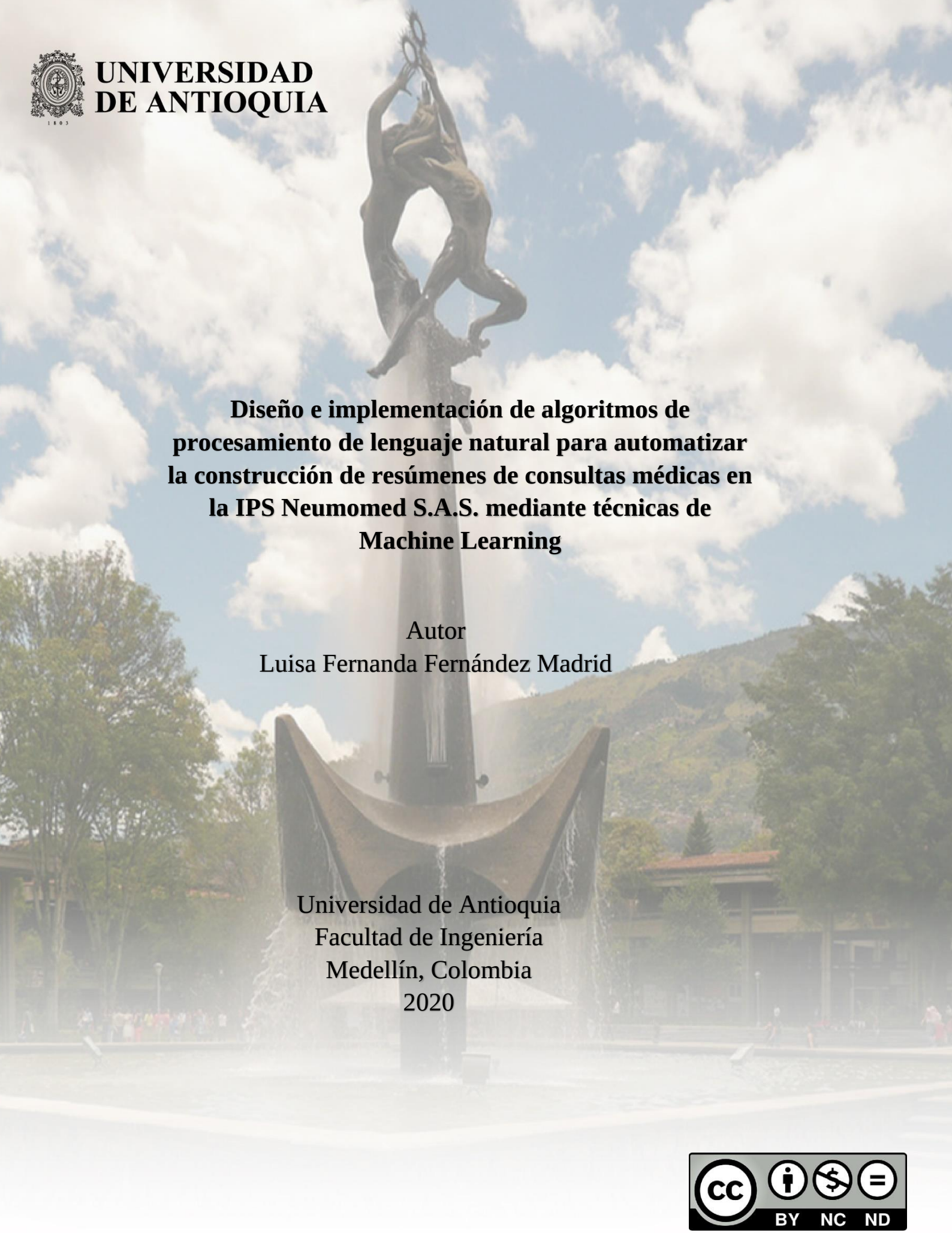

UNIVERSIDAD DE ANTIOQUIA

Diseño e implementación de algoritmos de procesamiento de lenguaje natural para automatizar la construcción de resúmenes de consultas médicas en la IPS Neumomed S.A.S. mediante técnicas de Machine Learning

Autor
Luisa Fernanda Fernández Madrid

Universidad de Antioquia
Facultad de Ingeniería
Medellín, Colombia
2020

Diseño e implementación de algoritmos de procesamiento de lenguaje natural para automatizar la construcción de resúmenes de consultas médicas en la IPS Neumomed S.A.S. mediante técnicas de Machine Learning

Luisa Fernanda Fernández Madrid

Tesis o trabajo de investigación presentada(o) como requisito parcial para optar al título de:
Bioingeniera

Asesores (a):
Jonathan Gallego Londoño, Bioingeniero, M.Sc Ingeniería Biomédica
María Camila Niebles Reyes, Bioingeniera

Línea de Investigación:
Informática médica

Universidad de Antioquia
Facultad de Ingeniería
Medellín, Colombia
2020.

Diseño e implementación de algoritmos de procesamiento de lenguaje natural para automatizar la construcción de resúmenes de consultas médicas en la IPS Neumomed S.A.S. mediante técnicas de Machine Learning

Resumen

Los textos resumen de una consulta, encuentro o seguimiento médico sirven para dar contexto de la historia clínica del paciente pues dan cuenta de datos importantes como: información general del paciente, antecedentes, medicamentos y otros hallazgos de la consulta. En este proyecto se presenta un algoritmo desarrollado por el departamento de T.I de la I.P.S Neumomed S.A.S para suplir la necesidad de generar en tiempo real el texto resumen de un encuentro médico. Para la creación de esta funcionalidad se identificaron los siguientes requerimientos: el algoritmo debe generar un texto resumen introductorio del paciente que contenga la información general de este y del último encuentro médico del paciente, debe recibir variables con información clínica del paciente y generar un texto resumen, debe ser escalable y fácilmente implementable en cualquier cliente de la I.P.S. Para cumplir con los requerimientos expresados anteriormente, se construyó un modelo supervisado de machine learning tipo árbol de decisiones, que cuenta una fase de entrenamiento y otra de implementación, para la generación automatizada de textos implementando técnicas de procesamiento de lenguaje natural. El modelo generado se validó mediante técnicas de regresión logística y análisis típicos de algoritmos de inteligencia artificial con base en la matriz de confusión y la medición de exactitud (accuracy). De igual manera se evalúo la calidad del texto generado por medio de revisiones manuales y automatizadas, para garantizar la legibilidad, orden, coherencia y buena ortografía del mismo, finalmente se realizaron validaciones de seguridad y desempeño del algoritmo.

Abstract

The electronic medical record serves to give context to the patient's medical history, since they account for important data such as: general patient information, history, medications and other findings of the consultation. This project presents an algorithm developed by the T.I department of Neumomed S.A.S to supply the need to generate medical record in real time. To create this functionality, the following requirements were identified: the algorithm must generate an introductory text about the patient, it must contain general information of this and the patient's last medical meeting, must receive variables with the patient's clinical information and generate a text summary, it must be scalable and easily deployable in any client. To accomplish the requirements expressed above, a supervised decision tree machine learning model was built, it includes a training phase and an implementation phase that implement natural language processing techniques. The generated model was validated using logistic regression techniques and was analysed based on the confusion matrix and the accuracy measurement. In the same way, the quality of the generated text was evaluated through manual and automated revisions, to guarantee its legibility, order, coherence, and good spelling. Finally, security and performance validations were carried.

Tabla de contenido

1.Introducción

Las consultas médicas son el servicio clínico más requerido en las instituciones prestadoras de servicios de salud, no solamente en Colombia, sino en el mundo [1]. En dichos encuentros entre el profesional de la salud y el paciente, se siguen una serie de pasos para garantizar la adecuada atención para los usuarios [2]. Uno de estos pasos, es el llenado de registros y formularios dentro de la historia clínica del paciente, la cual contiene información sobre el estado de salud, motivo de consulta, antecedentes, medicamentos, entre otros. Adicionalmente, se escribe en la historia clínica un texto resumen de la consulta, con la información más relevante sobre esta, con el fin de dejar una descripción de fácil consulta del historial médico del paciente.

En general, en las instituciones prestadoras de servicios de salud el resumen médico de las consultas se construye manualmente, siendo este también el caso de la IPS Neumomed. Esta que es una clínica de sueño, especializada en el tratamiento de pacientes con afecciones respiratorias, para esto se realizan diferentes encuentros con los pacientes, en los cuales la escritura del texto resumen de cada consulta se realiza de manera manual, es decir, en el momento de la consulta cada médico anexa en la historia clínica un resumen construido en el editor de texto de la aplicación web de historia clínica de la IPS, incluyendo en este los aspectos de la consulta que considera relevantes. Dicha actividad consume una cantidad significativa del tiempo de la consulta y en muchos casos presenta errores humanos o falta de información [3]. Tras la redacción de estos registros de historia clínica de los pacientes, se almacenan de manera electrónica, lo que haría posible extraerlos y analizarlos mediante algún algoritmo computacional. Actualmente por medio de las tecnologías de la información, se han implementado algunas herramientas para la redacción de la conversación entre pacientes y médicos; sin embargo, estas contienen toda la información intercambiada entre estos (como saludos, charlas y demás), lo que ocasiona que estos textos contengan mucha información que podría ser irrelevante. Es por esto, que en el presente proyecto se presenta el desarrollo de un algoritmo de procesamiento de lenguaje natural basado en técnicas de machine learning supervisado de tipo árbol de decisiones que tiene la capacidad de realizar las actividades de construcción de los resúmenes durante la consulta médica [4][5], quitando la carga al profesional de la salud de transcribir estos dentro de la historia clínica del paciente. Automatizar este proceso permite agilizar los diferentes encuentros médicos y mejora la atención brindada. La herramienta no reemplaza el

criterio médico ya que este tiene la posibilidad de editar, avalar y/o reportar el texto construido por el algoritmo.

2.Objetivos

2.1 Objetivo General

Diseñar e integrar algoritmos de procesamiento de lenguaje natural en la aplicación web de la IPS Neumomed S.A.S. para automatizar la construcción de resúmenes de consultas médicas mediante técnicas de Machine Learning.

2.2 Objetivos específicos

1. Diseñar un algoritmo de procesamiento de lenguaje natural basado en técnicas de machine learning.

2. Entrenar el algoritmo construido, haciendo uso de los resúmenes de consultas médicas existentes en la base de datos de la IPS Neumomed S.A.S.

