Diseño e Implementación de una Aplicación para el Diagnóstico y Detección Temprana de Diabetes Tipo 2

Sonya Valentina Castro Gomez
Ingeniería de Sistemas
Universidad del Norte
Barranquilla, Colombia
sonyac@uninorte.edu.co

Jeffrey Andres Felix Carvajal
Ingeniería de Sistemas
Universidad del Norte
Barranquilla, Colombia
felixj@uninorte.edu.co

Dario Jose Mejia Caballero
Ingeniería de Sistemas
Universidad del Norte
Barranquilla, Colombia
jdario@uninorte.edu.co

Wilson Nieto Bernal
Ingeniería de Sistemas
Universidad del Norte
Barranquilla, Colombia
wnieto@uninorte.edu.co

Eduardo David Angulo Madrid
Ingeniería de Sistemas
Universidad del Norte
Barranquilla, Colombia
edangulo@uninorte.edu.co

***Abstract*— La diabetes tipo 2 es una enfermedad crónica que representa un desafío significativo para la salud pública mundial, con una prevalencia en aumento y graves riesgos para la salud. En paralelo, el avance tecnológico, especialmente en el campo del Machine Learning (ML), ofrece oportunidades para mejorar el diagnóstico y tratamiento de enfermedades. Este estudio propone el desarrollo de un modelo de diagnóstico de diabetes tipo 2 basado en ML, con el objetivo de proporcionar una herramienta precisa y temprana para la detección de esta enfermedad. Se utilizarán técnicas avanzadas de ML para aprender de datos pasados y extraer características complejas, con el fin de crear un modelo que pueda ser implementado en una aplicación web interactiva. Esta aplicación permitirá a los usuarios ingresar síntomas relacionados con la diabetes tipo 2 y recibir una evaluación de riesgo personalizada. Los resultados esperados incluyen un modelo preciso y funcional, así como una aplicación web accesible que pueda mejorar la detección temprana y el tratamiento oportuno de la diabetes tipo 2, contribuyendo así a una atención médica más efectiva y personalizada.**

***Keywords*—Diabetes, Machine Learning, Diseases, Diagnostic**

I. Introducción

La diabetes es una enfermedad crónica que afecta a millones de personas en todo el mundo, con un aumento preocupante en su prevalencia, según datos de la Organización Mundial de la Salud (OMS) (OMS, 2024). Entre 1980 y 2014, el número de personas con diabetes aumentó de 108 millones a 422 millones, con un crecimiento más pronunciado en los países de ingresos bajos y medianos (OMS, 2024). Esta enfermedad conlleva riesgos graves para la salud, como ceguera, insuficiencia renal, infarto de miocardio, accidente cerebrovascular y amputación de miembros inferiores. Además, las tasas de mortalidad por diabetes han aumentado en un 3% entre 2000 y 2019, causando dos millones de defunciones en 2019, según las últimas estadísticas de la OMS (2024).

Por otro lado, en los últimos años, el avance acelerado de la tecnología ha generado un impacto significativo en diversos sectores, transformando la manera en que se llevan a cabo las operaciones y procesos administrativos en organizaciones públicas y privadas. Esta revolución tecnológica ha encontrado un campo de innovación en el ámbito del Machine Learning (ML), una rama de la Inteligencia Artificial (IA) que se centra en crear algoritmos y modelos para que las computadoras aprendan de los datos y realicen predicciones o juicios de forma autónoma (Gracious et al., 2023).

En el contexto de la salud, ML ha demostrado ser una herramienta poderosa que puede mejorar significativamente la precisión y eficiencia en el diagnóstico y tratamiento de enfermedades. Uno de los avances más significativos ha sido el desarrollo del Machine Learning (ML). Esta técnica permite a las computadoras aprender de los datos y realizar predicciones o juicios de manera autónoma (Gracious et al., 2023; Hamsagayathri & Vigneshwaran, 2021)

En particular, ML juega un papel fundamental en el desarrollo de sistemas de Apoyo a la Toma de Decisiones Clínicas (CDSS); estos sistemas ayudan a los a los profesionales de la salud a tomar decisiones informadas basadas en evidencia (Gracious et al., 2023). Desempeñando un papel crucial en la predicción de enfermedades, la estratificación de riesgos, la recomendación de tratamientos y el monitoreo en tiempo real, contribuyendo así a una atención más personalizada y eficaz para los pacientes (Gracious et al., 2023).

Estos modelos de ML han sido implementados para diversos propósitos, tal como se describe en (Azar et al., 2015), donde se desarrolla un modelo capaz de clasificar o predecir trastornos mentales basándose en una serie de síntomas ingresados a través de texto. En este caso, se emplea una variante del algoritmo genético. Del mismo modo, Mangal

and Jain (2022) desarrollaron un modelo capaz de predecir la enfermedad de la diabetes utilizando algoritmos de Random Forest para clasificar los síntomas médicos de los pacientes, logrando precisiones del 99%.

Este proyecto tiene como propósito desarrollar un modelo de diagnóstico de la diabetes tipo 2 utilizando principalmente técnicas avanzadas de ML. Al aprovechar la capacidad del ML para aprender de datos pasados para extraer características complejas de los datos, se busca crear una herramienta que pueda ayudar a los profesionales de la salud en la detección temprana y precisa de esta enfermedad. Adicionalmente, se busca que el modelo final sea implementado en un prototipo de aplicación web interactiva que permitirá a los usuarios ingresar una serie de síntomas relacionados con la diabetes tipo 2 y recibir una evaluación de riesgo de padecer dicha enfermedad basada en el análisis de datos clínicos y médicos. Con esto se busca que la herramienta contribuya a una atención médica más efectiva y personalizada, facilitando la detección temprana y el tratamiento oportuno de la diabetes tipo 2 .

II. Definición del problema

El campo de la ingeniería de sistemas enfrenta el reto de aplicar avances tecnológicos como el Machine Learning (ML) para resolver problemas complejos en diversos sectores. En el ámbito de la salud, estos avances prometen revolucionar el proceso diagnóstico y terapéutico, ofreciendo herramientas precisas para la toma de decisiones basadas en datos. El problema de investigación que se aborda es: ¿Cómo puede una página web, apoyada en técnicas de ML, mejorar la detección temprana y el tratamiento de la diabetes tipo 2, una enfermedad crónica que representa uno de los mayores desafíos para la salud pública mundial, y en qué medida puede asistir a los profesionales de la salud en la toma de decisiones informadas y basadas en la evidencia? Este problema refleja la necesidad de una solución computacional tangible que se materialice en un prototipo de página web interactivo donde los usuarios puedan ingresar síntomas y recibir una evaluación de riesgo basada en datos clínicos y médicos.

Para ilustrar las diferentes dimensiones de este desafío y cómo se interrelacionan en el contexto de nuestro proyecto, se ha desarrollado el siguiente Árbol del Problema (ver Figura 1). Este diagrama visual detalla la jerarquía de los factores implicados en la detección y manejo de la diabetes tipo 2, desde la recopilación y análisis de datos hasta el desarrollo de modelos predictivos y su implementación en una solución de software accesible.

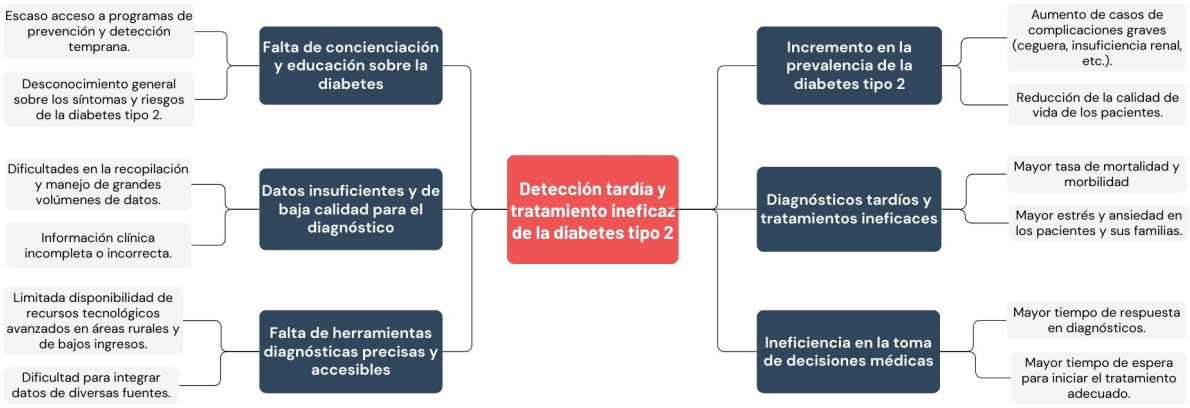

Fig. 1. Árbol del Problema

El desarrollo de Sistemas de Apoyo a la Decisión Clínica (CDSS) se presenta como una solución prometedora en este contexto. Estudios como el de Gracious et al. (2023) destacan la importancia de estas herramientas que asisten en la toma de decisiones médicas minimizando los errores de diagnóstico y tratamiento, sin embargo, reconocen que estos sistemas no pueden reemplazar a los profesionales de la salud sino complementar su juicio clínico.

En este sentido, ML se consolidan como una poderosa herramienta que mejora significativamente la precisión y eficiencia en el diagnóstico y tratamiento de enfermedades. Su capacidad para aprender de los datos y realizar predicciones o juicios de manera autónoma ha sido un avance significativo en el desarrollo de los CDSS, como lo señalan Hamsagayathri and

Vigneshwaran (2021), quienes describen cómo estos sistemas pueden desempeñar un papel crucial en la predicción de enfermedades, la estratificación de riesgos, la recomendación de tratamientos y el monitoreo en tiempo real, contribuyendo a una atención más personalizada y efectiva para los pacientes

Además, el uso de técnicas de aprendizaje supervisado como la regresión lineal, Support Vector Machine (SVM), Random Forest (RF) y Decision Tree (DT), aplicadas en la salud para la categorización de enfermedades y predicción de riesgos, demuestra la aplicabilidad y relevancia de ML en el diagnóstico médico, lo cual respalda la factibilidad de la propuesta del proyecto.

En resumen, este proyecto plantea un reto multidisciplinario: integrar los avances de ML en una plataforma web que pueda transformar la calidad de la atención médica en el tratamiento de la diabetes tipo 2. Este desafío no solo requiere una comprensión profunda de las tecnologías involucradas sino también una evaluación cuidadosa de su implementación y efectividad en entornos reales, lo que implica la posibilidad de realizar pruebas empíricas para observar su impacto en la realidad única y objetiva.

III. Justificación

Las técnicas de Machine Learning proveen herramientas para asistir a los practicantes de medicina en la prevención, diagnóstico y tratamiento (Arun Bhavsar et al., 2021; Caballé-Cervigón, Castillo-Sequera, Gómez-Pulido, Gómez-Pulido, & Polo-Luque, 2020). Un diagnóstico erróneo puede llevar a negar el cuidado necesario del paciente o a la aplicación errónea (Arun Bhavsar et al., 2021). Esto puede causar complicaciones en la salud del paciente además de aumentar los gatos médicos (Arun Bhavsar et al., 2021). Bajo este contexto, las herramientas de diagnóstico ML buscan reducir los errores humanos, mejorar los servicios médicos y reducir el costo y tiempo de estos (Arun Bhavsar et al., 2021; Alowais et al., 2023). Sin embargo, estas herramientas aún están en las etapas iniciales de integración a la práctica clínica (Alowais et al., 2023). Además, a pesar de la inmensa disponibilidad de datos disponibles, los modelos tienden a ser lineales con un rango reducido de variables (Caballé-Cervigón et al., 2020)

El presente proyecto propone una solución de diagnóstico basado en síntomas para asistencia de personal médico con un enfoque a la enfermedad de la diabetes tipo 2. Este proyecto busca crear un modelo para usarse mediante una interfaz sencilla y amigable con el usuario que permita ingresar una serie de síntomas relacionados con la diabetes tipo 2 para la obtención de un posible diagnóstico.

IV. Objetivos

A. Objetivo General

Diseñar e implementar una aplicación web basada en Machine Learning para diagnosticar y detectar tempranamente la Diabetes tipo 2, utilizando datos del Behavioral Risk Factor Surveillance System (BRFSS) de 2015. Esta herramienta estará diseñada para facilitar la identificación de individuos en riesgo de desarrollar esta condición, basándose en análisis predictivos avanzados.

B. Objetivos Específicos

1) Realizar una revisión sistemática de los avances en el uso de Machine Learning y Deep Learning para la detección y diagnóstico temprano de la Diabetes tipo 2, evaluando la efectividad de estas tecnologías en el análisis de grandes conjuntos de datos de salud.
2) Desarrollar la arquitectura de la solución para procesar el conjunto de los datos BRFSS 2015 asociados, identificando los factores de riesgo y patrones asociados con la diabetes tipo 2.
3) Implementar una aplicación web que integre un modelo de ML predictivo para ofrecer una solución práctica que ayude en la detección temprana de la Diabetes tipo 2. La aplicación proporcionará una interfaz intuitiva para que los usuarios (investigadores, posiblemente profesionales de la salud) puedan fácilmente ingresar datos y recibir evaluaciones de riesgo.
4) Validar la precisión del modelo y la funcionalidad de la aplicación a través de pruebas coherentes con el desarrollo del proyecto, para asegurar que la herramienta es efectiva en identificar correctamente a individuos en riesgo.

V. Metodología

Para el desarrollo del presente trabajo, se utilizará la metodología de desarrollo CRISP-ML (Q) (Cross-Industry Standard Process model for the development of Machine Learning applications with Quality assurance) (Studer et al., 2021). Inspirado en CRISP-DM (minería de datos), CRISP-ML(Q) es un modelo iterativo (por lo cual un paso anterior es fundamental proceso) que añade una capa de monitoreo y de calidad en cada etapa y tareas (Studer et al., 2021).

A. Comprensión de los datos

Studer et al. (2021) define las siguientes tareas de esta fase: Definir el alcance de la aplicación ML, definir los criterios de éxito, definir la factibilidad, colectar los datos, verificar la calidad de los datos, revisar resultados de la fase.

Para este proyecto, los datos fueron obtenidos de la plataforma **Kaggle**. Se selección el conjunto de datos "Diabetes Health Indicators Dataset" que contiene $236,378$ respuestas de la encuesta BRFSS 2021 (Nazreen, 2023). Esta encuesta es un estudio telefónico continuo, realizado anualmente por los Centros para el Control y la Prevención de Enfermedades (CDC), que recopila datos sobre comportamientos de riesgo para la salud, condiciones crónicas y uso de servicios preventivos entre adultos en Estados Unidos (Nazreen, 2023). Este conjunto de datos consta de 3 archivos principales.

1) *diabetes_012_health_indicators_BRFSS2021.csv* que contiene $236,378$ respuestas y la variable objetivo que indica sin diabetes (o gestacional), prediabetes y diabetes (Nazreen, 2023).

2) *diabetes_binary_health_indicators_BRFSS2021.csv* que contiene $236,378$ respuestas y la variable objetivo es binaria: sin diabetes o diabetes (Nazreen, 2023).
3) *diabetes_binary_5050split_health_indicators_BRFSS2021.csv* que realiza una división equilibrada entre individuos sin diabetes y aquellos con diabetes (prediabetes y diabetes) (Nazreen, 2023). Contiene $67,136$ respuestas y, como en el caso anterio, la variable objetivo es binaria: sin diabetes o diabetes (Nazreen, 2023).

Se seleccionó el primer archivo (*diabetes_012_health_indicators_BRFSS2021.csv*) como conjunto de datos. Este contiene 1 variable objetivo (*Diabetes_binary*) y 21 variables de entrada. A continuación, se muestra el nombre de las variables junto a su tipo y pregunta y/o descripción asociada con esta variable.

TABLE I: Descripción del conjunto de datos

Variable	Rol	Tipo	Descripción
ID	ID	Integer	ID del paciente
Diabetes_binary	Target	Binario	0 = No diabetes 1 = Pre-diabetes o diabetes
HighBP	Feature	Binario	¿Tiene presión arterial alta?
HighChol	Feature	Binario	¿Tiene colesterol alto?
CholCheck	Feature	Binario	¿Se ha realizado un chequeo de colesterol en 5 años?
BMI	Feature	Integer	Índice de masa corporal
Smoker	Feature	Binario	¿Ha fumado al menos 100 cigarros en su vida entera?
Stroke	Feature	Binario	¿(Alguna vez le dijeron) que tuvo un derrame cerebral?
HeartDiseaseorAttack	Feature	Binario	¿Tiene enfermedad coronaria o infarto de miocardio?
PhysActivity	Feature	Binario	¿Ha realizado actividad física en los últimos 30 días?(Sin incluir el trabajo)
Fruits	Feature	Binario	¿Consume fruta 1 o más veces al día?
Veggies	Feature	Binario	¿Consume verduras 1 o más veces al día?
HvyAlcoholConsump	Feature	Binario	¿Bebedores empedernidos? (hombres adultos que toman más de 14 tragos por semana y mujeres adultas que toman más de 7 tragos por semana)
AnyHealthcare	Feature	Binario	¿Tener cualquier tipo de cobertura de atención médica, incluidos seguros médicos, planes prepagos como HMO, etc.?
NoDocbcCost	Feature	Binario	¿Hubo algún momento en los últimos 12 meses en el que necesitó consultar a un médico pero no pudo debido al costo?
GenHlth	Feature	Integer	Diría usted que en general su salud es: escala 1-5 1 = excelente 2 = muy buena 3 = buena 4 = regular 5 = mala
MentHlth	Feature	Integer	¿Durante cuántos días durante los últimos 30 días su salud mental no fue buena? (Incluye estrés, depresión y problemas emocionales)
PhysHlth	Feature	Integer	¿Durante cuántos días durante los últimos 30 días su salud física no fue buena? (incluye enfermedades y lesiones físicas)
DiffWalk	Feature	Binario	¿Tiene serias dificultades para caminar o subir escaleras?
Sex	Feature	Binario	Sexo biológico. F = 0, M = 1
Age	Feature	Integer	1 = 18-24; 9 = 60-64; 13 = 80 o más

Education	Feature	Integer	Escala 1-6; 1 = Nunca asistió a la escuela o solo jardín de infantes; 2 = Grados 1 a 8 (primaria); 3 = Grados 9 a 11 (algo de escuela secundaria); 4 = Grado 12 o GED (escuela secundaria graduado); 5 = Universidad 1 año a 3 años (Alguna universidad o escuela técnica); 6 = Universidad 4 años o más (Graduado universitario).
Income	Feature	Integer	Escala de ingresos; 1 = menos de $ 10,000; 5 = menos de $35,000; 8 = $75,000 o más.

