ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

Sistema de Riego Inteligente usando Machine Learning

PROYECTO INTEGRADOR

Previo la obtención del Título de:

Ingeniero en Telemática

Presentado por:

Erick Fernando Cabay Yanqui

Kenyi Janell Panchana Merchan

GUAYAQUIL - ECUADOR

Año: 2023

DEDICATORIA

Erick Cabay Y.

A mis padres, Franklin Cabay y Fanny Yanqui, quienes siempre fueron pilares en mi vida y formaron quien soy en la actualidad. A mis amigos, hermanos y familia por siempre estar para apoyarme en lo que necesite. También a mi querido y peludo amigo Maylo, quien siempre me supo sacar una sonrisa en mis momentos más bajos con sus travesuras.

Kenyi Panchana M.

Quiero dedicar este logro a mis padres, Jorge y Jannet, quienes han sido un apoyo incondicional a lo largo de mi travesía universitaria; sin ustedes, no habría alcanzado este hito. A mis hermanos Joyner, Jairo y Elkin por ser fuente de constante inspiración. También deseo dedicar este éxito a cada una de mis amistades que me acompañaron en este proceso, en especial a Joel M., quien me motivó a seguir adelante y estuvo presente en cada pequeño paso del camino.

AGRADECIMIENTOS

Erick Cabay Y.

Quiero agradecer a mis hermanos pequeños, por haberme enseñado el valor de la paciencia y perseverancia, a mis padres por escucharme, criarme y apoyarme con todo lo que tienen siempre que lo necesité, también a mi compañera de tesis quien estuvo presente en cada paso para poder elaborar nuestro proyecto final.

Kenyi Panchana M.

Expreso mi agradecimiento a mi familia y amigos por alentarme a alcanzar la meta que me propuse. Además, quiero expresar mi profunda gratitud a la Familia Rubio Alvarado y a Raquel Ruiz, quienes generosamente me abrieron las puertas de sus hogares para que pudiera continuar con mis estudios. También deseo resaltar la valiosa contribución de mi compañero de tesis, cuya colaboración fue fundamental para culminar exitosamente este proceso.

DECLARACIÓN EXPRESA

“Los derechos de titularidad y explotación, nos corresponde conforme al reglamento de propiedad intelectual de la institución; Erick Cabay y Kenyi Panchana damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual”

Erick Cabay

Kenyi Panchana

EVALUADORES

Firmado electrónicamente por:
MARIA JOSE RAMIREZ PRADO

María José Ramírez

PROFESOR DE LA MATERIA

Firmado electrónicamente por:
NESTOR XAVIER ARREAGA ALVARADO

Néstor Arreaga

PROFESOR TUTOR

RESUMEN

Este trabajo investigativo se concentra en la mejora de una infraestructura preexistente de sensores de humedad en un jardín contiguo al laboratorio de telemática. Con el objetivo de fortalecer la red de sensores, se introdujeron dos nodos adicionales mediante microcontroladores que operan bajo el protocolo ESPNOW. Este enfoque se diseñó para ampliar la cobertura y obtener mediciones más precisas, aspecto crucial para la eficacia en la toma de decisiones respecto al riego. Como estrategia de optimización, se implementó un modelo de aprendizaje automático basado en Random Forest. Este modelo se nutre de datos provenientes de la API meteorológica Tomorrow.io y de la plataforma HayloT de la universidad ESPOL. La válvula inteligente de la marca Tuya, responsable de controlar los aspersores, responde activándose o desactivándose según el previo entrenamiento realizado, asegurando un riego eficiente del jardín hasta alcanzar niveles óptimos de humedad.

Desde la perspectiva de la programación, se llevaron a cabo solicitudes a ambas APIs mencionadas, integrando la información obtenida para obtener lecturas actualizadas de las variables requeridas para la predicción del modelo. Los scripts, que incluyen el modelo Random Forest, se ejecutan periódicamente en una máquina virtual configurada como servidor. Además, se implementó un sistema de registro detallado mediante archivos .txt que almacenan la hora, fecha y la acción específica de apertura o cierre de la válvula. Este registro se convierte en un historial esencial para el análisis y la evaluación del rendimiento del sistema a lo largo del tiempo.

Palabras Clave: Sensores de humedad, Válvula inteligente, Random Forest, Machine Learning, Red de sensores

ABSTRACT

This research work focuses on enhancing an existing infrastructure of moisture sensors in a garden adjacent to the telematics laboratory. To strengthen the sensor network, two additional nodes were introduced through microcontrollers operating under the ESPNOW protocol. This approach was designed to expand coverage and obtain more precise measurements, a crucial aspect for effective decision-making regarding irrigation. As an optimization strategy, a machine learning model based on Random Forest was implemented. This model is fed with data from the Tomorrow.io meteorological API and the HayloT platform of ESPOL University. The intelligent valve from the Tuya brand, responsible for controlling sprinklers, responds by activating or deactivating based on a predetermined humidity threshold, ensuring efficient garden irrigation until optimal moisture levels are reached.

From a programming perspective, requests were made to both mentioned APIs, integrating the obtained information to acquire updated readings of the variables required for the model's prediction. The scripts, including the Random Forest model, are periodically executed on a virtual machine configured as a server. Additionally, a detailed logging system was implemented using .txt files to store the timestamp, date, and the specific action of opening or closing the valve. This log becomes an essential history for the analysis and evaluation of the system's performance over time.

Keywords: Moisture sensors, Smart valve, Random Forest, Machine Learning, Sensor network

ÍNDICE GENERAL

ABREVIATURAS

ESPOL	Escuela Superior Politécnica del Litoral
WSN	Wireless Sensor Network
API	Application Programming Interface
IoT	Internet Of Things
LoRa	Long Range Modulation
LSTM	Long Short Term Memory networks
MQTT	Message Queuing Telemetry Transport
CO2	Carbon Dioxide
WiFi	Wireless Fidelity
IEEE	Institute of Electrical and Electronics Engineers
MAC	Media Access Control
IDE	Integrated Development Environment
SSH	Secure Shell
Telnet	Teletype Network
RDP	Remote Desktop Protocol
LST	Laboratorio de Sistemas Telemáticos
VPN	Virtual Private Network
USB	Universal Serial Bus

SIMBOLOGÍA

mins	Minutos
mg	Miligramo
pH	Potencial de Hidrógeno
m	Metro

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1

1. INTRODUCCIÓN

La gestión inadecuada del agua en la irrigación es un desafío de gran envergadura con importantes implicaciones tanto económicas como medioambientales. Los sistemas tradicionales de riego, que se basan en horarios predefinidos, suelen derrochar agua y energía al no adaptarse a las condiciones cambiantes en tiempo real, lo que se traduce en costos operativos innecesariamente elevados. Además, estos métodos pueden conllevar problemas como la irrigación insuficiente o excesiva, lo que afecta negativamente la calidad del suelo y el entorno. La solución para abordar esta problemática radica en la implementación de un Sistema de Riego Inteligente fundamentado en el Aprendizaje Automático (Machine Learning), que emplea datos en tiempo real para tomar decisiones precisas y eficientes en lo que respecta al riego. Este sistema se adapta automáticamente a las variaciones ambientales, reduciendo al mínimo el desperdicio de agua y energía, fomentando, de esta forma, una gestión más responsable de los recursos hídricos, protegiendo el medio ambiente y optimizando los costos operativos. El presente proyecto se enfoca en la expansión de la red de Sensores Inalámbricos (WSN) en el jardín que está ubicado en la parte posterior del Laboratorio de Sistemas Telemáticos de la Escuela Superior Politécnica del Litoral y la implementación de algoritmos de Aprendizaje Automático para optimizar el consumo de agua en el sistema de riego. El proyecto surge debido a la ineficiencia en el uso del agua en la agricultura y tiene como objetivo contribuir a la sostenibilidad y eficiencia en el riego. Los objetivos generales y específicos se centran en mejorar la infraestructura de la red, recopilar datos de diversas fuentes, utilizar el Aprendizaje Automático para crear modelos predictivos y gestionar el riego de manera eficaz. Sin embargo, el proyecto reconoce ciertas limitaciones, como el suministro de energía y las capacidades de procesamiento de los nodos, que requerirán una atención meticulosa durante su desarrollo. Asimismo, se destaca la importancia de

abordar la latencia en la comunicación de datos en el contexto del sistema de Aprendizaje Automático y el almacenamiento.

1.1 Definición de la problemática

El inadecuado manejo del agua en la irrigación representa una cuestión de importancia crítica, con consecuencias económicas y ambientales significativas. Los sistemas convencionales de riego que siguen horarios fijos tienden a despilfarrar agua y energía al no adaptarse a las condiciones en tiempo real, lo que resulta en costos operativos innecesariamente altos. Además, estos métodos pueden dar lugar a problemas de subirrigación o sobrerriegos, afectando negativamente la calidad del suelo y del entorno. La solución para abordar este desafío reside en la implementación de un Sistema de Riego Inteligente basado en Machine Learning, que utiliza datos en tiempo real para tomar decisiones precisas y eficientes en cuanto al riego. Este sistema se ajusta automáticamente a las variaciones ambientales, lo que minimiza el desperdicio de agua y energía, promoviendo así una gestión más responsable de los recursos hídricos, protegiendo el medio ambiente y optimizando los costos operativos.

1.2 Justificación del problema

El agua es un recurso esencial para el crecimiento de la agricultura no obstante con el tiempo se está volviendo escaso debido a la creciente competencia entre sus usos. Estudios recientes señalan que el empleo del agua en el uso agrícola es sumamente deficiente, ya que 54 por ciento del agua que se utiliza para riego no se aprovecha. (Miguel and Enrique, 2020)

En ESPOL está problemática también es evidente, específicamente en el marco de riego para las diferentes zonas de jardines o cultivos por tal motivo en la Tesis WSN (Wireless Sensor Network) con estandarización de datos consumibles vía HayloT como solución se implementó un sistema de riego automatizado para los aspersores mediante una WSN se contaba con nodos que monitoreaban y registraban de manera continua la humedad del suelo en el entorno de los aspersores, posterior a ello la WSN recopilaba y transmitía los datos de la humedad del suelo en tiempo real y los datos se enviaban de

manera inalámbrica a una plataforma de almacenamiento llamada HayloT para la toma de decisiones. (Indacochea and Valencia, 2023)

La optimización del agua en el riego es crucial para el ahorro del recurso, una adecuada irrigación al suelo ayuda en la productividad de las plantas que se encuentran en esta área. Además de prevenir el sobre-riego o sub-riego que perjudicaría la calidad de los frutos. En la actualidad con los avances tecnológicos la automatización no es suficiente, implementar en el sistema la inteligencia es de gran relevancia para la toma de decisiones eficaces y eficientes debido que se va adaptando a los diferentes cambios que pueden ir surgiendo en el ambiente. El desarrollo de algoritmos y estrategias para minimizar el desperdicio de recursos ha ido creciendo en los últimos años, ubicando como principal eje a la sostenibilidad, por tal motivo el presente proyecto tiene como finalidad contribuir con una solución que respalde estos criterios.

1.3 Objetivos:

1.3.1 Objetivo General

Implementar un algoritmo de aprendizaje de máquina al sistema “WSN (Wireless Sensor Network) con estandarización de datos consumibles vía HayloT” utilizando información de una interfaz de programación de aplicación de datos meteorológicos y valores de humedad del suelo para la optimización del agua consumida por los aspersores.

1.3.2 Objetivos Específicos

- Incrementar la infraestructura de la red “Wireless Sensor Network” (WSN) mediante la adición de dos nodos sensores. Estos nodos mejorarán la capacidad y ampliarán la cobertura de la red facilitando la adquisición de datos.
- Adquirir datos de diversas fuentes, que incluyen la interfaz de programación de aplicación de datos meteorológicos y la WSN. Estos datos se recopilarán de manera sistemática y precisa, fortaleciendo la calidad y cantidad de información disponible para análisis.
- Utilizar un enfoque basado en Machine Learning para la creación de un modelo predictivo. Para preveer las necesidades de riego a partir de datos meteorológicos y

condiciones del suelo.
- Determinar patrones y tendencias en los datos recopilados para la gestión más eficaz y eficiente del riego.

1.4 Alcance y Limitaciones

El presente proyecto se concentra en la ampliación de la red WSN actual mediante la inclusión de dos nuevos nodos con el propósito de aumentar su capacidad de vigilancia en el área de estudio. Esta incorporación de nodos ubicados estratégicamente pretende optimizar la cobertura y la obtención de datos en dicho entorno. Además de la expansión de la infraestructura de la red, se llevará a cabo la introducción de un sistema basado en Machine Learning. Este sistema empleará los datos recopilados por la WSN, junto con la información procedente de una API meteorológica, con el fin de anticipar las necesidades de riego. Esta anticipación facilitará una administración más eficaz del recurso hídrico y una respuesta activa a las variaciones en las condiciones climáticas y la humedad del suelo. A pesar de los avances propuestos, se reconocen ciertas restricciones que requerirán una atención meticulosa en el transcurso de este proyecto. En primer lugar, se encuentra la cuestión del suministro de energía. Dada la ubicación remota y de difícil acceso de algunos nodos, la utilización de baterías recargables podría resultar poco eficaz. En este sentido, se necesita una solución que garantice el suministro continuo de energía a todos los nodos. Además, los microcontroladores utilizados en los nodos presentan limitaciones en lo que respecta a la capacidad de procesamiento y almacenamiento de datos. La inclusión de nuevos nodos y la implementación de Machine Learning podrían agravar estas limitaciones. Por lo tanto, resulta esencial abordar la eficiencia en el procesamiento y almacenamiento de información. Otro aspecto crucial es la comunicación serial empleada en la transmisión de datos a través de la red mesh hacia el gateway y la posterior comunicación serial para el almacenamiento. Esta modalidad de comunicación podría introducir cierta latencia en la recopilación y análisis de datos, lo cual debe ser tenido en cuenta en el diseño del sistema de Machine Learning y las configuraciones de almacenamiento.

1.5 Estado del arte

La necesidad de mejorar la gestión del riego del suelo ha sido esencial en la exploración de innovadoras técnicas con el enfoque principal en la sostenibilidad. La adopción de sistemas avanzados que recopilan datos a través de dispositivos IoT ha surgido como una de las soluciones más notables. A pesar de esto, se han empleado diversas tecnologías según las necesidades y recursos de los diferentes entornos como se mencionan en los siguientes estudios realizados.

Los autores (S et al., 2020) En su investigación subrayan la importancia de la agricultura en la economía de cualquier país y cómo se enfrenta a desafíos debido al crecimiento de la población, los cambios en el clima y la escasez de recursos. Para hacer frente a estos desafíos, se desarrolló la agricultura de precisión, que utiliza el aprendizaje automático y la Internet de las Cosas (IoT). La investigación ha identificado aplicaciones del aprendizaje automático en la agricultura, como la predicción de parámetros del suelo, la anticipación del rendimiento de los cultivos y la detección de enfermedades y malezas. También se ha explorado la clasificación de imágenes de cultivos mediante la visión por computadora. Estas tecnologías tienen el potencial de mejorar la productividad agrícola, elevar la calidad de los productos y reducir la necesidad de mano de obra, al mismo tiempo que promueven la sostenibilidad. En resumen, el uso de estas tecnologías promete optimizar la agricultura y hacerla más sostenible, lo que es crucial dadas las actuales presiones en la producción de alimentos.

(Jiménez et al., 2023) Manifiestan que su investigación se centró en implementar un sistema inteligente de prescripción de riego agrícola, utilizando tecnologías emergentes para gestionar de manera eficiente el agua en la agricultura. Se utilizaron nodos inalámbricos para recopilar datos del suelo, temperatura de la cobertura vegetal, temperatura ambiente y humedad relativa. Un sistema de inferencia en una estación central procesó estos datos y determinó la cantidad de riego necesaria. La validación se realizó utilizando el software AquaCrop para modelar los cultivos. Este sistema permitió calcular prescripciones diarias de riego según el tipo de suelo y la etapa de crecimiento de los cultivos, evitando aplicaciones excesivas o insuficientes de agua.

Al mejorar la producción de una plantación de rosas, se creó un modelo de machine learning y un sistema embebido, donde para el desarrollo se utilizó el entorno de programación Python, con compatibilidad con la plataforma thethings.iO y librerías existentes, para obtener datos por medio de sensores logrando adquirir información sobre el comportamiento de la nave florícola, esta información se almacenaba y procesaba para una post evaluación, permitiendo entrenar a un conjunto de datos, los mismos que dependiendo de su estado y el entrenamiento del modelo accionan los actuadores dentro de la nave (Albán Bautista, 2022)

Según (Liakos et al., 2018) destaca la importancia del aprendizaje automático en la agricultura, gracias al uso de tecnologías de big data y computación de alto rendimiento. El artículo proporciona una revisión detallada de investigaciones que se centran en aplicar el aprendizaje automático en sistemas de producción agrícola. Al aprovechar el aprendizaje automático en datos de sensores, los sistemas de gestión agrícola están avanzando hacia programas de inteligencia artificial en tiempo real que brindan recomendaciones e información valiosa para respaldar las decisiones y acciones de los agricultores.

