Implementación de un Algoritmo de Aprendizaje Automático para Determinar Características de Espejos Utilizando los Patrones de Prueba de Ronchi en el Taller de Óptica de la Unillanos

Una Tesis
Presentada al
Programa de Especialización en Ingeniería de Software

por

Daniel Zapata Medina

En Cumplimiento Parcial
de los Requisitos para el Grado
Especialista en Ingeniería de Software

Director:
Andrés Fernando Jiménez López, M.Sc.

Codirector:
Angel Alfonso Cruz Roa, Ph.D.

Universidad de los Llanos
Facultad de Ciencias Básicas e Ingeniería
Villavicencio, Meta
Julio 2017

Implementación de un Algoritmo de Aprendizaje Automático para Determinar Características de Espejos Utilizando los Patrones de Prueba de Ronchi en el Taller de Óptica de la Unillanos

Nota de aceptación

_____

_____

_____

_____

_____

_____

Aprobado por:

_____

Jurado 1: Ing. Esp. Juan Fajardo Barrero,

_____

Jurado 2:

_____

Director: Ing. Andres Jimenez, M.Sc

_____

Codirector: Ing. Angel Cruz, M.Sc, Ph.D

Fecha de Aprobación _____

A Dios, mi Creador, a él que hizo todo perfecto.
A mis padres Margarita y Basilio
y hermanos Juan y Mila por su apoyo incondicional.

INTRODUCCIÓN

Actualmente en el laboratorio de óptica, adscrito a la Facultad de Ciencias Básicas e Ingeniería de la Universidad de Los Llanos, se realiza la denominada prueba de Ronchi a los espejos esféricos y parabólicos fabricados; los cuales tienen diferentes usos (ejemplo: telescopios, interferómetros o periscopios) siendo la aplicación principal en los telescopios ópticos. La prueba otorga certificados de calidad a los espejos, pero se busca aumentar esos niveles al mejorar los sistemas de prueba.

La prueba de Ronchi se lleva a cabo utilizando un interferómetro y el análisis de los patrones de interferencia producidos, los cuales se denominan Ronchigramas[1]. En la actualidad la prueba es realizada por un experto en materia de óptica; es un procedimiento que resulta complejo, demorado y su resultado es aproximado. El análisis de Ronchigramas es una observación detallada de patrones modelos que se encuentran referenciados o que se han adquirido por la experiencia. Su interpretación limita la medición únicamente a la parte cualitativa y otros aspectos de la visión humana que pueden aumentar el error ya que intervienen otros factores como son el punto de vista geométrico: la forma de los patrones, la distancia entre las franjas, etc. y el modelo matemático de la imagen desde el punto de vista físico.

Con el fin de obtener los mejores resultados y de mayor confiabilidad, los cuales aumentarán la posibilidad de expedición de certificados de calidad con mayor validez, es necesario un mejoramiento del sistema en la parte de software, al implementar una herramienta que ejecute el análisis de la imagen capturada y aplique cálculos y procesos de comparación con una imagen modelo dentro de una base de datos completa de soporte digital, para determinar características ópticas en espejos.

La visión por computador se ha convertido en una técnica muy importante para el análisis de información presentada en imágenes. En esta ocasión se presenta la información relacionada con el estudio, análisis e implementación de un algoritmo de aprendizaje automático para determinar características de espejos utilizando los patrones de la prueba de Ronchi. El algoritmo de aprendizaje es implementado en el Módulo de análisis de Ronchigramas por Computador (MARC), en la herramienta de Software ANGMAR – *Image Processing* V1.0, que compone el Sistema Asistido por Computador de la Prueba de Ronchi en el laboratorio de óptica de la Universidad de los Llanos (SAPRULL)[2].

Este libro se compone de cinco capítulos: el Primer Capítulo se compone del preámbulo de la investigación y la prueba de Ronchi realizada en el laboratorio de óptica de la Universidad de los Llanos, donde se exponen los objetivos y la justificación. En el Segundo Capítulo se exponen los aspectos de visión por computador y procesamiento de Imágenes, en el Tercer Capítulo se presenta el contexto de Maquinas

de aprendizaje y sus aplicaciones, en el Cuarto Capítulo se consigan los aspectos de la investigación correspondientes a la metodología y arreglo experimental del sistema e implementación del algoritmo registrando los resultados obtenidos y sus discusiones respectivas. En el Quinto Capítulo, se presentan las conclusiones de la investigación realizada y perspectivas del trabajo futuro. Al final, se registra la citación bibliográfica utilizada.

AGRADECIMIENTOS

- Agradecer a mis directores, los Ingenieros Andrés Jiménez y Ángel Cruz por la paciencia y el acompañamiento necesario para poder realizar este trabajo de grado, brindándome la oportunidad de aprender las técnicas correctas de investigación, la perseverancia y la continua mejora de mis habilidades. Lo mejor para aumentar nuestras capacidades y conocimientos es la lectura combinada con la práctica, porque brindan ese ambiente propicio de intercambio de ideas y retroalimentación de cada saber.

- Un agradecimiento al Profesor Nelson Baquero y al Señor Fernando Quimbay por su apoyo, enseñanza y compañía para aprender y entender el proceso de la prueba de Ronchi y su configuración en el laboratorio y así lograr realizar este proyecto.

- Un agradecimiento igualmente a las Ingenieras Marla y Angela por su asesoría del estado actual del sistema SAPRULL para iniciar el proyecto.

- Quiero agradecer a mi familia, mis padres Margarita y Basilio, a mis hermanos Juan y Mila por su apoyo en todo momento y sus consejos que han ayudado en toda mi formación personal. Dios me ha regalado esta oportunidad para agradecer y permitirme estas oportunidades que tiene la vida.

- Finalmente un especial agradecimiento a los docentes y directores del programa de especialización en Ingeniería de Software de la Universidad de los Llanos por ayudar al crecimiento continuo de las habilidades de investigación, innovación y emprendimiento en el campo del desarrollo de software.

Índice general

Índice de cuadros

Índice de figuras

RESUMEN

La prueba de Ronchi es un procedimiento que permite obtener patrones visuales (Ronchigramas), los cuales pueden usarse para determinar características ópticas en la superficie de espejos, como lo son las aberraciones y deformaciones ópticas. En este libro se presenta la implementación de un algoritmo de aprendizaje automático para la clasificación de imágenes de Ronchigramas, utilizando la técnica de extracción y descripción de características locales con el método *Scale Invariant Feature Transform* (*SIFT*), una bolsa de palabras visuales (*BOVWs*) para representación de imágenes y el entrenamiento del algoritmo de árboles de decisión (*Random Forest-RF*) en el esquema de clasificación con un rendimiento alcanzado de $0,88$ en términos de la medida de precisión media. Se realizó la selección de este algoritmo porque arrojó mejores resultados al ser comparado con el rendimiento del algoritmo de una Máquina de Vectores de Soporte (*Support Vector Machine-SVM*) que fue entrenado y evaluado con el mismo conjunto de datos y alcanzó un rendimiento de clasificación de $0,74$ en la medida de precisión media. El algoritmo es implementado en el Módulo de análisis de Ronchigramas (MARC), en la herramienta de Software ANGMAR – I*mage Processing* V1.0, que compone el Sistema Asistido por Computador de la Prueba de Ronchi en el taaller de óptica de la Universidad de los Llanos (SAPRULL)

Palabras claves: Bolsa de palabras visuales, Procesamiento de imágenes, Aprendizaje de máquina, Defectos ópticos, Espejos, Ronchi test, Máquina de Vectores de Soporte, árboles de decisión.

ABSTRACT

The Ronchi test is a procedure that allows to obtain visual patterns (Ronchigramas), which can be used to determine the optical characteristics on the surface of the mirrors, such as aberrations and optical deformations. This book presents the implementation of an algorithm of automatic learning for the classification of images of Ronchigramas, using the technique of extraction and the description of local features with the Scale Invariant Feature Transform ($SIFT$) method, a bag of visual words ($BOVWs$) for the representation of images and the training decision tree algorithm (*Random Forest-RF*) in the classification scheme with a performance achieved $0,88$ in terms of the average precision measure. The selection of this algorithm was performed because it showed better results in comparison to the Support Vector Machine (SVM) performance that was trained and evaluated with the same data set and the performance achieved was $0,74$ of the average precision measure. The algorithm is implemented in the Analysis Module of Ronchigramas ($MARC$), in the Software Tool *ANGMAR* - Image Processing V1.0, which composes the Computer Assisted System for the Ronchi test at the Optics Laboratory of the University of Los Llanos ($SAPRULL$).

Keywords: Image processing, Machine learning, Optical defects, Mirrors, Ronchi test, Support Vector Machine, Decision tree.

Capítulo I

PRUEBA RONCHI REALIZADA EN EL LABORATORIO DE ÓPTICA DE LA UNIVERSIDAD DE LOS LLANOS

1.1. Fabricación de espejos

La Universidad de los Llanos cuenta con un laboratorio de óptica, único en el país a nivel universitario, en el cual se producen componentes ópticos en vidrio. Actualmente en el laboratorio de óptica, se realiza la denominada prueba de Ronchi a los espejos esféricos y parabólicos fabricados; que tienen diferentes usos (telescopios, interferómetros, periscopios) y actualmente una aplicación principal en la fabricación de telescopios ópticos. La prueba otorga certificados de calidad a los espejos, pero se busca aumentar esos niveles al mejorar los sistemas de prueba.

Las fases para la fabricación de un espejo son: diseño, elaboración y pruebas ópticas (ver *Figura 1*). La prueba óptica de Ronchi es realizada a través del Sistema Asistido por Computador (SAPRULL).

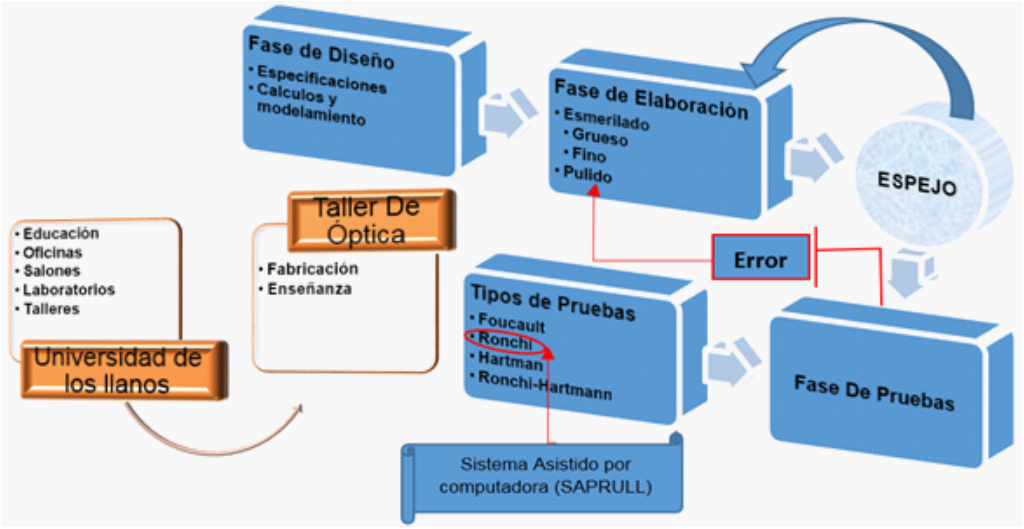

Figura 1: Fases de Fabricación de Espejos.

1. En la *Fase de diseño*, se detallan las especificaciones de uso y se realizan los cálculos, en el contexto teórico de la Física. Esta tarea es realizada manualmente y se procede al modelamiento ideal del espejo.

2. En la *Fase de elaboración*, es donde se realiza el proceso practico de construcción de un espejo, utilizando los dispositivo y máquinas que se encuentran en el laboratorio de óptica para la manipulación de la materia prima (el vidrio), este proceso se lleva a cabo en tres etapas [2]:

   a) *Esmerilado grueso*: Se utiliza la herramienta denominada esmeril y se procede a dar la concavidad al espejo.

   b) *Esmerilado fino*: En este proceso se da el perfil y la curvatura para el espejo.

   c) *Pulido*: Permite darle la forma final a la superficie.

3. En la *Fase de prueba*, se realizan diversas pruebas ópticas al espejo bajo estudio, verificando el estado para que se aproxime al modelo ideal.

El presente libro se enfatiza la prueba Ronchi, la cual es la más trabajada en el taller de óptica de la Universidad de los Llanos y actualmente modernizada, al instalarse el sistema asistido por computador SAPRULL.

1.2. Prueba Ronchi

La prueba de Ronchi es una prueba muy utilizada en los centros de óptica y en talleres de observación astronómica para determinar aberraciones ópticas despues del pulido de un espejo, además porque se compone de un método muy simple, pero a la vez provee resultados muy eficaces al ser interpretada correctamente. Desde su descubrimiento ha tenido aplicaciones desde diferentes perspectivas, ya sea con un análisis cualitativo o cuantitativo. La interpretación de la prueba de Ronchi casi siempre ha sido desde el punto de vista de la óptica geométrica, la cual es adecuada en casos generales y con fines prácticos. Esta interpretación tiene algunas limitaciones que son resueltas por el modelo físico y amplían el panorama de observación, ya que debe tenerse en cuenta los efectos ocasionados por la difracción, como son, el número de imágenes que se forman en la difracción (Ronchigramas), efectos de borde y ancho de banda de transmisión de la rejilla.

Históricamente la prueba de Ronchi es uno de los métodos más utilizados para evaluar y medir las aberraciones de un sistema óptico. *"El físico italiano Ronchi descubrió que cuando una rejilla se coloca cerca del centro de curvatura de un espejo, la imagen de la rejilla se superpone a la misma rejilla, produciendo un tipo de patrón Moiré que llamó franjas de combinación. Puesto que la forma de estas franjas de combinación depende de las aberraciones del espejo, pensó inmediatamente en aplicar el fenómeno a las pruebas de calidad de los espejos"* [1]. Estas franjas de combinación son imágenes (patrones de interferencia) finalmente denominadas **Ronchigramas**.

1.3. Fenómenos de Difracción e Interferencia

En física, algunos fenómenos característicos de las ondas son la difracción e interferencia los cuales consisten en la desviación de las ondas al atravesar una rendija (formando una imagen) o al encontrase con algún obstáculo y en la combinación de dos o más ondas que se traslapan en un punto determinado, respectivamente[12]. *"En ondas de luz se observa una configuración de interferencia si 1) las fuentes son coherentes y 2) las fuentes tienen longitudes de onda idénticas"* [13].

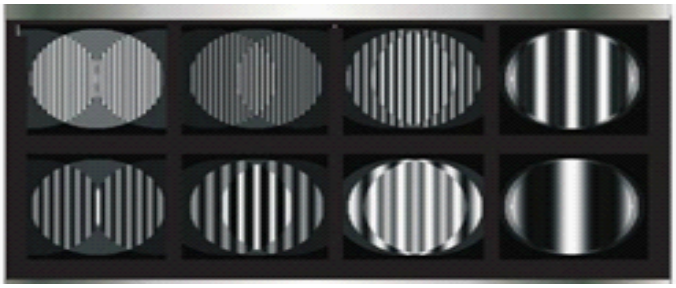

Figura 2: Imágenes Múltiples Generadas por una Rejilla de Difracción. **Fuente:** *Teoría Física de Rejillas de Amplitud y Fase con aplicaciones en la prueba de Ronchi (2011)*[7].

La difracción ocasiona algunos efectos que deben tenerse muy en cuenta para reducir el error en la interpretación de los resultados, uno de ellos, son las imágenes múltiples, ver *Figura 2*, las cuales son imágenes generadas por las bandas de la rejilla de difracción. En el caso de la prueba Ronchi se trabajan rejillas del orden de 2-8 líneas/mm y de esta forma el efecto no es muy evidente. El efecto también puede ser reducido utilizando luz blanca.

Figura 3: Efectos de borde. **Fuente:** *Optical Shop Testing (2007)*[1].

Un segundo aspecto a tratar, es el efecto de borde, ver *Figura 3*, el cual es ocasionado por el solapamiento de los órdenes de difracción, y se observa una ilusión de la línea doblada hacia afuera en los bordes del Ronchigrama, *"esto se debe a que el área del borde traslapado es más grande estando más lejos del centro de curvatura del espejo"* [7].

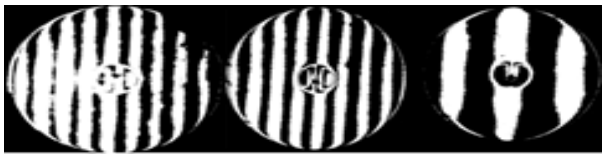

Figura 4: Efectos de ancho de banda. **Fuente**: Autor.

Un efecto final que debe considerarse al realizar la prueba Ronchi, es la dependencia del ancho de banda, es decir, el ancho de las bandas de la rejilla cuya transmisión varía produciendo diferentes patrones y que a su vez determina características evidentes o más profundas dependiendo del número de franjas en el Ronchigrama, ver *Figura 4*.

1.3.1. Teoría Geométrica

La prueba de Ronchi tiene un modelo geométrico y un modelo físico que son equivalentes. En el modelo geométrico, las franjas se interpretan como sombras de las bandas de la rejilla. En el modelo físico, las franjas se interpretan como sombras debido a la difracción y la interferencia. En el análisis del modelo geométrico, la observación es limitada, aunque para interpretar Ronchigramas con fines generales y prácticos es perfectamente adecuada y funciona en todo su estudio.

Como se explica en el libro de pruebas ópticas por Malacara [1], las pruebas ópticas permiten determinar características de espejos y medir sus aberraciones ópticas. *"La prueba de Ronchi mide la aberración transversal (TA) en una forma directa como muestra la Figura 5; en esta figura el objeto y la imagen están en el eje óptico, así la aberración transversal es medida desde el eje y puede ser vista al incluir desenfoque, así como otras aberraciones"* [1].

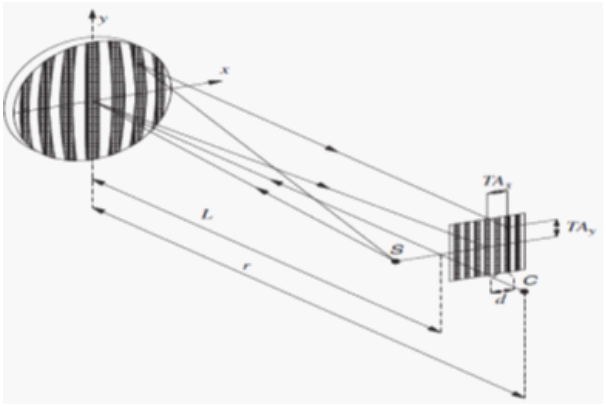

Figura 5: Geometría de la Prueba de Ronchi. **Fuente:** *Optical Shop Testing (2007)*[1].

La aberración de onda está definida usando la fórmula de Rayces:

$$\frac{\partial W}{\partial_X} = -\frac{TA_x}{r-W}; \qquad \frac{\partial W}{\partial_y} = -\frac{TA_y}{r-W} \qquad (1,4,1)$$

Se puede suponer que $W \leq r$ y se reescribe la *ecuación 1.4.1,* así:

$$\frac{\partial W}{\partial_X} = -\frac{TA_x}{r}; \qquad \frac{\partial W}{\partial_y} = -\frac{TA_y}{r} \qquad (1,4,2)$$

Donde r, es la distancia desde la superficie bajo prueba hasta la rejilla de Ronchi.

Entonces, si se asume una rejilla Ronchi espaciada d entre bandas para un punto (x, y), sobre determinada franja, en general se puede escribir:

$$\frac{\partial W}{\partial_X} \cos\varphi - \frac{\partial W}{\partial_y} \sin\varphi = -\frac{md}{r} \qquad (1,4,3)$$

Donde φ se asume como el ángulo de inclinación de la rejilla con respecto al eje y. Esta es fórmula básica del modelo geométrico de la prueba de Ronchi.

1.3.2. Patrones de Ronchi para aberraciones primaria

El frente de onda de un sistema con aberraciones primarias puede describirse aí:

$$W = A(x^2 + y^2)^2 + By(x^2 + y^2)^2 + C(x^2 + 3y^2) + D(x^2 + y^2) \qquad (1,4,2,1)$$

Donde A, B y C son cocientes de aberraciones, Esféricas, Coma y Astigmatismo, respectivamente.

El último coeficiente es el Defoco dado por la distancia l‘ entre la rejilla de Ronchi al foco paraxial del espejo dado por:

$$D = \frac{l‘}{2r^2}$$

El Defoco no causa distorsión en la imagen y se debe a un error longitudinal, ya que la imagen se aproxima en posiciones distintas al foco ideal, como se aprecia en la *Figura 6*, a diferentes desplazamientos dentro o fuera de foco se producirá un número de franjas y estando en el foco solo se observará el espejo sin franjas y por lo tanto no se considera como una aberración [1].

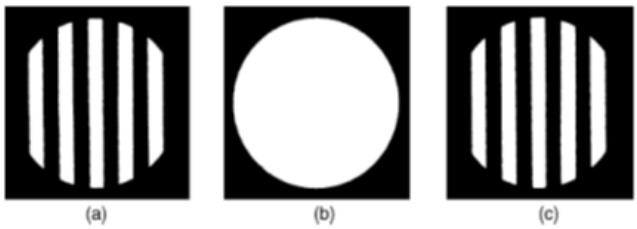

Figura 6: Ronchigrama con Defoco. (a) Fuera de Foco. (b) En el Foco. (c) Dentro del Foco. **Fuente**: *Optical Shop Testing (2007)*[1].

No se incluyen términos de tilt (inclinaciones) porque la prueba de Ronchi es inmune ante este tipo de aberraciones. Sustituyendo la *ecuación 1.4.2.1* en la *ecuación 1.4.3* se tendrá:

$$4A(x^2+y^2)^2(x\cos\varphi - y\sin\varphi) + B\left[2xy\cos\varphi - (3y^2+x^2)\sin\varphi\right] + 2C(x\cos\varphi - 3y\sin\varphi) + 2D(x\cos\varphi - y\sin\varphi) = -\frac{md}{r} \qquad (1,4,2,2)$$

En el estudio de las aberraciones es conveniente aplicar una rotación φ a la expresión por medio de las relaciones:

$$\begin{aligned} x &= \eta\cos\psi + \xi\sin\psi \\ y &= -\eta\sin\psi + \xi\sin\psi \end{aligned} \qquad (1,4,2,3)$$

Con η y ξ como los nuevos ejes de coordenadas.

1.3.3. Determinación de las deformaciones en espejos

Las deformaciones en espejos pueden ser determinadas utilizando el análisis de Ronchigramas. "*Probablemente Pacella (1927) fue el primero en identificar algún tipo de aberración*" [1]. Al observar los patrones de Ronchi, un Ronchigrama puede arrojar una idea cualitativa de alguna deformación en el espejo bajo estudio.

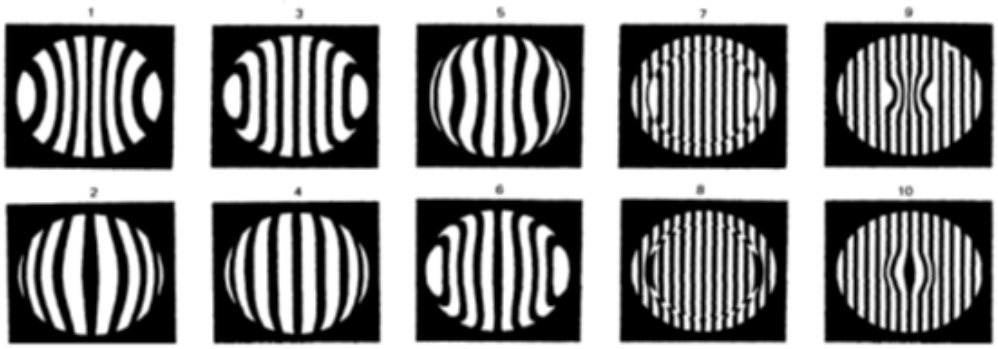

Figura 7: Ronchigramas para algún tipo de deformación en la superficie. **Fuente**: *Optical Shop Testing (2007)*[1].

Para entender más algunos de los patrones ver *Figura 7* y su correspondiente defecto en la superficie del espejo puede verse la *Figura 8*, donde además pueden relacionarse las anteriores ilustraciones mencionadas con el *cuadro 2*, presentadas en el libro de pruebas ópticas para el contexto de la prueba de Ronchi (Daniel Malacara).

Este análisis cualitativo, permite observar el patrón que se forma y a nivel general obtener resultados fiables del tipo de deformaciones en la superficie de un espejo.

