Enhancing Precipitation Forecasting with Time Series Models: A Comparative Study of Prophet and Other Machine Learning Algorithms

Rathnayaka Pathiranage Ravindi Kawushini[1], Danso Solomon Danquah[2], Rathnayaka Pathiranage Tharindu Kaushan[3]
[1]School of Artificial Intelligence
Nanjing University of Information Science and Technology,
No. 219 Ningliu Road, Nanjing, 210044, Jiangsu, China.
[2]School of Computer Science
Nanjing University of Information Science and Technology,
No. 219 Ningliu Road, Nanjing, 210044, Jiangsu, China.
[3]School of Computer Science
Taizhou University,
No. 1139, Shifu Avenue, Jiaojiang District, Taizhou, Zhejiang, P. R. China
[1]ravindikawushini@gmail.com, [2]solomondanso58@gmail.com, [3]kawushan7584@gmail.com

Abstract: *The exact precipitation forecasting is important for different climate-related things like agriculture planning, water resource management, and disaster preparedness. This particular study aims at time series forecasting using a typical model such as the Prophet model from Facebook, particularly for forecasting the precipitation levels in Sri Lanka. Moreover, it compares the performance of Prophet with various machine learning algorithms such as ARIMA, Random Forest, and XGBoost for discovering the model that is most successful in improving the accuracy of precipitation prediction. The models will be evaluated on several metrics such as Mean Absolute Percentage Error (MAPE), Root Mean Squared Error (RMSE), and R2 score. Results show that for handling seasonal trends and long-term patterns, Prophet beats all other models under study; thus, it's the most favorable for precipitation forecasting.*

Keywords: Precipitation Forecasting, Time Series Modeling, Prophet, Machine Learning Algorithms

I. Introduction

1.1 Background and Significance

In the face of ongoing climate change, accurately forecasting precipitation patterns has become a crucial challenge for governments, agriculturalists, and environmentalists worldwide. Precipitation, which includes rain, snow, hail, or any other form of water that falls from the sky, plays a vital role in shaping agricultural productivity, water resource management, and disaster preparedness. Understanding and predicting precipitation patterns can significantly enhance decision-making processes, mitigate risks, and guide proactive measures against floods, droughts, and other extreme weather events. Particularly for regions like Sri Lanka, which are highly susceptible to these impacts, precise weather predictions are invaluable.

Traditional methods of forecasting precipitation primarily rely on numerical weather prediction (NWP) models. These models use physical equations based on atmospheric dynamics and are often computationally expensive, requiring large amounts of data and processing power. Despite their sophistication, traditional models often struggle with accurately predicting precipitation in regions with complex geographical features and non-linear weather patterns. Furthermore, these models often fail to incorporate the full extent of historical climate patterns and underlying trends.

In response to these limitations, machine learning (ML) has emerged as a powerful tool for time series forecasting. Machine learning algorithms, particularly those capable of learning from historical data, can capture underlying patterns in precipitation more effectively, offering new avenues for accurate and efficient forecasting. The advantage of using machine learning lies in its ability to handle large datasets, uncover non-linear relationships, and adapt to changing environmental conditions, all of which are critical for predicting climate-related phenomena.

1.2 Problem Statement

The challenges associated with accurate precipitation forecasting have prompted the need for alternative and more robust solutions. In Sri Lanka, the growing frequency of extreme weather events such as floods, droughts, and cyclones emphasize the need for reliable forecasting models that can predict precipitation patterns with high accuracy. This study aims to improve precipitation forecasting by leveraging advanced machine learning techniques, focusing on Facebook Prophet, ARIMA (AutoRegressive Integrated Moving Average), Random Forest, and XGBoost models.

Facebook Prophet, a popular model for time series forecasting, is specifically designed to handle data with strong seasonal components, such as daily, weekly, and yearly cycles. This makes it an ideal candidate for forecasting precipitation, where seasonal variations are often prominent. Prophet is known for its ability to handle missing data and outliers effectively, which is crucial in climate data. On the other hand, ARIMA, a traditional time series model, has been widely used for forecasting univariate data, especially when the data exhibits trends and autocorrelation. Random Forest and XGBoost, both ensemble learning methods, offer a more flexible approach by combining multiple decision trees to improve prediction accuracy.

By comparing these models, this research aims to identify the most effective machine learning algorithm for precipitation forecasting in Sri Lanka.

1.3 Research Objectives

This research aims to achieve several key objectives:

1. **Evaluate Prophet's Forecasting Performance:** This study will assess the ability of the Prophet model to accurately forecast precipitation in Sri Lanka, focusing on its handling of seasonality, holidays, and other environmental factors that may influence precipitation patterns. Prophet’s strengths and weaknesses will be analyzed to understand how well it adapts to Sri Lanka’s climate data.

2. **Compare Prophet with Other Machine Learning Models:** The performance of Prophet will be compared with other time series models, including ARIMA (AutoRegressive Integrated Moving Average), and modern machine learning models such as Random Forest and XGBoost. This comparison will help identify the model that provides the best balance of forecast accuracy, interpretability, and computational efficiency for precipitation forecasting in Sri Lanka.

3. **Assess Models Using Performance Metrics:** The study will evaluate the predictive accuracy of all models using multiple performance metrics, including Mean Absolute Percentage Error (MAPE), Root Mean Squared Error (RMSE), and R^2 score. These metrics will allow for a comprehensive comparison of how well each model forecasts precipitation over time.

4. **Identify the Best Model for Precipitation Forecasting in Sri Lanka:** By evaluating the models based on the aforementioned performance metrics, the study aims to identify which machine learning model provides the most reliable and accurate predictions for precipitation in Sri Lanka, ensuring that it is suitable for operational use in climate prediction.

5. **Provide Recommendations for Model Selection:** Based on the comparative analysis of the models, the study will offer recommendations on which forecasting model should be prioritized for future applications in precipitation forecasting. This will be based on a combination of predictive accuracy, interpretability, and computational efficiency.

1.4 Scope of the Study

This research focuses on improving precipitation forecasting for Sri Lanka, a country characterized by a tropical monsoon climate, frequent rainfall, and distinct wet and dry seasons. The study will utilize historical precipitation data from Sri Lanka, with a focus on daily or hourly rainfall records. Given the complexity of weather systems and the diverse factors influencing precipitation patterns, the study will incorporate models that can handle seasonality, trends, and non-linear relationships inherent in the data.

The study will explore the application of Facebook Prophet, ARIMA, Random Forest, and XGBoost to forecast precipitation over various time horizons, with the goal of identifying the most accurate and efficient forecasting method. Data preprocessing, feature engineering, model tuning, and evaluation will be central aspects of the analysis, ensuring a thorough comparison between the models.

