

# NOAA GHCN Project Using Spark Medallion Architecture

*DAT535 Final Project Report*

Supervisor: Tomasz Wiktorski

B M Nafis Fuad  
*University of Stavanger*  
bm.fuad@stud.uis.no

Benjamin Birkelid  
*University of Stavanger*  
b.birkelid@stud.uis.no

**Abstract**—This project builds an end-to-end data pipeline for the NOAA GHCN-Daily climate dataset using the Spark Medallion Architecture (Bronze–Silver–Gold). Raw observations are ingested into Bronze, cleaned and standardized into daily station-level records in Silver, and transformed in Gold into monthly/yearly aggregates, climate normals, and temperature/precipitation anomalies for Norway. The Gold layer also supports machine-learning tasks, including forecasting, rainfall prediction, anomaly regression, climate-zone clustering, and heatwave classification. A final analysis notebook visualizes key climate trends, revealing clear warming patterns and regional climate differences. This demonstrates a scalable, reproducible Spark-based approach to climate data engineering and analytics.

**Index Terms**—Spark, Medallion Architecture, Data Engineering, Climate Analytics, Big Data Processing

GitHub link to the project source code:  
<https://github.com/nafis4139/NOAA-GHCN-Project-Using-Spark-Medallion-Architecture>

## I. INTRODUCTION

Climate data is essential for understanding long-term environmental variability, extreme weather behaviour, and the impacts of climate change. As global temperatures rise and precipitation patterns shift, large-scale climate datasets have become increasingly important for scientific analysis, policy development, and forecasting applications. One of the most widely used open datasets for such purposes is the National Oceanic and Atmospheric Administration (NOAA) Global Historical Climatology Network Daily (GHCN-Daily) dataset [1], which contains decades of daily observations from thousands of meteorological stations worldwide. These observations include maximum and minimum temperature, precipitation, and other meteorological variables.

Despite its value, the GHCN-Daily dataset presents several challenges: it is large, heterogeneous, text-based, and includes inconsistencies, missing values, unit variations, and variable data quality [2]. These issues make direct analysis difficult without a structured and scalable data engineering approach. To address these challenges, this project implements the Medallion Architecture, a layered data processing design

widely used in modern data lakehouse systems, using Apache Spark [3], [7], [9]. The Bronze–Silver–Gold pipeline provides a reproducible, modular, and efficient framework for transforming raw climate observations into high-quality analytical datasets [3], [10].

In this architecture, the Bronze layer ingests and stores raw GHCN-Daily records while preserving original structure and metadata. The Silver layer performs cleaning, quality filtering, unit conversion, and restructuring into daily station-level records suitable for downstream analytics. The Gold layer aggregates the cleaned data into monthly and yearly summaries, computes climate normals, derives temperature and precipitation anomalies, and generates datasets for machine learning applications. An additional Analysis layer provides visualisation of key outputs, including temperature trends, anomaly patterns, station-level behaviour, and model predictions.

This work focuses on climate observations from Norway and demonstrates how Spark-based Medallion Architecture can support scalable, end-to-end climate data engineering, analytics, and machine learning. The resulting pipeline enables efficient processing of historical climate data and provides insights into regional climate behaviour, long-term warming trends, and extreme event risks.

## II. BACKGROUND

### A. Dataset Overview: NOAA GHCN-Daily

Large-scale climate datasets are essential for analysing historical weather patterns, understanding regional climate variability, and assessing long-term climate change. Among the most widely used sources is the Global Historical Climatology Network Daily (GHCN-Daily) dataset provided by the National Oceanic and Atmospheric Administration (NOAA). GHCN-Daily compiles millions of daily climate observations from over 100,000 stations worldwide, including air temperature, precipitation, and snowfall measurements [2]. Each record includes associated quality flags, station identifiers, and metadata describing geographic coordinates, elevation, and

observational coverage. Although comprehensive, the GHCN-Daily dataset presents several challenges when used in analytical or machine learning workflows:

- The raw data is stored in large text-based files, often hundreds of megabytes
- Contains inconsistencies such as missing values, corrupted lines, and duplicate entries
- Requires unit conversion (e.g., temperature  $\times 0.1^{\circ}\text{C}$ , precipitation  $\times 0.1 \text{ mm}$ )
- Different stations display varying levels of completeness, making raw ingestion unsuitable for direct analysis

Because of this, GHCN-Daily cannot be used directly without a structured processing pipeline. These obstacles necessitate a robust, scalable data processing workflow capable of cleaning, validating, and enriching the dataset before meaningful insights can be extracted.

#### B. Medallion Architecture (Bronze – Silver – Gold)

Modern data lakehouse systems adopt the Medallion Architecture, a layered design that organises data into *Bronze*, *Silver*, and *Gold* tiers. This pattern provides a systematic framework for transforming raw data into trusted analytical assets.

The Bronze layer acts as the landing zone, capturing raw ingested files with minimal transformation while preserving lineage and ingestion metadata.

The Silver layer focuses on cleaning and standardisation, removing invalid values, applying quality checks, performing unit conversions, and reshaping records into consistent daily station-level facts.

Finally, the Gold layer produces high-value datasets such as aggregated statistics, climate normals, anomalies, and machine learning features.

Apache Spark is well suited for implementing this architecture due to its distributed computing capabilities and support for large-scale data processing [7]. Spark efficiently handles multi-gigabyte climate datasets, supports complex transformations via its DataFrame and RDD APIs [6], [8], and integrates seamlessly with machine learning algorithms through Spark MLlib [7]. By combining Spark with the Medallion Architecture, this project establishes a scalable and reproducible workflow for transforming NOAA GHCN-Daily observations into actionable climate insights [3], [9], [10].

While the architecture supports global ingestion, this project focuses specifically on Norwegian climate stations. Norway is climatically diverse-with coastal mild regions, cold inland valleys, and mountainous areas-which makes it an ideal case study for climate anomaly analysis, regional trend detection, climate zone clustering and machine learning applications. Insights derived from Norway in this project can be extended to larger-scale global studies in future work.

