



UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERÍA
CARRERA DE INGENIERÍA EN TELECOMUNICACIONES

**Implementación de una red de sensores inteligentes basada en
Análisis de Datos y Machine Learning para la detección temprana
y prevención de deslizamientos de tierra.**

Trabajo de Titulación para optar al título de:
INGENIERO EN TELECOMUNICACIONES

Autor:

Medina Sánchez Kevin Valentín

Tutor:

PhD. Leonardo Fabian Rentería Bustamante

Riobamba, Ecuador. 2025

DECLARATORIA DE AUTORÍA

Yo, **Medina Sánchez Kevin Valentín**, con cédula de ciudadanía **0605209709**, autor del trabajo de investigación titulado: **“IMPLEMENTACIÓN DE UNA RED DE SENSORES INTELIGENTES BASADA EN ANÁLISIS DE DATOS Y MACHINE LEARNING PARA LA DETECCIÓN TEMPRANA Y PREVENCIÓN DE DESLIZAMIENTOS DE TIERRA”**, certifico que la producción, ideas, opiniones, criterios, contenidos y conclusiones expuestas son de mí exclusiva responsabilidad.

Asimismo, cedo a la Universidad Nacional de Chimborazo, en forma no exclusiva, los derechos para su uso, comunicación pública, distribución, divulgación y/o reproducción total o parcial, por medio físico o digital; en esta cesión se entiende que el cesionario no podrá obtener beneficios económicos. La posible reclamación de terceros respecto de los derechos de autor (a) de la obra referida, será de mi entera responsabilidad; librando a la Universidad Nacional de Chimborazo de posibles obligaciones.

En Riobamba, 27 de junio del 2025.



Medina Sánchez Kevin Valentín

C.I: 0605209709



DICTAMEN FAVORABLE DEL PROFESOR TUTOR

Quien suscribe, **Leonardo Fabián Rentería Bustamante** catedrático adscrito a la Facultad de Ingeniería, por medio del presente documento certifico haber asesorado y revisado el desarrollo del trabajo de investigación **"Implementación de una red de sensores inteligentes basada en Análisis de Datos y Machine Learning para la detección temprana y prevención de deslizamientos de tierra"**, bajo la autoría de **Kevin Valentín Medina Sánchez**; por lo que se autoriza ejecutar los trámites legales para su sustentación.

Es todo cuanto informar en honor a la verdad; en Riobamba, a los 16 días del mes de junio de 2025.

PhD. Leonardo Fabián Rentería Bustamante.
C.I:1104064132

CERTIFICADO DE LOS MIEMBROS DEL TRIBUNAL

Quienes suscribimos, catedráticos designados Miembros del Tribunal de Grado para la evaluación del trabajo de investigación **IMPLEMENTACIÓN DE UNA RED DE SENSORES INTELIGENTES BASADA EN ANÁLISIS DE DATOS Y MACHINE LEARNING PARA LA DETECCIÓN TEMPRANA Y PREVENCIÓN DE DESLIZAMIENTOS DE TIERRA**, con cédula de identidad número **0605209709**, bajo la tutoría de PhD. Leonardo Fabián Rentería Bustamante; certificamos que recomendamos la **APROBACIÓN** de este con fines de titulación. Previamente se ha evaluado el trabajo de investigación y escuchada la sustentación por parte de su autor; no teniendo más nada que observar.

De conformidad a la normativa aplicable firmamos, en Riobamba 27 de Junio de 2025.

Dr. Antonio Meneses, PhD.
PRESIDENTE DEL TRIBUNAL DE GRADO



Ing. Juan Carlos Cepeda, PhD.
MIEMBRO DEL TRIBUNAL DE GRADO



Ing. José Jinez
MIEMBRO DEL TRIBUNAL DE GRADO





CERTIFICACIÓN

Que, **MEDINA SÁNCHEZ KEVIN VALENTÍN** con CC: **0605209709**, estudiante de la Carrera de **TELECOMUNICACIONES**, Facultad de **INGENIERÍA**; ha trabajado bajo mi tutoría el trabajo de investigación titulado **"IMPLEMENTACIÓN DE UNA RED DE SENSORES INTELIGENTES BASADA EN ANÁLISIS DE DATOS Y MACHINE LEARNING PARA LA DETECCIÓN TEMPRANA Y PREVENCIÓN DE DESLIZAMIENTOS DE TIERRA"**, cumple con el 3 %, de acuerdo al reporte del sistema Anti plagio **COMPILATIO**, porcentaje aceptado de acuerdo a la reglamentación institucional, por consiguiente autorizo continuar con el proceso.

Riobamba, 25 de 06 de 2025



Firmado electrónicamente por:
**LEONARDO FABIAN
RENTERIA BUSTAMANTE**
Validar únicamente con FirmaEC

PhD. Leonardo Rentería
TUTOR(A) TRABAJO DE INVESTIGACIÓN

DEDICATORIA

Este trabajo de investigación está dedicado a toda mi familia por su incansable apoyo.

AGRADECIMIENTO

Quiero expresar mi agradecimiento a mi tutor PhD. Leonardo Rentería por su paciencia y supervisión en el desarrollo de mi trabajo de titulación.

Quiero expresar mi agradecimiento a toda mi familia en especial a mis tías Rosa y Lupe Medina, y a mi abuela Tannia Medina quienes me acompañaron durante mi formación estudiantil, brindándome su apoyo incondicional.

También quiero expresar mi agradecimiento a mis amigos, por su ayuda, inspiración y compañía a lo largo de todos estos años de formación académica.

ÍNDICE GENERAL

DECLARATORIA DE AUTORÍA

DICTAMEN FAVORABLE DEL PROFESOR TUTOR

CERTIFICADO DE LOS MIEMBROS DEL TRIBUNAL

CERTIFICADO ANTIPLAGIO

DEDICATORIA

AGRADECIMIENTO

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

ABSTRACT

CAPÍTULO I	16
1.1 Introducción.....	16
1.2 Antecedentes.....	17
1.3 Planteamiento del problema.	18
1.4 Justificación.....	19
1.5 Objetivos.....	20
1.5.1 Objetivo General.....	20
1.5.2 Objetivos específicos	20
CAPÍTULO II.....	21
2.1 Estado del arte.	21
2.2 Fundamentación teórica.....	24
2.2.1 Redes de sensores	24
2.2.2 Características de las WSN.....	25
2.2.3 Topología en Malla.....	26
2.2.4 Protocolo de comunicación LoRa.....	26
2.2.5 Comparación con otros protocolos	27
2.2.6 Internet de las cosas (IoT)	27

2.2.6.1	Ventajas.....	27
2.2.7	Machine Learning.....	28
2.2.8	Tipos de ML	29
2.2.8.1	Aprendizaje supervisado.....	29
2.2.8.2	Aprendizaje no supervisado.....	29
2.2.9	Algoritmos de ML	30
2.2.9.1	Regresión logística	30
2.2.9.2	SVM	31
2.2.9.3	Random Forest.....	32
2.2.9.4	Redes neuronales (MLP)	34
2.2.10	Métricas de validación.....	35
2.2.10.1	Matriz de confusión.....	35
2.2.10.2	Precisión	36
2.2.10.3	Sensibilidad	36
2.2.10.4	F1-Score	36
2.2.10.5	Área bajo la curva ROC (AUC-ROC).....	37
2.2.11	Deslizamientos de tierra	37
2.2.11.1	Definición	37
2.2.11.2	Clasificación	37
2.2.12	Factores de riesgo	39
2.2.13	Factores condicionantes.....	39
CAPÍTULO III.		41
3.1	Metodología.....	41
3.1.1	Tipo de investigación	41
3.2	Método de investigación.....	41
3.2.1	Método cuantitativo.....	41
3.2.2	Método Experimental	41
3.3	Técnica de recolección de datos	42
3.3.1	Estudio bibliográfico	42
3.3.2	Recolección de datos con sensores.....	42
3.4	Población de estudio.....	42
3.4.1	Población	42
3.5	Operacionalización de las variables	42

3.6	Procedimiento.....	43
3.7	Fase 1: Recolección de datos históricos	45
3.7.1	Recolección de datos históricos.....	45
3.7.2	Inventario de deslizamiento.....	45
3.7.3	Selección de factores condicionantes	47
3.7.4	Relación de frecuencias	48
3.7.5	Normalización de los datos	49
3.7.6	Subdivisión de los datos históricos.....	49
3.8	Fase 2: Diseño del prototipo y elección de la tecnología de comunicación	50
3.8.1	Arquitectura del sistema	50
3.8.2	Sensores de monitoreo.....	50
3.8.3	Módulo ESP32 LoRa V3.2.....	52
3.8.4	Raspberry pi 5.....	53
3.8.5	Enlace Inalámbrico	54
3.8.6	Diseño del prototipo de red de sensores	55
3.9	Fase 3: Implementación del prototipo y evaluación en ambiente controlado	56
3.9.1	Diseño del ambiente controlado	56
3.9.2	Implementación del prototipo en el ambiente controlado	58
CAPÍTULO IV.....		64
4.1	Resultados y discusión	64
4.2	Fase 1: Recolección de datos históricos y selección de algoritmos	64
4.2.1	Preprocesamiento de los datos.....	64
4.2.2	Ponderación de variables cualitativas.....	65
4.2.3	Análisis del sistema predictivo con regresión logística.....	67
4.2.4	Análisis del sistema predictivo con SVM	69
4.2.5	Análisis del sistema predictivo con Random Forest.....	71
4.2.6	Análisis del sistema predictivo con redes neuronales (MLP)	73
4.3	Fase 2: Diseño del prototipo y elección de la tecnología de comunicación	74
4.3.1	Desempeño del enlace LoRa	76
4.4	Fase 3: Implementación del prototipo y evaluación en ambiente controlado	78
4.4.1	Análisis entre modelos	78
CAPÍTULO V.....		82
5.1	Conclusiones y recomendaciones.....	82

5.1.1	Conclusiones.....	82
5.1.2	Recomendaciones	83
BIBLIOGRAFÍA		84
ANEXOS		88

ÍNDICE DE TABLAS.

Tabla 1. Comparación de protocolos de comunicación [9], [21], [25], [26].	27
Tabla 2. Matriz de confusión.	36
Tabla 3. Tipos de deslizamientos de tierra [37].	38
Tabla 4. Operacionalización de las variables.	43
Tabla 5. Tabla de frecuencia de factores condicionantes.	48
Tabla 6. Características de los sensores de monitoreo [43]–[47].	50
Tabla 7. Características del módulo [48].	52
Tabla 8. Características de la Raspberry Pi 5 [49].	54
Tabla 9. Configuración del SX1262.	54
Tabla 10. Calibración del sensor de precipitación.	58
Tabla 11. Comparación de temperatura.	59
Tabla 12. Comparación de humedad.	59
Tabla 13. Comparación del índice UV.	61
Tabla 14. Comparación de la humedad del suelo.	62
Tabla 15. Variables de entrenamiento.	64
Tabla 16. Clasificación de land cover de acuerdo con el satélite MODIS.[50].	65
Tabla 17. Clasificación según el triángulo de USDA [51].	66
Tabla 18. Análisis del algoritmo Regresión Logística.	67
Tabla 19. Análisis del algoritmo SVM.	69
Tabla 20. Análisis del algoritmo Random Forest.	71
Tabla 21. Análisis de algoritmo de Redes Neuronales (MLP).	73
Tabla 22. Desempeño del enlace LoRa.	76
Tabla 23. Comparación de RSSI teórico y medido.	77
Tabla 24. Comparación entre modelos.	78
Tabla 25. Prueba de normalidad Shapiro-Wilk.	79
Tabla 26. Aciertos y fallos.	80
Tabla 27. Rango medio obtenido al hacer el estudio de Friedman.	81
Tabla 28. Test de Nemenyi (post-hoc).	81

ÍNDICE DE FIGURAS

Figura 1. Esquema de una red de sensores [9].	24
Figura 2. Topología en Malla [9].	26
Figura 3. Regresión logística	30
Figura 4. Ajustes de SVM [8].	32
Figura 5. Árbol de decisión y grupo de árboles de decisión(Random Forest) [8].	33
Figura 6. Esquema de una red neuronal simple con una sola capa oculta [8].	35
Figura 7. Procedimiento del trabajo de investigación.	44
Figura 8. Porcentaje de los tipos de deslizamientos del inventario.	46
Figura 9. Porcentaje de severidad de los deslizamientos.	47
Figura 10. Porcentaje del desencadenante de deslizamientos.	47
Figura 11. Sensor HW-028.	52
Figura 12. Sensor DHT22.	52
Figura 13. Sensor GUVA-S12SD.	52
Figura 14. Sensor FC-28.	52
Figura 15. Sensor GY-LSM6D3.	52
Figura 16. Microcontrolador ESP32 con módulo LoRa.	53
Figura 17. Raspberry Pi 5.	54
Figura 18. Diagrama del diseño de un nodo de recolección.	56
Figura 19. Diagrama de recolección de transmisión y recolección de datos.	56
Figura 20. Vista previa del ambiente controlado.	57
Figura 21. Curva de calibración del sensor HW-028.	58
Figura 22. Calibración de temperatura.	60
Figura 23. Calibración de humedad.	60
Figura 24. Calibración del sensor GUVA-S12SD.	61
Figura 25. Curva de calibración del sensor FC-28.	62
Figura 26. Ventana de sistema de predicción.	63
Figura 27. Matriz de confusión del algoritmo Regresión Logística.	68
Figura 28. Curva de precisión-Recall y ROC para regresión logística.	68
Figura 29. Matriz de confusión de SVM.	70
Figura 30. Curva de Precisión-Recall y ROC para SVM.	70
Figura 31. Matriz de confusión de Random Forest.	72
Figura 32. Curva Precisión- Recall y ROC de Random Forest.	72
Figura 33. Matriz de confusión de Redes Neuronales(MLP).	74
Figura 34. Curva Precisión-Recall y ROC de Redes Neuronales(MLP).	74
Figura 35. Diseño de la primera PCB,	75
Figura 36. Diseño de la segunda PCB.	75
Figura 37. Tasa de éxito vs distancia.	76
Figura 38. RSSI vs distancia.	77
Figura 39. Comparación entre RSSI teórico y medido.	77
Figura 40. Porcentaje de aciertos por algoritmo.	80

RESUMEN

En esta investigación se desarrolló una red de sensores inteligente para predecir deslizamientos de tierra mediante, integrando tecnologías IoT y algoritmos machine learning, con el objetivo de mitigar riesgos geológicos. El sistema usa nodos basados en placas ESP32 LoRa V3.2 programadas en MicroPython, que monitorean variables ambientales en tiempo real y transmiten datos vía LoRa a 915 MHz. Se compararon cuatro algoritmos: regresión logística (máxima verosimilitud), SVM (optimización convexa con kernels), Random Forest (bagging) y una red neuronal MLP (backpropagation) entrenados con datos históricos de entidades oficiales y evaluados en un entorno controlado que replicó condiciones de inestabilidad. Se midió el desempeño mediante precisión teórica y un índice de aciertos reales, este último calculado con pruebas físicas, el sistema logró latencias inferiores a 8 segundos y un alcance teórico de hasta 1.2 km con LoRa, destacando su escalabilidad y bajo costo. Los resultados muestran que si bien la regresión logística mostró alta precisión teórica (94%), su bajo índice de aciertos reales (57%) validó la necesidad de modelos complejos en fenómenos geológicos. El estudio concluye que la convergencia de sensores y una red neuronal MLP alcanza una precisión del 97% y minimiza los falsos negativos (9%) sobre la precisión global, debido a la capacidad del MLP para aprender representaciones jerárquicas de los datos lo posiciona como una herramienta adaptable a la variabilidad ambiental.

Palabras claves: deslizamientos de tierra, IoT, machine learning, red neuronal MLP, LoRa.

ABSTRACT

This research proposes an intelligent sensor network for landslide prediction that integrates IoT technologies with machine learning algorithms to reduce geological risks through real-time monitoring of environmental variables. The system is based on ESP32 LoRa V3.2 nodes, programmed in MicroPython, which collect and transmit data using LoRa communication at 915 MHz. To evaluate the system's predictive capabilities, four machine learning algorithms were tested: logistic regression (maximum likelihood estimation), support vector machines (SVM) with kernel-based convex optimization, random forest (ensemble bagging), and a multilayer perceptron (MLP) neural network trained via backpropagation. The models were trained on historical data obtained from official sources and validated under controlled conditions simulating unstable terrain. System performance was assessed through both theoretical accuracy and real-world hit rates, the latter validated by physical simulations. The network demonstrated latencies of less than 8 seconds and a theoretical LoRa communication range of 1.2 km, emphasizing its scalability and cost-effectiveness. Among the models, logistic regression achieved a theoretical accuracy of 94%; however, its real-world hit rate of 57% underscored the limitations of simpler models in dynamic environments. In contrast, the MLP neural network attained a theoretical accuracy of 97%, reducing false negatives to 9%, and proved to be highly adaptable to environmental variability. These results highlight the MLP's potential for reliable early warning applications in landslide-prone regions.

Keywords: landslides, IoT, machine learning, MLP neural network, LoRa.

Reviewed and improved by Jacqueline Armijos



CAPÍTULO I.

1.1 Introducción.

La seguridad y protección en zonas de montaña se han planteado como un desafío crítico en la gestión de riesgos geológicos. Entre los eventos más devastadores, los deslizamientos de tierra constituyen un desastre natural de gran alcance, frecuentemente desencadenado por lluvias intensas y causante de daños sustanciales tanto a la infraestructura como a las vidas humanas [1]–[4]. Pese a los esfuerzos por implementar métodos de monitoreo geodinámico, factores como topografía irregular y las condiciones climáticas variables han evidenciado las limitaciones de las tecnologías convencionales. En la sierra ecuatoriana la complejidad del terreno junto con pendientes pronunciadas combinada con suelos heterogéneos dificulta la detección temprana de movimientos de masa, dejando comunidades alejadas sin sistemas de alerta eficientes. En este contexto, surge la necesidad apremiante de contar con soluciones más adaptables.

A pesar de los métodos tradicionales y el avance de la tecnología en internet de las cosas (IoT) e inteligencia artificial para anticipar estos eventos, persiste la falta de un enfoque integrado que combine hardware especializado con algoritmos de Machine Learning (ML) adaptados para esta problemática. Es decir, no está claro como integrar de manera continua variables ambientales con algoritmos que aprendan con cada nuevo evento y se ajusten dinámicamente [1], [4]–[10]. Este proyecto parte de la idea de que, al combinar sensores con algoritmos de ML, es posible ofrecer una respuesta dinámica y precisa a los fenómenos geodinámicos presentes en entornos montañosos.

En este trabajo se plantea desarrollar una red de sensores basada en análisis de datos y Machine Learning para la detección temprana y prevención de deslizamientos de tierra, para esto se abordarán tres fases, la primera recolección de datos históricos de deslizamientos y programación de algoritmos de ML, segundo el diseño del prototipo de red de sensores y al final implementación en un ambiente controlado. Con este trabajo no solo se busca reducir los riesgos geológicos, sino también establecer un precedente para la aplicación de estas tecnologías en la gestión de riesgos.

1.2 Antecedentes.

En proyectos anteriores, los equipos de investigación instalaron inclinómetros, extensómetros y pluviómetros para monitorear la estabilidad del terreno y las condiciones meteorológicas locales, pero estos dispositivos no cubren amplias zonas ni garantizan una transmisión continua de datos, además algunos de dichos proyectos reportaron demoras de horas en la detección de movimientos críticos. Por ello, surgió la necesidad de fusionar información de diversas fuentes y procesarla al instante.

Con el crecimiento de nuevas tecnologías han surgido nuevas aproximaciones que permiten superar estas limitaciones, las infraestructuras de redes de sensores e internet de las cosas permiten el despliegue de diversos sensores, la gran cantidad de datos recopilados por estos sensores puede analizarse mediante algoritmos avanzados de ML para discernir patrones indicativos de posible inestabilidad y facilitar la detección temprana de precursores de deslizamientos en tiempo real permitiendo una cobertura continua y escalable de posibles áreas de riesgo [1].

En la región andina de Ecuador, estudios en la provincia de Loja revelan que la susceptibilidad a deslizamientos aumenta diez veces cerca de carreteras mal drenadas en Penipe, se constata que el 36,7 % del territorio presenta pendientes entre 35° y 45° [11]. Además, los aluviones de enero-febrero de 2022 en Quito ilustraron la urgencia de contar con alertas automáticas ante lluvias extremas.

La investigación sobre la predicción de otros desastres naturales demuestra la creciente aplicación de las redes de sensores y el ML, una tesis sobre "Evaluación del diseño de un prototipo de monitoreo Geo-IoT, aplicado a potenciales deslizamientos de tierra" [3] y otra sobre "Implementación de un Prototipo de Sistema de Alerta Temprana de Deslizamientos de Tierra" [12] abordan directamente el tema del monitoreo de deslizamientos mediante tecnología de sensores, lo que indica un creciente interés en esta área específica.

Estos proyectos suelen implicar análisis de requisitos técnicos, diseño de prototipos con diversos sensores (vibración, humedad, presión), su implementación y la realización de pruebas funcionales con el uso de la tecnología LoRa para redes de sensores aplicadas al monitoreo ambiental [9], [12].

1.3 Planteamiento del problema.

El 26 de marzo de 2023, tras varios días de lluvias intensas, un deslizamiento rotacional en Alausí (Chimborazo) sepultó más de 70 viviendas y arrastró un tramo de la Panamericana, dejando al menos 17 muertos, 72 desaparecidos y 163 casas afectadas según la Secretaría de Riesgos de Ecuador [13]. Medios como Reuters confirmaron 11 víctimas, 23 heridos y casi 50 desaparecidos inicialmente, mientras que El País informó 16 fallecidos y más de 500 damnificados. Esa noche, las grietas que habían aparecido en diciembre de 2022 no fueron registradas por la red de inclinómetros instalados puntualmente en la ladera.

Pese a los avances en instrumentación y monitoreo de anomalías geológicas, la detección temprana de deslizamientos de tierra sigue teniendo desafíos críticos que pueden llegar a poner en riesgo comunidades e infraestructuras. Tradicionalmente, se han empleado métodos convencionales que en su mayoría estaban basados en inclinómetros, extensómetros y pluviómetros para lograr evaluar la estabilidad en las zonas que son denominadas de alto riesgo. Sin embargo, estos sistemas presentan limitaciones como una cobertura reducida ya que al ser instaladas de manera puntual no ofrecen una cobertura continua lo que impide captar en tiempo real cambios sutiles que pueden preceder a un deslizamiento [14].

La dependencia de técnicas reactivas y la falta de integración entre sistemas de monitoreo dificultan la emisión de alertas tempranas, generando desfases entre la aparición de señales de inestabilidad y la puesta en marcha de medidas preventivas [1]. Adicionalmente, la complejidad de procesos que desencadenan un deslizamiento, tales como la variabilidad de las condiciones climáticas, la heterogeneidad del terreno y la interacción de varias variables ambientales requiere un enfoque multidimensional.

En estudios recientes se ha demostrado que técnicas de Machine Learning y Deep Learning puede mejorar significativamente la detección y predicción de estos fenómenos al identificar patrones complejos en grandes volúmenes de datos, reduciendo falsos positivos en más de un 30% comparado con modelos clásicos [3]. No obstante, la mayoría de los modelos presentados en la literatura se han validado en entornos controlados o en modelos a escala.

Este estudio propone abordar estas limitaciones mediante el uso de un enfoque que combine redes de sensores inteligentes y técnicas de análisis de datos para reducir el impacto de los deslizamientos de tierra en zonas vulnerables y permitir una gestión más eficaz y oportuna.

1.4 Justificación.

La magnitud de los deslizamientos de tierra, tanto en términos de pérdidas humanas como de daños a la infraestructura, constituye un problema crítico a nivel global. La creciente incidencia de estos eventos, intensificada por el cambio climático y la expansión descontrolada en áreas de alto riesgo, exige la implementación de estrategias de monitoreo que permitan anticipar y mitigar sus impactos [15].

Los métodos convencionales, aunque útiles en ciertos contextos, presentan limitaciones significativas en cobertura espacial y continuidad en la recolección de datos. Por ello, resulta indispensable incorporar tecnologías emergentes como redes de sensores y algoritmos de ML, cuyas ventajas incluyen la capacidad de capturar información simultánea desde múltiples puntos mediante topologías en malla. Esta característica posibilita la detección de cambios sutiles que, en conjunto, pueden predecir la inestabilidad en taludes [1].