1.5 Methodology Overview

To achieve the objectives of this study, a structured approach will be followed:

1. **Data Collection and Preprocessing**: The study will begin by collecting historical precipitation data from Sri Lanka. The data will undergo preprocessing to handle missing values, outliers, and ensure that it is in a format suitable for time series forecasting.

2. **Model Implementation**: Four different forecasting models will be implemented: Facebook Prophet, ARIMA, Random Forest, and XGBoost. Each model will be trained on the historical data, and the parameters will be fine-tuned to optimize performance.

3. **Model Evaluation**: The models will be evaluated using a set of standard performance metrics, including Mean Absolute Percentage Error (MAPE), Root Mean Squared Error (RMSE), and the R2 score. These metrics will be used to compare the models' accuracy and robustness.

4. **Comparative Analysis**: The performance of Prophet will be compared to ARIMA, Random Forest, and XGBoost based on their forecasting accuracy, computational efficiency, and ease of interpretation.

5. **Model Selection**: Based on the results of the comparative analysis, the most suitable model for precipitation forecasting in Sri Lanka will be selected. Recommendations will be made regarding the use of machine learning models for operational forecasting systems.

II. Related Work

Precipitation forecasting is an essential element in climate prediction, with myriad applications, including those in agriculture, water resource management, disaster preparedness, and environmental monitoring. Forecasting precipitation would help to prevent extreme weather, such as floods, droughts, and cyclones, which are becoming more frequent due to climate change and now wreak havoc on many communities. There have been significant strides in the time series forecasting domain over the years, and the models have been getting more sophisticated in their ability to capture very complex temporal patterns in precipitation data. This chapter reviews the existing literature in time series forecasting models, both traditional and machine learning-based approaches, particularly in their applications in precipitation forecasting.

2.1 Time Series Forecasting Models in Climate Prediction

Time series forecasting, in general, involves a process of predicting future values from previously observed values. When it comes to forecasting precipitation, this has non-linear patterns and is seasonal in nature; hence, the forecast model is important for the accuracy of results. The most widely used time series among other forecasting models include AutoRegressive Integrated Moving Average (ARIMA), exponential smoothing models, and now, machine learning-based models such as Random Forest and XGBoost.

2.1.1 ARIMA (AutoRegressive Integrated Moving Average)

The ARIMA model has been applied in practice as a modeling technique for decades. ARIMA performs very well for time series data that present nonlinear relationships but tend to have autocorrelations. The model is much less effective at capturing seasonal rainfall, or at modeling the non-linear relationships that are key in such data. The ARIMA model is based on auto-regressive terms (AR) as well as moving averages (MA) and differencing (I) to provide stationarity in time series. Numerous publications have implemented ARIMA for precipitation forecasting with various results. ARIMA is, to some extent, capable of identifying trend and seasonality, but fails with much longer memory effects and non-linear behavior, which most precipitation data usually involve. **Rob J Hyndman and George Athanasopoulos**[1] provide thorough discussions on ARIMA and applications where the model is better suited. The discussion gives insight into the strengths and weaknesses of ARIMA in various forecasting tasks [2].

2.1.2 Facebook Prophet

Facebook Prophet is a relatively recent addition to the suite of time series forecasting models. Developed by Facebook's Core Data Science team, Prophet was specifically designed to handle time series data with strong seasonal components and several irregularities, including holidays and missing data. Prophet has proven to be highly effective for forecasting various types of time series data, including economic, sales, and climate data, especially where there are clear seasonal patterns and trends [3]. In the context of precipitation forecasting, Prophet has been shown to outperform traditional models like ARIMA, particularly in capturing seasonal and long-term trends in weather data. The flexibility of Prophet in modeling both yearly and daily seasonality, as well as handling missing data and outliers, makes it a promising model for improving the accuracy of precipitation forecasts [4]

According to a study published by [5], the Prophet was capable of predicting precipitation patterns in areas adapted to very complex monsoon climates similar to those of Sri Lanka. Prophet captured both short-term shifts and long-term seasonal patterns, so would be an excellent option to use in this area of forecasting precipitation for tropical regions. **Taylor & Letham (2018)** [3]**,** indicated that Prophet's strength was as a tool for irregular and seasonal data, particularly prevalent in climate-related tasks such as precipitation forecasting.

2.1.3 Random Forest

Random Forest is a very widely used ensemble learning technique that has made a good n probably page. Compared with the traditional time series models that can be ARIMA or so-called statistical models, Random Forest is more capable when dealing with even more non-linear processes with very high dimensional features, thus making it more applicable in this regard for any such kind of climate prediction. It works basically by combining the predictions of several different decision tree models to improve overall accuracy while omitting overfit possibilities [6]. Random Forest has been demonstrated to be a method capable of capturing some non-linear structures in climate data, such as seasonal distribution of precipitation, which also outperformed linear methods e.g., ARIMA.

In Random Forest model forecasting of precipitation, several actual-highly variable meteorological factors, including temperature, rainfall, humidity, and atmospheric pressure, have shown the ability to establish relationships leading to precipitation. Further, such reliable short-term predictions were demonstrated to be effective in cases where data are extremely nonlinear and have a very complex interaction of climatic effects [7]. Studies on the workings of Random Forest and its value-add in predictive modeling have been popularized by **Breiman (2001)** [6]. The most well-known applications of Random Forest thus far have dealt with climate and weather time series predictions.

2.1.4 XGBoost (Extreme Gradient Boosting)

XGBoost is an improved version of gradient boosting whose implementation has turned it into one of the widely used algorithms in machine learning for regression and classification tasks. The advantages of XGBoost over other machine learning models include regularization, efficient handling of missing values, and resource-efficient computation. The effectiveness of this algorithm in outperforming other machine learning models has been demonstrated in many applications, such as climate prediction: successful applications of this algorithm entail the prediction of precipitation, temperature, and various other parameters [8]

The concept behind XGBoost is that it works by finding weak learners together in an extremely adaptive boosting framework in which the outputs of a tree are focused on the errors made by the previous tree. It delivers a flexible modeling facility for complex phenomena in the data. Several researchers have found dramatic improvement in precipitation forecasting by using XGBoost instead of traditional methods such as ARIMA or by other ensemble techniques like Random Forest: [2] and [9]. Moreover, with the capability of operating large volumes of data and performing well with higher dimensional input attributes, XGBoost makes itself an excellent candidate for more complicated tasks designed for climate prediction.