Un sistema de detección automática de problemas en los sistemas de riego de cultivos, que demuestra un sólido rendimiento en los dispositivos instalados en una infraestructura preexistente. La versatilidad de este sistema se manifiesta en su capacidad de expansión en la nube, permitiendo la incorporación de una cantidad indefinida de dispositivos recolectores de datos. Además, su flexibilidad se evidenció al poder gestionar información incompleta, como se comprobó en una serie de rigurosas pruebas de resistencia. El sistema mostró viabilidad, además de eficiencia en la detección automática de problemas en sistemas de riego, incluso en entornos rurales con conectividad limitada. En estas zonas, la transferencia de datos a la nube se llevó a cabo mediante tecnologías LoRa, facilitada por la conectividad satelital (Etchanchú, 2021)

En este trabajo de investigación se ha construido un sistema de riego autónomo basado en la Internet de las Cosas (IoT). Se emplean elementos de bajo costo y

hardware - software libre (Raspberry Pi, Arduino, Linux, Java, Wildfly, Python, etc.) para implementar Redes de Sensores Inalámbricos (WSN) que permiten obtener la información de las variables agroclimáticas (Humedad del suelo, temperatura ambiente, precipitación, etc.). Se implementó un sistema de Aprendizaje Máquina (Machine Learning) para la predicción del calendario de riego empleando servicios de Computación en la Nube. Los Servicios Web tipo RESTful y los formatos Json y Xml se emplearon para permitir la interoperabilidad entre los diferentes subsistemas (predicción, riego y cliente), hardware (Raspberry Pi, Xbee, computación en la nube) y software (Python, Java). Para hacer un uso eficiente del agua este prototipo de Agricultura Inteligente (Smart Farming) se apoya en la Internet de las Cosas y el Aprendizaje Máquina para responder a las preguntas de cuándo y cuánto regar (Castro-Silva, 2016).

(Carvajal and Leonel, 2020). Proponen un sistema de IoT para monitorear y automatizar variables ambientales en invernaderos. Utiliza sensores para medir datos como temperatura, humedad y nivel de agua, y muestra esta información en una interfaz gráfica. Las mediciones y las interacciones del usuario viajan a través de una aplicación intermediaria (bróker) usando módulos ESP8266 NodeMCU conectados a Internet a través de WiFi. El sistema emplea un bróker EMQX que utiliza el protocolo MQTT para la transmisión de datos. El sistema consiste en una aplicación web que lee, almacena y muestra los datos en tiempo real en una interfaz gráfica de usuario. La aplicación web conecta al usuario con el sistema y gestiona la lectura y escritura de datos. En su fase final, utiliza una red neuronal LSTM para realizar predicciones basadas en datos históricos de las variables ambientales.

(Nuñez-Agurto et al., 2020) Mencionan una solución de bajo costo que utiliza Internet de las Cosas (IoT) para mejorarla agricultura. Se enfoca en recopilar y analizar datos ambientales de cultivos de manera centralizada y remota, especialmente la humedad y la temperatura, con el objetivo de realizar pronósticos más precisos. Utiliza microcontroladores NodeMCU ESP8266, sensores de temperatura/humedad AM2302 y sensores de lluvia MH-RD. Se emplea una conexión inalámbrica WiFi a través del protocolo MQTT para interactuar con los sensores. Las pruebas se realizaron en parcelas agrícolas, y los resultados demostraron mayor precisión en comparación con otros métodos de medición.

‘‘Este trabajo presenta un sistema basado en visión artificial que identifica las características de las hojas de plantas de jitomate y determina sus requerimientos nutricionales’’ (Vega and Sebástian, 2020). Se construyó una base de datos de imágenes que refleja los síntomas de deficiencia de nutrientes, como nitrógeno, potasio y fósforo, basada en mediciones del suelo con y sin la aplicación de estos nutrientes. Se entrenó un algoritmo supervisado con esta base de datos, logrando una alta precisión en la identificación de síntomas de deficiencia nutricional 86 por ciento y en el sistema de control semiautomático 70 por ciento.

Según (Hortoinfo, 2023), el algoritmo de aprendizaje por refuerzo (RL) para controlar completamente el clima en un tercer compartimento de un invernadero. Aunque la inteligencia artificial en la horticultura de invernadero todavía está en sus primeras etapas, este fue uno de los primeros casos de control de un invernadero de manera autónoma mediante un algoritmo de aprendizaje por refuerzo. El modelo supervisó variables como la iluminación, el uso de pantallas, la concentración de CO2 y la calefacción, pero no estaba entrenado para gestionar el riego y la poda de frutos. Durante las primeras semanas, el control fue manual, pero una vez que los frutos comenzaron a formarse, el algoritmo RL asumió el control. A pesar de que esto resultó en un clima de invernadero diferente a los otros compartimentos, el cultivo se adaptó bien a las variaciones de temperatura, lo que condujo a una producción exitosa de frutos, incluso con una estrategia de poda predefinida.

En este documento se detalla la implementación de un modelo para analizar y prever la propagación de Covid-19 utilizando técnicas de inteligencia artificial, específicamente Machine Learning, mediante estrategias de aprendizaje supervisado en programas desarrollados en Python. El propósito es procesar grandes conjuntos de datos, aprender de experiencias anteriores y generar información predictiva de manera rápida y confiable. El enfoque abarca un análisis de datos extraídos de una fuente abierta, seguido de un análisis exploratorio. Se llevaron a cabo tres predicciones: la presencia de SARS-CoV-2 en un paciente, los días hasta la mortalidad y la tasa de mortalidad por Covid. Se implementaron algoritmos de clasificación y regresión, eligiendo los

modelos Random Forest y Redes Neuronales Artificiales basándose en investigaciones previas. Las métricas de confiabilidad respaldan la validez de las predicciones esperadas, proporcionando una base sólida para la toma de decisiones adecuada. (n.d.)

1.6 Marco Teórico

1.6.1 Redes de sensores inalámbricos (WSN)

Las redes de sensores inalámbricos (WSN) son sistemas conformados por una serie de dispositivos autónomos distribuidos en el espacio, diseñados para capturar datos del entorno mediante sensores y transmitir esa información de forma inalámbrica. Estos nodos, que constituyen las WSN, están equipados con sensores que permiten medir diversas variables, como temperatura, humedad, presión, luz y sonido, entre otras. Su función principal es recoger datos del entorno en el que están ubicados y posteriormente enviar estos datos de manera inalámbrica a una estación base o a un centro de procesamiento, donde se lleva a cabo su análisis y se toman decisiones pertinentes. (Cama-Pinto et al., n.d.)

Estas redes de sensores inalámbricos se aplican en una amplia variedad de contextos, desde la supervisión del entorno y la salud hasta la automatización industrial y la agricultura de precisión. Ejemplos específicos de aplicaciones abarcan la gestión de recursos naturales, el seguimiento de infraestructuras críticas, la monitorización de activos, la atención médica a distancia y la mejora de la eficiencia energética en edificios. Es relevante destacar que estos nodos suelen ser dispositivos de bajo consumo energético, ya que con frecuencia se ubican en lugares remotos o de difícil acceso, y suelen depender de baterías o fuentes de energía con recursos limitados.

La comunicación inalámbrica es esencial para que estas redes funcionen, ya que permite la interconexión de los nodos y la transmisión eficiente de datos en toda la red. En resumen, las redes de sensores inalámbricos (WSN) consisten en dispositivos equipados con sensores que capturan datos del entorno y los transmiten de forma inalámbrica, lo que facilita su procesamiento posterior y la toma de decisiones en diversas aplicaciones. Estas redes desempeñan un papel fundamental en el ámbito del Internet de las cosas (IoT), al permitir la recopilación de datos del mundo físico y su integración en sistemas

más amplios.

1.6.1.1 Topología de una Red WSN

La topología de la red en una Wireless Sensor Network (WSN) es fundamental. En una WSN, la comunicación inalámbrica implica que la topología se basa en la distribución física de los nodos y sus conexiones de comunicación. Esto se relaciona directamente con la continuidad de la investigación (Raghunathan et al., 2002)

1.6.1.2 Nodos de la WSN: Funcionamiento

Los nodos en una WSN son los componentes fundamentales que permiten la recopilación, transmisión y procesamiento de datos. Estos nodos pueden tener diferentes roles, siendo los más comunes el nodo máster y los nodos esclavos.

1.6.1.3 Nodo Máster

El nodo máster sigue siendo el núcleo central de la red. Sus funciones incluyen el control de la red, la transmisión a la plataforma central y la recopilación de datos. La comunicación directa punto a punto o punto a multipunto facilita la continuidad de esta investigación.

1.6.1.4 Nodo Esclavo

Los nodos esclavos, encargados de recopilar datos, transmiten información al nodo máster, colaborando en el enrutamiento de datos. Esto es crucial en la continuidad de la investigación, ya que permite la adquisición de datos de múltiples sensores y su transferencia eficiente.

1.6.2 Protocolo ESP NOW: Continuidad en la Comunicación Inalámbrica

El protocolo ESP-NOW, desarrollado por Espressif Systems (Rizal Isnanto et al., 2020), desempeña un papel central en la continuidad de esta investigación. Diseñado específicamente para microcontroladores y chips WiFi de la serie ESP8266 y ESP32,

ESP-NOW se destaca por su simplicidad y eficiencia en la comunicación directa entre dispositivos. A diferencia de los protocolos convencionales, como el estándar WiFi IEEE 802.11, ESP-NOW simplifica la comunicación eliminando capas innecesarias en el modelo OSI. Esto se traduce en una transmisión más rápida y una reducción de la congestión de paquetes en redes congestionadas. Con una velocidad de transferencia de 1 Mbps y la capacidad de cargar 250 Bytes de datos por transferencia, ESP-NOW ofrece varias ventajas clave:

Compatibilidad con Wi-Fi y Bluetooth LE: Coexiste con otras tecnologías inalámbricas, permitiendo una mayor flexibilidad en las comunicaciones.

- Eficiencia de Recursos: Requiere menos recursos CPU y flash, lo que es fundamental para dispositivos de baja potencia.
- Emparejamiento Rápido: Facilita la conexión de dispositivos “uno a muchos” y “de muchos a muchos”.
- Uso Independiente: Puede utilizarse para aprovisionamiento, depuración y actualizaciones de firmware.
- Eficiencia Energética: La sincronización de ventanas reduce significativamente el consumo de energía.

Dentro de este marco, se explorarán varias opciones de configuración para la transmisión de información entre dispositivos que utilizan ESP-NOW, incluyendo comunicación unidireccional y bidireccional.

1.6.2.1 Inicializar Direcciones MAC: Identificación de Nodos

Cuando los iniciadores se comunican con los respondedores, es crucial tener acceso a las direcciones MAC de los respondedores. Estas direcciones únicas y hexadecimales son esenciales en una red para identificar dispositivos. Aunque generalmente se queman en los dispositivos por el fabricante, es posible configurarlas manualmente. La capacidad de los nodos sensores para ajustar y adaptar las rutas de transmisión de datos en tiempo real, considerando cambios en la topología de la red, condiciones ambientales y requisitos de comunicación, es un elemento esencial en la continuidad de esta investigación. Este marco teórico, que se basa en la investigación previa y se expande para abordar los objetivos actuales, servirá como base sólida para la continuación de esta tesis. Los conceptos, tecnologías y protocolos aquí mencionados son esenciales para comprender

y desarrollar investigaciones posteriores.

1.6.3 Software

En el contexto de la tesis actual, se emplean diversas herramientas y plataformas que desempeñan un papel fundamental en el desarrollo de proyectos relacionados con el Internet de las cosas (IoT). Estas herramientas se utilizan para programar microcontroladores como el NodeMCU-ESP8266, administrar sistemas remotos y realizar tareas de red eficientemente, automatizar procesos mediante la conexión de diversas aplicaciones y servicios web, y recopilar, visualizar y analizar datos generados por dispositivos IoT. A continuación, se describen estas herramientas en el contexto de esta tesis:

1.6.3.1 Python

Para la implementación de algoritmos de aprendizaje automático, se utilizó Python, un lenguaje de programación de alto nivel ampliamente reconocido en la ciencia de datos y el aprendizaje automático. Python se destaca por su simplicidad y la disponibilidad de numerosas bibliotecas especializadas en el análisis de datos y la modelización predictiva. En particular, se aprovecharon las bibliotecas numpy y broadlink. Numpy es fundamental para el manejo y la manipulación de datos, lo que es esencial en el aprendizaje automático. Proporciona soporte para matrices y operaciones matemáticas eficientes en estas estructuras de datos. Finalmente, broadlink ofrece un módulo Python y CLI que facilita el control de dispositivos localmente. (Llerena Izquierdo, n.d.).

1.6.3.2 Docker

Docker es una plataforma de contenedores de código abierto que permite a los desarrolladores encapsular aplicaciones en contenedores estandarizados (“Docker”, n.d.). Esto aporta portabilidad y consistencia en distintos entornos que cuenten con Docker instalado. En el contexto de este proyecto, Docker desempeña un papel fundamental para garantizar la reproducibilidad y uniformidad en la ejecución del software de control inteligente. Mediante la encapsulación del software y sus dependencias en contenedores individuales, se asegura que el software se ejecute de manera idéntica

en cualquier entorno de despliegue. Además, Docker simplifica la implementación y actualización del software en dispositivos Raspberry Pi 3, bases de datos y el sistema de detección de personas utilizado en el proyecto. Esto se traduce en un funcionamiento eficiente y confiable del sistema, protegiéndolo contra posibles problemas y evitando problemas de agotamiento de memoria.

1.6.3.3 Arduino IDE

El uso del entorno de desarrollo integrado (IDE) de Arduino resulta altamente ventajoso para la programación de microcontroladores, como el NodeMCU-ESP8266. Este IDE se ha consolidado como una opción popular entre los desarrolladores debido a su facilidad de uso, el respaldo de una amplia comunidad de usuarios y su eficacia para agilizar proyectos. (Ismailov and Jo‘Rayev, 2022).

1.6.3.4 MobaXterm

Representa una solución integral para gestionar sistemas remotos y llevar a cabo tareas de red de manera eficiente. Esta plataforma combina múltiples herramientas en una única interfaz, simplificando la administración de sesiones SSH, telnet, RDP, VNC y otros protocolos. Además, ofrece características avanzadas como X11-forwarding, transferencia rápida de archivos y un emulador de terminal completo (“Características principales de MobaXterm”, n.d.), lo que la convierte en una elección idónea tanto para profesionales de TI como para usuarios principiantes.

1.6.4 Random forest classifier model: Aprendizaje por Refuerzo

El aprendizaje por refuerzo (RL) es un campo central de la inteligencia artificial que se enfoca en capacitar a un agente para tomar decisiones óptimas al interactuar con entornos desconocidos. Uno de los algoritmos más destacados en el ámbito del RL es el Q-Learning, diseñado para determinar una política óptima que permita al agente obtener la máxima recompensa acumulativa esperada.

Conceptos Clave

Entorno y Agente: En RL, se define un entorno con el cual el agente interactúa. En cada paso temporal, el agente se encuentra en un estado particular y elige una acción a

realizar. El entorno ejecuta la acción elegida y proporciona una recompensa al agente, además de indicar si la tarea se ha completado. Un episodio se representa como una secuencia de estados, acciones y recompensas.

Objetivo del Agente: En RL, el objetivo del agente es maximizar la recompensa acumulativa a lo largo del tiempo. Para lograrlo, el agente necesita descubrir una política óptima, que es una estrategia que dicta qué acción tomar en cada estado. (Amine, 2020).

Función de Valor-Q: El Q-Learning se basa en el aprendizaje y la actualización de la función de valor-Q (Q-Value). La función Q asigna un valor a cada par estado-acción, indicando la ventaja de llevar a cabo una acción en un estado específico. La tarea consiste en descubrir la función de valor-Q óptima que maximice las recompensas acumulativas.

1.6.4.1 Random Forest

Random Forest es una fusión de árboles predictivos, o clasificadores débiles, representando una adaptación del Bagging. En este método, se trabaja con una colección de árboles no correlacionados que se promedian (Hastie et al., 2001). Cada árbol en el bosque depende de los valores de un vector aleatorio extraído de la muestra de manera independiente y con la misma distribución.

La selección aleatoria de características para dividir cada nodo produce tasas de error comparables, e incluso superiores, al algoritmo AdaBoost (Freund and Schapire, 1996), y demuestra mayor robustez frente al ruido como clasificador fuerte. Las estimaciones internas supervisan el error, la fuerza y la correlación, utilizándose para evaluar la respuesta al aumento en el número de características utilizadas en la división. Estos cálculos internos también miden la importancia de las variables. Estas ideas son aplicables tanto a la clasificación como a la regresión.

En estos procedimientos, para el k-ésimo árbol se genera un vector aleatorio Θk, independiente de los últimos vectores aleatorios $\Theta 1$, …, $\Theta k–1$ pero con la misma distribución. Un árbol se desarrolla utilizando este conjunto y el conjunto de entrenamiento, lo que da como resultado un clasificador donde $h(x, \Theta k)$ es un vector de entrada.

Random Forest, basado en árboles de decisión, utiliza tests binarios en cada nodo, denominados “split”, para dirigir una muestra hacia una hoja que contiene la respuesta. Esta técnica simplifica problemas complejos al dividirlos en problemas más simples

durante la etapa de entrenamiento, optimizando los parámetros de las funciones de split a partir de las muestras de entrenamiento.

Durante la fase de entrenamiento, el algoritmo se dedica a mejorar los parámetros de las funciones de división utilizando las muestras de entrenamiento. Esto se realiza específicamente mediante la fórmula visualizada en la figura 1.1. Además, para calcular la ganancia de información, se emplea la fórmula representada en la figura 1.2. Estos procesos son fundamentales dentro del modelo Random Forest para optimizar la capacidad predictiva del algoritmo.

$$\theta_k{}^* = argmax_{\theta\ j \epsilon \tau_j} I_j$$

Figura 1.1: Fórmula de muestras de entrenamiento

Con este propósito, se emplea la función de ganancia de información siguiente:

$$I_j = H(j) - \sum_{i \in 1,2} \frac{|S_j^i|}{|S_j|} H(S_j^i)$$

Figura 1.2: Función de ganancia de información

En este contexto, S representa el conjunto de muestras presente en el nodo que se va a dividir, y S1 y S2 son los dos conjuntos resultantes de la división. La función evalúa la entropía del conjunto y su formulación varía según el tipo de problema que estamos abordando (Breiman, 2001).

