Rainfall Prediction in Bissau

Karine Alhakim
karine.alhakim@psl.eu

Leo Zi-You Assini
leo-zi-you.assini@psl.eu

Gašper Azinovič
gasper.azinovic@psl.eu

Serge Bressloff
sergei.bressloff@psl.eu

Cenzo Poirier De Carvalho
cenzo.poirierdecarvalho@psl.eu

Abstract

This study explores the effectiveness of machine learning in predicting daily rainfall in Bissau, Guinea-Bissau, a region characterized by the highly seasonal West African monsoons. Using a 50-year dataset (1975–2024), we compared several models ranging from a simple Logistic Regression baseline to a more complex time series model, SARIMAX. The main challenge was to introduce information about seasonality into the simpler models, which by default only take into account current features. Our final results found that even a linear approach achieved a strong ROC AUC of 0.92. Our results suggest that combining historical persistence with seasonal indicators yields similar forecasting precipitation to time-series models in monsoon-dependent regions.

1 Introduction

In this project, we investigate the feasibility of predicting daily rainfall in a geographically small and climatologically well-defined region: Bissau, the capital city of Guinea-Bissau in West Africa. Bissau exhibits a highly seasonal rainfall regime dominated by the West African monsoon, with distinct wet and dry seasons and relatively consistent annual patterns. This strong seasonality makes it an ideal testbed for evaluating machine learning–based precipitation prediction methods in a controlled setting.

Using fifty years of data (1975–2024), we frame rainfall prediction as a supervised learning problem based on daily meteorological variables. The primary goal is to compare the performance of several classes of predictive models:

- Standard machine learning models such as logistic regression, k-nearest neighbors, random forest and XGBoost
- A time-series–specific statistical model, Seasonal AutoRegressive Integrated Moving Average with eXogenous variables (SARIMAX)

By evaluating these models on the same dataset using a consistent temporal train–validation–test framework, this study aims to test whether simple Machine Learning models can capture trend and seasonality with engineered features as well as more specialised time-series approaches for local rainfall forecasting.

All code from this project can be found on github at: `https://github.com/LeoAssini/practical-ml-psl`.

2 Data

2.1 Data Source and Selection

The dataset is derived from the ERA5 reanalysis, which provides hourly global atmospheric variables on a regular latitude–longitude grid. For this study, data were extracted over a 5×5 grid centered on Bissau. To simplify the modeling task and focus on a single location, only the central grid cell, with coordinates: Latitude: 11.85° N, Longitude: −15.60° E, was used for the prediction.

Hourly values were aggregated to daily totals or daily means, depending on the variable, producing a continuous daily time series spanning 1975-01-01 to 2024-12-31. This results in fifty years of daily observations.

2.2 Variables

The following variables were used as model inputs and targets:

Table 1: Meteorological variables used in the study.

Variable	Description	Units
`datetime`	Observation date	–
`t2m`	2-meter air temperature	°C
`d2m`	2-meter dewpoint temperature	°C
`tcc`	Total cloud cover	%
`sp`	Surface pressure	hPa
`tp`	Total daily precipitation (target variable)	mm

2.3 Derived Features

To better characterise atmospheric moisture conditions, relative humidity (RH) was computed using the Magnus formula:

$$RH = 100 \times \frac{\exp\left(\frac{17.625\,T_d}{243.04+T_d}\right)}{\exp\left(\frac{17.625\,T}{243.04+T}\right)} \tag{1}$$

where T is the air temperature in $^\circ$C and T_d is the dewpoint temperature in $^\circ$C.

2.4 Target Definition and Class Imbalance

Daily precipitation was converted into a binary rainfall indicator defined as

$$y = \begin{cases} 1, & \text{if } tp > 0.1 \text{ mm (Rain)}, \\ 0, & \text{if } tp <= 0.1 \text{ mm (No Rain)}. \end{cases}$$

This formulation emphasises the prediction of rainfall occurrence rather than exact precipitation amounts. The resulting dataset exhibits a strong class imbalance:

Table 2: Distribution of rainfall classes in the dataset.

