GDPR Risk Pipeline

A reproducible project that fetches, processes, validates, and forecasts GDPR regulatory updates on an hourly basis using Apache Airflow, Python, and Prophet.

---

1. Project Overview

Purpose: Build an end-to-end pipeline that:

1. **Fetches** real-time GDPR/EDPB news
2. **Processes** raw JSON into clean time series
3. **Validates** both raw and processed data for sanity checks
4. **Forecasts** future update counts using Prophet
5. **Schedules** hourly runs via Airflow

Source: European Data Protection Board (EDPB) news — https://edpb.europa.eu/news/news_en

---

2. Repository Structure

```
gdpr-ccpa-risk-pipeline/
├── dags/
│   └── gdpr_ccpa_risk_pipeline.py
├── data/
│   ├── raw/                 # Raw JSON fetched from EDPB
│   │   └── edpb_news_<ts>.json
│   ├── processed/           # Cleaned policy counts
│   │   └── cleaned_policies.csv
│   └── forecasts/           # Prophet output CSVs
│       └── forecast_<ts>.csv
├── scripts/
│   ├── __init__.py
│   ├── fetch_policy_data.py
│   ├── process_policy_data.py
│   ├── validate_policy_data.py
│   └── forecast_policy_trends.py
└── validate_forecast.ipynb  # Notebook to inspect forecasts
```

---

3. Prerequisites

- **OS:** macOS or Linux
- **Python:** 3.8+ (with `venv`)
- **Airflow:** 2.7.x (in `venv`)
- **Libraries:** `prophet`, `requests`, `beautifulsoup4`, `pandas`, `lxml`

Install dependencies:

```
pip install --upgrade pip
pip install apache-airflow==2.7.1 prophet requests beautifulsoup4 pandas lxml
```

---

4. Scripts Detail

4.1 `scripts/fetch_policy_data.py`

- **New Source:** Scrapes EDPB news page (`news_en`) via BeautifulSoup
- Saves JSON array of `{title, link, date}` to `data/raw/edpb_news_<timestamp>.json`

4.2 `scripts/process_policy_data.py`

- Reads raw JSON files from `data/raw/` (latest file)
- Extracts `date` fields and counts daily occurrences
- Outputs `data/processed/cleaned_policies.csv` with columns `ds,date`, `y` counts

4.3 `scripts/validate_policy_data.py`

- Performs basic checks on raw JSON and processed CSV:
- Non-empty records
- Date parseability
- No negative counts

4.4 `scripts/forecast_policy_trends.py`

- Loads `cleaned_policies.csv` into Prophet DataFrame (`ds` , `y`)
- Fits model and predicts next 7 days (configurable)
- Saves `data/forecasts/forecast_<timestamp>.csv` with `ds,yhat,yhat_lower,yhat_upper`

---

5. Airflow DAG

File: `dags/gdpr_ccpa_risk_pipeline.py`

```
from airflow import DAG
from airflow.operators.python import PythonOperator
from datetime import datetime, timedelta

from scripts.fetch_policy_data import fetch_policy_data
from scripts.process_policy_data import process_policy_data
from scripts.validate_policy_data import validate_policy_data
from scripts.forecast_policy_trends import forecast_policy_trends

default_args = {...}

with DAG(
    "gdpr_ccpa_risk_pipeline",
    default_args=default_args,
    schedule_interval="@hourly",
    catchup=False,
) as dag:
    fetch    = PythonOperator(task_id="fetch_policy_data",
python_callable=fetch_policy_data)
    process  = PythonOperator(task_id="process_policy_data",
python_callable=process_policy_data)
    validate = PythonOperator(task_id="validate_policy_data",
python_callable=validate_policy_data)
    forecast = PythonOperator(task_id="forecast_policy_trends",
python_callable=lambda: forecast_policy_trends(periods=7))

    fetch >> process >> validate >> forecast
```

6. Downloading & Inspecting Input/Output

Run sequence locally to confirm files:

```
i.  python scripts/fetch_policy_data.py
ii. ls data/raw/edpb_news_*.json
iii. head data/raw/edpb_news_<ts>.json
iv. python scripts/process_policy_data.py
v. ls data/processed/cleaned_policies.csv
vi. head data/processed/cleaned_policies.csv
vii. python scripts/validate_policy_data.py
viii. python scripts/forecast_policy_trends.py
ix. ls data/forecasts/forecast_*.csv
x. head data/forecasts/forecast_<ts>.csv
```

7. Validation Notebook

File: `validate_forecast.ipynb`

A Jupyter notebook that:

- Loads raw and processed data
- Plots actual vs. forecasted trends
- Checks residuals and confidence intervals

Use it to visually inspect the plausibility of your model.