1.7 Descripción de escenarios

El proyecto integrador está orientado a mejorar la irrigación del suelo en el área verde que se encuentra en la parte posterior del Laboratorio de Sistemas Telemáticos (LST). Para

ello se integrarán dos nodos maestros al sistema “WSN (Wireless Sensor Network) con estandarización de datos consumibles vía HayIoT” efectuado en el semestre anterior, con la finalidad de poder abarcar en mayor rango el área y obtener datos con más frecuencia que en conjunto con los datos de una API meteorológica se enviarán para que se utilice como fuente principal para entrenar el modelo de aprendizaje de máquina ”Random Forest” para la toma de decisiones del tiempo de riego de manera más eficaz y eficiente. Los sistemas de riego inteligente basados en Machine Learning se aplican en diversos escenarios, como la agricultura a gran escala, la agricultura de precisión, huertos, viñedos, jardinería residencial, áreas verdes urbanas, invernaderos, campos de golf, zonas áridas, agricultura tradicional, y plantaciones forestales. Estos sistemas optimizan la gestión del agua y mejoran el rendimiento de cultivos y plantas en una amplia gama de entornos.

CAPÍTULO 2

2. METODOLOGÍA

2.1 Materiales

En esta sección se describen los elementos físicos y lógicos utilizados en la implementación del sistema de riego basado en una Red de Sensores Inalámbricos Inteligente usando Machine Learning.

2.1.1 Componentes de Hardware

Sensores de humedad del suelo HD-38: El sensor de humedad del suelo HD-38 es un dispositivo utilizado para controlar la humedad del suelo en aplicaciones de riego automático de plantas. Este sensor consta de una sonda resistente a la corrosión y un módulo que proporciona salidas digital y analógica. Este sensor es ideal para la implementación de sistemas automatizados de riego, ya que permite monitorear y controlar la humedad del suelo de manera precisa.

Microcontrolador NodeMCU ESP8266: En el ámbito de la electrónica y el IoT, el NodeMCU ESP8266 es un microcontrolador extremadamente versátil. Con su amplia gama de funciones de programación y comunicación inalámbrica, se ha convertido en una opción popular para crear proyectos de IoT, domótica y aplicaciones de control remoto. La sencilla interfaz del ESP8266 NodeMCU y su considerable comunidad de desarrolladores proporcionan un incentivo para su desarrollo.

Fuente de Energía: El microcontrolador recibirá su energía a través del cable de 5V suministrado por la unidad de procesamiento de datos del control remoto, lo que

garantizará una fuente de energía constante para su operación. Esta configuración elimina cualquier preocupación relacionada con la duración de la batería y garantiza un funcionamiento ininterrumpido al estar conectado a una fuente de alta potencia.

Estación base (Raspberry Pi V4): Debido a su mayor capacidad de procesamiento y memoria, la estación base es un componente crítico en una WSN, ya que recopila y almacena datos de los nodos sensores. El centro de control de esta red es la Raspberry Pi, que gestiona la comunicación entre nodos y recoge datos sobre la humedad del suelo. Es una opción ideal para estas aplicaciones debido a la potencia de procesamiento y la flexibilidad de su Raspberry Pi Modelo 4.

Válvula Inteligente WIFI y Bluetooth: Pueden controlar el flujo de agua a través de una conexión inalámbrica y sincronizarse con asistentes de voz como Alexa y Google Assistant. Permiten la planificación y el control remoto mediante WIFI o Bluetooth. Algunos modelos también ofrecen la opción de realizar actualizaciones manuales de inmediato. Debido a su conexión precisa y versatilidad, esta válvula está diseñada específicamente para instalaciones automáticas, lo que facilita su integración con otros electrodomésticos y sistemas de domótica en el hogar.

Controlador de Válvula Inteligente: Es un dispositivo que se encarga de abrir y cerrar automáticamente el paso de líquidos, como el agua, y se destaca por su capacidad de comunicación inalámbrica a través de tecnologías como Bluetooth Low Energy (BLE) y Wi-Fi. El controlador permite realizar ajustes y monitorear de forma remota, optimizando el funcionamiento de tuberías de agua y otros procesos que requieren un control preciso del flujo de agua.

2.1.2 Componentes de Software

Arduino IDE: Es una aplicación que posibilita a los desarrolladores la creación, depuración y carga de código en las placas Arduino. Este software es versátil, ya que es compatible con diversos sistemas operativos, realiza la detección automática de la placa conectada y proporciona información detallada sobre el uso de memoria Flash y SRAM. Además, permite la carga de programas a través de redes para las placas Arduino Yun. Para

obtenerlo, puedes descargarlo desde el sitio web oficial de Arduino, y es compatible con sistemas operativos como Windows, macOS y Linux.

MobaXterm: Es una aplicación integral para la gestión remota que unifica múltiples capacidades en una única interfaz. Ofrece conexiones seguras a sistemas remotos mediante diversos protocolos, facilita la administración de sesiones, permite la ejecución de aplicaciones gráficas y proporciona herramientas relacionadas con redes, desarrollo y transferencia de archivos. Su versatilidad y disponibilidad en versiones gratuitas y de pago la convierten en una herramienta valiosa tanto para profesionales de TI como para administradores de sistemas y desarrolladores.

Tinytuya: Es una librería en Python que simplifica la gestión de dispositivos compatibles con Tuya. Estos dispositivos pueden funcionar tanto en la red local como a través de la nube utilizando la API de TuyaCloud. Con esta herramienta, es posible administrar los dispositivos Tuya directamente sin necesidad de utilizar la nube, lo que permite supervisar su estado y enviar comandos.

ThingSpeak: Es una plataforma IoT que simplifica la obtención, evaluación y representación de datos generados por dispositivos conectados a la red. Su enfoque principal es la supervisión y gestión de sistemas. ThingSpeak es utilizado en proyectos relacionados con IoT y el seguimiento de sensores que monitorean condiciones ambientales. Para obtener información actualizada, se sugiere visitar el sitio web oficial de la plataforma.

HayIoT: El propósito de la plataforma creada en ESPOL es la estandarización para consolidar todos los módulos y sensores de IoT en una única aplicación.

Tomorrow.io: Plataforma que proporciona servicios y datos meteorológicos. Emplea tecnologías de vanguardia, como radar e inteligencia artificial, con el fin de suministrar datos meteorológicos exactos y específicos en tiempo real. Además, tiene disponible su API meteorológica que posibilita a los desarrolladores integrar datos en sus propios proyectos.

Python: Lenguaje de programación versátil de amplio uso en diversas esferas, encuentra aplicación en el desarrollo de software, abarcando desde aplicaciones de escritorio hasta sistemas embebidos. Python contiene una amplia gama de bibliotecas para el entrenamiento de los distintos lenguajes de máquina. Scikit-learn destaca como la biblioteca más frecuentemente empleada para llevar a cabo el entrenamiento de Random Forests (bosques aleatorios). Se trata de una librería de aprendizaje automático de código abierto que ofrece herramientas eficaces y sencillas para el análisis de datos y la construcción de modelos predictivos.

2.2 Métodos

2.2.1 Diseño de la Red de Sensores Inalámbricos (WSN)

En la primera etapa, se inicia con la planificación y configuración de la Red de Sensores Inalámbricos (WSN), que involucra la definición de la cantidad apropiada de nodos sensores a añadir. Durante este proceso, se determinan los parámetros de comunicación y se realizan las configuraciones necesarias en los microcontroladores NodeMCU 8266 para habilitar la transmisión inalámbrica de los datos recopilados por los sensores de humedad.

2.2.1.1 Protocolo de Comunicación en la red inalámbrica de sensores

El protocolo de comunicación a emplear será ESP-NOW, que permite una comunicación punto a punto directa entre los nodos. Esto significa que cada nodo principal debe tener conocimiento de la dirección MAC única del nodo esclavo o coordinador para establecer una comunicación exitosa.

2.2.1.2 Diseño del Nodo Esclavo o Nodo Coordinador

Los nodos de coordinación, también conocidos como nodos esclavos, desempeñan un papel crucial en la red ESP-NOW. Su función principal es actuar como intermediarios para recibir los datos enviados por los nodos maestros y facilitar la transmisión hacia la

estación base u otros nodos centrales encargados de recopilar los datos.

Configuración del nodo coordinador:
El nodo de coordinación NodeMCU8266 se configurará como se visualiza en la figura 2.1 para establecer una conexión mediante el protocolo ESP-NOW y habilitar la comunicación inalámbrica con los nodos sensores. Para llevar a cabo esta configuración, es esencial contar con la dirección MAC, y se utilizará el siguiente código en el microcontrolador designado como esclavo:

```
#include <ESP8266Wifi.h>
void setup() {
  Serial.begin(115200); // Setup Serial Monitor
  WiFi.mode(WIFI_MODE_STA); // Put ESP32 into Station mode
  Serial.print("MAC Address: ");
  Serial.println(WiFi.macAddress()); // Print MAC Address to Serial monitor }
void loop() { }
```

Figura 2.1: Configuración del nodo coordinador

2.2.1.3 Diseño Nodo Sensor

Los nodos sensores, tienen la tarea de recopilar los niveles de humedad en el suelo. Se utilizarán dos de estos nodos sensores, y cada uno de ellos se conectará al módulo sensor de humedad del suelo HD-38. Para la conexión de los sensores de humedad del suelo HD38 a los módulos NodeMCU8266, se empleará la salida analógica A0. La alimentación para estos módulos provendrá de un enchufe que suministrará una tensión de 5V DC. El microcontrolador suministrará la alimentación necesaria de 3.3V para los sensores de humedad. La ubicación de los sensores será en cumplimiento de una cuadricula con respecto a los tres sensores ubicados en el anterior periodo.

2.2.1.4 Estación Base

Conexión entre el nodo de coordinación y la RPI V4:
Se planea establecer la comunicación entre el nodo coordinador y la RPI V4 mediante una conexión física directa, específicamente a través de lo que se conoce como comunicación serial. En este proceso, cada vez que el nodo coordinador reciba datos, estos vendrán acompañados de un identificador. Esta etiqueta especial ayudará a identificar de dónde provienen los valores que se están adquiriendo. Por su parte, la RPI V4 estará a cargo

de leer todas las entradas enviadas por el nodo coordinador. Este proceso permitirá la incorporación automática de nodos, lo que significa que nuevos nodos podrán ser fácilmente agregados al sistema sin necesidad de realizar configuraciones adicionales.

Almacenamiento y transmisión de datos:
La Raspberry Pi será responsable de enviar los datos recopilados a la plataforma HayloT para que sean almacenados y presentados visualmente. El funcionamiento de la válvula inteligente será controlado mediante solicitudes POST dirigidas a la configuración de los microservicios encargados de supervisarlo.

2.2.1.5 Control de válvula inteligente

Una válvula inteligente TinyTuya puede ser controlada mediante un modelo de aprendizaje que utiliza un protocolo local compatible con Python, como MQTT o HTTP. En este proceso, se establece inicialmente una conexión con la válvula a través del protocolo local para enviar comandos y recibir datos. Posteriormente, un modelo de aprendizaje desarrollado en Python puede analizar información pertinente, como datos de sensores o patrones de uso, para generar decisiones o predicciones. Estas decisiones pueden ser traducidas en comandos a través del protocolo local, lo que permite al modelo influir en el control de la válvula en base a sus predicciones o análisis. Esta dinámica crea un sistema integrado donde el modelo de aprendizaje interactúa con la válvula inteligente TinyTuya empleando un protocolo compatible con Python. Este enfoque facilita el control y la adaptación de la válvula, alineándose con la lógica y los análisis del modelo de aprendizaje.

2.2.1.6 Servicio de reconexión a la VPN

Dado que se utilizó un plan gratuito de “ProtonVPN” para el envío de datos a la red, la VPN no puede estar siempre activa para la transmisión, por eso fue necesaria la creación de un script como se observa en la figura 2.2 el cual verificará si hay una conexión activa en la VPN, caso contrario se procedería a ejecutar un comando que se encargaría de conectar la VPN de nuevo al servidor más rápido disponible.

Además fué crucial dejar corriendo la reconexión como un service para su respetiva automatización, integración y gestión como se visualiza en la figura 2.3.

