EDUCACIÓN

SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico de Orizaba

DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

OPCIÓN I.- TESIS

TRABAJO PROFESIONAL

"DETECCIÓN TEMPRANA DE DESÓRDENES HEPÁTICOS A PARTIR DEL ANÁLISIS DE BIOMARCADORES UTILIZANDO MÉTODOS DE ENSAMBLES DE APRENDIZAJE AUTOMÁTICO"

QUE PARA OBTENER EL GRADO DE:

MAESTRO EN SISTEMAS COMPUTACIONALES

PRESENTA:

I.S.C. Luis Rodolfo Cabrera Elías

DIRECTOR DE TESIS:

Dr. José Luis Sánchez Cervantes

CODIRECTOR DE TESIS:

Dr. Giner Alor Hernández

ORIZABA, VERACRUZ, MÉXICO. **FEBRERO 2024**

Orizaba, Veracruz, 16/febrero/2024
Dependencia: **División de Estudios de Posgrado e Investigación**
Asunto: **Autorización de Impresión**
OPCION: I

C. LUIS RODOLFO CABRERA ELÍAS
CANDIDATO A GRADO DE MAESTRO EN:
SISTEMAS COMPUTACIONALES
P R E S E N T E.-

De acuerdo con el Reglamento de Titulación vigente de los Centros de Enseñanza Técnica Superior, dependiente de la Dirección General de Institutos Tecnológicos de la Secretaría de Educación Pública y habiendo cumplido con todas las indicaciones que la Comisión Revisora le hizo respecto a su Trabajo Profesional titulado:

" Detección temprana de desórdenes hepáticos a partir del análisis de biomarcadores utilizando métodos de ensambles de aprendizaje automático"

comunico a Usted que este Departamento concede su autorización para que proceda a la impresión del mismo.

A T E N T A M E N T E
Excelencia en Educación Tecnológica®
CIENCIA – TÉCNICA - CULTURA®

DRA. OFELIA LANDETA ESCAMILLA
ENCARGADA DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

EDUCACIÓN SECRETARÍA DE EDUCACIÓN PÚBLICA | TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLOGICO DE ORIZABA
DIV. DE EST. DE POSGRADO E INVEST.
OG-13-F06

Orizaba, Veracruz, 08/febrero/2024
Asunto: **Revisión de trabajo escrito**

C. OFELIA LANDETA ESCAMILLA
JEFE DE LA DIVISIÓN DE ESTUDIOS
DE POSGRADO E INVESTIGACIÓN
P R E S E N T E.-

Los que suscriben, miembros del jurado, han realizado la revisión de la Tesis del (la) C.

LUIS RODOLFO CABRERA ELÍAS

La cual lleva el título de:

Detección temprana de desórdenes hepáticos a partir del análisis de biomarcadores utilizando métodos de ensambles de aprendizaje automático

Y concluyen que se acepta.

A T E N T A M E N T E
Excelencia en Educación Tecnológica®
CIENCIA – TÉCNICA - CULTURA®

PRESIDENTE:	DR. JOSÉ LUIS SÁNCHEZ CERVANTES	FIRMA
SECRETARIO:	DR. GINER ALOR HERNÁNDEZ	FIRMA
VOCAL:	M.C.E. BEATRIZ ALEJANDRA OLIVARES ZEPAHUA	FIRMA
VOCAL SUP.:	M.S.C. LUIS ÁNGEL REYES HERNÁNDEZ	FIRMA

TA-09-18

CARTA DE ORIGINALIDAD

En la ciudad de Orizaba, Veracruz, el día 08 de febrero del año 2024, el (la) que suscribe **Luis Rodolfo Cabrera Elías**, alumno (a) del programa de **Maestría en Sistemas Computacionales** con número de control **M16011057**, manifiesta que es autor(a) del trabajo de tesis titulado **"Detección temprana de desórdenes hepáticos a partir del análisis de biomarcadores utilizando métodos de ensambles de aprendizaje automático"** y declaro que el trabajo es original ya que sus contenidos son producto de mi directa contribución intelectual. Todos los datos y las referencia a materiales ya publicados están debidamente identificados con su respectivo crédito e incluidos en las notas bibliográficas y en las citas que se destacan como tal y, en los casos que así lo requieran, cuento con las debidas autorizaciones de quieres poseen los derechos patrimoniales. Por lo tanto, me hago responsable de cualquier litigio o reclamación relacionada con derechos de propiedad intelectual, exonerando de toda responsabilidad al Tecnológico Nacional de México / Instituto Tecnológico de Orizaba.

Luis Rodolfo Cabrera Elías
Nombre y Firma

TA-09-F16

CARTA DE CESIÓN DE DERECHOS

En la ciudad de Orizaba, Veracruz, el día 08 del mes de febrero del año 2024, el (la) que suscribe **Luis Rodolfo Cabrera Elías**, alumno (a) del programa de **Maestría en Sistemas Computacionales** con número de control **M16011057**, manifiesta que es autor(a) del trabajo de tesis bajo la dirección de José Luis Sánchez Cervantes y cede los derechos del trabajo de tesis titulado **"Detección temprana de desórdenes hepáticos a partir del análisis de biomarcadores utilizando métodos de ensambles de aprendizaje automático"** al **Tecnológico Nacional de México / Instituto Tecnológico de Orizaba** para su difusión y divulgación, con fines académicos y de investigación.

Queda estrictamente prohibido reproducir el contenido textual, gráficas o datos del trabajo sin el permiso expreso del **Tecnológico Nacional de México / Instituto Tecnológico de Orizaba**. Este puede obtenerse escribiendo a la siguiente dirección: msc@orizaba.tecnm.mx. Si el permiso se otorga, cualquier usuario deberá dar el agradecimiento correspondiente y citar la fuente del mismo.

Luis Rodolfo Cabrera Elías
Nombre y Firma

Agradecimientos

Al Tecnológico Nacional de México – Campus Orizaba por la oportunidad de cursar el programa de posgrado de Maestría en Sistemas Computacionales.

Al Consejo Nacional de Humanidades, Ciencias y Tecnologías por el recurso financiero otorgado para la culminación de mis estudios de posgrado.

A mi madre Ana Luisa Elías Rodríguez QEPD., la cual me dio la vida y siempre me cuidó y a quien le hubiera dado mucho orgullo y felicidad de ver hasta donde he llegado. A mi abuela Vitalia Rodríguez QEPD., que fue siempre como una segunda madre y me dio todo su apoyo. A mi tía María Guadalupe Elías Rodríguez, quien es como mi tercera madre, y toda la vida ha visto por mí y siempre me ha alentado a cumplir mis sueños y metas. A mi padre Cornelio Cabrera Alvarado que siempre ha creído en mí.

A la familia Robles Casas quienes siempre me han acogido como un miembro más de su familia y me han dado su apoyo y cariño incondicional. A mi novia Andrea Robles Piza quien me ha dado motivación para no rendirme y superar cualquier desafío. A la mamá de mi novia, Josefina Piza Pineda la cual igual me ha acogido como si fuera un hijo más y, por todo el cariño y apoyo que me ha dado.

A mis maestros de la Maestría en Sistemas Computacionales quienes compartieron sus conocimientos y me ayudaron a crecer profesionalmente, especialmente al Dr. José Luis Sánchez Cervantes y a la maestra Beatriz Alejandra Olivares Zepahua por todo el apoyo proporcionado para concluir satisfactoriamente este proyecto de tesis.

A mis amigos de la Orden Oculta de Sistemas, los cuales son como hermanos para mí y nos retroalimentamos con conocimiento para resolver cualquier problemática presentada. A mis amistades más cercanas por creer siempre en mí.

Al Dr. Luis Rolando Guarneros Nolasco de la UTCV y al Dr. Ángel Enrique Escudero Fabre del Sanatorio Escudero, a quienes proporcionaron el conocimiento adecuado para mejorar y llegar a los resultados del presente trabajo.

Índice General

Índice de Tablas

Índice de Figuras

Índice de Listas

Resumen

La salud es un tema fundamental que hace que el ser humano se encuentre en condiciones favorables llevando una vida estable y duradera; sin embargo, si no se llega a cuidar, existe el riesgo que sea propenso a caer en alguna enfermedad y afecte a alguno de sus órganos, en especial al hígado, causando daño hepático por diversos factores, siendo el tema principal de este tema de investigación.

Para resolver este problema, se propuso utilizar algoritmos de ensambles de aprendizaje automático.

El objetivo del presente trabajo fue desarrollar un módulo basado en un algoritmo de ensamble de aprendizaje automático que permita la detección temprana de desórdenes hepáticos a partir del análisis de biomarcadores. Para llegar a la solución, primero se analizaron los trabajos relacionados con el tema propuesto, además se estudiaron los tipos de algoritmos de aprendizaje de ensamble tales como: *bagging, boosting* y *stacking* que permitan la detección de desórdenes hepáticos. Como segundo aspecto, se investigaron e identificaron los biomarcadores preponderantes para la detección de desórdenes hepáticos.

Una vez terminado el proceso de analizar y la parte de la investigación, se procedió a diseñar el modelo de entrenamiento basado en el algoritmo de ensamble factible para la detección temprana de desórdenes hepáticos.

Finalmente, se entrenó el módulo diseñado de tal manera que sea implementado con las interfaces Web junto con el repositorio de información con los que se integró el algoritmo de ensamble.

Para la comprobación de su efectividad, se evaluó al menos un caso de estudio como prueba de concepto que permitió describir los resultados y conclusiones obtenidas.

Abstract

Health is a fundamental issue that makes the human being to be in favorable conditions leading a stable and lasting life; however, if it is not taken care of, there is a risk of falling into some disease and affect any of its organs, especially the liver, causing liver damage by various factors, being the main topic of this research topic.

To solve this problem, it was proposed to use machine learning ensemble algorithms.

The objective of the present work was to develop a module based on a machine learning ensemble algorithm that allows the early detection of liver disorders from biomarker analysis. To reach the solution, first, the works related to the proposed topic were analyzed, and the types of ensembles learning algorithms such as bagging, boosting and stacking that allow the detection of liver disorders were studied. As a second aspect, the preponderant biomarkers for the detection of liver disorders were investigated and identified.

Once the analysis process and the research part were completed, we proceeded to design the training model based on the feasible assembly algorithm for the early detection of liver disorders.

Finally, the designed module was trained in such a way that it will be implemented with the Web interfaces together with the information repository with which it will be integrated.

To verify its effectiveness, at least one case study was evaluated as a proof of concept to describe the results and conclusions obtained.

Introducción

El hígado es uno de los órganos más importantes del cuerpo humano, que se encarga de controlar los niveles químicos de la sangre, además, ayuda a descomponer las grasas para su digestión y absorción, así como también metaboliza los medicamentos que se encuentran en la sangre para que sean más fáciles de utilizar por el cuerpo.

El principal problema que se estudia en esta investigación es que existen varios trastornos del hígado que necesitan tratamiento temprano por parte de los médicos, tal es el caso del daño hepático causado por obesidad, o hepatotoxicidad generada por el consumo de medicamentos, drogas o incluso remedios caseros.

Este trabajo proporciona una alternativa de solución a la problemática planteada haciendo uso de algoritmos de ensambles de aprendizaje automático, de tal manera que permita la pronta detección de desórdenes hepáticos, para así conocer las probabilidades de que una persona contraiga alguna enfermedad o cualquier otro problema de salud relacionada con el hígado y se detecte en tiempo y forma a fin de tomar las medidas preventivas. Se analizan los algoritmos de ensamble automático, de tal manera que se seleccione el mejor algoritmo que obtenga los mejores resultados y se llevó a cabo al menos un caso de estudio como caso de prueba.

Este trabajo se encuentra conformado por los siguientes cinco capítulos, el capítulo uno aborda los antecedentes del proyecto que incluye el marco teórico, planteamiento del problema, situación tecnológica, económica y operativa de la empresa, objetivo general y específicos y la justificación; en el capítulo dos se presenta el estado de la práctica y el análisis comparativo de los trabajos relacionados con el tema; el capítulo tres contiene la metodología, donde se describe la serie de pasos que se llevaron a cabo para dar solución a este trabajo; el capítulo 4 describe los resultados obtenidos mediante un caso de estudio y finalmente; el capítulo 5 describe las conclusiones del presente proyecto así como las recomendaciones para futuros trabajos.

Capítulo 1. Antecedentes

En este capítulo se expondrán temas que serán de interés para este tema de tesis, como son la problemática por tratar, objetivo general y específicos y la justificación de este trabajo.

1.1 Marco Teórico

A continuación, se definen algunos términos que se utilizarán para el trabajo de investigación.

1.1.1 Hígado

El hígado [1] es un órgano del cuerpo humano de tamaño grande, que se encuentra en el lado derecho del abdomen. El hígado tiene como principal función el filtrado de sangre originario en el tracto digestivo, previo a permitirle circular al resto del organismo, así como también de digerir, absorber y procesar los alimentos. El hígado incluso libera de tóxicos y sustancias químicas y permite la metabolización de los medicamentos y/o drogas. Al hacerlo, el hígado, produce bilis, esta termina diseminándose hasta los intestinos. El hígado además produce proteínas indispensables para permitir la coagulación de la sangre y otras funciones varias.

Es importante mencionar que el hígado se perjudica en procesos autoinmunes, infecciones víricas, toxicidad por fármacos y/o drogas considerando también sus respectivos metabolitos, metabolopatías, y distintas afectaciones genéticas.

1.1.2 Daño hepático

Lesión que afecta al hígado por consumo de medicamentos, abuso de drogas además causado por remedios caseros como es el caso de productos de herboristería [2].

1.1.3 Enfermedad del hígado graso no alcohólica (EHGNA)

Consiste en la afectación del metabolismo que se destaca por la aglomeración de grasa en los hepatocitos de las personas sin consumo relevante de medicamentos hepatotóxicos, alcohol, ni otros motivos conocidos que provoquen esteatosis. En la actualidad es la afectación hepática crónica más habitual a nivel mundial [3], que va a la par de la obesidad y diabetes; de tal manera que está enfermedad tiende a evolucionar llegando a causar cirrosis.

El hígado graso no alcohólico asimismo se destaca por la presencia de esteatosis simple desprovista de cambios inflamatorios, necrosis o fibrosis.

1.1.4 Cirrosis

La cirrosis es una condición grave que involucra la formación de tejido cicatricial en el hígado, generalmente causada por enfermedades hepáticas como la hepatitis o el consumo crónico de alcohol [4]. Cuando el hígado se lesiona, intenta recuperarse formando tejido cicatricial, pero a medida que la cirrosis progresa, se acumula más tejido cicatricial, dificultando la función hepática. En etapas avanzadas, la cirrosis es mortal y, por lo general, el daño hepático no es reversible. Sin embargo, con un diagnóstico temprano y tratamiento de la causa subyacente, es posible limitar el daño o, en casos excepcionales, revertir la condición.

1.1.5 Hepatitis

La hepatitis es una condición caracterizada por la inflamación del hígado, lo cual implica la hinchazón del órgano debido a lesiones o infecciones [5]. Esta inflamación causa daño al hígado y afectar su función adecuada. La hepatitis se presenta como una infección aguda, de corto plazo, o como una infección crónica, de largo plazo. Algunos tipos de hepatitis provocan solo infecciones agudas, mientras que otros dan lugar a infecciones tanto agudas como crónicas.

1.1.6 La Hepatotoxicidad

La hepatotoxicidad con siglas (HTX) [2] consiste en un daño al hígado o lesión generada por la ingesta de un medicamento u otros elementos no farmacológicos; dentro de este conjunto se consideran a los antiinflamatorios no esteroideos, los

antituberculosos, los antibióticos, asimismo medicamentos especializados en tratar afectaciones del sistema cardiovascular.

1.1.7 Biomarcadores

El término "biomarcador", acrónimo de "marcador biológico", consisten en las métricas que se utilizan para realizar una evaluación clínica, como la presión arterial, y se utilizan para controlar y predecir los estados de salud de las personas o de las poblaciones, de modo que se permita planificar la intervención terapéutica adecuada. Los biomarcadores se usan solos o en combinación para evaluar el estado de salud o enfermedad de un individuo.

Las características principales de los biomarcadores se encuentran [6]:

- Seguro y fácil de medir
- Seguimiento rentable
- Modificable con tratamiento

1.1.8 Inteligencia Artificial

La inteligencia artificial (IA) es una de las ramas de las ciencias de la computación y en término general implica el uso de una computadora para modelar un comportamiento inteligente, así como un pensamiento crítico, similar al de ser humano [7]. En el año 2016, la mayor parte de las inversiones en investigación de IA se realizaron en aplicaciones de atención médica en comparación con otros sectores [8].

Alguna de las ramas de la IA es: *Machine learning,* Redes neurales, Sistemas expertos y Procesamiento del lenguaje natural.

1.1.9 Aprendizaje Automático (*Machine learning*)

Es una derivación de la Inteligencia Artificial (IA) enfocada en replicar y/o imitar el aprendizaje tal como un ser humano lo hace, incrementando su precisión gradualmente [9]. El aprendizaje automático se utiliza con frecuencia en el campo de la predicción de diagnósticos médicos porque es capaz de descubrir importantes patrones subyacentes dentro de complejos datos médicos. La clasificación es una

de las tareas de aprendizaje automático más aplicadas que asigna un valor objetivo desconocido de una nueva muestra a una de las clases predefinidas.

Cuando se entrena un algoritmo de aprendizaje automático con datos, se genera una salida de información conocida como modelo de aprendizaje automático.

1.1.10 Aprendizaje supervisado

El aprendizaje supervisado [10] se deriva de *Machine Learning* (Aprendizaje automático), y se define como una técnica que se utiliza para analizar un conjunto de datos, donde los datos entrenamiento se etiquetan. El término “supervisado” hace referencia al sistema de aprendizaje sobre etiquetas, es decir, las etiquetas se utilizan en problemas de clasificación, de tal manera que los datos de entrenamiento etiquetados proporcionan criterios claros para la optimización del modelo.

Dentro del aprendizaje automático existen diversas técnicas. A continuación, se mencionan cinco técnicas que forman parte de los conceptos del presente proyecto de investigación [11].

1) Regresión logística
2) Máquinas de vectores de soporte (SVM)
3) Algoritmo de los vecinos más cercanos (k-NN)
4) Árboles de decisión
5) Algoritmo *RandomForest*

1.1.11 Regresión logística

La regresión logística [12] es un método estadístico utilizado para modelar la relación entre una variable dependiente categórica y una o más variables independientes. En lugar de predecir valores numéricos, como en la regresión lineal, la regresión logística se utiliza para predecir la probabilidad de que un evento ocurra o la pertenencia a una categoría específica.

La regresión logística utiliza una función logística para modelar la probabilidad de éxito o fracaso en función de las variables independientes. Esta función logística transforma la combinación lineal de las variables independientes en una

probabilidad en el rango de 0 a 1. Luego, se utiliza un umbral para clasificar las observaciones en diferentes categorías.

La regresión logística es ampliamente utilizada en diversas áreas, como la medicina, la psicología, la economía y el análisis de datos. Se utiliza para analizar el efecto de las variables independientes en la probabilidad de un evento, realizar predicciones y tomar decisiones basadas en la probabilidad estimada.

En resumen, la regresión logística es una técnica estadística utilizada para modelar la relación entre una variable dependiente categórica y variables independientes, estimando la probabilidad de un evento o la pertenencia a una categoría específica.

1.1.12 Máquinas de Vectores de Soporte (SVM, *Support Vector Machines*)

Las máquinas de vectores de soporte, también conocidas como SVM por sus siglas en inglés, son un conjunto de algoritmos de aprendizaje supervisado ampliamente utilizados en clasificación, regresión y detección de valores atípicos [13]. Estos algoritmos construyen un modelo basado en un subconjunto de puntos de entrenamiento para predecir la clase de nuevas muestras.

Las características distintivas de las SVM son las siguientes:

- Son adecuadas para casos en los que el número de dimensiones es mayor que el número de muestras.
- Utilizan un subconjunto de puntos de entrenamiento, conocidos como vectores de soporte, en la función de decisión. Esto resulta en un enfoque eficiente en términos de memoria.

En resumen, las máquinas de vectores de soporte son algoritmos de aprendizaje supervisado utilizados en clasificación, regresión y detección de valores atípicos. Se caracterizan por su capacidad para manejar conjuntos de datos de alta dimensionalidad y por su eficiencia en memoria al utilizar un subconjunto de puntos de entrenamiento en la función de decisión.

1.1.13 El vecino más cercano (*k-nearest neighbors*)

En el campo del aprendizaje automático, se implementó una técnica para reconocer patrones en los datos sin la necesidad de una coincidencia exacta con patrones almacenados previamente. En lugar de eso, se considera la proximidad entre los casos, donde los casos similares están cerca y los casos diferentes están separados entre sí. La distancia entre dos casos se utiliza como medida de disimilaridad [14].

Los casos que están cercanos entre sí se llaman "vecinos". Cuando se presenta un nuevo caso, se calcula su distancia con respecto a los casos del modelo. Luego, se toman las clasificaciones de los vecinos más cercanos y se asigna al nuevo caso la categoría que contiene el mayor número de vecinos más cercanos.

Se especifica un valor k que indica el número de vecinos más cercanos que se examinan en el proceso. Este enfoque se conoce como "método de vecinos más cercanos".

Además de la clasificación, el método de vecinos más cercanos también se utiliza para predecir valores continuos. En esta situación, se calcula la media o el valor objetivo promedio de los vecinos más cercanos para obtener el valor predicho del nuevo caso.

1.1.14 Árboles de decisión (*Decision Trees DTs)*

Los árboles de decisión [15] son métodos de aprendizaje supervisado que buscan crear un modelo que predice el valor de una variable objetivo utilizando reglas de decisión simples derivadas de las características de los datos. Estos árboles se utilizan tanto para la clasificación como para la regresión. A medida que el árbol se vuelve más profundo, las reglas de decisión se vuelven más complejas y el modelo se ajusta más precisamente a los datos.

Algunas características clave de los árboles de decisión son:

- Son fáciles de entender e interpretar, ya que se visualizan la estructura del árbol y cómo se toman las decisiones.
- Manejan tanto datos numéricos como categóricos.

- Permiten la validación del modelo a través de pruebas estadísticas para evaluar su fiabilidad.
- Utilizan un modelo de caja blanca, lo que significa que, si se observa una situación en el modelo, la explicación de la condición se entiende fácilmente mediante lógica booleana.

En resumen, los árboles de decisión son métodos de aprendizaje supervisado que utilizan reglas de decisión simples para predecir valores. Son fáciles de interpretar, manejan diferentes tipos de datos, permiten la validación del modelo y proporcionan una explicación clara de cómo se toman las decisiones en base a las características de los datos.

1.1.15 Bosque Aleatorio (*Random Forest)*

Random Forests es una extensión de la idea de *Bagging* (dicho concepto se explica más adelante), se utiliza para una variable de respuesta categórica, a la que se hace referencia como "clasificación", o una respuesta continua, a la que se hace referencia como "regresión" [16]. De manera similar, las variables predictoras se clasifican en categóricas o continuas. Además, combina la salida de múltiples árboles de decisión para llegar a un único resultado.

Características de *Random Forest*:

- Manejan naturalmente tanto la regresión como la clasificación (multiclase).
- Son relativamente rápidos para entrenar y predecir.
- Depende sólo de uno o dos parámetros de ajuste.
- Se utiliza directamente para problemas de alta dimensión.
- Se implementa fácilmente en paralelo.

Los árboles de decisión consideran todas las divisiones de características posibles, mientras que los *Random Forests* solo seleccionan un subconjunto de esas características.

1.1.16 Aprendizaje por conjuntos (*Ensemble Learning*)

Los métodos de aprendizaje por conjuntos explotan múltiples algoritmos de aprendizaje automático para producir resultados predictivos débiles basados en

características extraídas a través de una diversidad de proyecciones de datos, y fusionan los resultados con varios mecanismos de votación para lograr un mejor rendimiento que el obtenido solo con cualquier algoritmo constituyente.

Por lo cual, el aprendizaje en conjunto tiene como objetivo integrar la fusión de datos, el modelado de datos y la extracción de datos en un marco unificado [17]. Específicamente, el aprendizaje conjunto primero extrae un conjunto de características con una variedad de transformaciones. Sobre la base de estas características, se utilizan múltiples algoritmos de aprendizaje para producir resultados predictivos débiles. Finalmente, el aprendizaje conjunto fusiona el conocimiento informativo de los resultados anteriores obtenidos para lograr el descubrimiento de conocimiento y un mejor desempeño predictivo a través de esquemas de votación de manera adaptativa.

1.1.17 Bagging

El método de *Bagging* [18], también conocido como agregación de arranque, es una técnica comúnmente utilizada en el aprendizaje de conjuntos para reducir la varianza en conjuntos de datos ruidosos. En *Bagging*, se selecciona aleatoriamente una muestra de datos del conjunto de entrenamiento, permitiendo la posibilidad de seleccionar múltiples veces los mismos puntos de datos mediante reemplazo. Después de generar varias muestras de datos, se entrenan modelos débiles de forma independiente. Dependiendo del tipo de tarea, como regresión o clasificación, se promedian o se toma la mayoría de las predicciones generadas por estos modelos para obtener una estimación más precisa.

Algunas características fuertes de *Bagging* son:

- Facilidad de implementación: las bibliotecas de Python facilitan la combinación de predicciones de aprendices base o estimadores para mejorar el rendimiento del modelo.
- Reducción de la varianza: *Bagging* reduce la varianza dentro de un algoritmo de aprendizaje. Esto es particularmente útil con datos de alta dimensión, donde los valores faltantes conducen a una mayor varianza, lo que los hace

más propensos al sobreajuste y evita la generalización precisa a nuevos conjuntos de datos.

1.1.18 Boosting

Boosting [19] es una técnica predominante para abordar problemas de clasificación binaria. Este tipo de algoritmos mejoran la predicción al convertir una cantidad de aprendizajes débiles en aprendizajes fuertes. De acuerdo con este enfoque, las muestras en el conjunto de entrenamiento se vuelven a ponderar durante la fase de aprendizaje para generar múltiples aprendizajes.

En este tipo de algoritmos primero se construye un modelo en el conjunto de datos de entrenamiento, luego se construye un segundo modelo para rectificar los errores presentes en el primer modelo. Este procedimiento continúa hasta que los errores se minimicen y el conjunto de datos se prediga correctamente.

1.1.19 AdaBoost

Presentado por Yoav Freund y Robert Schapire, es un algoritmo de refuerzo, que en este tipo de algoritmo el peso de las muestras mal clasificadas en el conjunto de entrenamiento aumenta en cada iteración [19]. Por lo tanto, aumenta la posibilidad de seleccionar muestras mal clasificadas para el conjunto de entrenamiento y se clasifican correctamente más muestras. El algoritmo más común que se usa con AdaBoost son los árboles de decisión con un nivel, es decir, con árboles de decisión con solo una división.

Este algoritmo construye un modelo inicial donde todos los puntos de datos tienen pesos iguales. Luego, asigna pesos más altos a aquellos puntos que están clasificados incorrectamente. De esta manera, los puntos con pesos más altos se vuelven más importantes en la construcción del siguiente modelo. Este proceso se repite, entrenando sucesivos modelos hasta que se alcance un error mínimo.

1.1.20 Gradient Boosting

Es un procedimiento predefinido, preciso y efectivo que se utiliza ampliamente tanto en problemas de regresión como de clasificación en diversas áreas, incluyendo la búsqueda en la web con fines de clasificación [20].

Al igual que el aumento y la potenciación, el *boosting* de gradiente es una metodología que se aplica sobre otro algoritmo de aprendizaje automático. De manera informal, el aumento de gradiente implica dos tipos de modelos:

- Un modelo de aprendizaje automático "débil" que suele ser un árbol de decisión.
- Un modelo de aprendizaje automático "fuerte" que se compone de varios modelos débiles.

En el aumento de gradiente, en cada paso, se entrena un modelo débil nuevo para predecir el error del modelo fuerte actual (llamado pseudorespuesta). Por ahora, el "error" es la diferencia entre la predicción y una etiqueta de regresión. El modelo débil (es decir, el error) se agrega al modelo seguro con un signo negativo para reducir su error.

El *boosting* de gradientes es iterativo. Cada iteración invoca la siguiente fórmula:

$$F_{i+1} = F_i - f_i$$

Donde:

$F_i \; es \; el \; modelo \; sólido \; en \; el \; paso \; i.$

$f_i \;\; es \; el \; modelo \; débil \; en \; el \; paso \; i$

Esta operación se repite hasta que se cumple un criterio de detención, como una cantidad máxima de iteraciones o si el modelo (fuerte) comienza a sobre ajustar según lo medido en un conjunto de datos de validación por separado.

1.1.21 LightGBM

LightGBM [21] es un marco de trabajo de *boosting* basado en árboles de decisión que se utiliza para tareas de aprendizaje automático como categorización y clasificación. Su objetivo es maximizar o minimizar una función objetivo mediante la técnica de *gradient boosting*, que combina clasificadores simples como árboles de decisión de profundidad limitada. Se diseñó específicamente para la distribución y ofrece varias ventajas, como mayor velocidad de entrenamiento, menor uso de

memoria, mejor precisión y soporte para aprendizaje paralelo, distribuido y GPU. Además, es capaz de manejar conjuntos de datos a gran escala.

1.1.22 Stacking

El apilamiento [22] es la técnica introducida por Wolpert que implica la formación de combinaciones lineales de diferentes predictores para dar resultados generales mejorados. Se compone de dos fases. En la primera fase, se aprenden diferentes modelos como J48, *Naive Bayes* y *Random Forest* sobre la base del conjunto de datos y su salida ayudaría a generar un nuevo conjunto de datos. En la segunda fase, el conjunto de datos se usa con el algoritmo de aprendizaje para dar el resultado final.

Nota: Los ensambles de apilamiento se descartaron del presente trabajo debido a los siguientes motivos:

- Complejidad y costo computacional: Los ensambles de apilamiento tienden a ser más complejos y requieren más recursos computacionales en comparación con otros métodos de ensamble más simples, como el promedio o la votación. El proceso de entrenar múltiples modelos base, combinar sus predicciones y ajustar el metamodelo suele ser computacionalmente costoso y requerir más tiempo de entrenamiento.
- Tamaño y calidad del conjunto de datos: Los ensambles de apilamiento requieren de un conjunto de datos lo suficientemente grande y diverso para entrenar los modelos base y el metamodelo de manera efectiva. Si el conjunto de datos es pequeño o carece de diversidad, los ensambles de apilamiento no son beneficiosos y conducen a un sobreajuste.

1.1.23 Validación cruzada

La validación cruzada es una técnica utilizada en el aprendizaje automático para evaluar y validar modelos de manera más precisa [23]. En lugar de dividir los datos en un solo conjunto de entrenamiento y prueba, la validación cruzada divide los datos en múltiples conjuntos de entrenamiento y prueba, permitiendo que cada

muestra se utilice tanto para entrenamiento como para prueba en diferentes iteraciones.

1.1.24 Voting

Los modelos de aprendizaje por conjuntos basados en votación [24] son enfoques simples donde se entrenan varios modelos con el mismo conjunto de datos y se selecciona la predicción final mediante un sistema de votación. Cada modelo emite su propia predicción para cada caso y la predicción final es determinada por la mayoría de los votos de los modelos. Por ejemplo, es posible entrenar modelos basados en árboles de decisión, regresión logística y k-vecinos. Cada modelo realiza predicciones individuales y la predicción final es aquella que obtiene más votos. En caso de discrepancias entre los modelos, se selecciona la predicción con mayor consenso. Este enfoque permite combinar diferentes algoritmos y aprovechar la diversidad de los modelos para mejorar la precisión y robustez de las predicciones.

Nota: Los modelos de aprendizaje de votación se descartaron del presente trabajo a razón de los motivos expuestos a continuación:

- Desequilibrio en la competencia de los modelos base: En un ensamble de votación, todos los modelos base tienen igual peso en la decisión final. Sin embargo, si algunos modelos base son inherentemente más precisos o confiables que otros, darles el mismo peso no es apropiado. En lugar de utilizar un enfoque de votación igualitario, es más efectivo asignar pesos diferentes a los modelos base en función de su desempeño histórico o su nivel de confianza.
- Costo computacional y complejidad: Aunque los ensambles de votación son relativamente simples de implementar, suelen requerir más recursos computacionales en comparación con un solo modelo. Si el tiempo o los recursos computacionales son limitados, resulta más eficiente utilizar un solo modelo bien calibrado en lugar de un ensamble de votación.

1.1.25 API REST

Un API (*Application Programming Interface*, Interfaz de Programación de Aplicaciones) es un conjunto de especificaciones y reglas que facilitan la creación y combinación de aplicaciones [25]. En resumen, un API permite la interacción con otros sistemas para obtener información a través de solicitudes. Actúa como un mediador entre los clientes y los recursos que se desean obtener.

REST es un conjunto de restricciones arquitecturales y no un protocolo específico. Permite la transferencia de representaciones del estado de los recursos al cliente o *endpoint* mediante HTTP, en formatos como JSON (*JavaScript Object Notation*), HTML plano, XML (*Extensible Markup Language*) y otros.

Además, los parámetros y encabezados son importantes en los métodos HTTP, ya que contienen información relevante como datos de identificación, *cookies* y *tokens*, entre otros.

