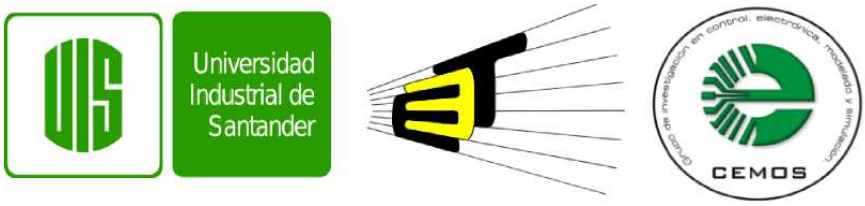
CLASIFICACIÓN DE PÉPTIDOS A PARTIR DE DIFERENTES MÉTODOS Y ESTRATEGIAS DE ENSAMBLE DE CLASIFICADORES EN CONDICIÓN DESBALANCEADA

Carlos Mauricio Lastre Domínguez
Ingeniero Electrónico



Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones
Facultad Ingenierías Fisicomecánicas
Universidad Industrial de Santander
Bucaramanga, 2016

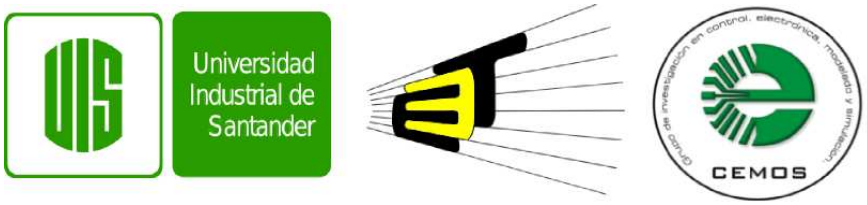
CLASIFICACIÓN DE PÉPTIDOS A PARTIR DE DIFERENTES MÉTODOS Y ESTRATEGIAS DE ENSAMBLE DE CLASIFICADORES EN CONDICIÓN DESBALANCEADA

Carlos Mauricio Lastre Domínguez
Ingeniero Electrónico

Trabajo de investigación presentado como requerimiento parcial para optar al título de:
Magister en Ingeniería Electrónica

Director
Daniel Alfonso Sierra Bueno
Ph.D. en Ingeniería Biomédica

Codirectora
Nydia Paola Rondón Villarreal
Ph.D(c). en Ingeniería Electrónica



Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones
Facultad Ingenierías Fisicomecánicas
Universidad Industrial de Santander
Bucaramanga, 2016

Dedicatoria

Dedicado: A mis padres, hermanos, sobrinos y mis futuros hijos.

Agradecimientos

Primero que todo agradecerle a Dios.

Agradecerle a mi Madre y Padre que incansablemente me prepararon como un buen guerrero para la vida. También me enseñaron a no tener miedo a las adversidades que se presentan y lo más importante luchar por lo que se quiere.

A mis hermanos por su gran apoyo y buenos concejos.

A la señora Emilce y sus hijas Lorena y Maria Amelia por su gran acogida en la ciudad de Bucaramanga y hacerme sentir en familia.

A un gran académico destacado, el profesor Daniel Alfonso Sierra por su gran bienvenida, dirección, compromiso, exigencia, atención y sabios concejos durante mi estancia en la Escuela de Ingenierías y el grupo CEMOS.

A una gran mujer investigadora comprometida con la ciencia, también una gran codirectora Paola Rondón por su compromiso y exigencia en entregar y presentar un buen trabajo.

A los profesores que hicieron parte del programa de Maestría en Ingeniería Electrónica.

A la Universidad Industrial de Santander.

Finalmente, a amigos y compañeros de estudio por sus buenos concejos y recomendaciones.

"Nuestra recompensa se encuentra en el esfuerzo y no en el resultado. Un esfuerzo total es una victoria completa"

Mahatma Gandhi.
1869-1948. Político y pensador indio.

Resumen

TÍTULO: CLASIFICACIÓN DE PÉPTIDOS A PARTIR DE DIFERENTES MÉTODOS Y ESTRATEGIAS DE ENSAMBLE DE CLASIFICADORES EN CONDICIÓN DESBALANCEADA [1]

AUTOR: CARLOS MAURICIO LASTRE DOMÍNGUEZ [2]

PALABRAS CLAVE: Ensamble de Clasificadores, Clasificación de Péptidos, Reglas de Combinación, Estrategias a Nivel de Datos y de Algoritmos.

DESCRIPCIÓN: El descubrimiento o síntesis de péptidos con propiedades antimicrobianas es una gran alternativa para combatir las bacterias multirresistentes. Sin embargo, existen limitaciones a la hora de encontrar estos péptidos. Por lo anterior, desde la bioinformática se trabaja en el uso de técnicas de clasificación para predecir la posible presencia de actividad antimicrobiana en un péptido candidato. Un reto asociado a estos estudios es que la cantidad de muestras de la clase antimicrobiana es poca ante la cantidad de muestras no antimicrobianas. En contraparte, en el caso de los pétidos antibacterianos son más los péptidos con características antibacterianas que los péptidos con características no antibacterianas. En la literarura se pueden encontrar diferentes estrategias y métodos de clasificación que tratan el problema del desbalanceo. En el presente trabajo se aplican metodologías de ensamble con estrategias a nivel de algoritmos y a nivel de datos buscando solucionar el problema del desbalanceo utilizando cinco reglas de combinación: media, máximo, mínimo, producto y mediana. Los péptidos utilizados en este trabajo fueron extraídos de la base de datos APD. Además, la evaluación de desempeño del ensamble con las diferentes estrategias de combinación se desarrolla a partir del análisis de las curvas ROC. En conclusión, nuestros resultados consideran que se debe estudiar en profundidad los algoritmos de clasificación de manera individual y explorar más las características de los datos.

[1] Trabajo de grado de maestría.

[2] Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones (E3T). Director: Daniel Sierra Bueno. Ph.D., Codirectora: Nydia Paola Rondón Ph.D.

Abstract

TITLE:CLASSIFICATION OF PEPTIDES FROM DIFFERENT METHODS AND STRATEGIES OF ENSEMBLE OF CLASSIFIERS WITH IMBALANCED DATA [3]

AUTHOR: CARLOS MAURICIO LASTRE DOMÍNGUEZ [4]

KEYWORDS: Algorithms level ensemble, classification of peptides, combination rules, ensemble of classifiers, data level ensemble, learning algorithms.

DESCRIPTION: The discovery and synthesis of peptides with antimicrobial properties is a promising alternative to fight against multi-resistant bacteria. There are multiple studies that deal with the classification of peptides according with their probability to possess antimicrobial activity. One of the challenges in these classification processes is related with the amount of available data. For the case of antibacterial peptides classifiers, the size of the positive class is much bigger than the negative class. In this work, we propose two strategies to deal with the imbalance situation of the data by using ensembles. The first one is based on algorithm modifications and the second one with data management. For each strategy we used five combination rules. The performance of the ensembles was calculated using the area under the ROC curve (AUC). Our results suggest that care must be taken with ensembles and that individual classifiers must be studied in-depth.

[3] Master thesis: Master on electronic engineering.

[4] Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones (E3T). Advisor: Daniel Sierra Bueno. Ph.D., Co-advisor: Nydia Paola Rondón Ph.D.(c).

Contenido

Lista de Figuras

Lista de Tablas

Introducción

La humanidad desde el pasado ha estado afectada por enfermedades infecciosas causadas por hongos y bacterias. Sin embargo, el descubrimiento de drogas como la penicilina produjo una protección fuerte para combatir agentes patógenos. A partir de la penicilina, otras variedades de moléculas y diferentes tipos de antibióticos han sido desarrollados. En la actualidad muchos de estos agentes han perdido eficacia y vienen a ser inútiles ante cepas de bacterias resistentes [2]. Por lo tanto, los péptidos antimicrobianos han surgido como alternativa terapéutica para tratar infecciones causadas por bacterias resistentes [3]. Estos péptidos tienen un gran potencial para aplicaciones terapéuticas futuras como: Agentes antinfecciosos individuales, combinación con antibióticos o antivirales convencionales para promover los efectos aditivos o sinérgicos, agentes inmunoestimulantes que mejoran la inmunidad innata natural y agentes neutralizantes de endotoxina para prevenir las complicaciones potencialmente mortales asociadas con factores de virulencia bacterianos que pueden causar choque séptico [4].

La diversidad estructural de los péptidos naturales proporciona un gran interés que arranca como punto de partida para la producción de péptidos artificiales y derivados con actividades biológicas más potentes y deseables para aplicaciones clínicas y comerciales. En el caso de los péptidos antimicrobianos se quiere encontrar nuevas estrategias de protección ante cepas de bacterias resistentes a partir del estudio de sus estructuras y propiedades antimicrobianas. Por lo tanto, se desarrollan aproximaciones mediante herramientas computacionales e informáticas para analizar las relaciones entre estructura y función de los péptidos las cuales permiten predecir la propiedad del péptido y diseñar nuevos péptidos con propiedades antimicrobianas fuertes [5]. Entre esas aproximaciones computacionales e informáticas, se cree que los métodos de clasificación por medio de ensamble son los más apropiados.

El ensamble es un sistema conformado por un conjunto de clasificadores, el cual es diseñado para proporcionar un desempeño con mayor precisión. [6]. Los ensambles son métodos de clasificación que están entre las aproximaciones más exitosas para tratar problemas de conjuntos desbalanceados [7]. Entre esos problemas se encuentran la información relacionada con la gestión de riesgos, el diagnóstico médico, la detección de

fraudes, el análisis de imágenes satelitales, entre otras [8], [9]. En los conjuntos desbalanceados el porcentaje de las muestras de una clase es mucho mayor que el de la otra y en la mayoría de casos los ejemplos que hacen parte de la clase minoritaria son los más importantes [10]. Estos conjuntos constituyen un problema presente en las bases de datos biológicos donde el conjunto de la clase positiva es de menor presencia en comparación con el conjunto de la clase negativa [11], por consiguiente el problema ocurre debido a la falta de datos o por grandes gastos implicados en la recolección de los mismos [12]. En el caso de los péptidos antimicrobianos, se presenta el problema de conjuntos desbalanceados con dos clases, donde la clase antimicrobiana es pequeña con respecto a la clase no antimicrobiana. Para el caso de los péptidos antibacterianos la mayoría de muestras la posee la clase positiva.

Uno de los principales retos del problema de los conjuntos desbalanceados es que las clases pequeñas son frecuentemente más útiles, pero la clasificación estándar tiende a favorecer las clases con más muestras [13], [14], [15]. Esto es debido a que muchos clasificadores asumen una distribución uniforme entre clases y un costo de clasificación errónea igual [16]. Por lo tanto, la clase minoritaria, la cual se quiere predecir con exactitud, está mal clasificada con un sistema de clasificación estándar, aunque el clasificador logre una alta precisión [17], [18]. La mayor parte de los enfoques de aprendizaje están diseñados para maximizar la medida general de la precisión, la cual es independiente de la distribución de la clase. Se produce una tendencia hacia la clase mayoritaria dando como resultado una menor sensibilidad para la identificación de las muestras de la clase minoritaria [19]. En los problemas de clasificación en conjuntos desbalanceados no se puede confiar en un clasificador que obtenga una alta precisión [7]. Otro inconveniente es que el aumentar la precisión en la clasificación de una clase puede afectar la clasificación de la otra clase, bajando su precisión [9].

Existen estrategias que buscan dar solución al problema de los conjuntos desbalanceados como: Estrategia a nivel de datos, Estrategia a nivel de algoritmos y métodos de sensibilidad al costo (*Cost-sensitive methods*) [20]. La aproximación a nivel de datos consiste en rebalancear la distribución de las clases remuestreando el espacio de los datos [21], [22]. La aproximación a nivel de algoritmos consiste en crear o modificar algoritmos existentes para favorecer el aprendizaje hacia la clase con menos muestras [20], [23]. Los métodos de sensibilidad al costo consisten en considerar el costo asociado a la clasificación errónea de las muestras [24]. Los métodos de sensibilidad al costo se desarrollaron principalmente considerando que los costos de clasificación errónea eran diferentes para distintas clases [25], [26]. Este método ha sido considerado como una buena solución para el problema del desbalanceo de clases [27]. Además, *Maloof* [28], indica que el aprendizaje a partir de los conjuntos de datos desbalanceados y el aprendizaje con costos de clasificación errónea diferentes pueden ser manejados de la misma manera. Existen diferentes métodos combinados con las estrategias anterior-

mente mencionadas; entre los más comunes se encuentran *Bagging, Boosting* y *Random Forest*. Sin embargo, hay otro método llamado ensamble de múltiples clasificadores [29]. Este método busca aprovechar la información complementaria que puede aportar cada clasificador aumentando la precisión en la clasificación.

Este trabajo propone la clasificación de péptidos con características antibacterianas a partir de diferentes estrategias y metodologías de ensamble en condición desbalanceada. Por lo tanto, se crearon ensambles bajo dos estrategias que tratan el desbalanceo: a nivel de datos y a nivel de algoritmos. Por cada estrategia se configuraron algoritmos de manera diferente. En la estrategia a nivel de datos se utilizaron cinco algoritmos de clasificación trabajando con un solo parámetro. Mientras, en la estrategia a nivel de algoritmos se trabajó con una cantidad de parámetros considerada para sintonizar los algoritmos utilizados. En el ensamble anteriormente mencionado se utilizaron tres algoritmos de clasificación diferentes. Para la creación de los ensambles se manipularon las salidas de los clasificadores base y se combinaron con reglas de combinación: Media, producto, máximo, mínimo y mediana. A cada uno de los ensambles se les evaluó su desempeño promedio y se compararon los resultados para encontrar la mejor estrategia que se ajustó al problema de los datos desbalanceados.

En este trabajo se obtuvo un artículo para ponencia publicado como: "*Proceedings of The 2015 Thirty Fifth Central American and Panama Convention (CONCAPAN XXXV)*", realizado en Tegucigalpa-Honduras 2015. Adicionalmente, se está desarrollando un artículo para revista que busca profundizar más en las características de los datos con la finalidad de complementar este trabajo de investigación.

Finalmente, el presente reporte está organizado a partir de los siguientes capítulos:

Capítulo 1: Contiene el marco teórico necesario para entender los conceptos aplicados en el trabajo de investigación y el estado del arte que comprende las investigaciones que han logrado impactos significativos.

Capítulo 2: Contiene la representación de los datos, la base de datos utilizada y selección de características de los datos.

Capítulo 3: Contiene los diferentes métodos y estrategias desarrollados y aplicados con los respectivos resultados y discusiones.

Capítulo 4: Contiene las respectivas conclusiones e investigaciones que se podrían realizar en un futuro.

1. Marco Teórico y Estado del Arte

En esta sección se presentan los conceptos fundamentales e investigaciones consideradas para el desarrollo de este trabajo de investigación. Primero se comienza con el ensamble de clasificadores y la importancia de su utilidad en problemas de clasificación según la literatura. También, se abordan los diferentes métodos para construir ensambles. Por otra parte, se abarcan los tipos de problemas de datos desbalanceados según la literatura y estrategias para tratar de solucionar dichos problemas. Luego, se incluyen los métodos para evaluar el desempeño de los clasificadores en dominios desbalanceados los cuales se van a aplicar por medio del análisis de curvas ROC teniendo en cuenta como medida el área bajo la curva (AUC); después, se sigue con métodos de ensambles populares como *Bagging*, *Boosting* y *Random Forest*; y finalmente, algoritmos de aprendizaje que se consideraron para la construcción de los ensambles.

1.1 Ensamble de clasificadores

Un ensamble de clasificadores consiste en una combinación de clasificadores que pueden ser homogéneos (iguales algoritmos) o heterogéneos (diferentes algoritmos) que juntos desempeñan una tarea de clasificación con mayor precisión [30]. Existen varios esquemas de combinación para tomar la decisión de la clasificación. Entre los más utilizados se encuentra la votación mayoritaria (*majoritary voting*) [31].
Los ensambles son diseñados para aumentar la precisión de un sólo clasificador entrenando varios clasificadores diferentes y combinando sus decisiones determinando a qué clase pertenece la muestra [6].
En años recientes, las aproximaciones por ensamble han llegado a ser una manera muy útil para avanzar en la clasificación de los datos desbalanceados, ya que pueden ser fácilmente adaptados para destacar las regiones de la clase minoritaria reequilibrando el subconjunto de formación desde el nivel de los datos o mediante la aplicación de diferentes costos [32].
Los ensambles de clasificadores han sido usados para el aprendizaje en los problemas de clasificación con distribuciones de clases desbalanceadas, sin la necesidad de cambios en los clasificadores bases [33].

La intención de combinar múltiples clasificadores en sí contribuirá en reducir la probabilidad de que ocurra sobre-entrenamiento. Existen muchas técnicas frecuentemente usadas con ensambles para mejorar la generalización de predecir la clase minoritaria [34]. Una de las finalidades de un aprendizaje de clasificación puede ser generalmente descrita para obtener un clasificador que genere alta precisión hacia la clase minoritaria sin que estrictamente afecte la precisión de la clase mayoritaria [35].

Por otro lado, según *Dietterich* [36] , hay tres razones por las cuales un ensamble de clasificadores es mejor que un ensamble individual: Estadística, computacional y representacional (ver figura 1). La razón estadística consiste que, entre más clasificadores se utilicen para un conjunto de muestras dados, la probabilidad de clasificar mal los datos disminuye. La razón computacional tiene que ver con los algoritmos heurísticos que muchas veces se estancan en un óptimo local difícil de superar, por este motivo el uso de varios clasificadores evita alguna de estas situaciones no deseables. Finalmente, representacional se refiere al hecho de que determinados tipos de datos pueden ser mejor identificados por un método de clasificación en particular. Por ejemplo, se tiene un conjunto de datos que se clasifican muy bien con un clasificador lineal, si esos mismos datos se aplicaran a un clasificador no lineal es posible que los resultados no sean los esperados. La combinación de clasificadores podría impedir este tipo de situaciones [36].

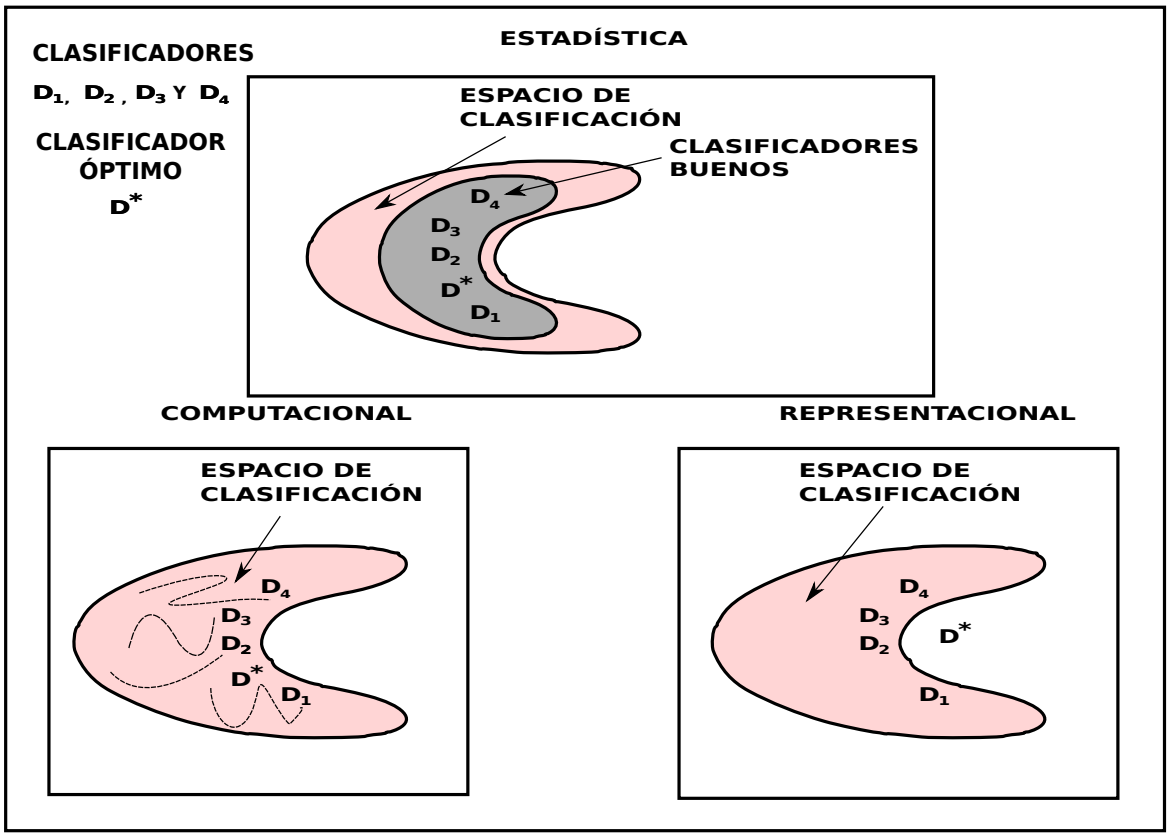


Figura 1: Razones por las cuales un ensamble es mejor que un clasificador. Tomado de [6].

