Contents

1. Introduction 2
2. Problem Description 3
3. Data 4
4. Method: 5
   4.1 Feature selection 6
   4.2 Model Selection 8
   4.3 Model Optimization 9
5. Results 10
6. Our Model Evaluation 11
Summary 14

1. Introduction

In recent years, machine learning (ML) has emerged as a powerful tool for streamflow forecasting. While traditional process-based hydrological models remain effective, they often rely on extensive physical measurements, involve complex setup procedures, makes strong physical assumptions, and can be computationally intensive—challenges that hinder their application in operational settings. Moreover, process-based models are typically developed on a basin-by-basin basis, limiting their generalizability and making them unsuitable for regional application across large numbers of ungauged basins. ML offers a compelling alternative through its ability to learn patterns directly from meteorological and watershed data, its capacity to develop generalizable regional models, and its potential to enable accurate and efficient streamflow predictions.

One key strength of ML models is their flexibility to diverse challenges in hydrological forecasting. For instance, they can efficiently account for uncertainty by incorporating probabilistic techniques and ensemble approaches, providing more credible and informative predictions. Predictive uncertainty stemming from data bias—such as uneven spatial or temporal data distributions—can be addressed through preprocessing, resampling strategies, and advanced model calibration. Additionally, ML models have shown strong potential in transferring knowledge to ungauged locations using methods like transfer learning and domain adaptation, making them particularly useful in regions with limited observational data. At the same time, the scalability of ML approaches makes them well-suited for large-scale applications, where they can process and learn from diverse datasets spanning multiple climatic and geographical zones.

Overall, machine learning methods offer flexible, scalable, and more accurate streamflow forecasts, addressing many of the limitations faced by traditional process-based hydrological models.

2. Problem Description

The project involves developing a predictive model to forecast weekly streamflow levels for river basins in France (Adour-Garonne, Rhône-Mediterranean) and Brazil (Doce River). The provided dataset includes:

- **Training data** from 1990 to 2003
- **Evaluation data** from 2004 to 2009, divided into multiple time segments

Each evaluation segment contains:

- 4 weeks of historical streamflow and climate data
- A 1-week weather forecast
- A 4-week prediction horizon for streamflow

The goal is to predict weekly streamflow values and quantify uncertainty for 52 hydrometric stations:

- **39 stations** are included in the training data (temporal split)
- **14 stations** have no historical data in the training set (spatio-temporal split)

Models must rely solely on the provided spatiotemporal data and geospatial features.

Performance is evaluated using the **Non-Negative Gaussian Log-Likelihood (NNGLL)** metric, which assesses both prediction accuracy and the reliability of uncertainty estimates. The NNGLL penalizes large prediction errors and poorly calibrated confidence intervals. To receive a score, models must achieve over **90% coverage**.

Submission and Limitation:

The results are submitted through the online portal at https://www.codabench.org/competitions/4335/. Each participant is allowed up to **10 submissions per day**, with a total limit of **100 submissions** until the deadline on **2025-04-21**.

Objective: The objective of the project is to develop a model that forecasts weekly streamflow across selected river basins in France and Brazil. The model must provide accurate predictions along with reliable uncertainty estimates using only the provided spatiotemporal and geospatial data.

3. Data

To support the development of our machine learning model for streamflow forecasting, we were provided a comprehensive dataset covering three hydrographic regions: the Adour-Garonne and Rhône-Mediterranean basins in France, and the Doce River basin in Brazil. The data was divided into two timeframes: a **training period** from 1990 to 2003 and a **testing period** from 2004 to 2009.

In France, there were 30 hydrometric stations in the training set and 39 in the testing set, with 9 of those stations appearing only in the test period. In Brazil, the dataset includes 9 stations for training and 13 for testing, with 4 stations used exclusively in the test period.

The data used in the model are:

Predictor Variables:

1. Meteorological:

- Temperature at 2 meters above ground (weekly mean)
- Total precipitation (weekly cumulative)
- Evaporation (weekly mean)
- Soil moisture (weekly mean volumetric content)

2. Static Watershed Attributes:

- Elevation (altitude_DEM): Derived from NASA's SRTM with 30m resolution
- Soil Properties: Provided at multiple depth intervals (0–200 cm) via Soilgrids:
  - Bulk density (bdod)
  - Coarse fragment volume (cfvo)
  - Clay and sand proportions

3. Weekly mean flow from the two preceding weeks

Target Variable:

- Streamflow: Four consecutive weekly mean flow measured at hydrometric stations

4. Method:

As part of the project, we were provided with three baseline machine learning models:

1. Quantile Random Forest
2. MAPIE with LightGBM (LGBM)
3. An Ensemble models

We initially ran all provided models to understand the baseline performance under the conditions provided. As expected, the forecast accuracy declined progressively with each forecast week—Week 1 performed best, followed by Week 2, then Week 3, and Week 4 showing the poorest results.

