Índice

[Índice 1](#_Toc151108948)

[Tablas 2](#_Toc151108949)

[Ilustraciones 2](#_Toc151108950)

[Esquemas 2](#_Toc151108951)

[Glosario de Términos: 2](#_Toc151108952)

[**CAPÍTULO II – “MATERIALES Y MÉTODOS”** 4](#_Toc151108953)

[2.1 Hardware 4](#_Toc151108954)

[2.1.1 Placas de desarrollo 4](#_Toc151108955)

[2.1.2 Raspberry Pi 3B 7](#_Toc151108956)

[2.1.2 ESP-32 9](#_Toc151108957)

[2.1.3 Sensores espectroscópicos 10](#_Toc151108958)

[2.1.4 Sensor AS7265X 11](#_Toc151108959)

[2.1.4 Protocolo I2C 12](#_Toc151108960)

[2.2 Software 14](#_Toc151108961)

[2.2.1 Frontend 14](#_Toc151108962)

[2.2.2 Python 15](#_Toc151108963)

[2.2.3 TensorFlow 16](#_Toc151108964)

[2.2.5 Base Datos 16](#_Toc151108965)

[2.2.6 Django 16](#_Toc151108966)

[2.2.7 Lenguaje C 16](#_Toc151108967)

[2.3 Detección Multivariable 16](#_Toc151108968)

[2.3.1 Métodos Detección Multivariable 16](#_Toc151108969)

[2.3.2 Introducción a Redes Neuronales 16](#_Toc151108970)

[- Neurona Bilógica 16](#_Toc151108971)

[- Neurona de McCulloch-Pitts 16](#_Toc151108972)

[- Neurona Artificial 16](#_Toc151108973)

[2.3.3 Redes Neuronales 16](#_Toc151108974)

[2.3.4 Entrenamiento de Redes Neuronales 16](#_Toc151108975)

[2.4 Conclusiones del capitulo 16](#_Toc151108976)

[Referencias 17](#_Toc151108977)

[Anexos 19](#_Toc151108978)

# Tablas

[Tabla 1: Parámetros Raspberry Pi3B 7](#_Toc151024743)

[Tabla 2: Parámetros ESP32 9](#_Toc151024744)

[Tabla 3: Funciones de activación 21](#_Toc151024745)

# Ilustraciones

[Ilustración 1: Arduino Mega 2560 Rev3 5](#_Toc151108979)

[Ilustración 2: Raspberry Pi 3 Model B 6](#_Toc151108980)

[Ilustración 3: Esp-WROMM-32 6](#_Toc151108981)

[Ilustración 4: Adafruit Feather RP2040 6](#_Toc151108982)

[Ilustración 5: BeagleBone Black Rev C 7](#_Toc151108983)

[Ilustración 6: Raspberry Pi 3 Modelo B y sus elementos fundamentales. 8](#_Toc151108984)

[Ilustración 7: ESP32 y sus pines GPIO 10](#_Toc151108985)

[Ilustración 8: Sensor de espectroscopía SparkFun Triad -AS7265x 12](#_Toc151108986)

[Ilustración 9: Protocolo I2C 14](#_Toc151108987)

[Ilustración 10: Logotipo Python 15](#_Toc151108988)

[Ilustración 11: Neurona Biológica 16](#_Toc151108989)

[Ilustración 12: Neurona McCulloch- Pitts 16](#_Toc151108990)

[Ilustración 13: Neurona Artificial 16](#_Toc151108991)

[Ilustración 14: Red Neuronal Multicapa 16](#_Toc151108992)

# Esquemas

[Esquema 1: Arquitectura MWC de Django 15](#_Toc151024766)

[Esquema 2: Arquitectura basada en URL de Django 15](#_Toc151024767)

# Gráficos

**No se encontraron entradas de tabla de contenido.**

Glosario de Términos:

* USB: (Universal Serial Bus). Protocolo universal de comunicación en serie entre dispositivos informáticos, usado comúnmente para conectar memorias flash, cámaras, impresoras, etc.
* UART: (Universal Asynchronous Receiver/Transmitter). Protocolo de comunicación en serie asíncrona entre dispositivos. Permite la transmisión de datos bit a bit.
* HDMI: (High-Definition Multimedia Interface). Interfaz multimedia de alta definición que permite la transmisión digital de video, audio y otros datos.
* GPIO (General Purpose Input/Output): Entrada/Salida de Propósito General. Se refiere a los pines de un microcontrolador o dispositivo electrónico que se pueden configurar tanto como entradas como salidas para interactuar con otros componentes del sistema.
* DBMS (Database Management System): Sistema de Gestión de Base de Datos. Es un software que permite crear, organizar y administrar bases de datos.
* I2C (Inter-Integrated Circuit): Circuito Interintegrado. Es un bus de comunicación de datos serial que permite la transferencia de información entre dispositivos electrónicos utilizando solo dos cables.
* API (Application Programming Interface): Interfaz de Programación de Aplicaciones. Es un conjunto de reglas y protocolos que permiten la comunicación entre diferentes componentes de software
* SPI (Serial Peripheral Interface): Interfaz Periférica en Serie. Es un bus de comunicación síncrono utilizado para la transferencia de datos entre microcontroladores y periféricos, como sensores, pantallas y memorias.
* SVM (Support Vector Machine): Máquina de Vectores de Soporte. Es un algoritmo de aprendizaje automático supervisado utilizado para la clasificación y regresión.
* ORM (Object-Relational Mapping): Mapeo Objeto-Relacional. Es una técnica de programación que permite relacionar objetos en un lenguaje de programación orientado a objetos con tablas en una base de datos relacional.
* URL (Uniform Resource Locator): Localizador Uniforme de Recursos. Es la dirección única que identifica la ubicación de un recurso en la web, como una página web, una imagen o un archivo
* RELU (Rectified Linear Unit): Unidad Lineal Rectificada. Es una función de activación utilizada en redes neuronales artificiales.
* DRY (Don't Repeat Yourself): No te repitas. Es un principio de programación que promueve la reutilización de código y la eliminación de duplicación innecesaria.
* MVC (Model-View-Controller): Modelo-Vista-Controlador. Es un patrón de diseño de software que separa los componentes de una aplicación.
* SSH: (Secure Shell). Protocolo de red que permite establecer sesiones remotas seguras en equipos de forma transparente.
* VNC: (Virtual Network Computing). Protocolo que permite el control remoto gráfico de computadoras a través de una red. Útil para asistencia remota.
* IOT: (Internet of Things). Internet de las cosas, paradigm donde dispositivos físicos se conectan a internet y pueden interactuar entre sí y con los usuarios.
* OTA: (Over-The-Air). Actualización de software mediante transmisión inalámbrica vía red celular u otros métodos.
* BLE: (Bluetooth Low Energy). Estándar Bluetooth de bajo consumo usado en dispositivos conectados.
* HTML: (HyperText Markup Language). Lenguaje de marcado para crear páginas web.
* CSS: (Cascading Style Sheets). Lenguaje usado para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado como HTML. Permite separar la estructura del contenido de su representación o estilos visuales.
* RNA: (Redes Neuronales Artificiales). Modelos computacionales inspirados en el funcionamiento de las redes neuronales biológicas, capaces de aprender tareas complejas a partir de grandes conjuntos de datos.

# **CAPÍTULO II – “MATERIALES Y MÉTODOS”**

## 2.1 Hardware

El hardware es la parte física y tangible de los sistemas digitales, encargada de procesar y almacenar la información de manera automática. Está conformado por los diversos componentes electrónicos como microprocesadores, microcontroladores, memorias, circuitos integrados, tarjetas de expansión, pantallas, sensores y actuadores, entre otros elementos.

El hardware constituye la base material sobre la cual se ejecutan las instrucciones lógicas definidas en el software. Los circuitos electrónicos son los que efectivamente realizan operaciones aritméticas, de comparación, almacenamiento temporal y permanente de datos, todo ello gracias a los transistores que los componen. De no ser por la cualidad física y eléctrica de estos elementos construidos con silicio y otros materiales, no sería posible procesar la información de manera automática.

La correcta selección de cada elemento de hardware, su diseño y ensamblado resultan determinantes para lograr sistemas robustos, escalables, de alto rendimiento y fiables; pudiendo superar límites y abrir nuevas posibilidades gracias a la tecnología. Un sólido desarrollo de hardware es pilar para la concreción exitosa de proyectos electrónicos y digitales.

### 2.1.1 Placas de desarrollo

El abaratamiento de los costos de producción y la búsqueda de dispositivos con mayores capacidades de procesamiento y cómputo, fue generando procesadores más potentes y más pequeños. Esta miniaturización logro evolucionar y hacer más baratos los dispositivos de procesamiento, logrando que hoy en día una persona tenga complejas microcomputadoras capaces de realizar procesamiento de alto rendimiento, permitiendo que prácticamente cualquier estudiante, o individuo, interesado en desarrollar sistemas ingenieriles tenga acceso casi inmediato a diferentes tarjetas de desarrollo y sistemas embebidos, las cuales pueden ser utilizadas en prácticamente cualquier idea que el usuario quiera realizar[1]

Una placa de desarrollo es una tarjeta de circuito impreso diseñada para facilitar el desarrollo de prototipos electrónicos, donde se montan y conectan diferentes componentes electrónicos como microcontroladores, memorias, sensores, conectores y otros elementos, permitiendo probar y depurar software y hardware de forma rápida.

Más allá de los componentes internos, todas las placas de desarrollo comparten ciertas funcionalidades básicas como comunicación por USB (Universal Serial Bus) o UART(Universal Asynchronous Receiver/Transmitter) para programación, E/S digitales y analógicas para interacción con el mundo físico, memoria embebida, espacio para ensamblaje para añadir módulos externos y tamaño pequeño con posibilidades portabilidad. Estas características genéricas permiten el desarrollo flexible de todo tipo de prototipos electrónicos y sistemas embebidos.

La comunidad de desarrolladores de estas placas ha crecido considerablemente, lo que significa que existe una amplia gama de recurso y documentaciones para su desarrollo.

Al emplear este tipo de dispositivos, se puede reducir significativamente el costo de los equipos, lo que los hace más accesibles para una variedad de aplicaciones y entornos. Esto puede ser especialmente beneficioso en países en desarrollo, pequeñas empresas o laboratorios con presupuestos limitados, donde la adquisición de equipos costosos puede ser un obstáculo.

