

**DESARROLLO DE UNA HERRAMIENTA BASADA EN
APRENDIZAJE AUTOMÁTICO QUE PERMITA LA
ESTIMACIÓN DEL ESTADO DE NITRÓGENO
PRESENTE EN LAS HOJAS DE UN CULTIVO DE
GULUPA USANDO IMÁGENES RGB Y
MULTIESPECTRALES.**

ZAIRA KATHERÍN LUNA CÓRDOBA
Código: 162215215

UNIVERSIDAD DE CUNDINAMARCA
FACULTAD DE INGENIERÍA
INGENIERÍA ELECTRÓNICA
FUSAGASUGÁ.

2020

**DESARROLLO DE UNA HERRAMIENTA BASADA EN
APRENDIZAJE AUTOMÁTICO QUE PERMITA LA
ESTIMACIÓN DEL ESTADO DE NITRÓGENO
PRESENTE EN LAS HOJAS DE UN CULTIVO DE
GULUPA USANDO IMÁGENES RGB Y
MULTIESPECTRALES.**

Trabajo de grado presentado como requisito para optar por el título
de ingeniera electrónica

ZAIRA KATHERÍN LUNA CÓRDOBA
Código: 162215215

Director:

LEONARDO RODRÍGUEZ MÚJICA
Ingeniero electrónico

Línea investigación:

Gestión tecnológica aplicada a los sectores agropecuarios,
agroindustrial y ambientales, software, sistemas emergentes
y nuevas tecnologías.

UNIVERSIDAD DE CUNDINAMARCA
FACULTAD DE INGENIERÍA
INGENIERÍA ELECTRÓNICA
FUSAGASUGÁ.
2020

Dedicatoria

A Dios, a mis padres William y Diana, a mis hermanos Felipe, Kevin y Willi, a mi abuela materna María Argenis y a Venus, por ser parte de mi motivación.

Agradecimientos

En primer lugar agradezco a Dios por permitirme vivir la experiencia de participar en este proyecto.

Agradezco a mis padres, por su amor, por su apoyo incondicional, por siempre estar presente en los momentos indicados, por confiar en mi y motivarme a no rendirme, porque desde que nací y hasta el ahora, han estado presentes en mi vida a pesar de las adversidades. También agradezco a mis hermanos porque también hicieron parte de este proceso con su apoyo.

A mi abuela materna por compartir tanto tiempo conmigo, por su dedicación, por su preocupación en mi bienestar. A mis abuelos paternos por su apoyo en este proceso.

A todos los profesores que me han ayudado a forjar este camino hacia la Ingeniería Electrónica, con ayuda de sus conocimiento e interés porque sus estudiantes aprendan, en especial al ingeniero Leonardo Rodríguez por ser mi líder en este proceso, por su paciencia, apoyo, por su ayuda en la busca de soluciones y compromiso con el proyecto.

A todas aquellas personas interesadas en mi bienestar y que siempre estuvieron dispuestas a ayudarme a resolver preguntas.

Resumen

El presente trabajo tiene como objetivo desarrollar una herramienta basada en aprendizaje automático que permita la estimación del estado de nitrógeno en las hojas de un cultivo de gulupa usando imágenes RGB y multiespectrales, para lo cual se llevó a cabo una metodología que incluye la toma de muestras, captura de imágenes con ayuda de una caja oscura y la cámara multiespectral Parrot Sequoia que entrega cinco imágenes, cuatro de ellas multiespectrales (rojo, verde, infrarrojo cercano y borde rojo) y una RGB, se llevaron las muestras al laboratorio para que su porcentaje de nitrógeno fuera evaluado por medio del método *kjeldahl*. La extracción de características de las muestras se realizó con ayuda de la biblioteca OpenCV implementada en el lenguaje de programación Python, se usaron técnicas de segmentación que permitieron conocer los datos únicamente del objeto interés, es decir, de la hoja. Esta segmentación fue útil para hallar los promedios de los índices de vegetación NDVI, GNDVI y OSAVI. También se extrajo características de textura de las cuatro imágenes multiespectrales generadas por la cámara.

Se implementaron algoritmos de aprendizaje automático que provee la biblioteca *sklearn*, y se utilizó el conjunto de datos obtenidos en la extracción de características para lograr la predicción entre dos clases, que fueron asignadas como dos rangos de porcentajes de nitrógeno, con base en los resultados de las pruebas químicas.

Abstract

The present work aims to develop a tool based on machine learning that allows the estimation of the nitrogen status in the leaves of a gulupa crop using RGB and multispectral images, for which a methodology was carried out that includes taking samples. , capturing images with the help of a dark box and the Parrot Sequoia multispectral camera that delivers five images, four of them multispectral (red, green, near infrared and Red Edge) and one RGB, the samples were taken to the laboratory so that their percentage nitrogen was evaluated by means of the textit kjeldahl method. The extraction of characteristics of the samples was carried out with the help of the OpenCV library implemented in the Python programming language, segmentation techniques were used that allowed knowing the data only of the object of interest, that is, of the sheet. This segmentation was useful to find the averages of the vegetation indices NDVI, GNDVI and OSAVI. Texture features were also extracted from the four multispectral images generated by the camera.

Machine learning algorithms provided by the textit sklearn library were implemented, and the data set obtained in the extraction of characteristics was used to achieve the prediction between two classes, which were assigned as two ranges of nitrogen percentages, based on in chemical test results.

Índice general

Índice de figuras	7
Índice de tablas	14
1. Capítulo 1. Contexto	18
1.1. Planteamiento del problema	20
1.2. Justificación	21
1.3. Alcances y limitaciones	22
1.3.1. Alcances	22
1.3.2. Limitaciones	22
2. Capítulo 2. Objetivos	24
2.1. Objetivo General	24
2.2. Objetivos Específicos	24
3. Capítulo 3. Marco de referencia	26
3.1. Estado del arte	26
3.2. Fundamentos teóricos	33

4. Capítulo 4. Metodología	52
4.1. Análisis	52
4.1.1. Sensor Parrot Sequoia	52
4.1.2. Lenguaje de programación en Python	55
4.1.3. Biblioteca OpenCV	56
4.1.4. Biblioteca Scikit-Learn	56
4.1.5. Software SolidWorks	56
4.2. Metodología	56
4.2.1. Recolección de muestras	58
4.2.2. Captura de imágenes	64
4.2.3. Extracción de características	70
4.2.4. Aprendizaje automático	95
5. Capítulo 5. Plan de trabajo y análisis de resultados	127
6. Capítulo 6. Presupuesto	133
Referencias	136

Índice de figuras

3.1. Parte superior de la caja de calibración	27
3.2. Parte interna de la caja de calibración	27
3.3. Parte frontal de la caja de calibración	28
3.4. Bombilla de Tungsteno Halógena LS-1-LL de 12 W. Citada por: . . .	28
3.5. Método de plantación.	30
3.6. Cámara oscura.	30
3.7. Método de toma de imágenes.	31
3.8. Diferencias de síntomas de hoja bajo diferentes niveles de N.	32
3.9. Cultivo de gulupa.	33
3.10. Espectro electromagnético.	41
4.1. Cuerpo de la cámara. Tomado:	53
4.2. Sensor de luz ambiental. Tomado:	54
4.3. Imagen en la banda del verde	54
4.4. Imagen en la banda del rojo.	54
4.5. Imagen en la banda del borde rojo.	55
4.6. Imagen en la banda del infrarrojo.	55

4.7. Imagen RGB.	55
4.8. Diagrama de flujo de la metodología.	57
4.9. Semitecho del cultivo.	58
4.10. Interior del cultivo con el semitecho.	59
4.11. Recolección de hojas para las muestras foliares.	60
4.12. Muestras colocadas en bolsa de papel.	61
4.13. Removido de tierra.	62
4.14. Hoyo en forma de “V” con 60cm de profundidad.	62
4.15. Mezclado de tierra.	63
4.16. Muestra de suelo empacada.	63
4.17. Medidas del cubo de la caja.	65
4.18. Medidas lámina interior de la tapa de la caja.	65
4.19. Plano vista superior de la tapa.	65
4.20. Plano vista inferior de la tapa.	65
4.21. Plano vista frontal de la tapa.	66
4.22. Plano vista posterior de la tapa.	66
4.23. Diseño del cubo	66
4.24. Tapa de la caja con los soportes.	66
4.25. Lámpara halógena dicróica (12V 10W) con conector Bi-pin.	67
4.26. Diseño físico de la caja oscura.	67
4.27. Diseño físico del interior de la caja oscura.	67
4.28. Diseño físico del interior de la tapa.	68

4.29. Toma de imágenes.	68
4.30. Diseño de la cámara en SolidWorks.	69
4.31. Diseño del banco de poder en SolidWorks.	69
4.32. Entrega de muestras en el laboratorio químico.	70
4.33. Imagen desalineada.	76
4.34. Imagen alineada.	76
4.35. Imagen creada en falso color basada en los índices NDVI.	78
4.36. Mascara basada en la imagen en falso color y segmentación.	79
4.37. Segmentación de la hoja.	80
4.38. Segmentación mejorada de la hoja.	81
4.39. Segmentación de imagen del canal verde.	82
4.40. Segmentación de imagen del canal del infrarrojo cercano.	82
4.41. Segmentación de imagen rojo.	82
4.42. Segmentación de imagen del canal borde rojo.	82
4.43. Imagen en falso color basado en el índice NDVI usando las imágenes multiespectrales segmentadas.	86
4.44. Imagen en falso color basado en el índice GNDVI usando las imágenes multiespectrales segmentadas.	86
4.45. Imagen en falso color basado en el índice OSAVI usando las imágenes multiespectrales segmentadas.	87
4.46. Textura de la imagen segmentada del canal verde.	93
4.47. Textura de la imagen segmentada del canal infrarrojo cercano.	93
4.48. Textura de la imagen segmentada del canal rojo.	94

4.49. Textura de la imagen segmentada del canal del borde rojo.	94
4.50. Diagramas de cajas y bigotes de los índices de vegetación.	101
4.51. Diagramas de dispersión de los índices de vegetación.	102
4.52. Matriz de confusión del algoritmo KNC.	108
4.53. Matriz de confusión del algoritmo LDA.	110
4.54. Gráfico de dispersión del promedio y la varianza de la textura.	113
4.55. Matriz de confusión del algoritmo KNC del promedio y la varianza de la textura.	114
4.56. Matriz de confusión del algoritmo LDA del promedio y la varianza de la textura.	115
4.57. Matriz de confusión del algoritmo DTC del promedio y la varianza de la textura.	116
4.58. Gráfico de dispersión de las características de promedio, varianza y desviación estándar de las imágenes RGB	120
4.59. Matriz de confusión del algoritmo LDA del promedio, varianza y des- viación estándar de imágenes RGB.	122
4.60. Matriz de confusión del algoritmo DTC del promedio, varianza y des- viación estándar de imágenes RGB.	123

Lista de algoritmos

1.	Función creada en la biblioteca <i>imutils</i> para calcular el promedio de los índices de vegetación.	71
2.	Función creada en la biblioteca <i>imutils</i> para generar falso color a partir de los índices de vegetación.	71
3.	Función creada en la biblioteca <i>imutils</i> para guardar datos en un documento con extensión csv.	72
4.	Código para indicar la ubicación de la carpeta con las imágenes.	72
5.	Ejecución desde la línea de órdenes del código principal de la extracción de características RGB, indicando la ubicación de la carpeta con imágenes y el nombre del archivo csv.	72
6.	Ejecución del código desde la línea de comandos.	73
7.	Bibliotecas utilizadas para la extracción de características de imágenes multiespectrales.	73
8.	Lectura del nombre del archivo csv y declaración de nombre de carpetas donde se guardarán las imágenes.	74
9.	Cargar imágenes.	75
10.	Alineación de las imágenes.	75
11.	Cálculo del NDVI para la segmentación.	77
12.	Establecimiento de falso color basado en los valores del índice de vegetación.	77

13.	Código para generar máscara.	79
14.	Filtros para mejorar la segmentación.	80
15.	Operación de la imagen segmentada con las imágenes multiespectrales alineadas.	81
16.	Cálculo de los índices de vegetación NDVI, GNDVI y OSAVI.	83
17.	Llamado de la función <i>prom_indices</i> de la librería <i>imutils</i>	84
18.	Función <i>prom_indices</i> de la librería <i>imutils</i>	84
19.	Código para imprimir índices en la línea de comandos y guardarlo en el archivo csv.	85
20.	Línea de comandos con los resultados de los índices.	85
21.	Código para guardar cada imagen en una carpeta.	87
22.	Establece 0 y 1.	93
23.	Promedio de la textura de la hoja.	94
24.	Varianza de la textura de la hoja.	95
25.	Bibliotecas usadas para el aprendizaje automático.	97
26.	Código para imprimir información del conjunto de datos.	98
27.	Resultado de la información del conjunto de datos.	99
28.	Generar diagrama de cajas y bigotes.	100
29.	Generar diagrama de dispersion de los indices de vegetacion.	101
30.	Resultados de matriz de correlación.	103
31.	División de datos.	104
32.	Entrenamiento de maquina usando los datos.	105
33.	Resultado del entrenamiento.	106

34.	Validación del entrenamiento del algoritmo de k vecinos más cercanos.	107
35.	Validación del entrenamiento del algoritmo de análisis de discriminante lineal.	109
36.	Matriz de correlación de los datos de promedio y varianza de la textura.	114
37.	Matriz de correlación de los datos de promedio, varianza y desviación estándar de las imágenes RGB.	121

Índice de tablas

3.1. Características de los tipos de iluminación artificial [16].	44
3.2. Técnicas de iluminación [16].	46
3.3. Estructura de una matriz de confusión [24].	48
4.1. Muestras del surco 1.	63
4.2. Muestras del surco 3.	63
4.3. Medidas de los elementos que conforman la caja negra	64
4.4. Costos totales de la caja oscura.	69
4.5. Índices de vegetación de la muestra S1HGA.	88
4.6. Índices de vegetación de la muestra S1HGV.	88
4.7. Índices de vegetación de la muestra S1HPA.	89
4.8. Índices de vegetación de la muestra S1HPV.	89
4.9. Índices de vegetación de la muestra S3HGA.	90
4.10. Índices de vegetación de la muestra S3HGV.	90
4.11. Índices de vegetación de la muestra S3HPA.	91
4.12. Índices de vegetación de la muestra S3HPV.	91
4.13. Resultados de las pruebas químicas.	96

4.14. Etiquetas del conjunto de muestras.	96
4.15. Matriz de confusión del algoritmo KNC.	108
4.16. Matriz de confusión del algoritmo LDA.	110
4.17. Resultados obtenidos de la clasificación de datos nuevos utilizando el modelo KNC para los índices de vegetación.	111
4.18. Resultados obtenidos de la clasificación de datos nuevos utilizando el modelo LDA para los índices de vegetación.	112
4.19. Matriz de confusión del algoritmo KNC para datos de textura.	115
4.20. Matriz de confusión del algoritmo LDA para datos de textura.	115
4.21. Matriz de confusión del algoritmo DTC para datos de textura.	116
4.22. Porcentajes de comportamiento de los modelos KNC, LDA y DTC para los datos de textura.	117
4.23. Resultados obtenidos de la clasificación de datos nuevos de textura utilizando el modelo KNC.	117
4.24. Resultados obtenidos de la clasificación de datos nuevos de textura utilizando el modelo LDA.	118
4.25. Resultados obtenidos de la clasificación de datos nuevos de textura utilizando el modelo DTC.	119
4.26. Matriz de confusión del algoritmo LDA para datos promedio, varianza y desviación estándar de imágenes RGB	122
4.27. Matriz de confusión del algoritmo DTC para datos promedio, varianza y desviación estándar de imágenes RGB.	123
4.28. Porcentajes de comportamiento de los modelos LDA y DTC para los datos de promedio, varianza y desviación estándar de imágenes RGB.	123
4.29. Resultados obtenidos de la clasificación de datos nuevos de promedio, varianza y desviación estándar de imágenes RGB utilizando el modelo LDA.	124

4.30. Resultados obtenidos de la clasificación de datos nuevos de promedio, varianza y desviación estándar de imágenes RGB utilizando el modelo DTC.	125
5.1. Plan de trabajo.	131
6.1. Costos totales del proyecto.	133

Capítulo 1

Contexto

La gulupa es en nuestro país un fruto que genera grandes ingresos, pues se encuentra entre los frutos cítricos de mayor exportación debido a que es muy apetecido en mercados europeos, no solo por su fruto si no también por su flor. Estas son algunas razones por las cuales se debe mantener en óptimas condiciones nutricionales un cultivo de gulupa, teniendo presente el cuidado del medio ambiente en el momento de aplicar los fertilizantes. El nitrógeno hace parte de los macronutrientes presentes en las plantas, este permite su crecimiento y además, les da su color verde característico. En caso de que se presente deficiencia de este nutriente ocasionaría que la planta no tenga un crecimiento adecuado.

El presente trabajo tiene la modalidad de auxiliar de investigación y se pretende evidenciar el desarrollo de una herramienta basada en aprendizaje automático que permita la estimación del estado de nitrógeno presente en la hoja de un cultivo de gulupa usando imágenes RGB y multiespectrales, este es la continuación de los proyectos de los ingenieros egresados de la Universidad de Cundinamarca, Santiago Alejandro Trujillo Fandiño quien es su proyecto “Implementación de un sistema capaz de facilitar la identificación del cambio de color en la hoja de gulupa conforme a la presencia o ausencia de nitrógeno mediante el procesamiento de imágenes RGB” [1] evidenció la extracción de características de las imágenes RGB de la hoja de la gulupa, y de Faryd Alejandro Peñuela González con el proyecto “Implementación de un sistema capaz de calcular el área foliar de una planta de gulúpa, a partir

de imágenes que representen dos dimensiones de la planta, mediante técnicas de procesamiento de imágenes” [2].

1.1. Planteamiento del problema

En Colombia la gulupa se cultiva entre los 1800 y 2400 msnm y su fruto se considera promisorio, además de que cuenta con alto potencial en el mercado colombiano y de exportación, es muy apetecida en el mercado internacional debido a su sabor, apariencia, disponibilidad además de su conotación exótica [3]. De acuerdo a informes de Asohfrucol, la gulupa se encuentra dentro de los cultivos frutales más rentables en Colombia por sus precios fuertes y estables, además pertenece al portafolio de exportación de frutas exóticas de nuestro país [4]. Debido a la importancia que tiene esta fruta, es importante que sea cultivada en óptimas condiciones nutricionales, teniendo presente el cuidado del medio ambiente.

El nitrógeno es uno de los macronutrientes presente en las plantas, el cual se evidencia logrando que las plantas posean un color verde vivo, además contribuye al incremento del área foliar. Al ser un macro nutriente, es importante que los agricultores tengan un manejo adecuado del nitrógeno, pues en caso de deficiencia la planta presenta problemas en su crecimiento además de hojas cloróticas, y por otro lado en exceso, podría presentar retraso en la producción o estar expuesta a enfermedades.

En la mayoría de cultivos de gulupa no se realiza un estudio previo de la planta que permita conocer su estado de nitrógeno antes de aplicar los fertilizantes, lo cual puede presentar un impacto ambiental desfavorable al no aplicarse la cantidad correcta de fertilizante, debido a que los nitratos no se adhieren fuertemente al suelo y con el flujo del agua presentan alta movilidad haciendo que los nitratos recorran el suelo hasta llegar a acuíferos [5]. Los cultivos en que si se hacen un estudio o prueba de laboratorio deben esperar de 6 a 12 días hábiles para conocer el resultado, lo cual lo hace poco eficiente.

Por estas razones es importante realizar la siguiente pregunta: ¿Cómo desarrollar una herramienta basada en aprendizaje automático, que permita la estimación del estado nitrógeno presente en las hojas de un cultivo de gulupa usando imágenes RGB y multispectrales?

1.2. Justificación

Este proyecto es realizado con la finalidad de estimar el estado del nitrógeno presente en las hojas de un cultivo de gulupa por medio de imágenes RGB y multiespectrales, de esta manera se obtendría una estimación más rápida que con el método tradicional, motivando a los agricultores a conocer el estado de nitrógeno de las plantas antes de la fertilización y así, partir de esos resultados para la toma de decisiones.

La razón por la que se usan imágenes multiespectrales, es por que estas contienen más información en el espectro visible o en la no visible que las imágenes RGB [6], permitiendo extraer características importantes en la agricultura como los índices de vegetación.

1.3. Alcances y limitaciones

1.3.1. Alcances

Con la herramienta basada en aprendizaje automático se pretende lograr una estimación del estado de nitrógeno en las hojas de un cultivo de gulupa por medio de imágenes RGB y multiespectrales, siguiendo el método que incluye la toma de muestras, el uso de una caja oscura, pruebas químicas y extracción de características. La clasificación será de dos clases (0.46 % y 0.76 % y 0.80 % y 1.20 %) basadas en rangos de nitrógeno obtenidos de las pruebas químicas realizadas.

1.3.2. Limitaciones

En primera instancia se menciona el limitante que se presentó en cuanto a la etiqueta que se le asigna a cada hoja basada en los resultados de las pruebas químicas, debido a que estas últimas solo se pueden realizar con una cantidad de gramos mínimo de hojas, por lo cual una muestra se debe conformar de varias hojas, tornándose difícil realizar la prueba química de manera individual. Esto impedía que cada hoja tuviera la etiqueta del contenido de nitrógeno real.

Además, se condiciona el uso de algoritmos de aprendizaje más eficientes, debido a que requieren un equipo computacional de alto rendimiento. También se tiene en cuenta que la estimación del nitrógeno no será de manera inmediata, pues primero se debe hacer la captura de la imagen y extraer sus características para luego para luego ser ingresadas en el programa que estimará si hay o no contenido de nitrógeno.

Otra limitante fue el no contar con un cultivo controlado de gulupa, pues esto hacía un poco ardua la labor de la identificación de las hojas con síntomas de deficiencias nutricionales en cuanto al nitrógeno.

Capítulo 2

Objetivos

2.1. Objetivo General

Desarrollar una herramienta basada en aprendizaje automático que permita la estimación del estado de nitrógeno presente en las hojas de un cultivo de gulupa usando imágenes RGB y multiespectrales.

2.2. Objetivos Específicos

- Estudiar diferentes métodos de aprendizaje automático que permitan estimar el estado de nitrógeno en cultivos de gulupa.
- Diseñar un método que permita estimar la deficiencia de nitrógeno basada en aprendizaje automático.
- Implementar el método de aprendizaje automático que permita estimar la deficiencia de nitrógeno a partir de imágenes multiespectrales.
- Validar el método diseñado mediante la comparación de los resultados entregados por un método convencional.

Capítulo 3

Marco de referencia

3.1. Estado del arte

Estimación del nitrógeno

En [7] mencionan que al realizar un uso más efectivo del abono puede evitar un impacto ambiental, este trabajo busca dosificar de la mejor manera el abono aplicado a los cultivos. Aluden a la relación estrecha que existe entre la concentración de clorofila en la hoja y el contenido de nitrógeno foliar, pues la ausencia de nitrógeno provoca una disminución en el contenido de clorofila, además de los síntomas típicos. El estado nutricional del nitrógeno en las plantas se puede determinar indirectamente por la concentración de clorofila. Aproximadamente el 10% del nitrógeno total de la planta se encuentra almacenado en las moléculas de clorofila, usaron medidores de clorofila como indicador del estrés causado por el nitrógeno en las plantas. Además mencionan que existe una correlación positiva entre la capacidad fotosintética y el contenido de nitrógeno foliar para cada especie. Usaron un radioespectrómetro STS-VIS, un sensor multiespectral MicaSense RedEdge que permitía comprobar el estado saludable del cultivo donde recogerían las muestras. Las mediciones las realizaron en campo de cultivo entre las 10:30 y las 12:00 con un ángulo fijo de visión a 90 grados. Colocaron debajo del aparato un plástico de color para evitar la pérdida de piezas pequeñas. Realizaron entre 5 y 10 mediciones en las mismas condiciones, las

tomas de las muestras las realizaron en una caja de calibración de plástico diseñada en SOLIDWORKS, con dimensiones 30x15x10 cm, además adhirieron cartulinas de color blanco en las paredes de la caja.