La aplicación de técnicas de ML ha demostrado eficacia en la identificación de patrones premonitorios. Un modelo entrenado con imágenes GeoTIFF y datos históricos alcanzó alta precisión en detección de áreas inestables [3]. De igual manera proyectos como “LandSlide4Sense” han proporcionado conjuntos de datos integrados que fusionan imágenes ópticas, modelos digitales de elevación y capas de pendiente, demostrando que la combinación de fuentes de información mejora la identificación de zonas críticas [16].

Esta investigación plantea como objetivo implementar las tecnologías mencionadas para detección de deslizamientos, mediante dispositivos especializados que midan parámetros como: humedad del suelo, inclinación del terreno y condiciones climáticas.

Los datos obtenidos serán transmitidos de forma inalámbrica a un centro de monitoreo, donde serán analizados mediante el algoritmo predictivo desarrollado en el lenguaje de

programación Python. El algoritmo se encargará de analizar los datos y determinar mediante un porcentaje la probabilidad de deslizamiento en la zona de estudio.

La viabilidad de este sistema radica en la posibilidad de realizar varias pruebas en ambientes controlados, lo que permite evaluar su eficacia y precisión en condiciones simuladas antes de su despliegue en entornos geográficos reales. Esta metodología de prueba ofrece la ventaja de ajustar y perfeccionar los algoritmos de Machine Learning, optimizando así la capacidad predictiva del sistema [17].

1.5 Objetivos.

1.5.1 Objetivo General

- Implementar una red de sensores basada en análisis de datos y Machine Learning para la detección temprana y prevención de deslizamientos de tierra.

1.5.2 Objetivos específicos

- Entrenar los algoritmos de Machine Learning para la detección temprana de deslizamientos de tierra, a partir del análisis de datos estadísticos de deslizamientos ocurridos en el pasado.
- Implementar una red de sensores y algoritmos que permitan la recolección y transmisión de datos a un centro de monitoreo de manera confiable y segura.
- Evaluar la precisión y eficacia del sistema de detección temprana de deslizamientos de tierra mediante la realización de pruebas de validación y calibración, utilizando ambientes controlados.

CAPÍTULO II.

2.1 Estado del arte.

La investigación de la literatura indica un creciente interés en aprovechar el Machine Learning y la inteligencia artificial en diversos ámbitos, la creciente frecuencia y el devastador impacto de los deslizamientos de tierra a nivel mundial exigen el desarrollo de sistemas eficaces de detección temprana y prevención [1]–[3]. Los métodos tradicionales de monitoreo y predicción de deslizamientos de tierra suelen basarse en modelos deterministas con un número limitado de variables estáticas, que pueden ser subjetivos y menos precisos [4]. La llegada de tecnologías avanzadas de sensores, protocolos de comunicación inalámbrica y técnicas sofisticadas de análisis de datos, en particular el Machine Learning y la inteligencia artificial, ofrece prometedoras vías para mejorar las capacidades de alerta temprana [1]–[6], [8], [18], [19].

Diversos estudios usan ML para diversos propósitos, por ejemplo [20] se implementó técnicas de Machine Learning para predecir el impacto que tendrá la humedad y la temperatura en el peso corporal de cuyes. Para lograr esto se realizó un proceso que involucró recolectar datos de peso, humedad y temperatura de los cuyes en un periodo de 92 días. Posteriormente, se aplicaron técnicas de Machine Learning para analizar estos datos y predecir la influencia de la humedad y la temperatura en el peso corporal. Los resultados obtenidos mostraron que el control ambiental tuvo un efecto positivo en el aumento de peso de los cuyes, con la jaula controlada mostrando un mayor incremento de peso. A pesar de no haber diferencias significativas entre grupos según el análisis estadístico, la jaula controlada en temperatura y humedad tuvo mejores resultados.

Otras fuentes usan el Machine Learning para predecir fenómenos de remoción de masas, [8] implementa algoritmos como Redes Neuronales Artificiales (ANN), Bosques Aleatorios (RF) y Máquina de Vectores de Soporte (SVM) con el fin de identificar las zonas con mayor probabilidad de que ocurra un evento geológico. Para lograr esto usaron 11 variables como inventario de FRM, topografía, pendiente, aspecto, índice topográfico de humedad, índice de potencial de flujo, infiltración, precipitación, litología, cobertura vegetal y tipo de suelo que posteriormente fueron procesadas para su implementación. Se obtuvo como resultado que tres modelos ANN, RF y SVM fueron obtenidos con una muy alta precisión, dando

como resultado que la vía de estudio Papallacta-Quebrada Negra presenta una muy alta susceptibilidad a sufrir un movimiento de masa.

Por otra parte [21] propone un método de monitoreo de deslizamientos que utiliza una red LoRa e IoT con sensores inteligentes. El estudio busca abordar las limitaciones de los sistemas existentes de monitoreo de deslizamientos, como la baja velocidad de la red, el retardo en la transmisión de datos, la distancia de transmisión limitada y el alto consumo de energía, que dificultan el monitoreo de redes a gran escala, de larga distancia y ad-hoc con confiabilidad garantizada, su sistema se compone de dos nodos importantes, uno de monitoreo y recolección in situ y otro nodo de enlace.

De la misma forma [22] ofrece una visión general del estado actual de los sistemas de monitoreo de caídas de rocas y deslizamientos que utilizan redes de sensores inalámbricas. Analiza la implementación y los métodos de diversas soluciones, junto con su arquitectura de sistema y aspectos relevantes del hardware. Los datos recopilados se utilizan para identificar las tendencias actuales y las posibilidades futuras en el campo del monitoreo de riesgos geológicos.

Para deslizamientos de tierra [6] muestra una propuesta para la detección de deslizamientos soportada en inteligencia artificial, en particular en el uso de algoritmos de Machine Learning que permiten la detección en grandes áreas de la superficie terrestre de una forma rápida a cualquier escala de trabajo. Para lograr esto se entrenó a los algoritmos con modelos como árbol de decisiones, SVM, redes neuronales artificiales, Gradient Boosting Machine y StackingClassifier con los datos obtenidos de tres zonas de deslizamientos distintas. Los resultados arrojados con niveles de generalización de variables están directamente relacionados con el accuracy de la técnica usada, la técnica donde se empleó un árbol de decisiones tiene un 20%, mientras que otras como el StackingClassifier posee un porcentaje del 97% teniendo un nivel de generalización bajo lo que resulta muy útil en algunas disciplinas que requieren estos altos niveles de precisión.

De manera similar [18], muestra el desarrollo de un modelo predictivo que servirá para identificar zonas potenciales de deslizamientos influenciados por la precipitación en Medellín. Para lograr esto se usaron técnicas de inteligencia artificial y se incorporó

información geoespacial y de precipitación de acceso público. El modelo se fundamentó en la aplicación de enfoques supervisados y series temporales, adaptados de manera híbrida para comprender la conexión entre la precipitación y los fenómenos de movimientos en masa. Los resultados obtenidos de esta investigación indican que el método de Light Gradient Boosting Machine se adapta con mayor precisión en comparación con varios métodos supervisados evaluados también se destacan las mejores métricas de precisión y evaluación al considerar las variables de estudio.

En otros estudios como en [23] se desarrolla un sistema de bajo costo basado en Internet de las cosas (IoT) para la detección, monitoreo, predicción y advertencia de deslizamientos de tierra. Para lograrlo el sistema utiliza una combinación de sensores, microcontroladores, tecnología IoT, análisis de datos en la nube y modelos de Machine Learning para recopilar, procesar y analizar los datos de los sensores y generar alertas y predicciones de riesgo de deslizamiento de tierra en tiempo real, se compone de dos configuraciones: una para la detección y otra para la detección y monitoreo posterior. La primera configuración utiliza sensores de aceleración, humedad del suelo, fuerza y ultrasonido para detectar y medir los movimientos del suelo y otros parámetros relevantes. La segunda configuración utiliza un sensor de ultrasonido y un módulo WIFI para detectar y monitorear los movimientos del suelo después de un deslizamiento de tierra. Los resultados muestran que el sistema es capaz de detectar y predecir el riesgo de deslizamiento de tierra con una precisión del 90%. Además, el sistema es capaz de generar alertas en tiempo real y monitorear los movimientos del suelo después de un deslizamiento de tierra.

Finalmente, en [24] se menciona el uso de sensores inteligentes para monitorear el movimiento de rocas y adoquín en deslizamientos de tierra. Para lograr esto, se realizaron experimentos de laboratorio en los que se instaló un tag equipado con acelerómetros, giroscopios y magnetómetros dentro de un adoquín. Los experimentos consistieron en dejar caer el adoquín en una pendiente inclinada, generando diferentes modos de movimiento como rodar, rebotar o deslizarse. La posición del adoquín se derivó de videos de cámaras y los datos de los sensores permitieron detectar el movimiento y separar los dos modos de movimiento, rodar y deslizarse. Además, se determinaron valores confiables para la posición, velocidad y aceleración mediante la alimentación de un filtro de Kalman con mediciones de sensores inteligentes y posiciones basadas en cámaras. Los resultados

mostraron que los sensores inteligentes pueden mejorar los sistemas de alerta temprana para deslizamientos de tierra y que hay potencial para su uso en la monitorización de deslizamientos de tierra en campo.

2.2 Fundamentación teórica.

2.2.1 Redes de sensores

Una red de sensores inalámbricos (WSN) se describe como un conjunto de numerosos nodos de sensores ampliamente distribuidos y conectados inalámbricamente para transmitir y recibir datos ambientales [23]. El objetivo principal de una WSN es recolectar datos, procesar información y enviarla a un nodo central a través de un elemento gateway para su almacenamiento y análisis [14].

Las redes de sensores son parte integral del concepto del Internet de las Cosas, que proporciona la plataforma necesaria para que los sistemas integren sensores y los conecten en tiempo real para monitorear eventos y recopilar datos [1]. En el contexto de la detección de deslizamientos de tierra, los sensores de una WSN se diseñarían para recopilar información relevante para el monitoreo de las condiciones ambientales.

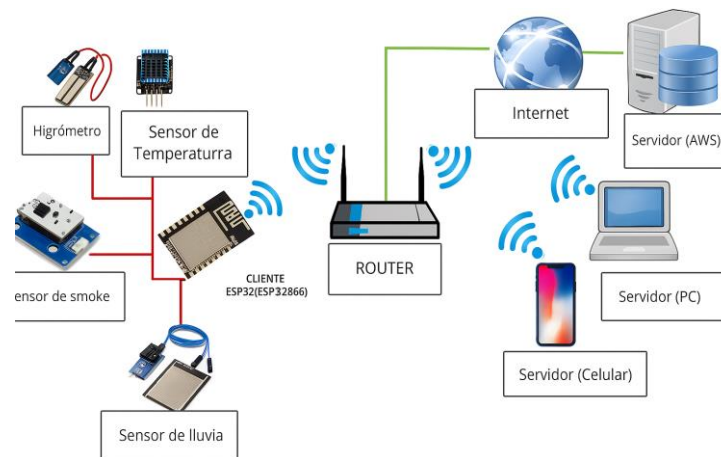


Figura 1. Esquema de una red de sensores [9].

Los nodos que hay en una WSN son:

- **Nodos sensoriales:** Su función principal es recopilar información del entorno mediante diversos sensores. Esta información puede estar relacionada con parámetros físicos o ambientales dependiendo de su aplicación [9], [14], [25].
- **Nodos enrutadores:** Estos nodos reciben datos de los nodos sensoriales y reenvían estos paquetes de datos al gateway u otros nodos intermedios [1], [14].
- **Nodos Gateway o Estación Base:** Este nodo actúa como el punto central de la red de sensores y sirve de puente hacia otras redes, como internet, para el análisis y almacenamiento de datos [14]. Suele tener mayor potencia, capacidad de procesamiento y una fuente de energía más estable que el resto de los nodos.

2.2.2 Características de las WSN

Las redes de sensores inalámbricos poseen características distintivas que son relevantes para su aplicación en la detección temprana de deslizamientos de tierra:

- **Naturaleza distribuida:** Se despliegan en un área geográfica, permitiendo una cobertura espacial amplia para la recopilación de datos sobre las condiciones del terreno [23].
- **Comunicación inalámbrica:** La comunicación entre los nodos se realiza sin necesidad de cables, lo que facilita su despliegue en terrenos difíciles y remotos, lo que es una característica de las zonas propensas a deslizamientos [1], [23].
- **Autonomía:** Los nodos sensores están diseñados para operar de forma autónoma después de su despliegue, recolectando y transmitiendo datos sin intervención humana constante [14].
- **Escalabilidad:** Idealmente, una WSN debe ser fácilmente expandible mediante la adición de nuevos nodos para aumentar la cobertura de información recopilada en el área de monitoreo [12], [14].
- **Monitoreo en tiempo real:** Una característica fundamental para la detección temprana es la capacidad de la WSN para monitorear eventos y recopilar datos en tiempo real, lo que permite detectar cambios significativos en las variables monitoreadas que podrían indicar la inminencia de un deslizamiento [1], [12], [14].

2.2.3 Topología en Malla

Los nodos sensores se comunican directamente con sus vecinos y los datos pueden ser enrutados a través de múltiples caminos hacia la estación base. Esta topología ofrece mayor robustez y tolerancia a fallas, ya que, si un nodo falla, los datos pueden ser transmitidos a través de otras rutas [1], [14]. Las redes malladas son especialmente útiles en áreas extensas o con obstáculos.

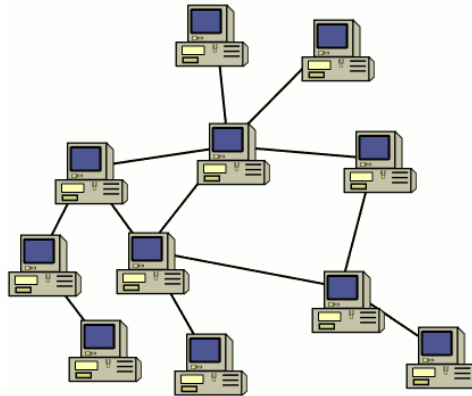


Figura 2. Topología en Malla [9].

Existen otras topologías que se pueden implementar como la estrella donde todos los nodos sensores se conectan directamente a uno central [14], o la de árbol que es una variante de la topología en estrella organizada por jerarquías [21], [25].

2.2.4 Protocolo de comunicación LoRa

Es una tecnología de modulación y demodulación inalámbrica de bajo consumo y larga distancia que utiliza un esquema de espectro ensanchado. Es conocida por su gran distancia de transmisión, alcanzando un alcance efectivo de 2 a 5 km en zonas urbanas y de 5 a 10 km en terreno [21]. LoRa admite frecuencias de trabajo flexibles y bandas de frecuencia ISM como 433/868/915 MHz. Permite un acceso flexible y una gran capacidad, con una única puerta de enlace que admite la conexión de miles de nodos, también admite comunicación full-duplex y concurrencia de enlaces ascendentes y descendentes [9]. LoRa utiliza modulación Chirp Spread Spectrum (CSS), que proporciona robustez frente a interferencias, es además capa física del protocolo LoRaWAN.

2.2.5 Comparación con otros protocolos

Tabla 1. Comparación de protocolos de comunicación [9], [21], [25], [26].

Características	LoRa/LoRaWAN	Zigbee	Wi-Fi
Alcance	Largo (2-10 km)	Corto	Corto (hasta 100 m)
Velocidad de datos	Baja	Baja a media (20-250 kbps)	Alta
Potencia	Muy baja	Baja	Alta
Aplicaciones	IoT de largo alcance, Ciudades inteligentes, Agricultura	Domótica, Redes de Seguridad Inalámbrica (WSN), Control industrial	Acceso a internet, Redes locales
Estándar	LoRa (Propietario), IEEE 802.15.4 (subyacente)	IEEE 802.15.4	IEEE 802.11

2.2.6 Internet de las cosas (IoT)

Se refiere a una red de objetos físicos equipados con sensores, software y otras tecnologías que permite la conexión e intercambio de datos a través de internet [5]. El IoT proporciona la plataforma necesaria para que los sistemas integren sensores y los conecten en tiempo real para monitorear eventos y recopilar datos [1]. Esto se alinea perfectamente con los requisitos de un sistema de detección temprana de deslizamientos.

2.2.6.1 Ventajas

- Monitoreo y recopilación de datos en tiempo real: Esta capacidad es crucial para aplicaciones de monitoreo de variables ambientales, los sensores pueden recopilar continuamente datos relevantes sobre factores que podrían desencadenar un deslizamiento de tierra [25], [26].

- Integración con otras tecnologías: En el contexto de los sistemas de alerta temprana, el IoT integra sensores y los conecta en tiempo real para monitorear eventos y recopilar datos [1].
- Mejora de la toma de decisiones: La gran cantidad de datos recopilados permite una toma de decisiones mejor informada en diversos ámbitos [5]. En la predicción de deslizamientos, se aplican algoritmos de Machine Learning a estos datos para su análisis y predicción.
- Fácil expansión: Una característica deseable de las redes de sensores basadas en el IoT es su potencial de fácil expansión mediante la adición de nuevos nodos en diferentes ubicaciones [12].

2.2.7 Machine Learning

El Machine Learning (ML) es una rama multidisciplinaria de la inteligencia artificial que se centra en el desarrollo de algoritmos y modelos que permiten a las computadoras aprender y mejorar su rendimiento en tareas específicas mediante la experiencia y la observación de datos [5]. Permite que las máquinas aprendan a resolver problemas basándose en diferentes experiencias de aprendizaje, sin necesidad de ser programadas explícitamente para una actividad específica [27]. En lugar de codificación manual, el ML ajusta los modelos matemáticos basándose en los datos de entrenamiento para realizar tareas como clasificación, regresión y agrupamiento [20].

En términos más simples, el ML consiste en crear modelos a partir de la información proporcionada para extraer conclusiones y resolver problemas. Estos modelos pueden adaptarse de forma independiente al exponerse a nuevos datos, aprendiendo de cálculos previos para tomar decisiones y producir resultados fiables o repetibles [19]. Un modelo de ML puede considerarse como una especie de filtro por el que entran nuevos datos y el resultado es la clasificación de esos datos según los patrones detectados.

Es importante destacar que el aprendizaje en ML no consiste en memorizar datos, sino en crear un modelo para realizar predicciones o clasificaciones su función principal es extraer información valiosa de los datos con el objetivo de generar nuevo conocimiento que permita la innovación, facilite la toma de decisiones y mejore los procesos [5].

2.2.8 Tipos de ML

2.2.8.1 Aprendizaje supervisado

El aprendizaje supervisado es una técnica en la que el modelo se entrena con una cierta cantidad de datos definidos o descritos mediante etiquetas, que actúan como entrada donde se conoce el resultado deseado, este tipo de aprendizaje necesita que una persona le enseñe a la computadora cómo realizar una acción [19]. Los algoritmos se entrenan con datos etiquetados, lo que significa que cada ejemplo tiene una entrada conocida y una salida deseada [20], [27]. Por medio de estas entradas y salidas, el modelo aprende las relaciones para realizar predicciones sobre datos nuevos y similares.

Estos algoritmos crean modelos para pronosticar los valores de una variable de respuesta (m-dimensional) a partir de los valores de una variable predictora (n-dimensional), estos modelos se crean a partir de un subconjunto de la información disponible (normalmente alrededor del 75%) que se utiliza para entrenar el modelo y posteriormente se validan o verifican con la información restante (25%) [17]. Se considera este modelo supervisado debido a que genera, prueba y evalúa con la información disponible, donde una persona expone la respuesta correcta para verificar la respuesta inferida del modelo.

2.2.8.2 Aprendizaje no supervisado

El aprendizaje no supervisado se centra en algoritmos que deben encontrar por sí solos la estructura oculta en datos no etiquetados. En este tipo de aprendizaje, no existen respuestas correctas y los algoritmos suelen realizar procesos de agrupación y asociación para clasificar los datos [20]. A diferencia del aprendizaje supervisado, el aprendizaje no supervisado no cuenta con un subconjunto de información para validar o evaluar el modelo, su función es recopilar toda la información disponible y observar patrones interesantes dentro de un conjunto de observaciones [17]. El objetivo es inferir las propiedades de los datos sin la ayuda de etiquetas que evalúe la fiabilidad o el error en cada observación; por lo tanto, las muestras se consideran no etiquetadas.

2.2.9 Algoritmos de ML

2.2.9.1 Regresión logística

Regresión logística mapea una combinación lineal de entradas a una probabilidad de clasificación usando una función sigmoide. Ajusta sus coeficientes vía máxima verosimilitud, interpretándolos a través de los log-odds y odds ratio. Es simple, eficaz y ampliamente usada para tareas de clasificación binaria, siempre que se cumplan ciertos supuestos y se tengan datos adecuados, su objetivo central es predecir la probabilidad de que los datos de entrada pertenezcan a una clase específica [2], [28].

Matemáticamente para un problema primero obtiene una combinación lineal o log-odds [29]:

$$z = \beta_0 + \sum_{j=1}^p \beta_j x_j \quad (1)$$

donde:

x_j : Valor de la variable independiente.

β_j : Coeficiente asociado a x_j que mide su impacto en el log-odds.

β_0 : intercepto o bias del modelo.

Luego aplicando la función sigmoide obtiene un valor de predicción entre 0 y 1 [29], la aplicación de este algoritmo se observa en al *Anexo 1*:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \quad (2)$$

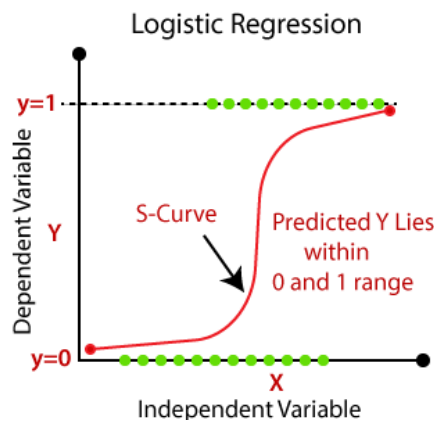


Figura 3. Regresión logística

2.2.9.2 SVM

Las máquinas de soporte de vectores (SVM) son métodos de clasificación y regresión que encuentra el hiperplano óptimo que separa las clases en el espacio de características con el mayor margen posible [2], [4], [7]. En problemas no lineales, se utiliza el truco del kernel para transformar los datos a un espacio de mayor dimensionalidad donde la separación lineal sea posible [27].

Matemáticamente, para un problema de clasificación binaria con puntos de datos (x_i, y_i) , donde x_i es el vector de características e $y_i \in \{-1, +1\}$ es la etiqueta de clase, la ecuación de un hiperplano de separación viene dada por:

$$w \cdot x + b = 0 \quad (3)$$

donde:

w es el vector de ponderación perpendicular al hiperplano.

x es el vector de características.

b es el término de sesgo (determina la posición del hiperplano).

" $\cdot$ " denota el producto escalar.

El objetivo de la SVM es encontrar w y b que maximicen el margen, que es la distancia entre el hiperplano y los puntos de datos más cercanos de cada clase (llamados vectores de soporte) [4], [6].

Para datos linealmente separables, el problema de optimización puede formularse como la minimización de $\|w\|^2$ sujeta a las restricciones:

$$y_i(w \cdot x_i + b) \geq 1 \text{ para todo } i \quad (4)$$

Para datos no linealmente separables, SVM utiliza una función kernel ($K(x_i, x_j)$) para mapear los datos en un espacio de mayor dimensión donde se puede encontrar un hiperplano lineal [4]. Las funciones kernel comunes incluyen funciones lineales, polinómicas y de base radial (RBF).

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j) \quad (5)$$

Donde:

x_i, x_j son vectores de características en el espacio original de entrada. Representan las observaciones “i” y “j” respectivamente.

$\phi(x)$ es una función de transformación que mapea el vector de características x desde el espacio original a un espacio de características de mayor dimensión. El objetivo de esta transformación es facilitar la separación lineal de los datos que no son linealmente separables en el espacio original.

$\phi(x_i)^T \phi(x_j)$ denota el producto punto entre los vectores transformados $\phi(x_i)$ y $\phi(x_j)$ en el espacio de características de mayor dimensión. Este producto interno mide la similitud entre las dos observaciones después de la transformación.