3. Implementar el algoritmo entrenado para la generación automática de los resúmenes de consultas médicas.

4. Estructurar los servicios del algoritmo en un patrón de arquitectura cliente/servidor, para que este pueda ser consumido por los aplicativos móvil y web de Neumomed S.A.S

3.Marco Teórico

3.1 Encuentro médico

Un encuentro médico se define como la atención brindada a un paciente por un médico general, médico familiar o especialista. Esta incluye un conjunto de actividades diagnósticas mediante las cuales se evalúa el estado de salud del paciente y se documenta su sintomatología y antecedentes, con el objeto de detectar circunstancias o patologías que puedan alterar su desarrollo y salud [1].

En dichos encuentros entre el paciente y el profesional de la salud se atraviesan diferentes etapas, como se puede evidenciar en la Figura 1. Para esto se examinan los aspectos listados a continuación:

• Enfermedad actual: síntomas, factores de riesgo

•Antecedentes: medicamentos, hospitalizaciones, factores ambientales, patologías familiares, alergias

• Tratamiento actual: medicamentos

• Síntomas actuales: revisión por sistemas

• Exámenes paraclínicos: revisión de exámenes

• Examen físico consulta: examen físico y revisión de signos vitales

• Diagnóstico: clasificación de la patología, farmacología

• Plan de consulta: inscripción a planes y programas

• Elaboración de texto resumen de la consulta (este se elabora durante el transcurso de la consulta e incluye toda la información anteriormente mencionada).

La información arrojada por cada uno de los exámenes y procedimientos debe ser registrada en la historia clínica del paciente, y gran parte de esta información debe ser incluida también en el texto resumen de la consulta.

3.2 Resumen de consulta médica

La elaboración de los resúmenes de las consultas es una actividad muy importante, pues deja en la historia clínica un registro de fácil lectura de todos los encuentros médicos que el paciente haya tenido, y es un recurso que podría ser útil en posteriores encuentros médicos o en consultas con un profesional de la salud distinto al habitual [2]. Sin embargo, considerando que el tiempo mínimo requerido por ley para cada consulta es de solo 20 minutos; en ocasiones los resúmenes de las consultas no se realizan de manera adecuada, o la redacción de estos genera retraso e incumplimientos en la agenda [3].

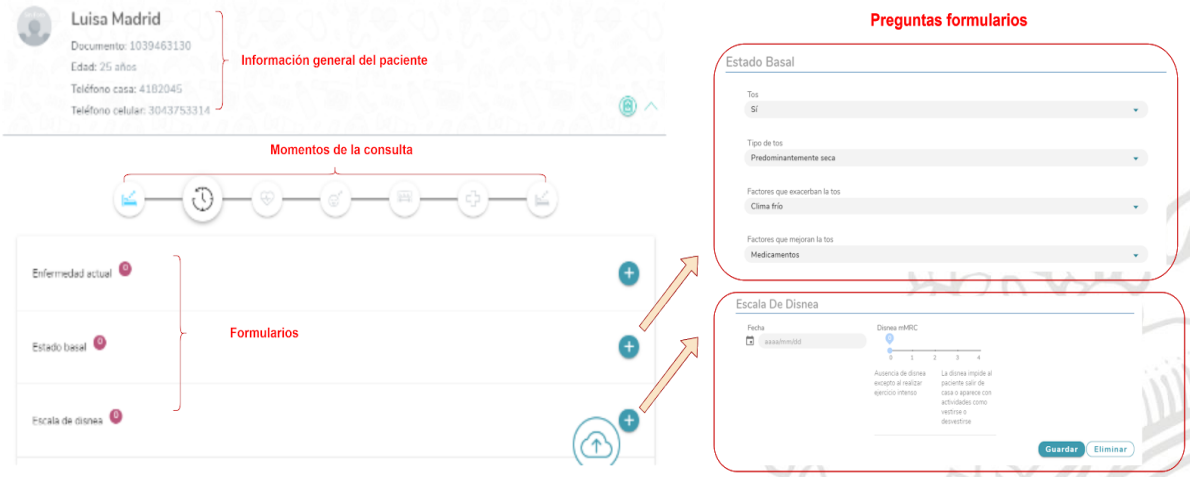

Figura 1: Flujo de la consulta médica

3.3 Procesamiento de lenguaje natural

En la actualidad, uno de los mayores desafíos de las diferentes instituciones es automatizar procesos como los anteriormente mencionados. Siendo los algoritmos basados en inteligencia artificial, una de las herramientas de tipo software más implementadas en los últimos años, una de estas herramientas es el procesamiento de lenguaje natural NLP (natural language processing), que se puede definir como la capacidad que tiene una computadora para comprender el lenguaje humano tal como se habla. De igual modo, los algoritmos de NLP pueden entrenarse para que aprendan a reconocer y reconstruir textos que tengan una estructura definida, siendo este el caso de los resúmenes de las consultas médicas. Es decir, se puede entrenar un algoritmo por medio de técnicas de machine learning para que reciba los datos recolectados durante una consulta médica y posteriormente genere de manera automática el resumen de esta [4].

3.4 Machine learning

El machine learning (ML) se basa en la solución de problemas mediante herramientas computacionales donde no se programa de manera explícita una máquina para que solucione el problema, sino que se llega a un modelo computacional mediante un entrenamiento o aprendizaje del mismo. Dichos algoritmos una vez entrenados, pueden ser cargados en máquinas virtuales, a las cuales se puede acceder de manera remota, de igual manera se pueden trabajar de manera local integrándose a procesos más grandes [5], haciendo posible la integración de estos a aplicaciones de tipo web, lo que permite una globalización de la información y mayor accesibilidad a esta [6].

3.5 Aprendizaje supervisado

En este tipo de aprendizaje las muestras de entrenamiento incluyen los atributos que queremos predecir y se busca asignar dicho atributo de un dato nuevo del cual no se conoce el atributo objetivo, a su vez estos algoritmos pueden dividirse en dos:

- Clasificación: las muestras pertenecen a 2 o más clases y queremos aprender a predecir con diferentes datos a qué clase pertenecen nuevas variables que el modelo no haya visto.

● Regresión: donde se quiere aprender a predecir el valor de un atributo de los datos, a partir de sus demás atributos.

Estos modelos cuando son entrenados, de acuerdo a las entradas que tengan, darán como respuesta el parámetro que se quería determinar. Dichos algoritmos una vez entrenados pueden ser cargados en máquinas virtuales a las cuales se puede acceder de manera remota. De igual manera se pueden trabajar de manera local integrándose a procesos más grandes [7].

En el aprendizaje automático, los modelos o algoritmos usados tienen hiperparametros o parámetros que deben ser ajustados para mejorar el desempeño y maximizar el aprendizaje del modelo [8], estos parámetros son especificados antes de que inicie el entrenamiento [9], pueden ser ajustados mediante diversas técnicas, algunas basadas en métodos estadísticos y algunos en ensayo y error. En los diferentes lenguajes de programación que permiten la creación de modelos de ML existen librerías que tienen herramientas propias para ayudar a ajustar estos parámetros.

Algunos modelos de entrenamiento supervisado utilizados en este estudio se describen a continuación.

3.6 Árbol de decisiones

Son modelos de reglas ampliamente usados en la inteligencia artificial o para la toma de decisiones en algoritmos más sencillos. Estos se componen por 3 elementos como se muestra en la Figura 2. Un nodo es un punto donde se toma una decisión o se evalúa una característica, una rama es la salida de un nodo y lleva esto a otro nodo o a una hoja, una hoja es el resultado final y contiene una etiqueta de acuerdo a los pasos ejecutados.

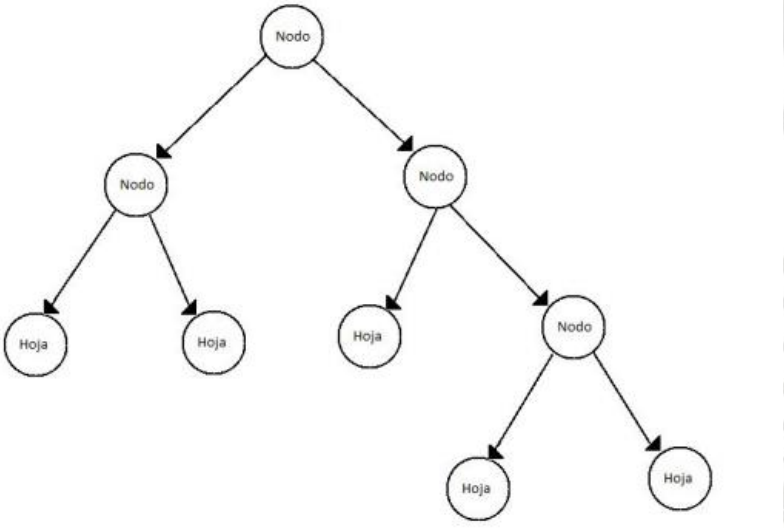

Figura 2: Ejemplo árbol de decisiones

Este tipo de algoritmos tiene hiperparametros como:

- Profundidad del árbol: que es la cantidad de nodos hijos que tiene el nodo raíz.

- Mínimo número de muestras para particiones: que da idea de la cantidad de mínima de muestras que cumplen la condición de nodo para que este se parta en 2 nodos hijos.
- Criterio de selección: este es un criterio matemático para calcular el error en el entrenamiento del modelo y su aprendizaje [10][11].

3.7 Regresión logística

El modelo de regresión logística es un modelo de aprendizaje automático basado en regresiones que se ajusta de manera probabilística, donde se determina un umbral de probabilidad de que un sujeto pertenezca o a una clase a medida que las demás variables cambian. En estos algoritmos se trata de predecir el valor de la variable dependiente de acuerdo con las variaciones que tienen las demás variables. Uno de los parámetros más importantes que tiene este tipo de modelos es la regularización, esto da idea del nivel de generalización que tiene el modelo, ya que estos son susceptibles al sobreajuste o memorización de los datos. Este parámetro se encarga de mitigar este sobreajuste [12][13].