### III. METHODOLOGY

The methodology for this project follows the Medallion Architecture, consisting of three core data-processing layers-Bronze, Silver, and Gold-implemented entirely in Apache Spark using PySpark. Each layer transforms the NOAA

GHCN-Daily dataset into progressively refined and analytics-ready forms. A final Analysis Layer is used for visualization and interpretation. The methodology is described below in details.

#### A. Bronze Layer – Raw Data Ingestion

The Bronze layer serves as the ingestion zone, where raw NOAA GHCN-Daily observations are collected and stored with minimal transformation to preserve data fidelity.

*1) Raw File Ingestion:* The dataset is ingested from a consolidated NOAA text file (`ghcn_all_years.txt`) containing all station observations across multiple years. Spark's `read.text()` function is employed to efficiently load the file, enabling distributed processing of millions of records. After loading the dataset, an initial count revealed a total of **587,797,263 raw lines**, confirming the large-scale nature of the ingestion process.

Listing 1. Bronze: Read combined raw text

```
input_path = os.path.join(RAW_DIR,
    "ghcn_all_years.txt")
if not os.path.isfile(input_path):
    raise FileNotFoundError(f"Expected
        combined TXT file at {input_path}.
        Run the download cell to generate
        it.")
df_text = spark.read.text(input_path)
```

#### Dataset Overview

- **Raw line count:** 587,797,263 (Bronze Layer)

*2) Safe Parsing & RDD Transformation:* The raw ingested text file was converted into a structured schema using a custom safe-parsing function. Each raw line was split into fields such as station ID, date, element type, raw measurement, and quality flags. To ensure no information was lost, a robust (`parse_line_safe()`) function was implemented: valid lines were parsed normally with (`_status = "valid"`), while corrupted or incomplete lines were retained with (`_status = "parse_error"`) and their full raw content stored in `_raw_data`. This design preserves the entire input dataset while keeping track of ingestion quality.

Before conversion to a DataFrame, the raw text input was transformed into an RDD. This allowed the safe-parsing function to be applied record-by-record through a distributed `map()` transformation. By processing the dataset as an RDD first, the pipeline ensured full control over line-level parsing logic and fault tolerance at massive scale, which is not as easily achievable with DataFrame-only ingestion. The parsed RDD was then materialised back into a Spark DataFrame using the predefined Bronze schema.

For valid records, the following fields are extracted:

- Station ID (station)
- Date string (YYYYMMDD) (date\_str)
- Element type (TMAX, TMIN, PRCP) (element)
- Raw measurement value (raw\_value)
- Metadata flags (mflag, qflag, sflag)
- Observation time (obstime)
- Derived year (year)

Each row also received an ingestion timestamp and source tag for traceability. So, all the added metadata fields are:

- Ingestion Time (\_ingestion\_timestamp)
- Source File (\_source)
- Parse Status (\_status)
- Raw Content (\_raw\_data)

The resulting Bronze table was written as a Spark DataFrame with **587,797,263** records, which confirms Spark successfully parsed everything.

**Listing 2. Bronze: Safe parser with metadata**

```
bronze_schema = StructType([
    StructField("station", StringType(), True),
    StructField("date_str", StringType(), True),
    StructField("element", StringType(), True),
    StructField("raw_value", StringType(), True),
    StructField("mflag", StringType(), True),
    StructField("qflag", StringType(), True),
    StructField("sflag", StringType(), True),
    StructField("obstime", StringType(), True),
    StructField("year", StringType(), True),
    StructField("_ingestion_timestamp",
               DoubleType(), False),
    StructField("_source", StringType(), False),
    StructField("_status", StringType(), False),
    StructField("_raw_data", StringType(), True),
])
]

SOURCE_TAG = "ghcn_txt"

def parse_line_safe(line: str):
    ts = time.time()
    try:
        parts = line.split(",")
        if len(parts) < 8:
            raise ValueError("not enough columns")
        station, date_str, element = parts[0],
            parts[1], parts[2]
        raw_value, mflag, qflag = parts[3],
            parts[4], parts[5]
        sflag, obstime = parts[6], parts[7]
        year = (date_str[:4] if date_str and
            len(date_str) >= 4 else None)
        return Row(station=station,
                  date_str=date_str, element=element,
                  raw_value=raw_value, mflag=mflag,
                  qflag=qflag,
                  sflag=sflag, obstime=obstime, year=year,
                  _ingestion_timestamp=ts, _source=SOURCE_TAG,
                  _status="valid", _raw_data=None)
    except Exception:
        return Row(station=None, date_str=None, ...,
                  _ingestion_timestamp=ts, _source=SOURCE_TAG,
                  _status="parse_error", _raw_data=line)
```

**3) Output Structure:** The processed Bronze data is stored in Parquet format and partitioned by year to support efficient downstream filtering and querying. The output preserves:

- all valid observations,
- all invalid/partially parsed observations,
- complete ingestion metadata.

This structure ensures that the Bronze layer remains fully auditable and serves as a reliable foundation for subsequent Silver and Gold transformations.

#### B. Silver Layer – Cleaning & Filtering

The Silver layer transforms raw Bronze records into validated, standardized, and aggregated daily climate facts suitable for downstream analytics and Gold-level modeling. Because the GHCN-Daily dataset contains heterogeneous variables, quality flags, and sentinel values across hundreds of millions of records, the Silver layer employs **RDD-based processing** [6] to enable fine-grained row-level control while preserving scalability in Apache Spark [7]. The result is a cleaned and analysis-ready dataset with approximately 186 million daily station-level observations.

#### Dataset Overview

- **Raw Bronze Row:** 587,797,263
- **Cleaned Silver Daily Row:** 186,418,577

**1) Input and Scope:** The Silver layer reads all Bronze Parquet partitions and restricts the processing window to the relevant study period:

- Input: data/bronze
- Years processed: 2010–2025

The dataset is initially loaded as a DataFrame, but is immediately converted to an RDD to support custom and tightly controlled cleaning operations.

