Water Supply Forecast Rodeo: Hindcast Report

Abstract

Using historical data from 26 different hydrologic sites we created an ensemble of gradient boosting models that provide a probabilistic forecast for the 0.10, 0.50, and 0.90 quantiles of cumulative, seasonal streamflow volume. There are two model architectures underlying the solution, both based on the Catboost implementation of gradient boosting on decision trees.[1] The first model architecture predicts the quantiles of total cumulative streamflow volume in each season, for each site. The second model architecture predicts the quantiles of monthly cumulative streamflow volume within the season, for each site. The monthly predicted quantiles are then summed over the season to give the seasonal streamflow volume. The models use static features from each site including, the latitude, longitude, elevation, and categorical site id. The models also use dynamic features including, the day of the year, and features derived from localized precipitation, temperature, streamflow measurement from streamgages, and snow water equivalent measurements. The quantile predictions from each model are combined using a weighted average to minimize the prediction error given by the mean quantile loss across the three quantiles. In this report I will further elaborate on the design decisions made while researching and developing the solution.

Technical Approach

Algorithm and Architecture Selection

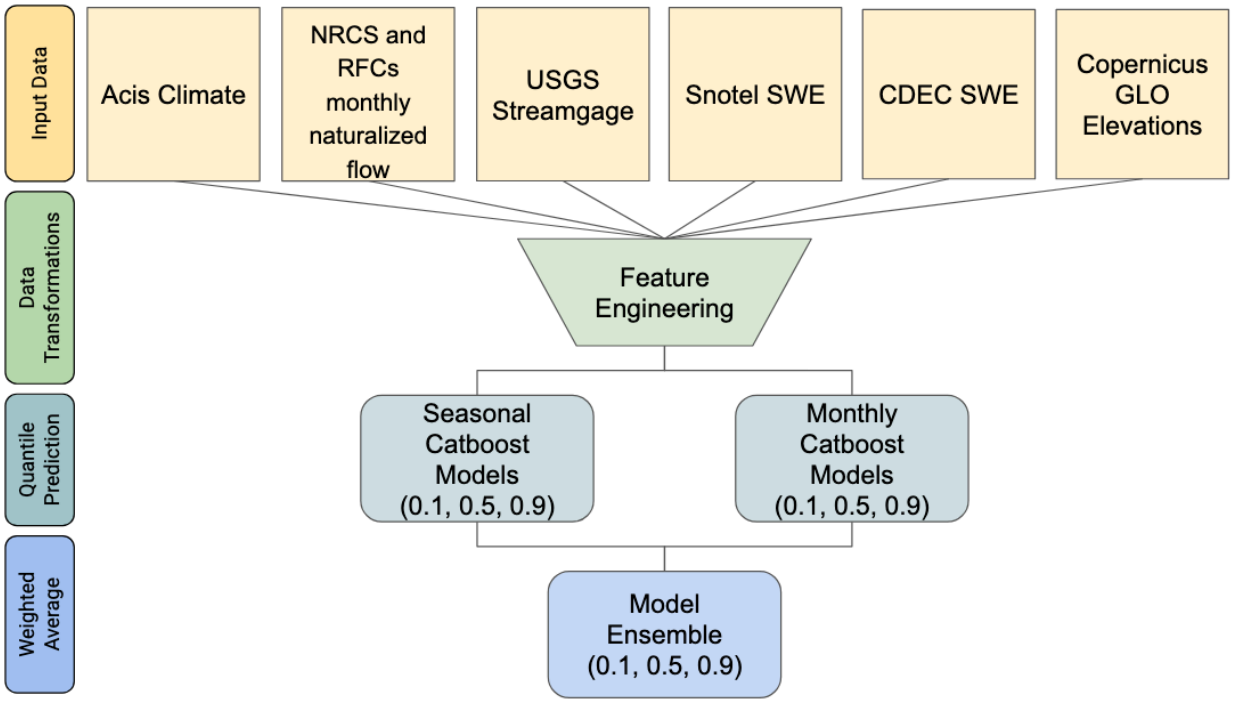

Figure 1: Model Architecture Diagram

Overview

The solution uses two model architectures which both use a Catboost implementation of gradient boosting with decision trees. Gradient boosting with decision trees is a supervised learning method.[2] The models are trained on historical data consisting of the input features and historical labels (targets) that represent the ground truth. The target variable for the first model architecture is the seasonal, naturalized streamflow measurement for each site. The second model architecture uses the monthly, naturalized streamflow measurement for each applicable site as a target and the monthly values are then summed over the season. For each model architecture there is a separately trained model for each of the quantiles, 0.1, 0.5, 0.9. The predicted seasonal quantiles from the two model architectures are combined using a weighted average to get the final seasonal streamflow quantiles.

