|  |
| --- |
|  |

**Manual técnico**

Proyecto PIA: OPTIMIZACIÓN DE CADENAS DE SUMINISTRO AGRÍCOLA CON MACHIN LERNING Y DEEP LESNING

**Manual Técnico**

**1. Descripción del Sistema**

* Arquitectura General: El sistema sigue una arquitectura en capas con módulos de entrada, procesamiento de datos, modelos de ML/DL, y visualización de resultados.
* **Tecnologías Utilizadas:**
  + Backend: Python, Flask (API REST).
  + Machine Learning: Scikit-Learn, TensorFlow, PyTorch.
  + Base de Datos: PostgreSQL.
  + Frontend: HTML/CSS, JavaScript (React para visualización).

**2. Componentes del Sistema**

* Ingestión de datos: Módulo que permite la carga y limpieza de datos históricos.
* Modelos de predicción:
  + Modelos de regresión y clasificación para predicción de demanda.
  + Redes neuronales recurrentes (RNN) y redes neuronales convolucionales (CNN) para series temporales y optimización de inventario.
* Optimización de inventario: Algoritmos de optimización lineal y programación dinámica para definir niveles óptimos de stock.
* Generación de rutas: Algoritmo de optimización (p. ej., Dijkstra o A\*) para cálculo de rutas más eficientes.

**3. Flujo de Datos**

1. Ingestión y Preprocesamiento: Los datos se cargan y pasan por una serie de transformaciones para asegurar su calidad.
2. Entrenamiento de Modelos: Los modelos de ML y DL se entrenan con datos históricos.
3. Predicción y Optimización: Los modelos realizan predicciones, y el sistema ajusta las decisiones de inventario y rutas.
4. Visualización y Reporte: Los resultados se envían al frontend para su visualización y generación de reportes.

**4. Guía de Implementación**

* Configuración: Asegurar que los archivos de configuración (config.ini) tengan los parámetros correctos de conexión y otros ajustes específicos.
* Entrenamiento de modelos: Los modelos pueden ser ajustados con diferentes hiperparámetros para mejorar su rendimiento.
* Testing: Se incluyen pruebas unitarias en tests/ que deben ejecutarse antes de cada despliegue.
* Despliegue: Desplegar en servidor (Docker recomendado) siguiendo pasos detallados en deployment.md.

**5. Mantenimiento y Escalabilidad**

* Monitoreo de Modelos: Definir métricas de desempeño (accuracy, RMSE) y alertas cuando el rendimiento disminuye.
* Actualización de Modelos: Reentrenar los modelos periódicamente con nuevos datos.
* Escalabilidad: Usar servidores en la nube para carga y procesamiento masivo (AWS, GCP, Azure).

**6. Seguridad y Privacidad**

* Autenticación y Autorización: Autenticación basada en tokens JWT.
* Encriptación de Datos: Asegurar que todos los datos sensibles estén encriptados en la base de datos.
* Auditoría: Log de accesos y cambios importantes en el sistema.

**7. Solución de Problemas Técnicos**

* Fallas en entrenamiento: Revisar formato de datos y verificar dependencias.
* Errores de predicción: Analizar métricas de error y revisar la calidad de los datos.

**Anexos**

* **Diagramas UML: Descripción visual de la arquitectura.**
* **Documentación de API: Detalles de los endpoints disponibles en el backend.**