**2) Switching to RDD Processing:** While DataFrames offer high-level optimizations, the cleaning logic required for GHCN-Daily custom-parsing, multi-stage validation, and record-level filtering is more naturally expressed in RDD form. Converting the Bronze table via `df_bronze.rdd` allows:

- full control over row-level operations,
- custom parsing and validation logic,
- more flexible filtering than DataFrame SQL expressions,
- precise manipulation of complex records across millions of observations.

Each Bronze row is first mapped into a simplified Python dictionary using the `to_rec()` function, extracting only the fields necessary for climate analysis.

```

Listing 3. Silver: Map Bronze rows to minimal records
def to_int_safe(x):
    try:
        return int(x)
    except:
        return None

def to_rec(row) -> Dict[str, Any]:
    d = row.asDict()
    return {
        "id": d.get("station"),
        "date": d.get("date_str"),
        "element": d.get("element"),
        "value": to_int_safe(d.get("raw_value")),
        "qflag": (d.get("qflag") or ""),
        "year": d.get("year"),
        "status": (d.get("_status") or "")
    }

# Map to simple dicts
rdd1 = rdd_raw.map(to_rec)

```

- Station ID (id)
- Date (date)
- Element type (element)
- Raw Integer Measurement (value)
- Quality Flag (qflag)
- Derived Year (year)
- Bronze parsing outcome (status)

This simplified structure forms the foundation for subsequent cleaning, unit conversion, deduplication, and aggregation in the Silver layer.

*3) RDD Cleaning Pipeline:* The Silver layer applies a multi-stage cleaning pipeline to remove invalid, unreliable, or irrelevant observations:

*a) Quality flag check:* Keep only observations that passed NOAA's quality checks which is an empty quality flag (`qflag == ""`).

*b) Bronze Parsing Validity:* Keep only records with  $\text{status} \in \{"", "valid"\}$ .

*c) Measurement Validity:* Remove sentinel values such as `-9999`, which indicate missing NOAA measurements.

*d) Element Restrictions:* Limit the dataset to the three core climate variables: `TMAX`, `TMIN`, and `PRCP`.

*e) Deduplication:* Deduplicate using the composite key (`station`, `date`, `element`). This guarantees that, for any given station-day-element combination, only the first valid observation is retained, preventing duplicate measurements from influencing downstream analysis.

*f) Unit Conversion:* Convert temperature and precipitation measurements into standard SI units. These transformations produce a standardized numerical field (`value_si`) suitable for climatological aggregation and statistical analysis.

- **TMAX / TMIN:** tenths of degrees → degrees Celsius (divide by 10),
- **PRCP:** tenths of millimeters → millimeters (divide by 10).

## Silver Cleaning Rules

- Keep only quality-passed rows: `qflag == ""`.
- Keep valid parse outcomes: `status ∈ {"", "valid"}`.
- Remove missing/sentinel values: `value != -9999`.
- Restrict to core variables: **TMAX**, **TMIN**, **PRCP**.
- Deduplicate by (`id`, `date`, `element`).
- Convert to SI units: `value_si = value/10.0`.

After applying these filters, the dataset is reduced from **587,797,263** raw observations to **321,942,526** high-quality records, meaning that only **54.8%** of the initial dataset is retained. This reduction is expected, as GHCN-Daily contains many variables irrelevant to this project, as well as a substantial number of missing or quality-failed measurements.

**Listing 4. Silver: Apply filters + unit conversion**

```

rdd_clean = (
    rdd1
    .filter(lambda r: r["qflag"] == "")
    .filter(lambda r: (r["status"] in ("",
                                         "valid")))
    .filter(lambda r: (r["value"] is not None
                      and r["value"] != -9999))
    .filter(lambda r: r["element"] in
            ELEMENTS)
)
# Deduplicate (id, date, element)
rdd_unique = (
    rdd_clean
    .map(lambda r: ((r["id"], r["date"],
                     r["element"]), r))
    .reduceByKey(lambda a, b: a)
    .map(lambda kv: kv[1])
)
rdd_si = rdd_unique.map(lambda r: {**r,
                                     "value_si": r["value"] / 10.0})

```

## 4) Constructing Daily Station-Level Records:

GHCN-Daily stores each climate variable as a separate row. Therefore, the Silver layer reconstructs full daily observations by grouping measurements by station id, date and year. Within each group, the element-value pairs are merged into a single structured daily record containing:

- `tmax`
- `tmin`
- `tavg` - computed when both `TMAX` and `TMIN` are present
- `prcp`

`tavg` is computed when both `tmax` and `tmin` are available. If one of the value is missing, it will set as `None`.

$$TAVG = \frac{TMAX + TMIN}{2}$$

This transformation reduces the dataset from approximately **321 million** element-level rows to **186,418,577** station-day records. The resulting structure is:

(`id`, `date`, `year`, `tmax`, `tmin`, `tavg`, `prcp`)

5) *Final Structuring*: The structured daily tuples are converted back into a Spark DataFrame using an explicitly defined schema. The month and day fields are derived via Spark date functions for downstream aggregations and analysis.

**Listing 5. Silver: Final Structuring**

```
# Defined Schema
schema = T.StructType([
    T.StructField("id", T.StringType(), False),
    T.StructField("yyyymmdd", T.StringType(), False),
    T.StructField("year", T.IntegerType(), True),
    T.StructField("tmax_c", T.DoubleType(), True),
    T.StructField("tmin_c", T.DoubleType(), True),
    T.StructField("tavg_c", T.DoubleType(), True),
    T.StructField("prcp_mm", T.DoubleType(), True),
])
df_silver = spark.createDataFrame(rdd_daily, schema)

# Parse date & add convenience fields
df_silver = (df_silver
    .withColumn("date",
        F.to_date(F.col("yyyymmdd"),
            "yyyyMdd"))
    .withColumn("month", F.month("date"))
    .withColumn("day", F.dayofmonth("date"))
    .drop("yyyymmdd")
)
```

6) *Output Structure*: The cleaned daily facts are written as Parquet, partitioned by year:

- Output: data/silver
- Format: Parquet with `partitionBy(year)`

A validation pass confirms that partitions for 2010–2025 exist, schemas are consistent, and the expected 185.9 million daily rows are produced.

