UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE INFORMÁTICA

"MÁSTER EN INVESTIGACIÓN EN INFORMÁTICA"

Aplicación de Predictores Conformales a Señales de Fusión

Trabajo Fin de Máster en Ingeniería Informática para la Industria

Autor: Norma Verónica Ramírez Pérez

Directores: Matilde Santos Peñas
Gonzalo Pajares Martinsanz

Convocatoria : Junio 2011
Calificación obtenida: 8.0 (Notable)

Curso 2010-2011

AUTORIZACIÓN DE DIFUSIÓN

*La abajo firmante, matriculada en el Máster en Investigación en Informática de la Facultad de Informática, autoriza a la Universidad Complutense de Madrid (UCM) a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a su autor el presente Trabajo Fin de Máster: "**APLICACIÓN DE PREDICTORES CONFORMALES A SEÑALES DE FUSION**", realizado durante el curso académico 2010-2011 bajo la dirección de Matilde Santos Peñas y Gonzalo Pajares Martinsanz en el Departamento de Ingeniería del Software e Inteligencia Artificial, y a la Biblioteca de la UCM a depositarlo en el Archivo Institucional E-Prints Complutense con el objeto de incrementar la difusión, uso e impacto del trabajo en Internet y garantizar su preservación y acceso a largo plazo.*

JULIO 2011

NORMA VERÓNICA RAMÍREZ PÉREZ

RESUMEN

En problemas de clasificación, reviste especial importancia el análisis de la fidelidad del clasificador de señales, sobre todo cuando se presentan dominios donde es fácil que haya errores en la clasificación. La capacidad de evaluar la calidad de los resultados es posible a través del uso de metodologías flexibles como la técnica de Predictores Conformales (CP), capaces de dar predicciones sobre la confianza y credibilidad asignada a cada resultado de una verificación. Los predictores conformales pueden acotar la tasa de error de las predicciones de una región mediante el establecimiento de un nivel de confianza. Uno de los objetivos planteados en el presente trabajo ha sido la realización de una herramienta computacional para clasificación utilizando el algoritmo de CP, el cual se ha aplicado a un problema multiclase de señales de fusión; permite además determinar las medidas de no conformidad utilizando algoritmos subyacentes de clasificación como: Máquinas de Soporte Vectorial (Support Vector Machine, SVM), Vecinos Cercanos (Nearest Neighbors) y medidas de no conformidad propias del mismo algoritmo CP. Se han comparado los resultados obtenidos por estos tres métodos de clasificación para el mismo conjunto de señales.

Palabras clave: Predictores Conformales, Máquinas de Soporte Vectorial SVM, Medidas de Confianza, Credibilidad, Vecinos Cercanos .

ABSTRACT

In classification problems, particularly important to analyze the fidelity of the signal classifier, especially when there are domains where it is easy to have errors in classification. The ability to assess the quality of results is possible through the use of flexible methodologies such as the Conformal Prediction (CP) technique, capable to give predictions on confidence and credibility assigned to each outcome of a verification. Conformal predictors can limit the rate of error of predictions of a region by establishing a level of confidence. One of the goals outlined in this paper has been to conduct a computational tool for classification using the CP algorithm, which has been applied to a multiclass problem of fusion signals, can also to determine the measures of non-compliance using underlying classification algorithms such as Support Vector Machine (SVM), Nearest Neighbors (NN) and measures of their own nonconformity of the CP algorithm. We compared the results obtained by these three methods of classification for the same set of signals.

Keywords: Conformal Prediction CP, Support Vector Machine SVM, Measures of confidence, Credibility, Nearest Neighbors.

AGRADECIMIENTOS

Quiero expresar un profundo agradecimiento a mis asesores: Matilde Santos Peñas y Gonzalo Pajares Martinsanz, por su valioso apoyo en la culminación de este proyecto.

A mi familia, particularmente a mis padres, a mis hijos y muy especial a mi esposo, quien ha estado a mi lado en este proceso.

Al CIEMAT de Madrid, España, particularmente a Jesús A. Sánchez Vega, por su incondicional apoyo.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1

1 INTRODUCCIÓN

Hoy día la llamada “era de la información”, se ha caracterizado por generación expansiva y acumulativa de datos en todo tipo de disciplinas. Una proporción importante de esta información se almacena en bases de datos automatizadas, las cuales permiten tener un fácil acceso mediante las herramientas informáticas existentes. Es por ello que la revolución digital ha hecho posible la simplificación de esta tarea de alto costo computacional inicial, y que la captura y almacenamiento tenga ahora un coste muy bajo. Enormes cantidades de información generadas y almacenadas continuamente en bases de datos para su posterior uso y análisis, evidencian la obsolescencia de los métodos habituales, sobre todo cuando se manejan grandes conjuntos de datos.

Actualmente se han desarrollado nuevos métodos semiautomáticos de análisis que han resultado ser necesarios para entender grandes cantidades de datos. Aunque se han tomado medidas para resolver este tipo de problemas, continúa la búsqueda para encontrar técnicas supervisadas y no supervisadas cada vez más versátiles, robustas y eficientes que puedan hacer frente a los cambios continuos. De ahí que la clasificación de datos se ha descrito como una técnica de aprendizaje automático, y un modelo de clasificación de datos también automático que puede servir como una herramienta para distinguir entre objetos de diferentes clases. Este proceso de clasificación incluye una fase de entrenamiento, con un conjunto de datos en la fase inicial, para predecir qué parámetros deberán ser ponderados y combinados, de tal manera que puedan ser separados en varias clases de objetos. El aprendizaje en esta etapa pretende descubrir una selección óptima a partir del mencionado conjunto de datos, cuya etiqueta de clase es conocida. A continuación sigue la

validación, los pesos determinados en la fase de entrenamiento son aplicados a un nuevo conjunto de datos (llamado conjunto de validación), cuyas etiquetas de clase se desconocen, con el objetivo de determinar a qué clase pertenecen. Existen algunos métodos de clasificación que involucran un enfoque heurístico que pretenden encontrar la solución “óptima” mediante procesos de optimización.

En la literatura, existen varias técnicas de clasificación convencionales como: Vecinos Cercanos (Nearest Neighbors, K-NN) [20], clasificadores Bayesianos [15], Redes Neuronales[33], Árboles de Decisión [25] y Máquinas de Soporte Vectorial (Support Vector Machine, SVM) [8]. Actualmente las redes neuronales son las más ampliamente usadas, sin embargo, al realizar una clasificación con éstas, es preciso tener en cuenta muchos factores debido a que son generalmente más sensibles al ruido a la hora de ser entrenadas y con un alto coste computacional. Los árboles de decisión, también son muy utilizados en problemas de clasificación, siendo usualmente más veloces que las redes neuronales en la fase de entrenamiento, aunque no presentan flexibilidad al modelar los parámetros [35]. Un simple clasificador puede estar implementado mediante la técnica de Vecinos Cercanos [18], la cual tiene la ventaja de que es sencilla de implementar, pero suele ser bastante lenta si el conjunto de datos de entrada es muy grande.

De acuerdo con el estudio de estos métodos, las SVM constituyen una de las técnicas reconocidas que se utilizan para optimizar la solución esperada [8], habiendo mostrado su eficiencia en relación a otros métodos de aprendizaje supervisado [2, 30, 34, 37, 38]. Se ha probado que las SVM están muy bien fundamentadas teóricamente y poseen, una buena capacidad de generalización y habiendo llegado a ser en las últimas décadas, uno de los métodos de clasificación más utilizados. Al emplear las SVM, los límites de decisión son determinados directamente a partir de los datos de entrenamiento, de tal manera que el margen entre los límites de decisión es maximizado en un espacio de alta dimensión denominado espacio de características. La implementación de esta estrategia minimiza los errores de clasificación de los datos de entrenamiento, obteniendo así una mejor generalización, excepto

cuando el número de datos es pequeño, pesa el que la generalización de la clasificación de las SVM y otras técnicas difieren significativamente.

Las SVM, consideradas como una poderosa técnica empleada en la clasificación de datos y análisis de regresión, poseen además una notable ventaja con respecto a otros métodos, que radica en el hecho de que éstas obtienen un subconjunto de vectores soporte durante la fase de aprendizaje, el cual es sólo una pequeña parte del conjunto de datos originales y son los únicos atributos necesarios para la clasificación. Para encontrar un hiperplano de separación entre clases, las SVM necesitan resolver un problema de programación cuadrática (Quadratic Programming, QP), mediante una matriz de densidad N x N. Debido a que la mayoría de las rutinas de QP poseen complejidad cuadrática, las SVM pueden requerir en algún caso, bastante tiempo computacional y memoria para grandes bases de datos [29,44], por lo que el nivel de dificultad de entrenamiento de las SVM, es dependiente del tamaño del conjunto de datos.

En el análisis de clases, no sólo necesitamos realizar la clasificación, sino también saber si las instancias están bien o mal clasificadas y para ello utilizamos la metodología de Predictor Conformal (Conformal Prediction, CP) [41], el cual se utiliza para determinar la fiabilidad de una clasificación, esto es, conocer si las instancias están o no correctamente clasificadas. Es través de una medida de no conformidad, determinada por medio de CP, como podemos evaluar la confiabilidad y credibilidad de las clases que deseamos clasificar. Las medidas de no conformidad las podemos aplicar a cualquier clasificador como SVM, Vecinos Cercanos y en general cualquier otro.

1.1 Descripción del problema

Frecuentemente se tiene la necesidad de clasificar datos, imágenes, señales, etc., que nos ayuden a tomar decisiones, es por eso que en este proyecto se ha aplicado un método que permite no sólo clasificar, sino además saber qué tan bien están clasificados dichos datos. Se pretende pues, contar con una herramienta que facilite automatizar la clasificación y la calidad de la misma. Para realizar la aplicación, se tuvo acceso a la base de datos del

CIEMAT (Centro de investigaciones Energéticas, Medioambientales y Tecnológicas) de Madrid, España, de donde se obtuvieron datos de señales extraídas de un dispositivo de fusión de tipo Stellarator Heliac Flexible TJ-II [10], con las cuales se llevó a cabo el análisis computacional. Las señales procedentes de los sistemas de diagnóstico se digitalizaron y almacenaron en bases de datos accesibles para su manejo con los algoritmos aquí implementados. Es importante mencionar que cada descarga del sistema produce cientos de señales dinámicas, en las que se utilizan frecuencias de muestreo distintas en función del diagnóstico, que suelen ser desde algunas muestras por segundo, hasta un millón de ellas. Para poder realizar una clasificación, es necesario desarrollar un preprocesamiento y la implementación de mecanismos automáticos de clasificación siendo la principal aportación de este trabajo de investigación.

1.2 Estructura de la memoria

Este documento está organizado en 4 capítulos. En el capítulo inicial se discuten las generalidades del trabajo y se exponen brevemente los aspectos más importantes. En el capítulo 2 se presenta la fundamentación teórica de la metodología que se propone como herramienta de solución al problema de la clasificación, se muestran las características más importantes de las máquinas de soporte vectorial para clasificación, además de la Predicción Conformal y algunos otros métodos de clasificación que servirán como referencia de comparación. En el capítulo 3 se presenta, de manera detallada, la metodología escogida para la clasificación de señales, iniciándose con el pre-procesamiento de los datos, y detallando los resultados obtenidos al aplicar la metodología. En el capítulo 4 se presentan las conclusiones y los trabajos futuros que devienen del desarrollo de esta metodología. Finalmente, al término de este trabajo, se presentan las referencias bibliográficas y de consulta utilizadas para su realización.

1.3 Objetivos

El objetivo esencial de este trabajo se centra en la investigación y aplicación de predictores conformales a señales de fusión, que se comparan con algoritmos existentes de clasificación: K-NN y SVM. Todo ello se desarrolla bajo el paraguas de una herramienta que lo implementa. El planteamiento de esta estrategia se puede generalizar para otros tipos de señales. De igual manera, se hizo el planteamiento de los siguientes objetivos específicos:

- Abordar un problema multiclase con señales reales
- Realizar una herramienta en Matlab, con librerías de SVM de Spider, para aplicar predictores conformales a señales de fusión
- Preprocesamiento de señales unidimensionales con transformada de wavelets
- Obtener medidas de no conformidad con el método de Predictor Conformal, utilizando algoritmos subyacentes K-NN, SVM y “AVERAGE”
- Comparar los distintos métodos de clasificación utilizados en términos de confianza y credibilidad

CAPÍTULO 2

2 ESTADO DEL ARTE

2.1 Introducción

La importancia de la posesión de información y la posterior adquisición de un conocimiento adecuado, nos han abierto caminos para resolver situaciones que hoy en día han jugado un papel importante en la historia del ser humano y es aún de mayor importancia el manejo de la cantidad de información disponible haciendo complicado su manejo y su tratamiento.

La introducción de la informática y las nuevas tecnologías han conseguido no sólo que se puedan obtener y almacenar grandes cantidades de datos, sino que además nos preguntemos si es posible su uso inteligente de éstos. De este modo, se intentan desarrollar métodos de análisis de manera que los ordenadores sean capaces de aprender sobre estos datos, mejorar la eficiencia de sus aplicaciones y obtener algún conocimiento sin apenas intervención humana. Como consecuencia de estas necesidades surgen nuevas áreas de investigación para el manejo de la información, que mencionaremos en este apartado.

Como se ha mencionado previamente, uno de los objetivos principales de este capítulo es el de presentar el estado del arte del área de Predictores Conformales, como marco general de las técnicas de predicción y medidas de confianza con algoritmos de clasificación, los cuales serán motivo de comparación en el presente proyecto. La estructura del capítulo, la dividimos en dos partes, en la primera se describirán los conceptos principales de las máquinas de aprendizaje y la clasificación, y en la segunda se profundizarán los conceptos de Predictores Conformales (Conformal Prediction, CP).

2.2 Máquinas de aprendizaje y métodos de clasificación

La capacidad de que un sistema informático aprenda de una manera automática, es fundamental en la Inteligencia Artificial, que para que un sistema pueda ser considerado inteligente debe ser al menos capaz de aprender automáticamente [20]. El inicio de aprendizaje automático se remonta a la década de los 50´s y su objetivo es el desarrollo de métodos computacionales para la implementación de diversas formas de aprendizaje.

De manera general, se puede decir que la tarea de aprendizaje automático puede describirse como un proceso automático que permite mejorar la solución que se le da a los problemas.

La tarea de aprendizaje automático se puede describir como un proceso que permite mejorar la solución de un problema a medida que se procesan los datos y el sistema aprende en consecuencia.

Una de las categorías en las que se encuadran los métodos de aprendizaje la constituyen los métodos inductivos, aquéllos en los que un conjunto de ejemplos positivos y negativos del concepto objetivo, construyen una descripción del mismo de la cual se pueden derivar todos los ejemplos positivos y ninguno de los negativos. Los métodos inductivos son los de nuestro interés en este trabajo.

Un método de aprendizaje automático puede clasificarse como supervisado, no supervisado y criticado según la información que contenga los ejemplos de aprendizaje que se le proporcionen al sistema.

- Método supervisado: es el método que aprovecha la supervisión de un algoritmo “maestro” durante el aprendizaje que informa sobre la relación entre el ejemplo a aprender y el concepto objetivo.
- Método no supervisado: no obtiene ningún tipo de ayuda externa para realizar su labor de aprendizaje. En este caso no hay ni crítico ni “maestro”.

- Método criticado: recibe una crítica sobre el acierto de sus decisiones en la que se basa para mejorar su comportamiento. También se le conoce como método de aprendizaje por refuerzo.

En lo que respecta a métodos de clasificación, existen varios cuya tarea principal es la de encuadrar algún objeto dentro de las categorías denominadas clases.

En la clasificación supervisada se tiene un conjunto de datos de prueba consistente en medidas sobre un conjunto de variables, de forma que asocia a cada dato una etiqueta que define a qué clase del objeto pertenece. Por otro lado, en la clasificación no supervisada, los datos no son etiquetados y únicamente se desea encontrar grupos en los datos que se distingan unos de otros a partir de sus características.

Diversos métodos de clasificación, como las redes neuronales y árboles de decisión, forman parte del aprendizaje supervisado. A la salida, estos clasificadores estiman o aprenden una función de decisión que está asociada a nuevos datos, o sea a una etiqueta de clase dentro de las clases proporcionadas.

A continuación se describen en detalle los tipos de clasificación utilizados para este trabajo.

2.3 Método de clasificación de Vecinos Cercanos K-NN

El clasificador conocido como K Vecinos Cercanos (K-Nearest Neighbors, K-NN) [1,4,9,14,16,43], tiene como fundamento básico que cuando un nuevo caso se va a clasificar, lo haga en la clase más frecuente a la que pertenecen sus K vecinos más cercanos. El paradigma se fundamenta en una idea simple e intuitiva, lo que unido a su fácil implementación, hace que sea un paradigma de clasificación muy extendido. Existen varios tipos de algoritmos de K-NN que se enuncian a continuación.

Algortimo K-NN básico

La notación que se utiliza se muestra en la figura 2.1

		X_1	.....	X_j	.....	X_n	C
(x_1, c_1)	1	x_{11}	.....	x_{1j}	.....	x_{1n}	c_1
(x_i, c_i)	i	x_{i1}	.....	x_{ij}	.....	x_{in}	c_i
(x_N, c_N)	N	x_{N1}	.....	x_{Nj}	.....	x_{Nn}	c_N
X	N+1	$x_{N+1,1}$	.....	$x_{N+1,j}$	.....	$X_{N+1,n}$	?

Figura 2.1 Notación para el Paradigma K-NN

Se cuenta un fichero denominado D de N casos, cada uno caracterizado por n variables predictores, x_1, …….. x_n y una variable a predecir, la clase C.

Los N casos se denotan por

(x_1, c_1), ……. (x_N, c_N) , donde

$x_i = (x_i, 1 \ldots . x_{i,n})$ para todo i = 1,…… N

$c_i \in \{c_1, \ldots, c_m\}$ para todo i = 1,…..N

C_1,……c_m expresan los m posibles valores de la variable clase C

El nuevo caso que se pretende clasificar se denota por $x(x_1, \ldots, x_n)$.

A continuación se muestra el pseudocódigo para el clasificador básico, donde se calculan las distancias de todos los casos ya clasificados al nuevo caso x que se quiere clasificar.

COMIENZO

Entrada: $D = \{(x_1, c_1), \ldots., (x_N c_N)\}$

$x = (x_1, \ldots. x_n)$ nuevo caso a clasificar

Para todo objeto ya clasificado (x_i, c_i)

Calcular $d_i = d(x_i, x)$

Ordenar $d_i (i = 1 \ldots.. N)$ en orden ascendente

Seleccionar los K casos D_x^K ya clasificados más cercanos a x

Asignar a *x* la clase más frecuente en D_x^K

FIN

Pseudocódigo K-NN básico

Cuando se seleccionan los K casos que ya están clasificados D_x^K más cercanos al nuevo caso x, se le asignará la clase correspondiente, o sea la variable C, que sea más frecuente entre los K objetos D_c^K. De acuerdo con lo anterior, se muestra de manera gráfica un ejemplo, figura 2.2 donde se tienen 24 casos ya clasificados en dos posibles valores de m (m=2).

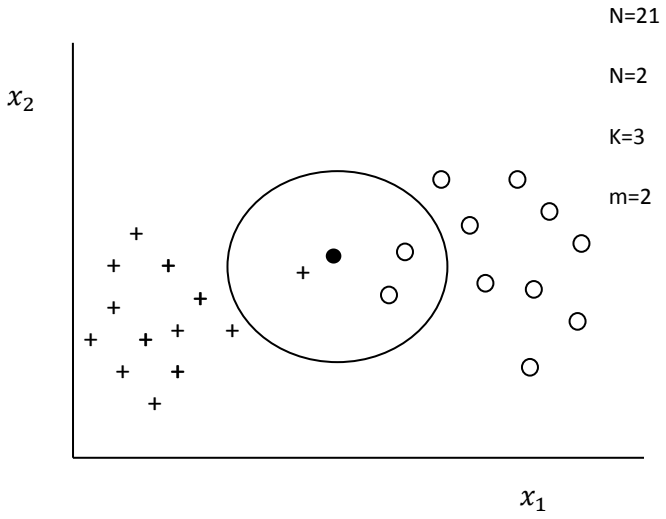


Figura 2.2. Aplicación del algoritmo K-NN básico

Las variables predictoras que se muestran en la figura 2.2 son x_1 y x_2, donde se ha seleccionado *K*= 3. Se puede observar que de los 3 casos ya clasificados se encuentran más cercanos al nuevo caso a clasificar x, representado por un círculo negro, dos de ellos pertenecen a la clase ´**o**´, por lo

tanto el clasificador k es 3-NN, y predice la clase ´**o**´ para el nuevo caso, donde notamos que el caso más cercano a x pertenece a la clase ´**+**´. Es decir, que si hubiéramos utilizado un clasificador 1-NN, x_1 hubiera sido asignado a la clase ´**+**´.

El paradigma K-NN es a veces un tanto atípico si lo comparamos con el resto de los paradigmas de clasificación, debido a que en el resto de ellos, la clasificación de un nuevo caso se lleva a cabo de dos maneras, como son la inducción y la deducción sobre el nuevo caso. En el paradigma K-NN, estas dos maneras de clasificar se encuentran colapsadas en lo que se denomina transducción. En caso de que se produzca un empate entre dos o más clases, conviene tener una regla heurística para su ruptura, como pueden ser, la selección de la clase que contenga al vecino más próximo, o seleccionar una clase con distancia media menor.

Es importante mencionar que la determinación del valor K, constata empíricamente que el porcentaje de casos bien clasificados es no monótono con respecto a K. Un ejemplo claro se explica en la figura 2.3, donde se muestra que una buena elección de valores de k, está entre 3 y 7.

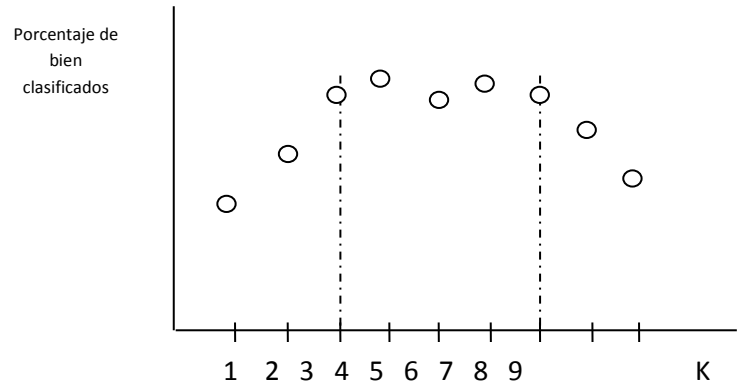


Figura 2. 3. Ejemplo de la no monotocidad del porcentaje de valores que están bien clasificados en función de K.

Como se ha mencionado previamente, el algoritmo K-NN es extendido. Algunas variantes sobre este algoritmo se detallan a continuación.

K-NN con rechazo

El K-NN con rechazo se utiliza para clasificar un caso en el que se deben tener ciertas garantías para asegurar que la clase a asignar sea la correcta. Para ilustrar esto, mencionaremos dos ejemplos para llevar a cabo una clasificación con garantía [9].

1) El número de votos obtenido por la clase deberá superar un umbral prefijado, si suponemos que trabajamos con K= 10 y m= 2, dicho umbral puede establecerse en 6.
2) El otro ejemplo sería del tipo de mayoría absoluta para la clase a asignar. Así, suponemos que K=20 y m= 4, se puede convenir que la asignación de un nuevo caso a una clase, sólo se lleve a cabo en el caso de que la diferencia entre las frecuencias mayor y la segunda mayor supere un valor de 3.

K-NN con distancia media

Esta variante se fundamente sobre la idea de asignar un nuevo caso a la clase cuya distancia media sea menor, un ejemplo se muestra en la figura 2.4.

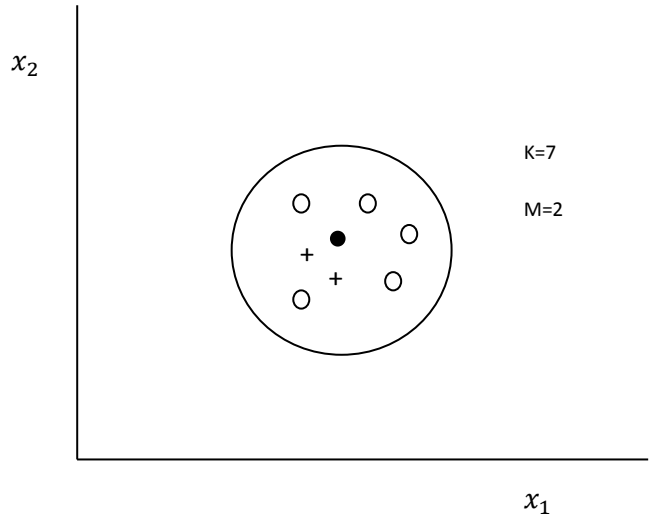


Figura 2.4. Ejemplo de K-NN distancia media

A pesar de que de los casos más cercanos al mismo pertenecen a la clase ´**o**´, el nuevo caso se clasifica como ´**+**´, ya que la distancia media a los dos casos ´**+**´ es menor que la distancia media a los cinco casos ´**o**´.

K-NN con distancia mínima

En esta variante de distancia mínima se comienza seleccionado un caso por clase, normalmente se elige el caso más cercano al baricentro de todos los elementos de dicha clase. En este paso se reduce la dimensión del fichero de casos a almacenar de N a m; a continuación se asigna el nuevo caso a la clase cuyo representante esté más cercano. Podemos ver cómo 1-NN aplicado a un conjunto de m casos (uno por cada clase), es inferior al K-NN genérico, estando su efectividad condicionada a la homogeneidad dentro de las clases (cuanto mayor sea la homogeneidad, el procedimiento es más efectivo). El coste computacional al aplicar este procedimiento es menor.

2.4 Máquinas de Vectores Soporte SVM

Definiremos las características teóricas de las SVM [8] para desarrollar problemas de clasificación de dos clases, donde se mencionará la función de decisión y la importancia de generalización, definiremos también el margen duro, para conjuntos de datos de entrenamiento que son linealmente separables en el espacio de entrada, así, como también nos extenderemos a los casos linealmente no separables, los cuales se trasladarán de un espacio de entrada con una cierta dimensión a un espacio de características con mayor dimensionalidad, cuyo propósito es separar linealmente los datos linealmente no separables.

