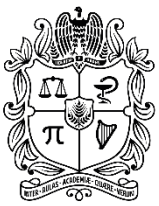


UNIVERSIDAD
NACIONAL
DE COLOMBIA

ANÁLISIS DE IMÁGENES HIPERESPECTRALES EN HOJAS DE MANGO EMPLEANDO LENGUAJE R.

Leonardo Xavier Ramón Poma

Universidad Nacional de Colombia

Facultad de Eléctrica y Electrónica

Bogotá, Colombia

2021

Trabajo final de Maestría

Leonardo Xavier Ramón Poma

Trabajo de investigación presentado como requisito parcial para optar al título de:

Magister en Automatización Industrial

Director (a):

Ph.D. **Flavio Augusto Prieto Ortiz.**

Línea de Investigación:

Espectrometría.

Universidad Nacional de Colombia

Facultad de Eléctrica y Electrónica

Bogotá, Colombia

2021

(Dedicatoria o lema)

El presente trabajo está dedicado a mis padres por darme la fuerza de seguir adelante a pesar de las adversidades y a nuestro maestro por enseñarnos la parte humana de ingeniería y su darnos nuevas armas para el ambiente laboral.

La preocupación por el hombre y su destino siempre debe ser el interés primordial de todo esfuerzo técnico. Nunca olvides esto entre tus diagramas y ecuaciones.

Albert Einstein

Lo mejor de cumplir un sueño es que sales tras del siguiente.

Flynn Rider

Agradecimientos

Quiero expresar mi gratitud a Dios, quien con su bendición llena siempre mi vida y a toda mi familia por estar siempre presentes. Mi profundo agradecimiento a todas las autoridades y personal que hacen la Universidad Nacional de Colombia por confiar en mí, abrirme las puertas y permitirme un desarrollo profesional y humano. A mis profesores por brindarme su tiempo y dedicación desde el primer día de clase, a mis compañeros Oscar Penagos y Juan David Valencia por liberar mis dudas y ser un apoyo, por darme el siguiente paso para salir adelante en momentos de falta de guía.

Finalmente quiero expresar mi más grande y sincero agradecimiento al PHD. Flavio Prieto principal colaborador durante todo este proceso, quien, con su dirección, conocimiento, enseñanza y colaboración permitió el desarrollo de este trabajo.

A mi esposa, a mis hijos, a mis amigos, todos aportaron con un poco a este trabajo.

Gracias…

Resumen

Análisis de imágenes hiperespectrales en hojas de mango empleando lenguaje R.

El Objetivo de este documento es presentar un análisis de las técnicas de clasificación para la identificación de la enfermedad conocida como la antracnosis en las hojas de mango de un repositorio de la universidad del año 2020, a través del procesamiento en lenguaje R. Primero se revela la importancia de lograr controlar la antracnosis en etapas temprana, a través de un estudio del arte de los mejores trabajos relacionados. Previo a la aplicación de técnicas de clasificación se presentó un preprocesamiento donde se realiza la lectura de las imágenes hiperespectrales, reducción de dimensionalidad, técnicas de normalización, obtención de bandas más relevante, organización en tres clases y subdivisión en conjuntos de pruebas y entrenamiento. Luego se estudia cinco técnicas de clasificación como son Máquina de vectores de Soporte (SVM), Análisis discriminante Lineal (LDA), Método de Potenciación de Gradiente (GBM), Bosques Aleatorios (RF) y Redes neuronales (NN), Finalmente se compara su desempeño y tiempo de entrenamiento del modelo.

Palabras clave: Antracnosis, Espectroscopia, Reflectancia, LDA, SVM, RF, NN, GBM.

Abstract

Hyperspectral imaging analysis in mango leaves using R

The objective of this document is to present an analysis of the classification techniques for the identification of the disease known as anthracnose in the mango leaves of a university repository of the year 2020, through processing in R language. First it is revealed the importance of achieving control of anthracnose in early stages, through a study of the art of the best related works. Prior to the application of classification techniques, a preprocessing was presented where hyperspectral images are read, dimensionality reduction, normalization techniques, obtaining the most relevant bands, organization into three classes and subdivision into test and training sets. Then five classification techniques are studied such as Support Vector Machine (SVM), Linear Discriminant Analysis (LDA), Gradient Potentiation Method (GBM), Random Forests (RF) and Neural Networks (NN), Finally their performance and training time of the model.

Keywords: Anthracnose, Spectroscopy, Reflectance, LDA, SVM, RF, NN, GBM.

Contenido

Lista de figuras

Índice de Tablas

Lista de Símbolos y abreviaturas

Abreviaturas

Abreviatura Términos

IA	Inteligencia Artificial
SVM	Máquina de vectores de soporte
RF	Random Forest
PCA	Análisis de componentes principales
LDA	Análisis discriminante Lineal
KNN	K vecinos más cercanos
GBM	Método Potenciación de Gradiente

1.Introducción

El análisis de la espectroscopia sobre las hojas de una planta nos provee diferentes variables que no se pueden apreciar a simple vista, con dichos datos, en ocasiones se puede percibir el estado actual de la salud de la planta. En nuestro caso de estudio se analizará si se puede detectar la enfermedad conocidas como antracnosis sobre las hojas de mango, fruto conocido por la comunidad colombiana como uno de los principales productos de exportación, por lo cual es primordial la inversión para garantizar una excelente calidad.

La antracnosis es un hongo de la variedad Colletotrichum que aparece con exceso de lluvias y climas húmedos, que afecta principalmente a tallos, hojas y frutos. Las manchas de apariencia mojada en el follaje o los frutos son los primeros indicios visibles. El tejido de las hojas sufre necrosis, adquiriendo una textura de papel y un color marrón a medida que la enfermedad se propaga. Los frutos que crecen a partir de las hojas que poseen la enfermedad se propagan en frutos adyacentes, de igual modo se propaga en arboles vecinos. Cuando la enfermedad ha sido identificada con cualquier método empleado se puede aplicar tratamientos preventivos a comienzos de la temporada y cuidar siempre las plantas a la primera señal de la enfermedad.

Las aplicaciones sobre una imagen hiperespectral se ha desarrollado en una infinidad de campos como la capacidad de detectar un patógeno intestinal como la Salmonella para distinguir un microorganismo Campylobacter de un no-Campylobacter, tarea muy complicada por ser muy parecidos. Esta tecnología es capaz de obtener un espectro de absorbancia por cada píxel de la imagen, al clasificar estos píxeles por su perfil composicional podemos agruparlos en diferentes categorías o clases.

Las longitudes de ondas ubicadas en el espectro visible al ojo humano están en el rango de 390 nm a 750 nm [1], en un rango de espectros ultravioleta hasta el infrarrojo-cercano. Las imágenes pueden ser clasificadas en 3 secciones según la cantidad de información espectral que ofrecen: las tres bandas BGR, azul comprendida entre 0,4 a 0,5 µm; verde comprendida entre 0,5 a 0,6 µm y rojo comprendida entre 0,6 a 0,7 µm, de esta forma es como se relacionan los colores base a las longitudes de onda; las imágenes multiespectrales están formadas por menos de 30 bandas espectrales; y las imágenes hiperespectrales con más de 30 bandas comúnmente en el orden de los cientos, al disponerse de forma continua en el espectro, ofrece una huella única llamada firma espectral.

Una firma hiperespectral es una identificación única por píxel en una imagen, que describe una curva a lo largo de las bandas que la componen, la firma espectral de las plantas verdes es muy característica. La clorofila de una planta en crecimiento absorbe la luz visible y especialmente la luz roja para usarla en la fotosíntesis.

La Figura 1.A se aprecia las firmas hiperespectrales tomadas de las muestras sanas de un mango, la Figura 1.B muestra un promedio de todas las firmas, delimitado por sus máximos y sus mínimos, mientras el resto de la imagen se observa el mango en sí. Las firmas hiperespectrales presentan valores que van desde los 350 nm hasta los 1900 nm y una reflectancia de 0,98.

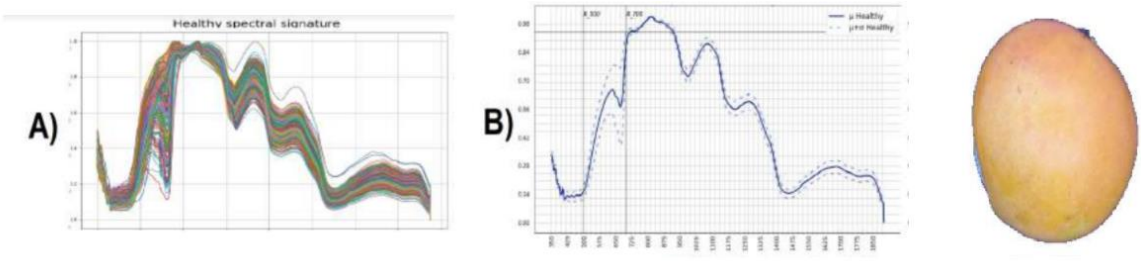


Figura 1 A. Firmas Espectrales de muestras sanas de un mango, B. Promedio de firmas espectrales. Tomada de 4

1.1 Imágenes hiperespectrales

Un sensor hiperespectral es capaz de trabajar en el campo óptico, el que incluye la región visible y la región infrarroja del espectro electromagnético, su función es crear imágenes hiperespectrales basándose en la proyección de rayos electromagnéticos [27], estos rayos electromagnéticos poseen diferente reflectancia al ser incidentes en diferentes materiales como se muestra en la Figura 2. Al ser creada una imagen hiperespectral gozará de gran información por sus componentes espectrales como espaciales. Su campo de estudio en los últimos años ha sido de gran aporte a la medicina, agricultura, minería y urbanismo.

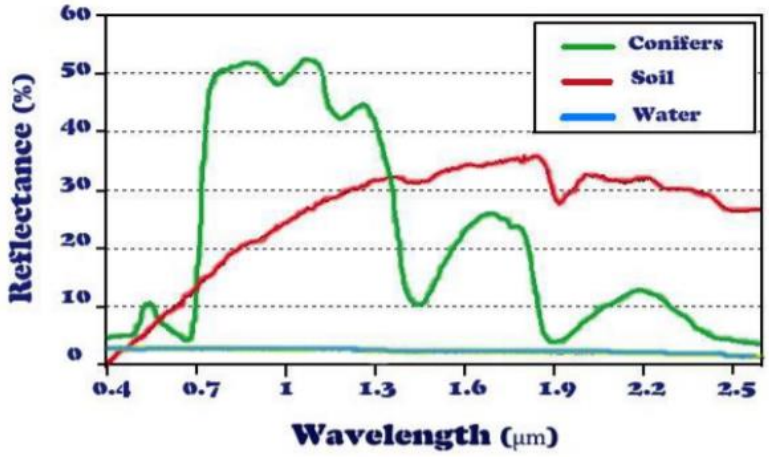


Figura 2. Firmas espectrales de diferentes materiales. Tomada de [26].

La radiación electromagnética tratas las diferentes formas que la energía viaja a través del espacio, de entre las diferentes fuentes de energías existen todas contienen la particularidad de se caracterizadas por su longitud de onda, cuya unida contiene un ciclo completo de onda y estas definida por la letra griega λ, relacionada directamente con la velocidad de la luz e inversamente con la frecuencia, la ecuación $c = \lambda v$ resumen lo antes dicho, teniendo en cuenta que la frecuencia será el numero de ciclos en una unidad de tiempo.

1.2 Cámaras Hiperespectrales

Las cámaras hiperespectrales son equipos que generan muchos más datos que un dispositivo normal de captura, posees sensores que pueden leer las longitudes de onda en el rango de los cientos, incluso fuera de espectro visible, en consecuente se obtiene un archivo de imagen de gran cantidad de información. En consecuencia, los sistemas de captura de imágenes hiperespectrales son los llamados espectrómetros que además de aportar información de una imagen normal capturan información química o física en tiempo real y con baja cantidad de error.

Una forma de analizar datos hiperespectrales se es a través de procesos quimio métricos, para llegar a la creación de clasificación confiable. Los datos que oporgan estas cámaras pueden ser usados tanto para clasificación como análisis de ingredientes. Su aplicación industrial comprende trabajos tanto en un entorno de laboratorio como las líneas de producción. Un modelo de cámara hiperespectral es mostrado en la Figura 3.

Figura 3. Cámara modelo HySpex VNIR-1800. Tomada de [13].

1.3 Software R

R es un lenguaje de programación que nació a partir del lenguaje S en Nueva Zelanda en el año de 1992, fue construido por los desarrolladores Robert Gentleman y Ross Ihaka [2]. Se compone de un conjunto de subprogramas dedicados a las operaciones como: manejo de datos, cálculos, graficas, simulaciones, y otros algoritmos más complejos como: organización de datos, desarrollo estadístico, creación de modelos, análisis de series de tiempo, clasificación, validación de modelos. Se recalca que al ser de código abierto se generan nuevas librarías en su repositorio oficial CRAM.

De la gran cantidad de librerías (actualmente disponibles más de 15303 librerías), se destaca la librería **hyperSpec** diseñada para proporcionar las herramientas para leer y procesar las imágenes hiperespectrales. El objetivo de hyperSpec es hacer que el trabajo con conjuntos de datos hiperespectrales (es decir, espectros resueltos espacialmente o en el tiempo, o espectros con cualquier otro tipo de información asociada con cada uno de los espectros) sea más cómodo.

Los espectros pueden ser datos obtenidos durante mediciones de espectroscopía XRF, UV, VIS, fluorescencia, AES, NIR, IR, Raman, NMR, MS, etc. De manera más general, cualquier dato que se registre sobre una variable discretizada.

¿Por qué utilizar R para este trabajo?

R es un lenguaje muy estructurado del cual se puedes citar las siguientes ventajas:

- Es rápido y preciso en la manipulación de datos para aplicaciones estadísticas.
- Se puede elaborar subprogramas o funciones para automatizar procesos sencillos: como leer archivos y realizar operaciones.
- Puede leer una amplia gama de archivos de entrada, como archivos .hyper y hdr.
- Es software libre.
- Tiene la capacidad de crear gráficos complejos de forma sencilla, fáciles de entender y de programar.
- Es un software multiplataforma.
- Detrás de R existe una comunidad que aporta constantemente a su mejoramiento, corrección de errores, creación de nuevas funciones y documentación completa, que se reflejan la satisfacción del usuario.
- Mejora su funcionalidad constantemente, ya que tiene detrás una comunidad bastante grande que crea nuevas funciones, corrige bugs y, sobre todo, documenta muy bien todo lo que va haciendo, de forma que la utilización de todas las funciones y métodos sea fácil a nivel de usuario.
- Está dividido en paquetes modulares que responden a necesidades específicas.

1.4 Objetivos

1.4.1 Objetivo General

Desarrollar un sistema de identificación que permita verificar la existencia de la enfermedad antracnosis a partir de imágenes hiperespectral de hojas de mango.

1.4.2 Objetivos específicos

- Desarrollar un estudio sistemático de algoritmos de extracción de características en imágenes hiperespectrales.
- Desarrollar un sistema para la extracción de las bandas que identifiquen la enfermedad de antracnosis en las hojas de mango.
- Evaluar el desempeño del sistema de identificación.

El documento se encuentra estructurado de la siguiente manera: En la sección 1.4 se ofrece una visión de los objetivos que se propuso alcanzar, el Capítulo 2 muestra un estudio profundo de trabajos referentes a detección y clasificación de enfermedades usando imágenes hiperespectrales. En el Capítulo 3 indica la metodología empleada para realizar la adquisición de imágenes hiperespectrales y se muestra el área sobre cual se va a trabajar, en el mismo capítulo se realiza un preprocesamiento y procesamiento de imágenes con técnicas revisadas en el estado del arte. En el Capítulo 4 se ofrece los resultados mediante gráficos comparativos y, finalmente, en el Capítulo 5 se realiza las conclusiones del trabajo y se plantea posibles trabajos futuros.

2. Estado del Arte

El mango proviene de las regiones sur asiáticas, el mismo posee gran capacidad de adaptación al clima, resistencia a estaciones y condiciones, por ello llego a América para convertirse en un producto de gran consumo humano y exportación comercial principalmente para países como México, Brasil, Cuba, Perú y Colombia. En Colombia existe la FEDEMANGO [1] creada en 2013 para trabajar a favor y desarrollo del sector manguicola, un sector de más de 18.500 hectáreas de cultivo, donde la gran mayoría no cuenta con el apoyo técnico de sus procesos de cultivo, fumigación o cosecha, allí es donde entra la FEDEMANGO, cuyos principales objetivos son promover, adelantar y gestionar el desarrollo de estudios e investigaciones sobre la producción del mango y cooperar en la divulgación y práctica de los resultados que ellos arrojen.

Dentro de los muchos estudios orientados a ofrecer un producto de calidad se encuentra la detección de enfermedades en etapas tempranas, una de las enfermedades más comunes y considerada como una de los problemas fitosanitarios [3] alarmantes es la antracnosis, un hongo del género Colletotrichum [2] (*Colletotrichum gloeosporioides, Colletotrichum acutatum, Colletotrichum boninense*), es una enfermedad que se da en las hojas y frutos, caracterizada por la coloración marrón alrededor de los nervios, que con el tiempo se hará más oscuras hasta llegar a la necrosis, es de fácil propagación al verse afectada por golpes o un desajuste nutricional, aprovecha estos signos de debilidad para multiplicarse y extenderse. La enfermedad disminuye la vida útil y limita la producción de la planta, la misma que ha causado pérdidas alrededor de 25% a 40% de la producción colombiana [6].

Algunas técnicas como [5] propone la evaluación de 5 fungicidas (Benomil (Benlate), Procloraz (Funcloraz), Mancozeb (Dithane M-45), Sulfato de cobre (Phyton 27), Extracto de Cítricos Libre de Nitrógeno (Lonlife)), con uso de pruebas in vitro con diferentes dosis. A pesar que los químicos aplicados cuentan con alto porcentaje de eficacia es un método

muy invasivo en la muestra aplicada por lo que afectaría a largo plazo el producto final, sería mejor una detección del patógeno no intrusivo para una eliminación localizada.

Técnicas de visión por computadora son no intrusivas y pueden detectar al patógeno en una etapa temprana, trabajos como [8] proponen la identificación por medio de técnicas de clasificación como Máquina de vectores de Soporte Multiclase aplicada con diferentes tipos de descriptores (RGB, TSL, HSV, MODELO Lab) sobre la imagen 2D, donde se obtiene resultados por encima de una exactitud de 69 % en el peor de los casos, identificando 4 clases: sanas, escala baja de esta enfermedad, mediana escala y alta escala de enfermedad. [9] propone una identificación más sencilla que intenta descubrir un tipo de característica muy específica, que consiste en encontrar vecindades de píxeles, los cuales permiten la caracterización del defecto antracnosis, en este caso en una fresa, obteniendo resultados de exactitud de 95 %.

La aplicación de imágenes hiperespectrales en la detección de enfermedades patógenas, es un campo innovador por la gran cantidad de información que esta imagen proporciona en un solo píxel, trabajos como [4] plantea la detección de la enfermedad en frutos de mango por medio de tres clases: sanas, inoculadas y enfermas, usando técnicas de reducción de dimensionalidad y selección de bandas más significativas obteniendo excelentes resultados en las bandas escogidas.

Las aplicaciones en el desarrollo de plantas son ideas frescas como lo es [20], donde se otorga un índice a una hoja de trigo para seguir su crecimiento, la gestión de fertilización y su desarrollo, por medio de un vehículo no tripulado (UAV) con una cámara hiperespectral incorporada, para su procesamiento se decidió usar cuatro métodos de extracción de bandas características y 3 clasificadores: mínimos cuadrados parciales

(PLSR), máquina de vectores de soporte (SVR) y aumento de gradiente extremo (Xgboost).

En el estudio [28] se analiza la presencia de materiales extraños en granos de café, con la finalidad de obtener un producto puro para el consumidor final, la técnica usada en mediante imágenes hiperespectrales para detectar y discriminar cuatro clases (madera, platico, piedras y residuos de plantas) dentro de 250 muestras de imágenes de café, se usaron tres modelos de clasificación, LDA, SVM y KNN, donde mejores resultados se obtiene el modelo de SVM con un 89.10%, donde se verifico el modelo obtenido con un conjunto de prueba externa.

