PAPER

# Time Series Analysis and Forecasting of Rainfall

**Maleke Chaker**[1]

[1] Department of Computer Science, Faculty of Sciences of Tunis, University of Tunis El Manar, 2092 Tunis, Tunisia.

## Abstract

**T**

his study initiates a comprehensive exploration of rainfall forecasting using time series analysis, specifically revisiting the work by Mithiya et al. (2020) titled "Time Series Analysis and Forecasting of Rainfall for Agricultural Crops in India: An Application of Artificial Neural Network." [7] The primary objective of this research is to enhance the forecasting accuracy of rainfall, a critical factor influencing agricultural productivity. Building on the methodologies presented in the original article, we applied several models, including Seasonal Autoregressive Integrated Moving Average (SARIMA) and Artificial Neural Networks (ANN), while introducing XGBoost as an additional model to evaluate their performance in predicting rainfall patterns.

Initial analyses revealed non-stationarity in the data, confirmed through ADF statistics and autocorrelation assessments. Subsequently, we optimized the SARIMA model using AutoARIMA, which demonstrated superior performance with the lowest Mean Absolute Error (MAE) of 79.57 compared to the other models tested. The ANN and XGBoost models were also evaluated, providing insights into their predictive capabilities and potential applications in agricultural risk management.

In addition to traditional forecasting methods, the integration of IoT technology holds promise for real-time data collection and analysis, further enhancing the accuracy of rainfall predictions. This work not only replicates the findings of Mithiya et al. but also suggests further refinements in model selection and hyperparameter tuning, emphasizing the importance of employing a diverse range of forecasting techniques, along with IoT data, to improve accuracy and reliability in agricultural rainfall predictions.

**Submitted:** 2024
**Accepted:** 2024
**Published:** 2024

**Vol.** 1, **No.** 1, 2024.
doi: N/A

## 1 Introduction

Rainfall plays a critical role in agricultural production worldwide, with significant regional variation affecting crop outcomes. As climate change continues to disrupt weather patterns, farmers, especially those relying on rain-fed agriculture, face mounting challenges. In regions such as Sub-Saharan Africa and India, where a large percentage of staple food production depends directly on rainfall, unpredictable precipitation can make or break harvests. In India, for instance, more than half of agricultural land remains un-irrigated and reliant on monsoon rains, making the sector highly vulnerable to erratic weather. This highlights the importance of accurate rainfall forecasting, which can empower farmers to make informed decisions about crop selection and management, reducing the risk of crop failure due to floods or droughts.

The integration of Internet of Things (IoT) technology and data-driven approaches to rainfall forecasting offers a promising solution for global agriculture. IoT devices can monitor real-time weather conditions, soil moisture, and other crucial factors, helping farmers optimize irrigation and improve overall productivity. By providing timely and precise rainfall predictions, especially in countries with limited water resources, forecasting tools can enhance agricultural resilience, ensuring food security in the face of an increasingly unpredictable climate.

## 2 Related Work

Here is a list of related works that have employed various forecasting methods to analyze rainfall patterns and other time series phenomena in different contexts:

- **Gulati Ashok et.al. (2013)**[2]: This study projected the impact of the robust 2013 monsoon rains on Agricultural Gross Domestic Product (GDP) growth in India. Using a log-linear model over the period 1996-97 to 2012-13, they found that 95% of agri-GDP variations could be explained by investments in agriculture, agricultural price incentives, and rainfall. The forecast suggested a growth rate of 5.2% to 5.7% for the agricultural year 2013-14, primarily driven by oilseeds, pulses, cotton, and cereals from rain-dependent regions.
- **Olatayo T. O. and Taiwo A. I. (2014)**[8]: The authors predicted rainfall in Ibadan, Nigeria, using 31 years of annual data (1982-2012). They employed Fuzzy Time Series (FTS), ARIMA, and Theil's regression. Based on Mean Absolute Error (MAE) and Root Mean Square Error (RMSE), the FTS model was found to outperform the other methods, making it a suitable tool for rainfall forecasting.
- **Karuiru Elias Kimani et al. (2016)**[5]: This study analyzed and forecasted precipitation in Kenya using both linear and non-linear models. The Time Lagged Feedforward Neural Network (TLFN) outperformed the Seasonal Autoregressive Integrated Moving Average (SARIMA) model, as indicated by lower Mean Absolute Deviation (MAD) and other diagnostic measures.
- **Kamath R.S. and Kamat R.K. (2018)**[4]: Using time series rainfall data from 2006-2016 in Iddukki, Kerala, the authors compared ARIMA, Artificial Neural Network (ANN), and Exponential Smoothing State Space (ETS) models. ARIMA was found to be the most accurate, based on Root Mean Squared Error (RMSE).
- **Pongdatu G. A. N. and Putra Y. H. (2018)**[9]: The authors compared SARIMA and Holt-Winter's Exponential Smoothing for forecasting retail clothing sales. While SARIMA provided more accurate results for short-term data, Holt-Winter's was better suited for seasonal time series. SARIMA $(1, 1, 0)(0, 1, 0)_{12}$ was concluded as the superior model.
- **Kistner Erica et.al. (2018)**[6]: The study explored the impact of temperature and precipitation fluctuations on specialty crops in the U.S. Midwest. They found that weather variability posed significant risks, with excess moisture being the most damaging factor. The study emphasized the need for crop-specific management tools and increased insurance coverage for specialty crop producers.
- **Renato Rossetti (2019)**[10]: This paper analyzed time series data to forecast console game sales in the Italian market using Exponential Smoothing and SARIMA. Based on measures of accuracy (MAPE, RMSE), SARIMA $(2, 1, 0)(1, 1, 0)_{12}$ was identified as the most reliable model for forecasting.

