UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

SISTEMATIZACIÓN COMPLETA DEL DISEÑO DE MÁQUINAS DE SOPORTE VECTORIAL ÓPTIMAS EMPLEANDO ALGORITMOS EVOLUTIVOS

T E S I S

QUE PARA OBTENER EL GRADO DE:

DOCTOR EN CIENCIAS (COMPUTACIÓN)

P R E S E N T A:

IVÁN MEJÍA GUEVARA

DIRECTOR DE LA TESIS: DR. ÁNGEL KURI MORALES

MÉXICO, D. F. **Febrero de 2008.**

Este trabajo está dedicado a mi familia:

Mi esposa Julia,
mis padres Dulce María y Juan José,
mi padre José Guadalupe,
mis hermanos, sobrinos y suegros.

Agradecimientos

Deseo expresar mi gratitud a la Universidad Nacional Autónoma de México, donde han transcurrido varias etapas de mi formación académica y experiencia profesional.

Quiero agradecer al Consejo Nacional de Ciencia y Tecnología, que por segunda ocasión me ha respaldado financieramente en otro periodo importante de mi formación académica.

Un agradecimiento muy especial al Dr. Ángel Fernando Kuri Morales, asesor y amigo, quién me ha iniciado en este fascinante mundo de la investigación y me ha guiado durante todo este proceso. Por sus comentarios, enseñanzas y su constante apoyo.

A la Dra. Cristina Verde Rodarte y al Dr. Rafael Pérez y Pérez, porque han seguido y evaluado este proceso desde su inicio. Por sus valiosos comentarios y recomendaciones.

Al Dr. Felipe Lara Rosano y al Dr. Christian Lemaître León, por sus valiosos comentarios.

Al Dr. Boris Escalante Ramírez, por el apoyo financiero recibido del Posgrado para la divulgación de los resultados en Memorias de Congresos Internacionales.

Al Dr. Ricardo Paramont Hernández García, por su amistad y enseñanzas durante mi primer año en el Posgrado.

A Lulú, Cecilia y Diana, por todo su apoyo en el manejo de los asuntos administrativos.

Resumen

Una Máquina de Soporte Vectorial (MSV) es un mecanismo eficiente para el entrenamiento de funciones de aprendizaje lineal en espacios inducidos por funciones kernel, con una capacidad de generalización sustentada por la Teoría Estadística del Aprendizaje. En los últimos años, la MSV ha demostrado su efectividad en la solución de una variedad muy amplia de problemas prácticos. No obstante, el óptimo funcionamiento de este método de aprendizaje requiere de la selección adecuada de sus parámetros libres.

En esta Tesis se aborda el problema de la selección óptima de estos parámetros con el uso de Algoritmos Genéticos (AGs) no convencionales. En particular, se diseña una propuesta autoadaptable, que también permite la selección de variables independientes y que es nombrada como Máquina de Clasificación Genética (MCG).

También se efectúa un análisis de estructuras polinomiales usadas para el diseño de MSVs y el método de Aproximación Polinomial Minimax. A partir de este análisis, se construye un nuevo kernel polinomial y se deriva un criterio para definir un rango apropiado de valores para el parámetro de regularización C de una MSV. Este criterio puede ser usado en la solución de problemas de clasificación binaria.

Finalmente, se pone a prueba el desempeño de la MCG empleando una propuesta de validación estadística, sustentada en la generación automática y no sesgada de problemas multivariados de clasificación binaria y el uso de pruebas de hipótesis.

Abstract

A Support Vector Machine (SVM) is an efficient mechanism for training linear learning functions in kernel induced feature spaces, with a generalization capability supported by the Statistical Learning Theory. In last years, the SVM has proven to be effective in solving a great variety of practical problems. However, the optimal performance of this learning method depends on the fine tuning of its free parameters.

In this Thesis the parameter selection problem is tackled by using a non-conventional Genetic Algorithm (GA). Particularly, a self-adaptive approach is proposed. The same GA is used for feature selection. It is called Genetic Support Vector Classifier (GSVC).

Polynomial structures are analyzed and used in the design of SVMs and Minimax Polynomial Approximation. Derived from this analysis, a new kernel function is proposed, together with a new criterion to define an appropriate range for the regularization parameter C in SVMs. This criterion can be used in the solution of binary classification problems.

Finally, the performance of the GSVC is tested by using a proposed validation method, which is based on the automatic construction of binary classification problems and the use of statistical hypothesis tests.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

1.1. Motivación

En los últimos 20 años, el desarrollo de las áreas de reconocimiento de patrones y aprendizaje de máquina ha sido impresionante. Actualmente se cuenta con múltiples técnicas que han sido desarrolladas bajo enfoques distintos, pero que persiguen objetivos similares. Uno de estos enfoques es el Bayesiano, que ha tomado mucha fuerza últimamente por el aumento en su aplicabilidad práctica [4]. Otro enfoque es el de modelos gráficos que ha resurgido como el marco principal para la descripción y aplicación de modelos probabilísticos [57]. Asimismo, los métodos de kernel han tenido un gran impacto desde el punto de vista teórico y práctico [119] [121].

Un método de kernel que surgió a inicios de la última década del siglo XX es la Máquina de Soporte Vectorial (MSV), desarrollada por Vapnik *et al* [6] y que ha resultado muy exitosa en la solución de muchas aplicaciones prácticas. La MSV es una técnica no paramétrica que puede operar como un modelo lineal; es decir, con la cual es posible encontrar funciones de decisión o regresión lineal, pero cuya aplicación se limita a la solución de problemas linealmente separables. Su rango de aplicación se amplia considerablemente cuando se considera como un método de kernel, con lo cual es posible abordar problemas de regresión no lineal y de clasificación de patrones no separables. La idea central en este método consiste en trasladar el problema original a un espacio superior, donde la función de respuesta construida con esta máquina de aprendizaje es lineal. Este efecto se logra al utilizar un tipo de función llamada kernel, que usualmente se formula como un producto interno en un espacio de características. Este procedimiento es conocido como *sustitución de kernel* [4] y también puede aplicarse a otro tipo de modelos lineales, algunos de ellos clásicos en la literatura estadística, como el Análisis de Componentes Principales (PCA) o el Discriminante Lineal de Fisher (FDA), con lo cual se obtienen versiones no lineales de ellos (Kernel PCA [118] y Kernel FDA [87] [106] [3]).

En algunos de los enfoques actuales para reconocimiento de patrones, generalmente hay que recurrir a ciertas suposiciones teóricas que, en ocasiones, limitan su aplicación práctica. En modelos Bayesianos, por ejemplo, hay que lidiar con supuestos de distribución a priori. En otros métodos estadísticos, se tienen supuestos de distribución o independencia que también limitan su rango de aplicación. En el caso de la MSV, no se requieren supuestos de esa naturaleza, pero en

su lugar existen algunos parámetros que tienen que ser calibrados para que su funcionamiento resulte óptimo. La forma de ajustar estos parámetros ha sido objeto de estudio en los últimos años y lo es en este trabajo de investigación.

La MSV también pude verse como una red neuronal con una sola capa oculta de unidades no lineales, definidas por funciones kernel [43]. Esta investigación se enfoca en el estudio de MSVs, particularmente en la determinación de sus parámetros libres. Para entender la evolución de esta técnica es fundamental tomar en cuenta el desarrollo histórico de las redes neuronales artificiales y la forma en que surge la MSV como una alternativa eficiente y con un sustento teórico basado en los principios de la Teoría Estadística del Aprendizaje (TEA). Por este motivo, se muestra enseguida esta evolución y la forma en que se ha tratado de superar el problema de la determinación de parámetros en MSVs.

1.2. Estado del Arte

El estudio moderno de las redes neuronales comienza [43] con el artículo de MacCulloch y Pitts [80], donde se unifica el estudio de la neuropsicología y la lógica matemática. El modelo formal se fundamenta en el carácter de *todo o nada* inherente en la actividad nerviosa. MacCulloch y Pitts demuestran que podría computarse cualquier tipo de función si se emplea un número suficiente de unidades simples - llamadas neuronas- y de conexiones sinápticas colocadas de manera apropiada. Además, se reconoce que este fue el primer trabajo que puede considerarse como de inteligencia artificial [113]. En 1949 se publica el libro *The Organization of Behavior* de Hebb [44], donde se establece por primera vez una regla de aprendizaje psicológico para la modificación sináptica; es decir, Hebb postula que las conexiones neuronales cambian continuamente conforme un organismo aprende nuevas tareas y que las estructuras neuronales se forman gracias a esos cambios. En consecuencia, se dice que los modelos que siguen esta teoría exhiben un *aprendizaje Hebbiano*.

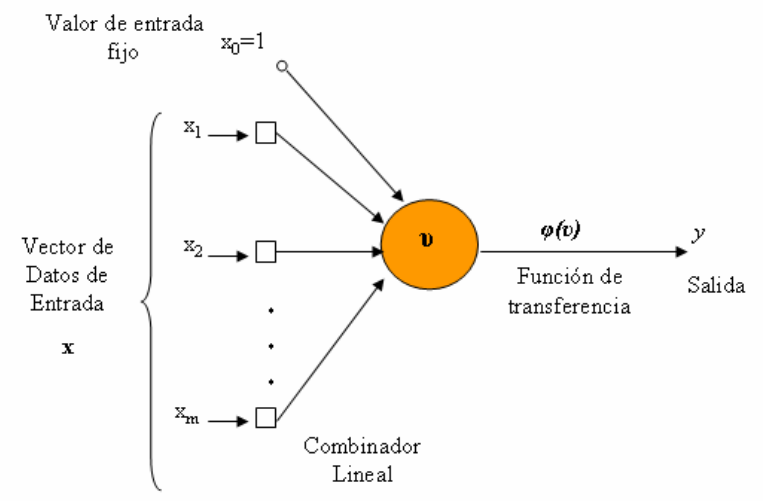

Figura 1.1: Arquitectura del Perceptrón de Rosenblatt

Quince años después del artículo publicado por McCulloc y Pitts, Rosenblatt [105] introduce un nuevo paradigma en el reconocimiento de patrones que llamó Perceptrón, un nuevo método de aprendizaje supervisado (Figura 1.1). El resultado más importante en el trabajo de Rosenblatt fue el teorema de convergencia del Perceptrón, en el que se afirma que para cualquier conjunto de datos linealmente separable, la regla de aprendizaje de este algoritmo garantiza la solución en un número finito de iteraciones. El entusiasmo generado en la década de los 60's con estos resultados y con la aparición de otro tipo de redes, conocida como *Adaline* [137](Figura 1.2(a)) (y *Madaline* para múltiples elementos adaptativos [136] (Figura 1.2(b))), hizo pensar a muchos que las redes neuronales eran capaces de resolver cualquier problema de reconocimiento de patrones. Sin embargo, en 1969 se publica *Perceptrons*, un libro en el cual Minsky y Papert [90] demuestran matemáticamente las limitaciones del Perceptrón y afirman que no hay razones para pensar que una extensión de éste pudiera mejorar su desempeño. Entre otras cosas, este trabajo le restó interés a la comunidad y frenó el financiamiento para seguir trabajando con redes neuronales durante toda la década de los 70's. No obstante, se lograron avances importantes en el desarrollo de métodos auto-organizados [133].

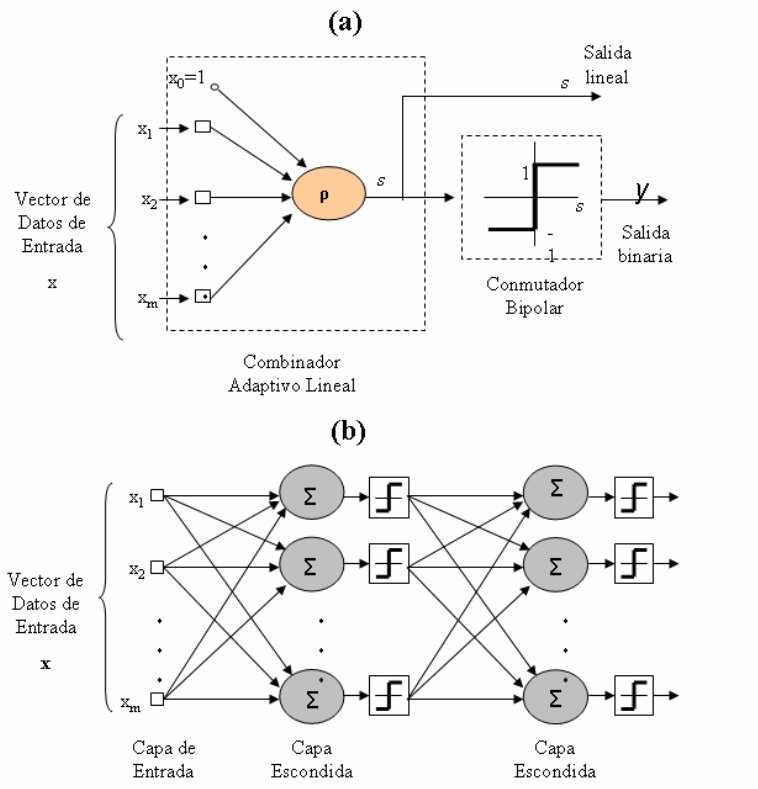

Figura 1.2: (a) Arquitectura de la red Adaline, (b) Arquitectura de la red Madaline

En 1980, se retoma el estudio de las redes neuronales. Grossberg [37] establece un nuevo principio de auto-organización conocido como Teoría de Resonancia Adaptiva (ART). Dos años después, Hopfield [46] emplea principios de física estadística para desarrollar una red recurrente con conexiones sinápticas simétricas, a este tipo de

redes se les denominaría más adelante como redes de Hopfield en su honor. La arquitectura de este tipo de red se ilustra en la Figura 1.3(a).

En 1983, Kirkpatrick, Gelatt y Vecchi describen un nuevo procedimiento llamado *recocido simulado*[1] que emplean para la solución de problemas combinatorios [59]. Esta idea considera principios de mecánica estadística y está basada en una técnica simple empleada por Metropolis *et al* en 1953 [85]. La idea de *recocido simulado* se emplea más adelante por Ackley, Hinton y Sejnowski en 1985 para el diseño de la Máquina de Boltzmann [1] (véase Figura 1.3(b)), que fue la primera realización exitosa de una red neuronal multicapa, con lo cual se rompe definitivamente con las especulaciones hechas por Minsky y Papert en 1969.

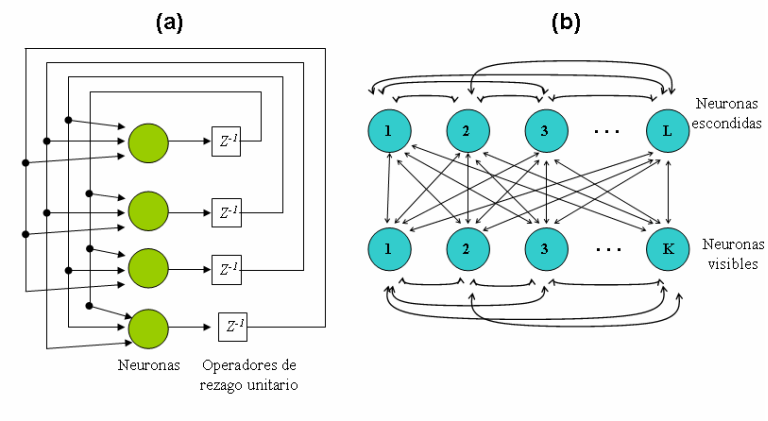

Figura 1.3: (a) Arquitectura de la red de Hopfield, (b) Arquitectura de la red Boltzmann

En 1986, Rumelhart, Hinton y Williams [108] [109] desarrollan el algoritmo de retropropagación. Asimismo, se publica en ese año un libro de dos tomos titulado *Parallel Distributed Processing: Explorations in the Microestructures of Cognition*, editado por Rumelhart y McClelland [110], donde se da gran importancia al uso del algoritmo de retropropagación, que a la postre resultaría como el algoritmo más popular en el entrenamiento de redes de perceptrones multicapa (RPMs). Es importante señalar que el aprendizaje por retropropagación se descubre de manera simultánea por otros dos autores por esas mismas fechas [100][75]. No obstante, había evidencia previa de que este algoritmo ya había sido descrito en 1974 por Werbos en su tesis doctoral en la Universidad de Harvard [134]; pero la idea básica de propagación realmente aparece en 1969 en el libro *Applied Optimal Control* de Bryson y Ho. [8]. Finalmente, se da los créditos de este algoritmo a Rumelhart *et al*, ya que fueron ellos quienes proponen su uso para el entrenamiento de máquina y para demostrar como se trabaja en ello. Además, se considera que este trabajo detonó el renacimiento de la investigación en redes neuronales artificiales. La arquitectura se muestra en la Figura 1.4

Para el año de 1988 Broomhead y Lowe [7] describieron un procedimiento para el diseño de redes neuronales con alimentación hacia adelante usando funciones de base radial (FBR), que es una idea alternativa al uso de redes de perceptrones multicapa (Figura 1.5).

[1] *Simulated annealing.*

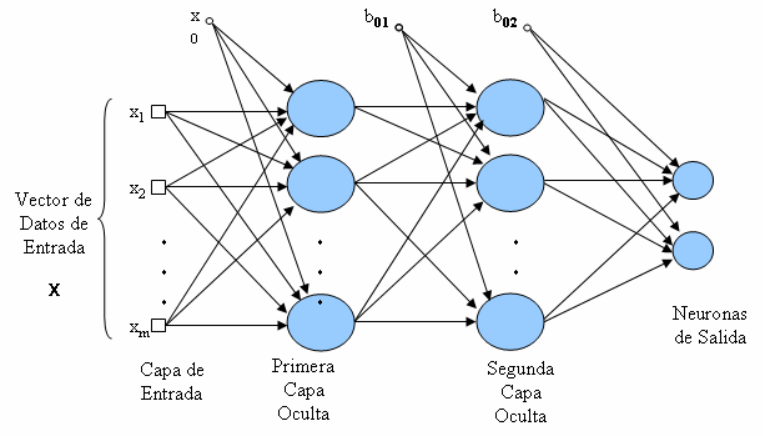

Figura 1.4: Arquitectura de una red de perceptrones multicapa

A inicios de la década de 1990, Vapnik *et al* [6] desarrollan una metodología que denominan Máquina de Soporte Vectorial. Se trata de una técnica de aprendizaje supervisado que hereda características de las redes neuronales, en pleno auge en ese entonces, pero cuyo sustento teórico se basa en el Principio de Minimización de Riesgo Estructural (PMRE) y no en el Principio de Minimización de Riesgo Empírico (PMREm), como hacen las RPMs, es decir, mientras que las RPMs minimizan el error de entrenamiento, las MSVs minimizan el error de prueba y, por lo tanto, su capacidad de generalizar es más alta (véase Figura 1.6). Estos principios de riesgo se derivan de la TEA, que comienza a desarrollarse durante la década de los 60's por Vapnik y Chervonenkis [130] y se consolida en los siguientes 30 años [131] [128]. Un elemento fundamental de la MSV es cómo el concepto de la dimensión de Vapnik-Chervonenkis (VC) se integra en el diseño de esta red. La dimensión VC provee una medida de la capacidad de aprendizaje de una máquina (MSV, por ejemplo) para aprender sobre un conjunto de datos [129] [128].

Para 1995, se diseña una nueva MSV para la solución de problemas linealmente no separables [18], siguiendo los mismos principios del diseño anterior, pero agregando un mecanismo de regularización para controlar los factores que afectan la habilidad de generalización de esta máquina de aprendizaje. En 1996, se emplea el mismo criterio de solución en las MSVs para la solución de problemas de regresión no lineal [23], donde se usa una función de pérdida como medida de error en la aproximación, la cual forma parte de la función de riesgo empírico que se pretende minimizar [18], en concordancia con los principios establecidos en la TEA. En [122] se describen muchas funciones de pérdida que pueden ser empleadas en la técnica de regresión con MSVs y se brinda un tratamiento muy completo de regresión no lineal con MSVs. En 1998 se publica el libro *Statistical Learning Theory* [129] de Vapnik, que es una versión más completa a la publicada en 1995 [128], en donde se brinda un tratamiento matemático riguroso de todos los elementos expuestos en el mismo.

La MSV se ha aplicado con éxito en la solución de problemas de muchos tipos. Algunas aplicaciones importantes en reconocimiento de patrones han sido las siguientes: reconocimiento de escritura hecha a mano [18] [116] [10], de objetos [5], identificación de voz [115], detección de fallas [54] y caracterización de textos [55].

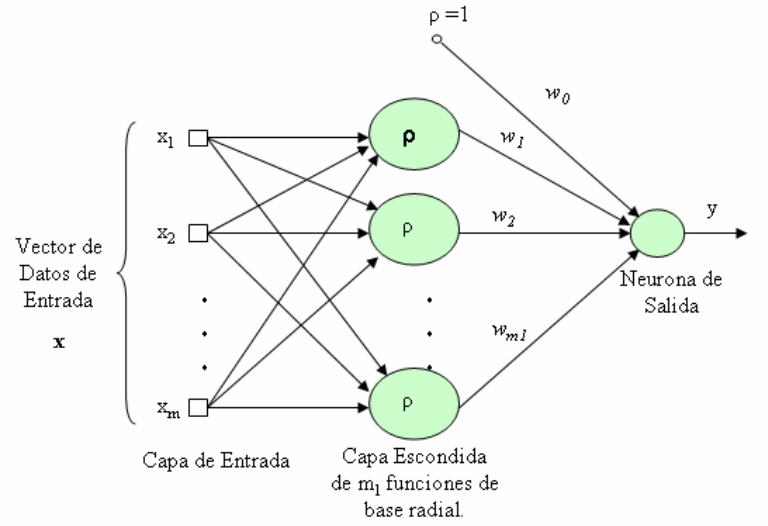

Figura 1.5: Arquitectura de una red de Base Radial

En el caso de regresión, la MSV se ha comparado con otras técnicas de series de tiempo [95] [94] [98]. También se ha estudiado su aplicación en la estimación de densidades [135].

Hasta entonces, la MSV había funcionado muy bien en la solución de una gran cantidad de problemas de regresión y clasificación, pero su entrenamiento dependía del uso de estrategias de optimización que resultaban impracticas e incluso inoperantes en la solución de problemas con miles de observaciones. La razón de esta limitante es que, en el planteamiento de una MSV como un problema de optimización restringido, se tienen que satisfacer una cantidad de restricciones que crece conforme lo hace el número de datos de entrenamiento, de hecho, la relación es de casi uno a uno en el caso de problemas de clasificación y de dos a uno (2 restricciones por cada punto de entrenamiento) en problemas de regresión. Además de la inestabilidad numérica que se encuentra en algoritmos de solución a este tipo de problemas, se tiene el gran inconveniente del tiempo, pues aunque algunos paquetes son capaces de resolver problemas relativamente grandes, el tiempo que les toma hacerlo resulta sumamente costoso en un sentido práctico.

La solución a este problema se da por primera vez con el empleo de un método que desarrolla Vapnik en 1982 conocido como Algoritmo de *Chunking* [127] y consiste básicamente en dividir el problema original en subproblemas más pequeños, donde se busca optimizar a cada uno de ellos, aprovechando el hecho de que algunos de los vectores de soporte son cero. Sin embargo, este algoritmo sigue resultando inadecuado con problemas a gran escala y, por lo tanto, se tienen que implementar técnicas de estructuras de datos muy sofisticadas, con lo cual se evita almacenar en memoria la matriz del problema de optimización cuadrática [127]. Osuna [98] [97] propuso una estrategia alternativa que consiste en dividir el problema original en subconjuntos de tamaño fijo, que se resuelven separadamente empleando alguna técnica de optimización cuadrática. El problema con este método es que, en conjuntos de datos muy numerosos, se siguen presentando problemas de estabilidad al resolver cada subproblema. Empleando el mismo principio de Osuna al

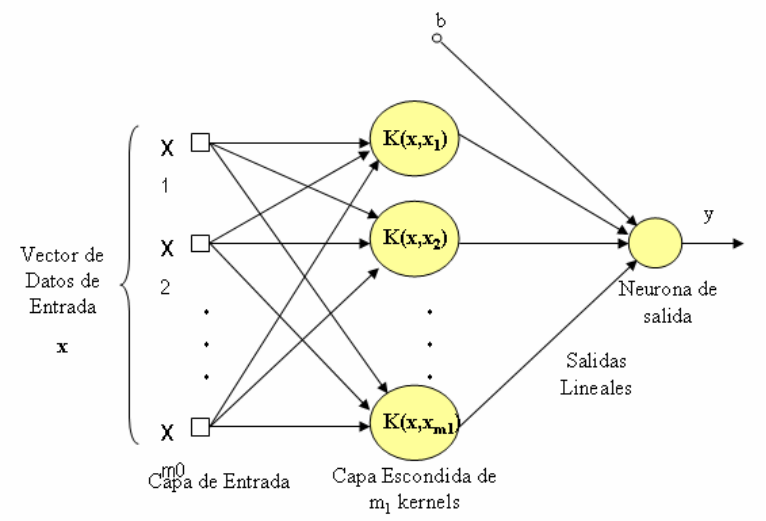

Figura 1.6: Arquitectura de una Máquina de Soporte Vectorial

subdividir el problema original, Platt [102] desarrolla una metodología que parte el problema original en los subconjuntos más pequeños posibles (de tamaño dos), con lo cual se evita el problema de almacenamiento de matrices en memoria y el uso de paquetería para la solución de problemas de optimización cuadrática. Esta técnica se conoce como Optimización Secuencial Mínima (OSM) y con ella no sólo es posible resolver problemas a gran escala, sino que el tiempo de solución se reduce de manera muy significativa. Otros investigadores han logrado reducir el tiempo de solución en problemas muy numerosos empleando otras variantes de MSVs (MSV Proximal (MSVP) [31], MSV Incremental (MSVI) [32]).

La solución de problemas de clasificación múltiple también ha sido posible con el empleo de MSVs. La MSV original se diseño para resolver problemas de clasificación binaria únicamente, pero algunos investigadores han empleado otro tipo de estrategias, como son las técnicas de *uno-contra-uno* y *uno-contra-todos* [2], en las que se construyen clasificadores binarios a partir de las clases del problema original y se califica su efectividad como clasificador múltiple, empleando ciertos criterios de asignación (por ejemplo el esquema de voto mayoritario). Sin embargo, con estos métodos se corre el riesgo de que, al efectuar decisiones basadas en clasificadores individuales, se den resultados inconsistentes en los cuales un mismo punto sea asignado a varias clases de manera simultánea [4]. Una variante del método *uno-contra-todos* se propone en [77], misma que es modificada en [139] desde un enfoque probabilístico y donde también se presenta un procedimiento de inferencia Bayesiano acorde con ella. Asimismo, otra variante de este método, junto con un refinamiento de la MSVP, se muestra en [33]. Una generalización del esquema de voto mayoritario para la estrategia de *uno-contra-uno*, basados en códigos de salida con error corregido, fue desarrollada por [22] y aplicada con MSVs en [2]. Finalmente, también se han desarrollado estrategias con MSVs para conjuntos de datos con una clase, que atacan problemas no supervisados relacionados con la estimación de densidades de probabilidad [117] [126].

Si bien la MSV ha resultado muy competitiva en la solución de diversos pro-

blemas, tiene el inconveniente de que se requiere la determinación de ciertos parámetros para su eficiente aplicación. Uno de esos parámetros es conocido como *parámetro de regularización* y generalmente se identifica como C, con el cual es posible mediar entre el poder expresivo del algoritmo de aprendizaje y el error de entrenamiento obtenido durante el proceso de optimización. También es necesario determinar otro tipo de parámetros en una MSV, los cuales están relacionados con el tipo de función que se emplea como kernel durante el proceso de entrenamiento y prueba de la máquina. El número de éstos dependerá del tipo de kernel elegido.

La determinación apropiada de los parámetros es fundamental para un funcionamiento óptimo de una MSV. Particularmente, una mala elección de C ocasiona que la máquina no sea capaz de resolver el problema en cuestión de manera óptima. En ese sentido, se corren dos riesgos importantes: 1) El diseño de una MSV que sea incapaz de generalizar satisfactoriamente sobre conjuntos de datos desconocidos (datos de prueba) o 2) El mal entrenamiento de ésta, con lo cual ni siquiera resulte efectiva para aproximar sobre el conjunto de datos de entrenamiento. En síntesis, el diseño de una red mal comportada ocasiona que no se aprovechen las ventajas teóricas que esta metodología ofrece sobre otras técnicas que le compiten (RPMs, por ejemplo).

Este problema ha sido abordado de diversas maneras, ya que tradicionalmente se han calibrado estos parámetros de manera arbitraria o atendiendo al buen juicio del investigador [43] [128]. Algunas aproximaciones hacen uso de criterios Bayesianos para estimar dichos parámetros, por ejemplo, en [35] se usa una interpretación probabilística de las MSVs para obtener expresiones que contribuyan a la determinación de los parámetros. Los resultados son superiores a la metodología tradicional de MSVs aplicada en varios conjuntos de datos, pero el inconveniente de este tipo de aproximaciones es que tales expresiones no tienen solución analítica y, por lo tanto, se deben aplicar técnicas de simulación para su aproximación. Además, se hacen suposiciones arbitrarias con relación al tipo de distribución a priori (cuando se desconoce la distribución de donde proviene la información, que es lo más común en la práctica).

Una propuesta de estimación analítica para el parámetro de regularización se da en [14], empleando kernels polinomiales en el entrenamiento de la MSV, así como una serie de premisas que impiden inferir un resultado generalizado a un número arbitrario de problemas. En 2001 se propuso en [13] un método de gradiente y un conjunto de cotas para el error de validación para estimar los parámetros del kernel, considerando al valor de C como uno de ellos; sin embargo, se requiere de la especificación de otros parámetros desarrollados durante la fase de optimización.

En muchos trabajos se sugiere el empleo de validación cruzada (CV) [49] [50] [76] [95] para la determinación de parámetros, particularmente el parámetro C. Una estrategia que es comúnmente sugerida para atacar este problema se conoce como *Método Grid* [50] [74] [88], en la cual se sugieren pares ordenados (C, parámetro de la función kernel)[2], los cuales forman una especie de celda. La selección de la mejor combinación de ellos se efectúa con base en los resultados de validación cruzada, cuando ésta corresponda con el menor error arrojado por el clasificador. El inconveniente de este método es que se asignan de manera arbitraria conjuntos discretos de potenciales valores para ese par ordenado, limitando de esta manera el espacio de búsqueda y la precisión en la elección de los valores óptimos correspondientes.

[2]También puede aplicarse en la determinación de más de dos parámetros, en cuyo caso se tendrían $n-adas$ ordenadas, formadas por cada uno de los n parámetros $(p_1, p_2, \ldots, p_n)$.

Además, esta técnica es computacionalmente costosa, ya que el modelo debe ser evaluado dentro de cada celda para cada uno de los parámetros y, por tanto, el tiempo de búsqueda crece exponencialmente con el número de ellos. La búsqueda resulta aún más costosa si se emplea validación cruzada de *k-dobleces* (k-FCV) o validación cruzada *dejando-fuera-una-observación* (LoO-CV) como criterio de selección de los mejores parámetros, lo que generalmente ocurre en la práctica.

En [99] se emplea *recocido simulado* para la determinación óptima de parámetros, pero se ha probado en [71] que el uso de diferentes variantes de Algoritmos Genéticos (AGs) son superiores en la solución de problemas de optimización, en comparación con la aplicación de algoritmos de búsqueda local.

En [40] se construye un sendero para el parámetro de Regularización C por medio de una función de decisión lineal, cuyos parámetros son funciones lineales por trozos de C, así como los vectores de soporte derivados del planteamiento dual de la MSV. El sendero que se construye con esta metodología incluye a todos los valores posibles de C y es empleado para resolver el problema de optimización dual para problemas de clasificación, desarrollando un algoritmo alternativo que funciona eficientemente y con el que se obtienen resultados superiores a MSVs aplicadas a un conjunto de problemas representativo. Finalmente, se emplea validación cruzada para construir el error de generalización de la máquina en función del parámetro C, con lo cual se decide el modelo apropiado. El problema con esta metodología es que la complejidad computacional es considerable, involucrando la solución de sistemas de ecuaciones en el proceso de construcción del sendero, donde en ocasiones intervienen matrices que no tienen rango completo y, por lo tanto, ocasionan dificultades numéricas.

También se han propuesto métodos evolutivos para la solución al problema de selección de parámetros. Aunque algunos de ellos resultan potencialmente interesantes, en general, no se establecen criterios estadísticos contundentes en sus conclusiones.

Un método usado en varias ocasiones para la solución de este tipo de problemas es el de Estrategias Evolutivas (EEs). En [111] una estrategia evolutiva paralela y asíncrona es usada para calibrar el parámetro C y el parámetro de un kernel de base radial (KBR)[3]. Los autores proponen algunos criterios de selección para la medición del ajuste de la MSV. Uno de estos criterios es el número de vectores de soporte derivados tras el proceso de entrenamiento. Sin embargo, los mismos autores reportan que su estrategia de búsqueda en algunos casos queda atrapada en óptimos locales. También encuentran que el uso del número de vectores de soporte como función objetivo también ocasiona ciertas dificultades durante la búsqueda. De igual manera, es importante señalar que en los experimentos efectuados únicamente se emplea un tipo de función kernel (el KBR) y la consecuente estimación de su parámetro, sin reportar los resultados obtenidos con funciones kernel de otro tipo. Otra EE se presenta en [30] donde se emplea una estrategia denominada Estrategia Evolutiva con Matriz de Covarianzas Adaptable (CMA-ES) para la optimización del parámetro de un kernel Gaussiano y el parámetro de regularización. En este artículo se emplean tres parametrizaciones de kernels Gaussianos. Los autores resaltan las ventajas de su método sobre otros métodos de gradiente y, con base en sus experimentos, también encuentran que su propuesta es significativamente superior a los resultados obtenidos con el Método *Grid*.

[3]En el Capítulo 3 de esta Tesis se presentan otros tipos de funciones kernel y en el Capítulo 4 se exponen algunos elementos teóricos relacionados con las funciones kernel y su caracterización.

En [48] se ofrece una buena discusión de los dos últimos artículos mencionados en el párrafo anterior, donde los autores muestran que aunque el KBR es considerado en la actualidad como el más popular, éste no siempre ofrece los mejores resultados en la práctica. Esta afirmación se desprende al aplicar la técnica de Programación Genética (PG) para la selección de un kernel óptimo en MSVs (ellos simbolizan su propuesta como GK SVM). En la fase experimental se emplean las tres funciones kernel más populares[4], donde se encuentran claras diferencias en los ajustes debidas al tipo de kernel empleado en la fase experimental. Finalmente, los autores reportan la habilidad de GK SVM para encontrar nuevas funciones kernel que compiten favorablemente con los kernels usados tradicionalmente, sin la necesidad de insertar parámetros para lograr un ajuste óptimo.

En [52] también se emplea una EE, que incluye un algoritmo determinístico para la adaptación de las probabilidades en la aplicación de operadores variacionales. Los autores prueban que su propuesta funciona cuando es usada para la optimización de la arquitectura de una red neuronal. Un aspecto importante de este algoritmo es que en este procedimiento es necesario determinar una serie de parámetros libres, por lo que no sólo los parámetros de la MSV deben ser determinados, sino también éstos que se derivan de su propia estrategia. Con relación a este hecho, los autores afirman (sin dar detalles) que cuentan con estrategias confiables para la determinación de los mismos.