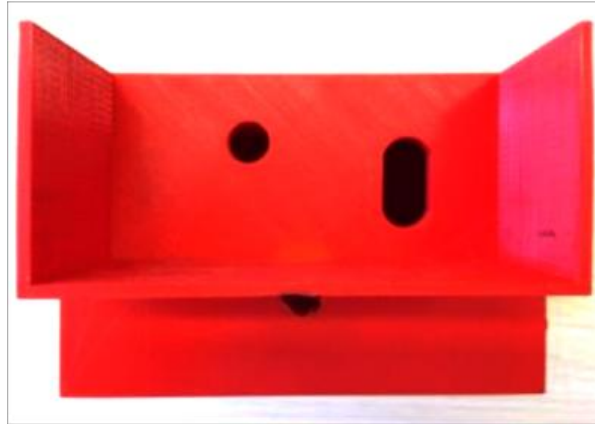


Figura 3.1: Parte superior de la caja de calibración
[7]

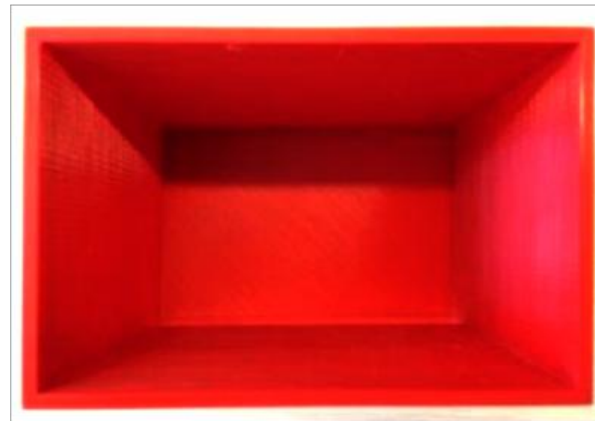


Figura 3.2: Parte interna de la caja de calibración
[7]

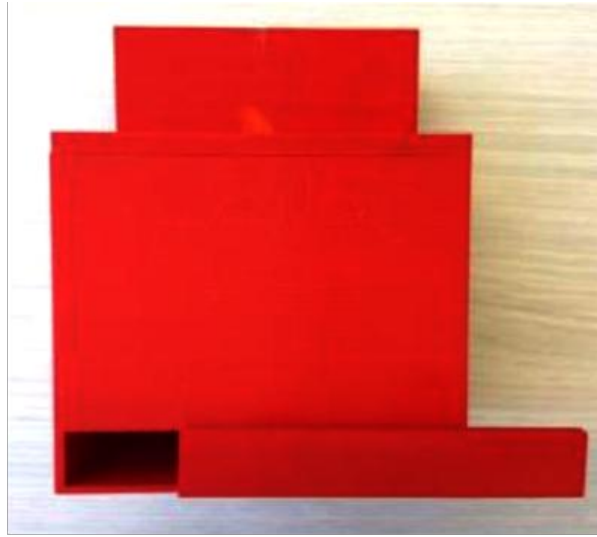


Figura 3.3: Parte frontal de la caja de calibración
[7]

Incorporaron también una bombilla de Tungsteno Halógena LS-1-LL de 12 W con el fin de no depender de un factor indispensable como el sol.



Figura 3.4: Bombilla de Tungsteno Halógena LS-1-LL de 12 W. Citada por:
[7]

La finalidad de utilizar dicha caja, fue evitar interferencias externas como la nubosidad, el viento y la elevación del sol.

Los autores mencionan que hicieron la recolecta de 3 muestras foliares de cada uno de 30 árboles diferentes, cada muestra la recogieron de 3 partes diferentes del árbol, es decir, de la parte alta, media y baja, para un total de 90 muestras, cada una tenía un peso aproximado de 20gr. Después de esto llevaron a cabo la captura de las imágenes

y luego realizaron las pruebas químicas a las hojas con el *Método Dumas*. En este documento recomiendan la realización de pruebas a mayor cantidad de muestras y en otras parcelas, pues los resultados gráficos no tenían un porcentaje de relación considerable.

Por otra parte [8] es un artículo que muestra un modelo para la estimación del nitrógeno total contenido en las hojas de sándalo. Además, explica que el sándalo es reconocido por su uso en fragancias y productos farmacéuticos, pero debido a la sobreexplotación y la destrucción de su ambiente, su producción ha disminuido de manera drástica a nivel mundial, además de causar un gran daño ambiental por el uso inadecuado de fertilizantes y del exceso de nitrógeno administrado. Afirman que, entre las tecnologías de diagnóstico de nutrición no invasiva como la tabla de colores de las hojas, el medidor de clorofila, el espectrómetro de mano, la detección remoto y la tecnología de procesamiento de imágenes digitales, son estas dos últimas las más confiables. También hablan de que algunos resultados han demostrado que los cambios en el color de la planta se correlacionaron significativamente con el contenido de clorofila total, además el contenido de nitrógeno total afectó fuertemente el contenido de clorofila.

Para llevar a cabo este proyecto los autores tomaron muestras de 3 granjas diferentes de la provincia de Hainan. La duración solar es diferente en cada parcela lo que afecta la tasa de fotosíntesis. Seleccionaron plántulas de sándalo de 3 años de edad con el mismo estado de crecimiento. Establecieron la distancia que habría entre cada plántula para que no compitieran por la luz del sol o el espacio entre ellas. Se recolectaron aproximadamente 5 g por muestra, las cuales se colocaron en una bolsa para mantenerlas frescas, se transportaron al laboratorio para la toma de imágenes multiespectrales y la determinación de la composición química.

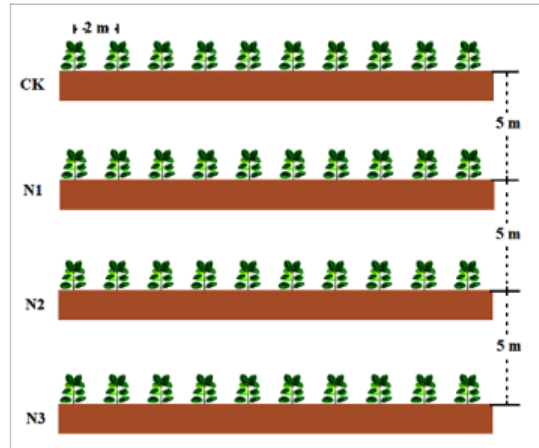


Figura 3.5: Método de plantación.
[8]

Para la toma de las imágenes utilizaron una cámara RedEdge 3, MicaSense, que puede detectar las bandas, B, G, R, RE Y NIR y las hojas fueron colocadas en una cámara oscura. La longitud de onda central de cada banda era de 475 nm, 560 nm, 668 nm, 717 nm y 840 nm, y el ancho de banda era de 20 nm, 20 nm, 10 nm, 40 nm y 10 nm para B, G, R, RE y NIR, respectivamente.



Figura 3.6: Cámara oscura.
[8]

La fuente de luz que usaron en la cámara oscura fue una lámpara LED de luz única que contaba de 10 bombillas cuyas longitudes de onda centrales correspondían a la de la cámara. La operación de corrección se realizó usando un tablero blanco y uno negro y la reflectancia espectral de cada banda se calculó de la siguiente manera:

$$I_c = \frac{O - B}{W - B} * 100\% \quad (3.1)$$

Donde I_c es la reflectancia corregida y O , B y W son los valores medios de los objetos, y los tableros negro y blanco respectivamente.

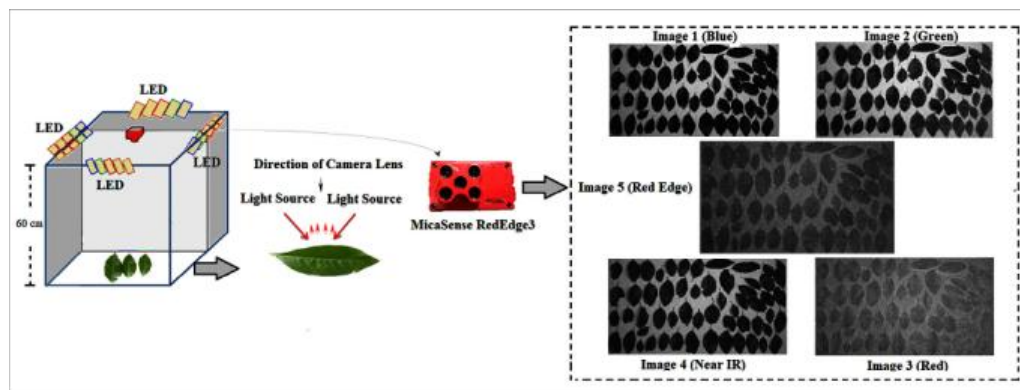


Figura 3.7: Método de toma de imágenes.

[8]

En este proyecto usaron BPNN o Back Propagation Neuronal Network el cual es utilizado para entrenar redes neuronales artificiales [9]. En el artículo mencionan que BPNN utiliza el método del descenso del gradiente dependiendo de la señal de error de la propagación hacia adelante.

En [10] realizan un diagnóstico rápido del estado nutricional del nitrógeno en el arroz basado en escaneo estático y extracción de características de hojas y vainas. Los autores aluden que en el momento en que en un cultivo de arroz se presentan deficiencias de nitrógeno sería notorio un cambio de color, el tamaño y la forma de las hojas y la vaina. En esta investigación los autores llevaron a cabo la tecnología de escaneo estático para recolectar las imágenes de hojas y vainas del arroz. Este

estudio lo hicieron bajo diferentes estados de nutrición de nitrógeno, las semillas las colocaron en arena húmeda por unos días a 30 °C.



Figura 3.8: Diferencias de síntomas de hoja bajo diferentes niveles de N.
[10]

Después las plántulas fueron trasplantadas dentro de macetas de 5L de PVC, cada una contenía arena de río limpia, tamizada y completamente lixiviada para permitir el control preciso de los nutrientes. La captura de las imágenes de las hojas y las vainas las realizaron después de 35 días, el experimento lo realizaron en un invernadero al que mantenían a una temperatura de 30 °C y 25 °C de día y de noche, la humedad relativa fue mantenida en 50%.

La adquisición de las imágenes fue con un escáner EPSON GT 20000, las tomaron el 4, 18, 27 de agosto y 8 de septiembre de 2013. En total recolectaron 600 muestras para todas las etapas de crecimiento.

Utilizaron el método de selección de características de vectores de soporte (SVFS) para seleccionar el conjunto de características óptimas.

3.2. Fundamentos teóricos

La gulupa

La gulupa es también conocida como la fruta de la pasión púrpura o maracuyá morado. A pesar de que esta fruta es conocida y consumida hace mucho tiempo, solo en los últimos años ha tenido un aumento en su producción en cultivos comerciales, con el fin de atender la demanda nacional e internacional especialmente el mercado europeo. Productores con experiencia en cultivos de otras pasifloras como el maracuyá, la granadilla y la curuba han implementado las técnicas de estos cultivos en la producción de la gulupa, las cuales han tenido gran éxito, aunque enfrentando problemáticas como la conocida ola invernal.

La gulupa es cultivada entre los 1600 y 2400 metros sobre el nivel del mar (msnm), con temperaturas entre los 10°C y 18°C. Su ciclo vegetativo dura aproximadamente de 1 a 3 años. La producción inicia entre los 7 y los 12 meses después del trasplante. La gulupa ha llegado a ser una buena alternativa económica para agricultores y empresarios de Boyacá, Antioquia, Huila, Tolima, Santander, Magdalena y Cundinamarca.

Los cambios de los ciclos del agua son una amenaza para los productores y comercializadores de gulupa. Por ejemplo, el exceso de humedad asociado a la Ola invernal, hace que las enfermedades se propaguen con severidad. Las enfermedades pueden ocasionar daños en las raíces, tallos, hojas, flores y frutos de la gulupa, afectando la calidad y volumen de la cosecha [11].



Figura 3.9: Cultivo de gulupa.
[12]

Fotosíntesis y su relación con los nutrientes

En [13] citan que en el efecto fotosintético se involucran variables donde cada una ejerce una función, pero también depende del estado nutricional de la planta, debido a que la maquinaria fotosintética absorbe la mitad del nitrógeno foliar, indicando que una de las grandes limitantes que tiene la fotosíntesis es la ausencia de nitrógeno, independientemente de la causa de dicha ausencia. También citan, que uno de los procesos de la fotosíntesis donde el nitrógeno cumple un papel muy importante, es el intercambio de gases, pues en este proceso la concentración de dióxido de carbono y disponibilidad de nitrógeno hacen un ciclo básico para el nutrimento.

Importancia del nitrógeno en el crecimiento de las plantas

El nitrógeno es un elemento necesario en la composición de proteínas, ácidos nucleicos y otros componentes celulares, siendo esta una molécula esencial para el crecimiento de todos los organismos. Aunque es extremadamente común (80 % por volumen) en la atmósfera en forma de gas (N_2) es generalmente inaccesible biológicamente debido a su alta energía de activación. Aproximadamente el 78 % de la atmósfera está formado por N_2 , sin embargo, el nitrógeno atmosférico representa sólo el 1.2 % del nitrógeno que hay sobre el planeta. Se debe tener en cuenta que el nitrógeno lo podemos encontrar presente en rocas y minerales, que representan un 98 % aproximadamente de total de nitrógeno que hay en la Tierra [14].

El nitrógeno es muy importante en el crecimiento de las plantas, la ausencia de este puede representar problemas en la producción agrícola. El nitrógeno se absorbe en la planta como NO_3 o como amonio NH_4 , el nitrógeno cumple la función de incrementar el crecimiento del tallo y la formación de frutos, pero en caso de que haya exceso de este elemento se puede presentar un crecimiento desproporcionado al follaje, como lo citan en [13].

Agricultura de precisión

Según [15], la agricultura de precisión son técnicas utilizadas con el fin de optimizar el uso de los insumos agrícolas, es decir, semillas, agroquímicos y correctivos, en función de la variabilidad espacial y temporal de la producción agrícola. La optimización se logra haciendo un uso adecuado de los insumos según la necesidad de los cultivos.

Imágenes digitales

Las imágenes son representaciones bidimensionales de un espacio tridimensional, resultado de adquirir la señal proporcionada por un sensor que convierte la información del espectro electromagnético en codificaciones numéricas. Una imagen digital es una matriz o vector de $N \times M$, cada elemento de la matriz se conforma por un valor discreto que indica el nivel de información de cada elemento, representado por un número finito de bits. El valor que representa cada elemento indica valores de luminosidad desde el más bajo y oscuro hasta el más alto y claro. El valor más bajo está representado por el negro y el valor más alto por el blanco.

Una imagen es una función bidimensional que proporciona información electromagnética para cada uno de sus valores. Cada uno de estos elementos se les denomina píxel o punto los cuales forman la imagen o fotografía. Se dice que una imagen es fruto de la representación espectral recibida por el sensor. El color de una imagen se genera por la superposición de tres componentes espectrales.

El proceso para la formación y captura de imágenes consiste en generar una señal electromagnética en forma de luz que se transmite refracta y refleja en la escena hasta que alcanza el sensor de la cámara. En el sensor se produce la transducción a una señal eléctrica y a partir de ese momento se cuenta con un conjunto de elementos electrónicos que digitalizan, formatean, transmiten y almacenan imágenes digitales. Dentro del primer conjunto de estos elementos se encuentran la fuente de iluminación, el medio de la escena (generalmente aire), los objetos de la escena, los filtros o difusores y las ópticas. Los elementos electrónicos que se encuentran a partir del sensor son la electrónica de la cámara, los cables de transmisión para un determinado protocolo de comunicación y las tarjetas de adquisición conectadas al bus del procesador [16].

Imágenes multiespectrales

Una imagen multiespectral se compone por varias imágenes monocromas que normalmente son capturadas con diferentes sensores. Cada imagen y sensor monocromático corresponde a una longitud de onda específica que también es llamada banda o canal. A diferencia de las imágenes RGB, las imágenes multiespectrales contienen mucha más información en la longitud de onda visible o la no visible.

El proceso de las imágenes multiespectrales se puede describir de la siguiente manera: Primero, la fuente de iluminación se refleja desde la superficie de los objetos y pasa a través de la lente de la cámara y luego la luz incidente se separa en varios filtros frente al sensor de imágenes [6].

Bandas espectrales

Banda roja (Red): La reflectancia que tiene el suelo depende de su composición, por lo cual, los suelos que contienen grandes cantidades de óxido de hierro tienen una reflectancia alta en esta banda debido a su color. La banda roja se suele utilizar para diferenciar objetos. Es una banda de absorción de clorofila muy útil para la clasificación de cubierta vegetal. La longitud de onda es de 0.64 a 0.67 μm [17].

Banda verde (Green): La parte verde de reflectancia cubre la parte superior de las superficies foliares. Esta banda se utiliza para discriminar amplias clases de vegetación y el material vegetal. Por lo tanto esta banda está diseñada para evaluar la vigorosidad de la vegetación sana, midiendo su reflectancia verde. La longitud de onda es de 0.53 a 0.59 μm [17].

Banda infrarrojo cercano (NIR): Esta banda permite clasificar la vegetación saludable, pues con la reflectancia NIR se puede discriminar el agua, la vegetación y otros objetos, debido a que el espectro del infrarrojo cercano es reflejado por las plantas sanas, mientras que es absorbido por el agua y otros objetos. Además el infrarrojo cercano se utiliza para evaluar la salud de las plantas, debido a que la estructura interna de la clorofila sana refleja la radiación infrarroja cercana, cuando

las plantas se marchitan sucede lo contrario, es decir, la respuesta de la banda NIR disminuye. La longitud de onda es de 0.76 a 0.90 μm [17].

Banda borde rojo (Red Edge): Esta banda se encuentra entre la banda roja y la banda del infrarrojo cercano, tiene alta reflectividad lo cual indica la salud y vitalidad de las plantas. La respuesta de vegetación desde esta banda es a menudo mayor tanto para el contenido de clorofila como para la estructura de la hoja. La banda de borde rojo es aplicada en la agricultura para identificar los tipos de cultivos y su nutrición. La longitud de onda es de 0.705 a 0.745 μm [17].

Índices de vegetación

Los índices de vegetación se refieren a un conjunto de operaciones algebraicas que se efectúan sobre los valores numéricos de los píxeles usando dos o más bandas pertenecientes a la misma escena. Un índice de vegetación puede ser definido como un parámetro calculado a partir de los valores de reflectancia a distintas longitudes de onda y que es particularmente sensible a la cubierta vegetal. También corresponde a un número generado por alguna combinación de bandas espectrales y que puede tener alguna relación con la cantidad de vegetación presente en un píxel dado.

Los valores bajos de los índices de vegetación usualmente indican vegetación poco vigorosa, mientras que los valores altos indican vegetación muy vigorosa. Sin embargo, algunos valores del índice de vegetación como los de RVI y NRVI son inversamente proporcionales a la cantidad de vegetación presente en el área [18].

Índice de Vegetación Ratio (RVI): Este índice puede usarse para estimar y monitorear la biomasa aérea, es efectivo para la estimación de la biomasa especialmente en áreas con vegetación densa. Se calcula con la siguiente ecuación [18]:

$$RVI = \frac{Rojo}{Infrarrojo\ cercano} \quad (3.2)$$

Índice de Vegetación de Diferencia (DVI): Según [19] este índice se desarrolló para distinguir entre el suelo y vegetación, y como su nombre lo indica es una

ecuación de diferencia simple entre las bandas roja e infrarroja cercana [19], su ecuación se muestra a continuación:

$$DVI = \text{Infrarrojo cercano} - \text{Rojo} \quad (3.3)$$

Índice de Vegetación de Diferencia Normalizada (NDVI): Fue desarrollada como un índice de “verdor” de las plantas e intenta rastrear la actividad fotosintética. Es uno de los índices más aplicados. Al igual que el RVI y el DVI, el NDVI también se basa en el principio de que las plantas vivas bien nutridas absorben la luz roja y reflejan la luz del infrarrojo cercano. También tiene en cuenta el hecho de que la vegetación estresada o muerta absorbe comparativamente menos luz roja que la vegetación sana, el suelo desnudo refleja la luz roja e infrarroja cercana casi por igual, y el agua abierta absorbe más luz infrarroja que roja. El NDVI es un valor relativo y no se debería usar para comparar imágenes tomadas en diferentes momentos o desde diferentes sensores. Los valores de este índice de vegetación oscilan entre -1 y 1, donde los valores positivos más altos indican la presencia de plantas más verdes y saludables [19]. La ecuación para el cálculo de este índice es:

$$NDVI = \frac{\text{Infrarrojo cercano} - \text{Rojo}}{\text{Infrarrojo cercano} + \text{Rojo}} \quad (3.4)$$

NDVI sintético: Es un índice que intenta predecir los valores del NDVI utilizando solo las bandas roja y verde. Por lo tanto, se puede aplicar a imágenes recopiladas de cualquier sensor RGB, incluidos los utilizados en drones de nivel de consumidor. Al igual que el NDVI, sus valores también oscilan entre -1 y 1, y los valores más altos sugieren la presencia de las plantas más sanas. Sin embargo, no es tan preciso como el NDVI y debe calibrarse usando información del terreno para que sea realmente útil. También se conoce como Índice de Vegetación Verde Rojo (GRVI) [19].

$$GRVI = \frac{\text{Verde} - \text{Rojo}}{\text{Verde} + \text{Rojo}} \quad (3.5)$$

Índice de Vegetación de Diferencia Visible (VDVI): Este índice también se puede calcular utilizando información de solo la parte visible del espectro electromagnético. Algunos estudios indican que VDVI es mejor para extraer información

de vegetación y predecir NDVI que otros índices de solo RGB [19]. La ecuación es la siguiente:

$$VDVI = \frac{((2 * Verde) - Rojo - Azul)}{(2 * Verde) + Rojo + Azul} \quad (3.6)$$

Excess Green Index (ExGI): contrasta la parte verde del espectro con el rojo y el azul para distinguir la vegetación del suelo y también se puede utilizar para predecir los valores del NDVI [19].

$$ExGI = (2 * Verde) - (R + B) \quad (3.7)$$

Índice de Vegetación Ajustado del Suelo (SAVI): Es una versión modificada del NDVI diseñada específicamente para áreas con muy poca cobertura vegetal, generalmente menos del 40% por área. Según el tipo y el contenido de agua, los suelos reflejan cantidades variables de luz roja e infrarroja. El SAVI explica esto suprimiendo los píxeles del suelo desnudo [19].

$$SAVI = \left[\frac{(Infrarrojo cercano - Rojo)}{Infrarrojo cercano + Rojo + L} \right] * (1 + L) \quad (3.8)$$

Donde L es una función de la densidad de Vegetación, el cálculo de L requiere información a priori sobre la presencia de Vegetación en el área de estudio. Va desde 0-1, con mayores coberturas de vegetación resultando valores cercanos a 1.

Índice de absorción de clorofila modificada en reflectancia (MCARI): Se desarrolló como un índice del estado de la vegetación. El índice de absorción de clorofila en reflectante se diseñó en primera instancia para distinguir el material no fotosintético de la vegetación fotosintéticamente activa. El MCARI es una modificación de este índice y se define como la profundidad de la absorción de clorofila en la región roja del espectro en relación con la reflectancia en las regiones verde y de borde rojo [19].

$$MCARI = (Borde\ rojo - Rojo) - 0,2 * (Borde\ rojo - Verde) * \left(\frac{Borde\ rojo}{Rojo} \right) \quad (3.9)$$

Fracción de vegetación: se define como el porcentaje de vegetación que ocupa el área del suelo, dado que se calcula utilizando valores generados a partir de un NDVI, está sujeto a los mismos errores.

$$\text{Fracción de vegetación} = \frac{[NDVI - NDVI(min)]}{NDVI(max) - NDVI(min)} \quad (3.10)$$

Procesamiento digital de imágenes

Los seres humanos estamos limitados a la banda visible del espectro electromagnético (EM), las máquinas pueden percibir casi el espectro completo, desde los rayos gamma hasta las ondas de radio [20].

Proceso de Bajo Nivel : Utilizan operaciones como el pre-procesamiento de imagen para reducir el ruido, mejora del contraste y filtros de enfoque. Se caracterizan porque sus entradas son imágenes y sus salidas también.

Proceso de Nivel Medio : Realizan operaciones como segmentación y clasificación de objetos individuales. Generalmente se caracterizan por tener como entradas imágenes pero sus salidas son atributos extraídos de esas imágenes.

Proceso de Alto Nivel : Implica el obtener algún significado de un conjunto de objetos reconocidos (análisis de imágenes) y realizar funciones cognitivas asociadas con la vista.

En 1666, Isaac Newton descubrió que al pasar un rayo de luz por un prisma de vidrio, el rayo de salida no es blanco si no que está compuesto de un espectro de colores continuo, que va desde el violeta hasta el rojo.

El rango de colores que los humanos percibimos es solo una muy pequeña parte del espectro electromagnético (EEM). En un extremo del espectro se encuentran las ondas de radio con longitudes de onda mil millones de veces más largas que los de la luz visible. En el otro lado están los rayos gama con longitudes de onda millones de veces más pequeños que los de la luz visible [20].