Algunas placas de desarrollo:

* Arduino
* Raspberry Pi
* ESP8266/ESP32
* BeagleBone



Ilustración 1: Arduino Mega 2560 Rev3



Ilustración 2: Raspberry Pi 3 Model B



Ilustración 3: Esp-WROMM-32



Ilustración 4: Adafruit Feather RP2040



Ilustración 5: BeagleBone Black Rev C

### 2.1.2 Raspberry Pi 3B

Uno de los dispositivos más utilizados dentro del área de los computadores de placa única es la Raspberry Pi, más específicamente si versión 3 modelo B, versión que presenta mejores prestaciones en cuanto a velocidad, puertos y potencia. Esta placa de computadora de bajo costo que permite la implantación de todo un sistema operativo, permitiendo utilizarla como un ordenador habitual, pero con menos prestaciones. Puede ser utilizada directamente desde una pantalla HDMI (High-Definition Multimedia Interface), hasta de forma remota mediante SSH (Secure Shell) o VNC (Virtual Network Computing). La versatilidad de esta placa permite desarrollar grandes proyectos

Cuenta con un potente procesador de cuatro núcleos y 1.2 GHz, lo que le permite realizar tareas computacionales de manera eficiente. Esto es fundamental para el procesamiento y análisis de datos. Viene con una amplia variedad de puertos de Entrada/Salida, incluyendo puertos USB, HDMI, Ethernet y GPIO (General Purpose Input/Output). Estos puertos permiten la conexión de diferentes sensores, actuadores y otros componentes. La versatilidad de la Raspberry Pi 3B hace que sea fácil de integrar con una amplia gama de dispositivos y periféricos externos.[2]

Otra característica destacada es que cuenta con Wi-Fi integrado y soporte para Bluetooth, lo que facilita la transferencia de datos y la comunicación con otros dispositivos.

Tiene un amplio soporte de software y una comunidad de desarrolladores activa. Existen numerosos sistemas operativos y entornos de desarrollo disponibles, como Raspbian (basado en Linux), que brindan herramientas y recursos para programar y personalizar la funcionalidad del dispositivo según las necesidades específicas de cada proyecto. La comunidad de usuarios y desarrolladores ofrece foros, tutoriales y proyectos compartidos que pueden servir como referencia y apoyo en la implementación del detector.

Algunos parámetros a resaltar de la Raspberry Pi3B:

| Parámetro | Valor |
| --- | --- |
| CPU | Broadcom BCM2837B0, Cortex-A53 de 64 bits, 1.2 GHz |
| Memoria RAM | 1 GB LPDDR2 SDRAM |
| Almacenamiento | MicroSD (no incluido) |
| Conectividad | Wi-Fi 802.11n, Bluetooth 4.2, Ethernet 10/100 Mbps |
| Puertos USB | 4 puertos USB 2.0 |
| Puertos GPIO | 40 pines GPIO |
| Salida de video | HDMI, salida de video compuesto |
| Audio | Salida de audio estéreo 3.5 mm, HDMI |
| Cámara | Conector de cámara CSI |
| Pantalla | Conector de pantalla DSI |
| Alimentación | Conector micro USB, 5V/2.5A |
| Dimensiones | 85 x 56 x 17 mm |

Tabla 1: Parámetros Raspberry Pi3B

**

Ilustración 6: Raspberry Pi 3 Modelo B y sus elementos fundamentales.

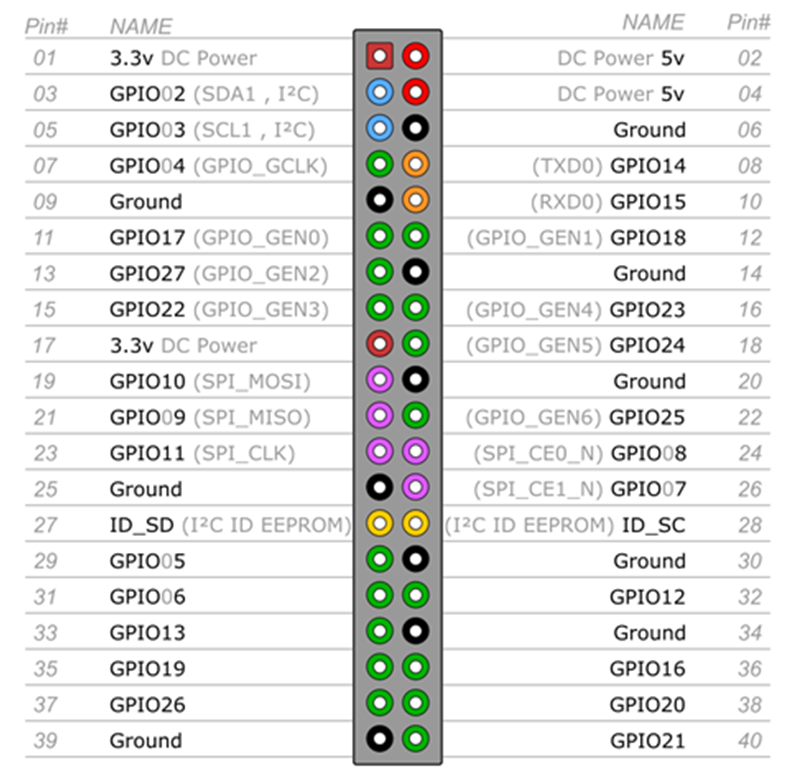


Ilustración 7: Raspberry 3B PINOUT

### 2.1.2 ESP-32

El ESP32 es un dispositivo altamente avanzado, asequible y fácil de utilizar que cuenta con una amplia gama de pines físicos que facilitan la comunicación con el entorno externo. Además, viene equipado con módulos de Wi-Fi y Bluetooth, lo que le permite establecer conexiones inalámbricas de manera sencilla, lo que lo convierte en una excelente opción para la programación OTA (Over-The-Air) y aplicaciones IOT (Internet of Things). Su capacidad para actualizaciones de firmware a través de la red y su capacidad de conectarse a una amplia gama de dispositivos y sensores lo hace extremadamente versátil y adaptable a diferentes escenarios de desarrollo.

A continuación, se presentan algunas de las aplicaciones más comunes del ESP32[3], [4]:

1. **Automatización industrial**: El ESP32 se puede utilizar en aplicaciones industriales para la automatización, supervisión y control de procesos. Gracias a sus capacidades de conectividad, como WiFi, Bluetooth LE y protocolos Ethernet industriales como Modbus TCP, el ESP32 puede integrarse fácilmente en sistemas de control industrial.
2. **Energías renovables**: El ESP32 se puede utilizar en aplicaciones relacionadas con energías renovables, como la energía eólica. Permite monitorear y controlar el rendimiento de las instalaciones, lo que ayuda a optimizar su eficiencia.
3. **Invernaderos**: El ESP32 es ideal para controlar y monitorear las condiciones ambientales en invernaderos. Puede utilizarse para controlar la humedad, la ventilación, el nivel de CO2 y la luminosidad, lo que ayuda a mantener un ambiente óptimo para el crecimiento de las plantas.
4. **Mantenimiento de maquinaria**: En la industria, es importante monitorear y controlar elementos críticos en las instalaciones. El ESP32 puede utilizarse para obtener información en tiempo real sobre el estado de la maquinaria y facilitar el mantenimiento preventivo.
5. **Industria 4.0**: El ESP32 es compatible con la tendencia de la Industria 4.0, que busca la automatización y la interconexión de los sistemas de producción. Con sus capacidades de conectividad y su capacidad para gestionar grandes cantidades de datos, el ESP32 puede ser utilizado en aplicaciones de automatización industrial avanzadas.
6. **Logística**: En el campo de la logística, el ESP32 puede utilizarse para mejorar la automatización y el control de los procesos de movimiento de mercancías. Permite supervisar y verificar de manera segura el flujo de mercancías, lo que ayuda a aumentar la eficiencia y reducir errores.
7. **Industria petrolera y química**: El ESP32 puede utilizarse en la industria petrolera y química para monitorear y controlar elementos críticos en las instalaciones. Su capacidad de conectividad y su compatibilidad con protocolos Ethernet industriales como Modbus TCP lo hacen adecuado para estas aplicaciones.

Algunos parámetros a destacar en el esp32[5]:

| **Parámetro** | **Valor** |
| --- | --- |
| CPU | Dual core, hasta 240 MHz |
| Memoria Flash | 4 MB |
| Memoria RAM | 520 KB |
| Wi-Fi | 802.11 b/g/n |
| Bluetooth | Bluetooth 4.2 BLE |
| GPIO | 25 pines GPIO |
| ADC | 18 canales ADC |
| DAC | 2 canales DAC |
| Interfaces de comunicación | UART, SPI, I2C, I2S |
| PWM | 16 canales PWM |
| RTC GPIO | 16 pines RTC GPIO |
| Consumo de energía | Corriente de reposo inferior a 5 μA |
| Sistema operativo | FreeRTOS con LwIP |