1) verificación de la calidad

Para asegurar la calidad de los datos, se realiza un análisis exploratorio de datos (EDA) en el cual se indicaron valores atípicos, se evaluó la distribución de variables y posibles correlaciones. A continuación, se comparten resultados encontrados.

a) Datos faltantes y duplicados

No se encontraron datos faltantes. Se encontraron 24,206 datos duplicados que fueron descartados quedando 229,474 datos.

b) Correlaciones

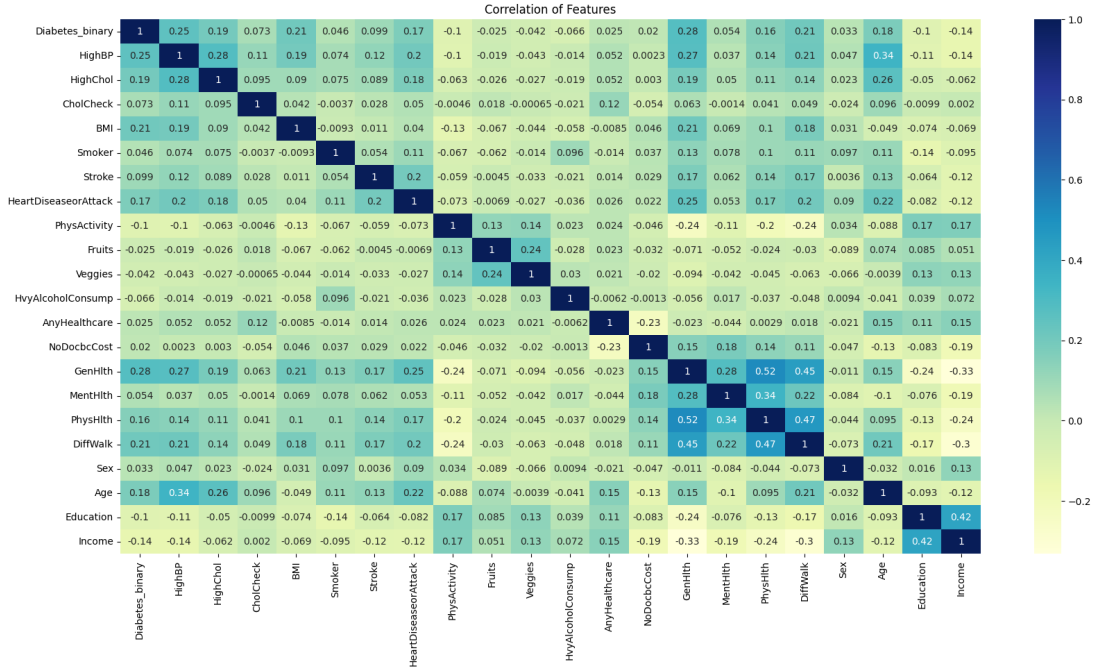

Fig. 2. Mapa de Correlaciones

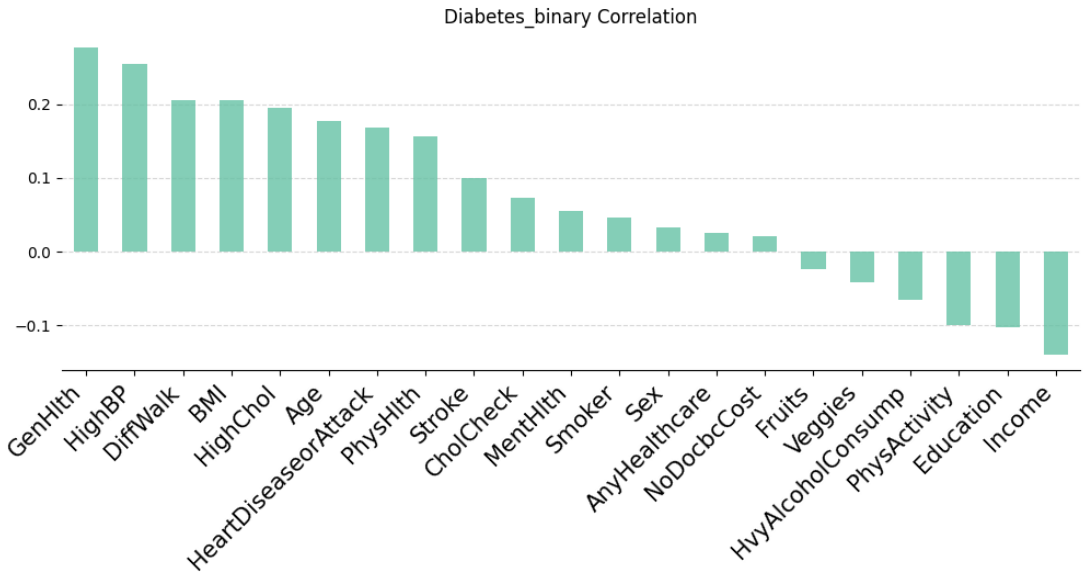

Fig. 3. Diabetes_binary - Correlaciones

Analizando el mapa de correlación que se observa en la figura 2, se observa que los siguientes pares de variables están correlacionadas positivamente una a la otra (*GenHlth*, *PhysHlth*), (*PhysHlth*, *DiffWalk*) y (*GenHlth*, *DiffWalk*). De forma similar, se observa que los pares de variables (*GenHlth*, *Income*) y (*DiffWalk*, *Income*) están correlacionadas negativamente. Observando la figura 3, se puede notar el nivel de correlación de las variables respecto a la variable objetivo *Diabetes_binary*. Se puede ver que las variables más correlacionadas son *GenHlth*, *HighBP*, *DiffWalk*, *BMI*, *HighChol*, *Age*, *HeartDiseaseorAttack*, *PhysHlth*, *Physactivity*, *Education*, *Income*. También se pueden observar las variables con menos correlación: *AnyHealthcare*, *NoDocbcCost*, *Fruits*, *Sex*, *Smoker*, *Veggies*.

c) Datos atípicos

Para el análisis de datos atípicos solo se tomaron en cuenta los datos de tipo reale (*BMI*, *GenHlth*, *MentHlth*, *PhysHlth*, *Age*, *Education*, *Income*). De estos, encontramos valores atípicos en *BMI*, *GenHlth*, *MentHlth* y *PhysHlth*. A continuación, se muestran gráficamente estos valores.

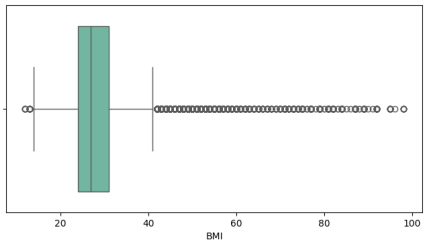

Fig. 4. BMI - Datos atípicos

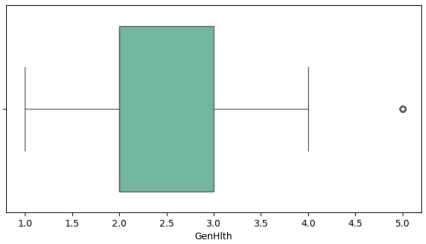

Fig. 5. GenHlth - Datos atípicos

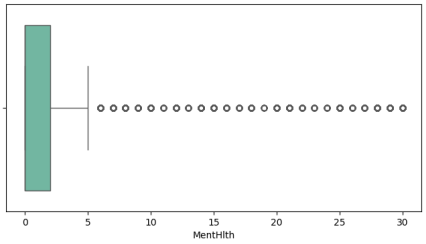

Fig. 6. MentHlth - Datos atípicos

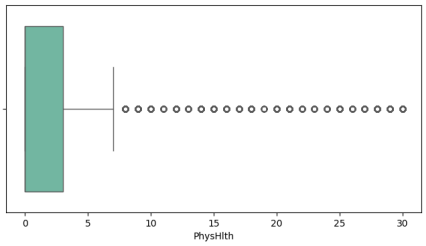

Fig. 7. PhysHlth - Datos atípicos

d) Desequilibrio de clase

Se encontró desequilibro de clase en la variable objetivo *Diabetes_binary* con 194,377 de los datos (84.71%) perteneciendo a la clase mayoritaria de No diabéticos y el 35,097 de los datos (15.29%) perteneciendo a la clase minoritaria de diabéticos.

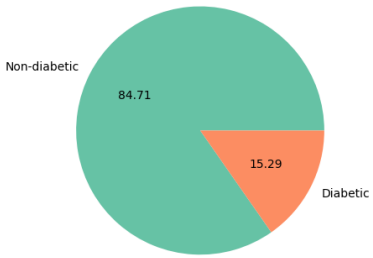

Fig. 8. Diabetes_binary - Class imabalance

B. Preparación de los datos

Esta fase provee de un conjunto de datos para la siguiente fase (entrenamiento) (Studer et al., 2021). Las tareas de esta fase incluyen la limpieza de datos, selección de variables (*feature selection*), selección de datos, resolver el desequilibrio de clase, construcción de datos (*feature engineering*), estandarización de datos (Studer et al., 2021).

1) Selección de variables

a) Correlación de Pearson

Como observamos en la tabla II, la variable *GenHlth* presenta el coeficiente de correlación de Pearson más alto (0.276940), indicando una asociación moderada con la diabetes tipo 2. Por otro lado, estableciendo un limite de 0.05 las variables *Smoker*, *Veggies*, *Sex*, *AnyHealthcare*, *Fruits*, *NoDocbcCost* tienen coeficientes de correlación bajos y, por tanto, pueden descartarse debido a su débil asociación.

TABLE II: Correlación de Pearson

Variable	Coeficiente
GenHlth	0.276940
HighBP	0.254318
DiffWalk	0.205302
BMI	0.205086
HighChol	0.194944
BMI_bins	0.194376
Age	0.177263
HeartDiseaseorAttack	0.168213
PhysHlth	0.156211
Income	0.140659
Education	0.102686
PhysActivity	0.100404
Stroke	0.099193
CholCheck	0.072523
HvyAlcoholConsump	0.065950
MentHlth	0.054153
Smoker	**0.045504**
Veggies	**0.041734**
Sex	**0.032724**
AnyHealthcare	**0.025331**
Fruits	**0.024805**
NoDocbcCost	**0.020048**

b) Chi^2 test

Como se puede observar en la tabla III, $p - values$ son bastante cercanos a ceros, lo que sugiere que la hipótesis nula puede ser rechazada para todas las variables. Esto indica que hay asociaciones fuertes entre estas características y la variables objetivos.

TABLE III: Chi^2

Variable	ChiSqr_Score	ChiSqr_pValue
PhysHlth	97988.761672	0.000000e+00
BMI	15507.736174	0.000000e+00
MentHlth	11419.584750	0.000000e+00
Age	8539.906340	0.000000e+00
HighBP	8098.548237	0.000000e+00
DiffWalk	7875.496177	0.000000e+00
GenHlth	7671.732832	0.000000e+00
HeartDiseaseorAttack	5822.145697	0.000000e+00
HighChol	4869.312739	0.000000e+00
Income	3377.099257	0.000000e+00
Stroke	2156.678382	0.000000e+00
BMI<_bins	1887.151798	0.000000e+00
HvyAlcoholConsump	937.401148	7.268457e-206
PhysActivity	617.563886	2.532822e-136
Education	479.112939	3.332472e-106
Smoker	253.826098	3.804877e-57
Sex	137.837135	7.910571e-32
NoDocbcCost	83.662830	5.867820e-20
Veggies	82.098846	1.294482e-19
Fruits	54.688897	1.412018e-13

CholCheck	48.904140	2.687828e-12
AnyHealthcare	7.949731	4.809451e-03
NoDocbcCost	0.000498	

c) Información mutua

Este método está diseñado para manejar variables de respuesta categóricas y puede proporcionar información sobre las relaciones entre las características binarias y la variable predictora. La información mutua mide en qué medida el conocimiento sobre una variable reduce la incertidumbre sobre otra; puntuaciones más altas significan una mayor reducción de la incertidumbre.

Como se observa en la tabla IV, `CholCheck` tiene la puntuación de información mutua más alta, lo que sugiere que el conocimiento de la frecuencia de control del colesterol de un paciente podría reducir más la incertidumbre sobre su estado diabético en comparación con otras características. Sin embargo, esto no significa necesariamente que los controles de colesterol causen o prevengan la diabetes. Podría ser que los proveedores de atención médica recomienden a las personas con diabetes que controlen su colesterol, lo que refleja un patrón de comportamiento en lugar de un vínculo causal.

De manera similar, `AnyHealthCare` también tiene una puntuación alta de información mutua, lo que indica una fuerte asociación con la variable objetivo. Una vez más, esto podría reflejar un patrón en el que las personas diabéticas están más comprometidas con los servicios de atención médica en general.

Si bien la información mutua ayuda a identificar relaciones, no distingue entre correlación y causalidad. Por lo tanto, funciones como `CholCheck` y `AnyHealthCare` pueden ser indicadores del estado diabético debido a comportamientos de salud relacionados o compromiso con la atención médica, no porque tengan un efecto directo sobre la diabetes.

TABLE IV: Información mutua

Variable	Mutual_info
HighBP	0.046057
CholCheck	0.045963
AnyHealthcare	0.040140
PhysActivity	0.036017
Veggies	0.033600
HighChol	0.033259
Fruits	0.026675
DiffWalk	0.021429
Smoker	0.019012
Sex	0.016186
HeartDiseaseorAttack	0.012232
Stroke	0.004226
HvyAlcoholConsump	0.003698

d) ANNOVA Test

ANOVA se utiliza normalmente para datos de respuesta continua donde las variables predicativas son categóricas y supone que la variable de respuesta se distribuye normalmente dentro de cada grupo. Para las características binarias del conjunto de datos, ANOVA no es apropiado porque el resultado binario viola el supuesto de normalidad.

La tabla V muestra los resultados de las pruebas ANOVA (Análisis de Varianza), que evalúan si existen diferencias estadísticamente significativas entre las medias de estas características para diferentes grupos, comparando poblaciones diabéticas y no diabéticas.

El estadístico F es una relación entre la varianza entre las medias del grupo y la varianza dentro de los grupos. Un estadístico F más alto indica una mayor probabilidad de que existan diferencias significativas entre las medias de los grupos. Los valores p en la tabla son extremadamente pequeños, lo que sugiere que la hipótesis nula (que las medias del grupo son iguales) puede rechazarse rotundamente para cada característica.

Comenzando con `GenHlth`, que tiene la estadística F más alta, podemos inferir que hay una fuerte evidencia de diferencia en el estado de salud general entre los grupos de diabéticos y no diabéticos. Las diferencias significativas continúan con el `IMC`, la edad, la salud física, los ingresos y la educación, todos los cuales indican disparidades en estos aspectos al comparar los dos grupos.

`MentHlth` tiene el estadístico F más bajo, pero aún indica una diferencia significativa, aunque menos fuerte que los demás. Esto sugiere que, si bien el estado de salud mental difiere entre los grupos de diabéticos y no diabéticos, la diferencia de medias no es tan pronunciada como lo es para la salud general o el `IMC`.

En resumen, ANOVA ha identificado que las medias de estas características no son las mismas en los grupos de diabéticos y no diabéticos.

TABLE V: ANOVA test

Variable	IV
GenHlth	0.648408
HighBP	0.532771
BMI	0.391081
Age	0.368591
BMI_bins	0.327313
HighChol	0.298296
DiffWalk	0.267031
HeartDiseaseorAttack	0.167530
PhysHlth	0.161874

Income	0.151785
Education	0.079076
CholCheck	0.077694
PhysActivity	0.071942
Stroke	0.057511
HvyAlcoholConsump	0.047814
MentHlth	0.024899
Smoker	0.015939
Veggies	0.012756
Sex	0.008216
AnyHealthcare	0.005533
Fruits	0.004702
NoDocbcCost	0.002950

e) Information Value (IV) y Weight of evidence (WOE)

TABLE VI: Valores de referencia IV

Variable	IV
$< .02$	Irelevante para predicciones
$.02 - .10$	Predictor debil
$.10 - 0.30$	Predictor mediano
$.30 - .50$	Predictor fuerte
$> .50$	Comportamiento sospechoso

Los resultados de la tabla VII muestran que la variable más significativa es `GenHlth`, lo que sugiere que esta percepción puede proporcionar información valiosa sobre otros aspectos de la salud y el comportamiento. Además, `HighBP` y `BMI` también son factores importantes, lo que indica la relevancia de la hipertensión y el peso corporal en el estudio. `Age` también juega un papel significativo en los resultados, lo cual es coherente con la comprensión común de que muchos problemas de salud aumentan con la edad.

Finalmente, factores como el colesterol alto (`HighChol`) también muestran una influencia importante en el estudio, lo que subraya la importancia de abordar los niveles de colesterol para la salud general. Estos hallazgos sugieren áreas importantes para intervenciones de salud pública o estrategias de prevención de enfermedades.

TABLE VII: Resultados IV y WOE test

Variable	IV
GenHlth	0.648408
HighBP	0.532771
BMI	0.391081
Age	0.368591
BMI_bins	0.327313
HighChol	0.298296
DiffWalk	0.267031
HeartDiseaseorAttack	0.167530
PhysHlth	0.161874
Income	0.151785
Education	0.079076
CholCheck	0.077694
PhysActivity	0.071942
Stroke	0.057511
HvyAlcoholConsump	0.047814
MentHlth	0.024899
Smoker	0.015939
Veggies	0.012756
Sex	0.008216
AnyHealthcare	0.005533
Fruits	0.004702
NoDocbcCost	0.002950

2) Manejo del desequilibrio de clase

a) Sin balancear

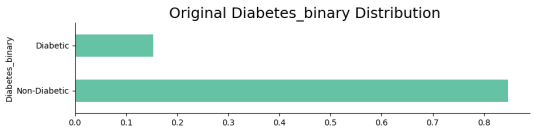

Fig. 9. Original Diabetes_binary Distribution

Como se puede observar en el gráfico 9, existe un desbalance significativo entre las clases, donde la gran mayoría de los casos pertenecen a la clase ”No Diabético” y una minoría a la clase ”Diabético”. Este desbalance también afecta al conjunto de datos seleccionado para el entrenamiento. Por lo tanto, no es viable entrenar el modelo sin antes realizar un balanceo adecuado de las clases. Los resultados preliminares del rendimiento con estos datos pueden ser observados en la tabla VIII y IX.

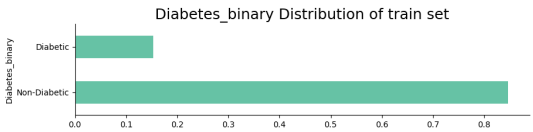

Fig. 10. Distribución de Diabetes_binary del conjunto de datos

TABLE VIII: Imbalanced dataset - Train Performance

Model	Accuracy	Recall	Precision
RandomForest	0.985794	0.920825	0.985722
KNeighbors	0.875080	0.360928	0.674214
CatBoost	0.866284	0.217213	0.711743
DecisionTree	0.864933	0.243402	0.664380
XgBoost	0.863619	0.206349	0.685984
MLP	0.855159	0.215835	0.575754
GradientBoosting	0.853739	0.168079	0.582385
GaussianNB	0.774446	0.501115	0.340655

TABLE IX: Imbalanced dataset - Test Performance

Model	Accuracy	Recall	Precision
GradientBoosting	0.853626	0.168012	0.552607
XgBoost	0.853059	0.167723	0.545966
CatBoost	0.852972	0.170605	0.544118
MLP	0.850837	0.195677	0.517925
DecisionTree	0.844169	0.176369	0.460150
RandomForest	0.839202	0.184438	0.426667
KNeighbors	0.831881	0.220749	0.398958
GaussianNB	0.777497	0.503458	0.340546

El entrenamiento inicial sin balancear mostró altos niveles de precisión y exactitud para algunos modelos como RandomForest, pero con un desempeño pobre en recall para la mayoría de los otros modelos, tanto en el conjunto de entrenamiento como en el de pruebas. Esto indica un sobreajuste significativo hacia la clase mayoritaria, lo que sugiere que el modelo no generaliza bien en la identificación de la clase minoritaria, es decir, los casos de diabetes.

b) Undersampling

Para abordar este desbalance, utilizaremos técnicas de undersampling como NearMiss y RandomUnderSampler. NearMiss selecciona ejemplos de la clase mayoritaria que están cerca de los ejemplos de la clase minoritaria, asegurando que el modelo aprenda a distinguir mejor entre las clases. En la tabla XI, se muestran los resultados de las pruebas utilizando NearMiss. Como se puede apreciar, esta técnica ayuda a equilibrar la distribución de las clases y mejora la capacidad del modelo para identificar correctamente los casos de ”Diabético”.

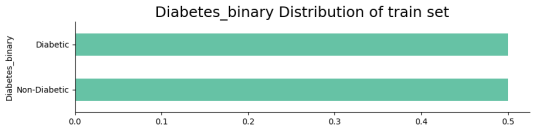

Fig. 11. Distribución de Diabetes_binary del conjunto de datos con NearMiss

TABLE X: NearMiss dataset - Train Performance

Model	Accuracy	Recall	Precision
RandomForest	0.956467	0.926492	0.985278
CatBoost	0.887738	0.810716	0.957507
XgBoost	0.885316	0.805815	0.957306
MLP	0.879170	0.810716	0.938451
GradientBoosting	0.865941	0.792216	0.928271
DecisionTree	0.865412	0.759056	0.963057
KNeighbors	0.858940	0.770858	0.934641
GaussianNB	0.821512	0.677216	0.950370

TABLE XI: NearMiss dataset - Test Performance

Model	Accuracy	Recall	Precision
MLP	0.869604	0.798775	0.932754
CatBoost	0.868797	0.791235	0.938723
XgBoost	0.866945	0.785768	0.940334
GradientBoosting	0.860915	0.785485	0.927339
DecisionTree	0.849091	0.737606	0.952068
RandomForest	0.833468	0.791989	0.866021
GaussianNB	0.817608	0.673516	0.949887
KNeighbors	0.806876	0.704053	0.889603

Después de aplicar NearMiss, se observa una mejora notable en recall, lo que indica que los modelos ahora pueden identificar mejor los casos de la clase minoritaria. Aunque hay una ligera disminución en precisión, en el conjunto de pruebas se mantiene un equilibrio entre recall y precisión, lo que sugiere que los modelos son más efectivos en detectar casos de diabetes sin sacrificar demasiado la precisión general.

