Clasificador robusto basado en máquinas de soporte vectorial para la localización de fallas en sistemas de distribución

Walter Julián Gil González

Universidad Tecnológica de Pereira
Facultad de Ingenierías: Eléctrica, Electrónica, Física y Ciencias de la Computación
Programa de Maestría en Ingeniería Eléctrica
Pereira, 2013

Clasificador robusto basado en máquinas de soporte vectorial para la localización de fallas en sistemas de distribución

Walter Julián Gil González

Proyecto de grado presentado como requisito para optar al título de Magíster en Ingeniería Eléctrica

Director: PhD. Juan José Mora Flórez

Universidad Tecnológica de Pereira
Facultad de Ingenierías: Eléctrica, Electrónica, Física y Ciencias de la Computación
Programa de Maestría en Ingeniería Eléctrica
Pereira, 2013

Clasificador robusto basado en máquinas de soporte vectorial para la localización de fallas en sistemas de distribución

Nota de aceptación

_____

_____
PhD. Juan José Mora Flórez
Director

_____
PhD. Mauricio Alexánder Álvarez López
Jurado

_____
MSc. Andrés Felipe Zapata Tapasco
Jurado

_____
MSc. Alberto Ocampo Valencia
Director Programa de Maestría en Ingeniería Eléctrica

Contenido

Listado de Figuras

Capítulo 2.

Capítulo 3.

Capítulo 4.

Anexo A

Listado de Tablas

Capítulo 4

Anexo B

Anexo D

Capítulo 1.

Introducción

1.1 Marco de referencia

La energía eléctrica constituye uno de los elementos fundamentales para el desarrollo económico y social de una región, dado que su disponibilidad determina en gran medida los niveles de productividad, las posibilidades de desarrollo agroindustrial y la calidad de vida de los pobladores. Por ello, el estudio de la continuidad del suministro de energía eléctrica ha tomado mucha fuerza y dentro de esta temática el problema de la localización de fallas se posiciona como uno de los más importantes [MORA,2005]. La continuidad en el servicio de energía eléctrica es un aspecto de la calidad, el cual es actualmente cuantificado mediante un índice de referencia agrupado de la discontinuidad (IRAD) y un índice trimestral agrupado de la discontinuidad (ITAD) en Colombia, por la comisión de regulación de energía y gas (CREG), cuyas metas permiten verificar el cumplimiento de los niveles mínimos de calidad del servicio [CREG,2008]. Para mantener estos índices en valores aceptables, es necesario conocer de manera confiable y rápida dónde ocurrió la interrupción del servicio, para la restauración de este en el menor tiempo posible. Sin embargo, en la gran mayoría de los casos la ubicación de la falla está asociada a la posible llamada telefónica de usuarios afectados o al tiempo que demora la cuadrilla en localizar la falla mediante una inspección visual de toda la red.

El problema de localización de fallas en sistemas de distribución se ha tratado de solucionar mediante un gran número de métodos, los cuales se basan en las medidas de tensión y corriente entre los estados de falla y prefalla en un extremo de la línea, para la estimación de la distancia de falla. Estos métodos se fundamentan en el modelo eléctrico de la red y proporcionan información sobre la distancia asociada a la impedancia desde la subestación hasta el lugar donde ha ocurrido la falla. Debido a la topología radial altamente ramificada, esta distancia puede coincidir en varios sitios del sistema, convirtiendo este en un problema de múltiple estimación, que dificulta la ubicación del lateral bajo la condición de falla. Adicionalmente, la exactitud de la estimación es altamente dependiente de un buen modelo del sistema [ZHU,1997]. Para la solución del problema se propone el uso de métodos basados en el conocimiento (MBC), que mediante la utilización de la información que proviene de los registros de tensión y corrientes medidos en la subestación permitan eliminar el problema de la múltiple estimación. Estos métodos también tienen como ventaja que no tienen una alta dependencia de un buen modelo del sistema de distribución, propio de los métodos basados en la estimación de la impedancia.

Los MBC se fundamentan en la extracción de conocimiento oculto en bases de datos. Generalmente, estas bases de datos requieren un procesamiento, el cual comprende diferente tareas tales como: manejo de ruido, manejo de datos faltantes, detección de datos anómalos, selección de atributos y normalización. Uno de los MBC más utilizados en los últimos años son las máquinas de soporte vectorial (SVM), debido a que su estudio es útil por características como la ausencia de mínimos locales, la alta capacidad de generalización

y el control y uso de funciones de núcleo o kernel [MORA,2006]. Las SVM son una técnica de clasificación basadas en los fundamentos teóricos de la teoría de aprendizaje estadístico desarrollada por Vapnik y Chervonenkis [KECM,2001]. El problema más común para hallar la zona en falla es el caso no lineal, que utiliza un hiperplano de clasificación lineal en un espacio característico.

Las SVM son una técnica robusta y han sido una herramienta aplicada con gran éxito en el problema de la localización de fallas en sistemas de distribución, como se presenta en [MORA,2005], [MORA,2008], [GAYA,2010], en estas investigaciones únicamente se utiliza un tipo de normalización, un tipo de kernel y un esquema de descomposición/reconstrucción. De otra parte, en los circuitos reales existen problemas cuando se aplica esta metodología debido a que se tiene incertidumbre fundamentalmente en el valor instantáneo y la naturaleza de la carga. En este proyecto se propone un análisis de sensibilidad para obtener un clasificador robusto basado en las SVM enfocado a la localización de fallas es sistemas de distribución, utilizando diferentes métodos de normalización, kernels, esquema de descomposición/reconstrucción y modelos de carga. El análisis de sensibilidad tiene como objetivo reducir la complejidad del problema y mostrar cuáles parámetros afectan en mayor proporción al clasificador, y así tener una herramienta de localización de fallas más robusta y precisa.

1.2 Delimitación de la investigación

La metodología propuesta en esta investigación tiene como objetivo principal realizar un análisis de sensibilidad sobre los parámetros de las máquinas de soporte vectorial, enfocado al problema de localización de fallas en sistemas de distribución.

El análisis de sensibilidad consiste en la generación de evaluaciones para determinar la influencia de la variación de los parámetros en las SVM. El principal problema consiste en generar un conjunto de incertidumbres que represente completamente el espacio y con un número de evaluaciones mínimas para reducir el costo computacional del problema. Una técnica adecuada que cumple con este propósito se denomina muestreo por *latin hypercube* (LH), ya que es una técnica que no involucra el modelo matemático del problema bajo estudio y permite evaluar de manera uniforme el espacio en consideración [FANG,2006] [VIAN,2009].

El análisis de sensibilidad se probará en dos circuitos para observar el comportamiento ante diferentes características y así determinar los parámetros que más afectan al localizador, y disminuir las dificultades al implementar esta metodología en circuitos reales.

Después de determinar los parámetros que más afectan en el desempeño de las SVM en el problema de localización de fallas, se realizará una prueba para determinar la precisión del localizador ante diferentes condiciones de operación del sistema en estudio.

1.3 Aportes de la tesis

Los aportes de esta tesis se presentan a continuación.

1.3.1 Artículos en revistas indexadas

- Artículo publicado "Análisis comparativo de metaheurísticas para calibración de localizadores de fallas en sistemas de distribución", Ingeniería y Competitividad. Universidad del Valle. 2013, Pp103 – 115.

- Artículo aprobado, en proceso de publicación "Análisis del procesamiento de los datos de entrada para un localizador de fallas en sistemas de distribución", Revista Tecnura. Universidad Distrital Francisco José de Caldas.

1.3.2 Herramientas desarrolladas

- Implementación del *latin hypercube* óptimo para construir unos escenarios teniendo en cuenta los parámetros propuestos en esta investigación para la máquina de soporte vectorial.

- Implementación de la máquina de soporte vectorial en el problema de localización de falla, con la posibilidad de variar todos los parámetros considerados.

- Desarrollo de una herramienta para realizar el análisis de regresión, la cual permite observar los parámetros que más afectan el desempeño del localizador.

- Desarrollo de un método de clasificación robusto basado en la máquina de soporte vectorial, que permita identificar la zona más probable de una falla en circuitos de distribución.

- Implementación de una herramienta computacional que reúne todos los algoritmos desarrollados y permite la conservación del conocimiento generado. El manual de la herramienta se encuentra en el Anexo A.

1.3.3 Trabajo en proyectos de investigación

- "Determinación de fallas paralelas de baja impedancia, como estrategia base para reducir la frecuencia y el tiempo de interrupción del suministro de energía eléctrica a los usuarios de las redes de distribución de EPM (LFEPM10)", desde Julio de 2011 hasta Diciembre de 2012.

- "Desarrollo de localizadores robustos de fallas paralelas de baja impedancia para sistemas de distribución de energía eléctrica -LOFADIS2012-". financiado por Colciencias, desde abril 2013. (actualmente se está trabajando en el).

1.4 Estructura del documento

El documento se divide en cinco capítulos. En el capítulo inicial se expone un marco de referencia del problema de localización de fallas en sistemas de distribución, se mencionan

los métodos utilizados para la solución del problema y se delimitan los alcances de la tesis. Finalmente, se mencionan los aportes del proyecto.

En el capítulo dos se presentan los aspectos teóricos asociados a la búsqueda tabú, técnica utilizada para la parametrización de la máquina de soporte vectorial y para el *latin hypercube* óptimo. Adicionalmente, se analizan los fundamentos de la máquina de soporte vectorial, los métodos de normalización más utilizados y el análisis de discriminación de datos. Finalmente, se presenta la estrategia diseñada para realizar el análisis de sensibilidad.

En el capítulo tres se presenta la metodología propuesta para el hacer el análisis de sensibilidad a la máquina de soporte vectorial aplicado a la localización de fallas en sistemas de distribución. Adicionalmente, se describe la metodología para crear el escenario del *latin hypercube* óptimo con búsqueda tabú.

En el capítulo cuatro se muestran las pruebas realizadas y los resultados obtenidos en la implementación de las técnicas en el problema de localización de fallas en los circuitos IEEE 34 y 44 nodos. Los circuitos se modelan con impedancia constante, potencia constante, corriente constante e híbrido. Además, se realizan variaciones a los parámetros del circuito tales como la carga nominal, la tensión en la subestación y la longitud de las líneas para determinar la robustez del método. Se presentan los parámetros que más influyen en el desempeño del clasificador al problema de localización de fallas en sistemas de distribución.

En el capítulo cinco se presentan las conclusiones más relevantes de la investigación realizada y las recomendaciones en la implementación para la solución de este problema.

Capítulo 2.

Aspectos teóricos

2.1 Introducción

En este capítulo se presentan los aspectos teóricos relacionados a la elaboración de esta investigación. Debido a los diversos temas tratados este capítulo se dividirá en secciones. Inicialmente se hará introducción de la técnica metaheurística búsqueda tabú. Luego se presenta una descripción de la SVM, con los kernels más utilizados y diferentes esquemas de descomposición/reconstrucción. Después, se muestran los métodos de normalización más utilizados en los MBC, además se presenta una ligera descripción del análisis de discriminación lineal de Fisher. Finalmente, se muestra una breve introducción al análisis de sensibilidad que se utilizó en este proyecto.

2.2 Búsqueda tabú

La búsqueda tabú es una técnica metaheurística de optimización que se utiliza para resolver problemas de alta complejidad. Este tipo de técnicas proporcionan soluciones factibles con bajos costos computacionales, aunque en algunos casos no alcanza el óptimo global de problema, siempre presenta soluciones de gran calidad. La idea básica de la búsqueda tabú es la utilización explícita de un historial de búsqueda (una memoria de corto plazo), tanto para escapar de los óptimos locales como para implementar una estrategia de exploración y evitar la búsqueda repetida en la misma región [GLOV,2002]. Esta memoria de corto plazo se implementa como una lista tabú, donde se mantienen las soluciones visitadas más recientemente para excluirlas de los próximos movimientos. En cada iteración se elige la mejor solución entre las permitidas y ésta se añade a la lista tabú.

El algoritmo de búsqueda tabú básico involucra los siguientes elementos:

Configuración inicial. Una configuración es el conjunto de variables enteras del problema dispuesto en un arreglo (vector o matriz). La configuración inicial se puede generar de manera aleatoria o se puede obtener utilizando un algoritmo constructivo que utilice factores de sensibilidad o cierta lógica heurística [GARC,2005].

Generación del vecindario. Un vecino de una configuración X es una configuración X_0 obtenida a partir de X por medio de una transición simple. En la mayoría de los casos el vecindario N(X) puede ser muy grande lo que implica un elevado esfuerzo de cómputo en el proceso de búsqueda. Debido a esto, se requiere reducir el número de vecinos a N1(X), redefiniendo las reglas de construcción del vecindario. Esta etapa es fundamental en el proceso, ya que de ella depende en gran parte, el éxito de la búsqueda [TORO,2008].

Selección del mejor vecino. Una vez que el vecindario es definido, cada vecino es evaluado para determinar el valor de su función objetivo y si cumple o no con las restricciones planteadas, de esta manera se determina la factibilidad de la configuración

vecina. El primer candidato de la lista (de mejor función objetivo) es seleccionado si él no es tabú (prohibido) y si es factible, de lo contrario se busca entre los siguientes vecinos. Este modo de selección es denominado búsqueda agresiva [GARC,2005].

Actualización de la estructura tabú. La estructura tabú almacena la información de los atributos que han cambiado o que han permanecido sin cambio alguno. Esta información se almacena en las memorias de corto y largo plazo.

Memoria de corto plazo. La memoria de corto plazo usa básicamente la información de atributos de configuraciones que han cambiado recientemente. Esta información es conocida como memoria basada en hechos recientes. La idea básica de este tipo de memoria es evitar volver a configuraciones ya visitadas, penalizando la búsqueda para evitar el ciclaje [TORO,2008].

2.3 Máquinas de soporte vectorial (SVM)

Las SVM son una consecuencia práctica de la teoría de aprendizaje y su estudio es útil por características como la ausencia de mínimos locales, la alta capacidad de generalización y el control y uso de funciones de núcleo o kernel [MORA,2005]. Las SVM son una herramienta usada en la clasificación de objetos puntuales de dos clases o más, que ha resultado ser muy poderosa, se basa en encontrar una superficie de clasificación determinada por ciertos puntos de un conjunto de entrenamiento, este conjunto de vectores debe estar en la frontera entre los dos subconjuntos en que se clasificarán los puntos, estos vectores son llamados vectores de soporte. La arquitectura de las SVM sólo depende de un parámetro de penalización denotado como C y la función kernel (incluyendo sus parámetros).

2.3.1 Fundamentación teórica del método

Considérense *n* datos de entrenamiento $\boldsymbol{N}$ dimensionales $(\vec{x}_i)$, con su respectiva etiqueta (y_i), tal como se muestra en la ecuación (2.1).

$$x_i \in \Re^N \text{ y } y_i \in \{+1, -1\} \tag{2.1}$$

Se busca estimar una función *f* tal que para una entrada en $\Re^N$ produzca una salida en {±1}, según se presenta en (2.2)

$$f: R^N \rightarrow \{+1, -1\} \tag{2.2}$$

Así se puede clasificar correctamente un nuevo dato, considerando que $y = f(\vec{x})$ para este nuevo dato es generado con la misma distribución de probabilidad $P(\vec{x}_i, y_i)$, de los datos de entrenamiento.

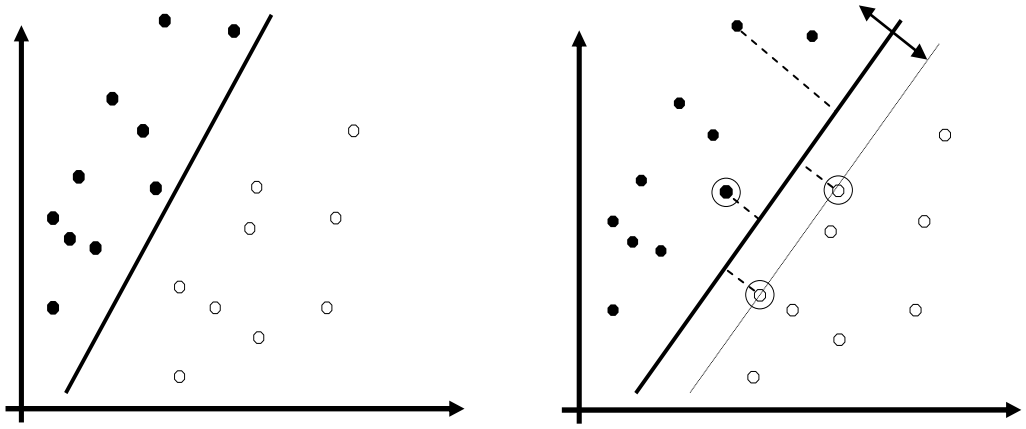
Como no se imponen restricciones en la función que se escoge, se pueden cometer errores en la estimación, ya que aunque se realice un buen entrenamiento, no necesariamente tiene una buena generalización para datos desconocidos. Por tanto, el aprendizaje perfecto no es

posible y la minimización del error de entrenamiento no implica que haya no error en la prueba.

2.3.2 Análisis del caso linealmente separable

a. Hiperplano clasificador óptimo

Los clasificadores con SVM se fundamentan en la obtención de hiperplanos que separen los datos de entrenamiento en dos subgrupos. Entre cada una de las clases etiquetadas como {−1, +1}, existe un único hiperplano óptimo de separación (OSH). Se busca que la distancia entre el hiperplano óptimo y el patrón de entrenamiento más cercano sea máxima, con la intención de forzar la generalización de la máquina de aprendizaje [BURG,1998].



a) **b)**

Figura 2.1. Hiperplanos que separan correctamente un conjunto de datos. a) Hiperplano de separación de datos. b) OSH con un mayor margen de separación entre clases.

El hiperplano óptimo de separación (OSH) de la figura 2.1, se presenta en (2.3).

$$g(\vec{x}) = (\vec{w} \cdot \vec{x}) + b = 0 \tag{2.3}$$

Para maximizar el margen, se proponen dos planos paralelos que contienen los puntos más cercanos al OSH. Si *1/||w||* es la distancia entre el punto más cercano al OSH, las ecuaciones de los planos están dadas por (2.4) y (2.5):

$$p_{+1} = (\boldsymbol{w}^T . \boldsymbol{x}_{+1}) + b = +1 \tag{2.4}$$
$$p_{-1} = (\boldsymbol{w}^T . \boldsymbol{x}_{-1}) + b = -1 \tag{2.5}$$

El margen definido como la distancia perpendicular entre (2.4) y (2.5) está dado por (2.6).

$$\begin{gathered} [(\vec{w} \cdot \vec{x}_{\mp 1}) + b] - [(\vec{w} \cdot \vec{x}_{-1}) + b] = +1 - (-1) \\ \vec{w} \cdot (\vec{x}_{+1} - \vec{x}_{-1}) = 2 \\ \frac{\vec{w}}{\|\vec{w}\|} \cdot (\vec{x}_{+1} - \vec{x}_{-1}) = \frac{2}{\|\vec{w}\|} \end{gathered} \tag{2.6}$$

La distancia entre el OSH y el origen de coordenadas está dada por (2.7)

$$\text{Si } (\vec{w} \cdot \vec{x}_i) + b = 0 \text{ y } \vec{x}_i = 0 \Longrightarrow \frac{(\vec{w} \cdot \vec{x}_i) + b}{\|\vec{w}\|} = \frac{b}{\|\vec{w}\|} \tag{2.7}$$

Por definición, entre los dos hiperplanos no deben existir datos de entrenamiento y por tanto los datos deben cumplir con (2.8) y (2.9).

$$(\vec{w} \cdot \vec{x}) + b \geq +1 \; para \; y_i = +1 \tag{2.8}$$
$$(\vec{w} \cdot \vec{x}) + b \leq -1 \; para \; y_i = -1 \tag{2.9}$$

Luego la función decisión $f_{w,b}(\vec{x}_i) = y_i$ corresponde al signo que resulta de evaluar un dato en la ecuación del OSH (2.3), tal como se presenta en (2.10).

$$f_{w,b}(\vec{x}_i) = sign[g(x_i)] = sign[(\vec{w} \cdot \vec{x}) + b] \tag{2.10}$$

Combinando (2.8) y (2.9) se obtiene (2.11).

$$y_i(\vec{w} \cdot \vec{x} + b) \geq 1 \tag{2.11}$$

Si existe un hiperplano que satisfaga (2.11), se dice que los datos son linealmente separables. Para encontrar el OSH se debe maximizar el margen (2.6), teniendo en cuenta la restricción (2.11), lo que es equivalente a resolver el problema planteado en (2.12) y (2.13).

$$mín_{w,b} \frac{1}{2}(\vec{w} \cdot \vec{w}) \tag{2.12}$$
$$sujeto \; a \; y_i(\vec{w} \cdot \vec{x} + b) \geq 1 \;\; \forall_i \tag{2.13}$$

La función (2.12) se llama función objetivo, y junto con (2.13) se conforma un problema de optimización cuadrático con restricciones. Los problemas de este tipo se tratan introduciendo el método de multiplicadores de *Lagrange*.

b. Solución al problema de optimización con restricciones

Para las restricciones de la forma $R_i \geq 0$, cada restricción se multiplica por α_i(un multiplicador de Lagrange positivo), y se restan de la función objetivo, para así formar la *función de Lagrange* presentada en (2.14) [BURG,1998].

$$L(\vec{w}, \vec{x}, b) = \frac{1}{2}\|\vec{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left[y_i(\vec{w} \cdot \vec{x} + b)\right] - 1 \tag{2.14}$$

La función de Lagrange (2.14) se debe minimizar con respecto a las variables primarias $\boldsymbol{w}$ y b, y maximizada sobre los α_i para encontrar el punto de silla [SCHO,2002]. Para el caso de $y_i(\vec{w} \cdot \vec{x} + b) - 1 > 0$, el correspondiente α_i debe ser cero, debido a que éste es el valor que maximiza a (2.14).

Los $\alpha_i$diferentes de cero son para el caso en que $y_i(\vec{w} \cdot \vec{x} + b) - 1 = 0$, que corresponden a los patrones de entrenamiento que quedan sobre los hiperplanos paralelos al OSH dados por la ecuaciones (2.4) y (2.5). Este último enunciado corresponde a las condiciones de *Karush-Kuhn-Tucker* o condiciones complementarias de optimalidad [KUHN,1951], presentadas de (2.15) a (2.17).

$$\alpha_i[y_i(\vec{w} \cdot \vec{x} + b) - 1] = 0, \forall_i \tag{2.15}$$

$$\frac{\partial}{\partial w} L(\vec{w}, \vec{x}, b) = 0 \implies \vec{w} = \sum_{i=1}^{n} \alpha_i y_i \vec{x}_i \tag{2.16}$$

$$\frac{\partial}{\partial b} L(\vec{w}, \vec{x}, b) = 0 \implies \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{2.17}$$

La solución de $\vec{w}$ en (2.16) queda en función de un subconjunto de patrones de entrenamiento cuyo multiplicador de Lagrange es diferente de cero. Es decir, el soporte de $\vec{w}$ está en los patrones de entrenamiento más cercanos al OSH. De aquí el nombre de *Máquinas de Soporte Vectorial*.