Class	Count	Percentage
No Rain (0)	16,284	89.2%
Rain (1)	1,979	10.8%

Such imbalance motivates the use of appropriate evaluation metrics and modeling strategies that are robust to skewed class distributions.

2.5 Train–Validation–Test Split

To preserve the temporal structure of the data and avoid information leakage, a chronological split was used:

- Training set: 70% (earliest observations)
- Validation set: 20% (subsequent observations)
- Test set: 10% (most recent observations)

This reflects a realistic forecasting scenario in which models are trained on historical data and evaluated on future, unseen periods.

3 Feature Engineering

Since standard Machine Learning models do not capture trend and seasonality, features were engineered from the basic meteorological data points for this purpose.

3.1 Lags

The most intuitive time period for predicting the weather is the past few days. Hence, lags were created from 1, 2, 3 and 7 days prior. To further enrich the models' context with seasonality, we added yearly lag because weather patterns in Bissau show heavy seasonality.

3.2 Rolling averages

To gain insight into whether the current week or month is generally dry or wet, averages over these periods were also calculated from the basic meteorological features.

4 Logistic Regression

4.1 Model Selection

We used Logistic Regression as a baseline to see if Bissau's rainfall could be predicted using a simple linear combination of features. While other models were used for their ability to handle non-linear patterns, the Logistic Regression model serves as a "sanity check". It tells us if the expensive computational power of ensemble trees is actually necessary or if the engineered features (like the 365-day lags) are doing the heavy lifting. Since linear models are sensitive to varying feature scales, we ran the data through a StandardScaler first. We also applied the class_weight='balanced' setting to handle the fact that it only rains about 10% of the time in Bissau. This prevents the model from just guessing "no rain" every day to inflate its accuracy.

4.2 Methodology

The model was built using a Scikit-Learn pipeline. This ensured that the scaling parameters from the training set (1975–2020) were correctly applied to the test set (2021–2024) without data leakage. Once the model generated a probability score, we used a standard 0.3 threshold to classify days as "Rain" or "No Rain." In a climate like Bissau's, a 0.5 cutoff is often too conservative and misses actual rain events. By lowering the threshold, we improved the model's sensitivity (Recall) to ensure that we catch more rain events even if it leads to a few more false alarms.This allowed us to calculate the exact same metrics (F1, Precision, AUC) used for the more complex models.

4.3 Feature Importance Analysis

The primary advantage of using a linear baseline is its transparency. By extracting the coefficients from the model, we can see exactly which variables drive the rainfall forecast in Guinea-Bissau. The most significant predictors were:

- **Persistence (Lag 1)**: This was consistently the most powerful predictor. If it rained yesterday, the model significantly increases the probability of rain today. This better reflects the cluster nature of the Bissau's West African climate where rain often persists over several days.
- **Seasonality (Month Encoding)**: The model assigned heavy negative coefficients to December through April, effectively penalizing any rain predictions during the dry season, regardless of other weather readings. Conversely, August and September had the highest positive weights
- **Atmospheric Indicators**: Beyond time-based features, relative humidity and **total column water vapor** were the strongest meteorological drivers. The model correctly identified that high moisture content is a prerequisite for rain, while higher surface pressure was treated as a signal for clear, dry skies.

5 Random Forest

We decided to use Random Forests as a simple model with strong flexibility and applicability to all sort of Machine Learning problems. For random forests to apply to Time Series problems, we need to create a sliding window using the following variables: the five base features and 1-7 day lag for those features. This totals 40 variables, remaining relatively simple, but allowing for significant context.