One key issue we initially observed was the high number of input features. While many of these features are relevant, using too many can lead to model overfitting, as the model attempts to assign importance to all variables and loses focus on what truly drives streamflow prediction. This is especially problematic given our goal to build a model that can generalize to both **unseen spatial and temporal conditions**.

4.1 Feature selection

To address the challenge of overfitting and improve model generalization, we performed feature selection to identify the most significant predictors of streamflow. We focused specifically on **Week 4**, as it consistently showed the weakest performance. Our feature selection process followed these key steps:

i. **Feature Importance Ranking**
We began by training a Random Forest model on the entire training dataset to rank the importance of each feature. This provided a global importance ranking that accounted for all spatial and temporal contexts present during the training period.

ii. **Custom Train-Validation Split**
To align with the objective of the project, we created a custom data split—spatially and temporally separated within the training set (Training data into *train* 80% and *validate* 20%). This design reflects the problem objective.

iii. **Model for Fast Experimentation**
Given the large number of features and the need for rapid experimentation, we used a parsimonious model based on XGBLinear. This model offered fast training times while maintaining consistency through fixed hyperparameters and a set random seed.

iv. **Incremental Feature Selection for Optimal Features Combination**
Using only the *train* portion of the split, we applied an incremental feature selection strategy. Starting with the top-ranked feature, we added one feature at a time in order of importance. At each step, the model was retrained, and performance was evaluated using R^2 on the *validate* set. Features were retained only if they improved model performance.

v. **Final Feature Set Selection**
The feature combination that yielded the highest R^2 on the *validate* set for Week 4 was selected as the optimal subset.

We found a combination of 12 feature variables gave best results. The selected features are:
- Lagged streamflow: *water_flow_lag_1w, water_flow_lag_2w*
- Soil moisture indicators: *soil_moisture_region, soil_moisture, soil_moisture_sub_sector*
- Precipitation variables: *Precipitation_region_lag_1w, precipitation_sector_lag_1w, precipitation_sub_sector_lag_1w, precipitation_sub_sector, precipitation_zone*
- Temperature variables: *temperatures, temperature_region*
- Evaporation: *evaporation_sub_sector_lag_1w*
- Spatial/geographical feature: *catchment*