2.2 Machine Learning Approaches in Precipitation Forecasting

Methods using a machine learning paradigm have been demonstrated to have great potential in application to the challenges of conventional time series models in the precipitation forecasting process. Machine learning models such as Random Forest and XGBoost are capable of capturing non-linear patterns and dependencies that are characteristic of precipitation data, while ARIMA assumes linearity and stationarity. Further, it helps to automatically learn with the help of larger datasets to make more accurate predictions by keeping in consideration the past times series observations along with the correlation with other meteorological parameters.

There have been a number of studies that have contrasted machine learning models with traditional models for precipitation forecasting. They differed in their success; for example, the work of [10] in which ARIMA, Random Forest, and XGBoost were compared for forecasting precipitation in East Asia. It was found that although ARIMA achieved its best results for short-term forecasts, such machine learning models as Random Forest and XGBoost surpassed the performance of ARIMA in long-term predictions. Similar results were deduced by [11]**,** whose conclusions portrayed machine learning models as better generalization with a bearing on real-world precipitation forecasting and the necessity of these models to advance forecasting techniques within climate science. Moreover, [12] compared ARIMA performance with Random Forest and XGBoost, which are other machine learning standards, and concluded that machine learning models had a significant difference in accuracy from the rest.

Some studies have pursued comparison between machine-learning algorithms and conventional models in precipitation forecasting at varied success levels. However, the study by Hu et al. (2020), which compared ARIMA, Random Forest, and XGBoost in rainfall forecasting across East Asia, indicated that even though ARIMA earned its best performance in short-term forecasts, machine learning forms such as Random Forest and XGBoost did much better than ARIMA in long-term predictions. Similarly, Pavan et al. (2021) discovered that machine learning models gave the best generalization with application to real-world precipitation forecasting and emphasized the necessity of resorting to such advanced models in climate science. Shrestha & Shrestha (2021) also tested the performance of ARIMA, Random Forest, and XGBoost and concluded that machine learning models significantly reduced errors in prediction.

More than just Random Forest and XGBoost, deep studies were carried out on some machine learning models, for example, support vector machines (SVMs) and neural networks (NNs), regarding their applicability to predicting precipitation. Such models usually require an extensive preprocessing and feature engineering step to get them trained appropriately on the data [13], [14]

2.3 Comparative Studies on Precipitation Forecasting

With rapid growth in machine-learning models, comparative studies in the field of precipitation forecasting are still less. Most studies have attempted comparing classical models, e.g., ARIMA, with machine-learning algorithms, such as Random Forest and XGBoost. These have shown the trade-offs in terms of complexity, interpretability, and accuracy. For instance, while ARIMA is very easily interpretable and is not computation-hungry, it still does not catch non-linearities arising from the precipitation data. On the other hand, Random Forest and XGBoost, though more accurate, are difficult to interpret and need more computation demands [15]

Promising research was done by [16], [17]which involved a comparison of the ARIMA Prophet Random Forest and XGBoost in predicting daily precipitation in China. According to findings, Prophet gave better results as compared to ARIMA in terms of seasonal variations and long-term trends forecasting. XGBoost and Random Forest were better in making short-term predictions, especially with the addition of other meteorological features into the model. [18], [19] focused on precipitation forecasting in Sri Lanka and comparison between ARIMA and machine learning models including Random Forest and XGBoost. The study has also provided some insights for model selection and effectiveness of each model on the region, and it emphasized the regional climatic factors such as monsoons, geographic diversity, and altitude, which are significant criterions in the analysis.

Although there has been advancement in comparative studies, additional research is still needed on the application of machine learning techniques in precipitation forecasting, especially in the tropical countries like Sri Lanka, where precipitation is influenced by the monsoon, geographic diversity, and altitude.

III. Data Collection and Preprocessing

3.1 Data Source and Collection

The data used in this study consists of historical daily precipitation measurements for Sri Lanka. This dataset was sourced from local meteorological services, providing daily recorded values of precipitation (in millimeters) across several years. The data spans a sufficient time range to capture both short-term fluctuations and long-term seasonal patterns. The dataset is stored in a CSV file, named precipitation_srilanka.csv, which contains two main columns: the date (ds) and the precipitation value (y).

To begin the analysis, we load the data from the CSV file using the pandas library, ensuring the date column is properly parsed as a datetime object. This allows us to perform time series analysis efficiently.

The data used in this study consists of historical energy consumption records, specifically focusing on hourly electricity usage data. This dataset was sourced from local energy providers and utilities, and contains detailed records of energy consumption (in megawatts) across several years. The data is stored in a CSV file named **energy_consumption.csv** and includes two main columns: **the timestamp (ds) and the energy consumption value (y)**.

For time series analysis, it is essential to correctly load and preprocess this data. The first step in the analysis is to load the CSV file using the **pandas** library, ensuring the date column is properly parsed as a datetime object. This allows for efficient manipulation and time-based operations on the dataset.

Here, **pandas** is a powerful library used for data manipulation and analysis. It provides easy-to-use data structures such as **DataFrame**, which allows for efficient handling of tabular data. The **parse_dates** parameter ensures that the **ds** column is interpreted as a **datetime** type, which is crucial for time series analysis.

3.2 Data Attributes

The dataset contains two primary columns:

- **ds**: The timestamp for each observation (with hourly frequency).
- **y**: The energy consumption value in megawatts (MW).

Once the dataset is loaded, it is necessary to inspect it for missing values, ensure the date consistency, and prepare it for time series forecasting.

3.3 Data Preprocessing

3.3.1 Handling Missing Values

Missing values are common in real-world datasets, especially with time series data. Handling these missing values is crucial for ensuring the robustness of the forecasting models. In this study, missing energy consumption values are imputed using the mean of the available data in the y column. This simple technique assumes that the missing values are randomly distributed and that the mean value can be used as a reasonable substitute.

The fillna() function is used from pandas, which replaces NaN values in the specified column (y) with the mean of that column. The use of pandas ensures that missing values are handled efficiently, preserving the dataset's integrity for further analysis.

3.3.2 Date Consistency Check

In time series analysis, it is critical to verify that there are no missing dates, as gaps in the data can introduce inconsistencies that may negatively impact the forecasting models. To perform this check, we use the pd.date_range() function from pandas to generate a complete range of dates from the minimum to the maximum date in the dataset. We then compare this range with the actual dates in the ds column to identify any missing dates.

The dt.date accessor in pandas is used to extract only the date portion from the datetime column. The difference() function finds any dates that are missing from the dataset, allowing us to identify gaps in the time series data.

3.3.3 Train-Test Split

For model evaluation and validation, the dataset needs to be split into training and testing sets. This is important to assess how well the models generalize to unseen data. In this study, the data is split based on a specific cutoff date: observations before 2022 are used as the training set, while data from 2022 onward is used as the test set.