Uno de los principales problemas que se pueda obtener al trabajar con imágenes hiperespectrales es su alto costo computacionales al procesar la gran cantidad de bandas requeridas, trabajos como [21] realizados sobre mediciones hiperespectrales aerotransportadas y mediciones de la conductividad eléctrica del suelo propone métodos de extracción de bandas características, tanto métodos supervisados como no supervisados. Se combina siete métodos no supervisados y tres supervisados obteniendo 7 bandas en el espectro rojo (600, 603, 636, 639, 642, 666 y 669 nm), 4 bandas en el espectro del infrarrojo cercano (738, 741, 744 y 747 nm) y 2 bandas cercanas al borde de los espectros azul y verde (498 y 501 nm), el trabajo mostró un proceso de obtención del número óptimo de bandas.

Un trabajo parecido es [4] donde se adquiere muestras de un mango y se intenta predecir si este posee antracnosis o esta enfermedad se encuentra en su etapa temprana, las condiciones de muestreo son en un ambiente controlado con excelentes resultados para la adquisición de imágenes hiperespectrales, a pesar que el mango posee superficie ovalada. Se emplea técnicas de reducción de dimensionalidad para trabajar con una

combinación lineal de las bandas más significativas siendo el proceso más eficiente y veloz.

Existen varios lenguajes de programación con funciones especiales para análisis de imágenes hiperespectrales, como Matlab [29], Python [30], R, etc. R es un lenguaje de gran potencial para procesamiento de datos hiperespectrales de mapas e imágenes satélites, pero no existen muchos estudios de cámaras más pequeñas o de ambientes controlados. R [10], además de ser un lenguaje libre, bajo una licencia GNU tiene soporte para diferentes plataformas como WINDOWS, LINUX, MacOS y Play Station 3, es un lenguaje dedicado al procesamiento de datos y cálculos estadísticos, con una sofisticada capacidad de mostrar resultados en gráficos. La funcionalidad de R está dada por sus paquetes modulares por lo que cada vez aparecen nuevas bibliotecas y actualizaciones periódicas, además de nuevos aportes de entusiastas a este lenguaje. R cuenta con soporte de librerías para el trabajo con imágenes hiperespectrales, además de herramientas para reducción de dimensionalidad, clasificación y gráficos para mostrar resultados.

3. Materiales y Métodos

El repositorio de imágenes hiperespectrales tomado para realizar el análisis fue elaborado por el autor [4], con una cámara hiperespectral Hyspex vnir 1800 en un ambiente de luz controlada Figura 4. En la carpeta encontramos tres clases llamadas control_groups (Sanas), inoculed_group (inoculadas) y Diseased_group (Enfermas).

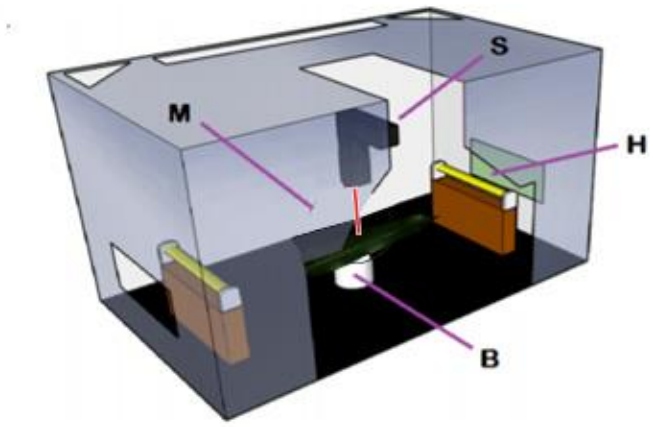


Figura 4. Escenario de adquisición de imagen hiperespectral con luz controlada. Editada de [4].

En el caso de la clase inoculada las muestras de hojas se etiquetan por días desde que se realizó la inoculación en puntos específicos, para elaborar una tabla de los píxeles oportunos se reunirá los 3 primeros días de inoculación es decir del día 24 al 26, días donde la enfermedad ya está en las hojas, pero aún no es visible, además estos puntos y

sus píxeles vecinos serán usados para tomar los puntos para el data frame de Inoculados.

La selección de píxeles se realizó manualmente sobre la muestra de la hoja, tal como se presenta en la Figura 5, en el caso de los puntos enfermos se seleccionan los píxeles en los que se aprecia la mancha marrón. Los puntos verdes son los píxeles sanos, en ambos casos extraemos la coordenadas [x,y], para posteriormente procesarla.

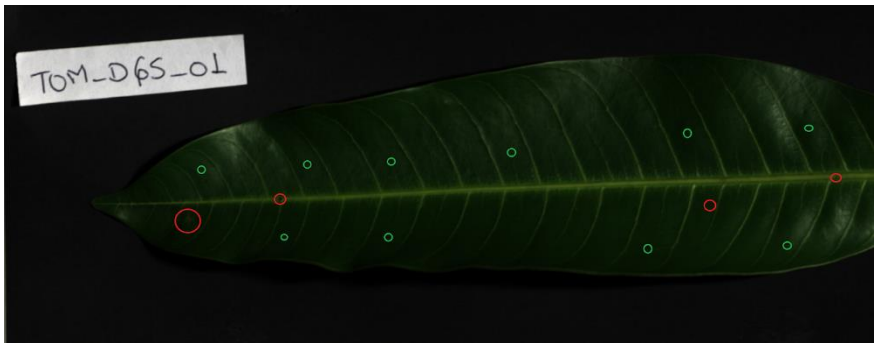


Figura 5. Rojo: Puntos con Enfermedad. Verde: Puntos sanos.

Se obtuvieron 240 puntos por cada clase, los valores obtenidos se observan en la Figura 6.

Hoja Sana	Hoja Inoculada	Hola Enferma
		

Figura 6. Hojas y sus diferentes clases.

Por cada imagen se obtiene un valor de 6080000 filas x 160 filas, que corresponden a representar las bandas o longitudes de onda, la imagen de cada hoja representa cerca de los 2 gigabytes de peso, que equivalen a un total de 972,800,000 datos a ser procesados.

De las muestras de píxeles tomadas se distingue 4 clases a simple vista: A Fondo Negro, B Fondo Blanco, C Antracnosis y D Hoja Sana. En el eje Y se encuentra la variable de la reflectancia, la cual puede variar por la luz incidente. En X se localizan las bandas espectrales que van desde la 400 nm hasta la 1000 nm como se muestran en la Figura 7. A y B son parte de la imagen, pero no representa un caso de estudio así que se descartan. El píxel con la enfermedad muestra un pico cerca de la banda 750 nm con un máximo inferior a las 7000 unidades de reflectancia, mientras un píxel sano en la misma banda llega a un valor superior a las 10000 unidades de reflectancia, además se puede observar un pequeño pico en la banda 720 nm. También, la zona entre 600 nm y 650 nm muestra una tendencia diferente en ambas clases que lo podemos usar para una futura diferenciación de características.

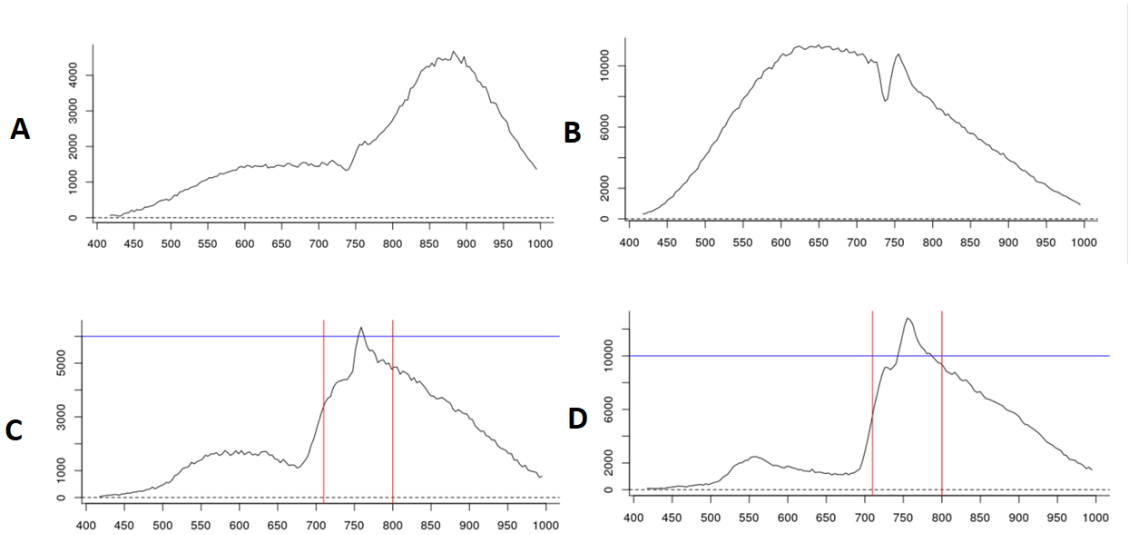


Figura 7. Firma hiperespectral de: A Fondo Negro, B Fondo Blanco, C Píxel con Antracnosis y D Píxel Sano.

Se realiza un estudio con ayuda del estado del arte basada en artículos científicos con los mejores trabajos sobre la extracción de bandas en diferentes campos de aplicación de las imágenes hiperespectrales, donde se puedan identificar algunas técnicas de

extracción de bandas y de reducción de la dimensionalidad como PCA, Boosting, luego comparar los mejores resultados.

La gran cantidad de información y procesamiento requiere de un equipo de cómputo de buen desempeño, por lo que se está usando un terminal DELL LATITUDE 7424 RUDDGET EXTREME, con 16 bg de RAM y procesador Core I7-8650U con una velocidad de 2.11 Ghz.

La metodología propuesta para el presente estudio involucra cuatro fases principales [16]: Datos: incluye la recolección de hojas de mango divididas en 3 lotes: sanas inoculadas y enfermas. Procesamiento de datos: incluye normalización de datos, división de conjuntos de datos con fines de entrenamiento y prueba, y desarrollo de algoritmos de modelos de SVM, RF, GBM, LDA y NN. Calibración de modelos: incluye optimización de modelos de ML mediante el ajuste fino con datos de entrenamiento. Validación de modelos: incluye validación de modelos de ML optimizados con conjuntos de datos de prueba, cálculo de métricas de error (precisión) y selección del mejor modelo de predicción. La Figura 8 muestra los pasos que se utilizan, mientras que la información detallada sobre los pasos se describe en los siguientes párrafos.

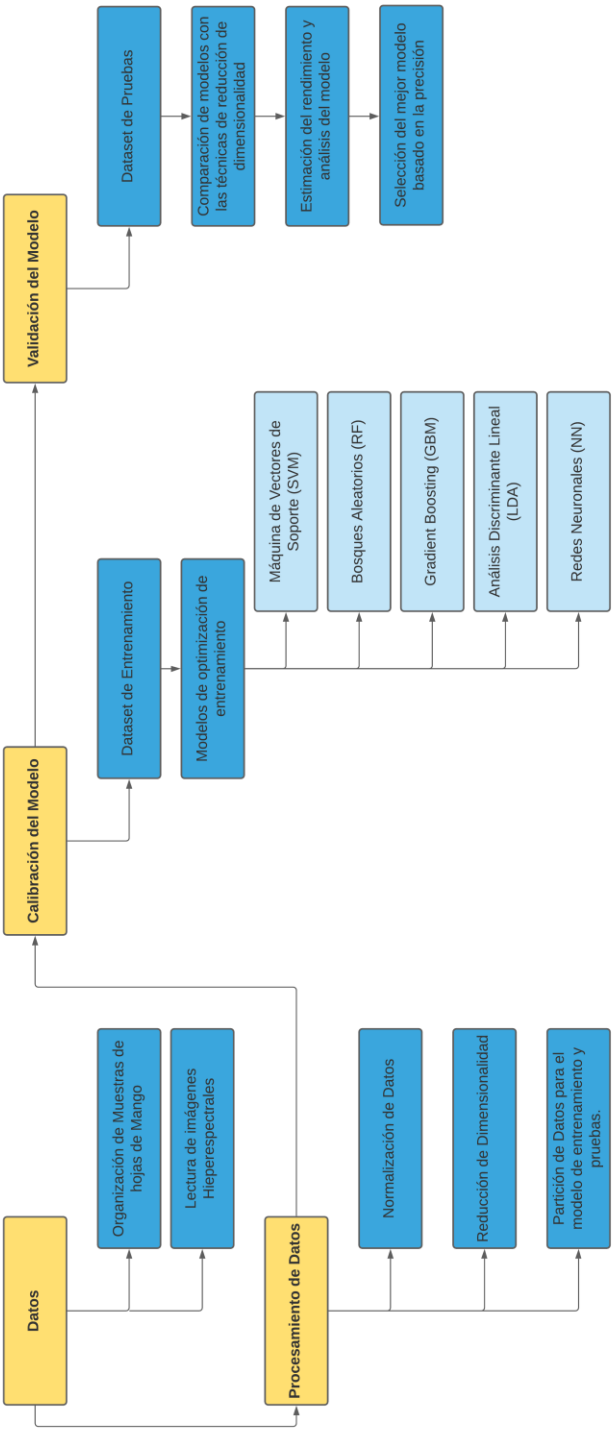


Figura 8. Esquema de metodología empleada en el procesamiento de datos.

Algunos conceptos que se estarán abordando a lo largo de la tesis se describe a continuación:

3.1 Medidas Estadísticas

En el análisis estadístico la población es un conjunto de elementos bien definidos, en el caso de una matriz hiperespectral vendría a ser las muestras de los píxeles y sus clases asignadas. Cuando pertenece un elemento de un conjunto de datos a una cantidad de variables (este caso las longitudes de onda o banda espectrales), dichas variables toman el nombre de multivariables. Los valores de la variable se transforman en un dataframe o en una matriz datos. Normalmente la matriz es rectangular (720 x 160 en este caso), compuesta de *n* filas por *k* columnas.

Sea la matriz de datos X y x_{ij}**,** donde ***i=1,..,n*** y ***j=1,….,k*** e indicaran la reflectancia de la banda j sobre la muestra i. La matriz de datos será de dimensión [n,k] y se presentara de la siguiente forma.

$$X = \begin{bmatrix} x_{11} & \cdots & x_{1k} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nk} \end{bmatrix} = \begin{bmatrix} x_1' \\ \vdots \\ x_n' \end{bmatrix} \quad (1)$$

donde ***x*** es una variable que se identifica como un vector fila de dimensiones k × 1, la cual representa los valores de las ***k*** variables con respecto al individuo ***i.***

3.1.1 Media

La media también es conocida como promedio, trata de establecer un punto intermedio de todos los valores existentes, es equivalente a la sumatoria de todos los valores a calcular sobre el número total de muestras. También es un dato necesario para hacer la comparación de diferentes conjuntos de datos.

$$\bar{x} = \begin{bmatrix} \bar{x}_1 \\ \vdots \\ \bar{x}_K \end{bmatrix} = \frac{1}{n} X' \tag{2}$$

3.1.2 Desviación Estándar

Representada por la letra σ, es un parámetro extraído de nuestro conjunto de datos que trata de establecer su dispersión respecto a la media de los mismo. El parámetro de desviación estándar es aplicado en el caso de cada variable individual o del conjunto total de datos.

$$\sigma = \sqrt{\frac{\sum_1^n (x_i - \bar{x})}{n}} \tag{3}$$

3.1.3 Matriz de correlación

Una matriz de correlación establece una comparativa de que tan están relacionadas son todas las variables de un conjunto total, los valores pueden llegar a tener el valor de -1 a +1, teniendo en cuenta que valores cercanos a uno denotan una mayor relación. Al elaborar la tabla de correlación, la diagonal principal será una diagonal de unos por la relación directa de una variable con si misma. También, si las variables tienen la tendencia de aumentar o disminuir, su valor de correlación será positivo.

$$R = \begin{bmatrix} 1 & \cdots & r_{1k} \\ \vdots & \ddots & \vdots \\ r_{n1} & \dots & 1 \end{bmatrix} = \begin{bmatrix} x_1' \\ \vdots \\ x_n' \end{bmatrix} \quad (4)$$

La nueva matriz **R,** una matriz semidefinida positiva, está relacionada directa a **S** por medio de la ecuación 5, en donde **D** es una matriz cuadrada diagonal de dimensión **k,** cuya diagonal principal están ubicados los valores de las desviaciones de las variables.

$$R = D^{-1}SD^{-1} \quad (5)$$

3.1.4 Matriz de covarianza

Definida por la letra ***S,*** es una matriz cuadrada nxn, que organiza en su diagonal principal la varianza de las variables del dataframe original y fuera de ella los valores de covarianza. Los valores positivos de covarianza fuera de la diagonal principal hacen referencia a la relación entre dos variables con valores por encima del promedio, de

forma contraria los valores negativos de covarianza indicaran los valores por debajo del promedio de ambas variables.

También, se debe tomar en cuenta que los valores del elemento de covarianza pueden ir desde menos infinito a mas infinitos, por lo cual no se emplea para ver la relación entre variables.

$$S = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x}) - (x_i - \bar{x})' \quad (6)$$

3.2 R Studio

En el procesamiento de grandes cantidades de datos, R ofrece diferentes paquetes fáciles de usar, y al ser de uso colaborativo se han ido perfeccionando estos algoritmos con el tiempo. Si se quiere usar para el Big Data se desenvuelve muy bien en ámbitos como manipulación, procesamiento y visualización gráfica de los datos. Por ejemplo, para presentar los resultados del procesamiento se puede crear visualizaciones de datos de alta calidad, crear dashboards para visualizar y analizar datos, crear informes automáticos o disponer de herramientas de análisis estadístico para ahondar en el

conocimiento de los datos. Para un mejor uso del lenguaje R se usará RStudio como plataforma de vinculación de R y análisis de datos.

El proceso de instalación es sencillo, se debe descargar del repositorio https://cran.rstudio.com/ el paquete de R correspondiente al sistema operativo que se usará. De igual forma se instala RStudio, un entorno de programació, el cual se debe descargar a través de un archivo ejecutable de este link https://www.rstudio.com/products/rstudio/download/, el mismo que brinda una interfaz amigable e intuitiva con el usuario. Ambos softwares son multiplataformas y de fácil instalación.

R Studio es una interfaz para usar R de una forma más fácil e intuitiva, la interfaz consta de 4 secciones que permiten un mejor orden y visualización durante la programación tal como se muestra en la Figura 9.

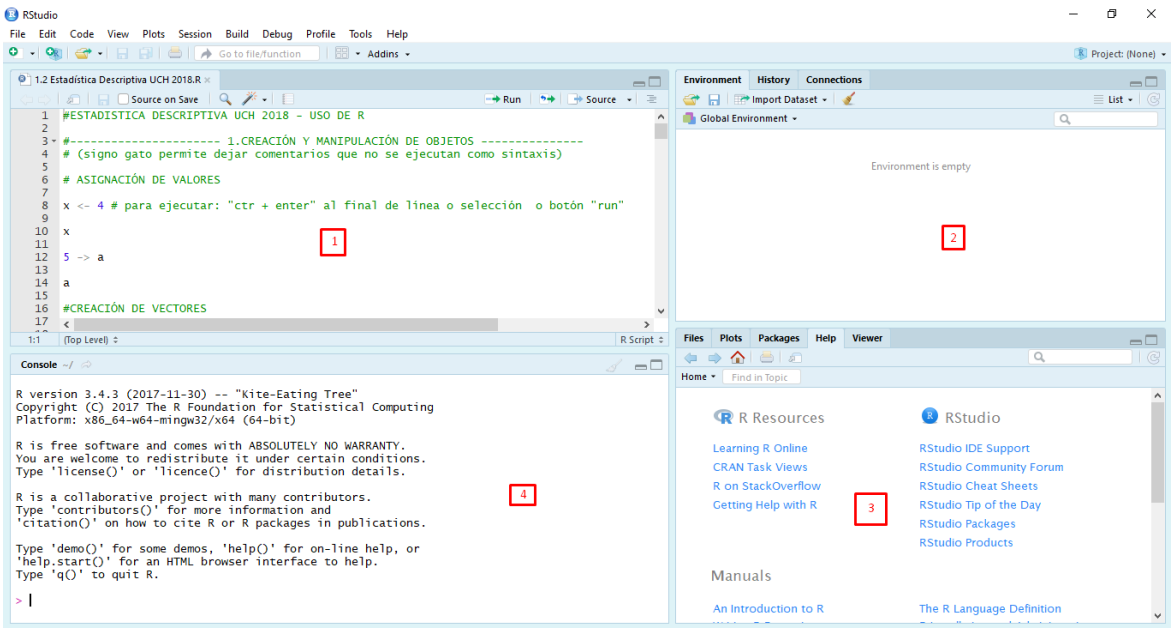


Figura 9 Interfaz del software RStudio.

