# Data Science Open-Ended Lab — Weather Data Analysis

**Title: Was the 2018 winter at Canberra unusually long and/or cold?**

**Dataset:** `daily-min-temp-CBR.csv`, `daily-max-temp-CBR.csv` (Canberra Airport station)

**Background:** The Bureau of Meteorology provides daily observational records (min/max temperatures and rainfall). The two CSV files supplied contain daily minimum and maximum temperatures for the Canberra Airport station. Data begins in September 2008 and contains missing entries. Your task is to design, implement, test, and present an analysis that answers (as precisely as possible) whether the 2018 winter (June–August 2018) was unusually long and/or cold compared to previous years.

---

## Learning objectives

1. Practice reading, cleaning and merging real-world CSV data with missing values.
2. Design and implement functions that compute well-defined climate statistics (counts, means, trends).
3. Create informative visualizations (time series, boxplots, histograms, heatmaps) to support your conclusions.
4. Justify choices (definitions of "winter", "cold", and "long") and assess statistical/empirical significance.
5. Write clear, reproducible code and a short report that defends your conclusions.

---

## Dataset description (columns)

- Column 0: Product code (ignore)
- Column 1: Station number (ignore)
- Column 2: Year (integer)
- Column 3: Month (01–12)
- Column 4: Day (01–number of days in month)
- Column 5: Temperature (decimal). May be empty (missing).
- Column 6: Number of consecutive days (always 1 for non-missing)
- Column 7: Quality check flag: 'Y' or 'N' (useful for filtering / reporting but not mandatory to drop rows)

**Note:** The first line is a header; data begins on the second line.

---

## Tasks (required)

### 1) Data ingestion & cleaning

i. Load both CSV files into dataframes.

ii. Parse Year, Month, Day into a single `date` column (ISO format, or pandas `datetime`).

iii. Convert temperature values to numeric and mark missing entries clearly (e.g., NaN).

iv. Merge min and max datasets into a single dataframe keyed by `date`.

v. Document how many missing values exist per month/year and how you handle them (drop? impute? explain your choice).

Deliverable: a cleaned `DataFrame` saved to disk (CSV or pickle), and a short log (text/markdown) of cleaning steps and counts of missing values.

---

### 2) Define precise metrics & baseline definitions

For the rest of the lab you must pick and clearly state each of the items below (you may also provide alternative analyses):

- Definition of *winter* (minimum: June–August). You may also explore extended definitions (e.g., May–September, or winter by meteorological thresholds such as consecutive days below X°C).
- Definition of *cold* (examples: nights with min temperature < 0°C; monthly average of daily minima; average of min and max; median; anomalies relative to long-term mean).
- Definition of *long* (examples: later end/beginning of winter defined by first/last day when daily max > threshold for N consecutive days; or months with statistically significantly lower mean than baseline months).

Deliverable: A concise bullet list in your report stating the definitions you used and why.

---

### 3) Implement analysis functions (examples)

Write modular functions with clear docstrings (you will be graded for clarity and testability):

- `count_subzero_nights(year:int, month:int) -> int`
- `monthly_mean_min_temp(year:int, month:int) -> float`
- `winter_average(year:int, months=[6,7,8]) -> float` (allowable to compute using min, max, or mean of the two)

- `rank_of_winter(year:int, measure:str) -> int`
- rank this winter among the years available using the chosen measure.

Include unit-style tests (simple asserts) that verify sensible behaviour (e.g., counts are between 0 and days-in-month; sum of nights below and above-or-equal-to-zero equals number of non-missing days).

Deliverable: jupyter notebook `.ipynb` with functions and tests.

---

## 4) Visualizations

Produce at least the **four** following visualizations, each with clear titles, labelled axes, legends (if needed), and short captions that interpret the plot.

1. **Time series of daily minimum temperatures (all years)**: plot daily minima as a thin line (or scatter) and overlay a rolling 30-day mean. Highlight the June–August 2018 period.
2. **Monthly boxplots of minimum temperatures for each winter month across years**: e.g., grouped boxplots for June, July, August showing distribution across years (helps see if 2018 is an outlier).
3. **Histogram (or density) comparison**: overlay histograms or density estimates of daily minima for winters: 2018 winter vs. the distribution of winters from other years.
4. **Heatmap calendar**: matrix with years on the vertical axis and day-of-year (or month-day grid) on the horizontal axis showing minimum temperature values or binary cold flag (e.g., sub-zero). This reveals seasonal patterns and the length/extent visually.
5. A plot of the first/last date of winter (by threshold) for each year.
6. **An anomaly plot:** daily min temperature minus long-term daily climatology (average for each calendar day across years).

**Implementation notes:** use matplotlib or your preferred plotting library. Make sure colours, markers, and figure sizes are readable. Save figures as PNGs and embed them in the report.

---

## 5) Statistical comparison & ranking

i. Compare the chosen measure for winter 2018 to the set of available years. Report rank (e.g., coldest = 1) and percentile.

ii. If appropriate, perform a simple paired or unpaired test (e.g., t-test or non-parametric test like Mann–Whitney) to assess whether 2018 winter mean differs significantly from the long-term mean (state assumptions and caveats).

iii. Discuss whether the observed difference (if any) is practically meaningful.

Deliverable: a summary table and short interpretation paragraph.

---

## 6) Robustness checks and testing

Follow the testing guidance in the problem statement. Show at least three sanity checks / cross-checks and explain what they confirm.

**Examples:**

- Ensure counts are between 0 and days-in-month.

- Implement the same summary statistic in two ways and compare results (e.g., compute winter mean by selecting daily minima for months vs. aggregating monthly means).

- Show code and results for at least two choices of missing-value handling (drop vs. simple imputation) and describe the sensitivity of your conclusions.

**Deliverable:** sections in the notebook showing these checks.

---

## Machine Learning for Prediction / Detection

1. **Trend analysis:** fit a simple linear trend to winter minima across years and report slope and confidence intervals.
2. **Season start/end detection:** programmatically detect start and end of winter for each year based on a temperature threshold and consecutive day rule; compute how the length changes with time.
3. **Map / external data:** (advanced) compare Canberra airport to another station (if data available) or fetch 2018 climate anomaly maps (requires internet access and citation).

---

## Report & submission format

i. Submit a single Jupyter Notebook file e.g. `<RegID>_CS_Lab_Weather.ipynb` with code, visualizations embedded, and textual explanations.

ii. `cleaned_data.csv` or `cleaned_data.pkl` (your cleaned merged dataset).

iii. `README.md` explaining how to run your notebook and summarizing definitions used.

iv. `Make a sparate section in jupyter notebook for tests` with simple test scripts.

Write a one-page summary (include in Jupyter Notebook) with your answer to the original question (explicitly stating the definitions you used and the evidence you relied on).

***Good luck***

be curious, document every decision, and explain your reasoning clearly.

***Prepared by: Ghulam Ali, Lecturer, DCS, Air University, AA Campus Kamra***