2.4.1 Funciones de decisión

Considerando el problema de clasificación de un punto, cuyas características están dadas por un vector x, tal que $x = (x_1, \ldots . x_p)$, el cual pertenece a una de las dos clases posibles. Supongamos que tenemos las funciones $f_1(x)$ y $f_2(x)$, conocidas como funciones de decisión que definen las clases 1 y 2, por lo cual podemos clasificar al punto x dentro de la clase 1 si:

$f_1(x) > 0$, $f_2(x) < 0$,

o bien, en la clase 2 si

$f_1(x) < 0$, $f_2(x) > 0$,

Las funciones de decisión se estiman a partir de pares de entrada-salida denominado *entrenamiento*, estos métodos convencionales de entrenamiento determinan las funciones de tal forma que cada par de entrada y salida sea correctamente clasificado dentro de la clase a la que pertenece.

La figura 2.5, muestra un ejemplo donde se asume que los datos son representados por los símbolos ‘cuadrado‘ y ‘circulo’ pertenecen respectivamente a la clase 1 y 2. Resulta claro que los datos de entrenamiento no se intersectan en ningún momento y siendo posible trazar una línea separando los datos de manera correcta.

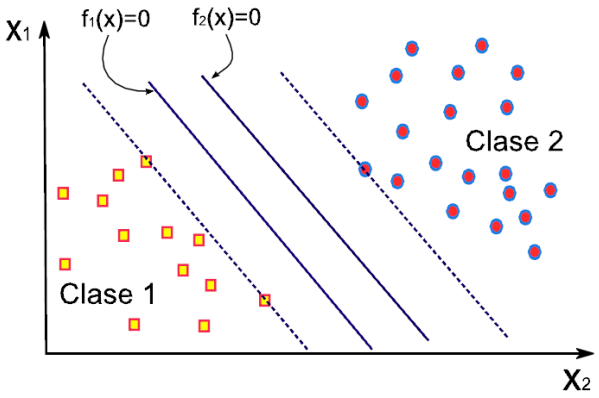


Figura 2.5 Funciones de decisión

En este tipo de clasificación, ya sea que la función de decisión $f_1(x)$ o la función $f_2(x)$ se muevan hacia la línea punteada de su propio lado, el conjunto de datos de entrenamiento aún sigue siendo correctamente clasificado y proporcionando la certeza de que es posible encontrar un conjunto infinito de hiperplanos que correctamente clasifiquen los datos de entrenamiento, es claro que la precisión de clasificación será directamente afectada por la posición de las funciones de decisión. Las SVM comparándolas con otros métodos de clasificación, consideran esta desventaja y encuentran la función de decisión de tal forma que la distancia entre los datos de entrenamiento sea maximizada. La función obtenida recibe el nombre de función de decisión óptima o hiperplano óptimo [8].

En la actualidad muchos investigadores han utilizado las SVM en aplicaciones tales como identificador de firmas, de rostros, categorización de texto y una cantidad cada vez más creciente de utilidades. Tomando en consideración que su enfoque está motivado por el aprendizaje estadístico [39,41], las SVM producen modelos matemáticos importantes y que son geométricamente intuitivos y realmente fundamentados. Como la principal motivación de las SVM es separar varias clases en el conjunto de entrenamiento con una superficie que maximice el margen entre éstas, su implementación consta del principio de minimización estructural que permite minimizar el error medio cuadrático sobre el conjunto de datos de entrenamiento. Esta filosofía se usa a menudo en los métodos de minimización de riesgo empírico. Para entrenar una SVM se requiere un conjunto de n ejemplos, donde cada ejemplo consiste en un vector de entrada x_i y una etiqueta y_i, asociada al vector de entrada.

La función de decisión de la SVM contiene n parámetros libres, los llamados Multiplicadores de Lagrange positivos α_i, i = 1,….n, donde cada α_i es una medida de cuánto, el correspondiente ejemplo de entrenamiento influye en la función. Muchos de los ejemplos no intervienen en la obtención de la función y consecuentemente los correspondiente α_i asociados resultan ser nulos.

2.4.2 Caso linealmente separable

Sea un problema de clasificación binaria, donde los datos de entrenamiento se estructuran como sigue:

$$(x_1, y_1), (x_2, y_2), \ldots\ldots, (x_1, y_1), x \in \mathbb{R}^n \in \{+1, -1\} \qquad (2.1)$$

La figura 2.6 muestra varios hiperplanos de decisión que pueden realizar la separación, y es claro distinguir que existen diversos hiperplanos que podrían realizar la separación sin ningún trabajo.

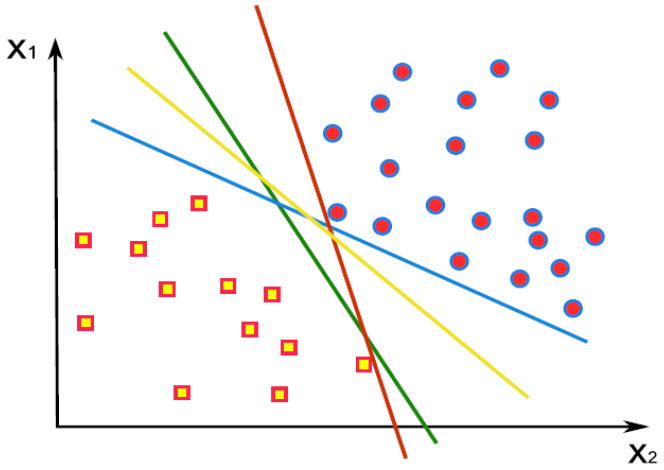


Figura 2.6 Clasificador clásico

Sin embargo, se pretende encontrar un hiperplano con el máximo margen de separación posible, este plano se conoce como hiperplano óptimo [8]; el límite de decisión representado por la línea que separa el espacio de entrada se define por la ecuación siguiente,

$$w^T x_i + b = 0 \qquad (2.2)$$

Por lo que, el problema radica en encontrar un mejor límite de decisión como la función de separación óptima. El caso más simple de las SVM es el caso linealmente separable en el espacio de características.

Al optimizar el margen geométrico, se fija el margen funcional k_i=1, (llamado también hiperplano canónico [8]), de ahí que, el clasificador lineal se represente como:

$$y_i = \pm 1,$$

$$< w \cdot x^+ > \ + b = 1$$

$$< w \cdot x^- > \ + b = -1 \qquad (2.3)$$

Y éstos pueden ser combinados dentro de un conjunto de desigualdades:

$$y_i(< w \cdot x_i > + b) \geq 1 \ \forall \ i \qquad (2.4)$$

El margen geométrico de x^+ y x^- es

$$\gamma_i = \frac{1}{2}\left(\langle \frac{w}{\|w\|} \cdot x^+ \rangle - \langle \frac{w}{\|w\|} \cdot x^- \rangle\right)$$

$$= \frac{1}{2\|w\|}[< w \cdot x^+ >]$$

$$= \frac{1}{\|w\|} \qquad (2.5)$$

Donde w define el hiperplano de separación óptima y b el sesgo, tomando en cuenta que la distancia entre el hiperplano de separación y el dato de entrenamiento que se encuentre más cercano, es denominado margen.

Es de notar que la habilidad de generalización va a depender de la localización del hiperplano de separación y el hiperplano con el máximo margen, llamado hiperplano de separación óptimo. Intuitivamente nos damos cuenta que la generalización es maximizada si el hiperplano de separación óptima es tomado en cuenta como el hiperplano de separación. Por lo que optimizar el margen geométrico significa minimizar la norma del vector de pesos, definida por la ecuación (2.5)

Mediante la técnica de programación cuadrática, se trata de encontrar el hiperplano óptimo y dos hiperplanos (H1 y H2) paralelos. Cuando la distancia entre H1 y H2 resulta ser máxima, algunos puntos de datos pueden situarse sobre el hiperplano H1 y algunos otros en el H2. A estos puntos de datos se les denomina vectores soporte [8,2], debido a que participan de forma directa para definir el hiperplano de separación. Los puntos que no se sitúan próximos a los hiperplanos, pueden ser removidos o cambiados sin cruzar los hiplerplanos H1 y H2 sin que esto suponga una modificación en la generalización del clasificador.

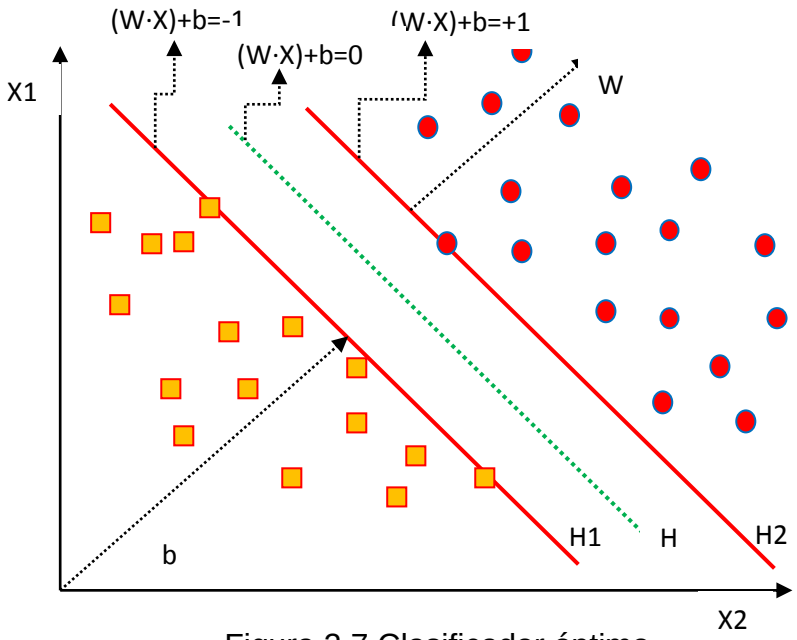


Figura 2.7 Clasificador óptimo

La solución que ofrece una SVM, para definir la función de decisión depende únicamente de un pequeño conjunto de vectores soporte, por lo que cualquier hiperplano puede ser representado mediante $w, x\ y\ b$, donde w es un vector perpendicular al plano.

En la figura 2.7 se muestra la representación geométrica del problema de programación cuadrádica, donde H representa el separador óptimo y H1 y H2 los hiperplanos, por lo que el problema original de optimización queda expresado de la siguiente manera:

Proposición 1 [8] : caso linealmente separable,

$S = [(x_1, y_i), \ldots\ldots, (x_l, y_l)]$, si el hiperplano (*w,b*) es la solución

$\min_{w,b} < w \cdot w \geq \|w\|^2$

$sujeto\ a: y_i(< w \cdot x_i > + b\ \geq 1$

Entonces, el hiperplano tiene un margen geométrico máximo $\gamma = \frac{1}{\|w\|}$

Continuando con la solución, se cambió el problema dual utilizando los Multiplicadores de Lagrange (ML), existiendo dos razones que lo avalan, la primera razón radica en el hecho de que las condiciones de las SVM en la ecuación (2.3), serán reemplazadas por los ML, debido a la facilidad que

ofrecen para conducir el proceso hacia la solución. La segunda razón es que en la reformulación del problema, los datos de entrenamiento únicamente aparecerán en la forma de producto punto cartesiano entre los vectores, debido a que es una propiedad fundamental que permitirá generalizar el procedimiento el caso no lineal, por lo que el langrangiano viene dado:

$$L(w,b,\alpha) = \frac{1}{2} < w \cdot w > \ - \sum_{i=l}^{l} \alpha_i \left[y_i (\langle w \cdot x_i \rangle + b) - 1 \right] \qquad (2.6)$$

donde α_i, son los Multiplicadores de Lagrange.

Por lo tanto, el dual se encuentra primero diferenciándolo con respecto a *w* y *b*.

$$\frac{\partial (w,b,\alpha)}{\partial w} = w - \sum_{i=l}^{l} \alpha_i y_i x_i = 0 \ \rightarrow w = \sum_{i=1}^{l} \alpha_i y_i x_i \qquad (2.7)$$

$$\frac{\partial (w,b,\alpha)}{\partial b} = - \sum_{i=l}^{l} \alpha_i y_i = 0 \ \rightarrow w = \sum_{i=1}^{l} \alpha_i y_i \qquad (2.8)$$

en segundo lugar, sustituyendo las relaciones obtenidas del langragiano original como sigue

$$L(w,b,\alpha) = \frac{1}{2} \langle w \cdot w \rangle - \sum_{i=1}^{l} \alpha_i \left[\ y_i (\langle w \cdot x_i \rangle + b) + b - 1 \right]$$

$$= \frac{1}{2} \langle \sum_{i=1}^{l} \alpha_i y_i x_i \cdot \sum_{i=1}^{l} \alpha_i y_i x_i \rangle - \ \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j \ \left(\langle x_j \cdot x_i \rangle + b \right) - \sum_{i=1}^{l} \alpha_i$$

$$= \frac{1}{2} \sum_{i=1}^{l} \alpha_i \, y_i \alpha_i y_j \langle x_i \cdot x_j \rangle - \sum_{i,j=1}^{l} \alpha_i \, y_i \alpha_i y_j \langle x_i \cdot x_j \rangle - \sum_{i=1}^{l} \alpha_i y_i b + \sum_{i=1}^{l} \alpha_i$$

$$= \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \, y_i \alpha_i y_j \langle x_i \cdot x_j \rangle - \sum_{i=1}^{l} \alpha_i \qquad (2.9)$$

Los puntos para los cuales α_i> 0, se denominan ‘vectores soporte’ situándose en uno de los hiperplanos H1 y H2. Para el resto de muestras de entrenamiento $\alpha_i = 0$, sitúandose sobre H1 y H2 de tal forma que las condiciones de la ecuación 2.4, se cumplen. Los vectores soporte son los elementos críticos del conjunto de entrenamiento y éstos son los que se acercan más al límite de decisión.

Cuando se entrena el conjunto inicial de datos, obtenemos un hiperplano que separa perfectamente los datos y es definido por un conjunto reducido de

vectores soporte. Aunque todos los demás puntos fueran eliminados o desplazados alrededor sin atravesar H1 o H2, y el entrenamiento fuera repetido, encontraríamos el mismo hiperplano de separación, el cual se define por el mismo conjunto. El problema original de optimización se describe a continuación para el caso linealmente separable.

$S = [(x_1, y_1), \dots (x_i, y_i)]$, si α_i es una solución al problema de optimización cuadrática

$$\max_{\alpha_i} = \frac{1}{2}\sum_{i=1}^{l} \alpha_i \, y_i \alpha_i y_j \langle x_i \cdot x_j \rangle - \sum_{i,j=1}^{l} \alpha_i \qquad (2.10)$$

$$Sujeto\ a\colon \sum_{i=1}^{l} \alpha_i \, y_i = 0$$

Entonces $\|w\|^2$ comprende el mínimo $w^* = \sum_{i=1}^{l} \alpha_i^* \, y_i x_i$, y el margen geométrico $\gamma^* = \frac{1}{\|w^*\|}$ es maximizado.

Condiciones de Karush-Kuhn-Tucker

Las condiciones de Karus-Kuhn-Tucker (KKT) [21,19], participan de manera importante en la teoría de optimización, debido a que proporcionan las condiciones necesarias para obtener una mejor solución en los problemas de optimización general.

Teorema [21]. Dado un problema de optimización formado por un dominio convexo:

$$\Omega \subseteq \mathbb{R}^n$$
$$minimizar\ f(w), \quad w \in$$
$$sujeto\ a \quad g_i(w) \leq 0, i = 1, \dots . k,$$
$$h_i(w) = 0, i = 1, \dots, m,$$

Con $f \in C^1$ convexa, las condiciones necesarias y suficientes para que un punto normal w^* sea un óptimo, son la existencia de α^*, β^* tal que

$$\frac{\partial L(w^*,\alpha^*,\beta^*)}{\partial w} = 0 \quad (2.11)$$

$$\frac{\partial L(w^*,\alpha^*,\beta^*)}{\partial \beta} = 0 \quad (2.12)$$

$\alpha_i^* g_i(w^*) = 0, i = 1, \dots k,$

$\alpha_i^*(w^*) \leq 0, i = 1, \dots k,$

$\alpha_i^* \geq 0, i = 1, \dots k,$

De acuerdo a las condiciones KKT, se tiene que, si el conjunto de entrenamiento es linealmente separable, debe verificar lo siguiente:

$$\|w^*\|^2 = \langle w^* \cdot w^* \rangle = (\sum_{i \epsilon sv} \alpha_i^*)^{-\frac{1}{2}} \quad (2.13)$$

Por lo cual, la distancia máxima de un hiperplano es

$$\frac{1}{\|w^*\|} = (\sum_{i \epsilon sv} \alpha_i^*)^{-\frac{1}{2}} \quad (2.14)$$

2.4.3 Hiplerplanos con márgenes suaves

El problema de aprendizaje que se mencionó anteriormente, es válido para el caso donde los datos a clasificar son linealmente separables, lo cual significa que los datos de entrenamiento no solapan entre ellos. En la práctica este tipo de problemas no son comunes, sin embargo, existen algunos ejemplos en los que el hiperplano de separación suele dar buenos resultados, aún cuando los datos se intersecten en la solución de programación cuadrática. No suele ser una buena solución para los casos que presentan este tipo de intersecciones, debido a la condición $y_i(\langle w \cdot x_i \rangle + b) \geq 1 \, \forall i$. En la figura 2.8, se puede observar que los puntos que se encuentran en la intersección, y no pueden ser correctamente clasificados o en general, para cualquier dato mal clasificado x_i, su correspondiente α_i tenderá a infinito.

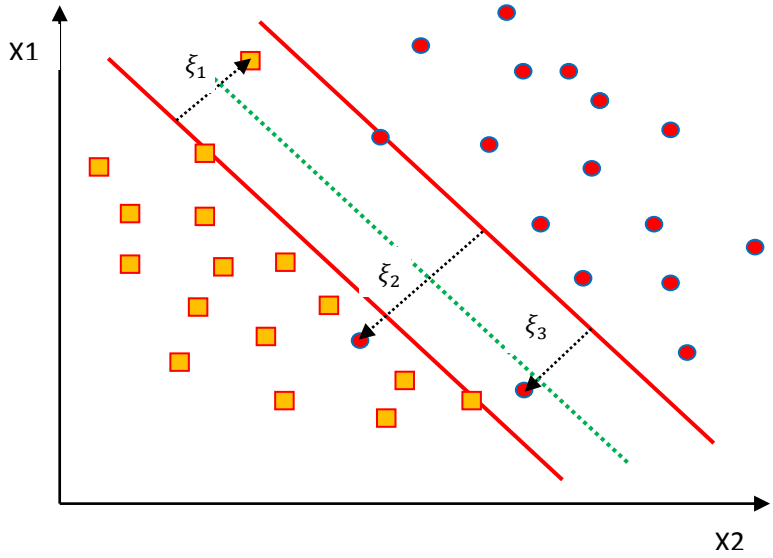


Figura 2.8 Hiperplano con margen suave

Cuando se desea encontrar un clasificador con margen máximo, se deberán introducir unas variables de holgura no negativas $\xi_i(\geq 0)$, como se muestra en la ecuación 2.15.

$$y_i(\langle w \cdot x_i \rangle + b) \geq 1 - \xi_i \ \forall i \qquad (2.15)$$

En la ecuación (2.15) las variables ξ_i, se introducen con el fin de encontrar una solución factible. Para los datos de entrenamiento x_i, si $0 < \xi_i < 1$, éstos no poseen el margen máximo, pero pueden ser correctamente clasificados. Por lo tanto, el ancho del margen blando puede ser supervisado por el parámetro de penalización C, el cual determina la relación entre el error de entrenamiento y la dimensión Vapnik-Chervonenkis (VC) del módulo.

La dimensión VC describe la capacidad de un conjunto de funciones implementadas en una máquina de aprendizaje. Para la clasificación binaria, h es el máximo número de puntos en el que pueden ser separadas dos clases en todas las 2^{h} formas posibles usando las funciones de la máquina de aprendizaje.

Cuando C es grande, se ha comprobado que el número de errores de clasificación es pequeño, a la vez que $w^t w$ es grande. Es importante mencionar que cuando se toma C = ∞, el número de datos mal clasificados tiendan a ser 0. Sin embargo, este caso no es posible, ya que el problema

puede ser factible únicamente para algún valor C < ∞ . Por lo tanto, al introducir estas variables de holgura no negativas $\xi_i (i = 1, l)$ al problema de optimización, en lugar de las condiciones de la ecuación (2.6), el hiperplano de separación deberá satisfacer :

$$\min_{w,b,\xi_i} \langle w \cdot w \rangle + C \sum_{i=1}^{l} \xi_i^2 \qquad (2.16)$$

Sujeto a :

$$y_i(< w \cdot x_i > + b) \geq 1 - \xi_i \quad \text{con} \quad \xi_i \geq 0$$

y cumpliendo

$$\langle w \cdot x_i \rangle + b \geq +1 - \xi_i, para\ y_i = +1, \quad con \quad \xi_i \geq 0$$

$$\langle w \cdot x_i \rangle + b \geq -1 + \xi_i, para\ y_i = -1, \quad con \quad \xi_i \geq 0$$

Si $\xi_i < 0$, entonces

$$y_i(\langle w \cdot x_i \rangle + b) \geq -\xi_i \geq 1, \qquad (2.17)$$

por lo tanto, no se considera el caso en que $\xi_i < 0$.

Para el máximo suave, el langrangiano original está dado por:

$$L(w, b, \xi_i, \alpha) = \frac{1}{2} \langle w \cdot w \rangle - \sum_{i=1}^{l} \alpha_i \ [y_i(\langle w \cdot x_i \rangle + b) - 1 + \xi_i \ ] + \frac{C}{2} \sum_{i=1}^{l} \xi_i^2 \quad (2.18)$$

De la misma manera que en el caso linealmente separable, el primer caso se obtiene diferenciando con respecto a w y b, y para despues sustituir en el lagrangiano original, de tal manea que el problema dual sería de la siguiente manera:

$$\max_{\alpha_i} -\frac{1}{2} \sum_{i,j=1}^{l} \alpha_i y_i \alpha_j y_j \left[\langle x_i \cdot x_j \rangle + \frac{1}{C} \delta_{ij} \right] + \sum_{i=1}^{l} \alpha_i \qquad (2.19)$$

Sujeto a $\sum_{i=1}^{l} \alpha_i y_i = 0$

La condición de Kuhn-Tucker es

$$\alpha_i^* [y_i(\langle w^* \cdot x_i \rangle + b^*) - 1 + \xi_i \ ] = 0 \qquad (2.20)$$

Asimismo, el problema de optimización cuadrática es el mismo que en el caso separable, diferenciándolo únicamente en las cotas modificadas con los

multiplicadores de lagrange α_i. Así pues, el parámetro C es fija por el usuario, el cual debe ser estima usando una técnica de validación cruzada [2][8].

2.4.4 Tipos de Kernel

En las SVM, tal y como se ha mencionado anteriormente, su hiperplano óptimo se determina para realizar una maximizacion en su habilidad de generación. En el caso no lineal, el clasificador obtenido no tiene una alta habilidad de generalización, aún cuando el hiperplano sea determinado óptimamente. Un ejemplo sería el maximizar el espacio entre clases, es decir el espacio de entrada transformarlo a un espacio de alta dimensión, el cual se denomina “espacio de características”.

En SVM, en el caso no lineal, su idea principal es transformar los vectores de entrada $x \in \mathbb{R}^n$ dentro de vectores $\emptyset(x)$ a un espacio de características de alta dimensión[2], donde $\emptyset$ representa el mapeo: $\mathbb{R}^n \rightarrow \mathbb{R}^f$ y así resolver el problema de clasificación lineal en este espacio de características.

$$x \in \mathbb{R}^n \rightarrow \emptyset(x) = [\emptyset_1(x), \emptyset_2(x), \ldots\ldots\ldots \emptyset_n(x)]^T \in \mathbb{R}^f \qquad (2.21)$$

Para el conjunto de hipótesis, se consideran las siguientes funciones:

$$f(x) = \sum_{i=1}^{l} w_i \emptyset_i(x) + b \qquad (2.22)$$

Donde $\emptyset: X \rightarrow F$ define una proyección no lineal desde un espacio de entrada a un espacio de características, el procedimiento de aprendizaje consta de dos pasos: el primero, una proyección no lineal transforma los datos dentro de un espacio de características F y después, una máquina lineal se utiliza para clasificar los datos en dicho espacio de características.

Una propiedad de las máquinas de aprendizaje lineal es que éstas pueden ser expresadas en una representación dual, esto significa que la ecuación (2.20) puede ser una combinación lineal de los puntos de entrenamiento.

$$f(x) = \sum_{i=1}^{l} \alpha_i y_i \langle \phi(x_i) \cdot \phi(x)) \rangle + b \qquad (2.23)$$

Para completar la construcción de una máquina de aprendizaje no-lineal, es necesario capturar el producto $\langle \phi(x_i) \cdot \phi(x)) \rangle$ en el espacio de características, como una función de los puntos de entrada originales, a esto se le llama función Kernel[38].

Un Kernel se define como una función K, tal que para para todo $x, z \, \varepsilon \, X$.

$$K(x, z) = \langle \phi(x_i) \cdot \phi(z) \rangle \qquad (2.24)$$

Donde ϕ representa una proyección de X a un espacio de características F.