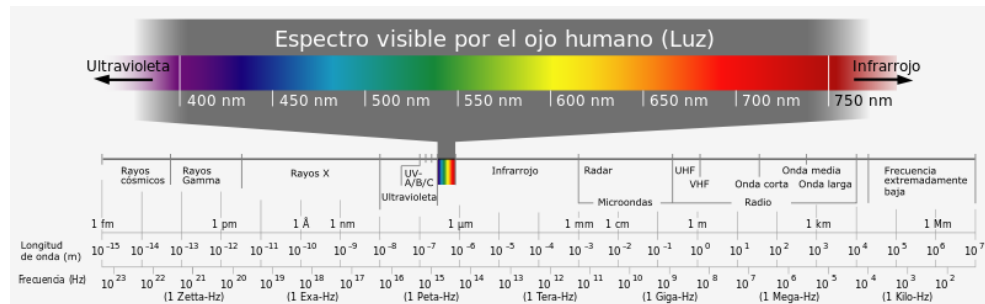


Figura 3.10: Espectro electromagnético.
[21]

Los colores que percibimos en un objeto son determinados por la naturaleza de la luz reflejada por dicho objeto. Un cuerpo reflejante que está balanceado en todas las longitudes de onda visibles aparece blanco para el observador. Un cuerpo que refleja un rango en particular del espectro visible se ve de cierto color.

Luz monocromática: La luz que no tiene color se llama o monocromática. Su único atributo es su intensidad o cantidad. En general se usa el término de nivel de grises para definir la intensidad monocromática porque esta va desde el negro hasta el blanco, pasando por una gama de grises.

Luz cromática: Esta se trata de la luz de color, para la cual se usan tres cantidades para describir la calidad de una fuente cromática:

- Radiancia: cantidad total de energía que fluye de una fuente de la luz. Se mide en watts.
- Luminancia: Cantidad de energía que un observador percibe de una fuente de luz. Se mide en lúmenes.
- Brillo: engloba la noción de intensidad.

Iluminación

Según [16] en el diseño de sistemas de visión industriales la iluminación no controlada que suelen tener de las escenas no es la mejor, debido a que se obtienen imágenes con

bajo contraste, reflexiones especulares, sombras y destellos. Si se tiene un sistema de iluminación bien diseñado, permite una obtención de imagen idónea que permite un buen procesamiento y extracción de características de la misma.

Iluminación artificial: Esta iluminación hace referencia a la utilizada para mejorar la captura de imágenes, dependiendo de la aplicación que vaya a tener. Existen diferentes tipos de iluminación con características propias cada una, como el rango de longitud de onda en que emite la luz, durabilidad de la misma, variaciones a lo largo del tiempo y la temperatura que pueden tomar. A continuación se muestra una tabla con las características de los tipos de iluminación artificial [16].

Tipo de iluminación	Ventajas	Inconvenientes
Incandescente/Halógena	<p>Bajo coste y Fáciles de utilizar.</p> <p>Permiten ajustar la intensidad de luz.</p> <p>Oscila 50 veces por segundo.</p>	<p>Desprenden gran cantidad de calor.</p> <p>Su espectro se centra en el rojo siendo deficiente para azules verdes o amarillos.</p>
Fluorecentes	<p>Calienta menos que el incandescente</p> <p>Su espectro se centra en los colores del ojo humano.</p> <p>La duración está estimada en torno a 10.000 horas.</p>	<p>La longitud de onda de la luz cambia con el uso.</p> <p>Para que sean válidos en aplicaciones industriales tienen que trabajar a una frecuencia del orden de 25 KHz.</p>
Led	<p>Gran durabilidad. (100.000 horas)</p> <p>Posibilidad de encender y apagar solamente en el tiempo de captura de la imagen.</p> <p>Fácil elección de la longitud de onda de la fuente de luz dentro del espectro visible e infrarrojo.</p> <p>Las fuentes de luz se pueden construir en multitud de formas.</p>	Precio.
Láser	<p>Se utilizan para generar luz estructurada con forma diversas tales como líneas, líneas paralelas, líneas cruzadas, retículas, puntos y matriz de puntos. Para generar las formas se utilizan ópticas específicas.</p>	Precio

	<p>Están disponibles en multitud de longitud de ondas desde el visible al infrarrojo cercano.</p> <p>Dado que el ojo humano es muy sensible al verde, un diodo láser en esta longitud de onda genera un mejor contraste en los bordes especialmente sobre superficies rojas.</p>	
Fibra óptica	<p>Se utiliza para llevar la luz a cualquier punto distante de la fuente de luz.</p> <p>Permite iluminar pequeñas áreas.</p> <p>Proporciona luz fría, es decir, no se calienta.</p>	<p>Precio.</p> <p>Sólo sirve para iluminar pequeñas áreas.</p>

Tabla 3.1: Características de los tipos de iluminación artificial [16].

Técnicas de iluminación

El buen funcionamiento de una aplicación de visión por computador, está muy ligada a la iluminación, por lo cual es muy importante la técnica utilizada para iluminar la escena, pues en caso de no hacer uso de la adecuada podría traer problemas de brillos, sombras o contrastes, haciendo que el algoritmo no tenga un buen funcionamiento o simplemente no funcione. Una imagen correcta para el procesado es aquella que permite diferenciar los píxeles del objeto de interés de los que no, además no deben existir lugares saturados, o sombras que oculten el lugar de interés [16].

Técnica	Descripción	Pros	Contras
Direccional	La iluminación de la escena se realiza con uno o varios puntos de luz en la que el ángulo de incidencia no es paralelo ni perpendicular al eje de la cámara.	Flexible, adaptable y barata	Produce brillos y genera sombras.
Lateral o darkfield	Se utiliza luz direccional en la que el ángulo de incidencia es paralelo a la superficie a inspeccionar y perpendicular al eje de la cámara. Se puede utilizar en objetos opacos y transparentes.	Resalta la textura de la superficie del objeto. Descubre grietas, burbujas incluso en el interior del objeto si es transparente.	Aparecen zonas quemadas y sombras. Poco contraste del borde.
Difusa	Iluminando la escena de forma indirecta se consigue una luz suave.	Reduce brillos y sombras. Iluminación suave.	Sistema de iluminación de gran tamaño, dificultad para encajar en pequeños espacios y las características de la superficie se difuminan.
Anillo	Luz direccional con ángulo de incidencia en la misma dirección que el eje de la cámara. La lente se coloca en el centro del anillo.	Reduce sombras, se puede conseguir una iluminación suave si se utiliza un filtro en el anillo de luz.	Reflejos con forma del anillo circular en la escena.
Difusa axial	Luz difusa alineada con la el eje óptico de la cámara. Se utiliza un cristal polarizado.	No existen sombras. Iluminación suave.	Poca intensidad.

Estructurada	Se proyecta un patrón en la escena tipo línea, matriz de puntos o círculos generados por láser.	Se obtiene la estructura del objeto.	No se distinguen colores.
Contraluz	El objeto a inspeccionar se sitúa entre la fuente de luz y la cámara.	Se obtiene una imagen del borde bien definida.	Elimina los detalles de la superficie.

Tabla 3.2: Técnicas de iluminación [16].

Aprendizaje automático

El aprendizaje automático o machine learning, surgió de la teoría de que las computadoras pueden aprender a partir de los datos. Es un campo de investigación donde se unen la estadística, inteligencia artificial y la informática. Hoy en día es muy común ver la aplicación de los algoritmos de aprendizaje, por ejemplo, en recomendaciones automáticas de que películas ver o que productos comprar, muchos sitios web y dispositivos electrónicos cuentan con estos algoritmos [22].

Además de las aplicaciones comerciales, el aprendizaje automático también tiene otras aplicaciones como en el campo de la investigación. Anteriormente, los sistemas inteligentes eran programados con reglas “si” y “si no”, y eran sistemas diseñados por expertos. Este tipo de sistemas pueden resultar útiles para algunas aplicaciones, específicamente para aquellas en las que un humano tiene clara la decisión que debe ser tomada, pero estos sistemas tienen algunas complicaciones como:

- Limita su uso al emplear lógica para una sola tarea.
- El diseño de las reglas requiere de un análisis profundo de como un experto tomaría la decisión.

Estas complicaciones se pueden notar por ejemplo en la detección de rostros, pues la manera en que una computadora percibe un rostro es diferente como la percibimos los humanos. Esto hace que para un humano sea imposible encontrar las reglas que le permita a la máquina identificar rostros en una imagen, pero esto es un problema

que se puede resolver utilizando aprendizaje automático y una base de datos con diferentes rostros [22].

Aprendizaje supervisado

El aprendizaje supervisado es uno de los más utilizados y exitosos. Consiste en predecir el resultado de datos nuevos no vistos antes, a partir de otro conjunto de datos, es decir, se pretende obtener un resultado determinado a partir de una entrada determinada. El aprendizaje supervisado requiere normalmente un poco de esfuerzo humano para etiquetar los datos con los cuales será entrenada la máquina, pero después de esto se automatiza el proceso [22].

Clasificación y regresión

Existen dos tipos de algoritmos principales de aprendizaje supervisado, se conocen como clasificación y regresión.

La clasificación consiste en predecir una etiqueta a partir de una lista de datos etiquetados previamente, pueden existir problemas de clasificación binaria, es decir, cuando solo se puede predecir la etiqueta entre dos opciones, o problemas de clasificación múltiple, que ocurre cuando hay más de dos clases o etiquetas como opciones para predecir [22].

La regresión consiste en la predicción de un número continuo o un número de punto flotante en términos de programación, en donde el resultado puede ser un número arbitrario.

Lo que se busca con el aprendizaje supervisado es construir un modelo a partir de un conjunto de entrenamiento, para luego realizar predicciones sobre datos nuevos, es decir, que no hayan sido usados antes en el algoritmo. Si el modelo es capaz de predecir correctamente estos nuevos datos, se puede afirmar que es capaz de generalizar desde el conjunto de entrenamiento al conjunto de prueba, ese es el objetivo, encontrar un modelo que generalice lo mejor posible.

En ocasiones ocurre que el modelo funciona muy bien con el conjunto de entrenamiento, pero no generaliza bien los datos, a este problema se le conoce como

“*overfitting*” o sobreajuste, y ocurre porque el diseño del modelo se centra demasiado en el conjunto de datos de entrenamiento. Una manera de evitar el sobreajuste es limitarse a construir modelos sencillos [22].

Se debe tener cuidado, pues el diseño de un modelo muy sencillo puede ocasionar lo que se conoce como “*underfitting*” o desajuste, debido a que no se capta la variación entre las etiquetas al no explicar lo suficiente el resultado [22].

Modelo de análisis de discriminante lineal

El modelo de análisis de discriminante lineal, es un modelo de clasificación supervisado, es decir, que los nuevos datos son clasificados entre una de diferentes clases, este modelo se basa en el teorema de Bayes. LDA estima la probabilidad de que una muestra, dado un determinado valor de sus características (predictores) pertenezca a cada una de las clases [23].

$$P(A|B) = \frac{P(AB)}{P(A)} \quad (3.11)$$

El teorema de Bayes considera dos eventos A y B y estima la probabilidad de que ocurra B habiendo ocurrido A, teniendo en cuenta la probabilidad de que A y B ocurran al mismo tiempo, dividida entre la probabilidad que ocurra A [23].

Matriz de confusión

La matriz de confusión es una herramienta que permite medir el comportamiento del modelo que se está utilizando, cuando se le ingresan datos nuevos [24].

	Positivos(predicción)	Negativos(predicción)
Positivo(observación)	VP	FN
Negativo(observación)	FP	VN

Tabla 3.3: Estructura de una matriz de confusión [24].

En donde:

- **VP** es la cantidad de datos positivos que fueron clasificados correctamente por el modelo.
- **VN** es la cantidad de datos negativos que fueron clasificados correctamente por el modelo.
- **FN** es la cantidad de datos positivos que fueron clasificados de manera errónea como negativos por el modelo.
- **FP** es la cantidad de datos negativos que fueron clasificados de manera errónea como positivos por el modelo.

Con basa en la matriz de confusión se pueden obtener los siguientes datos:

Exactitud (Accuracy): muestra que porcentaje de los datos clasifica correctamente.

$$\text{Exactitud} = \frac{VP + VN}{\text{Total}} \quad (3.12)$$

Tasa de error (Misclassification Rate): muestra que porcentaje de los datos clasifica incorrectamente.

$$\text{Tasa de error} = \frac{FP + FN}{\text{Total}} \quad (3.13)$$

Sensibilidad (Tasa de verdaderos positivos): en caso de que la clase sea positiva, muestra que porcentaje logra clasificar.

$$\text{Sensibilidad} = \frac{VP}{\text{Total positivos}} \quad (3.14)$$

Especificidad (Tasa de verdaderos negativos): en caso de que la clase sea negativa, muestra que porcentaje logra clasificar.

$$\text{Especificidad} = \frac{VN}{\text{Total negativos}} \quad (3.15)$$

Precisión: cuando predice positivos, muestra que porcentaje predice correctamente.

$$\text{Precisión} = \frac{VP}{\text{Total clasificados positivos}} \quad (3.16)$$

Valor de predicción negativos(VPN): cuando predice negativos, muestra que porcentaje predice correctamente.

$$\text{VPN} = \frac{VN}{\text{Total clasificados negativos}} \quad (3.17)$$

Capítulo 4

Metodología

En esta sección se da una explicación sobre los procesos que se llevaron a cabo para predecir el estado de nitrógeno de las hojas de un cultivo de gulupa, desde la toma de muestras hasta la implementación del algoritmo de aprendizaje, utilizando como lenguaje de programación Python y bibliotecas como OpenCV y Scikit-Learn.

4.1. Análisis

La herramienta desarrollada está basada en las imágenes capturadas por la cámara multiespectral Parrot Sequoia, las procesa y extrae las características que serán usadas para el entrenamiento de la máquina. La herramienta permite conocer al usuario si la hoja de gulupa presenta o no ausencia de nitrógeno. A continuación se presentan los elementos necesarios para llevar a cabo dicha herramienta.

4.1.1. Sensor Parrot Sequoia

Sequoia es un sensor que fue diseñado para la agricultura. Su diseño se basa en una excelente precisión, tamaño y peso reducidos al máximo y gran facilidad de uso. Permite obtener imágenes de parcelas agrícolas en diferentes bandas espectrales que miden el estado de la vegetación, estas bandas son: verde (longitud de onda de

550 nm), rojo(longitud de onda de 660 nm), Red Edge (longitud de onta de 735 nm) e infrarrojocercano (longitud de onda de 790 nm) [25].



Figura 4.1: Cuerpo de la cámara. Tomado:
[26]

- Posee 4 cámaras espectrales de 1.2 Mpx que recogen los datos de las bandas espectrales verde, roja, Red-Edge e infrarrojo cercano [25].
- Cámara RGB de 16 Mpx.
- Un indicador luminoso que sirve como referencia para las tomas de fotografías y calibración.
- Activador del modo ráfaga, Wi-Fi y tomar una foto
- Puerto micro USB host en donde se conecta el sunshine.
- Puerto micro USB device para conectar el banco de poder.



Figura 4.2: Sensor de luz ambiental. Tomado:
[26]

El sunshine sensor permite calibrar las imágenes en función de la incidencia solar. Gracias a esto es posible comparar imágenes en el tiempo, pese a las variaciones de luz durante la toma. El sunshine sensor cuenta con alojamiento para una tarjeta SD [25].

Imágenes capturadas por la cámara:

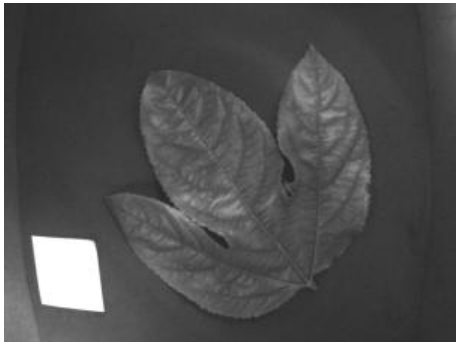


Figura 4.3: Imagen en la banda del verde



Figura 4.4: Imagen en la banda del rojo.



Figura 4.5: Imagen en la banda del borde rojo.

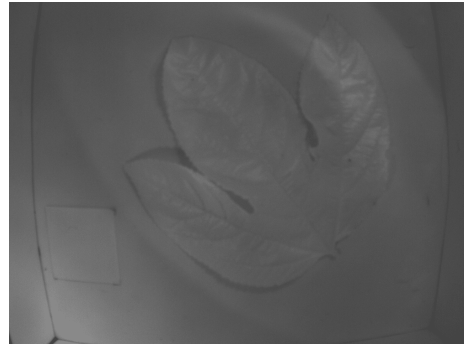


Figura 4.6: Imagen en la banda del infrarrojo.



Figura 4.7: Imagen RGB.

4.1.2. Lenguaje de programación en Python

Python combina el poder de los lenguajes de programación de propósito general con la facilidad de uso de lenguajes específicos de dominio como MATLAB o R. Python tiene bibliotecas para la carga de datos, visualización, estadística, procesamiento de lenguaje natural, procesamiento de imágenes y más. Esta amplia caja de herramientas proporciona a los científicos de datos una gran variedad de funciones de propósito general y especial. Una de las principales ventajas de usar Python es la capacidad de interactuar directamente con el código, usando un terminal. El aprendizaje automático y el análisis de datos son procesos fundamentalmente iterativos, en los que los datos impulsan el análisis. Es esencial que estos procesos tengan herramientas que permitan una iteración rápida y una interacción fácil.

4.1.3. Biblioteca OpenCV

OpenCV significa Open Computer Vision (Vision Artificial Abierta). Es una de las bibliotecas más usadas para la visión artificial. Algunas de sus aplicaciones son la detección de movimiento, reconocimientos de objetos y reconstrucción 3D a partir de imágenes. La popularidad de OpenCV se debe a que permite ser usada de manera libre para propósitos comerciales y de investigación, ofrece soporte para diferentes sistemas operativos y varias arquitecturas de hardware, además es documentada y explicada [27].

4.1.4. Biblioteca Scikit-Learn

Es una biblioteca para aprendizaje automático [28], además es gratis para python, contiene algoritmos de clasificación, regresión, clustering y reducción de dimensionalidad, es compatible con otras librerías como NumPy, SciPy y matplotlib. Los algoritmos de Scikit-Learn se combinan y depuran con otras estructuras de datos como Pandas [29].

4.1.5. Software SolidWorks

Para el desarrollo del diseño de la caja se utilizó el software SOLIDWORKS, el cual es un software de diseño CAD 3D, es decir que es asistido por computadora, es utilizado para modelar y ensamblar piezas 3D y planos 2D. Ofrece soluciones intuitivas además de contar con una amplia gama de herramientas complejas que permiten mejorar el proceso del diseño de una pieza [30].

4.2. Metodología

La metodología planteada para llevar a cabo el proyecto fue basada en artículos consultados, principalmente en [8], donde realizaron el diseño de una caja oscura para la captura de las imágenes y además utilizaron las características de índices de vegetación, promedio y varianza de la textura de las imágenes multiespectrales.

A continuación se muestra el diagrama de flujo que evidencia los procesos que se llevaron a cabo en el presente proyecto.

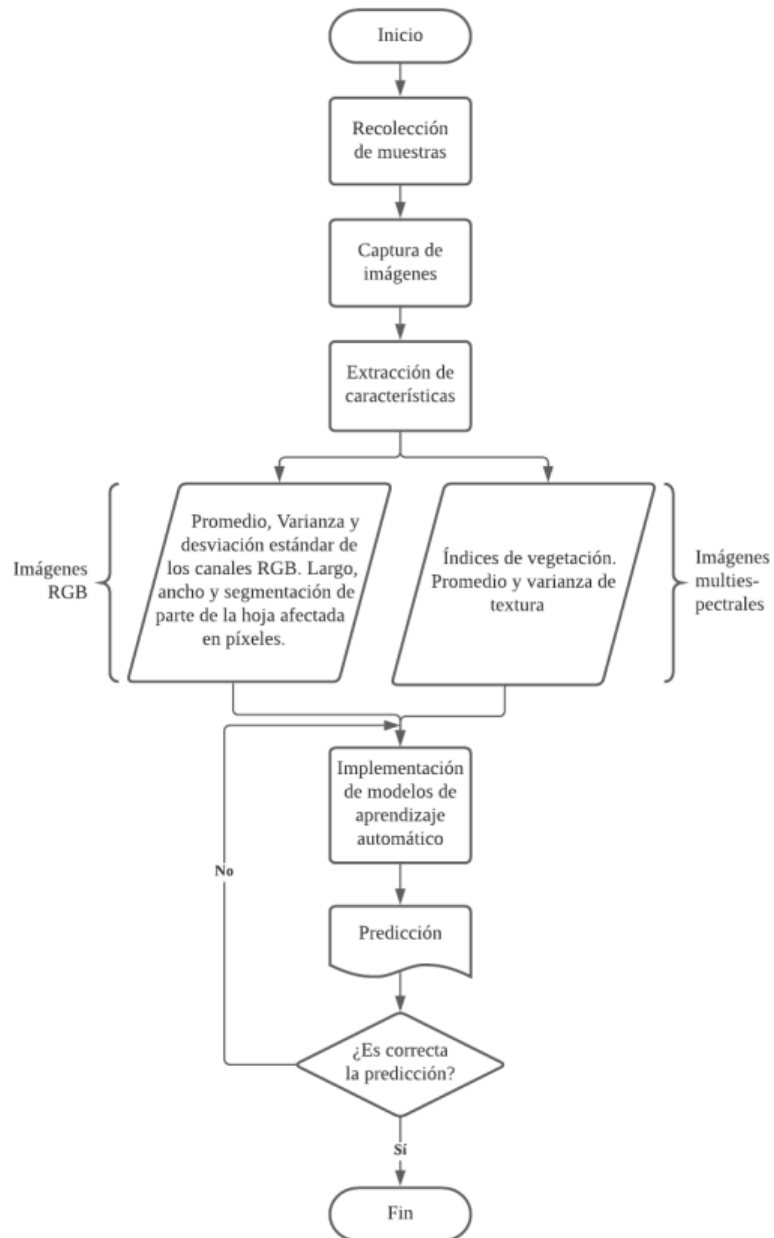


Figura 4.8: Diagrama de flujo de la metodología.

4.2.1. Recolección de muestras

La recolección de muestras se llevó a cabo en la finca La Pradera, ubicada en la vereda Lázaro Fonte en el municipio de Pasca Cundinamarca. El cultivo está conformado por 1250 plantas de gulupa en una hectárea y se encuentra a una altitud aproximada de 2310 msnm, la distancia que hay entre las plantas es de 4 m y entre cada surco hay una distancia de 2 m. Además contaba con un semitecho que cubría cada surco, esto se realiza principalmente para evitar bacterias en los cultivos debido a los cambios drásticos de clima.

Según indicaciones de la ingeniera agrónoma que realizaba el acompañamiento en las visitas al cultivo, este presentaba algunas enfermedades y plagas como mancha de aceite, roña, trips, entre otras.



Figura 4.9: Semitecho del cultivo.



Figura 4.10: Interior del cultivo con el semitecho.

Para las visitas al cultivo se tuvo acompañamiento de una ingeniera agrónoma quien nos guiaba acerca de los síntomas que puede presentar la planta cuando se presenta ausencia de nitrógeno, además de esto, ayudaba con la identificación de las hojas que serían utilizadas como muestras. Durante la visita fueron tomadas 2 tipos de muestras, las muestras foliares (hojas) y las muestras de suelo. Para llevar a cabo la toma de las mismas se escogieron 2 surcos al azar, teniendo en cuenta que hubiera una distancia considerable entre cada uno.

Toma de muestras foliares: Las muestras foliares fueron tomadas en diferentes partes de cada surco seleccionado, después de recolectar varias de ellas se procedió a clasificar cada hoja en uno de los siguientes grupos:

- Hojas grandes verdes.
- Hojas grandes amarillas.
- Hojas pequeñas verdes.
- Hojas pequeñas amarillas.



Figura 4.11: Recolección de hojas para las muestras foliares.

Al clasificar cada hoja de esta manera se obtuvieron 4 muestras foliares diferentes para cada surco. Este proceso se hizo de esta manera debido a que durante las pruebas químicas no es posible obtener un resultado de nitrógeno de cada hoja por individual, a causa de que es necesario un peso mínimo para que se puedan realizar las pruebas químicas. Por este motivo se escogieron grupos de hojas con características similares, obteniendo así 4 muestras por cada surco, es decir, un total de 8 muestras foliares de un peso medio de 60g cada una.

Después de separar los grupos de hojas ya mencionados, se almacenaron en bolsas de papel, que por el material que están hechos permite que la hoja se conserve más tiempo y se evita la descomposición de la misma, mientras las muestras son llevadas al laboratorio.



Figura 4.12: Muestras colocadas en bolsa de papel.

