Implementación de un método diagnóstico para la enfermedad de Chagas utilizando espectrometría de masas (MALDI-TOF) y aprendizaje automatizado *Machine Learning*

Yenny Fernanda Velandia Hernández

Trabajo de Grado para Optar al Título de Química

Director

Enrique Mejía Ospino

Doctor en Ciencias Químicas

Codirectora

Yuly Andrea Prada Vargas

Doctora en Química

Universidad Industrial de Santander

Facultad de Ciencias

Escuela de Química

Química

Bucaramanga

2024

Dedicatoria

A mis padres Fernando y Yeny.

A Copo, quién ya no está en este mundo.

A David quien me acompañó y animó en este camino

A mis amigos y primas quienes siempre me apoyaron.

Sin ustedes todo esto no hubiera sido posible.

Agradecimientos

En primer lugar, debo agradecer a mi familia por su apoyo y aliento a lo largo de los años. Especialmente a mis padres porque gracias a ellos he llegado a este punto y sin ellos no lo habría logrado.

A la Universidad Industrial de Santander, la Facultad de Ciencias y la Escuela de Química por la inestimable formación académica.

Al Dr. Enrique Mejía Ospino y a la Dra. Yuly Andrea Medina Prada, mi más sincero agradecimiento por recibirme en el Laboratorio de Espectroscopia Atómica y Molecular y por su paciencia, conocimiento, enseñanza, dedicación, carisma y amabilidad como tutores durante todo el desarrollo de la investigación.

Al Grupo de Espectroscopia Atómica y Molecular LEAM y al Laboratorio de Espectrometría de Masas, por la oportunidad que me dieron de aprender y crecer profesionalmente en un entorno enriquecedor; igualmente, por su ayuda. Su apoyo ha sido fundamental para el éxito de mi trabajo.

Al Grupo de Investigación en Bioquímica y Microbiología GIBIM y sus integrantes, por su ayuda, consejos y guías en el entendimiento de la digestión de proteínas además de haberme permitido utilizar sus instalaciones durante mi investigación.

Al grupo de Investigación en Inmunología y Epidemiología Molecular GIEM, por la oportunidad de trabajar con los sueros sanguíneos de pacientes con la enfermedad de Chagas. Estas muestras han sido elementales para el desarrollo de la investigación y entendimiento de esta enfermedad.

Al Centro de Investigaciones en Enfermedades Tropicales CINTROP por permitirme el uso de sus instalaciones en momentos críticos de la investigación.

Tabla de Contenido

Pág.

Lista de Tablas

Pág.

Lista de Figuras

Pág.

Lista de Apéndices

pág.

Glosario

EC: Enfermedad de Chagas

FASP: Preparación de muestras asistida por filtro

FN: Falso Negativo

FP: Falso Positivo

GBDT: Árboles de decisión potenciados por gradiente

IDE: Entorno de desarrollo integrado

IgG: Inmunoglobulina G

IgM: Inmunoglobulina M

KNN: K Vecinos Más Cercanos lograron

LinearSVC: Vectores de Soporte Lineal Escalable para Clasificación

MALDI TOF MS: Espectrometría de masas de ionización-desorción asistida por matriz con tiempo de vuelo

ML: Aprendizaje automatizado

MLP: Perceptrones Multicapa

NN: Redes Neuronales

PCA: análisis de componentes principales

RF: Bosque aleatorio

SDG: Descenso de gradiente estocástico

SVM: Máquinas de vectores de soporte

T.Cruzi*: Trypanozoma Cruzi*

TOF: Tiempo de vuelo

VN: Verdadero Negativo

VP: Verdadero Positivo

Resumen

Título: Implementación de un método diagnóstico para la enfermedad de Chagas utilizando espectrometría de masas (MALDI-TOF) y aprendizaje automatizado *Machine Learning*[*]

Autor: Yenny Fernanda Velandia Hernández[**]

Palabras Clave: Enfermedad de Chagas, diagnóstico, Aprendizaje automatizado

Descripción: La enfermedad de Chagas es causada por el parásito *Trypanosoma cruzi*, y representa una amenaza persistente y potencialmente letal. Las pruebas confirmatorias son clave para prevenir la progresión de la enfermedad a la fase crítica y evitar consecuencias como la insuficiencia cardíaca y muerte. En el diagnóstico, las técnicas iniciales basadas en tinciones sanguíneas y las pruebas serológicas han enfrentado desafíos con falsos negativos y positivos en distintas fases de la enfermedad.
En este estudio, se empleó la Espectrometría de masas MALDI-TOF junto con herramientas computacionales basadas en el aprendizaje automatizado supervisado para formular un modelo predictivo de la enfermedad de Chagas. Este enfoque novedoso en el diagnóstico siguió un proceso secuencial que inició con la selección de muestras, determinación de proteínas totales presentes en suero, preparación de muestras por digestión enzimática de proteínas y finalmente la obtención de los perfiles proteómicos para la formulación de modelos predictivos basados en *Machine learning*. Como resultado del estudio se obtuvieron 9 modelos de predicción con exactitud del 83-100%. De los cuales, los algoritmos de Máquinas de Vectores de Soporte y Clasificación de Vector Nu-Apoyo (NuSVC) brindaron la exactitud más alta (100%) mientras que los algoritmos correspondientes a Perceptrones Multicapa (MLP) y K-Vecinos Más Cercanos (KNN) obtuvieron la exactitud más baja con 90% y 83% respetivamente. Estos resultados respaldan la efectividad de la metodología propuesta, destacando su potencial para una detección precisa y eficiente de la enfermedad de Chagas.

[*] Trabajo de Grado

[**] Facultad de Ciencias. Escuela de Química. Director: Enrique Mejía Ospino. Dr. En Ciencias Químicas. Codirectora: Yuly Andrea Prada Vargas. Dra. En Química.

Abstract

Title: Implementation of a diagnostic method for Chagas disease using mass spectrometry (MALDI-TOF) and machine learning*

Author(s): Yenny Fernanda Velandia Hernandez**

Key Words: Chagas disease, diagnostic, Machine learning

Description: Chagas disease, caused by the *Trypanosoma cruzi* parasite, poses a persistent and potentially lethal threat. Confirmatory tests are crucial to prevent the progression of the disease to critical stages and to avoid consequences such as heart failure and death. In diagnosis, initial techniques based on blood stains and serological tests have encountered challenges with false negatives and positives at various stages of the disease.

In this study, Matrix-Assisted Laser Desorption/Ionization Time-Of-Flight Mass Spectrometry (MALDI-TOF MS) was employed along with computational tools based on supervised machine learning to formulate a predictive model of Chagas disease. This novel approach to diagnosis followed a sequential process that began with sample selection, determination of total proteins present in serum, sample preparation by enzymatic protein digestion, and finally obtaining proteomic profiles for formulating predictive models based on machine learning.

As a result of the study, 9 prediction models were obtained with an accuracy ranging from 83-100%. Among these, Support Vector Machines (SVM) and Nu-Support Vector Classification (NuSVC) algorithms provided the highest accuracy (100%), while Multilayer Perceptrons (MLP) and K-Nearest Neighbors (KNN) algorithms achieved the lowest accuracy with 90% and 83%, respectively. These results support the effectiveness of the proposed methodology, highlighting its potential for accurate and efficient detection of Chagas disease.

---

* Degree Work

** Facultad de Ciencias. Escuela de Química. Director: Enrique Mejía Ospino. Dr. En Ciencias Químicas. Codirectora: Yuly Andrea Prada Vargas. Dra. En Química.

Introducción

La enfermedad de Chagas (EC) o Tripanosomiasis Americana es una enfermedad tropical causada por el parásito protozoario *Trypanosoma cruzi* (Canals et al., 2017) endémica en 21 países de las Américas. La EC afecta aproximadamente a 6 millones de personas y otros 75 millones de personas están en riesgo de infección (World Health Organization, 2021); siendo Colombia el quinto país con la quinta carga mundial más grande de EC, con un estimado de entre 700.000 y 1.200.000 personas infectadas con *Trypanosoma cruzi* y otras 131.000 afectadas por miocardiopatía relacionada con EC (Instituto Nacional de Salud, 2023; Olivera et al., 2019; World Health Organization, 2021)

A pesar de la alta prevalencia de la EC en Colombia, el acceso al diagnóstico y tratamiento sigue siendo un desafío importante. Se estima que solo el 0,4% de los casos recibe tratamiento anti-tripanosómico oportuno, debido a los retrasos en el diagnóstico confirmatorio (Cucunubá et al., 2017a) Además, menos del 1% de las personas afectadas acceden a un diagnóstico y tratamiento digno y debido a barreras sociales como la falta de información, la inequidad en el acceso a servicios de salud y la escasez de herramientas de diagnóstico confirmatorias (Organización Panamericana de la Salud, 2020; World Health Organization, 2021) esto tiene graves consecuencias para la salud pública como la alta progresión de la enfermedad y complicaciones miocárdicas letales (Cantey et al., 2019b; Moncayo & Silveira, 2009)

Por lo tanto, este trabajo propone desarrollar una estrategia de diagnóstico alternativa para la enfermedad de Chagas basada en el análisis de sueros sanguíneos. La estrategia se basa en técnicas de espectrometría de masas con ionización/desorción asistida por una matriz (MS-MALDI) y analizador de tiempo de vuelo (TOF) para adquirir los perfiles proteómicos de las

muestras de pacientes seronegativos, asintomáticos y sintomáticos. En este sentido, se logró recopilar bases de datos proteicas de cada una de las clases mencionadas previamente y se analizaron mediante aprendizaje automatizado supervisado. Como resultado, se desarrollaron 9 algoritmos de predicción que posibilitaron el diagnóstico de la EC con una precisión promedio del 83%.

1. Objetivos

1.1 Objetivo General

Desarrollar una estrategia de análisis basada en espectrometría de masas MALDI- TOF y modelos de aprendizaje Machine Learning para el diagnóstico de la enfermedad de Chagas.

1.2 Objetivos Específicos

1. Implementar un protocolo de preparación de muestras sanguíneas de pacientes con enfermedad de Chagas para la obtención de perfiles proteómicos por Espectrometría de masas MALDI-TOF
2. Establecer los parámetros instrumentales del espectrómetro de masas MALDI-TOF adecuados para la adquisición de espectros de calidad como afluencia del láser, método de ionización, relación matriz-muestras y rango de masas.
3. Construir un modelo predictivo para el diagnóstico de la enfermedad de Chagas a partir de los perfiles proteómicos obtenidos por MS-MALDI-TOF usando Machine Learning.

2. Estado del arte

En la última década, la identificación de enfermedades tropicales como la malaria, el dengue y la enfermedad de Chagas en los laboratorios de diagnóstico clínico se basaba principalmente en técnicas convencionales de identificación de secuencias genéticas y fenotípicas. El desarrollo de dispositivos de espectrometría de masas ha revolucionado la identificación y caracterización de rutina de péptidos y proteínas de microorganismos tropicales, permitiendo un diagnóstico más rápido, preciso y eficiente de estas enfermedades. Un ejemplo notable es el uso de espectrometría de masas para la identificación de biomarcadores de malaria en la sangre, lo que permite un diagnóstico temprano y el tratamiento oportuno de la enfermedad (Laroche et al., 2017); otro uso reciente de la espectrometría de masas en la microbiología clínica fue analizar metabolitos en el líquido cefalorraquídeo para diagnosticar pacientes con errores congénitos del metabolismo, lo que evidencia su utilidad en el diagnóstico de trastornos metabólicos (Caro-Miró et al., 2022) y no sólo eso, también, se ha aplicado para la detección de drogas en la saliva, consiguiendo detectar marihuana, cocaína, opiáceos, entre otras lo que demostrando su potencial en toxicología clínica y forense (Rotemberg et al., 2022).

El desarrollo del aprendizaje automático también ha sido significativo en el ámbito del diagnóstico clínico, mejorando su precisión y eficacia. Por ejemplo, Morais et al. (2022) llevaron a cabo la detección del parásito *Trypanosoma cruzi* en frotis de sangre mediante un enfoque de aprendizaje automático aplicado a imágenes de teléfonos móviles. Utilizando tres tipos de modelos, lograron una precisión que oscilaba entre el 87% y el 90%.

También, la revisión de la literatura revela la eficacia de la combinación de espectrometría de masas y aprendizaje automático en una amplia gama de aplicaciones, desde el diagnóstico de

enfermedades tropicales hasta la detección de índices de incendios. Por ejemplo, Croxatto et al., (2012) demostraron que este acople entre técnicas ayudó a identificar correctamente el 98% de los casos de malaria, con una especificidad del 99%. De manera similar, Maldonado et al., (2018); encontraron que la espectrometría de masas con aprendizaje automático logró identificar el 95% de los casos de dengue, con una especificidad del 97%. Además, Kingsley et al. (2023) introdujeron una nueva aplicación que combina la espectrometría de masas con técnicas de aprendizaje automático en tiempo real para la detección temprana de indicadores químicamente específicos de incendios y eventos relacionados, resaltando la versatilidad y el potencial de este enfoque innovador.

Para ilustrar aún más el potencial de esta tecnología, a continuación, se presenta una tabla que resume algunos estudios recientes que han utilizado la espectrometría de masas y el aprendizaje automático para el diagnóstico de enfermedades:

Tabla 1 . Revisión de la literatura

Autores	Tecnología	Investigación
(SINGHAL et al., 2016)	MALDI-TOF MS	Recopilación de la identificación de patógenos bacterianos y fúngicos, ideal para la detección temprana como Leishmaniasis, Giardiasis, Criptosporidiosis y Amebiasis.
(Cobo, 2013)	MALDI-TOF MS	Identificación de varias mutaciones en virus y de diferentes cepas de microorganismos, lo que podría ayudar al diagnóstico rápido y preciso de virus
(Bader, 2013; Vella et al., 2017)	MALDI-TOF MS	Identificación de especies de hongos y fármacos antimicóticos.

(Bader, 2013; Vella et al., 2017)	ML-MS	Diagnóstico basado en el perfil de espectros para el diagnóstico de SARS-CoV-2.
(Y.-C. Huang et al., 2020)	ML-MS	Desarrollo de una herramienta predictora de cáncer de mama basado en aprendizaje automatizado con perfiles lipídicos y metabólicos
(Zhang et al., 2022)	ML-MALDI TOF MS	Identificación de muestras de leche bovina procesadas térmicamente a partir de sus señales peptídicas a partir de algoritmos ML.
(Y. Li et al., 2022)	ML-MALDI TOF MS	Desarrollo de nuevos métodos de clasificación para identificar las especies de Listeria mediante la extracción de características latentes de los datos de MALDI-TOF MS.
(Mortier et al., 2021)	ML-MALDI TOF MS	Identificación de especies bacterianas mediante espectrometría de masas MALDI-TOF y técnicas de aprendizaje automático.
(Timm et al., 2008)	ML-MALDI TOF MS	Desarrollo de técnicas de ML para la predicción de intensidad máxima para espectros de masas MALDI.

3. Marco teórico

3.1 Enfermedad de Chagas: Enfermedad y epidemiología

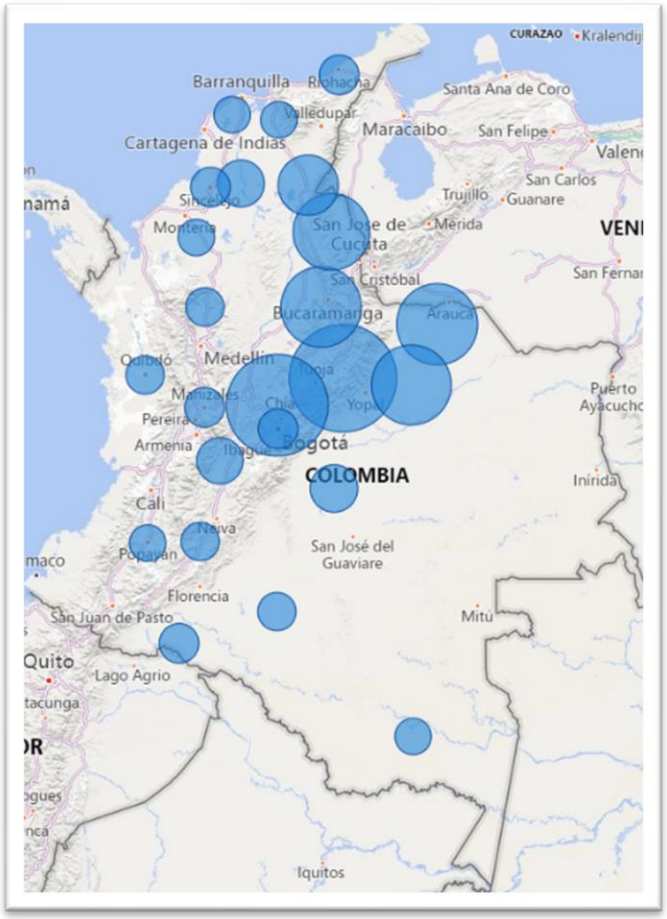
Según la OMS, existen 5.742.167 personas infectadas con la enfermedad de Chagas, lo que provoca cerca de 13.000 muertes al año y una gran carga de morbilidad en las poblaciones afectadas (Instituto Nacional de Salud, 2023; Moncayo & Silveira, 2009). Esta enfermedad, que antes se consideraba endémica del hemisferio sur se ha extendido a países debido a la alta migración. Esta expansión representa un problema de salud mundial, ya que aumenta la incidencia en la morbilidad de la población mundial (Institute of Medicine, 2008).

La enfermedad se presenta en dos fases: i) la aguda que dura aproximadamente dos meses después de la infección y es el periodo con un elevado número de parásitos circundantes en la sangre (Organización mundial de la salud, 2022) ii) la crónica en la cual hay un mayor porcentaje de parásitos en el músculo cardíaco y en el sistema digestivo. Actualmente, un 30% de los pacientes con EC sufren alteraciones cardiacas y hasta un 10% sufren afecciones digestivas (típicamente agrandamiento del esófago y/o colon), neurológicas o mixtas. En los años posteriores a la infección, las patologías miocárdicas pueden conducir a la muerte súbita (Moncayo & Silveira, 2009).

En Colombia, la enfermedad de Chagas está bajo vigilancia en el sistema de salud pública (Sivigila), el cual registra tanto los casos agudos como crónicos. En la actualidad, los departamentos con mayor prevalencia de EC son: Boyacá, Cundinamarca, Arauca, Santander, Casanare, Norte de Santander, Cesar y Meta, como se ilustra en la Figura 1. En estas áreas, se han

identificado 25 especies de triatominos responsables de transmitir el *Trypanosoma cruzi* (Guhl, 2005).

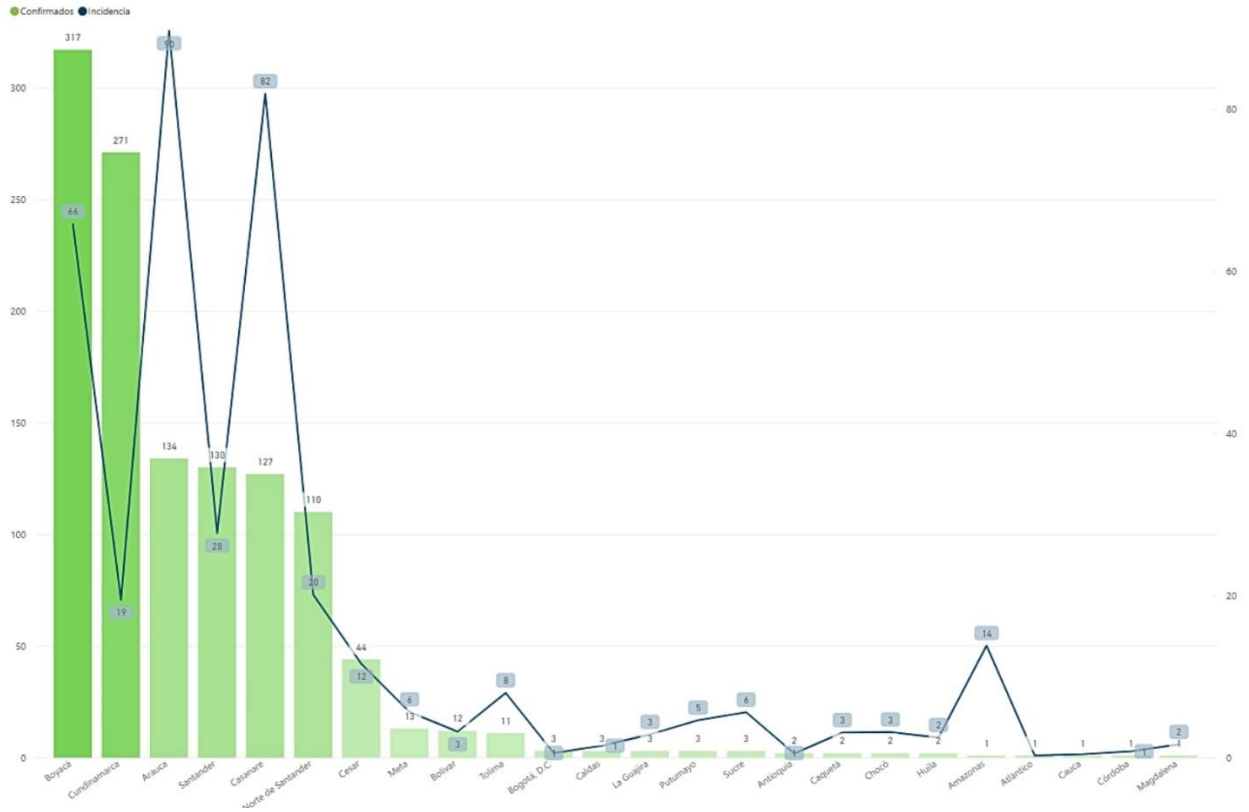
Figura 1 Mapa de la fuerza de infección de la enfermedad de Chagas en Colombia (2022–2023)



Nota: Tomado de (Instituto Nacional de Salud, 2023)

Entre 2022 y 2023, Sivigila registró un total de 2090 casos acumulados, de los cuales 1315 corresponden a la fase aguda de la EC. En el caso del departamento de Santander durante el mismo período, se notificaron 280 casos, de los cuales 130 fueron confirmados los cuales representan aproximadamente el 10% del total de diagnósticos reportados en el país como se muestra en la Figura 2 (Instituto Nacional de Salud, 2023).

Figura 2 Prevalencia y número de casos de la EC en Colombia.

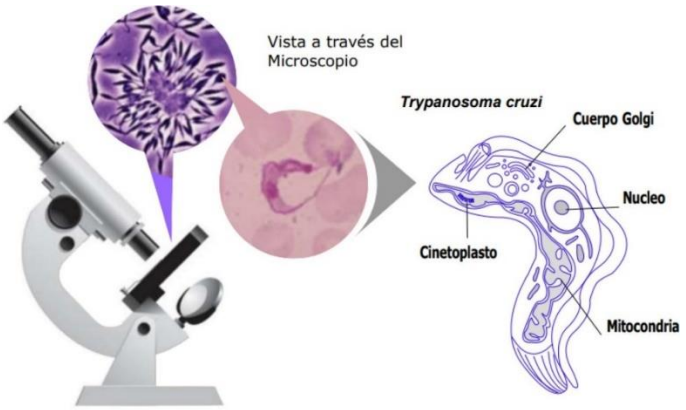


Nota: Tomade de (Instituto Nacional de Salud, 2023)

3.2 Taxonomía y ciclo de vida del *T. Cruzi*

La enfermedad de Chagas es causada por el parásito *Trypanosoma cruzi*, un protozoo flagelado unicelular que pertenece al género *Trypanosoma*. Este parásito se encuentra en sangre y tejidos de diversas especies animales (Figura 3), incluyendo aves, reptiles, anfibios, peces y mamíferos. Se transmite a través de vectores intradomiciliarios como la "chinche besadora" o "vinchuca", artrópodos hematófagos de la familia *Reduviidae* y subfamilia *Triatominae* (Palmezano Díaz et al., 2015).

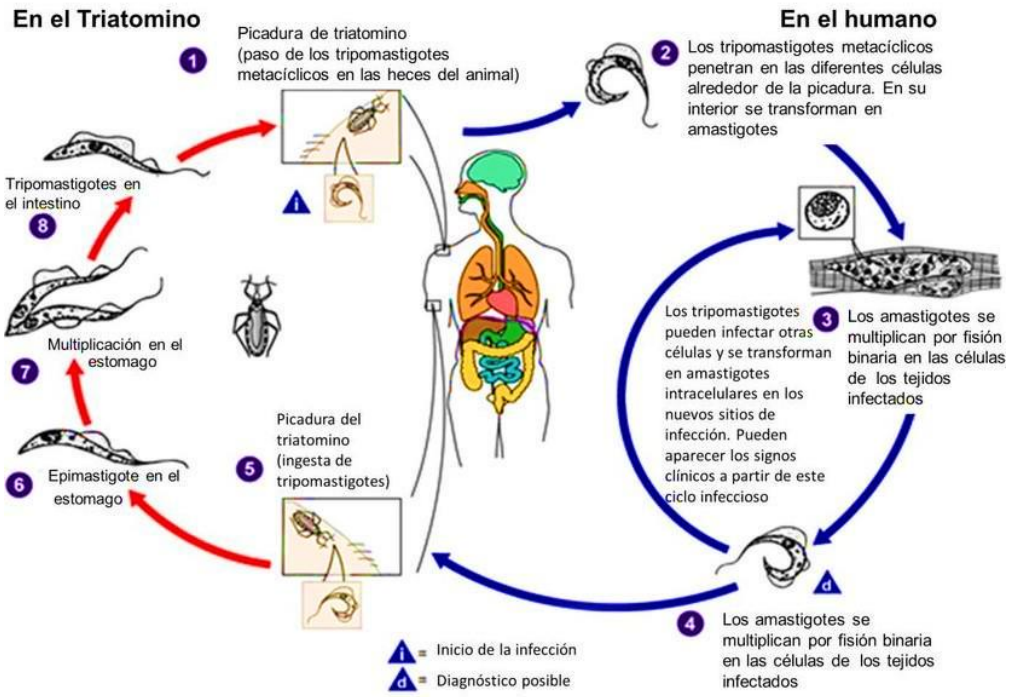
Figura 3 Trypanosoma cruzi a través del microscopio.



Nota: Tomado de (Varona uribe et al., 2014)

Este parásito causante de la enfermedad de Chagas tiene un ciclo de vida complejo que involucra a los triatominos como vectores transmisores. Como puede observarse en la Figura 4, el *Trypanosoma cruzi* tiene un ciclo de vida complejo que se desarrolla en dos huéspedes: los triatominos y los mamíferos.

Figura 4 Ciclo de vida de la infección por tripanosomiasis americana en humanos



Nota. Tomado de (Cantey et al., 2019a)

Un insecto vector triatomino infectado libera tripomastigotes en sus heces cerca del sitio de la mordedura. Estos ingresan al huésped a través de la herida o membranas mucosas como la conjuntiva. Los vectores comunes son *Triatom, Rhodnius y Panstrongylus*. Una vez en el huésped, los tripomastigotes invaden células cercanas, convirtiéndose en amastigotes intracelulares que se multiplican y se transforman nuevamente en tripomastigotes en la sangre. Estos infectan células de diversos tejidos, generando manifestaciones clínicas. A diferencia de los tripanosomas africanos, los tripomastigotes del torrente sanguíneo no se replican, reanudando la replicación al ingresar a otra célula o ser ingeridos por otro vector. La "chinche besucona" se infecta al alimentarse de sangre con parásitos circulantes. Los tripomastigotes ingeridos se transforman en epimastigotes en el intestino medio del vector, multiplicándose y diferenciándose en tripomastigotes metacíclicos infecciosos en el intestino posterior (Cantey et al., 2019a) (Cantey et al., 2019a).

3.3 Diagnóstico de EC

La EC presenta desafíos en su detección y tratamiento (Alducin-Téllez et al., 2011). El diagnóstico varía según la etapa de la enfermedad. Durante la fase aguda, la observación del parásito en un frotis de sangre confirma la infección. Sin embargo, la sensibilidad de estas pruebas disminuye con el tiempo debido a la reducción de parásitos en la sangre. Después de 20 a 30 días, se pueden usar pruebas de concentración para mejorar la sensibilidad. En casos de EC aguda no congénita, se sugiere realizar pruebas de seguimiento para un diagnóstico más preciso (de Andrade et al., 2011; Palmezano Díaz et al., 2015).

En la fase aguda de Chagas, se producen anticuerpos IgM, como en otras enfermedades infecciosas, detectables entre 10 y 15 días después de los síntomas. Sin embargo, la falta de kits comerciales específicos dificulta la obtención de controles positivos (Herazo et al., 2022). Los altos niveles de IgM no son concluyentes, ya que pueden encontrarse en algunos pacientes con enfermedad crónica. El factor reumatoide, elevado en la fase aguda de enfermedades infecciosas, puede generar falsos positivos en las pruebas de IgM. Además, la mayoría de los recién nacidos infectados no presentan IgM (Luquetti & Schmuñis, 2017).

En la EC congénita, se necesita un examen parasitológico negativo al nacer y la ausencia de IgG anti-T. cruzi después de los 9 meses, junto con la presencia de anticuerpos maternos. La PCR en tiempo real (rtPCR) y otras técnicas como la inoculación en animales o el xenodiagnóstico pueden ser útiles en casos específicos. La reactivación se diagnostica mediante la detección directa de parásitos o análisis cuantitativos que muestren un aumento gradual de la parasitemia. La serología sólo es útil en receptores seronegativos de órganos de donantes seropositivos. Por lo tanto, el enfoque diagnóstico de la enfermedad de Chagas requiere una combinación de herramientas y un abordaje multidisciplinario (Sousa et al., 2023; Picado et al., 2018).