Tabla 2: Parámetros ESP32

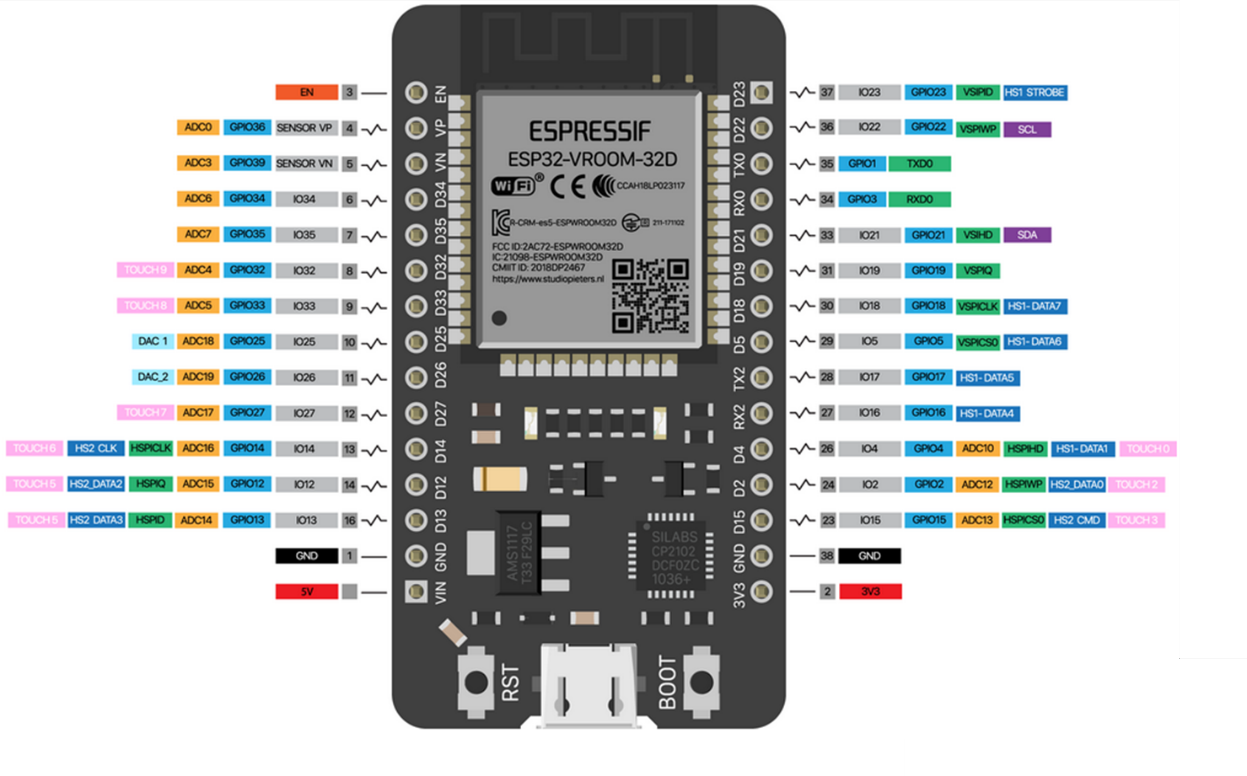


Ilustración 8: ESP32 y sus pines GPIO

### 2.1.3 Sensores espectroscópicos

Cuando se considera el panorama de los sensores espectroscópicos, existen múltiples opciones disponibles en el mercado. A continuación, se presentan alguna de ellas:

* **Sensor AS7265x:** Es fabricado por ams AG, ofrece una amplia gama de aplicaciones en el análisis espectral. Su rango de longitud de onda cubre aproximadamente desde 410 nm (visible) hasta 940 nm (infrarrojo cercano). En términos de precio, el sensor AS7265x suele tener un costo que varía entre los 50 y 150 dólares estadounidenses, dependiendo del proveedor y las características adicionales que pueda ofrecer.
* **Sensor Hamamatsu S11639:** Este sensor de alta calidad y resolución espectral está diseñado para aplicaciones científicas y de investigación que requieren un análisis detallado del espectro. Si bien los rangos de longitud de onda específicos pueden variar según el modelo exacto, en general se puede esperar que cubra el rango 200 nm hasta 1000 nm. En términos de precio, el sensor Hamamatsu S11639 suele tener un costo más elevado debido a sus características avanzadas y precisión, estimándose en un rango de precios entre los 200 y 500 dólares estadounidenses, o incluso más, dependiendo del modelo y las especificaciones precisas.

El sensor AS7265x destaca como la mejor opción debido a las siguientes razones. En primer lugar, ofrece un amplio rango de longitud de onda que abarca desde aproximadamente 410 nm hasta 940 nm, lo que permite realizar mediciones precisas en el espectro visible e infrarrojo cercano. Esta versatilidad es especialmente útil en aplicaciones que requieren un análisis detallado y completo de la luz.

En segundo lugar, la relación calidad-costo es muy buena, debido a su buena resolución espectral y su capacidad para proporcionar mediciones precisas y simultáneas en 18 canales espectrales diferentes. Esto permite un análisis más detallado y una mejor caracterización de la luz y sus componentes. Ofrece una integración sencilla y es compatible con una variedad de plataformas y microcontroladores, lo que facilita su implementación en diferentes sistemas y aplicaciones.

### 2.1.4 Sensor AS7265X

La familia AS7265x incorpora tres chips para ofrecer un conjunto de sensores multiespectrales de 18 canales que cubren las longitudes de ondas de 410nm a 940nm. El AS72651 cubre las longitudes de onda de 600nm a 870nm y sirve como controlador maestro del conjunto. Cuando se combina con el AS72652 (respuesta espectral de 560nm a 940nm) y el AS72653 (respuesta espectral de 410nm a 535nm) el conjunto entero entrega 18 canales con filtro gaussiano con un ancho de banda de media anchura (FWHM) de 20nm.

El funcionamiento del conjunto de chips AS7265x requiere del uso de un firmware que debe cargarse en una memoria flash a través de una interfaz UART. Los sensores AS72651, AS72652 Y AS72653 están precalibrados con una fuente de luz específica.

La comunicación con el sensor AS7265X se realiza a través de los protocolos I2C (Inter-Integrated Circuit) o UART. Estos protocolos permiten la transferencia de datos entre las placas de desarrollo y el sensor AS7265X.

Cada dispositivo tiene dos controladores LED integrados con corriente programable y puede ser temporizado para aplicaciones de obturación electrónica. La familia de dispositivos integra filtros gaussianos que también proporcionan aperturas incorporadas para controlar la luz que entra en el conjunto de sensores[6].

| **Característica** | **AS72651** | **AS72652** | **AS72653** |
| --- | --- | --- | --- |
| Longitudes de onda | 410 nm, 435 nm, 460 nm, | 450 nm, 500 nm, 550 nm, | 610 nm, 680 nm, 730 nm, |
|  | 485 nm, 510 nm, 535 nm, | 570 nm, 600 nm, 650 nm, | 760 nm, 810 nm, 860 nm, |
|  | 560 nm, 585 nm, 610 nm | 700 nm, 730 nm, 760 nm, | 900 nm, 940 nm, 980 nm, |
|  |  | 810 nm, 860 nm, 900 nm, |  |
| Resolución espectral | 1.2 nm | 10 nm | 20 nm |
| Interfaz de comunicación | I2C | I2C | I2C |
| Rango dinámico | 6.3 nW/cm²/nm a 2.5 µW/cm²/nm | 0.5 µW/cm²/nm a 10 µW/cm²/nm | 2.5 µW/cm²/nm a 50 µW/cm²/nm |
| Temperatura de operación | -40°C a 85°C | -40°C a 85°C | -40°C a 85°C |
| Tensión de alimentación | 3.3 V | 3.3 V | 3.3 V |
| Dimensiones | 18 mm x 18 mm | 18 mm x 18 mm | 18 mm x 18 mm |

Tabla 3: Características AS7265x



Ilustración 9: Sensor de espectroscopía SparkFun Triad -AS7265x

### 2.1.5 Protocolo I2C

El protocolo I2C (Inter-Integrated Circuit) es un protocolo de comunicación serial utilizado para conectar y controlar dispositivos electrónicos en un sistema. Fue desarrollado por Philips Semiconductor (ahora NXP Semiconductors) en la década de 1980 y se ha convertido en un estándar de la industria.

El funcionamiento del protocolo I2C se basa en una comunicación maestro-esclavo, donde un dispositivo maestro controla la comunicación y uno o varios dispositivos esclavos responden a las solicitudes del maestro

Líneas de comunicación: El bus I2C utiliza dos líneas para la comunicación:

* **SDA** (Serial Data Line): Esta línea se utiliza para la transmisión de datos entre el maestro y los esclavos.
* **SCL** (Serial Clock Line): Esta línea se utiliza para sincronizar la comunicación entre el maestro y los esclavos.

rama de comunicación: El protocolo I2C utiliza una trama de bits para enviar datos entre el maestro y los esclavos. Algunos de los bits importantes en la trama son:

* **Inicio** (Start): Indica el comienzo de una comunicación.
* **Parada** (Stop): Indica el final de una comunicación.
* **Confirmación** (ACK): Indica que se ha recibido correctamente un dato.
* **No Confirmación** (NACK): Indica que no se ha recibido correctamente un dato.
* **Lectura**/**Escritura** (L/W): Indica si se va a leer o escribir datos.
* **Bits de dirección**: Se utilizan para identificar el dispositivo esclavo al que se desea comunicar.
* **Bits de datos**: Contienen la información que se desea transmitir.

I2C permite la transferencia de datos en ambas direcciones entre un dispositivo maestro y uno o varios dispositivos esclavos. Cada dispositivo esclavo en el bus I2C tiene una dirección única que le permite ser identificado por el dispositivo maestro. Admite diferentes velocidades de transferencia, como el modo estándar (100 kbps), el modo rápido (400 kbps) y el modo rápido plus (hasta 1 Mbps)[7].

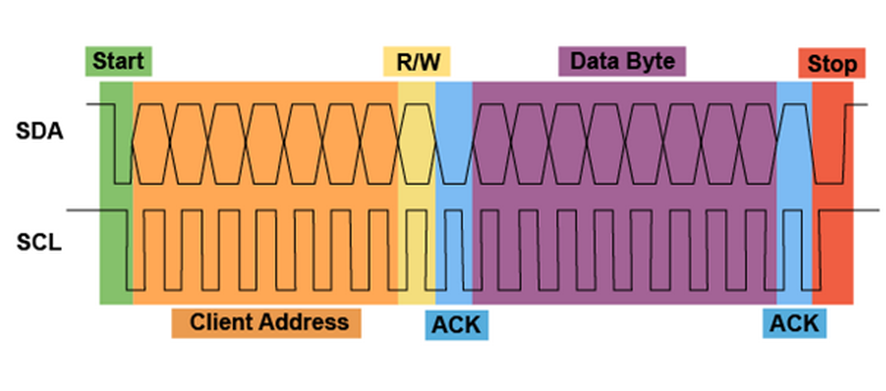


Ilustración 10: Protocolo I2C

## 2.2 Software

Para la integración correcta y fluida entre los componentes físicos de un sistema y su parte lógica y software, es fundamental contar con un conjunto robusto de instrucciones y protocolos de comunicación. A través de estos mecanismos es posible el intercambio sistemático de datos e instrucciones entre el hardware tangible y las capas superiores de interfaz con el usuario.

Los programas, lenguajes y marcos de programación cumplen precisamente con el objetivo de lograr esta simbiosis entre lo físico y lo lógico. En la actualidad existen numerosas alternativas según el tipo de sistema, pero todos comparten la premisa de definir especificaciones claras sobre cómo debe producirse el diálogo digital entre los distintos niveles de abstracción.