Toma de muestras de suelo: La toma de muestras de suelo se tomaron en cada surco donde fueron tomadas las muestras foliares. Para la toma de muestras de suelo se hizo uso de los siguientes elementos:

- Un balde
- Un palín
- Guantes
- Agua filtrada
- Bolsas Zip Lock

En primera instancia se tuvo en cuenta que los elementos utilizados debían estar completamente limpios, sin residuos de tierra u otros materiales que puedan alterar la muestra, por lo cual fueron limpiados con agua filtrada.

Para la toma de estas muestras se debía cavar el suelo a una profundidad de 60 cm en forma de “V” con ayuda del palín.



Figura 4.13: Removido de tierra.

Al finalizar se procedía a raspar una lámina de tierra por el borde de cada lado de la “V”, obteniendo así partes de tierra de cada capa del suelo.



Figura 4.14: Hoyo en forma de “V” con 60cm de profundidad.

La tierra removida de los bordes del hoyo después de cavar a 60cm de profundidad, fue colocada en el balde para ser mezclada con ayuda de los guantes, retirar posibles rocas y deshacer grumos para luego ser empacada en la bolsa Zip Lock, teniendo en cuenta que se debía tener un peso aproximado de 500gr de tierra.



Figura 4.15: Mezclado de tierra.



Figura 4.16: Muestra de suelo empacada.

Al finalizar se obtuvo un total de 10 muestras, las cuales se nombraron de la siguiente manera:

SURCO 1		
Nombre	Tipo	Grupo
S1HGV	Foliar	Hoja grande verde
S1HGA	Foliar	Hoja grande amarilla
S1HPV	Foliar	Hoja pequeña verde
S1HPA	Foliar	Hoja pequeña amarilla
TierraS1	Suelo	Tierra

Tabla 4.1: Muestras del surco 1.

SURCO 3		
Nombre	Tipo	Grupo
S3HGV	Foliar	Hoja grande verde
S3HGA	Foliar	Hoja grande amarilla
S3HPV	Foliar	Hoja pequeña verde
S3HPA	Foliar	Hoja pequeña amarilla
TierraS3	Suelo	Tierra

Tabla 4.2: Muestras del surco 3.

Ya con las muestras listas se procedió a la captura de imágenes de las muestras foliares, al finalizar se llevaron las muestras al laboratorio Asinal para que se le realizaran las pruebas químicas con el método Kjeldahl.

4.2.2. Captura de imágenes

Diseño de caja oscura

La captura de imágenes se realizó con ayuda de una caja oscura, la finalidad era no tener alteraciones por el ambiente, es decir, lluvia o nubosidad. Teniendo de esta manera un control de luminosidad.

La caja fue diseñada en el software SolidWorks, para lo cual se realizó el diseño por separado de las partes que llevaría la caja, incluyendo: caja, tapa con soportes, cámara, banco de poder y lámparas halógenas, de esta manera se realizó el ensamble al estar terminadas el total de las piezas. En el diseño se tuvo en cuenta las medidas exactas de cada elemento, las cuales se muestran a continuación:

Elemento	Características
Caja	300 x 300 x 300 mm
Tapa de la caja	300 x 300 mm
Lámina de la parte interior de la tapa de la caja	280 x 280 mm
Cámara	59 x 41 x 29,5 mm
Banco de poder	160 x 80 x 22 mm

Tabla 4.3: Medidas de los elementos que conforman la caja negra

Esta caja fue fabricada en material MDF de 0.9 mm, sus dimensiones fueron 300 x 300 x 300 mm, la tapa contaba con una lámina extra de dimensión 280 x 280 mm adherida en su interior, la finalidad de esto era tener mayor cubrimiento evitando al máximo la incidencia de luz externa dentro de la caja, además, la tapa contenía dos soportes los cuales fueron utilizados para sostener la cámara y el banco de poder.

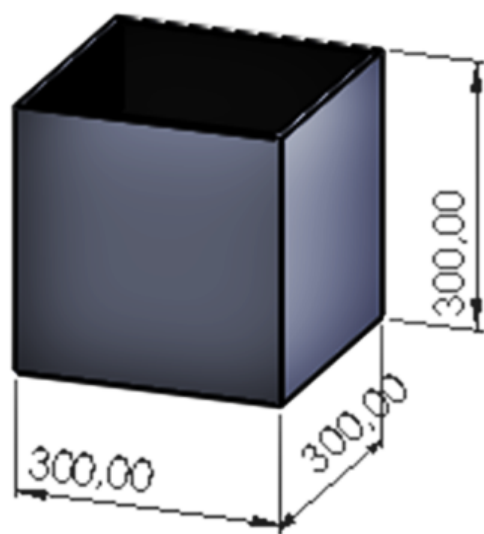


Figura 4.17: Medidas del cubo de la caja.

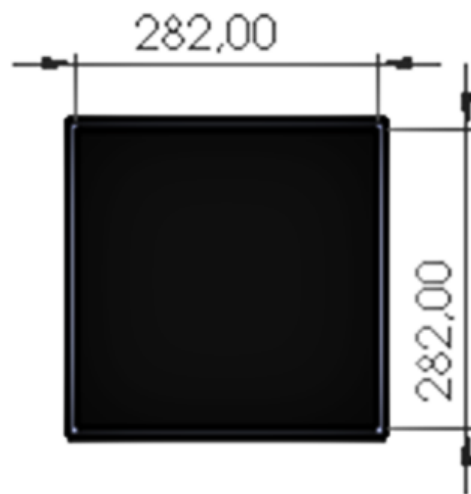


Figura 4.18: Medidas lámina interior de la tapa de la caja.

En la figura 4.20 se observa la tapa con la lámina adicional y los soportes de la cámara y el banco de poder, a continuación se muestran los planos respectivos a la tapa y sus soportes:

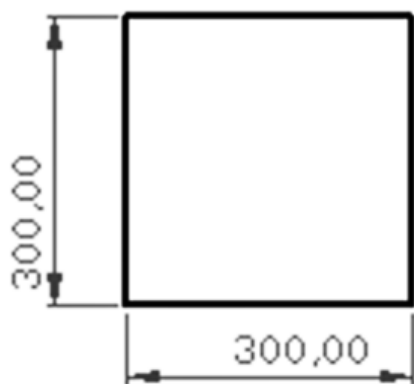


Figura 4.19: Plano vista superior de la tapa.

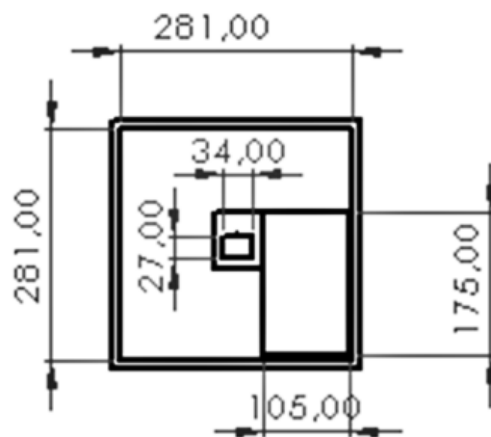


Figura 4.20: Plano vista inferior de la tapa.

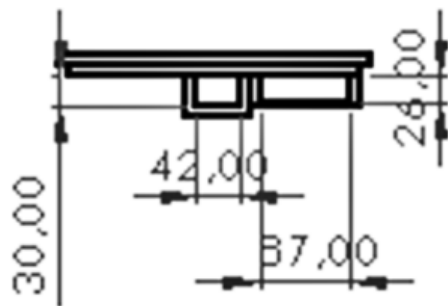


Figura 4.21: Plano vista frontal de la tapa.

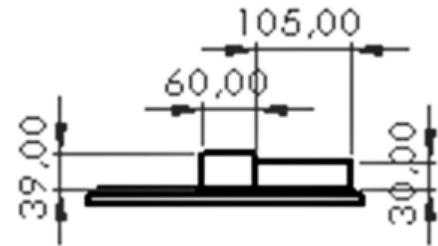


Figura 4.22: Plano vista posterior de la tapa.

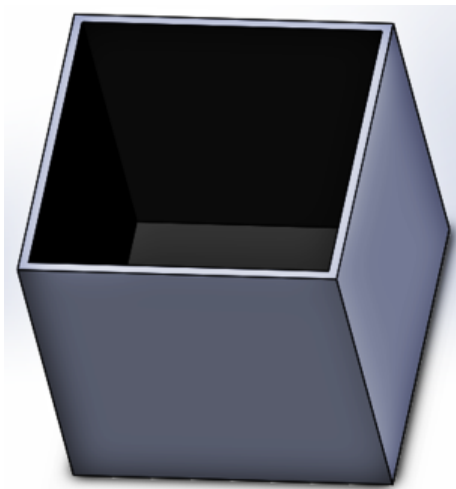


Figura 4.23: Diseño del cubo

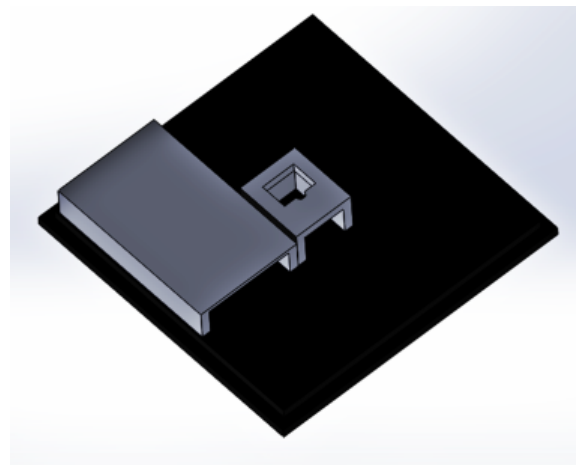


Figura 4.24: Tapa de la caja con los soportes.

Además de esto la caja contaba con un sistema de iluminación difusa que constaba de dos lámparas halógenas dicroicas (12V 10W) con conector Bi-pin, alimentadas con una fuente de poder, el hecho de que las lámparas sean dicroicas implica una reducción de la carga térmica sobre el objeto iluminado, haciendo que el calor se vaya hacia atrás. Las lámparas fueron ubicadas en esquinas opuestas dentro de la parte superior de la caja, dando de esta manera la iluminación adecuada para obtener las imágenes de las hojas. Las lámparas se ubicaron de tal manera que no coincidieran directamente en la hoja, pues esto causaba brillo en las imágenes



Figura 4.25: Lámpara halógena dicróica (12V 10W) con conector Bi-pin.

En el interior de las paredes de la caja y de la tapa se colocó papel aluminio, de esta manera la iluminación tenía mayor distribución en la caja. En el fondo se colocó una lámina de etilvinilacetato (foamy) fucsia, de tal manera que en las imágenes RGB existiera un contraste entre la hoja y el fondo.



Figura 4.26: Diseño físico de la caja oscura.



Figura 4.27: Diseño físico del interior de la caja oscura.

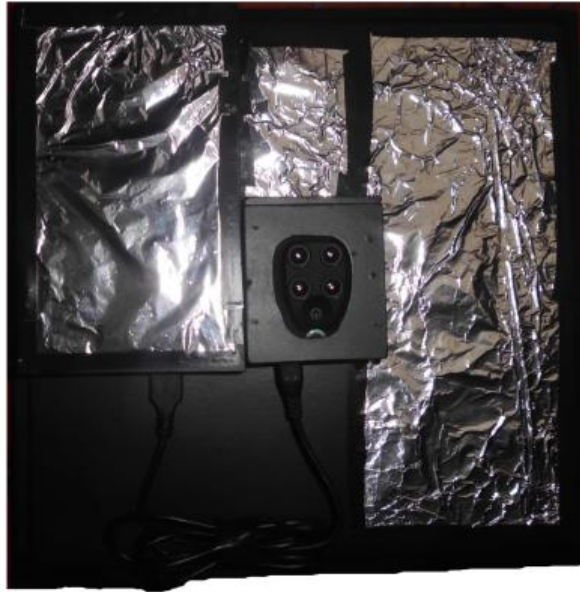


Figura 4.28: Diseño físico del interior de la tapa.

La caja también contaba con un cuadro de referencia de papel de 50x50mm que después de realizar varias pruebas se llegó a la conclusión de que el color más adecuado debido al umbral de segmentación era el azul claro. El cual se ubicó cerca a una esquina sobre la lámina de etilvinilacetato.



Figura 4.29: Toma de imágenes.

A continuación se muestra el diseño realizado en SOLIDWORKS de la cámara parrot sequoia y el banco de poder:



Figura 4.30: Diseño de la cámara en SolidWorks.

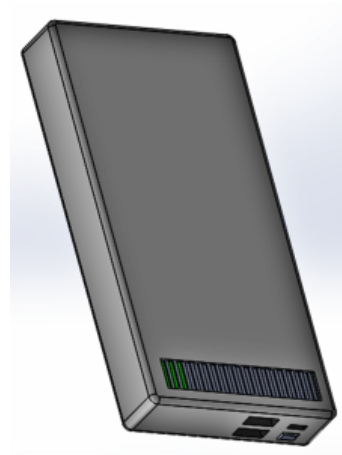


Figura 4.31: Diseño del banco de poder en SolidWorks.

El precio de la fabricación de la caja se muestra a continuación:

Elemento	Valor
Caja con tapa	\$ 40.000
Soportes para la cámara y banco de poder	\$ 10.000
Lámparas halógenas	\$ 7.000
Soportes para lámparas halógenas	\$ 5.000
TOTAL	\$ 62.000

Tabla 4.4: Costos totales de la caja oscura.

El procedimiento para la captura de imágenes fue ubicar la hoja de gulepa en el fondo de la caja, después se colocaba la tapa con la cámara y el banco de poder ubicados previamente, se se accedía a la interfaz de la cámara con su IP desde el celular, se encendían las lámparas y se capturaba la imagen.

Las imágenes fueron tomadas a cada muestra que sería llevada al laboratorio, por lo cual se tuvo en cuenta las carpetas donde se guardaban las imágenes en la cámara de

cada muestra. Al finalizar la captura, las imágenes se almacenaron en el computador y luego las muestras fueron llevadas al laboratorio para realizar las pruebas químicas.



Figura 4.32: Entrega de muestras en el laboratorio químico.

4.2.3. Extracción de características

Para hacer uso de algoritmos de aprendizaje automático es necesario realizar extracción de características de las imágenes capturadas, dichas características serán usadas para el entrenamiento de la máquina.

Extracción de características de las imágenes RGB

Para la extracción de características de las imágenes RGB se hizo uso del código realizado por el ingeniero electrónico Santiago Alejandro Trujillo Fandiño en su tesis “Implementación de un sistema capaz de facilitar la identificación del cambio de color en la hoja de gulupa conforme a la presencia o ausencia de nitrógeno mediante el procesamiento de imágenes RGB” [1]. Al cual se le realizó la modificación de lograr procesar varias imágenes guardadas en una carpeta y que además los datos de cada imagen fueran guardados directamente en un archivo csv. También se agruparon las funciones principales del código principal en una biblioteca llamada *imutils* y se agregaron nuevas funciones que fueron utilizadas en la extracción de características

de las imágenes multiespectrales.

```
1 def prom_indices (indice_veg, fil, col):
2     i2=0
3     contador=0
4
5     for i in range(fil):
6         for j in range (col):
7             if indice_veg[i,j] < 0 and indice_veg[i,j]>=-1 or
indice_veg[i,j]>0 and indice_veg[i,j]<=1:
8                 contador=contador+1
9                 i2=indice_veg[i,j]+i2
10
11     prom=i2/contador
12     return prom
```

Algoritmo 1: Función creada en la biblioteca *imutils* para calcular el promedio de los índices de vegetación.

El algoritmo 1 muestra la función creada en la biblioteca *imutils* para calcular el promedio de los índices de vegetación que se verá más adelante.

```
1 def falso_color (indice):
2     if indice.min()<0:
3         indice = indice + (indice.min()* -1)
4
5     indice = (indice * 255)/indice.max()
6     indice = indice.round()
7
8     indice_image1 = np.array(indice, dtype=np.uint8)
9     im_color2 = cv2.applyColorMap(indice_image1, cv2.
COLORMAP_RAINBOW)
10     return im_color2
```

Algoritmo 2: Función creada en la biblioteca *imutils* para generar falso color a partir de los índices de vegetación.

La función *falso_color*, genera una imagen en falso color a partir de los índices de

vegetación calculados.

```
1 def featurespixmap (lista,archivo):
2     with open (archivo,'wb') as file:
3         writer = csv.writer(file)
4     writer.writerows(lista)
```

Algoritmo 3: Función creada en la biblioteca *imutils* para guardar datos en un documento con extensión csv.

La función del algoritmo 3 fue creada inicialmente para guardar los datos generados por la extracción de características de las imágenes RGB.

```
1 parser = argparse.ArgumentParser()
2 parser.add_argument('-d', '--direccion',help='Dirección de la
    carpeta donde se encuentran las imágenes')
3 parser.add_argument('-a', '--archivo', help='Nombre del archivo CSV
    ')
4 args = parser.parse_args()
5
6 archivo = "{archivo}".format(archivo=args.archivo)
7 imagesPath = "{direccion}".format(direccion=args.direccion)
8 imagesPathList = os.listdir(imagesPath)
9 print('imagesPathList',imagesPathList)
```

Algoritmo 4: Código para indicar la ubicación de la carpeta con las imágenes.

El código del algoritmo 4 fue colocado al inicio del código principal creado por el ingeniero Santiago Trujillo [1], esto fue para indicar en la línea de órdenes la dirección de la ubicación de la carpeta que contenía las imágenes RGB y también se ingresaba el nombre del archivo csv donde se guardarían los datos extraídos.

```
1 zaira@debian:~/Documentos$ python cargarimg2.py -d /home/zaira/
    Documentos/rgbS3HPV -a S3HPVcm.csv
```

Algoritmo 5: Ejecución desde la línea de órdenes del código principal de la extracción de características RGB, indicando la ubicación de la carpeta con imágenes y el nombre del archivo csv.

En este caso la dirección donde se encontraba la carpeta que contenía las imágenes RGB era en la dirección */home/zaira/Documentos/mejora/mejora2/rgbS3HPV* y el nombre que se le asignó al archivo csv fue *S3HPVcm.csv*.

Al realizar la extracción de características de las imágenes RGB, se obtuvo como resultado las tablas de los surcos con las características de los datos estadísticos de media, varianza y desviación estándar de cada canal de la imagen (RGB). Además de los datos del largo, ancho y de la parte de la hoja que presenta cambio de color, estos últimos tres datos en píxeles.

Extracción de características de las imágenes imágenes multiespectrales

La cámara Parrot Sequoia además de capturar una imagen RGB también captura cuatro imágenes multiespectrales, la del rojo, infrarrojo cercano, verde y borde rojo, las cuales fueron procesadas para extraer las características de los índices de vegetación NDVI, GNDVI y OSAVI, además se extrajeron las características de promedio y varianza de la textura de cada una de los cuatro canales de las imágenes multiespectrales.

Cálculo de índices de vegetación

Para ejecutar el programa se colocaba la palabra *“python”* seguida del nombre del archivo que contiene el código, además del nombre del archivo csv donde se guardarán los datos. En este caso el nombre del archivo con el código se llama *“grises.py”* y el archivo csv *“featurespix.csv”*.

```
1
2 zaira@debian:~/Documentos/alinear_final$ python grises.py -a
  featurespix.csv
```

Algoritmo 6: Ejecución del código desde la línea de comandos.

Las bibliotecas utilizadas fueron:

```
1 import cv2
2 import imutils
3 import numpy as np
4 import matplotlib
5 import os
6 import argparse
```

Algoritmo 7: Bibliotecas utilizadas para la extracción de características de imágenes multiespectrales.

Es importante tener en cuenta que la biblioteca *imutils* y las cuatro imágenes multiespectrales deben estar en la misma carpeta donde se encuentra el archivo con el código. Después se cargan las imágenes.

```
1 parser = argparse.ArgumentParser()
2 parser.add_argument('-a', '--archivo', help='Nombre del archivo CSV
  ')
3 args = parser.parse_args()
4 archivo = "{archivo}".format(archivo=args.archivo)
5
6 datos_indices= []
7 lista= []
8
9 path1 = "S1HGA_unidas"
10 path2 = "S1HGA_falsocolor"
11 path3 = "S1HGA_mascara"
12 path4 = "S1HGA_imaseg"
13 path5 = "S1HGA_NIR1"
14 path6 = "S1HGA_RED1"
15 path7 = "S1HGA_GRE1"
16 path8 = "S1HGA_REG1"
17 path9 = "S1HGA_NDVI"
18 path10 = "S1HGA_GNDVI"
19 path11 = "S1HGA_OSAVI"
```

Algoritmo 8: Lectura del nombre del archivo csv y declaración de nombre de carpetas donde se guardarán las imágenes.

Las cuatro primeras líneas del algoritmo 8 son para leer el nombre del archivo csv donde se guardarán los datos, además se crean dos listas con los nombres *datos_indices* y *lista*, las cuales irán almacenando los datos. Las variables de nombre *path* fueron creadas con los nombres de carpetas en donde se irán guardando cada

imagen durante el proceso.

```
1 image = cv2.imread('IMG_700101_001056_0000_NIR.TIF',0)
2 image2= cv2.imread('IMG_700101_001056_0000_RED.TIF',0)
3 image3= cv2.imread('IMG_700101_001056_0000_GRE.TIF',0)
4 image4= cv2.imread('IMG_700101_001056_0000_REG.TIF',0)
5
6 imagen1=cv2.resize(image,(width,height), interpolation=cv2.
    INTER_LINEAR)
7 imagen2=cv2.resize(image2,(width,height), interpolation=cv2.
    INTER_LINEAR)
8 imagen3=cv2.resize(image3,(width,height), interpolation=cv2.
    INTER_LINEAR)
9 imagen4=cv2.resize(image4,(width,height), interpolation=cv2.
    INTER_LINEAR)
```

Algoritmo 9: Cargar imágenes.

En el algoritmo 9 se cargan las imágenes multiespectrales, colocando en cada variable el nombre de cada imagen. También se redujo el tamaño de las imágenes para que el procesamiento fuera más rápido.

Para hallar los índices de vegetación se deben hacer operaciones de la reflectancia de cada píxel de una imagen de un canal con respecto a la de otro canal, por lo cual, es necesario realizar la alineación de las imágenes, debido a que por ubicación de los lentes en la cámara hace que las imágenes queden ligeramente desalineadas. Lo que se desea es que los puntos de interés de las imágenes coincidan.

```
1 M = np.float32([[1,0,0],[0,1,0]])
2 M2 = np.float32([[1,0,57],[0,1,6]])
3 M3 = np.float32([[1,0,45],[0,1,65]])
4 M4 = np.float32([[1,0,-15],[0,1,50]])
5
6 imageOut=cv2.warpAffine(imagen1,M,(width,height))
7 imageOut2=cv2.warpAffine(imagen2,M2,(width,height))
8 imageOut3=cv2.warpAffine(imagen3,M3,(width,height))
9 imageOut4=cv2.warpAffine(imagen4,M4,(width,height))
10 unidas =cv2.merge((imageOut,imageOut2,imageOut3,imageOut4))
```

Algoritmo 10: Alineación de las imágenes.

Para la alineación de las imágenes se creó una matriz de desplazamiento de 2×3 para cada imagen multiespectral, cada matriz contaba con dos componentes de desplazamiento, x que se encontraba en la primera fila tercera columna y el componente y que estaba en la segunda fila tercera columna, estos valores fueron los modificados para realizar el desplazamiento de cada imagen. La imagen del infrarrojo cercano fue la tomada como referencia y con base a esa se desplazaron las imágenes de los otros tres canales. Por tal razón se puede ver que en la matriz de desplazamiento M que es la del infrarrojo cercano, no se le han alterado sus valores para que esta no presente un desplazamiento.

Las matrices creadas fueron M , $M2$, $M3$ Y $M4$, con la función de OpenCV *warpAffine* se realiza el desplazamiento, a esta función se le deben entregar tres parámetros, la imagen, la matriz de desplazamiento (creada previamente), el ancho y alto de la imagen, que en este caso se le asignó el tamaño al que se había redimensionado en el algoritmo 9.

La función *merge* realiza la superposición de las imágenes, y de esta manera poder visualizar la alineación finalizada de las cuatro imágenes multiespectrales.

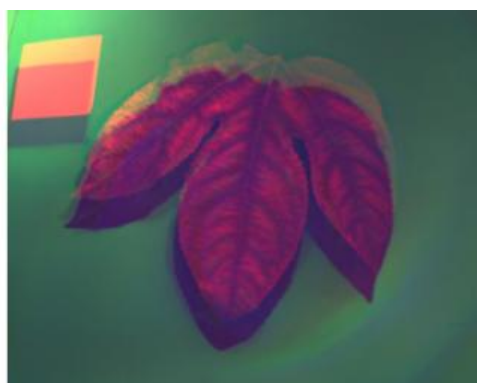


Figura 4.33: Imagen desalineada.