Se localiza 4 ventanas de las cuales se encargan de:

- Ventana 1. Se encuentra el editor de sintaxis encargado de albergar el código de programación previa a la ejecución.
- Ventana 2. Se encuentra en entorno de trabajo donde se muestra el conjunto de datos y los diferentes tipos de objetos como resultados, variables, arreglos, frames, etc.
- Ventana 3. Se localiza una pequeña ventana portadora de varas pestañas, como: **files** muestra los archivos abiertos por R studio; **plots** donde se puede visualizar los gráficos y esquemas generados; **packages** donde se muestran los paquetes instalados y buscar nuevos paquetes para instalación; y finalmente **help** donde buscamos ayuda en el repositorio CRAN.
- Ventana 4. Se localiza la ventada de visualización de los resultados de ejecución del código, vendría a ser R en su versión básica.

Los comandos de ejecución en R son como se muestra en la Figura 10. El Objeto: es un nombre cualquiera elegido por el usuario. Asignadores: expresa todo lo que se encuentra a la derecha será guardado en el objeto. La Función: es la acción a realizar, este caso muestra una función de leer un archivo de Excel, dentro de la función se ofrece los argumentos para realizarla. El argumento 1: por lo general será los datos de ingreso, el resto de argumentos indicará configuraciones para la ejecución de la función.

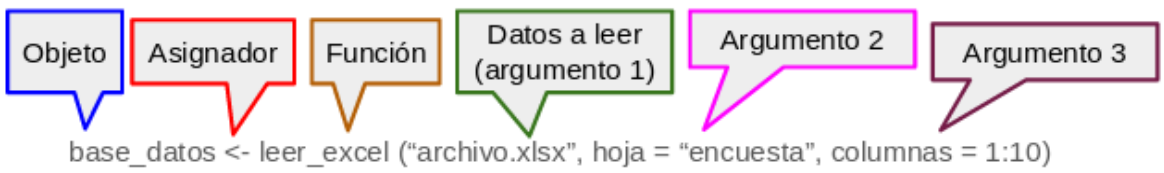


Figura 10. Línea de Comandos en R.

Aquí se usará las herramientas descritas anteriormente para atacar problemas de reducción de dimensionalidad y clasificación, un problema de clasificación consiste en tratar de predecir una variable categórica a partir de un vector aleatorio.

Es importante conocer el tiempo del procesamiento de los diferentes clasificadores, para compararlo al usar las bandas completas o las bandas reducidas con algunas técnicas de reducción de dimensionalidad planteadas. El comando usado en R es ***TicToc***, el proporciona las funciones de temporización tic y toc eso se puede anidar. Las llamadas ***tic*** y ***toc*** ponen en marcha el cronómetro que inicia con el tic y detiene el temporizador toc. El comportamiento predeterminado es imprimir un mensaje simple con el tiempo transcurrido en la llamada del toc. A continuación, se presenta un ejemplo de su uso:

```
tic()
Sys.sleep(0.5)
toc()
## 0.505 sec elapsed
```
(1)

3.3 Implementación y Desarrollo

En el presente capítulo se desarrolla el algoritmo en R para el procesamiento de las imágenes hiperespectrales, se comenzará con la lectura de la imagen, se continúa con la selección de las clases y se elabora un data frame de 3 secciones definidas, se procese a normalizar y a dividir en dos particiones el 70% pertenece al entrenamiento y el 30% a datos de prueba. Con los nuevos data frame creados se procederá a entrenar 5 tipos diferentes de modelos y se validará los mismo. Los algoritmos usados se adjuntan junto con las librerías que los contienen.

3.3.1 Lectura de Imágenes hiperespectrales

De las imágenes obtenidas en la Tabla 1 tenemos 5 muestras diferentes por cada clase, por cada imagen se tiene dos archivos con la extensión hyspec y. hdr con un tamaño cerca a los 2000 megabytes. Para leer este tipo de archivos se usará una librería llamada **hyperSpec**.

HyperSpec es un paquete R que permite el manejo conveniente de conjuntos de datos hiperespectrales, conjuntos de datos que combinan espectros con datos adicionales por espectro. Este paquete para R permite de forma muy cómoda trabajar con conjuntos de datos hiperespectrales, es decir. espectros resueltos espacialmente o en el tiempo, o espectros con cualquier otro tipo de información asociada a cada uno de los espectros. Los espectros pueden ser datos obtenidos en XRF, UV / VIS, fluorescencia, AES, NIR, IR, Raman, NMR, MS, etc. Más generalmente, cualquier dato que se registre sobre una variable discretizada, con una absorbancia = f (longitud de onda), almacenada como un vector de valores de absorbancia para longitudes de onda discretas.

La función ***read.ENVI*** permite la importación de datos ENVI como objeto hyperSpec, cuya función permite leer los archivos de la cámara hiperespectral usada y convertirla en un data.frame para su posterior uso. Teniendo en cuentra que los datos ENVI generalmente constan de dos archivos, un encabezado ASCII y un archivo de datos binarios (.hypex y .hdr) [14]. El encabezado contiene toda la información necesaria para leer correctamente el archivo binario.

```
hoja = read.ENVI("tom_dgs_01_16000_us_2x_2019-11-29T115006_corr.hyspex",
"tom_dgs_01_16000_us_2x_2019-11-29T115006_corr.hdr")
```
(2)

Para acceder a los parámetros internos de cada imagen hyperespec ofrece fácil acceso a través de los parámetros [15]:

- @wavelength que contiene un vector numérico con el eje de longitud de onda de los espectros.
- @data un data.frame con los espectros y toda la información adicional perteneciente a los espectros.
- @label una lista con etiquetas apropiadas (nombres de las bandas).

La Figura 11 muestra un píxel sano o libre de antracnosis, la gráfica muestra el promedio de todas las muestras adquiridas de píxeles sanos, además muestra la firma espectral mínima y la firma con la reflectancia mayor conocida como máxima.

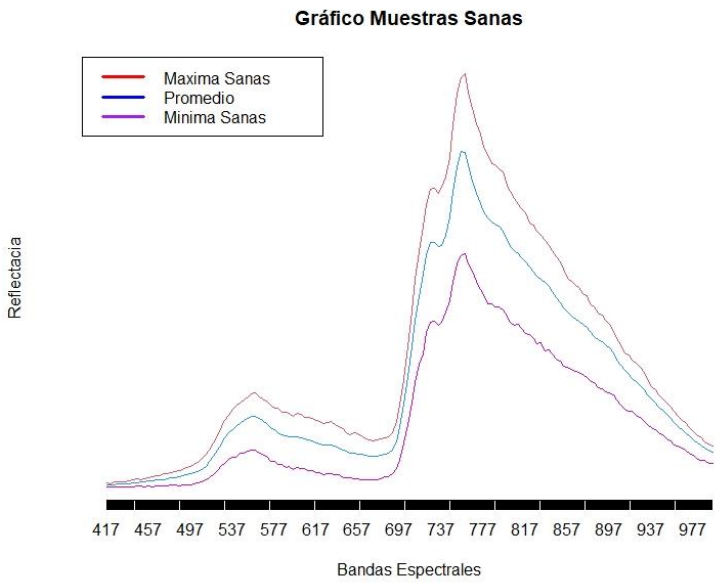


Figura 11. Promedio, máximos y mínimos de la clase de muestras sanas.

El promedio de las firmas hiperespectrales en este caso inoculadas, es la sumatoria de todas las reflectancias aplicando el cociente para el número total de muestras tomadas. Así se muestra en la Figura 12.

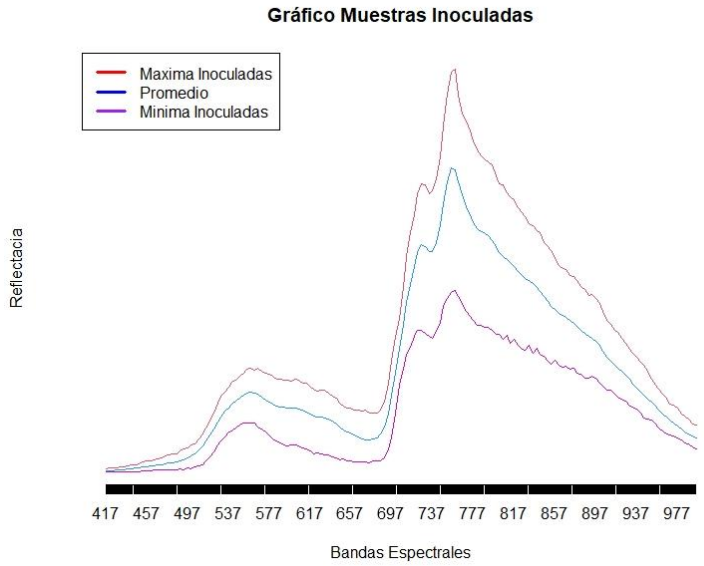


Figura 12. Promedio, máximos y mínimos de la clase de muestras Inoculadas.

Se aplica el mismo algoritmo con las muestras enfermas como se muestra en la Figura 13, su promedio delimitado por sus máximos y mínimos.

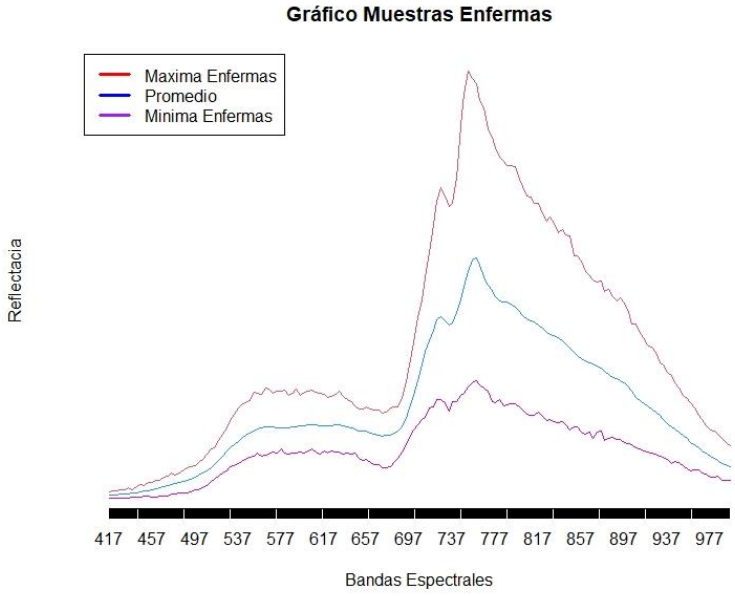


Figura 13. Promedio, máximos y mínimos de la clase de muestras Enfermas.

Una comprensión más general de las firmas espectrales por clase se realiza promediando las bandas obtenidas en cada clase, como se muestra en la Figura 14, en el eje de las abscisas se muestra las longitudes de ondas o bandas hiperespectrales, dejando de lado la reflectancia (eje de ordenadas) se puede apreciar como la diferencia en ciertos puntos es más prominente, teniendo tendencia a decrecer o aumentar unas más que otras, estas bandas se las puede apreciar como bandas que ofrecer un distintivo característico para su respectiva clase, bandas como 555, 559, 562, 566, 726, 730, 733, 737, 752, 756, 759, 785, 788 y 792, se las puede considerar como una bandas seleccionadas manualmente para un futuro procesa de clasificación.

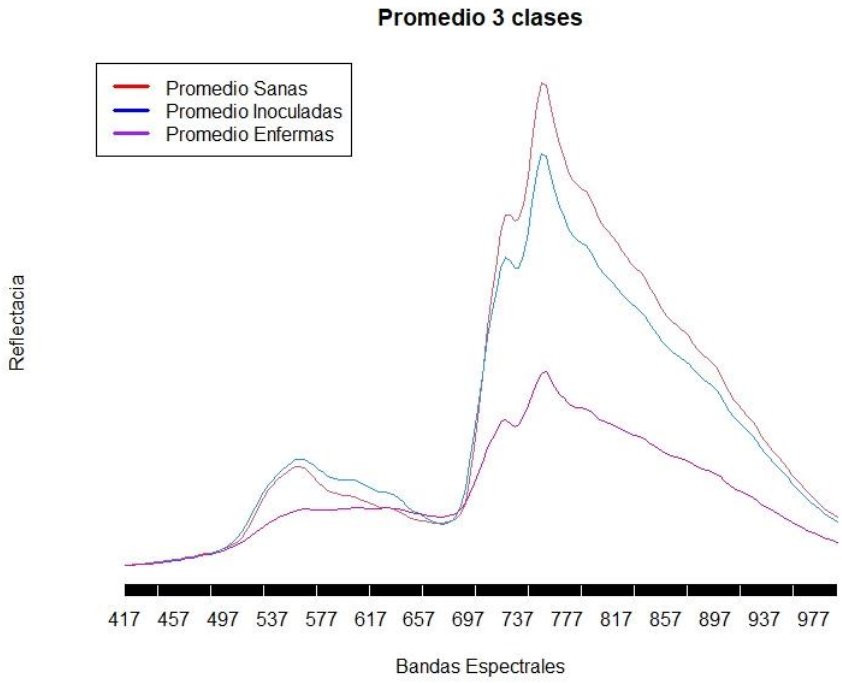


Figura 14. Promedio de las 3 clases por separado.

3.3.2 Selección de bandas

Al procesar una gran cantidad de información como lo es la dimensión de una imagen hiperespectral (160 bandas por n número de muestras) se consume recursos y tiempo, por lo cual, con base en el trabajo de [4], donde se demuestra que no es necesario usar dentro para el entrenamiento de los modelos de clasificación todas las variables (longitudes de onda), sino es escoger ciertas bandas que ofrecen características mejor definidas.

3.3.2.1 Selección manual de bandas

En la Figura 15 se muestra un promedio de las 3 clases en la cual se seleccionan las bandas que se consideran que ofrecen mayor contraste. Las bandas seleccionadas son 14, estas bandas se agrupan en un data frame para un futuro estudio de clasificación. Dichas bandas seleccionadas son: 555, 559, 562, 566, 726, 730, 733, 737, 752, 756, 759, 785, 788 y 792.

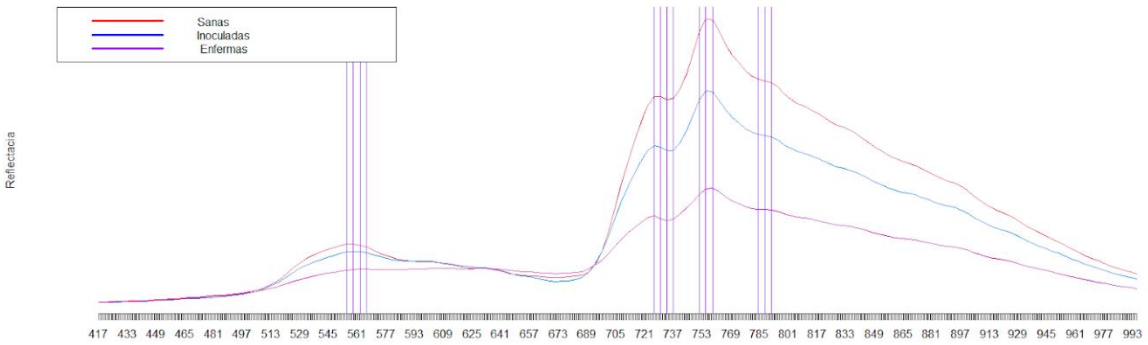


Figura 15. Selección de Bandas en los promedios Sanos y Enfermos.

3.3.2.2 Método de selección de Bandas automático

Otra forma de realizar la selección de bandas es mediante un modelo de regresión lineal. El modelo de regresión lineal optimiza el entrenamiento, ahorra recursos y logra una gran precisión con menos variables. Para validar el modelo que ofrece las mejores bandas se usa el parámetro ***AIC***. AIC o **A**kaike **I**nformation **C**riterion es definido por la ecuación 7:

$$AIC = -2 * logLik + k * n_{par} \quad (7)$$

Donde **logLik** corresponde al valor de log-verosimilitud del modelo y mientras la banda que contenga el valor mayor de **logLik** será considerada que ofrece un mejor modelo, ***k*** es un valor personalizado por el exceso de parámetros y n_{par} es el número de parámetros del modelo, con el resultado se puede decir que el mejor modelo es obtenido por las bandas que presentan AIC es de menor valor.

Existen dos clases de modelos para realizar la selección de variables: Forward donde una banda ingresa al modelo si su valor disminuye el AIC, y Backward donde una banda sale del modelo si su valor disminuye el AIC.

Método Forward

Se parte de un modelo sin variables explicativas o modelo vacío hasta un modelo completo con todas las 161 bandas, al realizar el primer paso se asigna a cada banda un AIC, se compara y se escoge la banda con el menor AIC, luego es agregado a la fórmula de modelo vacío y sigue con la siguiente iteración. El proceso se detendrá hasta que ya no pueda agregar bandas significativas. Como resultado se obtiene 70 bandas que explican el 73.95% del modelo original. Los valores de cada banda, así como sus parámetros AIC se muestran en el Anexo 1.

Implementación en R

```
rlm.fordward<-step(rlm.vacio,trace=TRUE,
                   scope = list(lower=rlm.vacio,upper=rlm.completo),
                   direction = "forward")
summary(rlm.fordward)
```
(3)

rlm.vacio es el modelo vacío, solo ingresados las clases.

rlm.completo: es el modelo creado a partir de las 160 bandas.

direction= Es el tipo de método a ejecutar.

trace: muestra las iteraciones, como se van agregando las bandas al bodelo.

El modelo final es el siguiente:

lm(formula = clase ~ `609.921` + `479.263` + `700.655` + `682.508` + `533.704` + `987.376` + `588.144` + `584.515` + `606.291` + `686.138` + `530.074` + `697.026` + `736.949` + `569.997` + `526.445` + `954.712` + `595.403` + `693.396` + `540.962` + `573.627` + `718.802` + `871.236` `824.054` + `972.859` + `457.487` + `537.333` + `885.754` + `903.901` + `805.907` + `678.879` + `490.151` + `453.857` + `428.452` + `889.383` + `976.488` + `802.278` + `809.537` + `983.747` `617.179` + `707.914` + `980.118` + `860.348` + `776.872` + `758.725` + `722.432` + `900.271` + `511.927` + `918.418` + `831.313` + `744.208` + `566.368` + `925.677` + `464.745` + `867.607` + `827.684` + `435.71` + `450.228` + `784.131` + `765.984` + `940.194` + `613.55` + `631.697` + `628.068` + `671.62` + `689.767` + `602.662` + `519.186` + `562.739` + `642.585` + `638.956`, data = matriz3clases)

Método Backward

Este método es lo contrario del método anterior, realiza un barrido partiendo del un modelo completo hacia un modelo vacío, y va quitando las bandas menos significativas o las bandas cuyo valor del parámetro AIC es el más bajo o ya no se puede quitar bandas menos significativas. Como resultado se obtiene 94 bandas que explican el 73.92% del modelo original. Los valores de cada banda, así como sus parámetros AIC se muestran en el Anexo 2.