1.2 Métodos para construir ensambles

Existen 5 métodos para crear ensambles de clasificadores: 1) Votación Bayesiana, 2) Manipulación de muestras de entrenamiento, 3)Manipulación del conjunto de características de entrada, 4)Manipulación de los objetos de salida e 5) Inyección de aleatoriedad.

El primer método, llamado Votación Bayesiana (enumeración de la hipótesis) considera que en un conjunto probabilístico Bayesiano, cada hipótesis h define una distribución de probabilidad: $h(x) = P(f(x) = y|x, h)$. Dados una nueva muestra $\mathbf{x}$ y un conjunto de muestras de entrenamiento S, el problema de predecir el valor de $f(x)$ puede ser visto como el problema de computar $P(f(x) = y|S, \mathbf{x})$. Esto se puede escribir nuevamente como la sumatoria de los pesos sobre todas las hipótesis en H.

$$P(f(x) = y|S, \mathbf{x}) = \sum_{h \in H} h(\mathbf{x})P(h|S). \quad (1)$$

Esto se puede ver como un método de ensamble que está formado por todas las hipótesis en H y cada peso por su probabilidad posterior $P(h|S)$ [36].

El segundo método consiste en manipular las muestras de entrenamiento para manejar múltiples hipótesis. El algoritmo de aprendizaje corre varias iteraciones, donde cada iteración trabaja con un subconjunto diferente de las muestras de entrenamiento. Esta técnica trabaja bien especialmente para algoritmos de aprendizaje **inestables**, los cuales son aquellos algoritmos cuya salida padece grandes cambios en respuesta ante pequeños cambios del conjunto de entrenamiento. Algoritmos como árboles de decisión, redes neuronales y algoritmos de aprendizaje de reglas son inestables. En cambio, algoritmos como regresión lineal, vecinos más cercanos y algoritmos de umbral lineal son generalmente algoritmos estables [36].

El tercer método consiste en manipular el conjunto de características de entrada disponibles para generar múltiples clasificadores. Algunos trabajos encontraron que eliminar tan solo pocas características perjudica el desempeño de los clasificadores individuales tanto que la votación del ensamble no va a tener un buen desempeño. Esta técnica sólo trabaja cuando las características tienen grandes redundancias [36].

El cuarto método consiste en manipular los valores de las salidas que están dadas por el algoritmo de aprendizaje para construir un ensamble de clasificadores. Trabajos de investigación realizados reportan que esta técnica mejora el desempeño en algoritmos de árboles de decisión C4.5 y algoritmos de redes neuronales *backpropagation* en una variedad de problemas de difícil clasificación [36].

Finalmente, se puede inyectar aleatoriedad dentro del algoritmo de aprendizaje. En el algoritmo *backpropagation* para entrenar redes neuronales, los pesos iniciales de la red son conjuntos aleatorios. Si el algoritmo es aplicado para el mismo conjunto de

entrenamiento pero con diferentes pesos iniciales, el clasificador resultante puede ser muy diferente [36].

1.3 Tipos de datos desbalanceados

Según la literatura, los datos desbalanceados se pueden clasificar de la siguiente manera: Tamaño de muestra pequeña, entrecruzamiento de clases *(Overlapping)* y pequeños conjuntos disyuntos *(small disjunt)*.

El problema de tamaño de muestra pequeña se presenta cuando los conjuntos de datos no tienen suficientes muestras en una clase [6]. Se evidencia por la relación que existe entre el tamaño del conjunto de la clase minoritaria y el tamaño de la clase mayoritaria. Una relación de desbalanceo grande implica mayor dificultad al problema de clasificación [37]. Investigaciones reportan que la tasa de error causada por la distribución de clases desbalanceadas decrece cuando el número de muestras de la minoría es más representativo. De esta manera los patrones que son definidos por instancias positivas pueden ser mejor enseñados a pesar de la distribución de clases desiguales [6].

El entrecruzamiento de clases consiste en que la clase minoritaria se encuentra muy mezclada con la clase mayoritaria, haciendo que las reglas discriminantes sean duras de implementar a la hora de clasificar bien una clase. Un clasificador estándar logra una buena clasificación si no existe solapamiento entre clases, independiente de la distribución [6].

El problema de los pequeños disyuntos consiste en que la clase minoritaria se encuentra organizada en pequeñas regiones. La existencia de estos subconjuntos también incrementa la complejidad del problema a causa de que la cantidad de instancias entre ellos no es usualmente balanceada [6]. Este es un fenómeno que contribuye a una porción significativa del total de los errores de prueba en la clasificación [38].

1.4 Estrategias que tratan el problema de los datos desbalanceados

Existen tres estrategias que tratan el problema de los conjuntos de datos desbalanceados. Estos son: Aproximación a nivel de datos, aproximación a nivel de algoritmos y métodos de sensibilidad al costo.

La aproximación a nivel de datos es una solución cuyo objetivo consiste en rebalancear la distribución de clases muestreando el espacio de los datos para disminuir el efecto causado por el desbalanceo, actuando como una aproximación externa [39]. Existen dos aproximaciones básicas para la clase minoritaria (sobremuestreo) y para la clase mayoritaria (submuestreo) [40]. *Chawla et al* [41], propuso una técnica para generar

miembros de clase minoritaria interpolando entre varias muestras positivas que las deja encerradas juntas; este método es conocido como *SMOTE* y se explicará más adelante [37].

La aproximación a nivel de algoritmos comprende el diseño de nuevos algoritmos de clasificación o la modificación de los ya existentes para manejar el sesgo introducido por el desbalanceo de clases [42]. Además pueden adaptar el umbral de decisión para crear una preferencia hacia la clase minoritaria. Los métodos de aproximación a nivel de algoritmo requieren especial conocimiento del correspondiente clasificador y campo de aplicación comprendiendo porque el clasificador falla cuando la distribución es desigual [40].

Finalmente, respecto a los métodos de sensibilidad al costo, muchos investigadores estudiaron el problema de la clase desbalanceada, donde la penalización de la clasificación errónea es diferente para instancias de clases distintas y proponen soluciones al problema de desbalanceo de clases [42]. Los métodos de sensibilidad al costo son procedimientos que inducen modelos a partir de datos con distribución de clases desbalanceadas e impactos, cuantificando y luchando contra el desbalanceo [43].

1.5 Métodos para preprocesar datos

Los métodos para preprocesar datos son muy importantes a la hora de crear los clasificadores. Según la literatura los métodos más frecuentes son: Submuestreo aleatorio, sobremuestreo aleatorio, técnica de sobremuestreo minoritario sintético (*SMOTE*), técnica de sobremuestreo minoritario sintético modificado (*MSMOTE*) y preprocesamiento selectivo de datos desbalanceados (*SPIDER*).

El submuestreo aleatorio es un método no heurístico cuyo objetivo es balancear la distribución de clases a través de una eliminación aleatoria de muestras de la clase mayoritaria [6].

El método de sobremuestreo aleatorio tiene la misma finalidad del submuestreo pero la gran diferencia es que se trata de replicar aleatoriamente muestras de la clase minoritaria. Sin embargo, este método puede incrementar la probabilidad de que ocurra sobreentrenamiento, debido a que se hacen copias de instancias existentes [6].

El método *SMOTE* consiste en realizar un sobremuestreo, donde la idea principal es crear nuevas muestras de clase minoritaria interpolando muchas instancias de esta clase que están juntas. El *SMOTE* crea instancias aleatoriamente a partir de técnicas de estimación no parámetrica como los k vecinos más cercanos k-NN [6].

El método *MSMOTE* es una versión modificada de *SMOTE*. Este algoritmo tiene como finalidad dividir las instancias de la clase minoritaria en tres grupos, instancia de

confianza, frontera y de error latente para el cálculo de las distancias entre todas las muestras [6].

Finalmente el método *SPIDER* consiste en una combinación de sobremuestreo local de la clase minoritaria con muestras difíciles de filtrar a partir de la clase mayoritaria. Este método comprende dos fases: identificación y preprocesamiento. La fase de identificación determina las instancias eliminadas como error por KNN. La fase de procesamiento depende de tres opciones establecidas que son: débil, reetiquetación y fuerte; cuando la opción es débil las instancias de la clase minoritaria se amplifican. Cuando la opción es reetiquetación la clase minoritaria se reamplifica y reetiqueta las instancias de clase mayoritaria y finalmente cuando la opción es fuerte amplifica fuertemente la clase minoritaria [6].

1.6 Evaluación de desempeño en dominios desbalanceados

El criterio de evaluación es un factor fundamental para calcular el desempeño de la clasificación y guiar el modelo del clasificador. En un conjunto desbalanceado con dos clases, la matriz de confusión mostrada en la tabla 1 permite observar los resultados de la clasificación dadas las muestras de cada clase [39].
Existen métricas que evalúan el desempeño de cualquier clasificador. Tradicionalmente una de las métricas empíricas comúnmente utilizadas es la precisión (Ver ecuación 2, TP: Verdaderos positivos, TN: Verdaderos negativos, FP:Falsos positivos, FN: Falsos negativos), pero, según la literatura para los conjuntos de datos desbalanceados, no es una gran medida que pueda distinguir entre el número de muestras correctamente clasificadas de clases diferentes. Por lo tanto, se puede llegar a conclusiones erróneas.

$$Acc = \frac{TP + TN}{TP + FN + FP + TN} \tag{2}$$

Cuando trabajamos en dominios desbalanceados, hay métricas más apropiadas que la precisión. A partir de la tabla 1 podemos obtener 4 métricas para medir el desempeño de clasificación de la clase positiva y la clase negativa de forma independiente.

$$TP_{rate} = \frac{TP}{TP + FN} \tag{3}$$

$$TN_{rate} = \frac{TN}{FP + TN} \tag{4}$$

$$FP_{rate} = \frac{FP}{FP + TN} \tag{5}$$

$$FN_{rate} = \frac{FN}{TP + FN} \tag{6}$$

Donde TP_{rate} dado por la ecuación (3) es el porcentaje de instancias positivas correctamente clasificadas, TN_{rate} dado por la ecuación (4) es el porcentaje de instancias negativas correctamente clasificadas, FN_{rate} dado por la ecuación (5) es el porcentaje de instancias negativas erróneamente clasificadas y FP_{rate} dado por la ecuación (6) es el porcentaje de instancias positivas erróneamente clasificadas [6].

Tabla 1: Matriz de confusión para problema de dos clases

	Predicción Positiva	Predicción Negativa
Clase Positiva	Verdaderos Positivos (TP)	Falsos Negativos (FN)
Clase Negativa	Falsos Positivos (FP)	Verdaderos Negativos (TN)

1.6.1 Análisis de curva ROC

El análisis de las características operativas del receptor, en inglés *Receiver Operating Characteristic* (ROC), emergió a partir de la teoría de decisión estadística y fue desarrollada entre 1950 y 1960 para evaluar las señales detectadas en radar y reacciones de estímulos en la psicología sensorial [44]. Sin embargo, ha sido muy útil en el campo del aprendizaje automático, en la evaluación de desempeño de algoritmos de aprendizaje [45]. La curva ROC es construida a partir de la variación del umbral de decisión [46]. En muchos casos la variación del umbral de decisión se da a partir de la variación de los costos de clasificación errónea o de algún parámetro que haga parte del algoritmo de aprendizaje [45], [47]. La curva ROC representada en la figura 2 [6], permite visualizar el compromiso entre los beneficios (TP_{rate}) y los costos de (FP_{rate}). La medida AUC es calculada para obtener el área de la gráfica. Específicamente se aproxima esta área siguiendo la siguiente formula [39]:

$$AUC = \frac{1 + TP_{rate} - FP_{rate}}{2} \tag{7}$$

1.7 Métodos de ensamble más populares

Según la literatura existen tres métodos de ensamble muy comunes estos son: *Boosting, Bagging y Random Forest.*

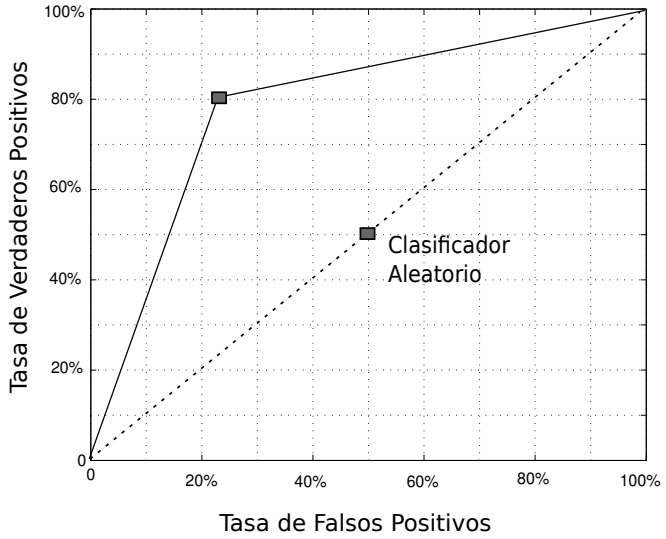


Figura 2: Ejemplo de una gráfica ROC. Dos curvas de clasificadores están descritas: La línea punteada representa un clasificador aleatorio y la línea solida otro clasificador que es mejor que el clasificador aleatorio. Tomado de [6].

El método *Boosting* fue introducido por Schapire en 1990 [48] y es conocida también como *ARCing*,(*adaptive resampling and combining*). Schapire logró que un clasificador débil pasara a ser un clasificador fuerte [6]. Los algoritmos crean múltiples clasificadores iterativamente, un clasificador inicial es construido con el conjunto de entrenamiento original y basado en la clasificación errónea en cada iteración los algoritmos adicionan más pesos a muestras que son previamente mal clasificadas y construye un nuevo clasificador para los datos de entrenamiento modificado. Esto quiere decir que el ensamble es creado combinando clasificadores a partir de múltiples iteraciones [49]. Los métodos *boosting* logran disminuir la varianza decreciendo el sesgo [29].

El método *Bagging* consiste en entrenar diferentes clasificadores con réplicas *bootstrapped*. El *bootstrapped* es un remuestreo que se realiza a partir del conjunto original. Esto significa que un nuevo conjunto de datos es formado para entrenar a cada clasificador de manera aleatoria. Por lo tanto, la diversidad es obtenida con el procedimiento de remuestreo por el uso de diferentes subconjuntos de datos. Finalmente, la salida de cada clasificador es tratada en un sistema de votación o medida de peso mayoritario, el cual es usado para determinar a qué clase pertenece la muestra [6]. Los métodos *bagging* mejoran la generalización disminuyendo la varianza [29].

Finalmente, el método *Random Forest* **RF** es un método que consiste en crear múltiples árboles de regresión y clasificación *(CART)* cada uno es entrenado sobre una muestra *bootstrap* del conjunto de muestra original y busca a través del subconjunto seleccionado de variables de entrada para determinar la división. El RF tiene algunas

características importantes como tener un método efectivo para estimar la búsqueda de datos, estrategias de ponderación para balancear el error en conjuntos desbalanceados y estimar la importancia de variables usadas en la clasificación [50].
En la tabla 2 se presentan algunos trabajos relacionados con las estrategias de ensamble de clasificadores utilizados para tratar los conjuntos de datos desblanceados.

Tabla 2: Trabajos relacionados con el estado del arte en estrategias de ensambles que han tratado conjuntos desbalanceados

Título del trabajo de investigación	**Enfoque**
Cost-Sensitive Learning with Neural Networks [51].	Este trabajo presenta diferentes enfoques para modificaciones de métodos sensibles al costo (*Cost-Sensitive methods*) del algoritmo de aprendizaje (*backpropagation*) en redes neuronales multicapa. Desarrollaron 4 métodos diferentes de aprendizaje sensible al costo con redes neuronales: Clasificación de coste razonable, producción de adaptación, tasa de aprendizaje adaptativo y minimización de los costes de clasificación errónea)
The influence of Class Imbalance on Cost-Sensitive Learning:An Empirical Study [27].	En este trabajo se presenta un estudio empírico utilizando 38 conjuntos de datos, que dan a conocer a qué clase de desbalanceo a menudo afecta el desempeño de los clasificadores sensibles a los costos. Emplean una ecuación que combina y relaciona los pesos sensibles a los costos como también los pesos de la clase desbalanceada.
Learning When Data Sets are Imbalanced on Cost-Sensitive Learning when Cost are Unequal and Unknown [28].	En este trabajo se presentan resultados a partir del estudio de los conjuntos de datos sesgados y desiguales, pero con costos de error no conocidos. Se aplica un submuestreo y sobremuestreo en los datos y se comparan sus respectivos resultados. Además examinan la forma en que el umbral de decisión y el análisis ROC ayudaron con el problema de los conjuntos de datos desbalanceados y presenta evidencias que el submuestreo y sobremuestreo producen casi los mismos clasificadores como lo hacen el movimiento del umbral de decisión y la variación de la matriz costo.

Does Cost-Sensitive Learning Beat Sampling for Classifying Rare Class? [52].	Comparan dos estrategias básicas para trabajar con datos que tienen una distribución sesgada de clases y costos de clasificación errónea no uniformes. Una estrategia se basa en el aprendizaje sensitivo al costo mientras que la otra estrategia emplea muestreo para crear una distribución de clase más balanceada en el conjunto de entrenamiento. El propósito de este trabajo es determinar qué técnica produce un mejor clasificador general y bajo qué circunstancias. Cuando se trabaja con grandes conjuntos de datos con más de 10.000 ejemplos en total, el método sensible al costo supera a los métodos de muestreo, aunque no se dé en todos los casos. El sobremuestreo tuvo mejor comportamiento que el submuestreo.
Cost-Sensitive Boosting for Classification of Imbalanced Data [53].	Este trabajo tiene como objetivo investigar metatécnicas aplicables a la mayoría de algoritmos base, con el objetivo de avanzar en la clasificación de los datos desbalanceados: propone tres nuevos algoritmos *boosting cost-sensitive AdaC1,AdaC2 y AdaC3* los cuales son estudiados con varios algoritmos relacionados existentes, donde indican que el *AdaC2* tiene mejores características superiores ante sus rivales.
Training Cost-Sensitive Neural Networks with Methods Addressing the Class Imbalance Problem [54].	Este artículo estudia empíricamente el efecto del muestreo y el movimiento del umbral en el entrenamiento de redes neuronales sensibles al costo, se analizaron los efectos de sobremuestreo, submuestreo, movimiento del umbral, ensamble duro, ensamble suave y *SMOTE* en el entrenamiento de redes neuronales sensibles al costo empíricamente en 21 conjuntos, con tres tipos de matrices de costo y un conjunto de datos sensible a los datos reales.
A Hybrid Approach to Coping wich High Dimensionality and Imbalance for Software Defect Prediction [55].	Este trabajo consiste en dos clasificadores base y seis técnicas de selección de características. Este trabajo propone una nueva técnica llamada *SelecRUSBoost*, que es una forma de ensamble que incorpora el muestreo de datos para aliviar el desbalanceo de clases y la selección de características para resolver la alta dimensionalidad.

Disturbing Neighbors Ensembles of Trees for Imbalanced Data [7].	Este trabajo propone un método para generar ensambles de clasificadores. Además, este es combinado con cualquier otro método de ensamble, generalmente para dar mejores resultados. Se trabaja con árboles de decisión modelo *(MP5)* y árboles de decisión *Hellinger* las cuales son mejores árboles que los árboles de decisión estándar para datos desbalanceados, sin embargo, cuando se usó el ensamble *Disturbing Neighbors* los mejores resultados lo obtuvieron los árboles de decisión estándar.
An improved ensemble approach for imbalanced classification problems [20].	Este trabajo trata de una aproximación de ensamble mejorado para la clasificación de datos desbalanceados. El algoritmo es basado en el submuestreo de la clase mayoritaria para crear subespacios de objetos balanceados para entrenar clasificadores base individuales. Se propone un ensamble basado en la idea de crear varios clasificadores base a partir de subespacios desbalanceados y adicionar un paso de eliminación ensamble para descartar clasificadores redundantes, este método se llama *PUSBE*.
An Evaluation of Classifier Ensembles for Class Imbalance Problems [15].	En este trabajo evalúan 7 ensambles a partir del estado del arte, en un extenso conjunto de experimentos, comparan sus desempeños sobre 5 conjuntos de datos. Los ensambles investigados son *SMOTE-Bagging, SMOTEBoost, IIVotes, EasyEnsemble, Evolutionary Cost-Sensitive Ensemble,Under-Sampling Balanced Ensemble y Pruned Under-Sampling Balanced Ensemble(PUSBE)*, donde el *PUSBE* fue el ensamble que dio mejores resultados en desempeño y por lo tanto es el más indicado comparado con los otros ensambles.
Constructing Support Vector Machine Ensemble With Segmentation for Imbalance Classification problems [13].	El trabajo comprende la aplicación de un nuevo método de ensamble de clasificación llamada ensamble con segmentación a partir de máquinas de soporte vectorial *(SeEnSVM)*, para tratar el problema de los conjuntos de datos desbalanceados. El nuevo enfoque que presenta el trabajo es la construcción de un conjunto de clasificadores *SMV (suport Vector Machine)* con técnicas VQ(Vector de Quantización). Este algoritmo fue aplicado a 6 bases de datos y los resultados confirmaron mejor desempeño ante métodos como *SVM, Under-Sampling, SMOTE, Cost-sensitive learning*, para el problema del desbalanceo.