3.8 Evaluación de modelos supervisados

Los modelos supervisados de clasificación una vez entrenados se pueden evaluar con distintas métricas, pero entre las más usadas está la precisión, la cobertura o recall y los F-score. Estas medidas se basan en la cantidad de verdaderos positivos (TP) sujetos etiquetados de manera correcta, verdaderos negativos (TN) sujetos no asignados a una clase de manera verdadera, falsos positivos (FP) sujetos etiquetados en un grupo incorrecto y falsos negativos (FN) sujetos no asignados a la clase que pertenecen. Conociendo como se etiquetaron los sujetos se puede calcular la precisión(p) y recall(r) usando las ecuaciones 1 y 2 [16]:

$$p = \frac{TP}{TP + FP}$$

Ecuación 1: Cálculo de precisión (p)

$$r = \frac{TP}{TP + FN}$$

Ecuación 2: Cálculo de recall (r)

Las ecuaciones 1 y 2 dan información relevante; pero no muestra un panorama completo de la calidad del modelo. Por esto existen las F-score, que juntan en una sola medida la precisión y el recall. se calculan por medio de la ecuación 3 [17].

$$F_\beta = (1 + \beta^2)\frac{p\,r}{r + \beta^2\,p} = \frac{(1 + \beta^2)TP}{(1 + \beta^2)TP + \beta^2 FN + FP}$$

Ecuación 3: cálculo del valor F-score

Otra forma de evaluar el desempeño de un algoritmo o modelo de clasificación es la matriz de confusión, especialmente significativa en clasificaciones multiclase. Esta matriz relaciona los sujetos de una clase con las etiquetas asignadas a dichos sujetos

y muestra los sujetos etiquetados de manera correcta situados en la diagonal principal de la matriz y qué etiquetas asignó a los sujetos que fueron mal clasificados como se muestra en la figura 3 [18].

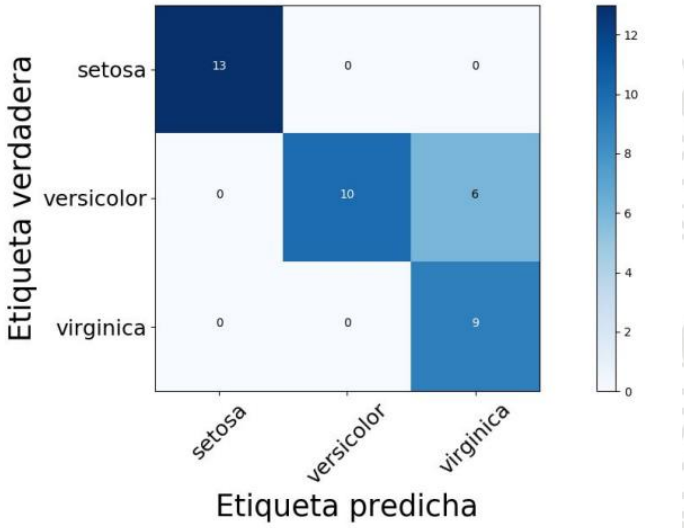

Figura 3. Ejemplo de matriz de confusión para clasificación de tipos de flores.

3.9 Lenguaje y librerías

Para generar estos modelos se utilizan librerías en Python, entre los más conocidos se encuentran:

3.9.1 Scikit learn

Es una librería open source desarrollada en 2007 por David Cournapeau enfocada en rutinas de machine learning, simplificadas y de fácil levantamiento, además de poseer módulos de procesamiento de datos y análisis de modelos entrenados. Para este estudio se usa la versión 0.19.1 [14]

3.9.2 Tensor Flow

Tensor Flow es una librería open source desarrollada por Google y lanzada en 2015, enfocada en dar soluciones de machine learning de alto y bajo nivel, especializada en redes neuronales y procesamiento tensorial, cuenta con módulos de análisis de resultados y monitoreo del entrenamiento. En este proyecto se usa la versión 1.12.0 [15]

3.10 Arquitectura de software

Los modelos construidos a partir de las diferentes librerías se estructuran empleando un patrón de arquitectura de software. En dichos patrones se expresa una descripción de los subsistemas y componentes de un sistema de software que establece las relaciones entre ellos. Dichos subsistemas y componentes generalmente se especifican en diferentes vistas para mostrar las propiedades funcionales y no funcionales relevantes de un sistema de software, la definición de un patrón de arquitectura es el resultado principal de la actividad de diseño de software [19].

3.10.1 Arquitectura cliente-servidor

Este tipo de patrón de arquitectura se utiliza cuando el servidor y el cliente se conectan a una red. Aquí el servidor actúa como proveedor de servicios y el cliente es el consumidor del mismo. Normalmente el servidor está ubicado en internet, sin embargo, en los casos de manejo de información sensible el servidor se puede ubicar en una red de área local, garantizando que los usuarios externos no puedan acceder al servidor, pero los usuarios internos sí [20].

3.11 Industrialización

Un modelo de machine learning que pasa la etapa de entrenamiento y su rendimiento cumple los requerimientos para solucionar el problema se puede llevar a la denominada etapa de producción o industrialización. En el caso de modelos de clasificación, estos predicen la etiqueta de un sujeto que nunca ha visto a partir de sus características, este puede levantarse en un servidor y funcionar como una aplicación (API) a la cual se consulta la predicción. Específicamente para python hay diferentes opciones para levantar un servidor, como Django o Flask, en este proyecto se usará Flask un framework enfocado al desarrollo web y especialmente al levantamiento de servidores de manera rápida, donde en un mismo servidor pueden vivir diferentes modelos y funciones con el fin de acceder a ellos de manera rápida y remota [21].

Metodología

El conjunto de actividades ejecutadas para el desarrollo del proyecto se lista a continuación, estas se evidencian también en el diagrama de flujo de la Figura 7.

- Revisión bibliográfica y del estado del arte: se utilizaron diferentes motores de búsqueda para hacer revisiones en bases de datos y repositorios especializados en ingeniería de software, bioingeniería e inteligencia artificial. Empleando términos claves en la fórmula de búsqueda como: natural language processing, neural networks, artificial intelligent, machine learning, Python, bioengineering, clinical notes, entre otras.
- Levantamiento de requerimientos técnicos: se definieron los requerimientos de operación y funcionamiento que debía tener el algoritmo para que este pudiera ser implementable en la historia clínica digital de la IPS y se identificaron las problemáticas presentadas con soluciones que se intentaron implementar en el pasado. Los requerimientos técnicos que se definieron para el algoritmo se presentan en la Tabla 1.

Tabla 1: Requerimientos técnicos

R 001 - Obtención de información relevante y específica del paciente
Se refiere a la acción de obtener información clínicamente relevante del paciente
R 002 - Generación automatizada de textos resumen
Esta función deberá generar textos resúmenes por medio de la aplicación de técnicas de machine learning
R 003 - Creación de un servicio con potencial de escalabilidad e interoperabilidad entre otros servicios

El servicio generado debe poder interactuar con los otros servicios existentes en el asistente virtual de Neumomed.
R 004 - Creación de un servicio que sea consumible desde diferentes clientes
El servicio debe poderse visualizar en cualquier equipo mientras este tenga un navegador y acceso a internet.
R 005 - Calidad
La funcionalidad del sistema debe ser de buena calidad para generar confiabilidad a los usuarios.
R 006 - Rendimiento
El servicio debe tener una adecuada velocidad de respuesta y el menor consumo de recursos posible
R 007 - Mantenimiento
El sistema estará en constante mantenimiento, verificación y actualización.
R 008 - Restricciones
-El servicio solo permite que solo los encargados de las áreas puedan ingresar -Una vez terminada la consulta los textos no pueden alterarse

- Levantamiento de requerimientos clínicos: en conjunto con el personal asistencial de la IPS se definieron las necesidad que el algoritmo debía suplir, además se definieron los elementos fundamentales de un resumen de consulta médica y su estructura. Los requerimientos clínicos definidos son los factores que determinan la estructura gramatical del texto, pues es necesario que el texto generado contenga toda la información médicamente relevante del paciente.

  A continuación, se presenta la estructura definida para el texto:

  Título del evento médico

  -Información general del paciente: este texto debe contener información de identificación del paciente, algunos de los elementos presentes en este son:

  - Nombre del paciente
  - Edad
  - Género
  - Estado Civil
  - Raza
  - Profesión u Ocupación
  - Fecha de Ingreso

- Motivo de consulta y enfermedad actual: se indica la razón por la cual el paciente asiste a la consulta, los síntomas y padecimientos manifestados por el paciente y la enfermedad base diagnosticada (EPOC, SAOS, asma, entre otras).

-Antecedentes: se refiere a todos los antecedentes patológicos del paciente, estos pueden ser de tipo:

- Familiares
- Personales
- Fisiológicos
- Patológicos
- Maritales
- Quirúrgicos
- Psicológicos
- Psiquiátricos
- Entre otros

-Examen físico general: se refiere al estado físico del paciente en el momento de la consulta, las variables incluidas en este son:

- Signos vitales
- Peso
- Talla
- Medidas antropométricas

-Hallazgos en la consulta: se indican los hallazgos médicos tanto positivos como negativos pertinentes que permitan describir el estado de salud actual del paciente.

-Concepto: opinión profesional emitida por el asistencial tras considerar los hallazgos en la consulta.

-Plan: conjunto de instrucciones que se deben ejecutar en el tratamiento del paciente tras la consulta.

-Observaciones: se incluyen las apreciaciones médicas particulares de cada caso

- Selección del tipo de algoritmo de machine learning implementado: se seleccionó el tipo de algoritmo teniendo como criterios de inclusión tener suficiente documentación, haber sido implementada anteriormente en el desarrollo de software para aplicaciones en la bioingeniería, ser código abierto, soportar alimentación de datos en español, no tener vulnerabilidades de seguridad evidentes, implementable en Python.
- Diseño del algoritmo de machine learning: una vez seleccionado el tipo de algoritmo, se construyó un algoritmo de generación de texto que cumpliera con los requerimientos levantados desde el aspecto técnico.

  Con base en revisiones bibliográficas se determinó que el algoritmo de machine learning sería de tipo árbol de decisiones, considerando que tienen alta precisión, bajo requerimiento computacional y menores tiempos de ejecución.

El algoritmo se construyó haciendo uso de la librería scikit learn de python, como parámetros base se escogieron los que la librería recomienda para problemas de clasificación multiclase.

El árbol de decisiones obtenido tras la implementación de la librería se ilustra en la Figura 4. En la lectura de este es posible notar que el número mínimo de decisiones que el algoritmo toma para generar el texto son 3 y el máximo son 6.