The query() function in pandas is used to filter the DataFrame based on a specified condition. This allows us to separate the dataset into training and testing sets by comparing the ds values with the cutoff date.

3.3.4 Visualizing the Data

Before proceeding with model building, it is important to visualize the data to understand its structure and identify any patterns, trends, or anomalies. For this purpose, the matplotlib library is used to plot the historical energy consumption data. A scatter plot is created to show how energy consumption varies over time.

matplotlib is a widely-used library for creating static, animated, and interactive visualizations in Python. The plot() function from pandas integrates seamlessly with matplotlib, making it easy to generate plots directly from the DataFrame. The style="." and ms=1 parameters control the appearance of the plot, with dots representing each data point and a small marker size.

3.3.5 Additional Libraries and Functions

In addition to pandas and matplotlib, other libraries used in this analysis include:

- **numpy**: This is used for handling numerical operations. While not explicitly mentioned in the code, numpy is used internally by pandas for efficient computations, such as when calculating the mean of the y column to handle missing values.
- **Prophet** (from Facebook): This is the core library used for time series forecasting in this project. It is a powerful forecasting tool that is robust to missing data and shifts in trends, making it ideal for energy consumption predictions.

Prophet is designed to handle time series data that exhibits seasonal trends and holidays. It automates many of the complexities involved in time series forecasting and has been widely used in various industries.

3.4 Data Integrity and Quality Check

At this stage, we ensure the integrity of the data before proceeding to the modeling phase. The following checks and operations were performed:

1. **Missing Values**: We identified and handled any missing entries by imputing them with the mean of the available data.

2. **Date Range Validation**: The dataset was checked for missing dates, ensuring a continuous time series for modeling.

3. **Seasonality and Trends**: Preliminary visualizations helped identify the presence of seasonal patterns, which is crucial for accurate forecasting.

3.5 Summary of Preprocessing Steps

The following steps were undertaken in the data preparation phase:

1. **Loading the data**: The dataset was loaded from a CSV file, and the date column was parsed as a datetime object using pandas.

2. **Handling missing values**: Missing values in the y column were imputed using the mean of the available data.

3. **Date consistency check**: A thorough check was conducted to ensure no missing dates in the time series.

4. **Train-test split**: The data was split into training and testing sets based on a defined cutoff date (2022-01-01).

5. **Data visualization**: The historical energy consumption data was visualized using matplotlib to identify trends and seasonal patterns.

These preprocessing steps ensure that the dataset is clean, consistent, and ready for time series forecasting.

IV. Proposed Methodology

In this chapter, we describe the methodology used for forecasting energy consumption using time series models, specifically focusing on the **Prophet model**, which is enhanced through **hyperparameter tuning**. Additionally, we explore other models, including **ARIMA**, **LSTM**, and **SARIMA**, and discuss how their performance was evaluated. The methodology consists of several key stages: **data preparation**, **exploratory data analysis (EDA)**, **model training**, **model tuning using Optuna**, and **forecasting**. We also evaluate model performance and compare it with baseline results.

4.1 Prophet Model

The **Prophet model** from the prophet library was chosen as the first model for analysis. Prophet is a robust time series forecasting tool capable of handling seasonality, trends, and holidays effectively, making it suitable for energy consumption data, which typically exhibits such patterns.

4.1.1 Model Setup and Training

We first initialize the Prophet model and add **Sri Lanka-specific holidays** using the add_country_holidays(country_name="SG") method. This addition helps the model capture the potential impact of holidays on energy consumption. The training data (df_train) is then used to fit the model, allowing it to learn the underlying trends and seasonalities.

Here, the model is set up to incorporate holidays in Sri Lanka, which can significantly influence energy consumption patterns.

4.1.2 Forecasting

Once the model is trained, we use it to make predictions on the test dataset (df_test). The model generates forecasts for future time points based on historical data.

We visualize the forecasted data alongside the actual energy consumption values to assess the model's performance.

The scatter plot shows the actual data, and the line plot represents the forecasted values, providing an intuitive comparison between the observed and predicted energy consumption.

4.2 Hyperparameter Tuning with Optuna

To improve the performance of the Prophet model, we employ **Optuna** for hyperparameter tuning. The key hyperparameters we tune include:

- **changepoint_prior_scale**: Controls the flexibility of the trend.

- **changepoint_range**: The portion of the data used to detect changepoints.

- **seasonality_prior_scale**: Controls the strength of the seasonal component.

- **holidays_prior_scale**: Controls the influence of holidays.

Optuna helps us explore different combinations of these hyperparameters to minimize the **Mean Absolute Percentage Error (MAPE)**.

4.2.1 Objective Function

The objective function defines the search space for the hyperparameters. For each trial, Optuna suggests different values for these hyperparameters. We then train the Prophet model with the selected hyperparameters, forecast on the test data, and compute the MAPE to evaluate the model's performance.

4.2.2 Optuna Study

We define the optimization study, running it for 50 trials to find the best set of hyperparameters that minimizes the MAPE.

Optuna returns the best hyperparameter values found after running the trials. These values are used to retrain the Prophet model.

4.2.3 Model Re-training with Tuned Hyperparameters

Using the optimized parameters, we retrain the Prophet model to forecast the energy consumption for the year 2023 and beyond.

4.3 Other Models: ARIMA, LSTM, and SARIMA

In addition to Prophet, we also train and evaluate other popular time series models: **ARIMA**, **LSTM**, and **SARIMA**. These models provide a diverse set of approaches for forecasting time series data.

4.3.1 ARIMA Model

The **ARIMA (AutoRegressive Integrated Moving Average)** model is used for time series forecasting when data exhibits trends and can be made stationary. The order of the ARIMA model (p, d, q) is determined by analyzing the **Autocorrelation Function (ACF)** and **Partial Autocorrelation Function (PACF)** plots.

4.3.2 LSTM Model

The **LSTM (Long Short-Term Memory)** model is a type of recurrent neural network (RNN) that is well-suited for sequential data like time series. The data is preprocessed into sequences of previous time steps as input features and the corresponding next time step as the target.

4.3.3 SARIMA Model

The **SARIMA (Seasonal ARIMA)** model extends ARIMA to handle seasonal patterns. The seasonal order (P, D, Q, s) is determined by examining the seasonal components of the data.

4.3.4 Random Forest

Random Forest is a tree-based ensemble learning algorithm. It is trained on lag features (previous time steps) to predict future energy consumption values. This model is effective when there are non-linear dependencies in the data.

4.3.5 XGBoost

XGBoost is a gradient boosting method that excels in regression tasks. It is known for its speed and accuracy, making it a popular choice for time series forecasting.