Además de NearMiss, también hemos aplicado la técnica RandomUnderSampler para manejar el desbalance de clases. RandomUnderSampler reduce aleatoriamente el número de ejemplos de la clase mayoritaria, simplificando el problema y ayudando a equilibrar la distribución de las clases. En las tablas XII y XIII se pueden observar los resultados obtenidos con esta técnica.

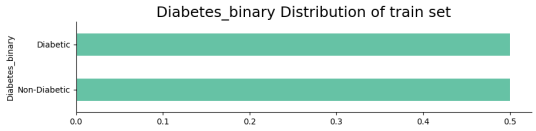

Fig. 12. Distribución de Diabetes_binary del conjunto de datos con RandomUnderSampler

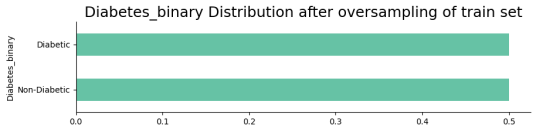

Fig. 13. Distribución de Diabetes_binary del conjunto de datos con SMOTE

TABLE XII: RandomUnderSampler dataset - Train Performance

Model	Accuracy	Recall	Precision
RandomForest	0.990801	0.989825	0.991680
DecisionTree	0.793060	0.823041	0.775128
KNeighbors	0.787006	0.807661	0.774203
XgBoost	0.785463	0.816784	0.767249
CatBoost	0.773353	0.808713	0.753870
MLP	0.759934	0.780058	0.748331
GradientBoosting	0.741769	0.776901	0.724333
GaussianNB	0.697756	0.641637	0.720562

TABLE XIII: RandomUnderSampler dataset - Test Performance

Model	Accuracy	Recall	Precision
GradientBoosting	0.731780	0.766738	0.720349
CatBoost	0.729335	0.764722	0.717910
XgBoost	0.722271	0.756521	0.711828
MLP	0.720777	0.744421	0.714728
RandomForest	0.709774	0.733934	0.704115
DecisionTree	0.707736	0.738102	0.699809
KNeighbors	0.687767	0.709465	0.684170
GaussianNB	0.681247	0.618580	0.712560

La técnica de RandomUnderSampler mejoró la capacidad del modelo para equilibrar las clases, resultando en altos niveles de precisión y recall para el modelo RandomForest, pero con un rendimiento más modesto en otros modelos. En las pruebas, aunque mostró mejoras en recall, lo hizo a costa de una precisión global más baja. Esto sugiere que la técnica es efectiva pero puede llevar a una pérdida de información valiosa, impactando negativamente la precisión general.

c) Oversampling

Además de probar con técnicas de undersampling, también realizamos pruebas con técnicas de oversampling como SMOTE las cuales arrojaron resultados similares en cuanto a balanceo de clases para el set de entrenamiento de los modelos.

TABLE XIV: SMOTE dataset - Train Performance

Model	Accuracy	Recall	Precision
RandomForest	0.993127	0.992362	0.993847
DecisionTree	0.804339	0.820839	0.793818
XgBoost	0.802766	0.829639	0.786541
KNeighbors	0.789683	0.816355	0.774191
CatBoost	0.780243	0.814446	0.761469
MLP	0.760992	0.806725	0.738262
GradientBoosting	0.745053	0.778414	0.728799
GaussianNB	0.698683	0.642009	0.722778

TABLE XV: SMOTE dataset - Test Performance

Model	Accuracy	Recall	Precision
GradientBoosting	0.738794	0.777458	0.723811
CatBoost	0.732709	0.771697	0.718063
XgBoost	0.728168	0.761521	0.716143
MLP	0.721986	0.768625	0.705250
RandomForest	0.714645	0.741167	0.706184
DecisionTree	0.696580	0.715054	0.692065
GaussianNB	0.694842	0.634793	0.724523
KNeighbors	0.690301	0.711982	0.684891

La aplicación de SMOTE resultó en una mejora significativa tanto en recall como en precisión durante el entrenamiento y las pruebas. Los modelos equilibrados con SMOTE lograron mantener una precisión considerable. Sin embargo, en el conjunto de pruebas, no se destacaron por encima de los métodos anteriores.

d) SMOTE y NearMiss

Además de probar las técnicas de Oversampling y Undersampling por separado, también se realizaron pruebas para mezclar estas tecnicas y así intentar obtener mejores resultados en las predicciones del modelo. Con SMOTE y NearMiss, el objetivo es crear nuevas instancias sintéticas de la clase minoritaria y luego seleccionar ejemplos de la clase mayoritaria que estén más cercanos a los ejemplos de la clase minoritaria. De esta aplicación pudimos obtener los siguiente resultados en cuanto a los set de entrenamiento.

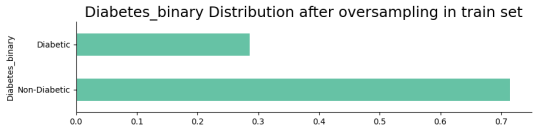

Fig. 14. Distribución de Diabetes_binary después de SMOTE en clase minoritaria del conjunto de datos

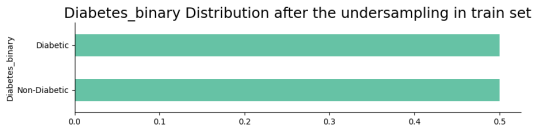

Fig. 15. Distribución de Diabetes_binary después de NearMiss en clase mayoritaria del conjunto de datos

TABLE XVI: SMOTE and NearMiss dataset - Train Performance

Model	Accuracy	Recall	Precision
RandomForest	0.960928	0.958781	0.962871
KNeighbors	0.843859	0.831651	0.852303
CatBoost	0.817106	0.751144	0.865083
XgBoost	0.813330	0.744526	0.863107
DecisionTree	0.806725	0.730498	0.861666
MLP	0.805034	0.723880	0.863901
GradientBoosting	0.796913	0.734819	0.838786
GaussianNB	0.740312	0.588954	0.844271

TABLE XVII: SMOTE and NearMiss dataset - Test Performance

Model	Accuracy	Recall	Precision
RandomForest	0.825681	0.831014	0.822632
CatBoost	0.804673	0.738237	0.851885
XgBoost	0.802015	0.734127	0.849998
MLP	0.797706	0.716830	0.855719
GradientBoosting	0.795198	0.731815	0.838591
DecisionTree	0.788317	0.708439	0.843718
KNeighbors	0.767546	0.746157	0.780021
GaussianNB	0.738264	0.586077	0.843386

Combinar SMOTE con NearMiss resultó en una alta precisión y recall durante el entrenamiento, aprovechando las ventajas de ambos métodos. En las pruebas, aunque el rendimiento fue ligeramente menor, se mantuvo consistente. Esto indica que esta estrategia ayuda a los modelos a detectar casos de diabetes de manera efectiva sin perder demasiada precisión general.

e) SMOTE y Random Under Sampler

Con el mismo propósito que antes, se empleó Random Under Sampler como técnica de undersampling y SMOTE nuevamente como técnica de oversampling para obtener una distribución de datos diferente y así intentar mejorar los resultados. Los resultados obtenidos en relación con los conjuntos de entrenamiento fueron los siguientes.

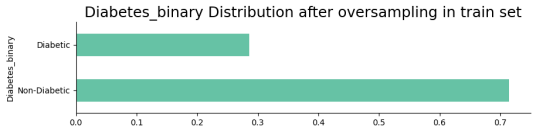

Fig. 16. Distribución de Diabetes_binary después de SMOTE en clase minoritaria del conjunto de datos

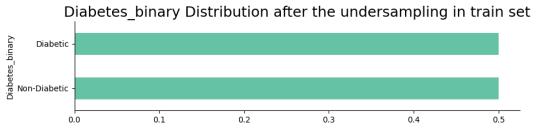

Fig. 17. Distribución de Diabetes_binary después de RUS en clase mayoritaria del conjunto de datos

TABLE XVIII: SMOTE and RandomUnderSampler dataset - Train Performance

Model	Accuracy	Recall	Precision
RandomForest	0.981194	0.986340	0.976271
KNeighbors	0.805374	0.856820	0.776724
XgBoost	0.761158	0.806979	0.739043
CatBoost	0.756867	0.801739	0.735520
DecisionTree	0.756840	0.810362	0.731816
MLP	0.740478	0.772066	0.725979
GradientBoosting	0.731759	0.776736	0.712426
GaussianNB	0.689701	0.690328	0.689213

TABLE XIX: SMOTE and RandomUnderSampler dataset - Test Performance

Model	Accuracy	Recall	Precision
RandomForest	0.722416	0.647550	0.303895
CatBoost	0.703721	0.745245	0.304200
XgBoost	0.699625	0.744669	0.300780
GradientBoosting	0.696444	0.774640	0.302976
GaussianNB	0.696270	0.720749	0.294166

DecisionTree	0.680451	0.745533	0.286268
KNeighbors	0.669339	0.681268	0.267239
MLP	0.668511	0.783862	0.284014

El uso combinado de SMOTE y RandomUnderSampler mostró mejoras significativas en recall para todos los modelos durante el entrenamiento, con un buen nivel de precisión. En las pruebas, mostró mejoras en la capacidad de los modelos para detectar casos de diabetes, aunque algunos modelos aún enfrentaron desafíos en mantener altos niveles de precisión, sugiriendo que esta combinación puede requerir ajustes adicionales.

f) Random undersamplier y Oversampling

Por otra parte, se llevó a cabo la combinación de oversamplig y undersampling, duplicando y reduciendo los datos respectivamente. Se empleó RandomOverSampler para duplicar los datos de la clase minoritaria, mientras que RandomUnderSampler se utilizó para disminuir los datos de la clase mayoritaria. Los resultados obtenidos en relación con los conjuntos de entrenamiento originales fueron los siguientes.

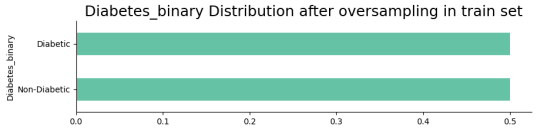

Fig. 18. Distribución de Diabetes_binary después de ROS en clase minoritaria del conjunto de datos

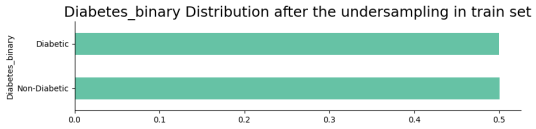

Fig. 19. Distribución de Diabetes_binary después de RUS en clase mayoritaria del conjunto de datos

TABLE XX: RandomUnderAndOverSampler dataset - Train Performance

Model	Accuracy	Recall	Precision
RandomForest	0.982571	0.993838	0.971960
KNeighbors	0.866529	0.968560	0.804545
CatBoost	0.764808	0.811722	0.742333
DecisionTree	0.763070	0.827828	0.733137
XgBoost	0.759818	0.806420	0.737911
MLP	0.741036	0.792577	0.718766
GradientBoosting	0.737420	0.776295	0.720557
GaussianNB	0.693162	0.635085	0.718922

TABLE XXI: RandomUnderAndOverSampler dataset - Test Performance

Model	Accuracy	Recall	Precision
GaussianNB	0.638714	0.483067	0.702422
MLP	0.627117	0.666310	0.618463
GradientBoosting	0.509239	0.028771	0.762770
CatBoost	0.503387	0.013786	0.709251
DecisionTree	0.499936	0.002869	0.638095
XgBoost	0.499314	0.000000	0.000000
KNeighbors	0.499314	0.000000	0.000000
RandomForest	0.499035	0.000557	0.333333

La combinación de RandomUnderSampler y RandomOverSampler mostró una mejora notable en precisión y recall durante el entrenamiento. Sin embargo, en el conjunto de pruebas, los resultados fueron mixtos, con algunos modelos alcanzando buenos niveles de recall pero a costa de la precisión. Esto sugiere que, aunque beneficiosa, esta estrategia necesita ajustes finos para evitar el sobreajuste y la pérdida de información crítica.

g) Técnica seleccionada

Después de evaluar los diversos modelos seleccionados y aplicar diferentes técnicas de oversampling y undersampling al conjunto de datos de entrenamiento, una estrategia particular destacó por su capacidad para equilibrar las clases y mejorar significativamente los resultados previos de los modelos. Esta estrategia logró obtener los puntajes más altos en las métricas de precisión, recall y accuracy, los cuales se sitúan entre el 73% y el 83% en todos los modelos evaluados como se refleja en la tabla XVII. Esta estrategia implicó la combinación de técnicas de oversampling y undersampling, utilizando SMOTE (Synthetic Minority Over-sampling Technique) y NearMiss, respectivamente. El uso de SMOTE permitió generar instancias sintéticas de la clase minoritaria, mientras que NierMiss redujo el número de instancias de la clase mayoritaria, logrando así un conjunto de datos más equilibrado para el entrenamiento de los modelos. Esta combinación de técnicas demostró ser efectiva para abordar el desbalance de clases y mejorar la capacidad predictiva de los modelos en la tarea de predicción de la diabetes.

C. Entrenamiento

Después de aplicar las técnicas de manejo de desequilibrio de clases, el conjunto de datos procesado se utiliza para entrenar los modelos seleccionados. Se divide el conjunto de datos en dos partes: el 70% se utiliza para el entrenamiento de los modelos y el 30% restante se reserva para realizar pruebas. Además, se emplea la técnica de validación cruzada, la cual consiste en dividir repetidamente el conjunto de datos en subconjuntos de entrenamiento y validación. Como resultado, las métricas de evaluación (como precision, recall y accuracy) se calculan en cada iteración de validación y luego se promedian para obtener una estimación general del rendimiento

del modelo. Esta metodología permite una evaluación robusta y objetiva del modelo en diferentes particiones de datos, lo que contribuye a una mejor comprensión de su rendimiento y generalización.

a) Naive-bayes Classifier

El modelo de Gaussian Naive-Bayes mostró los puntajes de precision, recall y accuracy más bajos de todos con 69.42%, 69.96% y 69.42% respectivamente. A demás, al examinar la matriz de confusión, se identifica un número considerablemente alto de falsos negativos y el el área bajo la cuerva fue de 0.77, esto sugiere que el rendimiento no es del todo bueno, sin embargo, el modelo es bastante efectivo y tiene una capacidad razonable para realizar predicciones correctas. Este hallazgo sugiere que, al menos por el momento, este modelo no ofrece una fiabilidad óptima.

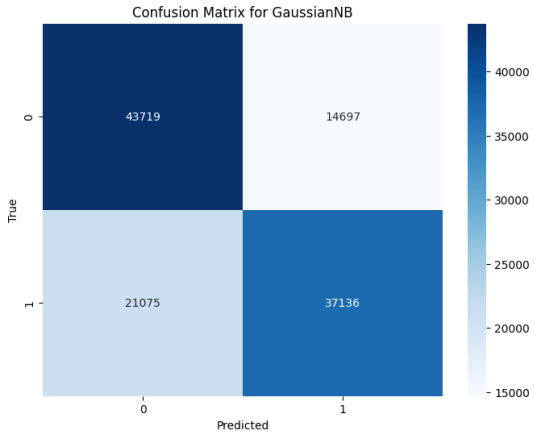

Fig. 20. Matrix de confusión GaussianNB

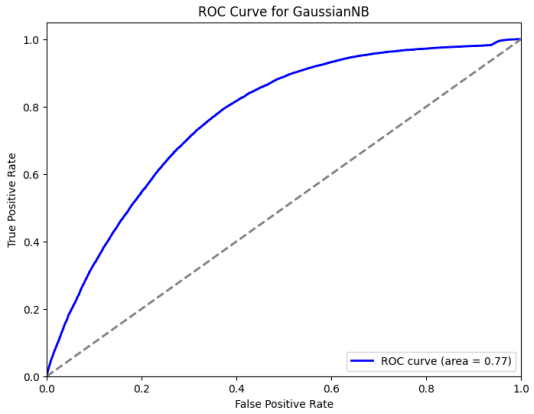

Fig. 21. Curva de ROC del GaussianNB

b) KNN

Con resultados de 76.74%, 77.72% y 76.76% para las métricas de accuracy, precisión y recall respectivamente, el modelo KNN no parece tan prometedor a primera vista. No obstante, al examinar la matriz de confusión, se observa que los falsos negativos son notablemente bajos. Esto sugiere que el modelo tiene potencial para mejorarse y obtener resultados más favorables. Y si vemos el área bajo la curva de ROC, esta fue de 0.88 lo sugiere que el modelo tiene una alta capacidad para distinguir entre las clases, lo que implica que es bastante preciso en sus predicciones.

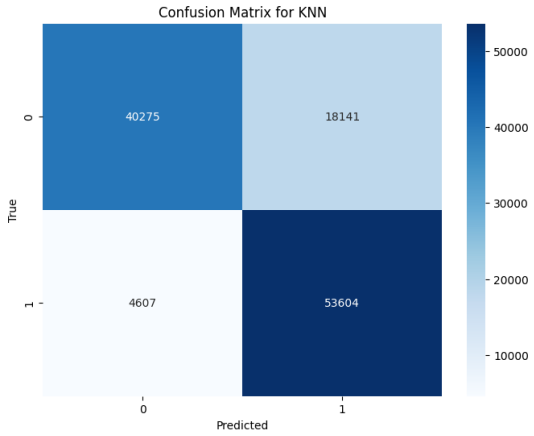

Fig. 22. Matrix de confusión KNN

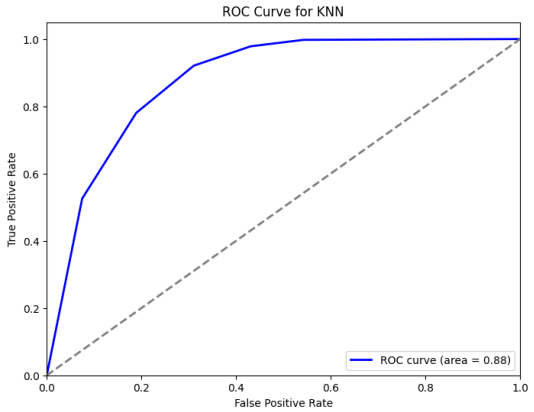

Fig. 23. Curva de ROC del KNN

c) Decision tree

El modelo de árbol de decisión mostró resultados moderados en términos de métricas de precisión, recall y accuracy, registrando un 73.97%, 73.75% y 73.74% respectivamente. Sin embargo, al observar la matriz de confusión, se evidencia que los valores de falsos negativos son significativamente altos, lo cual no inspira una confianza sustancial para el propósito perseguido. Sin embargo, el aréa bajo la curva de ROC no fue tan baja, teniendo un valor de 0.82 lo que es un buen indicio de la presión de este modelo.

