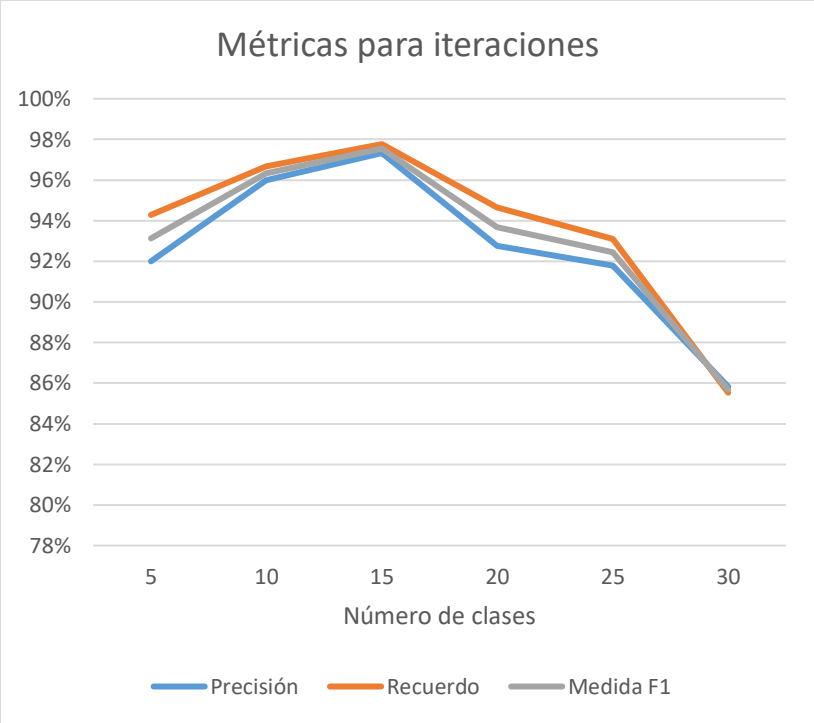

Figura 10 *Gráfico de resultados de métricas para iteraciones con clases incrementales*

Es evidente que este proceso de evaluación se debe ampliar a mayor cantidad de documentos, desafortunadamente, los trabajos previos en el área cuentan con data sets privados y no se logró tener acceso a estos durante el trabajo de grado. Por otro lado, los documentos facilitados por la empresa también son confidenciales y no se cuenta con el permiso para abrirlos al público.

CAPÍTULO 4

4 DISEÑO DETALLADO DEL SISTEMA

4.1 ARQUITECTURA GENERAL DEL SISTEMA

Para poder realizar el desarrollo de una aplicación web basada en microservicios se hizo necesaria la tarea de buscar patrones de arquitectura para la elaboración de este tipo de software, para esto se tomó como referencia los patrones de microservicios propuesto por Chris Richardson [52].

La arquitectura de la aplicación se dividió en Back-end y Front-end. El Back-end se encuentra dividido en 7 microservicios:

- Autenticación y autorización (Auth service).
- Usuarios (User service).
- Entrenamiento (Training service).
- Verificación (Verification service).
- Elastic (Elastic service).
- Clasificación (Clasification service).
- Puerta de entrada (API Gateway).

El API Gateway se encarga de coordinar las peticiones recibidas desde el Front-end hacia los microservicios correspondientes. El Front-end se compone de una aplicación web que proporciona la interfaz necesaria para que el usuario pueda realizar las tareas de entrenamiento y verificación de los modelos. La arquitectura de la aplicación se muestra en la **Figura 11**.

Cada servicio fue desplegado en un contenedor distinto, los servicios Training service, User service y Verification service cuentan con una base de datos propia. La aplicación en su totalidad cuenta con seguridad basada en JSON Web Tokens (JWT). El servicio de Elastic service consume los servicios de Elasticsearch a través de peticiones REST a una instancia local de este. Los servicios Elastic service, Text extraction service y Clasification service son solo accedidos de manera interna, es decir, no se hacen llamados directa desde el cliente web (están protegidos de Internet).

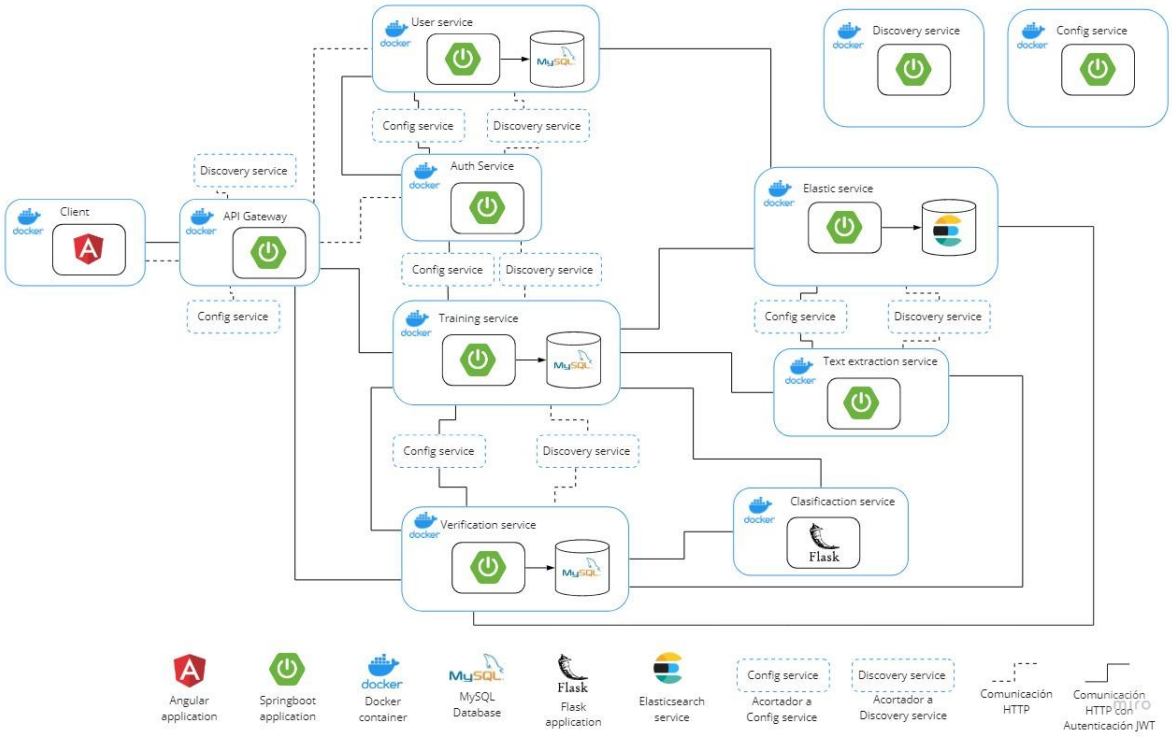

Figura 11 *Arquitectura de la aplicación*

Dentro de la aplicación se manejan cuatro roles distintos:

- **Empresa (Company)**: Es el usuario principal de la aplicación, con este rol puede realizar entrenamientos de modelos, verificaciones de los modelos creados y creación de cuentas secundarias.
- **Entrenador (Trainer)**: Es un usuario que es creado por una empresa, tiene permisos de realizar el entrenamiento de modelos.
- **Configurador (Configurer)**: Al igual que el entrenador, es creado por una empresa y tiene permisos de realizar verificaciones.
- **Administrador**: Es el encargado de verificar información de registro de la empresa, visualizar uso del sistema por parte de estas y desactivarlas si es necesario.

Toda petición hecha desde el Front-end la recibe al API Gateway, y en caso de que sea necesario, verifica el JWT con la ayuda del servicio Auth service. A continuación, se describe cada componente de la aplicación web.

4.1.1 ARQUITECTURA DEL FRONT-END

El Front-end es una aplicación web desarrollada en Angular. Esta aplicación se comunica a través del API Gateway con los microservicios. Las funcionalidades principales de la aplicación son: registro de empresas, autenticación, gestión de administrador, gestión de modelos, y la verificación y predicción con el uso de los modelos. Estas funcionalidades se definen a continuación organizadas por cada rol.

Registro de empresa: Una persona o empresa que esté interesada en hacer uso de los servicios de la aplicación debe primero realizar un registro, a este tipo de usuario se le llama empresa dentro de la aplicación. Una vez registrado debe esperar a que un administrador acepte su solicitud, y solo así, puede iniciar sesión y hacer uso de las funcionalidades de la aplicación (ver **Figura 12**).

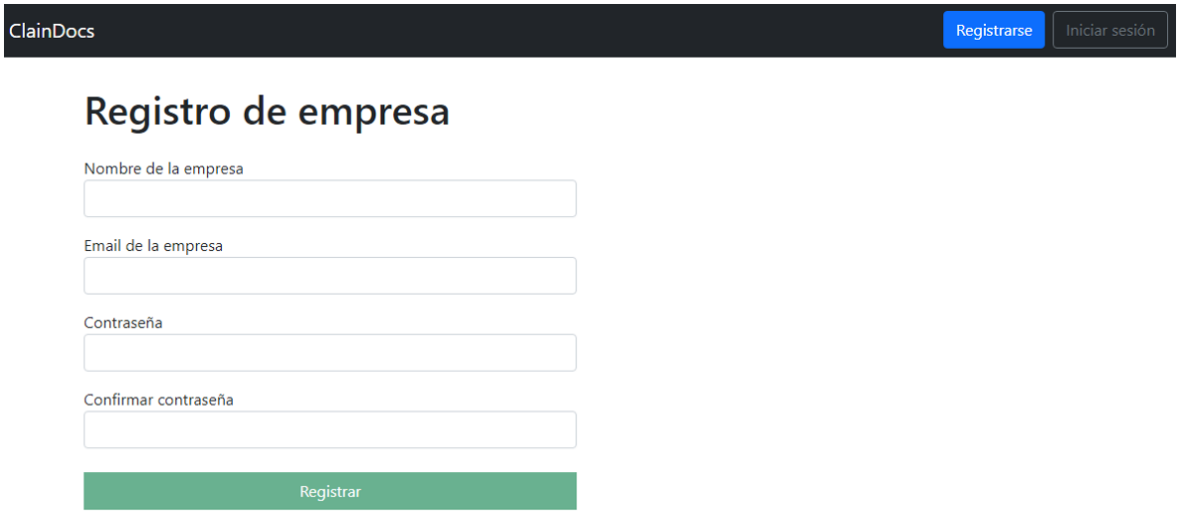

Figura 12 *Formulario de registro de empresa*

Inicio de sesión: Desde este formulario, los tres roles pueden ingresar a la aplicación web, usando únicamente un correo electrónico y su contraseña.

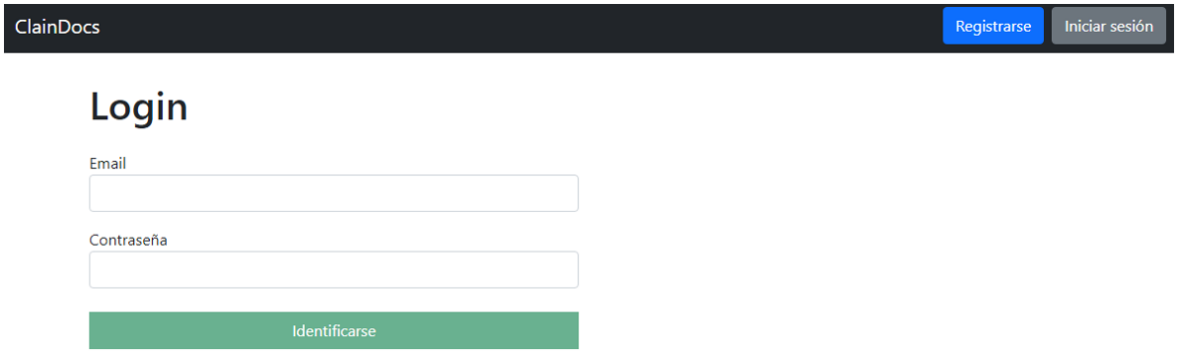

Figura 13 *Interfaz de inicio de sesión*

Administrador: Debe existir el rol de administrador del sistema para poder gestionar las empresas, aceptar el registro de estas, verificar su uso de la aplicación y para poder desactivar una empresa en caso de que sea necesario.

En la página principal del administrador, como se muestra en la **Figura 14**, se muestran las empresas que han solicitado registrarse en la aplicación, y de forma similar, en la **Figura 15**, se listan todas las empresas registradas en el sistema, con el estado, el nombre, la fecha de registro y un botón para ver más detalles de cada una de estas.

Figura 14 *Lista de solicitudes de registro de empresa*

Figura 15 *Lista de las empresas registradas en el sistema*

Al entrar a ver los detalles de la empresa (ver **Figura 16**) se muestra la fecha de registro de la empresa, la fecha en que se activó, en caso de que se haya activado y las subcuentas que tiene registrada la empresa y los roles que posee. También, se muestra en una lista las validaciones y predicciones que ha hecho (ver **Figura 18**), por último, la funcionalidad de desactivar la empresa en caso de que sea necesario (ver **Figura 17**).