Reemplazando (2.16) y (2.17) en (2.14), se eliminan las variables primarias $\vec{w}$ y b llegando así al *problema de optimización dual de Wolfe*, el cual se resuelve mediante (2.18).

$$mín_\alpha \sum_{i=1}^{n} \alpha_i - \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j (\vec{x}_i \cdot \vec{x}_j) \tag{2.18}$$

$$\text{sujeto a: } \alpha \geq 0, \forall_i \; y \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{2.19}$$

Ahora, la ecuación del OSH y la función de decisión se pueden expresar como se presentan en (2.20) y (2.21).

$$\mathrm{g}(\vec{x}) = \sum_{i=1}^{n} [\alpha_i y_i (\vec{x}_i \cdot \vec{x})] + b \tag{2.20}$$

$$f(\vec{x}) = sign\left(\sum_{i=1}^{n} [\alpha_i y_i (\vec{x}_i \cdot \vec{x})] + b \right) \tag{2.21}$$

La utilización de los multiplicadores de Lagrange se debe a dos razones fundamentales:

- Las restricciones presentadas en (2.13), quedan en función de α_i (2.19), que las hace más fácil de resolver.
- En la reformulación del problema, los datos de entrenamiento $\vec{x}_i$ sólo aparecen en forma de productos punto entre ellos mismos (2.18), (2.20) y (2.21).

$$b = y_i - \vec{w} \cdot \vec{x}_i \tag{2.22}$$

2.3.3 Análisis del caso linealmente no separable

a. Planteamiento general

La implementación de clasificadores basados en SVM desarrolladas como se presenta en las secciones anteriores, puede tener altos errores, debido a que en la práctica no necesariamente existe un hiperplano separador, y si existe, no siempre es la mejor solución para el problema de clasificación. Cuando existen datos erróneos, ruido o alto traslapamiento de clases en los datos de entrenamiento, ésta solución puede no ser la mejor.

Debido a lo anterior, se propone otra alternativa que busque el mejor hiperplano tolerando ruido en los datos de entrenamiento. Una solución aparente consiste en encontrar el hiperplano que conduzca al menor número de errores de entrenamiento, pero desafortunadamente, esto se convierte en un problema combinatorial difícil de aproximar. *Cortes y Vapnik* [CORT,1995], proponen un planteamiento diferente para las SVM, basándose en [BENN,1992], para permitir la posibilidad de ejemplos que violen la restricción (2.13), por la consideración de variables de relajación ξ_i (slack).

$$\xi_i \geq 0 \tag{2.23}$$

La nueva restricción se presenta en (2.24). En la figura 2.2 se presenta caso linealmente no separable.

$$y_i(\vec{w} \cdot \vec{x} + b) \geq 1 - \xi_i \ \forall_i \tag{2.25}$$

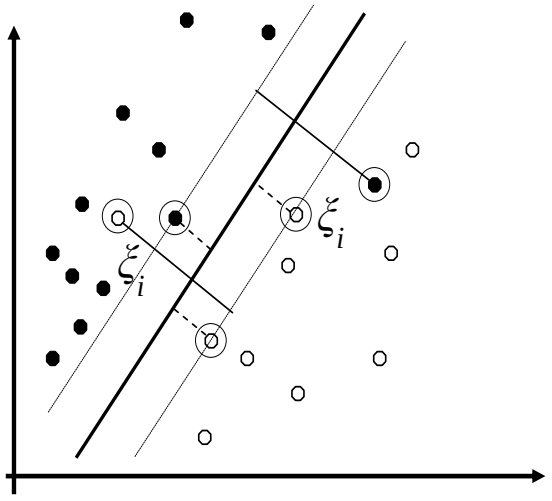


Figura 2.2. Hiperplano lineal clasificador para el caso no separable

Luego se encuentra el clasificador que mejor generaliza, controlando su capacidad de clasificación (con $\|w\|$), y el límite superior del número de errores de entrenamiento

$\left(\sum_{i=1}^{n} \xi_i\right)$. Una posible forma de obtener el hiperplano óptimo con margen débil es minimizando la función (2.25) sujeta a (2.26).

$$mín_{w,b} \frac{1}{2}(\vec{w} \cdot \vec{w}) + C \sum_{i=1}^{n} \xi_i \tag{2.25}$$

$$sujeto\ a\ y_i(\vec{w} \cdot \vec{x}_i + b) \geq 1 - \xi_i,\ \forall_i \tag{2.26}$$

Un alto valor del parámetro *C* corresponde a una alta penalización a los errores. Con los multiplicadores de Lagrange el problema se transforma en (2.27), sujeto a (2.28).

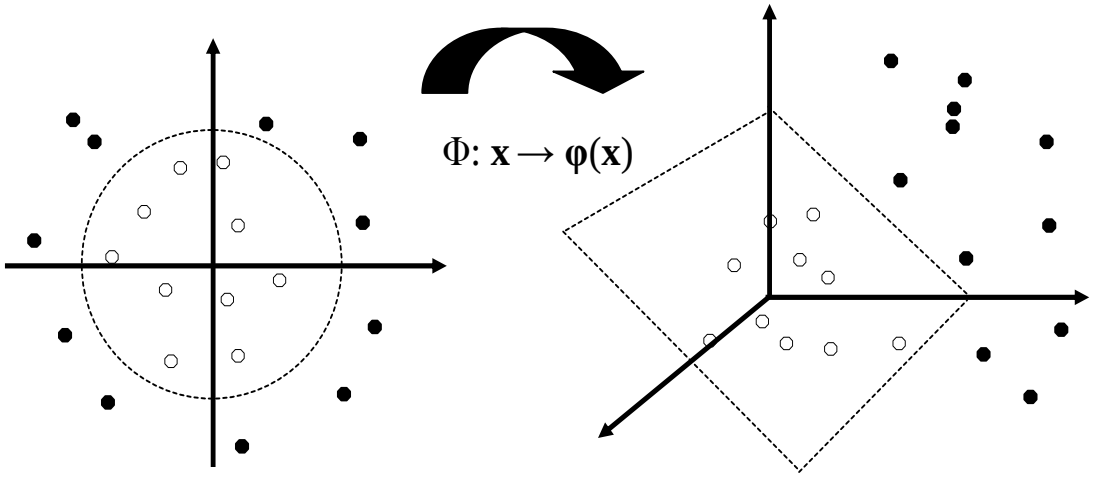
$$mín_{\alpha} \sum_{i=1}^{n} \alpha_i - \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \left(\vec{x}_i \cdot \vec{x}_j\right) \tag{2.27}$$

$$\text{sujeto a: } 0 \leq \alpha \leq C,\ \forall_i\ y \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{2.28}$$

La solución es la misma que para el caso lineal, ecuación (2.16). El hiperplano separador solución se puede expresar como (2.20), y la función de decisión como (2.21).

b. Máquinas de soporte no lineales

El principio de las SVM no lineales consiste en "mapear" o establecer una relación entre el espacio de entrada y un espacio de representación de dimensión alta, a través de una función no lineal elegida a priori [BOSE,1992], tal como se presenta en la figura 2.3.



a) b)

Figura 2.3 a) espacio de entrada, b) espacio de representación de alta dimensión

Por medio de una función (Φ) se trazan los datos de entrada ($\vec{x}_i \in R^N$) a algún espacio de mayor dimensión y con producto punto definido (2.29). Este espacio se llama *espacio característico* (***F***).

$$\Phi: \Re^N \rightarrow \boldsymbol{F} \tag{2.29}$$

Así, la función (2.20) que depende del producto punto de los vectores en el espacio de entrada, pasa a una función que depende del producto punto de los vectores en el espacio característico, como se muestra en (2.30).

$$g(\vec{x}) = \sum_{i=1}^{n} \left[\alpha_i y_i \left(\Phi(\vec{x}_i) \cdot \Phi(\vec{x})\right)\right] + b \tag{2.30}$$

Entonces se define una función que sea el producto punto de los vectores en el espacio característico, presentada en (2.31).

$$k(\vec{x}) = \left(\Phi(\vec{x}_i) \cdot \Phi(\vec{x})\right) \tag{2.31}$$

Debido a que (***F***) es de alta dimensión, el lado derecho de la ecuación (2.30) es costoso en términos computacionales. Sin embargo existe una función *kernel Mercer* (*k*) que puede evaluarse eficazmente y se puede demostrar que corresponde a un trazado de (Φ) en un espacio que abarca todos los productos punto. Un ejemplo para el *kernel* polinomial se muestra en (2.32), (2.33) y (2.34).

$$k(\vec{x}, \vec{y}) = (\vec{x} \cdot \vec{y})^d \; con \; (\vec{x} \cdot \vec{y}) \; \in \; \Re^2 \; y \; d = 2 \tag{2.32}$$

Se tiene que:

$$\begin{gathered} k(\vec{x}, \vec{y}) = \left[\binom{x_1}{x_2} \cdot \binom{y_1}{y_2}\right]^2 \\ k(\vec{x}, \vec{y}) = [x_1 y_1 + x_2 y_2]^2 \\ k(\vec{x}, \vec{y}) = {x_1}^2 {y_1}^2 + 2 x_1 y_1 x_2 y_2 + {x_2}^2 {y_2}^2 \end{gathered} \tag{2.33}$$

La ecuación (2.32) se puede escribir como (2.33).

$$k(\vec{x}, \vec{y}) = \left[\begin{pmatrix} {x_1}^2 \\ \sqrt{2} x_1 x_2 \\ {x_2}^2 \end{pmatrix} \cdot \begin{pmatrix} {y_1}^2 \\ \sqrt{2} y_1 y_2 \\ {y_2}^2 \end{pmatrix}\right] = \Phi(\vec{x}) \cdot \Phi(\vec{y}) \tag{2.34}$$

Con la función *kernel* no se necesita definir explícitamente la función Φ, ya que ésta entrega directamente al resultado del producto punto, que es lo que realmente interesa para la aplicación de clasificación. Generalizando, se puede probar que por cada función *kernel* que presente una matriz definida positiva, se puede construir una función Φ que cumpla con (2.31). Algunos de los *kernels* más utilizados son el polinomial (2.35), función de base radial (RBF) (2.36), sigmoide (2.37) y radial Laplace (2.38) [SALA,2012].

$$k(\vec{x},\vec{y}) = (\vec{x}\cdot\vec{y}+1)^{\gamma}\, para\ c>0 \tag{2.35}$$

$$k(\vec{x},\vec{y}) = e^{-\gamma|\vec{x}-\vec{y}|^2} \tag{2.36}$$

$$k(\vec{x},\vec{y}) = \tanh\left(\left(\kappa(\vec{x},\vec{y})\right)+\gamma\right) \tag{2.37}$$

$$k(\vec{x},\vec{y}) = e^{-\gamma|\vec{x}-\vec{y}|} \tag{2.38}$$

Incluyendo la función *kernel* se puede reescribir (2.27) sujeta a (2.28), como (2.39).

$$\begin{gathered} mín_{\alpha} \sum_{i=1}^{n}\alpha_i - \sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j k\left(\vec{x}_i,\vec{x}_j\right) \\ \text{sujeto a: } 0\le\alpha\le C,\ \forall_i\ y\ \sum_{i=1}^{n}\alpha_i y_i = 0 \end{gathered} \tag{2.39}$$

Las ecuaciones del OSH (2.20) y función decisión (2.21) se reescriben como (2.40) y (2.41)

$$\mathrm{g}(\vec{x}) = \sum_{i=1}^{n}[\alpha_i y_i k(\vec{x}_i,\vec{x})] + b \tag{2.40}$$

$$f(\vec{x}) = sign\left(\sum_{i=1}^{n}[\alpha_i y_i k(\vec{x}_i,\vec{x})] + b\right) \tag{2.41}$$

2.3.4 Máquinas biclasificadoras generalizadas

Este tipo de máquina construye una función clasificadora global a partir de un conjunto de funciones biclasificadoras. Existen técnicas de descomposición y reconstrucción que permiten a las SVM biclasificadoras manejar problemas de multiclasificación con mayor simplicidad y/o menor tiempo de respuesta que una SVM generalizada a multiclasificación [MORA,2006].

a. Esquemas de descomposición

En los esquemas de descomposición estándar se construyen *m* máquinas biclasificadoras, en paralelo que son entrenados sobre modificaciones del conjunto de aprendizaje. Para esto se crea una matriz de descomposición, donde los elementos de unas clases son asignados a salidas positivas, los de otras a salidas negativas. Algunas arquitecturas de descomposición se presentan en [MORA,2005]. A continuación se presentan las arquitecturas de descomposición utilizadas en esta tesis.

- **Uno contra uno**

Conocido como 1-*v*-1 (*one-versus-one*), este esquema consiste en entrenar $l\,(l-1)/2$ clasificadores binarios, donde l es el número total de clases. El entrenamiento de cada

clasificador binario se realiza con sólo dos de las clases en el grupo de datos de entrenamiento [MORA,2006], siguiendo el esquema presentado en (2.42).

$$D_{i,j} = \begin{cases} +1 & si\ n\ \in n_j \\ -1 & si\ n\ \in n_p \\ 0 & si\ n\ \in n_r \end{cases} \tag{2.42}$$

Donde, n_j datos correspondientes a la clase $j \in \{1,2,\dots,k\}$ se les asigna la etiqueta $t_j = +1$ y a los n_i datos pertenecientes a la clase $i \in \{1,2,\dots,k\}\ i \neq j$ se les etiqueta con $t_i = -1$; el resto de datos, $n_r = n - n_j - n_i$ no participan en el entrenamiento, por lo que se etiquetan con $t_r = 0$. La matriz de descomposición para un problema con cuatro clases se presenta en (2.43).

$$D_{1\ vs\ 1} = \begin{pmatrix} +1 & -1 & 0 & 0 \\ +1 & 0 & -1 & 0 \\ +1 & 0 & 0 & -1 \\ 0 & +1 & -1 & 0 \\ 0 & +1 & 0 & -1 \\ 0 & 0 & +1 & -1 \end{pmatrix} \tag{2.43}$$

- **Uno contra el resto**

Conocido como 1-v-r (*one-versus-rest*), este esquema se basa en la idea de que si existe un grupo de *n* datos de entrenamiento donde existen *l* clases ($l > 2$), se pueden tener un grupo de *m* clasificadores binarios (donde $m = l$), cada uno entrenado para separar una clase del resto de clases existentes ($l - 1$) [MORA,2006]. La descomposición se realiza siguiendo el esquema presentado en (2.44).

$$D_{i,j} = \begin{cases} +1 & si\ n\ \in n_j \\ -1 & si\ n\ \in n_r \end{cases} \tag{2.44}$$

Donde, n_j datos pertencientes a la clase $j \in \{1,2,\dots,l\}$ se etiquetan con $t_j = +1$ y el resto de datos $n_r = n - n_j$, se etiquetan con $t_r = -1$. La matriz de descomposición para un problema con cuatro clases se presenta en (2.45).

$$D_{1\ vs\ r} = \begin{pmatrix} +1 & -1 & -1 & -1 \\ -1 & +1 & -1 & -1 \\ -1 & -1 & +1 & -1 \\ -1 & -1 & -1 & +1 \end{pmatrix} \tag{2.45}$$

- **ECOC**

Conocido como ECOC (*Error Correcting Output Codes*) [MORA,2006], utiliza la codificación estándar para obtener robustez contra fallos en las máquinas biclasificadoras. Se denomina codificación estándar a cada una de las posibles particiones de todo el conjunto de clases $y_i \in \{1, \dots, l\}$, en problemas de biclasificación, que asignan etiquetas positivas $t_p = +1$ a los patrones de entrenamiento n_j de un cierto subconjunto de clases Y_j, y

etiquetas negativas $t_p = -1$ a los patrones de entrenamiento n_r representantes del resto de clases Y_r. La descomposición se realiza siguiendo el esquema presentado en (2.46), debe ser tan diferente como sea posible en términos de la distancia Hamming para añadir redundancia, en este caso $m = 2^{l-1}-1$ (donde m es el número de biclasificadores y l es el número de clases) La matriz de descomposición para un problema con cuatro clases se presenta en (2.47).

$$D_{i,j} = \begin{cases} +1 & si\ n\ \in n_j \\ -1 & si\ n\ \in n_r \end{cases} \tag{2.46}$$

$$D_{ECOC} = \begin{pmatrix} +1 & -1 & -1 & -1 \\ +1 & -1 & -1 & +1 \\ +1 & -1 & +1 & -1 \\ +1 & -1 & +1 & +1 \\ +1 & +1 & -1 & -1 \\ +1 & +1 & -1 & +1 \\ +1 & +1 & +1 & -1 \end{pmatrix} \tag{2.47}$$

b. Esquemas de reconstrucción

Los métodos de reconstrucción toman cada máquina biclasificadora entrenada que emite una respuesta en forma numérica $z^i = g_i(\vec{x})$ a una entrada $\vec{x}$. La información más importante en esta respuesta, en principio, se encuentra en el signo $s^i = f_i(\vec{x}) = sing\big(g_i(\vec{x})\big)$ que adopta la función de decisión. A continuación se muestra el esquema de reconstrucción para cada esquema de descomposición:

- *i*-ésimo 1-v-1 máquina biclasificador

$$\Theta\big(s^i\big) = \begin{cases} y_i & si\ s^i = +1 \\ y_p & si\ \ s^i = -1 \end{cases} \tag{2.48}$$

- *i*-ésimo 1-v-r máquina biclasificadora

$$\Theta\big(s^i\big) = \begin{cases} y_i & si\ s^i = +1 \\ 0 & si\ \ s^i = -1 \end{cases} \tag{2.49}$$

- *i-ésimo* ECOC máquina biclasificadora

$$\Theta\big(s^i\big) = \begin{cases} Y_i & si\ s^i = +1 \\ Y_r & si\ \ s^i = -1 \end{cases} \tag{2.50}$$

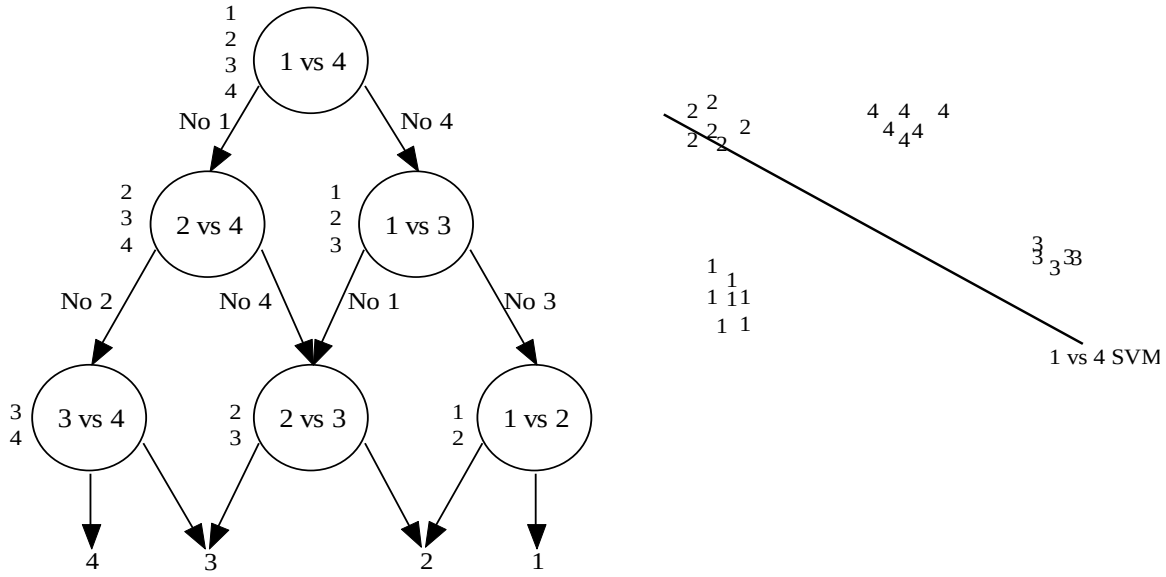
El esquema de reconstrucción más utilizado son los esquemas de votación. Algunos de los más utilizados se presentan a continuación:

- Votación por unanimidad: se determina como respuesta aquella única clase que haya obtenido todos los votos posibles en las predicciones.

- Votación por mayoría absoluta: se determina como respuesta aquella única clase que haya obtenido más de la mitad de los votos posibles.

- Votación por mayoría simple: se determina como respuesta final aquella única que haya obtenido más votos que el resto de clases.

También para la descomposición 1-v-1 se puede utilizar un árbol de decisión (Decision Directed Acyclic Graph - DDAG) en la reconstrucción. El DDAG funciona sólo evaluando dos clases. Este esquema de recontrucción elige una de las dos clases y la otra clase es eliminada. Se hace esto sucesivamente hasta que sólo quede una clase, para un problema *l*-clases, el número decisiones que se debe hacer para obtener una respuesta es *l*–1 [KIJS,2002].



a) **b)**

Figura 2.4. El DDAG. a) El DDAG para encontrar la mejor clase de cuatro clases. b) Un diagrama del espacio de entrada para un problema de cuatro clases. Un SVM 1-*v*-1 sólo puede excluir a una clase en consideración.

2.4 Métodos de normalización

La normalización de los datos es necesaria para adecuarlos a los problemas de clasificación, debido que los datos no están definidos en las mismas escalas numéricas y en algunos casos siguen diferentes distribuciones. Las normalizaciones más utilizadas para los problemas de clasificación son: *Min-max, Z-score, Median and median absolute deviation* y *Sigmoid funtion* [FARR,2005] [ANIL,2005] [SNEL,2005].

2.4.1 Min-Max (MM):

En este método los valores mínimo y máximo de la base de datos se desplazan a los valores 0 y 1, respectivamente, y todos los demás datos se transforman en el rango {0,1}, utilizando la ecuación (2.51) [FARR,2005].

$$s'_{\mathrm{ij}} = \frac{s_{\mathrm{ij}} - min(s_{\mathrm{j}})}{max(s_{\mathrm{j}}) - min(s_{\mathrm{j}})} \tag{2.51}$$

Donde, s'_{ij} es el dato *i* transformado del conjunto de datos *j*; s_{ij} es el dato *i* original del conjunto de datos *j*; min_{j} es el valor mínimo del conjunto de datos *j*; max_{j} es el valor máximo del conjunto de datos *j*.

2.4.2 Z-score (ZS):

Este método transforma los datos a una distribución con media 0 y desviación estándar 1. En la ecuación (2.52), los operadores $mean()$ y $std()$ denotan la media aritmética y la desviación estándar, respectivamente [SNEL,2005].

$$s'_{ij} = \frac{s_{ij} - mean(S_j)}{std(S_j)} \tag{2.52}$$

Donde, $S_j$es el conjunto de datos *j*; $mean(S_j)$ es la media aritmética del conjunto de datos *j*; $std(S_j)$ es la desviación estándar del conjunto de datos *j*.