En la fase crónica, el diagnóstico parasitológico directo es difícil debido a la baja parasitemia habitual. Por lo tanto, la mayoría de las pruebas buscan anticuerpos IgG anti-T.cruzi, detectados de por vida en personas no tratadas. Aproximadamente el 75-80% de las personas con infección presentan altos niveles de anticuerpos, mientras que un 15% tiene niveles intermedios y un 5% niveles bajos, lo que complica el seguimiento del tratamiento. La curación espontánea es muy rara, estimada en menos del 1%.

Debido a que las pruebas de detección no son concluyentes, la OMS (World Health Organization, 2002) y la OPS (Pan American Health Organization, 2019) recomiendan el uso de

al menos dos pruebas serológicas que representen diferentes principios metodológicos: una de alta sensibilidad y otra de alta especificidad. Si ambas pruebas son reactivas, el suero es positivo y pertenece a un individuo infectado con *T.cruzi*. Si ambos no reaccionan, el individuo no está infectado. Si una prueba es reactiva y la otra no reactiva, se debe utilizar una tercera prueba (Cucunubá et al., 2017b; Herazo et al., 2022; Whitman, 2023).

3.4 Tratamiento de la enfermedad de Chagas

El diagnóstico temprano de la EC es crucial para su tratamiento efectivo. Los estudios, como el realizado por (Morillo C.A et al., 2015), han demostrado que la eficacia de los medicamentos disponibles, benznidazol y nifurtimox, es significativamente mayor en la fase aguda de la enfermedad. En esta etapa, la parasitemia es alta, lo que facilita la eliminación del parásito con el tratamiento. A diferencia, en la fase crónica, la parasitemia disminuye considerablemente, lo que reduce la eficacia de los medicamentos. Por lo tanto, el diagnóstico temprano y el tratamiento oportuno de la enfermedad de Chagas son esenciales para prevenir la progresión a la fase crónica y así evitar complicaciones graves.

3.5 Ciencias Ómicas

Las ciencias ómicas representan un conjunto de disciplinas interrelacionadas que se centran en el análisis global de componentes biológicos específicos dentro de un organismo o sistema biológico. Estas disciplinas incluyen la genómica, transcriptómica, proteómica y metabolómica. Cada una de estas áreas se enfoca en estudiar aspectos específicos de la biología molecular y

celular, contribuyendo al entendimiento holístico de la estructura y función de los sistemas biológicos (Regan et al., 2019).

Dentro del conjunto de ciencias ómicas, la proteómica emerge como una disciplina clave para el estudio exhaustivo de las proteínas en un sistema biológico dado. Esta se encarga de investigar la totalidad de las proteínas, sus interacciones y modificaciones en un contexto celular o tisular empleando distintas metodologías de análisis como la electroforesis en gel y la espectrometría de masas, que permiten mapear expresiones proteicas, analizar modificaciones y detectar biomarcadores de enfermedades(Molassiotis et al., 2013). Aunque su aplicación clínica aún está en desarrollo, la proteómica se utiliza para identificar proteínas en muestras biológicas, comparar perfiles proteómicos y estudiar sus interacciones (Agrawal et al., 2013).

La metodología proteómica sigue cuatro pasos: separación de proteínas, digestión, detección de fragmentos peptídicos por espectrometría de masas e identificación de proteínas (Qian et al., 2023). Actualmente, se emplea para identificar proteínas en muestras biológicas, comparar perfiles proteómicos, estudiar interacciones proteicas y encontrar biomarcadores para diagnóstico y tratamiento de enfermedades (Al-Amrani et al., 2021; Qian et al., 2023; Wilkins et al., 1996).

3.6 Espectrometría de masas MALDI-TOF

La espectrometría de masas MALDI-TOF es una técnica analítica revolucionaria que ha transformado diversos campos como el diagnóstico clínico, la seguridad alimentaria y la vigilancia ambiental. Su capacidad para identificar y caracterizar moléculas con precisión y rapidez la ha

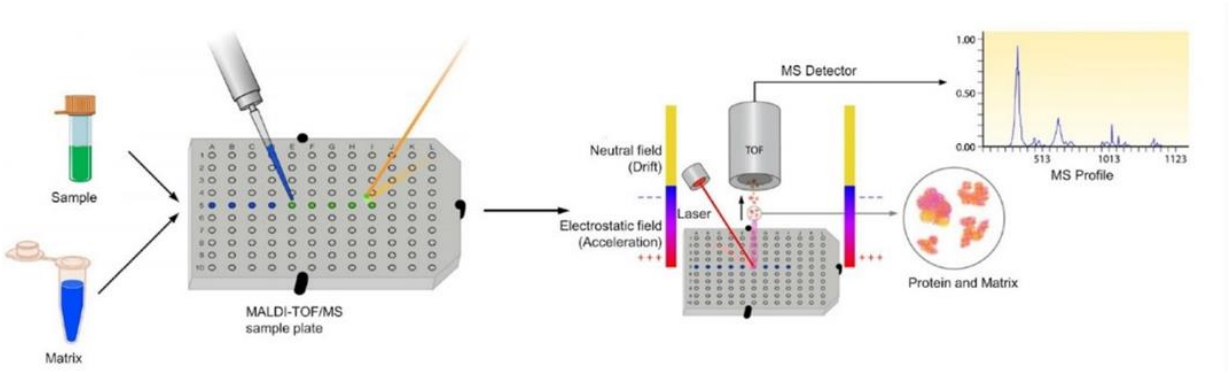
convertido en una herramienta indispensable en la investigación científica y la práctica clínica (Gross, 2017).

Está técnica se basa en la detección de moléculas respecto a su relación masa-carga (m/z) una vez son ionizadas. Mediante esta técnica pueden ser analizadas una amplia gama de moléculas, incluidos polímeros, péptidos, lípidos, metabolitos y pequeñas moléculas (Carbonnelle et al., 2011). La identificación de biomoléculas se basa en el perfil de masas característico obtenido a partir de las señales correspondientes a los iones, el cual representa la "huella digital" de la muestra (Dieckmann & Malorny, 2011).

En la espectrometría de masas MALDI-TOF, la muestra o el analito se co-cristaliza con la matriz y se ioniza mediante de la radiación de un láser. En este sentido, la matriz absorbe la energía fotónica del láser, desorbiendo e ionizando efectivamente las moléculas.

Una vez, formados los iones estos atraviesan un tubo de tiempo de vuelo donde son separados en función de su tiempo de llegada al detector, generando un espectro característico para cada muestra como se ilustra en la Figura 5; para el caso del análisis de la digestión enzimática de proteínas, a este espectro se le denomina ‘huella peptídica’(Benagli et al., 2011). Esta información puede ser tratada y analizada empleando herramientas computacionales y bases de datos espectrales para finalmente desarrollar métodos de diagnóstico clínico e identificación de diferentes analitos y marcadores moleculares de interés biológico (Momo et al., 2013).

Figura 5 Esquema de análisis de espectrometría de masas MALDI-TOF



Nota: Tomado de: (Raiesi et al., 2019)

3.7 Machine Learning

ML es un subcampo de la inteligencia artificial (IA) y consta de algoritmos de reconocimiento de patrones que se utilizan para definir relaciones y hacer predicciones después de los conjuntos de datos de entrenamiento. El poder de los algoritmos de ML proviene de la capacidad de aprender automáticamente estos patrones de grandes conjuntos de datos para las predicciones, lo que permite al usuario obtener conocimiento a partir de datos anteriores y aplicarlo a predicciones futuras (Nachtigall et al., 2020).

El uso de la herramienta de aprendizaje automático para discriminar enfermedades a partir de datos distintos transforma automáticamente los datos sanitarios en minutos, en vez de semanas o meses, para que los médicos aceleren el diagnóstico y proporcionen una intervención temprana. Esto se debe a que Machine Learning ayuda a estructurar, normalizar e interpretar los datos sanitarios, de manera que pueda ser utilizado para la toma de decisiones rápidas, ya sea un diagnóstico preciso mediante secuenciación proteómica, detección temprana de cáncer o

visualización cardíaca avanzada con modelos de Machine Learning personalizados (Liebal et al., 2020).

Dentro del alcance de la ciencia bioquímica, el aprendizaje automático está impulsando descubrimientos en campos no explorados en el conocimiento derivado del ser humano. Los análisis proteómicos abarcan una complejidad que va más allá de la simple identificación de todos los péptidos no modificados en el proteoma y de aquí viene el principal problema a la hora del análisis. Por eso, en el aprendizaje supervisado, un subconjunto del campo del aprendizaje automático aprovecha herramientas matemáticas generalizadas que usan datos de tipos de muestras conocidas para hacer predicciones sobre muestras desconocidas. Los ejemplos de tipos de muestras incluyen un estado de enfermedad frente a uno saludable, o péptidos con una secuencia o característica particular, generando resultados más deseables para la automatización, adaptándose a la tendencia de los datos (Desaire et al., 2022; Yakimovich et al., 2021).

Hay muchos modelos y clasificadores de aprendizaje automático diferentes. Algunos de los más comunes incluyen:

3.7.1 Máquinas de vectores de soporte (SVM).

Las máquinas de vectores de soporte constituyen uno de los métodos de aprendizaje automático con supervisión con una amplia variedad de aplicabilidad para solucionar problemas de clasificación y regresión. Entre sus aplicaciones se encuentran el procesamiento de señales en medicina, procesamiento del lenguaje natural y reconocimiento de imágenes y voz (Leng et al., 2013).

La función matemática utilizada para la transformación se conoce como función kernel; esta herramienta matemática permite mapear los datos a un espacio de características de mayor dimensión. En este espacio de mayor dimensión brinda más precisión en la clasificación y predicción de los datos (MATLAB, 2023a).

Tabla 2 Tipos de Kernel en SVM

Tipo de Kernel	Descripción	Ventajas	Desventajas
Lineal	La forma más simple de kernel. Funciona bien cuando los datos ya son linealmente separables.	Sencillo y eficiente.	Puede no funcionar bien para datos no lineales.
Polinomial	Eleva los datos a una potencia determinada. Puede ayudar a separar las clases de datos que no son linealmente separables.	Puede ser muy flexible para modelar relaciones no lineales.	Puede ser computacionalmente costoso para grandes conjuntos de datos.
RBF (Función de base radial)	Se basa en la distancia euclidiana entre los puntos de datos. Uno de los kernels más populares.	Generalmente funciona bien para una variedad de problemas.	Puede ser sensible a la elección del parámetro gamma.
Sigmoide	Similar al kernel del tipo polinomial, pero utiliza una función sigmoide.	Puede ser útil para modelar relaciones no lineales complejas.	Puede ser computacionalmente costoso y puede tener problemas de convergencia.

Nota: Adaptado de (Leng et al., 2013; MATLAB, 2023a).

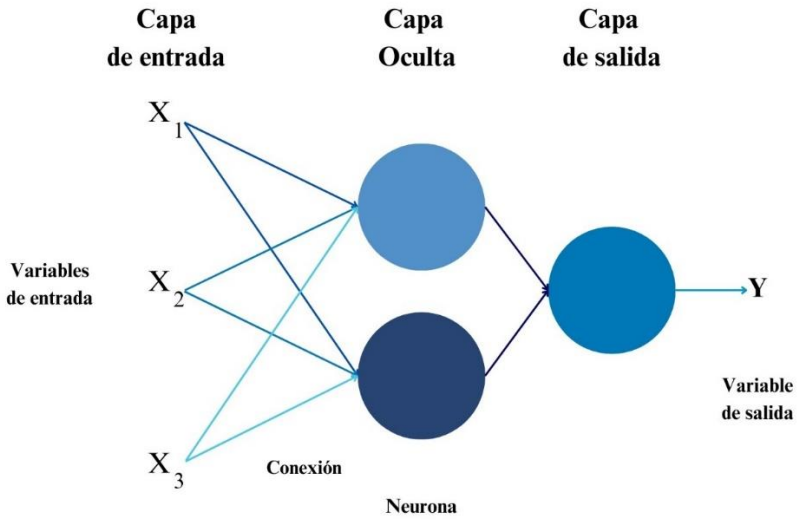
3.7.2 Redes neuronales (NN).

Las redes neuronales son otro enfoque de ML existente para la selección de modelos que pertenece a un grupo de algoritmos de aprendizaje supervisado. Se utiliza para identificar el modelo que mejor se ajusta a un conjunto de datos determinado. Para ello, la red se entrena con un conjunto de datos etiquetados, donde cada etiqueta representa el modelo óptimo para cada caso

(Amazon, 2023; MATLAB, 2023b; Yuan et al., 2023). Posteriormente, la red puede utilizarse para predecir el modelo más adecuado para nuevos conjuntos de datos sin necesidad de etiquetas.

Una red neuronal básica tiene neuronas artificiales interconectadas en tres capas, como lo muestra la Figura 6:

Figura 6 Arquitectura de una Red Neuronal simple.



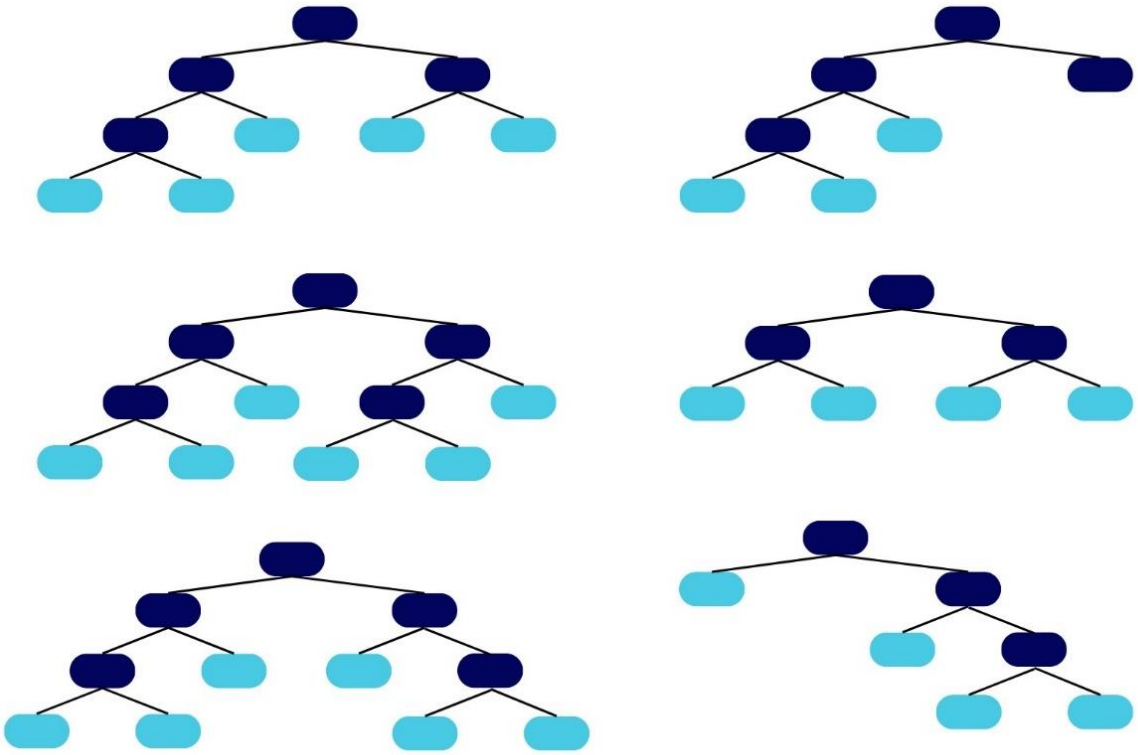
- Capa de entrada; es aquella que recibe los datos de entrada y los nodos de entrada procesan los datos, los analizan o los clasifican y los pasan a la siguiente capa. La cantidad de neuronas en esta capa depende del número de características o variables independientes del problema (Amazon, 2023).
- Capa Oculta; en esta fase se procesan los datos de entrada y extrae características relevantes de la capa de entrada o de otras capas antecesoras, cada capa oculta está compuesta por un conjunto de neuronas interconectadas(Amazon, 2023)

- Capa de Salida; la capa de salida proporciona el resultado final de todo el procesamiento de datos que realiza la red neuronal artificial. (Amazon, 2023; Barrios Peréz Camilo, 2016).

3.7.3 Bosque aleatorio (*Random Forest*)

Es un algoritmo comúnmente usado en el aprendizaje supervisado que pertenece al aprendizaje por conjuntos, que combina múltiples árboles predictores (Figura 7), cada uno de los cuales depende de los valores de un vector aleatorio muestreado de forma independiente y con la misma distribución para todos los árboles del bosque (Breiman, 2001).

Figura 7 Visualización del funcionamiento de Random Forest.



Se basa en el aprendizaje conjunto, utilizando árboles de decisión como su clasificador básico (Breiman, 2001; Raicu & Bologa, 2019). Ampliamente utilizado para clasificación y regresión, ofrece alta precisión y capacidad para calcular errores de generalización, identificar variables importantes, detectar valores atípicos e imputar valores faltantes. Además, se ha aplicado eficientemente en problemas semi-supervisados, manteniendo una alta precisión y eficiencia computacional tanto en entrenamiento como en evaluación (Osmangazi Tıp Dergisi Osmangazi ; Ozen & Bal, 2020). Su excepcional rendimiento predictivo lo ha convertido en un método importante en varios campos.

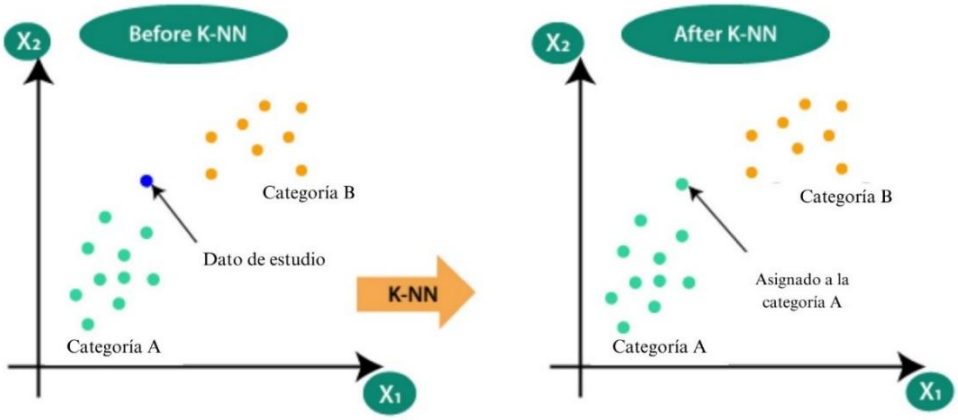
3.7.4 Clasificador de descenso de gradiente estocástico (SDG).

Es un algoritmo de optimización iterativo que se utiliza en el aprendizaje automático para encontrar los valores de los parámetros que minimizan una función de pérdida (Nugroho et al., 2020a). Es una variante del algoritmo de descenso del gradiente, que utiliza el gradiente de la función de pérdida para actualizar los valores de los parámetros. SGD funciona calculando el gradiente de la función de pérdida con respecto a un solo ejemplo de entrenamiento, y luego usando ese gradiente para actualizar los valores de los parámetros (Pedregosa et al., 2011). Este es un algoritmo eficiente para optimizar modelos de aprendizaje automático, ya que solo requiere calcular el gradiente con respecto a un solo ejemplo de entrenamiento en cada iteración. Esto lo hace adecuado para entrenar modelos en grandes conjuntos de datos (Nugroho et al., 2020b).

3.7.5 Vecinos más cercanos (KNN).

KNN es un algoritmo de aprendizaje automático supervisado utilizado ampliamente en clasificación y regresión (Nedyalkova et al., 2022). Este clasificador no paramétrico se basa en la proximidad y la similitud para predecir la agrupación de datos individuales, destacándose por su buena precisión y simplicidad, como se muestra en la Figura 8 (S. Huang et al., 2020; Nedyalkova et al., 2022). Su aplicación abarca diversos dominios, desde la evaluación de riesgos viales hasta en medicina debido a su alta precisión en el reconocimiento de conjuntos de datos médicos (S. Huang et al., 2020).

Figura 8 Ilustración grafica del algoritmo KNN



Nota: Para clasificar un punto de datos X_1 en las categorías A o B, usamos el algoritmo K-NN se elige el número de "vecinos", k, y se calcula la distancia euclidiana para asignar el punto a la categoría con la mayoría de vecinos más cercanos. Tomado y adaptado de (JAVA, 2019).

3.7.6 Regresión logística (*Logistic Regression*)

Es un método estadístico utilizado para la clasificación binaria, donde el resultado es una variable categórica con dos niveles. El modelo de regresión logística se utiliza ampliamente en diversos campos, como la medicina, la psicología y la informática. Es un algoritmo de aprendizaje supervisado que pretende predecir la probabilidad de un resultado binario según una o más variables predictoras. El algoritmo funciona ajustando una función sigmoidea a los datos de entrada, lo que le permite modelar la relación entre las variables independientes y la probabilidad de un resultado particular (Subasi, 2020; Zaidi, 2022). Esto hace que la regresión logística sea particularmente útil para comprender la influencia de múltiples factores en un resultado binario, como la presencia o ausencia de una enfermedad.

En el contexto del aprendizaje automático, la regresión logística se utiliza a menudo como punto de referencia para evaluar el rendimiento de otros algoritmos de clasificación. También se utiliza en combinación con otros métodos de aprendizaje supervisado, como máquinas de vectores de soporte, para mejorar la precisión de la detección automatizada de objetos en datos experimentales (Subasi, 2020).

3.7.7 Árboles extra (*ExtraTrees*)

Es un método de aprendizaje conjunto utilizado en problemas de regresión y clasificación supervisada. Es similar al algoritmo de *Random Forest*, que utiliza árboles de decisión como estimador base; su diferencia radica la estrategia de partición que usa, ya que esta es completamente aleatoria para construir los árboles, también, en cada nodo se selecciona un

subconjunto aleatorio de características y un punto de corte aleatorio para dividir el espacio de características (Ma et al., 2022).

El método de ET ajusta varios árboles de decisión aleatorios en diferentes partes de los datos, mejorando la precisión de la predicción y controlando el sobreajuste. El algoritmo *ExtraTrees* construye árboles completamente aleatorios en casos extremos y su estructura es independiente del valor de salida de la muestra de aprendizaje (Chu et al., 2020) el cual brinda una elevada versatilidad en diversos aspectos del lenguaje automático.

3.8 Análisis de componentes principales (PCA)

Una de las técnicas quimiométricas más populares es el análisis de componentes principales (PCA), que se utiliza ampliamente en todas las disciplinas científicas (Rohman & Putri, 2019) Además, los métodos quimiométricos, en particular la calibración multivariada, desempeñan un papel importante en la resolución de problemas complejos, como el análisis de mezclas y el seguimiento de procesos (M. Li et al., 2017).

El PCA es una técnica fundamental en el análisis quimiométrico, ampliamente utilizada para extraer información esencial de conjuntos de datos complejos. Es un método estadístico multivariado que permite reducir la dimensionalidad de los datos reteniendo la mayor cantidad de información posible (Jollife & Cadima, 2016). Este método adaptativo ha sido útil en diversas disciplinas con diferentes tipos de datos. En el contexto de la quimiometría, la PCA se emplea como un método no supervisado para visualizar tendencias en conjuntos de datos y proporcionar la máxima información química mediante el análisis de datos (Jollife & Cadima, 2016).

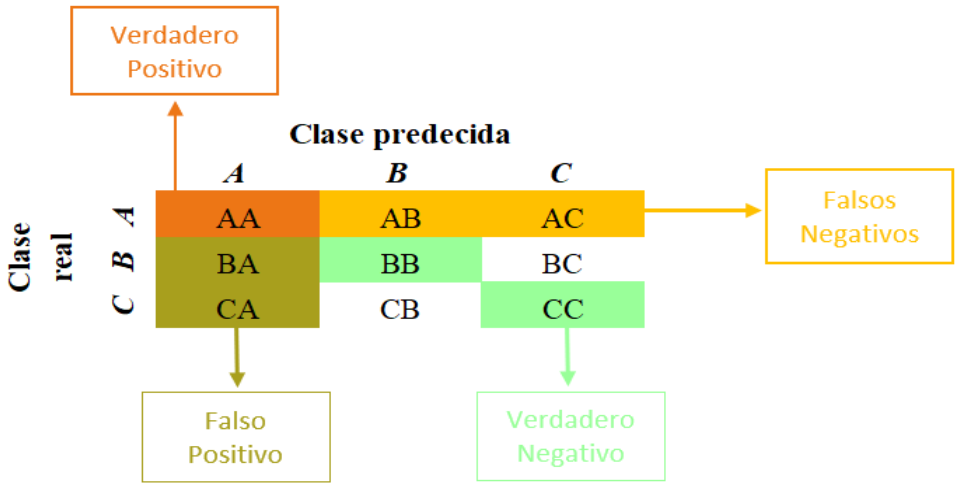
3.9 Parámetros de calidad en un modelo

Para evaluar la calidad de un modelo de aprendizaje supervisado, se pueden emplear varios métodos y métricas. Un enfoque común es utilizar una matriz de confusión, que proporciona un desglose detallado del desempeño del modelo. La matriz de confusión permite calcular diversas métricas de rendimiento, como la precisión, la exactitud, la sensibilidad y la puntuación F1. Estas métricas son esenciales para evaluar la eficacia del modelo de aprendizaje supervisado (Subasi, 2020).

3.9.1 Matriz de confusión

Una matriz de confusión es una tabla que se utiliza a menudo para describir el rendimiento de un modelo de clasificación en un conjunto de datos de prueba cuyos valores verdaderos se conocen. Permite la visualización del desempeño de un algoritmo. La matriz de confusión muestra el número de predicciones verdaderas positivas, verdaderas negativas, falsas positivas y falsas negativas como es ilustrado en la Figura 9.

Figura 9 Interpretación de una Matriz de Confusión.



Nota: La matriz de confusión que se muestra en la Figura 9 es una matriz de 3x3 que se utiliza para evaluar el rendimiento de un modelo de clasificación de tres clases. Las filas de la matriz representan la clase real de las instancias, y las columnas representan la clase predicha por el modelo. En la Figura 9 se seleccionó como clase de interés (A) y se observa que en la diagonal principal se encuentran las predicciones correctas para cada clase, pero la única con los intereses para la variable (A) es (AA) siendo un verdadero positivo y el resto dentro de la diagonal principal verdaderos negativos (BB, CC). Los falsos positivos dentro de la figura son aquellas evaluaciones que se le asignaron a una clase pero que no corresponden a esa clase como lo son (BA, CA) y por último se encuentran los falsos negativos son aquellas predicciones que se le asignan a otras variables, C y B, siendo de A, como es el caso de AB y AC.

3.9.2 Métricas de rendimiento

La evaluación de modelos de aprendizaje supervisado implica el uso de diversas métricas para evaluar su desempeño. Estas métricas proporcionan información sobre la capacidad del modelo para realizar predicciones precisas y su eficacia general (Deng et al., 2016). Algunas de las métricas clave utilizadas para evaluar los modelos de aprendizaje supervisado incluyen (Subasi, 2020):

1. Exactitud (accuracy): Mide la proporción de instancias clasificadas correctamente sobre el total de instancias. Es una métrica fundamental para evaluar el rendimiento general de un modelo. Sin embargo, debe usarse con precaución, ya que puede no proporcionar una imagen completa del desempeño del modelo, especialmente en presencia de desequilibrio de clase.

$$Accurancy = \frac{VP + VN}{(VP + FP + VN + FN)} \quad \textbf{(1)}$$

2. Precisión (precision): Mide la proporción de predicciones positivas verdaderas entre todas las predicciones positivas. Es particularmente útil cuando el costo de los falsos positivos es alto. La precisión ayuda a comprender la capacidad del modelo para evitar falsos positivos.

$$Precision = \frac{VP}{(VP + FP)} \quad \textbf{(2)}$$

3. Sensibilidad (recall): Mide qué tan bien un modelo identifica todos los casos positivos reales. Es importante cuando los falsos negativos son altos. Proporciona información sobre la capacidad del modelo para capturar todas las instancias positivas.

$$Recall = \frac{VP}{(VP + FN)} \quad \textbf{(3)}$$

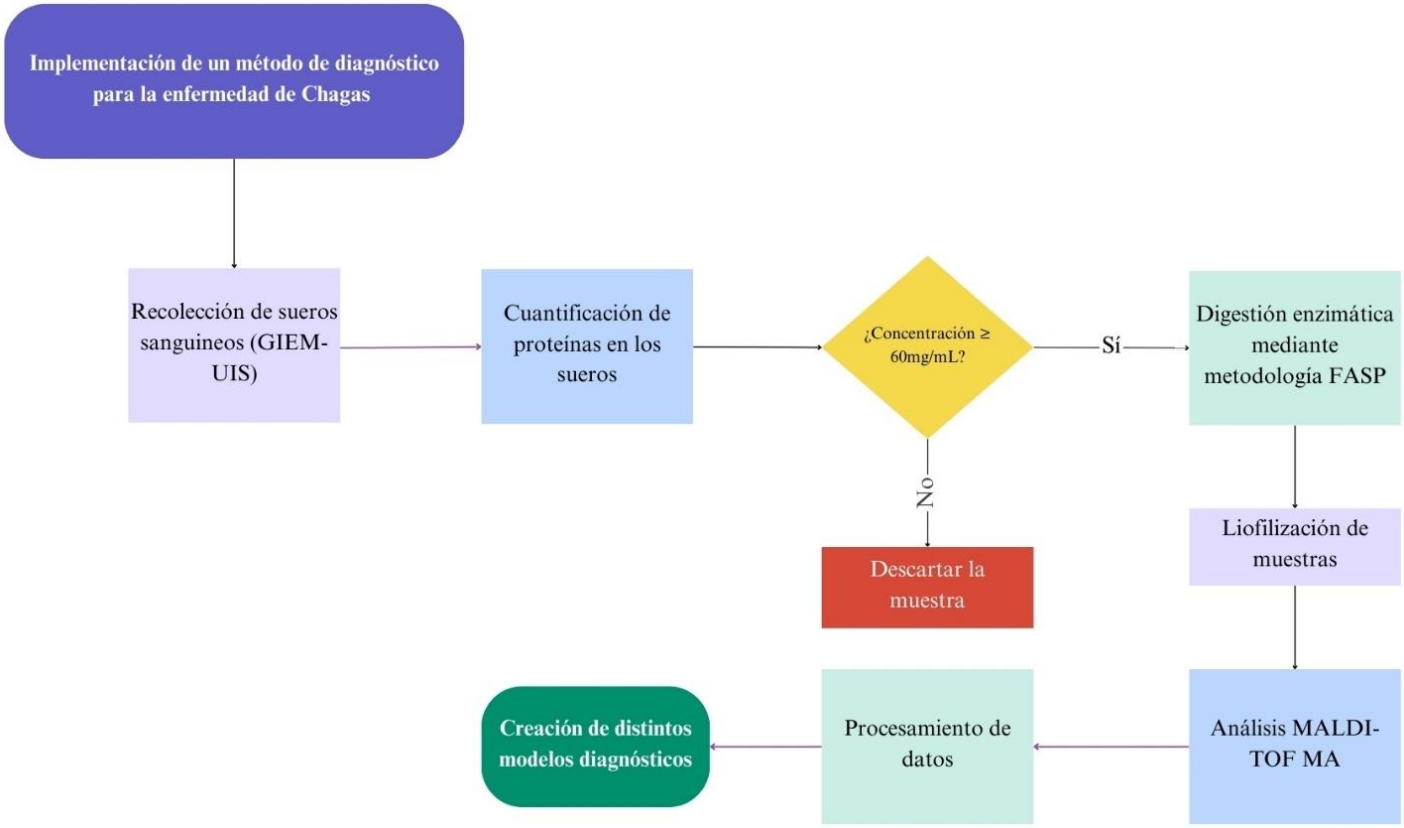
4. Puntuación F1 (F1-score): Es la media armónica de precisión y sensibilidad. Proporciona una medida equilibrada del rendimiento de un modelo, especialmente cuando se trata de distribuciones de clases desequilibradas. Esta métrica es útil para comparar el rendimiento de diferentes modelos.

$$F1 - Score = 2 \times \frac{(Precision \times Recall)}{(Precision + Recall)} \quad \textbf{(4)}$$

4. Metodología

El objetivo de este trabajo de investigación fue desarrollar una estrategia de análisis basada en espectrometría de masas MALDI- TOF y modelos de aprendizaje *Machine Learning* para el diagnóstico de la enfermedad de Chagas. Para lograr este objetivo, se siguió de manera general la siguiente metodología:

Figura 10 Diagrama de flujo de la metodología usada en el proyecto de investigación.