Log Volumes

Decision tree algorithms are insensitive to the scale of predictors (features) which means that performance is usually not improved by scaling features. However, in this case, the mean quantile loss of the models was minimized when using a log transform on the target variable. Depending on the model architecture, the target variable is the volume or monthly volume transformed using the natural logarithm. The models are trained to predict the log transformed target variable. During test execution the predictions are transformed back using the exponential function (the inverse transformation) to get the final predicted streamflow value for each quantile. Why did using the log transform on the target variable minimize the loss function? The cumulative seasonal streamflow for most sites has a skew-normal distribution with a longer tail and more outliers on the side of larger streamflow values. It may be that log transforming the streamflow values improves the predictions because the loss functions which minimize quantile loss are less strongly influenced by outlier values.

Why Two Models?

The model that is trained on the seasonal value of naturalized streamflow is necessary because some of the sites do not have monthly data labels but all of the sites have historical seasonal values. In addition to the seasonal model we developed the monthly model because it offers an advantage when there are ground truth monthly observations within a season. The model is trained to predict the monthly quantile values for each of the relevant months within the streamflow measurement season. The quantiles for each of the months are then summed to obtain the total cumulative streamflow quantiles for the season. In the later months of the season, it is possible to observe ground truth labels for monthly streamflow that occurred in prior months within the season. Those ground truth values can be swapped with the monthly predictions, increasing the overall accuracy of the cumulative, seasonal streamflow prediction. Combining the predictions from the two models using a weighted average further lowers the mean quantile loss of the predictions.

Gradient Boosting for Decision Trees

The gradient boosting trees algorithm was selected for initial experimentation because it trains quickly (allowing for rapid experimentation with feature engineering/selection), provides good baseline predictive accuracy, and offers many tools for interpreting the model. Gradient boosting

with decision trees can also handle NaN value data inputs which greatly expands the possibilities for feature engineering and training data. We originally started by using the Xgboost implementation of gradient boosting with decision trees, which recently implemented quantile regression in the 2.0.0 release.[3] We made progress minimizing the mean quantile loss using the Xgboost implementation but we found even better success experimenting with Catboost.

Catboost and Categorical Features

One downside to the Xgboost implementation of gradient boosting trees is that it doesn't easily handle categorical features. We tried using one-hot encoding of categorical variables but the performance was not significantly improved. We hypothesized that the categorical site_id for each stream site would be a useful feature for the model, allowing the model to build sub-trees unique to each stream site. The Catboost implementation of gradient boosting trees was designed with the intention of maximizing the utility of categorical features and also offers quantile regression.[4] Catboost provided the best model performance.

Future Improvements and Simplifications

The API for the Catboost algorithm is well-defined and designed for minimal hyperparameter tuning. Therefore the majority of the complexity in the solution is in the feature engineering. If we were to try to simplify the solution without sacrificing significant accuracy the first step would be to remove specific features that only provide marginal improvements in accuracy (discussed more in section on Data Sources and Feature Engineering).

A limitation of gradient boosting for decision trees as a regressor is that it does not extrapolate. It is only capable of making predictions for a target variable within the distribution observed during the training process. For the 26 stream sites in this competition the inability to extrapolate did not appear to be an issue because there was enough training data to establish a sufficient distribution. However, the model might struggle predicting stream sites with less training data or could struggle in certain years that have extreme outlier streamflow particularly on the high side of the distribution.

Other Models Considered

To explore a model capable of extrapolating outside the training distribution we experimented with a linear quantile regressor.[5] The best performance for this model was achieved by identifying the three features that were most correlated with the target variable for each issue month, site_id pair. Then a single variable linear quantile regressor was trained for each feature, issue month, site_id. Finally the predictions for the three models using the most highly correlated features for each issue month, site_id pair were averaged. This model performed surprisingly well given its simplicity but it was not able to achieve the accuracy of the Catboost models. It also required NaN filtering and other complications in the data engineering pipeline. This model may perform better in cases where there is limited training data for a stream site and with further effort an ensemble of this model with the Catboost models could provide marginally improved predictive accuracy over the final solution proposed in this report.