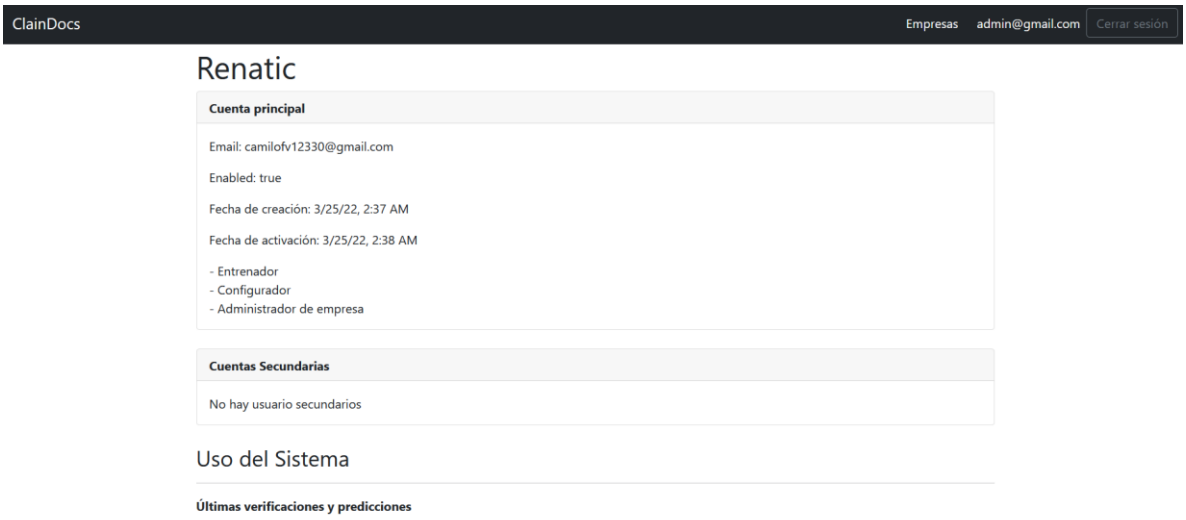

Figura 16 *Detalle de cuenta principal y subcuentas de empresa*

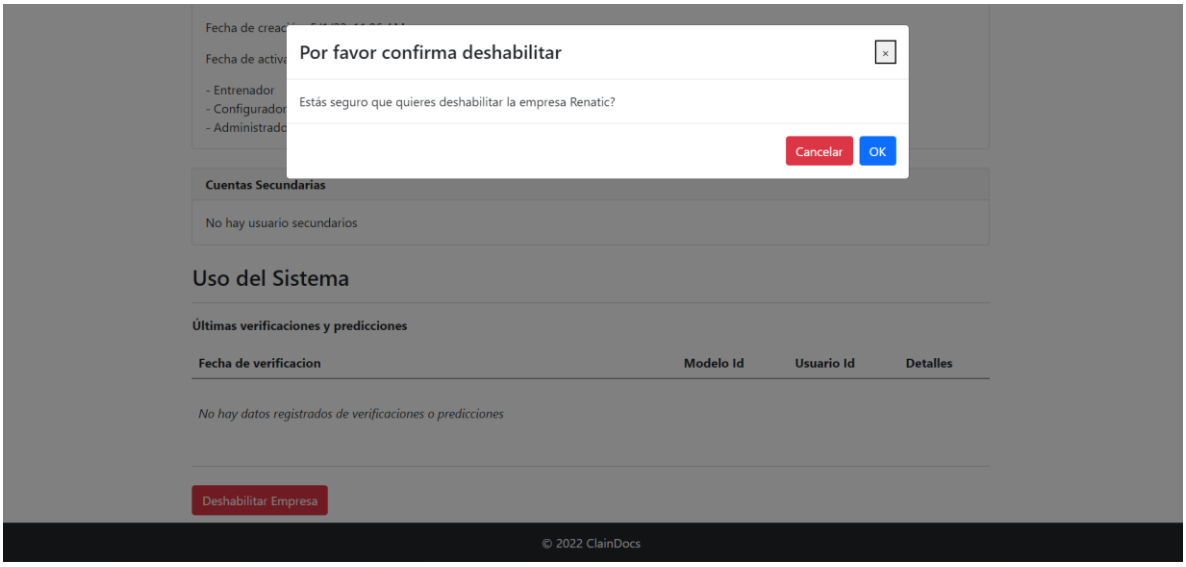

Figura 17 *Opción de deshabilitar empresa*

Empresa: Una vez la empresa esté activa dentro de la plataforma, tiene disponible las funcionalidades de entrenamiento, verificación y creación de subcuentas. La empresa es el cliente del sistema, este rol se compone de dos sub roles: Administrador de empresa y Entrenador.

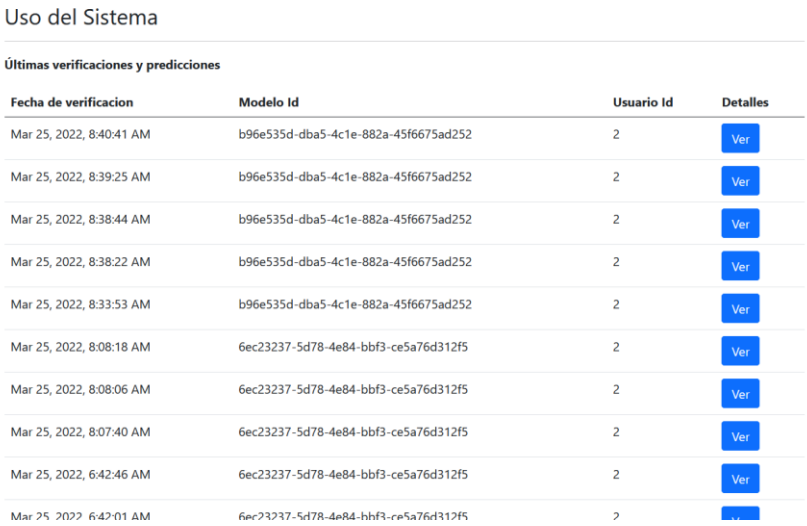

Figura 18 *Lista de predicciones y validaciones de empresa*

El administrador de la empresa es el usuario que realizó el primer registro. El configurador es un tipo de subcuenta que es creada directamente por el administrador del sistema.

Tanto el administrador de la empresa como el entrenador pueden iniciar sesión con sus credenciales desde la página de inicio de sesión. La página principal para ambos roles de empresa se ve como se muestra en la **Figura 19**, solo que el entrenador no tiene la opción de *cuentas secundarias*. Esta página principal

contiene una lista con el resumen de los modelos que se han guardado anteriormente, seguido de una lista de verificaciones y predicciones que ha realizado dentro del sistema.

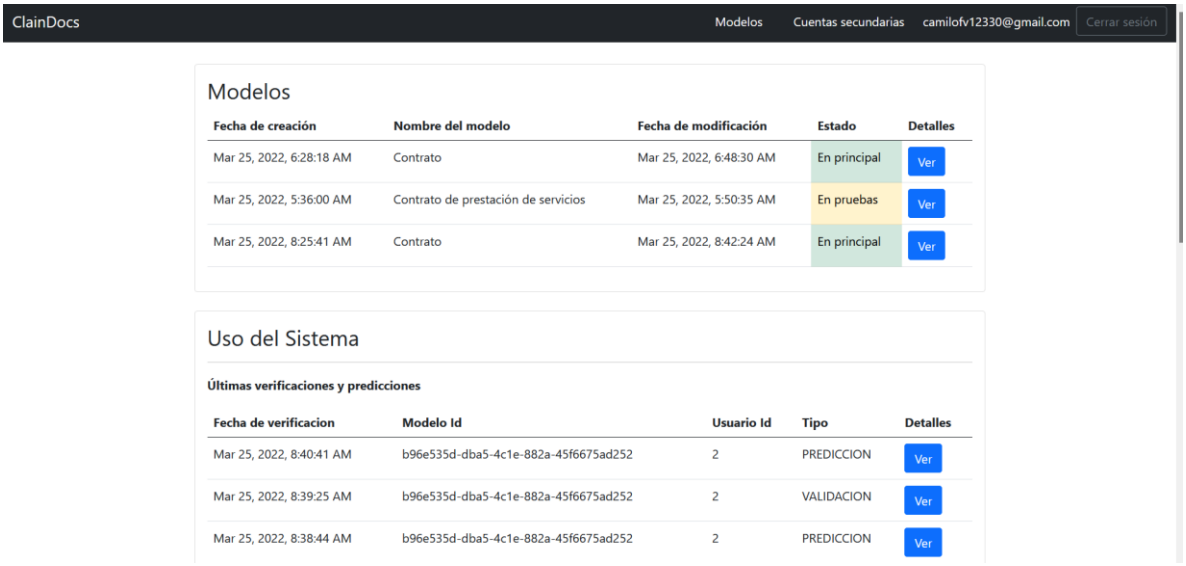

Figura 19 *Página principal de empresa*

Para el caso del administrador de la empresa se listan las predicciones que ha hecho él y los usuarios correspondientes a sus subcuentas, en cambio, para el entrenador, solo se muestran las que ha realizado él mismo.

Registro de subcuentas: Cuando el usuario es un administrador de empresa se puede acceder a la opción de subcuentas desde la barra de navegación. Una vez el administrador de empresa ingresa a esta interfaz (ver **Figura 20**) donde puede observar la lista de las subcuentas que ha asignado, se muestra el correo que tiene registrado, si está activo o no y los roles asignados a este. También la opción de *modificar* que permite borrar la cuenta o desactivarla para no permitir que se pueda autenticar hasta que se vuelva activar (ver **Figura 21**). La **Figura 22** muestra además la opción de crear una nueva subcuenta.

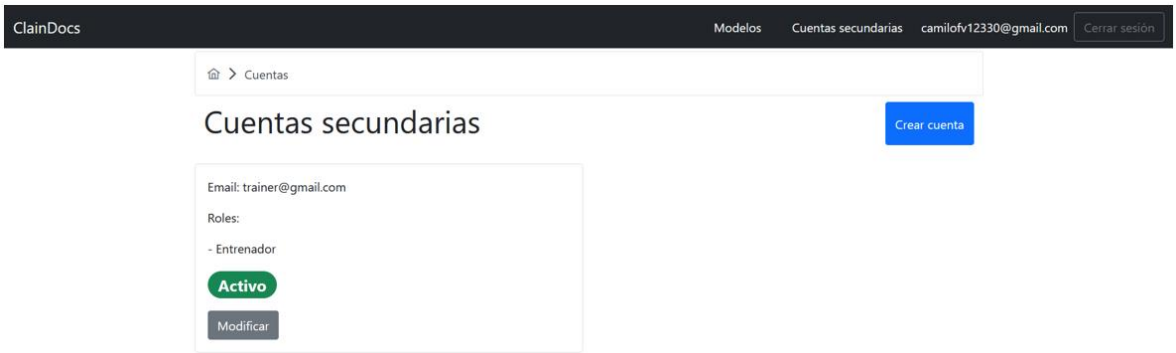

Figura 20 *Interfaz de gestión de subcuentas de empresa*

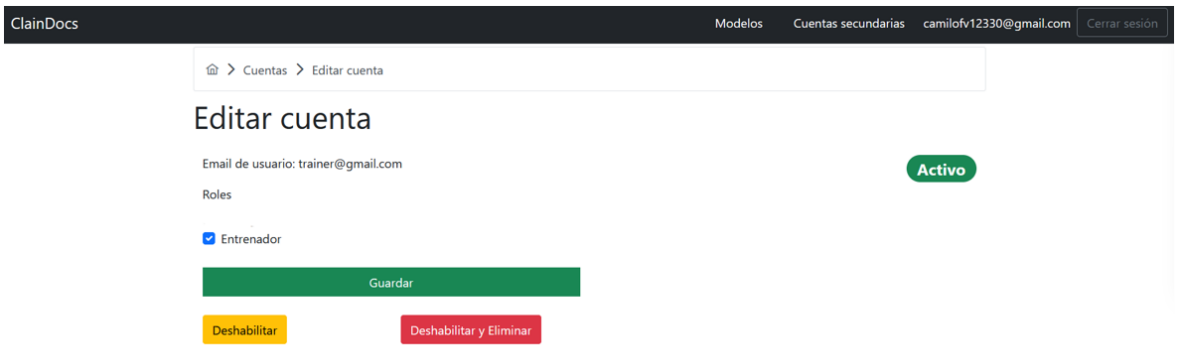

Figura 21 *Interfaz de modificación de subcuenta de empresas*

Entrenador: Cabe resaltar nuevamente que el rol de administrador de empresa contiene los permisos de entrenador, por lo tanto, este también puede acceder a funcionalidades del rol entrenador. Una vez el entrenador inicia sesión en el sistema, y accede a la opción de modelos ubicada en la barra de navegación, este podrá visualizar información de los modelos (ver **Figura 23**). Por cada modelo, se muestra información de la fecha de creación, fecha de última modificación, nombre del modelo, y el estado, que representa si el modelo se encuentra en estado de pruebas o en producción (principal). También tiene la opción de ver los detalles de cada modelo y crear un nuevo modelo. Cuando el entrenador quiere crear un nuevo modelo, debe ingresar solo el nombre distintivo, como se muestra en la **Figura 24**.