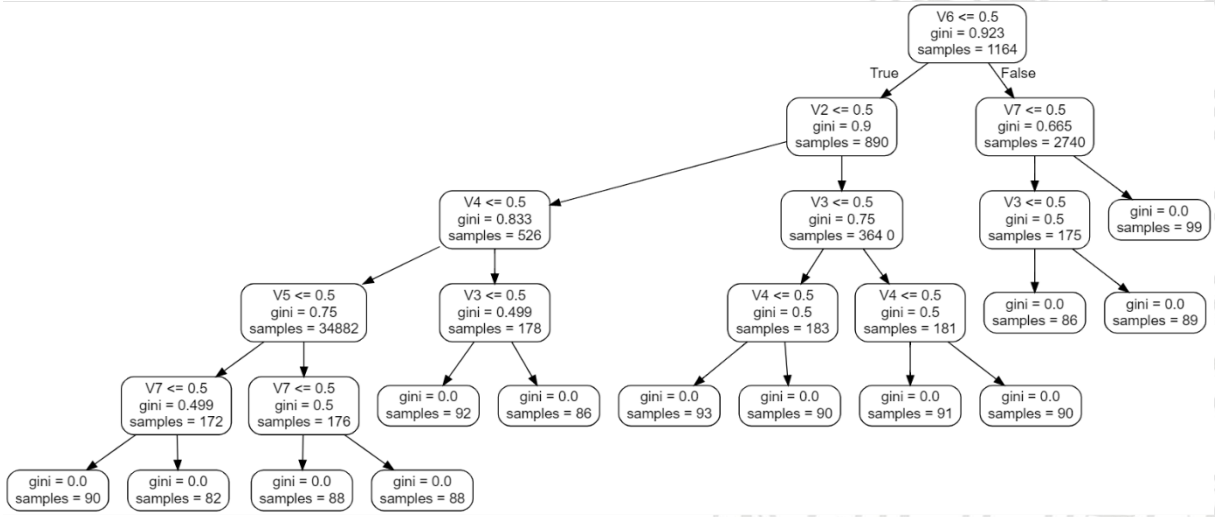

Figura 4: representación gráfica del árbol de decisiones construido

- Selección de los parámetros de entrenamiento: con base en la literatura se definieron los parámetros de entrenamiento del modelo.

  Para la selección de los parámetros relacionados con la profundidad del árbol y el número de divisiones, se realizó un ajuste paramétrico por medio de regresiones logísticas, para esto se evaluó cómo varía la precisión del modelo a medida que cambian dichos parámetros, como se evidencia en las Figuras 5 y 6 . Finalmente se eligieron los valores en los que la precisión fuera del 100%.

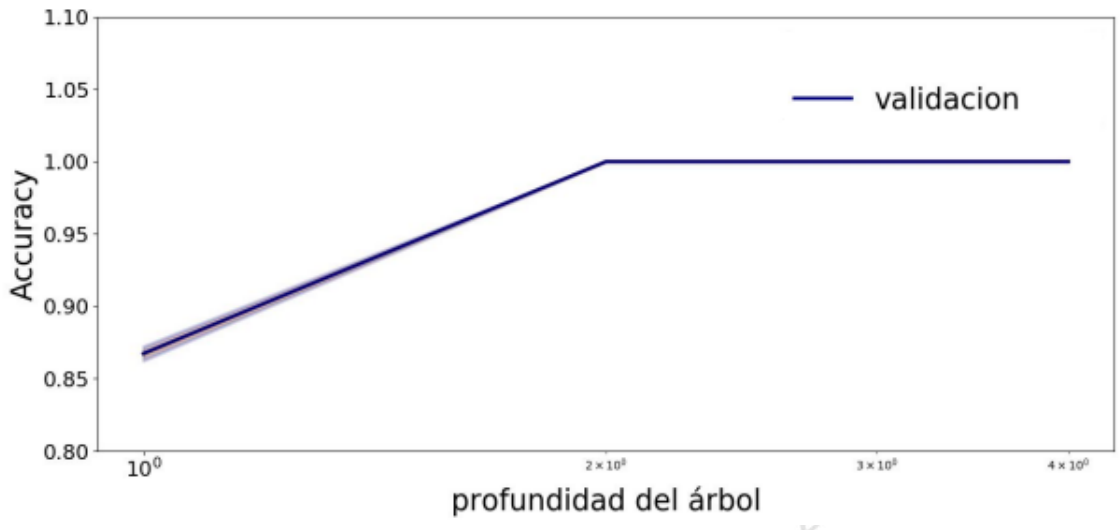

Figura 5. Cambio de la precisión respecto a la profundidad del árbol

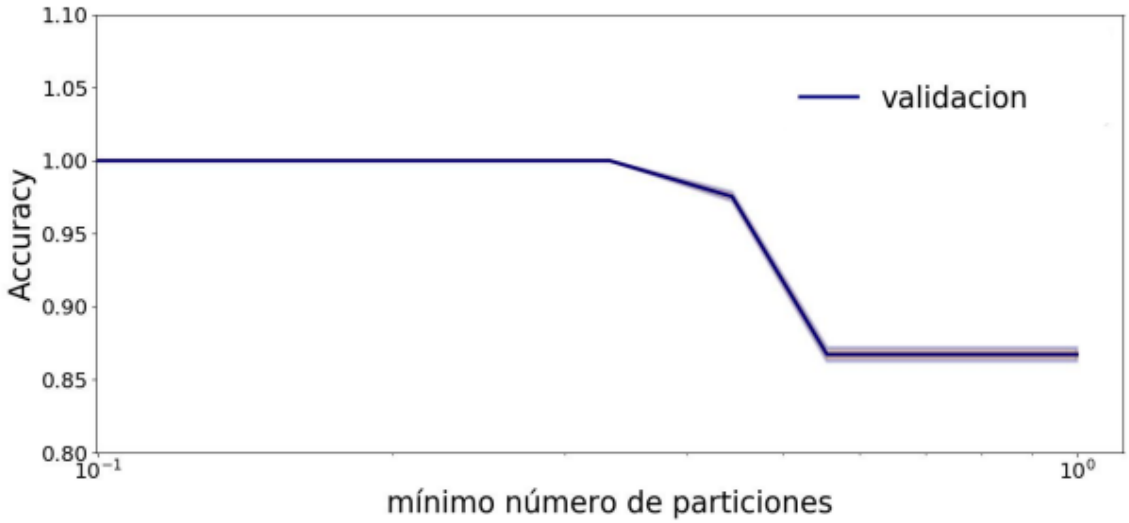

Figura 6. Cambio de la precisión respecto al mínimo de particiones

- Entrenamiento del modelo de machine learning: se crearon las matrices de entrenamiento del modelo por medio de un ciclo de condicionales y se determinó que estas fueran de tipo binario.
- Evaluación de desempeño del algoritmo de machine learning: empleando librerías de python se evalúa la precisión del modelo entrenado.
- Implementación del modelo de machine learning en algoritmo de procesamiento de lenguaje natural: se construye el algoritmo de implementación del modelo y se prueba inicialmente con parámetros simulados haciendo uso de diversas librerías de Python. Algunas de estas librerías son: Natural Language Toolkit, Gensim, polyglot, TextBlob, CoreNLP, Pattern, Vocabulary, Tensor Flow Python, Keras Python, Scikit-learn Python, PyTorch, NumPy, Pandas.
- Extracción de los textos resumen de la base de datos general de la IPS: se hicieron consultas en Firebase a la base de datos de la IPS, filtrando las consultas médicas de pacientes que cumplan con las siguientes condiciones: consulta de ingreso con médico general, formularios de consulta completos, pacientes ingresados entre 2018-2020 y que tengan en la historia clínica resumen de la consulta.
- Construcción de plantillas de entrenamiento: a partir de los textos extraídos de la base datos, se construyeron las plantillas de entrenamiento del modelo buscando que se conservara la estructura que se indicó desde el requerimiento clínico.
- Construcción de la base de datos: la base de datos se construyó en Firestore, en esta se almacenaron las plantillas de entrenamiento anteriormente diseñadas.
- Construcción automática de resúmenes de las consultas implementado el algoritmo: se implementó el algoritmo entrenado usando postman para simular la interacción que tendrían los clientes (web o móviles) con el algoritmo.
- Evaluación manual de la calidad de los resúmenes generados: se revisaron uno por uno 24 de los textos construidos por el algoritmo con el fin de identificar patrones de fallos, faltas de ortografía o carencias de redacción en el texto. La evaluación del texto se realizó por un equipo conformado por: 2 miembros del departamento de ingeniería, uno de medicina y uno de comunicaciones.

- Preguntas criterio de evaluación médica:

  -¿El texto es coherente?
  -¿El texto es técnicamente válido?
  -¿Se incluyen todas las variables relevantes en el texto?
  -¿El texto cumple con la estructura que se definió desde los requerimientos clínicos?

- Preguntas criterio de evaluación de ingeniería:

  -¿El texto tiene fallos de construcción que estén inminente relacionados con el modelo?
  -¿El texto presenta variables nulas o vacías?
  -¿El texto cumple con la estructura que se definió desde los requerimientos clínicos?

- Preguntas criterio de evaluación de comunicaciones:

  -¿El texto es de fácil lectura?
  -¿El texto es entendible?
  -¿El texto tiene buena ortografía?