Finalmente, también han surgido algunos trabajos donde se reporta la aplicación de algoritmos genéticos en la determinación de parámetros de MSVs, así como en la reducción de características de los conjuntos de datos de entrenamiento. En [114], por ejemplo, se aplican AGs para la determinación del parámetro del kernel y no para en la búsqueda del valor de C. Otra aplicación con AGs se sugiere en [29] para la selección de características y la optimización de C en MSVs. Una contribución importante es el uso de cotas teóricas para el error de generalización de una MSV en lugar de emplear CV. Aún cuando los autores hacen referencia a la capacidad de su propuesta para optimizar el parámetro de la función kernel, en sus experimentos se limitan a utilizar funciones de kernel lineal, donde no hay parámetros a ser optimizados. Los autores también reportan que su aproximación arrojó mejores y más rápidos resultados cuando se emplea una función de ajuste basada en estas cotas. Sin embargo, al analizar detenidamente la expresión analítica de las cotas empleadas (una de ellas es conocida como la cota de Jaakkola-Haussler [53]), es posible apreciar que la misma función kernel empleada durante el proceso de entrenamiento se encuentra involucrada en su estimación. Entonces, si el parámetro del kernel (por ejemplo en KBR) también es optimizado durante el proceso genético, los valores de las cotas deben ser actualizados tomando en cuenta los cambios en la función kernel que resulten del proceso evolutivo. Por tanto, el efecto que pudiera traer en el tiempo y calidad del ajuste podría ser diferente a lo que los autores concluyen en su análisis. Otro aspecto importante a considerar en [29] es el uso de codificación decimal cuando al tratar de resolver el problema de selección de características, se conoce de antemano el número de éstas; de otra forma, se sugiere el empleo de codificación binaria. En [15] [51] [89] se reporta la aplicación de AGs en la selección de parámetros en MSVs, la reducción de características en los datos de entrenamiento y se prueba su efectividad en la solución de problemas prácticos. Sin embargo, en algunos casos la selección de los parámetros del algoritmo genético

[4]Polinomial, KBR y Sigmoidal. En el capítulo 4 se ofrece un tratamiento más detallado de este tema.

es completamente arbitraria o no se reporta la forma en que se determinan. Los criterios de validación se basan en la comparación con otras técnicas en la solución de conjuntos de problemas prácticos y, se afirma, *representativos*. Otro aspecto a considerar es la decisión arbitraria con relación al espacio de búsqueda de los parámetros que se pretende optimizar.

Es importante mencionar que la aplicación de AGs para la solución a problemas de optimización presenta varias ventajas sobre otros métodos (métodos de gradiente, por ejemplo), entre otras: a) un AG analiza de manera simultánea varias posibilidades de solución en el espacio de búsqueda, en cambio otros métodos (de gradiente, por ejemplo) van explorando mejoras de una única posible solución, b) con el uso de AGs no se impone ninguna restricción a la función objetivo, como por ejemplo la restricción de que sea continua y/o derivable, como se hace en otros métodos, c) Teóricamente es posible garantizar convergencia cuando se emplean AGs elitistas que pueden ser modelados como Cadenas de Markov [107]. También debe señalarse que existen distintas variantes de AGs que compiten muy bien en la solución de problemas de optimización, particularmente un mecanismo autoadaptable, como se muestra en [34]. El mecanismo autoadaptable debe su nombre al hecho de que los propios parámetros del AG se determinan en el proceso de optimización. Dada la importancia que este tipo de parámetros tiene en la solución de problemas de optimización con AGs, sorprende que en otras investigaciones, como [51] [15] [89], no se reporta la forma en que éstos se implementaron.

En conclusión, en los artículos discutidos anteriormente no presentan criterios de validación estadística contundentes con relación al desempeño de sus propuestas; en algunos se asumen una serie premisas que limitan su aplicación y/o se proponen estrategias computacionalmente costosas. En otros casos, la experimentación se enfoca en casos particulares (como en el uso de un tipo de función kernel). Por ello, en esta Tesis se aborda el problema de selección de parámetros en MSVs empleando un Algoritmo Genético autoadaptable. También se efectúan pruebas estadísticas exhaustivas para comprobar la efectividad de esta propuesta. Los objetivos y las acciones a emprender se describen con más detalle en la siguiente sección.

1.3. Objetivos

En esta Tesis se plantean los siguientes objetivos:

1. Utilizar una variante de AG autoadaptable para la determinación automática de parámetros en MSVs, con lo que se busca evitar el mayor sesgo posible en su determinación.

2. Estimar una cota para el rango de valores que puede tomar el parámetro C. Esta cota se puede obtener de manera analítica, apoyándose en los resultados de un análisis polinomial multivariado de la información.

3. A partir de este análisis, y comparando la forma funcional de las funciones de discriminación entre una MSV y un polinomio obtenido con el método de Análisis Polinomial Minimax (APM), se propone la definición de un nuevo tipo de kernel polinomial que, al usarse con una MSV, permite definir la función de discriminación en una forma explícita.

4. Se analizará la conveniencia en el uso de este nuevo kernel con MSVs en

problemas de clasificación de patrones y regresión no lineal y, potencialmente, con otros métodos de kernel.

5. Se propone una metodología para la generación automática de un número arbitrario de problemas multivariados de clasificación binaria.

6. Tomando como base los problemas de clasificación generados, se emplearán criterios de validación estadística para medir la efectividad de la metodología genética autoadaptable en la solución de problemas de clasificación binaria.

7. Finalmente, la misma propuesta de AG autoadaptable también se usa en la solución de problemas de regresión no lineal con MSVs, aunque se efectúan comparaciones con otros métodos sobre la base de un conjunto de problemas prácticos, no se efectúa una análisis estadístico exhaustivo, como en el caso del problema de clasificación binaria.

Este trabajo de investigación está organizado de la siguiente forma. En el capítulo 2 se describen algunos conceptos teóricos que ayudarán a entender mejor el planteamiento del problema y el procedimiento seguido para la consecución de los objetivos anteriores. Específicamente, se tratarán algunos elementos de la Teoría de Optimización, AGs, APM y funciones de Walsh y Rademaker. En el capítulo 3 se describe el planteamiento de MSVs para problemas de clasificación y regresión, se resalta la importancia de los parámetros de esta máquina de aprendizaje en su optimización, la descripción de algoritmos de solución para MSVs y algunos conceptos derivados de la teoría estadística del aprendizaje, directamente relacionados con este tema. En el capítulo 4 se muestran las aportaciones teóricas derivadas de esta investigación, concernientes al comparativo entre funciones de discriminación obtenidas con APM y MSVs, la propuesta de una forma de acotar el valor del parámetro C en MSV y la demostración algebraica de una nueva propuesta de kernel polinomial. El capítulo 5 está dedicado a la descripción de los diferentes métodos desarrollados en esta Tesis: a) para la optimización de parámetros en MSVs con AGs, b) para la generación automática de problemas multivariados de clasificación binaria, c) para la validación estadística en la solución de problemas de clasificación binaria y d) para la optimización de parámetros en problemas de regresión con MSVs. En el capítulo 6 se describen de manera detallada la forma en que se llevaron a cabo los experimentos desarrollados para la comprobación empírica de los algoritmos propuestos, los resultados derivados de diversas formas de validación y aquellos obtenidos de las comparaciones con otras metodologías de aprendizaje. Finalmente, se presentan las conclusiones y algunas sugerencias para investigaciones futuras.

Capítulo 2

Conceptos Generales

Se enlistan a continuación algunos temas relevantes que se encuentran estrechamente relacionados con los objetivos planteados en esta Tesis y se abordan en este capítulo:

2.1 Optimización. La descripción teórica de algunos conceptos de optimización permitirá entender de mejor manera la razón por la cual una MSV resuelve problemas de clasificación y regresión planteando problemas de optimización con restricciones, así como las ventajas teóricas relacionadas con este tipo de planteamientos. Estos elementos han resultado fundamentales en el surgimiento de algoritmos eficientes de solución para MSVs.

2.2 Algoritmos Genéticos. El empleo de AGs como metodología de optimización ha resultado eficaz en la práctica, tomando en cuenta las ventajas de su uso en comparación con otros métodos de optimización. Asimismo, el uso de alternativas no convencionales resulta muy ventajoso para superar algunas deficiencias que presentan los enfoques tradicionales. Resalta, en ese sentido, la variante de Vasconcelos, que es la base de esta investigación como mecanismo de optimización de parámetros en MSVs, así como la aplicación de una estrategia autoadaptable.

2.3 Aproximación Polinomial Minimax. El uso de APM en la determinación de una función de regresión o clasificación y la relación de este tipo de funciones en comparación con las obtenidas con MSVs. También se aprovecha esta relación para estimar una cota para el rango de valores para el parámetro C en problemas de clasificación binaria. Asimismo, con base en estos análisis fue posible descubrir un nuevo tipo de kernel polinomial, que se prueba de manera algebraica y experimental en capítulos posteriores.

2.4 Funciones Multivariadas de Walsh. Finalmente, se describe la forma en que se construyen funciones de Walsh y Rademacher, que serán la base para la determinación de un número arbitrario de problemas multivariados de clasificación binaria y que se usarán para la validación estadística de la propuesta de optimización genética de parámetros en MSVs.

2.1. Optimización

2.1.1. Optimización no restringida

Un concepto de trascendental importancia en optimización es el de convexidad. Generalmente, el objetivo en muchas aplicaciones prácticas consiste en la maximización o minimización de una función, donde generalmente se requiere conocer si esos óptimos son globales y únicos o sólo locales. En lo general interesan y se buscan los primeros, pero no es posible tener puntos de buen comportamiento sin exigir a las funciones objeto de estudio propiedades que aseguren dicha globalidad [25]. Por ello es que resulta importante mencionar algunos conceptos relevantes relacionados con convexidad de conjuntos y funciones. Es importante señalar que algunas de las definiciones que se muestran más adelante son aplicables a espacios más generales, sin embargo, para propósitos de esta investigación es suficiente considerar sólo espacios reales.

Se dice que un conjunto es convexo, si para cualquier par de puntos pertenecientes al conjunto, el segmento de recta que los une también se encuentra en el conjunto. En términos formales, $S \subseteq R^n$ es convexo si $\forall x_1,\ x_2 \in S$, entonces $\lambda x_1 + (1-\lambda)x_2 \in S$[1], donde $\lambda \in [0,1]$. En la Figura 2.1 se muestra un conjunto que cumple la condición de convexidad y otro que no la cumple.

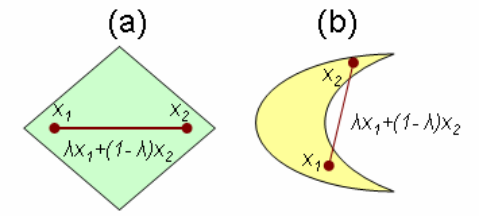

Figura 2.1: (a) Ejemplo de conjunto convexo, (b) Ejemplo de conjunto no convexo.

Por otro lado, una función es convexa si al evaluarse en una combinación convexa de cualesquiera dos puntos de su dominio, el valor resultante es menor o igual que la combinación convexa de la función evaluada en esos puntos. Si la desigualdad es estricta, la función se conoce como estrictamente convexa. En otros términos, una función escalar $f(z)$ se llama convexa para $z \in R^n$ si para todo $z,\ u \in R^n$, y para alguna $\lambda \in (0,1)$, $f(\lambda z + (1-\lambda)u) \leq \theta f(z) + (1-\theta) f(u)$. La Figura 2.2 ilustra muy bien este concepto, donde se muestra una función que es estrictamente convexa. Como consecuencia, resulta evidente que cualquier función lineal es convexa, pero no estrictamente. La definición de función cóncava se obtiene al cambiar el sentido de la desigualdad en la expresión anterior. A partir de estas definiciones, se deriva que es posible obtener una función cóncava a partir de una conexa, si esta última se multiplica por una constante negativa y, viceversa. También puede demostrarse que la suma de funciones convexas es convexa y la suma de funciones cóncavas resulta en una función cóncava [25].

En general, las condiciones de convexidad o concavidad resultan demasiado fuertes, por ello, en algunos casos es suficiente con exigirles a las funciones condiciones más débiles, pero que conserven algunas propiedades de las funciones

[1]Esta expresión se conoce como combinación convexa y define un segmento de recta entre ambos puntos.

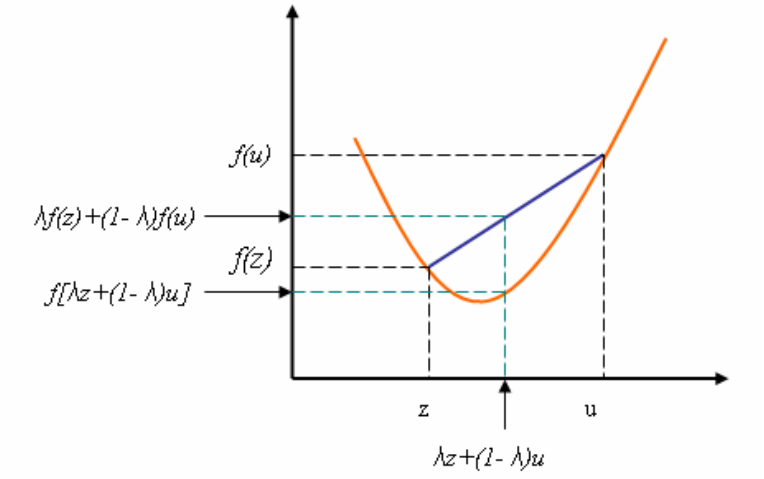

Figura 2.2: Función estrictamente convexa.

cóncavas y convexas en virtud de la búsqueda de máximos y mínimos. Estas condiciones se refieren a las funciones cuasicóncavas y cuasiconvexas, pero antes de definirlas es necesario introducir nuevos conceptos. Se dice que el contorno de una función en y $(C_f(y))$ está definido por todos los puntos, x, en el dominio de la función para los cuales $f(x) = y$. El contorno superior de la función en y $(CS_f(y))$ consiste de todos los vectores x para los cuales $f(x) \geq y$. Análogamente, el contorno inferior de una función en y $(CS_f(y))$ está constituido por todas los x tales que $f(x) \leq y$. La representación gráfica de estos conceptos se observa en la Figura 2.3.

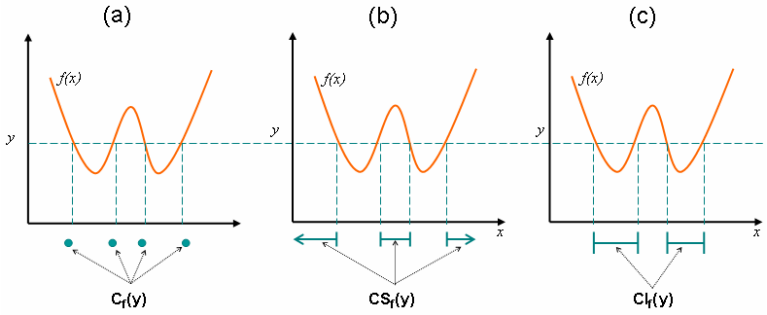

Figura 2.3: (a) Contorno de una función escalar en y, $C_f(y)$, (b) Contorno superior $CS_f(y)$ y, (c) Contorno inferior $CI_f(y)$.

Por tanto, una función f con dominio en un conjunto convexo $X \subset R^n$ y contradominio en los reales, es cuasicóncava (cuasiconvexa), si y sólo si, el contorno superior (inferior) de la función en y es un conjunto convexo para cualquier y del rango de f. Es evidente que la función de la Figura 2.3 no es cuasicóncava, ni cuasiconvexa, ya que $CS_f(y)$ y $CI_f(y)$, no son conjuntos convexos (están definidos por segmentos de recta o semirecta separados). En la Figura 2.4 se dan ejemplos de funciones cuasicóncavas y cuasiconvexas. En particular, la Figura 2.4(b) muestra un ejemplo de función cuasicóncava y cuasiconvexa al mismo tiempo, la razón se debe

a que esa función es monótona creciente y está definida sobre un conjunto convexo (la recta real)[25]. A partir de esta observación, resulta claro que las funciones lineales representan funciones que son al mismo tiempo cuasicóncavas y cuasiconvexas (inclusive la recta con pendiente 0). Formas alternativas de definir cuasiconvexidad y cuasiconcavidad se pueden encontrar en [132].

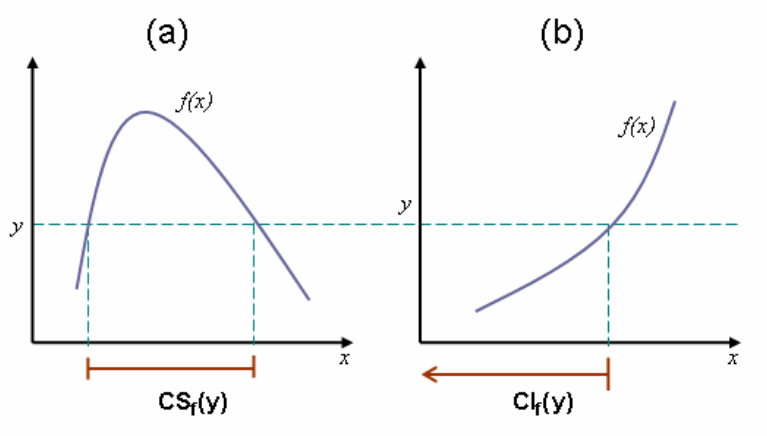

Figura 2.4: (a) Ejemplo de función cuasicóncava, (b) Ejemplo de función cuasiconvexa.

Las funciones cuasicóncavas (convexas) y cóncavas (convexas) están íntimamente relacionadas con la búsqueda de óptimos de las respectivas funciones. En particular, la concavidad se asocia a los máximos globales y la convexidad a los mínimos. En muchos casos se requiere saber no sólo si los máximos o mínimos son globales sino únicos. Para establecer condiciones necesarias y suficientes para la existencia de máximos y mínimos, no sólo para funciones reales de variable real, sino para funciones escalares (con dominio en R^n) es conveniente definirlo en términos matriciales.

Así, una matriz $\boldsymbol{A}$ de $n \times n$ simétrica es negativa-definida (n.d.) si $x^T Ax < 0$ para todo $x \in R^n$,$x \neq 0$. Se dice que $\boldsymbol{A}$ es negativa-semidefinida (n.s.d.) si en la definición anterior se cambia la restricción $<$ por $\leq$. Asimismo, A es positiva (semi)-definida (p.d.) si se invierten las desigualdades en las expresiones anteriores [25]. Ahora bien, si f es una función escalar definida sobre un conjunto convexo y abierto[2] y si f es una función de clase C^2 [3], entonces se define [72]:

$$\mathbf{H} = \begin{pmatrix} f_{11} & f_{12} & \cdots & f_{1n} \\ f_{21} & f_{22} & \cdots & f_{2n} \\ \cdots & & & \\ f_{n1} & f_{n2} & \cdots & f_{nn} \end{pmatrix}. \tag{2.1}$$

donde $f_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}$. $\mathbf{H}$ se llama matriz *Hessiana* asociada a la función f. De esta manera, una función C^2 con dominio en un conjunto convexo y abierto, es convexa (cóncava), si y sólo si, la matriz *Hessiana* de f es p.s.d. (n.s.d.) para todo x en el dominio de la función.

[2]En un conjunto abierto, la vecindad de cualquier punto se encuentra completamente contenida en el conjunto [39].

[3]Una función es de clase C^k si sus derivadas de orden k están bien definidas y son continuas.

El valor óptimo de una función es un mínimo global, si la función evaluada en ese punto alcanza su valor más pequeño en comparación con los valores obtenidos con cualquier otro punto en el dominio de la función. El valor es un mínimo local, si la función alcanza un mínimo en una vecindad del valor óptimo. En símbolos, $z*$ es un mínimo global si $\forall z \in \Omega$, $f(z*) < f(z)$; y $z*$ es un óptimo local si $\exists \epsilon > 0$, $f(z*) \leq f(z)$ para cualquier $z \in \Omega$ tal que $\|z* - z\| < \epsilon$. En el caso de máximos globales y locales, sólo se requiere cambiar el sentido de la desigualdad en las definiciones anteriores. En la Figura 2.5 se ilustran estos conceptos.

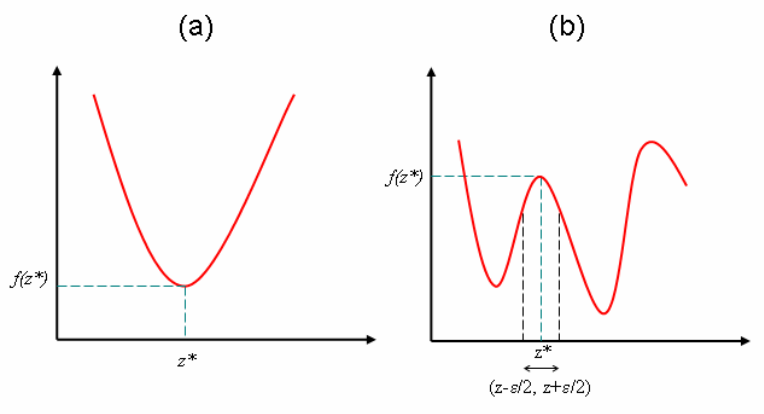

Figura 2.5: (a) Ejemplo de mínimo global de una función, (b) Ejemplo de máximo local de una función.

El resultado más importante en la optimización sin restricciones consiste en establecer condiciones necesarias y suficientes para máximos o mínimos locales, esto es:

1. Condiciones Necesarias
   - Si f es C^1 y x^* es un máximo (mínimo) local de f, entonces $\nabla f(x^*) = 0$.
   - Si f es C^2 y x^* es un máximo (mínimo) local de f, entonces $H(x^*)$ es n.s.d. (p.s.d.).
2. Condiciones Suficientes. Si f es C^2, si $\nabla f(x^*) = 0$ y
   - **H**(x) es n.s.d. para todo x en una vecindad de x*, entonces $x*$ es un máximo local de f.
   - $\mathbf{H}(x^*)$ es n.d., entonces $x*$ es un máximo global de f.
   - $\mathbf{H}(x^*)$ es indefinida (no es p.s.d. ni n.s.d.), entonces $x*$ no es un máximo ni un mínimo local.
   - f en una función cóncava, entonces x^* es una máximo global de f.
   - Un máximo global de una función estrictamente cóncava (convexa) sobre un conjunto convexo X, es único.

Existen otras condiciones relacionadas con conceptos de cuasiconvexidad fuerte y estricta que se omiten en esta investigación, pero que pueden ser consultadas en [25].

2.1.2. Optimización con restricciones

Los problemas de optimización sin restricciones representan los casos menos comunes en la práctica. Usualmente los problemas de optimización implican buscar valores óptimos para las funciones, pero con limitaciones o restricciones sobre las variables. En términos matemáticos, las soluciones en problemas con restricciones implica la búsqueda de óptimos en regiones delimitadas por éstas y no en el dominio de la función, como en el caso no restringido.

En términos generales, la forma general de un problema restringido de optimización, conocido como Problema de Programación no Lineal (PPnL) [125], se plantea de la siguiente forma:

Dadas las funciones $f, g_i,\ i = 1, \ldots, k$ y $h_i,\ i = 1, \ldots, m$, definidas en el dominio $\Omega \in R^n$, se define la forma primal de un problema de optimización como:

$$\begin{array}{rl} Min & f(z),\ z \in \Omega \\ sujeto\, a: & g_i(z) \leq 0,\ i = 1, \ldots, k, \\ & h_j(z) = 0,\ j = 1, \ldots, m, \end{array} \tag{2.2}$$

donde f se conoce como función objetivo y las expresiones restantes se denominan, restricciones de desigualdad e igualdad, respectivamente. La región definida por las restricciones del problema se conoce como región factible. Un problema de optimización en el cual la función objetivo, las restricciones de igualdad y desigualdad son lineales, se conoce como problema de programación lineal (PPL). Cuando la función objetivo es cuadrática, pero las restricciones son lineales, se habla de un problema de programación cuadrática (PPC) [125]. La Figura 2.6 ilustra estos dos tipos de planteamientos.

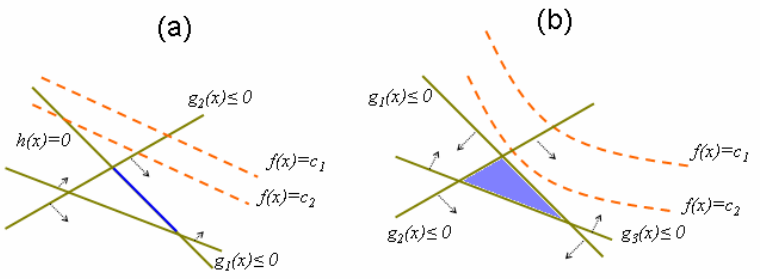

Figura 2.6: (a) Problema de Programación Lineal con dos restricciones de desigualdad ($g_i,\ i = 1, 2$) y una restricción de igualdad (h), (b) Problema de Programación Cuadrática con tres restricciones de desigualdad. Regiones factibles en azul.

2.1.3. Optimización de Langrange

Cuando en un problema de optimización restringido sólo se tienen restricciones de igualdad, se aplica la teoría de Lagrange para caracterizar las soluciones del problema.

Dado un problema de optimización con función objetivo $f(z)$, y las restricciones de igualdad $h_j,\ j = 1, \ldots, m$, se define la función Lagrangiana como sigue:

$$L(z,\alpha) = f(z) + \sum_{j=1}^{m} \alpha_j h_j(z) \tag{2.3}$$

donde los coeficientes α_j son conocidos como multiplicadores de Lagrange (ML).

El objetivo del método de Lagrange consiste en transformar un problema de optimización restringido en un problema sin restricciones y, con ello, aplicar las condiciones de optimalidad correspondientes a problemas no restringidos. Esto es:

Teorema (Lagrange). Una condición necesaria para que un punto $z*$ sea un mínimo de $f(z)$ sujeto a $h_j,\ j = 1,\ldots,m$, con $f,\ h_j \in C^1$, es:

$$\frac{\partial L(z*,\alpha^*)}{\partial z} = 0 \tag{2.4}$$

$$\frac{\partial L(z*,\alpha^*)}{\partial \alpha} = 0 \tag{2.5}$$

para algunos valores de α^*. Estas condiciones también son suficientes cuando la función $L(z,\alpha^*)$ es una función convexa de z. La primera condición representa un sistema de ecuaciones, mientras que la segunda representa las restricciones originales.

En el caso más general, en el que las restricciones son tanto de igualdad como de desigualdad, se aplica el método de Karush-Kuhn-Tucker[4] (KKT) para caracterizar las soluciones del problema.

Teorema (Karush-Kunh-Tucker). Dado un problema de optimización con dominio en un conjunto convexo $\Omega \in R^n$,

$$\begin{array}{rc} Min & f(z),\ z \in \Omega \\ sujeto\,a: & g_i(z) \leq 0,\ i = 1,\ldots,k, \\ & h_j(z) = 0,\ j = 1,\ldots,m, \end{array} \tag{2.6}$$

con $f \in C^1$ y g_i, h_j funciones afines (donde por ejemplo $h(z)$ se puede expresar en la forma $h(z) = Az + b$, donde A y b representan una matriz y un vector, respectivamente). Las condiciones necesarias y suficientes para que un punto $z*$ sea un óptimo son la existencia de $z*$, β^* tal que:

$$\begin{aligned} \frac{\partial L(z*,\alpha^*,\mu^*)}{\partial z} &= 0 \\ \frac{\partial L(z*,\alpha^*,\mu^*)}{\partial \alpha} &= 0 \\ \alpha_i^* g(z*) &= 0,\ i = 1,\ldots,k. \\ g(z*) &\leq 0,\ i = 1,\ldots,k. \\ \alpha_i^* &\geq 0,\ i = 1,\ldots,k. \end{aligned} \tag{2.7}$$

[4]Las condiciones necesarias para problemas con restricciones de desigualdad fueron publicadas por primera vez en 1939 por William Karush en su tesis de maestría [58], pero tomaron renombre después de la presentación del artículo publicado por Harold W. Kuhn y Albert W. Tucker en 1951 [61].

El problema Lagrangiano dual asociado el problema de optimización primal está definido como sigue:

$$\begin{aligned} Max \quad & \Theta(\alpha, \mu) \\ sujeto\, a: \quad & \alpha \geq 0 \end{aligned} \tag{2.8}$$

donde $\Theta(\alpha, \mu) = inf_{z \in \Omega} L(z, \alpha, \mu)$

Es posible demostrar que bajo ciertas condiciones la solución del problema dual y el problema primal coinciden, pero eso está fuera del alcance de esta investigación, para mayores detalles se puede consultar [125].

2.2. Algoritmos Genéticos

Los AGs son algoritmos de búsqueda inspirados en los mecanismos de selección y genética natural. Esta metodología fue propuesta por primera vez por Holland, sus colegas y estudiantes en 1975 [45]. El número de aplicaciones prácticas en las que este método ha probado su efectividad es muy amplia y esto se debe principalmente a que los AGs son computacionalmente simples y muy efectivos en la búsqueda, además de que no se encuentran limitados por una serie de supuestos en el espacio de solución, como suposiciones de continuidad y de existencia de derivadas. De manera general, los AGs se diferencian de otros métodos de optimización y búsqueda (por ejemplo los métodos de gradiente) en cuatro aspectos [45]:

1) Los AGs trabajan con la codificación del conjunto de parámetros, no con los parámetros en sí mismos.

2) Los AGs efectúan la búsqueda sobre una población de puntos (potenciales soluciones) y no con una a la vez.

3) Los AGs emplean la información de la función de adaptación durante el proceso de optimización, en lugar de derivadas u otras medidas de conocimiento auxiliar.

4) Los AGs emplean reglas de transición probabilistas, en lugar de reglas deterministas.

La terminología empleada por un AG encuentra su contraparte en la empleada por la genética natural. La correspondencia entre ambas terminologías se resume en el Cuadro 2.1.

Codificación

La codificación del conjunto de parámetros que caracteriza a los AGs se puede hacer de varias formas, el tipo más común es la codificación binaria, pero ésta puede resultar inapropiada para cierto tipo de problemas. Por ejemplo, en el problema del agente viajero no se recomienda codificar el genoma en binario, pues al hacerlo se generan individuos que no pertenecen a la población [63]. Al codificar en binario, también existen varias posibilidades de representar números reales, las más populares son: a) Codificación en punto flotante, b) Codificación en punto fijo

Genética Natural	Algoritmos Genéticos
cromosoma	cadena
gen	característica, caracter o detector
alelo	valor de la característica
locus	posición de la cadena
genotipo	estructura
fenotipo	conjunto de parámetros, solución alternativa, estructura codificada
epistasis	no linealidad

Cuadro 2.1: Equivalencia entre la terminología de genética natural y artificial.

y c) Codificación de Gray. A lo largo de esta Tesis se utilizó siempre la codificación de punto fijo, ya que se ha comprobado experimentalmente que su aplicación con AGs resulta muy competitiva en comparación con las otras dos alternativas [62]. Este tipo de representación se ilustra en el Cuadro 2.2.

Signo	Parte Entera	Parte Fraccionaria
1 bit	N_e bits	N_f bits

Cuadro 2.2: Codificación de punto fijo de un número real.

2.2.1. Algoritmo Genético Simple

El nombre de Algoritmo Genético Simple (AGS) se debe a Goldberg [36] quien toma el algoritmo de Holland y considera la codificación de los individuo en forma binaria. En términos generales, las características del AGS son las siguientes:

1) Población fija.

2) Selección proporcional.

3) Cruzamiento de un punto.

4) Mutación uniforme.

5) Selección no elitista.

La primera característica se refiere al hecho de mantener fijo el tamaño de la población durante todas las generaciones. La selección proporcional, también conocida como de ruleta, considera la elección de los individuos con base en su medida de aptitud, que es una función que le asigna una calificación a los individuos dentro de la población. Esto es, aquellos individuos mejor calificados tendrán una probabilidad más alta de ser seleccionados y viceversa. En el cruzamiento de un punto, se generan nuevos cromosomas a partir de la mezcla de información de dos individuos previamente seleccionados, donde se elige para ambos un mismo punto de cruce dentro del cromosoma y se procede a intercambiar los segmentos derechos (o

izquierdos, indistintamente) de cada individuo. La decisión de efectuar o no esta operación está dictada por una probabilidad de cruzamiento que se mantiene fija para todas las generaciones y todas las parejas. La mutación uniforme se refiere al intercambio de bits en ciertas posiciones de la cadena genética de los individuos, donde todas las posiciones tienen la misma probabilidad de ser seleccionadas. Para que se de el intercambio, se considera una probabilidad de mutación que se mantiene fija para todas las generaciones y todas las posiciones de los individuos. Finalmente, en la selección no elitista se considera el paso de individuos de una generación a otra sólo a través del proceso de selección aleatoria.

Teóricamente, el AGS se sustenta en el denominado teorema del esquema, que ha sido uno de los pocos mecanismos formales para modelar algoritmos genéticos. En este teorema, se toman en cuenta los tipos de operadores del AGS y se definen rangos para las probabilidades que un esquema se preserve después de la aplicación de estos operadores. Sin embargo, el teorema del esquema ha sido criticado por las siguientes razones: a) no es exacto, b) no es confiable en la predicción del comportamiento de largo plazo del AG, c) sólo considera los efectos destructivos de los operadores genéticos y no los efectos constructivos, d) es muy particular, pues sólo es aplicable dadas las características del AGS. Un tratamiento completo de este tema se puede encontrar en [91][63].

Desventajas del AGS

En [91], [92] y [28], Mitchell *et al* resaltan algunas deficiencias que presenta el AGS de acuerdo a los resultados obtenidos por ciertos experimentos que originalmente tenían la intención de investigar el procesamiento de esquemas y la recombinación en algoritmos genéticos simples.

Uno de los problemas que presenta el AGS es la presencia de *correlación espuria* y se refiere al hecho de que, una vez que ha sido encontrado un representante con alta clasificación de un esquema de orden alto, este esquema se propaga copiosamente y con rapidez en la población, con 0's en las demás posiciones. Esto ocasiona que se disminuya la proporción de otros esquemas que son necesarios para la construcción de la solución correcta [63]. En los experimentos que efectúo Mitchell, se encontró que un escalador aleatorio resultaba más apropiado para resolver un problema que fue diseñado para favorecer a un AG. Este análisis y sus resultados fueron reportados en [92] [28].

Otra desventaja se presenta con los denominados *problemas engañosos*, en los cuales es posible engañar al AG en situaciones que el teorema del esquema no prevé y que puedan llevar a perder la solución óptima irremediablemente. Un tratamiento detallado se puede encontrar en [63].

2.2.2. Algoritmos Genéticos no Convencionales

Para subsanar las deficiencias que el AGS presenta, muchos investigadores han probado otras posibilidades de manipular las poblaciones genéticas. Considerando al AGS como la forma estándar de un AG, se ha llamado [63] Algoritmos Genéticos no convencionales a aquellas variantes que buscan superar las deficiencias del AGS. Enseguida se muestra el Algoritmo Genético Idealizado (AGI) propuesto por [91], en el cual se especifican las características de un AG basado en la hipótesis de los bloques constructores (HBC).

Algoritmo Genético Idealizado

La HBC sostiene que los AGs parten de bloques de corta longitud y poca especificidad para ir integrando bloques más específicos de longitud mayor. De acuerdo al análisis de las limitaciones del AGS y el AGI, se observa que un AG se aproxima a un AGI si se cumple lo siguiente [63]:

a) Muestras independientes. Ningún *locus* se debe fijar a un valor en un número grande de las cadenas de la población.

b) Secuestro de esquemas. Se busca preservar los esquemas deseados mediante una selección lo suficientemente fuerte, pero también suficientemente lenta para evitar correlaciones espurias en algunos esquemas altamente eficientes.

c) Cruzamiento *instantáneo*. La tasa de cruzamiento debe ser tal que el tiempo de cruza que combine dos esquemas deseados sea pequeño con respecto al tiempo que toma descubrir dichos esquemas.

d) Cadenas grandes. Lo suficientemente grande para que el factor de aceleración sea significativo.

Algoritmo Genético de Vasconcelos

El Algoritmo Genético de Vasconcelos (AGV) es una variante de AG [71] que busca aproximarse en la medida de lo posible a las características de un AGI. Las características principales del AGV son:

1. Elitismo total. Se mantiene una copia de los mejores n individuos hasta la generación k, del total de los nk individuos considerados hasta ese momento (Figura 2.7(a)).

2. Selección Determinística. No se considera el ajuste del individuo para determinar los descendientes más deseables. En su lugar, se propone una variedad genética imponiendo una estrategia en la cual se cruza el individuo i para cruzarlo con el individuo $n\text{-}i\text{+}1$ determinísticamente. Con esta estrategia se refuerza el cruzamiento de individuos predefinidos (Figura 2.7(a)).

3. Cruzamiento anular. El genoma no se ve como una colección de bits sino como un anillo cuyo bit de la extrema izquierda es contiguo al bit de la extrema derecha. El intercambio en este caso se da entre semi-anillos, lo cual es equivalente a intercambiar dos porciones no contiguas en la representación del genoma en forma de cadena. Bajo esta metodología, se deben considerar dos parámetros para el intercambio: a) el punto de inicio de intercambio y b) la longitud del semi-anillo (Figura 2.7(b)).