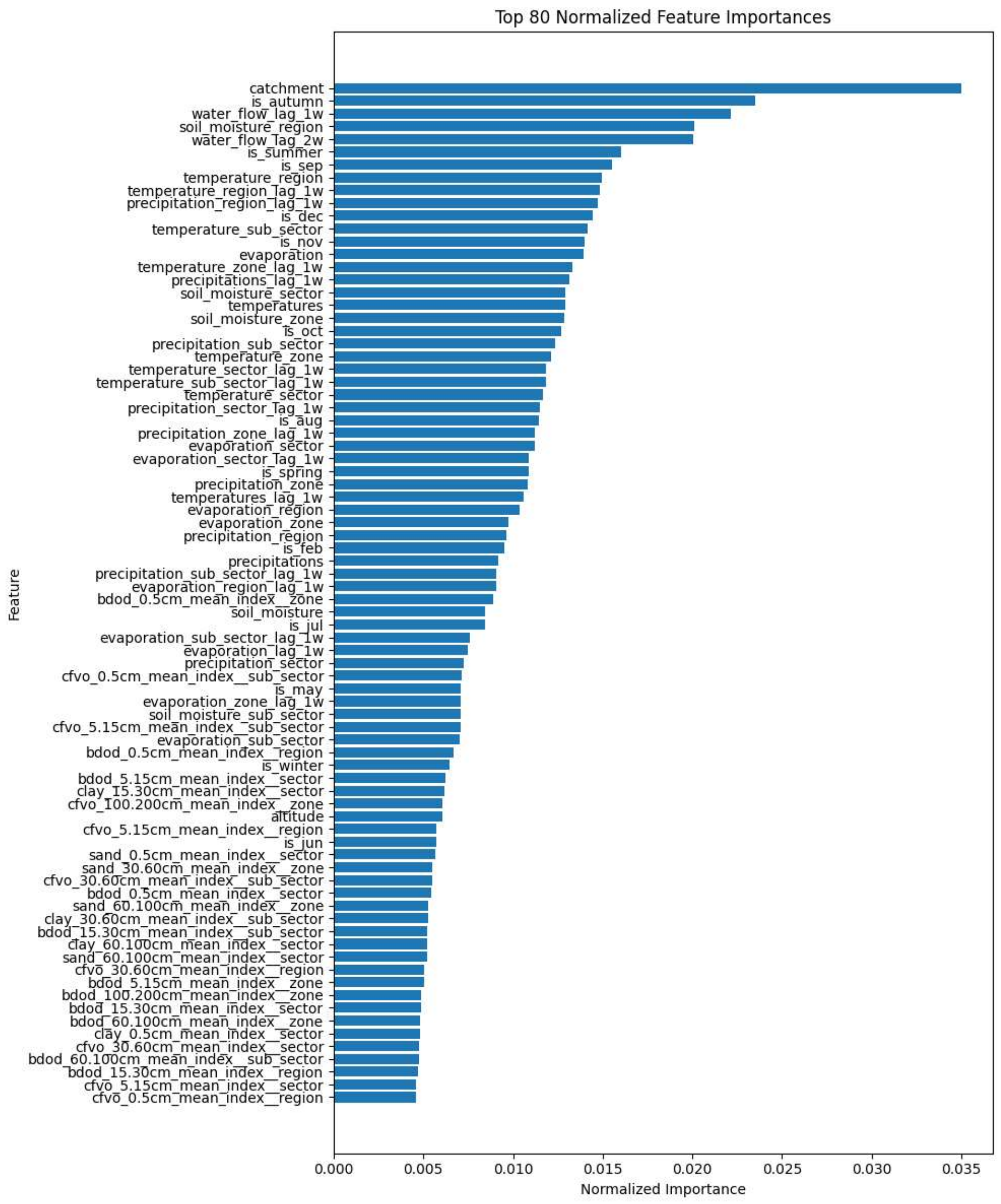


Figure: *Normalized feature importance plot (top 80 features shown for clarity). Among the most important features, we tested multiple combinations that best represented meteorological and catchment characteristics - which is key for model generalizability and out-of-sample performance. The final feature set was selected based on such feature combination that achieved the best validation score.*

4.2 Model Selection

We used the selected 12 features to develop our models. Initially, we worked with the baseline models provided as part of the project. For evaluation, we now used the **Non-Negative Gaussian Log-Likelihood (NNGLL)** as required by the project. This allowed to select a model that not only performed well in terms of expected values but also produced reliable probabilistic predictions, aligning with the goals of this project.

From our initial assessment, the **Quantile Random Forest (QRF)** - designed by default to produce probabilistic forecasts – consistently delivered the best performance across all forecast weeks. QRF not only provided accurate predictions but also required the least computational time among the tested models. Additionally, we developed a **Multi-Layer Perceptron (MLP)** neural network to evaluate its suitability for the task, but despite its higher computational demands, it did not surpass the QRF in either accuracy or efficiency. While another type of deep neural network called **Long Short-Term Memory (LSTM)** networks are widely regarded for sequence-to-sequence modeling due to their inductive bias toward capturing long-term dependencies, they come with significant computational costs for training and hyperparameter tuning. Given our focus on developing robust, generalizable, and lightweight models for streamflow forecasting, we chose not to include deep architectures like LSTM in our model suite.

Given the time constraints and result submission requirements, we selected the QRF model for its robustness, suitability for probabilistic forecasting and computational efficiency, and focused our efforts on optimizing it for the task at hand.