1.1.26 Python

Python es un lenguaje de programación interpretado de propósito general, que cuenta con una sintaxis clara y un enfoque orientado a objetos [26]. Se conoce por ser fácil de aprender y usar, lo que lo hace popular tanto para principiantes como para programadores experimentados. La amplia biblioteca estándar es una de sus características distintivas, y se encuentran módulos para casi cualquier tarea imaginable. Es versátil y se utiliza en muchos entornos diferentes, incluyendo aplicaciones web, ciencia de datos, inteligencia artificial y automatización de tareas. Python es de código abierto y cuenta con una gran comunidad de desarrolladores y usuarios que contribuyen a su desarrollo y mejora continua. En resumen, Python es un lenguaje de programación popular y eficiente con una comunidad activa y comprometida de desarrolladores y usuarios.

1.1.27 Flask

Flask es un micro marco de trabajo de Python que permite construir aplicaciones web con una cantidad mínima de código, siendo descrito como "simple, fácil de usar y extensible". Su enfoque minimalista lo hace altamente personalizable y se integra bien con otras bibliotecas y herramientas de Python. Además, Flask tiene una curva

de aprendizaje relativamente baja y enfatiza la importancia de la legibilidad del código para facilitar la colaboración en el desarrollo de proyectos. En resumen, Flask es un micro marco de trabajo de Python simple, fácil de usar y altamente personalizable para la creación de aplicaciones web [27].

1.1.28 Flask-RESTful

Flask-RESTful [28] es un micro marco de trabajo para Flask, un marco de desarrollo web en Python, que facilita la creación de servicios web RESTful. Proporciona herramientas y funcionalidades para implementar API REST de manera sencilla y eficiente.

Esta extensión permite definir recursos y rutas utilizando clases y métodos en Python. Los recursos son representaciones de objetos que están disponibles para acceder, crear, actualizar o eliminar a través de las operaciones HTTP estándar, como GET, POST, PUT y DELETE.

1.1.29 Joblib

Joblib es una biblioteca de Python que proporciona herramientas para el almacenamiento eficiente de objetos de Python en disco y su posterior recuperación [29]. Está diseñada utilizarse en tareas de procesamiento paralelo y distribuido, y es especialmente útil para el procesamiento de datos científicos y el aprendizaje automático.

Esta biblioteca ofrece una función llamada "dump" que permite almacenar objetos de Python en disco de manera eficiente. Esto es útil cuando se trabaja con grandes conjuntos de datos o modelos de aprendizaje automático que ocupan mucha memoria. La función "load" se utiliza para recuperar los objetos almacenados y cargarlos en la memoria.

1.1.30 Scikit-learn

Scikit-learn es una biblioteca de aprendizaje automático de código abierto para Python que proporciona herramientas eficientes y fáciles de usar para análisis de datos y modelado predictivo. Según la documentación oficial de scikit-learn [30],

esta biblioteca ofrece una amplia gama de algoritmos de aprendizaje automático, incluyendo clasificación, regresión, agrupamiento y reducción de dimensionalidad.

1.1.31 Angular

Angular es un marco de trabajo de JavaScript diseñado por Google para aplicaciones web escalables y dinámicas, que combina las mejores características de bibliotecas populares de una manera fácil de usar. Una de las características distintivas de Angular es su enfoque en la arquitectura MVC, que ayuda a mantener el código organizado. Además, Angular maneja grandes aplicaciones web gracias a su sistema de detección de cambios y su capacidad para trabajar con módulos y componentes. También cuenta con un enfoque en la reutilización de código y una amplia gama de herramientas y bibliotecas adicionales para ampliar su funcionalidad [31].

1.1.32 MongoDB

MongoDB [32] es una base de datos de documentos que ofrece escalabilidad y flexibilidad, y un modelo de consultas e indexación avanzado; que se encuentra disponible en la nube o como servidor.

El modelo de documentos de MongoDB resulta muy fácil de aprender y usar, y proporciona a los desarrolladores todas las funcionalidades que necesitan para satisfacer los requisitos más complejos a cualquier escala. Se proveen controladores para más de diez lenguajes, y la comunidad desarrolló varias decenas más.

1.1.33 Docker

Docker es una plataforma de código abierto que permite a los desarrolladores y administradores de sistemas construir, empaquetar y distribuir aplicaciones de manera eficiente y portátil. Según el sitio web oficial de Docker [1], esta tecnología utiliza contenedores para encapsular aplicaciones y sus dependencias en entornos aislados, lo que facilita su implementación y ejecución en diferentes sistemas operativos y entornos de ejecución.

Docker se basa en el concepto de contenedores, que son entornos ligeros y portátiles que contienen todo lo necesario para ejecutar una aplicación, incluyendo el código, las bibliotecas y las configuraciones. Estos contenedores son independientes y aislados entre sí, lo que permite ejecutar aplicaciones de manera consistente en diferentes entornos.

La plataforma de Docker ofrece una serie de características y herramientas que facilitan la gestión de contenedores, como la creación de imágenes, la gestión de volúmenes de datos y la orquestación de múltiples contenedores. Además, Docker cuenta con un ecosistema de aplicaciones y servicios complementarios que amplían sus funcionalidades y permiten integrarse con otras herramientas de desarrollo y despliegue.

1.1.34 RandomizedSearchCV

El módulo *RandomizedSearchCV* [33] de la biblioteca scikit-learn es una herramienta para la búsqueda aleatoria de hiperparámetros en modelos de aprendizaje automático. Proporciona una forma eficiente de explorar diferentes combinaciones de valores de hiper-parámetros en un espacio definido, mejorando así la capacidad de encontrar la configuración óptima para un modelo.

En lugar de probar todas las combinaciones posibles, es computacionalmente costoso, RandomizedSearchCV selecciona aleatoriamente un subconjunto de combinaciones y las evalúa mediante validación cruzada. Esto permite una búsqueda más eficiente en el espacio de hiper-parámetros, manteniendo un buen rendimiento del modelo.

1.1.35 JWT

JWT [34] es un estándar de seguridad que permite la autenticación y autorización en aplicaciones web mediante el uso de tokens JSON firmados digitalmente. Proporciona una forma eficiente y segura de transmitir información entre el cliente y el servidor. JWT se basa en una estructura de tres partes: encabezado, carga útil y firma. El encabezado contiene información sobre el algoritmo de firma utilizado, mientras que la carga útil almacena los datos relevantes, como el identificador del

usuario y los roles. La firma asegura que el *token* no se modificó durante la transmisión y que solo sea verificado por el servidor.

1.1.36 PyJWT

La biblioteca "pyjwt" es una herramienta útil para trabajar con *tokens* JWT en aplicaciones Python, proporcionando funciones simples y eficientes para su generación, codificación y verificación [35].

1.1.37 Pandas

Pandas es una biblioteca de Python diseñada para el análisis y manipulación de datos de forma eficiente [36]. Proporciona estructuras de datos flexibles y herramientas poderosas que facilitan la limpieza, transformación y exploración de datos.

Pandas trabaja con datos tabulares, como hojas de cálculo y bases de datos, utilizando objetos como *DataFrames* y Series. Estos objetos permiten realizar operaciones avanzadas de indexación, filtrado y agregación de datos. Además, Pandas ofrece funcionalidades para importar y exportar datos en varios formatos, como CSV, Excel, SQL, entre otros. También incluye herramientas para el manejo de fechas y tiempos, manipulación de cadenas de texto y cálculos estadísticos.

1.1.38 NumPy

NumPy [37] es una biblioteca de Python utilizada para realizar operaciones numéricas y científicas eficientes. Proporciona estructuras de datos de alto rendimiento, como arreglos multidimensionales, que permiten realizar cálculos rápidos y optimizados en grandes conjuntos de datos.

NumPy permite realizar operaciones matemáticas y estadísticas avanzadas, como álgebra lineal, transformaciones de Fourier, generación de números aleatorios y manipulación de matrices. Esta biblioteca también ofrece funcionalidades para trabajar con datos faltantes, ordenar y filtrar arreglos, y realizar operaciones *element-wise* de forma eficiente. Y también permite la integración con otras bibliotecas científicas y de análisis de datos en Python.

1.1.39 ReportLab

ReportLab es una biblioteca de Python utilizada para generar documentos PDF de forma dinámica [38]. Permite la creación de PDFs desde cero o la incorporación de contenido dinámico como texto formateado, gráficos, tablas y más. Con características como el manejo de fuentes, soporte Unicode y generación de códigos de barras, es comúnmente utilizado en aplicaciones que requieren la creación de informes, facturas y otros documentos PDF personalizados.

1.1.40 OpenPyXL

Openpyxl [39] es una biblioteca de Python especializada en la manipulación de archivos Excel (.xlsx). Ofrece funcionalidades para la inserción de datos, formato de celdas, manejo de gráficos, estilos y otras operaciones comunes en hojas de cálculo Excel. Su documentación integral proporciona ejemplos y detalles para facilitar su uso.

1.2 Situación tecnológica, económica y operativa de la empresa

El Instituto Tecnológico de Orizaba es una institución pública, que pertenece al Tecnológico Nacional de México. Se encuentra ubicado en Avenida Oriente 9 Núm. 852, Colonia Emiliano Zapata, C.P. 94320, Orizaba, Veracruz. Esta institución a nivel licenciatura ofrece carreras como es la licenciatura en sistemas computacionales una de las 8 carreras ofertadas; y a nivel posgrado cuenta con 5 maestrías tal es el caso de la maestría en sistemas computacionales y 1 doctorado.

1.3 Planteamiento del problema

El daño hepático causado por obesidad (hígado graso), o hepatotoxicidad generada por el consumo de medicamentos, drogas o incluso remedios caseros, por mencionar algunos casos, se está convirtiendo en un problema principal en la salud en México, principalmente, la enfermedad por hígado graso no alcohólico (EHGNA) que afecta a un tercio de la población mundial, siendo México uno de los países cuya población reúne varios factores de riesgo para esta enfermedad.

Por tal motivo, en este proyecto se desarrolla un módulo basado en algoritmos ensambles de aprendizaje automático que permita la detección temprana de desórdenes hepáticos, por ejemplo: cirrosis, daño hepático en general, hígado

graso no alcohólico o cualquier otro problema de salud relacionado con el hígado a fin de tomar las medidas preventivas que resulten necesarias.

Los ensambles de aprendizaje automático son algoritmos integrados por algoritmos más simples (tradicionales), que permiten incrementar los porcentajes de sensibilidad y especificidad al momento de clasificar y detectar. Por lo cual el uso de algoritmos de ensambles de aprendizaje automático permitirá la detección pronta de desórdenes hepáticos a partir del análisis de biomarcadores. Para esto, se realiza un análisis de los algoritmos de ensamble que permitan obtener los mejores resultados, con base a lo anterior, se selecciona un algoritmo para implementarlo en el módulo. Finalmente, el módulo desarrollado se utiliza para evaluar el método propuesto en este proyecto.

1.4 Objetivo general y específico

A continuación, se presenta el objetivo general y los objetivos específicos.

1.4.1 Objetivo general

Desarrollar un módulo basado en algoritmos de ensambles de aprendizaje automático que permita la detección temprana de desórdenes hepáticos tales como: cirrosis, hepatitis y la probabilidad de complicaciones en un paciente diagnosticado con hígado graso no-alcohólico, a partir del análisis biomarcadores.

1.4.2 Objetivos específicos

1. Analizar trabajos relacionados para establecer la principal aportación del proyecto a través de la identificación de diferencias entre iniciativas existentes y el tema de tesis propuesto.
2. Analizar los tipos de algoritmos de aprendizaje de ensamble de las clasificaciones *bagging* y *boosting* para identificar los más convenientes para la detección temprana de desórdenes hepáticos.
3. Identificar los biomarcadores preponderantes para la detección temprana de desórdenes hepáticos.
4. Diseñar el modelo de entrenamiento basado en el algoritmo de ensamble factible para la detección temprana de desórdenes hepáticos.

5. Entrenar el modelo del algoritmo de ensamble para la detección de desórdenes hepáticos.
6. Implementar las interfaces Web y repositorio de información con los que se integrará el algoritmo de ensamble implementado para la gestión de la información.
7. Evaluar al menos un caso de estudio como prueba de concepto que permita describir los resultados y conclusiones obtenidas.

1.5 Justificación

México es uno de los países cuya población reúne varios factores de riesgo por el daño hepático causado por obesidad (hígado graso), o hepatotoxicidad y su prevalencia supera el 50% [3]; es por eso por lo que el panorama a mediano plazo resulta ser pesimista si no se toman acciones inmediatas para contrarrestar lo que ya se considera un problema de salud nacional.

Por medio de este trabajo, el desarrollar un módulo basado en algoritmos de ensambles de aprendizaje automático para la detección temprana de desórdenes hepáticos, permite conocer las probabilidades de que una persona contraiga alguna enfermedad como: cirrosis, daño hepático en general, hígado graso no alcohólico o cualquier otro problema de salud relacionada con el hígado a fin de tomar las medidas preventivas que resulten necesarias.

Capítulo 2. Estado de la práctica

Antes de introducirse al tema por completo, se revisó un considerable número de artículos de los cuales se tomó la información más relevante y se revisaron meticulosamente las investigaciones relacionadas con el tema propuesto.

A continuación, se presenta una breve descripción de los más importantes.

2.1 Trabajos relacionados

Santos et al. [40] expuso que el carcino hepatocelular es el sexto tipo de cáncer más diagnosticado y que este representa en más del 90 por ciento el principal tipo de cáncer en el hígado. Se desarrollaron en el transcurso de los años métodos de aprendizaje automático para extraer conocimiento de los datos clínicos, pero tales enfoques tienen limitaciones, tales como la presencia de datos faltantes y no consideran la heterogeneidad entre los pacientes. En este trabajó se propuso un nuevo enfoque basado en agrupamiento con sobre muestreo robusto para tratar con conjuntos de datos pequeños o no balanceados. El trabajo concluyó en que este nuevo método, es capaz de predecir la supervivencia de un paciente en rango de un año y se utilizó un conjunto de datos compuesto por 165 pacientes y este se obtuvo del Hospital y Centro Universitario de Coimbra.

En [41] se habló sobre la cirrosis que consiste en la formación de contenido cicatricial en el hígado, por lo cual, las pruebas no invasivas actuales tienen un rendimiento muy limitado para detectar un progreso avanzado de dicha enfermedad. Azhie et al. presentó el uso de los algoritmos de aprendizaje automático para identificar pacientes con cirrosis en estado avanzado. Para entrenar y validar estos métodos se utilizaron 1703 biopsias de la Clínica para el Hígado de Toronto (TLC) del inglés (*Toronto Liver Clinic*). El resultado del estudio consistió en que un algoritmo de conjunto de: *Support Vector Machine, Random Forest, Gradient Boosting*, Regresión Logística y Redes neuronales, logró clasificaciones determinadas del 100% y un puntaje AUROC de 0.870 en el conjunto de validación de TLC, y un AUROC de 0.716 en el conjunto de validación de MUHC. Este algoritmo superó a todos los biomarcadores utilizados rutinariamente y logró un

rendimiento comparable al de los hepatólogos en términos de AUROC y porcentaje de clasificaciones indeterminadas en ambos conjuntos de validación.

Doganer et al. [42] disertó respecto la importancia del uso de un algoritmo de ensamble de aprendizaje automático de rendimiento alto para la detección temprana del diagnóstico y clasificación de los carcinomas de células renales. Para validar cuál de los enfoques de aprendizaje automático obtiene mejores resultados se utilizaron y compararon métodos de *bagging, boosting* y *stacking*, y este último tiene algoritmos tales como: IB1, IBk, Kstar, LWL, REPTree, *Random Forest* y SMO. La conclusión del trabajo demostró que los algoritmos de *stacking* obtuvieron los mejores resultados, específicamente REPTree con la más alta precisión del 86.7 por ciento.

En [43] Ponnaganti y Anitha expusieron que el cáncer de mama se considera como una de las enfermedades más mortales para las mujeres en el mundo y cómo la detección temprana ayuda a salvar vidas. Para esta enfermedad se propuso en este artículo, un algoritmo de ensamble de clasificación *Bagging Weighted Voting (EBWvc)* para permitir la clasificación del cáncer de pecho. Para comprobar la eficacia del algoritmo de ensamble se usó un conjunto de datos del hospital Madison de la Universidad de Wisconsin. El *EBWvc* propuesto demostró una clasificación mejorada con respecto a las técnicas de clasificación existentes.

Verma y Mehta [22] propusieron un nuevo método de algoritmo de ensamble de aprendizaje automático llamado “BBS *method*” el cual se encuentra preparado con *Bagging, Boosting* y *Stacking* con los apropiados clasificadores base. Para la clasificación y prueba de dicho algoritmo de ensamble propuesto, se utilizaron cinco conjuntos de datos del UCI *Machine Learning Repository*. Los experimentos se realizaron utilizando Weka y Eclipse con Java. El resultado de dicha experimentación expuso que el “BBS *method*” obtuvo una mejor precisión comparada a los algoritmos de ensambles medidos individualmente y que es apto para utilizarse en el campo de la bioinformática.

En [44] la enfermedad cardiaca es un problema grave de salud pública, cada año, 17 millones de personas alrededor del planeta sufre por enfermedades

cardiovasculares letales como infartos o accidentes cerebrovasculares. Para el análisis de los biomarcadores se usó un conjunto de datos del UCI *Machine Learning Repository* y a estos se les aplicó tres algoritmos de clasificación los cuales se presentan: *Random Forest* con un algoritmo de ensamble de *bagging*, K-Nearest, *Naive Bayes* utilizando validación cruzada de 10 hojas. El modelo que se sugirió obtuvo un 85.18% de porcentaje de precisión a través del *Random Forest con Bagging.*

Chicco y Jurman [45] declararon la problemática causada principalmente por la enfermedad infecciosa de la hepatitis C, la cual afecta más de 70 millones personas en el mundo y que es posible aprovechar los registros electrónicos de salud (EHR del inglés *Electronic Health Records*). Para aprovechar los registros mencionados con anterioridad en este trabajo se analizaron 540 registros electrónicos y se aplicaron clasificadores de aprendizaje automático tales como: *Random Forests*, Regresión lineal, Árbol de decisión para detectar las variables de diagnóstico más destacables de la hepatitis C. Se probaron estos algoritmos en un conjunto de datos de Lichtinghagen. El resultado final mostró la importancia del uso de los métodos de ensamblaje de aprendizaje automático para la predicción del diagnóstico de la hepatitis C y la cirrosis.

En [46] se hizo hincapié en el hecho, de que el cáncer cervical es uno de los cánceres más comunes entre las mujeres a nivel mundial, el cómo la investigación de esta enfermedad y el pronóstico permite dar aplicaciones clínicas al paciente. En este trabajo, se propuso un modelo computacional que es capaz predecir los diagnósticos de cáncer cervical de una manera eficiente. Para probar este método, se usó un conjunto del repositorio de la *University of California Irvine*. Dicho estudio demostró que, la arquitectura propuesta, aplicada por primera vez en el conjunto de datos, usó sólo 5.6 segundos para su procesamiento con un área bajo la curva del 99.7%, demostrando un rendimiento mejorado respecto a los métodos utilizados en trabajos anteriores.

Khanam y Mondal [47] manifestaron que el cáncer cérvico uterino es una de las formas de más mortales de cáncer y que un diagnóstico temprano tiene mucha

relevancia para prevenirlo. En varios países tienen limitadas las facilidades para obtener diagnósticos de dicha enfermedad. Este trabajó se centró en la aplicación de algoritmos de aprendizaje automático para obtener un diagnóstico del cáncer cervical. Para encontrar el atributo más relevante para determinar se aplicó un árbol extra de clasificación dentro de un conjunto de datos acumulado de un repositorio de la *University of California Irvine*. El resultado mostró que el uso de *stacking* combinado con los algoritmos de *random forest, support vector machine*, clasificador de árbol extra, *XGBoost* y *bagging* obtuvieron la más alta calificación con un 94.4 por ciento de precisión.

Un accidente cerebrovascular ocurre cuando el suministro de sangre que va hacia el cerebro se bloquea, dando paso a una falta de oxígeno a la sangre. En [48] se habló sobre la necesidad de un diagnóstico temprano de un accidente cerebrovascular para afrontar dicha situación emergente de una manera eficiente. Para esta problemática, la incorporación de las técnicas de aprendizaje automático provee decisiones rápidas y precisas. Se propuso un sistema proporcione un pronto diagnóstico de un accidente cerebrovascular mediante el uso de un algoritmo de ensamble de regresión logística homogénea, el cual es un algoritmo lineal que hace uso de la metodología de máxima semejanza. Para validar dicho algoritmo de ensamble se utilizó un conjunto de datos del repositorio Kaggle. Este trabajo comprobó que un algoritmo de regresión logística homogénea obtuvo mayor precisión comparado a la regresión logística simple.

En [49] Buyrukoglu explicó que la dificultad de obtener la detección temprana del Alzheimer da como resultado, un deterioro de la memoria. Por consecuente, la temprana detección de dicha enfermedad es benéfica para las zonas del cerebro del cerebro que controlan funciones vitales. Diversos enfoques de detección del Alzheimer se desarrollaron con métodos simples de aprendizaje automático, en este estudio se intentó mejorar esos métodos previamente desarrollados, añadiendo algoritmos de ensambles de *bagging, boosting* y *stacking*. Para comprobar las mejoras de clasificación del nuevo enfoque, se usó un conjunto de datos del *Alzheimer's Disease Neuroimaging Intiative.* Dicho estudio reveló que la

clasificación incrementó entre el 3.2% y el 7.2% mediante el algoritmo de ensamble AdaBoost utilizado frente a los enfoques simples de aprendizaje automático.

En el mundo actual, la detección automática de la hepatitis es un gran problema entre el personal médico. La hepatitis es una enfermedad en la cual las células del hígado se inflaman y esta es provocada por varios tipos de virus. En [50] Singh et al. propusieron un enfoque híbrido inteligente para el diagnóstico de dicha enfermedad hepática, con el propósito de reducir el tiempo de evaluación. Para obtener dicho diagnostico se realizó una combinación del algoritmo de agrupamiento k-means y el aprendizaje por ensamble impulsado mejorado. Para probar el funcionamiento del sistema híbrido inteligente se utilizó un conjunto de datos del repositorio de *machine learning* de la *University of California*. El resultado comprobó en términos de precisión, que el enfoque híbrido mejorado logró mejores resultados de predicción que otros algoritmos individuales e integrados.

En [51] los infartos al miocardio se provocan por la destrucción del tejido del corazón ocasionado por la obstrucción del suministro de sangre al músculo de este. En este estudio se comparó el rendimiento, en el uso de dos algoritmos de ensamble automático: *bagging y boosting* con 5 clasificadores base, los cuales son: *Support Vector Machine, K-Nearest Neightbor, Naive Bayes, Decision Tree y Random Forest* para predecir los síntomas de un infarto al miocardio en una etapa temprana. Para obtener la mejor precisión se utilizó un conjunto de datos del *National Hear Foundation of Bangladesh*. El experimento arrojó que el algoritmo de *bagging* en conjunto con *Random Forest* obtuvo la mayor precisión con 96.50 de porcentaje.

En [9] Taser consideró que con la pronta detección de la diabetes es posible evitar posibles complicaciones de alto riesgo. Para resolver tal problemática, se realizó una comparación con enfoque de algoritmos de ensamble automático: *bagging* y *boosting* y seis clasificadores basados en árboles de decisión (DTB), incluyendo sus implementaciones individuales también. Para obtener cuál algoritmo tiene el mayor porcentaje de precisión se utilizó un conjunto de datos de UCI *Machine Learning Repository*, el cual consiste en reportes de síntomas relacionados a la diabetes, dicho conjunto incluyó 17 atributos y 520 registros. El resultado de dicha

comparación demostró que: el clasificador NBTree mediante un algoritmo de *boosting* obtuvo la precisión más alta con un porcentaje del 98.65%.

Lin et al. [52] sugirió sobre cómo la relación del resultado funcional y la función cognitiva en la esquizofrenia son medidas por los síntomas clínicos. El resultado funcional es evaluado por la Escala de Calidad de Vida (QLS) del inglés (*Quality Life Scale*) y Evaluación Global del Funcionamiento (GAF) del inglés (*Global Assessment of Functioning*). En este estudio se propuso que para determinar el resultado funcional del QLS y el GAF, un marco de trabajo con un algoritmo de ensamble *bagging* y se comparó con otros algoritmos de clasificación tales como: redes neuronales*, support vector machine, random forests* y regresión lineal. Para realizar dicha validación se utilizó el registro de 302 pacientes con esquizofrenia provenientes de Taiwán. El estudio obtuvo como resultado mediante validación cruzada de 10 hojas para predecir la QLS y la GAF, y el algoritmo *bagging* con selección de características obtuvo el mejor resultado de predicción respecto a los demás enfoques.

En [53] se habló respecto a la importancia de la digestión de la comida y como es importante para coordinar las actividades del organismo, de tal manera que, las sustancias tóxicas que se desechan del cuerpo mediante una apropiada digestión y sus enzimas, y como el hígado está directamente relacionado con el proceso antes mencionado. En este trabajo se intentó predecir las enfermedades del hígado mediante el análisis de un conjunto de datos, obtenido del UCI *Machine Repository.* Para entrenar este modelo se utilizaron los siguientes enfoques de algoritmos de ensambles de aprendizaje automático: *Gradient Boost, Extra Trees y Random Forest* mediante la implementación de script de Python utilizando el IDE de Anaconda Spyder. El estudio mostró que clasificador de *Random Forest* obtuvo la mayor precisión con un porcentaje del 74%.

Yadav y Pal [54] expusieron en su artículo la problemática que causa la diabetes mellitus, en la cual el organismo es incapaz de administrar la insulina. En este estudio experimental se aplicaron tres algoritmos de clasificación basados en reglas tales como: Tablas de decisión, OneR y JRIP y posteriormente se administraron por

métodos de algoritmos de ensamble *bagging y boosting*. Para evaluar cual algoritmo tiene el mejor rendimiento se utilizó un conjunto de datos del UCI *Learning Machine Repository* con datos de mujeres diabéticas. El articuló demostró que el algoritmo de ensamble de *bagging* obtuvo 98% de precisión a diferencia de los demás enfoques que obtuvieron muchísimo menos.

2.2 Análisis comparativo

La Tabla 2.1 presenta un análisis comparativo sobre los artículos relacionados con este proyecto, así como una breve descripción acerca de la problemática, contribución, tecnologías utilizadas, resultados obtenidos y el estado del artículo.

Tabla 1.1 Análisis comparativo de los trabajos relacionados

Artículo	Situación problemática	Contribución	Tecnologías	Resultados	Estatus
Santos et al. [40]	Limitaciones al momento de procesar datos faltantes o no heterogéneos en los algoritmos de aprendizaje automático para el diagnóstico del carcinoma hepatocelular.	Integración de un nuevo enfoque basado en agrupamiento con un sobre muestreo.	• K-Means	El nuevo método desarrollado permitió predecir la supervivencia de un paciente en el rango de un año.	Terminado
Azhie et al. [41]	El rendimiento limitado de las pruebas actuales no invasivas para la detección de la cirrosis en el hígado.	Uso de algoritmos de aprendizaje automático para identificar cirrosis en estado avanzado de un paciente,	• Support Vector Machine • Random Forest • Gradient Boosting • Regresión Logística • Redes Neuronales	En el estudio, fue posible detectar la cirrosis en estado avanzado con un rendimiento superior gracias a los algoritmos de aprendizaje automático.	Terminado
Doganer et al. [42]	Falta de detección y clasificación precisa de los carcinomas de células renales.	Validación de diferentes algoritmos de ensambles de bagging, boosting, stacking para ver cual obtuvo los mejores resultados.	• IB1 • IBk • Kstar • REPTTree • Random Forest • SMO	Los carcinomas detectados con un alto desempeño utilizando los algoritmos de ensambles, el que obtuvo los mejores resultados fue REPTTree.	Terminado
Ponnaganti y Anitha [43]	Carencia de detección temprana del	Implementación de un algoritmo de ensamble de	• Ensamble Bagging Weighted	El enfoque propuesto demostró un rango de	Propuesta

Artículo	Situación problemática	Contribución	Tecnologías	Resultados	Estatus
	cáncer de mama en las mujeres.	aprendizaje automático para clasificar el cáncer de pecho.	Voting (EBWvc)	clasificación superior a las técnicas actuales.	
Verma y Mehta [22]	Necesidad de desarrollar un algoritmo de ensamble de aprendizaje automático para mejorar algoritmos clasificadores base.	Creación de un enfoque nuevo preparado con algoritmos de ensamble de aprendizaje automático.	• Bagging • Boosting • Stacking • Eclipse JDK • JAVA • Weka	El método propuesto obtuvo una mejor precisión comparándola con sus contrapartes individuales.	Terminado
Rahman y Mahmood [44]	Escases de pronta detección de una enfermedad cardiaca.	Comparación de tres algoritmos de clasificación utilizando biomarcadores.	• Random Forest con ensamble de bagging. • K-Nearest • Naive Bayes	El algoritmo de Random Forest con bagging obtuvo la más alta precisión para predecir enfermedades coronarias con un porcentaje del 85.18%.	Propuesta
Chicco y Jurman [45]	Falta de diagnóstico temprano de la enfermedad infecciosa de la hepatitis C.	Utilización de los registros de salud electrónicos mediante clasificadores de aprendizaje automático para un diagnóstico pronto de la enfermedad.	• Random Forest • Regresión lineal • Árbol de decisión	El enfoque aplicado a los registros de salud electrónicos permitió clasificar controles de salud para pacientes con hepatitis C o enfermedades graves como la cirrosis en unos cuantos minutos.	Propuesta
Gupta y Gupta [46]	Carencia de diagnóstico pronto del cáncer cervical.	Se propuso un modelo computacional mediante una arquitectura de ensamble de stacking.	• K-Nearest Neighbors • Random Over Sampling • Extremely Randomized Trees • Random Forest	La arquitectura presentada por primera vez sobre un conjunto de datos demoró 5.6 segundos para su procesamiento con un área bajo la curva del 99.7%, con un rendimiento mayor a los enfoques de trabajos anteriores.	Propuesta
Khanam y Mondal [47]	Falta de predicción temprana del cáncer cervicouterino a	Aplicación de algoritmos de aprendizaje automático para obtener el	Algoritmo de ensamble de stacking combinado con: • Random Forest	El algoritmo de ensamble de stacking en combinación con Random Forest.	Propuesta

Artículo	Situación problemática	Contribución	Tecnologías	Resultados	Estatus
	causa de las limitadas facilidades en algunos países.	atributo más importante en la detección de esta mortal enfermedad.	• Support Vector Machine • Extra Tree Classifier • XGBoost • Bagging	SVM, XGBoost y el clasificador de bagging obtuvo el 94.4% de precisión y que tiene el potencial de diagnosticar cáncer cervicouterino.	
Anisha y Saranya [48]	Necesidad de tomar decisiones rápidas y precisas en un accidente cerebrovascular.	Implementación de un sistema que proporcione un pronto diagnóstico mediante el uso de algoritmos de ensambles de aprendizaje automático.	• Regresión lineal homogénea.	El sistema propuesto obtuvo errores reducidos en la predicción y una precisión del 91% frente a la regresión logística simple para el diagnóstico de accidentes cerebrovasculares.	Propuesta
Buyrukoglu[49]	Escases de una detección pronta del Alzheimer para evitar degradación de la memoria.	Mejorar los enfoques ya existentes de los algoritmos de aprendizaje automático para la detección de dicha enfermedad, pero implementado los algoritmos de ensambles.	• Bagging • AdaBoost (Boosting) • Stacking	El estudio reveló que se obtuvo un incremento entre el 3.2% y 7.2% respecto a los métodos simples de aprendizaje automático mediante AdaBoost.	Terminado
Singh et al. [50]	Menester de acelerar el tiempo de evaluación y detección de la hepatitis causada por diferentes cepas de virus.	Se presentó un método híbrido inteligente para el diagnóstico temprano de la enfermedad hepática.	• K-Means • Aprendizaje por ensamble impulsado mejorado	El enfoque híbrido mejorado mostró resultados superiores a otros algoritmos integrados o individuales propuestos con anterioridad.	Terminado
Hakim et al. [51]	Falta de detección temprana de los infartos al miocardio.	Se comparó el rendimiento de cinco clasificadores bases dirigidos mediante dos algoritmos de ensamble automático.	• Bagging • Boosting • Support Vector Machine • K-Nearest Neighbor • Naive Bayes • Árbol de decisión • Random Forest	El algoritmo de ensamble de bagging en conjunto con el clasificador Random Forest obtuvo la más alta calificación de precisión con 96.50%.	Terminado
Taser [9]	Complicaciones provocadas por	Se realizó una comparación	• C4.5	El clasificador NBTree en	Terminado

Artículo	Situación problemática	Contribución	Tecnologías	Resultados	Estatus
	la diabetes detectada a destiempo.	utilizando enfoques de algoritmos de ensamble automático de *boosting* y *bagging* y seis clasificadores base.	• Random Tree • REPTree • NBTree • Decision Stump • Hoeffding Tree	conjunto con el algoritmo de boosting obtuvo una precisión del 98.65%, posicionándola como la más alta respecto a los demás clasificadores.	
Lin et al. [52]	Falta de facilidad para determinar la Escala de Calidad de Vida (QLS) y Evaluación Global del Funcionamiento (GAF) en pacientes con esquizofrenia.	Se desarrolló un marco de trabajo utilizando algoritmos de ensambles de bagging junto a otros algoritmos clasificadores para clasificar las escalas mencionadas anteriormente.	• Redes Neuronales • Support Vector Machine • Random Forest • Regresión lineal • Ensamble de bagging • Ensamble de bagging con selección de características	El algoritmo de ensamble de bagging con selección de características obtuvo el mejor resultado de predicción a diferencia de los demás enfoques propuestos.	Terminado
Shyamala Devi et al. [53]	Necesidad de predecir las enfermedades del hígado.	Predicción de enfermedad hepáticas mediante la comparación de distintos algoritmos clasificadores.	• Gradient Boost • Extra Trees • Random Forest • Anaconda Spyder IDE • Python	El clasificador Random Forest se llevó el puntaje más alto, con un 74% de precisión.	Terminado
Yadav y Pal [54]	Falta de predicción de la diabetes mellitus.	Comparación de algoritmos basados en reglas implementados con ensambles de *bagging* y *boosting*.	• Tablas de decisión • OneR • JRIP	El trabajo concluyó en que el ensamble de *bagging* consiguió un 98% de precisión respecto a los demás métodos comparados.	Propuesta

Para el contexto del proyecto actual, se observa que los diferentes autores utilizaron algoritmos de ensambles de aprendizaje automático para dar solución a sus problemáticas de salud pública y que perfectamente encaja en el desarrollo del módulo para la pronta detección de desórdenes hepáticos a partir de biomarcadores.

