DACH Ski Resort Advisor

1. Overview

DACH Ski Resort Advisor is a command-line interface (CLI) application written in Julia designed to help users find the best region in Germany (D), Austria (A), or Switzerland (CH) to go skiing or snowboarding based on their preferences of snowfall, temperature, precipitation, and wind. The toolkit evaluates the alpine region by ranking them based on snow and weather observations, including wind, rain, snow quality, and user-defined priorities.

The application automatically downloads the latest dataset from a remote source, caches it locally, normalizes the data schema, and computes fresh-snow deltas. It then presents actionable reports through either an interactive menu or scripted commands. A key feature is the weighted scoring system, which allows users to prioritize factors like fresh powder, a deep base, or calmer weather to match their trip goals.

2. Features

- **Interactive Menu**: A user-friendly, interactive text UI guides users through selecting a resort, filtering country, and adjusting metric weights.
- **Remote Dataset**: The application automatically fetches the geographical data from a remote source and caches it locally, allowing repeated runs to operate offline.
- **Custom Weights**: Users can customize the importance of core metrics: **fresh snow**, **snow depth**, **temperature**, **precipitation**, and **wind**. The tool provides presets (e.g., balanced, powder, family) and allows for manual adjustments.
- **Yearly Snowfall Leaderboard**: Ranks regions by the total amount of fresh snow received in the last 12 months.
- **Region Deep Dive**: Provides detailed historical data for selected regions, with the option to generate and save PNG plots of snow trends and attribute comparisons.
- **Non-interactive Mode**: Supports scripting and automation through command-line arguments, allowing users to run a full report, list regions, or get a summary for a specific region without user interaction.
- **Localization**: Supports multiple languages, with English and German translations included by default.
- **Speech Support**: Speech commands can be enabled by setting an environment variable to an executable that outputs recognized text.

3. Project Structure

The project is organized into the following directories and files:

- **bin/**: Contains the main executable script dach_resort_advisor. This script bootstraps the CLI, checks for dependencies, and calls the main application function.
- **src/**: Contains the Julia source code, organized into modules.
    - **SkiLookup.jl**: The main module that initializes the application, sets shared constants (like the dataset URL and cache paths), and includes all other submodules.
    - **command_line_interface.jl**: Handles parsing of command-line arguments and environment variables and routes to the chosen subcommand.
    - **interactive_menu.jl**: Implements the interactive user menu.
    - **language_support.jl**: Manages multi-language support and translation tables.
    - **reporting.jl**: Generates reports, tables (including the yearly leaderboard and monthly overviews), narrative hints, and plots. It has a reporting_components subdirectory for more granular logic.
    - **transforms.jl**: Handles data loading, discovery, remote downloading, cleaning, column normalization, and transformation (e.g., adding the Snow_New (cm) column). It has a transforms_components subdirectory for modules related to data paths and enrichment.
    - **weights.jl**: Manages the weighting of different metrics, including default presets, environment flag handling, and normalization logic to ensure weights sum to 100%.
    - **utils.jl**: Provides shared utility functions for strings, numbers, user input, TTY checks, and plot directory management. It has a utils_components subdirectory for more specific helpers.
- **data/**: Contains ski resort data. A remote_cache/ subdirectory stores downloaded CSV files, keyed by SHA1, so they are only fetched once.
- **plots/**: The destination folder for generated PNG charts.
- **test/**: Contains tests for the application. The main test script is runtests.jl.
- **Clean data/**: Contains the scripts used for cleaning and processing raw data.

4. How it Works

Entry Point and Initialization

The application starts from the bin/dach_resort_advisor script, which sets up the Julia environment and calls the main() function in the SkiLookup module. The main function, located in command_line_interface.jl, performs the following initialization steps:

1. **Parses CLI arguments** and environment variables to configure the application.

2. **Loads data** by identifying the dataset path (from flags, environment variables, or defaults).
3. **Applies filters** to the data based on user-defined criteria like season or date range.
4. **Initializes metric weights** with default values, which can be overridden by the user.
5. **Dispatches the command** to the appropriate function based on the user's choice (report, menu, list, or region).