4.1 Materiales y reactivos

Para el desarrollo de este trabajo todas las soluciones fueron preparadas en agua desionizada Milli-Q (18,2 MΩ.cm) y se emplearon urea grado analítico, Tris-HCl, acetonitrilo (ACN) grado HPLC, ácido trifluoroacético (TFA) y los filtros amicon de ultracentrifugación de 3kDa fueron adquiridos en Merck Millipore. La tripsina grado secuenciación, el ditiotreitol (DTT) grado análisis y la iodoacetamida (IAA) fueron comprados a Sigma-Aldrich. El kit de ensayo de proteínas Pierce BCA fue obtenido de Thermo Fisher Bioreagents.

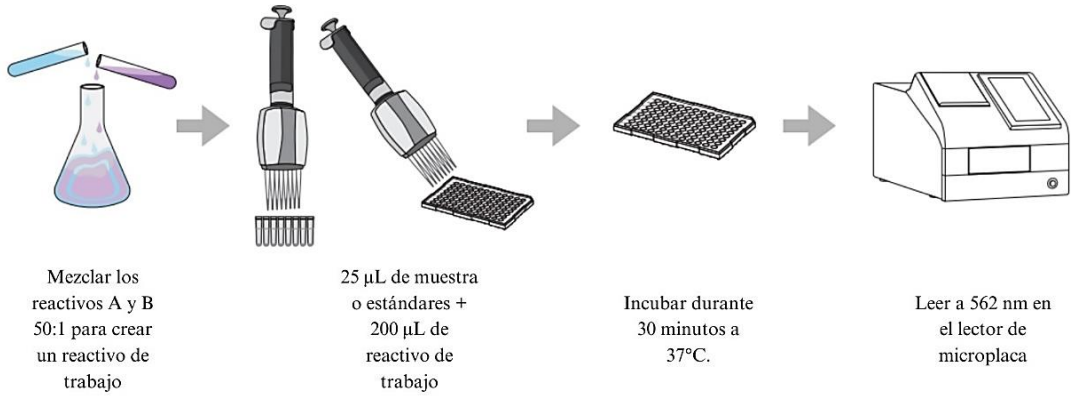
4.2 Recolección de muestras

Las muestras de sueros sanguíneos fueron recolectadas de habitantes de Santander sanas y que padecen de manera sintomática o asintomática la enfermedad de Chagas las cuales fueron suministradas por el grupo de Investigación en Inmunología y Epidemiología Molecular (GIEM) de la Universidad Industrial de Santander.

4.3 Determinación de la concentración de proteínas

La concentración total de proteínas presentes en cada muestra fue determinada empleando un kit BCA Pierce de Thermo Scientific ™. Este ensayo se basa en la detección colorimétrica de un complejo de cobre con las proteínas presentes en las muestras para su posterior cuantificación por la lectura de su absorbancia a 562 nm como es ilustrado en la Figura 11 (ThermoFisherScientific, 2020).

Figura 11 Resumen del procedimiento



Nota: Tomado y adaptado de (ThermoFisherScientific, 2020)

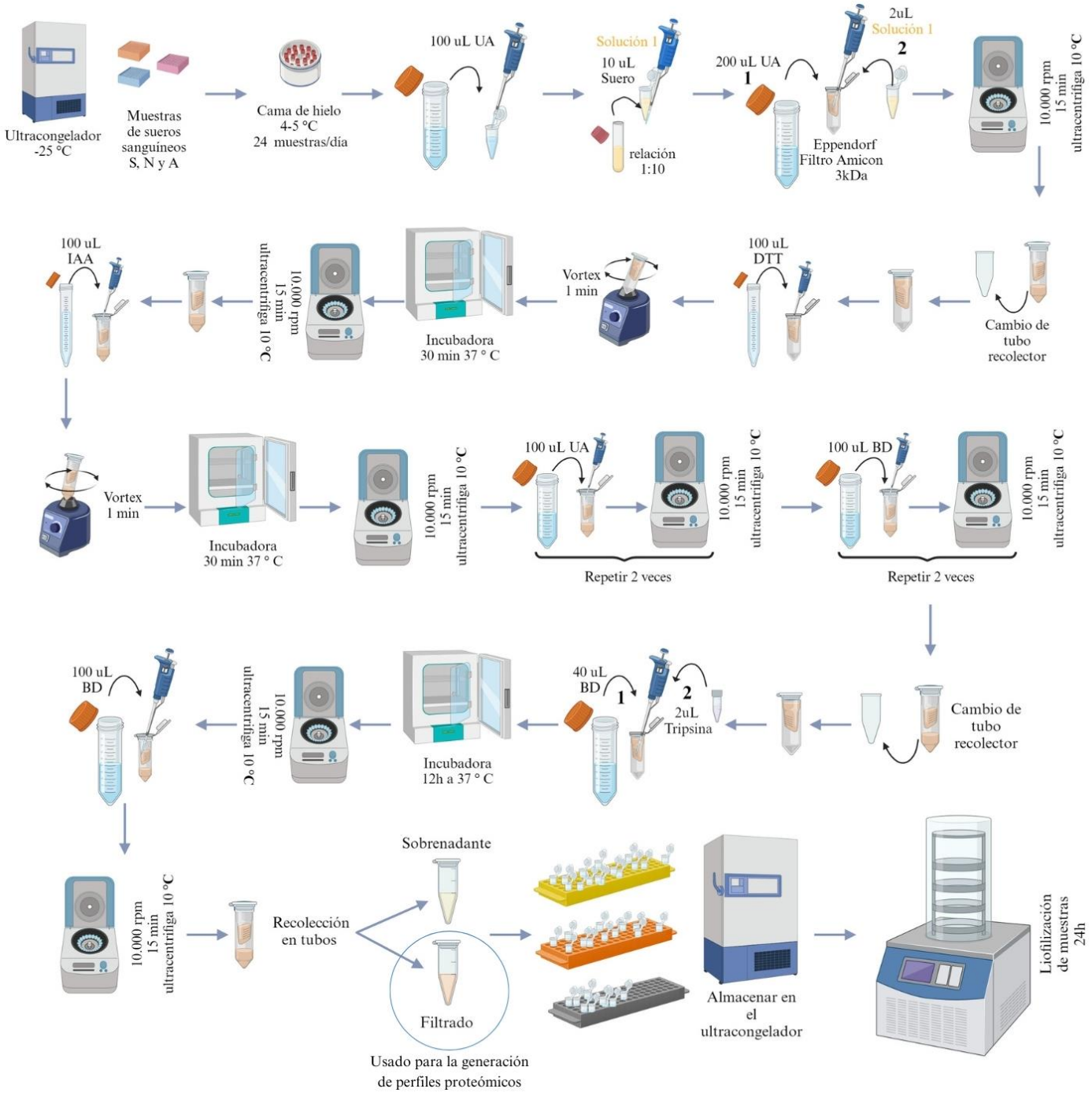
Para determinar la concentración de proteínas en una muestra se realizó la curva de calibración empleando diluciones seriadas de albúmina sérica bovina (BSA) como estándar de referencia. Finalmente, la concentración de proteína total en las muestras se determina mediante regresión lineal R^2:0,996 con un rango dinámico lineal de 0 a 2000 µg/mL.

4.4 Preparación de muestras asistidas por filtro (FASP)

El tratamiento de los sueros sanguíneos se realizó en base a los protocolos establecidos en (Wiśniewski, 2018) este es un método emergente para la preparación de muestras en proteómica. Se basa en la deposición de la muestra de proteínas sobre un filtro de membrana, permitiendo lavados y digestión en el mismo sitio.

Después de haber clasificado los sueros sanguíneos mayores o iguales a 60 mg/mL en cada una de las tres categorías que se tienen seronegativos (N) sintomático (S) y asintomático (A) para proceder a hacer la digestión enzimática de proteínas séricas como en el método (Wiśniewski, 2018) representado en el siguiente diagrama:

Figura 12 Esquema metodológico para la digestión enzimática FASP



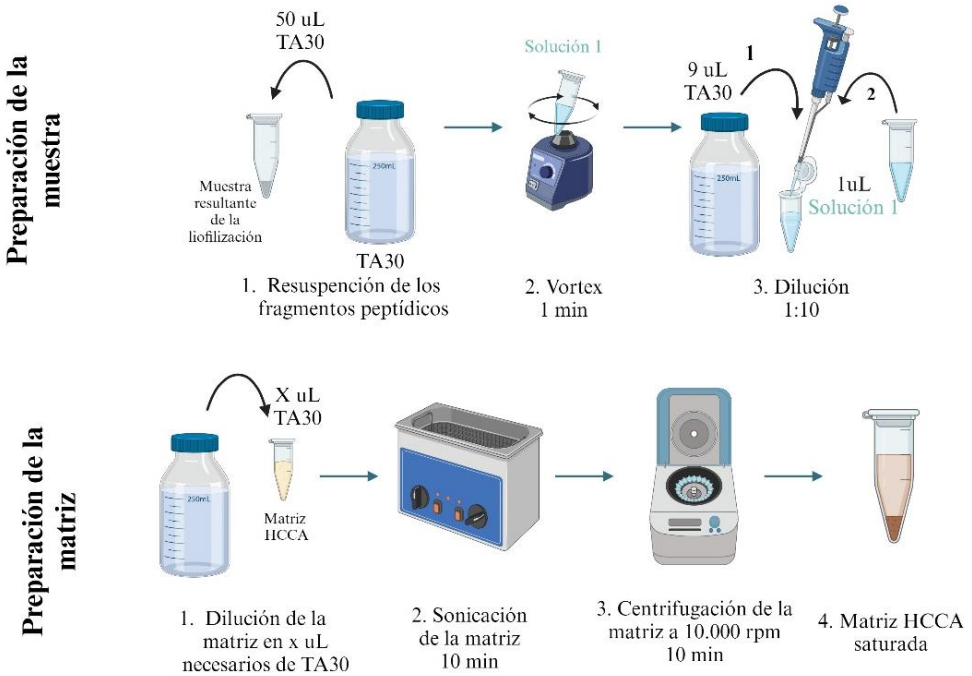
Nota: Creado con Biorender.

4.5 Análisis por espectrometría de masas MALDI-TOF

El análisis por espectrometría de masas MALDI-TOF de las proteínas digestadas, se realizó según el Protocolo propuesto por Bruker Daltonics para el análisis de estas especies químicas (Bruker Daltonics, 2012; Pecks et al., 2010).

Los fragmentos peptídicos en solución, resultantes de la digestión enzimática fueron liofilizados en un equipo Telstar LyoQuest a una temperatura de congelación de -53 ±3°C y vacío de de 0,01 mBar. Una vez liofilizadas las muestras, estas fueron resuspendidas en TA30 (30 ACN:70 Agua; 0.01% TFA) en una una relación de 1:10. La matriz empleada para el análisis fue la HCCA (ácido alfa-ciano-4-hidroxicinámico), la cual fue preparada como una solución saturada (~20 mg/mL) como se detalla en la Figura 13.

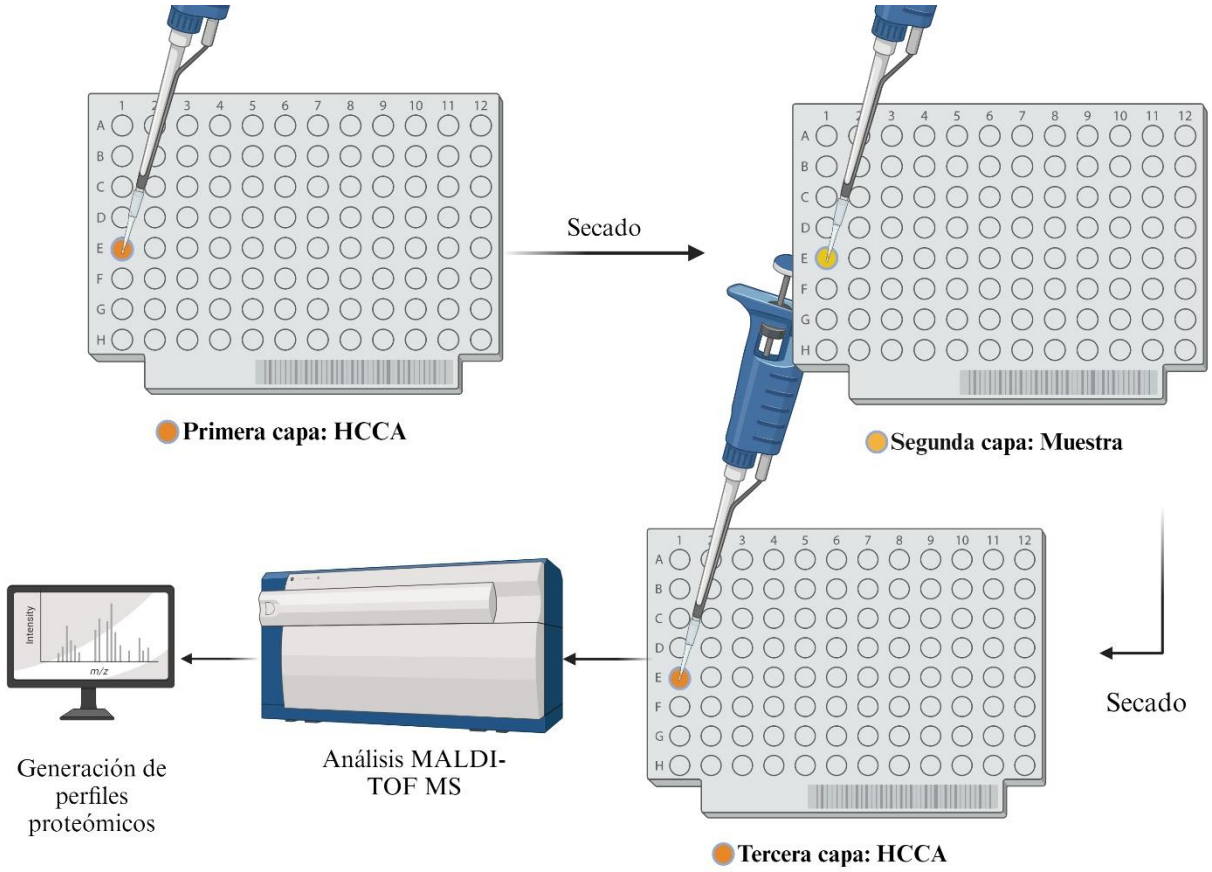
Figura 13 Preparación de muestras y matriz para EM MALDI-TOF



Nota: Fuente del Autor creado con Biorender.

Para la deposición de las muestras en el portamuestras, se usó el método de doble capa; el cual consiste en colocar en cada uno de los ‘spots’ 0,5 µL de matriz HCCA, seguido de 0,5 µL de muestra y finalmente una última capa de 0,5 µL de matriz HCCA dejando secar entre capa y capa a temperatura ambiente como se ilustra en la Figura 14.

Figura 14 Preparación de muestras para el análisis



Nota: Creado con Biorender.

Los espectros de masas fueron adquiridos en modo lineal de iones positivos a una potencia de 80% del láser en un rango de masa de 600 a 6000 kDa. El voltaje de aceleración entre las placas fue de 89kV con una frecuencia de 200 Hz y resolución de 20.000 FWH. Cada espectro

correspondió a la sumatoria de los espectros correspondientes a 10 disparos a la muestra realizados de forma manual.

4.6 Preprocesamiento de datos

Una vez obtenidos todos los espectros de los diferentes grupos de muestras (pacientes sintomáticos, pacientes asintomáticos y pacientes sanos) se realizó un pretratamiento con el fin de eliminar y reducir variaciones no deseadas que pudieran interferir en la creación de los modelos predictivos el cual incluye la corrección de la línea base para eliminar el ruido de fondo, el suavizado de las señales para mejorar la relación señal-ruido y finalmente los datos se exportaron en formato txt desde el software flexAnalysis versión 3.3 de Bruker Daltonics.

4.7 Desarrollo de modelos predictivos mediante aprendizaje supervisado

Para el análisis de datos se entrenaron algoritmos de aprendizaje automático supervisados, comenzando con la creación de conjuntos de datos de calibración y validación. La proporción de datos utilizada para esta separación es de 80:20, donde el 80% de los datos formó parte del conjunto de calibración, utilizado para construir el modelo, mientras que el 20% restante se destinó al conjunto de prueba, empleado para evaluar la capacidad predictiva del modelo

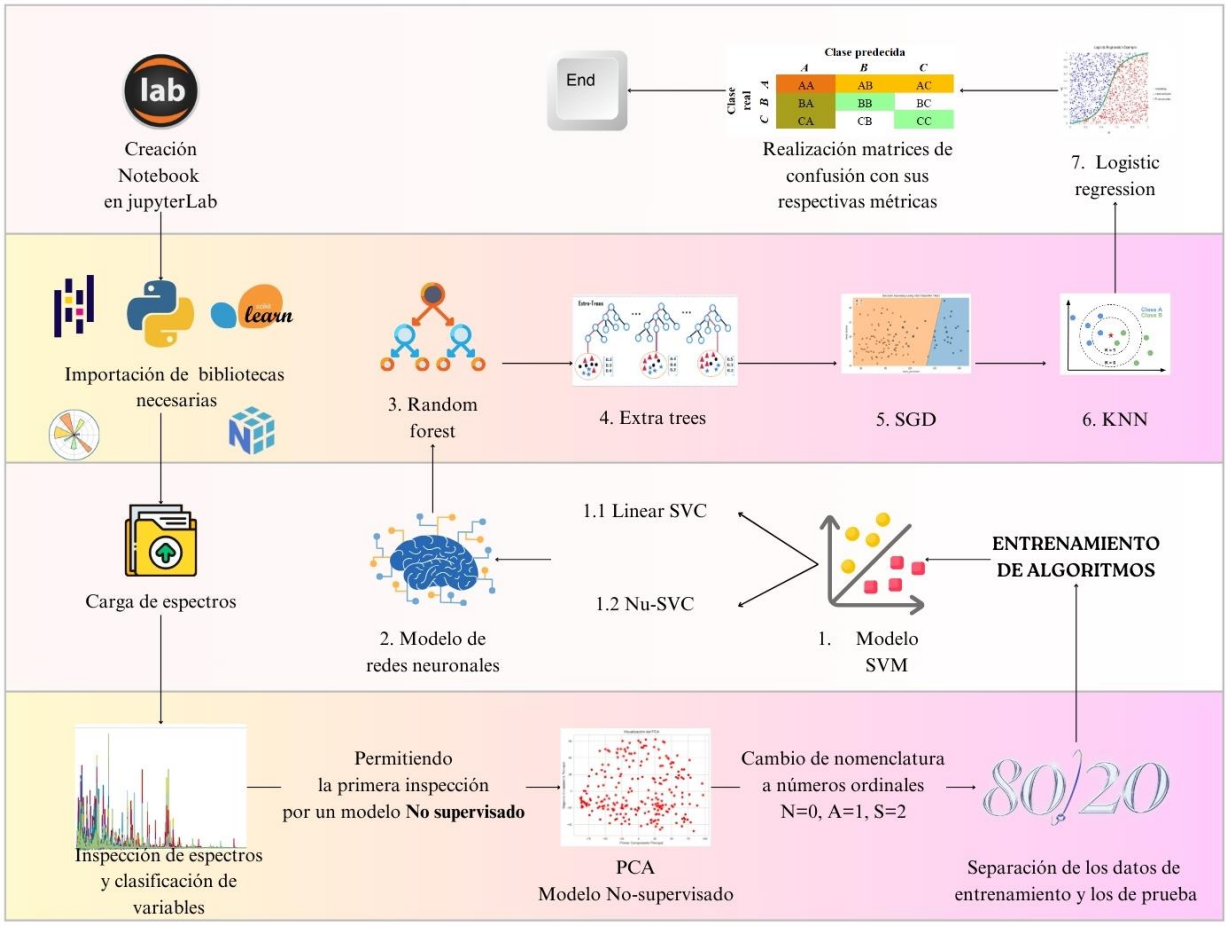
Para facilitar y optimizar el proceso de análisis, se implementó un cuaderno de código abierto en el ambiente de JupyterLab diseñado específicamente para este propósito. Este cuaderno abarca diversas funciones, desde la importación de librerías esenciales hasta la carga de datos, preprocesamiento, análisis estadístico, visualización, selección de modelos, evaluación y

comparación. Este enfoque garantiza una mayor transparencia y accesibilidad en todas las etapas del proceso, facilitando la comprensión y replicación del trabajo realizado.

Se desarrollaron distintos tipos de algoritmos de clasificación para el análisis de los datos, mostrados en la Figura 15, se alimentó el algoritmo con la información separada de cada tipo de clase de los espectros de masas. Se definieron las variables de entrada, los espectros de masa y los valores de salida correspondientes al diagnóstico: 0-Seronegativos, 1-Asintomáticos y 2-Sintomático.

Para la evaluación y optimización del algoritmo, se emplea la Matriz de Confusión o Matriz de Error que es una matriz de 3x3 que se usa para describir el rendimiento de un modelo de clasificación en un conjunto de datos de prueba para los cuales se conocen los valores reales, como se muestra en la siguiente figura:

Figura 15 Entrenamiento de algoritmos para el análisis de espectros de masas.



5. Resultados y discusión

5.1 Determinación de la concentración de proteínas

Con el fin de determinar que sueros sanguíneos eran aptos para la aplicación del protocolo de (Wiśniewski, 2018) se realizó una determinación de concentración de proteínas por medio del kit Pierce™ BCA Protein Assay Kit.

En este ensayo, se llevan a cabo dos reacciones. En la primera, denominada reacción de Biuret, en donde los enlaces peptídicos tienen la capacidad de reducir el ion cúprico (Cu^{2+}) a ion cuproso (Cu^{1+}) en un medio alcalino. Posteriormente, en la segunda etapa, la interacción del ion cuproso con dos moléculas de BCA forma otro complejo aún más estable, el cual es medido por espectroscopia UV-Vis a una longitud de onda de 562 nm. En este sentido, la intensidad de absorbancia resultante es proporcional a la concentración del complejo y por tanto a la cantidad de proteína presente en la muestra (Astrof & Horowitz, 2018).

El rango de concentración de proteínas totales para cada grupo de muestras fue de 8.27 a 190.18 mg/mL. Estos valores se determinaron a partir de la regresión lineal que es presentada en el Apéndice a.

A partir de las concentraciones de proteínas calculadas se seleccionaron todos aquellos sueros sanguíneos con concentración de proteína superior a 60 mg/mL para garantizar la integridad de las muestras y la eficiencia en cada uno de los pasos de pre-tratamiento de las muestras.

5.2 Digestión enzimática de proteínas séricas: método FASP

Durante este proceso, se llevaron a cabo diversas etapas químicas y bioquímicas. Inicialmente, se utilizó una concentración elevada de urea (8 M) para desnaturalizar las proteínas y eliminar el detergente, lo que facilita la contracción y disociación de las micelas de detergente, siendo este paso crucial para lograr una eliminación cuantitativa y rápida del detergente. Posteriormente, las proteínas desnaturalizadas fueron sometidas a modificaciones químicas, principalmente carboamidometilación de residuos de cisteína con yodoacetamida. Esta modificación se realiza después de la desnaturalización con urea, permitiendo que las proteínas adopten conformaciones no nativas y contribuyendo a mantener una estructura más estable, lo que es esencial para la siguiente etapa.

La tripsina, una enzima proteolítica, desempeña un papel central en el método FASP. En la etapa de digestión enzimática, la tripsina realiza cortes específicos de los enlaces peptídicos C-terminal de los aminoácidos lisina y arginina. Este proceso tiene lugar después de que las proteínas han sido desnaturalizadas, modificadas químicamente y liberadas del detergente. La digestión se llevó a cabo mediante una incubación de 12 horas en el amicon de 3kDa, lo que asegura la pureza del digesto y permite retener fragmentos no digeridos y otras macromoléculas presentes en el lisado, como ácidos nucleicos y oligosacáridos.

Los fragmentos de estudio fueron aquellos péptidos pequeños que pudieron atravesar de la membrana de ultrafiltración de 3kDa. Estos fragmentos fueron extraídos con una micropipeta y puestos en tubos eppendorf limpios.

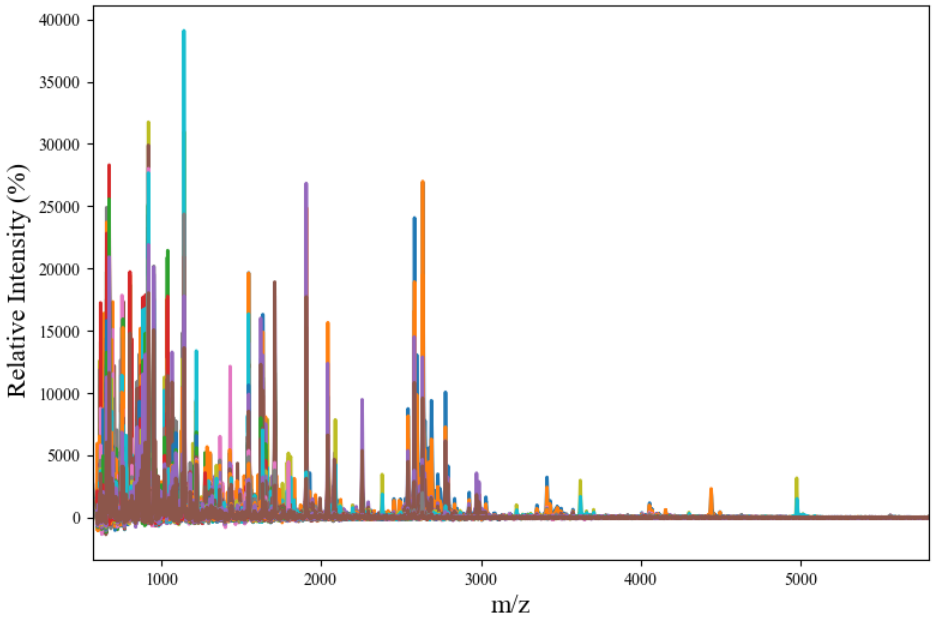
5.3 Análisis por espectrometría de masas MALDI-TOF

Las muestras resultantes se sometieron a liofilización según las condiciones descritas en la sección 4.5. Este proceso implicó congelar la muestra y luego eliminar el agua mediante sublimación, sin pasar por el estado líquido. La repercusión en este paso es evaporar el agua, junto con algunos solutos y sales que podrían afectar el análisis por espectrometría de masas MALDI-TOF.

Un conjunto completo de 236 muestras, distribuidas en 104 sueros sintomáticos, 64 sueros negativos y 68 sueros asintomáticos, fue sometido a un minucioso análisis mediante la tecnología de MALDI-TOF MS. El proceso se reflejó en la obtención de espectros de alta calidad para todas las muestras examinadas, los cuales fueron claramente visualizados y evaluados tanto mediante el software FlexAnalysis v.3.3 cómo a través de Python 3.

El procesamiento de los espectros de masa según el protocolo detallado en la sección 4.6, arrojó como resultado el conjunto de espectros como se presenta en la Figura 16. Este enfoque no solo ayudó a la limpieza de los espectros, sino que también contribuyó significativamente al aumento de la resolución y discriminación de las características espectrales, facilitando así una interpretación más precisa y detallada de los datos.