La característica más importante de esta metodología es la selección determinística. En esta estrategia, se destruyen las características deseables de los mejores individuos al cruzarlos deliberadamente con los peores. No obstante, cuando se considera también elitismo total, este análisis permite maximizar la exploración del espacio de soluciones [63]. En resumen, el AGV permite atrapar los mejores esquemas sin restringir el espacio de búsqueda en una forma sensible. El AGV se acerca al AGI por varias razones:

- Determinísticamente se afecta a los posibles *locus* problemáticos con una cruza mejor-peor.
- Con la estrategia elitista se preservan los esquemas deseados y se evita correlación espuria al cruzar individuos disímiles.
- El elitismo total garantiza que el peor individuo de la población k está en el estrato que corresponde al $1/k$ porcentual.

El AGV ha mostrado su superioridad sobre otras variantes de AG y sobre un escalador aleatorio. El análisis se presenta en [71] y se muestra de manera estadística la efectividad de este algoritmo en la optimización de funciones no restringidas de una variable real. A lo largo de esta Tesis se emplea el AGV, pero en la propuesta definitiva que se analiza más adelante se complementa su uso con una estrategia autoadaptable, debido a que también se ha dado gran importancia en este trabajo de Tesis al hecho de evitar en lo posible la intervención del investigador en la determinación de parámetros y, en ese sentido, se sugiere mantener esta prioridad en la elección de las probabilidades de cruzamiento y mutación.

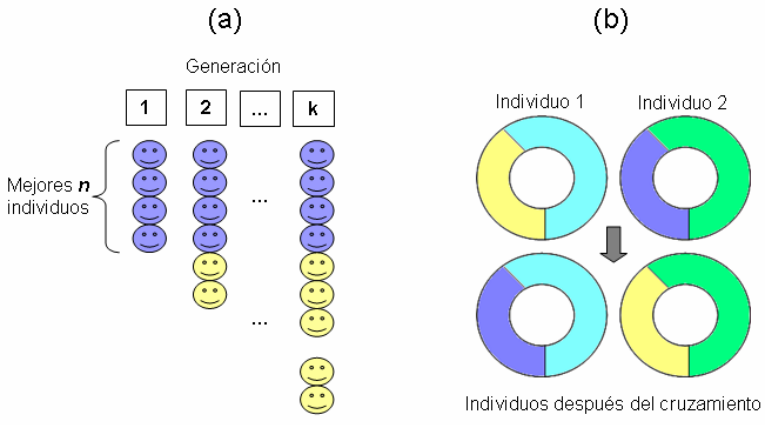

Figura 2.7: (a) Selección Determinística, (b) Cruzamiento Anular

2.2.3. Estrategia Autoadaptable

El mecanismo autoadaptable referido en la sección anterior consiste en incluir los parámetros de mutación y cruzamiento como parte del genoma y, de esta manera, dejar el trabajo de su determinación al AG. Esta estrategia obedece a evitar en lo posible la intervención del investigador en la determinación de parámetros y su selección arbitraria. Este mecanismo se conoce como de autoadaptación poblacional, el cual se describe y prueba experimentalmente en [34] y básicamente mantiene un principio individualista en el que los parámetros afectan a cada individuo por separado dentro de la población. La forma en que se aplicará este principio durante este trabajo de investigación está dada de la siguiente manera:

(i) Probabilidad de cruzamiento (Pc): si p_c representa la probabilidad de cruzamiento para cierto individuo, p_c se codifica en notación de punto fijo. De esta manera, para cada individuo de la población se tendrá una probabilidad de

cruzamiento dada por $(p_c)_i, i = 1, ..., N_p$ y, así, la Pc en cada generación se determina por medio de la expresión: $\sum_{i=1}^{N_p} (p_c)_i / N_p$. Es importante señalar que en la literatura [63] se sugieren valores altos para esta probabilidad.

(ii) Probabilidad de mutación (Pm): si p_m representa la probabilidad de mutación de un bit para cierto individuo, p_m se codifica en notación de punto fijo, con N_e bits en la parte entera y N_d bits en la parte decimal. De esta manera, para cada individuo de la población se tendrá una probabilidad de mutación dada por $(p_m)_i, i = 1, ..., N_p$ y, así, la Pm en cada generación se determina por medio de la expresión: $\sum_{i=1}^{N_p} (p_m)_i / N_p$. Cabe señalar que en la literatura [63] se sugiere que las probabilidades de mutación deben ser bajas para garantizar diversidad en la población.

2.3. Aproximación Polinomial Minimax

Antes de abordar el tema de APM, resulta conveniente definir algunos conceptos importantes relacionados con el problema de clasificación de patrones. Un problema de este tipo consiste básicamente en la separación de un conjunto de objetos u observaciones en diferentes clases. Cada objeto se representa por un vector $\boldsymbol{x}$ y las L clases por medio de $C_l,\ l = 1, \ldots, L$. En el caso más simple, todas las clases son disjuntas y, por tanto, cada observación se asigna a una sola clase. Todas las observaciones posibles definen un espacio de entrada y este espacio queda dividido en regiones de decisión, cuyos límites se denominan superficies de decisión. Las superficies de decisión son funciones de $\boldsymbol{x}$, que pueden ser lineales o no lineales. De esta manera, si m es la dimensión del espacio de características o variables independientes de $\boldsymbol{x}$, entonces una superficie de decisión lineal está caracterizada por un hiperplano de dimensión *m-1*. Un conjunto de datos que puede ser separado sin error en sus respectivas clases por medio de este tipo de superficies, se conoce como linealmente separable. El tipo de modelos que construyen funciones de decisión lineal se conocen como modelos lineales [4]. Ejemplos de modelos lineales son el Perceptrón, Regresión Logística (RL), FDA, MSV[5] [4]. Las superficies de decisión también son conocidas como funciones de discriminación, ya que cumplen con la tarea de decidir a que clase asignar cada vector del espacio de entrada. En la Figura 2.8 se ilustran las funciones de discriminación lineal y no lineal.

El método de APM tiene la finalidad de aproximar funciones polinomiales para la solución de problemas de clasificación de patrones y regresión. El conjunto de datos para este tipo de aproximaciones se conoce comúnmente como conjunto de datos de entrenamiento y, en metodologías de aprendizaje supervisado, se define como sigue: $\tau = \{\mathbf{x}_i, y_i\}$, donde $\mathbf{x}_i$ representa un vector de características o variables independientes y y_i el valor de salida, ambos correspondientes al punto i. En problemas de clasificación, y_i sólo puede tomar valores discretos, mientras que en problemas de regresión y_i representa valores continuos. Particularmente, en un problema de clasificación binaria, sólo se admiten dos valores para esta variable, usualmente se emplean los valores de -1 y +1, para diferenciar a cada clase. Por su parte, en problemas de regresión, y_i puede tomar cualquier valor continuo.

[5] En su diseño para conjuntos linealmente separables.

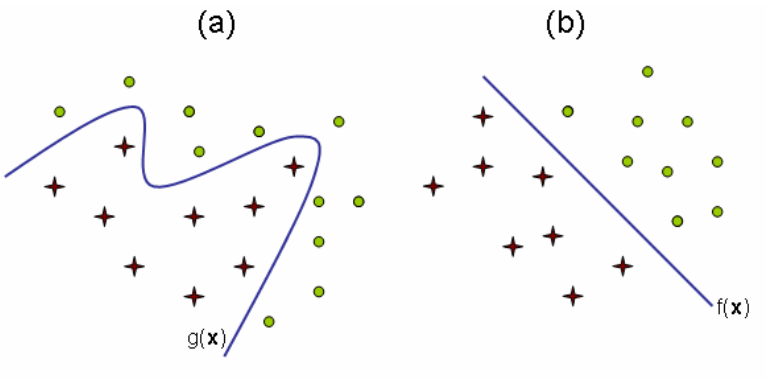

Figura 2.8: (a) Conjunto linealmente no separable con una función de discriminación no lineal g(x), (b) Conjunto linealmente separable con función de discriminación lineal f(x).

De esta manera, el objetivo consiste en aproximar la función:

$$F(\mathbf{x}) = \sum_{i=1}^{v} c_i T_i \tag{2.9}$$

Una forma directa de tratar de resolver este problema consiste en plantearlo como un problema de optimización, donde la función objetivo puede medirse en términos de una medida del error de aproximación. Existen varias alternativas al respecto, probablemente una de las más socorridas en la literatura sea el error de mínimos cuadrados. Bajo esta función, el objetivo consiste en minimizar la suma de errores cuadráticos $\sum_{i=1}^{N} (y - F(x))^2$. La forma de proceder bajo este camino es sustituir la función (2.9) en la expresión de error de mínimos cuadrados y aplicar las condiciones de primer orden sobre los coeficientes polinomiales. Esto deriva en un sistema de ecuaciones que al ser resuelto provee la solución buscada. El problema con este tipo de procedimiento es que los sistemas generalmente resultan mal condicionados para un número relativamente pequeño de coeficientes y de variables independientes, es decir, las matrices que definen el sistema son aproximadamente iguales a una matriz de Hilbert [16], que son numéricamente inestables. Una alternativa que permite sortear estas desventajas es plantear el error de aproximación en términos de la norma L_∞. En (2.9), T_i denota el $i-$ésimo término de la forma de aproximación seleccionada y v denota el número de términos deseados. En L_∞ la idea es reemplazar q $(q = v + 1)$ puntos de τ en la expresión (2.9) y expresar el error como la diferencia entre el valor de salida estimado y el valor verdadero de y, lo que deriva en el siguiente sistema:

$$\begin{pmatrix} e_1 & T_{11} & \cdots & T_{1v} \\ e_2 & T_{21} & \cdots & T_{2v} \\ \cdots & \cdots & \cdots & \\ e_q & T_{q1} & \cdots & T_{qv} \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \\ \cdots \\ c_q \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \cdots \\ y_q \end{pmatrix} \tag{2.10}$$

En términos matriciales, la ecuación (2.10) se expresa como $\mathbf{T}\ \mathbf{c}\ = \mathbf{y}$

El error *minimax* consiste en minimizar el error más grande que se puede obtener en el proceso de optimización. Este error se puede usar como alternativa al error de

mínimos cuadrados. Bajo esta alternativa, la idea es sustituir q puntos de τ en la expresión (2.9) y expresar el error que se obtiene al comparar esa expresión con el correspondiente valor de y, entonces, se tendría el siguiente sistema: $\mathbf{T}\ \mathbf{c}\ = \mathbf{y}$. El objetivo en (2.10) es minimizar el error $e_{max} = max\left(|e_1|, \ldots, |e_q|\right)$, dado $e_k : |e_k| \geq |e_i|,\ i = 1, \ldots, q$, y entonces $e_i = \gamma_i e_k$. Debe notarse que hay $2q$ desconocidas y sólo q condiciones. No obstante, resulta simple resolver el sistema reemplazando la última expresión en (2.10) y aplicando la regla de Cramer para encontrar e_k. Por lo tanto, se tiene:

$$e_k = \frac{D\begin{pmatrix} y_1 & T_{11} & \ldots & T_{1v} \\ y_2 & T_{21} & \ldots & T_{2v} \\ \ldots & \ldots & \ldots & \ldots \\ y_q & T_{q1} & \ldots & T_{qv} \end{pmatrix}}{\gamma_1 D_1 + \ldots + \gamma_q D_q} \tag{2.11}$$

donde $D\left(\cdot\right)$ es la función determinante y D_i es el determinante de la matriz del numerador, dejando fuera la fila i y columna 1.

Es evidente que el valor de e_k se minimiza si el denominador de (2.11) se maximiza y éste se logra cuando: i) los valores de las γ_i's se maximizan, ii) Los signos de γ_i y aquellos del determinante D_i son iguales. La condición i) implica que $|\gamma_i| = 1, i = 1, \ldots, q$; mientras que la condición ii) implica que los determinantes asociados D_i son no singulares; en cuyo caso se dice que satisfacen la condición de Haar y, por tanto, forman lo que se conoce como conjunto de Chebyshev [64].

Los coeficientes $\mathbf{c}$ del polinomio minimax para los q puntos seleccionados del conjunto de entrenamiento resultan a partir de la solución de de (2.10). Si la cardinalidad del conjunto de datos es más grande que q, la solución para el conjunto de entrenamiento completo puede ser calculada usando el Algoritmo de Ascenso (AA) [64].

Algoritmo de Ascenso

Supóngase que el conjunto de datos consiste de N puntos.

(a) Elegir un conjunto con q puntos de τ, el cual se denomina conjunto de referencia. Fijar $i = 1$.

(b) Encontrar los coeficientes del polinomio minimax para el conjunto de referencia. El error resultante más grande se denota por ϵ_i.

(c) Encontrar el error máximo para los $N - q$ puntos restantes en τ.

(d) Si el error máximo global es más pequeño que el error máximo para el conjunto de referencia, los coeficientes corresponden con aquellos del polinomio máximo global.

(e) Sino, elegir el punto cuyo error máximo $x_{i,max}$ e intercambiarlo por un punto en el conjunto de referencia tal que el nuevo error ϵ_{i+1} es más grande que ϵ_i. Hacer $i = i + 1$ y regresar al paso (b).

Este algoritmo computa los coeficientes $\mathbf{c}$ del conjunto minimax polinomial completo. Véase [64] para más detalle. Es importante notar que cuando la condición de Haar no se satisface, no es posible aplicar el algoritmo de ascenso. Sin embargo, es posible agregar una perturbación aleatoria aceptable y trabajar con las variables

resultantes. Esto conlleva a una muy buena aproximación al problema original y permite al algoritmo la labor de generalización [64].

Por otro lado, el Algoritmo de Ascenso es efectivo pero no eficiente: tiene una tasa de convergencia de $O\left(q^4\right)$ que lo hace impráctico para muchas aplicaciones. Sin embargo, es posible lograr tasas de convergencia de $O\left(q^2\right)$ como se muestra en [64], aplicando técnicas formales y heurísticas.

2.4. Funciones Multivariadas de Walsh

El empleo de funciones de Walsh permite definir un número arbitrario de funciones de clasificación multivariada de manera automática y no sesgada. En principio, es posible construir funciones en cualquier base, pero en esta investigación doctoral se ha optado por trabajar en base dos, en la medida en que la codificación de los genes en el algoritmo genético propuesto usa esta representación. En la definición de funciones de Walsh es fundamental la consideración de otro tipo de funciones, conocidas como funciones de Rademaker [42][73][17], que pueden definirse para cualquier base numérica, pero también por consistencia con la codificación de genes en el AG se emplea, en esta Tesis, el tipo definido en código binario. Las funciones de Rademacher en base b se definen de la siguiente manera: Sea $b \geq 2$ un número entero y $\omega = \ell^{\frac{2\pi i}{b}}$. La función de Rademacher en base b es:

$$\varphi_0^{(b)}(x) = \omega^k \tag{2.12}$$

para $\frac{k}{b} \leq x \leq \frac{k+1}{b}$, $k = 0, \ldots, b-1$, y para $n \geq 0$ por:

$$\begin{aligned} \varphi_n^{(b)}(x+1) &= \varphi_n^{(b)}(x) \\ &= \varphi_0^{(b)}(b^n x) \end{aligned} \tag{2.13}$$

Mientras que las funciones de Walsh en base b, se definen por [73]: $wal_0^b(x) = 1$, y si $n = e_1 b^{n_1}, \ldots, e_s b^{n_s}$ con $0 \leq e_s \leq b$ y $n_1 \geq n_2 \geq \ldots$, por: $wal_n^b(x) = \varphi_{n_1}^{e_1}(x), \ldots, \varphi_{n_s}^{e_s}(x)$ donde φ representa la función de Rademacher en base b. Para dimensiones de $m \geq 2$ y $k_1, \ldots, k_m \geq 0$, se tiene [73]:

$$wal_{k_1,\ldots,k_m}^{(b)}(x_1, \ldots, x_m) = wal_{k_1}^{(b)}(x_1) \ldots wal_{k_m}^{(b)}(x_m) \tag{2.14}$$

donde para $k_i \geq 0, i = 1, \ldots, s$; (2.14) forma un sistema ortonormal y completo de funciones en $L^2\left([0,1)^2\right)$ [73].

La ecuación (2.14) es conocida como función de Walsh Multivariada y es usada para la construcción automática y no sesgada de funciones de clasificación. El número de variables y observaciones se define aleatoriamente para cada función.

Capítulo 3

Máquina de Soporte Vectorial

El interés de esta investigación se centra en la determinación automática de parámetros en MSVs, particularmente, el parámetro de regularización C, para lo cual resulta fundamental conocer cuál es el papel que juegan éstos en el proceso de entrenamiento de esta máquina. Por ello, en este capítulo se exponen los elementos teóricos más relevantes de esta metodología en la solución de problemas de clasificación binaria y regresión no lineal. Además, se describen algunos conceptos derivados de la TEA, con los cuales es posible explicar las ventajas teóricas del empleo de MSVs.

3.1. Clasificación Binaria

En el caso de clasificación binaria, el objetivo de una MSV es encontrar un hiperplano que permita discriminar los puntos de un conjunto de entrenamiento que pertenecen a una clase de aquellos que pertenecen a la clase opuesta. La forma en que la MSV realiza esta tarea es maximizando el margen de separación que forman los puntos más cercanos de ambas clases. Geométricamente, supóngase que se tiene un hiperplano H que separa puntos en la clase positiva (y=+1) de aquellos en la clase negativa (y=-1), donde los puntos $\boldsymbol{x}$ que se encuentran sobre el hiperplano satisfacen **w.x**+b, siendo **w** el vector de coeficientes del hiperplano separador y, por tanto, ortogonal a éste. La distancia perpendicular entre el origen y el hiperplano separador está dada por $|b| \,/\, \|\mathbf{w}\|$[1]. Sea d^- la distancia del punto de la clase negativa más cercano a H y d^+ la distancia más corta del punto de la clase positiva más cercano, entonces, el margen de separación entre las clases se define por $\rho = d^+ + d^-$. Ahora bien, supóngase que todos los puntos en τ satisfacen las siguientes restricciones:

$$y_i \left(x_i.w + b\right) - 1 \geq 0 \ \forall i \tag{3.1}$$

Considere ahora todos los puntos en la clase positiva que satisfacen las restricciones de (3.1) con igualdad. Entonces, estos puntos satisfacen la ecuación del

[1]Se deriva aplicando la fórmula para la distancia de un punto a un plano en R^n siguiente: $\frac{|w.z+b|}{\|w\|}$, donde z es un punto en este espacio y w el vector ortogonal al plano.

hiperplano $H_1 : w \cdot x_i + b = 1$, cuya distancia al origen está dada por $|1 - b| / \|\mathbf{w}\|$. Análogamente, los puntos de la clase negativa que cumplen con las restricciones de (3.1) con igualdad se encuentran en el hiperplano $H_2 : w \cdot x_i + b = -1$ y la distancia perpendicular de H_2 al origen es $|-1 - b| / \|\mathbf{w}\|$. Como la distancia de H al origen es $|b| / \|\mathbf{w}\|$, entonces $d^- = d^+$ y, por tanto, $\rho = 2/\|\mathbf{w}\|$. De esta manera, como el objetivo de la MSV es maximizar el margen de separación entre las clases, eso es equivalente a maximizar la separación entre los planos H_1, H_2 con H. Se tiene entonces un problema de optimización que consiste en la minimización de $\|\mathbf{w}\|^2$ (esto maximiza el margen), sujeto a (3.1). La Figura 3.1(a) ilustra lo anterior.

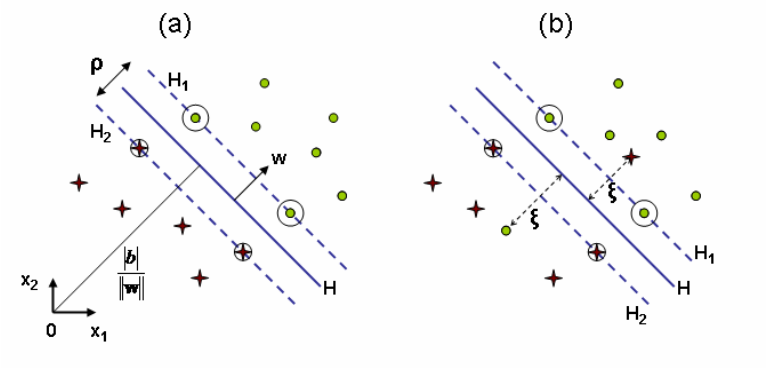

Figura 3.1: (a) Representación geométrica de la MSV lineal, (b) Penalización de observaciones mal clasificadas por medio de variables de relajación ξ.

El proceso anterior describe la forma en que la MSV resuelve problemas con patrones linealmente separables. El caso más general se muestra en la Figura 3.1(b), donde se han introducido un conjunto de variables $\{\xi_i,\ i = 1, \ldots, l\}$, conocidas como variables de relajación, con las cuales se penalizan aquellos puntos que no satisfacen el criterio de separabilidad lineal. Es decir, se agrega una variable de relajación por cada punto de mala clasificación. Este tipo de puntos puede encontrarse en el lado opuesto de la clase a la que pertenecen, ya sea dentro o fuera del margen de separación.
Un problema de clasificación de este tipo, con patrones linealmente no separables y empleando MSVs, se puede resolver con el siguiente problema de optimización cuadrática [43]:

$$\begin{aligned} Min \quad & \tfrac{1}{2} w^T w + C \textstyle\sum_{i=1}^{N} \xi_i \\ sujeto\ a: \quad & y_i\,(x_i.w + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0,\ i = 1, \ldots, N \end{aligned} \tag{3.2}$$

Este problema es conocido como problema primal. Bajo este planteamiento, la función objetivo se compone de dos términos, el primero de ellos tiene por objeto el de maximizar el margen de separación que dista entre los puntos más cercanos en ambas clases, mientras que el segundo término se enfoca en minimizar el número de puntos de mala clasificación. La constante C se conoce como parámetro de regularización, ya que tiene la encomienda de decidir el grado de penalización que se le da a los puntos de mala clasificación y, con ello, se da preferencia a la amplitud

del margen de separación o a la cantidad de puntos mal discriminados que admite el clasificador, según sea el caso. Es decir, si el valor de C es muy pequeño, se pone menos énfasis a la capacidad de clasificación de la máquina y el margen se hace muy amplio; por otro lado, un valor muy grande para C enfatiza el poder de clasificación, pero el margen se hace pequeño [18] [9]. En particular, si el valor de C tiende a infinito, se tiene el problema de separabilidad lineal. La determinación de este parámetro es un punto central para esta investigación, como se ha referido en el estado del arte y como más adelante se expone. Las restricciones de (3.2) indican la clase en la que se encuentra cada uno de los puntos del conjunto de entrenamiento, ésta es la razón de que exista una restricción por cada observación en ese conjunto.

En general, la función objetivo de (3.2) puede expresarse en la forma: $\frac{1}{2}w^T w + CF\left(\sum_{i=1}^{N}\xi_i^{\upsilon}\right)$, donde $F(\cdot)$ es una función convexa. Debe notarse, sin embargo, que el problema de construir un hiperplano que minimice el número de errores en el conjunto de entrenamiento es, en general, NP-completo. Entonces, para evitar que ésto ocurra, se considera el caso en el cual $\upsilon = 1$; o sea, el menor valor de υ para el cual este problema de optimización tiene solución única [18]. Por ello, normalmente se considera la representación mostrada en (3.2). También conviene señalar que aunque existen otras formas de formular el problema de la MSV, por ejemplo la que se muestra en [123] (donde se sustituye el parámetro C por otro tipo de parámetro) en este trabajo doctoral sólo se considera la representación original dada por (3.2).

La función Lagrangeana asociada al problema primal es como sigue:

$$L(w,b,\xi,\alpha,r) = \frac{1}{2}w^T w + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i\left[y_i\left(x_i.w + b\right) - 1 + \xi_i\right] - \sum_{i=1}^{N} r_i\xi_i$$

con $\alpha_i \geq 0$ y $r_i \geq 0$ (multiplicadores de Lagrange), es posible obtener el planteamiento dual correspondiente, diferenciando L con respecto a w, ξ y b e igualando a cero (condiciones de primer orden), se tiene:

$$\begin{aligned}
\frac{\partial L(w,b,\xi,\alpha,r)}{\partial w} &= w - \sum_{i=1}^{N} y_i\alpha_i x_i = 0 \\
\frac{\partial L(w,b,\xi,\alpha,r)}{\partial \xi_i} &= C - \alpha_i - r_i = 0 \\
\frac{\partial L(w,b,\xi,\alpha,r)}{\partial b} &= \sum_{i=1}^{N} y_i\alpha_i = 0
\end{aligned}$$

y sustituyendo las ecuaciones obtenidas en el problema primal, se obtiene la función objetivo de la forma dual:

$$L(w,b,\xi,\alpha,r) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N} y_i y_j \alpha_i \alpha_j x_i^T \cdot x_i$$

Las restricciones $C - \alpha_i - r_i = 0$, junto con $r_i \geq 0$, implican $C \geq \alpha_i$; mientras que si $r_i = 0$, entonces $\xi_i \neq 0$ y, por tanto, $C = \alpha_i$. Entonces, las condiciones de KKT están dadas de la siguiente forma:

$$\alpha_i \left[y_i \left(x_i.w + b \right) - 1 + \xi_i \right] = 0,$$
$$\xi_i \left(\alpha_i - C \right) = 0.$$

Debe observarse que las KKT indican que las variables de relajación sólo pueden ser diferentes de cero cuando $\alpha_i = C$. Los puntos que corresponden a variables de relajación distintos de cero son: $1/\left\|w\right\|$ menos márgenes de error, ya que este margen es menor que $1/\left\|w\right\|$. Puntos para los cuales $0 < \alpha_i < C$ se encuentran a una distancia menor a $1/\left\|w\right\|$ del hiperplano separador.
Dado lo anterior, se desprende, el planteamiento dual , que representa un PPC, está dado de la siguiente manera:

$$\begin{aligned} Max \quad & \textstyle\sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j K\left(x_i, x_j\right) \\ sujeto\ a: \quad & \textstyle\sum_{i=1}^{N} \alpha_i d_i = 0 \\ & 0 \leq \alpha_i \leq C \ i = 1, \ldots, N \end{aligned} \tag{3.3}$$

Los componentes del vector α se conocen como multiplicadores de Lagrange (MLs) y su número corresponde con el número de restricciones en el problema primal (N). La forma dual (3.3) también representa un problema de optimización cuadrática definido sobre un conjunto convexo, con las ventajas que representan estas características en problemas de este tipo, según lo expuesto en el capítulo 2. Además, en la forma dual sólo se requiere la optimización de los valores de α_i, mientras que en (3.2) se deben determinar $\mathbf{w}$, b y $\xi_i,\ i = 1, \ldots, N$.

La función $K\left(x, x_i\right)$ se conoce como kernel y se emplea para proyectar el espacio solución a dimensiones superiores, donde el conjunto de datos linealmente no separable tiene mayor probabilidad de convertirse en linealmente separable [19] [43]. Es importante señalar que no cualquier función puede ser empleada como kernel, para ello es necesario que se cumplan las condiciones de Mercer [84] [43], como se expone en el capítulo siguiente. Ejemplos de funciones kernel empleadas comúnmente son [119] [121]:

$$Lineal: \ K\left(x, x_i\right) = a.x \cdot x_i \tag{3.4}$$
$$Polinomial: \ K\left(x, x_i\right) = \left(a + x \cdot x_i\right)^p \tag{3.5}$$
$$Base\ Radial: \ K\left(x, x_i\right) = exp\left(-\gamma \left\|x - x_i\right\|^2\right) \tag{3.6}$$
$$Perceptrón\ de\ dos\ capas: \ K\left(x, x_i\right) = tanh\left(\beta_0 x^T x_i + \beta_1\right) \tag{3.7}$$

Nótese que para cada tipo de kernel se tiene al menos un parámetro que define su forma funcional. En la práctica, la optimización de este(os) parámetro(s) también ha sido objeto de estudio, pues tradicionalmente su determinación dependía del juicio del investigador. En esta investigación también se aborda el problema de su selección automática. En lo sucesivo, se hará referencia a los kernels anteriores como KL, KPC, KBR y KP, respectivamente.

La función de discriminación o superficie de clasificación correspondiente se construye a partir del valor óptimo del vector α (α^*) y está dada por la siguiente expresión:

$$f(x) = \sum_{i=1}^{N} \alpha_i^* d_i K(x, x_i) + b \tag{3.8}$$

En la expresión (3.8), la función de decisión se expresa como una combinación lineal de la función kernel y los MLs derivados de la optimización de (3.3). Esta característica es común en el tipo de modelos lineales usados para clasificación (y también regresión) que admiten una representación dual (por ejemplo el Perceptrón o el Modelo de Regresión Lineal). En todos ellos, la introducción de una función kernel se da de manera natural ya que, bajo esta representación, la función de decisión (o regresión) correspondiente se expresa como una combinación lineal del producto escalar entre los datos de entrenamiento y el punto en el conjunto de prueba que se somete a evaluación (es decir, x). Esto se ilustra muy bien en el caso de la MSV, observando que, en ausencia de la función kernel en (3.8), la función de decisión sería como sigue: $\sum_{i=1}^{N} \alpha_i d_i x^T \cdot x_i + b$. De igual forma, la función objetivo de la forma dual (3.3) estaría dada por: $\sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j x_i^T \cdot x_j$. Además de las ventajas ya mencionadas, este hecho resalta aún más la importancia de este planteamiento para los modelos que admiten representaciones similares.

En el proceso de optimización quedó pendiente la determinación de b en (3.8), éste se puede calcular empleando los MLs α^*. En (3.8) se aprecia que los valores donde $\alpha_i^* > 0$ son los únicos que aportan en la definición de la función de discriminación y, por eso, a los puntos del conjunto de entrenamiento asociados con ellos se les conoce como **vectores de soporte** (VS), un resultado sumamente importante que le da nombre a esta metodología y que consiste básicamente en elegir un subconjunto de los datos de entrenamiento para la construcción de esta función. Geométricamente (Figura 3.1(b)), los vectores de soporte se encuentran justamente en el límite definido por el margen de separación a ambos lados del hiperplano óptimo, sobre los planos H_1 y H_2, resaltados con un círculo exterior. La función kernel que se emplea en (3.8) tiene el efecto de construir una superficie lineal en el espacio proyectado por $K(\cdot, \cdot)$, pero no lineal en el espacio de los datos de entrenamiento. Más adelante se exponen algunas elementos teóricos relacionados con las funciones kernel.

3.2. Regresión

En el caso de regresión no lineal, la forma de abordar el problema con una MSV es muy similar, pero lo más importante en este caso es la definición de una función de pérdida, la más popular se conoce como función de pérdida ϵ-insensitiva (FPϵI). La FPϵI permite definir un margen de tolerancia, de tamaño ϵ, en torno a la función de regresión que se construya con la MSV (Figura 3.2(a)). Esto es, se penaliza solo a los puntos que se sitúen por fuera de este margen -conocido como ϵ-tubo-, mientras que el error para aquellos puntos que se encuentren dentro del margen es igual a cero (Figura 3.2(b)). La FPϵI es como sigue:

$$L_\epsilon(y,\hat{y}) = \begin{cases} |y-\hat{y}| - \epsilon & \text{para } |y-\hat{y}| \geq \epsilon \\ 0 & \text{en otro caso} \end{cases}$$

donde $\hat{y}$ representa los valores de salida estimados por la MSV.

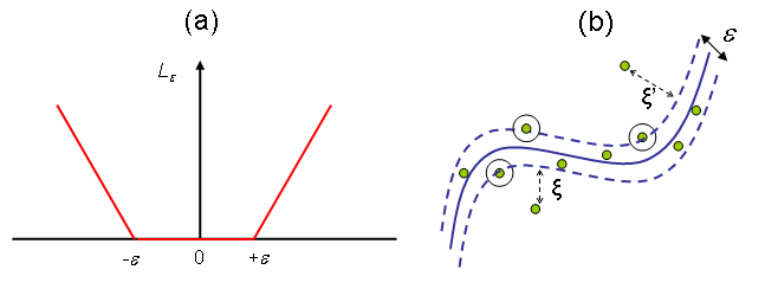

Figura 3.2: (a) Función de pérdida ϵ-Insensitiva, (b) Función de regresión no lineal con MSV y penalización de puntos que sobrepasan los límites del ϵ-tubo.

Para abordar el problema de regresión no lineal se plantea como objetivo la minimización de una función de riesgo $R_\epsilon = \frac{1}{N}\sum_{i=1}^{N} L_\epsilon(y,\hat{y})$, sujeta a la restricción $\|w\|^2 \leq c_0$, siendo c_0 una constante. Nótese que R_ϵ es función de la FPϵ. En la Figura 3.2(b) se observa el caso del problema de regresión sobre patrones linealmente no separables, la función de regresión es no lineal y los puntos que se encuentran fuera del ϵ-tubo son penalizados usando una variable de relajación, como en el caso de clasificación, sólo a que ahora se utilizan dos conjuntos de variables de penalización, correspondientes a puntos que se encuentran de un lado de la función de regresión y puntos que se ubican en el lado opuesto. La forma primal del problema de optimización correspondiente está dado como sigue [122]:

$$\begin{aligned} Min \quad & \tfrac{1}{2}w^T w + C\left(\sum_{i=1}^{N}\left(\xi_i + \xi_i^{'}\right)\right)\alpha_i - \sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j d_i d_j x_i^T \cdot x_j \\ sujeto\ a: \quad & y_i - w^T\varphi(x_x) \leq \epsilon + \xi_i,\ i = 1,2,\ldots,N \\ & w^T\varphi(x_x) - y_i \leq \epsilon + \xi_i^{'},\ i = 1,2,\ldots,N \\ & \xi_i,\ i = 1,2,\ldots,N \\ & \xi_i^{'},\ i = 1,2,\ldots,N \end{aligned} \tag{3.9}$$

Es importante notar que además de los dos conjuntos de variables de penalización, $\{\xi_i\}_{i=1}^N$ y $\left\{\xi_i^{'}\right\}_{i=1}^N$, también se emplean dos conjuntos de multiplicadores de Lagrange: $\{\alpha_i\}_{i=1}^N$ y $\left\{\alpha_i^{'}\right\}_{i=1}^N$.

La forma dual asociada a (3.9) es como sigue:

$$
\begin{aligned}
Max \quad & \sum_{i=1}^{N} y_i \left(\alpha_i - \alpha_i^{'}\right) - \epsilon \sum_{i=1}^{N} \left(\alpha_i + \alpha_i^{'}\right) \\
& -\tfrac{1}{2} \sum_{i=1}^{N} \left(\alpha_i - \alpha_i^{'}\right) \left(\alpha_j - \alpha_j^{'}\right) K(x, x_i) \\
sujeto\ a: \quad & \sum_{i=1}^{N} \sum_{j=1}^{N} \left(\alpha_i - \alpha_i^{'}\right) = 0 \\
& 0 \leq \alpha_i \leq C,\ i = 1, 2, \ldots, N \\
& 0 \leq \alpha_i^{'} \leq C,\ i = 1, 2, \ldots, N
\end{aligned}
\tag{3.10}
$$

La función kernel cumple con la misma función para el caso de regresión, en el sentido de construir una función de regresión que es lineal en el espacio de características, pero no lineal en el espacio del conjunto de entrenamiento.

Se observa en (3.9) y (3.10) que el valor de C también se considera en este planteamiento y cumple prácticamente con la misma función que tenía en el caso de clasificación, pero ahora la regularización se da entre los puntos que se sitúan fuera del margen definido por ϵ [122].

Como en el caso de clasificación, la función de regresión se define de la siguiente manera:

$$
f(x) \quad = \quad \sum_{i=1}^{N} \left(\alpha_i - \alpha_i^{'}\right) K\left(x, x_i\right) + b \tag{3.11}
$$

En este caso, los vectores de soporte son aquellos puntos para los cuales $\left(\alpha_i - \alpha_i^{'}\right) \neq 0$. Por lo tanto, también se consideran solo aquellos vectores que aporten en la definición de (3.11) y, gráficamente, corresponden a los puntos que se encuentran justo en el margen definido por el $\epsilon - tubo$.