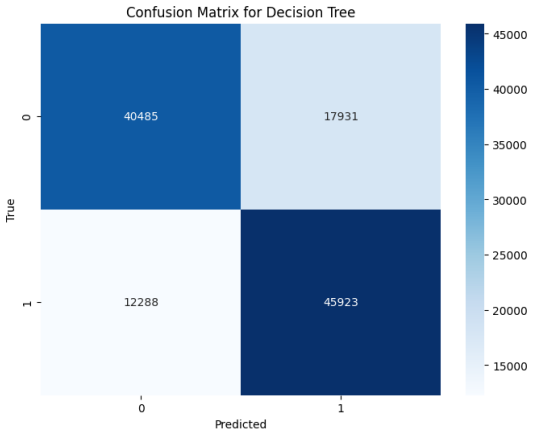

Fig. 24. Matrix de confusión Decision Tree

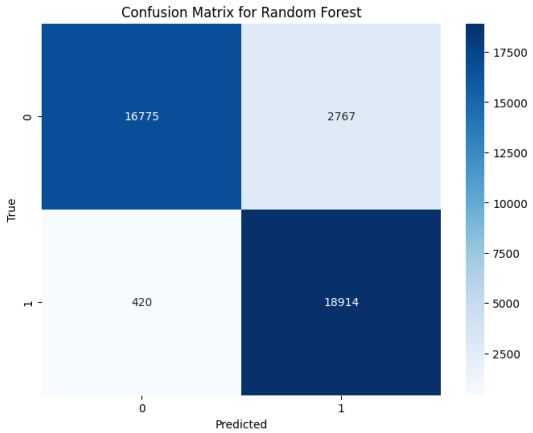

Fig. 26. Matrix de confusión Random Forest

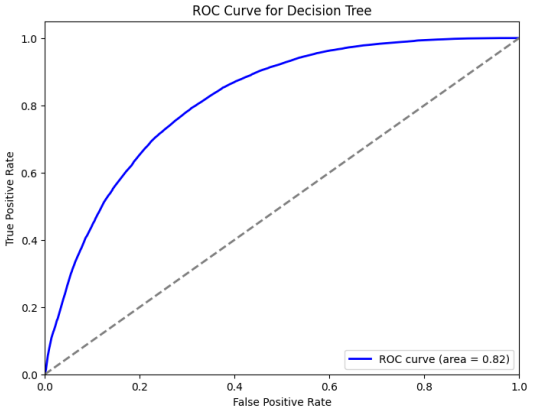

Fig. 25. Curva de ROC del Decision Tree

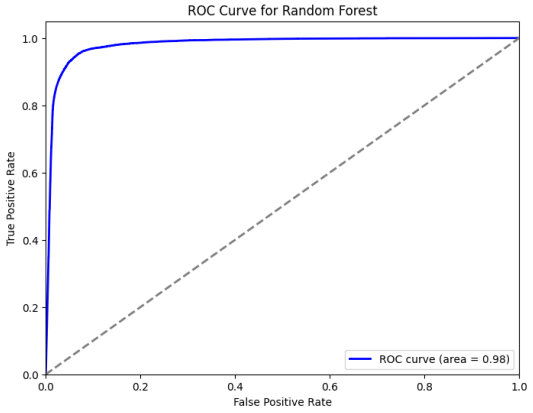

Fig. 27. Curva de ROC del Random Forest

d) Random Forest

El modelo de Random Forest exhibe uno de los mejores resultados en términos de métricas de precisión, recall y accuracy, siendo estas 89.08%, 88.38% y 88.38% respectivamente. A demás se puede observar de su matrix de confusión el bajo número de falsos negativos, y altos números de verdaderos positivos y falsos negativos. Su aréa bajo la curva de ROC es de 0.98 lo cual es un indicio de que el modelo es muy bueno prediciendo la diabetes o no diabetes.

e) MLP classifier

En el caso del MLP Classifier, se obtuvieron resultados de 73.79%, 73.96% y 73.79% para las métricas de accuracy, precisión y recall respectivamente. Estos resultados inicialmente generan una impresión positiva, indicando que el modelo está progresando en la dirección correcta. Además, su área bajo la curva es equivalente al del modelo de Decision Tree, 0.82, lo que es un buen indicio para la presión de tiene este modelo. No obstante, como se ha observado durante el análisis, el número de falsos negativos es considerable, lo que subraya la necesidad de seguir trabajando para mejorar su desempeño.

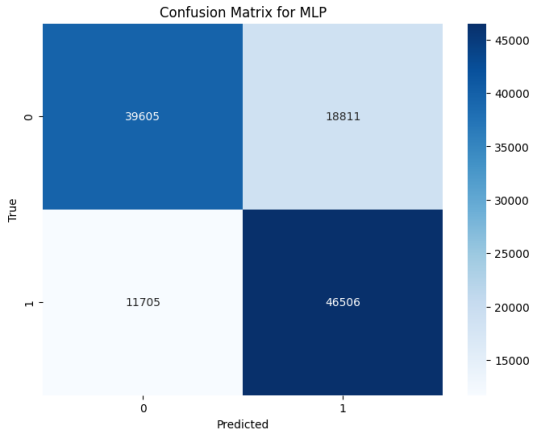

Fig. 28. Matrix de confusión MLP Classifier

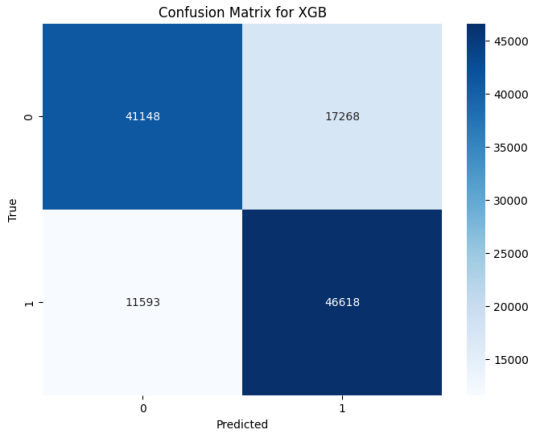

Fig. 30. Matrix de confusión XGB Classifier

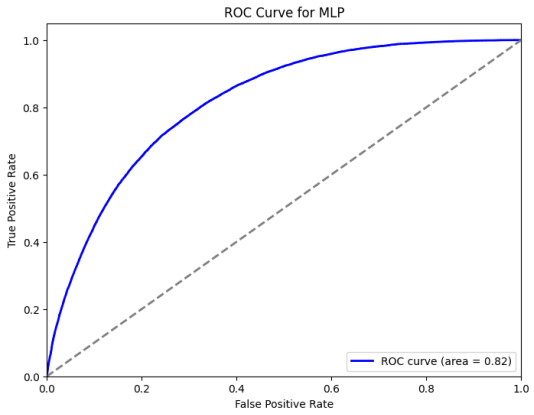

Fig. 29. Curva de ROC del MLP Classifier

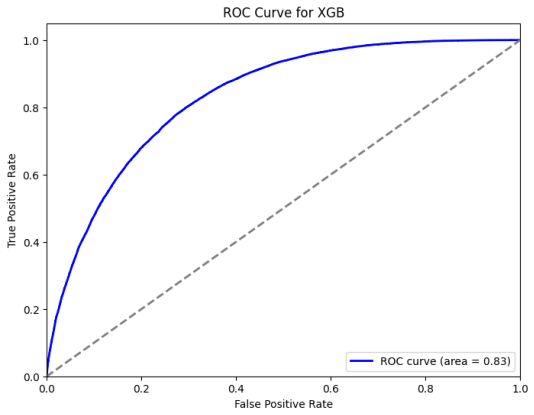

Fig. 31. Curva de ROC del XGB Classifier

f) XGB Classifier

En el caso del clasificador XGB, se registraron puntajes de 75.04%, 75.57% y 75.03% para las métricas de accuracy, precisión y recall respectivamente. Estos resultados, en principio, parecen prometedores, sugiriendo un avance positivo del modelo. De la misma manera, la precisión obtenida por el área bajo la curva de ROC es 0.83, lo que representa un buen indicio. Sin embargo, como se ha notado durante la evaluación, la cantidad de falsos negativos es significativa, lo que resalta la importancia de continuar mejorando su rendimiento.

D. Evaluación del modelo

Tomando los datos evaluación, se evaluará el rendimiento de cada uno de los algoritmos utilizados para su posterior comparación. Inicialmente, se entrenaron modelos de clasificación utilizando los conjuntos de datos sin realizar ajustes especiales en los hiperparámetros, solo aplicando la técnica de validación cruzada. Con el objetivo de mejorar la eficiencia y el rendimiento de estos modelos, se aplicaron técnicas de optimización de hiperparámetros, como GridSearchCV, para encontrar las configuraciones óptimas que maximizaran la precisión y el rendimiento general de los modelos. GridSearchCV es una técnica de búsqueda sistemática que evalúa exhaustivamente todas las combinaciones posibles de parámetros en la cuadrícula, con el propósito de encontrar la configuración óptima (Ahmad, Fatima, Ullah, Salah Saidi, & Imdadullah, 2022). Este enfoque ayuda a mitigar el riesgo de sobreajuste al proporcionar una estimación más precisa del rendimiento del modelo en datos no vistos (Ahmad et al., 2022). En conjunto, la combinación de validación cruzada y GridSearchCV constituye una estrategia sólida para optimizar los hiperparámetros del modelo y mejorar su generalización a nuevos datos (Ahmad et al., 2022).

Para la medición del rendimiento del modelo, se medirá su exactitud (*accuracy*), su precisión, *recall* (sensibilidad o tasa de verdaderos positivos), *F1-Score* y el *roc-auc*.

a) Random Forest

Para el caso del Random Forest, se define una cuadrícula de parámetros (`param_grid_rdf`) con diversas opciones para los hiperparámetros: el número de estimadores (`n_estimators`), las características máximas a considerar para la división (`max_features`), la profundidad máxima de los árboles (`max_depth`), la utilización de bootstrap para las muestras (`bootstrap`), y el peso de las clases en problemas de clasificación desbalanceada (`class_weight`). Estos parámetros se seleccionaron para explorar una variedad de configuraciones que pueden influir significativamente en el rendimiento del modelo.

- Número de estimadores (`n_estimators`): Este parámetro define el número de árboles en el bosque. Al probar con 100 y 200 estimadores, se busca un equilibrio entre precisión y eficiencia computacional. Un mayor número de árboles generalmente mejora la estabilidad y precisión del modelo, aunque con un costo computacional mayor.
- Características máximas para la división (`max_features`): Las opciones 'auto' y 'sqrt' permiten probar cómo el uso de diferentes cantidades de características en cada división afecta el rendimiento del modelo. 'Auto' selecciona todas las características, mientras que 'sqrt' selecciona la raíz cuadrada del número total de características, lo que puede reducir la correlación entre árboles y mejorar la generalización.
- Profundidad máxima de los árboles (`max_depth`): Establecer límites en la profundidad máxima (10, 20, y sin límite) ayuda a controlar el sobreajuste. Profundidades mayores permiten que los árboles capten relaciones más complejas en los datos, pero también aumentan el riesgo de sobreajuste. Evaluar varias profundidades permite encontrar un balance óptimo.
- Utilización de bootstrap para las muestras (`bootstrap`): Este parámetro determina si se utilizarán muestras bootstrap para construir cada árbol. Probar tanto True como False ayuda a evaluar si la resampling mejora la robustez del modelo.
- Peso de las clases (`class_weight`): En problemas de clasificación desbalanceada, ajustar los pesos de las clases (por ejemplo, 0: 1, 1: 12) es crucial para evitar que el modelo favorezca la clase mayoritaria. Al asignar un peso mayor a la clase minoritaria, el modelo se incentiva a prestar más atención a estas instancias, mejorando el rendimiento en conjuntos de datos desbalanceados.

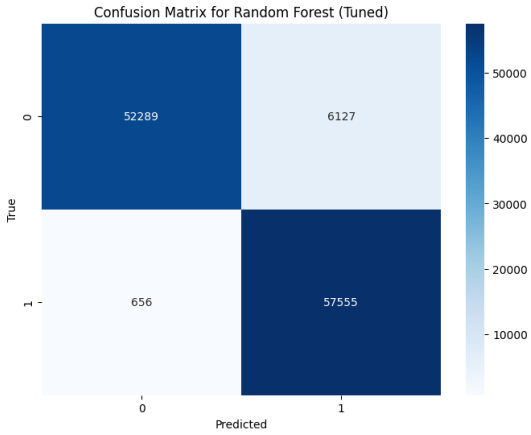

Fig. 32. Matrix de confusión Random Forest Ajustado

- Accuracy Promedio: 90.62%
- Precision Promedio: 91.07%
- Recall Promedio: 90.62%

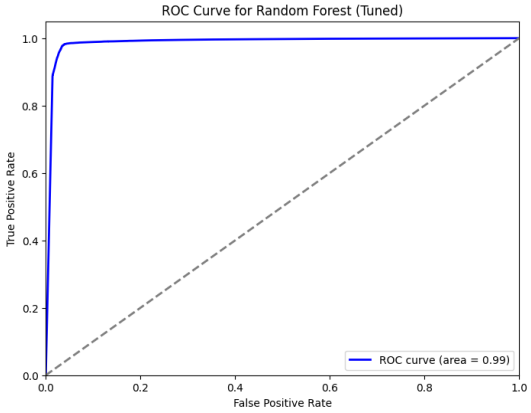

Fig. 33. Curva de ROC del Random Forest Ajustado

Este proceso involucró el ajuste de 24 combinaciones de parámetros, evaluadas a través de 3 pliegues, lo que resultó en un total de 72 ajustes. La búsqueda arrojó los siguientes hiperparámetros como los mejores para nuestro conjunto de datos:

- `bootstrap`: `False`
- `class_weight`: `{0: 1, 1: 12}`
- `max_depth`: `None`
- `max_features`: `auto`
- `n_estimators`: `100`

Estos resultados indican que el modelo de Random Forest obtuvo su mejor rendimiento sin el uso de muestreo *bootstrap*, asignando un peso significativo a la clase minoritaria (1:12), sin restricción en la profundidad máxima de los árboles, considerando todas las características disponibles para las divisiones (`auto`), y con 100 árboles en el bosque.

La importancia de estos resultados radica en que los hiperparámetros óptimos permiten al modelo manejar mejor el desbalance de clases en nuestros datos, mejorar su capacidad

de generalización y reducir el sobreajuste. La configuración sin límite en la profundidad de los árboles y sin *bootstrap* sugiere que el modelo puede capturar relaciones complejas sin necesidad de muestreo repetitivo, lo que es beneficioso dado nuestro conjunto de datos específico.

Al implementar estos hiperparámetros, el modelo de Random Forest no solo alcanzó una alta precisión (*accuracy*) sino que también mostró mejoras en otras métricas como *recall* y precisión, comparables a las obtenidas por otros modelos en nuestro estudio. Tanto la área bajo la curva de ROC(0.99) como los resultados vistos en la matrix de confusión son fuertes indicadores de que el modelo puede predecir los casos de diabetes o no diabetes con precisión. Esta afinación cuidadosa de los hiperparámetros demuestra la eficacia de *GridSearchCV* en la optimización del rendimiento del modelo, lo cual es esencial para garantizar un desempeño robusto en aplicaciones del mundo real.

b) Decision Tree

Para el árbol de decisión (Decision Tree), se define una cuadrícula de parámetros (`param_grid_dt`) con varias opciones para los hiperparámetros. Estos incluyen la profundidad máxima del árbol (`max_depth`), las características máximas a considerar para la división (`max_features`), el número mínimo de muestras requeridas para dividir un nodo interno (`min_samples_split`), el número mínimo de muestras requeridas para estar en un nodo hoja (`min_samples_leaf`), y el peso de las clases en problemas de clasificación desbalanceada (`class_weight`). Estos parámetros se seleccionaron cuidadosamente para explorar una gama de configuraciones que pueden afectar significativamente el rendimiento del modelo.

- Profundidad máxima del árbol (`max_depth`): Controla la profundidad máxima del árbol. Se exploran las opciones de 10, 20 y sin límite de profundidad para equilibrar el riesgo de sobreajuste y subajuste.
- Características máximas para la división (`max_features`): Define el número máximo de características a considerar al buscar la mejor división en cada nodo. Se prueban 'auto', 'sqrt' y 'log2' para determinar qué enfoque es más efectivo en la partición del espacio de características.
- mínimo de muestras para dividir un nodo (`min_samples_split`): Establece el número mínimo de muestras requeridas para dividir un nodo interno. Este parámetro influye en la capacidad del modelo para capturar relaciones más sutiles en los datos sin ajustarse demasiado.
- mínimo de muestras en un nodo hoja (`min_samples_leaf`): Especifica el número mínimo de muestras requeridas para estar en un nodo hoja. Controla la profundidad de la partición del árbol y evita divisiones que resulten en nodos hoja con un número muy pequeño de muestras.
- de las clases (`class_weight`): Ajusta los pesos de las clases para manejar conjuntos de datos desbalanceados. Se utiliza para otorgar mayor importancia a las instancias de la clase minoritaria, lo que ayuda a prevenir el sesgo hacia la clase mayoritaria durante el entrenamiento del árbol de decisión.

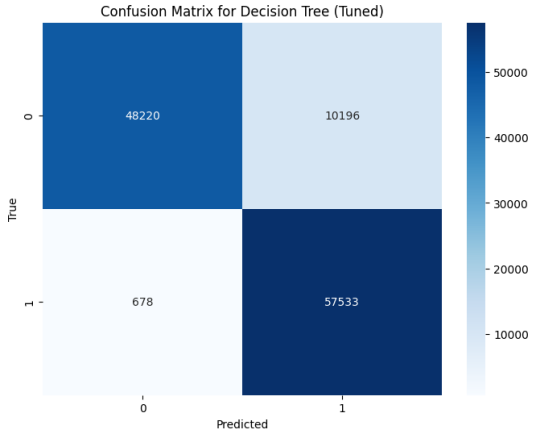

Fig. 34. Matrix de confusión Decision Tree Ajustado

- Accuracy Promedio: 87.05%
- Precision Promedio: 88.21%
- Recall Promedio: 87.05%

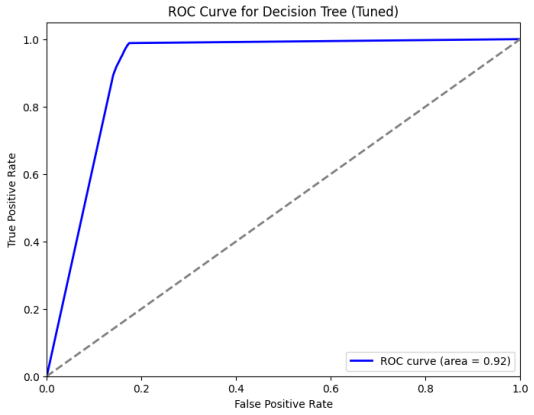

Fig. 35. Curva de ROC del Decision Tree Ajustado

Este proceso involucró la evaluación de 81 combinaciones de parámetros a través de 3 pliegues de validación cruzada, resultando en un total de 243 ajustes. Los mejores hiperparámetros identificados para nuestro conjunto de datos fueron los siguientes:

- `class_weight`: `{0: 1, 1: 12}`
- `max_depth`: `None`
- `max_features`: `auto`
- `min_samples_leaf`: `1`
- `min_samples_split`: `2`

Estos resultados indican que el modelo de Decision Tree alcanzó su mejor rendimiento sin imponer restricciones en la profundidad máxima del árbol (`max_depth`), utilizando todas las características disponibles para las divisiones (`max_features` configurado como `auto`), y con los valores

mínimos posibles para las muestras requeridas tanto para dividir un nodo (`min_samples_split`) como para estar en un nodo hoja (`min_samples_leaf`).

La importancia de estos hiperparámetros óptimos se refleja en la capacidad del modelo para manejar desequilibrios de clase de manera efectiva, mejorar la precisión en la predicción de la clase minoritaria y mantener una estructura de árbol simple y eficiente. La falta de límite en la profundidad del árbol y el uso de todas las características disponibles permiten que el modelo capture relaciones complejas dentro de los datos sin sobreajuste, dado el ajuste fino a través de los pliegues de validación cruzada.

Implementar estos hiperparámetros permitió que el modelo de Decision Tree alcanzara un alto rendimiento en términos de precisión (*accuracy*), *recall* y precisión, con resultados comparables a los obtenidos por otros modelos en nuestro análisis. Los resultados de la matrix de confusión indican que el modelo es capaz de predecir con mucha precisión los casos de diabetes y no diabetes. Del mismo modo, el área bajo la curva de ROC (0.92) demuestra el potencial del modelo para predecir con suficiente precisión los casos de diabetes tipo 2.

c) KNN

Para el clasificador K-Nearest Neighbors (KNN), se define una cuadrícula de parámetros (`param_grid_knn`) con varias opciones para los hiperparámetros. Estos incluyen el número de vecinos (`n_neighbors`), el algoritmo utilizado para calcular los vecinos más cercanos (`algorithm`), y la métrica de distancia utilizada (`p`). Estos parámetros se seleccionaron para explorar una variedad de configuraciones que pueden afectar significativamente el rendimiento del modelo.

- Número de vecinos (`n_neighbors`): Controla la cantidad de vecinos considerados al realizar una predicción. Se prueban opciones de 3, 5 y 10 vecinos para determinar cuál ofrece el mejor equilibrio entre sesgo y varianza en el modelo.
- Algoritmo (`algorithm`): Especifica el algoritmo utilizado para calcular los vecinos más cercanos. Se exploran las opciones 'auto' y 'ball_tree' para determinar cuál es más eficiente en función del conjunto de datos dado.
- Métrica de distancia (`p`): Define la métrica de distancia utilizada para medir la cercanía entre puntos. Se prueba con las opciones 1 y 2, que corresponden a la distancia de Manhattan y la distancia Euclidiana, respectivamente.