4.3 Model Optimization

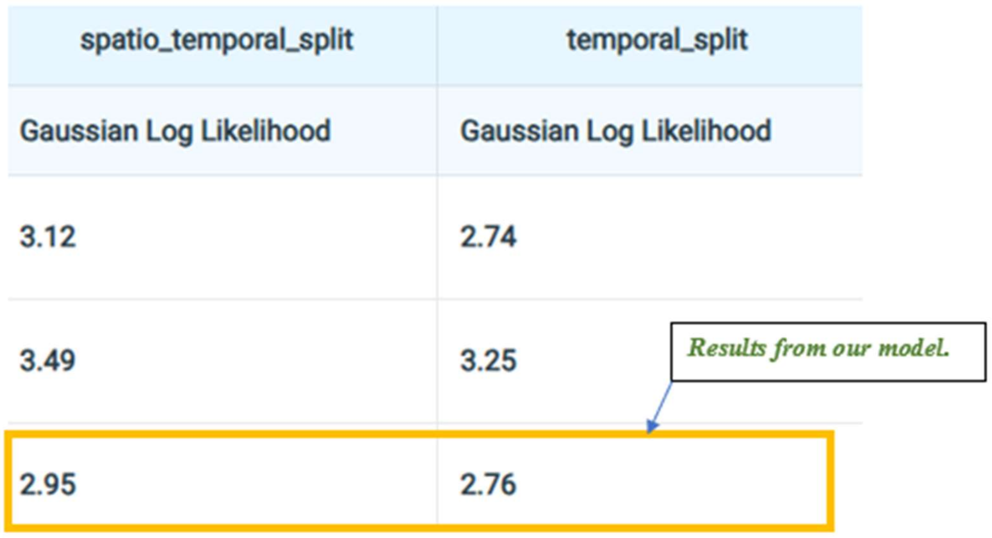
We performed **hyperparameter tuning** to optimize the model's performance. For this, again used our *train* and *validate* split from the training data. We experimented with different combinations of n_estimators, max_depth, and min_samples_leaf. To ensure **reproducibility**, we set a constant random seed of 42. The models were evaluated using the **Non-Negative Gaussian Log-Likelihood (NNGLL)** score, with an additional focus on capturing a high percentage of key streamflow events. Our tuning process began with a broad search over a wide **range of hyperparameter** values to map the overall response surface, followed by a more focused search within narrowed ranges where performance was most promising. This approach allowed us to identify hyperparameter settings under which the model consistently performed well.

Hyper parameter	Range
Number of estimators	Sequence (0, 2000, 100)
Max Depth	Sequence (0, 100, 5)
Min Sample Leaf	Sequence (0, 100, 5)

At this point, we had already missed the deadline for the mini-challenge, which focused on improving predictions for unseen stations. As a result, we shifted our focus to **prioritizing model performance under spatial and temporal splits** for station both in training and test. Based on our internal evaluation, we selected the best-performing hyperparameter set and submitted the corresponding predictions to the online evaluation platform. According to the leaderboard metrics, the combination of **n_estimators=1000**, **max_depth=25**, and **min_samples_leaf=25** achieved the best results.

5. Results

Based on our optimized model, we had a score of 2.95 for the spatio_temporal_split and 2.76 for the temporal split. The lower value is considered better.



spatio_temporal_split	temporal_split
Gaussian Log Likelihood	Gaussian Log Likelihood
3.12	2.74
3.49	3.25
2.95	2.76

Results from our model.

6. Our Model Evaluation

We have previously outlined our modeling strategies and methodological choices that led to the performance reported on the leaderboard and submitted for Phase 1 of the competition. Below, we provide a summary of our work across four key sections: Model Description & Novelty, Explainability, Robustness, and Frugality.

• Model Description & Novelty:

Our approach combined conventional machine learning techniques with careful tailoring to the competition's specific objectives. We began by thoroughly reviewing the project goals, baseline code, and scoring metrics. These elements informed an iterative development process in which we optimized our model within the provided time and our computational resource constraints.

Since we missed the mini-challenge, which focused on prediction for unseen sites, we concentrated our efforts on achieving strong generalization performance for the seen stations. This goal shaped many of our modeling decisions throughout the development process.

One key innovation that we believe significantly improved model performance was effective feature engineering. The dataset included a large number of input features, many of which appeared redundant or highly collinear. To address this, we began by analyzing the feature importance plot, which highlighted the predictive contribution of each feature in the decision tree model. Based on this analysis, we retained models only those features that demonstrated substantial predictive power in streamflow estimation.

Likewise, recognizing the importance of building not just a high-performance model on training dataset but also a robust and generalizable one, we adopted a dual perspective - both data-driven and domain-informed. After identifying a core set of impactful features from the feature importance plot, we curated combinations of variables that represented key watershed characteristics and meteorological factors known to play a key role in hydrologic response. We tested multiple such combinations and selected the one that resulted in the best validation performance.