7) *Auxiliary Metadata Tables*: In addition to daily climate facts, the Silver layer processes NOAA's fixed-width metadata files into structured Parquet tables under `data/silver_meta`.

a) *Stations Metadata*: A fixed-width parser extracts (i) Station ID, (ii) Latitude & longitude, (iii) Elevation and (iv) State and Station name. Additional cleaning ensures valid coordinate ranges, removes NaNs, and normalises text fields.

b) *Inventory Metadata*: This file includes (i) Station ID, (ii) Supported climate elements, (iii) First and last year of observations. After parsing, the dataset is filtered to essential elements: TMAX, TMIN, PRCP, TAVG, SNOW, SNWD.

A coverage table was generated summarizing, for each station:

- First and last year of available observations,
- Availability of TMAX, TMIN, and PRCP elements.

These metadata assets are later integrated in the Gold layer to attach geographic and temporal context to the cleaned Silver dataset.

### C. Gold Layer – Aggregation, Climate Normals, Anomalies, and ML Features

The Gold layer represents the analytical and modelling stage of the Medallion Architecture. While the Bronze and Silver layers focus on raw ingestion and data quality improvements, the Gold layer transforms the cleaned daily climate records into higher-level climatological products. These include station-level and regional summaries, long-term climate normals, temperature and precipitation anomalies, and several machine learning models for forecasting and classification. All computations are implemented using Apache Spark to ensure scalability across millions of observations.

1) *Loading Daily Silver Data and Metadata*: The Gold pipeline begins by loading the cleaned Silver daily station-level dataset (2010–2025) and enriching it with auxiliary metadata from the `silver_meta` tables:

- `stations.parquet` - station coordinates, elevation, and names,
- `inventory.parquet` - available climate variables per station,
- `coverage.parquet` - first and last year of valid observations.

A country identifier is derived from the station ID prefix, enabling regional filtering. To focus on regional climate behaviour, the dataset is filtered to include only Norwegian stations (NO), reducing the input to **1,787,937** daily observations.

#### Dataset Overview

- **Input Silver Data:** 186,418,577
- **Filtered Data (Only Norway):** 1,787,937

2) *Monthly Station-Level Aggregation*: Daily observations are aggregated to the station-month level. For each station and month, the following metrics are computed:

- mean TMAX, TMIN, and TAVG,
- total monthly precipitation,
- number of wet days ( $PRCP > 0$ ),
- count of days with valid observations for each variable.

To ensure climatological reliability, completeness flags are generated:

- `is_complete_temp` = True if  $\geq 20$  valid temperature days are present,
- `is_complete_prcp` = True if  $\geq 20$  valid precipitation days are present.

This monthly aggregation step produces a total of **59,100** station-month records, which are written to the Gold table:

`gold/station_monthly/`

3) *Station-Level Yearly Climate Aggregations*: Monthly station data is further aggregated to the station-year level. For each station and year, the following metrics are computed:

- annual mean TMAX, TMIN, and TAVG,
- annual precipitation total,
- total number of wet days,
- number of months that satisfy the completeness criteria.

Years containing at least **10 complete months** are classified as climatologically reliable.

The resulting yearly dataset is written to:

```
gold/station_yearly/
```

This table serves as the foundation for computing climate normals and temperature and precipitation anomalies.

**4) Climate Normals (2010–2020):** Climate normals represent average climate conditions over a long, stable reference period. Although NOAA typically uses the 1991–2020 climatological baseline [4], [5], the present dataset contains full coverage only for 2010–2020. Therefore, this 11-year period is used to compute:

- normal monthly TMAX, TMIN, and TAVG,
- normal monthly precipitation totals.

Normals are computed for each month  $m$  using:

$$\text{normal}(m) = \frac{1}{N} \sum_{y=2010}^{2020} x_{m,y}$$

where  $x_{m,y}$  is the observed monthly value for month  $m$  in year  $y$ , and  $N = 11$  is the number of years in the reference period.

The resulting climate normals are stored in:

```
gold/normal_1991_2020/
```

These normals serve as the climatological baseline for subsequent anomaly calculations.

**5) Monthly and Yearly Climate Anomalies:** Climate anomalies quantify deviations from long-term normal conditions. Monthly anomalies are computed by joining the station\_monthly table with the climate normals on (station\_id, month) and calculating the following metrics.

*a) Temperature Anomalies:* For each month, the temperature anomaly is defined as:

$$\text{tavg_anom} = \text{tavg_mean} - \text{normal_tavg}$$

with analogous formulas for TMAX and TMIN:

$$\text{tmax_anom} = \text{tmax_mean} - \text{normal_tmax}$$

$$\text{tmin_anom} = \text{tmin_mean} - \text{normal_tmin}$$

*b) Precipitation Ratios:* Precipitation anomalies are expressed as a ratio:

$$\text{prec_ratio} = \frac{\text{monthly_prestipitation}}{\text{normal_prestipitation}}$$

Monthly anomaly outputs are written to:

```
gold/anomalies_monthly/
```

*c) Yearly Anomaly Aggregation:* Monthly anomalies are then aggregated to produce station-year anomaly metrics:

- mean annual temperature anomaly,
- mean annual precipitation ratio.

These yearly summaries are stored in:

```
gold/anomalies_yearly/
```

**6) Regional Climate Aggregations (Norway):** To analyze climate behaviour at the national scale, station-level anomalies from all Norwegian stations are aggregated along two dimensions:

- **year and month** → regional monthly anomalies,
- **year** → regional yearly anomalies.

The resulting regional metrics include:

- mean temperature anomaly,
- mean TMAX and TMIN anomaly,
- mean precipitation ratio,
- number of contributing stations.

The outputs are stored in:

```
gold/region_monthly/
gold/region_yearly/
```

**7) Machine Learning Models:** The Gold layer incorporates several machine-learning models to generate climate predictions, detect patterns, and classify extreme events. All models are implemented using Spark MLlib.

*a) Linear Regression – National Temperature Trend Forecast:* This model predicts Norway's long-term temperature anomaly trend using the regional yearly anomaly dataset.