Además del parámetro C, se tiene que fijar el valor de ϵ para la solución de problemas de regresión, lo que dificulta su solución en comparación con el caso de clasificación binaria. En problemas de regresión también se debe determinar uno o más parámetros adicionales, dependiendo del tipo de kernel que se use en el planteamiento del problema. Hasta ahora tampoco es posible definir que tipo de kernel emplear para un problema específico dado y la determinación de sus parámetros también es motivo de investigación. Los tipos de funciones usadas en este trabajo fueron polinomiales, de base radial y un nuevo tipo de kernel polinomial derivado de esta Tesis.

3.3. Optimización Secuencial Mínima

Se ha dicho hasta ahora que para el entrenamiento de una MSV *sólo* es necesario resolver un problema de optimización cuadrática restringido y donde es posible garantizar soluciones únicas por las propiedades de convexidad de la función objetivo y de la región factible. Sin embargo, el problema que plantea la MSV en la práctica no es fácil de resolver, ya que es necesario almacenar el kernel en una matriz cuyas dimensiones crecen de manera cuadrática con el número de datos de entrenamiento; es decir, en un conjunto con n observaciones, es necesario computar

una matriz de $n \times n$. Los algoritmos de optimización tradicionales empleados en la solución de este tipo de problemas resultan lentos o incapaces de hacerlo cuando los conjuntos de entrenamiento son muy numerosos.

Para solventar estas deficiencias, se han propuesto en el pasado alternativas capaces de resolver problemas grandes, reduciendo además el tiempo de cómputo. La primera aproximación es un método desarrollado por Vapnik [127] que en la literatura es popularmente conocido como método *Chunking*. El algoritmo *Chunking* toma en cuenta el hecho de que el valor de la forma cuadrática en la ecuación (3.3) no se altera cuando se eliminan las filas y columnas de la matriz que corresponden a los multiplicadores de Lagrange iguales a cero. Por lo tanto, el PPC puede ser descompuesto en una serie de problemas más pequeños, cuya finalidad es identificar todos los MLs distintos de cero y eliminar los que son iguales a cero. En cada iteración, el algoritmo de *Chunking* resuelve un PPC que consiste de MLs distintos de cero resultantes de la etapa previa y agrega un número M de los peores ejemplos que violan las condiciones de KKT. Cada subproblema se inicializa en cada iteración con los resultados del subproblema anterior. En cada etapa, se identifica un conjunto de MLs distintos de cero y, por lo tanto, el último paso resuelve el PPC original. Aunque *Chunking* reduce el tamaño de la matriz a aproximadamente el número de MLs distintos de cero, esta última tampoco puede almacenarse en memoria para problemas a gran escala. En la parte superior de la Figura 3.3 se ilustra la forma en que este algoritmo procesa la información. Las líneas horizontales representan al conjunto de entrenamiento, mientras que las barras simbolizan los MLs siendo optimizados en cada iteración. En cada paso, se va agregando un número fijo de datos y se van descartando los MLs iguales a cero.

Otra alternativa de solución fue planteada por Osuna *et al* [97], quienes probaron un teorema que establece la posibilidad de dividir PPCs grandes en una serie de PPCs pequeños. En esta estrategia se garantiza la convergencia, ya que en cada iteración se agrega un punto que viola las KKT al conjunto del subproblema previo y, por tanto, en cada etapa se calcula la función objetivo completa y se mantiene un punto factible que cumple con todas las restricciones. Estos autores sugieren mantener una matriz de tamaño constante para todos los subproblemas, lo que implica agregar y borrar el mismo número de ejemplos en cada paso (véase la Figura 3.3) y con lo cual es posible atacar problemas con un número arbitrario de datos de entrenamiento. Debe notarse que el algoritmo de *Chunking* satisface las condiciones del teorema, por lo cual está garantizada su convergencia. El algoritmo de Osuna *et al* sugiere agregar y sustraer un ejemplo en cada paso, lo que cual resulta ineficiente, ya que se debe ejecutar el proceso de optimización completo con la única finalidad de asegurar que una sola observación cumpla con las condiciones de KKT. En la práctica, se suelen agregar y sustraer varias observaciones al mismo tiempo para evitar esto. De cualquier forma, se requiere el uso de métodos numéricos para la solución de PPC para todos estos métodos, para los cuales resulta notorio el uso de manipulaciones que garanticen su buen funcionamiento, pues hay muchas cuestiones de precisión numérica que deben ser abordadas [101].

El algoritmo de Optimización Secuencial Mínima fue descubierto por Platt [102] como una alternativa que permite superar las deficiencias de los métodos anteriores, además de reducir el tiempo de cómputo de manera claramente significativa. La idea central de OSM no sólo es resolver de manera iterativa problemas más pequeños, sino resolver el problema con el menor conjunto de datos posible, es decir, aquellos que involucran el uso de únicamente dos observaciones. La razón de que se utilicen dos puntos para el entrenamiento se debe a que los MLs deben cumplir la restricción

de igualdad de la ecuación (3.3), donde evidentemente se requieren al menos dos MLs. En términos generales, el método de OSM elige dos MLs para optimizar, encuentra los valores óptimos para estos multiplicadores y actualiza la MSV para tomar en cuenta los nuevos valores óptimos. La gran ventaja de esta estrategia es que el PPC con dos puntos se puede resolver en forma analítica, con lo que se evitan completamente dificultades de estabilidad numérica y el uso de algoritmos especiales para su solución. Una ventaja adicional es que tampoco es necesario el almacenaje de matrices y los problemas a gran escala pueden atacarse incluso en computadores personales. Existen dos pasos en el proceso de entrenamiento con esta estrategia: la solución analítica para dos MLs y el empleo de un heurístico para elegir que multiplicadores serán optimizados en cada etapa. El algoritmo completo y los detalles algebraicos pueden ser consultados en [102] [101]. En la Figura 3.3 se observa que sólo dos observaciones son optimizadas (analíticamente) en cada iteración, por lo cual el algoritmo resulta ser muy veloz.

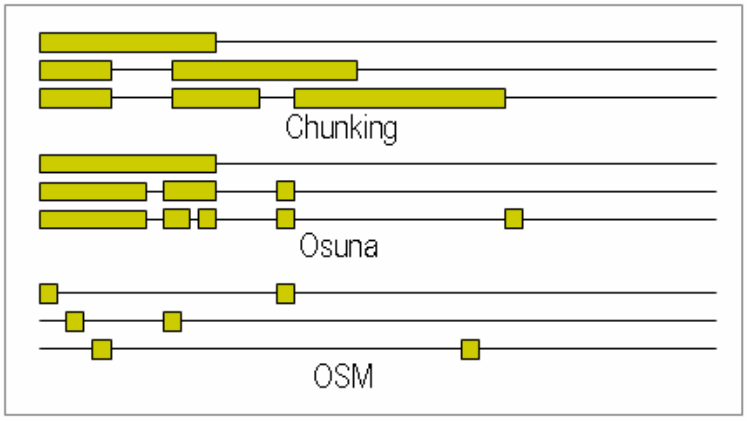

Figura 3.3: Algoritmos de Entrenamiento para MSV.

Desde un punto de vista gráfico, las restricciones de desigualdad de (3.3) implican que los MLs tienen que buscarse dentro de un cuadrado de longitud C, mientras que la restricción de igualdad reduce el espacio de búsqueda a un segmento de recta dentro de ese cuadrado. Por lo tanto, en cada iteración, el algoritmo de OSM tiene que encontrar un valor óptimo de la función objetivo que se encuentre dentro de ese segmento de recta. La Figura 3.4(a) ilustra esta situación.
En términos generales, la región factible para el PPC completo se forma con la intersección de todos los conjuntos que definen las restricciones del problema, es decir, por medio de $N-1$ restricciones de desigualdad y una de igualdad, con la cual se limita aún más el espacio de búsqueda. Por lo tanto, el espacio factible quedará representado por la porción de una superficie lineal (hiperplano) definida en R^n dentro de un hipercubo, cuyas aristas tienen lados de longitud C. Eso quiere decir que el valor de C define explícitamente la región de búsqueda en la cual se explorarán las soluciones del problema. De esta manera, un valor de C muy alto amplia el espacio de búsqueda (aumenta el tamaño del hipercubo) y viceversa. Cuando el valor de C tiende a infinito, el espacio de búsqueda definido sólo con las restricciones de desigualdad coincide con el ortante positivo de R^n y de ahí que los vectores de soporte no se encuentren acotados superiormente, como ocurre en el

caso de la MSV para patrones linealmente separables (conocida en la literatura como MSV con margen duro[2]), cuya forma dual únicamente se diferencia de la MSV para patrones linealmente separables (MSV con margen suave[3]) por la siguiente restricción: $0 \leq \alpha_i,\ i = 1, 2, \ldots, N$. Ahora bien, queda claro que la región factible no se encuentra en el espacio de características, sino en el espacio definido por el número de datos de entrenamiento (N). Por ejemplo, si se cuenta con tres observaciones, con dos variables independientes cada una, éstos se pueden representar gráficamente en un plano cartesiano (R^2), sin embargo, el espacio factible definido por el problema dual de la MSV usando estos tres puntos es R^3 (Figura 3.4(b)), y así sucesivamente.

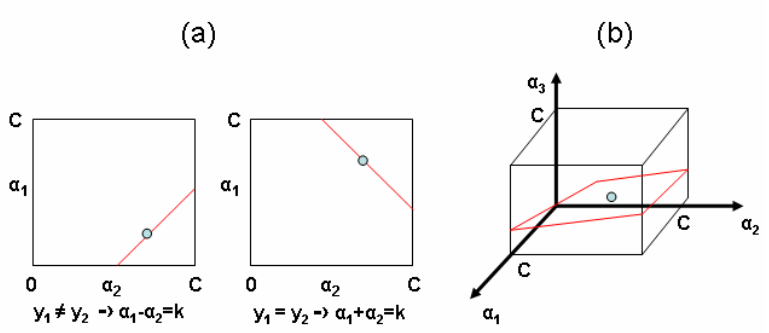

Figura 3.4: (a) Optimización del PPC con dos variables, (b) Representación del PPC en el espacio de R^3.

3.4. Teoría Estadística del Aprendizaje

Antes de finalizar este capítulo, resulta conveniente mencionar algunos conceptos relacionados con la TEA, la cual se comenzó a desarrollar a partir de la década de los 60's por Chervonenkis y Vapnik [131] [130], donde este último ha logrado consolidarla más recientemente [128] [129]. En este apartado se resaltan sólo aquellos conceptos que están directamente relacionados con la MSV.

En la teoría estadística del aprendizaje el interés se centra en la diferencia entre una función objetivo $f(x)$ y el valor de la función de respuesta $F(x)$ que arroja una máquina de aprendizaje (MSV por ejemplo). Esta desviación se presenta en términos estadísticos.

Considerando la función que se desea aproximar ($f(x)$) y dado un conjunto de entrenamiento τ, se dice que un modelo regresivo se encuentra caracterizado por la siguiente expresión:

$$Y = f(X) + \epsilon \tag{3.12}$$

donde $f(\cdot)$ es una función determinista del vector X y ϵ es una variable aleatoria que representa el error entre la dependencia de X y Y. De la misma forma, una máquina de aprendizaje provee una aproximación del modelo regresivo,

[2] *Hard Margin Support Vector Machine.*
[3] *Soft Margin Support Vector Machine.*

donde el valor de salida obtenido con esta aproximación, dado X, se denota por: $Y = F(x, \beta)$.

En términos generales, el objetivo de la máquina de aprendizaje es encontrar el vector de parámetros β que minimicen una función de costo, definida por [129]:

$$E(\beta) = \frac{1}{2} \sum_{i=1}^{N} (y_i - F(x_i, \beta)) \tag{3.13}$$

de tal manera que la diferencia entre la respuesta esperada y_i y la respuesta estimada $F(x_i, \beta)$ por la máquina de aprendizaje sea mínima.

El planteamiento de este problema implica un sacrificio entre el sesgo y la varianza resultantes del uso de $F(x_i, \beta)$ como una aproximación de $f(x)$, el cual se soluciona incrementando infinitamente la muestra de entrenamiento, es decir, el sesgo y la varianza se aproximan a cero simultáneamente incrementando el tamaño de la muestra de entrenamiento, lo que implica un aumento en el tiempo de convergencia [128].

Considerando un modelo de aprendizaje supervisado, la teoría estadística del aprendizaje consiste en seleccionar una función particular $F(x_i, \beta)$ que aproxime la respuesta deseada de manera óptima en un sentido estadístico, donde la selección está basada en un conjunto de N observaciones independientes e idénticamente distribuidos (i.i.d.). La factibilidad del aprendizaje supervisado se basa en la cuestión de si los ejemplos de entrenamiento contienen suficiente información para construir una máquina capaz de lograr una buena aproximación.

De acuerdo a lo anterior, se define $L(y, F(x, \beta))$ como una función de pérdida entre y y $F(x, \beta)$, donde una alternativa es la función de pérdida cuadrática, que se define como:

$$L(y, F(x, \beta)) = (y - F(x, \beta))^2 \tag{3.14}$$

El punto importante en la teoría del aprendizaje estadístico es que esta teoría no depende de manera crítica en la forma en que se defina la función de pérdida [43].

De esta manera, es posible definir una medida de riesgo en términos de la función de pérdida anterior y, por consiguiente, que dependa de la clase de funciones de aproximación $\{F(x, \beta)\}$. Entonces, el objetivo se centra en encontrar una aproximación de ese tipo que minimice una función defina por:

$$Min\ R(\beta) = \int L(y, F(x, w))\, dF_{X,D}(x, y) \tag{3.15}$$

La expresión anterior, representa el promedio de desviaciones que la máquina de aprendizaje (una vez entrenada) incurre al tratar de aprender el fenómeno. Esta medida supone que la distribución original de los datos $P(x, y)$ es conocida, lo cual es lo menos común en problemas reales. Por ello, se sugiere el uso del principio de Minimización del Riesgo Empírico.

El PMREm establece la sustitución de la función de riesgo esperado por una función de riesgo empírico $R_{emp}(\beta)$, definida por:

$$R_{emp}(\beta) = \frac{1}{2l} \sum |y - f(x, \beta)| \tag{3.16}$$

donde puede observarse que esta función no es más que una versión discreta de la primera, con la cual se evita la consideración de la función de distribución original.

Ambas funciones dependen del parámetro β, ya que generalmente hay que recurrir en este tipo de parámetros para definir la máquina de aprendizaje, por ejemplo, en una MSV se tiene el parámetro C, el(los) parámetro(s) del kernel y ϵ en el caso de regresión que, como ya se ha visto, influyen en la elección correcta de la máquina entrenada. Otro criterio considerado en esta definición establece que $R_{emp}(\beta)$ converge en probabilidad a los valores mínimos posibles de la función de riesgo $R(\beta)$, $\beta \in B$, conforme el tamaño de las muestras de entrenamiento se hace infinito, en otras palabras, este criterio garantiza la consistencia de $R_{emp}(\beta)$. Finalmente, establece que la convergencia uniforme es una condición necesaria y suficiente para la consistencia del principio de minimización del riesgo empírico. Un tratamiento formal y más completo de este principio se puede consultar en [128] [129].

3.4.1. Principio de Riesgo Estructural

Un concepto de gran importancia en el TEA es la dimensión de Vapnik-Chervonenkis (dimensión VC), con la cual es posible medir la capacidad de clasificación de un algoritmo estadístico. Formalmente, la dimensión VC es una propiedad de un conjunto de funciones $\{f(\beta)\}$ y puede ser definida para cualquier tipo de función f, pero aquí se consideran funciones de discriminación para el problema de clasificación binaria. Entonces, si un conjunto de l puntos puede ser etiquetado en todas las formas posibles $\left(2^l\right)$, y para cada etiquetamiento es posible encontrar una función perteneciente a $\{f(\beta)\}$ que los clasifique de manera correcta, se dice entonces que ese conjunto de puntos puede ser desmembrado por ese conjunto de funciones. Por lo tanto, la dimensión VC para ese conjunto $\{f(\beta)\}$, se define como el máximo número de puntos de entrenamiento que pueden ser desmembrados por $\{f(\beta)\}$. Por ejemplo, en la Figura 3.5(a), se muestra un ejemplo con 3 puntos, en donde el clasificador lineal (la línea recta) puede separar sin error todas las configuraciones posibles para tres puntos, sin embargo, para el caso de cuatro puntos ese mismo plano no puede clasificar la configuración que se muestra en la Figura 3.5(b) [9], entonces, la dimensión VC es igual a 3.

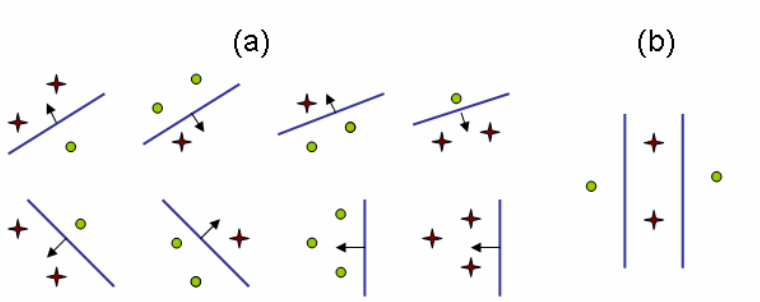

Figura 3.5: (a) En R^2, la dimensión VC para un clasificador lineal es igual a 3, ya que ese es el máximo número de puntos que pueden ser clasificados correctamente en todas sus configuraciones posibles, (b) Configuración en R^2 para cuatro puntos que no es posible clasificar con una función de discriminación lineal.

Considerando nuevamente la función de riesgo empírico $R_{emp}(\beta)$ en (3.16) como

una aproximación del riesgo esperado en (3.15), resulta claro que el $R_{emp}(\beta)$ no garantiza un error bajo del riesgo esperado si el número de observaciones en el conjunto de entrenamiento es limitado, pues un error pequeño sobre el conjunto de entrenamiento no implica un error pequeño en el conjunto de prueba. Por ello, algunos investigadores se han dado a la tarea de desarrollar técnicas estadísticas que permitan sacar ventaja de un conjunto de datos limitado. El Principio de Minimización de Riesgo Estructural es una de estas técnicas, donde la dimensión VC juega un papel importante porque está involucrada en la definición de una cota superior para la función de riesgo esperado y que permite establecer la relación entre la capacidad de una máquina de aprendizaje y su poder de representación. Esto es, el PMRE se basa en la definición de una cota superior para $R(\beta)$, de la siguiente forma:

$$R(\beta) \leq R_{emp} + \sqrt{\left(\frac{h\left(log\left(2l/h\right)+1\right) - log\left(\eta/4\right)}{l}\right)} \tag{3.17}$$

donde h es la dimensión VC. La parte derecha de la desigualdad en (3.17) se conoce como *cota de riesgo* y el segundo término de esta cota de riesgo se denomina *nivel de confianza VC*. A partir de (3.17), se puede observar que no es necesario conocer la distribución $P(x, y)$ y, aunque $R(\beta)$ generalmente no puede ser computada, es posible establecer una cota superior que se calcula fácilmente si se conoce h. Un punto importante que hay que considerar es que la cota superior aumenta conforme lo hace h, por lo tanto, en la práctica es deseable construir modelos con la menor dimensión VC y el menor error de entrenamiento (riesgo empírico). De esta manera, h depende de un conjunto de funciones $\{f_\delta : \delta \in \Lambda\}$ (Λ es un conjunto arbitrario, no necesariamente subconjunto de R^n) que la máquina de aprendizaje puede implementar. Para controlar h, se introduce una estructura de subconjuntos anidados $S_n := \{f_\delta : \delta \in \Lambda_n\}$ de $\{f_\delta : \delta \in \Lambda\}$, tal que:

$$S_1 \subset S_2 \subset \ldots \subset S_n \subset \ldots, \tag{3.18}$$

donde las dimensiones de VC correspondientes satisfacen:

$$h_1 \leq h_2 \leq \ldots \leq h_n \leq \ldots$$

Por lo tanto, para un conjunto de observaciones $(x_1, y_1), \ldots, (x_l, y_l)$, el PMRS elige la función $f_{\delta_l^n}$ en el conjunto $\{f_\delta : \delta \in \Lambda_n\}$ para la cual la *cota de riesgo* es mínima.

Como se ha mencionado, los conceptos anteriores pueden ser aplicados a cualquier metodología de aproximación y la dificultad de computar la cota superior depende en última instancia de que tan complicado resulte la estimación de h. En el caso de la MSV, estos conceptos son de gran relevancia, debido al siguiente resultado, demostrado por Vapnik [128] [129]:

Teorema. Si D es el diámetro de una bola que contiene a todos los vectores de entrada $x_1, \ldots, x_N$. El conjunto de hiperplanos óptimos descritos por la ecuación: $w_0^T x + b_0 = 0$ tiene una dimensión VC h acotada superiormente por:

$$h \leq min\left\{\left\lceil \frac{D^2}{\rho^2}, m_0 \right\rceil\right\} + 1 \tag{3.19}$$

donde $\lceil \cdot \rceil$ es la función mínimo entero mayor o igual a, $\rho = 2/\left\|w_0\right\|$ es el margen de separación y m_0 es la dimensión del espacio de entrada.

La intuición de la expresión anterior es que es posible ejercer control sobre la dimensión VC al seleccionar de manera apropiada el margen de separación, independientemente de la dimensionalidad del espacio de entrada. Más aún, la maximización del margen de separación óptimo tiene el efecto de minimizar la cota superior para h y al mismo tiempo minimizar el error de entrenamiento de la MSV. Por lo tanto, estos objetivos satisfacen lo que se requería para la minimización de la cota para el riesgo esperado.

Capítulo 4

Funciones Kernel y Clasificadores Polinomiales en APM y MSV

En el capítulo anterior se hizo mención de las funciones kernel, con las cuales es posible definir clasificadores lineales en espacios de dimensión superior. Particularmente, en la MSV ayudan a construir funciones de decisión y regresión altamente efectivas, de acuerdo a sus características de no linealidad. También se ha mencionado que es necesario imponer ciertas condiciones para que una función determinada pueda considerarse como un kernel válido. En ese sentido, en este capítulo se presentan algunos elementos teóricos concernientes a la caracterización de funciones kernel y sus propiedades.

Con base en algunos de estos elementos teóricos, un aporte de este trabajo doctoral consistió en la definición de un nuevo tipo de kernel polinomial, denominado Kernel Polinomial MP (KP-MP). La idea de esta nueva función surge al efectuar comparaciones entre las funciones de discriminación de las metodologías de MSVs y APM, empleando en ambos casos funciones polinomiales para su entrenamiento. Se expone más adelante la forma en que se efectuaron estos comparativos, las condiciones bajo las cuales se implementaron, así como los detalles de la caracterización del nuevo KP-MP. Es importante mencionar que este kernel puede emplearse en la solución de problemas de clasificación y regresión, aunque en este capítulo sólo se haga referencia a problemas de clasificación.

De igual forma, aprovechando la similitud entre las funciones de discriminación de las metodologías citadas y mostrando que en ambos casos se puede usar el mismo tipo de funciones polinomiales, se explora en este capítulo la posibilidad de definir una cota para el parámetro de regularización en MSVs, usando un kernel polinomial y los coeficientes derivados del entrenamiento del método de APM.

4.1. Funciones kernel

En este apartado se presentan algunos conceptos relacionados con la caracterización de funciones kernel y sus propiedades. Las definiciones y conceptos principales fueron tomados de [121] y son reproducidos algunos de ellos aquí con la

finalidad de entender mejor la forma en que se demuestra más adelante la construcción del nuevo kernel MP. En [121] [119] puede encontrarse un tratamiento teórico excelente de métodos kernel y muchas de sus variantes.

La definición formal de una función kernel es como sigue:

Definición [función kernel]. Un kernel es una función κ tal que para todo $x, z \in X$ se tiene,

$$K(x, z) = \langle \phi(x), \phi(z) \rangle, \tag{4.1}$$

donde $\langle \cdot, \cdot \rangle$ representa el producto interno entre dos vectores y ϕ es una función (usualmente no lineal) que proyecta a los elementos de X sobre un espacio de producto interno de dimensión superior (espacio de características) Ξ; es decir,

$$\phi : x \longmapsto \phi(x) \in \Xi.$$

Un concepto de gran relevancia para el uso de un kernel es el de la matriz de Gram, que se define como $G_{ij} = \langle x_i, x_j \rangle$, donde x_i, x_j son vectores pertenecientes a un conjunto $S = \{x_1, x_1, \ldots, x_l\}$. Nótese que S puede ser el conjunto de entrenamiento τ que se ha mencionado a lo largo de este trabajo. Ahora bien, considérese la función ϕ y sea κ una función kernel, entonces la matriz de Gram definida en el espacio de características está dada por: $G_{ij} = \langle \phi(x_i), \phi(x_j) \rangle = \kappa(x_i, x_j)$. La matriz resultante también se conoce como matriz de kernel $\mathbf{K} = (\kappa(x_i, x_j))_{i,j=1}^{l}$. La matriz $\boldsymbol{K}$ es simétrica y contiene toda la información necesaria para computar las distancias entre todos los puntos de entrenamiento en el espacio de características. Además, un hecho importante acerca del uso de funciones kernel es que para el aprendizaje en el espacio de características, no se requiere conocer explícitamente la forma de la función (ϕ) que traslada la información a este espacio. Como se ha resaltado antes, la ventaja de trasladar el espacio a uno de dimensión superior, permite el uso de métodos de clasificación o regresión lineales en ese espacio. En la Figura 4.1 se puede apreciar este efecto.

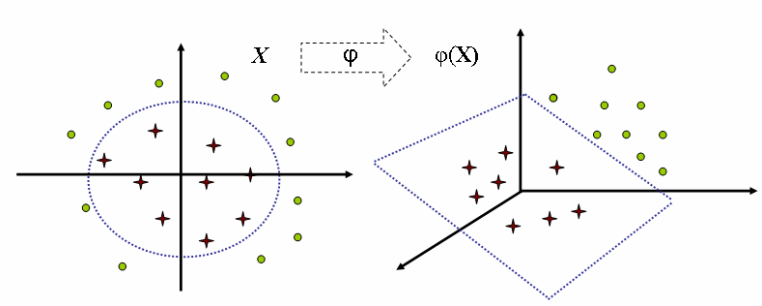

Figura 4.1: Proyectar el problema original a un espacio superior puede simplificar el problema de clasificación.

Ahora bien, si se recuerda el planteamiento dual de la MSV (3.3), los términos de esta matriz aparecen en la función objetivo y su cómputo, en problemas con conjuntos de entrenamiento muy grandes, ocasiona que los métodos de optimización tradicionales resulten poco eficientes o inoperantes y, por ello, la necesidad de usar la técnica de OSM. Además, con este tipo de matrices se pierde información sobre la orientación con respecto al origen de los puntos originales y también información

acerca del alineamiento entre los puntos y los ejes de coordenadas. La razón de esta pérdida de información se debe a que la matriz de Gram es invariante a rotaciones en el sistema de coordenadas; esto es, cualquier rotación de los ejes con respecto al origen no altera los valores de la matriz de productos internos [20].

Para la caracterización matemática de una función kernel, el siguiente resultado es fundamental:

Proposición. Sea X un espacio de entrada finito con $K(x,z)$ una función simétrica definida sobre X, entonces $K(x,z)$ es una función kernel, si y sólo si, la matriz $\mathbf{K} = (\kappa(x_i, x_j))_{i,j=1}^{l}$, es positiva semidefinida.

La generalización de este resultado en un espacio de Hilbert [1] se puede obtener introduciendo las constantes λ_i, de tal manera que:

$$\langle \phi(x), \phi(z) \rangle = \sum_{i=1}^{\infty} \lambda_i \phi_i(x) \phi_i(z) = K(x,z),$$

Este resultado se formaliza con el teorema de Mercer [84], el cual establece condiciones necesarias y suficientes para que una función continua y simétrica $K(x,z)$ admita tal representación, dados los valores de λ_i no negativos. Lo anterior es equivalente a decir que $K(x,z)$ es un producto interno en el espacio de características $\Xi \subseteq \phi(X)$, donde Ξ es el espacio l_2 de todas las sucesiones $\psi = (\psi_1, \psi_2, \ldots, \psi_i, \ldots)$ para las cuales $\sum_{i=1}^{\infty} \lambda_i \psi_i^2 < \infty$. Formalmente, el teorema se expresa de la siguiente manera [84]:

Teorema [Mercer]. Sea $K(x,z)$ una función kernel simétrica y continua definida en $X \times X$, donde X es un espacio compacto subconjunto de R^n. La función $K(x,z)$ puede expandirse en la forma

$$K(x,z) = \sum_{i=1}^{\infty} \lambda_i \phi_i(x) \phi_i(z) \tag{4.2}$$

Con coeficientes positivos, λ_i para todo i. Para que esta expresión sea válida y converja absoluta y uniformemente, es necesario y suficiente que la condición

$$\int_{X \times X} K(x,z) \phi(x) \phi(z) \, dxdz \geq 0$$

Se cumpla para toda $\phi(\cdot)$, tal que

$$\int_{X} \phi(x) \, dx < \infty$$

Las funciones $\phi_i(x)$ son conocidas como funciones propias de la expansión y los números λ_i son los valores propios asociados . El hecho de que la los valores propios sean positivos indica que la función kernel es positiva definida.

El teorema de Mercer establece condiciones necesarias y suficientes para comprobar que una función puede usarse como un kernel válido, pero no dice nada de cómo elegir las funciones $\phi_i(x)$, esto debe hacerse en la práctica. No obstante, es posible construir funciones kernel a partir de kernels válidos empleando una serie de propiedades conocidas como propiedades de clausura, que se enumeran a continuación [121]:

[1] Para un tratamiento más detallado de los espacios de Hilbert puede consultarse [96].

$$\kappa(x,z) = \kappa_1(x,z) + \kappa_2(x,z) \quad (4.3)$$
$$\kappa(x,z) = a\kappa_1(x,z) \quad (4.4)$$
$$\kappa(x,z) = \kappa_1(x,z)\,\kappa_2(x,z) \quad (4.5)$$
$$\kappa(x,z) = f(x)\,f(z) \quad (4.6)$$
$$\kappa(x,z) = \kappa_3(\phi(x),\phi(z)) \quad (4.7)$$
$$\kappa(x,z) = x^{'}Bz \quad (4.8)$$

donde κ_1 y κ_2 son funciones kernel definidas sobre $X \times X$, $X \subseteq R^n$, $a \in R^+$, $f(.)$ una función de variable real en X, $\phi : X \rightarrow R^N$ con κ_3 un kernel sobre $R^N \times R^N$, y $\mathbf{B}$ una matriz $n \times n$ simétrica y positiva semidefinida.

En la caracterización de funciones kernel, estas propiedades son muy útiles en la decisión acerca de si una función dada es candidato para ser un kernel válido y la combinación de éstas para formar otros posiblemente más útiles y complejos. La propiedad (4.3) establece que la suma de kernels es un kernel, (4.4) define kernels a partir de kernels válidos multiplicados por constantes positivas, en (4.5) se tiene que el producto de kernels es un kernel; también lo es el producto de funciones reales (4.6), funciones kernel cuyos argumentos son funciones vectoriales (4.7) y los derivados al usar una matriz positiva definida (4.8).

4.2. Funciones de Discriminación Polinomiales en APM y MSVs

El empleo de APM permite construir una función de discriminación o aproximación de funciones de la forma (2.9), donde pueden emplearse funciones polinomiales durante su implementación. Dicha representación permite expresar de manera explícita todas las relaciones entre las variables independientes y sus potencias. En muchas aplicaciones prácticas resulta conveniente conocer esas interacciones y una medida de su impacto en el modelo, definida precisamente por los coeficientes estimados. En ese sentido, la magnitud y signo de un coeficiente asociado al término monomial provee información valiosa en la relación que guardan las variables involucradas (o sus potencias).

En muchas otras metodologías no es posible obtener este tipo de representación explícita, ya sea por la estructura funcional con la cual se efectúa el entrenamiento o porque se construye una función de decisión que resulta ser compleja y de difícil manipulación en la práctica. Un ejemplo son las redes neuronales, donde la compleja estructura de la red entrenada dificulta la extracción de conocimiento y que, a su vez, limita un mejor entendimiento del problema en cuestión [120].

Dado el caso particular de una MSV, en esta Tesis se exploran las condiciones bajo las cuales resulta posible representar de manera explícita la función de discriminación de esta máquina de aprendizaje [69]. Se encontró que las funciones de kernel polinomial permiten representaciones de este tipo pues, en este caso específico, es posible representar la función kernel como el producto de dos funciones que proyectan los puntos del conjunto de entrenamiento en el espacio de característi-

cas (como en (4.1)), donde la forma funcional de estas funciones ($\phi(\cdot)$) se puede obtener de manera explícita.

Derivado de lo anterior, se exploró la posibilidad de establecer una equivalencia entre las funciones de discriminación para ambas metodologías de aprendizaje. Para ello, se analizó el uso de la siguiente función en (2.9):

$$F(\mathbf{x}) = \sum_{j_1}^{p} \sum_{j_2}^{p} \cdots \sum_{j_m}^{p} \left(x_1^{j_1} x_2^{j_2} \ldots x_m^{j_m} \right), \tag{4.9}$$

Claramente se tiene una expresión polinomial, pero en (3.8) está involucrada una función kernel que, como se ha dicho, puede representar una variedad muy amplia de tipos de funciones. En ese sentido, resulta natural la elección de un kernel polinomial.

De esta manera, se utilizó el kernel tradicional (3.5), que primero tiene que ser desarrollado y expresado como la suma de un conjunto de monomios. Una forma de hacerlo es aplicando el teorema del binomio de Newton en sucesivas ocasiones, como se muestra enseguida:

$$\begin{aligned}
K\left(x^T, x_i\right) &= \left(1 + x^T x_i\right)^p \\
&= \left(1 + \sum_{j=1}^{m} x_j x_{ij}\right)^p \\
&= \sum_{i_1=1}^{p} C_{i_1}^{p} \left[\sum_{j=1}^{m} x_j\right]^{p-i_1} \\
&= \sum_{i_1=1}^{p} C_{i_1}^{p} \sum_{i_2=1}^{p-i_1} C_{i_2}^{p-i_1} \left(x_1 x_{i1}\right)^{i_2} \left[\sum_{j=2}^{m} x_j x_{ij}\right]^{p-i_1-i_2} \\
&= \sum_{i_1=1}^{p} C_{i_1}^{p} \sum_{i_2=1}^{p-i_1} C_{i_2}^{p-i_1} \left(x_1 x_{i1}\right)^{i_2} \cdots \sum_{i_m=j}^{p-\sum_{j=1}^{m-1} i_j} C_{i_m}^{\sum_{j=1}^{m-1} i_j}
\end{aligned} \tag{4.10}$$

donde C_k^l representa las combinaciones de l en k. En una primera aproximación, la ecuación resultante (4.10) se introduce en la función de discriminación (3.8), lo que deriva en efecto en una expresión polinomial explícita, que en realidad mantiene la misma estructura de la función kernel empleada, pero como combinación lineal de sus términos monomiales. Con esta última expresión, se efectúo una comparación término a término con la función (4.9) y se calcularon los coeficientes derivados con ambas metodologías con un conjunto de problemas. Los detalles del experimento se muestran en el capítulo 6.

Con base en esos resultados y observando que aunque en ambos métodos se cuenta con funciones de discriminación explícitas de tipo polinomial, estas expresiones no coinciden en todos sus términos; esto es, existen términos monomiales que aparecen en una, pero no en la otra función y, viceversa.

Con la finalidad de superar este inconveniente, se plantearon dos alternativas:

(1) Emplear el kernel polinomial (3.5) en ambas metodologías, o

(2) Considerar la expresión (4.9) como un tipo de kernel polinomial y usarlo en una MSV. Nótese que en la expresión anterior se tienen todas las combinaciones posibles para las potencias de las variables independientes. La máxima potencia que puede alcanzar una variable independiente es p, por lo que el grado del polinomio es $m \times p$.

Aunque ambas posibilidades permiten obtener expresiones equivalentes término a término, se presentaban dos inconvenientes. En el primer caso, se tenía el problema de verificar que es posible emplear un kernel polinomial, como el usado tradicionalmente en MSVs, en la metodología de APM y, por tanto, verificar si la metodología de entrenamiento admitía esa posibilidad. En el segundo caso, se tenía el problema de verificar que la expresión (4.9) en efecto podría representar un kernel válido, de acuerdo a las condiciones establecidas anteriormente.