Implementación en R

```
rlm.backward<-step(rlm.completo,trace=TRUE,
                   scope = list(lower=rlm.vacio,upper=rlm.completo),
                   direction = "backward")
summary(rlm.backward)
```
(4)

El modelo backward quedara representado por las siguientes bandas.

```
lm(formula = clase ~ `428.452` + `450.228` + `453.857` + `457.487` +
                     `461.116` + `464.745` + `475.634` + `511.927` + `519.186` +
                     `526.445` + `530.074` + `533.704` + `537.333` + `540.962` +
                     `562.739` + `566.368` + `569.997` + `573.627` + `584.515` +
                     `588.144` + `595.403` + `602.662` + `609.921` + `613.55` +
                     `617.179` + `628.068` + `631.697` + `642.585` + `671.62` +
                     `678.879` + `682.508` + `686.138` + `689.767` + `693.396` +
                     `697.026` + `700.655` + `707.914` + `722.432` + `736.949` +
                     `744.208` + `758.725` + `765.984` + `776.872` + `784.131` +
                     `802.278` + `805.907` + `809.537` + `824.054` + `827.684` +
                     `831.313` + `860.348` + `867.607` + `871.236` + `885.754` +
                     `889.383` + `900.271` + `903.901` + `918.418` + `925.677` +
                     `940.194` + `954.712` + `972.859` + `976.488` + `980.118` +
                     `983.747` + `987.376`, data = matriz3clases)
```

Método Stepwise

Es una combinación de los métodos forward y backward, Como resultado se obtiene 73 bandas que explican el 73.95% del modelo original. Los valores de cada banda, así como sus parámetros AIC se muestran en el Anexo 1.

Implementación en R

```
rlm.fordward<-step(rlm.vacio,trace=TRUE,
                   scope = list(lower=rlm.vacio,upper=rlm.completo),
                   direction = "both")
summary(rlm.fordward)
```
(5)

```
Call:
lm(formula = clase ~ `609.921` + `700.655` + `682.508` + `533.704` +
    `987.376` + `588.144` + `584.515` + `606.291` + `686.138` +
```

```
`530.074` + `697.026` + `736.949` + `569.997` + `526.445` +
`954.712` + `595.403` + `693.396` + `540.962` + `573.627` +
`871.236` + `824.054` + `972.859` + `457.487` + `537.333` +
`885.754` + `903.901` + `805.907` + `678.879` + `490.151` +
`453.857` + `428.452` + `889.383` + `976.488` + `802.278` +
`809.537` + `983.747` + `617.179` + `707.914` + `980.118` +
`860.348` + `776.872` + `758.725` + `722.432` + `900.271` +
`566.368` + `918.418` + `925.677` + `744.208` + `831.313` +
`784.131` + `867.607` + `464.745` + `450.228` + `940.194` +
`765.984` + `613.55` + `628.068` + `631.697` + `562.739` +
`671.62` + `689.767` + `642.585` + `602.662` + `519.186` +
`511.927` + `638.956` + `827.684`, data = matriz3clases)
```

El parámetro R cuadrado ajustado indica que cuando más variables se agregue (las nuevas bandas seleccionadas), más varianza explicará. Entonces, la mayor proporción de varianza explica el mejor modelo que se ha creado y está dada por la siguiente fórmula.

$$R^2 = 1 - \frac{\sum_0^i R_i^2}{\sum_0^i (y_i - y^*)^2} \quad (8)$$

donde $R_i$es el *i* -ésimo residual, donde *y** es la media de *y* si hay una intersección y cero en caso contrario. Los 3 métodos de extracción de bandas características producen resultados muy similares, las bandas seleccionadas en su gran mayoría son las mismas, así que se tomará el valor de R cuadrado ajustado de la Tabla 1 de mayor valor, ya que explicará el modelo total, y se creará como matriz de bandas seleccionadas a las obtenidas por el método de stepwize para futuro procesamiento.

R^2 Backward	R^2 Forward	R^2 Stepwize
0.7391526	0.7394537	0.7395232

Tabla 1. R cuadrado ajustado de los 3 modelos de selección de bandas.

3.3.3 Normalización

Al extraer todos los datos y organizarlos en una solo data frame, se debe normalizar para ajustar los valores medidos en diferentes escalas respecto a una escala común. Esta técnica permite **definir datos más naturales y limpios**, reduciendo su tamaño y simplificando la estructura para que los datos de producto sean más fáciles de localizar, contrastar y recuperar.

Es muy importante el proceso de normalización para evitar que una variable obtenga más importancia en un dataframe. R permite realizar este proceso a través de la siguiente línea de código.

```
var_ind_norm <- scale(var_ind)
```
(6)

El Proceso de normalización se realiza al reducir la dimensión de nuestro data frame por medio del comando *prcomp,* habilitando su parámetro *scale=TRUE.*

```
mi_pca <- prcomp(matriz3clases[1:160],scale. = TRUE,center = TRUE)
```
(7)

3.3.4 Implementación de técnica de reducción de dimensionalidad

La gran cantidad de bandas de la matriz de datos pueden lograr un resultado muy preciso al momento de entrenar el modelo de clasificación, pero tener muchas bandas no quiere decir un mejor entrenamiento por el contrario el crecimiento del modelo tenderá a disminuir por el exceso de variable, este fenómeno se conoce como “la maldición de la

dimensionalidad” [9]. Dicho en otras palabras, la densidad de la muestra disminuye exponencialmente con el aumento de más dimensiones.

Otro fenómeno es el sobreajuste que ocurre cuando el modelo puede corresponder demasiado a un conjunto de datos provocando un sobredimensionamiento hacia una clase en específico que derivará en el buen funcionamiento con los datos de entrenamiento, pero fallará con datos de prueba.

La reducción de dimensionalidad ataca estos dos fenómenos basándose en la eliminación de características y extracción de características.

- Eliminación de características: la cual consiste en eliminar variables redundantes o variables que no aportan ninguna información.
- Extracción de variables: consiste en formar nuevas variables a partir de las conocidas.

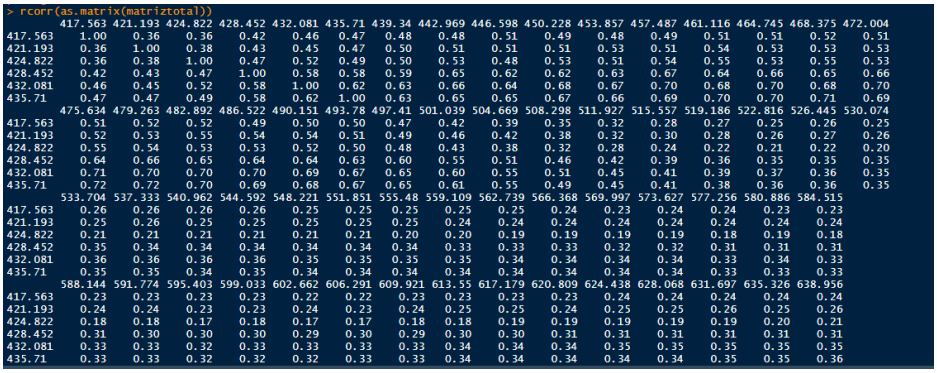
La ventaja de una nueva matriz aplicada al algoritmo de reducción de dimensionalidad es su facilidad de implementación, además, el nuevo conjunto de datos es más pequeño, sin perder información en el proceso.

3.3.4.1 **Análisis de la Correlación**

La matriz de correlación sirve para evaluar la relación de las variables de un data frame. Cuando este valor es alto indica que las variables están altamente correlacionadas y poseen las mismas características, por otro lado, si posee un valor bajo de correlación indica que no están correlacionadas en nada y tienen diferentes características. También, se puede decir que mide la relación lineal entre dos variables. Este parametro se mide con un coeficiente que va de -1 a 1, de la siguiente manera:

- $0 < r < 1$ la relación es positiva

- r = 0 no hay relación lineal
- -1 < r < 0 la relación es negativa
- r = -1 la relación es negativa perfecta.



```
> rcorr(as.matrix(matriztotal))
        417.563 421.193 424.822 428.452 432.081 435.71 439.34 442.969 446.598 450.228 453.857 457.487 461.116 464.745 468.375 472.004
417.563    1.00    0.36    0.36    0.42    0.46  0.47  0.48    0.48    0.51    0.49    0.48    0.49    0.51    0.51    0.52    0.51
421.193    0.36    1.00    0.38    0.43    0.45  0.47  0.50    0.51    0.51    0.51    0.53    0.51    0.54    0.53    0.53    0.53
424.822    0.36    0.38    1.00    0.47    0.52  0.49  0.50    0.53    0.48    0.53    0.51    0.54    0.55    0.53    0.55    0.53
428.452    0.42    0.43    0.47    1.00    0.58  0.58  0.59    0.65    0.62    0.62    0.63    0.67    0.64    0.66    0.65    0.66
432.081    0.46    0.45    0.52    0.58    1.00  0.62  0.63    0.66    0.64    0.68    0.67    0.70    0.68    0.70    0.68    0.70
435.71     0.47    0.47    0.49    0.58    0.62  1.00  0.63    0.65    0.65    0.67    0.66    0.69    0.70    0.70    0.71    0.69
        475.634 479.263 482.892 486.522 490.151 493.78 497.41 501.039 504.669 508.298 511.927 515.557 519.186 522.816 526.445 530.074
417.563    0.51    0.52    0.52    0.49    0.50  0.50  0.47    0.42    0.39    0.35    0.32    0.28    0.27    0.25    0.26    0.25
421.193    0.52    0.53    0.55    0.54    0.54  0.51  0.49    0.46    0.42    0.38    0.32    0.30    0.28    0.26    0.27    0.26
424.822    0.55    0.54    0.53    0.53    0.52  0.50  0.48    0.43    0.38    0.32    0.28    0.24    0.22    0.21    0.22    0.20
428.452    0.64    0.66    0.65    0.64    0.64  0.63  0.60    0.55    0.51    0.46    0.42    0.39    0.36    0.35    0.35    0.35
432.081    0.71    0.70    0.70    0.70    0.69  0.67  0.65    0.60    0.55    0.51    0.45    0.41    0.39    0.37    0.36    0.35
435.71     0.72    0.72    0.70    0.69    0.68  0.67  0.65    0.61    0.55    0.49    0.45    0.41    0.38    0.36    0.36    0.35
        533.704 537.333 540.962 544.592 548.221 551.851 555.48 559.109 562.739 566.368 569.997 573.627 577.256 580.886 584.515
417.563    0.26    0.26    0.26    0.26    0.25  0.25  0.25    0.25    0.25    0.24    0.23    0.24    0.24    0.23    0.23
421.193    0.25    0.26    0.25    0.25    0.25  0.25  0.25    0.24    0.24    0.24    0.24    0.24    0.24    0.24    0.24
424.822    0.21    0.21    0.21    0.21    0.21  0.21  0.20    0.20    0.19    0.19    0.19    0.19    0.18    0.19    0.18
428.452    0.35    0.34    0.34    0.34    0.34  0.34  0.34    0.33    0.33    0.33    0.32    0.32    0.31    0.31    0.31
432.081    0.36    0.36    0.36    0.36    0.35  0.35  0.35    0.35    0.34    0.34    0.34    0.34    0.33    0.34    0.33
435.71     0.35    0.35    0.34    0.35    0.34  0.34  0.34    0.34    0.34    0.34    0.34    0.34    0.33    0.33    0.33
        588.144 591.774 595.403 599.033 602.662 606.291 609.921 613.55 617.179 620.809 624.438 628.068 631.697 635.326 638.956
417.563    0.23    0.23    0.23    0.23    0.22  0.22  0.23    0.23    0.23    0.23    0.24    0.24    0.24    0.24    0.24
421.193    0.24    0.24    0.23    0.23    0.24  0.23  0.24    0.25    0.25    0.24    0.25    0.25    0.26    0.25    0.26
424.822    0.18    0.18    0.17    0.18    0.17  0.17  0.18    0.18    0.19    0.19    0.19    0.19    0.19    0.20    0.21
428.452    0.31    0.30    0.30    0.30    0.29  0.30  0.29    0.30    0.30    0.31    0.31    0.31    0.31    0.31    0.31
432.081    0.33    0.33    0.32    0.33    0.33  0.33  0.33    0.34    0.34    0.34    0.35    0.35    0.35    0.35    0.35
435.71     0.33    0.33    0.32    0.32    0.32  0.33  0.33    0.34    0.34    0.34    0.34    0.34    0.35    0.35    0.36
```

Figura 16. Tabla de Correlación de bandas en R Studio.

Comúnmente la diagonal principal presenta el valor de 1 porque es el máximo valor, lo que quiere decir que esta banda se relaciona con ella misma, tal y como se muestras en la Figura 16.

La Figura 17 muestra la relación entre las 160 bandas y que tanto dependen la una y la otra. Este proceso previo a la reducción de dimensionalidad ayuda a visualizar gráficamente como la Figura 15 que no todas las bandas aportan información trascendental y se puede reducir las bandas que tienen una correlación cerca de 1, o en el caso del gráfico que se aproxime a la zona azul.

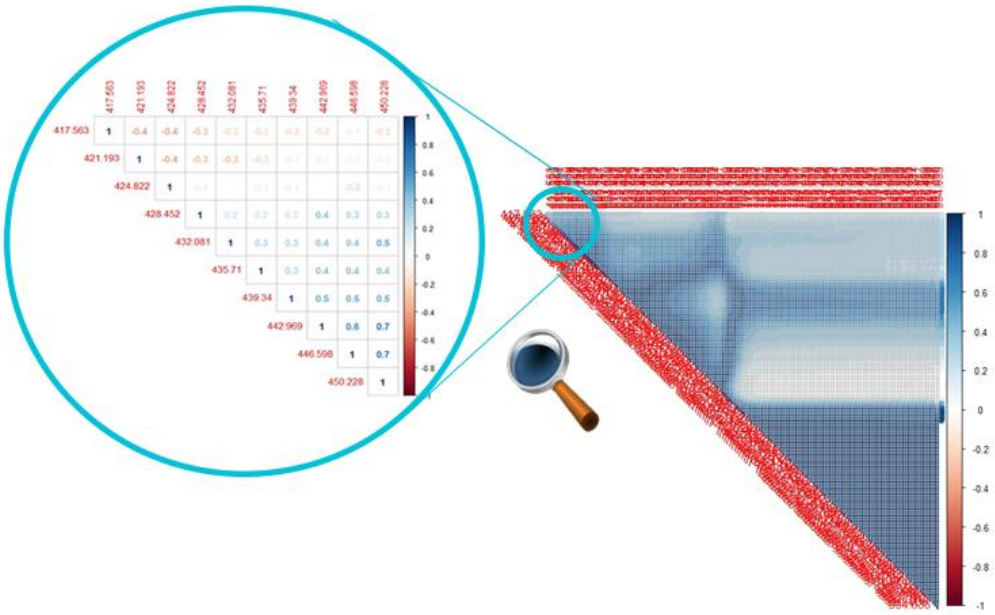


Figura 17. Esquema de Correlación entre bandas.

Podemos graficar con el comando corrplot. Lo primero es calcular la matriz de correlación y guardarla en un objeto y luego graficarlo. En este caso vamos a graficar los coeficientes.

Implementación en R

```
correlacion<-round(cor(matriztotal), 1)
corrplot(correlacion, method="number", type="upper")
```

3.3.4.2 **Análisis de componentes Principales (PCA)**

Es un método estadístico lineal con el cual se busca reducir la dimensión de un data frame y así acelerar su procesamiento, a la vez que se conserva la información de la muestra original [5].

En nuestro data frame original encontramos 160 variables que serán las 160 bandas de la imagen hiperespectral, lo que quiere decir que el espacio de muestreo tiene 160 dimensiones, **PCA** encontrará un espacio de fatores ***z*** menor a 160, tal que la nueva dimensión explicará lo más aproximadamente las variables originales, las nuevas variables ***z*** encontradas se conocen como componentes principales.

Cada componente **z** se obtiene al combinar linealmente las variables de la matriz original, de tal forma que, la primera componente principal tendrá la mayor varianza de las variables y se calculará del producto de la combinación lineal normalizada de dichas variables, dicho argumento se lo expresa en la Ecuación 9, donde Xn representa el valor original.

$$z_1 = \emptyset_{11}X_1 + \emptyset_{21}X_2 + \emptyset_{31}X_3 + \cdots + \emptyset_n X_n \tag{9}$$

Donde la normalización implicará que:

$$\sum_{j=1}^{160} \emptyset_{j1}^2 = 1 \tag{10}$$

Dicha normalización implica que la media sea cero, se procede a restar a cada valor de la media de la variable a la que pertenece. También se aplica el cálculo de vectores y valores propios sobre la matriz de covarianza de modo que se resuelva el problema de optimización que permita maximizar la varianza.

El comando *prcomp* de ***R*** realiza un análisis de componentes principales en la matriz de datos dada y devuelve los resultados como un objeto de clase, se debe añadir las 160 bandas sin incluir la clase como se indica a continuación:

```
mi_pca <- prcomp(matriz3clases[1:160],scale. = TRUE,center = TRUE)
```
(8)

El parámetro *scale=TRUE,* usa un valor lógico verdadero para indicar si las variables deben escalarse para tener varianza unitaria antes de que se lleve a cabo el análisis. Mientras el parámetro *center=TRUE* usa el valor lógico verdadero para indicar que las variables deben desplazarse para centrarse en cero.

```
summary(mi_pca)
```
(9)

El comando *summary* brinda un resumen completo del PCA que permitirá escoger los Componentes principales adecuados. La Tabla 2 nos muestra los 8 primeros componentes principales que será propuesto para la creación de un nueva data frame que será llevado a un proceso de clasificación futuro. En base a la proporción de la varianza podemos decir que el primer componente principal o PC1 explica el 62.97% del total de los datos y el segundo un 25.76%, juntos explicarían el 88.73% del total de los datos, como se muestra en la proporción acumulada.

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8
DESVIACIÓN ESTANDAR	10.037	6.4205	3.09559	1.26164	0.81708	0.8001	0.78175	0.71869
PROPORCIÓN DE VARIANZA	0.6297	0.2576	0.05989	0.00995	0.00417	0.0040	0.00382	0.00323
PROPORCIÓN ACUMULADA	**0.6297**	**0.8873**	**0.94720**	**0.95714**	**0.96132**	**0.9653**	**0.96914**	**0.97237**

Tabla 2. Componentes Principales

La Figura 18 es otra forma de visualizar la importancia de los componentes principales.

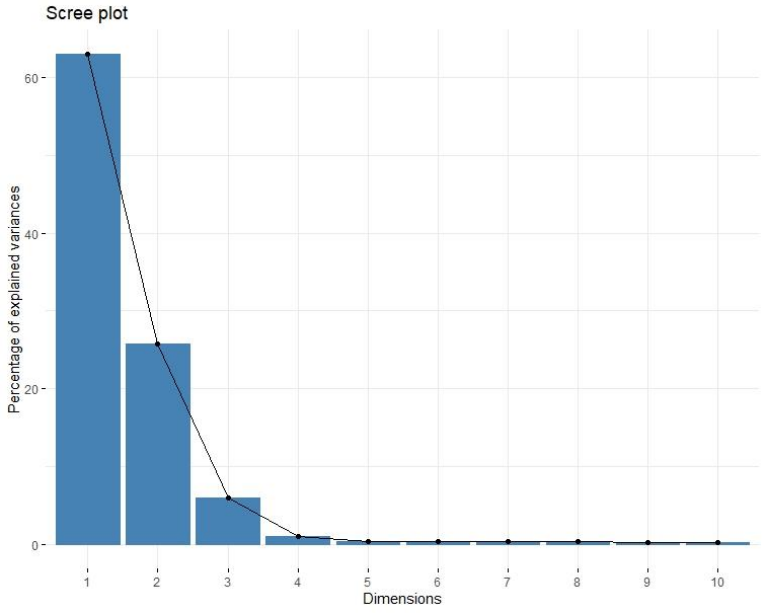


Figura 18. Diagrama de Barra de componentes principales.

Se crean un total de 160 nuevos componentes principales que son el resultado de la combinación lineal de las 160 bandas originales, según un análisis el componente PC1 tiene un 0.6297% de la varianza total lo que se puede interpretar como un 62.97% del componente de los datos. Por lo tanto, se tratará obtener un mayor porcentaje, como indica la Tabla 2. Componentes Principales en la parte de PROPORCIÓN ACUMULADA se logra un 97.23% hasta el PC8, por lo que se puede resumir la tabla general con los ocho primeros componentes principales.