Como se observa en cada figura de la *Figura 7*, existe un patrón característico para una aberración determinada y que a su vez varias deformaciones pueden estar presentes en una misma superficie del espejo bajo estudio.

Cuadro 1: Relación entre Ronchigramas de la *Figura 7* y deformación de la *Figura 8*[1].

	Superficie			
	Reflexión		Refracción	
Ronchigrama	Rejilla Fuera del foco	Rejilla Dentro del foco	Rejilla Fuera del foco	Rejilla Dentro del foco
1	B	A	A	B
2	A	B	B	A
3	D	C	C	D
4	C	D	D	C
5	F	E	E	F
6	E	F	F	E
7	H	G	G	H
8	G	H	H	G
9	J	I	I	J
10	I	J	J	I

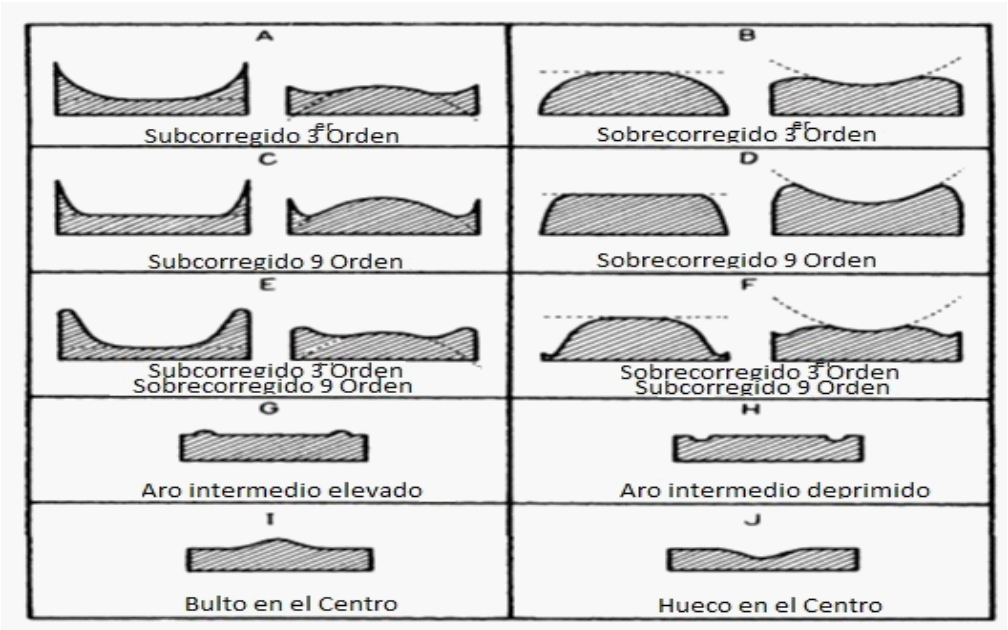

Figura 8: Deformaciones en la superficie correspondiente a los Ronchigramas de la Figura 7. **Fuente:** *Optical Shop Testing (2007)*[1].

Capítulo II

VISIÓN POR COMPUTADOR

La inteligencia artificial busca construir maquinas inteligentes, que en cierta forma imiten patrones del comportamiento humano. Este tipo de inteligencia se basa en la aplicación de algoritmos de aprendizaje y se ve reflejado en campos como la robótica, la visión por computador, sistemas de análisis de datos, etc [14]. Actualmente podemos obtener imágenes fácilmente de forma digital, las cuales pueden ser analizadas por algoritmos diseñados en programas de computador, los cuales buscan entender las imágenes y sus características, convirtiéndose en el campo especifico de la visión por computador.

La visión por computador es una poderosa herramienta en diferentes aplicaciones como la búsqueda de imágenes en sistemas geográficos, análisis de imágenes médicas, identificación de patrones en sistemas complejos, entre otros [3]. La idea básica en la visión por computador es automatizar la extracción de información de las imágenes, dicha información puede ser la detección y reconocimiento de patrones, la clasificación de imágenes por características, búsqueda de imágenes, posicionamiento de cámaras; la información se analiza con el fin de imitar la visión humana, ya sea a través de métodos de aproximación estadística o análisis geométrico [14] [15].

2.1. Representación de las Imágenes

La imagen es una señal que contiene información y sobre ella se trabajan las principales tareas de la visión por computador, ya que puede representarse como funciones matemáticas, como matrices, conjuntos o grafos y estas representaciones pueden estar bajo estudio para realizar cálculos computacionales. Estudiando la imagen como una señal es posible definirla a través de cinco características presentes en cualquier señal (*ver cuadro 3*).

Cuadro 2: Características de una señal. **Fuente**: *Procesamiento y análisis de imágenes digitales*[3].

Característica	Valores	
Número de Variables	una variable	múltiples variables
Dimensionalidad	escalar	vectorial (multicanal)
Variables Independientes	discretas	continuas
Valores de la Señal	discretos	continuos
Naturaleza Estadística	deterministas	aleatorias

2.1.1. Imágenes digitales

En el espacio discreto, las imágenes pueden representarse en forma de rejilla rectangular, definidas por el dominio de coordenadas discretas y finitas. Dicho dominio es mapeado hacia un conjunto de valores vectoriales en $\mathbb{R}^n$, donde n equivale a la dimensión de la señal. Así, la función f representará una imagen en el espacio discreto y se define como:

$$f \longrightarrow \mathbb{R}^n$$

"*donde* $\chi = \{0, 1, 2, 3, 4....D-1\}^d \subset \mathbb{N}^d$ *es el conjunto de posiciones validas de los pixeles en un espacio de d dimensiones. Así* $f\ (\chi)$*es una función que retorna al vector que representa la composición espectral de una imagen sobre la posición* χ*. Si el codominio* $\mathbb{R}^n$ *es remplazado por un conjunto discreto de valores* $\bar{V}_n$ *entonces se tiene una imagen digital*"[3].

En general, las funciones que representan una imagen se pueden definir como vectores. Luego la función $F(\chi)$ tendrá un canal k con una única componente $f_k(\chi)$, $k = 1, 2, 3....n$.

Y en una imagen bidimensional implicará la notación de un vector como par ordenado $\bar{V} = (x, y)$.

En el manejo computacional de imágenes bidimensionales, el dominio de la función puede representarse en forma de matrices, luego la matriz F representa a la misma señal dada por la función f si se cumple:

$$f_{\ ji} = f\left(\left[\begin{array}{c} i \\ j \end{array}\right]\right)$$

$$F = \left[\begin{array}{cccccc} f_{0,0} & f_{0,1} & f_{0,2} & . & . & f_{0,C-1} \\ f_{1,0} & f_{1,1} & f_{1,2} & . & . & f_{1,C-1} \\ f_{2,0} & f_{2,1} & f_{2,2} & . & . & f_{2,C-1} \\ . & . & . & . & & . \\ . & . & . & & . & . \\ . & . & . & & . & . \\ f_{R-1,0} & f_{R-1,1} & f_{R-1,2} & . & . & f_{R-1,C-1} \end{array}\right]$$

con R igual al número de filas y C igual al número de columnas.

De igual manera, si $f_{ji} \in \mathbb{R}^n$ se tiene una imagen en espacio discreto, y si $f_{ji} \in \mathbb{N}^n$ se dice que F es una imagen digital.

La resolución de una imagen hace referencia al tamaño T en pixeles que contienen la imagen. Las dimensiones físicas ($ancho \times alto$)de una fotografía digital, por ejemplo, tendrá un tamaño T de 4608×3072 px y una resolución de (14155776 *pixeles*), alrededor de 14,2 *Megapíxeles*.

2.2. Clasificación de Imágenes

Se pueden definir seis fases generales en un sistema de visión de computador para la clasificación de patrones, ver *Figura 9.*

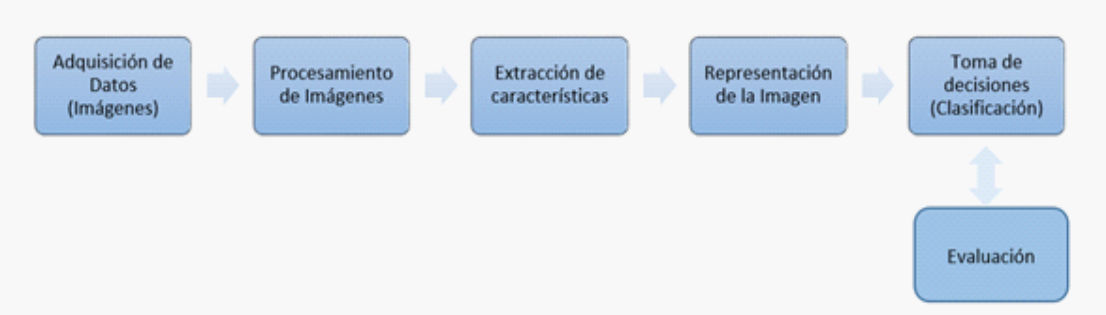

Figura 9: Fases de un Sistema Clasificador de Imagenes. **Fuente**: Autor.

1. **Adquisición de Imágenes:** En esta fase se realiza el reconocimiento básico de la información, y su forma de captura. En el caso de imágenes, el sensor utilizado puede ser la cámara digital (*CCD-CMOS*).

2. **Acondicionamiento de Imágenes:** Incluye técnicas de procesamiento de imágenes, como el filtrado para la reducción de ruido y mejores condiciones de análisis por computador.

3. **Extracción de características:** En esta etapa se realiza el proceso de detección y descripción de puntos o características de interés de una imagen.

2. **Representación de la Imagen:** En esta etapa se asignan valores representativos a las características extraídas en cada imagen.

3. **Toma de decisiones (Clasificación):** En esta fase se interpretan o estiman los resultados y se procede a la clasificación de la información obtenida.

4. **Evaluación:** Es el proceso necesario para medir los indicadores de desempeño, como precisión y exactitud del rendimiento del clasificador seleccionado y de esta manera adecuar los parámetros y mejorar los resultados de clasificación.

Continuando con el proceso de clasificación de imágenes, se puede asegurar que la técnica se basa en asignar etiquetas a una imagen a fin de reconocer el contenido y obtener palabras claves que relacionan el contenido visual de la imagen.

El proceso de clasificación de una imagen requiere métodos de aprendizaje máquina, para, "aprender" características visuales más comunes entre imágenes y aquellas que las diferencian entre sí. Esta clasificación utiliza la asignación de clases iniciales que identifican ciertos tipos de imágenes con características específicas y similares[8].

Los tipos de clasificación dependen de la estrategia o técnica manejada, en algunos casos la clasificación puede ser supervisada, parcialmente supervisada o no supervisada[16], y es donde se relacionan con el tipo de aprendizaje utilizado, ya que existen diferentes algoritmos de aprendizaje que dan soluciones efectivas de acuerdo al caso de estudio y que se abordara de forma más específica en el siguiente capítulo de libro.

Al tener un conjunto de clases previamente separadas y organizadas que identifican un patrón para cada categoría, es la identificación previa de la similitud entre imágenes de acuerdo a una serie de parámetros que contiene cada imagen bajo estudio. Los parámetros presentes en cada imagen entregan la variabilidad del contenido visual de las imágenes ver *Figura 10*.

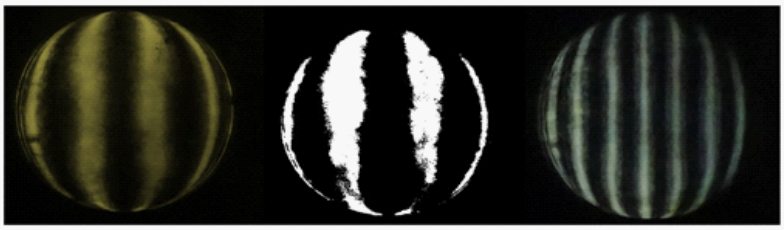

Figura 10: Ronchigramas con Parámetros de Luz diferente. **Fuente:** Autor.

2.2.1. Extracción de Características

Siguiendo con el esquema general de la clasificación de imágenes, existen varios métodos para la extracción de características, inicialmente este proceso viene precedido del acondicionamiento de la imagen como señal, ya que debe pasar por filtros y transformaciones que ayuden a preparar los datos bajo análisis. Las características presentes en una imagen puede variar dependiendo del contenido de la imagen y del método de extracción; los métodos (detección y descripción) calculan a partir de un algoritmo, los puntos de interés de la imagen para luego obtener una representación de la imagen (*Figura 11*).

Figura 11: Extracción de Caracteristicas. **Fuente**: Autor.

2.2.2. *Scale Invariant Feature Transform* (*SIFT*)

Es un algoritmo de detección y descripción de características de una imagen, *ya que transforma los datos de la imagen en coordenadas escalar-invariantes en relación con las características locales. Un aspecto importante de este enfoque es que genera un gran número de características que cubren densamente la imagen en toda la gama de escalas y ubicaciones* [17].

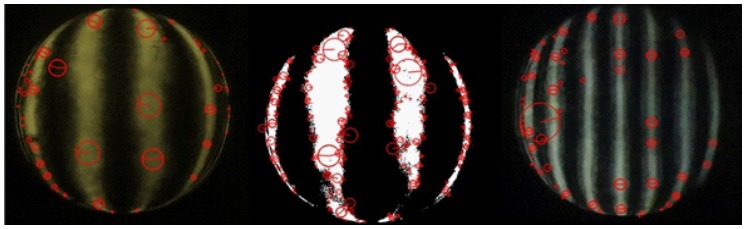

Figura 12: Extracción de Características con *SIFT*. **Fuente**: Autor

Una imagen de tamaño 256×256 *píxeles* dará lugar a un número determinado de características invariantes (este número depende del contenido de la imagen y de los parámetros de variación presentes en la imagen, ver *Figura 12*). La extracción y descripción de puntos para la imagen da lugar a resultados diferentes como se observa en el Ronchigrama de la derecha donde se extrajeron 60 *keypoints* (puntos de interes), el Ronchigrama del centro presenta 292 *keypoints* y el de la izquierda presenta 82 *Keypoints*.

2.2.3. Característica Local (*Local Feature*)

Es un parámetro de las imágenes que describe las propiedades de un píxel con relación a su entorno; proporcionando información sobre estructuras de la imagen, un ejemplo de característica local, son las regiones (blobs) o contornos (edges) en una imagen [18]. Para detectar características locales se puede aplicar la transformación en cada punto de la imagen definiendo así un filtro o *kernel* de forma local, a esta operación de transformación usada en teoría de señales se le conoce como convolución, con el fin de descomponer una señal en una serie de funciones más simples.

A continuación, se presentan las principales etapas de cálculo utilizadas para generar el conjunto de características de la imagen[17]:

1. Detección de extremos en la escala-espacio: Se implementa utilizando la función Gaussiana, un método seleccionado como único Kernel para identificar puntos de interés potenciales que son invariantes a escala y orientación, en esta etapa de cálculo, se exploran todas las escalas y ubicaciones de las imágenes.

El espacio-escala de una imagen se define como una función, $L(x, y, \sigma)$, que se produce a partir de la convolución, con una función gaussiana centrada en cada uno de los píxeles de la imagen, $G(x, y, \sigma)$, con una imagen de entrada, $I(x, y)$:

$$L(x, y, \sigma) = G(x, y, \sigma) \star I(x, y) \qquad (2,2,3,1)$$

donde * es la operación convolucion en x y y,

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} c^{\frac{-\left(x^2+y^2\right)}{2\sigma^2}} \qquad (2,2,3,2)$$

El grado de suavizado depende de la desviación estándar σ de la función gaussiana (*ecuación* 2,2,3,2*;* determina la escala (*resolución*) a la que se detectan los cambios de intensidad)[8].Valores mayores de σ implican mayor suavizado de la imagen, Ver *Figura 13*.

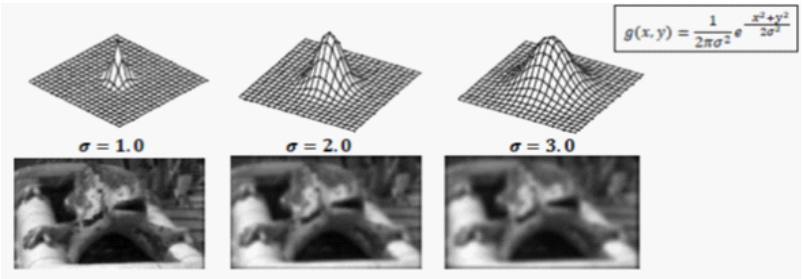

Figura 13: Filtrado Gaussiano. **Fuente:** *Introducción a la clasificación de imágenes. Detección de características locales: SIFT. Ernest Valveny* [8]

La diferencia de gausianas (*DoG*), ver *Figura 14*, se obtiene de la diferencia entre dos imágenes suavizadas con un filtro gaussiano con σ creciente, detectando regiones con cambios de intensidad. El valor final de cada píxel se obtiene a partir de la contribución de todos sus vecinos ponderados según la función gaussiana [8].

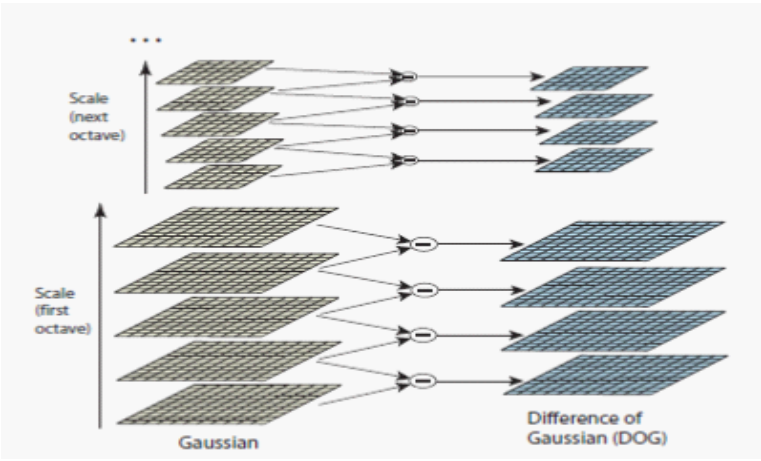

Figura 14: Diferencias de Gaussianas*(DoG)*. **Fuente**: http://docs.opencv.org/3.1.0/da/df5 /tutorial_py_sift_intro.html

2. Localización del punto clave: Los puntos de interés se seleccionan sobre las medidas de su estabilidad, es decir cada característica local es un modelo detallado apto para determinar la ubicación y la escala.

Los puntos de interés se detectan en máximos y mínimos locales que sean a la vez máximos o mínimos respecto a las escalas anterior y posterior para asegurar que se detectan a la escala correcta, es decir la escala de mayor respuesta[8].

3. Asignación de orientación: Se asignan uno o más vectores a cada ubicación del punto de interés basado en las vectores del gradiente de la imagen local. Todas las operaciones futuras se realizan en datos de imagen que se han transformado en relación con la orientación, escala y ubicación asignadas para cada característica, proporcionando de este modo invariancia a estas transformaciones.

Para seleccionar las regiones de interés se utiliza un gradiente que consiste en el cambio direccional en la intensidad de la imagen (*Figura 15*), el cual es definido por dos valores, la Magnitud del cambio en la dirección de máxima variación y la dirección donde el cambio de intensidad es máximo (*ecuaciones* 2,2,3,3 - 2,2,3,4).

$$dx = I\,(x+1, y) - I(x-1, y), \qquad dy = I\,(x, y+1) - I\,(x, y-1) \qquad (2,2,3,3)$$

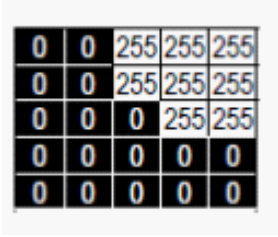

Figura 15: Cambios de Intensidad, en regiones de interés para una imagen a escala de grises. **Fuente**:*Introducción a la clasificación de imágenes. Detección de características locales: SIFT. Ernest Valveny [8].*

$$\theta\,(x, y) = \arctan\left(\tfrac{dy}{dx}\right)$$
$$m\,(x, y) = \sqrt[2]{dx^2 + dy^2} \qquad (2,2,3,4)$$

4. Descriptor del punto clave: Los gradientes de la imagen local se miden en la escala seleccionada, en la región alrededor de cada punto clave. Estos se transforman en una representación que permite niveles significativos de distorsión de forma local y cambio en la iluminación (*Figura 16*). El método del gradiente descendente se basa en buscar la dirección donde una mínima variación del vector de pesos hace que decrezca el error de manera más rápida (*ecuación* 2,2,3,5).

$$\bigtriangledown x\left(x,y\right)=ix\left(x,y\right)-\frac{x}{\delta x}\qquad\bigtriangledown y\left(x,y\right)=iy\left(x,y\right)-\frac{y}{\delta x}\qquad(2,2,3,5)$$

$$w_i\left(x,y\right)=\begin{cases}\bigtriangledown x\left(x,y\right),i=ix\left(x,y\right)\\1-\bigtriangledown x\left(x,y\right),i=ix\left(x,y\right)+1\\0,\textit{en caso contrario}\end{cases}\qquad(2,2,3,6)$$

$$w_j\left(x,y\right)=\begin{cases}\bigtriangledown y\left(x,y\right),j=iy\left(x,y\right)\\1-\bigtriangledown y\left(x,y\right),j=iy\left(x,y\right)+1\\0,\textit{en caso contrario}\end{cases}$$

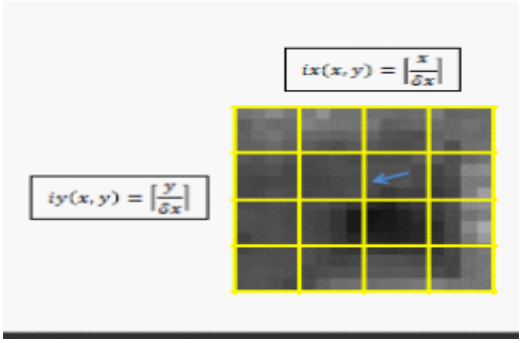

Figura 16: Gradiente descendete en una región de interés. **Fuente**:*Introducción a la clasificación de imágenes. Detección de características locales: SIFT. Ernest Valveny [8].*

$$h_k=\sum_{x,y}w_k\left(x,y\right)w_i\left(x,y\right)w_j\left(x,y\right)m\left(x,y\right)G\left(x,y\right)\qquad(2,2,3,7)$$

2.2.4. Representación de la Imagen

En el esquema de clasificación de imágenes, surgen inconvenientes para el tratamiento de la imagen, que van desde el costo computacional de la comparación de imágenes, la sensibilidad al ruido y variabilidad entre imágenes y además la representación de la imagen debe ser un vector numérico, el cual pueda ser invariante para mejorar los niveles de procesamiento; por lo anterior una de las etapas más relevantes es la representación de la imagen, ya que en esta etapa se identifica el entorno de características de una imagen y para ello se busca un modelo que permita identificar características comunes y relevantes.

El modelo conocido como *Bag of Visual Words* (*BOVW*, bolsa de palabras), busca resolver la identificación de las características extraídas inicialmente, *este modelo se basa en la cuantificación vectorial de los descriptores afines invariantes sobre parches de la imagen. Las principales ventajas del método son, que es simple, computacionalmente eficiente e intrínsecamente invariante. Además, se ha demostrado que el método es robusto para el desorden de fondo y produce una buena precisión de categorización incluso sin explotar información geométrica*[19].

⁕ Los principales pasos, para la implementación del método *BOVW*[19], son:

1. Detección y descripción de puntos de interés de una imagen.
2. Asignación de descriptores locales a un conjunto de clústeres predeterminados (un vocabulario) con un algoritmo de cuantificación vectorial.
3. Construir la bolsa de característica, que cuenta el número de características locales asignados a cada grupo.
4. Aplicar un clasificador de varias clases, tratando el Bag of Visual Words como el vector de características y así determinar qué categoría asignar a la imagen.

El modelo *BOVW*, necesita la construcción de un vocabulario de palabras visuales (ver *Figura 21*), que consisten en la agrupación de características locales. Por lo tanto, los descriptores extraídos en el primer paso deben ser Invariantes a variaciones que son irrelevantes para la tarea de categorización (transformaciones de imagen, variaciones de iluminación y oclusiones) pero lo suficientemente ricas como para llevar suficiente información como para ser discriminatorias a nivel de categoría[19]. Los pasos del método *BOVW*, están diseñados para reducir el costo computacional y maximizar la exactitud de la clasificación.

2.2.5. Construcción del vocabulario Visual

El vocabulario visual es una forma de construir un vector de características, para luego clasificar y así relacionar los descriptores de las nuevas imágenes bajo consulta con los descriptores del vocabulario creado a partir de entrenamientos (*Figura 17*).