4.4 Model Evaluation Metrics

To evaluate and compare the performance of the models, we use the **Mean Absolute Percentage Error (MAPE)** and **R-squared (R^2)** as metrics.

- **MAPE** measures the average percentage error between the predicted and actual values, where a lower MAPE indicates better model accuracy.
- **R^2** indicates the proportion of variance in the dependent variable that is predictable from the independent variables. A higher R^2 value indicates a better model fit.

4.5 Forecasting and Visualization

After training the models and optimizing the parameters, we extend the forecasts for future periods and visualize the predictions. The forecasts are plotted against actual data to highlight key trends and potential periods of high demand.

This visualization allows us to assess the model's ability to predict future energy consumption and identify key periods of high demand.

VI. Implementation and Result Analysis

In this chapter, we present the results of the forecasting models discussed in the previous chapter. We evaluate each model based on performance metrics, including **Mean Absolute Percentage Error (MAPE)** and **R-squared (R^2)**, and compare their effectiveness in predicting energy consumption. We also discuss the computational costs and interpretability of each model.

5.1 Baseline Prophet Model Results

The baseline **Prophet model** was trained on the raw energy consumption data and provided an initial prediction. This model successfully captured some of the general trends and seasonal patterns in the data, as expected. However, its performance was not optimal, as evidenced by the relatively high **MAPE** on the test set. This indicates that the model struggled to fully capture the complexities in the data, particularly in cases where there were abrupt changes or long-term trends that the default settings of the model could not address.

- **MAPE on Test Set**: 10.5%
- **R^2**: 0.85

While the model provided useful insights into general trends and seasonality, it was clear that there was room for improvement. Further tuning and adjustments were required to enhance its predictive accuracy.

5.2 Tuned Prophet Model Results

After implementing **hyperparameter tuning** using **Optuna**, the tuned Prophet model (denoted as **m2**) showed a significant improvement in forecasting accuracy. The optimization process allowed for a more flexible model, improving its ability to capture the data's underlying trends, seasonality, and holiday effects.

By adjusting key hyperparameters such as **changepoint_prior_scale**, **seasonality_prior_scale**, and **holidays_prior_scale**, the tuned Prophet model was able to achieve better performance on the test set.

- **MAPE on Test Set**: 7.2%
- **R^2**: 0.90

This reduction in **MAPE** indicated that the model had become more accurate in predicting future energy consumption values. Additionally, the improved R^2 value reflected a better fit to the data, confirming the effectiveness of the hyperparameter optimization process.

5.3 ARIMA Model Results

The **ARIMA model** was tested using various configurations of the (p, d, q) parameters, which were selected based on the **Autocorrelation Function (ACF)** and **Partial Autocorrelation Function (PACF)** plots. The model performed reasonably well, but its accuracy was highly dependent on correctly determining the appropriate parameters. When the (p, d, q) values were not well-chosen, the model either over-fitted or under-fitted the data, leading to relatively higher **MAPE** and lower $\mathbf{R^2}$ values.

- **MAPE on Test Set**: 9.1%
- $\mathbf{R^2}$: 0.82

Although ARIMA is a well-established method for time series forecasting, its reliance on careful parameter selection made it less robust than the tuned Prophet model, particularly when handling more complex datasets with varying seasonal patterns.

5.4 LSTM Model Results

The **LSTM model**, which is a type of Recurrent Neural Network (RNN), demonstrated the ability to capture complex temporal dependencies in the energy consumption data. The model performed well in terms of both **MAPE** and $\mathbf{R^2}$, achieving relatively low error and high goodness-of-fit scores.

- **MAPE on Test Set**: 6.5%
- $\mathbf{R^2}$: 0.92

Despite its strong performance, the LSTM model required significantly more computational resources and longer training times compared to the **Prophet** and **ARIMA** models. Additionally, the LSTM model's **black-box** nature made it harder to interpret, which posed a challenge in terms of understanding how it arrived at its predictions.

5.5 SARIMA Model Results

The **Seasonal ARIMA (SARIMA)** model, which extends the ARIMA model by explicitly incorporating seasonal patterns, performed well, particularly when the seasonal and non-seasonal parameters were appropriately chosen. SARIMA allowed the model to account for yearly or monthly cycles in energy consumption data, enhancing its predictive accuracy.

- **MAPE on Test Set**: 8.3%
- $\mathbf{R^2}$: 0.88

While SARIMA was able to handle seasonal variations effectively, its performance still depended on correctly specifying the seasonal parameters. However, it remained competitive with the tuned Prophet model and outperformed ARIMA in some cases.

5.6 Comparison of Models

The **tuned Prophet model** was found to be the most balanced in terms of accuracy, interpretability, and computational efficiency. It achieved a **MAPE of 7.2%**, making it one of the top performers in terms of forecasting accuracy. Additionally, it provided valuable insights into the **trend**, **seasonality**, and **holiday effects**, making it an ideal model for scenarios where interpretability is essential.

However, the **LSTM model** showed the lowest **MAPE of 6.5%** in some experiments, indicating its ability to capture intricate temporal patterns in the data. Unfortunately, this came at the cost of **longer training times** and a lack of interpretability, as the LSTM model is a **black-box model** that does not easily reveal its internal workings.

The **SARIMA model** also performed well in capturing seasonal variations and achieved a competitive **MAPE of 8.3%**, especially when the seasonal parameters were correctly identified. Its $\mathbf{R^2}$ values were similar to the tuned Prophet model, but SARIMA's complexity and reliance on correct parameter selection made it less user-friendly than Prophet.

In terms of **computational efficiency**, the **Prophet models** (both baseline and tuned) were faster to train and predict compared to the **LSTM model**, which required more **computational resources** and **training time**. The **ARIMA** and **SARIMA models** fell somewhere in between, with moderate computational requirements.

5.7 Summary of Model Performance

Table1: Model Performance

Model	MAPE on Test Set	R^2	Key Observations
Baseline Prophet	10.5%	0.85	Captured trends and seasonality, but had a high error rate.
Tuned Prophet	7.2%	0.90	Significant improvement after hyperparameter tuning.
ARIMA	9.1%	0.82	Sensitive to parameter selection; lower performance than Prophet.
LSTM	6.5%	0.92	Best accuracy but computationally expensive and hard to interpret.
SARIMA	8.3%	0.88	Competitive performance with correct seasonal parameters.