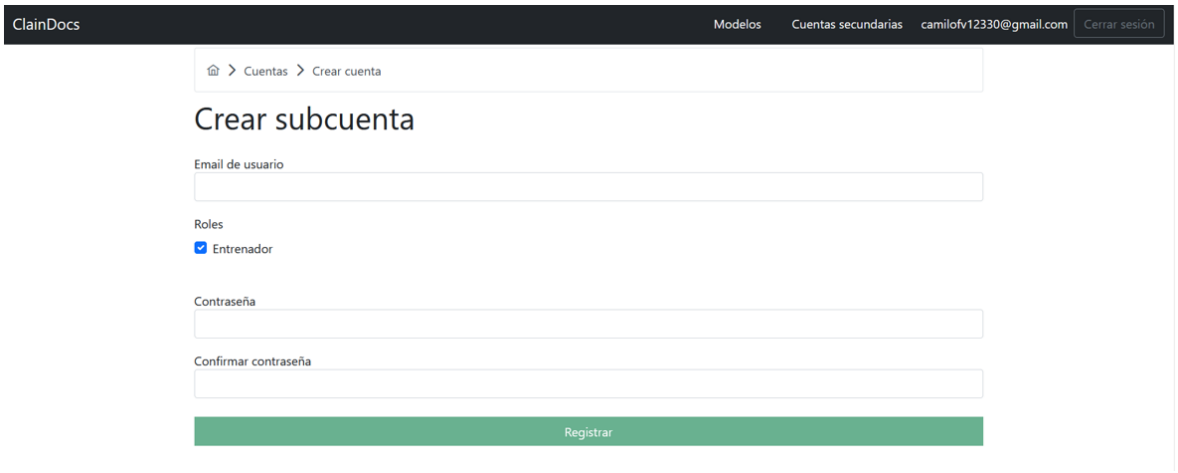

Figura 22 *Interfaz de creación de subcuentas de empresa*

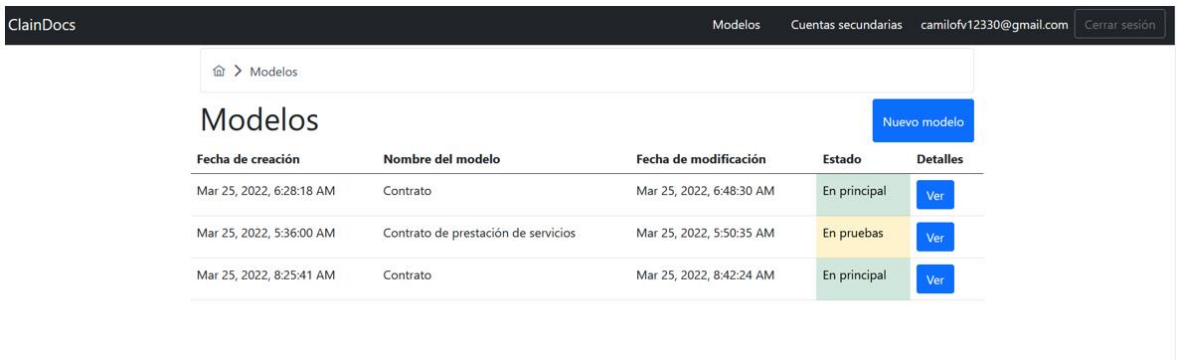

Figura 23 *Interfaz de lista de modelos del entrenador*

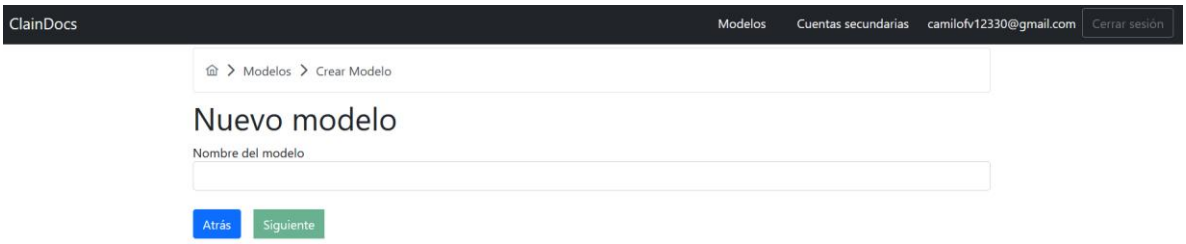

Figura 24 *Interfaz de creación de modelo*

Al crear el modelo el entrenador accede a la vista de detalle de un modelo (ver **Figura 25**), en el caso de crear un nuevo modelo este se encuentra vacío. Esta interfaz se compone de 4 secciones principales, que también se pueden dividir en orden para realizar un entrenamiento y verificación:

- **Documentos**: Son los documentos que hacen parte del modelo, también tiene la opción de agregar más documentos si es necesario.
- **Método de clasificación**: Se puede visualizar la información del clasificador seleccionado para realizar predicciones, además de la opción de pasar a configurarlo.
- **Verificaciones**: Están las opciones para redirigirse a realizar predicciones, validaciones o ir al histórico de verificaciones.
- **Cambio de versión del modelo**, es la sección a la derecha del título del modelo, con las opciones *Pruebas* y *Principal* que permiten cambiar la versión de modelo actual. Cabe recordar que la versión *Pruebas* indica que el modelo es utilizado en un ambiente de pruebas y el *Principal* está pensado para ser usado en un ambiente de producción.

En este mismo orden de componentes se puede ver el flujo de trabajo para realizar la tarea de entrenar y verificar los documentos.

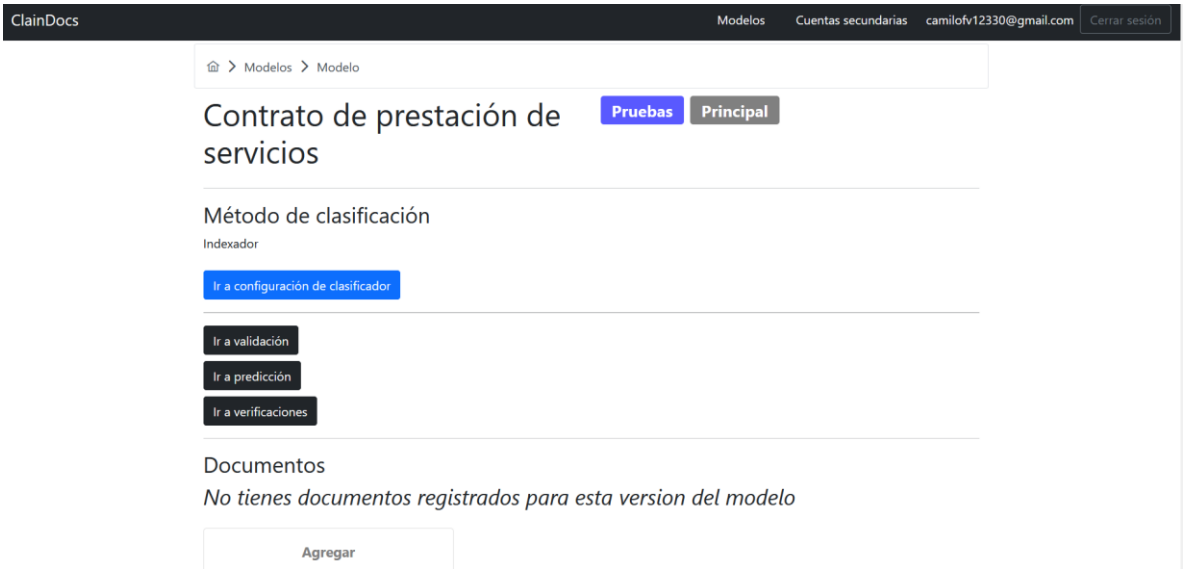

Figura 25 *Interfaz de detalle de modelo creado*

Paso 1 - Documentos

Al seleccionar la opción de crear un nuevo documento se muestra una interfaz que permite cargar un documento, que el entrenador desee tomar como base para entrenar el modelo (ver **Figura 26**). Los formatos aceptados para cargar el documento son PDF, JPG, JPE o PNG y el archivo debe ser de máximo 25 megabytes.

Luego de cargar el documento y dar click en siguiente, el sistema extraerá el texto de este documento y será mostrado en la siguiente página (ver **Figura 27**) que permite visualizar el detalle del documento. Este se compone de una primera parte con el texto que se extrajo del documento subido en el paso anterior. La segunda parte corresponde a la creación, eliminación y modificación de secciones. La primera vez que se hace la extracción de texto por cada página detectada en el documento subido se crea una sección.

Figura 26 *Interfaz para carga de documento y luego usarlo para entrenamiento*

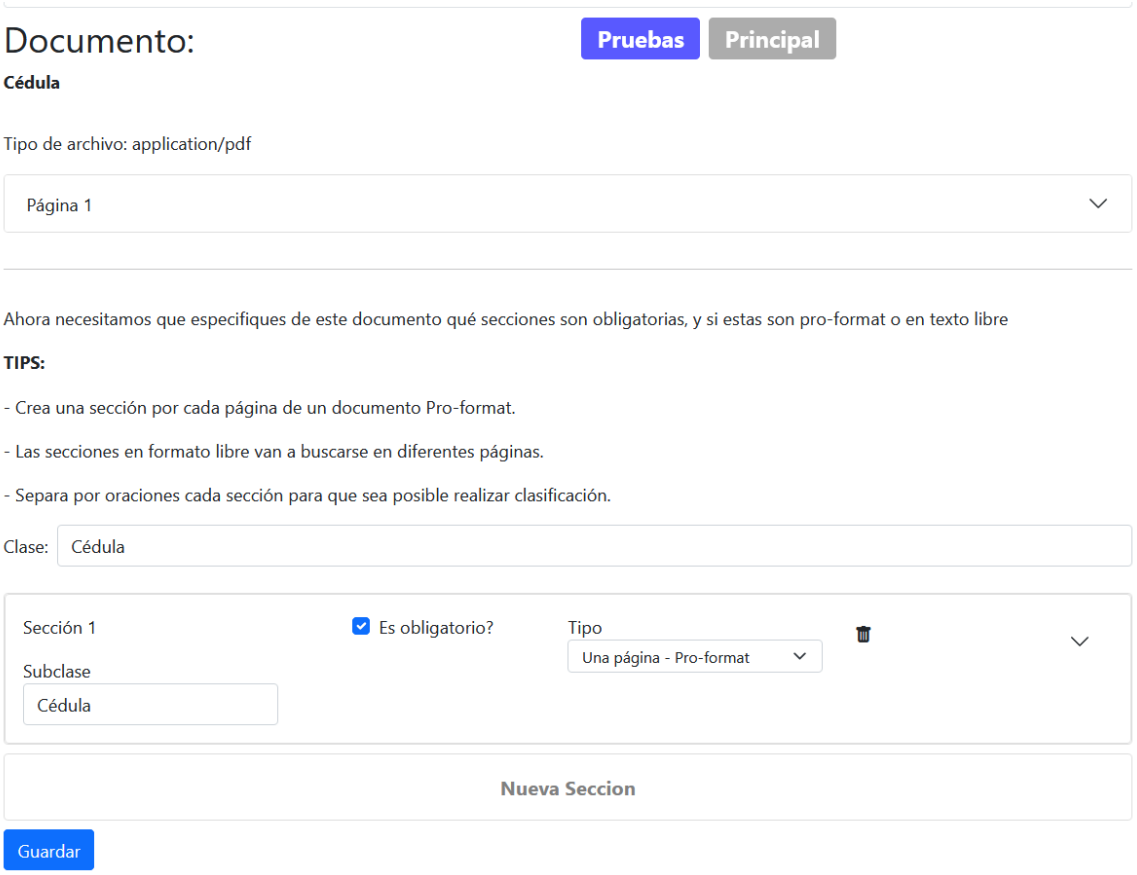

Figura 27 *Interfaz de detalle de un documento*

Cada sección puede ser de tipo *Una página* o *Varias páginas:* Una sección categorizada como *una página* representa una página de archivo que se está entrenando y significa que su contenido solo es encontrado en una página; en cambio, una sección categorizada como *varias páginas* puede ser encontrada en múltiples páginas, es decir, puede presentarse en una, dos o más páginas. Esta categoría de sección es útil cuando un documento puede variar en longitud por la cantidad de información que agrega una persona (ver **Figura 28**).

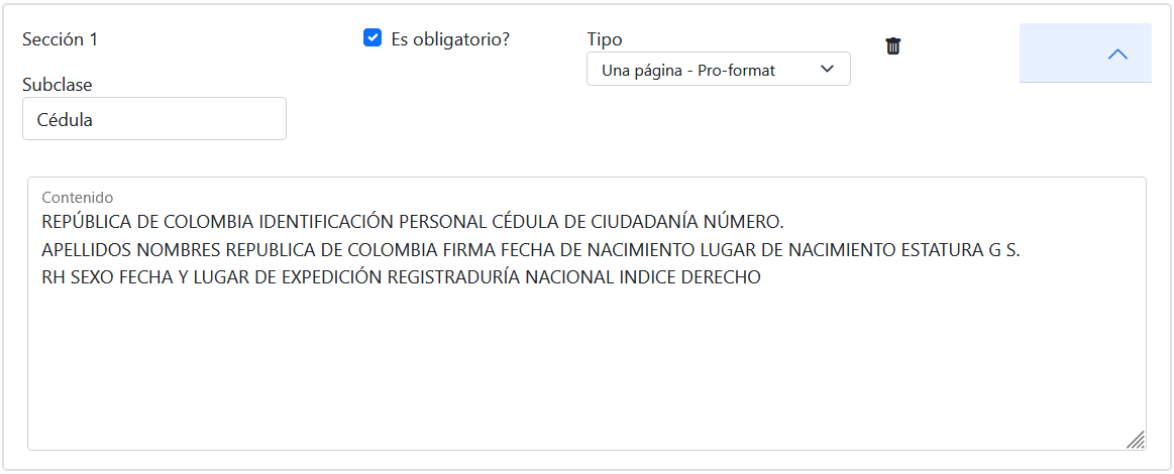

Figura 28 *Interfaz de entrada de datos para una sección de documento*

La sección también puede ser marcada como obligatoria o no obligatoria, esto indica que puede o no encontrarse en la entrada de los documentos al realizar una predicción y no afecta los resultados. Por último, el campo de texto de la sección tiene que ser filtrado por el entrenador, es decir, especificar solo las frases y palabras que determinan que una sección es de la clase que se especificó en el nombre, esto incluye eliminar texto que genere ruido e información específica del documento. Por ejemplo, en la **Figura 29** se muestra el filtrado de texto de una cédula de ciudadanía colombiana, el texto tachado es información sensible de una persona y debe ser eliminado. También, al ser todo este texto el extraído por el OCR, puede tener muchos errores ortográficos o texto que no tiene sentido, por esto es muy importante la tarea del entrenador ya que entre mejor filtre el texto, mejores serán los resultados de predicción que se realicen después sobre el modelo.

Una vez las secciones quedan configuradas de acuerdo con las necesidades del documento, este se guarda. Al volver, al menú principal del modelo se puede ver ahora el documento agregado en la vista de documentos. Luego, se pueden crear y agregar más documentos si esto es requerido (ver **Figura 30**).