Capítulo 3. Aplicación de la metodología

En este capítulo se presenta la metodología de Kanban, de la cual se usa para el presente trabajo de tesis, con la intención de dar solución a la problemática planteada y a los objetivos definidos para el desarrollo del módulo de detección temprana de desórdenes hepáticos utilizando métodos de algoritmos de ensambles de aprendizaje automático.

El uso de Kanban en este proyecto de tesis se justifica por varias razones:

- En primer lugar, Kanban proporciona una metodología visual y transparente para la gestión de tareas y el seguimiento del progreso. Permite a todo el equipo tener una visión clara de las tareas pendientes, en progreso y completadas, lo que facilita la comunicación y la colaboración entre los miembros del equipo.
- Además, Kanban facilita la identificación y resolución rápida de problemas o cuellos de botella en el proceso de desarrollo de software. Al visualizar el flujo de trabajo, se detectan fácilmente las áreas problemáticas y tomar medidas correctivas de manera oportuna.
- Otra ventaja de utilizar Kanban es la capacidad de adaptarse y gestionar cambios de manera efectiva. Como el entorno de desarrollo de software es propenso a cambios y requisitos fluctuantes, Kanban permite una fácil reconfiguración de las prioridades y la asignación de recursos en función de las necesidades cambiantes del proyecto.

Puesto que Kanban no tiene fases definidas de manera predeterminada se utilizaron las 4 fases propuestas por el método científico (Análisis, Diseño, Implementación y Pruebas).

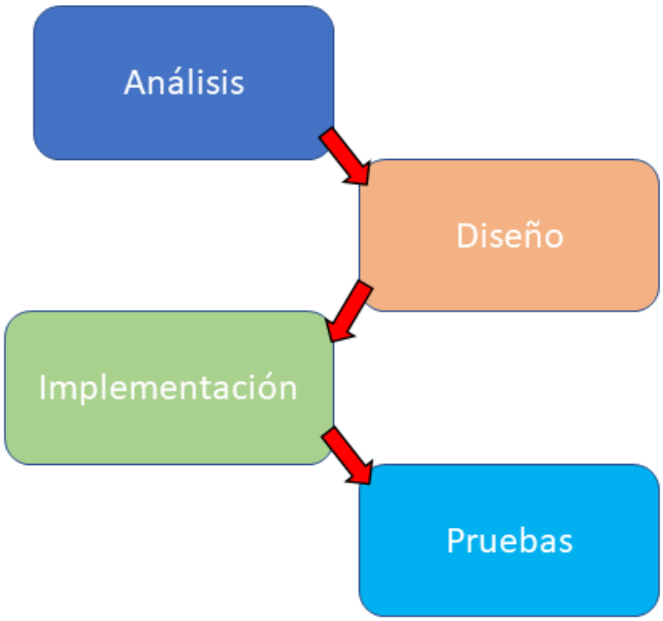


Figura 3.1 Fases del método científico

3.1 Análisis

En esta sección se definen las acciones preliminares requeridas para la realización del presente trabajo, se muestran en formato de “tarjetas Kanban”, las cuales en otras metodologías su equivalente son las “historias de usuario”.

3.1.1 Conjuntos de datos para la identificación de desórdenes hepáticos

En esta sección se describirán cuatro conjuntos de datos que se utilizaron para detección de desórdenes hepáticos mediante la ejecución de este módulo.

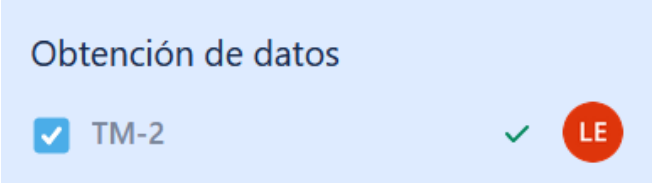


Figura 3.2 tarjeta Kanban “Obtención de datos”

En los dos repositorios más grandes para *Machine learning* (Kaggle y UCI) se consiguieron los siguientes conjuntos de datos.

- *Cirrhosis Prediction Dataset*: [55] contiene información sobre pacientes con cirrosis hepática crónica y no crónica. Incluye 27 variables de entrada que incluyen edad, sexo, síntomas, signos de laboratorio, hallazgos histopatológicos y diagnósticos, así como una variable de salida que indica si el paciente tiene cirrosis crónica o no. El objetivo del conjunto de datos (*dataset*), es predecir si un paciente tiene cirrosis crónica o no en función de las variables de entrada. El conjunto de datos contiene 386 casos, 186 de los cuales son casos positivos de cirrosis crónica. El *dataset* se encuentra en formato CSV y está disponible para descargar en Kaggle.
- *Indian Liver Patient Dataset* (ILPD): ILPD [56] es un conjunto de datos médicos que contiene información sobre pacientes indios con y sin enfermedades hepáticas. El conjunto de datos consta de 583 instancias con 10 atributos, incluyendo información demográfica del paciente, resultados de pruebas de sangre y diagnósticos de enfermedades hepáticas. El objetivo del conjunto de datos es predecir si un paciente tiene una enfermedad hepática o no, lo que lo convierte en un conjunto de datos útil para tareas de clasificación binaria en el campo de la salud.
- *Liver Disorders Data Set*: [57] este conjunto de datos proveniente del repositorio UCI *Machine Learning* contiene los resultados de pruebas de laboratorio de pacientes con trastornos hepáticos, como hepatitis viral, cirrosis, hemocromatosis y otros. El conjunto de datos consta de 345 instancias y 6 atributos, incluyendo el volumen corpuscular medio, la fosfatasa alcalina, la alanina aminotransferasa y el aspartato aminotransferasa. El objetivo del conjunto de datos es predecir si un paciente tiene o no un trastorno hepático basado en los valores de los atributos.
- *Non-alcohol fatty liver disease* (NAFLD): NAFLD [58] alojado en Kaggle contiene información sobre pacientes diagnosticados con hígado graso no alcohólico en un hospital universitario de Corea del Sur. Incluye datos demográficos y clínicos de 17,549 pacientes, como edad, género e índice de masa corporal. El objetivo del conjunto de datos es ayudar a predecir la probabilidad de complicaciones por la contracción de esta enfermedad.

La Tabla 3.1 muestra una comparación breve entre los cuatro conjuntos de datos mencionados anteriormente, haciendo hincapié en el número de instancias y los atributos.

Tabla 3.1 Conjuntos de datos utilizados para este módulo

Nombre	**Repositorio**	**Instancias**	**Atributos**
Cirrhosis Prediction Dataset [55]	Kaggle	424	19
ILPD (Indian Liver Patient Dataset) [56]	UCI	583	10
BUPA Liver Disorders Dataset [57]	UCI	345	6
Non-alcohol fatty liver disease [58]	Kaggle	17,549	10

En la Tabla 3.2 se exponen los resultados de análisis de las columnas de objetivo de los conjuntos de datos.r

Tabla 3.2 Objetivos de la predicción

Conjunto de datos	**Predicción**	**Clase 0:**	**Clase: 1**
Cirrhosis Prediction Dataset	Enfermedad de cirrosis	No tiene	Sí tiene
ILPD (Indian Liver Patient Dataset)	Desórdenes hepáticos	No tiene	Sí tiene
BUPA Liver Disorders Dataset	Desórdenes hepáticos	No tiene	Sí tiene
Non-alcohol fatty liver disease	Probabilidad de complicaciones al contraer EHGNA	No tiene	Sí tiene

3.1.2 Preprocesamiento de los conjuntos de datos

En esta parte del capítulo se exhibe la forma en la cual se preparó a los conjuntos de datos para utilizarse tanto para entrenamiento como prueba en los algoritmos de métodos de ensamble de aprendizaje automático.

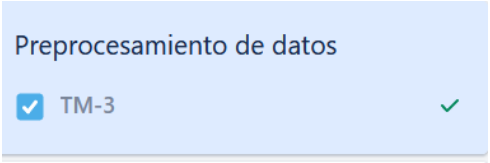


Figura 3.3 tarjeta Kanban "Preprocesamiento de datos"

Una vez obtenidos los *datasets*, en Python estos se cargan en memoria mediante la biblioteca DataFrame del paquete Pandas y de manera individual a cada uno se les hizo limpieza y transformaciones a los datos, en el sentido de reemplazar valores de cadena por numéricos, eliminar valores irrelevantes como el identificador único del dato o columnas que simplemente no tienen utilidad, así como especificar la variable objetivo (la clase a predecir).

En el conjunto de datos de *Cirrhosis Prediction Dataset* se hicieron los siguientes ajustes:

- Sustitución de datos nulos por moda y mediana.
- Reemplazar la edad en días por años.
- Utilizar 0 y 1 para hombre y mujer respectivamente en el género del paciente.
- Reemplazar "N" y "Y" por 0 y 1 respectivamente en los biomarcadores: ascitis, hígado agrandado, *spiders* y edemas.
- Utilizar 0 para "*D-penicillamine*" y 1 para placebo en los medicamentos consumidos por el paciente.
- Eliminar las columnas de: id, número de días desde la afectación (esta variable se removió debido a que, en un contexto de predicción, en la mayoría de los casos es difícil o imposible de saber cuándo empezaron las afectaciones) y el estado del paciente (ya que esta variable sólo indica si el pronóstico se encuentra censurado).
- Establecer la columna "Stage" como el objetivo a predecir.

Para el conjunto de datos *Indian Liver Patient Dataset* se realizaron estas modificaciones:

- Sustitución de datos nulos por moda y mediana.
- Añadir las cabeceras de los atributos al .csv.
- Reemplazar “Male” y “Female” por 0 y 1 respectivamente.
- Reemplazar en variable de clase el número 2 (no enfermo) por 0 y 1 como si el paciente tiene algún desorden.
- Establecer “*Liver Disorder*” como variable de clase.

En el conjunto *BUPA Liver Disorders Dataset* se especificaron las siguientes acciones:

- Sustitución de datos nulos por moda y mediana.
- Añadir las cabeceras de los atributos al .csv.
- Reemplazar en variable de clase el número 2 (no enfermo) por 0 y 1 como si el paciente tiene algún desorden.
- Definir “*Liver Patient*” como variable de clase.

A continuación, en la Lista 1, se exhibe el proceso de preprocesamiento de los datos con su respectiva transformación y limpieza de los datos, así como la selección de la variable objetivo.

```
data = pd.read_csv("assets/datasets/BUPA Liver Disorders/bupa.data",
names=['MCV','Alkphos','SGPT','SGOT','Gammagt','Drinks','Liver Disorder'])
self.manejar_datos_nulos(data)
self._datos = data
#Se reemplaza 2 por 1 (Con desorden) y lo demás 0
data['Liver Disorder'] = np.where(data['Liver Disorder'] == 2,1,0)
self.XF = data.drop('Liver Disorder', axis=1)
self.YF = data['Liver Disorder']
self.klass = 'Liver Disorder'
```

Lista 1 Fragmento de código de pre-procesamiento del *dataset* BUPA Liver Disorders Dataset.

Finalmente, en el conjunto *Non-alcohol fatty disease* se hicieron los siguientes arreglos:

- Sustitución de datos nulos por moda y mediana.
- Eliminar columnas de id e id del diagnóstico.
- Asignar "*status*" como variable de predicción.

3.1.3 Identificación de los biomarcadores preponderantes para la identificación de desórdenes hepáticos

En esta sección se muestra cómo se obtuvieron los biomarcadores más destacables y que se utilizan para obtener las predicciones pertinentes de cada uno de los desórdenes previamente presentados. Mediante el uso de un *RandomForest* con un número equiespaciado de árboles generado por NumPy y posteriormente, una búsqueda aleatoria de las mejores columnas (biomarcadores) mediante un *RandomSearchCV* y finalmente este *RandomForest* un *DataFrame* de Pandas con los biomarcadores mejores evaluados y un *random grid* que es necesario para el entrenamiento posterior de los modelos.

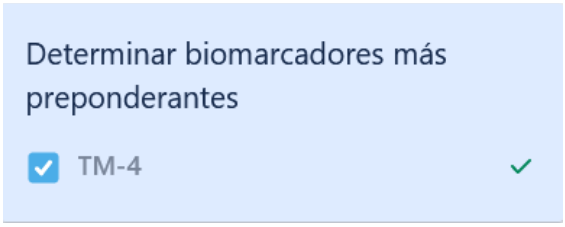


Figura 3.4 tarjeta Kanban "Determinar biomarcadores preponderantes"

A continuación, en la Lista 2 se presenta un fragmento del código que el módulo utiliza para obtener los mejores biomarcadores de un desorden hepático mediante el uso de un *RandomForest*, tal cómo se explica en este punto.

```
rf = RandomForestClassifier(n_estimators=500,max_features=0.25,criterion='entropy',
random_state=42)
rf.fit(XF,yf)
feat_imp = pd.DataFrame(rf.feature_importances_)
feat_imp.index = pd.Series(data.iloc[:,:-1].columns)
feat_imp = (feat_imp*100).copy().sort_values(by=0,ascending=False)
feat_imp = feat_imp.reset_index()
# Número de árboles en el RandomForest
n_estimators = [int(x) for x in np.linspace(start = 200, stop = 2000, num = 10)]
# Número de características en cada split
max_features = ['auto', 'sqrt']
# Profundidad máxima
max_depth = [int(x) for x in np.linspace(100, 500, num = 11)]
# Crear un grid aleatorio
random_grid = {'n_estimators': n_estimators,'max_features': max_features,
'max_depth': max_depth}
# Random search de los parámetros
rfc_random = RandomizedSearchCV(estimator = rf, param_distributions = random_grid,
n_iter = 100, cv = 3, verbose=2, random_state=42, n_jobs = -1)
# Ajustar el modelo
rfc_random.fit(XF, yf)
#Asignando mejores resultados para el modelo
N_trees = rfc_random.best_params_.get("n_estimators")
Max_depth=rfc_random.best_params_.get("max_depth")
Max_features = rfc_random.best_params_.get("max_features")
Random_state = 42
# Colocando los mejores valores para la clasificación de la ejecución anterior
rf_mejorado = RandomForestClassifier(n_estimators=N_trees, max_depth=Max_depth,
```

Lista 2 Fragmento de código de la obtención de los mejores biomarcadores

```
max_features=Max_features,random_state=Random_state)
rf_mejorado.fit(XF,yf)
feat_imp_m = pd.DataFrame(rf_mejorado.feature_importances_)
feat_imp_m.index = pd.Series(data.iloc[:,:-1].columns)
feat_imp_m = (feat_imp_m*100).copy().sort_values(by=0,ascending=False)
feat_imp_m = feat_imp_m.reset_index()
feat_imp_m.columns = ["Feature","Importance_score"]
rg = {'n_estimators': N_trees,'max_features': Max_features,'max_depth': Max_depth,
‘random_state’ : Random_state}
Return (feat_imp_m, rg)
```

Lista 2 Fragmento de código de la obtención de los mejores biomarcadores (continuación)

Cabe resaltar que, para el presente trabajo, únicamente se utilizan los mejores 5 biomarcadores obtenidos en cada uno de los 4 conjuntos de datos, obtenidos mediante el código presentado en la Lista 1 y se presentan a continuación.

3.1.3.1 Mejores biomarcadores para Cirrhosis Prediction Dataset

En la tabla 3.3 se presentan los cinco biomarcadores clasificados por su calificación de importancia, del conjunto de datos Cirrhosis Prediction.

Biomarcador	Calificación de importancia
Protrombina	13.67
Albumina	12.93
Plaquetas	11.93
Edad	9.14
Bilirrubina	8.56

Tabla 3.3 Mejores biomarcadores para el conjunto de datos Cirrhosis Prediction Dataset.

3.1.3.2 Mejores biomarcadores para Indian Patient Liver Dataset

La tabla 3.4 presenta los cinco biomarcadores del *dataset* Indian Patient Liver, obtenidos por su calificación de importancia.

Biomarcador	Calificación de importancia
Fosfatasa alcalina	14.81
Aspartato Aminotransferasa (AST)	13.06
Alanina aminotransferasa (ALT)	13.008
Edad	12.90
Total de proteínas	10.38

Tabla 3.4 Mejores biomarcadores para Indian Patient Liver Dataset

3.1.3.3 Mejores biomarcadores para BUPA Liver Disorders Dataset

En la tabla 3.5 se exponen los cinco biomarcadores de BUPA Liver Disorders Dataset, por su calificación de importancia.

Biomarcador	Calificación de importancia
Gamma glutamil transpeptidasa (Gammagt).	20.84
Alanina aminotransferasa (ALT).	20.47
Aspartato Aminotransferasa (AST).	16.48
Fosfatasa alcalina.	16.42
Volumen corpuscular medio.	13.95

Tabla 3.5 Mejores biomarcadores para BUPA Liver Disorders Dataset.

3.1.3.4 Mejores biomarcadores para Non-alcohol fatty disease

La tabla 3.6 se presentan los cinco biomarcadores mejor calificados del conjunto de datos BUPA Liver Disorders Dataset, por su calificación de importancia.

Biomarcador	Calificación de importancia
Edad	33.34
Índice de masa muscular	25.18
Peso	23.45
Altura	16.66
Género	1.34

Tabla 3.6 Mejores biomarcadores para Non-alcohol fatty disease.

Con los biomarcadores preponderantes detectados, se continúa a la parte del análisis comparativo y entrenamiento de los modelos de los algoritmos de ensamble de aprendizaje automático.

3.1.4 Análisis comparativo de los conjuntos de datos mediante algoritmos de *Machine learning*.

En esta sección se presenta el análisis de los algoritmos de aprendizaje automático en los cuatro conjuntos de datos a los que se hizo referencia en la sección 3.1.2 y utilizando los biomarcadores preponderantes obtenidos en la sección anterior, se utilizaron los siguientes clasificadores base (en modo embolsado "*bagging*"): *LogisticRegression*, SVM (*Support Vector Machine*), KNN (*K-Nearest-Neighbor*), Árbol de decisión y Bosques aleatorios. Así como para el enfoque de *boosting* los mencionados a continuación: AdaBoost, GradientBoosting y LightGBM. Todo esto fue posible gracias a las bibliotecas diseñadas para implementarse con Python de scikit-learn, las cuales se enfocan para utilizarse en aprendizaje automático.

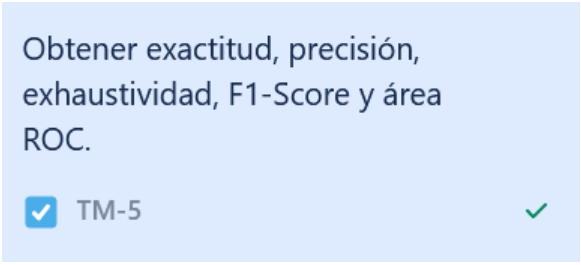


Figura 3.5 tarjeta Kanban "Obtener métricas de los conjuntos de datos"

Realizando el análisis de cinco clasificadores base de *bagging* y los tres definidos para *boosting*; en cada uno de los cuatro conjuntos de datos obtenidos para el presente trabajo, se obtuvieron los siguientes resultados:

3.1.3.1 *Cirrhosis Prediction Dataset* (*Bagging*)

A continuación, en la Tabla 3.7 se detallan las métricas obtenidas para el conjunto de datos *Cirrhosis Prediction Dataset* obtenido del repositorio de *Machine learning* de Kaggle mediante Regresión Logística, SVC, K-NN, Árboles de decisión y Bosques Aleatorios.

Tabla 3.7 Resultados de las métricas con los algoritmos de *Bagging* para el *dataset* Cirrhosis Prediction Dataset.

Algoritmo	Accuracy	Precision	Recall	F1 Score	ROC AUC
LR	69.04%	56.75%	47.72%	51.85%	64.10%
SVC	67.46%	54.83%	38.63%	45.33%	60.78%
KNN	62.69%	45.45%	34.09%	38.96%	56.06%
DT	73.80%	68.96%	45.45%	54.79%	67.23%
RF	68.25%	56.25%	40.90%	47.36%	61.91%

3.1.3.2 *Cirrhosis Prediction Dataset* (*Boosting*)

En la Tabla 3.8 se detallan las métricas obtenidas para el conjunto de datos Cirrhosis Prediction Dataset a través de los algoritmos de *boosting* de AdaBoost, GradientBoosting y LightGBM.

Tabla 3.8 Resultados de las métricas con los algoritmos de *Boosting* para el *dataset* Cirrhosis Prediction Dataset.

Algoritmo	Accuracy	Precision	Recall	F1 Score	ROC AUC
AdaBoost	68.25%	54.16%	59.09%	56.52%	66.13%
GradientBoosting	68.25%	55.55%	45.45%	50%	62.97%
LightGBM	66.66%	52.38%	50%	51.16%	62.80%

3.1.3.3 ILPD (*Indian Liver Patient Dataset*) (*Bagging*)

En seguida, en la Tabla 3.9 se exhiben las métricas alcanzadas para el conjunto de datos ILPD obtenido del repositorio de *Machine learning* de UCI mediante Regresión Logística, SVC, KNN, Árboles de decisión y Bosques Aleatorios.

Tabla 3.9 Resultados de las métricas con los algoritmos de *Bagging* para el *dataset* ILPD.

Algoritmo	Accuracy	Precision	Recall	F1 Score	ROC AUC
LR	72%	74.53%	93.75%	83.04%	53.25%
SVC	73.14%	73.14%	100%	84.48%	50%
KNN	69.14%	78.46%	79.68%	79.06%	60.05%
DT	73.14%	78.32%	87.50%	82.65%	60.77%
RF	77.71%	82.96%	87.50%	85.17%	69.28%

3.1.3.4 ILPD (*Indian Liver Patient Dataset*) (*Boosting*)

Posteriormente, en la Tabla 3.10 se exhiben las métricas adquiridas para el conjunto de datos ILPD a través de los algoritmos de *boosting* de AdaBoost, GradientBoosting y LightGBM.

Tabla 3.10 Resultados de las métricas con los algoritmos de *Boosting* para el *dataset* ILPD.

Algoritmo	Accuracy	Precision	Recall	F1 Score	ROC AUC
AdaBoost	68%	79.03%	76.56%	77.77%	60.62%
GradientBoosting	74.28%	80.74%	85.15%	82.88%	64.91%
LightGBM	75.42%	82.94%	83.59%	83.26%	68.39%

3.1.3.5 *BUPA Liver Disorders Dataset* (*Bagging*)

En esta subsección, en la Tabla 3.11 se especifican los valores de las métricas conseguidas para el conjunto de datos Cirrhosis Prediction Dataset obtenido del repositorio de *Machine learning* de UCI mediante Regresión Logística, SVC, KNN, Árboles de decisión y Bosques Aleatorios.

Tabla 3.11 Resultados de las métricas con los algoritmos de *Bagging* para el *dataset* BUPA Liver Disorders Dataset.

Algoritmo	Accuracy	Precision	Recall	F1 Score	ROC AUC
LR	71.15%	73.43%	78.33%	75.80%	69.84%
SVC	71.15%	74.19%	76.66%	75.40%	70.15%
KNN	69.23%	70.58%	80%	75%	67.27%
DT	69.23%	70%	81.66%	75.38%	66.96%
RF	75.96%	74.64%	88.33%	80.91%	73.71%

3.1.3.6 *BUPA Liver Disorders Dataset* (*Boosting*)

Consecutivamente, en la Tabla 3.12 se especifican los valores de las métricas conseguidas para el conjunto de datos ILPD a través de los algoritmos de *boosting* de AdaBoost, GradientBoosting y LightGBM.

Tabla 3.12 Resultados de las métricas con los algoritmos de *Boosting* para el *dataset* BUPA Liver Disorders Dataset.

Algoritmo	Accuracy	Precision	Recall	F1 Score	ROC AUC
AdaBoost	64.42%	68.25%	71.66%	69.91%	63.10%
GradientBoosting	69.23%	70.58%	80%	75%	67.27%
LightGBM	69.23%	72.58%	75%	73.77%	68.18%

3.1.3.7 *Non-alcohol fatty disease* (*Bagging)*

Antes de finalizar esta comparativa, en la Tabla 3.13 se presentan las métricas calculadas en el conjunto de datos Non-alcohol fatty disease obtenido del repositorio de *Machine learning* de Kaggle mediante Regresión Logística, SVC, KNN, Árboles de decisión y Bosques Aleatorios.

Tabla 3.13 Resultados de las métricas con los algoritmos de *Bagging* para el *dataset* Non-alcohol fatty disease.

Algoritmo	Accuracy	Precision	Recall	F1 Score	ROC AUC
LR	92.26%	56.71%	9.13%	15.73%	54.26%
SVC	92.09%	0%	0%	0%	50%
KNN	92.15%	51.32%	13.94%	21.92%	56.40%
DT	92.11%	100%	0.24%	0.47%	50.12%
RF	92.21%	52.05%	18.26%	27.04%	58.41%

3.1.3.8 *Non-alcohol fatty disease* (*Boosting*)

Finalmente, en la Tabla 3.14 se presentan las métricas calculadas en el conjunto de datos Non-alcohol fatty disease obtenido del repositorio de *Machine learning* de Kaggle a través de los algoritmos de *boosting* de AdaBoost, GradientBoosting y LightGBM.

Tabla 3.14 Resultados de las métricas con los algoritmos de *Boosting* para el *dataset* Non-alcohol fatty disease.

Algoritmo	Accuracy	Precision	Recall	F1 Score	ROC AUC
AdaBoost	92.15%	51.06%	17.30%	25.85%	57.94%
GradientBoosting	91.37%	40.68%	19.95%	26.77%	58.72%
LightGBM	91.39%	40.88%	19.95%	26.81%	58.73%

3.1.5 Entrenamiento de los mejores modelos de métodos de ensamble automático para los conjuntos de datos

Antes de continuar con esta sección, se presentan los mejores modelos obtenidos tanto para *bagging* como *boosting*. Para el presente trabajo se eligieron únicamente los mejores dos para los ensambles mencionados anteriormente.

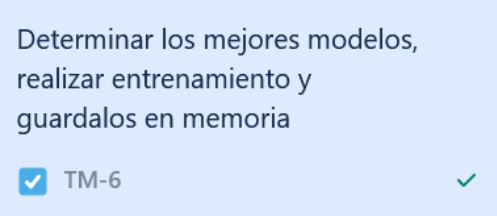


Figura 3.6 tarjeta Kanban "Determinar y entrenar modelo"

Nota: En el módulo, al momento de realizar una predicción, se ejecuta principalmente el mejor modelo para *bagging* y si este modelo está en uso, utiliza el segundo, si el segundo está en uso, procede con el mejor modelo determinado para *boosting* y en caso de que este tercer modelo se encuentré trabajando en una tarea, se toma el segundo mejor de *boosting*.

En la Tabla 3.15 se muestra el orden de utilización de los módulos en caso de que se encuentren realizando alguna tarea adicional.

Tabla 3.15 Cola de ejecución de los modelos

Primero en cola	Segundo en cola	Tercero en cola	Cuarto en cola
1er. mejor modelo de *Bagging*	2do. mejor modelo de *Bagging*	1er. mejor modelo de *Boosting*	2do. mejor modelo de *Boosting*

A continuación, en la Tabla 3.16 se exponen los mejores modelos determinados mediante el análisis de las Tablas 3.7 – 3.14.

Tabla 3.16 Mejores modelos de *bagging* y *boosting* de los conjuntos de datos

Dataset	**1er. mejor modelo de *Bagging***	**2do. mejor modelo de *Bagging***	**1er mejor modelo de *Boosting***	**2do mejor modelo de *Boosting***
Cirrhosis Prediction Dataset	Árbol de decisión (DT)	Regresión logística (LR)	GradientBoosting	AdaBoost
ILPD	Bosque aleatorio (RF)	Árbol de decisión (DT)	LightGBM	GradientBoosting
BUPA LD	Bosque aleatorio (RF)	Máquina de Soporte de Vectores (SVM)	GradientBoosting	LightGBM
NAFLD	DT	LR	AdaBoost	LightGBM

Para el entrenamiento de los modelos de ensambles de aprendizaje automático se utilizaron las bibliotecas de scikit learn de Python y se utilizó la biblioteca de joblib para almacenarlas de manera local en el sistema.

En la Lista 3 se muestra cómo se instancia el entrenamiento y se almacena de manera persistente para su posterior uso al momento de solicitar las predicciones.

```
for name, model in self._modelos.items():
        print(f'Evaluando dataset: "{self._name}" con algoritmo de boosting: {name}')
            row = {}
            #Construir el modelo
            modelo = model
            model.fit(self._X_train, self._y_train)
            #Evaluando el modelo
            y_pred = modelo.predict(self._X_test)
            row['Algoritmo'] = name
...
            row['ROC AUC'] = roc_auc_score(self._y_test, y_pred)* 100
            body = pd.concat([body, pd.DataFrame([row])])
            self.guarda_modelo(model,'Boosting', self._name, name)
    def guarda_modelo(self, modelo, tipo:str, fn:str, an:str):
            joblib.dump(modelo,f'{self._models_output}/{tipo}_{an}_{fn}.sav')
```

Lista 3 Fragmento de código de entrenamiento del modelo de ensamble de aprendizaje automático.

3.1.6 Validación cruzada de los modelos de entrenados

Finalmente, para evaluar el rendimiento de los mejores modelos entrenados previamente, de manera más confiable y robusta se utilizó la técnica de validación cruzada.

En la Tabla 3.17 se muestran los resultados obtenidos de validación cruzada para el conjunto de datos Cirrhosis Prediction Dataset con su respectiva desviación estándar.

Tabla 3.17 Resultados de validación cruzada en Cirrhosis Prediction Dataset

Algoritmo	Accuracy | Std	Precision | Std	Recall | Std	F1 Score | Std	ROC AUC | Std
Árbol de decisión (DT)	72.97% | 6.29	66.74% | 11.3	41.71% | 10.62	54.36% | 7.82	76.78% | 6.88
Regresión logística (LR)	72.25%| 5.73	64.19% | 14.29	41.08% | 10.58	50.23% | 8.93	74.83% | 5.48
GradientBoosting	69.86% | 7.39	59.13% | 11.90	47.08% | 11.69	47.73% | 9.34	71.59% | 7.49
AdaBoost	67% | 7.03	53.64% | 15.69	49.42% | 12.07	50.31% | 10.27	67.92% | 9.20

En la Tabla 3.18 se exponen los resultados obtenidos de validación cruzada para el conjunto de datos ILPD con su respectiva desviación estándar.

Tabla 3.18 Resultados de validación cruzada en Indian Liver Patient Dataset

Algoritmo	Accuracy | Std	Precision | Std	Recall | Std	F1 Score | Std	ROC AUC | Std
Bosque aleatorio (RF)	70.64% | 5.91	75.85% | 4.56	87.13% | 5.35	81.28% | 3.77	75.10% | 6.38
Árbol de decisión (DT)	71.51% | 4.89	73.86% | 4.59	92.65% | 5.43	81.03% | 2.83	73.64% | 7.51
LightGBM	71.17% | 2.52	78.31% | 3.71	82.47% | 1.65	80.28% | 1.97	72.82% | 4
GradientBoosting	69.45% | 4.86	76.32% | 5.17	82.89% | 5.38	79.34% | 4.07	71.48% | 4.53

En la Tabla 3.19 se exhiben los resultados obtenidos de validación cruzada para el conjunto de datos BUPA LD con su respectiva desviación estándar.

