Impacto de la Contaminación Atmosférica en los Casos de Asma en California

Grupo Morado:
Julio Agüero Encinas
Fabián Camargo
Miguel Corvo Domosti
Samuel Martín Martínez
Jiaze Zhang

Curso 2025-2026

Índice general

1. Fuentes de datos **2**
1.1. Datos de calidad del aire . . . 2
1.1.1. Geografía de California . . . 3
1.1.2. Descarga de datos históricos . . . 7
1.1.3. Procesamiento de los datos descargados . . . 9
1.2. Datos de prevalencia de asma en California . . . 11

2. Plan de Preservación **12**
2.1. Almacenamiento . . . 12
2.1.1. Almacenamiento físico . . . 12
2.2. Gestión de los datos . . . 13
2.2.1. Limpieza y curación . . . 13
2.2.2. Metadatos . . . 14
2.2.3. Acceso y reutilización . . . 14
2.3. Seguridad . . . 14
2.4. Estrategia de preservación a largo plazo . . . 15
2.4.1. Backup . . . 16
2.4.2. Almacenamiento en la nube . . . 17

A. Código para la fase ETL **18**

Capítulo 1

Fuentes de datos

En este capítulo se describen las fuentes de datos utilizadas en el análisis del impacto de la contaminación atmosférica en los casos de asma en California.
Se detallan las características de cada conjunto de datos, su origen, y cómo se integran en el estudio. En este estudio se han empleado principalmente dos fuentes de datos:

- Datos de calidad del aire obtenidos de la iniciativa `OpenAQ`.
- Datos de prevalencia de asma en la población californiana proporcionados por `data.gov`, la plataforma de datos abiertos del gobierno de Estados Unidos.

Para la extracción y filtrado de los datos, se han realizado una serie de tareas que se describirán en detalle en las secciones siguientes.
Todo este proceso ha sido recogido en un cuaderno de Jupyter, que se puede localizar en el repositorio del proyecto, en la ruta `code/ETL.ipynb`.

A lo largo de las secciones, se facilitarán algunas capturas de pantalla de fragmentos del cuaderno para ilustrar ciertas partes del proceso.

1.1. Datos de calidad del aire

Los datos de calidad del aire se han obtenido de la iniciativa `OpenAQ`, que proporciona acceso abierto a mediciones de contaminantes atmosféricos a nivel global.

Entre otros contaminantes, las entidades colaboradoras exponen datos sobre niveles de dióxido de nitrógeno (NO_2), ozono (O_3) y material particulado fino ($PM_{2,5}$), que son relevantes para el estudio del asma.

Además de ofrecer un visor web, `OpenAQ` cuenta con una API que permite la descarga programática de datos históricos y en tiempo real. En la documentación oficial de la API se detallan los *endpoints* disponibles, los parámetros de consulta y los formatos de respuesta.

En concreto, es de particular relevancia el hecho de que la API tiene un límite de peticiones para distintas franjas temporales. En el caso de California, ha sido de relevancia considerar los siguientes límites, cuyas razones revelaremos más adelante:

- 60 peticiones por minuto.
- 2000 peticiones por hora.

Estas restricciones han sido el principal cuello de botella a la hora de descargar grandes volúmenes de datos históricos.

1.1.1. Geografía de California

Los datos expuestos por el gobierno de Estados Unidos están desglosados por condados, que son las subdivisiones administrativas de primer nivel dentro de los estados.

Esto es de relevancia, puesto que las estaciones de medición de calidad del aire exponen su ubicación geográfica en coordenadas de latitud y longitud (EPSG:4326), por lo que es necesario asignar cada estación a su condado correspondiente.
Para este cometido, se han realizado dos tareas principales:

- **Obtener los límites geográficos de los condados de California.**

  Para ello, se han empleado los datos de *shapefiles* proporcionados por el *U.S. Census Bureau*[1]. Estos archivos contienen la geometría de los condados, que se ha utilizado para determinar si una estación de medición se encuentra dentro de un condado específico.

- **Filtrar las estaciones de medición mediante un `bbox`**

  (*bounding box*, i.e., un rectángulo delimitador) que englobe toda California, puesto que la API de `OpenAQ` permite filtrar por este criterio.