Después, por cada documento se pueden ejecutar las acciones de borrar, ver los detalles e indexar, con esta última opción se guarda el documento en el índice de Elasticsearch. Cabe resaltar que es necesario realizar esta acción sobre los documentos si se quiere realizar la clasificación por indexación más adelante, de lo contrario los resultados no arrojarán coincidencias con los nuevos documentos.

Paso 2 – Configurar el clasificador

Después de tener agregados los documentos necesarios para crear el modelo, se prosigue a realizar la configuración del método de clasificación, al seleccionar la opción de *Ir a configurar* el clasificador (ver **Figura 31**) se puede ver en la parte superior que se puede seleccionar el método de clasificación por *Indexador* o *Clasificador*.

Figura 29 *Filtrado de texto en un documento hecho por un entrenador*

Figura 30 *Vista de documentos de un modelo*

Figura 31 *Selección del método de clasificación para un modelo*

Al seleccionar el método *Indexador se* indica que se va a usar el índice como forma para realizar las predicciones dentro del modelo. Seleccionar el método *Clasificador* indica que se va a configurar un método basado en machine learning (K-NN, Naïve Bayes Gaussiano, Bosques aleatorios y perceptrón multicapa), y para cada algoritmo se pueden fijar sus parámetros (ver **Figura 32**, **Figura 33** y **Figura 34**).

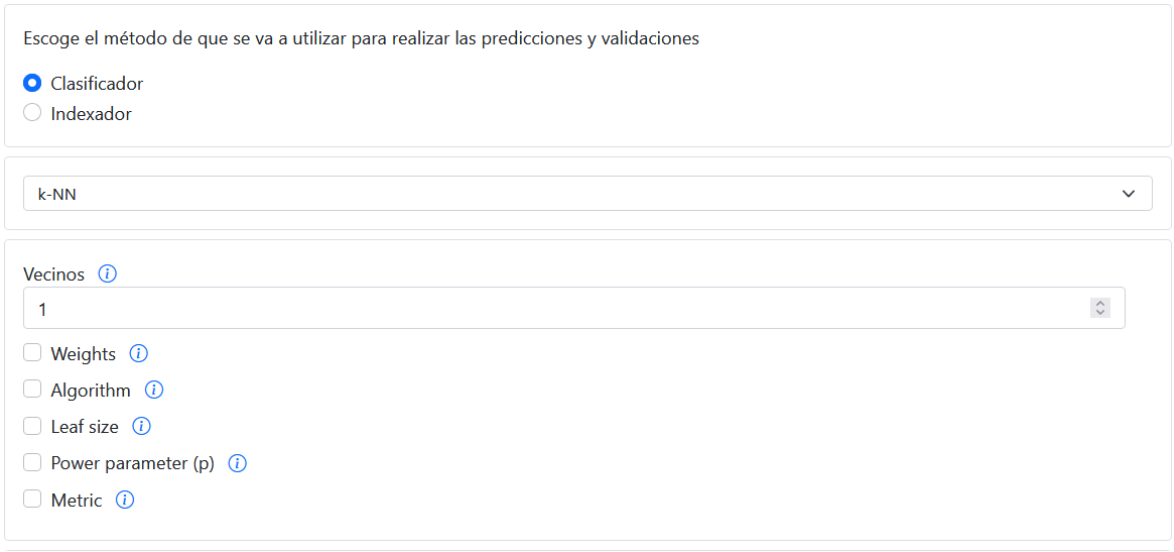

Figura 32 *Interfaz para fijar parámetros del algoritmo K-NN*

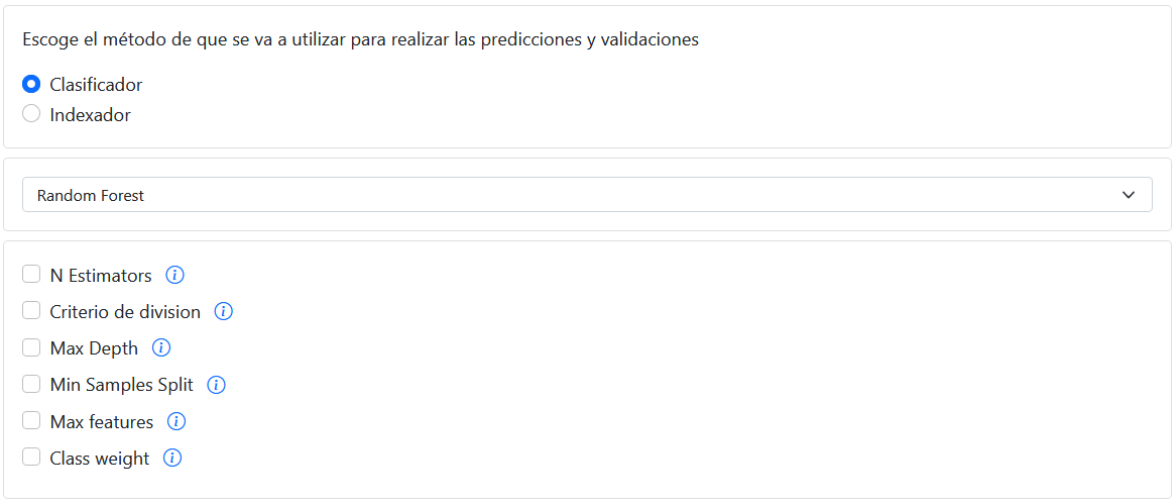

Figura 33 *Interfaz para fijar parámetros del algoritmo Random Forest*

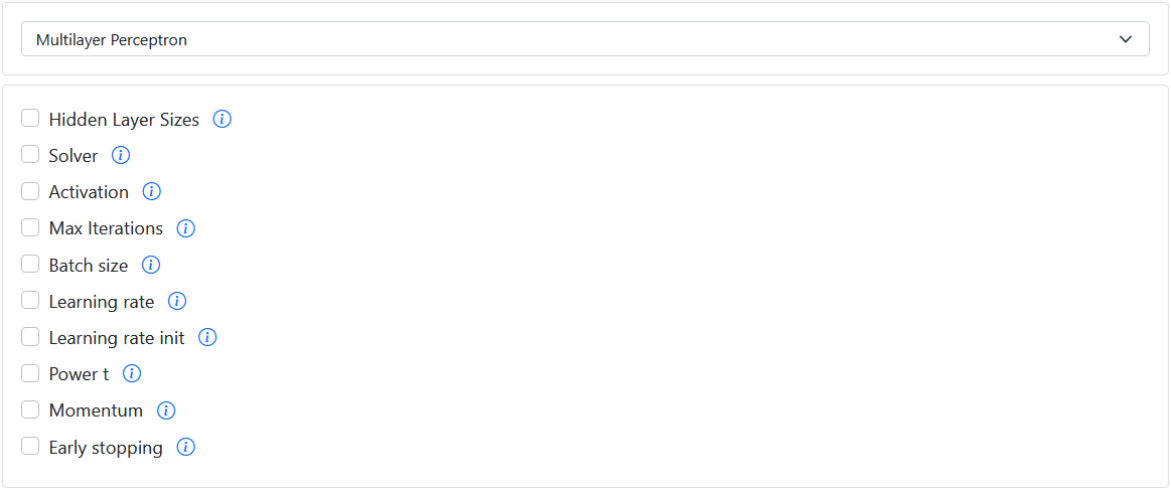

Figura 34 *Interfaz para para fijar parámetros del algoritmo Multilayer Perceptron*

Para el clasificador Naïve Bayes Gaussiano no se tienen parámetros para afinar por la naturaleza misma del clasificador. Cuando se quiere afinar un parámetro se marca la casilla del parámetro correspondiente y se activa un campo adicional en el cual se ingresa el valor para ese parámetro. Por último, para cualquiera de los clasificadores, opcionalmente se puede activar utilizar n-gramas para generar el data set, y obligatoriamente se tiene que escoger la estrategia de división (*Divide strategy*), con dos posibles opciones (ver **Figura 35**). La primera opción es dividir por porcentaje de cantidad de datos para probar, por ejemplo, 30% de los datos para realizar las pruebas (validación) de predicción y 70% para realizar el entrenamiento del modelo. La segunda opción es la validación cruzada, para el cual se debe especificar el número de carpetas o folders (*folds)*.

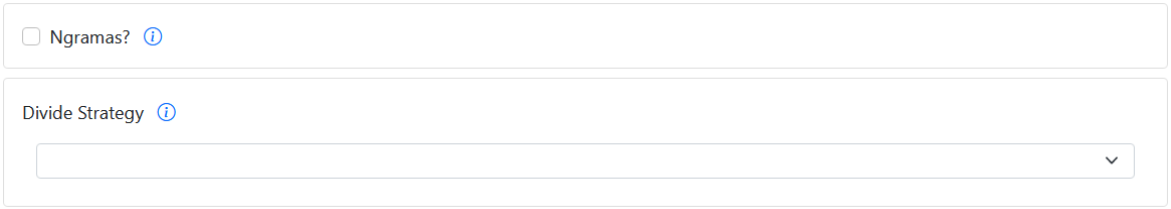

Figura 35 *Interfaz de variación de parámetros para entrenamiento*

Una vez terminado de afinar los parámetros para el clasificador seleccionado este se guarda y el sistema realiza el entrenamiento y validación del método de machine learning con los datos de los documentos que se han ingresado en el paso anterior, y el sistema arroja un porcentaje de precisión del modelo configurado.

Paso 3 – Predicción y Validación

El modelo debe ser probado con el clasificador y parámetros configurados, para esto se dan las opciones de: *Ir a predicción*, *Ir a validación, e Ir a verificaciones* (ver **Figura 36**).

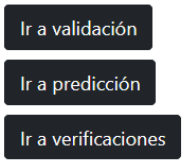

Figura 36 Opciones para probar el modelo

En la opción de predicción, se pueden subir uno o más documentos en formato PDF, JPG, JPEG o PNG para después predecir en qué clases se presenta mayor similitud por cada página del documento subido. En la **Figura 37** se presenta un ejemplo de un resultado para cuando se sube una cédula (documento que se agregó en el ejemplo del paso 2). Se muestran los resultados ordenados por archivo y luego por páginas, por cada página se muestran las predicciones de secciones y un valor que representa si se escoge como clase o no. Para el caso de realizar la predicción por Elasticsearch se muestra la calificación obtenida en la búsqueda sobre el índice texto de la página, la predicción con calificación más alta es la seleccionada como predicción final. Ahora, para el caso de utilizar clasificadores de machine learning se muestra el porcentaje de precisión que se obtuvo en la clasificación de la sección, y el mayor porcentaje se escoge como la predicción final.

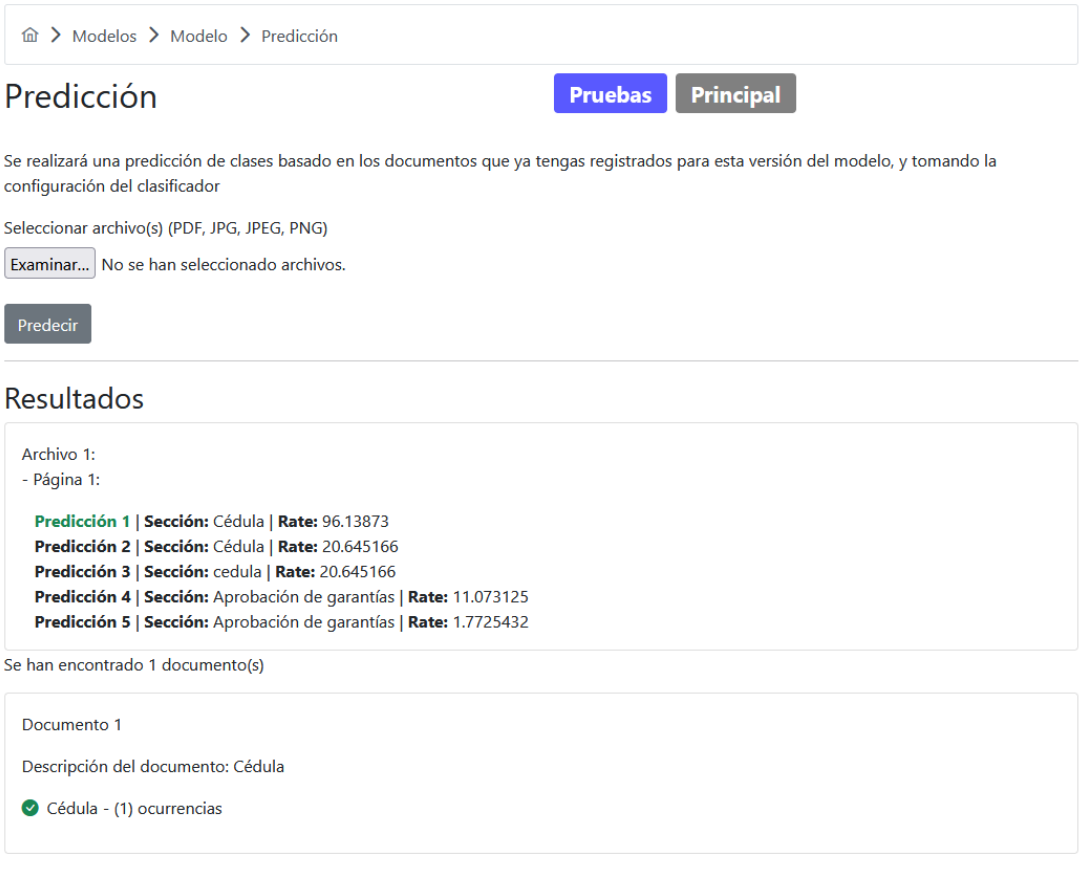

Figura 37 Interfaz de predicción con un ejemplo de resultados para tipo de documento que ya ha sido agregado en el modelo

La opción de validación permite cargar varios archivos y seleccionar a qué clases de documentos se espera que pertenezcan, una vez realizado el proceso de validación sobre los documentos, el sistema indica si se encontraron las clases seleccionadas o no. Un ejemplo de validación para un archivo que contiene un documento con clase *aprobación de garantías* se puede ver en la **Figura 38**, en la cual se selecciona una clase de un documento que ha sido agregado anteriormente y se carga un formato para validar si realmente se encuentra o no.