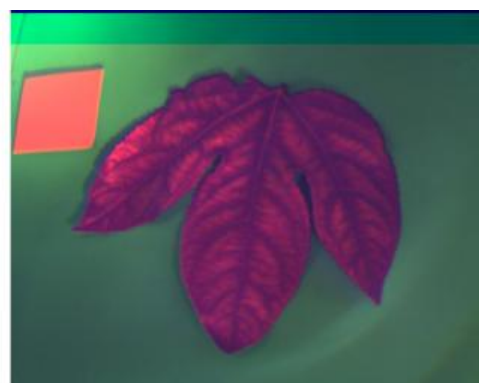


Figura 4.34: Imagen alineada.

Para encontrar los índices de vegetación se debe asegurar de segmentar la hoja y de esta manera encontrar el valor del índice correspondiente solo de la parte segmentada. Las imágenes multiespectrales presentan dificultades para ser segmentadas de la misma manera que se hace con una imagen RGB, pues en su mayoría se encuentra en un tono gris que dificulta la separación del objeto de interés del fondo, por lo cual se debe buscar la manera de poder diferenciar la hoja del resto de la imagen.

El NDVI lo que hace es reflejar la parte más vigorosa de las plantas, es decir la actividad fotosintética característica de las plantas, lo cual es una herramienta muy importante para la segmentación de la hoja, pues el NDVI daría valores muy diferentes de la hoja y el fondo, facilitando la segmentación.

```

1 NIR = np.array(imageOut , dtype=float)
2 RED = np.array(imageOut2 , dtype=float)
3
4 check = np.logical_and(RED>1, NIR>1)
5
6 NDVI = np.where(check , (NIR - RED)/(NIR + RED) , 0)
7 NDVI_index = NDVI

```

Algoritmo 11: Cálculo del NDVI para la segmentación.

Inicialmente se convierten la capa del infrarrojo cercano y la del rojo en arreglos para poder operar matemáticamente las dos matrices con la biblioteca de “numpy”, la operación para hallar el NDVI se muestra en la ecuación 3.4. Este procedimiento se realiza en la línea 6, la función *np.where* realiza la operación que se le indica y el resultado lo va guardando en *NDVI*.

```

1 if NDVI.min() < 0:
2     NDVI = NDVI + (NDVI.min()* -1)
3
4 NDVI = (NDVI * 255)/NDVI.max()
5 NDVI = NDVI.round()
6
7 NDVI_image = np.array(NDVI , dtype=np.uint8)
8 im_color = cv2.applyColorMap(NDVI_image , cv2.COLORMAP_RAINBOW)

```

Algoritmo 12: Establecimiento de falso color basado en los valores del índice de vegetación.

Debido a que se desea tener una visualización de las imágenes multiespectrales de tal manera que se logre diferenciar claramente el fondo de la hoja, se realiza una imagen en falso color basada en los valores calculados en el *NDVI*.

Para esto primero se verifica que los valores del índice se encuentren sobre cero, para luego establecerles un valor de 0 a 255 dependiendo del valor *NDVI*. La función *cv2.applyColorMap* crea la imagen en falso color.

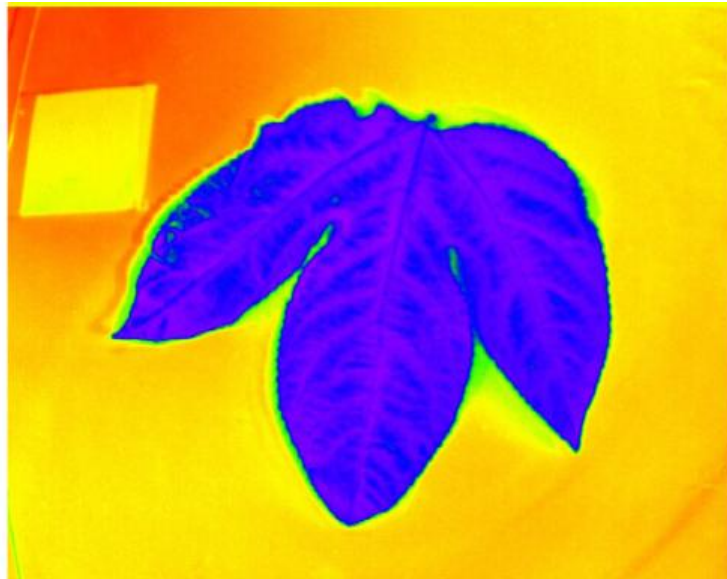


Figura 4.35: Imagen creada en falso color basada en los índices NDVI.

Como se observa en la figura 4.35, ya es posible diferenciar claramente la hoja del resto de la imagen, pues el color azul representa las partes con vida, que en este caso es la hoja, por el contrario, el fondo se muestra en un color amarillo y rojo, pues se trata de materiales sin vida o sintéticos.

Al realizar cálculo del NDVI, lo que se obtiene es una matriz con los índices de vegetación en cada píxel de la imagen, pero lo que se necesita es obtener un solo índice de vegetación para ser ubicado la gráfica del vector de características para el aprendizaje automático, para lo cual se decide realizar un promedio de los valores del NDVI, pero solo de la hoja, es decir, sin tomar los valores que representan el fondo, pues esto afectaría el resultado del promedio.

Ya teniendo la imagen en falso color se podía generar la máscara.

```
1 mascara = imutils.umbral(im_color,45,255)
2
3 fil , col, ch= im_color.shape
4
5 imaseg1=np.ones([fil,col],np.uint8)
6 for i in range(fil):
7     for j in range (col):
8         if mascara[i,j,0]>=240:
9             imaseg1[i,j]=255
10        else:
11            imaseg1[i,j]=0;
```

Algoritmo 13: Código para generar máscara.

En el algoritmo 13 se realiza la segmentación de la hoja en la imagen de falso color, se usa la librería *imutils* para utilizar la función de umbral y así encontrar la máscara para la segmentación.



Figura 4.36: Mascara basada en la imagen en falso color y segmentación.

Como se ve en la imagen 4.36 nuestro objeto de interés (la hoja), se encuentra de color azul, por lo cual se facilita la segmentación. Teniendo en cuenta que OpenCV por defecto toma las imágenes en el espacio BGR, se debe indicar al programa que al

recorrer la imagen reemplace por un valor de 255 (blanco), los píxeles en los cuales su componente en B (azul) sea mayor o igual a 240, esto proceso se muestra de la línea 6 a la 11 del algoritmo 13.



Figura 4.37: Segmentación de la hoja.

Se puede ver que la segmentación de la hoja aún presenta un poco de inconsistencia, es decir, aún posee algunos puntos negros dentro de la forma de la hoja y su contorno aún no está bien definido, para lo cual se implementan filtros para mejorar la segmentación.

```
1 erosionho = cv2.erode(imaseg1, kernel, iterations = 1)
2 openingho = cv2.morphologyEx(erosionho, cv2.MORPH_OPEN, kernel)
3
4 kernelclos= np.ones((3,3), np.uint8)
5 kernelcloss= np.ones((7,7), np.uint8)
6
7 imaseg= cv2.morphologyEx(openingho, cv2.MORPH_CLOSE, kernelclos)
8 imaseg = cv2.morphologyEx(openingho, cv2.MORPH_CLOSE, kernelcloss)
9 imaseg=cv2.medianBlur(imaseg, 13)
```

Algoritmo 14: Filtros para mejorar la segmentación.

La función `openingho` quita los puntos blancos que se encuentran fuera del espacio segmentado, después en la línea 6 y 7 se quitan los puntos negros que se encuentran dentro de la parte segmentada. En la línea 9 `medianBlur` realiza el suavizado del contorno de la segmentación.

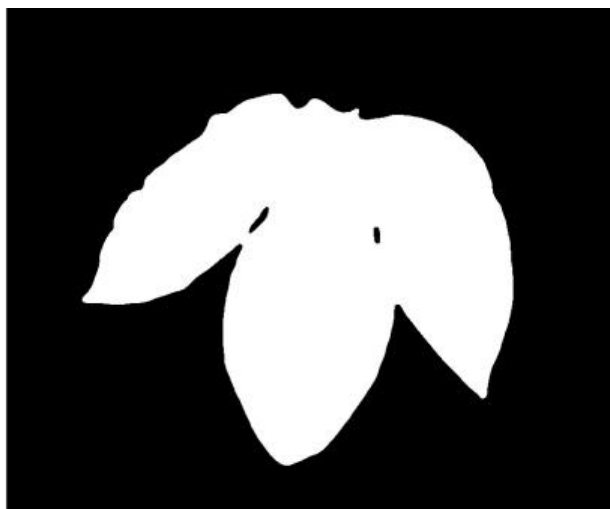


Figura 4.38: Segmentación mejorada de la hoja.

Como se ve en la imagen 4.38 la segmentación de la hoja mejoró, en cuanto a que desaparecieron los puntos negros dentro de la segmentación y el contorno quedó más definido.

Con la imagen ya segmentada se puede lograr quitar el fondo de las imágenes multiespectrales dejando solo la parte de la hoja.

```
1 NIR1=cv2.bitwise_and(imageOut ,imageOut ,mask=imaseg)
2 RED1=cv2.bitwise_and(imageOut2 ,imageOut2 ,mask=imaseg)
3 GRE1=cv2.bitwise_and(imageOut3 ,imageOut3 ,mask=imaseg)
4 REG1=cv2.bitwise_and(imageOut4 ,imageOut4 ,mask=imaseg)
```

Algoritmo 15: Operación de la imagen segmentada con las imágenes multiespectrales alineadas.

Con la función *bitwise_and* se opera cada imagen multiespectral alineada, pues en los píxeles blancos de la imagen segmentada, se mostrarán los píxeles que ocupen ese espacio en la imagen multiespectral, de esta manera el fondo de la imagen multiespectral desaparecería. Al realizar este proceso con las imágenes de los 4 canales se obtienen las imágenes sin fondo del infrarrojo cercano, verde, rojo y borde rojo.



Figura 4.39: Segmentación de imagen del canal verde.



Figura 4.40: Segmentación de imagen del canal del infrarrojo cercano.



Figura 4.41: Segmentación de imagen rojo.



Figura 4.42: Segmentación de imagen del canal borde rojo.

La finalidad de tener las imágenes como se muestra en 4.39, 4.40, 4.41 y 4.42 es poder realizar el cálculo de los índices de vegetación teniendo en cuenta solo los

píxeles que conforman la hoja, es decir, sin el fondo.

```
1 nir = np.array(NIR1, dtype=float)
2 red = np.array(RED1, dtype=float)
3 reg = np.array(REG1, dtype=float)
4 gre = np.array(GRE1, dtype=float)
5
6 check = np.logical_and(red>1, nir>1)
7 NDVI1 = np.where(check, (nir - red)/(nir + red), 0)
8 NDVI_index1 = NDVI1
9
10 check1 = np.logical_and(gre>1, nir>1)
11 GNDVI = np.where(check1, (nir-gre)/(nir+gre), 0)
12 GNDVI_index=GNDVI
13
14 OSAVI = np.where(check, ((nir-red)/(nir+red + 0.16)), 0)
15 OSAVI_index = OSAVI
```

Algoritmo 16: Cálculo de los índices de vegetación NDVI, GNDVI y OSAVI.

Las imágenes multiespectrales segmentadas se convierten en arreglos en las líneas 1 a la 4 en el algoritmo 16, para poder operarlas y encontrar las matrices con los índices de vegetación correspondientes al NDVI, GNDVI y OSAVI. Se utilizan las ecuaciones para calcular cada uno. Para los índices NDVI y OSAVI se utilizan los canales rojo e infrarrojo cercano, y para el GNDVI se utiliza los canales verde e infrarrojo cercano.

Los resultados como ya se ha mencionada son matrices con los índices de cada píxel de la imagen, con la diferencia de que ahora el resultado del fondo es 0, es decir, todos los píxeles del fondo están en cero.

Para realizar el promedio de cada índice se creó la función *prom_indices* en la librería

imutils que fue llamada desde el código principal.

```
1 prom_NDVI= imutils.prom_indices(NDVI_index1,fil,col )
2 prom_GNDVI = imutils.prom_indices(GNDVI_index,fil,col)
3 prom_OSAVI = imutils.prom_indices(OSAVI_index,fil,col)
```

Algoritmo 17: Llamado de la función *prom_indices* de la librería *imutils*.

```
1 def prom_indices (indice_veg, fil, col):
2     i2=0
3     contador=0
4
5     for i in range(fil):
6         for j in range (col):
7             if indice_veg[i,j] < 0 and indice_veg[i,j]>=-1 or
indice_veg[i,j]>0 and indice_veg[i,j]<=1:
8                 contador=contador+1
9                 i2=indice_veg[i,j]+i2
10
11     prom=i2/contador
12     return prom
```

Algoritmo 18: Función *prom_indices* de la librería *imutils*.

La función *prom_indices* recibe la matriz del índice de vegetación al que se le hallará el promedio y la cantidad de filas y columnas de la matriz. Las variables *i2* y *contador* servirán como auxiliares en el cálculo del promedio.

Se realizan dos *for* con el rango de las filas y las columnas para recorrer la matriz, el *if* de la línea 7 establece que el valor que se va a operar debe estar entre -1 y 1 sin contar el 0, esto es porque el fondo tiene valores de 0, de esta manera no se tienen en cuenta esos píxeles. Se declaró un contador para tener en cuenta la cantidad de valores que se están sumando. Así, *i2* es la sumatoria de todos los valores de los píxeles que conforman la hoja, y se realiza la división entre el contador obteniendo de esta manera el promedio del índice de vegetación de solo la hoja, y ese es el valor que retorna la función.

Los valores de los índices se van agregando al arreglo *datos*, para luego guardarlos

en el archivo csv.

```
1 datos_indices.append(prom_NDVI)
2 datos_indices.append(prom_GNDVI)
3 datos_indices.append(prom_OSAVI)
4
5 lista.append(datos_indices)
6 x=imutils.featurespixmap(GNDVI_index,archivo)
7 print ('NDVI es:' + str(float(prom_NDVI)))
8 print ('GNDVI es:' + str(float(prom_GNDVI)))
9 print ('OSAVI es:' + str(float(prom_OSAVI)))
```

Algoritmo 19: Código para imprimir índices en la línea de comandos y guardarlo en el archivo csv.

Las listas de *datos* se añade a la lista *lista*, para guardar los datos en el archivo csv. También se imprimen los valores en el terminal.

```
1 zaira@debian:~/Documentos/alinear_final$ python grises.py -a
  featurespixmap.csv
2 NDVI es:0.47326537317105005
3 GNDVI es:-0.08430012109011194
4 OSAVI es:0.47264435423995543
5
```

Algoritmo 20: Línea de comandos con los resultados de los índices.

Las matrices de los índices de las imágenes multiespectrales también se realizaron en falso color.

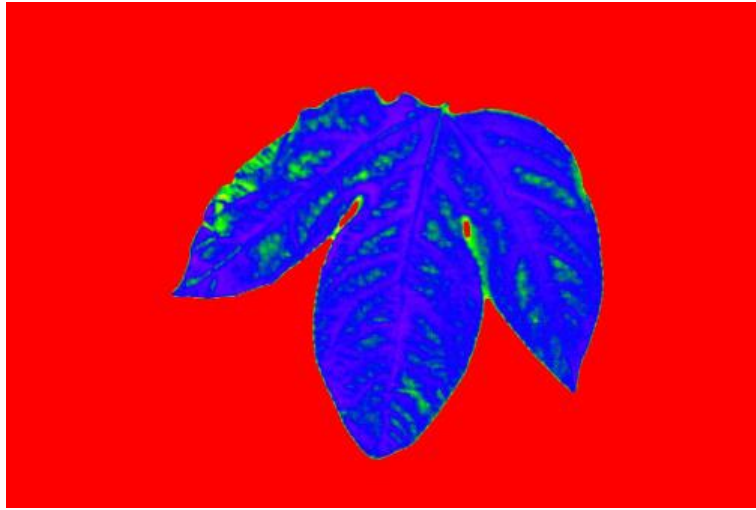


Figura 4.43: Imagen en falso color basado en el índice NDVI usando las imágenes multiespectrales segmentadas.

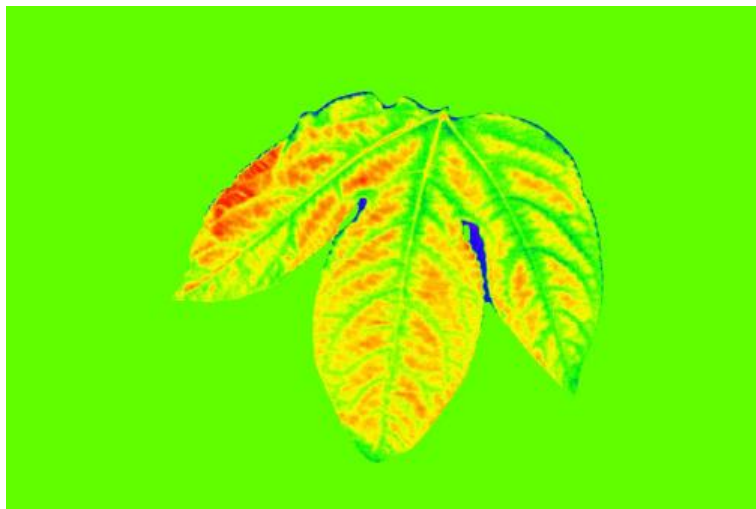


Figura 4.44: Imagen en falso color basado en el índice GNDVI usando las imágenes multiespectrales segmentadas.



Figura 4.45: Imagen en falso color basado en el índice OSAVI usando las imágenes multiespectrales segmentadas.

El resultado de los índices de vegetación de NDVI y OSAVI son valores muy parecidos.

```

1 cv2.imwrite(os.path.join(path1, "S1HGA_0426_unidas.jpg"), unidas)
2 cv2.imwrite(os.path.join(path2, "S1HGA_0426_falsocolor.jpg"),
  im_color)
3 cv2.imwrite(os.path.join(path3, "S1HGA_0426_mascara.jpg"), mascara)
4 cv2.imwrite(os.path.join(path4, "S1HGA_0426_imaseg.jpg"), imaseg)
5 cv2.imwrite(os.path.join(path5, "S1HGA_0426_NIR1.jpg"), NIR1)
6 cv2.imwrite(os.path.join(path6, "S1HGA_0426_RED1.jpg"), RED1)
7 cv2.imwrite(os.path.join(path7, "S1HGA_0426_GRE1.jpg"), GRE1)
8 cv2.imwrite(os.path.join(path8, "S1HGA_0426_REG1.jpg"), REG1)
9 cv2.imwrite(os.path.join(path9, "S1HGA_0426_NDVI.jpg"), im_colorNDVI)
10 cv2.imwrite(os.path.join(path10, "S1HGA_0426_GNDVI.jpg"),
  im_colorGNDVI)
11 cv2.imwrite(os.path.join(path11, "S1HGA_0426_OSAVI.jpg"),
  im_colorOSAVI)

```

Algoritmo 21: Código para guardar cada imagen en una carpeta.

Las tablas de los índices de vegetación de cada surco se muestran a continuación:

Surco 1

Nombre	NDVI	GNDVI	OSAVI
S1HGA_0415	0.473	-0.084	0.472
S1HGA_0416	0.475	-0.010	0.474
S1HGA_0417	0.482	-0.055	0.481
S1HGA_0419	0.487	-0.003	0.487
S1HGA_0421	0.502	-0.020	0.502
S1HGA_0422	0.467	-0.011	0.466
S1HGA_0423	0.515	0.014	0.514
S1HGA_0425	0.534	0.002	0.533
S1HGA_0426	0.465	0.008	0.464
S1HGA_0427	0.543	0.045	0.542

Tabla 4.5: Índices de vegetación de la muestra S1HGA.

Nombre	NDVI	GNDVI	OSAVI
S1HGV_0390	0.513	0.086	0.512
S1HGV_0391	0.514	0.032	0.513
S1HGV_0392	0.538	0.127	0.537
S1HGV_0393	0.535	0.091	0.535
S1HGV_0401	0.585	0.195	0.585
S1HGV_0403	0.604	0.251	0.603
S1HGV_0404	0.620	0.197	0.620
S1HGV_0405	0.622	0.192	0.622
S1HGV_0406	0.616	0.258	0.616
S1HGV_0408	0.624	0.249	0.624

Tabla 4.6: Índices de vegetación de la muestra S1HGV.

Nombre	NDVI	GNDVI	OSAVI
S1HPA_0477	0.421	-0.175	0.421
S1HPA_0478	0.469	-0.069	0.468
S1HPA_0479	0.402	-0.146	0.402
S1HPA_0480	0.517	0.007	0.517
S1HPA_0481	0.509	0.033	0.508
S1HPA_0482	0.463	-0.058	0.463
S1HPA_0483	0.529	0.002	0.528
S1HPA_0484	0.488	0.017	0.487
S1HPA_0485	0.498	-0.030	0.498
S1HPA_0486	0.520	0.054	0.519

Tabla 4.7: Índices de vegetación de la muestra S1HPA.

Nombre	NDVI	GNDVI	OSAVI
S1HPV_443	0.515	0.079	0.515
S1HPV_444	0.524	0.099	0.524
S1HPV_445	0.528	0.066	0.528
S1HPV_446	0.510	0.009	0.509
S1HPV_447	0.531	0.132	0.531
S1HPV_448	0.540	0.129	0.539
S1HPV_449	0.568	0.115	0.567
S1HPV_450	0.555	0.147	0.555
S1HPV_451	0.567	0.140	0.567
S1HPV_452	0.556	0.141	0.555

Tabla 4.8: Índices de vegetación de la muestra S1HPV.

Surco 3

Nombre	NDVI	GNDVI	OSAVI
S3HGA_0299	0.478	-0.034	0.477
S3HGA_0300	0.478	-0.031	0.478
S3HGA_0301	0.475	-0.056	0.474
S3HGA_0302	0.482	-0.005	0.481
S3HGA_0303	0.412	-0.067	0.411
S3HGA_0304	0.528	0.098	0.527
S3HGA_0305	0.506	0.018	0.506
S3HGA_0306	0.515	0.057	0.515
S3HGA_0307	0.544	0.092	0.543
S3HGA_0308	0.491	-0.026	0.491

Tabla 4.9: Índices de vegetación de la muestra S3HGA.

Nombre	NDVI	GNDVI	OSAVI
S3HGV_232	0.494	0.106	0.493
S3HGV_233	0.493	0.042	0.493
S3HGV_236	0.535	0.140	0.535
S3HGV_237	0.540	0.124	0.540
S3HGV_238	0.535	0.099	0.534
S3HGV_239	0.533	0.150	0.532
S3HGV_240	0.545	0.176	0.544
S3HGV_241	0.558	0.195	0.558
S3HGV_242	0.557	0.124	0.556
S3HGV_243	0.566	0.212	0.565

Tabla 4.10: Índices de vegetación de la muestra S3HGV.

Nombre	NDVI	GNDVI	OSAVI
S3HPA_0258	0.479	-0.066	0.479
S3HPA_0259	0.432	-0.130	0.431
S3HPA_0260	0.469	-0.056	0.468
S3HPA_0261	0.371	-0.158	0.371
S3HPA_0262	0.498	-0.006	0.498
S3HPA_0263	0.490	0.020	0.490
S3HPA_0264	0.441	-0.1	0.440
S3HPA_0266	0.525	0.062	0.525
S3HPA_0267	0.464	-0.047	0.464
S3HPA_0268	0.434	-0.061	0.434

Tabla 4.11: Índices de vegetación de la muestra S3HPA.

Nombre	NDVI	GNDVI	OSAVI
S3HPV_0194	0.518	0.082	0.517
S3HPV_0195	0.532	0.119	0.531
S3HPV_0196	0.528	0.033	0.528
S3HPV_0197	0.551	0.140	0.550
S3HPV_0198	0.552	0.168	0.551
S3HPV_0199	0.554	0.141	0.553
S3HPV_0200	0.550	0.194	0.55
S3HPV_0201	0.551	0.155	0.55
S3HPV_0202	0.560	0.168	0.56
S3HPV_0203	0.580	0.203	0.58

Tabla 4.12: Índices de vegetación de la muestra S3HPV.