La función de decisión para un nuevo punto de datos x' se convierte en [7]:

$$f(x') = \text{sing}(w^T x' + b) \quad (6)$$

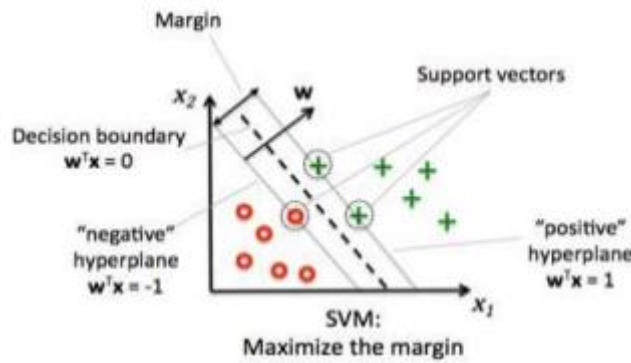


Figura 4. Ajustes de SVM [8].

En Python, Scikit-Learn ofrece la clase *SVC* para clasificación y *SVR* para regresión. La implementación de este algoritmo se observa en el Anexo 2.

2.2.9.3 Random Forest

Bosque aleatorio es un método de aprendizaje por conjuntos que construye múltiples árboles de decisión en diferentes subconjuntos de datos y luego promedia sus predicciones (para la regresión) o toma la mayoría de los votos (para la clasificación) [4], [5]. Esto ayuda a reducir el sobreajuste y a mejorar la robustez del modelo.

El proceso de construcción de un Bosque aleatorio implica:

Muestreo bootstrap: Creación de múltiples subconjuntos (normalmente cientos o miles) de los datos de entrenamiento originales mediante muestreo con reemplazo. Cada subconjunto se utiliza para entrenar un único árbol de decisión [5], [17], [30]:

- Submuestreo de características: Al construir cada árbol, en cada nodo, solo se considera un subconjunto aleatorio de las características disponibles para la división. Esto introduce mayor aleatoriedad y reduce la correlación entre los árboles [5], [17].
- Inducción del árbol de decisión: Cada árbol de decisión se desarrolla en la muestra bootstrap utilizando las características seleccionadas aleatoriamente [31], [32]. El árbol se desarrolla típicamente hasta que cada nodo hoja contiene un pequeño número de instancias o hasta que se alcanza cierta profundidad. El criterio de división en cada nodo busca maximizar la separación de clases utilizando técnicas como la impureza de Gini o la ganancia de información para la clasificación, y la reducción de la varianza para la regresión [28], [33].

Para realizar una predicción para un nuevo punto de datos:

- Clasificación: Cada árbol del bosque realiza una predicción (etiqueta de clase). La predicción final es la clase que recibe la mayoría de los votos de todos los árboles [17].
- Regresión: Cada árbol predice un valor continuo. La predicción final es el promedio de las predicciones de todos los árboles [17].

$$\hat{y} = \text{mode}\{h_t(x), \quad t = 1, \dots, T\} \quad (7)$$

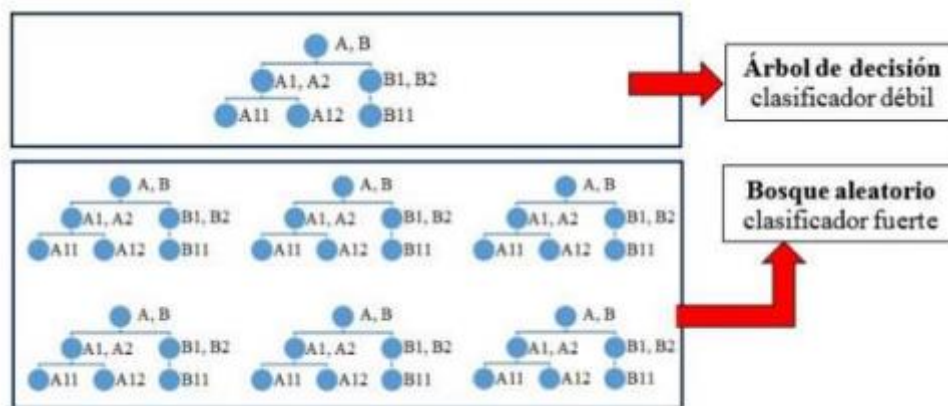


Figura 5. Árbol de decisión y grupo de árboles de decisión (Random Forest) [8].

En Python, Scikit-Learn ofrece la clase *RandomForestClassifier* para clasificación y *RandomForestRegressor* para regresión. La implementación de este algoritmo se observa en el *Anexo 3*.

2.2.9.4 Redes neuronales (MLP)

Las redes neuronales son un conjunto de algoritmos inspirados en la estructura y función del cerebro humano. Consisten en nodos interconectados llamados neuronas, organizados en capas, una capa de entrada, una o más capas ocultas y una capa de salida [17], [19].

Una sola neurona en una red neuronal suele realizar el siguiente cálculo:

- Suma ponderada de entradas: Recibe entradas de otras neuronas (o de la capa de entrada) y calcula una suma ponderada de estas entradas, junto con un término de sesgo [17]:

$$z = \sum_{i=1}^n w_i x_i + b = w^T x + b \quad (8)$$

Donde:

x_i son las entradas.

w_i son los pesos correspondientes.

b es el sesgo.

La suma ponderada se pasa a través de una función de activación no lineal (g) para generar la salida de la neurona [17]:

$$y = g(z) = g(w^T x + b) \quad (9)$$

Entre las funciones de activación más comunes se incluyen la sigmoidea, la ReLU (Unidad Lineal Rectificada) y la tangente hiperbólica [19]. La no linealidad de la función de activación es lo que permite que la red aprenda relaciones complejas en los datos que esté analizando.

Una red aprende ajustando las ponderaciones y los sesgos durante un proceso llamado entrenamiento [17], [19]. Para el aprendizaje supervisado, la red se entrena con datos etiquetados. El proceso implica:

- **Pase directo:** Los datos de entrada se introducen en la red y se calcula la salida (predicción).
- **Función de pérdida:** Una función de pérdida o también llamada función de coste mide la discrepancia entre la predicción de la red y el valor objetivo real [17], [19]. Las funciones de pérdida más comunes incluyen el error cuadrático medio (MSE) para la regresión y la entropía cruzada para la clasificación.
- **Retropropagación:** Los gradientes de pérdida con respecto a las ponderaciones y los sesgos se calculan y se propagan a través de la red.
- **Optimización:** Un algoritmo de optimización como descenso de gradiente, utiliza estos gradientes para actualizar las ponderaciones y los sesgos en una dirección que minimice la pérdida [19]. Este proceso se repite en múltiples iteraciones (épocas) hasta que la red aprende a realizar predicciones precisas.

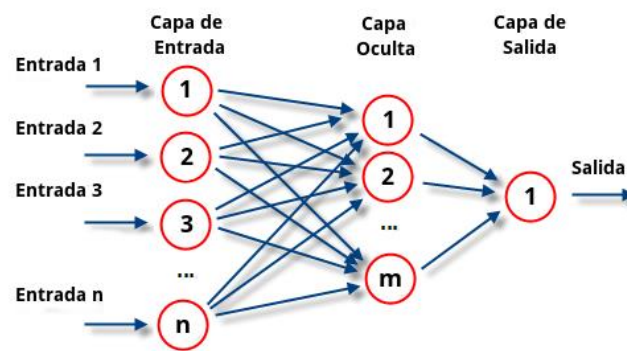


Figura 6. Esquema de una red neuronal simple con una sola capa oculta [8].

La implementación de este algoritmo se observa en el *Anexo 4*.

2.2.10 Métricas de validación

2.2.10.1 Matriz de confusión

Una matriz de confusión es una herramienta fundamental en el ámbito del aprendizaje automático supervisado, utilizada para evaluar el rendimiento de modelos de clasificación. Esta matriz es una tabla que compara las predicciones del modelo con los valores reales, permitiendo identificar aciertos y errores en las clasificaciones realizadas [2].

Tabla 2. Matriz de confusión.

	Predicción Positiva	Predicción Negativa
Condición Positiva	Verdaderos Positivos (VP)	Falsos Negativos (FN)
Condición Negativa	Falsos Positivos (FP)	Verdaderos Negativos (VN)

2.2.10.2 Precisión

La precisión indica la frecuencia con la que el modelo acierta al predecir un deslizamiento. Una precisión alta significa que, cuando el modelo indica que ocurrirá un deslizamiento, es muy probable que sea cierto.

Es calculada como:

$$Accuracy = \frac{VP + VN}{VP + VN + FP + FN} \quad (10)$$

Cuántas de las precisiones positivas fueron correctas:

$$Recall = \frac{VP}{VP + FP} \quad (11)$$

2.2.10.3 Sensibilidad

La sensibilidad indica la capacidad del modelo para identificar todos los deslizamientos que realmente ocurren. Una sensibilidad alta significa que el modelo es capaz de capturar la mayoría de los deslizamientos reales [4].

Es calculada como:

$$Recall = \frac{VP}{VP + FN} \quad (12)$$

2.2.10.4 F1-Score

Proporciona una métrica única que equilibra tanto la precisión como la sensibilidad. Es especialmente útil cuando existe una distribución de clases desigual lo cual suele ocurrir en la predicción de deslizamientos [2].

Es calculado como:

$$F1 = 2 \frac{Presicion * Recall}{Presicion + Recall} \quad (13)$$

2.2.10.5 Área bajo la curva ROC (AUC-ROC)

Representa la probabilidad de que el clasificador clasifique una instancia positiva elegida aleatoriamente por encima de una instancia negativa elegida aleatoriamente. Un AUC de 0,5 indica una capacidad de discriminación nula, mientras que un AUC de 1,0 representa un clasificador perfecto [4], [18].

La curva ROC grafica el TPR contra el FPR:

$$TPR = \frac{VP}{VP + FN} , \quad FPR = \frac{FP}{FP + VN} \quad (14)$$

2.2.11 Deslizamientos de tierra

2.2.11.1 Definición

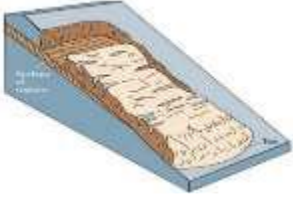
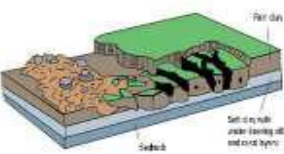
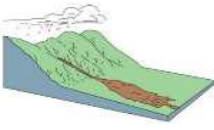
Un deslizamiento de tierra se define como el movimiento de una masa de roca, escombros o tierra pendiente abajo [4], [12], [34], [35]. Cuando una pendiente se derrumba o se produce un movimiento en ella, se denomina deslizamiento de tierra, estos fenómenos son naturales y de gran peligro, con una capacidad destructiva que puede ocasionar importantes pérdidas materiales y humanas [4], [35]. En general, los deslizamientos de tierra ocurren cuando una parte de una pendiente no soportar su propio peso, lo que provoca que se vuelva muy inestable. También se describen como eventos generados por procesos gravitacionales pendiente abajo de roca, regolito y suelo [6].

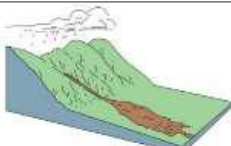
2.2.11.2 Clasificación

Para poder clasificar los deslizamientos de masa hay diversos tipos de modelos basados que se basan en criterios para poder diferenciarlos.

Uno de los más aceptados pese a su antigüedad es el observado en [36], así, divide los deslizamientos en seis tipos: caídas, vuelcos, deslizamientos, propagación lateral, reptación y flujos. En la tabla 3 se puede observar una clasificación más detallada

Tabla 3. Tipos de deslizamientos de tierra [37].

Tipo	Esquema	Descripción
Caída		Masa de cualquier tamaño que se desprende de un talud de pendiente fuerte, a lo largo de una superficie.
Vuelcos		Consiste en una rotación hacia delante de una o más unidades de material rocoso.
Deslizamiento Rotacional		Movimiento lento inicialmente de una masa de suelo, roca o los dos a lo largo de una superficie en forma circular y cóncava.
Deslizamiento Traslacional		Movimiento lento o rápido de una masa de suelo, roca o los dos a lo largo de una superficie plana.
Propagación Lateral		Movimiento que ocurre por expansión del material.
Reptación		Movimiento muy lento del suelo superficial sin una superficie definida.
Flujos		Movimiento que durante su desplazamiento exhibe un comportamiento como un fluido.

Avalancha		Desprendimientos de rocas en laderas escarpadas que pueden venir con hielo o nieve.
------------------	---	---

2.2.12 Factores de riesgo

Los deslizamientos de tierra se producen debido a un desequilibrio entre las fuerzas que actúan sobre el terreno. La estabilidad de un talud se rige por la relación entre la resistencia al corte del suelo o macizo rocoso y el esfuerzo cortante que le imponen la gravedad y otros factores.

2.2.13 Factores condicionantes

Son características intrínsecas del material y su resistencia, que representan las razones subyacentes de la inestabilidad [8]. Entre los factores condicionantes importantes se incluyen:

- **Topografía:** El ángulo y la geometría de la zona en peligro influyen significativamente en la distribución del peso del terreno y el potencial de movimiento [4], [38]. Las pendientes más pronunciadas suelen ser más propensas a deslizamientos.
- **Geología y litología:** El tipo de material geológico presente afecta significativamente las propiedades físicas y mecánicas del suelo, influyendo en su susceptibilidad a fallas, las variaciones en la litología afectan la resistencia al corte y la permeabilidad [4], [6], [8], [29]. Algunas unidades litológicas son más propensas a deslizamientos según sus condiciones.
- **Propiedades del suelo:** Factores como la porosidad, la plasticidad, el contenido de humedad y la resistencia al corte desempeñan un papel fundamental en la estabilidad de taludes [4], [6], [18]. Los cambios en la presión intersticial del agua pueden reducir significativamente la resistencia efectiva al corte del suelo.
- **Cobertura vegetal:** La vegetación puede influir en la estabilidad de taludes a través de sus sistemas radiculares, que pueden fijar las partículas del suelo y reducir la erosión, sin embargo, la deforestación o los cambios en la cobertura vegetal pueden disminuir la estabilidad [4], [8], [39].

- Condiciones hidrogeológicas: La presencia y el movimiento del agua subterránea pueden afectar significativamente la estabilidad de taludes al aumentar la presión intersticial del agua y reducir la tensión efectiva [8], [40]. Comprender las condiciones de drenaje del terreno es esencial para un análisis confiable de la estabilidad de taludes.

CAPÍTULO III.

3.1 Metodología.

3.1.1 Tipo de investigación

El presente estudio tiene un enfoque tanto cuantitativo como experimental. Es de tipo cuantitativo debido a la naturaleza objetiva del estudio, centrado en la recopilación y análisis de datos variables meteorológicas. Dada la necesidad de cuantificar la relación entre los datos recopilados por la red de sensores y la detección temprana de deslizamientos de tierra a través de algoritmos de Machine Learning, este enfoque proporcionará una base sólida para evaluar la eficacia del sistema propuesto.

A su vez es tipo experimental, ya que se evalúa el comportamiento de la red de sensores combinada con los algoritmos de Machine Learning para la detección temprana de deslizamientos de tierra. Este enfoque implica la manipulación de variables ambientales como la humedad, inclinación y precipitación en un entorno controlado, con el objetivo de observar cómo afectan al sistema y validar su capacidad de respuesta.

3.2 Método de investigación

3.2.1 Método cuantitativo

El enfoque cuantitativo predominará en toda la investigación, especialmente en las fases de Entrenamiento de algoritmos de Machine Learning y evaluación del sistema de detección temprana, en donde se utilizarán técnicas como análisis de regresión y clasificación, para entrenar los algoritmos con datos históricos de deslizamientos.

3.2.2 Método Experimental

Esta investigación aplica un método experimental, especialmente en la fase 3 en donde se diseñarán y ejecutarán pruebas controladas que reproducirán condiciones específicas de deslizamientos de tierra. Estas pruebas permitirán manipular variables relevantes y observar cómo el sistema responde en situaciones controladas. Las pruebas de validación y calibración se estructurarán como experimentos, donde se introducirán deliberadamente condiciones variables para evaluar la robustez y la adaptabilidad del sistema.

La integración de métodos experimentales y cuantitativos permitirá una investigación robusta, abordando tanto la validación estadística de la efectividad del sistema como la evaluación experimental de su rendimiento en condiciones controladas.

3.3 Técnica de recolección de datos

3.3.1 Estudio bibliográfico

Se llevará a cabo una revisión exhaustiva de la bibliografía científica y técnica relacionada con la implementación de redes de sensores y algoritmos de Machine Learning. Esta técnica permitirá contextualizar la investigación en el marco de conocimientos existentes, identificar mejores prácticas y lecciones aprendidas, y proporcionar una base teórica sólida para el desarrollo del sistema.

3.3.2 Recolección de datos con sensores

Se utilizaron sensores especializados para capturar datos en tiempo real sobre las condiciones del suelo y variables meteorológicas. Los sensores geotécnicos proporcionarán información sobre la estabilidad del suelo, mientras que los meteorológicos registrarán datos climáticos relevantes, como lluvias y temperaturas.

3.4 Población de estudio

3.4.1 Población

La población de estudio para esta investigación se centra en el conjunto de todas las salidas de probabilidad que el modelo genera bajo condiciones controladas. Este parámetro se define como $\hat{p}$ una variable continua entre $[1, 0]$. La investigación se propone analizar y cuantificar esta variable, con el objetivo de comprender las variaciones entre distintos algoritmos predictivos de Machine Learning.

3.5 Operacionalización de las variables

En la tabla 4 se detalla la operacionalización de las variables a utilizar en esta investigación las cuales tendrán una gran importancia para la comprobación y resultados obtenidos.

Tabla 4. Operacionalización de las variables.

Variables	Conceptos	Escala	Indicadores	Unidad de análisis	Técnicas/instrumentos
DEPENDIENTES					
PEO	Probabilidad estimada de ocurrencia de un deslizamiento de tierra en un ambiente controlado	Razón	-Valor que oscila entre 0 y 1.	Experimento	-Observación -Sistemas de monitoreo por la red de sensores
INDEPENDIENTES					
Tipo de algoritmo de detección.	Conjunto de instrucciones que permite a una computadora aprender de los datos y realizar tareas sin ser programada explícitamente.	Nominal	-Tipo de algoritmo (regresión logística, SVM, RF, MLP)	Modelo evaluado	-Observación directa -Scripts de evaluación

3.6 Procedimiento

El diagrama de flujo expuesto en la figura 7 presenta el procedimiento para el desarrollo de esta investigación, la cual se ha dividido en tres fases.

En la primera fase de la se recopilaron datos históricos de deslizamientos de tierra, incluyendo variables relevantes como condiciones climáticas y topográficas, luego estos datos son sometidos a técnicas de preprocesamiento, incluyendo limpieza, normalización y selección de características para garantizar la calidad y relevancia de la información para finalmente diseñar y configurar modelos de ML.

En la segunda fase se seleccionan los sensores y la tecnología de comunicación que llevará el prototipo, a su vez se establecerá el centro de monitoreo que recibirá y gestionará los datos provenientes de la red de sensores.

En la última fase se replicaron fenómenos de deslizamientos de tierra en ambientes controlados, así como pruebas para validar y calibrar el mismo, ajustando parámetros y mejorando los algoritmos en función de los resultados obtenidos, para finalmente evaluar la respuesta del sistema.

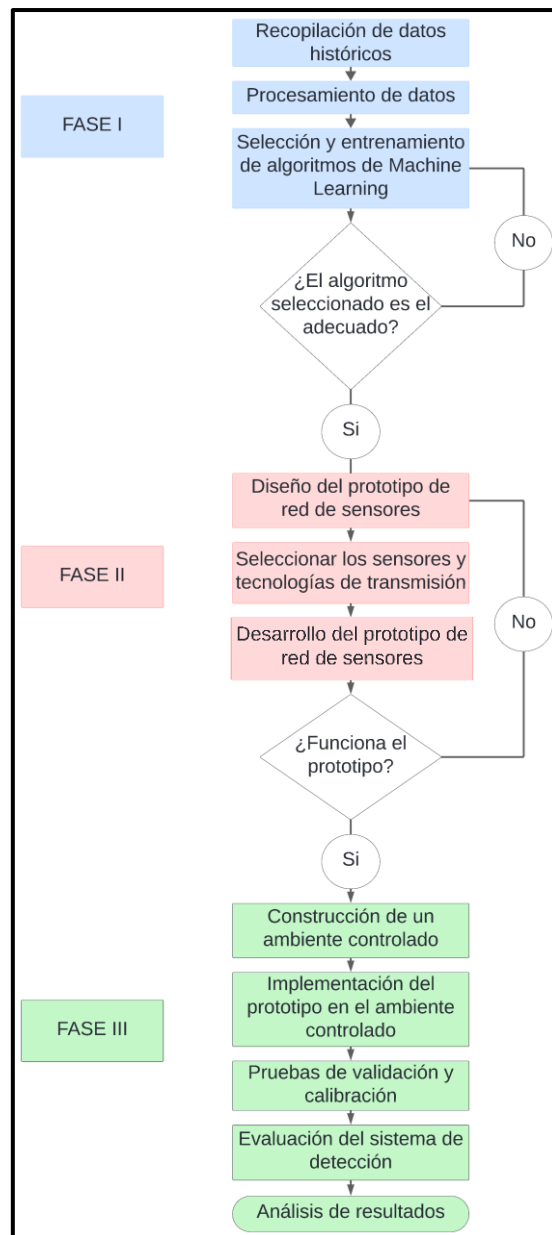


Figura 7. Procedimiento del trabajo de investigación.

3.7 Fase 1: Recolección de datos históricos

3.7.1 Recolección de datos históricos

Para la obtención de datos históricos relevantes, se recurrió a diversas fuentes primarias y secundarias, tanto locales como externas, que garantizan una perspectiva integral y rigurosa. En primer lugar, se coordinó con instituciones locales clave, como el Gobierno Autónomo Descentralizado del Cantón Alausí (GADMA) [41], el cual proporcionó registros históricos sobre el deslizamiento de tierra que ocurrió el 26 de marzo del 2023 y datos georreferenciados vinculados a dicha zona. Asimismo, se estableció colaboración con la Dirección de Gestión de Riesgos de la Provincia de Chimborazo, entidad que facilitó información técnica, reportes de eventos adversos y estrategias de mitigación implementadas en la región.

Adicionalmente, se incorporaron fuentes externas de acceso abierto disponibles en plataformas digitales. Entre estas destacan bases de datos y publicaciones de organismos internacionales como la Organización de las Naciones Unidas para la Alimentación y la Agricultura (FAO) [42], cuyos informes técnicos y datos históricos fueron de vital importancia para completar y corroborar datos que se utilizó para el entrenamiento del modelo predictivo de Machine Learning. A su vez se complementó esta información con conjuntos de datos públicos de repositorios académicos y portales gubernamentales, seleccionados bajo criterios de actualidad, confiabilidad y pertinencia temática. La incorporación de estas fuentes permitió contrastar y validar la información, asegurando una base sólida para el desarrollo de este proyecto.

3.7.2 Inventario de deslizamiento

La construcción de un inventario de deslizamientos robusto y diversificado es fundamental para el desarrollo de modelos predictivos confiables y generalizables ya que “el pasado y presente son las claves para el futuro” [4]. Un inventario de alta calidad no solo debe incluir datos detallados sobre la ubicación, magnitud y características morfológicas de los eventos registrados, sino también abarcar una variedad de contextos geográficos, climáticos y geotécnicos. Esto permite capturar la heterogeneidad de los factores desencadenantes y evitar sesgos inherentes a estudios focalizados en una única región.

Para garantizar la representatividad del modelo, las pruebas se realizaron en ambientes controlados utilizando un inventario compuesto por datos históricos de deslizamientos a escala global. Esta aproximación multirregional permite integrar casos de diferentes latitudes, ecosistemas y condiciones geológicas, para fortalecer la capacidad del modelo para extrapolar patrones en escenarios no vistos durante el entrenamiento.

La validación espacial de los deslizamientos incluidos en el inventario se apoyó en herramientas de georreferenciación, destacando el uso de Google Earth Pro. Esta plataforma permitió verificar la ubicación exacta de 741 deslizamientos a nivel mundial mediante la superposición de coordenadas y la comparación con registros históricos documentados.

Para el inventario de deslizamientos se tomó en cuenta solo dos tipos de deslizamiento, de tierra y de lodo con un porcentaje del 90% y 10% respectivamente como se observa en la figura 8.