A nivel de hardware, factores como la arquitectura de buses de datos, protocolos seriales y paralelos, y definición de registros de control, resultan cruciales para que los datos fluyan de forma ordenada. Asimismo, es necesario normalizar lenguajes de máquina y códigos de operación compatibles con cada microprocesador o microcontrolador.

Por otro lado, el software y firmware aportan la capa de abstracción e interfaces que permiten gestionar los eventos en forma entendible para el programador. Lenguajes de alto nivel, drivers, sistemas operativos embebidos, librerías y las API (Application Programming Interface) cumplen esta labor mediante funciones, clases y métodos que se mapean luego a las instrucciones a nivel de hardware.

### 2.2.1 Frontend

Para el desarrollo efectivo de una interfaz web que permita la interacción entre usuarios y sistemas físicos, es fundamental contar con una capa intermediaria que se encargue de la comunicación de forma transparente y sencilla para el usuario final.

**HTML, CSS y JavaScript**

HTML (HyperText Markup Language), CSS (Cascading Style Sheets) y JavaScript son tres lenguajes para desarrollar páginas web que tienden a desarrollarse juntos, sus funcionalidades los hacen dependientes unos de los otros. Esta forma de programación es una de más básicas en cuanto al diseño web, y es necesaria para cualquier interfaz visual web.

El HTML es el código que se utiliza para estructurar y desplegar una página web y sus contenidos. Sus contenidos podrían ser párrafos, una lista con viñetas, o imágenes y tablas de datos. Es el cuerpo de una pagina web, dotando de bloques, texto imágenes y colores a la interfaz. CSS es el lenguaje que define los estilos de una interfaz web, indica los colores, formas, bordes y posicionamiento de los elementos, su uso dota a una página de elementos responsables, o sea que se adapten a los distintos entornos y pantallas. JavaScript es un lenguaje de programación del lado del cliente dotando de dinamismos y secuencias lógicas a una interfaz web , permite adjuntar información de eventos , comunicación con servidores y lógicas para ocultar , mostrar o modificar información proveniente de la parte visual de HTML [8]–[10].

**Bootstrap**

Bootstrap es un marco de desarrollo web gratuito y de código abierto. Está diseñado para facilitar el proceso de desarrollo de los sitios web responsivos y orientados a los dispositivos móviles, proporcionando una colección de sintaxis para diseños de plantillas[11].

Este permite crear interfaces web con CSS y JavaScript, cuya particularidad es la de adaptar la interfaz del sitio web al tamaño del dispositivo en que se visualice. Viene integrado como librerías para creación de elementos predefinidos y dinámicos, así como eventos para integrar avisos con los servidores

**AdminLTE**

AdminLTE es una plantilla de diseño de interfaz de usuario basada en Bootstrap para aplicaciones web de administración. Está diseñada para ser fácil de usar y personalizar, y se utiliza a menudo en proyectos de desarrollo de software para proporcionar una apariencia atractiva y consistente para las interfaces de administración; es una plantilla de código abierto.

Incluye una gran cantidad de elementos de diseño, como menús de navegación, barras laterales, formularios, tablas, botones, iconos y mucho más. También incluye un conjunto de widgets y páginas predefinidos, como páginas de inicio, páginas de inicio de sesión y páginas de error, que se pueden personalizar fácilmente para adaptarse a las necesidades del proyecto

### 2.2.2 Python

Python es un lenguaje de programación ampliamente utilizado en las aplicaciones web, el desarrollo de software, la ciencia de datos y el machine learning. Los desarrolladores utilizan Python porque es eficiente y fácil de aprender y tiene una robusta comunidad, además de que se puede ejecutar en muchas plataformas diferentes. El software Python se puede descargar gratis, se integra bien a todos los tipos de sistemas y aumenta la velocidad del desarrollo. Además, es un lenguaje interpretado, lo que significa que ejecuta directamente el código línea por línea. Si existen errores en el código del programa, su ejecución se detiene. Así, los programadores pueden encontrar errores en el código con rapidez. [12].

Este lenguaje de programación potente y fácil de aprender tiene estructuras de datos de alto nivel eficientes y un simple pero efectivo sistema de programación orientado a objetos. La elegante sintaxis de Python y su tipado dinámico, junto a su naturaleza interpretada lo convierten en un lenguaje ideal para scripting y desarrollo rápido de aplicaciones en muchas áreas, para la mayoría de plataformas[13].

Python posee una de las mayores comunidades de desarrolladores, por lo tanto, programar en este lenguaje tendrá siempre un gran respaldo de documentación, reduciendo el tiempo de arreglo de errores, implementación de códigos y creación de librerías La creación de grandes proyectos puede requerir de mucho menos tiempo en comparación con otros lenguajes.



Ilustración 11: Logotipo Python

### 2.2.3 TensorFlow

TensorFlow es una biblioteca de software de código abierto que se utiliza para el aprendizaje automático y la inteligencia artificial. Permite a los desarrolladores crear y entrenar modelos de aprendizaje automático, especialmente redes neuronales profundas. Algunas de las cosas que se pueden hacer con TensorFlow son[14]:

1. **Construir y entrenar modelos de aprendizaje automático**: TensorFlow proporciona herramientas y funciones para construir y entrenar modelos de aprendizaje automático. Esto incluye la definición de la arquitectura del modelo, la selección de funciones de pérdida y optimizadores, y el ajuste de los hiperparámetros del modelo.
2. **Realizar inferencia y predicciones**: Una vez que se ha entrenado un modelo con TensorFlow, se puede utilizar para realizar inferencia y hacer predicciones sobre nuevos datos. Esto es útil en aplicaciones como el reconocimiento de imágenes, el procesamiento del lenguaje natural, la detección de objetos, entre otros.
3. **Procesamiento distribuido**: TensorFlow permite distribuir el procesamiento de los modelos de aprendizaje automático en múltiples dispositivos o máquinas. Esto permite acelerar el entrenamiento y la inferencia de los modelos, especialmente en conjuntos de datos grandes.

**TensorFlow Lite**:

TensorFlow Lite para C es una versión de TensorFlow Lite diseñada específicamente para ser utilizada con el lenguaje de programación C. TensorFlow Lite es una versión ligera de TensorFlow que está optimizada para ejecutar modelos de aprendizaje automático en dispositivos con recursos limitados, como microcontroladores y dispositivos móviles. Está diseñado para ejecutar modelos de aprendizaje automático en microcontroladores y otros dispositivos con pocos kilobytes de memoria. Esto permite llevar la inteligencia artificial a dispositivos pequeños y de bajo consumo, como electrodomésticos, dispositivos de Internet de las cosas (IoT) y juguetes, sin requerir hardware costoso o conexiones a internet confiables[15].

**Keras-Tuner**

Keras Tuner es una biblioteca que ayuda a seleccionar el conjunto óptimo de hiperparámetros para programas de TensorFlow. El proceso de selección de los hiperparámetros adecuados para una aplicación de aprendizaje automático se conoce como ajuste de hiperparámetros o hiperajuste[16].

Keras-Tuner implementa varios métodos para búsqueda de hiperparametros óptimos:

* **Búsqueda aleatoria (Random Search)**: Este enfoque selecciona configuraciones de hiperparámetros al azar de un espacio de búsqueda predefinido.
* **Búsqueda en cuadrícula (Grid Search)**: En este enfoque, se definen valores específicos para cada hiperparámetro y se realiza una búsqueda exhaustiva en todas las combinaciones posibles.
* **Búsqueda de hiperbanda (Hyperband Search)**: Esta estrategia optimiza el uso de los recursos computacionales al asignar de manera eficiente el tiempo de entrenamiento a diferentes configuraciones de hiperparámetros
* **Búsqueda de optimización bayesiana (Bayesian Optimization Search)**: Utiliza técnicas de optimización bayesiana para encontrar de manera eficiente la mejor combinación de hiperparámetros.

### 2.2.5 Base Datos

El manejo de la información es de vital importancia en todo proyecto, tener almacenada de manera organizada y consistente los datos permiten un control y utilización óptima. Las bases de datos son una herramienta necesaria para lograr este objetivo.

Una base de datos es una recopilación de datos sistemática y almacenada electrónicamente. Puede contener cualquier tipo de datos, incluidos palabras, números, imágenes, vídeos y archivos. Puede usar un software denominado DBMS(Database Management System) para almacenar, recuperar y editar datos[17].

Algunas de la principales DBMS son:

* MySQL
* MariaDB
* SQLite
* PostgreSQL
* MongoDB

**SQLite**

SQLite es una herramienta de software libre, que permite almacenar información en dispositivos empotrados de una forma sencilla, eficaz, potente, rápida y en equipos con pocas capacidades de hardware. Las tablas se asocian mediante relaciones, las cuales definen cómo se vinculan conceptualmente dos o más tablas. Las relaciones más comunes son uno a uno, uno a muchos o muchos a muchos[18].