Cálculo del promedio y la varianza de la textura de las imágenes multispectrales

Para hallar la textura de las imágenes multispectrales se hizo uso del descriptor *Local Binary Patterns* (LBP) o *Patrones Binarios Locales*, que básicamente es escoger un píxel central y comparar su valor con el de los píxeles vecinos.

```

1 for i in range(fil):
2     fila=fila+1
3
4     for j in range (col):
5         columna=columna+1
6         if fila>1 and fila<height and columna>1 and columna<width:
7             bit=[]
8             contador=0
9             textura2=0
10            if NIR[i-1,j-1]>= NIR[i,j]:
11                bit.append(1)
12            else:
13                bit.append(0)
14            if NIR[i-1,j]>= NIR[i,j]:
15                bit.append(1)
16            else:
17                bit.append(0)
18            if NIR[i-1,j+1]>= NIR[i,j]:
19                bit.append(1)
20            else:
21                bit.append(0)
22            if NIR[i,j+1]>= NIR[i,j]:
23                bit.append(1)
24            else:
25                bit.append(0)
26            if NIR[i+1,j+1]>= NIR[i,j]:
27                bit.append(1)
28            else:
29                bit.append(0)
30            if NIR[i+1,j]>= NIR[i,j]:
31                bit.append(1)
32            else:
33                bit.append(0)
34            if NIR[i+1,j-1]>= NIR[i,j]:
35                bit.append(1)
36            else:
37                bit.append(0)
38            if NIR[i,j-1]>= NIR[i,j]:
39                bit.append(1)
40            else:
41                bit.append(0)

```

```

1         for a in bit:
2             contador=contador+1
3             if a == 1:
4                 textura1 = 2**(contador-1)
5                 textura2 = textura2+textura1
6             else:
7                 textura2=textura2
8
9             NIR[i,j]=textura2
10
11
12     columna=0

```

Algoritmo 22: Establece 0 y 1.

El principio de LBP es asignarle 1 o 0 al píxel vecino dependiendo de si su valor es mayor o menor que el valor del píxel central respectivamente. Estos valores se van almacenando en un arreglo, y al finalizar la comparación del píxel central con sus píxeles vecinos, se calcula el valor de 0 a 255 correspondiente, esto se hace con el *for* que se encuentra en la línea 44 del algoritmo 22.

El programa recorre la imagen menos los bordes, pues se genera error debido a los píxeles de los bordes no tienen vecinos en todo su alrededor, pero esto no afecta el resultado, pues los píxeles de interés son los de la hoja.

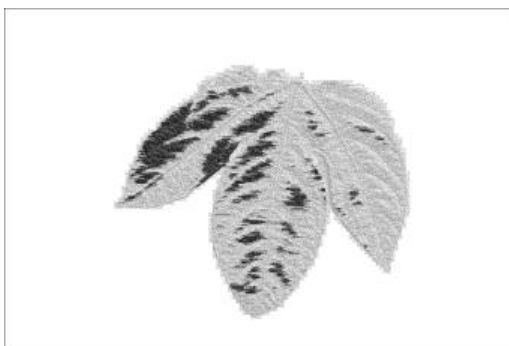


Figura 4.46: Textura de la imagen segmentada del canal verde.



Figura 4.47: Textura de la imagen segmentada del canal infrarrojo cercano.

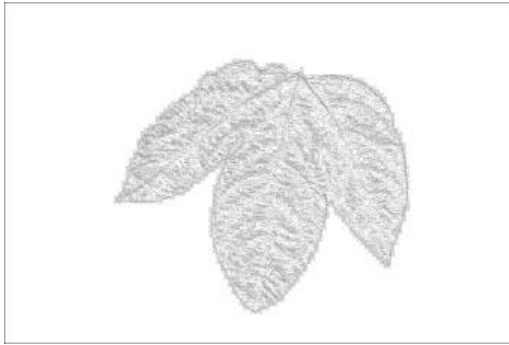


Figura 4.48: Textura de la imagen segmentada del canal rojo.

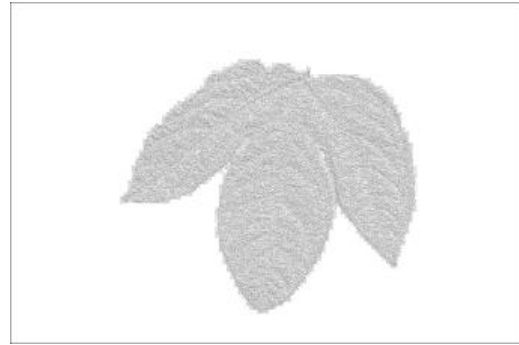


Figura 4.49: Textura de la imagen segmentada del canal del borde rojo.

Las figuras 4.46, 4.47, 4.48 y 4.49 muestran la textura de cada canal. Al igual que los índices de vegetación se obtienen matrices con los valores de la textura de cada píxel y para realizar el promedio de esos valores no se tuvo en cuenta los píxeles con valor de 255, pues esos píxeles eran los del fondo de la imagen.

```
1 i2=0
2 contador=0
3 for i in range(fil):
4     for j in range (col):
5         if NIR[i,j] < 255:
6             contador=contador+1
7             i2=NIR[i,j]+i2
8
9 promtext=i2/contador
```

Algoritmo 23: Promedio de la textura de la hoja.

En el algoritmo 23 se halla el promedio del total de los píxeles que conforman la hoja, con los valores que resultaron de la aplicación del LBP.

```

1 contador =0
2 i2=0
3 vartext=0
4 for i in range(fil):
5     for j in range (col):
6         if NIR[i,j] < 255:
7             contador=contador+1
8             i2=((NIR[i,j]-promtext)**2)
9             vartext=vartext+i2
10 vartext=vartext/contador
11 print ('El promedio es:' + str(float(promtext)))
12 print ('La varianza es:' + str(float(vartext)))

```

Algoritmo 24: Varianza de la textura de la hoja.

En el algoritmo 24 se calcula la varianza de la textura, tomando cada valor de la hoja y restándole el promedio de la textura, este resultado se eleva al cuadrado y al final del recorrido de la imagen se divide el valor obtenido entre la cantidad de píxeles.

4.2.4. Aprendizaje automático

El uso del aprendizaje automático permite realizar la predicción del estado de nitrógeno gracias a sus algoritmos de aprendizaje. Para esta estimación se utilizó el aprendizaje supervisado de tipo de clasificación, debido a que se realizó el proceso de etiquetar las muestras teniendo en cuenta los resultados de las pruebas químicas entregados por el laboratorio. El entrenamiento es realizado a partir de los datos obtenidos en la extracción de características de las imágenes. Los resultados arrojados por el laboratorio después de realizar las pruebas químicas a las muestras, fueron:

Muestra	Porcentaje de N (%)
S1HGA	1.20
S1HGV	1.11
S3HPV	0.91
S1HPV	0.80
S3HGA	0.76
S1HPA	0.70
S3HGV	0.52
S3HPA	0.46

Tabla 4.13: Resultados de las pruebas químicas.

Teniendo en cuenta la tabla anterior, los resultados se dividieron en dos grupos, el primero que está en el rango de porcentaje de nitrógeno de 0.80% a 1.20% y el segundo que está en el rango de 0.46% a 0.76% y de esa misma manera se etiquetaron las muestras por grupos.

Muestra	Porcentaje de N (%)
S1HGA S1HGV S3HPV S1HPV	0.80_y_1.20
S3HGA S1HPA S3HGV S3HPA	0.46_y_0.76

Tabla 4.14: Etiquetas del conjunto de muestras.

Las posibles salidas o respuestas que entregará la herramienta (“0.80_y_1.20” y “0.46_y_0.76”) serán las “clases”, este es un problema de clasificación de dos clases, debido a que cada muestra pertenece a una de los dos estados de nitrógeno.

Para este proceso se siguieron los pasos de: cargar datos, análisis de datos, visualización de los datos, entrenamiento y validación de datos.

Cargar datos

Los datos cargados al programa fueron los archivos csv que se crearon con la extracción de características.

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 import numpy as np
4 import seaborn as sns
5 import sklearn
6 import matplotlib.pyplot as plt
7 import pandas as pd
8 import statistics as stats
9 import os
10 import argparse
11 import csv
12 from sklearn import metrics
13 from sklearn.neighbors import KNeighborsClassifier
14 from sklearn.linear_model import LogisticRegression
15 from sklearn.model_selection import train_test_split
16 from sklearn.tree import DecisionTreeClassifier
17 from sklearn.discriminant_analysis import
18     LinearDiscriminantAnalysis
19 from sklearn.naive_bayes import GaussianNB
20 from sklearn.svm import SVC
21 from sklearn.model_selection import KFold
22 from sklearn.model_selection import cross_val_score
23 from sklearn.metrics import accuracy_score
24 from sklearn.metrics import confusion_matrix
25 from sklearn.metrics import classification_report
26 from string import ascii_uppercase
27 from collections import Counter
28 from random import randrange
29
30 surco1=pd.read_csv('datos_trainG6.csv')
31 surco1=surco1.drop('NOMBRE',axis=1)
```

Algoritmo 25: Bibliotecas usadas para el aprendizaje automático.

Se debe resaltar que la biblioteca que contiene los algoritmos de aprendizajes utilizados es *sklearn*. El archivo que contiene el conjunto de datos fue ubicado en la misma

carpeta donde se encuentra el código, y se cargó colocando el nombre del archivo en el programa, además de esto se eliminó la columna que contenía los nombres de las muestras.

Análisis de datos

```
1 print(surco1.head(10))
2 print(surco1.describe())
3 print(surco1.groupby('NITROGENO').size())
4 print(surco1.shape)
```

Algoritmo 26: Código para imprimir información del conjunto de datos.

La información del conjunto de datos indica que todas las columnas tienen 60 datos, en las primeras se muestran datos flotantes y en la última contiene datos objeto y ahí se encuentra la información del estado de nitrógeno de la hoja. Cada fila representa una muestra de una hoja y las primeras tres columnas son las medidas que se obtuvieron en el proceso de la extracción de características.

```

1 zaira@debian:~/Documentos/GRAFICAS$ python3 GRAFICARindices.py
2          NDVI          GNDVI          OSAVI          NITROGENO
3 0    0.513480    0.086370    0.512770    0.80__y__1.20
4 1    0.514450    0.032570    0.513780    0.80__y__1.20
5 2    0.538560    0.127890    0.537850    0.80__y__1.20
6 3    0.535920    0.091590    0.535220    0.80__y__1.20
7 .
8 .
9 .
10 56   0.618350   0.255980   0.617710   0.46__y__0.76
11 57   0.618070   0.244990   0.617440   0.46__y__0.76
12 58   0.616730   0.280680   0.616100   0.46__y__0.76
13 59   0.613400   0.197330   0.612780   0.46__y__0.76
14 Información del dataset:
15 <class 'pandas.core.frame.DataFrame'>
16 RangeIndex: 60 entries, 0 to 59
17 Data columns (total 4 columns):
18 #   Column          Non-Null Count  Dtype
19 ---  ---
20 0   NDVI             60 non-null     float64
21 1   GNDVI            60 non-null     float64
22 2   OSAVI            60 non-null     float64
23 3   NITROGENO        60 non-null     object
24 dtypes: float64(3), object(1)
25 memory usage: 2.0+ KB
26 None
27
28          NDVI          GNDVI          OSAVI
29 count    60.000000    60.000000    60.000000
30 mean      0.546348     0.117453     0.545717
31 std       0.057135     0.109092     0.057105
32 min       0.371712    -0.158241     0.371272
33 25%       0.514207     0.057263     0.513527
34 50%       0.542669     0.133786     0.542089
35 75%       0.596002     0.197425     0.595355
36 max       0.641190     0.280680     0.640520
37 NITROGENO
38 0.46__y__0.76    30
39 0.80__y__1.20    30
40 dtype: int64
41 (60, 4)

```

Algoritmo 27: Resultado de la información del conjunto de datos.

Se debe recordar que en aprendizaje automático los elementos individuales son llamados *muestras* y sus propiedades son llamadas *características*.

En las líneas 38 y 39 se evidencia la cantidad de datos que hay para cada clase, es decir, para cada estado de nitrógeno establecido. Esto significa que la cantidad de datos está balanceada, teniendo la misma cantidad de muestras para cada clase (0.46 *_y_0* .76 y 0.80 *_y_1* .20), de lo contrario el aprendizaje podrá ser tendencioso hacia un tipo de respuesta y cuando el modelo intente generalizar el conocimiento fallará.

Se observa el valor medio, desviación estándar, percentiles, valor mínimo y máximo, esto se logra con la función *describe*, de esta manera se puede tener una visión estadística del conjunto de datos.

Visualización de los datos

Antes de construir un modelo de aprendizaje automático se debe inspeccionar los datos para ver si el problema se puede resolver fácilmente sin el aprendizaje automático, o si la información deseada no se encuentra en los datos. Además, esto permite visualizar alguna anomalía que puedan tener los datos, pues en el mundo real las inconsistencias en los datos son muy comunes.

En este documento la visualización de los datos se verán en dos tipos de gráficas, de cajas y bigotes y de dispersión.

Para las gráficas de cajas y bigotes se usaron las siguientes líneas de código:

```
1 plt.figure(figsize=(15,20))
2 plt.subplot(3,2,1)
3 sns.boxplot(x='NITROGENO',y='NDVI',data=surco1)
4 plt.subplot(3,2,2)
5 sns.boxplot(x='NITROGENO',y='GNDVI',data=surco1)
6 plt.subplot(3,2,3)
7 sns.boxplot(x='NITROGENO',y='OSAVI',data=surco1)
8
9 plt.savefig('GRUPO2bigotesindices.pdf', dpi=400, bbox_inches='tight', pad_inches=0.1)
```

Algoritmo 28: Generar diagrama de cajas y bigotes.

Y se obtuvo la siguiente gráfica:

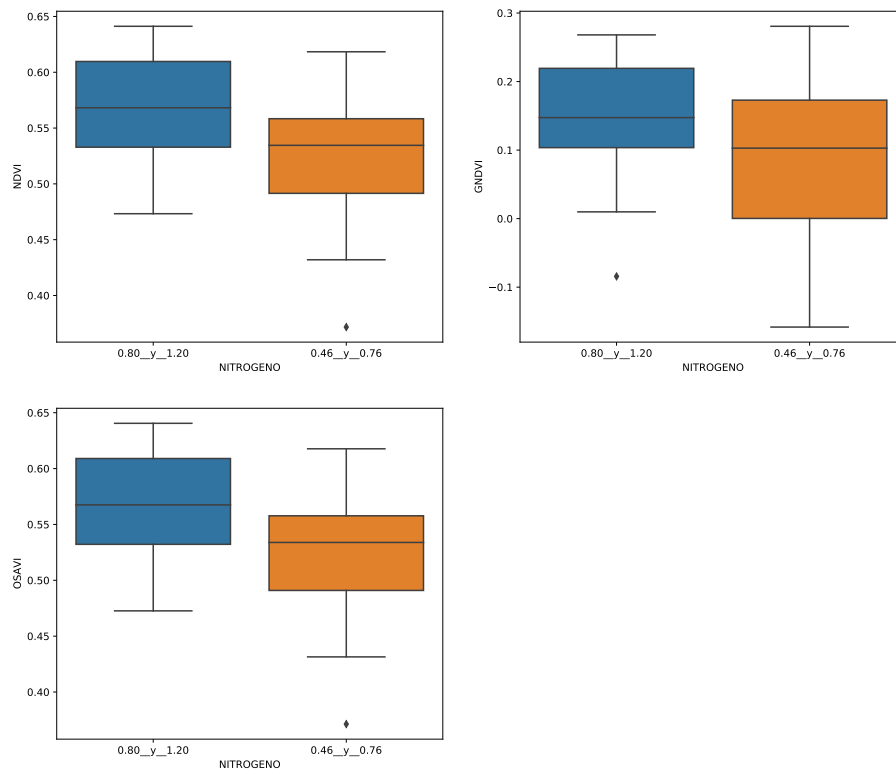


Figura 4.50: Diagramas de cajas y bigotes de los índices de vegetación.

En la figura 4.50 se puede evidenciar que los grupos de datos tienen gran cantidad de variabilidad, es decir, no se diferencia muy bien las clases.

Se puede ver que los datos de NDVI y de OSAVI presentan mucha similitud debido a que sus datos son muy similares.

Para la gráfica de dispersión se utilizaron las siguientes líneas de código:

```
1 sns.pairplot(surco1,hue='NITROGENO', height=2, diag_kind="kde")
2 sns.pairplot(surco1,hue='NITROGENO')
3 plt.savefig('GRUPO2dispersionindices.pdf', dpi=400, bbox_inches='
    tight', pad_inches=0.1)
```

Algoritmo 29: Generar diagrama de dispersión de los índices de vegetación.

Y se obtuvo la siguiente gráfica:

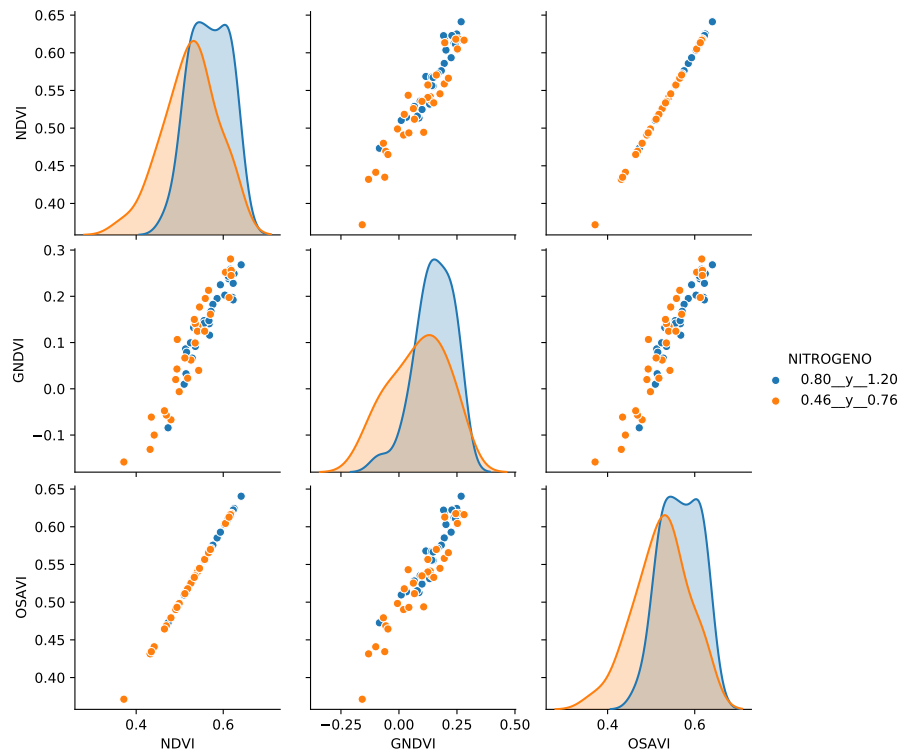


Figura 4.51: Diagramas de dispersión de los índices de vegetación.

En la figura 4.51 se puede ver que los datos presentan una variabilidad alta y la diferenciación entre los dos estados de nitrógeno se torna difícil.

Ahora, para tener mayor certeza del tipo de correlación de cada gráfica, se generó la matriz de correlación, en la cual se debe tener en cuenta:

- Correlación positiva: son los valores mayores a cero y están relacionados directamente
- Correlación negativa: son los valores menores a cero y están relacionados inversamente.
- Correlación cercana a cero: no hay correlación.

1		NDVI	GNDVI	OSAVI
2	NDVI	1.00000	0.947480	1.000000
3	GNDVI	0.94748	1.000000	0.947437
4	OSAVI	1.00000	0.947437	1.000000

Algoritmo 30: Resultados de matriz de correlación.

En el resultado de la matriz de correlación se puede ver que los valores al ser altos presentan una correlación fuerte, esto quiere decir que los datos tienen mucha similitud.

Entrenamiento y validación de datos

Antes de aplicar el modelo a los conjuntos nuevos de datos es necesario conocer si se puede confiar en dicho modelo, para lo cual se debe tener en cuenta que no se puede evaluar el modelo con los mismos datos de entrenamiento, pues el modelo siempre podrá recordar el conjunto de entrenamiento, por lo tanto, es muy posible que prediga la etiqueta correcta para cualquier punto del conjunto de entrenamiento[22].

Para evaluar el rendimiento del modelo es necesario mostrarle al modelo nuevos datos (que no ha visto antes) para los cuales se tiene etiquetas, por lo cual se realiza la división del conjunto de datos total en dos, uno para el entrenamiento y el otro para la validación, para esto se suele usar un porcentaje de 80 %-20 % respectivamente y se toman muestras aleatorias, no en secuencia[22].

La parte de datos que es usada para la construcción del modelo es llamada *datos de entrenamiento* o *conjunto de entrenamiento* y el resto de datos que se usa para saber que tan bien funciona el modelo se denominan *datos de prueba*, *conjunto de prueba* o *conjunto de reserva*.

Scikit-learn contiene la función *train_ test_ split* que se encarga de mezclar y dividir el conjunto de datos. Lo que hace esta función es extraer el 80 % de los datos con sus respectivas etiquetas para el conjunto de entrenamiento, y el 20 % restante para el conjunto de prueba[22].

En Scikit-learn usualmente se denota los datos con X mayúscula y las etiquetas con

y minúscula[22].

```
1 X = array[:,0:3]
2 Y = array[:,3]
3 validation_size = 0.2
4 seed = randrange(10)#número pseudoaleatorio para mezclar los datos
5 print ("seed: ",seed)
6 #División de los datos en train y test con 80-20
7 X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size
    =validation_size,
8 random_state=seed)
9 y_pred=y_test
```

Algoritmo 31: División de datos.

La función *train_test_split* mezcla el conjunto de datos utilizando un generador de números pseudoaleatorios antes de realizar la división. Esto se hace con el fin de asegurar que el conjunto de prueba contenga datos de todas las clases[22].

X_{train} contiene el 80% de los datos, y_{train} son las “etiquetas” de los resultados esperados de X_{train} X_{test} contiene el restante 20% de los datos que serán usados para la validación, y_{test} son las “etiquetas” de los resultados de X_{test} .

En el momento de realizar el entrenamiento puede suceder que las métricas de *train* y *test* den valores muy diferentes, por lo cual sirve usar el modelo de Cross-Validation. En este punto existirá un set de validación, el cual no se trata de un tercer grupo si no que se encuentra dentro del conjunto de *train*.

Para esta validación se usará *K-folds* que en este caso será de 10 splits para entrenar, esto quiere decir que no le pasarán todos los datos al modelo de una vez, si no que $k-1$ grupos se emplean para entrenarlo y uno de los grupos se emplea como test, este proceso se repite k veces utilizando un grupo distinto como test en cada iteración. El proceso genera k estimaciones del test error cuyo promedio se emplea

como estimación final, en este entrenamiento k será igual a 10:

```
1 results = []
2 names = []
3 models = []
4
5 models.append(('LDA', LinearDiscriminantAnalysis()))
6 models.append(('CART', DecisionTreeClassifier()))
7 models.append(('NB', GaussianNB()))
8 models.append(('SVM', SVC()))
9 models.append(('KNC', KNeighborsClassifier(2)))
10
11 for name, model in models:
12     kf = KFold(n_splits=10)
13     clf = model
14     clf.fit(X_train, y_train)
15     score = clf.score(X_train, y_train)
16     print("-----Modelo ", model, "-----")
17     print("Metrica del modelo", score)
18     results.append(score)
19     names.append(name)
20     scores = cross_val_score(clf, X_train, y_train, cv=kf, scoring=
"accuracy")
21     print("Metricas cross_validation", scores)
22     print("Media de cross_validation", scores.mean())
23     preds = clf.predict(X_test)
24     score_pred = metrics.accuracy_score(y_test, preds)
25     print("Metrica en Test", score_pred)
```

Algoritmo 32: Entrenamiento de maquina usando los datos.

Para realizar el entrenamiento con diferentes modelos, estos fueron guardados en

una lista para que por medio de un *for* fueran aplicados todos a los datos.