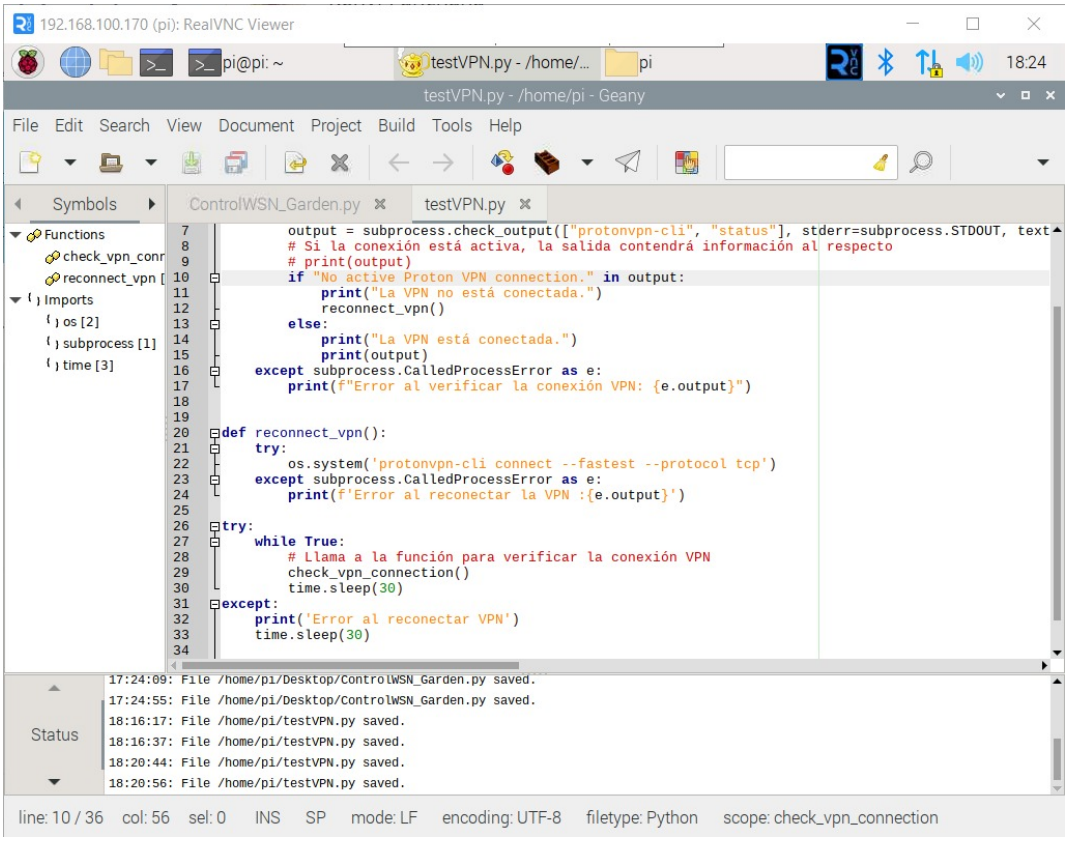


Figura 2.2: Script para reconectar la VPN

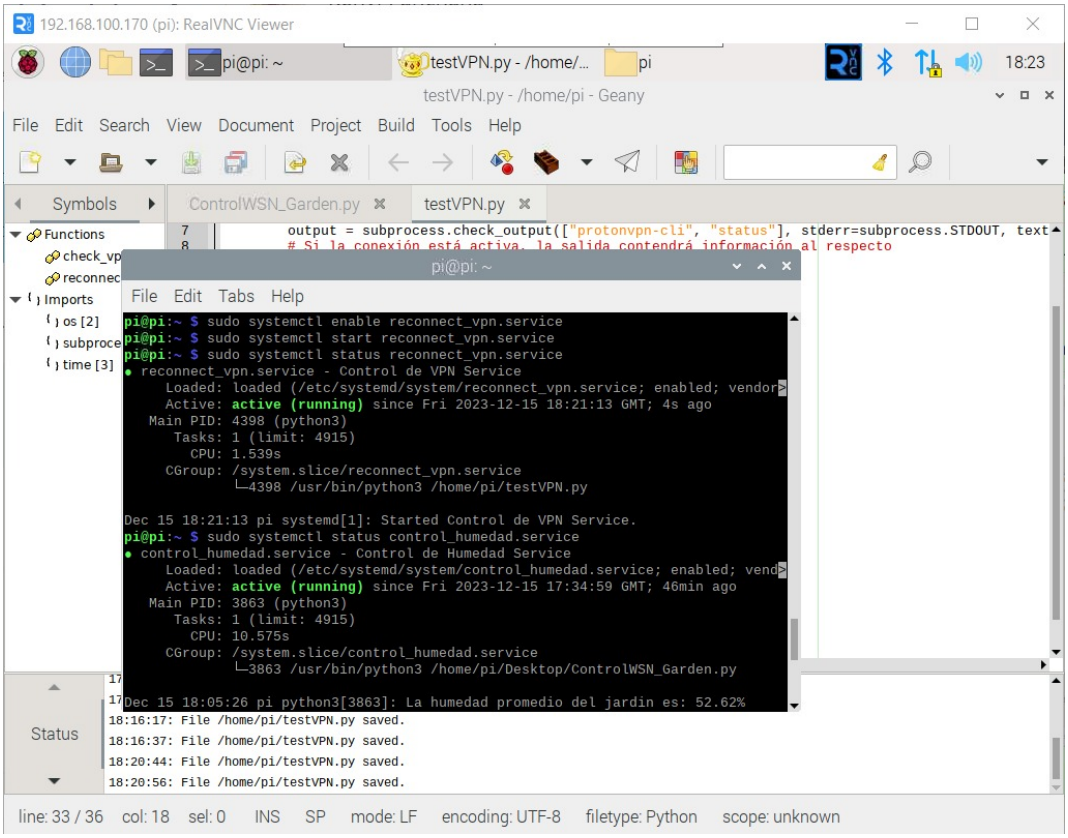
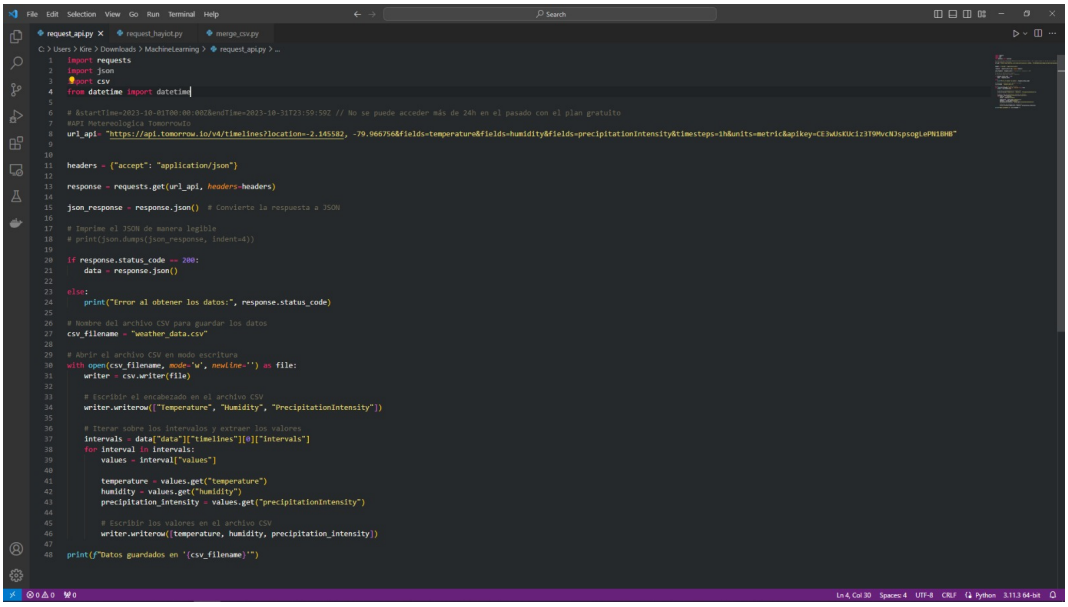


Figura 2.3: El script de reconexión corriendo como un service de Linux

2.3 Obtención de datos

2.3.1 Datos meteorológicos

Las variables meteorológicas como la humedad y temperatura del ambiente más las precipitaciones serán fundamentales para el entrenamiento del modelo de predicción. Para ello se requirió hacer uso de una interfaz de programación de Aplicaciones para obtener los datos de una manera más sencilla. Es necesario que los usuarios se registren en la plataforma web, adquirir una clave API, autenticar sus solicitudes y enviar consultas mediante HTTP con parámetros particulares, todo aquello mediante código como se observa en la figura 2.4. Las respuestas, que se presentan en formato JSON, deben ser tratadas y examinadas de acuerdo con los requisitos de la aplicación.

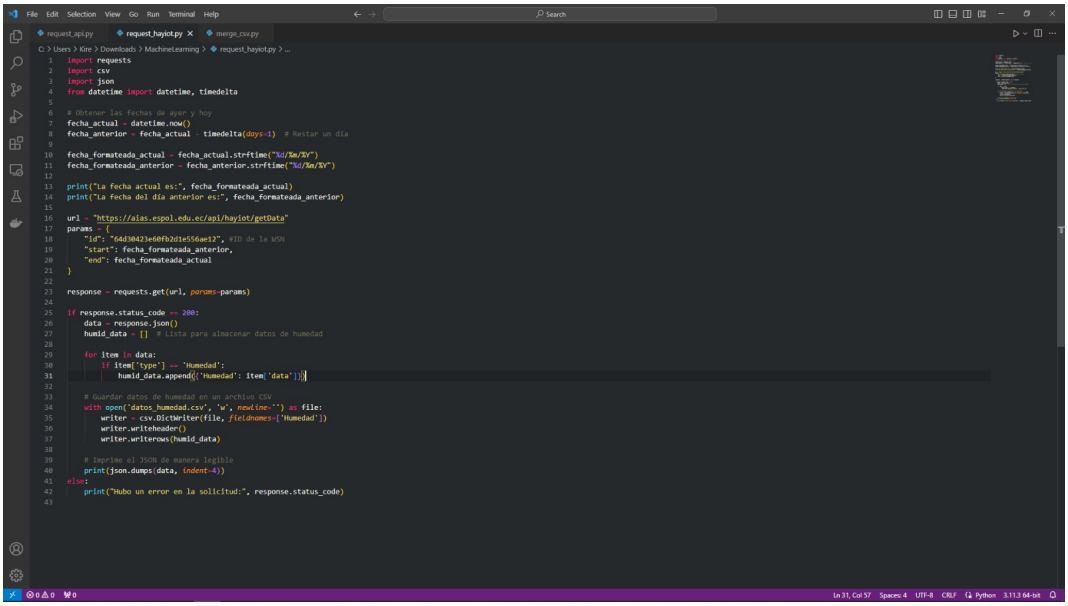


```python
import requests
import json
import csv
from datetime import datetime

# &startTime=2023-10-01T00:00:00Z&endTime=2023-10-31T23:59:59Z // No se puede acceder más de 24h en el pasado con el plan gratuito
#API Meteorologica Tomorrow.io
url_api= "https://api.tomorrow.io/v4/timelines?location=-2.145582, -79.966756&fields=temperature&fields=humidity&fields=precipitationIntensity&timesteps=1h&units=metric&apikey=CE3wUsKUci3T9MvcNJspsogLePN1BHB"


headers = {"accept": "application/json"}

response = requests.get(url_api, headers=headers)

json_response = response.json()  # Convierte la respuesta a JSON

# Imprime el JSON de manera legible
# print(json.dumps(json_response, indent=4))

if response.status_code == 200:
    data = response.json()

else:
    print("Error al obtener los datos:", response.status_code)

# Nombre del archivo CSV para guardar los datos
csv_filename = "weather_data.csv"

# Abrir el archivo CSV en modo escritura
with open(csv_filename, mode='w', newline='') as file:
    writer = csv.writer(file)

    # Escribir el encabezado en el archivo CSV
    writer.writerow(["Temperature", "Humidity", "PrecipitationIntensity"])

    # Iterar sobre los intervalos y extraer los valores
    intervals = data["data"]["timelines"][0]["intervals"]
    for interval in intervals:
        values = interval["values"]

        temperature = values.get("temperature")
        humidity = values.get("humidity")
        precipitation_intensity = values.get("precipitationIntensity")

        # Escribir los valores en el archivo CSV
        writer.writerow([temperature, humidity, precipitation_intensity])

print(f"Datos guardados en '{csv_filename}'")
```

Figura 2.4: Solicitud de los datos desde la API

2.3.2 Obtención de datos HayloT

El valor de humedad del suelo que se obtiene mediante los nodos incorporados se lo obtiene desde la plataforma HayloT mediante consultas get veáse en la figura 2.5 para posteriormente ser llevados y tratados en Python.



```python
import requests
import csv
import json
from datetime import datetime, timedelta

# Obtener las fechas de ayer y hoy
fecha_actual = datetime.now()
fecha_anterior = fecha_actual - timedelta(days=1)  # Restar un día

fecha_formateada_actual = fecha_actual.strftime("%d/%m/%Y")
fecha_formateada_anterior = fecha_anterior.strftime("%d/%m/%Y")

print("La fecha actual es:", fecha_formateada_actual)
print("La fecha del día anterior es:", fecha_formateada_anterior)

url = "https://aias.espol.edu.ec/api/hayiot/getData"
params = {
    "id": "64d30423e60fb2d1e5564ae12",  #ID de la WSN
    "start": fecha_formateada_anterior,
    "end": fecha_formateada_actual
}

response = requests.get(url, params=params)

if response.status_code == 200:
    data = response.json()
    humid_data = []  # Lista para almacenar datos de humedad

    for item in data:
        if item['type'] == 'Humedad':
            humid_data.append({'Humedad': item['data']})

    # Guardar datos de humedad en un archivo CSV
    with open('datos_humedad.csv', 'w', newline='') as file:
        writer = csv.DictWriter(file, fieldnames=['Humedad'])
        writer.writeheader()
        writer.writerows(humid_data)

    # Imprime el JSON de manera legible
    print(json.dumps(data, indent=4))
else:
    print("Hubo un error en la solicitud:", response.status_code)
```

Figura 2.5: Solicitud de los datos desde HayloT

La obtención de los datos de ambas fuentes se realiza de manera separada para luego unificar toda la información en un solo archivo csv con el siguiente script realizado en

Python como se muestra en la figura 2.6.

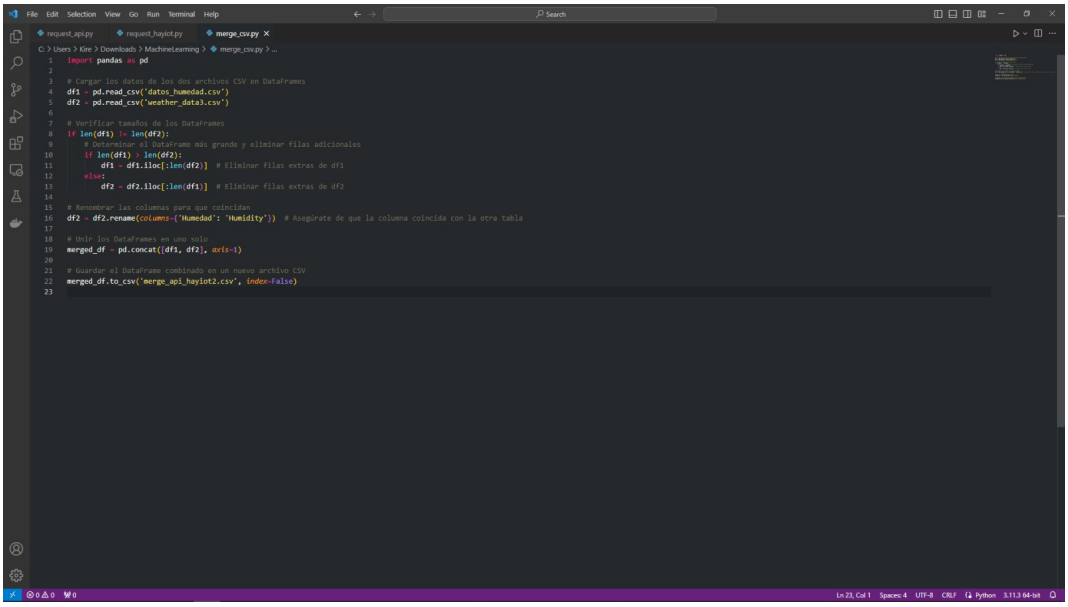


Figura 2.6: Código para unificar los datos en un solo archivo

2.3.2.1 Entrenamiento del modelo random forest

El algoritmo de aprendizaje automático conocido como Random Forest destaca por su capacidad para clasificar conjuntos de datos y su versatilidad para abordar características no lineales. En el proceso de entrenamiento del modelo, se emplearán cuatro variables previamente preparadas y estructuradas. Estas variables se dividirán en conjuntos de entrenamiento y prueba. Posteriormente, se procederá a la creación del modelo utilizando la biblioteca scikit-learn. Este modelo se entrenará según parámetros predefinidos para realizar predicciones, seguido de una evaluación exhaustiva del rendimiento del modelo. La elección de Random Forest para el sistema de riego inteligente se apoya en consideraciones técnicas específicas: Random Forest sobresale por su precisión y adaptabilidad al abordar múltiples variables, como la temperatura, humedad del suelo y ambiente, y precipitaciones. Esta capacidad permite la captura de relaciones complejas entre estas variables, generando predicciones más precisas. Su habilidad para manejar múltiples características sin requerir codificación especial es esencial, especialmente en situaciones con varias variables como en este caso. Esto otorga al modelo versatilidad y eficiencia al trabajar con datos de naturaleza numérica y categórica.

La robustez de Random Forest frente a datos ruidosos y valores atípicos es otro aspecto

destacado. Al considerar múltiples árboles y promediar los resultados, el modelo reduce el impacto de estos elementos, mejorando la estabilidad del sistema. La eficiencia computacional del modelo es crucial, permitiendo trabajar con grandes conjuntos de datos y entrenar rápidamente. Esto es esencial en un sistema de riego que necesita tomar decisiones en tiempo real.

2.4 Esquema de diseño propuesto

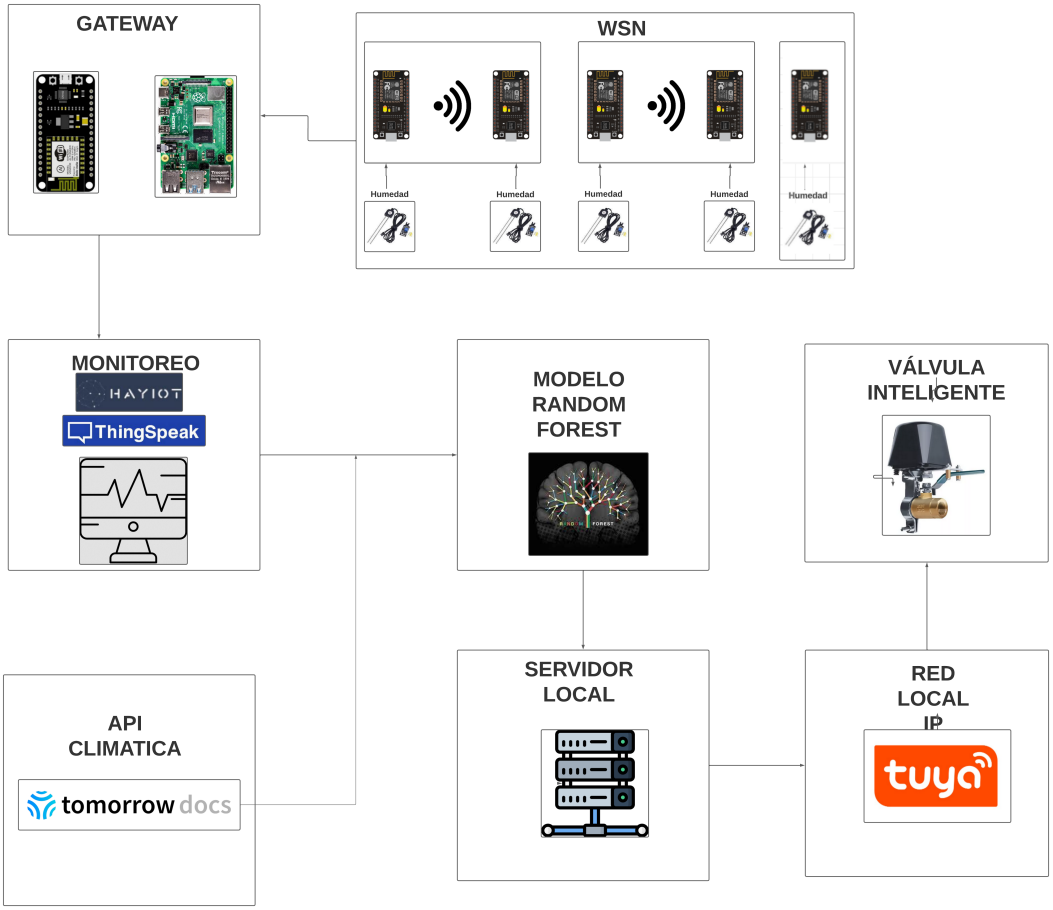


Figura 2.7: Esquema de diseño propuesto

En la figura 2.7 se presenta el diseño propuesto para la WSN con la implementación de machine learning. Este diseño incluirá dos nodos sensores adicionales de humedad del suelo, integrándolos a los ya existentes, dando un total de 5 nodos sensores. Además, se contará con un nodo coordinador NodeMCU ESP8266 conectado a través del protocolo ESP-NOW para la trasmisión de datos entre ellos, y se establecerá una conexión adicional mediante USB hacia una RPI V4. La RPI V4 se encargará que

los datos recolectados se almacenen y envíen a la nube a través de la plataforma de estandarización HayIoT. Posteriormente se necesitarán los datos de la humedad del suelo además de los de datos de la plataforma Tomorrow.io para hacer el entrenamiento del modelo random forest con tales variables. Para finalizar una vez entrenado y validado el modelo realizado se implementará un script final para hacer el control de la válvula inteligente de tal manera que en base a la toma de datos en tiempo real y al entrenamiento realizado con anterioridad tome la decisión de encender (ON) u apagar (OFF) la válvula con la finalidad de mejorar el proceso de riego del jardín.

2.5 Esquema de implementación

En la figura 2.8 se visualiza el respectivo esquema de implementación que tiene como objetivo emplear datos de una API climática (tomorrow docs) y la plataforma HayIoT para clasificar las condiciones climáticas. Se establecerán métricas de evaluación, como la precisión, para medir el desempeño del modelo. En primera instancia, se recolectarán datos de la API climática, incluyendo variables como temperatura, humedad del ambiente y precipitación. Asimismo, se accederá a la plataforma HayIoT para obtener datos de humedad del suelo registrados por los sensores. Una vez obtenidos, se procederá a unificar y depurar los datos, asegurando que estén en un formato apropiado para el modelo.