Uno de los puntos importantes es encontrar una función kernel que pueda ser evaluada de manera eficiente. Después de esto, la función puede expresarse

$$f(x) = \sum_{i=1}^{l} \alpha_i y_i K \langle x_{i,} \cdot x \rangle + b \qquad (2.25)$$

Las condiciones de Mercer [5], proveen una caracterización cuando una función k(x,z) es un kernel, dado un espacio de entrada infinito $X = \{x_1, \ldots . x_n\}$, y suponiendo que K(x,z) es una función simétrica de X, la matriz resulta ser:

$$K = (K(x_i \cdot x_j))_{i,j=1}^{n} \qquad (2.26)$$

Considerando que K es una matriz simétrica, existe una matriz ortogonal V tal que K= $\mathrm{V} \, \Lambda \, \mathrm{V}'$, donde Λ es la matriz diagonal que contiene los autovalores λ_t de K, con sus correspondientes autovalores $v_t = (v_{ti})_{i=1}$. Asumiendo que todos los autovalores son no-negativos y considerando la proyección siguiente,

$$\phi: x_i \rightarrow \left(\sqrt{\lambda_t} v_{ti}\right)_{t=1}^{n} \in \mathbb{R}^n, i = 1, \ldots \ldots n.$$

Por lo tanto, se tiene

$$\langle \phi(x_i) \cdot \phi(x_j) \rangle = \sum_{t=1}^{n} \lambda_t v_{ti} v_{tj} = (\mathrm{V} \, \Lambda \, \mathrm{V}')_{ij} = K_{ij} = K(x_i \cdot x_j) \qquad (2.27)$$

La ecuación anterior expresa que $K(x, z)$ es una función “kernel” correspondiente a la proyección ϕ. Se require que los autovalores de K sean

no negativos, existe un autovalor negativo λ_s en el autovector v_s, el punto definido por la ecuación (2.28)

$$z = \sum_{i=1}^{n} v_{si}\, \phi(x_i) = \sqrt{\Lambda}\, \mathrm{V}'\mathrm{v}_s \qquad (2.28)$$

en el espacio de características podría tener una norma cuadrada definida por (2.29)

$$\|z\|^2 = \langle z \cdot z\rangle = \mathrm{v}_s\, \mathrm{V}\sqrt{\Lambda}\sqrt{\Lambda}\, \mathrm{V}'\mathrm{v}_s = \mathrm{v}'_s \mathrm{V}\Lambda\mathrm{V}'\mathrm{v}_s = \mathrm{v}'_s K \mathrm{v}_s = \lambda_s < 0, \qquad (2.29)$$

contradiciendo la geometría de este espacio. Esto nos lleva a la siguiente **Proposición**: sea X un espacio de entrada finito con una función simétrica sobre $K(x,z)$. Decimos que $K(x,z)$ es una función kernel si y solamente si la matriz definida en (2.30) es semidefinida positiva, esto es posee autovalores no negativos.

$$K = (K(x_i \cdot x_j))_{ij=1}^{n} \qquad (2.30)$$

Permitiendo una ligera generalización de un producto escalar en un espacio de Hilbert [8] e introduciendo un peso λ_i para cada dimensión se tiene la expresión (2.31),

$$\langle \phi(x) \cdot \phi(z)\rangle = \sum_{i=1}^{\infty} \lambda_i \phi_i\,(x) \cdot \phi_i(z) = K(x,z) \qquad (2.31)$$

por lo tanto, el vector de características sería

$$\phi(x) = (\phi_i(x), \phi_i(x), \ldots\ldots, \phi_i(x), \ldots\ldots) \qquad (2.32)$$

El teorema de Mercer, proporciona las condiciones necesarias y suficientes para que una función simétrica continua K(x,z) sea representada por:

$$K(x,z) = \sum_{i=1}^{\infty} \lambda_i \phi_i\,(x) \cdot \phi_i(z) \qquad (2.33)$$

con λ_i son no negativos, lo que es equivalente a que K(x,z) sea un producto en el espacio de características F⊇ $\phi(X)$, donde F es el espacio l_2 de todas las secuencias:

$$\psi = (\psi_1, \psi_2, \ldots\ldots, \psi_i, \ldots)$$

$$\sum_{i=1}^{\infty} \lambda_i \psi_i^2 < \infty \qquad (2.34)$$

Esto implícitamente induce un espacio definido por el vector de características y como una consecuencia directa una función lineal en F puede ser representada por

$$f(x) = \sum_{i=1}^{\infty} \lambda_i, \psi_i, \phi_i(x) + b = \sum_{j=1}^{l} \alpha_i, y_i K\left(x, x_j\right) + b \qquad (2.35)$$

Donde la primera expresión indica la representación original y la segunda es el dual [2]. La relación entre los dos está dada por la siguiente ecuación,

$$\psi = \sum_{j=1}^{l} \alpha_i y_i \phi(x_j) \qquad (2.36)$$

En la representación original, el número de términos en la suma es igual a la dimensionalidad en el espacio de características, mientras que en el dual existen l términos. La analogía con el caso finito es similar, la contribución a partir del análisis funcional nos conduce al problema para ecuaciones integrales de la forma:

$$\int_x K(x, z)\phi(z)dz = \lambda_i\, \phi(x) \qquad (2.37)$$

donde K(x,z) es una función kernel acotada, simétrica y positiva y x es un espacio compacto.

Teorema de Mercer. Sea X un subconjunto compacto de $\mathbb{R}^n$.y suponiendo que K es una función simétrica continua

$$T_K: L_2(\mathrm{X}) \rightarrow L_2(x),$$

$$(T_K f)(\cdot) = \int_x \mathrm{K}(\cdot, x) f(x) dx, \qquad (2.38)$$

donde el operador integral es positivo, esto es,

$$\int_{X\,x\,X} K(x, z) f(x) f(z) dx dz \geq 0 \qquad (2.39)$$

Para toda $f \in L_2(X)$, entonces K(x,z) puede ser expandida en una serie uniformemente convergente (sobre $X\,x\,X$) en términos de las autofunciones

$\phi_j \in L_2(X)$, normalizadas de tal forma que $\|\phi_j\|_{L_2}$ = 1 y autovalores asociados positivos $\lambda_j \geq 0$,

$$K(x,z) = \sum_{j=1}^{\infty} \lambda_j \, \phi_j(x)\phi_2(z) \qquad (2.40)$$

Tabla 2.1 Tipos de Kernel utilizados en SVM

Tipo de "kernel"	Expresión
Lineal	$\boldsymbol{K(x,y) = x^T y}$
Polinomial	$\boldsymbol{K(x,y) = (1 + \langle x,y \rangle)^d}$, con grado d
Función de Base Radial	$\boldsymbol{K(x,y) = \exp^{\{-\|x-y\|/2\sigma^2\}}}$

2.4.5 Caso linealmente no separable

Los clasificadores lineales presentados en las dos secciones anteriores resultan ser relativamente limitados. En la mayoría de las clases, no únicamente se solapan o intersectan los datos al generar un hiperplano de separación, sino que la separación genuina de estos datos está dada por hiperplanos no-lineales. Una característica del enfoque presentado anteriormente radica en que éste puede ser fácilmente extendido para crear cotas de decisión no lineal. El motivo de tal extensión es que una SVM puede crear un hiperplano de decisión no lineal, capaz de clasificar datos separables no linealmente. Generalmente, para patrones de entrada n dimensionales, en lugar de una curva no lineal, una SVM creará un hiperplano de separación no lineal.

El problema de optimización utilizando "kernels" se representa gráficamente de la siguiente manera [2][38]:

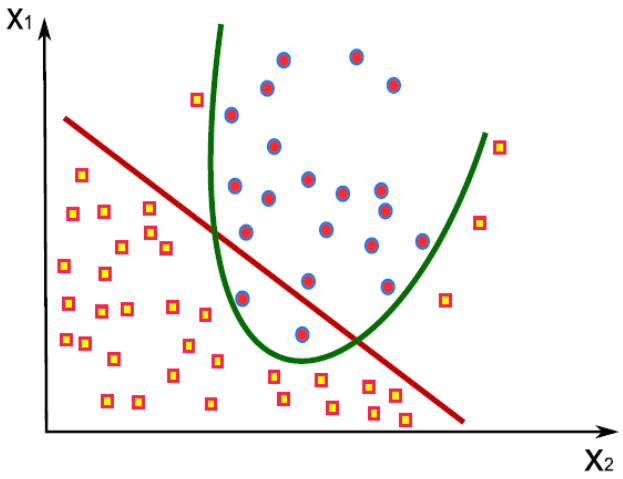


Figura 2.9 Clasificador no lineal

Dado un conjunto de datos de entrenamiento

$$S = [(x_1, y_1) \ldots\ldots (x_l, y_l)] \tag{2.41}$$

En un espacio de características $\phi(x)$ definido por el kernel

$K(x,z) = \langle \phi(x) \cdot \phi(y) \rangle$, la solución de $\max_{\alpha_i} -\frac{1}{2}$

$$\sum_{ij=1}^{l} \alpha_i y_i \alpha_j y_j \left[K(x_i, x_j) + \frac{1}{C} \delta_{ij} \right] + \sum_{i=1}^{l} \alpha_i \tag{2.42}$$

sujeto a : $\sum_{i=1}^{l} \alpha_i \, y_i = 0$

con α_i^*, $f(x) = \sum_{i=1}^{l} \alpha_i^* \; y_i K(x_i, x) + b^*$ (2.43)

donde b^* es elegido tal que

$$y_i f(x_i) = 1 - \xi^* = 1 - \frac{\alpha^*}{C}$$

$$w^* = \sum_{i=1}^{l} \alpha_i^* y_i K(x, x) \tag{2.44}$$

La regla de decisión $sgn[f(x)]$ es equivalente al hiperplano en el espacio de características definido por el kernel $K(x,z)$, el cual resuelve el problema de optimización. El margen geométrico está dado por

$$\gamma^* = \left(\sum_{i \in sv} \alpha_i^* - \frac{1}{C}\langle \alpha^* \cdot \alpha^* \rangle\right)^{-\frac{1}{2}} \quad (2.45)$$

Utilizando el Kernel

$$K'(x,z) = K(x,z) + \frac{1}{C}\delta_x(z) \quad (2.46)$$

El margen blando en L1, queda como sigue,

$$\min_{w,b,\xi_i} \langle w \cdot w \rangle + C \sum_{i=1}^{l} \xi_i$$

Sujeto a: $y_i(\langle w \cdot x_i \rangle + b) \geq 1 - \xi_i \quad (2.47)$

Con $\xi_i \geq 0$

Donde el langrangiano original es

$$L(w,b,\xi_i,\alpha) = \frac{1}{2}\langle w \cdot w \rangle - \sum_{i=1}^{l} \alpha_i \left[y_i(\langle w \cdot x_i \rangle + b) - 1 + \xi_i\right] + C \sum_{i=1}^{l} \xi_i - \sum_{i=1}^{l} \gamma_i \xi_i \quad (2.48)$$

Por otra parte, el dual está dado por

$$w(\alpha) = -\frac{1}{2}\sum_{i,j=1}^{l} \alpha_i y_i \alpha_j y_j \langle x_i \cdot x_j \rangle + \sum_{i=1}^{l} \alpha_i \quad (2.49)$$

Este es el mismo que el margen máximo, sibien sujeto a las restricciones

$$C - \alpha_i - \gamma_i = 0, \gamma_i \geq 0 \Longrightarrow C \geq \alpha_i \quad (2.50)$$

con las condiciones de Kuhn-Tucker

$$\gamma_i \xi_i = 0 \text{ ó } (\alpha_i - C)\xi_i = 0 \quad (2.51)$$

$$\alpha_i[y_i(\langle w \cdot x_i \rangle + b) - 1 + \xi_i] = 0 \quad (2.52)$$

donde $\xi_i \neq 0$, $\gamma_i = 0, \Rightarrow C = \alpha_i$ con $\xi_i = 0$ el margen es máximo, α_i es positivo y puede incrementarse hasta C, por lo tanto C $\geq \alpha_i \geq 0$

Dado un conjunto de datos de entrenamiento

$$S = [(x_1, y_1) \ldots\ldots (x_l, y_l)], \quad (2.53)$$

un espacio de características $\phi(x)$ definido por el kernel

$K(x,z) = \langle \phi(x) \cdot \phi(y) \rangle$, la solución de

$$\max_{\alpha_i} -\frac{1}{2}\sum_{ij=1}^{l} \alpha_i y_i \alpha_j y_j K(x_i, x_j) + \sum_{i=1}^{l} \alpha_i \quad (2.54)$$

Sujeto a: $\sum_{i=1}^{l} \alpha_i y_i = 0, C \geq \alpha_i \geq 0$

La ecuación $\alpha_i^*, f(x) = \sum_{i=1}^{l} \alpha_i^* \; y_i K(x_i, x) + b^*$, define la regla de decisión sgn$[f(x)]$ y es equivalente al hiperplano en el espacio de características definido por el Kernel(x, Z), el cual resuelve el problema de optimización. El margen geométrico está dado por

$$\gamma^* = \left(\sum_{i \in sv} \alpha_i^*\right)^{-\frac{1}{2}} \quad (2.55)$$

Cuando la cota de α_i es C, se origina el problema del máximo margen.

Elegir C es lo mismo que obtener v en

$$\max_{\alpha_i} -\frac{1}{2}\sum_{ij=1}^{l} \alpha_i y_i \alpha_j y_j K(x_i, x_j) \quad (2.56)$$

Sujeto a: $\sum_{ij=1}^{l} \alpha_i y_i = 0$

$\sum_{ij=1}^{l} \alpha_i \geq v,$

$\frac{1}{l} \geq \alpha_i \geq 0$

Para una solución no-óptima, $\hat{\alpha}$ es el valor actual de las variables duales. El vector de pesos se calcula mediante la igualdad siguiente $\frac{\partial L}{\partial w} = 0$.

La solución $\hat{w} \cdot \hat{w}$ satisface las condiciones originales con

$$\frac{1}{2}\|\hat{w}\|^2 + c\sum_{i=1}^{l} \xi_i \; \inf_{w,b} L(w, b, \hat{\alpha}) \quad (2.57)$$

como solución dual factible, ya que:

$$L = \frac{1}{2}\langle w \cdot w \rangle - \sum_{i=1}^{l} \alpha_i \left[y_i(\langle w \cdot x_i \rangle + b) - 1 \right] \quad (2.58)$$

$$w = \sum_{i=1}^{l} \hat{\alpha}_i \, y_i x_i \quad (2.59)$$

$$\frac{\partial L}{\partial b} = 0 \rightarrow \sum_{i=1}^{l} \alpha_i y_i = 0 \qquad (2.60)$$

Se calcula la diferencia entre las soluciones original y dual factibles $C - \alpha_i = \gamma_i$

$$\frac{1}{2}\|w\|^2 + c\sum_{i=1}^{l}\xi_i - \inf_{w,b} L(w, b, \hat{\alpha})$$

$$= \sum_{i=1}^{l} \hat{\alpha}_i \left[y_i \left(\sum_{i=1}^{l} y_i \alpha_i K\langle x_j \cdot x_i \rangle + b \right) - 1 + \xi_i \right] + \sum_{i=1}^{l} \gamma_i \xi_i$$

$$= C\sum_{i=1}^{l}\xi_i + \sum_{i=1}^{l} \hat{\alpha}_i \left[y_i \left(\sum_{i=1}^{l} y_i \alpha_i K\langle x_j \cdot x_i \rangle + b \right) - 1 \right]$$

$$= \sum_{i=1}^{l} \hat{\alpha}_i y_i y_j K\langle x_j \cdot x_i \rangle - \sum_{i=1}^{l} \hat{\alpha}_i + C\sum_{i=1}^{l}\xi_i$$

$$= \sum_{i=1}^{l} \hat{\alpha}_i - 2w(\alpha) + C\sum_{i=1}^{l}\xi_i \qquad (2.61)$$

2.4.6 Construcción de un clasificador SVM

El hiperplano no lineal de separación [32] puede encontrarse como la solución de

$$Max\ W(\alpha) = \sum_{i=i}^{l} \alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{i=j}^{l} \alpha_i\, \alpha_j y_i y_j K(x_i, x_j) \qquad (2.62)$$

$$sujeto\ a \qquad \sum_{i=1}^{l} y_i\, \alpha_i = 0, \qquad 0 < \overline{\alpha}_i < C, \qquad i = 1, \ldots \ldots l$$

La función de decisión es:

$$f(x) = sgn\,(w \cdot z + b) sgn\left(\sum_{i=1}^{l} \alpha_i y_i\, K\left(x_i, x_j\right) + b \right) \qquad (2.63)$$

2.4.7 SVM Multiclase

En principio, las SVM han sido propuestas para clasificación binaria. Luego este método se ha extendido a clasificaciones multiclase, en la cual se emplean combinaciones de SVM binarias con la finalidad de considerar todas las clases a la vez.

Método ***uno contra todos***. Consiste en combinar clasificadores binarios entrenados independientemente para resolver distintas partes del problema. Este método construye *k* modelos de SVM, donde *k* es el número de clases. Cada SVM *k-ésimo* entrenará con todos los ejemplos, etiquetando los de la

clase *k-ésima* con etiqueta positiva, y al resto de elementos de las demás clases, con etiqueta negativa.

Método ***todos contra todos***. En este método se construyen *K(K-1)/2* clasificadores, uno para cada par de clases posibles, enfrentando así a todas las clases una a una. En cada entrenamiento se emplea únicamente los datos de las dos clases involucradas. Así pues, un problema con 4 clases generaría los siguientes clasificadores: 1 vs 2, 1 vs 3, 1 vs 4, 2 vs 3, 2 vs 4 y 3 vs 4. Una vez realizado esto, se someten los datos de *test* a todos estos clasificadores, donde se añade un voto a la clase ganadora para cada caso. Finalmente, aquella que más votos obtenga será la clase propuesta por el sistema.

2.5 Predictores Conformales

2.5.1 Transducción

Vapnik [38,41] formuló una distinción entre la inducción y la transducción que se aplican a los problemas de predicción, en la figura 2.10 se muestra un esquema relativo al proceso de predicción. En la predicción inductiva en primer lugar, los ejemplos se manejan con una regla más o menos general, que podríamos llamar como una regla de predicción o decisión, un modelo o una teoría. En el paso inductivo, cuando se presenta un nuevo ejemplo, obtenemos una predicción más general que en el paso deductivo. En la predicción transductiva, se toma un acceso directo, por lo que se pasa de los ejemplos anteriores, directamente a la predicción de un nuevo ejemplo.

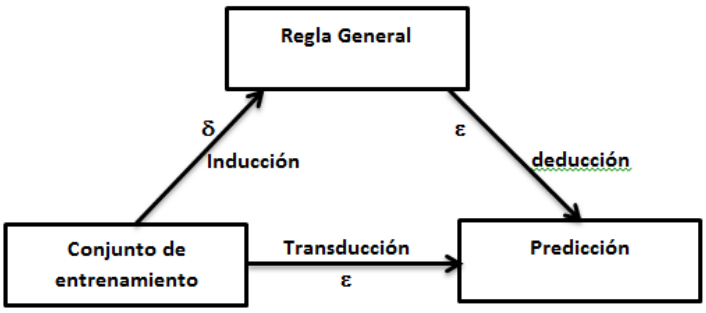


Figura 2.10 Predicción Inductiva y transductiva

Algunos ejemplos típicos de la etapa de la inducción sobre la estimación de parámetros junto con la búsqueda de un concepto estadístico relativo a la teoría de aprendizaje [22], nos indica que los ejemplos de predicción transductiva son la estimación de las futuras observaciones en las estadísticas asociadas a los datos [6] y cómo se comporta también, en la máquina de aprendizaje de vecinos cercanos.

En el caso de las predicciones sencillas, la distinción entre la inducción y la transducción resulta ser menos nítida.

Un método para realizar la transducción, en línea, es un método para predecir y_n de $x_1, y_1, \ldots\ldots, x_{n-1}, y_{n-1}, x_n$. Este método proporciona una predicción para cualquier ejemplo que pueda ser presentado como x_n, y así lo define, al menos implícitamente, por regla general, podría ser extraído de $x_1, y_1, \ldots\ldots, x_{n-1}, y_{n-1}$ (inducción), y posteriormente se aplica a x_n para predecir y_n (deducción).

La estimación fiable de la confianza sigue siendo un reto importante, por lo que es conveniente prestar atención a los algoritmos de aprendizaje, los cuales son muy utilizados en el mundo real, como son las aplicaciones en reconocimiento de patrones en señales, o en algunas aplicaciones científicas. En los últimos años, Vovk, Shafer y Gammerman [42], han propuesto un enfoque de la teoría de juegos a la estimación de la confianza, llamado Predicción Conformal (CP, Conformal Predictor en inglés), que posee varias propiedades importantes para su posible uso en aplicaciones del mundo real, esta teoría CP está basada en los principios de aleatoriedad algorítmica, en la inferencia transductiva y en la prueba de hipótesis. También se basa en la relación entre la inferencia derivada transductiva y en la complejidad Kolmogorov [42] de una *iid* (idénticamente distribuidos de forma independiente), en una secuencia de ejemplos de los datos.

El método de CP es relativamente reciente, por lo que se describen a continuación los detalles del mismo, incluyendo el algoritmo de Predicción Conformal para clasificación.

Con el método de Predicción Conformal, se puede predecir que un nuevo ejemplo tiene una etiqueta, y que lo hace de forma similar a los ejemplos anteriores de una manera determinada. Para la estimación de la confianza en la predicción, los predictores conformales también son los predictores de confianza.

La figura 2.11 muestra el ejemplo de una familia de conjuntos anidados en la predicción, donde se muestra en negro la zona con menor confianza, la confianza media en gris obscuro y muy confiados en gris claro.

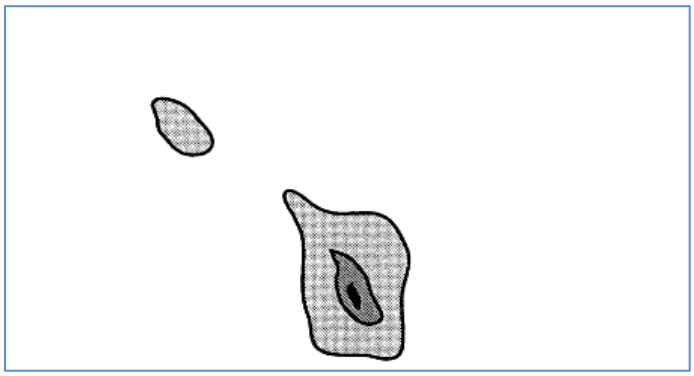


Figura 2.11 Ejemplo de una familia de conjuntos anidados en la predicción.

Los predictores conformales conforman un método basado en una serie de predicciones sucesivas, y más específicas en la confianza. Es importante resaltar, que para describir el método, debemos asumir las suposiciones siguientes:

1) las salidas reales son parejas sucesivas

$$(x_1, y_1), (x_2, y_2), \ldots \ldots, (x_n, y_n) \qquad (2.64)$$

Donde la secuencia de datos se denominan ejemplos, cada ejemplo (x_i, y_i) de un dato x_i con etiqueta y_i. Los ejemplos son elementos del espacio medible X**,** llamado el espacio de ejemplos, mientras que las etiquetas son elementos de un espacio medible Y**,** llamado espacio de etiquetas [42].

2) Asumimos que X es no vacío y que Y contiene al menos 2 elementos esenciales diferentes. Cuando necesitamos una definición más compacta, escribimos z_i para (x_i, y_i) ,

$$Z \coloneqq X \; x \; Y \tag{2.65}$$

Y llamamos Z al espacio de ejemplos, por lo tanto, la secuencia infinita de datos (2.59) es un elemento de un espacio medible Z^{∞}.

Cuando los ejemplos están ausentes, nos referimos a que $|X| = 1$. En este caso x_i no conlleva ninguna información y no necesita ser considerado, entonces al identificar Y y Z, se asume que se eligen en realidad los ejemplos de forma independiente de alguna distribución de probabilidad Q en Z, es decir, que la secuencia infinita $z_1, z_2, \ldots., z_n$, se extrae de la distribucion de potencia de probabilidad Q^{∞} en Z^{∞}.

2.5.2 Predictores simples

Cuando se realiza una prueba, se tiene el ejemplo y después se le asigna la etiqueta a y_n, o si simplemente se quiere predecir y_n, se necesita la siguiente función.

$$D: Z^{*} \; x \; X \; \rightarrow Y \tag{2.66}$$

A esta función se le denomina predictor simple, siempre que se puede medir, para cualquier secuencia de ejemplos, es decir $x_1, \; y_1, \ldots\ldots, \; x_{n-1}, \; y_{n-1} \; \in Z^{*}$, y para algún nuevo ejemplo, digamos $x_n \in X$, genera la secuencia $(x_1, y_1, \ldots\ldots, x_{n-1}, y_{n-1}, x_n) \; \in Y$, siendo y_n su predicción para la nueva etiqueta.

Es posible tener una noción más compleja en la predicción [41], en lugar de limitarse a la elección de un único elemento de y, como la predicción para y_n, se amplía la misma a un rango más o menos preciso en la predicción de y de cada etiqueta con un cierto grado de confianza de esta forma, se obtienen subconjuntos de y que sean lo suficientemente grandes, tales que se pueda estar seguro de que y_n, caiga entre ellos, esto supone una cierta seguridad,

además de obtener subconjuntos más pequeños en el que estamos más seguros de la existencia del elemento buscado.

Un algoritmo que permite predecir en este sentido, requiere una entrada adicional que se identifica como $\varepsilon \in (0,1)$, y se denomina nivel de significación. El valor complementario 1-ε se conoce como nivel de confianza, dando todas estas entradas tenemos

$$x_1, y_1, \ldots\ldots, x_{n-1}, y_{n-1}, x_n, \varepsilon, \tag{2.67}$$

La representación del predictor Γ, con las salidas de los subconjuntos de Y, se expresa a continuación.

$$\Gamma^{\varepsilon}\ (x_1, y_1, \ldots\ldots, x_{n-1}, y_{n-1}, x_n) \tag{2.68}$$

Siguiendo la notación generalmente aceptada, posicionamos ε como un exponente en lugar de colocarlo con los demás argumentos.

Para reducir el subconjunto a la medida del nivel de significación ε, el predictor Γ debe satisfacer

$$\Gamma^{\varepsilon}\ (x_1, y_1, \ldots\ldots, x_{n-1}, y_{n-1}, x_n) \subseteq \Gamma^{\varepsilon_2}\ (x_1, y_1, \ldots\ldots, x_{n-1}, y_{n-1}, x_n) \tag{2.69}$$

Cada vez que se cumple, $\varepsilon_1 \geq \varepsilon_2$ y una vez que se observa que la secuencia de datos incompletos

$$x_1, y_1, \ldots\ldots, x_{n-1}, y_{n-1}, x_n \tag{2.70}$$

y se elige un nivel de significación ε , el predictor Γ predice que

$$y_n \in\ \Gamma^{\varepsilon}\ (x_1, y_1, \ldots\ldots, x_{n-1}, y_{n-1}, x_n) \tag{2.71}$$

El valor más pequeño del nivel de significación ε, es el más seguro de la confianza de la predicción. De acuerdo a la condición (2.69), se tiene más confianza en menos predicciones específicas. Formalmente, se llama una función medible.

$$\Gamma\colon Z^*\ x\ X\ x(0,1) \rightarrow 2^Y \tag{2.72}$$

donde 2^Y es el conjunto de todos los subconjuntos de Y que satisface (2.61). Para todos los niveles de significación, $\varepsilon_1 \geq \varepsilon_2$, todos los enteros positivos n, y todas las secuencias de datos incompletos de (2.70), es un indicador de confianza (o predictor de confianza determinista). Para que el predictor Γ sea medible, para cada n, el conjunto de secuencia $\varepsilon, x_1, y_1, \ldots, x_n, y_n$ que satisface (2.71), es un subconjunto medible (0,1) x $(X\ x\ Y)^n$.