```
1 -----Modelo  LinearDiscriminantAnalysis() -----
2 Metrica del modelo 0.7125
3 Metricas cross_validation [0.75  0.625 0.5   0.875 0.875 0.75  0.5
   0.75  0.5   0.625]
4 Media de cross_validation 0.675
5 Metrica en Test 0.8
6 -----Modelo  DecisionTreeClassifier() -----
7 Metrica del modelo 1.0
8 Metricas cross_validation [0.75  0.625 0.375 0.25  0.25  0.5   0.5
   0.625 0.75  0.375]
9 Media de cross_validation 0.5
10 Metrica en Test 0.7
11 -----Modelo  GaussianNB() -----
12 Metrica del modelo 0.6375
13 Metricas cross_validation [0.5   0.25  0.375 0.875 0.75  0.875
   0.625 0.75  0.75  0.5   ]
14 Media de cross_validation 0.625
15 Metrica en Test 0.6
16 -----Modelo  SVC() -----
17 Metrica del modelo 0.65
18 Metricas cross_validation [0.5   0.25  0.25  0.75  0.5   0.875
   0.625 0.625 0.75  0.625]
19 Media de cross_validation 0.575
20 Metrica en Test 0.5
21 -----Modelo  KNeighborsClassifier(n_neighbors=2) -----
22 Metrica del modelo 0.7625
23 Metricas cross_validation [0.625 0.375 0.25  0.5   0.25  0.625 0.75
   0.5   0.375 0.375]
24 Media de cross_validation 0.4625
25 Metrica en Test 0.75
```

Algoritmo 33: Resultado del entrenamiento.

Las líneas de *Metricas cross_validation* contienen los resultados de cada grupo formado con el grupo de entrenamiento X_{train} , lo que se mencionó anteriormente como *K-folds*. Se pueden observar los porcentajes de certeza por cada modelo, y esto se logra al usar los datos que se tenían reservados para prueba, es decir, al 20% de los datos, pues estos datos no se utilizaron para la construcción del modelo pero sus etiquetas son conocidas. De esta manera se puede ver que tan bien funcionan los

modelos, por ejemplo, el modelo de k vecinos más cercanos obtuvo una precisión de 0.75, lo que indica que se puede esperar que el modelo realice la predicción correcta el 75 % de las veces y del modelo de análisis de discriminante lineal el 80 %.

Validación de la precisión

Después de observar el comportamiento de los modelos se escogió el modelo de K vecinos más cercanos y el de análisis de discriminante lineal. La construcción del modelo de K vecinos más cercanos consiste en almacenar el conjunto de entrenamiento. Para hacer una predicción para un nuevo punto de datos, el algoritmo encuentra el punto en el conjunto de entrenamiento que está más cerca del nuevo punto. Luego, asigna la etiqueta de este punto de entrenamiento de datos más cercano al nuevo punto de datos. La letra k en k vecinos más cercanos, indica que se puede utilizar diferentes números de vecinos más cercanos. Una k muy pequeña puede hacer que el modelo clasifique de manera incorrecta, por ejemplo, si algunos ejemplos fueron clasificados incorrectamente, Se predecirá que cualquier ejemplo no etiquetado que sea el más cercano al vecino etiquetado incorrectamente tendrá la clase incorrecta, incluso si otros nueve vecinos más cercanos hubieran votado de manera diferente.

```
1 knc = KNeighborsClassifier(2)
2 knc.fit(X_train, y_train)
3 predictions = knc.predict(X_test)
4 print(accuracy_score(y_test, predictions))
5 print(confusion_matrix(y_test, predictions))
6 print(classification_report(y_test, predictions))
```

Algoritmo 34: Validación del entrenamiento del algoritmo de k vecinos más cercanos.

La matriz de confusión del algoritmo de k vecinos más cercanos es:

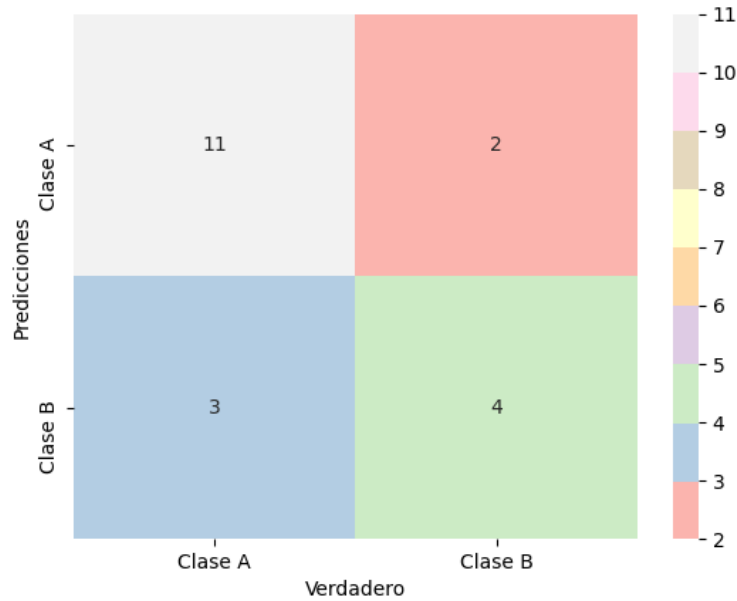


Figura 4.52: Matriz de confusión del algoritmo KNC.

En la figura 4.52 la *Clase A* representa al rango de 0.46 --y-- 0.76 y la *Clase B* representa a 0.80--y--1.20, se observa que de 13 datos que el programa debía clasificar como de la *Clase A*, 11 de ellos los clasificó correctamente y 2 de ellos los clasificó de manera errónea, en cuando a los 7 datos que debía clasificar de la *Clase B* 4 los clasificó de manera correcta y 3 erróneamente. También se puede ver de la siguiente manera:

	0.46 --y-- 0.76 (Predicción)	0.80--y--1.20 (Predicción)
0.46 --y-- 0.76 (Real)	11	2
0.80--y--1.20(Real)	3	4

Tabla 4.15: Matriz de confusión del algoritmo KNC.

De la tabla 4.15 se puede observar que se obtuvo 11 datos como verdaderos positivos (VP), 4 datos verdaderos negativos(VN), 2 datos falsos negativos (FN) y 3 datos falsos positivos (FP). Con estos datos se pueden obtener los siguientes porcentajes:

$$\text{Exactitud} = \frac{11 + 4}{20} = 0,75 \quad (4.1)$$

$$\text{Tasa de error} = \frac{3 + 2}{20} = 0,25 \quad (4.2)$$

$$\text{Sensibilidad} = \frac{11}{13} = 0,85 \quad (4.3)$$

$$\text{Especificidad} = \frac{4}{7} = 0,57 \quad (4.4)$$

$$\text{Precisión} = \frac{11}{14} = 0,79 \quad (4.5)$$

$$\text{VPN} = \frac{4}{6} = 0,66 \quad (4.6)$$