Tabla 3.19 Resultados de validación cruzada en BUPA Liver Disorders Dataset

Algoritmo	Accuracy | Std	Precision | Std	Recall | Std	F1 Score | Std	ROC AUC | Std
Bosque aleatorio (RF)	73.07% | 5.61	73.27% | 6.25	82.85% | 9.75	76.62% | 4.94	76.30% | 7.78
Máquina de soporte de vectores	68.05% | 5.78	68.78% | 9.59	75.22% | 11.81	73.82% | 4.85	71.93% | 5.76
GradientBoosting	69.01% | 4.02	72.15% | 5.95	77.46% | 11.58	73.86% | 4.35	73.54% | 8.38
LightGBM	69.31% | 7.69	72.90% | 9.93	76.86% | 8.63	74.14% | 6.27	73.79% | 8.38

En la Tabla 3.20 se presentan los resultados obtenidos de validación cruzada para el conjunto de datos Non-alcohol fatty disease con su respectiva desviación estándar.

Tabla 3.20 Resultados de validación cruzada en Non-alcohol fatty disease

Algoritmo	Accuracy | Std	Precision | Std	Recall | Std	F1 Score | Std	ROC AUC | Std
Árbol de decisión (DT)	92.25% | 0.42	45.53% | 35.65	2.27% | 2.09	3.08% | 3.82	81.67% | 1.99
Regresión logística (LR)	92.55% | 0.54	65.03% | 6.98	9.84% | 1.24	17.07% | 2.02	82.12% | 1.98
AdaBoost	92.56% | 0.54	57.17% | 7.38	17.39% | 2.72	26.61% | 3.83	81.01% | 1.38
LightGBM	91.60% | 0.50	40.68% | 4.29	17.83% | 2.69	24.77% | 3.36	77.16% | 1.09

Después de obtener estos resultados, se concluye que los modelos tienen consistencia y están listos para utilizarse para las predicciones requeridas.

La aplicación se construye mediante el marco de trabajo Angular que permitirá manejar registros de pacientes y predicciones en formato JSON y estos se almacenan en la base de datos no relacional de MongoDB (esto con el fin de no convertir la información en JSON, puesto que en MongoDB se almacenan en dicho formato). Esto a través de la API programada en Python: Flask, el cual gestiona las peticiones HTTP correspondientes a las operaciones que el usuario necesite.

3.2 Diseño del sistema

En esta sección se expone la arquitectura del módulo para la detección temprana de desórdenes hepáticos que cumple con la problemática planteada.

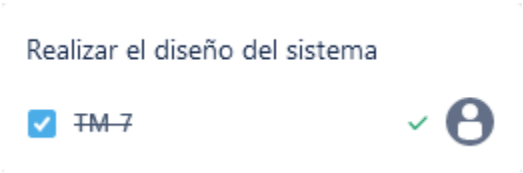


Figura 3.7 tarjeta Kanban "Realizar el diseño del sistema".

3.2.1 Arquitectura del sistema

En la figura 3.7 se muestra la arquitectura de tres capas, de las cuales se conforma este módulo.

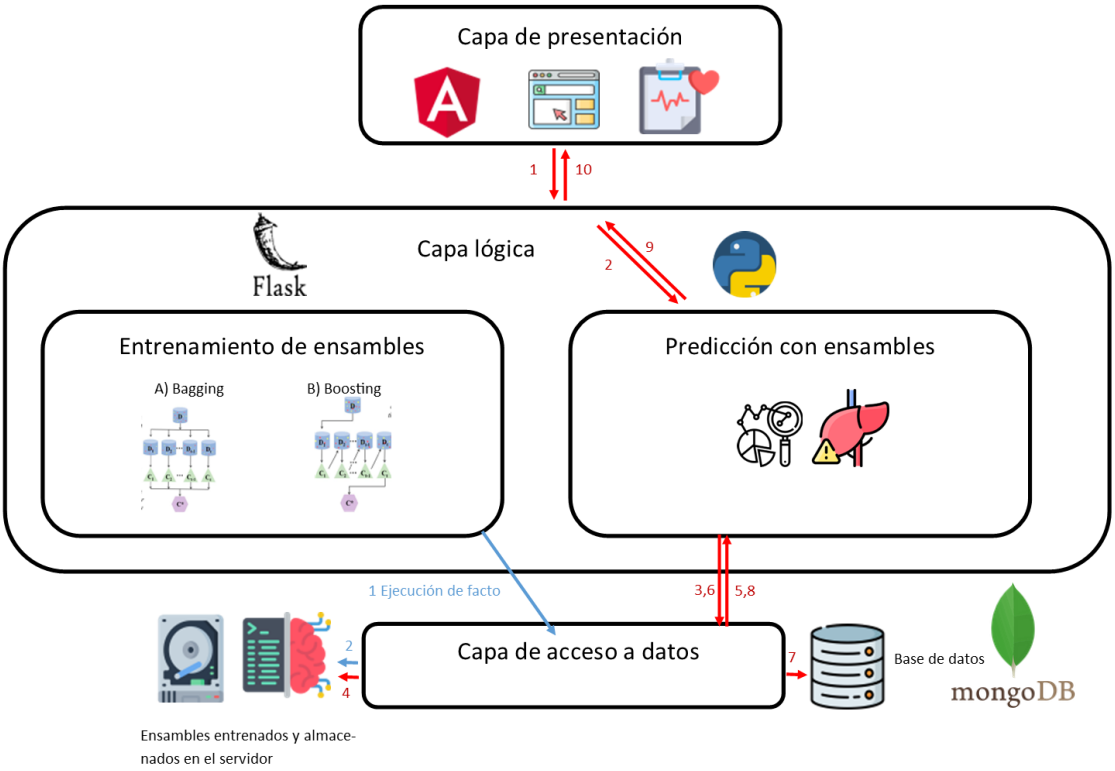


Figura 3.8 Arquitectura del módulo.

3.2.1.1 Capas de la arquitectura

A continuación, se presentan las tres capas que conforman al módulo del presente trabajo.

- **Capa de presentación:** En esta capa se encuentra la interfaz gráfica diseñada en Angular en la que el médico o personal sanitario interactúa con la aplicación. En dicha interfaz, se le proporciona al sistema los biomarcadores de un paciente en específico para que el módulo proporcione una predicción pertinente y mostrar en la pantalla dicho resultado obtenido. En caso de no proporcionar los biomarcadores preponderantes no se le permite proseguir con la predicción.
- **Capa lógica**: La capa lógica es la más importante del presente módulo, ya que aquí es donde se aloja el núcleo principal de este sistema y el *backend* proporcionado por la biblioteca Flask de Python. Una vez introducidos los datos en la capa anterior se procesan en la sección de predicción de ensambles, donde mediante las bibliotecas de scikit-learn y joblib se ejecutan los modelos de algoritmos de ensamble de aprendizaje automático previamente entrenados, obteniendo así una predicción del padecimiento seleccionado, consecutivamente almacena dicha predicción en la base de datos y el resultado obtenido se muestra al médico para que proceda a prescribir un tratamiento oportuno.
- **Capa de acceso a datos:** La capa de acceso a datos provee las APIs necesarias para la comunicación de los componentes externos al sistema, tales como: la base de datos de MongoDB (donde se almacenan los registros para llevar un historial) y los modelos entrenados con anticipación guardados previamente en el almacenamiento del sistema.

La parte lógica del sistema (Python) y la base de datos (MongoDB) se mantendrán en contenedores Docker, esto para evitar conflictos con el sistema en el que estén implementados y mejorar su rendimiento.

3.2.1.2 Flujo de trabajo

- Lectura de biomarcadores en la interfaz gráfica.
- Recepción de la lectura en la Capa lógica.
- Uso del ensamble entrenado con anterioridad y almacenado en la capa de acceso a datos para realizar una predicción con la información recibida y la obtención del resultado.
- Almacenaje del resultado en la base de datos de la aplicación para generar una historia del paciente.
- Devolver el resultado a la Capa de presentación.
- Observar el resultado en pantalla.

3.2.2 Análisis de requerimientos

En la Figura 3.9 se muestra el diagrama de casos de uso, con las acciones que el médico o administrador realizan en el sistema del módulo. Se incluye la única acción que el sistema realiza por sí mismo.

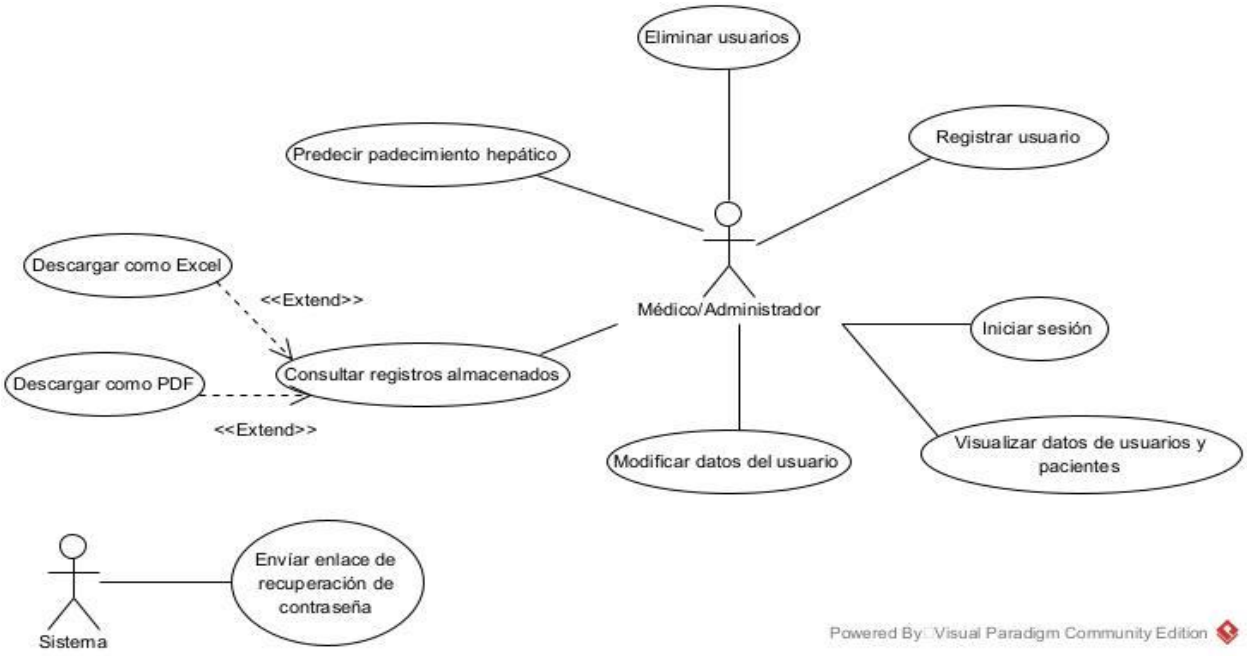


Figura 3.9 Diagrama de casos de uso.

En la Tabla 3.21 se expone la descripción del diagrama de casos de uso.

Clave	Caso de uso	Descripción
CU01	Alta usuarios	El médico/admin. registra a otros administradores, médicos o pacientes.
CU02	Baja usuarios	El médico/admin elimina del sistema a un usuario previamente registrado.
CU03	Modificación usuarios	El médico/admin modifica los datos de un usuario previamente registrado.
CU04	Predecir padecimiento hepático	El médico/admin hace una predicción para determinar si un paciente posee desórdenes hepáticos y si es médico proporcionar un tratamiento.
CU05	Ver datos de paciente	El médico/admin consulta el historial de un paciente en específico.
CU06	Consultar registros almacenados	El médico/admin consulta las predicciones almacenadas en el sistema.
CU06.01	Descargar como PDF	El médico/admin descarga la información en formato PDF.
CU06.02	Descargar como Excel	El médico/admin descarga la información en formato de hojas de cálculo de Excel.
CU07	Enviar enlace de recuperación de contraseña.	El sistema genera un token, el cual es enviado al usuario para que restablezca sus credenciales de acceso.

Tabla 3.21 Descripción de los casos de uso

Nota: El término usuario sin ser el que ejecuta la acción se refiere a un administrador, médico y/o paciente.

3.2.3 Modelo de datos

En esta sección se presenta el modelo de datos utilizado para la base de datos no relacional de MongoDB.

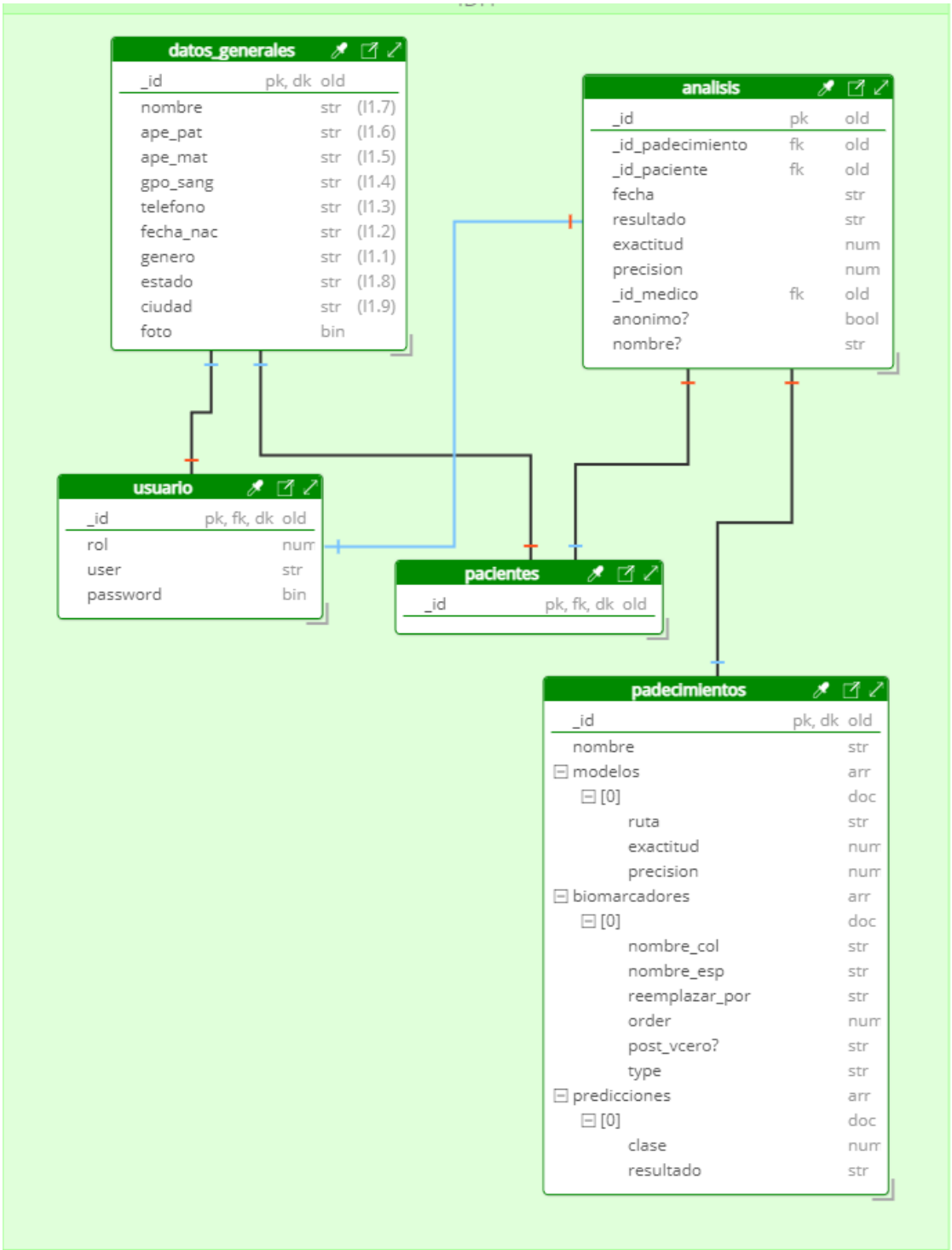


Figura 3.10 Modelo de datos de MongoDB.

A continuación, en la Tabla 3.22 se muestra una descripción de cada uno de los cinco documentos.

Tabla 3.22 Descripción de los documentos de MongoDB

Documento	Descripción
Datos generales	Este documento almacena información básica del perfil tanto de pacientes, médicos y administradores.
Usuario	Este documento almacena los datos que necesita un médico o administrador para iniciar sesión en el sistema.
Paciente	Este documento clasifica a un usuario como paciente.
Padecimiento	Este documento representa a un padecimiento para predicción y tiene valores clave tales como: los modelos previamente entrenados y los biomarcadores que necesita para proporcionar una predicción.
Análisis	Este documento representa a un análisis con una predicción previamente realizada a un paciente en específico.

Consecuentemente, en las siguientes tablas se exponen a detalle los datos y sus respectivos tipos almacenados en los documentos previamente presentados.

Tabla 3.23 Descripción de los datos del documento Datos generales

Datos	Tipo	Descripción	Relación
_id_usuario	Number	Es el identificador del usuario	No tiene
nombre	String	Nombre del usuario	No tiene
ape_pat	String	Apellido paterno del usuario	No tiene
ape_mat	String	Apellido materno del usuario	No tiene
gpo_sang	String	Grupo sanguíneo del usuario	No tiene
telefono	String	Número telefónico del usuario	No tiene
fecha_nac	Date	Fecha de nacimiento del usuario	No tiene
genero	String	Género del usuario	No tiene
estado	String	Estado de la república donde reside el usuario	No tiene
ciudad	String	Ciudad de la república donde reside el usuario	No tiene
foto	Binary	Fotografía del usuario	No tiene

Tabla 3.24 Descripción de los datos del documento Usuario

Datos	Tipo	Descripción	Relación
_id_usuario	ObjectID	Es el identificador de los datos generales del administrador o médico.	Datos generales
user	String	Correo electrónico con el que el usuario accede al sistema	No tiene
password	String	Contraseña con la el que el usuario accede al sistema	No tiene
rol	Number	Rol del usuario: **0** para administrador y **1** para médico.	No tiene

Tabla 3.25 Descripción de los datos del documento Paciente

Datos	Tipo	Descripción	Relación
_id_usuario	ObjectID	Es el identificador de los datos generales del paciente.	Datos generales

Tabla 3.26 Descripción de los datos del documento Análisis

Datos	Tipo	Descripción	Relación
_id_analisis	Number	Es el identificador del análisis realizado previamente	No tiene
_id_padecimiento	ObjectID	Es el identificador del padecimiento predicado para obtener sus detalles.	Padecimiento
_id_paciente	ObjectID	Es el identificador del paciente al cual se somete a la predicción.	Paciente
_id_medico	ObjectID	Es el identificador del médico que realizó dicha predicción	Usuario
fecha	Date	Es la fecha en la cual se realizó dicha predicción	No tiene
resultado	String	Es el resultado del pronóstico.	No tiene
exactitud	Number	Es la exactitud proporcionada por el modelo al realizar la predicción	No tiene
precisión	Number	Es la precisión proporcionada por el modelo al realizar la predicción	No tiene
anonimo?	Boolean	Bandera que determina si fue una predicción anónima	No tiene

Datos	Tipo	Descripción	Relación
nombre?	String	Nomenclatura del paciente, la cual únicamente aparece si fue una predicción anónima.	No tiene

Tabla 3.27 Descripción de los datos del documento Padecimiento

Datos	Tipo	Descripción	Relación
_id_padecimiento	Number	Es el identificador del padecimiento.	No tiene
nombre	String	Es el nombre del padecimiento.	No tiene
modelos	Documento	Contiene la información de los modelos entrenados previamente para este padecimiento con datos como: **ruta del modelo**, **exactitud** y **precisión.**	No tiene
biomarcadores	Lista de documentos	Proporciona la lista de biomarcadores preponderantes para realizar una predicción tales como: **nombre** y el **nombre del valor a reemplazar** (si se necesita).	No tiene
predicciones	Lista de documentos	Provee la definición de la clase a predecir para proporcionar un diagnóstico al padecimiento seleccionado.	No tiene

3.2.4 Diagrama de clases del sistema

En esta sección en la Figura 3.11, se presenta el diagrama de clases del *backend* por el cual está conformado el sistema.

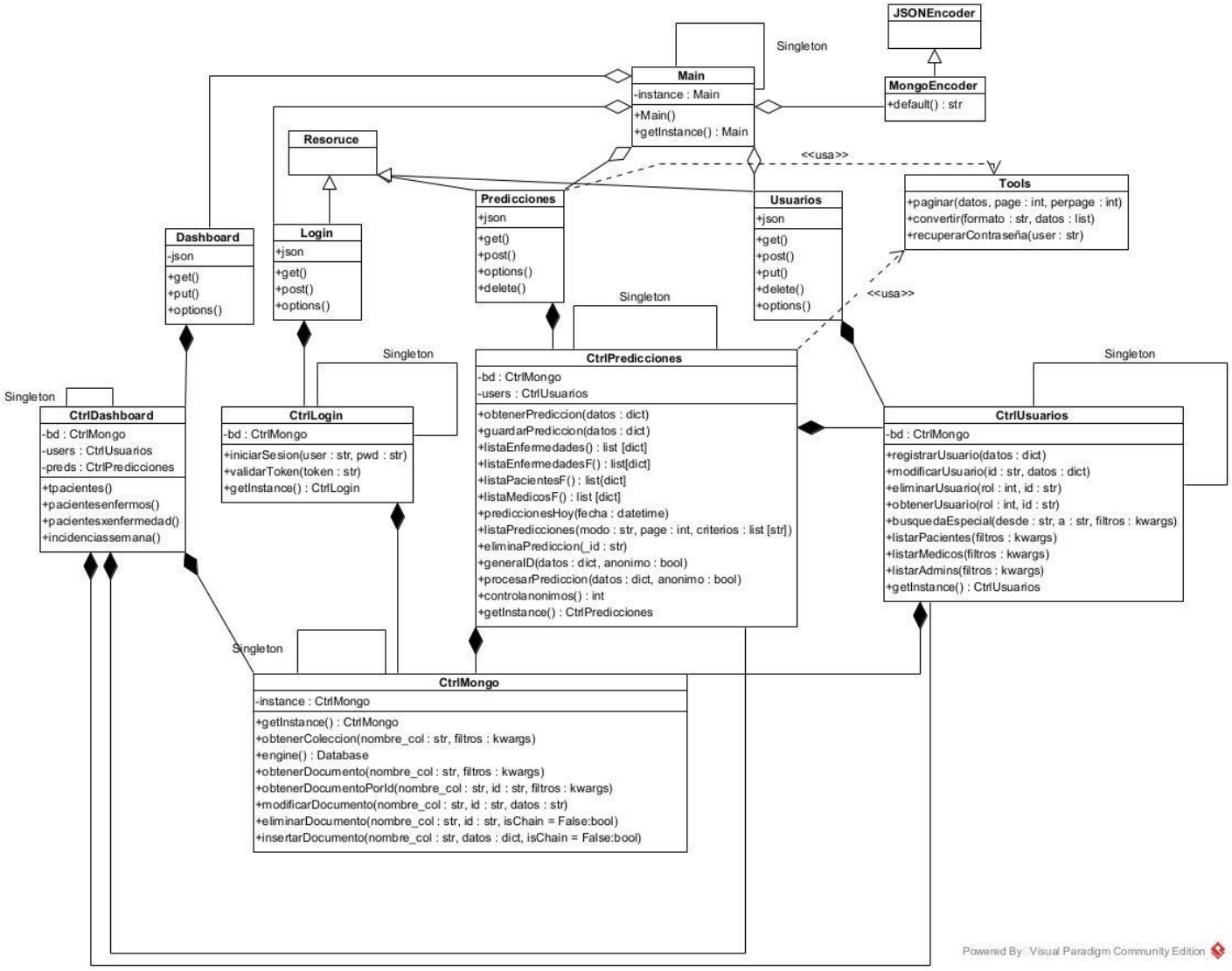


Figura 3.11 Diagrama de clases del sistema.

3.2.5 Modelado del funcionamiento

En esta sección se muestran la secuencia de comunicación entre el usuario y la aplicación web del presente trabajo.

En la Figura 3.12 se exhibe el diagrama de actividades de la predicción de un padecimiento, cabe mencionar que el usuario debió iniciar previamente sesión en el sistema o que su sesión no expiró. En dicho diagrama se exhibe el procedimiento de inserción y validación de los biomarcadores, así como lo que sucede en caso de éxito o error.

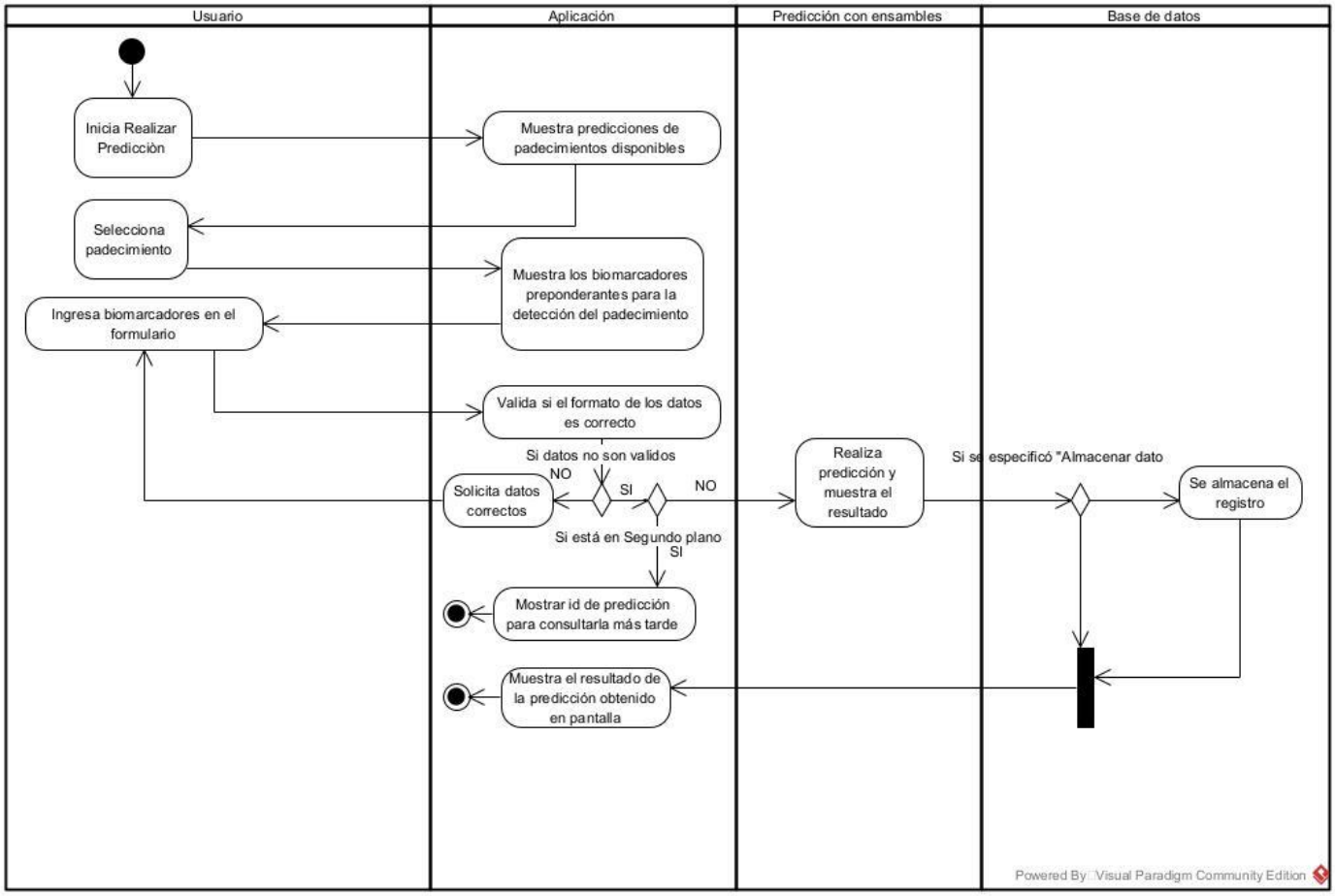


Figura 3.12 Diagrama de actividades de Realizar predicción.

En la Figura 3.13 se presenta el diagrama de actividades de la obtención de una predicción de un padecimiento almacenado con anterioridad, cabe mencionar que el usuario debió iniciar previamente sesión en el sistema o que su sesión no expiró. En este diagrama se presenta el proceso en el cual un médico obtiene la predicción y las diferentes acciones con las cual dispone de esa información que se obtuvo.

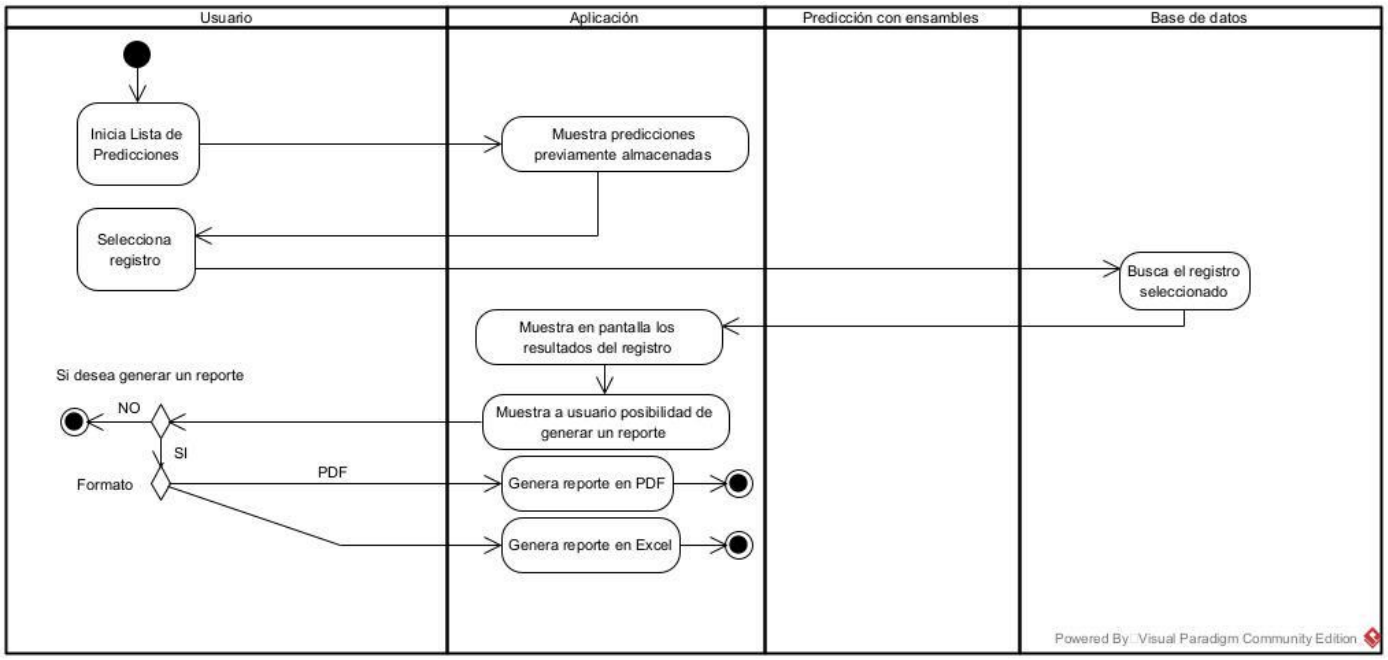


Figura 3.13 Diagrama de actividades de obtención de predicciones almacenadas.

3.2.6 Diseño del sistema

En esta sección se exhiben los bosquejos para la interfaz gráfica y la presentación de resultados en pantalla.

En la Figura 3.14 se presenta el *Mockup* de *Login*, con el cual el usuario podrá ingresar al sistema del presente trabajo. Cabe recalcar que un administrador previamente debió registrar al médico u otro administrador. Dicho administrador existe gracias a la carga previa de datos en el sistema.

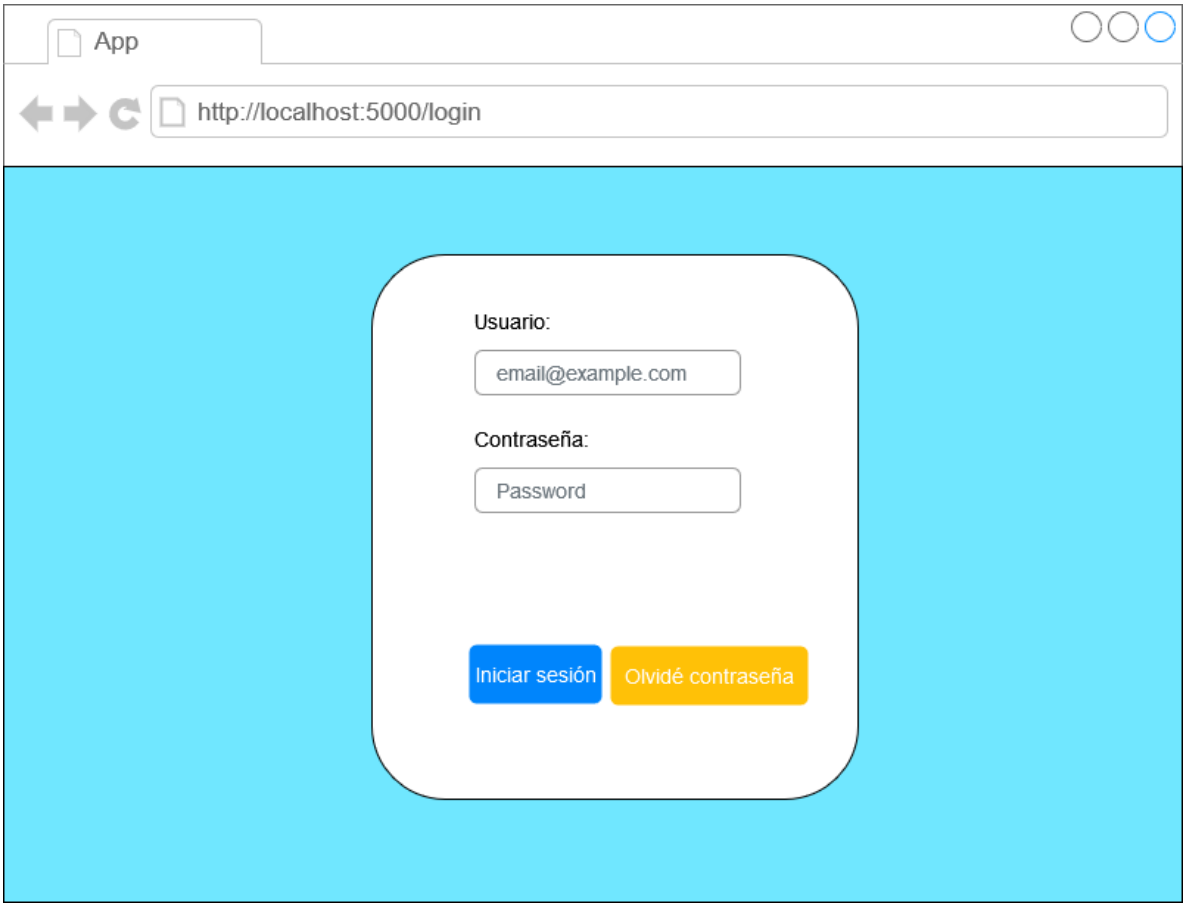


Figura 3.14 *Mockup* para la autenticación del usuario.

