Model documentation and write-up

1. Who are you (mini-bio) and what do you do professionally? If you are on a team, please complete this block for each member of the team.

I work as a researcher and developer at the Finnish Meteorological Institute, mostly working on developing weather impact forecasting tools using machine learning.

2. What motivated you to compete in this challenge?

I have participated in these kinds of challenges previously with good success, most importantly the Streamflow Forecasting Rodeo a couple years ago. In that competition I was using the acronym *salmiaki*. I really like using the scientific approach to find a well performing model, even though it is quite a laborious task. The background knowledge of meteorology and related fields is an advantage.

3. Please provide the machine specs and time you used to train your model and to produce the communication outputs.

The solution was run on Puhti CRAY UNIX supercomputer (https://docs.csc.fi/computing/systems-puhti/) using 13 computing nodes per request, fitting all models for one target month. A total of seven requests were needed to optimize and fit models for all issue dates and sites.

Each of the Xeon Gold 6230 nodes contain 2 x 20 cores @ 2,1 GHz and have 192 GB of RAM. A workload of one computing node contains two target sites and four issue dates, totalling eight tasks. Each task was run with five CPU cores and 23 GB of RAM.

Running the code was also tested on a Dell Precision 3561 laptop (GPU was not used).

The heaviest version of the code, used in the Final Prize Stage and Explainability Stage, takes about 10 hours per month to fit on the Puhti supercomputer on average, including optimization and fitting of the SHAP analysis models. Out of all months, January was the most time consuming, taking over 16 hours to fit.

The most memory-intensive phase of the processing is the dimensionality expansion part, i.e. the calculation of the moving averages, lags, and differences. It can peak the memory consumption temporarily, especially for the PDSI and ECMWF gridded data sets containing multiple grid cells per target domain. The exact memory requirement is not clear, but 23 GB is known to be enough: the peak consumption is most likely in the range of 8–16 GB per task.

Inference is a quite lightweight process compared to fitting. Producing the historical forecasts for 2004–2023 should not take more than an hour, most likely much less when using a supercomputing environment which is recommended for this task also.

Summaries of the forecasts were plotted with one Python script (explain_forecasts.py). The most time consuming phase of that script is the application of the SHAP analysis models, which can take several minutes per target site. This script does not require the supercomputing environment, but processing multiple sites simultaneously would be naturally faster with more resources available.

4. Anything we should watch out for or be aware of in using your model (e.g. code quirks, memory requirements, numerical stability issues, etc.)?

The model is extremely sensitive to the correctness of the input data. The downloading and processing scripts are designed to produce technically stable output, but the current version of the code can not handle missing data very well which can cause the results to explode in some cases. This is the drawback of using linear models with noisy input data. A better handling of missing data should be developed.

Some problems have been encountered with downloading the SNOTEL data lately.

The dimensionality increasing part of the code can temporarily increase the memory consumption. This can happen very easily especially with the gridded datasets (ECMWF, PDSI) which contain multiple grid points.

5. Did you use any tools for data preparation or exploratory data analysis that aren't listed in your code submission?

I did use simple time series plotting with Matplotlib very extensively in parts of the model development:
● time series of the raw input data
● time series of the dimensionality expanded variables
● time series of the principal components (these are produced during inference) ● time series of the output of the models and comparison with the observed streamflow values

This helps keeping track of data sanity at all stages of the modeling process.

6. What are some other things you tried that didn't necessarily make it into the final workflow (quick overview)?

I have used XGBoost for different purposes, and it is the standard when I start a new modeling project. It was the main forecasting tool also in this competition for quite long, but in the end I noticed simple linear models provide extremely good results in this context, and they are easier to fit and use. They are also often easier to interpret.

7. If you were to continue working on your explainability/communication solution for the next year, what methods or techniques might you try in order to build on your work so far? Are there other metrics or visualizations you felt would have been very helpful to have?

I would continue developing the analysis of the principal components. Their interpretation and full potential is still largely unused in the current output. How do different components describe the spatiotemporal distribution of key variables around the catchment domain? Also what roles do the long term teleconnection features play in the early forecast dates of the forecast year?