Data Sources and Feature Engineering

One of the advantages of using gradient boosting for decision trees is that training models is simple and fast which allows for quick iteration while experimenting with different features. Most of the complexity in the solution is in the feature engineering that creates the inputs into the Catboost models. While the source data for both models is the same, the actual features used by the seasonal and monthly model architectures are subject to different transformations. Recursive Feature Elimination with Cross-Validation (RFECV) was employed for feature selection. Features were chosen for each model using an iterative approach where the model was trained with and without the feature and the feature was kept if it improved the model's loss score. The goal through research and development of the model was to explore as many of the approved data sources as possible.

Static Features
The solution uses both static (permanent characteristics of each site) and dynamic (time-sensitive) features. The static features from each site include, the latitude, longitude, elevation, and categorical site id.

- **site_id:** The categorical site identifier for each of the 26 streamflow sites.
- **pred_month**: The month to predict a naturalized streamflow value for (only used in the monthly model). Used as a categorical variable.
- **elevation**: The elevation where the streamflow measurement occurs for each site as provided in the metadata file.
- **elevation_stds**: The standard deviation of the elevation over the entire basin of each site. The source for this data is the Copernicus Digital Elevation Model GLO-90.[6]
- **latitude**: The latitude for each site as provided in the metadata file.
- **longitude**: The longitude for each site as provided in the metadata file.

Dynamic Features
The dynamic features include, the day of the year, and features derived from localized precipitation, temperature, streamflow measurement from streamgages, and snow water equivalent measurements. Many of the dynamic features used by the model are z-scores of physical measurements localized to a specific site basin. To obtain the z-scores individual measurements are normalized by subtracting the historical mean and dividing by the historical standard deviation.[7] The historical mean and standard deviation are calculated with respect to a day of the year and measurement location. These variables are suffixed with "_deviation" below. If the feature is an aggregation of multiple measurements, like the Snotel station measurements, then the historical mean and standard deviation are first calculated for each station and the z-scores for all the stations on a given date are averaged with respect to the stream site.

The majority of the dynamic features were also transformed by taking a rolling average, using either a 30-day window or a window that encompasses the entire streamflow season up to the day before the issue date. The first day of each streamflow season is October 1st of the year preceding the forecast year.

- **day_of_year:** The day of the year corresponding to the issue date.[8]
- **prev_month_volume**: The volume of the naturalized flow at the forecast sites for the month prior to the issue date. The source of this data is the NRCS[9] and the RFCs[10].
- **precip_deviation**: The z-score of precipitation as measured by weather stations within the site basin using either a 30-day or season long window for the rolling average. The source for this data is the Applied Climate Information System (ACIS), maintained by the NOAA Regional Climate Centers (RCCs).[11]
- **max_temperature_deviation**: The z-score of max temperature as measured by weather stations within the site basin using either a 30-day or season long window for the rolling average. The source for this data is the Applied Climate Information System (ACIS), maintained by the NOAA Regional Climate Centers (RCCs).[12]
- **combined_swe_deviation**: The z-score of snow water equivalent as measured by stations in sites located in or near the site basin using either a 30-day or season long window for the rolling average. There are two sources for this data. All of the sites used data from applicable NRCS Snow Telemetry (Snotel) stations.[13] The California sites used data from applicable California Data Exchange Center (CDEC) stations.[14]
- **streamflow_deviation**: The z-score of streamflow measurements for each site or an applicable site. These measurements represent actual observed flow of water at specific locations, and not the naturalized flow being forecasted. The source of this data are streamgages managed by the U.S. Geological Survey (USGS).[15]

The full list of features used by the seasonal streamflow model: site_id, latitude, longitude, elevation, elevation_stds, prev_month_volume, day_of_year, streamflow_deviation_30, streamflow_deviation_season, precip_deviation_season, combined_swe_deviation_season. The seasonal streamflow model performed better when the window for the rolling average of the dynamic features was the entire streamflow season.

The full list of features used by the monthly streamflow model: site_id, pred_month, latitude, longitude, elevation, elevation_stds, prev_month_volume, day_of_year, streamflow_deviation_30, precip_deviation_30, maxt_deviation_30, combined_swe_deviation_30. The monthly streamflow model performed better when the window for the rolling average of the dynamic features was the 30 days prior to the issue date.