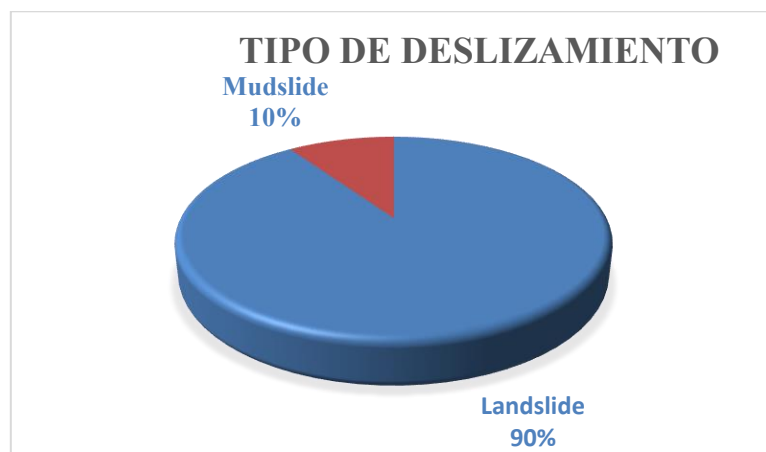


Figura 8. Porcentaje de los tipos de deslizamientos del inventario.

En cuando a la severidad de los deslizamientos presentes en el inventario son de severidad mediana con un 48.31%, seguido de pequeños con el 35.90%, deslizamientos grandes un 11.20% y afortunadamente un pequeño porcentaje corresponde a deslizamientos muy grandes y catastróficos, de algunos deslizamientos no se pudo obtener con exactitud su severidad exactamente un 3.64% del total del inventario como se observa en la figura 9.

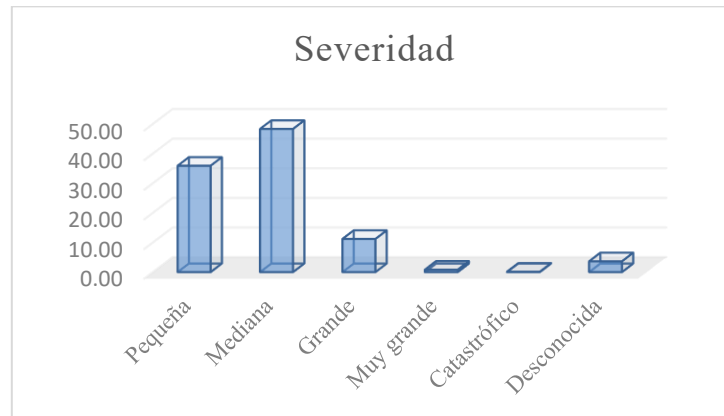


Figura 9. Porcentaje de severidad de los deslizamientos.

De la gran parte de los deslizamientos presentes en el inventario se pudo identificar un desencadenante que produjo el evento, obteniendo un mayor porcentaje los aguaceros con 42.78%, seguido de las lluvias con un 27.26%, al igual que las lluvias continuas con un 24.02% tal como se muestra en la figura 10.

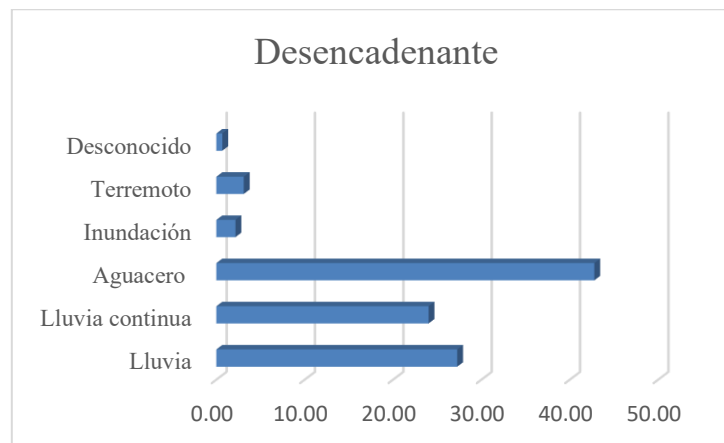


Figura 10. Porcentaje del desencadenante de deslizamientos.

3.7.3 Selección de factores condicionantes

En este apartado se escogen las variables que son responsables de la ocurrencia del evento y cuyos valores deben ser incorporados en los datos de entrenamiento del modelo predictivo. De acuerdo con la revisión bibliográfica las variables que más utilizan en estos casos son factores geomorfológicos, meteorológicos, uso del suelo, cobertura vegetal y el tipo de suelo.

Para los datos históricos se tomó de cada categoría las variables que son consideradas más relevantes para el estudio considerando la disponibilidad de los datos y la influencia de las variables aplicando la técnica de Relación de Frecuencias.

3.7.4 Relación de frecuencias

Es un método estadístico ampliamente utilizado en estudios de susceptibilidad de deslizamientos u otros fenómenos naturales para evaluar la correlación entre la ocurrencia de un evento y clases específicas de factores condicionantes. Se utiliza este FR para determinar las variables más influyentes para la ocurrencia de un deslizamiento.

Se usa la técnica de Jenks para dividir cada variable en 10 clases y el cálculo de la frecuencia se realiza con la ecuación 15.

$$k = 1 + 3.322 * \log_{10}(n) \quad (15)$$

donde “n” es el número total de datos.

$$FR = \frac{(N^{\circ} \text{ de deslizamientos en la clase} / \text{Total de deslizamientos})}{(N^{\circ} \text{ de no deslizamientos de la clase} / \text{Total de no deslizamientos})} \quad (16)$$

Aplicando la ecuación 16 se obtiene un ranking de las clases en que fueron subdivididos los factores condicionantes, un factor de FR superior a 1 indica un mayor impacto sobre los deslizamientos mientras que si el factor de FR se acerca o es igual a 0 no afecta en la ocurrencia del fenómeno, en la tabla 5 se muestra un listado con los valores más altos de FR.

Tabla 5. Tabla de frecuencia de factores condicionantes.

Factor	Clase	FR
Precipitación acumulada 5 días	70.40-92.30	6.2
Pendiente	52.13-60.10	5.75
Humedad del suelo	39.82-44.64	4.76
Land cover	Sabana arbórea	3.36
Tipo de suelo	Franco arcilloso	3.21
Evapotranspiración	73.47-93.00	2.93
Porcentaje de presencia de vegetación	23.00-32.00	2.74
Precipitación acumulada 10 días	95.60-122.60	2.57
Radiación solar	159.07-185.15	2.43

Precipitación acumulada 20 días	105.20-146.10	2.28
Temperatura mínima	20.56-27.78	2.22
Precipitación acumulada 25 días	137.50-186.90	2.21
Precipitación acumulada 15 días	86.90-117.20	2.15
Porcentaje de suelo desnudo	0.00-9.00	1.89
Precipitación acumulada 30 días	109.00-153.60	1.87
Porcentaje de ausencia de vegetación	44.00-53.00	1.72
Temperatura máxima	28.10-34.50	1.5
Humedad relativa del ambiente	79.97-86.69	1.2
Temperatura promedio	23.60-28.20	1.15

3.7.5 Normalización de los datos

Los datos numéricos de las distintas variables presentes en los datos, varían en escalas y medidas muy diferentes, para resolver esto que afecta directamente al proceso de entrenamiento de los modelos predictivos es altamente recomendable normalizar los datos es decir que matemáticamente serán llevados a una misma escala. Una de las técnicas más utilizadas es dividir los valores de cada variable en rangos que van entre [0-1], para lograrlo se dividieron los datos en dos partes una para entrenamiento que representa al 80% de todos los datos históricos y otro para prueba que constituye el 20% restante, para evitar filtraciones de datos (data leakage).

3.7.6 Subdivisión de los datos históricos

Una de las tareas más importante al momento de entrenar un modelo predictivo es asegurarse en dividir estos datos en al menos 2 subconjuntos, uno de entrenamiento que sirva para aprender de los datos y ajustar los parámetros del modelo y otro de prueba que sirva para evaluar el grado de acierto que tiene el modelo en base a los datos del entrenamiento. La proporción más común es 80-20%, pero puede variar dependiendo de la literatura.

3.8 Fase 2: Diseño del prototipo y elección de la tecnología de comunicación

3.8.1 Arquitectura del sistema

El prototipo se basa en una topología IOT de baja potencia, con nodos sensores inalámbricos que capturan variables ambientales y las transmiten mediante tecnología LoRa a un nodo central. Los datos son procesados en una Raspberry Pi 5, que actúa como un servidor local para ejecutar algoritmos predictivos, la arquitectura se divide en:

- **Capa física:** Sensores y placa Helteck ESP32 LoRa.
- **Capa de comunicación:** Enlace en la banda de 915 MHz.
- **Capa de procesamiento:** Raspberry Pi 5 con broker HiveMQ, y modelos predictivos.

3.8.2 Sensores de monitoreo

La elección de los sensores se hizo en base a algunos factores como precisión, integración con el microcontrolador, consumo energético, costo y considerando que es un prototipo. A continuación, en la tabla 6 se detalla un análisis técnico de cada dispositivo.

Tabla 6. Características de los sensores de monitoreo [43]–[47].

HW-028	DHT22	GUVA-S12SD	FC-28	GY-LSM6DS3
Goteo de lluvia	Digital combinada	Fotodiodo Schottky	Conductividad	3 ejes acelerómetro 3 ejes giroscopio
3.3-5 V DC	3.3-5 V DC	3.3-5 V DC	3.3-5 V DC	1.71-3.6 V DC
>15 mA	Máx. 2.5 mA	5 mA	15 mA	0.9 mA (normal) 25.1 mA (alto)
2 M Ω (seco) -100 k Ω (mojado)	0-100 % RH -40 – 80 °C	240 – 370 nm	0-100%	± 2 g, ± 4 g, ± 8 g, ± 16 g ± 125 °/s, ± 245 °/s, ± 500 °/s, $\pm$ 1000 °/s, ± 2000 °/s

Potenciómetro integrado	±2 % RH ± 0.5 °C	±1 UV index	Potenciómetro integrado	1.7 muestras/s	kilo
VCC GND Digital Analógico	VCC DATA GND	VCC GND OUT	VCC GND Digital Analógico	VCC GND SDA SCL SDO/SA0 INT1/INT2	
Detecta lluvia mediante un divisor resistivo	Usa un termistor NTC para temperatura y un polímero capacitivo para humedad. La conversión de la señal digital de 16 bits a valores físicos sigue	La corriente fotogenerada (J_{UV}) depende de la irradiación (E_{UV}) y la longitud de onda (λ): $J_{UV} = \int_{200\text{ nm}}^{370\text{ nm}} S(\lambda) E(\lambda) d\lambda [A]$ Donde $S(\lambda) = 0.15 A/W$ es la sensibilidad espectral.	La conductividad del suelo (σ) se relaciona con la humedad volumétrica (θ).	La inclinación (ϕ) se estima combinando acelerómetro (a_x, a_y, a_z) y giroscopio (ω_x, ω_y), mediante un filtro complementario	
V_{out} $= V_{cc} \frac{R_s}{R_s + R_{fijo}}$ donde $R_{fijo} = 10\text{ k}\Omega$ y $V_{cc} = 3.3V$	$Temp.$ $= \frac{VALOR_{raw}}{65536} \times 165 - 40 [^{\circ}C]$ $Hum.$ $= \frac{VALOR_{raw}}{65536} \times 100 [\% HR]$	Amplificador de transimpedancia V_{UV} $= -J_{UV}R_f + V_{offset}$ Con $R_f = 1\text{ M}\Omega$ y $V_{offset} = 0.5\text{ V}$ para evitar salidas negativas.	$\sigma = \frac{1}{R_s L} \left(\frac{A}{d} \right)$ Donde $L=5.5\text{ cm}$ la longitud de los electrodos, $A=0.8\text{ cm}^2$ área y $d=2\text{ cm}$ la separación.	$\phi_{filtrado} = \alpha(\phi_{prev} + \omega_x \cdot \Delta t) + (1 - \alpha) \cdot \arctan\left(\frac{a_y}{a_z}\right)$	
Comparado con otros sensores del mismo tipo, evita falsos positivos	Comparado con un termopar tipo K, tiene un error	Relación UV Index y voltaje $UV\ Index = 12.5V_{UV} - 1.2$	Pese a ser un resistivo y poseer una menor precisión del ±7% en	Error RMS de 0.3° en el eje X y 0.4° en el eje Y, además de un consumo de	

por polvo y posee un bajo costo.	cuadrático medio de 0.4 °C en un rango de 10°C a 60 °C, comparado con SHT31, posee una mayor latencia 2s, pero consume menos corriente 1.2 mA.	Cubre UV-B (290–320 nm), crítico para estudios de impacto ambiental, mientras que sensores como el VEML6075 solo detectan UV-A.	comparación con otros es inmune a interferencias por sales disueltas.	energía 0.9 mA en comparación con un MPU6050.
----------------------------------	--	---	---	---



Figura 11. Sensor HW-028.



Figura 12. Sensor DHT22.



Figura 13. Sensor GUV4-S12SD.

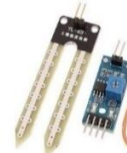


Figura 14. Sensor FC-28.

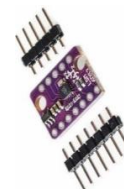


Figura 15. Sensor GY-LSM6D3.

3.8.3 Módulo ESP32 LoRa V3.2

Tabla 7. Características del módulo [48].

Características	Especificaciones
Núcleo	Dual-core Xtensa 32-bit LX7, hasta 240 MHz
Voltaje de alimentación	1.71 – 3.6 V DC
Memoria ROM sleep	384 KB
Memoria SRAM	12 KB + 16 KB RTC SRAM
Flash	8 MB
Conectividad	2.4 GHz IEEE 802.11 b/g/n, hasta 150 Mbps Bluetooth 5 (LE)

Interfaces	3×UART, 2×SPI, 2×I2C, 2×I2S, USB OTG, RMT, PWM, TWAI (CAN)
GPIO	45 pines programables
ADC	2×SAR de 12 bits, hasta 20 canales
Módulo LoRa	SX1262
Rango de frecuencia	150 – 960 MHz
Potencia de TX	Hasta +22 dBm
Sensibilidad RX	Hasta –148 dBm
Protocolos	LoRa (SF5–SF12) FSK/GMSK
Rango de frecuencia	150 – 960 MHz
Potencia de TX	Hasta +22 dBm

Se usa este firmware para la adquisición de datos mediante el uso de multitareas con FreeRTOS, priorizando la recolección de datos, transmisión LoRa y gestión energética, la ESP32 entra en modo Deep Sleep entre ciclos, activados por un temporizador interno RTC. La corriente de sleep (I_{sleep}) y activa (I_{active}) determinan su autonomía.

$$I_{avg} = I_{sleep} + \frac{t_{active}}{t_{cycle}} I_{active} \quad (17)$$



Figura 16. Microcontrolador ESP32 con módulo LoRa.

3.8.4 Raspberry pi 5

Fue seleccionada por su capacidad para manejar cargas de trabajo complejas en el Edge, es usada como nodo central para el procesamiento de los datos y la aplicación de los diferentes modelos predictivos, su uso se basa en su gran capacidad para ejecutar simultáneamente procesos como Broker MQTT, y la ejecución de los algoritmos predictivos con una carga de

CPU<20%, además es 2.5x más rápido que su predecesora en tareas de procesamiento numérico, si se la compara con otro hardware como Jetson Nano su costo es menor y tiene un consumo ajustado de 7W en comparación de 10W, además frente a servidores cloud posee una baja latencia inferior a 3s y posee autonomía en entornos sin internet.

Tabla 8. Características de la Raspberry Pi 5 [49].

Características	Especificaciones
Procesador	Broadcom BCM2712 (ARM Cortex-A76 de 64 bits, 4 núcleos a 2.4 GHz).
GPU	VideoCore VII (soporte OpenGL ES 3.1, Vulkan 1.2, 4Kp60 H.265).
RAM	8 GB LPDDR4X-4267 SDRAM
Almacenamiento	MicroSD UHS-I (hasta 2 TB) + PCIe 2.0 x1 para conectores NVMe M.2
Conectividad	Wi-Fi 6 (802.11ax) y Bluetooth 5.0 (BLE).
	Gigabit Ethernet (2.4 Gbps con soporte PoE+).
	2 puertos USB 3.0 (5 Gbps) + 2 puertos USB 2.0.
Expansión	GPIO de 40 pines
Dimensiones	85.6 × 56.5 mm
Alimentación	USB-C, 5 V / 5 A (25 W)



Figura 17. Raspberry Pi 5.

3.8.5 Enlace Inalámbrico

Parámetros de transmisión

Tabla 9. Configuración del SX1262.

Parámetro	Valor
Frecuencia	915 MHz
Bandwidth	125 KHz
CR	4/5

Spreading Factor	10
Potencia del transmisor	17 dBm
Sensibilidad del receptor	-137 dBm

El tiempo en el aire del paquete se obtiene:

$$t_{air} = \frac{2^{SF}}{BW} \times \frac{Payload + 8}{CR} = \frac{2^{10}}{125 \times 10^3} \times \frac{32 + 8}{\frac{4}{5}} = 1.2 \text{ s} \quad (18)$$

El payload representa al comprimido de bytes que se realiza para enviar la trama de datos este caso 32 bytes mediante codificación CBOR.

El margen de enlace para la red de sensores se puede calcular de la siguiente manera:

$$Margen = P_{TX} - P_{RX_min} - 20\log_{10}(d) - 20\log_{10}(f) + 147.55 \quad (19)$$

La red de sensores implementó el protocolo LoRa para asegurar comunicación de largo alcance en implementación en campo y bajo consumo energético, adaptándose a las necesidades de zonas remotas con riesgo de deslizamientos.

3.8.6 Diseño del prototipo de red de sensores

El diseño de la red de sensores se fundamentó en una arquitectura escalable, utilizando el protocolo de comunicación LoRa para garantizar cobertura en áreas extensas con bajo consumo de energía en casos de implementación en sitio. La red se compone de nodos finales equipados con sensores de temperatura, humedad ambiental, precipitación, humedad del suelo y radiación solar, integrados a microcontroladores con módulos LoRa para la transmisión inalámbrica. Los nodos se conectan a un gateway central, que envía los datos hacia un Broker MQTT, para asegurar la autonomía, los nodos poseen baterías recargables.

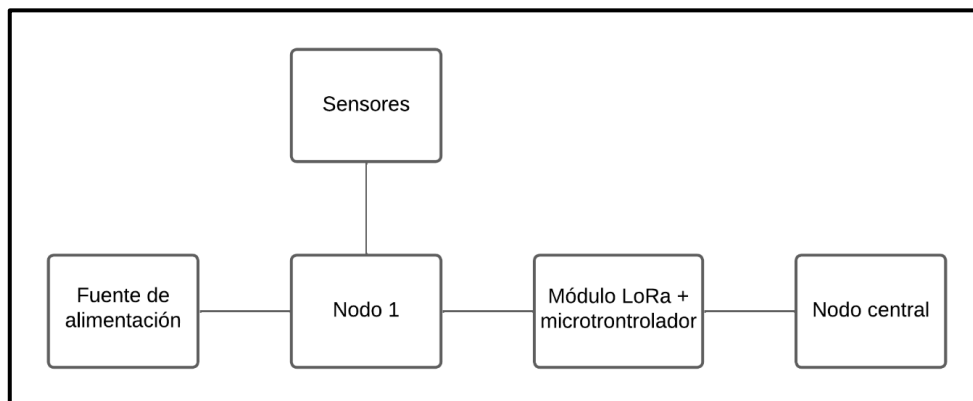


Figura 18. Diagrama del diseño de un nodo de recolección.

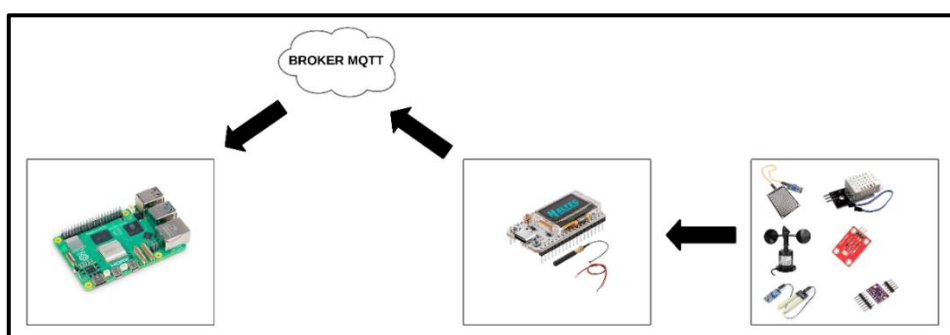


Figura 19. Diagrama de recolección de transmisión y recolección de datos.

3.9 Fase 3: Implementación del prototipo y evaluación en ambiente controlado

3.9.1 Diseño del ambiente controlado

Estructura física y materiales

La construcción del ambiente controlado se hizo en dimensiones de 100 cm de largo, 150 cm de ancho y 80 cm de altura, las paredes y base son de tabla contrachapada de 18 mm de grosor con un recubrimiento de pintura epóxica para proteger contra la humedad y degradación, se diseñó para el aislamiento parcial de variables externas mientras se permite la exposición controlada a luz solar.

Condiciones ambientales controladas

Posee una exposición directa a la luz solar en horarios de 8:00 a 16:00, tiene una ventilación natural debido a que la cara frontal de la caja fue removida para que exista una buena ventilación y simule corrientes de aire reales sin la necesidad de incorporar ventiladores

externos, la temperatura interna oscila entre los 18-35 °C y la humedad relativa del ambiente entre los 40-85%.

Suelo y sustrato

El tipo de suelo con el que fue construido el ambiente controlado es de tipo franco-limoso-arcilloso con unas proporciones aproximadas de 51% arena, 34% limo y 15% arcilla, la distribución por el ambiente se hizo homogéneamente tamizando para eliminar grumos demasiado gruesos.

Variables controladas y no controladas

Controlada

Se usa un sistema que consta de un aspersor de jardín con boquilla de nebulización para simular intensidad de lluvia 10 mm/h de agua sobre todo el ambiente controlado, la pendiente también se considera una variable controlada ya que puede ser modificada al incrementar o disminuir la inclinación de la zona estudiada.

No controladas

Estas variables no pueden ser controladas, pero son monitoreadas ya que la interacción entre ellas genera detonantes para que ocurran deslizamientos de tierra, estas son temperatura ambiental, humedad relativa del ambiente, humedad del suelo y la radiación UV.



Figura 20. Vista previa del ambiente controlado.

En la figura 20 se muestra una toma aérea del ambiente controlado donde se implementará la red de sensores y el sistema de simulación de lluvia, para la recolección de las variables ambientales y el posterior envío para su análisis.

3.9.2 Implementación del prototipo en el ambiente controlado

Calibración individual de sensores

HW-028

Para calibrar el sensor de precipitación es fundamental relacionar su salida eléctrica con la cantidad de agua aplicada, para esto se usó un sistema de riego controlado para aplicar agua con intensidad conocida aproximadamente 10 mm/h ajustando el sensor para definir el umbral de sensibilidad deseado, donde mayor sensibilidad detecta gotas más pequeñas.

Tabla 10. Calibración del sensor de precipitación.

Voltaje	Intensidad real	Intensidad ajustada
0.10	0.00	0.70
0.80	1.20	1.38
1.50	2.80	2.64
2.30	5.00	4.79
3.00	7.50	7.28
3.80	10.00	10.28
4.50	15.00	14.57

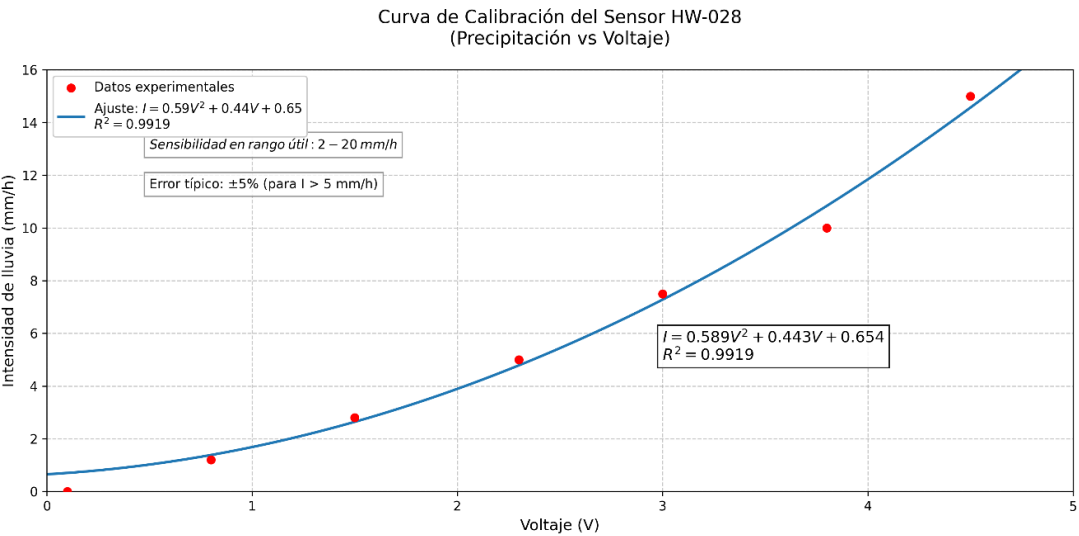


Figura 21. Curva de calibración del sensor HW-028.