2.4.3 Median and median absolute deviation (MMAD):

Este método es insensible a valores atípicos y puntos en colas de la distribución. Por lo tanto, un esquema de normalización combinando la media y la desviación media absoluta de los datos sería un método más robusto. Los datos se normalizan según la ecuación (2.53) [ANIL,2005].

$$s'_{ij} = \frac{s_{ij} - mean(S_j)}{\mathrm{MAD}} \tag{2.53}$$

Donde, MAD se calcula como se muestra en la ecuación (2.54).

$$\mathrm{MAD} = mean(\left|S_j - mean(S_j)\right|) \tag{2.54}$$

2.4.4 Sigmoial funtion (SF)

Esta función también se conoce como una *squashing function*, porque transforma los datos en un rango de entrada entre 0 a 1. Esta función es no lineal y diferenciable, permitiendo que las técnicas de clasificación tengan un mejor manejo de los datos en problemas que no son linealmente separables. Los datos se normalizan según la ecuación (2.55) [HAN,2006].

$$s'_{ij} = \frac{1}{1 + e^{-\mathrm{T}_{ij}}} \tag{2.55}$$

Donde, T_{ij} se calcula como se muestra en la ecuación (2.52).

2.5 Análisis de discriminación de datos

El análisis de discriminación de datos es un método que trata de encontrar una función o modelo capaz de diferenciar datos de entrada entre diferente clases. El análisis de discriminación lineal de Fisher (FLDA) busca una dirección **w** que separe los medios de las clases (cuando se proyecta sobre la dirección encontrada), mientras consigue una pequeña varianza en torno a estos medios [MIAN,2010]. En el caso lineal para un problema de *M*-clase, el objetivo es encontrar una dirección que maximiza la varianza entre clases en razón de la varianza de clase definida en la ecuación (2.55).

$$J(w) = \frac{w^t \cdot S_B \cdot w}{w^t \cdot S_w \cdot w} \tag{2.55}$$

Donde, S_B se define como la matriz de varianza entre clases y S_w como la matriz de varianza por clases y se calculan como se muestra en las ecuaciones (2.56) y (2.57), respectivamente.

$$S_B = \sum_{j=1}^{M} n_j \cdot (m_j - m) \cdot (m_j - m)^T \tag{2.56}$$

$$S_w = \sum\nolimits_{x_i \epsilon C_j} (x_i - m_j) \cdot (x_i - m_j)^T \tag{2.57}$$

Donde, m_j y n_j son la media y el número de datos de entrenamiento de la clase C_j, respetivamente, y m es la media de todos los datos de entrenamiento.

La dirección **w** se puede determinar resolviendo la ecuación (2.59).

$${S_w}^{-1} \cdot S_B \cdot w = \lambda \cdot w \tag{2.59}$$

Donde, λ son los valores propios.

En las figuras 2.5 y 2.6 se muestra un problema de cuatro clases sin aplicar FLDA y aplicando FLDA, respectivamente.

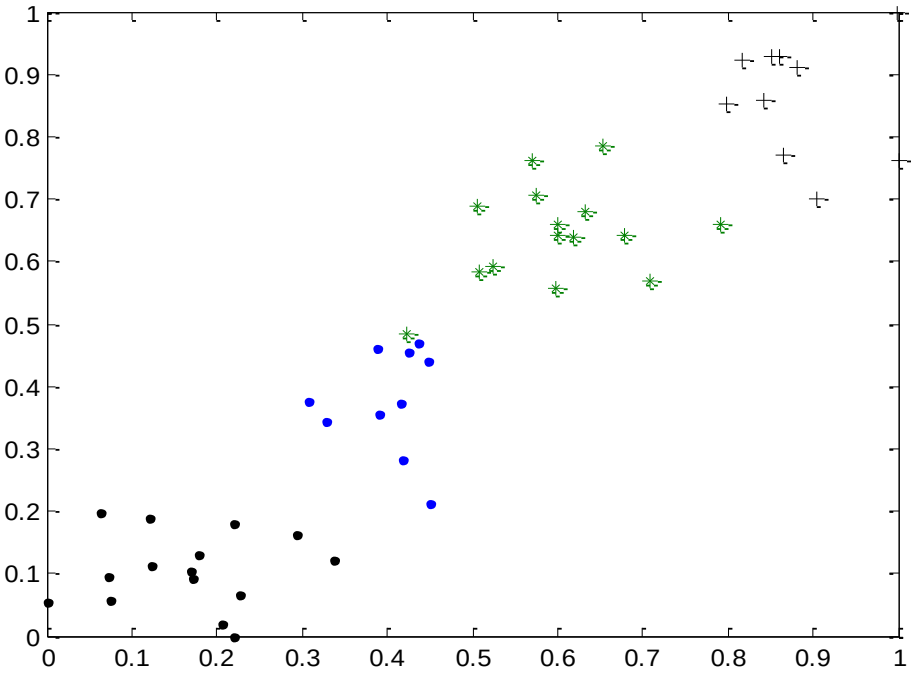


Figura 2.5. Ejemplo de cuatro clases sin aplicar FLDA

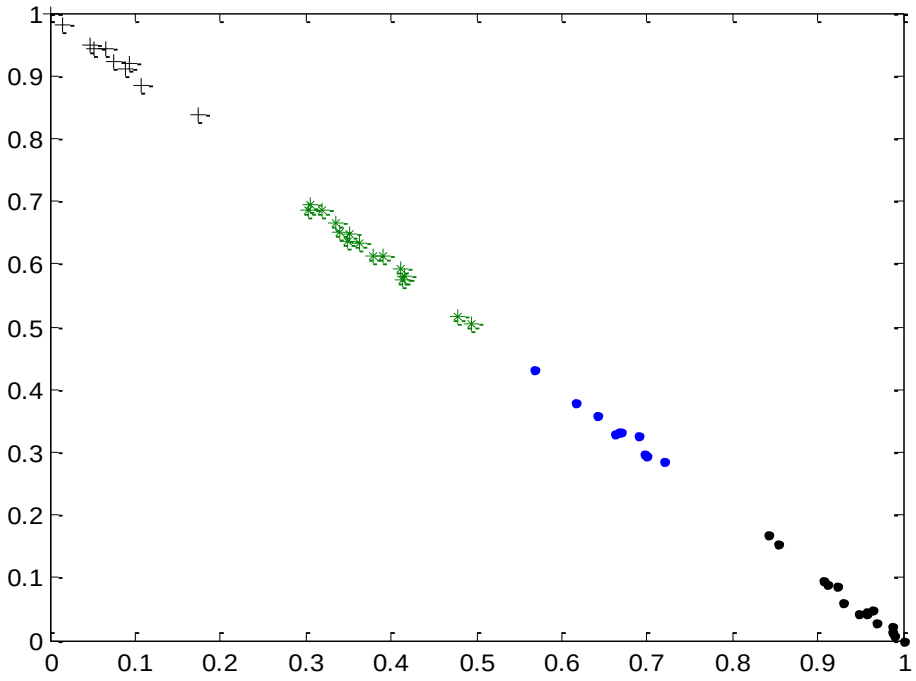


Figura 2.6. Ejemplo de cuatro clases aplicando FLDA

2.6 Análisis de sensibilidad

El análisis de sensibilidad es un diseño de experimentos, que pueden ser analíticos o numéricos, con los que se pretende cuantificar la incertidumbre que se producen en el modelo por las variaciones de las variables de salida respecto a las variables de entrada.

Este análisis es útil debido a que puede mostrar errores en la estructura del modelo, determinar regiones críticas en el espacio de las variables de entrada con el objetivo de establecer prioridades y/o realizar simplificaciones en la investigación [CRIS,2011].

Un análisis de sensibilidad clásico está compuesto por tres etapas, las cuales están sujetas a cambio si el análisis que se desea realizar lo requiere [MARI,2013].

- Muestreo del conjunto de datos.
- Evaluación de la muestra en el modelo.
- Análisis del conjunto de datos de muestra-resultados mediante una técnica de sensibilidad.

En la figura 2.7 se presenta un esquema general de un análisis de sensibilidad. En la figura 2.7 se muestra la búsqueda sobre un conjunto de puntos definidos por una técnica de muestreo y su respectivos valores de salida $y(i)$. Estos datos se deben analizan por medio de una técnica de senbilidad, la cual determina los efectos de los factores seleccionados, para este caso los parámetros son definidos por *x1, x2 … xm*.

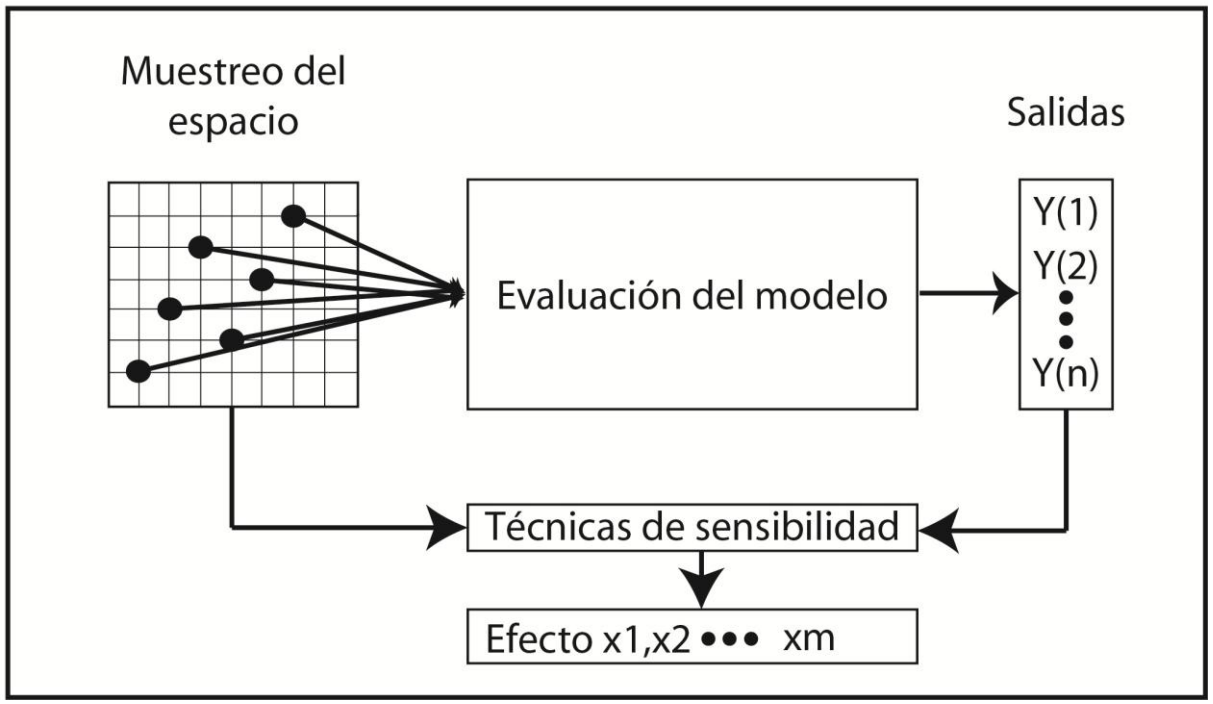


Figura 2.7. Esquema general del análisis de sensibilidad

2.6.1 Métodos de muestreo

Los métodos de muestreo consisten en estimar parámetros poblacionales a partir del estudio del modelo. Estos métodos incluyen generación y exploración de cada parámetro en un espacio determinado para cada variable de entrada, y su principal objetivo es obtener un vector de salidas del modelo $y(i)$ [MARI,2013].

En la literatura existen diferentes metodologías que proponen un muestreo sobre el conjunto de incertidumbres, entre éstas se destacan: muestreo aleatorio, muestreo por *latin hypercube*, muestreo por simulación de Montecarlo. Debido a las características del problema a tratar, se seleccionó como método de muestreo el *latin hypercube* óptimo, ya que este método no depende del modelo matemático y garantiza el buen desempeño de la técnica de sensibilidad utilizada [FANG,2006] [MARI,2013]. La metodología del *latin hypercube* óptimo utilizado en esta tesis es la que se presenta [MARI,2013].

2.6.2 Evaluación del modelo

Una vez se define la técnica y la muestra, es necesario evaluar las muestras sobre el modelo. Estas evaluaciones representan finalmente el vector (*y(i)*) en las variables de salida. Si las evaluaciones en el modelo en el cual se está trabajando son computacionalmente costosas, se recomienda hacer una reducción en las evaluaciones del modelo [VIAN,2009].

2.6.3 Técnica de sensibilidad

La técnica utilizada para sensibilidad se denomina análisis de regresión, y se basa en determinar los parámetros de modelado que más contribuyen a la variabilidad de la salida.

El análisis de regresión se representa por medio de un modelo de la forma que se muestra en (2.60), donde $\boldsymbol{x_j}$ son las variables de entrada que se están considerando, $\boldsymbol{b_j}$ son los coeficientes de regresión que deben ser determinados y *s* denota el número de variables de entrada.

$$\hat{y} - \bar{y} = \sum_{j=1}^{n} \frac{b_j * \hat{S}x_j}{\hat{S}y} * \frac{x_j - \bar{x}_j}{\hat{S}x_j} \tag{2.60}$$

En donde $\bar{y}$ es la media del vector de resultados, y $\bar{x}_j$ es la media del vector columna de variación de cada parámetro.

Los términos $\bar{y}$,$\bar{x}_j$, $\hat{S}x_j$ *y* $\hat{S}y$ se presentan en (2.61), (2.62), (2.63) y (2.64) respectivamente.

$$\bar{y} = \sum_{k=1}^{n} \frac{y_k}{n} \tag{2.61}$$

$$\bar{x}_j = \sum_{k=1}^{n} \frac{x_{kj}}{n} \tag{2.62}$$

$$\hat{S}x_j = \sqrt{\sum_{k=1}^{n} \frac{(x_{kj} - \bar{x}_j)^2}{n-1}} \tag{2.63}$$

$$\hat{S}y = \sqrt{\sum_{k=1}^{n} \frac{(y_k - \bar{y})^2}{n-1}} \tag{2.64}$$

Los coeficientes $b_{j*}Sx_j/Sy$ son llamados coeficientes independientes estandarizados de regresión o coeficientes *beta*. El valor absoluto de los coeficientes *beta*, indica la importancia de los parámetros x_j sobre el modelo, si y sólo si, éstos son independientes [SALT,2000].

Capítulo 3.

Metodología propuesta

3.1 Introducción

Como alternativa de solución al problema de localización de fallas en sistemas de distribución, se propone un localizador robusto basado en máquinas de soporte vectorial, con el fin de tener una metodología más robusta y precisa. Para esto se va realizar un análisis de sensibilidad utilizando el método de *Latin hypercube* óptimo y un análisis de regresión con el objetivo de determinar cuáles son los parámetros que más afectan a las SVM en el problema de localización de fallas en sistemas de distribución.

Esta metodología se desarrolló con ayuda del software Matlab ® para implementar los algoritmos necesarios. Para la simulación de circuitos eléctricos se utilizó el software ATP y el software *simulaciónRF* que permite una rápida obtención de las bases de datos necesarias para el proceso de clasificación.

Este capítulo se divide en dos secciones principales. En la sección 3.2 se presenta la metodología seguida para implementar MBC en el problema de localización de fallas. En la sección 3.3 se muestra como se implementó el LH óptimo para crear los escenarios a evaluar.

3.2 Estrategia general

La metodología general utilizada para el clasificador robusto basado en las SVM con el propósito de la localización de fallas en sistemas de distribución, se presenta en el esquema mostrado en la figura 3.1. Este esquema consta de tres fases que se realizan fuera de línea.

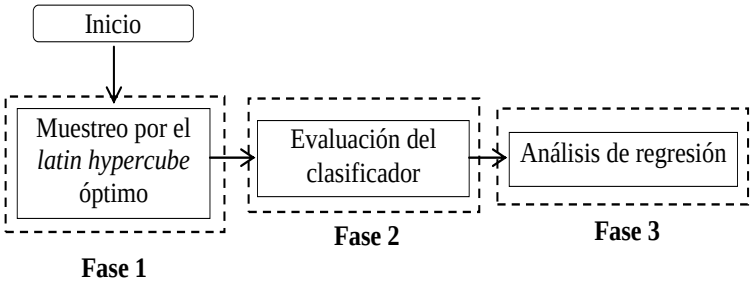


Figura 3.1 Esquema general de la metodología propuesta

El esquema mostrado en la figura 3.1 permite hacer el análisis de sensibilidad al localizador.

3.2.1 Fase 1: *Latin hypercube* óptimo

En esta tesis para resolver el problema del *latin hypercube* óptimo se utilizó búsqueda tabú como técnica de optimización, el algoritmo básico utilizado se muestra en la figura 3.2 [GALL,2008].

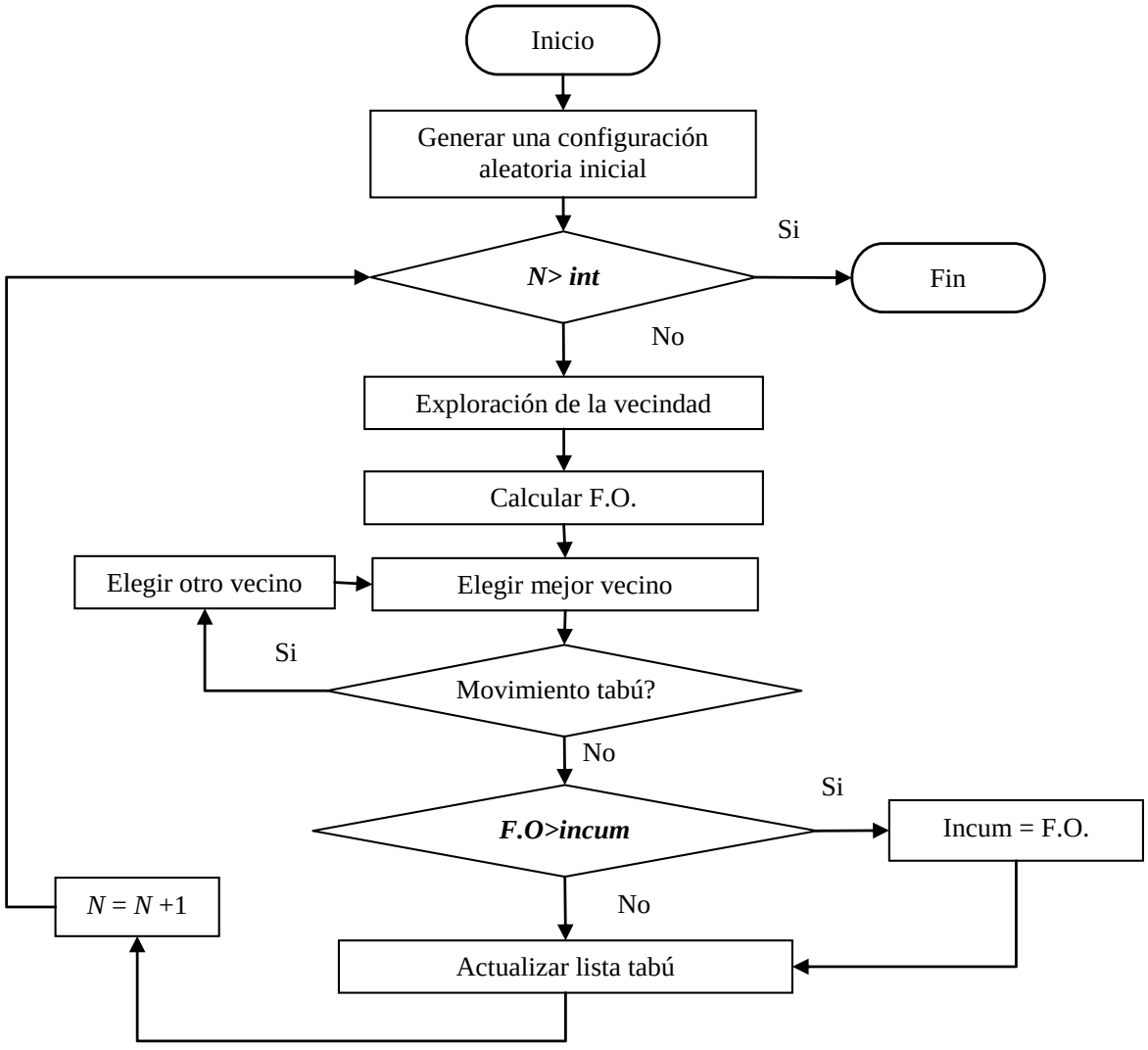


Figura 3.2 Diagrama de flujo básico para el algoritmo TS.

Se utilizó una codificación de coordenadas, la cual no entra a configuraciones infactibles, es decir que en un mismo nivel se encuentren 2 o más puntos [MARI,2013]. En la figura 3.3 se muestra un ejemplo de un escenario del LH.

1	2	4	4	3
4	4	1	1	1
1	4	1	2	3
⋮	⋮	⋮	⋮	⋮
3	2	4	4	4
3	4	4	2	3

Figura 3.3 Escenario de LH

Cada columna hace referencia a un parámetro diferente, los cuales tienen un rango de variación de 1 a 4. Para implementar el problema del LH, el rango de variación de cada parámetro debe ser igual. Los parámetros considerados en este análisis de sensibilidad son:

- **Tipos de normalización**

1. Min-Max
2. Z-score
3. Median and median absolute deviation
4. Sigmoidal funtion
5. Min-Max con FLDA
6. Z-score con FLDA
7. Median and median absolute deviation con FLDA
8. Sigmoidal funtion con FLDA

- **Atributos**

1. dV, dI, $d\theta_V$
2. dV, dI, $d\theta_I$
3. dV, $d\theta_V$, $d\theta_I$
4. dV, dI, $d\theta_V$, $d\theta_I$
5. dV_L, dI_L, $d\theta_{VL}$
6. dV_L, dI_L, $d\theta_{IL}$
7. dI_L, $d\theta_{VL}$, $d\theta_{IL}$
8. dV_L, d_{IL}, $d\theta_{VL}$, $d\theta_{IL}$

- **Kernel**

1. Polinomial
2. Función de base radial
3. Sigmoide
4. Radial Laplace

- **Descomposición/reconstrucción**
  1. Uno contra uno (1-v-1)
  2. Uno contra el resto(1-v-all)
  3. ECOC
  4. Uno contra uno con árbol de decisión (DDAG)

- **Modelos de la carga**

  1. Impedancia constante
  2. Potencia constante
  3. Corriente constante
  4. Híbrido

El tipo de normalización y los atributos se trabajaron en dos grupos de cuatro parámetros, debido a que el LH necesita que el rango de variación de los parámetros sea igual. Para realizar todas las combinaciones posibles se corre cuatro LH óptimo por circuito.

El algoritmo comienza con una configuración inicial aleatoria, la cual está compuesta de *n* puntos. A esta configuración se calcula la función objetivo como se muestra en (2.66) y su resultado se almacena como incumbente. Inicialmente, se halla la distancia entre todos los puntos de la configuración actual del LH y se selecciona los dos más cercanos entre sí, si existen varios puntos con igual valor se escoge aleatoriamente uno de ellos. Con estos dos puntos se crea la vecindad como se muestra en la figura 3.4.