Data Transformation Workflow

The transforms.jl module is responsible for loading and preparing the data.

1. The path to the source CSV file is resolved. If it's a remote URL, the file is downloaded and cached locally in data/remote_cache/ if it doesn't already exist.
2. The CSV is read into a DataFrame.
3. Columns are normalized to a consistent format; for instance, date columns are auto-detected and renamed to :Date, and metric headers are harmonized. Metric columns are converted to Float64 to ensure statistical operations work correctly.
4. The data is cleaned, missing values are handled, and a Snow_New (cm) column is derived by tracking increases in snow depth.
5. Country identifiers (e.g., DE, AUT, Deutschland) are reconciled to consistent names for easier grouping.

Reporting Flow

When a report is generated (either via the report command or the interactive menu), the reporting.jl module executes the following steps:

1. **Prints active filters and weights** to inform the user of the current settings.
2. **Prints a yearly snowfall leaderboard**, showing the regions with the most new snow.
3. **Prints a monthly overview table** with average metrics for the latest month, along with narrative tips.
4. **Prints a weighted ranking** of the regions based on the customized score.
5. **Prompts the user to select a region** for a more detailed analysis, which can include historical data and trend plots.

5. Usage

How to Run the DACH Ski Resort Advisor: Open your system terminal. This is the command line interface for your operating system, not the Julia REPL that starts with julia>.

Navigate to your project directory. Use the cd command to go to the folder that contains the Project.toml and Manifest.toml files

Enter: “cd path/to/your/Ski-Resort-Buddy" where you have replaced “path/to/your/Ski-Resort-Buddy" with your file path to the folder containing the Project.toml and Manifest.toml files.

Now to Install dependencies (once): julia --project=. -e 'import Pkg; Pkg.instantiate()'

Launch the application: julia --project=. bin/dach_resort_advisor

When you run the program, you’ll see a guided text menu. Here’s what happens step-by-step:

Step 1 – Select Your Language
The program will ask which language to use (currently English or German). Type your choice and press Enter.

Step 2 – Overview or Country
Next, you’ll be prompted to choose either to select a country or see an overview of all regions.

If you choose to select a country you will then be asked to select the you are interested in visiting: Germany, Austria, Switzerland.

Step 3 – Adjust Metric Weights
You will now be prompted to declare if you want adjust the weighting of the main variables. It is recommended for first time user to select N for No, to gain a better understanding of the program.

If you have chosen y, you will now see the option to either:

Press enter and choose your own weighting

Press 1 to choose the preconfigured Balanced Allrounder

Press 2 to choose the preconfigured Powder Hunter

Press 3 to choose the preconfigured Family friendly

Press 4 to choose the preconfigured Sunny Cruiser

Press 5 to choose again to enter your own weighting

If you choose to enter your own weights, you must now enter custom weight percentages for (total must equal 100) or press enter to keep the default balanced weights. The weight determines how much each factor influences the ranking resort.

Step 4: Active Filter and Weighting
You will now see all active filters that are being applied to the data:

Season: The current season of the year.
Date range: The current range of data is correct.
Region: Whether you have preselected the region.
Observations: The total number of data points being analzyed.

You will see below the active filters the default weighting we are applying to the data:
Fresh snow: Estimates new average snow fall based on snow depth and temperature
Snow Depth: The overall snow fall in centimeters
Temperature: The overall temperature in celcius
Precipitation: The overall rainfall in millimeters
Wind: The wind measure by the Beaufort scale (0-12). A lower value is better.

Step 5 – View the Results
After confirming your weights:
Either the Top ski regions by your chosen weighting will appear or if you requested an overview The yearly snowfall leaderboard appears, showing top-performing regions.
A monthly summary table provides weather and snow averages.
You can then select a region (by number or name) to see more granular data about:
Historical data

Weighted scores

Optional trend plots (saved in the plots/ folder)

Step 6 – Exit the Program
When finished: You can type “exit” or choose the Exit option in the menu. The program will close safely, and your cached data remains saved locally next time.