En la siguiente figura se expone el *Mockup* de *Dashboard*, en el cual el usuario visualiza un resumen general y los botones que le proporcionan acceso a diferentes secciones del sistema.

En la Figura 3.15 se presenta el *Mockup* de la interfaz de Recuperar contraseña, la cual permite a un usuario de la aplicación, recuperar su acceso en caso de olvidar su credencial de acceso. El sistema envía un correo electrónico a la dirección con la cual se registró el usuario y le da su token de recuperación.

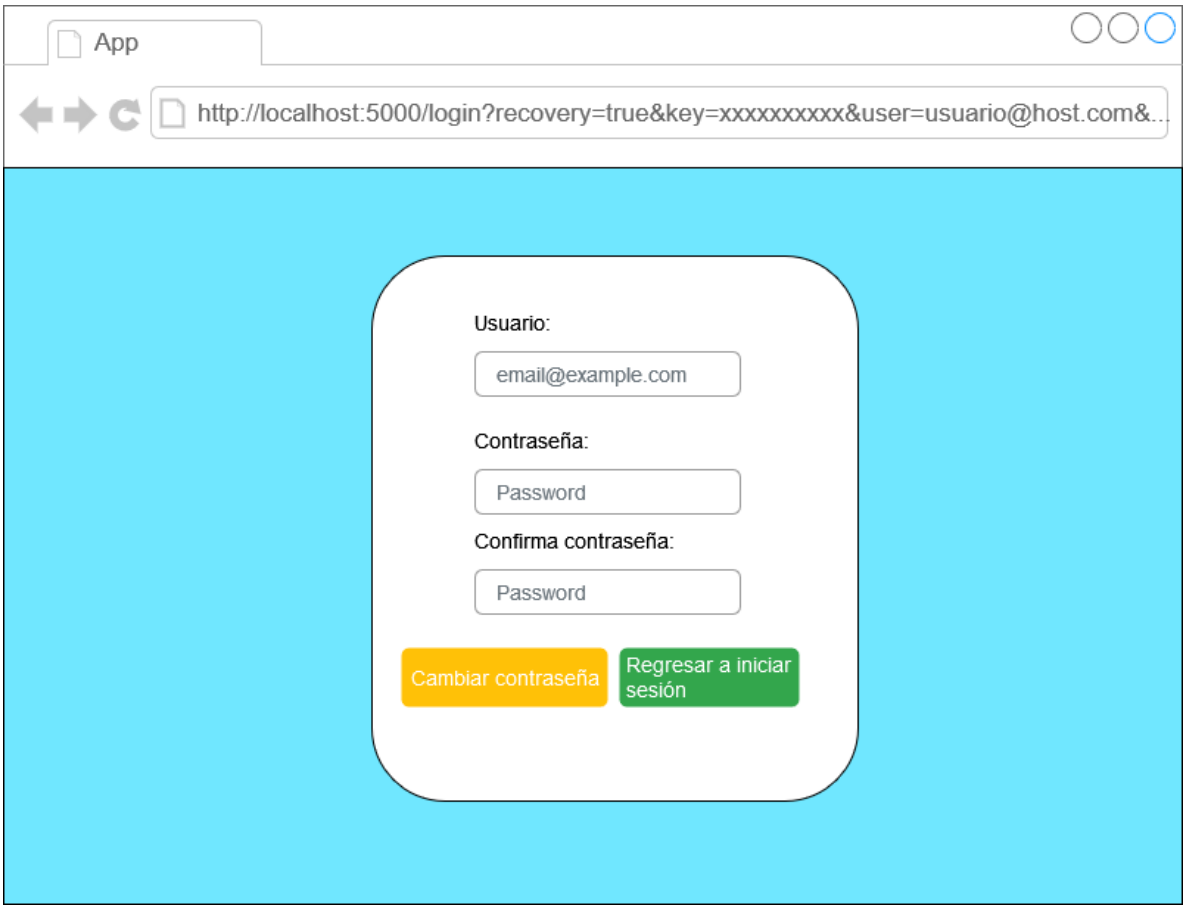


Figura 3.15 *Mockup* para la recuperación de contraseña.

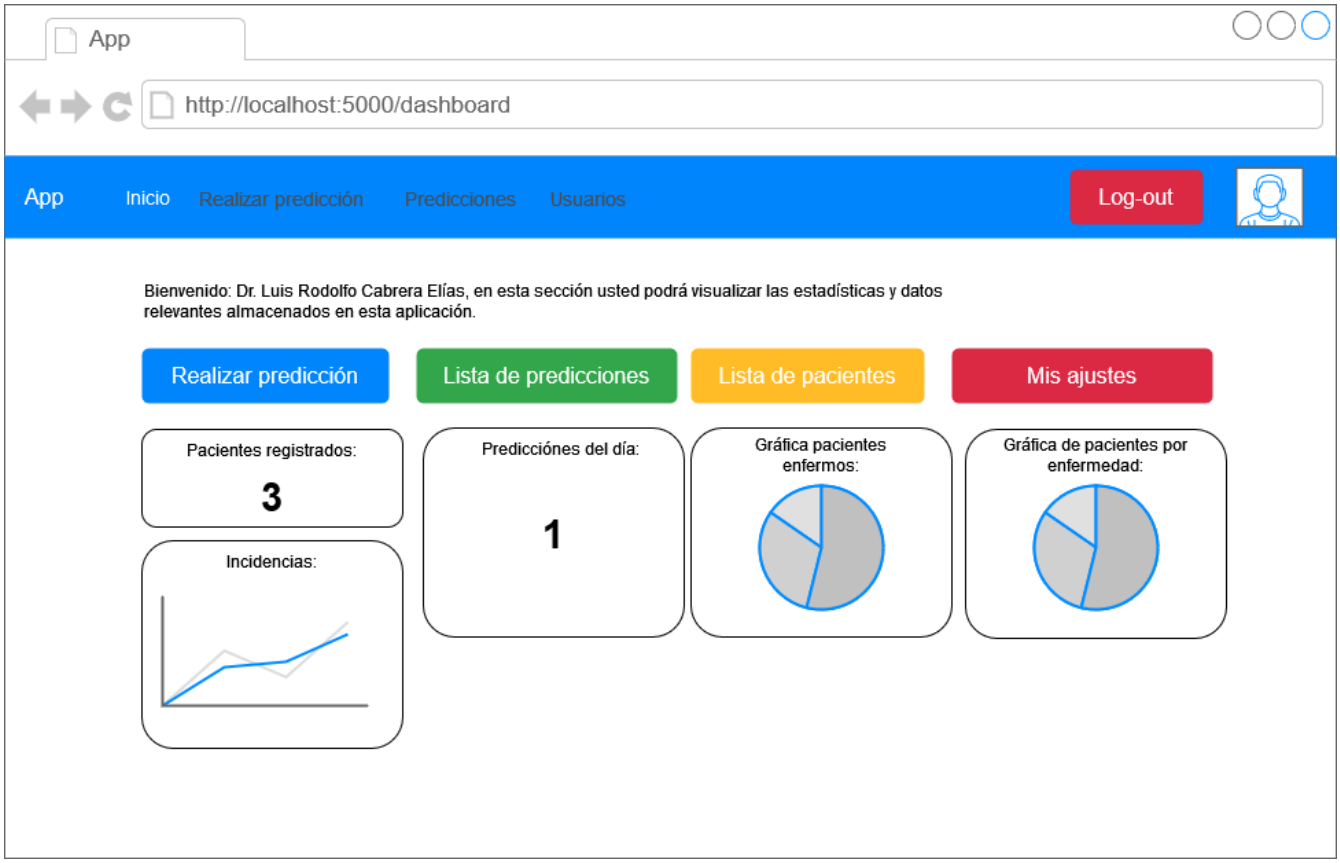


Figura 3.16 *Mockup* para el *dashboard* del usuario.

Si el usuario da clic en el botón de Realizar predicción, se redirige a la sección de Predicciones, tal como se presenta en la siguiente figura.

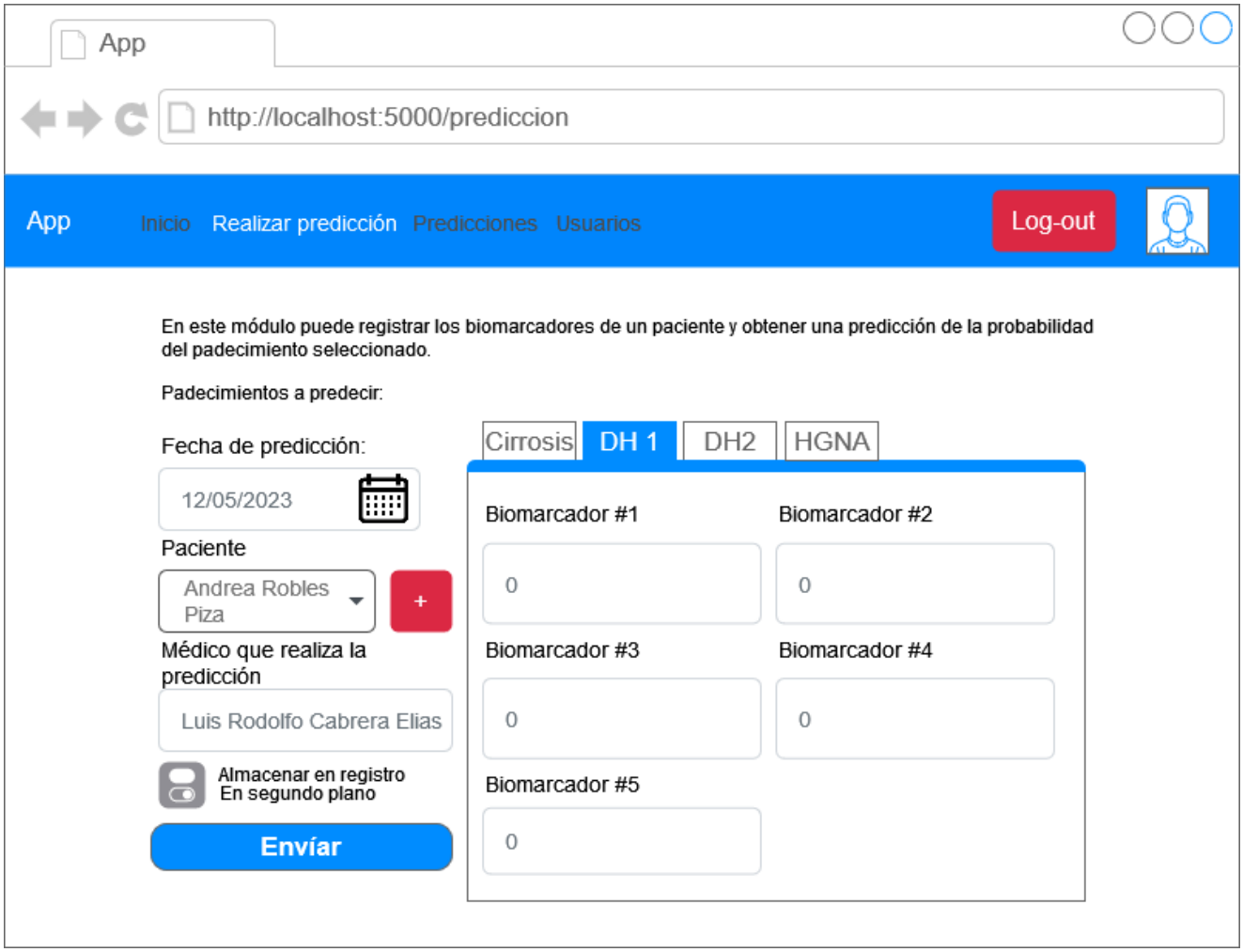


Figura 3.17 *Mockup* de Realizar predicción.

En esta sección al médico se le presentan los siguientes campos:

- Fecha de predicción: Campo en el que se especifica la fecha del análisis, por defecto viene la del presente día.
- Paciente: La lista con los pacientes registrados en el sistema para asignarle a uno la predicción.
- Médico que realiza la predicción: Nombre del médico que está realizando la predicción actual (únicamente con fin informativo).
- Pestañas de padecimientos: En estas pestañas, se establecen los campos de los cinco biomarcadores preponderantes para realizar la predicción.

- Almacenar en registro: Esta opción indica al sistema si esta predicción se almacena para el historial del paciente o realizarla sin seguimiento alguno.
- En segundo plano: Esta opción otorga el id de la predicción para consultarlo más tarde, muy útil si el sistema se encuentra ocupado recibiendo varias peticiones.

En la siguiente figura se presenta se exhibe la interfaz de Lista de usuarios, a la cual se accede si el usuario da clic en el botón Usuarios en la barra de navegación.

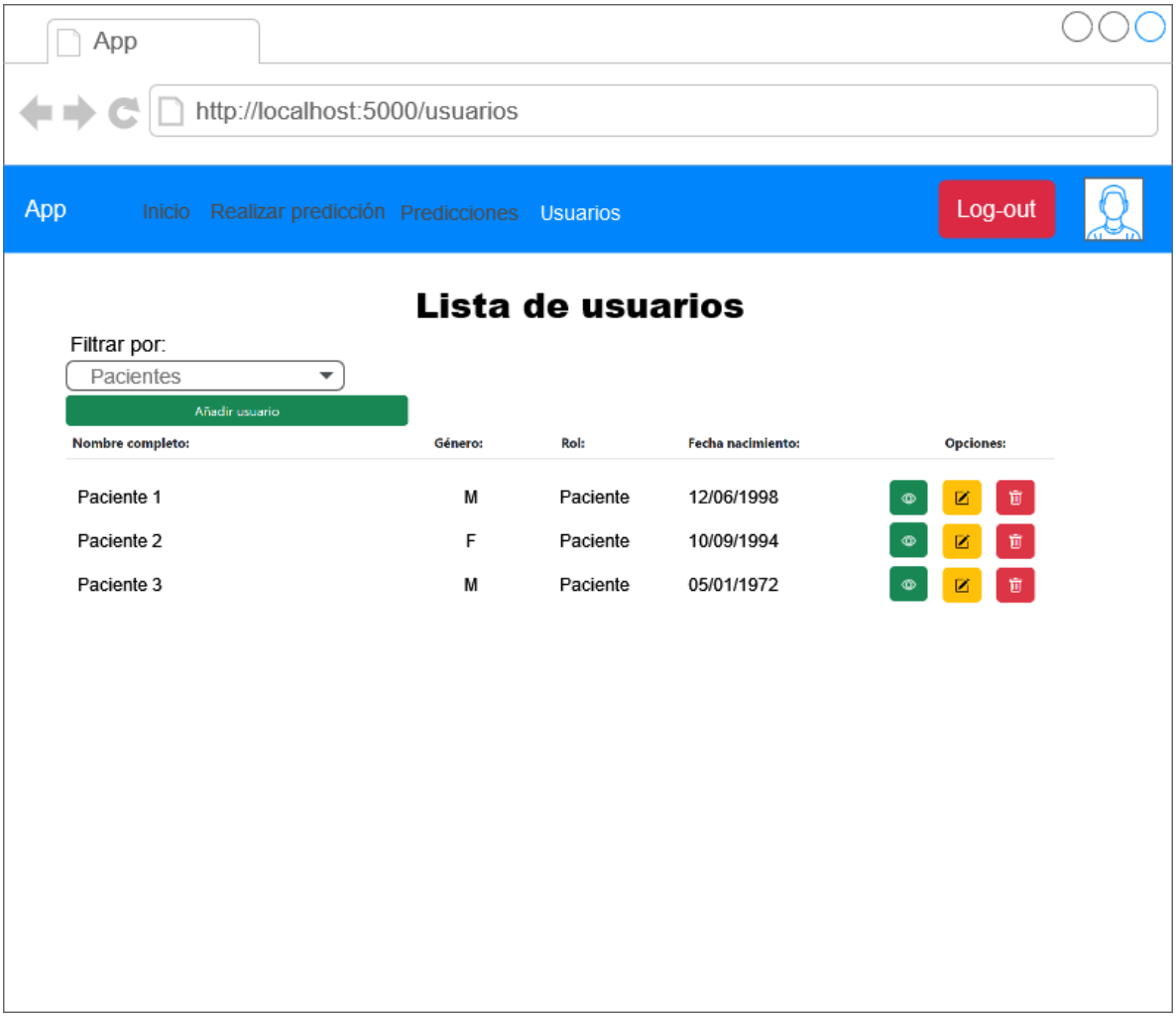


Figura 3.18 *Mockup* Lista de usuarios.

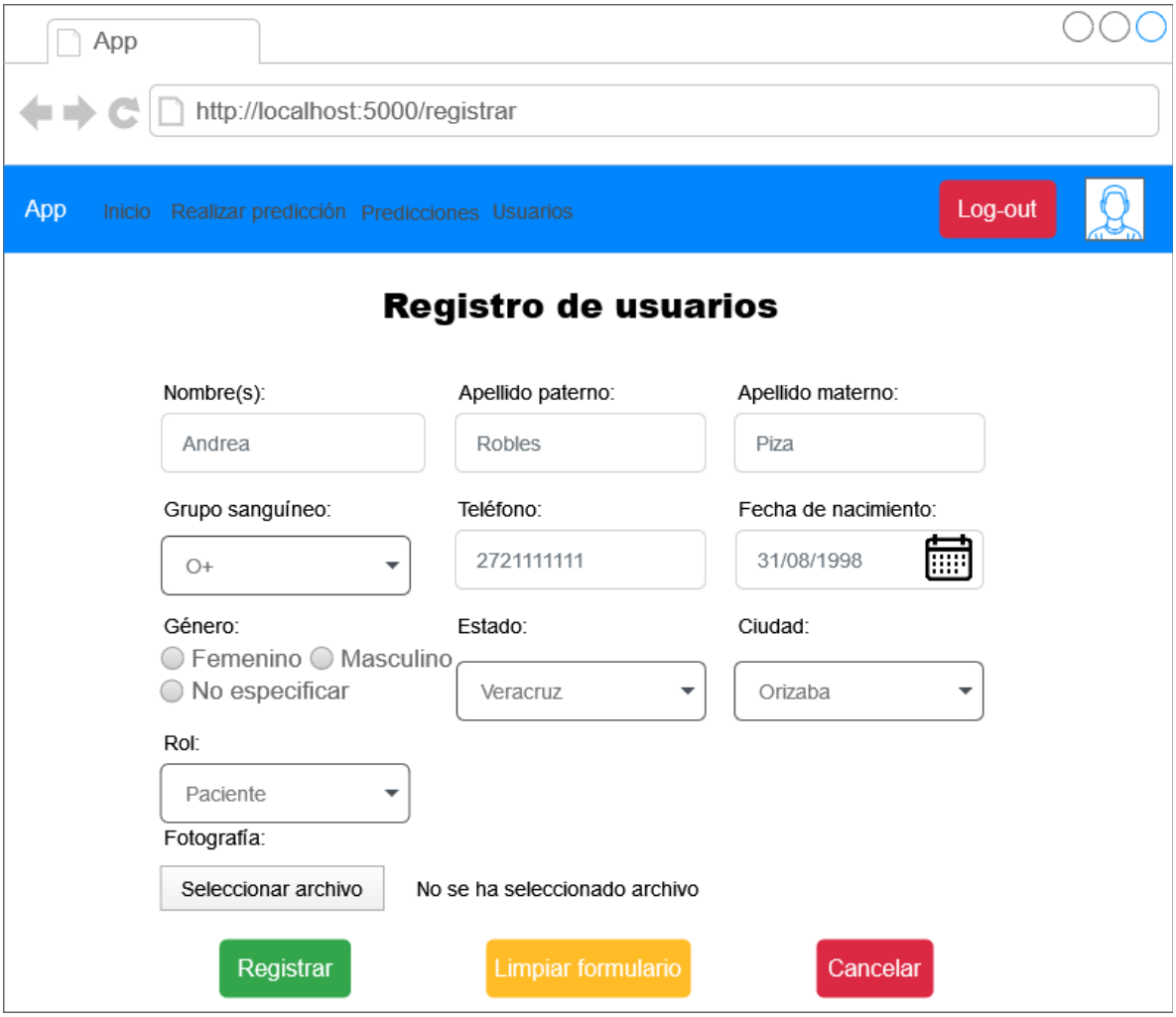


Figura 3.19 *Mockup* Registro de usuario (Paciente)

En este formulario se solicitan los datos generales del administrador, médico o paciente que se registra en el sistema. En caso de ser un administrador o un médico, se le habilitan dos campos adicionales para establecer el email y contraseña tal como se muestra en la siguiente figura.

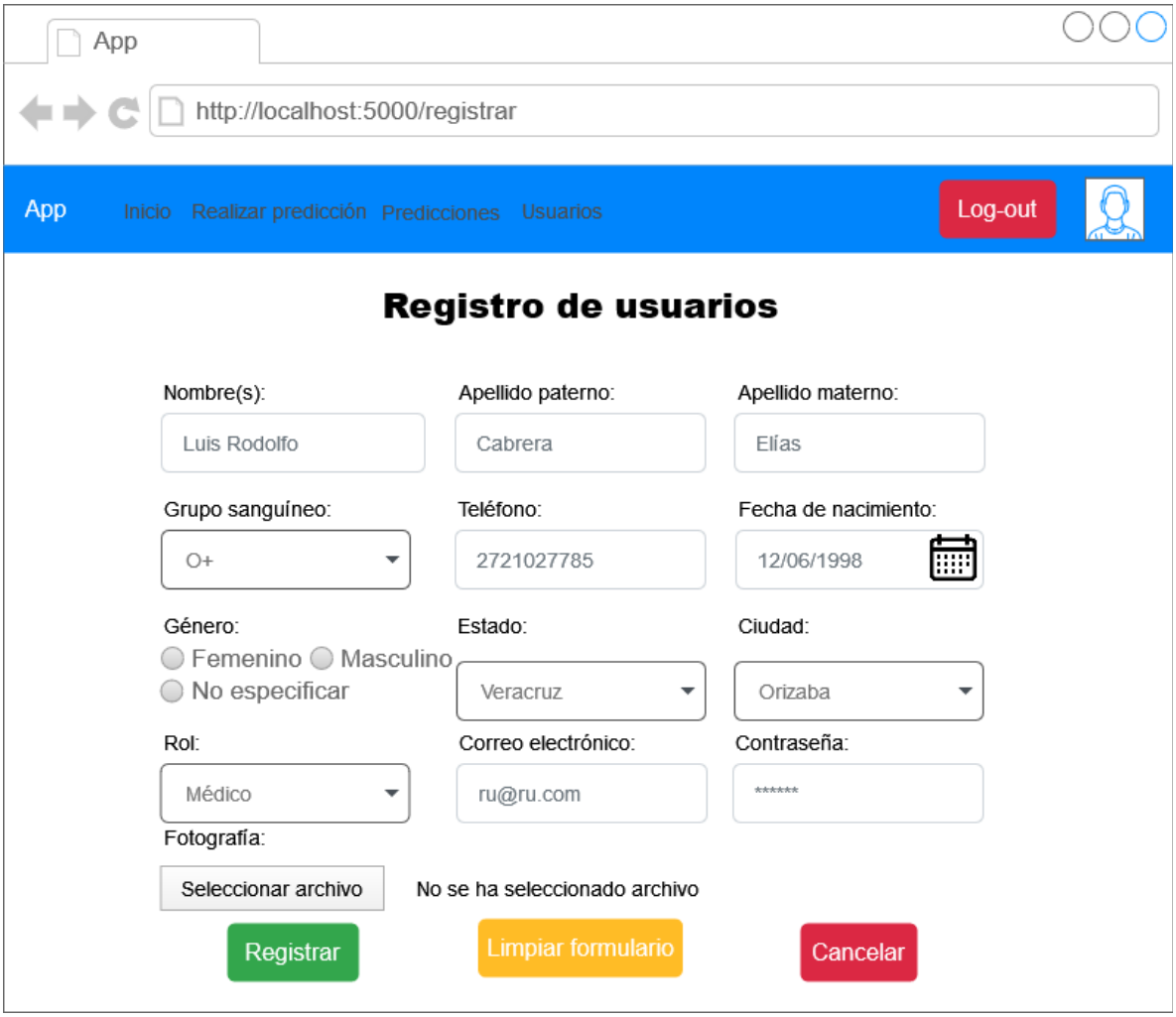


Figura 3.20 *Mockup* Registro de usuario (Médico o Administrador)

En la siguiente figura se muestra la interfaz de Lista de predicciones, la cual se accede si el usuario da clic en el segundo botón del *dashboard* y permite visualizar las predicciones previamente almacenadas con distintos filtros para facilitar su búsqueda, ya sea: **todos**, **rango de fecha** o **paciente**.

Cabe recalcar que también dichos registros tienen disponible la opción de exportarlos a PDF o Excel.

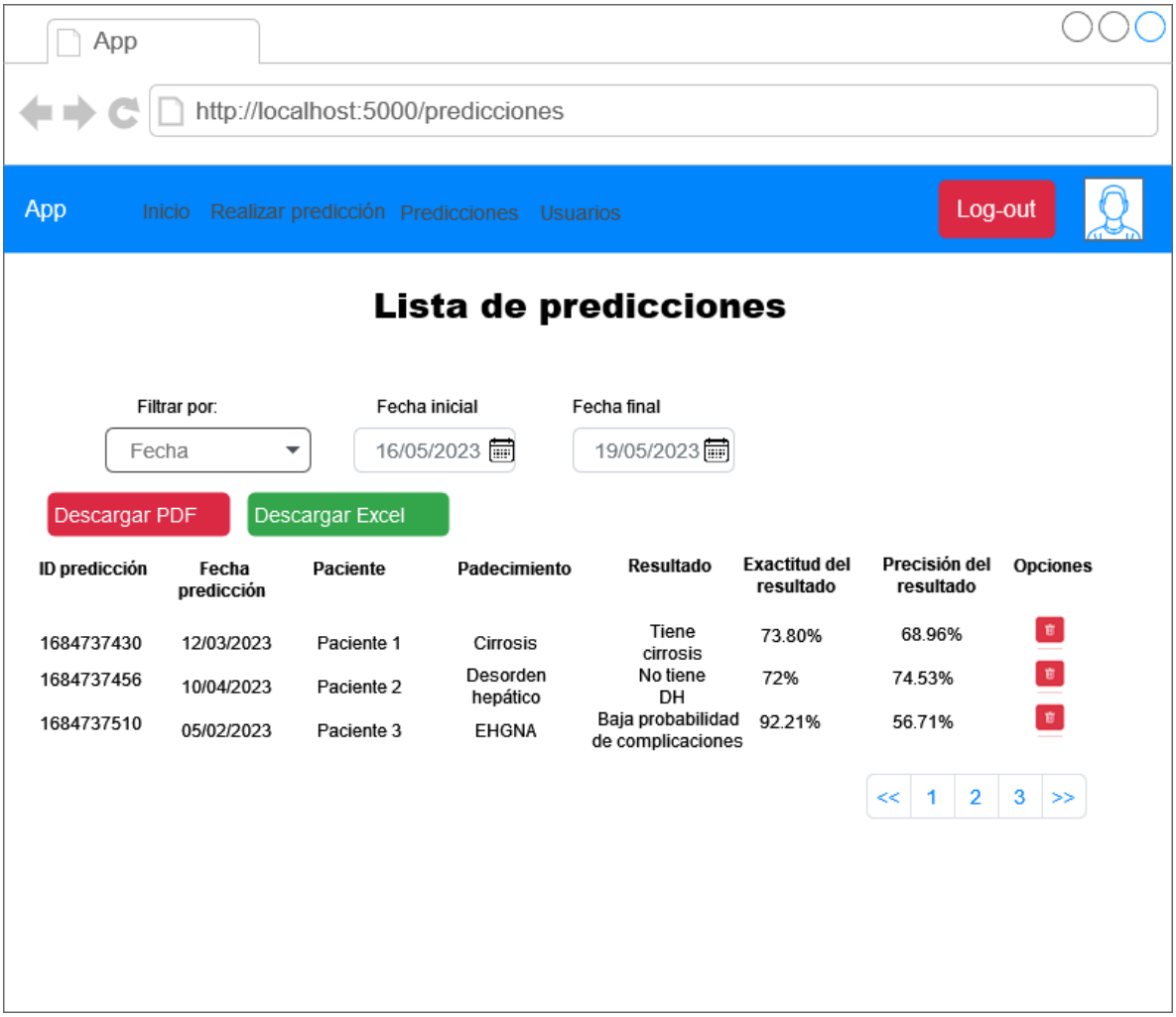


Figura 3.21 *Mockup* de Lista de predicciones

En la presente figura, se muestran los registros previamente obtenidos de un paciente en particular tal como se enseña en la Figura 3.21.

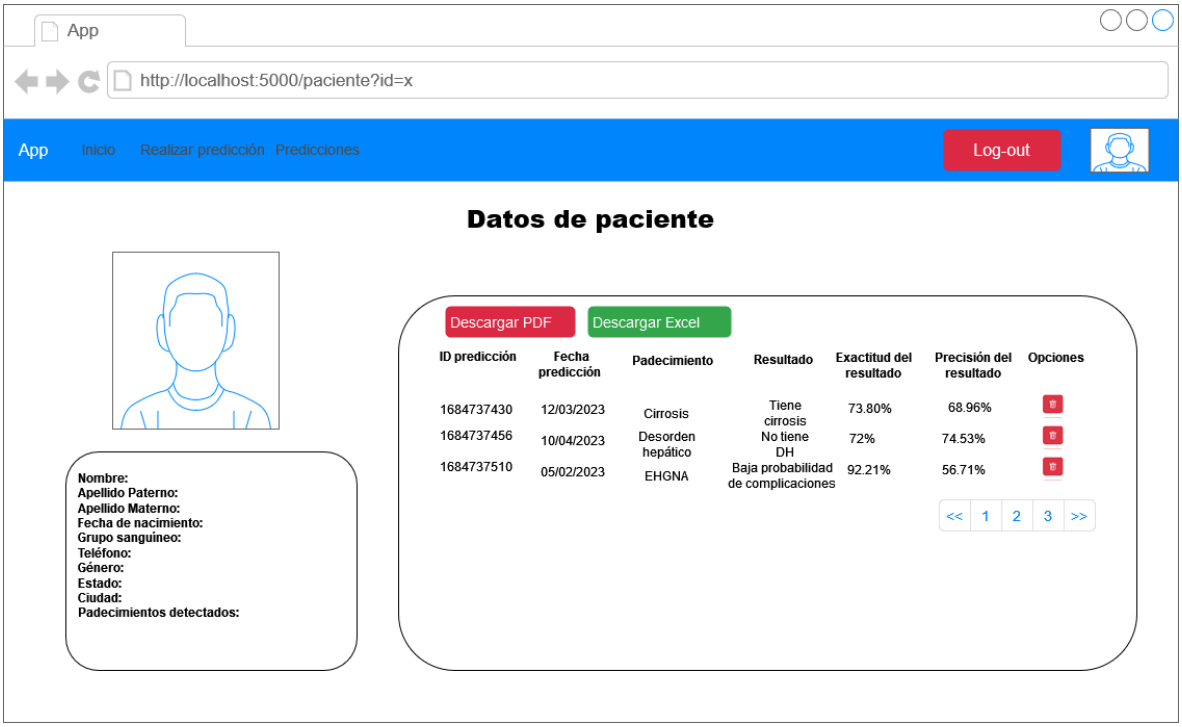


Figura 3.22 *Mockup* de Datos de paciente

En la Figura 3.23 se muestra la sección de Mis ajustes, donde el usuario es capaz de cambiar sus credenciales de acceso en caso de requerirlo.