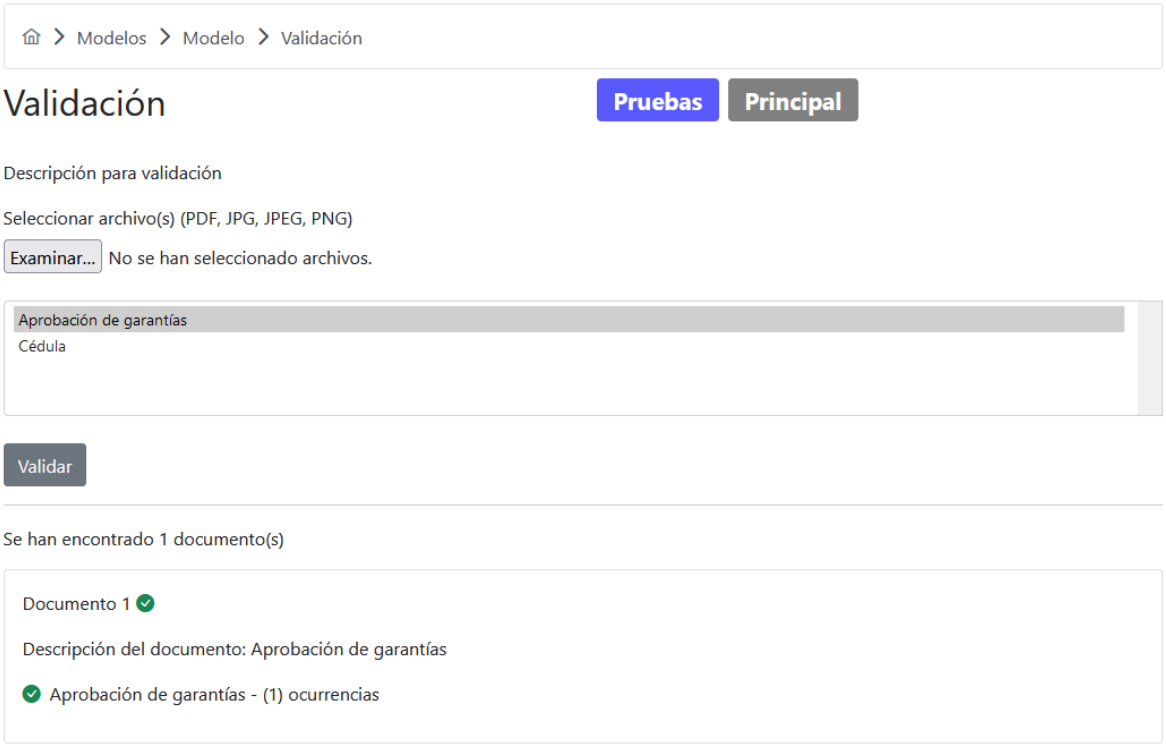

Figura 38 *Interfaz de validación con un ejemplo de resultados para un tipo de documento que ya ha sido agregado en el modelo*

La última opción de la lista son las verificaciones, este es un historial de todas las predicciones y validaciones que se han hecho sobre el modelo (ver **Figura 39**). Sobre cada verificación se dan detalles de la fecha, el modelo, el usuario que realizó la verificación, el tipo de verificación que se realizó y una opción para ver los detalles de los resultados.

Paso 4 – A producción

Una vez el entrenador esté satisfecho con los resultados de las predicciones y validaciones que ha realizado con la configuración actual del clasificador, este puede aplicar los cambios y migrar su configuración actual a un ambiente de producción. Desde la página principal del modelo se usa la opción de *Mover al principal haciendo* una copia de los documentos y configuración del modelo al ambiente principal. Terminada la migración de estos datos, se podrá visualizar en la pestaña de ambiente principal el modelo (ver **Figura 40** y **Figura 41**).

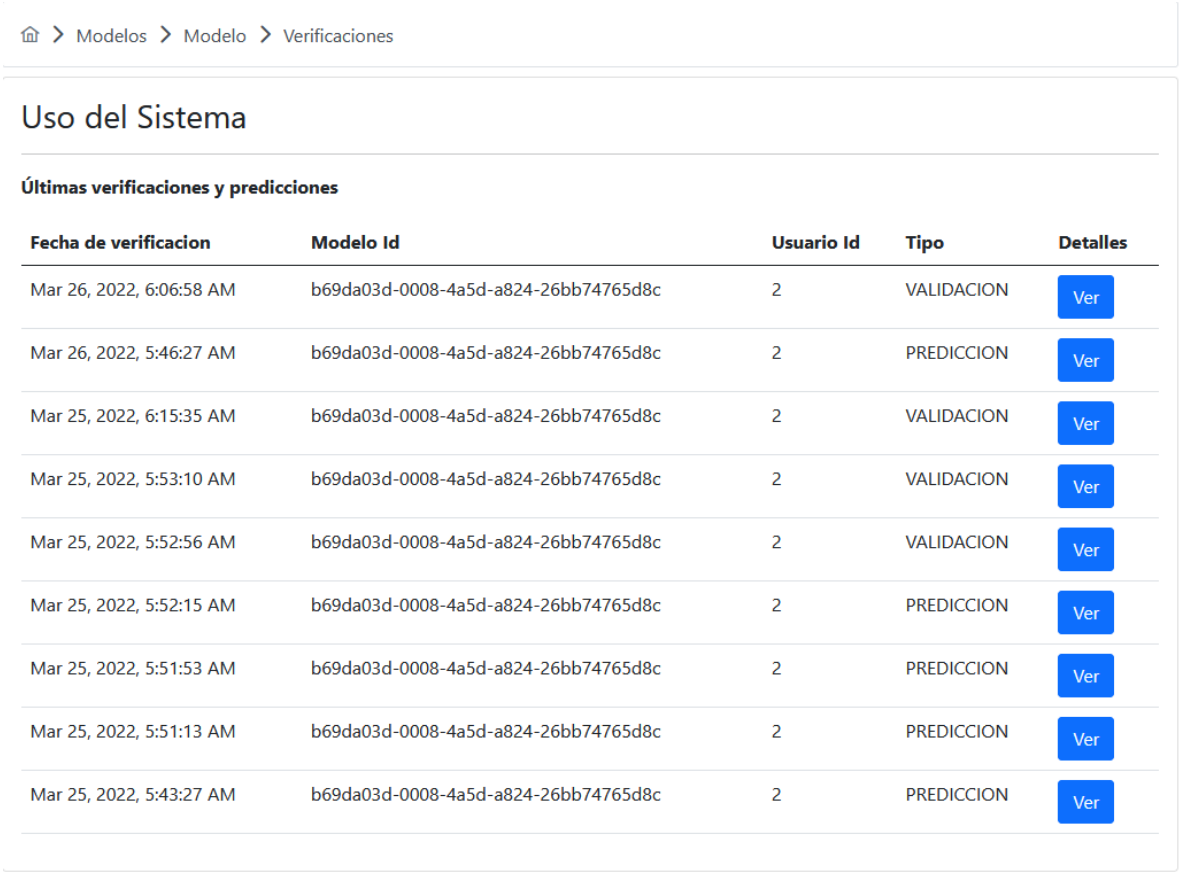

Figura 39 Lista de últimas verificaciones hechas sobre el modelo

Figura 40 Detalles del modelo en ambiente de pruebas

Figura 41 Detalles del modelo en ambiente principal

4.1.2 ARQUITECTURA DEL BACK-END

En una arquitectura de gran escala, donde existen diferentes servicios enviando solicitudes entre ellos, es necesario tener una forma de autenticación y autorización entre ellos, por esto se utiliza la estrategia de JSON Web Tokens en todos los microservicios.

API Gateway: Como patrón de arquitectura se creó el microservicio de API Gateway, el cual es el encargado de recibir las peticiones desde el Front-end, permitiendo tener un mejor control sobre las peticiones que se realizan al Back-end, ya que al tener un solo punto de entrada y uno de salida se disminuye la probabilidad

de posibles errores a comparación de una solución donde se pueda acceder directamente a todos los microservicios. Además, se puede autenticar al usuario desde el primer momento en que llega una petición. Cada vez que se realice una petición desde el Front-end esta es recibida por el API Gateway y este redirige la petición a los servicios correspondientes. Este microservicio no contiene datos almacenados, sus funciones se limitan a redirigir y autorizar las peticiones que vienen de fuera del ambiente y fue desarrollado en Springboot y desplegado en un contenedor Docker.

Microservicio de Descubrimiento: En soluciones basadas en microservicios, es común que los servicios deban comunicarse con los otros, cada servicio puede estar en diferente ubicación y esta puede cambiar constantemente. Para resolver este problema de comunicación entre servicios, se usó el microservicio de *descubrimiento por parte del cliente* (un microservicio de descubrimiento, eureka server), y los demás servicios dentro del ecosistema se registran ante este y cuando se hacen llamadas de un servicio a otro se pregunta primero al microservicio de descubrimiento el cual resuelve su ubicación y disponibilidad. Para hacer esto posible, cada servicio debe conocer la dirección exacta del microservicio de descubrimiento. Este limita al pleno funcionamiento del sistema, por lo tanto, no contiene manejo de base de datos, fue desarrollado en Springboot y desplegado en un contenedor de Docker.

Microservicio de Configuración: Se creó un microservicio de configuración para ayudar a guardar información que no puede ser almacenada en los microservicios, por ejemplo, la clave que se utiliza para encriptar contraseñas, y también el usuario y contraseña necesaria para realizar peticiones desde un cliente (autenticación básica). Cuando un microservicio necesita de estas variables se realiza una petición al servicio, y al estar desacoplado de los otros microservicios se evita tener redundancia en estas variables. También es posible tener diferentes perfiles de estas propiedades para poder separar las variables que se utilizan en desarrollo y las que se utilizan en producción.

Microservicio de autenticación y autorización: También mencionado en los patrones de microservicios, es necesario un patrón de autenticación y autorización para restringir el acceso a los servicios. Para esto se creó el microservicio de autenticación y autorización, el cual es el encargado de crear claves en formato JSON a partir de una clave secreta, este recibe datos del email y la contraseña del usuario, luego hace un llamado al servicio de usuarios para verificar la autenticidad de las credenciales para luego poder generar un token en formato JSON que luego es utilizado por el Front-end para realizar otras peticiones. Este token además de contener información del usuario tiene el identificador de la empresa al que pertenece, es decir, contiene información del tenant, ya que es necesario identificar dentro del sistema qué inquilino está realizando peticiones. Además, el token generado tiene expiración de una hora, es decir que después de una hora de haberse generado el token ya no sirve para realizar peticiones y se debe realizar la

petición para generar un nuevo token. Al tener una duración relativamente corta, si un usuario está realizando acciones sobre el sistema como entrenar un clasificador, o realizando verificaciones, este no tendrá más opción que volver a iniciar sesión; por lo tanto, para trabajos futuros se plantea implementar una estrategia para estos casos. Este microservicio también tiene un punto de entrada diferente cuando se requiere realizar autenticación desde otros servicios, es decir, cuando un microservicio recibe una petición siempre debe verificar la autenticidad del token para acceder a sus recursos.

Microservicio de Usuarios: Este microservicio es el encargado de gestionar la información de los usuarios. El modelo físico para esta base de datos se presenta en la **Figura 42**. Este microservicio contiene un punto de acceso que permite al servicio de autenticación y autorización obtener información de un usuario por su email, incluyendo el identificador de la empresa al cual pertenece.

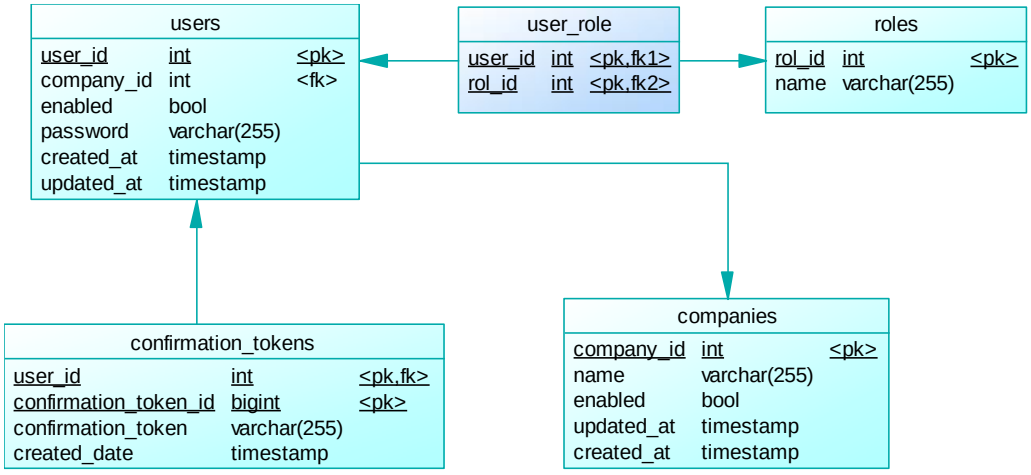

Figura 42 *Modelo físico de base de datos de usuarios*

Microservicio de Entrenamiento: Este microservicio gestiona la información correspondiente a los modelos y documentos de cada empresa. También realiza peticiones a los servicios elastic service, text extraction service y classification service. Para realizar un entrenamiento se necesita el texto de los documentos base, por esto el servicio realiza un llamado al servicio de extracción de texto para extraer esta información para luego poder mostrarla al usuario final. Una vez el usuario realiza el filtro del texto, este ejecuta el llamado al servicio de Elasticsearch en caso de que el método de entrenamiento seleccionado sea por indexación, o al servicio de clasificación en caso de que el método de entrenamiento sea por algoritmos de machine learning. Este servicio también realiza la gestión de la información de los modelos por lo que realiza peticiones al servicio de Elasticsearch y de clasificación. En la **Figura 43** se puede ver el modelo físico de la base de datos que soporta este microservicio.