#### Inputs:

- region\_yearly temperature anomalies
- Feature: year\_centered = year – 2010
- Target: yearly mean temperature anomaly

#### Model results:

- Coefficient  $\approx +0.0886^{\circ}\text{C}$  per year
- Positive coefficient indicates a warming trend
- $R^2 \approx 0.29$  (moderate explanatory power)

#### Forecast for 2026–2030:

Year	Predicted Tavg Anomaly ( $^{\circ}\text{C}$ )
2026	0.910
2027	0.999
2028	1.088
2029	1.176
2030	1.265

Outputs stored in:

```
gold/ml_lr_region_forecast/
```

*b) Random Forest – Predicting Station Annual Rainfall:* This model predicts annual precipitation totals for each station.

#### Features:

- Latitude, longitude, elevation
- Annual mean TMAX, TMIN, TAVG
- Year

**Training size:** 1,204 station-year observations.

The Random Forest model captures spatial precipitation gradients effectively (e.g., wetter coastal regions vs. drier inland regions), though extreme wet stations exhibit minor prediction bias.

Results stored in:

```
gold/ml_rf_station_prcp/
```

*c) Gradient-Boosted Trees – Predicting Annual Temperature Anomalies:* This model predicts annual temperature anomalies for each station.

#### Features:

- Year
- Station latitude and longitude
- Elevation

#### Model findings:

- Strong positive anomalies in years such as 2014, 2020, and 2024
- Clear spatial gradient influenced by latitude and elevation

Stored in:

```
gold/ml_gbt_station_tanom/
```

#### *d) K-Means Clustering – Climate Zones of Norway:*

Using long-term station normals, K-Means clustering ( $k = 4$ ) identifies distinct Norwegian climate zones.

#### Clustering features:

- Latitude, longitude
- Annual normal temperature
- Annual normal precipitation

#### Cluster interpretation:

- **Cluster 0:** Arctic & Inland Cold
- **Cluster 1:** Central Norway Mixed
- **Cluster 2:** West Coast Wet Maritime
- **Cluster 3:** South/East Mild Climate

Results stored in:

```
gold/ml_kmeans_clusters/
```

#### *e) Logistic Regression – Heatwave Year Classification:*

A heatwave year is defined as one where the mean summer (JJA) temperature anomaly exceeds  $2^{\circ}\text{C}$  [11].

#### Features:

- Summer temperature anomaly
- Summer precipitation ratio
- Latitude and longitude

The classifier predicts whether a station–year qualifies as a heatwave year. Because most Norwegian summers do not reach  $+2^{\circ}\text{C}$  anomaly, the majority of classifications are non-heatwave (0).

Stored in:

```
gold/ml_logr_heatwave/
```

The Gold layer converts the cleaned daily data into monthly and yearly summaries, climate normals, and station- and national-scale anomalies that clearly reflect Norway’s recent climate variability and warming trends. It also enables predictive modelling through machine learning tasks such as temperature forecasting, rainfall estimation, climate zone clustering, and heatwave classification. Overall, the Gold layer demonstrates how the Medallion Architecture supports scalable and data-driven climate analysis.

## IV. VISUALIZATION AND ANALYSIS

The end-to-end processing of the GHCN-Daily dataset through the Medallion Architecture produced a rich set of

climatological products for Norway covering 2010–2025. The results illustrate both the effectiveness of the data engineering pipeline and the climate patterns embedded in the dataset. This section summarises the main findings from the monthly and yearly aggregations, anomaly calculations, regional metrics, and the machine learning analyses.

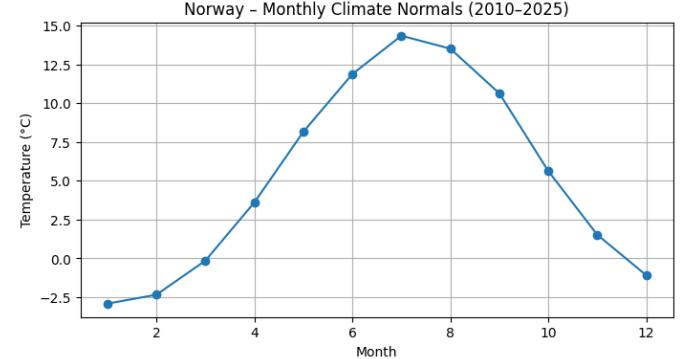


Fig. 1. Norway – Monthly Climate Normals

#### A. Climate Normals (2010–2025)

To establish a baseline climate, monthly temperature and precipitation normals were derived from all complete station-months in the Silver layer. The national temperature normals (Figure 1) show a typical Norwegian seasonal cycle: sub-zero temperatures in winter, rapid warming in spring, and a summer peak near  $14\text{--}15^{\circ}\text{C}$ .

Precipitation normals (Figure 2) indicate that autumn (September–November) is the wettest period, with totals above 120–130 mm, while winter and early spring are comparatively drier.

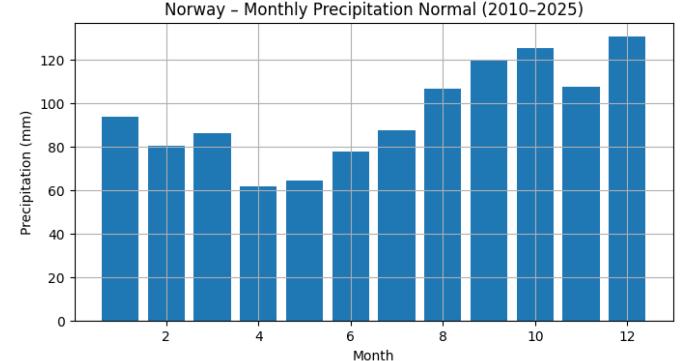


Fig. 2. Norway – Monthly Precipitation Normal

These patterns confirm that the computed normals accurately reflect Norway’s well-known climate of cold winters, mild summers, and high autumn rainfall.