En esta Tesis doctoral se abordaron ambos problemas en forma exitosa. En el primero de ellos, se utilizó otra alternativa para representar al kernel polinomial (3.5) que resulta más directa y que se obtiene al aplicar la fórmula multinomial[2], como sigue:

$$\begin{aligned} K\left(x^T, x_i\right) &= \left(1 + x^T x_i\right)^p \\ &= \sum_{l_1 \ldots l_m} \frac{p!}{l_1! \ldots l_m!} \left(x_1 x_{i1}\right)^{l_1} \cdots \left(x_m x_{im}\right)^{l_m} \end{aligned} \tag{4.11}$$

De esta manera, a la expresión anterior se le agrega un coeficiente por cada monomio y se emplea en el entrenamiento con APM, en lugar de utilizar la expresión (4.9). La función de discriminación obtenida con el método de APM es como sigue:

$$F\left(x\right) = \sum_{l_1 \ldots l_m} a_{l_1 \ldots l_m} \frac{p!}{l_1! \ldots l_m!} x_1^{l_1} \cdots x_m^{l_m} \tag{4.12}$$

Los parámetros $a_{l_1 l_2 \ldots l_m}$ ($a_{0,0,\ldots,0} = 1$) de este clasificador se estiman como se mostró en la sección 2.3, por medio del Algoritmo de Ascenso. El número de monomios en (4.12) se definirá en adelante por nM.
La equivalencia entre las funciones de clasificación respectivas se obtiene al sustituir (4.11) en (3.8), que es la función de discriminación de la MSV, y haciendo algunas manipulaciones algebraicas para dejar explícita esta función en términos de las variables independientes. El desarrollo algebraico y la expresión final se muestran enseguida:

[2]Esta fórmula se define como sigue: $(a_1 + a_{12} + \ldots + a_m)^n = \sum_{i_1 \ldots i_m} \frac{n!}{i_1! \ldots i_m!} a_1^{i_1} \ldots a_m^{i_m}$ [104].

$$
\begin{aligned}
f(x) &= \sum_{i=1}^{N} \alpha_i y_i K\left(x^T, x_i\right) + b \\
&= \sum_{i=1}^{N} \alpha_i y_i \left[\left(1 + x^T x_i\right)^p\right] + b \\
&= \sum_{i=1}^{N} \alpha_i y_i \sum_{l_1 \ldots l_m} \frac{p!}{l_1! \ldots l_m!} \left(x_1 x_{i1}\right)^{l_1} \cdots \left(x_m x_{im}\right)^{l_m} + b \\
&= \sum_{l_1 \ldots l_m} \frac{p!}{l_1! \ldots l_m!} \sum_{i=1}^{N} \alpha_i y_i \left(x_1 x_{i1}\right)^{l_1} \cdots \left(x_m x_{im}\right)^{l_m} + b \\
&= \sum_{l_1 \ldots l_m} \sum_{i=1}^{N} \alpha_i y_i \left(x_{i1}^{l_1} \cdots x_{im}^{l_m}\right) \frac{p!}{l_1! \ldots l_m!} \left(x_1^{l_1} \cdots x_m^{l_m}\right) + b \\
&= \sum_{l_1 \ldots l_m} \left[\sum_{i=1}^{N} \alpha_i y_i k_{l_1 \ldots l_m}(i)\right] \frac{p!}{l_1! \ldots l_m!} \left(x_1^{l_1} \cdots x_m^{l_m}\right) + b
\end{aligned}
\tag{4.13}
$$

donde $k_{l_1 \ldots l_m}(i) = \left(x_{i1}\right)^{l_1} \cdots \left(x_{im}\right)^{l_m}$.

Por lo tanto, puede observarse que las ecuaciones (4.12) y (4.13) representan formas algebraicas equivalentes. Este resultado será muy útil más adelante en la definición de un criterio para acotar el rango de valores del parámetro de regularización en MSV.

4.2.1. Kernel Polinomial MP

La segunda alternativa mencionada antes para obtener funciones de discriminación equivalentes entre una MSV y APM, implica demostrar algebraicamente que la función (4.9) en efecto puede ser usada como un kernel válido. El primer inconveniente al tratar de usar esa expresión es que (4.9) tiene como argumento un vector de dimensión m, mientras que en un kernel se requieren dos argumentos. Este problema fue superado considerando la siguiente expresión:

$$
K(\mathbf{x}, \mathbf{x}_i) = \sum_{j_1}^{p} \sum_{j_2}^{p} \cdots \sum_{j_m}^{p} \left(x_1^{j_1} x_2^{j_2} \ldots x_m^{j_m}\right) \cdot \left(x_{i1}^{j_1} x_{i2}^{j_2} \ldots x_{im}^{j_m}\right) \tag{4.14}
$$

donde $\mathbf{x}$ es un vector que representa cualquier elemento del conjunto de entrenamiento, con m variables independientes y x_i, la i-ésima observación tomada de este conjunto. Nótese que la estructura funcional de (4.9) y (4.14) es muy similar, salvo el término $\left(x_1^{j_1} x_2^{j_2} \ldots x_m^{j_m}\right)$, obtenido a partir de los elementos de x_i, pero que puede considerarse como constante.

La estructura (4.14) ahora sí tiene forma de kernel, sólo falta probar que cumple con los requerimientos para serlo. En la proposición siguiente se aborda formalmente la caracterización matemática de esta función como un kernel válido, donde se emplean las propiedades de clausura para construir funciones kernel a partir de otras válidas.

Proposición [Kernel Polinomial MP]. La expresión en (4.14) es una función con dominio en $R^m \times R^m$ y cumple los requerimientos para ser un kernel válido.
Demostración:
El primer paso en esta demostración consiste en observar que la ecuación (4.14) se puede representar en la forma:

$$K(\mathbf{X}, \mathbf{X}_i) = [X_{j1} \otimes X_{j2} \otimes \cdots \otimes X_{jm}]^T \cdot [X_1 \otimes X_2 \otimes \cdots \otimes X_m] \quad (4.15)$$

donde,

$$X_i = \begin{pmatrix} x_i^0 \\ x_i^1 \\ x_i^2 \\ \cdots \\ x_i^{(1+p)^m} \end{pmatrix},\ X_{ji} = \begin{pmatrix} x_{ji}^0 \\ x_{ji}^1 \\ x_{ji}^2 \\ \cdots \\ x_{ji}^{(1+p)^m} \end{pmatrix}, \mathbf{X} = [X_1, \ldots, X_m]\ y\ \mathbf{X}_i = [X_{1i}, \ldots, X_{mi}]$$

siendo $\otimes$ el producto de Kronecker para matrices [47]. De esta manera, desarrollando y aplicando las propiedades de funciones kernel se tiene lo siguiente:

$$\begin{aligned} K(\mathbf{X}, \mathbf{X}_i) &= [X_{j1} \otimes X_{j2} \otimes \cdots \otimes X_{jm}]^T \cdot [X_1 \otimes X_2 \otimes \cdots \otimes X_m] \\ &= \Phi(X_j)^T \cdot \Phi(X) \\ &= \left[\Phi_1(X_j), \cdots, \Phi_{(1+p)^m}(X_j)\right] \cdot \left[\Phi_1(X), \cdots, \Phi_{(1+p)^m}(X)\right]^T \\ &= \sum_{k=1}^{(1+p)^m} \Phi_k(X_j) \cdot \Phi_k(X) \\ &= \sum_{k=1}^{(1+p)^m} i_k^T [x_{j1} \otimes x_{j2} \otimes \cdots \otimes x_{jm}]^T \cdot i_k^T [x_1 \otimes x_2 \otimes \cdots \otimes x_m] \\ &= \sum_{k=1}^{(1+p)^m} \left(x_{j1}^{[k]} \cdot x_{j2}^{[k]} \cdots x_{jm}^{[k]}\right)^T \cdot \left(x_1^{[k]} \cdot x_2^{[k]} \cdots x_m^{[k]}\right) \end{aligned} \quad (4.16)$$

En (4.16) se tiene el producto entre escalares que pueden agruparse en pares como $x_{j1}^{[k]} x_1^{[k]}$ y por la propiedad (4.6), el producto de funciones reales representa un kernel, en particular de funciones constantes, siendo el tipo de kernel más simple posible. Por ello, se tiene lo siguiente:

$$K(\mathbf{X}, \mathbf{X}_i) = \sum_{k=1}^{(1+p)^m} \kappa\left(x_{j1}^{[k]}, x_1^{[k]}\right) \cdots \kappa\left(x_{jm}^{[k]}, x_m^{[k]}\right) \quad (4.17)$$

Finalmente, de acuerdo a la propiedad (4.5), dado que el producto de kernels también lo es, entonces:

$$K(\mathbf{X}, \mathbf{X}_i) = \sum_{k=1}^{(1+p)^m} \kappa^{[k]}(x_j, x) \quad (4.18)$$

Esta última expresión no es más que la suma de kernels válidos que, por la propiedad (4.3), también resulta ser un kernel válido. Con esto último termina la demostración.

Un ejemplo simple de la proposición anterior es como sigue: suponga dos variables independientes ($m = 2$) y $p = 2$, entonces $K(\mathbf{X}, \mathbf{X}_i) = [X_{j1} \otimes X_{i2}]^T \cdot [X_1 \otimes X_2]$, donde:

$$\begin{aligned} X_1 \otimes X_2 &= \left[x_1^0 x_2^0, x_1^0 x_2^1, x_1^0 x_2^2, x_1^1 x_2^0, x_1^1 x_2^1, x_1^1 x_2^2, x_1^2 x_2^0, x_1^2 x_2^1, x_1^2 x_2^2\right] \\ X_{j1} \otimes X_{i2} &= \left[x_{j1}^0 x_{i2}^0, x_{j1}^0 x_{i2}^1, x_{j1}^0 x_{i2}^2, x_{j1}^1 x_{i2}^0, x_{j1}^1 x_{i2}^1, x_{j1}^1 x_{i2}^2, x_{j1}^2 x_{i2}^0, x_{j1}^2 x_{i2}^1, x_{j1}^2 x_{i2}^2\right] \end{aligned}$$

y, en consecuencia:

$$K(\mathbf{X}, \mathbf{X}_i) = (x_1^0 x_{j1}^0)(x_2^0 x_{i2}^0) + \ldots + (x_1^2 x_{j1}^2)(x_2^2 x_{i2}^2) \tag{4.19}$$

Con este resultado, se tiene la posibilidad de entrenar MSV usando este nuevo kernel y, con ello, es posible encontrar una estructura polinomial explícita equivalente a (4.9) y con la cual se pueden hacer comparaciones término a término. Sólo resta verificar la conveniencia de su uso en aplicaciones prácticas, para lo cual se efectuaron una serie de experimentos en aplicaciones de clasificación binaria y regresión no lineal. Los detalles y resultados relacionados con estos experimentos se describen en la sección 6.1.2, en el capítulo 6. Estos resultados fueron reportados en [82]. Un punto adicional importante es que al demostrar la validez del kernel polinomial MP, su uso no se limitaría en aplicaciones con MSVs, sino con otro tipo de métodos de kernel.

4.2.2. Cota Inferior para el valor de C usando APM

En las secciones anteriores se ha probado como es posible establecer expresiones equivalencias entre las funciones de discriminación para problemas de clasificación con APM y MSVs. Con base en estos resultados, en esta Tesis propone una forma para acotar inferiormente el parámetro de regularización C de una MSV en la solución de este tipo de problemas. La idea principal se deriva de la comparación entre las expresiones (4.12) y (4.13), ya que en ambas se tiene el mismo número de monomios y, en consecuencia, es posible establecer una relación uno a uno entre sus respectivos coeficientes. Esta correspondencia se muestra enseguida:

$$a_{l_1 l_2 \ldots l_m}^h = \sum_{i=1}^{N} \alpha_i y_i k_{l_1 l_2 \ldots l_m}(i) \tag{4.20}$$
$$h = 1, \ldots, nM - 1$$

$$a_{l_1 l_2 \ldots l_m}^0 = \sum_{i=1}^{N} \alpha_i y_i + b \tag{4.21}$$

Como se ha mencionado, los coeficientes $a_{l_1 l_2 \ldots l_m}^h$ pueden ser estimados con APM. Los valores $k_{l_1 l_2 \ldots l_m}(i)$ puede ser determinados usando el conjunto de entrenamiento y los y_is son los valores de salida correspondientes en el mismo conjunto. Las únicas variables que se desconocen en estas ecuaciones son el parámetro b y

los términos $\alpha_i, i = 1...N$, que son los multiplicadores de Lagrange asociados a los vectores de soporte de la MSV. El parámetro b y los MLs $(\alpha_i, i = 1...N)$, que corresponden con los vectores de soporte de la MSV, son desconocidos. La determinación de b representa un problema adicional ya que, de acuerdo a la ecuación (4.21), no es posible expresarla en términos de variables conocidas. Este problema se resuelve recordando que este parámetro se puede estimar con el uso de los vectores de soporte y usando las condiciones de KKT del problema primal [9]. Las condiciones de KKT se satisfacen cuando $\alpha_i > 0$ y, por lo tanto, $b = y_i - \sum_j \alpha_j y_j K(x_i, x_j) \, \forall i \in \{1, 2, ..., N\}$ (asumiendo que todos los puntos en el conjunto de entrenamiento (N) satisfacen las KKT). Una forma más adecuada de estimar el valor de b es promediando los valores de todo el conjunto de entrenamiento, como sigue: $b^* = \sum_{i=1}^{N} \frac{y_i}{N} - \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_j y_j K(x_i, x_j)$. Entonces, reemplazando esta última expresión en el lado derecho de (4.21), se tiene:

$$a^0_{l_1 l_2 ... l_m} = \sum_{i=1}^{N} \alpha_i y_i + \sum_{i=1}^{N} \frac{y_i}{N} - \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_j y_j K(x_i, x_j) \tag{4.22}$$

Finalmente, tomando en cuenta la restricción que los valores de $\alpha_i, i = 1, 2, ..., N$ tienen en el planteamiento dual en una MSV para patrones linealmente no separables, donde $0 \leq \alpha_i \leq C, i = 1...N$, entonces es posible establecer una medida con la cual acotar el valor del parámetro C usando las ecuaciones (4.20) y (4.22). Dado que C representa una cota superior para estos parámetros, resulta conveniente usar el término α_{max}, como el máximo de los vectores de soporte en este conjunto $(h = 0, \ldots, nM - 1)$. Con esta representación y usando (4.20), resulta claro que:

$$\begin{aligned} a^h_{l_1 l_2 ... l_m} &= \sum_{i=1}^{N} \alpha_i y_i k_{l_1 l_2 ... l_m}(i) \\ &\leq \alpha^h_{max} \Gamma, \qquad (4.23) \\ if\ \Gamma &= \sum_{i=1}^{N} y_i k_{l_1 l_2 ... l_m}(i) > 0, \end{aligned}$$

$$\begin{aligned} a^0_{l_1 l_2 ... l_m} - \sum_{i=1}^{N} \frac{y_i}{N} &= \sum_{i=1}^{N} \alpha_i y_i - \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_j y_j k(x_i, x_j) \\ &\leq \alpha^0_{max} \Psi, \qquad (4.24) \\ if\ \Psi &= \sum_{i=1}^{N} y_i - \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} y_j k(x_i, x_j) > 0 \end{aligned}$$

Análogamente, si $\Gamma \leq 0$ y $\Psi \leq 0$

$$
\begin{aligned}
a^h_{l_1 l_2 \ldots l_m} &= \sum_{i=1}^{N} \alpha_i y_i k_{l_1 l_2 \ldots l_m}(i) \\
&\geq \alpha^h_{max} \Gamma \qquad (4.25) \\
a^0_{l_1 l_2 \ldots l_m} - \sum_{i}^{N} \frac{y_i}{N} &= \sum_{i=1}^{N} \alpha_i y_i - \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_j y_j k(x_i, x_j) \\
&\geq \alpha^0_{max} \Psi \qquad (4.26)
\end{aligned}
$$

y, por lo tanto, se concluye que:

$$
\begin{aligned}
\alpha^h_{max} &\geq \frac{a^h_{l_1 l_2 \ldots l_m}}{\Gamma}, \qquad (4.27) \\
\alpha^0_{max} &\geq \frac{a^0_{l_1 l_2 \ldots l_m} - \sum_{i=1}^{N} \frac{y_i}{N}}{\Psi} \qquad (4.28)
\end{aligned}
$$

Ahora, como estos valores fueron calculados sobre todos los monomios de $h = 0, \ldots, nM-1$ entonces, el valor α_{max} se calcula como sigue:

$$
\alpha_{max} = Max\left\{\alpha^0_{max}, \alpha^1_{max}, \cdots, \alpha^{nM-1}\right\} \qquad (4.29)
$$

Reiterando, puesto que $\alpha_{max} < C$, entonces α_{max} representa un valor mínimo para el parámetro C y, por lo tanto, puede servir como referencia en el entrenamiento de una MSV.

En síntesis, con este procedimiento se ha probado algebraicamente cómo construir un valor que acote el valor de C, empleando los coeficientes que resultaron al atacar el mismo problema con APM. Evidentemente, este método depende de los resultados arrojados por este clasificador después de su entrenamiento y, por tanto, se requiere verificar su utilidad en la práctica de manera exhaustiva. No obstante, también debe considerarse que la cota para el parámetro de regularización fue construida considerando el peor de los casos, en el sentido de considerar al mayor coeficiente obtenido de entre todos los términos monomiales del clasificador polinomial entrenado. De cualquier forma, lo que se ha obtenido es un valor que permite guiar al investigador en la definición de un rango apropiado para la búsqueda de este parámetro en un sentido práctico, algo que no se ha tomado muy en cuenta en el estado del arte.

En particular, en esta Tesis se usará esta cota como una guía para la codificación genética del parámetro de regularización, donde resulta relevante establecer un rango para la búsqueda del valor apropiado, cuando éste es estimado por medio de un AG. El procedimiento para la obtención automática de este parámetro con AGs y su validación estadística es materia del siguiente capítulo.

Capítulo 5

Máquina de Soporte Vectorial Evolutiva y Validación Estadística

En el capítulo 2 se expusieron algunos conceptos básicos relacionados con la teoría de optimización matemática en problemas restringidos y no restringidos. También se resaltaron algunas características que hacen de los AGs una técnica confiable en la solución de este tipo de problemas. Por otra parte, en el capítulo 3 se presentaron las principales características de una MSV, cuya representación depende de la solución de un problema de optimización con restricciones, tanto en problemas de clasificación de patrones, como en problemas de regresión no lineal. Sin embargo, también se ha mencionado que la solución eficiente de este tipo de problemas de optimización está sujeto a la calibración de ciertos parámetros, a saber: C, parámetro de la función kernel (kP[1]) y ϵ.

Con base en estos antecedentes, en este capítulo se proponen dos metodologías genéticas para la calibración de estos parámetros haciendo uso del AGV y una variante de éste. La primera propuesta desarrollada en esta Tesis se muestra en el siguiente apartado.

5.1. Máquina de Soporte Vectorial Genética: Entrenamiento y optimización evolutiva del parámetro C

La idea central considerada inicialmente en este trabajo doctoral está basada en la solución de la forma dual del problema de optimización que caracteriza a la MSV. Si se recuerda, esta solución implica encontrar los multiplicadores de Lagrange que maximizan la función objetivo de (3.3), pero que al mismo tiempo deben cumplir las restricciones impuestas en ese problema. Cuando se abordan problemas con patrones linealmente no separables, el parámetro de regularización forma parte de

[1]En adelante, se empleará este término cuando se haga referencia al parámetro del kernel, puede ser p para kernel polinomial y kernel polinomial MP ó γ para el kernel de base radial.

estas restricciones, donde los MLs que permitirán definir los vectores de soporte se encuentran acotados inferiormente por 0 y superiormente por C.

De esta manera, la primera solución planteada consistió en la aplicación del AGV en la búsqueda no sólo de los MLs, sino también del valor de C que limita a éstos; es decir, el cromosoma del AG representa a $N+1$ variables en codificación de punto fijo. Este método de aprendizaje se nombra en esta Tesis como Máquina de Soporte Vectorial Genética (MSVG). Con este tipo de codificación, el número de bits que resultan para la representación de todos los valores es de $(N+1)(1+N_e+N_f)$ y el rango de valores reales que cada variable (x) puede tomar es: $-2^{N_e}+2^{-N_f} \leq x \leq +2^{N_e}-2^{-N_f}$, donde N_e es el número de bits en la parte entera y N_f el número de bits en la parte decimal.

La función de adaptación usada para este problema fue la función objetivo del problema dual, donde se considera una mejor adaptación siempre que esta función vaya aumentando, ya que se trata de un problema de maximización.

Otro aspecto importante de esta aplicación es el manejo de restricciones, ya que tradicionalmente los AGs han sido diseñados para resolver problemas de optimización no restringida y, por lo tanto, fue necesario emplear alguna metodología adicional.

La forma en que se buscó superar este problema fue el uso de una función de penalización, que se sustrae (en problemas de minimización se suma) a la función objetivo, pero su magnitud va disminuyendo conforme los puntos que representan potenciales soluciones al problema van cumpliendo con las restricciones del mismo. La forma de esta función de penalización se muestra en (5.1), que se compone de dos términos: la constante Z, que representa un número positivo muy grande y el segundo término a la derecha que es función de esta constante y que se sustrae cada vez que una nueva restricción es satisfecha por las variables en cuestión. El parámetro s representa el número de restricciones que se cumplen en una fase del entrenamiento, ocasionando que la penalización sea cero cuando toma el valor de $N+1$; es decir, cuando todas las restricciones han sido satisfechas.

$$P(x) = \begin{cases} \left[Z - \sum_{i=1}^{s} \frac{Z}{t}\right] - f(x) & s \neq t \\ 0 & \text{en otro caso} \end{cases} \tag{5.1}$$

Diferentes pruebas con esta metodología en la solución de problemas de clasificación binaria y múltiple fueron reportados en [67], [70]. Los detalles de estos experimentos, así como las conclusiones se exponen más adelante, en la sección 6.2.1.

5.2. Máquina de Clasificación Genética: Entrenamiento usando OSM y optimización evolutiva de parámetros

La metodología propuesta en este apartado representa una variante a las presentadas en [15] [89]. La idea central en este método consiste en la determinación genética de los parámetros en la MSV, pero el entrenamiento se lleva a cabo a través de un algoritmo de optimización. El algoritmo empleado en esta Tesis fue el de OSM de Platt. En [15] se emplea una metodología para la determinación

del parámetro C, el parámetro kP y variables adicionales que indican la reducción en la dimensionalidad del problema. Este método se nombra en esta investigación doctoral como Máquina de Clasificación Genética (MCG).

En esta investigación se utilizó una cadena de bits de longitud igual al número de variables independientes del problema, donde un "1" indica la presencia de la variable en esa posición y "0" en otro caso. Por ejemplo, en un problema con 7 variables independientes, la cadena "1001011" indica que únicamente las variables 1, 4, 6 y 7 serán consideradas durante el entrenamiento. Además de la determinación de estos parámetros, se agregan la probabilidad de cruzamiento y la probabilidad de mutación, tomando en cuenta la metodología presentada en el capítulo 2. La agregación de estos últimos parámetros convierte al AG en una variante auto-adaptable con lo cual se evita sesgo en la fijación de estos valores.

Otra variante importante que se emplea en esta Tesis es el uso de un AG no convencional, a saber el AGV que, como ya se ha dicho, emplea un tipo de cruzamiento anular, selección determinista y mutación uniforme. Este punto es importante, ya que se ha probado estadísticamente [71] que el uso de estrategias genéticas convencionales resultan menos favorable en la solución de problemas de optimización.

Finalmente, a diferencia de otras aproximaciones, en este trabajo doctoral sí se toma en cuenta el rango de valores que pueden tomar estos parámetros y que está determinado por el número de bits en la codificación empleada; en particular, el empleo de codificación de punto fijo. De esta manera, se define un rango de bits empleado para el parámetro C, ya que una determinación inadecuada puede incidir en el ajuste que pueda lograrse con el algoritmo. Al respecto, se propone emplear los resultados del capítulo anterior, respaldados en la obtención de coeficientes polinomiales para definir un subconjunto de búsqueda adecuado para este parámetro. Cabe recordar que esto último ha sido probado analíticamente únicamente con el empleo de funciones kernel polinomiales, aunque podría extenderse a otro tipo de funciones, siempre que sea posible definir de manera explícita los términos asociados a las variables de entrenamiento en la función de discriminación de la MSV. Este problema es materia de futura investigación.

Con relación a la función de adaptación, se definieron dos propuestas. La primera de ellas considera el porcentaje de buenas clasificaciones obtenidas por el algoritmo OSM sobre un subconjunto de prueba elegido en forma aleatoria. La segunda propuesta también toma en cuenta esta medida de ajuste, pero derivada de la aplicación de (k-FCV). Una de las ventajas al aplicar esta última propuesta consiste en el empleo de todos los datos para la fase de entrenamiento y, por lo tanto, no se desperdicia nada de información. Sin embargo, el costo de su utilización se da con el aumento en el tiempo de entrenamiento. Los resultados derivados de la aplicación de ambos procedimientos se presentan en el siguiente capítulo.

En síntesis, la MCG que se propone para la especificación de parámetros y selección de variables en MSVs se reduce a:

1) Definir el genoma para la representación de los parámetros: C, kP, P_c, P_m y fS[2]

2) Generar una población aleatoria inicial,

3) Evaluar la función de aptitud para cada individuo de la población,

[2]Este parámetro representa una cadena binaria y se usa para el mecanismo de selección de variables que efectúa el AG, como se explicó anteriormente.

4) Aplicar el Algoritmo Genético de Vasconcelos, junto con la variante autoadaptable,

5) Terminar si se cumple con el criterio para hacerlo, de lo contrario, regresar al paso 3).

Las características que resaltan en este algoritmo son: a) Codificación de punto fijo, b) El uso de una estrategia genética eficiente (AGV) y, b) El mecanismo de autoadaptación, donde P_c y P_m se determinan durante el proceso evolutivo.

Finalmente, es importante señalar que esta metodología es aplicable, en principio, con varios tipos de funciones kernel, pero se emplea el kernel polinomial con la finalidad de definir un rango aceptable para el parámetro C.

5.3. Criterio de Validación Estadística de Algoritmos de Clasificación

Una contribución adicional de este trabajo doctoral es el estudio de una propuesta para la validación estadística de modelos de clasificación binaria.

Se han sugerido en el pasado algunos métodos para abordar el problema de selección de modelos. Sin embargo, existe un amplio debate en la comunidad científica del área de aprendizaje de máquina, con relación al mejor método de comparación [93].

Por ejemplo, muchos investigadores consideran sólo algunos conjuntos de datos como representativos y, con base en los resultados obtenidos sobre éstos, se decide la efectividad de alguna metodología en particular [56]. En otras ocasiones, se proponen una serie de problemas que, bajo ciertos criterios, se consideran complicados, son generados de manera artificial y son empleados para comparar algoritmos de aprendizaje [124] [86] [112]. Este tipo de comparativos tiene el efecto de demostrar que alguna técnica en particular resulta exitosa en la solución de tales problemas, en comparación con sus competidores. Sin embargo, bajo esos criterios no existe una garantía de que la estrategia ganadora realmente vaya a ser exitosa en un sentido amplio. Además, de acuerdo a los denominados "teoremas de no libre almuerzo" (*No free lunch theorems*) para optimización [138], resulta claro que el hecho de que un algoritmo que funciona bien en un conjunto limitado de funciones no garantiza, en forma alguna, que sea competitivo en un conjunto diferente de problemas.

De manera alternativa, se han propuesto metodologías estadísticas para la comparación de algoritmos de aprendizaje. Por ejemplo, en [60] se plantea el uso de una técnica conocida como *10-FCV estratificada* como solución a este problema. Mientras tanto, en [21] se presenta una taxonomía sobre una serie de preguntas estadísticas, pero el enfoque se emplea sólo para el caso de problemas con dominio simple, donde los algoritmos de aprendizaje se comparan con datos para un solo problema que, usualmente, cuentan con datos limitados.

En esta Tesis se considera el caso de problemas con múltiples dominios, donde dos algoritmos de aprendizaje se comparan con más de un conjunto de datos (casos) que tienen su origen en fuentes distintas. La determinación de la superioridad de un modelo basando su comparación en problemas con múltiples dominios es, tal vez, la cuestión más difícil y fundamental en el área de aprendizaje de máquina [21]. En esta Tesis se propone el uso de criterios estadísticos para atacar este problema. No obstante, se reconoce su enorme complejidad problema y, por ello, no es

posible argumentar que el método que se propone ofrecerá conclusiones de validez universal, tampoco que el conjunto de casos, que (teóricamente) representa a un número infinito de problemas de este tipo, resulta ser necesariamente la mejor elección posible. Aún así, en esta Tesis doctoral se decidió trabajar en el caso de múltiples dominios, ya que se busca en forma objetiva, probar la robustez del modelo genético de clasificación al evaluar su eficiencia y eficacia sobre la base de conjuntos de problemas que sean grandes, estadísticamente acotados y con el menor sesgo posible en su construcción; en lugar de probarlos sobre un simple y limitado dominio, como es generalmente el caso. Esta propuesta comienza con el reconocimiento de que, para establecer condiciones estadísticas de validación, generalmente se requiere información suficiente. Desafortunadamente, contar con un número abundante de problemas de clasificación con el que puedan hacerse ejercicios de este tipo no es tarea fácil; además, existe la opción de generar problemas de manera artificial pero, en ese caso, el criterio del investigador es decisivo. Por lo tanto, hay dos cuestiones directamente relacionadas para verificar la validez de este método: a) la generación automática y no sesgada de problemas de clasificación y b) la comprobación estadística.

Para abordar la primera cuestión, se propone una metodología para generar un número determinado de problemas en forma automática, donde la forma de asignar valores a las variables dependientes e independientes no depende del *criterio* del investigador, sino que se lleva a cabo empleando funciones de Walsh y Rademaker. De esta manera, se tiene un método no sesgado para generar problemas multivariados de clasificación binaria. No obstante, debe decidirse la forma en que se elegirán el número de variables independientes, el número de observaciones, así como el número de casos de estudio a ser generados.

La dificultad de definir el número de variables y el tamaño de los conjuntos es enorme, debido a la infinidad de posibilidades que se tiene. Por ello, se sugiere un criterio aleatorio para establecer un límite para ambas variables y definir los rangos correspondientes en cada problema, lo cual sigue siendo limitado, pero mucho menos que elegir un conjunto de funciones arbitrario.

Con relación al número de conjuntos de entrenamiento, donde también se reconoce que el número de casos que pueden ser considerados se extiende al infinito, pues aún cuando el número de variables independientes fuera limitado, los valores reales que pueden tomar estas variables -discretos y/o continuos- y sus combinaciones son innumerables. Además, también tendrían que considerarse todas las posibles combinaciones de valores de salida que cada uno de estos conjuntos de datos podría tomar. Para abordar este problema, se propone el uso de la siguiente fórmula para determinar el número de casos de clasificación que deben ser generados para el análisis [79]:

$$n = \frac{z^2 s^2}{e^2} \tag{5.2}$$

La expresión anterior es comúnmente empleada en estudios de muestreo para la estimación de tamaño de muestra para poblaciones infinitas. El valor de e se conoce como error de muestreo y determina la precisión, es decir, refleja en que medida el promedio de la variable real de la población se aleja de la media obtenida a partir de la muestra. El valor de z es el percentil para un $(1 - \alpha)$ por ciento de confianza de una distribución normal y s^2 es la varianza.

Ahora que se cuenta con un criterio para definir el número de conjuntos de entrenamiento, falta definir la forma en que se realiza la validación estadística. En

esta Tesis se propone el uso de pruebas de hipótesis estadísticas para la diferencia de medias de los ajustes obtenidos con cada técnica. Para definir la efectividad de la prueba, se presentan dos alternativas:

a) Prueba t-pareada (PtP). En la comparación de dos métodos de aprendizaje, las observaciones se consideran pareadas. Entonces, se toman las diferencias $W_i = Y_i - Z_i,\ i = 1, \ldots, n$ (Y y Z representan dos muestras sometidas a comparación) para la prueba de hipótesis $H_0 : \bar{W} = 0$. Por lo tanto, cuando W_i puede considerarse una observación derivada de una distribución normal, entonces:

$$\frac{\bar{W} - (\mu_1 - \mu_2)}{s\left\{\bar{W}\right\}} \tag{5.3}$$

que tiene una distribución t con $n-1$ grados de libertad, donde μ_1, μ_2 son las medias respectivas de ambas poblaciones, $\bar{W} = \frac{\sum_i W_i}{n}$ y $s\left\{\bar{W}\right\} = \frac{n\sum_i \left(W_i - \bar{W}\right)^2}{n-1}$ [78].

b) Prueba de Permutación (PP). La idea de esta prueba fue introducida por primera vez por R. A. Fisher en la década de los 30's [24] con la intención de dar argumentos teóricos en favor de la prueba t de *Student*. Con esta prueba es posible efectuar contrastes de Hipótesis entre dos muestra independientes tomadas de dos distribuciones, es decir, supóngase que $y_1, \ldots, y_n$ y $z_1, \ldots, z_m$ dos muestra extraídas de las poblaciones F y G, respectivamente. Una vez observadas estas dos muestras, se desea probar la *hipótesis nula* H_0 de que no hay diferencia entre las poblaciones F y G, es decir, $H_0 : F = G$. La prueba de hipótesis comienza con un *estadístico de prueba* $\hat{\theta}$. El estadístico que se usa en esta Tesis es la diferencia de promedios entre los ajustes de las dos poblaciones; esto es, $\hat{\theta} = \bar{z} - \bar{y}$ (es decir, la diferencia en los promedios de ambas muestras), donde se considerará que la hipótesis nula es falsa para valores grandes de $\hat{\theta}$. Una vez observado el valor de $\hat{\theta}$, el *nivel de significancia alcanzado* de la prueba (NSA), que denota la probabilidad de observar al menos ese valor alto para el estadístico de prueba [24]:

$$NSA = Prob_{H_0\left\{\hat{\theta}^* \geq \hat{\theta}\right\}} \tag{5.4}$$

lo que significa que mientras más pequeño sea el valor de NSA, mayor será la evidencia en contra de H_0. $\hat{\theta}^*$ es la distribución de la hipótesis nula, es decir, la distribución $\hat{\theta}$ si H_0 es verdadera. En la práctica el valor de NSA es un estadístico de prueba que se usa para decidir acerca de la validez de la hipótesis nula, con base en ciertas convenciones. De manera más formal, se elige un valor de α (umbral) y se toma la decisión de rechazar H_0 si NSA es menor que este valor[3]. La prueba de permutación de Fisher es una manera ingeniosa de estimar NSA para la hipótesis $F = G$. El procedimiento se resume enseguida:

1. Elegir B vectores independientes $g_1^*, g_2^*, \ldots, g_B^*$, donde cada vector contiene n z's y m y's, cada una de las cuales es seleccionado aleatoriamente de entre el conjunto de todos los C_n^N posibles vectores.
2. Evaluar el estadístico θ correspondiente a cada vector de permutación, $\hat{\theta}^* = S$

[3]Generalmente se considera evidencia fuerte en contra de H_0 para valores de NSA menores que 0.05.

3. Aproximar NSA_{perm} por medio de $N\hat{S}A_{perm} = \#\left\{\hat{\theta}^*(b) \geq \hat{\theta}\right\}/B$.

En los experimentos efectuados en esta Tesis, el número de permutaciones B se determina como el número de iteraciones para el cual $N\hat{S}A_{perm}$ se estabiliza.

5.4. Máquina de Regresión Genética

La MCG fue diseñada para la solución de problemas de clasificación binaria, pero es posible extender su aplicación a la solución de problemas de regresión no lineal. Esta nueva propuesta fue nombrada como Máquina de Regresión Genética (MRG). La extensión más importante para lograr esto, consiste en la introducción del parámetro ϵ de la función ϵ-insensitiva dentro del cromosoma y de la introducción de una nueva propuesta para la función de adaptación del AG, que consiste en la minimización del error cuadrático medio. Estas modificaciones no afectan en mucho la forma en que opera la MCG, pero eso no quiere decir que el problema pueda resolverse de manera trivial, basta recordar que el problema de regresión es mucho más complicado que el problema de clasificación, pues se tienen que calibrar al menos tres parámetros para su solución y el número de restricciones de la forma dual es prácticamente el doble.