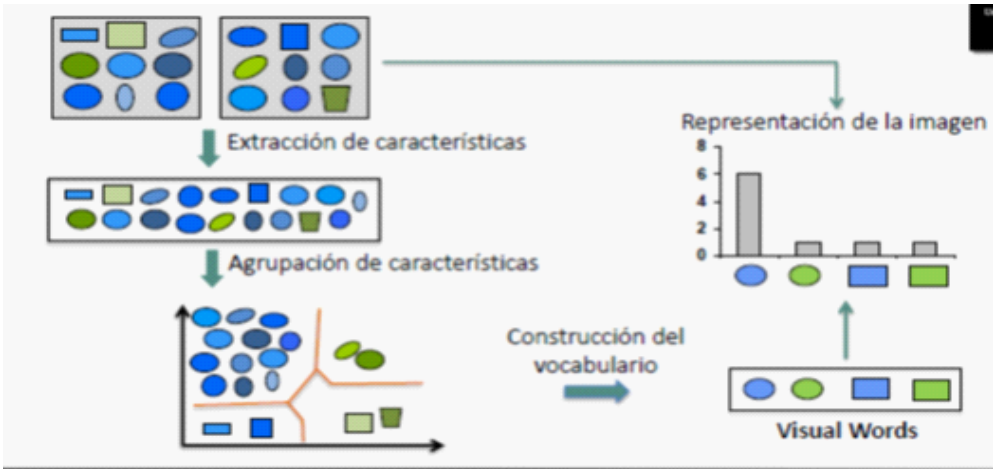

Figura 17: Construcción de un Vocabulario Visual. **Fuente:***Introducción a la clasificación de imágenes. Detección de características locales: SIFT. Ernest Valveny* [8].

Luego de agrupar las características, se genera un vocabulario representativo afín de organizarlas y para ello se utilizan los "*métodos de agrupamiento sobre todo el conjunto de palabras visuales y los centroides de los grupos obtenidos serían equivalentes a los términos que componen el diccionario*"[20],un método para realizar este agrupamiento es el método conocido como: *K-means.*

El algoritmo *K-means,* es un método iterativo de agrupamiento (Algoritmo de *Clustering*), basado en asignar a cada clúster la muestra con su representante más cercano con el fin de encontrar la agrupación que minimice la distancia de la muestra al representante de cada clúster. Su inicialización consta de un parámetro de entrada y que corresponde al número final de clústers (K). *De esta forma, el objetivo final del algoritmo consiste en encontrar una agrupación que minimice la distancia de todas las muestras al representante de cada uno de los clusters [8].*

Cuadro 3: Algoritmo de Agrupamiento K-means [4].

Algoritmo de Agrupamiento K-means
1. Escoge el número de agrupaciones(clusters) c y una medida de similitud S(a, b) entre dos objetos a y b. Inicialice los c centros de las agrupaciones (centroides)(es decir, seleccionando aleatoriamente c puntos de Z para ser estos centros). 2. Etiquetar todos los puntos de Z con respecto a su similitud con los centros de las agrupaciones: cada punto se asigna al grupo con el centro más similar. 3. Calcule los nuevos centros de las agrupaciones utilizando los puntos de la agrupación respectiva. 4. Repita los pasos 2 y 3 hasta que no se produzca ningún cambio en los centros.

- Los pasos del algoritmo de *K-means* son [8]:

  1. Inicializar los representantes de cada clúster de forma aleatoria con una muestra cualquiera del conjunto de entrenamiento (*Figura 18*).

Figura 18: Conjuntos de Entrenamiento.

2. Paso de asignación: asignar cada muestra al clúster con representante más cercano (*Figura 19*).

$$\{c_1, c_2, ..., c_k\} \quad | \quad c_j = \{x_i \text{ ı } d(x_i, c_j) = \text{mín}_r\, d(x_i, c_r)\} \qquad (2,2,5,1)$$

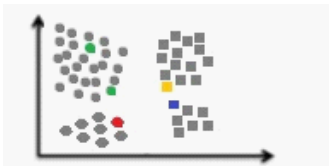

Figura 19: Representantes Más cercanos de cada muestra.

3. Paso de modificación: recalcular el representante de cada clúster como la media de todas las muestras asignadas al clúster (*Figura 20*).

$$c_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} d(x_i, c_j) \qquad (2,2,5,2)$$

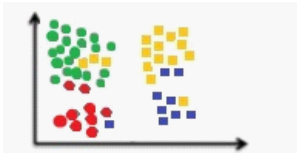

Figura 20: Muestras asignadas del conjunto de aprendizaje

4. Repetir pasos 2 (asignación) y 3 (modificación) hasta que no haya cambios de asignación. Luego se da la necesidad de fijar un número K de palabras visuales (Representantes de los K clústers que se obtienen con el algoritmo *K-means*). Entonces se procede a: probar diferentes valores y seleccionar el valor con mejor rendimiento en la clasificación (*Figura 21*). Normalmente, valores de K grandes (> 1000).

$$\hat{C} = \arg\text{mín}_c \sum_{j=i}^{k} \sum_{x_i \in C_j} d(x_i, c_j) \qquad (2,2,5,3)$$

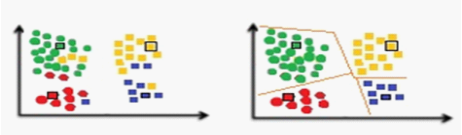

Figura 21: Representantes de los K clústers.

2.2.6. Histograma Normalizado

Luego de asignar los descriptores asociados a cada región de interés, se procede a asignar a cada característica local la palabra visual más cercana (*Figura 22*), acumulando el número de característica asignadas y comparándolas entre ellas. Para esta comparación donde se agrupan las características pueden usarse algoritmos de agrupación o de cuantificación como lo son las estrategias de indexación (*K-d-tree*) o métodos de búsqueda del vecino más cercano.

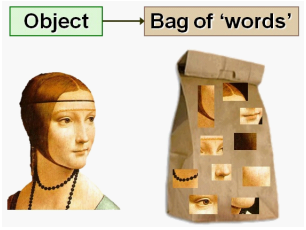

Figura 22: Bolsa de Palabras. **Fuente:**B*ag of Words Models for visual categorization: https://gilscvblog.com/2013/08/23 /bag-of-words-models-for-visual- categorization/*

- Ponderación del histograma: term frequency – inverse document frequency (*tf-idf*)
  - El valor de cada palabra en el histograma es función de dos valores:
  - *Term frequency* - *tf*: número de veces que la palabra aparece en la imagen (normalizado por el número total de características en la imagen, *Figura 23*)
  - I*nverse document frequency* - *idf*: las palabras más significativas (que aparecen con menos frecuencia en el conjunto de imágenes) deben tener un peso mayor

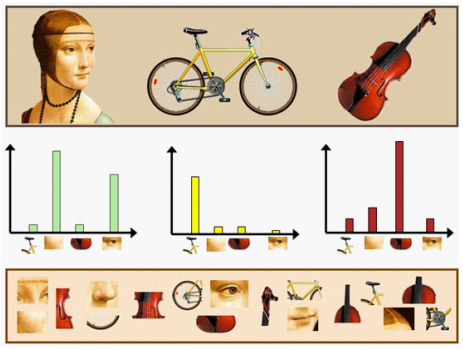

Figura 23: Histogramas de palabras para diferentes imagenes. **Fuente:**B*ag of Words Models for visual categorization: https://gilscvblog.com/2013/08/23 /bag-of-words-models-for-visual- categorization/*

Normalización del histograma: Necesidad de normalizar para evitar dependencia del número total de palabras en la imagen.

$$x_i = \frac{n_i^D}{N_D} \log \frac{N}{N_i}$$

n_i^D :número de repeticiones de la palabra i en la imagen D
N_D :número total de características en la imagen D
N : número total de imágenes utilizadas para construir el vocabulario
N_i :número de imágenes donde aparece la palabra i

Normalización L2

$$\Longrightarrow \qquad x' = \left(\frac{x_1}{\|x\|_2}, \frac{x_2}{\|x\|_2}, ... \frac{x_s}{\|x\|_2}\right) \qquad \|x\|_2 = \sqrt{\sum_{i=i}^{s} x_i^2} \qquad (2,2,6,1)$$

Normalización L1

$$\Longrightarrow \qquad x' = \left(\frac{x_1}{\|x\|_1}, \frac{x_2}{\|x\|_1}, ... \frac{x_s}{\|x\|_1}\right) \qquad \|x\|_1 = \sqrt{\sum_{i=i}^{s} |x_i|} \qquad (2,2,6,2)$$

Capítulo III

ALGORITMOS DE APRENDIZAJE

El filtrado de *spam* [14], en cuentas de correo, es un ejemplo de aprendizaje automático. Al analizar miles de correos electrónicos que han sido previamente etiquetados como s*pam* o no deseado, los filtros de spam aprenden a clasificar los nuevos mensajes para una cuenta de correo electrónico. De esta manera, el aprendizaje de máquina es el diseño y desarrollo de algoritmos que utilizan la experiencia pasada para tomar decisiones futuras.

Arthur Samuel, un informático que fue pionero en el estudio de la inteligencia artificial, dijo que el aprendizaje automático es: "*el estudio que da a las computadoras la capacidad de aprender sin ser explícitamente programadas*". A lo largo de los años 1950 y 1960, Samuel desarrolló programas que jugaban a las damas[14]. Dicha experiencia arrojo resultados de estudio sobre el objetivo fundamental del aprendizaje automático, ya que el computador aprendía a partir de ejemplos de aplicación de una regla en el juego de damas y lograba generalizar o inducir una regla desconocida para generar nuevas jugadas mejoradas con las cuales llegó hasta vencer a personas expertas en el juego.

De igual manera el informático Tom Mitchell define el aprendizaje automático más formalmente: *"Se puede decir que un programa aprende de la experiencia E con respecto a alguna clase de tareas T y medida de rendimiento P, si su desempeño en tareas en T medido por P, Mejora con la experiencia E"*[14].

3.1. Tareas de aprendizaje automático

Podemos separar los problemas de aprendizaje en unas pocas categorías: *aprendizaje supervisado y no supervisado*. En el aprendizaje supervisado los datos vienen con atributos adicionales que queremos predecir. Este problema puede ser solucionado por tareas comunes, que pueden ser realizadas por sistemas de aprendizaje automático, como son la clasificación y la regresión.

3.1.1. Clasificación:

Las muestras pertenecen a dos o más clases y queremos aprender de los datos ya etiquetados, ¿cómo predecir la clase de datos no etiquetados?. Un ejemplo de problema de clasificación, sería el ejemplo de reconocimiento de dígitos escrita a mano, donde el objetivo es asignar a cada vector de entrada un número finito de categorías discretas. Otra forma de pensar en la clasificación es en forma discreta (en oposición a continua) de aprendizaje supervisado, donde se tiene un número limitado de categorías y para

cada una de las n muestras proporcionadas, una de ellas trata de etiquetarse con la categoría o clase correcta[16].

En las tareas de clasificación, el programa debe aprender a predecir valores a partir de una o más variables explicativas. Por ejemplo, un programa podría aprender a clasificar imágenes observando las imágenes que han sido ordenadas inicialmente en una categoría. Suponga que tiene una colección de imágenes, las imágenes de cada categoría representan pájaros y ardillas. Una tarea, podría ser la clasificación de las imágenes en colecciones separadas de imágenes de pájaros y ardillas. Luego, para conocer el rendimiento del clasificador podría evaluarlo calculando el porcentaje de imágenes correctamente clasificadas. Y de igual manera, para nuevas imágenes con referencia a pájaros y ardillas el programa debe predecir la categoría, clase o etiqueta más probable a la que pertenece. Otras aplicaciones de clasificación tratan temas como: predecir si el precio de una acción subirá o bajará, o decidir a qué sección pertenece un artículo de noticias.

3.1.2. Regresión:

Si la salida deseada consiste en una o más variables continuas, entonces la tarea se llama regresión. Un ejemplo de un problema de regresión sería la predicción de la longitud de un salmón en función de su edad y peso[16].

En los problemas de regresión, el programa debe predecir el valor de una variable de respuesta continua. Los problemas de regresión incluyen por ejemplo, predecir las ventas de un nuevo producto o el salario de un trabajo basado en su descripción. Similar a la clasificación, los problemas de regresión requieren un aprendizaje supervisado[14].

En el Aprendizaje no supervisado, los datos de entrenamiento consisten en un conjunto de vectores de entrada χ sin valores objetivos correspondientes[21]. *El objetivo en este tipo de problemas puede ser para descubrir grupos de ejemplos similares dentro de los datos, donde se llama a la agrupación o para determinar la distribución de los datos dentro del espacio de entrada, conocida como estimación de la densidad, también para proyectar los datos de un espacio de alta dimensión bajo dos o tres dimensiones para el propósito de la visualización*[16]. Una tarea para este tipo de aprendizaje, consiste en descubrir grupos de observaciones relacionadas, llamadas clústers, agrupación o análisis de conglomerados, dentro de los datos de entrenamiento, asignando observaciones a grupos, con el fin de que las observaciones dentro de los grupos sean más similares entre sí, basándose en alguna medida de similitud que dentro de los otros grupos[4].

Los clústering a menudo se usan para explorar un conjunto de datos, *en aplicaciones como: descubrir segmentos de clientes dentro de un mercado para un producto. Al comprender qué atributos son comunes a determinados grupos de clientes, los vendedores pueden decidir en qué aspectos de sus campañas deben enfatizar*[14].

3.1.3. Datos de entrenamiento y datos de prueba

El conjunto de entrenamiento contiene los datos que, bajo observación previa, permite al algoritmo la experiencia que utilizará para aprender. En los problemas de aprendizaje supervisado, cada dato analizado consiste en una variable de respuesta observada y una o más variables explicativas de la muestra.

El conjunto de pruebas, es una colección similar de muestras que se utiliza para evaluar el rendimiento del modelo utilizando alguna métrica de rendimiento. Es importante no utilizar elementos de la muestra de entrenamiento, ya que esto, causará sesgos en la información, porque será difícil evaluar si el algoritmo ha aprendido a generalizar a partir del conjunto de entrenamiento o simplemente lo ha memorizado. Es decir, un programa que ha memorizado puede presentar alto rendimiento de clasificación con respecto a muestras ya observadas, pero tendrá bajo rendimiento para predecir y clasificar en la categoría correcta el valor de la variable de respuesta para nuevos ejemplos. Ya que podría memorizar relaciones y estructuras que son ruido o coincidencia. La memorización y generalización de equilibrio, o el ajuste excesivo (*Overfitting*) y el subconjunto, es un problema común a muchos algoritmos de aprendizaje de máquina. Por lo cual existe una variable de regularización, que puede aplicarse a muchos modelos para reducir el *Overfitting*[14].

3.2. Máquina de Vectores de Soporte (Support Vector Machine-SVM)

Es un tipo de máquina de aprendizaje, basado en un conjunto de métodos de aprendizaje supervisado comúnmente utilizados en problemas de clasificación, regresión y detección de valores atípicos.

3.2.1. Formulación Básica: SVM Lineal:

Las técnicas de aprendizaje supervisado para los problemas de clasificación, utilizan un conjunto muestra S, que se define como:

$$S, \{(x_n, y_n) \,|n = 1, 2..., N, x_n \in \Re^d y_n \in \{1, 2, .., C\}\}$$

donde N es el número de elementos del conjunto de datos, d el número de variables para definir los ejemplos x_n del conjunto y C el número de clases distintas. Los ejemplos representan las características del conjunto y se definen por el vector x_n, el cual tiene asociada una etiqueta y_n que corresponde a la clave de la característica. Los vectores x_n se conocen comúnmente como vectores de características.

Un algoritmo de aprendizaje suministra un tratamiento a los datos, a fin de encontrar una función $f : \Re^d \to \{1, 2, .., C\}$, capaz de predecir aproximadamente la clase de nuevos elementos con margen de error mínima, es decir, $\neq \{n \,|f(x_n) \neq y_n\} \simeq 0$. Los algoritmos de aprendizaje permiten ajustar y optimizar los parámetros de un clasificador de acuerdo a su función.

El entrenamiento de una máquina de vectores de soporte primal (SVM-Primal) para la clasificación binaria, dado un conjunto de entrenamiento $(x_1, y_1), ..., (x_n, y_n)$ con $x_i \in \Re^N$ y $y_n \in \{-1, +1\}$, requiere hallar una funcion que se encuentre dentro del espacio de hipótesis y que defina el hiperplano de separación entre clases, lo que significa resolver un problema de optimización[22].

Clasificador binario lineal: para este tipo de clasificador, existe una única frontera de separación representada por una línea en $\Re^2$ en un espacio bidimensional (*Figura 28*) y en un espacio multidmensional es un hiperplano $\Re^n$. Otras extensiones para este tipo de máquinas son la clasificación multiclase y los conjuntos no linealmente separables.

Existe otras maneras para trazar lineas solución (solución de hiperplanos), para el problema de dos clases:

•**Modelos generativos:** Que consisten en la construcción de lineas de separación entre clases mediante la estimación probabilística, usando métodos como: Naïve Bayes (función de densidad) o el método de *Fisher discriminant anaylisis(LDA) [8].*

•**Modelos discriminativos:** Consisten en la solución a partir de un conjunto de entrenamiento cuyas muestras sean suficientemente representativas de discriminar, en este modelo se encuentran la regresión logística, redes neuronales y Máquinas de Vectores de Soporte -*Support Vector Machine* (*SVM*).

Un hiperplano es la frontera de decisión de un *SVM*. A partir de la solución de un problema de optimización se obtiene el hiperplano de separación entre clases: que consiste en la distancia máxima entre los hiperplanos que contienen los vectores de soporte de ambas clases (margen máximo), Ver *Figura 24*.

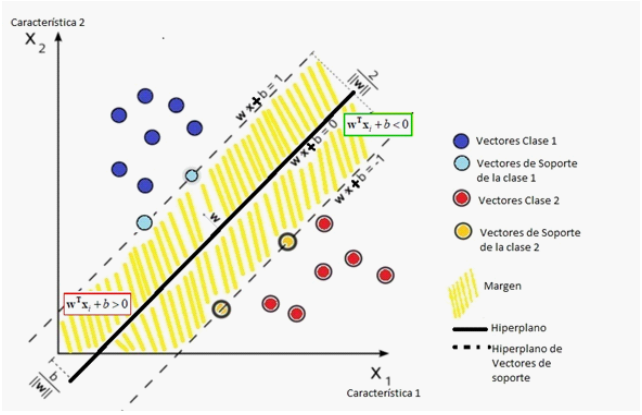

Figura 24: Frontera de decisión *SVM*(*Support Vector Machine*).

Las muestras de cada clase tienen una etiqueta:
Clase $1 : y1 = +1$
Clase $2 : y2 = -1$

El *Support Vector Machine* tiene una solución característica, que se basa en la generación de una frontera de separación a partir de un número limitado de muestras del conjunto de entrenamiento (vectores de soporte) con propiedades concretas y contenidas en los hiperplanos de separación. La separación o distancia máxima entre los vectores de soporte es el margen máximo.

$$w^T x_i + b = 0 \qquad \begin{matrix} \textit{Hiperplano solución } W \\ w^T = (w_{1,\ldots} w_n) \end{matrix} \qquad (3,2,1)$$

Para el caso de un clasificador *SVM lineal* (*Figura 24*), la solución se expresará con la ecuación de una línea recta (*ecuación* 3,2,1) donde w es el vector ortogonal al hiperplano y b el coeficiente de intersección, luego se tendrán entonces la separación de las muestras positivas y negativas que cumplirán las condiciones solución:

para las muestras positivas $w^T x_i + b > 0$
para las muestras negativas $w^T x_i + b < 0$

La Función del Clasificador lineal *SVM* se puede expresar a partir del signo (sgn), al aplicar el hiperplano a la muestra x (*ecuación* 3,2,2).

$$f(x) = sgn(w^T x + b) \qquad (3,2,2)$$

En el caso del *Support Vector Machine*, el hiperplano medio $w^T x_i + b = 0$; que es solución, está entre los dos hiperplanos de muestras positivas y negativas h^+ y h^-, respectivamente y que son paralelos entre sí, definidos por:

$h^+ \longrightarrow w^T x_i + b = +1$
$h^- \longrightarrow w^T x_i + b = -1$

La condición de clasificación que tienen que cumplir todas las muestras de entrenamiento, se expresa en la *ecuación* 3,2,3 :

$$y(w^T x_i + b) \geq 1 \qquad (3,2,3)$$

Así, el *margen* delimitado y representado por la región amarilla en la *Figura 24* se podrá calcular a partir de la distancia d^+, d^- entre los dos planos h (*ecuación 4.4*)

$$d^+ = d^- = \frac{|wx+b|}{\|w\|} = \frac{1}{\|w\|}$$
$$margen = d^+ = d^- = 2\frac{1}{\|w\|} \qquad (3,2,4)$$

De acuerdo a la *ecuación 3.2.4*, el margen depende únicamente de w. Para la clasificación se debe obtener el mayor margen, por lo tanto se debe maximizar la funcion *margen,* pero se debe tener en cuenta que al maximizar esta función equivaldrá a minimizar el problema inverso representado por la función Φ (*ecuación* 3,2,5), la cual es proporcional al producto escalar entre w^T por sí mismo, sujeto a la condición de clasificación (*ecuación* 3,2,3) y se resuelve tratando dicha solución del SVM como un problema de optimización cuadrática (*ecuación* 3,2,6).

$$\Phi(w) = \tfrac{1}{2} w^T w \qquad (3,2,5)$$
$$\|w\|^2 = w^T w$$

El objetivo es obtener w, b, pertenecientes al hiperplano solución $w^T x_i + b = 0$

$$Minimizar_{w,b}\ \Phi(w) = \tfrac{1}{2} \|w\|^2 \qquad (3,2,6)$$
$$\text{sujeto a: } y_i(w^T x_i + b) \geq 1$$

3.2.2. Desarrollo Matemático del *SVM*

Obtener w, b del hiperplano solución a partir de $w^T x_i + b = 0$.

Tratando la función como un problema de optimización cuadrática, se debe plantear la construccion de una función auxiliar conocida como el *Lagrangiano* L.

El *Lagrangiano* se construye a partir de la suma entre la función que se quiere optimizar y las restricciones a las que está sujeto el problema, multiplicadas por unos coeficientes α que son los multiplicadores de *Lagrange* (*ecuación* 3,2,7):

$$L(x, \alpha) = f(x) + \sum_i \alpha_i, g_i(x) \qquad \forall \alpha_i \geq 0 \qquad (3,2,7)$$

$$f(x) \longrightarrow \Phi(w)$$

$$g(x) \longrightarrow y_i(w^T x_i + b) - 1 \geq 0$$

El *Lagrangiano* construido a partir de las funciones $f(x)$ y $g(x)$ también se representa en la *ecuación* (3,2,8) :

$$L(w, b, a) = \tfrac{1}{2} w^T w - \sum_i \alpha [y_i(w^T x_i + b) - 1] \qquad (3,2,8)$$

Teniendo la transformación, se procede a minimizar inicialmente el *Lagrangiano* L con respecto a las variables $(w.b)$ de la función $f(x)$ y después maximizar con respecto a los multiplicadores de Lagrange y se obtiene la *ecuación* 3,2,9 :

$$max_\alpha(min_{w,b}(L(w, b, \alpha)) \qquad (3,2,9)$$

La minimización del *Lagrangiano* implica que las derivadas parciales con respecto a w y b sean iguales a 0, (ver *ecuación* 3,2,10).

$$\begin{aligned} \tfrac{\partial L}{\partial w} &= w - \sum_i \alpha_i y_i x_i = 0 \qquad \Longrightarrow \qquad w = \sum_i \alpha_i y_i x_i \\ \tfrac{\partial L}{\partial b} &= w - \sum_i \alpha_i y_i = 0 \qquad \Longrightarrow \qquad w = \sum_i \alpha_i y_i = 0 \qquad ; \alpha_i \geq 0 \end{aligned} \tag{3,2,10}$$

Reemplazando w de la *ecuación* 3,2,10 en la *ecuación* 3,2,8, se obtiene una nueva función $\Theta(\alpha)$

$$\begin{aligned} L(w,b,\alpha) &= \tfrac{1}{2}\left[\sum_i \alpha_i y_i x_i\right]^T \sum_j \alpha_j y_j x_j - \sum_i \alpha_i [y_i (\left[\sum_j \alpha_j y_j x_j\right]^T x_i + \sum_i \alpha_i \\ &= -\tfrac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_i x_i^T x_j + \sum_i \alpha_i \qquad ; \\ \Theta(\alpha) &= -\tfrac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_i x_i^T x_j + \sum_i \alpha_i \end{aligned} \tag{3,2,11}$$

Luego se procede a maximizar la función $\Theta(\alpha)$ y el valor máximo de esta función, está sujeto a la derivada parcial respecto a b, lo cual constituye un nuevo problema de optimización, llamado *Support Vector Machine Dual.* En la solución de este problema encontramos los vectores de soporte y en su formulación se puede observar (*ecuación* 3,2,12) que las muestras x aparecen únicamente como productos escalares tanto para la función de clasificación como para el problema de optimización, lo cual es una propiedad muy útil al trabajar con *Kernels* en los casos no linealmente separables.

$$\begin{aligned} &Maximizar\ \Theta(\alpha) = -\tfrac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_i x_i^T x_j + \sum_i \alpha_i \\ &Sujeto\ a: \sum_i \alpha_i y_i = 0 \qquad ; \alpha_i \geq 0 \\ &f(x) = sgn(w^T x + b) \\ &f(x) = sgn(\sum_i \alpha_i y_i x^T x_i + b) \end{aligned} \tag{3,2,12}$$

La solución de este problema de optimización permite encontrar la formulación del hiperplano solución, al calcular indirectamente los valores óptimos de los multiplicadores α y los vectores de soporte, a fin de definir la función de clasificación. Así, la solución para el hiperplano W va a ser la combinación lineal de aquellas muestras que únicamente tengan asociado un multiplicador α diferente de 0 definiendo así los vectores de soporte, ya que los multiplicadores de Lagrange calculados en el problema de optimización serán todos positivos o nulos.