A continuación se muestra el código empleado para descargar los shapefiles de los condados:

```
1
2 import requests
3 from pathlib import Path
```

[1] `https://www2.census.gov/geo/tiger/TIGER2025/COUNTY/tl_2025_us_county.zip`

```
# download county shapefiles from US Census Bureau
shapes_url =
↪  'https://www2.census.gov/geo/tiger/TIGER2025/COUNTY/tl_2025_us_county.zip'
geo_dir = Path("data/geographical")
geo_dir.mkdir(parents=True, exist_ok=True)
shapes_path = geo_dir / "tl_2025_us_county.zip"
if not shapes_path.exists():
    print(f"Downloading county shapefiles from {shapes_url}...")
    r = requests.get(shapes_url)
    with open(shapes_path, 'wb') as f:
        f.write(r.content)
    print(f"Downloaded to {shapes_path}")
else:
    print(f"Shapefiles already exist at {shapes_path}")

```

El archivo comprimido descargado se encuentra en la ruta `data/geographical/tl_2025_us_county.zip`, cuya versión descomprimida se halla en la sub-carpeta `california_counties`; en caso de que el lector desee explorar los archivos.

En la Figura 1.1 se muestra una captura de pantalla de los condados de Estados Unidos visualizados en QGIS, un software de sistemas de información geográfica (*GIS*).
Los condados de California están resaltados en rosa.

Son de particular interés las siguientes columnas:

- `STATEFP`: código FIPS del estado (California es 06).
- `NAME`: nombre del condado, valor que empleamos para relacionarlo con los datos de asma.
- `geometry`: geometría del condado en formato poligonal, es decir, una cadena de texto en formato WKT (*Well-Known Text*).

  Se facilita un ejemplo ilustrativo en la Figura 1.2.

Para procesar estos datos geográficos, se ha empleado la librería `geopandas`, que extiende las funcionalidades de `pandas` para manejar datos espaciales.
En concreto, nos permite usar el método `Polygon.contains()` para determinar si las coordenadas de una estación de medición están dentro de la geometría de un condado.