5.1 Hyperparameter Tuning

The hyperparameter tuning was done using the TimeSeriesSplit and RandomizedSearchCV from the sklearn library. These allow to try (at random) many combinations of parameters, while ensuring no data leakage thanks to the lack of shuffling in TimeSeriesSplit. The grid of parameters had 3 options for each concerned parameter. After a total of 150 fits, 3 folds for 50 candidates, the best parameters are:

- number of estimators = 300
- minimum number of samples per split = 5
- minimum samples per leaf = 4
- maximum amount of features to consider per split = sqrt
- maximum depth = 20
- class weight = balanced

The results of this tuning show that this problem requires large forests (high number of estimators, high depth, high samples per leaf). As expected, the balanced class weight is more adapted to our problem given the class imbalance. To check that even larger forests where not necessarily needed, we checked with more estimators and the metrics decreased slightly, showing that 300 estimators seems to be a good balance between large forests and not overfitting in any way.

5.2 Model Analysis

The model seems to mainly find importance in cloud cover and humidity, which coincides with our expectations. The lags used are mostly the dewpoint and slightly the cloud cover.

6 XGBoost

6.1 Model Selection

XGBoost (eXtreme Gradient Boosting) was selected for this classification task due to its proven effectiveness on tabular data with mixed feature types. The algorithm constructs an ensemble of decision trees sequentially, where each tree corrects the errors of its predecessors through gradient descent optimization. This approach is particularly well-suited for capturing the complex, nonlinear relationships between meteorological variables and rainfall occurrence.

To maximize model performance, extensive feature engineering was performed, resulting in 57 predictive features across four categories:

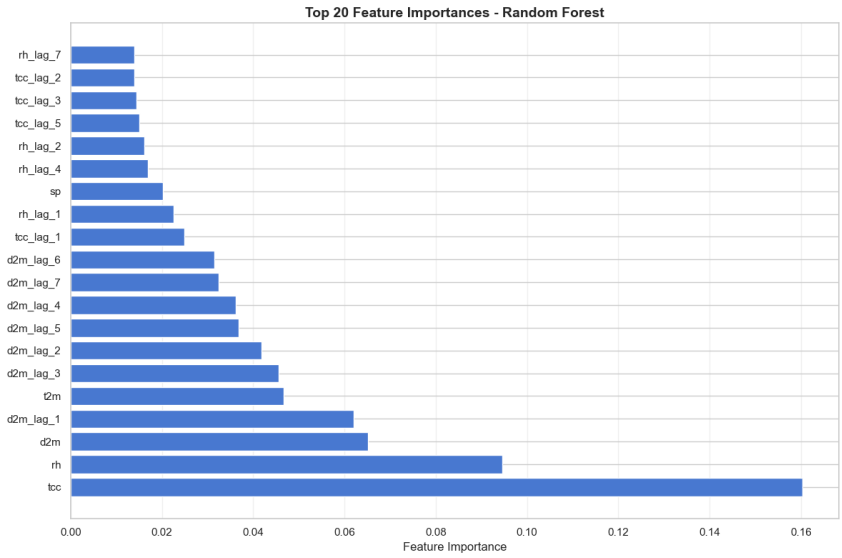

Figure 1: Top 20 feature importances for Random Forest.

- **Base meteorological features** (5): temperature, dewpoint, cloud cover, surface pressure, relative humidity
- **Lag features** (25): values from 1, 2, 3, and 7 days prior for each base variable
- **Rolling statistics** (20): 7-day and 30-day rolling means and standard deviations
- **Yearly features** (10): 365-day lag and 365-day rolling mean to capture annual seasonality

To address the substantial class imbalance (89.2% no-rain vs. 10.8% rain), the `scale_pos_weight` parameter was set to the ratio of negative to positive samples (approximately 8.2). This parameter increases the penalty for misclassifying rain days during training, effectively treating each rain observation as if it appeared 8.2 times in the dataset. Without this adjustment, the model would be biased toward predicting "no rain" for all days, achieving high accuracy (89%) while failing to identify actual rainfall events.