2.5.3 Validación

En la validez, para cada nivel de significación ε, se quiere tener la confianza de 1-ε en la predicción (2.71) sobre y_n. Esto significa que la probabilidad de que la predicción sea por error, -la probabilidad del evento (2.63) no sucede- debe ser ε. Por otra parte, se construye una secuencia completa de las predicciones [41], primero para y_1 y después para y_2, y así sucesivamente, se busca que estos eventos de error sean independientes. Si se cumplen estas condiciones, no importa lo intercambiable de la distribución de probabilidad P debido a que gobierna la secuencia de ejemplos, entonces se dice que el predictor de confianza Γ es 'exactamente valido' o más brevemente 'exacto', lo que significa que al cometer errores con el predictor de confianza Γ es como conseguir realizar lanzamientos independientes de una moneda sesgada cuya probabilidad es siempre ε. Si las probabilidades de los errores pueden ser incluso menores que ε, entonces se dice que el predictor de confianza Γ es "conservador válido" o, de forma simplificada "conservador".

Reformulando las definiciones de una manera lo suficientemente clara, se introduce una notación formal de los errores del predictor de confianza Γ para cuando se procesa la secuencia de datos.

$$\omega = (x_1, y_1, x_2, y_2, \ldots\ldots) \tag{2.73}$$

El nivel de significación ε., nos indica que si el predictor de confianza Γ comete un error, la enésima prueba puede ser representada por 1 en el caso de un error y 0 en el caso de no existir un error.

$$err_n^{\varepsilon}(\Gamma,\omega) = \begin{cases} 1 & if\ \ y_n\ \notin\ \Gamma^{\varepsilon}\ (x_1, y_1, \ldots., x_{n-1}, y_{n-1}, x_n) \\ 0 & de\ otra\ forma \end{cases} \qquad (2.74)$$

y el número de errores durante los primeros ensayos n es

$$Err_n^{\varepsilon}(\Gamma,\omega) = \sum_{i=1}^{n} err_n^{\varepsilon}(\Gamma,\omega) \qquad (2.75)$$

Resulta fácil, pensar en el número de errores err_n^{ε} y Err_n^{ε} que se producirán durante la predicción, por lo que en el contexto del siguiente protocolo, se presentan ejemplos y predicciones que pueden ocurrir cuando existen estos errores:

$Err_n^{\varepsilon} \coloneqq 0 \quad para\ todo\ \ \varepsilon\ (0,1)$

$FOR\ \ n = 1,2, \ldots \ldots \ldots$

$Salidas\ reales \quad x_n \in X;$

$Salida\ predictor \quad err_n^{\varepsilon} \subseteq Y \qquad para\ todos \qquad \varepsilon \in (0,1);$

$Salidas\ reales\ y_n\ \in Y$

$Err_n^{\varepsilon} \coloneqq \begin{cases} 1\ if\ y_n \notin\ \Gamma_n^{\varepsilon} \\ 0\ otraforma \end{cases} para\ todos\ \varepsilon\ \in (0,1)$

$Err_n^{\varepsilon} \coloneqq\ Err_{n-1}^{\varepsilon} +\ err_n^{\varepsilon}\ \ para\ todos\ \varepsilon\ \in (0,1)$

$End\ FOR$

Figura 2.12 Protocolo de aprendizaje de error

El protocolo de aprendizaje de error es un conjunto que considera las estrategias para ambos conjuntos y una predicción real. Se ha asumido que en realidad la estrategia es al azar y sus movimientos se generan a partir de una distribución de probabilidad intercambiable de P en Z^{∞}. Un Predictor de confianza Γ es por definición, una estrategia de cómo medir el predictor.

Si ω son extraídos de una distribución de probabilidad intercambiable P, el número de $err_n^{\varepsilon}\ (\Gamma\ , P)$, y el predictor de confianza Γ es exactamente válido si para cada ε

$$err_1^{\varepsilon}\ (\Gamma, \mathrm{P}), err_2^{\varepsilon}\ (\Gamma, \mathrm{P}), \ldots err_n^{\varepsilon}\ (\Gamma, \mathrm{P}) \qquad (2.76)$$

La expresión anterior representa una secuencia de variables independientes del tipo Bernoulli al azar con el parámetro ε, es decir que si se trata de una secuencia de variables aleatorias independientes, alguna de ellas tiene la probabilidad ε de ser 1 y una probabilidad 1-ε de ser cero, no importando la intercambiabilidad de probabilidad P.

Según el teorema 2.1 [41], el predictor de confianza no es exactamente válido. Formalmente, lo anterior se concreta en el hecho de que el predictor de confianza Γ es conservador y válido si por alguna distribución de probabilidad intercambiable $P^{\infty}\ en\ Z^{\infty}$, existe un espacio de probabilidad con dos conjuntos del tipo siguiente,

$$\left(\xi_n^{(\varepsilon)}: \varepsilon\ \in (0,1), n = 1,2 \ldots\ldots\right), (\eta_n^{(\varepsilon)}: \varepsilon\ \in (0,1), n = 1,2 \ldots\ldots) \qquad (2.76)$$

De {0,1} variables aleatorias con valores tales que, cumplan las siguientes restricciones:

- Para fijar ε , $\xi_1^{(\varepsilon)}, \xi_2^{(\varepsilon)}$,.....debe ser una secuencia de variables independientes de Bernoulli con el parámetro ε
- Para todo n y ε, $\eta_n^{(\varepsilon)}\ \leq\ \xi_n^{(\varepsilon)}$
- La unión de la distribución para $err_n^{\varepsilon}(\Gamma, \mathrm{P}), \varepsilon\ \in (0,1), n = 1,2 \ldots\ldots$
- Coincide con la distribución conjunta de $\eta_n^{(\varepsilon)}, \varepsilon\ \in (0,1), n = 1,2 \ldots\ldots$

Podría resultar lógico requerir sólo $\eta_n^{(1)}\ \leq\ \xi_n^{(2)}$ cada vez que $\varepsilon_1 \geq \varepsilon_2$, si bien la inclusión de esta restricción no introduce novedades con respecto a las formuladas previamente.

Para concluir, se define válido el predictor de confianza Γ debido a que es asintóticamente exacto mediante alguna distribución de probabilidad intercambiable P en Z^{∞} en la generación del ejemplo $z_1, z_{2,\ldots\ldots\ldots}\ z_n$ y cualquier nivel e importancia de ε

$$\lim_{n\to\infty} \frac{Err_n^{\varepsilon}(\Gamma,\mathrm{P})}{n} =\ \varepsilon \qquad (2.77)$$

Si existe una probabilidad baja, el predictor de confianza Γ resulta ser es asintóticamente conservador por alguna distribución de probabilidad

intercambiable P en Z y cualquier nivel de significación ε, tal y como se expresa en la siguiente ecuación,

$$\lim_{n\to\infty} \frac{Err_n^{\varepsilon}(\Gamma,\mathrm{P})}{n} \leq \varepsilon \qquad (2.78)$$

Según se demuestra en [41], un predictor de confianza exacto es también asintóticamente exacto. Un predictor de confianza conservador es asintóticamente conservador. Y es una consecuencia inmediata de la ley de los grandes números.

2.5.4 Predictores aleatorios de confianza

Los predictores aleatorios de confianza, predicen al azar la confianza de los elementos pertenecientes a un espacio de probabilidad auxiliar. La ventaja principal de la asignación al azar en este contexto, es que hay muchos factores predictivos de confianza aleatorios que son exactamente válidos. Formalmente se define un factor de confianza con asignación al azar mediante la siguiente función medible.

$$\Gamma\colon (X\ x\ [0,1]\ x\ Y)^*\ x\ (X\ x\ [0,1]x(0,1) \to 2^Y \qquad (2.79)$$

Para todo nivel de significación $\varepsilon_1 \geq \varepsilon_2$, todo entero positivo n, y todas las secuencias de datos incompletas,

$$x_1, \tau_1, y_1, \ldots\ldots\ldots\ldots, x_{n-1}, \tau_{n-1}, y_{n-1}, x_n, \tau_n, \qquad (2.80)$$

donde

$x_i \in X$, $\tau_i \in [0,1]$ e $y_1 \in Y$ para todos i, satisface

$$\Gamma^{\varepsilon_1}(x_1, \tau_1, y_1, \ldots\ldots\ldots\ldots, x_{n-1}, \tau_{n-1}, y_{n-1}, x_n, \tau_n) \subseteq$$
$$\Gamma^{\varepsilon_2}\ (x_1, \tau_1, y_1, \ldots\ldots\ldots\ldots, x_{n-1}, \tau_{n-1}, y_{n-1}, x_n, \tau_n)$$

Se supone que τ_1 y τ_2 son variables aleatorias que son independientes entre ellas mismas y de cualquier otra variable, y que se distribuyen de manera uniforme en el intervalo [0,1], tal y como se espera obtener de un generador de números aleatorios. Se definen $err_1^{\varepsilon}\ (\Gamma, \omega), \text{y}\ Err_1^{\varepsilon}\ (\Gamma, \omega),$ como se indicó previamente, sólo que ahora dependen también de $\tau's$. En otras palabras

$err_1^{\varepsilon}(\Gamma, \omega)$ es definida por (2.74) con x_i, siendo los ejemplos $x_i \in X\ x\ [0\ 1]$; $err_1^{\varepsilon}(\Gamma, \omega)$, se define a través dela ecuación (2.75)

En general, muchas definiciones de los predictores de confianza al azar son casos especiales de las definiciones correspondientes a los predictores de confianza determinista, éste último sólo debe aplicarse en el espacio de ejemplos extendido $\mathrm{X} \times [0,1]$; (cuyos elementos son tanto al ejemplo y el número al azar que se utiliza en una prueba determinada).

Si Γ es un predictor de confianza al azar, P es una distribución canjeable en Z y $n \in N$, se dice

$$err_n^{\varepsilon}\big(\Gamma, (x_1, \tau_1, y_1, x_2, \tau_2, y_2)\big) \tag{2.81}$$

y

$$Err_n^{\varepsilon}\big(\Gamma, (x_1, \tau_1, y_1, x_2, \tau_2, y_2, \dots\dots\dots)\big) \tag{2.82}$$

donde $(x_1, y_1)(x_2, y_2)$ se han extraído de P y τ_1, τ_2, se elaboran de forma independiente de U^{∞}, la distribución uniforme en $[0,1]^{\infty}$, son variables aleatorias que se denominarán $err_n^{\varepsilon}(\Gamma, \mathrm{P})$ y $Err_n^{\varepsilon}(\Gamma, \mathrm{P})$, respectivamente. Decimos que el predictor Γ es exactamente válido si por cada $\varepsilon(0,1)$, la expresión (2.76) es una secuencia de variables independientes de Bernoulli al azar con el parámetro ε.

2.5.5 Predictores conformales

En este apartado se define el concepto de una medida de No conformidad, la cual intuitivamente, es una manera de medir qué tan diferente es un nuevo ejemplo de los ejemplos anteriores. Existen muchas medidas de no conformidad diferentes, y cada una define un factor de predicción de conformación.

El orden en que aparecen los ejemplos que deben ser aprendidos, no debería alterar el propio proceso de aprendizaje. Para la formalización de este punto, se define el concepto de *bolsa* (también llamado conjunto múltiple). Una

bolsa de tamaño $n \in N$ es un conjunto de n elementos, donde algunos pueden ser idénticos, siendo similar a un conjunto en el que el orden de sus elementos no es relevante, pero difiere de un conjunto en que la repetición es permitida. Para identificar una *bolsa*, es necesario determinar los elementos que contiene y existen de cada clase, en caso de existir repetición [41].

Se escribe $\int z_1, \ldots\ldots\ldots z_n \int$ para identificar la bolsa que contiene en los elementos $z_1, \ldots\ldots\ldots z_n$, algunos de los cuales pueden ser idénticos entre sí.

Aunque los elementos de una *bolsa* no tienen la necesidad de estar ordenados, no existe un orden en la notación, y se hará uso de ella. El siguiente ejemplo sencillo aclara este concepto, se tiene una *bolsa* z de tamaño 20, formada por $\int z_1, \ldots\ldots\ldots z_{20} \int$; de la bolsa, obtenemos $z_1, \ldots\ldots\ldots z_{20}$ cuando ignoramos su orden, porque hemos identificado la *bolsa* con elementos que aciertan, se puede manipular utilizando el conocimiento de orden, por ejemplo hablando de la *bolsa*, y se pretende eliminar z_6 (dejando cualquier otra z_i que podría ser igual al z_6), tras esta operación queda una *bolsa* de tamaño 19, de la *bolsa* $\int z_1, \ldots z_5 \ldots z_7 \ldots z_{20} \int$

Escribimos $Z^{(n)}$ para el conjunto de todas las bolsas de tamaño n de los elementos de un espacio medible Z. El conjunto $Z^{(n)}$ es en sí mismo un espacio medible, puede definirse formalmente como todo el espacio $Z^{(n)}$ con un álgebra no estándar, que consta de subconjuntos medibles de $Z^{(n)}$ que contiene todas las permutaciones de sus elementos. Escribimos Z(*) para el conjunto de todas las *bolsas* de los elementos de Z (la unión de todos $Z^{(n)}$).

2.5.6 Medidas de conformidad y no conformidad

Como ya hemos mencionado, una medida de no conformidad es una forma de marcar lo diferente que es un nuevo ejemplo de una *bolsa* de ejemplos anteriores. Formalmente, una medida de no conformidad es una asignación mensurable.

$$A: Z(*)x\, Z \rightarrow \overline{\mathbb{R}}; \qquad (2.83)$$

Para cada *bolsa* posible de ejemplos anteriores y cada posible nuevo ejemplo, A asigna un puntaje numérico que indica cuán diferente es el nuevo ejemplo de los anteriores.

Para obtener medidas de no conformidad, es fácil cuando ya se dispone de métodos para la predicción manual:

- Si los ejemplos no son más que números ($Z = R$) y es natural que se tome el promedio de los ejemplos anteriores, como el predictor simple de la nueva muestra, entonces se podría definir la no conformidad de un nuevo ejemplo como el valor absoluto de su diferencia del promedio de los ejemplos anteriores. Por otra parte, se podría usar en lugar del valor absoluto de su diferencia, la mediana de los ejemplos anteriores.

En la teoría general [41], se define cualquier función medible en $Z(*)\ x\ Z$ tomando valores en la recta real extendida una medida de no conformidad.

Dada una medida de no conformidad A, y una secuencia $z_1, \ldots z_l$ de ejemplos, y el nuevo ejemplo z, se puede notar lo diferente que z es de la *bolsa* $z_1, \ldots z_l$, por lo que A($z_1, \ldots z_l$) es la medida de no conformidad para Z.

Formalmente una medida de no conformidad es una función medible del tipo $Z(*)x\ Z \to \overline{\mathbb{R}}$. Al no haber diferencia entre las medidas de conformidad como objetos matemáticos, las medidas de no conformidad aparecen como una idea derivada. Dada una medida de conformidad de B, se puede definir una medida de no conformidad con cualquier transformación estrictamente decreciente, por ejemplo $A := -B$ o tal vez, si B toma sólo valores positivos $A := A/B$. De acuerdo con los objetivos planteados por los autores en [41], la predicción, a partir de la conformidad, consiste en prever que la nueva etiqueta será mejor que una nueva muestra. Las etiquetas que se incluyen en la predicción, será una de las mejores etiquetas, que hacen que un nuevo ejemplo sea conforme con los ejemplos anteriores. Las etiquetas que se incluyen en la predicción (la más conforme), en lugar de las etiquetas que se han excluido (las más disconformes). A menudo resulta más natural comenzar con las medidas de no conformidad, por ejemplo, cuando se compara un nuevo ejemplo con un promedio de ejemplos anteriores, se suele definir en primer

lugar una distancia entre los dos más cercanos. Por esta razón, hacemos hincapié en la no conformidad y conformidad.

A veces resulta conveniente considerar por separado como una medida de no conformidad, ofertas de medidas con *bolsas* de diferentes tamaños: si A es una medida de no conformidad, para cada n = 1,2,. . . definimos la función como:

$$A_n: Z^{n-1}\, x\, Z \rightarrow \overline{\mathbb{R}} \qquad (2.84)$$

que expresa la restricción de A a Z^{n-1} x Z. La secuencia $(A_n: n \epsilon N)$, donde (A_n) abreviado, también expresa medidas de no conformidad.

Dada una medida de no conformidad (A_n) y la bolsa $z_1, \ldots z_n$, se puede calcular la puntuación de no conformidad como sigue,

$$\alpha_i = A_n(\int z_1, \ldots . z_{i-1}, z_{i+1}, \ldots . , z_n \int z_i \qquad (2.85)$$

Para cada ejemplo z_i, en una *bolsa*, una medida de no conformidad (A_n) , puede ser escalado, el valor numérico de α_i no lo es por sí solo, nos dice lo inusual de la medida (A_n) para el z_i encontrado, para esto, necesitamos una comparación de α_i para el resto α_j. Un procedimiento apropiado para hacer esta comparación es computar la fracción.

$$\frac{\left|\{j=1,\ldots..,n:\, \alpha_j \geq \alpha_i\}\right|}{n} \qquad (2.86)$$

Esta fracción, que toma valores en el rango $1/n$ y 1, se denomina "*p-value"* para z_i. La fracción de los ejemplos en la *bolsa*, es la no conformidad z_i, si es pequeño (cerca de su límite inferior $1/n$ para un *n* grande), entonces z_i no es muy conforme (es un valor atípico). Si es grande (cerca de su límite superior 1), entonces z_i es muy conforme.

Si se comienza con una medida de conformidad (B_n) en lugar de una medida de no conformidad, entonces podemos definir el p-value para z_i, con

$$\frac{\left|\{j=1,\ldots..,n:\, \beta_j \geq \beta_i\}\right|}{n} \qquad (2.87)$$

Donde β_j representan puntuaciones de conformidad, mediante esta expresión se consigue el mismo resultado que el obtenido en (2.86) usando una medida de no conformidad (A_n) y obtenidos a partir de (β_n) por medio de una transformación estrictamente decreciente.

2.5.7 Confianza y Credibilidad

Toda medida de no conformidad determina un factor de predicción de confianza. Dado un nuevo ejemplo $\mathrm{x_n}$ y un nivel de importancia, este factor predictivo proporciona un conjunto de predicción que debe contener la etiqueta del ejemplo, y se obtiene un conjunto al suponer que $\mathrm{y_n}$, tendrá un valor que hace ($\mathrm{x_n}$, $\mathrm{y_n}$) conforme con los ejemplos anteriores, el nivel de significación determina la cantidad de la conformidad (una medida de "*p-value"*) que se requiere [41].

Formalmente, el predictor de conformación determinado por una medida de no conformidad (A_n), es un predictor de confianza Γ, obtenido mediante la siguiente expresión,

$$\Gamma^{\varepsilon}\ (x_1, y_1, \ldots\ldots, x_{n-1}, y_{n-1}, x_n) \tag{2.88}$$

Siendo igual al conjunto de todas las etiquetas $y \in Y$ de tal manera que

$$P(y) = \frac{|\{i=1,\ldots..,n:\ \alpha_i \geq \alpha_n\}|}{n} > \varepsilon, \tag{2.89}$$

donde

$$\alpha_i := A_n(\int (x_1, y_1, \ldots\ldots, x_{n-1}, y_{n-1}), (x_{i+1}, y_{i+1}), \ldots\ldots, (x_{n-1}, y_{n-1}),$$

$$(x_i, y)\int, (x_i, y_i)), i = 1, \ldots\ldots, n-1$$

$$\alpha_i := A_n(\int (x_1, y_1, \ldots\ldots, x_{n-1}, y_{n-1})\int \quad, (x_n, y)) \tag{2.90}$$

Confianza

El valor de confianza definido en [16], podría ser el nivel de aleatoriedad *iid* (independientes e idénticamente distribuidos) de la etiqueta y, sin embargo, esta etiqueta puede diferir de la verdadera etiqueta del ejemplo sin etiquetar; si

este es el caso, entonces el nivel de aleatoriedad de la etiqueta que se desea predecir no es un límite de probabilidad de una predicción incorrecta, por lo tanto, el mayor nivel de aleatoriedad no se puede utilizar como el nivel de confianza. Por el contrario, el nivel de aleatoriedad *iid* es un límite superior en el que la probabilidad de las etiquetas $Y|\{y\}$ son correctas. De ello se deduce que el valor de la confianza sea uno menos el segundo mayor valor de aleatoriedad *iid*, definida por:

$$\sup\{1-\varepsilon : |\Gamma^{\varepsilon}| \leq 1\} \tag{2.91}$$

Credibilidad

El mayor nivel de aleatoriedad *iid* o la credibilidad, es una medida de la calidad de los datos disponibles. Cuando la credibilidad es baja, significa que los datos de entrenamiento no son *iid* al azar o que el ejemplo sin etiquetar no es representativo de los datos de entrenamiento. La credibilidad resulta ser claramente una medida importante, ya que es lo más probable que las etiquetas sean rechazadas, la etiqueta preferida aún puede generar una larga secuencia con bajo nivel de aleatoriedad *iid. S*e ha comprobado que la mayoría de la clasificación que ha resultado incorrecta posee baja credibilidad [7].

$$\inf\{\in |\Gamma^{\varepsilon}| = 0 \tag{2.92}$$

2.5.8 Clasificación con Predictores Conformales

En la clasificación, el protocolo de aprendizaje de la figura 2.13 es como se define en la figura 2.12, pero esta vez se incluye no solo las variables de Err_n^{ε}, (el total de número de errores cometidos e incluyendo la prueba n en el nivel de significancia ε) y err_n^{ε} (la variable binaria que muestra si se comete un error en el *n* de prueba), pero también las variables con valores analógicos $Mult_n^{\varepsilon}, mult_n^{\varepsilon}, Emp_n^{\varepsilon}, emp_n^{\varepsilon}$ para múltiples predicciones (conteniendo más de una etiqueta) y vacío (no contiene etiquetas).

$Err_0^\varepsilon := 0, Mult_0^\varepsilon := 0, Emp_0^\varepsilon = 0\ para\ todo\ \varepsilon \in (0,1);$
Ciclo n, 1,2,….:
Salidas reales $x_n \in \boldsymbol{X}$;
Salidas predicciones $\Gamma_n^\varepsilon \subseteq Y\ para\ todos\ \varepsilon \in (0,1)$
$err_n^\varepsilon := \begin{cases} 1\ if y_n \notin \Gamma_n^\varepsilon \\ 0\ Otra\ forma \end{cases}\ para\ todos\ \varepsilon \in (0,1);$
$Err_n^\varepsilon := Err_{n-1}^\varepsilon + err_n^\varepsilon\ para\ todos\ \varepsilon \in (0,1);$
$mult_n^\varepsilon := \begin{cases} 1\ if\ |\Gamma_n^\varepsilon| > 1 \\ 0\ Otra\ forma \end{cases}\ para\ todos\ \varepsilon \in (0,1);$
$Mult_n^\varepsilon := Mult_{n-1}^\varepsilon + mult_n^\varepsilon\ para\ todos\ \varepsilon \in (0,1);$
$emp_n^\varepsilon := \begin{cases} 1\ if\ |\Gamma_n^\varepsilon| = 0 \\ 0\ Otra\ forma \end{cases}\ para\ todos\ \varepsilon \in (0,1);$
$Emp_n^\varepsilon := Emp_{n-1}^\varepsilon + emp_n^\varepsilon\ para\ todos\ \varepsilon \in (0,1);$
Fin Ciclo

Figura 2.13 Protocolo de aprendizaje de error, de múltiple predicción y predicción vacía.

a) Medidas de no conformidad con SVM

Las SVM[13], determinan los casos en los límites de cada clase, y fijan un hiperplano de separación que maximiza el margen entre ellos. Con el fin de construir Predictores Conformales usando SVM (CP-SVM), se utiliza la distancia de cada señal del hiperplano de separación, y la clase a la que pertenece, con el objetivo de producir resultados de no conformidad. Para Y = {-1,1} utilizamos una medida de no conformidad.

$$\alpha_i = -y_i h(x_i) \qquad (2.93)$$

Donde α_i representa la medida de no conformidad de SVM, y $h(x_i)$ es la salida de la SVM para el caso dado (x_i), si la salida del $h(x_i)$ es negativo cuando la instancia pertenece a la clase -1, y positivo si pertenece a la clase 1. Si la predicción es correcta, entonces el mayor valor representa la instancia del hiperplano.

b) Medidas de no conformidad K-NN

El método de los vecinos más cercanos k (k-NN), calcula la distancia de una instancia de prueba respecto de las otras instancias que se proporcionan en el conjunto de entrenamiento, y encuentra su instancia más cercana. La

predicción del algoritmo resulta ser la clase que contiene mayoría de los casos k. En el caso de la construcción de un CP basado en k-NN, utilizamos las distancias de los k casos más cercanos para definir una medida de no conformidad. El enfoque más simple consiste en calcular el total de las distancias de los casos k que pertenecen a la clase de x_i, por ejemplo, cuánto más cerca esté de su clase, será menos extraño a ella. Sin embargo, para que una medida de no conformidad sea más precisa, también se debe tener en cuenta las distancias de los casos k más cercanos que pertenecen a otras clases, ya que cuánto más cerca está la instancia x_i a otras clases, tanto más extraño. La construcción de k-NN-CP con la medida de no conformidad se define como,

$$\alpha_i = \frac{\sum_{j=1,\ldots k} S_{ij}}{\sum_{j=1,\ldots k} 0_{ij}} \qquad (2.94)$$

Donde α_i, es la medida de no conformidad de K-NN y S_{ij} es la j-th distancia más corta de x_i de las instancias de la misma clase, y la 0_{ij} es la j-th distancia más corta de x_i de las instancias de otras clases.

c) Medida de no conformidad "Average"

Este método está basado en una función de la distancia entre medias de los experimentos y está definida por [36]:

$$A(B, z) \coloneqq |(average\ of\ z\ and\ all\ the\ examples\ in\ B) - z| \qquad (2.95)$$

La expresión anterior coloca z en la bolsa, la cual se reduce con la siguiente ecuación,

$$\alpha_i = \left| \frac{\sum_{j=1}^{n} zj}{n} - z_i \right| \qquad (2.96)$$

Donde α_i es la medida de no conformidad de Average, y z_j es el conjunto de las instancias tratadas y z_i es la nueva instancia.