## 3 Rainfall Timeseries Data

### 3.1 Overview

The Rainfall Timeseries dataset[3] is used for forecasting rainfall patterns and is derived from the Prediction of Worldwide Energy Resources (POWER) project. This project provides meteorological data from

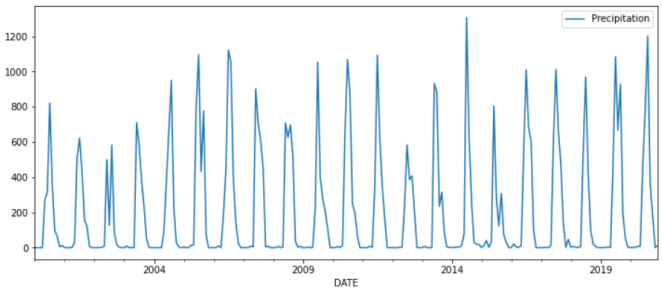


**Figure 1.** Rainfall Data Plot

NASA research to support renewable energy, building energy efficiency, and agricultural development.

### 3.2 Data Collection

Data was collected from the Power Data Access Viewer. The POWER meteorological data includes predictions and observations from NASA's GMAO MERRA-2 assimilation model. This dataset contains monthly frequency data for a specific latitude and longitude in Mumbai, covering the period from 2000 to 2020.

### 3.3 Dataset Characteristics

The dataset contains a total of 252 rows and 4 columns, It contains the following columns:


**Figure 2.** NASA's GMAO MERRA-2

| Column Name | Description |
|---|---|
| Year | The year of the observation |
| Month | The month of the observation |
| Day | The day of the observation |
| Specific Humidity | The specific humidity value |
| Relative Humidity | The relative humidity value |
| Temperature | The temperature value |
| Precipitation | The total monthly precipitation |

**Table 1.** Dataset characteristics of the Rainfall Timeseries data.

### 3.4 Data Exploration

To explore the dataset, we performed a seasonal decomposition of the rainfall data. A time series data is composed of three main components:

- **Trend:** The overall direction of the data over time.
- **Seasonality:** A periodic component that repeats itself within a particular time period.
- **Residuals:** The random fluctuations left over when the trend and seasonality have been removed.

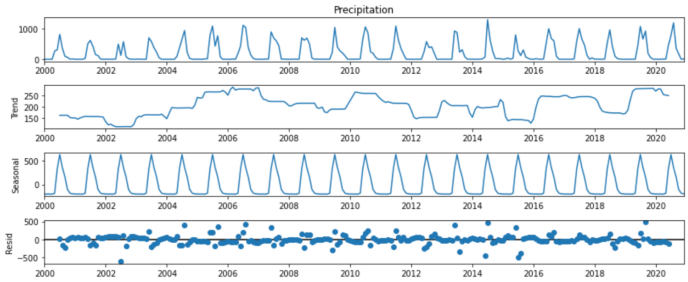


**Figure 3.** Seasonal decomposition of rainfall data.

We utilized seasonal decomposition to analyze the monthly precipitation data.

Figure 3 displays the seasonal decomposition of the precipitation data.

Looking at the plot, it is clearly visible that there is seasonality in the data, along with some underlying trend.

## 4 Data Preparation

In order to ensure the quality and relevance of the datasets used for training our models, we applied several preprocessing steps across both traditional statistical models and machine learning models.

