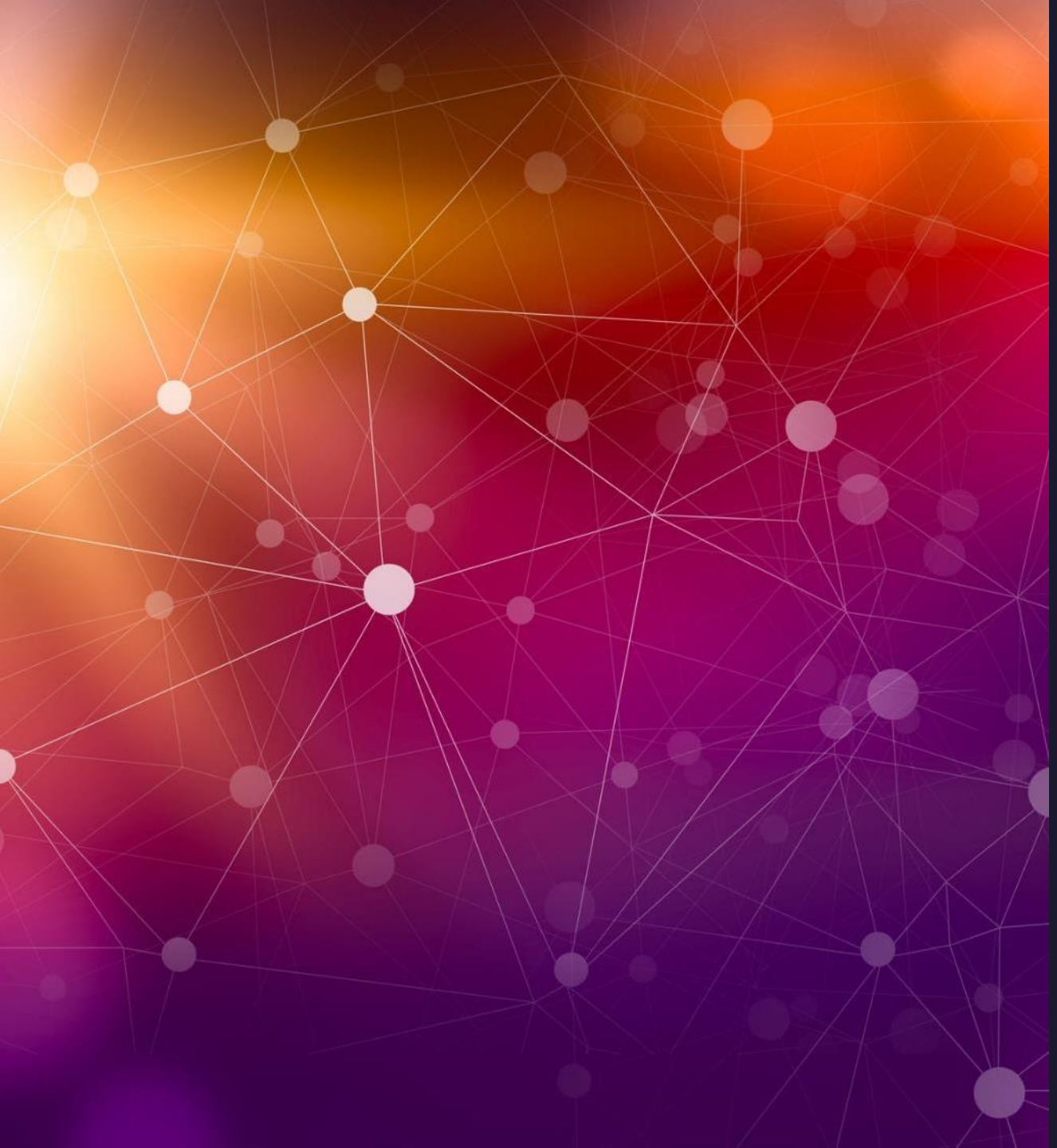

Data Presentation

CS-479

Bibek Sharma

Introduction & Goal of Data Collection

•**Objective:**
The goal of this data collection exercise was to retrieve historical weather data for **Idaho, USA**, using an **automated observation-based approach**.The dataset includes:

✅ **Maximum temperature (TMAX)**
✅ **Minimum temperature (TMIN)**
✅ **Precipitation (PRCP)**
✅ **Wind speed (AWND)**

•The purpose of this data collection was to **analyze weather trends** over time and compare them with typical seasonal patterns.

