Valtteri Airaksinen

Final Report

TTV20SAI

Customer Project 1

Fall 2025

KAMK • University of Applied Sciences

Table of Contents

1 Introduction

The objective of this project was to develop a system for fetching weather data from the Finnish Meteorological Institute (FMI) interface, preprocessing it, and storing it for later use in machine learning models. The assignment was given to me by Valio and is being documented thoroughly to simultaneously get credit for the Customer Project 1 course.

The project plan defined three main goals:

1. Implement flexible Python functions for fetching data from the FMI database and storing it in Azure Blob Storage.

2. Implement calculations for missing features (e.g., min, max, mean values, and vapor pressure).

3. Compute and integrate effective temperature sums into the dataset for machine learning pipelines.

The plan also emphasized documentation and version control. The entire codebase and related documentation were made available in a public GitHub repository.

2 Requirements

The project requirements were as follows:

- Ability to fetch observed and forecasted weather data from the FMI API.
- Daily automated data retrieval and storage in Azure Blob Storage.
- Capability to compute missing features not directly provided by FMI.
- A modular and well-documented codebase for easy reuse and extension.
- Data is prepared in a format suitable for direct use in machine learning pipelines.

3 Data

3.1 Data Available

This whole project is based around the main goal of successfully fetching data from FMI -interface, that includes various kriging -models and forecast models. The kriging -models required an API-key that was provided to me by FMI as a sign of their support for the prediction model project.

Different kriging models used:

- kriging_suomi_daily
- kriging_suomi_kasvukausi
- kriging_suomi_synop
- kriging_suomi_hourly
- kriging_suomi_snow

For the weather forecasts I used virenwc forecasts, I only needed temperature to calculate the effective temperature sum in the future, in further development when more features are added to the training data other features will be needed too.

3.2 Preprocessing

For preprocessing, the collected data needed to be stored in the same format as the historical weather dataset already available at Valio, which had been purchased from the Natural Resources Institute Finland (Luke). To create a matching dataset, several additional features had to be calculated. These included various minimum, maximum, and mean values, as well as vapor pressure.

While the min, max, and mean values were straightforward to compute, calculating vapor pressure required the use of Tetens' formula. I implemented this as a Python function, which can be found in:

weather-collector/shared/utils.py

4 Architecture

The architecture for the functions created in this project are divided into two separate Azure Function Apps:

- weather-collector
- digestibility-model

Full architecture for weather-collector can be found in the repository. Since the function app is being deployed directly from GitHub, there is a separate folder called **DataCollectorFunc** which contains files **__init__.py** & **function.json**. This folder and its contents are used to define the Azure Function. **__init__.py** is calls **shared/main.py** which calls functions from **shared/utils.py**.

This function is triggered once a day, this is defined in the **function.json** -file

Effective temperature sum calculation is implemented in digestibility-model function app which uses similar architecture but runs every 15 minutes checking for new samples. If a new sample is found, the full model ready data is then constructed. This includes calculating effective temperature sum. While I have not included the full function app for that in this repository, calculating effective temperature sum can be found in **forecasts.ipynb** with thorough documentation.

5 Connection to machine learning methods

While no machine learning methods are used when collecting data, this data will be used to train and run a dense neural network. Data collected in weather-collector is processed to match the already obtained data Valio has. This now extended data along with agricultural data gathered from other sources is then combined to train a prediction model used to predict changes in the quality of silage.

A version of the prediction model trained with the formerly obtained weather data has been running the past summer, this model used the effective temperature sum calculation within its pipeline.

For the past summer, collection of agricultural data was also vastly improved so not only is the dataset being expanded, but the data quality is also being improved.

6 Results

Flexible functions were successfully created for parsing and gathering data from FMI's kriging models, performing the required calculations and preprocessing steps, and finally uploading the processed data to Azure Blob Storage. These functions can be found in:

weather-collector/shared/utils.py.

In addition, the logic for calculating the effective temperature sum for future days was designed and implemented successfully. This solution was deployed and used throughout the summer without any issues, despite being developed under a tight deadline. The implementation and documentation of the effective temperature sum calculation can be found in **forecasts.ipynb**.

In total, achieving these results required approximately 193 hours of work, leaving about one full working day dedicated to finalizing the remaining documentation.

7 Discussion of Results

Reliability:

FMI data is reliable, though understanding and using the API documentation required significant effort.

Reproducibility:

The daily scheduled Azure Function ensures reproducibility and automation. Data is now being collected daily and can be used for many potential future projects.

Generalizability:

While tailored for FMI data, the approach can be adapted to other APIs with similar structures. Thought from my experience, most other APIs are much simpler and easier to use.

8 Future Work

Now that summer has ended and a substantial amount of new data has been collected, the dataset used to train the prediction model can be significantly expanded. Incorporating this new data is expected to improve the model's performance, as a larger and more diverse dataset will better capture seasonal variability and weather-related patterns.

In addition to simply adding more records, new weather features will also be introduced into the dataset. These may include advanced indicators such as global radiation, humidity indices, or growing degree days, depending on their relevance to the prediction task. The next step will be to conduct a feature importance analysis to identify which of these features contribute most to model accuracy.

This process may involve:

- Performing statistical correlation analysis between candidate features and the target variable.
- Using feature selection techniques (e.g., permutation importance, SHAP values, or tree-based feature importance scores).
- Experimenting with different subsets of features to evaluate their impact on predictive performance.

By iteratively refining the feature set, the goal is to build a more robust model that balances complexity with predictive accuracy. Ultimately, the expanded dataset combined with carefully chosen features should result in better generalization and more reliable predictions.