```
1 lda=LinearDiscriminantAnalysis()  
2 lda.fit(X_train, y_train)  
3 predictions2 = lda.predict(X_test)  
4 print(accuracy_score(y_test, predictions2))  
5 print(confusion_matrix(y_test, predictions2))  
6 print(classification_report(y_test, predictions2))
```

Algoritmo 35: Validación del entrenamiento del algoritmo de análisis de discriminante lineal.

La matriz de confusión del algoritmo de análisis de discriminante lineal es:

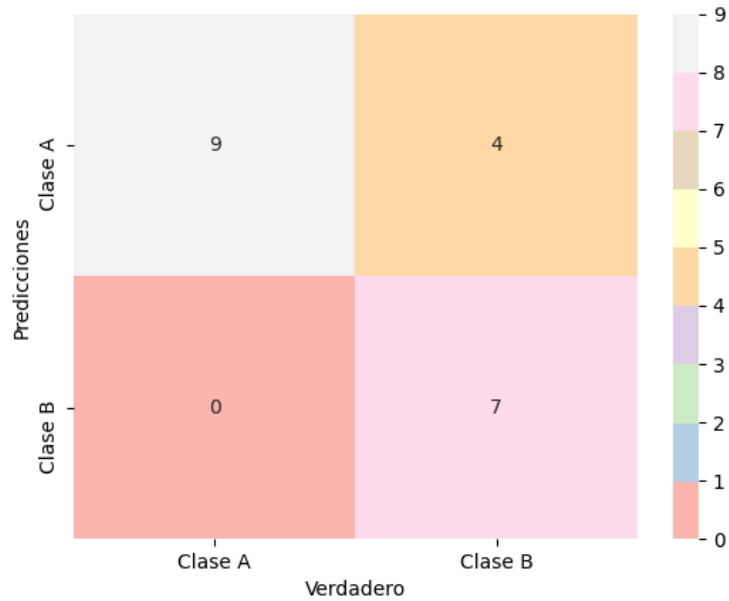


Figura 4.53: Matriz de confusión del algoritmo LDA.

	0.46 --y-- 0.76 (Predicción)	0.80--y--1.20 (Predicción)
0.46 --y-- 0.76 (Real)	9	4
0.80--y--1.20(Real)	0	7

Tabla 4.16: Matriz de confusión del algoritmo LDA.

Los porcentajes del comportamiento del modelo LDA son:

$$\text{Exactitud} = \frac{9 + 4}{20} = 0,65 \quad (4.7)$$

$$\text{Tasa de error} = \frac{0 + 4}{20} = 0,2 \quad (4.8)$$

$$\text{Sensibilidad} = \frac{9}{13} = 0,69 \quad (4.9)$$

$$\text{Especificidad} = \frac{7}{7} = 1 \quad (4.10)$$

$$\text{Precisión} = \frac{9}{9} = 1 \quad (4.11)$$

$$\text{VPN} = \frac{7}{11} = 0,63 \quad (4.12)$$

Para la validación final de estos modelos se organizó un archivo csv con características de muestras que el programa no había visto antes, es decir, datos que no se encontraban en el archivo csv cargado inicialmente para el entrenamiento de los modelos. Los datos de este nuevo archivo csv serían completamente nuevos para el programa, obteniendo la siguiente clasificación:

Nombre	Predicción esperada		Predicción KNC	
	0.80__y__1.20	0.46__y__0.76	0.80__y__1.20	0.46__y__0.76
S1HGA_0423	x		x	
S1HGA_0425	x		x	
S1HGA_0426	x		x	
S1HGA_0427	x			x
S1HGA_0429	x			x
S1HGA_0430	x			x
S1HGA_0431	x		x	
S1HGA_0432	x			x
S1HGA_0433	x		x	
S1HGA_0434	x		x	
S1HPA_0477		x		x
S1HPA_0478		x		x
S1HPA_0479		x		x
S1HPA_0480		x	x	
S1HPA_0481		x		x
S1HPA_0482		x		x
S1HPA_0483		x	x	
S1HPA_0484		x		x
S1HPA_0485		x		x
S1HPA_0486		x		x

Tabla 4.17: Resultados obtenidos de la clasificación de datos nuevos utilizando el modelo KNC para los índices de vegetación.

Nombre	Predicción esperada		Predicción LDA	
	0.80--y--1.20	0.46--y--0.76	0.80--y--1.20	0.46--y--0.76
S1HGA_0423	x		x	
S1HGA_0425	x		x	
S1HGA_0426	x			x
S1HGA_0427	x		x	
S1HGA_0429	x		x	
S1HGA_0430	x		x	
S1HGA_0431	x		x	
S1HGA_0432	x		x	
S1HGA_0433	x		x	
S1HGA_0434	x		x	
S1HPA_0477		x		x
S1HPA_0478		x		x
S1HPA_0479		x		x
S1HPA_0480		x	x	
S1HPA_0481		x	x	
S1HPA_0482		x		x
S1HPA_0483		x	x	
S1HPA_0484		x		x
S1HPA_0485		x		x
S1HPA_0486		x		x

Tabla 4.18: Resultados obtenidos de la clasificación de datos nuevos utilizando el modelo LDA para los índices de vegetación.

En las tablas 4.17 y 4.18 se muestran los resultados de la clasificación realizadas por el modelo KNC y LDA respectivamente. Se observa que tienen un comportamiento similar, de 20 muestras para clasificar, el modelo KNC clasificó correctamente 14, en cuanto al modelo LDA, 16 muestras fueron clasificadas correctamente. Se debe aclarar que estas muestras también hacen parte de los resultados de las muestras entregados por el laboratorio.

Los pasos anteriores para lograr la clasificación de los estados de nitrógeno, también fueron realizados para las características extraídas de las imágenes RGB y para las características de textura obtenidas de las imágenes multiespectrales. Lo única diferencia fueron los archivos csv utilizados para el entrenamiento, los archivos csv utilizados para las pruebas y los modelos de aprendizaje utilizados.

Aprendizaje automático con datos de promedio y varianza de textura en imágenes multiespectrales

Además de características de índices de vegetación, también se extrajeron características de promedio y varianza de la textura de los cuatro canales de las imágenes multiespectrales (rojo, verde, infrarrojo cercano y borde rojo), las cuales fueron utilizadas para implementar otro modelo de aprendizaje. La visualización de los datos se realizó con un diagrama de dispersión.



Figura 4.54: Gráfico de dispersión del promedio y la varianza de la textura.

Después de realizar varias pruebas y realizar la visualización de los datos, se decidió usar los datos del promedio del canal del infrarrojo cercano y los datos de varianza de los 4 canales. En la gráfica 4.54 se puede ver que los datos tienen gran variabilidad, a pesar de esto se logró diferenciar en cierta medida los rangos de nitrógeno representados.

La matriz de correlación obtenida de los datos de promedio y varianza de cada canal

de las imágenes multiespectrales fue:

	PROMNIR	VARGRE	VARNIR	VARRED	VARREG	
1						
2	PROMNIR	1.000000	0.106626	-0.922570	0.218777	-0.197994
3	VARGRE	0.106626	1.000000	-0.128906	0.339723	0.290232
4	VARNIR	-0.922570	-0.128906	1.000000	-0.182284	0.109028
5	VARRED	0.218777	0.339723	-0.182284	1.000000	0.677389
6	VARREG	-0.197994	0.290232	0.109028	0.677389	1.000000

Algoritmo 36: Matriz de correlación de los datos de promedio y varianza de la textura.

La matriz de correlación de los datos de promedio y varianza de la textura, muestra una correlación baja en la mayoría de sus datos, lo que beneficia el aprendizaje automático.

Para esta clasificación se decide usar los modelos de K vecinos más cercanos, análisis de discriminante lineal y de árbol de decisión. La matriz de confusión se muestra a continuación:

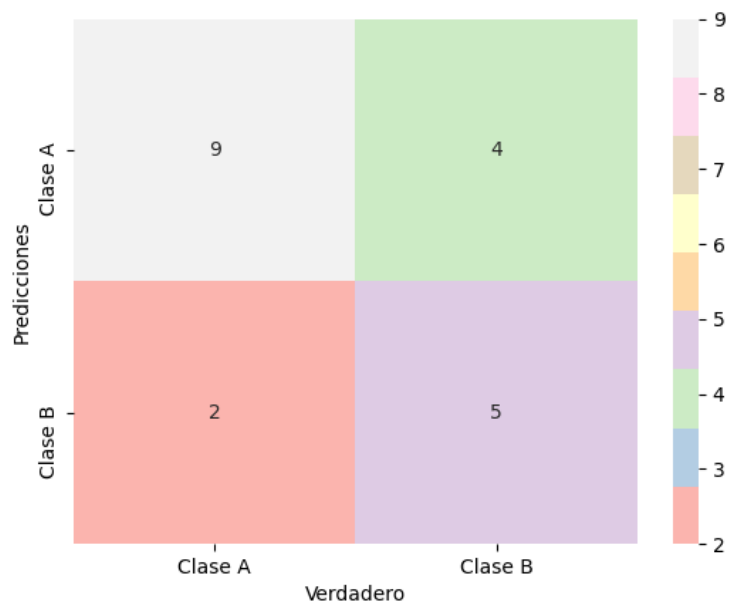


Figura 4.55: Matriz de confusión del algoritmo KNC del promedio y la varianza de la textura.

	0.46 __y__0 .76 (Predicción)	0.80__y__1 .20 (Predicción)
0.46 __y__0 .76 (Real)	9	4
0.80__y__1 .20(Real)	2	5

Tabla 4.19: Matriz de confusión del algoritmo KNC para datos de textura.

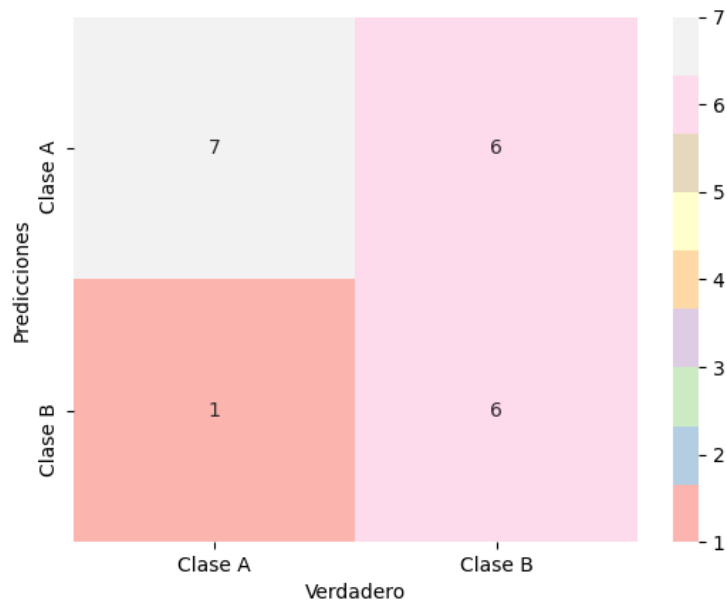


Figura 4.56: Matriz de confusión del algoritmo LDA del promedio y la varianza de la textura.

	0.46 __y__0 .76 (Predicción)	0.80__y__1 .20 (Predicción)
0.46 __y__0 .76 (Real)	7	6
0.80__y__1 .20(Real)	1	6

Tabla 4.20: Matriz de confusión del algoritmo LDA para datos de textura.

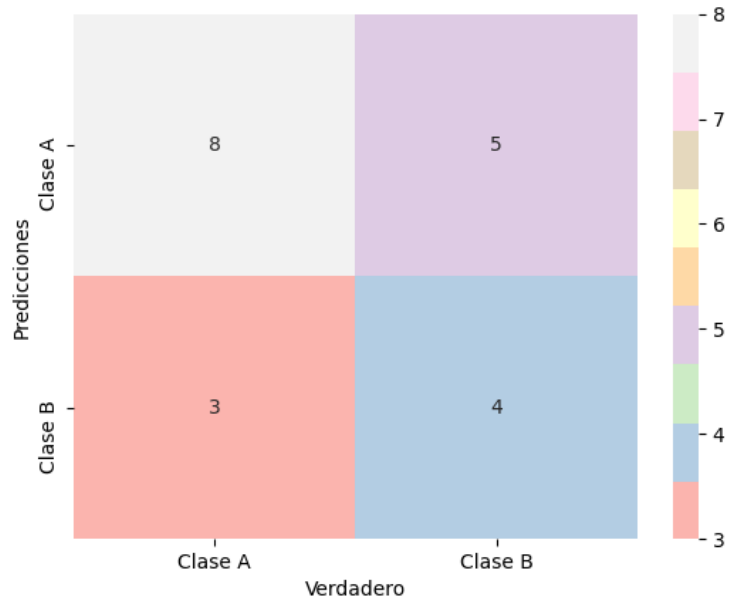


Figura 4.57: Matriz de confusión del algoritmo DTC del promedio y la varianza de la textura.

	0.46 __y__0 .76 (Predicción)	0.80__y__1 .20 (Predicción)
0.46 __y__0 .76 (Real)	8	5
0.80__y__1 .20(Real)	3	4

Tabla 4.21: Matriz de confusión del algoritmo DTC para datos de textura.

Las tablas 4.19, 4.20 y 4.21 muestra las matrices de confusión de los modelos empleados con las características de textura extraídas de las imágenes multiespectrales. a continuación se muestran los porcentajes de comportamiento que presentan dichas matrices.

	KNC	LDA	DTC
Exactitud	0.7 %	0.65 %	0.6 %
Tasa de error	0.3 %	0.35 %	0.4 %
Sensibilidad	0.69 %	0.53 %	0.61 %
Especificidad	0.71 %	0.85 %	0.57 %
Precisión	0.81 %	0.87 %	0.72 %
Valor de predicción negativo	0.55 %	0.5 %	0.44 %

Tabla 4.22: Porcentajes de comportamiento de los modelos KNC, LDA y DTC para los datos de textura.

La tabla 4.22 muestra que el modelo con mayor porcentaje de exactitud es el modelo KNC, en el cual se podría confiar el 70 % de las veces en su clasificación. También se realizó una última validación de estos modelos utilizando un conjunto de datos que no fueron utilizados para el entrenamiento ni para la validación anterior.

Nombre	Predicción esperada		Predicción KNC	
	0.80_y_1.20	0.46_y_0.76	0.80_y_1.20	0.46_y_0.76
S1HPV_0470	x		x	
S1HPV_0471	x		x	
S1HPV_0472	x		x	
S1HPV_0473	x		x	
S1HPV_0474	x		x	
S1HPV_0475	x		x	
S3HPV_0194	x		x	
S3HPV_0195	x		x	
S3HPV_0196	x		x	
S3HPV_0197	x		x	
S1HPA_0479		x		x
S1HPA_0480		x	x	
S1HPA_0481		x	x	
S1HPA_0482		x		x
S1HPA_0483		x	x	
S1HPA_0484		x	x	
S1HPA_0485		x		x
S1HPA_0486		x		x
S1HPA_0487		x		x
S1HPA_0488		x		x

Tabla 4.23: Resultados obtenidos de la clasificación de datos nuevos de textura utilizando el modelo KNC.

Nombre	Predicción esperada		Predicción LDA	
	0.80--y--1.20	0.46--y--0.76	0.80--y--1.20	0.46--y--0.76
S1HPV_0470	x		x	
S1HPV_0471	x		x	
S1HPV_0472	x		x	
S1HPV_0473	x		x	
S1HPV_0474	x		x	
S1HPV_0475	x		x	
S3HPV_0194	x		x	
S3HPV_0195	x		x	
S3HPV_0196	x		x	
S3HPV_0197	x		x	
S1HPA_0479		x		x
S1HPA_0480		x	x	
S1HPA_0481		x	x	
S1HPA_0482		x		x
S1HPA_0483		x	x	
S1HPA_0484		x		x
S1HPA_0485		x		x
S1HPA_0486		x	x	
S1HPA_0487		x	x	
S1HPA_0488		x		x

Tabla 4.24: Resultados obtenidos de la clasificación de datos nuevos de textura utilizando el modelo LDA.

Nombre	Predicción esperada		Predicción DTC	
	0.80__y__1.20	0.46__y__0.76	0.80__y__1.20	0.46__y__0.76
S1HPV_0470	x		x	
S1HPV_0471	x		x	
S1HPV_0472	x		x	
S1HPV_0473	x		x	
S1HPV_0474	x		x	
S1HPV_0475	x			x
S3HPV_0194	x		x	
S3HPV_0195	x		x	
S3HPV_0196	x		x	
S3HPV_0197	x		x	
S1HPA_0479		x		x
S1HPA_0480		x		x
S1HPA_0481		x		x
S1HPA_0482		x		x
S1HPA_0483		x		x
S1HPA_0484		x		x
S1HPA_0485		x		x
S1HPA_0486		x		x
S1HPA_0487		x		x
S1HPA_0488		x		x

Tabla 4.25: Resultados obtenidos de la clasificación de datos nuevos de textura utilizando el modelo DTC.

En las tablas 4.23, 4.24 y 4.25, se muestran los resultados de la clasificación realizada por los modelos KNC, LDA y DTC con los datos de textura, al comparar esas tablas con la tabla 4.22, se puede ver que a pesar de que el modelo DTC presentó menor exactitud comparado con los otros modelos, utilizando las nuevas muestras para validación este modelo realiza mejor clasificación que los otros dos modelos, pues de 20 muestras clasificó correctamente 19.

Aprendizaje automático con datos de promedio, varianza y desviación estándar en imágenes RGB

Las características utilizadas fueron el promedio, la varianza y la desviación estándar de los canales de las imágenes RGB, se debe recordar que estas características fueron

obtenidas gracias al código del ingeniero Santiago Trujillo [1]. Al realizar el gráfico de dispersión se obtuvo lo siguiente:



Figura 4.58: Gráfico de dispersión de las características de promedio, varianza y desviación estándar de las imágenes RGB .

Al observar la gráfica 4.58 se puede notar que los datos presentan una alta variabi-

idad. Ahora veamos la matriz de correlación.

	MED_B	VAR_R	VAR_G	VAR_B	DES_R	DES_G
1	DES_B					
2	MED_B	1.000000	-0.750536	-0.622941	-0.168126	-0.744567
						-0.627573
						-0.171257
3	VAR_R	-0.750536	1.000000	0.790079	0.504918	0.998411
						0.799513
						0.507962
4	VAR_G	-0.622941	0.790079	1.000000	-0.006827	0.787227
						0.999196
						-0.002490
5	VAR_B	-0.168126	0.504918	-0.006827	1.000000	0.508740
						0.007373
						0.994524
6	DES_R	-0.744567	0.998411	0.787227	0.508740	1.000000
						0.798147
						0.512913
7	DES_G	-0.627573	0.799513	0.999196	0.007373	0.798147
						1.000000
						0.011292
8	DES_B	-0.171257	0.507962	-0.002490	0.994524	0.512913
						0.011292
						1.000000

Algoritmo 37: Matriz de correlación de los datos de promedio, varianza y desviación estándar de las imágenes RGB.

La matriz de correlación indica una relación baja en la mayoría de los datos, esto beneficia el aprendizaje automático, pues de esta manera puede diferenciar con mayor facilidad una clase de otra.

Para este conjunto de entrenamiento se decide utilizar los modelos de análisis de discriminante lineal y clasificador de árbol de decisión.

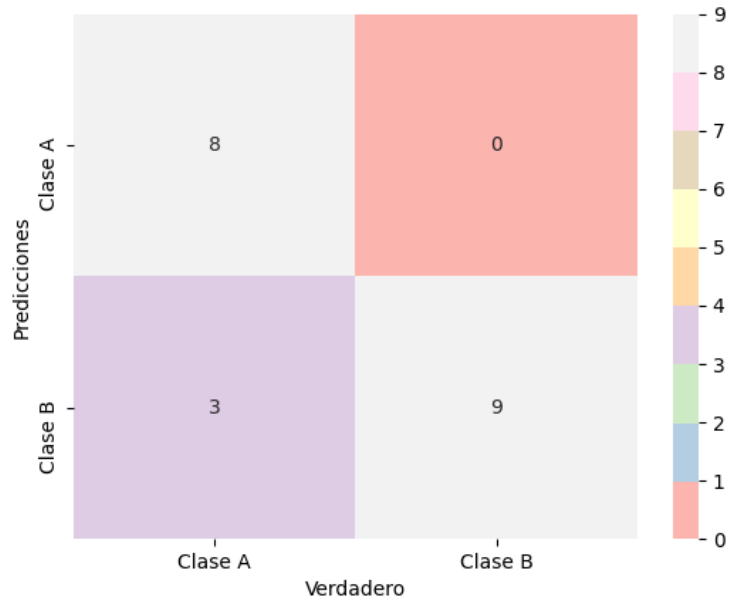


Figura 4.59: Matriz de confusión del algoritmo LDA del promedio, varianza y desviación estándar de imágenes RGB.

	0.46 __y__0 .76 (Predicción)	0.80__y__1 .20 (Predicción)
0.46 __y__0 .76 (Real)	8	0
0.80__y__1 .20(Real)	3	9

Tabla 4.26: Matriz de confusión del algoritmo LDA para datos promedio, varianza y desviación estándar de imágenes RGB .

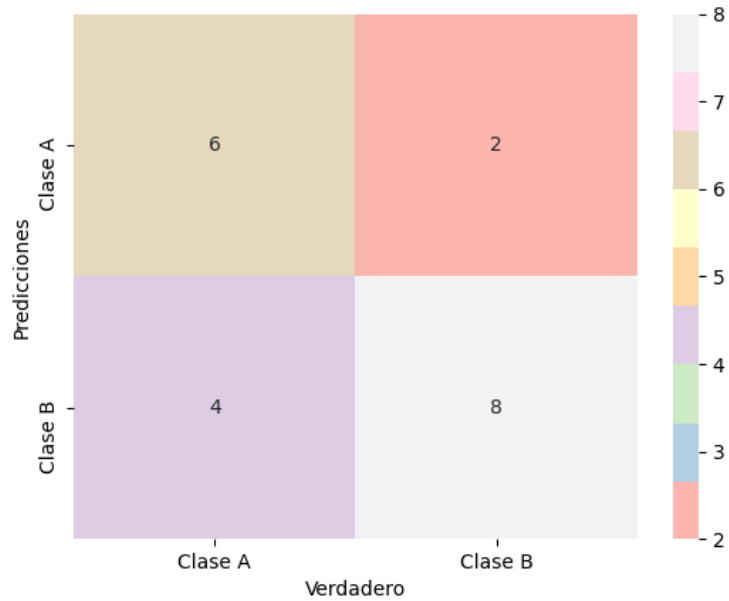


Figura 4.60: Matriz de confusión del algoritmo DTC del promedio, varianza y desviación estándar de imágenes RGB.

	0.46 __y__0 .76 (Predicción)	0.80__y__1 .20 (Predicción)
0.46 __y__0 .76 (Real)	6	2
0.80__y__1 .20(Real)	4	8

Tabla 4.27: Matriz de confusión del algoritmo DTC para datos promedio, varianza y desviación estándar de imágenes RGB.

	LDA	DTC
Exactitud	0.85 %	0.7 %
Tasa de error	0.15 %	0.3 %
Sensibilidad	1 %	0.75 %
Especificidad	0.75 %	0.66 %
Precisión	0.72 %	0.6 %
Valor de predicción negativo	1 %	0.8 %

Tabla 4.28: Porcentajes de comportamiento de los modelos LDA y DTC para los datos de promedio, varianza y desviación estándar de imágenes RGB.

Nombre	Predicción esperada		Predicción LDA	
	0.80__y__1.20	0.46__y__0.76	0.80__y__1.20	0.46__y__0.76
S3HPA_002711		x		x
S3HPA_000655		x		x
S3HPA_000241		x		x
S3HPA_001356		x		x
S3HPA_003559		x		x
S3HPA_003336		x		x
S3HPA_004129		x		x
S3HPA_004026		x		x
S3HPA_004316		x		x
S3HPA_001308		x		x
S3HPV_000516	x		x	
S3HPV_000350	x			x
S3HPV_000535	x		x	
S3HPV_000634	x			x
S3HPV_000730	x		x	
S3HPV_000812	x		x	
S3HPV_000444	x		x	
S3HPV_000222	x		x	
S3HPV_001415	x		x	

Tabla 4.29: Resultados obtenidos de la clasificación de datos nuevos de promedio, varianza y desviación estándar de imágenes RGB utilizando el modelo LDA.

Nombre	Predicción esperada		Predicción DTC	
	0.80--y--1.20	0.46--y--0.76	0.80--y--1.20	0.46--y--0.76
S3HPA_002711		x		x
S3HPA_000655		x		x
S3HPA_000241		x		x
S3HPA_001356		x		x
S3HPA_003559		x		x
S3HPA_003336		x		x
S3HPA_004129		x		x
S3HPA_004026		x		x
S3HPA_004316		x		x
S3HPA_001308		x		x
S3HPV_000516	x			x
S3HPV_000350	x			x
S3HPV_000535	x		x	
S3HPV_000634	x			x
S3HPV_000730	x		x	
S3HPV_000812	x		x	
S3HPV_000444	x		x	
S3HPV_000222	x		x	
S3HPV_001415	x		x	

Tabla 4.30: Resultados obtenidos de la clasificación de datos nuevos de promedio, varianza y desviación estándar de imágenes RGB utilizando el modelo DTC.

Como se evidenció, se utilizaron básicamente tres modelos de aprendizaje. Para las características de las imágenes RGB se utilizaron los modelos de LDA y DTC, para las características de los índices de vegetación en las imágenes multiespectrales se utilizaron los modelos de KNC y LDA, finalmente para las características de textura en imágenes multiespectrales se utilizaron los modelos de KNC, LDA y DTC.

Durante la visualización de los datos se pudo notar que presentaban una correlación alta, es decir, las clases en que debían ser clasificadas las muestras no se diferenciaban fuertemente, trayendo como consecuencia que los porcentajes de exactitud no fueran lo suficientemente confiables. A pesar de esto, se pudo observar que los datos con los cuales se realizaron las pruebas con cada vector de características, fueron clasificados correctamente en su mayoría, pues los modelos utilizados se basan en diferentes condiciones que permiten llegar a la clasificación más acertada.

El modelo de discriminante lineal (LDA) al estar basado en el teorema de Bayes, clasificaba las muestras en función de cada una de sus características o predictores, determinando la probabilidad de que una muestra pertenezca a una de las clases (0.46_y_0.76 y 0.80_y_1.20) observando cada uno de sus predictores.

Por su parte el modelo de árbol de decisión (DTC) divide los datos de acuerdo a condiciones generadas, de tal manera que la muestra que se desea clasificar debe ir respondiendo a dichas condiciones iniciando desde un nodo raíz, hasta llegar a la respuesta. El modelo de K-vecinos más cercanos (KCN), lo que hace es asignarle la clase a la que pertenece el punto más cercano al punto de prueba.

Con las características de las imágenes RGB se obtuvo una precisión de 0.85 utilizando LDA y de 0.7 utilizando DTC, con las características de los índices de vegetación se obtuvo una precisión de 0.75 con KCN y 0.65 con LDA, finalmente con las características de textura se obtuvo una exactitud de 0.7 con KCN, 0.65 con LDA y 0.6 utilizando el modelo DTC. A pesar de que el modelo DTC con características de textura tuvo un porcentaje de exactitud relativamente bajo, en la tabla 4.25 se puede evidenciar que al realizar la validación con datos que no habían sido vistos por el modelo antes, realiza la clasificación correcta de la mayoría de ellos.

Capítulo 5

Plan de trabajo y análisis de resultados

En este capítulo se analizarán los resultados obtenidos durante el proceso realizado para la estimación del nitrógeno.

Objetivo 1. Estudiar diferentes métodos de aprendizaje automático que permitan estimar el estado de nitrógeno en cultivos de gulupa.

Para cumplir este objetivo se realizaron consultas acerca del estado del arte que existe en cuanto a la estimación del nitrógeno, no se encontró específicamente de la gulupa pero si, de otras plantas.

Uno de los métodos utilizados para la estimación del nitrógeno en hojas, fue el diseño de una caja oscura, con el fin de tener mayor control de luminosidad sin tener alteraciones del ambiente externo a la caja, como lo hicieron en los artículos [7] y [8], en los dos artículos realizaron en diseño de una caja oscura donde hicieron la captura de las imágenes.

La mayoría de trabajos encontrados siguieron el método de recolección de muestras, captura de imágenes, extracción de características e implementación del aprendizaje automático.

Objetivo 2. Diseñar un método que permita estimar la deficiencia de nitrógeno basada en aprendizaje automático.

La metodología de este proyecto consistió en la toma de muestras, captura de imágenes, realización de pruebas químicas, extracción de características e implementación del algoritmo de aprendizaje.

Toma de muestras

La toma de muestras fue realizada en la finca La Pradera ubicada en la vereda Lázaro Fonte en el municipio de Pasca Cundinamarca. La toma de muestras fue planeada previamente a las visitas al cultivo, con el fin de ahorrar tiempo y además de preparar lo necesario antes de la llegada a la finca.

Captura de imágenes

Para la captura de las imágenes se realizó el diseño de una caja oscura en SolidWorks, la finalidad era adquirir las imágenes teniendo el mayor control posible de la luminosidad, por tal razón la caja cuenta con una tapa que contiene en su interior un soporte para la cámara y otro para el banco de poder, así se cerraba la caja completamente al momento de realizar la captura. Además cuenta con dos bombillos halógenos que brindaban la iluminación adecuada para que las imágenes multiespectrales fueran capturadas.

Pruebas químicas

Las muestras se llevaron al laboratorio después de que se realizó la captura de las imágenes, se llevaron en bolsas de papel para evitar la descomposición de las hojas. Estas pruebas fueron realizadas para darle una etiqueta a las muestras dependiendo de los resultados obtenidos en el análisis químico.

Extracción de características

Las características extraídas de las imágenes multispectrales fueron los índices de vegetación NDVI, GNDVI y OSAVI, y el promedio y varianza de la textura de las imágenes de los canales rojo, verde, infrarrojo cercano y borde rojo.

Las características utilizadas para la implementación de los modelos de aprendizaje en las imágenes RGB fueron el promedio, la varianza y la desviación estándar de cada canal. Para la extracción de características de las imágenes RGB se implementó el código realizado por el ingeniero Santiago Trujillo en su tesis [1].

Esta extracción de características fue realizada para utilizar dichos conjuntos de datos en el aprendizaje automático.

Aprendizaje automático

Después de tener los conjuntos de datos de la extracción de características, se procedió a la implementación de los algoritmos de aprendizaje que brinda la biblioteca *sklearn*.

Objetivo 3. Implementar el método de aprendizaje automático que permita estimar la deficiencia de nitrógeno a partir de imágenes multispectrales

La metodología mencionada en el objetivo 2 fue implementada para llevar a cabo la estimación del estado de nitrógeno en las hojas de gulupa.

La implementación de la caja oscura para la captura de las imágenes de la gulupa, se realizó después de llevar a cabo diferentes pruebas, esto para asegurarse de que la iluminación permitiera la visualización de las imágenes multispectrales y de esta manera lograr la extracción de las características.

En cuanto a la implementación de los modelos de aprendizaje, se realizaron diferentes pruebas para las características de las imágenes RGB y multispectrales, evaluando su comportamiento por medio de la matriz de confusión.

Las predicciones realizadas por los modelos entrenados con las características de las imágenes multiespectrales presentaron mejores resultados utilizando los valores del promedio y la varianza de la textura, lo cual se puede evidenciar en la tabla 4.25, donde se muestran los resultados de la predicción utilizando el modelo de clasificación de árbol de decisión.

Con las características de las imágenes RGB que se obtuvo mejores resultados, fue con el promedio, varianza y desviación estándar de los canales RGB, pues al utilizar parámetros como el largo, el ancho y la segmentación de la hoja afectada por el cambio de color en píxeles, las predicciones fueron en su mayoría erróneas, debido a que el resultado de las pruebas químicas arrojaron que el porcentaje de nitrógeno de las muestras no estaban directamente relacionados con el tamaño y color de las hojas.

Objetivo 4. Validar el método diseñado mediante la comparación de los resultados entregados por un método convencional

La validación fue realizada con base en los resultados obtenidos en las pruebas químicas, los cuales eran la predicción esperada por los modelos. Los resultados de las predicciones utilizando los índices de vegetación se pueden observar en las tablas 4.17, 4.18, estas muestran los resultados de los modelos KNC y LDA, se observa que presentan resultados similares, de las 20 muestras KNC clasificó correctamente 15 y LDA 16; los resultados de las predicciones utilizando las características de textura se pueden ver en las tablas 4.23, 4.24 y 4.25, en donde el modelo DTC tuvo la mejor predicción, pues de las 20 muestras clasificó correctamente 19; los resultados de las predicciones utilizando características de las imágenes RGB se observan en las tablas 4.29 y 4.30.

A pesar de que en el momento de la visualización de los datos se evidenció que las clases no se diferenciaban de la mejor manera, en la validación realizada con cada grupo de predictores de imágenes RGB y multiespectrales, se observó que los modelos clasificaban correctamente la mayoría de los datos, pues el modelo LDA se basaba en el teorema de Bayes, es decir, en la probabilidad de que cada muestra perteneciera a determinada clase teniendo en cuenta cada característica, en cuento

al modelo DTC, dividía los datos y generaba condiciones a partir de ellos, logrando clasificar la muestra en una de las clases establecidas, de igual manera teniendo en cuenta sus predictores.

Actividad	Tiempo de ejecución
Realizar el estado del arte con artículos que relacionen el aprendizaje automático y procesamiento de imágenes multiespectrales para la estimación del nitrógeno presente en cultivos.	10 semanas
Considerar métodos de aprendizaje automático que permitan estimar el estado de nitrógeno en la planta de gulupa	
Identificar el mejor método de aprendizaje automático que permita estimar el estado de nitrógeno en la planta de gulupa.	8 semanas
Implementar el método de estimación del estado de nitrógeno basado en de aprendizaje automático.	6 semanas
Efectuar ajustes en el modelo implementado en caso de ser necesario.	3 semanas
Evaluar el desempeño de la herramienta en comparación a la estimación del nitrógeno de método tradicional.	

Tabla 5.1: Plan de trabajo.

Conclusiones

- Se consultaron diferentes artículos relacionados con la estimación del nitrógeno en diferentes tipos de plantas por medio de imágenes adquiridas de sus hojas. Esto permitió generar un plan de trabajo que estableció las actividades realizadas para lograr los objetivos propuestos.
- La metodología implementada consistió en la recolección de muestras que se realizó durante las visitas al cultivo, la captura de imágenes a las hojas que

fueron recolectadas durante las visitas, pruebas químicas realizadas en un laboratorio químico luego de haber capturado las imágenes, extracción de características y la implementación de los modelos de aprendizaje.

- Debido a que no se contó con un cultivo de gulupa controlado, las muestras recolectadas se dividieron en grupos dependiendo de las similitudes de tamaño y color, y teniendo en cuenta los resultados de las pruebas químicas se le asignó a cada muestra un rango de porcentaje de nitrógeno (0.46_y_0.76 o 0.80_y_1.20), aunque esto no significaba que todas las hojas pertenecientes a un determinado grupo presentaran en general el mismo porcentaje de nitrógeno, haciendo que los datos presentaran alta variabilidad, siendo esto un motivo por el cual las clases no eran diferenciadas en su totalidad. En cuanto a la captura de imágenes, el diseño de la caja oscura permitió que las imágenes fueran capturadas con un control de luminosidad, además de facilitar este procedimiento, pues solo se tenía que colocar la hoja dentro y cerrar la caja sin realizar otro ajuste. Además permitía tener la cámara en una misma posición siempre.
- Las predicciones realizadas por los modelos de aprendizaje en las imágenes multiespectrales, presentaron mejores resultados utilizando los predictores de promedio y varianza de la textura. En cuanto a las imágenes RGB se evidenció que se presentaron mejores resultados usando las características de promedio, varianza y desviación estándar de los canales RGB que utilizando características de tamaño y color de la hoja, pues al utilizar estos predictores las predicciones fueron erróneas en su mayoría, esto se puede asociar a que los porcentajes de nitrógeno de resultados de las pruebas químicas, no estaban estrechamente relacionados con el tamaño y el color de las hojas. En general los resultados de las predicciones hechas por los modelos superaron en su mayoría el 70% de exactitud, realizando predicciones correctas en un gran porcentaje de las muestras en la validación, lo cual se observó realizando la comparación de la predicción esperada con la predicción realizada por los modelos de aprendizaje automático, tanto en las imágenes RGB con las características de promedio, varianza y desviación estándar, como en las imágenes multiespectrales con las características de los índices de vegetación, promedio y varianza de la textura.

Capítulo 6

Presupuesto

Para el desarrollo del presente proyecto se utilizó una cámara multiespectral Parrot Sequoia, para alimentarla se utilizó una Powerbank. Además de esto se realizaron las pruebas químicas a las muestras recogidas de los surcos, también se diseñó y fabricó una caja oscura en MDF, la cual incluía dos lámparas halógenas.

Elemento	Costo
Camara Parrot Sequoia	\$15'217.222
Pruebas de laboratorio	\$571.200
Powerbank RPL-58	\$95.000
Cuerpo de la caja	\$40.000
Soportes de la cámara y el banco de poner	\$10.000
Lámpara halógenas	\$7.000
Soportes para lámparas halógenas	\$5.000
Horas invertidas por la estudiante	\$939.800
Horas invertidas por el director del proyecto	\$626.533
Visitas al cultivo	\$100.000
TOTAL	\$17'611.755

Tabla 6.1: Costos totales del proyecto.

Bibliografía

- [1] Santiago Alejandro Trujillo Fandiño. *Implementación de un sistema capaz de facilitar la identificación del cambio de color en la hoja de gulupa conforme a la presencia o ausencia de nitrógeno mediante el procesamiento de imágenes RGB*. 2020.
- [2] Faryd Alejandro Peñuela González. *Implementación de un sistema capaz de calcular el área foliar de una planta de gulupa, a partir de imágenes que representen dos dimensiones de la planta, mediante técnicas de procesamiento de imágenes*. 2020.
- [3] Natalia Fonseca Trujillo y col. «Caracterización molecular de materiales cultivados de gulupa (*Passiflora edulis* f. *edulis*)». En: *Universitas Scientiarum* 14.2-3 (2009), págs. 135-140.
- [4] Agronegocios. *El marañón y la gulupa, entre los cultivos que se proyectan con más potencial*. <https://www.agronegocios.co/agricultura/el-maranon-y-la-gulupa-entre-los-cultivos-que-se-proyectan-con-mas-potencial-2991177>.
- [5] Remtavares. *La contaminación de las aguas subterráneas por nitratos*. <https://www.madrimasd.org/blogs/remtavares/2006/07/12/35033>.
- [6] Bangyong Sun y col. «Design of four-band multispectral imaging system with one single-sensor». En: *Future Generation Computer Systems* 86 (2018), 670-679.
- [7] Pedro Caldentey Aventián. «Utilización de sensores multispectrales e hiperespectrales embarcados en RPAS con el objetivo de dosificar abonos en cultivos.» B.S. thesis. Universitat Politècnica de Catalunya, 2017.

- [8] Zhulin Chen y Xuefeng Wang. «Model for estimation of total nitrogen content in sandalwood leaves based on nonlinear mixed effects and dummy variables using multispectral images». En: *Chemometrics and Intelligent Laboratory Systems* 195 (2019), pág. 103874.
- [9] Wikipedia. *Propagación hacia atrás* — *Wikipedia, La enciclopedia libre*. https://es.wikipedia.org/w/index.php?title=Propagaci%C3%B3n_hacia_atr%C3%A1s&oldid=130466308. [Internet; descargado 17-octubre-2020]. 2020.
- [10] Chen Lisu, Sun Yuanyuan y Wang Ke. «Rapid diagnosis of nitrogen nutrition status in rice based on static scanning and extraction of leaf and sheath characteristics». En: *International Journal of Agricultural and Biological Engineering* 10.3 (2017), págs. 158-164.
- [11] Luis Carlos Leiva. *Manejo de problemas fitosanitarios del cultivo de gulupa*. Produmedios, 2011.
- [12] John Ocampo Pérez y Kris Wyckhuys. *Tecnología para el cultivo de Gulupa en Colombia*. Dirección de Publicaciones UJTL, 2012.
- [13] Jayson Heriberto Alzate Calixto y Raul Andres Melo Duque. *Evaluación de dos fuentes nutricionales en un cultivo de gulupa (Passiflora edulis. Sims.) bajo las condiciones del municipio de Pacho Cundinamarca*. 2019.
- [14] Lucia Constanza Corrales y col. «Bacterias anaerobias: procesos que realizan y contribuyen a la sostenibilidad de la vida en el planeta». En: (2015).
- [15] Rodolfo Bongiovanni y col. *Agricultura de precisión: Integrando conocimientos para una agricultura moderna y sustentable*. 2006.
- [16] Enrique Alegre, Gonzalo Pajares y Arturo de la Escalera. *Conceptos y métodos en VISIÓN POR COMPUTADOR*. CEA, 2016.
- [17] David Saavedra Mora y col. *Manual de interpretación y aplicación de imágenes multiespectrales en cultivos de importancia agrícola en el norte del Huila*. Feb. de 2020. ISBN: 978-958-15-0521-0.
- [18] Pedro Muñoz Aguayo. «Apuntes de Teledetección: Índices de Vegetación». En: (2013).
- [19] Nandini Mehrotra y Shashank Srinivasan. *Análisis de imágenes de satélites y drones utilizando índices de vegetación*. <https://www.techforwildlife.com/blog/\tag/drone>.

- [20] José Ramón Mejiéa Vilet. «Procesamiento digital de Imágenes». En: *Facultad de Ingeniería UASLP, Documento PDF, disponible en la página*http://read.pudn.com/downloads159/ebook/711796/Procesamiento_Digital_de_Imagenes.pdf[Citado 22 de septiembre de 2012] (2005).
- [21] *Espectro Electromagnético*. <https://sites.google.com/site/tecnologiascomunicacionutiel/home/espectro-electromagnetico>.
- [22] Andreas C Müller, Sarah Guido y col. *Introduction to machine learning with Python: a guide for data scientists*. "O'Reilly Media, Inc.", 2016.
- [23] Joaquín Amat Rodrigo. *Análisis discriminante lineal y análisis discriminante cuadrático*. https://www.cienciadedatos.net/documentos/28_linear_discriminant_analysis_lda_y_quadratic_discriminant_analysis_qda#An%C3%A1lisis_discriminante_lineal.
- [24] Carlos Zelada. *Evaluación de modelos de clasificación*. <https://rpubs.com/chzelada/275494>.
- [25] manualslib the ultimate manuals library. *Presentación De Sequoia; Español - Parrot Sequoia User Manual*. <https://www.manualslib.com/manual/1228008/Parrot-Sequoia.html?page=38manual>.
- [26] tecniTop. *Parrot Sequoia+ con Pix4D*. <https://tecniTop.com/es/parrot-sequoia-con-pix4d/>.
- [27] Wikipedia. *OpenCV — Wikipedia, La enciclopedia libre*. <https://es.wikipedia.org/w/index.php?title=OpenCV&oldid=128322880>. [Internet; descargado 17-octubre-2020]. 2020.
- [28] Wikipedia. *Scikit-learn — Wikipedia, La enciclopedia libre*. <https://es.wikipedia.org/w/index.php?title=Scikit-learn&oldid=130746122>. [Internet; descargado 10-octubre-2020]. 2020.
- [29] Universidad de Alcalá. *Scikit-Learn, herramienta básica para el Data Science en Python*. <https://www.master-data-scientist.com/scikit-learn-data-science/>.
- [30] SOLIDBI. *SOLIDWORKS. Qué es y para qué sirve*. <https://www.the-imagen.com/pixeles-por-pulgada-de-verdad-es-tan-complicado/>.