Figura 16 Espectros de masa de la totalidad de las muestras.



La inspección visual de los espectros mostrados en las Figuras Figura 17-Figura 19 los cuales corresponden a los tres grupos de muestras estudiados, los cuales evidencian semejanzas significativas entre sus diversas clases. Esto puede deberse a que todos los espectros fueron adquiridos bajo las mismas condiciones instrumentales y operacionales descritas en las secciones 4.3-4.5.

Figura 17 Muestras totales y promedio de la clasificación Asintomática.

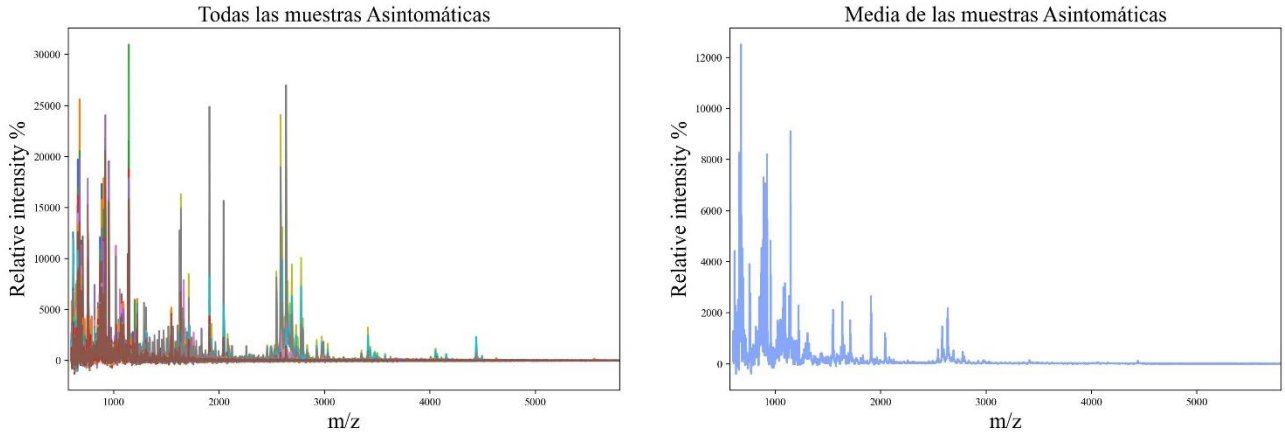


Figura 18 Muestras totales y promedio de la clasificación Seronegativa.

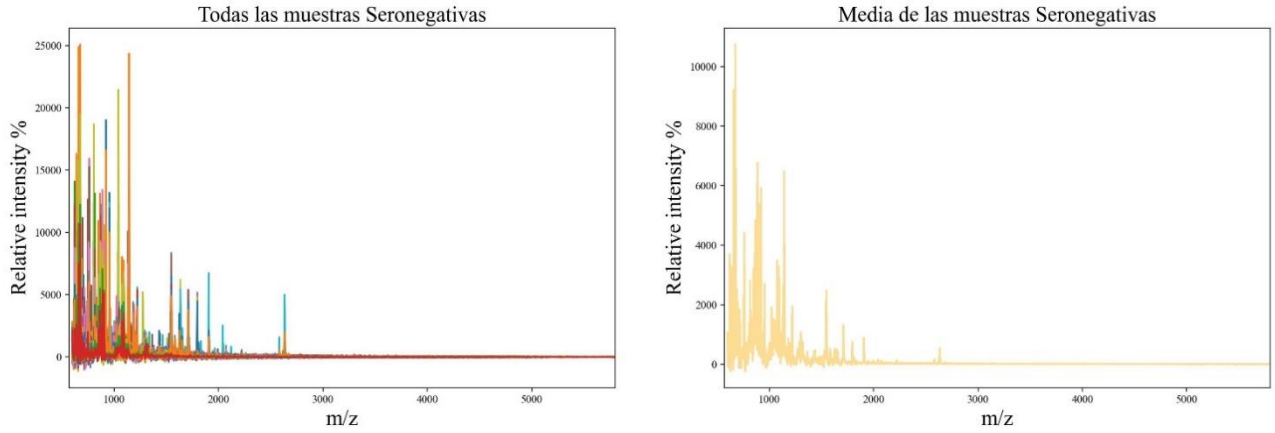
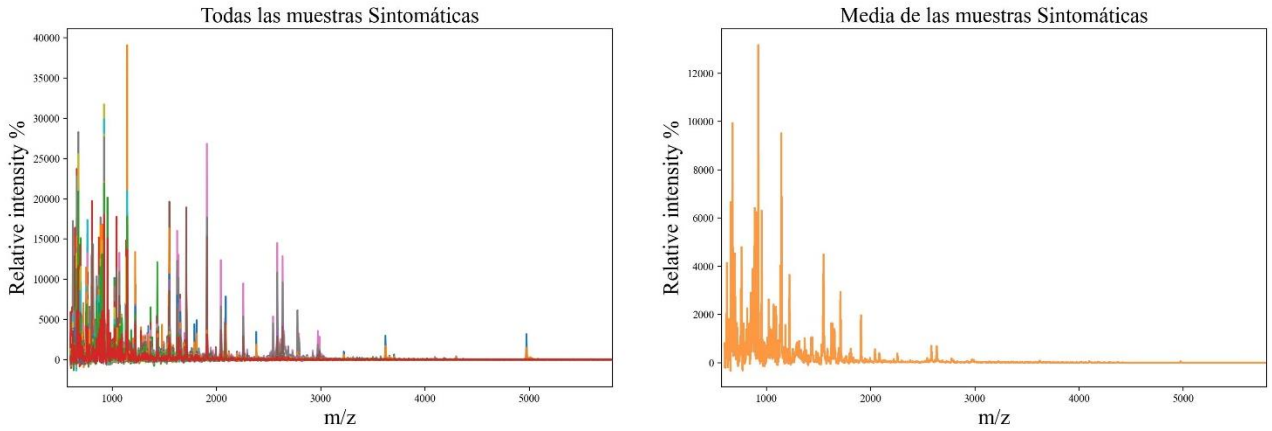


Figura 19 Muestras totales y promedio de la clasificación Sintomática.



Asimismo, la presencia de señales en un rango de masas de 0 a 1000 m/z, correspondientes a la matriz HCCA de la Figura 20 están presentes en todos los espectros de los grupos de muestras como puede ser observado en la Figura 21 Superposición de muestras de clasificación Asintomática, Sintomática y Seronegativa..

Figura 20 Espectro de la Matriz de MALDI: HCCA.

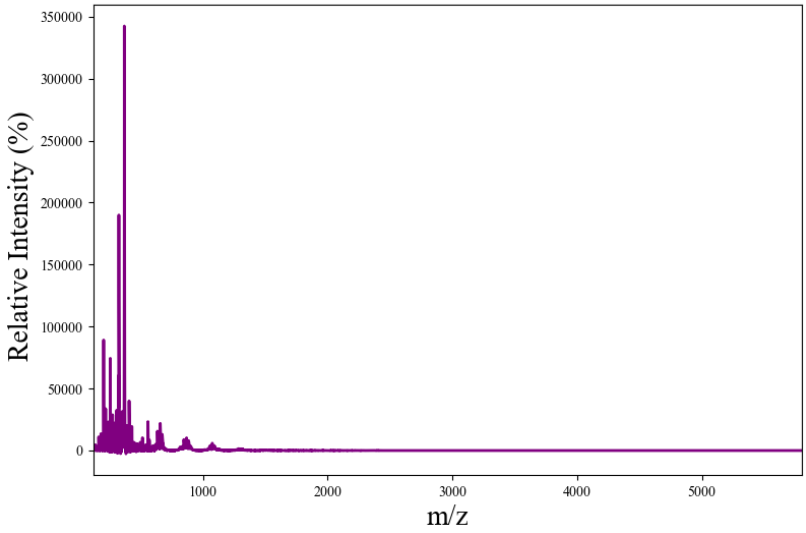
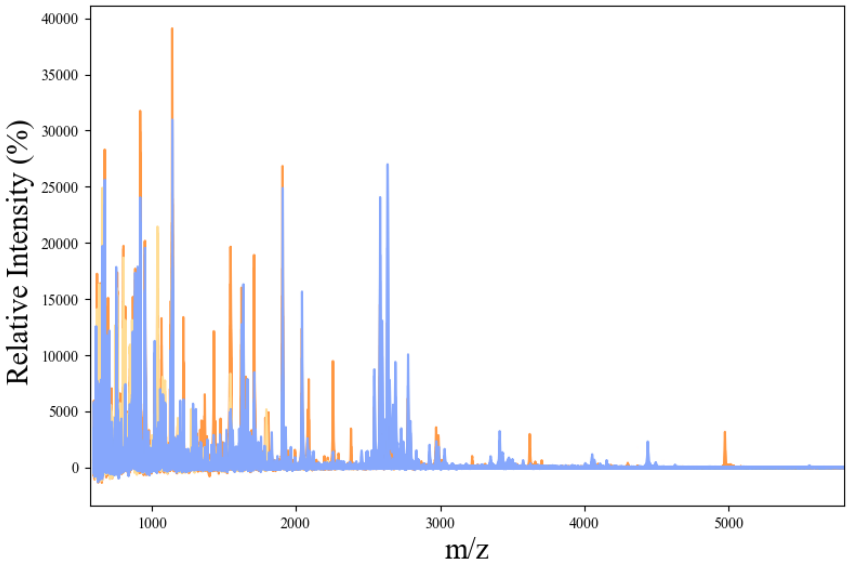


Figura 21 Superposición de muestras de clasificación Asintomática, Sintomática y Seronegativa.



Nota: El espectro de masas promedio de color azul pertenece a la clasificación Asintomática, el de color naranja pertenece a la clasificación Sintomática y, el de color amarillo pertenece a la clasificación Seronegativa.

Dadas las similitudes de clase mencionadas, es crucial establecer criterios de diferenciación para cada una de ellas. La inspección visual no ofrece una respuesta definitiva y clara. Por lo tanto, en la siguiente sección, se delimitan las características específicas que ayudan a identificar patrones a lo largo de todas las muestras. Esto permite una discriminación más precisa y objetiva entre las clases.

5.4 Formulación de los modelos predictivos mediante aprendizaje supervisado en ML

Los espectros obtenidos del preprocesamiento de datos se trataron utilizando el lenguaje de programación de código libre Python (versión 3) implementando un cuaderno en el ambiente 'Jupyter Lab' utilizando el software Anaconda.

Seleccionar un modelo que se ajuste a las necesidades específicas del problema requiere una comprensión profunda del tipo de datos disponibles y una definición clara del objetivo a abordar. En este proyecto, los datos disponibles fueron apropiados para desarrollar modelos de clasificación usando aprendizaje supervisado. La finalidad es que el algoritmo aprenda patrones similares, principalmente para asignar etiquetas que indiquen si una persona presenta la enfermedad de Chagas de manera sintomática, asintomática o si es negativa a la enfermedad.

De este modo, se obtuvieron nueve modelos clasificadores distintos para diagnosticar la enfermedad de Chagas en las muestras tratadas. Estos modelos fueron: Linear SVC (Clasificador SVC lineal); SGD Classifier (Clasificador de descenso de gradiente estocástico); SVM (Máquina de vectores de soporte); NuSVC (Máquina de vectores de soporte con núcleo Nu); K-Neighbors (Vecinos más cercanos); ExtraTrees (Arboles extra); Random Forest (Bosque aleatorio); MLP (Red neuronal multicapa); Logistic Regression (Regresión logística).

Antes de iniciar el proceso de aprendizaje supervisado, se realizó una inspección detallada del conjunto de datos con el propósito de evaluar la variabilidad presente en cada clase. Para lograr este objetivo se realiza un análisis de componentes principales (aprendizaje no supervisado) en el cual busca transformar las variables originales en un conjunto nuevo y no correlacionado de variables, llamadas componentes principales. Estos componentes se ordenan de tal manera que el primero captura la máxima variabilidad en los datos, el segundo, captura la siguiente mayor cantidad, y así sucesivamente.

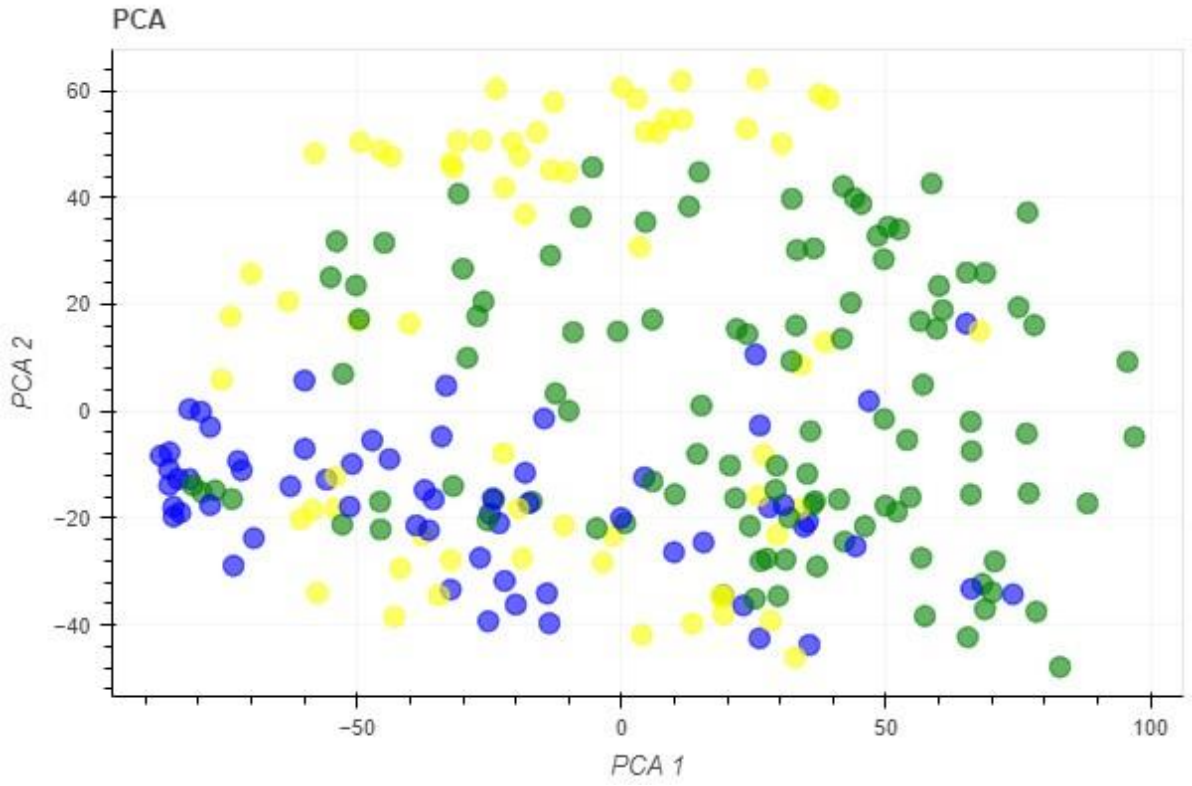
Para evaluar la variabilidad en este conjunto de datos, se realizaron cálculos de componentes principales mediante el modelo PCA en *scikit Learn*. Luego, se determinó la varianza total en el conjunto de datos original y, finalmente, se calculó la varianza explicada, es decir, una medida que indica cuánta variabilidad total hay en un conjunto de datos cual es explicada por un conjunto específico de variables. En este caso, en la Tabla 3 Valores de varianza en el modelo de PCA, se reportan el conjunto de datos de componentes principales obtenidos.

Tabla 3 Valores de varianza en el modelo de PCA

PC	Varianza	Varianza Explicada
PC1	0.286674	0.287
PC2	0.110237	0.397
PC3	0.081763	0.479
PC4	0.074810	0.554
PC5	0.065494	0.619
PC6	0.044530	0.664
PC7	0.039043	0.703
PC8	0.036767	0.740
PC9	0.023298	0.763
PC10 (185 datos)	0.020652	0.784

Estos datos representan las combinaciones lineales de las variables originales, trazando la mayor variabilidad. Sin embargo, se observa que a medida que la varianza explicada aumenta, la variabilidad en los datos también incrementa. Esta observación se debe a que los componentes principales están diseñados para captar la máxima variabilidad presente en los datos. Esta consideración se refleja en el gráfico de PCA (Figura 22) donde la variabilidad es tan amplia que las clases no son fácilmente distinguibles. Por ende, para abordar esta complejidad, fue necesario entrenar el algoritmo de manera controlada utilizando conjuntos de datos entrenamiento y prueba.

Figura 22 PCA de la totalidad de muestras.

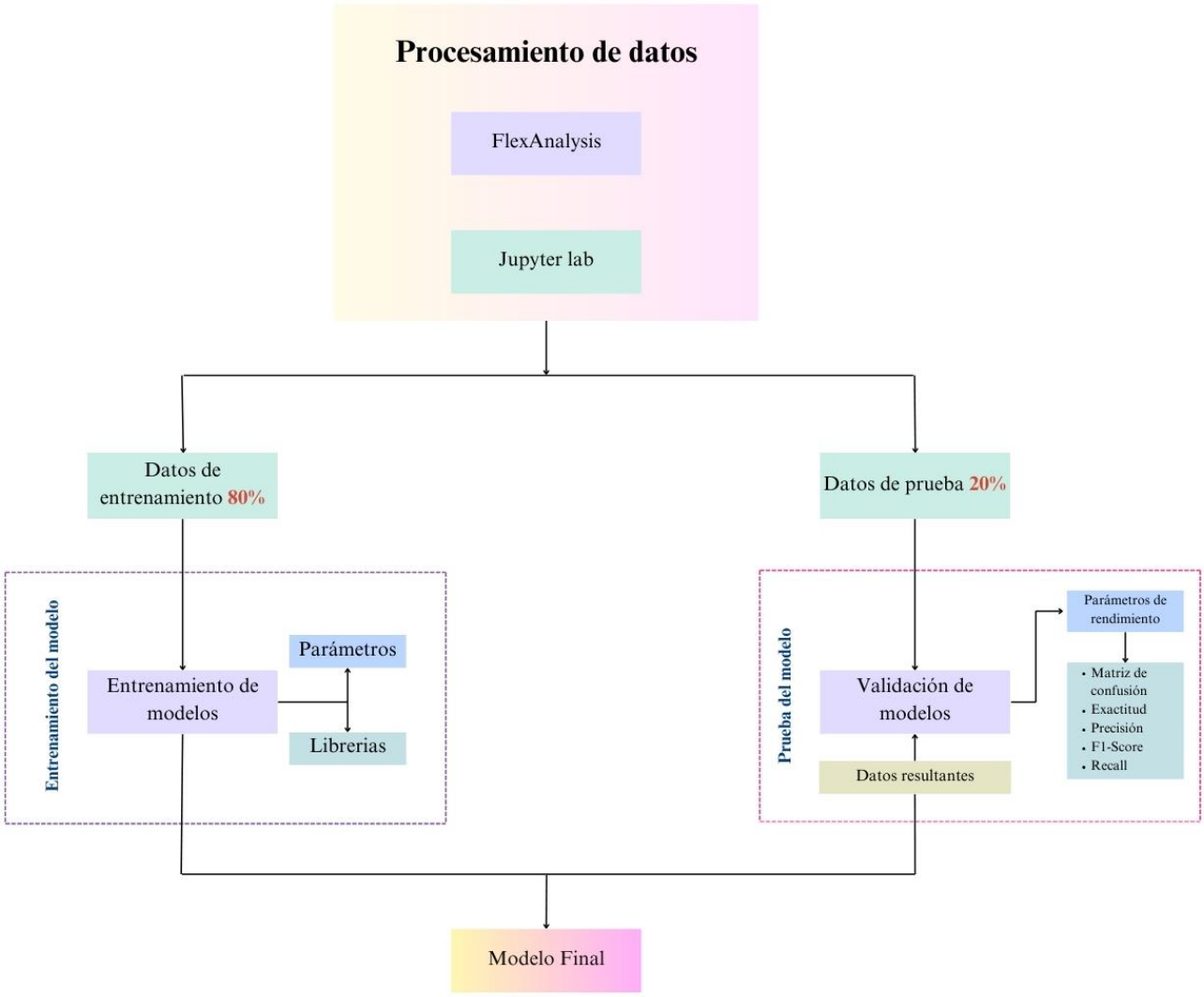


Nota: El color amarillo representa la clasificación asintomática, el azul los seronegativos y el verde, sintomáticos.

5.4.1 Modelos predictivos

Para la preparación de los modelos se dividieron los datos en una proporción de 80/20, como lo muestra en la Figura 23 y más a detalle en la Tabla 4.

Figura 23 Preparación del modelo



El cuaderno con cada uno de los modelos entrenados está disponible en GitHub y en el Apéndice b.

Tabla 4 El número de objetos por clases utilizados en los conjuntos de entrenamiento y prueba.

Clase	Entrenamiento	Prueba	Total
Asintomático	54	14	68
Seronegativos	50	13	64
Sintomático	84	21	105
Total	195	48	236

Para aprender a identificar los patrones de los 195 espectros de prueba, el entrenamiento se empieza por etapas:

1. Inicialización: En esta etapa, se inicializan los parámetros del modelo. Los parámetros son las variables dependiendo de qué tipo de algoritmo sea.
2. Etapa de entrenamiento: En esta etapa, el modelo se alimenta con los datos de entrenamiento y a aplicar sus parámetros a cada uno de los datos de entrenamiento.

Después de haberse aplicado los pasos anteriores, empieza la etapa de evaluación del modelo en el que se evalúan el total de 41 espectros que no se utilizaron para entrenar el modelo.

5.4.1.1 Máquina de vectores de soporte (SVM)

Para el modelo máquinas de soportes vectoriales se utilizaron los siguientes parámetros:

- C=40 Este valor indica la regularización de los hiperplanos L1 y L2. Un número mayor en C resulta en la restricción de los parámetros del optimizador SMO y esto

se traduce en una menor sensibilidad al ruido por lo que son menos propensos a ser desviados por los puntos de datos ruidosos.

- Kernel=”rbf” Este es el parámetro que controla el vector de soporte, en este caso se escogió el Radial Basis Function que permite hacer un mapeo de los datos de manera no lineal, por lo que fue el que más se ajustó a la tendencia de los datos.
- Random state=123 Se estableció este número constante para que el modelo se inicialice de la misma manera cada vez que se ejecute el código y evitar resultados diferentes cada que se entrene el modelo.

Como resultado de la aplicación de los parámetros se obtienen los siguientes resultados:

Figura 24 Matriz de confusión SVM.

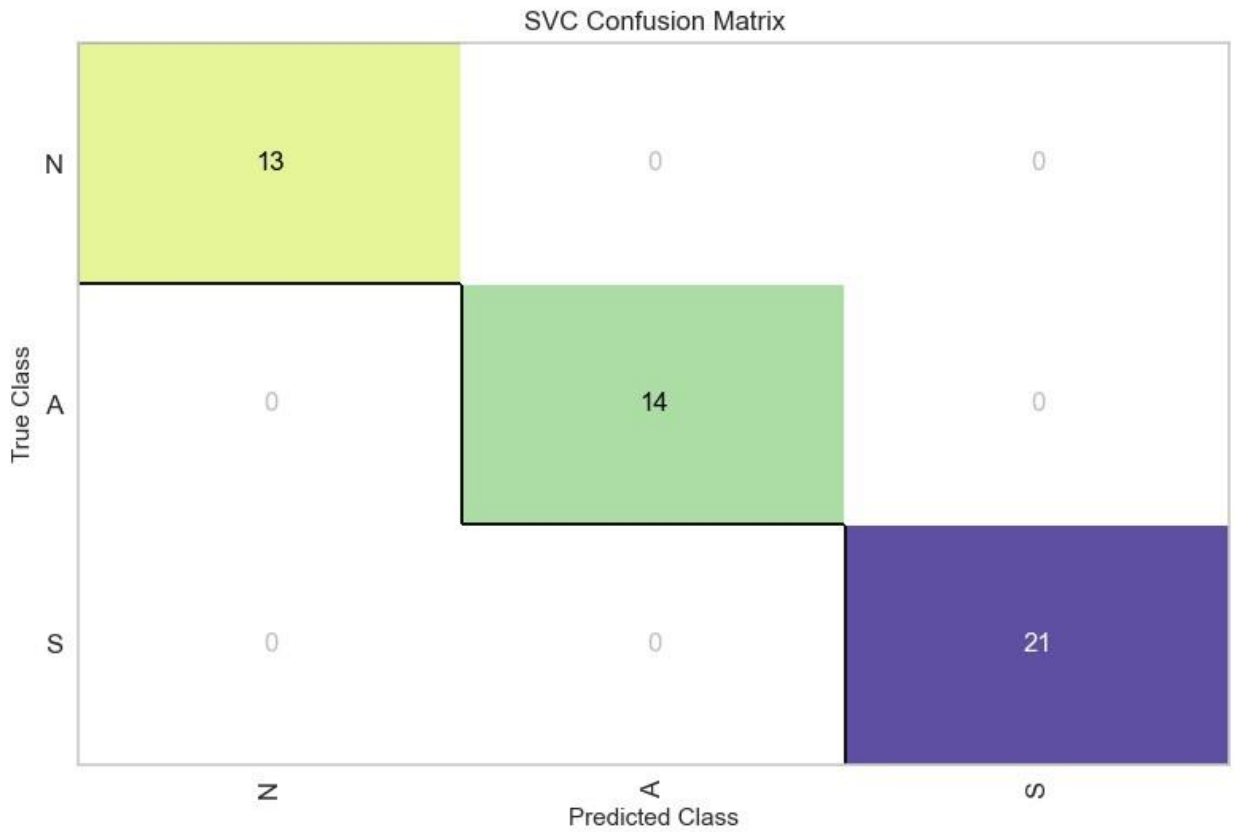


Tabla 5 Métrica de resultados para las Maquinas de soporte vectorial

	Precisión	Sensibilidad	Puntuación F1	Soporte
0.0	1,0000	1,0000	1,0000	13
1.0	1,0000	1,0000	1,0000	14
2.0	1,0000	1,0000	1,0000	21
Exactitud			1,0000	48
Promedio macro	1,0000	1,0000	1,0000	48
Promedio ponderado	1,0000	1,0000	1,0000	48

El modelo obtenido no obtuvo ningún error en las predicciones y obtuvo una predicción del 100%.

5.4.1.2 Máquina de vectores de soporte con núcleo Nu (NuSVC)

Para el modelo derivado de SVM, NuSVC, se deben explicar las diferencias para entender el porqué de los parámetros usados. En lugar de utilizar hiperplanos como divisor entre las clases, NuSVC utiliza una hiperesfera centrada en el origen. La medida de distancia entre ejemplos en NuSVC es proporcional al cuadrado de la distancia entre los puntos en el espacio de características.

se utilizaron los siguientes parámetros:

- Nu=0.1. Este es un parámetro no lineal que permite la existencia de errores al clasificar ejemplos. El valor de 0.1 tomado significa que el modelo permitirá una cantidad moderada de errores en los datos de entrenamiento para evitar sobre ajustarse a ellos.

- Gamma=Auto. Este parámetro ajusta la distancia mínima entre los ejemplos positivos y negativos en lugar de controlar la "curvatura" de la función RBF como lo haría en SVC.

A continuación, se presenta una tabla que resume las métricas obtenidas para una mejor comprensión y análisis de los resultados.

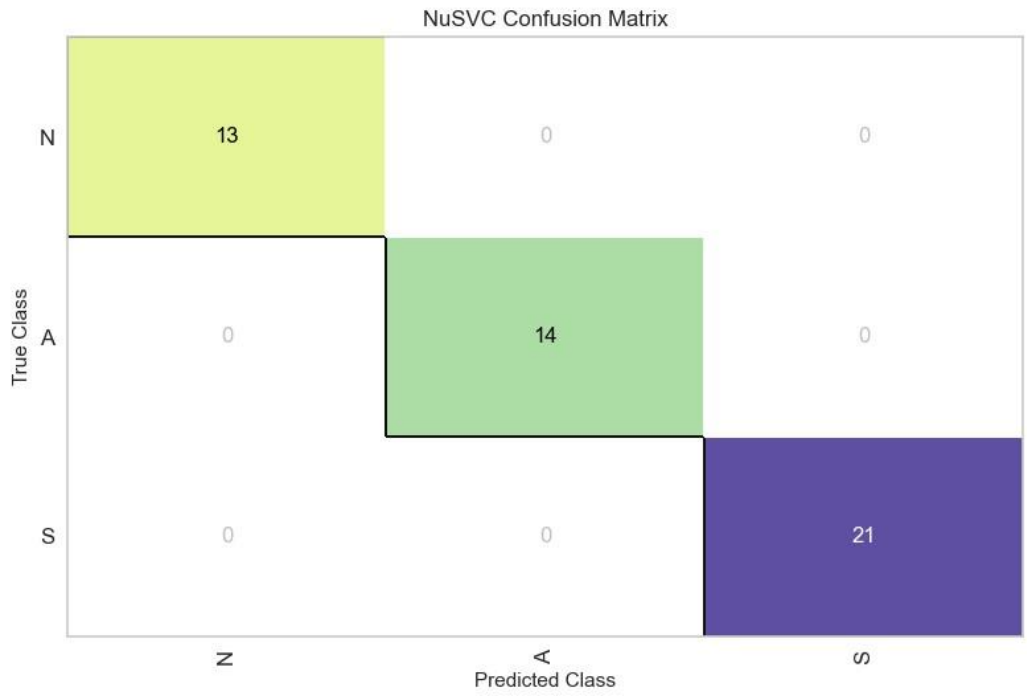
Tabla 6 Métricas de resultados NuSVM

	Precisión	**Sensibilidad**	**Puntuación F1**	**Soporte**
0.0	1,0000	1,0000	1,0000	13
1.0	1,0000	1,0000	1,0000	14
2.0	1,0000	1,0000	1,0000	21
Exactitud			1,0000	48
Promedio macro	1,0000	1,0000	1,0000	48
Promedio ponderado	1,0000	1,0000	1,0000	48

El modelo obtenido al igual que su predecesor, SVC, no obtuvo ningún error en las predicciones y obtuvo una predicción del 100%. Cabe recalcar que NuSVC es un modelo más sensible a los valores iniciales de los parámetros y que un ajuste incorrecto puede llevar a un modelo inadecuado, por lo que se buscó por prueba y error sus valores óptimos.