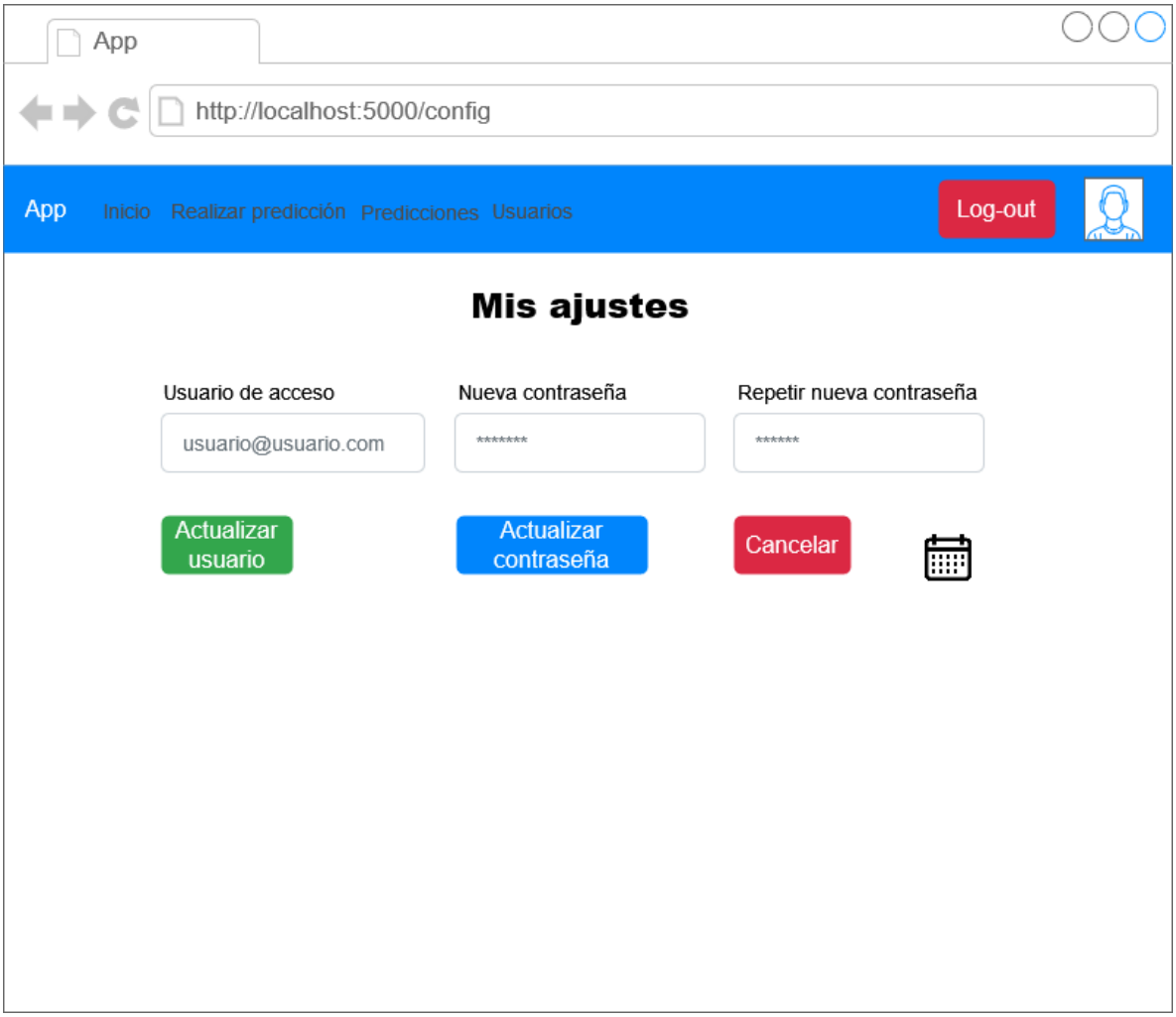


Figura 3.23 *Mockup* de Mis ajustes

3.2.7 Mapa de navegación

En esta sección se exhibe el mapa navegacional de la aplicación en la Figura 3.24 donde se muestra el flujo desde que el usuario aprueba su autenticación y navega en todas las secciones presentadas en los *mockups* o restablece su contraseña en caso de haberla olvidado o tener comprometida su seguridad de las credenciales.

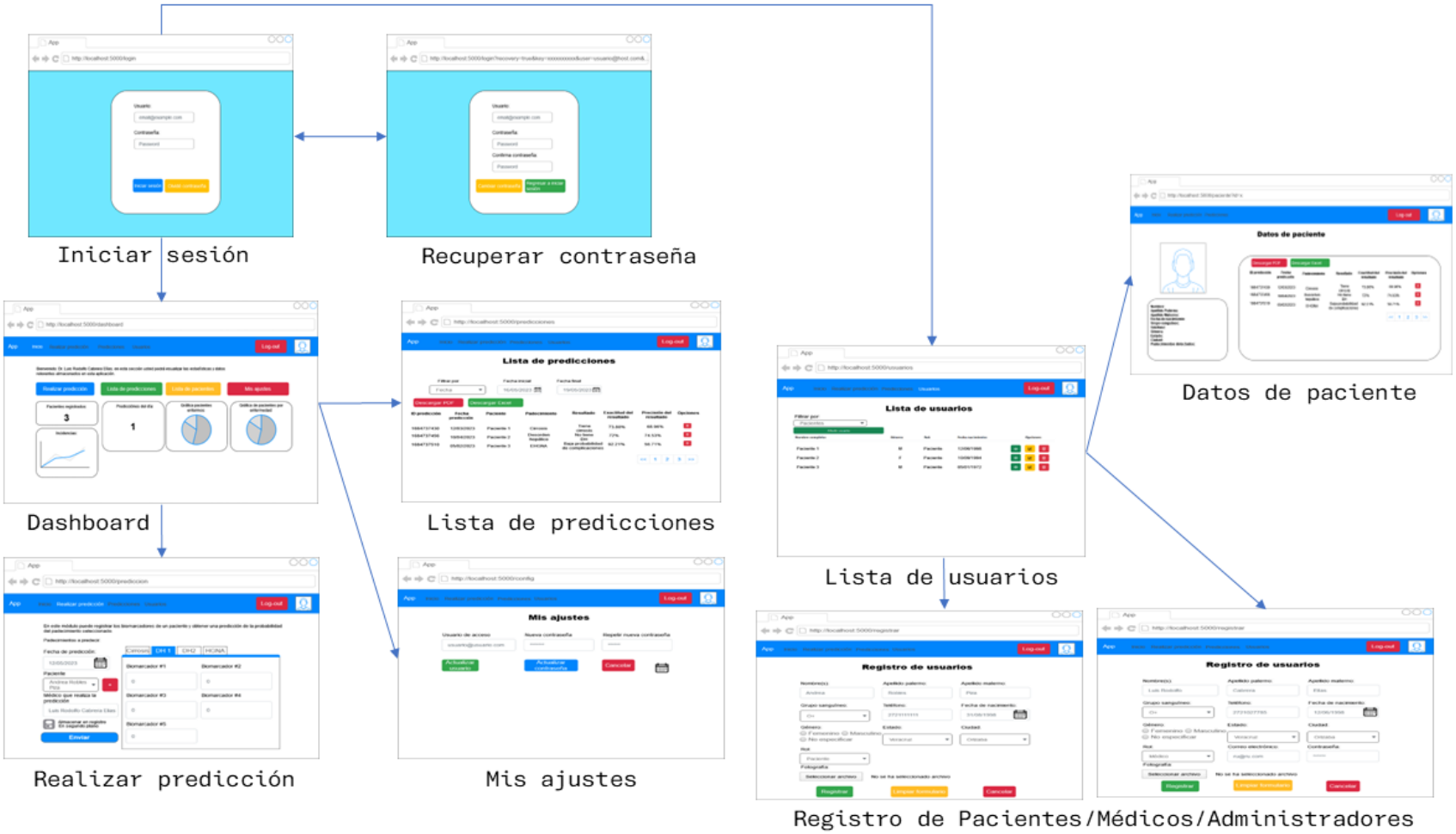


Figura 3.24 Mapa de navegación.

3.2.8 Despliegue de la aplicación

En esta sección se presenta como se realiza el despliegue de la aplicación web mediante el uso del entorno *Docker*, el cual garantiza seguridad al separar en entornos diferentes las partes y/o módulos del presente trabajo.

A continuación, en la Figura 3.25 se presenta el diagrama de despliegue antes mencionado.

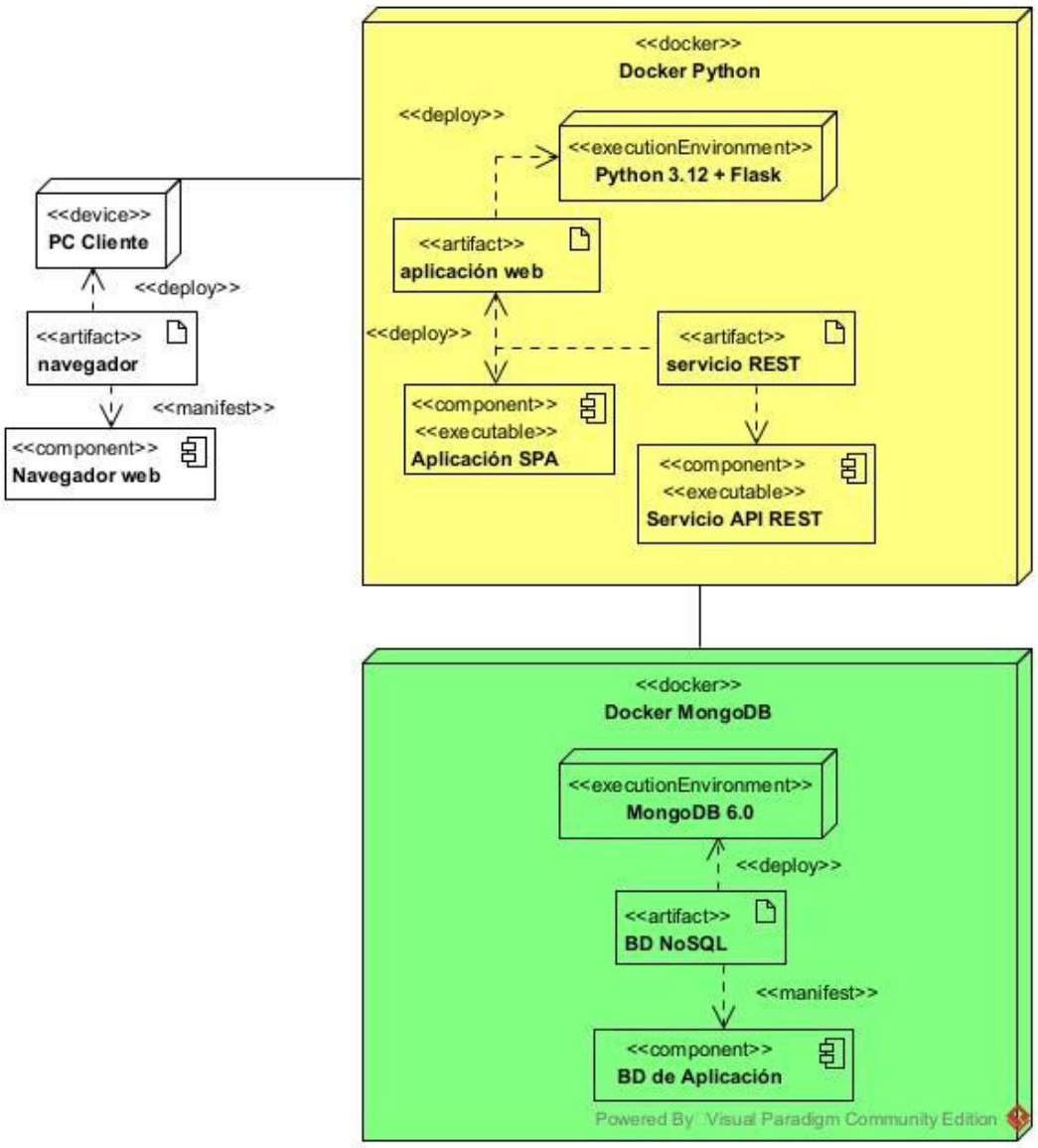


Figura 3.25 Diagrama de despliegue de la aplicación.

3.3 Implementación

En la presente sección se detalla el desarrollo de la aplicación web.

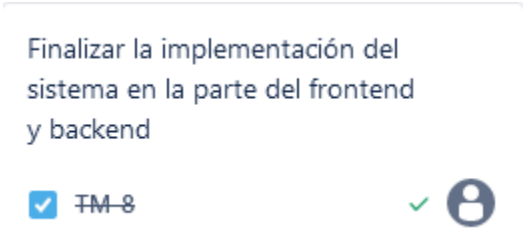


Figura 3.26 Tarjeta Kanban "Finalizar la implementación del sistema en la parte del *frontend y backend*".

3.3.1 Predicción de desorden hepático

En el módulo REST de la aplicación, dentro de la sección de Predicciones, una vez el médico tratante envía la información a través de la aplicación web, el sistema hace uso de los modelos previamente entrenados para realizar una predicción y mostrar el registro posteriormente y guardarlo (sólo en caso de especificarlo) o si es un registro anónimo, se almacena únicamente como: **Paciente <n>** (donde **n**, es el número de predicciones anónimas que el sistema realizó).

Cada padecimiento hepático tiene diferentes biomarcadores y estos se solicitan en la aplicación web y únicamente se envían al modelo el cual da una respuesta.

En la Lista 4 se presenta el procedimiento de predicción del desorden hepático, desde que recibe la información hasta que la pasa al primer modelo disponible del trastorno a predecir.

```
def procesarPredicción(self, datos:dict, anonimo:bool = False):
        if not anonimo:
            _id_paciente = datos['_id_paciente']
_id_medico = datos['_id_medico']

_id_padecimiento = datos['_id_padecimiento']

modelo = None
fecha = datos['fecha']

tipo_pred = datos['tipo_pred']
biomarcadores = {k:v for (k,v) in datos.items() if k not in
['_id_paciente','_id_medico','_id_padecimiento', 'fecha','tipo_pred']}
opt_padecimiento=self._bd.obtenerDocumentoPorId("padecimientos", _id_padecimiento)

…
try:
   for mod in opt_padecimiento[‘modelos’]:

       modelo = joblib.load(mod[‘ruta’])

       res = modelo.predict([biomarcadores_ev])

       res = res[0]

       exactitud = mod[‘exactitud’]
       precisión = mod[‘precision’]
       …
       return {“ok” : a} | registro
```

Lista 4: Fragmento de código de realizar predicción

3.3.2 *Dashboard* del usuario

En el módulo de *dashboard*, se generan gráficas que permiten conocer los **pacientes registrados**, **predicciones del día**, **incidencias de la semana**, **gráfica total de los pacientes enfermos** y **el total de pacientes enfermos por padecimiento**. Esto con el fin de llevar un control visual amigable de los registros almacenados dentro de este sistema. Esta interfaz intuitiva facilita a los médicos la interpretación rápida de la información, brindando una visión global de la situación de salud de los pacientes en cualquier momento.

En la Lista 5 se exhibe el fragmento de código que genera las gráficas que se visualizan en el *dashboard*.

```
async pintarGraficas(){
const spacientes = await lastValueFrom(this.serv.spacientes());
this.enfermos = spacientes['enfermos'];_id_medico = datos['_id_medico']
this.sanos = spacientes['sanos'];modelo = None

const grafica1 = new Chart((<HTMLCanvasElement>document.getElementById("g1"))
,{
type:'doughnut',
data : {
labels: ['Enfermos','No enfermos'],
datasets: [{
label : "",
data: [this.enfermos,this.sanos]
}]
}
});
```

Lista 5: Fragmento de código de generar gráficas

3.3.3 Predicción del desorden hepático (interfaz web)

En el módulo de realizar predicción, se toman los biomarcadores del paciente al cual se le quiere realizar la predicción, cabe recalcar que la aplicación pedirá diferentes biomarcadores dependiendo del padecimiento seleccionado. Una vez se han elegido las opciones de la predicción y ningún biomarcador se ha dejado vacío, se solicita al sistema ejecutar el pronóstico pertinente.

En caso de solicitar una predicción en segundo plano, aparece el siguiente mensaje únicamente con el id de la operación para consultarlo en cuanto le sea posible al médico.

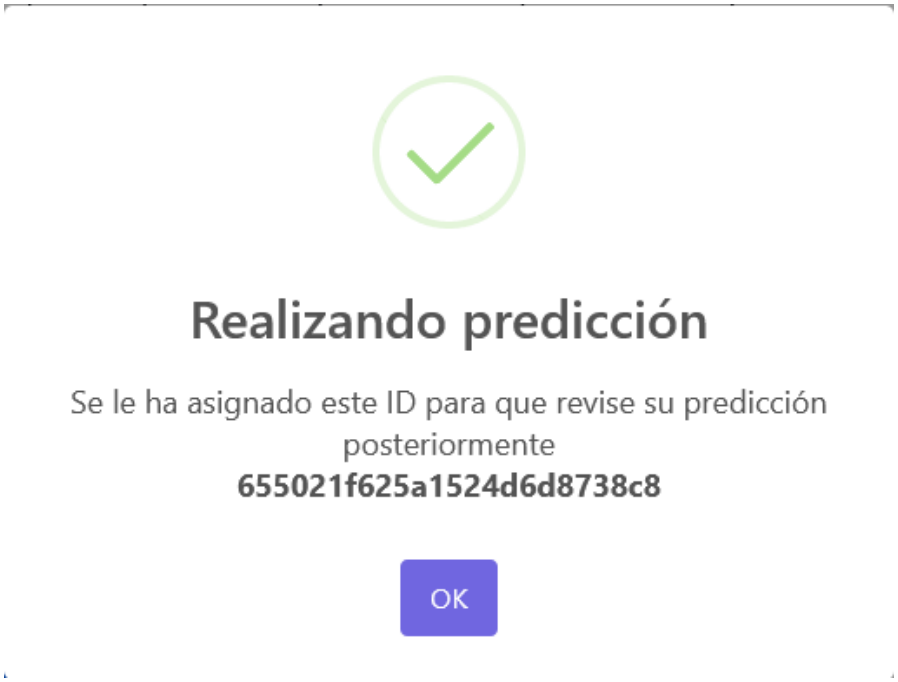


Figura 3.27 Predicción en segundo plano.

En caso de seleccionar en modo normal, aparece la predicción con los resultados arrojados por el sistema.

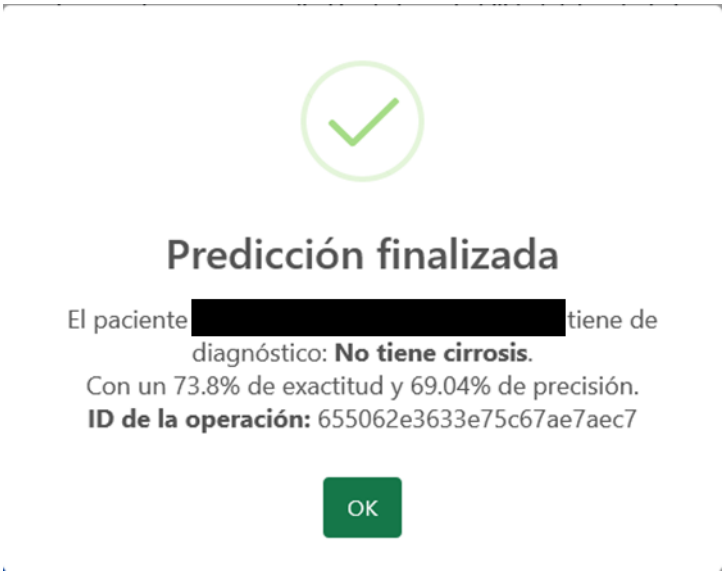


Figura 3.28 Predicción en modo normal o anónimo.

En la lista 5, se presenta una sección del código que hace posible enviar los biomarcadores al sistema (lín. 19) y desplegar en pantalla el resultado (líns. 31 en adelante).

```
1   async submit(){
    …
11  let body = Object.assign(this.datos, this.biomarcadores);
12  body._id_paciente = this.paciente._id;
13
14
15  body._id_padecimiento = this.padecimientos[this.padecimiento_idx]._id;
16
17  body.backg = this.backg;
18  body.anonym = this.anonym;
19  const res = await lastValueFrom
    (this.serv.procesarPrediccion(Object.assign(this.datos, this.biomarcadores)));
    …


31  await Swal.fire({
32  title: 'Predicción finalizada', icon: "success",
33  html: `El paciente <b>${(this.anonym ? `Anónimo (${res.nombre})`:
    `${this.paciente.nombre} ${this.paciente.ape_pat} ${this.paciente.ape_mat}`)}</b>
    tiene de diagnóstico: <b>${res.resultado}</b>.
    …
```

Lista 6: Fragmento de código del *submit* solicitar predicción.

3.3.4 Capa de presentación

Aunque previamente se muestra un poco de la interfaz gráfica mediante los mensajes recibidos dependiendo del tipo de predicción, en esta sección se aborda a detalle las interfaces gráficas ya desplegadas del presente sistema. Inicialmente, es necesario que el usuario se autentique para acceder a la funcionalidad de esta plataforma médica.

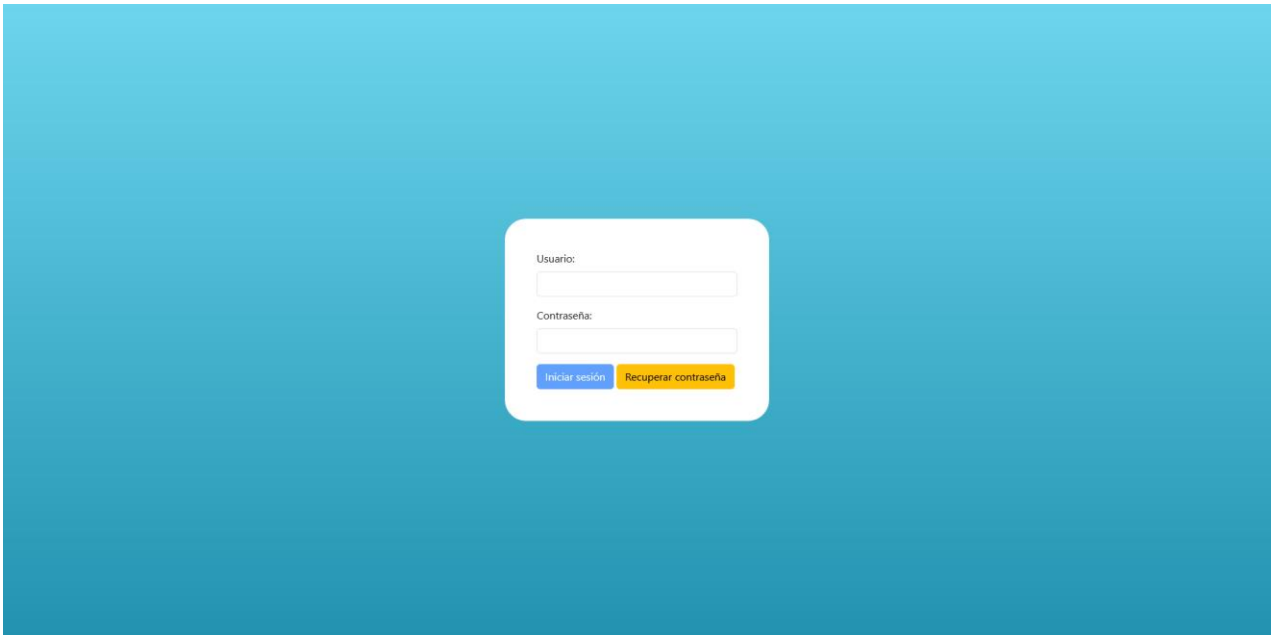


Figura 3.29 Interfaz de inicio de sesión

En caso de que el usuario haya olvidado o simplemente perdido el acceso al sistema, tiene como opción, solicitar un restablecimiento de contraseña, lo cual lo lleva a la interfaz de recuperar contraseña (que se muestra en la Fig. 3.30), donde primero especifica su correo electrónico, para que el sistema le envíe el método de recuperación.

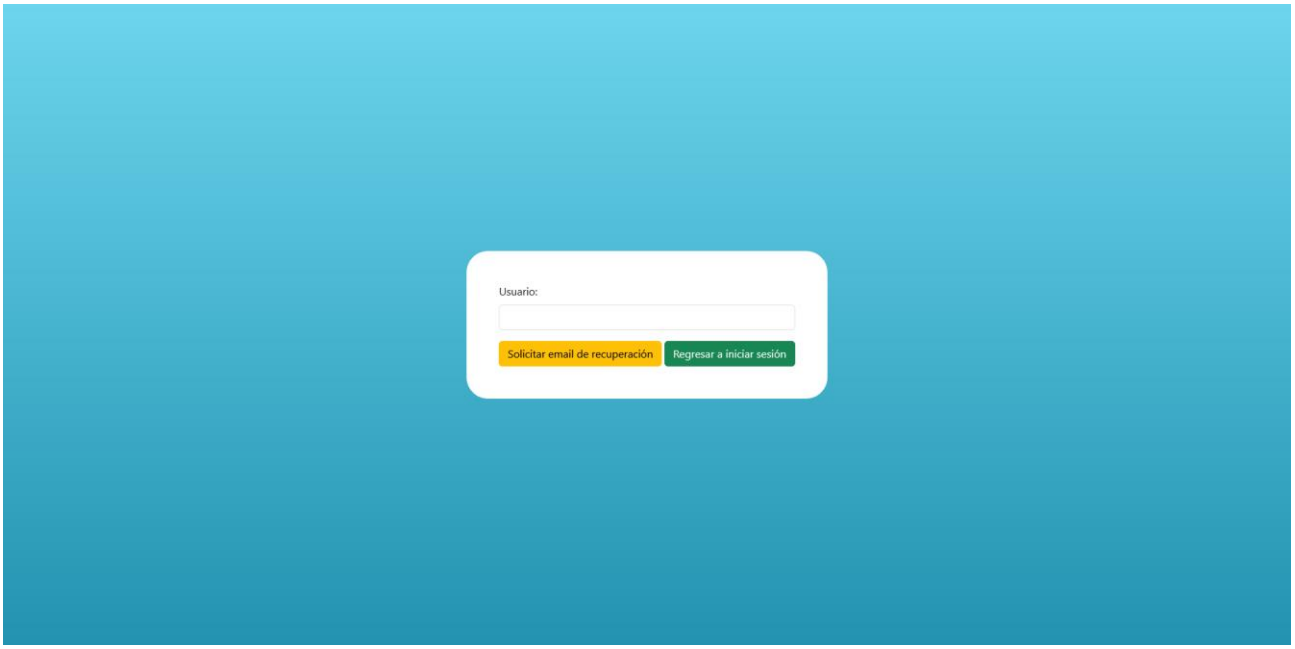


Figura 3.30 Interfaz de recuperación de contraseña (solicitud)

Una vez solicitada, en el correo electrónico del usuario, se le envía un enlace, el cual, al abrirlo, lo redirige a la misma interfaz, pero ahora con la posibilidad de cambiar su contraseña y así restablecer sus credenciales que se exhibe en Fig. 3.31.

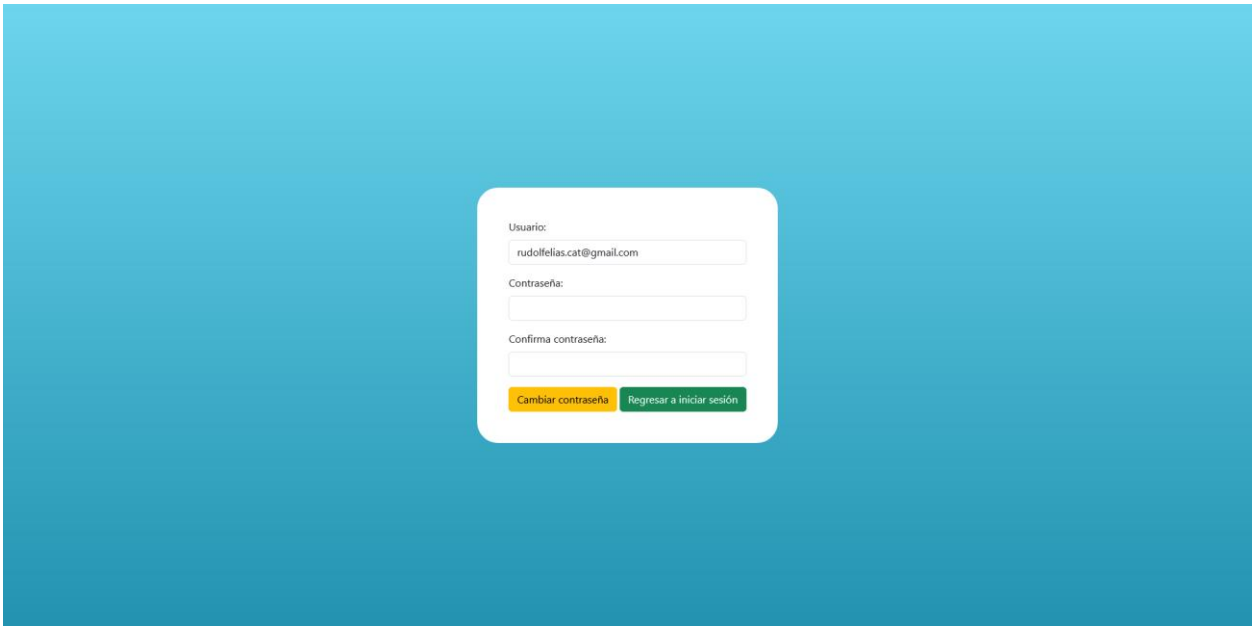


Figura 3.31 Interfaz de recuperación de contraseñas (validación lista)

Autenticado el usuario, lo primero que se le presenta es la interfaz del *dashboard* (presentado en la Fig. 3.32), el cual es un menú interactivo en el que se le permite ver las estadísticas en tiempo real de los pacientes del sistema, así como acceder a las opciones que proporciona la aplicación.

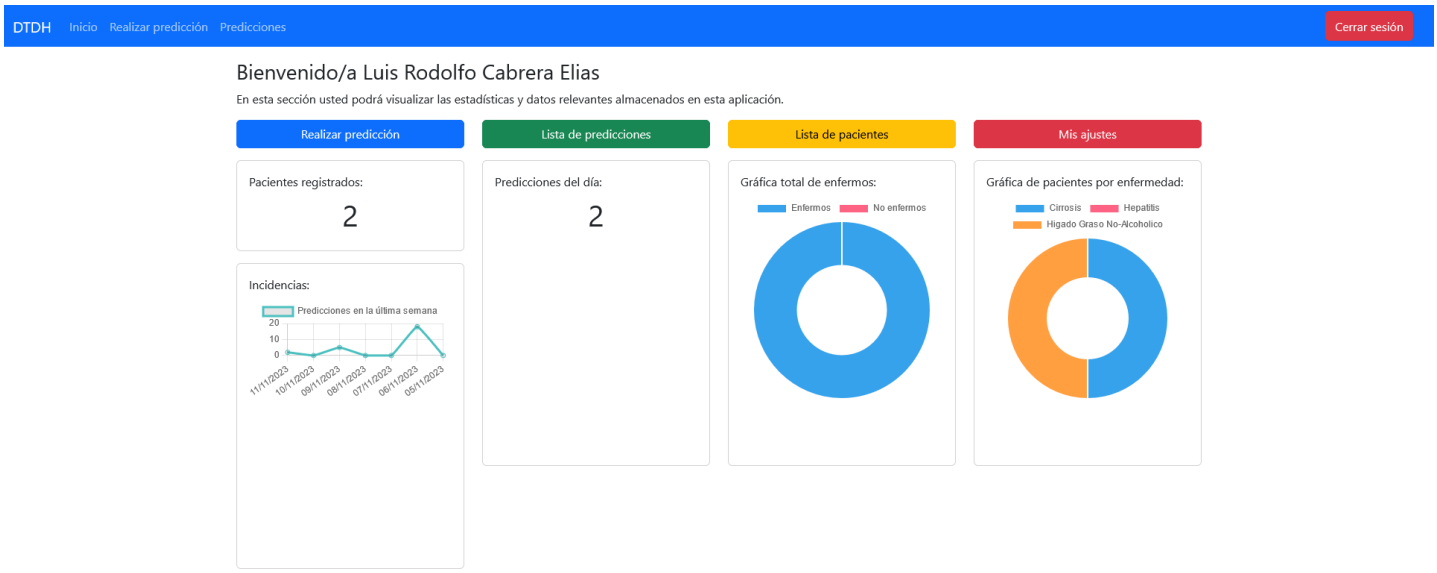


Figura 3.32 Interfaz de *dashboard*.

Si el usuario da clic en Realizar predicción, se redirige a dicho módulo y en este recae la principal funcionalidad de este sistema, el pronóstico de un trastorno hepático. En la figura 3.33 se muestra la interfaz de este módulo, con las opciones de la predicción, el desorden a predecir y los biomarcadores preponderantes que se le solicitan al médico. Si los campos se llenan correctamente, es posible llevar a cabo el pronóstico y se muestra el resultado en pantalla dependiendo de los ajustes seleccionados como en las Figs. 3.26 y 3.27.

Figura 3.33 Interfaz de Realizar predicción.

