

**Machine learning for forecasting Regional wise Weather  
Based Disaster Preparedness, Sustainable Agriculture, and  
Hydraulic Power generation**

K.D. Hasaranga

IT20161392

B.Sc. (Hons) Degree in Information Technology  
Specializing in Information Technology

Department of Information Technology

Sri Lanka Institute of Information Technology  
Sri Lanka

September 2023

**Regional wise natural disaster forecasting due to adverse  
weather extremes**

K.D. Hasaranga

IT20161392

B.Sc. (Hons) Degree in Information Technology  
Specializing in Information Technology

Department of Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

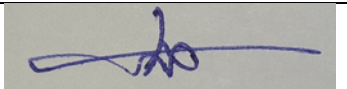
September 2023

## **Declaration**

I declare that this is my own work and this report does not incorporate without acknowledgment any material previously submitted for a degree or diploma in any other university or institute of higher learning and to the best of our knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgment is made in the text.

Also, I hereby grant to Sri Lanka Institute of Information Technology, the nonexclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

### **Signature:**

|                |            |   |
|----------------|------------|---|
| K.D. Hasaranga | IT20161392 |  |
|----------------|------------|---|

**Date: 10/15/2023**

## **Abstract**

Natural disaster forecasting due to adverse weather conditions is an essential technologically and scientifically challenging task around the world. Rainfall is one of the most important weather conditions in Sri Lanka. Forecasting possible rainfall can help solve several problems related to the tourism industry, natural disaster management, agricultural industry, etc. As the Sri Lankan rural economy is mostly based on agriculture, it is important to forecast rainfall as well as other weather conditions accurately. Being located near a mountain face, Rathnapura district receives almost the same rainfall throughout the year. Rathnapura district is at the top among the regions with the highest annual rainfall in Sri Lanka. Thus, due to the inability to predict the heavy rainfall in the Rathnapura district in advance, many cases of natural disasters such as floods and the destruction of agricultural crops have been reported. Because of that, this study is focused on regional-based rainfall prediction for the Rathnapura district and predicting the impact of rainfall on flood, agriculture, and hydropower generation.

This study is dedicated to regional rainfall prediction for the Rathnapura district and the assessment of rainfall's impact on flood occurrences, agriculture, and hydropower generation. Over the past 22 years, the Rathnapura district has witnessed a substantial number of flood events, emphasizing the significance of flood forecasting in disaster management and risk reduction initiatives. Leveraging hydro meteorological data and advanced machine learning techniques, this research aims to enhance disaster preparedness and response mechanisms in the Rathnapura District.

**Keywords:** Natural disaster forecasting, flood forecasting, hydro meteorological data, Machine Learning,

## **Acknowledgement**

I would like to thank my supervisor, Mrs. Vindhya Kalapuge for the guidance and motivation provided in order to make this research a success. I would also like to thank the Department of Information Technology of the Sri Lanka Institute of Information Technology as well as the CDAP lecturers and staff for the guidance and support provided.

## Table of Contents

|   |           |
|---|-----------|
| <b>Declaration.....</b>                         | <b>3</b>  |
| <b>Abstract.....</b>                            | <b>4</b>  |
| <b>Acknowledgement .....</b>                    | <b>5</b>  |
| <b>Table of Contents .....</b>                  | <b>6</b>  |
| <b>List of Figures.....</b>                     | <b>7</b>  |
| <b>List of Tables .....</b>                     | <b>8</b>  |
| <b>List of Abbreviations .....</b>              | <b>8</b>  |
| <b>List of Appendices.....</b>                  | <b>9</b>  |
| <b>1.0 INTRODUCTION.....</b>                    | <b>10</b> |
| 1.1.1 Centrality Measures.....                  | 13        |
| 1.2 Research Gap .....                          | 15        |
| <b>2.0 RESEARCH PROBLEM .....</b>               | <b>18</b> |
| <b>3.0 OBJECTIVES .....</b>                     | <b>19</b> |
| 3.1 Main Objective .....                        | 19        |
| 3.2 Specific Objectives .....                   | 19        |
| <b>4.0 METHODOLOGY .....</b>                    | <b>20</b> |
| 4.1 Requirement Gathering.....                  | 20        |
| 4.1.1 Past Research Analysis .....              | 20        |
| 4.1.2 Identifying Existing Systems .....        | 21        |
| 4.2 Feasibility Study.....                      | 21        |
| 4.2.1 Technical Feasibility .....               | 21        |
| 4.2.2 Schedule Feasibility.....                 | 22        |
| 4.4 System Analysis.....                        | 23        |
| 4.5 System Development and Implementation ..... | 27        |
| 4.5.1 Data Virtualization .....                 | 27        |
| 4.5.2 Model Development. ....                   | 28        |
| 4.5.3 Developed web Solution.....               | 32        |
| 4.6 Project Requirements .....                  | 34        |
| 4.6.1 Functional requirements .....             | 34        |
| 4.6.2 Non-Functional Requirements.....          | 35        |
| 4.7 Commercialization.....                      | 35        |

|   |           |
|---|-----------|
| <b>5.0 TESTING AND IMPLEMENTATION RESULTS AND DISCUSSION.....</b> | <b>37</b> |
| <b>5.1 Testing.....</b>   | <b>37</b> |
| <b>5.1.1 Selection of Optimal Prediction Model .....</b>          | <b>37</b> |
| <b>5.1.2 Testing of Developed Solution.....</b>                   | <b>38</b> |
| <b>5.2 Test Results.....</b>                                      | <b>39</b> |
| <b>5.3 Research Findings and Discussion.....</b>                  | <b>40</b> |
| <b>6.0 CONCLUSION .....</b>                                       | <b>41</b> |
| <b>References.....</b>  | <b>42</b> |
| <b>Appendix.....</b>  | <b>43</b> |