Con la finalidad de medir la efectividad de esta estrategia en la selección de los parámetros y la solución de estos problemas, se propuso su comparación con otras dos técnicas que han sido socorridas en la literatura en la solución de este problema. Se trata de la propuesta presentada por Cherkassky [14] y el empleo de Validación Cruzada.

En el primer método se sugiere el uso de $C = max\left(\left|\overline{y} + 3\sigma_y\right|, \left|\overline{y} - 3\sigma_y\right|\right)$ para el cálculo de C, es decir, con base en la media y la desviación estándar de los valores de salida del conjunto de entrenamiento. La expresión anterior es válida únicamente para funciones kernel de base radial y se deriva tomando en cuenta la función de regresión construida por la MSV y las restricciones del problema dual en las que se involucran los vectores de soporte y la constante C. Los detalles se encuentran en [14].

El cálculo de ϵ se hace por medio de $\epsilon \propto \frac{\sigma}{\sqrt{n}}$, donde σ es la desviación de ruido del problema. No obstante, la expresión anterior sólo es recomendable cuando el tamaño de N es chico, ya que de lo contrario se estiman valores para ϵ demasiado pequeños. Dado este inconveniente, el autor sugiere el uso de la expresión alternativa: $\epsilon = \tau_1\sigma\sqrt{\frac{lnN}{N}}$, donde τ_1 es una constante que tiene que ser calibrada de manera empírica. El valor de σ en la práctica puede ser estimado en esta aproximación por medio de la siguiente ecuación:

$$\hat{\sigma} = \frac{1}{n-d}\sum_{i=1}^{N}(y - \hat{y})^2 \tag{5.5}$$

donde $\hat{y}$ son los valores de salida obtenidos al estimar un polinomio de alto orden. A lo largo de esta investigación, se han probado varias metodologías para la estimación de este factor, como la aplicación de la técnica de Polinomios Genéticos Mutivariados (PGM), PM o MSV con kernel polinomial de grado elevado[4].

[4]En esta aproximación se definen valores arbitrarios para la MSV.

Es importante señalar que a pesar de la estimación de C y ϵ, resta la calibración del parámetro kP de la función de base radial. Esta calibración se llevo a cabo sugiriendo algunos valores predeterminados o seleccionados usando validación cruzada. Para la determinación de estos parámetros empleando CV, se propone un conjunto de valores para cada parámetro, algunas veces seleccionados de manera aleatoria en algún intervalo predefinido o simplemente propuesto bajo cierto criterio de separación entre ellos.

En el siguiente capítulo se analizan de manera experimental estas tres propuestas y, con base en los resultados, se evalúan las ventajas y/o desventajas de cada una de ellas.

Capítulo 6

Experimentos y Análisis de Resultados

En este capítulo se describen los experimentos efectuados a lo largo de esta investigación doctoral, en los cuales se abordaron dos tipos de problemas: clasificación de patrones y regresión no lineal. Como el objetivo primordial durante este proceso ha sido la búsqueda de una estrategia que ayude en la selección óptima de parámetros en MSV, particularmente el parámetro de regularización C, se abordan de manera empírica las estrategias presentadas en los Capítulos 4 y 5, que han contribuido de manera directa e indirecta en la búsqueda de este objetivo. También se describen los detalles de la experimentación y las pruebas estadísticas -y no estadísticas- de validación en cada caso.

6.1. Aproximación Polinomial Minimax y Máquina de Soporte Vectorial: Evidencia empírica

En el Capítulo 4 se describe la forma en que es posible obtener funciones de discriminación equivalentes a partir de dos estrategias de clasificación que operan bajo principios diferentes. El elemento esencial de este hecho se basa en la construcción de clasificadores polinomiales en ambos casos y la relación entre sus elementos constituyentes (monomios). A continuación se describen los resultados empíricos derivados de la comparación entre clasificadores polinomiales obtenidos con una MSV y usando APM y, enseguida, se describen algunos experimentos donde se implementa el kernel polinomial MP.

6.1.1. Análisis de estructuras polinomiales con MSV y APM

En esta sección se describen los experimentos efectuados para determinar si existe alguna semejanza entre una MSV entrenada con un kernel polinomial clásico y un PGM [65], entrenado con la expresión (4.9). Los experimentos realizados para este propósito consistieron de dos problemas de regresión, con datos obtenidos del

UCI Machine Learning Repository (UMLR)[1] y *StatLib-Dataset Archive* (*Statlib*)[2]. El primer conjunto se abrevia con las siglas *mpg* y se refiere al problema de predecir el rendimiento en consumo de combustible aplicado a varios modelos, considerando 7 características: 1) número de cilindros, 2) año, 3) origen, 4) desplazamiento, 5) caballos de fuerza, 6) peso y 7) aceleración. El número de observaciones consideradas fue de 392. El otro conjunto de datos se abrevia como *bodyfat* obtenido de *Statlib* y esta compuesto por ciertos estimados del porcentaje de grasa corporal, medida en 252 hombres, que representa la variable dependiente. En este conjunto se tienen 14 variables independientes, referentes básicamente a diferentes medidas corporales. Los detalles de este experimento y sus resultados fueron reportados en [69].

Se emplea la metodología de PGMs, donde el método de APM se combina con un AG para elegir sólo un subconjunto de monomios de la expresión final, como función de clasificación o regresión, según sea el caso [65]. Los parámetros genéticos utilizados con esta aproximación para el conjunto *mpg* fueron: P_c igual a 0.9, P_c igual a 0.005, tamaño de la población N_p igual a 50 individuos y 50 generaciones (N_g). Se emplearon los mismos parámetros para el conjunto *bodybat*, a excepción de P_c que se fijó en 1. El grado máximo permitido para cada variable en ambos problemas fue de 2 y 1, respectivamente. El número de monomios que se mantuvieron durante todo el entrenamiento fue de 20 para ambos conjuntos, donde el AG tiene la función de elegir a los mejores de ellos de entre todos los posibles monomios en la expresión (4.9), que depende del número de variables independientes y los grados permitidos para cada variable. Es importante señalar que los valores de los parámetros se eligieron después de observar otras alternativas que resultaron menos adecuadas en términos de ajuste.

Los parámetros ϵ y C correspondientes a la MSV, fueron aproximados empleando la metodología propuesta por [14], donde se usaron polinomios de alto orden para la estimación de σ, determinados a través de una MSV con un kernel polinomial de grado 5. Los grados del kernel polinomial se fijaron en función del mayor número de expresiones monomiales que coincidían para ambas metodologías. Los valores obtenidos fueron: C=0.62, p=2 y $\epsilon = 0{,}0228$ para *mpg* y de C=0.50, p=3 y $\epsilon = 0{,}024$ para *bodyfat*.

Los resultados demuestran que, para estos ejemplos particulares, el desempeño de la MSV fue superior en ambos casos, comparando el ECM sobre los conjuntos de entrenamiento (14.5 % para GMP contra 6.6 % para la MSV en *mpg*; 8 % contra 2.5 % en *bodyfat*, respectivamente). Más allá del ajuste obtenido, la contribución más importante de este experimento consistió en encontrar una expresión explícita para la función de regresión obtenida con la MSV, es decir, fue posible obtener de manera explícita los coeficientes correspondientes de la expresión derivada del uso de kernels polinomiales. De esta manera, es posible que el mejor ajuste de la MSV se lograra a expensas de emplear un mayor número de términos en la expresión polinomial final.

Finalmente, aunque se sabía de antemano que había una disparidad, no sólo en el número de términos de las expresiones polinomiales derivadas de ambas metodologías, sino también en el grado de las variables en sus respectivos monomios, fue posible observar los términos que tenían más peso en cada caso al observar la magnitud y signo de sus coeficientes, véase [69].

[1] http://www.ics.uci.edu/ mlearn/MLRepository.html.
[2] http://lib.stat.cmu.edu/datasets/.

Cabe señalar que los experimentos fueron efectuados usando el software *LIBSVM* v-2.8 [12][3] tanto para el entrenamiento de la MSV, como para la estimación de σ.

Estos ejercicios marcaron la pauta de un resultado todavía más importante en esta Tesis y es el hecho de poder emplear kernels polinomiales como función de aproximación en la metodología de APM, pero también de poder usar la expresión (4.9) como un kernel polinomial que contiene a todos los monomios posibles para el grado seleccionado y, por lo tanto, como un kernel válido que pudiera ser usado en el entrenamiento de MSV (o en otros métodos de kernel), como fue demostrado en el Capítulo 4.

6.1.2. Kernel Polinomial MP

En esta sección se muestran los resultados obtenidos con la implementación del KP-MP en la solución de problemas de clasificación y regresión con MSV. Los experimentos y resultados fueron reportados en [82], donde se muestra que con este nuevo kernel es posible obtener errores de clasificación y regresión comparables a los obtenidos con funciones kernel polinomiales y de base radial, es decir, los kernels más populares en la literatura y que han resultado muy efectivos en múltiples aplicaciones. Los datos de clasificación empleados en este análisis fueron tomados de *UMLR* y se refieren a mediciones de diabetes efectuadas sobre un grupo de mujeres descendientes de tribus de indios Pima. Las observaciones para regresión también se refieren a casos de diabetes, pero la idea concierne el estudio de los factores que afectan los patrones de dependencia de insulina en niños[4].

La validación de los resultados se efectúo de manera estadística en el caso de clasificación, empleando la prueba conocida como $5 \times 2cv$, que es una prueba t *pareada* basada en 5 iteraciones de 2-FCV. Esta prueba fue propuesta en [21], donde se explica con detalle su funcionamiento en la comparación de modelos. La calibración de los parámetros kP y C se efectuó utilizando 10-FCV.

El problema de regresión fue validado usando el estadístico *Press* [104], que se usa en la selección de modelos y se calcula sumando los cuadrados de los residuales obtenidos a partir de N entrenamientos de la MSV, dejando una observación diferente en cada entrenamiento. Cada residual es obtenido como la diferencia entre el valor estimado y el valor verdadero, correspondiente a esa observación. Este estadístico es comúnmente empleado en la comparación de modelos de regresión y es equivalente al error obtenido con LoO-CV cuando se evalúa el ECM. El error de estimación para cada función se determinó una vez que se eligieron los parámetros adecuados para cada kernel, el valor de ϵ de la función ϵ-insensitiva y el parámetro C. Estos valores fueron elegidos empleando la estrategia propuesta por [14], en el caso en que se usó la KBR. La estimación de σ se efectúo aplicando la metodología de APM [69], eligiendo el grado con el cual se obtuviera el menor valor para el estadístico *Press*. El valor de ϵ previamente estimado también fue usado en los modelos con funciones kernel polinomiales, pero los grados de estos polinomios y el valor del parámetro C se calibraron empleando LoO-CV, donde el valor apropiado

[3]Este software se encuentra disponible en: http://www.csie.ntu.edu.tw/ cjlin/*LIBSVM*/.

[4]Esta información puede descargarse en:
http://www.liacc.up.pt/ ltorgo/Regression/DataSets.html.

para C se escogió de entre el conjunto [1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1028][5], como aquel con el menor valor para el estadístico *Press*.

El cuadro 6.1 resume los resultados de las pruebas de hipótesis para el problema de clasificación, en donde se observa que no existe evidencia estadística suficiente en favor de ningún modelo y, por lo tanto, la implementación del KP-MP, compite con el KPC y la KBR tradicionales.

Prueba	**Est t**	**Hip Nula**
KPC vs. KP-MP	-0.12	No se rechaza
KPC vs. KBR	0.41	No se rechaza
KP-MP vs. KBR	0.41	No se rechaza

Cuadro 6.1: Resultados de la prueba *t-pareada* para la comparación de ajustes de clasificación entre kernels polinomiales y de base radial.

En el caso de regresión, los parámetros de la MSV fueron estimados usando 6-FCV. El Cuadro 6.2 muestra los valores del estadístico *Press* para cada uno de los kernels, donde el KP-MP y el KPC presentan los mismos valores, siendo ligeramente superados por el KBR. De igual forma, se muestran los errores LoO-CV y un valor de R^2 calculado con el estadístico *Press* [104]. La R^2 indica el poder de expresividad de la máquina de aprendizaje correspondiente, siendo el KBR el que mejor desempeño obtuvo en este caso.

Modelo	***Press***	R^2	**error LoO-CV**
KPC	14.4	0.34	0.34
KP-MP	14.5	0.33	0.34
KBR	12.5	0.43	0.29

Cuadro 6.2: Estadístico *Press* y error LoO-CV para la comparación de ajustes de regresión entre kernels polinomiales y de base radial.

La implementación se llevó a cabo haciendo uso del software desarrollado por [38][6], escrito en lenguaje *Matlab*. Fue necesario hacer unas modificaciones al código para la implementación del KP-MP.

6.2. Selección Automática de Parámetros en MSVs y Validación Estadística: Evidencia empírica

6.2.1. MSVG

En esta primera estrategia se hace uso de AGs para el entrenamiento y codificación del parámetro C en la MSV. La idea básica consiste en incluir dentro del genoma no sólo al parámetro C, sino también a los multiplicadores de Lagrange que

[5]Esta propuesta se sugiere en: http://www.autonlab.org/tutorials/, aunque no deja de ser arbitraria.

[6]Disponible en el sitio web: http://www.isis.ecs.soton.ac.uk/resources/svminfo/.

permiten resolver el problema de optimización dual de una MSV en problemas de clasificación. Se aplicó la codificación de punto fijo para la representación y conversión a valores reales de estas variables. Esta estrategia evolutiva estuvo sustentada en el AGV, donde se usó la función objetivo del problema de optimización dual de la MSV como criterio de aptitud para este algoritmo. El manejo de las restricciones del planteamiento dual trae consigo una dificultad adicional, debido a que los AGs fueron diseñados inicialmente para la solución de problemas de optimización no restringida. Este inconveniente fue superado usando la estrategia de penalización de la sección 5.2.

Diferentes experimentos efectuados con este método demostraron su efectividad en la solución de problemas de clasificación. Uno de ellos consistió en la solución al problema *XOR* (OR Exclusivo), ejemplo clásico de no linealidad de patrones, donde se encontró con relativa facilidad la solución teórica. Con relación a los parámetros utilizados en esta aproximación se estableció lo siguiente: a) en codificación de punto fijo se utilizó un bit para signo, uno para la parte entera y 20 para la parte fraccionaria; b) se emplearon funciones kernel polinomiales con grados dos, tres y cuatro; c) $P_c = 0{,}9$ y $P_m = 0{,}05$; d) $N_p = 200$ individuos y; e) $N_g = 200$ generaciones en cada corrida. Los detalles y resultados fueron reportados en [66] [68].

Los otros experimentos se efectuaron sobre conjuntos de datos que representan problemas más complicados que el *XOR*. Estos datos fueron tomados de *UCI-MLR* y han sido utilizados ampliamente en la literatura para comparar métodos de clasificación. El primer conjunto de datos se refiere a información de cáncer de pulmón (c-pulmón), problema de tres clases (tipos de cáncer) que originalmente tiene 27 observaciones y 56 atributos. Este conjunto fue enriquecido con más información empleando *natural splines* (*NS*) [26], para contar con 180 observaciones. El segundo conjunto es conocido en la literatura como el problema de reconocimiento de vinos, que cuenta con tres tipos de vino (3 clases) de una misma región en Italia, para un total de 178 observaciones -59 de la clase 1, 71 de la clase 2 y 48 de la clase 3- con 13 atributos cada una. Se consideró un tercer conjunto tomado de *UCI-MLR* conocido como *Iris Plant Database* (Iris), considerado quizás como el problema más famoso de clasificación de patrones en la literatura. Este conjunto también cuenta con 4 atributos y 3 clases con 50 instancias cada una, donde cada clase se refiere a un tipo diferente de iris de planta. Es importante señalar que uno de los atributos es linealmente separable del resto.

Dos tipos de aplicaciones se efectuaron con esta información. En la primera de ellas, se aplicaron las metodologías de *uno-contra-todos* y *uno-contra-uno*, pero únicamente se consideró la eficiencia obtenida en las clasificaciones binarias correspondientes sobre conjuntos de prueba y entrenamiento. Es decir, únicamente se tomaron en cuenta las predicciones promedio obtenidas en cada caso como criterio de comparación (con tasas superiores o cercanas al 90 %). Además de los problemas antes citados, también se efectuaron pruebas sobre un conjunto de clasificación binario generado de manera sintética, en donde se consideró una combinación complicada de funciones trigonométricas y trascendentales, obteniéndose un error de entrenamiento cercano a 98 % sobre el conjunto de entrenamiento y del 100 % sobre el conjunto de prueba [7][67].

En la segunda aplicación se trató con el problema de clasificación multi-clase,

[7] A excepción del problema *XOR*, en todos los experimentos se dividieron los conjuntos originales en un grupo de entrenamiento y uno de prueba, con una proporción aproximada cercana al 70-75 %.

donde se aprovecharon los mismos conjuntos comentados en el párrafo anterior, a excepción del problema artificial. Aquí también fueron usadas las metodologías de *uno-contra-todos* y *uno-contra-uno*, pero esta vez se utilizaron criterios eficientes de asignación para cada clase y para medir la efectividad de la clasificación múltiple. En el caso de *uno-contra-todos* se empleó la estrategia de el *ganador-toma-todo* (*winner-takes-all*) y en la metodología de *uno-contra-uno* fue acompañada de la estrategia de *voto-mayoritario* (*majority-voting-scheme*). En la primera estrategia, se construyen k clasificadores binarios (uno por clase) y cada nueva observación (en el conjunto de prueba) se asigna de acuerdo con el clasificador que mejor ajuste arroje (Figura 6.1(a)). Para la segunda estrategia, se construyen $k(k-1)/2$ clasificadores binarios para las k clases (k=3 en los conjuntos considerados) y la clasificación de nuevas observaciones se efectúa conforme a la clase que mayores votos obtenga, esto es, dada una nueva observación x_i, la clase l gana un voto si el clasificador l_m (clase l vs. clase m) dice que x_i pertenece a la clase l, de otra forma, se asigna un voto a m; una vez que las $k(k-1)/2$ emiten su voto, x_i es asignada a la clase con mayores votos (Figura 6.1(b)).

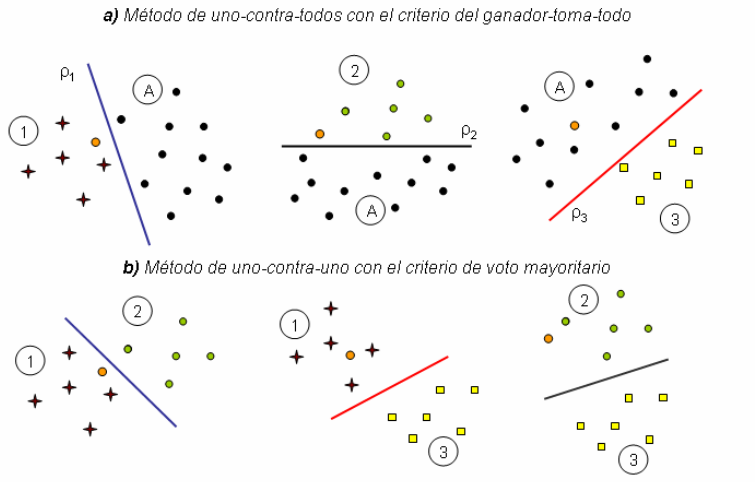

Figura 6.1: a) En esta estrategia, el punto naranja (nueva observación) se asigna a la clase 1, ya que ésta consigue el mayor número de votos (2), b) Si a(ρ_2)>a(ρ_1) y a(ρ_2)>a(ρ_3), entonces el punto naranja se asignará a la clase 2, a(.) significa ajuste.

Tomando en cuenta los ajustes sobre conjuntos de prueba se encontró que el método de *uno-contra-todos* junto con la metodología de el *ganador-toma-todo* se comportó mejor en promedio con alrededor de 95 % contra aproximadamente 92 % que obtuvo su competidora. Todos los entrenamientos se efectuaron con un kernel de base radial Gaussiana con parámetro $\sigma_k = 2$ y el valor de C se obtuvo de manera automática, donde se usó una codificación de punto fijo, con 4 bits en la parte entera y 20 en la parte fraccionaria. Los resultados de este ejercicio fueron presentados en [70] y se muestran en el Cuadro 6.3.

A pesar de los buenos resultados obtenidos con esta aproximación, se tiene el inconveniente de que la función kernel se tiene que almacenar en una matriz o tiene que ser computada en cada evaluación de la función objetivo durante el proceso de aprendizaje. La alternativa de computar la matriz hace el proceso de entrenamiento

Problemas	Ajuste de Entrenamiento	Ajuste de Prueba	C
uno-contra-uno			
c-pulmón	99.3	96.7	4.17
vinos	96.0	92.9	13.67
iris	88.0	88.0	12.03
uno-contra-todos			
c-pulmón	92.0	93.3	12.00
vinos	94.7	92.9	11.00
iris	97.6	100.0	12.01

Cuadro 6.3: Ajustes de la MSVG en problemas de clasificación múltiple con las estrategias *uno-contra-uno* y *uno-contra-todos*.

más rápido, pero sus dimensiones están relacionadas directamente con el número de observaciones en el conjunto de entrenamiento, provocando problemas de estabilidad numérica para conjuntos grandes o simplemente la incapacidad de manejar estas matrices cuando el número de observaciones es muy elevado. Además, si también se quisiera optimizar el parámetro de la función kernel de forma evolutiva, se tendrían que computar las matrices correspondientes para cada parámetro en el espacio de búsqueda del AG. La otra alternativa tiene el inconveniente de ser muy lenta en el proceso de entrenamiento, ya que se tiene que computar el kernel cada vez que se requiera evaluar la función de aptitud en el AG.

Por las razones anteriores, el siguiente paso en esta investigación fue la búsqueda de una alternativa que permitiera superar estos contratiempos y que, al mismo tiempo, pudiera aprovechar las ventajas que la estrategia genética presenta en la calibración de parámetros en MSV, particularmente la exitosa aplicación del AGV, que hasta ahora ha resultado ser eficaz. En lo que resta de esta sección, se muestra el proceso seguido en la sección 5.3.1 para desarrollar esa alternativa y la propuesta de validación estadística al hacer comparativos con otras alternativas de aprendizaje (sección 5.3.2).

El software para la implementación de estos experimentos fue desarrollado en lenguaje FoxPro.

6.2.2. Generación Automática de Problemas de Clasificación.

La fuente de información empleada para la validación estadística de problemas de clasificación se basó en la generación automática de un número de casos que resultaran estadísticamente significativos. Los parámetros elegidos aquí fueron: nivel confianza del 95 por ciento, α=0.05, error estándar del 5 por ciento, e=0.05, y para la varianza se consideró el peor de los casos de acuerdo al rango de valores que pueden tomar las variables bajo prueba. Es decir, como las pruebas se hacen sobre ajustes definidos como la proporción de buenas clasificaciones en conjuntos de prueba, su rango de valores se encuentra en el intervalo $[0, 1]$. Por tanto, el caso extremo -con mayor varianza- se da cuando 50 % de las observaciones presenta un ajuste de 0 y el resto un ajuste de 1. La varianza en este caso es de aproximadamente 0.25. La varianza máxima corresponde al peor de los casos y generalmente

se usa cuando se desconoce de antemano el valor real de la varianza poblacional para la variable en cuestión, como ocurre en esta Tesis. De esta manera, el tamaño de muestra que se obtuvo con este procedimiento fue de $n = 400$.

En los experimentos siguientes, se emplearon 400 problemas de clasificación multivariados generados a partir de funciones de Walsh y Rademaker. Los conjuntos inicialmente generados fueron subdivididos a su vez en conjuntos de entrenamiento y prueba, en una proporción de 70 y 30 por ciento, respectivamente. La selección de las observaciones en cada caso se realizó aleatoriamente.

El número de observaciones y variables independientes también se determinó de manera aleatoria, acotando los rangos de número de observaciones entre 50 y 400, mientras que el rango para el número de variables independientes fue de 2 a 6. La distribución de los respectivos tamaños se presenta en la tabla siguiente:

Variables	**Número**	**%**
2	107	26.8
3	88	22.0
4	95	23.8
5	75	18.8
6	35	8.8

Cuadro 6.4: Distribución de problemas de clasificación de acuerdo al número de variables independientes.

La implementación se hizo con código propio desarrollado en lenguaje *Matlab*. Los resultados se reportan en [83].

En los siguientes cuatro apartados se hace uso de estos conjuntos de datos en el entrenamiento y validación de problemas de clasificación binaria.

6.2.3. Determinación de cotas inferiores para el parámetro C usando APM

Una vez construidos los conjuntos de entrenamiento y prueba para los 400 problemas de clasificación binaria generados automáticamente, el siguiente paso consistió en la estimación de 400 valores (uno por cada problema) que pudieran ser usados como cotas inferiores para el parámetro C en MSVs.

Lo primero que se hizo en esta estimación consistió en la implementación de un KPC como parte de la metodología de APM[8]. Con esta implementación fue posible asegurar que se tuvieran estructuras funcionales idénticas entre esta metodología y una MSV entrenada con el mismo tipo de kernel. En la práctica, y como sucede cuando se entrenan MSVs, fue necesario especificar el grado del kernel polinomial a ser usado durante la fase de entrenamiento y prueba. Este punto es muy importante, ya que el número de monomios en la expansión polinomial resultante depende de su grado y, por lo tanto, repercute directamente en la determinación de la cota inferior para C, pues ésta es construida con los coeficientes correspondientes. De esta manera, la cota estimada podría ser usada en MSVs sólo con polinomios de ese mismo grado y, como se verá más adelante, de grados inferiores. Además, este hecho

[8]Esta implementación fue realizada con *software* desarrollado en lenguaje FoxPro por el Dr. Ángel Kuri Morales.

impone restricciones al conjunto de búsqueda que tendría el AG para la selección automática del grado del polinomio en MSV.

Modelo	Min	Q1	Mediana	Media	Q3	Max	DE
APM1	0.2593	0.4490	0.4921	0.4954	0.5418	0.7576	0.0741
APM2	0.2174	0.4583	0.5000	0.5007	0.5455	0.7143	0.0713
APM3	0.2174	0.4556	0.5000	0.4994	0.5429	0.7500	0.0741

Cuadro 6.5: Estadísticas descriptivas para los ajustes de diferentes grados de kernel polinomiales, usando el método de APM.

La decisión que se tomó en esta fase de experimentación fue la elección de tres grados de polinomio, a saber: 1, 2 y 3. Por lo tanto, se entrenaron 400 problemas con un polinomio lineal, 400 con uno cuadrático y 400 más con uno cúbico. Por su parte, esta selección significó la obtención de 3 conjuntos de 400 cotas inferiores de C, al seguir la metodología propuesta en el aparatado 4.2.2.

La precisión obtenida con cada grado de polinomio fue medida como el número de predicciones correctas efectuadas por el clasificador en el conjunto de prueba. En el Cuadro 6.5 se muestran las principales estadísticas para los ajustes en cada caso y se observa que la predicción echa por medio de estos polinomios es de prácticamente el 50 % en promedio en los tres casos. El valor máximo de clasificación se alcanza empleando polinomios de grado uno y tres, con valores alrededor del 75 %, mientras que el valor mínimo cercano al 21 % se obtiene con polinomios de grado dos y tres. Si bien la media de los estimadores es prácticamente la misma para los tres grados de polinomio, la distribución de los valores con ajustes iguales o superiores al 50 % se da en polinomios cuadráticos y cúbicos, con 48.5 y 47.5 %, respectivamente. La desviación estándar (DE) es similar en los tres casos y se ubica alrededor de 0.07. El rango inter-cuantil ($IQR = Q3 - Q1$) es cercano a 0.09 en los tres casos, lo que indica que la distribución está altamente concentrada o, en otras palabras, que hay una dispersión estadística baja. Enseguida se muestra gráficamente la distribución de los valores de clasificación obtenidos (Figura 6.2).

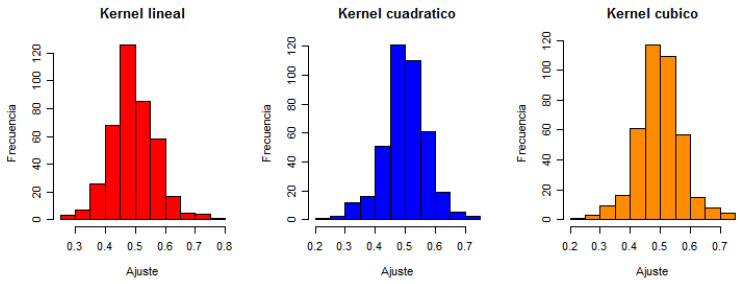

Figura 6.2: Histogramas de las predicciones obtenidas con APM.

Aún cuando el análisis descriptivo permite conocer la distribución de las variables de respuesta, no es posible establecer conclusiones con relación a que modelo resultó más conveniente en la clasificación de los conjuntos en cuestión. Para indagar con más detalle sobre este asunto, se efectúo otro tipo de análisis empleando

curvas ROC(*Receiver Operating Characteristic*) [27], así como una serie de pruebas de hipótesis estadísticas.

La comparación sobre que grado polinomial es más adecuado para un problema específico se podría efectuar empleando una curva ROC, construyendo la envolvente convexa correspondiente [27]. Sin embargo, este procedimiento resulta impráctico para estos experimentos ya que, de hacerlo, implicaría la construcción de 400 gráficas de este tipo (una por cada problema). En su lugar, se calcularon la tasa de positivos falsos (tpf) y la tasa de positivos verdaderos (tpv) [27], para todos los problemas involucrados en el análisis y se graficaron de acuerdo al grado polinomial empleado en cada caso. Los resultados se muestran en la Figura 6.3.

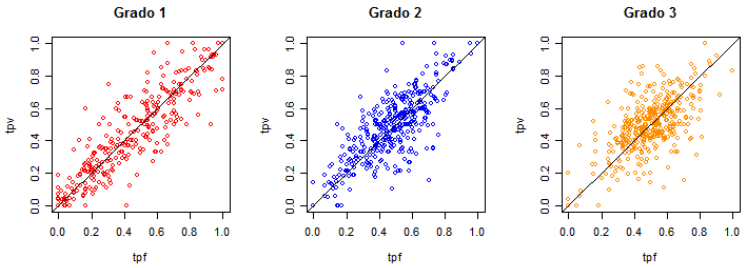

Figura 6.3: Sensitividad vs especificidad para los problemas de clasificación entrenados con diferentes grados de kernel polinomial.

La línea de 45^o de la Figura 6.3 indica el ajuste que se obtendría con un clasificador aleatorio, por lo que puntos que se encuentran por debajo de este nivel se consideran peores y los puntos por encima mejores a los obtenidos por este clasificador. El punto situado en el extremo superior izquierdo (0,1), representa el mejor de los mundos para una función de clasificación binaria. A partir de estos resultados se encontró que el 39.8 % de los problemas analizados con un kernel de grado 1 presentaron un poder de clasificación inferior al que se obtendría de la aplicación de un clasificador aleatorio, ya que ese es el porcentaje de puntos que se encuentran por debajo de la línea de 45 grados. Para el caso de polinomios de segundo y tercer grado, estos porcentajes son del 43.5 y 47 por ciento, respectivamente. La gráfica anterior muestra información relevante relacionada con el poder de clasificación de cada modelo, sin embargo, tampoco es posible concluir en favor de un grado en particular.

A efecto de tratar de validar si es que realmente existe alguna diferencia significativa entre los clasificadores obtenidos con distintos grados polinomiales, se procedió a efectuar pruebas de hipótesis estadísticas bajo la hipótesis nula de que no existe diferencia entre el valor promedio de los ajustes de dos modelos distintos, para lo cual se efectuaron dos tipos de prueba. En el primer caso se utilizó una PtP, considerando las combinaciones de los tres clasificadores utilizados ($p_1 vs p_2$, $p_1 vs p_3$ y p_2 *vs* p_3). En todas las pruebas se consideró un nivel de confianza del 95 %. La variable de prueba fue el ajuste obtenido por los clasificadores sobre conjuntos de prueba. Los resultados se resumen en la segunda y tercera columnas del Cuadro 6.6, donde se muestran los valores del estadístico t y los $valores - p$ correspondientes.

Como puede observarse, en ninguna de las tres pruebas se encuentra evidencia estadística suficiente en favor de alguno de los clasificadores, es decir, en ningún caso

Hip Nula	Hip Alt	t	valor-p	PP
$\bar{p}_1 - \bar{p}_2 = 0$	$\bar{p}_1 - \bar{p}_2 < 0$	-1.1487	0.1257	0.0965
$\bar{p}_1 - \bar{p}_3 = 0$	$\bar{p}_1 - \bar{p}_3 < 0$	-0.8241	0.2052	0.1753
$\bar{p}_2 - \bar{p}_3 = 0$	$\bar{p}_2 - \bar{p}_3 < 0$	0.2624	0.6034	0.62025

Cuadro 6.6: Resultados de la prueba *t-pareada* y de Permutación para la comparación de ajustes con diferentes grados de kernels polinomiales.

se rechaza la hipótesis nula de que la diferencia entre los ajustes de los clasificadores respectivos es igual a cero.

La prueba t se utilizó bajo la premisa de que los valores bajo prueba siguen una distribución normal, como se sugiere en la literatura [104] cuando se cuenta con un número suficiente de puntos, en general mayor de 30 puntos. Para estar seguros de esta situación se efectuaron dos tipos de pruebas de normalidad para los valores de ajuste: la construcción de gráficos Cuantil-Cuantil[9] (gráficos-CC) y la prueba Shapiro Wilks [104]. Las Figura 6.4 muestra los gráficos-CC correspondientes.

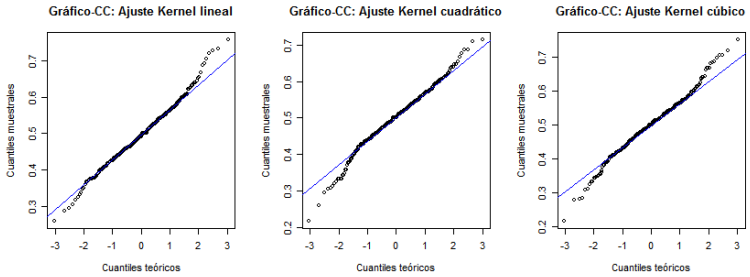

Figura 6.4: Gráficos-CC como prueba de normalidad para los ajustes con diferentes grados de kernels polinomiales.

La observación de las gráficas anteriores muestra que en las colas de la distribución hay bastantes puntos que se alejan de la línea de 45 grados, lo que hace dudar acerca de si realmente es posible suponer una distribución normal de los datos. Una prueba de normalidad adicional, la prueba *Shapiro-Wilks*, confirma que no es posible suponer una distribución normal (Cuadro 6.7), ya que los $valores - p$ obtenidos así lo indican.

Clasificador	Estadístico SW	valor-p
1	0.9907	0.0129500
2	0.9851	0.0003981
3	0.9851	0.0003981

Cuadro 6.7: Prueba *Shapiro-Wilks* para probar normalidad en los ajustes obtenidos con diferentes grados de kernel polinomiales.

Se utilizó la prueba de Permutación (PP) como alternativa para efectuar pruebas

[9] *QQ-plots.*

de hipótesis estadísticas, en donde no se considera ningún supuesto de distribución y con la cual también es posible efectuar pruebas pareadas. En esta prueba se emplean estadísticos t que son estimados a partir de permutaciones de los datos originales y en los que no se considera en absoluto la distribución de los mismos. Las hipótesis nulas y alternativas son las mismas a las empleadas para el caso del estadístico t y el número de permutaciones que se utilizó en cada caso fue de 3000. Este número se justifica observando como el $valor - p$ de la prueba se estabiliza conforme aumenta el número de iteraciones (Figura 6.5). De hecho, se nota que hubiera sido suficiente con menos iteraciones para obtener estimadores similares, pero como el tiempo de cómputo no es significativo para este tipo de pruebas en la actualidad[10] se consideran al menos este número de iteraciones a lo largo de esta Tesis. Una excelente exposición de este tema se puede encontrar en [24]. Los resultados de la aplicación de esta prueba se ilustran en la columna 5 del Cuadro 6.8.

Hip Nula	Hip Alt	t	valor-p	PP
$\bar{p}_1 - \bar{p}_2 = 0$	$\bar{p}_1 - \bar{p}_2 < 0$	-1.1487	0.1257	0.01000
$\bar{p}_1 - \bar{p}_3 = 0$	$\bar{p}_1 - \bar{p}_3 < 0$	-0.8241	0.2052	0.1913
$\bar{p}_2 - \bar{p}_3 = 0$	$\bar{p}_2 - \bar{p}_3 < 0$	0.2624	0.6034	0.6050

Cuadro 6.8: Resultados de la prueba *t-pareada* y de Permutación para la comparación de ajustes con diferentes grados de kernel polinomiales.