El diseño del modelo Random Forest se llevará a cabo utilizando una biblioteca de machine learning en Python. Se importarán las librerías necesarias, se cargarán los datos preparados y se dividirán en conjuntos de entrenamiento y prueba para el modelo. A continuación, se iniciará el entrenamiento del modelo. Se creará una instancia del modelo Random Forest, se ajustarán los hiperparámetros (número de árboles, profundidad, etc.) y se entrenará utilizando los datos de entrenamiento. Durante este proceso, se evaluará el modelo para evitar el sobreajuste.

En la fase de validación del modelo, se utilizará el conjunto de prueba para evaluar la precisión y otras métricas del modelo entrenado. Los resultados obtenidos permitirán

analizar y ajustar los hiperparámetros si es necesario para mejorar el rendimiento. La integración del modelo en el sistema se logrará desarrollando un script o módulo que utilice el modelo entrenado para realizar predicciones climáticas en tiempo real. Se configurará esta integración para recibir datos de la API y la plataforma HayIoT, permitiendo la realización de predicciones basadas en esta información.

Se llevarán a cabo pruebas exhaustivas del sistema integrado, incluyendo escenarios límite y situaciones inesperadas. Se realizarán ajustes y optimizaciones según los resultados obtenidos durante estas pruebas, con el objetivo de mejorar la precisión del modelo.

Finalmente, el sistema será implementado en producción y se establecerá un monitoreo constante para asegurar su correcto funcionamiento. Se realizarán actualizaciones y mejoras según sea necesario para mantener la precisión del modelo y el rendimiento general del sistema.

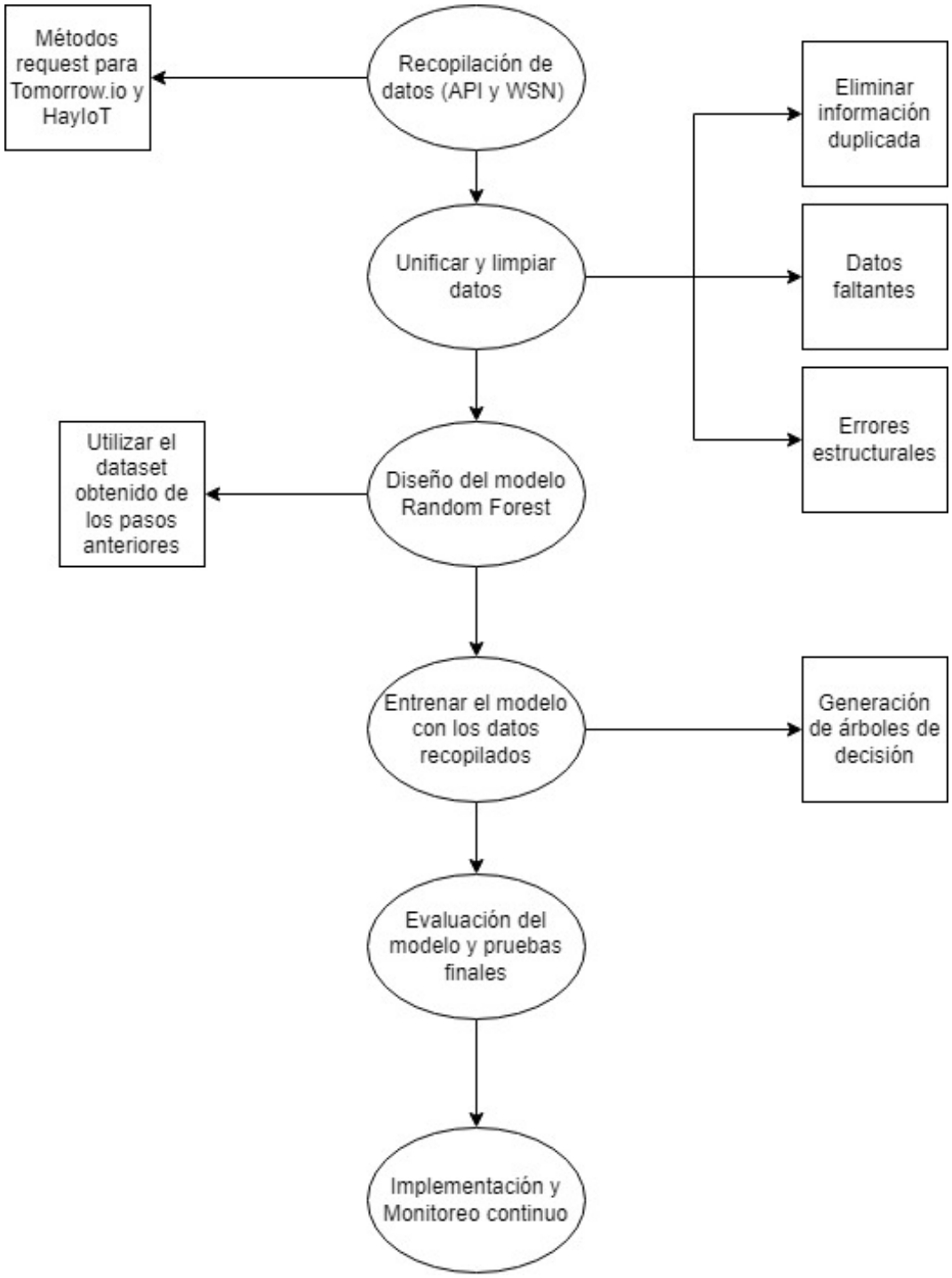


Figura 2.8: Esquema de implementación

CAPÍTULO 3

3. PRUEBAS Y RESULTADO

El sistema de Riego Inteligente se evalúa exhaustivamente en un entorno particular con el objetivo de realizar un análisis detallado de la solución propuesta. En este capítulo, se abordarán aspectos esenciales, describiendo el contexto de evaluación y los resultados obtenidos en los diversos modos de operación.

3.1 Pruebas

3.1.1 Pruebas de la trasmisión de datos

Parte esencial de la finalidad del proyecto fue la integración de dos nodos sensores al sistema previamente desarrollado en el proyecto integrador anterior. Para lograr este propósito, se llevó a cabo la integración física y configuración correspondiente en el entorno de programación Arduino IDE de los nuevos nodos. Esto aseguró una convergencia efectiva con los tres nodos ya existentes en el sistema.

Una etapa esencial fue establecer en los códigos de todos los nodos sensores la dirección MAC del nodo receptor o coordinador. Esta configuración permitió que el valor capturado se enviará como señal al sensor correspondiente.

Tras la conclusión de esta fase, se procedió a la validación del código del nodo receptor de las señales. La finalidad de esta validación fue confirmar que el valor promedio resultante estuviera en concordancia con el número total de sensores presentes en el sistema. Este enfoque asegura la obtención de una estimación representativa de la humedad del suelo en la zona de estudio, proporcionando un valor medio preciso que refleja las condiciones hídricas del entorno analizado.

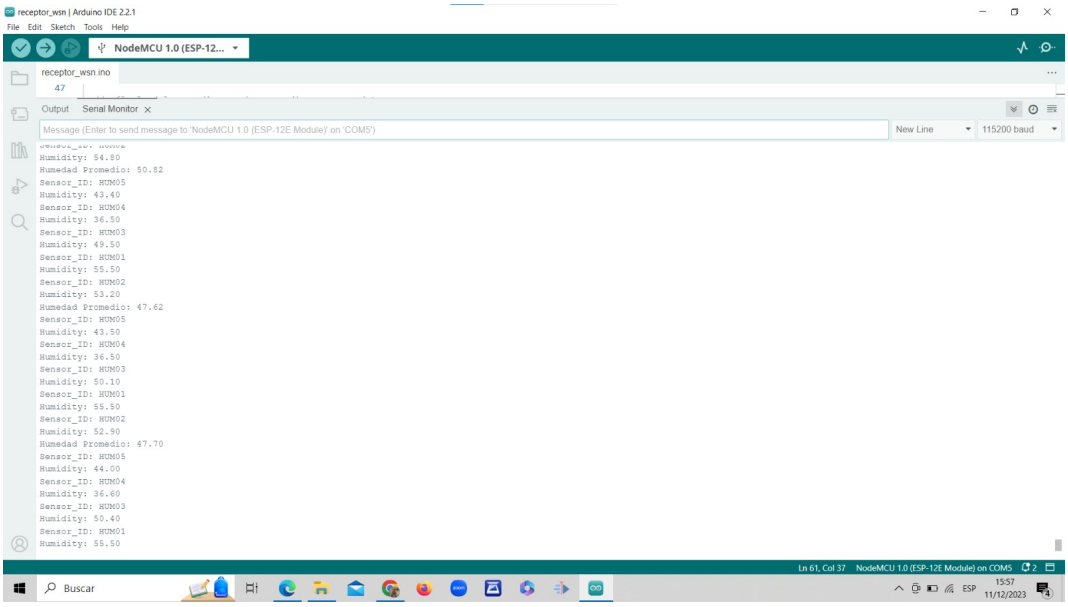


Figura 3.1: Nodo coordinador monitoreando los dos nuevos nodos 4 y 5

En la figura 3.1 podemos observar que la nueva información proporcionada por los nodos cuatro y cinco ha empezado a llegar al nodo coordinador, señal de una integración exitosa de los nuevos nodos.

En la implementación del sistema, se configura el acceso a la Raspberry Pi (RPI) utilizando Mobaxterm y una conexión SSH, aprovechando la dirección IP conocida de la RPI. Este enfoque simplifica la conexión USB hacia el nodo coordinador, lo que facilita la lectura de datos capturados y el cálculo del valor promedio. Posteriormente, este valor se envía a HayIoT para su monitoreo y análisis.

En el código de la RPI, se realiza una configuración inicial importando las librerías necesarias, como serial, time y requests. Se ajusta el puerto serial para que tenga el mismo baud rate que el nodo coordinador ESP8266, y se definen variables booleanas para controlar la apertura y cierre de la válvula, así como el valor de humedad configurado. La transmisión de datos a HayIoT se lleva a cabo en formato JSON para optimizar la eficiencia y legibilidad de la comunicación. La función senddatatoHayIoT se encarga de realizar una solicitud POST y verificar la respuesta del servidor.

Se implementa un bucle infinito para recibir y procesar datos desde el puerto serial. Dentro de este bucle, se realizan varias operaciones, como la impresión de datos en la salida estándar, el almacenamiento en un archivo de texto y la posibilidad de enviarlos a HayIoT.

Además, se gestiona las excepciones para cerrar la conexión serial de manera adecuada y proporcionar notificaciones claras sobre el cierre exitoso. Al mismo tiempo, se realiza la transmisión de datos a través de la conexión USB a la Raspberry Pi 4, generando un archivo de respaldo para asegurar la integridad de los datos en caso de fallos de conexión a la red.

Durante las pruebas, se establece un intervalo de 500 segundos para la transmisión de datos, lo que garantiza una actualización continua en el panel de control de HayIoT. Este enfoque permite realizar un seguimiento en tiempo real de los datos recopilados, mejorando la monitorización del sistema.

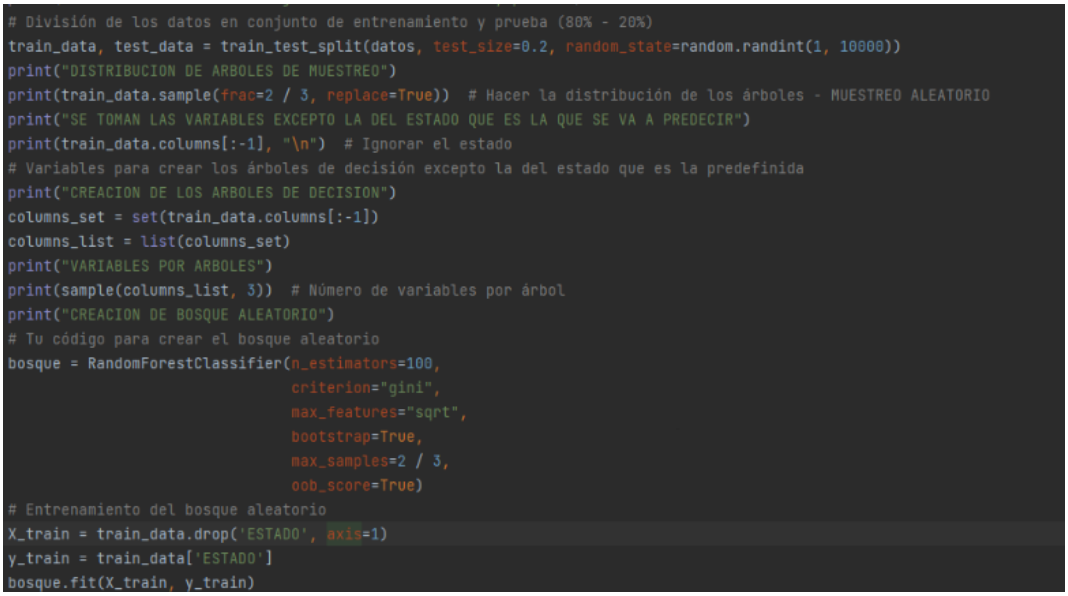
3.1.2 Pruebas de la trasmisión de datos

3.1.2.1 Preparación de Datos

Al consolidar las variables en un archivo CSV, la extracción de datos en las siguientes etapas se simplifica. Inicialmente se carga datos desde un archivo CSV, crea una nueva columna llamada 'ESTADO' basada en ciertas condiciones, y luego imprime los resultados para su análisis. La columna 'ESTADO' clasifica los datos según las condiciones establecidas en el código.

3.1.2.2 Algoritmo

Como se puede observar en la figura 3.2, el proceso comienza dividiendo el conjunto de datos original en conjuntos de entrenamiento y pruebas. El 80 % de los datos se asigna al conjunto de entrenamiento, mientras que el 20 % restante se reserva para las pruebas. Se utiliza el parámetro random_state para garantizar que la división aleatoria sea consistente. Luego, se realiza un muestreo aleatorio con reemplazo en el conjunto de entrenamiento, extrayendo el 2/3 de los datos. A continuación, se crea un objeto RandomForestClassifier con parámetros específicos, como 100 árboles, el criterio "gini" y el uso del 2/3 de las muestras para entrenar cada árbol. Finalmente, se separan las características (X_train) y la variable objetivo (y_train) para entrenar el bosque aleatorio utilizando el método "fit()". Este enfoque estructurado y detallado garantiza un proceso coherente y transparente en la preparación del modelo.



```
# División de los datos en conjunto de entrenamiento y prueba (80% - 20%)
train_data, test_data = train_test_split(datos, test_size=0.2, random_state=random.randint(1, 10000))
print("DISTRIBUCION DE ARBOLES DE MUESTREO")
print(train_data.sample(frac=2 / 3, replace=True))  # Hacer la distribución de los árboles - MUESTREO ALEATORIO
print("SE TOMAN LAS VARIABLES EXCEPTO LA DEL ESTADO QUE ES LA QUE SE VA A PREDECIR")
print(train_data.columns[:-1], "\n")  # Ignorar el estado
# Variables para crear los árboles de decisión excepto la del estado que es la predefinida
print("CREACION DE LOS ARBOLES DE DECISION")
columns_set = set(train_data.columns[:-1])
columns_list = list(columns_set)
print("VARIABLES POR ARBOLES")
print(sample(columns_list, 3))  # Número de variables por árbol
print("CREACION DE BOSQUE ALEATORIO")
# Tu código para crear el bosque aleatorio
bosque = RandomForestClassifier(n_estimators=100,
                                criterion="gini",
                                max_features="sqrt",
                                bootstrap=True,
                                max_samples=2 / 3,
                                oob_score=True)
# Entrenamiento del bosque aleatorio
X_train = train_data.drop('ESTADO', axis=1)
y_train = train_data['ESTADO']
bosque.fit(X_train, y_train)
```

Figura 3.2: Imagen de división de datos

3.1.2.3 Evaluar el rendimiento del modelo de bosque aleatorio (Random Forest) en el conjunto de prueba

Evaluar el rendimiento en un conjunto de prueba es una práctica fundamental para garantizar que el modelo de Bosque Aleatorio funcione de manera efectiva en situaciones del mundo real y para evitar problemas como el sobreajuste.

Matriz de confusión

La matriz de confusión mostrada en la figura 3.3 presenta una representación detallada de las predicciones del modelo, clasificando las instancias en cuatro cuadrantes distintos:

1. Negativo Verdadero (cuadrante superior izquierdo):

Representa 15 instancias donde tanto en la realidad como en la predicción se clasificaron correctamente como ‘OFF’ (o no ‘ON’). Estos son casos donde el modelo acertó al prever que no se cumplían las condiciones especificadas para ‘ON’.

2. Falso Positivo (cuadrante superior derecho): En este cuadrante se registran 3 valores. Indica que el modelo clasificó incorrectamente instancias como ‘ON’ cuando, en realidad, debería haber sido clasificada como ‘OFF’.

3. Falso Negativo (cuadrante inferior izquierdo): Se obtuvo 1 valor en este cuadrante. Significa que hubo una instancia clasificadas incorrectamente como ‘OFF’ cuando deberían haber sido clasificadas como ‘ON’.

4. Verdadero Positivo (cuadrante inferior derecho): Representa 41 instancias donde

tanto en la realidad como en la predicción se clasificaron correctamente como ‘ON’. Estos son casos en los que el modelo acertó al anticipar que se cumplían las condiciones especificadas para ‘ON’. Los términos “Verdadero Positivo”, “Negativo Verdadero”, “Falso Positivo” y “Falso Negativo” describen cómo el modelo clasifica las instancias en relación con la realidad. En este contexto, el modelo muestra una alta precisión al prever correctamente los casos ‘ON’, evidenciado por el número significativo de Verdaderos Positivos.