Physical Explanation or Intuition
One of the driving intuitions for converting the physical measurements of temperature, precipitation, snow water equivalent, and streamflow into z-scores was to create unitless features that allow for easy aggregation across stations and increase the generality of the model between stream sites. The z-scores calculated for a measurement are invariant with respect to time and space, meaning that you wouldn't a priori expect the z-score for a measurement to increase at any specific date in the streamflow season or in different geographic regions. However, the z-scores are correlated in time and space, so you would expect the z-score of measurements near each other to rise and fall together. For these reasons the correlation between the features based on physical measurements and the target variable, the log of

naturalized streamflow, increased after they were transformed into a z-score for each streamflow site.

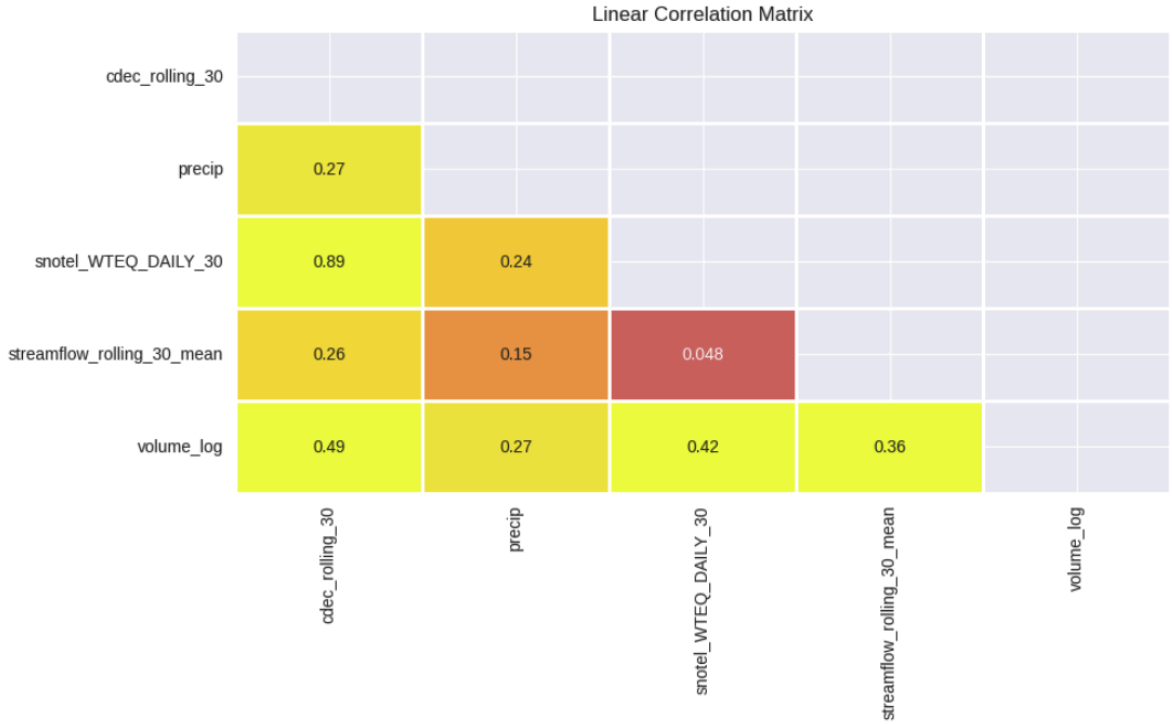

Figure 2: Log volume correlations with raw physical measurements averaged by site