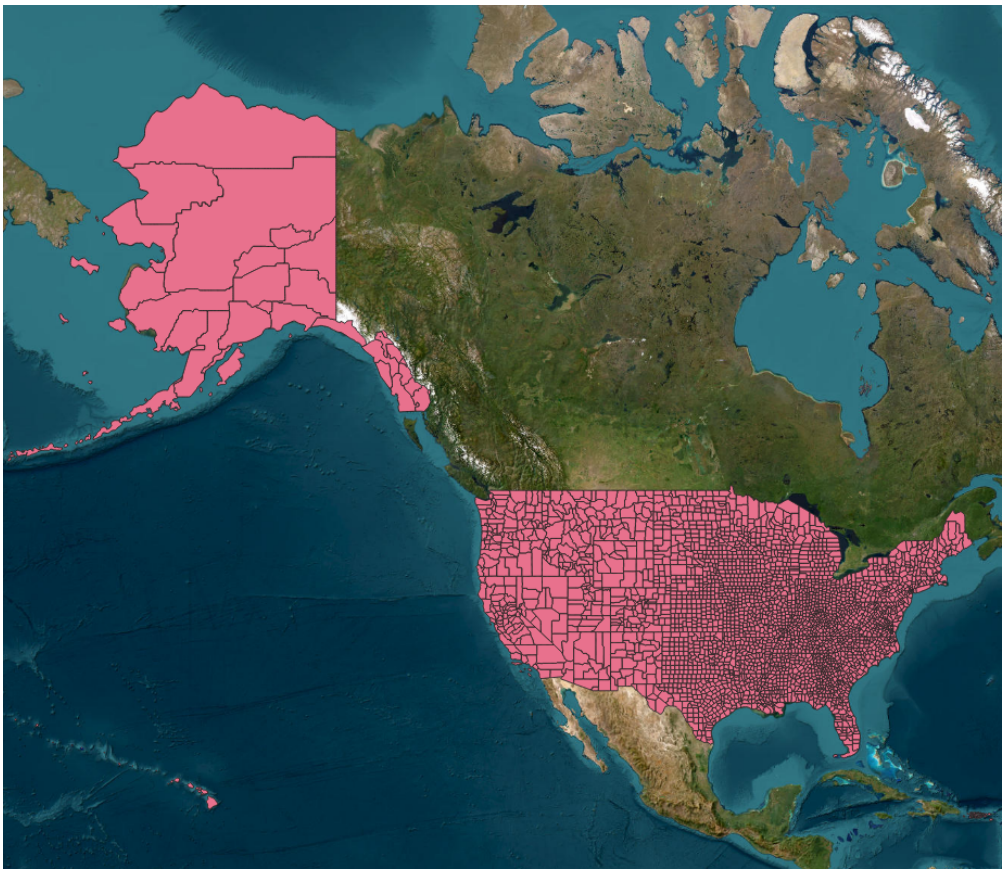

Figura 1.1: Visualización de condados en QGIS

```
POLYGON ((-122.373121 37.883884, -122.371142 37.884364, ... ))
```

Figura 1.2: Ejemplo: geometría en WKT (ilustrativo)

Una vez que tenemos definido el flujo a seguir para clasificar las estaciones por condado, procedemos a exponer la cuestión de decidir cómo filtrar las estaciones de interés a través de la API de `OpenAQ`.

Como ya se ha mencionado, la API permite filtrar las estaciones mediante un *bounding box*, que se define por las coordenadas de sus esquinas suroeste y noreste.

Para California, simplemente se ha usado QGIS para obtener unas coordenadas aproximadas que engloban todo el estado. En la Figura 1.3 se muestra una captura de pantalla de QGIS donde se procede a copiar las coordenadas.

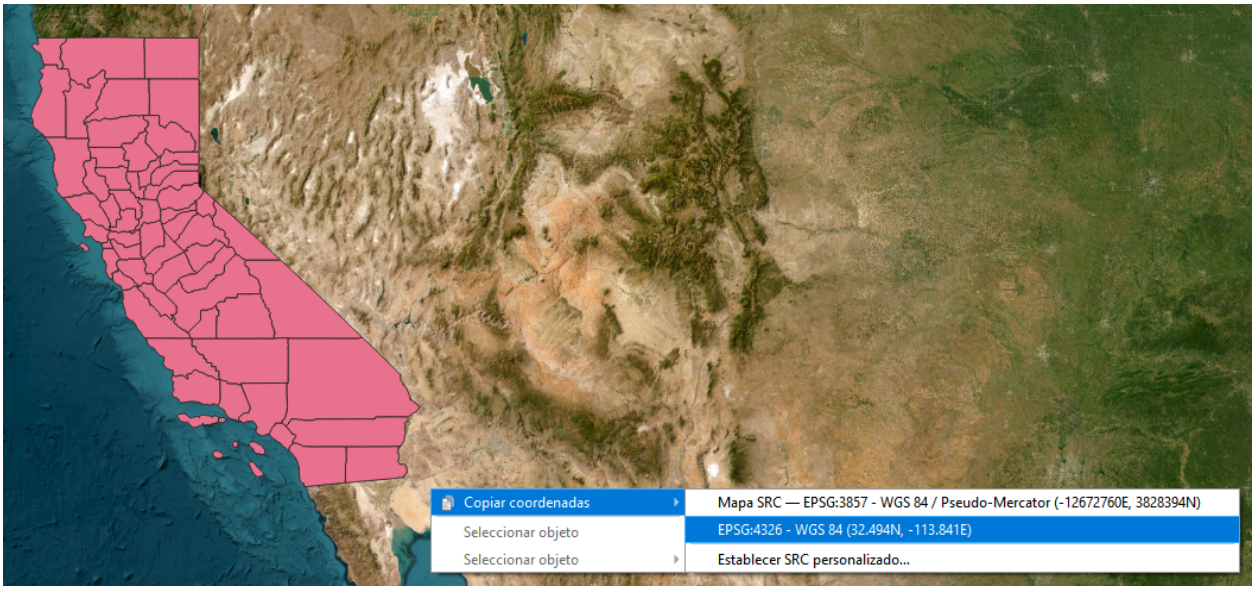

Figura 1.3: Obtención del `bbox` de California en QGIS

Se puede consultar el código empleado en el Apéndice A, en el bloque de Código 1.
Posteriormente, estos datos se han procesado para asignar cada estación a su condado correspondiente, como se ha descrito anteriormente.
Se proporciona una tabla con la distribución de estaciones por condado en California (se ha mostrado solo los 10 primeros condados ordenados por recuento) en la Tabla 1.1.