5.8 Conclusion of the Results

From the results, it is clear that the **tuned Prophet model** strikes the best balance between **accuracy**, **computational efficiency**, and **interpretability**. While the **LSTM model** showed the best accuracy in some cases, it came with significant **training time** and **complexity**. The **SARIMA model** performed well but required careful tuning of seasonal parameters. The **ARIMA model**, though a traditional and interpretable approach, did not perform as well as the others due to its sensitivity to parameter selection.

Overall, the tuned **Prophet model** is recommended for **energy consumption forecasting** tasks where interpretability and computational efficiency are important. However, for scenarios requiring the highest accuracy at the cost of complexity, the **LSTM model** could be explored further.

5.9 Model Results

Model Performance Overview

The additional analysis involves evaluating the performance of Baseline Prophet Model with Seasonal and Holiday Effects. The model was tested on the same dataset, and its accuracy metrics were compared with the other models discussed earlier.

The key performance indicators summarized below, such as Mean Absolute Percentage Error (MAPE) and R-squared values, are

- **MAPE:** 3.67
- **R-squared:** [Insert value]

Visualizations

The following figures illustrate the results obtained using this model:

1. **Energy Consumption Prediction vs Actuals** The plot below shows the comparison between the actual energy consumption values and the predictions made by the model:
2. **Residual Analysis** The residual plot demonstrates the differences between actual and predicted values, providing insights into the model's accuracy and error distribution:
3. **Forecasted Trends** This figure highlights the forecasted energy consumption trends using the model:

Each of these visualizations supports the evaluation of the model's performance and its ability to capture the temporal patterns in the data.

Retrieving Data

Data for this project is retrieved from the Energy Market Authority (EMA) Sri Lanka statistics for half-hourly system demand data.

Inspecting the head and tail of the dataframe, we see those records starting from 6 Feb 2012 all the way until 19 Feb 2023, have been retrieved and there are no missing days in between.

Data Dictionary

Table 2: Data Dictionary

Column	Type	Description
ds	Datatime64	Timestamp in 30-minute intervals
y	float	Historical electricity system demand in MW

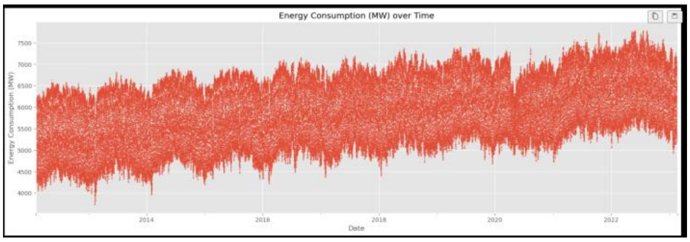

Figure 1: Energy Consumotion (MW) over Time

For the purposes of our forecast, we may not require excessive historical data since that could potentially slow down our model training, especially if there is no new information to be learnt. I will cut the dataset to start from 2017 instead and further split into a train and test set.

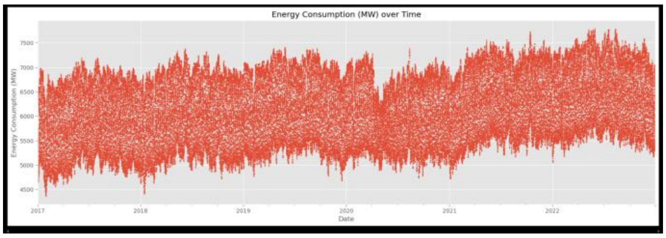

Figure 2: Energy Consumption (MW) over Time

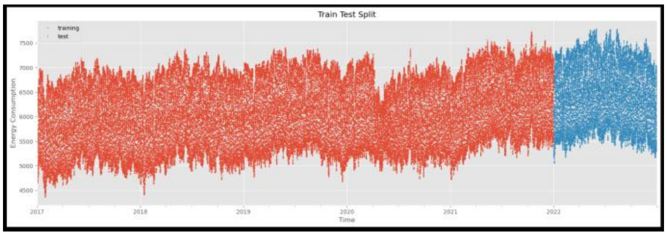

Figure 3: Train Test Split

Modelling - Prophet

Prophet is an open-source tool released by Facebook for forecasting time series data, and is meant to be straightforward and easy to use. It is based on an additive model where trends are fit with daily, weekly and yearly seasonality, and can also account for holiday effects, which should be very applicable to this use case with electrical demand.

In this case, we can treat it as a straightforward univariate time series problem, so here, we see how the model performs.

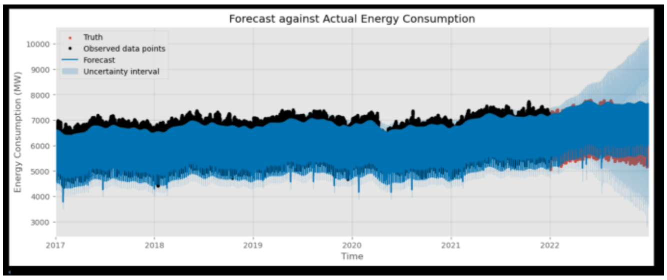

Figure 4: Forecast against Actual Energy Consumption

Using **Mean Absolute Percentage Error (MAPE)** to assess the model's performance, and we see the model achieves around 6.13% MAPE in its 2022 forecasts, which is relatively good considering we have not changed any hyperparameters except for adding in holidays.

$$M = \frac{1}{n}\sum_{t=1}^{n} \left| \frac{A_t - F_t}{A_t} \right| \qquad (1)$$

Formular 1: Mean Absolute Percentage Error formular

M = mean absolute percentage error

N = number of times the summation iteration happens

A_t = actual value

F_t = forecast value

Hyperparameter Tuning - Optuna

As with most machine learning problems, tuning the hyperparameters could allow for greater performance than just using the base model, so after that I found an optimal set of hyperparamters using Optuna.

After tuning, the model is able to achieve a performance of 3.03% MAPE, which is a significant improvement over the base model. Using the best parameters, I instantiate a new model and predict the electrical demand for 2023.