Como se ha mencionado el *Support Vector Machine Dual,* representa un nuevo problema de optimización aplicable en los casos no linenalmente separables. Inicialmente se tiene que los vectores de soporte aparecen como productos escalares y se puede definir una función K *(Kernel, ecuación* 3,2,13*)* que los represente en el espacio

mapeado de la función φ. La función φ, es el nuevo espacio de características (*Figura 25*) sobre el cual se trabajan las muestras a fin de que las clases sean linealmente separables en este nuevo espacio.

$$\begin{aligned} &x \longrightarrow \varphi(x) \qquad \textit{Función de Mapeo} \\ &K(x,z) = \varphi(x)^T \varphi(z) \quad \textit{Función de Mapeo} \end{aligned} \tag{3,2,13}$$

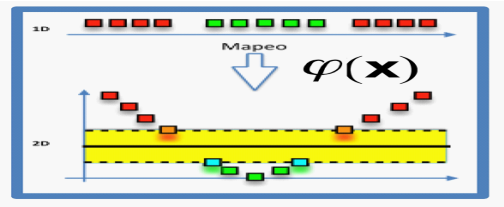

Figura 25: Espacio de muestras Mapeadas a una funcion φ. **Fuente:***Introducción a la clasificación de imágenes. Ernest Valveny* [8].

Existen diferentes ejemplos de funciones *kernel*, que de igual manera representan el producto escalar entre vectores y que se pueden integrar de forma directa en la solución del *Support Vector Machine*, así:

$$\begin{aligned} &Maximizar\ \Theta(\alpha) = -\tfrac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_i K(x_i, x_j) + \sum_j \alpha_i \\ &b = y_i - W^T \varphi(x_i) = y_i - \sum_j \alpha_i y_j K(x_i, x_j) \\ &f\,(x) = sgn(\sum_i \alpha_i y_i K(x_i, x_j) + b) \end{aligned} \tag{3,2,14}$$

- Algunos tipos de funciones *Kernel* son:

  a) *Kernel* Lineal: es una función muy eficiente ya que reduce el costo computacional en aprendizaje y clasificacón pero bajos resultados de clasificación para datos no linealmente separables. $K(x,z) = \langle x, z\rangle$

  b) *Kernels* No Lineales: Para estas funciones se eleva el coste computacional, pero mejoran los resultados de clasificación ya que no utiliza solo el producto escalar entre muestras sino que se debe calcular el valor del kernel de las muestras con cada uno de los vectores de soporte.

  i. Polinomial: $K(x,z) = \langle x, z\rangle^d$

  ii. *Kernel* gausiano de base radial (RBF): $K(x,z) = e^{-\frac{|x-z|^2}{2\sigma}}$

  iii. Sigmoide: $K(x,z) = \tanh(K\langle x, z\rangle - \delta)$

  iv. Multi-cuadrático inverso: $K(x,z) = \left(\|x-z\|^{\frac{1}{2}} 2\sigma + c^2\right)^{-1}$

  v. *Kernel* de Intersección: $K(x,z) = \sum_{i=1}^{n} \text{mín}(x\,(i)\,, z(i))$

Estos tipos de Funciones *Kernels* definen una función de similitud entre dos muestras e introducen parámetros σ, δ, c; que deben calcularse de forma óptima para cada función. Estos valores óptimos se encuentran durante el aprendizaje del clasificador mediante la validación cruzada.

Para conjuntos no linealmente separables, se habla del termino **margen suave**, que es, la *margen o limite* que permite ciertos valores de tolerancia durante la clasificación, ayudando a minimizar el error. El **margen suave**, se implementa con base en *variables de holgura* ξ, que condicionan bajo un valor definido el conjunto de vectores que violen la condición de margen, ver *Figura 26*.

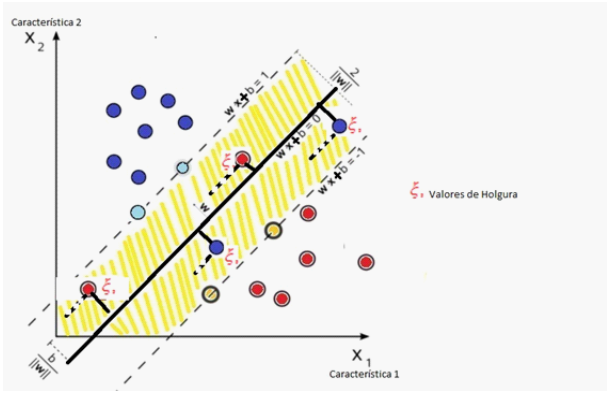

Figura 26: Margen Suave, Valores de Holgura.

Con el **margen suave**, el problema de optimización para un clasificador *SVM-Primal* tendra la función de minimización (*ecuación* 3,2,6) que se puede reescribir teniendo en cuenta los valores de holgura ξ. Ver *ecuación* 3,2,15.

$$\begin{aligned} &Minimizar_{w,\xi_i}\ \Phi(w) = \tfrac{1}{2}w^T w + C\sum_{i=1}^{n}\xi_i \\ &\text{Sujeto a}: \forall i \in \{1,...,n\} : \left(W^T\phi\left(x_i\right) + b\right) \geq 1 - \xi_i \end{aligned} \qquad (3,2,15)$$

El parámetro C que aparece al reescribir la función del clasificador es un parámetro de regularización y es fijado de manera estándar a partir de validación cruzada, que se detallara más adelante en la seccion de pruebas de clasificación. Este parámetro C se encarga de controlar la variación de maximización del margen y la minimización del error en las muestras de entrenamiento. Cuando este parámetro toma valores muy grandes da como resultado la anulación de la aproximación de margen suave.

Para la solución final del clasificador dual (*Support Vector Machine Dual*, *ecuación* 3,2,13), se puede reescribir la función, introduciendo las variables de holgura ξ_i, y estará sujeta a la definición de una nueva inecuación que modula el aporte de los vectores que tiene α entre 0 y C, ver *ecuación* 3,2,15.

$$Maximizar_{\alpha}\ \Theta(\alpha) = -\tfrac{1}{2}\sum_i\sum_j \alpha_i\alpha_j y_i y_i K(x_i, x_j) + \sum_i \alpha_i$$

$$Sujeto\ a: \begin{cases} 0 \leq \alpha \leq C \\ \sum_i \alpha_i y_i = 0 \end{cases} \qquad (3,2,15)$$

$$b = y_i\,(1 - \xi_i) - \sum_j \alpha_i y_j K(x_i, x_j)$$

$$f\,(x) = sgn(\sum_i \alpha_i y_i K(x_i, x_j) + b) \qquad (3,2,16)$$

Por la *ecuación* 3,2,16*,* se puede decir que el parámetro C y las variables de holgura ξ_i solo son tenidas en cuenta durante el proceso de optimización.

3.2.3. *Kernels* Especiales

Cuando la frontera de decisión se plantea para casos no linealmente separables, los desarrollos en máquinas de aprendizaje, proponen distintas funciones que proyectan las muestras a otros espacios de representación a fin de encontrar las relaciones lineales para el cálculo de la distancia entre muestras. *Entre estas funciones, está el kernel de Intersección de Histogramas, que nos permite calcular la similitud existente entre histogramas de dos imágenes*[8].

3.2.3.1. ***Kernel de Intersección de Histogramas***

Es una función de comparación entre Histogramas, mide las frecuencias en los histogramas, eligiendo la frecuencia mínima para cada valor del histograma, a fin de contar el número de correspondencias de una misma palabra visual, en diferentes regiones y niveles de resolución de las imágenes.

$$K_{\Delta}(X, Y) = I(H_X^{(i)}, H_Y^{(i)}) = I^l = \sum_{i=1}^{D} \text{mín}(H_X^l(i), H_Y^l(i)) \qquad (3,2,17)$$

A nivel de vocabulario se puede aplicar este *Kernel* como técnica muy eficiente. El *Kernel* de Comparación de Pirámides, se basa en una idea propuesta originalmente por Grauman y Darrell en el 2005[23]. El objetivo es calcular las correspondencias aproximadas, entre descriptores visuales de dos imágenes, en diferentes regiones y resoluciones, para poder estimar su similitud de forma eficiente. La técnica es concatenar histogramas de varios niveles, aumentando proporcionalmente las dimensiones del histograma en los niveles posteriores.

Posteriormente Svetlana Lazebnik, Cordelia Schmid y Jean Ponce en el 2006, adaptan esta idea de calcular la similitud de histogramas para aplicarla en histogramas de palabras visuales[8].

La aplicación de este *Kenerl* se realiza a la entrada del clasificador tanto para el entrenamiento como para la evaluación y como se ha mencionado los *Kernels* se utilizan para proyectar un conjunto de datos a un espacio euclidiano a fin de encontrar las relaciones lineales entre sus muestras. *Es decir, en el SVM básico, los Kernels utilizan como proyección el producto escalar para calcular la similitud de las muestras, sus posiciones relativas y determinar el hiperplano que los separa mejor* [8].

El tema de *Kernels* especiales, es muy innovador y ha demostrado gran rendimiento y eficiencia y puede profundizarse en otra oportunidad donde se aborde los temas de la representación de palabras visuales aplicando el concepto de pirámides espaciales. *Las pirámides espaciales consisten en la colocación de una secuencia de rejillas cada vez más finas en una imagen para representarla mediante histogramas de palabras visuales de diferentes niveles y a diferentes regiones de la imagen [8],* como lo exponen los autores Grauman y Darrell, ICCV 2005 y Svetlana Lazebnik, Cordelia Schmid y Jean Ponce en el 2006 [23][24].

- Las ventajas de este tipo de máquinas son [16]:

  i. Eficaz en espacios de alta dimensión.
  ii. Sigue siendo efectivo en casos en los que el número de dimensiones es mayor que el número de muestras.
  iii. Utiliza un subconjunto de puntos de entrenamiento en la función de decisión (llamados vectores de soporte), por lo que también es eficiente en la memoria.
  iv. Versátil: se pueden especificar diferentes funciones del *kernel* para la función de decisión. Los *kernels* comunes se proporcionan, pero también es posible especificar *kernels* personalizados.

- Las desventajas de las máquinas de vectores son:

  i. Si el número de características es mucho mayor que el número de muestras, es probable que el método dé un rendimiento pobre.
  ii. Las *SVM* no proporcionan estimaciones de probabilidad directamente, éstas se calculan usando una costosa validación.

Support Vector Machine, aunque inicialmente su formulación no es parametrizada, en las aplicaciones reales se tiene en cuenta el margen suave, lo cual implica la regularización a través del parametro C y además el uso de funciones *Kernels*, las cuales adicionan parámetros naturales y propios de cada función.

3.3. Árboles de Decisión (Métodos de Ensamble)

Un árbol de decisión es un procedimiento determinado por una secuencia ordenada de preguntas y bifurcaciones donde la respuesta a la pregunta conlleva a la subpregunta y así sucesivamente hasta la pregunta actual. Dichas cuestiones son formuladas sobre las variables del vector x_n, a fin de asignarle una etiqueta de la correspondiente clase, ver *Figura 27.*

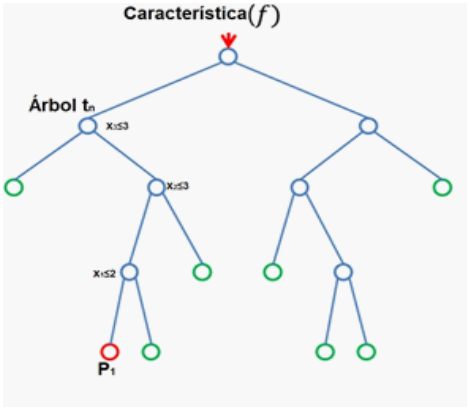

Figura 27: Ejemplo de árbol de decisión. **Fuente**:https://www.analyticsvidhya.com/blog/2015/06/tuning-random-forest-model/

La forma natural de representar el procedimiento, es mediante un árbol (*Figura 27),* donde el primer nodo se conoce como nodo raíz y es donde se ubica la característica f, para luego conectarse sucesivamente con el resto de nodos hasta llegar a los nodos hoja, los cuales no tienen descendientes. En los nodos hojas es asignada la probabilidad P_1 de pertenecer a la clase de la etiqueta y. Además, se puede observar en la figura que las preguntas en cada nodo pueden ser interpretadas por reglas del tipo *si* $[x \leq 3$ y $x \leq 2]$, entonces sucede algo, $x \in y_1$.

3.3.1. Árboles CART

El termino divide y vencerás hace referencia a un paradigma de diseño algorítmico muy importante y es el principio base de los árboles de decisión[25].

El procedimiento de construcción de un árbol *CART* [*Classication and Regression Trees*, Breiman.1984], inicia con el conjunto total de datos y se van definiendo los cortes posibles, dividiendo el conjunto inicial en dos subconjuntos. Por tanto, los tres problemas a resolver son[26]:

1. Seleccionar la variable y el valor de corte en cada nodo.
2. Asignar una clase a cada nodo hoja.
3. Decidir el criterio para definir un nodo como hoja o continuar construyendo el árbol.

3.3.1.1. Seleccionar la variable y el valor de corte

La selección del corte en un árbol de decisión tiene como objetivo principal, buscar la pureza de los nodos con respecto a su nodo inicial o padre, dado un nodo t, su descendiente t_L derecho e izquierdo t_R, deben ser lo más puros posible. Es decir, que las clases subsiguientes (descendientes) siempre estén mejor separadas que en los progenitores. *CART esta centrado en la construcción de árboles binarios (cada nodo tiene o dos hijos o ninguno) debido a su simplicidad y a que todo árbol con más ramas en alguno de sus nodos puede ser reducido a un árbol binario aumentando su profundidad* [26].

Para esta selección de corte, el criterio de impureza para cada nodo $i(t)$, *debe ser máxima cuando varias clases se presentan en t (nodo menos puro posible) y ser* 0 *cuando solo haya elementos de una de las clases (nodo más puro posible). Formalmente, una función de impureza es una función* ϕ *definida sobre* (p_1,p_2) *tal que* $p_1,p_2 \geq 0$ *y* $p_1 + p_2 = 1$[26].

Dada una función de impureza ϕ se puede definir la impuerza en un nodo así:

$$i(t) = \phi(p(1\,|t)\,, p(2\,|t)) \qquad (3,3,1)$$

donde $p(1\,|t)$ es la probabilidad de pertenecer a la clase y dado el nodo t.

La función de impureza más utilizada por el algoritmo *CART* es la impureza *GINI*, la cual se definne como:

$$i(t) = \phi(p(1\,|t) \cdot p(2\,|t)) = (p(1\,|t) \cdot (1 - (p(1\,|t)) \qquad (3,3,2)$$

Existen también otras funciones de impureza como la función de *entropía*:

$$i(t) = -(p(1\,|t) \cdot \log p(1\,|t) + p(2\,|t) \cdot \log p(2\,|t)) \qquad (3,3,3)$$

Se ha observado que la elección de algunas de las funciones de impureza no es muy determinante en cuanto a la capacidad de generalización del árbol final [Classification and Regression Trees, Breiman et al., 1984]. [26]

Luego de que la función de impureza es definida, se procede a encontrar la función de corte óptimo para que maximice la diferencia entre la impureza del nodo raíz y la de sus descendientes en proporción a cada uno de ellos. Para obtener el valor óptimo, se debe buscar exhaustivamente en todos los cortes posibles de cada una de las variables.

$$\Delta i(t) = i(t) - p_L \cdot i(t_L) - p_R \cdot i(t_R) \qquad (3,3,4)$$

donde $i(t_L)$ y $i(t_R)$ son las funciones de impureza de los hijos, p_Ly p_R la proporción de datos de t.

3.3.1.2. Asignación de clase a los nodos hoja.

La asignación de etiqueta se realiza despues de que se ha decidido la correspondencia del nodo actual con la clase asignada. Si el árbol ha logrado asignar una sola clase para todos los elementos (impureza 0), la clase será asignada al nodo correspondiente. La asignación de pureza o impureza de los nodos despues de ser cortados (*pruning*), definen la etiqueta del elemento a la clase con más afinacion o concordancia. Esto es:

$$t \rightarrow etiqueta\ y \Longleftrightarrow y = \arg \text{máx}_y\ p(y\,|t) \qquad (3{,}3{,}5)$$

3.3.1.3. Criterios de parada. Poda.

El entrenamiento de un clasificador por el método de árboles de decisión puede presentar sobre entrenamiento(*Overfitting*), como en el caso de las máquinas de vectores de soporte que se mencionaron en secciones anteriores, suele suceder en este caso, porque durante la construcción del árbol se busca que todos los nodos tengan impureza 0, una estimación que resulta ineficiente ya que se pierde la capacidad de generalización de las clases, y aunque las muestras del conjunto de entrenamiento queden bien clasificadas, los errores suelen ser elevados durante la clasificación del conjunto de prueba(*test*). Esta situación puede ser resuelta utilizando, criterios de parada (prepoda) o podando el árbol (poda).

Al utilizar criterios de parada, lo que se busca es, introducir una regla que permita cierto umbral dentro del cual se permitirá que las muestras puedan pertenecer a una clase de acuerdo al ajuste fijado para dicho umbral. Es decir, se tendrá un umbral que propone una margen para que las muestras que superen el umbral pertenezcan a una clase, si el umbral permite muchas aproximaciones de las muestras lo que sucederá es que aumente el número de falsos positivos y como lo sugiere Breiman et al., 1984, el resultado no será del todo satisfactorio. *De igual forma se sugiere que para construir un árbol con la capacidad de generalización, se utilice el procedimiento de poda*[26]. Para este procedimiento de poda primeramente se establece un estimador del error local $R(t) = r(t) \cdot p(t)$, para un nodo t, definido como la probabilidad de que una de las muestras del conjunto sea mal clasificada $r(t) = \text{mín}\{p(1\,|t)\,, p(2\,|t)\}$.

Al obtener el error $R(t)$, el cual puede aumentar o disminuir durante el entrenamiento, se busca hasta alcanzar un mínimo tras la poda del árbol, logrando un punto de equilibrio donde todos los datos tengan el mismo error, ya que se tienen un mejor estimador del error esperado y de esta manera se admita la creación de un árbol $T\,\text{máx}$ donde los nodos terminales contengan muestras de una sola clase. *Finalmente construido, CART utiliza un criterio de poda denominado criterio de* ***costo-complejidad****, el cual selecciona un sub-árbol de entre todos los contenidos en* $T\,\text{máx}$, el sub-árbol T tendra la notación $T \preceq T\,\text{máx}$ medido por el criterio ***costo-complejidad*** definido como: $R_\alpha(T) = R(T) + \alpha \left|\tilde{T}\right|$, donde α es el parámetro de complejidad $\alpha \geq 0$ y $\left|\tilde{T}\right|$ el número de nodos terminales de T.

Luego se debe minimizar el criterio planteado y por lo tanto se deben hallar los valores de α que logren este objetivo, de esta manera se formará una secuencia finita de sub-árboles con los distintos valores de α donde el árbol minimizador cambia, así:

$$T\,\text{máx} = T(0) \succeq T(\alpha_1) \succeq T(\alpha_2)... \qquad (3,3,6)$$
$$T(\alpha_k) = raiz(T)$$

Para obtener la secuencia finita mencionada, se realiza un procedimiento que no se describe en profundidad en esta sección del libro, pero a nivel general se resume en 3 pasos:

1. La obtención del término $T(\alpha_1)$, el cual es el árbol de partida y el más pequeño, con error global igual al de $T\,\text{máx}$.

2. Obtener α_2 y el sub-árbol $T(\alpha_2)$ asociado. Aquí se define dos medidas de *costo-complejidad* las cuales van eliminando las ramas más débiles, de forma que el nodo t_1 será el que minimice la *expresión* 3,3,8.

$$R(t) + \alpha = R(T_t) + \alpha \left|\tilde{T}\right| \Longrightarrow \alpha = \frac{R(t)-R(T_t)}{\left|\tilde{T}\right|-1} \qquad (3,3,7)$$

   ⋇ *Luego el nodo donde la igualdad se satisface con un alfa más pequeño, es aquel en el que la diferencia entre el error del nodo t_1 y del árbol T_{t_1} es más pequeña (ponderada con la complejidad)*, ver *expresion* 3,3,9 *[26]*

$$T(\alpha_2) = T(\alpha_1) - T_{t_1}, \text{ donde } \alpha_2 = \text{mín}_{t_1 \in T(\alpha_1)} \frac{R(t)-R(T_t)}{\left|\tilde{T}\right|-1} \qquad (3,3,8)$$

3. Repetir el paso 2 hasta que el valor de α sea α_k con $T(\alpha_k) = raiz(T)$.

3.3.2. Bosques Aleatorios (*Random Forest-RF*)

Al inicio de este capítulo, se ha mencionado que las tareas de clasificación pueden ser abordadas con diferentes modelos de clasificación, pero lo que se busca es encontrar un modelo que arroje mejores precisiones durante la clasificación y se reduzca la varianza de los errores presentes en el entrenamiento, por lo tanto, se proponen las técnicas de conjuntos de clasificadores (ensamble). El procedimiento de esta técnica puede observarse en el *cuadro 4*.

El objetivo de los métodos de ensamble, es construir clasificadores dados por la combinación de las predicciones de varios estimadores base, con el fin de mejorar la generalización y robustez sobre un único estimador[27]. Se pueden mencionar dos familias de métodos de ensamble, los métodos promediados (Métodos *Bagging*, Bosques aleatorios-*Random Forest*) "*Con ello se consiguen las probabilidades estimadas conjuntas y se realiza la clasificación mediante la técnica del punto de corte óptimo de la probabilidad estimada*"[28] y los métodos *boosting* (*AdaBoost*, *Gradient Tree Boosting*).

Cuadro 4: Procedimiento de los Métodos de Ensamble[5]

Procedimiento General para Métodos de Ensamble.
1. Entrada: Dado un conjunto de datos de entrenamiento D, k el número de clasificadores base y T el conjunto de prueba.
2. for $i = 1$ hasta k do
a. Crear un subconjunto de entrenamiento D_i del conjunto original D
b. Construir un clasificador Base C_i de D
3. end for
4. for cada registro de prueba $x \in T$ do
1. clasificar por "voto mayoritario", $C^*(x) = Voto(C_1(x), C_2(x), ..., C_k(x))$
5.end for

Los conjuntos de clasificadores pueden construirse a partir de la respuesta dada en un nivel de cuestionamientos. En la literatura Rokach[4], propone una taxonomía de cinco dimensiones (1.Combinador, 2.Construcción del Ensamble, 3.Diversidad, 4.Tamaño del Ensamble y 5.Universalidad), para la construcción del clasificador, se puede consultar el libro Combining Pattern Classifiers[4], donde se encuentra representada graficamente y de forma simplificada. Resaltando la dimension de Universalidad, se puede decir con respecto a esta dimensión, que el enfoque de los clasificadores base, en algunos modelos pueden ser usados con cualquier modelo clasificador, mientras que otros están vinculados con un tipo de clasificador especifico, un ejemplo de clasificador específico es el bosque aleatorio (*Random Forest*), cuyo sello es, su clasificador base, el árbol aleatorio (*Random Tree*)[4].