Cuadro 1.1: Recuento por condado

Condado	Recuento
Los Angeles	403
Alameda	190
Mono	94
Contra Costa	87
Sacramento	83
San Francisco	66
Monterey	50
Sonoma	48
San Diego	46
San Luis Obispo	44

1.1.2. Descarga de datos históricos

Ahora que tenemos los datos de las estaciones de interés, el siguiente paso es realizar una petición a la API de `OpenAQ` para descargar los datos históricos de calidad del aire.
La API devuelve información sobre los sensores de cada estación, que miden distintos contaminantes atmosféricos.
A partir de esto, se ha diseñado un bucle que itera sobre los 5333 sensores identificados en California, empleando el *endpoint* `/sensors` de la API.
Dado que la API impone límites en el número de peticiones por minuto y por hora, se ha implementado una lógica para monitorizar las cabeceras de respuesta `X-Rate-Limit-Remaining` y pausar las peticiones cuando sea necesario.
Este proceso ha tenido una naturaleza iterativa en cuanto al código, puesto que han surgido varios inconvenientes durante la descarga masiva de datos.
Sin embargo, puesto que se han guardado los datos descargados en archivos intermedios, no se ha perdido el progreso realizado hasta el momento.
Es por esta razón que la barra de progreso muestra un tiempo estimado menor del real.
Teniendo en cuenta los límites impuestos por la API, se estima que la descarga completa de los datos históricos de calidad del aire en California lleva entre 2 y 3 horas.
Se adjunta una captura de pantalla en la Figura 1.4 que ilustra el progreso de la descarga.

```python
# The API has a rate limit of 60 per minute and 2000 per hour.
from tqdm import tqdm
import time, requests

SENSOR_BASE = "https://api.openaq.org/v3/sensors"
aq_data_path = Path("data/openaq_data")
aq_data_path.mkdir(parents=True, exist_ok=True)
pbar = tqdm(total=selected_stations['sensors_count'].sum())
headers = {"X-API-Key": API_KEY}
for _, row in selected_stations.iterrows():
    station = row
    sensors_list = json.loads(row['sensors'].replace("'", '"'))
    for sensor in sensors_list:
        # check if sensor data file already exists
        sensor_filename = aq_data_path / "sensors" / f"sensor_{sensor['id']}.json"
        if sensor_filename.exists():
            pbar.update(1)
            continue
        sensor_id = sensor['id']
        r = requests.get(
            f"{SENSOR_BASE}/{sensor_id}/measurements",
            headers=headers,
            timeout=60,
            params={
                "date_from": "2015-01-01T00:00:00+00:00",
                "date_to": "2024-12-31T23:59:59+00:00",
            }
        )
        if r.headers.get('X-RateLimit-Remaining') == '0':
            reset_time = int(r.headers.get('X-RateLimit-Reset', 60))
            pbar.set_description(f"Rate limit reached. Sleeping for {reset_time + 1} seconds...")
            time.sleep(reset_time + 1)
            r = requests.get(
                f"{SENSOR_BASE}/{sensor_id}/measurements",
                headers=headers,
                timeout=60,
                params={
                    "date_from": "2015-01-01T00:00:00+00:00",
                    "date_to": "2024-12-31T23:59:59+00:00",
                }
            )
        sensor_data = r.json().get("results", {})
        # save sensor data to file
        sensor_file_path = aq_data_path / "sensors" / f"sensor_{sensor_id}.json"
        with open(sensor_file_path, 'w') as f:
            json.dump(sensor_data, f, indent=4)
        pbar.set_description(f"Saved sensor data to {sensor_file_path}")
        # append to metadata dataframe
        pbar.update(1)
        if r.headers.get('X-RateLimit-Remaining') == '0':
            reset_time = int(r.headers.get('X-RateLimit-Reset', 60))
            pbar.set_description(f"Rate limit reached. Sleeping for {reset_time + 1} seconds...")
            time.sleep(reset_time + 1)
pbar.close()
```

```
Saved sensor data to data/openaq_data/sensors/sensor_14949145.json: 100%|██████████| 5333/5333 [43:04<00:00,  2.06it/s]
```