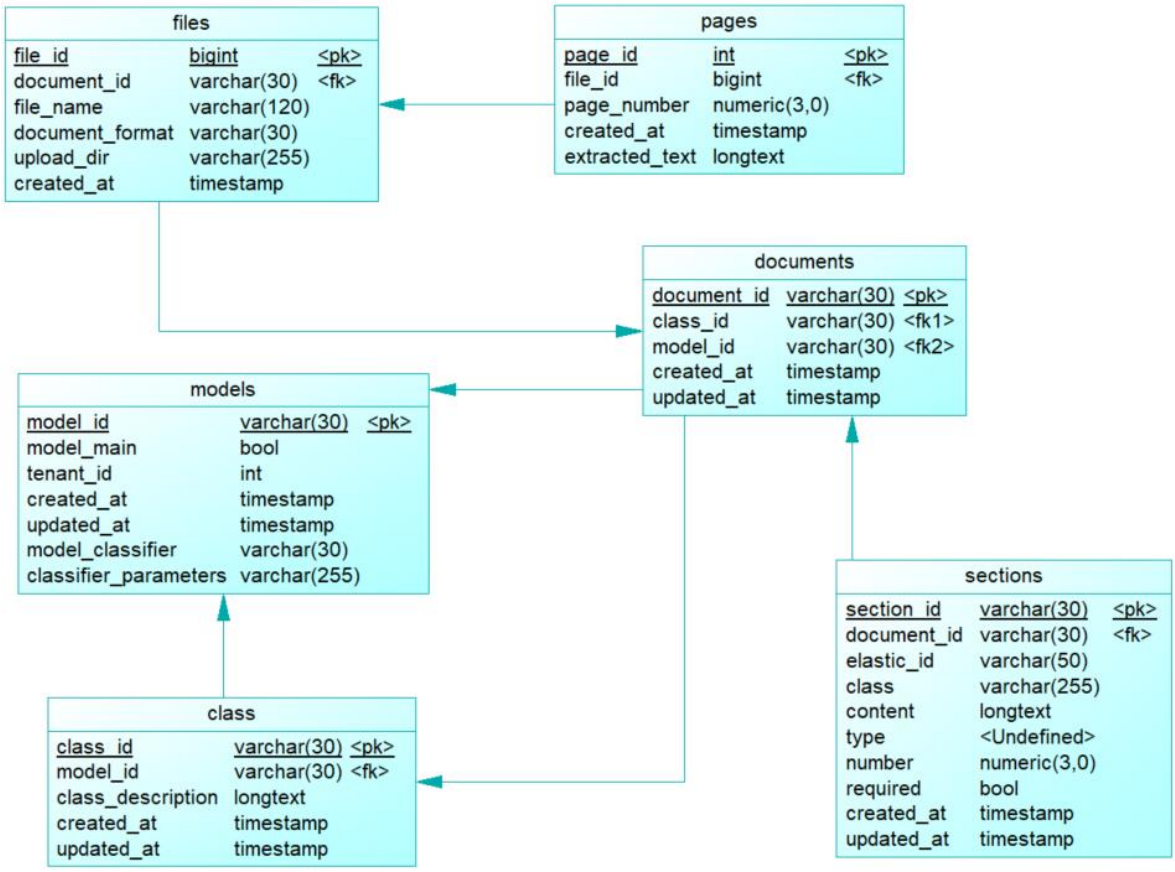

Figura 43 *Modelo físico de base de datos de entrenamiento*

Microservicio Verificación: Este microservicio es el encargado de ejecutar validaciones y predicciones sobre modelos creados y entrenados o indexados anteriormente. Para esto, se guarda cada verificación que un usuario realiza con sus respectivos resultados determinados en valores de precisión para el caso de clasificadores machine learning, y el puntaje de la búsqueda para el caso de clasificador basado en índice. Este microservicio fue desarrollado en Springboot y desplegado en un contenedor de Docker. En la **Figura 44** se puede ver el modelo físico de la base de datos que soporta este microservicio.

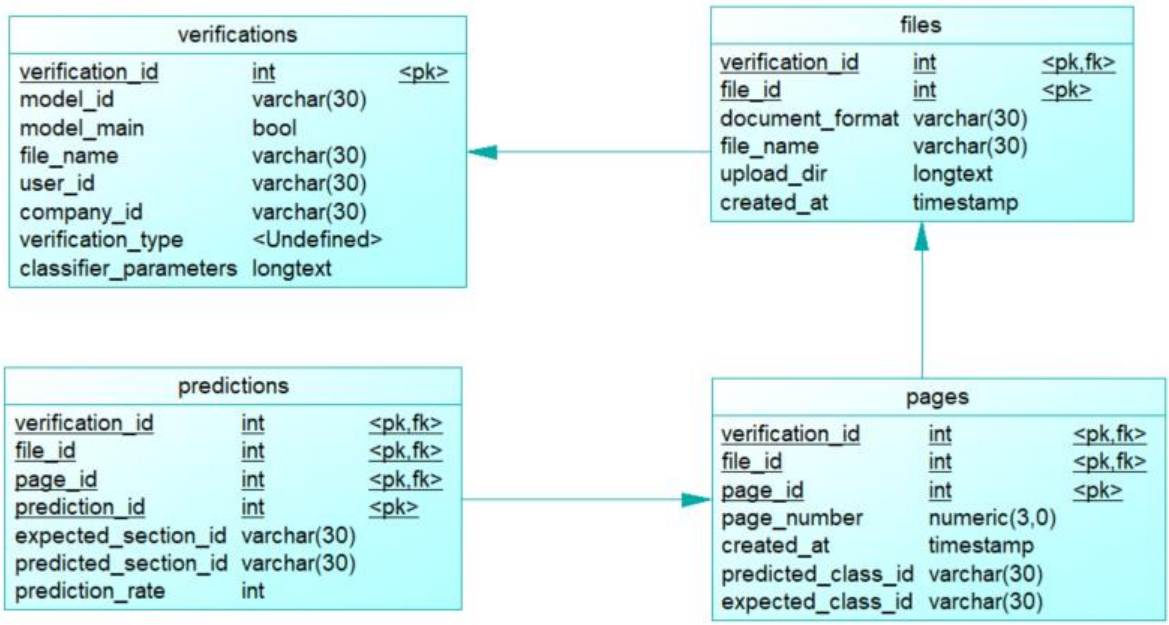

Figura 44 *Modelo físico de base de datos de verificaciones*

Microservicio de Clasificación: Es el encargado de realizar los entrenamientos y predicciones de clasificadores basados en machine learning. Para el procedimiento de entrenar un clasificador, se reciben las frases de una página junto con la clase, el clasificador escogido (Random Forest, Multilayer Perceptron, Naive Bayes o k-NN) y los parámetros para ejecutar el clasificador. Luego se realiza un preprocesamiento al texto, excluyendo palabras que no son necesarias (stop words). Con esto el servicio puede entrenar el clasificador especificado para después guardarlo y utilizarlo cuando se requiera.

Para el procedimiento de predecir, también se reciben las frases, pero sin clasificar, junto con un valor que identifica el clasificador que ha sido guardado y entrenado anteriormente. Al igual que en el entrenamiento, al texto se le realiza un preprocesamiento excluyendo palabras que no son necesarias. Con esto se realiza la predicción de clases de las frases de los documentos y se extrae el porcentaje de precisión de cada resultado.

Este microservicio fue desarrollado en el lenguaje python con el framework de desarrollo web Flask, debido a que la librería utilizada para utilizar clasificadores de machine learning está desarrollada en Python (sci-kit learn) [53] y se hace mucho más sencillo hacer el llamado directo desde el lenguaje base.

Microservicio de Extracción de texto: Este microservicio se encarga de extraer textos de documentos que están en formato PDF o imagen; para poder realizar esto, se hace uso del servicio OCR de Google [9]. Debido a que el servicio de verificación y entrenamiento necesitan realizar extracción de texto sobre documentos, se

decidió separar esta funcionalidad en un microservicio aparte. Además, para poder utilizar el servicio OCR de Google es necesario guardar las credenciales directamente en variables de entorno en el sistema, por esta razón, solo fueron configuradas en el contenedor de Docker de este servicio.

Instancia de Elasticsearch: Esta es una instancia local del servicio Elasticsearch que fue desplegada en un contenedor Docker.

Microservicio Elastic: Por último, está el servicio elastic, su función consiste en realizar la comunicación con la instancia de Elasticsearch. Se implementó un servicio aparte para no utilizar directamente la instancia de Elasticsearch, ya que un usuario debe ser identificado antes de poder tener acceso a los índices. La tarea principal de este servicio consiste en ejecutar las acciones sobre la instancia de Elasticsearch, incluyendo la creación, indexación, eliminación de documentos y ejecución de consultas sobre los índices.

CAPÍTULO 5

5 RESULTADOS DE EVALUACIÓN DEL SISTEMA

Este capítulo presenta el proceso de evaluación realizado al modelo y prototipo de la aplicación mediante la aplicación de una encuesta de satisfacción del cliente a la empresa colaboradora en el proyecto ATIX Digital, se presenta cómo se compone la encuesta y finalmente los resultados obtenidos.

5.1 PARTICIPANTES

En este caso, el prototipo final de la aplicación fue evaluado con la ayuda de la empresa ATIX Digital S.A.S, que cuenta con experiencia en el área de clasificación de documentos y algoritmos de inteligencia artificial. Esta empresa tiene un producto software que realiza la clasificación de documentos con algoritmos basados en aprendizaje de máquina y que obtiene muy buenos resultados cuando cuenta con una gran cantidad de información de entrenamiento. Además, los tres (3) colaboradores expertos que participaron en la evaluación son autores de este producto lo cual permitió una mayor retroalimentación.

5.2 IMPLEMENTACIÓN

La encuesta fue tomada de las plantillas existentes de la página web de www.encuestafacil.com denominada satisfacción del Cliente (Producto) y fue modificada para que el usuario defina dentro de un índice CSAT [54] un nivel numérico para la característica satisfacción de la ISO 9001:2015, en el cual se define un rango de 1 a 5 según el nivel de satisfacción del encuestado donde el número 5 es el mayor y el número 1 es el más bajo (ver **Tabla 9**).

Tabla 9 *Escala de calificación definida por el índice CSAT*

Valor numérico	Nivel de satisfacción
1	Completamente insatisfecho
2	Insatisfecho
3	Neutral
4	Satisfecho

5	Completamente satisfecho

Las preguntas para la encuesta fueron modificadas de acuerdo con la evaluación de las funcionalidades que posee la aplicación y se dividieron en cuatro componentes:

- **Grado de satisfacción general de la aplicación**: Se realizaron dos preguntas para calificar el grado de satisfacción general de la aplicación. (Utilizaría de nuevo la aplicación, y nivel de satisfacción con el uso de la aplicación).
- **Características principales**: Preguntas específicas para las funcionalidades principales de la aplicación (Gestión de modelos, entrenamiento y validación).
- **Características adicionales**: Sobre funcionalidades adicionales al núcleo de la aplicación (Creación de subcuentas, algoritmos de machine learning y manejo de histórico de validaciones y verificaciones).
- **Retroalimentación**: Un espacio para que los encuestados describan posibles mejoras al sistema.

5.3 APLICACIÓN DE LA PRUEBA

Antes de aplicar la encuesta se aseguró que cada interesado conociera las funcionalidades realizadas y el estado actual de la aplicación, por lo cual el primer paso consistió en realizar una exposición mediante una presentación de la aplicación. En dicha exposición se presentó una introducción al problema, el objetivo y el alcance del trabajo de grado. Luego, se realizó la demostración de la aplicación y cuál sería el flujo de trabajo para un usuario que acaba de ingresar. Como se mencionó anteriormente, los encuestados cuentan con experiencia en el área de clasificación de documentos así que dominan el tema y, por lo tanto, durante la presentación surgen preguntas de cómo funciona el sistema por dentro en algunas de las características, entre estas, el método de clasificación usando el índice y la integración también de los métodos basados en machine learning. Al final de la exposición se ven muy interesados en el método propuesto. Se verifica las funcionalidades de creación de modelos, sus documentos y cómo se puede sacar ventaja del método al registrar secciones de documentos.

Los encuestados verifican que tan fácil es realizar el entrenamiento de los modelos por indexación, al igual que con clasificadores de machine learning. También durante la exposición se mostraron las funcionalidades secundarias de gestión de subcuentas e históricos de verificaciones.

Finalmente, se realizó la retroalimentación respectiva de la aplicación tratando entre otros temas la forma cómo se aborda el manejo de clases incrementales para cuando se ingresan nuevos documentos, y debido a que los encuestados tienen

experiencia en el tema, plantean diferentes escenarios de entrenamiento para ver cómo actuaría el sistema, por ejemplo, cuando dos clases de documentos son muy similares.

5.4 RESULTADOS

En la **Figura 45** se muestran los resultados para la pregunta que busca confirmar el grado de satisfacción general de los expertos con la aplicación. Los resultados obtenidos para todos los encuestados fue de satisfecho, que en la escala de calificación corresponde a 4.