### 4.1 Data Transformation

We combined the separate `Year`, `Month`, and `Day` columns into a single datetime column named `DATE`. This conversion allows for easier indexing and manipulation of the data, which is essential for time series analysis.

| | Year | Month | Day | Specific Humidity | Relative Humidity | Temperature | Precipitation |
|---|---|---|---|---|---|---|---|
| 0 | 2000 | 1 | 1 | 8.06 | 48.25 | 23.93 | 0.00 |
| 1 | 2000 | 2 | 1 | 8.73 | 50.81 | 25.83 | 0.11 |
| 2 | 2000 | 3 | 1 | 8.48 | 42.88 | 26.68 | 0.01 |
| 3 | 2000 | 4 | 1 | 13.79 | 55.69 | 22.49 | 0.02 |
| 4 | 2000 | 5 | 1 | 17.40 | 70.88 | 19.07 | 271.14 |

**Figure 4.** Data Before Transformation

| DATE | Specific Humidity | Relative Humidity | Temperature | Precipitation |
|---|---|---|---|---|
| 2000-01-01 | 8.06 | 48.25 | 23.93 | 0.00 |
| 2000-02-01 | 8.73 | 50.81 | 25.83 | 0.11 |
| 2000-03-01 | 8.48 | 42.88 | 26.68 | 0.01 |
| 2000-04-01 | 13.79 | 55.69 | 22.49 | 0.02 |
| 2000-05-01 | 17.40 | 70.88 | 19.07 | 271.14 |

**Figure 5.** Data After Transformation

As shown in Figure 4, the original dataset contains separate columns for Year, Month, and Day. After applying the transformation (Figure 5), these were combined into a single `DATE` column for better data manipulation and analysis.

### 4.2 Deleting Useless Columns

To improve model efficiency, we identified and removed irrelevant columns that do not contribute to rainfall prediction, such as: Specific Humidity, Relative Humidity, Temperature.

This refined the dataset to focus solely on relevant precipitation data.

### 4.3 Creating the Dataset with Lagged Observations (for Machine Learning Models)

We structured the dataset into sequences with lagged observations to facilitate time series model training.

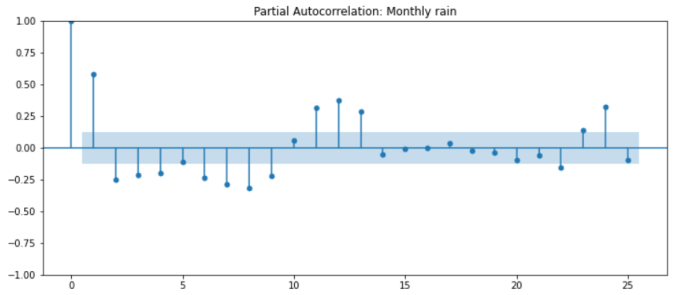


**Figure 6.** Partial Autocorrelation: Monthly rain

Using the PACF plot, we identified 13 lags (The point where the values drop off significantly) as optimal, enhancing the model's ability to detect trends and patterns.

### 4.4 Splitting the Data

We split the dataset into training and testing sets.

- For traditional models, the testing period is the last 12 months of observations.
- For machine learning models, we reserved 20% of the data for testing, ensuring effective model evaluation.

### 4.5 Normalization (for Machine Learning Models)

For machine learning algorithms, we applied normalization, scaling the data between 0 and 1 to improve algorithm performance.

The normalization parameters were fitted to the training data and applied consistently across the dataset.

## 5 Modeling

Before delving into the specific models used in this study, it is important to provide an overview of the key theoretical concepts behind time series forecasting. These theories form the foundation for both traditional statistical methods and modern machine learning techniques.

### 5.1 Time Series Forecasting Overview

Time series forecasting involves predicting future values based on previously observed data points. The goal is to model temporal dependencies and patterns, such as trends, seasonality, and noise.

In general, time series models can be categorized into two main types:

- **Statistical Models:** These models, such as Exponential Smoothing and SARIMA, rely on mathematical assumptions about the underlying data generation process. They often capture patterns like seasonality, trends, and autoregressive components.
- **Machine Learning Models:** These models, such as Neural Networks and XGBoost, are data-driven and leverage past observations to learn patterns without strict assumptions about the underlying distribution.

### 5.2 Traditional Statistical Models

In this section, we evaluate the traditional statistical models used for forecasting, including Exponential Smoothing methods and the SARIMA model, which are commonly applied to time series with trends and seasonality.

#### 5.2.1 Exponential Smoothing

Exponential Smoothing is a forecasting method [1] that assigns more weight to recent observations while diminishing the influence of older data. It is particularly useful for short-term forecasts, with different variants suited to various time series patterns.