Un *Random Forest* es un algoritmo de predicción basado en la familia de árboles de decisión, utiliza la técnica de *Bagging* para combinar diferentes árboles de tal manera que cada árbol depende de los valores de un vector aleatorio muestreado independientemente y con la misma distribución para cada uno de los árboles[25], fue propuesto por Leo Breiman in the 2000's[29]. El algoritmo de *Random Forest-RF* consta de parámetros sencillos de ajustar, se adecua a muchas aplicaciones reales ya que a menudo es capaz de lograr el mejor rendimiento de la clase con respecto al error de generalización[30]. De esta forma, *Dado un conjunto de entrenamiento etiquetado* (X, Y)*, un RF es una colección de árboles clasificadores* $\mathcal{F} = \{h_k(x), k = 1, ..., k\}$*, y probablemente la mejor forma de entenderlo, es dibujar la forma en que se construyen habitualmente los árboles de decisión[31].*

3.3.2.1. Formulación Básica del Random Forest:

En 2001, Breiman propuso una variante del Bagging que él llama un bosque al azar (Random Forest-RF). El bosque aleatorio es una clase general de métodos de ensamble utilizando un árbol de decisión como el clasificador base[4]*. Su autor Leo Breiman define un Random Forest como un clasificador que consiste en una colección*

de clasificadores de árboles estructurados $\{h(x, \Theta_k), k = 1, ..., L\}$ *donde los* $\{\Theta_k\}$ *son vectores aleatorios distribuidos de forma idéntica y cada árbol arroja un voto unitario para la clase más popular en la entrada* x [25]. La definición indica que la construcción de un *Random Forest* (*Figura 28*), se puede realizar por muestreo aleatorio del conjunto de datos o del conjunto de caracteristicas y es una dimension conocida como diversidad que tienen los modelos ensamblados.

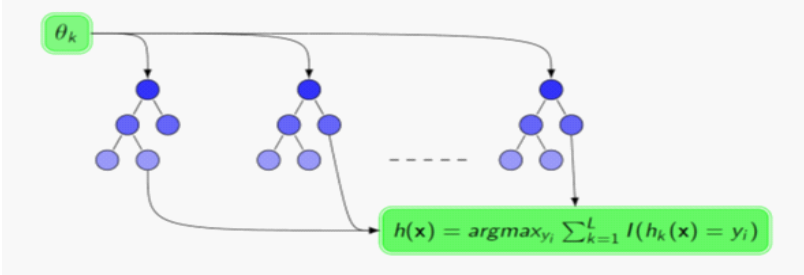

Figura 28: Random Forest-RI. **Fuente**:Simon Bernard Document and Learning research team LITIS laboratory University of Rouen, France d_ecembre 2014.[9]

Además, *Random Forest* es una técnica mejorada de *Bagging, que mejora la precisión en la clasificación mediante la incorporación de aleatoriedad en la construcción de cada clasificador individual. El algoritmo Random Forest, a diferencia de Bagging introduce de forma aleatoria en cada nodo* ρ *variables de todas las originales, y de estas selecciona la mejor para realizar la partición[28].* Es decir, que la selección aleatoria de características se realiza en el nodo de cada árbol, de esta selección y división se generará un subconjunto S con M características en cada nodo(cultivo del árbol). *El árbol se cultiva utilizando la metodología CART [Breiman et al. Classification and Regression Trees.1984]hasta el tamaño máximo, sin poda*[29]. *Breiman sugiere que crezca un árbol CART completo (sin poda). El valor recomendado de M* es $[\log_2 n + 1]$[4].

- En forma resumida, el *Random Forest* puede seguir un procedimiento de 4 pasos [32]:

1. Selecciona aleatoriamente elementos de un conjunto de datos para crear un conjunto de entrenamiento (*Bootstrap*-muestras de tamaño n). Con los elementos restantes de entrenamiento se forma el conjunto de validación o *Out Of Bag data* (*OOB data*).

2. Crea un árbol de decisión con cada uno de los elementos y divide el árbol o nodo sobre un subconjunto, m.

   a) Selecciona aleatoriamente d características sin reemplazo, en busca de la mejor variable.

b) Se implementa una función de impureza (*entropía* o *Gini*), para buscar la mejor división de los datos de las variables seleccionadas en el subconjunto m.

3. Los pasos se repiten k veces hasta obtener los diferentes conjuntos de árboles entrenados.

4. Evalúa y predice los nuevos datos, clasificando por el "voto mayoritario", la clase a la que pertenece, y para regresión promediando el valor de los resultados.

Continuando con la revisión de algoritmo de *Random Forest,* este a su vez, presenta la selección al azar de los subconjuntos desde el conjunto de entrenamiento completo. Una forma en la que se selecciona el subconjunto, es filtrar aleatoriamente las filas con reemplazo de la misma manera que el algoritmo de agregación de *bootstrap* de Brieman[15]. El algorimto de un *Random Forest* se puede observar en el *cuadro 5*, tomado de Trevor Hastie (2009).

Cuadro 5: Algoritmo Random Forest Para Regresión y Clasificación[6].

Algoritmo Random Forest Para Regresión y Clasificación, Hastie(2009).
1. Entrada: Conjunto $S, \{(x_n, y_n) \big\vert n = 1, ..., N, x_n \in \Re^d y_n \in \{1, .., C\}\}$ k el número de variables en cada nodo, T el número de árboles. 2. for $t = 1$ hasta T do a. Dibuja un conjunto de entrenamiento ($MuestreoBootstrap$-muestras Z^* de tamaño N) del conjunto original S b. Cultivar un árbol de decisión B_t repitiendo de forma recursiva los siguientes pasos para cada nodo terminal del árbol, hasta que se alcance el tamaño de nodo mínimo n mín. i. Selecciona aleatoriamente x características de k variables. ii. Elige la mejor división de los datos de las variables seleccionadas en el subconjunto m iii. Divide el nodo en dos hijos nodo. 3.Salida: ensamble de árboles $\{B_t\}_1^T$
Regresión: $\hat{f}_{rf}^T(x) = \frac{1}{2}\sum_{t=1}^T B_t(x)$.
Clasificación: Sea $\hat{C}_t(x)$ la predicción de clase del t-árbol deL Bosque Aleatorio. Entonces $\hat{C}_{rf}^T(x) =$ voto mayoritario $\{\hat{C}_t(x)\}_1^T$

La heurística aleatoria del bosque tiene la intención de diversificar el conjunto. Los enfoques alternativos incluyen árboles difusos y pesos de características aleatorias. Una cierta exactitud del árbol de decisión se puede sacrificar, pero se paga con la diversidad creciente[4].Los estudios empíricos han demostrado que *Random Forest* supera significativamente el *Tree Bagging* y otros métodos de ensamble en términos de precisión. En términos de grado de aleatorización, este algoritmo es más fuerte

que *Tree Bagging*, especialmente si K es pequeño en comparación con el número de atributos, n [33]. En la literatura Brieman expone dos métodos similares para la implementación de *Random Forest, Forest-RI* y *Forest-RC*.

3.3.2.2. Formulación Matemática del Random Forest:

La colección de árboles definidas para *Random Forest* es $\{r_n(x_n, \Theta_m, D_n)m \geq 1\}$ donde Θ_1, Θ_2*...son salidas independientes e idénticamente distribuidas de la variable aleatoria* Θ. *Estos árboles aleatorios se combinan para formar la estimación de regresión agregada*[29].

$$\bar{r}_n(X, D_n) = E_\Theta[r_n(x_n, \Theta_m, D_n)],$$

Donde E indica la expectativa con respecto al parámetro aleatorio, condicionado en X y el conjunto de datos D_n. *La variable aleatoria* Θ *se utiliza para determinar cómo se realizan los cortes sucesivos al construir los árboles individuales, como la selección de la coordenada a dividir y la posición de la división*[29].

Para la caracterización de la medida exactitud, se detallan tres conceptos, la convergencia, la fuerza y correlación del *Random Forest*. Estos terminos son explicados profundamente, con un amplio análisis matemático por Breiman (2001) y otros autores en varios artículos y libros, por tal razón a manera de resumen se consignan las expresiones para los términos mencionados y una breve descripción de su contenido.

Dado un conjunto de calsificadores $h_1(x), h_2(x), ..., h_K(x)$ y con el conjunto de entrenamiento seleccionado aleatoriamente de la distribución del vector aleatorio Y, X, definen la función de margen como

$$mg(X, Y) = av_k I(h_k(X) = Y) - \text{máx}_{j \neq Y} = av_k I(h_k(X) = j),$$

Donde $I(\cdot)$ es la función del indicador. Cuanto mayor sea el margen, mayor será la confianza en la clasificación. El error de generalización está dado por:

$$PE^* = P_{X,Y}(mgP(X, Y) < 0),$$

Para un gran número de árboles, se deriva de la Ley Fuerte de Grandes Números, y como el número de árboles incrementa, para casi todas las secuencias Θ, existe un teorema que expresa que PE^* converge a

$$P_{X,Y}(P_\Theta(h(X, \Theta) = Y) - \text{máx}_{j \neq Y} P_\Theta(h(X, \Theta) = Y) < 0) \ ,$$

para la comprobación de este teorema se recomienda consultar la explicación dada en el *apéndice 1* del artículo escrito por Leo Breiman(2001), *este resultado explica por qué los bosques aleatorios no se sobreponen a medida que se agregan más árboles, pero producen un valor limitante del error de generalización.*[25].

La formulación matemática de *Random Forest*, tiene cualidades a nivel de conjunto indicadas como Fuerza y Correlación y en ellas interviene la función de margen expresada anteriormente $mg(X, Y)$. La fuerza del conjunto de clasificadores se define como: $s = E_{X,Y} mg(X, Y)$,y para el caso de dos clases, se tendrá:

$$s = E_{X,Y}[E_{\Theta}[c(X,\Theta) = Y) - E_{\Theta}[c(X,\Theta) \neq Y]],$$

y la correlación entre los árboles dentro del conjunto se mide de la siguiente forma:

$$\bar{\rho} = E_{\Theta}E_{\Theta'}[\rho(c(\cdot,\Theta), c(\cdot,\Theta^{'}))]$$

para estimar la correlación a partir del conjunto de evaluación, es conveniente utilizar la expresión en términos de la varianza y el margen, utilizando la siguiente expresión:

$$\bar{\rho} = \frac{var_{X,Y}mg(X,Y)}{[E_{\Theta}[sd(\Theta)])^2}$$

Luego de obtener las estimaciones de fuerza y correlación para un determinado *Random forest*, se puede definir el ratio $c/s2$ [25], igual a la correlación dividida por el cuadrado de la fuerza:

$$c/s2 = \frac{\bar{\rho}}{s^2}$$

Se pueden mencionar algunas ventajas y desventajas para el *Random Forest,* que han quedado como resultado de la experiencia, al utilizar esta técnica de aprendizaje automático.

- Pros:
  - Buen funcionamiento en grandes conjuntos de datos.
  - Puede ser utilizado para la extracción de variables importantes.
  - No requiere ingeniería de funciones (escalado y normalización)
  - Es uno de los modelos de decisión más precisos en aprendizaje de máquina.
  - Buenos resultados de exactitud.

- Contras:
  - Baja respuesta frente a datos ruidosos generando Overfitting.
  - Los resultados son más complejos de interpretar.
  - Necesita una buena afinación de Hyperparametros para una mejorar la precisión.
  - En cuanto a velocidad, requiere mayor tiempo de ejecución.

3.4. Métodos de Evaluación

Los métodos de evaluación, son utilizados para determinar si la solución a un problema de clasificación es correcta o la más adecuada, para ello se utilizan índices de calidad calculados a partir de métodos y procesos de rendimiento del clasificador, verificando durante cada prueba los niveles de mejoría y la presencia de errores. En la evaluación del rendimiento del clasificador se cuenta con una muestra de datos de entrenamiento y otra de prueba, durante esta evaluación existe una serie de medidas que indican si la solución en respuesta a los datos es correcta. De esta manera encontrar los parámetros más adecuados para el clasificador con mejores resultados. Las medidas conocidas son, el *Accuracy* o *exactitud*, la *AP* o *Average Precision*, la *Sensibilidad* (*sensitivity – recall*), *Curva precision-recall* y la *Precisión media*.

La matriz de confusión permite analizar el resultado del clasificador para cada una de las categorías etiquetadas en el *Ground truth,* contabilizando aciertos y errores del resultado de la clasificación. Así, en una columna se tendrá los positivos reales, que son aquellas imágenes etiquetadas como reales y también aquellas imágenes etiquetadas como negativas por el clasificador (falsos positivos), en otra columna se encontrara los falsos negativos y los negativos reales dados por el clasificador, ver *cuadro 6*.

Cuadro 6: Matriz de Confusión para problema de dos clases.

		Resultado del Clasificador	
		Clase 1	Clase 2
Resultado correcto	Clase 1	Positivos Reales	Falsos Negativos
	Clase 2	Falsos Positivos	Negativos Reales

Exactitud (Accuracy): Es la medida que indica la proximidad entre el resultado del clasificador y la clasificación exacta.

$$Exactitud = \frac{PositivosReales + NegativosReales}{\text{Pr}ediccionesTotales} \qquad (3,2,17)$$

Precisión: Es la medida que indica la calidad de la respuesta en que fueron clasificadas las muestras que son reales.

$$Presición = \frac{PositivosReales}{PositivosReales + FalsosPositivos} \qquad (3,2,18)$$

Sensibilidad (sensitivity – recall): Es la medida que indica la eficiencia en la clasificación de todos los elementos de una misma clase.

$$Recall = \frac{PositivosReales}{PositivosReales + FalsosNegativos} \qquad (3,2,19)$$

Curva Precisión-recall: Esta curva se obtiene al graficar los datos de Precisión vs Recall, con el fin de obtener la precisión media AP (*Average Presicion*) (*Figura 29*).

$$AP = \sum_{k=1}^{n} precisión(k)\Delta recall(k) \qquad (4,20)$$

k: todos los posibles umbrales de decisión
$presición(k)$: precisión para el umbral de decisión k.
$\Delta recall(k)$: incremento de recall al pasar del umbral $k-1$ al umbral k

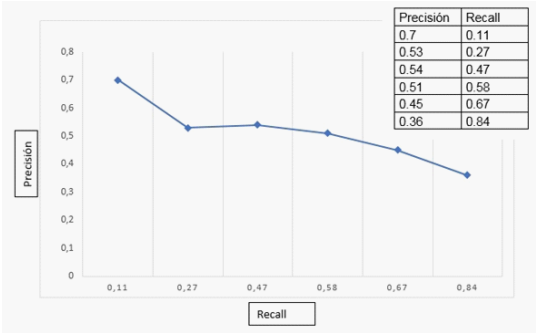

Figura 29: Curva de Precisión-Recall. **Fuente**: Autor.

Durante la evaluación se requiere tener un conjunto bastante significativo de muestras para el entrenamiento, podría utilizarse un porcentaje del 70 % para la etapa de entrenamiento y un 30 % para el conjunto de prueba. El conjunto de prueba representara las muestras que se requieren para medir el rendimiento del calificador y de esta manera conocer el aprendizaje del modelo y proporcionar una estimación de su desempeño en el mundo real.

El *Support Vector Machine*, aunque inicialmente su formulación no es parametrizada, en las aplicaciones reales se tiene en cuenta el **margen suave**, lo cual implica la regularización a través del parámetro C y además, el uso de funciones *Kernels*, las cuales adiciona parámetros naturales y propios de cada función. La técnica consiste en probar todas las posibilidades de los parámetros del clasificador con **validación cruzada** y elegir la mejor combinación, es una técnica muy útil para medir el rendimiento. La **validación cruzada**, Consiste la partición del conjunto de entrenamiento en muestras de aprendizaje y evaluación. El clasificador *Support Vector Machine*, tiene un aspecto importante por resaltar y es el umbral de decisión durante la clasificación, ya que por defecto clasifica las muestras en función de si el resultado es positivo o negativo. Es decir, que inicialmente el umbral se sitúa en cero, y el signo asignado a cada muestra determina la clase o umbral al que pertenece. Sin embargo, el umbral se puede modificar, de manera que la posición de la frontera de decisión se modifique.

Para estimar la medida de error del *Random Forest*, se utiliza una caracteristica muy importante y es el análisis de las muestras *Out-Of-Bag* (OOB), esta característica se usa para determinar la impureza en los nodos terminales. *"Para cada observación $z_i = (x_i, y_i)$, elaborar su predictor Random Forestes predictor, promediando sólo aquellos árboles que corresponden a muestras de bootstrap en las que z_i no apareció. Una estimación del error oob es casi idéntica a la obtenida mediante la validación cruzada"*[6]

Capítulo IV

DESARROLLO DEL PROYECTO

4.1. Aspectos prácticos de la prueba de Ronchi

La referencia experimental para obtener los Ronchigramas es la prueba de Ronchi y estos se forman al exponer una luz controlada, que atraviesa una rendija hacia el espejo y por el fenómeno de reflexión se obtiene la imagen sobre la rejilla, que luego es capturada a través de una cámara. La imagen digitalizada, es almacenada en el computador con el fin de crear un conjunto de entrenamiento que contiene las diferentes muestras que representan tanto un espejo sin defecto, como aquellos espejos que presenten los diferentes tipos de errores o defectos en la superficie, después del pulido del mismo.

Existen diferentes configuraciones y modelos para hacer la Prueba de Ronchi, pero el fin es el mismo, encontrar patrones de interferencia como los que se aprecian en la *Figura 7*, estos patrones son formados al colocar una rejilla dentro o fuera del centro de curvatura de una superficie óptica (espejo); como se ha explicado, estos patrones se relacionan con las aberraciones ópticas en la superficie del espejo. Cuando el sistema óptico es puesto bajo estudio, utilizando el haz de luz que pasa a través de una rendija, se obtiene una imagen de la rejilla, la cual se traslapa con la rejilla original produciendo así un patrón conocido como Ronchigrama. El Ronchigrama es el patrón que se analiza para encontrar y determinar el estado del espejo.

Recientemente, se trabajan diferentes métodos para la parte de procesamiento de Ronchigramas[34], las líneas futuras buscan la implantación de algoritmos de aprendizaje con el fin de evaluar y encontrar patrones que se obtienen durante la prueba de Ronchi[35] Un artículo escrito en el 2012 se titula: *"Probador de Ronchi con precisión de* 1λ *en las aberraciones del frente de onda, para sistemas ópticos convergentes usando la interferométria de desplazamiento de fase, donde se describe el diseño y la implementación de un dispositivo para verificar cuantitativamente la calidad de cualquier sistema óptico convergente, además como fuente de iluminación utilizan un conjunto de LEDs que ayudan a reducir variaciones en la irradiación captada por la cámara, este es una mejora muy relevante cuando se trabaja en las aberraciones de frente de onda. La técnica PSI fue implementada dando movimientos a la rejilla de Ronchi con un motor a pasos"*[36].

En los laboratorios de óptica se llevan a cabo diferentes procesos, en algunos se fabrican espejos que deben cumplir con características específicas por parte de un cliente, al cual debe garantizarse, que la elaboración del espejo, se está realizando bajo óptimos procesos de calidad, es decir, bajo sistemas automatizados y de altos niveles de tecnología. La prueba más utilizada es la prueba de Ronchi, que analiza las

características de espejos y lentes arrojando los datos que pueden indicar el defecto en un espejo. Si el proceso es automatizado, se puede obtener un resultado más exacto de la prueba.

La Universidad de los Llanos cuenta con un taller de óptica, único en el país a nivel universitario, en el cual se producen componentes ópticos en vidrio, desde una lupa simple hasta el espejo objetivo de un telescopio astronómico reflector. Además, realiza la prueba de Ronchi y para realizar esta prueba se utiliza el sistema asistido por computador de la prueba de Ronchi. En la *Figura 30* se muestra el sistema utilizado en el taller de óptica en la Universidad de los Llanos. Compuesto por: 1. Riel de 1 metro, 2. Fuente de poder $12V$. 3. Motores paso a paso NEMA 17, 4. Circuito puesto sobre la Plataforma móvil movimiento largo, 5. Fuente de luz blanca, laser portátil 6. Cámara, Rejilla y rendija ubicadas en la plataforma de movimiento corto, 7. Espejo y soporte.

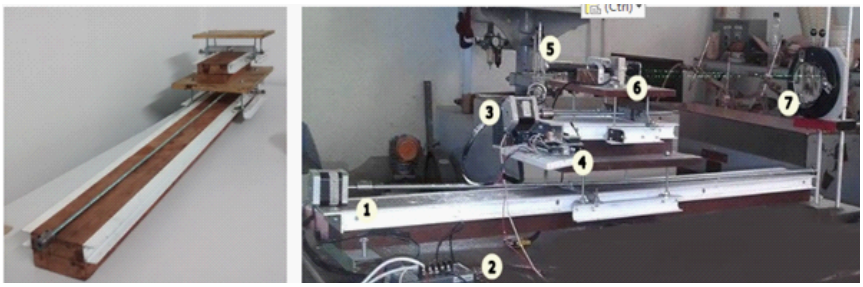

Figura 30: Sistema SAPRULL del Laboratorio de Óptica de la Universidad de los Llanos. **Fuente:** *Diseño e implementación de un sistema de adquisición y análisis de información de Ronchigramas, mediante tratamiento digital de imágenes* [2].

Como se observa en la *Figura 30*, el arreglo practico distingue tres elementos primordiales, la fuente de luz blanca, rejilla periódica (*4 líneas /mm*) y la cámara digital integrada por el sensor semiconductor complementario de óxido metálico (en inglés *complementary metal-oxide-semiconductor CMOS*). *"Si el frente de onda es esférico, las franjas formadas por la rejilla son líneas rectas y paralelas. Si existen imperfecciones en el sistema óptico, las franjas ya no son paralelas, mostrando las correspondientes distorsiones de la superficie bajo prueba"* [37].

El sistema asistido por computador SAPRULL, consta del Módulo Interferómetro de Ronchi asistido por computador (IRAC) y el Módulo de Análisis de Ronchigramas por computador (MARC). El módulo IRAC es controlado por un operador, quien ingresa las especificaciones de movimiento de forma remota desde el computador, en el Software ANGMAR. El movimiento de la rejilla tiene una precisión del orden de $10 - 2$ *metros* y un desplazamiento unidimensional en línea directa con el foco del espejo de 0 a 0,5 *metros*. El Desplazamiento unidimensional del conjunto luz-cámara en un rango de $0{,}9m$ a $2{,}5m$ con respecto al espejo bajo prueba. Además, presenta confiabilidad en los desplazamientos al tener una respuesta en el tiempo baja (durante los movimientos).

El sistema automatizado, garantiza una mejor posición de los diferentes puntos donde se debe ubicar la cámara y la rejilla de Ronchi, con el fin de garantizar mediciones más fiables. Dependiendo del diámetro del espejo o lente, intervienen distancias como el punto focal y el centro de curvatura. Una de las características más importantes del sistema, es la calibración de la cámara, la cual debe situarse en el punto deseado, porque es donde se van obtener las muestras correctas para su posterior procesamiento, estas muestras son imágenes capturadas con buena resolución y pueden ser en tiempo real o durante un intervalo de muestreo para ser almacenadas. El Módulo de Análisis de Ronchigramas por computador (**MARC**), disponible también en el Software **ANGMAR** (*Figura 31*)**,** permite la adquisición de la imagen y control del dispositivo de captura escogido, realizando a su vez el análisis e interpretación cualitativa y cuantitativa de los patrones capturados e ingresados al computador.

Figura 31: Interfaz gráfica de Usuario ANGMAR **Fuente:***Diseño e implementación de un sistema de adquisición y análisis de información de Ronchigramas, mediante tratamiento digital de imágenes*[2].

4.2. Clasificación de Ronchigramas

El desarrollo del proyecto, profundiza en el uso de técnicas de aprendizaje automático en el campo de la visión por computador, a fin de implementar un algoritmo de aprendizaje capaz de determinar características de espejos y lograr describir aproximadamente el estado de un espejo con respecto a la superficie donde se forma la imagen. El problema de clasificación de Ronchigramas, puede abordarse desde el concepto de aprendizaje supervisado, ya que teniendo un conjunto de imágenes representativas de cada error dentro de una biblioteca, se utilizarán para entrenar el algoritmo seleccionado.