1.8 Algoritmos de aprendizaje

Los algoritmos utilizados en el presente trabajo son: Los árboles de decisión, k-vecinos más cercanos, máquinas de vectores de soporte, análisis discriminante lineal y gradiente descendiente estocástico.

Los Árboles de Decisión son técnicas que clasifican a instancias ordenándolas en base a valores de funciones. Cada nodo en un árbol de decisión representa una característica en una instancia para ser clasificado y cada rama representa un valor que el nodo puede asumir. Las instancias se clasifican a partir del nodo raíz [56]. Los árboles de decisión son fundamentales a la hora de tomar una buena decisión siendo fáciles de construir y entender [57].

Los K-vecinos Más Cercanos (*k-nearest neighbors*) es una de la técnicas de estimación no parámetrica más ampliamente estudiada. Si tenemos un conjunto de n muestras etiquetadas (conjunto de entrenamiento), es $X = (x_1, \omega_1), (x_2, \omega_2), , (x_n, \omega_n)$ el clasificador K-NN consiste en asignar una muestra de entrada x a la clase más frecuentemente representada entre las k instancias cerradas en el conjunto de entrenamiento [11]. La regla de decisión del vecino más cercano asigna a una muestra no clasificada la clase más representativa entre sus k vecinos más cercanos [58].

Las Máquinas de Soporte Vectorial (*SVM*) son algoritmos de clasificación construidos a partir de la suposición que ambas clases son linealmente separables. Esta suposición permite utilizar una función discriminante lineal para dividir las instancias dentro de las dos clases. Una función discriminante lineal usa la fórmula $g(\mathbf{x}/\mathbf{w}, w_0) = \mathbf{w}^T\mathbf{x} + w_0$, donde $\mathbf{w}$ es el vector de pesos y w_0 es el sesgo. Por lo general, la *SVM* es un clasificador lineal que selecciona el discriminante que maximiza la distancia entre las dos clases [14]. Con el objetivo de trabajar con conjuntos en condición desbalanceada en [15] proponen una estrategia con máquinas de soporte vectorial para solucionar el problema del desbalanceo. Por lo tanto, las ecuaciones 8, 9 y 10 demuestran cómo las máquinas de soporte vectorial son aplicadas teniendo en cuenta la incidencia de cada clase:

$$min_{(w,\xi,b)} E_P = \frac{1}{2}\|w\|^2 + C^+ \sum_{\substack{i=1 \\ y=+1}}^{l} \xi_i + C^- \sum_{\substack{i=1 \\ y=-1}}^{l} \xi_i \tag{8}$$

$$y_i(w, \Phi(x_i) + b) \geq 1 - \xi_i \;\; \forall i = 1, ...l \tag{9}$$

$$\xi_i \geq 0 \; \forall i = 1, ...l \tag{10}$$

l es el total de muestras del conjunto, C^+ y C^- están determinados por las siguientes formulas:

$$C^+ = \frac{C\sqrt{\lambda}}{l} \tag{11}$$

$$C^- = \frac{C}{l\sqrt{\lambda}} \tag{12}$$

C es un hiperparámetro y λ está determinado por la siguiente ecuación:

$$\lambda = \frac{l^-}{l^+} \tag{13}$$

l^- es el número de muestras de la clase negativa, l^+ es el número de muestras de la clase positiva.

El Análisis Discriminante Lineal es un conjunto de métodos de regresión y clasificación. Este caso se definirá como clasificación. Por lo tanto, este método de clasificación parte de la suposición de separar con fronteras de decisión lineal los datos [59]. Una idea general es suponer que se tiene un conjunto de muestras de entrenamiento $\mathbf{x}_1, \ldots \mathbf{x}_n$ cada una de las muestras tiene asignada una clase ω_1 o ω_2 (Caso biclase). Teniendo este conjunto de datos, se busca un vector de pesos $\mathbf{w}$ y un umbral w_0 tal que

$$\mathbf{w}^T\mathbf{x} + w_0 \begin{cases} > 0 \\ < 0 \end{cases} \Rightarrow \mathbf{x} \in \begin{cases} \omega_1 \\ \omega_2 \end{cases} \tag{14}$$

El Gradiente Descendente Estocástico se utiliza para regularizar métodos de predicción y clasificación lineal tales como: Regresión con mínimos cuadrados, regresión logística y máquinas de soporte vectorial para clasificación, los cuales han sido extensamente usados en estadística y aprendizaje automático. Por consiguiente, en [60] se habla sobre la aplicación de gradiente descendiente estocástico y se explica el problema de predicción de una salida no observada y, basado en un vector de entrada observado $\mathbf{x}$. Además, asumen que la calidad de un predictor $p(x)$ es medida por una función $\phi(p(x), y)$, y los datos (X,Y) son extraídos a partir de una distribución D subyacente no conocida, cuyo objetivo es encontrar el $p(x)$ tanto que la verdadera pérdida esperada de p dada en la ecuación 15 sea tan pequeña como sea posible:

$$Q(p(.)) = E_{x,y}\phi(p(X), Y), \tag{15}$$

$E_{X,Y}$ denota la esperanza con respecto a la distribución D real subyacente. También se asume que la función de pérdida $\phi(p, y)$ es una función convexa de p y se enfoca en predictores que tomen la forma $p(x) = \mathbf{w}^T\mathbf{x}$.

El Método de Clasificación Naive-Bayes es un algoritmo de aprendizaje supervisado basado en la aplicación del teorema Bayes con la de Naive que asume una independencia entre las características que representan los datos. Por lo tanto, la relación del teorema de Bayes dada una clase y y una característica dependiente x_1 del vector x_n es:

$$P(y|x_1, ..., x_n) = \frac{P(y)P(x_1, ..., x_n)|y}{p(x1, ..., x_n)} \tag{16}$$

Suponiendo la independencia del teorema de Naive

$$P(x_i|y, x_1, ..., x_{i-1}, x_{i+1}, ..., x_n), \tag{17}$$

para todo i, la relación representada en la ecuación 16 es simplificada a

$$P(y|x_1, ..., x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{p(x1, ..., x_n)} \tag{18}$$

El factor $P(x_1, ..., x_n)$ es considerado como constante y se realiza la siguiente regla de combinación:

$$\begin{gathered} P(y \mid x_1, \ldots, x_n) \propto P(y)\prod_{i=1}^{n} P(x_i|y) \\ \Downarrow \\ \hat{y} = \arg\max_y P(y)\prod_{i=1}^{n} P(x_i \mid y)) \end{gathered} \tag{19}$$

Podemos usar la estimación máxima a posterior para estimar $P(y)$ y $P(x_i \mid y)$; lo anterior es entonces la frecuencia relativa de clase y en el conjunto de entrenamiento.

Para Naive-Bayes gaussiano se considera como la verosimilitud de las características $P(x_i \mid y)$:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right) \tag{20}$$

Los parámetros σ_y y μ_y son estimados usando máxima verosimilitud.

2. Representación de los datos y Selección de características

Esta sección presenta la representación de los datos y selección de las características. Los datos se representan en características, que pueden ser de descripción de la cadena primaria o fisicoquímicas. Por tanto, en este trabajo se consideraron 17 características y para la selección de éstas se utilizó el método *Secuencial Forward Selection*. Por medio de este método se determinó el número de características del péptido que hacen al clasificador tener una buena discriminación. El proceso de selección de características parte de un trabajo previo, el cual considera al clasificador k-vecinos más cercanos como el mejor estimador. Finalmente, en este capítulo no se hace un análisis profundo en el desarrollo de selección de características, si no, permite dar un enfoque intuitivo sobre la representación de las características de los péptidos.

2.1 Base de datos

La base de datos utilizada es la APD [61], [62], la cual está conformada por 2436 péptidos antimicrobianos (241 péptidos provienen de bacterias, 2 provienen de archaea o arqueobacterias, 7 provienen de protista, 12 provienen de fúngicos , 311 provienen de plantas, y 1822 provienen de animales) con las siguientes actividades: antibacterial, antiviral, antifúngica, antiparasitante, antitumoral o anticancerigena, antiprotista, insecticida, espermicida, anti-VIH-1, quimiotáctica, antioxidante e inhibición de proteasa.

2.2 Aminoácidos

Los péptidos están conformados por 20 aminoácidos base estos son: Alanina(A), Cisteína(C), Aspargina(D), glutamina(E), fenilalanina(F), glicina(G), histidina(H), Isoleucina(I), Lisina(K), Leucina(L), Metionina(M), Asparagina(N), Prolina(P), Glutamina(Q), Arginina(R), Serina(S), Treonina(T), Valina(V), Triptófano(W) y Tirosina(Y) [63].

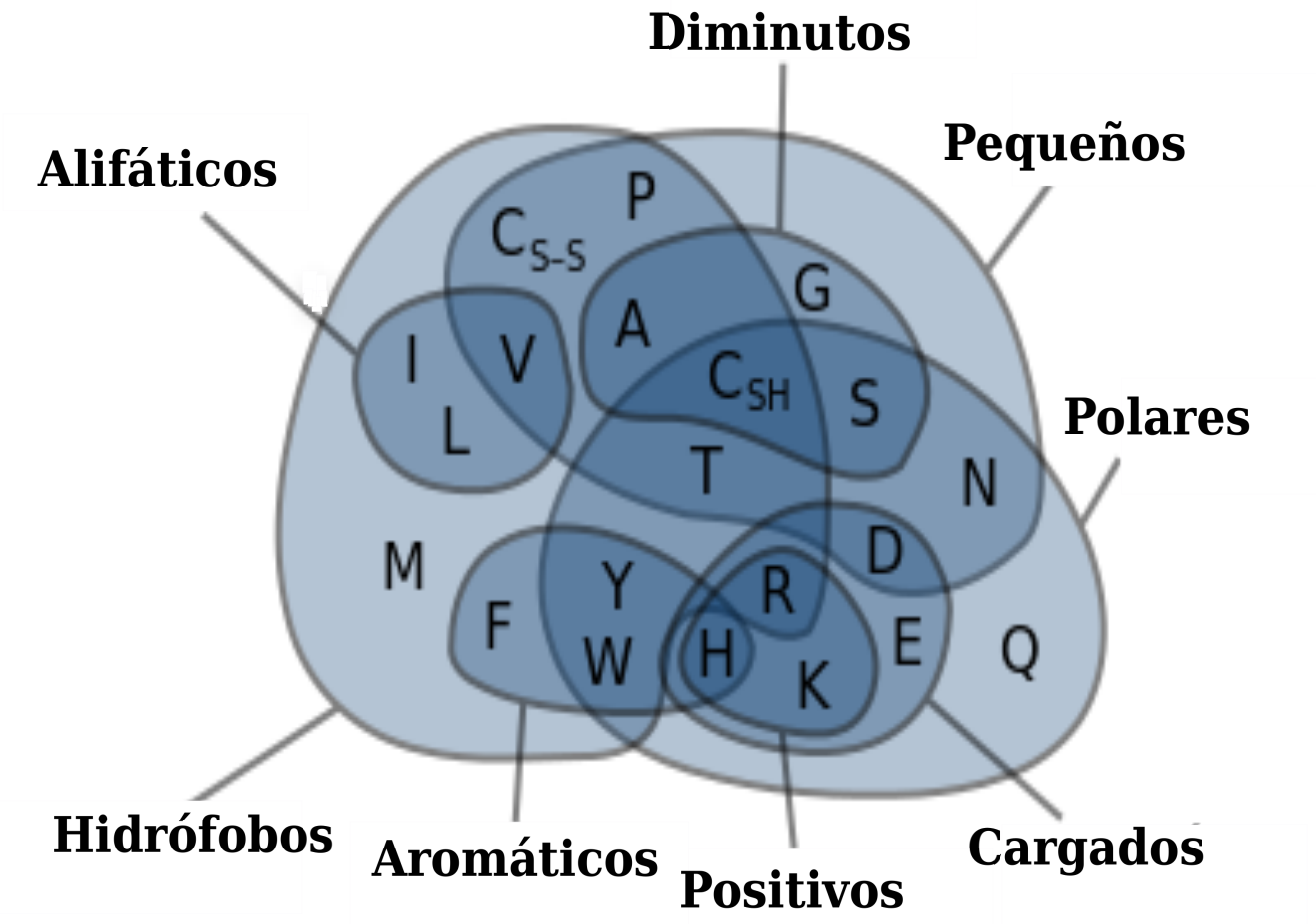


Figura 3: Diagrama de Venn de las propiedades de los aminoácidos [63]

Diferentes agrupamientos de aminoácidos determinan propiedades químicas importantes como: hidrofóbica, polar, ácida o básica (ver figura 3). Estas propiedades son extraidas a partir de la estructura primaria del péptido o polipéptido (proteína). En el desarrollo de este trabajo se partió de una estructura primaria para representar los aminoácidos de manera que hagan diferenciar un péptido con propiedades antibacterianas a otro que nos las posea. Los péptidos fueron representados bajo 17 conjuntos de características: Autocorrelación de *Moreau-Broto* normalizada, Autocorrelación de *Moran*; autorrelación de *Geary*; descriptores de distribución, transición y composición; k números de secuencias ordenadas; descriptores de semisecuencias ordenadas, hidrofobicidad, volumen de *Van der Waals* normalizado; polaridad; polarizabilidad; carga; autocorrelación; composición del aminoácido; composición dipéptida, cantidad de aminoácidos en el terminal N, cantidad de aminoácidos en el terminal C y vector de momento de composición.

2.3 Representación de los Péptidos

La representación por la **composición de aminoácidos** consiste en tener una secuencia de aminoácidos s sobre un alfabeto A, por consiguiente la composición c_0 está definida como:

$$c_0(s) = \frac{count(l, s)}{|s|} \; \forall l \in A \tag{21}$$

$count(l, s)$ es una función que cuenta el número de ocurrencias de la letra l en una secuencia s, y $|s|$ es el tamaño de la secuencia. Por lo tanto, el tamaño del vector de características c_0 depende del tamaño del alfabeto A [65]. c_0 constituye un histograma de la composición del aminoácido. Está composición produce 20 descriptores por los 20 tipos de aminoácidos en cada péptido.

La representación de **cantidad de aminoácidos en el terminal-N** contiene el número de veces que se encuentran cada uno de los aminoácidos que hacen parte de una secuencia s. Esta composición aplica para las primeras 40 letras de la secuencia de un péptido. Esta representación se puede calcular como:

$$c_1(s) = count(l, s) \; \forall l \in A \tag{22}$$

La composición dipéptida está definida como:

$$f_r(r, s) = N_{rs}/(N - 1) \tag{23}$$

Tabla 3: Representación de los 20 amino ácidos base [64]

Aminoácido	Código de tres letras	Código de una letra
Alanina	Ala	A
Cisteína	Cys	C
Asparagina	Asp	D
Glutamina	Glu	E
Fenilalanina	Phe	F
Glicina	Gly	G
Histidina	His	H
Isoleucina	Ile	I
Lisina	Lys	K
Leucina	Leu	L
Metionina	Met	M
Asparagina	Asn	N
Prolina	Pro	P
Glutamina	Gln	Q
Arginina	Arg	R
Serina	Ser	S
Treonina	Thr	T
Valina	Val	V
Triptófano	Trp	W
Tirosina	Tyr	Y

$r, s = 1, 2, 3, \cdots, 20$ y N_{rs} es el número de dipéptidos o subsecuencias contiguas de aminoácidos tipo r y s. Esta composición produce un total de 400 descriptores por cada péptido.

La autocorrelación describe el nivel de correlación entre dos péptidos en terminos de sus propiedades fisicoquímicas. Estas propiedades están basadas en la distribución de aminoácidos dada una secuencia. Hay ocho propiedades de aminoácidos para derivar descriptores de autocorrelación [1]. La primera es la escala de hidrofobicidad que parte del volumen de caracter hidrofóbico para los 20 tipos de aminoácidos en 60 estructuras protéicas. La segunda es el índice de flexibilidad promedio derivado a partir del promedio estadístico de los B-factores de cada tipo de aminoácidos en las estructuras protéicas dadas por cristalografía de rayos X. La tercera el parámetro de polarizabilidad calculado a partir de grupos refractivos molares. La cuarta es la energía disponible de la solución de aminoácidos en agua. La quinta es el residuo accesible a áreas de la superficie los cuales provienen de valores promedio de las proteínas plegadas. La sexta es el volumen de residuos de aminoácidos medidos por Fisher. La séptima son los parámetros estéricos derivados por el radio de *Van der Waals* de los átomos que hacen parte de la cadena lateral del aminoácido. La octava es la mutabilidad relativa obtenida multiplicando el número de mutaciones por la frecuencia de ocurrencia de un aminoácido. Cada propiedad es normalizada como:

$$P'_r = \frac{(P_r - \bar{P})}{\sigma} \tag{24}$$

Donde $\bar{P}$ es el promedio y σ es la desviación estándar de las propiedades de los 20 aminoácidos , $\bar{P}$ y σ son calculados por las siguientes ecuaciones:

$$\bar{P} = \frac{\sum_{r=1}^{2} 0P_r}{20} \tag{25}$$

Tres características diferentes basadas en autocorrelación son calculadas a partir de las ocho propiedades mencionadas [1]. Estas autocorrelaciones producen 240 valores de descriptores. La primera es la **autocorrelación de *Moreau-Broto*** [1] :

$$AC(d) = \sum_{i=1}^{N-d} P_i P_{i+d} \tag{26}$$

d es el lapso de la autocorrelación, P_i y P_{i+d} son las propiedades del aminoácido en la posición i y $i + d$, respectivamente.

La **autocorrelación de *Moreau-Broto* Normalizada** [1] está definida como:

$$ATS(d) = AC(d)/(N - d) \tag{27}$$

Donde $d = 1, 2, 3, \cdots 30$.

La segunda es la **autocorrelación de *Moran*** está definida como:

$$I(d) = \frac{\frac{1}{N-d}\sum_{i=1}^{N-d}(P_i - \bar{P})(P_{i+d} - \bar{P})}{\frac{1}{N}\sum_{i=1}^{N}(P_i - \bar{P})^2} \qquad d = 1, 2, 3, \cdots, 30 \tag{28}$$

La tercera característica es la **autocorrelación de *Geary*** [1] la cual es utilizada para analizar frecuencias alelas y estructuras de población, está definida como:

$$C(d) = \frac{\frac{1}{2(N-d)} \sum_{i=1}^{N-d} (P_i - P_{i+d})^2}{\frac{1}{N-1} \sum_{i=1}^{N} (P_i - \bar{P})^2} \qquad d = 1, 2, 3, \cdots, 30 \tag{29}$$

Los descriptores **composición**, **transición** y **distribución** describen la composición global de una propiedad aminoácida dada en una proteína, la frecuencia con las cuales las propiedades cambian a lo largo de la secuencia de aminoacidos en un péptido y el patrón de distribución de la propiedad a lo largo de la secuencia. En [66] muestran un ejemplo (ver figura 4) sencillo que consta de una secuencia constituida por dos aminoácidos

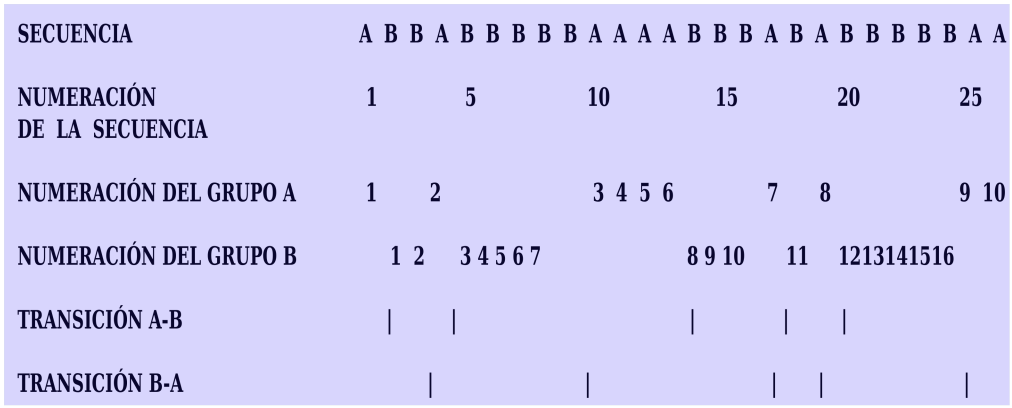


Figura 4: Secuencia constituida por dos tipos de residuos A y B [66]

Esta secuencia tiene 10 residuos (cantidad de aminoácidos) tipo A y 16 tipo B. Supongamos que n_1 y n_2 representan la cantidad de aminoácidos de tipo A y B respectivamente. Entonces, considerando que $n_1 = 10$ y $n_2 = 16$, los porcentajes de **composición**, **C** para cada tipo de aminoácidos se calculan como:

$n_1 \times 100{,}0/(n_1 + n_2) = \mathbf{38.5\,\%}$ para A y $n_2 \times 100{,}0/(n_1 + n_2) = \mathbf{61.5\,\%}$ para B.

