Data Management Plan

Predicting Frost Days in Vienna Using Random Forest Classification

FAIR Data Science Exercise Part 2 TU Wien SS 2026

Author / PI	Student 12512103
Institution	TU Wien (Vienna University of Technology)
DMP version	1.0, Initial DMP, project start April 2026
Related document	12512103-use-case-description.pdf (Part 1)
Target repository	TU Wien Research Data Repository (researchdata.tuwien.ac.at)

1. Data Description and Collection

1.1 What data will be collected, observed, or generated?

The experiment works with four types of data:

- **Raw API download:** daily weather records for Vienna Hohe Warte station (ID 5904), years 2000-2023, pulled once from the GeoSphere Austria API and saved unchanged as *raw_geosphere_hohe_warte_2000_2023.csv*. Columns: date, mean/max/min temperature (C), precipitation (mm), sunshine duration (h), relative humidity (%), wind speed (m/s), snow depth (cm). Roughly 8,760 rows and 9 columns, about 500 KB. The raw file will never be overwritten.
- **Processed dataset:** a cleaned version of the above with a new binary column *frost_day* (1 if min temperature < 0C, else 0). Rows with missing values in key columns will be dropped. Features are scaled with StandardScaler. The file will be named with the processing date, e.g. *vienna_weather_processed_2026-04-18.csv*, so I can track when it was generated.
- **Train / validation / test splits:** stratified 70/15/15% split using a fixed *random_state=42*. The split indices will be saved alongside the processed data so the exact same split can be reproduced from the raw file without re-running everything.
- **Model and outputs:** the final selected Random Forest model saved as *rf_frostday_final.joblib*; intermediate models during hyperparameter search kept locally only and not deposited. Output figures (histograms, heatmap, confusion matrix, ROC curve, feature importance) saved as PNG; evaluation metrics exported to *metrics_summary.csv*.

1.2 What is the origin of the data?

The input data comes from the **GeoSphere Austria Data Hub** (data.hub.geosphere.at), specifically the dataset *Messstationen Tagesdaten v2*. I chose this source because it is listed on **data.gv.at** (Austrian Open Data Portal, an approved source for this exercise) and also appears in **re3data**. The licence is **CC0 1.0**, which means I can use, modify, and share it freely without any restrictions. Data is accessed via a stable REST API endpoint, so there is no manual download step that could introduce errors. I will record the exact query URL and retrieval date in the README.

1.3 What formats and volumes are expected?

The total project size is expected to stay well below 50 MB, so storage is not a concern. All file formats are plain text or widely supported binary formats that do not require proprietary software to open.

Artefact	Format	Est. Size	Deposited?
Raw input CSV (unchanged API download)	CSV, UTF-8	~500 KB	No (source stays at GeoSphere API)
Processed dataset (with frost_day col)	CSV, UTF-8	~500 KB	Yes
Final trained model	.joblib (scikit-learn)	1-5 MB	Yes

Intermediate / candidate models	.joblib	1-5 MB each	No (local only)
EDA figures (histograms, heatmap)	PNG	~200 KB each	Yes
Evaluation figures (conf. matrix, ROC)	PNG	~150 KB each	Yes
Metrics summary	CSV, UTF-8	<10 KB	Yes
Source code (notebook + scripts)	.ipynb / .py	~50 KB	Yes
Environment spec	requirements.txt	<5 KB	Yes
DMP (this document)	PDF	~200 KB	Yes

2. Documentation and Metadata

2.1 What metadata and documentation will accompany the data?

The project folder will be structured as follows so that it is obvious what goes where without needing to read the README first:

- *frost-day-predictor/data/raw/*: the original unmodified API download
- *frost-day-predictor/data/processed/*: the cleaned and split dataset
- *frost-day-predictor/models/*: final model only; intermediate models stay in a local scratch folder outside the repo
- *frost-day-predictor/figures/*: all PNG outputs
- *frost-day-predictor/reports/*: metrics CSV and this DMP
- *frost-day-predictor/notebooks/*: the main Jupyter Notebook

A **README.md** in the root will explain: what the project does, how to reproduce the results step by step (including which Python version and how to install dependencies), where the data comes from (API URL, station ID 5904, retrieval date), what each output file contains, and what random seed was used. I plan to write this README as I go rather than at the end, so it reflects actual decisions rather than a cleaned-up summary.

The Jupyter Notebook will include markdown cells at each CRISP-DM phase explaining what I did and why, for instance, why I dropped certain rows, how I chose the hyperparameter search range, and what the evaluation results mean in the context of frost prediction.

2.2 What metadata standards will be used?

The TU Wien Research Data Repository runs on InvenioRDM and uses the **DataCite Metadata Schema 4.x**. I plan to fill in the following fields when creating the deposit. Code and data will be uploaded as a single combined record since they are tightly coupled and the dataset is small enough that splitting them would be more confusing than helpful.

Title	Predicting Frost Days in Vienna Using Random Forest Classification
Creators	Student 12512103, TU Wien
Resource type	Dataset (data and model); Software (code): combined deposit
Description	Short abstract of the experiment matching Part 1 description
Subjects / keywords	meteorology; frost prediction; machine learning; Random Forest; Vienna; Hohe Warte
Related identifiers	GeoSphere API endpoint (IsDerivedFrom); data.gv.at landing page (IsSourceOf)
Rights / licence	CC BY 4.0 for data and figures; MIT for code; CC0 noted for input source
Version	1.0
Funding / context	TU Wien course DaSt 2026, FAIR Data Science exercise

3. Storage and Backup During the Research Process

3.1 How will data and code be stored and backed up?

During the project I will work locally on my laptop. The folder structure described in Section 2.1 will live under *~/projects/frost-day-predictor/*. I will commit to a private Git repository (GitHub) after each meaningful step, roughly after data download, after preprocessing, after the first model run, and after final evaluation. The commit messages will say what changed, not just 'update'. Binary files like the model and figures are not tracked in Git; they get uploaded to the repository at the end.