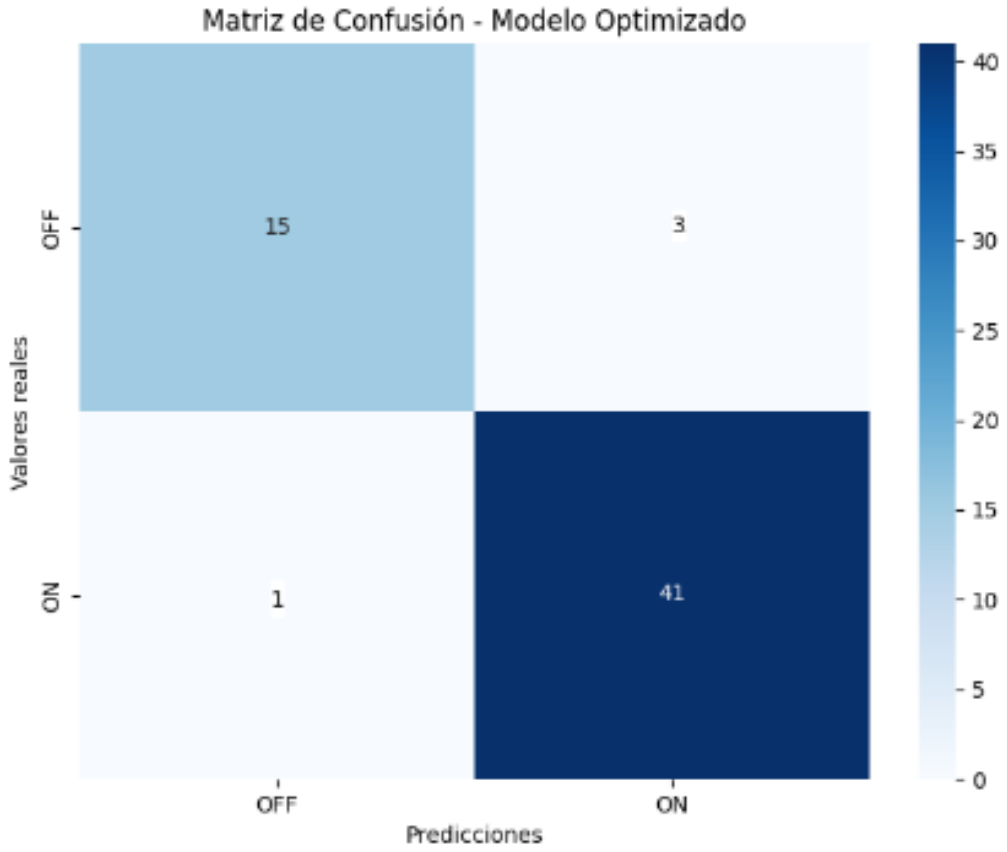


Figura 3.3: Matriz de Confusión

El reporte de clasificación

El informe de clasificación de la figura 3.4 brinda una detallada evaluación del desempeño del modelo en la clasificación de las clases ‘OFF’ y ‘ON’. A continuación, se presenta una interpretación más a fondo de las métricas esenciales:

1. Precisión: La precisión para la clase ‘OFF’ alcanza el valor de 0.98, denotando que la gran mayoría de las instancias clasificadas como ‘OFF’ por el modelo son, de hecho, de

esta clase. Asimismo, la precisión para la clase ‘ON’ es de 0.98, reflejando la precisión de las predicciones positivas asociadas con esta categoría.

2. Recall: En lo que respecta a la clase ‘OFF’, el recall se sitúa en 0.96, indicando que el modelo logra capturar la mayoría de los casos reales de esta clase. Por otro lado, el recall para la clase ‘ON’ asciende a 0.97, lo que sugiere que el modelo identifica la mayoría de los casos reales pertenecientes a dicha categoría.

3. F1-Score: El F1-Score, que representa la armónica entre precisión y recall, exhibe un valor de 0.95 para ‘OFF’ y de 0.98 para ‘ON’. Este indicador proporciona una evaluación equilibrada entre la precisión y la capacidad del modelo para identificar instancias relevantes.

4. Support: Representa el número de instancias de cada clase en el conjunto de prueba. Se constatan 15 instancias de ‘OFF’ y 41 instancias de ‘ON’. Esta métrica ofrece contexto respecto a la distribución de clases en el conjunto de datos evaluado.

Reporte de Clasificación - Modelo Optimizado:

	precision	recall	f1-score	support
OFF	0.98	0.96	0.95	15
ON	0.95	0.97	0.98	41
accuracy			0.94	56

Figura 3.4: Reporte de Clasificación

Evaluación Cruzada

La puntuación de precisión (accuracy) obtenida mediante validación cruzada es del 94.81 por ciento como se puede ver en la figura 3.5, con un intervalo de confianza de +/- 0.0039 por ciento. Este resultado indica un alto nivel de exactitud en las predicciones del modelo durante la validación cruzada, demostrando su consistencia y eficacia en la clasificación de los datos.

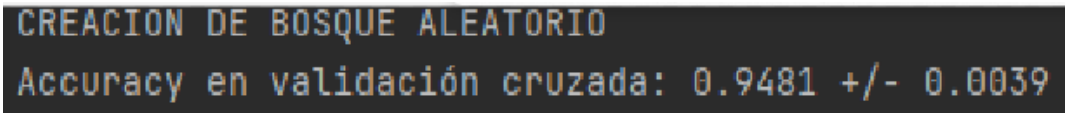


Figura 3.5: Reporte de Clasificación

3.1.3 Pruebas del modelo de aprendizaje en la válvula inteligente

Se ha desarrollado un script en Python con el propósito de evaluar la eficacia deL modelo de machine learning mediante el control de La válvula, destacando su aplicación práctica en el ámbito agrícola. La configuración inicial del entorno se realiza mediante la inclusión de bibliotecas claves, como 'tinytuya' para interactuar con dispositivos Tuya Smart (válvula), 'joblib' para cargar modelos de machine learning, 'subprocess' para ejecutar comandos del sistema y 'datetime' para la gestión de fechas y horas.

El script se inicia con la definición de la función 'registrarestadovalvula', encargada de registrar las acciones sobre la válvula junto con la fecha y hora actuales en un archivo de registro. La integración con el dispositivo Tuya se logra mediante la inicialización de un objeto 'OutletDevice' de 'tinytuya', configurando parámetros como identificador, dirección, clave local y versión del protocolo, y estableciendo una conexión persistente.

La adquisición de datos es esencial y se lleva a cabo ejecutando scripts externos ('iot.py', 'tomorrow.py', 'merge.py') diseñados para generar archivos CSV con datos provenientes de las variables relevantes para las predicciones del modelo. Posteriormente, se carga un modelo de machine learning preentrenado utilizando la biblioteca 'joblib'. La última línea del archivo CSV generado ('mergeapihayiot.csv') se extrae y transforma para realizar predicciones del modelo. Dependiendo de la predicción ('ON' o 'OFF'), se ejecuta la acción correspondiente: encender o apagar la válvula. En casos de predicciones desconocidas, se manejan con mensajes informativos.

Este script se distingue en la evaluación y aplicación de modelos de machine learning en entornos prácticos, donde el control de dispositivos responde de manera precisa a las predicciones generadas por el modelo entrenado.

3.2 Resultados

3.2.1 Integración de los dos nodos sensores a la red establecida en el anterior proyecto integrador

Se llevó a cabo la instalación y codificación necesarias para incorporar los nodos a la red de sensores, con el objetivo de ampliar la cobertura de detección en el suelo y obtener datos más beneficiosos para el entrenamiento del modelo. La figura 3.6 ilustra la disposición estratégica de los nodos en el espacio.

Figura 3.6: Ubicación de los nodos sensores

3.2.2 Información sensada en HayloT

El sistema de monitoreo de temperatura experimentó mejoras significativas tras la expansión de la red de sensores. Inicialmente, con tres sensores, se observó un rendimiento estable y preciso en el seguimiento de las variaciones de temperatura. Tras la ampliación a cinco sensores, se evidenció una notable mejora en la precisión y la

capacidad para detectar cambios sutiles en el ambiente térmico.

Gracias a los nuevos nodos, en la figura 3.7 HayIoT se verifica una menor variación en la temperatura sensada ya que al ser más información el promedio resultante es más exacto

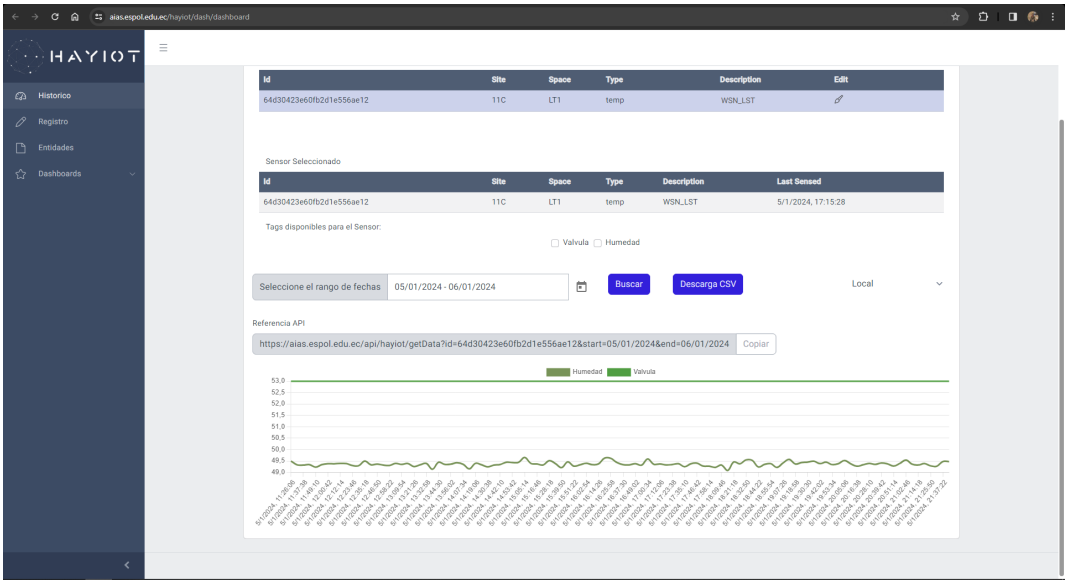


Figura 3.7: Datos nuevos y más precisos reflejados en HayIoT

3.2.3 Predicciones del Modelo y Control de la Válvula

Al realizar las predicciones del modelo (figura 3.8), se llevan a cabo acciones específicas, como el encendido o apagado de la válvula, basándose en los datos recopilados en tiempo real y evaluando las predicciones generadas por el modelo de lenguaje de máquina previamente entrenado. Esta toma de decisiones resulta en la ejecución de la acción de encender la válvula, que controla los aspersores del sistema de riego. Se realiza un muestreo promedio cada 15 minutos para actualizar continuamente el análisis de los datos y tomar decisiones informadas. El objetivo principal de este proceso es garantizar un consumo de agua adecuado y mantener el suelo con los niveles necesarios de humedad, evitando tanto el exceso como la insuficiencia de riego, ya que esto tiene un impacto crucial en el desarrollo de las plantas en la zona.

Figura 3.8: Ejecución de la predicción

3.2.4 Registro de la válvula

Una vez aplicado el modelo, se lo dejó establecido para que cada quince minutos esté ejecutando el script de las predicciones y según los resultados de las acciones tomadas por el modelo se procedió a llenar el archivo registro-valvula.txt como se ve en la figura 3.9, para poder llevar un seguimiento de cómo han estado funcionando las ejecuciones del modelo.

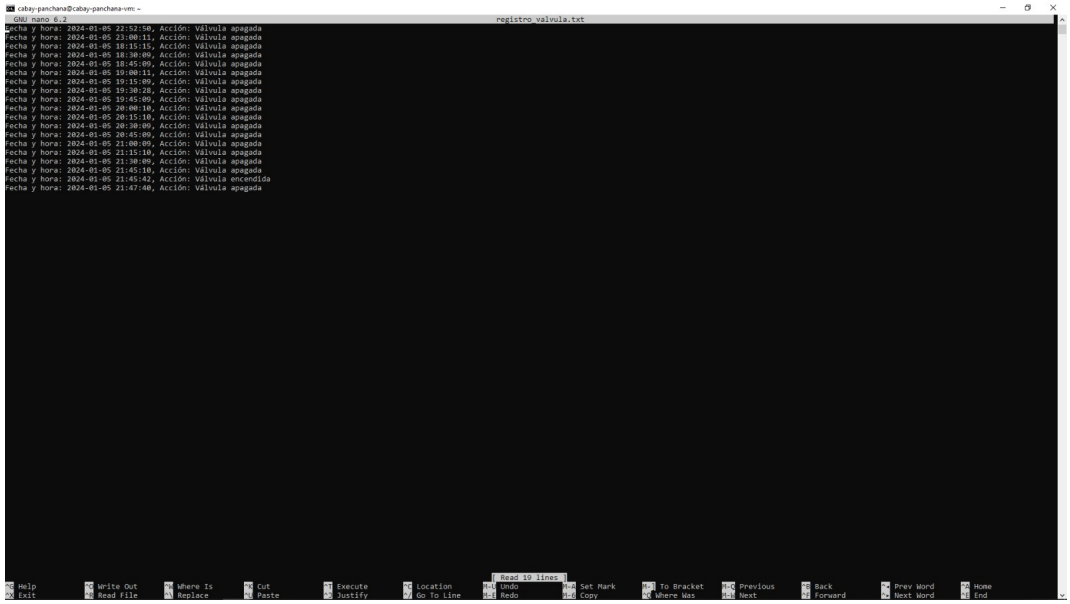


Figura 3.9: Registro del estado de la válvula en el tiempo

3.2.5 Comparativa con el modelo anterior

La red de sensores que manejaba el sistema de riego antes de que se le aplicara el algoritmo de aprendizaje de máquina, presentaba valores muy altos que indican una pérdida en el consumo de agua, ya que se sigue regando, aunque el valor de la humedad ya sea el indicado. Además, se obtuvieron periodos largos en los que no había riego alguno como se puede ver en la figura 3.10, por lo que la humedad bajaba drásticamente dañando los cultivos del lugar.

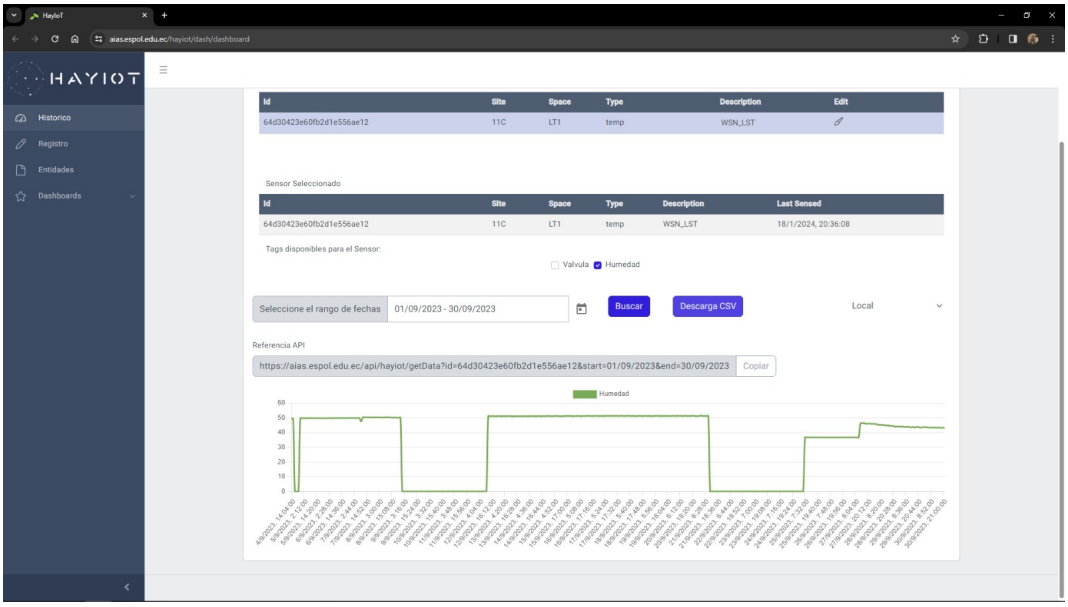


Figura 3.10: Gráfico en HayIoT antes de la aplicación del modelo

Luego de aplicar el algoritmo de Random Forest se pueden observar (figura 3.11) variaciones pequeñas e incluso de sólo decimales entre los valores de humedad, esto indica una mejor administración del sistema de riego y un ahorro significativo en el consumo del agua.