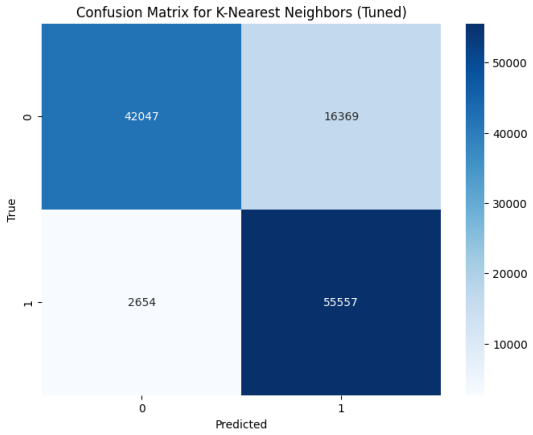

Fig. 36. Matrix de confusión KNN Ajustado

- Accuracy Promedio: 79.73%
- Precision Promedio: 81.21%
- Recall Promedio: 79.72%

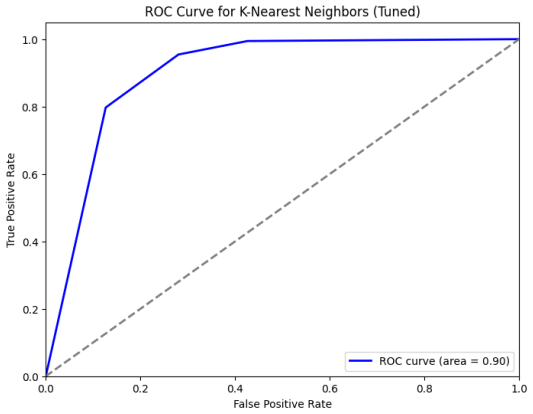

Fig. 37. Curva de ROC del KNN Ajustado

Este proceso involucró la evaluación de 12 combinaciones de parámetros a través de 3 pliegues de validación cruzada, resultando en un total de 36 ajustes. Los mejores hiperparámetros identificados para nuestro conjunto de datos fueron los siguientes:

- `algorithm`: `auto`
- `n_neighbors`: 3
- `p`: 1

Estos resultados indican que el modelo de KNN obtuvo su mejor rendimiento utilizando el algoritmo `auto`, que selecciona automáticamente el algoritmo más apropiado en función del conjunto de datos. El número óptimo de vecinos (`n_neighbors`) se determinó que es 3, lo que implica que para predecir la clase de un punto, el modelo considera las tres instancias más cercanas en el espacio de características. El parámetro `p` se estableció en 1, lo que corresponde a la distancia de Manhattan (L1), sugiriendo que esta métrica de distancia es la más adecuada para nuestro conjunto de datos.

La selección de estos hiperparámetros es crucial ya que afecta directamente la precisión y eficiencia del modelo. Utilizar el algoritmo `auto` permite al modelo adaptarse dinámicamente al mejor enfoque para el conjunto de datos, mientras que la configuración de 3 vecinos asegura un equilibrio entre suavidad y precisión en la clasificación. La distancia de Manhattan es preferida en este caso, probablemente debido a la naturaleza de los datos y las relaciones entre las características.

Implementar estos hiperparámetros permitió que el modelo de KNN alcanzara un rendimiento competitivo en términos de exactitud (*accuracy*), *recall* y precisión. Además, los valores de verdaderos positivos y falsos positivos (matriz de confusión) y el área bajo la curva ROC respaldan su viabilidad como uno de los mejores modelos para nuestra aplicación.

d) MLP Classifier

Para el clasificador MLP (Perceptrón Multicapa), se define una cuadrícula de parámetros (`param_grid_mlp`) con varias opciones para los hiperparámetros. Estos incluyen el tamaño de las capas ocultas (`hidden_layer_sizes`), la función de activación (`activation`), el algoritmo de optimización (`solver`), y el parámetro de regularización (`alpha`). Estos parámetros se seleccionaron para explorar una variedad de configuraciones que pueden afectar significativamente el rendimiento del modelo.

- Tamaño de las capas ocultas (`hidden_layer_sizes`): Especifica el número de neuronas en cada capa oculta. Se prueban tres configuraciones diferentes: una capa oculta con 50 neuronas, una capa oculta con 100 neuronas y dos capas ocultas con 100 y 50 neuronas, respectivamente.
- Función de activación (`activation`): Define la función de activación utilizada en las capas ocultas. Se exploran las opciones 'tanh' y 'relu' para determinar cuál proporciona el mejor rendimiento en la red neuronal.
- Algoritmo de optimización (`solver`): Especifica el algoritmo utilizado para optimizar los pesos de la red. Se prueban los algoritmos 'sgd' (Descenso de Gradiente Estocástico) y 'adam' (Adaptative Moment Estimation) para encontrar el más adecuado para el conjunto de datos dado.
- Parámetro de regularización (`alpha`): Controla la fuerza de regularización en la red neuronal. Se exploran los valores de 0.0001 y 0.001 para evitar el sobreajuste.

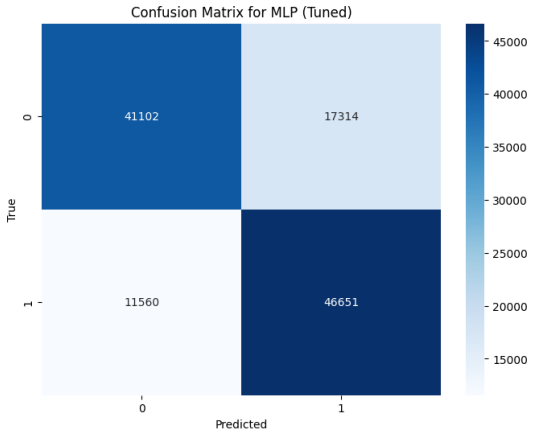

Fig. 38. Matrix de confusión MLP Ajustado

- Accuracy Promedio: 74.50%
- Precision Promedio: 74.63%
- Recall Promedio: 74.50%

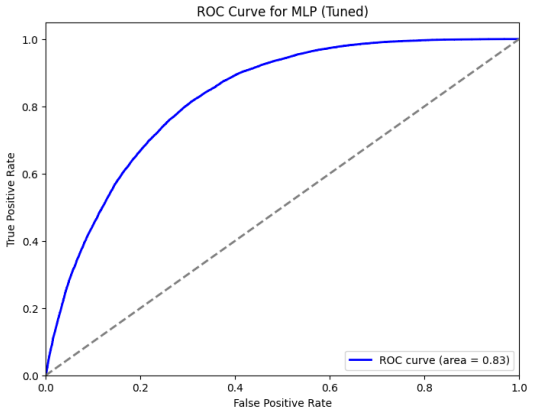

Fig. 39. Curva de ROC del MLP Ajustado

Este proceso involucró la evaluación de 24 combinaciones de parámetros a través de 3 pliegues de validación cruzada, resultando en un total de 72 ajustes. Los mejores hiperparámetros identificados para nuestro conjunto de datos fueron los siguientes:

- `activation`: `tanh`
- `alpha`: `0.0001`
- `hidden_layer_sizes`: `(100, 50)`
- `solver`: `adam`

En este procedimiento, se utilizando la función de activación `tanh`, que puede capturar relaciones no lineales complejas en los datos. El parámetro de regularización `alpha` se estableció en 0.0001, ayudando a prevenir el sobreajuste al penalizar los pesos grandes en la red. La arquitectura de la red neuronal óptima consistió en dos capas ocultas con 100 y 50 neuronas respectivamente (`hidden_layer_sizes`), lo que proporciona una buena capacidad de aprendizaje sin ser excesivamente complejo. El algoritmo de optimización `adam` se seleccionó por su eficiencia y adaptabilidad durante

el entrenamiento. La función de activación `tanh` es adecuada para capturar variaciones en los datos, mientras que el valor de `alpha` ayuda a regularizar el modelo. La estructura de las capas ocultas permite al modelo aprender representaciones profundas de los datos, y el optimizador `adam` asegura una convergencia rápida y robusta durante el entrenamiento.

A pesar de implementar estos hiperparámetros, se pudo observar que el modelo de MLPClassifier no alcanzo un rendimiento competitivo en términos de precisión (*accuracy*), *recall* y precisión. Sus valores en la matrix de confusión son tampoco fueron los deseados, teniendo un número muy elevado de falsos negativos. Así mismo, el área bajo la curva de ROC no es muy diferente a la obtenida antes de realizar las modificaciones a los hiperparametros; 0.82 y 0.83 respectivamente.

e) XGB Classifier

Para el clasificador XGBoost (Extreme Gradient Boosting), se define una cuadrícula de parámetros (`param_grid_xgb`) con varias opciones para los hiperparámetros. Estos incluyen el número de estimadores (`n_estimators`), la profundidad máxima de los árboles (`max_depth`), la tasa de aprendizaje (`learning_rate`), la proporción de muestras utilizadas en cada árbol (`subsample`), y la proporción de características utilizadas en cada árbol (`colsample_bytree`). Estos parámetros se seleccionaron para explorar una variedad de configuraciones que pueden afectar significativamente el rendimiento del modelo.

- Número de estimadores (`n_estimators`): Especifica el número de árboles que se construirán en el ensamble. Se prueban opciones de 100 y 200 estimadores para encontrar un equilibrio entre el tiempo de entrenamiento y el rendimiento del modelo.
- Profundidad máxima de los árboles (`max_depth`): Controla la profundidad máxima de cada árbol en el ensamble. Se exploran las opciones de 3, 6 y 9 niveles de profundidad para capturar relaciones más complejas en los datos.
- Tasa de aprendizaje (`learning_rate`): Determina la contribución de cada árbol al ensamble. Se prueban tasas de aprendizaje de 0.01, 0.1 y 0.2 para controlar la velocidad de aprendizaje del modelo.
- Proporción de muestras utilizadas en cada árbol (`subsample`): Especifica la proporción de muestras utilizadas para entrenar cada árbol. Se exploran opciones de 0.8 y 1.0 para determinar la cantidad de aleatorización en el entrenamiento.
- Proporción de características utilizadas en cada árbol (`colsample_bytree`): Controla la proporción de características utilizadas para entrenar cada árbol. Se prueban opciones de 0.6, 0.8 y 1.0 para determinar la diversidad de los árboles en el ensamble.

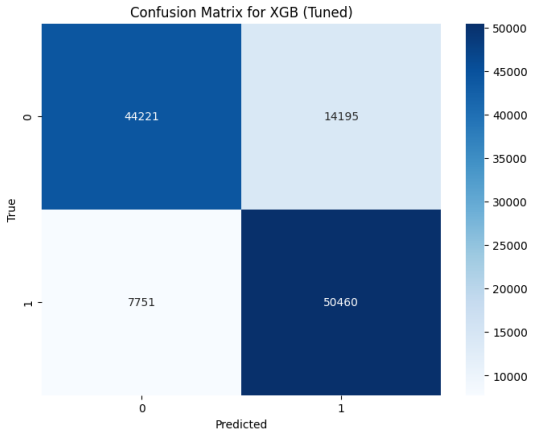

Fig. 40. Matrix de confusión XBG Ajustado

- Accuracy Promedio: 80.18%
- Precision Promedio: 80.54%
- Recall Promedio: 80.18%

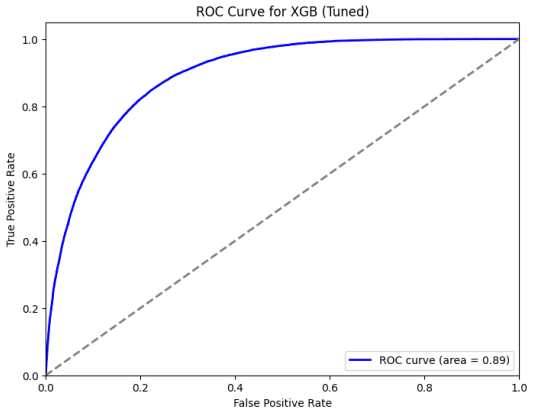

Fig. 41. Curva de ROC del XGB Ajustado

Este proceso involucró la evaluación de 108 combinaciones de parámetros a través de 3 pliegues de validación cruzada, resultando en un total de 324 ajustes. Los mejores hiperparámetros identificados para nuestro conjunto de datos fueron los siguientes:

- `colsample_bytree`: `1.0`
- `learning_rate`: `0.2`
- `max_depth`: `9`
- `n_estimators`: `200`
- `subsample`: `0.8`

Estos resultados indican que el modelo de XGB Classifier alcanzó su mejor rendimiento utilizando una proporción de columnas por árbol (`colsample_bytree`) de 1.0, lo que significa que todas las características están disponibles para cada árbol. La tasa de aprendizaje (`learning_rate`) se estableció en 0.2, lo que permite que el modelo se ajuste de manera más agresiva, pero aún conservando estabilidad. La profundidad máxima de los árboles (`max_depth`) se configuró en 9, proporcionando una gran capacidad para capturar

relaciones complejas en los datos. El número de estimadores (`n_estimators`) se determinó en 200, lo que da al modelo suficiente capacidad de aprendizaje sin ser excesivamente complejo. La proporción de muestra (`subsample`) se estableció en 0.8, ayudando a prevenir el sobreajuste al usar solo una fracción de los datos de entrenamiento para cada árbol.

La implementación de estos hiperparámetros permitió que el modelo XGB Classifier lograra un rendimiento notable en términos de exactitud (*accuracy*), *recall* y precisión. Además, el área bajo la curva ROC de 0.89 respalda su capacidad para distinguir entre clases, consolidándolo como uno de los modelos más efectivos para nuestra aplicación.

f) Naive-bayes Classifier

Para el clasificador Gaussian Naive Bayes (GaussianNB), se define una cuadrícula de parámetros (`param_grid_gauss_nb`) con una opción para el hiperparámetro. Este hiperparámetro es `var_smoothing`, que controla la suavización aditiva (Laplace / Lidstone) de la varianza. Se seleccionan varios valores de `var_smoothing` para explorar cómo esta suavización afecta el rendimiento del modelo.

- Suavización de la varianza (`var_smoothing`): Controla la magnitud de la suavización aditiva aplicada a la varianza de las características. Se prueban varios valores de suavización, incluyendo 10^{-9}, 10^{-8}, 10^{-7}, 10^{-6} y 10^{-5}, para determinar el efecto de esta suavización en el modelo GaussianNB.

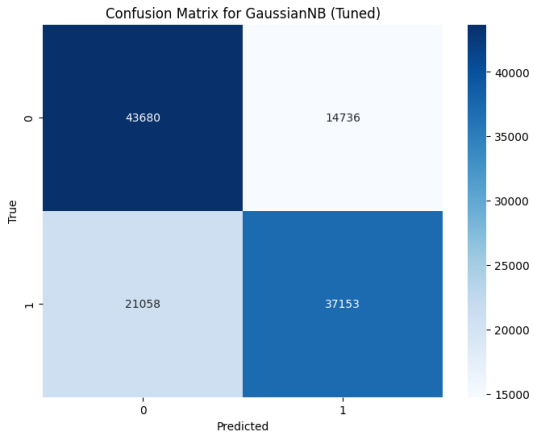

Fig. 42. Matrix de confusión GaussianNB Ajustado

- Accuracy Promedio: 69.33%
- Precision Promedio: 69.59%
- Recall Promedio: 69.34%

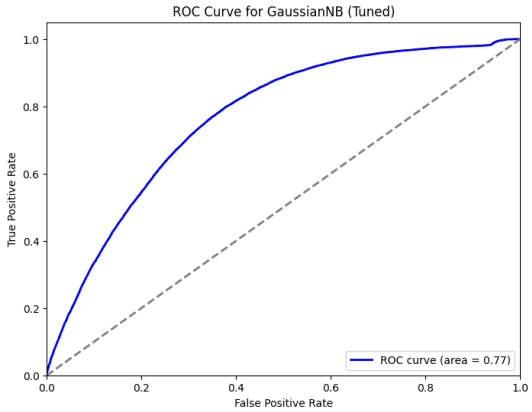

Fig. 43. Curva de ROC del GaussianNB Ajustado

Este proceso involucró la evaluación de 5 combinaciones de parámetros a través de 3 pliegues de validación cruzada, resultando en un total de 15 ajustes. El mejor hiperparámetro identificado para nuestro conjunto de datos fue el siguiente:

- `var_smoothing`: `1e-09`

Es importante destacar que, a pesar de realizar ajustes en los hiperparámetros, los resultados obtenidos fueron muy similares a los logrados mediante validación cruzada sin ajuste de hiperparámetros. Esto sugiere que el modelo de GaussianNB es relativamente robusto respecto a su único hiperparámetro ajustable, `var_smoothing`, el cual controla el suavizado aplicado a la variancia de cada característica.

La búsqueda de hiperparámetros identificó que un valor de `1e-09` para `var_smoothing` proporciona un rendimiento óptimo. Sin embargo, la similitud de los resultados con los obtenidos sin ajuste específico indica que el modelo de *GaussianNB* tiene una sensibilidad limitada a este parámetro en nuestro conjunto de datos.

La implementación de estos hiperparámetros permitió verificar que el modelo GaussianNB mantiene un rendimiento consistente en términos de exactitud (*accuracy*), *recall* y precisión, independientemente de ajustes minuciosos en `var_smoothing`. La curva ROC se mantuvo constante, con un valor de 0.77 antes y después de los ajustes.

Basado en el análisis previo, queda claro que la búsqueda y ajuste de hiperparámetros ha conducido a mejoras notables en la eficiencia y el rendimiento de los modelos en comparación con sus configuraciones predeterminadas. Sin embargo, es crucial destacar que, aunque esta optimización ha sido efectiva en varios modelos, no todos han experimentado mejoras significativas. Por ejemplo, el Gaussian Naive-Bayes no demostró mejoras sustanciales, incluso después de ajustar sus hiperparámetros. Esto subraya la importancia de considerar la pertinencia de la optimización de hiperparámetros para cada modelo y conjunto de datos específico, reconociendo que su impacto puede variar dependiendo de diversos factores.

E. Despliegue o implementación

La fase de despliegue se centró en llevar los modelos de Machine Learning a una aplicación web práctica y accesible

para usuarios finales, como lo sugiere Studer et al. (2021).. La interfaz de usuario se desarrolló con Streamlit y se desplegó en Streamlit Community Cloud. El backend se implementó utilizando FastAPI y se alojó en Koyeb, una plataforma serverless que facilita el despliegue y escalabilidad. Los modelos se almacenaron en un Storage Account de Azure, asegurando un acceso seguro y confiable. Esta arquitectura permite la integración eficiente de los modelos con la aplicación web, facilitando evaluaciones de riesgo de diabetes tipo 2 de manera precisa y sencilla. La implementación busca que los usuarios finales puedan utilizar el modelo de forma intuitiva, beneficiándose de las predicciones precisas para la detección temprana de la enfermedad.

VI. Marco Conceptual

A. Diabetes

La diabetes puede ser clasificada en varias categorías. La **diabetes tipo 1** es generada por la destrucción auto-inmune de las células β, causando deficiencia de insulina (ElSayed et al., 2022). La **diabetes tipo 2** ocurre a una perdida progresiva (no auto-inmune) de segregación de insulina las células β (ElSayed et al., 2022). Además de los anteriores, se incluye la diabetes gestacional, que se diagnostica en el segundo o tercer trimestre de embarazo, y otros tipos de diabetes ocurren por otras causas, como la diabetes inducida por químicos, diabetes neonatal, entre otras (ElSayed et al., 2022). La **prediabetes** no es una entidad clínica por su cuenta, sino un factor de riesgo para denominar a individuos cuyos niveles de glucosa no entran en la categoría de diabetes pero tienen un metabolismo anormal de clorhidratos que puede progresar en diabetes y enfermedades cardiovasculares (ElSayed et al., 2022).

B. Sistemas de apoyo a la decisión clínica

Los sistemas de apoyo a la decisión clínica (CDSS por sus siglas en ingles Clinical Decision Support Systems) son utilizados para asistir a los practicantes de salud en la toma de decisiones, tomando como base evidencias obtenidas mediante técnicas de ML (Machine Learning) y DL (Deep Learning) (Gracious et al., 2023). El uso de Ml en la asistencia de toma de decisiones médicas, se busca minimizar los errores de diagnóstico y tratamiento (Arun Bhavsar et al., 2021). Sin embargo, estos sistemas no pueden remplazar a los practicantes de salud, solo asistirlos en la toma de decisiones (Arun Bhavsar et al., 2021).

C. Machine Learning

El Machine Learning tiene como objetivo la creación de algoritmos que puedan utilizar datos y utilizarlos para aprender automáticamente (sin asistencia humana) a hacer predicciones y juicios (Gracious et al., 2023; Hamsagayathri & Vigneshwaran, 2021). El principal objetivo es crear un sistema con la capacidad aprender y mejorar sin la asistencia humana (Hamsagayathri & Vigneshwaran, 2021). A diferencia de los acercamientos convencionales, en el cual se programan el modelo en su totalidad, ML permite aprender de experiencias pasadas para mejorar (Arun Bhavsar et al., 2021). Para ello, el modelo aprende del mapeo de predictores (características de entrada) y el objetivo (variable salida), y lo generaliza para funcionar con datos no observados (Arun Bhavsar et al., 2021). ML tiene un gran impacto en diferentes áreas (Park et al., 2021) y aplicaciones donde juega un rol vital, desde procesamiento de lenguaje natural hasta diagnóstico de enfermedades (Hamsagayathri & Vigneshwaran, 2021). Existen diferentes técnicas de ML que pueden ser agrupadas en aprendizaje supervisado, no supervisado y por refuerzo (Gracious et al., 2023; Dahiwade, Patle, & Meshram, 2019; Ahsan, Luna, & Siddique, 2022).