El segundo descriptor, T (**transición**) representa en porcentaje la frecuencia con el cual A es seguido por B y B seguido por A. En este caso, hay un total de 10 transiciones las cuales un tipo de aminoácido sigue el otro (Ver figura 4). Por lo tanto, el descriptor **T** se calcula como:

$$(10/25) \times 100{,}0 = 40\,\%.$$

El tercer descriptor, D (Distribución) se basa en encontrar las distribuciones de los tipos de aminoácidos dados en la secuencia en un 0,0 %, 25 %, 50 %, 75 % y 100 % del

tamaño de la secuencia relacionándola con la posición de la secuencia de aminoácidos. Por ejemplo, para la primera posición se encuentra el residuo A (ver figura 4) y coincide con el primer residuo de la cadena, por lo tanto, la distribución 0,0 % del aminoácido A en la secuencia se considera con un valor de 0,0. El 25 % de la cantidad de aminoácidos A se aproxima a 2 y se encuentra en la posición 4 de la secuencia (ver figura 4). Finalmente, esta posición se divide por el tamaño de la secuencia, resultando como:

$$(4/26) * 100{,}0 = 15{,}4\,\%.$$

Similarmente el 50 % de los aminoácidos A se encuentran comprendidos hasta la posición 12, por lo tanto, la distribución queda como:

$$(12/26) * 100{,}0 = 46{,}1\,\%.$$

El 75 % y 100 % del descriptor de distribución son 73 % y 100 %, respectivamente.

Los valores de descriptores de distribución para el aminoácido B son: 7,69 %, 23,1 %, 53,8 %, 76,9 % y 92,3 %.

En general, todos los aminoácidos están divididos dentro de tres grupos: polar, neutro e hidrofóbico. La composición del descriptor C consiste de tres valores: la composición del porcentaje global polar, neutral e hidrofóbico en el péptido. El descriptor de transición T, también consiste de tres valores: El porcentaje de la frecuencia de un residuo polar seguido por un residuo neutral o un residuo neutral por un polar, un residuo polar seguido por un hidrofóbico o viceversa, un residuo hidrofóbico seguido por un polar. La distribución D como se mencionó anteriormente consiste de 5 valores por cada grupo: Polar, neutral e hidrofóbico. Por lo tanto, la cantidad de descriptores está establecido de la siguiente manera: $3(C) + 3(T) + 5 \times 3(D) = 21$ solo para un atributo. Si son siete atributos producirá un total de $7 \times 3 = 21$ descriptores y $7 \times 21 = 147$ valores de descriptores (ver tabla 4).

Las características de secuencias ordenadas pueden también ser usadas para representar patrones de distribución de los aminoácidos de una propiedad fisicoquímica a lo largo de secuencia peptídica. Estas características son usadas para predecir localizaciones subcélulares proteínicas. El nivel del número de secuencias ordenadas y unidas está definido como:

$$\tau_d = \sum_{i=1}^{N-d} (d_{i,i+d})^2 \qquad d = 1, 2, 3, \cdots, 30 \tag{30}$$

$d_{i,i+d}$ es la distancia que hay entre dos aminoácidos, de la posición i a la $i + d$. Hay dos tipos de descriptor definidos como semi-secuencias ordenadas [67], [1]: tipo-1 y tipo-2. La tipo-1 se puede calcular como:

$$X_r = \frac{f_r}{\sum_{r=1}^{20} f_r + w \sum_{=1}^{30} \tau_d} \qquad r = 1, 2, 3, \cdots, 20 \tag{31}$$

f_r es la ocurrencia normalizada del aminoácido tipo i y w es un factor de peso(w = 0,1). La semi-secuencia ordenada tipo-2 está definida como:

$$X_d = \frac{w\tau_{d-20}}{\sum_{r=1}^{20} f_r + w\sum_{=1}^{30} \tau_d} \qquad d = 21, 22, 23, \cdots, 50 \tag{32}$$

Para finalizar, la tabla 5, muestra el conjunto de características utilizados para la selección de las características.

Tabla 4: Atributos y división de aminoácidos dentro de tres grupos por cada atributo [1]

Atributo	**División**		
Hidrofobicidad	**Polaridad**	**Neutralidad**	**Hidrofobicidad**
	RKEDQN	GASTPHY	CLVIMFW
***Van der Waals* Normalizado**	**0-2.78**	**0-2.78**	**0-2.78**
	GASTPD	NVEQIL	MHKFRYW
Polaridad	**4.9-6.2**	**8.0-9.2**	**10.4-13.0**
	LIFWCMVY	PATGS	HQRKNED
Polarizabilidad	**0-1.08**	**0.128-120.186**	**0.219-0.409**
	GASDT	CPNVEQIL	KMHFRYW
Carga	**0-2.78**	**0-2.78**	**0-2.78**
	KR	ANCQGHIL	DE
		MFPSTWYV	
Estructura Secundaria	**Helice**	**Hoja**	**Espiral**
	EALMQKRH	VIYCWFT	GNPSD
Accesibilidad al Solvente	***Buried***	**Expuesto**	**Intermedio**
	ALFCGIVW	PKQEND	MPSTHY

Tabla 5: Conjuntos de características

Conjunto Carac.	**No. de Carac.**	**Descripción**
f_0	240	Autocorrelación de Moreau-Broto Normalizada
f_1	240	Autocorrelación de Moran
f_2	240	Autocorrelación de Geary
f_3	147	Composición, transición y distribución
f_4	90	Número de secuencias juntas ordenadas
f_5	50	Semi-Secuencias Ordenadas
f_6	21	Hidrofobicidad
f_7	21	Vander Waals Normalizado
f_8	21	Polaridad
f_9	21	Polarizabilidad
f_{10}	21	Carga
f_{11}	19	Autocorrelación
f_{12}	20	Composición del aminoácido
f_{13}	400	Composición Dipéptida
f_{14}	20	Cantidad de aminoácidos en el terminal-N
f_{15}	20	Cantidad de aminoácidos en el terminal-C
f_{16}	40	Vector de Momento Compuesto

2.4 Selección de Características

Para encontrar las características que mejor representan los datos se optó por el método ***Secuencial Forward Selection*** (ver [68]). Este método de selección de características arranca con un solo subconjunto de características y progresivamente se van adicionando nuevos conjuntos de características para llegar a un objetivo, que es mejorar el índice de precisión [69]. Suponiendo que se tiene un conjunto d_1 características, X_{d_1}. Por cada ξ_j características aún no seleccionadas, la función $J_j = J(X_{d_i} + \xi_j)$ es evaluada. La característica que produce el máximo valor de J_i es escogida como una de las características que se van a acumular a un conjunto X_{d_1}. Por lo tanto, en cada etapa, la característica se escoge, se adiciona al conjunto actual de características y maximiza el criterio de selección. El algoritmo consiste en comenzar con un solo conjunto de características y termina cuando el desempeño del clasificador comienza a empeorar o cuando llega hasta el máximo número de características permitidas.

Algoritmo 1. Algoritmo con validación cruzada para la selección del mejor conjunto de características para un procedimiento de búsqueda secuencial [68]

1: Dividir los datos en conjuntos de entrenamiento y prueba.
2: Especificar la estrategia de selección, en este caso es *Secuencial Forward Selection*
En cada etapa del algoritmo:

a) Generar subconjuntos de características para evaluar

b) Procedimiento de Validación Cruzada

- Dividir el conjunto de entrenamiento en 10 partes iguales, asegurando que todas las clases están representadas en cada parte, usar nueve partes para entrenar y el conjunto restante para prueba.
- Entrenar el clasificador por cada subconjunto de características, h sobre cada subconjunto k, de los datos de entrenamiento a la vez y evaluar sobre el conjunto restante. Obtener el desempeño ($CV(h,k)$, en este caso AUC).
- Promediar el resultado

$$CV(h) = \frac{1}{10}\sum_k CV(h,k)$$

c) Seleccionar el subconjunto de características más pequeño, S_{h^*}, tanto que $CV(h)$ es óptimo o cerca del óptimo, para el siguiente paso de busqueda.

3: Evaluar sobre los datos de prueba usando el subconjunto de características más pequeño, S_{h_*}, buscar el mejor desempeño, entrenar sobre el conjunto de entrenamiento y evaluar el desempeño sobre el conjunto de prueba.

La tabla 6 representa las mejores características acumuladas de arriba hacia abajo para cada conjunto creado por la validación cruzada. Para dar una mejor explicación, el número 12 representa en dicha tabla la característica asociada a la composición del aminoácido (ver tabla 5) y fue la característica individual que arroja el mejor AUC en la mayoría de conjuntos dados por la validación cruzada. La característica 14 fue la característica que acompañada con otra característica arrojó el mejor desempeño en varias oportunidades.

Tabla 6: Mejores características acumuladas con validación cruzada 10

Mejores Carac.	**Conjuntos de la validación cruzada**									
Acumuladas	1	2	3	4	5	6	7	8	9	10
Acum.1	**12**	4	4	15	**12**	8	**12**	10	1	**12**
Acum.2	4	5	5	**14**	4	15	**14**	4	**14**	3
Acum.3	14	10	7	10	1	6	0	11	11	13
Acum.4	15	12	6	16	15	14	9	5	12	4
Acum.5	10	11	15	9	6	4	6	6	6	14
Acum.6	0	16	8	12	14	12	5	7	16	7
Acum.7	1	6	10	11	11	9	7	8	5	15
Acum.8	11	9	14	0	5	16	8	14	15	9
Acum.9	6	8	9	6	16	11	4	9	9	6
Acum.10	8	0	12	8	8	7	1	3	8	5
Acum.11	16	14	16	4	0	3	2	13	10	0
Acum.12	2	3	11	5	2	13	11	15	4	1
Acum.13	9	2	0	7	7	10	16	16	13	8
Acum.14	7	7	13	13	9	5	10	12	0	11
Acum.15	5	1	1	2	13	0	3	0	2	16
Acum.16	13	15	2	1	3	1	15	1	7	10
Acum.17	3	15	3	3	10	2	13	2	2	2

En la tabla 7 se observan los desempeños asociados a la acumulación de características y su promedio. Se podría encontrar mejores desempeños con más características pero, en sí, no hay diferencias significativas para encontrar el mejor desempeño y el costo computacional al trabajar con más características es alto. Por lo tanto, se utilizaron dos conjuntos de características que proporcionaron buenos resultados en el desempeño del clasificador. También, se consideró que estas características podrían representar propiedades fisicoquímicas importantes del péptido. Las herramientas *web* [70] y [71] se utilizaron para obtener las características.

Tabla 7: Desempeños AUC del Clasificador KNN con las mejores características acumuladas

Carac.	**Conjuntos de la Validación Cruzada**										**Prom.**
Acumuladas	1	2	3	4	5	6	7	8	9	10	**AUC**
Acum.1	0,96	0,92	0,94	0,96	0,97	0,87	0,79	0,84	0,90	0,96	0,91
Acum.2	0,97	0,88	0,94	0,94	0,96	0,91	0,76	0,85	0,97	0,97	0,92
Acum.3	0,97	0,92	0,95	0,97	0,98	0,92	0,80	0,85	0,97	0,98	0,93
Acum.4	0,97	0,94	0,94	0,98	0,99	0,94	0,78	0,85	0,96	0,99	0,94
Acum.5	0,97	0,95	0,95	0,98	0,99	0,95	0,78	0,89	0,97	0,99	0,94
Acum.6	0,98	0,94	0,94	0,98	0,99	0,95	0,78	0,89	0,96	0,99	0,94
Acum.7	0,99	0,94	0,94	0,97	0,99	0,94	0,78	0,89	0,95	0,99	0,94
Acum.8	0,99	0,93	0,94	0,97	0,99	0,95	0,78	0,89	0,94	0,99	0,94
Acum.9	0,99	0,93	0,94	0,97	0,99	0,95	0,79	0,89	0,94	0,99	0,94
Acum.10	0,99	0,92	0,95	0,96	0,99	0,95	0,77	0,90	0,94	0,99	0,94
Acum.11	0,99	0,92	0,95	0,97	0,99	0,89	0,76	0,90	0,95	0,99	0,93
Acum.12	0,99	0,93	0,95	0,96	0,98	0,94	0,76	0,90	0,94	0,99	0,93
Acum.13	0,990	0,92	0,95	0,95	0,98	0,94	0,76	0,90	0,95	0,99	0,93
Acum.14	0,98	0,93	0,96	0,95	0,98	0,92	0,76	0,90	0,94	0,99	0,93
Acum.15	0,97	0,91	0,96	0,95	0,98	0,92	0,77	0,90	0,96	0,99	0,93
Acum.16	0,98	0,92	0,95	0,94	0,98	0,92	0,78	0,90	0,96	0,99	0,93
Acum.17	0,98	0,91	0,95	0,94	0,97	0,91	0,80	0,9	0,96	0,99	0,93

3. Métodos y Estrategias de Ensamble

En esta sección se presentan las metodologías y estrategias de ensamble para tratar conjuntos de datos desbalanceados. Primero se explican las diferentes reglas de combinación como: media, producto, máximo, mínimo y mediana. Estas reglas fueron utilizadas para todas la creación de los ensambles propuestos. Las estrategias que se aplicaron para tratar el desbalanceo fueron: A nivel de datos y a nivel de algoritmos. A nivel de datos se trabajó con un ensamble constituido por cinco algoritmos base de clasificación: Máquinas de soporte vectorial, k vecinos más cercanos, Naive-Bayes Gaussiano y análisis discriminante lineal. Para la estrategia a nivel de algoritmos se creó un ensamble construido por tres algoritmos de clasificación: Máquinas de soporte vectorial, gradiente descendente estocástico y k-vecinos más cercanos. Cada algoritmo trabajó con cinco parámetros. Además, se consideraron parámetros relacionados con el desbalanceo de los datos en la máquina de soporte vectorial y el gradiente descendiente estocástico. Adicionalmente, se trabajó con ensambles populares basados en *Boosting* and *Random Forest*. Para el desarrollo de este trabajo, se utilizaron las herramientas de $scikit-learn$, las cuales están basadas en aprendizaje automático (*Machine learning*) y se pueden importar como librerías para el lenguaje de programación *Python*.

3.2 Reglas de combinación

En general, la decisión de cada clasificador base i es configurada para proporcionar la probabilidad que tiene un péptido de pertenecer a la clase positiva $(d_{i,+}(x))$ y a la clase negativa $(d_{i,-}(x))$. Luego se aplican reglas de combinación en cada ensamble para obtener una probabilidad asociada a la clase positiva $(P_+(x))$. Para llegar a la probabilidad mencionada hay que tener en cuenta la siguiente regla de decisión:

$$x \in + \quad si \quad P_+(x) > P_-(x) \quad \forall(x_+) \neq (x_-) \tag{33}$$

x_+ y x_- son muestras positivas y negativas respectivamente. Las reglas de combinación utilizadas fueron: media, máximo, mínimo, productoria y mediana, según lo especifican las ecuaciones (34a) a (34e) [72].

$$P_+(x) = Media_+(x) = \frac{1}{L}\sum_{j=1}^{L} d_{i,+}(x) \tag{34a}$$

$$P_+(x) = Prod_+(x) = \frac{\prod_{j=1}^{L} d_{i,+}(x)}{\prod_{j=1}^{L} d_{i,+}(x) + \prod_{j=1}^{L} d_{i,-}(x)} \tag{34b}$$

$$P_+(x) = Max_+(x) = \max_{+}\{d_{i,+}(x)\},\ i = 1, \ldots, L \tag{34c}$$

$$P_+(x) = Min_+(x) = \min_{+}\{d_{i,+}(x)\},\ i = 1, \ldots, L \tag{34d}$$

$$P_+(x) = Mediana_+(x) = \text{median}\{d_{i,+}(x)\},\ i = 1, \ldots, L \tag{34e}$$

$d_{i,+}$ es la salida de cada clasificador asociando probabilidad de la clase positiva dada la muestra x y L es el número de clasificadores.

3.3 Ensamble de clasificadores a nivel de datos

La figura 5 representa la metodología del ensamble con estrategia a nivel de datos. La cantidad de péptidos utilizados fue de 1200, distribuidos en 1008 péptidos antibacterianos y 192 péptidos no antibacterianos. El experimento consistió en realizar una división del conjunto de péptidos en un 10 % para prueba y un 90 % de entrenamiento. Teniendo los conjuntos creados, se realiza un balanceo de los datos con una cantidad igual a la clase minoritaria que en este caso son los péptidos no antibacterianos. En el balanceo se aplicó submuestreo sin reemplazo para garantizar que ningún dato esté repetido en otro conjunto y evitar una posible redundancia o pérdida de diversidad entre los clasificadores base. Luego de tener los conjuntos balanceados se crea un clasificador por cada conjunto. Las decisiones de los clasificadores se combinan a partir de las reglas de combinación media, máximo, mínimo, producto y mediana. El experimento se realiza 5 veces y se extrae el promedio del desempeño AUC tanto en los clasificadores base como en las reglas de decisión. Los algoritmos utilizados para la creación del ensamble a nivel de datos fueron: Máquinas de soporte vectorial *SVM* con $C = 1$, árboles de decisión *Dtree* con nivel de profundidad igual 3, k vecinos más cercanos *K-NN* con $k = 3$, Naive Bayes gausiano *GausianNB* y Análisis discriminante lineal *LDA*.