Figura 1.4: Celda del cuaderno ilustrando las peticiones

Puesto que se han guardado los datos descargados en archivos JSON individuales, se ha optado por comprimirlos en un archivo ZIP para facilitar su manipulación posterior.
Además, puesto que no se van a modificar, facilita su almacenamiento y distribución.
El bucle se puede consultar en el cuaderno de Jupyter. No lo incluimos aquí por su extensión.
En la Figura 1.5 se muestra un ejemplo de la estructura de una respuesta en formato JSON, cuyo contenido ha sido truncado para mayor claridad.

```
[
    {
        "value": 9.0,
        "flagInfo": {
            "hasFlags": false
        },
        "parameter": {
            "id": 2,
            "name": "pm25",
            "units": "\u00b5g/m\u00b3",
            "displayName": null
        },
        "period": {
            "label": "raw",
            "interval": "01:00:00",
            "datetimeFrom": {
                "utc": "2016-03-06T19:00:00Z",
                "local": "2016-03-06T11:00:00-08:00"
            },
            "datetimeTo": {
                "utc": "2016-03-06T20:00:00Z",
                "local": "2016-03-06T12:00:00-08:00"
            }
        },
        ...
    }
]
```


Figura 1.5: Ejemplo de respuesta JSON de la API de `OpenAQ`

1.1.3. Procesamiento de los datos descargados

Con la idea de integrar los datos de calidad del aire con los datos de asma, se ha creado una base de datos SQLite para facilitar el análisis posterior.

Se ha diseñado un esquema de base de datos que incluye tablas para las estaciones de medición, los sensores, las mediciones realizadas ... entre otros.

En la Figura 1.6 se muestra un esquema generado por DBeaver de la base de datos.

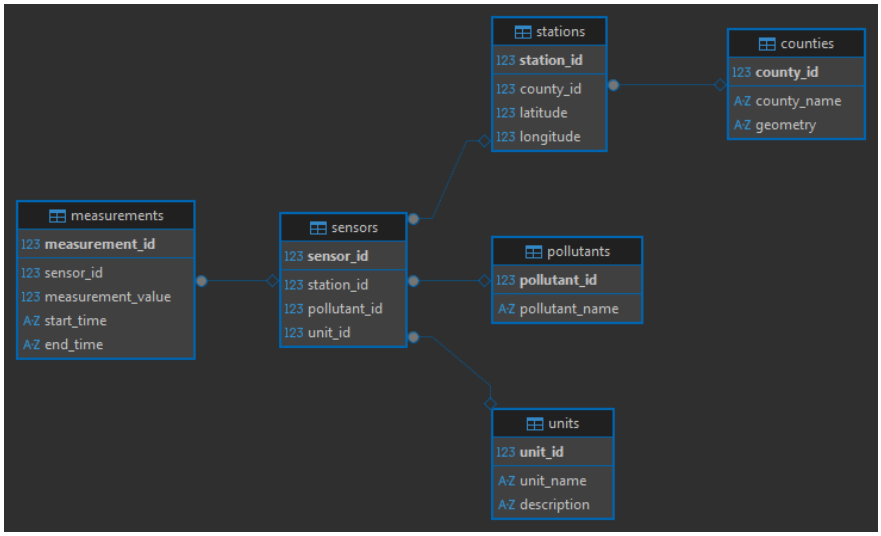

Figura 1.6: Esquema de la BD en la Fase I

Más adelante, la base de datos se ampliará para incluir los datos de asma, permitiendo realizar consultas complejas que relacionen ambos conjuntos de datos.

1.2. Datos de prevalencia de asma en California

Capítulo 2

Plan de Preservación

El presente Plan de Preservación tiene como objetivo garantizar la integridad, disponibilidad y reutilización a largo plazo de los activos digitales generados en el proyecto *Impacto de la Contaminación Atmosférica en los Casos de Asma en California*. Dado que el estudio maneja series temporales críticas y datos de salud pública, se ha diseñado una estrategia que cubre desde el almacenamiento físico hasta la gestión de derechos de acceso, alineándose con los principios FAIR (*Findable, Accessible, Interoperable, Reusable*).