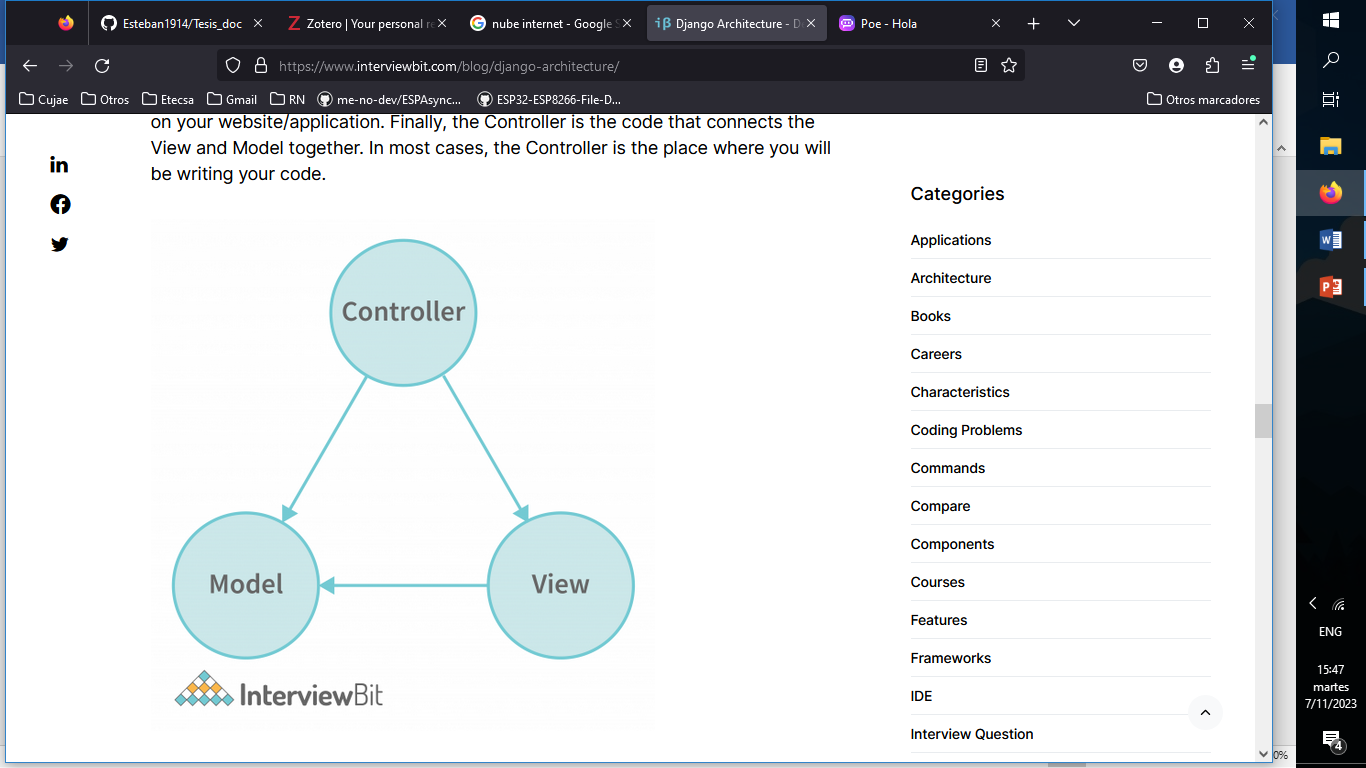
### 2.2.6 Django

Django es un marco web de alto nivel escrito en Python que se utiliza para el desarrollo rápido y eficiente de aplicaciones web, desplegándose tanto en el lado del cliente como del servidor. Django se basa en el principio del desarrollo ágil de software y sigue una filosofía de diseño DRY (Don't Repeat Yourself), lo que significa que promueve la reutilización de código y la eficiencia en el desarrollo.

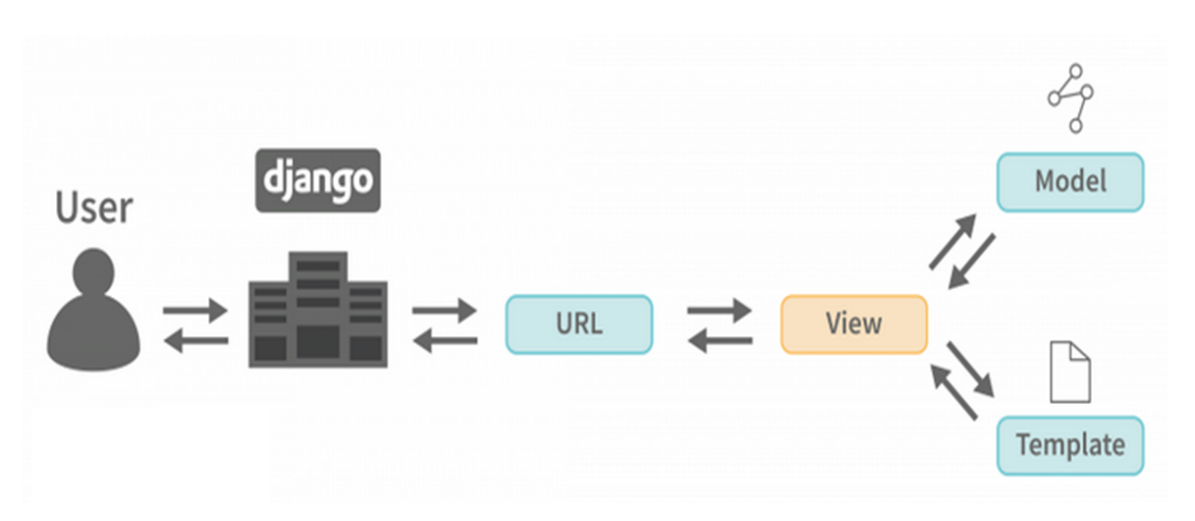
Django sigue el patrón de arquitectura MVC (Model-View-Controller). En este patrón, el modelo representa la estructura de datos de la aplicación y se encarga de interactuar con la base de datos. La vista se encarga de la lógica de presentación y la generación de la interfaz de usuario. El controlador maneja las solicitudes del usuario y coordina la interacción entre el modelo y la vista.

Django proporciona un ORM (Mapeo Objeto-Relacional) que permite interactuar con la base de datos utilizando objetos Python en lugar de escribir consultas SQL directamente. Esto simplifica el acceso y la manipulación de datos, ya que las consultas se expresan en términos de modelos y objetos Python en lugar de tablas y filas de la base de datos. El ORM de Django también se encarga de la abstracción de la base de datos, lo que significa que es compatible con varios motores de base de datos, como PostgreSQL o MySQL, por defecto ya viene integrado con SQLlite3.

Django utiliza un enfoque basado en URL (Uniform Resource Locator) para manejar las solicitudes web. Un archivo de configuración de URL define las URL y las asocia con las vistas correspondientes. Cuando un usuario realiza una solicitud a una URL específica, Django utiliza el archivo de configuración de URL para determinar qué vista debe manejar la solicitud. Esto permite una organización clara y mantenible de las rutas de la aplicación[19]–[21].



Esquema 1: Arquitectura MWC de Django

****

Esquema 2: Arquitectura basada en URL de Django

### 2.2.7 Lenguaje C

Es un lenguaje compilado, imperativo, estructurado y de bajo nivel. Se caracteriza por ser cercano al hardware y tener abstracciones como objetos. Opera directamente sobre registros y memoria, permitiendo un control completo.

El lenguaje C fue creado a inicios de los años 70 por Dennis Ritchie en los laboratorios Bell, con el propósito de ser portable, eficiente y tener un bajo nivel de abstracción. Rápidamente se convirtió en el lenguaje dominante para sistemas operativos y otro software crítico.

A pesar de contar con varios años, C sigue siendo ampliamente usado en sistemas embebidos, kernels de SO, drivers, programas de propósito general, videojuegos, motores gráficos y más. Su portabilidad, velocidad y control hacen que no tenga un reemplazo directo. También sirve de base para lenguajes modernos como C++

Ofrece un bajo nivel de abstracción que permite un control completo, pero requiere cuidado para evitar errores. Es portable, rápido y de código compacto, pero su sintaxis simple aumenta la posibilidad de bugs. Su aprendizaje exige dominar conceptos de programación de sistemas[22].

## 2.3 Detección Multivariable

La detección multivariable permite la clasificación de elementos a partir de varias variables de entradas, esta es una tarea primordial en muchos campos que involucran el análisis de datos y patrones. Solamente comprendiendo la naturaleza subyacente de los elementos que se observan es posible predecir su comportamiento futuro y actuar en consecuencia.

Contar con métodos robustos de clasificación multidimensional es indispensable. Esto se debe a que la mayoría de fenómenos reales involucran múltiples variables interrelacionadas que aportan información complementaria sobre la situación. Los métodos de aprendizaje estadístico multivariable permiten descubrir patrones sutiles que de otra forma pasarían desapercibidos. Al correlacionar dimensiones, se puede inferir el estado subyacente del objeto de estudio con mayor precisión.

### 2.3.1 Métodos Detección Multivariable

Para la detección multivariable existen distintos métodos[23]:

* **LDA** (Análisis de Discriminante Lineal): Es una técnica de clasificación supervisada que busca encontrar una combinación lineal de características que maximice la separación entre clases en un conjunto de datos de entrenamiento.
* **KNN** (K-vecinos más cercanos): Clasifica un nuevo punto de datos según la clase mayoritaria de sus k vecinos más cercanos en el espacio de características. La distancia se calcula utilizando una medida de distancia, como la distancia euclidiana.
* **Redes Neuronales**: Las redes neuronales son modelos computacionales inspirados en el cerebro humano. Consisten en capas de nodos interconectados (neuronas) que procesan y transmiten información. Las redes neuronales pueden aprender a clasificar datos ajustando los pesos de las conexiones entre las neuronas durante un proceso de entrenamiento.
* **Árboles de Decisión**: Los árboles de decisión son estructuras de tipo árbol donde cada nodo interno representa una característica o atributo, y las ramas del árbol representan las posibles opciones o valores para esa característica. Los árboles de decisión se construyen dividiendo los datos en función de las características más relevantes y se utilizan para clasificar nuevos ejemplos siguiendo el camino desde la raíz hasta las hojas del árbol.
* **SVM** (Support Vector Machines): Las Máquinas de Vectores de Soporte son modelos de aprendizaje automático que buscan encontrar un hiperplano óptimo que separe las diferentes clases de datos en un espacio de características.
* **Naive Bayes** (Clasificador Bayesiano Ingenuo): El clasificador bayesiano ingenuo se basa en el teorema de Bayes y asume independencia condicional entre las características. Calcula la probabilidad de que un ejemplo pertenezca a una clase determinada utilizando la probabilidad de la clase y la probabilidad de cada característica dado un valor de clase.
* **Random Forest** (Bosques Aleatorios): Los bosques aleatorios son un conjunto de árboles de decisión combinados. Cada árbol se entrena con una muestra aleatoria del conjunto de datos y produce una clasificación. La clasificación final se obtiene por votación o promediando las predicciones de los árboles individuales.

### 2.3.2 Introducción a Redes Neuronales

#### - Neurona Bilógica

Una red neuronal biológica es un conjunto de conexiones sinápticas ordenadas que se produce como resultado de la unión de las neuronas en el cerebro de los seres vivos. Estas redes son la base del sistema nervioso y son responsables de procesar y transmitir información en forma de señales eléctricas y químicas[24].

Si bien existen variados tipos de células nerviosas en la naturaleza, de manera esquemática una neurona típica presenta:

* Un cuerpo o soma donde se aloja el núcleo
* Un prolongamiento filiforme denominado axón
* Variadas ramificaciones arbóreas conocidas como dendritas
* Zonas sinápticas que le permiten comunicarse eléctrica y químicamente con otras neuronas.