La figura 21 muestra la relación entre la salida de voltaje del sensor y la intensidad de lluvia real, permitiendo convertir las lecturas eléctricas en valores físicos de precipitación.

DHT22

Tanto la temperatura como la humedad relativa que son medidas con este sensor fueron calibrados tomando como referencia un dispositivo de grado militar (TC-700PRO) para medir la calidad del aire que nos ayudó a ver y ajustar la precisión de nuestro sensor obteniendo la tabla 11 para la temperatura:

Tabla 11. Comparación de temperatura.

Temp. Real °C	Temp DHT °C	Error Abs.	Error real %
12.3	13.02	0.72	5.85
15.4	15.50	0.1	0.67
16.7	16.52	0.18	1.09
20.3	20.27	0.03	0.15
24.6	24.46	0.14	0.57
25.4	25.67	0.27	1.08
30.3	30.13	0.17	0.55
33.7	34.12	0.42	1.26

Y para la humedad se obtuvo la tabla 12:

Tabla 12. Comparación de humedad.

HR Real %	HR DHT %	Error Abs.	Error real %
17.8	16.43	1.37	7.67
19.6	16.86	2.74	13.67
33.6	33.79	0.19	0.58
40.6	38.99	1.61	3.97
45.8	44.54	1.26	2.75
50.6	49.37	1.23	2.43
64.6	65.88	1.28	1.98
69.9	69.17	0.73	1.04

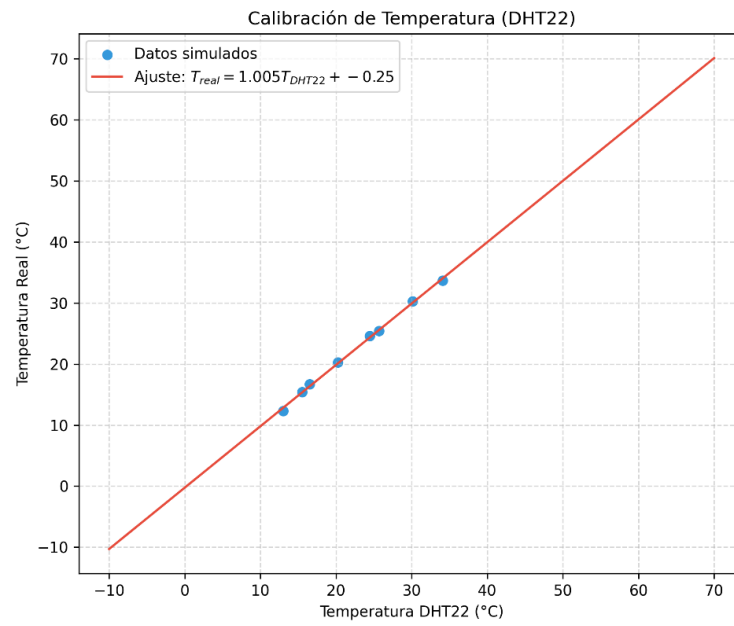


Figura 22. Calibración de temperatura.

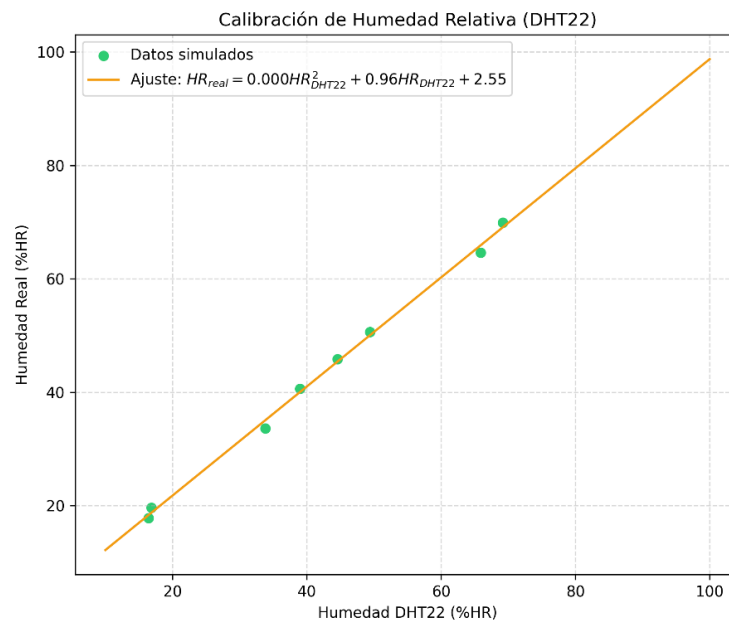


Figura 23. Calibración de humedad.

GUVA-S12SD

Para calibrar el sensor de radiación UV se comparó con un luxómetro profesional, obteniendo así los siguientes valores que se representan en la tabla 13:

Tabla 13. Comparación del índice UV.

Voltaje V	Índice UV real	Índice UV ajustado	Error Abs.	Error Relativo
0.10	0.00	-0.42	0.42	0.00%
0.30	1.00	0.89	0.11	11.0%
0.50	2.00	2.20	0.20	10.1%
0.70	3.00	3.52	0.52	17.2%
1.00	5.00	5.48	0.48	9.7%
1.20	7.00	6.80	0.20	2.9%
1.50	9.00	8.77	0.23	2.6%
1.80	11.00	10.73	0.27	2.4%
2.10	13.00	12.70	0.30	2.3%

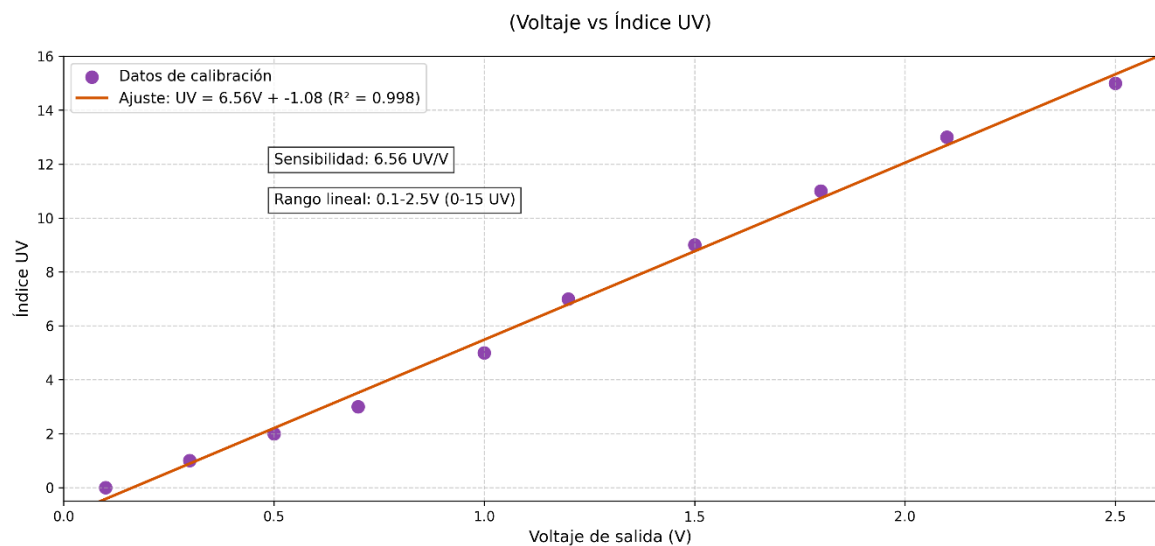


Figura 24. Calibración del sensor GUV-A-S12SD.

FC-28

Para calibrar el sensor de humedad del suelo, se comparó los datos recolectados en el tipo de suelo que se usó para el ambiente controlados y compararlos con valores de humedad reales obtenidos de estudios de suelo previos, obteniendo la tabla 14.

Tabla 14. Comparación de la humedad del suelo.

Voltaje V	Humedad Real %	Humedad ajustada %	Error Abs.	Error Relativo
3.1	5	4.62	0.4	7.7%
2.8	10	9.64	0.4	3.6%
2.6	15	14.93	0.1	0.4%
2.3	20	21.67	1.7	8.4%
1.9	30	30.13	0.1	0.4%
1.6	40	39.27	0.7	1.8%
1.2	50	49.26	0.7	1.5%
0.9	60	60.15	0.1	0.2%
0.6	70	70.30	0.3	0.4%

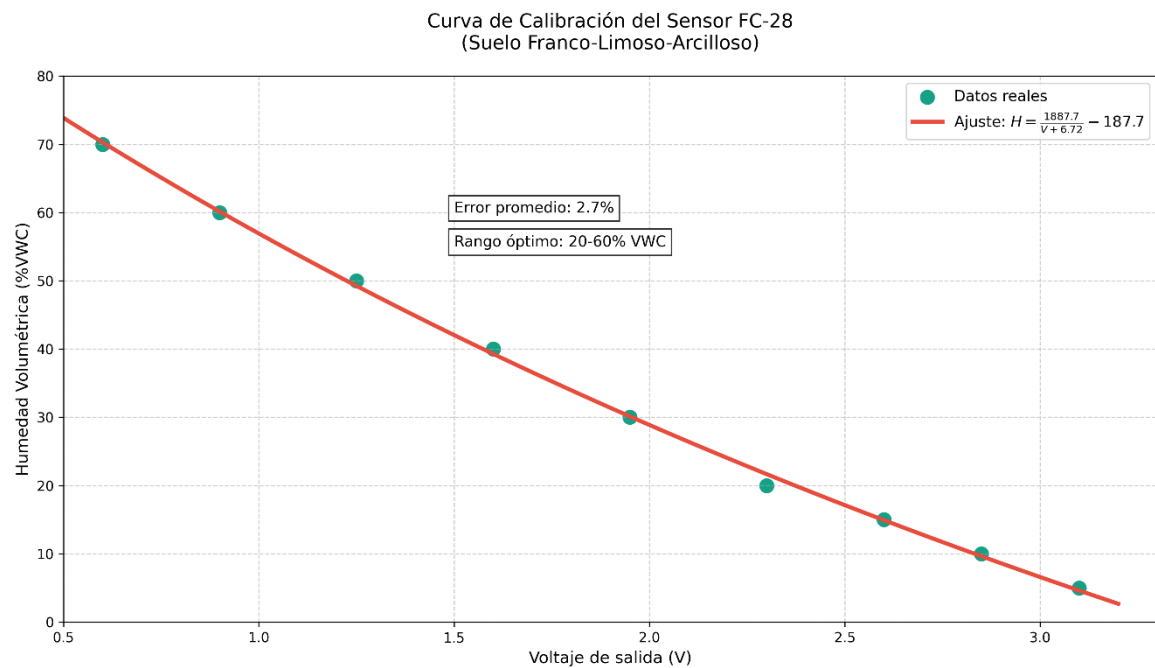


Figura 25. Curva de calibración del sensor FC-28.

Tras completar la calibración individual de los sensores para garantizar la precisión y fiabilidad de sus mediciones, se procedió a desplegar dos nodos de monitoreo y un nodo que se encarga de la comunicación en el entorno controlado diseñado para simular condiciones que desencadenen un deslizamiento de tierra.

Este ambiente, construido bajo parámetros estandarizados y replicables, permitió evaluar la eficacia del modelo predictivo mediante la inducción controlada de variables críticas asociadas a fenómenos de inestabilidad geotécnica, tales como cambios en la humedad del suelo y precipitación controlada.

Los dos nodos se instalaron en puntos clave del entorno simulado, integrando los sensores calibrados para capturar datos en tiempo real. Cada nodo operó bajo el protocolo de comunicación Lora, sincronizado para garantizar la coherencia temporal y espacial de las mediciones. El sistema de adquisición de datos registró variables a intervalos de 30 segundo, almacenando la información y luego enviándola para su posterior análisis.

SISTEMA DE PREDICCIÓN DE DESLIZAMIENTOS		
Resultados de Predicción		
Modelo	Probabilidad (%)	Estado
Regresión	96.61%	ALERTA ROJA
SVM	11.93%	NORMAL
Random Forest	62.17%	ALERTA AMARILLA
Red Neuronal	0.00%	NORMAL

Última actualización: 11:55:47

Figura 26. Ventana de sistema de predicción.

CAPÍTULO IV.

4.1 Resultados y discusión

4.2 Fase 1: Recolección de datos históricos y selección de algoritmos

4.2.1 Preprocesamiento de los datos

El preprocesamiento de datos es un paso fundamental para el entrenamiento de un modelo predictivo, ya que esta es la materia prima con la que el modelo aprenderá patrones y anomalías para dar una predicción más precisa. En la tabla 15 se detallan las variables predictivas que van a ser utilizadas en el entrenamiento del modelo mientras que al conjunto de datos se le agregara una variable objetivo 0 o 1 que representa la ocurrencia o no del evento, a su vez la variable precipitación fue subdividida en precipitación acumulada en 5, 10, 15, 20, 25 y 30 días dando un total de 19 variables predictivas.

Tabla 15. Variables de entrenamiento.

Variable	Clase	Unidad de medida	Tipo	Campo
Landslide	Binaria	-	Objetivo	Riesgo
Prc. Acum. 5	Cuantitativa	mm	Predictora	Meteorológica
Prc. Acum. 10	Cuantitativa	mm	Predictora	Meteorológica
Prc. Acum. 15	Cuantitativa	mm	Predictora	Meteorológica
Prc. Acum. 20	Cuantitativa	mm	Predictora	Meteorológica
Prc. Acum. 25	Cuantitativa	mm	Predictora	Meteorológica
Prc. Acum. 30	Cuantitativa	mm	Predictora	Meteorológica
Temp. Min.	Cuantitativa	°C	Predictora	Meteorológica
Temp. Max.	Cuantitativa	°C	Predictora	Meteorológica
Temp. Mean	Cuantitativa	°C	Predictora	Meteorológica
Rel. Humidity	Cuantitativa	%	Predictora	Meteorológica
ETo	Cuantitativa	mm	Predictora	Meteorológica
Soil moisture	Cuantitativa	%	Predictora	Meteorológica
Solar radiation	Cuantitativa	W/m ²	Predictora	Meteorológica
Land cover	Cualitativa	-	Predictora	Uso de suelo
Soil	Cualitativa	-	Predictora	Uso de suelo
Tree	Cuantitativa	%	Predictora	Cobertura vegetal
Non tree	Cuantitativa	%	Predictora	Cobertura vegetal
Bare	Cuantitativa	%	Predictora	Cobertura vegetal
Slope	Cuantitativa	°	Predictora	Geomorfológica

4.2.2 Ponderación de variables cualitativas

Como se observa en la sección anterior las variables land cover y soil tienen un carácter cualitativo, por lo que es importante convertir sus valores a números antes de ingresarlos al entrenamiento del modelo predictivo. Para esta ponderación se usa un enfoque mixto combinando datos observados, principios hidrológicos y ajustes expertos con el objetivo de generar señales numéricas que generen el riesgo intrínseco de cada categoría.

Land Cover: Se refiere a la capa física de materiales que cubren la superficie terrestre, incluyendo elementos naturales y antrópicos. Describe qué hay en el terreno, como bosques, cuerpos de agua, áreas urbanas, cultivos, matorrales o suelos desnudos. A diferencia del land use, que se enfoca en la función humana del territorio el land cover clasifica la composición biofísica observable. Su clasificación se muestra en la tabla 16, de acuerdo con estudios de campo e imágenes obtenidas del satélite MODIS.

Tabla 16. Clasificación de land cover de acuerdo con el satélite MODIS [50].

Code	Land Cover
0	Agua
1	Bosque perenne de aciculares
2	Bosque perenne de hoja ancha
3	Bosque de acículas caducifolio
4	Bosque de hoja ancha caducifolio
5	Bosque mixto
6	Matorral cerrado
7	Matorral abierto
8	Sabana arbórea
9	Sabana
10	Praderas
11	Humedales permanentes
12	Tierras de cultivo
13	Áreas urbanas
14	Mosaicos cultivo/vegetación
15	Nieve y hielo
16	Áreas áridas o rocosas

Usando la tabla 16 se transformó esta variable en un índice de riesgo numérico usando ponderaciones basadas en frecuencia de deslizamientos en las zonas estudiadas y ajustes expertos.

Soil: La variable soil indica la textura que tiene el suelo, en su clasificación se usó el triángulo de USDA que asigna cada textura a cada uno de los grupos que se observan en la tabla, usando el porcentaje de arcilla, arena y limo para su clasificación.

Tabla 17. Clasificación según el triángulo de USDA [51].

Característica	Textura
$Sa \geq 85\%; Cl \leq 10\%; Si \leq 10\%$	Arenoso
$70\% \leq Sa < 85\%; Cl < 15\%; Si < 30\%$	Arenoso franco
$43\% \leq Sa < 85\%; Cl < 20\%; Si < 50\%$	Franco arenoso
$23\% \leq Sa < 52\%; 7\% \leq Cl < 27\%; 28\% \leq Si < 50\%$	Franco
$Cl < 27\%; 50\% \leq Si < 88\%; Sa < 52\%$	Franco limoso
$Si \geq 88\%; Cl < 12\%$	Limoso
$45\% \leq Cl < 60\%; 20\% < Sa < 45\%$	Franco arcilloso arenoso
$27\% \leq Cl < 40\%; 20\% < Sa < 45\%$	Franco arcilloso
$27\% \leq Cl < 40\%; Si > 40\%$	Franco limoso arcilloso
$Cl \geq 35\%; Sa > 45\%$	Arcilloso arenoso
$Cl \geq 40\%; Si > 40\%$	Arcilloso limoso
$Cl \geq 40\%; Sa < 45\%; Si < 40\%$	Arcilloso

De acuerdo con la tabla 17 se usó la fórmula, para calcular el índice de riesgo de cada textura de suelo.

$$Riesgo = 0.60(\% \text{ Arcilla}) + 0.3(\% \text{ Limo}) + 0.1(\% \text{ Arena}) \quad (20)$$

4.2.3 Análisis del sistema predictivo con regresión logística

Tabla 18. Análisis del algoritmo Regresión Logística.

Métrica	Valor	Interpretación
Accuracy	0.9425	El 94.25 % de las predicciones (tanto positivas como negativas) son correctas. Este alto valor indica buen desempeño global, aunque puede enmascarar desequilibrios entre clases (mayoría vs. minoría).
F1-score	0.9405	Media armónica entre precisión y recall. Un 0.9405 refleja un excelente equilibrio entre falsos positivos y falsos negativos, crucial cuando la clase “deslizamiento” es minoritaria.
Recall	0.9071	El modelo detecta correctamente el 90.71 % de los deslizamientos reales, dejando escapar el 9.29 % (falsos negativos), lo cual es relevante en aplicaciones de gestión de riesgos.
Log Loss	0.2124	Mide la desviación de las probabilidades pronosticadas respecto a los eventos reales. Un valor de 0.2124 indica buena calibración, aunque inferior al RF (0.1561), penalizando más las predicciones confiadamente erróneas.
Brier Score	0.04528	Error cuadrático medio (MSE) de las probabilidades. Un 0.0453 denota que las probabilidades del modelo están razonablemente bien calibradas, si bien algo peor que el RF (0.0409).
AUC-ROC	0.9795	Área bajo la curva ROC. Con 0.9795, el modelo discrimina muy eficazmente entre sitios con y sin deslizamiento ($\approx 97.95\%$ de parejas correctamente ordenadas).
AUC-PR	0.9850	Área bajo la curva Precisión-Recall. Un 0.9850 refleja que, incluso en un conjunto desbalanceado, el modelo mantiene alta precisión en la detección de la clase minoritaria sin aumentar demasiado las falsas alarmas.

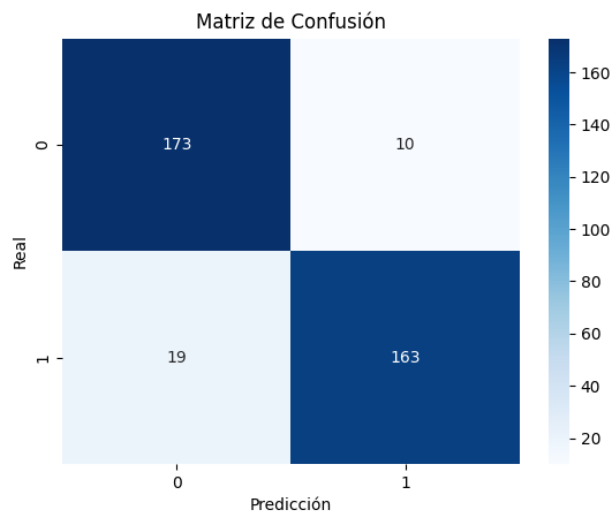


Figura 27. Matriz de confusión del algoritmo Regresión Logística.

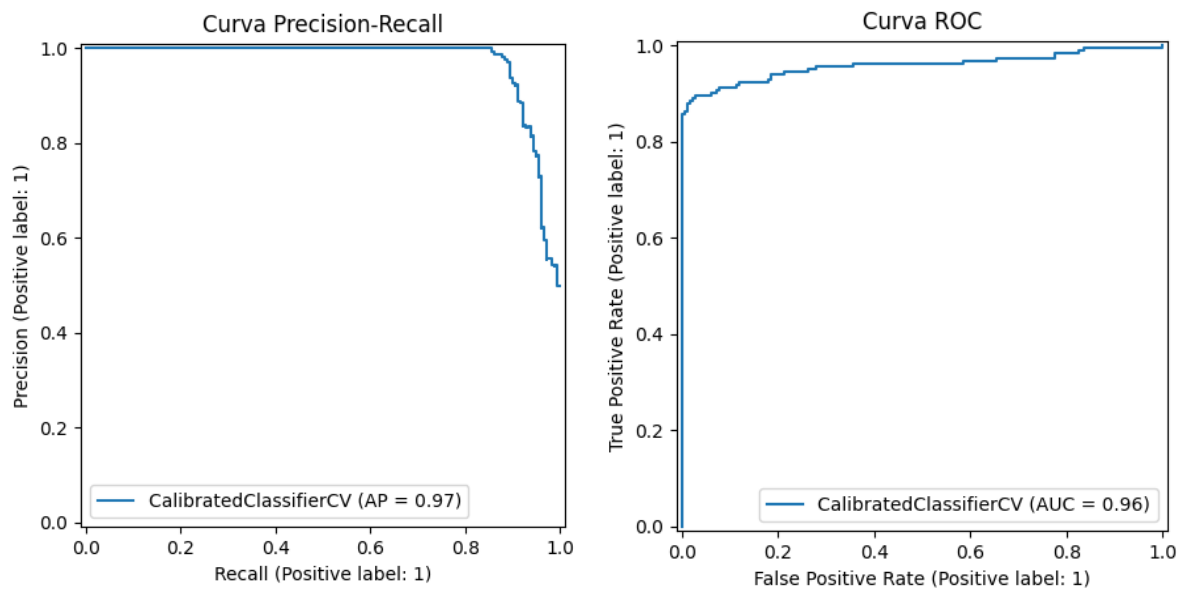


Figura 28. Curva de precisión-Recall y ROC para regresión logística.

4.2.4 Análisis del sistema predictivo con SVM

Tabla 19. Análisis del algoritmo SVM

Métrica	Valor	Interpretación
Accuracy	0.9699	Mide la proporción de clasificaciones (positivas y negativas) correctamente asignadas. Un 96.99 % indica que el modelo sólo se equivoca en el 3.01 % de los casos.
F1-score	0.9694	Media armónica de precisión y recall; valora por igual falsos positivos y negativos. Un 0.9694 refleja un equilibrio casi perfecto entre detectar deslizamientos y minimizar falsas alarmas.
Recall	0.9508	También llamado TPR, mide el % de deslizamientos reales correctamente identificados. Con 95.08 %, sólo un 4.92 % de los deslizamientos verdaderos quedan sin detectar.
Log Loss	0.1116	Penaliza las predicciones demasiado confiadas y erróneas. Un valor cercano a 0 indica excelente calibración; 0.1116 es muy bajo, por lo que las probabilidades pronosticadas están bien ajustadas.
Brier Score	0.0255	Es el MSE de las probabilidades pronosticadas. Un 0.0255 indica una calibración sobresaliente (probabilidades muy cercanas a las frecuencias reales).
AUC-ROC	0.9902	Área bajo la curva ROC. Con 0.9902, el modelo discrimina casi perfectamente entre clases en todos los umbrales posibles.
AUC-PR	0.9932	Área bajo la curva Precisión-Recall; indicada para conjuntos desbalanceados. Un 0.9932 muestra un desempeño casi perfecto en la detección de la clase minoritaria sin inflar las falsas alarmas.