2.1. Almacenamiento

La estrategia de almacenamiento se ha diseñado teniendo en cuenta el volumen de los datos geo-espaciales y la necesidad de diferenciar claramente entre los datos de uso frecuente y los archivos destinados a la preservación estática.

2.1.1. Almacenamiento físico

La infraestructura física local constituye el primer nivel de nuestra jerarquía de datos. Debido a la carga computacional que requieren las operaciones espaciales en **ArcGIS Pro** —especialmente el cruce de capas vectoriales de códigos postales (ZCTA) con las series temporales de contaminantes—, hemos priorizado el rendimiento en los equipos de trabajo.

Para el **almacenamiento activo**, utilizamos unidades de estado sólido (**SSD**) en las estaciones de trabajo locales. Esto nos permite evitar cuellos de botella durante la renderización de mapas y la ejecución de los scripts de limpieza en R. Adicionalmente, mantenemos un nivel de **almacenamiento en frío** (*cold storage*) mediante discos duros externos mecánicos (HDD) de alta capacidad. En estos dispositivos se custodian las copias originales de los *raw*

data descargados de la EPA y *catalog.data.gov*, liberando espacio en los discos de trabajo principales y sirviendo como repositorio local de los datos fuente inalterados.

2.2. Gestión de los datos

El presente estudio se adhiere a los principios FAIR (Findable, Accessible, Interoperable, Reusable) para garantizar la transparencia y la reproducibilidad de la investigación. La gestión de la información no se limita a la recolección, sino que abarca la totalidad del ciclo de vida de los datos, estableciendo procedimientos claros para su manejo desde la obtención inicial hasta su preservación a largo plazo.

2.2.1. Limpieza y curación

La transformación de los datos brutos en información analizable se ha llevado a cabo mediante un flujo de trabajo reproducible ejecutado en el entorno estadístico **R**. En lugar de correcciones manuales, que son propensas a errores y difíciles de auditar, hemos desarrollado scripts automatizados utilizando la librería `tidyverse` para estandarizar los formatos de fecha y unificar las unidades de medida de los contaminantes (NO_2, O_3) a partes por millón (ppm).

Para el tratamiento de inconsistencias, los scripts aplican reglas de validación lógica que identifican registros duplicados y valores fuera de rango. Específicamente, la detección de *outliers* en las lecturas de los sensores se realiza mediante el cálculo del rango intercuartílico (IQR), filtrando automáticamente aquellas mediciones que exceden el umbral de $1{,}5 \times IQR$ por considerarse errores instrumentales. Respecto a los valores nulos (NaNs) en las series temporales, cuando los huecos de información son inferiores a 4 horas, se imputan mediante algoritmos de interpolación lineal disponibles en el paquete `zoo`, garantizando la continuidad de la serie sin introducir sesgos significativos.

Por último, la integridad espacial de los datos se verifica utilizando **ArcGIS Pro**. Antes de realizar el cruce espacial entre las estaciones de monitoreo y los códigos postales (ZCTA), empleamos las herramientas de topología de ArcGIS para asegurar que las geometrías de los *shapefiles* no presentan superposiciones ni huecos que pudieran alterar la asignación de los niveles de exposición.

2.2.2. Metadatos

Para facilitar la recuperación e interpretación de los datos por terceros, se ha adoptado el esquema de metadatos estandarizado **Dublin Core**. La elección de este estándar internacional permite describir los recursos digitales mediante un conjunto de elementos esenciales —tales como Título, Creador, Cobertura Espacial, Fecha y Formato—, garantizando que el contexto de recolección y las características técnicas del *dataset* sean comprensibles sin ambigüedades para la comunidad científica.

Como parte fundamental de esta documentación, se adjunta un diccionario de datos o *codebook* detallado que define explícitamente cada variable, especifica las unidades de medida empleadas (ej. ppm o $\mu g/m^3$) y desglosa el significado de cualquier codificación utilizada en las tablas. Adicionalmente, para asegurar el orden, todos los archivos siguen una convención de nombrado sistemática (ej. `YYYYMMDD_Nombre_v01`) que facilita la identificación rápida de versiones.