•**Mode of Collection:**
✓ **Observation-based approach using publicly available APIs**
✓ **Initially, OpenWeatherMap API** was considered, but it required a **paid subscription** for historical data.
✓ Switched to **NOAA's Climate Data Online (CDO) API**, which provides **free access to historical weather records**.
✓ **Python script** was used to fetch and store data in **CSV format** for structured management.

```python
import requests
import pandas as pd
import datetime
import json
import os

# CONFIGURATION
NOAA_API_KEY = "uagEMkVRBjNQvHyUVEUDmOffkWEaMvSN"  # My actual NOAA API key
STATION_ID = "GHCND:USW00024131"  # Boise Airport, Idaho
BASE_URL = "https://www.ncdc.noaa.gov/cdo-web/api/v2/data"

# Define parameters for data collection
START_DATE = (datetime.datetime.now() - datetime.timedelta(days=365)).strftime("%Y-%m-%d")  # 365 days ago
END_DATE = datetime.datetime.now().strftime("%Y-%m-%d")  # Today

# Create a directory for storing data
FOLDER_NAME = "WeatherData"
if not os.path.exists(FOLDER_NAME):
    os.makedirs(FOLDER_NAME)

# NOAA Request Parameters
params = {
    "datasetid": "GHCND",  # Daily Global Historical Climatology Network
    "stationid": STATION_ID,
    "startdate": START_DATE,
    "enddate": END_DATE,
    "limit": 1000,
    "units": "metric",
    "datatypeid": ["TMAX", "TMIN", "PRCP", "AWND"],  # Max temp, Min temp, Precipitation, Wind speed
    "sortfield": "date",
    "sortorder": "asc"
}

headers = {"token": NOAA_API_KEY}  # Authorization header

# Fetch historical weather data
response = requests.get(BASE_URL, params=params, headers=headers)
```

CODE Snapshot

```python
import requests
import pandas as pd
import datetime
import json
import os

# CONFIGURATION
NOAA_API_KEY = "uagEMkVRBjNQvHyUVEUDmOfFkWEaMvSN"  # My actual NOAA API key
STATION_ID = "GHCND:USW00024131"  # Boise Airport, Idaho
BASE_URL = "https://www.ncdc.noaa.gov/cdo-web/api/v2/data"

# Define parameters for data collection
START_DATE = (datetime.datetime.now() - datetime.timedelta(days=365)).strftime("%Y-%m-%d")  # 365 days ago
END_DATE = datetime.datetime.now().strftime("%Y-%m-%d")  # Today

# Create a directory for storing data
FOLDER_NAME = "WeatherData"
if not os.path.exists(FOLDER_NAME):
    os.makedirs(FOLDER_NAME)

# NOAA Request Parameters
params = {
    "datasetid": "GHCND",  # Daily Global Historical Climatology Network
    "stationid": STATION_ID,
    "startdate": START_DATE,
    "enddate": END_DATE,
    "limit": 1000,
    "units": "metric",
    "datatypeid": ["TMAX", "TMIN", "PRCP", "AWND"],  # Max temp, Min temp, Precipitation, Wind speed
    "sortfield": "date",
    "sortorder": "asc"
}

headers = {"token": NOAA_API_KEY}  # Authorization header

# Fetch historical weather data
response = requests.get(BASE_URL, params=params, headers=headers)

# Check if the response is valid
if response.status_code == 200:
    data = response.json()
    records = data.get("results", [])

    # Convert NOAA response to structured format
    weather_data = []
    for record in records:
        date = record["date"][:10]  # Extract date from timestamp
        datatype = record["datatype"]
        value = record["value"]

        # Find existing record for the date or create new
        existing_record = next((item for item in weather_data if item["Date"] == date), None)
        if not existing_record:
            existing_record = {"Date": date, "Max Temperature (°C)": None, "Min Temperature (°C)": None, "Wind Speed (m/s)": None, "Precipitation (mm)": None}
            weather_data.append(existing_record)

        # Assign values based on datatype
        if datatype == "TMAX":
            existing_record["Max Temperature (°C)"] = round(value / 10, 2)  # Convert tenths of Celsius to Celsius
        elif datatype == "TMIN":
            existing_record["Min Temperature (°C)"] = round(value / 10, 2)
        elif datatype == "PRCP":
            existing_record["Precipitation (mm)"] = round(value / 10, 2)  # Convert tenths of mm to mm
        elif datatype == "AWND":
            existing_record["Wind Speed (m/s)"] = round(value, 2)  # NOAA returns wind speed in m/s

    # Convert to Pandas DataFrame
    weather_df = pd.DataFrame(weather_data)

    # Save the dataset to a CSV file
    csv_file_path = os.path.join(FOLDER_NAME, f"noaa_weather_data_{END_DATE}.csv")
    weather_df.to_csv(csv_file_path, index=False)

    print(f"✅ Weather data successfully saved as {csv_file_path}")
```