#### B. Temperature Anomalies and Interannual Variability

Monthly and annual temperature anomalies were computed relative to the 2010–2020 climate normals. The anomaly

heatmap (Figure 3) highlights a distinctly cold year in **2010** (anomalies down to  $-5^{\circ}\text{C}$ ) and widespread warm anomalies in **2014, 2018, 2020, 2024, and 2025**. Variability is strongest during winter and early spring, when departures from normal are largest.

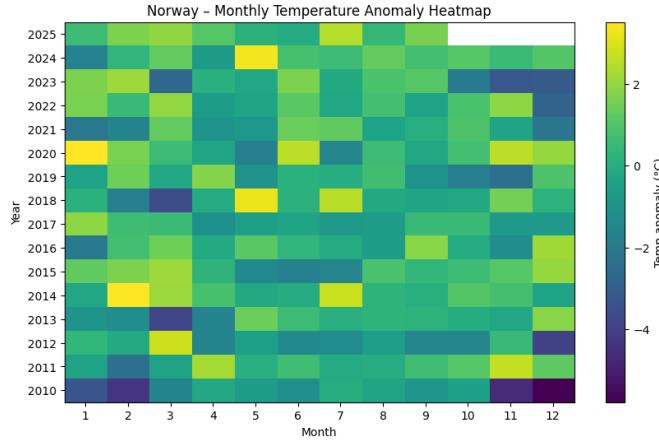


Fig. 3. Norway – Monthly Temperature Anomaly Heatmap

Annual anomalies (Figure 4) show a clear warming trend after 2010, with most years between 0 and  $+1^{\circ}\text{C}$ . The warmest year in the dataset is **2025**, exceeding  $+1^{\circ}\text{C}$ .

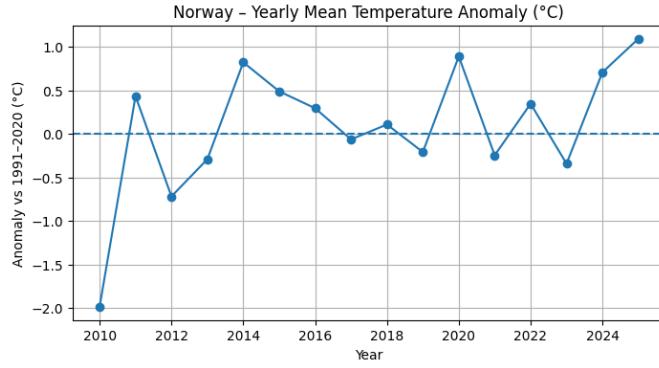


Fig. 4. Norway – Yearly Mean Temperature Anomaly ( $^{\circ}\text{C}$ )

A comparison of 2010 and 2020 (Figure 5) illustrates this contrast: 2010 shows persistently cold anomalies, while 2020 exhibits warm summer and late-autumn anomalies, in some cases above  $+2^{\circ}\text{C}$ .

Overall, the Gold layer effectively captures Norway's interannual temperature variability, including both extreme cold events and recent warm years.

#### C. Machine Learning Model Performance

The Gold layer included several ML models to evaluate the dataset's suitability for predictive tasks.

##### a) Linear Regression - Trend Analysis and Forecasting:

A linear regression on Norway's annual temperature anomalies reveals a warming trend of approximately  $+0.09^{\circ}\text{C}$  per year, projecting mean anomalies above  $+1.3^{\circ}\text{C}$  by 2030 as shown in Figure 6

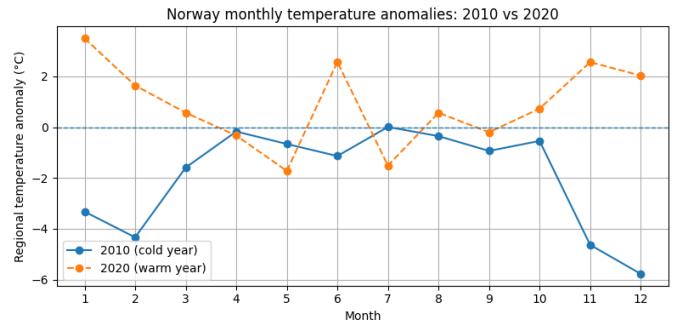


Fig. 5. Norway – Monthly Temperature Anomalies: Cold Year vs Warm Year

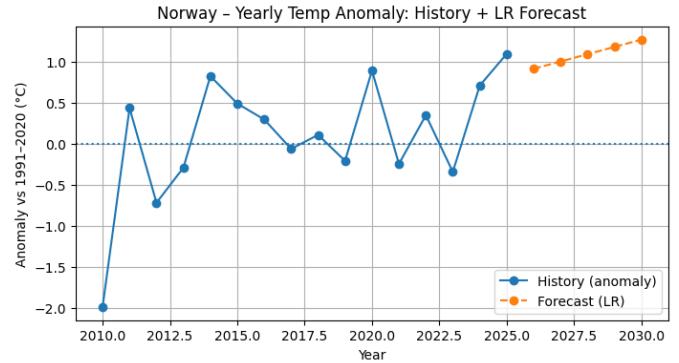


Fig. 6. Norway – Yearly Temp Anomaly: History + LR Forecast

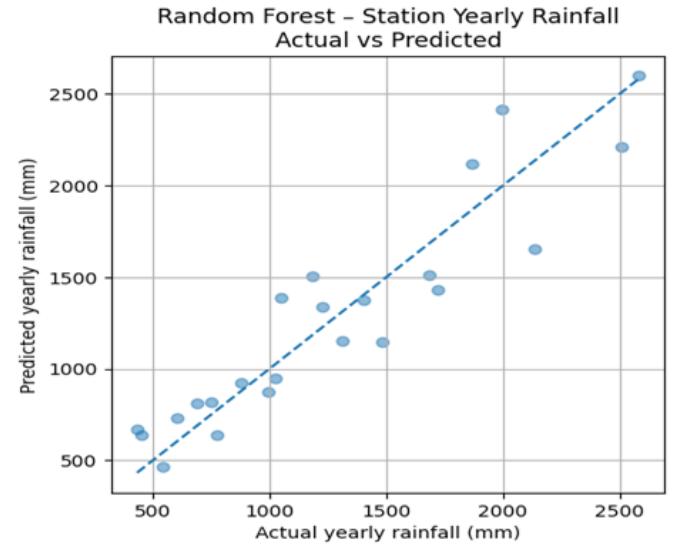


Fig. 7. Random Forest – Station Yearly Rainfall (Actual vs Predicted)

##### b) Random Forest — Annual Precipitation Prediction:

The Random Forest model predicts annual station-level precipitation reasonably well, though the actual-versus-predicted scatter (Figure 7) shows reduced accuracy for extreme rainfall values above  $\sim 2000 \text{ mm}$ , reflecting model limitations and sparse coverage of very wet stations.