Igual los médicos de la aplicación tienen acceso a la lista de predicciones, la cual se exhibe en la figura 3.34, desde la que mediante algunos filtros tales como: **rango de fechas**, **médico que realizó la predicción**, **padecimiento** y **paciente**, obtener los registros almacenados previamente e incluso exportarlos a formatos de uso cotidiano tales así tal como: **pdf** y **xlsx.** Esto mediante las bibliotecas ReportLab y OpenPyXL de Python.

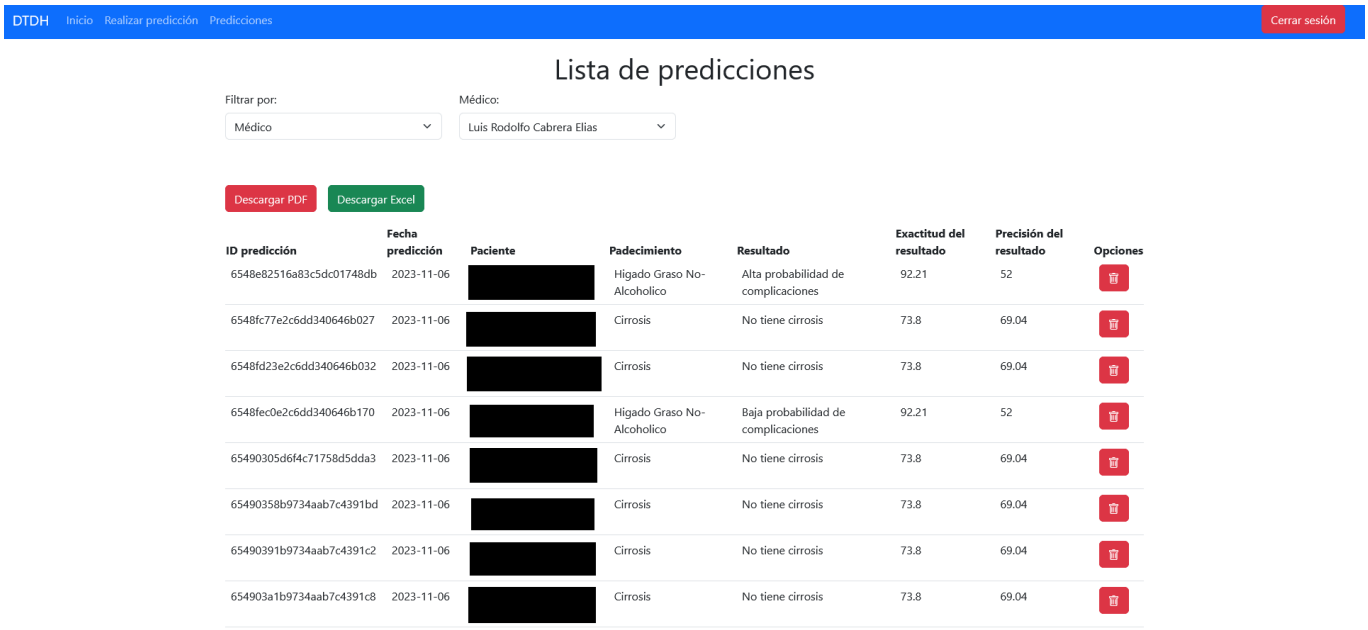


Figura 3.34 Interfaz de Lista de predicciones.

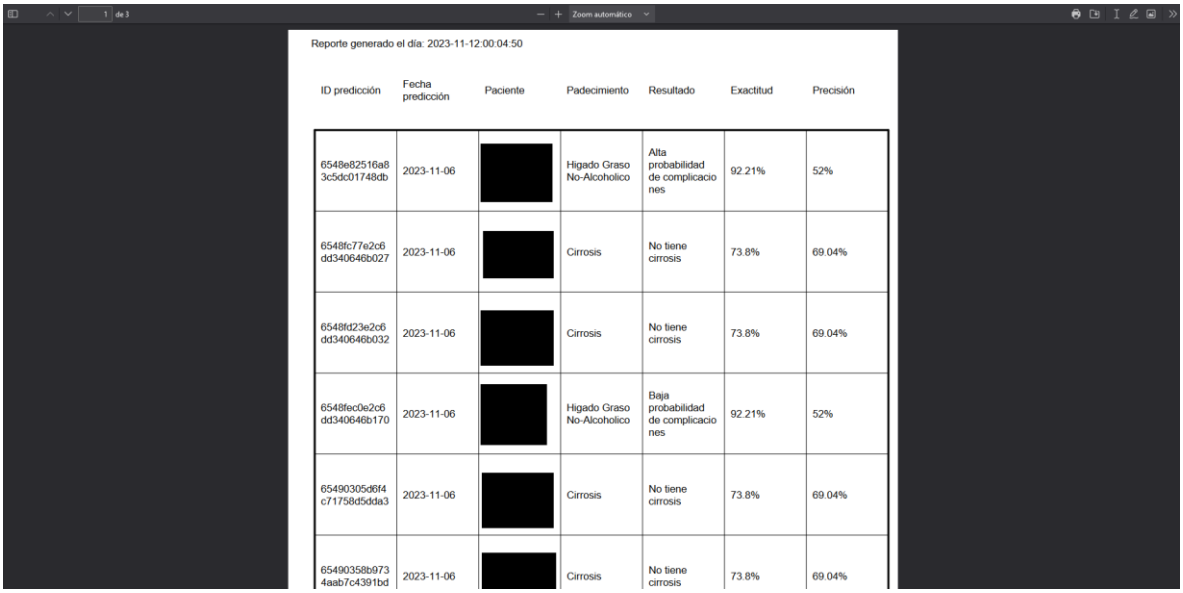
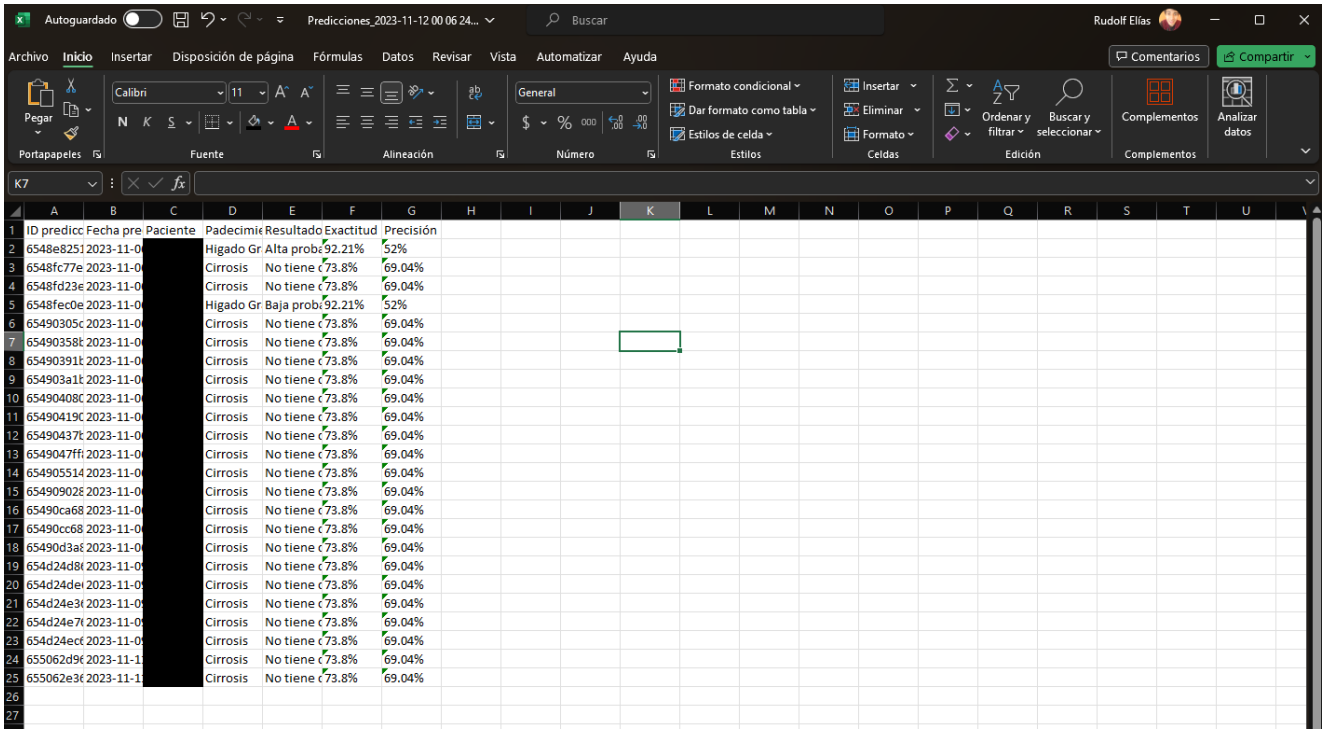


Figura 3.35 Archivo de formato PDF generado con los datos presentados en la Fig. 3.34.



	A	B	C	D	E	F	G
1	ID predicc	Fecha pre	Paciente	Padecimie	Resultado	Exactitud	Precisión
2	6548e825	2023-11-0		Higado Gr	Alta proba	92.21%	52%
3	6548fc77e	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
4	6548fd23e	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
5	6548fec0e	2023-11-0		Higado Gr	Baja proba	92.21%	52%
6	65490305c	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
7	65490358	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
8	65490391	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
9	654903a1	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
10	65490408	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
11	65490419	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
12	65490437	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
13	6549047ff	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
14	65490551	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
15	65490902	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
16	65490ca68	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
17	65490cc68	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
18	65490d3a	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
19	654d24d8	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
20	654d24de	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
21	654d24e3	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
22	654d24e7	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
23	654d24ec	2023-11-0		Cirrosis	No tiene C	73.8%	69.04%
24	655062d9	2023-11-1		Cirrosis	No tiene C	73.8%	69.04%
25	655062e3	2023-11-1		Cirrosis	No tiene C	73.8%	69.04%

Figura 3.36 Archivo de formato Excel generado con los datos exhibidos de la Fig. 3.34.

La interfaz mostrada a continuación, es de tipo ABC (Altas, Bajas, Consultas) de los usuarios de este sistema, esto incluye **Administradores**, **Médicos** y **Pacientes**.

Es importante destacar que: administradores modifican a ellos mismos, médicos y pacientes. Médicos únicamente modifican a pacientes.

Figura 3.37 Lista de usuarios.

En caso de que los listados sean Pacientes, al presionar el botón verde con el icono del ojo se accede a sus datos personales, lo cual incluye predicciones realizadas y almacenadas de dicha persona, tal como se visualiza en la Fig. 3.38.

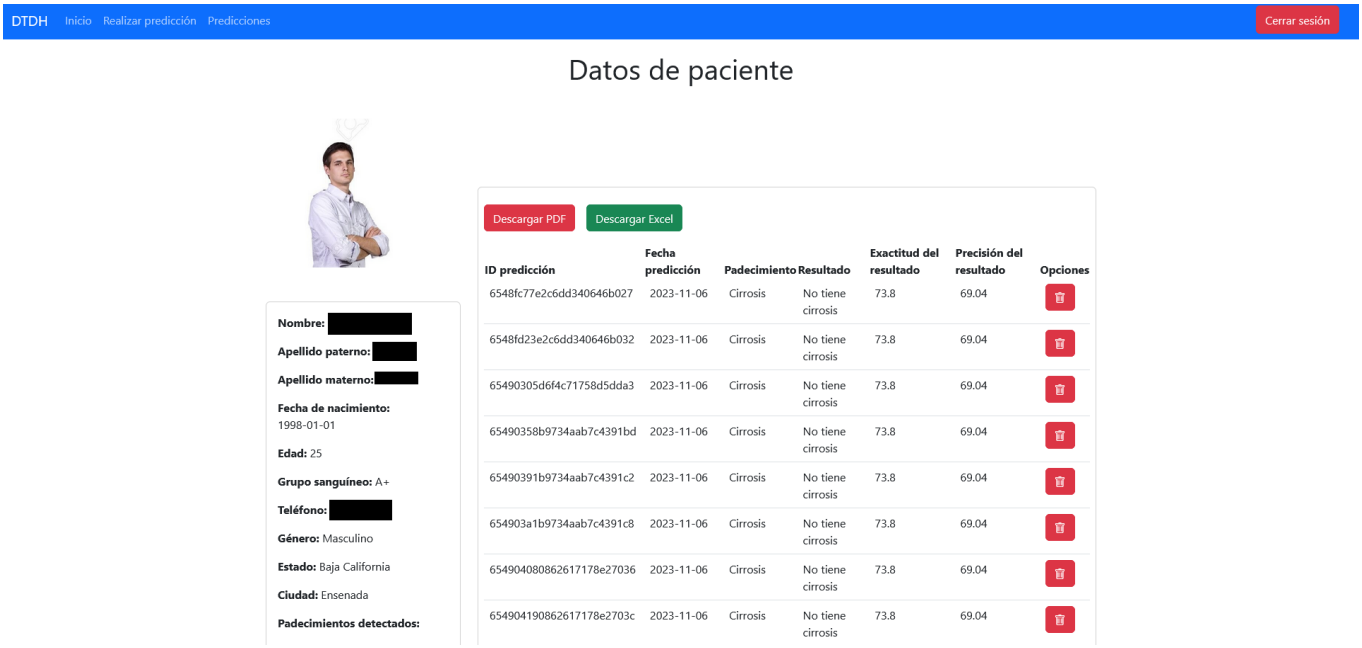


Figura 3.38 Datos de paciente.

Si en la Lista de usuarios, el médico o administrador en turno da clic en **Añadir usuario**, se envía a dicho módulo. Donde se da de alta a un **Paciente**, **Médico** o **Administrador**, siguiendo las mismas restricciones dependiendo del rol del usuario, tal cómo se especificó en **Lista de usuarios** (que se despliega en la Fig. 3.39).

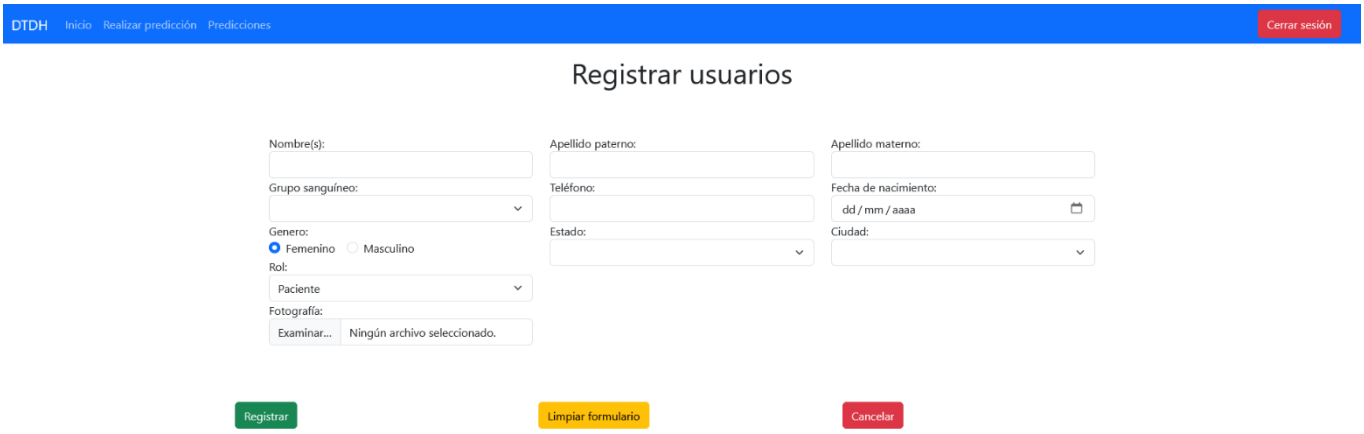


Figura 3.39 Registrar usuarios.

Finalmente, y no por menos importante se tiene la interfaz de ajustes, este módulo permite al usuario en turno del sistema modificar sus credenciales de acceso, pero con mucho más detalle a que si olvidó su contraseña. Permitiéndole cambiar su correo electrónico y contraseña directamente sin medidas de autenticación adicionales.

Figura 3.40 Mis ajustes.

Nota: Todos los datos mostrados en los reportes del punto 3.3 son datos artificiales generados únicamente con fines ilustrativos y los datos verdaderos se presentan en el siguiente capítulo en su subsección del Caso de estudio.

Por consiguiente, se llegó a la conclusión que Kanban es una metodología sencilla pero que cumple su cometido de dirigir un desarrollo tal como se mostró en todos los objetivos del presente capítulo. En el Capítulo IV se muestran los resultados obtenidos gracias al desarrollo exhibidos en este capítulo.

Capítulo 4. Resultados

Este capítulo tiene como propósito exhibir los resultados del presente trabajo de tesis, algunas modificaciones sugeridas por el Dr. Ángel Escudero Fabre del "Sanatorio Escudero" de Orizaba, Veracruz y el caso de estudio para satisfacer el problema propuesto en este trabajo de tesis y sus respectivos objetivos.

Este sistema proporciona una predicción de un trastorno hepático mediante el análisis de los biomarcadores de un paciente. Esto con el fin de proporcionar una herramienta de apoyo al momento de diagnosticar un desorden del hígado y tomar las medidas pertinentes.

4.1 Modificaciones a los modelos entrenados.

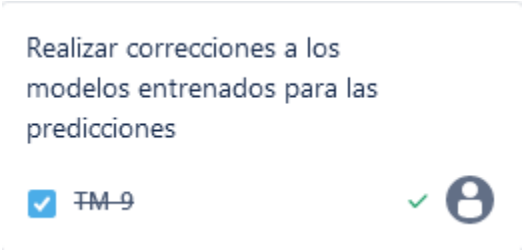


Figura 4.1 Tarjeta Kanban "Realizar correcciones a los modelos entrenados para las predicciones".

Para mejorar los resultados de las predicciones del actual proyecto de tesis, se realizaron una serie de correcciones a los modelos entrenados del Capítulo III. Dichas modificaciones consistieron en añadir algunos biomarcadores adicionales a los modelos de los desórdenes hepáticos. A continuación, las antes mencionadas actualizaciones se presentan en formato de tabla de correcciones.

4.1.1 Cirrohsis Prediction Dataset

Biomarcadores iniciales	**Biomarcadores propuestos**
• **Protrombina** • **Albumina** • **Plaquetas** • **Edad** • **Bilirrubina**	• Protrombina • Albumina • Plaquetas • Edad • Genero • Bilirrubina
Notas:	Se añadió género.

Tabla 4.1 Tabla de correcciones de Cirrohsis Prediction Dataset.

4.1.2 Indian Liver Patient Dataset

Biomarcadores iniciales	**Biomarcadores propuestos**
• **Fosfatasa alcalina** • **Aspartato Aminotransferasa (AST)** • **Alanina aminotransferasa (ALT)** • **Edad** • **Total de proteínas**	• Fosfatasa alcalina • Aspartato Aminotransferasa (AST) • Alanina aminotransferasa (ALT) • Edad • Genero • Total de proteínas • Bilirrubina • Albúmina
Notas:	Se añadió género, bilirrubina y albúmina.

Tabla 4.2 Tabla de correcciones de Indian Liver Patient Dataset.

4.1.3 NAFLD

Biomarcadores iniciales	Biomarcadores propuestos
● **Edad** ● **Índice de masa muscular** ● **Peso** ● **Altura** ● **Género**	● Edad ● Índice de masa muscular ● Peso ● Altura ● Género
Notas:	No se hicieron cambios, pero se pedirá que únicamente se utilice si el paciente fue previamente diagnosticado con EHGNA por un médico especialista.

Tabla 4.3 Tabla de correcciones de NAFLD.

4.1.4 Eliminación del diagnóstico del conjunto de datos BUPA Liver Disorders Dataset.

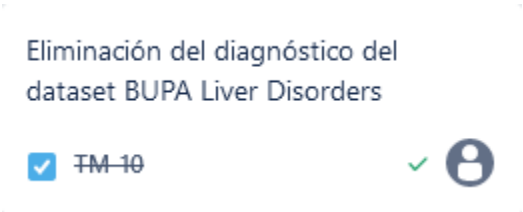


Figura 4.2 Tarjeta Kanban "Eliminación del diagnóstico del *dataset* BUPA Liver Disorders".

La eliminación de este conjunto de datos se justifica principalmente debido a una interpretación errónea y generalizada del séptimo campo (selector). En el pasado, este campo se malinterpretó ampliamente como una variable dependiente que representaba la presencia o ausencia de un trastorno hepático [57]. Sin embargo, esta interpretación es incorrecta, y se aclara que el séptimo campo se creó por investigadores de BUPA con el propósito de servir como un selector de entrenamiento/prueba, y no como una variable adecuada para la clasificación basada en la presencia o ausencia de un trastorno hepático.

Es importante destacar que el conjunto de datos en cuestión no contiene ninguna variable que represente de manera precisa la presencia o ausencia de un trastorno hepático. La confusión en torno al séptimo campo llevó a malentendidos significativos y a la aplicación incorrecta del conjunto de datos en investigaciones y análisis que buscan abordar cuestiones relacionadas con la salud hepática.

Por lo tanto, la eliminación de este conjunto de datos se basa en la necesidad de evitar la propagación de información incorrecta y la realización de análisis erróneos que surgieron de la interpretación inapropiada del séptimo campo como una variable dependiente relevante para la clasificación de trastornos hepáticos.

4.2 Caso de estudio

Para el presente trabajo de tesis se adquirieron los datos y se realizaron pruebas en la institución de salud “Sanatorio Escudero” el cual tiene su inmueble ubicado en Poniente 7 # 905, Orizaba, Centro, Veracruz. El sanatorio provee los siguientes servicios médicos [59].

Generales:

- Hospitalización
- Quirófano con arco en C, Microscopio y Laparoscopia
- Área de recuperación y cuarto de choque
- Cuneros
- Sala de urgencias las 24 horas
- Rayos X
- Tomografías
- Ultrasonido convencional
- Mastografía
- Colposcopía

- Láser para angiología
- Unidad de video endoscopía
- Electromiografía
- Laboratorio de análisis clínicos
- Farmacia las 24 horas
- Convenios con aseguradoras

Especialidades:

- Ginecología y Obstetricia
- Gastroenterología
- Cirugía general y laparoscopía
- Angiología y cirugía vascular
- Traumatología y Ortopedia
- Cardiología y ecocardiografía
- Neurología y Neurocirugía
- Odontología
- Medicina familiar
- Medicina física y rehabilitación
- Oftalmología
- Pediatría
- Urología
- Nutriología

- Otorrinolaringología
- Dermatología y Alergología
- Oncología
- Nefrología

Los datos de los análisis se consiguieron desde finales de octubre de 2023 debido a las mejoras que se hicieron a los modelos de predicción del presente trabajo. Se utilizan los datos de los biomarcadores de un paciente por cada uno de los tres desórdenes hepáticos. Omitiendo los datos sensibles y privados, los cuales son reemplazados por su nomenclatura de paciente anónimo, tal como se especificó en el Capítulo III en el punto 3.3.3.

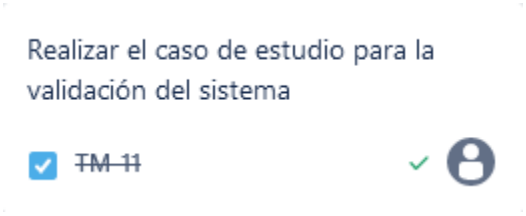


Figura 4.3 Tarjeta Kanban "Realizar el caso de estudio para la validación del sistema".

Para este caso de estudio se utilizaron datos de tres pacientes con los siguientes padecimientos: cirrosis, hepatitis y EGHNA respectivamente. Los datos con biomarcadores de los pacientes son los presentados a continuación:

Paciente con Cirrosis					
Protrombina	**Albúmina**	**Plaquetas**	**Edad**	**Género**	**Bilirrubina**
14.1	3.4	90	64	F	1.8

Tabla 4.4 Datos de paciente con Cirrosis.

Paciente con Hepatitis							
Fosfatasa alcalina	**Aspartato aminotransferasa**	**Alanina aminotransferasa**	**Edad**	**Género**	**Total de proteínas**	**Bilirrubina**	**Albúmina**
636	450	300	45	M	7.3	2.1	4.1

Tabla 4.5 Datos de paciente con Hepatitis.

Paciente con EHGNA				
Edad	**IMC**	**Peso**	**Altura**	**Género**
50	32.84	86.19	162 cm	F

Tabla 4.6 Datos de paciente con EHGNA.

4.2.1 Paciente con Cirrosis

Para determinar que el **Paciente anónimo 1** tiene cirrosis se llena el formulario con los biomarcadores presentados en la Tabla 4.4 tal como se exhibe a continuación:

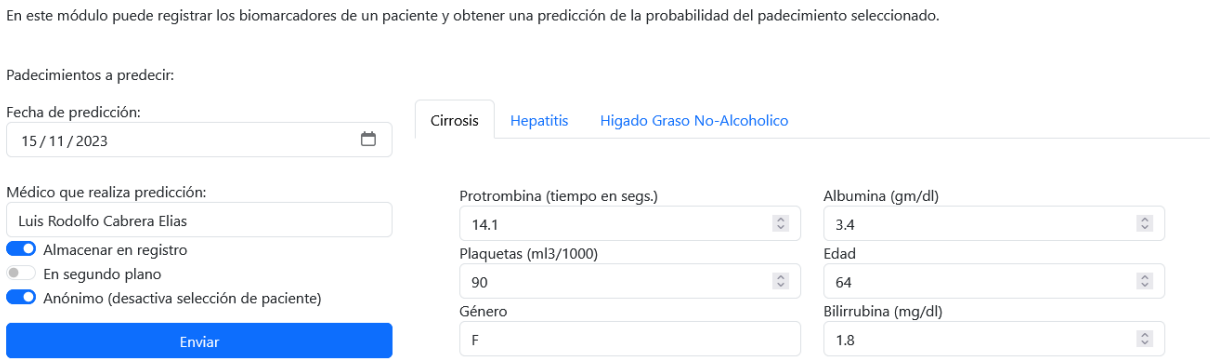


Figura 4.4 Llenado de formulario de Paciente 1.

Finalmente se obtuvo el resultado de predicción en pantalla.

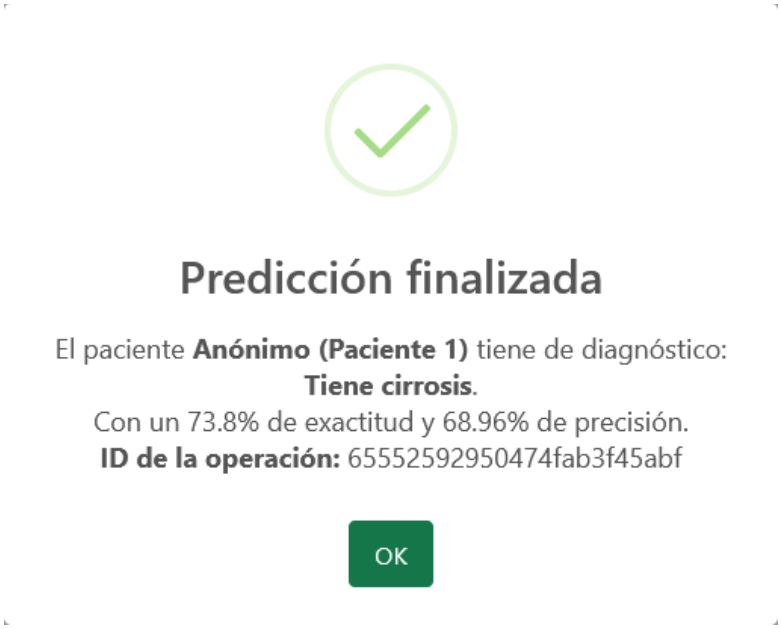


Figura 4.5 Resultado de predicción de Paciente 1.

Se concluye que el paciente tiene cirrosis y el algoritmo utilizado para la predicción obtuvo un **73.8%** de exactitud y **68.96%** de precisión.

Nota: Cabe recalcar que la **precisión** evalúa la coherencia y la dispersión de los resultados y la **exactitud** la proximidad de los resultados al valor real o correcto.

4.2.2 Paciente con Hepatitis

Para establecer que el **Paciente anónimo 2** padece hepatitis, al igual que en punto 4.2.1, se envía el formulario con los biomarcadores mostrados en la Tabla 4.5 tal como se presenta en la Figura 4.6:

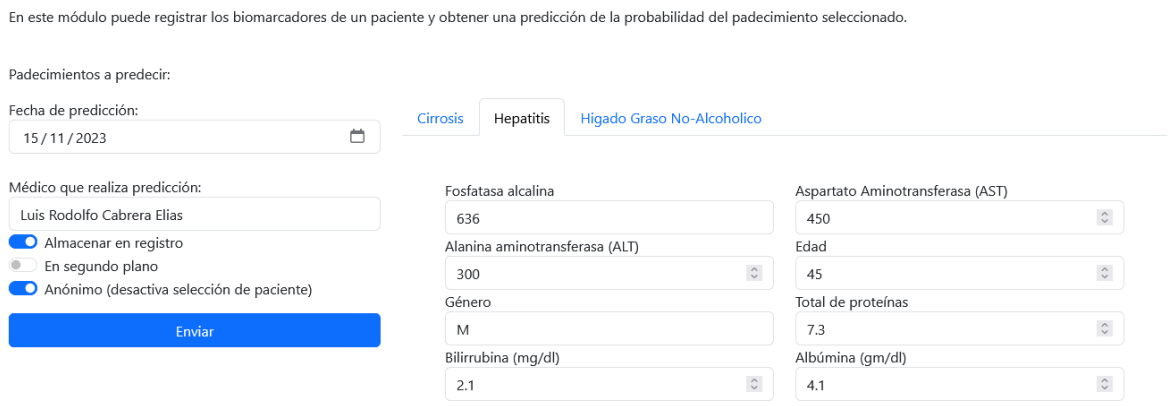


Figura 4.6 Llenado de formulario de Paciente 2.

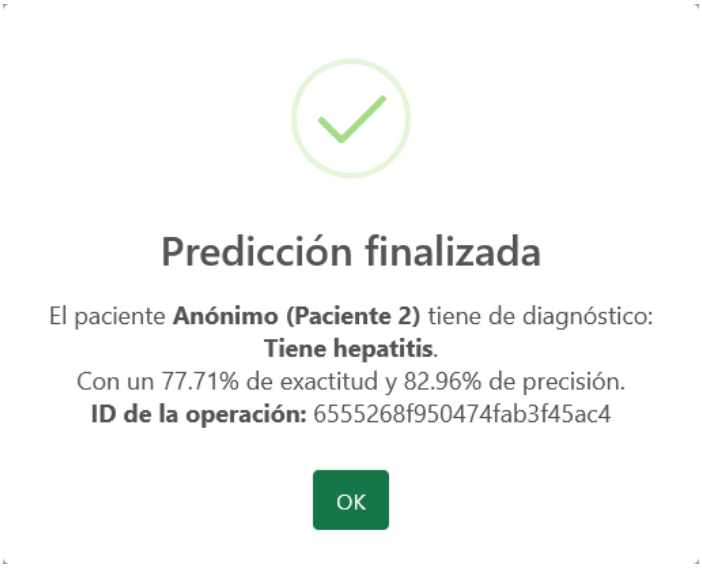


Figura 4.7 Resultado de predicción de Paciente 2.

Se concluye que el paciente tiene hepatitis y el algoritmo manejado en la predicción obtuvo un **77.71%** de exactitud y **82.96%** de precisión.

4.2.3 Paciente con EHGNA

Para concluir el caso de estudio que abarca el presente capítulo, el **Paciente anónimo 3** tiene una baja probabilidad de padecer complicaciones a largo plazo después de que se diagnosticó con EGHNA. Así como se presentó en los dos puntos anteriores, se llena el formulario con los biomarcadores expuestos en la Tabla 4.6 tal como se enseña a continuación:

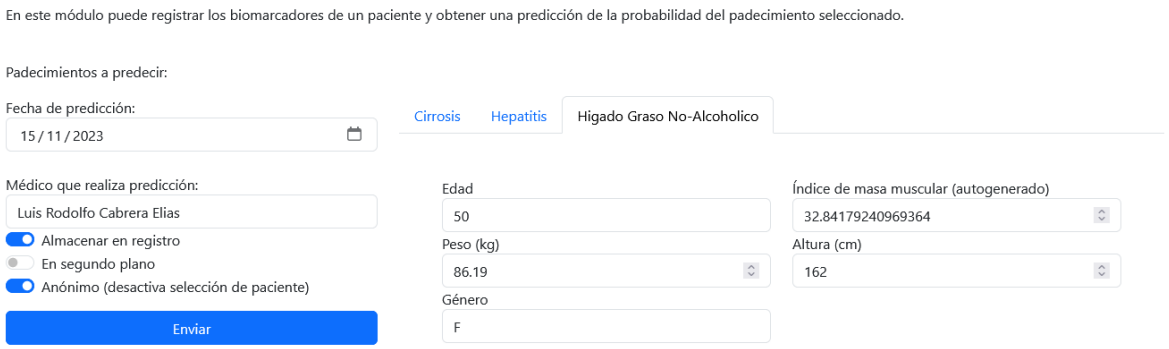


Figura 4.8 Llenado de formulario de Paciente 3.