El aprendizaje supervisado asociar los datos de entrada con las etiquetas y aprende este mapeo para hacer predicciones o juicios (Gracious et al., 2023). Técnicas bajo la categoría de aprendizaje supervisado incluyen regresión linear (LR), Support Vector Machine (SVM), Random Forest (RD) y Decision Tree (DT), que son utilizadas en el área de la salud para la categorización de enfermedades, predicción de riesgos, entre otros (Gracious et al., 2023).

El aprendizaje no supervisado aprende de datos de entada no etiquetados sin una salida en particular para buscar patrones ocultos, agrupar datos relacionados y descubrir estructuras subyacentes (Gracious et al., 2023). Técnicas de aprendizaje no supervisado tienes la capacidad de agrupar basado en características similares, con esto, se puede mejorar el tratamiento y terapias personalizadas (Gracious et al., 2023). Técnicas bajo la categoría de aprendizaje no supervisado incluyen el agrupamiento k-medias, agrupamiento jerárquico y modelo de mezcla gaussiana (Gracious et al., 2023).

El aprendizaje por refuerzo se basa en aprender con fallo y error para obtener retroalimentación (en formas de recompensas o penalizaciones) para maximizar las recompensas (Gracious et al., 2023). En el área de la salud, se utiliza para la optimización de terapia y tratamiento adaptativo (Gracious et al., 2023).

Además de los acercamientos anteriores, también se habla del aprendizaje profundo (Deep learning; DL) como una rama de ML (Gracious et al., 2023). En DL, se modelan relaciones complejas y se extraen características de alto nivel utilizando redes neuronales con muchas capas (Gracious et al., 2023). Estas redes pueden aprender jerarquías de datos permitiendo capturar patrones y conexiones complejas (Gracious et al., 2023).

D. Herramientas de desarrollo de modelos

1) Kaggle

Kaggle es una plataforma en línea que reúne a una comunidad global de científicos de datos y entusiastas del aprendizaje automático. Esta plataforma proporciona acceso a conjuntos de datos públicos, herramientas de análisis de datos y competiciones de ciencia de datos. Los usuarios de Kaggle pueden explorar datos, colaborar en proyectos, participar en desafíos de ciencia de datos y competir en competiciones de aprendizaje automático patrocinadas por empresas y organizaciones.

2) BigQuery

BigQuery es una plataforma de almacenamiento de datos empresariales completamente gestionada que facilita la gestión y análisis de datos gracias a sus características integradas, como el aprendizaje automático, el análisis geoespacial y la inteligencia empresarial. La arquitectura sin servidores de BigQuery permite realizar consultas en SQL para abordar las cuestiones más complejas de una organización sin la necesidad de gestionar la infraestructura. Además, las consultas federadas posibilitan la lectura de datos desde fuentes externas, mientras que la funcionalidad de transmisión permite actualizaciones de datos continuas. Con su motor de análisis distribuido y escalable, BigQuery es capaz de procesar terabytes de datos en cuestión de segundos y petabytes en minutos.

3) ScikitLearn

Scikit-learn (Sklearn) es una biblioteca esencial en el ámbito del aprendizaje automático dentro del entorno de programación Python. Su utilidad radica en proporcionar una amplia gama de herramientas eficientes y de fácil acceso para una variedad de tareas de modelado estadístico. Desde clasificación y regresión hasta agrupamiento y reducción de dimensionalidad, Sklearn ofrece una interfaz coherente y fácil de usar en Python, lo que facilita su aplicación en diversos proyectos de investigación y desarrollo.

E. Algoritmos de Machine Learning

1) Naive-bayes Classifier

BC es uno de los algoritmos de clasificación más poderos (Hamsagayathri & Vigneshwaran, 2021). El BC utiliza técnicas de modelado probabilísticas para la representación de variables y dependencias condicionales entre ellas, permitiendo crear modelos para hacer predicciones rápidas (Arun Bhavsar et al., 2021; Hamsagayathri & Vigneshwaran, 2021). Entre los clasificadores bayesianos, se destaca Naïve Bayes (NB) (Arun Bhavsar et al., 2021; Hamsagayathri & Vigneshwaran, 2021).

2) Decision tree

El algoritmo de árboles de decisión, una técnica fundamental dentro del aprendizaje supervisado, es una herramienta versátil utilizada tanto para problemas de regresión como de clasificación en inteligencia artificial y minería de datos (Hamsagayathri & Vigneshwaran, 2021). Este método se basa en la construcción de un árbol jerárquico donde cada nodo representa una decisión tomada en función de una característica específica del conjunto de datos (Caballé-Cervigón et al., 2020). Esto se lleva a cabo mediante el análisis de la importancia relativa de cada variable, lo que permite dividir el conjunto de datos de manera recursiva hasta alcanzar nodos hoja que contienen las clasificaciones finales (Caballé-Cervigón et al., 2020). Vale destacar que la complejidad del árbol puede variar, con árboles gruesos, medianos y finos que determinan el número máximo de divisiones permitidas en cada nodo (Keniya et al., 2020).

3) Random Forest

RF es un algoritmo de clasificación que contine arboles de decisión (Hamsagayathri & Vigneshwaran, 2021). Este algoritmo se emplea tanto en problemas de clasificación como de regresión. Su enfoque principal radica en el concepto de modelado grupal, una técnica que combina múltiples clasificadores para abordar un problema específico y mejorar la eficacia del modelo.

Random Forest se caracteriza por generar múltiples árboles de decisión utilizando diferentes subconjuntos de datos del conjunto de entrenamiento original. Durante el proceso de predicción, cada árbol ofrece su pronóstico, y la decisión final se determina mediante una votación mayoritaria entre los árboles del bosque, en lugar de depender exclusivamente de un único árbol de decisión (Hamsagayathri & Vigneshwaran, 2021). Esta técnica de votación contribuye a mejorar la capacidad predictiva del modelo, proporcionando resultados más robustos y precisos.

4) MLP classifier

El clasificador MLP (Perceptrón Multicapa) emplea la arquitectura de red neuronal feedforward (FFNN) y utiliza el algoritmo de retropropagación para aprender a clasificar instancias a través de un perceptrón multicapa (Permai, Sagala, Zakiyyah, Tanty, & Harefa, 2022). El MLP consta de tres tipos de capas: la capa de entrada, la capa oculta y la capa de salida. La capa de entrada recibe las variables independientes y, en esta investigación, contiene ocho nodos. Las capas ocultas, que pueden consistir en una o más subcapas, están compuestas por nodos que aplican funciones de activación no lineales, como ReLU, para modelar relaciones complejas en los datos. La capa de salida posee nodos que corresponden al número de clases en el problema de clasificación; en un problema multiclase, existirían tantos nodos de salida como clases, empleando una función de activación softmax para generar probabilidades (Permai et al., 2022). Cada nodo en las capas de entrada y oculta tiene un peso asociado, el cual se ajusta minimizando el error mediante el descenso de gradiente y la función de pérdida, como la entropía cruzada en problemas de clasificación. Este ajuste se realiza durante el proceso de entrenamiento de la red, asegurando que el modelo aprenda de los datos de manera efectiva.

F. XGBoost

XGBoost, según Gündoğdu, es una técnica de refuerzo de alto rendimiento que minimiza la función de pérdida mediante diversas optimizaciones. Este método de aumento de gradiente agrega modelos a un conjunto de manera iterativa a través de bucles. El principio básico del refuerzo se centra en las instancias que el modelo no puede predecir correctamente o que son difíciles de predecir (Gündoğdu, 2023). A estos casos se les otorga mayor énfasis, sesgando la distribución de las observaciones para que dichas instancias sean más probables en una muestra. De este modo, Gündoğdu afirma que el siguiente modelo débil se concentrará más en predecir correctamente los casos difíciles. Al combinar todas las reglas de predicción simples en un modelo global, se obtiene un potente predictor: XGBoost (Gündoğdu, 2023).

G. K-Nearest-Neighbour

El algoritmo k-NN es una técnica fundamental y simple de minería de datos, especialmente valiosa en el reconocimiento de patrones (Gündoğdu, 2023).Gündoğdu explica que el concepto básico radica en que, si un punto a evaluar pertenece a la misma categoría que sus vecinos más cercanos en el conjunto de entrenamiento, se puede inferir que dicho punto comparte las mismas características y atributos. Los vecinos más cercanos en el conjunto de entrenamiento tienen un impacto significativo en la precisión de las predicciones. La estimación de un nuevo punto se basa en los valores de estos vecinos cercanos (k-NN). El algoritmo k-NN ha encontrado una amplia aplicación en diversos campos, incluyendo la detección de anomalías, la regresión y el reconocimiento de patrones (Gündoğdu, 2023).

H. Selección de variables

1) Correlación de Pearson

La correlación de Pearson se define como una medida de correlación lineal que se fundamenta en la matriz de covarianza de las variables aleatorias. Aunque esta medida no tiene la capacidad de identificar correlaciones no lineales, su selección se justifica como una referencia inicial para comparaciones (Llamas, Garcia, & Mendez, 2019).

$$\rho_{X,Y} = \frac{C(X,Y)}{\sigma_X \cdot \sigma_Y} \tag{1}$$

donde $C(X,Y)$ es la covarianza y σ_X y σ_Y son desviaciones estándar para X e Y, respectivamente.

2) Chi^2 test

La puntuación de Chi-cuadrado es una medida de cuánto divergen los recuentos esperados y los observados: cuanto mayor es la puntuación, más evidencia hay de una asociación.

Cada característica tiene un valor p enumerado, que es la probabilidad de observar los datos si la hipótesis nula (sin asociación) es verdadera. Los valores p aquí son muy cercanos a cero, lo que sugiere que la hipótesis nula puede rechazarse para todas las características, lo que indica que existen asociaciones significativas entre estas características y la condición que se está probando.

3) Información mutua

El método de información mutua se clasifica como un método discriminatorio en la correlación de características (Ting & Qingsong, 2012). Dentro del contexto de selección de características, la información mutua se emplea para cuantificar el grado de asociación entre la característica *t* y la categoría *C* (Ting & Qingsong, 2012). Su fórmula de cálculo se representa de acuerdo con la expresión matemática proporcionada.

$$\begin{aligned} I(t,C) &= \log \frac{P(t,C)}{P(t)^* P(C)} = \log \frac{P(t|C)^* P(C)}{P(t)^* P(C)} \\ &= \log \frac{P(t|C)}{P(t)} \end{aligned} \tag{2}$$

En esta expresión, $P(t,C)$ representa la probabilidad de co-ocurrencia de t y C, $P(t)$ la probabilidad del texto t ocurriendo en todo el conjunto de entrenamiento, y $P(C)$ la probabilidad de C ocurriendo en el conjunto de entrenamiento (Ting & Qingsong, 2012).

4) ANNOVA Test

ANOVA se utiliza normalmente para datos de respuesta continua donde las variables predictivas son categóricas y supone que la variable de respuesta se distribuye normalmente dentro de cada grupo. Para las características binarias del conjunto de datos, ANOVA no es apropiado porque el resultado binario viola el supuesto de normalidad.

I. Manejo del imabalance de clase

1) Oversampling

A diferencia del undersampling, el oversampling es un método de muestreo que consiste en aumentar la cantidad de datos en las clases minoritarias para compensar o acercarse a la cantidad de datos de la clase mayoritaria (Choirunnisa & Lianto, 2018). El concepto de agregar datos en el oversampling se divide en dos enfoques principalmente: el primero utiliza datos originales, como el método de oversampling aleatorio, y el segundo emplea datos sintéticos, como la técnica de Synthetic Minority Oversampling Technique (SMOTE), Borderline SMOTE, Safe Level SMOTE, Majority Weighted Minority Oversampling Technique (MWMOTE) y Adaptive Semi-Unsupervised Weighted Oversampling (A-SUWO) (Choirunnisa & Lianto, 2018).

Según Choirunnisa and Lianto, un problema común en el oversampling es la aparición de sobreajuste debido a la adición repetida de datos, lo que hace que el límite de decisión sea más estricto. Por lo tanto, el método de oversampling ha evolucionado para evitar la simple duplicación de datos, optando en su lugar por generar nuevos datos que mantengan similitud con los datos existentes.

2) Undersampling

La técnica de undersampling es un proceso de muestreo que se realiza reduciendo o eliminando algunos datos de la clase mayoritaria. Este proceso de eliminación puede llevarse a cabo de manera aleatoria, siendo este el método más simple y comúnmente conocido como undersampling aleatorio (Choirunnisa & Lianto, 2018). Además, el undersampling puede implementarse mediante cálculos estadísticos, lo que se conoce como undersampling informado. En esta técnica, también se utilizan métodos de iteración y técnicas de limpieza de datos para filtrar más eficazmente los datos de la clase mayoritaria (Choirunnisa & Lianto, 2018).

J. SMOTE

En el ámbito del muestreo sintético, se destaca la eficacia de la técnica conocida como sobremuestreo minoritario sintético (SMOTE), la cual ha sido ampliamente reconocida por su efectividad en una variedad de contextos (He & Garcia, 2009). Esta estrategia algorítmica genera datos sintéticos mediante la identificación de similitudes en el espacio de características entre las muestras minoritarias presentes en el conjunto de

datos, ofreciendo así una solución efectiva para abordar el desbalance de clases (He & Garcia, 2009).

K. Random Oversampling and Undersampling

La idea detrás del oversampling aleatorio se deriva intuitivamente de su definición, que implica la creación de un conjunto adicional E a partir de la clase minoritaria. Para lograr esto, se selecciona aleatoriamente un subconjunto E de ejemplos de la clase minoritaria S_{min} y se duplican estos ejemplos, añadiéndolos al conjunto original S. De esta manera, el número total de ejemplos en S_{min} aumenta en $|E|$, lo que ajusta el equilibrio de distribución de clases en S en consecuencia (He & Garcia, 2009). Esta técnica proporciona un método para variar el grado de equilibrio de distribución de clases según sea necesario.

La técnica de Undersampling aleatorio, a diferencia del oversampling, consiste en eliminar datos del conjunto de datos original. Específicamente, se selecciona aleatoriamente un conjunto de ejemplos de las clases mayoritarias en S_{maj} y se eliminan estas muestras de S de manera que $|S| = |S_{\text{min}}| + |S_{\text{maj}}| - |E|$ (He & Garcia, 2009). En este sentido, el Undersampling ofrece un método directo para ajustar el equilibrio del conjunto de datos original S.

1) Mixto

El método híbrido es una técnica de balanceo de datos que combina el uso de oversampling y undersampling (Choirunnisa & Lianto, 2018). Este enfoque reduce la cantidad de datos de las clases mayoritarias a través del concepto de undersampling y aumenta la cantidad de datos de las clases minoritarias mediante el concepto de oversampling (Choirunnisa & Lianto, 2018).

L. Métricas de rendimiento de modelos

Algunas de las medidas de rendimiento incluyen la exactitud (Accuracy), la precisión (P), el Recall (R) y el F1-Score (F1) que son ampliamente usadas en el diagnostico de enfermedades (Ahsan et al., 2022). Para el calculo de estas métricas, se utilizan cuatro tipos de resultados de la clasificación binaria:

- Verdadero Positivo (VP) ocurre cuando el modelo predice correctamente un diagnóstico positivo basado en los síntomas y ese diagnóstico es realmente positivo (Hicks et al., 2022).
- Falso Positivo (FP) ocurre cuando el modelo predice incorrectamente un diagnostico positivo basado en los síntomas, pero en realidad el diagnóstico es negativo (Hicks et al., 2022).
- Verdadero Negativo (VN) ocurre cuando el modelo predice correctamente un diagnóstico negativo basado en los síntomas y ese diagnóstico es realmente negativo (Hicks et al., 2022).
- Falso Negativo (VF) ocurre cuando el modelo predice incorrectamente un diagnóstico negativo basado en los síntomas, pero en realidad el diagnóstico es positivo (Hicks et al., 2022).

1) Matriz de confusión

$$\mathrm{C} = \begin{pmatrix} VP & FP \\ FN & VN \end{pmatrix}$$

2) Exactitud (Accuracy)

La exactitud (E) se define como se define como la proporción de predicciones correctas sobre el total de predicciones realizadas (Ahsan et al., 2022)

La precisión está limitada a [0, 1], donde 1 representa predecir correctamente todas las muestras positivas y negativas, y 0 representa no predecir correctamente ninguna de las muestras positivas o negativas.

$$E = \frac{VP + VN}{VP + FP + VN + FN} \tag{3}$$

Es una de las métricas más utilizadas en la evaluación de modelos médicos (Hicks et al., 2022). Sin embargo, puede traer problemas cuando se trabaja con datos con clases no balanceadas (Hicks et al., 2022). Fácilmente se puede asignar a todas las muestras la clase mayoritaria para lograr una exactitud alta (Hicks et al., 2022).

3) Recall

El *recall* (R) es la proporción de verdaderos positivos sobre el total de instancias que son realmente positivas (Ahsan et al., 2022). También es conocido como Tasa de verdaderos positivos (TPR) (Hicks et al., 2022).

$$R = \frac{VP}{VP + FN} \tag{4}$$

En aplicaciones medicas, los errores tipo II (falsos negativos) podrían ser más graves que los errores tipo I (falsos positivos) (Canbek, Sagiroglu, Temizel, & Baykal, 2017). Por ello, el *recall* es una de las métricas más importantes para los modelos médicos, ya que el objetivo es reducir el número de falso negativos (Hicks et al., 2022).

4) precisión

La precisión (P) es la proporción de verdadero positivos (instancias correctamente clasificadas como positivas) sobre el total de instancias clasificadas como positivas (Ahsan et al., 2022).

$$P = \frac{VP}{VP + FP} \tag{5}$$

5) Especificidad

Similar al *recall*, la especificad es la proporción de verdaderos negativos sobre el total de instancias que son realmente negativas (Hicks et al., 2022).

$$S = \frac{VN}{VN + FN} \tag{6}$$

6) F1-Score

El *F1-Score* es la media armónica entre la precisión y el recall (Ahsan et al., 2022; Hicks et al., 2022).

$$F1 = \frac{P \times R}{P + R} \tag{7}$$

7) ruc-auc

VII. Estado del arte

A. Revisión sistemática de la literatura

Criterios De Búsqueda

1) Machine Learning & Diseases & Symptoms
2) Deep Learning & Diseases & Symptoms
3) Machine Learning & Diabetes
4) Machine Learning & Diagnostic & Diabetes

TABLE XXII: Criterios de búsqueda

Criterios	Fuente				
	IEEE	Zu Scholars	Nature	MDPI	Springer
ML & Diseases	2,826	14	3050	264	62,549
DL & Diseases	1,774	32	8026	151	100,212
ML & Diabetes	2,618	3	3440	420	41,620
ML & Diagnostic & Diabetes	395	14	1540	34	19,507

B. Antecedentes

En un estudio realizado por Tan citado por (Hamsagayathri & Vigneshwaran, 2021) se consideró la idea de unir 2 algoritmos de Machine Learning, el algoritmo genético y SVM, como una estrategia hibrida. En este se utilizó LIBSVM y weka, una herramienta de extracción de datos. Para llevar al cabo dicho estudio se hizo uso de dos sets de datos enfocados en enfermedades de diabetes y enfermedades cardiacas tomados de UC Irvine ML repository. De la evaluación de este modelo hibrido para la diabetes se obtuvo una precisión del 84.07% y para las enfermedades del corazón se obtuvo una precisión del 78.26% (Hamsagayathri & Vigneshwaran, 2021) de lo cual se destacó la correcta clasificación, así como la disminución del sobre ajuste en los datos, sin embargo, el costo computacional que requiere el modelo es alto y por ende el modelo trabaja de manera lenta (Hamsagayathri & Vigneshwaran, 2021).