Algoritmo conformal para clasificación

Dada una bolsa $\prec z_1, z_2, \ldots, z_{n-1} \succ$, un sistema de clasificación, una medida de no conformidad A y un nuevo ejemplo x_n para clasificarlos como perteneciente a las clases de 1 a L

Ciclo $(j = 1, \ldots . L)$

Conjunto provisional $z_n = (x_n, C_j)$

Ciclo $(i = 1, \ldots \ldots, n)$

Conjunto $\alpha_i = A(\prec z_1, \ldots \ldots, z_n \succ, z_i\ )$

End ciclo(i)

Conjunto p-value $p_j = \frac{\#\{i=1,\ldots..,n\}|\alpha_i \geq \alpha_n}{n}$

End ciclo(j)

2.6 Transformada de wavelets

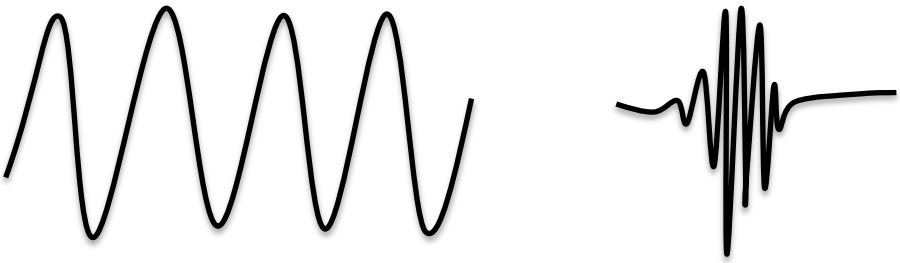
El análisis mediante wavelets representa el paso lógico siguiente a la STFT (Short Time Fourier Transform) [27]: una técnica mediante ventanas con regiones de tamaño variable. El análisis wavelets permite el uso de intervalos grandes de tiempo en aquellos segmentos en los que se requiere mayor precisión en baja frecuencia, y regiones más pequeñas donde se requiere información en alta frecuencia. Lo anterior se muestra de forma esquemática en la figura 2.14

Figura 2.14 Esquema de transformación de la transformada wavelets

Una manera sencilla del modo de operación de esta transformada es pensar que la señal en base del tiempo se procesa mediante por varios filtros pasa-alto y pasa-bajo [30,11]. Esto permite separar las porciones de la señal de alta frecuencia de aquellas otras de baja frecuencia.

2.6.1 Análisis mediante trasformada wavelets

Wavelet es una señal (o forma de onda) de duración limitada cuyo valor medio es cero. Si se comparan las wavelets con las funciones sinusoidales (que son la base del análisis de Fourier), podemos resaltar que la principal diferencia radica en que las señales sinusoidales no tienen duración limitada, son datos que se extienden desde -∞ a +∞. Además, mientras que las señales sinusoidales son suaves y predecibles, las wavelets tienden a ser irregulares y asimétricas como se puede apreciar en la figura 2.15



Sine Wave Wavelet (db10)

Figura 2.15 Ejemplos de señal sinusoidal y señal wavelets

El análisis de señales mediante la transformada wavelets descompone la señal en versiones trasladadas (en tiempo) y escaladas de la Wavelet original mejor conocida como *wavelet madre*.

A partir de las gráficas de la figura 2.15, resulta intuitivo pensar que las señales con cambios bruscos serán mejor analizadas mediante wavelets irregulares que a través de suaves sinusoidales; debido a esto, una de las principales ventajas que provee la transformada de wavelets es su facultad para el análisis de áreas localizadas de señales de gran extensión a lo largo del tiempo.

El análisis mediante la transformada wavelets puede ser aplicado a datos matriciales de dos dimensiones (imágenes), además de datos de mayor dimensión.

2.6.2 Cálculo de la transformada de wavelets

Al aplicar la transformada de wavelets, es importante elegir la *wavelet madre*, la que será la Wavelet madre siendo está la que servirá como prototipo para las ventanas que se emplean en el proceso. Existen varias familias de funciones wavelets que han resultado ser especialmente útiles tal es el caso de las siguientes: Haar, Daubechis, Biortogonal, Coiflets, Symlets, Morlet, Sombrero mexicano o Meyer, entre otras muchas [27].

Pasos para determinar la transformada wavelets de una señal:

Paso 1: Se debe conmenzar con un determinado valor de S (escala), por ejemplo 1, para la señal waveleta, se ubica ésta al comienzo de la señal a analizar (en t=0). Luego, se multiplican entre sí ambas señales y el resultado se integra sobre todo el espacio de tiempo. El resultado de dicha integral se multiplica por el inverso de la raíz cuadrada de s, con el objeto de normalizar la energía y de este modo obtener una función transformada con la misma energía a cualquier escala. Este resultado es el valor de la transformación wavelets con el segmento de la señal original, lógicamente el resultado dependerá de la elección de la función wavelets con el segmento de la señal original. Este paso queda representado en la figura 2.16.

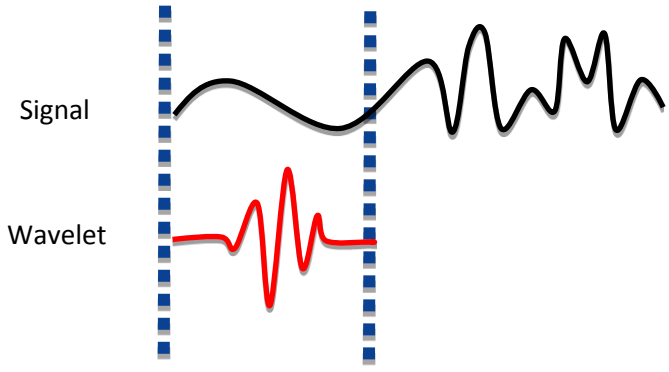


Figura 2.16 Paso 1 para la obtención de la transformada wavelets

Paso 2: La función wavelets (en la misma escala, por ejemplo s=1) se traslada en tiempo (hacia la derecha) un valor dado por τ, y se vuelve a realizar el procedimiento descrito en el paso 1. Se debe repetir este proceso hasta

alcanzar el final de la señal a analizar. Este paso queda ilustrado en la figura 2.17

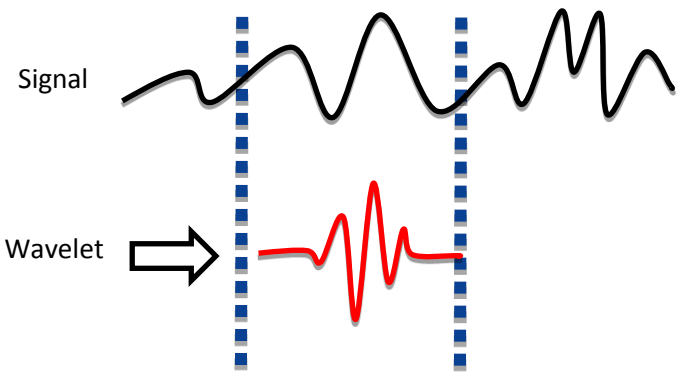


Figura 2.17 Paso 2 para la obtención de la transformada wavelets

Paso 3. Se varía el valor de S (escala) y se vuelven a realizar los pasos 1 y 2 hasta haber barrido todo el rango de frecuencias que se desea analizar. Obsérvese que se trata de una transformación continua, tanto la variación en tiempo como la variación de escala, se deben realizar en forma continua. Sin embargo, si es necesario obtener la Transformada Wavelet por medios computacionales, la condición anterior se reduce a considerar un paso suficientemente pequeño. Cada cálculo para un determinado valor del factor de escala S completa la correspondiente fila de datos del plano tiempo-escala. Este paso se ilustra en la figura 2.18

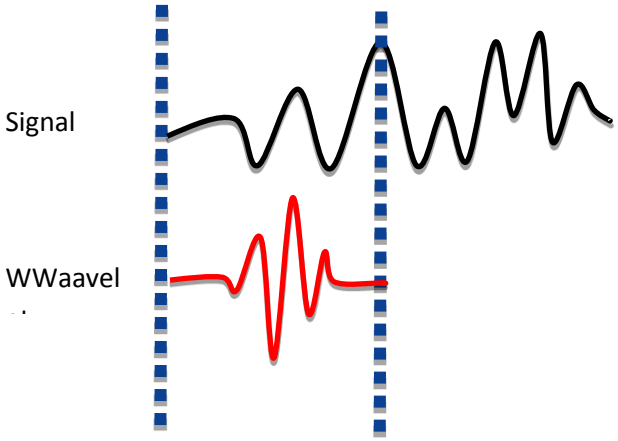


Figura 2.18 Paso 3 para la obtención de la transformada Wavelets

Cuando se finaliza el cálculo para todos los valores de S, se habrá obtenido la transformada wavelets continua de la señal.

2.6.3 Transformada wavelets discreta

Para aplicar la transformada wavelets a una serie de datos numéricos, se hace necesario implementar una transformada discreta. La idea fue desarrollada por [23], quien diseñó un algoritmo basado en un banco de filtros que permite obtener una transformada wavelets en forma instantánea a partir de los datos de interés.

A continuación se describen algunos ejemplos de filtrado, descomposición, determinación de niveles y reconstrucción wavelets.

a. Filtros de un nivel.

La mayoría de las señales son componentes de baja frecuencia que le otorgan a la señal la mayor parte de su información, o bien, le dan una especie de identidad a la misma. Las componentes de alta frecuencia de una señal se encargan de incorporar características muy particulares, y se subdividen en dos categorías.

- Aproximaciones (baja frecuencia)
- Detalles (alta frecuencia)

Es posible separar estas dos componentes a través de filtros. Lo cual puede esquematizarse en el diagrama mostrado en la figura 2.19.

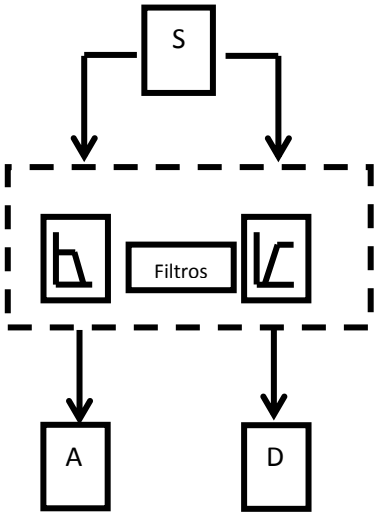


Figura 2.19 Diagrama de descomposición de señales.

Considerando la figura 2.19, S es la señal que se desea analizar, A es la salida del filtrado pasa-bajo y D la salida del filtrado pasa-alto. Naturalmente, los filtros son diseñados de tal manera que sean complementarios, es decir la suma de A y D debe ser S. Si se diseñaran los filtros en forma separada, se perdería información o en caso contrario, se estaría amplificando la banda de entrecruzamiento. Sin embargo, este procedimiento tiene la desventaja que se aumenta al doble el número de datos originales, pues por cada muestra de S se genera un par de muestras (A,D) por lo que el costo matemático y computacional, se incrementa. Para remediar este problema, se propone un método que guarda la mitad de los puntos (A,D), sin perder en ello información de la señal S. Este procedimiento es conocido como submuestreo. La idea se ilustra en la figura 2.20

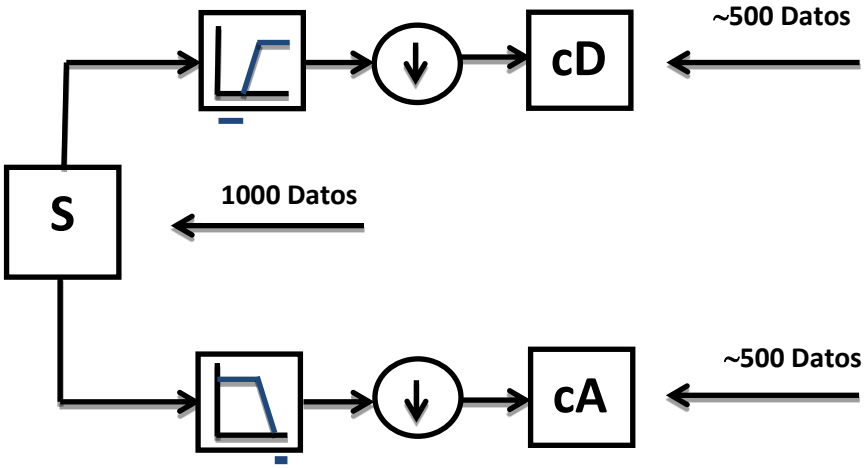


Figura 2.20 Diagrama de descomposición de señales

Los círculos con flechas representan la eliminación de datos o submuestreo. Luego, cD y cA son los nuevos coeficientes obtenidos de la etapa de filtración. Intuitivamente se puede concluir que al tener cD y cA, en conjunto, se tiene la misma cantidad de datos que las de la señal original S, a la vez que se ha mantenido la información necesaria. En la figura 2.20, se ejemplifica la idea de una señal S de 1000 datos, obteniéndose en la salida dos series de aproximadamente 500 datos cada una. El concepto de aproximado, se debe a que el proceso de filtración se realiza a través de convolución de la señal de entrada con la función de transferencia (discreta) del filtro, lo que puede introducir eventualmente una o dos muestras más.

Sin embargo, para muchas señales de mayor complejidad, no basta con dos bandas de frecuencia (alta y baja), sino que más bien debe hacerse una descomposición de más niveles para poder separar las características y poder analizarlas independientemente. Surge la idea entonces de filtros multiniveles.

b. Filtros multiniveles

Para realizar este proceso de filtrado, se aplica el mismo procedimiento a las señales de salida de la primera etapa, y así sucesivamente hasta el nivel de descomposición que se desee. Lo anterior da origen a una descomposición multinivel conocida como ramificación o árbol de descomposición wavelets, como se muestra en la figura 2.21

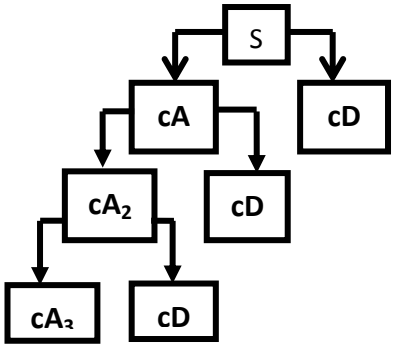


Figura 2.21 Árbol de descomposición wavelets

En la figura anterior podemos notar que cD_1 resulta ser la componente de más alta frecuencia de la señal, y cA_3 la de menor frecuencia. Al ser descompuesta la señal con mayor cantidad de bandas de frecuencia, se posee una información más detallada acerca de S, por lo que esta metodología es conocida como multiresolución. Surge en forma inmediata la inquietud acerca del diseño del algoritmo, relativo al número de niveles a utilizar para analizar la señal en cuestión.

C. Determinación del número de niveles

Como se trata de un proceso recursivo, se podría iterar de forma sucesiva infinitas veces. Sin embargo, en la práctica, sólo se puede descomponer hasta que un intervalo o nivel posea una sola muestra (o píxel en el caso bidimensional, para el análisis de imágenes).

Podría pensarse de forma intuitiva que se obtienen resultados óptimos con un mayor número de niveles tal como se expresa en [24]. Se recomienda una ramificación que esté de acuerdo con la naturaleza de la señal a estudiar, o bien elegir métodos que buscan la descomposición óptima, como por ejemplo, el de la entropía.

D. Reconstrucción wavelets

La reconstrucción de wavelets o transformada inversa de wavelets (discreta), se corresponde con una metodología que sigue el razonamiento en dirección contraria, es decir a partir de los coeficientes cA_i y cD_i (i depende del número de niveles), debe obtenerse S, ver la figura 2.22

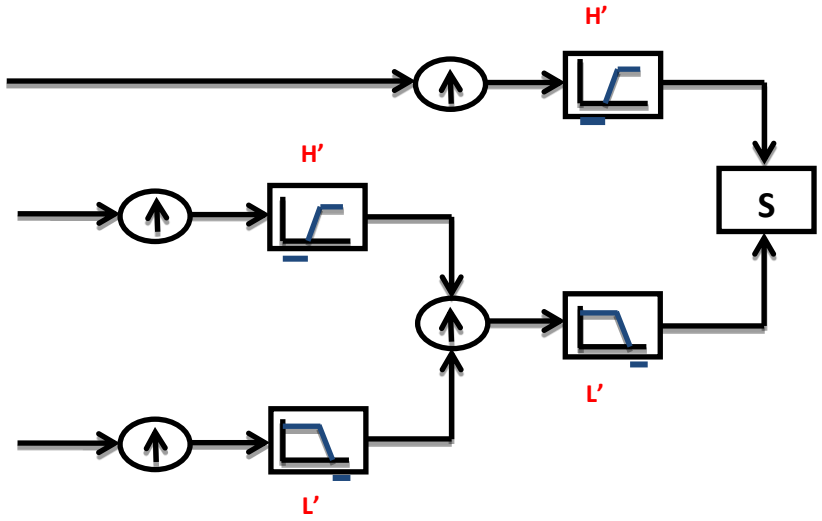


Figura 2.22 Esquema de reconstrucción wavelets

En este caso se debe realizar una sobre-representación de la muestra para compensar el submuestreo realizado en el proceso de descomposición, luego se realiza un filtrado, para finalmente obtener la función original. La etapa crítica en este proceso es el filtrado, pues la elección de los filtros es determinante en la calidad de la reconstrucción. En [24] se discute el diseño de los mismos, introduciendo filtros de descomposición H y L (para pasa-alto y pasa-bajo respectivamente) y sus filtros de reconstrucción correspondientes H' Y L'.

CAPÍTULO 3

3 APLICACIÓN DE LOS PREDICTORES CONFORMALES A SEÑALES DE FUSIÓN Y RESULTADOS

3.1 Introducción

Como se ha visto en el capítulo anterior, existen diversos métodos de clasificación. En este apartado se muestra una aplicación de algunos, entre ellos el multiclasificador de “Predictores Conformales”.

Previo a la aplicación de los métodos, es necesario precisar el origen de las señales, las cuales se han tomado de una base de datos del TJ-II ubicado en el CIEMAT (Madrid, España). El TJ-II es un dispositivo de fusión por confinamiento magnético de tamaño medio de tipo "stellarator" [10] (tipo heliac, $B(0) \leq 1.2$ T, $R(0) = 1.5$ m, $\langle a \rangle \leq 0.22$ m). Este dispositivo funciona en modo pulsado, es decir, su actividad se organiza en pulsos de una duración aproximada de 500 ms. Tras cada pulso, se reconfigura todo el sistema para el siguiente pulso y así sucesivamente. Uno de los sistemas esenciales asociados al TJ-II es el sistema de adquisición de datos [13].

El ciclo de un pulso consiste en una serie de fases, como muestra la fig. 3.1. Existe una fase de puesta a punto del sistema donde se configuran los equipos y se establecen los parámetros de adquisición (como por ejemplo la frecuencia de muestreo o las muestras que se toman por canal). A continuación, durante la descarga, los canales de adquisición muestrean y codifican los valores producidos por los equipos de diagnóstico y generan señales de evolución temporal, cuyos datos son depositados en almacenes temporales locales. Al finalizar la descarga, los canales envían los datos almacenados en sus

almacenes temporales locales al sistema de compresión y almacenamiento de la información. Este sistema es el encargado de unificar toda la información relativa al pulso en una estructura de datos llamada "fichero de descarga" [38], y un segundo nivel, se encarga de comprimir la información. La técnica de compresión utilizada [12] permite recuperar los datos sin ningún tipo de distorsión espectral y en el caso del TJ-II, la tasa de compresión media es superior al 70%. Una vez creado el fichero de descarga es posible realizar los procesos de análisis sobre los datos obtenidos. En la figura 3.1 se muestra el ciclo de un pulso del sistema TJ-II.

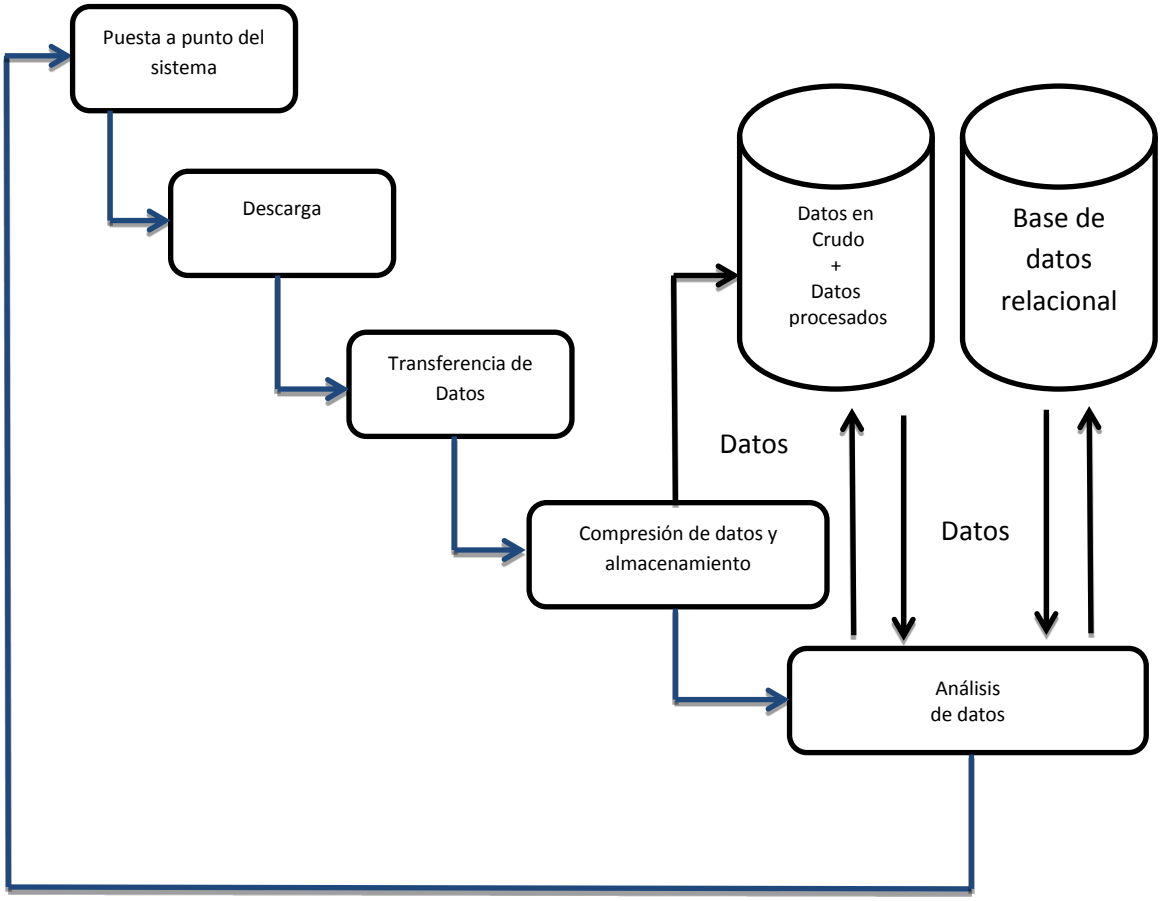


Figura 3.1 Ciclo de un pulso del sistema TJ-II

El TJ-II puede explorar un amplio rango de transformación rotacional (0.9 ≤ι/2p ≤2.2). En el TJ-II los plasmas se producen y se calientan con ECRH (2 girotrones de 300 kW cada uno, 53.2 Ghz, segundo armónico, modo de polarización X), cuyo método de calentamiento es la inyección de ondas electromagnéticas a la frecuencia ciclotrónica electrónica o sus armónicos

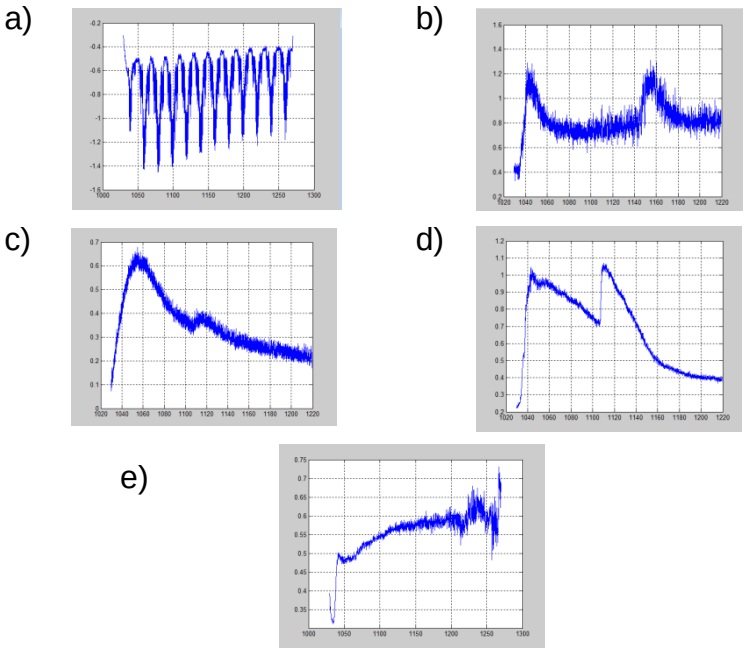
(ECRH son las siglas de su expresión en inglés: Electron Cyclotron Resonance Heating) [5], y con NBI (300 Kw).

Las señales que se obtienen del dispositivo denominadas ‘descargas’, son señales unidimensionales representativas del comportamiento del plasma, por lo que para una descarga en particular, existen diferentes señales asociadas al tipo de sensor analizado (clases), que se presentan en la tabla 3.1y en la figura 3.2, indicando el tipo de señales.