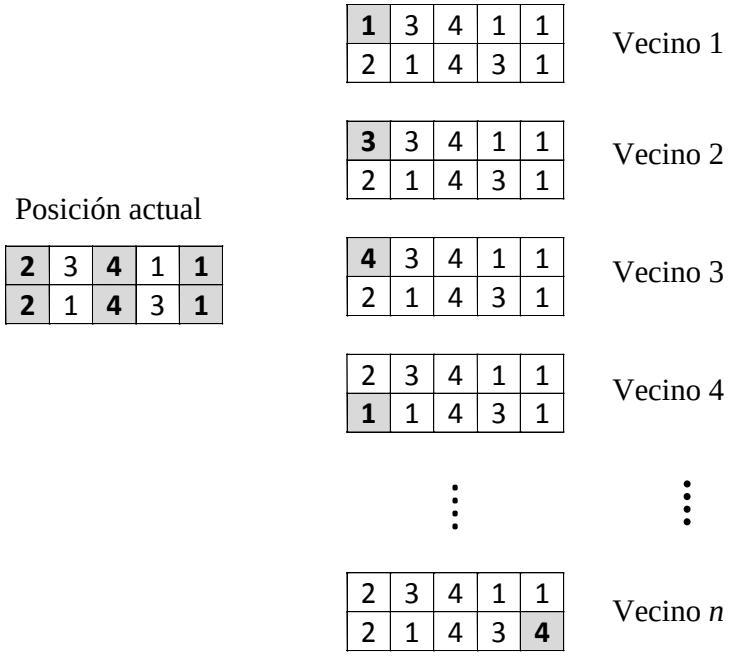


Figura 3.4 Vecindad

Las posiciones a cambiar son los valores de columnas que tienen igual valor. Por ejemplo, en la figura 3.4 se observa que los valores de las columnas uno, tres y cinco son iguales.

Los vecinos se generan cambiando los valores de cada posición de estas columnas, por los valores de rango de variaciones de ese parámetro. Por ejemplo, la posición (1,1) puede ser modificada por tres valores que corresponde a Vecino 1, Vecino 2 y Vecino 3, y así sucesivamente con el resto de posiciones. De esta manera se obtiene un conjunto de vecinos de la posición actual.

Una vez generada la vecindad se evalúa la función objetivo de cada escenario. Se elige la mejor solución de la vecindad que no esté contenida en la lista tabú, si existen varias soluciones con igual valor se escoge aleatoriamente una de ellas. Se compara ésta solución con la solución global, y si ésta es menor se guarda como la nueva incumbente. El vecino elegido se guarda en la lista tabú de tamaño *M* y queda restringido por las próximas *M* iteraciones (donde *M* es igual a 4). El criterio de parada utilizado es un límite de iteraciones.

3.2.2 Fase 2: Evaluación del modelo

La metodología general utilizada para el clasificador robusto basado en las SVM con el propósito de la localización de fallas en sistemas de distribución, se presenta en el esquema mostrado en la figura 3.5. Este esquema consta de cuatro etapas que se realizan fuera de línea, después de tener el clasificador calibrado ante una falla en el sistema de distribución, el localizador da una respuesta inmediata.

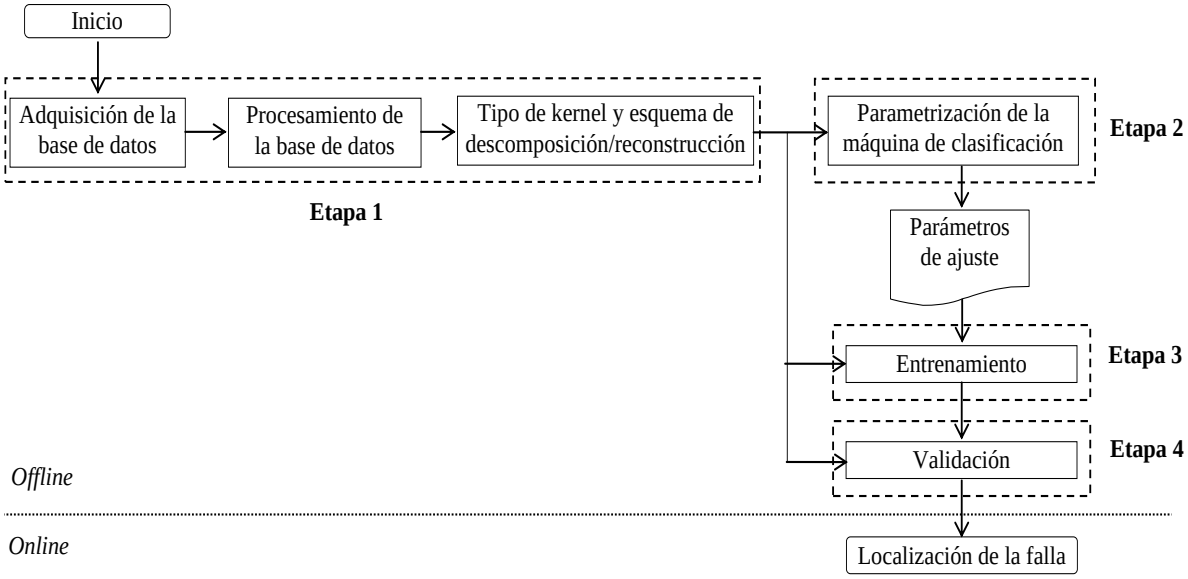


Figura 3.5 **Esquema general del localizador**

3.2.2.1 Etapa 1

Esta etapa consiste en extraer y preparar la base de datos y seleccionar los parámetros en las variables de entrada para el MBC. Esta etapa sigue varios pasos, los cuales se describen a continuación:

3.2.2.1.1 Adquisición de datos

Los datos utilizados para el entrenamiento se obtienen de la simulación del sistema de distribución analizado, a partir de un proceso conjunto de la herramienta de simulación Alternative Transients Program (ATP) y MATLAB. Esta herramienta permite la simulación automática de condiciones de falla monofásica, bifásica y trifásica, con diferentes valores de resistencia de falla [PERE,2010]. En esta tesis se trabajó con cuatro diferentes modelos de carga, los cuales se presentan en [HERR,2013], con la idea de reducir la incertidumbre que se presenta en el valor y naturaleza de la carga en los circuitos reales y así poder obtener una localización más precisa.

3.2.2.1.2 Zonificación de la red de distribución

Los métodos de clasificación requieren que para cada uno de los datos de entrenamiento se le asigne una clase, con la cual se realizará la clasificación de un nuevo dato. Esta clase corresponde a una zona del sistema de distribución bajo análisis. Una zona no debe tener más de un lateral del circuito, para eliminar el problema de múltiple estimación [GUTI,2010]. Además, es posible proponer una zonificación con zonas grandes, si así lo requiere el operador de red, aunque también es posible reducir su tamaño y dar importancia a aquellas zonas donde la probabilidad de ocurrencia de fallos sea mayor, o donde sea necesario restaurar el servicio de forma más rápida. La zonificación utilizada en cada circuito de prueba en este proyecto es tomada de [ZAPA,2013], la cual es obtenida mediante una herramienta automática de zonificación.

3.2.2.1.3 Procesamiento de los datos

El procesamiento de datos es un paso muy importante para los problemas de clasificación, debido a que este proceso incluye extracción de características, depuración, normalización y selección de los datos, lo cual ayuda a mejorar la predicción de los MBC con menores esfuerzos computacionales. Si los datos no son procesados pueden ser inconsistentes, o contener información errónea e irrelevante para los MBC, lo cual puede producir menor precisión en los resultados [KOTS,2006].

El primer paso del procesamiento de los datos está asociado con la extracción de información significativa denominada atributos, para este problema son registros de fallas tomados en la subestación. En esta investigación, los atributos utilizados corresponden a la variación en magnitud de la tensión (dV) y de la corriente (dI), y la variación angular de la tensión ($d\theta_V$) y de la corriente de fase ($d\theta_I$). Para cada atributo se consideran medidas de fase y de línea. Se analizan sólo las combinaciones que contienen atributos de tensión y corriente: $dV,dI,d\theta_V,d\theta_I$ debido a que estas combinaciones han presentado buenos resultados como se muestra en [GIL,2013]. Adicionalmente, a cada atributo se le asigna una etiqueta relacionada con la zona en la cual ocurrió la falla [GUTI,2010].

Finalmente, la base de datos se normaliza, debido a que puede existir gran diferencia entre los valores de un mismo atributo [KOTS,2006], o entre los valores de diferentes atributos. Con la normalización se evita que atributos con magnitudes más altas dominen en el cálculo de la zona. Adicionalmente, esta normalización puede disminuir el costo

computacional y mejorar el desempeño y la precisión del método de clasificación [SHAL,2006] [SOLA,1997] [SNEL,2005].

Otros problemas requieren el manejo de ruido, detección de datos anómalos y estrategias para manejar datos faltantes. Sin embargo, ninguna de estas tareas se realiza en este caso debido a que las bases de datos obtenidas mediante simulación están completas.

3.2.2.1.4 Tipo de kernel y esquema de descomposición/reconstrucción

En esta parte de la metodología se escoge un kernel y un esquema de descomposición/reconstrucción como entrada a la etapa dos.

En esta tesis se utilizan los cuatro tipos de kernels, los cuales se encuentran en los más utilizados para las SVM [SALA,2012], los cuales se presentan en las ecuaciones (2.34) a (2.37). Además, se utilizan cuatros esquemas de descomposición/reconstrucción los cuales se explican en la sección 2.3.4.

3.2.2.2 Etapa 2

En esta etapa se determinan los parámetros óptimos para el clasificador (C y γ, C donde parámetro de penalización propio de las SVM y γ es el parámetro del kernel), se implementa la búsqueda tabú (TS) mediante la técnica de validación cruzada. El error de validación cruzada se utiliza como la función objetivo de la técnica de optimización, y así el algoritmo evoluciona hasta encontrar el menor error de validación.

La validación cruzada es un método que consiste en dividir la base de datos de entrenamiento en n partes iguales. A continuación, el localizador basado en SVM se entrena con los datos contenidos en las $n-1$ partes de la base de datos y se utiliza la parte restante para hallar el error de validación, calculada como se muestra en la ecuación (3.1). Este proceso se repite n veces, lo que permite utilizar todas las muestras para hallar un error de validación con esta base de datos. Por último, se promedian los n valores de error encontrados y se obtiene un sólo error de validación [MORA,2005].

$$Error\ de\ validación = \frac{Número\ de\ datos\ mal\ clasificados}{Número\ de\ datos\ de\ prueba} \cdot 100\% \qquad (3.1)$$

La figura 3.6 muestra el algoritmo básico utilizado para la búsqueda de los parámetros óptimos del clasificador, la técnica basada en TS comienza con 20 puntos iniciales aleatorios, a los cuales se les calcula la función objetivo y el mejor resultado se almacena como incumbente. Inicialmente, se crea la vecindad alrededor del punto inicial, determinada como se muestra en la ecuación (3.2) y se evalúa la función objetivo de cada punto. El punto actual se guarda en la lista tabú de tamaño M y queda restringido por las próximas M iteraciones (en esta investigación se utiliza M igual a 10, debido a que con este valor se han obtenido buenos resultados para este problema) [GIL,2013]. Se elige la mejor solución de la vecindad y se compara con la solución global, y si ésta es menor se guarda

como la nueva incumbente. El criterio de parada utilizado es un límite de iteraciones y un valor mínimo de la función objetivo.

$$C_{i+1} = R_C \cdot \sin\theta + C_i \quad con \;\; R_C = \frac{límite\ máximo\ de\ C}{C'}$$
$$\gamma_{i+1} = R_\gamma \cdot \cos\theta + \gamma_i \quad con \;\; R_\gamma = \frac{límite\ máximo\ de\ \gamma}{\gamma'} \tag{3.2}$$

Los parámetros C' y γ' están relacionados con la vecindad del punto actual. En este problema se trabajó con $C' = 250$ y $\gamma' = 33$, para que las nuevas soluciones sean diferentes a la actual. El valor de θ está espaciado en 45° entre una solución y otra, para que se explore uniformemente toda la vecindad.

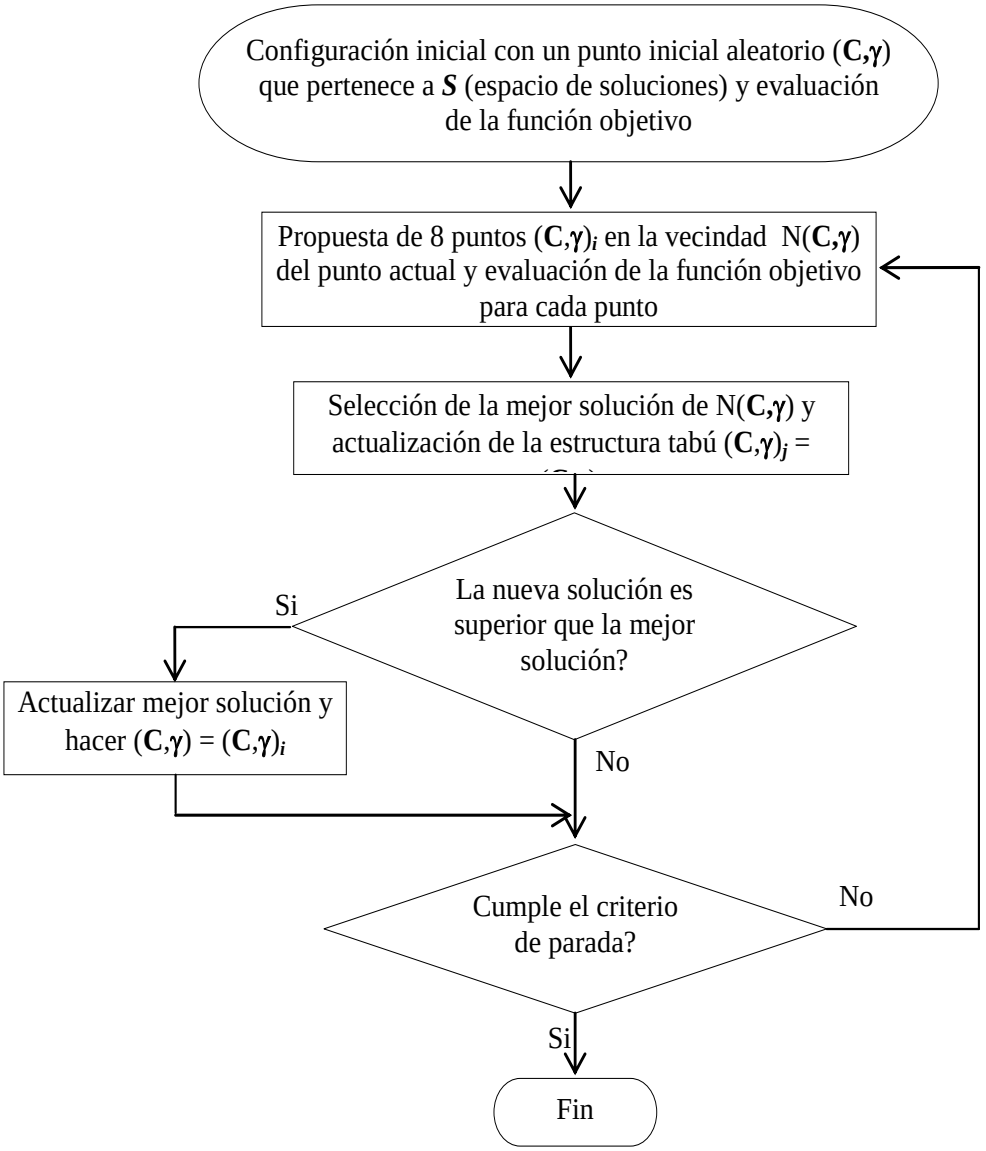


Figura 3.6 Diagrama de flujo básico para el algoritmo TS asociado al problema de parametrización de la SVM

3.2.2.3 Etapa 3

En esta etapa se utiliza los parámetros encontrados en la etapa de parametrización. El objetivo es encontrar los hiperplanos que definen la SVM, que serán utilizados en la etapa de validación.

3.2.2.4 Etapa 4

A partir de unos registros de fallas se valida la SVM, la validación consiste en evaluar el rendimiento del localizador a partir del cálculo la precisión del método ante datos desconocidos (datos que no fueron considerados en la etapa de parametrización y entrenamiento). La precisión se calcula como se muestra en la ecuación (3.3).

$$Precisión = \frac{Número\ de\ datos\ bien\ clasificados}{Número\ de\ datos\ de\ prueba} \cdot 100\% \qquad (3.3)$$

3.2.3 Técnica de sensibilidad

El análisis de sensibilidad se realiza por medio de la técnica de análisis de regresión que se muestra en la sección 2.6, en la cual la salida se representa por los coeficientes beta estandarizados de los parámetros en análisis. El cálculo de estos coeficientes se realiza para cada uno de los parámetros considerados en la sección 3.2.1. Donde, x_j son los escenarios propuestos por el latin hypercube y $y(i)$ son los calculados por la ecuación (3.3) para cada escenario.

Capítulo 4.

Aplicación de la metodología propuesta

En este capítulo se presentan los resultados obtenidos al aplicar la metodología propuesta en el capítulo 3, como una solución al problema de localización de fallas utilizando las SVM. El análisis de sensibilidad sólo se realizó para fallas monofásicas debido a la alta carga computacional de las pruebas y a que estas fallas representan aproximadamente un 70% de las fallas que se presentan en un circuito de distribución [MORA,2006].

Este capítulo se divide en dos secciones principales. En la primera sección se describe el sistema de pruebas utilizado y las condiciones operativas simuladas. En la segunda sección se muestran los resultados obtenidos del análisis de sensibilidad.

4.1 Circuitos analizados y escenarios de operación simulados

Los escenarios considerados para probar el algoritmo incluyen fallas en todos los nodos del sistema excepto en la subestación. Se simularon siete diferentes condiciones de carga además de la promedio, los escenarios simulados fueron: carga 10-30%, 30-60%, 60-100%, 100-135%, 135-145% y 10-145%. Se simularon cuatro escenarios de variación de la tensión en la subestación y cuatro escenarios de variación de la longitud de las líneas en los siguientes rangos: 95-98%, 98-102%, 102-105% y 95-105%.

Para cada escenario propuesto se utilizan diferentes resistencias de falla, las cuales toman valores entre 0,05 Ω y 40 Ω, los cuales son valores típicos que caracterizan las fallas en sistemas de distribución [DAGE,2000].

4.1.1 IEEE 34 nodos

El sistema IEEE 34 nodos se presenta en los "*test feeders*" del "*Distribution System Analysis Subcommittee*" del "*Institute Electrical and Electronics Engineers*" [IEEE,2000], el cual es un circuito de distribución real ubicado en Arizona, operado a 26,7 kV. El sistema presenta cargas desbalanceadas, laterales monofásicos y múltiples calibres de conductor. La zonificación de este circuito es tomada de [ZAPA,2013] (el circuito se divide en 12 zonas), tal como se muestra en la figura 4.1.

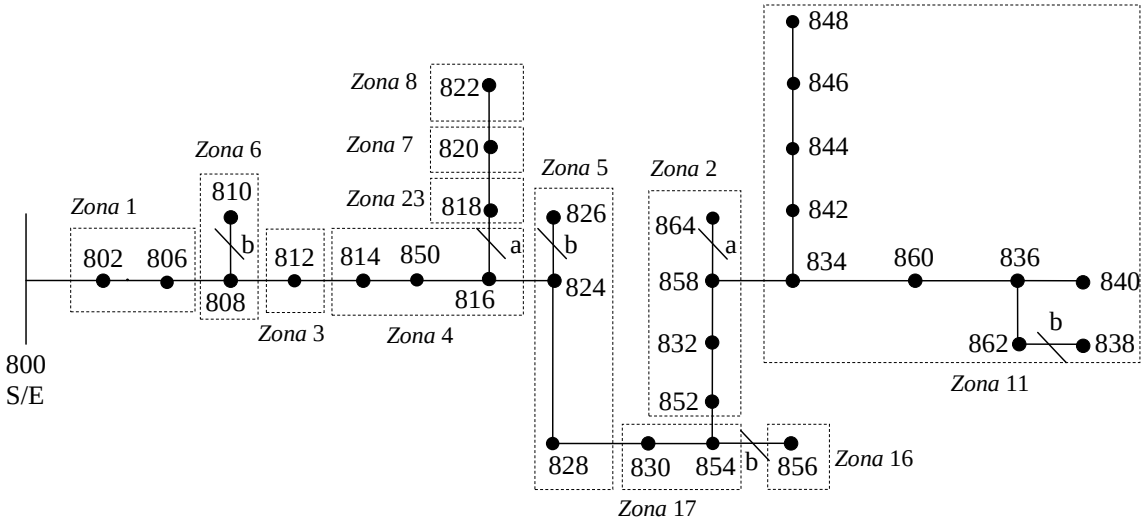


Figura 4.1 Zonificación automática sistema IEEE 34 nodos.

4.1.2 Circuito de 44 nodos

Este circuito de prueba está presentado en [HERR,2013], opera a 34,5 kV, tiene 3 laterales y está diseñado a partir de un circuito de distribución real en Bogotá, Colombia. Todos los laterales del circuito son trifásicos. La zonificación de este circuito es tomada de [ZAPA,2013] (el circuito se divide en 13 zonas), tal como se muestra en la figura 4.2.

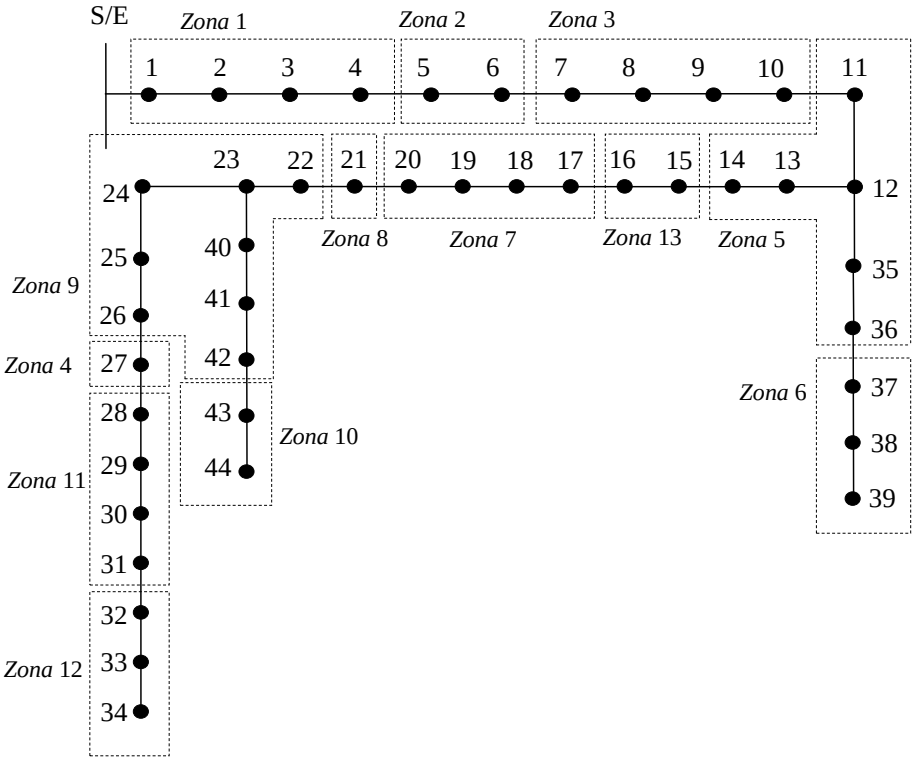


Figura 4.2 Zonificación automática circuito de 44 nodos.