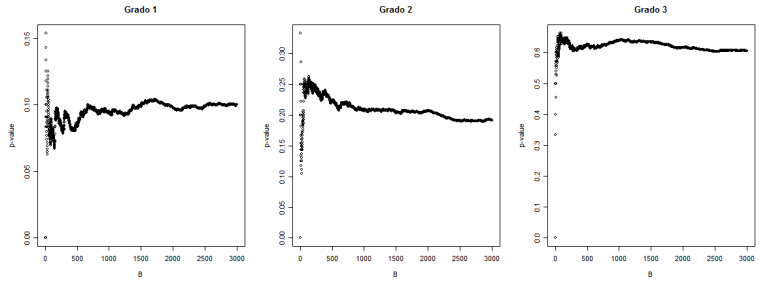

Figura 6.5: Estabilidad del *valor-p* en la Prueba de Permutación como criterio para evaluar el número de submuestreos (B) apropiados.

Los estadísticos obtenidos en la prueba de Permutación confirman los resultados previamente observados con la PtP en el sentido de que no es posible concluir en el 95 % de las ocasiones que el ajuste logrado con una función de clasificación sea superior a otra. Cabe señalar que a pesar de que las pruebas de normalidad fueron claramente rechazadas, la PtP resultó ser robusta en la comparación de los ajustes correspondientes obtenidos con cada kernel polinomial.

Finalmente, al contar con tres cotas distintas para cada conjunto de clasificación, se eligió la mayor de ellas en cada caso para ser empleado posteriormente en la definición de un rango para C en el entrenamiento de MSV con algoritmos genéticos y como una aproximación en el entrenamiento de MSV sin el empleo de AGs. Esta

[10]Dados los recursos de cómputo con los que se cuenta en nuestros días, el tiempo ya no es problema como lo era cuando se descubrieron este tipo de técnicas.

decisión estuvo basada en el hecho de que el AG explorará entre tres alternativas de grados polinomiales y, por ello, debe contar con un rango que resulte representativo para los tres casos. Los resultados se reportan en [83].

6.2.4. MSV

En este apartado se detalla la forma en que se entrenaron los mismos 400 casos de la sección anterior, pero empleando una MSV con una función de kernel polinomial. La idea de esta estrategia consiste en poder comparar los resultados derivados de ambos ejercicios.

El entrenamiento con la MSV se realizó utilizando 20 valores de C (10 por debajo del límite definido por el APM y 10 por encima, manteniendo una distancia constante entre ellos) para cada problema y para cada grado polinomial, una especie de validación cruzada, pero los ajustes se determinaron con base en el error sobre conjuntos de prueba. Los valores óptimos para C y el parámetro del kernel polinomial fueron aquellos para los cuales se obtuvo el menor error de prueba. De esta manera, los mejores ajustes fueron los correspondientes a estos parámetros y fueron utilizados para comparar esta metodología y la de APM. En este último caso, también se eligieron los mejores ajustes de entre todos los obtenidos con diferentes grados polinomiales. La gráfica 6.6(a) resume las estadísticas de ambos conjuntos óptimos y permite visualizar las diferencias obtenidas.

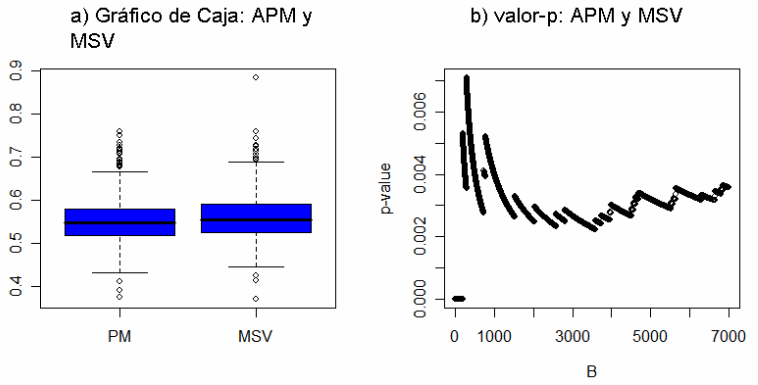

Figura 6.6: (a) Gráfico de caja correspondiente a los ajustes obtenidos con APM y MSVs, (b) Estabilidad del *valor-p* en la Prueba de Permutación como criterio para evaluar el número de submuestreos (B) apropiados.

A partir de ahora, el desempeño de los algoritmos de aprendizaje se medirá con relación al desempeño de la MSV para cada problema. Estas comparaciones *pareadas* son consistentes con las pruebas estadísticas *pareadas* usadas con los algoritmos de clasificación que se emplean aquí. La elección de la MSV se debe a su probadas características teóricas, su capacidad de resolver diversos tipos de problemas y su éxito en la solución de gran cantidad de problemas prácticos [6][116][18] [95] [9][20].

Por lo tanto, en la Figura 6.8 se muestran las estadísticas de los ajustes del método APM relativas a la MSV (con Gráficos de caja). Se puede apreciar que el ajuste relativo de APM es muy similar al obtenido con la MSV. Sin embargo, para

poder establecer conclusiones válidas se requiere efectuar las pruebas de hipótesis correspondientes.

Una prueba de hipótesis para estas diferencias se encuentra en la fila 1 del Cuadro 6.10, donde la PtP y la PP indican que es posible desechar la hipótesis nula de que no existe diferencia entre los ajustes promedio de ambos conjuntos. La evidencia estadística va en favor de la hipótesis alternativa, es decir, existe evidencia con un 95 % de confianza de que el ajuste con MSV es superior al de APM. Para la prueba de Permutación se ejecutaron B=7000 submuestreos (véase Figura 6.6(b)).

Con relación a los diferentes valores de C, se eligieron 3 de ellos de entre los 20 analizados previamente: el menor de ellos, el mayor y el estimado por el APM. Se efectuaron comparaciones entre los errores de prueba obtenidos, pero no se encontró evidencia estadística suficiente en favor de alguno en particular. El *software* empleado para los entrenamientos con la MSV fue *LIBSVM*[11] [12], usando la interfase para *Matlab*. Los resultados se reportan en [83]. Conviene adelantar que en los experimentos que se describen más adelante también se empleo este mismo *software* en los casos en los que se emplean MSVs.

6.2.5. Método Grid

El método Grid es una técnica comúnmente sugerida para la estimación de parámetros en MSVs. Ésta consiste en proponer una serie de pares ordenados de parámetros (en el caso en que se desean optimizar dos de ellos, por ejemplo C y kP) y seleccionando el par con el cual se obtiene el menor error, después de la aplicación de validación cruzada [49]. En los experimentos efectuados con esta técnica, se usaron dos tipos de funciones kernel: un KPC y un KBR.

Kernel polinomial

Los experimentos efectuados en esta fase fueron los mismos de la sección anterior. Es decir, se eligieron los mismos 20 valores de C para el entrenamiento de estos conjuntos, pero ahora el error de prueba se midió por medio de validación cruzada. De esta manera, los valores óptimos para este parámetro y para el grado polinomial se obtuvieron después de la aplicación de 10-FCV. En adelante, se usará la abreviación GRID-P para hacer referencia a este método.

El mejor ajuste para cada problema también fue elegido bajo este criterio, obteniéndose un conjunto con los mejores 400 ajustes obtenidos con este método. Los resultados se utilizaron para otras comparaciones. La primera de ellas se hizo contra el mejor ajuste de APM y la segunda en contra de la MSV. El criterio de comparación se repitió y los resultados de las pruebas de contraste estadístico se encuentran en las filas 2 y 3 del Cuadro 6.10, respectivamente. El ejercicio se implementó usando la interfase para *Matlab* de *LIBSVM* [12].

En primera instancia, en estas pruebas se observa que la aplicación de 10-FCV en lugar de mejorar el ajuste promedio de la MSV, lo empeora, ya que la comparación entre estas dos metodologías favorece estadísticamente a la MSV con un 95 % de confianza. Considerando la comparación entre el APM y GRID-P, se encontró que el ajuste promedio de ambas es comparable y no existe diferencia significativa en favor de alguno de ellos. En el gráfico de caja de la Figura 6.7(b) se aprecia gráficamente

[11] Cabe señalar que el mismo ejercicio se desarrollo usando el *software* [11] desarrollado en *Matlab* para plataforma LINUX, el cual se encuentra disponible en:
http://theoval.sys.uea.ac.uk/ gcc/svm/toolbox/.

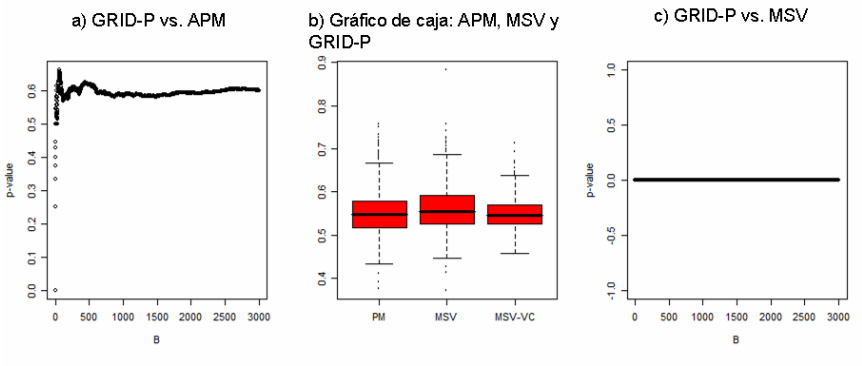

Figura 6.7: a) Estabilidad de la PP para la prueba entre APM y GRID-P, con B=3000 permutaciones, b) Distribución de los mejores ajustes para APM, MSV y GRID-P, c) y Estabilidad de la PP para la prueba entre la MSV y GRID-P, con B=3000 permutaciones.

la superioridad de la MSV sobre sus competidoras en términos de ajuste promedio. Sin embargo, también se aprecia en esa figura que el método GRID-P resultó ser más robusto, pues aunque su ajuste promedio es inferior, también su desviación estándar lo es (0.037), de hecho, es inferior a la que presentan la APM (0.058) y la MSV (0.057). Las pruebas de contraste estadístico nuevamente fueron la PtP y la PP, cuyos valores también se encuentran en las filas 2 y 3 del Cuadro 6.10. Por su parte, las Figuras 6.7(a) y 6.7(c) contienen la evolución del *valor-p* para la PP, donde es importante notar que para la prueba entre MSV *vs.* GRID-P, este valor permanece constante e igual a 0, lo que evidencia como la PtP nuevamente resulta ser bastante robusta. Estos resultados se reportan en [83].

Kernel de base radial

Los mismos experimentos de la sección anterior fueron implementados aquí, pero ahora se usó un KBR y una partición más fina para los valores de C a ser considerados (con 30 valores de C). De igual forma, se usó el mismo rango para el parámetro C obtenido con el kernel polinomial, debido a que no se cuenta con un criterio que permita definir de antemano ese rango cuando se emplean funciones de base radial. Por tanto, se eligieron los 30 valores de C, 15 de ellos considerados por debajo de la cota definida anteriormente para cada problema y 15 por encima de ella, manteniendo una distancia constante entre ellos. Para el caso del parámetro del KBR, se aplicó el mismo criterio, al tomar 30 valores distintos para este valor comenzando con 0.1 y con una separación constante de 0.1 para cada uno de ellos. Nuevamente, el error de prueba se midió usando 5-FCV. En esta Tesis, este método se abrevia por GRID-R.

Los resultados obtenidos fueron comparados con los arrojados por la MSV. No se encontró evidencia estadística suficiente (con base en las pruebas PtP y PP) para aceptar la hipótesis de que el ajuste promedio del método GRID-R sea superior al reportado con la MSV. Los valores estadísticos correspondientes se encuentran en la columna 4 del Cuadro 6.10 y los Gráficos de caja en la Figura 6.8.

6.2.6. Otros Clasificadores

En esta sección se muestran los resultados obtenidos al aplicar otros clasificadores binarios convencionales usando la muestra de 400 casos discutidos con anterioridad. El objetivo con estos clasificadores es hacer comparaciones con las alternativas propuestas anteriormente y con los métodos genéticos de la sección siguiente. En primera instancia, se consideró la aplicación del método de Análisis de Discriminante Lineal (ADL), que se describe en [41]. El error de estimación se midió en dos formas: a) aplicando 10-FCV y b) usando los mismos conjuntos de prueba previamente usados con los métodos APM y MSV. Dado que no se encontró diferencia estadística entre el error con CV y el obtenido con los conjuntos de prueba y, puesto que las desviaciones estándar fueron similares (0.1034 y 0.1070, respectivamente), se mantuvo la primera aproximación para futuras comparaciones. Una prueba estadística adicional (basada en los estadísticos PtP y PP) al comparar el ajuste derivado del método ADL con la MSV y GRID-R, reveló que estos últimos clasificadores resultaron significativamente (los estadísticos de prueba fueron 21.1899 y 24.7629, respectivamente y los $valores - p$ para la PP fueron igual a 0 en ambos casos).

Otro método usado en la comparación fue el clasificador del *vecino más cercano* (1-NN) [41], donde se utilizó 10-FCV para la medición del error de prueba. Las pruebas estadísticas confirman que 1-NN y ADL compiten de manera similar, dado que no se encontró evidencia suficiente en favor del ajuste promedio de alguno de ellos en comparación con el otro (los $valores - p$ para la PtP y la PP fueron 0.0849 y 0.0849, respectivamente).

Dados los pobres resultados observados con la aplicación de 1-NN y ADL, se decidió aplicar el método de *k vecinos más cercanos* (k-NN), donde k (número de vecinos más cercanos) se determina como aquel con el cuál se obtenga el menor error de CV, entre 15 valores probados (de 1 a 15). Para la medición de las distancias se usó la métrica euclidiana y el algoritmo se implementó en *Matlab*, donde la distancia de cada punto a las k observaciones más cercanas se calculó de manera exhaustiva. La asignación de los puntos a cada una de las clases se efectuó empleando el criterio de *voto-mayoritario* en las clases de los k vecinos más cercanos. Los resultados fueron significativamente mejores a los obtenidos con ADL y 1-NN, similares al método GRID-P, pero peores que la MSV y GRID-R, conforme a los resultados de las pruebas PtP y PP. Los valores de las pruebas estadísticas en la comparación con la MSV y GRID-R se encuentran en las filas 5 y 6 del Cuadro 6.10, respectivamente. Las principales estadísticas (representadas con Gráficos de caja) de los ajustes para APM, GRID-P, GRID-R, ADL, 1-NN y k-NN, relativos a la MSV, se muestran en la Figura 6.8 (representados con gráficos de caja). Todas las pruebas de Permutación implementadas resultaron estables con 3000 permutaciones.

6.2.7. MCG

Kernel polinomial

Los experimentos efectuados en esta fase consistieron principalmente en el entrenamiento de los conjuntos generados empleando la Máquina de Clasificación Genética, propuesta en el capítulo anterior, haciendo uso de un KPC. Por ello, las siglas MCG-P se emplean en adelante al hacer referencia a este clasificador. Las características del algoritmo genético ya fueron explicadas anteriormente, solo resta comentar acerca de la forma específica en que se determinaron ciertos parámetros

Variable	Parte Entera	Parte Fraccionaria	Rango
C	variable	20	$(0, 2 * cota]$
p	2	0	$[1, 3]$
P_c	0	20	[0,7, 1]
P_m	0	20	[0,001, 0,1]
P	0	20	$[0, 1]$

Cuadro 6.9: Parámetros empleados en el método de MCG-P.

a la hora del entrenamiento. Se ha mencionado antes que se utilizó una codificación binaria de punto fijo para representar las cadenas de caracteres que definen el genotipo. En el Cuadro 6.9 se muestra el número de bits que se emplearon para cada una de las variables representadas.

El número de bits empleados para la parte entera en la codificación del parámetro C estuvo en función del valor obtenido para la cota inferior con la metodología de APM. Es decir, como este parámetro teóricamente representa un número positivo, se determinó un número de bits suficiente para garantizar al menos una cota inferior que pudiera ser representada en la codificación que aquí se emplea. El número de bits empleado estuvo determinado por la cota obtenida con APM. Este punto es crucial, ya que en el pasado no se ha tomado en cuenta este detalle al emplear la metodología de AGs para la determinación de este tipo de parámetros, ya que usualmente se decide un número que a juicio del investigador es adecuado para el problema específico, lo cual requiere de mucha experiencia en el conocimiento de este problema o alguna otra guía que indique el rango de valores apropiado. La especificación del número de bits determinado por este rango tiene la ventaja adicional de reducir el tiempo de cómputo, ya que la cadena genética contiene sólo el número de bits requerido para la representación de sus parámetros. Más adelante se resalta la importancia de este punto tomando en cuenta los resultados obtenidos en estos experimentos.

En el caso del grado del polinomio, se usaron únicamente dos bits, ya que el rango de este parámetro fue especificado para polinomios de grado uno hasta tres, como se mencionó anteriormente. También puede observarse que en todas las variables donde tuviera sentido el uso de decimales, se utilizaron 20 bits en cada una de ellas para su especificación. Con relación a las probabilidades de cruzamiento y mutación, se utilizaron 20 bits para la parte decimal y ninguno para la parte entera, pero la representación real resultante fue escalada en el rango que se especifica en la última columna de la tabla, es decir, en el rango de [0.7,1] para P_c y de [0.005, 0.1] para P_m[12].

Otros dos aspectos a considerar fueron el tamaño de la población y el número de generaciones a emplear durante el entrenamiento. En el primer caso, se eligieron

[12]La expresión empleada para escalar estos valores fue la siguiente:

$$x_n^* = \frac{x_n - x_{min}}{x_{max} - x_{min}} (x_0 - x_u) + x_u \quad \forall n = 1, ..., N;$$

los términos x_{min} y x_{max} son los valores mínimo y máximo que la variable x puede alcanzar, respectivamente. Por su parte, el intervalo $[x_u, x_0]$ es el rango donde se desean escalar estas variables

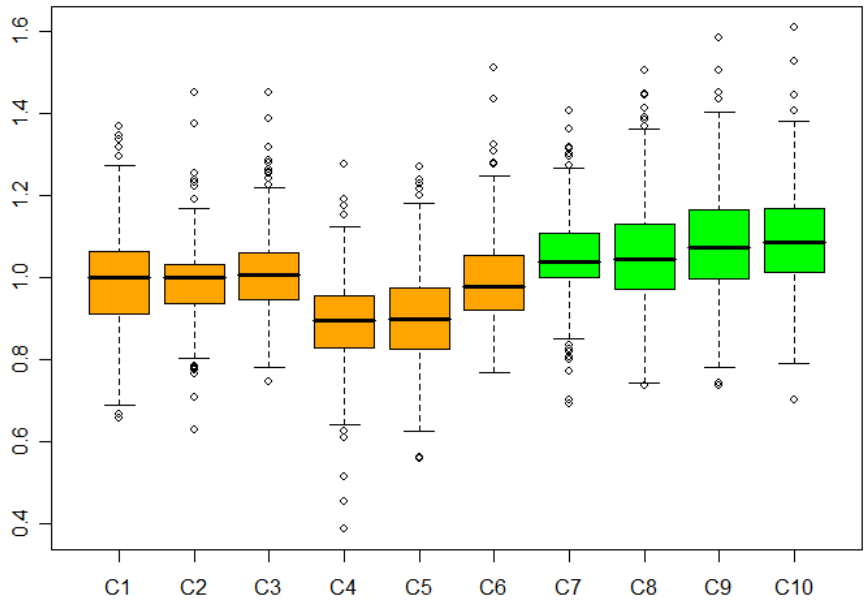

Figura 6.8: Gráficos de caja de los ajustes obtenidos por los algoritmos de aprendizaje relativos a la MSV: APM (C1), GRID-P (C2), GRID-R (C3), ADL (C4), 1-NN (C5), k-NN (C6), MCG-P (C7), MCG-PVC (C8), MCG-R (C9) y MCG-RVC (C10).

30 individuos en la población para todos los problemas entrenados, manteniendo este número para cada generación durante el entrenamiento, como lo sugiere la metodología de Vasconcelos. Con relación al número de generaciones, se tomó una submuestra de problemas que fueron entrenados con 50 y 150 generaciones. Al término de este entrenamiento se efectúo una comparación entre los resultados obtenidos con ambos procedimientos. La comparación se basó en una prueba pareada de Permutación, donde el estadístico resultante fue de 0.1465. Este resultado indica que no existe evidencia estadística suficiente, con un 95 % de confianza, para rechazar la hipótesis nula de que el ajuste promedio alcanzado con 150 generaciones supere al que se obtuvo con 50 generaciones. De esta manera, el resto de los problemas fue entrenado con 50 generaciones.

Una vez finalizado el entrenamiento, también se efectuaron pruebas estadísticas para comparar esta variante genética con los modelos previamente analizados. Los resultados (relativos a la MSV) para todos los clasificadores, representados por gráficos de caja, se presentan en la Figura 6.8.

En la Figura 6.8, se observa claramente que la mediana de los ajustes correspondientes al clasificador genético es superior a la obtenida por el resto de sus competidores. También puede notarse que la distribución de los datos parece simétrica

Cuadro 6.10: Pruebas de Hipótesis para la validación estadística de los algoritmos de aprendizaje: APM (C1), MSV (C2), GRID-P (C3), GRID-R (C4), k-NN (C5), MCG-P (C6), MCG-PVC (C7), MCG-R (C8) y MCG-RVC (C9).

Prueba	**Hipótesis Nula**	**Hipótesis Alternernativa**	***t***	**valor-p**	**PP**
1	$H_1 \equiv \bar{p}_{C2} - \bar{p}_{C1} = 0$	$H_1 > 0$	2.612	0.0047	0.003
2	$H_2 \equiv \bar{p}_{C1} - \bar{p}_{C3} = 0$	$H_2 > 0$	0.221	0.4127	0.601
3	$H_3 \equiv \bar{p}_{C2} - \bar{p}_{C3} = 0$	$H_3 > 0$	3.775	0.0001	0.000
4	$H_4 \equiv \bar{p}_{C4} - \bar{p}_{C2} = 0$	$H_4 > 0$	0.479	0.3159	0.684
5	$H_5 \equiv \bar{p}_{C2} - \bar{p}_{C5} = 0$	$H_5 > 0$	2.653	0.0042	0.012
6	$H_6 \equiv \bar{p}_{C4} - \bar{p}_{C5} = 0$	$H_6 > 0$	4.2602	0.0000	0.000
7	$H_7 \equiv \bar{p}_{C6} - \bar{p}_{C1} = 0$	$H_7 > 0$	8.979	0.0000	0.000
8	$H_8 \equiv \bar{p}_{C6} - \bar{p}_{C2} = 0$	$H_8 > 0$	10.927	0.0000	0.000
9	$H_9 \equiv \bar{p}_{C6} - \bar{p}_{C4} = 0$	$H_9 > 0$	8.4826	0.0000	0.000
10	$H_{10} \equiv \bar{p}_{C6} - \bar{p}_{C7} = 0$	$H_{10} > 0$	0.625	0.2661	0.283
11	$H_{11} \equiv \bar{p}_{C9} - \bar{p}_{C8} = 0$	$H_{11} > 0$	3.583	0.0002	0.001
12	$H_{12} \equiv \bar{p}_{C9} - \bar{p}_{C6} = 0$	$H_{12} > 0$	11.0399	0.0000	0.000

y no existe ninguna tendencia hacia los cuantiles superiores o inferiores en ningún caso.

Al efectuar pruebas estadísticas a la MCG-P contra los algoritmos previamente analizados, se observa que la evidencia es contundente en todos los casos en favor de ésta. Las pruebas estadísticas así lo comprueban, ya que en todos los casos los estadísticos de la prueba *t-pareada* resultaron claramente significativos, con lo que fue rechazada la hipótesis nula, en favor de la alternativa, de que el ajuste promedio de la MCG-P es superior. La prueba de Permutación claramente respalda este resultado. En las filas 7, 8 y 9 del Cuadro 6.10 se resumen las pruebas estadísticas y los estadísticos para las pruebas de contraste entre la MCG-P y los siguientes clasificadores: APM, GRID-P y GRID-R.

Un aspecto importante a considerar en la Figura 6.8 es que no sólo el ajuste relativo de la MCG es superior, sino también su variabilidad es una de las más bajas. Esto es, las desviaciones estándar fueron: 0.089, 0.093 y 0.094 para GRID-P, GRID-R y MCG-P, respectivamente.

Aunque la diferencia en las desviaciones estándar no es significativa, se decidió combinar los métodos GRID-P y MCG-P, aplicando *5-FCV* para medir el ajuste de la MCG (el clasificador resultante se denomina MCG-PVC) y verificar si la variabilidad de la MCG-P pudiera ser reducida aún más. La distribución de los ajustes también se muestra en la Figura 6.8, donde se puede apreciar que la distribución de MCG-PVC no mejoró en términos de variabilidad, aunque mantuvo el mismo nivel de ajuste promedio. Los resultados estadísticos confirmaron estas conclusiones, ya que la PtP no fue estadísticamente significativa, lo que significa que no fue posible rechazar la hipótesis nula. La PP también respalda esta conclusión. Las pruebas estadísticas se muestran en la fila 10 del Cuadro 6.10. Por lo tanto, la MCG-P resultó ser el mejor algoritmo en términos de ajuste y robustez. Esto significa que el tiempo adicional necesario para la implementación de Validación Cruzada se puede evitar sin sacrificar ajuste y obteniendo menor variabilidad. Es-

tas conclusiones se sustentan de manera estadística. Además, la variante genética tiene la ventaja adicional de atacar el problema de reducción de dimensionalidad sin sacrificar ajuste en el proceso.

Kernel de base radial

Los mismos experimentos de la sección anterior fueron implementados en esta sección, pero esta vez usando un KBR. Se decidió efectuar este análisis para verificar si existen diferencias significativas en los resultados encontrados con el clasificador genético al trabajar con otro tipo de función kernel. El rango elegido para la codificación del parámetro C fue el mismo que se usó con el método GRID-R. El parámetro del KBR también fue incluido en el genoma para su óptima selección (en lugar del grado polinomial, por supuesto), así que sólo se tuvieron que hacer cambios mínimos en el algoritmo para la implementación de este clasificador. La medición del error de entrenamiento también se efectuó de dos maneras distintas. En el primer caso, se usaron los conjuntos de prueba (este método fue nombrado como MCG-R), mientras que en el segundo caso se consideró el uso de 5-FCV para medir el error (MCG-RVC). Para la comparación entre las diferencias (si es el caso) de los ajustes entre estas dos metodologías se implementó una prueba de hipótesis, donde se encontró evidencia estadística suficiente a favor del ajuste promedio arrojado por MCG-RVC (fila 11 del Cuadro 6.10). Otro resultado relevante encontrado es que la desviación estándar de este último clasificador resultó inferior (0.112) a la obtenida por la MCG-R (0.131), pero más alta que la reportada previamente para la MCG-P (0.094). Contrario a los resultados reportados con el uso del kernel polinomial, en el caso del KBR no sólo se obtuvo el mejor ajuste, sino la menor variabilidad con el uso de CV como medida de error en el algoritmo. Con relación a la PP, ésta resultó estable con 3000 permutaciones. Finalmente, dado que del análisis previo se observó la superioridad de la MCG-P en comparación con las alternativas que habían sido analizadas antes, se decidió comparar su ajuste promedio con el obtenido con la MCG-RVC. Los resultados se muestran en la fila 12 del Cuadro 6.10, donde se puede apreciar la superioridad de la MCG-RVC, aunque su variabilidad también resulta más alta, como se mencionó antes.

Aún cuando se aprecian diferencias entre los clasificadores genéticos (dependiendo del tipo de función kernel empleada en el entrenamiento de la MSV) se ha demostrado en esta Tesis doctoral que éstos resultan claramente superiores (en verde en la Figura 6.8) en comparación con todas las metodologías no genéticas con las cuales fueron comparados. Todos estos resultados fueron reportados en [83]

Para la implementación de todos estos experimentos se usó *Matlab* y la interfase de *LIBSVM* para *Matlab* [12] para el entrenamiento de la MSV, como se mencionó previamente. Todas las pruebas estadísticas implementadas fueron desarrolladas con el paquete estadístico *R* [103].

La MCG también puede emplearse en la solución de problemas de regresión con MSV. En la siguiente y última sección de la Tesis se implementa este algoritmo para abordar este tipo de problemas y donde se comparan los resultados con otras estrategias para la determinación automática de parámetros en MSV. A diferencia del número de problemas y la validación estadística efectuada en este apartado, en el caso de regresión se tomaran algunos problemas que han servido como referencia en la literatura para la comprobación y comparación de nuevos algoritmos, ya que no se cuenta con una metodología que permita generar problemas de regresión no lineal multivariados de manera automática.

6.3. MRG: Evidencia empírica

Este apartado esta dedicado a la experimentación de la metodología denominada Máquina de Regresión Genética en la solución de problemas de aproximación no lineal de funciones, donde se determinan de manera automática los parámetros C, kP y ϵ. La eficiencia de este algoritmo también se mide en comparación con otras dos estrategias con las cuales es posible la selección automática de estos parámetros. La primera de ellas se debe a [14] y ya se ha expuesto su funcionamiento anteriormente. La segunda metodología fue la aplicación de 5-FCV para elegir cada parámetro de entre una lista de valores elegidos de manera aleatoria en un rango que fue definido de antemano (Método Grid). Los criterios de comparación fueron el ECM obtenido al efectuar el entrenamiento con *5-FCV* y la medición del tiempo de ejecución con un procesador Intel-Centrino de 1.8 GH y 512 MB de memoria. Los problemas empleados para la comparación fueron tomados de UCI-MLR y éstos fueron: mpg, mg y diabetes. Las características de mpg y diabetes ya fueron discutidas en secciones anteriores, pero en el caso de mg se tienen 1385 observaciones con 6 variables independientes.

Para cada conjunto de datos se efectuaron entrenamientos empleando funciones kernel polinomiales y funciones de base radial. Los parámetros en cada caso también fueron calibrados por el AG para el método MRG y empleando 5-FCV en los otros casos. Es importante señalar que el método de Cherkassky sólo es aplicable cuando se emplean funciones de base radial y la estimación del parámetro σ se hizo usando un polinomio de alto grado, empleando nuevamente la técnica de PGMs.

Los ECMs, los valores de los parámetros obtenidos y los tiempos de ejecución para cada algoritmo se muestran en el Cuadro 6.11.

De acuerdo a lo mostrado en la tabla anterior, puede apreciarse que la MRG resulta ser bastante competitiva en comparación con los otros algoritmos. Los ECMs fueron al menos tan buenos o mejores en todos los casos en comparación con el método de Cherkassky, aunque en tiempo de ejecución este último supera al primero. Una ventaja adicional de la MRG sobre el método de Cherkassky es que la primera no está limitada con el tipo de kernel empleado para su entrenamiento y, por lo tanto, resulta de aplicación más general. Por su parte, la aplicación del método Grid compite favorablemente con la MRG en el caso de mpg, pero con los otros problemas es superada por esta última, aunque no por un amplio margen. En términos de costo computacional, el desempeño del método Grid es superado claramente por la MRG en términos generales, donde esta diferencia se acentúa en conjuntos con mayor número de datos y, sobretodo, empleando funciones de kernel polinomiales. Por ejemplo, el tiempo que le toma al método Grid en el problema mpg es casi 8 veces mayor al que ocupa la MRG -con 30 individuos y 50 generaciones. En ambos casos, un elemento importante en ambas metodologías es que se debe elegir de antemano un rango de valores en donde puedan elegirse los parámetros correspondientes, donde la elección de los parámetros en el método Grid esta sujeto a la aleatoriedad, más que tratarse de un proceso de optimización plenamente justificado.

Por lo tanto, puede concluirse que la MRG resulta ser una estrategia de aprendizaje que compite de manera muy favorable, resultando superior en la mayoría de los casos tratados en estos experimentos, en comparación con técnicas que han sido muy socorridas en la literatura. Finalmente, como en el caso de clasificación, la MRG tiene la facultad adicional de elegir las variables independientes que realmente son significativas durante el entrenamiento sin sacrificar ajuste.

Problemas	Tiempo	ECM	C	ϵ	kP
mpg					
MRG (KBR)	6.22	6.59	15.8509	0.3067	0.53
MRG (KPC)	48.07	7.16	4.4020	0.4582	3.00
Grid (KBR)	46.64	6.81	13.3348	0.8437	0.50
Grid (KPC)	92.71	7.17	43.4900	0.84374	3.00
Cherkassky		7.17	46.8609	0.4726	0.50
diabetes					
MRG (KBR)	1.38	0.32	14.9694	0.4956	0.22
MRG (KPC)	1.32	0.31	1.2283	0.4917	2.00
Grid (KBR)	1.53	0.28	5.5644	0.6800	0.50
Grid (KPC)	0.87	0.28	5.5644	0.7000	2.00
Cherkassky		0.34	6.9082	0.5083	0.50
mg					
MRG (KBR)	28.23	0.01	2.4429	0.0927	3.66
MRG (KPC)	847.80	0.02	30.9100	0.1000	3.00
Grid (KBR)	120.02	0.01	0.0987	0.0621	1.10
Grid (KPC)	939.00	0.02	8.7700	0.0100	3.00
Cherkassky		0.02	1.6094	0.0485	0.80

Cuadro 6.11: Errores, tiempo (en minutos) y parámetros obtenidos al aplicar la metodología de MRG.

En futuras investigaciones convendría diseñar una estrategia de validación estadística, aplicada en un número de problemas significativo, probablemente generados de manera automática. Con ello, se podrían sacar conclusiones más generales en la aplicación de este algoritmo, como se hizo en el caso de clasificación.

El código generado con esta estrategia es muy similar al usado en la MCG. Las modificaciones correspondientes se desarrollaron en lenguaje *Matlab* y también se utilizó la interfase para *Matlab* de *LIBSVM*. Los resultados descritos en este apartado fueron reportados en [81].

Capítulo 7

Conclusiones

Esta investigación doctoral se ha enfocado principalmente al estudio de Máquinas de Soporte Vectorial, un método de aprendizaje que en los últimos tiempos ha despertado un gran interés por parte de la comunidad científica del área de aprendizaje de máquina y que ha sido aplicada con éxito en múltiples problemas prácticos. La MSV se distingue por ser un sistema para el eficiente entrenamiento de máquinas de aprendizaje lineal en espacios inducidos por funciones kernel, respetando los principios de la teoría de regularización y aprovechando algunos resultados importantes de la teoría de optimización.

El énfasis principal en este estudio está relacionado con el problema de la selección óptima de parámetros libres en MSVs, un elemento muy importante para la eficiente aplicación de esta metodología. Uno de ellos es el parámetro de regularización C, que se encarga de controlar la capacidad de clasificación y, al mismo tiempo, la complejidad en el diseño de la máquina de aprendizaje. Otros parámetros involucrados en su diseño son: el(los) parámetro(s) de la función kernel y el parámetro ϵ en el problema de regresión no lineal.

Después de analizar el estado del arte en la solución de este importante problema, se concluyó que los Algoritmos Genéticos representan una alternativa atractiva como mecanismo de optimización en la selección de parámetros. Las razones principales por las cuales se opta por este mecanismo de optimización se deben a su flexibilidad y probada efectividad en la solución de problemas de optimización. También se mencionó la importancia de elegir un tipo de AG no convencional, dadas las limitaciones del AGS exhibidas en el pasado. Además, se resaltó la importancia de involucrar una estrategia autoadaptable para evitar sesgo en la implementación de los operadores genéticos, particularmente el mecanismo de cruzamiento y mutación. El tipo de algoritmo que reunía estas características es el Algoritmo Genético de Vasconcelos, modificado con una estrategia autoadaptable.