Al igual que las métricas obtenidas para el modelo derivado de SVM, NuSVC, es relevante destacar que la matriz de confusión mostrada en la Figura 25 no se reportaron errores en las predicciones, lo cual refleja una elevada precisión en el modelo.

Figura 25 Matriz de confusión NuSVC.



5.4.1.3 Clasificador SVC lineal (LinearSVC)

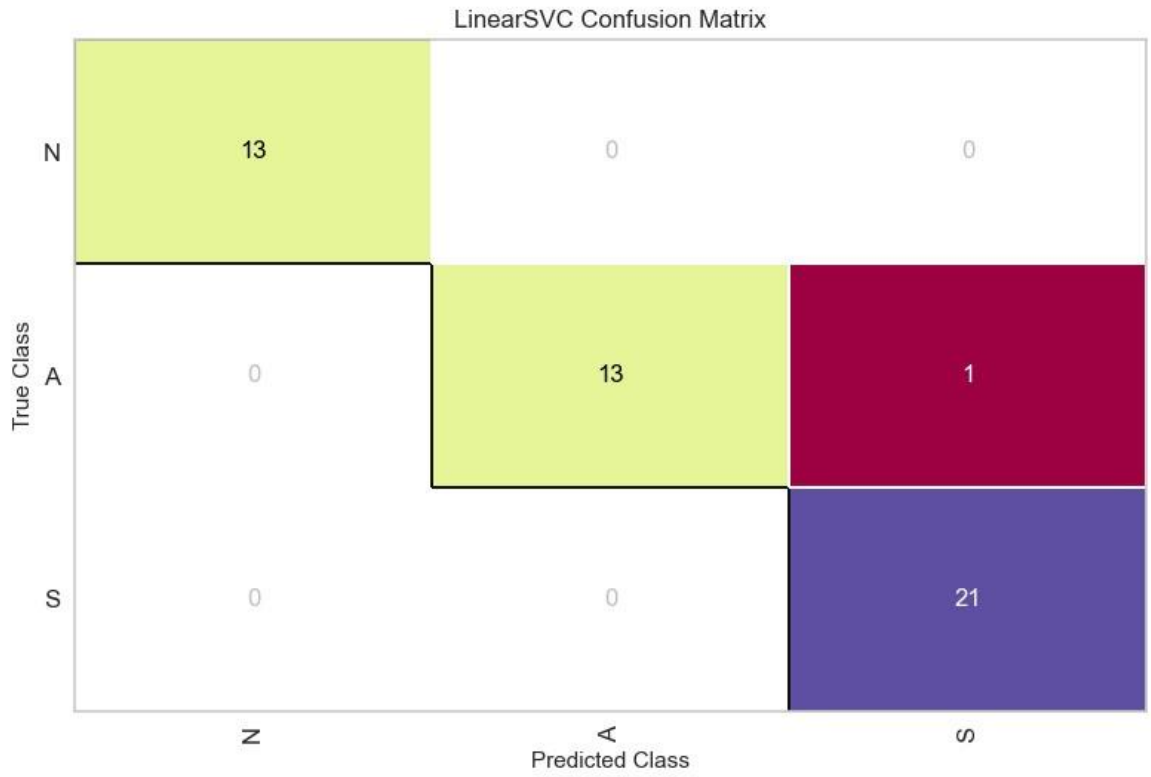
Para el modelo derivado de SVM, LinearSVC, se han utilizado los valores predeterminados para cada parámetro, según la documentación oficial de *SciKit-Learn*, dado que los parámetros están vacíos en este caso. A continuación, se presentan las métricas resultantes con las configuraciones predeterminadas del modelo:

Tabla 7 Métricas LinearSVC

	Precisión	**Sensibilidad**	**Puntuación F1**	**Soporte**
0.0	1,0000	1,0000	1,0000	13
1.0	1,0000	0,9286	0,9630	14
2.0	0,9545	1,0000	0,9767	21
Exactitud			0,9792	48
Promedio macro	0,9848	0,9762	0,9799	48
Promedio ponderado	0,9801	0,9792	0,9790	48

El modelo LinearSVC alcanzó una exactitud del 97.92%, lo que indica que clasificó correctamente la gran mayoría de los datos de entrenamiento. Sin embargo, al examinar la matriz de confusión Figura 26, se observa un falso positivo en la clasificación de los sintomáticos y un falso negativo en la clasificación de los asintomáticos. Estos errores revelan que el modelo tiene dificultades para distinguir entre las clases similares, que en este caso son ambas portadoras de la enfermedad de Chagas. Es crucial abordar estas deficiencias para mejorar la precisión del modelo en futuras predicciones.

Figura 26 Matriz de confusión LinearSVC.



5.4.1.4 Redes Neuronales: MLP

Para este modelo clasificador de redes neuronales multicapa (MLP) los parámetros usados en el código fueron:

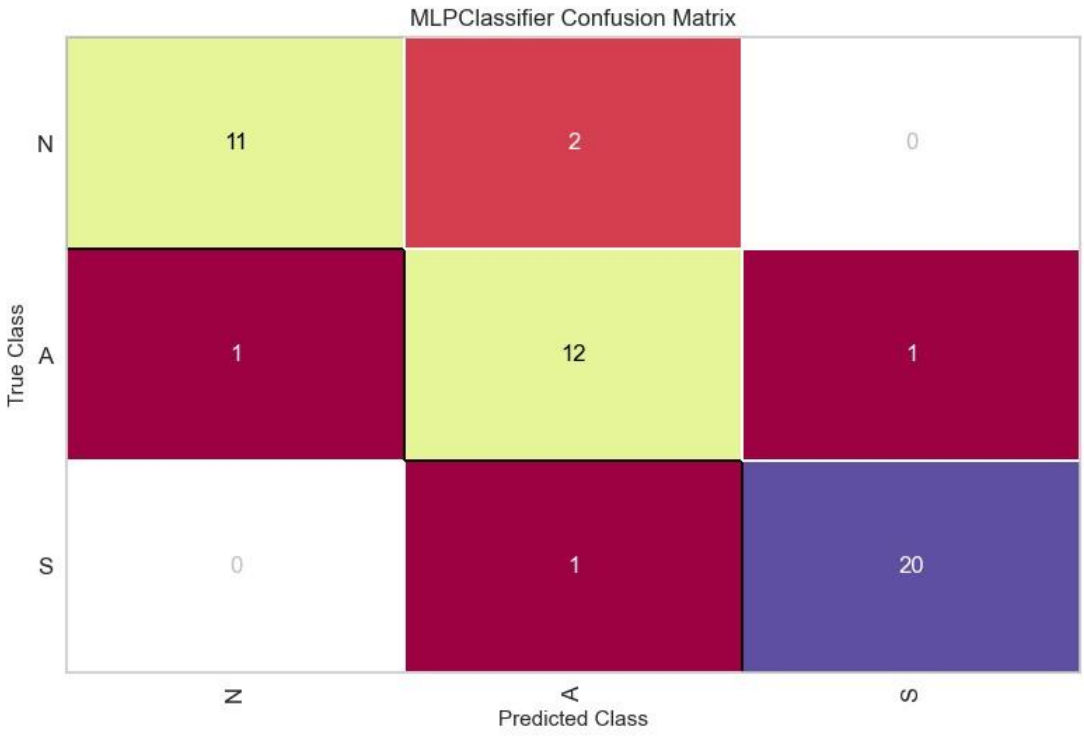
- hidden_layer_sizes= (20,20): Representan las dimensiones de las dos capas ocultas en la red neuronal. En este caso, ambas capas ocultas tienen 20 neuronas cada una.
- learning_rate_init=0.01: Esta es la tasa de aprendizaje inicial que el optimizador utiliza para actualizar la red neuronal. En este caso, el valor de tasa de aprendizaje inicial es 0.01.
- solver='lbfgs': Esto determina el algoritmo de optimización que se utilizará para calcular la pérdida de la red neuronal. En este caso, se utiliza el algoritmo 'lbfgs' (A Broyden–Fletcher–Goldfarb–Shanno) para resolver la pérdida no convexa.
- max_iter=5000: Es el número máximo de iteraciones que el algoritmo (lbfgs) de optimización ejecutará.
- random_state=123: Esto es un entero utilizado como generador de números aleatorios.

Tabla 8 Métricas de las redes neuronales: MLPClassifier

	Precisión	Sensibilidad	Puntuación F1	Soporte
0.0	0,9167	0,8462	0,8800	13
1.0	0,8000	0,8571	0,8276	14
2.0	0,9524	0,9524	0,9524	21
Exactitud			0,8958	48
Promedio macro	0,8897	0,8852	0,8867	48
Promedio ponderado	0,8983	0,8958	0,8964	48

En los resultados de las métricas del modelo de redes neuronales se denota un descenso en la exactitud del modelo, predijo el 89,58% de los espectros de prueba asignados. Sin embargo, también se observa que no sólo bajo la exactitud en las métricas sino en la matriz de confusión también:

Figura 27 Matriz de confusión Redes neuronales con MLPClassifier.



El modelo predijo erróneamente por clase:

- Seronegativos: Lograron únicamente 11 predicciones acertadas, cometiendo 2 errores de falsos negativos al confundir a una persona con la enfermedad como si no la tuviera, y a otra persona sin la enfermedad como si la tuviera.
- Asintomáticos: En las predicciones del modelo, se observa que solo 12 de las 14 personas han sido diagnosticadas correctamente. Se presentan dos casos de falsos negativos, donde el diagnóstico para uno de ellos indica ausencia de la enfermedad de manera asintomática en lugar de sintomática, mientras que en el otro caso clasifica a una persona infectada como sana.
- Sintomáticos: En esta categorización de los 21 pacientes sintomáticos con la enfermedad de Chagas, el modelo acierta en la predicción de 20 de ellos. Sin embargo, se registra un falso negativo al confundir a un paciente sintomático como asintomático, y un falso positivo al clasificar erróneamente a un paciente asintomático como sintomático

La red neuronal por clasificador MLP tiene algunos problemas para distinguir entre las clases sintomáticas y asintomáticas.

5.4.1.5 Bosque aleatorio (*Random Forest*)

Para este modelo solo se escogió el parámetro “random_state=90” en la que funciona igual que en el resto de los modelos, controlando la generación de números aleatorios.

Figura 28 Matriz de confusión Random Forest.

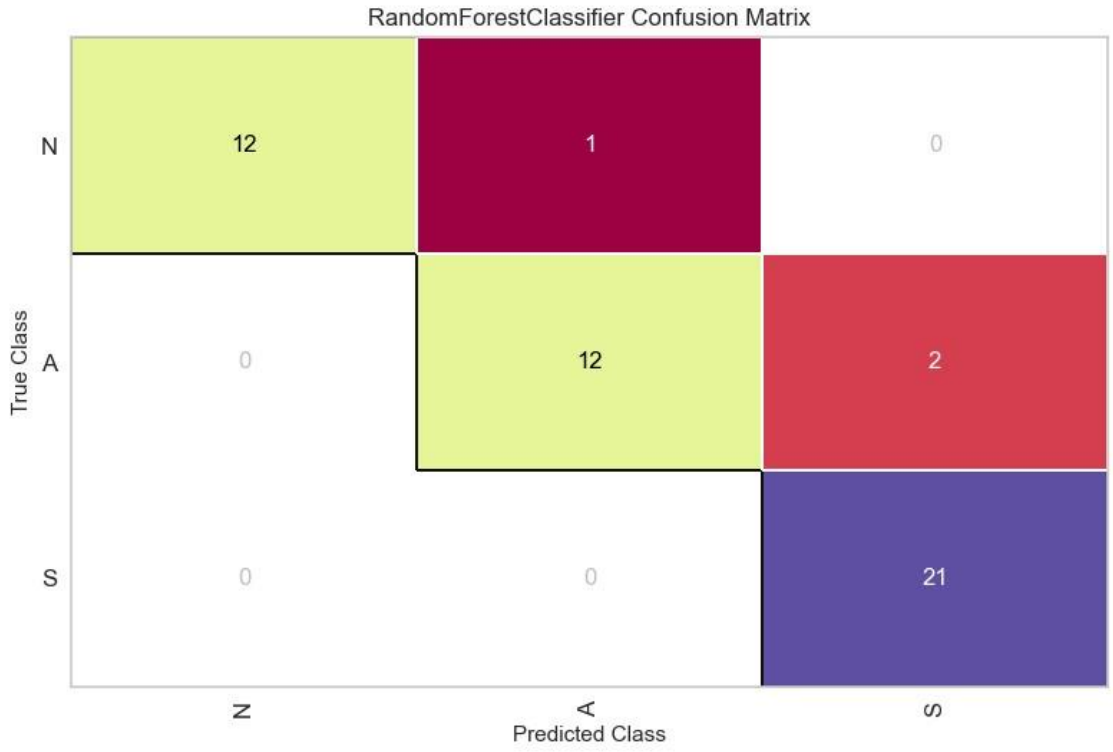


Tabla 9 Métricas *Random Forest*

	Precisión	**Sensibilidad**	**Puntuación F1**	**Soporte**
0.0	1,0000	0,9231	0,9600	13
1.0	0,9231	0,8571	0,8889	14
2.0	0,9524	0,9524	0,9545	21
Exactitud			0,9375	48
Promedio macro	0,9454	0,9267	0,9345	48
Promedio ponderado	0,9395	0,9375	0,9369	48

De las métricas de desempeño del modelo se concluye que tiene una exactitud promedio de 93.75%, mostrando que tiene un buen desempeño general en la clasificación de las tres clases. Sin embargo, con la matriz de confusión se muestra donde falló el modelo:

- Seronegativos: En esta clasificación, se lograron 12 de 13 predicciones precisas, con un único error al identificar erróneamente a una persona sana como enferma, generando un falso negativo
- Asintomáticos: En los resultados predichos por el modelo se obtuvo que sólo 12 de las 14 personas están bien diagnosticadas, se tienen dos falsos negativos en el cual el diagnóstico para dos de ellos es que no tiene la enfermedad de manera asintomática sino sintomáticamente.
- Sintomáticos: Se logró un acertado diagnóstico en 21 pacientes, pero también se diagnosticó erróneamente a dos pacientes que tenían la enfermedad de manera asintomática como si fuera sintomática, generando así dos falsos positivos.

El modelo tiene un buen desempeño general, pero tiene dificultades para clasificar correctamente los pacientes con enfermedad asintomática y sintomática.

5.4.1.6 Clasificador de descenso de gradiente estocástico (SDG)

Los parámetros usados en este algoritmo fueron:

- Max_iter=100: Controla el número máximo de iteraciones que se ejecutarán durante el proceso de entrenamiento, cabe recalcar que el objetivo de una iteración es encontrar una combinación de parámetros que minimice el error entre las predicciones del modelo y los valores reales de las características.
- Tol=1e-3: Este parámetro es la tolerancia de criterio de convergencia, es decir, la precisión máxima a la que los parámetros pueden mejorar en cada iteración.

Tabla 10 Métricas SDG.

	Precisión	Sensibilidad	Puntuación F1	Soporte
0.0	1,0000	0,9231	0,9600	13
1.0	0,8750	1,0000	0,9333	14
2.0	1,0000	0,9524	0,9756	21
Exactitud			0,9583	48
Promedio macro	0,9589	0,9585	0,9563	48
Promedio ponderado	0,9635	0,9583	0,9591	48

Las predicciones hechas con el modelo SDG tuvo predicciones precisas ya que su tasa se acerca a 1 y no sólo eso, tuvo un excelente rendimiento no sólo con sus parámetros, también lo muestra su matriz de confusión.

Figura 29 Matriz de confusión SDG.

- Seronegativos: Se lograron 12 predicciones verdaderas positivas, aunque se cometió un único error al identificar incorrectamente a una persona sana como enferma asintomática, generando así un falso negativo.
- Asintomáticos: El modelo acertó en el diagnóstico de las 14 personas con enfermedad de Chagas asintomáticas. No obstante, se generaron dos falsos positivos, uno al clasificar incorrectamente a una persona sana y otro al hacer lo mismo con una persona enferma sintomática.
- Sintomáticos: Se logró el acierto de 20 verdaderos positivos para esta clasificación, en la que el falso negativo fue predicho como un sintomático.

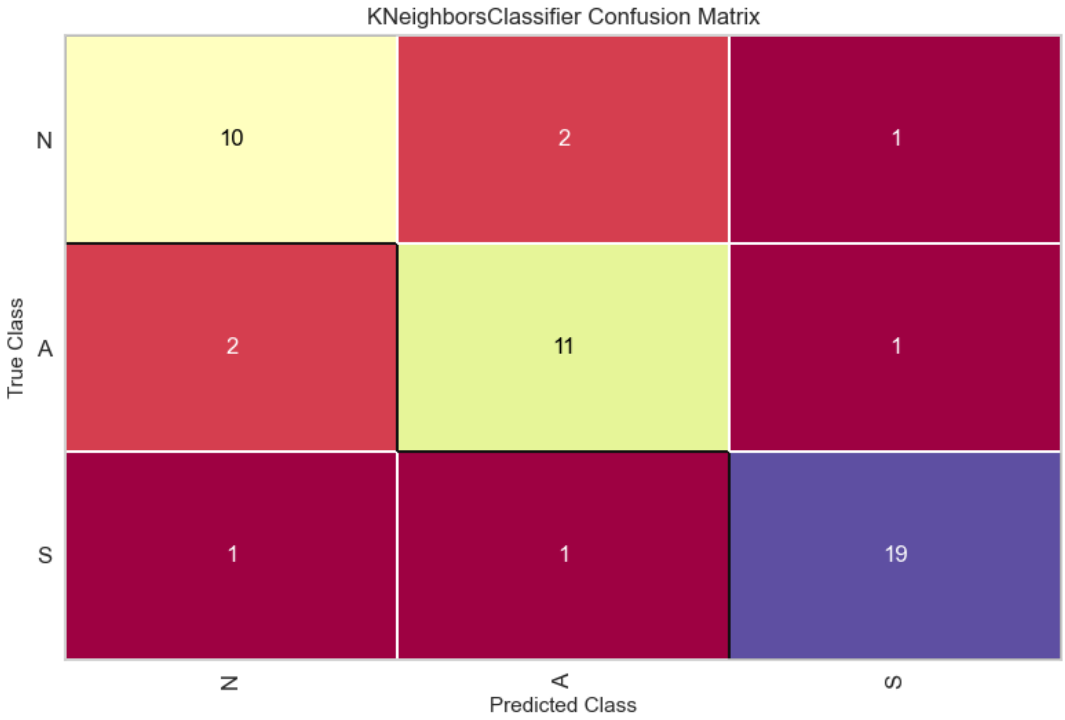
5.4.1.7 Vecinos más cercanos (KNN)

En este modelo, se emplearon los parámetros predefinidos del algoritmo, ya que demostraron ser los más eficaces para la tarea en cuestión.

Tabla 11 Métricas KNN.

	Precisión	**Sensibilidad**	**Puntuación F1**	**Soporte**
0.0	0,7692	0,7692	0,7692	13
1.0	0,7857	0,7857	0,7857	14
2.0	0,9048	0,9048	0,9048	21
Exactitud			0,8333	48
Promedio macro	0,8199	0,8199	0,8199	48
Promedio ponderado	0,8333	0,8333	0,8333	48

Figura 30 Matriz de confusión KNN.



A partir de las métricas resultantes del modelo se observa una exactitud no muy alta, del 83.33%, al igual que la precisión, sensibilidad y puntuación F1 con el mismo valor del 81,99%. Al analizar la matriz de confusión se denota una clara dificultad del modelo para clasificar correctamente todas las instancias:

- Seronegativos: En esta clasificación, se lograron 10 predicciones verdaderas positivas, con dos falsos negativos. El primero ocurrió al diagnosticar a dos pacientes como enfermos asintomáticos, y el segundo al detectarlo como enfermo sintomático. Además, se presentaron falsos positivos al clasificar a 3 personas enfermas con la enfermedad: 2 asintomáticas y 1 sintomática.

- Asintomáticos: El modelo acertó en el diagnóstico de 11 personas con enfermedad de Chagas asintomáticas. Sin embargo, se produjeron tres falsos negativos: dos al clasificar a una persona con enfermedad de Chagas asintomática como sana y otra como enferma sintomática. Asimismo, se identificaron tres falsos positivos, al diagnosticar incorrectamente a dos personas sanas como enfermas y a un paciente sintomático como asintomático.
- Sintomáticos: Se logró el acierto de 19 de 21 pacientes con la enfermedad activa de manera sintomática. De la misma manera se obtuvieron de manera equivalente dos falsos positivos y dos falsos negativos en los cuales hubo una confusión igualitaria con cada una de las clases restantes.

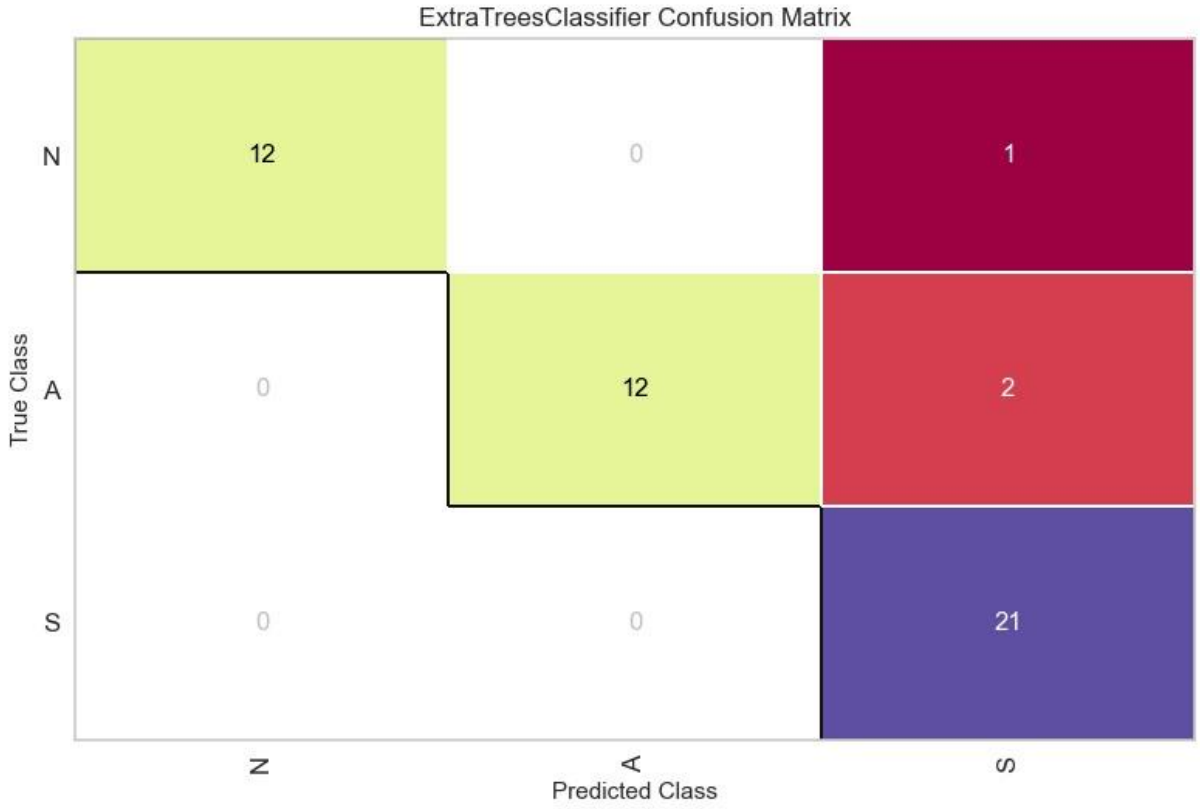
5.4.1.8 Árboles extra (*ExtraTrees*)

El parámetro usado en este modelo es “n_stimators=300”. Este es un parámetro que especifica el número de árboles de decisión que se utilizarán en el modelo de clasificación de *ExtraTrees*, en general este valor es proporcional a la mejora del modelo. El valor tomado para este parámetro significa que el modelo utilizará 300 árboles de decisión para clasificar los datos.

Tabla 12 Métricas ExtraTrees

	Precisión	Sensibilidad	Puntuación F1	Soporte
0.0	1,0000	0,9231	0,9600	13
1.0	1,0000	0,8571	0,9231	14
2.0	0,8750	1,0000	0,9333	21
Exactitud			0,9375	48
Promedio macro	0,9583	0,9267	0,9388	48
Promedio ponderado	0,9453	0,9375	0,9376	48

Figura 31 Matriz de confusión ExtraTrees.



Las métricas generadas indican que el modelo presenta un buen rendimiento en la tarea de clasificación, con una exactitud del 93.75%. En este desempeño se destaca especialmente en

términos de precisión y puntuación F1. Al comparar estos resultados con la matriz de confusión se encuentra una coherencia ya que tuvo muy pocos desaciertos. Se diagnosticaron correctamente las clases seronegativas y asintomáticas, 12 en cada una, habiendo solo falsos negativos, dos para la predicción de la clase asintomática y otro falso negativo con las personas sin enfermedad que el modelo clasificó una de ellas enferma. Por otro lado, el modelo tuvo éxito en la predicción de los pacientes con la enfermedad sintomática, con la diferencia que este tuvo tres falsos positivos.

5.4.1.9 Regresión logística (*Logistic Regression*)

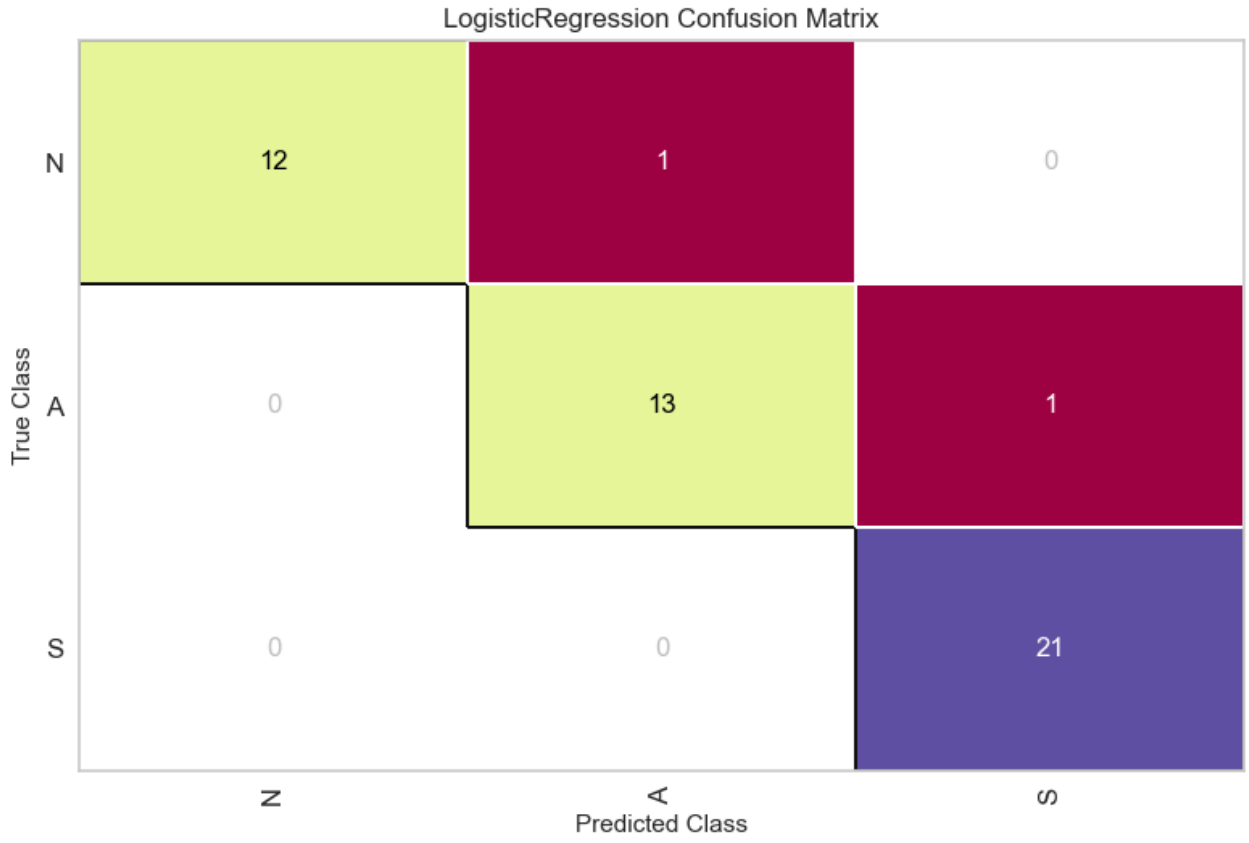
El parámetro utilizado en este modelo, solver='liblinear', especifica el algoritmo de optimización que se emplea para ajustar los parámetros del modelo de regresión logística. En este caso, 'liblinear' se selecciona para resolver problemas de optimización lineal, es adecuado para problemas de clasificación de gran escala con un número de características mucho mayor que el número de muestras

Tabla 13 Métricas Logistic Regression

	Precisión	**Sensibilidad**	**Puntuación F1**	**Soporte**
0.0	1,0000	0,9231	0,9600	13
1.0	0,9286	0,9286	0,9286	14
2.0	0,9545	1,0000	0,9767	21
Exactitud			0,9583	48
Promedio macro	0,9593	0,9505	0,9583	48
Promedio ponderado	0,9593	0,9583	0,9582	48

En este caso, las métricas reflejan un desempeño destacado del modelo, logrando una exactitud del 95.83% y manteniendo valores similares en otras métricas. La matriz de confusión (Figura 32) respalda estos resultados al indicar que el modelo de regresión logística ha clasificado correctamente casi todas las muestras, falla poniendo falsos negativos en las predicciones de seronegativos y asintomáticos, otra cosa en la que falla es en detectar falsos positivos en la clase asintomático y seronegativo.