6. Environment Variables

Several environment variables can be used to configure the application's behavior without command-line flags:

CSV_PATH: Overrides the default dataset location with a file path or URL.

FROM_DATE / TO_DATE: Applies default date filters in YYYY-MM-DD format.

REGION / COUNTRY: Pre-filters the data to a specific area.

WEIGHT_*: Provides numeric overrides for each metric's weight (e.g., WEIGHT_SNOW_NEW=40).

SKI_LOOKUP_LANG / LANG: Pre-selects the interface language (e.g., en or de).

7. Testing

The test suite ensures that every core function performs as expected under different input conditions. All tests are implemented using Julia's built-in Test framework and are in the file test/runtests.jl.

Testing guarantees that the logic behind data handling, scoring, and normalization remains consistent, even as the system evolves. The main purpose of this suite is to:

- Validate the correctness of helper functions within the SkiLookup module.
- Ensure consistent and predictable outputs across all supported inputs.
- Detect and prevent issues early during development.
- Improve the reliability of ski resort rankings and data analysis results.
- Each test set corresponds to a specific function and verifies its behavior against expected results, ensuring smooth integration with the rest of the DACH Ski Resort Advisor system.
- The test suite uses the following Julia packages:
- Test – provides testing and validation tools.
- Dates – handles date values for snow and weather data.
- DataFrames – supports structured data management.
- SkiLookup – the main project module whose helper functions are being tested.
- The tests are executed using: julia --project=. test/runtests.jl

All assertions are automatically verified, and results are displayed as a concise summary showing the number of tests passed and failed.

Tests:

1. Numeric Data Cleaning

This section validates that non-numeric and missing entries are removed correctly from data series while keeping their corresponding date values in sync. It ensures that only valid numeric data is passed into later calculations, preventing downstream errors and inaccuracies in snow or temperature analysis.

2. Valid Value Collection

These tests confirm that only valid numeric entries are collected while ignoring placeholders such as missing, nothing, or NaN. The goal is to maintain clean and reliable datasets for further mathematical computations such as averaging or normalization.

3. Rolling Mean Calculation

This group verifies that the rolling average function correctly applies to a trailing window over a numeric series. It checks that the function uses proper averaging logic, producing smooth trend values over time. Additionally, it confirms that when the window size is one, the original input values remain unchanged, ensuring no unnecessary transformation.

4. Country Canonicalization

This section ensures that country names and codes are standardized to their canonical form. For example, ISO country codes or padded names should be correctly mapped to a consistent name format (e.g., “DE” becomes “Germany”).

This guarantees accurate grouping and filtering by country within reports and leaderboards.

5. Weight Value Parsing

This test verifies that user-entered metric weights are correctly interpreted regardless of format. It confirms that inputs containing whitespace, percentage signs, or locale-specific decimal separators are all recognised as valid numeric values. It also checks that invalid inputs return no result, preventing errors during weight adjustments in the interactive menu.

6. Boolean Parsing

The Boolean parsing test ensures that the program can correctly interpret typical true/false user inputs. It validates tolerance for common keywords such as “yes”, “no”, “on”, and “off”, while safely ignoring unrecognised terms. This enhances usability and robustness when handling configuration or environment settings.

7. Weight Normalisation

This test validates that metric weights are properly scaled to maintain a total of 100 percent.

It confirms that when all weights are set to zero, the system automatically restores default weighting values. By verifying this behavior, the test ensures that weighted resort scores remain mathematically consistent and comparable across sessions.

8. Region Name Conversion

The final test checks the “slugify” function, which converts region names into clean, filesystem-friendly text for saving files or plots. It ensures that special characters, spaces, and mixed cases are converted into safe, lowercase, underscore-separated strings.This prevents issues when generating file names or chart exports.

Functions:

To see a full list of all the functions please download the Function List