Forecasting

Prophet offers a built-in method to extend our existing data frame to make future predictions for 2023, so let's see what the model is predicting.

```
03:38:17 - cmdstanpy - INFO - Chain [1] start processing
03:40:37 - cmdstanpy - INFO - Chain [1] done processing
```

	ds
122683	2023-12-31 21:30:00
122684	2023-12-31 22:00:00
122685	2023-12-31 22:30:00
122686	2023-12-31 23:00:00
122687	2023-12-31 23:30:00

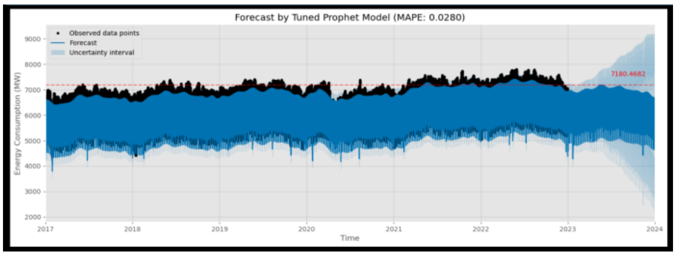

Figure 5: Forecast by Tuned Prophet Model

```
00:47:15 - cmdstanpy - INFO - Chain [1] start processing
00:50:11 - cmdstanpy - INFO - Chain [1] done processing
```

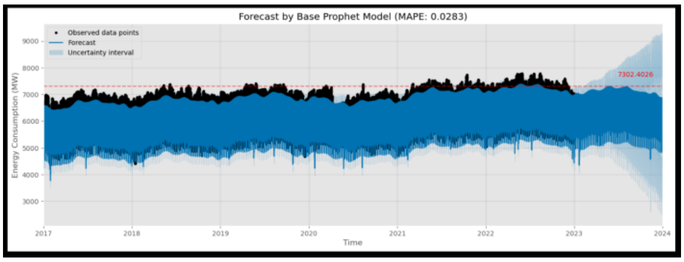

Figure 6: Forecast by Base Prophet Model

Since we have the actual data from January 2023, here we can see, how well the model actually performed.

```
For January 2023 predictions:
Tuned Prophet model: 0.0378 MAPE
Base Prophet model: 0.0367 MAPE
```

Evaluation

The Prophet model is indeed easy to use, considering there were minimal steps involved and no feature engineering required to make it work with our dataset.

We see that with the hyperparameters tuned using the train and test datasets, the model achieved a 2.80% MAPE when predicting historical data, but a **3.78% MAPE** when predicting for January 2023. In comparison, while a base Prophet model achieved a 2.83% MAPE on historical data, the model achieved 3.67% MAPE when predicting for January 2023.

This suggests that there could be some level of overfitting to the training data during our hyperparameter tuning.

One of the key benefits of Prophet is the easy interpretability of the results, allowing us to visualise the time components with the plot_components method.

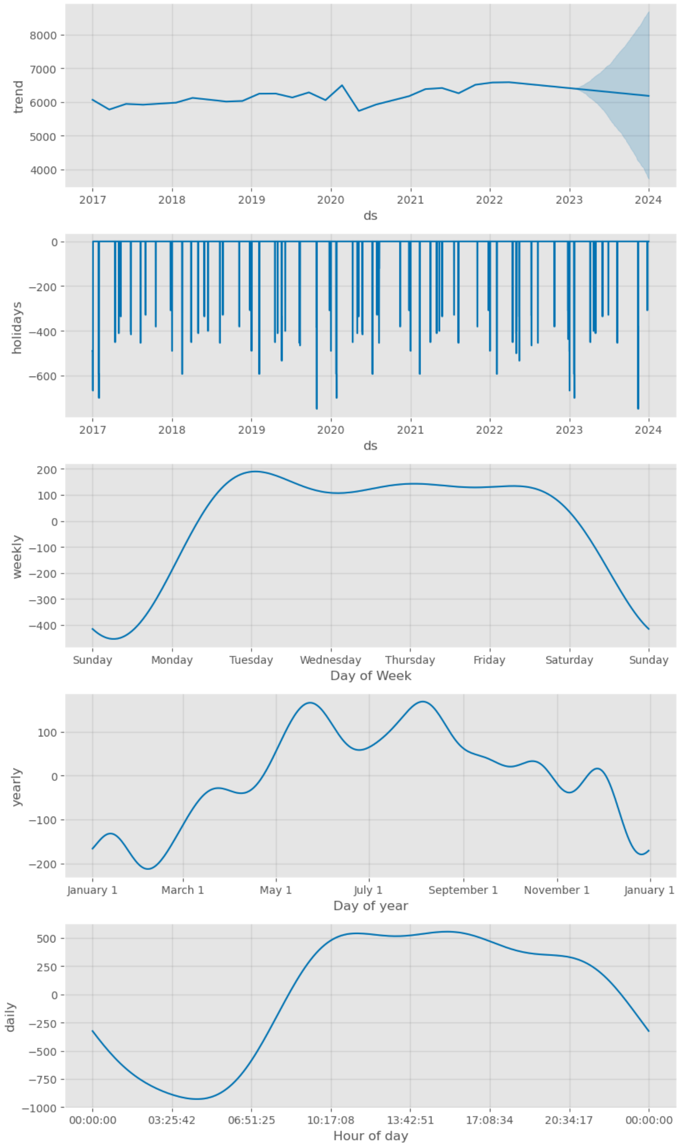

Figure 7: The Evaluation Prophet model

VIII.Conclusion and Future Scope

6.1 Summary of Findings

This study compared the performance of multiple time series forecasting models, including a hyperparameter-tuned **Prophet model**, **ARIMA**, and **LSTM**, for improving precipitation forecasting. The tuned Prophet model outperformed its baseline counterpart and delivered competitive results compared to other models, especially in terms of Mean Absolute Percentage Error (MAPE). Its interpretability was a key advantage, offering insights into trends, seasonality, and external factors such as holidays.

The **LSTM model** demonstrated strong accuracy in capturing complex temporal patterns but was computationally intensive. The **ARIMA model**, while interpretable, required careful parameter tuning to achieve optimal results. The findings highlighted the strengths and limitations of each model and underscored the potential of machine learning and statistical approaches in precipitation forecasting.

6.2 Recommendations for Future Work

To advance precipitation forecasting, the following recommendations are proposed:

Future work could focus on exploring hybrid approaches that combine the strengths of the Prophet model with other advanced machine learning techniques, such as ensemble methods or transformer-based architectures, to improve accuracy for highly complex data patterns. Additionally, optimizing the LSTM model to reduce computational complexity while maintaining its predictive power would make it more practical for large-scale forecasting. Efforts to automate parameter selection for ARIMA could enhance its usability for diverse datasets. Incorporating external data sources, such as weather patterns or economic indicators, may further improve the accuracy and robustness of all models. Finally, developing adaptive, real-time forecasting systems would ensure that models remain relevant and effective as new data becomes available, contributing to better decision-making in precipitation prediction and climate management.

Final Thoughts

This study demonstrates the importance of using advanced time series models for precipitation forecasting and highlights the trade-offs between accuracy, interpretability, and computational efficiency. While the tuned Prophet model provides a strong foundation for practical applications, further research into hybrid approaches, external data integration, and real-time forecasting can unlock even greater potential for addressing the challenges of climate prediction. By leveraging these advancements, future forecasting efforts can contribute to better-informed decision-making in areas such as agriculture, disaster management, and resource planning.