El mecanismo funcional primario de las células nerviosas implica la recepción, procesamiento y propagación de impulsos eléctricos a través de la red neuronal. Dichos potenciales de acción se originan en las dendritas, recorren el axón y son transmitidos a otras neuronas a través de las uniones sinápticas[25].

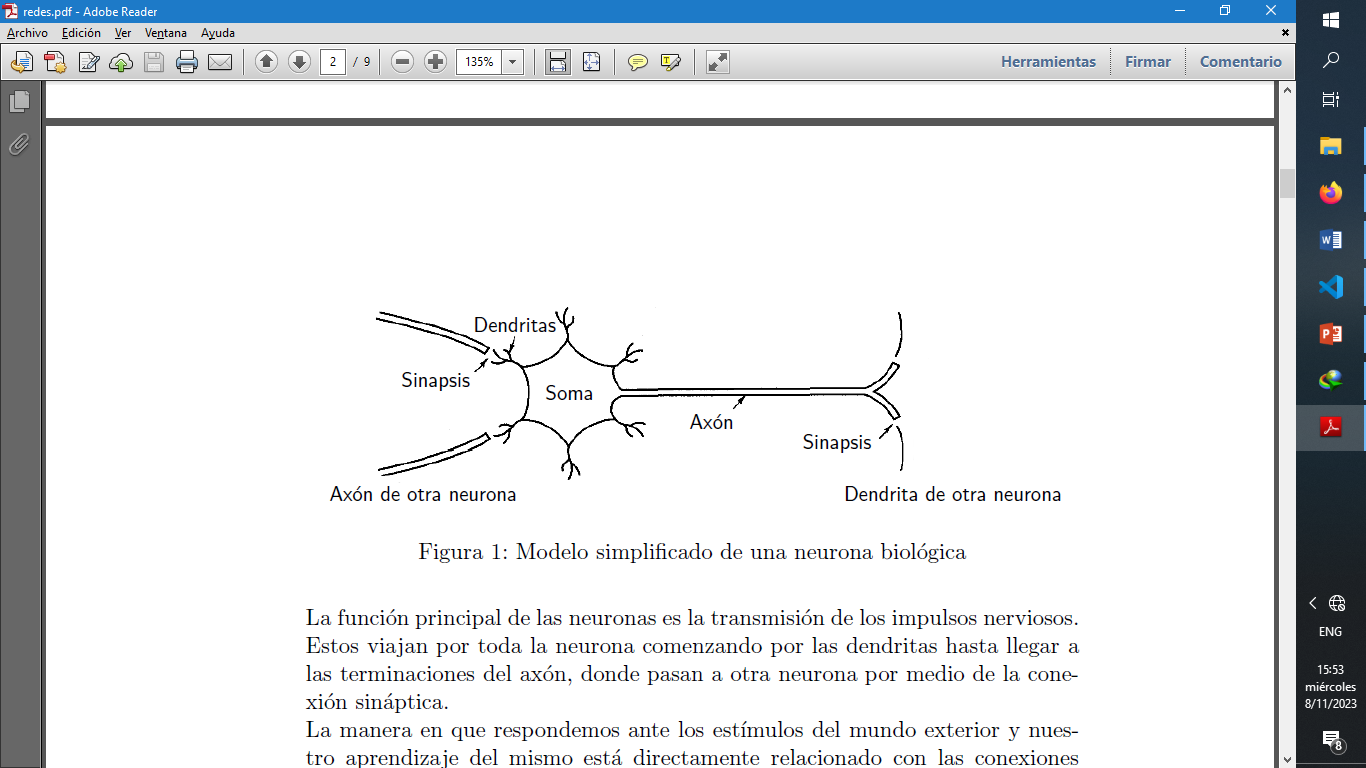
~~~~

Ilustración 12: Neurona Biológica

Las redes neuronales biológicas son la base del funcionamiento del cerebro humano y de otros organismos, permitiendo la realización de funciones complejas como el pensamiento, la memoria, el movimiento y las emociones.

#### - Neurona de McCulloch-Pitts

Los orígenes de las Redes Neuronales Artificiales modernas se remontan al 1943, cuando el psiquiatra y anatomista Warren McCulloch junto matemático Walter Pitts desarrollaron el que puede considerarse el primer modelo matemático de la unidad básica de procesamiento en estas redes: la neurona artificial.

Mediante una construcción formal muy simplificada que buscaba reflejar los elementos esenciales de la célula nerviosa biológica, McCulloch y Pitts sentaron las bases para que posteriormente pudieran simularse redes neuronales computacionalmente y usarlas para resolver diversas tareas.

Su modelo definía a la neurona como un procesador que recibe entradas binarias representando señales presinápticas, les asigna pesos y realiza sumas ponderadas. Luego aplica una función de decisión para producir una salida postsináptica. Pese a su carácter elemental, este enfoque abrió la puerta al desarrollo de aplicaciones pioneras[26].

La neurona consta de (ver Ilustración ???):

* Las entradas x e y
* Los pesos sinápticos w1 y w2 correspondientes a cada entrada
* Un término aditivo b
* Una función de activación f
* Una salida z

La suma ponderada de las entradas se calcula multiplicando cada entrada por su respectivo peso y sumándolas junto al valor del término aditivo.

(2.1)

La salida se determina aplicando una función de umbral a la suma ponderada. Si la suma ponderada es mayor o igual a un valor umbral, la salida es 1; de lo contrario, la salida es 0.

Si >= Umbral:

Salida = 1

Sino:

Salida = 0

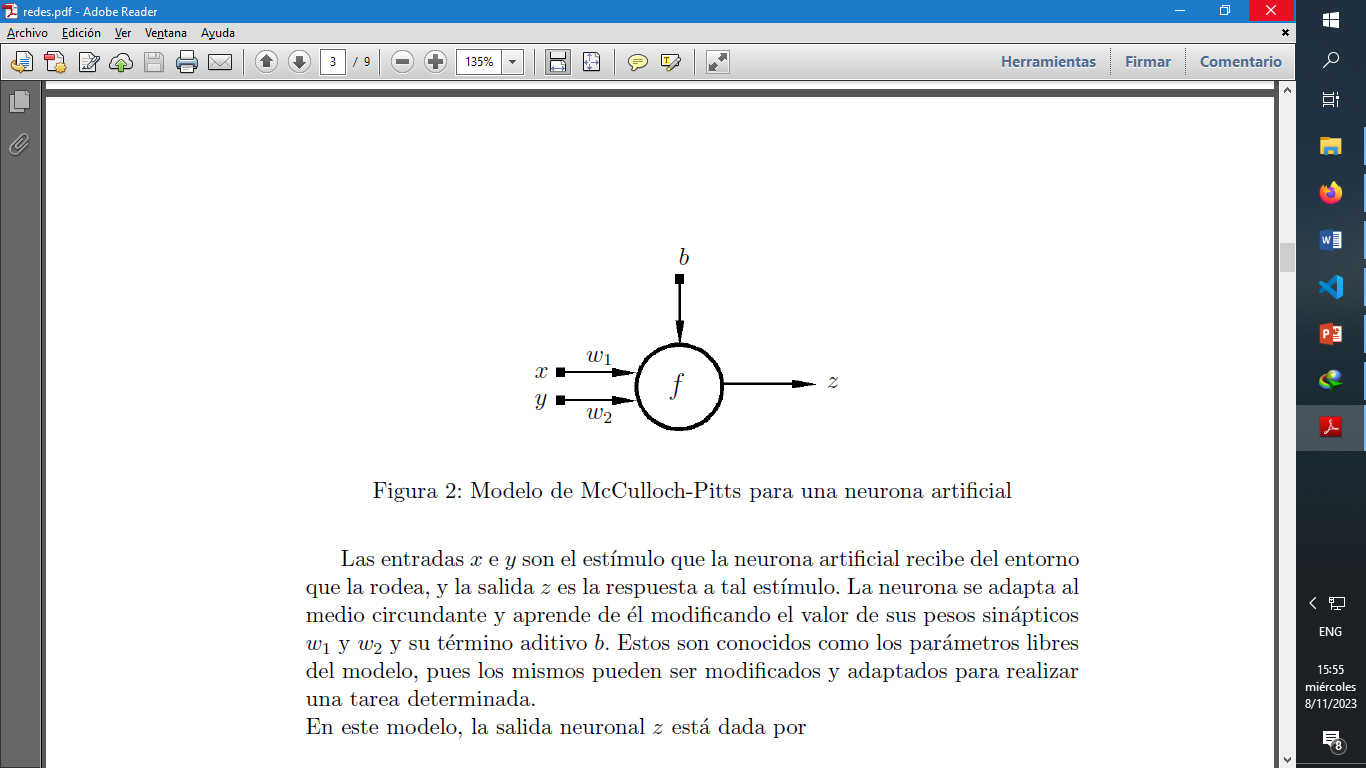
~~~~

Ilustración 13: Neurona McCulloch- Pitts

#### - Neurona Artificial

Unaneurona artificial no es más que un modelo matemático o computacional empleado en estadística, psicología cognitiva o inteligencia artificial, vagamente inspirado en el comportamiento observado en su homólogo biológico[24].

La neurona recibe múltiples señales independientes de entrada procedentes de salidas previas de otras neuronas o de datos externos. Consta de cuatro elementos principales:

* **Entradas**: Pueden ser datos binarios o continuos, dependiendo del modelo en cuestión, las entradas puedes provenir de otras neuronas.
* **Pesos**: Representa la magnitud de interacción de la neurona con otras neuronas enlazadas, cada neurona en el proceso de entrenamiento ira ajustando este valor
* **Función de activación:** Transforma la suma ponderada en la activación o salida de la neurona
* **Salida**: Salida de la neurona que es enviada hacia las siguientes neuronas o a la salida del sistema

**Propagación de la señal en una neurona artificial:**

Una neurona artificial recibe múltiples señales de entrada provenientes de las salidas de otras neuronas a través de las conexiones sinápticas entrantes. Cada entrada xi se multiplica por un peso sináptico asociado, representado la fortaleza de la conexión; seguido todos los productos son sumados junto a un peso adicional para un ajuste mayor de la neurona denominado Bias. La ecuación seria muy parecida a la expuesta por McCulloch-Pitts (ver ecuación 2.1)

Esta suma ponderada calculada constituye la entrada de la función de activación de la neurona. La función de activación transfiere no linealmente el impulso en la salida de activación a de la neurona, de acuerdo a una curva matemática preestablecida. Algunas de las funciones de activación más utilizadas son la sigmoide, tangente hiperbólica y RELU (Rectified Linear Unit), (ver tabla ???). La salida de la función de activación constituye, a su vez, la señal que será propagada por la neurona hacia las demás neuronas de destino o hacia la capa de salida en último caso[27].