Figura 32 Matriz de confusión Logistic Regression.



5.5 Comparación de modelos

Después del desarrollo de nueve algoritmos clasificadores basados en la diferenciación de clases (asintomático, sintomático y seronegativo) se observaron los resultados mostrados en la Tabla 14 y en la Figura 33.

Figura 33 Comparación de modelos

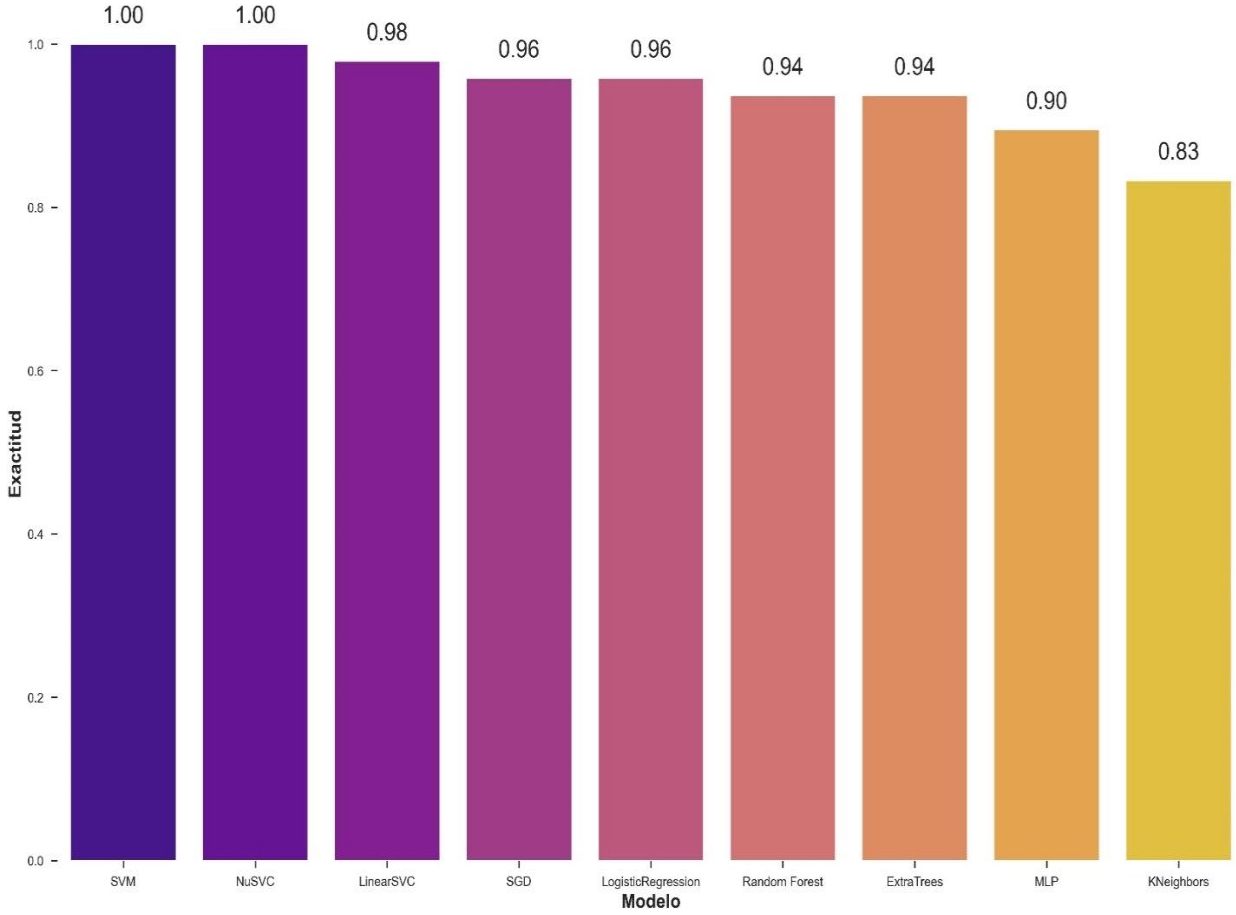


Tabla 14 Rendimiento de predicción de modelos en los conjuntos de prueba

Modelo	Exactitud	Precisión	Sensibilidad	F1-score
SVC	1,0000	1,0000	1,0000	1,0000
NuSVC	1,0000	1,0000	1,0000	1,0000
LiSVC	0,9792	0,9801	0,9792	0,9790
SGD	0,9583	0,9635	0,9583	0,9591
Logistic Regression	0,9583	0,9593	0,9583	0,9582
Random Forest	0,9375	0,9395	0,9375	0,9369
ExtraTrees	0,9375	0,9453	0,9375	0,9376
MLP	0,8958	0,8983	0,8958	0,8964
KNN	0,8333	0,8333	0,8333	0,8333

Comparando las métricas de los modelos entrenados, se destaca que los clasificadores SVC y NuSVC lograron un rendimiento perfecto del 100%, indicando que pudieron identificar cada una de las clases sin dificultad alguna.

LiSVC, Logistic Regression, SGD, Extra Trees y Random Forest exhibieron un rendimiento elevado, superando el 93% en todas sus métricas. Esto significa que estos modelos realizaron más de 93 de cada 100 predicciones correctamente.

Finalmente, los dos modelos restantes, MLP y KNN, mostraron un desempeño por debajo del 90%. Aunque para este tipo de análisis pueda considerarse aceptable, se debe tener en cuenta que no alcanzan un nivel de confiabilidad óptimo, especialmente en el contexto de búsqueda de herramientas diagnósticas.

6. Conclusiones

El estudio se centró en la implementación de un protocolo de preparación de muestras sanguíneas para obtener perfiles proteómicos, con el objetivo de realizar análisis mediante espectrometría de masas MALDI-TOF y utilizar los resultados para crear modelos predictivos para diagnóstico.

La técnica de preparación de muestras asistida por filtro, conocida como FASP, demostró ser altamente adaptable a las demandas del análisis de sueros sanguíneos en pacientes con la Enfermedad de Chagas. Su versatilidad se destaca no solo por su facilidad de uso, sino también por su eficiente capacidad para procesar un gran número de muestras de suero sanguíneo, optimizando los tiempos de tratamiento sin comprometer la integridad de las proteínas en la muestra.

En el marco del análisis por espectrometría de masas MALDI-TOF se logró identificar y definir con éxito los parámetros instrumentales esenciales como la afluencia del láser, el método de ionización, la relación matriz-muestras y el rango de masas para garantizar la adquisición de espectros de alta calidad.

Por último, el estudio demostró la viabilidad de utilizar herramientas computacionales para el desarrollo de herramientas de predicción para el diagnóstico de enfermedades. Se evaluaron nueve modelos de clasificación de aprendizaje automático supervisado con el objetivo de desarrollar una o varias herramientas diagnósticas alternativas, utilizando sueros sanguíneos. La aplicación de estas herramientas demostró su eficacia al clasificar exitosamente las tres variables problema, a pesar de las diferentes distribuciones de cantidad de muestra. Los resultados detallan que los modelos lograron una clasificación con una exactitud promedio del 83.88%, exhibiendo

un rango que varía desde un mínimo del 83% hasta un máximo del 100%. Con esto, se destaca la capacidad de las herramientas computacionales para abordar desafíos diagnósticos, sino también de su adaptabilidad para ofrecer resultados precisos y fiables en la clasificación de enfermedades, apoyando a los diagnósticos médicos.

7. Referencias Bibliográficas

Agrawal, G. K., Sarkar, A., Righetti, P. G., Pedreschi, R., Carpentier, S., Wang, T., Barkla, B. J., Kohli, A., Ndimba, B. K., Bykova, N. V., Rampitsch, C., Zolla, L., Rafudeen, M. S., Cramer, R., Bindschedler, L. V., Tsakirpaloglou, N., Ndimba, R. J., Farrant, J. M., Renaut, J., … Rakwal, R. (2013). A decade of plant proteomics and mass spectrometry: translation of technical advancements to food security and safety issues. *Mass Spectrometry Reviews*, *32*(5), 335–365. https://doi.org/10.1002/MAS.21365

Al-Amrani, S., Al-Jabri, Z., Al-Zaabi, A., Alshekaili, J., & Al-Khabori, M. (2021). Proteomics: Concepts and applications in human medicine. *World Journal of Biological Chemistry*, *12*(5), 57. https://doi.org/10.4331/WJBC.V12.I5.57

Alducin-Téllez, C., Rueda-Villegas, E., Medina-Yerbes, I., Hernández, O., López, R., Peña-Hernández, V., & Monteón, V. (2011). Prevalencia de serología positiva para Trypanosoma cruzi en pacientes con diagnóstico clínico de miocardiopatía dilatada en el Estado de Campeche, México. *Archivos de Cardiología de México*, *81*, 204–207.

Amazon. (2023). *¿Qué es una red neuronal? - Explicación de las redes neuronales artificiales - AWS*. https://aws.amazon.com/es/what-is/neural-network/

Astrof, N. S., & Horowitz, G. (2018). Protein Colorimetry Experiments That Incorporate Intentional Discrepancies and Historical Narratives. *Journal of Chemical Education*, *95*(7), 1198–1204. https://doi.org/10.1021/ACS.JCHEMED.7B00633/ASSET/IMAGES/MEDIUM/ED-2017-00633R_0004.GIF

Bader, O. (2013). MALDI-TOF-MS-based species identification and typing approaches in medical mycology. *Proteomics*, *13*(5), 788–799. https://doi.org/10.1002/pmic.201200468

Barrios Peréz Camilo. (2016). Zonificación agroecológica para el cultivo de arroz de riego (Oryza Sativa L.) en Colombia. *Nacional de Colombia*. https://doi.org/10.13140/RG.2.2.28064.79361/1

Benagli, C., Rossi, V., Dolina, M., Tonolla, M., & Petrini, O. (2011). Matrix-Assisted Laser Desorption Ionization-Time of Flight Mass Spectrometry for the Identification of Clinically Relevant Bacteria. *PLOS ONE*, *6*(1), e16424. https://doi.org/10.1371/JOURNAL.PONE.0016424

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324/METRICS

Bruker Daltonics. (2012, September). *Bruker Guide to MALDI Sample Preparation*. https://researchservices.pitt.edu/sites/default/files/Bruker_Guide%20for%20MALDI_Sample_Preparation.pdf

Canals, M., Cáceres, D., Alvarado, S., Canals, A., & Cattan, P. E. (2017). Modeling Chagas disease in Chile: From vector to congenital transmission. *BioSystems*, *156–157*, 63–71. https://doi.org/10.1016/j.biosystems.2017.04.004

Cantey, P. T., Stramer, S. L., Townsend, R. L., Kamel, H., Ofafa, K., Todd, C. W., Currier, M., Hand, S., Varnado, W., Dotson, E., Hall, C., Jett, P. L., & Montgomery, S. P. (2019a). CDC - Chagas Disease - Epidemiology & Risk Factors. *Transfusion*, *52*(9), 1922–1930. https://doi.org/10.1111/J.1537-2995.2012.03581.X/FULL

Cantey, P. T., Stramer, S. L., Townsend, R. L., Kamel, H., Ofafa, K., Todd, C. W., Currier, M., Hand, S., Varnado, W., Dotson, E., Hall, C., Jett, P. L., & Montgomery, S. P. (2019b). Chagas Disease - Epidemiology & Risk Factors. *Transfusion*, *52*(9), 1922–1930. https://doi.org/10.1111/J.1537-2995.2012.03581.X/FULL

Carbonnelle, E., Mesquita, C., Bille, E., Day, N., Dauphin, B., Beretti, J. L., Ferroni, A., Gutmann, L., & Nassif, X. (2011). MALDI-TOF mass spectrometry tools for bacterial identification in clinical microbiology laboratory. *Clinical Biochemistry*, *44*(1), 104–109. https://doi.org/10.1016/J.CLINBIOCHEM.2010.06.017

Caro-Miró, M. A., Morales-Romero, B., & García-Villoria, J. (2022). Utilidad de la espectrometría de masas para el análisis de metabolitos en líquido cefalorraquídeo de pacientes con errores congénitos del metabolismo. *Revista de Medicina de Laboratorio*. https://doi.org/10.20960/REVMEDLAB.00150

Chu, Z., Yu, J., & Hamdulla, A. (2020). Throughput prediction based on ExtraTree for stream processing tasks. *Computer Science and Information Systems*, *18*(1), 1–22. https://doi.org/10.2298/CSIS200131031C

Cobo, F. (2013). Application of maldi-tof mass spectrometry in clinical virology: a review. *The Open Virology Journal*, *7*, 84–90. https://doi.org/10.2174/1874357920130927003

Croxatto, A., Prod'hom, G., & Greub, G. (2012). Applications of MALDI-TOF mass spectrometry in clinical diagnostic microbiology. *FEMS Microbiology Reviews*, *36*(2), 380–407. https://doi.org/10.1111/J.1574-6976.2011.00298.X

Cucunubá, Z. M., Manne-Goehler, J. M., Díaz, D., Nouvellet, P., Bernal, O., Marchiol, A., Basáñez, M. G., & Conteh, L. (2017a). How universal is coverage and access to diagnosis and treatment for Chagas disease in Colombia? A health systems analysis. *Social Science and Medicine*, *175*, 187–198. https://doi.org/10.1016/j.socscimed.2017.01.002

Cucunubá, Z. M., Manne-Goehler, J. M., Díaz, D., Nouvellet, P., Bernal, O., Marchiol, A., Basáñez, M. G., & Conteh, L. (2017b). How universal is coverage and access to diagnosis

and treatment for Chagas disease in Colombia? A health systems analysis. *Social Science & Medicine*, *175*, 187–198. https://doi.org/10.1016/J.SOCSCIMED.2017.01.002

de Andrade, J. P., Neto, J. A. M., de Paola, A. A. V., Vilas-Boas, F., Oliveira, G. M. M., Bacal, F., Bocchi, E. A., Almeida, D. R., Filho, A. A. F., Moreira, M. da C. V., Xavier, S. S., de Oliveira, W. A., & Dias, J. C. P. (2011). I Latin American Guidelines for the diagnosis and treatment of Chagas' heart disease: executive summary. *Arquivos Brasileiros de Cardiologia*, *96*(6), 434–442. https://doi.org/10.1590/S0066-782X2011000600002

de Sousa, A. S., Vermeij, D., Ramos, A. N., & Luquetti, A. O. (2023). Chagas disease. *The Lancet*. https://doi.org/10.1016/S0140-6736(23)01787-7

Deng, X., Liu, Q., Deng, Y., & Mahadevan, S. (2016). An improved method to construct basic probability assignment based on the confusion matrix for classification problem. *Information Sciences*, *340–341*, 250–261. https://doi.org/10.1016/J.INS.2016.01.033

Dieckmann, R., & Malorny, B. (2011). Rapid screening of epidemiologically important Salmonella enterica subsp. enterica serovars by whole-cell matrix-assisted laser desorption ionization-time of flight mass spectrometry. *Applied and Environmental Microbiology*, *77*(12), 4136–4146. https://doi.org/10.1128/AEM.02418-10

Gross, J. H. (2017). *Mass Spectrometry*.

Guhl, F. (2005, May 6). *Memorias del Primer Taller Internacional sobre Control de la Enfermedad de Chagas.* Organización Panamericana de La Salud. https://www.paho.org/hq/dmdocuments/2012/V-Reunion-IPA-2010.pdf

Herazo, R., Torres-Torres, F., Mantilla, C. A. G., Carillo, L. P., Cuervo, A., Camargo, M. A. M., Moreno, J. F., Forsyth, C., Vera, M. J., Díaz, R. A. C., & Marchiol, A. (2022). On-site experience of a project to increase access to diagnosis and treatment of Chagas disease in

high-risk endemic areas of Colombia. *Acta Tropica*, *226*. https://doi.org/10.1016/j.actatropica.2021.106219

Huang, S., Huang, M., & Lyu, Y. (2020). An Improved KNN-Based Slope Stability Prediction Model. *Advances in Civil Engineering*, *2020*. https://doi.org/10.1155/2020/8894109

Huang, Y.-C., Chung, H.-H., Dutkiewicz, E. P., Chen, C.-L., Hsieh, H.-Y., Chen, B.-R., Wang, M.-Y., & Hsu, C.-C. (2020). Predicting Breast Cancer by Paper Spray Ion Mobility Spectrometry Mass Spectrometry and Machine Learning. *Analytical Chemistry*, *92*(2), 1653–1657. https://doi.org/10.1021/acs.analchem.9b03966

Institute of Medicine. (2008). Vector-Borne Diseases: Understanding the Environmental, Human Health, and Ecological Connections: Workshop Summary. *Vector-Borne Diseases*. https://doi.org/10.17226/11950

Instituto Nacional de Salud. (2023). *Protocolo de Vigilancia en Salud Pública de Chagas*. https://doi.org/10.33610/infoeventos.58

JAVA. (2019). *K-Nearest Neighbor(KNN) Algorithm for Machine Learning - Javatpoint*. https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning

Jollife, I. T., & Cadima, J. (2016). Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *374*(2065). https://doi.org/10.1098/RSTA.2015.0202

Kingsley, S., Xu, Z., Jones, B., Saleh, J., & Orlando, T. M. (2023). A Mass Spectrometry-Machine Learning Approach for Detecting Volatile Organic Compound Emissions for Early Fire Detection. *Journal of the American Society for Mass Spectrometry*, *34*(5), 826–835. https://doi.org/10.1021/JASMS.2C00304/ASSET/IMAGES/LARGE/JS2C00304_0009.JPEG

Laroche, M., Almeras, L., Pecchi, E., Bechah, Y., Raoult, D., Viola, A., & Parola, P. (2017). MALDI-TOF MS as an innovative tool for detection of Plasmodium parasites in Anopheles mosquitoes. *Malaria Journal*, *16*(1), 1–10. https://doi.org/10.1186/S12936-016-1657-Z/TABLES/4

Leng, Y., Xu, X., & Qi, G. (2013). Combining active learning and semi-supervised learning to construct SVM classifier. *Knowledge-Based Systems*, *44*, 121–131. https://doi.org/10.1016/J.KNOSYS.2013.01.032

Li, M., Zhang, L., Yao, X., & Jiang, X. (2017). Membrane introduction mass spectrometry combined with an orthogonal partial-least squares calibration model for mixture analysis. *Analytical Sciences*, *33*(11), 1225–1230. https://doi.org/10.2116/ANALSCI.33.1225/METRICS

Li, Y., Gan, Z., Zhou, X., & Chen, Z. (2022). Accurate classification of Listeria species by MALDI-TOF mass spectrometry incorporating denoising autoencoder and machine learning. *Journal of Microbiological Methods*, *192*, 106378. https://doi.org/https://doi.org/10.1016/j.mimet.2021.106378

Luquetti, A. O., & Schmuñis, G. A. (2017). Diagnosis of Trypanosoma cruzi infection. *American Trypanosomiasis Chagas Disease: One Hundred Years of Research: Second Edition*, 687–730. https://doi.org/10.1016/B978-0-12-801029-7.00030-7

Ma, X., Chen, T., Xv, F., & Li, J. (2022). *PM2.5 concentration forecasting in the area of Jing-Jin-Ji using models based on RF, RR, SVM, and ExtraTrees*. https://doi.org/10.21203/RS.3.RS-2319186/V1

Maldonado, N., Robledo, C., & Robledo, J. (2018). La espectrometría de masas MALDI-TOF en el laboratorio de microbiología clínica. *Infectio*, *22*(1), 35–45. https://doi.org/10.22354/in.v0i0.703

MATLAB. (2023a). *Conceptos clave de Support Vector Machine (SVM) - MATLAB & Simulink*. https://la.mathworks.com/discovery/support-vector-machine.html

MATLAB. (2023b). *¿Qué es una red neuronal? - MATLAB & Simulink*. https://la.mathworks.com/discovery/neural-network.html

Molassiotis, A., Tanou, G., Filippou, P., & Fotopoulos, V. (2013). Proteomics in the fruit tree science arena: New insights into fruit defense, development, and ripening. *PROTEOMICS*, *13*(12–13), 1871–1884. https://doi.org/10.1002/PMIC.201200428

Momo, R. A., Povey, J. F., Smales, C. M., O'Malley, C. J., Montague, G. A., & Martin, E. B. (2013). MALDI-ToF mass spectrometry coupled with multivariate pattern recognition analysis for the rapid biomarker profiling of Escherichia coli in different growth phases. *Analytical and Bioanalytical Chemistry*, *405*(25), 8251–8265. https://doi.org/10.1007/S00216-013-7245-Y

Moncayo, Á., & Silveira, A. C. (2009). Current epidemiological trends for Chagas disease in Latin America and future challenges in epidemiology, surveillance and health policy. *Memorias Do Instituto Oswaldo Cruz*, *104 Suppl 1*(SUPPL. 1), 17–30. https://doi.org/10.1590/S0074-02762009000900005

Morais, M. C. C., Silva, D., Milagre, M. M., de Oliveira, M. T., Pereira, T., Silva, J. S., da Costa, L. F., Minoprio, P., Cesar, R. M., Gazzinelli, R., de Lana, M., & Nakaya, H. I. (2022). Automatic detection of the parasite Trypanosoma cruzi in blood smears using a machine

learning approach applied to mobile phone images. *PeerJ*, *10*. https://doi.org/10.7717/PEERJ.13470/SUPP-3

Morillo C.A, Altcheh, J., & Marin-Neto J.A. (2015). Randomized trial of benznidazole for chronic Chagas' cardiomyopathy. In *Archivos Argentinos de Pediatria* (Vol. 114, Issue 2, pp. e124–e125). Sociedad Argentina de Pediatria. https://doi.org/10.1056/nejmoa1507574

Mortier, T., Wieme, A. D., Vandamme, P., & Waegeman, W. (2021). Bacterial species identification using MALDI-TOF mass spectrometry and machine learning techniques: A large-scale benchmarking study. *Computational and Structural Biotechnology Journal*, *19*, 6157–6168. https://doi.org/https://doi.org/10.1016/j.csbj.2021.11.004

Nedyalkova, M., Vasighi, M., Azmoon, A., Naneva, L., & Simeonov, V. (2022). Sequence-Based Prediction of Plant Allergenic Proteins: Machine Learning Classification Approach. *ACS Omega*, *8*, 3698–3704. https://doi.org/10.1021/ACSOMEGA.2C02842/ASSET/IMAGES/LARGE/AO2C02842_0003.JPEG

Nugroho, A., Widyawan, & Kusumawardani, S. S. (2020a). Distributed Classifier for SDGs Topics in Online News using RabbitMQ Message Broker. *Journal of Physics: Conference Series*, *1577*(1), 012026. https://doi.org/10.1088/1742-6596/1577/1/012026

Nugroho, A., Widyawan, & Kusumawardani, S. S. (2020b). Distributed Classifier for SDGs Topics in Online News using RabbitMQ Message Broker. *Journal of Physics: Conference Series*, *1577*(1), 012026. https://doi.org/10.1088/1742-6596/1577/1/012026

Olivera, M. J., Fory, J. A., Porras, J. F., & Buitrago, G. (2019). Prevalence of Chagas disease in Colombia: A systematic review and meta-analysis. *PLOS ONE*, *14*(1), e0210156-. https://doi.org/10.1371/journal.pone.0210156

Organización mundial de la salud. (2022, January 10). *La tripanosomiasis africana (enfermedad del sueño)*. https://www.who.int/es/news-room/fact-sheets/detail/trypanosomiasis-human-african-(sleeping-sickness)

Organización Panamericana de la Salud. (2020). *Enfermedad de Chagas*. https://www.paho.org/es/temas/enfermedad-chagas

Osmangazi Tıp Dergisi Osmangazi ; Ozen, H., & Bal, C. (2020). A Study on Missing Data Problem in Random Forest. *Osmangazi Journal of Medicine*, *42*(1), 103–109. https://doi.org/10.20515/OTD.496524

Palmezano Díaz, J. M., Plazas Rey, L. K., Rivera Castillo, K. E., & Rueda Rojas, V. P. (2015). Enfermedad de chagas: realidad de una patología frecuente en Santander, Colombia. *Médicas UIS*, *28*(1), 81–90. https://revistas.uis.edu.co/index.php/revistamedicasuis/article/view/4908

Pan American Health Organization. (2019). *Guidelines for the diagnosis and treatment of Chagas disease*.

Pecks, U., Seidenspinner, F., Röwer, C., Reimer, T., Rath, W., & Glocker, M. O. (2010). Multifactorial analysis of affinity-mass spectrometry data from serum protein samples: a strategy to distinguish patients with preeclampsia from matching control individuals. *Journal of the American Society for Mass Spectrometry*, *21*(10), 1699–1711. https://doi.org/10.1016/J.JASMS.2009.12.013

Pedregosa, F., Michel, V., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Vanderplas, J., Cournapeau, D., Pedregosa, F., Varoquaux, G., Gramfort, A., Thirion, B., Grisel, O., Dubourg, V., Passos, A., Brucher, M., Perrot andÉdouardand, M., Duchesnay, andÉdouard, & Duchesnay EDOUARDDUCHESNAY, Fré. (2011). Scikit-learn: Machine Learning in Python Gaël Varoquaux Bertrand Thirion Vincent Dubourg Alexandre Passos

PEDREGOSA, VAROQUAUX, GRAMFORT ET AL. Matthieu Perrot. In *Journal of Machine Learning Research* (Vol. 12). http://scikit-learn.sourceforge.net.

Picado, A., Cruz, I., Redard-Jacot, M., Schijman, A. G., Torrico, F., Sosa-Estani, S., Katz, Z., & Ndung'u, J. M. (2018). The burden of congenital Chagas disease and implementation of molecular diagnostic tools in Latin America. *BMJ Global Health*, *3*(5), e001069. https://doi.org/10.1136/BMJGH-2018-001069

Qian, L., Sun, R., Xue, Z., & Guo, T. (2023). Mass Spectrometry–Based Proteomics of Epithelial Ovarian Cancers: A Clinical Perspective. *Molecular & Cellular Proteomics*, *22*(7), 100578. https://doi.org/10.1016/J.MCPRO.2023.100578

Raicu, I., & Bologa, R. (2019). *MULTI-CLASS TEXT SUPERVISED CLASSIFICATION ON ROMANIAN FINANCIAL BANKING REVIEWS*. https://doi.org/10.12948/ie2019.01.06

Raiesi, O., Mohseni, M., & Shamsaei, S. (2019). *Human Fungal Pathogen Identification and Diagnosis: (elementary to advanced)*.

Regan, M., Engler, M. B., Coleman, B., Daack-Hirsch, S., & Calzone, K. A. (2019). Establishing the Genomic Knowledge Matrix for Nursing Science. *Journal of Nursing Scholarship*, *51*(1), 50–57. https://doi.org/10.1111/JNU.12427

Rohman, A., & Putri, A. R. (2019). The Chemometrics Techniques in Combination with Instrumental Analytical Methods Applied in Halal Authentication Analysis. *Indonesian Journal of Chemistry*, *19*(1), 262–272. https://doi.org/10.22146/IJC.28721

Rotemberg, E., Rotemberg, E., Picapedra, A., & Kreiner, M. (2022). Detección de drogas en saliva: aspectos metodológicos y legales. *Odontología Sanmarquina*, *25*(1), e22076. https://doi.org/10.15381/os.v25i1.22076

SINGHAL, N., KUMAR, M., & VIRDI, J. S. (2016). MALDI-TOF MS in clinical parasitology: applications, constraints and prospects. *Parasitology*, *143*(12), 1491–1500. https://doi.org/10.1017/S0031182016001189

Subasi, A. (2020). Machine learning techniques. *Practical Machine Learning for Data Analysis Using Python*, 91–202. https://doi.org/10.1016/B978-0-12-821379-7.00003-5

ThermoFisherScientific. (2020). *Preparation of the BCA working reagent (WR) Microplate procedure (sample to WR ratio = 1:8)*. https://www.thermofisher.com/bcafaqs

Timm, W., Scherbart, A., Böcker, S., Kohlbacher, O., & Nattkemper, T. W. (2008). Peak intensity prediction in MALDI-TOF mass spectrometry: A machine learning study to support quantitative proteomics. *BMC Bioinformatics*, *9*(1), 443. https://doi.org/10.1186/1471-2105-9-443

Varona uribe, M., Celia Montiel, A., Beltrán Durán, M., & Lugo Vargas Coordinadora Grupo Entomología RNL, L. (2014). *Enfermedad de Chagas Kuigja (Pito, insecto vector)*.