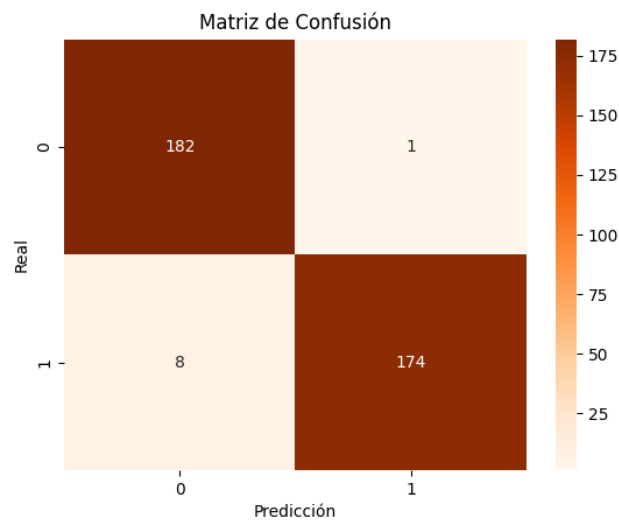


Figura 29. Matriz de confusión de SVM.

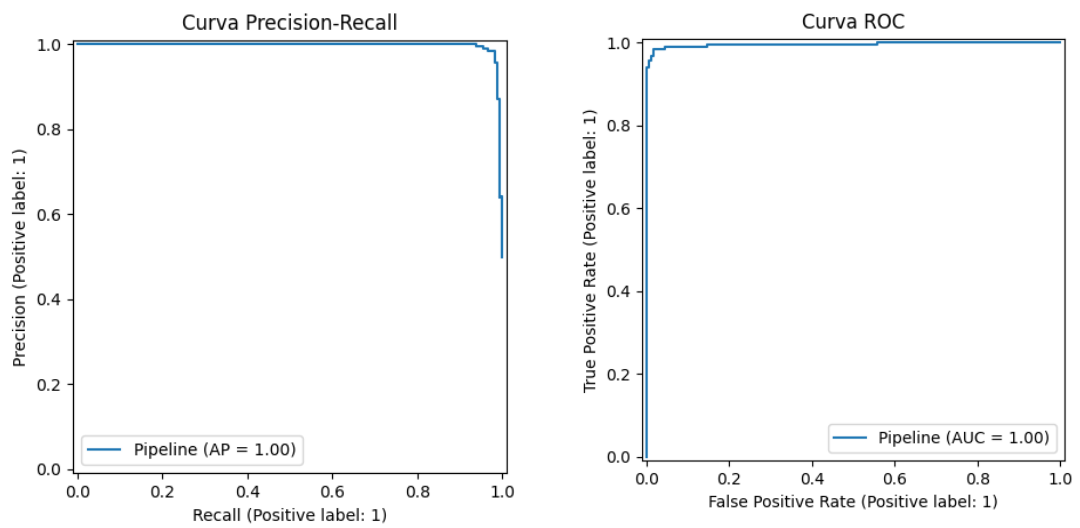


Figura 30. Curva de Precisión-Recall y ROC para SVM.

4.2.5 Análisis del sistema predictivo con Random Forest

Tabla 20. Análisis del algoritmo Random Forest

Métrica	Valor	Interpretación
Accuracy	0.9534	Proporción de predicciones correctas (positivas y negativas) sobre el total. Un 95.34 % indica muy buen desempeño global, aunque esta métrica puede inflarse si hay desbalance de clases.
F1-score	0.9518	Media armónica entre precisión y recall. Con 0.9518 refleja que el modelo mantiene un excelente equilibrio entre falsos positivos y falsos negativos, fundamental cuando la clase “deslizamiento” es minoritaria.
Recall	0.9180	Tasa de verdaderos positivos: porcentaje de deslizamientos reales correctamente detectados. Un 91.80 % significa que solo el 8.20 % de los eventos críticos quedan sin identificar (falsos negativos), aspecto determinante en gestión de riesgos.
Log Loss	0.1561	“Cross-entropy” aplicada a probabilidades: penaliza predicciones muy confiadas y erróneas. Un valor de 0.1561 (cercano a 0) indica que, en promedio, las probabilidades asignadas están muy próximas a los resultados reales.
Brier Score	0.04093	Error cuadrático medio de las probabilidades pronosticadas. Con 0.0409, el Random Forest exhibe una calibración sobresaliente, pues las probabilidades predichas casi coinciden con las frecuencias reales.
AUC-ROC	0.9869	Área bajo la curva ROC: mide la capacidad de separar ambas clases en todos los umbrales. Un AUC-ROC de 0.9869 significa que, dada una pareja aleatoria, el modelo ordena correctamente la positiva el 98.69 % de las veces.
AUC-PR	0.9902	Área bajo la curva Precisión-Recall: especialmente relevante en casos desbalanceados. Un 0.9902 evidencia un desempeño casi perfecto para la clase minoritaria, manteniendo alta precisión sin disparar falsas alarmas.

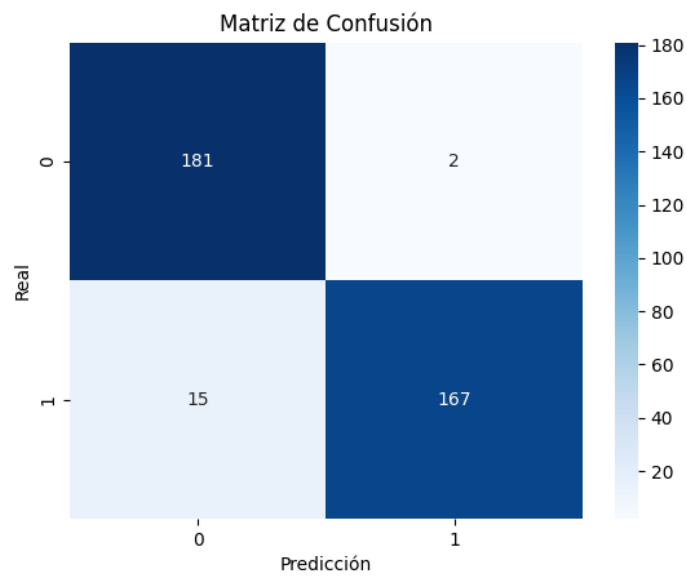


Figura 31. Matriz de confusión de Random Forest.

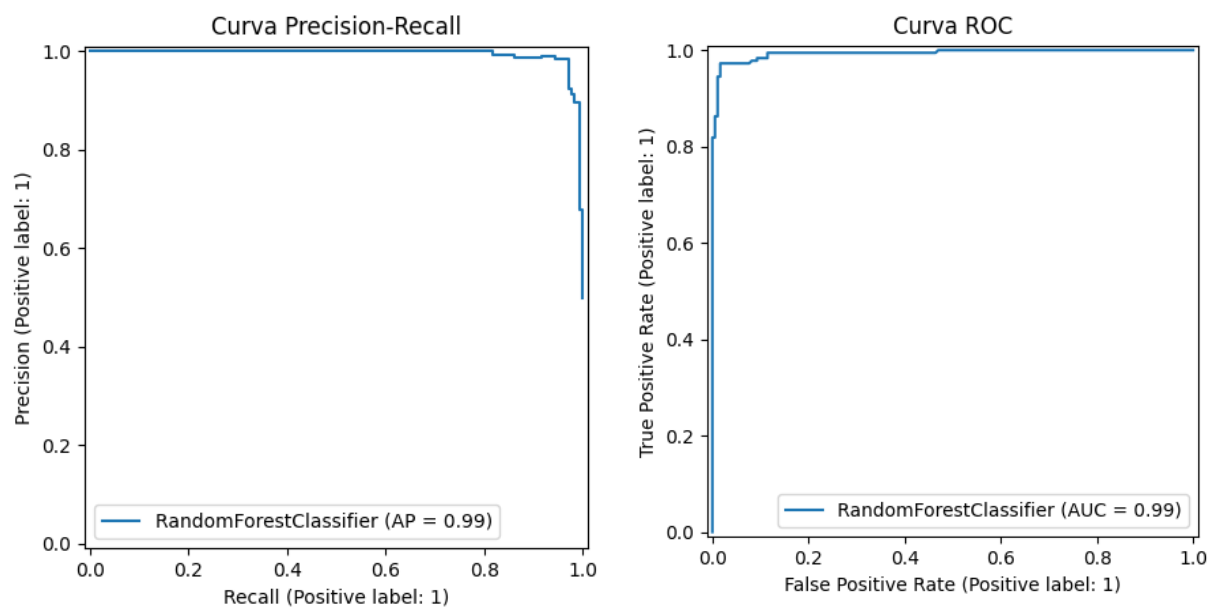


Figura 32. Curva Precisión- Recall y ROC de Random Forest.

4.2.6 Análisis del sistema predictivo con redes neuronales (MLP)

Tabla 21. Análisis de algoritmo de Redes Neuronales (MLP).

Métrica	Valor	Interpretación
Accuracy	0.9726	Proporción de predicciones correctas. Un 97.26 % indica que solo el 2.74 % de los casos fueron mal clasificados, lo que refleja un desempeño global muy alto.
F1-score	0.9724	Media armónica de precisión y recall. Con 0.9724 muestra un equilibrio excelente entre falsos positivos y falsos negativos, esencial en datasets desbalanceados como el de deslizamientos.
Recall	0.9792	Tasa de verdaderos positivos. Detecta el 97.92 % de los deslizamientos reales, dejando apenas un 2.08 % de falsos negativos, lo cual es crítico para minimizar omisiones de eventos de riesgo.
Log Loss	0.0992	Penaliza fuertemente predicciones probabilísticas muy confiadas y erróneas. Un valor por debajo de 0.1 indica calibración excepcional: las probabilidades asignadas reflejan muy bien la incertidumbre real de los eventos.
Brier Score	0.0199	MSE de las probabilidades pronosticadas. Al ser tan bajo (≈ 0.02), confirma una calibración superior: las probabilidades predichas coinciden estrechamente con las frecuencias observadas.
AUC-ROC	0.99099	Área bajo la curva ROC. Un valor de 0.99099 refleja que, dada una muestra con deslizamiento y otra sin, la red neuronal escogerá correctamente la positiva el 99.10 % de las veces.
AUC-PR	0.99371	Área bajo la curva Precisión-Recall. Con 0.99371, incluso en un contexto de clases desbalanceadas, la red mantiene alta precisión en la detección de la clase minoritaria sin aumentar falsas alarmas.

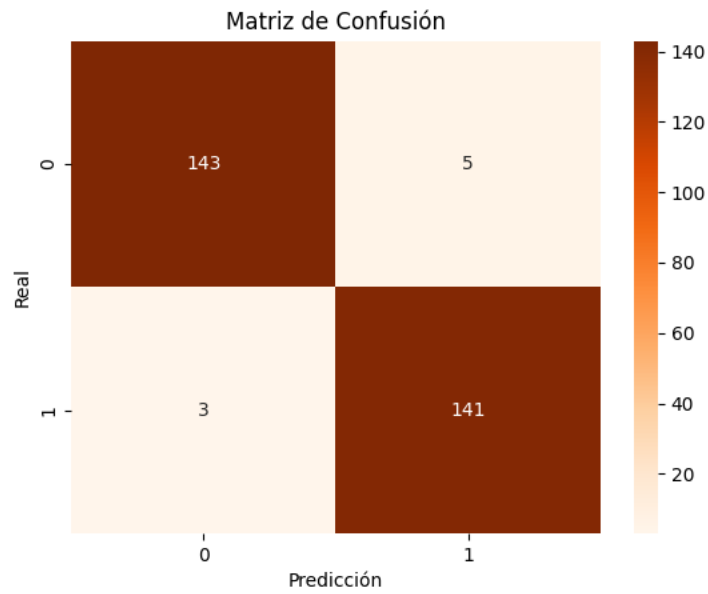


Figura 33. Matriz de confusión de Redes Neuronales (MLP).

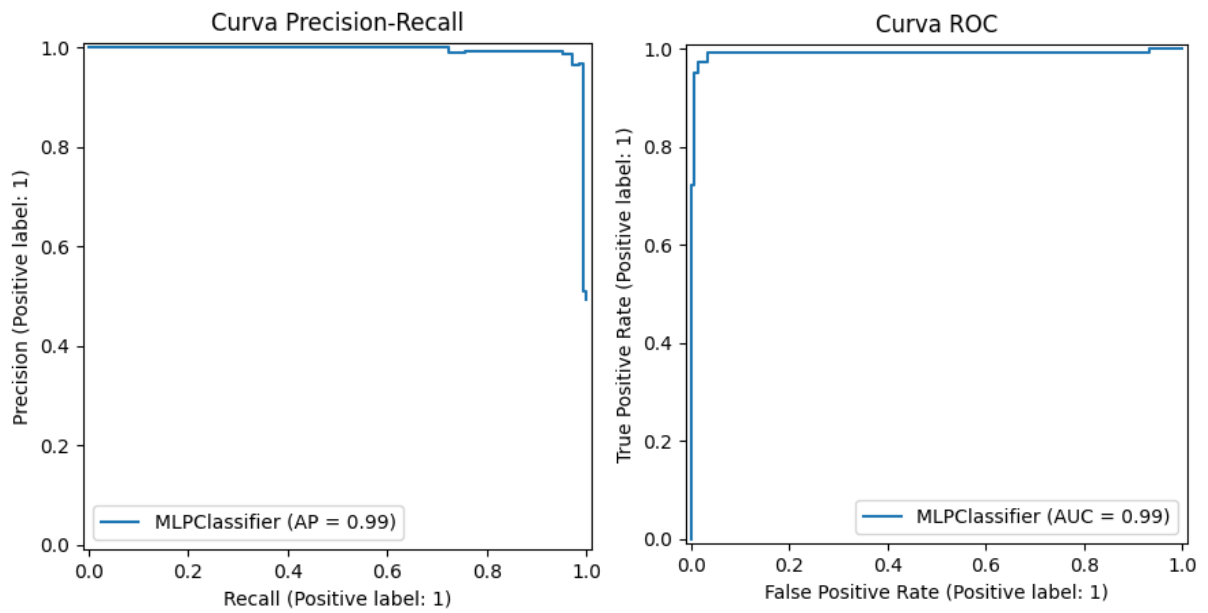


Figura 34. Curva Precisión-Recall y ROC de Redes Neuronales (MLP).

4.3 Fase 2: Diseño del prototipo y elección de la tecnología de comunicación

Tras la elección de los sensores se diseñó una PCB donde irán incorporados el microcontrolador y los sensores que recolectarán las variables ambientales, el diseño de la PCB se muestra en las figuras 35, 36.

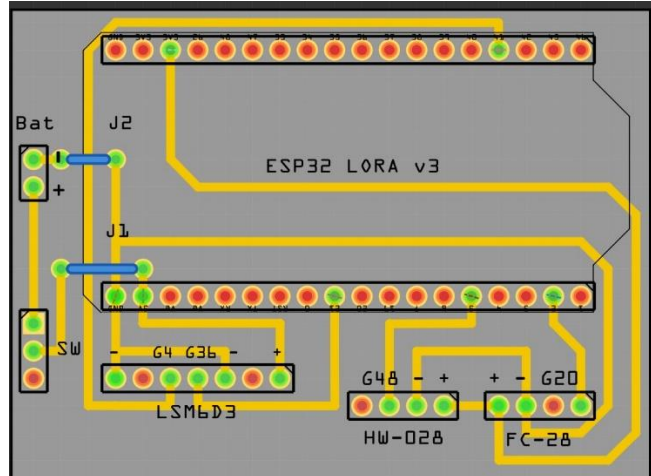


Figura 35. Diseño de la primera PCB.

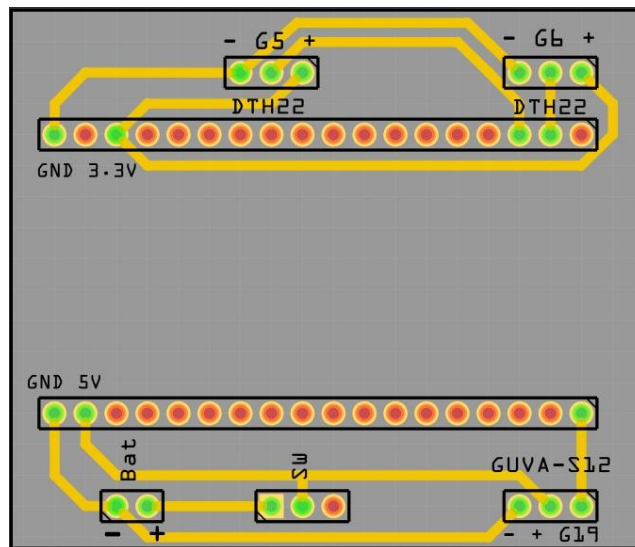


Figura 36. Diseño de la segunda PCB.

La batería que alimenta las PCB es una batería de LiPo de 600 mAh, lo que teóricamente proporcionará una duración de:

$$I_{avg} = I_{sleep} + \frac{t_{active}}{t_{cycle}} I_{active} = 10 \mu A + \frac{1.2s}{30s} (92mA) = 3.69mA$$

$$Duración = \frac{600mAh}{3.69mA} = 163 \text{ horas} = 7 \text{ días}$$

Lo que permite 7 días de autonomía.

4.3.1 Desempeño del enlace LoRa

El enlace LoRa fue puesto a prueba en distintas distancias y evaluado por métricas como, tasa de éxito y RSSI (sensibilidad)

Tabla 22. Desempeño del enlace LoRa.

Distancia m	Tasa de éxito %	RSSI dBm
5	99.9	-35
20	98.7	-42
40	97.9	-48
60	97.6	-53
80	96.9	-58
100	96.8	-63

La tasa de éxitos para todos los casos supera los 96% a 100 metros de distancia se pierde un 3% de paquetes debido a interferencias por señales Wi-Fi, y otras señales.

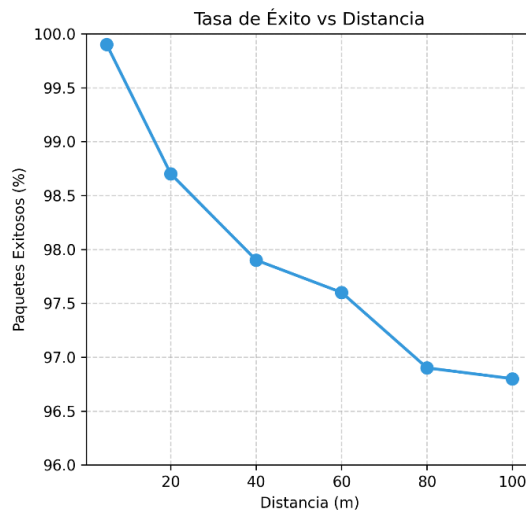


Figura 37. Tasa de éxito vs distancia.

Para entender la figura 37 se compara el RSSI teórico con el medido:

$$FSPL(dB) = 20\log_{10}(d) + 20\log_{10}(f) - 147.55 \quad (21)$$

Por último, para calcular el RSSI:

$$RSSI = P_{TX} - FSPL \quad (22)$$

Los resultados se muestran en la tabla 23:

Tabla 23. Comparación de RSSI teórico y medido.

FSPL	RSSI Teórico	RSSI Medido
45.7	-28.7	-35.3
57.7	-40.7	-42.5
63.7	-46.7	-48.8
65.7	-48.7	-53.2
69.7	-52.7	-58.1
71.7	-54.7	-63.9

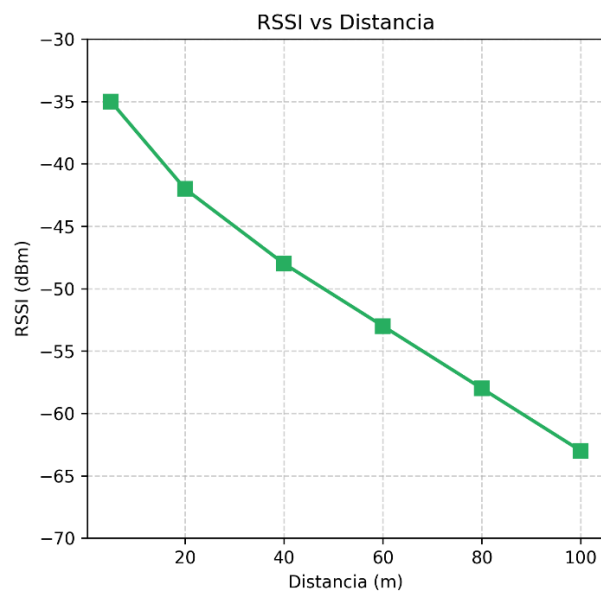


Figura 38. RSSI vs distancia.

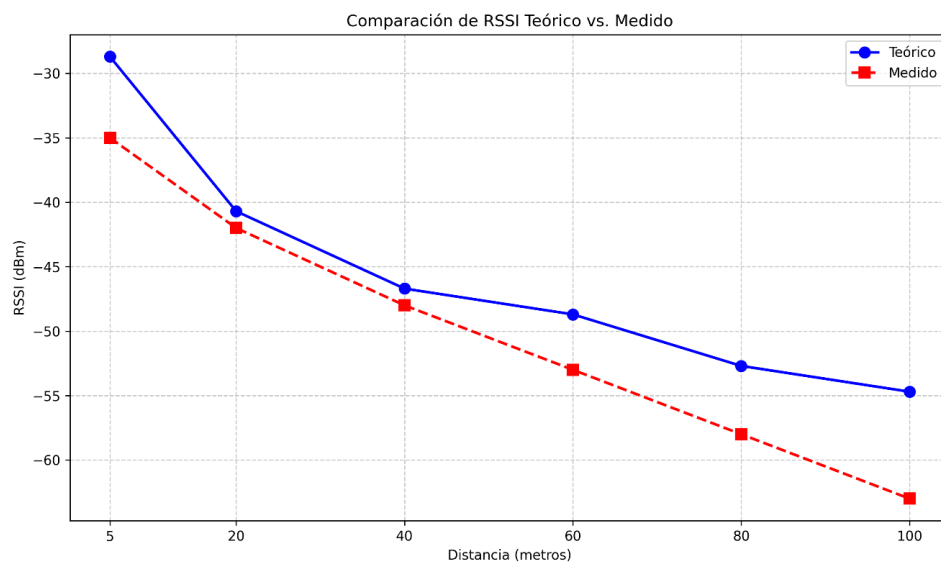


Figura 39. Comparación entre RSSI teórico y medido.

4.4 Fase 3: Implementación del prototipo y evaluación en ambiente controlado

4.4.1 Análisis entre modelos

Se realizaron 32 pruebas en el ambiente controlado con los 4 algoritmos propuestos anteriormente obteniendo lo siguiente:

Tabla 24. Comparación entre modelos.