La Figura 19 muestra los dos primeros componentes principales, que representan el 88.73%: PC1 con un 62.97% y PC2 con un 25.76% de la variabilidad de los datos, de las 3 clases existentes, se observa separado la clase inoculadas mientras las sanas/enfermas más unidas, de esta forma se consigue una apreciación de la separación de clases para un posterior proceso de clasificación.

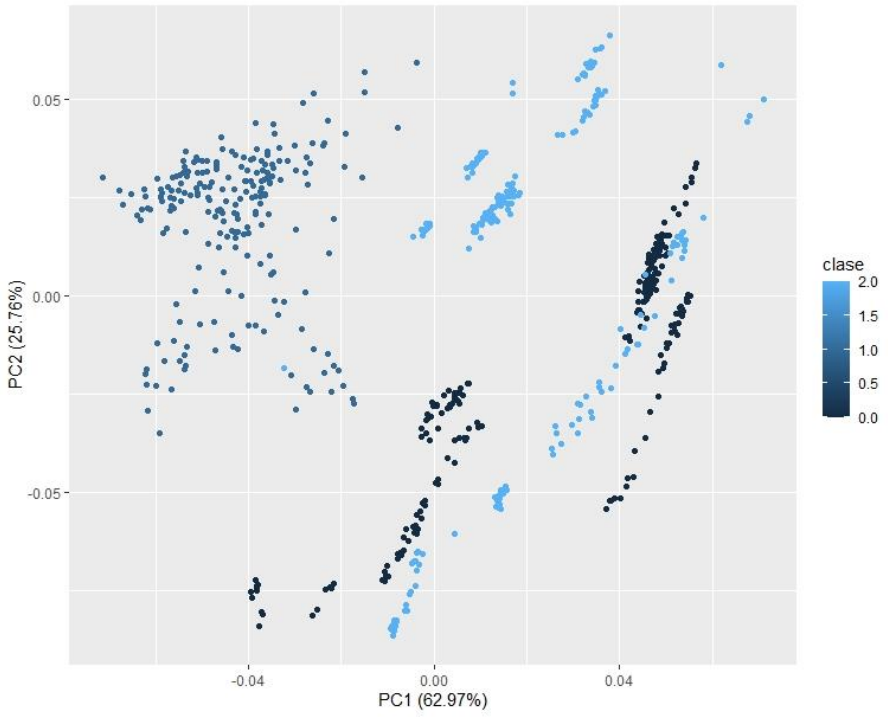


Figura 19. PC1 vs PC2

3.4 Implementación de técnicas de clasificación

Un sistema de clasificación propone un modelo que depende de una entrada, se procesa en una función entrenada y otorga una salida validada, se puede citar algunos métodos de clasificación estadísticos o inteligentes como: como análisis discriminante, regresión logística, redes neuronales, arboles de decisión o redes bayesianas, cualquiera de los citados cumple un objetivo final de identificar un parámetro entrenado que son tareas propias de la clasificación.

Previo a crear el modelo de clasificación se procede a dividir el conjunto de datos de firmas espectrales en dos subconjuntos: un conjunto de entrenamiento equivalente a un 70% y un subconjunto de validación correspondiente al 30%, el cual nos servirá para comprobar el modelo estimado. Cada fila de nuestra base de datos debe pertenecer a un subconjunto de forma aleatorio para lo cual se emplea el algoritmo (10).

```
#3.  dividir en data de entrenamiento y data de testeo
ind<-sample(2,nrow(df_norm),replace=TRUE,prob=c(0.7,0.3))
testset<-df_norm[ind==2,]
trainset<-df_norm[ind==1,]
trainset
testset
```

(10)

3.4.1 Máquina de vectores de soporte (SVM)

Máquina de Vector Soporte es un método de clasificación binaria propuesto es los años 90, actualmente soporta problemas de regresión lineal y clasificación de varias clases, incluso problemas de datos hiperespectrales, en este caso para clasificación de 3 clases.

Se le considera como un método de aprendizaje supervisado que desarrolla métodos de relación entre problemas de clasificación y regresión. El cual consiste en una matriz p-dimensional a los cuales cada fila tiene asignada una categoría de modo que el algoritmo SVM se encarga de establecer un modelo capaz de predecir si una nueva fila (con categoría desconocida) pertenece o no a otra categoría.

El proceso de modelado parte de un banco de entrenamiento que es etiquetado con el número de las clases pertinentes, dichas muestras son clasificadas y se separan de la manera más discriminante posible, luego, al llegar nuevas muestras o conjunto de pruebas serán ubicadas en la clase a la que pertenezca.

Encontrar la agrupación óptima es el fuerte de SVM, el cual encuentra un hiperplano que representa la distancia máxima entre las clases, por lo que también se le conoce como clasificador de margen máximo. Por lo mismo, las categorías se encontrarán separadas a ambos lados del hiperplano, como se muestra en la Figura 20.

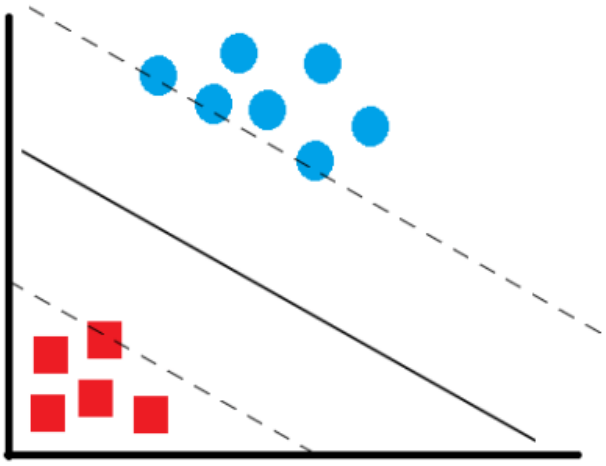


Figura 20. Esquema de apreciación de clasificación de máquina de vectores. Tomada de (7).

Una manera simple es la separación por medio de una línea recta, sino es suficiente se separa por medio de un plano recto, si no es suficiente se usa un plano N-dimensional, como seria en la mayoría son casos reales. En estos casos se emplean curvas no lineales para encontrar el nuevo kernel de separación, esto a su vez conlleva mayor cálculo computacional.

Las librerías en R que soportan las diferentes formas de emplear el modelo SVM son: e1071 y LiblineaR, aportan tanto modelos de regresión lineal como clasificación múltiple. El modelo resultante posee un kernel de base lineal, pero puede configurarse de forma para ser sigmoide o polinomial.

```
modelSVM<-svm(clase~.,data=trainset,kernel="polynomial")
```
(11)

3.4.2 Bosques Aleatorios (RF)

Consiste en un método de clasificación no lineal y algoritmos de regresión. Se define a un bosque aleatorio como una colección de árboles de decisión que busca las categorías con una probabilidad mayor, ya que los árboles carecen de precisión, se logra mejor desempeño sobre ajustando sus valores.

Es un algoritmo superior a la clasificación por árboles de decisión, o se puede decir que es un enfoque de alto rendimiento a este método, utiliza el método de ensacado de árboles para aumentar la precisión del modelo de aprendizaje. ¿Como lo consigue? Buscando las mejores características de un conjunto aleatorio de árboles, proporcionando esta propiedad de aleatoriedad al modelo para obtener un mejor modelo y más precisión.

Dentro de la clasificación se determina como el aprendizaje supervisado en proporción a las características proporcionadas, dicho de otra manera, se categoriza los datos estructurados o no estructurados en clases.

Se considera un árbol de decisión como un clasificador fiable, está compuesto por las partes normales de un árbol: nodos, hojas y ramas, donde los nodos equivalen a los datos de entrada, las hojas serian la salida de la clasificación o las clases propiamente dichas, mientras las ramas son los valores que las variables de entrada pueden tomar. El primer nodo será la variable de mayor relevancia para el clasificador, de donde partirá las siguientes etapas, a partir de aquí se puede llegar a existir un sobre ajuste, que consiste en el entrenamiento del modelo a partir de datos incoherentes, se puede evitar podando el árbol de decisión mediante la modificación del algoritmo de aprendizaje, el proceso se muestra en la Figura 21.

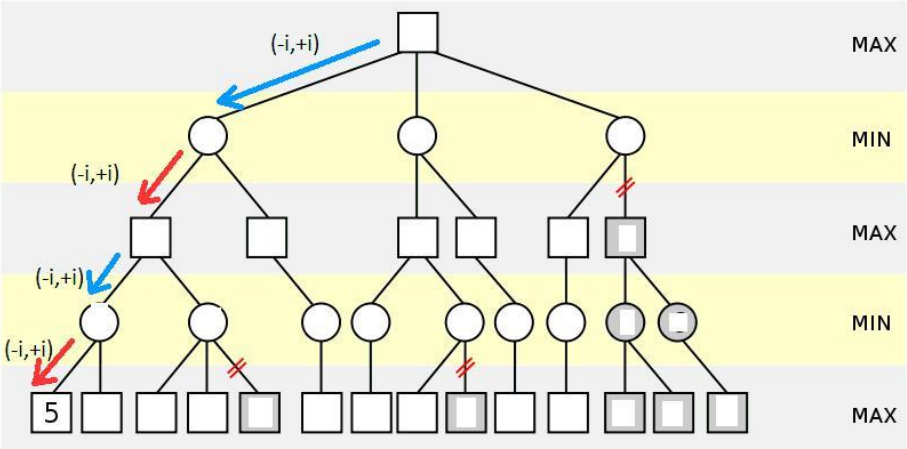


Figura 21. Esquema de funcionamiento de un árbol de decisión. Tomada de (7).

Para establecer el límite de crecimiento del árbol se establecen reglas cuyo objetivo es detener el crecimiento de una rama de acuerdo a las siguientes decisiones y reducir la complejidad del mismo:

- *Pureza de nodo*. Cuando un nodo solo pertenece a una única clase, por lo cual el crecimiento se da por finalizado.

- *Cota de profundidad*. Se detiene el proceso al alcanzar un valor de cota previamente establecido, que limita la profundidad del mismo.

- *Umbral de soporte*. Se conoce como parámetro de fiabilidad, se establece como un sinónimo de valor mínimo de los nodos, al llegar el nodo a este valor mínimo se detiene el proceso.

.

La poda del árbol se basa en un a algoritmo de poda pesimista, que consiste en encontrar un error llamado error de sustitución, que previene los casos de clasificación incorrectos, eliminando los subárboles que no tiene mucho aporte y coste-complejidad que se encarga de equilibrar el tamaño del árbol y la precisión.

El paquete de R *randomFores()* ofrece una gama de alternativas para crear y analizar bosques aleatorios, así como clasificación y generación gráficos de inferencia.

```
mod<-randomForest(x=matriztotal[training.ids,1:160],
                  y=matriztotal[training.ids,161],
                  ntree = 1,
                  keep.forest = TRUE)
```
(12)

Adicionalmente existe otra librería llamada “rpart” y “rpart.plot” que ofrece una visualización de las decisiones de cada árbol con un gráfico sencillo, graficada con su propio modelo. El resultado se muestra en la Figura 22:

```
Install.packages("rpart.plot")

library(rpart.plot)

modeloArbolDecicion <- rpart(class~.,trainset,method = "class")

prp(modeloArbolDecicion, type = 2, extra = 102)
```

(13)

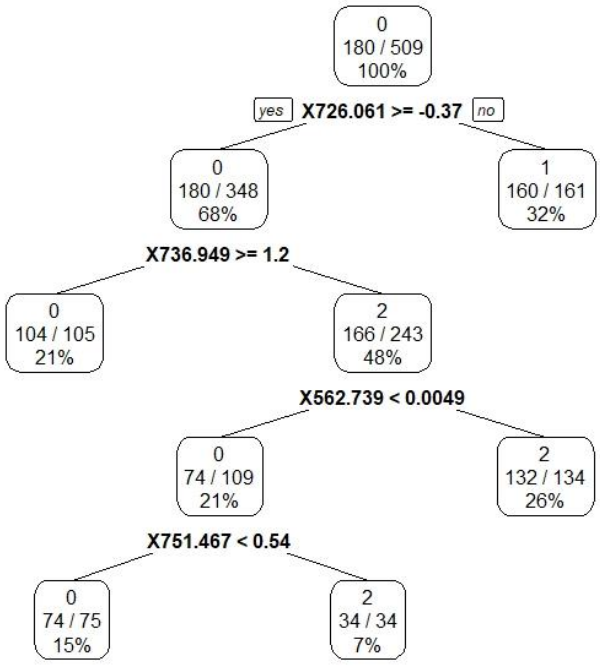


Figura 22. Toma de decisión de un modelo Randon Forest

3.4.3 Gradient Boosting (GBM)

El algoritmo del modelo Gradient Boosting está diseñado en base de un Random Forest pero de forma más compleja, se conforma de varios árboles de decisión que se entrenan de forma individual y secuencial, el error generado en el primer árbol es corregido en el siguiente, y así sucesivamente, de tal forma que el modelo ya entrenado se compone de las predicción individual de cada árbol.

Los resultados obtenidos con este método son muy buenos por lo cual se cataloga como un método referente para el análisis predictivo, además posee amplios casos de aplicación.

El objetivo es el mismo de los modelos anteriores, aunque el algoritmo tiene algunas variaciones, se enfoca en el entrenamiento secuencial para llevar el error a un nuevo ajuste. Se procede a ajustar el parámetro *f1* weak *learner* para tratar de encontrar la predicción de la variable *y*, para luego encontrar los residuos de la forma *y–f1(x)*. Después se ajusta un nuevo modelo, que intenta predecir los residuos del modelo anterior, corrigiendo los errores del modelo.

$$f_1(x) \approx y$$
$$f_2(x) \approx y - f_1(x) \tag{11}$$

Si *f1* y *f2* no son capaces de corregir los errores se seguirá ajustando con un *f3* hasta reducir al máximo el error:

$$f_3(x) \approx y - f_1(x) - f_2(x) \tag{12}$$

El proceso se realiza en un bucle de M iteraciones hasta conseguir que el error sea minimizado con respecto al modelo anterior.

Para evitar el overfitting, ya que el proceso de minimización de residuos es iterativo, se ajusta el valor de learning *rate* ($\boldsymbol{\lambda}$), conocido como parámetro de regularización, que limita la influencia de cada modelo en el conjunto del *ensemble*, en consecuencias se crearan más modelos, pero se obtendrá óptimos resultados.

$$f_3(x) \approx y - \lambda f_1(x) - \lambda f_2(x)$$
$$y \approx \lambda f_1(x) - \lambda f_2(x) - \lambda f_3(x) + \cdots + \lambda f_m(x) \quad (13)$$

Implementación en R

```
boost=gbm(class~ . ,data = trainset,distribution = "gaussian",
                                    n.trees = 10000,
                                    shrinkage = 0.01,
                                     interaction.depth = 4)
```
(14)

3.4.4 Análisis discriminante Lineal (LDA)

El análisis discriminante lineal fue introducido por Fisher en el 1936, es un método de clasificación supervisado que buscar encontrar variables dependientes partiendo de la combinación lineal de variables independientes. El objetivo del análisis discriminante lineal es descifrar las características que más diferencian a un grupo y predecir a que clase pertenece.

El análisis discriminante lineal admite solo variables cuantitativas para su desarrollo, por lo que si alguna variable independiente es categórica no se podrá aplicar su algoritmo. El proceso de ejecución se lo puede realizar de dos formas:

- Encontrar la función discriminante.
- Empleando técnicas de correlación canónica y componentes principales, lo que se conoce como análisis discriminante canónico.

La primera forma es la más usada, donde por medio de la combinación lineal de variables explicativas, formará funciones lineales que intenten explicar nuevas variables y donde existirá una función de más valor a la cual pertenezca de forma más probable. La combinación lineal obtendrá una función Z resultante de la combinación lineal de las variables p, de tal forma se puede escribir la ecuación de Fisher de la siguiente manera:

$$Z_i = \beta_0 + \beta_{1}X_{1i} + \beta_{2}X_{2i} + \ldots + \beta_{p}X_{pi} \quad (14)$$

donde,

- Z_i valor de Z discriminante para un objeto i.
- β_0 constante.
- β_j valor discriminante de la variable independiente j.
- X_{ji} valor de la variable independiente j para el objeto i.

Entonces la función Z existe para conseguir mayor distancia entre los centroides, de tal manera que se pueda conseguir la mayor separación entre grupos. Podemos expresar esta distancia de la siguiente manera:

$$h = \bar{z}_I - \bar{z}_{II} \quad (15)$$

En la Figura 23 se tiene la función Z que intenta dividir los dos centroides de los grupos G1 Y G2 de la mejor forma posible, pero si los grupos no difieren en las variables independientes, no podrá encontrar una dimensión en la que los grupos difieran:

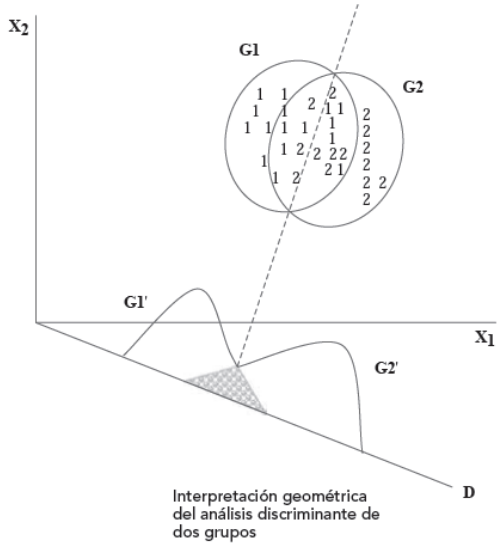


Figura 23. Interpretación de análisis discriminante de dos grupos. Tomada de (7).

El método de clasificación LDA es muy eficiente al clasificar nuevos casos, teniendo en cuenta que, al otorgar a una clase a un nuevo caso, es muy diferente de la estimación de la función discriminante.

El modelo entrenado nos otorga una función discriminante, el cual se puede validar con los datos de entrenamiento para comprobar su eficacia, para posteriormente probarla con datos nuevos o variables independientes.

Para realizarlo se calculará la distancia de los centroides de cada grupo y se establece un punto de división equidistante $Z_0 = \frac{n_I \bar{z}_I + n_{II} \bar{z}_{II}}{n_I + n_{II}}$. El valor de Z_0 establecerá un punto de corte discriminante, de tal forma los nuevos casos que sean mayor al punto de corte serán parte del grupo superior y de forma contraria los nuevos casos serán para el grupo inferior.

El modelo final de Fisher, se define como una función $\boldsymbol{Z}$, que es igual a la combinación lineal de $\boldsymbol{p}$ variables en la clase $\boldsymbol{X}$, como se muestra en la ecuación 16:

$$\boldsymbol{Z_i = \beta_0 + \beta_{1X1i} + \beta_{2X2i} + \ldots + \beta_{pXpi}} \; \boldsymbol{con\; i = 1, \ldots, n} \tag{16}$$

donde z_i será la puntuación discriminante. La forma matricial de la misma queda expresado como:

$$\mathrm{Z} = \mathrm{X}\beta \tag{17}$$

Donde las variables relacionan su desviación respecto a la media y, de esta manera, las puntuaciones z_i también lo hará. La variabilidad de la función discriminante se expresa entonces como:

$$\boldsymbol{Z'Z = \beta' X' X \beta} \tag{18}$$

Siendo $X'X$, una matriz simétrica, considerada como la suma de cuadrados totales de las variables perteneciente a la matriz X, $X'X$ se puede descomponer en la suma de la matriz F y la matriz W, para obtener una variabilidad denotada por la ecuación 19:

$$\boldsymbol{Z'Z = \beta' X' X \beta = \beta' F \beta + \beta' W \beta} \quad (19)$$

Tomando en cuenta que **F** y **W** se calculan con los datos muestrales, mientras que los coeficientes β1, ⋯, βp son los próximos a calcular.

El nuevo valor de los coeficientes β_j se obtiene maximizando la variabilidad de la matriz **F** en razón de la variabilidad de la matriz **W,** el objetivo es encontrar la mayor variabilidad entre matrices, de tal forma que, se encuentre una función discriminante poseedora de un eje discriminante que separe de la mejor forma a estas matrices y su dispersión sea mínima.