Vella, A., de Carolis, E., Mello, E., Perlin, D. S., Sanglard, D., Sanguinetti, M., & Posteraro, B. (2017). Potential Use of MALDI-ToF Mass Spectrometry for Rapid Detection of Antifungal Resistance in the Human Pathogen Candida glabrata. *Scientific Reports*, *7*(1). https://doi.org/10.1038/s41598-017-09329-4

Whitman, J. D. (2023). Chagas Disease: a Review and Perspective on Laboratory Diagnostics in the United States. *Clinical Microbiology Newsletter*, *45*(17), 141–149. https://doi.org/10.1016/J.CLINMICNEWS.2023.09.001

Wilkins, M. R., Sanchez, J. C., Gooley, A. A., Appel, R. D., Humphery-Smith, I., Hochstrasser, D. F., & Williams, K. L. (1996). Progress with proteome projects: why all proteins expressed

by a genome should be identified and how to do it. *Biotechnology & Genetic Engineering Reviews*, *13*(1), 19–50. https://doi.org/10.1080/02648725.1996.10647923

Wiśniewski, J. R. (2018). Filter-Aided Sample Preparation for Proteome Analysis. In D. Becher (Ed.), *Microbial Proteomics: Methods and Protocols* (pp. 3–10). Springer New York. https://doi.org/10.1007/978-1-4939-8695-8_1

World Health Organization. (2002). *Control of Chagas disease : second report of the WHO expert committee*. https://iris.who.int/handle/10665/42443?show=full

World Health Organization. (2021, April 1). *La enfermedad de Chagas* . https://www.who.int/es/news-room/fact-sheets/detail/chagas-disease-%28american-trypanosomiasis%29

Yuan, X., Puvogel, S., van Rhijn, J. R., Ciptasari, U., Esteve-Codina, A., Meijer, M., Rouschop, S., van Hugte, E. J. H., Oudakker, A., Schoenmaker, C., Frega, M., Schubert, D., Franke, B., & Nadif Kasri, N. (2023). A human in vitro neuronal model for studying homeostatic plasticity at the network level. *Stem Cell Reports*, *18*(11), 2222–2239. https://doi.org/10.1016/J.STEMCR.2023.09.011

Zaidi, A. (2022). Mathematical justification on the origin of the sigmoid in logistic regression. *CEMJP*, *30*(4), 1327–1337. https://doi.org/10.57030/23364890.CEMJ.30.4.135

Zhang, S., Li, H., Hu, Q., Wang, Z., & Chen, X. (2022). Discrimination of thermal treated bovine milk using MALDI-TOF MS coupled with machine learning. *Food Control*, *142*, 109224. https://doi.org/https://doi.org/10.1016/j.foodcont.2022.109224

8. Apéndices

Apéndice a Memorias de la cuantificación de proteínas

Absorbancia 1

Longitud de onda: 562 nm

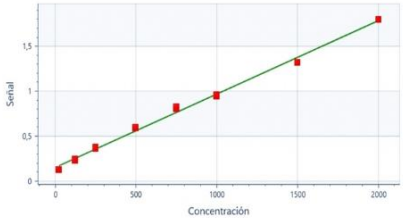
Placa 1

Abs	1	2	3	4	5	6	7	8	9	10	11	12
A	~~1,9588~~	0,8529	0,8588	0,7520	2,3515	0,5089	0,6221	0,4698	0,2689	0,3688	0,3577	1,8045
B	1,3254	0,3470	0,3515	0,4419	1,0058	0,3301	0,4185	0,4488	0,5921	0,3425	0,3451	~~1,2223~~
C	0,9522	0,2939	0,3367	0,4354	1,9437	0,7997	2,3697	0,5460	0,6500	0,2988	0,2890	0,9640
D	0,8278	0,8214	0,5719	0,2746	0,9598	0,2381	2,8700	0,3403	1,3075	0,4976	0,5070	0,8068
E	0,5979	0,4068	1,1053	0,6695	1,1422	0,4794	2,7376	0,2565	1,3475	0,5798	0,6125	0,6055
F	0,3679	0,5020	1,4279	0,7800	0,6424	0,2235	1,4489	0,2257	1,1188			0,3819
G	0,2500	0,5122	1,3109	0,6046	1,1374	0,4691	1,5156	0,5204	1,8039			0,2373
H	0,1299	0,1040		0,5552	0,8154	0,6824	1,7046				0,1062	0,1351

R^2: 0,996

y = 0,000816081x + 0,154894

a 0,000816 **b** 0,154893683



MUESTRAS

Asintomaticos		**Concentración ug/mL**	**Factor**	**[]**	**mg/mL**	**# muestra**
1	0,8529	855,3150		128297,2493	128,2972493	**530**
2	0,3470	235,4010		35310,15615	35,31015615	**736**
3	0,2939	170,3340		25550,09557	25,55009557	**2922**
4	0,8214	816,7159		122507,3829	122,5073829	**3093**
5	0,7560	736,5768	150	110486,5173	110,4865173	**1735**
6	0,5020	425,3332		63799,97518	63,79997518	**1799**
7	0,5122	437,8319		65674,78907	65,67478907	**1633**
8	0,7520	731,6753		109751,2962	109,7512962	**522**
9	0,7239	697,1812		104577,1774	104,5771774	**529**
10	1,1896	1267,8353		190175,2982	190,1752982	**489**
Sintomaticos		**Concentración ug/mL**	**Factor**			
1	0,2746	146,6844		22002,65359	22,00265359	**22**
2	0,6695	630,5824		94587,36025	94,58736025	**2787**
3	0,7800	765,9856		114897,8441	114,8978441	**2879**
4	0,6046	551,0560		82658,39732	82,65839732	**2098**
5	0,5552	490,5228	150	73578,41629	73,57841629	**682**
6	0,5089	433,7882		65068,23164	65,06823164	**2217**
7	0,3301	214,6923		32203,84685	32,20384685	**2937**
8	0,7997	790,1254		118518,8082	118,5188082	**3000**
9	0,2381	101,9584		15293,76072	15,29376072	**2896**
10	0,4794	397,6398		59645,97576	59,64597576	**810**
Seronegativas		**Concentración ug/mL**	**Factor**			
1	0,8362	834,8513		125227,7011	125,2277011	**1560**
2	0,4691	385,0185		57752,78133	57,75278133	**2168**
3	0,6824	646,3897		96958,44842	96,95844842	**2190**
4	0,4698	385,8763		57881,44503	57,88144503	**2660**
5	0,4488	360,1436	150	54021,53407	54,02153407	**2155**
6	0,5460	479,2494		71887,40767	71,88740767	**2220**
7	0,8239	819,7793		122966,8961	122,9668961	**1831**
8	0,8020	792,9437		118941,5604	118,9415604	**1907**
9	0,6723	633,9522		95092,82478	95,09282478	**1606**
10	1,1622	1234,2602		185139,0334	185,1390334	**1850**

Los valores resultantes se determinaron gracias a la ecuación de la recta $y = mx + b$, en la que “y” representa la absorbancia, “x” la concentración de proteínas, “m” la pendiente de la

recta y “b” el intercepto. La concentración de proteínas en los sueros sanguíneos se puede calcular interpolando la absorbancia de la muestra en la curva estándar. La concentración de proteínas se calcula utilizando la siguiente ecuación:

$$x = \frac{y - b}{m} \quad \textbf{(5)}$$

Teniendo la concentración se multiplica por el factor de dilución para obtener la concentración original de la muestra antes de la dilución. Con la concentración real de cada uno de los sueros sanguíneos, se determinó el número de sueros aptos para proceder a la fase de digestión enzimática.

Apéndice b Realización del código

Predictive Model

Machine learning-based Chagas disease diagnostic tool with proteomic profiles

Proyecto de grado para optar por el titulo de Quimica

Yenny Velandia

Samples resulting from filtration less than 3kDa

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.decomposition import PCA
from sklearn import preprocessing
from sklearn.metrics import explained_variance_score, mean_absolute_error, mean_squared_error, r2_score
from sklearn.decomposition import PCA
```

Spectrum reading and Loading of Dataset

```
import glob
fn = []
all_spectra = pd.DataFrame()

for f in glob.glob("./all spectra/*.txt"):
    df = pd.read_csv(f, header=None, delimiter=' ' )
    all_spectra = pd.concat([all_spectra, df], axis=1)
    fn.append(f)
Datamz = all_spectra[0] # realciones m/z
Data = all_spectra[1] # Intensidades
```

Inspección

```
Data.columns=range(Data.shape[1]) # Aquí le ponemos indices ordenados a las columnas de las intensidades
Data

Datamz.tail(10241)

#everything has signals
```

	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
12799	2862.608	2862.608	2862.608	2862.608	2862.608	2862.608	2862.608	2862.608	2862.608	2862.608	...	2862.608	2862.608	2862.608	2
12800	2862.851	2862.851	2862.851	2862.851	2862.851	2862.851	2862.851	2862.851	2862.851	2862.851	...	2862.851	2862.851	2862.851	2
12801	2863.094	2863.094	2863.094	2863.094	2863.094	2863.094	2863.094	2863.094	2863.094	2863.094	...	2863.094	2863.094	2863.094	2
12802	2863.337	2863.337	2863.337	2863.337	2863.337	2863.337	2863.337	2863.337	2863.337	2863.337	...	2863.337	2863.337	2863.337	2
12803	2863.580	2863.580	2863.580	2863.580	2863.580	2863.580	2863.580	2863.580	2863.580	2863.580	...	2863.580	2863.580	2863.580	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
23035	5887.533	5887.533	5887.533	5887.533	5887.533	5887.533	5887.533	5887.533	5887.533	5887.533	...	5887.533	5887.533	5887.533	5
23036	5887.882	5887.882	5887.882	5887.882	5887.882	5887.882	5887.882	5887.882	5887.882	5887.882	...	5887.882	5887.882	5887.882	5
23037	5888.230	5888.230	5888.230	5888.230	5888.230	5888.230	5888.230	5888.230	5888.230	5888.230	...	5888.230	5888.230	5888.230	5
23038	5888.578	5888.578	5888.578	5888.578	5888.578	5888.578	5888.578	5888.578	5888.578	5888.578	...	5888.578	5888.578	5888.578	5
23039	5888.926	5888.926	5888.926	5888.926	5888.926	5888.926	5888.926	5888.926	5888.926	5888.926	...	5888.926	5888.926	5888.926	5

10241 rows × 236 columns

m/z

```
Dsna=Datamz[0:23040]
Dsna
```

	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
0	598.375	598.375	598.375	598.375	598.375	598.375	598.375	598.375	598.375	598.375	...	598.375	598.375	598.375	
1	598.486	598.486	598.486	598.486	598.486	598.486	598.486	598.486	598.486	598.486	...	598.486	598.486	598.486	
2	598.597	598.597	598.597	598.597	598.597	598.597	598.597	598.597	598.597	598.597	...	598.597	598.597	598.597	
3	598.708	598.708	598.708	598.708	598.708	598.708	598.708	598.708	598.708	598.708	...	598.708	598.708	598.708	
4	598.819	598.819	598.819	598.819	598.819	598.819	598.819	598.819	598.819	598.819	...	598.819	598.819	598.819	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
23035	5887.533	5887.533	5887.533	5887.533	5887.533	5887.533	5887.533	5887.533	5887.533	5887.533	...	5887.533	5887.533	5887.533	5
23036	5887.882	5887.882	5887.882	5887.882	5887.882	5887.882	5887.882	5887.882	5887.882	5887.882	...	5887.882	5887.882	5887.882	5
23037	5888.230	5888.230	5888.230	5888.230	5888.230	5888.230	5888.230	5888.230	5888.230	5888.230	...	5888.230	5888.230	5888.230	5
23038	5888.578	5888.578	5888.578	5888.578	5888.578	5888.578	5888.578	5888.578	5888.578	5888.578	...	5888.578	5888.578	5888.578	5
23039	5888.926	5888.926	5888.926	5888.926	5888.926	5888.926	5888.926	5888.926	5888.926	5888.926	...	5888.926	5888.926	5888.926	5

23040 rows × 236 columns

```
D=Dsna.isnull().any()
```

```
df = D[D[0]==True]
df
```

```
Series([], dtype: bool)
```

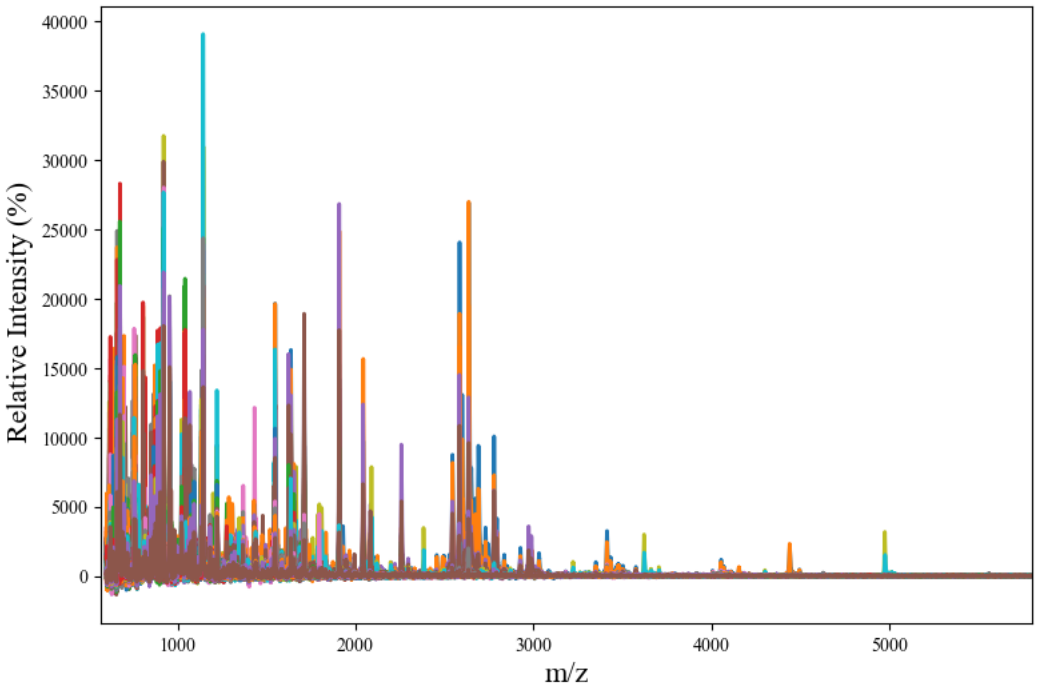
Intensities

```
Dataint=Data[0:23040]
Dataint
```

	0	1	2	3	4	5	6	7	8	9	...	226	227	228	229	230	231	232	233	234	235
0	267	234	776	823	872	1121	-6	92	923	451	...	768	163	763	383	172	166	1213	1290	202	157
1	381	415	896	879	999	1279	223	171	1018	546	...	910	183	815	464	316	331	1221	1390	202	166
2	588	535	994	939	1025	1440	422	323	1048	656	...	932	221	796	497	369	386	1228	1484	285	115
3	735	561	910	882	854	1322	626	469	818	636	...	784	196	743	340	316	325	1233	1571	146	173
4	803	432	814	856	548	1134	539	492	807	535	...	519	136	588	302	229	235	1236	1653	61	100
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
23035	9	15	-5	11	-6	-6	2	12	-6	9	...	-2	7	3	5	11	11	12	4	13	7
23036	7	14	-2	14	0	23	-2	12	2	20	...	8	3	-1	0	9	8	12	4	14	12
23037	15	4	3	14	17	5	6	6	-12	19	...	5	11	1	5	-3	-2	13	5	17	14
23038	-1	1	-4	29	2	7	18	6	-1	10	...	8	-3	-6	11	-3	-3	13	5	31	1
23039	3	8	-7	36	-2	-8	6	-3	9	16	...	7	-11	-10	8	-8	-8	14	6	32	-1

23040 rows × 236 columns

```
plt.rcParams["font.family"] = 'Times New Roman'
fig, ax = plt.subplots(figsize=(9,6))

ax.plot(Datamz, Dataint, lw=2)
ax.set_xlabel('m/z',  fontsize=16)
ax.set_ylabel(' Relative Intensity (%)', fontsize=16)
ax.set_xlim([570, 5800])
plt.savefig('espectros de masas.png')
plt.show()
```




```
C = pd.read_excel('clasificacion.xlsx')
C
```

Mostrar el resultado oculto

```
Y= C[["clasification"]]
Y1=Y.set_axis(['Clase'], axis=1)
Y1
```

Mostrar el resultado oculto

Principal Component Analysis (PCA)

Unsupervised learning

```
#data normalization
from sklearn.preprocessing import StandardScaler, Normalizer
scaler=StandardScaler()

scaler.fit(Dataint) # calculo la media para poder hacer la transformacion
x_scaled=scaler.transform(Dataint)# Ahora si, escalo los datos y los normalizo


valt=Dataint.T

valtn = preprocessing.normalize(valt, norm='l1')# Normaliza los datos
# Iniciamos el desarrollo del PCA en este caso con 10 componentes
pca=PCA(n_components=10) # Otra opción es hacer pca hasta obtener un mínimo explicado ej.: pca=PCA(.85)
pca1=pca.fit(x_scaled.T) # obtener los componentes principales
datos_pca=pca.transform(x_scaled.T) # convertimos nuestros datos con las nuevas dimensiones de PCA, scores



# Esta celda es para observar la varianza explicada con 5 componentes, se podría variar a los que se quisiese
print("shape of datos_pca", datos_pca.shape)
expl = pca.explained_variance_ratio_
print(expl)
print('suma:',sum(expl[0:10]))
#Vemos que con 5 componentes tenemos algo mas del 85% de varianza explicada
datos_pca1 = pd.DataFrame(datos_pca) # Convierte los datos pca en un DataFrame
datos_pca1=pd.concat([datos_pca1, Y1] ,axis=1) # Se agrega la columna del tipo de muestra
```

```
shape of datos_pca (236, 10)
[0.28667401 0.11023705 0.08176251 0.07480978 0.06549369 0.0445296
```

```
 0.03904263 0.03676679 0.0232983  0.02065167]
suma: 0.7832660305108794
```

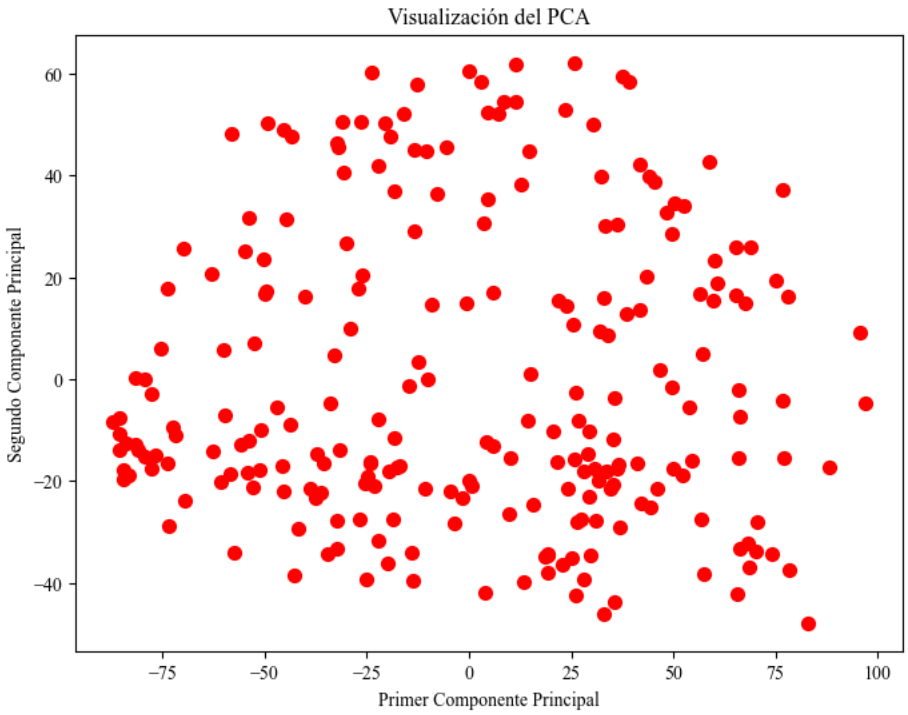
```
scores_df = pd.DataFrame(datos_pca, columns = [f"PC{i+1}" for i in range(datos_pca.shape[1])])
scores_df.head()
```

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9	PC10
0	29.410385	-23.123084	42.882846	-7.151239	-19.041101	-15.682061	24.079692	20.195048	-4.472833	-2.559562
1	-3.518009	-28.237275	27.074341	-9.170726	-13.748236	-16.431063	21.333752	4.153629	-9.482735	2.270429
2	34.556415	-21.604162	-32.645251	-15.742305	2.193066	-21.736075	-10.716339	2.518625	-9.435843	-12.287719
3	-26.753048	-27.426063	-27.410647	0.213840	-8.324547	-18.324358	-18.948341	22.685727	-12.135569	-15.713979
4	-24.098123	-16.630922	-40.284457	22.352345	-12.595149	-14.122556	12.858122	-13.102087	13.549774	16.334443

```
variancia_explicada = np.insert(expl, 0, 0)
variancia_acomulada = np.cumsum(np.round(variancia_explicada, decimals=3))
pc_df = pd.DataFrame(['']  + [f"PC{i+1}" for i in range(datos_pca.shape[1])], columns =['PC'])
variancia_explicada_df = pd.DataFrame(variancia_explicada, columns=["Variancia Explicada"])
variancia_acomulada_df = pd.DataFrame(variancia_acomulada, columns=['Variancia Acumulada']
)
df_variancia_explicada = pd.concat([pc_df, variancia_explicada_df, variancia_acomulada_df], axis =1)
df_variancia_explicada
```

	PC	Variancia Explicada	Variancia Acumulada
0		0.000000	0.000
1	PC1	0.286674	0.287
2	PC2	0.110237	0.397
3	PC3	0.081763	0.479
4	PC4	0.074810	0.554
5	PC5	0.065494	0.619
6	PC6	0.044530	0.664
7	PC7	0.039043	0.703
8	PC8	0.036767	0.740
9	PC9	0.023298	0.763
10	PC10	0.020652	0.784

```
import matplotlib.pyplot as plt

plt.figure(figsize=(8, 6))
plt.scatter(datos_pca1.iloc[:, 0], datos_pca1.iloc[:, 1], c='red', s=50)
plt.xlabel('Primer Componente Principal')
plt.ylabel('Segundo Componente Principal')
plt.title('Visualización del PCA')
plt.savefig('PCA.png');
plt.show()
```




```
from bokeh.plotting import figure
from bokeh.models import ColumnDataSource, HoverTool
from bokeh.io import show

TOOLS = "hover,save,pan,box_zoom,reset,wheel_zoom"

colormap = {'A': 'yellow', 'N':'blue' , 'S': 'green', }
colors = [colormap[x] for x in Y1['Clase']]

#datos_pca1 = pd.DataFrame(data={'PCA1': X_pca[:,0], 'PCA2': X_pca[:,1]})

p = figure(title='PCA', width=600, height=400,
           x_axis_label='PCA 1', y_axis_label='PCA 2', toolbar_location="above", tools=TOOLS)
p.grid.grid_line_alpha=0.3
p.circle(datos_pca1[0], datos_pca1[1], color=colors, size=10, alpha=0.6)

show(p)
```

Supervised Learning

Support vector machine (SVM)

supervised learning

Model 80:20

```
#Aquí se convierten las variables en números ordinales A es 0, N es 1 y S es 2

from sklearn.preprocessing import OrdinalEncoder
from sklearn.preprocessing import OneHotEncoder

encoder = OrdinalEncoder()
encoder.fit(Y1[['Clase']])
Y1_code = encoder.transform (Y1[['Clase']])
Y1_code = pd.DataFrame(Y1_code)
Y1_code
```

Mostrar el resultado oculto

Separacion de datos 80% de entrenamiento y 20% de prueba

```
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.model_selection import GridSearchCV #permite realizar una búsqueda exhaustiva en un conjunto de parámetros especificados por el
from sklearn.metrics import accuracy_score #permite calcular la precisión de un modelo de clasificación

X_tr2, X_te2, y_tr2, y_te2 = train_test_split(x_scaled.T, Y1_code.values.ravel(), train_size   = 0.8, random_state = 12, shuffle = True)

modelo2 = SVC(C = 40, kernel = 'rbf', random_state=200)
modelo2.fit(X_tr2, y_tr2)
```

```
SVC(C=40, random_state=200)
```

```
# Predictions

predicciones = modelo2.predict(X_te2)
predicciones
```

```
array([0., 0., 1., 1., 0., 2., 2., 2., 1., 2., 2., 0., 2., 1., 0., 1., 2.,
       0., 2., 0., 2., 2., 0., 2., 1., 2., 0., 1., 0., 1., 1., 2., 0., 2.,
       2., 0., 1., 2., 1., 1., 0., 2., 0., 2., 0., 2., 2., 0.])
```

Metrics for the confusion matrix

```
from sklearn.linear_model import LinearRegression

# Fit the model to the training data
clf = LinearRegression().fit(X_tr2, y_tr2)

# Print the weight coefficients
print(clf.coef_)
```

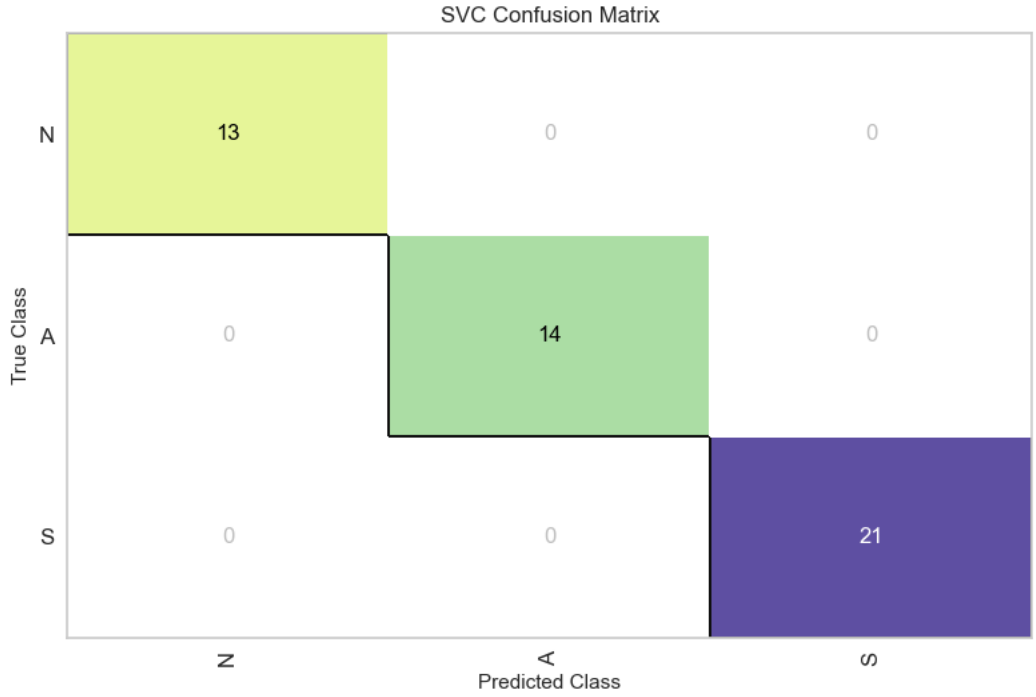
```
[ 1.06573079e-03 -8.36590284e-05 -1.10906869e-03 ... -2.13317066e-04
 -3.26745168e-04 -5.84073552e-04]
```

```
def opt_svc(X, y, xt, rs):

    # Defining PLS and the number of components

    svc = SVC(C = 40, kernel = 'rbf', random_state=123)
    svc.fit(X, y)
    y_pred = svc.predict(x_test)

    # Metric calculation

    ex = accuracy_score(y_true = y_test, y_pred = y_pred, normalize = True)

    return (y_pred, ex)


# Testing with 30 components

exs = []
yps = []
rss = []
arr_rs = np.arange(1, 200)

for rs in arr_rs:
    x_train, x_test, y_train, y_test = train_test_split(x_scaled.T, Y1_code.values.ravel(), train_size   = 0.8, random_state = rs, shuffle =
    y_pred, ex = opt_svc(x_train, y_train, x_test, rs)
    rss.append(rs)
    yps.append(y_pred)
    exs.append(ex)
```

```
x_train, x_test, y_train, y_test = train_test_split(x_scaled.T, Y1_code.values.ravel(), train_size   = 0.8, random_state = rss[np.argmax(exs

# Creación del modelo SVM

svc1 = SVC(C = 40, kernel = 'rbf', random_state=123)
svc1.fit(x_train, y_train)
y_pred_svc = svc1.predict(x_test)
ex_svc = accuracy_score(y_true = y_test, y_pred = y_pred_svc, normalize = True)
ex_svc
```

```
1.0
```

```
from yellowbrick.classifier import confusion_matrix
from tabulate import tabulate as tabulate_fn

confusion_matrix(
    SVC(C = 40, kernel = 'rbf', random_state=123),
    x_train, y_train, x_test, y_test,
    classes=['N', 'A', 'S'], cmap='Spectral'
)

plt.savefig('CM_SVC.png');
plt.tight_layout();

from sklearn import metrics
print(metrics.classification_report(y_test,y_pred_svc, digits = 4))
```




```
              precision    recall  f1-score   support

         0.0     1.0000    1.0000    1.0000        13
         1.0     1.0000    1.0000    1.0000        14
         2.0     1.0000    1.0000    1.0000        21

    accuracy                         1.0000        48
   macro avg     1.0000    1.0000    1.0000        48
weighted avg     1.0000    1.0000    1.0000        48