**Simple or Single Exponential Smoothing (SES)** This model is used when the data has no trend or seasonality. It smooths fluctuations, assuming a constant level over time. However, since our dataset showed both trend and seasonality, this method was not sufficient.

**Double Exponential Smoothing** When a time series shows a trend but no seasonality, Double Exponential Smoothing is appropriate. It captures both the level and the trend but was still inadequate for our dataset due to the presence of strong seasonal patterns.

**Triple Exponential Smoothing (Holt-Winters)** Triple Exponential Smoothing extends the previous

models to handle data with both trend and seasonality. This method adjusts for the level, trend, and seasonal components, making it ideal for our precipitation data. By using an additive seasonal model, it captured the recurring patterns and provided accurate forecasts.

**Summary** The application of Triple Exponential Smoothing allowed us to model the complex dynamics of the precipitation time series, effectively capturing both trends and seasonal variations, leading to reliable forecasts.

### 5.2.2 Seasonal Autoregressive Integrated Moving Average (SARIMA)

The SARIMA model is an extension of the ARIMA model that incorporates seasonal components, making it highly suitable for time series data with both trend and seasonal patterns, such as precipitation. SARIMA captures both the non-seasonal and seasonal aspects of the data, by applying differencing to remove trends and by including seasonal autoregressive (AR) and moving average (MA) terms to account for repeating patterns at regular intervals.

**Components:**

- **AutoRegressive (AR) Component:** This captures the relationship between the current value and its past values.
- **Moving Average (MA) Component:** This accounts for the influence of past forecast errors (or residuals) on the current value.
- **Integration (I):** Differencing is applied to make the data stationary, especially useful when the data shows trends.
- **Seasonality:** Additional AR, MA, and differencing terms are added to capture recurring seasonal patterns.

**Application:**

1. **Stationarity Test:**

   The first step in our modeling process was to test for stationarity using the Augmented Dickey-Fuller (ADF) test. The test provided the following key statistics:

   - **ADF test statistic:** -2.4663
   - **p-value:** 0.1239
   - **Lags used:** 12
   - **Observations:** 239
   - **Critical values:**
     - 1%: -3.4580
     - 5%: -2.8737
     - 10%: -2.5733

   Given that the **p-value** is above the standard significance levels and the ADF statistic is greater than the critical values at all levels, we fail to reject the null hypothesis.

   This suggests that the data is **non-stationary** and contains a unit root, which was further confirmed by inspecting the Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) plots.

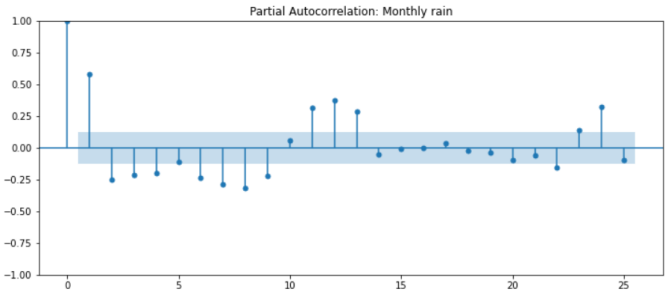


**Figure 7.** Partial Autocorrelation Plot

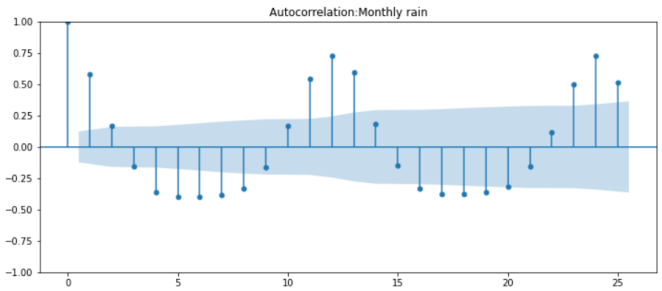


**Figure 8.** Autocorrelation Plot

These figures 7 and 8 showed slow decay, indicating that the data had both trend and seasonality components.

2. **SARIMA model:**

   Given the non-stationarity and the presence of seasonal patterns, we applied the Seasonal Autoregressive Integrated Moving Average (SARIMA) model, which is well-suited for such data.

Initially, we employed the hyperparameters suggested in the original article[7] to set up the SARIMA model for our time series data.

- `order` = $(0, 0, 0)$
- `seasonal_order` = $(5, 1, [0], 12)$

However, to further optimize model performance, we used the `AutoARIMA` tool from the `pmdarima` package. This automated process adjusted the initial configuration to find more optimal hyperparameters, leading to improved results.