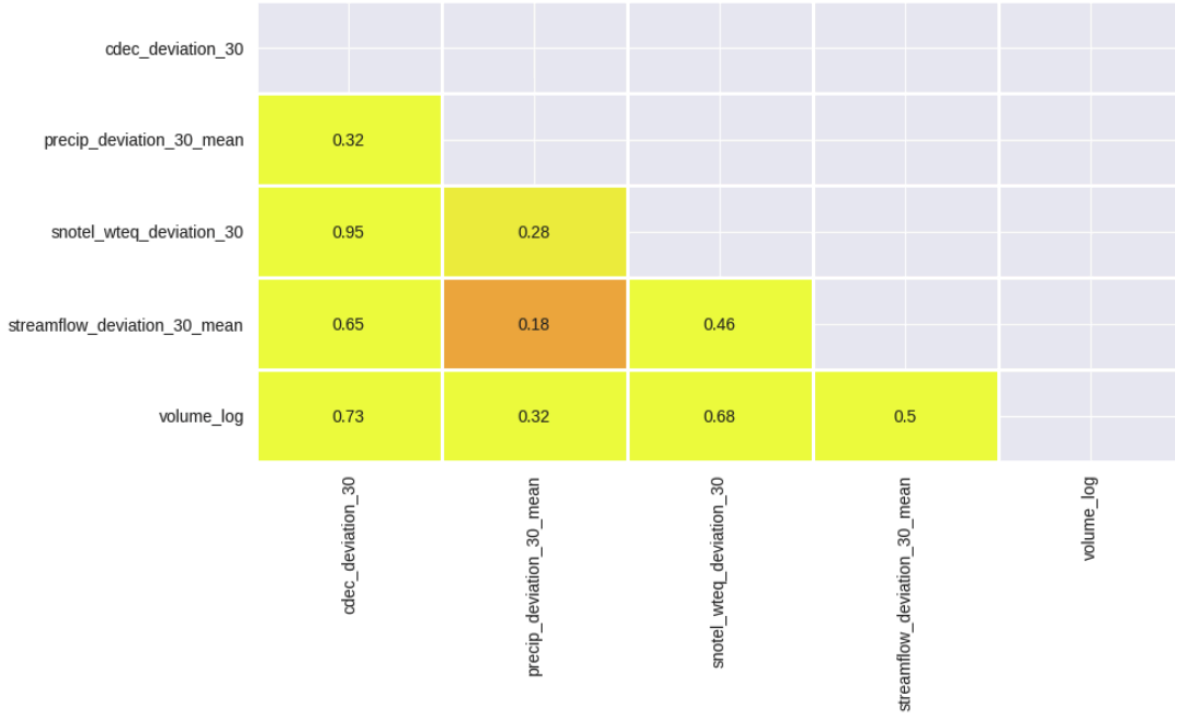

Figure 2: Log volume correlations with z-score of physical measurements averaged by site

Other Data Sources considered

Transforming the predictions from the Snow Data Assimilation System (Snodas)[17] into a z-score and using it in the combined SWE feature improved the model loss function marginally. However, the Snodas data added significant complexity to the engineering pipeline because the data uses a cumbersome file and compression format that adds significant execution run time to the solution. Additional processing was required to restrict the predictions to specific site basins. We experimented with the CPC Seasonal Outlooks[18] for precipitation, trying various transformations looking for correlations with streamflow volume. Ultimately the CPC Seasonal Outlooks didn't reduce the loss of the final model. We also tried using the climate teleconnection indices, specifically the Oceanic Nino Index (ONI) in the models[19]. This feature exhibited potential for early season predictions on streamflow, especially with certain stream sites. The correlations between streamflow volume and the ONI in early January and February were equal to or better than some of the features used for a handful of streamflow sites. This feature was useful in the simplified linear regression model discussed earlier in this report. However, this feature did not improve the Catboost models and therefore was left out of the final feature list.

Uncertainty Quantification

There are six total models, one model for each of the quantiles (0.1, 0.5, 0.9) of each architecture type (seasonal, monthly). Each of the models are Catboost implementations of gradient boosting with decision trees fit using a Quantile loss:

$$\frac{\sum_{i=1}^{N} (\alpha - I(t_i \leq a_i))(t_i - a_i) w_i}{\sum_{i=q}^{N} w_i}$$

Where $\boldsymbol{\alpha}$ is a quantile in (0.1, 0.5, 0.9). The seasonal model architecture predicts the quantiles for the cumulative naturalized streamflow for the entire season. The monthly model architecture predicts the quantiles for the naturalized streamflow for each month in the season. The monthly predictions are then summed to calculate the final cumulative streamflow for the season. If there are observed values of naturalized streamflow in months prior to the month in which the issue date occurs, those are substituted for the quantile predictions. The quantiles for the cumulative seasonal streamflow predictions for each model type are combined using a weighted average that allocates a weight of 0.6 to the monthly model and 0.4 to the seasonal model for the first 4 months of issue dates, before allocating all of the prediction to the monthly model for the final 3 months of issue dates. For the models that do not have monthly labels (american_river_folsom_lake, merced_river_yosemite_at_pohono_bridge, san_joaquin_river_millerton_reservoir) the predicted quantiles from the seasonal streamflow models are used as the final predictions.