6.2 Hyperparameter Tuning

Hyperparameter optimization was conducted using RandomizedSearchCV with TimeSeriesSplit cross-validation (3 folds) to preserve temporal ordering and prevent data leakage. The search explored 100 random parameter combinations from a grid including tree depth, learning rate, regularization terms, and subsampling ratios. The F1 score was used as the optimization criterion given the class imbalance. The best-performing configuration was:

Parameter	Value
`n_estimators`	300
`max_depth`	10
`learning_rate`	0.01

The low learning rate (0.01) combined with a large number of estimators (300) indicates that the model benefits from gradual, incremental learning. The great tree depth allows for complex feature interactions.

6.3 Feature Importance Analysis

Feature importance analysis revealed that precipitation-related lag features dominate the predictive signal, indicating strong temporal autocorrelation in rainfall patterns (Figure 2). The mean participation of the last 7 days (`tp_rollmean7`) emerged as the single most important predictor, followed

by other short-term precipitation lags and rolling statistics. This finding aligns with meteorological intuition: rainfall events in tropical monsoon climates often persist over multiple consecutive days.

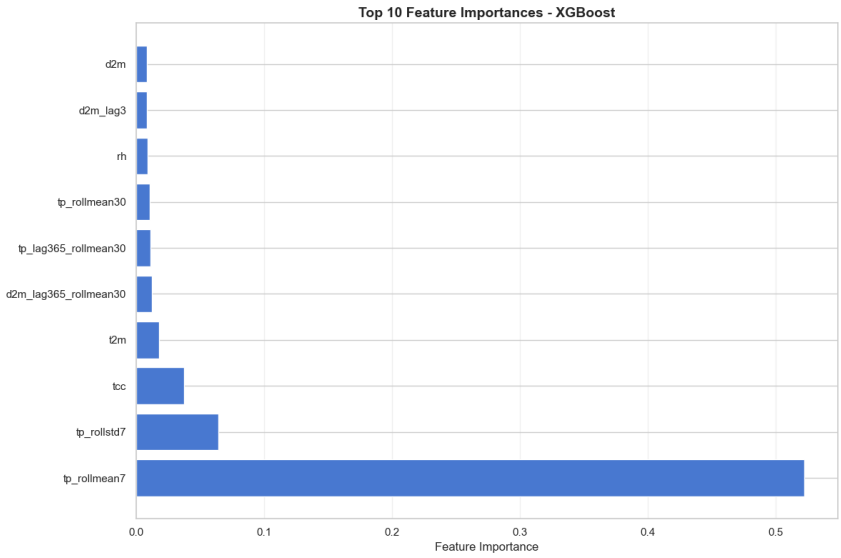

Figure 2: Top 10 feature importances for the XGBoost model. Precipitation rolling features dominate, with `tp_rollmean7` (mean participation previous 7 days) being the most predictive feature.

7 K-Nearest Neighbours

k-nearest neighbors (k-NN) algorithm is not the most commonly used in time-series analysis. Usually statistical models such as the other models used in this report are preferred. However k-nn has shown promising results in the past for various situations: river flow predictions [1], predicting financial markets [2] and forecasting the next day electricity market prices in Spain [3].

The strengths of this technique are the simplicity of the algorithm and the ability to capture non-linear patterns. The main idea of the implemented k-NN is to compare the current time period with similar time-frames in the past, take the k most similar and aggregate the past results to make a future prediction.

7.1 Methodology

Let the historical dataset be represented as a multivariate time series $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, where each $\mathbf{x}_i \in \mathbb{R}^d$ represents a vector of d meteorological features at time step i. We define a **feature window** W_t of length L ending at time t as:

$$W_t = [\mathbf{x}_{t-L+1}, \ldots, \mathbf{x}_{t-1}, \mathbf{x}_t] \tag{2}$$

7.1.1 Similarity Metric

For a current query window Q, we calculate the dissimilarity against a historical window H_i using the Mean Squared Error (MSE):

$$dist(Q, H_i) = \frac{1}{L \cdot d} \sum_{j=0}^{L-1} \|\mathbf{q}_j - \mathbf{h}_{i,j}\|^2 \tag{3}$$

The search space is restricted to historical windows occurring within the same seasonal month as Q, as well as the month before and after, to enforce physical consistency. This approach was inspired by the work done by Martínez et al. [3].