3.4 Ensamble de clasificadores a nivel de algoritmos

La figura 6 representa la metodología del ensamble con estrategia a nivel de algoritmos. En esta metodología se trabajó con 1200 datos. Este ensamble consiste en utilizar algoritmos de clasificación con parámetros relacionados con el desbalanceo de los datos. Cada clasificador se entrena con el mismo conjunto de datos desbalanceados y sus decisiones son combinadas por medio de las reglas de combinación establecidas al principio de la sección. En este ensamble se aplicó validación cruzada 10 (ver figura

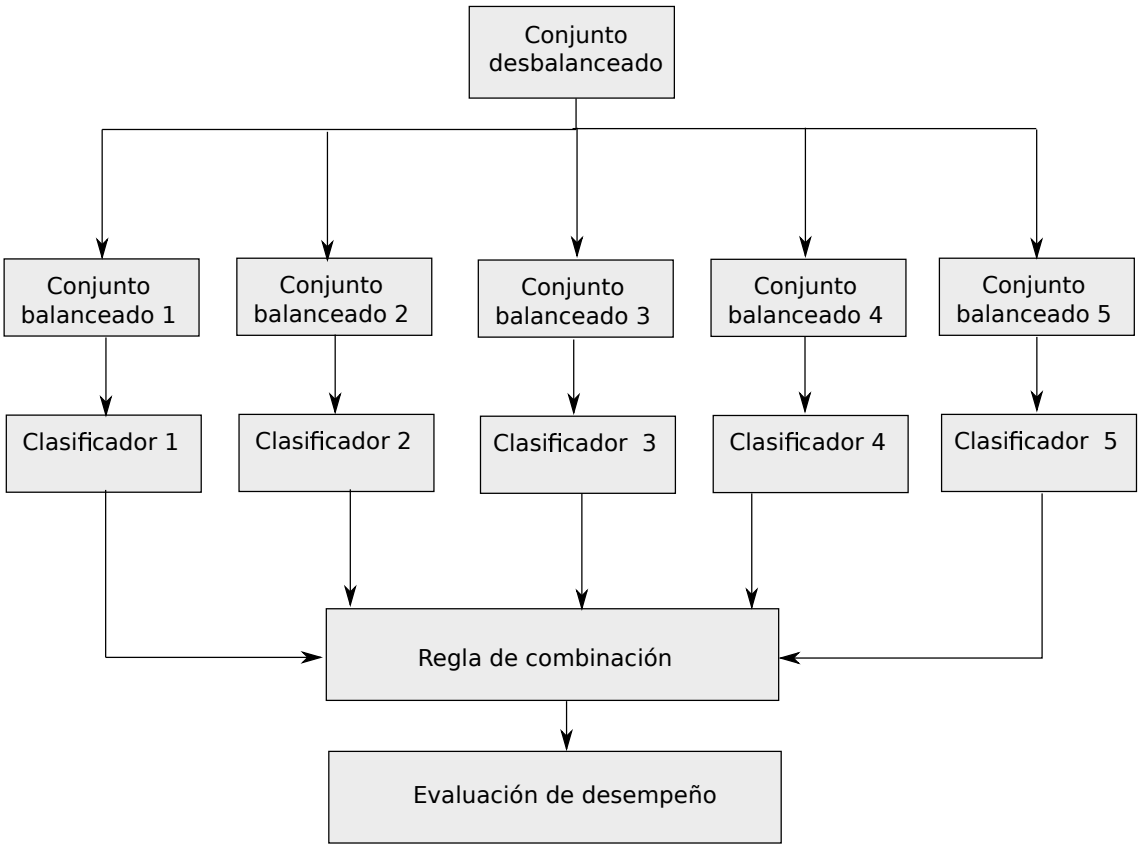


Figura 5: Ensamble con estrategia a nivel de datos

7) y muestreo estratificado para mantener el desbalanceo de los datos entre las clases. Para la construcción de este ensamble, se utilizaron tres algoritmos: *SVM*, *SGD* y *K-NN*. Los clasificadores fueron configurados teniendo en cuenta la combinación de parámetros establecidos por el cuadrado latino. El cuadrado latino se utilizó para aprovechar las posibles combinaciones de parámetros resultantes y se construye a partir de la formación de bloques en dos direcciones: filas y columnas. Por lo tanto, las filas y las columnas representan en realidad dos restricciones sobre la aleatorización. En general, un cuadrado latino para p factores o cuadrado latino $p \times p$, es un cuadrado con p renglones y p columnas. Cada una de las p^2 celdas resultantes contiene una de las p letras que corresponde a los tratamientos y cada letra ocurre una y sólo una vez en cada renglón y columna [16]. El diseño del cuadrado latino se realizó a partir de las diferentes combinaciones de parámetros de cada algoritmo de aprendizaje. Con el fin de comprender lo anteriormente dicho, la ecuación (1) es una matriz cuadrada 5×5 que representa las respectivas combinaciones de los parámetros de cada clasificador.

$$C_{latino} = \begin{pmatrix} c_1, k_1, \alpha_1 & c_1, k_2, \alpha_2 & c_1, k_3, \alpha_3 & c_1, k_4, \alpha_4 & c_1, k_5, \alpha_5 \\ c_2, k_1, \alpha_2 & c_2, k_2, \alpha_3 & c_2, k_3, \alpha_4 & c_2, k_4, \alpha_5 & c_1, k_5, \alpha_1 \\ c_3, k_1, \alpha_3 & c_3, k_2, \alpha_4 & c_3, k_3, \alpha_5 & c_3, k_4, \alpha_1 & c_3, k_5, \alpha_2 \\ c_4, k_1, \alpha_4 & c_4, k_2, \alpha_5 & c_4, k_3, \alpha_1 & c_4, k_4, \alpha_2 & c_4, k_5, \alpha_3 \\ c_5, k_1, \alpha_5 & c_5, k_2, \alpha_1 & c_5, k_3, \alpha_2 & c_5, k_4, \alpha_3 & c_5, k_5, \alpha_4 \end{pmatrix} \quad (35)$$

Los parámetros c_1, c_2, c_3, c_4 y c_5 son los parámetros de penalización de *SVM* y representan las filas en la matriz. Los parámetros k_1, k_2, k_3, k_4 y k_5 son los k-vecinos del clasificador k-vecinos más cercanos y representan las columnas en la matriz. Los parámetros α_1, α_2, α_3, α_4 y α_5 son los parámetros de regularización del clasificador *SGD*. La máquina de soporte vectorial fue configurada para trabajar con conjuntos desbalanceados. Se aplicó un peso (1:5) correspondiente a la relación de desbalanceo de las clases que conforman el conjunto de péptidos 1008/192=5.25 y con los parámetros de regularización C =$\{0.01, 0.1, 1, 10, 100\}$. Para configurar el gradiente descendiente estocástico (*SGD*), se trabajó como función de pérdida *huber* modificado. Ésta es una alternativa en que puede trabajar el *SGD* donde el gradiente actúa como máquina de soporte vectorial cuadráticamente suavizada, variando algunos parámetros [17]. Además, La función de pérdidas *huber* es utilizada en regresión robusta, y es menos sensible a los valores atípicos que la función de costo de error cuadrático. Por lo tanto, en muchos casos es muy utilizada en clasificación [18]. Para el *SGD* se aplicó un $\alpha = \{$ 0.001, 0.1, 1, 10, 1000 $\}$, con un número de iteraciones igual a 15000 y un peso $(1:5)$. Finalmente, para el algoritmo k-vecinos más cercanos se trabajó con k= $\{1, 2, 5, 10, 20\}$. Finalmente las decisiones obtenidas por cada clasificador se combinan por medio de las reglas de combinación: Media, máximo, mínimo, producto y mediana.

3.5 Resultados y Discusión

3.5.1 Análisis de dispersión y variabilidad en el desempeño de clasificadores base utilizados para el ensamble a nivel de algoritmos

En la figura 8 está representados los AUC promedio y desviaciones estándar de los clasificadores utilizados en el ensamble con estrategia a nivel de algoritmos. Respecto a la figura mencionada, los clasificadores de mejor desempeño promedio AUC fueron *SVM* y *K-NN*. Estos clasificadores presentaron poca dispersión y mantuvieron una variabilidad de 0.8 y 0.93 AUC promedio, los cuales son resultados considerablemente

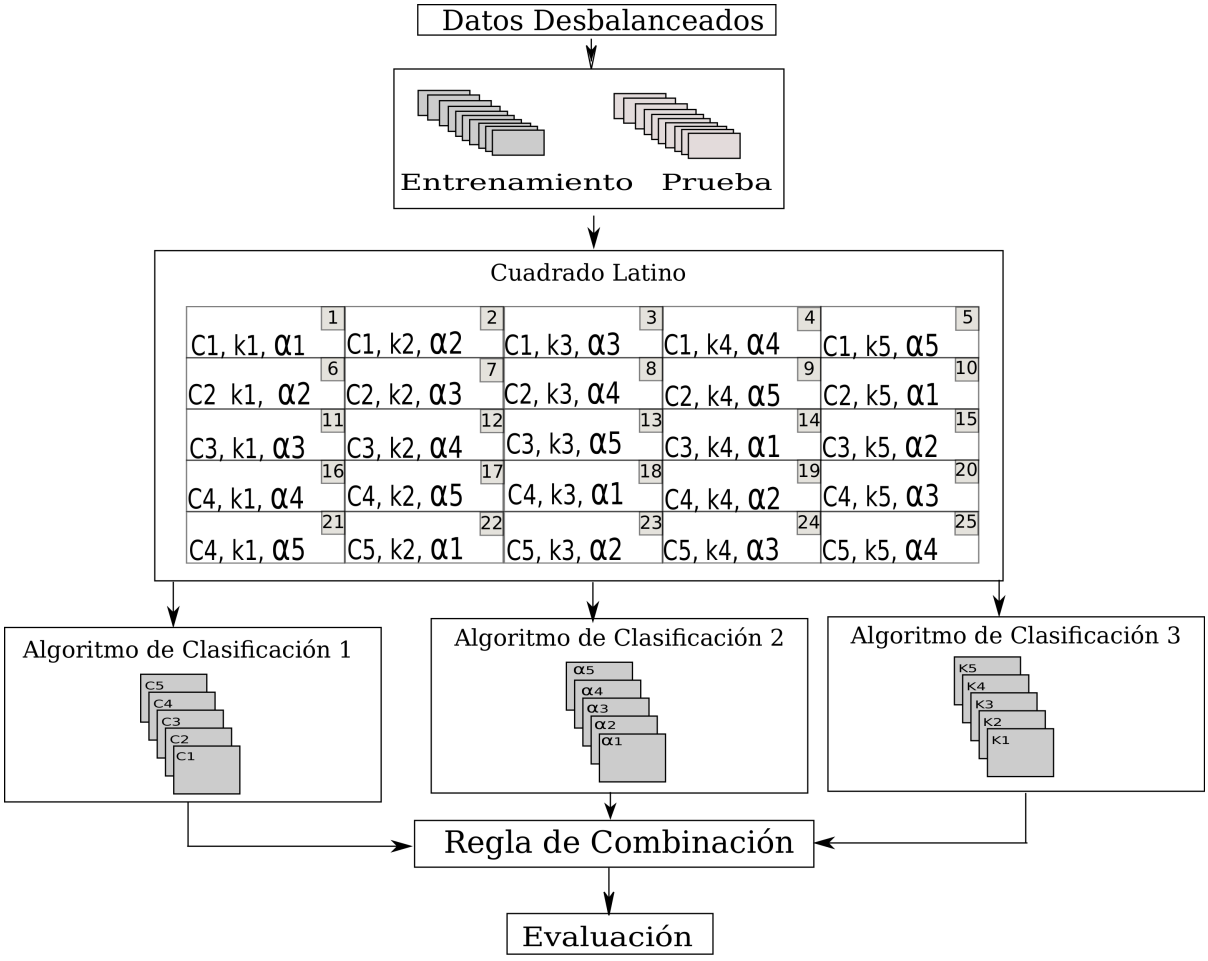


Figura 6: Ensamble con estrategia a nivel de algoritmos

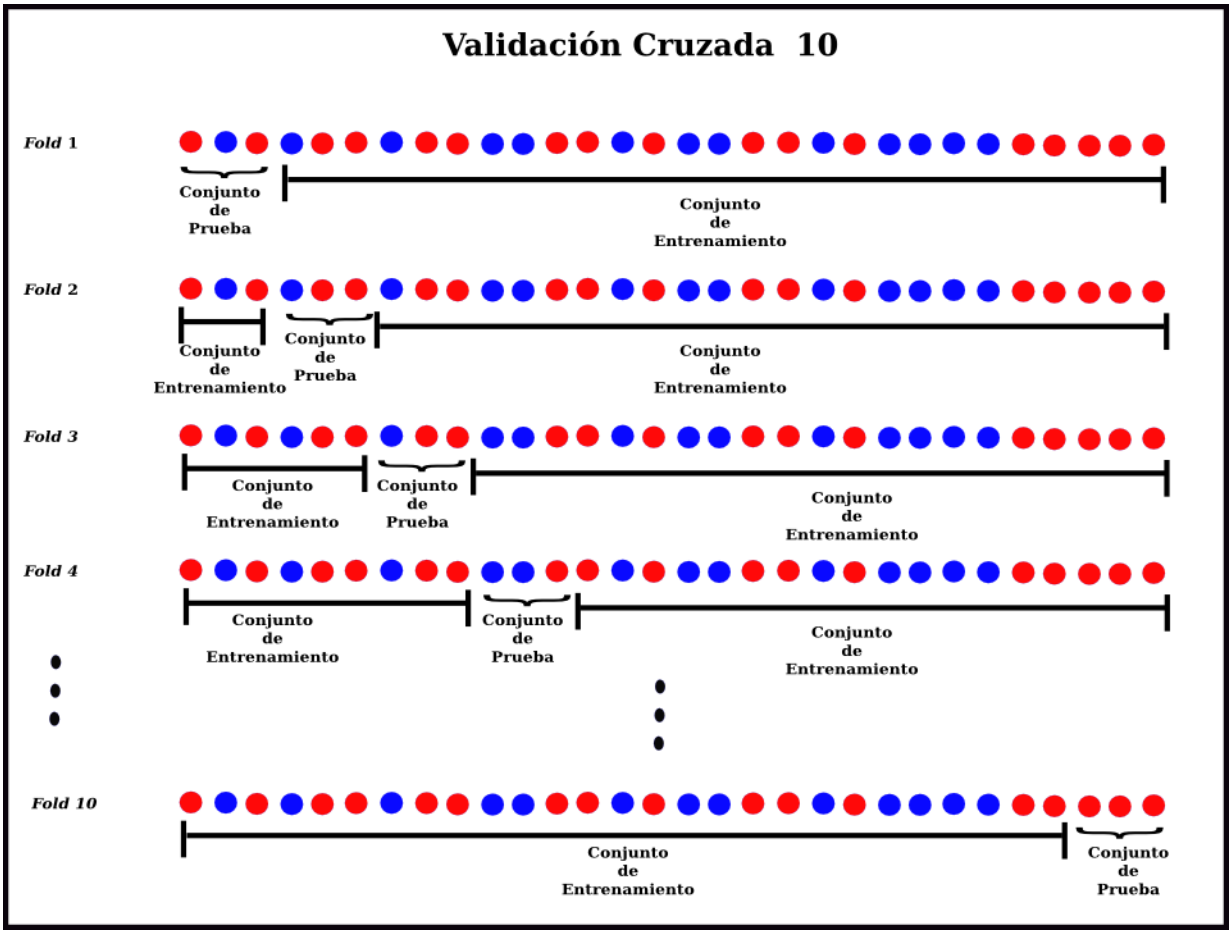


Figura 7: Validación Cruzada

buenos. Por otro lado, los clasificadores basados en *SGD* presentaron alta dispersión y variabilidad en casi todos los parámetros excepto en el primero. En resumen, la mayoría de clasificadores base arrojaron resultados buenos, aún así, se podría esperar mejores resultados al combinar las decisiones de cada uno de los clasificadores creados.

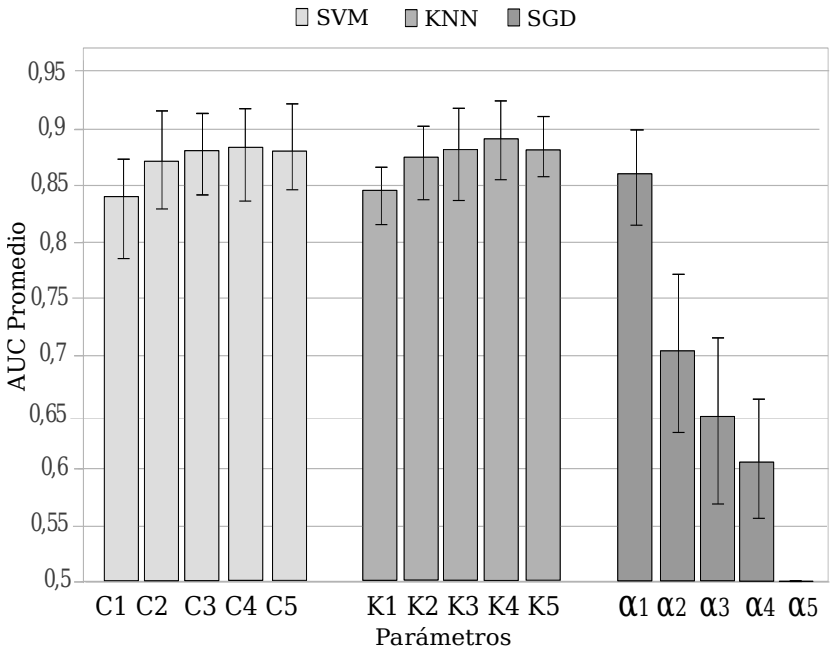


Figura 8: Representación de los desempeños promedios y sus desviaciones estándar del AUC de cada clasificador según los parámetros establecidos, donde $C1$, $C2$, $C3$, $C4$ y $C5$ son los parámetros de *SVM*; $K1$, $K2$, $K3$, $K4$ y $K5$ son los parámetros de *K-NN* y $\alpha 1$, $\alpha 2$, $\alpha 3$, $\alpha 4$ y $\alpha 5$ son los parámetros de *SGD*

3.5.2 Análisis de dispersión y variabilidad del desempeño promedio en cada regla de combinación utilizada en el ensamble de clasificadores a nivel de algoritmos

La figura 9 representa el AUC promedio de cada regla de combinación utilizando los clasificadores base mencionados. Para la representación de estos desempeños se utilizaron diagramas de cajas y bigotes. Éstos son el resultado de cada una de las configuraciones de parámetros establecidos por el cuadrado latino. Por lo tanto, la figura permite visualizar la dispersión de cada una de las observaciones establecidas por cada regla. Entonces, las reglas que mostraron menos dispersión entre sus desempeños fueron la media, el producto y la mediana. Sin embargo, se puede apreciar en la regla producto,

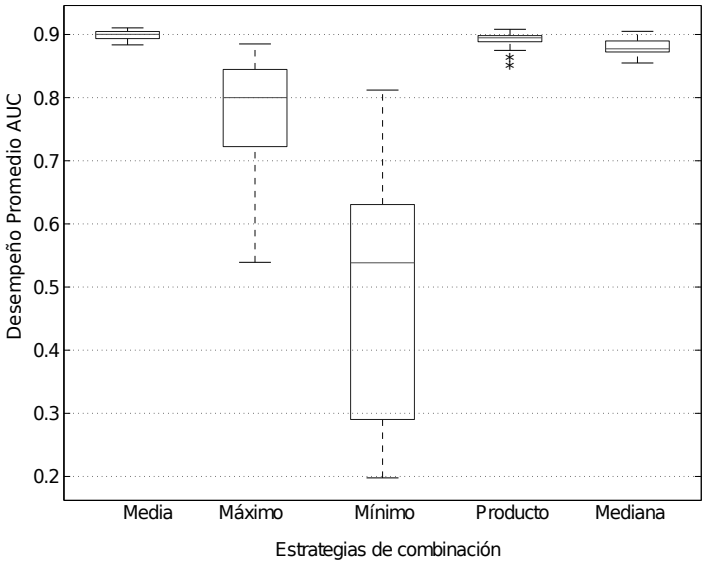


Figura 9: Diagrama de cajas para cada regla, el $*$ representa el valor atípico de las diferentes observaciones utlizando la combinacionación de los tres algoritmos: *SVM*, *SGD* y *K-NN*.

valores atípicos por tener observaciones numéricamente muy cercanas. Además, en casi todos los diagramas los desempeños promedios comprendidos entre el 25 % y el 50 % de la población de desempeños están más dispersas que el 50 % y el 75 % de la población, aun así, en el caso de la estrategia mediana ocurre que las medias comprendidas entre el 50 % y el 75 % de su población estan más dispersas que el 25 % y el 50 % de la misma. Lo anterior quiere decir que los desempeños más pequeños en la mayoría de la reglas se encuentran más alejados entre si que los grandes desempeños. Sin embargo, en la regla mediana ocurre lo contrario aunque no es tan notorio, puesto que las medidas númericamente tienen una alta concentración. En el 25 % de los desempeños promedio más pequeños y más grandes en las reglas media, producto y mediana tienen concentraciones de 0.85 y 0.91 AUC promedio. Sin embargo, en las reglas máximo y mínimo hay diferencias en el 25 % de las concentraciones de desempeños promedios más pequeños y más grandes.

En las figuras 10, 11, 12, 13 y 14 están representadas los desempeños promedio AUC de las diferentes estrategias de combinación por medio de diagrama de barras respectivamente según el número de combinaciones de parámetros. Por lo tanto, se realizaron dos ensambles: el primer ensamble está constituido por *SVM*, *SGD* y *K-NN* y el segundo ensamble tiene como clasificadores base *SVM* y *KNN*. Por consiguiente, el segundo ensamble se realizó con los mejores clasificadores base, puesto que el primer ensamble presentó variabilidades en algunas reglas de combinación con diferentes parámetros.