Ilustración 14: Neurona Artificial

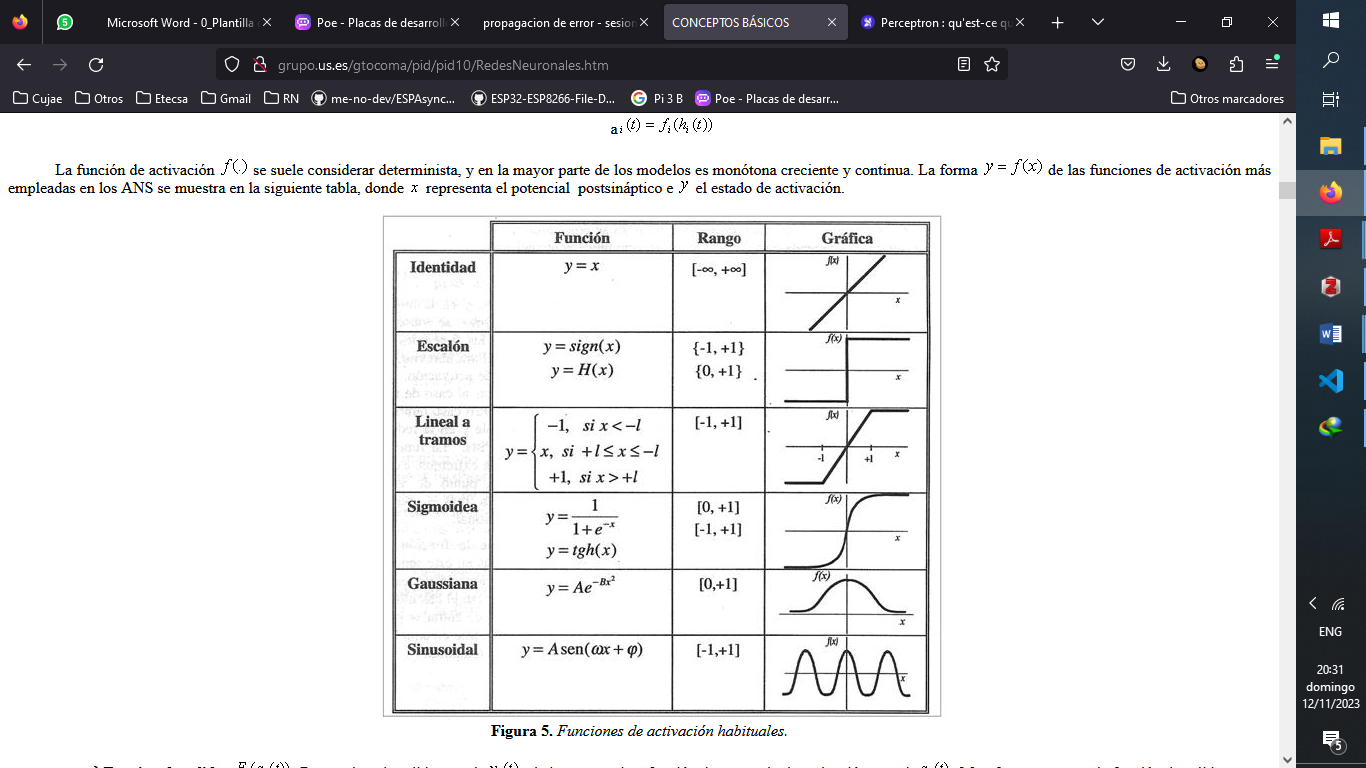


Tabla 4: Funciones de activación

### 2.3.3 Redes Neuronales

Las Redes Neuronales Artificiales (RNA) representan uno de los modelos de aprendizaje automático más poderosos y extendidos en la actualidad. Estas surgen inspiradas en el funcionamiento distribuido y paralelo del sistema nervioso biológico, el cual mediante la interconexión de miles de millones de neuronas es capaz de realizar complejos procesos de percepción, razonamiento y respuesta.

En una red neuronal, las neuronas artificiales se agrupan y organizan en estructuras denominadas "capas”. Una capa puede contener desde una única neurona hasta millones de ellas, dependiendo del tamaño y complejidad del modelo.

Dentro de la red neuronal se distinguen principalmente tres tipos de capas:

* **Capa de entrada**: Recibe los datos de entrada al modelo, como características o predictores.
* **Capas ocultas**: Ubicadas entre la capa de entrada y salida, su función es extraer representaciones internas de los datos. Es donde reside el aprendizaje.
* **Capa de salida:** Genera los resultados del modelo a partir de las representaciones extraídas por las capas previas. Puede ser regresión o clasificación.

Las capas ocultas, también llamadas "capas internas", contienen las neuronas encargadas de procesar y transformar progresivamente los patrones de entrada a lo largo de la red.

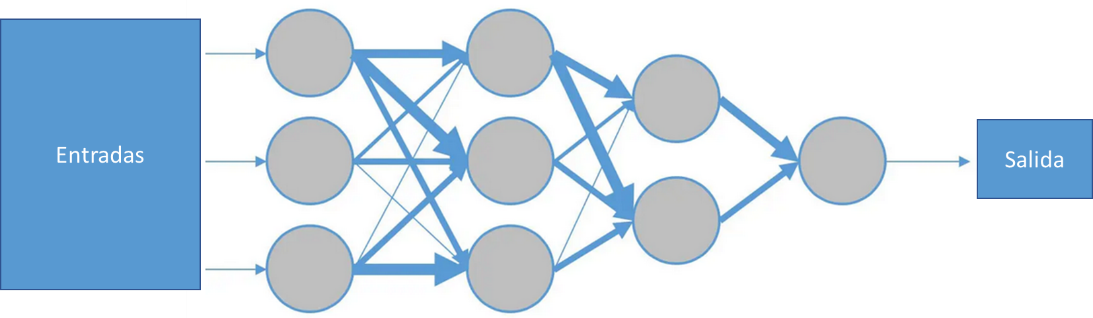


Ilustración 15: Red Neuronal Multicapa

### 2.3.4 Entrenamiento de Redes Neuronales

El entrenamiento de redes neuronales es un proceso fundamental en el campo del aprendizaje automático y el procesamiento de datos. Consiste en ajustar los pesos de las conexiones entre las neuronas de la red para que los resultados de la capa de salida se ajusten lo más posible a las respuestas correctas[27].

Formas de entrenar una red neuronal[28]:

* **Aprendizaje supervisado**: Se proporciona a la red neuronal un conjunto de datos de entrenamiento que contiene una etiqueta que indica la respuesta correcta. Este método es ampliamente utilizado en problemas de clasificación y regresión.
* **Aprendizaje no supervisado**: La red neuronal se entrena sin etiquetas. El objetivo es encontrar patrones o estructuras ocultas en los datos de entrada. Este enfoque es comúnmente utilizado en problemas de agrupamiento o detección de anomalías.
* **Aprendizaje por refuerzo**: En este tipo de entrenamiento, la red neuronal aprende a través de la interacción con un entorno. La red toma decisiones y recibe recompensas o castigos según su desempeño. El objetivo es maximizar la recompensa acumulada a lo largo del tiempo. Este enfoque es utilizado en problemas de toma de decisiones secuenciales, como juegos o robótica.
* **Transferencia de aprendizaje**: En este método, se aprovecha el conocimiento previamente aprendido por una red neuronal en un problema y se utiliza para acelerar el entrenamiento en otro problema relacionado. Esto es especialmente útil cuando se dispone de pocos datos de entrenamiento para el nuevo problema.

**Aprendizaje Supervisado**

El entrenamiento supervisado implica tener un conjunto de datos de entrenamiento etiquetados, donde se conocen las entradas y las salidas deseadas. El objetivo es ajustar los pesos y sesgos de la red neuronal para que pueda predecir correctamente las salidas a partir de las entradas.

Algunos algoritmos de entrenamiento supervisado[29]:

* **Descenso del gradiente**: Ajusta los pesos de la red de acuerdo con la dirección del gradiente de la función de pérdida. Hay diferentes variantes del descenso del gradiente, como el descenso del gradiente estocástico y el descenso del gradiente por lotes.
* **Retropropagación (Backpropagation)**: Se basa en la regla de la cadena y permite propagar el error desde la capa de salida hasta la capa de entrada de la red, ajustando los pesos en cada capa.
* **Algoritmo de Adam**: Es un algoritmo de optimización que combina el descenso del gradiente estocástico con técnicas de adaptación de la tasa de aprendizaje. El algoritmo de Adam ajusta la tasa de aprendizaje de forma adaptativa para cada parámetro de la red, lo que puede mejorar la velocidad de convergencia y la estabilidad del entrenamiento.

Durante el entrenamiento, los datos son presentados de forma iterativa a la red en formas de épocas; una época se refiere a una pasada completa de todo el conjunto de datos de entrenamiento durante el proceso de optimización. Inicialmente los pesos sinápticos se establecen de forma aleatoria. A continuación, la red calcula sus salidas y mide el error cometido respecto a las salidas objetivo, donde se van actualizando los pesos de todas las neuronas para ir reduciendo progresivamente ese error medio. De esta forma, ajustando los pesos a través de repetidos ciclos o épocas de entrenamiento sobre el conjunto de muestras, la red neuronal aprende las asociaciones subyacentes en los datos y mejora su habilidad para predecir las salidas correspondientes a nuevas entradas[27].

**Graficas de Evaluación**

Las gráficas de entrenamiento de pérdida y precisión son muy útiles para visualizar el progreso de un modelo de aprendizaje automático a medida que se entrena. Permiten ver si el modelo está aprendiendo efectivamente o si existe algún problema en el proceso de entrenamiento.

La gráfica de pérdida/época en machine learning muestra cómo cambia la función de pérdida a medida que el modelo se entrena a lo largo de las épocas. Por otro lado, la gráfica de precisión/época representa la evolución de la precisión del modelo a medida que se entrena en cada época. Ambas gráficas son herramientas importantes para evaluar el rendimiento y el progreso del modelo durante el entrenamiento.[30].