I do not expect backup to be a major issue given the small dataset size. The processed CSV and all figures together are probably under 10 MB. If my laptop dies I can re-pull the raw data from the GeoSphere API and rerun the pipeline. The random seed is fixed (*random_state=42* throughout) so results should be identical.

3.2 How will sensitive data be handled?

There is no sensitive data in this project. The dataset is weather measurements from a public monitoring station, no personal information, no health data, nothing that would require special handling. I do not need anonymisation or access controls.

4. Legal and Ethical Requirements

4.1 How will copyright and intellectual property rights be managed?

The input data is published under **CC0 1.0**, so I can use and adapt it without asking permission or crediting the source, though I plan to credit GeoSphere Austria anyway in the README and repository metadata, since that is good practice. I will licence my own code under **MIT** and the output files (figures, processed dataset, trained model) under **CC BY 4.0**, which requires attribution if someone reuses them but otherwise puts no restrictions on what they can do with them. All Python libraries I am using (scikit-learn, pandas, NumPy, Matplotlib, Seaborn, joblib, requests) are open source, so there are no licence conflicts.

4.2 Are there any ethical or privacy issues?

No personal or sensitive data are used here. The dataset contains only atmospheric measurements (temperature, precipitation, wind, etc.) from a publicly operated meteorological station. I do not expect any ethics approval to be needed, and there are no privacy considerations.

5. Data Sharing and Long-term Preservation

5.1 How and when will data be shared?

The plan is to deposit all final project outputs to the **TU Wien Research Data Repository** (researchdata.tuwien.ac.at) once the experiment is complete. This repository has CoreTrustSeal certification, which is one of the explicitly approved options in the exercise. I will not re-upload the raw GeoSphere data since it is already openly available at the source API. Instead the deposit metadata will include a RelatedIdentifier pointing to the exact API endpoint and the data.gv.at landing page, so anyone trying to reproduce the work can find the original.

The planned deposit will contain:

- Processed dataset: *vienna_weather_processed_<date>.csv* (with frost_day column and split information)
- Final model: *rf_frostday_final.joblib*
- All output figures as PNG files
- Evaluation metrics: *metrics_summary.csv*
- Source code: the Jupyter Notebook and any helper scripts

- Environment: *requirements.txt* with pinned package versions
- README.md and this DMP

The current plan is to make everything publicly accessible immediately upon publication, with no embargo, assuming no course-related restrictions apply. A DOI will be assigned automatically by the repository.

5.2 Are there any restrictions on data sharing?

I am not aware of any restrictions that would prevent open sharing. The input data is CC0, my outputs will be CC BY 4.0, and the code will be MIT-licensed. There are no commercial, contractual, or confidentiality issues.

5.3 How will long-term preservation be ensured?

Long-term preservation is handled by the TU Wien Research Data Repository, which provides persistent DOIs, checksums, and institutional backup. On my end, using plain text formats (CSV, PNG, PDF) means the files should remain readable without any special software for the foreseeable future. The *.joblib* format for the model is tied to scikit-learn versioning, which is a known limitation, but since the environment spec and the code to retrain the model are also deposited, anyone who needs to reuse the model can recreate it.

6. Responsibilities and Resources

6.1 Who is responsible for implementing the DMP?

This is a solo project, so I (student 12512103) am responsible for everything, data collection, preprocessing, modelling, writing the DMP, and depositing the outputs. There is no team and no supervisor to delegate to.

6.2 What resources are required?

Everything needed for this project is freely available. The software stack (Python 3.11, scikit-learn, pandas, NumPy, Matplotlib, Seaborn, joblib, requests) is open source. The dataset is small enough to run on a laptop, there is no need for cloud compute or a GPU. Repository storage at TU Wien is free for students. The main resource is time, not money or infrastructure.

DMP versioning note: This is version 1.0, written at project start. A final updated version will be submitted with Part 4 once the experiment is complete.

7. FAIR Compliance Summary

The table below maps each FAIR principle to a concrete action in this project. The goal is to show how specific decisions, not just general intentions, support each principle.

Principle	What I will do
Findable	Deposit to researchdata.tuwien.ac.at, which assigns a DataCite DOI automatically. Fill in rich metadata: title, description, keywords (meteorology, frost prediction, Random Forest, Vienna, Hohe Warte), and RelatedIdentifiers linking back to the GeoSphere API and data.gv.at. Add the deposit to the DaSt 2026 community so other students and reviewers can find it.
Accessible	Set the deposit to open access (no login required, no embargo). Both the repository and the GeoSphere source API use standard HTTPS, no special tools or credentials needed. Metadata will remain accessible through the repository even if files are removed.
Interoperable	Use plain, widely supported formats: CSV for tabular data, PNG for figures, PDF for the DMP, .joblib for the model (with scikit-learn version noted). Use SPDX identifiers for licences (CC-

	BY-4.0, CC0-1.0, MIT) and DataCite controlled vocabulary for subjects. Include a RelatedIdentifier pointing to station 5904 data at the GeoSphere API endpoint.
Reusable	Licence everything clearly: CC BY 4.0 for outputs, MIT for code, CC0 noted for the input source. Write a README that explains what the data is, where it came from (including the exact API URL and the retrieval date), how to run the code, and what each output file contains. Fix random_state=42 everywhere and pin all package versions in requirements.txt so results can be reproduced from scratch.