4.2 Resultados con el circuito de prueba IEEE34 nodos

Mediante la metodología mostrada en la sección 3.3, El *latin hypercube* óptimo creó 400 escenarios, donde cada punto es una combinación de parámetros para las SVM. Para cada punto obtenido mediante el *latin hypercube* óptimo se realizó la parametrización, entrenamiento y validación.

4.2.1 Parametrización

En esta parte se parametriza cada punto del *latin hypercube* óptimo como se muestra en la sección 3.2.2., para cada parametrización se utilizaron fallas simuladas a condición nominal. La base de datos está conformada por 540 casos simulados con resistencia de fallas de 8Ω a 40Ω en pasos de 8Ω.

4.2.2 Entrenamiento y validación

En esta parte se consideran todas las variaciones de carga propuestas en la sección 4.1. La base de datos para entrenamiento contiene 4455 datos para cada modelo de carga y está compuesta con valores de resistencia de fallas entre 0,05Ω a 40Ω en pasos de 4Ω. La base de datos de la validación está conformada por 12150 casos simulados para cada modelo de carga con resistencia de falla entre 1Ω a 39 Ω en pasos de 1Ω (sin incluir los casos ya utilizados en el entrenamiento).

4.2.3. Análisis de sensibilidad

En esta parte teniendo la evaluación de cada punto se calculó los coeficientes *betas* de la técnica de análisis de regresión que se presenta en la sección 2.6. En la figura 4.3 se muestran los valores de los coeficientes *betas* para el circuito IEEE34 nodos.

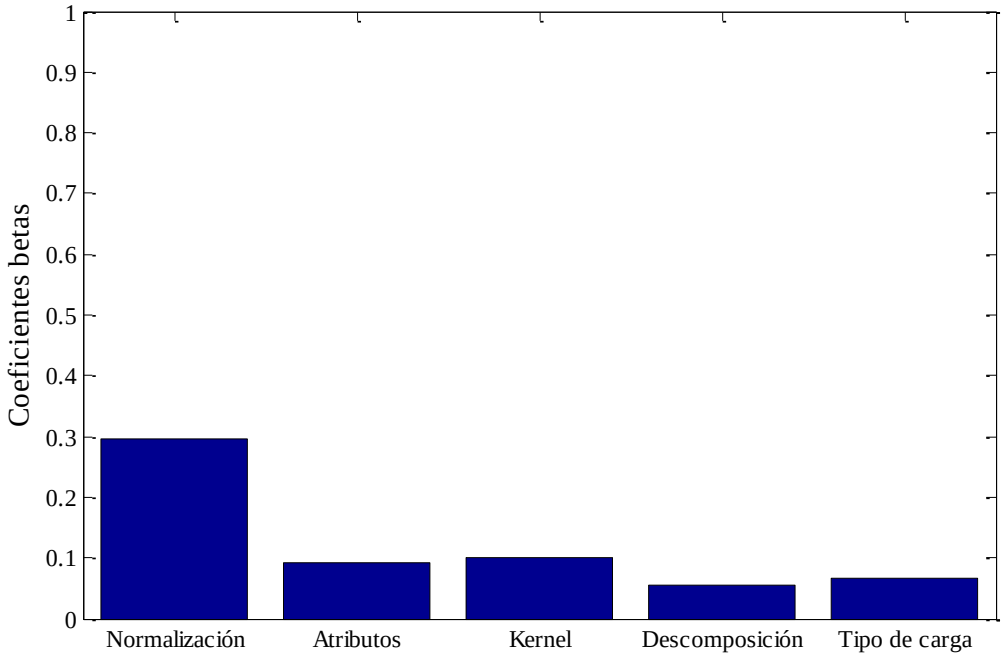


Figura 4.3 Coeficiente beta para cada parámetro.

En la figura 4.3 se puede observar que el parámetro que más afecta en el desempeño de la SVM para la localización de fallas es la normalización.

De los ochos tipos de normalización utilizados, presentados en la sección 3.3, se aprecia que al utilizar FLDA la SVM obtiene resultados de precisión menor, por tal motivo esta técnica no se va a utilizar en este problema. Se obtuvo un resultado promedio 83,61% utilizando FLDA y un resultado promedio de 92,83% sin utilizar FLDA en la normalización. Adicionalmente, se tiene un tiempo de simulación mayor con FLDA, aproximadamente 7.6 veces más en promedio.

En la figura 4.4 se muestra el desempeño promedio de los métodos de normalización sin FLDA en las SVM. Se observa en la figura 4.4 que el método de normalización SF presentó mejor desempeño que el resto de métodos utilizados con una precisión promedio de 98,16%. En la figura 4.5 se muestran los desempeños promedio para los kernels utilizados en la tesis solamente utilizando el método de normalización SF.

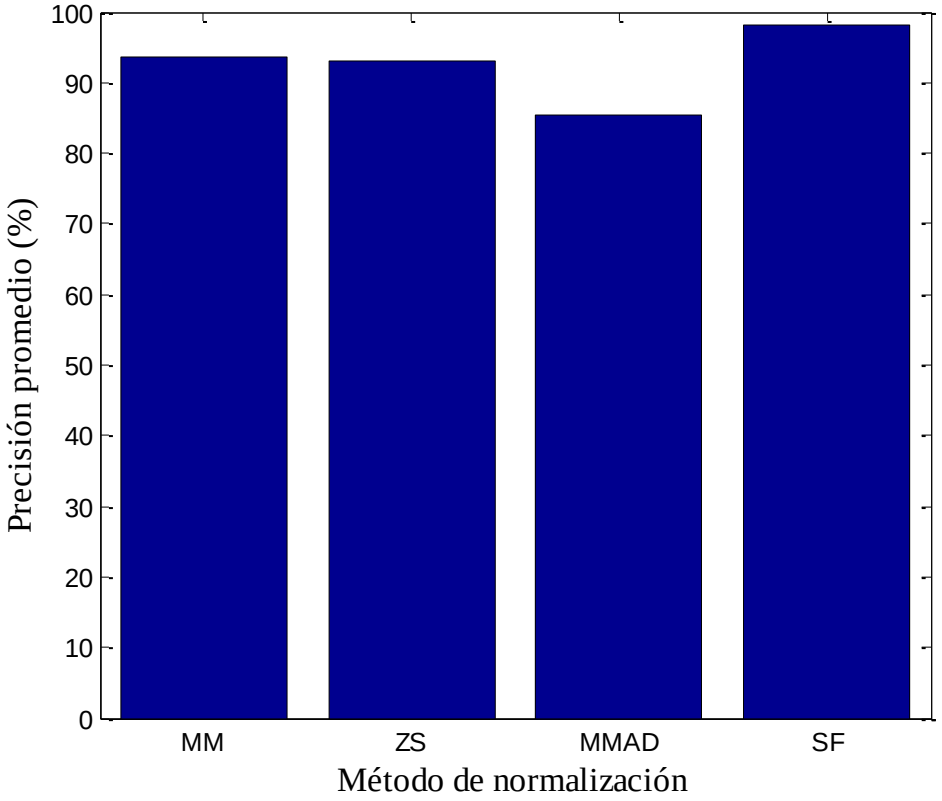


Figura 4.4 Precisión promedio de cada método de normalización.

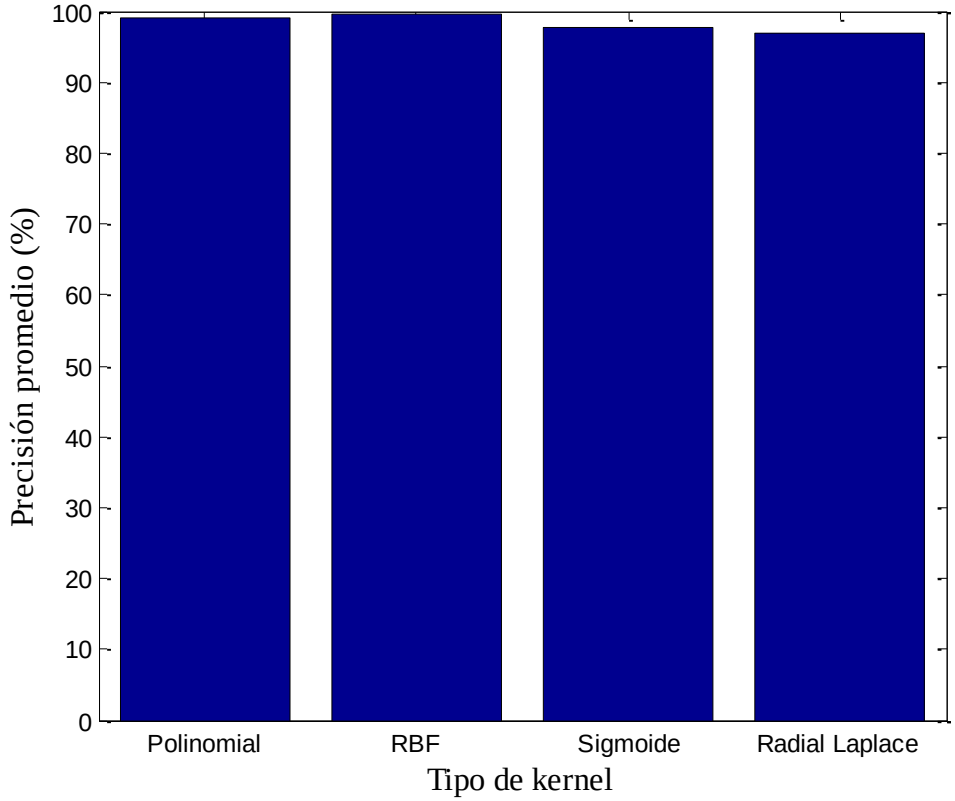


Figura 4.5 Precisión promedio de los kernels con normalización SF

Se puede notar en la figura 4.5 que aunque todos los kernels tienen un buen desempeño, el kernel RBF obtiene mejores resultados con una precisión promedio de 99,59%. Como se observó en la figura 4.3, los parámetros correspondientes a combinación de atributos y al esquema de descomposición no influyen mucho en el desempeño de las SVM, debido a esto para seleccionar que parámetro utilizar se observó el tiempo computacional de cada uno. Adicionalmente, se observa que el tipo de carga utilizado en el modelo del circuito tampoco afecta el desempeño del método. En las figuras 4.6, 4.7 y 4.8 muestran el tiempo computacional promedio en segundos para los atributos, descomposición y tipo de carga, respectivamente.

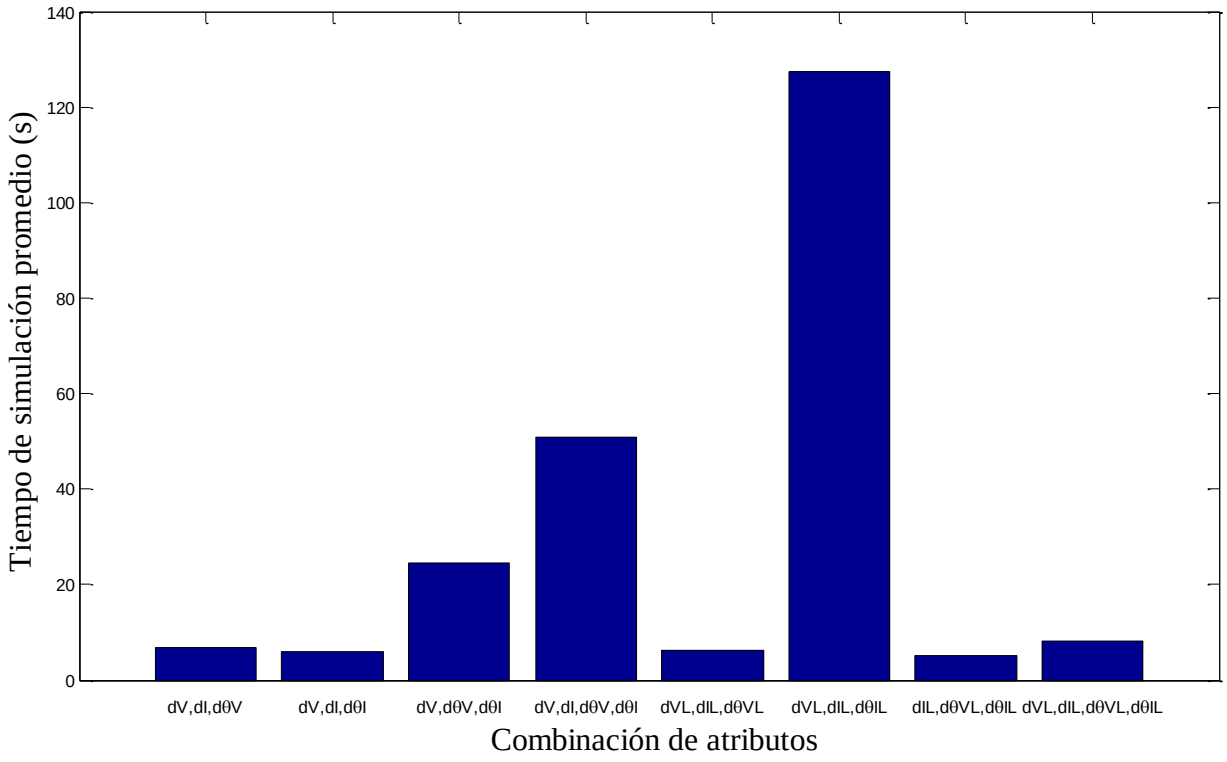


Figura 4.6 Tiempo computacional promedio de las combinaciones de atributos

Se observa en la figura 4.6, la combinación de atributos *dIL, dθVL, dθIL* es la que tuvo menor costo computacional con un tiempo promedio de 5,13 segundos. En la figura 4.7 se observa que la descomposición ECOC es la que tiene mayor costo computacional. Esto se debe a que se entrenan muchos más biclasificadores en comparación con los otros esquemas. Adicionalmente, para cada biclasificador se utilizan todos los datos de entrenamiento en la base de datos. La descomposición 1-vs-1 es la que tiene menor costo computacional que los otros esquemas utilizados en esta tesis con un tiempo promedio de 15,9 segundos.

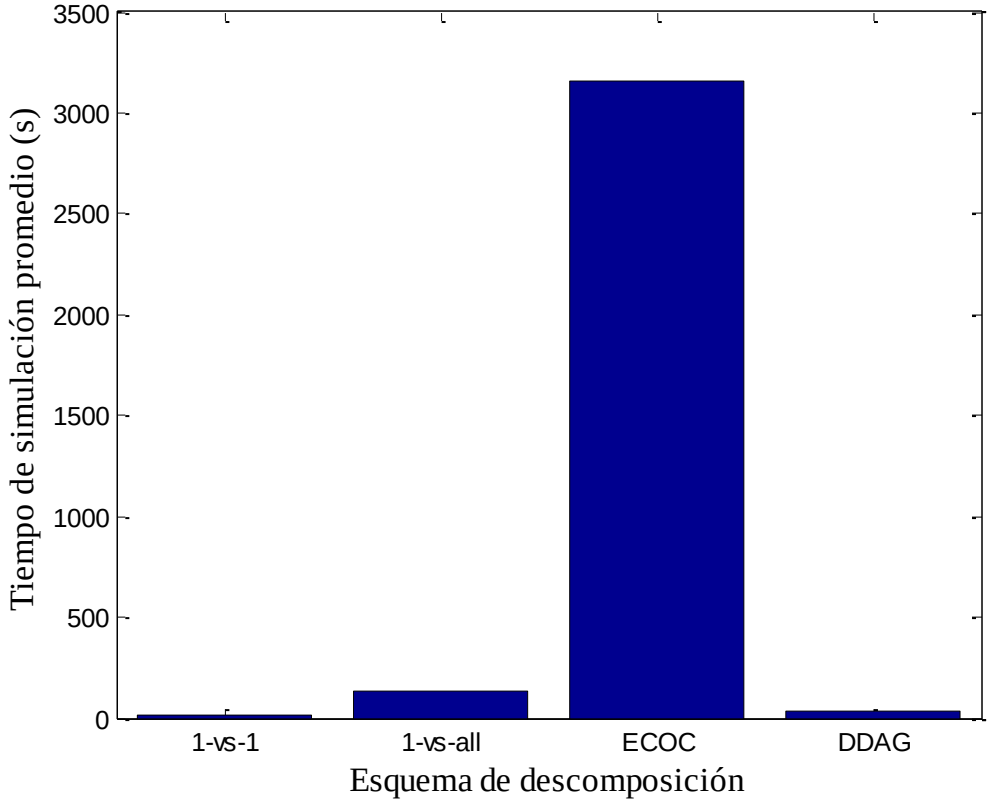


Figura 4.7 Tiempo computacional promedio del esquema de descomposición

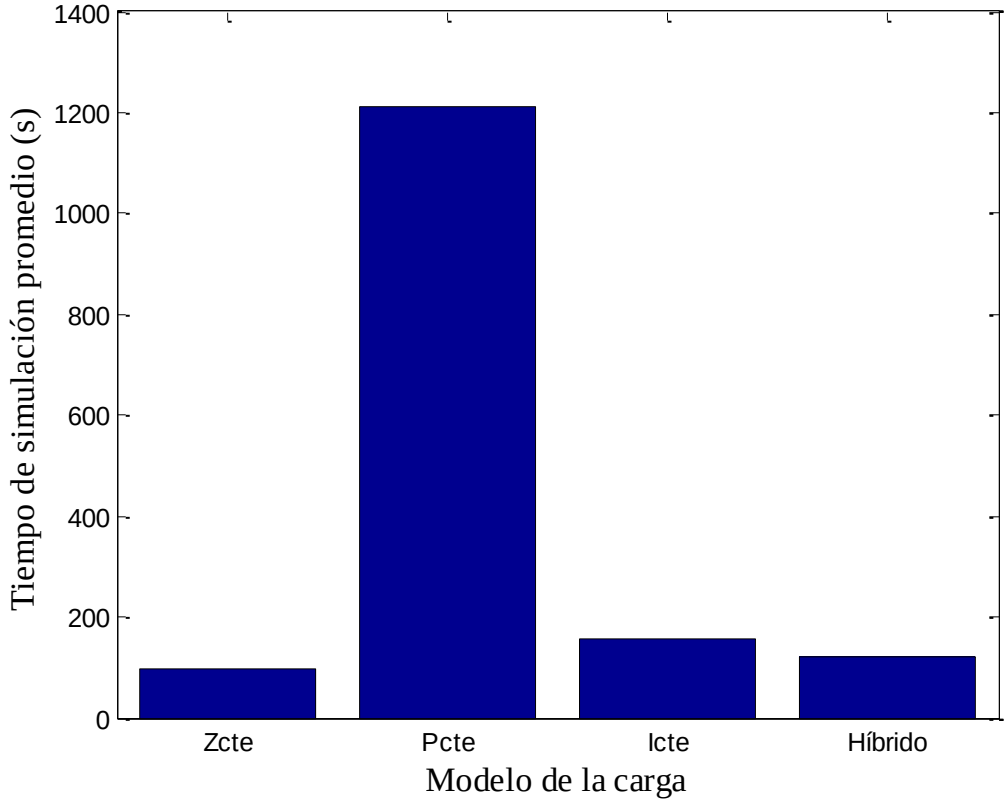


Figura 4.8 Tiempo computacional en segundos del tipo carga

En la figura 4.8 se observa que cuando se utilizan bases de datos obtenidas en el circuito modelado con tipo de carga impedancia constante, las SVM realiza un menor esfuerzo computacional, con un tiempo promedio de 97,9 segundos.

Después de tener los parámetros fijo del localizador se probó el desempeño para todos los tipos de fallas, primero se parametriza como se muestra en la sección 4.2.1. Se obtiene una base de datos completa para entrenamiento de 39105 casos considerando resistencias de falla de 0,05 a 40Ω en pasos de 4Ω. La base de datos de prueba contiene 106650 casos con resistencia de falla entre 1 a 39Ω en pasos de 1Ω, sin incluir los casos utilizados en la parametrización. Los resultados del desempeño del localizador se muestran en la tabla 4.1.

Tipo de falla	No. datos		Precisión (%)
	Entrenamiento	Prueba	
Monofásica a-g	4455	12150	100
Monofásica b-g	4290	11700	100
Monofásica c-g	3795	10350	100
Bifásica a-b	3795	10350	100
Bifásica b-c	3795	10350	100
Bifásica c-a	3795	10350	100
Bifásica a tierra a-b-g	3795	10350	100
Bifásica a tierra b-c-g	3795	10350	100
Bifásica a tierra c-a-g	3795	10350	100
Trifásica a-b-c	3795	10350	100

Tabla 4.1 Resultados de validación del circuito IEEE 34 con diferentes condiciones de operación.

Se nota en la tabla 4.1 que el clasificador tiene un gran desempeño para todos los tipos de fallas con precisiones de 100%. Adicionalmente, a este circuito se le realizaron otras pruebas, que consistió en variar todos los parámetros del circuito al mismo tiempo (ver anexo D).

4.3 Resultados con el circuito de prueba 44 nodos

Mediante la metodología mostrada en la sección 3.3, El *latin hypercube* óptimo creó 400 escenarios, donde cada punto es una combinación de parámetros para SVM. Para cada punto del *latin hypercube* óptimo se realizó la parametrización, entrenamiento y validación.

4.3.1 Parametrización

En esta parte se parametriza cada punto del *latin hypercube* óptimo como se muestra en la sección 3.2.2., para cada parametrización se utilizaron fallas simuladas a condición nominal. La base de datos está conformada por 880 casos simulados con resistencia de fallas de 8Ω a 40Ω en pasos de 8Ω.

4.3.2 Entrenamiento y validación

En esta parte se consideran todas las variaciones de carga propuestas en la sección 4.1. La base de datos para entrenamiento contiene 7260 datos para cada modelo de carga y está compuesta con valores de resistencia de fallas entre 0,05Ω a 40Ω en pasos de 4Ω. La base de datos de la validación está conformada por 19800 casos simulados para cada modelo de carga con resistencia de falla entre 1Ω a 39Ω en pasos de 1Ω (sin incluir los casos ya utilizados en el entrenamiento).