Tabla 3.1 Tipos de señales

Clases de señales	
BOL5	Señal Bolométrica
ECE7	Emisión ciclotrónica
RX306	Rayo x suave
HALFAC3	Hα
DENSIDAD2	Línea media de densidad electrónica

Las regiones de interés de las ‘descargas’ tienen una extensión que oscila entre los 150 y 250 [ms], y dependiendo de la frecuencia de muestreo, el número de muestras comprende de 2000 a 24000 por descarga.



a) ECE7, b) HALFAC3, c) RX306, d) Bol5, e) Densidad2

Figura 3.2 Clases de la tabla 3.1para una descarga en particular

3.2 Preprocesamiento de señales

Para la realización de los experimentos de clasificación, se hizo en primera instancia un preprocesamiento de las señales con la finalidad de eliminar el ruido y poder redimensionarlas, debido a que tienen un tamaño diferente para cada clase, e incluso en el interior de cada clase, las señales provenientes de ese sensor tienen distinta longitud.

Dada la complejidad mencionada, se implementó en Matlab un programa para hacer un ajuste de la dimensión a 16384 muestras, para que todas tengan el mismo tamaño. Además se filtraron las señales con filtros del tipo paso-bajo, con una frecuencia de corte 0.001 y frecuencia de Nyquist sin normalizar, con la finalidad de eliminar ruido de las señales.

Después de realizar el ajuste de las muestras, fue necesario aplicar una transformación de wavelets [23], con el fin de disminuir la dimensión de las señales para su clasificación. Dicha transformada permite comparar la señal con versiones desplazadas y escaladas de la *wavelet madre.*

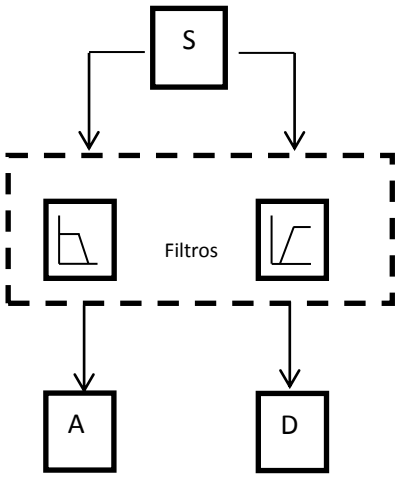


Figura 3.3 Diagrama de descomposición de señales usando bancos de filtros

El filtro que proporciona la transformada de wavelets está ejemplificado en el diagrama de la figura 3.3, donde se muestra que S es la señal que se desea analizar, A es la salida resultante del filtrado pasa-bajo, y D es la salida del filtro paso-alto. Los filtros son diseñados de tal manera que sean

complementarios, es decir, la suma de A y D debe ser S. Un diseño de filtros muy separados provocaría la pérdida de información, en caso contrario se estaría amplificando la banda de entrecruzamiento.

Inicialmente las señales contenían muestras de 2000 a 24000 y después al realizar el redimensionado con el programa Matlab, se normalizaron a 16384. Debido a que la dimensión sigue siendo alta, se decidió utilizar una transformación de wavelets, aplicando un filtro multinivel como se muestra en la figura 3.4, donde cA_i es la componente de la señal de menor frecuencia y cD_i la de más alta frecuencia, para i=1,….,n, siendo n el nivel más alto de descomposición.

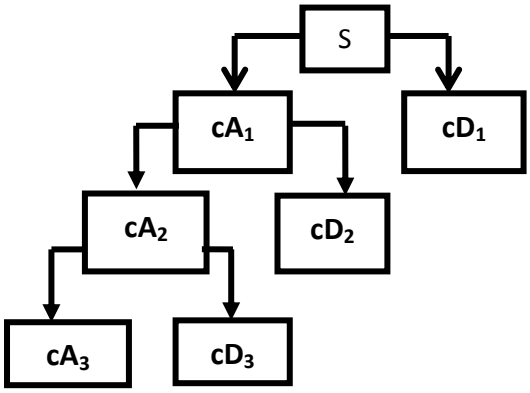


Figura 3.4 Árbol de descomposición wavelets para nivel 3

La figura 3.4 muestra que cD_1 resulta ser la componente de más alta frecuencia de la señal, y cA_3 la de menor frecuencia. Al ser descompuesta la señal en mayor cantidad de bandas de frecuencia, se posee una información más detallada acerca de S. Esta metodología es conocida como multiresolución [23].

Para elegir el nivel de descomposición, se iteró el proceso de filtrado. Es decir, se aplicó el mismo procedimiento a las señales de salida de la primera etapa, y así sucesivamente, como se muestra en la figura 3.4. A las bases de señales seleccionadas y aquí utilizadas, se aplicó una descomposición a diferentes niveles utilizando la función madre ‘Haar’, siendo la función más adecuada para evitar la pérdida de información como se muestra en la tabla 3.2.

Tabla 3.2 Dimensión de las señales y niveles de descomposición

Dimensión de las señales	**Nivel de descomposición wavelets t**		
	Nivel 4	Nivel 7	Nivel 8
16384	1024	128	64

En la figura 3.5, se muestra gráficamente el resultado de las muestras con los diferentes niveles de descomposición.

Señal Original **Nivel 4** **Nivel 7** **Nivel 8**

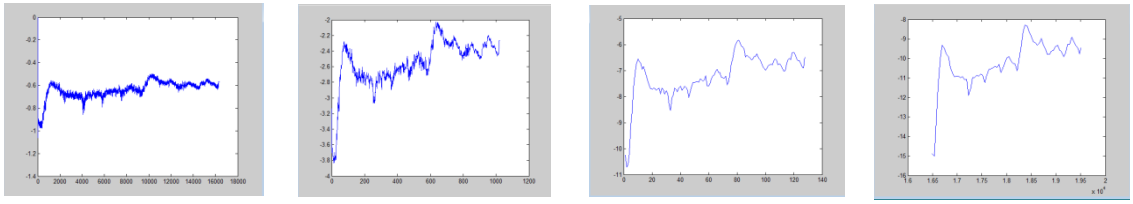


Figura 3.5 Comparativo de señales ECE7 original y con transformada de wavelets en los niveles 4, 7 y 8.

Después de realizar varios experimentos, se decidió utilizar un nivel 8, que reduce las señales a una dimensión de 64 muestras para mantener la información necesaria para la clasificación, debido a que mayores reducciones podrían causar errores en la clasificación por pérdida de información. La aplicación de la transformada de wavelets se muestra en las figuras 3.6 a 3.10, en donde es posible observar la señal original y su descomposición al nivel 8 de cada una de las señales estudiadas.

Clase ECE7_10104 etiqueta 1

Señal Original Transformada de wavelets Nivel 8

Figura 3.6 Comparativo de señales ECE7, original y wavelets nivel 8

Clase HALFAC3_10104 etiqueta 2

Señal Original Transformada wavelets Nivel 8

Figura 3.7 Comparativo de señales HALFAC3, original y wavelets nivel 8

Clase RX306_10104 etiqueta 3

Señal Original Transformada de wavelets t Nivel 8

Figura 3.8 Comparativo de señales RX306, original y wavelets nivel 8

Clase BOL5_10104 etiqueta 4

Señal Original Transformada de wavelets Nivel 8

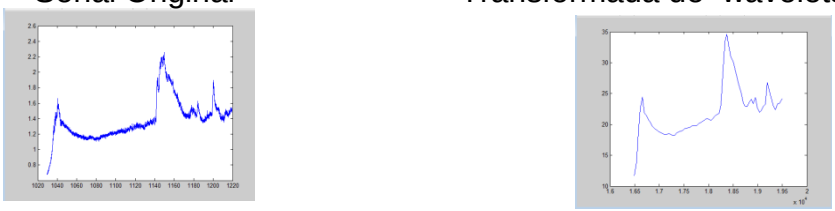


Figura 3.9 Comparativo de señales BOL5, original y wavelets nivel 8

Clase DENSIDAD2_10104 etiqueta 5

Señal Original Transformada de wavelets Nivel 8

Figura 3.10 Comparativo de señales DENSIDAD2, original y wavelets nivel 8

3.3 Clasificación de las señales

La fase de clasificación consiste en aplicar un sistema que divida en varias clases a las señales, según su patrón de comportamiento. Se ha diseñado un algoritmo multiclase para las señales de la tabla 3.1. En primer lugar se han aplicado los métodos de SVM y K-NN para realizar la clasificación.

El procedimiento desarrollado en la herramienta se aplicó al total de las señales, de las cuales se tomó de manera aleatoria el 70% de ellas para su entrenamiento y el 30% para la validación. A continuación se muestra en la tabla 3.3, las señales de estudio.

Tabla 3.3 Señales a clasificar

Señales	Entrenamiento 70%	Validación 30%	Total
ECE7	51	20	71
HALFAC3	49	22	71
RX306	48	20	68
BOL5	51	20	71
DENSIDAD2	49	21	70
Totales	248	103	351

Se realizaron dos experimentos, el primero consistió en probar con 3 clases: ECE7, HALFAC3 y RX306. Se tomaron de manera aleatoria 145 señales para entrenamiento y 62 para validación. El segundo experimento se realizó con la totalidad de las clases, quedando con 248 señales para su entrenamiento y 103 para su validación.

3.3.1 Clasificación de señales con SVM

Para realizar la clasificación en SVM, se ha hecho uso de las librerías de Spider [45] con el método *uno contra el resto*, como se muestra en la figura 3.11.

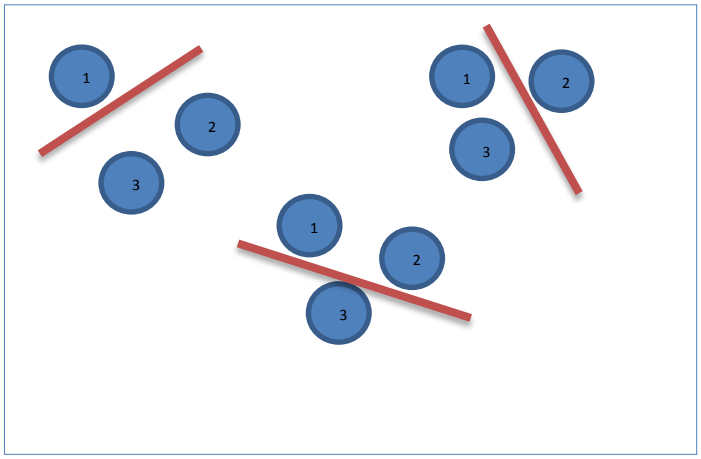


Figura 3.11 Ejemplo de *uno contra el resto* SVM de 3 clases

El método consiste en combinar clasificadores binarios en los que, para poder realizar lo anterior, se requiere definir una matriz con un número de columnas igual al número de clases. Etiquetando los elementos de la clase i-ésima con etiqueta positiva y el resto de las clases, con etiqueta negativa, tal como se expresa en el siguiente ejemplo sencillo

$$Y = \begin{Bmatrix} +\mathbf{1} & -1 & -1 \\ -1 & +\mathbf{1} & -1 \\ -1 & -1 & +\mathbf{1} \end{Bmatrix}$$

donde Y es el conjunto de etiquetas de las clases en las que se deben incluir las muestras a clasificar.

Ajuste de la función núcleo o “kernel”.

Un paso fundamental en la aplicación de las técnicas de SVM es la selección de las funciones núcleo o “kernels”, ya que es lo que permite la transformación a un espacio de características sin necesidad de que se requiera conocer algún algoritmo explícito, y con esto, la posibilidad de manejar datos linealmente no separables de una manera mucha más sencilla en la clasificación.

Existen diversos tipos de “kernels”, que son comúnmente utilizados para diferentes tipos de estudio, cada uno de ellos con una estructura en particular y asociados a ciertos parámetros. Entre los “kernels” más destacados se pueden mencionar: el lineal, el polinomial, el RBF (Función de base radial) y el

sigmoidal. En este trabajo de investigación se utilizaron los siguientes tipos de "kernel" que se muestran en la tabla siguiente.

Tabla 3.4 Tipos de "Kernel" utilizados en SVM

Tipo de "kernel"	Expresión
Lineal	$\boldsymbol{K(x,y) = x^T y}$
Polinomial	$\boldsymbol{K(x,y) = (1 + \langle x,y\rangle)^d}$, con grado d
Función de Base Radial	$\boldsymbol{K(x,y) = \exp^{\{-\|x-y\|/2\sigma^2\}}}$

Los tipos de "kernel" llevan consigo la asignación de valores de parámetros, donde se considera una etapa importante en la aplicación de las SVM. La obtención de buenos resultados de la clasificación, dependerá, en gran parte de la selección de un "kernel" adecuado y, por supuesto de la utilización de valores apropiados de los parámetros de los mismos.

Dentro del algoritmo SVM, se puede cambiar el parámetro C (2.56), el cual hace referencia al valor de penalidad para el error, en otras palabras, es el que va a permitir la holgura para tener cierto error de clasificación a cambio de tener una mejor generalización de los datos; sin embargo, el valor de este parámetro varía de acuerdo a los datos que se desean clasificar, por lo que se recomienda realizar una búsqueda exhaustiva de un valor adecuado que permita clasificar correctamente la mayor cantidad de instancias desconocidas, tanto como sea posible. Se realizaron varias configuraciones con la librería Spider variando los parámetros C y σ (ver tabla 2.1), con la intención de determinar cuál de ellas nos arrojaría mejores resultados de clasificación.

Es importante tener en cuenta que un valor muy elevado de C tiene una alta penalización para puntos no separables y se tiende a tener muchos vectores soporte, lo que puede llevar al sobreajuste y con esto, la mala generalización. Por otra parte, un valor muy pequeño de C, puede hacer que el modelo sea muy rígido, lo que puede conducir a un subajuste.

Tomando en cuenta los valores de C que se utilizaron para medir el desempeño de los tipos de "kernel", con C=1 para todos ellos y para el "kernel"

'RBF' el valor de σ fue probado con 10 y 100, los mejores resultados se obtuvieron con σ=100 y con un "kernel" polinómico de grado 2.

Una vez determinados y ajustados los tipos de "kernel" y sus parámetros, se hizo la clasificación de las clases ECE7, HALFAC3, y RX306 con SVM. En la figura 3.12 se muestran los resultados con los valores de los parámetros antes mencionados.

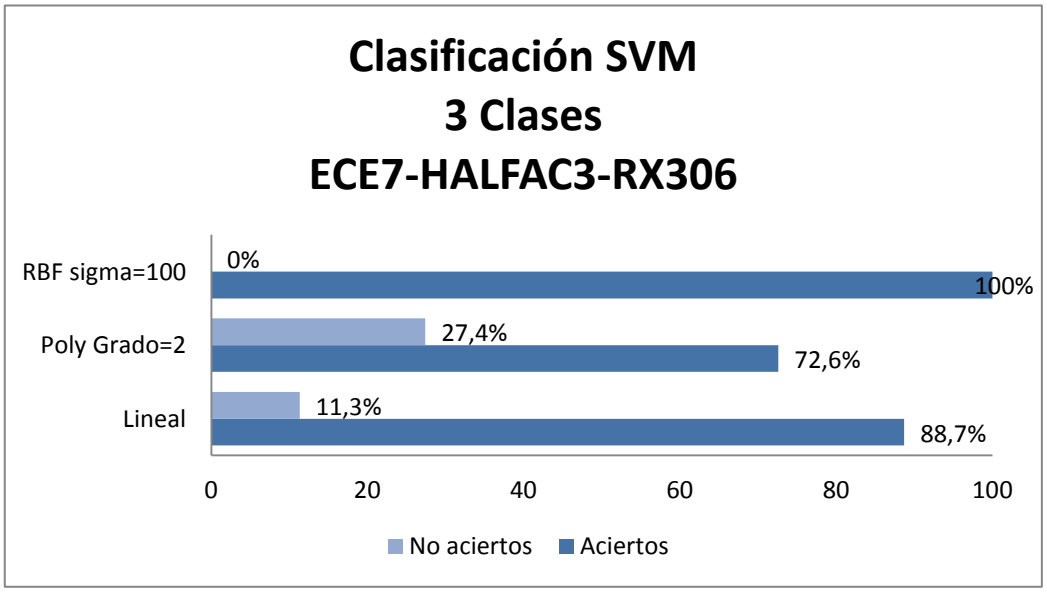


Figura 3.12 Resultados de Clasificación SVM de 3 clases

En la figura 3.12, se observa, al realizar los experimentos que el "kernel" polinómico de grado=2, ha sido el que presentó resultados más bajos para clasificar con un porcentaje del 72,6% de aciertos y 27,4% de no aciertos. En el caso Lineal, se obtuvo un 88,7% de aciertos y 11,3% de no aciertos, mientras que en el caso del RBF con σ =100, se obtuvo un acierto del 100%.

En el caso de la clasificación de 5 clases, ECE7,HALFAC3,RX306, BOL5,DENSIDAD 2, los resultados obtenidos se muestran en la figura 3.13.

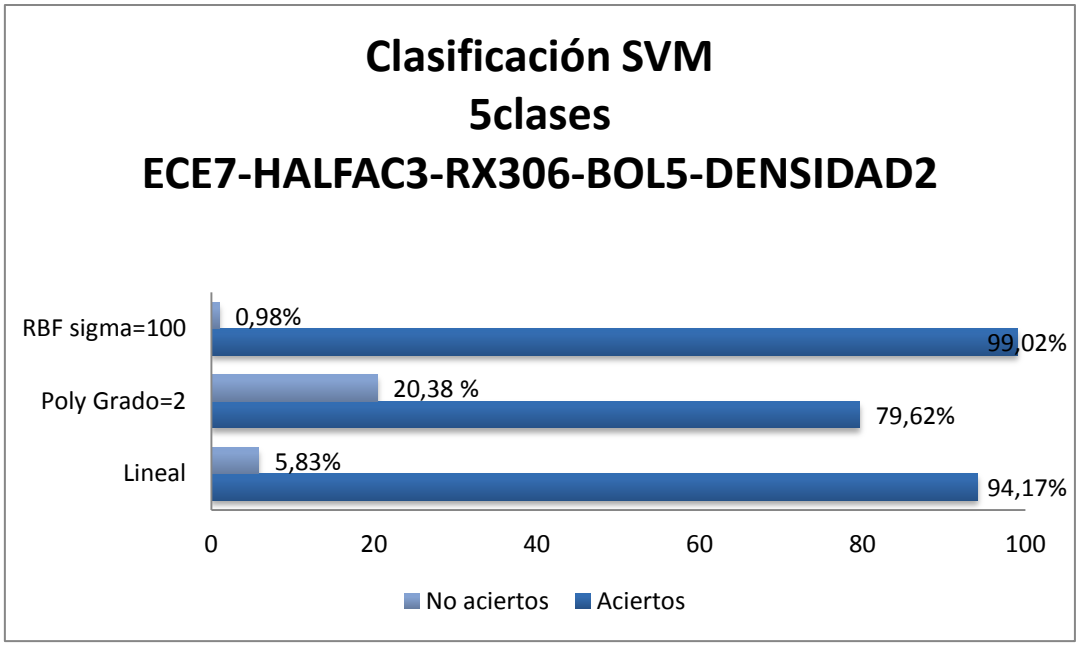


Figura 3.13 Resultados de Clasificación SVM de 5 clases

Como podemos observar en la figura 3.13, se muestran los resultados con la mejor clasificación en SVM con el "kernel" 'RBF' y σ=100, en donde del total del conjunto de validación con 103 señales, se clasificaron 102, que representa un 99.05% y una señal no clasificada que representa un 0.97%; mientras que con el "kernel" Lineal, se clasificaron 97, que representa un 94.17%, mientras que 6 no se clasificaron, dando un porcentaje de 5.83%. Finalmente se observa que el "kernel" polinómico de grado 2, resultó ser el más bajo, pues sólo pudo clasificar el 79.61% de las señales.

3.3.2 Clasificación de señales con el Método de Vecinos Cercanos K-NN

El método calcula la distancia de una señal de validación con respecto de las otras señales que se proporcionan en el conjunto de entrenamiento, y encuentra su señal más cercana, ver sección (2.5.8). Se realizaron pruebas con K=3. Los resultados de las clases ECE7, HALFAC3 y RX306, con este tipo de clasificador son mostrados en la figura 3.14.

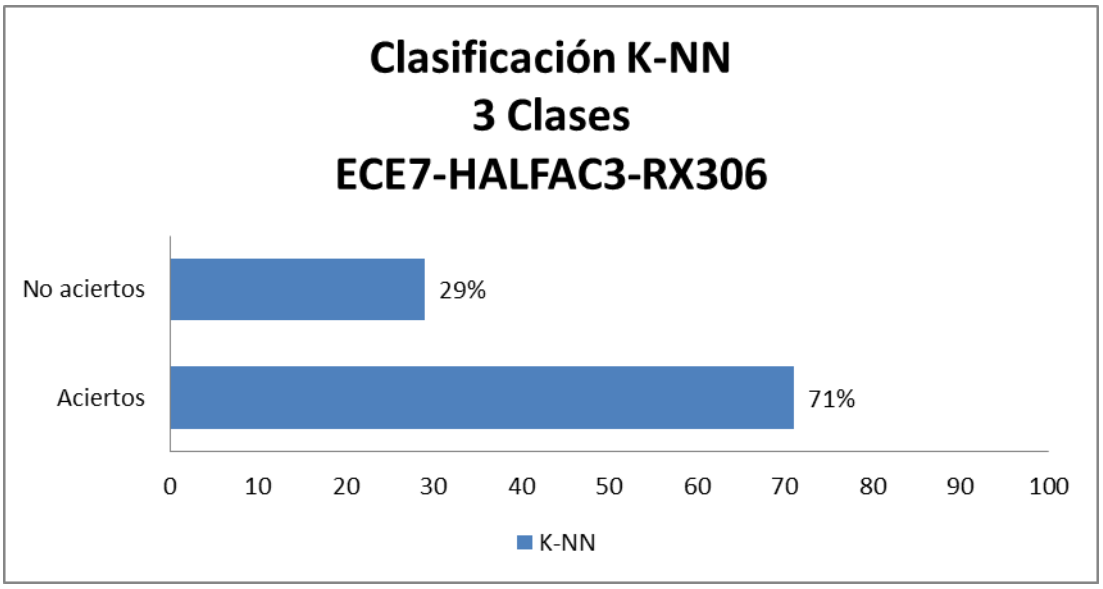


Figura 3.14 Resultados de la clasificación K-NN de 3 clases

La herramienta K-NN, clasificó las señales con un 71% de aciertos y un 29% de errores. En el caso de las 5 clases ECE7, HALFAC3, RX306, BOL5 y DENSIDAD2, se evaluaron con K= 3, la gráfica 3.15 muestra los resultados de la clasificación.

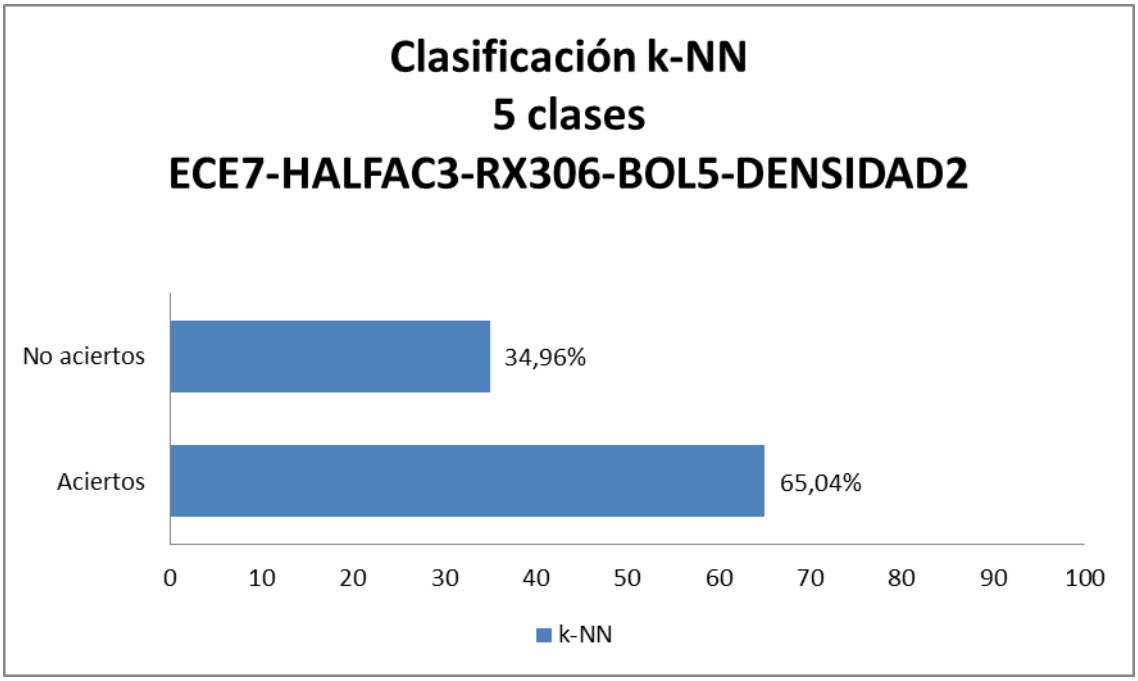


Figura 3.15 Resultados de la clasificación K-NN de 5 clases

En la clasificación de las 5 clases, se puede observar aciertos del 65,04%, y un 34.96% de no aciertos. Es posible apreciar claramente que la clasificación con SVM logra tener mayor porcentaje de aciertos, por encima de K-NN.

Después de haber analizado las señales y hacer su clasificación con los métodos SVM y K-NN, nos enfocaremos ahora en lo que es el método de Predicción Conformal CP, donde se utilizaron los métodos de clasificación SVM, K-NN como algoritmos subyacentes para realizar medidas de no conformidad, además de utilizar el método de “Average”, método propuesto en el artículo sobre Predicción Conformal [41].

3.4 Aplicación de predictores conformales a la clasificación

La herramienta computacional desarrollada en este estudio está programada en Matlab con el uso de librerías de Spider [44]. En la figura 3.16 se muestra el interfaz gráfica del multiclasificador para las señales de fusión.