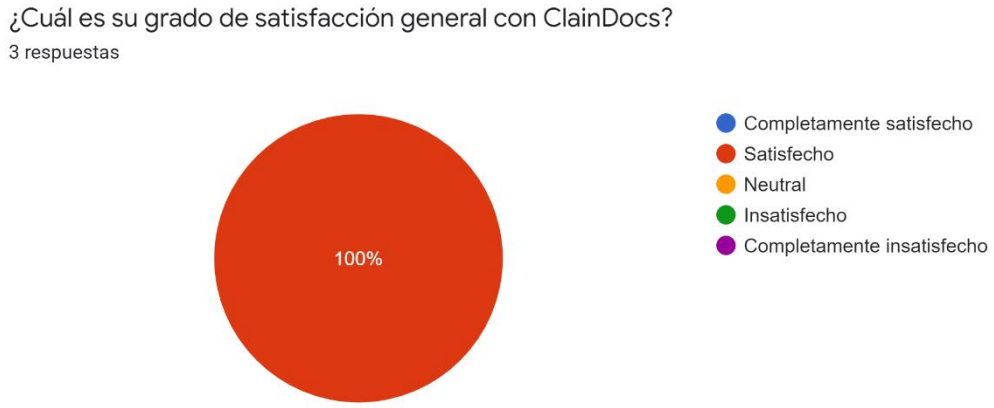

Figura 45 *Pregunta de satisfacción general de la aplicación*

En la **Figura 46** se muestran los resultados para la pregunta que buscó definir el nivel de satisfacción con respecto a las tres funcionalidades principales de la aplicación. La primera, gestión de modelos muestra un grado de satisfacción entre satisfecho y totalmente satisfecho. La segunda muestra el grado de satisfacción con el entrenamiento de los modelos el cual incluye la forma en que se crean los documentos para cada modelo y el entrenamiento de algoritmos de machine learning con un resultado que esta para los 3 casos en satisfecho. La última busca evaluar la manera en que se realizan verificaciones de los modelos a través de predicciones y validaciones; los resultados de este ítem estuvieron entre totalmente satisfecho y satisfecho.

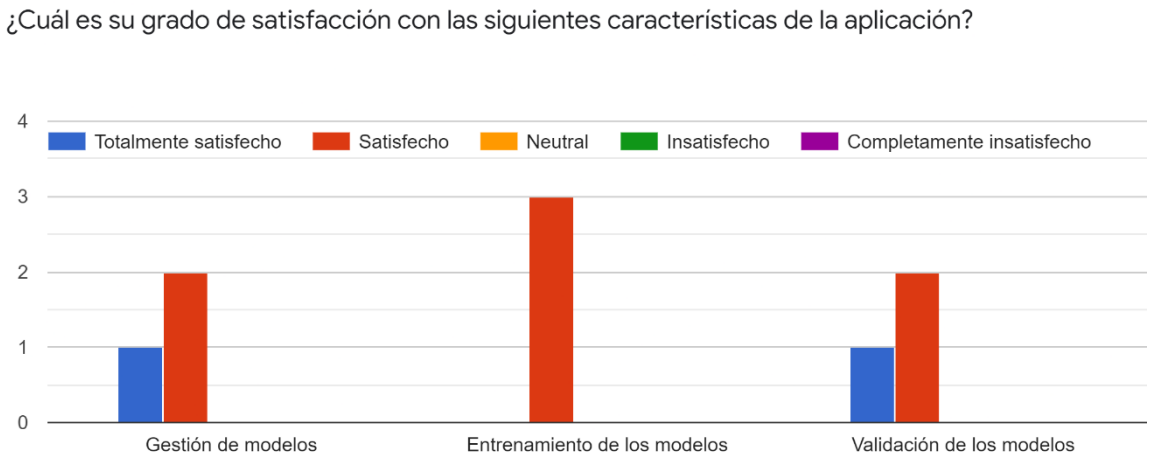

Figura 46 *Pregunta de satisfacción con respecto a funcionalidades principales*

Los resultados de la pregunta presentada en la **Figura 47** muestran la satisfacción con la funcionalidad secundaria de la aplicación (poder crear subcuentas para delegar funcionalidades). Los resultados estuvieron entre totalmente útil y muy útil.

Figura 47 *Pregunta de satisfacción con funcionalidad secundaria de creación de subcuentas*

En los resultados de la pregunta presentada en la **Figura 48** se buscó definir el grado de satisfacción de los usuarios con la característica de poder utilizar otro tipo de clasificadores para realizar la clasificación (los clasificadores de machine learning). Todos los encuestados respondieron que era totalmente útil.

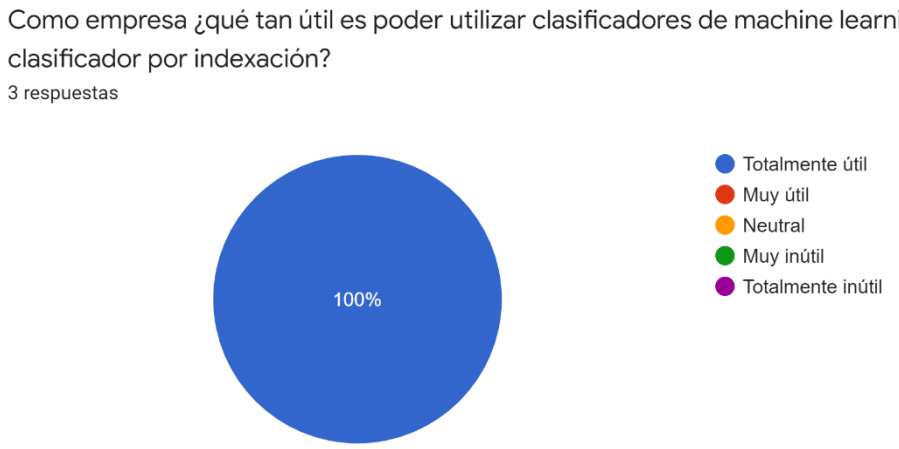

Figura 48 *Pregunta de satisfacción con funcionalidad secundaria de uso de clasificadores de machine learning*

En la **Figura 49** se muestran los resultados de la pregunta que se buscó definir el grado de satisfacción de los usuarios con respecto a la característica de visualización del histórico de predicciones y validaciones hechas desde la aplicación. Las respuestas se encuentran en el rango de totalmente útil y muy útil.

Figura 49 *Pregunta de satisfacción con funcionalidad secundaria de histórico de predicciones y validaciones*

Después de la evaluación de las características principales, se abrió un espacio para que los evaluadores dieran a conocer posibles recomendaciones de cambio para las características existentes en la aplicación conforme fue presentada (ver **Figura 50**). Se obtuvo una única respuesta en la cual se recomienda tener en cuenta el orden de las palabras cuando se realiza el entrenamiento de las secciones de los documentos. Esta recomendación de cambio queda planteada como trabajo futuro para el grupo de investigación.

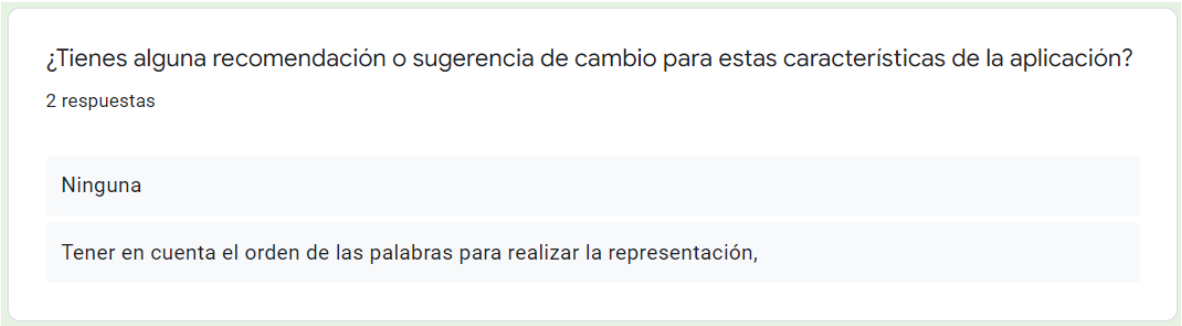

Figura 50 *Pregunta de recomendaciones de cambio para características principales de la aplicación*

Con la última pregunta (**Figura 51**), se buscó evaluar si la aplicación es lo suficientemente útil para el usuario como para que vuelva a utilizarla en un futuro. Las respuestas estuvieron en el rango de seguro que sí y probablemente sí.

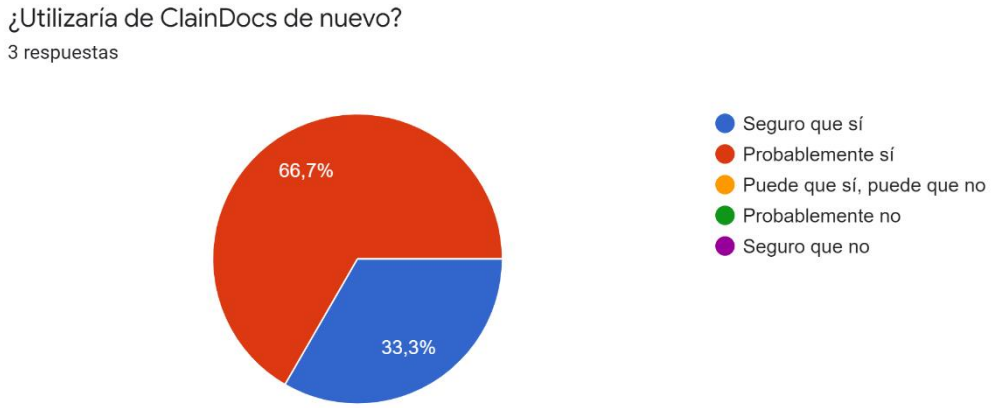

Figura 51 *Pregunta de satisfacción general ¿utilizaría de nuevo la aplicación?*

Por último, se calculó el porcentaje de satisfacción del usuario por cada pregunta del formulario mediante el índice CSAT, para esto se sumaron todos los interesados que marcaron las puntuaciones más altas, en un rango de 1 a 5, las que marcaron 4 y 5, y se dividen entre el número total de personas que participaron en la encuesta. Los resultados se muestran en la **Tabla 10**.

Tabla 10 *Porcentaje de satisfacción del encuestado por cada pregunta*

Satisfacción del interesado para preguntas específicos	
Numero pregunta	**Satisfacción del interesado**
1	100%
2.1	100%
2.2	100%
2.3	100%
3	100%
4	100%

5	100%
6	100%

Luego de calcular el porcentaje de satisfacción por pregunta se buscó un resultado numérico general de la encuesta. Para esto, por cada pregunta se calcula el promedio de la puntuación CSAT (ver **Tabla 11**) y luego cada resultado se suma con el promedio de las demás preguntas para finalmente calcular el puntaje general, el cual resulta ser de 4.4, lo que en general indica un alto grado de satisfacción con la aplicación realizada. Es preciso comentar que esta apreciación no es generalizable dada la baja cantidad de expertos evaluadores, pero si es un indicio de que las funcionalidades provistas son las deseables en un entorno empresarial.

***Tabla 11**. Promedio de satisfacción de los interesados por pregunta.*

Promedio resultante por pregunta	
Numero pregunta	**Promedio respuestas**
1	4
2	4.2
3	4.6
4	5
5	4.3
6	4.3

CAPÍTULO 6

6 CONCLUSIONES Y TRABAJO FUTURO

En el presente trabajo de grado se propuso un método de clasificación de documentos basado en indexación con poca información y con clases que incrementan en el tiempo, que puede ser aplicado en las empresas para disminuir el reproceso de distintos trámites que no entregan la documentación apropiada. El método permite que la estructura de los documentos puede cambiar en el tiempo, por medio de la modificación o eliminación de estos en los índices sin afectar otros documentos. También se contempla que un documento pueda ser obligatorio u opcional lo cual puede influir en la clasificación de estos. Además, se generalizó la estructura de las secciones internas de un documento en dos categorías: una página que no cambia de estructura o una sección que no debe tener un número fijo de páginas.

De la evaluación realizada al método se puede concluir que este es sensible al incremento de clases, es decir, cuando se agregan muchas clases de documentos distintos las métricas de calidad disminuyen. Se espera que el sistema funcione de forma similar un sistema de huellas digitales, donde si se agregan más huellas en el futuro no se presenta deterioro en la precisión de este.

Se desarrolló y se desplegó una aplicación web basada en una arquitectura de microservicios que permite realizar el entrenamiento (indexación) de modelos con el método propuesto. Desde la misma aplicación se pueden verificar los modelos entrenados, luego ingresando archivos reales se puede predecir (clasificación basada en recuperación de información) a qué clase pertenece. Adicionalmente se contempla el uso de clasificadores basados en aprendizaje de máquina, lo cual no estaba definido en el anteproyecto, sin embargo, este módulo es un buen complemento de la aplicación ofreciendo más alternativas a las empresas por medio de la comparación de diferentes modelos con respecto a la precisión en la clasificación de documentos.

La investigación sobre patrones de diseño para usar microservicios no está estandarizada, pues existen diferentes puntos de vista. Este proyecto se basó en los patrones de microservicios propuestos por Richardson [52], teniendo en cuenta que presenta una gran diversidad de patrones y una descripción completa de los mismos.

El prototipo de la aplicación web fue analizado y evaluado por usuarios expertos en el tema clasificación de documentos y algoritmos de inteligencia artificial. Para la evaluación se aplicó una encuesta basada en el índice CSAT y como resultado se obtuvo una calificación de 100% en la escala de satisfacción del cliente. Adicionalmente, los encuestados proporcionaron comentarios y sugerencias de posibles mejoras en el método, por ejemplo, tener en cuenta la importancia del orden de las oraciones dentro de un documento, o que una oración específica sea muy relevante para la clasificación y por lo tanto se les pueda asignar pesos a cada una.

Esto también es un primer paso a que clientes finales de las empresas ahorren tiempo y dinero al no tener que desplazarse hasta una ventanilla física para la verificación de sus documentos, sino que lo pueda hacer a través de una aplicación de la empresa tramitadora que con anterioridad ha entrenado sus modelos y ha puesto a disposición un servicio que puede ser usado por sus clientes.