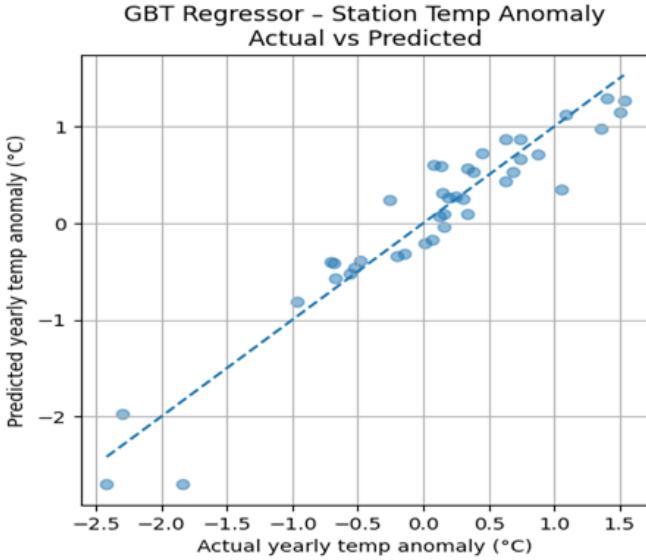


Fig. 8. GBT Regressor – Station Temp Anomaly (Actual vs Predicted)

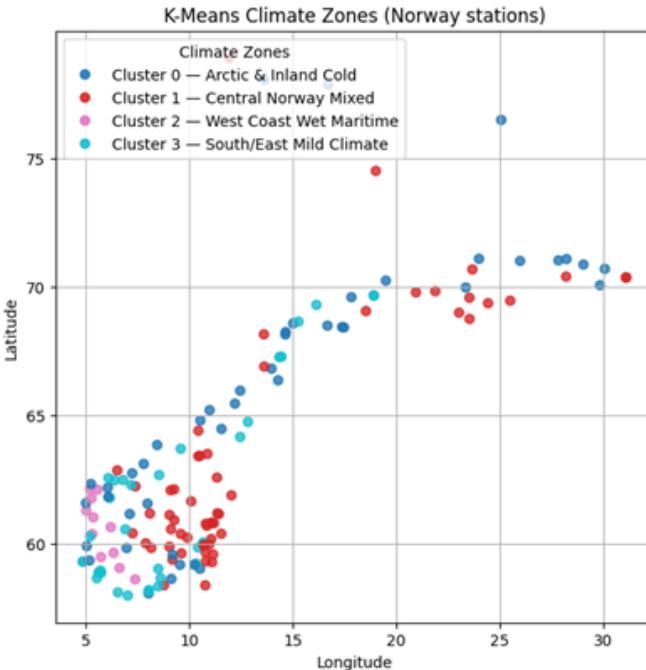


Fig. 9. K-Means Climate Zones (Norway stations)

*c) Gradient Boosted Trees — Temperature Anomaly Prediction:* The GBT model shows strong alignment between predicted and observed annual temperature anomalies (Figure 8), performing best near the climatological mean while still capturing extreme years, demonstrating the Gold layer's ability to support meaningful predictive modeling despite the limited 2010–2025 record.

*d) K-Means - Climate Zone Clustering:* The K-Means clustering algorithm identifies four major climate zones across Norway based on station normals and geographic attributes

(Figure 9). These clusters align well with known Norwegian climate patterns, confirming that the Silver and Gold layer features capture meaningful real-world geographical and meteorological structure.

## V. LIMITATIONS

Although the Medallion Architecture pipeline successfully processed hundreds of millions of GHCN–Daily observations and produced robust Gold-layer climatological outputs, several limitations affect the scope and interpretability of the results.

A primary limitation is the restricted temporal coverage (2010–2025), chosen due to computational constraints. This 16-year period is adequate for short-term anomaly analysis but falls short of the WMO's recommended 30-year baseline, making the derived normals more sensitive to anomalous years and limiting long-term trend assessments [4].

A second limitation is uneven and incomplete station coverage. Many stations report variables inconsistently (e.g., missing TAVG), contain seasonal gaps, or lack stable precipitation normals. Metadata files also include missing or inconsistent entries. These issues required strict completeness filtering, reducing the number of stations available for regional aggregation and machine-learning tasks.

Data quality issues in the GHCN–Daily source further constrain results. The dataset contains sentinel values (-9999), inconsistent flags, and occasional duplicates. Although the Silver layer performed extensive cleaning, the final analyses remain dependent on the accuracy of the original measurements. Parsing fixed-width and irregular NOAA files also required RDD-based ingestion, which increased complexity and processing time.

Finally, infrastructure constraints affected performance. Processing 586 million Bronze records, large Parquet outputs, and heavy aggregations led to memory pressure and Spark warnings. Some computations required tuning or partial filtering, and a larger distributed environment would allow the pipeline to scale more efficiently.

## VI. FUTURE WORKS

Several extensions could further enhance this project. Expanding the temporal and spatial coverage to include the full historical GHCN–Daily archive would enable proper 30-year climate normals, more robust long-term trend analysis, and advanced time-series methods. Adding additional variables such as snowfall, humidity, wind, and solar radiation would also improve dataset completeness.

Scaling the pipeline from Norway to the global GHCN–Daily network would support large-scale anomaly studies, global climate zone clustering, and extreme-event detection across continents, though this would require greater computational resources and further optimisation.