Al analizar la gráfica de precisión/época, se pueden observar diferentes patrones[31]:

1. **Convergencia**: Si la precisión del modelo aumenta y la función de pérdida disminuye de manera constante y ambas se estabilizan a medida que aumentan las épocas esto indica que el modelo está aprendiendo y mejorando su rendimiento.
2. **Sobreajuste**: Si la precisión del modelo en los datos de entrenamiento sigue aumentando mientras que la precisión en los datos de validación disminuye o se estabiliza, o la función de pérdida disminuye rápidamente en las épocas de entrenamiento, pero aumenta en las épocas de validación; puede indicar que el modelo se está sobreajustando a los datos de entrenamiento y no generaliza bien a nuevos datos.
3. **Inestabilidad**: Si la función de pérdida o la precisión muestran fluctuaciones significativas en las épocas de entrenamiento, puede indicar inestabilidad en el proceso de entrenamiento y la necesidad de ajustar hiperparámetros o realizar cambios en el modelo.

La gráfica de matriz de confusión es una representación visual de los resultados de un modelo de clasificación. Esta matriz muestra la relación entre las clases reales y las clases predichas por el modelo. Es una herramienta útil para evaluar el rendimiento de un modelo de aprendizaje automático. La matriz de confusión se organiza en forma de una tabla cuadrada, donde cada fila representa la clase real y cada columna representa la clase predicha. Los elementos de la matriz representan la cantidad de instancias clasificadas en cada categoría[32].

## 2.4 Conclusiones del capitulo

Las placas de desarrollo son sistemas electrónicos que están diseñadas para facilitar el desarrollo y la prototipado de proyectos electrónicos. Se analizaron placas de desarrollo populares para prototipado electrónico de bajo costo: la Raspberry Pi y la ESP32. Ambas permiten materializar proyectos electrónicos a precios asequibles, acortando los tiempos y costos de desarrollo frente a otras alternativas estas son herramientas ideales para pruebas preliminares y prueba de concepto. Además, cuentan con grandes comunidades de usuarios que facilitan el soporte y documentación para nuevos desarrolladores.

El sensor AS7265X también resulta ser una excelente opción para implementar funciones de detección de adulterantes en la leche de manera confiable y a bajo costo.

El desarrollo frontend permite implementar interfaces web amigables para el usuario final. La utilización de tecnologías como HTML, CSS y JavaScript facilitan la visualización y procesamiento de datos. El framework Django es idóneo para integrar el frontend de manera dinámica con el backend. El uso de SQLite como bases de datos aporta una solución sencilla de almacenamiento local para las placas de desarrollo.

Su nivel de abstracción bajo y control total sobre la memoria y procesador hicieron de C la opción ideal para programar algoritmos optimizados que extraen el máximo rendimiento del hardware.

La detección de adulterantes en la leche es un problema que involucra múltiples variables, por lo que se utilizó una metodología de detección multivariable basada en el espectro de luz. Esto permitió combinar parámetros como intensidades a diferentes longitudes de onda, mejorando significativamente los resultados. Analizar múltiples variables o características simultáneamente es necesario para obtener una visión más completa de un problema.

Los modelos de aprendizaje automático son inspirados en el cerebro humano, que consisten en neuronas artificiales conectadas en capas para realizar tareas de procesamiento de información. El perceptrón sentó las bases para el aprendizaje supervisado y la clasificación de patrones.

El entrenamiento de modelos de redes neuronales implica ajustar los pesos y conexiones entre las neuronas para que el modelo pueda aprender a realizar tareas específicas. Este proceso utiliza algoritmos de optimización, como el descenso del gradiente, para minimizar la función de pérdida y mejorar el rendimiento del modelo en los datos de entrenamiento. Estos conceptos forman parte de un campo en constante evolución que ha demostrado ser poderoso en una amplia gama de aplicaciones, desde reconocimiento de imágenes hasta procesamiento del lenguaje natural y más allá.

# Referencias

[1] «Tarjetas de desarrollo – Sistemas Digitales». Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://virtual.cuautitlan.unam.mx/intar/sistdig/tarjetas-de-desarrollo/

[2] «Raspberry Pi Documentation - Raspberry Pi hardware». Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://www.raspberrypi.com/documentation/computers/raspberry-pi.html#raspberry-pi-3-model-b

[3] «✅ Industrial ESP32 PLC - Product Range ideal for Internet of Things», Boot & Work Corp. S.L. Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://www.industrialshields.com/es\_ES/industrial-esp32-plc-products-family-ideal-for-iot-solutions

[4] «PLC ESP32 Múltiples aplicaciones para múltiples sectores – TUPUNATRON». Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://tupunatron.com/plc-esp32-solucion-ideal-para-iot/

[5] «The Internet of Things with ESP32». Accedido: 11 de noviembre de 2023. [En línea]. Disponible en: http://esp32.net/

[6] «AS7265X pdf, AS7265X Description, AS7265X Datasheet, AS7265X view ::: ALLDATASHEET »: Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://pdf1.alldatasheet.com/datasheet-pdf/view/1244817/AMSCO/AS7265X.html

[7] «I2C - Puerto, Introducción, trama y protocolo», HeTPro-Tutoriales. Accedido: 16 de noviembre de 2023. [En línea]. Disponible en: https://hetpro-store.com/TUTORIALES/i2c/

[8] «Conceptos básicos de HTML - Aprende desarrollo web | MDN». Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Learn/Getting\_started\_with\_the\_web/HTML\_basics

[9] «CSS | MDN». Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Web/CSS

[10] «JavaScript | MDN». Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://developer.mozilla.org/es/docs/Web/JavaScript

[11] M. O. contributors Jacob Thornton, and Bootstrap, «Get started with Bootstrap». Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://getbootstrap.com/docs/5.3/getting-started/introduction/

[12] «¿Qué es Python? - Explicación del lenguaje Python - AWS». Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://aws.amazon.com/es/what-is/python/

[13] «El tutorial de Python», Python documentation. Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://docs.python.org/3/tutorial/index.html

[14] «¿Qué es TensorFlow y para qué sirve?», ¿Qué es TensorFlow y para qué sirve? Accedido: 16 de noviembre de 2023. [En línea]. Disponible en: https://www.incentro.com/es-ES/blog/que-es-tensorflow

[15] «TensorFlow Lite para microcontroladores». Accedido: 16 de noviembre de 2023. [En línea]. Disponible en: https://www.tensorflow.org/lite/microcontrollers?hl=es-419

[16] «Introducción al afinador Keras | TensorFlow Core», TensorFlow. Accedido: 19 de noviembre de 2023. [En línea]. Disponible en: https://www.tensorflow.org/tutorials/keras/keras\_tuner?hl=es-419

[17] «¿Qué es una base de datos? - Explicación de las bases de datos en la nube - AWS», Amazon Web Services, Inc. Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://aws.amazon.com/es/what-is/database/

[18] «Information Modeling and Relational Databases - Terry Halpin, Tony Morgan - Google Books». Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://books.google.com/books?hl=en&lr=&id=puO\_VlbR\_x4C&oi=fnd&pg=PP1&dq=Halpin,+T.+(2014).+Information+modeling+and+relational+databases.+Morgan+Kaufmann.&ots=Ld1ArTpU2s&sig=jrVrSjLWVjO-FMAqUaf\_JU-ghOI#v=onepage&q&f=false

[19] «Django», Django Project. Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://www.djangoproject.com/

[20] «Django Architecture - Detailed Explanation - InterviewBit». Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://www.interviewbit.com/blog/django-architecture/

[21] A. Holovaty y J. Kaplan-Moss, *The Definitive Guide to Django: Web Development Done Right*. Apress, 2009.

[22] B. W. Kernighan y D. M. Ritchie, *El lenguaje de programación C*. Pearson Educación, 1991.

[23] C. Lozares Colina y P. López-Roldán, «El análisis multivariado: definición, criterios y clasificación», *Pap. Rev. Sociol.*, n.o 37, pp. 009-029, 1991, doi: 10.5565/rev/papers/v37n0.1594.

[24] «Red neuronal», *Wikipedia, la enciclopedia libre*. 19 de diciembre de 2020. Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://es.wikipedia.org/w/index.php?title=Red\_neuronal&oldid=131809201

[25] C. J. Tablada y G. A. Torres, «Redes neuronales artiﬁciales», *Rev. Educ. Matemática*, vol. 24, n.o 3, Art. n.o 3, 2009.

[26] «Neurona de McCulloch-Pitts», *Wikipedia, la enciclopedia libre*. 15 de enero de 2022. Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://es.wikipedia.org/w/index.php?title=Neurona\_de\_McCulloch-Pitts&oldid=140975434

[27] admin@xeridia.com, «Redes Neuronales artificiales | Blog Xeridia», Xeridia. Accedido: 12 de noviembre de 2023. [En línea]. Disponible en: https://www.xeridia.com/blog/redes-neuronales-artificiales-que-son-y-como-se-entrenan-parte-i

[28] Jesús, «Estos son los Elementos Básicos para Entrenar una Red Neuronal», DataSmarts. Accedido: 15 de noviembre de 2023. [En línea]. Disponible en: https://www.datasmarts.net/estos-son-los-elementos-basicos-para-entrenar-una-red-neuronal/

[29] «Opciones para entrenar una red neuronal de deep learning - MATLAB trainingOptions». Accedido: 15 de noviembre de 2023. [En línea]. Disponible en: https://www.mathworks.com/help/deeplearning/ref/trainingoptions\_es.html

[30] «Monitorizar el progreso del entrenamiento de deep learning - MATLAB & Simulink». Accedido: 29 de noviembre de 2023. [En línea]. Disponible en: https://www.mathworks.com/help/deeplearning/ug/monitor-deep-learning-training-progress\_es.html

[31] manashgoswami, «Evaluación de los resultados de los experimentos de aprendizaje automático automatizado - Azure Machine Learning». Accedido: 29 de noviembre de 2023. [En línea]. Disponible en: https://learn.microsoft.com/es-es/azure/machine-learning/how-to-understand-automated-ml?view=azureml-api-2

[32] «La matriz de confusión y sus métricas – Inteligencia Artificial –». Accedido: 29 de noviembre de 2023. [En línea]. Disponible en: https://www.juanbarrios.com/la-matriz-de-confusion-y-sus-metricas/

# Anexos