7.1.2 Forecasting

Let $\mathcal{N}_k$ be the set of indices of the k historical windows minimizing D. The binary rainfall forecast $\hat{y}_t$ is derived via majority voting:

$$\hat{y}_t = \underset{c \in \{0,1\}}{\operatorname{argmax}} \sum_{i \in \mathcal{N}_k} \mathbb{I}(y_i = c) \tag{4}$$

Other predictors were considered for this step, Troncoso Lora et al.[2] weighted the nearest neighbours, so the more similar they were the higher the weight. This was eventually discarded as it seemed to have limited impact on the accuracy.

To enable ROC-AUC analysis, we estimate the conditional probability of rainfall, denoted as $\hat{P}(y_t = 1|\mathbf{x}_t)$, rather than a hard class label. This is calculated as the fraction of the k nearest neighbors that belong to the positive class (Rain):

$$\hat{p}_t = \frac{1}{k} \sum_{i \in \mathcal{N}_k} \mathbb{I}(y_i = 1) \tag{5}$$

7.2 Hyperparameter Tuning

Though there are no parameters to be learned the are still some parameters which can be modified. The best-performing configuration was:

Parameter	Value
`k`	5
`window_size`	1

Interestingly the ideal window size was of 1. The results for 3 and 7 days were similar but still inferior, suggesting that for this model the local data is the most important to make the prediction.

8 SARIMAX

8.1 Autocorrelation Analysis

The strong seasonality of rainfall in Bissau suggests time-series–specific models may be well suited for this task. To investigate the temporal dependence of the daily rainfall, we examined the autocorrelation function (ACF) and partial autocorrelation function (PACF), which are commonly used to identify appropriate autoregressive (AR) and moving-average (MA) orders in ARIMA-type models.

The ACF revealed strong short-term autocorrelation at low lags as well as pronounced seasonal periodicity at multiples of approximately 365 days, consistent with annual monsoon-driven rainfall cycles. The PACF showed significant correlations at small lags, indicating that low-order autoregressive components could capture short-term persistence, while the strong yearly periodicity motivated the use of a seasonal ARIMA (SARIMA) formulation with an annual seasonal period.

8.2 SARIMA Formulation

Based on the observed seasonal structure, a SARIMA model of the form:

$$SARIMA(p, d, q) \times (P, D, Q)_m \tag{6}$$

with seasonal period *m=365* days was initially considered. Here, *(p,d,q)* denote the non-seasonal autoregressive, differencing and moving-average orders, and *(P,D,Q)* represent their seasonal counterparts.

However, the extremely large seasonal period required for daily data with annual seasonality (*m=365*) made SARIMA estimation computationally expensive. Even when the differencing order was set to

d=0 and the seasonal orders were restricted to small values of *P* and *Q*, model fitting remained impractically slow over the full 50-year dataset. This limitation is primarily due to the high-dimensional parameter estimation and matrix inversions required for long seasonal cycles. As a result, a full SARIMA model could not be reliably trained within reasonable computational constraints.

8.3 Fourier Feature Approximation of Seasonality

To retain the benefits of explicit seasonal modeling while avoiding the computational burden of SARIMA with a large seasonal period, seasonality was instead modeled using Fourier features. Fourier series provide a compact approximation of periodic behaviour and can efficiently represent long seasonal cycles using a small number of sine and cosine terms.

For a given time index *t*, Fourier seasonal features were constructed as:

$$S_k^{\sin}(t) = \sin\left(\frac{2\pi kt}{m}\right), \qquad S_k^{\cos}(t) = \cos\left(\frac{2\pi kt}{m}\right), \qquad k = 1, 2, \ldots, K, \tag{7}$$

This approach preserves explicit seasonal modeling in an ARIMA-style regression framework, while remaining computationally tractable.