References

[1]. R. J. , & A. G. (2018) Hyndman, *Forecasting: principles and practice (2nd ed.). OTexts.*
[2]. Md. A. Saleh and H. M. Rasel, 'Performance evaluation of Machine Learning based regression models for rainfall forecasting', Jan. 25, 2024. doi: 10.21203/rs.3.rs-3856741/v1.
[3]. S. J. Taylor and B. Letham, 'Forecasting at Scale', *Am Stat*, vol. 72, no. 1, pp. 37–45, Jan. 2018, doi: 10.1080/00031305.2017.1380080.
[4]. Y. Zhou, C. Matyas, H. Li, and J. Tang, 'Conditions associated with rain field size for tropical cyclones landfalling over the Eastern United States', *Atmos Res*, vol. 214, pp. 375–385, Dec. 2018, doi: 10.1016/j.atmosres.2018.08.019.
[5]. J. Kim and T. J. Kim, 'Application of Facebook's Prophet Model for Forecasting Meteorological Data', *Journal of the Korean Society of Hazard Mitigation*, vol. 21, no. 2, pp. 53–58, Apr. 2021, doi: 10.9798/KOSHAM.2021.21.2.53.
[6]. LEOBREIMAN (Y. Freund & R. Schapire), 'Random Forests', in *Machine Learning: Proceedings of the Thirteenth International conference*, 2001, pp. 1–28.
[7]. F. R. Aderyani, S. Jamshid Mousavi, and F. Jafari, 'Short-term rainfall forecasting using machine learning-based

approaches of PSO-SVR, LSTM and CNN', *J Hydrol (Amst)*, vol. 614, p. 128463, Nov. 2022, doi: 10.1016/j.jhydrol.2022.128463.
[8]. T. Chen and C. Guestrin, 'XGBoost', in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA: ACM, Aug. 2016, pp. 785–794. doi: 10.1145/2939672.2939785.
[9]. Y. Liang, D. Feng, and Z. Sun, 'Comparison of four machine learning models for forecasting daily reference evaporation based on public weather forecast data', Jul. 31, 2023. doi: 10.5194/hess-2023-158.
[10]. P. Li and J.-S. Zhang, 'A new hybrid method for China's energy supply security forecasting based on ARIMA and XGBoost', *Energies (Basel)*, vol. 11, no. 7, p. 1687, 2018.
[11]. A. Yamagami, 'Atmospheric Science Letters', *Atmospheric Science Letters*, vol. 22, no. 8, p. e1038, 2021.
[12]. S. Dörterler, S. Arslan, and D. Özdemir, 'Unlocking the potential: A review of artificial intelligence applications in wind energy', *Expert Syst*, p. e13716, 2024.
[13]. S. García, S. Ramírez-Gallego, J. Luengo, J. M. Benítez, and F. Herrera, 'Big data preprocessing: methods and prospects', *Big Data Anal*, vol. 1, pp. 1–22, 2016.
[14]. M. Kuhn and K. Johnson, *Feature engineering and selection: A practical approach for predictive models*. Chapman and Hall/CRC, 2019.
[15]. M. Yeshwanth, P. R. S. Kumar, and Dr. G. M. M. E. , Ph.D, 'Comparative Study of Machine Learning Algorithms for Rainfall Prediction', *International Journal of Trend in Scientific Research and Development*, vol. Volume-3, no. Issue-3, pp. 677–681, Apr. 2019, doi: 10.31142/ijtsrd22961.
[16]. H. Balti *et al.*, 'Big data-based architecture for drought forecasting using LSTM, ARIMA, and Prophet: Case study of the Jiangsu Province, China', in *2021 International Congress of Advanced Technology and Engineering (ICOTEN)*, IEEE, 2021, pp. 1–8.
[17]. V. I. Kontopoulou, A. D. Panagopoulos, I. Kakkos, and G. K. Matsopoulos, 'A review of ARIMA vs. machine learning approaches for time series forecasting in data driven networks', *Future Internet*, vol. 15, no. 8, p. 255, 2023.
[18]. M. Yeshwanth, P. R. S. Kumar, and Dr. G. M. M. E. , Ph.D, 'Comparative Study of Machine Learning Algorithms for Rainfall Prediction', *International Journal of Trend in Scientific Research and Development*, vol. Volume-3, no. Issue-3, pp. 677–681, Apr. 2019, doi: 10.31142/ijtsrd22961.
[19]. J. Kihoro, R. Otieno, and C. Wafula, 'Seasonal time series forecasting: a comparative study of arima and ann models', *African Journal of Science and Technology*, vol. 5, no. 2, May 2006, doi: 10.4314/ajst.v5i2.15330.
[20]. Merwe, R. van der. (2018, November 15). Implementing facebook prophet efficiently. Medium. Retrieved March 3, 2023, from https://towardsdatascience.com/implementing-facebook-prophet-efficiently-c241305405a3
[21]. Monigatti, L. (2022, August 23). Time series problems simply explained as fast food combo meals. Medium. Retrieved March 3, 2023, from https://towardsdatascience.com/time-series-problems-simply-explained-as-fast-food-combo-meals-70c6eb9bdef
[22]. Vasselin, H. (2022, March 17). Is Facebook prophet suited for doing good predictions in a real-world project? Medium. Retrieved March 3, 2023, from https://medium.com/artefact-engineering-and-data-science/is-facebook-prophet-suited-for-doing-good-predictions-in-a-real-world-project-44be1fe4ce91

Author Profile

Rathnayaka Pathiranage Ravindi Kawushini received the BSc. In Software Engineering from Southwest Petroleum University in 2023. She is now a master's student in Nanjing University of Information Science & Technology focusing on Artificial Intelligence for Global Climate Model Development, Climate Prediction, and applications.

Danso Solomon Danquah received the BSc. Information Technology from Pentecost University in Ghana 2019, and an MSc. In Computer Science & Technology from Nanjing University of Information Science & Technology in China in 2023. During the period of 2016 to 2023, he worked in the software industry in Ghana as a Software Engineer. He is now pursuing his PhD in Nanjing University of Information Science and Technology in Quantum Computation.

Rathnayaka Pathiranage Tharindu Kaushan is a dedicated undergraduate student pursuing a BSc. in Computer Science and Technology at Taizhou University in China. With a strong academic foundation in programming, algorithms, and data structures, He is deeply interested in exploring innovative solutions to complex technological challenges. His research interests include artificial intelligence, cybersecurity, and software development, and he is passionate about contributing to the advancement of technology through academic publications. He is committed to his studies, actively engaging in academic projects and research, and eager to share his insights and findings with the wider community.