4.3.3. Análisis de sensibilidad

En esta parte teniendo la evaluación de cada punto se calculó los coeficientes *betas* de la técnica de análisis de regresión que se presenta en la sección 2.6. En la figura 4.9 se muestran los valores de los coeficientes *betas* para el circuito 44 nodos.

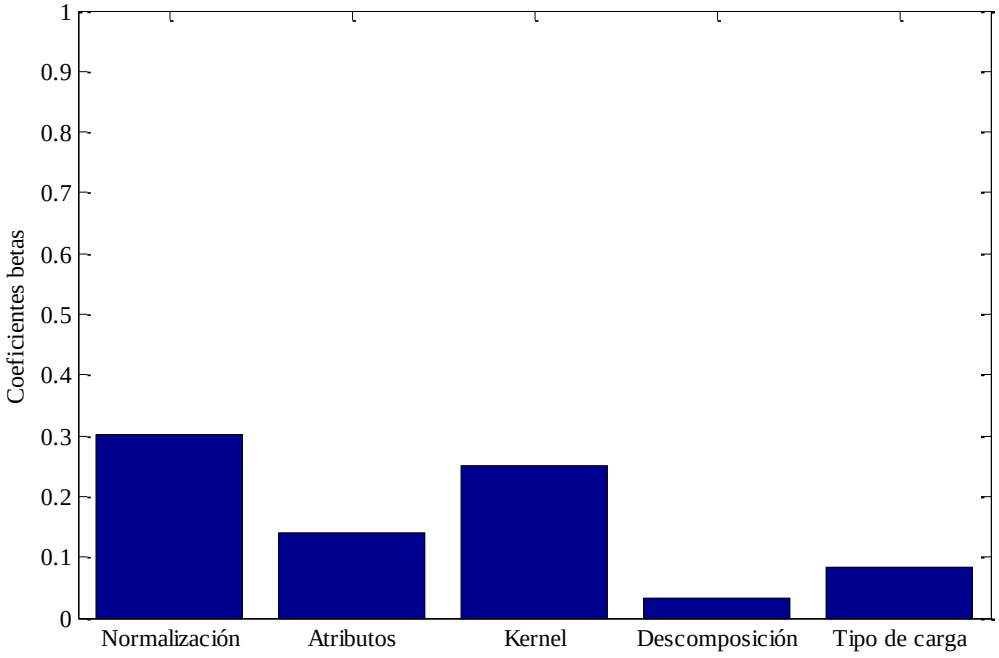


Figura 4.9 Coeficiente beta para cada parámetro.

En la figura 4.9 se puede observar que el parámetro que más afecta en el desempeño de la SVM para la localización de fallas es la normalización.

De los ochos tipos de normalización utilizados, presentados en la sección 3.3, se notó en las prueba que al utilizar FLDA se tiene resultados de precisión menor comparado en no utilizarlo. Se obtuvo un resultado promedio 71,16% utilizando FLDA y un resultado promedio de 86,05% sin utilizar FLDA en la normalización.

En la figura 4.10 se muestran las precisiones promedios para cada método de normalización.

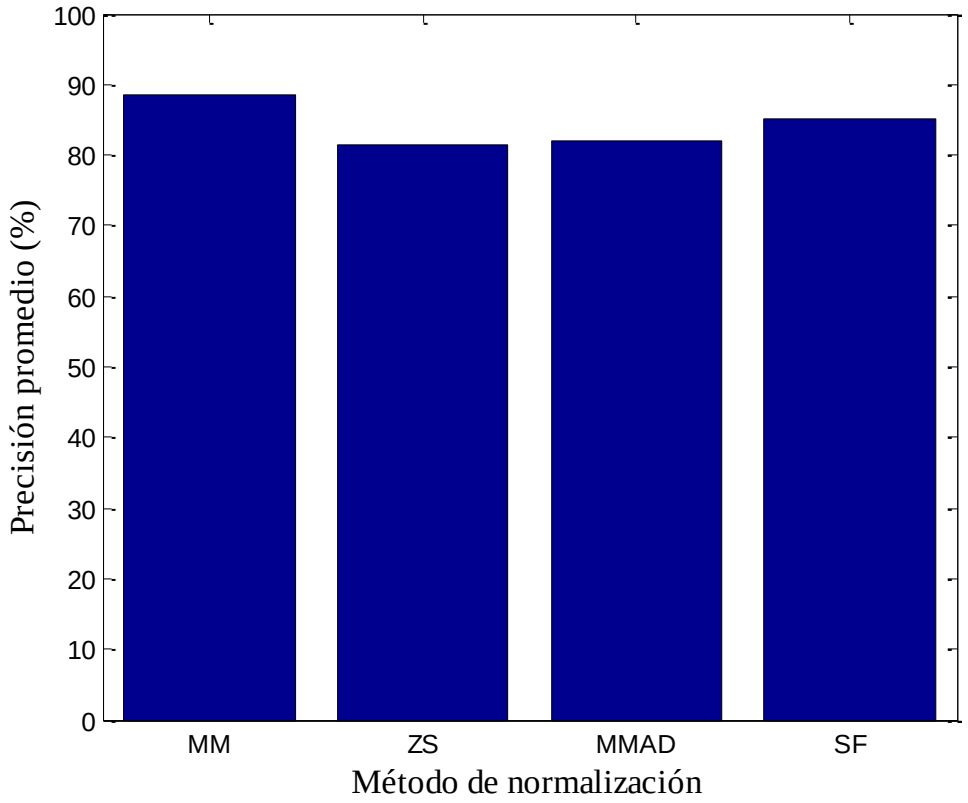


Figura 4.10 Precisión promedio de cada método de normalización.

Se observa en la figura 4.10 que el método de normalización MM presentó mejor desempeño que el resto de método utilizados con una precisión promedio de 88,13%. En la figura 4.11 se muestran los desempeños promedio para los kernels utilizados en la tesis solamente utilizando el método de normalización MM.

Se puede notar en la figura 4.11 que los kernels polinomial y RBF tienen un buen desempeño en comparación con los otros dos, sin embargo, el kernel RBF presentó un mejor desempeño con una precisión promedio de 89,24%. Como se observó en la figura 4.3, los parámetros correspondientes a combinación de atributos y al esquema de descomposición no influyen mucho en el desempeño de las SVM, debido a esto para seleccionar que parámetro utilizar se observó el tiempo computacional de cada uno. En las figuras 4.12, 4.13 y 4.14 muestran el tiempo computacional promedio en segundos para los atributos, descomposición y tipo de carga, respectivamente.

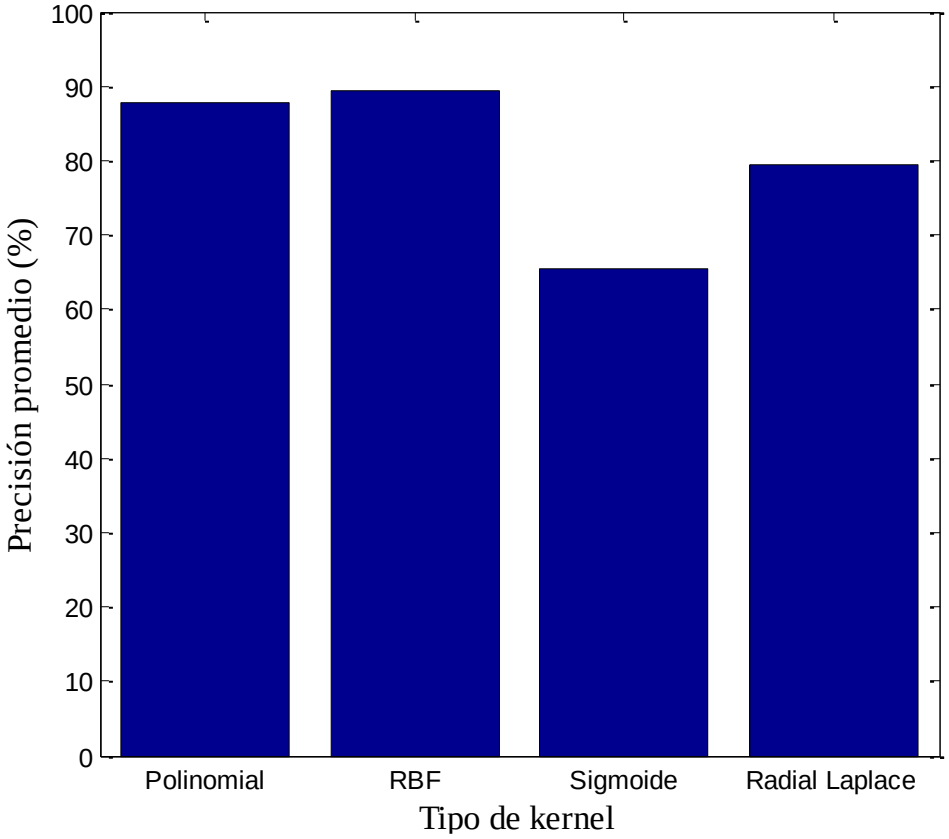


Figura 4.11 Precisión promedio de los kernels con normalización MM

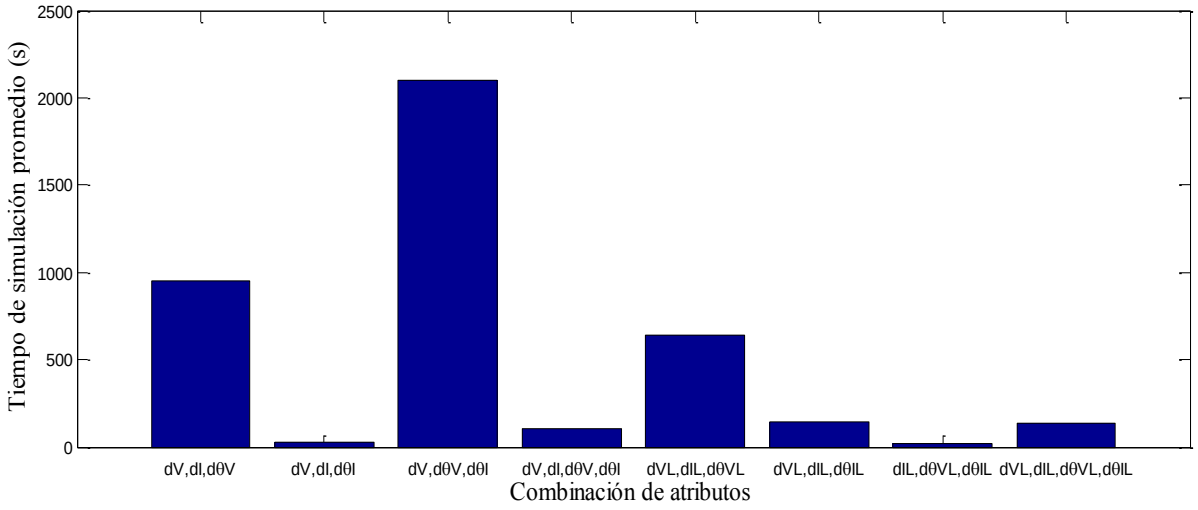


Figura 4.12 Tiempo computacional promedio de las combinaciones de atributos

Se observa en la figura 4.12, un comportamiento igual que para el circuito IEEE34, siendo la combinación de atributos *dIL ,dθVL, dθIL* la que tuvo menor costo computacional, con un tiempo promedio de 21,02 segundos.

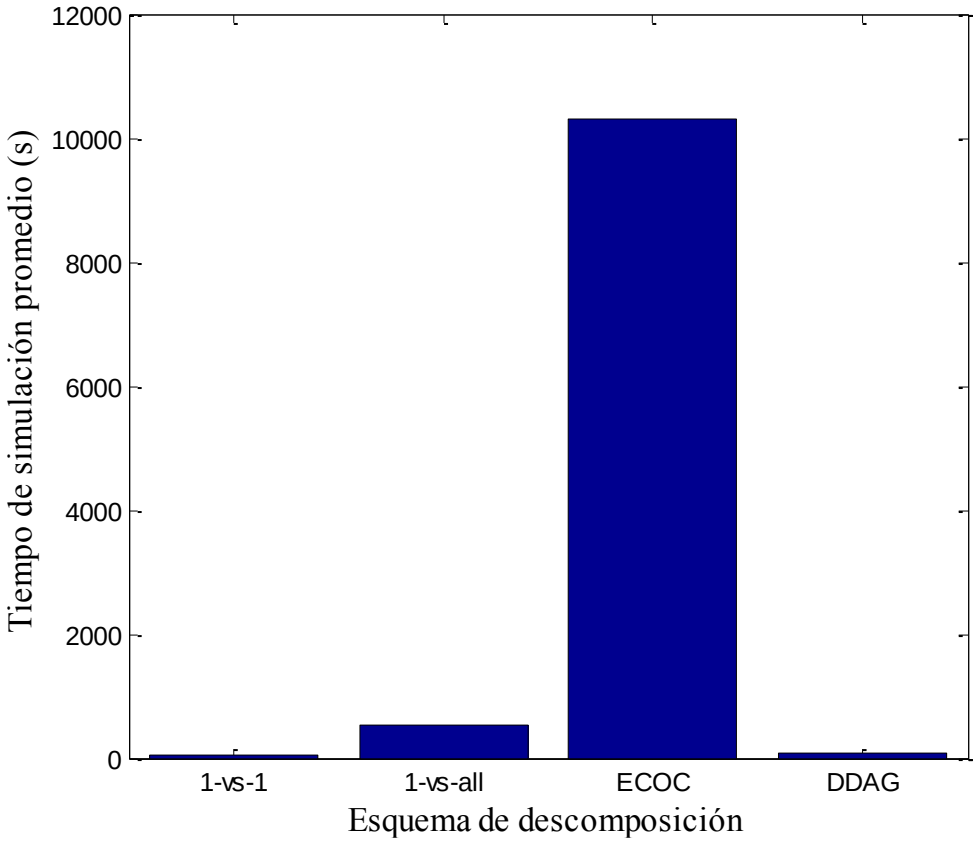


Figura 4.13 Tiempo computacional promedio del esquema descomposición

En la figura 4.13 se observa nuevamente que la descomposición ECOC es la que tiene mayor costo computacional. Además, se puede notar en la figura 4.13 que la descomposición 1-vs-1 es la que tiene menor costo computacional comparado con los otros esquemas utilizados en esta tesis con un costo promedio de 59,89 segundos.

En la figura 4.14 se observa que el tipo de carga impedancia constante es la que tuvo menor costo computacional, con un tiempo promedio de 689,11 segundos.

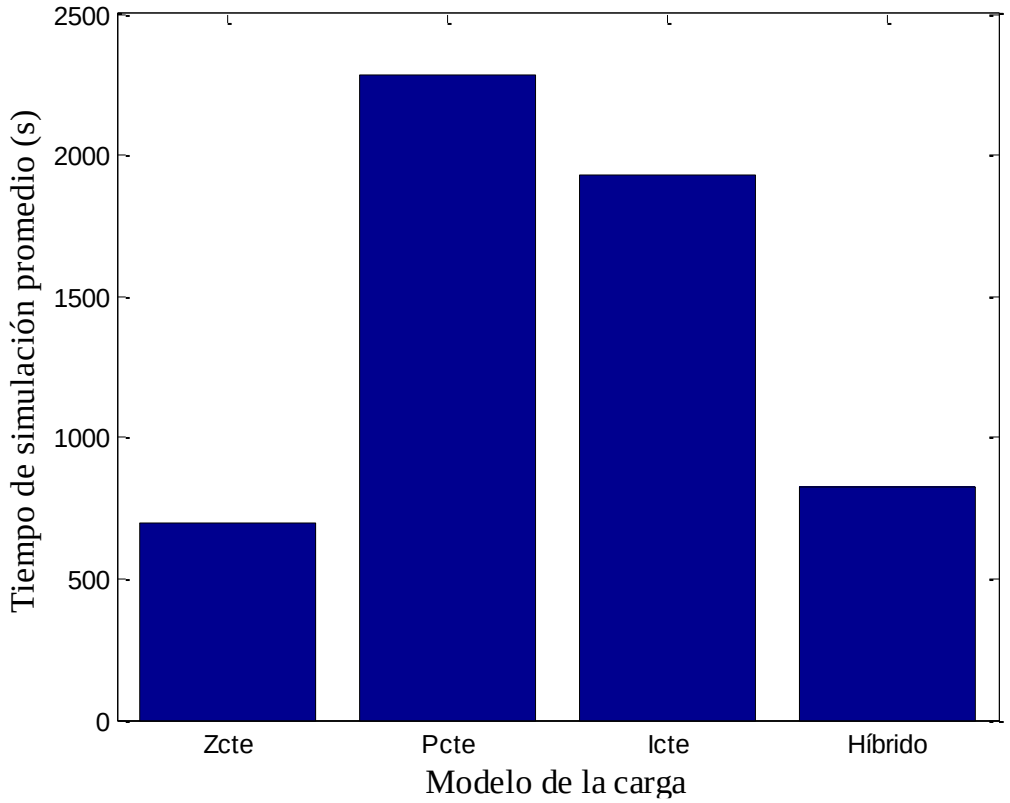


Figura 4.14 Tiempo computacional promedio del tipo carga

Después de tener los parámetros del localizador se probó el desempeño para todos los tipos de fallas, primero parametrizando como se muestra en la sección 4.3.1. Obteniendo una base de datos completa para entrenamiento de 72600 casos considerando resistencias de falla de 0,05 a 40Ω en pasos de 4Ω. La base de datos de prueba contiene 198000 casos con resistencia de falla entre 1 a 39 Ω en pasos de 1 Ω, sin incluir los casos utilizados en la parametrización. Los resultados del desempeño del localizador se muestran en la tabla 4.2.

Tipo de falla	No. datos		Precisión (%)
	Entrenamiento	Prueba	
Monofásica a-g	7260	19800	99,49
Monofásica b-g	7260	19800	99,11
Monofásica c-g	7260	19800	98,79
Bifásica a-b	7260	19800	99,91
Bifásica b-c	7260	19800	99,67
Bifásica c-a	7260	19800	100
Bifásica a tierra a-b-g	7260	19800	98,94
Bifásica a tierra b-c-g	7260	19800	99,39
Bifásica a tierra c-a-g	7260	19800	95,73
Trifásica a-b-c	7260	19800	99,51

Tabla 4.2 Resultados de validación del circuito 44 nodos con diferentes condiciones de operación.

Se puede notar en la tabla 4.2 que el clasificador tiene un gran desempeño para todos los tipos de fallas con precisiones mayores al 95,73%. Adicionalmente, para todos los tipos de fallas se realizó una serie de entrenamientos, en los cuales se modificó el número de subconjuntos en el proceso de validación cruzada. La idea principal es que la base de entrenamiento sea grande y la de validación pequeña, lo cual ocurre en la etapa de aplicación para un registro real (*online*). Para cada subconjunto de validación cruzada se halla la precisión según lo propuesto en la ecuación (3.3). Posteriormente se obtiene el promedio de los subconjuntos.

En la figura 4.15 y 4.16 se muestran las precisiones promedios y tiempos de simulación para cada número de subconjuntos, respectivamente. Sólo se mostró el peor caso en precisión que fue para la falla bifásica a tierra c-a-g.

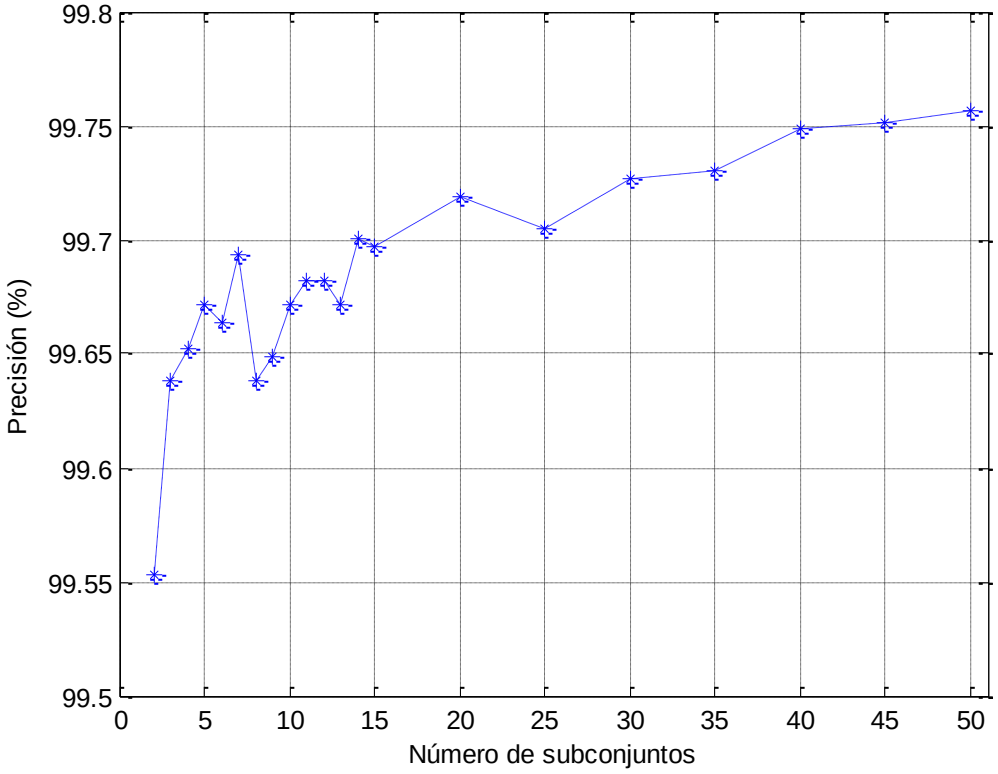


Figura 4.15 Precisión promedio para cada validación cruzada

Se puede notar en la figura 4.15 que a medida que se aumenta el número de subconjuntos la precisión promedio tiende a mejorar, esto es debido a que entre mayor sea el número de subconjuntos, mayor es la base de datos de entrenamiento en cada proceso. Por ejemplo, si se utiliza dos subconjuntos, la base de entrenamiento para cada proceso es el 50% del total de los datos, mientras si se utiliza 10 subconjuntos, la base de entrenamiento en cada proceso es el 90% del total de datos, y este porcentaje aumenta a medida que se considere un mayor número de subconjuntos. Es importante notar que el peor de los escenarios en la validación cruzada (99,56%) mostró mejor desempeño en comparación con la falla bifásica a tierra c-a-g mostrada en la tabla 4.2.