Expresando la variabilidad entre grupos como $Max\ \lambda = \frac{\boldsymbol{\beta' F \beta}}{\boldsymbol{\beta' W \beta}}$, dicho esto la solución se obtiene derivando λ respectos a β e igualando a cero, como se muestra a continuación.

$$\frac{\delta \lambda}{\delta \beta} = \frac{2F\beta(\beta' WB) - 2W\beta(\beta' FB)}{(\beta' WB)^2} = 0$$

$$F\beta(\beta' WB) - W\beta(\beta' FB) = 0 \quad (20)$$

$$\frac{F\beta}{W\beta} = \frac{(\beta' FB)}{(\beta' WB)} = \lambda$$

$$F\beta = W\beta\lambda$$

$$W^{-1}F\beta = \lambda\beta$$

Donde β será el primer vector propio y a su vez el primer eje discriminante obtenido de la matriz no simétrica $W^{-1}F$.

λ será el ratio a maximizar, este valor será equivalente a la efectividad del primer eje discriminante, se debe tener en cuenta que si se trata de dos clases solo será necesario un eje discriminante.

Del proceso de maximización, $(\widehat{\beta_1}, ..., \widehat{\beta_P})$ son coordenadas de los vectores propios unitarios extraído del mayor valor propio de la matriz W^{-1}. Finalmente, el algoritmo LDA implica restar el valor de z0 a la función discriminante, como se muestras en la siguiente ecuación.:

$$d_i = z_i - z_0 = \widehat{\beta_1}x_{1i} + \widehat{\beta_2}x_{2i} + \cdots + \widehat{\beta_p}x_{pi} - z_0 \quad (21)$$

Implementación en R

La función determinada en R para realizar ***lda*** de la libreria “Mass” y su algoritmo necesita la matriz de entrada que vendría a ser la matriz de entrenamiento, donde previamente se especifica la columna de clases.

```
modLDA<-lda(clase~.,trainset)
```
(15)

Para la predicción usamos la función *predict*, en donde especificamos el modelo en los datos de prueba sin contar la columna de las clases.

3.4.5 Redes Neuronales (NN)

Una red neuronal se considera una función matemática diseñado a partir del modelo de neuronas humanas, su tasa aprendizaje es mediante ensayos repetitivos, para reorganizarse mejor a sí mismas en su etapa intermedia y lograr una mejor predicción en su etapa final.

En 1943 se propuso la primera red neuronal de la idea del psiquiatra Warren McCulloch y el desarrollo del matemático Walter Pitts. Consta de un conjunto de nodos que actúan como entradas salidas y procesos intermedios. Los nodos se conectan entre sí por medio de trayectorias ponderadas, se evalúa el error de las ponderaciones y se las modifica para lograr mejores predicciones, de la misma forma se vuelve a evaluar y se realiza cambios a los valores ponderados de acuerdo a su error. Este ciclo de evaluación se llama preparación o evaluación. La calibración final del modelo se logra evaluando por medio de muestras de pruebas.

En 1958, Rosenblat F. generó el primer perceptrón el cual consistía en un algoritmo compuesto por dos capas internas que podía recocer patrones a partir de simples operaciones de sumas y restas. Luego durante por dos décadas no hubo avances significativos hasta los años 80 donde se produjo el importante avance teórico, hasta lograr crear el 1990 el algoritmo de BackPropagation creado por Werbos P.

Tipos de modelos de red neuronal

Los modelos más usados actualmente en redes neuronales con aplicación practicas sobre el 90% son de 4 tipos:

1. **MLP** Modelo perceptrón Multicapa.
2. **SOFM** Mapas autoorganizados de Kohonen.
3. **LVQ** Vector de cuantificación.
4. **RBF** Redes de Base Radial.

De los cuatro modelos el modelo perceptrón multicapa es usado en un 70 % en aplicaciones prácticas. Como se muestra en la figura 24.

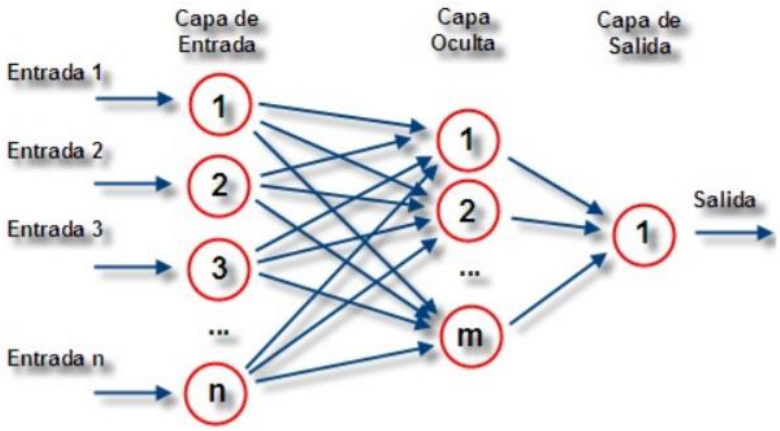


Figura 24. Esquema de Red Neuronal con una capa oculta. Tomada de (14).

Modelo perceptrón Multicapa.

Este modelo se basa en las neuronas del cerebro humano, Figura 25, se compone de nodos que cumplen diferentes funciones. Los primeros valores se construyen a través de un acercamiento las entradas, donde el resumen del procesamiento es equivalente a las entradas por un valor ponderado, este valor será procesado por una función que llevará dicho valor al siguiente nodo del sistema. Finalmente llegará a la salida que será el resultado de la función.

En este proceso se distinguen tres secciones en la red, los nodos de entrada, los nodos intermedios o también llamados capas ocultas y los nodos de salida. La función de activación es no lineal y permite ir de un nodo al siguiente.

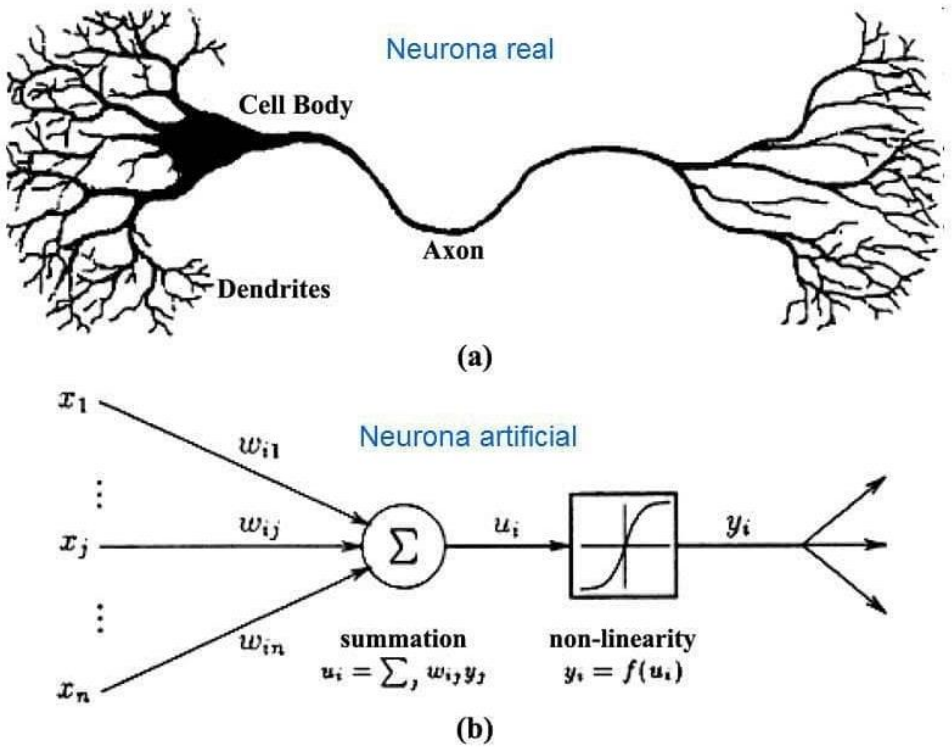


Figura 25. Comparación de una red neuronal con una neurona biológica. Tomado de (31)

Tolerancia a fallos: El sistema es tolerante aun cuando esté presente fallos, si los nuevos datos son redundantes o fuera de contexto el modelo creado puede otorgar una clasificación aceptable.

Paralelismo Masivo: Se refiere a los cálculos de procesamiento en tiempo real, logrando resultados agradables para implementaciones reales.

Implementación en R

Neuralnet es un paquete de R que permite la creación de un modelo a través de una red neuronal de forma sencilla, entendiendo por forma sencilla el modelo de un perceptrón cuya función no es más que una función sinusoide realizada en los años Pitts a principios de los 40 y mejorada por Fran Mc Culloch a finales de los años 50.

$$\sin(x) = \begin{Bmatrix} 1: x \geq 0 \\ -1: x \leq 0 \end{Bmatrix} \quad (22)$$

Pero un simple perceptrón no puede resolver problemas no lineales, por lo que en los años 80 se desarrolló un algoritmo de propagación de errores hacia adelante que resolvía este problema y permitía estimar los pesos de las capas intermedias.

Uso de la librería neuralnet.

```
library(neuralnet)
```

Código de implementación con las clases de entrenamiento.

```
modelo <- neuralnet(class ~ .,data = trainset,hidden = c(10,20,5))
```

(16)

R ofrece la opción de dibujar nuestro modelo de red neuronal creado mediante el comando *plot(modelo),* el modelo muestra las entradas, 3 capas ocultas de 10, 20, y 5 nodos y una única salida, la cual sería la clase (un valor entre 0 y 2), además la conexión entre nodos unidos por los pesos. La arquitectura de red y los parámetros que la componen se muestran en la siguiente Figura 26:

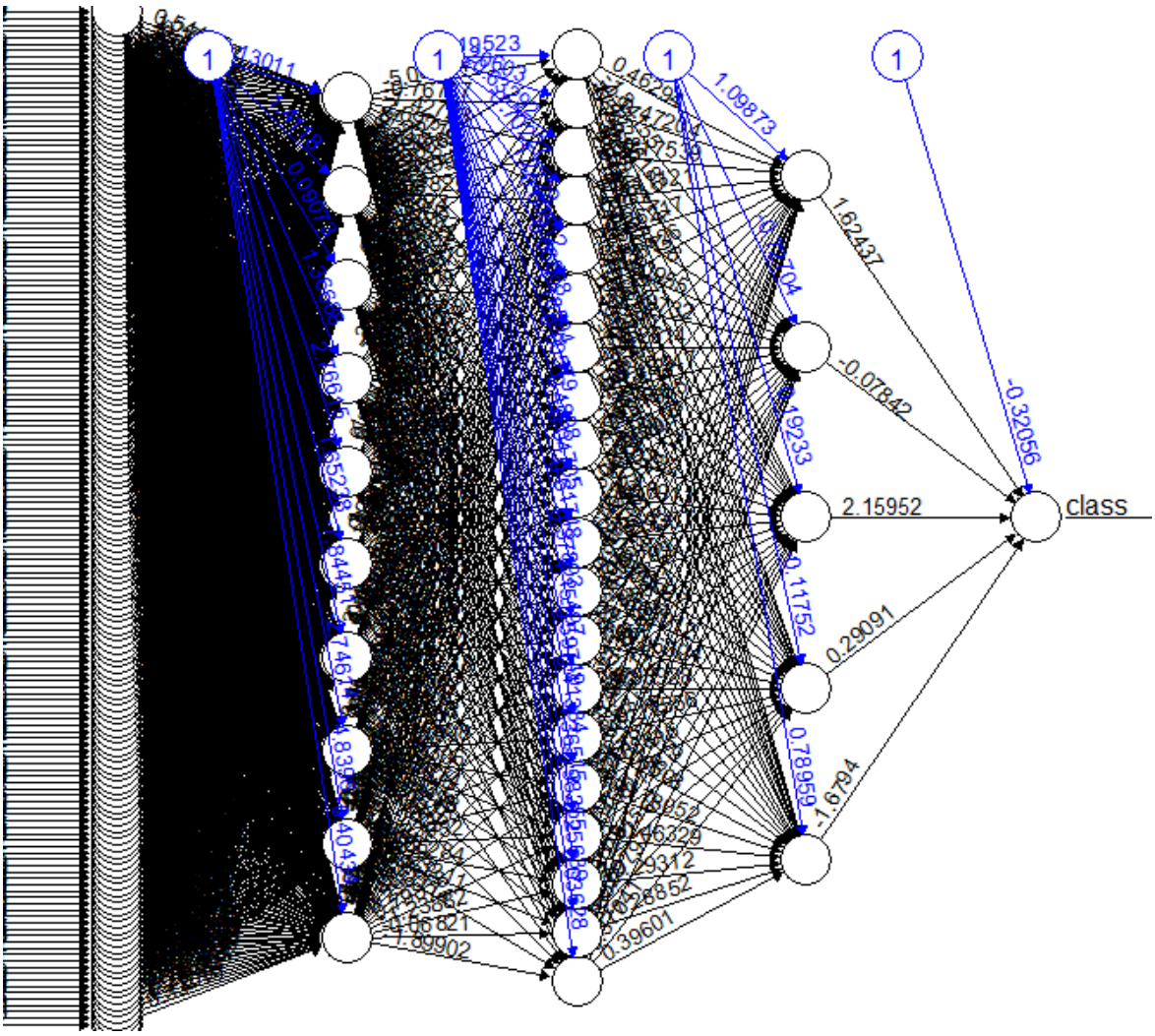


Figura 24. Esquema de una red neuronal de 20, 1 0 ,5 capas ocultas.

4. Resultados

Siguiendo la metodología planteada en el esquema de la Figura 6, donde se parte de organizar las muestras de hojas de mango y realizar la lectura de las imágenes hiperespectrales, luego seleccionar las clases y elaborar la matriz de datos con las 3 clases propuestas, se procede a normalizar los datos, selección de bandas, reducir su dimensionalidad y dividir en dos conjuntos: grupos de entrenamiento y pruebas, con los datos de entrenamiento se procede a entrenar 5 modelos distintos. En el presente capítulo se usará el conjunto de prueba para realizar una validación de los modelos creados, se presenta una comparación de los diferentes modelos propuestos, se ilustra a través de diagramas de confusión y tablas comparativas hasta poder encontrar un modelo ideal.

4.1 Validación

Una vez el dataframe de prueba haya sido procesado por cualquiera de los modelos de clasificación planteados, el resultado nos otorga dos clases de errores que pueden llegar a existir, tomando un caso binario tendríamos que los ceros se pueden clasificar incorrectamente como unos y unos que pueden ser clasificados incorrectamente como ceros. A partir de este enunciado se contruye la Tabla 3:

Valor estimado Yi / Valor real Yi	Yi=0	Yi=1	Yi=2
Yi=0	P11	P12	P13
Yi=1	P21	P22	P23
Yi=2	P31	P32	P33

Tabla 3. Variables de valores propuestos y estimados.

La exactitud quiere decir que tan cerca se aproxima el resultado al llegar al valor real. Se refiere a los verdaderos predichos por el modelo y todos los casos positivos. Mientras, la precisión es la dispersión de los valores obtenidos a partir de la repetitividad de su magnitud, se representa por los casos predichos correctamente (tanto positivas y negativas), y el total de predicciones. La explicación de la exactitud y la precisión se puede apreciar en la Figura 27.

INDICE	DEFINICION	EXPRESION
Tasa de aciertos, Accuracy (Exactitud)	Porcentaje de aciertos del modelo acertados de todas las clases.	$\frac{P_{11}+P_{22}+P_{33}}{P_{11}+P_{12}+P_{13}+P_{21}+P_{22}+P_{23}+P_{31}+P_{32}+P_{33}}$
Tasa de errores	Cociente entre las predicciones incorrectas y el total de predicciones	$\frac{P_{12}+P_{13}+P_{21}+P_{23}+P_{31}+P_{32}}{P_{11}+P_{12}+P_{21}+P_{22}}$

Precisión	Es un indicador de calidad del modelo entrenado. Se define como el cociente de los valores correctos para los valores positivos del mismo.	$\frac{P_{33}}{P_{13} + P_{23} + P_{33}}$
Sensibilidad, Recall (Exhaustividad)	Cantidad del modelo que se va a identificar, se define como el cociente entre los unos correctos y el total de valores uno observados.	$\frac{P_{33}}{P_{31} + P_{32} + P_{33}}$
F1	Es un indicador resumen de recall y precisión. Se define como la media armónica entre la precisión y recall.	$\frac{2 * \text{precisión} * \text{recall}}{\text{precisión} + \text{recall}}$

Tabla 4. Índices de validación de modelos.

La exactitud quiere decir que tan cerca se aproxima el resultado al llegar al valor real. Se refiere a los verdaderos predichos por el modelo y todos los casos positivos. Mientras, la precisión es la dispersión de los valores obtenidos a partir de la repetitividad de su magnitud, se representa por los casos predichos correctamente (tanto positivas y negativas), y el total de predicciones. La explicación de la exactitud y la precisión se puede apreciar en la Figura 27.

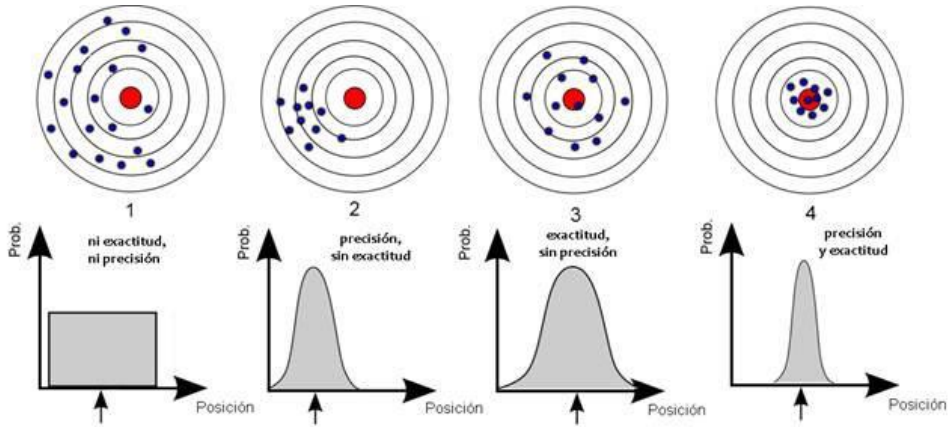


Figura 25. Explicación de Precisión y Exactitud. Tomada de 12

Por otro lado, el índice Recall o Exhaustividad indica lo bien que un modelo obtenido puede identificar una clase. También existe el parámetro F1 o F1 Score calcula por medio del medio armónico entre la precisión y la exhaustividad.

Matriz de confusión

Siendo ***n*** el total de las clases propuestas, una matriz de confusión es la representación de una matriz cuadrada nxn, arreglada de tal forma que las filas estén las clases propuestas y en las columnas las clases predichas. El objetivo de una matriz de confusión es mostrar cuando una clase es clasificada en otra y poder calcular distintos tipos de errores.

Con la ayuda de la librería “tidyverse” se graficará las matrices de confusión en R, con uso del conjunto de prueba y el modelo ya entrenado. El gráfico es intuitivo y fácil de interpretar, además ofrece un porcentaje de acierto en cada clase, como se muestra en la Tabla 5.

```
d_multi <- tibble("target" =testset$class,"prediction" = y_pred_red)
d_multi
conf_mat <- confusion_matrix(targets= testset$class, predictions = y_pred_red)
conf_mat
```
(17)

```
plot_confusion_matrix(conf_mat$'Confusion Matrix'[[1]],  add_sums = TRUE)
```

En nuestro estudio para verificar el mejor modelo se usará el parámetro “exactitud”, con un valor de 0 a 1, que se obtendrá de la matriz de confusión de dicho modelo, el parámetro se comparará con el de los demás modelos para llegar a una conclusión final. Para obtener dicho valor se usa el siguiente código:

```
MC<-table(testset[,9],modelo)
MC
exactitud<-(sum(diag(MC)))/(sum(MC))
exactitud
```

(18)

A continuación, se detallan los resultados por cada modelo creado además de algunas ventajas de cada modelo.