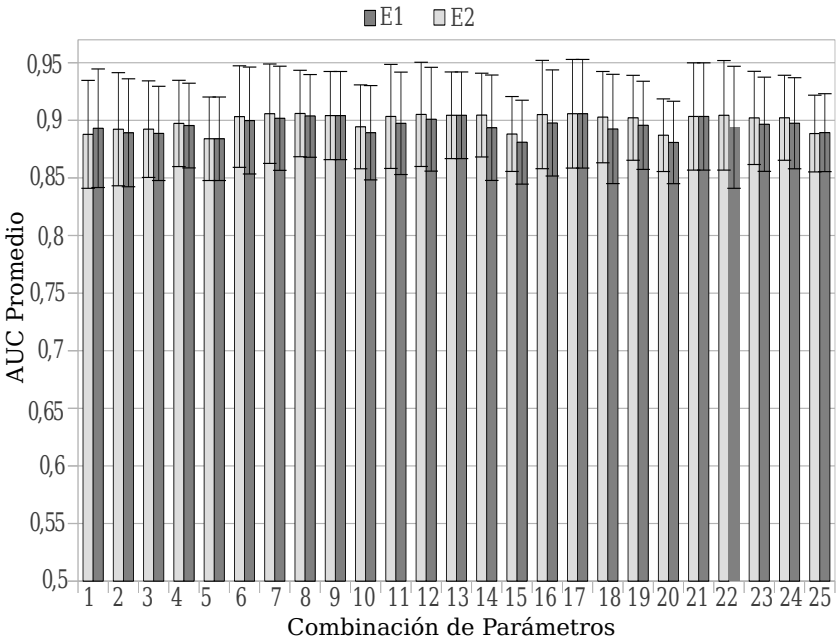


Figura 10: Representación de los desempeños promedio AUC y sus desviaciones estándar del AUC de cada configuración de parámetros según cuadrado latino con estrategia de combinación media. $E1$= (SGD+KNN+SVM) y $E2$= (KNN+SVM)

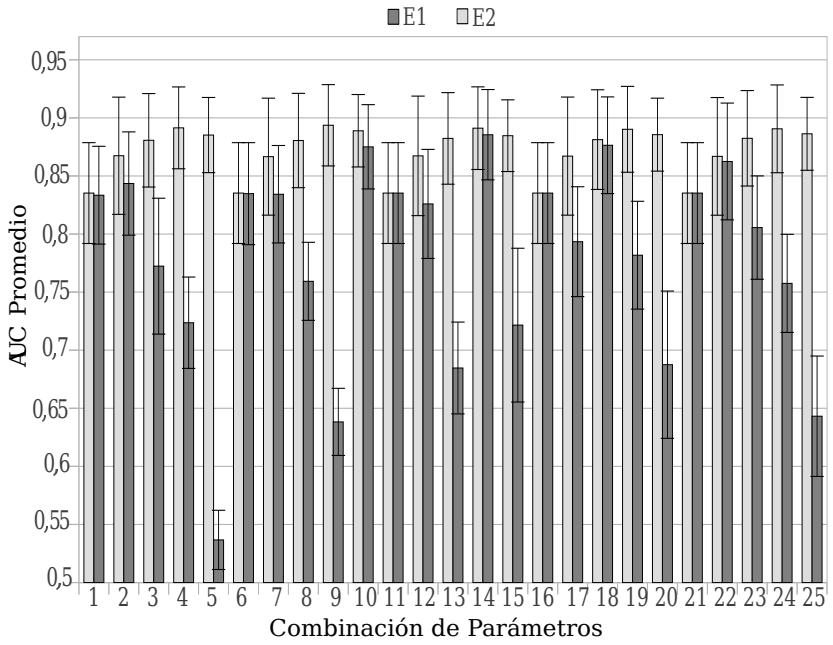


Figura 11: Representación de los desempeños promedio AUC y sus desviaciones estándar de cada configuración de parámetros según cuadrado latino con estrategia de combinación máximo. $E1$= (SGD+KNN+SVM) y $E2$= (KNN+SVM)

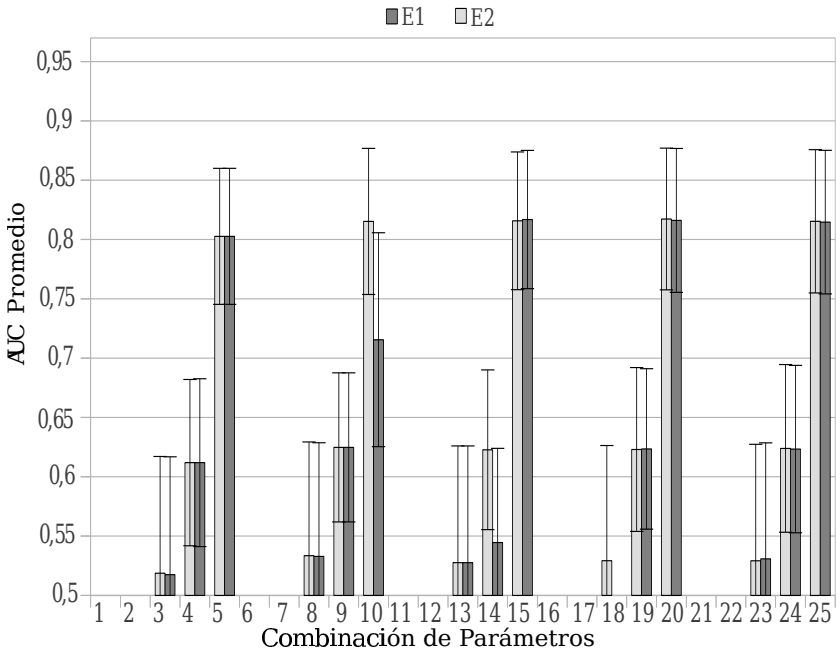


Figura 12: Representación de los desempeños promedios AUC y sus desviaciones estándar de cada configuración de parámetros según cuadrado latino con estrategia de combinación mínimo. $E1$= (SGD+KNN+SVM) y $E2$= (KNN+SVM)

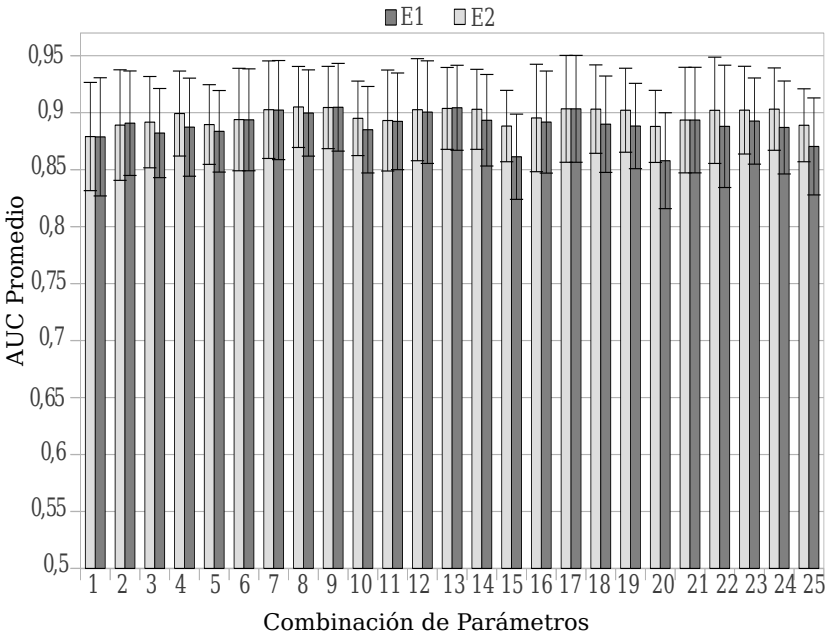


Figura 13: Representación de los desempeños promedios AUC y sus desviaciones estándar de cada configuración de parámetros según cuadrado latino con estrategia de combinación producto. $E1$= (SGD+KNN+SVM) y $E2$= (KNN+SVM)

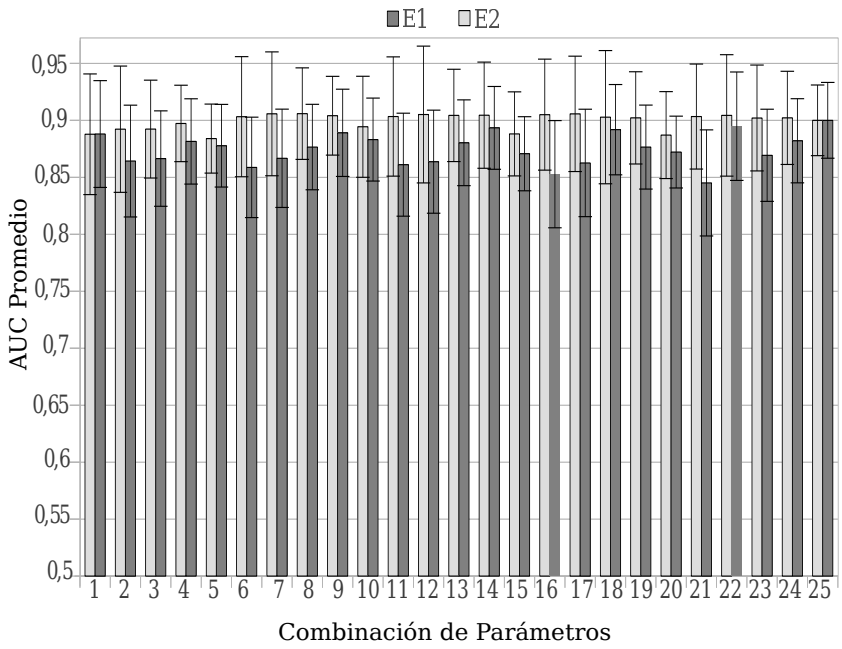


Figura 14: Representación de los desempeños promedios AUC y sus desviaciones estándar de cada configuración de parámetros según cuadrado latino con estrategia de combinación mediana. $E1$= (*SGD+KNN+SVM*) y $E2$= (*KNN+SVM*)

En la figura 10, se representan los desempeños promedio y desviaciones estándar del AUC de cada configuración de parámetros dado el cuadrado latino con regla media. Se puede apreciar que los promedios de cada ensamble E1(Ensamble con *SVM*, *K-NN* y *SGD*) y E2(*SVM*, *K-NN*) se mantienen concentrados numéricamente dadas las combinaciones de parámetros en los diferentes ensambles realizados. Por consiguiente, hay poca dispersión en la población de medias. La variabilidad presente en esta regla tiene un rango de valores máximos y mínimos de desempeño promedio AUC con valores de 0.84 y 0.95 respectivamente. Por lo tanto, en el peor de los casos el desempeño más pequeño podría ser un buen resultado. Se considera que es un buen resultado porque todas las combinaciones presentaron valores superiores a los clasificadores individuales y presentan una baja variabilidad.

La figura 11 representa los desempeños promedios y desviaciones estándar de AUC de cada configuración de parámetros dado el cuadrado latino con regla máximo. Se puede apreciar que hay un aumento de variabilidad en el primer ensamble (E1). Sin embargo, en el segundo ensamble (E2), hay un aumento de la concentración, la cual induce a disminuir un poco la dispersión de los desempeños promedios AUC. Por lo tanto, la mejora en variabilidad del segundo ensamble respecto al primero es significativa.

La figura 12 representa los desempeños AUC promedio y desviaciones estándar de cada configuración de parámetros dado el cuadrado latino con estrategia mínimo. Se puede apreciar un gran aumento en la variabilidad del desempeño AUC en el primer

ensamble. En este ensamble es notorio que algunos desempeños no superan el 50 % de desempeño AUC tanto en el primer ensamble (E1) como el segundo (E2). Por lo tanto, es una regla que produce un desempeño pobre y no es recomendado utilizarlo. De hecho, el resultado de esta regla es producto de las decisiones mínimas de los clasificadores base y al parecer, el desempeño pobre de un clasificador perjudica al ensamble.

La figura 13 representa los desempeños promedios AUC y desviaciones estándar de cada configuración de parámetros dado el cuadrado latino con estrategia producto. En los ensambles se presentan variabilidades muy pequeñas que en comparación con la regla media son notorias. Sin embargo, la variabilidad presente en las reglas mediana y producto no conllevan a un desempeño pobre.

La figura 14 representa los desempeños promedios AUC y desviaciones estándar de cada configuración de parámetros dado el cuadrado latino con estrategia mediana. Esta regla igual que la regla media y producto produjo muy buenos resultados en los ensambles creados. Sin embargo, se puede notar pequeñas diferencias entre el primer ensamble (E1) y el segundo (E2). En el primer ensamble (E1) los desempeños promedio AUC no superan el 0.9 de AUC promedio. En cambio, en el segundo ensamble (E2) sus desempeños promedio superan la cifra anterior. Aún así, el desempeño promedio AUC más pequeño en ambos ensambles de esta regla es superior a un AUC promedio de 0.8, esto significa que los ensambles son buenos.

En general, las reglas media, producto y mediana tuvieron mejores desempeños que las reglas máximo y mínimo. Adicionalmente, comparando el ensamble a nivel de algoritmos con algunos clasificadores base, se puede apreciar que hubo mejoras en los resultados obtenidos. Por lo tanto, en estas reglas hubo un incremento leve en el desempeño promedio AUC y los resultados son más estables.

3.5.3 Análisis de dispersión y variabilidad del desempeño promedio en los clasifcadores base y cada regla de combinación utilizada en el ensamble de clasificadores a nivel de datos

La figura 15, representa los diferentes desempeños promedio de los clasificadores base y sus respectivas reglas de combinación. Con respecto a la figura, los clasificadores base y las reglas de combinación presentaron variabilidades. Sin embargo, los clasificadores basados en máquinas de soporte vectorial, k-vecinos más cercanos y *Naive Bayes* Gaussiano presentaron resultados con menor variabilidad. Aún así, los resultados del ensamble con los clasificadores base mencionados no son tan comprometedores, debido a que se mantiene la variabilidad. En cuanto a las reglas de combinación, la reglas mínimo y máximo proporcionan en el ensamble los resultados más pobres. Esto quiere decir, que hay que tener cuidado con estas reglas, pues al parecer son muy sensibles a las decisiones erróneas de los clasificadores base. Por otro lado, la regla media ofrece mejores resultados que las demás reglas. Aunque, cierta regla promete dar buenos resultados,

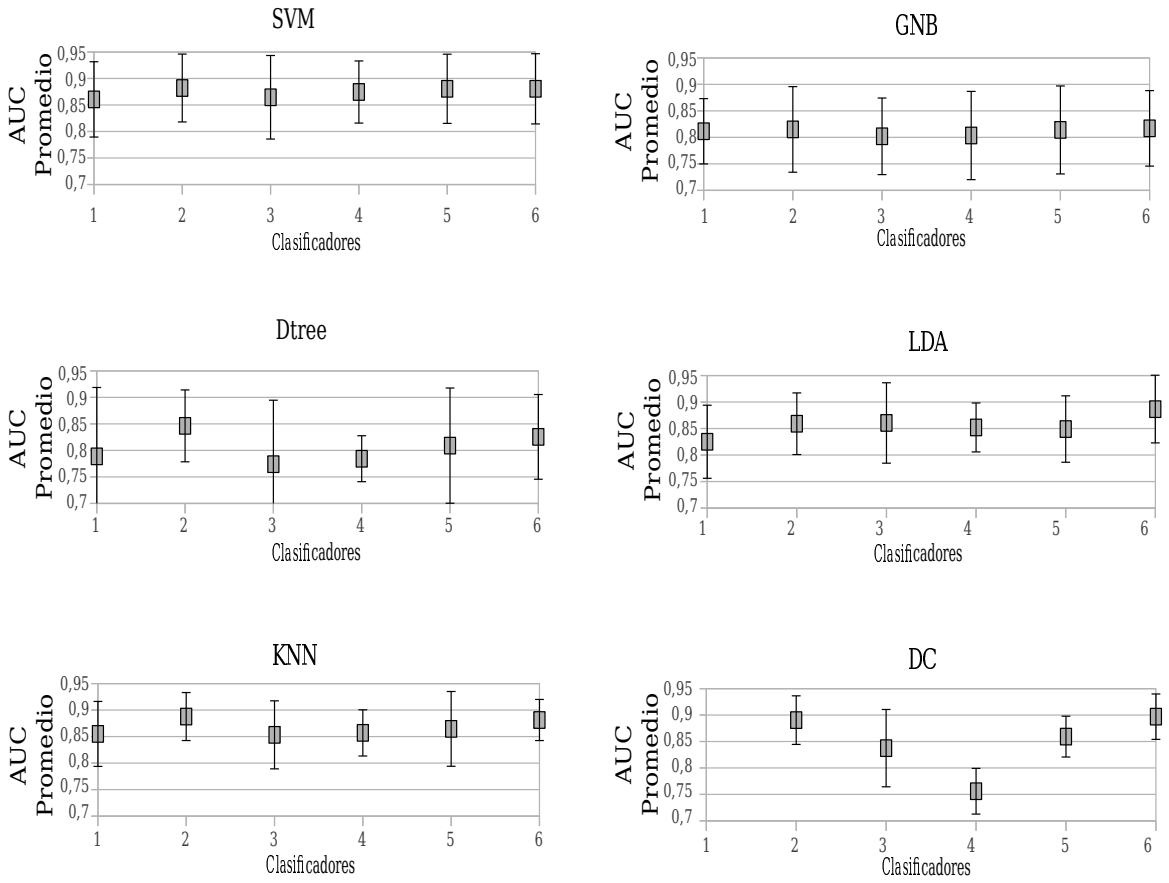


Figura 15: Representación de los desempeños promedios AUC de los clasificadores base y las reglas de combinación en el ensamble a nivel de datos. El clasificador 1 representa el clasificador base y los números del 2 al 6 representan las reglas de combinación: media, máximo, mínimo, producto, mediana, respectivamente. DC significa diferentes clasificadores, en este caso, solo se aprecia el desempeño de las reglas de combinación puesto que es la fusión de las decisiones de los diferentes algoritmos de clasificación mencionados en la figura.

éstos no proporcionan una diferencia significativa ante los resultados de cualquier clasificador base, de manera que en ciertas circunstancias un clasificador base puede llegar a obtener mejores resultados que el ensamble.

3.5.4 Comparación entre las técnicas de ensamble utilizadas para solucionar el problema del desbalanceo de datos y métodos de ensamble populares

En ambas estrategias se puede observar en qué circunstancias mejoran los desempeños y las potencialidades que aporta cada una. En algunos casos, en el ensamble a nivel de algoritmos las mejoras en cada una de las reglas de combinación no dependieron del desempeño de los clasificadores base sino de algunas de las reglas de combinación. De hecho, se puede observar que en la reglas de combinación las cuales tienen en cuenta todas las decisiones para dar la mejor decisión como: media y producto produjeron valores que no estuvieron afectados por el clasificador con peor desempeño. En cambio, con la regla máximo y mínimo se pueden apreciar desempeños pobres que puden ser producidos por las decisiones erróneas de los clasificadores base. Por lo tanto, en algunos casos, el desempeño dependió del tipo de regla utilizado. En otros casos, la influencia de un clasificador pobre conllevó a un desempeño regular en algunas reglas.

En cuanto al ensamble a nivel de datos, la mayoría de clasificadores base tuvieron un comportamiento con ciertas variabilidades con diferentes conjuntos de entrenamiento. Sin embargo, hubo clasificadores que no resaltaron tanta variabilidad, como: máquina de soporte vectorial (SVM) y k-vecinos más cercanos. Con respecto a las reglas combinación, hubo desempeños buenos por parte de las reglas media, producto y mediana. Sin embargo, las reglas mínimo y máximo fueron las reglas que tuvieron desempeños pobres.

Los resultados obtenidos en cada estrategia de ensamble a nivel de algoritmos y datos se presentan en la tabla 8. Adicionalmente se resaltan los mejores desempeños promedio que obtuvieron las reglas de combinación utilizadas en este trabajo. También, se incluyeron los desempeños de ensambles basados en estrategias a nivel de algoritmos.

Los parámetros tenidos en cuenta para el primer ensamble fueron: $C = 10$, $k = 2$ y $\alpha = 1000$. y para el segundo ensamble fueron: $C = 1$ y $k = 5$. Cualquier conjunto de parámetros puede ser escogido debido a que los desempeños en las distintas combinaciones no presentan diferencias significativas.

Adicionalmente, al realizar una comparación con el desempeño de los mejores clasificadores base no hubo diferencias significativas. La diferencia significativa se apreció en los clasificadores base *SGD* y en la reglas de combinación mínimo y máximo en la mayoría de veces.

Con respecto a los clasificadores individuales utilizados en las estrategias anteriormente mencionadas, en la tabla 9 se presentan los mejores desempeños AUC para los clasificadores base con datos balanceados y desbalanceados. Estos clasificadores presen-

taron desempeños similares a las reglas(ver tabla 8), por lo cual no se observó alguna diferencia significativa en utilizar datos desbalanceados o balanceados. Más aún, los clasificadores individuales con datos balanceados tienen desempeños cercanos a las reglas de combinación dadas, por tanto, habría que análizar los beneficios e implicaciones que podrían darse al utilizar ambas metodologías. Por otro lado, en el ensamble a nivel de algoritmos, los clasificadores base *SVM*, *KNN* (Ver tabla 9) tuvieron buenos resultados igual que las reglas de combinación planteadas.

En los ensambles populares basados en boosting como el *Adaboost* y el *gradiente boosting* y el *Random Forest* se configuraron con 5, 10, 20, 30 y 50 clasificadores base y los desempeños AUC promedio en cada uno estuvieron cercanos entre si, es decir, el número de clasificadores utilizados no afecto en el desempeño del ensamble. Aún así, en el ensamble *Adaboost* con clasificador base *SGD* el desempeño fue regular comparado con los demás ensambles.

Tabla 8: Representación de las mejores áreas bajo la curva (*AUC's*) y sus respectivas reglas de combinación de los ensambles utilizados.