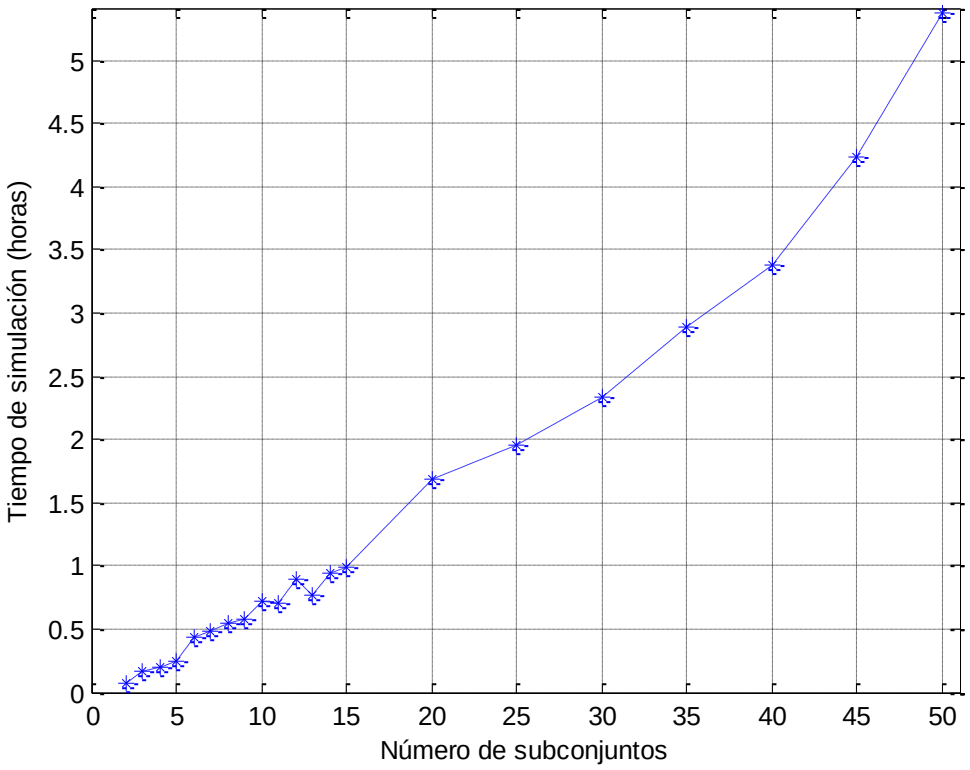


Figura 4.16 Tiempo de simulación en horas para cada validación cruzada

En la figura 4.16 se observa que a medida que se aumenta el número de subconjuntos en la validación cruzada, aumenta el tiempo de simulación, esto es debido a dos sucesos. Primero, que el número de entrenamiento que se hacen va aumentando con el número de veces que en se divide el conjunto de datos. Segundo, a medida que se hagan más subconjuntos, la base de entrenamiento es más grande para cada subconjunto, debido que a la base datos de cada subconjunto es más pequeñas.

Capítulo 5.

Conclusiones y recomendaciones

5.1 Conclusiones generales

En los circuitos reales existen incertidumbres en los valores y la naturaleza de las cargas, lo que conlleva a que no se conozcan con precisión todos los parámetros del circuito en el momento de ocurrencia de una falla. Este hecho puede afectar el desempeño de los métodos de localización de fallas, y por tanto existe la necesidad de tener un localizador robusto que garantice más confiabilidad en la estimación de la zona en falla.

Para implementar un localizador robusto basado en las SVM es necesario determinar qué parámetros afectan más el desempeño en este problema. La determinación de los parámetros que tienen mayor influencia se logró a partir de un análisis de sensibilidad, el cual es una metodología útil para determinar que sensibilidad tiene algunos parámetros sobre un problema, y que en este caso se adaptó al problema de localización de fallas. Se observó que aspectos como la normalización y el tipo de kernel mostraron tener mayor influencia en el desempeño del clasificador.

El análisis de sensibilidad requiere de la generación de evaluaciones para las pruebas, por lo que en esta tesis se empleó *latin hypercube*. El *latin hypercube* es una herramienta muy útil, debido a que no depende del modelo y se pueden utilizar variables cuantitativas como cualitativas sin tener mayores dificultades. Adicionalmente, el *latin hypercube* tiene otra característica que permite cubrir el espacio de incertidumbre con menos evaluaciones, lo cual es una ventaja en este problema debido a que cada evaluación del modelo tiene un costo computacional alto.

La aplicación de estas técnicas al problema de localización de fallas permite refinar el desempeño de los métodos empleados, lo que permite la ubicación óptima y rápida de las fallas en sistemas de distribución, para disminuir el tiempo de interrupción y restauración del servicio, y así mejorar los índices de calidad establecidos por la CREG y por lo tanto cumplir con las normas impuestas. Estos índices son IRAD (índice de referencia agrupado de la discontinuidad) e ITAD (índice trimestral agrupado de la discontinuidad).

5.2 Conclusiones asociadas a los parámetros de las SVM

El análisis de sensibilidad cuenta con una etapa dónde necesita una técnica que permite mostrar qué parámetros tienen mayor relevancia en el problema a tratar. La técnica de sensibilidad utilizada en esta tesis fue la de análisis de regresión, debido a que es una de las más utilizadas, y además por su fácil implementación.

El análisis de regresión que mostró que el parámetro con mayor influencia en el desempeño del localizador de fallas fue la normalización. Esto muestra que los métodos basados en el

conocimiento se ven altamente afectados por el procesamiento de los datos, lo cual influye directamente en su rendimiento tanto en desempeño y costo computacional. Las normalizaciones Min-max y Función sigmoidal fueron las que presentaron mejor desempeño para los circuitos analizados en esta tesis.

Adicionalmente, el kernel es otro parámetro que mostró tener influencia en el desempeño de las SVM. A partir de los resultados mostrados en esta tesis, se observa que el kernel RBF presentó siempre un mejor comportamiento que los otros kernel utilizados. Los demás parámetros considerados no presentaron mayor efecto en el rendimiento, sin embargo, mostraron tener gran influencia en costo computacional.

A partir del análisis de regresión, se puede apreciar que todos los esquemas de descomposición/reconstrucción presentan un buen desempeño. Sin embargo, el esquema ECOC no es eficiente computacionalmente como se muestran en los resultados, debido a que requiere muchos más biclasificadores en comparación con los otros esquemas y además, utiliza todos los datos de entrenamiento para cada biclasificador. Por tal motivo, se recomienda utilizar el esquema uno contra uno.

En los métodos basados en el conocimiento, la selección de atributos es un paso muy importante, debido a que éstos deben contener características más significativas del problema a trabajar, con el fin de mejorar el desempeño predictivo del clasificador para obtener el máximo rendimiento con el mínimo esfuerzo. Para todas las combinaciones de atributos consideradas en esta investigación se obtuvo buen rendimiento, mostrando que trabajar con atributos que contengan variaciones de tensión y corriente tiene una representación significativa de los circuitos.

5.3 Conclusiones asociadas al localizador basado en las SVM

Las SVM son una buena herramienta en el problema de localización de fallas en sistemas de distribución, ya que solucionan el problema de múltiple estimación, propio de los métodos basados en el modelo de los sistemas de distribución de energía eléctrica. Los resultados de la localización de la falla corresponden a la zona del circuito donde pudo haber ocurrido el evento, y al realizar la combinación de esta respuesta con la distancia de falla obtenida con los métodos basados en el modelo, se puede tener un lugar específico del circuito donde ocurrió la falla.

Los resultados obtenidos prueban la validez y eficiencia del localizador basado en las SVM para la solución del problema de localización. Adicionalmente, en la mayoría de los casos bajo estudio se utilizó una base de prueba 2,7 veces mayor que la base de entrenamiento, lo cual muestra la capacidad de generalización de las SVM.

Finalmente, es importante notar que una precisión del 100% del localizador ante una condición específica del sistema de distribución, no garantiza que el método no pueda tener clasificaciones erróneas ante otras condiciones de operación. Para tener certeza de que el algoritmo tiene un alto desempeño, es necesario evaluar el clasificador ante todas las posibles variaciones que se puedan presentar en un circuito real, como se propuso en esta investigación para variaciones de resistencia de falla, de la carga, de la tensión en la fuente

y la longitud de línea. Sin embargo, las SVM tienen un buen coeficiente de generalización, lo que implica que provee buenos resultados ante datos desconocidos, el cual la hace útil en sistemas de distribución reales.

5.4 Recomendaciones

Unas de las etapas más importantes y con mayor costo computacional es la parametrización de las SVM, la cual también afecta directamente el desempeño del localizador. En esta etapa se recomienda trabajar con pocos datos, debido a que el tiempo requerido aumenta considerablemente cuando se aumenta el número de datos. Adicionalmente, el conjunto de datos debe tener una buena representación del circuito bajo análisis, es decir, utilizar inicialmente sólo los datos obtenidos a carga nominal.

Como se mostró en los resultados, el esquema de descomposición/reconstrucción ECOC tiene un alto costo computacional, y a medida que los circuitos contengas más zonas su costo aumentará significativamente. Por esto, no se recomienda trabajar con la descomposición ECOC. Sin embargo, si desea trabajar con ella se recomienda utilizar variaciones de la descomposición\reconstrucción ECOC, donde varían la distancia de Hamming como la doble capa entre otros.

Finalmente, como se observó en los resultados, el FDLA no presentó los resultados satisfactorios. Se recomienda seguir trabajando en este temas y utilizar un FDLA multi clase para así considerar todo y que los datos no pierdan dependencia.

5.5 Trabajos futuros

Ya que los sistemas de distribución son una parte importante del sistema de potencia y están directamente relacionados con el usuario final, el cual requiere altos niveles en la calidad de la energía, es importante seguir investigando en localizadores más robustos, los cuales permitan mejorar los índices de continuidad de suministro del sistema. Por lo tanto, como trabajo futuro se propone implementar esta metodología en otros métodos basados en clasificación, con el fin de disminuir los efectos de los parámetros con mayor influencia, lo que finalmente conlleva a localizadores más robustos con estimaciones más confiables ante una falla. De esta forma, diferentes metodologías pueden trabajar de manera conjunta, de tal manera que ante una misma falla, la localización sea mucho más confiable.

Adicionalmente, para tratar de representar con mucha más fiabilidad los circuitos reales se deben simular muchas más condiciones de operación del mismos, los cuales, deben tener en cuenta variaciones como la carga tanto en magnitud como factor de potencia, variaciones de la resistencia de fallas, de la magnitud de la tensión y desbalance de la misma, también variaciones de la longitud de las líneas, de la frecuencia del sistema y de la resistividad del terreno.

También, se propone trabajar los circuitos con diferentes modelos de carga al mismo tiempo, siendo que las cargas son unas de las mayores incertidumbres en la operación del circuito. En este caso es necesario refinar su modelado para tratar de representar cargas tipo en los circuitos de distribución, tales como; residencial, comercial e industrial, ya que cada

una de las cargas tipo tienen diferentes comportamientos ante diferentes escenarios de fallas.

Bibliografía

[ALZA,2013] N. Alzate. "Influencia de la variación de parámetros del sistema de potencia en la localización de fallas con métodos de clasificación". Tesis de pregrado. Universidad Tecnológica de Pereira. Pereira, Colombia. 2013.

[ANIL,2005] J. Anil, N. Karthik, R. Arun, "Score normalization in multimodal biometric systems", Pattern Recognition, vol. 38, no.12, pp. 2270–2285, Dec. 2005.

[BENN,1992] K. Bennett, O. Mangasarian. "Robust linear programming discrimination of two linearly separable set" Optimizations methods and Software. pp 23-24. 1992.

[BOSE,1992] B. Boser, I. Guyon, V. Vapnik. "A training algorithm for optimal margin classifier" Proc V ACM Workshop on Computational Learning Theory. pp 144-152. 1992.

[BURG,1998] C. Burges. "A tutorial on support vector machines for pattern recognition". Data Mining and Knoledge Discovery, 1998.

[CORT, 1995] C. Cortes, V. Vapnik. "Support vector networks". Machine learning. pp 273-297. 1995.

[CREG,2008] Comisión de Regulación de Energía y Gas. "Resolución CREG No. 097 de 2008". Ministerio de minas y energía. 2008. pp 1-135.

[CRIS,2011] M. Cristaldi. "Diseño de Experimentos Dinámicos basado en Modelos para Optimización". Tesis Doctoral. Universidad Nacional del Litoral. Santa Fe, Argentina. 2011.

[DAGE,2000] Dagenhart J. "The 40-ohms ground-fault phenomenon". IEEE Transactions on Industry Applications. Vol. 36, no. 1. pp 30-32. 2000.

[FANG,2006] K. Fang, L. Runze. Design and modeling for computer experiments. Chapman and Hall/CRC. United States. 2006. pp. 1-71.

[FARR,2005] M. Farrús, J. AnGUTIa, J. Hernando, et al., "Fusión de sistemas de reconocimiento basados en características de alto y bajo nivel", In Actas del III Congreso de la Sociedad Española de Acústica Forense, Santiago de Compostela, Spain, Oct. 2005.

[GALL,2008] Gallego R, Escobar A, Toro E. "Técnicas Metaheurísticas de Optimización". Ed 2. Colombia. 2008.

[GARC,2005] A. Garcés, J. Galvis, A. Escobar. "Aplicación del Algoritmo de Búsqueda Tabú al Problema de Despacho Hidrotérmico". Scientia et Technica Año XI, No 29. Universidad Tecnológica de Pereira. Colombia. 2005. ISSN 0122-1701.

[GAYA,2010] K. Gayathri, N. Kumarappan "Accurate fault location on EHV lines using both RBF based support vector machine and SCALCG based neural network". Expert Systems with Applications 37 (2010) pp 8822–8830. Annamalai University. India 2010.

[GUTI,2010] J. Gutiérrez, J Mora, S. Pérez. "Strategy based on genetic algorithms for an optimal adjust of a support vector machine used for locating faults in power distribution systems". Revista Ingeniería UDEA, No.53, pp.174-184. Colombia. 2010

[GIL,2013] W. Gil, J. Mora, S. Pérez, "Análisis comparativo de metaheurísticas para calibración de localizadores de fallas en sistemas de distribución", *Ingeniería y Competitividad*, vol. 15, no. 1, pp. 103-115, 2013.

[GLOV,2002]. F. Glover, G. Kochenberger, Handbook of Metaheuristics. 1a. ed. EEUU: Kluwer Academic Publishers, 2002.

[HAN,2006] J. Han, M. Kamber, Data Mining: Concepts and Techniques. 2a. ed. New York, Morgan Kaufmann Publishers, 2006.

[HERR,2013] A. Herrera "Análisis de los efectos de la variación de los parámetros del modelo de línea, de carga y de fuente, en la localización de fallas en sistemas de distribución". Tesis de Maestría. Universidad Tecnológica de Pereira. Pereira. 2013.

[KIJS,2002] B. Kijsirikul, N. Ussivakul, S. Meknavin. "Adaptive Directed Acyclic Graphs for Multiclass Classification" Springer, vol. 2417, pp. 158–168, 2002.

[KOTS,2006] S. Kotsiantis, D. Kanellopoulos, P. Pintelas, "Data Preprocessing for Supervised Leaning", International Journal of Computer Science, vol. 1, no. 2, pp. 111-117, 2006.

[KUHN,1951] H. Kuhn, A Tucker. "Nonlinear programming". Proceedings II Berkeley Symposium on Mathematical Statistics and probabilistics. University of California Press. pp 481-492. United States. 1951.

[MARI,2013] J. Marín. "Análisis del efecto de la variación de parámetros de modelo de un sistema de distribución sobre las metodologías de localización de fallas paralelas". Tesis de Maestría. Universidad Tecnológica de Pereira. Colombia. 2013.

[MIAN,2010] F. Mianji, Y. Zhang. "*Improved hyperspectral land-cover analysis using Relevance vector machine*". IEEE International Conference on Image Processing (ICIP) 17th, pp.2281-2284, Sept. 2010.

[MORA,2005] G. Morales, A. Gómez. "Estudio e Implementación de una herramienta basada en Máquinas de soporte vectorial aplicada a la Localización de fallas en sistemas de distribución". Trabajo de Grado. Universidad Industrial de Santander. Colombia. 2005.

[MORA,2006] J. Mora. "Localización de Faltas en Sistemas de Distribución de Energía Eléctrica usando Métodos basados en el Modelo y Métodos de Clasificación Basados en el Conocimiento". Tesis Doctoral. Universidad de Girona. España. 2006. ISBN 978-84-690-4513-8

[MORA,2008] J. Mora, G. Morales, S. Pérez. "Classification methodology and feature selection to assist fault location in power distribution system". Revista Ingeniería UDEA, No. 44. pp. 83-96. Colombia. 2008

[PERE,2010] L. Pérez, J. Mora, S. Pérez, "Diseño de una herramienta eficiente de simulación automática de fallas en sistemas eléctricos de potencia", Dyna, vol. 77, no. 164, pp. 178-188, 2010.

[SALA,2012] D. Salazar. "Comparación de Máquinas de Soporte Vectorial vs. Regresión Logística. ¿Cuál es más recomendable para discriminar?". Tesis de Maestría. Universidad Nacional de Colombia, Sede Medellín. Colombia. 2012

[SALT,2000] A. Saltelli, K. Chan. Sensitivity Analysis. John Wiley & Sons Ltd. United Kingdom. 2000. pp. 101-152.

[SHAL,2006] L. Al Shalabi, Z. Shaaban, "Normalization as a Preprocessing Engine for Data Mining and the Approach of Preference Matrix", International Conference on Dependability of Computer Systems, pp. 207-214, May. 2006.

[SNEL,2005] R. Snelick, U. Uludag, A. Mink, et al., "Large-scale evaluation of multimodal biometric authentication using state-of-the-art systems," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.27, no.3, pp. 450-455, Mar. 2005.

[SOLA,1997] J. Sola, J. Sevilla, "Importance of Input Data Normalization for the Application of Neural Networks to Complex Industrial Problems", IEEE Transactions on nuclear science, vol. 44, no. 3, pp. 1464-1468, Jun. 1997.

[TORO,2008] E. Toro, A. Rueda, H Ruiz. "Efecto De La Configuración Inicial en la Solución del Problema de Corte Bidimensional Usando el Algoritmo Búsqueda Tabú". Revista Colombiana de Tecnologías avanzadas.

Volumen 1, Número 11. Colombia. 2008. ISSN 1692-725.

[VIAN,2009] F. Viana, G. Venter. "An algorithm for fast optimal *Latin hypercube* design of experiments'' International journal for numerical methods in engineering. United States. 2009.

[ZAPA,2013] A. Zapata. "Implementación y comparación de técnicas de localización de fallas en sistemas de distribución basadas en minería de datos". Tesis de Maestría. Universidad Tecnológica de Pereira. Pereira. 2013.

[ZHU,1997] J. Zhu, D. Lubkeman, A. Girgis. "Automated fault location and diagnosis on electric power distribution feeders". IEEE Transactions on Power Delivery 1997, pp. 801-809

Anexo A. Manual de usuario de la herramienta

A.1 Instalación de la herramienta

Para la instalación es necesario agregar al *sethpath* del software de simulación Matlab la carpeta que contiene el software de localización de fallas SVMGUI. Este proceso se realiza en tres pasos.

a. Ejecutar Matlab y dar clic en el menú File. De la lista de opciones se da clic en '*Set path…*' como se muestra en la figura A.1.

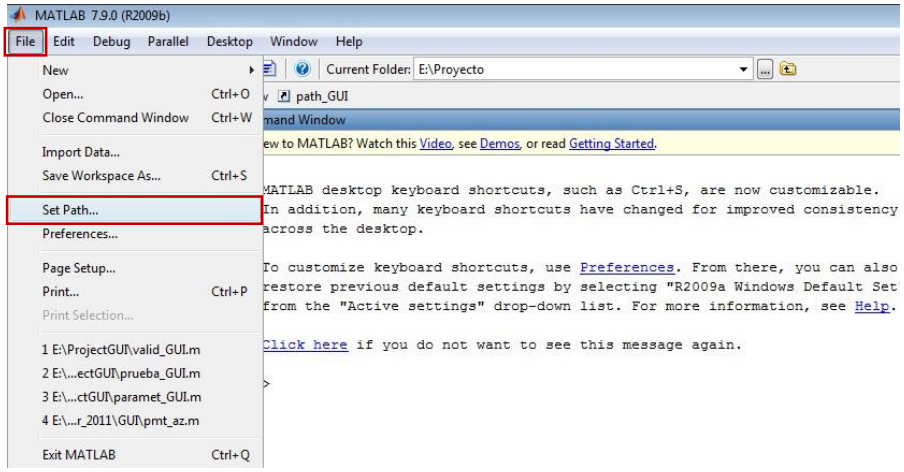


Figura A.1 Abrir el *Set path* de Matlab.

b. Dar clic en '*Add with subfolders…*' y seleccionar la carpeta que contiene el software de localización de fallas SVM_GUI, como se muestra en A.2. clic en 'Aceptar' y se guarda el *path*.

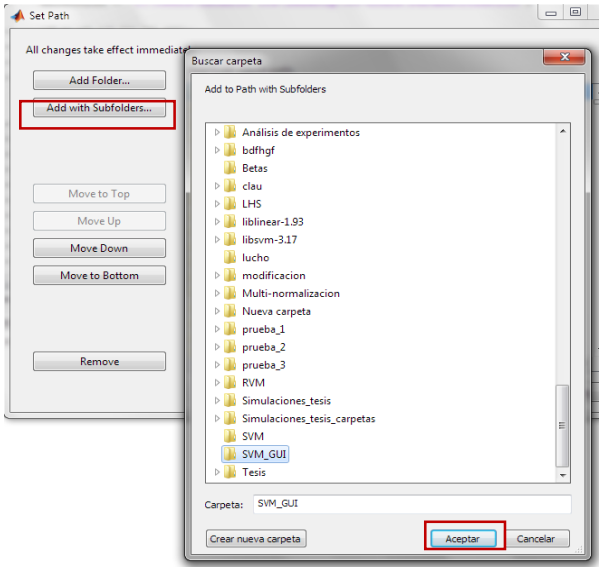


Figura A.2 Seleccionar la carpeta con la herramienta.

c. Para ejecutar el software se digita en el *CommandWindow* de Matlab la función MBC_SVM, tal como se muestra en la figura A.3.

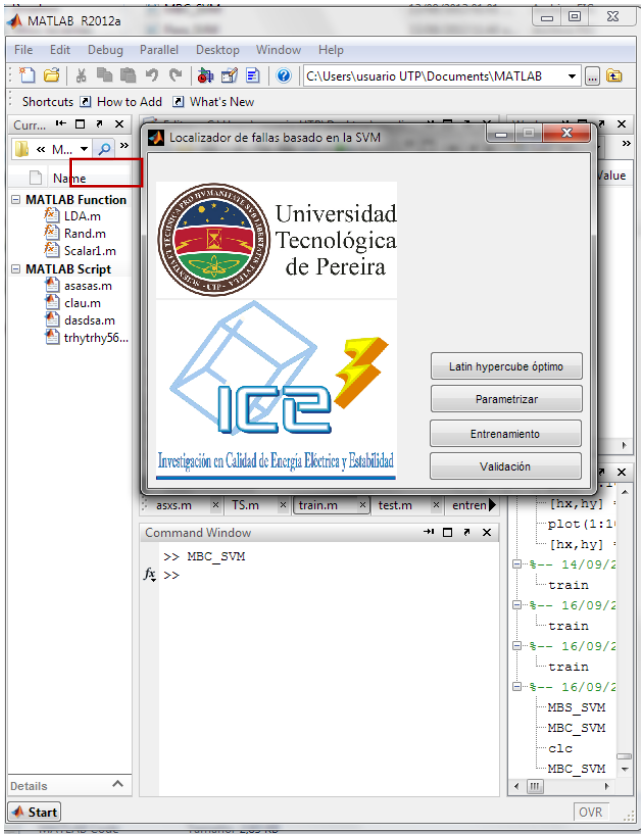


Figura A.3 Ejecución del método de localización en Matlab.