4.2 Máquina de Vectores de Soporte

El resultado al ejecutar el algoritmo de Máquina de vectores de soporte devuelve un modelo entrenado con una exactitud de 96.88%, mientras que al aplicar el mismo algoritmo con menos datos (los 8 componentes principales), se obtiene una exactitud de 96.76%. La diferencia de exactitud es menor al 1 %, la cual se considera como mínima, obteniendo una ventaja de menor procesamiento de datos y obtener casi el mismo resultado. Los resultados se muestran en las siguientes matrices de confusión.

En el caso de las 73 bandas seleccionadas el modelo encontrado consta de 373 vectores de soporte, los cuales otorgan una exactitud del 75.72%. Además, de su bajo rendimiento se generó una clase adicional al redondear los valores finales, aunque es mínima provoca que el clasificador no sea tan eficiente con estos valores de entrada.

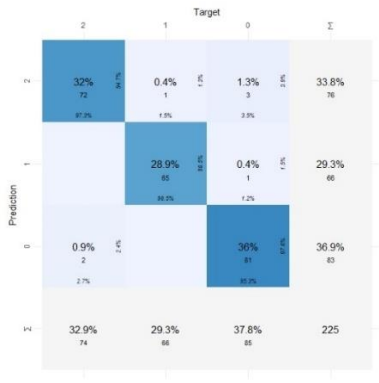
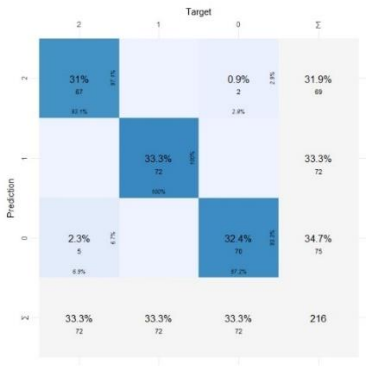
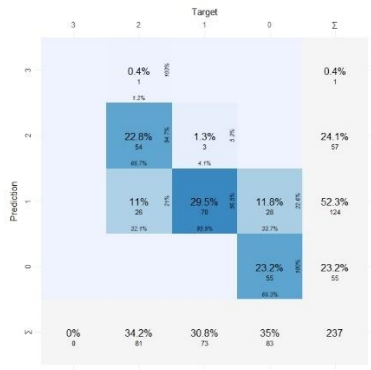
160 bandas espectrales	8 componentes principales PCA	Selección de 73 bandas
		

Tabla 5. Matrices de confusión con diferentes entradas con el modelo SVM.

4.3 Random Forest

En el caso del clasificador random forest se analizará la exactitud en tres casos, primero se analizará en todo el espectro con las 160 bandas, luego al aplicar la técnica de reducción de dimensionalidad PCA con las primeras 8 bandas, elegidas como la combinación de bandas de mayor relevancia y finalmente con las 73 bandas seleccionadas.

La exactitud al ejecutar el algoritmo de Random forest nos devuelve un modelo entrenado con una exactitud de 98.40%, mientras que al aplicar el mismo algoritmo con los 8 componentes principales se obtiene una exactitud de 96.75%, la diferencia de exactitud es de 1.64, la cual se la puede considerar mínima, obteniendo ventajas al procesar menos datos y obtener casi el mismo resultado. Los resultados se muestran en las siguientes matrices de confusión.

En el caso de la selección de bandas obtenemos un error cuadrático medio de 0.04 y con este modelo el 94.93, estos valores se obtienen al emplear 500 árboles de decisión.

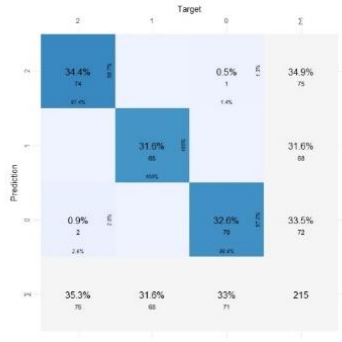
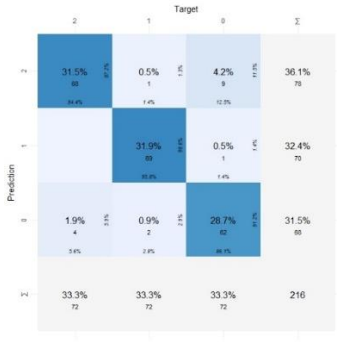
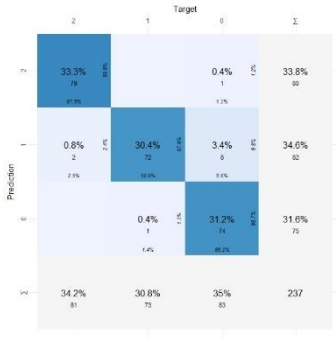
160 bandas espectrales	**8 componentes principales PCA**	**Selección de 73 bandas**
		

Tabla 6. Matrices de confusión con diferentes entradas con el modelo RF.

4.4 Gradient Boosting

El modelo de clasificador Boosting aplicado es un modelo de Boosted Gradiente el cual es generado por 10000 árboles, y el parámetro de contracción *lamda=0,01,* que también es una especie de tasa de aprendizaje. El siguiente parámetro es la profundidad de interacción D que es el total de divisiones que se quieren hacer, así se puede identificar que cada árbol es un árbol pequeño con tan solo 4 divisiones.
Se puede resumir el modelo en la Figura 28, donde se muestra la relevancia de las bandas, mostrando cuales son más significativas según este algoritmo, dicho gráfico muestra en diagrama de barras la parte superior como banda más importante la banda 736, seguida de la banda 787 así sucesivamente, hasta llegar a la menos importante. Además, se presenta características importantes que explican la varianza máxima en el conjunto de datos es, es decir, la banda más significativa (porcentaje).

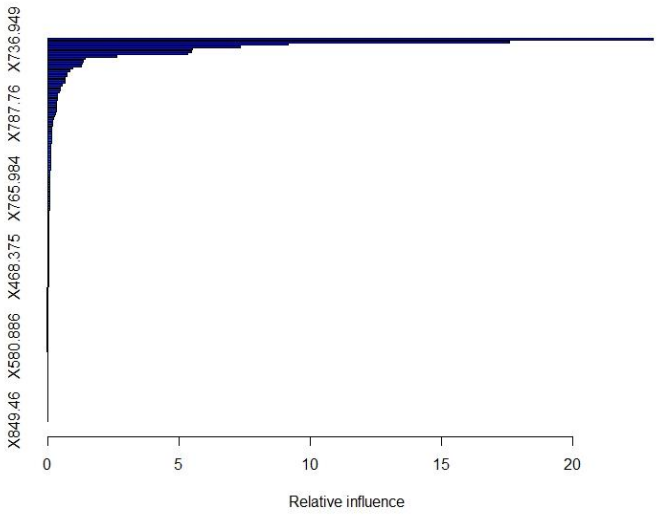


Figura 26.Influencia por banda según Boosting de 160 longitudes de onda.

La exactitud al ejecutar el algoritmo de Gradient Boosting devuelve un modelo entrenado con una exactitud de 97.15%, mientras que al aplicar el mismo algoritmo con los 8 componentes principales se obtiene una exactitud de 95.83%, la diferencia de exactitud es de 1.64%, la cual se la puede considerar mínima, obteniendo ventajas al procesar menos datos y obtener casi el mismo resultado. Los resultados se muestran en las siguientes matrices de confusión.

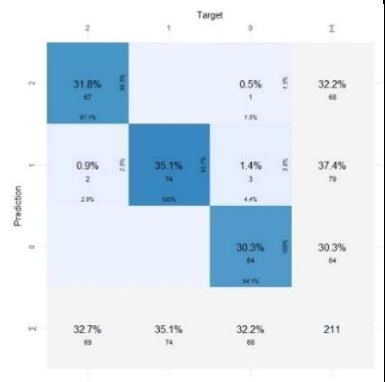
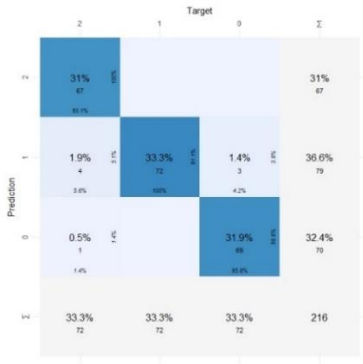
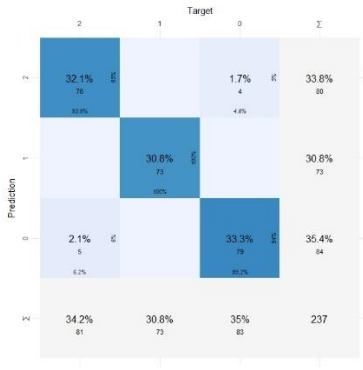
160 bandas espectrales	**8 componentes principales PCA**	**Selección de 73 bandas**
		

Tabla 7. Matrices de confusión con diferentes entradas con el modelo Gradient Boosting.

En el caso de las 73 bandas seleccionadas, Gradient Boosting en R muestra la intervención de las bandas en el modelo de clasificación por ejemplo la banda la 736 con su varianza de 37.86, la banda 707 con su varianza de 31.30, después de aquella, la banda 744 representa una 5.17 como se muestra en la Figura 29, las 3 otorgan una apreciación del 74.33%. La exactitud del modelo es de un 95.35%. Se realizó las pruebas con 10000 árboles y con 1000 árboles logrando resultados similares.

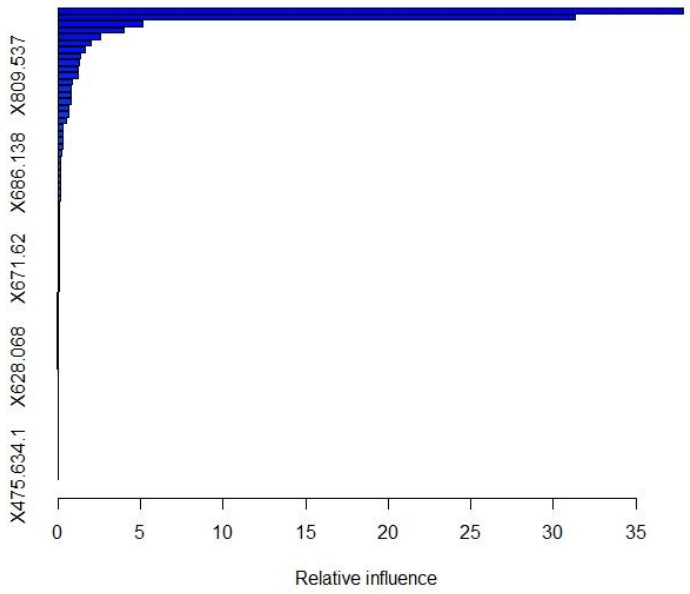


Figura 27.Influencia Por Banda según Boosting de 73 longitudes de onda seleccionadas.

4.5 LDA

La exactitud al ejecutar el algoritmo de LDA con un total de 160 bandas devuelve un modelo entrenado con una exactitud de 93.31%, mientras que al aplicar el mismo algoritmo con los 8 componentes principales se obtiene una exactitud de 86.11%, la diferencia de exactitud es de 7.2%, la cual se la puede considerar considerable, pero en un rango aceptable considerando la disminución de la matriz a procesar. Los resultados se muestran en las siguientes matrices de confusión.

En el caso de las 73 bandas seleccionadas se obtiene una exactitud del 96.20%, en este caso se puede resaltar la fácil configuración del código para LDA.

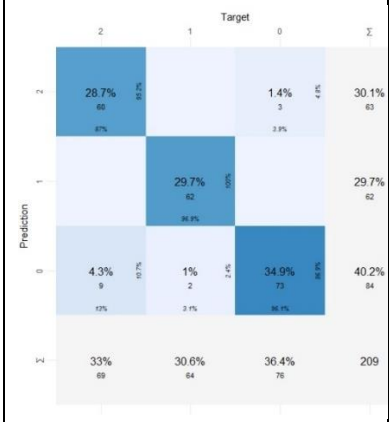
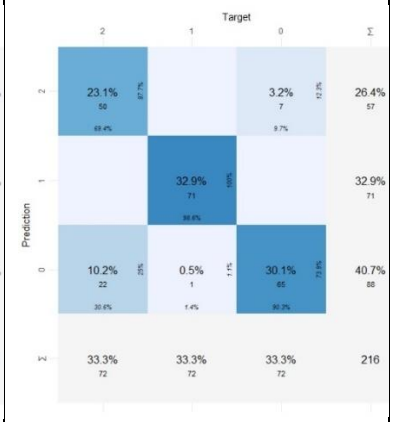
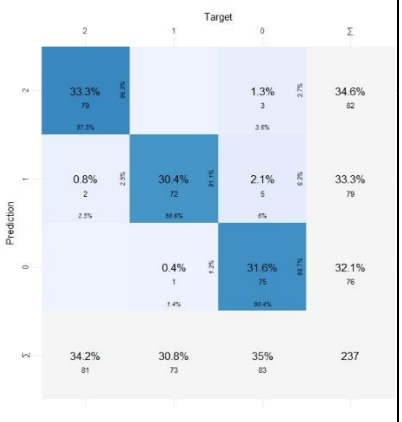
160 bandas espectrales	8 componentes principales PCA	Selección de 73 bandas
		

Tabla 8. Matrices de confusión con diferentes entradas con el modelo LDA.

4.6 REDES NEURONALES

La validación del modelo de redes neuronales al aplicar las 160 bandas de entradas da como resultado una exactitud 96,63%, este es el mejor valor obtenido al otorgar diferentes números de capas ocultas. En el caso de las 73 bandas seleccionadas se realizó con 10, 20, 5 nodos es su capa oculta, aunque el modelo demora en generarse se obtiene una exactitud de 97.89%.

En el modelo de red neuronal creado a partir de los 8 componentes principales, obtuvo algunos problemas, se probó con diferentes valores en sus capas ocultas pero el modelo no era óptimo y creaba clases adicionales fuera de contexto, por esta razón se decidió descartar. Los resultados obtenidos se presentan en la Tabla 9.

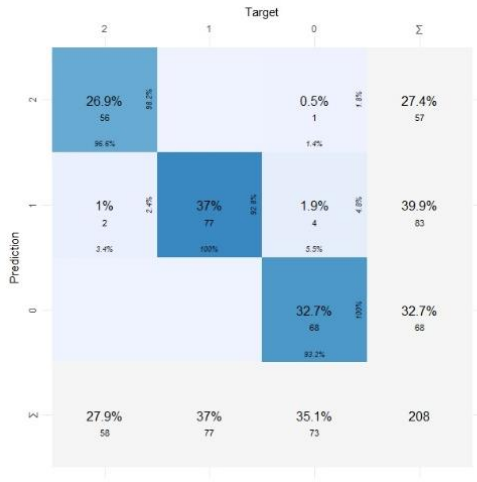
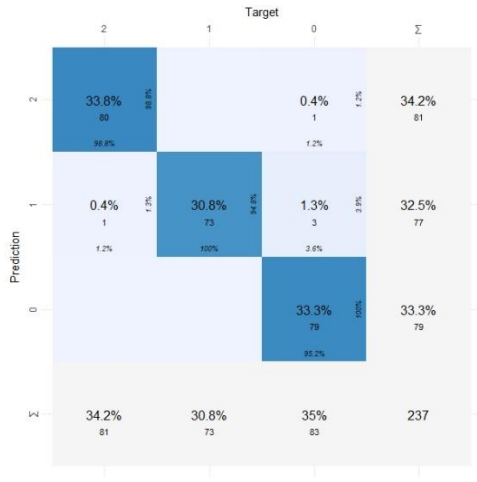
160 bandas espectrales	Selección de 73 bandas
	

Tabla 9. Matrices de confusión con diferentes entradas con el modelo NN.

4.7 Comparación de tiempo de entrenamiento

Como resultado adicional se considera el tiempo de procesamiento de las diferentes técnicas de clasificación y se elabora una tabla comparativa para una mejor apreciación de los resultados. Se debe tomar en cuenta que los resultados dependen del CPU que se esté usando, en este caso los resultados de los clasificadores se realizaron sobre una laptop Lenovo procesador Intel Core I7 de 8 gb de Ram.

Al crear un modelo sobre las 160 longitudes de ondas y los 8 componentes principales, LDA lo hace más rápido, mientras el modelo Boosting es el que se demora cerca del minuto y 0.02 segundos en el segundo caso. En el caso de las 73 bandas seleccionadas, SVM es el modelo en entrenarse más rápido, Los resultados se presentan en la Tabla 10.

Clasificador	***160 bandas***	***PCA***	***Selección de 73 bandas***
SVM	0.44 s	0.09 s	0.20s
Random Forest	0.68 s	0.08 s	5.83s
Boosting	50.89 s	3.02 s	2.53s
LDA	0.39s	0.02 s	0.46s
Redes Neuronales	1.49s	NA	4.05s

Tabla 10. Comparación de tiempo de creación de cada modelo.

4.8 Comparación final de modelos

Se presentará dos tablas de comparación para descifrar cual modelo nos ofrece mayor desempeño: La primera tabla nos presenta una comparación de los parámetros precisión, exhaustividad y el valor F1, donde los modelos SVM y redes neuronales obtienen el rendimiento más alto del Recall al ser entrenado con las 73 bandas seleccionadas con el método stepwize, alcanzo el 100% lo que quiere decir el total de todas las bandas serán identificadas en la clase infectadas. Al tratarse de la precisión del modelo mejores resultados obtuvieron SVM Y reden neuronales al trabajar con las 160 bandas, obteniendo una precisión de 99% y 100% respectivamente, lo que nos quiere decir que trabajando de esta manera el modero tendrá una mejor calidad de clasificación de puntos infectados. No necesariamente el valor combinado de estas métricas pertenece a los modelos antes citados, los mejores valores en las medidas F1 obtenidos en Random Forest con un 98% al ser entrenado con 160 bandas y redes neuronales con el mismo valor, pero con 73 bandas aluden calidad y cantidad de bandas identificadas en el proceso de clasificación. La información se resume en la Tabla 11.

SVM			
	160 bandas	8 componentes principales	Selección de 73 bandas
Precisión	0.95	0.97	0.66
Recall	0.98	0.93	1.00
F1	0.96	0.95	0.80
RANDON FOREST			
	160 bandas	8 componentes principales	Selección de 73 bandas
Precisión	0.99	0.86	0.89
Recall	0.97	0.91	0.99
F1	0.98	0.89	0.94
GRADIENT BOOSTING			
	160 bandas	8 componentes principales	Selección de 73 bandas
Precisión	0.94	0.96	0.95
Recall	1.00	0.99	0.94
F1	0.97	0.97	0.95

LDA			
	160 bandas	8 componentes principales	Selección de 73 bandas
Precisión	0.96	0.97	0.90
Recall	0.87	0.93	0.99
F1	0.91	0.95	0.94
NN			
	160 bandas	8 componentes principales	Selección de 73 bandas
Precisión	0.93	NA	0.95
Recall	1.00	NA	1.00
F1	0.96	NA	0.98

Tabla 11. Resumen de indicadores Precisión, Recall y F1 de los 5 modelos propuestos.

Para establecer cual clasificador ofrece el mejor desempeño tomando en cuenta la exactitud, se ofrece la siguiente tabla comparativa, en cual se puede apreciar que el mejor resultado se obtiene al trabajar con todas las bandas del píxel, aunque requiere más recursos del sistema, así se puede mencionar que el clasificador Random Forest con una exactitud del 98.60%, ofrece mejores resultados frente a las demás técnicas. Por el contrario, el peor resultado cae sobre el clasificador LDA con 93.31%, debido que no es un problema que se pueda enfrentar con una técnica lineal. En cuanto a los clasificadores usando como entradas los componentes principales, se observa que mejor desempeño ofrece SVM con un 96.75%, con una diferencia inferior al 0.001% de las bandas del espectro completo. Finalmente, por encima de los demás modelos, redes neuronales con 73 bandas seleccionadas, y una configuración de 10, 20 y 5 nodos en sus capas ocultas ofrecen la mejor precisión de todas y un rendimiento óptimo. Se resume la presentación de estos resultados en la Tabla 12.