La implementación del algoritmo de aprendizaje, se organizó en seis etapas, ver esquema en la *Figura 32*, inicialmente se tienen dos etapas externas que son: la primera, donde se obtienen las imágenes desde la cámara y la segunda, es donde se acondicionan, para tener un modo escala idéntico y resolución proporcional para todas las imágenes, luego se procede a crear las carpetas con las clases que componen la Biblioteca.

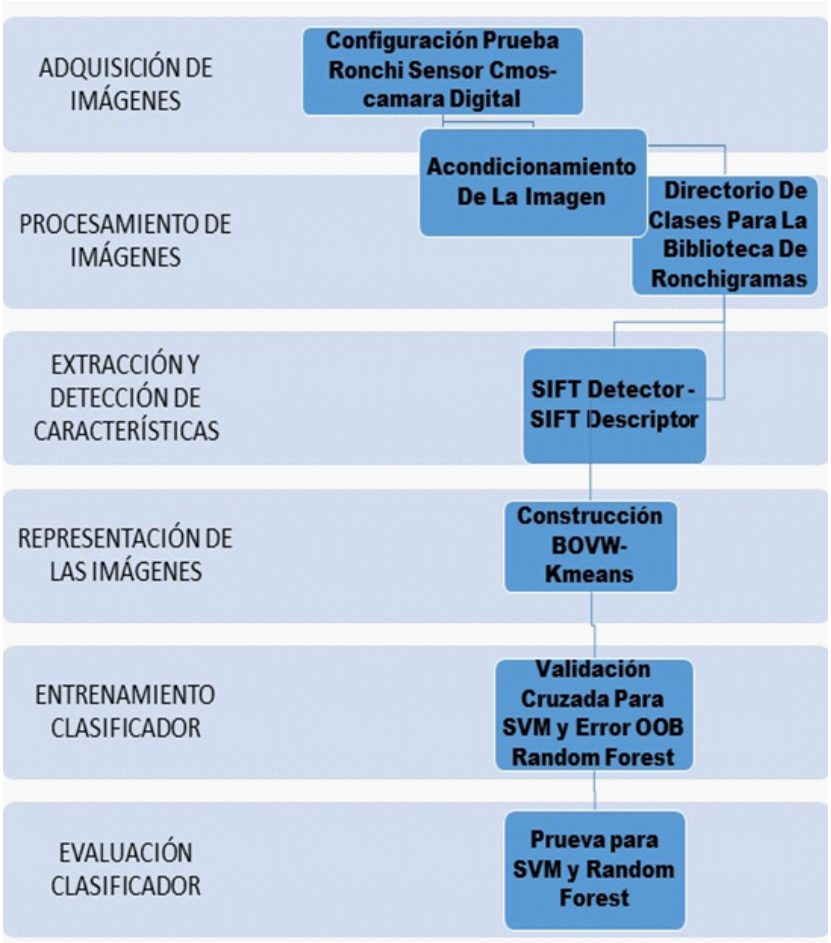

Figura 32: Esquema de Implementación del Algoritmo de Aprendizaje Automático.

Las etapas siguientes corresponden al esquema interno de la implementación del algoritmo de aprendizaje: en la tercera etapa, es donde se realiza la extracción de características y su descripción utilizando el algoritmo *SIFT*, para la cuarta etapa, se utiliza el método de *Bag of Visual Words* a fin de crear el vocabulario de palabras y generar el histograma normalizado para el caso de las palabras visuales y obtener la representación de la imagen. En la quinta etapa, se procedió a dividir la biblioteca en dos conjuntos de muestras, conjunto de entrenamiento y conjunto de evaluación, el primero representaría un porcentaje de muestras para el entrenamiento y el otro porcentaje para la evaluación. Para el entrenamiento de los dos algoritmos de aprendizaje, el primero la Maquina de Vectores de Soporte y el segundo un *Random Forest*.

Se utilizó el conjunto de entrenamiento a fin de encontrar las medidas de rendimiento óptimas, utilizando las técnicas de validación cruzada y estimación del error *OOB* (para el caso del *Random Forest*). En la última etapa, con los parámetros obtenidos de acuerdo al mejor rendimiento para cada algoritmo, se procede a evaluarlos, utilizando el conjunto de prueba que representaba el segundo porcentaje de muestras, obteniéndose de esta manera los diferentes resultados en las medidas de rendimiento del clasificador frente a las muestras desconocidas.

Adquisición de Imágenes: En esta fase se realiza el reconocimiento básico de la información, para la capturada de las imágenes se utilizó una cámara digital con sensor CMOS Exmor R® tipo 1/2,3 (7,82 mm), 20,4 MP, la cual brinda mejor resultado en espacios oscuros y en este caso la prueba de Ronchi se llevó a cabo en un espacio oscuro para capturar los Ronchigramas.

► *Metodología Para Armar El Arreglo Experimental*

1. Fijación del espejo en el soporte para objetos del sistema. El sistema no utiliza desplazamiento del espejo, por lo tanto se mantiene estático.

2. Alineación de la cámara, junto con el haz de luz y la rejilla Ronchi. Se utilizó una cámara CMOS a color de 20 Mpx. La cámara debe estar ubicada en el sistema de desplazamiento del modulo IRAC. La rejilla utilizada es de $100lpi(4$ lineas $/mm)$ y se coloca en un marco alineado con la rendija del haz de luz..

3. Difusor de luz y espejo reflector. Es un componente que se utiliza como filtro para suavizar el haz de luz del led.

4. Digitalización de las imágenes. La imagen capturada por la cámara CMOS es almacenada en el computador, en formato *jpg*, con una resolución de 256×256 pixeles a 8 bits por píxel.

Procesamiento de Imágenes: Incluye las técnicas de procesamiento de imágenes, como el filtrado para la reducción de ruido, tamaño y ubicación proporcional para todas las imágenes para mejores condiciones de análisis por computador. Las imágenes fueron seleccionadas visualmente, a fin de elegir imágenes con información relevante y sin afectar la muestra.

La muestra consistía en 14 espejos esféricos parabólicos para telescopios, con diferentes distancias focales y estos a su vez representaban algún tipo de aberración o espejos sin defectos. De esta forma se expusieron bajo la prueba Ronchi 14 espejos, 2 de los cuales representan la referencia de un espejo sin defectos, los espejos restantes presentan errores o defectos en la superficie y que, para algunos casos, un defecto es común para varios espejos; de esta forma la selección conformo cinco clases presentes en el conjunto de imágenes o biblioteca utilizada. En general la biblioteca de imágenes presenta en su totalidad 2327 imágenes alrededor de 100 y 700 imágenes por clase. Algunos ejemplos de las imágenes contenidas en las clases pueden observarse en la *Figura 35*. Las imágenes están trabajadas bajo ciertas propiedades o detalles

generales que identifican las imágenes y se puede mencionar algunas como el tamaño equivalente a 256×256 pixeles y en modo escala de gris (8 bits). Conociendo los aspectos teóricos de la prueba de Ronchi y la interpretación de los patrones (Ronchigramas) por un experto en fabricación de espejos, en el laboratorio de óptica de la Unillanos, se lograron formar las clases que fueron nombradas con el defecto referido en cuanto a la interpretación dada por la teoría de la prueba de Ronchi sobre los patrones (Ronchigramas) formados por los fenómenos de difracción e interferencia.

Los Ronchigramas se obtuvieron desde diferentes distancias con respecto a la distancia focal (ver el *cuadro 7*), utilizando la medida de milímetros para realizar movimientos cortos en el recorrido dentro del foco obteniéndose imágenes que varían en el número de franjas y para el caso fuera del foco las imágenes no eran capturadas visiblemente por la cámara.

$2{,}0cm$

$0cm$_____|_____

Foco $1{,}5cm$

_____|_____

Dentro de Foco

$1{,}0cm$

_____|_____

Dentro de Foco

Cuadro 7: Características de Espejos bajo la Prueba Ronchi.

	Características	
Id	Diámetro (cm)	Radio de Curvatura (mm)
Espejo N°1	20,5	2400
Espejo N°2	20	2560
Espejo N°3	17,5	2080
Espejo N°4	17,5	1910
Espejo N°5	15	2470
Espejo N°6	14,5	2450
Espejo N°7	17,5	2160
Espejo N°8	20	2400
Espejo N°9	14,7	2300
Espejo N°10	15	2440
Espejo N°11	15	2460
Espejo N°12	14	2400
Espejo N°13	17,5	2270
Espejo N°14	14,5	3200

Al realizar los desplazamientos dentro del foco$(0{,}5cm, 1{,}0cm, 1{,}5cm, 2{,}0cm)$, el número de franjas en cada Ronchigrama era diferente (Ver *Figura 33*), se observaba

aumento y disminución del número de franjas, pero representando el mismo error. Para el diagnóstico visual por parte de una persona experta, estos desplazamientos validan su concepto respecto a una aberración óptica, es decir son muy útiles para la correspondencia de errores desde diferentes distancias focales.

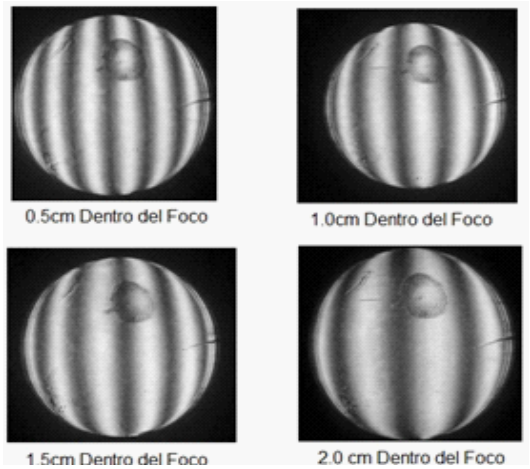

Figura 33: Rochigramas con desplazamientos dentro del foco.

Luego de tener las imágenes iniciales, para aumentar el número de muestras por espejo, se procede a realizar el sobremuestreo (*Oversampling*), que consistía en realizar giros a la imagen, sin alterar sus características, es decir, que se tenían 3 muestras adicionales (Giro Horizontal o efecto Espejo, Giro 180° y giro Vertical o efecto espejo para 180°) por cada Ronchigrama inicial, ver *Figura 34*.

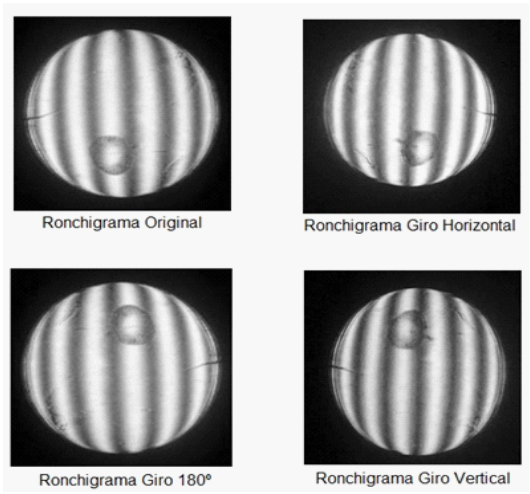

Figura 34: Sobremuestreo para cada Ronchigrama Inicial.

De esta forma se presenta el *cuadro 8* donde se relacionan la clase a la que pertenece cada espejo y el número de Ronchigramas tanto para el entrenamiento como para la evaluación; estos valores se tomaron relativamente con proporcionalidad de 70 % y 30 % respectivamente de cada clase de la biblioteca de Ronchigramas Original.

Cuadro 8: Clases y Cantidad de Ronchigramas en el conjunto de entrenamiento y de prueba.

	Muestra de Espejos	Cantidad de Ronchigramas	
	Tipo: Parabólico		
Clases	Id	Train	Test
1.Sin Defectos	Espejo N°1	561	240
	Espejo N°12		
2.SobreCorr3(Defectos en el Centro)	Espejo N°2	464	199
	Espejo N°3		
	Espejo N°14		
3.SobreCorr9(Defectos en el Centro)	Espejo N°4	342	146
	Espejo N°7		
	Espejo N°11		
	Espejo N°13		
4.Sobre9SubCorr3(Defectos Bordes y Centro)	Espejo N°5	337	145
	Espejo N°6		
	Espejo N°8		
	Espejo N°9		
5.SubCorr9(Defectos en los Bordes)	Espejo N°10	84	36

Los Ronchigramas obtenidos se pueden observar en la *Figura 35*, de derecha a izquierda se tienen los Ronchigramas representativos de las clases del 1 al 5 respectivamente del cuadro 8, para la clase 5 se tenía la muestra de un solo espejo y por lo tanto la clase presenta un desbalance en el número de imágenes para la conformar el conjunto de entrenamiento y de evaluación de esa clase con respecto a las otras.

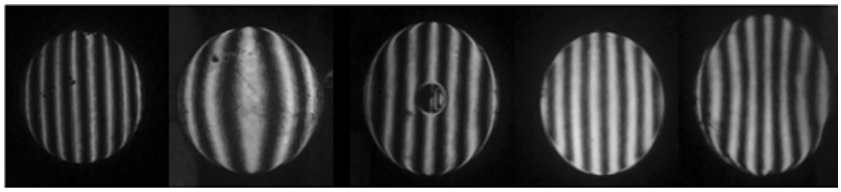

Figura 35: Ronchigramas representativos dentro de la Biblioteca de Ronchigramas.

Inicialmente se pensó en realizar aumento del conjunto original, utilizando la técnica de *data augmentation*, aplicando diferentes transformaciones como, movimientos, cambios de modo, desenfoques y otras modificaciones (ver *Figura 36*), pero se observó que se alteraban las características del patrón y no habría sentido al momento de entrenar el algoritmo de aprendizaje. Por lo anterior, se optó trabajar con el número de imágenes iniciales del *cuadro 8* sin provocar otros efectos distintos a los descritos por la técnica del sobre-muestreo.

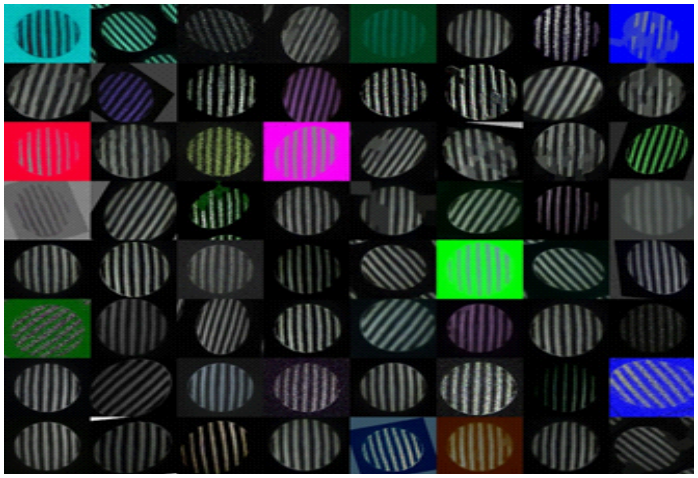

Figura 36: Técnica de Data Aumentation para el Espejo 10 de la clase 5.

4.3. Experimentos con Clasificadores SVM y Random Forest

Continuando con las etapas del esquema de implementación, la tercera etapa corresponde a la **Detección y Descripción de características**: En esta etapa se realiza la detección y descripción de puntos o características de interés de una imagen, utilizando el algoritmo *Scale Invariant Feature Transform* (*SIFT*)[35]. SIFT es una de las técnicas usuales para encontrar características locales de una imagen y muestra buen desempeño al trabajar con los Ronchigramas; otras conocidas y que también pueden utilizarse son: *Speeded Up Robust Features* (*SURF*), que es similar y tiene la misma idea que *SIFT*, pero es mas rapido y robusto frente a desenfoques, aunque su capacidad se ve disminuida frente a problemas de iluminación. Existen otras tecnicas y por mencionar aquellas que trabajan por *Selección de Muestreo* estan: el muestreo *Denso* y *Aleatorio[8]*, pero ambas conllevan tanto a la parametrización de escala y espacio como al aumento del costo computacional.

Representación de la Imagen: En esta etapa se asignaron los valores representativos a las características extraídas en cada imagen; se utilizó el método de *Bag of Visual Words* (***BOVW***) a fin de crear el vocabulario de palabras visuales y mediante el histograma normalizado de las palabras visuales obtener la representación de la imagen (*Figura 37).*

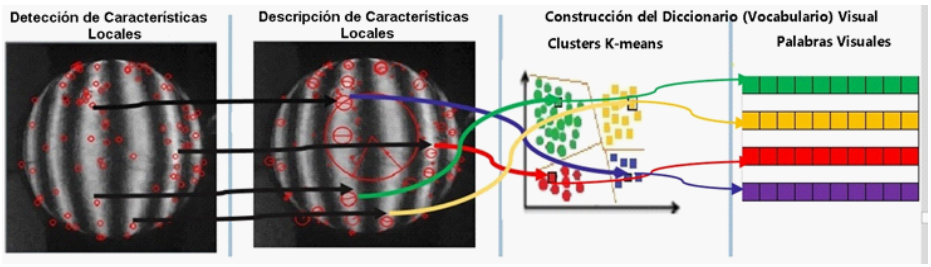

Figura 37: Palabras Visuales obtenidas por el método de BOVW.

⁕ Implementación del Método; los principales pasos del Metodo *BOVW*, aplicados fueron:

- Detección y descripción de puntos de interés de la imagen, usando *SIFT*.
- Asignación de descriptores locales a un conjunto de clústeres predeterminados, usando el algoritmo de *K-means* para la cuantificación vectorial. Se parametrizo el Vocabulario con 50,000 puntos de interés y trabajando distintos valores para el número de palabras $256, 512$ y 1,024 del vocabulario visual (k).

4.3.1. Entrenamiento del clasificador SVM y Random Forest

Seguidas las etapas cinco y seis del esquema de implementación, se procede a entrenar los algoritmos de aprendizaje, las etapas iniciales son aprovechadas para el entrenamiento de ambos clasificadores, tanto para la Máquina de Vectores de Soporte (SVM) como para el algoritmo de bosques aleatorios (*Random Forest*).

En primer lugar, cada clase es recorrida por el metodo *SIFT,* ver *Figura 38,* y así generar el vector de descriptores de cada imagen, luego de extraer los detectores y descriptores usando el método SIFT para cada imagen, se proporciona una serie de puntos de interés (keypoints) con el fin de construir la representación de cada imagen usando el método *BOVW*, ver *Figura 37*, para generar las palabras visuales que conformaran el vocabulario visual, además la *Figura 39*, hace énfasis a la tarea de entrenamiento marcada con un cuadro rojo para su explicación más adelante ver *Figura 39* y *Figura 40*.

Teniendo los vectores de características obtenidos por *BOVW*, se procedio a crear un amplio vocabulario de descriptores y luego asignar cada palabra de los Ronchigramas a una del vocabulario, generando así un histograma, para proceder a representar cada imagen tanto para el conjunto de entrenamiento como para el de prueba.

El número de muestras utilizado fue de 50,000 para la extracción y un número de centroides o palabras de $256, 512$ y 1,024 para formar los diferentes vocabularios del entrenamiento. Para la elección del número de muestras, se consultó en la literatura que un número de descriptores trabajado generalmente por encima de las 50,000

muestras es una buena elección y calculando hasta el total de imágenes por clase en el conjunto de entrenamiento, se obtuvieron 27 descriptores por imagen (50,000/1,788 = 27, 964).

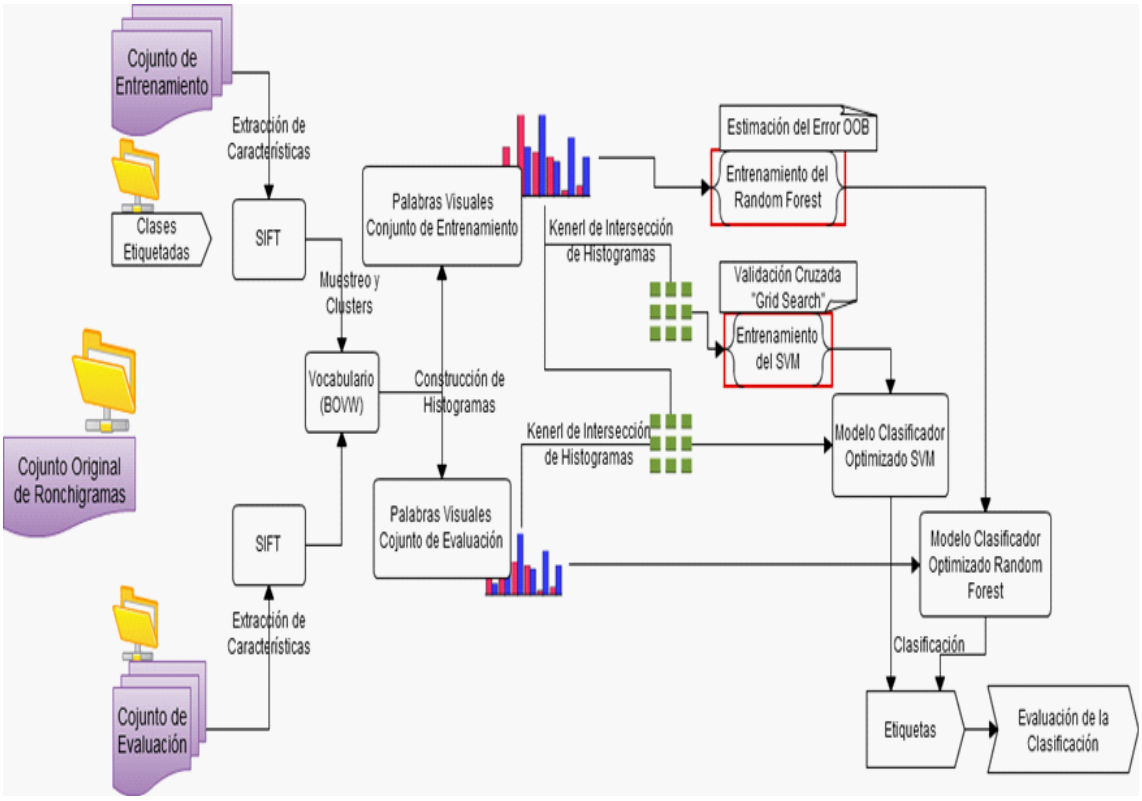

Figura 38: Diagrama de Flujo de la Implementación de los Clasificadores *SVM* y *Random Forest*.

La Máquina de Vectores de Soporte permite la inclusión de funciones de *Kernels* externas, para el método de *Bag of Visual Words*, se utilizó un kernel especial de Intersección de Histograma.

El *kernel* especial, fue la entrada para el entrenamiento y evaluación del *SVM*. comparar histogramas y contar las frecuencias de palabras visuales en cada imagen y elegir el valor mínimo que representa la palabra visual, así el entrenamiento con este tipo de kernel es usado por la *SVM* recibiendo la matriz obtenida de la concatenación de histogramas de las palabras visuales que representan las imágenes del conjunto de entrenamiento, ver *Figura 39*.

Con los vocabulario creados de acuerdo al número de palabras en el rango de 256, 512 y 1024 y teniendo las palabras visuales que representan las imágenes del conjunto de entrenamiento, se aplica el *kernel* especial (Intersección de Histogramas) y se procede a la tarea de entrenamiento para buscar los parámetros óptimos para la Máquina de Vectores Soporte.

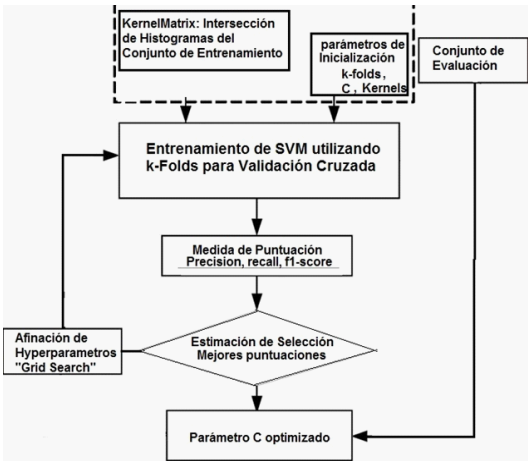

Figura 39: Diagrama de Flujo del Entrenamiento del clasificador *SVM*. **Fuente***: Intelligent Optimization Methods for High-Dimensional Data Classification for Support Vector Machines* [10].

En el caso del *Random Forest*, su parametrizacón es mas simple, por lo tanto se procede a entrenar y evaluar con los conjuntos de muestras correspondientes, ver *Figura 40*.