The optimized SARIMA model parameters obtained from AutoARIMA were as follows:

- `order` = $(1, 0, 0)$
- `seasonal_order` = $(2, 0, [1], 12)$

## 5.3 Machine Learning Models

This section presents machine learning models tailored for time series forecasting. These models, such as Artificial Neural Networks (ANN), are capable of capturing complex, non-linear patterns in data, offering a powerful alternative to traditional statistical methods.

### 5.3.1 Artificial Neural Network (ANN)

We implemented an Artificial Neural Network (ANN) for forecasting with the following architecture:

- **Input Layer:** 64 neurons corresponding to input features.
- **Batch Normalization:** Normalizes inputs to stabilize training.
- **Dropout Layer:** 50% dropout rate to prevent overfitting.
- **Hidden Layer:** 32 neurons with ReLU activation for non-linearity.
- **Output Layer:** A single neuron for regression predictions.

The model was compiled using the Adam optimizer with mean squared error (MSE) as the loss function. We utilized callbacks such as EarlyStopping to prevent overfitting, ReduceLROnPlateau for dynamic learning rate adjustment, and TensorBoard for logging training performance.

The model was trained for up to 100 epochs with a batch size of 10.

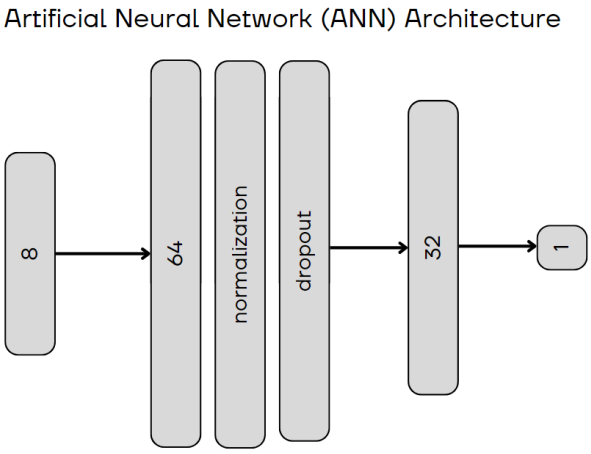


**Figure 9.** Artificial Neural Network (ANN) Architecture

### 5.3.2 XGBoost Model

This XGBoost model was proposed by the author and is not included in the original article [7]. To optimize the model, we employed Grid Search Cross Validation to identify the best hyperparameters.

The best parameters found were:

- **subsample:** 0.7
- **n_estimators:** 300
- **max_depth:** 3
- **learning_rate:** 0.01
- **colsample_bytree:** 0.7

The optimized model was then fitted and used for predictions on the test set.

# 6 Evaluation

This section evaluates the performance of traditional statistical models and machine learning models. Forecast errors are measured using the following metrics: Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). Each model is also visually compared using forecast plots.

## 6.1 Traditional Statistical Models

### 6.1.1 Exponential Smoothing

The Triple Seasonal Exponential Smoothing model was evaluated on the test data. The performance metrics are as follows:

- **MAE**: 110.05

- **MSE**: 37350.37
- **RMSE**: 193.26

The following plots depict the comparison between the predictions of the Triple Exponential Smoothing model and the actual data.

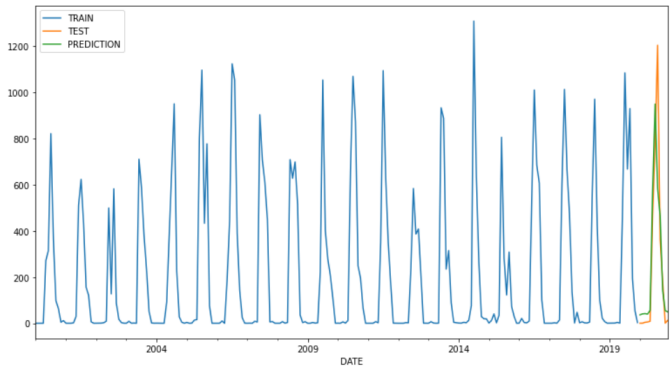


**Figure 10.** Triple Exponential Smoothing model predictions compared to actual data.

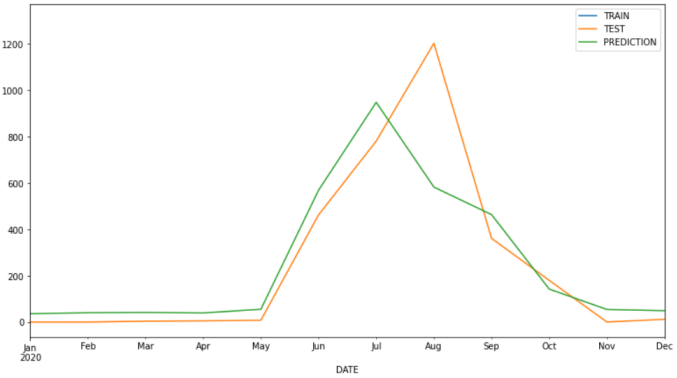


**Figure 11.** Zoomed-in view of Triple Exponential Smoothing model predictions compared to actual data.

The first plot shows the overall forecast performance, while the second zooms in to highlight the model's predictive behavior over a shorter time frame.