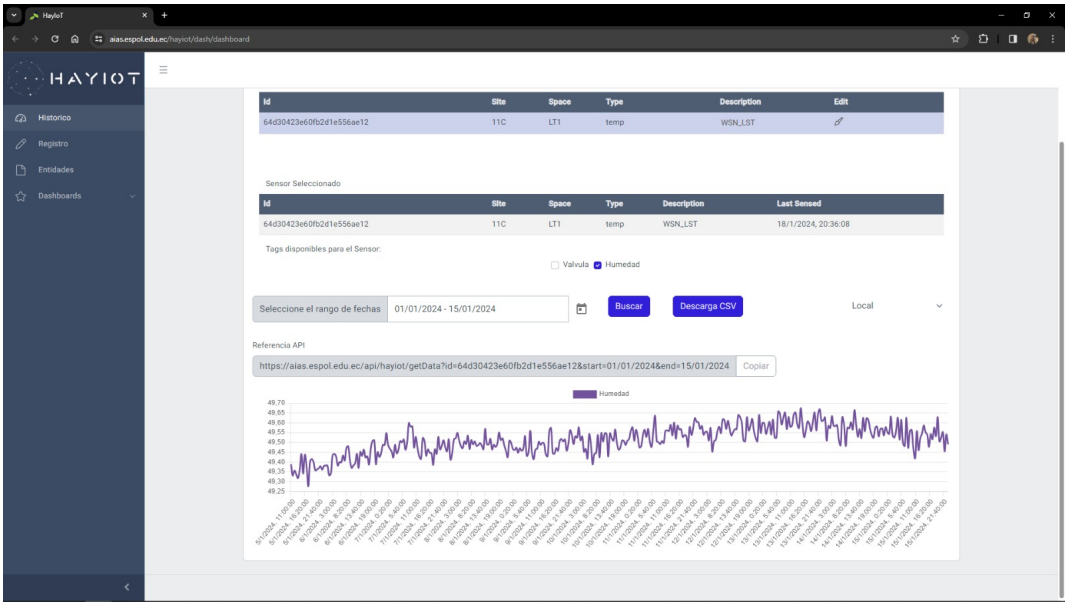


Figura 3.11: Variaciones en HayIoT luego de aplicar Random Forest

3.2.6 Estadísticas

Al comparar el modelo automatizado utilizado en el periodo anterior con el modelo inteligente implementado para el riego en el periodo actual, se ha observado una notable reducción en el tiempo en que la válvula estuvo activada, esto se puede notar en la figura 3.12. Este decremento sugiere una mejora en la eficiencia del sistema de riego, indicando que el modelo inteligente podría estar tomando decisiones más precisas y adaptadas a las necesidades reales del cultivo.

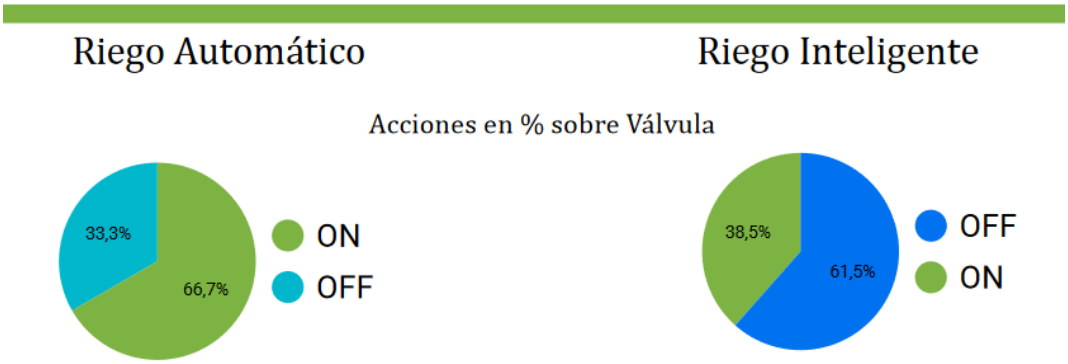


Figura 3.12: Acciones de la Válvula Inteligente Porcentual

En el contexto del sistema de riego automatizado, resultó fundamental examinar la precisión de los valores de humedad detectados y su variación a lo largo del tiempo. Esta evaluación adquirió especial importancia al observar un patrón exponencial de aumento en los valores de humedad cada vez que la válvula se activaba, contrastado con una permanencia en cero cuando la válvula estaba apagada. Esta discrepancia motivó la necesidad de optimizar el sistema.

La implementación de dos nodos adicionales al sistema demostró ser un punto de inflexión. Se notó una significativa mejora en la estabilización de los valores de humedad. En este nuevo enfoque, los valores se establecieron de manera más precisa y se mantuvieron en un rango específico, generalmente entre 40 y 55, dependiendo del proceso de humectación del suelo. Este fenómeno sugiere una mayor efectividad en el control del sistema, contribuyendo a una gestión más eficiente del riego.

La estabilización de los valores de humedad mostrada en la figura 3.13, puede atribuirse a la capacidad mejorada del modelo para interpretar y responder de manera más precisa a las condiciones reales del suelo. Este ajuste preciso genera un entorno más óptimo

para el cultivo, subrayando así el éxito del nuevo modelo inteligente en comparación con la versión automatizada previa.

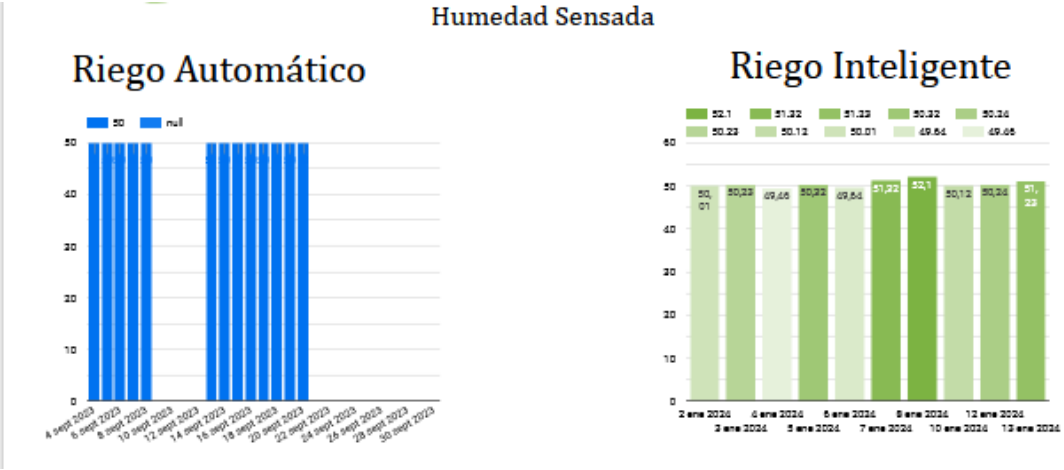


Figura 3.13: Humedad Sensada en el jardín

La comparación mostrada en la figura 3.14 entre el tiempo de riego del Modelo Inteligente y el modelo sistematizado implementado anteriormente revela una disminución significativa. Al analizar un periodo de 13 días para ambos modelos, se observa que el promedio del tiempo de riego del Modelo Inteligente se mantiene en 45.4, mientras que en el modelo sistematizado es de 120 minutos. Esta disparidad se atribuye a la capacidad del modelo inteligente para tomar decisiones más informadas.

En el modelo actual, el tiempo de riego se ajusta dinámicamente según las condiciones del entorno. Si la humedad del suelo, la temperatura, humedad y precipitación del ambiente están por debajo del umbral, la válvula se enciende. En caso contrario, permanece cerrada. Además, el tiempo de encendido de la válvula también está sujeto a estas condiciones. Este enfoque más adaptativo garantiza que el riego se realice de manera eficiente y precisa, optimizando el uso del agua.

En contraste, en el modelo anterior, la válvula se encendía diariamente durante 180 minutos, mayormente por la intervención del jardinero. Este método resultaba en un despilfarro de agua, ya que el suelo alcanzaba la humedad requerida mucho antes de ese periodo, evidenciando la falta de eficiencia del modelo sistematizado previo. La implementación del Modelo Inteligente representa, por lo tanto, una mejora sustancial en la gestión del riego, brindando beneficios tanto en términos de conservación del agua

como de optimización de recursos.

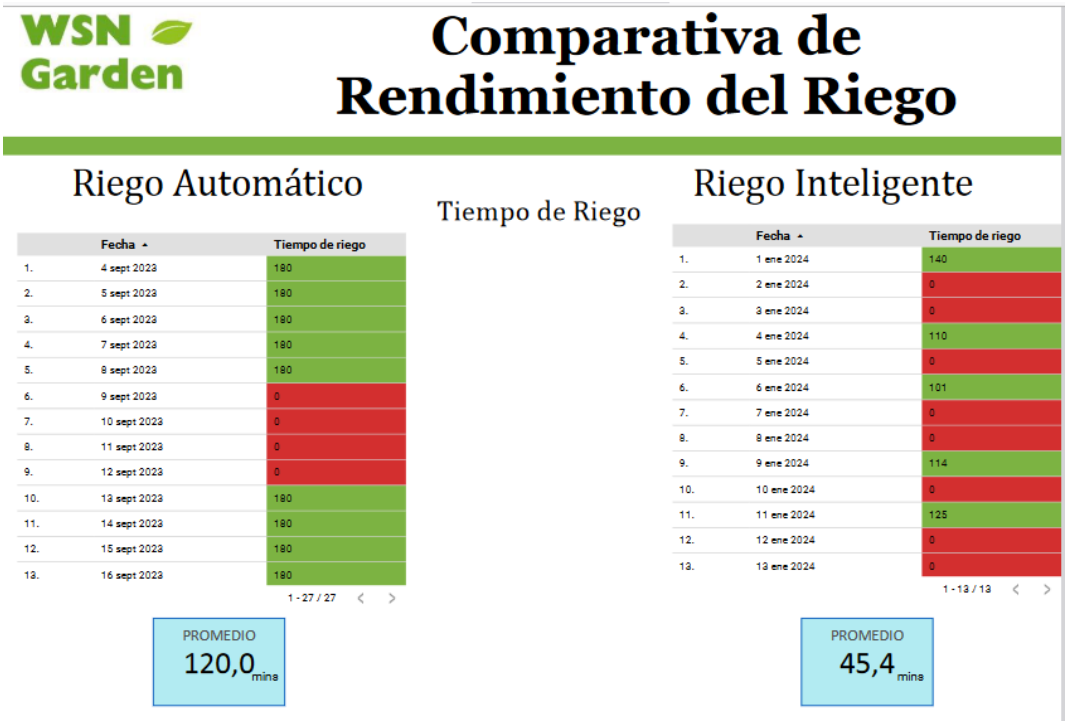


Figura 3.14: Tiempo de Riego

CAPÍTULO 4

4. CONCLUSIONES Y LÍNEAS FUTURAS

4.1 Conclusiones

El desarrollo e implementación del Sistema de Riego Inteligente ha culminado con resultados significativos y mejoras palpables en la gestión hídrica y eficiencia del riego. A continuación, se presentan las principales conclusiones derivadas de este proyecto:

1. La integración de dos nodos sensores adicionales ha marcado un avance significativo en la precisión de las mediciones de humedad del suelo. Esta mejora se traduce en una reducción notable de la variabilidad de los datos y en una estabilización efectiva de los valores registrados. La monitorización resultante ofrece una visión más fiable y detallada del estado hídrico del entorno de cultivo, fundamental para la toma de decisiones precisa en la gestión del riego.
2. La implementación del modelo de aprendizaje automático, basado en el algoritmo Random Forest, ha demostrado su eficacia en la optimización del sistema de riego. La válvula inteligente, guiada por este modelo, ha evolucionado en la toma de decisiones, logrando una mayor información y adaptabilidad. Este enfoque ha permitido evitar tanto el exceso como la insuficiencia de riego, mejorando sustancialmente la eficiencia global del sistema.
3. El sistema de riego inteligente ha experimentado una reducción drástica en el tiempo de activación de la válvula en comparación con su versión anterior. Este logro destaca una respuesta dinámica y precisa del sistema ante las condiciones cambiantes del entorno. La adaptabilidad del sistema ha sido clave para este avance, traduciéndose en un uso eficiente del agua y en un ahorro considerable de recursos hídricos.
4. La implementación de un registro detallado de las acciones ejecutadas por la válvula ha emergido como una herramienta invaluable para el análisis retrospectivo. Este

registro no solo permite evaluar el rendimiento del sistema a lo largo del tiempo, sino que también posibilita la identificación de patrones y la realización de ajustes estratégicos. La capacidad de análisis posterior se erige como un componente crucial para la mejora continua y la adaptación del sistema de riego inteligente a condiciones cambiantes.

4.2 Recomendaciones

1. Optimización del Modelo: Explorar métodos avanzados de ajuste de hiperparámetros y validación cruzada para maximizar la eficiencia del modelo Random Forest. Además, considerar la aplicación de técnicas de reducción de dimensionalidad o selección de características para mejorar la velocidad de entrenamiento y la generalización del modelo.

2. Integración de Sensores Avanzados: Evaluar la viabilidad de incorporar sensores adicionales, como temperatura y luminosidad, con el objetivo de obtener una perspectiva más completa del entorno y mejorar la precisión de las predicciones del modelo. Asimismo, se sugiere investigar la utilización de sensores más avanzados que permitan la detección temprana de condiciones ambientales adversas.

3. Seguridad y Respaldo de Datos: Implementar medidas robustas de seguridad en la máquina virtual, tales como la autenticación de usuarios y la encriptación de datos, para salvaguardar la integridad del sistema. Además, se aconseja establecer un sistema de respaldo regular con el fin de garantizar la continuidad del sistema en situaciones de fallos o pérdida de datos.

4. Escalabilidad: Investigar la escalabilidad del sistema para abordar áreas de jardín más extensas o su posible aplicación en entornos agrícolas. Esto debe hacerse con la consideración de ajustes necesarios en la infraestructura y en el modelo predictivo para garantizar un rendimiento óptimo a medida que el sistema se expande.

Análisis de Costos: Realizar un análisis exhaustivo de costos que incluya tanto los costos de implementación como los operativos a largo plazo. Es crucial evaluar la viabilidad

económica del sistema. Identificar oportunidades para optimizar costos, como explorar alternativas de hardware o evaluar servicios en la nube que puedan reducir los gastos operativos, es parte integral de la planificación financiera.

4.3 Líneas Futuras

El desarrollo y mantenimiento exitoso de un Sistema de Riego Inteligente requiere la implementación de diversas estrategias. En primer lugar, se sugiere la incorporación de actualizaciones periódicas en el modelo de aprendizaje automático, permitiendo así que el sistema se adapte continuamente a cambios en las condiciones ambientales y mejore la precisión de sus predicciones con el tiempo. Además, la evaluación de la viabilidad para agregar sensores adicionales contribuirá a obtener una cobertura más extensa del entorno agrícola, posibilitando una monitorización más precisa y adaptable. La creación de un sistema de alertas desempeñará un papel crucial al notificar a los usuarios sobre condiciones anómalas, mejorando la capacidad de respuesta del sistema y facilitando intervenciones manuales cuando sea necesario. Complementariamente, una interfaz de usuario intuitiva y la capacitación adecuada garantizarán que los agricultores puedan supervisar eficientemente el sistema, maximizando así los beneficios de esta tecnología innovadora en el ámbito agrícola. Estas acciones conjuntas no solo optimizarán el rendimiento del Sistema de Riego Inteligente, sino que también contribuirán a la sostenibilidad y eficiencia en la gestión del riego agrícola.

BIBLIOGRAFÍA

(n.d.).

Albán Bautista, J. (2022). *Desarrollo de un modelo de aprendizaje automático en la nube para mejorar la producción de una plantación de rosas* [Doctoral dissertation]. http://dspace.ups.edu.ec/handle/123456789/24255

Amine, A. (2020). Q-learning algorithm: From explanation to implementation. *Towards Data Science*. https://towardsdatascience.com/q-learning-algorithm-from-explanation-to-implementation-cdbeda2ea187

Breiman, L. (2001). *Mach. Learn.*, *45*(1), 5–32.

Cama-Pinto, A., De-La-Hoz-Franco, E., & Cama-Pinto, D. (n.d.). Las redes de sensores inalámbricos y el internet de las cosas. *Inge Cuc*. https://repositorio.cuc.edu.co/handle/11323/1546

Características principales de mobaxterm. (n.d.). https://mobaxterm.mobatek.net/features.html

Carvajal, B., & Leonel, M. (2020, December). *Design and implementation of a system of automation, monitoring and control of crops in plastic greenhouses using* [Doctoral dissertation]. Universidad de Investigación de Tecnología Experimetal Yachay.

Castro-Silva, J. A. (2016). *Sistema de riego autónomo basado en la internet de las cosas* [Doctoral dissertation]. https://reunir.unir.net/handle/123456789/3648

Docker. (n.d.). https://www.docker.com/what-docker

Etchanchú, F. B. (2021). Monitoreo de riego de cultivos y detección de anomalías con machine learning. http://sedici.unlp.edu.ar/handle/10915/140911

Freund, Y., & Schapire, R. E. (1996). Experiments with a new boosting algorithm. In L. Saitta (Ed.), *Proceedings of the thirteenth international conference on machine learning (icml 1996)* (pp. 148–156). Morgan Kaufmann. http://www.biostat.wisc.edu/~kbroman/teaching/statgen/2004/refs/freund.pdf

Hastie, T., Tibshirani, R., & Friedman, J. H. (2001). *The elements of statistical learning: Data mining, inference, and prediction*. Springer Science & Business Media.

Hortoinfo. (2023, August). El uso de invernaderos controlados por inteligencia artificial está más cerca, tras la validación del ensayo AGROS [Accessed: 2023-10-31].

Indacochea, N., & Valencia, J. (2023). *Wsn (wireless sensor network) con estandarización de datos* (tech. rep.). ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL, Guayaquil.

Ismailov, A. S., & Jo'Rayev, Z. B. (2022). Study of arduino microcontroller board. *Science and Education*, *3*(3), 172–179. https://www.researchgate.net/profile/Alisher-Ismailov/publication/359502443_

Study_of-arduino-microcontroller-board/links/62402fca8068956f3c50ea36/Study-of-arduino-microcontroller-board.pdf

Jiménez, A., Velásquez, F., & Puente, S. (2023). Centro de investigación de la universidad distrital francisco josé de caldas. *Vis. Electron.*, *17*(1).