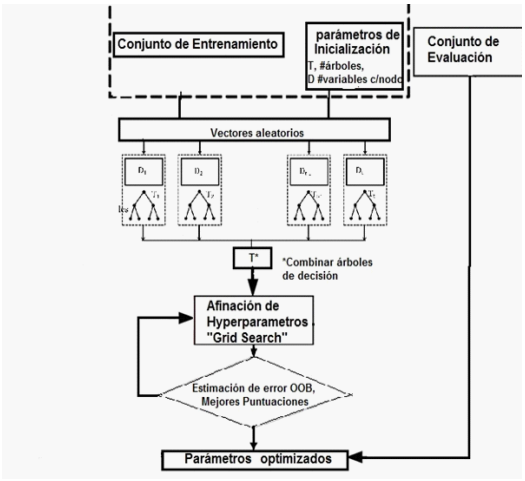

Figura 40: Diagrama de Flujo del Entrenamiento del clasificador *Random Forest.* **Fuente***: Evaluating total inorganic nitrogen in coastal waters through fusion ofmulti-temporal RADARSAT-2 and optical imagery using randomforest algorithm* [11].

Usando la técnica de Validación cruzada ver *Figura 41* y dado el caso para este problema de clasificación, donde una de las clases tiene desbalance de muestras como es la clase 5 (con apenas 86 imágenes), no resulta aconsejable utilizar la medida de desempeño de exactitud como medida de evaluación del rendimiento para el clasificador *SVM*. Sin embargo para la elección de Hiperparámetros (*tuned_parameters*) se

presentan los resultados para las mejores puntuaciones obtenidas usando un vocabulario de 1,024 palabras visuales y el análisis de las medidas de Precisión (*Precision*) y Sensibilidad (*Recall*) ambas disponible en la librería de Sckit learn para la estimación de hiperparámetros mediante búsqueda de cuadrícula (*GridSearchCV*) con Validación Cruzada [**sklearn.model_selection.GridSearchCV**]. Los parámetros para realizar la validación cruzada en el aprendizaje del clasificador son: 5 particiones (*folds*), un rango de 10 valores diferentes con valor inicial de 0,0001 y final de 0,02 para elegir el factor de regularización C y dos kernels internos diferentes "*kernel Lineal*" y "*kernel Precomputed*".

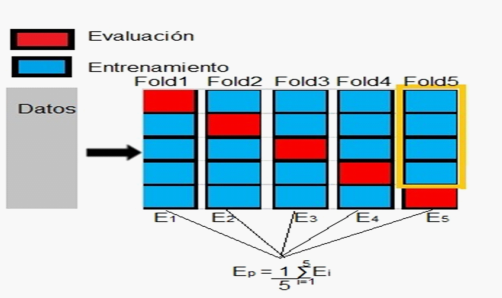

Figura 41: Validación Cruzada, búsqueda de Hiperparámetros *SVM*. **Fuente**: Autor

En el caso del clasificador *Random Forest*, la validación de parámetros se realiza inicializando el número de arboles dependiendo del resultado obtenido en la curva *OOB*, donde se observa la estabilización del error y el número de variables por cada nodo.

4.3.2. Resultados de Evaluación del Clasificador SVM y Random Forest

Utilizando la Matriz de Confusion para medir el rendimiento de los clasificadores, se pueden encontrar las medidas de Exactitud (*acc*), Precisión y Recall.

		Resultado del Clasificador		
		Clase 1	Clase 2	
Resultado Correcto	Clase 1	*TP*	*FN*	*P*
	Clase 2	*FP*	*TN*	*N*

$TP(True\ Positive) = Verdaderos\ Positivos$
$FP(False\ Positive) = Falsos\ Positivos$
$FN(False\ Negative) = Falsos\ Negativos$
$TN(True\ Negative) = Verdaderos\ Negativos$
$P\ (Positive) = Condicíón\ Positiva\ (TP + FN)$
$N(Negative) = Condición\ Negativa\ (FP + TN)$

$$acc = \frac{TP+TN}{P+N},$$
$$precision = \frac{TP}{TP+FP}$$

Para este problema especial de clasificación se utilizan las mejores puntuaciones obtenidas en las medidas de Precisión y Sensibilidad (*Recall*) para evaluar el rendimiento del clasificador ya que se tiene una clase con menor número de muestras, generando un desbalance; además se utilizó la curva ROC(*receiver operating characteristic*) la cual representa el valor de *TPR (True Positive Rate)* en función del *FPR (False Positive Rate),* indicando el coste de obtener falsos positivos (FPR) para obtener el beneficio de reales positivos(TPR), alcanzado por el modelo clasificador para cada una de las clases y la curva de **precision vs recall** a fin de utilizar la medida de precisión media (*average precision*) como medida representativa para elegir el mejor modelo clasificador optimizado.

***False Positive Rate*(*FPR*):** Es la proporción de Ronchigramas de una clase marcados como correctos en algunas de las otras clases con respecto del total de Ronchigramas:

$$FPR = \frac{FP}{N}$$

***True Positive Rate*(*TPR*):**Es la proporción de Ronchigramas correctamente clasificados de una clase con respecto del total de Ronchigramas clasificados en esa clase:

$$TPR = \frac{TP}{N}$$

Además de la curva de *Precisión vs Recall* vista en una sección anterior del presente libro, se obtiene la curva ROC la cual permite medir el rendimiento teórico del clasificador al calcular el área bajo la curva (AUC), de los elementos clasificados correctamente y los mal clasificados, mostrando una oscilación entre 0 y 1, respectivamente.

En la práctica la curva ROC tambien se presenta como una buena fuente de información para analizar el comportamiento del modelo clasificador y al final de acuerdo al valor obtenido en esta curva junto con la medida de pecisión media obtenida por la curva *Precisión vs Recall* se procede a elegir el mejor rendimiento para el modelo clasificador.

4.3.2.1. Evaluación del Clasificador SVM

El *cuadro 9*, presenta los mejores parámetros optimizados por la búsqueda de cuadricula (*GridSearch*) durante el entrenamiento, utilizando validación cruzada; se inicializó con el *Kernel* "*Lineal*" y "*Precomputado"*, para esta validación cruzada es recomendable un tercer conjunto de evaluación. El tiempo de ejecución varia de acuerdo al número de parámetros que se inicialicen para validación y tambien de la *CPU* donde se ejecute el programa.

Cuadro 9: Mejores parámetros seleccionados para el Clasificador *SVM*.**Fuente**: Autor

Mejores Puntuaciones durante el aprendizaje	Parámetros	Tiempo
	kernel: precomputed	1013 seg
Presicion:0.78 - Recall:0.77	C: 0.018	

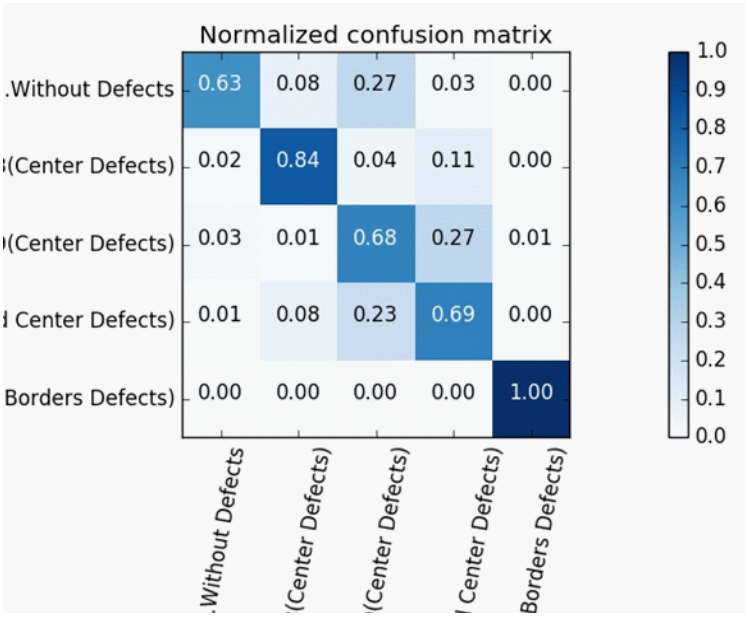

Figura 42: Matriz de Confusión del Clasificador *SVM*.

La *Figura 42,* muestra un rendimiento del clasificador *SVM* de 74 % en términos de la medida de precisión media (*average precision*) y una desviación estandar de $\sigma = 2.6\,\%$ de muestras clasificadas correctamente, la dispersión se ve marcada por el desbalance de muestras.

El clasificador *SVM* se confunde en tres clases marcadas por debajo del 70 % de Ronchigramas clasificados correctamente, de esta forma se observa que las clases 3 y 4 por tener Ronchigramas con errores en el centro pueden corresponder a características marcadas de una o de la otra.

Para la clase 1, del espejo sin defectos, un 30 % de Ronchigramas son mal clasificadas, representando un falso positivo para la clase 3 con errores en el centro.

La Máquina de Vectores de Soporte al ser un clasificador que divide las muestras por medio de hiperplanos, presenta ciertas márgenes, las cuales son controladas por el factor de regularización y esto hace, que ciertas muestras puedan violar dicha margen y se conviertan en falsos positivos, confundiendo la tarea de clasificación.

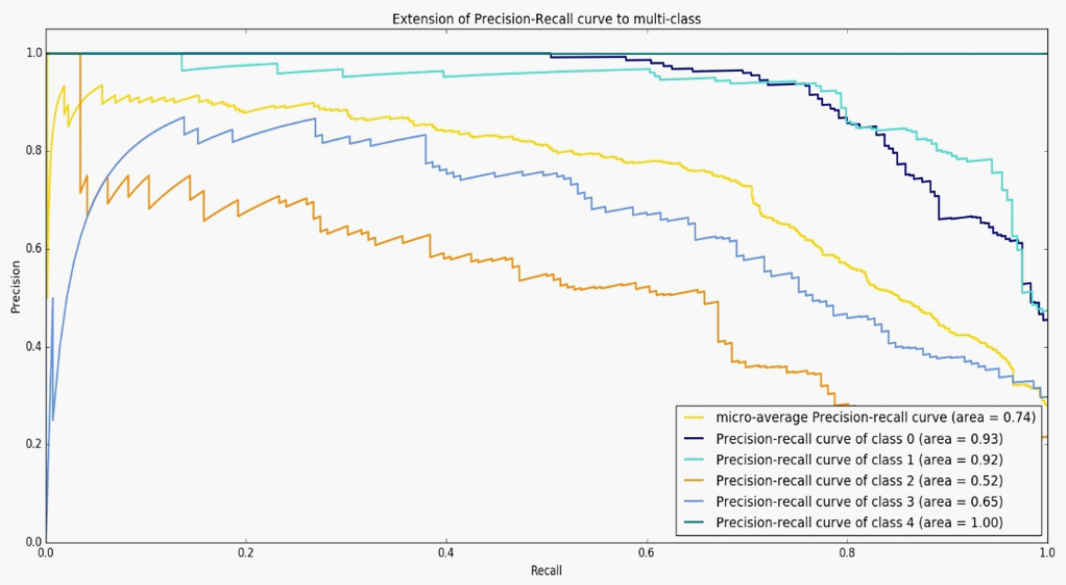

Figura 43: Curva *Precision vs Recall* del Clasificador *SVM*.

La precisión media alcanzada por el clasificador *SVM* fue de 0.74 (*Figura 43*).

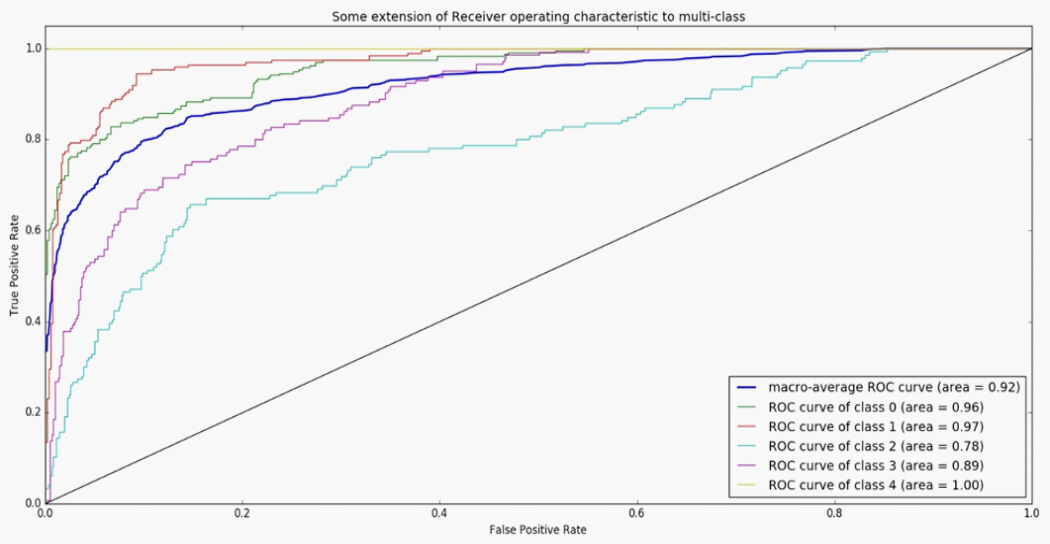

Figura 44: Curva *ROC* del Clasificador *SVM*.

La curva ROC del clasificador *SVM* (*Figura 44*), presenta una buena generalización, permitiendo que el margen de separación en cada una de las clases admita muestras pertenecientes a otras (Falsos Positivos) y que el costo sea menor, aprovechando las muestras reales clasificadas correctamente, verificable también con la matriz de confusión, donde la mayoría de las muestras es clasificada correctamente.

4.3.2.2. Evaluación del Clasificador Random Forest

Para la parametrización del *Random Forest* se procede a entrenar el algoritmo bajo la búsqueda de cuadricula (*GridSearch*), inicializando diferentes valores de árboles y de acuerdo al número de características, el número de variables en cada nodo; de esta forma se procede a estimar el error *OOB* (*out of bag*) para estimar aquellos valores donde el error se estabilice[38] para encontrar el número de árboles óptimo de acuerdo al número de características para la clasificación de Ronchigramas.

Esta característica del *Random Forest*, lo hace más eficiente frente a la Maquina de Vectores de Soporte ya que para optimizar los parámetros no se necesita realizar la validación cruzada, esto se debe a que el atributo *OOB*, hace referencia a aquellas muestras que quedaron fuera (*out of bag*) durante la primera iteración del árbol, y no son usadas para crear el árbol. Durante cada iteración, para estas observaciones se realiza la predicción y se calcula el error de la misma para estimar el error *OOB*.

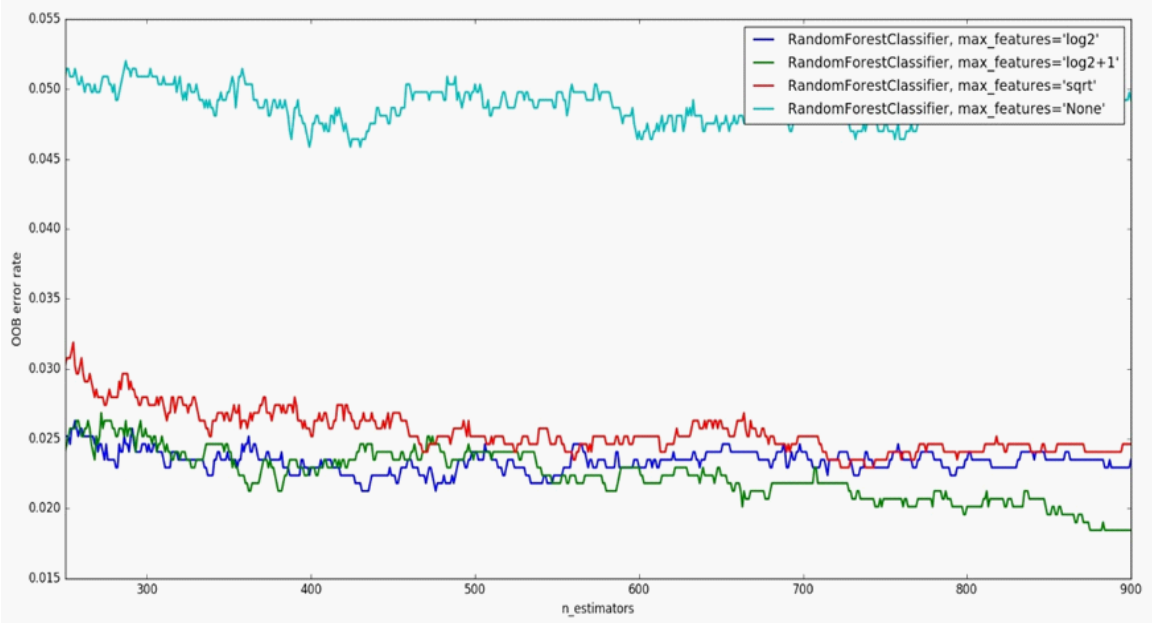

Figura 45: Estimación del Error *OOB* para el Clasificador *Random Forest*.

Se observa en la *Figura 45* que el error se estabiliza casi en los 900 árboles y que la curva con menor estimación del error *OOB*, es para un máximo de características igual al *log2+1* del total de características descritas por el *BOVW*.

El clasificador *Random Forest* muestra una interesante respuesta de clasificación como se observa en la matriz de confusión normalizada (ver *Figura 47*), donde se alcanzan altos valores de predicciones contra los reales, tal es el caso de la clase 1, 2 y 5 donde las muestras del conjunto de evaluación son correctamente clasificadas y también se observa que solo se confunde en las clases 3 y 4.

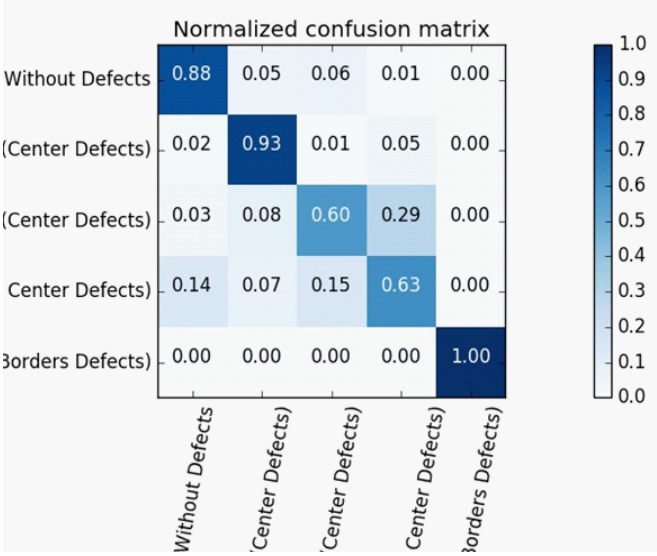

Figura 46: Matriz de Confusión del Clasificador *Random Forest*.

Se observa en la *Figura 46* que el *Random Forest* alcanza un rendimiento del 88 % en términos de la precisión media y una desviación estandar de $\sigma = 3.5\,\%$, un valor representativo frente al presentado por la Maquina de Vectores de Soporte.

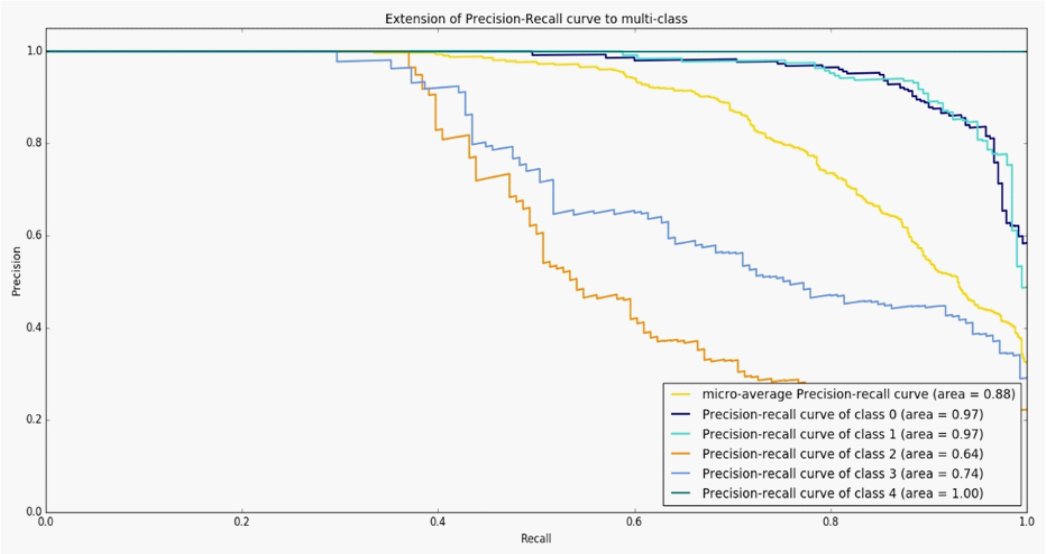

Figura 47: Curva *Precision vs Recall* del Clasificador *Random Forest*.

La medida de precisión media es muy importante y es utilizada en este proyecto para elegir el nivel de aprendizaje del algoritmo, ya que esta precisión alcanzada es con respecto a las muestras del conjunto de evaluación el cual son desconocidas por el clasificador.

La desviación estandar alcanzada es de 3.5 % muestras clasificadas correctamente.

Una hipótesis planteada para la confusión de las clases 3 y 4, es que realmente algunas imágenes dentro de estas clases representan características de un patrón marcado en la clase 2, se puede observar que tanto las clases 2, 3 y la 4 representan el error de defectos en el centro, solo con alguna modificación en cuanto al valor de sobre corregimiento como la adición del defecto de borde respectivamente.

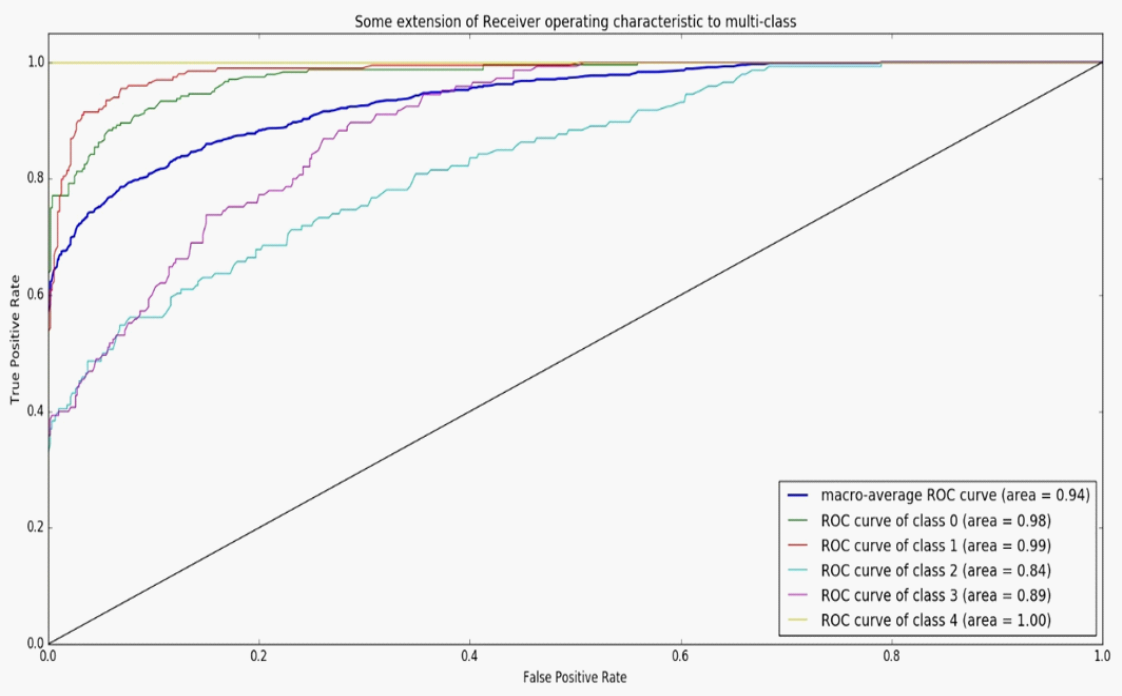

Figura 48: Curva *ROC* del Clasificadro *Random Forest*.

El resultado obtenido en la curva *ROC* para el *Random Forest* (*Figura 48)*, muestra un promedio de área bajo la curva de 0.94, demostrando que el modelo clasificador tienen buena generalización entre cada una de las clases .

Finalmente, en la matriz de confusión algunas muestras en su minoría pueden pertenecer a otra clases que también tenga características del patrón en una determinada aberración óptica como lo son las clase 3 y 4.

En la *cuadro 10*, se registra la precisión media alcanzada para cada uno de los clasificadores por cada vocabulario con diferente número de palabras visuales. La Máquina de Vectores de Soporte, presenta valores signficativos que indican que el clasificador mejora a medida que se aumentan el vocabulario. Para el clasificador *SVM* además de las funciones *kernels* especiales, junto con la validación cruzada, muestra que requiere de un importante análisis de parametrización, a differencia del *Random Forest* que mantiene buena precisión para los distinto vocabularios.