A.2 Modo de selección

En su interfaz principal la herramienta cuenta con cuatro opciones que le permiten ingresar a cada uno de las fases de implementación del análisis de sensibilidad. Se encuentra un botón para ingresar al *latin hypercube* óptimo, uno para la parametrización, otro para el entrenamiento y uno para la validación. Como se muestra en la figura A.3.

A.3 Modo de *latin hypercube* óptimo

En este modo se permite el crear el escenario de *n* puntos para hacer el análisis de sensibilidad. El modulo tiene dos botones, uno se denomina "Evaluar" y sirve para ejecutar el latin hypercube y el otro se denomina "Regresar" y sirve para regresar a la interfaz principal. En la figura A.4 se presenta la interfaz de *latin hypercube* óptimo (LHS).
Primero, en la parte "número de puntos" elije el número de escenarios que desea crear. Luego, Si se desea crear los escenarios necesita ingresar la dirección donde desea guardar los escenarios creados por el LHS y luego da clic en "Evaluar".

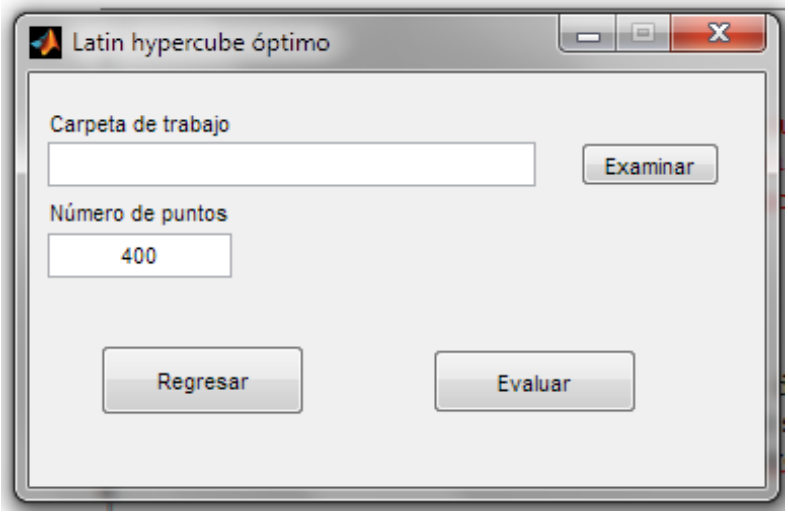


Figura A.4 Modo de *latin hypercube* óptimo.

A.4 Modo de parametrizar

En el modo de parametrizar se tiene igualmente un panel donde se ingresa información general tal como el escenario creado por LHS, la carpeta donde se guardarán los resultados, la dirección donde se encuentran las bases de datos de parametrización y el archivo de zonificación. En la figura A.5 se muestra la interfaz para el modo de parametrizar.

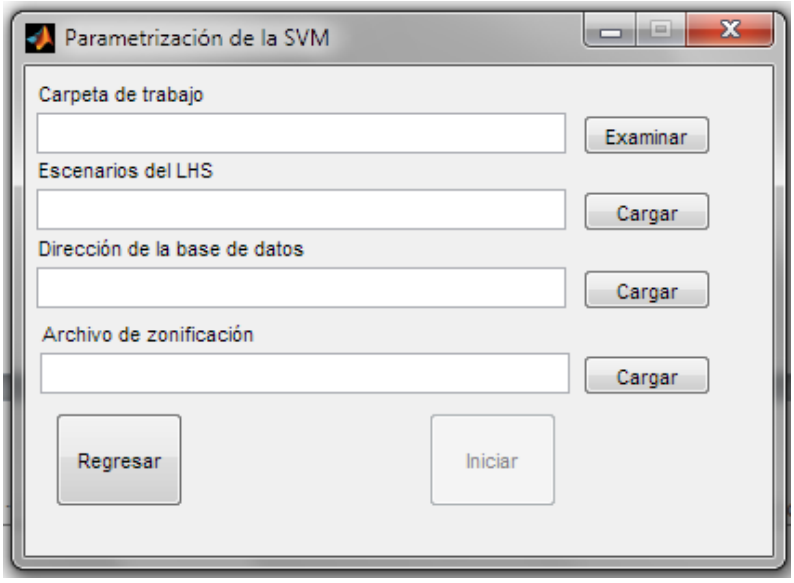


Figura A.5 Modo de parametrización.

La herramienta realiza la parametrización de todos los puntos del LHS. Los resultados de la parametrización se guardan en la carpeta de trabajo uno por cada punto del LHS. Al igual que el modo LHS tiene un botón “Regresar”, que tiene la misma función en toda la interfaz gráfica de regresa el menú principal y otro botón “Iniciar”, que tiene la misma función en toda la interfaz gráfica de ejecutar el programa donde esté trabajando. Después de cargar todo el botón “Iniciar” se habilita.

A.5 Modo de entrenamiento

En el modo entrenamiento se tiene igualmente un panel donde se ingresa información general tal como la carpeta donde se guardarán los resultados, dirección donde se encuentran los archivos generados del modo parametrizar y dirección donde se encuentran las bases de datos de entrenamiento. Después de ingresar toda la información se habilita el botón "Iniciar". Como se presenta en la figura A.6.

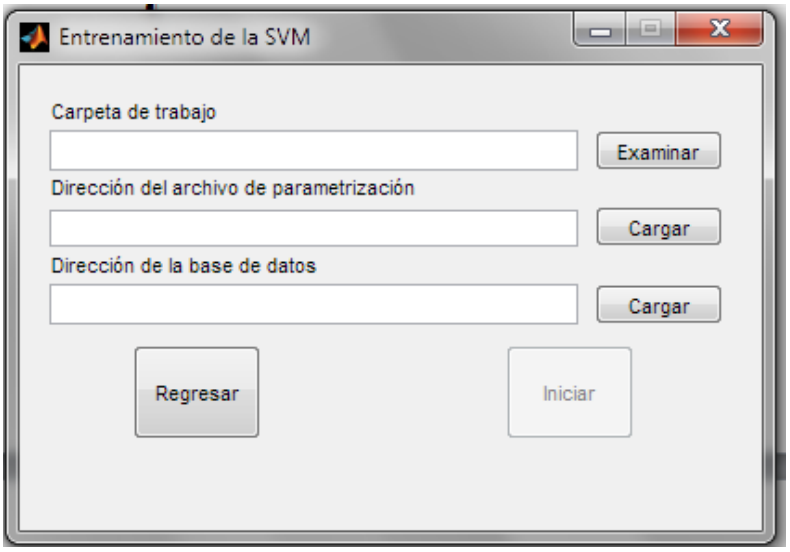


Figura A.6 Modo de entrenamiento.

A.6 Modo de validación

En el modo validación tiene una primera parte donde se evalúan los escenarios, tiene igualmente un panel donde se ingresa información general tal como la carpeta donde se guardarán los resultados, dirección donde se encuentran los archivos generados del modo entrenamiento, la dirección donde se encuentran las bases de datos de entrenamiento. Adicionalmente, tiene otra sección donde hallan los parámetros que más influyen en el problema, y necesita la dirección donde se encuentran los escenarios evaluados por este modo. Después de ingresar toda la información se habilita el botón "Iniciar". Como se presenta en la figura A.7.

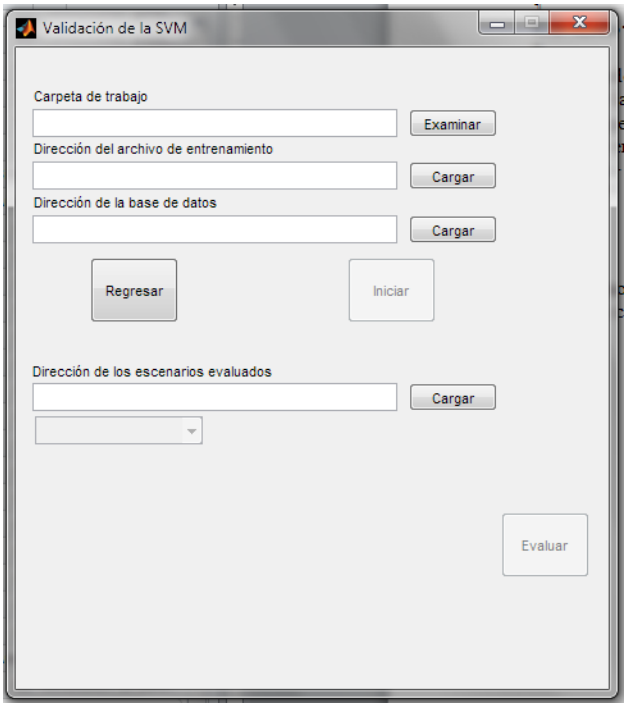


Figura A.7 Modo de validación.

En este modo se evalúan los escenarios propuestos por el LHS y luego se hace el análisis de regresión como se muestra en la figura A.8

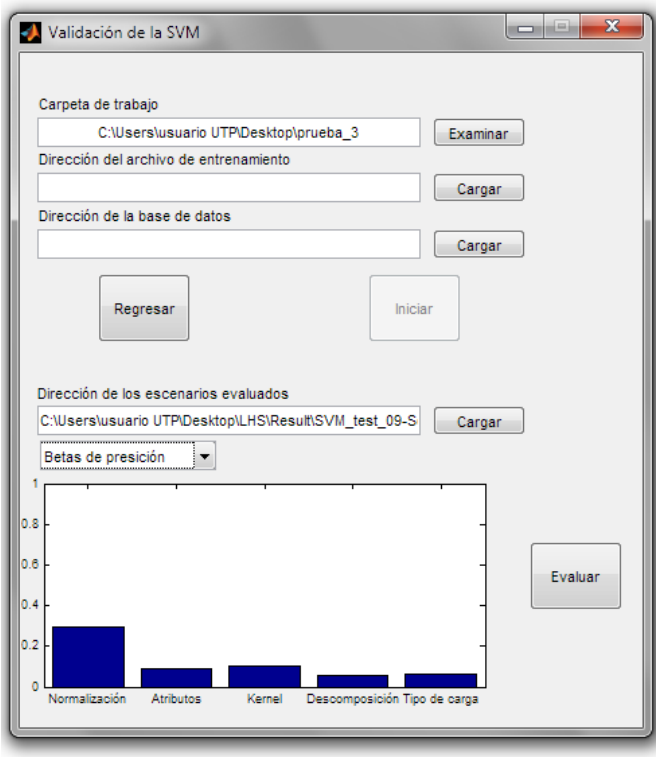


Figura A.8 Modo de validación.

A.7 Requerimientos de la herramienta

a. Requerimientos de software

Se requiere tener instalado Matlab 7 o superior para el correcto funcionamiento de la herramienta.

b. Requerimientos de hardware

Procesador: mayor a 1.0 GHz
Memoria instalada (RAM): mayor a 1 GB
Disco duro: Mayor a 20 GB

Anexo B. Parametrización de las SVM

B1. Resultados de parametrización de las SVM en el circuito IEEE 34 nodos

A continuación se muestran los resultados de la parametrización para el circuito IEEE 34 nodos considerando una normalización MM, los atributos dIL,dθVL,dθIL, el kernel RBF, la descomposición 1-v-1 y el modelo de la carga Zcte.

Tipo de falla	*C*	γ	Error de validación cruzada (%)
Monofásica a-g	415710,52	0,94	0,00
Monofásica b-g	2112926,87	0,65	0,64
Monofásica c-g	59432,16	0,04	0,00
Bifásica a-b	4925486,55	0,06	0,00
Bifásica b-c	293939,22	0,06	0,00
Bifásica c-a	9999325,53	0,53	0,00
Bifásica a tierra a-b-g	1016330,41	0,08	0,00
Bifásica a tierra b-c-g	484577,60	0,56	0,00
Bifásica a tierra c-a-g	259052,79	0,85	0,00
Trifásica a-b-c	174197,38	0,43	0,00

Tabla B.1 Parametrización con datos a condición nominal del circuito IEEE34.

B2. Resultados de parametrización de las SVM en el circuito en el circuito 44 nodos

A continuación se muestran los resultados de la parametrización para el circuito 44 nodos considerando una normalización SF, los atributos dIL,dθVL,dθIL, el kernel RBF, la descomposición 1-v-1 y el modelo de la carga Zcte.

Tipo de falla	*C*	γ	Error de validación cruzada (%)
Monofásica a-g	169649,74	0,21	12,12
Monofásica b-g	1204151,46	2,33	15,91
Monofásica c-g	55526,85	1,56	8,71
Bifásica a-b	30104,13	0,43	2,27
Bifásica b-c	650425,93	0,09	0,76
Bifásica c-a	861728,34	0,50	1,89
Bifásica a tierra a-b-g	97808,64	0,31	9,85
Bifásica a tierra b-c-g	2754359,44	3,22	10,23
Bifásica a tierra c-a-g	319116,85	0,69	10,61
Trifásica a-b-c	460705,11	1,65	13,64

Tabla B.2 Parametrización con datos a condición nominal del circuito 44 nodos.

Anexo C. Pseudocódigo de los algoritmos implementados

C1. Parametrización *SVM*

Programa: TS
Entorno: dato_param,direccion, son variables de texto
zonificacion,data,tf,rf son variables enteras

Algoritmo:
Leer train, zonas, ns, kmax, direccion, nombsist
Obtener los datos de parametrización de la carpeta
Obtener la zonificación del archivo zonas, hoja nodos y zona
Obtener los nodos de parametrización y prueba en el archivo zonas

Asignar a cada dato de entrenamiento la zona correspondiente y normalizar cada dato (manejo_datos)

Para el tipo de falla de entrada

Para la combinación de descriptores de entrada, que viene en data

Obtener los datos correspondientes al tipo de falla y los descriptores seleccionados

Elegir 20 puntos aleatorios de C y γ y evaluar en validación cruzada (Val_cru)
Guardar los errores de validación cruzada para cada punto
Escoger el error de validación cruzada mínimo y guárdalo como Incumbente

Para K = 1 hasta Kmax
Elegir el mínimo error de validación cruzada y escogerlo como punto de arranque
Evaluar la vecindad y evaluar en validación cruzada (Val_cru)
Si Error(k) es menor Incumbente
Guardar Error y parámetros
Incumbente es igual a Incumbente
Fin si
Fin para

Guardar la parametrización óptima, el error de validación cruzada.

Fin para
Fin para
Crear dirección para guardar los resultados
Crear un archivo '.mat' con los resultados de parametrización

Fin programa

C2. Manejo de datos

Programa: manejo_datos
Entorno: Datos, zonificación,rf,tf, son variables reales
Normaliazación es variable de texto

Algoritmo:
Elegir tipo de falla
Elegir nodos de parametrización

Elegir resistencias de fallas
Asignar a cada dato de entrenamiento la zona correspondiente
Normalizar cada dato

Fin programa

C3. Validación cruzada

Programa: val_crus
Entorno: C,γ, datos, son variables reales
Tipodekernel es variable de texto
Ns, Labels son variables enteras

Algoritmo:
Leer C,γ, datos
Para k=1 hasta ns

Elegir datos de entrenamiento y validar
Entrenar la SVM (descomp) con los datos de entrenamiento
Evaluar la SVM (reconst) datos de validar
Calcular error(k), como el número de datos mal clasificados sobre el número total de datos de validar

Fin para

Promediar error(k)

Fin programa

C4. Descomposición y entrenamiento de la SVM

Programa: descomp
Entorno: data_train,Labels, parametros son variables reales
Tipodescompos es variable de texto

Algoritmo:
Leer data_train,Labels, parametros
Escoger tipo de descomposición

Si Tipodescompos es igual a uno contra uno
    Descomponer en N(N-1)/2 clasificadores
    Para i = 1 hasta N(N-1)/2
    Entrenar classificador i
    Fin para
Si Tipodescompos es igual a uno contra todos
    Descomponer en Nclasificadores
    Para i = 1 hasta N
    Entrenar classificador i
    Fin para
Fin para Si Tipodescompos es igual a ECOC
    Descomponer en $2^{N-1}-1$ clasificadores
    Para i = 1 hasta $2^{N-1}-1$
    Entrenar classificador i
    Fin para
Fin si

Guardar biclasificadores de la SVM
Fin programa

C5. Reconstrucción y validación de la SVM

Programa: recons
Entorno: data_test,Labels, parámetros, entrenamiento son variables reales
Tiporecons es variable de texto

Algoritmo:
Leer data_test,Labels, parámetros, entrenamiento
Escoger tipo de descomposición

Si T Tiporecons es igual a uno contra uno
    Para k = 1 hasta número de datos_test
        Para i = 1 hasta N(N-1)/2
        Validar dato k classificador i
        Fin para
        Reconstrucción y dar etiqueta al dato k
    Fin Para
Si Tiporecons es igual a uno contra todos
    Para k = 1 hasta número de datos_test
        Para i = 1 hasta N
        Validar dato k classificador i
        Fin para
        Reconstrucción y dar etiqueta al dato k
    Fin Para
Si Tiporecons es igual a ECOC
    Para k = 1 hasta número de datos_test
        Para i = 1 hasta $2^{N-1}-1$
        Validar dato k classificador i
        Fin para
        Reconstrucción y dar etiqueta al dato k
    Fin Para
Si T Tiporecons es igual a uno contra uno con árbol de decisión
    ara k = 1 hasta número de datos_test
        Para i = 1 hasta N-1
        Validar dato k classificador i
        Fin para
        Reconstrucción y dar etiqueta al dato k
    Fin Para
Fin si
Calcular precisión y matriz de confución
Guardar Precisión y matriz
Fin programa

C6. Entrenamiento

Programa: Train_SVM
Entorno: data_train,Labels, parametros son variables reales
Tipodescompos es variable de texto
Tf es variable entera
Algoritmo:
Leer data_train, Labels, parámetros

Manejor de datos

Para todos los tipos de fallas
    Entrar al programa descomp
Fin para

Guardar cada entrenamiento

Fin programa

C7. Validación

Programa: Train_SVM
Entorno: data_test,Labels, entrenamiento son variables reales
Tiporescons es variable de texto
Tf es variable entera

Algoritmo:
Leer data_test Tf, entrenamiento

Manejor de datos

Para todos los tipos de fallas
    Entrar al programa recons
Fin para

Guardar cada validación

Fin programa

C8. Precisión

Programa: Precision
Entorno: data_test, entrenamiento son variables reales
Tiporescons es variable de texto

Algoritmo:
Leer data_test, entrenamiento, Tiporescons

Manejo de datos

Para todos los datos de prueba
    Clasificar el tipo de falla
    Clasificar el dato desconocido
Fin para

Fin programa

Anexo D. Pruebas adicionales al circuito IEEE34

D1. Descripción de la prueba y resultados

Aunque el localizador presentó muy buenos resultados, se hace necesario realizar más pruebas exhaustivas, para tratar de validar por completo la metodología. Esto se logra considerando más condiciones operativas del sistema de distribución y se implementó con la ayuda de la herramienta desarrollada por [ALZA,2013].

La herramienta puede crear *n* condiciones operativas para un circuito con la ayuda del *latin hypercube*, para explorar todo el espacio de incertidumbre, y así tratar de considerar cualquier posible condición del circuito. Adicionalmente, la herramienta permite hacer variaciones de los parámetros del circuito como: la carga promedio (magnitud potencia aparente), el factor de potencia, la longitud de la línea, la configuración de la línea, la resistividad del terreno, la magnitud de la tensión en la fuente, el ángulo de la tensión en la fuente y la frecuencia de la red.

a. Descripción de la prueba

Para tratar de considerar cualquier condición operativa del circuito IEEE 34 nodos se simularon 10000 escenarios operativos. Las variaciones que se hicieron para cada condición del circuito se muestran en la tabla D.1. Para algunos parámetros como la resistividad del terreno y el cambio de la configuración de la línea no se pudieron tener en cuenta en esta prueba, debido a que el circuito está modelado con líneas LCC.

Parámetros	**Rango de incertidumbre**	
	Mínimo	**Máximo**
Magnitud de tensión en la fuente	0,95 p.u	1,05 p.u
Ángulo de tensión en la fuente	-3,4°	3,4°
Carga promedio	10%	150%
Factor de potencia	-0,02	0,02
Longitud del conductor	95%	105%
Frecuencia de la red	59,8 Hz	60,2 Hz

Tabla D.1 Rangos de la variación de los parámetros del circuito

Las variaciones como la magnitud de la tensión en la fuente, el ángulo de la tensión en la fuente y la frecuencia de la red son variaciones globales, esto quiere decir sólo se realizan una vez por circuito. Variaciones como el factor de potencia y la longitud de la línea también se realizaron variaciones globales, es decir, para cada factor de potencia en la carga y cada segmento de línea se realizó el mismo porcentaje de variación. Como trabajo futuro se plantea hacer variaciones locales de a cada línea y factor de potencia de cada carga. Por otra parte, las variaciones de la carga que se realizaron fueron locales, lo cual indica que cada carga se varió individualmente en cada circuito.

b. Resultados de la prueba

Para cada uno de los 10000 circuitos creados por la herramienta, se simularon fallas monofásicas (en la fase a) en todos los nodos, con valores de resistencia de fallas entre 0,05Ω a 40Ω en pasos de 4Ω. La base total de datos simulada fue de 2’970.000 fallas y el tiempo de procesamiento fue aproximadamente 12,5 días.

Primero, se realizó una pequeña prueba donde se escogieron aleatoriamente 74250 fallas y con esta base de datos se realizó una validación cruzada de cuatro subconjuntos, para determinar cómo respondía el localizador. La precisión promedio y la desviación estándar del localizador fue de 92,08 ± 0,051% y el proceso tardó aproximadamente 7,12 horas.

Adicionalmente, se realizó otra prueba donde se entrenó con 300000 fallas seleccionadas aleatoriamente de la base de datos, el cual tardó aproximadamente 32,58 horas. La validación se realizó con el resto de fallas (2’670.000 fallas) simuladas, donde se obtuvo una precisión del 81,09%, y tardó aproximadamente 22 minutos. El desempeño del localizador en esta prueba es muy bueno debido a que no se tuvieron muchas condiciones operativas y la base de prueba fue 8,9 veces mayor que la base de entrenamiento, lo cual muestra la capacidad de generalización del localizador. También, demuestra que la metodología es robusta, pese a que aún falta realizar más pruebas para tener un localizador mucho más confiable. Los datos de parametrización utilizados en estas pruebas son los que se muestran en la tabla B.1 y el PC utilizado en estas pruebas tiene las siguientes características: Core i5-3470 @3.20 GHz, 8GB RAM.