Como trabajo futuro también se busca evaluar más extensivamente el método como una solución al problema de clasificación de documentos de varias páginas (Multi Page Document Classification) y asegurar su uso con documentos que tienen secciones estructuradas, semi estructuradas y totalmente des estructuradas. También, implementar un servicio REST que integre la autenticación de un usuario/empresa y enviando el documento desde su propio sitio web pueda recibir una respuesta sobre la clasificación del documento, sin tener que entrar a la aplicación. Además, se plantea investigar cómo se puede mejorar el método de tal manera que no sea afectado por el incremento de clases en el tiempo. Finalmente, tener en cuenta la retroalimentación de los usuarios sobre el clasificador para que el método pueda reentrenarse por sí solo, buscando mejorar los resultados de clasificación.

CAPÍTULO 7

7 BIBLIOGRAFÍA

[1] "Boletín anual de estadísticas de trámites 2018." .

[2] "La transformación digital como motor de la banca latinoamericana." .

[3] "Banco de Bogotá | Informe de gestión 2018 | Capítulo 4." .

[4] "El fin del trámite eterno: Ciudadanos, burocracia y gobierno digital | Publications." .

[5] "Trámites virtuales se dispararon durante la pandemia." https://www.bluradio.com/economia/tramites-virtuales-se-dispararon-durante-la-pandemia-262182-ie3509872 (accessed Aug. 18, 2020).

[6] M. Thangaraj, "Text Classification Techniques: A Literature Review," vol. 13, pp. 117–135, 2018.

[7] K. Kowsari, K. J. Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, "Text classification algorithms: A survey," *Information (Switzerland)*, vol. 10, no. 4. pp. 1–68, 2019, doi: 10.3390/info10040150.

[8] S. K. Dwivedi and C. Arya, "Automatic text classification in information retrieval: A Survey," *ACM Int. Conf. Proceeding Ser.*, vol. 04-05-Marc, 2016, doi: 10.1145/2905055.2905191.

[9] "Detect text in images | Cloud Vision API | Google Cloud." https://cloud.google.com/vision/docs/ocr (accessed Aug. 09, 2020).

[10] G. Kiradoo, "Software engineering quality to enhance the customer satisfaction level of the organization," *Int. J. Adv. Res. Eng. Technol.*, vol. 10, no. 3, pp. 297–302, Jun. 2019, doi: 10.34218/IJARET.10.3.2019.028.

[11] ISO, "ISO 9001:2015(es) Sistemas de gestión de la calidad — Requisitos," *https://www.iso.org/obp/ui/#home*, 2021. .

[12] J. Lu, S. Wu, Z. Xiang, and H. Cheng, "Intelligent document-filling system on mobile devices by document classification and electronization," *Comput. Intell.*, no. December 2019, pp. 1–17, 2020, doi: 10.1111/coin.12279.

[13] X. Deng, Y. Li, J. Weng, and J. Zhang, "Feature selection for text classification: A review," *Multimed. Tools Appl.*, vol. 78, no. 3, pp. 3797–3816, Feb. 2019, doi: 10.1007/s11042-018-6083-5.

[14] S. T. Selvi, P. Karthikeyan, A. Vincent, V. Abinaya, G. Neeraja, and R. Deepika, "Text categorization using Rocchio algorithm and random forest algorithm," in *2016 Eighth International Conference on Advanced Computing (ICoAC)*, 2017, pp. 7–12, doi: 10.1109/ICoAC.2017.7951736.

[15] K. L. Gitanjali, "A Novel Approach of Sensitive Data Classification using Convolution Neural Network and Logistic Regression," *Int. J. Innov. Technol.*

Explor. Eng., vol. 8, no. 8, pp. 2883–2886, 2019.

[16] Z. Qu, X. Song, S. Zheng, X. Wang, X. Song, and Z. Li, “Improved Bayes Method Based on TF-IDF Feature and Grade Factor Feature for Chinese Information Classification,” *Proc. - 2018 IEEE Int. Conf. Big Data Smart Comput. BigComp 2018*, pp. 677–680, 2018, doi: 10.1109/BigComp.2018.00124.

[17] S. Xu, “Bayesian Naïve Bayes classifiers to text classification,” *J. Inf. Sci.*, vol. 44, no. 1, pp. 48–59, 2018, doi: 10.1177/0165551516677946.

[18] W. Qin, W. Guo, X. Liu, and H. Zhao, “A Novel Scheme for Recruitment Text Categorization Based on KNN Algorithm,” in *SmartCom 2019: Smart Computing and Communication*, 2019, pp. 376–386, doi: 10.1007/978-3-030-34139-8_38.

[19] D. P. Hapsari, I. Utoyo, and S. W. Purnami, “Text Categorization with Fractional Gradient Descent Support Vector Machine,” in *Journal of Physics: Conference Series*, Mar. 2020, vol. 1477, no. 2, p. 022038, doi: 10.1088/1742-6596/1477/2/022038.

[20] A. I. Taloba and S. S. I. Ismail, “An Intelligent Hybrid Technique of Decision Tree and Genetic Algorithm for E-Mail Spam Detection,” *Proc. - 2019 IEEE 9th Int. Conf. Intell. Comput. Inf. Syst. ICICIS 2019*, pp. 99–104, 2019, doi: 10.1109/ICICIS46948.2019.9014756.

[21] L. Fatima-Ezzahra, Z. Houssaine, and Y. Kettani El, *An Efficient Model of Text Categorization Based on Feature Selection and Random Forests: Case for Business Documents*, vol. 3, no. 2. Springer International Publishing, 2019.

[22] T. Chen, R. Xu, Y. He, and X. Wang, “Improving sentiment analysis via sentence type classification using BiLSTM-CRF and CNN,” *Expert Syst. Appl.*, vol. 72, pp. 221–230, 2017, doi: 10.1016/j.eswa.2016.10.065.

[23] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, “Hierarchical Attention Networks for Document Classification,” in *Proceedings of the 2016 Conference of the North {A}merican Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 1480–1489, doi: 10.18653/v1/N16-1174.

[24] K. Kowsari, D. E. Brown, M. Heidarysafa, K. Jafari Meimandi, M. S. Gerber, and L. E. Barnes, “HDLTex: Hierarchical Deep Learning for Text Classification,” in *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2017, pp. 364–371, doi: 10.1109/ICMLA.2017.0-134.

[25] J. Chen, S. Yan, and K. C. Wong, “Verbal aggression detection on Twitter comments: convolutional neural network for short-text sentiment analysis,” *Neural Comput. Appl.*, vol. 0, pp. 1–10, 2018, doi: 10.1007/s00521-018-3442-0.

[26] M. Jiang *et al.*, “Text classification based on deep belief network and softmax regression,” *Neural Comput. Appl.*, vol. 29, no. 1, pp. 61–70, 2018, doi: 10.1007/s00521-016-2401-x.

[27] Z. Haj-Yahia, A. Sieg, and L. A. Deleris, “Towards unsupervised text classification leveraging experts and word embeddings,” *ACL 2019 - 57th Annu. Meet. Assoc. Comput. Linguist. Proc. Conf.*, pp. 371–379, 2020, doi: 10.18653/v1/p19-1036.

[28] H. Jo and C. Cinarel, “Delta-training: Simple semi-supervised text classification using pretrained word embeddings,” *EMNLP-IJCNLP 2019 - 2019 Conf. Empir. Methods Nat. Lang. Process. 9th Int. Jt. Conf. Nat. Lang. Process. Proc. Conf.*, pp. 3458–3463, 2020, doi: 10.18653/v1/d19-1347.

[29] B. S. Harish, “Text Document Classification: An Approach Based on Indexing,” *Int. J. Data Min. Knowl. Manag. Process*, vol. 2, no. 1, pp. 43–62, 2012, doi: 10.5121/ijdkp.2012.2104.

[30] M. M.S., H. B.S., and R. M.B., *Indexing-Based Classification: An Approach Toward Classifying Text Documents*, vol. 1. Springer Singapore, 2018.

[31] A. M. Aubaid and A. Mishra, “A rule-based approach to embedding techniques for text document classification,” *Appl. Sci.*, vol. 10, no. 11, 2020, doi: 10.3390/app10114009.

[32] J. Novotný and P. Ircing, “The Benefit of Document Embedding in Unsupervised Document Classification,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11096 LNAI, pp. 470–478, Sep. 2018, doi: 10.1007/978-3-319-99579-3_49.

[33] R. Karpinski and A. Belaid, “Combination of Structural and Factual Descriptors for Document Stream Segmentation,” *Proc. - 12th IAPR Int. Work. Doc. Anal. Syst. DAS 2016*, pp. 221–226, Jun. 2016, doi: 10.1109/DAS.2016.21.

[34] A. Hamdi, J. Voerman, M. Coustaty, A. Joseph, V. P. D’Andecy, and J. M. Ogier, “Machine Learning vs Deterministic Rule-Based System for Document Stream Segmentation,” *Proc. Int. Conf. Doc. Anal. Recognition, ICDAR*, vol. 5, pp. 77–82, Jan. 2018, doi: 10.1109/ICDAR.2017.332.

[35] “Multi Page Document Classification using NLP and ML | Doc2Vec | Towards Data Science.” https://towardsdatascience.com/multi-page-document-classification-using-machine-learning-and-nlp-ba6151405c03 (accessed Mar. 09, 2022).

[36] A. Hamdi, M. Coustaty, A. Joseph, V. P. D’Andecy, A. Doucet, and J. M. Ogier, “Feature selection for document flow segmentation,” *Proc. - 13th IAPR Int. Work. Doc. Anal. Syst. DAS 2018*, pp. 245–250, Jun. 2018, doi: 10.1109/DAS.2018.66.

[37] S. S. Bukhari and A. Dengel, “Visual appearance based document classification methods: Performance evaluation and benchmarking,” *Proc. Int. Conf. Doc. Anal. Recognition, ICDAR*, vol. 2015-November, pp. 981–985, Nov. 2015, doi: 10.1109/ICDAR.2015.7333908.

[38] L. P. De Las Heras, O. R. Terrades, J. Llados, D. Fernandez-Mota, and C. Canero, “Use case visual Bag-of-Words techniques for camera based identity document classification,” *Proc. Int. Conf. Doc. Anal. Recognition, ICDAR*, vol. 2015-November, pp. 721–725, Nov. 2015, doi: 10.1109/ICDAR.2015.7333856.

[39] O. Agin, C. Ulas, M. Ahat, and C. Bekar, “An approach to the segmentation of multi-page document flow using binary classification,” *Sixth Int. Conf. Graph. Image Process. (ICGIP 2014)*, vol. 9443, p. 944311, Mar. 2015, doi: 10.1117/12.2178778.

[40] I. Gallo, L. Noce, A. Zamberletti, and A. Calefati, “Deep Neural Networks for Page Stream Segmentation and Classification,” *2016 Int. Conf. Digit. Image Comput. Tech. Appl. DICTA 2016*, Dec. 2016, doi:

10.1109/DICTA.2016.7797031.
[41] G. Wiedemann and G. Heyer, “Page Stream Segmentation with Convolutional Neural Nets Combining Textual and Visual Features,” *Lr. 2018 - 11th Int. Conf. Lang. Resour. Eval.*, pp. 3675–3680, Oct. 2017, doi: 10.48550/arxiv.1710.03006.
[42] G. Wiedemann and G. Heyer, “Multi-modal page stream segmentation with convolutional neural networks,” *Lang. Resour. Eval. 2019 551*, vol. 55, no. 1, pp. 127–150, Sep. 2019, doi: 10.1007/S10579-019-09476-2.
[43] X. Yang, E. Yumer, P. Asente, M. Kraley, D. Kifer, and C. L. Giles, “Learning to extract semantic structure from documents using multimodal fully convolutional neural networks,” *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-January, pp. 4342–4351, Nov. 2017, doi: 10.1109/CVPR.2017.462.
[44] “(PDF) Multimodal Deep Neural Networks for Banking Document Classification.” https://www.researchgate.net/publication/336552928_Multimodal_Deep_Neural_Networks_for_Banking_Document_Classification (accessed Mar. 09, 2022).
[45] T. Dauphinee, N. Patel, and M. Rashidi, “Modular Multimodal Architecture for Document Classification,” Dec. 2019, doi: 10.48550/arxiv.1912.04376.
[46] A. Guha and D. Samanta, “Real-Time Application of Document Classification Based on Machine Learning,” pp. 366–379, Oct. 2019, doi: 10.1007/978-3-030-38501-9_37.
[47] “Use of language models for document stream segmentation - Inria.” https://hal.inria.fr/hal-02975046 (accessed Mar. 09, 2022).
[48] S. Pramanik, S. Mujumdar, and H. Patel, “Towards a Multi-modal, Multi-task Learning based Pre-training Framework for Document Representation Learning,” Sep. 2020, doi: 10.48550/arxiv.2009.14457.
[49] A. Guha, A. Alahmadi, D. Samanta, M. Z. Khan, and A. H. Alahmadi, “A Multi-Modal Approach to Digital Document Stream Segmentation for Title Insurance Domain,” *IEEE Access*, vol. 10, pp. 11341–11353, 2022, doi: 10.1109/ACCESS.2022.3144185.
[50] “Introduction to Information Retrieval.” https://nlp.stanford.edu/IR-book/information-retrieval-book.html (accessed Mar. 24, 2022).
[51] “User-Based Data | Elasticsearch: The Definitive Guide [2.x] | Elastic.” https://www.elastic.co/guide/en/elasticsearch/guide/current/user-based.html (accessed Mar. 24, 2022).
[52] “What are microservices?” https://microservices.io/ (accessed Mar. 24, 2022).
[53] “About us — scikit-learn 1.0.2 documentation.” https://scikit-learn.org/stable/about.html (accessed Mar. 26, 2022).
[54] P. W. Farris, N. T. Bendle, P. E. Pfeifer, and D. JReibstein, “MARKETING METRICS SECOND EDITION.”