The machine learning layer could be strengthened through richer feature engineering (e.g., ENSO, NAO, topographic indices) and more advanced models such as XGBoost, LightGBM, LSTMs, or spatio-temporal architectures. Interactive

geospatial tools (e.g., Folium, Kepler.gl, Plotly Dash) could offer more intuitive visualisation of climate patterns.

Finally, automating the entire Medallion pipeline with workflow tools such as Airflow, Prefect, or Databricks Workflows would enable continuous ingestion, processing, ML retraining, and dashboard updates, creating a fully operational climate-monitoring system.

## VII. CONCLUSION

This project demonstrated the successful design and implementation of a large-scale climate data processing pipeline using the Spark Medallion Architecture. By applying a structured Bronze–Silver–Gold workflow to the NOAA GHCN-Daily dataset, the system was able to ingest, clean, transform, and analyze hundreds of millions of observations while preserving traceability, quality metadata, and reproducibility. The Bronze layer established a robust ingestion foundation through RDD-based safe parsing, ensuring that all valid and corrupted records were retained for auditability. The Silver layer consolidated raw element-level rows into standardized daily station records, resolving quality flags, handling missing values, and producing an analysis-ready dataset of nearly 186 million daily observations. The Gold layer then transformed these daily facts into meaningful climatological products, including monthly and annual summaries, 2010–2020 climate normals, temperature and precipitation anomalies, regional climate signals, and several machine-learning models for forecasting, classification, and climate-zone clustering.

The results confirm clear and scientifically consistent climate patterns for Norway during the study period, including annual variations in temperature anomalies, regional precipitation differences, and distinct clustering of Norwegian stations into spatially coherent climate zones. The machine-learning experiments—such as linear trend forecasting, random forest precipitation prediction, GBT anomaly modeling, and heatwave classification—demonstrate the analytical potential of the Gold layer, even though their performance was constrained by limited temporal coverage and regional scope. Nevertheless, the pipeline provides a solid foundation for scalable climate analytics and can readily accommodate expanded datasets and more sophisticated models.

Overall, this project highlights the effectiveness of distributed data engineering techniques for environmental data analysis [7], [8]. It shows that Spark-based architectures can support both operational data pipelines and scientific research workflows, enabling the transformation of raw climate measurements into actionable insights [3], [9]. Although several limitations remain—particularly the restricted timeframe, incomplete station coverage, and infrastructure constraints—the modular pipeline developed here is flexible, extensible, and capable of powering future large-scale climatological studies, including global analysis and long-term trend assessment [5].

## VIII. DISCLOSURE

The following AI tools were utilized to assist with development and writing:

- **GitHub Copilot (GPT-5):** Used within VS Code to assist with PySpark notebook authoring and refactoring, generating code suggestions, improving readability, and drafting boilerplate for data processing steps (Bronze, Silver, Gold). Copilot also helped with report editing (e.g., formatting, citation placement, and minor phrasing). All code and text generated or suggested by Copilot were reviewed, tested where applicable, and validated.
- **ChatGPT:** Used to brainstorm structure for sections, summarise technical steps for narrative clarity, and propose bibliography entries and citation styles. ChatGPT was also used to improve explanatory passages and to provide alternative wording.

## REFERENCES

- [1] NOAA National Centers for Environmental Information (NCEI), “Global Historical Climatology Network - Daily (GHCN-Daily), Version 4,” Dataset, <https://www.nci.noaa.gov/products/land-based-station/global-historical-climatology-network-daily>, DOI: 10.7289/V5D21VHZ, Accessed: Nov. 28, 2025.
- [2] NOAA NCEI, “GHCN-Daily Documentation and Format Specification,” <https://www.nci.noaa.gov/pub/data/ghcn/daily/readme.txt>, Accessed: Nov. 28, 2025.
- [3] Databricks, “What is the Medallion Architecture?” Blog/Tech Article, <https://www.databricks.com/glossary/medallion-architecture>, Accessed: Nov. 28, 2025.
- [4] World Meteorological Organization, “WMO Guidelines on the Calculation of Climate Normals,” WMO-No. 1203, 2017. [Online]. Available: <https://library.wmo.int/idurl/4/41302>. Accessed: Nov. 28, 2025.
- [5] IPCC, “Climate Change 2023: Synthesis Report. Contribution of Working Groups I, II and III to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change,” 2023, DOI: 10.59327/IPCC/AR6-9789291691647. Available: <https://doi.org/10.59327/IPCC/AR6-9789291691647>.
- [6] M. Zaharia et al., “Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing,” Proc. 9th USENIX NSDI, pp. 15–28, 2012. Available: <https://dl.acm.org/doi/10.5555/2228298.2228301>
- [7] M. Zaharia et al., “Apache Spark: A Unified Engine for Big Data Processing,” Commun. ACM, vol. 59, no. 11, pp. 56–65, 2016, DOI: 10.1145/2934664. Available: <https://doi.org/10.1145/2934664>.
- [8] M. Armbrust et al., “Spark SQL: Relational Data Processing in Spark,” Proc. ACM SIGMOD, pp. 1383–1394, 2015, DOI: 10.1145/2723372.2742797. Available: <https://doi.org/10.1145/2723372.2742797>.
- [9] Databricks, “The Data Lakehouse: A New Generation of Open Platforms that Unify Data Warehousing and Advanced Analytics,” Whitepaper, Accessed: Nov. 28, 2025. Available: <https://www.databricks.com/research/lakehouse-a-new-generation-of-open-platforms-that-unify-data-warehousing-and-advanced-analytics>
- [10] M. Armbrust et al., “Delta Lake: High-Performance ACID Table Storage Over Cloud Object Stores,” arXiv:1906.01990, 2020. <https://arxiv.org/abs/1906.01990>. Accessed: Nov. 28, 2025.
- [11] S. E. Perkins and L. V. Alexander, “On the Measurement of Heat Waves,” J. Climate, vol. 26, pp. 4500–4517, 2013, DOI: 10.1175/JCLI-D-12-00383.1. Available: <https://doi.org/10.1175/JCLI-D-12-00383.1>.