- Re-entrenamiento correctivo del sistema: se realizó nuevamente el entrenamiento supervisado del sistema basado en los resultados obtenidos de los criterios de evaluación de los textos. Para esto se incluyeron en el algoritmo librerías y diccionarios de semántica española con el objetivo de mejorar la ortografía.
- Implementación de arquitectura cliente-servidor: para garantizar el consumo libre del proyecto desde cualquiera de los clientes (movil, web, server) de la IPS, se implementó la arquitectura cliente-servidor en el algoritmo.
- Implementación de seguridad en el proyecto: se implementaron condiciones de autenticación y cifrado de la información en el algoritmo para garantizar la confidencialidad en el tratamiento de información sensible.
- Consumo del algoritmo desde el frontend: se diseñó una interfaz gráfica en la historia clínica digital de la IPS para consumir el algoritmo de machine learning.
- Análisis estadístico de los resultados: se realiza un análisis estadísticos de los tiempos de ejecución del algoritmo en comparación con las soluciones que existían anteriormente en la IPS para la generación de textos. Adicionalmente se evalúa estadísticamente la asertividad gramatical de los textos generados por el algoritmo de machine learning diseñado en el presente proyecto. Esta revisión se realizó por medio de la implementación de la librería grammar-check de python, la cual se encarga de analizar la gramática y sintaxis de un texto y encontrar los errores cometidos.
- Divulgación de resultados

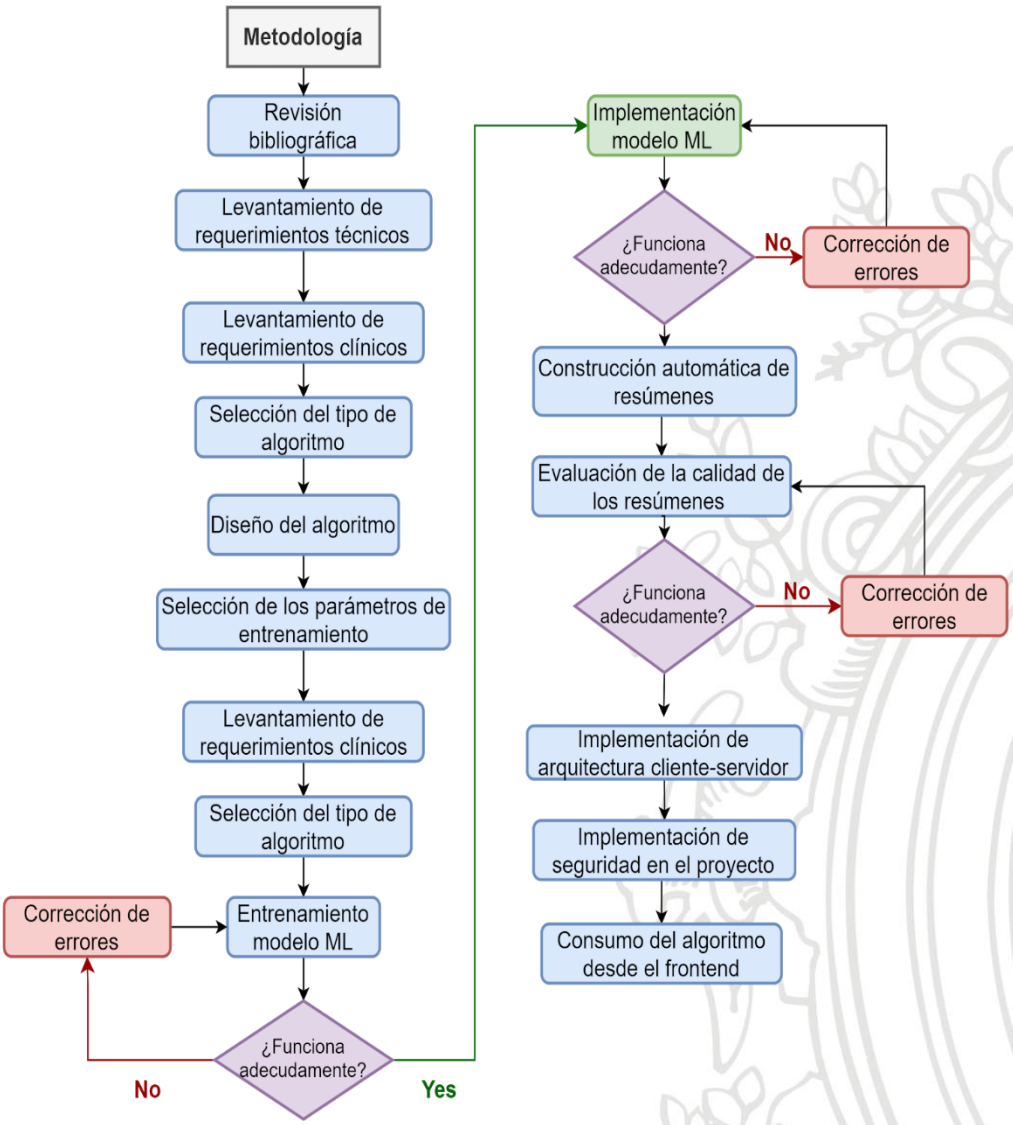

Figura 7: Diagrama de flujo de la metodología

5.Resultados y análisis

5.1 Entrenamiento del modelo de machine learning

En la Figura 8 se ilustra el diagrama UML de ejecución del algoritmo de entrenamiento del modelo, en este diagrama se presenta:
-El flujo de consumo de funciones en el algoritmo
-Las librerías empleadas
-Bases de datos implementadas
-Funciones principales del algoritmo, indicando parámetros de entrada y salida
-Funciones intermedias de consumo de otros servicios y nombre del servicio consumido

En la Tabla 2 se presenta un diccionario con las funciones representadas en el diagrama.

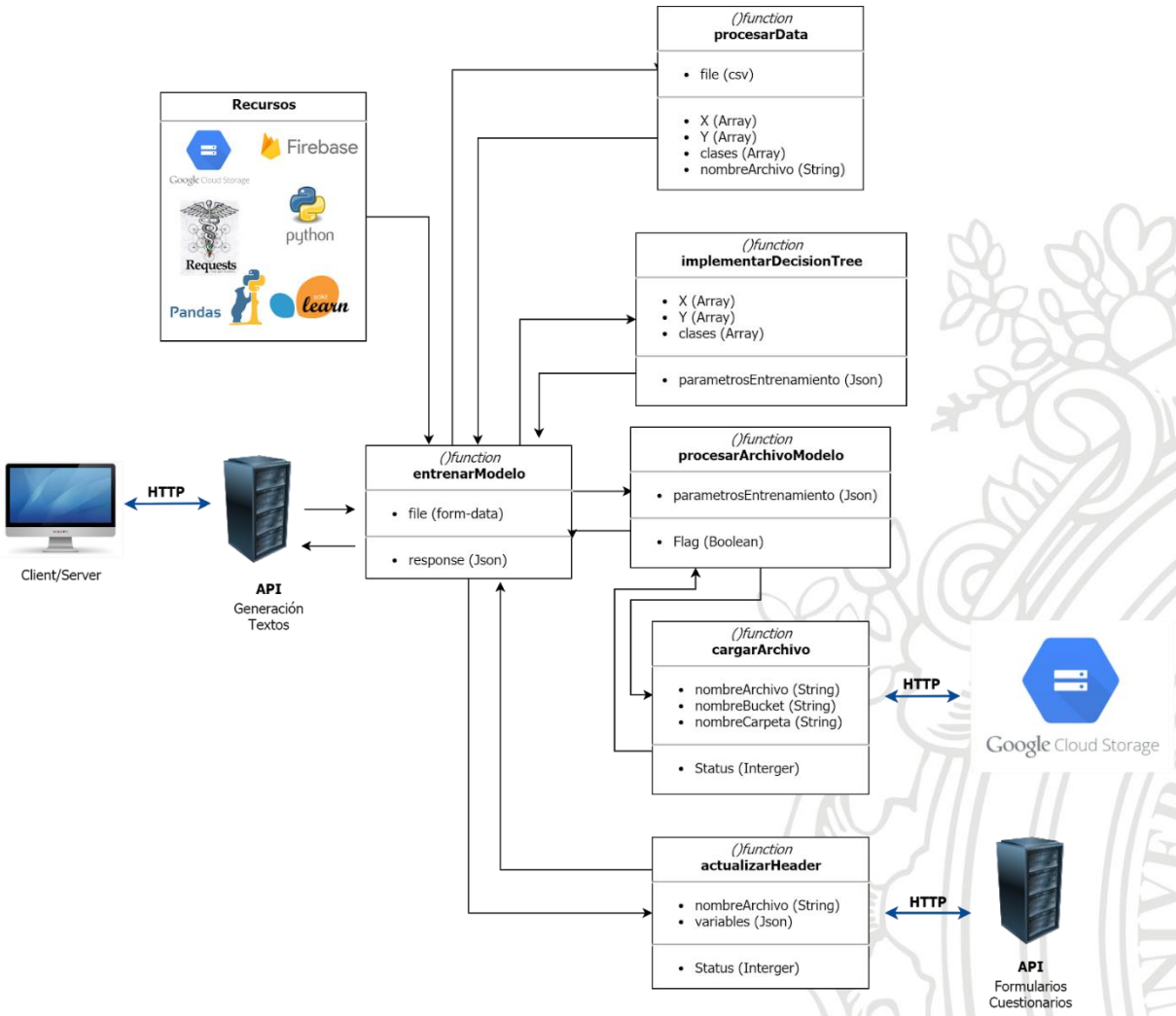

Figura 8: Diagrama UML entrenamiento del modelo

Tabla 2: diccionario de funciones de entrenamiento

Nombre	**Descripción**	**Salida**
entrenarModelo	Es la función principal del algoritmo, recibe de parte del cliente una petición http en la cual se ingresa el archivo que contiene la matriz de entrenamiento del modelo.	-Status -Message -Data: Ruta de almacenamiento del modelo
procesarData	Por medio de la librería pandas hace una lectura del archivo que contiene la matriz, limpia los datos que se hayan corrompido en el proceso de la petición y los ordena para su procesamiento	-Matriz de entrenamiento
implementarDecisionTree	Implementa el árbol de decisiones diseñado con los parámetros definidos, tomando como entrada la matriz de entrenamiento y genera el modelo de	-Modelo de machine learning

	machine learning por medio de la librería sklearn,	
procesarArchivosModelo	Comprime el modelo generado utilizando la librería pkl de python.	-Modelo comprimido en formato .pkl
cargarArchivo	Consume una API de Google Cloud Storage para almacenar el modelo comprimido y retorna a la función principal del algoritmo la ruta de almacenamiento	-Ruta de almacenamiento del modelo
actulizarHeader	Genera un registro en la base de datos de neumomed para generar un historial de archivos.	-Status -Message

5.2 Evaluación de desempeño del algoritmo de machine learning

Una vez entrenado el modelo y usando diferentes sets de validación cuyos valores de salida eran conocidos, se construyó la matriz de confusión del modelo para determinar la exactitud de este y determinar si era necesario hacer modificaciones paramétricas o estructurales en el código.

Como se aprecia en la Figura 9, el modelo generado tiene una precisión adecuada, pues clasifica todos los datos en la categoría correspondiente, sin importar si el tamaño de la muestra es grande (caso B) o pequeño (caso C).

Esto significa que el modelo está correctamente parametrizado y no es necesario reajustarlo ni re-entrenarlo.