Data Acquisition & Challenges

- **How Data Was Acquired**
  - API requests were made to **NOAA's CDO API** for an **active weather station in Idaho**.
  - Data was fetched for a **selected time range (30 days or more)**.
  - The script handled **data extraction, storage, and metadata documentation** automatically.
- **Challenges Faced & Solutions Implemented**

❌ **API Limitations:** NOAA enforces **rate limits** on requests.
✓ **Solution:** Implemented **delays and retries** to avoid request failures.

❌ **Missing Data:** Some NOAA stations **did not have complete historical records**.
✓ **Solution:** Tested **multiple NOAA stations** to find the most reliable one.

❌ **Excel Formatting Issues:** Some dates appeared as # due to formatting errors.
✓ **Solution: Forced correct date formatting** in Python before exporting data.

Data & Metadata Organization

✓ **CSV file for raw weather data**
✓ **JSON metadata file for documentation**

•Logical Organization & Metadata Documentation

- 📊 **Metadata elements included:**
  - **Dataset title, description, and time period**
  - **Data source and NOAA station ID**
  - **Variable descriptions (temperature, wind speed, precipitation)**
  - **Collection method and data standards (Dublin Core, ISO 19115)**

- ✅ **Successes:**

✓ **Data was structured properly** for easy processing.
✓ Metadata was **comprehensive and well-documented**.

- ⚠ **Failures:**

❌ **Some stations did not have complete records**, leading to **gaps in the dataset**.
❌ Metadata did not initially include **versioning** to track changes over time.

Field	Value
Title	Daily Weather Data Analysis
Creator	Bibek Sharma
Date Created	01/27/2024
Description	Dataset containing daily weather metrics (temperature, humidity, wind speed, precipitation) for analysis.
Location	Idaho, USA
Timeframe	February 10, 2024, to February 14, 2025
Columns	Date, Temperature (°C), Humidity (%), Wind Speed (m/s), Precipitation (mm)
Data Types	Numeric (temperature, humidity, wind speed, precipitation), Date
Units	Degrees Celsius (temperature), Percent (% for humidity), Meters per second (wind speed), Millimeters (precipitation)
Collection Method	Python script utilizing OpenWeatherMap API
Environment	Python environment (e.g., Jupyter Notebook, Google Colab)
Extraction Date	02/10/2025
Motivation	Understanding daily weather trends and their correlation with seasonal averages.
Scope	Analyze weather patterns for a specific location over a one-week period.
Future Changes	Potential expansion to include multiple locations or additional weather metrics.