Regresión Logística			SVM			Random Forest			Redes Neuronales		
%	Resul.	Eval.	%	Resul.	Eval.	%	Resul.	Eval.	%	Resul.	Eval.
95	1	Acierto	71	1	Fallo	70	1	Fallo	96	1	Acierto
96	1	Acierto	91	1	Acierto	97	1	Acierto	70	0	Acierto
77	0	Acierto	72	0	Acierto	76	0	Acierto	87	1	Acierto
76	0	Acierto	79	0	Acierto	70	0	Acierto	66	0	Acierto
88	0	Fallo	93	1	Acierto	91	0	Fallo	86	1	Acierto
94	1	Acierto	91	0	Fallo	67	1	Fallo	66	0	Acierto
90	0	Fallo	94	1	Acierto	69	0	Acierto	98	1	Acierto
77	1	Fallo	89	1	Acierto	95	1	Acierto	79	0	Acierto
77	0	Acierto	90	1	Acierto	97	1	Acierto	78	0	Acierto
78	0	Acierto	76	0	Acierto	68	0	Acierto	81	0	Fallo
78	1	Fallo	83	1	Acierto	69	0	Acierto	68	1	Fallo
76	1	Fallo	94	1	Acierto	92	1	Acierto	75	0	Acierto
85	1	Acierto	80	0	Fallo	76	0	Acierto	69	0	Acierto
78	0	Acierto	65	0	Acierto	65	0	Acierto	93	1	Acierto
75	0	Acierto	84	0	Fallo	82	1	Acierto	88	1	Acierto
97	1	Acierto	81	1	Acierto	73	0	Acierto	67	0	Acierto
84	1	Acierto	92	1	Acierto	91	1	Acierto	71	0	Acierto
82	1	Acierto	75	0	Acierto	84	1	Acierto	66	0	Acierto
76	1	Fallo	77	0	Acierto	77	1	Fallo	84	1	Acierto
95	1	Acierto	97	1	Acierto	74	0	Acierto	82	0	Fallo
78	0	Acierto	80	1	Acierto	80	0	Fallo	85	0	Fallo
95	1	Acierto	74	1	Fallo	77	0	Acierto	96	1	Acierto
94	1	Acierto	93	1	Acierto	69	0	Acierto	78	0	Acierto
77	1	Fallo	67	0	Acierto	91	1	Acierto	70	0	Acierto
76	0	Acierto	98	1	Acierto	70	0	Acierto	79	0	Acierto
85	0	Fallo	89	0	Fallo	94	1	Acierto	74	1	Fallo
78	0	Acierto	87	1	Acierto	85	1	Acierto	80	0	Fallo
95	1	Acierto	83	1	Acierto	80	0	Fallo	92	1	Acierto
76	1	Fallo	93	1	Acierto	86	1	Acierto	91	1	Acierto
79	1	Fallo	75	0	Acierto	94	0	Fallo	75	0	Acierto
81	0	Fallo	74	0	Acierto	84	1	Acierto	66	0	Acierto
79	0	Acierto	81	0	Fallo	94	1	Acierto	65	0	Acierto

Se realizó una prueba de normalidad con las siguientes hipótesis a cada algoritmo para identificar si las predicciones siguen una distribución normal obteniendo los siguientes resultados:

- **Hipótesis nula:** Los porcentajes de las predicciones de los algoritmos de Machine Learning siguen una distribución normal.
- **Hipótesis alternativa:** Los porcentajes de las predicciones de los algoritmos de Machine Learning no siguen una distribución normal.

Tabla 25. Prueba de normalidad Shapiro-Wilk

Prueba de normalidad			
Algoritmo	Shapiro-Wilk		
	Estadístico	GL	p-value
Regresión Logística	0.81625	32	0.00008363
SVM	0.9018	32	0.036856
Random Forest	0.91997	32	0.02076
Redes Neuronales	0.93558	32	0.04615

Ya que el p-value de todos los algoritmos presentan un valor menor a 0.05 ($p\text{-value} < 0.05$) se descarta la hipótesis nula, por lo que se usa una prueba no paramétrica para analizar los resultados obtenidos de los algoritmos. Para ello se plantea las siguientes hipótesis:

- **Hipótesis Nula:** El tipo de algoritmo de Machine Learning utilizado en la red de sensores inteligente no tiene un efecto significativo sobre la probabilidad estimada de ocurrencia de un deslizamiento de tierra en un ambiente controlado.
- **Hipótesis Alternativa:** El tipo de algoritmo de Machine Learning utilizado en la red de sensores inteligente tiene un efecto significativo sobre la probabilidad estimada de ocurrencia de un deslizamiento de tierra en un ambiente controlado.

Se realizó un análisis de desempeño de los 4 algoritmos de Machine Learning aplicados a la predicción de deslizamientos de tierra en un ambiente controlado. Los resultados obtenidos se presentan en la siguiente tabla:

Tabla 26. Aciertos y fallos.

Algoritmo	Aciertos	Fallos	% Acierto	IC Inferior	95% IC 95% Superior
Regresión Logística	17	15	53.13%	36.45%	69.13%
SVM	25	7	78.13%	61.25%	88.98%
Random Forest	25	7	78.13%	61.25%	88.98%
Redes Neuronales	29	3	90.63%	75.78%	96.76%

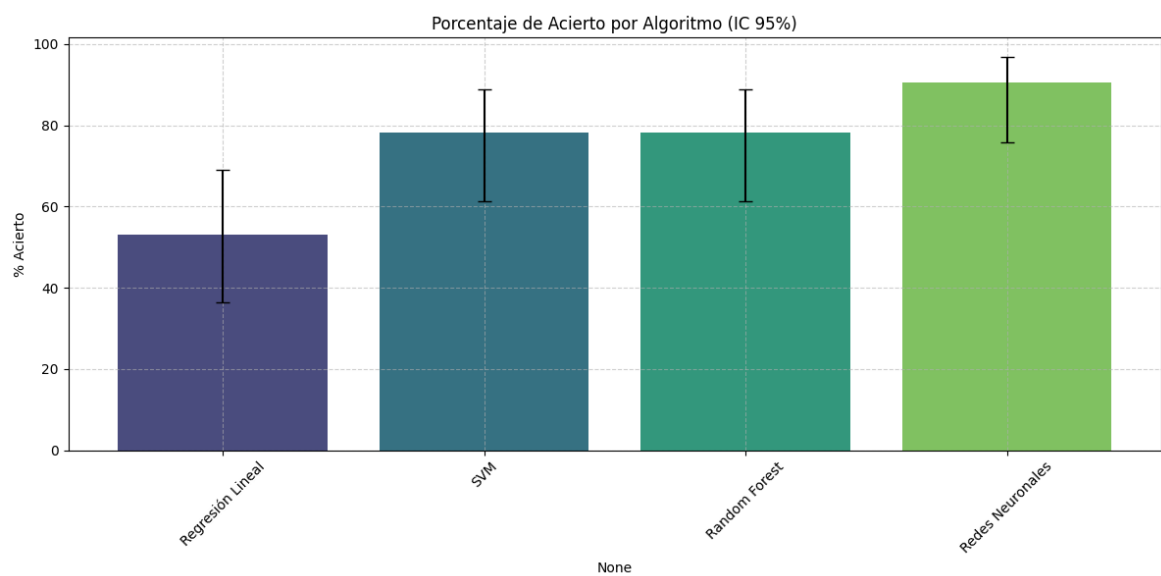


Figura 40. Porcentaje de aciertos por algoritmo.

Para determinar si existen diferencias significativas en el desempeño de los cuatro algoritmos, se aplicó una prueba de Friedman. Los resultados fueron:

- **Estadístico H:** 12.3243
- **P-value:** 0.0064

Dado que p-value es menor que 0.05 se rechaza la hipótesis nula con lo que se concluye que existen diferencias estadísticamente significativas entre los modelos en términos de precisión.

Según el rango medio obtenido de la prueba se clasifica de mejor a peor los 4 algoritmos, un rango medio menor indica mejor desempeño relativo.

Tabla 27. Rango medio obtenido al hacer el estudio de Friedman.

Modelo	Rango medio	Grupos
Redes Neuronales	2.1875	Grupo 1
SVM	2.4375	Grupo 2
Random Forest	2.4375	
Regresión Logística	2.9375	Grupo 3

Para validar este ranking se usó una prueba de Nemenyi (post-hoc), para hacer comparaciones pareadas entre los algoritmos, obteniendo la siguiente tabla:

Tabla 28. Test de Nemenyi (post-hoc).

Grupo 1	Grupo 2	mean-diff	p-value	lower	upper	Diferencia
Redes Neuronales	Regresión Logística	-0.375	0.000532	-0.6469	-0.1031	Verdadero
	SVM	-0.225	0.004550	-0.3969	-0.1469	Verdadero
	Random Forest	-0.225	0.004550	-0.3969	-0.1469	Verdadero
	Regresión Logística	-0.125	0.04081	-0.2719	-0.0219	Verdadero
SVM	Random Forest	0.0	1.0000	-0.2719	0.2719	Falso
Random Forest	Regresión Logística	-0.125	0.04081	-0.2719	-0.0219	Verdadero

Con estas tablas se interpreta que el algoritmo de redes neuronales es significativamente mejor que el resto mientras el algoritmo de regresión logística es significativamente peor que el resto y finalmente que los algoritmos de SVM y Random Forest no presentan una diferencia significativa, por lo que se puede rechazar la hipótesis nula y decir que “El tipo de algoritmo de Machine Learning utilizado en la red de sensores inteligente tiene un efecto significativo sobre la probabilidad estimada de ocurrencia de un deslizamiento de tierra en un ambiente controlado.”

CAPÍTULO V.

5.1 Conclusiones y recomendaciones

5.1.1 Conclusiones

En el presente trabajo se diseñó un sistema de monitoreo y predicción de deslizamientos de tierra mediante la convergencia de tecnologías IoT y técnicas de ML, estableciendo un paradigma tecnológico innovador para la gestión de riesgos geológicos. La arquitectura propuesta basada en una red de sensores inalámbricos autónomos, demostró su viabilidad técnica en un ambiente controlado, destacando su escalabilidad, eficiencia energética limitada al ambiente controlado y capacidad de procesamiento en tiempo real.

El entrenamiento de algoritmos de ML basado en datos históricos permitió identificar patrones clave para la detección temprana de deslizamientos de tierra. Los resultados confirman que modelos avanzados, como redes neuronales MLP, ofrecen mayor precisión y capacidad de adaptación a la variabilidad ambiental, destacando su utilidad para fortalecer los sistemas de alerta y reducir riesgos geológicos.

La selección estratégica de componentes como la placa ESP32 LoRa V3.2, programada en MicroPython, permitió la recolección de variables ambientales (humedad del suelo, precipitación, temperatura, entre otras), mientras que el protocolo LoRa en la banda de 915 MHz aseguró una transmisión de datos estable y de largo alcance con interferencia mínima. La Raspberry Pi 5, actuando como gateway y nodo de procesamiento, no solo centralizó la información mediante HiveMQTT, sino que ejecutó modelos predictivos de forma local, reduciendo la dependencia de infraestructura cloud y optimizando la latencia en la generación de alertas.

El análisis comparativo de los algoritmos reveló umbrales críticos para la selección de modelos en aplicaciones geotécnicas. Si bien la regresión logística mostró una precisión teórica elevada (94%), su bajo índice de aciertos reales (57%) expuso limitaciones inherentes a su simplicidad paramétrica, incapaz de capturar no linealidades en fenómenos multifactoriales como los deslizamientos. En contraste, el SVM (Support Vector Machine), con un 97% de precisión y 78% de aciertos, evidenció superioridad al emplear kernels para proyectar datos a espacios de alta dimensionalidad, resolviendo problemas de separabilidad

no lineal. Sin embargo, fue el clasificador MLP (Multilayer Perceptron), una red neuronal de tres capas entrenada con backpropagation, el que emergió como el modelo óptimo al alcanzar un 90% de aciertos reales, a pesar de una precisión ligeramente inferior (97%).

5.1.2 Recomendaciones

Para garantizar la efectividad operativa y el impacto social del sistema propuesto, se recomienda priorizar la optimización energética de los nodos mediante circuitos híbridos baterías recargables y paneles solares, y algoritmos de muestreo adaptativo que ajusten la frecuencia de captura según condiciones de riesgo, a su vez para la implementación en campo usar sensores especializados acompañados de dispositivos de medición de deformación y redundancia en la adquisición de datos para mitigar fallas.

En el modelado predictivo, implementar arquitecturas híbridas que fusionen las fortalezas del MLP (altos aciertos reales) y SVM (precisión teórica), utilizando técnicas de ensemble learning, y adoptar frameworks de IA explicable para transparentar las decisiones del modelo ante autoridades y comunidades. En validación y escalabilidad, ejecutar pilotos graduales en zonas de riesgo real, iniciando en laderas instrumentadas bajo supervisión geotécnica, seguido de despliegues en áreas rurales con monitoreo remoto, integrando datos satelitales para correlacionar variables locales con movimientos tectónicos regionales.

En investigaciones futuras se espera explorar el uso de redes neuronales temporales (LSTM) para capturar patrones estacionales, así como acoplar el sistema a modelos físicos de infiltración de suelos, asegurando que la tecnología complemente y no reemplace el conocimiento experto en geología.

BIBLIOGRAFÍA

- [1] Córdova Miranda, Tito Leoncio, Rivera Herrera, Hugo Javier, Mamani Coaquira, Jeanpier Billy, Salgado Villanueva, William Eduardo, y Maquera Aguilar, Mauricio Andree, “Sistema de monitoreo y alerta temprana para deslizamientos de tierra por lluvias en una quebrada de Tacna: Implementación de una red inalámbrica de sensores”, *Ing. INVESTIGA*, vol. 5, oct. 2023, doi: 10.47796/ing.v5i0.831.
- [2] M. S. Al-Batah, M. Salem Alzboon, H. Solayman Migdadi, M. Alkhasawneh, y M. Alqaraleh, “Advanced Landslide Detection Using Machine Learning and Remote Sensing Data”, *Data Metadata*, vol. 3, oct. 2024, doi: 10.56294/dm2024.419.
- [3] L. R. ALVARADO NEIRA, “Evaluación del diseño de un prototipo de monitoreo Geo-IoT, aplicado en potenciales deslizamientos de tierra”, Universidad Salesiana, Cuenca, 2024.
- [4] B. Rodriguez, “Predicción de deslizamientos aplicando técnicas de aprendizaje automático”, Universidad de Alicante, 2023.
- [5] P. I. Gómez, “Modelo para el mantenimiento predictivo de segmentos especiales de vía”, UNIVERSIDAD DE CANTABRIA, 2023.
- [6] J. M. F. Rodríguez, “Detección de deslizamientos de tierra utilizando técnicas de inteligencia artificial IA (2022)”, 2022, doi: 10.11144/Javeriana.10554.61566.
- [7] A. F. Riaño Quintanilla, “Desarrollo de un sistema de alertas tempranas por movimientos en masa basado en inteligencia artificial para el departamento de Cundinamarca, Colombia”, Universidad de los Andes, Bogotá, 2023.
- [8] O. A. Gallardo Chávez, “Evaluación de susceptibilidad por fenómenos de remoción en masa mediante machine learning. Caso de estudio: vía Papallacta-Baeza”, Universidad Central del Ecuador, Quito, 2022.
- [9] M. G. ORTIZ SOSA, “DESARROLLO DE UNA RED DE SENSORES INALÁMBRICOS UTILIZANDO TECNOLOGÍA LORA PARA EL MONITOREO DE UN SISTEMA”, UNIVERSIDAD POLITÉCNICA SALESIANA, Quito, 2020.
- [10] E. Posadinu, “Inteligencia artificial para la prevención de desastres naturales y resolución de problemas ambientales. Un estudio aplicado al archipiélago canario”, Universidad Abierta de Cataluña, Cataluña, 2022.
- [11] D. Veloz-Cherrez, M. Ojeda, D. Moreno, y J. Del Pozo, “Susceptibility to Translational Landslides in Ecuador Caused by Changes in Electrical Permittivity of Andepts Soils Using Software-Defined Radar for Detection”, *Earth*, vol. 5, núm. 4, pp. 670–689, oct. 2024, doi: 10.3390/earth5040035.
- [12] J. A. Quintanilla Armijo, “IMPLEMENTACIÓN DE UN PROTOTIPO DE SISTEMA DE ALERTA PARA DETECCIÓN EN TIEMPO REAL DE DESLIZAMIENTOS DE TIERRA”, ESCUELA POLITÉCNICA NACIONAL, Quito, 2021.
- [13] C. Mella, “Un deslave en el centro de Ecuador deja al menos siete fallecidos y cientos de afectados”, *El País*. Consultado: el 11 de junio de 2025. [En línea]. Disponible en: <https://elpais.com/internacional/2023-03-27/un-deslave-en-el-centro-de-ecuador-deja-al-menos-16-fallecidos-y-cientos-de-afectados.html>

- [14] A. Ticonipa Gutierrez, “Sistema de Monitoreo para la detección de deslizamientos en zonas de riesgo mediante algoritmos de triangulación y una red de sensores inalámbricos basado en topología Mesh”, abr. 2021.
- [15] E. Tarbuck y F. Lutgens, *Essentials of Geology*, vol. 13th. PEARSON EDUCACION, 2018. Consultado: el 11 de junio de 2025. [En línea]. Disponible en: <https://www.pearson.com/en-us/subject-catalog/p/essentials-of-geology/P2000000007002/9780134857299>
- [16] O. Ghorbanzadeh, Y. Xu, P. Ghamisi, M. Kopp, y D. Kreil, “Landslide4Sense: Reference Benchmark Data and Deep Learning Models for Landslide Detection”, *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–17, 2022, doi: 10.1109/TGRS.2022.3215209.
- [17] S. P. Fernández y E. A. Izquierdo, “Applied machine learning and predictive modeling techniques for soil profiling. Practical CPTU application.”, 2021.
- [18] D. Mesa, “MODELO PREDICTIVO PARA IDENTIFICAR ZONAS POTENCIALES DE DESLIZAMIENTOS INFLUENCIADOS POR PRECIPITACIÓN EN LA CIUDAD DE MEDELLÍN”, UNIVERSIDAD EIA, 2022.
- [19] P. Nolasco Ramírez, “Aplicación de Machine Learning para pronóstico de desplazamiento de lluvias usando imágenes del radar de lluvias de UDEP”, Universidad de Piura, Piura, 2023.
- [20] F. R. Ugenio Yépez, “Implementación de Técnicas de Machine Learning para la predicción del impacto de humedad y temperatura sobre el peso corporal de los cuyes”, UNIVERSIDAD NACIONAL DE CHIMBORAZO, Riobamba, 2023.
- [21] C. Wang, W. Guo, K. Yang, X. Wang, y Q. Meng, “Real-Time Monitoring System of Landslide Based on LoRa Architecture”, *Front. Earth Sci.*, vol. 10, p. 899509, jun. 2022, doi: 10.3389/feart.2022.899509.
- [22] M. Ragnoli, M. Scarsella, A. Leoni, G. Ferri, y V. Stornelli, “Wireless Sensor Network-Based Rockfall and Landslide Monitoring Systems: A Review”, *Sensors*, vol. 23, núm. 16, p. 7278, ago. 2023, doi: 10.3390/s23167278.
- [23] A. Sharma, R. Mohana, A. Kukkar, V. Chodha, y P. Bansal, “An ensemble learning-based experimental framework for smart landslide detection, monitoring, prediction, and warning in IoT-cloud environment”, *Environ. Sci. Pollut. Res. Int.*, vol. 30, núm. 58, pp. 122677–122699, dic. 2023, doi: 10.1007/s11356-023-30683-6.
- [24] A. Sgarabotto *et al.*, “Evaluating the use of smart sensors in ground-based monitoring of landslide movement with laboratory experiments”, el 27 de noviembre de 2023, *Landslides and Debris Flows Hazards*. doi: 10.5194/egusphere-2023-2596.
- [25] L. F. Espinoza Trejo, “IMPLEMENTACIÓN Y ANÁLISIS DE DESEMPEÑO DE UNA RED DE SENSORES INALÁMBRICOS INTELIGENTES PARA SU APLICACIÓN EN MONITOREO Y CONTROL DE VARIABLES.”, INSTITUTO TECNOLÓGICO DE CHIHUAHUA II, Chihuahua, 2021.
- [26] A. Goercke y J. Diego, “Desarrollo de un prototipo IoT para la detección temprana de incendios forestales”, Universidad Salesiana, Cuenca, 2024.
- [27] C. A. BRUSIL CRUZ, “ANÁLISIS COMPARATIVO ENTRE APRENDIZAJE SUPERVISADO Y APRENDIZAJE SEMI-SUPERVISADO PARA LA

CLASIFICACIÓN DE SEÑALES SÍSMICAS VULCANOLÓGICAS DEL VOLCÁN COTOPAXI”, ESCUELA POLITÉCNICA NACIONAL, Quito, 2020.

- [28] Z. Ma, G. Mei, y F. Piccialli, “Machine learning for landslides prevention: a survey”, *Neural Comput. Appl.*, vol. 33, núm. 17, pp. 10881–10907, sep. 2021, doi: 10.1007/s00521-020-05529-8.
- [29] I. Ullah *et al.*, “An Integrated Approach of Machine Learning, Remote Sensing, and GIS Data for the Landslide Susceptibility Mapping”, *Land*, vol. 11, núm. 8, p. 1265, ago. 2022, doi: 10.3390/land11081265.
- [30] A. L. Achu *et al.*, “Machine-learning based landslide susceptibility modelling with emphasis on uncertainty analysis”, *Geosci. Front.*, vol. 14, núm. 6, p. 101657, nov. 2023, doi: 10.1016/j.gsf.2023.101657.
- [31] G. Titti, G. N. Napoli, C. Conoscenti, y L. Lombardo, “Cloud-based interactive susceptibility modeling of gully erosion in Google Earth Engine”, *Int. J. Appl. Earth Obs. Geoinformation*, vol. 115, p. 103089, dic. 2022, doi: 10.1016/j.jag.2022.103089.
- [32] G. Agboola, L. H. Beni, T. Elbayoumi, y G. Thompson, “Optimizing landslide susceptibility mapping using machine learning and geospatial techniques”, *Ecol. Inform.*, vol. 81, p. 102583, jul. 2024, doi: 10.1016/j.ecoinf.2024.102583.
- [33] S. Akosah, I. Gratchev, D.-H. Kim, y S.-Y. Ohn, “Application of Artificial Intelligence and Remote Sensing for Landslide Detection and Prediction: Systematic Review”, *Remote Sens.*, vol. 16, núm. 16, p. 2947, ago. 2024, doi: 10.3390/rs16162947.
- [34] R. López, R. Márquez, L. Urrego, y G. Cuervo, “MODELO PARA EL PRONOSTICO DE LA AMENAZA POR DESLIZAMIENTOS EN TIEMPO REAL”, *Inst. Hidrol. Meteorol. Estud. Ambient.*, pp. 1–8.
- [35] B. M. Das, Ed., *Geotechnical engineering handbook*. Ft. Lauderdale, FL: J. Ross Pub, 2011.
- [36] D. Cruden y D. Varnes, “Landslide Types and Processes, Special Report”, *Spec. Rep. Natl. Res. Counc. Transp. Res. Board*, ene. 1996.
- [37] Highland, Lynn y Brobrowsky, Peter, *The landslide handbook-A guide to understanding landslide*. U.S. Geological Survey. 2008.
- [38] A. Joshi, J. Grover, D. Prasanna Kanungo, y R. Kumar Panigrahi, “Real-time Landslide Monitoring, Detection and Early Warning System for Tangni Landslide”, *SSRN Electron. J.*, 2019, doi: 10.2139/ssrn.3511001.
- [39] P. Rahu, M. Altaf, y B. Manoj, “A real time IoT based System Prediction and Monitoring of Landslides”, *IFJANS Int. JOURNAL OF FOOD Nutr. Sci.*, 2024, Consultado: el 11 de junio de 2025. [En línea]. Disponible en: https://www.researchgate.net/publication/382454631_A_real_time_IoT_based_System_Prediction_and_Monitoring_of_Landslides
- [40] M. Hakimi Zohari, I. B. Mohamad Syahmi, y O. Abu Hassan, “Landslide Monitoring System Based on Internet of Things (IoT)”, *Int. J. Sci. Eng. Appl.*, vol. 13, mar. 2024, doi: 10.7753/IJSEA1304.1006.
- [41] GADMA, “Alcaldía de Alausí”. Consultado: el 11 de junio de 2025. [En línea]. Disponible en: <https://alausi.gob.ec/index.php>