Liakos, K., Busato, P., Moshou, D., Pearson, S., & Bochtis, D. (2018). Machine learning in agriculture: A review. *Sensors (Basel)*, *18*(8), 2674.

Llerena Izquierdo, J. (n.d.). *Codifica en python*. https://dspace.ups.edu.ec/handle/123456789/19346

Miguel, C. H. M., & Enrique, G. B. (2020). *La tecnología en el uso sostenible del agua para riego en méxico. el caso del acuífero de tecamachalco, puebla, 2017*.

Nuñez-Agurto, D., Benavides-Astudillo, E., Rodríguez, G., & Salazar, D. (2020). Propuesta de una plataforma de bajo costo basada en internet de las cosas para agricultura inteligente. *Cumbres (En línea)*, *6*(1), 53–66.

Raghunathan, V., Schurgers, C., Park, S., & Srivastava, M. B. (2002). Energy-aware wireless microsensor networks. *IEEE Signal Processing Magazine*, *19*(2), 40–50. https://doi.org/10.1109/79.985679

Rizal Isnanto, R., Eko Windarto, Y., Imago Dei Gloriawan, J., & Noerdiyan Cesara, F. (2020). Design of a robot to control agricultural soil conditions using esp-now protocol. *2020 Fifth International Conference on Informatics and Computing (ICIC)*, 1–6. https://doi.org/10.1109/ICIC50835.2020.9288575

S, A., J, A., G, P., & C, V. (2020). Machine learning applications for precision agriculture: A comprehensive review. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9311735

Vega, C., & Sebástian, C. (2020). Monitoreo y análisis de las hojas de las plantas de jitomate a partir de un sistema de visión, un algoritmo supervisado y un sistema de control semiautomático.

APÉNDICES

A Algoritmos

A.1 Algoritmo - Machine Learning

```
# Importación de bibliotecas y lectura de datos
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall
import random
import pickle
from random import sample
# ... (resto de tus importaciones y código)

# Especifica la ruta del archivo CSV
ruta_archivo = r"/home/cabay-panchana/merge.csv"

# Lee el archivo CSV
datos = pd.read_csv(ruta_archivo, delimiter=',')  # Reemplaza el delimitador por el q
print("DATOS")
print(datos)

# Crear una nueva columna 'ESTADO' basada en las condiciones
datos['ESTADO'] = datos.apply(
    lambda row: 'ON' if ((row['Humedad'] < 55) and ((row['Temperature'] < 23) or (row

# Imprimir el DataFrame con la nueva columna
print("DATOS CON SU ESTADO RESPECTIVO")
print(datos)
```

```
# Imprimir la nueva columna 'ESTADO' como lista
print("LECTURA DE ESTADOS")
estado_lista = datos['ESTADO'].tolist()
print("ESTADO:", estado_lista)
print(" ")

print("Dividir los datos en conjuntos de entrenamiento y prueba")
# División de los datos en conjunto de entrenamiento y prueba (80% - 20%)
train_data, test_data = train_test_split(datos, test_size=0.2, random_state=random.randi

print("Tamaño del conjunto de entrenamiento:", len(train_data))
print("Tamaño del conjunto de prueba:", len(test_data))
print(" ")

print("DISTRIBUCION DE ARBOLES DE MUESTREO")
print(train_data.sample(frac=2 / 3, replace=True))  # Hacer la distribución de los árbol
print(" ")

print("SE TOMAN LAS VARIABLES EXCEPTO LA DEL ESTADO QUE ES LA QUE SE VA A PREDECIR")
print(train_data.columns[:-1], "\n")  # Ignorar el estado

# Variables para crear los árboles de decisión excepto la del estado que es la predefini
print("CREACION DE LOS ARBOLES DE DECISION")
columns_set = set(train_data.columns[:-1])
columns_list = list(columns_set)
print("VARIABLES POR ARBOLES")
print(sample(columns_list, 3))  # Número de variables por árbol

print("CREACION DE BOSQUE ALEATORIO")
# Tu código para crear el bosque aleatorio
bosque = RandomForestClassifier(n_estimators=100,
                                criterion="gini",
```

```
                                    max_features="sqrt",
                                    bootstrap=True,
                                    max_samples=2 / 3,
                                    oob_score=True,
                                    class_weight='balanced',  # Puedes probar 'balanced' 
                                    min_samples_split=5,  # Ajusta según sea necesario
                                    min_samples_leaf=2)  # Ajusta según sea necesario

# Entrenamiento del bosque aleatorio
X_train = train_data.drop('ESTADO', axis=1).values
y_train = train_data['ESTADO'].values
bosque.fit(X_train, y_train)

from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, 

# Realizar predicciones en el conjunto de prueba utilizando el nuevo modelo
X_test = test_data.drop('ESTADO', axis=1)
y_pred = bosque.predict(X_test)

# Calcular la matriz de confusión en el conjunto de prueba
y_true = test_data['ESTADO']
conf_matrix = confusion_matrix(y_true, y_pred)

# Visualizar la matriz de confusión con anotaciones
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Blues", xticklabels=bosque.classe
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Confusion Matrix of the Random Forest Model")
plt.show()

# Evaluaciones adicionales del modelo
```

```
accuracy = accuracy_score(y_true, y_pred)
precision = precision_score(y_true, y_pred, pos_label='ON')  # 'ON' es la clase positiva
recall = recall_score(y_true, y_pred, pos_label='ON')
f1 = f1_score(y_true, y_pred, pos_label='ON')

print(f"Precisión: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1 Score: {f1:.4f}")

# Reporte de clasificación con precisión, recall y F1-score para ambas clases
print("\nReporte de Clasificación:")
print(classification_report(y_true, y_pred, target_names=bosque.classes_))

# Predecir probabilidades en el conjunto de prueba
probas = bosque.predict_proba(X_test)

# Verificar si el arreglo de probabilidades tiene solo una columna
if probas.shape[1] == 1:
    # Usar la columna única para el cálculo de la curva de precisión-recall
    precision, recall, _ = precision_recall_curve(y_true, probas[:, 0], pos_label='ON')
else:
    # Usar la segunda columna para el cálculo de la curva de precisión-recall
    precision, recall, _ = precision_recall_curve(y_true, probas[:, 1], pos_label='ON')

print(probas.shape)

pr_auc = auc(recall, precision)

# Área bajo la curva de precisión-recall
precision, recall, _ = precision_recall_curve(y_true, probas[:, 0], pos_label='ON')
pr_auc = auc(recall, precision)
```

```
# Set default values for fpr, tpr, and roc_auc
fpr, tpr, roc_auc = [0], [0], 0

# Curva de característica de operación del receptor (ROC) para clasificación binaria
if bosque.n_classes_ == 2:
    fpr, tpr, thresholds = roc_curve(y_true, probas[:, 1], pos_label='ON')
    roc_auc = auc(fpr, tpr)  # Calculate ROC AUC

# Plot the ROC curve
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'Curva ROC (área = {roc_auc:.2f})
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlabel('Tasa de Falsos Positivos')
plt.ylabel('Tasa de Verdaderos Positivos')
plt.title('Curva ROC')
plt.legend(loc='lower right')
plt.show()

from joblib import dump
# Guardar el modelo en un archivo
dump(bosque, 'modelo_bosque.joblib')

with open('modelo_bosque.pkl', 'wb') as file:
    pickle.dump(bosque, file)

print(bosque.predict([[35,40,71,81]]))
```

A.2 Código ESP-COORDINADORA

```
#include <ESP8266WiFi.h>
#include <espnow.h>
// Structure for receive data
typedef struct struct_message {
```

```
  char a[32];
  String id_sensor;
  String values;
} struct_message;

// Create a struct_message called myData
struct_message myData;
unsigned long epochTime;

// Variables para el cálculo del promedio
int numReadings = 0;
float totalHumidity = 0.0;


void setup() {
  // Initialize Serial Monitor

  Serial.begin(115200);
  // Set device as a Wi-Fi Station
  WiFi.mode(WIFI_STA);
  // Init ESP-NOW
  if (esp_now_init() != 0) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  // Once ESPNow is successfully Init, we will register for recv CB to
  // get recv packer info
  esp_now_set_self_role(ESP_NOW_ROLE_SLAVE);
  esp_now_register_recv_cb(OnDataRecv);

}
```

```
// Callback function that will be executed when data is received
void OnDataRecv(uint8_t *mac, uint8_t *incomingData, uint8_t len) {
  memcpy(&myData, incomingData, sizeof(myData));

  //Print received data
  //Serial.print("Sensor_ID: ");
  //Serial.println(myData.id_sensor);
  //Serial.print("Humidity: ");
  //Serial.println(myData.values);

  // Cálculo del promedio cuando se recibe un nuevo dato
  float humidityValue = myData.values.toFloat();

  totalHumidity += humidityValue;
  numReadings++;

  // Si se recibieron datos de los cinco sensores, calcular el promedio y mostrarlo
  if (numReadings == 5) {
    float averageHumidity = totalHumidity / 5.0;
    // Obtener el valor de epoch actual
    //epochTime = getTimeEpoch();
    //String mensaje = String(epochTime) + ","+ "Promedio" + "," + String(averageHumi
    //Serial.println(mensaje);
    //Serial.print("Humedad Promedio: ");
    Serial.println(averageHumidity);
    // Reiniciar los contadores y sumas para el siguiente promedio
    numReadings = 0;
    totalHumidity = 0.0;
  }

}
```

```
unsigned long getTimeEpoch() {
// Obtener el tiempo actual en segundos desde el inicio de la operación
return millis() / 1000;
}
void loop() {
// Actualizar la conexión ESP-NOW
//esp_now_loop();
// Aquí puedes agregar cualquier otra lógica que necesites en el loop
}
```

A.3 Código de Predicciones

```
import tinytuya
import joblib
import subprocess
import datetime

# Declaramos una función para guardar registros del control de la valvula
def registrar_estado_valvula(accion):
    ahora = datetime.datetime.now()
    with open('registro_valvula.txt', 'a') as archivo:
        archivo.write(f"Fecha y hora: {ahora.strftime('%Y-%m-%d %H:%M:%S')}, Acción: {ac

# Inicializar TinyTuya con los detalles del dispositivo
device = tinytuya.OutletDevice(dev_id='eba052ee2b9cb3745ai8mz',address='200.126.13.206',
data = device.status()
device.set_socketPersistent(True)

# Ejecutamos scripts para obtener datos de HayIoT y de la API
# Ejecutar iot.py para generar el primer archivo CSV
subprocess.run(["python3", "iot.py"])

# Ejecutar tomorrow.py para generar el segundo archivo CSV
```

```
subprocess.run(["python3", "tomorrow.py"])

# Ejecutar merge.py para unir los archivos CSV
subprocess.run(["python3", "merge.py"])

print('set_status() result %r' % data)
# Cargar el modelo entrenado
modelo = joblib.load('modelo_bosque.joblib')  # Reemplaza 'modelo_bosque.joblib' con

# Recuperamos los datos más recientes del archivo merge.csv

with open('merge_api_hayiot.csv', 'r') as file:
    last_line = file.readlines()[-1].strip()  # Leer la última línea y eliminar espac

# Separar los valores y convertirlos a una lista de flotantes
valores = last_line.split(',')
valores_numericos = [float(valor) for valor in valores]

# Organizar los datos en la estructura que necesitas
datos_transformados = [valores_numericos]
print(datos_transformados)



# Realizar predicciones
nuevos_datos = datos_transformados
#Datos nuevos a predecir
prediccion = modelo.predict(nuevos_datos)

#prediccion = modelo.predict([[30,29.5,61.9,0.05]])

# Controlar la válvula según la predicción
print(prediccion[0])
```

```
if prediccion[0] == 'ON':  # Verifica la clase predicha
    # Encender la válvula
    device.turn_on()
    print("Válvula encendida")
    registrar_estado_valvula("Válvula encendida")
elif prediccion[0] == 'OFF':
    # Apagar la válvula
    device.turn_off()
    print("Válvula apagada")
    registrar_estado_valvula("Válvula apagada")
else:
    # Manejo para situaciones desconocidas
    print("Predicción desconocida")
```

A.4 Código de recuperar datos HayloT

```
import requests
import csv
import json
from datetime import datetime, timedelta

# Obtener las fechas de ayer y hoy
fecha_actual = datetime.now()
fecha_anterior = fecha_actual - timedelta(days=1)  # Restar un día

fecha_formateada_actual = fecha_actual.strftime("%d/%m/%Y")
fecha_formateada_anterior = fecha_anterior.strftime("%d/%m/%Y")

print("La fecha actual es:", fecha_formateada_actual)
print("La fecha del día anterior es:", fecha_formateada_anterior)

url = "https://aias.espol.edu.ec/api/hayiot/getData"
```

```
params = {
    "id": "64d30423e60fb2d1e556ae12", #ID de la WSN
    "start": fecha_formateada_anterior,
    "end": fecha_formateada_actual
}

response = requests.get(url, params=params)

if response.status_code == 200:
    data = response.json()
    humid_data = []  # Lista para almacenar datos de humedad

    for item in data:
        if item['type'] == 'Humedad':
            humid_data.append({'Humedad': item['data']})

    # Guardar datos de humedad en un archivo CSV
    with open('datos_humedad.csv', 'w', newline='') as file:
        writer = csv.DictWriter(file, fieldnames=['Humedad'])
        writer.writeheader()
        writer.writerows(humid_data)

    # Imprime el JSON de manera legible
    # print(json.dumps(data, indent=4))
    print('Datos de HayIoT recuperados correctamente')
else:
    print("Hubo un error en la solicitud:", response.status_code)
```

A.5 Código de recuperar datos Tomorrow.io

```
import subprocess
import csv
```

```
import json

# Comando curl
curl_command = [
    "curl",
    "--compressed",
    "--request",
    "GET",
    "--url",
    "https://api.tomorrow.io/v4/timelines?location=-2.145582,-79.966756&fields=temperatu
]

try:
    # Ejecutar el comando curl desde Python y capturar la salida
    result = subprocess.run(curl_command, capture_output=True, text=True, check=True)

    # Obtener el JSON de la salida del comando
    data = json.loads(result.stdout)

    # Nombre del archivo CSV para guardar los datos
    csv_filename = "weather_data_api.csv"

    # Abrir el archivo CSV en modo escritura
    with open(csv_filename, mode='w', newline='') as file:
        writer = csv.writer(file)

        # Escribir el encabezado en el archivo CSV
        writer.writerow(["Temperature", "Humidity", "PrecipitationIntensity"])

        # Iterar sobre los intervalos y extraer los valores
        intervals = data["data"]["timelines"][0]["intervals"]
        for interval in intervals:
```

```
            values = interval["values"]

            temperature = values.get("temperature")
            humidity = values.get("humidity")
            precipitation_intensity = values.get("precipitationIntensity")

            # Escribir los valores en el archivo CSV
            writer.writerow([temperature, humidity, precipitation_intensity])

    print("Datos de tomorrowAPI recuperados correctamente")

except subprocess.CalledProcessError as e:
    # Capturar cualquier error si el comando falla
    print("Error al ejecutar el comando curl:", e)
except KeyError as key_error:
    # Manejar el error si la estructura JSON no coincide con lo esperado
    print("Error en la estructura del JSON:", key_error)
except json.JSONDecodeError as json_error:
    # Manejar el error si hay un problema al decodificar el JSON
    print("Error al decodificar el JSON:", json_error)
```

A.6 Código merge para obtener el dataset

```
import pandas as pd

# Cargar los datos de los dos archivos CSV en DataFrames
df1 = pd.read_csv('datos_humedad.csv')
df2 = pd.read_csv('weather_data_api.csv')

# Verificar tamaños de los DataFrames
if len(df1) != len(df2):
    # Determinar el DataFrame más grande y eliminar filas adicionales
```

```
    if len(df1) > len(df2):
        df1 = df1.iloc[:len(df2)]  # Eliminar filas extras de df1
    else:
        df2 = df2.iloc[:len(df1)]  # Eliminar filas extras de df2

# Renombrar las columnas para que coincidan
df2 = df2.rename(columns={'Humedad': 'Humidity'})  # Asegúrate de que la columna coincid

# Unir los DataFrames en uno solo
merged_df = pd.concat([df1, df2], axis=1)

# Guardar el DataFrame combinado en un nuevo archivo CSV
merged_df.to_csv('merge_api_hayiot.csv', index=False)
```

A.7 Comando para obtener la local key de la válvula

```
python3 -m tinytuya wizard
```

Costo del Proyecto Integrador

Nodos Sensores

Dispositivo	Precio (USD)
ESP8266 (x2)	12
Sensor de Humedad HD – 38 (x2)	40
Total	**52**

Tabla 1: Costo de Nodos Sensores.

Elemento	Precio (USD)
Cable para extensión eléctrica (20 m)	20
Caja de proyectos electrónicos (x2)	30
Tubos y codos PVC	20
Cinta aislante	6
Cable MicroUSB (x2)	14
Adaptador de corriente (x2)	20
Total	**110**

Tabla 2: Costo de Componentes de hardware.

Componentes de hardware

Mano de Obra

Cargo	Salario mensual (USD)
Ing. Telemático	850
Ing. Telemático	850
Total x 4 meses	**6800**

Tabla 3: Costo de Mano de Obra.

Costos de Procesamiento

Instancia de Máquina Virtual en la nube

Instancia	**Precio Mensual(USD)**
Máquina Virtual en la nube:	
Ubuntu 22.04 LTS	
6 Gb	
Almacenamiento 80 Gb	0.0042 x Hora (Bajo Demanda)
Total	**730 horas/mes =3.06**

Tabla 4: Costo de Instancia de Máquina Virtual.