### 6.1.2 Seasonal Autoregressive Integrated Moving Average (SARIMA)

Two SARIMA variants were tested: the model suggested in the original paper and a variant using AutoARIMA.

**SARIMA Model Proposed by the Paper** :

- **MAE**: 90.14
- **MSE**: 45492.67
- **RMSE**: 213.29

The following plots illustrate the performance of the SARIMA model proposed in the paper.

The first plot displays the model's predictions compared to the actual data, while the second plot provides a closer look at specific time intervals to better assess the model's accuracy.

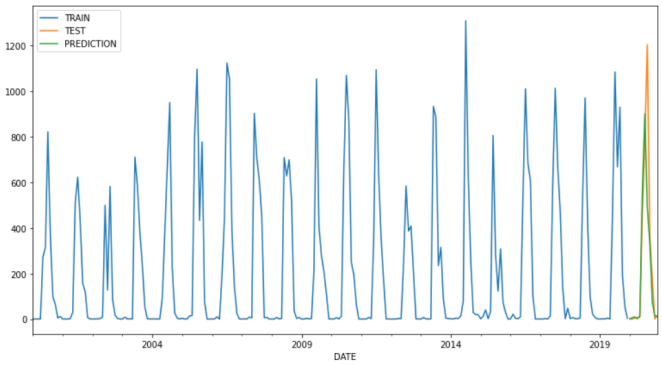


**Figure 12.** SARIMA (paper) model predictions compared to actual data.

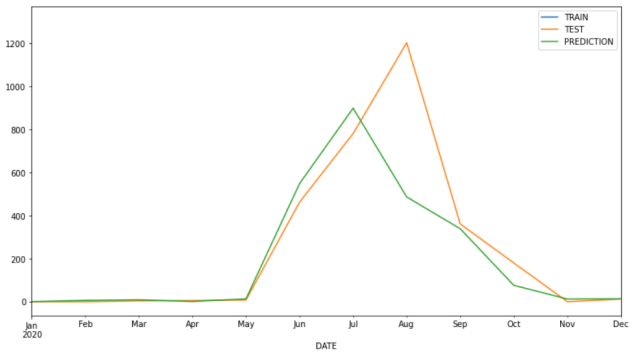


**Figure 13.** Zoomed-in SARIMA (paper) model predictions compared to actual data.

**SARIMA Model Optimized by AutoARIMA** :

- **MAE**: 79.57
- **MSE**: 37989.93
- **RMSE**: 194.91

Similarly, the plots below showcase the performance of the SARIMA model optimized by AutoARIMA.

The first figure presents the predictions alongside the actual data, while the second zooms in on particular segments to highlight the model's effectiveness.

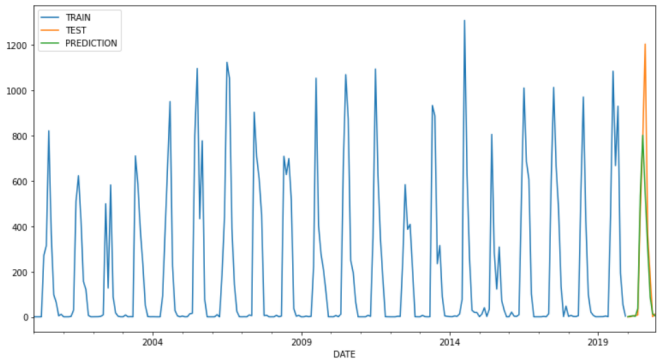


**Figure 14.** SARIMA (AutoARIMA) model predictions compared to actual data.

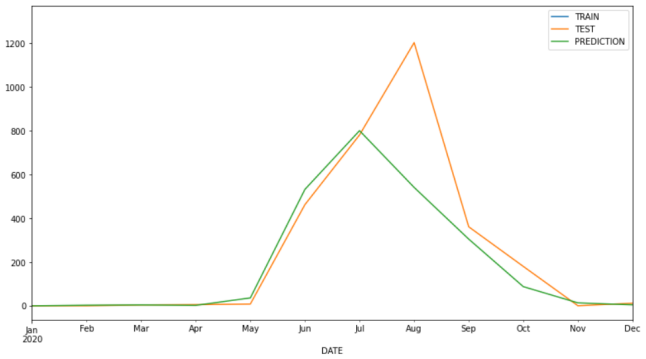


**Figure 15.** Zoomed-in SARIMA (AutoARIMA) model predictions compared to actual data.

## 6.2 Machine Learning Models

### 6.2.1 Artificial Neural Network (ANN)

The Artificial Neural Network (ANN) was evaluated on the same dataset. The performance metrics are as follows:

- **MAE**: 144.66
- **MSE**: 37158.28
- **RMSE**: 192.76

To visualize the performance of the ANN, the first figure illustrates the loss over epochs during the training process, indicating how well the model learned from the data.

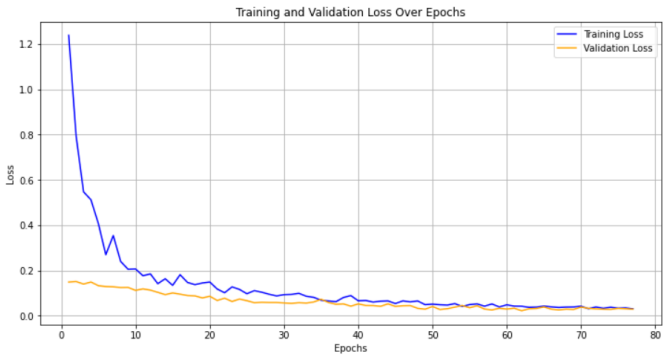


**Figure 16.** Loss over epochs for the ANN.

The second figure presents the predictions made by the ANN compared to the actual data, providing insight into the model's forecasting capabilities.

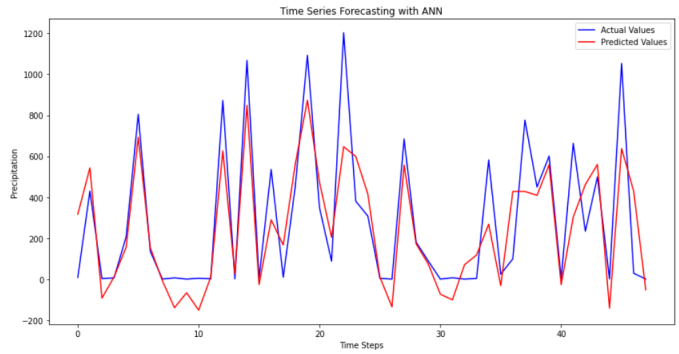


**Figure 17.** ANN model predictions compared to actual data.

### 6.2.2 XGBoost Model

The XGBoost model, proposed in this study, was not present in the original paper. After optimization through hyperparameter tuning (RandomizedSearchCV), this model showed competitive results, as indicated by the following performance metrics:

- **MAE**: 134.54
- **MSE**: 51179.29
- **RMSE**: 226.23

To provide a visual comparison of the model's performance, the following figure illustrates the predictions made by the XGBoost model alongside the actual data:

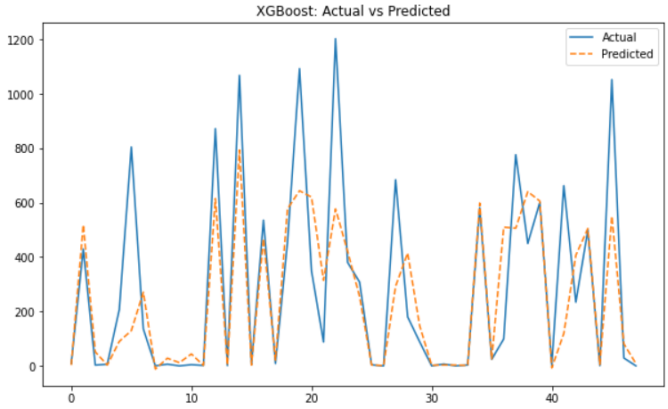


**Figure 18.** XGBoost model predictions compared to actual data.

## 7 Model Comparison

To better understand the performance of the various forecasting models employed in this study, the following table summarizes their respective Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) values:

| Model | MAE | MSE | RMSE |
|---|---|---|---|
| Exponential Smoothing | 110.05 | 37350.37 | 193.26 |
| SARIMA (Paper Model) | 90.14 | 45492.67 | 213.29 |
| SARIMA (AutoARIMA) | 79.57 | 37989.93 | 194.91 |
| ANN Model | 144.66 | 37158.28 | 192.76 |
| XGBoost Model | 134.54 | 51179.29 | 226.23 |

**Table 2.** Comparison of Model Performance Metrics

Among the models evaluated, **the AutoARIMA-optimized SARIMA model** achieved the lowest Mean Absolute Error (MAE) of 79.57, showcasing its effectiveness in predicting rainfall. However, it does not excel in Mean Squared Error (MSE) and Root Mean Squared Error (RMSE), where other models outperformed it.