- [42] FAO, “FAO Climate Info Tool”. Consultado: el 11 de junio de 2025. [En línea]. Disponible en: <https://aquastat.fao.org/climate-information-tool/>
- [43] “Sensor para detección de gotas de lluvia”, Novatronic. Consultado: el 11 de junio de 2025. [En línea]. Disponible en: <https://novatronicec.com/index.php/product/sensor-para-deteccion-de-gotas-de-lluvia/>
- [44] “SENSOR DHT22”, Grupo Electrostore. Consultado: el 11 de junio de 2025. [En línea]. Disponible en: <https://grupoelectrostore.com/shop/sensores/sensor-dht22/>
- [45] “Módulo de intensidad ultravioleta Solar, Sensor UV 240-370nm, GUVVA-S12SD 3528, CC 3,3-5V, para Arduino /Raspberry pi”, aliexpress. Consultado: el 11 de junio de 2025. [En línea]. Disponible en: https://es.aliexpress.com/item/1005005486243061.html?src=ibdm_d03p0558e02r02&sk=&aff_platform=&aff_trace_key=&af=&cv=&cn=&dp=&aff_short_key=
- [46] “MÓDULO SENSOR DE HUMEDAD Y TEMPERATURA DE SUELO FC-28 HIGRÓMETRO”, Grupo Electrostore. Consultado: el 11 de junio de 2025. [En línea]. Disponible en: <https://grupoelectrostore.com/shop/sensores/temperatura/modulo-sensor-de-humedad-y-temperatura-de-suelo-fc-28-higrometro/>
- [47] Kechengwei Store, “GY-LSM6DS3 LSM6DS3 Acelerómetro Giroscopio Módulo Sensor de Temperatura Digital Incorporado SPI IIC I2C 8kb FIFO Buffer 5V - AliExpress 502”, aliexpress. Consultado: el 11 de junio de 2025. [En línea]. Disponible en: https://es.aliexpress.com/item/1005008369475742.html?src=ibdm_d03p0558e02r02&sk=&aff_platform=&aff_trace_key=&af=&cv=&cn=&dp=&aff_short_key=
- [48] Amazon, “Juego de 3 placas de desarrollo ESP32 LoRa V3 + N30 Protect Case Set – con antena de 915 MHz y dispositivos SX1262 LoRa V3 para Meshtastic Arduino LoRawan IOT”. Consultado: el 11 de junio de 2025. [En línea]. Disponible en: https://www.amazon.com/-/es/Juego-placas-desarrollo-ESP32-Protect/dp/B0DP6FJK69/ref=sr_1_3?__mk_es_US=%C3%85M%C3%85%C5%BD%C3%95%C3%91&sr=8-3&th=1
- [49] “RASPBERRY PI 5 MODELO B 8GB RAM”, Grupo Electrostore. Consultado: el 11 de junio de 2025. [En línea]. Disponible en: <https://grupoelectrostore.com/shop/placas-para-programacion/raspberry/raspberry-pi-5-modelo-b-8gb-ram/>
- [50] MODIS Web, “Land Cober”. Consultado: el 11 de junio de 2025. [En línea]. Disponible en: <https://modis.gsfc.nasa.gov/>
- [51] “TEXTURA DEL SUELO”, FAO. Consultado: el 11 de junio de 2025. [En línea]. Disponible en: https://www.fao.org/fishery/static/FAO_Training/FAO_Training/General/x6706s/x6706s06.htm

ANEXOS

```
# Importación de librerías necesarias
import pandas as pd
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split, StratifiedKFold
from sklearn.metrics import (
    roc_auc_score, average_precision_score, confusion_matrix,
    accuracy_score,
    precision_score, recall_score, f1_score, log_loss, brier_score_loss,
    RocCurveDisplay,
    PrecisionRecallDisplay
)
from sklearn.calibration import CalibratedClassifierCV
from sklearn.preprocessing import StandardScaler
from sklearn.feature_selection import SelectFromModel
from sklearn.ensemble import RandomForestClassifier
from sklearn.utils.class_weight import compute_class_weight
from skopt import BayesSearchCV
from skopt.space import Real
import matplotlib.pyplot as plt
import seaborn as sns
import joblib
import warnings
warnings.filterwarnings("ignore")

# Variables de archivo
entrenamiento = 'dataset_entrenamiento.xlsx'
evaluar = 'evaluar.xlsx'
output_file = 'resultado_prediccion_2.xlsx'

# Cargar y preparar los datos
def load_data(filepath):
    df = pd.read_excel(filepath)
    df.dropna(inplace=True) # Eliminar filas con valores faltantes

    X = df[[
        'Prc_acum_5', 'Prc_acum_10', 'Prc_acum_15', 'Prc_acum_20',
        'Prc_acum_25', 'Prc_acum_30', 'temp. Min.', 'temp. Max.', 'temp.
Mean.',
        'Rel. Humidity', 'Eto', 'Solar_radiation', 'Soil_mosture',
        'Tree_cover_prc',
        'Non_tree_cover_prc', 'Bare_cover_prc', 'Slope', 'Soil',
        'Land_cover']]
    y = df['Landslide']
    return X, y
```

```

# Escalado y selección de características
def preprocess_data(X, y):
    scaler = StandardScaler()
    X_scaled = scaler.fit_transform(X)

    selector = SelectFromModel(RandomForestClassifier(n_estimators=100,
random_state=42))
    X_selected = selector.fit_transform(X_scaled, y)
    return X_selected, scaler, selector

# Entrenamiento con calibración, balanceo y optimización bayesiana
def train_logistic_regression(X, y):
    class_weights = compute_class_weight(class_weight='balanced',
classes=np.unique(y), y=y)
    weight_dict = {i: w for i, w in zip(np.unique(y), class_weights)}

    base_model = LogisticRegression(solver='liblinear',
class_weight=weight_dict, random_state=42)
    param_space = {
        'C': Real(1e-4, 1e2, prior='log-uniform')
    }

    cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
    opt = BayesSearchCV(base_model, param_space, cv=cv, scoring='roc_auc',
n_iter=30, n_jobs=-1)
    opt.fit(X, y)

    best_model = CalibratedClassifierCV(opt.best_estimator_, cv=5)
    best_model.fit(X, y)

    print("\nMejor hiperparámetro C encontrado:", opt.best_params_)
    return best_model

# Evaluar el modelo
def evaluate_model(model, X_test, y_test):
    y_proba = model.predict_proba(X_test)[:, 1]
    y_pred = model.predict(X_test)

    print("\n--- Evaluación del Modelo ---")
    print("Accuracy:", accuracy_score(y_test, y_pred))
    print("F1-score:", f1_score(y_test, y_pred))
    print("Sensibilidad (Recall):", recall_score(y_test, y_pred))
    print("Log Loss:", log_loss(y_test, y_proba))
    print("Brier Score:", brier_score_loss(y_test, y_proba))
    print("AUC-ROC:", roc_auc_score(y_test, y_proba))
    print("AUC-PR:", average_precision_score(y_test, y_proba))

    cm = confusion_matrix(y_test, y_pred)

```

```

sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title("Matriz de Confusión")
plt.xlabel("Predicción")
plt.ylabel("Real")
plt.show()

RocCurveDisplay.from_estimator(model, X_test, y_test)
plt.title("Curva ROC")
plt.show()

PrecisionRecallDisplay.from_estimator(model, X_test, y_test)
plt.title("Curva Precision-Recall")
plt.show()

# Predecir y guardar resultados
def predict_new_data(model, new_data_path, scaler, selector):
    new_df = pd.read_excel(new_data_path)
    new_df.dropna(inplace=True)

    X_new = new_df[[
        'Prc_acum_5', 'Prc_acum_10', 'Prc_acum_15', 'Prc_acum_20',
        'Prc_acum_25', 'Prc_acum_30', 'temp. Min.', 'temp. Max.', 'temp.
Mean.',
        'Rel. Humidity', 'Eto', 'Solar_radiation', 'Soil_mosture',
        'Tree_cover_prc',
        'Non_tree_cover_prc', 'Bare_cover_prc', 'Slope', 'Soil',
        'Land_cover']]

    X_scaled = scaler.transform(X_new)
    X_selected = selector.transform(X_scaled)
    probas = model.predict_proba(X_selected)[: , 1] * 100

    new_df['Probabilidad_Deslizamiento_%'] = probas
    new_df.to_excel(output_file, index=False)
    print(f"\nPredicciones guardadas en: {output_file}")

# Programa principal
if __name__ == "__main__":
    X, y = load_data(entrenamiento)
    X_prep, scaler, selector = preprocess_data(X, y)

    X_train, X_test, y_train, y_test = train_test_split(X_prep, y,
test_size=0.25, stratify=y, random_state=42)

    model = train_logistic_regression(X_train, y_train)
    evaluate_model(model, X_test, y_test)

    joblib.dump(model, "logistic_model.pkl")

```

```

joblib.dump(scaler, "scaler.pkl")
joblib.dump(selector, "selector.pkl")
print("\nModelo, escalador y selector guardados.")

predict_new_data(model, evaluar, scaler, selector)

```

Anexo 1. Código de algoritmo de regresión logística.

```

# Importación de librerías necesarias
import pandas as pd
import numpy as np
from sklearn.svm import SVC
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split, StratifiedKFold
from sklearn.metrics import (
    roc_auc_score, average_precision_score, confusion_matrix,
    accuracy_score,
    recall_score, f1_score, log_loss, brier_score_loss, RocCurveDisplay,
    PrecisionRecallDisplay
)
from skopt import BayesSearchCV
from skopt.space import Real, Categorical
import matplotlib.pyplot as plt
import seaborn as sns
import joblib
import warnings
warnings.filterwarnings("ignore")

# Variables de entrada
entrenamiento = 'dataset_entrenamiento.xlsx'
evaluar = 'evaluar.xlsx'
output_file = 'resultado_prediccion_3.xlsx'

# Carga y preparación de datos

def cargar_datos(ruta):
    df = pd.read_excel(ruta)
    df.dropna(inplace=True)

    # Variables predictoras (excluyendo la variable objetivo)
    X = df[[
        'Prc_acum_5', 'Prc_acum_10', 'Prc_acum_15', 'Prc_acum_20',
        'Prc_acum_25', 'Prc_acum_30', 'temp. Min.', 'temp. Max.', 'temp.
Mean.',
        'Rel. Humidity', 'Eto', 'Solar_radiation', 'Soil_mosture',
        'Tree_cover_prc',

```

```

        'Non_tree_cover_prc', 'Bare_cover_prc', 'Slope', 'Soil',
        'Land_cover']]

    # Variable objetivo
    y = df['Landslide']
    return X, y

# Entrenamiento del modelo SVM con optimización bayesiana
def entrenar_svm(X, y):
    # Pipeline con estandarización y SVM
    pipeline = Pipeline([
        ('scaler', StandardScaler()),
        ('svm', SVC(probability=True, random_state=42))
    ])

    # Espacio de hiperparámetros
    search_space = {
        'svm__C': Real(1e-3, 1e3, prior='log-uniform'),
        'svm__gamma': Real(1e-4, 1e1, prior='log-uniform'),
        'svm__kernel': Categorical(['rbf', 'poly', 'sigmoid'])
    }

    # Validación cruzada estratificada
    cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)

    # Búsqueda bayesiana
    opt = BayesSearchCV(
        pipeline, search_space,
        cv=cv, scoring='roc_auc', n_iter=50,
        n_jobs=-1, verbose=0, random_state=42
    )
    opt.fit(X, y)
    print("\nMejores hiperparámetros encontrados:", opt.best_params_)
    return opt.best_estimator_

# Evaluación del modelo

def evaluar_modelo(modelo, X_test, y_test):
    y_proba = modelo.predict_proba(X_test)[: , 1]
    y_pred = modelo.predict(X_test)

    print("\n--- Evaluación del Modelo SVM ---")
    print("Accuracy:", accuracy_score(y_test, y_pred))
    print("F1-score:", f1_score(y_test, y_pred))
    print("Sensibilidad (Recall):", recall_score(y_test, y_pred))
    print("Log Loss:", log_loss(y_test, y_proba))
    print("Brier Score:", brier_score_loss(y_test, y_proba))
    print("AUC-ROC:", roc_auc_score(y_test, y_proba))

```

```

print("AUC-PR:", average_precision_score(y_test, y_proba))

cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d', cmap='Oranges')
plt.title("Matriz de Confusión")
plt.xlabel("Predicción")
plt.ylabel("Real")
plt.show()

RocCurveDisplay.from_estimator(modelo, X_test, y_test)
plt.title("Curva ROC")
plt.show()

PrecisionRecallDisplay.from_estimator(modelo, X_test, y_test)
plt.title("Curva Precision-Recall")
plt.show()

# Predicción con nuevos datos

def predecir_nuevos_datos(modelo, archivo_entrada, archivo_salida):
    nuevos = pd.read_excel(archivo_entrada)
    nuevos.dropna(inplace=True)

    X_nuevos = nuevos[[
        'Prc_acum_5', 'Prc_acum_10', 'Prc_acum_15', 'Prc_acum_20',
        'Prc_acum_25', 'Prc_acum_30', 'temp. Min.', 'temp. Max.', 'temp.
Mean.',
        'Rel. Humidity', 'Eto', 'Solar_radiation', 'Soil_mosture',
        'Tree_cover_prc',
        'Non_tree_cover_prc', 'Bare_cover_prc', 'Slope', 'Soil',
        'Land_cover']]

    probas = modelo.predict_proba(X_nuevos)[: , 1] * 100
    nuevos['Probabilidad_Deslizamiento_%'] = probas
    nuevos.to_excel(archivo_salida, index=False)
    print(f"\nPredicciones guardadas en '{archivo_salida}'")

# Programa principal
if __name__ == '__main__':
    # Cargar datos
    X, y = cargar_datos(entrenamiento)

    # Dividir datos
    X_train, X_test, y_train, y_test = train_test_split(
        X, y, test_size=0.25, stratify=y, random_state=42)

    # Entrenar modelo
    modelo_svm = entrenar_svm(X_train, y_train)

```

```

# Evaluar modelo
evaluar_modelo(modelo_svm, X_test, y_test)

# Guardar modelo
joblib.dump(modelo_svm, 'modelo_svm_deslizamiento.pkl')
print("\nModelo SVM guardado como 'modelo_svm_deslizamiento.pkl'")

# Predecir en nuevos datos
predecir_nuevos_datos(modelo_svm, evaluar, output_file)

```

Anexo 2. Código de algoritmo SVM.

```

# Importación de librerías necesarias
import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split, StratifiedKFold
from sklearn.metrics import (
    roc_auc_score, average_precision_score, confusion_matrix,
    accuracy_score,
    precision_score, recall_score, f1_score, log_loss, brier_score_loss,
    RocCurveDisplay,
    PrecisionRecallDisplay
)
from skopt import BayesSearchCV
from skopt.space import Real, Integer
import matplotlib.pyplot as plt
import seaborn as sns
import joblib
import warnings
warnings.filterwarnings("ignore")

# Variables globales para rutas de archivos
entrenamiento = 'dataset_entrenamiento.xlsx'
evaluar = 'evaluar.xlsx'
modelo_salida = 'random_forest_landslide_model.pkl'
output_file = 'resultado_prediccion.xlsx'

# Función para cargar y preparar los datos desde un archivo .xlsx
def load_data(filepath):
    df = pd.read_excel(filepath)
    # Eliminar filas con valores faltantes
    df.dropna(inplace=True)

    # Variables predictoras (features)
    X = df[[

```

```

        'Prc_acum_5', 'Prc_acum_10', 'Prc_acum_15', 'Prc_acum_20',
        'Prc_acum_25', 'Prc_acum_30', 'temp. Min.', 'temp. Max.', 'temp.
Mean.',
        'Rel. Humidity', 'Eto', 'Solar_radiation', 'Soil_mosture',
        'Tree_cover_prc',
        'Non_tree_cover_prc', 'Bare_cover_prc', 'Slope', 'Soil',
        'Land_cover']]

    # Variable objetivo binaria (1 si hubo deslizamiento, 0 si no)
    y = df['Landslide']

    return X, y

# Función para entrenar modelo con optimización bayesiana
def train_random_forest(X, y):
    rf = RandomForestClassifier(random_state=42)

    # Definición de espacio de búsqueda para hiperparámetros
    param_space = {
        'n_estimators': Integer(100, 1000),
        'max_depth': Integer(5, 50),
        'min_samples_split': Integer(2, 20),
        'min_samples_leaf': Integer(1, 20),
        'max_features': Real(0.1, 1.0, prior='uniform')
    }

    # Validación cruzada estratificada
    cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)

    # Búsqueda bayesiana
    opt = BayesSearchCV(
        rf, param_space,
        cv=cv, scoring='roc_auc',
        n_iter=50, n_jobs=-1, verbose=0, random_state=42
    )

    # Ajuste del modelo
    opt.fit(X, y)

    # Mejor modelo entrenado
    best_model = opt.best_estimator_
    print("\nMejores hiperparámetros encontrados:", opt.best_params_)
    return best_model

# Función para evaluar el modelo
def evaluate_model(model, X_test, y_test):
    y_proba = model.predict_proba(X_test)[: , 1]
    y_pred = model.predict(X_test)

```

```

print("\n--- Evaluación del Modelo ---")
print("Accuracy:", accuracy_score(y_test, y_pred))
print("F1-score:", f1_score(y_test, y_pred))
print("Sensibilidad (Recall):", recall_score(y_test, y_pred))
print("Log Loss:", log_loss(y_test, y_proba))
print("Brier Score:", brier_score_loss(y_test, y_proba))
print("AUC-ROC:", roc_auc_score(y_test, y_proba))
print("AUC-PR:", average_precision_score(y_test, y_proba))

# Matriz de confusión
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title("Matriz de Confusión")
plt.xlabel("Predicción")
plt.ylabel("Real")
plt.show()

# Curvas
RocCurveDisplay.from_estimator(model, X_test, y_test)
plt.title("Curva ROC")
plt.show()

PrecisionRecallDisplay.from_estimator(model, X_test, y_test)
plt.title("Curva Precision-Recall")
plt.show()

# Función para predecir con nuevos datos y retornar probabilidades
def predict_new_data(model_path, new_data_path):
    new_df = pd.read_excel(new_data_path)
    new_df.dropna(inplace=True)

    X_new = new_df[[
        'Prc_acum_5', 'Prc_acum_10', 'Prc_acum_15', 'Prc_acum_20',
        'Prc_acum_25', 'Prc_acum_30', 'temp. Min.', 'temp. Max.', 'temp.
Mean.',
        'Rel. Humidity', 'Eto', 'Solar_radiation', 'Soil_mosture',
        'Tree_cover_prc',
        'Non_tree_cover_prc', 'Bare_cover_prc', 'Slope', 'Soil',
        'Land_cover']]

    model = joblib.load(model_path)
    predictions = model.predict_proba(X_new)[: , 1] * 100

    for i, prob in enumerate(predictions):
        print(f"Instancia {i+1}: Probabilidad de deslizamiento =
{prob:.2f}%")

```

```

# Guardar resultados en un archivo Excel
new_df['Probabilidad_Deslizamiento_%'] = predictions
new_df.to_excel(output_file, index=False)
print(f"\nResultados guardados en: {output_file}")

return predictions

# Ejecución principal (solo si se ejecuta este script directamente)
if __name__ == "__main__":
    X, y = load_data(entrenamiento)

    # División de datos para entrenamiento y prueba
    X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.25, stratify=y, random_state=42)

    # Entrenar modelo con optimización bayesiana
    model = train_random_forest(X_train, y_train)

    # Evaluación del modelo
    evaluate_model(model, X_test, y_test)

    # Guardar modelo entrenado
    joblib.dump(model, modelo_salida)
    print(f"\nModelo guardado como '{modelo_salida}'")

    # Predicción en nuevos datos
    predict_new_data(modelo_salida, evaluar)

```

Anexo 3. Código de algoritmo Random Forest.

```

import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import (accuracy_score, confusion_matrix,
classification_report,
                             log_loss, brier_score_loss, roc_auc_score,
precision_recall_curve,
                             roc_curve, f1_score, recall_score)
from sklearn.neural_network import MLPClassifier
import matplotlib.pyplot as plt
import seaborn as sns
import joblib

# Paso 1: Cargar los datos
base_datos = pd.read_excel('dataset_entrenamiento.xlsx')
evaluar_datos = pd.read_excel('evaluar.xlsx')

```

```

# Paso 2: Separar características y etiquetas
X = base_datos.drop(columns=['Landslide'])
y = base_datos['Landslide']

# Paso 3: Dividir los datos en entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Paso 4: Escalar las características (excepto columnas ya normalizadas si
aplica)
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
evaluar_scaled = scaler.transform(evaluar_datos[X.columns])

# Paso 5: Crear y entrenar el modelo de red neuronal
model = MLPClassifier(hidden_layer_sizes=(64, 32, 16), activation='relu',
solver='adam',
                        max_iter=2000, random_state=42, early_stopping=True)
model.fit(X_train, y_train)

# Paso 6: Predecir en el conjunto de prueba
y_pred = model.predict(X_test)
y_proba = model.predict_proba(X_test)[:, 1]

# Paso 7: Evaluar el modelo
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred)
logloss = log_loss(y_test, y_proba)
brier = brier_score_loss(y_test, y_proba)
roc_auc = roc_auc_score(y_test, y_proba)
f1 = f1_score(y_test, y_pred)
sensitivity = recall_score(y_test, y_pred)

print(f'Precisión: {accuracy:.4f}')
print(f'Log Loss: {logloss:.4f}')
print(f'Brier Score: {brier:.4f}')
print(f'ROC AUC: {roc_auc:.4f}')
print(f'F1 Score: {f1:.4f}')
print(f'Sensibilidad: {sensitivity:.4f}')
print('Matriz de Confusión:')
print(conf_matrix)
print('Reporte de Clasificación:')
print(class_report)

# Paso 8: Graficar curvas ROC y Precision-Recall

```

```

fpr, tpr, _ = roc_curve(y_test, y_proba)
precision, recall, _ = precision_recall_curve(y_test, y_proba)

plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)
plt.plot(fpr, tpr, label=f'AUC = {roc_auc:.2f}')
plt.plot([0, 1], [0, 1], linestyle='--', color='gray')
plt.title('Curva ROC')
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.legend()

plt.subplot(1, 2, 2)
plt.plot(recall, precision, label='Curva PR')
plt.title('Curva Precision-Recall')
plt.xlabel('Recall')
plt.ylabel('Precision')
plt.legend()

plt.tight_layout()
plt.show()

# Paso 9: Predecir la probabilidad de deslizamientos para nuevos datos
probabilities = model.predict_proba(evaluar_scaled)[: , 1]
evaluar_datos['Probabilidad Deslave'] = probabilities

# Mostrar los resultados
columnas_resultado = [col for col in evaluar_datos.columns if col in
X.columns] + ['Probabilidad Deslave']
print(evaluar_datos[columnas_resultado])

# Paso 10: Interpretación de importancia de variables (pesos de entrada)
importances = np.abs(model.coefs_[0]).mean(axis=1)
feature_importances = pd.DataFrame({'Feature': X.columns, 'Importance':
importances})
feature_importances = feature_importances.sort_values(by='Importance',
ascending=False)

plt.figure(figsize=(10, 6))
sns.barplot(x='Importance', y='Feature', data=feature_importances)
plt.title('Importancia de las Variables')
plt.xlabel('Importancia')
plt.ylabel('Característica')
plt.tight_layout()
plt.show()

# Paso 11: Guardar resultados y modelo

```

```

evaluar_datos.to_excel('resultados_prediccion4.xlsx', index=False)
joblib.dump(model, 'modelo_red_neuronal.pkl')
joblib.dump(scaler, 'escalador.pkl')

```

Anexo 4. Código de algoritmo de red neuronal (MLP).

DESCRIPCION AGROECOLOGICA PREDIO PAGMA 1.3908 Ha.	
USO	INTERVENCION ANTROPOGENICA
ZONAS DE VIDA NATURAL	BOSQUE HUMEDO - PISO MONTANO - REGION TEMPLADA FRIA
FACTORES BIOTICOS (EXISTENCIA FLORA Y FAUNA NATIVOS)	SI
PRESIPITACION ANUAL	800 - 1000 mm/año
TEMPERATURA MEDIA	12º - 18ºC
ALTITUD PROMEDIO	1800 m.s.n.m.
SUELO	
TAXONOMIA	ENTISOLES
DESCRIPCION	AZONALES Y ALGUNOS GLEY POCO HUMICOS
HORIZONTE	SUPERIOR
TEXTURA	FRANCO-LIMO-ARCILLOSO
ESTRUCTURA	SUELTA
DRENAJE	BUENO
PROCESO EROSIVO	HIDRICO - EOLICO-ANTROPOGENO
PENDIENTE	10 - 20%
REGIMEN DE HUMEDAD	ESTACIONAL(INVIERNO)
PH	5.8 - 6.5 ACIDO
APTITUD	AGROPECUARIA-FORESTAL
FERTILIDAD	N-BAJO,P-BAJO,K-MEDIO
DESARROLLO AGROPECUARIO	
USO ACTUAL	PASTOS NATURALES
ROTACION	MAIZ - FREJOL - PASTOS ARTIFICIALES
DISPONIBILIDAD DE AGUA DE RIEGO	NOI
DOTACION DE AGUA ENTUBADA	SI
TECNOLOGIA APLICADA	SEMITECNIFICADA
INTENSIDAD Y NIVELES DE PRODUCCION	MEDIO
PROGRESO DEL SECTOR AGROPECUARIO	MEDIO
VIAS DE ACCESO	CARRETERO DE SEGUNDO ORDEN
FRACCIONAMIENTO EN 2 LOTES	POSIBILIDAD: SI...(X)
ING.AGR. DIEGO F. ORMAZA DELGADO	
REGISTRO SENECYT 1005-12-1112605	
 PERITO CALIFICADO ACREDITACION 6000182 SENECYT 1005 12 1112605	

Anexo 5. Estudio del suelo del ambiente controlado.