Guía para la estructura organizacional de los datos

Proceso

Gestión Estratégica de Tecnologías de la Información

Versión 1

25/04/2024

TABLA DE CONTENIDO

1. INTRODUCCIÓN

En la era del análisis de datos, la gestión y organización de grandes volúmenes de información es parte de las actividades prioritarias de cualquier organización; por ello el Ministerio de Ambiente y Desarrollo Sostenible tiene como estrategia la adopción del AWSS3 el cual es parte del storage del Data Lake, una solución de almacenamiento en la nube que proporciona la infraestructura para almacenar, recuperar y administrar conjuntos de datos. Sin embargo, la eficacia de esta herramienta no solo depende de su capacidad de almacenamiento, sino también de cómo se estructuran y nombra a los datos dentro de ella.

2. OBJETIVO

Definir la estructura de almacenamiento de los conjuntos de datos dentro del lago de datos del Ministerio de Ambiente y Desarrollo Sostenible.

3. ALCANCE

Esta Guía detalla la estructura de directorios y nomenclatura para gestionar diferentes conjuntos de datos en AWS S3, incluyendo recomendaciones adicionales para garantizarla unicidad y organización de las fuentes de datos.

4. ROLES Y RESPONSABILIDADES

Ingeniero de Datos: Es el responsable de acceder y administrar técnicamente el conjunto de datos dentro de la infraestructura donde están almacenados.

Custodio de Datos: Es responsable de la definición, administración y gestión de los datos por fuente de datos.

5. NORMAS Y POLÍTICAS

- Toda información que se ingesta en el lago de datos debe estar registrada en el sistema de gestión de datos y metadatos.

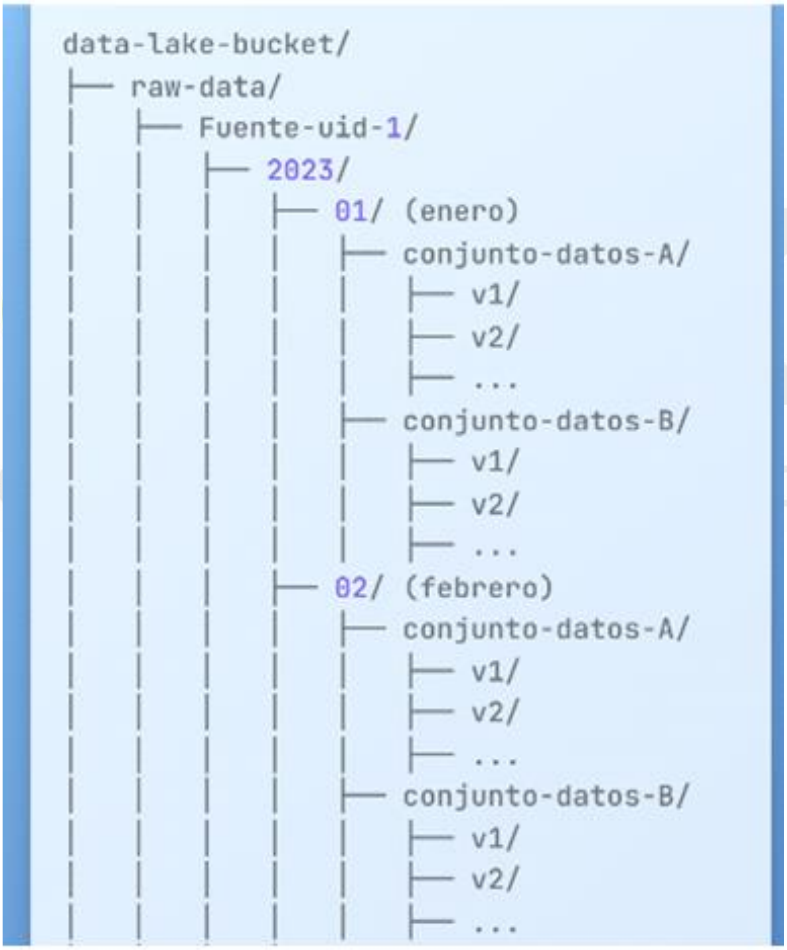
- Para la ingesta de datos, el ingeniero de datos debe implementar el mecanismo definido desde el registro de la fuente, con el acompañamiento del Custodio de Datos.