En una primera propuesta, se utilizó el AGV sin la variante autoadaptable para el diseño de una MSV, la MSVG. Esta propuesta consistió en dejar al AG la responsabilidad del proceso completo de optimización de la MSV en problemas de clasificación de patrones. Es decir, no sólo los parámetros libres de la máquina se codifican y optimizan a partir del uso de un AG, sino también los multiplicadores de Lagrange en la formulación dual de la MSV. Esta propuesta fue probada empíricamente con resultados satisfactorios en problemas de clasificación binaria y múltiple, empleando algunos conjuntos de datos ampliamente usados en la literatura para estos propósitos. Un inconveniente detectado con esta propuesta es el almacenamiento

y tratamiento de la matriz de kernel en problemas a gran escala. Esto se debe a que el número de elementos de la matriz crece de manera cuadrática conforme lo hace el tamaño de muestra, lo que la hace inmanejable para conjuntos con miles de observaciones. Una alternativa consistiría en computar los elementos de la matriz cada vez que sean requeridos en el proceso de entrenamiento; sin embargo, esa posibilidad fue descartada porque también resulta extremadamente costosa.

Una propuesta más eficiente se basó en la determinación evolutiva de los parámetros libres, dejando el trabajo de la optimización a un algoritmo diseñado explícitamente para acelerar el entrenamiento de la MSV, como es el caso del algoritmo de Optimización Secuencial Mínima. De esta manera, se diseñó la MCG, un método autoadaptable que además de abordar el problema de la selección óptima de parámetros, tiene la capacidad de seleccionar las variables más relevantes del conjunto de entrenamiento, haciendo aún más eficiente el proceso de aprendizaje y, lo más importante, sin sacrificar ajuste.

Un elemento importante a considerar en el diseño de la MCG es la cuestión del rango de definición del parámetro C, donde éste se encuentra altamente influenciado por el tipo de función kernel, en la medida en que este tipo de funciones proyectan el espacio de solución de la MSV a un espacio de características. Esto es, conforme cambia la dimensión de este espacio, el rango posible de valores para este parámetro también se modifica.

Una propuesta para tratar de definir este rango se derivó al comprobar la equivalencia algebraica entre funciones de clasificación polinomiales derivadas de los métodos de Aproximación Polinomial Minimax y MSVs. Para ello, un aporte adicional de esta Tesis se dio al confirmar la posibilidad de usar funciones kernel polinomiales, que tradicionalmente se emplean en MSVs o en otros métodos de kernel, pero no en APM. Dadas las similitudes encontradas, se determinó una forma algebraica de acotar inferiormente el valor de C, empleando los coeficientes de una función de decisión entrenada con el método de APM. Aunque se reconoce que la diferencia en las estrategias de entrenamiento usadas en ambas metodologías podría afectar en la calidad de la cota obtenida, se considera que esta aproximación puede resultar en una guía que ayude a definir un rango apropiado para C en aplicaciones prácticas. Asimismo, se abre un espacio de investigación para explorar más a fondo este tema, no sólo con el empleo de estructuras polinomiales, sino con otras formas funcionales, aprovechando la gran cantidad de posibilidades que pueden derivarse con el uso de funciones kernel.

Para la comprobación experimental de la MCG se buscaba un método que resultara estadísticamente confiable, trabajando con problemas de múltiple dominio y evitando la práctica común de usar conjuntos de problemas generados artificialmente y que, a juicio de algún(os) investigador(es), son considerados como complicados y representativos para la comparación de algoritmos de aprendizaje. En una situación ideal, sería posible obtener conclusiones de carácter general, al efectuar comparaciones con otros métodos sobre conjuntos de prueba de múltiples dominios y representativos de un vasto universo de posibilidades. Tomando en cuenta lo anterior, se diseñó una estrategia con las siguientes características:

1. Se construyeron conjuntos de problemas multivariados de clasificación binaria en forma automática, empleando funciones de Walsh y Rademacher,

2. El número de casos fue definido con base en un criterio, usado comúnmente en la Teoría de Muestreo Estadístico, para la elección de tamaños de muestra en poblaciones infinitas,

3. Se establecen criterios de validación basados en pruebas de hipótesis estadísticas, donde se comparan los promedios, sobre muestras pareadas, de los ajustes obtenidos con dos algoritmos de aprendizaje,

4. Las pruebas estadísticas implementadas son de dos tipos: 1) paramétricas, usando la prueba *t-pareada* y, 2) remuestreo, empleando la Prueba de Permutación. En la primera de ellas debe suponerse que las muestras son extraídas de poblaciones normales, mientras que en la segunda no se requieren suposiciones de distribución.

Con base en lo anterior, se reconoce la enorme complejidad de la comparación de modelos de clasificación sobre la base de problemas con múltiples dominios y, por ello, no es posible argumentar que el método propuesto ofrecerá conclusiones de validez universal, tampoco que el conjunto de casos, que (teóricamente) representa a un número infinito de problemas de este tipo, resulta ser necesariamente la mejor elección posible. Aún así, en este Tesis doctoral se decidió trabajar en el caso de múltiples dominios, ya que se buscaba en forma objetiva probar la robustez del modelo genético de clasificación al evaluar su eficiencia y eficacia sobre la base de conjuntos de problemas que fueran grandes, estadísticamente acotados y con el menor sesgo posible en su construcción; en lugar de probarlos sobre un simple y limitado dominio, como es generalmente el caso. Esta propuesta comienza con el reconocimiento de que, para establecer condiciones estadísticas de validación, generalmente se requiere información suficiente. Desafortunadamente, contar con un número abundante de problemas de clasificación con el que puedan hacerse ejercicios de este tipo, representa un reto importante. Por lo tanto, el método propuesto captura dos cuestiones directamente relacionadas para verificar la validez de la MCG, consistentes en la generación automática y no sesgada de problemas de clasificación, así como su comprobación estadística.

Empíricamente, la aplicación de este método permitió verificar, contundentemente, la superioridad en el desempeño de la MCG en comparación con otras estrategias de aprendizaje supervisado no genéticas. Las pruebas estadísticas así lo revelaron en todos los casos y, en algunos de ellos, con márgenes ampliamente significativos.

También se efectuaron pruebas para hacer comparaciones entre las estrategias genéticas al usar dos tipos de funciones kernel distintas durante su entrenamiento; a saber, un kernel polinomial y uno de base radial. En estos últimos experimentos, para la medición del tipo de error se tomaron en cuenta dos criterios: a) considerando subconjuntos de prueba generados aleatoriamente y extraídos de los conjuntos de casos construidos en forma automática (los clasificadores genéticos entrenados con este criterio fueron etiquetados como: MCG-P y MCG-R, para distinguirlos por el tipo de kernel usado, ya se kernel polinomial o de base radial, respectivamente) y, b) empleando validación cruzada (del tipo k-FCV) (bajo la misma lógica usada en a), los clasificadores genéticos fueron etiquetados como: MCG-PVC y MCG-RVC).

Los resultados estadísticos fueron mixtos. Por un lado, se encontró que la MCG-P superaba a su contraparte MCG-PVC en todos los sentidos considerados, ya que su ajuste promedio fue superior y su varianza fue más pequeña, donde cabe señalar que este clasificador fue uno de los que menor variabilidad presentó entre todos los clasificadores considerados (incluidos los no genéticos). Por otra parte, la MCG-RVC superó al clasificador MCG-R también en todos los sentidos (ajuste y varianza). Finalmente, la MCG-P logró superar a todos los clasificadores con los

que había sido comparada previamente, pero no logro hacerlo al ser comparada con la MCG-RVC, pues esta última la superó claramente en ajuste, aunque no en variabilidad. En resumen, el efecto de la validación cruzada en los clasificadores genéticos es incierto y depende del tipo de kernel empleado; pues cuando se usan funciones kernel polinomiales, no sólo empeora el ajuste sino también la variabilidad, mientras que usando funciones kernel de base radial sucede lo contrario.

En síntesis, existen varios factores por los cuales la MCG resulta sumamente recomendable:

1. La selección de parámetros se efectúa de manera automática y eficiente.
2. El ajuste promedio resulta claramente superior en comparación con otros clasificadores, lo que fue comprobado en forma estadística.
3. Además del buen ajuste conseguido con esta estrategia, el AG también es capaz de reducir el número de variables independientes usadas dentro del entrenamiento, sin sacrificar ajuste.
4. La variante autoadaptable del AGV resultó ser muy efectiva en la solución del problema y, además, ayuda a evitar sesgo en la selección de las probabilidades de cruzamiento y mutación durante su implementación.

Una variante de la MCG fue desarrollada para abordar el problema de regresión no lineal, que se denominó Máquina de Regresión Genética (MRG). Esta propuesta también permite la optimización de parámetros de la MSV, la selección de variables independientes y es autoadaptable. La única variación es la incorporación del parámetro ϵ para su selección óptima. Por ahora, no fue posible efectuar un análisis estadístico como el desarrollado para la MCG, ya que no se cuenta con una metodología para la generación automática y no sesgada de problemas de regresión no lineal. En respuesta, se efectuaron comparaciones con otros métodos de selección de parámetros para MSVs en problemas de regresión sobre la base de un conjunto de casos tomados de repositorios de datos conocidos. Los resultados obtenidos prueban la competitividad de la MRG en comparación de sus competidores. En unos casos, este método resultó más eficiente en términos de costo computacional y ajuste, mientras que en otros lo fue en términos de flexibilidad en su aplicación. También se deriva de este análisis una vía para futura investigación en la que pueda desarrollarse una metodología de validación estadística para problemas de regresión no lineal.

Adicionalmente a los métodos genéticos desarrollados en esta Tesis, otra aportación, basada en un análisis más detallado del uso de funciones polinomiales en MSVs y APM, derivó en el diseño de un nuevo tipo de kernel polinomial, que fue nombrado como Kernel Polinomial MP. El empleo de este tipo de funciones permite encontrar expresiones explícitas, donde puedan ser analizadas las relaciones entre las variables independientes y sus respectivas potencias, algo que en problemas prácticos suele ser un aspecto importante. En ese sentido, debe señalarse que este nuevo kernel considera todas las combinaciones de variables independientes para los puntos en el conjunto de entrenamiento y sus potencias respectivas. Por ello, resulta evidente que esta función proyecta a espacios de alta complejidad aún en problemas con pocas variables independientes. La implementación de este kernel en una MSV y su aplicación en algunos problemas de clasificación y regresión (también tomados de repositorios de datos conocidos), permitió concluir que es posible obtener soluciones competitivas a las que se obtienen con funciones kernel

tradicionales. No obstante, se sugiere como tema de futuro investigación la valoración sobre la conveniencia de su aplicación en problemas de alta dimensionalidad, así como la viabilidad de su uso práctico con otros métodos de kernel.

Finalmente, algunas propuestas adicionales para futura investigación derivadas de este trabajo de Tesis incluyen: a) El desarrollo de métodos de solución en tiempo real de problemas de clasificación y aproximación de funciones con MSVs, donde se incorpore la selección automática de parámetros libres y, b) La combinación eficiente y óptima de funciones kernel que pueda usarse en el entrenamiento de métodos de kernel, incluida la MSV, que posiblemente contribuya a resolver el problema de la elección de una función kernel para cada aplicación y la selección respectiva de su(s) parámetro(s).

Nomenclatura

Adaline	Red Neuronal Lineal Adaptiva (*Adaptive Linear Element*)
1-NN	Clasificador del vecino más cercano (*nearest neighbor classifier*)
ADL	Análisis de Discriminante Lineal
AG	Algoritmo Genético
AGI	Algoritmo Genético Idealizado
AGS	Algoritmo Genético Simple
AGV	Algoritmo Genético de Vasconcelos
APM	Aproximación Polinomial Minimax
ART	Teoría de Resonancia Adaptiva (*Adaptive Resonance Theory*)
C_f	Contorno de una función f
CI_f	Contorno Inferior de una función f
CMA-ES	Estrategia Evolutiva con Matriz de Covarianzas Adaptable (*Covariance Matrix Adaptation Evolution Strategy*)
CS_f	Contorno Superior de una función f
CV	Validación Cruzada (*Cross Validation*)
Dimensión VC	Dimensión de Vapnik-Chervonenkis
ECM	Error Cuadrático Medio
EE	Estrategia Evolutiva
FBR	Función de Base Radial
FDA	Discriminante Lineal de Fisher (*Fisher Discriminant Analysis*)
FPϵI	Función de Pérdida ϵ-Insensitiva
GK SVM	Máquina de Soporte Vectorial Genética para la elección de la función Kernel *Genetic Kernel Support vector Machine*
HBC	Hipótesis de los Bloques Constructores
k-FCV	Validación Cruzada con k dobleces (*k-fold Cross Validation*)
k-NN	Clasificador de k vecinos más cercanos (*k-nearest neighbor classifier*)
KBR	Kernel de Base Radial
Kernel FDA	Discriminante de Fisher no Lineal (*Kernel Fisher Discriminant Analysis*)

Kernel PCA	Análisis de Componentes Principales no Lineal (*Kernel Principal Component Analysis*)
KKT	Condiciones de *Karush-Kunh-Tucker*
KL	Kernel Lineal
KP	Kernel de Perceptrones
KP-MP	Kernel Polinomial MP
KPC	Kernel Polinomial Clásico
LIBSVM	Librería para Máquinas de Soporte Vectorial (*Library for Support Vector Machines*)
LoO-CV	Valiadción Cruzada dejando fuera una observación (*Leave-one-Out Cross Validation*)
Madaline	Red Neuronal Lineal Adaptiva Múltiple (*Multiple Adaptive Linear Element*)
MCG	Máquina de Clasificación Genética
MCG-P	Máquina de Clasificación Genética Polinomial
MCG-PVC	Máquina de Clasificación Genética Polinomial con Validación Cruzada
MCG-R	Máquina de Clasificación Genética de base Radial
MCG-RVC	Máquina de Clasificación Genética de base Radial con Validación Cruzada
ML	Multiplicadores de Lagrange
MRG	Máquina de Regresión Genética
MSV	Máquina de Soporte Vectorial
MSVG	Máquina de Soporte Vectorial Genética
MSVI	Máquina de Soporte Vectorial Incremental
MSVP	Máquina de Soporte Vectorial Proximal
NSA	Nivel de Significancia Alcanzado
OSM	Optimización Secuencial Mínima
P_c	Probabilidad de cruzamiento
P_m	Probabilidad de mutación
p.d.	Positiva Definida
p.s.d.	Positiva Semidefinida
n.d.	Negativa Definida
n.s.d.	Negativa Semidefinida
PCA	Análisis de Componentes Pincipales (*Principal Component Analysis*)
PG	Programación Genética
PGM	Polinomios Genéticos Multivariados
PMRE	Principio de Minimización de Riesgo Estructural
PMREm	Principio de Minimización de Riesgo Empírico
PP	Prueba de Permutación
PPC	Problema de Porgramación Cuadrática
PPL	Problema de Porgramación Lineal
PPnL	Problema de Programación no Lineal
RPM	Red de Perceptrones Multicapa
PtP	Prueba t Pareada
RL	Regresión Logística
Statlib	Repositorio de Datos para Algoritmos de Aprendizaje (*StatLib-Dataset Archive*)
TEA	Teoría Estadística del Aprendizaje
UCI-MLR	Repositorio de Datos para Algortimos de Aprendizaje (*University of California Irvine Machine Learning Repository*)
ROC	Curvas ROC (*Receiver Operating Characteristic*)

Bibliografía

[1] Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. A learning algorithm for Boltzmann Machines. *Cognitive Science 9* (1985), 147–169.

[2] Allwein, E. L., Schapire, R. E., and Singer, Y. Reducing multiclass to binary: A unifying approach for margin classifiers. In *Proc. 17th International Conf. on Machine Learning* (2000), Morgan Kaufmann, San Francisco, CA, pp. 9–16.

[3] Baudat, G., and Anouar, F. Generalized discriminant analysis using a kernel approach. *Neural Computation 12*, 10 (2000), 2385–2404.

[4] Bishop, C. *Pattern Recognition and Machine Learning*. Springer, 2006.

[5] Blanz, V., Schölkopf, B., Bülthoff, H. H., Burges, C., Vapnik, V., and Vetter, T. Comparison of view-based object recognition algorithms using realistic 3d models. In *ICANN* (1996), pp. 251–256.

[6] Boser, B. E., Guyon, I., and Vapnik, V. A training algorithm for optimal margin classifiers. In *COLT* (1992), pp. 144–152.

[7] Broomhead, D. S., and Lowe, D. Multivariable functional interpolation and adaptive networks. *Complex Systems 2* (1988), 321–355.

[8] Bryson, A. E., and Ho, Y. C. *Applied optimal control*. Blaisdell, New York, 1969.

[9] Burges, C. J. C. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery 2*, 2 (1998), 121–167.

[10] Burges, C. J. C., and Schölkopf, B. Improving the accuracy and speed of support vector machines. In *NIPS* (1996), pp. 375–381.

[11] Cawley, G. C. MATLAB support vector machine toolbox (v0.55β) [`http://theoval.sys.uea.ac.uk/~gcc/svm/toolbox`]. University of East Anglia, School of Information Systems, Norwich, Norfolk, U.K. NR4 7TJ, 2000.

[12] Chang, C., and Lin, C. LIBSVM: a library for support vector machines, 2001.

[13] Chapelle, O., Vapnik, V., Bousquet, O., and Mukherjee, S. Choosing multiple parameters for support vector machines. *Machine Learning 46*, 1/3 (2002), 131.

[14] Cherkassky, V., and Ma, Y. Practical selection of SVM parameters and noise estimation for SVM regression. *Neural Networks 17*, 1 (Jan. 2004), 113–126.

[15] Chih, W., Gwo, T., Yeong, G., and Wen, F. A real-valued genetic algorithm to optimize the parameters of support vector machine for predicting bankruptcy. *Expert Systems with Applications In press* (2006).

[16] Choi, M.-D. Tricks or treats with the Hilbert matrix. *American Mathematical Monthly 90* (1983), 301–312.

[17] Clearwater, S. H., and Hogg, T. Problem structure heuristics and scaling behavior for genetic algorithms. *AIJ: Artificial Intelligence 81* (1996).

[18] Cortes, C., and Vapnik, V. Support-vector networks. *Machine Learning 20* (1995), 273.

[19] Cover, T. M. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Trans. on Electronic Computers EC-14* (1965), 326–334.

[20] Cristianini, N., and Shawe-Taylor, J. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge University Press, Cambridge, U.K., 2000.

[21] Dietterich, T. G. Approximate statistical test for comparing supervised classification learning algorithms. *Neural Computation 10*, 7 (1998), 1895–1923.

[22] Dietterich, T. G., and Bakiri, G. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research 2* (1995), 263–286.

[23] Drucker, H., Burges, C. J. C., Kaufman, L., Smola, A. J., and Vapnik, V. Support vector regression machines. In *NIPS* (1996), M. Mozer, M. I. Jordan, and T. Petsche, Eds., MIT Press, pp. 155–161.

[24] Efron, B., and Tibshirani, R. *An Introduction to the Bootstrap*. Chapman and Hall, London, 1993.

[25] Escobar, D. E. *Economía Matemática*. Alfaomega, 2001.

[26] Ferguson, J. Multivariable curve interpolation. *Journal of the ACM 11*, 2 (Apr. 1964), 221–228.

[27] Flach, P. A. The many faces of ROC analysis in machine learning. University of Bristol, UK, 2004.

[28] Forrest, S., and Mitchell, M. Relative building-block fitness and the building-block hypothesis. In *Foundations of Genetic Algorithms 2*, L. D. Whitley, Ed. Morgan Kaufmann, San Mateo, CA, 1993, pp. 109–126.

[29] Fröhlich, H. Feature Selection for Support Vector Machines by Means of Genetic Algorithms. Master's thesis, University of Marburg, 2002.

[30] Friedrichs, F., and Igel, C. Evolutionary tuning of multiple SVM parameters. *Neurocomputing 64* (2005), 107–117.

[31] Fung, G., and Mangasarian, O. L. Proximal support vector machine classifiers. In *KDD* (2001), pp. 77–86.

[32] Fung, G., and Mangasarian, O. L. Incremental support vector machine classification. In *SDM* (2002), R. L. Grossman, J. Han, V. Kumar, H. Mannila, and R. Motwani, Eds., SIAM.

[33] Fung, G., and Mangasarian, O. L. Multicategory proximal support vector machine classifiers. *Machine Learning 59*, 1-2 (2005), 77–97.

[34] Galaviz, J., and Kuri, A. A self-adaptive genetic algorithm for function optimization. In *ISAI/IFIPS* (1996), p. 156.

[35] Gold, C., Holub, A., and Sollich, P. Bayesian approach to feature selection and parameter tuning for support vector machine classifiers. *Neural Networks 18*, 5-6 (2005), 693–701.

[36] Goldberg, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley, 1989.

[37] Grossberg, S. How does the brain build a cognitive code? *Psychological Review 87* (1980), 1–51.

[38] Gunn, S. Support vector machines for classification and regression. Tech. rep., Apr. 07 1998.

[39] Haaser, N. B. *Análisis Matemático 2. Curso Intermedio*. Trillas, 1992.

[40] Hastie, T., Rosset, S., Tibshirani, R., and Zhu, J. The entire regularization path for the support vector machine. *Journal of Machine Learning Research 5* (2004), 1391–1415.

[41] Hastie, T., Tibshirani, R., and Friedman, J. H. *The Elements of Statistical Learning*. Springer, July 2001.

[42] Haweel, T., El-Bakry, M., and El-Metwally, K. Hadron-hadron interactions a high energy via Rademacher funtions. *Chaos Solitions and Fractals 18* (2003), 159–168.

[43] Haykin, S. *Neural networks: A comprehensive foundation*. MacMillan, New York, 1994.

[44] Hebb, D. *Organization of behavior*. Science Edition, 1961.

[45] Holland, J. H. *Adaptation in natural artificial systems*. University of Michigan Press, Ann Arbor, 1975.

[46] Hopfield, J. J. Neural networks and physical systems with emergent collective computational abilities. In *Proceedings of the Nat. Acad. of Sciences 79* (1982), pp. 2554–2558.

[47] Horn, R. A., and Johnson, C. *Topics in Matrix Analysis*. Cambridge University Press, Cambridge, 1991.

[48] Howley, T., and Madden, M. G. The genetic kernel support vector machine: description and evaluation. *Artificial Intelligence Review 24(3-4)* (2005), 379–395.

[49] Hsu, C.-W., Chang, C.-C., and Lin, C.-J. A practical guide to support vector classification.

[50] Hsu, C.-W., and Lin, C.-J. A simple decomposition method for support vector machines. *Machine Learning 46*, 1/3 (2002), 291.

[51] Huang, C.-L., and Wang, C.-J. A GA-based feature selection and parameters optimization for support vector machines. *Expert Syst. Appl. 31*, 2 (2006), 231–240.

[52] Igel, C., and Kreutz, M. Operator adaptation in structure optimization of neural networks. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)* (San Francisco, California, USA, 7-11 2001), L. Spector, E. D. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, and E. Burke, Eds., Morgan Kaufmann, p. 1094.

[53] Jaakkola, T., and Haussler, D. Probabilistic kernel regression models. In *Probabilistic kernel regression models. In Proceedings of the 1999 Conference on AI and Statistics. 1999.* (7-11 1999), T. S. Jaakkola and D. H. (eds), Eds., Morgan Kaufmann.

[54] Jack, L. B., and Nandi, A. K. Support vector machine for detection and characterization of rolling element bearing faults. *Journal of Mechanical Engineering Science 215* (2000), 1065–1074.

[55] Joachims, T. Text categorization with support vector machines: Learning with many relevant features. Tech. Rep. LS VIII-Report, Universität Dortmund, Dortmund, Germany, 1997.

[56] Jong, K. D. An analysis of the behavior of a class of genetic adaptive systems. *Doctoral dissertation, University of Michigan* (1975).

[57] Jordan, M. I. Graphical models. *Statistical Science (Special Issue on Bayesian Statistics) 19*, 1 (2004), 140–155.

[58] Karush, W. Minima of functions of several variables with inequalities as side constraints. Master's thesis, Dept. of Mathematics, Univ. of Chicago, 1939.

[59] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. Optimization by simulated annealing. *Science 220* (1983), 671–680.

[60] Kohavi, R. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI* (1995), pp. 1137–1145.

[61] KUHN, H. W., and TUCKER, A. W. Nonlinear programming. *Econometrica 19* (1951).

[62] Kuri, A. *A Comprehensive Approach to Genetic Algorithms in Optimization, and Learning. Theory and Applications, Vol. 1: Foundations.* IPN, 1999.

[63] Kuri, A., and Galaviz, J. *Algoritmos Genéticos*. IPN-UNAM-FCE, México, 2002.

[64] Kuri-Morales, A. Heuristic techniques for the generalization and acceleration of an algorithm for multivariate minimax approximation. *Agrociencia 2*, 4 (1993), 89–127.

[65] Kuri-Morales, A. Approximation and classfication with genetic multivariate polynomias. *WSEAS Transactions on Computers 4*, 3 (2005), 645–652.

[66] Kuri-Morales, A., and Mejía-Guevara, I. Determination of the regularization parameter for support vector machines via Vasconcelos Genetic Algorithm. In *Proc. of WSEAS 5th Int. Conf. on Soft Computing, Optimization, Simulation and Manufacturing Systems, WSEAS Press* (2005), A. Kuri-Morales and N. Mastorakis, Eds., pp. 86–91.

[67] Kuri-Morales, A., and Mejía-Guevara, I. Evolutionary training of SVM for classification problems with self-adaptive parameters. *Research on Computer Science: Advances in Artificial Intelligence Theory 16* (2005).

[68] Kuri-Morales, A., and Mejía-Guevara, I. Genetic algorithms and the determination of the regularization parameter for support vector machines via Vasconcelos GA. *WSEAS Transactions on Circuits and Systems 4*, 4 (2005).

[69] Kuri-Morales, A., and Mejía-Guevara, I. Analysis of algebraic expressions derived from genetic multivariate polynomials and support vector machines: A case study. In *CIARP-06, LNCS, Springer* (2006), J. F. Martínez-Trinidad, J. A. Carrasco-Ochoa, and J. Kittler, Eds., pp. 974–984.

[70] Kuri-Morales, A., and Mejía-Guevara, I. Evolutionary training of SVM for multiple category classification problems with self-adaptive parameters. In *IBERAMIA/SBIA'06, LNAI, Springer* (2006), J. Simão-Sichman, H. Coelho, and S. Oliveira-Rezende, Eds., pp. 329–338.

[71] Kuri-Morales, A. F. A methodology for the statistical characterization of genetic algorithms. *Lecture Notes in Computer Science 2313* (2002), 79–88.

[72] Lang, S. *Introducción al Análisis Matemático*. Adison-Wesley Iberoamericana, 1990.

[73] Larcher, G. W., Schmid, C., and Wolf, R. Quasi-monte carlo methods for the numerical integration of multivariate walsh series. *Mathl. Comput. Modelling 23*, 8/9 (1996), 55–67.

[74] LaValle, S. M., Branicky, M. S., and Lindemann, S. R. On the relationship between classical Grid search and probabilistic roadmaps. *I. J. Robotic Res. 23*, 7-8 (2004), 673–692.

[75] LeCun, Y. Une procedure dápprentissage pour reseau a seuil assymetrique. *Proc. Cognitiva 85* (1985), 599–604.

[76] Lee, S.-Y., Lee, S.-T., and Chen, D.-Y. Automatic video summary and description. In *Visual Information and Information Systems* (2000), pp. 37–48.

[77] Lee, Y., Lin, Y., and Wahba, G. Multicategory support vector machines. In *Proceedings of the 33rd Symposium on the Interface* (2001).

[78] Leter, J. *Applied Linear Statistical Models, 4th edition.* Trillas, 1996.

[79] Lohr, S. L. *Muestreo: diseño y análisis.* Thompson, México, 2000.

[80] McCulloch, W. S., and Pitts, W. A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics 5* (1943), 115–133.

[81] Mejía-Guevara, I., and Kuri-Morales, Á. Evolutionary feature and parameter selection in support vector regression. In *MICAI-07, LNAI, Springer* (2007), pp. 399–408.

[82] Mejía-Guevara, I., and Kuri-Morales, Á. MP-Polynomial Kernel for training support vector machines. In *CIARP-07, LNCS, Springer* (2007), pp. 584–593.

[83] Mejía-Guevara, I., and Kuri-Morales, A. Genetic support vector classification and minimax polynomial approximation. In *Submitted (Oct 2007)* (http://www.geocities.com/gauss75/ivan.html).

[84] Mercer, J. Functions of positive and negative type, and their connection with the theory of integral equations. *Transactions of the London Philosophical Society (A) 209* (1909), 415–446.

[85] Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E. Equations of state calculations by fast computing machines. *Journal of Chemical Physics 21 or 6* (1953), 1087–1091.

[86] Michalewicz, Z., and Schoenauer, M. Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary Computation 4*, 1 (1996), 1–32.

[87] Mika, S., Rätsch, G., Weston, J., Schölkopf, B., and Müller, K. Fisher discriminant analysis with kernels. In *Neural Networks for Signal Processing IX* (1999), Y.-H. Hu, J. Larsen, E. Wilson, and S. Douglas, Eds., IEEE, pp. 41–48.

[88] Min, J. H., and Lee, Y. Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters. *Expert Syst. Appl. 28*, 4 (2005), 603–614.

[89] Min, S.-H., Lee, J., and Han, I. Hybrid genetic algorithms and support vector machines for bankruptcy prediction. *Expert Syst. Appl. 31*, 3 (2006), 652–660.

[90] Minsky, M., and Papert, S. *Perceptrons.* MIT Press, Cambridge, MA, 1969.

[91] Mitchell, M. *An Introduction to Genetic Algorithms.* The MIT Press, 1996.

[92] Mitchell, M., Holland, J. H., and Forrest, S. When will a genetic algorithm outperform hill climbing. In *Advances in Neural Information Processing Systems* (1994), J. D. Cowan, G. Tesauro, and J. Alspector, Eds., vol. 6, Morgan Kaufmann Publishers, Inc., pp. 51–58.

[93] Mitchell, T. *Machine Learning*. McGraw-Hill, 1997.

[94] Mukherjee, S., Osuna, E., and Girosi, F. Nonlinear prediction of chaotic time series using a support vector machine. In *IEEE Neural Network for Signal Processing (NNSP'97)* (Amelia Island, FL, USA, 1997).

[95] Müller, K.-R., Smola, A. J., Rätsch, G., Schölkopf, B., K., J., and Vapnik, V. Predicting time series with support vector machines. In *ICANN* (1997), pp. 999–1004.

[96] n. Redy, J., and Rasmussen, M. L. *Análisis matemático Avanzado. Con aplicaciones a ingeniería y ciencias*. Limusa, Valencia, Spain, 1989.

[97] Osuna, E., and Girosi, F. Reducing the run-time complexity of support vector machines, 1998.

[98] Osuna, E., and Girosi, F. F. An improved learning algorithm for support vector machines. In *Neural Networks for Signal Processing VII, IEEE Workshop* (Amelia Island, FL., 1997).

[99] Pai, P.-F., and Hong, W.-C. Software reliability forecasting by support vector machines with simulated annealing algorithms. *Journal of Systems and Software 79*, 6 (2006), 747–755.

[100] Parker, D. B. Learning-logic: Casting the cortex of the human brain in silicon. Tech. Rep. TR-47, Center for Computational Research on Economics and Management Science, MIT, Cambridge, MA, 1985.

[101] Platt, J. Sequential minimal optimization: A fast algorithm for training support vector machines. Tech. Rep. MSR-TR-98-14, Microsoft Research (MSR), Apr. 1998.

[102] Platt, J. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods — Support Vector Learning* (Cambridge, MA, 1999), B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds., MIT Press, pp. 185–208.

[103] R Development Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2005. ISBN 3-900051-07-0.

[104] Rice, J. A. *Mathematical Statistics and Data Analysis*. Duxbury Press, 1995. Good intermediary statistics book.

[105] Rosenblatt, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review 65* (1958), 386–408.

[106] Roth, V., and Steinhage, V. Nonlinear discriminant analysis using kernel functions. In *NIPS* (1999), S. A. Solla, T. K. Leen, and K. R. Müller, Eds., The MIT Press, pp. 568–574.

[107] Rudolph, G. Convergence analysis of canonical genetic algorithms. *IEEE Transactions on Neural Networks 5*, 1 (1994), 96–101.

[108] Rumelhart, D., Hinton, G., and Williams, R. Learning representation of back-propagation errors. *Nature (London) 323* (1986), 533–536.

[109] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. Learning internal representations by error propagation. In *Parallel Distributed Processing – Explorations in the Microstructure of Cognition*. MIT Press, 1986, ch. 8, pp. 318–362.

[110] Rumelhart, D. E., and McClelland, J. L. *Explorations in Parallel Distributed Processing*. The MIT Press, Cambridge, MA, 1988.

[111] Runarsson, T. P. Asynchronous parallel evolutionary model selection for support vector machines. *Neural Information Processing - Letters and Reviews 3(3)* (2004), 59–67.

[112] Runarsson, T. P., and Yao, X. Stochastic ranking for constrained evolutionary optimization. *IEEE Transactions on Evolutionary Computation 4*, 3 (2000), 284–294.

[113] Russell, S., and Norvig, P. *Artificial Intelligence: A Modern Approach, 2nd Ed.* Prentice-Hall, NJ, p.p. 16, 1995.

[114] Samanta, B., Al-Balushi, K. R., and Al-Araimi, S. A. Artificial neural networks and genetic algorithm for bearing fault detection. *Soft Comput. 10*, 3 (2006), 264–271.

[115] Schmidt, M. Identifying speakers with support vector networks, 1996.

[116] Schölkopf, B., Burges, C., and Vapnik, V. Extracting support data for a given task. In *KDD* (1995), pp. 252–257.

[117] Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., and Williamson, R. C. Estimating the support of a high-dimensional distribution. *Neural Computation 13*, 7 (2001), 1443–1471.

[118] Schölkopf, B., Smola, and Müller, K. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation 10(5)* (1998), 1299–1319.

[119] Schölkopf, B., Smola, and Müller, K. *Learning with kernels*. MIT Press, 2002.

[120] Setiono, R., Leow, W. K., and Thong, J. Y. L. Opening the neural network black box: an algorithm for extracting rules from function approximating artificial neural networks. In *ICIS '00: Proceedings of the twenty first international conference on Information systems* (Atlanta, GA, USA, 2000), Association for Information Systems, pp. 176–186.

[121] Shawe-Taylor, J., and Cristianini, N. *Kernel Methods for Pattern Analysis*. CUP, June 2004.

[122] Smola, A., and Schölkopf, B. A tutorial on support vector regression. *Statistics and Computing* (2001, Forthcoming).

[123] Smola, A., Schölkopf, B., Williamson, R., and Bartlett, P. New support vector algorithms. *Neural Computation 12*, 5 (2000), 1207–1245.

[124] Srinivas, N., and Deb, K. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation 2*, 3 (1995), 221–248.

[125] Taha, H. A. *Investigación de Operaciones, quinta edición*. Alfaomega, 1994.

[126] Tax, D. M. J., and Duin, R. P. W. Data domain description using support vectors. In *ESANN* (1999), pp. 251–256.

[127] Vapnik, V. *Estimation of Dependencies Based on Empirical Data*. Springer-Verlag, Berlin, 1982.

[128] Vapnik, V. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[129] Vapnik, V. *Statistical Learning Theory*. Wiley, 1998.

[130] Vapnik, V., and Chervonenkins, A. A note on a class of perceptrons. automation and remote control. *Remote Control 25* (1964), 103–109.

[131] Vapnik, V., and Chervonenkis, A. J. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications 16*, 2 (1971), 264–280.

[132] Varian, H. R. *Microeconomic Analysis*. W. W. Norton & Company, February 1992.

[133] von der Malsburg, C. Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik 14* (1973), 85–100.

[134] Werbos, P. J. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Harvard University, 1974.

[135] Weston, J., Gammerman, A., Stitson, M., Vapnik, V., Vovk, V., and Watkins, C. Density estimation using support vector machines, 1997.

[136] Widrow, B. Generalization and information storage in networks of Adaline 'neurons'. In *Self-Organizing Systems 1962* (Chicago, Illinois, 1962), M. C. Yovits, G. T. Jacobi, and G. D. Goldstein, Eds., Spartan, pp. 435–461.

[137] Widrow, B., and Hoff, M. E. Adaptive switching circuits. In *Proceedings WESCON* (1960), pp. 96–104.

[138] Wolpert, D. H., and Macready, W. G. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation 1*, 1 (April 1997), 67–82.

[139] Zhang, Z., and Jordan, M. I. Bayesian multicategory support vector machines. In *In Uncertainty in Artificial Intelligence (UAI), Proceedings of the Twenty-Second Conference.* (2006).