Por otra parte, Fathima realizo un estudio para la predicción de la enfermedad del dengue haciendo uso del modelo de ML SVM. Para la toma de datos se hizo uso de diversas encuestas de varios hospitales y laboratorios de Chennai y Tiruelveli en la india, así como datos obtenidos del instituto king de medicina preventiva (Hamsagayathri & Vigneshwaran, 2021). De estos datos se utilizaron 28 características de 5006 muestras. La precisión alcanzada de este estudio por el modelo SVM fue de 0.9043 (Hamsagayathri & Vigneshwaran, 2021). En un estudio reciente, se ha demostrado que el uso del big data en el sector de la salud ha permitido realizar predicciones más precisas y descubrir patrones ocultos (Silahtaroğlu & Yılmaztürk, 2019). En este se propone un modelo de prediagnóstico de aprendizaje automático para departamentos de emergencia, el cual utiliza las quejas verbales de los pacientes como datos de entrada. Este modelo alcanza una precisión mínima del 75.5% y se basa en dos enfoques de aprendizaje automático: Red neuronal probabilística y Árbol de decisión de bosque aleatorio. La información utilizada para este estudio fue proporcionada por un hospital privado en Estambul, Turquía. Todos los registros son anónimos y están relacionados con el departamento de emergencias. Los datos contienen las quejas de los pacientes y el diagnóstico final de cada caso, con un total de 5,394 registros después de la limpieza necesaria (Silahtaroğlu & Yılmaztürk, 2019). Este estudio destaca la importancia de utilizar datos generados por los usuarios para mejorar la precisión en la predicción de diagnósticos y proporcionar un mejor soporte en la toma de decisiones en los departamentos de emergencia.

En un artículo de la revista Scientific Reports, se describe un modelo de red neuronal (DL) entrenado con 88 parámetros diferentes, que incluyen 86 características de pruebas de laboratorio, sexo y edad, con el fin de asistir a los médicos en el diagnóstico de enfermedades (Park et al., 2021). Para validar este algoritmo DL, se llevó a cabo una validación cruzada estratificada de cinco veces y se empleó el criterio TOP5 (cinco enfermedades más probables) para evaluar el modelo. Además, se evaluó el rendimiento de cada modelo utilizando puntuaciones F1, dado un desequilibrio en el número de 39 enfermedades. Las puntuaciones F1 y la precisión del modelo DL fueron del 80% y 91%, respectivamente. Para las cinco categorías de enfermedades más comunes, la precisión y el recuerdo fueron del 77% y el 87%, respectivamente (Park et al., 2021).

También en una investigación dirigida a mejorar la precisión de los modelos de predicción del riesgo de enfermedad, especialmente en el caso del infarto cerebral, se exploró la combinación de modelos para obtener resultados más efectivos en comparación con los enfoques convencionales. Para ello, se utilizaron diversos algoritmos de aprendizaje automático que abordaron tanto datos estructurados como no estructurados, obtenidos de registros hospitalarios (Chen, Hao, Hwang, Wang, & Wang, 2017). Los datos estructurados comprendían información de laboratorio y datos básicos del paciente, como la edad, el género y los hábitos de vida. Mientras tanto, los datos no estructurados incluían la narración del paciente sobre su enfermedad, los registros de interrogatorio del médico y los diagnósticos, entre otros. Para los datos estructurados, se aplicaron algoritmos tradicionales como Naive Bayes (NB), K-Nearest Neighbors (KNN) y Decision Trees (DT) para predecir el riesgo de infarto cerebral. Además, se propuso un nuevo algoritmo de predicción de riesgo de enfermedad multimodal basado en una red neuronal convolucional (CNN-MDRP), que integraba datos estructurados y no estructurados. Se destacó que este enfoque proporcionaba una precisión de predicción del 94.8%, superando la velocidad de convergencia de otros modelos unimodales basados en CNN (Chen et al., 2017). A partir de estas investigaciones, se concluyó que la precisión de la predicción del riesgo de enfermedad depende de la diversidad y la calidad de las características de los datos

hospitalarios. Por lo tanto, se propuso la combinación de datos estructurados y no estructurados como una estrategia efectiva para mejorar la precisión en la predicción del riesgo de infarto cerebral y posiblemente otras enfermedades complejas (Chen et al., 2017).

En una investigación previa, se exploró un enfoque de predicción general de enfermedades basado en los síntomas del paciente. Este estudio hizo uso de algoritmos de aprendizaje automático como K-Nearest Neighbor (KNN) y redes neuronales convolucionales (CNN) para llevar a cabo la predicción. Para este propósito, se utilizó un conjunto de datos que comprendía síntomas de enfermedades, junto con información sobre los hábitos de vida y los resultados de los chequeos médicos para mejorar la precisión de la predicción (Dahiwade et al., 2019). Los resultados obtenidos revelaron que la precisión de la predicción general de enfermedades mediante el uso de CNN fue del 84,5%, superando al algoritmo KNN. Además, se observó que el algoritmo KNN requería más tiempo y memoria en comparación con CNN. Después de la predicción de la enfermedad general, el sistema desarrollado en este estudio pudo proporcionar información sobre el riesgo asociado, que variaba según la enfermedad detectada (Dahiwade et al., 2019).

TABLE XXIII: Comparación de la precisión de diferentes modelos por tema y autores

Autores	Tema	Modelo	Precisión
Tan (citado por Gracious et al. (2023))	Predicción de enfermedades del corazón.	SVM y Algoritmo genético	84.07%
	Predicción de enfermedades de diabetes.	SVM	78.26%
Fathima (citado por Hamsagayathri and Vigneshwaran (2021))	Predicción de la enfermedad del dengue	SVM	90.43%
Silahtaroğlu and Yılmaztürk (2019)	Prediagnóstico según quejas de los pacientes	Random forest	77.65%
		Deep Learning Model	91.0%
Chen et al. (2017)	Predicción de enfermedades (infarto cerebral)	convolutional neural network based multimodal disease risk prediction. (CNN-MDRP)	94.8%
Dahiwade et al. (2019)	Predicción de enfermedades basado en los síntomas del paciente	Convolutional neural network (CNN)	84.5%

VIII. Arquitectura

A. Analítica

Se diseñó una arquitectura de análisis de datos basada en cuatro capas. La primera etapa consiste en la ingesta de datos, que implica la recopilación de datos desde su fuente original, en este caso, Kaggle. Seguidamente, en la fase de almacenamiento, los datos se almacenan en BigQuery. La tercera fase, de transformación de datos, abarca el proceso de preparación y entrenamiento de los mismos. Finalmente, la fase de extracción de datos integra el análisis analítico, la visualización de los datos y la selección de modelos, utilizando los resultados obtenidos en las etapas anteriores.

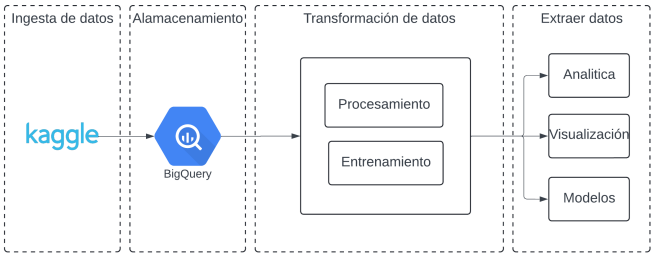

Fig. 44. Arquitectura de Machine Learning

B. Web

Para el desarrollo del prototipo web, se utilizaran varias tecnologias para el desarrollo de una interfaz amigable y una api sencilla de usar que permita acceder a los modelos.

Para el front, se utiliza Streamlit para desarrollar la interfaz de usuario. Este componente del sistema se despliega en Streamlit Community Cloud, aprovechando su capacidad de alojamiento y distribución de aplicaciones web. Por otro lado, en el backend, se emplea FastAPI para construir una API que se implementa en Koyeb, una plataforma de servidor sin servidor que permite desplegar aplicaciones de forma sencilla y escalable.

Para el almacenamiento de modelos, se utiliza un Storage Account de Azure, proporcionando un lugar seguro y confiable para almacenar y acceder a los modelos. Esto asegura que los modelos estén disponibles para su uso en cualquier momento y desde cualquier parte del sistema.

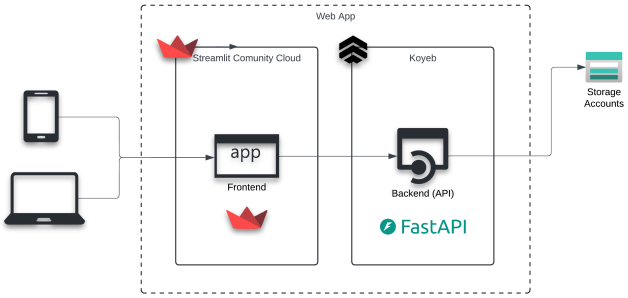

Fig. 45. Arquitectura Web

IX. Resultados

En este estudio, se ha realizado la evaluación de varios modelos de aprendizaje automático con el propósito de predecir la presencia de diabetes en pacientes de manera temprana. Tras realizar el análisis correspondiente a los datos obtenidos, se determinó a través de técnicas como la Correlación de Pearson, Chi2, ANOVA e información mutua, que las características más importantes a tener en cuenta para el entrenamiento del modelo fueron las siguientes:

TABLE XXIV: Características seleccionadas.

Característica	**Descripción**
HighBP	Presión arterial alta
HighChol	Colesterol alto
BMI	Índice de masa corporal
Smoker	Fumador
Stroke	Accidente cerebrovascular
HeartDiseaseorAttack	Enfermedad cardíaca o ataque cardíaco
PhysActivity	Actividad física
HvyAlcoholConsump	Consumo de alcohol pesado
GenHlth	Salud general
MentHlth	Salud mental
PhysHlth	Salud física
DiffWalk	Dificultad para caminar
Age	Edad
Education	Educación
Income	Ingresos

Todas estas características muestran una correlación significativa con la variable objetivo (diabetes_binary), superando el umbral de 0.05 en la correlación de Pearson y un puntaje superior a 250,000 en la prueba Chi-cuadrado. Además, el análisis mediante ANOVA identificó diferencias significativas en las medias de características como GenHlth, HighBP, BMI y Age entre los grupos de diabéticos y no diabéticos, indicando su relevancia en la predicción de la diabetes. Por otro lado, la información mutua destacó la importancia de variables como HighBP, CholCheck y AnyHealthcare, al mostrar una alta reducción de la incertidumbre respecto a la variable objetivo. Por lo tanto, estas variables se consideran fundamentales para el entrenamiento de los modelos, ya que proporcionan una base sólida para realizar predicciones precisas y mejorar la capacidad del modelo en la detección temprana de diabetes tipo 2.

a) Class imbalance

Para abordar el desbalance entre las clases en el conjunto de datos, se emplearon diversas estrategias. Se determinó que una de las opciones más efectivas fue la combinación de técnicas de oversampling y undersampling, utilizando SMOTE (Synthetic Minority Over-sampling Technique) y NierMiss, respectivamente. Estas metodologías equilibraron la distribución de clases, lo que mejoró significativamente el rendimiento de los modelos en la detección de casos positivos. Específicamente, la estrategia de oversampling mediante SMOTE generó instancias sintéticas de la clase minoritaria (diabéticos), contribuyendo así a igualar la distribución de clases y a mejorar la capacidad predictiva de los modelos

b) Modelos seleccionados

Después de realizar pruebas con todos los modelos seleccionados para este estudio, se determinó que el modelo con los mejores resultados fue el Random Forest. Al evaluar dicho modelo utilizando la técnica de validación cruzada y división de datos en conjuntos de entrenamiento y evaluación, se obtuvieron valores de 90.62% 91.07% y 90.62% en sus métricas de*accuracy* , *precision* y *recall* respectivamente.

Estos resultados señalan un buen equilibrio entre precisión y recall en la predicción de la diabetes, lo que sugiere la efectividad de los modelos evaluados en esta tarea específica. Además, en las gráficas 32 y 33 se puede apreciar la eficiencia del modelo en el manejo de verdaderos y falsos positivos, así como la reducción de falsos negativos, cuya importancia ya hemos resaltado en diversas ocasiones. Sin embargo, el tiempo que tomo en entrenarse fue considerablemente mayor al tiempo de entrenamiento de otro modelo con resultados similares; Decision Tree.

El modelo de Decision Tree mostró resultados comparables al del modelo de Random Forest. Los valores obtenidos fueron 87.05% 88.21% y 87.05% en sus métricas de *accuracy*, *precision* y *recall* respectivamente. Adicionalmente, la capacidad del modelo para identificar verdaderos y falsos positivos, así como para reducir los falsos negativos, se hace evidente en las gráficas 24 y 25, destacando su relevancia en diversas situaciones.

Como se puede obsevar, estos valores son muy similares entre ambos modelos, con diferencias mínimas de 3.57%, 2.86% y 3.57% respectivamente. Los gráficos de las curvas ROC indican que ambos modelos presentan un comportamiento ideal. Al analizar la cantidad de falsos negativos, que es un factor crítico en nuestra selección final, se observa que ambos modelos presentan cifras comparables, con 656 y 678 falsos negativos para los modelos de Random Forest y Decision Tree, respectivamente. Además, el tiempo de entrenamiento del modelo de árbol de decisión fue significativamente menor en comparación con el modelo de Random Forest, y el tamaño del modelo también se redujo considerablemente. La diferencia en términos de almacenamiento fue de 808.7 MB, teniendo el modelo Random Forest un peso de 816.9 MB y el modelo Decision Tree 8.2 MB. Debido a estas ventajas, el modelo de árboles de decisión fue seleccionado para su implementación en la aplicación web.

Por otra parte, otro de los modelos seleccionados para el despliegue en la app web fue XGB Classifier. Esto debido a los valores alcanzados en sus métricas de *accuracy*, *precision* y *recall*, los cuales fueron 80.18% 80.54% y 80.18% respectivamente. En las matrices de confusión y las curvas ROC, gráficas 40 y 41, se destaca el eficiente manejo de los verdaderos positivos y falsos positivos, así como la gestión de los falsos negativos. Aunque estos últimos son superiores a los observados en los modelos de Decision Tree y Random Forest, siguen siendo notoriamente bajos en comparación con los obtenidos por otros modelos.

El tercer y ultimo modelo seleccionado fue el modelo K-Nearest Neighbors (KNN), con unos valores alcanzados en sus métricas de *accuracy*, *precision* y *recall*, los cuales fueron 79.73% 81.21% y 79.72% respectivamente. Una vez más, en las gráficas de matrix de confusión y curva de ROC (36 y 37) muestran un adecuado manejo de los verdaderos positivos y falsos positivos, aunque un poco inferiores en comparación con los otros modelos seleccionados.

Una vez que los modelos estuvieron listos, se llevó a cabo un análisis de importancia de características para identificar las variables que más influían en el comportamiento de cada modelo. Este análisis tuvo como objetivo determinar el peso relativo de cada variable en los modelos, proporcionando una comprensión más profunda de cómo cada característica afectaba las predicciones. De esta manera, se pudo obtener una idea previa del funcionamiento interno y la lógica de los modelos, lo cual es esencial tanto para la interpretación de los resultados como para la mejora y optimización de los modelos en futuras iteraciones. Los resultados de dicho analsis arrojaron lo siguiente:

c) Random Forest

La gráfica 46 es una representación de la importancia de las características (*feature importance*) en el modelo de Random Forest. De la gráfica podemos sacar las siguientes conclusiones respecto a las variables procesadas por este modelo:

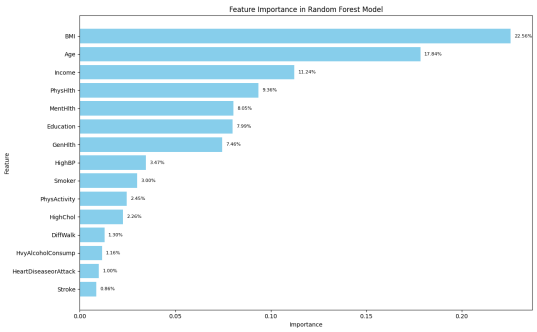

Fig. 46. Importancia de caracteristicas en Random Forest

- Variables Más Importantes:
  - BMI (Índice de Masa Corporal): Es la característica más importante, con un valor de importancia del 22.56%. Esto sugiere que el *BMI* tiene un impacto significativo en el modelo y es una variable clave para realizar las predicciones.
  - Age (Edad): La segunda característica más importante con un 17.84% de importancia. Esto indica que la edad también juega un papel crucial en el desempeño del modelo.
- Variables Moderadamente Importantes:
  - Income (Ingreso): Con un 11.24% de importancia, el ingreso es otra característica relevante.
  - PhysHlth (Salud Física) y MentHlth (Salud Mental): Estas características también son importantes, con valores de 9.36% y 8.05%, respectivamente.
  - Education (Educación): Con un 7.99% de importancia, la educación influye de manera significativa en las predicciones del modelo.
  - GenHlth (Salud General): Con un 7.46% de importancia, la salud general también es relevante.
- Variables Menos Importantes:
  - HighBP (Presión Arterial Alta): Tiene una importancia del 3.47%.
  - Smoker (Fumador): Con un 3.00% de importancia.

  - PhysActivity (Actividad Física): Con un 2.45% de importancia.
  - HighChol (Colesterol Alto): Con un 2.26% de importancia.
- Variables de Importancia Baja:
  - DiffWalk (Dificultad para Caminar), HvyAlcoholConsump (Consumo de Alcohol en Exceso), HeartDiseaseorAttack (Enfermedad del Corazón o Ataque), Stroke (Accidente Cerebrovascular): Estas características tienen una importancia menor (menos del 1.5%), lo que indica que tienen un impacto reducido en el modelo.

d) XGB Classifier

La gráfica 47 muestra la importancia de las características (*feature importance*) en un modelo de XGB Classifier. A continuación, se exponen las conclusiones que se pueden derivar de esta gráfica:

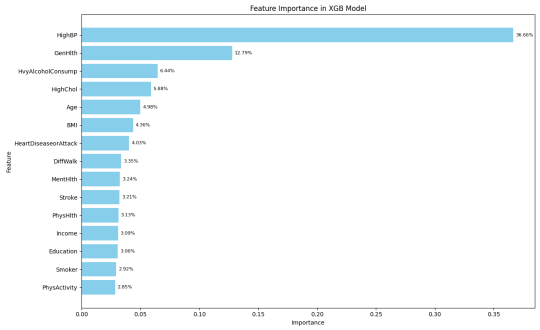

Fig. 47. Importancia de caracteristicas en XGB Classifier

- Variables Más Importantes:
  - HighBP (Presión Arterial Alta): Es la característica más importante con un valor de importancia del 36.66%. Esto sugiere que la presión arterial alta tiene un impacto significativo en el modelo y es una variable crucial para realizar las predicciones.
  - GenHlth (Salud General): La segunda característica más importante con un 12.79% de importancia. Esto indica que la salud general también juega un papel importante en el rendimiento del modelo.
- Variables Moderadamente Importantes:
  - HvyAlcoholConsump (Consumo de Alcohol en Exceso): Con un 6.44% de importancia, el consumo excesivo de alcohol es una característica relevante.
  - HighChol (Colesterol Alto): Con un 5.88% de importancia.
  - Age (Edad): Con un 4.98% de importancia, la edad es también una variable significativa.
  - BMI (Índice de Masa Corporal): Con un 4.36% de importancia.
  - HeartDiseaseorAttack (Enfermedad del Corazón o Ataque): Con un 4.03% de importancia.
- Variables Menos Importantes:
  - DiffWalk (Dificultad para Caminar): Tiene una importancia del 3.35%.
  - MentHlth (Salud Mental): Con un 3.24% de importancia.
  - Stroke (Accidente Cerebrovascular): Con un 3.21% de importancia.
  - PhysHlth (Salud Física): Con un 3.13% de importancia.
  - Income (Ingreso): Con un 3.09% de importancia.
  - Education (Educación): Con un 3.06% de importancia.
  - Smoker (Fumador): Con un 2.92% de importancia.
  - PhysActivity (Actividad Física): Con un 2.85% de importancia.

Como implementación adicional, se creó un modelo de ensamblado para combinar los resultados de predicción de los tres mejores modelos seleccionados. El objetivo era lograr una predicción más precisa y realista aprovechando las fortalezas combinadas de estos modelos. Los valores obtenidos en las pruebas para las métricas usadas regularmente a lo largo del estudio, *accuracy*, *precision* y *recall*, fueron del 88.06%, 88.36% y 88.06% respectivamente. Estos valores resultan ser muy similares a los valores del modelo DT y superiores a los valores de los modelos KNN y XGB, generando así una predicción más eficiente. Además, el ensamblado mantuvo los números de falsos negativos en niveles mínimos, consolidando su eficacia en la detección precisa de los casos relevantes.