6. ESTRUCTURA DE DIRECTORIOS Y NOMENCLATURA PARA LA GESTIÓN DE CONJUNTOS DE DATOS EN AWSS3

6.1 Estructura de Directorios

Para construir una estructura de directorios en S3 que refleje adecuadamente la naturaleza versionada de los conjuntos de datos y permita su fácil gestión, es esencial mantener una estructura organizada y coherente. Pasos por seguir:

Ilustración 1 Estructura de carpetas en ambientes del S3 en AWS



Esta figura refleja la estructura de directorios propuesta en el documento, con carpetas para datos sin procesar, datos procesados, resultados de análisis, scripts y conjuntos de datos versionados.

6.2 Estructura de las Carpetas del S3 en AWS:

- **data-lake-bucket:** Este es el bucket principal de S3 donde se almacenan todos los datos del data lake.
- **raw-data:** Aquí es donde se almacenan los datos en su forma original sin ningún procesamiento.
- **processed-data:** Después de que los datos brutos se procesan (limpieza, transformación, etc.), se almacenan aquí.
- **analytics:** Aquí es donde se almacenan los resultados de los análisis realizados en los datos procesados.
- **scripts:** Contiene todos los scripts utilizados para procesos ETL y análisis de datos.
- **conjunto-datos-A, conjunto-datos-B, ...:** Estos son diferentes conjuntos de datos que se almacenan en el datalake.
- **v1, v2, ...:** Representa las diferentes versiones de cada conjunto de datos. Cada vez que se actualiza un conjunto de datos, se crea una nueva versión, lo que permite el seguimiento y la gestión de cambios en el tiempo.

Para garantizar la unicidad de las fuentes de datos, se recomienda la siguiente estructura:

Fuente-uid: Cada fuente de datos debe estar contenida en una carpeta específica con un nombre único generado por un UUID (Universal Unique Identifier) para asegurar que no existan duplicados de fuentes de datos. Dentro de esta carpeta, se pueden segmentar los datos por fechas, creando subcarpetas para cada año y mes.

Ilustración 2 Estructura para garantizar la unicidad de las fuentes de datos

```
data-lake-bucket/
├── raw-data/
│   ├── Fuente-uid-1/ (por ejemplo, 7f4dc6b5-75d2-4e6e-9a8c-65a6620f6c7a)
│   │   ├── 2023/
│   │   │   ├── 01/ (enero)
│   │   │   ├── 02/ (febrero)
│   │   │   ├── ...
│   ├── Fuente-uid-2/ (por ejemplo, 8a4e1c3f-6a5b-4d3d-8b9e-71f3635d8c8d)
│   │   ├── 2023/
│   │   │   ├── 01/ (enero)
│   │   │   ├── 02/ (febrero)
│   │   │   ├── ...
│   ├── ...
```

6.3 Beneficios de esta Estructura para el DataLake:

- **Unicidad de fuentes de datos:** El uso de UUID garantiza que cada fuente de datos sea única, evitando duplicados y posibles conflictos.

- **Seguimiento temporal:** La organización por año y mes permite un seguimiento claro de cuándo se cargaron los datos y facilita la gestión de datos históricos.

- **Separación de modelos y versiones:** Cada modelo de IA y/o conjunto de datos se mantiene separado y versionado, lo que facilita la administración y la referencia a versiones anteriores.

- **Scripts independientes:** Los scripts se mantienen separados de los datos, lo que simplifica la gestión y el mantenimiento de los procesos ETL y análisis.

- **Organización clara:** La estructura proporciona una organización clara y coherente para los proyectos de IA, loque facilita la gestión y el acceso a los datos y resultados.

6.4 Explicación de la Nomenclatura:

Nomenclatura:

Versionado:

- Formato: vX

- Ejemplo: v1, v2, v3.

- Descripción**:** Se utiliza el formato vX para versionar los datos dentro de cada proyecto. Esto permite tener múltiples versiones del mismo conjunto de datos, lo que resulta beneficioso para gestionar cambios, actualizaciones y mantener un historial de las modificaciones realizadas.

Nombres de conjuntos de datos:

- Formato: [nombre_del_proyecto]
  Ejemplo: análisis_sentimientos, noticias_falsas, descriptivo_sirh.

  Descripción: Los nombres de los conjuntos de datos son descriptivos y reflejan la naturaleza del proyecto. Se utilizan guiones bajos (_) para separar palabras y lograr una nomenclatura clara y comprensible.