Despite this limitation, the SARIMA model remains valuable in providing insights for managing agricultural risks related to rainfall.

## 8 Conclusion

In summary, this study effectively re-evaluates the forecasting of rainfall using various statistical and machine learning models, with a particular focus on improving accuracy compared to the original work by Mithiya et al. The AutoARIMA-optimized SARIMA model achieved the best performance in terms of Mean Absolute Error, indicating its effectiveness for rainfall prediction in agricultural contexts. However, while it excelled in MAE, other models demonstrated better performance in Mean Squared Error (MSE) and Root Mean Squared Error (RMSE).

Future research could explore the incorporation of additional external variables, such as climate indicators or soil moisture levels, to further enhance model robustness. Moreover, the integration of IoT technology for real-time monitoring of weather conditions and soil health could provide valuable data inputs, significantly improving model accuracy and decision-making. Experimenting with ensemble methods that combine the strengths of different models, alongside IoT data, may yield even more accurate predictions, providing critical insights for effective agricultural planning and risk management.

## 9 Acknowledgments

We would like to express our heartfelt gratitude to Professor Moez Hizem for his invaluable guidance and support throughout this study. His insights and expertise have been instrumental in shaping our research.

We also extend our appreciation to the Faculty of Science at the University of Tunis for providing the resources and environment necessary for our work.

## 10 Author Contributions Statement

Maleke Chaker designed the study, conducted the experiments, analyzed the results, and wrote the manuscript. This work serves as an initiation to revisit and expand upon the findings of the article [7]. All stages were reviewed and approved by M.C.

## 11 Data Availability

The Rainfall Timeseries dataset[3] used in this study is sourced from the Prediction of Worldwide Energy Resources (POWER) project, which provides comprehensive meteorological data for various applications, including renewable energy and agricultural development. This dataset includes monthly observations from NASA's GMAO MERRA-2 assimilation model, covering the period from 2000 to 2020 in Mumbai.

The dataset is publicly accessible through the Power Data Access Viewer. It consists of 252 rows and

includes the following columns: Year, Month, Day, Specific Humidity, Relative Humidity, Temperature, and Precipitation. Researchers and practitioners interested in rainfall forecasting can utilize this dataset to explore further analyses or model development.

## References


[1] Everette S Gardner Jr. Exponential smoothing: The state of the art. *Journal of forecasting*, 4(1):1–28, 1985.

[2] Ashok Gulati, Shweta Saini, and Surbhi Jain. Monsoon 2013: estimating the impact on agriculture. Technical report, Working Paper, 2013.

[3] Pooja Gupta. Rainfall timeseries data, 2022. Data retrieved from NASA's POWER project, containing meteorological data including specific humidity, relative humidity, temperature, and monthly precipitation from 2000 to 2020 in Mumbai. The dataset is public domain under the CC0 license.

[4] RS Kamath and RK Kamat. Time-series analysis and forecasting of rainfall at idukki district, kerala: Machine learning approach. *Disaster Adv*, 11(11):27–33, 2018.

[5] Elias Kimani Karuiru, George Otieno Orwa, and John Mwaniki Kihoro. Sarima versus time lagged feedforward neural networks in forecasting precipitation. *American Journal of Theoretical and Applied Statistics*, 12(3):359–364, 2016.

[6] Erica Kistner, Olivia Kellner, Jeffrey Andresen, Dennis Todey, and Lois Wright Morton. Vulnerability of specialty crops to short-term climatic variability and adaptation strategies in the midwestern usa. *Climatic change*, 146:145–158, 2018.

[7] Debasis Mithiya, Kumarjit Mandal, and Simanti Bandyopadhyay. Time series analysis and forecasting of rainfall for agricultural crops in india: An application of artificial neural network. *Research in Applied Economics*, 12(4):1–21, 2020.

[8] JN Onyeka-Ubaka, MA Halid, and RK Ogundeji. Optimal stochastic forecast models of rainfall in south-west region of nigeria. 2021.

[9] GAN Pongdatu and YH Putra. Seasonal time series forecasting using sarima and holt winter's exponential smoothing. In *IOP Conference Series: Materials Science and Engineering*, volume 407, page 012153. IOP Publishing, 2018.

[10] Renato Rossetti. Forecasting the sales of console games for the italian market. *Econometrics*, 23(3):76–88, 2019.