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 write and self-publish books on machine learning, focusing on topics that go beyond just training the models, such as uncertainty quantification and interpretability. I have a master's degree in statistics and did my PhD on interpretable machine learning.

Besides writing, I also offer consulting and workshops on interpretable machine learning and related topics.

2. What motivated you to compete in this challenge?

I love to write, but sometimes I need practical projects. Especially when writing practical books, I don't want to lose touch with the practical side of machine learning. The Water Supply Forecasting Rodeo fit that perfectly and checked many other boxes too: I am interested in all things earth system science, and the challenge had an additional bonus track on explainability and communication, as well as attractive prizes.

3. High level summary of your approach: what did you do and why?

I describe the approach in the model report. You can find a high-level explanation in the abstract and a detailed description in 1.1 Algorithm and Architecture Selection.

4. Do you have any useful charts, graphs, or visualizations from the process?

See Figure 1 in the model report for the overall prediction pipeline. See also Figures 2, 3, and 5 for details on how I computed the features.

5. Copy and paste the 3 most impactful parts of your code and explain what each does and how it helped your model.

Shortly before the challenge, xgboost version 2.0 was released which introduced quantile regression to the library. This was very helpful for the challenge because it allowed me to optimize directly for the challenge metric.

```
xgb_params = {
        "objective": "reg:quantileerror",
        'seed': random_state,
        'verbosity': 0,
      'enable_categorical': True
}
```

When I analyzed the model errors by issue date, I observed an unexpected pattern: The errors increased toward the end of the month, only to decrease again, while I would expect the errors to decrease monotonically toward the end of the season. I realized that was due to how I was handling the snow features, which I was always calculating from the day before the issue date. Introducing a conditional date helped to fix this problem. See details in the Model Report, p. 4 “Fixing the “End-of-Month Melt Bias” with a conditional date”.

```
def compute_conditional_swe_date(site_id: str, issue_date: pd.Timestamp) -> pd.Timestamp:
        """
        Computes the most relevant data for the SWE measurement for a site

        This function takes the site and issue date and computes the date at which teh swe
        should be captured.
        """
        in_season = {
          "detroit_lake_inflow": [4, 5, 6],
          "pecos_r_nr_pecos": [3, 4, 5, 6, 7]
        }.get(site_id, [4, 5, 6, 7])·
 ....
        no_flow_sites = [
          'san_joaquin_river_millerton_reservoir',
          'merced_river_yosemite_at_pohono_bridge',
          'american_river_folsom_lake',·
        ]
 ....
        if issue_date.month in in_season:
          if site_id in no_flow_sites:
            return issue_date.replace(month=3, day=31)
          else:
          return issue_date - pd.offsets.MonthEnd(1)
      else:
        return issue_date - pd.offsets.Day(1)
```

While not part of the official evaluation, the coverage between the 10% and 90% quantiles was also tracked in the leaderboard. Using a conformal prediction approach, I was able to get the coverage to the desired 80% per site and issue date. To my surprise, it didn’t even hurt the performance of my modeling approach.

```
cp_adjustments = predictions.groupby(['site_id', 'issue_day']).apply(lambda x: {
  ‘lower': np.quantile(x[0.1] - x[TARGET], 0.9),
   ‘upper': np.quantile(x[TARGET] - x[0.9], 0.9)
}).to_dict()
```

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

- CPU (model): Apple M1
- GPU (model or N/A): Apple M1
- Memory (GB): 16 GB
- OS: macOS 14
- Train duration (excluding data downloads):
    - 6 seconds for one year
    - ~2 min for the Leave One Year Out CV
- Inference duration: <1min

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

Nothing I can think of. The modeling setup is pretty standard, stable, and not too computationally demanding (I ran it on my laptop).

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

For most data sources I had a separate Jupyter notebook to explore the data and better understand it. I ended up not using many of the data sources. I also explored correlations between the site water flows and looked at permutation feature importance to learn more about which features were important.

9. How did you evaluate performance of the model other than the provided metric, if at all?

Given the multiple stages of the competition (Hindcast, Forecast, Overall) and the lack of a private leaderboard, it was a challenge not to overfit. That's why I evaluated all modeling steps after the forecast challenge using the target metric (Averaged Mean Pinball Loss) with LOOCV on the years from 1994 to 2003. In addition, I monitored the coverage. I also only made changes that improved the rigor of the model (as well as I could judge the rigor as a non-hydrologist).

For more details, see the Model Report p. 8 1.4.1 Validation.

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

- I tried other models to predict the quantiles, such as CatBoost and quantile regression.
- I tried many of the provided data sources (like the teleconnection data) but ended up using only snow and water flow features.
- I have evaluated several times whether training separate models for each issue date, site, or issue date + site would be better than throwing them all together and using site id and issue date as features.

11. If you were to continue working on this problem for the next year, what methods or techniques might you try in order to build on your work so far? Are there other fields or features you felt would have been very helpful to have?

Something that I think has cost my approach some predictive performance was not using weekly water flow information during the season. Within the season, the model only had the information on how much water had flowed the month before, but not the week before. An improvement would be to incorporate these flows.

I went through a lot of trial and error because I wasn't working together with a hydrologist. Teaming up with a hydrologist and making the model more rigorous would be very helpful.

I would also spend time thinking about the evaluation. The current evaluation focuses more on the larger sites because they have much more water volume and therefore dominate the error metric.

The way my models handle snow water equivalent estimates is very crude. I would try to improve both the spatial aspect of the estimate by, for example, narrowing down the relevant snow stations per site, and the temporal aspect, for example by trying site-specific lag values for when snow melt will arrive at the site.

12. What simplifications could be made to run your solution faster without sacrificing significant accuracy?

The models are already quite fast for both training and inference. It may be possible to reduce the number of iterations and reduce the tree size to get smaller boosted tree models.