<Figure size 800x550 with 0 Axes>
```

```
np.max(exs), rss[np.argmax(exs)]
```

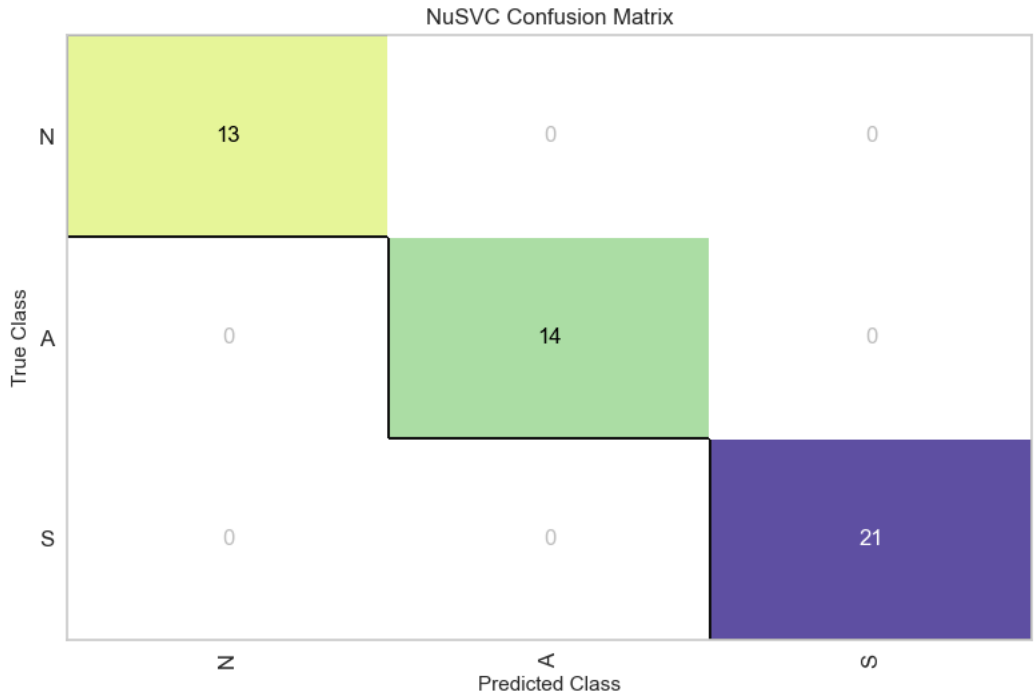
```
(1.0, 42)
```

Nu-Support Vector Classification (NuSVM)

```
from sklearn.svm import NuSVC

X_tr2, X_te2, y_tr2, y_te2 = train_test_split(x_scaled.T, Y1_code.values.ravel(), train_size = 0.8, random_state = 12, shuffle = True)

modeloNu = NuSVC(nu=0.1, gamma='auto')
modeloNu.fit(X_tr2, y_tr2)
```

```
NuSVC(gamma='auto', nu=0.1)
```

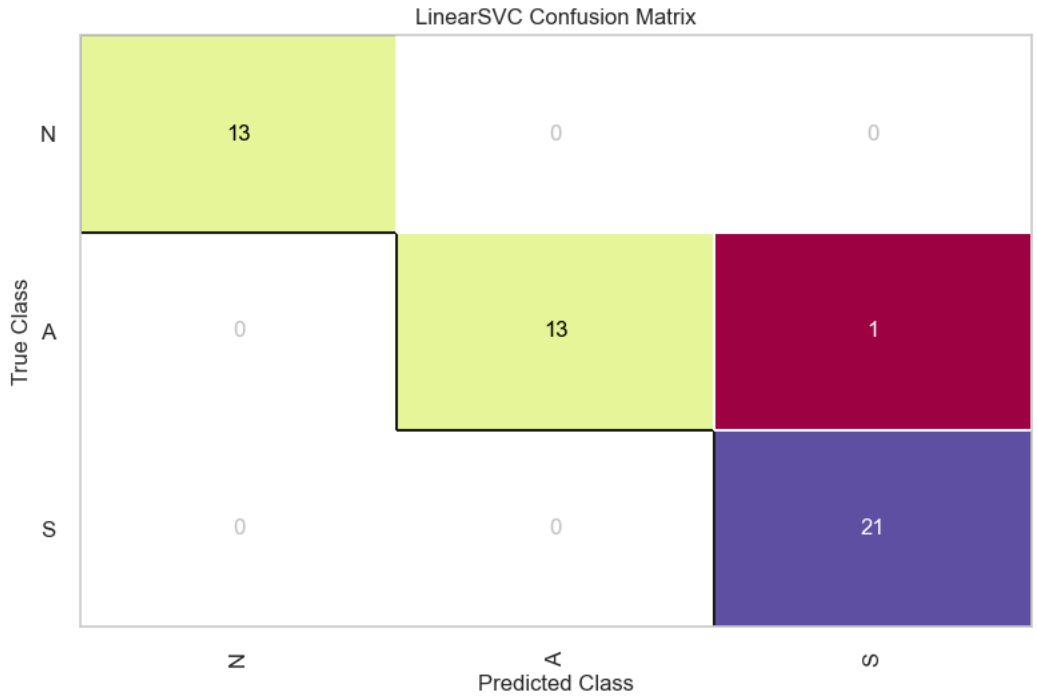
```
from yellowbrick.classifier import confusion_matrix
confusion_matrix(
   NuSVC(nu=0.1, gamma='auto'),
    x_train, y_train, x_test, y_test,
    classes=['N', 'A', 'S'], cmap='Spectral'
)
plt.savefig('CM_NuSVC.png')
plt.tight_layout();

#Predicción con base a las variables ya entrenadas

y_predNu = modeloNu.predict(x_test)

#Generación de soporte

from sklearn import metrics
print(metrics.classification_report(y_test, y_predNu, digits = 4))
```




```
              precision    recall  f1-score   support

         0.0     1.0000    1.0000    1.0000        13
         1.0     1.0000    1.0000    1.0000        14
         2.0     1.0000    1.0000    1.0000        21

    accuracy                         1.0000        48
   macro avg     1.0000    1.0000    1.0000        48
weighted avg     1.0000    1.0000    1.0000        48

<Figure size 800x550 with 0 Axes>
```

Linear Support Vector Classification

```
from sklearn.svm import LinearSVC


x_train, x_test, y_train, y_test = train_test_split(x_scaled.T, Y1_code.values.ravel(), train_size   = 0.8, random_state = rss[np.argmax(exs

# Creación del modelo linearSVM

LiSVC = LinearSVC( max_iter=10000)
LiSVC.fit(x_train, y_train)
#y_pred_svc = s
#svc1.predict(x_test)
Lisvc = accuracy_score(y_true = y_test, y_pred = y_pred_svc, normalize = True)



# Predictions
p = LiSVC.predict(x_test)


from yellowbrick.classifier import confusion_matrix
confusion_matrix(
    LinearSVC (max_iter=10000), x_train, y_train, x_test, y_test,
    classes=['N', 'A', 'S'], cmap='Spectral'
)
plt.savefig('CM_LinearSVC.png')
plt.tight_layout();
```




```
<Figure size 800x550 with 0 Axes>
```

```
#Generación de soporte

print(metrics.classification_report(y_test, p, digits = 4))
```

```
                  precision    recall  f1-score   support

             0.0     1.0000    1.0000    1.0000        13
             1.0     1.0000    0.9286    0.9630        14
             2.0     0.9545    1.0000    0.9767        21

        accuracy                         0.9792        48
       macro avg     0.9848    0.9762    0.9799        48
    weighted avg     0.9801    0.9792    0.9790        48
```

Neural networks (Redes Neuronales)

```
from sklearn.neural_network import MLPClassifier
x_train, x_test, y_train, y_test = train_test_split(x_scaled.T, Y1_code.values.ravel(), train_size=0.8, random_state=rss[np.argmax(exs)], sh
ann = MLPClassifier(hidden_layer_sizes=(20, 20), learning_rate_init=0.01, solver = 'lbfgs', max_iter = 5000, random_state = 123)
ann.fit(X=x_train, y=y_train)
```

```
MLPClassifier(hidden_layer_sizes=(20, 20), learning_rate_init=0.01,
              max_iter=5000, random_state=123, solver='lbfgs')
```

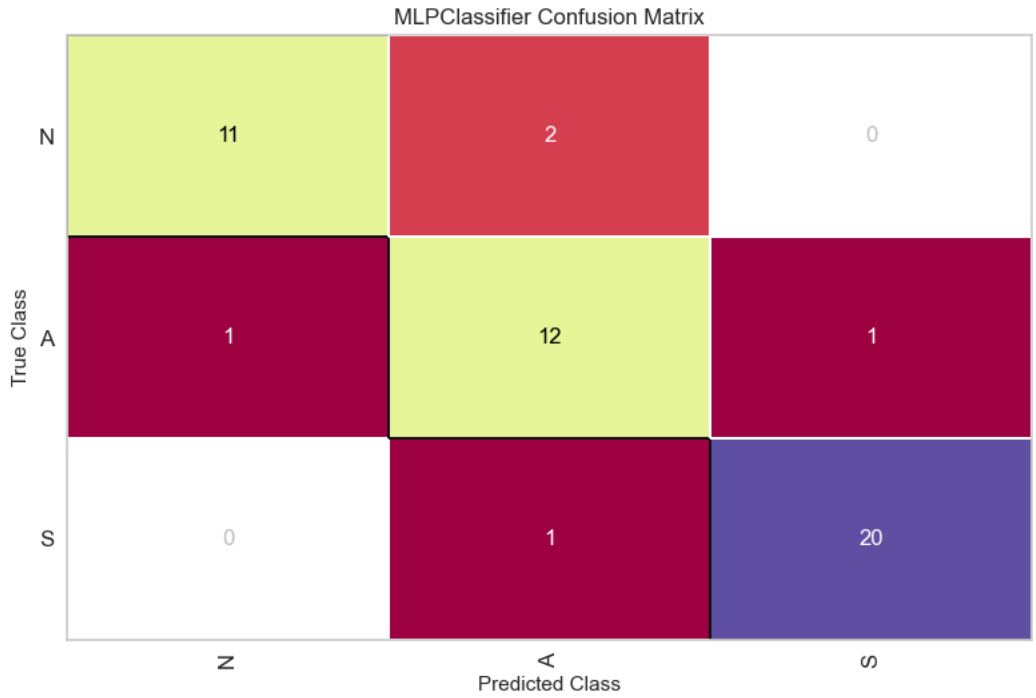
```
#Predicción con base a las variables ya entrenadas

y_pred = ann.predict(x_test)
score = ann.score(x_test, y_test)
score
```

```
0.8958333333333334
```

```
#Matriz de confusion para saber donde acierta y donde falla

from sklearn.metrics import confusion_matrix, precision_score, recall_score, f1_score, cohen_kappa_score
confusion_matrix(y_test, y_pred)
```

```
array([[11,  2,  0],
       [ 1, 12,  1],
       [ 0,  1, 20]], dtype=int64)
```

```
from yellowbrick.classifier import confusion_matrix

confusion_matrix(
    MLPClassifier(hidden_layer_sizes=(20, 20), learning_rate_init=0.01, solver = 'lbfgs', max_iter = 5000, random_state = 123),
    x_train, y_train, x_test, y_test,
    classes=['N', 'A', 'S'], cmap='Spectral'
)
plt.savefig('CM_MLPclassifier_neural networks.png')
plt.tight_layout();

#Generación de soporte

from sklearn import metrics
print(metrics.classification_report(y_test, y_pred, digits = 4))
```

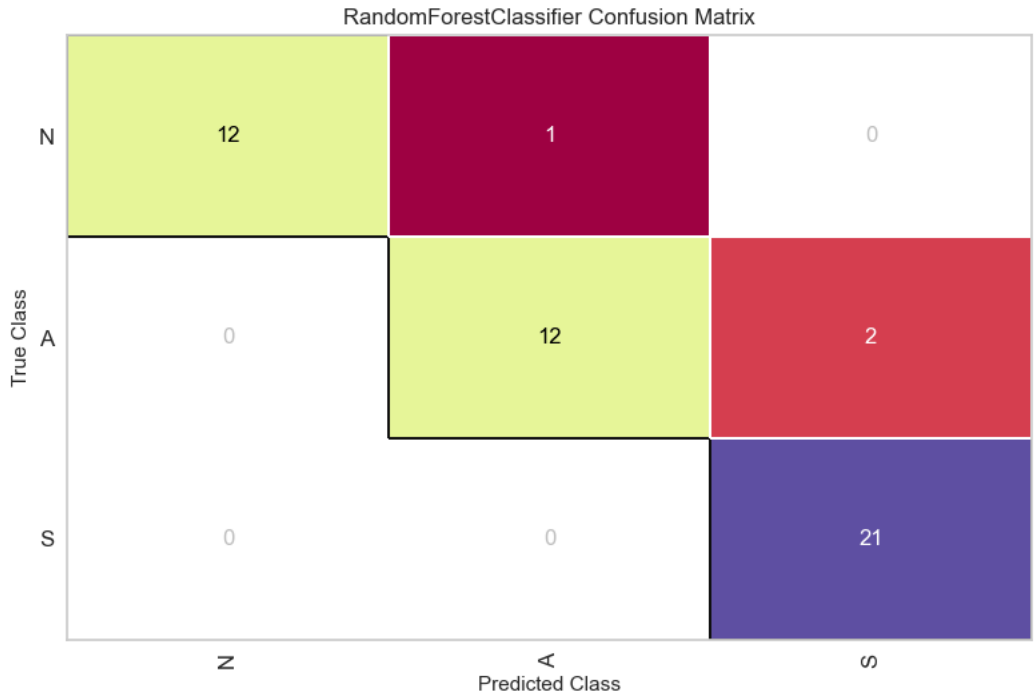



```
              precision    recall  f1-score   support

         0.0     0.9167    0.8462    0.8800        13
         1.0     0.8000    0.8571    0.8276        14
         2.0     0.9524    0.9524    0.9524        21

    accuracy                         0.8958        48
   macro avg     0.8897    0.8852    0.8867        48
weighted avg     0.8983    0.8958    0.8964        48

<Figure size 800x550 with 0 Axes>
```

Classifiers

Random Forest

```
from yellowbrick.classifier import confusion_matrix

from sklearn.ensemble import RandomForestClassifier

rfc = RandomForestClassifier(random_state=90)
rfc.fit(x_train, y_train)

confusion_matrix(
   RandomForestClassifier(random_state = 90),
    x_train, y_train, x_test, y_test,
    classes=['N', 'A', 'S'], cmap='Spectral'
)

plt.savefig('CM_RandomForest.png')
plt.tight_layout();

#Predicción con base a las variables ya entrenadas

y_pred1 = rfc.predict(x_test)


#Generación de soporte

from sklearn import metrics
print(metrics.classification_report(y_test, y_pred1, digits = 4))
```



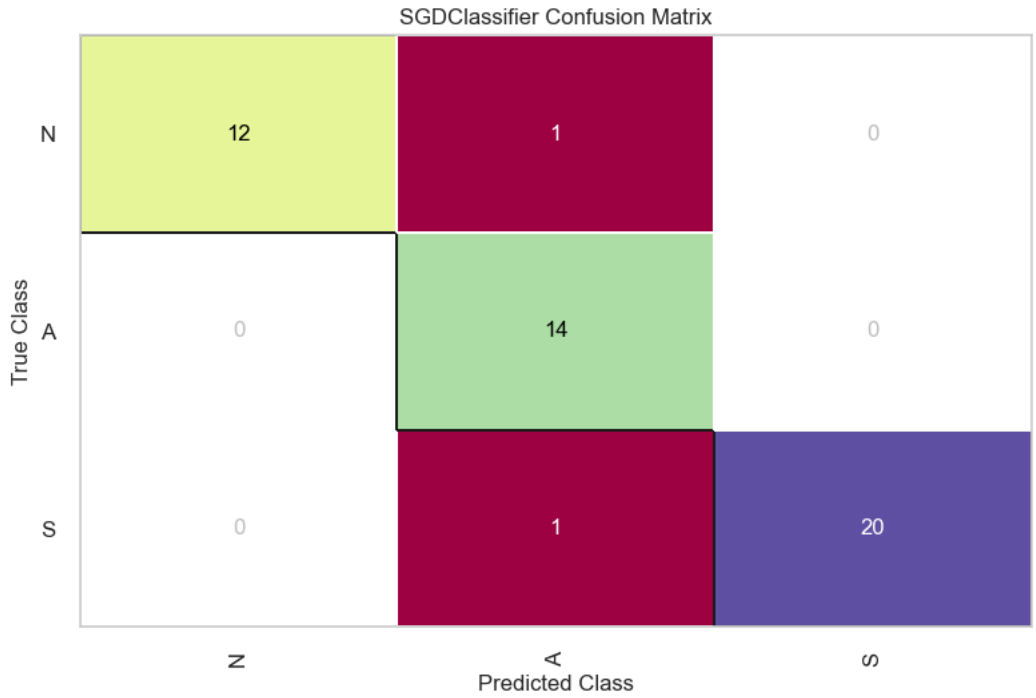

```
              precision    recall  f1-score   support

         0.0     1.0000    0.9231    0.9600        13
         1.0     0.9231    0.8571    0.8889        14
         2.0     0.9130    1.0000    0.9545        21

    accuracy                         0.9375        48
   macro avg     0.9454    0.9267    0.9345        48
weighted avg     0.9395    0.9375    0.9369        48

<Figure size 800x550 with 0 Axes>
```

Stochastic Gradient Descent-Classifier (SGDClassifier)

```
from sklearn.linear_model import SGDClassifier

sgdc = SGDClassifier(max_iter=100, tol=1e-3)
sgdc.fit(x_train, y_train)

y_pred_sgdc = sgdc.predict(x_test)
score_sgdc = sgdc.score(x_test, y_test)
score_sgdc
```

```
0.9583333333333334
```

```
from yellowbrick.classifier import confusion_matrix

visualizer = confusion_matrix(
    sgdc, x_train, y_train, x_test, y_test,
    classes=['N', 'A', 'S'], cmap='Spectral'
)
visualizer.fit(x_train, y_train)
visualizer.score(x_test, y_test)

plt.savefig('sdg.png')
plt.tight_layout();
plt.show()

#Generación de soporte

from sklearn import metrics
print(metrics.classification_report(y_test, y_pred_sgdc, digits = 4))
```




```
<Figure size 800x550 with 0 Axes>
              precision    recall  f1-score   support

         0.0     1.0000    0.9231    0.9600        13
         1.0     0.8750    1.0000    0.9333        14
         2.0     1.0000    0.9524    0.9756        21

    accuracy                         0.9583        48
   macro avg     0.9583    0.9585    0.9563        48
weighted avg     0.9635    0.9583    0.9591        48
```

K-Nearest Neighbors(KNN)

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.inspection import permutation_importance


knc = KNeighborsClassifier()
knc.fit(x_train, y_train)
```

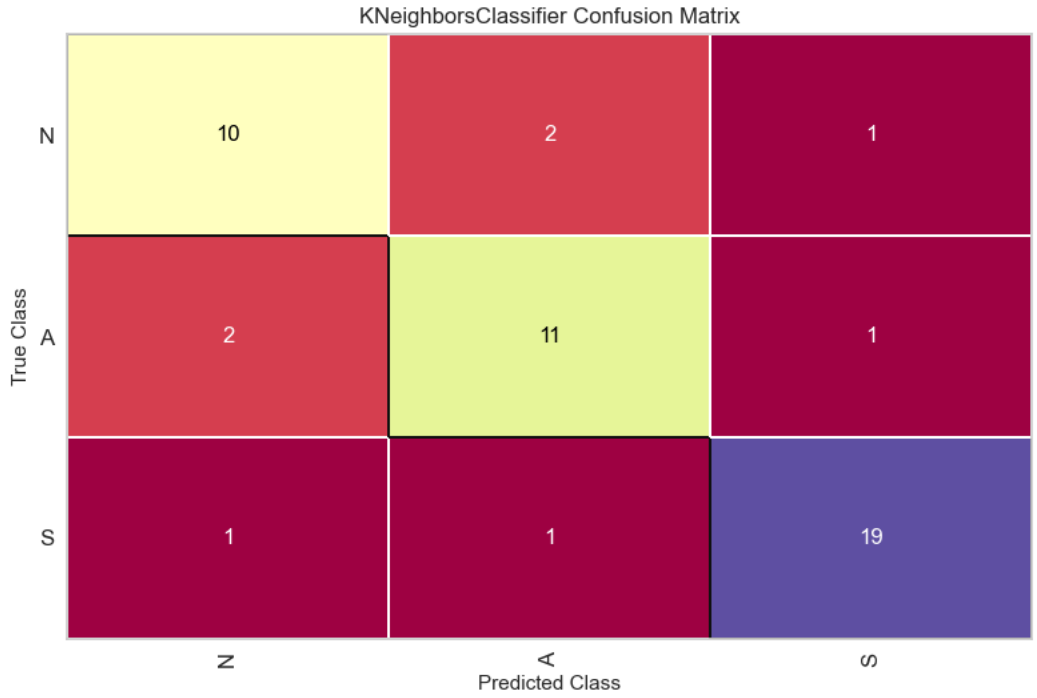
```
KNeighborsClassifier()
```

```
y_pred3 = knc.predict(x_test)
score_knc = knc.score(x_test, y_test)
score_knc
```

```
C:\Users\Yenny\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228: FutureWarning: Unlike other reduction functions (e.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
C:\Users\Yenny\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228: FutureWarning: Unlike other reduction functions (e.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
0.8333333333333334
```

```
from yellowbrick.classifier import confusion_matrix

visualizer = confusion_matrix(
    knc, x_train, y_train, x_test, y_test,
    classes=['N', 'A', 'S'], cmap='Spectral'
)
visualizer.fit(x_train, y_train)
visualizer.score(x_test, y_test)
plt.tight_layout();
plt.show()
plt.savefig('CM_KNN.png')

#Generación de soporte

from sklearn import metrics
print(metrics.classification_report(y_test, y_pred3, digits = 4))
```

```
C:\Users\Yenny\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228: FutureWarning: Unlike other reduction functions (e.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
C:\Users\Yenny\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228: FutureWarning: Unlike other reduction functions (e.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```




```
C:\Users\Yenny\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228: FutureWarning: Unlike other reduction functions (e.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
C:\Users\Yenny\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228: FutureWarning: Unlike other reduction functions (e.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
<Figure size 800x550 with 0 Axes>
              precision    recall  f1-score   support

         0.0     0.7692    0.7692    0.7692        13
         1.0     0.7857    0.7857    0.7857        14
         2.0     0.9048    0.9048    0.9048        21

    accuracy                         0.8333        48
   macro avg     0.8199    0.8199    0.8199        48
weighted avg     0.8333    0.8333    0.8333        48

<Figure size 800x550 with 0 Axes>
```

Logistic Regression

```
from sklearn.linear_model import LogisticRegression

lr = LogisticRegression(solver='liblinear')
lr.fit(x_train, y_train)
```

```
LogisticRegression(solver='liblinear')
```

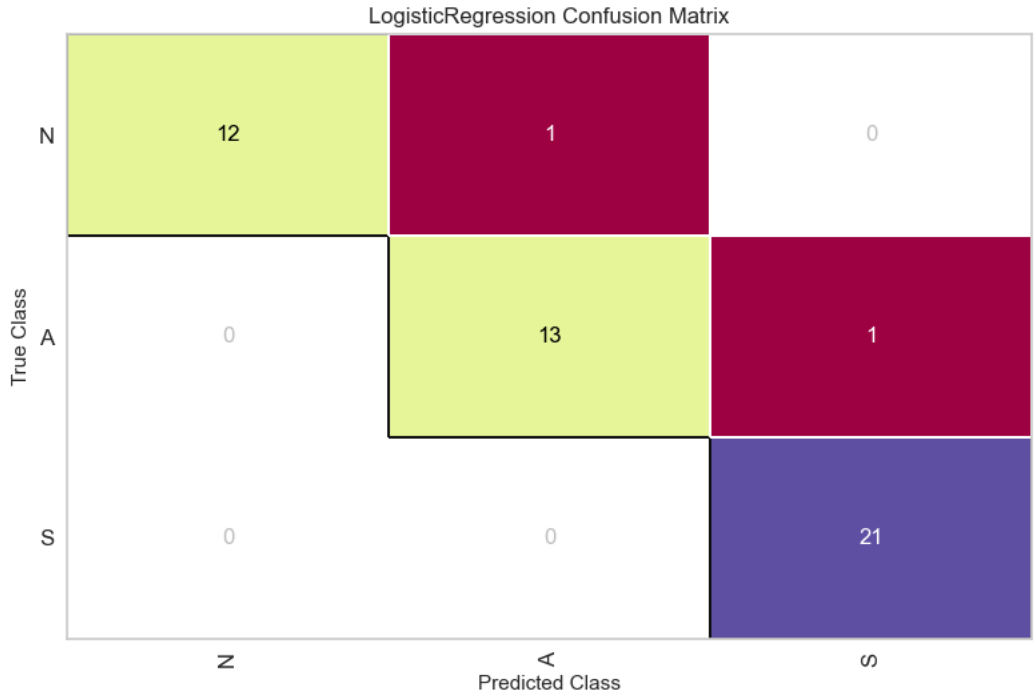
```
ypred_lr=lr.predict(x_test)
lr.score(x_test, y_test)
#print(classification_report(y_test,ypred))
```

```
0.9583333333333334
```

```
from yellowbrick.classifier import confusion_matrix

visualizer = confusion_matrix(
    lr, x_train, y_train, x_test, y_test,
    classes=['N', 'A', 'S'], cmap='Spectral'
)
```

```
visualizer.fit(x_train, y_train)
visualizer.score(x_test, y_test)

plt.savefig('CM_LogisticRegression.png')
plt.tight_layout();
plt.show()
```

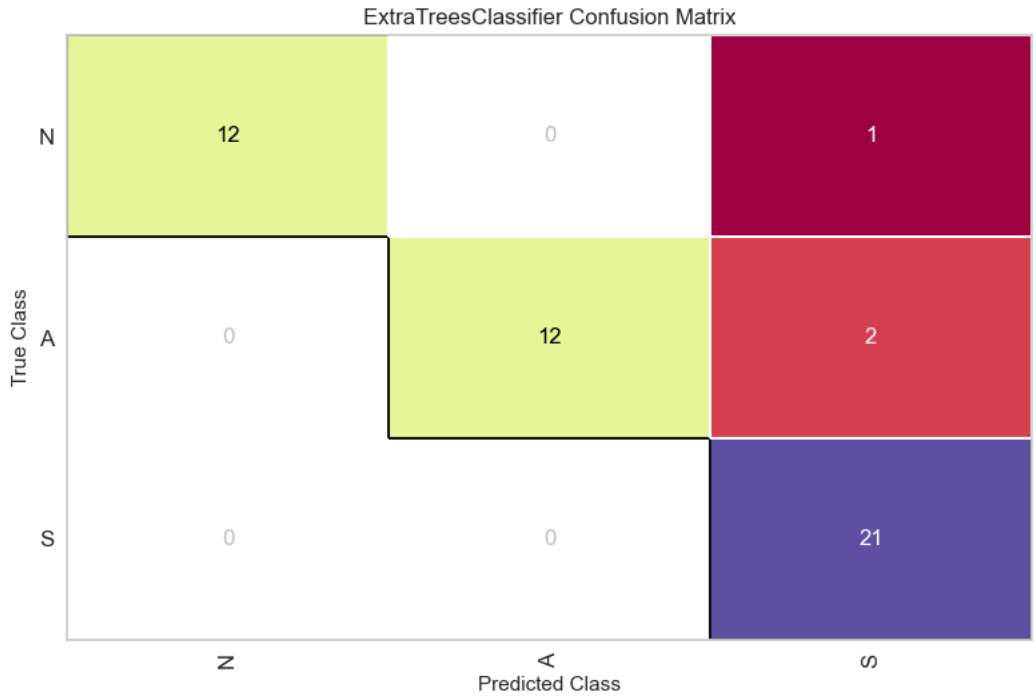
```
#Generación de soporte

from sklearn import metrics
print(metrics.classification_report(y_test, ypred_lr, digits = 4))
```




```
<Figure size 800x550 with 0 Axes>
              precision    recall  f1-score   support

         0.0     1.0000    0.9231    0.9600        13
         1.0     0.9286    0.9286    0.9286        14
         2.0     0.9545    1.0000    0.9767        21

    accuracy                         0.9583        48
   macro avg     0.9610    0.9505    0.9551        48
weighted avg     0.9593    0.9583    0.9582        48
```

ExtraTrees Classifier

```
from sklearn.ensemble import ExtraTreesClassifier

T = ExtraTreesClassifier(n_estimators=300)
T.fit(x_train, y_train)

ypred_T=T.predict(x_test)

visualizer = confusion_matrix(
    T, x_train, y_train, x_test, y_test,
    classes=['N', 'A', 'S'], cmap='Spectral'
)
visualizer.fit(x_train, y_train)
visualizer.score(x_test, y_test)
plt.tight_layout();
plt.show()
plt.savefig('CM_ExtraTrees.png')

#Generación de soporte

from sklearn import metrics
print(metrics.classification_report(y_test, ypred_T, digits = 4))
```




```
<Figure size 800x550 with 0 Axes>
              precision    recall  f1-score   support

         0.0     1.0000    0.9231    0.9600        13
         1.0     1.0000    0.8571    0.9231        14
         2.0     0.8750    1.0000    0.9333        21

    accuracy                         0.9375        48
   macro avg     0.9583    0.9267    0.9388        48
weighted avg     0.9453    0.9375    0.9376        48

<Figure size 800x550 with 0 Axes>
```

Comparation of the different Models and Quantifiers

```
import matplotlib.pyplot as plt
import seaborn as sns
from xgboost import XGBClassifier
```

```
# Precisión de los modelos

model_names = ['SVM', 'NuSVC','LinearSVC', 'SGD','LogisticRegression','Random Forest','ExtraTrees'  ,'MLP',  'KNeighbors' ]
accuracies = [1, 1, 0.979, 0.958, 0.958 , 0.937, 0.937, 0.895, 0.833,]

# Establecer un estilo de gráfico "seaborn"
sns.set(style="ticks")


plt.figure(figsize=(20, 12))
bars = sns.barplot(x=model_names, y=accuracies, palette="plasma")
plt.xlabel('Modelo', fontweight='bold', fontsize=16)
plt.ylabel('Exactitud', fontweight='bold', fontsize=16)

plt.ylim(0, 1)

# Mostrar las etiquetas de los valores en las barras
for bar, accuracy in zip(bars.patches, accuracies):
    plt.text(bar.get_x() + bar.get_width() / 2, accuracy + 0.02, f'{accuracy:.2f}', ha='center', va='bottom', fontsize=20)


for spine in plt.gca().spines.values():
    spine.set_visible(False)
plt.savefig('model_accuracies.jpg', dpi=300, bbox_inches='tight')
plt.show()
```