8.4 Improvements

Model performance was improved by expanding the Fourier seasonal basis and incorporating other meteorological features. The number of Fourier harmonics was increased from *K=4* to *K=6*, allowing the seasonal representation to capture higher-frequency structure in the annual rainfall cycle. Exogenous meteorological variables (temperature, dewpoint, cloud cover, surface pressure and relative humidity) were included along with one-, two- and three-day lagged values. These predictors allow recent weather conditions to inform rainfall probability. Together, the expanded Fourier basis and lagged meteorological features substantially improved predictive performance.

9 Evaluation

All models were evaluated on the held-out test set consisting of the most recent 10% of observations. Because the dataset is highly imbalanced, overall accuracy was not used as a primary metric; instead, model performance was assessed using precision, F1 score, and ROC AUC, which more directly reflect skill on the minority rain class.

Model	Precision	F1	ROC AUC
Logistic Regression	0.625	0.5869	0.9243
Random Forest	**0.6667**	0.5123	0.9262
XGBoost	0.5817	**0.6122**	**0.9294**
k-Nearest Neighbours	0.6341	0.4469	0.8226
SARIMAX	0.4197	0.5657	0.9204

Table 3: Evaluation metrics on the test set (higher is better).

Across models, XGBoost achieved the best overall balance of performance, obtaining the highest F1 score and ROC AUC, indicating strong discrimination between rain and no-rain days and the best trade-off between precision and recall. Logistic regression and SARIMAX performed slightly worse but still showed high ROC AUC values, demonstrating that even relatively simple linear or classical time-series approaches can effectively separate the two classes when provided with appropriate features. Random Forest achieved relatively high precision but lower F1, suggesting a conservative tendency to predict rain less often while being correct when it does so. k-Nearest Neighbours performed worst overall, indicating difficulty in handling the high class imbalance and high-dimensional feature space. Overall, the results show that tree-based gradient boosting provides the strongest predictive performance for this task, while classical statistical and linear models remain competitive baselines.

10 Conclusion

After attempting to predict precipitation as a classification problem in Bissau with models of increasing complexity, we noticed that more complex models do not always signify higher metrics. Logistic regression itself sets an impressive baseline but more powerful models struggle to improve on it significantly. The answer to which model is best suited to this problem largely depends on what metrics we want to optimize. We considered the most important metric to be precision due to dry seasons making it simple to predict when rain would not fall. When our highest-precision model (Random Forest) predicts rain, it is correct 66.6% of the time. This is still underwhelming, especially given that its F1 score is quite low at 0.5123. We took Random Forest to be one of the simpler models and thus were quite satisfied to see that it held up to its algorithmically more complex counterparts in SARIMAX and XGBoost. Should also be remembered that the version used of SARIMAX was not the full model due to computational limitations, so we would expect better results had we used the complete model. In conclusion, modeling the data with more complex models such as SARIMAX, the seasonality features can also be modeled from the raw data and used as data points in the more basic models, outperforming them.

References

[1] Ehsan Ebrahimi and Mojtaba Shourian. "River Flow Prediction Using Dynamic Method for Selecting and Prioritizing K-Nearest Neighbors Based on Data Features". In: *Journal of Hydrologic Engineering* 25 (May 2020), p. 04020010. DOI: 10.1061/(ASCE)HE.1943-5584.0001905.

[2] Alicia Troncoso Lora et al. "Electricity Market Price Forecasting Based on Weighted Nearest Neighbors Techniques". In: *IEEE Transactions on Power Systems* 22.3 (2007), pp. 1294–1301. DOI: 10.1109/TPWRS.2007.901670.

[3] Francisco Martínez et al. "Dealing with seasonality by narrowing the training set in time series forecasting with kNN". In: *Expert Systems with Applications* 103 (2018), pp. 38–48. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2018.03.005. URL: https://www.sciencedirect.com/science/article/pii/S0957417418301441.