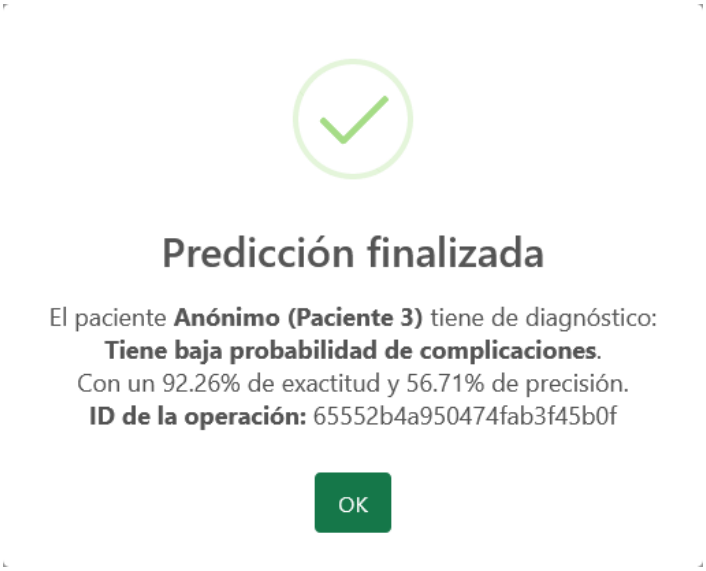


Figura 4.9 Resultado de predicción de Paciente 3.

Se concluye que el paciente tiene baja probabilidad de contraer complicaciones a futuro con una predicción que obtuvo un **92.26%** de exactitud y **56.71%** de precisión.

4.2.4 Exportado de datos

El presente módulo está provisto con la facultad de listar y convertir los resultados previamente obtenidos con el fin de llevar registros médicos. En esta subsección se exhibe sus diferentes formas de presentación:

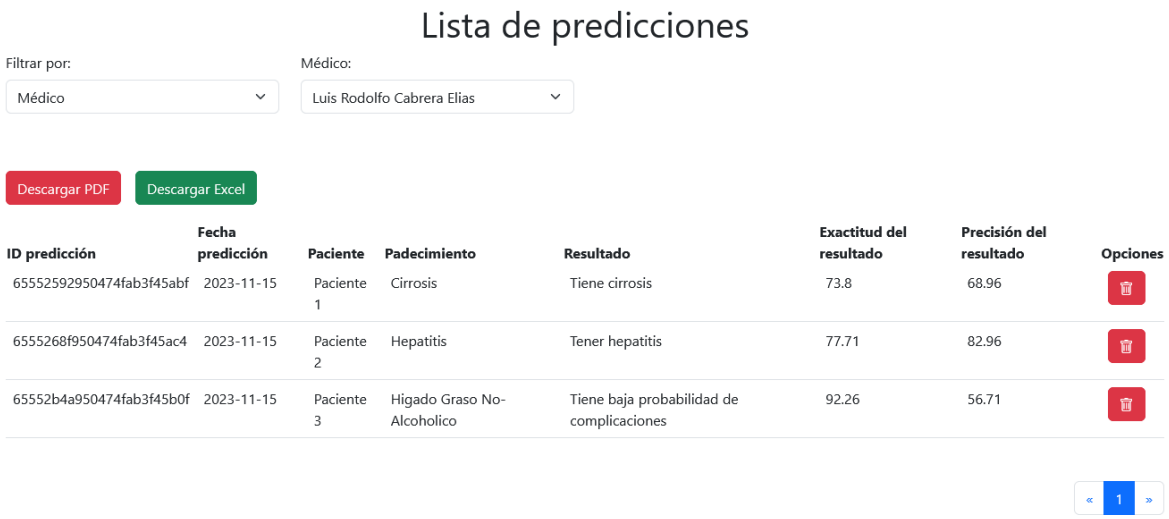


Figura 4.10 Presentación de datos en Interfaz Web.

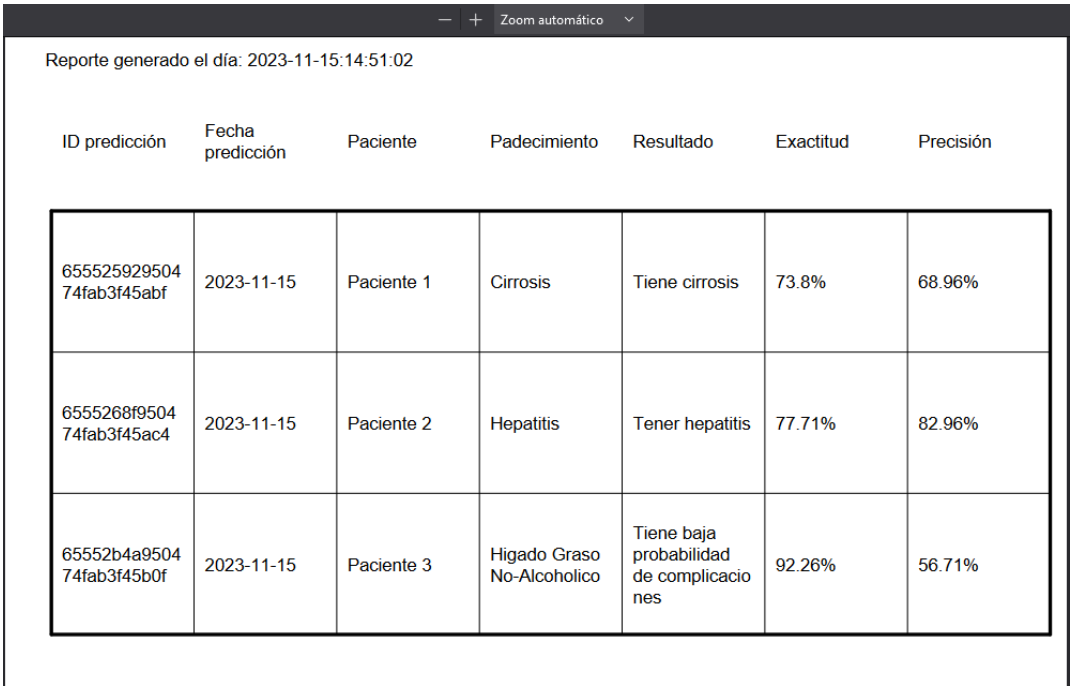


Figura 4.11 Presentación de datos en Archivo PDF.

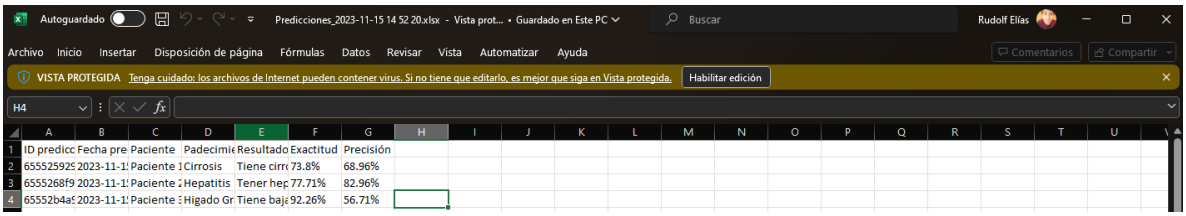


Figura 4.12 Presentación de datos en Archivo XLSX (Excel)

Para dar fin a esta subsección y capítulo, se presentan a continuación los datos del *dashboard* con las predicciones realizadas.

Nota: Aunque no hay pacientes dados de alta dentro del sistema, los registros anónimos cuentan como si lo fueran, entonces se toman en cuenta al momento de llenar las gráficas de la interfaz que se exhibe a continuación.

Figura 4.13 *Dashboard* del usuario.

Capítulo 5. Conclusiones y recomendaciones

En este capítulo se exhiben las conclusiones y recomendaciones del presente trabajo de tesis con nombre “Detección temprana de desórdenes hepáticos a partir del análisis de biomarcadores utilizando métodos de ensambles de aprendizaje automático”.

5.1 Conclusiones

Para el presente trabajo de tesis se finalizaron todos los objetivos planteados inicialmente con el desarrollo del módulo “Detección temprana de desórdenes hepáticos a partir del análisis de biomarcadores utilizando métodos de ensambles de aprendizaje automático”.

Para realizar las predicciones se utilizaron tres conjuntos de datos los cuales fueron: Cirrhosis Prediction Dataset para el diagnóstico de cirrosis, hepatitis mediante ILPD (Indian Liver Patient Dataset) y Non-alcohol fatty liver disease para determinar las complicaciones a futuro de un paciente diagnosticado con EHGNA.

Los mejores biomarcadores utilizados para predecir los principales padecimientos abordados en este trabajo fueron:

Para la cirrosis son los siguientes: Protrombina, Albúmina. Plaquetas, Edad, Género y Bilirrubina. Sus mejores clasificadores son listados a continuación: DT, LR, GradientBoosting y Adaboost.

En el caso de la hepatitis se muestran a continuación: Fosfatasa alcalina, Aspartato Aminotransferasa, Alanina Aminotransferasa, Edad, Género, Total de proteínas, Bilirrubina y Albúmina. Sus clasificadores base más precisos que se exhiben son los siguientes: RF, DT, LightGBM y GradientBoosting.

Finalmente, para EHGNA son: Edad, IMC, Peso, Altura y Género siempre y cuando el paciente se diagnosticó con dicha enfermedad. Sus mejores clasificadores base que son usados son: DT, LR, AdaBoost y LightGBM.

El presente módulo busca ser una herramienta de apoyo al momento de detectar desórdenes hepáticos de una manera preliminar y otorgar a los pacientes el tratamiento médico oportuno y mejorar sus posibilidades de supervivencia.

5.2 Recomendaciones

El uso de los algoritmos de aprendizaje automático es relevante para todas las áreas de estudio, sobre todo en el campo de la medicina. Pero cabe recalcar que, al ser un enfoque relativamente nuevo, aún queda mucho por afinar, es por eso que enumeran algunas recomendaciones para futuros trabajos a continuación:

- Hay que considerar que las poblaciones tienen diferencias genéticas, esto quiere decir que hay variaciones con los diagnósticos, por ejemplo: de una persona europea y una asiática.
- Para el punto anterior, se opta por realizar más conjuntos de datos de diferentes regiones o del área donde se quiere aplicar el caso de estudio, aunque claro, esto es una tarea muy ardua.
- Implementar más algoritmos clasificadores base
- Abordar más enfermedades y/o desórdenes hepáticos
- Incluir más variables en el proceso de detección de enfermedades hepáticas.

Las recomendaciones anteriores surgieron a las observaciones durante los casos de estudio y durante un congreso donde se expuso el presente trabajo.

Productos académicos

PDF de artículo

ISSN 1870-4069

Arquitectura de un módulo para la identificación de factores de riesgo para la detección de desórdenes hepáticos a partir del análisis de biomarcadores utilizando métodos de ensambles de aprendizaje automático

Luis Rodolfo Cabrera-Elías[1], José Luis Sánchez-Cervantes[2], Giner Alor-Hernández[1], Beatriz Alejandra Olivares-Zepahua[1], Luis Ángel Reyes-Hernández[1]

[1] Tecnológico Nacional de México
Instituto Tecnológico de Orizaba
México

[2] Consejo Nacional de Ciencia y Tecnología,
Tecnológico Nacional de México,
México

{m16011057, jose.sc, giner.ah, luis.rh}@orizaba.tecnm.mx,
bolivares@ito-depi.edu.mx

Resumen. La salud es importante para llevar una vida estable y duradera, pero la falta de cuidado aumenta el riesgo de enfermedades y dañar órganos, especialmente el hígado. En este trabajo se presenta la arquitectura de un módulo de aprendizaje automático basado en ensambles para identificar factores de riesgo en la detección de desórdenes hepáticos a través del análisis de biomarcadores. Además, se presenta el análisis de los trabajos relacionados con el tema y se examinan algoritmos de ensamble de aprendizaje automático, como bagging, boosting, para la detección de desórdenes hepáticos. Como trabajo a futuro se identificarán los principales biomarcadores para la detección de factores de riesgo, así como se diseñará un modelo de entrenamiento basado en el algoritmo de ensamble para la identificación de factores de riesgo. Finalmente, el módulo se entrenará y se integrará con interfaces Web y un repositorio de información.

Palabras clave: Biomarcadores, daño hepático, ensambles de aprendizaje automático.

Architecture of a module for identification of risk factors for the detection of liver disorders from biomarker analysis using ensemble machine learning method

Abstract. Health is important to lead a stable and long-lasting life, but lack of care increases the risk of diseases and damaging organs, especially the liver. This work presents the architecture of a machine learning module based on ensembles

Reconocimiento de presentación de artículo

COMIA 2023

Cinvestav Guadalajara

La Sociedad Mexicana de Inteligencia Artificial (SMIA) y el Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional (CINVESTAV) unidad Guadalajara

OTORGAN EL PRESENTE CERTIFICADO A:

Luis Rodolfo Cabrera Elías, José Luis Sanchez, Giner Alor Hernandez, Beatriz Olivares Zepahua, Luis Angel Reyes Hernandez

por la presentación del artículo:

Arquitectura de un módulo para la identificación de factores de riesgo para la detección de desórdenes hepáticos a partir del análisis de biomarcadores utilizando métodos de ensambles de aprendizaje automático

durante el XV Congreso Mexicano de Inteligencia Artificial - COMIA 2023
Guadalajara, Jalisco, México, del 30 de mayo al 3 de junio de 2023

Sociedad Mexicana de Inteligencia Artificial, A.C. www.smia.org.mx RFC SMI871231FB7

Dr. Hiram Calvo, Presidente SMIA
Dr. Gilberto Ochoa Ruiz, Presidente Comité de Programa
Dra. Iris I. Méndez Gurrola, Presidente Comité de Programa
Dr. Nestor Velasco Bermeo, Presidente Comité de Programa
Dr. Andrés Méndez Vázquez, Comité Local COMIA

Vídeo y póster

Detección temprana de desórdenes hepáticos a partir del análisis de biomarcadores utilizando métodos de ensambles de aprendizaje automático.

Luis Rodolfo Cabrera-Elías, José Luis Sánchez-Cervantes, Giner Alor-Hernández, Beatriz Alejandra Olivares-Zepahua, Luis Ángel Reyes-Hernández
Maestría en Sistemas Computacionales; División de Estudios de Posgrado e Investigación, Instituto Tecnológico de Orizaba

INTRODUCCIÓN

El hígado es uno de los órganos más importantes del cuerpo humano que se encarga de controlar los niveles químicos de la sangre, además, ayuda a descomponer las grasas para su digestión y absorción, así como también metaboliza los medicamentos que se encuentran en la sangre para que sean más fáciles de utilizar por el cuerpo. El principal problema que se observa en esta investigación es que existen varios trastornos del hígado que necesitan tratamiento temprano por parte de los médicos, tal es el caso de, el daño hepático causado por obesidad, o hepatotoxicidad generada por el consumo de medicamentos, drogas o incluso remedios caseros. Este trabajo proporciona una alternativa de solución a la problemática planteada haciendo uso de algoritmos de ensambles de aprendizaje automático, de tal manera que, permita la pronta detección de desórdenes hepáticos, para así conocer las probabilidades de que una persona contraiga alguna enfermedad o cualquier otro problema de salud relacionada con el hígado y sea detectado en tiempo y forma a fin de tomar las medidas preventivas.

OBJETIVO

Desarrollar un módulo basado en algoritmos de ensambles de aprendizaje automático que permita la detección temprana de desórdenes hepáticos tales como: cirrosis, daño hepático en general e hígado graso no-alcohólico, a partir del análisis de biomarcadores.

RESULTADOS

La aplicación web que realiza el proceso de detección temprana de desórdenes hepáticos a partir del análisis de biomarcadores utilizando métodos de ensambles de aprendizaje automático: es una aplicación web que permite predecir desórdenes hepáticos de un paciente mediante el análisis de biomarcadores, haciendo uso de algoritmos de métodos de ensambles de aprendizaje automático con enfoques de *bagging* y *boosting* complementados con varios clasificadores base. Esto permite al módulo predecir si un paciente tiene o no un desorden hepático.

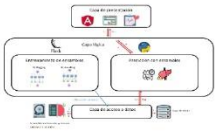


Figura 1 Flujo de trabajo de detección de desórdenes hepáticos

La Figura 1 muestra el flujo de trabajo que se lleva a cabo al momento de solicitar una predicción de un desorden hepático, el cual es: **Lectura de biomarcadores** en la interfaz gráfica, **Recepción de la lectura** en la capa lógica, y finalmente el **Uso del ensamble entrenado** con anterioridad y almacenado en la capa de acceso a datos para realizar una predicción con la información recibida y la obtención del resultado. Opcionalmente se almacenaje del resultado en la base de datos de la aplicación para generar un historial del paciente. La interfaz principal (*dashboard*) se exhibe en la Figura 2.

Figura 2 Vista inicial de la plataforma

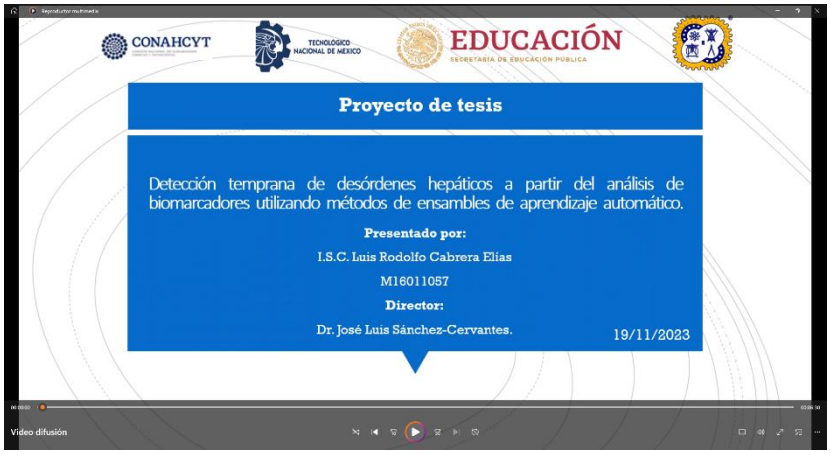
Posteriormente se realizó un caso de estudio como prueba de concepto para evaluar el desempeño de esta aplicación web. Se utilizaron 3 pacientes, los cuales poseen los padecimientos de: **cirrosis**, **hepatitis**, **EGHNA** respectivamente. La aplicación potenciada con los modelos entrenados pudo predecir satisfactoriamente los padecimientos de los pacientes antes mencionados con tasas aceptables de exactitud y precisión.

Figura 3 Predicciones satisfactorias de los pacientes del caso de estudio.

CONCLUSIONES

En este trabajo, se presentó una aplicación web, qué mediante el uso de algoritmos de métodos de ensamble automático es capaz de diagnósticas desordenes hepáticos con el uso de diferentes clasificadores base. Siendo así una herramienta de apoyo para proporcionar a los pacientes un tratamiento oportuno y mejorar su calidad de vida.

Luis Rodolfo Cabrera-Elías, José Luis Sánchez-Cervantes, Giner Alor-Hernández, Beatriz Alejandra Olivares-Zepahua, Luis Ángel Reyes-Hernández, (2023) Detección temprana de desórdenes hepáticos a partir del análisis de biomarcadores utilizando métodos de ensambles de aprendizaje automático.



Referencias

[1] "Liver (Anatomy): Picture, Function, Conditions, Tests, Treatments". Consultado: el 20 de octubre de 2022. [En línea]. Disponible en: https://www.webmd.com/digestive-disorders/picture-of-the-liver

[2] Tejada F., "Hepatotoxicidad por Fármacos", 2010, Consultado: el 20 de octubre de 2022. [En línea]. Disponible en: https://scielo.isciii.es/pdf/albacete/v3n3/especial1.pdf

[3] R. Bernal-Reyes *et al.*, "Consenso mexicano de la enfermedad por hígado graso no alcohólico", *Rev Gastroenterol Mex*, vol. 84, núm. 1, pp. 69–99, ene. 2019, doi: 10.1016/J.RGMX.2018.11.007.

[4] P. Ginès, A. Krag, J. G. Abraldes, E. Solà, N. Fabrellas, y P. S. Kamath, "Liver cirrhosis", *The Lancet*, vol. 398, núm. 10308, pp. 1359–1376, oct. 2021, doi: 10.1016/S0140-6736(21)01374-X.

[5] "Hepatitis: MedlinePlus en español". Consultado: el 14 de noviembre de 2023. [En línea]. Disponible en: https://medlineplus.gov/spanish/hepatitis.html

[6] "What is a Biomarker?" Consultado: el 20 de octubre de 2022. [En línea]. Disponible en: https://www.news-medical.net/health/What-is-a-Biomarker.aspx

[7] Amisha, P. Malik, M. Pathania, y V. K. Rathaur, "Overview of artificial intelligence in medicine", *J Family Med Prim Care*, vol. 8, núm. 7, p. 2328, 2019, doi: 10.4103/JFMPC.JFMPC_440_19.

[8] "Healthcare Remains The Hottest AI Category For Deals". Consultado: el 20 de octubre de 2022. [En línea]. Disponible en: https://www.cbinsights.com/research/artificial-intelligence-healthcare-startups-investors/

[9] P. Y. Taser, "Application of Bagging and Boosting Approaches Using Decision Tree-Based Algorithms in Diabetes Risk Prediction", MDPI AG, mar. 2021, p. 6. doi: 10.3390/proceedings2021074006.

[10] P. Cunningham, M. Cord, y S. J. Delany, "Supervised learning", *Cognitive Technologies*, pp. 21–49, 2008, doi: 10.1007/978-3-540-75171-7_2/COVER.

[11] "Aprendizaje supervisado y no supervisado - healthdataminer.com". Consultado: el 19 de mayo de 2023. [En línea]. Disponible en: https://healthdataminer.com/data-mining/aprendizaje-supervisado-y-no-supervisado/

[12] "Regresión Logística - Documentación de IBM". Consultado: el 20 de octubre de 2022. [En línea]. Disponible en: https://www.ibm.com/docs/es/spss-statistics/saas?topic=regression-logistic

[13] A. J. Smola y B. Schölkopf, "A tutorial on support vector regression", *Stat Comput*, vol. 14, núm. 3, pp. 199–222, ago. 2004, doi: 10.1023/B:STCO.0000035301.49549.88.

[14] “Análisis de vecinos más cercanos - Documentación de IBM”. Consultado: el 20 de octubre de 2022. [En línea]. Disponible en: https://www.ibm.com/docs/es/spss-statistics/saas?topic=features-nearest-neighbor-analysis

[15] “1.10. Decision Trees — scikit-learn 1.1.2 documentation”. Consultado: el 20 de octubre de 2022. [En línea]. Disponible en: https://scikit-learn.org/stable/modules/tree.html

[16] A. Cutler, D. R. Cutler, y J. R. Stevens, “Random Forests”, *Ensemble Machine Learning*, pp. 157–175, 2012, doi: 10.1007/978-1-4419-9326-7_5.

[17] X. Dong, Z. Yu, W. Cao, Y. Shi, y Q. Ma, “A survey on ensemble learning”, *Frontiers of Computer Science 2019 14:2*, vol. 14, núm. 2, pp. 241–258, ago. 2019, doi: 10.1007/S11704-019-8208-Z.

[18] “What is Bagging? | IBM”. Consultado: el 20 de octubre de 2022. [En línea]. Disponible en: https://www.ibm.com/cloud/learn/bagging

[19] “AdaBoost Algorithm - A Complete Guide for Beginners - Analytics Vidhya”. Consultado: el 20 de octubre de 2022. [En línea]. Disponible en: https://www.analyticsvidhya.com/blog/2021/09/adaboost-algorithm-a-complete-guide-for-beginners/

[20] “Árboles de decisión con boosting del gradiente | Machine Learning | Google Developers”. Consultado: el 20 de octubre de 2022. [En línea]. Disponible en: https://developers.google.com/machine-learning/decision-forests/intro-to-gbdt

[21] “Welcome to LightGBM’s documentation! — LightGBM 3.3.2 documentation”. Consultado: el 20 de octubre de 2022. [En línea]. Disponible en: https://lightgbm.readthedocs.io/en/v3.3.2/

[22] A. Verma y S. Mehta, “A comparative study of ensemble learning methods for classification in bioinformatics”, en *Proceedings of the 7th International Conference Confluence 2017 on Cloud Computing, Data Science and Engineering*, Institute of Electrical and Electronics Engineers Inc., jun. 2017, pp. 155–158. doi: 10.1109/CONFLUENCE.2017.7943141.

[23] “Validación cruzada - Amazon Machine Learning”. Consultado: el 19 de mayo de 2023. [En línea]. Disponible en: https://docs.aws.amazon.com/es_es/machine-learning/latest/dg/cross-validation.html

[24] “1.11. Ensemble methods — scikit-learn 1.1.2 documentation”. Consultado: el 20 de octubre de 2022. [En línea]. Disponible en: https://scikit-learn.org/stable/modules/ensemble.html?highlight=voting#voting-classifier

[25] “What is a REST API?” Consultado: el 29 de octubre de 2022. [En línea]. Disponible en: https://www.redhat.com/en/topics/api/what-is-a-rest-api

[26] “What is Python? Executive Summary | Python.org”. Consultado: el 27 de octubre de 2022. [En línea]. Disponible en: https://www.python.org/doc/essays/blurb/

[27] “Welcome to Flask — Flask Documentation (2.2.x)”. Consultado: el 19 de mayo de 2023. [En línea]. Disponible en: https://flask.palletsprojects.com/en/2.2.x/

[28] “Flask-RESTful — Flask-RESTful 0.3.8 documentation”. Consultado: el 19 de mayo de 2023. [En línea]. Disponible en: https://flask-restful.readthedocs.io/en/latest/

[29] “Joblib: running Python functions as pipeline jobs — joblib 1.3.0.dev0 documentation”. Consultado: el 19 de mayo de 2023. [En línea]. Disponible en: https://joblib.readthedocs.io/en/latest/

[30] “scikit-learn: machine learning in Python — scikit-learn 1.2.2 documentation”. Consultado: el 9 de marzo de 2023. [En línea]. Disponible en: https://scikit-learn.org/stable/

[31] “Angular - What is Angular?” Consultado: el 29 de octubre de 2022. [En línea]. Disponible en: https://angular.io/guide/what-is-angular

[32] “What Is MongoDB?”, MongoDB. Consultado: el 21 de julio de 2022. [En línea]. Disponible en: https://www.mongodb.com/es/what-is-mongodb

[33] “sklearn.model_selection.RandomizedSearchCV — scikit-learn 1.2.2 documentation”. Consultado: el 19 de mayo de 2023. [En línea]. Disponible en: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html

[34] “JSON Web Tokens - jwt.io”. Consultado: el 19 de mayo de 2023. [En línea]. Disponible en: https://jwt.io/

[35] “GitHub - jpadilla/pyjwt: JSON Web Token implementation in Python”. Consultado: el 19 de mayo de 2023. [En línea]. Disponible en: https://github.com/jpadilla/pyjwt/

[36] “pandas - Python Data Analysis Library”. Consultado: el 19 de mayo de 2023. [En línea]. Disponible en: https://pandas.pydata.org/

[37] “NumPy”. Consultado: el 19 de mayo de 2023. [En línea]. Disponible en: https://numpy.org/

[38] “Crear documentos PDF en Python con ReportLab - Recursos Python”. Consultado: el 11 de noviembre de 2023. [En línea]. Disponible en: https://recursospython.com/guias-y-manuales/crear-documentos-pdf-en-python-con-reportlab/

[39] “openpyxl - A Python library to read/write Excel 2010 xlsx/xlsm files — openpyxl 3.1.2 documentation”. Consultado: el 11 de noviembre de 2023. [En línea]. Disponible en: https://openpyxl.readthedocs.io/en/stable/

[40] M. S. Santos, P. H. Abreu, P. J. García-Laencina, A. Simão, y A. Carvalho, “A new cluster-based oversampling method for improving survival prediction of hepatocellular carcinoma patients”, *J Biomed Inform*, vol. 58, pp. 49–59, dic. 2015, doi: 10.1016/j.jbi.2015.09.012.

[41] A. Azhie *et al.*, “Development and validation of an ensemble machine learning framework for detection of all-cause advanced hepatic fibrosis: a retrospective cohort study”, 2022. [En línea]. Disponible en: www.thelancet.com/

[42] A. DOĞANER, C. ÇOLAK, F. KÜÇÜKDURMAZ, y C. ÖLMEZ, “Prediction of Renal Cell Carcinoma Based on Ensemble Learning Methods”, *Middle Black Sea Journal of Health Science*, mar. 2021, doi: 10.19127/mbsjohs.889492.

[43] N. D. Ponnaganti y R. Anitha, “A Novel Ensemble Bagging Classification Method for Breast Cancer Classification Using Machine Learning Techniques”, *Traitement du Signal*, vol. 39, núm. 1, pp. 229–237, feb. 2022, doi: 10.18280/ts.390123.

[44] F. Rahman y M. A. Mahmood, “A Dynamic Approach to Identify the Most Significant Biomarkers for Heart Disease Risk Prediction utilizing Machine Learning Techniques”.

[45] D. Chicco y G. Jurman, “An ensemble learning approach for enhanced classification of patients with hepatitis and cirrhosis”, *IEEE Access*, vol. 9, pp. 24485–24498, 2021, doi: 10.1109/ACCESS.2021.3057196.

[46] S. Gupta y M. K. Gupta, “Computational Prediction of Cervical Cancer Diagnosis Using Ensemble-Based Classification Algorithm”, *Computer Journal*, vol. 65, núm. 6, pp. 1527–1539, jun. 2022, doi: 10.1093/comjnl/bxaa198.

[47] F. Khanam y M. R. H. Mondal, “Ensemble Machine Learning Algorithms for the Diagnosis of Cervical Cancer”, en *2021 International Conference on Science and Contemporary Technologies, ICSCT 2021*, Institute of Electrical and Electronics Engineers Inc., 2021. doi: 10.1109/ICSCT53883.2021.9642612.

[48] C. D. Anisha y K. G. Saranya, “Early diagnosis of stroke disorder using homogenous logistic regression ensemble classifier”, *International Journal of Nonlinear Analysis and Applications*, vol. 12, núm. Special Issue, pp. 1649–1654, dic. 2021, doi: 10.22075/IJNAA.2021.5851.

[49] S. Buyrukoglu, “Improvement of Machine Learning Models Performances based on Ensemble Learning for the detection of Alzheimer Disease”, en *Proceedings - 6th International Conference on Computer Science and Engineering, UBMK 2021*, Institute of Electrical and Electronics Engineers Inc., 2021, pp. 102–106. doi: 10.1109/UBMK52708.2021.9558994.

[50] A. Singh, J. C. Mehta, D. Anand, P. Nath, B. Pandey, y A. Khamparia, “An intelligent hybrid approach for hepatitis disease diagnosis: Combining enhanced k-means clustering and improved ensemble learning”, en *Expert Systems*, Blackwell Publishing Ltd, ene. 2021. doi: 10.1111/exsy.12526.

[51] M. A. Hakim, N. Jahan, Z. A. Zerin, y A. B. Farha, “Performance Evaluation and Comparison of Ensemble Based Bagging and Boosting Machine Learning Methods for Automated Early Prediction of Myocardial Infarction”, en *2021 12th International Conference on Computing Communication and Networking Technologies, ICCCNT 2021*, Institute of Electrical and Electronics Engineers Inc., 2021. doi: 10.1109/ICCCNT51525.2021.9580063.

[52] E. Lin, C. H. Lin, y H. Y. Lane, “Applying a bagging ensemble machine learning approach to predict functional outcome of schizophrenia with clinical symptoms and cognitive functions”, *Sci Rep*, vol. 11, núm. 1, dic. 2021, doi: 10.1038/s41598-021-86382-0.

[53] M. Shyamala Devi *et al.*, “Feature Predominance Ensemble Inquisition towards Liver Disease Prediction using Machine Learning”, 2014, doi: 10.1016/j.aebj.2014.10.012.

[54] D. C. Yadav y S. Pal, “An Experimental Study of Diversity of Diabetes Disease Features by Bagging and Boosting Ensemble Method with Rule Based Machine Learning Classifier Algorithms”, *SN Comput Sci*, vol. 2, núm. 1, feb. 2021, doi: 10.1007/s42979-020-00446-y.

[55] “Cirrhosis Prediction Dataset | Kaggle”. Consultado: el 9 de marzo de 2023. [En línea]. Disponible en: https://www.kaggle.com/datasets/fedesoriano/cirrhosis-prediction-dataset

[56] “UCI Machine Learning Repository: ILPD (Indian Liver Patient Dataset) Data Set”. Consultado: el 9 de marzo de 2023. [En línea]. Disponible en: https://archive.ics.uci.edu/ml/datasets/ILPD+%28Indian+Liver+Patient+Dataset%29

[57] “UCI Machine Learning Repository: Liver Disorders Data Set”. Consultado: el 9 de marzo de 2023. [En línea]. Disponible en: https://archive.ics.uci.edu/ml/datasets/liver+disorders

[58] “Non-alcohol fatty liver disease | Kaggle”. Consultado: el 9 de marzo de 2023. [En línea]. Disponible en: https://www.kaggle.com/datasets/jamescorden/nonalcohol-fatty-liver-disease?select=nafld.csv

[59] “Servicios - Sanatorio Escudero”. Consultado: el 11 de noviembre de 2023. [En línea]. Disponible en: https://sanatorioescudero.com/servicios/