Clasificadores Base	Media	Máx.	Mín.	Producto.	Mediana.
SVM	**0.88**	**0.86**	**0.87**	**0.88**	**0.88**
Dtree	**0.84**	**0.77**	**0.78**	**0.8**	**0.82**
K-NN	**0.88**	**0.85**	**0.85**	**0.86**	**0.88**
GaussianNB	**0,81**	**0.80**	**0.80**	**0.81**	**0.81**
LDA	**0.88**	**0.85**	**0.85**	**0.87**	**0.88**
DC	**0.89**	**0.83**	0.75	**0.85**	**0.89**
SVM+SGD+KNN	**0.906**	**0,793**	0,312	**0,903**	**0,862**
SVM+KNN	**0.906**	**0.880**	0.533	**0,905**	**0,906**

Por otra parte, para validar los resultados en cada técnica, se hizo un análisis estadístico por medio de $p - value$, el cual ayudó a determinar la existencia de posibles diferencias significativas entre los resultados obtenidos. Entonces, se compararon los desempeños AUC de los clasificadores base y sus reglas de combinación dada la estrategia. Los AUC promedios resaltados en negrita producidos por diferentes métodos de clasificación representaron un buen desempeño, en otro caso, según el análisis de $p - value$, las medidas AUC no resaltadas de métodos de clasificación no se consideraron como buenos desempeños.

Tabla 9: Representación de los mejores desempeños promedio áreas bajo la curva (*AUC's*) de los clasificadores base con datos balanceados y desbalanceados. SVM trabajó con $c = 1$ y SGD con $\alpha = 0{,}01$ con relación balanceada y desbalanceada

Clasificadores Base	AUC datos Balanceados	AUC datos Desbalanceados
SVM $(1:1)$	**0.86**	**0.87**
Dtree	**0.84**	0.76
K-NN	**0.88**	**0.88**
GaussianNB	**0.81**	**0.85**
LDA	**0.86**	**0.88**
SGD$(1:1)$		**0.86**
SVM $(1:5)$		**0.87**
SGD $(1:5)$		**0.85**

Tabla 10: Representación de los desempeños promedio áreas bajo la curva (*AUC's*) de los ensambles populares con datos desbalanceados. El clasificador base SVM trabajó con $c = 1$ y el SGD con $\alpha = 0{,}01$. Ambos clasificadores base se configuraron con relación balanceada y desbalanceada

Ensambles Populares	AUC datos Balanceados	AUC datos Desbalanceados
Adaboost$(SVM\ (1:1))$	**0,85**	**0,86**
Adaboost$(SGD\ (1:1))$	**0,81**	0,75
Adaboost $(Dtree)$	**0,88**	**0,91**
Adaboost $(SVM\ (1:5)$	**0,84**	**0.87**
Adaboost $(SGD\ (1:5))$	0,78	0.72
Random Forest	**0,90**	**0.91**
Gradient Adaboost	**0,89**	**0.90**

4. Conclusiones y Trabajos Futuros

En este trabajo se propusieron y evaluaron metodologías de ensamble con estrategias a nivel de datos y a nivel de algoritmos para clasificar péptidos en condición desbalanceada. Por lo tanto, se trabajó con varios algoritmos de clasificación: Máquinas de soporte vectorial *SVM*, k-vecinos más cercanos *K-NN*, gradiente descendiente estocástico *SGD*, árboles de decisión *Dtree*, Naive Bayes gausiano *GausianNB* y análisis discriminante lineal *LDA*. Las decisiones de los clasificadores fueron combinadas a partir de las reglas: Media, máximo, mínimo producto y mediana.

En el ensamble a nivel de algoritmos el AUC promedio de algunas reglas dependió de la estabilidad de los algoritmos individuales que construyeron dicho ensamble. El anterior comportamiento se vio reflejado en la regla máximo, ésta al trabajar con los clasificadores base estables, mejoró la decisión. En la regla mínimo no se vio reflejado ningún cambio, siempre reflejó el mismo comportamiento. Por otro lado, las reglas media, producto y mediana mantuvieron buenos resultados y no dependieron del rendimiento que obtuvo el clasificador base pobre.

En el ensamble a nivel de datos los desempeños promedios AUC dependieron de que tan organizados se encontraban los datos. Es probable que en este experimento los datos presentaron un verdadero muestreo aleatorio, debido a que los clasificadores base tuvieron AUC promedio cercanos en la mayoría de veces. Los clasificadores base presentaron resultados similares a los de las reglas de combinación. Sin embargo, al trabajar con diferentes clasificadores hubo una diferencia significativa entre las reglas producto, mediana y mínimo.

Al comparar los clasificadores base con sus respectivas reglas de combinación en la mayoría de veces los resultados fueron similares. En el caso del ensamble a nivel de algoritmos, los clasificadores base como: Máquinas de soporte vectorial y k vecinos más cercanos tuvieron buenos desempeños comparados con el desempeño del gradiente descendiente estocástico. Sin embargo, este último clasificador basado en gradiente al trabajar con un parámetro α pequeño puede dar un buen desempeño.

A nivel de datos los clasificadores base tienen buenos desempeños. Sin embargo, el clasificador basado en árboles de decisión presentó una gran variabilidad que afectó el desempeño de algunas reglas de combinación. Los clasificadores base con parámetros balanceados como la máquina de soporte vectorial y el gradiente descendiente estocástico presentaron resultados parecidos en algunas ocasiones. Al comparar estos resultados

con los mismos clasificadores configurados de manera desbalanceada se llega a que no hay diferencias significativas. Incluso, al comparar los resultados de los algoritmos anteriormente mencionados con otros resultados de algoritmos de clasificación, no existe un resultado predominante. También, los desempeños de los clasificadores base no se vieron tan afectados al trabajar los datos tanto en condición balanceada como en condición desbalanceada.

En los ensambles populares el número de clasificadores base no afectó en el desempeño del ensamble, es decir, un pequeño conjunto de clasificadores base puede tener el mismo desempeño que trabajar con una gran cantidad de clasificadores base en estos tipos de métodos. Sin embargo, el ensamble *Adaboost* con clasificadores base *SGD* produjo un desempeño pobre comparado con las demás técnicas. También, en casi todas estas técnicas los resultados no se vieron afectados al trabajar los datos en condición desbalanceada.

En general, la mayoría de las técnicas utilizadas trabajan bien o mantienen desempeños similares independientemente de la condición de los datos. Aun así, hay que tener cuidado en el uso de ciertas técnicas. En el ensamble a nivel de algoritmos el costo computacional es alto y esto es debido al uso de clasificadores con muchos parámetros. Incluso, en esta técnica hay clasificadores base que tienen también un alto costo computacional como son las máquinas de soporte vectorial y el gradiente descendiente estocástico. Por otro lado, el ensamble a nivel de datos puede verse afectado por el proceso de muestreo de los datos. En este caso hubo un submuestreo sin reemplazo de la clase mayoritaria para garantizar que ningún dato de cierta clase este repetido en el entrenamiento del clasificador. Este proceso de muestreo podría implicar una pérdida de información importante y puede afectar en la clasificación.

Como trabajos futuros según los resultados de este trabajo de investigación se sugiere trabajar con clasificadores individuales debido a que un ensamble puede ser más costoso computacionalmente o puede haber una pérdida de información importante en el momento de submuestrear los datos que nos interesa. Además, en la mayoría de resultados, entre los clasificadores individuales y el ensamble no hubo diferencias significativas que concluyan que una técnica es mejor que la otra. Sin embargo, se podría profundizar con ensambles tanto a nivel de algoritmos como a nivel de datos. Si se cuenta con un centro de computo de alto desempeño se podría realizar un ensamble a nivel de algoritmos y trabajar con más parámetros y hacer un barrido más amplio en la medida que haya claridad en un desempeño complementario de los clasificadores. Adicionalmente, se podrían implementar métodos de computación paralela que permitan la acción simultánea de varios algoritmos de clasificación. También, se podría trabajar con diferentes técnicas de sobremuestreo que favorezcan la reducción del efecto del desbalanceo.

También, hay que explorar con más detalle los datos según su representación en el espacio de características. Este trabajo de investigación solo aplicó diferentes técnicas de clasificación asumiendo una selección de características dada por un trabajo previo. Es importante encontrar características con la finalidad de aumentar la propie-

dad discriminante en un clasificador. Esto podría producir una clasificación con mejor precisión.

Bibliografía

AIDOS, H., DUIN, R. P. W., FRED, A. L. N. (2013). The Area under the ROC Curve as a Criterion for Clustering Evaluation. In Proc International Conf. on Pattern Recognition Applications and Methods - ICPRAM (pp. 1˘5). Barcelona, España.

BARANDELA, R., SÁNCHEZ, J., GARCÍA, V., RANGEL, E. (2003). Strategies for learning in class imbalance problems. Pattern Recognition, 36(3), 849–851.

BARNES, M. R. (2007). Bioinformatics for Geneticists Second Edition A bioinformatics primer for the analysis of genetic data. Breast Cancer Research and Treatment (Second Edi.). Harlow, Essex, UK: John Wiley and Sons, Ltd.

BHOWAN, U., JOHNSTON, M., ZHANG, M., YAO, X. (2013). Evolving Diverse Ensembles Using Genetic Programming for Classification With Unbalanced Data. IEEE Transactions on Evolutionary Computation, 17(3), 368–386.

BLAGUS, R., LUSA, L. (2010). Class prediction for high-dimensional class-imbalanced data. BMC Bioinformatics, 11, 1–17.

BRADLEY, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognition, 30(7), 1145–1159.

BRAGA, A. C., OLIVEIRA, P. (n.d.). Diagnostic analysis based on ROC curves: theory and applications in medicine. International Journal of Health Care Quality Assurance, 16(4), 191–198.

BREU, F., GUGGENBICHLER, S., WOLLMANN, J. (2004). Machine Learning ECML 2004. In D. (Eds. . Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi (Ed.), 15th European Conference on Machine Learning Pisa, Italy, September 2004 Proceedings (Vol. 3201, pp. 51–62). Pisa, Italia: Springer.

CAO, D.-S., XU, Q.-S., LIANG, Y.-Z. (2013). propy: a tool to generate various modes of Chou's PseAAC. Bioinformatics, 29(7), 960–962.

CAO, P., YANG, J., LI, W., ZHAO, D., ZAIANE, O. (2013). Ensemble-based hybrid probabilistic sampling for imbalanced data learning in lung nodule CAD. Computerized Medical Imaging and Graphics : The Official Journal of the Computerized Medical Imaging Society.

CAO, P., ZHAO, D., ZAIANE, O. (2013). Cost sensitive adaptive random subspace ensemble for computer-aided nodule detection. In Proceedings of the 26th IEEE International Symposium on Computer-Based Medical Systems (pp. 173–178), IEEE.

CHAWLA, N. V, BOWYER, K. W., HALL, L. O., KEGELMEYER, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research, 16, 321–357.

CHOU, K. (2000). Prediction of Protein Subcellular Locations by Incorporating Quasi-Sequence-Order Effect. Biochemical and Biophysical Research Communications, 278(2), 477–483.
COVER, T., HART, P. (1967). Nearest neighbor pattern classification. IEEE Transactions on Information Theory, 13(1), 21–27.

HARO-GARCIA, A., GARCIA-PEDRAJAS, N. (2011). A scalable method for instance selection for class-imbalance datasets. In 2011 11th International Conference on Intelligent Systems Design and Applications (Vol. 1, pp. 1383–1390). IEEE.

DIETTERICH, T. G. (n.d.). Ensemble Methods in Machine Learning. In First International Workshop, MCS 2000 Cagliari, Italy, June 21–23, 2000 Proceedings (Springer., Vol. 1857, pp. 1–15). Cagliary, Italia.

DOMINGOS, P. (1999). MetaCost: A General Method for Making Classifiers Cost-Sensitive. In In Proceedings of the 5th ACM S16KDD International Conference on Knowledge Discovery and Data Mining (pp. 155–164). San Diego, CA.

DUBCHAK, I., MUCHNIK, I., HOLBROOK, S. R., KIM, S. H. (1995). Prediction of protein folding class using global description of amino acid sequence. Proceedings of the National Academy of Sciences of the United States of America, 92(19), 8700–8704.

DUBEY, R., ZHOU, J., WANG, Y., THOMPSON, P. M., YE, J. (2013). Analysis of sampling techniques for imbalanced data: An n=648 ADNI study. NeuroImage, 87C, 220–241.

DUCH, A., MASULLI, P., PALM, G. (2012). Artificial Neural Networks and Machine Learning–ICANN 2012. In A. E. P. V. alessandro. villa@unil. c. (16) (Ed.), 22nd International Conference on Artificial Neural Networks, Lausanne, Switzerland, September 11-14, 2012, Proceedings, Part I (Vol. 7552, p. pp 1–8). Heidelberg, Springer Berlin.

ELKAN, C. (2001). The Foundations of Cost-Sensitive Learning. In In Proceedings of the 17th International Joint Conference on Artificial Inteligence. (pp. 973–978). Seatle, Washington.

FLEISS, L., LEVIN, B., PAIK, M. C. (1981). The measurement of interrater agreement. In In Statistical methods for rates and proportions (2nd ed (pp. 212–236). Wiley.

GALAR, M., FERNANDEZ, A., BARRENECHEA, E., BUSTINCE, H., HERRERA, F. (2012). A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews).

GAO, K., KHOSHGOFTAAR, T. M., NAPOLITANO, A. (2012). A Hybrid Approach to Coping with High Dimensionality and Class Imbalance for Software Defect Prediction. In 2012 11th International Conference on Machine Learning and Applications (pp. 281–288).

GARCÍA, E. A. (2009). Learning from Imbalanced Data. IEEE Transactions on Knowledge and Data Engineering, 21(9), 1263–1284.

GARCÍA, V., MOLLINEDA, R. A., SÁNCHES, J. S. (2007a). On the k-NN performance in a challenging scenario of imbalance and overlapping. Pattern Analysis and Applications, 11(3-4), 269–280.

GARCÍA-PEDRAJAS, N., GARCÍA-OSORIO, C. (2011). Constructing ensembles of classifiers using supervised projection methods based on misclassified instances. Expert Systems with Applications, 38(1), 343–359.

GIANCINTO, G., ROLI, F. (2001). Design of Effective Neural Network Ensembles for Image Classification Purposes. Image Vision And Computing Journal, 19, 699–707.

GODASE, A., ATTAR, V. (2012). Classifier Ensemble for Imbalance Data Stream Classification. In Proceedings of the CUBE International Information Technology Conference on - CUBE '12 (pp. 284–289). New York, New York, USA: ACM Press.

GORDON, Y. J., ROMAOWSKI, E. G., MCDERMOTT, A. M. (2005). A review of antimicrobial peptides and their therapeutic potential as anti-infective drugs. Current Eye Research, 30(7), 505–515.

GUYON, I., GUNN, S., (eds.), M. N. (2006). Feature extraction : foundations and applications. Berlin: Springer.

HaNSEN, L. K., SALAMON, P. (1990). Neural network ensembles. IEEE Transactions on Pattern Analysis and Machine, 12(10), 993–1001.

HUANG, H.-Y., LIN, Y.-J., CHEN, Y.-S., LU, H.-Y. (2012). Imbalanced data classification using random subspace method and SMOTE. The 6th International Conference on Soft Computing and Intelligent Systems, and The 13th International Symposium on Advanced Intelligence Systems, 817–820.

I.KUCHEVA, L. (n.d.). Combining Pattern Classifiers. John Wiley and Sons.

KHALILIA, M., CHAKRABORTY, S., POPESCU, M. (2011). Predicting disease risks from highly imbalanced data using random forest. BMC Medical Informatics and Decision Making, 11, 1–13.

KOHAVI, R., WOLPERT, D. H. (1996). Bias Plus Variance Decomposition for Zero-One Loss Functions. In Machine Learning: Procedings Of The Thirteenth International (pp. 275–283).

KOTSIANTIS, S. B., ZAHARAKIS, I. D., PINTELAS, P. E. (2007). Machine learning: a review of classification and combining techniques. Artificial Intelligence Review, 26(3), 159–190.

KRAWCZYK, B., SCHAEFER, G., WOZNIAK, M. (2013). An evaluation of classifier ensembles for class imbalance problems. In 2013 International Conference on Informatics, Electronics and Vision (ICIEV) (pp. 1–4). IEEE.

KRZANOWSKI, W., PARTRIDGE, D. (1997). Software Diversity: Practical Statistics for its Measurement and Exploitation. Information and Software Technology, 39, 39–707.

KUKAR, M., KONONENKO, I. (n.d.). Cost-Sensitive Learning with Neural Networks. In H. Prade (Ed.), ECAI 98. 13th European Conference on Artificial Intelligence (pp. 445–449). JohnWiley and Sons, Ltd.

LANDAU, L. (1937). N. IN I. SONG, J. EDER, T. M. NGUYEN (Eds.), Data Warehousing and Knowledge Discovery (pp. 283–292). Turin, Italia: Septiembre 2008.

LI, Q., YANG, B., LI, Y., DENG, N., JING, L. (2012). Constructing support vector machine ensemble with segmentation for imbalanced datasets. Neural Computing and Applications, 22(S1), 249–256.

LIAO, J., SHIH, C., CHEN, T., HSU, M. (2014). An ensemble-based model for two-class imbalanced financial problem. Economic Modelling, 37, 175–183.

LIN, W.J., CHEN, J. J. (2013). Class-imbalanced classifiers for high-dimensional data. Briefings in Bioinformatics, 14(1), 13–26.

LIU, X., ZHOU, Z. (2006). The Influence of Class Imbalance on Cost-Sensitive Learning: An Empirical Study. In Sixth International Conference on Data Mining (ICDM'06) (pp. 970–974). Ieee.

LÓPEZ, V., DEL RÍO, S., BENÍTEZ, J. M., HERRERA, F. (2014). Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data. Fuzzy Sets and Systems, 1, 1–34.

MALOOF, M. A. (2003). Learning When Data Sets are Imbalanced and When Costs are Unequal and Unknown (pp. 1-8).

MCCARTHY, K., ZABAR, B., WEISS, G. (2005). Does cost-sensitive learning beat sampling for classifying rare classes? Proceedings of the 1st International Workshop on Utility-Based Data Mining - UBDM '05, 69–77.

METZ, C. (1978). Basic principles of ROC analysis. Seminars in Nuclear Medicine, 8(4), 283–298.

NAPIERAZA, K., STEFANOWSKI, J., WILK, S. (2010). Learning from imbalanced data in presence of noisy and borderline examples. In Rough Sets and Current Trends in Computing (Vol. 6086, pp. 158–167). Warsaw, Polonia: Springer Berlin Heidelberg.

NIJNIK, A., HANCOCK, R. (2009). Host defence peptides: antimicrobial and immunomodulatory activity and potential applications for tackling antibiotic-resistant infections. Emerging Health Threats Journal, 2, 1–7.

SELZER, R.J. MARHOFER, and A. ROHWER. (2008). Applied Bioinformatics An Introduction. (P. M. Selzer, R. J. Marhöfer, A. Rohwer, Eds.)October. Berlin, Heidelberg: Springer Berlin Heidelberg.

POLIKAR, R. (2006). Ensemble based systems in decision making. Circuits and Systems Magazine, IEEE, 6(3), 21–45.

PORTO, W., SILVA, O., and FRANCO, O. (2012). Prediction and Rational Design of Antimicrobial Peptides. In E. Faraggi (Ed.), Protein Structure, InTech (pp. 377–396). Brazil: InTech.

RADTKE, P. V. W., GRANGER, E., SABOURIN, R., GORODNICHY, D. (2012). Adaptive Selection of Ensembles for Imbalanced Class Distributions. In Proceedings

of the 21st International Conference on Pattern Recognition (ICPR2012) (Vol. 1, pp. 2980–2984). Tsukuba, Japon: IEEE.

RANI, T. S., SOUJANYA, P. V. (2013). An ensemble method using small training sets for imbalanced data sets: Application to drugs used for kinases. In 2013 Sixth International Conference on Contemporary Computing (IC3) (pp. 516–521). IEEE.

RAO, H. B., ZHU, F., YANG, G. B., LI, Z. R., CHEN, Y. Z. (2011). Update of PROFEAT: A web server for computing structural and physicochemical features of proteins and peptides from amino acid sequence. Nucleic Acids Research, 39(SUPPL. 2), 32–37.

RICHARDSON, A. M., LIDBURY, B. A. (2013). Infection status outcome, machine learning method and virus type interact to affect the optimised prediction of hepatitis virus immunoassay results from routine pathology laboratory assays in unbalanced data. BMC Bioinformatics, 14(1), 206.

RODRIGUEZ, J. J., DIEZ-PASTOR, J. F., MAUDES, J., GARCÍA-OSORIO, C. (2012b). Disturbing Neighbors Ensembles of Trees for Imbalanced Data. 2012 11th International Conference on Machine Learning and Applications, 83–88.

Schapire, R. E. (1990). The Strength of Weak Learnability. Machine Learning, 5, 197–227.

SHEEN, S., AISHAWARYA, S. V, ANITHA, R., RAGHAVAN, S. V, BHASKAR, S. M. (2012). Ensemble Pruning Using Harmony Search. In S.-B. Corchado Rodriguez, E.S., Snasel, V., Abraham, A., Wozniak, M., Grana, M., Cho (Ed.), 7th International Conference, HAIS 2012, Salamanca, Spain, March 28-30th, 2012. Proceedings, Part II (Vol. 7209, pp. 13–24). Springer Berlin Heidelberg.