Cuadro 10: Medidas de Rendimiento Alcanzadas por los Clasificadores SVM y RF.

Número de Palabras del Vocabulario	Random Forest			SVM		
	AV	Acc	T	AV	Acc	T
256	0,86	0,75	11 seg	0,68	0,69	318 seg
512	0,86	0,75	14 seg	0,68	0,69	421 seg
1024	0,88	0,78	19 seg	0,74	0,72	532 seg

AV = Precisión Media
Acc = Exactitud
T= Tiempo de ejecución

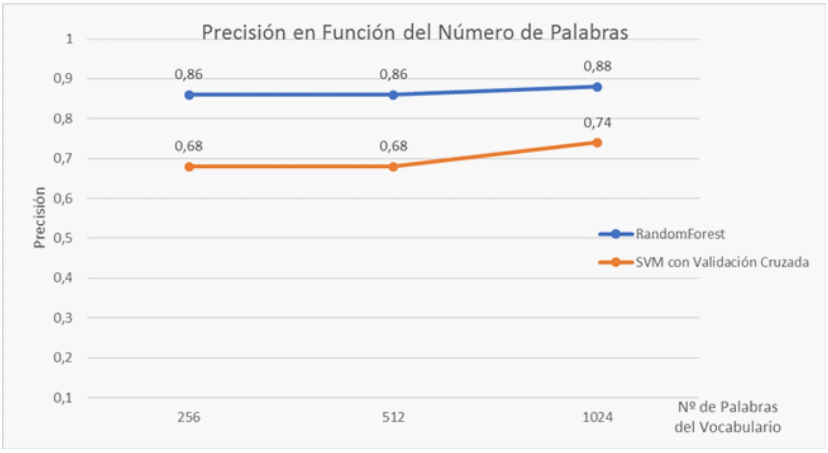

Figura 49: Precisión alcanzada por los clasificadores *SVM* y *RF* frente al número de palabras visuales del vocabulario.

En la *Figura 49*, se observa el aumento del rendimiento por número de palabras en cada vocabulario, para cada uno de los calsificadores. El *Random Forest* muestra estabilidad en los vocabularios de 256 y 512 palabras visuales, alcanzando 86 % de rendimiento en términos de precisión.

4.4. Implementación en Python

Con los resultados obtenidos en el rendimiento del clasificador, se logró identificar y elegir el algoritmo Random Forest como el más conveniente para ser implementado en la tarea de análisis de Ronchigramas del Módulo M**ARC**.

Las fuentes bibliográficas y desarrollos a nivel de código para algoritmos, trabajan en entornos científicos, como MATLAB, con un enfoque de presentación explicita de los datos. Para la codificación trabajada en el módulo MARC del proyecto SAPRULL,

utilizando el lenguaje de programación Python, se eligió el entorno de desarrollo SPYDER (Scientific PYthon Development EnviRonment), el cual es un IDE interactivo de uso libre que está incluido en Ancaconda, una plataforma abierta para la ciencia de datos impulsada por Python. *"La versión de código abierto de Anaconda es una distribución de alto rendimiento de Python y R e incluye más de 100 de los paquetes de Python, R y Scala más populares para la ciencia de los datos"* [13]. Cabe resaltar que Python desde su creación en 1991, iniciando con una versión 0.9.0, tenía fines matemáticos y de análisis de datos, pasando de su versión 1.0 hasta llegar a las versiones disponibles 3.6.1 / 2.7. Para el año 2017, se ha convertido en uno de los lenguajes de programación más utilizado y una herramienta muy potente para el desarrollo de algoritmos en cualquier ámbito, logrando gran auge por su claridad en cuanto a sintaxis de código.

Para la codificación del proyecto (*Figura 50*), se utilizó Python 2.7 en el entorno SPYDER, utilizando diferentes paquetes de librería disponibles y necesarias para el desarrollo de los algoritmos.

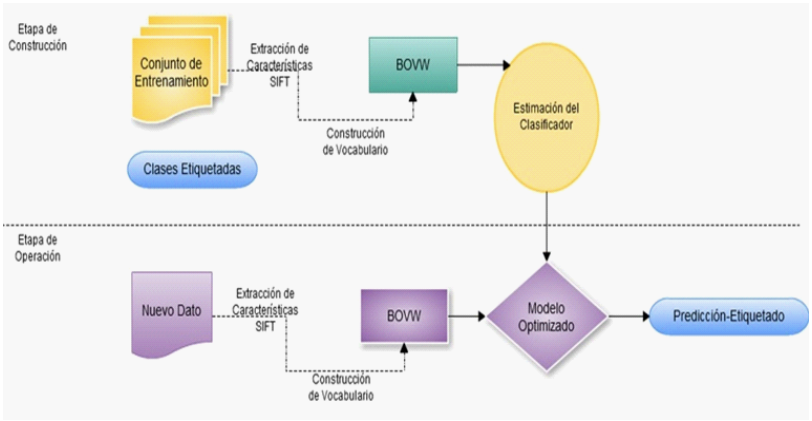

Figura 50: Diagrama de Flujo de Implementación del Clasificador Random Forest

Para describir con una breve reseña la funcionalidad de las librerias, se mencionan algunas de ellas internas y otras externas que requirieron ser instaladas, para la implementación del algoritmo de aprendizaje:

Pylab: Es el paquete o conjunto de librerías que incluye numpy, scipy, sympy, pandas, matplotlib, ipython, un ejemplo de su uso es la interfaz PyLab de Matplotlib donde se encuentra el conjunto de funciones que permiten al usuario crear tramas.

Matplotlib: En las tareas de análisis matemático y trazo de puntos y graficas o líneas y curvas en imágenes, Matplotlib es una librería de gráficos potente que permite la ejecución de funciones para realizar las tareas mencionadas. Matplotlib es de código abierto y está disponible gratuitamente en http://matplotlib.sourceforge.net/. [14]

Pickle: Se utiliza para guardar algunos resultados o datos para su uso posterior, el módulo Pickle puede tomar casi cualquier objeto Python y convertirlo en una

representación de cadena. [14]. Este proceso busca persistir. Esta representación de cadena puede ser fácilmente almacenada o transmitida.

NumPy: Librería que brinda soporte en las operaciones con matrices y vectores.

SciPy: Es una librería de código abierto para el análisis matemático, que se basa en NumPy y proporciona rutinas eficientes para una serie de operaciones, incluyendo la integración numérica, optimización, estadísticas, procesamiento de señales y procesamiento de imágenes. Disponible en http://scipy.org. [14]

sklearn: El caso de scikit-learn es una librería centrada en machine learning, permitiendo ejecutar algoritmos ya implementados para el aprendizaje de máquina, en temas de clasificación, regresión y agrupación, máquinas de vectores de soporte(SVM), gradiente descendente, k-means además permite interoperar con librerías como NumPy y SciPy.

OpenCV: cv2, es una librería C ++ con módulos que cubren muchas áreas de la visión por computador. La interfaz de Python permite el uso de algunas funciones, aunque no todas están documentadas, se puede consultar en: https://opencv-python-tutroals.readthedocs.io/en/latest/index.html (consulta realizada Mayo 2017).

P**yQt:** Es la librería para el entrono grafico Qt para Python.

4.4.1. Interfaz gráfica

Para el procesamiento de Ronchigramas, la interfaz gráfica utilizada es muy sencilla para el usuario (*Figura 51*), consta de los botones cargar y evaluar, el primero permite elegir en una dirección donde se aloja el Ronchigrama que se desea evaluar, el segundo predice la clase de error a la que pertenece el Ronchigrama y su grado de incertidumbre de acuerdo al rendimiento del clasificador y esta respuesta, es entregada a partir del modelo entrenado, el cual se persiste en formato de datos masivos (.pkl).

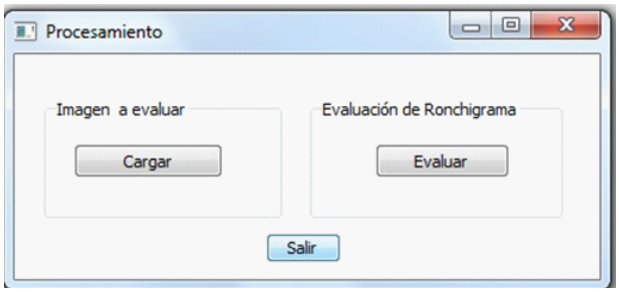

Figura 51: Interfaz gráfica de la herramienta para el procesamiento de Ronchigramas.

Algunas excepciones que advierten al usuario para el manejo de la herramienta, es el caso de no cargar el Ronchigrama antes de evaluar, por lo tanto, no puede existir predicción (*Figura 52*).

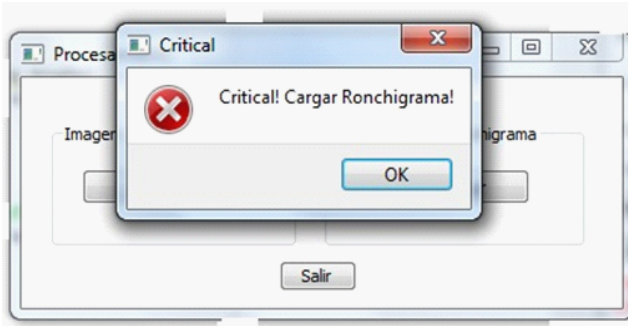

Figura 52: Excepción de error Critico. "No se ha cargado Ronchigrama".

Para el caso del algún error interno del programa, se muestra el error presentado para que pueda ser solucionado (*Figura53*).

Figura 53: Excepción Error interno. El vocabulario no se ha definido.

Capítulo V

CONCLUSIONES Y TRABAJO FUTURO

5.1. Conclusiones

El desarrollo del proyecto permitió la evaluación de los algoritmos de aprendizaje automático, para cada clasificador. Se analizaron las medidas de precisión media alcanzada, tanto para la Maquina de Vectores de Soporte como para el Random Forest y de esta forma elegir el mejor rendimiento para el modelo de clasificación. El modelo del clasificador optimizado de la Maquina de Vectores de Soporte alcanzo 74 % de precisión media y para el *Random Forest* se alcanzó 88 %, este valor final demuestra la superioridad del clasificador y se elige como el modelo optimizado para el procesamiento de patrones de Ronchi para determinar características de espejos.

Las técnicas de procesamiento de imágenes son generalmente muy específicas, es decir, dependiendo de la imagen a tratar existen métodos que se adaptan mejor y permiten facilidad para el análisis de datos, en el caso de Ronchigramas, las características locales *SIFT* funcionaron bien, siendo una de las primeras técnicas para la detección y descripción de puntos de interés, esto permitió describir el contenido de la imagen y lograr extraer dichos puntos para la creación del vocabulario.

El vocabulario BOVW con mayor número de Palabras visuales 1024, mejoró el rendimiento de clasificación para el caso del clasificador SVM, mientras que para el Random Forest se mantuvo estable, lo que demuestra la independencia de este último clasificador con respecto a algunos parámetros de inicialización.

Se logró abordar el problema de clasificación de Ronchigramas desde la perspectiva de la óptica geométrica y medir la aberración transversal (defecto óptico) en una forma directa, al capturar el patrón que se forma y computarizar los datos obtenidos de la imagen, influyendo inicialmente los factores cualitativos de la observación de un experto para generar el nombre de las clases en la biblioteca de Ronchigramas. Cada clase correspondió a una aberración óptica o estado característico de un espejo bajo prueba.

Los patrones Ronchi pueden variar en cuanto al número de franjas, sin modificarse el tipo de aberración, es decir que el desplazamiento dentro y fuera del foco, arroja una cantidad de información relevante que ayuda a validar el diagnostico y que por lo tanto las imágenes obtenidas a diferentes distancias no alteraron el problema de clasificación en la parte computacional.

La implementación de una máquina de vectores de soporte, se convierte en una tarea dispendiosa, la cual debe retroalimentarse constantemente, es decir, conocer

el comportamiento de la máquina y parametrizar su entrenamiento. Un factor importante para el aprendizaje de máquina es el conjunto de validación durante el entrenamiento, usando las técnicas de validación cruzada que ayudan al algoritmo a encontrar la solución más óptima, aprendiendo constantemente de los resultados obtenidos en cada validación.

Las medidas de desempeño *precisión y sensibilidad* al ser trabajadas como puntuaciones en casos reales, permiten obtener diferentes resultados. Estas puntuaciones al ser graficadas representan curvas, como lo es la curva *precisión-sensibilidad*, la cual muestra que ambas medidas son inversamente proporcionales, porque al tener una mayor sensibilidad, las muestras tanto verdaderos positivos como falsos positivos se verán clasificados con una *precisión* baja y en cambio sí disminuye la sensibilidad la *precisión* de la clasificación aumentara. Para el problema de clasificación de Ronchigramas y la presencia de un desbalance de clases, no fue recomendable enfocar la evaluación del rendimiento en un alto puntaje de *exactitud*, sino que se eligió un alto puntaje en la precisión para que el clasificador tuviera la capacidad de generalización en cada una de las clases, como se observa en la sección de resultados y junto con la literatura consultada la medida de exactitud es un valor que tiene importancia principalmente en casos ideales.

El problema de clasificación de Ronchigramas no es linealmente separable y se deduce, al observar el comportamiento del clasificador *SVM* que muestra bajos resultados de clasificación para modelos de clasificadores con funciones lineales y para este caso, se necesitó de un kernel especial de Intersección de Histogramas a la entrada del *SVM* y la parametrización del factor de regularización, para de esta forma, mejorar un poco el rendimiento del clasificador.

Se logró implementar el algoritmo de aprendizaje del *Random Forest* como clasificador de Ronchigramas a fin de determinar las aberraciones ópticas en espejos, la codificación se llevó a cabo en el lenguaje de programación Python para ser implementada en el Módulo **MARC** del Sistema Asistido por Computador de la Prueba de Ronchi en el Laboratorio de óptica de la Universidad de los Llanos (**SAPRULL**). La parametrización de este algoritmo, para el problema de clasificación de Ronchigramas, es más sencilla que la realizada para el *SVM*, ya que la búsqueda de parámetros, solo necesito del tamaño del bosque o número de árboles y el valor paramétrico del número de variables por cada nodo, esto dependiendo del número de características de las imágenes.

Random Forest lidia muy bien con problemas de desbalance de clases y nuevas técnicas han abordado este tema para mejorar los rendimientos del clasificador. En la teoría, *Random Forest,* como método de conjunto de clasificadores, resulta viable en casos de decisión para las muestras que presenten un poco de ruido porque permite generalizar las características de cada una de las muestras permitiendo un margen de infracción y el resto se penaliza para lograr clasificar correctamente cada muestra en la clase respectiva.

Finalmente, *Random Forest* como clasificador, establece una buena solución al problema de clasificacion de Ronchigramas, alcanzando una *precisión media* del 88 % ±3,5 y un tiempo de ejecución de 19 segundos para la evaluación, sobrepasando en rendimiento y tiempo de ejecución con respecto al *SVM*, el cual alcanzo 74 % de precisión media en 532 segundos para la evaluación. La parametrización y robustez del *Random Forest* frente a ruidos de las imágenes lo hace idóneo para lograr determinar las aberraciones ópticas en espejos.

5.2. Trabajo Futuro

El número total de Ronchigramas para conformar la biblioteca fue un valor relativamente representativo, ya que debe aumentarse el número de muestras y el número de espejos para la clase de aberración óptica que presenta menor número de Ronchigramas con respecto a las otras clases, el aumento debe realizarse con la ayuda de un experto en óptica, al ser expuestos bajo prueba, un significativo número de espejos a los cuales se les aplique el pulido y se provoque algún tipo de defecto óptico al espejo. Mejorando así la capacidad de aprendizaje del algoritmo con respecto a nuevas muestras.

Para profundizar en las tareas de la visión por computador, cabe también mencionar, los procedimientos previos al procesamiento de imágenes, como es la mejora en la forma de lectura de la información que contiene la imagen, de esta forma se procede no solo a extraer la características sino a realizar un conjunto de técnicas asociadas al procesamiento de imágenes, como son, los *kernels* de pirámides espaciales o utilizando el Histograma de Gradientes Orientados (*HOG*) que consiste en describir la imagen desde distintas orientaciones a fin de obtener una función que represente la imagen de manera individual, es decir, se convierta en una firma para cada patron bajo análisis, de esta forma mejorar los características de interés en cada una de las imágenes en su respectiva clase y obtener el contenido más representativo de una imagen, esto ayuda mucho a los algoritmos de aprendizaje para potenciar su rendimiento, ya que será más información relevante por analizar.

Para algunos problemas de clasificación, teniendo en cuenta las salidas de un clasificador, se plantean mejoras al utilizar técnicas de fusión o combinación de clasificadores, es decir, que para un problema de clasificación, se debe estudiar el origen de los datos y sus características, por ejemplo, la clasificación de Ronchigramas puede presentar no solo la formación de franjas. sino otras variables que correspondan a la determinación de un error, como lo puede llegar a ser el desplazamiento, esto, visto desde otro modelo de la prueba Ronchi. Para el análisis geométrico, la visualización de los patrones, permite obtener la información relevante del tipo de defecto óptico, pero amerita mencionar, que utilizar la combinación de clasificadores se convierte en un tema para tratar y ayudar a mejorar los niveles de clasificación; además existen técnicas que ayudan a mejorar el rendimiento de un clasificador como en el caso del Random Forest en cuanto al desbalance de clases, tratándolo con una técnica que se

conoce como aprendizaje sensible al coste (*cost sensitive learning*)[39].*Mediante esta técnica, se asigna un determinado peso a cada una de las clases de tal forma que se penaliza más el clasificar mal los elementos de una clase que los de otra[26].*

El desplazamiento dentro del foco es un aspecto para obtener diferente número de franjas, haciendo que el patrón cambie y es un aspecto que no deja de convertirse en una opción para trabajar una subclase, que requiera un tratamiento especial de clasificación, dependiendo la distancia donde se capture el Ronchigrama, lo cual puede trabajarse con Redes Neuronales Convolucionales (*CNN-Convolutional Neural Network*), la cual es una técnica muy potente de aprendizaje automático, pero igualmente para que amerite este tipo de técnica se debe aumentar el numero de Ronchigramas a nivel de cientos de miles.

REFERENCIAS

[1] Malacara, D.: Optical shop testing. Volume 59. John Wiley & Sons (2007)

[2] Prieto, M., Forero, Á.R.: Diseño e implementación de un sistema asistido por computador de la prueba de Ronchi en el Taller de Óptica de la Universidad de los Llanos - SAPRULL. Universidad de los Llanos (2015)

[3] Moya, J.P.A.: Procesamiento y análisis de imágenes digitales. libro electrónico disponible en www. ie. itcr. ac. cr/palvarado/PAID/paid. pdf [consultada el 17 de julio de 2012] (2012)

[4] Kuncheva, L.I.: Combining pattern classifiers: methods and algorithms. John Wiley & Sons (2004)

[5] Everaldo Aguiar, R.J.: Data mining course cse - spring 2014, university of notre dame. Validation & Interpretation, Week 13 (4/7-4/13), Random forests, Ensemble methods (2014-2017)

[6] Friedman, J., Hastie, T., Tibshirani, R.: The elements of statistical learning. Volume 1. Springer series in statistics New York (2001)

[7] Toto-Arellano, N.I., Rodríguez-Zurita, G., Serrano-García, D.I., Hernández-Orduña, G.: Teoria Fisica de Rejillas de Amplitud y Fase. Noel Ivan Toto Arellano (2011)

[8] Valveny Ernest, González Sabaté Jordi, B.C.R.: Clasificación de imágenes: cómo reconocer el contenido de una imagen (2010)

[9] Bernard, S.: Random forests-parametrization and dynamic induction, document and learning research team litis laboratory university of rouen, france (2014)

[10] Ding, S., Chen, L.: Intelligent optimization methods for high-dimensional data classification for support vector machines. Intelligent Information Management **2**(06) (2010) 354

[11] Liu, M., Liu, X., Li, J., Ding, C., Jiang, J.: Evaluating total inorganic nitrogen in coastal waters through fusion of multi-temporal radarsat-2 and optical imagery using random forest algorithm. International Journal of Applied Earth Observation and Geoinformation **33** (2014) 192–202

[12] Jiménez López, A.F., et al.: Diseño de un sistema para la medición de potencia refractiva de lentes progresivas empleando el test de Hartmann Design of a refractive power measurement system of Progressive Addition Lenses by Using the Hartmann Test. PhD thesis, Universidad Nacional de Colombia (2011)

[13] Serway, R.A., Beichner, R.J., Jewett, J.W.: Physics for scientists and engineers with modern physics. Saunders College Publishing Philadelphia PA (2000)

[14] Hackeling, G.: Mastering Machine Learning with scikit-learn. Packt Publishing Ltd (2014)

[15] Bowles, M.: Machine learning in Python: essential techniques for predictive analysis. John Wiley & Sons (2015)

[16] scikit-learn developers: An introduction to machine learning with scikit-learn. An introduction to machine learning with scikit-learn (2010-2016)

[17] Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International journal of computer vision **60**(2) (2004) 91–110

[18] Howse, J.: OpenCV Computer Vision with Python. Packt Publishing Ltd (2013)

[19] Csurka, G., Dance, C., Fan, L., Willamowski, J., Bray, C.: Visual categorization with bags of keypoints. In: Workshop on statistical learning in computer vision, ECCV. Volume 1., Prague (2004) 1–22

[20] Cruz Roa, A.A., et al.: Anotación Automática de Imágenes Médicas Usando la Representación de Bolsa de Características. PhD thesis, Universidad Nacional de Colombia (2011)

[21] Witten, I.H., Frank, E., Hall, M.A., Pal, C.J.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann (2016)

[22] Joachims, T.: Training linear svms in linear time. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM (2006) 217–226

[23] Grauman, K., Darrell, T.: The pyramid match kernel: Discriminative classification with sets of image features. In: Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on. Volume 2., IEEE (2005) 1458–1465

[24] Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: Computer vision and pattern recognition, 2006 IEEE computer society conference on. Volume 2., IEEE (2006) 2169–2178

[25] Breiman, L.: Random forests. Machine learning **45**(1) (2001) 5–32

[26] Hidalgo Ruiz-Capillas, S., et al.: Random forests para detección de fraude en medios de pago. Master's thesis, Universidad Autónoma de Madrid, Escuela Politécnica Superior, Departamento de Ingeniería Informática (2014)

[27] scikit-learn developers: An introduction to machine learning with scikit-learn. Ensemble methods (2010-2016)

[28] MartÃŋnez G., V.: Metodología de Minería de datos contra el fraude empresarial. Facultad de Estudios Estadísticos (2015)

[29] Biau, G.: Analysis of a random forests model. Journal of Machine Learning Research **13**(Apr) (2012) 1063–1095

[30] Wyner, A.J., Olson, M., Bleich, J., Mease, D.: Explaining the success of adaboost and random forests as interpolating classifiers. arXiv preprint arXiv:1504.07676 (2015)

[31] Ruiz, F.E., Pérez, P.S., Bonev, B.I.: Information theory in computer vision and pattern recognition. Springer Science & Business Media (2009)

[32] Raschka, S.: Python machine learning. Packt Publishing Ltd (2015)

[33] Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Machine learning **63**(1) (2006) 3–42

[34] Cordero-Dávila, A., González-García, J.: Surface evaluation with ronchi test by using malacara formula, genetic algorithms, and cubic splines. In: International Optical Design Conference 2010, International Society for Optics and Photonics (2010) 76521F–76521F

[35] Cordero-Dávila, A., González-García, J., Robledo-Sánchez, C.I., Leal-Cabrera, I.: Local and global surface errors evaluation using ronchi test, without both approximation and integration. Applied optics **50**(24) (2011) 4817–4823

[36] Sánchez-Paredes, J., Silva-Ortigoza, G., Castro-Ramos, J., Sasian, J.: 1λ ronchi tester to obtain the wave front aberration in converging optical systems using phase shifting interferometry. Opt. Pura Apl **45**(4) (2012) 461–473

[37] Baquero, N., Hernández, W., Rincón, R.: Diseño y construcción de un telescopio cassegrain clásico. Orinoquia **10**(2) (2006) 7–12

[38] Kian Ho-hui.kian.ho@gmail.com, Gilles Louppe-g.louppe@gmail.com, A.M.a.b.: Oob errors for random forests. Scikit-Learn (2010-2016)

[39] Chen, C., Liaw, A., Breiman, L.: Using random forest to learn imbalanced data. University of California, Berkeley **110** (2004)