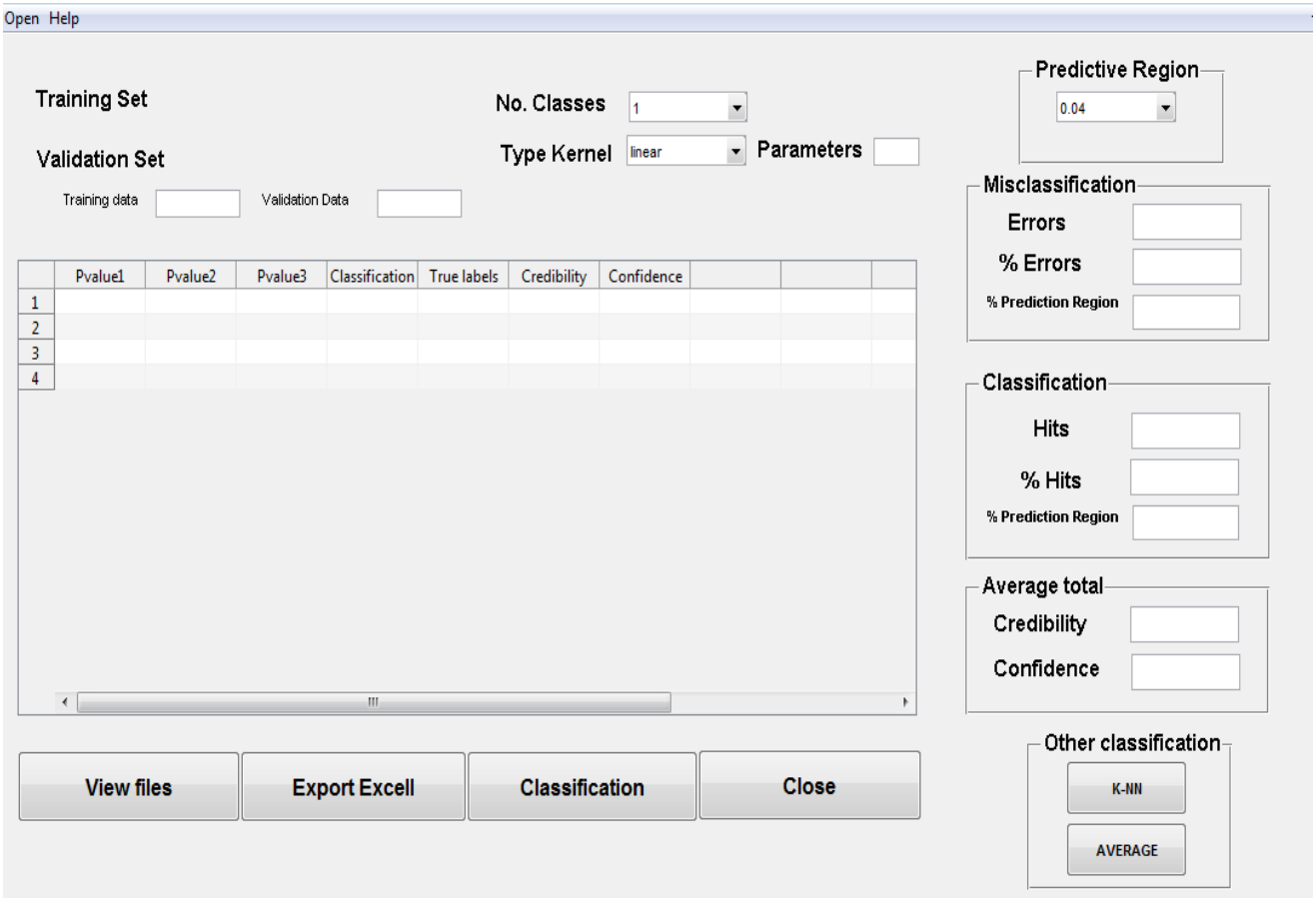


Figura 3.16 Interfaz gráfica del multiclasificador de Señales de Fusión

Como se puede observar en la figura 3.16, en primer lugar en el entorno gráfico principal se puede realizar la predicción con SVM, por lo que existen diversas opciones para el usuario, que se describen a continuación.

"Training Set", se seleccionan los archivos para entrenamiento y validación, dichos archivos tienen extensión .txt, las primeras columnas de los archivos tienen los datos a clasificar y dependiendo del número de clases, las últimas columnas variarán, es decir, si la clasificación es de 2 clases, entonces serán 2 columnas las que contendrán las clases.

"Validation Set" se seleccionan los archivos para validación, las primeras columnas tendrán los datos a clasificar, y en la última columna contendrán las clases verdaderas.

"No. Classes" el usuario podrá seleccionar el número de clases con los que estamos trabajando.

"Type Kernel" se podrá seleccionar el tipo de "kernel" que el usuario desee utilizar: Lineal, Polinomico y RBF.

"Parameters" se seleccionará el parámetro a utilizar en el tipo de "Kernel".

"Predictive Region", el usuario puede escoger la región que desea realizar a las pruebas, es decir si se quiere obtener un 95% de confianza, el 90%, etc.

"Misclassification", nos muestra el número de errores, y la tasa de error, además de mostrar la región de confianza de la mala predicción.

"Clasification", nos muestra el número de aciertos, la tasa de aciertos, y la región obtenida en la predicción.

"Average Total", da el promedio de la credibilidad y la confianza obtenidas en la predicción.

Además cuenta con los botones siguientes: **"View Files",** donde podemos ver los archivos de validación y entrenamiento; en ***"Export Excel",*** se exportan en archivos de Excel, los resultados como los "p-value", predicción,

credibilidad y confianza, además de las regiones de confianza escogida por el usuario; en ***"Classification",*** una vez que el usuario ha determinado el tipo de "kernel", junto con el resto de parámetros se muestran en la tabla los "p-value" de cada una de las clases, así como la predicción, las etiquetas verdaderas, la credibilidad y la confianza. Finalmente, con el botón ***"Close"***, se cierra el entorno.

Por último, en el apartado de **"Other classification"**, además de realizar la clasificación por SVM, el usuario podrá realizar la predicción con el método de K-NN y el método **"Average"**, por lo que al pulsar en el botón **"K-NN"**, mostrará la ventana de la figura 3.17.

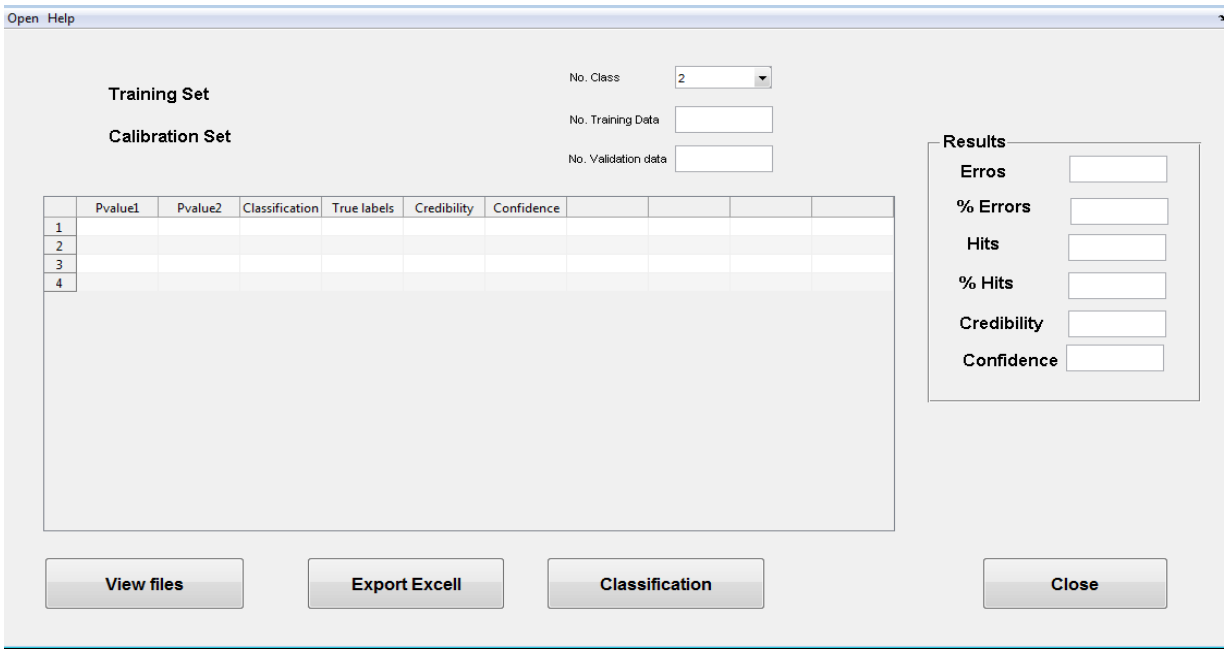


Figura 3.17 Interfaz para la realización de la predicción con el método K-NN.

Como podemos observar en la figura 3.17, las opciones son muy similares a la interfaz principal de la herramienta computacional, donde la única variación es que no se obtienen las regiones de confianza, añadiéndose **"No. Neighbours",** donde el usuario podrá escoger el número de vecinos para realizar la predicción.

En **“Average”,** al pulsar esta opción se mostrará la siguiente ventana:

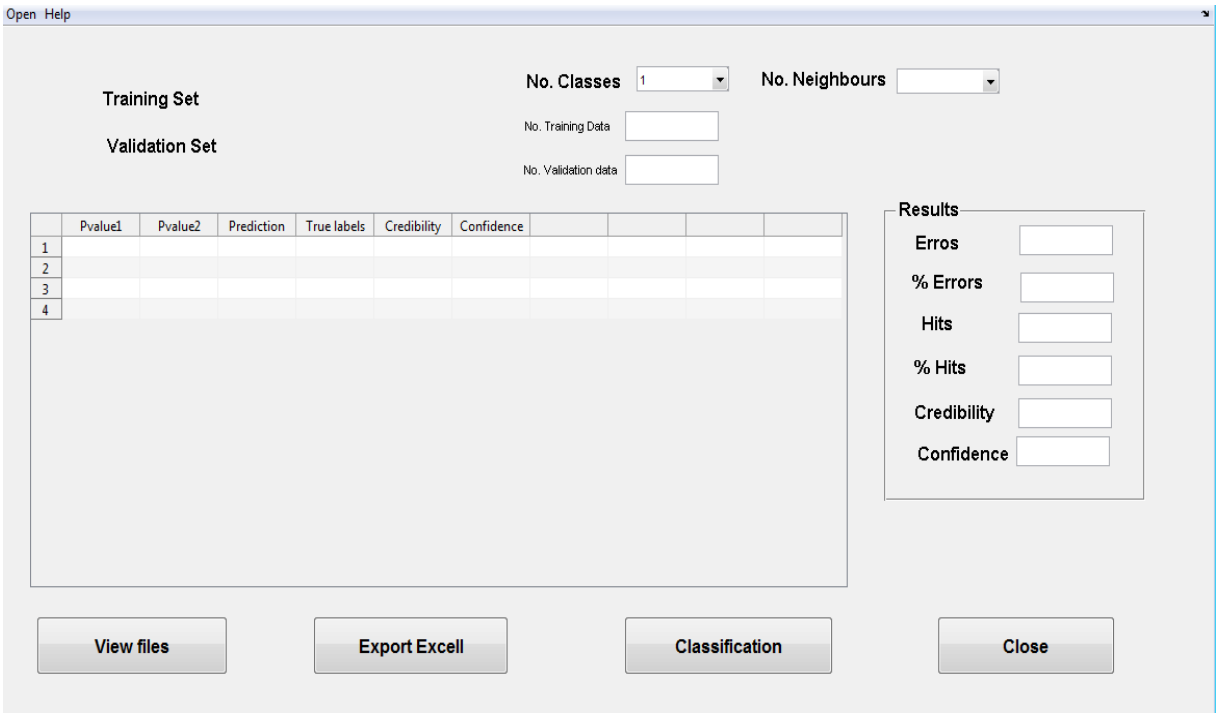


Figura. 3.18 Interfaz para la realización de la predicción con el método “Average”

Salidas de la herramienta computacional.

La herramienta desarrollada, además de mostrar en línea los resultados de la interfaz, proporciona salidas también en Excel de las medidas de no conformidad (“p-value”) de cada una de las señales, así como los valores de confianza y credibilidad.

En la figura 3.19, se muestra en las 3 primeras columnas los “p-value” de cada una de las clases, en la cuarta columna la predicción, en la quinta columna las etiquetas verdaderas de las señales, y en la penúltima columna la credibilidad obtenida por medio de la ecuación (2.91) y en la última columna la confianza obtenida con la ecuación (2.92).

Pvalues1	Pvalues2	Pvalues3	Prediction	True labels	Credibility	Confidence
0.0134	0.5638	0.0470	2	2	0.5638	0.9530
0.0067	0.5705	0.0671	2	2	0.5705	0.9329
0.0134	0.1812	0.5034	3	3	0.5034	0.8188
0.0134	0.2013	0.5168	3	3	0.5168	0.7987
0.0134	0.1812	0.5101	3	3	0.5101	0.8188
0.4027	0.0067	0.0134	1	1	0.4027	0.9866
0.4228	0.0067	0.0134	1	1	0.4228	0.9866
0.4228	0.0067	0.0134	1	1	0.4228	0.9866
0.0067	0.5235	0.0738	2	2	0.5235	0.9262
0.0134	0.5705	0.0537	2	2	0.5705	0.9463
0.0134	0.5839	0.0604	2	2	0.5839	0.9396
0.0336	0.2148	0.5034	3	3	0.5034	0.7852
0.0067	0.0403	0.5906	3	3	0.5906	0.9597
0.0134	0.1745	0.5436	3	3	0.5436	0.8255
0.4228	0.0134	0.0067	1	1	0.4228	0.9866
0.5101	0.0134	0.0067	1	1	0.5101	0.9866

Figura 3.19 Ejemplo de resultados de 3 clases de señales

3.5 Medidas de confianza y credibilidad de la clasificación

Después de describir brevemente los procedimientos integrados en la herramienta. A continuación se muestra la aplicación de los Predictores Conformales para los clasificadores de SVM con diferentes tipos de “kernel”, y para KNN, para obtener las medidas de no conformidad, la predicción, la confianza y la credibilidad, además del procedimiento de medidas de no conformidad “Average” propuesto en [42].

De igual forma a la llevada a cabo en la clasificación, se utilizaron las mismas señales para obtener la predicción, credibilidad y confianza. A continuación se muestran los resultados obtenidos para 3 clases con 145 señales para entrenamiento y 62 para validación y para 5 clases con 248 señales de entrenamiento y 103 señales para validación respectivamente. En ambos casos, las señales se escogieron de manera aleatoria.

3.5.1 Regiones de Confianza

Al obtener las regiones de confianza, se puede observar que cuanto mayor es el porcentaje para un nivel de confianza, la predicción resulta ser la más exacta posible.

En la tabla 3.5 se muestran los resultados para 3 niveles de confianza, del 95%, 85% y 75%. El valor que se indica para el método de clasificación, representa la tasa de error que nos permite comprobar el comportamiento de la predicción y mostrar qué tan seguro es la eficiencia del algoritmo.

Tabla 3.5 Niveles de confianza de las tasas de error

CP-CLASIFICADORES	**Nivel de Confianza**		
	95%	**85%**	**75%**
CP-SVM-LINEAL	3.22%	0	3.22%
CP-SVM-POLINOMIAL GRADO 2	1.62%	3.22%	0
CP-SVM-RBF-σ=100	0	0	0
CP-K-NN	3.22%	11.29%	8.06%
CP-AVERAGE	6.45%	8.06%	19.35%

La tasa de error se calcula como la tasa de las regiones de predicción que no contienen la etiqueta real de las señales. Se puede apreciar que en la región del 95%, el único que no presentó ningún error fue CP-SVM-RBF.

Por otro lado, en la tabla 3.6, podremos observar la tasa de aciertos de todos los Predictores Conformales con los mismos porcentajes para los niveles de confianza utilizados previamente.

Tabla 3.6 Niveles de confianza de las tasas de acierto

CP-CLASIFICADORES	**Nivel de Confianza**		
	95%	**85%**	**75%**
CP-SVM-LINEAL	48.38%	24.19%	14.52%
CP-SVM-POLINOMIAL GRADO 2	43.54%	11.30%	6.45%
CP-SVM-RBF-σ=100	50%	14.52%	0
CP-K-NN	37.09%	8.06%	9.67%
CP-AVERAGE	46.77%	6.45%	12.90%

3.5.2 Credibilidad

Otra de las ventajas o medidas de confianza que se pueden extraer mediante la aplicación del método de Predictores Conformales, es la obtención de la credibilidad que, como se había mencionado, es el mayor “p-value” que tomamos para predecir la etiqueta correcta.

En la tabla 3.7 se muestran los valores en promedio de la credibilidad y la confianza encontradas en todos los predictores conformales con sus correspondientes algoritmos de aprendizaje. Se muestran los resultados para 3 clases: ECE7,HALFAC3,RX306, con un conjunto de entrenamiento de 145 señales y otro de validación con 62 señales.

Tabla 3.7 Comparación de resultados de los 3 algoritmos para 3 clases

CP-CLASIFICADORES	**Predicción Conformal**		
	P_i		
	Predicción	**Credibilidad**	**Confianza**
CP-SVM-LINEAL	**90%**	**48.81%**	**92.41%**
CP-SVM-POLY GRAD0=2	**76%**	**47.55%**	**82.36%**
CP-SVM-RBF SIGMA=100	**100%**	**43.55%**	**91.22%**
CP-KNN	**61%**	**63.11%**	**87.29%**
CP-AVERAGE	**63%**	**23.75%**	**91.35%**

A partir de los resultados mostrados en la tabla 3.7, se puede observar la precisión alcanzada por los tres algoritmos citados, donde los predictores conformales (P_i) del conjunto de validación, no son incluidos en el conjunto de entrenamiento. La diferencia de la exactitud de cada método con el correspondiente predictor conformal no es tan significativa como se esperaba. Sin embargo, sí se observa que se están produciendo predicciones más informativas.

En promedio, se puede observar que CP-SVM-LINEAL ofrece la mejor precisión resultando ser del 92.41% de confianza con una credibilidad del 48.81%; si observamos en el caso de CP-SVM-RBF sigma=100, su predicción es del 100%, con una confianza del 91.22% y una credibilidad del 43.55%; mientras que con CP-SVM Polinómico Grado=2, nos proporciona una

predicción del 76%, con una credibilidad del 47.55% y una confianza del 82.36%, en tanto que con vecinos más cercanos, su predicción es del 61% con una confianza del 87.29% y una credibilidad del 63.11%.

Finalmente, con “Average”, se obtuvo una confianza del 91.35% y una credibilidad del 23.75%, con una predicción del 63%. Podemos concluir que las precisiones que se muestran son muy similares debido a que los conjuntos de entrenamiento tienen más o menos las mismas muestras,

Sin embargo, lo que se está interpretando son los resultados promediados de cada una de las predicciones, credibilidad y confianza, por lo que para ver el comportamiento de cada uno de los “p-value” del conjunto validación de una manera individual, realizamos un aprendizaje on-line, con el fin de asegurar que los datos sean ***iid,*** (independientes, e idénticamente distribuidos), este proceso se muestra en el diagrama siguiente:

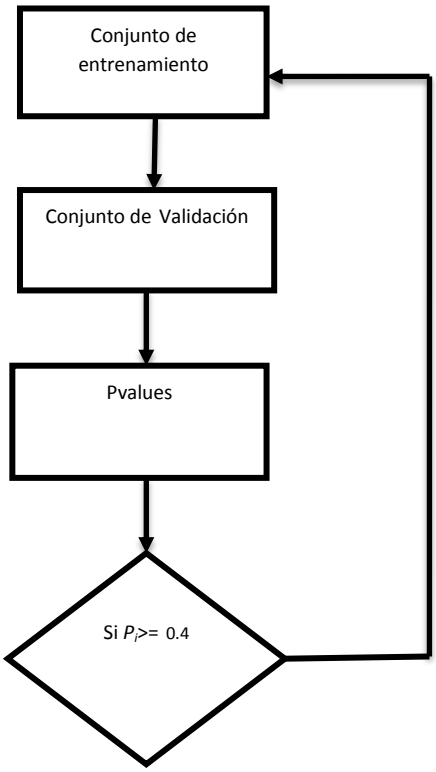


Figura 3.20 Diagrama para evaluar la credibilidad de Predictor Conformal

De acuerdo al diagrama de la figura 3.20, se utilizó el parámetro P_i>= 0.4, considerando que la credibilidad en promedio nos muestra un comportamiento de la credibilidad por encima del 0.4; en la tabla 3.8 se muestran las precisiones alcanzadas con dicho aprendizaje.

Tabla 3.8 Precisiones alcanzadas con Predicción Conformal para 3 clases

CP	Predicción Conformal		
	P_i >= 0.4		
	Predicción	Credibilidad	Confianza
CP-SVM-LINEAL	87%	67%	72%
CP-SVM-POLY GRAD0=2	71%	66%	66%
CP-SVM-RBF SIGMA=100	87%	65%	71%
CP-KNN	66%	50%	92%
CP-AVERAGE	62%	22%	90%

Por otra parte, cuando integramos los predictores conformales (P_i) del conjunto de validación en el conjunto de entrenamiento de forma “on-line” y en el caso de que el “p-value” sea mayor o igual a 0.4, en promedio se puede observar en la tabla 3.8 que el CP-SVM-Lineal ofrece la mejor precisión con el 87% de predicción, una confianza del 72% y una credibilidad del 67%. Con el conjunto CP-SVM-RBF, la predicción es del 87% con una confianza del 71% y una credibilidad del 65%. En el caso Polinómico, se alcanzó una predicción del 71%, con la confianza y la credibilidad del 66% en ambos casos; para el CP-KNN, se alcanzó una predicción del 66%, con una confianza del 50% y una credibilidad del 92% y finalmente en cuanto al CP-AVERAGE, se obtuvo una predicción del 62%, un 22% de confianza y una credibilidad del 90%.

Podemos observar que las precisiones alcanzadas variaron de manera significativa, pues al realizar dicho aprendizaje, la confianza de cada una de las predicciones, cayó dramáticamente, esto se debe a que algunos “p-value” de las clases obtuvieron los mismos valores. Como se puede observar en la tabla 3.9, al obtener una credibilidad del 100%, obtenemos un 0% en la confianza.

Tabla 3.9 Algunos resultados de la credibilidad de predicción de 3 clases.

46	Pvalues1	Pvalues2	Pvalues3	Prediction	true labels	Credibility	Confidence
47	0.0103	0.3093	0.5231	3	2	0.5231	0.6907
48	0.0667	0.2959	0.9388	3	3	0.9388	0.7041
49	0.1684	0.6396	0.9645	3	3	0.9645	0.3604
50	0.3503	0.6970	0.8939	3	3	0.8939	0.3030
51	0.6364	0.1256	0.0151	1	1	0.6364	0.8744
52	0.6533	0.1200	0.0200	1	1	0.6533	0.8800
53	0.6250	0.3433	0.3085	1	1	0.6250	0.6567
54	0.1343	1	0.3267	2	2	1	0.6733
55	0.1337	1	0.3498	2	2	1	0.6502
56	0.2512	1	0.4020	2	2	1	0.5980
57	1	1	1	1	3	1	0
58	0.2341	1	1	2	3	1	0
59	1	0.3430	0.3333	1	1	1	0.6570
60	1	0.3558	0.3413	1	1	1	0.6442
61	1	1	0.4498	1	2	1	0
62	0.1483	1	0.5000	2	2	1	0.5000

La muestra no. 57 de la tabla 3.9, nos indica que ha existido un empate en las tres clases, por lo que la confianza nos da 0, de forma que al realizar el promedio de todas las muestras la confianza baja considerablemente, esto se observa claramente en las gráficas de la figura 3.19 a la figura 3.26.

A continuación, en la tabla 3.10 se muestra la comparación para el total de las 5 clases a clasificar ECE7, HAALFAC3, RX306, BOL5 y DENSIDAD2, con un conjunto de entrenamiento de 248 descargas y un conjunto de validación de 103 descargas.

Tabla 3.10 Comparación de resultados de 5 clases de los tres algoritmos

CP's	**PREDICCION CONFORMAL**					
	P_i			P_i >= 0.4		
	Predicción	**Credibilidad**	**Confianza**	**Predicción**	**Credibilidad**	**Confianza**
CP-SVM-LINEAL	**94.17%**	**61%**	**88%**	**89.32**	**72.31%**	**69.55%**
CP-SVM-POLY GRAD0=2	**79.61%**	**57%**	**78%**	**84%**	**66%**	**66%**
CP-SVM-RBF SIGMA=100	**99%**	**50%**	**91%**	**93.13%**	**65%**	**69%**
CP-KNN	**65%**	**64%**	**85%**	**64%**	**64.18%**	**80.10%**
CP-AVERAGE	**70%**	**21%**	**94%**	**73.78**	**39%**	**81.65%**

Se pueden observar en la tabla 3.10, los resultados de las 5 clases, cuando no se integra los predictores conformales ($\boldsymbol{P_i}$) del conjunto de validación al conjunto de entrenamiento. Se muestra que, en promedio, el CP-SVM-RBF ahora es el que proporciona una mejor predicción, resultando ser del 99%, con una confianza del 50% y un 91% de credibilidad; mientras que el CP-SVM-LINEAL obtiene una predicción del 94.17%, con una confianza del 61% y una credibilidad del 88%; el CP-SVM-Polinómico alcanzó una predicción del 79.61% con una confianza del 57%, y una credibilidad del 78%; en el caso del CP-KNN observamos que cuenta con una predicción del 65%, una confianza del 64% y una credibilidad del 85%, y por último el CP-Average, alcanzó una predicción del 70%, una confianza del 94% y credibilidad de sólo el 21%. Como puede observarse en la clasificación de 5 clases, la predicción resultó ser más baja, pero, sin embargo nos mostró una precisión más alta en la credibilidad, como se mencionó previamente, los resultados se están proporcionando de manera conjunta, al promediar tanto la confianza como la credibilidad, y por supuesto la baja predicción se debe al hecho de que cuántas más clases existan más difícil resulta la clasificación.

De igual manera, en las tres últimas columnas de la tabla 3.10, se observa que cuando los predictores conformales ($\boldsymbol{P_i}$) del conjunto de validación son incluidos en el conjunto de entrenamiento, en este caso con $\boldsymbol{P_i \geq 0.04}$, se obtienen valores distintos. En el caso de CP-SVM-Lineal, se obtiene una predicción del 89.32% con una confianza del 72.31% y un 69.55% de credibilidad; mientras que el CP-SVM-Polinómico muestra una predicción del 84% con una confianza y credibilidad del 66%. En esta ocasión podemos ver que el CP-SVM-RBF nos da una mejor predicción siendo del 93.13% con una confianza del 65% y una credibilidad del 69%; en el caso de CP-KNN nos proporciona una predicción del 64%, con un 64.18% de confianza y una credibilidad del 80.10% y por último el CP-Average nos predice un 73.78% con un 39% de confianza y una credibilidad del 81.65%, es claro observar que la precisión resulta ser más alta en la credibilidad, por el hecho de que nos indica que los valores del conjunto de validación son muy similares al conjunto en entrenamiento.