This domain-informed approach to feature selection was a critical step. Rather than allowing tree-based models to overfit by exploiting spurious correlations in a high-dimensional feature space, we carefully curated inputs that were both statistically relevant and hydrologically meaningful. This helped improve the model's out-of-sample performance and reinforced its ability to capture the true underlying processes governing streamflow.

• Explainability:

In our analysis, we developed decision tree-based models, which are not only powerful in predictive tasks but also widely appreciated for their inherent interpretability capabilities. One major advantage of these models is their ability to quantify feature importance by evaluating how much each feature contributes to information gain while growing decision trees across all the ensembles. This allows us to generate feature importance plots and gain a transparent understanding of how the model makes its predictions and which input variables drive the forecasted streamflow values.

Several insightful patterns emerged from our analysis. We observed that the model relied more heavily on meteorological variables than on static watershed attributes. This result is expected given that our model was primarily trained and validated on the same set of stations. When training and test data originate from the same locations, static features tend to offer less additional information compared to dynamic inputs. However, we recognize that static attributes play a critical role in generalization, especially in transfer learning tasks or when making predictions for unseen stations, where they help encode long-term watershed behavior.

Additionally, lagged variables also emerged as highly important. This aligns with hydrological understanding, as streamflow is inherently an autoregressive process - past observations carry valuable information about future conditions. Incorporating lagged inputs thus served as a meaningful and effective strategy for improving forecast accuracy.

Overall, the interpretability of our model not only helped us validate its behavior but also allowed us to align its predictive mechanisms with physical hydrologic principles.

• Robustness:

The robustness of our model is evidenced by its strong performance on the NNGLL score for both spatial and temporal folds. A major contributing factor to this robustness was the representativeness and balance of the training dataset provided. On our end, we took additional care to avoid overfitting by performing careful feature selection – guided by predictive power and domain understanding, applying balanced hyperparameter tuning, and systematic evaluation of model performance on both training and validation sets. This approach ensured consistent performance and generalizability.

• Frugality:

Throughout the modeling process, frugality remained an important guiding principle. Our objective was to develop a robust model that generalizes well to out-of-sample data while remaining lightweight and easy to deploy in operational settings.

With this in mind, we carefully evaluated our modeling options and found that Quantile Regression Forests (QRF) struck an ideal balance between computational efficiency and predictive performance. While deep learning models such as LSTMs or deep MLPs can sometimes yield superior accuracy, they come with significant computational overhead, making them less suitable given our constraints and goals.

Alternative approaches like MAPIE with LightGBM (LGBM) offered faster runtimes, but they fell short in terms of accuracy - particularly when evaluated across multiple forecast lead times. On the other hand, ensemble-based methods, though generally accurate, were highly resource-intensive and time-consuming to train and tune across multiple scenarios.

Given these trade-offs, we selected QRF as our base model. It provided strong performance across forecast lead times, required comparatively low computational resources, and natively supported probabilistic forecasting, which was central to our task. This choice enabled us to simulate results for different scenarios quickly, test various feature configurations, and optimize performance without exceeding available time or resource constraints.

Summary

Step	Description
1. Baseline Model Evaluation	Assessed performance of provided machine learning models to establish initial benchmarks and understand forecast behavior.
2. Overfitting Diagnosis	Identified high feature dimensionality as a cause of overfitting and poor generalization to new regions/times.
3. Feature Importance Ranking	Used Random Forest to rank predictors by global importance across the entire training dataset.
4. Iterative Feature Subset Creation	Built models using incrementally larger subsets of top-ranked features to test their impact on performance.
5. Evaluation Framework Design	Created custom train-test splits with spatial and temporal separation to simulate generalization to unseen conditions.
6. Model Selection for Experimentation	Chose a fast, lightweight model (XGBLinear) for testing multiple feature sets efficiently.
7. Optimal Feature Set Selection	Identified the best-performing subset of features for the hardest prediction target (Week 4 streamflow).
8. Final Model Choice	Selected the most effective baseline model (QRF) for continued development and final predictions.
9. Hyperparameter Optimization	Tuned key model parameters using validation performance to improve robustness and accuracy.
10. Submission and Evaluation	Submitted predictions based on the optimized model and feature set for leaderboard scoring.