2.2.3. Acceso y reutilización

Una vez finalizado el estudio, los datos procesados se depositarán en el repositorio de acceso abierto **Zenodo**, gestionado por el CERN. Esta plataforma asignará automáticamente un Identificador de Objeto Digital (DOI) único al conjunto de datos, asegurando que sea localizable y citable de manera permanente en futuras investigaciones.

Los datos se distribuirán bajo la licencia **Creative Commons Atribución 4.0 Internacional (CC-BY 4.0)**. Esta licencia abierta fomenta la reutilización, permitiendo a terceros compartir y adaptar el material para cualquier propósito, incluso comercial, con la única condición de reconocer la autoría original. En cuanto a las consideraciones éticas, al trabajar con datos de prevalencia agregados a nivel de condado y no con historias clínicas individuales, se garantiza la privacidad de los pacientes sin necesidad de aplicar técnicas adicionales de anonimización sobre los resultados finales.

2.3. Seguridad

La seguridad de la información en este proyecto se ha abordado desde una perspectiva integral que contempla tanto la integridad de los ficheros como la privacidad de los datos sensibles, aspectos críticos cuando se trabaja con información relacionada con la salud pública. Aunque los datos de prevalencia de asma utilizados son agregados y no contienen información per-

sonal identificable (PII) de pacientes individuales, hemos aplicado protocolos estrictos para evitar cualquier riesgo de reidentificación o manipulación malintencionada de los resultados.

En primer lugar, garantizamos la integridad de los datos mediante el uso de funciones de resumen o *hashing*. Cada vez que se descarga un conjunto de datos original de las fuentes gubernamentales o se genera un *dataset* consolidado tras la limpieza, se calcula su huella digital. Esto nos permite verificar periódicamente que los archivos almacenados en nuestros discos locales no han sufrido alteraciones silenciosas, conocidas como *bit rot*, ni modificaciones accidentales durante su manipulación. Si la suma de verificación actual no coincide con la registrada originalmente, el sistema nos alerta para restaurar una copia limpia desde el sistema de copias de seguridad.

Por otro lado, el control de acceso a los datos en fase de desarrollo se gestiona mediante permisos estrictos en el repositorio de código. Hemos establecido una política de roles donde únicamente los miembros del equipo de investigación tienen permisos de escritura y modificación sobre los *scripts* de análisis y los datos brutos. Cualquier cambio en el código que procesa los datos de contaminación o salud debe pasar por una revisión por pares antes de ser fusionado con la rama principal del proyecto. Esta trazabilidad completa nos asegura que ningún dato ha sido alterado de forma arbitraria para forzar una correlación estadística inexistente, manteniendo así la ética y la validez científica del estudio.

Finalmente, para el tránsito de información entre los equipos locales y los sistemas de almacenamiento remoto, utilizamos exclusivamente protocolos cifrados. Tanto la sincronización con la nube institucional como las operaciones de confirmación de cambios en el repositorio de código se realizan a través de canales seguros HTTPS y SSH, protegiendo la propiedad intelectual del proyecto y los datos de posibles interceptaciones en redes no seguras.

2.4. Estrategia de preservación a largo plazo

La preservación digital a largo plazo es un desafío que va más allá del simple almacenamiento de archivos; implica asegurar que la información permanezca legible, comprensible y ejecutable a medida que la tecnología evoluciona. Nuestra estrategia se fundamenta en la migración proactiva de formatos y la documentación exhaustiva, con el objetivo de que este estudio sobre el asma en California pueda ser reproducido con exactitud dentro de diez o veinte años, independientemente del software que exista en ese momento.
Una de las decisiones más importantes que hemos tomado para garantizar esta perdurabilidad

es la renuncia al uso de formatos propietarios para el archivo definitivo. Aunque durante la fase activa del proyecto utilizamos formatos nativos de Excel o archivos de proyecto de ArcGIS Pro por su eficiencia operativa, somos conscientes de que estos formatos podrían quedar obsoletos o requerir licencias de software costosas en el futuro. Por ello, nuestra política de preservación dicta que todo activo digital debe tener una versión equivalente en un estándar abierto:

- **Datos tabulares:** Se convierten a **CSV** con codificación UTF-8, asegurando su legibilidad por cualquier editor de texto básico.
- **Datos geográficos:** La cartografía vectorial se exporta a **GeoJSON**, un formato basado en texto y ampliamente soportado.
- **Documentación:** La memoria y los manuales se archivan en **PDF/A**, el estándar ISO diseñado específicamente para el archivo a largo plazo que incrusta todas las fuentes y elementos visuales necesarios.