Training and Evaluation Process

Missing Values Processing and Historical Data

The first step in training the model was to generate training data. While some streamflow sites have seasonal training labels dating back to the early 1900's many of the data sources used to generate features do not have data going back that far. The Catboost implementation of gradient boosting on decision trees allows for null data in features, therefore, we can train on years where some of the features are null for all sites. We used Catboost's default implementation to process missing values, "missing values are processed as the minimum value (less than all other values) for the feature. It is guaranteed that a split that separates missing values from all other values is considered when selecting trees."[20] This means that the model can recognize null data and train efficiently. However, the data points trained with null data may not apply to data points with non-null values for all features because they will take different paths in the decision trees. We are more likely to encounter data points with non-null features in the more recent years that make up the test population. So while training on historical data with null value features isn't likely to significantly hurt the models' performance, it isn't likely to help it either because the sub-trees that branch on null values won't apply to the data points that are not null. For that reason we chose a cut-off date of 1960 to generate training data, however we also experimented with 1980 (around the time that many Snotel stations started generating snow water equivalent measurements) and the performance difference was minimal.

Cross Validation

As described in other parts of this report there are six total models, one model for each quantile that predicts log-transformed seasonal streamflow and one model for each quantile that predicts log-transformed monthly streamflow. The predictions are then transformed using the exponential function to obtain values that apply to the actual labels. All model architecture decisions were made using cross-validation on the training dataset. The hold-out test dataset used 10 years of data (odd years from 2005–2023). While the models were trained on all data after the year 1960 outside of the hold-out test dataset the models were only scored on the even years from 2004-2022 (hereinafter referred to as validation dataset). We hypothesized that these years would be more applicable (and have more similar feature representations including non-null data) to the hold-out test dataset than years prior to this time period. The cross-validation scheme held out 2-3 years from the validation dataset while training on the remainder of training data. It iterated through all of the possible validation dataset permutations and then generated a score using mean quantile loss. The basis for all decisions on whether to include features, how to transform features or targets, and hyperparameter tuning were made based on whether they lowered the cross-validation score for the model. If adding a feature significantly increased the complexity of the solution but only decreased the loss function by a small amount then that feature was not included in the final model.

Machine Specifications

The models were trained using the CPU on a MacBook Pro with a 2.6 GHz 6-Core Intel Core i7 processor and 16 GB 2667 MHz DDR4 memory. The python environment was configured to resemble the runtime available via the water-supply-forecast-rodeo-runtime Github repository.[21]

References

1. https://catboost.ai/
2. https://en.wikipedia.org/wiki/Supervised_learning
3. https://xgboost.readthedocs.io/en/latest/python/examples/quantile_regression.html
4. https://catboost.ai/en/docs/concepts/loss-functions-regression#Quantile
5. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.QuantileRegressor.html
6. https://planetarycomputer.microsoft.com/dataset/cop-dem-glo-90
7. https://en.wikipedia.org/wiki/Standard_score
8. https://pandas.pydata.org/docs/reference/api/pandas.Period.dayofyear.html
9. https://www.nrcs.usda.gov/
10. https://water.weather.gov/ahps/rfc/rfc.php
11. https://www.rcc-acis.org/docs_webservices.html
12. https://www.rcc-acis.org/docs_webservices.html
13. https://www.nrcs.usda.gov/wps/portal/wcc/home/aboutUs/monitoringPrograms/automatedSnowMonitoring/
14. https://cdec.water.ca.gov/
15. https://waterservices.usgs.gov/docs/dv-service/daily-values-service-details/
16. https://climate.arizona.edu/snowview/csv/Download/Watersheds/
17. https://nsidc.org/data/g02158/versions/1
18. https://www.cpc.ncep.noaa.gov/pacdir/NFORdir/HUGEdir2/explanation_fdf.html
19. https://www.cpc.ncep.noaa.gov/data/indices/oni.ascii.txt
20. https://catboost.ai/en/docs/concepts/algorithm-missing-values-processing
21. https://github.com/drivendataorg/water-supply-forecast-rodeo-runtime