Se puede concluir a partir de las tablas anteriormente presentadas, que existen diferencias entre incluir o no el conjunto de validación de forma “on-line” en el conjunto de entrenamiento, ya que al hacer las predicciones se obtiene un conjunto de múltiples predicciones con más de una etiqueta por predicción. Esto no se considera un error, sino simplemente indicios de que la cantidad de información no es suficiente para obtener una buena decisión del nivel de confianza seleccionado. Es por ello que al hacerlo, y forzar a que el “p-value” tanto de la predicción de la etiqueta como la credibilidad, sea mayor que p >=0.4, hace que exista un empate entre ellas y considerando que la confianza es 1-ε, ésta tiende a bajar como lo observaremos en las gráficas ilustradas en las figuras de la 3.21 a la 3. 27.

En las figuras 3.21 y 3.22, se muestran los resultados del CP-SVM-LINEAL, que nos dio la mejor predicción para las tres clases: ECE7,HALFAC3,RX306, donde la credibilidad se mantiene baja de manera individual, mientras que la confianza se mantiene alta, por lo que el promedio de la credibilidad es del 43.55%, con una confianza en promedio del 91.22%.

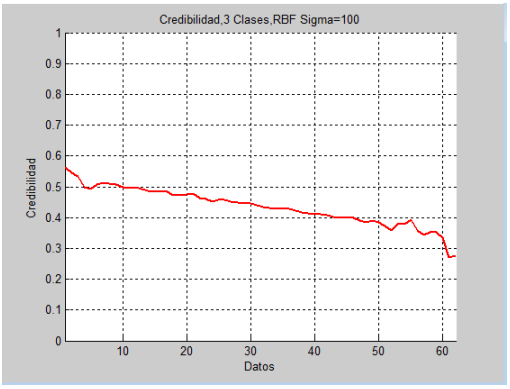


Figura 3.21 Credibilidad RBF σ=100, para 3 clases

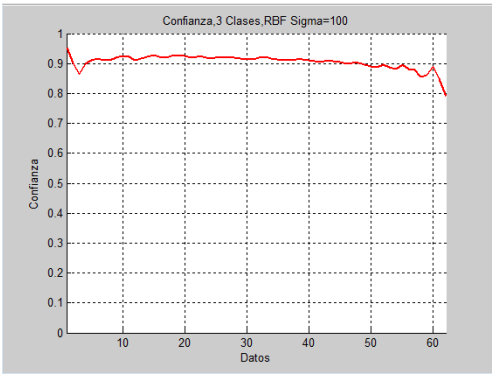


Figura 3.22 Confianza RBF σ =100, para 3 clases

En las gráficas de las figuras 3.23 y 3.24 se muestra el comportamiento de la credibilidad y confianza de las clases: ECE7,HALFAC3,RX306. Se puede observar claramente en estas gráficas que mientras la credibilidad tiende a aumentar, la confianza tiende a disminuir, al realizar el aprendizaje “on-line”. Si

observamos la figura 3.23 la credibilidad comienza por encima del 0.5 y finaliza con una credibilidad en la última muestra con un valor de la unidad, lo cual indica que efectivamente los datos no son extraños. Sin embargo, en la gráfica de la figura 3.24 donde se muestra la confianza, aparece un pico al final, esto significa que existe un empate, pues la confianza en ese punto es nula, por lo que nos da un promedio de la credibilidad del 65% y la confianza del 71%.

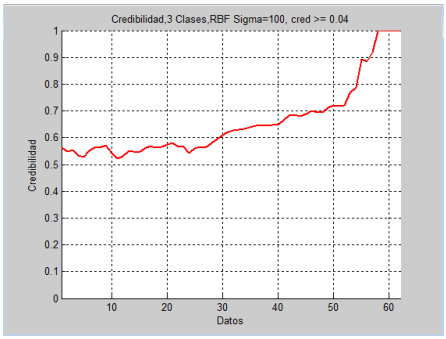


Figura 3.23 Credibilidad RBF σ =100, para 3 clases

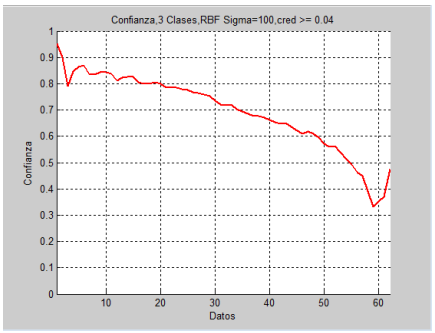


Figura 3.24 Confianza RBF σ =100, para 3 clases

En las Figuras 3.25 y 3.26, se muestra el comportamiento de las cinco clases, donde podemos observar en la figura 3.25 que la credibilidad comienza con valores por encima de 0.5, sin embargo baja dramáticamente al final, mientras que en la figura 3.26 la confianza comienza por encima de 0.9 pero bajando al término de las muestras, por lo que resulta un promedio de 50% de la credibilidad y un 91% de confianza.

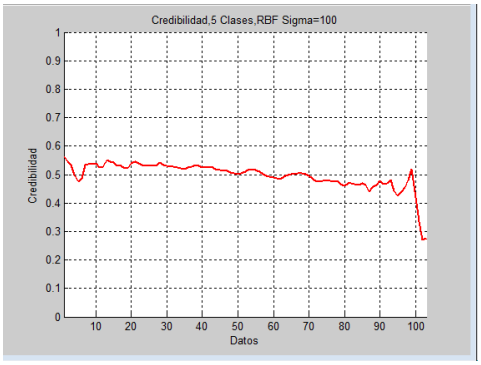


Figura 3.25 Credibilidad RBF σ =100, para 5 clases

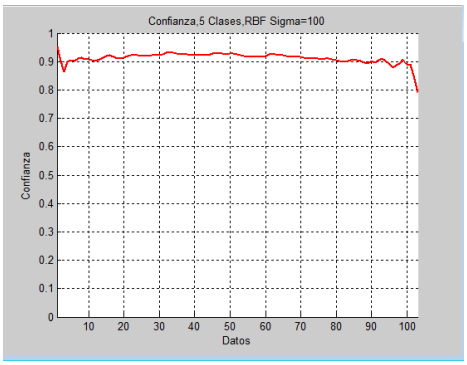


Figura 3.26 Confianza RBF σ=100, para 5 clases

Las figuras 3.27 y 3.28 muestran la credibilidad y confianza de las clases ECE7, HALFAC3, RX306, BOL5 y DENSIDAD 2, con los predictores conformales de CP-SVM-RBF, al realizarlo de manera “on-line”, se observa en la figura 3.27 que la credibilidad se inicia con un valor ligeramente superior al 0.5, existiendo un pico de bajada que nos muestra, que alguna de las señales obtiene una credibilidad menor que 0.5, pero al ser mayor que 0.4, se incluye al conjunto de entrenamiento.

En la figura 3.28 la confianza, empieza con un valor mayor que 0.9 observándose que sube al final, debido a que la credibilidad tiende a bajar en el último punto de la gráfica 3.27. En promedio, se obtiene un 65% para la credibilidad, mientras que la confianza alcanza un promedio del 69%,

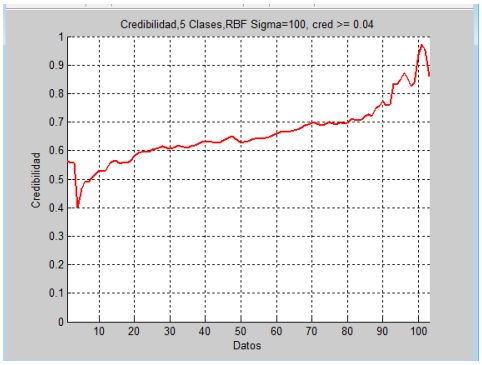


Figura 3.27 Credibilidad RBF σ =100, para 5 clases

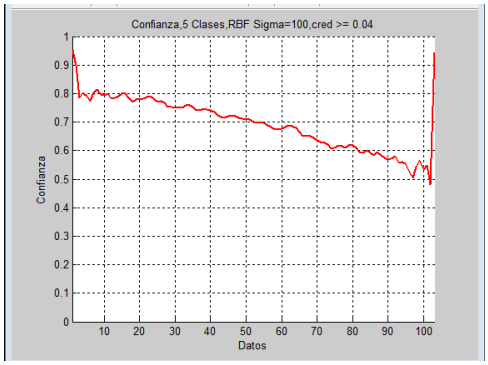


Figura 3.28 Confianza RBF σ =100, para 5 clases

Como se ha visto en este capítulo, la confianza nos informa la probabilidad de la clasificación prevista, en comparación con todas las clasificaciones realizadas, mientras que la credibilidad nos proporciona la idoneidad de los Predictores Conformales, esto debido a que los valores de baja credibilidad, indican que las señales, no pertenecen al conjunto de entrenamiento, por lo que al hacerlo de manera “on-line”, las gráficas nos mostraron el comportamiento individual de cada señal. Se puede comprobar que efectivamente la credibilidad es más alta en cada predicción, lo que indica

que efectivamente las señales del conjunto de validación no son extraños al conjunto de entrenamiento.

Por todo lo anterior, los predictores conformales resultan ser bastante buenos, pues no sólo nos ayudan a clasificar, sino que también, podemos ver la calidad de la clasificación, y saber con certeza si se está realizando la separación de las clases de manera correcta, y aunque se ha demostrado que casi cualquier algoritmo de clasificación, se puede transformar en un factor de predicción conformal, en este proyecto el CP-SVM, resultó ser el más adecuado para separar las señales de fusión.

CAPÍTULO 4

4 CONCLUSIONES Y TRABAJO FUTURO

En esta capítulo se sintetizan los resultados obtenidos durante la investigación realizada sobre el uso de predictores de conformación para determinar la calidad de clasificación de datos, se discuten los resultados desde una amplia perspectiva y finalmente se muestran las futuras direcciones de investigación.

4.1 Conclusiones

Los predictores conformales suponen un reciente método con fundamentos teóricos, por su buena capacidad de predicción, gracias a los cuales se disponen de dos alternativas para poder ver la calidad de la clasificación, la credibilidad y la confianza, sobre todo porque nos permite utilizar algoritmos subyacentes de clasificación. Por lo que para aplicar este método, se desarrolló una herramienta computacional para analizar datos de la predicción, credibilidad, la cual nos proporciona salidas de regiones de confianza. En el diseño de la herramienta se ha tenido en cuenta su generalización para ser utilizada para cualquier tipo de señales

La herramienta se ha aplicado a señales de fusión, de una base de datos del dispositivo TJ-II ubicado en el CIEMAT (Madrid, España).

Para poder utilizar las señales de fusión, se aplicó la transformada de wavelets, para disminuir la dimensión de las señales, además de filtrarlas para la eliminación de ruido, todo ello con el fin de facilitar la clasificación y su posterior reconstrucción, para que la tarea computacional fuera más eficiente.

En este trabajo, se ha aplicado el marco de predicción conformal en tres algoritmos para obtener las medidas de no conformidad con algoritmos de clasificación SVM y K-NN, además del método “Average”.

Se realizaron las comparaciones de los distintos algoritmos de clasificación, utilizados en términos de la credibilidad y confianza.

En el caso de las SVM, se realizaron pruebas con diferentes tipos de “Kernel” como: Lineal, Polinomial de grado=2, y RBF σ=100, que dieron resultados con mayor precisión en la predicción, en las tres pruebas realizadas, siendo la más eficiente el SVM-RBF. También se realizaron pruebas con K-NN con k=3 y con el método “Average”, que mostraron resultados inferiores a la predicción, comparada con las de SVM.

4.2 Trabajo Futuro

Para dar continuidad al trabajo aquí presentado, quedan por analizar diferentes vertientes que pueden mejorar las medidas de confianza utilizando métodos de aprendizaje.

Además, se podría experimentar con diferentes medidas de no conformidad para optimizar aún más la precisión y las medidas de la confianza de los métodos aquí aplicados.

En el caso de las SVM, se puede realizar una mejor selección de los parámetros en los que están basados los “kernels”, los cuales tienen un efecto significativo en el desempeño de los clasificadores, y así poder obtener mejores resultados.

BIBLIOGRAFÍA

[1] Ahad D., Kibler M.K., *Instance-based learning algorithms, Machine Learning* 6, 37-66, 1991.

[2] Burges, C. J., *A Tutorial on Support Vector Machines for Pattern Recognition Data*, Min. Knowl. Discov. Vol.2, No. 2, pp. 121-167, 1998.

[3] Carbonell, J., Michalski, R. y Mitchell T., *Machine Learning: The Artificial Intelligence* Aproach Vol. II. Morgan Kaufmann. 1986.

[4] Cost S., Salzberg S., *A weighted nearest neighbour algorithm for learning with symbolic features*, Machine Learning 10(1), 57-78, 1993.

[5] Courant R., Hilbert D., *Methods of mathematical physics*, Vol. 1, Inter science London, England, 1953.

[6] Cox David R. and V. Hinkley David., *Theoretical Statistics*, Chapman and Hall, London. 1974

[7] Craig S., Gammerman A., Vovk V., *computationally eficient transductive Machines*, In Toshio Okamoto, Roger Hartley, Kinshuk, and John Klus, editors, 11th International Conference on Algorithmic Learning Theory (ICALT-2000), pages 325-333, Madison, WI, USA, IEEE Computer Society Press, august 6-8 2000

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

[9] Dasarathy B.V., *Nearest Neighbour (NN), Norms: NN Pattern, Recognition Techniques*, IEEE Computer Society Press, 1991.

[10] Dormido- Canto S., Farias G., Dormido R., Vega J. Sánchez J., Santos M., *TJ-II wave forms analysis with wavelets and support vector machines The TJ-II Team*, Review Of Scientific Instruments Volume 75, Number 10, October 2004

[11] Dormido S., De la Cruz J.M., Vega J., Santos M., Dormido-Canto S., Sánchez J., Dormido-Canto,R., Farias, G., *Análisis de formas de onda de plasmas con wavelets y support vector machines*, 3ra.

Conferencia Iberoamericana en Sistemas, Cibernética e Informática CISCI. (2004).

[12] Duro N., J. Vega, R. Dormido G. Farias, S. Dormido-Canto , Sánchez J., Santos M., Pajares G., *Automated clustering procedure for TJ-II experimental signals*, Fusion Engineering and Design, ELSERVIER ,1987–1991

[13] Farias G., Dormido-Canto S., Vega J., Sánchez J., Duro N., Dormido R., Ochando M., Pajares G., Santos M., *Searching patterns inTJ-II temporal evolution signals with support vector machines*, *Fusion Engineering and Design*, 2006.,Vol. 81, pp 1987-1991, 2006.

[14] Fix E., Hodges Jr J.L., *Discriminatory analysis, nonparametric discrimination*, Project 21-49-004, Rept. 4, USAF School of Aviation Medicine, Randolf Field, 1951.

[15] Friedman N., Geiger D., Goldszmidt M., *Bayesian network classifiers*, Machine Learning, 29, 131-163., 1997.

[16] Gammerman A., Vovk V., Hedging Predictions in Machine Learning, Advance Access publication on February 1, 2007.

[17 Hart P.E., *The condensed nearest neighbour rule*, IEEE Transactions on Information Theory, IT-14, 515-516, 1968.

[18] Hastie T. and Tibshirani R., *Discriminant adaptive nearest neighbor classification*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 18, No 6, pp. 607-616, 1996.

[19] Khan L., Awad M. and Thuraisingham B., *A new intrusion detection system using support vector machines and hierarchical clustering*, The VLDB Journal, Vol. 16, No. 4, pp. 507-521, 2007.

[20] Kubat M., Holte R., & Matwin S., *Machine learning for the detection of oil spills in satellite radar images*, Machine Learning 30, 195–215, 1998.

[21] Kuhn H. W., Tucker A. W., *Nonlinear programming*. In Proceedings of the Second Berkeley Symposium, J. Neyman, Ed., University of California Press, Berkeley, pp. 481-492, 1951.

[22] Leslie G. V., *A theory of the learnable*, Communications of the ACM, 27:1134-1142.,1984.

[23] Mallat. S., *A theory for multiresolution signal descomposition: the*

wavelet Representation, IEEE Pattern Anal. And Machine Intell., vol. 11 No. 7, pp. 674-693, 1989.

[24] Misiti.M. Y Misiti G., Openheim y Poggy J.M., *Wavelet Toolbox, User Guide*, Version 2. The Math Works, Inc. 2000.

[25] Mitchell T., *Machine Learning*, McGraw-Hill, pp. 52-81,1997.

[26] Olson, David L., Delen, Dursun, *Advanced Data Mining Techniques*, Springer,2008.

[27] Pajares G., J. de la Cruz, *Visión por Computador, Imágenes y Digitales y Aplicaciones*, Editorial Rama, 2da. Edición, 2007.

[28] Papadópoulos H., Vovk V., Gammerman A., *Qualified predictions for large data sets in the case of pattern recognition*. ,In: Proceedings of the 2002 International Conference on Machine Learning and Applications (ICMLA'02), CSREA Press (2002) 159_163, 2002.

[29] Platt J., *Fast Training of support vector machine using sequential minimal optimization*. In A.S.B. Scholkopf, C. Burges, editor, Advances in Kernel Methods: support vector machine . MIT Press, Cambridge, MA 1998.

[30] Polikar R., *The Wavelet Tutorial Durham Computation Center*, Iowa State University USA, 1995. Documento disponibloe en: http://users.rowan.edu/~polikar/WAVELETS/WTtutorial.html

[31] Proedrou, K., Nouretdinov, I., Vovk, V., Gammerman, A., *Transductive confidence machines for pattern recognition*, Proceedings of the 13th European Conference on Machine Learning (ECML'02). Volume 2430 of Lecture Notes in Computer Science., 381-390,Springer 2002.

[32] Ramírez N, Santos M., Laguna M., *Aplicación y Método SVM (SUPPORT VECTOR MACHINE),* Segundo Congreso Nacional y Primer Congreso Internacional de Computación e Informática (CONACI 2010) Campeche, México.

[33] Ripley, B.D. "*Neural Networks and Related Methods for Classification*", Journal of the Royal Statistical Society, B, 56, pag 409-456. 1994

[34] Rychetsky M., *Algorithms and Architectures for Machine Learning based*

on Regularized Neural Networks and Support Vector Approaches, Shaker Verlag, 2001.

[35] Safavian R., Landgrebe D., *A survey of decision tree classifier methodology,* IEEE Transactions on Systems, Man, and Cybernetics, Vol. 21, No3, pp. 660-674, 1991.

[36] Shafer G., & Vovk, V., *A tutorial on conformal prediction*, Journal of Machine Learning Research, 9, 371–421., 2008.

[37] Shawe T., Cristianini N., *Kernel Methods for Pattern Analysis*, Cambridge University Press, 2004.

[38] Suykens J.A.K., De Brabanter J., Lukas L., Vandewalle J, *Weighted least squares support vector machines: robustness and sparse approximation*, Neurocomputing, Vol. 48, pp 85-105, 2002.

[39] Vapnik V., *Statistical Learning Theory*, Springer, N.Y., 1998.

[40] Vapnik V., *The Nature of Statistical Learning Theory*, Springer, N.Y., 1995.

[41] Vapnik V.N., *The Nature of Stastical Learning Theory*, 2nd edn, Springer, New York, 2000.

[42] Vovk V., Gammerman A., & Shafer G., *Algorithmic learning in a random World*, New York: Springer 2005.

[43] Wilson D.L., *Asymptotic properties of nearest neighbour rules using Edited data*, IEEE Transactions on Systems,Man and Cybernetics, Vol 2, 408-421, 1972.

[44] Yu H., Yang J. y Han Jiawei, *Classifying Large Data Sets Using SVMs With Hierarchical Clusters*, Proceedings of the 9 th ACM SIGKDD 2003, pp.3006-315, 2003.

[45] http://people-kyb.tuebingen.mpg.de/spider/tutorial.html

APÉNDICE A

Implementación de algoritmo CP-SVM, realizado en Matlab y Librerías de Spider

```
function[R,p,data1]=multiclase3(d,vec,numclase,p1,W1,tipe)
%d vector de entrenamiento, se incluye las columnas de las clases
%vec vector de validacion, (no se incluyen las clases)
% parametro de SVM, (de los kernel)
% Tipo de kernel
[nvec,npro]=size(vec);
vec1=vec(:,npro-1);
numc=numclase;
p=0;
W1;
tipe;
u=[];
[n,m]=size(d);
%determinamos el numero de clases
nclase= m+1-numclase;
if nargin==6
  u=[1:m-numclase];
end
dusado=d(:,u);
 %u
%[n,mm]=size(d);
data1=dusado;
clase=d(:,nclase:end);% define vector clase
%vec
% como se esta utilizando SVM one_vs_rest se compara solo con -1 como
clase
% principal que compara a las demas
Y=[-1];
%Longitud de Y
ny=length(Y);

%[nvec,npro]=size(vec);% nvec determina el tamaño de muestra a insertar
%para cada objeto
hh=0;
for i=1:nvec
%añado un el elemento para evaluar frente a los de entrenamientos
  data1(n+1,:)=vec(i,u);
% Se incrementa esta variable para ser usada en R para guardar las alphas
  hh=hh+1;
```

```
  %para cada clase
  for j=1:ny
    clase(n+1,:)=Y(j);
% Se realiza la llamada de la funcion por cada una de las columnas de las
% clases
for g=1:numc
D2=data(data1,clase(:,g));

 [r,a2]=train(one_vs_rest(svm({'C=1',kernel(W1,p1)})),D2)

q=a2.alpha;
% Se añade cada una de las alphas a R
R(hh,g)=q(n+1);
% Se saca la medida de no conformidad (no se incluyen en los datos de
% entrenamiento
  p(i,g)=mean(abs(q)>=abs(q(n+1)));
%  %_incremento de conjunto entrenamiento  mayor que 0.4
%   [ss,findxx,cind]=mayo(p(i,4:5));
 [s,findx,cind]=mayo(p(i,:));
  ind(i)=findx
  sm(i)=s
  ind'
mayor=sm'

%  credibilidad arriba de 0.04
  if mayor(i) >= 0.04
    data1(n+i,:)=vec(i,u);
    clase(n+i,:)=Y(j)
  end
      mayor2(i)=mayor(i);

end

  end
end
```

****

APÉNDICE B

Implementación de algoritmo CP-KNN, realizado en Matlab

```
function[R,p]=experimentoKNN(d,vec,W1)
W1
u=[];
[n,m]=size(d);
if nargin==3
  u=[1:m-1];
else
for i=1:length(v)
  if v(i)>=m
    disp('esta columa no se encuentra o es de la clase')
  else
 u=[u,v(i)] ;
  end

end

end
% conjunto de entrenamiento
dusado=d(:,u);

%u
[n,mm]=size(d);
data1=dusado;

clase=d(:,end);

%vec
switch W1
  case 1
    Y=[1]
  case 2
    Y=[1 2]
  case 3
    Y=[ 1 2 3];
  case 4
    Y=[1 2 3 4];
  case 5
   Y=[1 2 3 4 5];

end
```

```
disp(Y)
ny=length(Y);
%vec=objetos

 [nvec,npro]=size(vec);
%para cada objeto
hh=0;
for i=1:nvec

  %añado un el elemento para evaluar frente a los de entrenamientos
  data1(n+1,:)=vec(i,u);

  hh=hh+1;
  %para cada clase
  for j=1:ny
    clase(n+1,1)=Y(j);


 [q]=distancia(data1,clase);
 %end
 %q=alpha;

R(hh,j)=q(n+1);

 disp(q')
  p(i,j)=mean(abs(q)>=abs(q(n+1)));

  %disp(p')
  end

end

*****************************************************************************************

function[s]=distancia(x,y)

for i=1:length(x)
  c=[];
  t=0;e=0;
  for j=1:length(x)

  if j~=i & y(j)==y(i)
    t=t+1;
```

```
 c(t)=abs(x(j)-x(i));
end

if j~=i & y(j)~=y(i)
  e=e+1;
  f(e)=abs(x(j)-x(i));
end
 end
dn(i)=min(c);
dm(i)=min(f);
if dn(i)==0

  s(i)=0;
else
  s(i)=dn(i)/dm(i);
end

end
```

APÉNDICE C

Implementación de algoritmo CP-AVERAGE, realizado en Matlab

```
function[R,p]=experimentoPROM(d,vec,W1)
u=[];
[n,m]=size(d);
if nargin==3
  u=[1:m-1];
else
for i=1:length(v)
  if v(i)>=m
    disp('esta columa no se encuentra o es de la clase')
  else
 u=[u,v(i)] ;
  end
 end
 end

dusado=d(:,u);
%u
[n,mm]=size(d);
data1=dusado;

clase=d(:,end);

%vec
switch W1
  case 1
    Y=[1]
  case 2
    Y=[1 2]
  case 3
    Y=[ 1 2 3];
  case 4
    Y=[1 2 3 4];
  case 5
   Y=[1 2 3 4 5];

end
 ny=length(Y);
%vec=objetos
%disp(vec)
[nvec,npro]=size(vec);
```

```
%para cada objeto
hh=0;
for i=1:nvec

 %añado un el elemento para evaluar frente a los de entrenamientos
 data1(n+1,:)=vec(i,u);

 hh=hh+1;
 %para cada clase
 for j=1:ny
   clase(n+1,1)=Y(j);

   %calculo las medidas de no conformidad
for k=1:length(data1)
  if clase(k,1)==Y(j)
alpha(k)=AB(data1(find(clase==Y(j))),data1(k));
  else
 alpha(k)=AB(data1(find(clase==-Y(j))),data1(k));
  end
end
q=alpha;

R(hh,j)=q(n+1);

  %calculo los p-value
 p(i,j)=mean(abs(q)>=abs(alpha(n+1)));
 %Y(j)
   % F.Alpha
   %data
   %clase

 disp(alpha')
 end

end

*********************************************************
function[s]=AB(z,z1)

s=abs(sum(z)/length(z)-z1);
```