2.4.1. Backup

Para mitigar el riesgo de pérdida catastrófica de datos, ya sea por fallos de hardware, errores humanos o ataques de software malicioso, hemos implementado una política de copias de seguridad rigurosa basada en el estándar de la industria conocido como la regla **3-2-1**.
Esta metodología asegura que no exista un único punto de fallo que pueda comprometer la totalidad del proyecto.
El protocolo establecido dicta que debemos mantener en todo momento al menos **tres** copias completas de todos los datos, almacenadas en **dos** soportes de diferente naturaleza, con **una** copia ubicada fuera de sitio (*off-site*):

1. **Copia de Trabajo:** Reside en las unidades SSD locales para el procesamiento diario.
2. **Copia de Seguridad Local (Air Gap):** Se realiza semanalmente en los discos duros externos (HDD). La característica clave de esta copia es su aislamiento: los discos se mantienen **desconectados física y lógicamente** de la red y de la corriente eléctrica cuando no se está realizando la copia. Esta técnica, conocida como "brecha de aire" (*air gap*), proporciona nuestra defensa más robusta contra el *ransomware*, ya que es físicamente imposible que un software malicioso encripte una unidad que no está conectada al sistema infectado.
3. **Copia Remota:** Se almacena en la nube institucional, asegurando la recuperación ante desastres físicos en el lugar de trabajo, como incendios o robos.

2.4.2. Almacenamiento en la nube

El almacenamiento en la nube juega un doble rol fundamental en nuestro proyecto: facilita la colaboración distribuida durante la fase activa y actúa como repositorio final para la preservación estática.

Durante el desarrollo, utilizamos **Google Drive/OneDrive** (vinculados a cuentas institucionales) y **GitHub** para la sincronización inmediata de documentos y código entre los miembros del equipo. Sin embargo, para la preservación a largo plazo, dependemos de **Zenodo**. La elección de este repositorio del CERN no es trivial: a diferencia de las nubes comerciales que requieren pagos recurrentes o mantenimiento de cuentas activas, Zenodo garantiza la preservación de los archivos científicos durante al menos 20 años, asegurando que la evidencia generada en este estudio permanezca accesible a la comunidad científica de manera perpetua.

Apéndice A

Código para la fase ETL

En este apéndice se presenta el código utilizado para la fase de Extracción, Transformación y Carga (ETL) de los datos en nuestro proyecto.
Las variables no definidas en los fragmentos de código se pueden consultar en la siguiente lista:

- `API_KEY`: Clave de API para acceder a la API de `OpenAQ`.
- `selected_stations`: DataFrame de pandas que contiene las estaciones seleccionadas para la descarga de datos históricos.

```python
# define bounding box for California
x_range = (-125, -113)
y_range = (32, 42.5)

import requests

BASE = "https://api.openaq.org/v3"
headers = {"X-API-Key": API_KEY}

all_results = []
page = 1
limit = 1000   # OpenAQ v3 maximum

while True:
    r = requests.get(
        f"{BASE}/locations",
        headers=headers,
        params={
            "bbox": f"{x_range[0]},{y_range[0]},{x_range[1]},{y_range[1]}",
            "limit": limit,
            "page": page,
            "country": "US",   # optional but helps validation
        },
        timeout=60,
    )
    r.raise_for_status()
    data = r.json()

    results = data.get("results", [])
    if not results:
        break

    all_results.extend(results)
    print(f"page {page}: {len(results)}")

    page += 1

print("TOTAL returned:", len(all_results))
print("first name:", all_results[0].get("name"))
```

Código 1: Petición a la API de `OpenAQ` dentro de un `bbox`

Bibliografía