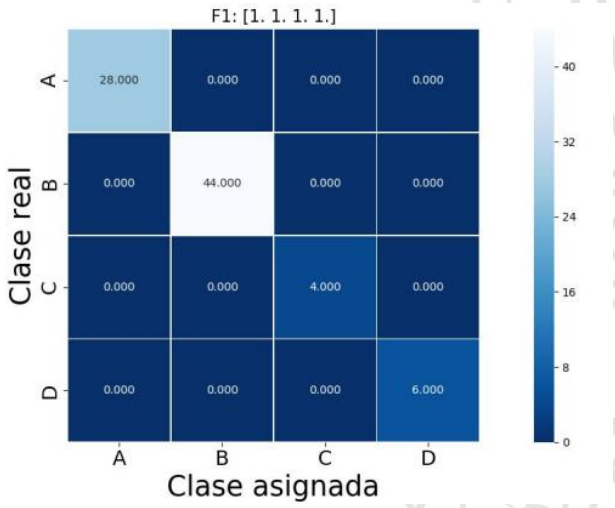

Figura 9: Matriz de confusión del modelo

5.3 Implementación del modelo de machine learning en algoritmo de procesamiento de lenguaje natural

En la Figura 10 se ilustra el diagrama UML de ejecución del algoritmo de implementación del modelo, en este diagrama se presenta:

-El flujo de consumo de funciones en el algoritmo
-Las librerías empleadas
-Bases de datos implementadas
-Funciones principales del algoritmo, indicando parámetros de entrada y salida
-Funciones intermedias de consumo de otros servicios y nombre del servicio consumido

En la Tabla 3 se presenta un diccionario con las funciones representadas en el diagrama.

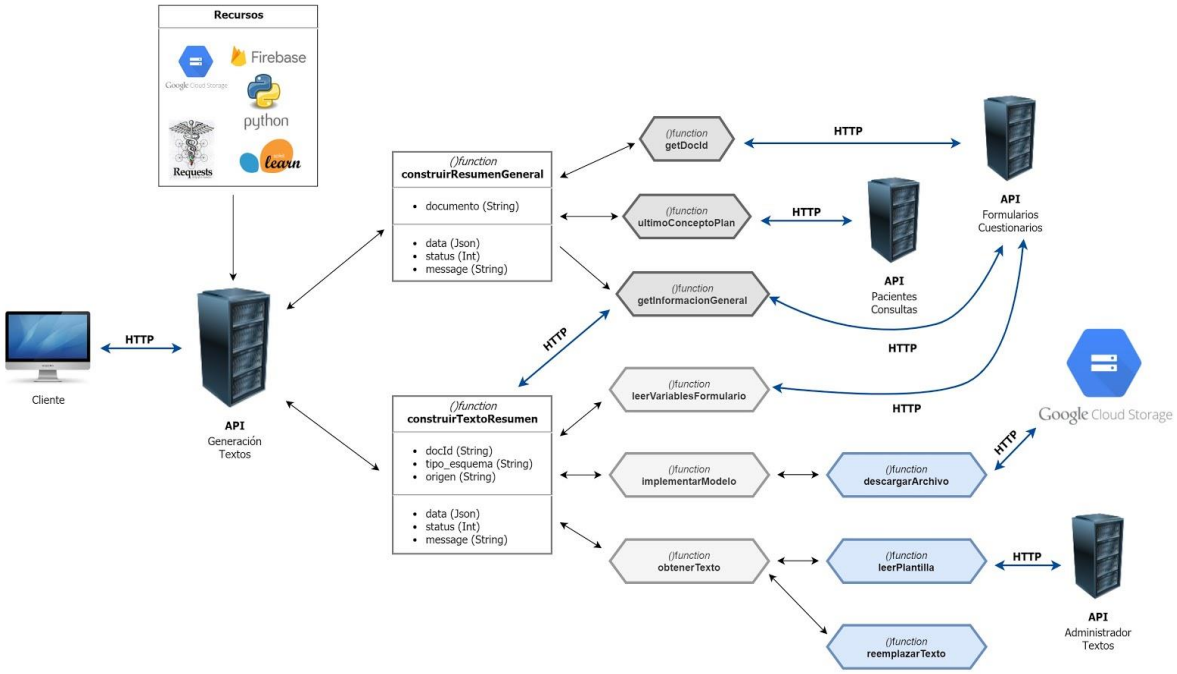

Figura 10: Diagrama UML implementación del modelo

Tabla 3: diccionario de funciones de implementación

Nombre	Descripción	Salida
construirResumenGeneral	Esta función se encarga de generar un texto resumen introductorio del paciente, que incluye información general y el texto resumen de la última consulta del paciente en la I.P.S	texto resumen general y última consulta
getDocId	Función para obtener el identificador del texto resumen de la última consulta	docId
ultimoConceptoPlan	Consulta el texto resumen de la última consulta del paciente en la I.P.S	Texto resumen última consulta

getInformacionGeneral	Genera el texto resumen con información general	Texto resumen información general
construirTextoResumen	Se encarga de generar el texto resumen de la consulta actual	Texto resumen consulta actual
leerVariablesFormulario	Identifica las variables que se incluirán en el texto resumen	Vector de variables
implementarModelo	Implementa el modelo que se generó en el entrenamiento	-Status -Message -Data
descargarArchivo	Obtiene el archivo con el modelo desde el google cloud storage, para poderse implementar.	-Modelo en formato pkl
obtenerTexto	A partir de la implementación del modelo se hace el procesamiento del lenguaje natural para obtener el texto resumen	Texto resumen consulta actual

5.4 Construcción automática de resúmenes de consultas implementado el algoritmo

Tras ejecutar el consumo del algoritmo utilizando postman como cliente se obtuvo como salida la respuesta ilustrada en la Figura 11.

La estructura de la respuesta se presenta en formato: status, message, data. Haciendo una lectura de este, podemos evidenciar que el estado de la respuesta fue:

-Status: 200, lo que para las peticiones http indica 'ok', es decir que el algoritmo respondió correctamente a la petición.

-message: Este es el texto resumen de la consulta, indica que la información obtenida proviene de una consulta médica.

-data: texto, en el data se contiene la información objetiva del cliente, es decir sería lo que se visualice en pantalla.

Figura 11: Consumo de la aplicación desde postman

5.5 Evaluación manual de la calidad de los resúmenes generados

Los resultados generales de los criterios de evaluación médica, de ingeniería y comunicaciones fueron:

- Criterio de evaluación médica: los textos introducen de manera adecuada al paciente, incluyen sintomatología, antecedentes, medicamentos, patologías y diagnóstico base. No se evidencian fallas inminentes en su construcción
- Criterio de evaluación de ingeniería: los textos generados coinciden con lo esperado según las variables de los casos clínicos de los pacientes de prueba.
- Criterio de evaluación de comunicaciones: los textos presentan una estructura coherente, sin embargo existen fallos de ortografía específicamente en las puntuaciones de las frases y acentuaciones de las palabras.

En la Figura 12 se presenta uno de los 24 textos evaluados

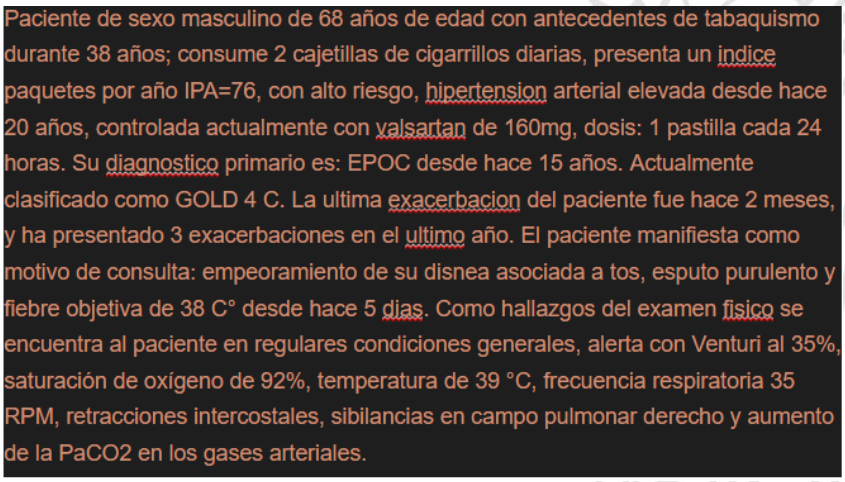

Figura 12: Ejemplo de texto generado

5.6 Re-entrenamiento correctivo del sistema

En la Figura 13 se evidencia que con las correcciones hechas en el entrenamiento del modelo se lograron corregir las faltas ortográficas. Se presentan inconvenientes con los nombres asociados a patologías, medicamentos y nombres propios en general. Sin embargo es una situación común con los generados de texto al ser palabras que generalmente no están incluidas en los diccionarios.

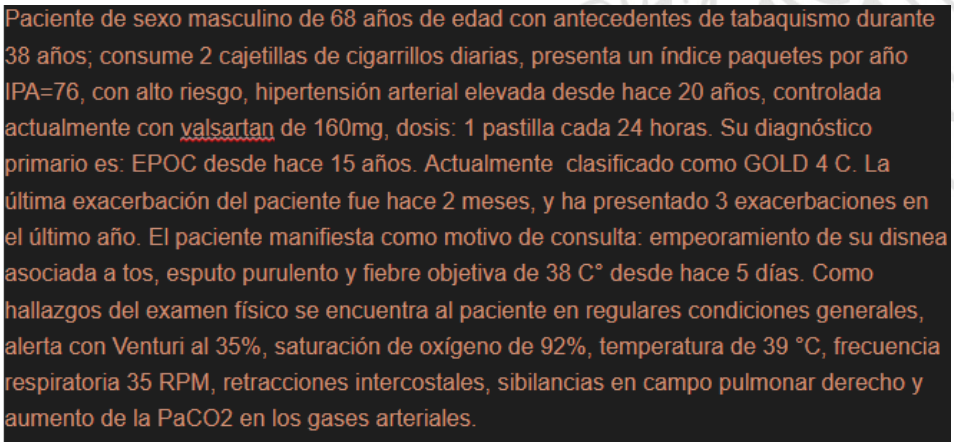

Figura 13: Ejemplo de texto corregido

5.7 Implementación de arquitectura cliente- servidor

En la Figura 14 se presenta el diagram UML de consumo del algoritmo, este se creó como un servicio en el servidor y puede ser consumido por medio de peticiones HTTP desde cualquiera de los clientes existentes en la I.P.S que son:

-Aplicación móvil multiplataforma
-Aplicación web multiplataforma
-Servicios de interoperabilidad con Google
-Otros servicios internos de Neumomed