- Formato de Archivos con Fecha:
  Formato: [descripción_del_archivo]_AAAA-MM-DD.[extensión]
  Ejemplo: datos_demograficos_2023-04-21.csv, modelo_entrenado_v1_2023-04-21.pkl.

Descripción: Los nombres de los archivos comienzan con una descripción que refleja su contenido. Luego, se incluye la fecha en la que se creó o modificó el archivo en formato AAAA-MM-DD. Finalmente, se agrega una extensión que indica el formato del archivo, como .csv para datos tabulares o .pkl para modelos serializados.

Formato de archivos:

- Descripción: Los archivos dentro de cada directorio tienen un prefijo que refleja su contenido, como "opiniones" o "modelo_entrenado," seguido de una extensión que indica su formato, como ".csv" o ".pkl." Esto proporciona una identificación clara del tipo de datos que contiene cada archivo.

6.5 Fases de los Modelos de IA para los Proyectos:

La división de cada modelo de IA en tres fases (raw, processed y analytics) tiene varios propósitos y beneficios:

- **Organización y Claridad:** Cada fase tiene un propósito específico, lo que facilita la organización y la identificación de los datos en diferentes etapas de procesamiento. Los datos sin procesar se almacenan en "raw," los datos limpios o transformados en "processed," y los resultados y modelos en "analytics."

- **Seguimiento de Procesamiento:** Permite un seguimiento claro del proceso de procesamiento de datos. Los datos fluyen desde "raw" (sin procesar) a "staged" (procesados) y finalmente a "analytics" (resultados). Esto facilita la comprensión de cómo se llegó a los resultados.

- **Reproducibilidad:** Al mantener un registro de las diferentes versiones de datos en cada fase, se facilita la reproducibilidad de los análisis y modelos. Puedes volver a una versión anterior en caso de necesidad.

- **Colaboración:** La división en fases permite una colaboración más eficiente entre los miembros del equipo. Cada fase tiene un propósito claro, lo que simplifica la colaboración en diferentes etapas del proyecto.

- **Mantenimiento de Versiones:** Cada fase puede tener sus propias versiones (v1, v2, etc.), lo que brinda un control preciso sobre las modificaciones realizadas en cada etapa del proceso.

6.6 Diseño del UUID para identificar los recursos:

El UUID se diseña siguiendo una estructura específica que se adapta al contexto de las PoC y se utiliza para identificar de manera única cada modelo de IA desarrollado en la plataforma. Esta estructura consta de varios elementos clave:

- **Prefijo de Oficina o Proyecto (aws):** El UUID comienza con un prefijo que identifica al cliente o proyecto principal. En este caso, "aws" se utiliza para identificar el cliente o proyecto principal que utiliza el lago de datos para desplegar modelos de IA.

- **Número de Componente (002, 003, etc.):** El número de componente se utiliza para indicar el orden de desarrollo del modelo dentro del proyecto "bosques". Cada número representa un modelo específico y suposición en el desarrollo. Por ejemplo, "002" sería el segundo modelo desarrollado por "bosque", mientras que"003" sería el tercero, y así sucesivamente.

- **Nombre del Componente** (bosques_001_imagenes_covima, bosques_002_pronóstico_ bosques, etc.): A continuación del número de componente, se incluye el nombre del componente de IA. Este nombre describe la funcionalidad principal del modelo. Por ejemplo, "bosques_imagenes_covima" indica que el modelo se especializa en la predicción de imágenes de árboles, mientras que " bosques_002_pronóstico_ bosques" se refiere a un modelo de predicción de inversión.

Propósito del UUID:

El diseño del UUID se implementa con los siguientes propósitos:

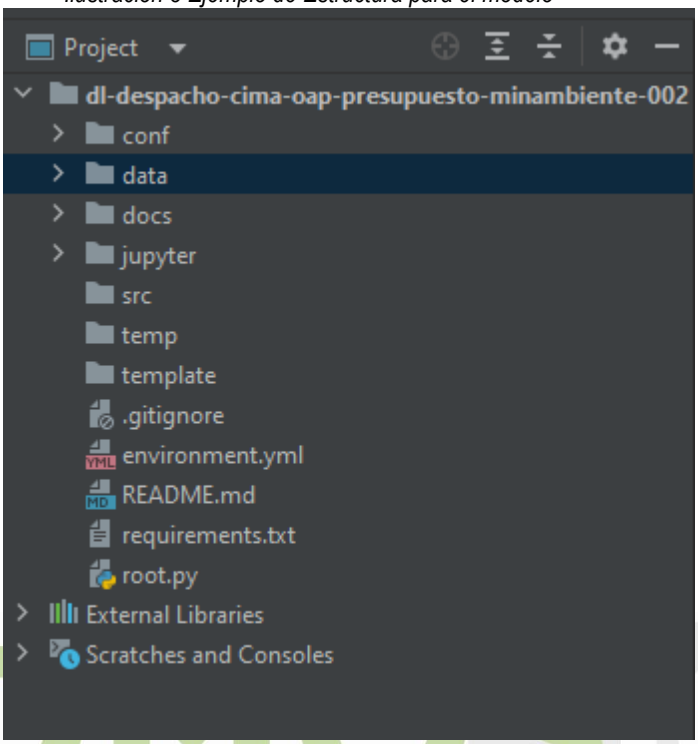
- **Identificación Única:** El UUID garantiza que cada modelo de IA tenga una identificación única dentro del proyecto DL. Esto evita cualquier ambigüedad en la referencia o gestión de modelos y datos relacionados.

- **Orden y Seguimiento:** El número de componente ayuda a ordenar los modelos en función de su desarrollo cronológico. Esto permite un seguimiento claro de cuándo se crearon los modelos y en qué secuencia.

- **Clasificación Funcional:** El nombre del componente describe la función principal del modelo, lo que facilita la identificación de modelos específicos según su especialización.

- **Evitar Conflictos de Nombres:** La combinación de prefijo de cliente o proyecto, número de componente y nombre del componente reduce la probabilidad de conflictos de nombres entre modelos de diferentes oficinas o proyectos en la plataforma DL.

Ilustración 3 Ejemplo de Estructura para el modelo



6.7 DESARROLLO DEL PoC

Diseño de Arquitectura

Debe existir un diseño de la arquitectura de la solución y este debe ser modular, favoreciendo la escalabilidad y la integración fluida con sistemas preexistentes. Se adopta un enfoque basado en microservicios, donde cada servicio encapsula una capacidad analítica específica. Esto permite la actualización y el mantenimiento independientes, además de facilitar la prueba de conceptos individuales sin afectar el sistema completo.

Desarrollo del Modelo

Para la selección y el entrenamiento de modelos de aprendizaje automático, se utilizan datos representativos y se asegura su reproducibilidad y trazabilidad. Esto implica:

- **Entrenamiento del Modelo:** Seleccionar algoritmos adecuados y entrenar los modelos utilizando frameworks como TensorFlow o PyTorch.

- **Versionado del Modelo:** Utilizar herramientas como DVC (Data Versión Control) para

mantener un historial de las versiones del modelo.

- **Serialización del Modelo:** Guardar los modelos entrenados en un formato serializado (como " **.pkl"** para Python pickle) para su posterior carga y evaluación.

Estructura de Carpetas y Código

La estructura de carpetas en el repositorio se organiza de la siguiente manera:

- **rc/**: Contiene el código fuente de la aplicación, incluyendo scripts de entrenamiento y evaluación de modelos, así como la lógica de negocio.

- **models/** : Almacena los modelos de machine learning entrenados y serializados. Es esencial para la transferencia entre las fases de desarrollo y producción.

- **data/** : Incluye los datos utilizados para el entrenamiento y la evaluación del modelo. Esta carpeta debería contener también un archivo " **.gitignore"** para excluir los datos del control de versiones por razones de seguridad y privacidad.

7 TÉRMINOS Y DEFINICIONES

ETL: Es un tipo de integración de datos que hace referencia a los tres pasos (extraer, transformar, cargar) que se utilizan para mezclar datos de múltiples fuentes.

Ingesta de datos: Es el proceso de importar grandes archivos de datos de múltiples fuentes a un único sistema de almacenamiento

PoC: El objetivo principal del desarrollo de un PoC es demostrar la funcionalidad y, por tanto, verificar un determinado concepto o teoría que se puede alcanzar en su desarrollo. La prueba de concepto también se conoce como prueba de principio.

UUID (Universal Unique Identifier): Un UUID es un valor de 128 bits que se genera utilizando una combinación de la hora actual, la dirección de red del sistema y un número generado aleatoriamente.

8 BIBLIOGRAFÍA

https://docs.aws.amazon.com/glue/latest/dg/transforms-uuid.html

https://docs.aws.amazon.com/AmazonS3/latest/userguide/using-folders.html

https://docs.amplify.aws/cli/reference/files/#clijson