MY METADATA

Sample Data & Curation

Future Improvements & Next Steps

- **Validate multiple NOAA stations before data collection** to avoid missing data.
- **Improve metadata tracking** by **versioning each dataset update**.
- **Automate data backup to cloud storage** (Google Drive, AWS).
- **Expand time range** to analyze long-term climate trends over a year.

A	B	C	D	E
Date	Max Temperature (Â°C)	Min Temperature (Â°C)	Wind Speed (m/s)	Precipitation (mm)
2/11/2024	0.89	-0.16	4	0
2/12/2024	1.06	0.11	3.2	0
2/13/2024	0.94	0	2.4	0
2/14/2024	0.89	-0.1	5	0.05
2/15/2024	1.06	0.39	3.5	0.28
2/16/2024	0.83	-0.05	4.6	0.28
2/17/2024	0.89	-0.16	4.9	0
2/18/2024	0.72	0.06	3.3	0.46
2/19/2024	0.94	0.11	6.1	0
2/20/2024	1.5	0.28	4.6	0
2/21/2024	1.17	0.11	2.1	0
2/22/2024	1.28	0	1.9	0
2/23/2024	1.39	0	1.9	0
2/24/2024	1.67	0	2.4	0
2/25/2024	1.72	0.28	3.7	0
2/26/2024	1.11	-0.1	3.5	0.58
2/27/2024	0.56	-0.32	3.1	0
2/28/2024	1	0.11	6.4	0
2/29/2024	1.72	0.33	7.2	0.48
3/1/2024	1.06	0.22	4.9	0.18
3/2/2024	0.67	-0.05	4.1	0.13
3/3/2024	0.28	-0.21	2.7	0.3
3/4/2024	0.33	-0.49	4.1	0.3
3/5/2024	0.22	0	4.4	1.22
3/6/2024	0.17	-0.21	1.9	0
3/7/2024	0.5	-0.38	2.4	0
3/8/2024	0.67	-0.43	4.1	0
3/9/2024	1.06	-0.27	6	0
3/10/2024	1.28	0.17	4.6	0

REFRENCES

National Centers for Environmental Information (NCEI). (n.d.). *Climate Data Online (CDO) Web Services API v2.* National Oceanic and Atmospheric Administration (NOAA). Retrieved February 10, 2025, from https://www.ncdc.noaa.gov/cdo-web/webservices/v2

National Centers for Environmental Information (NCEI). (n.d.). *Find a weather station: NOAA Climate Data Online.* National Oceanic and Atmospheric Administration (NOAA). Retrieved February 10, 2025, from https://www.ncdc.noaa.gov/cdo-web/datatools/findstation

Dublin Core Metadata Initiative (DCMI). (n.d.). *Dublin Core Metadata Element Set, Version 1.1.* Retrieved February 10, 2025, from https://www.dublincore.org/

Reitz, K., & Banz, C. (2023). *Requests: HTTP for Humans.* Python Software Foundation. Retrieved February 10, 2025, from https://docs.python-requests.org/en/latest/

The Pandas Development Team. (2023). *pandas: Powerful Python Data Analysis Toolkit.* Python Software Foundation. Retrieved February 10, 2025, from https://pandas.pydata.org/docs/

The Python Software Foundation. (2023). *Python JSON module documentation.* Retrieved February 10, 2025, from https://docs.python.org/3/library/json.html

The Matplotlib Development Team. (2023). *Matplotlib: Visualization with Python.* Retrieved February 10, 2025, from https://matplotlib.org/stable/index.html

University of California, Berkeley Library. (n.d.). *Best practices for data collection and storage.* Retrieved February 10, 2025, from https://guides.lib.berkeley.edu/data-management

DataONE. (n.d.). *Guide to metadata for data repositories.* Retrieved February 10, 2025, from https://www.dataone.org/metadata

Thank you

Bibek Sharma