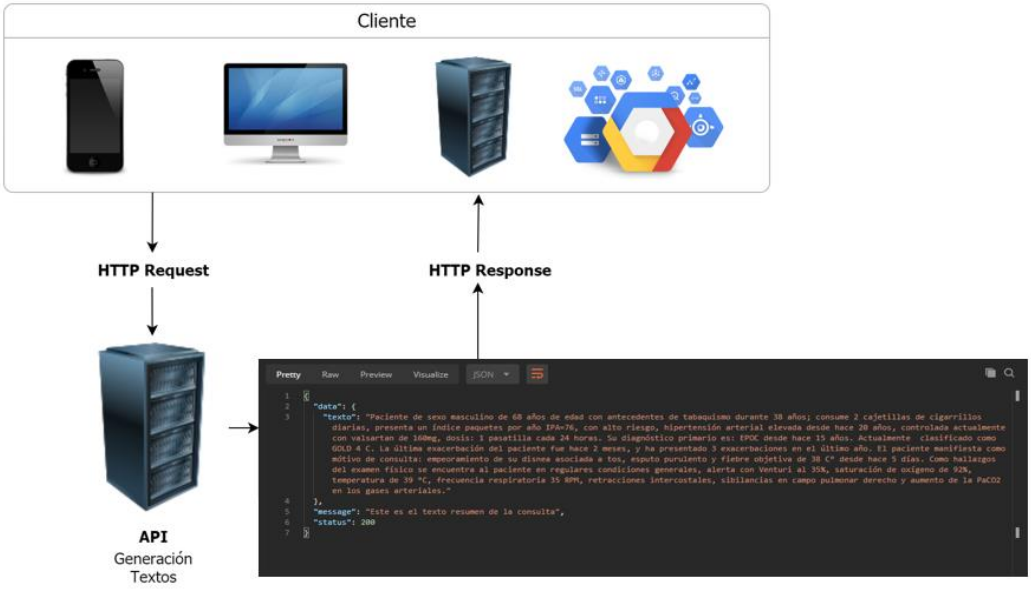

Figura 14: Diagrama UML arquitectura Cliente-Servidor

5.8 Implementación de seguridad en el proyecto

Se implementó el esquema de seguridad de Firebase para mantener protegidos los datos y reducir al mínimo el acceso a ellos, por medio de medidas de seguridad como:

-restringir el acceso a una selección de usuarios que tienen un fin administrativo o clínico para acceder a los datos personales de los pacientes.
-registrar el acceso de los usuarios a los sistemas que contienen información sensible.
-Firebase solo permite que accedan a los datos los usuarios que se encuentren autenticados.

En la Figura 15 se muestra la respuesta del algoritmo cuando se recibe una petición de un cliente no autenticado

Figura 15: Ejemplo de error de seguridad

5.9 Consumo del algoritmo desde el frontend

En la Figura 16 se presenta el diseño de la interfaz de usuario en la historia clínica digital de Neumomed. En la sección de "último encuentro" se presenta el texto que contiene la información general del paciente e información de la última consulta médica que este haya tenido en la I.P.S.

En la sección de "encuentro actual" se presenta el texto de la consulta en tiempo real, este texto es editable durante la consulta. Sin embargo una vez terminado el encuentro médico esta queda almacenada y no se puede modificar.

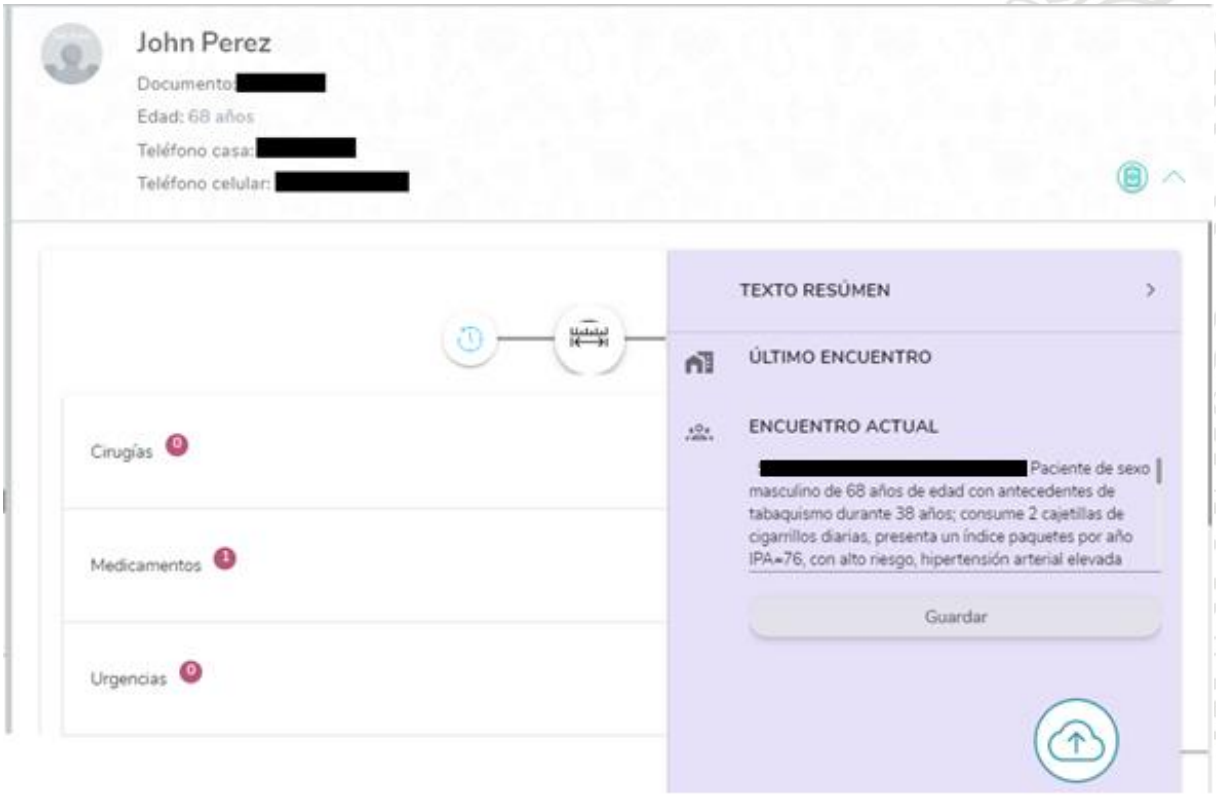

Figura 16: Interfaz gráfica de la historia clínica

5.10 Tiempos de ejecución del algoritmo

En la Figura 17 se ilustra la comparación del tiempo de ejecución del modelo vs el tiempo de ejecución de otro algoritmo de generación de textos de la IPS. Para esta prueba se usaron 24 casos clínicos que se seleccionaron en orden ascendente según la cantidad de variables que se llenaron en la consulta, es decir se muestra el comportamiento de ambos algoritmos cuando la cantidad de variables que deben incluirse aumenta.

Se puede apreciar claramente que el tiempo de ejecución del modelo es considerablemente menor, siendo su máximo de 930 ms, lo que significa una diferencia de 890 ms con el algoritmo existente lo que significa una reducción de aproximadamente 41.9% del tiempo de ejecución.

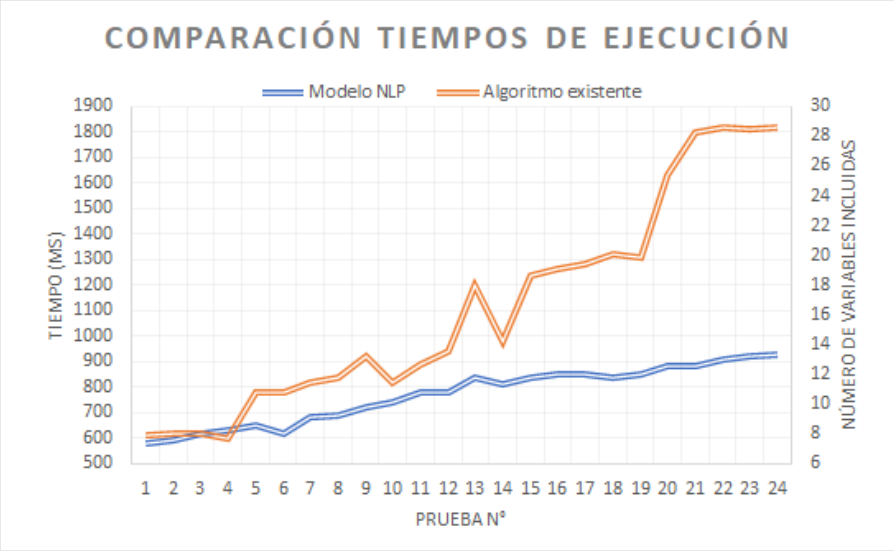

Figura 17: Tiempos de ejecución

5.11 Revisión de la gramática del texto generado

En la Figura 18 se muestra la comparación de la cantidad de errores gramaticales hallados después de revisiones de lectura de los textos generados con el modelo en contraste con los textos generados por el algoritmo de generación de textos de la IPS.

Para esta prueba se usaron 24 casos clínicos que se seleccionaron en orden ascendente según la cantidad de variables que se llenaron en la consulta, es decir se muestra el comportamiento de ambos algoritmos cuando la cantidad de variables que deben incluirse aumenta.

El máximo de errores cometidos por el modelo generado fue de 16 cuando se incluyeron 30 variables, lo que significa una diferencia de 56 errores en contraste con el algoritmo existente, denotando una reducción de 78% de error.

Desde la experiencia de los usuarios, se recibió como comentario que el texto generado por el algoritmo tiene mejor estructura, es de fácil lectura y presenta la información de manera adecuada.