SILVA, O. N., MULDER, K. C. L., BARBOSA, A. E. A D., OTERO-GONZÁLEZ, A. J., Lopez-Abarrategui, C., Rezende, T. M. B., ... Franco, O. L. (2011). Exploring the pharmacological potential of promiscuous host-defense peptides: from natural screenings to biotechnological applications. Frontiers in Microbiology, 2(232), 1–14.

SKALAK, D. B. (1996). The Sources of Increased Accuracy for Two Proposed Boosting Algorithms. In In Proc. American Association for Arti Intelligence, AAAI-96, Integrating Multiple Learned Models Workshop (pp. 120–125).

STEFANOWSKI, J., WILK, S. (2008). Selective pre-processing of imbalanced data for improving classification performance. In I. Song, J. Eder, T. M. Nguyen (Eds.), Data Warehousing and Knowledge Discovery (pp. 283–292). Turin, Italia: Septiembre 2008.

SUN, Y., KAMEL, M. S., WONG, A. K. C., WANG, Y. (2007). Cost-sensitive boosting

for classification of imbalanced data. Pattern Recognition, 40(12), 3358–3378

SUN, Z., SONG, Q., ZHU, X. (2012). Using Coding-Based Ensemble Learning to Improve Software Defect Prediction. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 42(6), 1806–1817. doi:10.1109/TSMCC.2012.2226152

VAN DEN BERG, B. A, REINDERS, M. J., ROUBOS, J. A, RIDDER, D. DE. (2014). SPiCE: a web-based tool for sequence-based protein classification and exploration. BMC Bioinformatics, 15(1), 93.

VAN DEN BERG, B. A., REINDERS, M. J. T., HULSMAN, M., WU, L., PEL, H. J., ROUBOS , J. A., and de RIDDER, D. (2012). Exploring Sequence Characteristics Related to High-Level Production of Secreted Proteins in Aspergillus niger. PLoS ONE, 7(10), 1–11.

GRANGER W, E., SABORIN, R., GORODNICHY, D. (n.d.). Adaptive Selection of Ensembles for Imbalanced Class Distributions. In Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012) (Vol. 1, pp. 2980–2984). Tsukuba, Japon: IEEE.

WANG, G., LI, X., WANG, Z. (2009). APD2: the updated antimicrobial peptide database and its application in peptide design. Nucleic Acids Research, 37(Database issue), D933–7.

WANG, S., YAO, X. (2012). Multiclass Imbalance Problems: Analysis and Potential Solutions. IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics, 42(4), 1119–1130.

WANG, S., YAO, X. (2013). Relationships between Diversity of Classification Ensembles and Single-Class Performance Measures. IEEE Transactions on Knowledge and Data Engineering, 25(1), 206–219.

WANG, Z., WANG, G. (2004). APD: the Antimicrobial Peptide Database. Nucleic Acids Research, 32(Database issue), D590–D592.

WEBB, A. R. (2002). Statistical Pattern Recognition (Vol. 9). John Wiley and Sons, Ltd.

WEBB, A. R., Copsey, K. D. (2011). Statistical Pattern. Analysis.

XIAO, J., XIE, L., HE, C., JIANG, X. (2012). Dynamic classifier ensemble model for customer classification with imbalanced class distribution. Expert Systems with Applications, 39(3), 3668–3675.

YANG, P., YANG, Y. H., ZHOU, B. B., ZOMAYA, A. Y. (2010). A review of ensemble methods in bioinformatics, 5(4), 296–308.

YANG, P., YOO, P. D., FERNANDO, J., ZHOU, B. B., ZHANG, Z., ZOMAYA, A. Y. (2014). Sample Subset Optimization Techniques for Imbalanced and Ensemble Learning Problems in Bioinformatics Applications. IEEE Transactions on Cybernetics, 44, 445–455.

YIN, J., TIAN, L. (2014). Joint inference about sensitivity and specificity at the optimal cut-off point associated with Youden index. Computational Statistics and Data Analysis, 77, 1–13.

ZHANG, J., CAO, M.-Y., GAY, W., LI, B. (2013). Performance Comparison of ESVM and CSVM for Classifying the Lung Nodules on CT Scans. 2013 Seventh International Conference on Image and Graphics, 409–413.

ZHANG, T. (2004). Solving Large Scale Linear Prediction Problems Using Stochastic Gradient Descent Algorithms. In ICML 2004: Proceedings of the Twenty-First InternationalConference On Machine Learning. Omnipress (pp. 919–926).

ZHENXING QIN CHENQQI ZHANG, T. W., ZHANG, S. (n.d.). Advanced Data Mining and Applications. In J. Cao, Longbing, Feng, Yong, Zhong (Ed.), 6th International Conference, ADMA 2010, Chongqing, China, November 19-21, 2010, Proceedings, Part I (Vol. 6440, pp. 1–11). Berlin, Heidelberg.

ZHOU, Z., LIU, X. Y. (2006). Training cost-sensitive neural networks with methods addressing the class imbalance problem. IEEE Transactions on Knowledge and Data Engineering, 18(1), 63–77.

Referencias Bibliográficas

[1] H. B. Rao, F. Zhu, G. B. Yang, Z. R. Li, and Y. Z. Chen. Update of PROFEAT: A web server for computing structural and physicochemical features of proteins and peptides from amino acid sequence. *Nucleic Acids Research*, 39(SUPPL. 2):32–37, 2011.

[2] Osmar N Silva, Kelly C L Mulder, Aulus E a D Barbosa, Anselmo J Otero-Gonzalez, Carlos Lopez-Abarrategui, Taia M B Rezende, Simoni C Dias, and Octávio L Franco. Exploring the pharmacological potential of promiscuous host-defense peptides: from natural screenings to biotechnological applications. *Frontiers in microbiology*, 2(2):1–14, noviembre, 2011 2011.

[3] WF Porto, ON Silva, and OL Franco. Prediction and Rational Design of Antimicrobial Peptides. In Eshel Faraggi, editor, *Protein Structure, InTech*, chapter 17, pages 377–396. InTech, Brazil, abril 2012.

[4] Y Jerold Gordon, Eric G Romanowski, and Alison M McDermott. A review of antimicrobial peptides and their therapeutic potential as anti-infective drugs. *Current eye research*, 30(7):505–515, julio 2005.

[5] A Nijnik and Rew Hancock. Host defence peptides: antimicrobial and immunomodulatory activity and potential applications for tackling antibiotic-resistant infections. *Emerging health threats journal*, 2:1–7, enero 2009.

[6] M. Galar, a. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera. A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(4):463–484, julio 2012.

[7] Juan J. Rodriguez, Jose F. Diez-Pastor, Jesus Maudes, and Cesar Garcia-Osorio. Disturbing Neighbors Ensembles of Trees for Imbalanced Data. *2012 11th International Conference on Machine Learning and Applications*, pages 83–88, diciembre, 2012 2012.

[8] Hsiao-Yun Huang, Yi-Jhen Lin, Youg-Siang Chen, and Hung-Yi Lu. Imbalanced data classific[ation using random subspace method and SMOTE. *The 6th International Conference on Soft Computing and Intelligent Systems, and The 13th International Symposium on Advanced Intelligence*, pages 817–820, noviembre 2012.

[9] Urvesh Bhowan, Mark Johnston, Mengjie Zhang, and Xin Yao. Evolving Diverse Ensembles Using Genetic Programming for Classification With Unbalanced Data. *IEEE Transactions on Evolutionary Computation*, 17(3):368–386, junio 2013.

[10] Abhijeet Godase and Vahida Attar. Classifier Ensemble for Imbalance Data Stream Classification. In *Proceedings of the CUBE International Information Technology Conference on - CUBE '12*, pages 284–289, New York, USA, septiembre 2012. ACM Press.

[11] V. García, R. a. Mollineda, and J. S. Sánchez. On the k-NN performance in a challenging scenario of imbalance and overlapping. *Pattern Analysis and Applications*, 11(3-4):269–280, septiembre 2007.

[12] T Sobha Rani and P. V. Soujanya. An ensemble method using small training sets for imbalanced data sets: Application to drugs used for kinases. In *2013 Sixth International Conference on Contemporary Computing (IC3)*, pages 516–521. IEEE, agosto 2013.

[13] Qian Li, Bing Yang, Yi Li, Naiyang Deng, and Ling Jing. Constructing support vector machine ensemble with segmentation for imbalanced datasets. *Neural Computing and Applications*, 22(S1):249–256, julio 2012.

[14] Rok Blagus and Lara Lusa. Class prediction for high-dimensional class-imbalanced data. *BMC bioinformatics*, 11(1):1–17, enero 2010.

[15] Bartosz Krawczyk and Gerald Schaefer. An improved ensemble approach for imbalanced classification problems. In *2013 IEEE 8th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, pages 423–426. IEEE, mayo 2013.

[16] Peng Cao, Dazhe Zhao, and Osmar Zaiane. Cost sensitive adaptive random subspace ensemble for computer-aided nodule detection. *Proceedings of the 26th IEEE International Symposium on Computer-Based Medical Systems*, pages 173–178, junio 2013.

[17] Zhongbin Sun, Qinbao Song, and Xiaoyan Zhu. Using Coding-Based Ensemble Learning to Improve Software Defect Prediction. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):1806–1817, noviembre 2012.

[18] Jin Xiao, Ling Xie, Changzheng He, and Xiaoyi Jiang. Dynamic classifier ensemble model for customer classification with imbalanced class distribution. *Expert Systems with Applications*, 39(3):3668–3675, febrero 2012.

[19] Jui-jung Liao, Ching-hui Shih, Tai-feng Chen, and Ming-fu Hsu. An ensemble-based model for two-class imbalanced financial problem. *Economic Modelling*, 37:175–183, febrero 2014.

[20] Bartosz Krawczyk, Gerald Schaefer, and Michal Wozniak. An evaluation of classifier ensembles for class imbalance problems. In *2013 International Conference on Informatics, Electronics and Vision (ICIEV)*, pages 1–4. IEEE, mayo 2013.

[21] Jerzy Stefanowski and Szymon Wilk. Selective pre-processing of imbalanced data for improving classification performance. In Il-yeo Song, Johann Eder, and Tho Manh Nguyen, editors, *Data Warehousing and Knowledge Discovery*, pages 283–292, Turin, Italia, 2008. Septiembre 2008.

[22] K Napierała, Jerzy Stefanowski, and Szymon Wilk. Learning from imbalanced data in presence of noisy and borderline examples. *Rough Sets and Current Trends in Computing*, 6086:158–167, Junio, 2010.

[23] R Barandela, J.S Sánchez, V García, and E Rangel. Strategies for learning in class imbalance problems. *Pattern Recognition*, 36(3):849–851, marzo Marzo, 2003.

[24] E.a. Garcia. Learning from Imbalanced Data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, September Septiembre, 2009.

[25] P. Domingos. MetaCost: A General Method for Making Classifiers Cost-Sensitive. In *In Proceedings of the 5th ACM S16KDD International Conference on Knowledge Discovery and Data Mining*, pages 155–164, San Diego, CA., diciembre 1999.

[26] Charles Elkan. The Foundations of Cost-Sensitive Learning. In *In Proceedings of the 17th International Joint Conference on Artificial Inteligence.*, pages 973–978, Seatle, Washington, 2001.

[27] Xu-ying Liu and Zhi-hua Zhou. The Influence of Class Imbalance on Cost-Sensitive Learning: An Empirical Study. *Sixth International Conference on Data Mining (ICDM'06)*, pages 970–974, diciembre 2006.

[28] Marcus A. Maloof. Learning when data sets are imbalanced and when costs are unequal and unknown. In *ICML-2003 Workshop on Learning from Imbalanced Data Sets II*, 2003.

[29] Pengyi Yang, Yee Hwa Yang, Bing B Zhou, and Albert Y Zomaya. A review of ensemble methods in bioinformatics. 5(4):296–308, diciembre, 2010.

[30] Nicolás García-Pedrajas and César García-Osorio. Constructing ensembles of classifiers using supervised projection methods based on misclassified instances. *Expert Systems with Applications*, 38(1):343–359, enero 2011.

[31] Shina Sheen, S V Aishwarya, R Anitha, S V Raghavan, and S M Bhaskar. Ensemble Pruning Using Harmony Search. In S.-B Corchado Rodriguez, E.S., Snasel, V., Abraham, A., Wozniak, M., Grana, M., Cho,, editor, *7th International Conference, HAIS 2012, Salamanca, Spain, Mar 28-30th, 2012. Proceedings, Part II*, volume 7209, pages 13–24. Springer, marzo, 2012.

[32] Shuo Wang and Xin Yao. Using Class Imbalance Learning for Software Defect Prediction. *IEEE Transactions on Reliability*, 62(2):434–443, junio 2013.

[33] Radtke Paulo V W, Eric Granger, Robert Sabourin, and Dmitry Gorodnichy. Adaptive Selection of Ensembles for Imbalanced Class Distributions. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, volume 1, pages 2980–2984, Tsukuba, Japon, noviembre, 2012. IEEE.

[34] Shuo Wang and Xin Yao. Relationships between Diversity of Classification Ensembles and Single-Class Performance Measures. *IEEE Transactions on Knowledge and Data Engineering*, 25(1):206–219, enero 2013.

[35] Shuo Wang and Xin Yao. Multiclass Imbalance Problems: Analysis and Potential Solutions. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics*, 42(4):1119–1130, marzo 2012.

[36] Thomas G. Dietterich. Ensemble Methods in Machine Learning. In *First International Workshop, MCS 2000 Cagliari, Italy, June 21–23, 2000 Proceedings*, volume 1857, pages 1–15, Cagliary, Italia, junio, 2000.

[37] Wei-Jiun Lin and James J Chen. Class-imbalanced classifiers for high-dimensional data. *Briefings in bioinformatics*, 14(1):13–26, enero 2013.

[38] Peng Cao, Jinzhu Yang, Wei Li, Dazhe Zhao, and Osmar Zaiane. Ensemble-based hybrid probabilistic sampling for imbalanced data learning in lung nodule CAD. *Computerized medical imaging and graphics : the official journal of the Computerized Medical Imaging Society*, diciembre 2013.

[39] Victoria López, Sara del Río, José Manuel Benítez, and Francisco Herrera. Cost-sensitive linguistic fuzzy rule based classification systems under the MapReduce framework for imbalanced big data. *Fuzzy Sets and Systems*, 1:1–34, febrero 2014.

[40] Aida de Haro-Garcia and Nicolas Garcia-Pedrajas. A scalable method for instance selection for class-imbalance datasets. In *2011 11th International Conference on Intelligent Systems Design and Applications*, volume 1, pages 1383–1390. IEEE, noviembre 2011.

[41] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 16:321–357, junio 2002.

[42] Rashmi Dubey, Jiayu Zhou, Yalin Wang, Paul M Thompson, and Jieping Ye. Analysis of sampling techniques for imbalanced data: An n=648 ADNI study. *NeuroImage*, 87C:220–241, octubre 2013.

[43] Tao Wang Zhenxing Qin, Chengqi Zhang and Shichao Zhang. Advanced data mining and applications. In Jiang Cao, Longbing, Feng, Yong, Zhong, editor, *6th International Conference, ADMA 2010, Chongqing, China, November 19-21, 2010, Proceedings, Part I*, volume 6440 of *Lecture Notes in Computer Science*, pages 1–11, Berlin, Heidelberg, noviembre, 2010.

[44] Ana Cristina Braga and Pedro Oliveira. Diagnostic analysis based on ROC curves: theory and applications in medicine. *International Journal of Health Care Quality Assurance*, 16(4):191–198, Abril, 2003.

[45] Andrew P. Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159, July 1997.

[46] CE Metz. Basic principles of ROC analysis. *Seminars in nuclear medicine*, 8(4):283–298, Octubre,1978.

[47] Helena Aidos, RPW Duin, and ALN Fred. The Area under the ROC Curve as a Criterion for Clustering Evaluation. In *Proc International Conf. on Pattern Recognition Applications and Methods - ICPRAM*, pages 1–5, Barcelona, España, 2013.

[48] Robert E Schapire. The Strength of Weak Learnability. *Machine Learning*, 5:197–227, 1990.

[49] Pengyi Yang, Paul D Yoo, Juanita Fernando, Bing B Zhou, Zili Zhang, and Albert Y Zomaya. Sample Subset Optimization Techniques for Imbalanced and Ensemble Learning Problems in Bioinformatics Applications. *IEEE transactions on cybernetics*, 44:445–455, marzo 2014.

[50] Mohammed Khalilia, Sounak Chakraborty, and Mihail Popescu. Predicting disease risks from highly imbalanced data using random forest. *BMC medical informatics and decision making*, 11:1–13, enero 2011.

[51] M Kukar and Igor Kononenko. Cost-Sensitive Learning with Neural Networks. In Henri Prade, editor, *ECAI 98. 13th European Conference on Artificial Intelligence*, pages 445–449. JohnWiley & Sons, Ltd, agosto, 1998.

[52] Kate McCarthy, Bibi Zabar, and Gary Weiss. Does cost-sensitive learning beat sampling for classifying rare classes? *Proceedings of the 1st international workshop on Utility-based data mining - UBDM '05*, pages 69–77, 2005.

[53] Yanmin Sun, Mohamed S. Kamel, Andrew K.C. Wong, and Yang Wang. Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40(12):3358–3378, diciembre 2007.

[54] Zhi-hua Zhou and XY Liu. Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering*, 18(1):63–77, Enero 2006.

[55] Kehan Gao, Taghi M. Khoshgoftaar, and Amri Napolitano. A Hybrid Approach to Coping with High Dimensionality and Class Imbalance for Software Defect Prediction. In *2012 11th International Conference on Machine Learning and Applications*, pages 281–288. IEEE, diciembre 2012.

[56] S B Kotsiantis, I D Zaharakis, and P E Pintelas. Machine learning: a review of classification and combining techniques. *Artificial Intelligence Review*, 26(3):159–190, noviembre 2007.

[57] FX Breu, S Guggenbichler, and JC Wollmann. Machine Learning ECML 2004. In D. (Eds.) Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, editor, *15th European Conference on Machine Learning Pisa, Italy, September 2004 Proceedings*, volume 3201, pages 51–62, septiembre, 2004.

[58] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, enero 1967.

[59] Andrew R. Webb. *Statistical Pattern Recognition*, volume 9. John Wiley and Sons, Ltd, July 2002.

[60] Tong Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *ICML 2004: Proceedings of the Twenty-First InternationalConference On Machine Learning. Omnipress*, pages 919–926, 2004.

[61] Guangshun Wang, Xia Li, and Zhe Wang. APD2: The updated antimicrobial peptide database and its application in peptide design. *Nucleic Acids Research*, 37(SUPPL. 1):933–937, 2009.

[62] Zhe Wang and Guangshun Wang. APD: the Antimicrobial Peptide Database. *Nucleic acids research*, 32(Database issue):D590–D592, 2004.

[63] Michael R. Barnes. *Bioinformatics for Geneticists Second Edition A bioinformatics primer for the analysis of genetic data*. John Wiley and Sons, Ltd, Harlow, Essex, UK, second edi edition, July.

[64] P.M.Selzer, R.J.Marhofer, and A.Rohwer. *Applied Bioinformatics An Introduction*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[65] Bastiaan a. van den Berg, Marcel J T Reinders, Marc Hulsman, Liang Wu, Herman J. Pel, Johannes a. Roubos, and Dick de Ridder. Exploring Sequence Characteristics Related to High-Level Production of Secreted Proteins in Aspergillus niger. *PLoS ONE*, 7(10):1–11, 2012.

[66] I Dubchak, I Muchnik, S R Holbrook, and S H Kim. Prediction of protein folding class using global description of amino acid sequence. *Proceedings of the National Academy of Sciences of the United States of America*, 92(19):8700–8704, 1995.

[67] Kuo-chen Chou. Prediction of Protein Subcellular Locations by Incorporating Quasi-Sequence-Order Effect. *Biochemical and Biophysical Research Communications*, 278(2):477–483, November 2000.

[68] Andrew R Webb and Keith D Copsey. *Statistical Pattern*. 2011.

[69] Isabelle Guyon, Steve Gunn, and Masoud Nikravesh... [et al.] (eds.). *Feature extraction : foundations and applications*. Studies in fuzziness and soft computing. Springer, Berlin, 2006.

[70] Bastiaan a van den Berg, Marcel Jt Reinders, Johannes a Roubos, and Dick De Ridder. SPiCE: a web-based tool for sequence-based protein classification and exploration. *BMC bioinformatics*, 15(1):93, 2014.

[71] D.-S. Cao, Q.-S. Xu, and Y.-Z. Liang. propy: a tool to generate various modes of Chou's PseAAC. *Bioinformatics*, 29(7):960–962, April 2013.

[72] Robi Polikar. Ensemble based systems in decision making. *Circuits and Systems Magazine, IEEE*, 6(3):21–45, 2006.