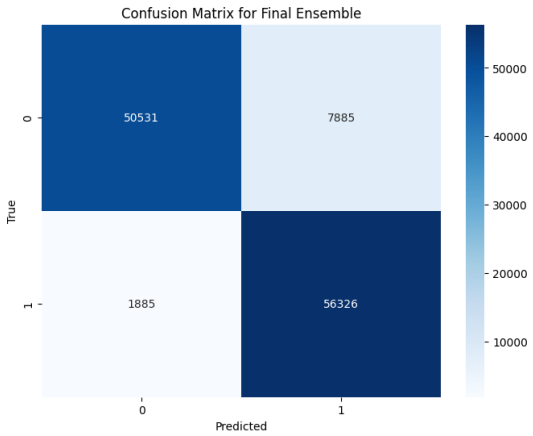

Fig. 48. Matrix de confusión Final Ensemble

Se empleó el método de votación dura (`voting='hard'`), en el cual cada clasificador base emite un voto, y la clase final predicha corresponde a aquella que recibe la mayoría de los votos. Este método se utilizó para combinar las fortalezas de múltiples modelos y mejorar la precisión general del sistema de diagnóstico.

Para asignar diferentes niveles de importancia a cada clasificador base, se utilizaron pesos específicos en la votación:

- Decision Tree: peso de 2
- KNN: peso de 1
- XGBoost: peso de 1

Esto implica que el clasificador de árbol de decisión tiene el doble de influencia en la predicción final comparado con los otros dos modelos. La asignación de estos pesos se basó en un análisis previo del rendimiento individual de cada modelo, en el cual se observó que el árbol de decisión presentaba un rendimiento superior en este contexto específico. Este enfoque permite mejorar la precisión del modelo ensamblado al darle mayor relevancia al clasificador más efectivo.

X. Prototipo

A. Descripción general

El prototipo desarrollado para la detección temprana de diabetes tipo 2 se basa en una aplicación web interactiva que integra múltiples tecnologías para ofrecer una herramienta intuitiva y precisa. La interfaz de usuario está diseñada con Streamlit, permitiendo a los usuarios ingresar datos personales y de salud a través de un formulario sencillo y claro. Este formulario incluye variables clave como altura, peso, sexo, edad, nivel educativo, ingresos, consumo de alcohol, presión arterial alta, colesterol alto y hábitos de tabaquismo.

Una vez ingresados los datos, la aplicación utiliza modelos de Machine Learning previamente entrenados para evaluar el riesgo de diabetes. Entre los modelos implementados se destacan Random Forest, Decision Tree y XGBoost, optimizados mediante técnicas como SMOTE y RandomUnderSampler para manejar el desequilibrio de clases. La arquitectura backend, desarrollada con FastAPI, se encarga de procesar las entradas del usuario y generar predicciones en tiempo real, utilizando un almacenamiento seguro en Azure para los modelos.

Los resultados de la evaluación se presentan de manera clara y accesible. Si el modelo detecta una alta probabilidad de diabetes, se muestra una alerta junto con información adicional relevante, como el índice de masa corporal (BMI) del usuario. Por otro lado, si la probabilidad es baja, se informa al usuario con un mensaje tranquilizador y detalles sobre su estado físico.

Este prototipo no solo facilita la detección temprana de diabetes tipo 2, sino que también demuestra la viabilidad y efectividad de integrar soluciones de Machine Learning en aplicaciones web para mejorar la toma de decisiones en el ámbito de la salud. La implementación se complementa con un despliegue en plataformas serverless como Koyeb, garantizando escalabilidad y facilidad de mantenimiento.

B. Capturas

Fig. 49. Instrumento de recolección de datos

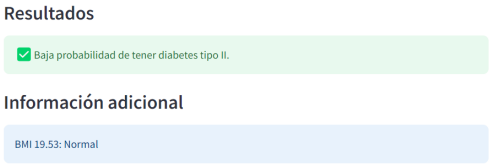

Fig. 50. Mensaje de alerta para el usuario

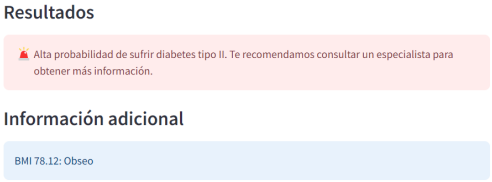

Fig. 51. Mensaje tranquilizador para el usuario

C. Enlace y Repositorios

El prototipo se encuentra disponible en el siguiente link https://diabetes-prediction-uninorte.streamlit.app. Además, el código fuente del backend, frontend y cuarderno de jupyter (donde se entranaron los modeos) se encuentran disponibles en los siguientes repositorios de github:

- https://github.com/Sonya-c/diabetes-prediction-frontend
- https://github.com/Jeffrey-Felix095/diabetes-prediction-backend/
- https://github.com/Jeffrey-Felix095/Jeffrey-Felix095-diabetes-prediction-analytic

D. Evaluación exterior del prototipo

Para validar que el prototipo y el proyecto en su totalidad cumplen con los objetivos planteados, fue necesario llevar a cabo una evaluación entre miembros de distintos proyectos. En el marco de la asignatura Proyecto Final, se realizó

una validación del prototipo por parte de grupos de pares, empleando el estándar ISO 15504 XXV y el Instrumento de Evaluación del Prototipo XXVI. En esta evaluación, una puntuación de 5 representa ”totalmente de acuerdo”, 1 ”totalmente en desacuerdo” y NA ”no aplica”. Los resultados obtenidos, muestran una aceptación favorable del prototipo, resaltando su facilidad de uso y precisión en la detección de la diabetes tipo 2.

TABLE XXV: Escala Likert / ISO 15504

	Calificación	**Descripción**
Totalmente en desacuerdo	1	Muy bajo
En desacuerdo	1	Bajo
Ni acuerdo, ni desacuerdo	3	Medio
De acuerdo	4	Alto
Totalmente de acuerdo	5	Muy alto

TABLE XXVI: Modelo de evaluación basado en el estandar ISO 9126 ISo 15504 + Escala de Linkert

Característica	**Descripción**	**Calificación**
Understandability	¿Fácil de comprender?	5
Documentation	¿Documentación de usuario completa, apropieda y bien estructurada?	4
Buildability	¿Fácil de construit en un sistema compartible? (Close-Open)	NA
Installability	¿Fácil de instalar en un sistema compartible?	NA
Learnability	¿Fácil de aprender a usasr sus funciones?	5
Identity	¿La identidad del proyecto / software es clara y única?	4
Copyright Licecing	¿Adopción de la licencia apropiada?	NA
Licencing	Adopción de la licencia apropiada	4
Governance	¿Fácil de entender cómo se ejecuta el proyecto y cómo se gestiona el desarrollo del software?	5
Community	¿Evidencia de comunidad actual/futura?	5
Accessibility	¿Evidencia de capacidad de descarga actual / Futura?	4
Testability	¿Fácil de probar la correción de funciones caja negra?	4
Portability	¿Utilizable en múltiples plataformas?	NA
Supportability	¿Evidencia de soporte para desarrolladores actuales / futuros?	4
Analysability	¿Fácil de entender a nivel fuente?	NA
Changeability	¿Fácil de modificar y aportar cambios a los desarrolladores?	5
Evolvability	’Evidencia de desarrollo actual / futuro?	4
Interoperability	¿Interoperabilidad con otro software requerido / relaciona?	NA

XI. Conclusiones

El proyecto de desarrollo de un modelo de diagnóstico temprano de diabetes tipo 2 mediante técnicas de Machine Learning ha logrado resultados significativos, demostrando la viabilidad y efectividad de diversas técnicas de aprendizaje automático aplicadas a este ámbito de la salud. A lo largo del proyecto, se evaluaron múltiples modelos, incluyendo Naive Bayes, Decision Tree, Random Forest, MLP Classifier, K-Nearest Neighbors (KNN) y XGBoost (XGB), cada uno optimizado y ajustado para maximizar su rendimiento en la detección de diabetes.

Inicialmente, se llevó a cabo una exhaustiva revisión de la literatura con el objetivo de recopilar diversas propuestas previas y establecer un punto de partida claro. Esta búsqueda bibliográfica permitió identificar una variedad de metodologías y resultados asociados con la predicción de la diabetes tipo 2. De este conjunto, se seleccionaron los modelos más frecuentemente empleados para dicho propósito: Random Forest,

Decision Tree, XGBoost Classifier, Gaussian Naive-Bayes, MLP classifier y K-Nearest Neighbor.

Adoptando la metodología CRISP-ML, se procedió a desglosar cada etapa necesaria para alcanzar los objetivos establecidos. Esto incluyó un análisis exploratorio de los datos, donde se identificaron y abordaron posibles problemas como valores atípicos, datos faltantes y duplicados. Además, se llevó a cabo un balanceo de clases para mitigar sesgos en los resultados del modelo, concluyendo que la técnica de SMOTE con NierMiss ofrecía los mejores resultados.

Esto resultó en mejoras notables en la precisión y recall de los modelos, destacando la importancia de estas técnicas en la preparación de datos para aplicaciones médicas. La validación cruzada y la optimización de hiperparámetros a través de GridSearchCV fueron cruciales para mejorar el rendimiento de los modelos.

El modelo Random Forest se destacó por su rendimiento superior, alcanzando métricas de accuracy, precision y recall de 90.62%, 91.07% y 90.62%, respectivamente. Este modelo mostró una gestión eficiente de los verdaderos positivos y falsos negativos, lo que es crucial en la detección temprana de enfermedades. Adicionalmente, se desarrolló una API diseñada para conectar las predicciones de dichos modelos con una página web específicamente diseñada para exponer la información a los usuarios y recabar datos de estos. A pesar del destacado rendimiento del modelo Random Forest, no se consideró su inclusión en el despliegue de la API debido a las limitaciones en herramientas y capacidades para implementar y mantener un modelo tan pesado. Como alternativa, se implemento en el despliegue el ensamble de los 3 mejores modelos restantes, cuyos resultados fueron igualmente buenos, alcanzando valores de 88.06%, 88.36% y 88.06% para accuracy, precision y recall respectivamente.

El sitio web presenta tanto el informe completo del proyecto como una página para que los usuarios ingresen los datos necesarios para que el modelo realice predicciones. La API se diseñó de manera que solo recibe los datos necesarios para realizar las predicciones correspondientes, simplificando así el proceso de conexión con la página web.

El proyecto cumplió con sus objetivos de desarrollar y evaluar modelos de Machine Learning para la detección temprana de diabetes tipo 2. Los resultados obtenidos no solo demuestran la efectividad de estas técnicas, sino que también proporcionan una base sólida para futuras investigaciones y mejoras en el diagnóstico médico asistido por tecnología. La implementación de técnicas avanzadas de preprocesamiento de datos y el uso de arquitecturas de modelos más complejas, como redes neuronales profundas, podrían mejorar aún más la precisión y confiabilidad de los diagnósticos.

En conclusión, el proyecto ha alcanzado todos los objetivos establecidos, demostrando la viabilidad de utilizar modelos de Machine Learning para la detección temprana de la diabetes tipo 2. No obstante, es crucial resaltar que aún hay espacio para continuar mejorando. En este sentido, se planea llevar a cabo encuestas dirigidas al personal de salud con el fin de obtener retroalimentación sobre el prototipo, lo que permitirá identificar posibles mejoras para su implementación futura. Además, la aplicación práctica de estos modelos en una plataforma web, utilizando tecnologías como Streamlit y FastAPI, subraya la viabilidad de implementar soluciones de Machine Learning en entornos clínicos reales, ofreciendo herramientas precisas y accesibles para la detección temprana de enfermedades.

References

Ahmad, G. N., Fatima, H., Ullah, S., Salah Saidi, A., & Imdadullah. (2022). Efficient medical diagnosis of human heart diseases using machine learning techniques with and without gridsearchcv. *IEEE Access*, *10*, 80151-80173. doi: 10.1109/ACCESS.2022.3165792

Ahsan, M. M., Luna, S. A., & Siddique, Z. (2022, March). Machine-learning-based disease diagnosis: A comprehensive review. *Healthcare*, *10*(3), 541. doi: 10.3390/healthcare10030541

Alowais, S. A., Alghamdi, S. S., Alsuhebany, N., Alqahtani, T., Alshaya, A. I., Almohareb, S. N., … Albekairy, A. M. (2023, September). Revolutionizing healthcare: the role of artificial intelligence in clinical practice. *BMC Medical Education*, *23*(1). doi: 10.1186/s12909-023-04698-z

Arun Bhavsar, K., Abugabah, A., Singla, J., Ali AlZubi, A., Kashif Bashir, A., & Nikita. (2021). A comprehensive review on medical diagnosis using machine learning. *Computers, Materials amp; Continua*, *67*(2), 1997–2014. doi: 10.32604/cmc.2021.014943

Azar, G., Gloster, C., El-Bathy, N., Yu, S., Neela, R. H., & Alothman, I. (2015, May). Intelligent data mining and machine learning for mental health diagnosis using genetic algorithm.
doi: 10.1109/eit.2015.7293425

Caballé-Cervigón, N., Castillo-Sequera, J. L., Gómez-Pulido, J. A., Gómez-Pulido, J. M., & Polo-Luque, M. L. (2020, July). Machine learning applied to diagnosis of human diseases: A systematic review. *Applied Sciences*, *10*(15), 5135. doi: 10.3390/app10155135

Canbek, G., Sagiroglu, S., Temizel, T. T., & Baykal, N. (2017, October). Binary classification performance measures/metrics: A comprehensive visualized roadmap to gain new insights. In *2017 international conference on computer science and engineering (ubmk).* IEEE. doi: 10.1109/ubmk.2017.8093539

Chen, M., Hao, Y., Hwang, K., Wang, L., & Wang, L. (2017). Disease prediction by machine learning over big data from healthcare communities. *IEEE Access*, *5*, 8869–8879. doi: 10.1109/access.2017.2694446

Choirunnisa, S., & Lianto, J. (2018). Hybrid method of undersampling and oversampling for handling imbalanced data. In *2018 international seminar on research of information technology and intelligent systems (isriti)* (p. 276-280). doi: 10.1109/ISRITI.2018.8864335

Dahiwade, D., Patle, G., & Meshram, E. (2019, March). Designing disease prediction model using machine learning approach. In *2019 3rd international conference on*

computing methodologies and communication (iccmc). IEEE. doi: 10.1109/iccmc.2019.8819782

ElSayed, N. A., Aleppo, G., Aroda, V. R., Bannuru, R. R., Brown, F. M., Bruemmer, D., … Association, A. D. (2022, 12). 2. Classification and Diagnosis of Diabetes: Standards of Care in Diabetes—2023. *Diabetes Care*, *46*(Supplement$_1$), $S19 - S40$.

Gracious, L. A., Jasmine, R. M., Pooja, E., Anish, T., Johncy, G., & Subramanian, R. S. (2023, October). Machine learning and deep learning transforming healthcare: An extensive exploration of applications, algorithms, and prospects. In *2023 4th ieee global conference for advancement in technology (gcat)*. IEEE. doi: 10.1109/gcat59970.2023.10353476

Gündoğdu, S. (2023). Efficient prediction of early-stage diabetes using xgboost classifier with random forest feature selection technique. *Multimedia Tools and Applications*, *82*(22), 34163–34181.

Hamsagayathri, P., & Vigneshwaran, S. (2021, February). Symptoms based disease prediction using machine learning techniques. In *2021 third international conference on intelligent communication technologies and virtual mobile networks (icicv)*. IEEE. doi: 10.1109/icicv50876.2021.9388603

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, *21*(9), 1263-1284. doi: 10.1109/TKDE.2008.239

Hicks, S. A., Strümke, I., Thambawita, V., Hammou, M., Riegler, M. A., Halvorsen, P., & Parasa, S. (2022, April). On evaluation metrics for medical applications of artificial intelligence. *Scientific Reports*, *12*(1). doi: 10.1038/s41598-022-09954-8

Keniya, R., Khakharia, A., Shah, V., Gada, V., Manjalkar, R., Thaker, T., … Mehendale, N. (2020). Disease prediction from various symptoms using machine learning. *SSRN Electronic Journal*. Retrieved from http://dx.doi.org/10.2139/ssrn.3661426 doi: 10.2139/ssrn.3661426

Llamas, E., Garcia, I., & Mendez, A. (2019). Detecting dependency between discrete random variables and application. In *2019 ieee international conference on big data (big data)* (p. 4532-4536). doi: 10.1109/BigData47090.2019.9006099

Mangal, A., & Jain, V. (2022). Performance analysis of machine learning models for prediction of diabetes. In *2022 2nd international conference on innovative sustainable computational technologies (cisct)* (p. 1-4). doi: 10.1109/CISCT55310.2022.10046630

Nazreen, J. (2023, 11). *Diabetes Health Indicators Dataset*. Retrieved from https://www.kaggle.com/datasets/julnazz/diabetes-health-indicators-dataset

OMS. (2024). *Diabetes*. World Health Organization. Retrieved from https://www.who.int/es/news-room/fact-sheets/detail/diabetes

Park, D. J., Park, M. W., Lee, H., Kim, Y.-J., Kim, Y., & Park, Y. H. (2021, April). Development of machine learning model for diagnostic disease prediction based on laboratory tests. *Scientific Reports*, *11*(1). doi: 10.1038/s41598-021-87171-5

Permai, S. D., Sagala, N. T. M., Zakiyyah, A. Y., Tanty, H., & Harefa, J. (2022). Multiclass classification for air quality in jakarta using support vector machine and multilayer perceptron classifier. In *2022 3rd international conference on artificial intelligence and data sciences (aidas)* (p. 198-202). doi: 10.1109/AiDAS56890.2022.9918697

Silahtaroğlu, G., & Yılmaztürk, N. (2019, June). Data analysis in health and big data: A machine learning medical diagnosis model based on patients' complaints. *Communications in Statistics - Theory and Methods*, *50*(7), 1547–1556. doi: 10.1080/03610926.2019.1622728

Studer, S., Bui, T. B., Drescher, C., Hanuschkin, A., Winkler, L., Peters, S., & Müller, K.-R. (2021, April). Towards crisp-ml(q): A machine learning process model with quality assurance methodology. *Machine Learning and Knowledge Extraction*, *3*(2), 392–413. doi: 10.3390/make3020020

Ting, L., & Qingsong, Y. (2012). Spam feature selection based on the improved mutual information algorithm. In *2012 fourth international conference on multimedia information networking and security* (p. 67-70). doi: 10.1109/MINES.2012.203

XII. Anexos

A. Tabla de revisión sistemática de la literatura

TABLE XXVII: Revisión sistemática de la literatura

No.	Título	Palabras Clave	Fuente
1	Machine Learning and Deep Learning Transforming Healthcare: An Extensive Exploration of Applications, Algorithms, and Prospects	Deep learning, Surveys, Drugs, Decision support systems, Ethics, Data privacy, Precision medicine	IEEE
2	Symptoms Based Disease Prediction Using Machine Learning Techniques	Heart, Solid modeling, Decision making, Machine learning, Diabetes, Medical diagnostic imaging, Diseases	IEEE

3	A comprehensive review on medical diagnosis using machine learning	Diagnostic system, Healthcare applications, Machine learning, medical diagnosis	Zu Scholars
4	Development of machine learning model for diagnostic disease prediction based on laboratory tests	Experimental models of disease, Information theory and computation, Machine learning	Nature
5	Designing Disease Prediction Model Using Machine Learning Approach	Diseases, Prediction algorithms, Machine learning algorithms, Machine learning, Classification algorithms, Biomedical monitoring, Cloud computing	IEEE
6	Machine Learning Applied to Diagnosis of Human Diseases: A Systematic Review	human disease, machine learning, data mining, artificial intelligence, big data	MDPI
7	Revolutionizing healthcare: the role of artificial intelligence in clinical practice	AI, Healthcare, Patient care, Quality of life, Clinicians, Decision-making, Personalized treatment plans	Springer
8	Data analysis in health and big data: A machine learning medical diagnosis model based on patients' complaints	Machine learning, text mining, patients' complaints, diagnosis	Taylor & Francis
9	Disease Prediction by Machine Learning Over Big Data From Healthcare Communities	Diseases, Hospitals, Prediction algorithms, Machine learning algorithms, Big Data, Data models, Big data analytics, machine learning, healthcare	IEEE
10	Machine Learning-based Diabetes Prediction: A Cross-Country Perspective	Machine learning algorithms, Predictive models, Prediction algorithms, Boosting, Data models, Diabetes, Decision trees, Diabetes, Machine Learning Models, Missing values, Outliers	IEEE
11	Performance analysis of machine learning models for prediction of diabetes	Machine learning algorithms, Computational modeling, Predictive models, Prediction algorithms, Medical tests, Diabetes, Predictive Analysis, Machine Learning, Diabetes Prediction, Random Forest, Logistic Regression	IEEE