Clasificador	*160 bandas*	*PCA (%)*	*Selección de 73 Bandas*
SVM	0.9688888	0.9675926	0.7572
Random Forest	0.98604651	0.9212963	0.9493
Boosting	0.97156398	0.9583333	0.9535
LDA	0.9331435	0.8611111	0.9620
Redes Neuronales	0.96634615	NA	0.9789

Tabla 12. Comparación de la exactitud de cada modelo.

5. Conclusiones

Partiendo de la metodología propuesta en el esquema de la Figura 8 y con el objetivo de detectar la antracnosis en muestras de hojas de mango de la variante Tommy en etapas tempranas o en etapas ya enfermas, con muestras obtenidas de un repositorio compartido, con base para nuestro análisis en el lenguaje R, y partiendo de la lectura e identificación de las tres clases propuestas: sanas, inoculadas y enfermas, se observó que al organizar y normalizar el dataframe se tiene mejores resultado que un dataframe sin procesar, ya que al tomar en cuenta la varianza de longitudes de onda y calcular en su misma escala elevada al cuadrado, se estandarizan todas las variables para que tengan media 0 y desviación estándar 1, de esta forma se evita que aquellas variables cuya escala sea mayor tengan más importancia en el nuevo dataframe.

Se observó que realizar el proceso de reducción de dimensionalidad por el método PCA es muy viable previo a un entrenamiento, se obtiene un modelo de alta exactitud y un modelo que consume menos recursos computacionales, además el tiempo de creación del modelo es menor según la Tabla 10. Por lo tanto, el modelo de clasificación más favorecido por este proceso fue LDA que resultó en el entrenamiento más rápido con una entrada de bandas combinadas por PCA, aunque no se obtuvo la mejor exactitud.

El proceso de selección de las bandas más significativas obtuvo un mejor resultarlos a combinar los métodos de selección Forward y Backward, pasando de 70 bandas y 94 bandas respectivamente, a 73 que reflejan el 73.95% del dataframe original.

El mejor modelo obtenido fue el de redes neuronales con las entradas de 73 longitudes de onda seleccionadas, destacando las 609, 700, 682, 533, 987, y 588 nm, obtenidas mediante el método de stepwize, las cuales otorgan una eficiencia del 97.89% y una rapidez de entrenamiento de su modelo de 4.05 segundos. También el modelo es respaldado por las métricas de precisión con un 95%, exhaustividad con el 100% y valor F1 con el 98%, que son sinónimos de calidad del modelo de clasificación.

Finalmente, el modelo Grandient bosting genera 31 bandas características que representan el 91.94 % de todas las 160 bandas, se puede proponer como trabajo a futuro un nuevo modelo con estas bandas. Adicionalmente R y RStudio no trabajan solo como un software de entrega de resultados sino como una plataforma que brinda reportes completos de los análisis de fácil entendimiento.

6. Bibliografía

1. Rodríguez, A. T., Dávila, J. F. R., Siclán, M. L. S., Vildózola, Á. C., Zamora, F. I. M., & Díaz, A. V. L. (2020). Distribución espacial de antracnosis (Colletotrichum gloeosporioides Penz) en aguacate en el Estado de México, México. *Revista Argentina de Microbiología*, *52*(1), 72-81.
2. Silva-Rojas, H. V., & Ávila-Quezada, G. D. (2011). Phylogenetic and morphological identification of Colletotrichum boninense: a novel causal agent of anthracnose in avocado. *Plant Pathology*, *60*(5), 899-908.
3. Rodríguez, A. T., Dávila, J. F. R., Siclán, M. L. S., Vildózola, Á. C., Zamora, F. I. M., & Díaz, A. V. L. (2020). Distribución espacial de antracnosis (Colletotrichum gloeosporioides Penz) en aguacate en el Estado de México, México. *Revista Argentina de Microbiología*, *52*(1), 72-81.
4. Ardila, C. E. C., Ramirez, L. A., & Ortiz, F. A. P. (2020). Spectral analysis for the early detection of anthracnose in fruits of Sugar Mango (Mangifera indica). *Computers and Electronics in Agriculture*, *173*, 105357.
5. Rondón, O., Sanabría de Albarracín, N., & Rondón, A. (2006). Respuesta in vitro a la acción de fungicidas para el control de antracnosis, Colletotrichum gloeosporioides Penz, en frutos de mango. *Agronomía tropical*, *56*(2), 219-235.
6. Páez Redondo, A. R. (2003). Tecnologías sostenibles para el manejo de la antracnosis (Colletotrichum gloesporioides (Penz.) Penz. & Sacc.) en papaya (Carica papaya L.) y mango (Mangifera indica L.).
7. Páez, A. R. (1997). Respuesta de cultivares de mango (Magnifera indica L.) a la antracnosis en la Costa Atlántica colombiana. *Ciencia y Tecnología Agropecuaria*, *2*(1), 45-53.

8. Galvis, J. P. A. (2017). *Evaluación de un método de Aprendizaje Supervisado para la detección de las enfermedades, Antracnosis y Phytophthora Infestans en cultivos de fruta de Risaralda* (Doctoral dissertation, Universidad Tecnológica de Pereira).

9. Sevilla, M. A. C. S., Rocha, M. A. J. R., & Pintor, M. G. A. H. Identificación de enfermedad (antracnosis) en fresa a partir de imágenes digitales tomadas a la variedad de fresa festival.

10. Santana, J. S., & Farfán, E. M. (2014). El arte de programar en R: un lenguaje para la estadística. *Instituto Mexicano de Tecnología del Agua*, *1*.

11. hyperSpec Introduction, Claudia Beleites, https://cran.r-project.org/web/packages/hyperSpec/vignettes/hyperspec.pdf, November 27, 2020.

12. Package 'hyperSpec', Work with Hyperspectral Data, i.e. Spectra + Meta Information (Spatial, Time, Concentration, ...), Claudia Beleites, https://cran.r-project.org/web/packages/hyperSpec/hyperSpec.pdf, 2020-11-27.

13. Singha, S., Pasupuleti, S., Singha, S.S., Singh, R., Kumar, S.57194762416;57194568754;57194762415;57215037747;57211640379; Prediction of groundwater quality using efficient machine learning technique (2021) Chemosphere, 276, art. no. 130265, . https://www.scopus.com/inward/record.uri?eid=2-s2.0-5102890532&doi=10.1016%2fj.chemosphere.2021.130265&partnerID=40&md5=fb8827a43d04acdee18bfc452ebf41b2

14. R: The R Project for Statistical Computing R: The R Project for Statistical Computing. (2020). Retrieved 6 April 2020, from https://www.r-project.org/

15. RStudio | Open source & professional software for data science teams, RStudio | Open source & professional software for data science teams. (2020). Retrieved 6 April 2020, from https://rstudio.com/

16. X Kangab, P Duanab, S Liab, Hyperspectral image visualization with edge-preserving filtering and principal component analysis, Information Fusion, Volume 57, May 2020, Pages 130-143.

17. Y. Liu, X. Chen, Z. Wang, Z.J. Wang, R.K. Ward, X. Wang, Deep learning for pixel-level image fusion: recent advances and future prospects, Inf. Fusion, 42 (2018), pp. 158-173, ArticleDownload PDFView Record in ScopusGoogle Scholar

18. P. Perona, J. Malik, Scale-space and edge detection using anisotropic diffusion, IEEE Trans. Pattern Anal. Mach. Intel., 12 (7) (1990), pp. 629-639.

19. F. Rottensteiner, J. Trinder, S. Clode, K. Kubik, Using the Dempster–Shafer method for the fusion of LIDAR data and multi-spectral images for building detection, Inf. Fusion, 6 (4) (2005), pp. 283-300.

20. Zhang J.,Cheng T., Guo W., Xu X. Qiao H., Xie Y., Ma X., Leaf area index estimation model for UAV image hyperspectral data based on wavelength variable selection and machine learning methods, Volume 17, Issue 1, Article number 49, December 2021, 10.1186/s13007-021-00750-5.

21. P Bajcsy, P Groves, Methodology for hyperspectral band selection, Photogrammetric Engineering and Remote Sensing journal, Vol. 70, Number 7, July 2004, pp. 793-802, 21-May-2014.

22. J Gonzalez, S Castelblanco, Aplicación de técnicas de machine learning para estimar propiedades físicas de hidrocarburos a partir de firmas espectrales, pp 7-30, https://repositorio.uniandes.edu.co/bitstream/handle/1992/34930/u820869.pdf?sequence=1.

23. García Navarrete, O. (2013). *Detección temprana de daños mecánicos por golpe en el manejo poscosecha de la manzana Fuji a través de imágenes hiperespectrales.*

24. The Comprehensive R Archive Network", Cran.r-project.org, 2020. [Online]. Available: https://cran.r-project.org/. [Accessed: 13- Apr- 2020].

25. The Comprehensive R Archive Network", Cran.r-project.org, 2020. [Online]. Available: https://cran.r-project.org/. [Accessed: 13- Apr- 2020].

26. A. Roman-Gonzalez, N Indira Vargas-Cuentas. Análisis de imágenes hiperespectrales. Revista Ingenieria & Desarrollo, 2013, Año 9 (N° 35), pp.14-17. ffhal-00935014f.

27. I. O. Sigirci and G. Bilgin, "Hyperspectral image segmentation using the Dirichlet mixture models," *2014 22nd Signal Processing and Communications Applications Conference (SIU)*, 2014, pp. 983-986, doi: 10.1109/SIU.2014.6830396.

28. Saeidan, A., Khojastehpour, M., Golzarian, M. R., Mooenfard, M., & Khan, H. A. (2021). Detection of foreign materials in cocoa beans by hyperspectral imaging technology. Food Control, 129 doi:10.1016/j.foodcont.2021.108242

29. Lan W., Jaillais B., Renard C., Leca A., Songchao Chen, Le B., B. Sylvie, A method using near infrared hyperspectral imaging to highlight the internal quality of apple fruit slices, Postharvest Biology and Technology, Volume 175, 2021, 111497, ISSN 0925-5214, https://doi.org/10.1016/j.postharvbio.2021.111497.

30. Lassalle, G., Fabre, S., Credoz, A. *et al.* Mapping leaf metal content over industrial brownfields using airborne hyperspectral imaging and optimized vegetation indices. *Sci Rep* **11,** 2 (2021). https://doi-org.ezproxy.unal.edu.co/10.1038/s41598-020-79439-z.

31. Trujillano, J., March, J., & Sorribas, A. (2004). Aproximación metodológica al uso de redes neuronales artificiales para la predicción de resultados en medicina. *Medicina Clínica*, *122*(Supl.1), 59-67. doi: 10.1157/13057536.

Referencia WEB

1. Aplicando la imagen hiperespectral para detectar contaminantes, https://www.ainia.es/tecnoalimentalia/tecnologia/aplicando-la-imagen-hiperespectral-para-detectar-contaminantes/.

2. Qué es R y por qué utilizarlo | OpenWebinars, https://openwebinars.net/blog/que-es-r-y-por-que-utilizarlo/ Accessed: 2021-01-30.

3. ¿Qué es R?, https://www.r-project.org/about.html, Accedido 15- 02 - 2020.

4. Cámaras Hiperespectrales, https://mesurex.com/product-category/camaras-hiperespectrales/

5. Análisis de Componentes Principales (Principal Component Analysis, PCA) y t-SNE, Joaquín Amat Rodrigo,

6. https://www.cienciadedatos.net/documentos/35_principal_component_analysis#Otros_m%C3%A9todos_de_reducci%C3%B3n_de_dimensionalidad, Junio, 2017.

7. Métodos de clasificación**,** https://bookdown.org/content/2274/metodos-de-clasificacion.html#analisis-discriminante.

8. Aumento de gradiente en R | DataScience +, https://datascienceplus.com/gradient-boosting-in-r/, Accessed: 2021-04-12.
9. Reducción de la Dimensionalidad - Aprende IA, https://aprendeia.com/reduccion-de-la-dimensionalidad-machine-learning/, Accessed: 2021-05-25.
10. Step function – Rdocumentation, https://www.rdocumentation.org/packages/stats/versions/3.6.2/topics/step, Accessed: 2021-05-30.
11. Resumen de regresión lineal (lm): interpretación en R – Boostedml, https://boostedml.com/2019/06/linear-regression-in-r-interpreting-summarylm.html, Accessed: 2021-05-30.
12. J Barrios, La matriz de confusión y sus métricas – Inteligencia Artificial, Consultores estratégicos en Ciencia de Datos, https://www.juanbarrios.com/la-matriz-de-confusion-y-sus-metricas/, Accessed: 2021-05-31.
13. HySpex. (2021). from https://www.hyspex.com/hyspex-products/hyspex-classic/hyspex-vnir-1800/, Retrieved 11 June 2021.
14. Minería_Datos Inteligencia_artificial « Inteligencia en el Negocio. Descifrando el 3.0…. (2021). Retrieved 15 July 2021, from http://rtdibermatica.com/?tag=mineria_datos-inteligencia_artificial.
15. Heras, J. (2021). Precision, Recall, F1, Accuracy en clasificación - IArtificial.net. Retrieved 20 July 2021, from https://www.iartificial.net/precision-recall-f1-accuracy-en-clasificacion/

7. Anexos

1. Tabla de selección de bandas por método forward, backward y StepWize

	Backward	AIC	Forward	AIC	StepWize	AIC
1	417.563	-1193.474863	428.452	-1135.265731	428.452	-1135.265731
2	421.193	-1130.491168	435.71	-1186.186572	450.228	-1187.694296
3	424.822	-1146.269947	450.228	-1186.446019	453.857	-1131.673963
4	432.081	-1125.068826	453.857	-1131.673963	457.487	-1098.193424
5	435.71	-1178.317956	457.487	-1098.193424	464.745	-1186.765823
6	439.34	-1117.689345	464.745	-1183.456391	475.634	-1194.283486
7	442.969	-1075.327346	479.263	-450.133395	475.634	-1180.37007
8	446.598	-1192.502661	490.151	-1128.38855	479.263	-1173.393412
9	468.375	-1113.93445	511.927	-1174.368898	479.263	-450.133395
10	472.004	-1141.103529	519.186	-1191.572024	490.151	-1128.38855
11	479.263	-1191.885699	526.445	-1005.853137	511.927	-1194.644497
12	482.892	-1063.426194	530.074	-940.1336577	519.186	-1194.512513
13	486.522	-1165.540666	533.704	-704.7725908	526.445	-1005.853137
14	490.151	-1193.014258	537.333	-1103.149172	530.074	-940.1336577
15	493.78	-1182.493566	540.962	-1048.961804	533.704	-704.7725908
16	497.41	-1184.328793	562.739	-1191.768206	537.333	-1103.149172
17	501.039	-1177.280484	566.368	-1179.998193	540.962	-1048.961804
18	504.669	-1106.384163	569.997	-994.0993793	562.739	-1192.397264
19	508.298	-1092.929488	573.627	-1057.29612	566.368	-1175.927932
20	515.557	-1166.969313	584.515	-878.3521965	569.997	-994.0993793
21	522.816	-1073.349984	588.144	-846.7519391	573.627	-1057.29612
22	544.592	-1110.164365	595.403	-1030.968528	584.515	-878.3521965
23	548.221	-1100.64094	602.662	-1190.836232	588.144	-846.7519391
24	551.851	-1172.550955	606.291	-894.3619283	595.403	-1030.968528
25	555.48	-1180.441146	609.921	-359.0629129	602.662	-1194.454261
26	559.109	-1085.173063	613.55	-1187.712932	606.291	-894.3619283
27	577.256	-1128.697588	617.179	-1156.81433	609.921	-359.0629129
28	580.886	-1186.541566	628.068	-1188.874808	613.55	-1189.449256

29	591.774	-1112.042385	631.697	-1188.564192	617.179	-1156.81433
30	599.033	-1108.287962	638.956	-1192.065144	628.068	-1190.574386
31	606.291	-1194.69855	642.585	-1191.873548	631.697	-1191.768843
32	620.809	-1067.402055	671.62	-1189.273221	638.956	-1194.71272
33	624.438	-1126.905497	678.879	-1122.538157	642.585	-1194.169691
34	635.326	-1135.813713	682.508	-656.8507726	671.62	-1192.847146
35	638.956	-1194.80985	686.138	-917.0466798	678.879	-1122.538157
36	646.215	-1194.620323	689.767	-1190.134415	682.508	-656.8507726
37	649.844	-1191.379124	693.396	-1042.638227	686.138	-917.0466798
38	653.473	-1094.863096	697.026	-954.8131455	689.767	-1193.549884
39	657.103	-1123.224383	700.655	-544.4476766	693.396	-1042.638227
40	660.732	-1137.581511	707.914	-1158.680579	697.026	-954.8131455
41	664.361	-1175.023426	718.802	-1064.774265	700.655	-544.4476766
42	667.991	-1156.228158	722.432	-1171.738883	707.914	-1158.680579
43	675.25	-1104.474688	736.949	-973.3100841	718.802	-1191.138555
44	704.285	-1081.250844	744.208	-1178.327117	718.802	-1064.774265
45	711.543	-1173.79434	758.725	-1168.937394	722.432	-1171.738883
46	715.173	-1057.436956	765.984	-1187.073869	736.949	-973.3100841
47	718.802	-1188.97079	776.872	-1166.140401	744.208	-1182.035677
48	726.061	-1151.459125	784.131	-1186.638075	758.725	-1168.937394
49	729.69	-1188.093124	802.278	-1145.497043	765.984	-1188.916509
50	733.32	-1098.719596	805.907	-1118.668819	776.872	-1166.140401
51	740.578	-1171.173094	809.537	-1149.893339	784.131	-1185.022943
52	747.837	-1059.436068	824.054	-1084.734967	802.278	-1145.497043
53	751.467	-1149.796024	827.684	-1185.954443	805.907	-1118.668819
54	755.096	-1147.945494	831.313	-1176.822632	809.537	-1149.893339
55	762.355	-1169.74123	860.348	-1163.464806	824.054	-1084.734967
56	769.614	-1077.304167	867.607	-1184.939748	827.684	-1194.936882
57	773.243	-1065.416632	871.236	-1072.922309	831.313	-1184.171518
58	780.502	-1157.810111	885.754	-1108.141715	860.348	-1163.464806
59	787.76	-1139.360752	889.383	-1138.387644	867.607	-1185.714237
60	791.39	-1119.570104	900.271	-1173.043402	871.236	-1072.922309
61	795.019	-1083.215684	903.901	-1113.750935	885.754	-1108.141715
62	798.649	-1090.99258	918.418	-1175.540561	889.383	-1138.387644
63	813.166	-1160.857221	925.677	-1182.166909	900.271	-1174.724376
64	816.795	-1096.797408	940.194	-1187.45962	903.901	-1113.750935
65	820.425	-1164.004471	954.712	-1018.994641	918.418	-1177.168408
66	834.942	-1162.426525	972.859	-1092.240049	925.677	-1178.696623
67	838.572	-1144.563508	976.488	-1142.429281	940.194	-1188.364106

68	842.201	-1190.630954	980.118	-1160.850801	954.712	-1018.994641
69	845.831	-1189.794995	983.747	-1153.647938	972.859	-1092.240049
70	849.46	-1069.387063	987.376	-773.0192012	976.488	-1142.429281
71	853.089	-1179.41499			980.118	-1160.850801
72	856.719	-1185.388722			983.747	-1153.647938
73	863.977	-1183.458546			987.376	-773.0192012
74	874.866	-1153.065878				
75	878.495	-1142.856792				
76	882.124	-1115.804022				
77	893.013	-1193.802946				
78	896.642	-1089.05699				
79	907.53	-1121.399727				
80	911.159	-1194.289227				
81	914.789	-1134.035223				
82	922.048	-1055.437648				
83	929.306	-1102.555606				
84	932.936	-1154.652827				
85	936.565	-1087.119906				
86	943.824	-1176.169519				
87	947.453	-1159.362186				
88	951.083	-1132.260541				
89	958.341	-1187.412562				
90	961.971	-1079.281105				
91	965.6	-1181.429474				
92	969.23	-1061.434219				
93	991.006	-1071.369518				
94	994.635	-1168.351857				