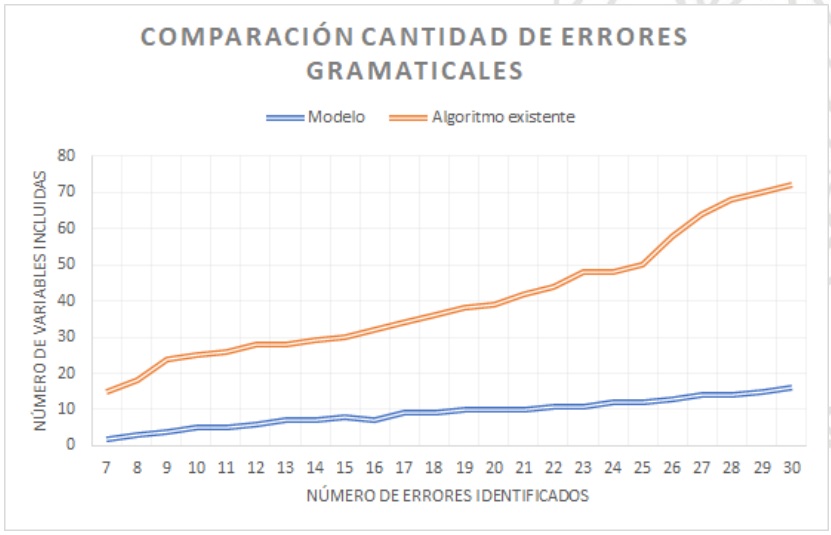

Figura 18: Comparación cantidad de errores gramaticales

6.Conclusiones

Contar con un servicio de generación de textos mediado por inteligencia artificial es una solución adecuada para la necesidad de resúmenes de consulta, pues estos al ser adaptables, escalables y parametrizables se pueden acondicionar enfocándose en la necesidad puntual del usuario. El algoritmo generado, actualmente se encuentra en fase de prueba, lo que significa que aún no está listo para ser industrializado, es decir únicamente se puede consumir desde el servidor de testeo de Neumomed; pero aún no ha sido implementado en la historia clínica digital. Sin embargo, en las pruebas realizadas en dicho servidor el algoritmo presenta una disminución en el tiempo de ejecución del 41.9% respecto a algoritmos de generación de texto existentes en la institución, esto significa que el sistema tendrá mejor rendimiento y menos posibilidades de tener errores relacionados con el tiempo de ejecución, además se traduce en una mejor experiencia de usuario al no presentar demoras en las consultas. Adicionalmente el algoritmo presenta una disminución de 78% de errores gramaticales y de sintaxis en el texto generado, lo que mejora la experiencia de usuario pues hace que el texto sea más claro, preciso, legible y confiable. Respecto a la arquitectura cliente-servidor se comprobó que es adecuada para los algoritmos de procesamiento del lenguaje natural, pues permite que los textos generados sean consumidos desde cualquier cliente, esto se demostró al consumir el algoritmo desde diferentes clientes como postman y la aplicación web de la historia clínica digital de Neumomed, además esta arquitectura permite corregir errores en el algoritmo sin que signifique una refactorización completa del código. Es necesario finalizar la fase de prueba del algoritmo para determinar si este suple completamente las necesidad para las que fue creado antes de industrializarlo, cabe aclarar que, —en caso de necesitar cambios, dada la modularidad y escalabilidad del proyecto, estos pueden construirse sobre el algoritmo existente sin necesidad de reprocesar el código. Adicionalmente como trabajo a futuro se propone que los textos generados sean procesados por medio de algoritmos de aprendizaje automatizado para extraer información médica relevante de estos. Con el fin de identificar parámetros como afecciones médicas, medicamentos, dosis, concentración, frecuencia y marcaciones de los pacientes. También se puede usar la información médica extraída para crear aplicaciones que asistan casos de uso como el soporte de decisiones clínicas y gestión de los pacientes.

7.Referencias Bibliográficas

[1].Castrillón Restrepo, A., Grisales Valencia, D. P., Londoño Posada, J., & Rua Reinosa, L. C. (2017). ¿Cuáles son los factores que llevan los porcentajes de inasistencia a la consulta médico programada y de especialistas en la IPS Interconsultas SAS durante el primer trimestre de 2017?.

[2].González, A. A. Aspectos legales de la historia clínica informatizada. Informes SEIS, 229.

[3].Álvarez Grisalez, A. D., Padilla Cuartas, S., & Villa Valderrama, D. A. (2014). Calidad de la atención médica de 20 minutos, en CEMEV IPS, sedes Bello, Envigado y Villanueva, 2014.

[4].Ohno-Machado, L. (2011). Realizing the full potential of electronic health records: the role of natural language processing. Journal of the American Medical Informatics Association, 18(5), 539-539.

[5]. McGhee, K. (2019). Machine Learning in Medicine. Machine Learning.

[6].“An introduction to machine learning with scikit-learn”. scikit-learn.org, scikit-learn developers. (2019). [Online]. Disponible en: http://scikit-learn.org/stable/tutorial/basic/tutorial.html [Accedido: 01- May- 2020].

[7].Jordan, M. I., & Mitchell, T. M. (2018). Machine learning: Trends, perspectives, and prospects. Science, 349(6245), 255–260. https://doi.org/10.1126/science.aaa8415

[8].“Cloud machine learning engine”, cloud.google.com. (2019). Disponible en :https://cloud.google.com/ml-engine/ [Acedido: 01- May- 2020].

[9].Claesen, M., & De Moor, B. (2018). Hyperparameter Search in Machine Learning, 10–14. Retrieved from http://arxiv.org/abs/1502.02127

[10].Lan, T., Zhang, Y., Jiang, C., Yang, G., & Zhao, Z. (2018). Automatic identification of Spread F using decision trees. Journal of Atmospheric and Solar-Terrestrial Physics.

[11].DecisionTreeClassifier, documentacion libreria, 2018, https://scikit-learn.org/stable/modules/generated/sklearn.tree.Decision TreeClassifier.html [Accedido: 8- May- 2020]

[12].Liu, W., Wang, Z., Liu, X., Zeng, N., Liu, Y., & Alsaadi, F. E. (2019). A survey of deep neural network architectures and their applications. Neurocomputing

[13].Dreiseitl, S., & Ohno-Machado, L. (2002). Logistic regression and artificial neural network classification models: A methodology review. Journal of Biomedical Informatics.

[14].About Us, Documentacion scikitlearn, 2018 https://scikit-learn.org/stable/index.html [Accedido: 28- May- 2020]

[15].Tensorflow, Google, 2018 https://www.tensorflow.org/ [Accedido: 28- May- 2020]

[16]Goutte, C., & Gaussier, E. (2005). A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation, 3408, 345–359. https://doi.org/10.1007/978-3-540-31865-1_25

[17].Accuracy, Precision, Recall or F1?, Ping Shung, Koo ,2018, https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb 37c5cb9 [Accedido: 28- May- 2020]

[18].Ruuska, S., Hämäläinen, W., Kajava, S., Mughal, M., Matilainen, P., & Mononen, J. (2018). Evaluation of the confusion matrix method in the validation of an automated system for measuring feeding behaviour of cattle. Behavioural Processes.

[19].García-Holgado, A., & García-Peñalvo, F. J. (2014, November). Architectural pattern for the definition of eLearning ecosystems based on Open Source developments. In 2014 International Symposium on Computers in Education (SIIE) (pp. 93-98). IEEE.

[20].Milosavljević, B., & Tešendić, D. (2010). Software architecture of distributed client/server library circulation system. The Electronic Library.

[21].Flask, Documentacion Flask, 2020 http://flask.pocoo.org/ [Accedido: 18- May- 2020]

8.Anexos

8.1 Anexo 1: Documentación del proyecto.

Endpoint para entrenar el modelo NLP

Descripción: este endpoint ejecuta el algoritmo de machine learning y genera un archivo .pkl que contiene el modelo entrenado, luego carga ese archivo a google cloud storage.

PATH: https://pruebas-api.neumobot.com/generacion_textos/formularios/api/v1/entrenarModelo
Method: POST
Params:

```
{
"doc": Form-data,
"origen": String
}
```

Response:

```
{
  "data": {
        "f1": Array,
        "formulario": String,
        “max_depth”: Number,
        “min_samples_split”: Number
  },
  "message": String,
  "status": Number
}
```

Error: Error en el entrenamiento del modelo

Response:

```
{
        "data": [],
        "status": 500,
        "message": "Error al entrenar el modelo "
}
```

Endpoint para construir el texto resumen de un formulario
Descripción: este endpoint debe ejecutarse una vez llenados los formularios de la consulta médica, para generar el texto resumen de la consulta. Recibe como parámetros de entrada el docId, tipo esquema y origen de los formularios. Retorna un Json con el docId del formulario y un texto que contiene el texto resumen.

Notas:

1. La variable de salida “texto” contiene el texto resumen generado como un string plano
2. La variable de salida “textoApp” contiene el texto resumen generado, separado en varias frases dentro dentro un array

PATH: https://pruebas-api.neumobot.com/generacion_textos/formularios/api/construirTextoResumen
Method: GET

Params:

```
{
"docId": String,
"tipo_esquema: String,
"origen": String
}
```

Response:

```
{
  "data": {
        "docId": String,
        "texto": String,
        “textoApp”: Array
  },
  "message": String,
  "status": Number
}
```

Error: Consultar una plantilla inexistente en la base de datos

Response:

```
{
        "data": [],
        "status": 500,
        "message": "No existe plantilla creada para este caso "
}
```

Error: enviar docId de un formulario no existente

```
{
        "data": [],
        "status": 404,
        "message": "Error en la lectura del formulario "
}
```

Error: el modelo NLP no es adecuado o dejó de funcionar correctamente

```
{
        "data": [],
        "status": 500,
        "message": "Error en la implementación del modelo NLP "
}
```

Error: error interno en la ejecución del reemplazo del texto en la plantilla

```
{
        "data": [],
        "status": 500,
        "message": "Error en la generación de texto."
}
```

Endpoint para construir el texto resumen de información general y último encuentro del paciente

Descripción: este endpoint debe ejecutarse en el inicio de un encuentro, recibe como parámetro de entrada el documento del paciente y retorna un Json con el docId del formulario de

información general del paciente y un texto que contiene un resumen con las variables más relevantes de dicho formulario e informa cuál y cuándo fue el último encuentro del paciente, y el concepto y plan emitidos en este.

Nota:

1. Este proyecto consume el endPoint *construirTextoResumen* por tanto hereda de éste algunos casos de manejo de errores.

PATH: https://pruebas-api.neumobot.com/generacion_textos/formularios/api/v1/construirResumenGeneral
Method: GET

Params:

```
{ "documento": Number }
```

Response:

```
{
  "data": {
        "docId": String,
        "texto": String,
  },
  "message": String,
  "status": Number
}
```

Error: Error en la lectura de información general del paciente

```
{
        "data": [],
        "status": 500,
        "message": "Error en la lectura de información general del paciente"
}
```

Error: Error en la consulta del concepto o plan

```
{
        "data": [],
        "status": 500,
        "message": "No se pudo obtener concepto o plan del último encuentro."
}
```