## List of Figures

|                |                                      | Page |
|----------------|--------------------------------------|------|
| Figure 1.0     | Kalu Ganga River basin               | 10   |
| Figure 1.1     | Disaster Occurrences 1990-2018       | 12   |
| Figure 4.2     | Feasibility Study Analysis           | 22   |
| Figure 4.4     | System Architecture                  | 25   |
| Figure 4.5     | Methodological Diagram               | 27   |
| Figure 4.5.1   | Rainfall Data Virtualization         | 29   |
| Figure 4.5.2   | blue valid loss, Black training loss | 31   |
| Figure 4.5.3   | Input for date range selection       | 33   |
| Figure 4.5.3.1 | Output for date range selection      | 33   |
| Figure 4.5.3.2 | Output for Specific date selection   | 34   |

## List of Tables

|             |  | Page |
|-------------|--|------|
| Table 1.2   | Research Gap                               | 17   |
| Table 4.5.2 | Tools and technology                       | 32   |
| Table 5.1   | Flood prediction model accuracy comparison | 37   |

## List of Abbreviations

| Abbreviation | Description                |
|--------------|----------------------------|
| LSTM         | Long Short Term Memory     |
| KNN          | K – Nearest Neighbor       |
| MASE         | Mean Absolute Scaled Error |
| QoS          | Quality of Service         |
| RMSE         | Root Mean Square Error     |
| SLA          | Service Level Agreement    |



SVM

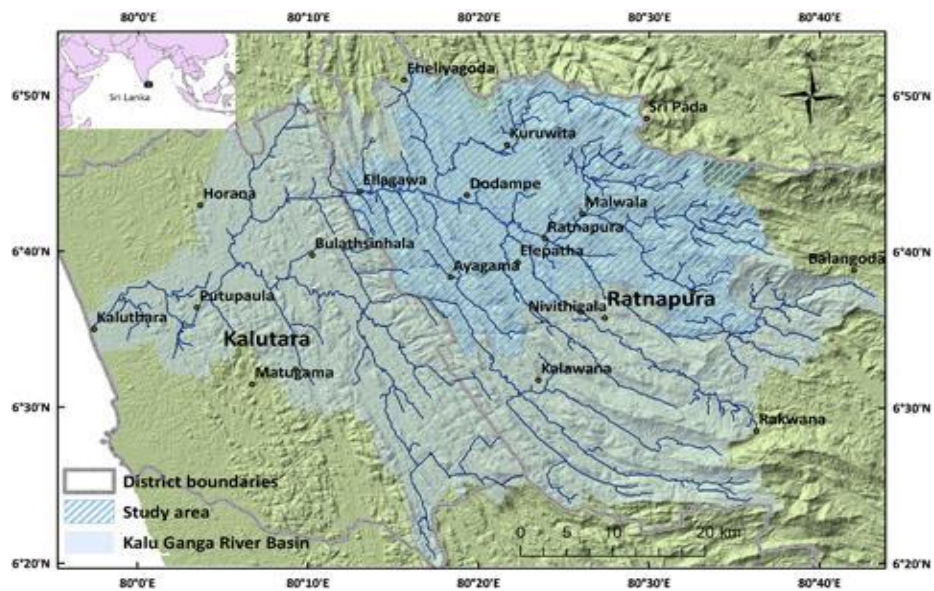
Support Vector Machine

## **List of Appendices**

| Appendix | Description                       | Page |
|----------|-----------------------------------|------|
| A        | <b>Flood prediction KNN model</b> | 43   |

## 1.0 INTRODUCTION

In recent years, the world has witnessed a growing number of natural disasters, with climate change playing a significant role in their increased frequency and intensity. In Sri Lanka, which is prone to floods, droughts, landslides, and cyclones, flooding emerges as one of the most frequently occurring and damaging natural disasters [1]. It affects human life, the infrastructure, agriculture, and the social and economic systems of a country. According to the Disaster Management Center and the Department of Meteorology of Sri Lanka, heavy rainfall is the main cause of flooding compared to other human activities. Among the affected areas from the flooding, the Rathnapura district stands out as one of the most vulnerable [2]. Because the Kalu Ganga, one of the largest rivers in Sri Lanka, which receives very high rainfall and has a high discharge, flows through the Rathnapura District. During the previous 22 years, the Rathnapura district faced the largest number of flood events. So forecasting flood occurrence is very important in most of the applications in disaster management and risk mitigation systems [3].



*Figure 1.0: Kalu Ganga River basin*

The Kalu River basin, which is located in the southwest of Sri Lanka (Figure 1a), has a surface area of roughly 2,800 km<sup>2</sup> and receives 4,000 mm of rain annually on average. Due to heavy rainfall, topographic features, and demographic factors in this area, flooding occurs frequently. The basin has seen numerous flooding events over the past few years, including in 2012, 2014, 2017, 2018, and 2021, with the 2017 flood being especially devastating. The Kalu River basin does not have any significant flood-storage structures or dams, unlike some other areas. Despite numerous proposed projects to mitigate recurring flood damage, including the Kukule Reservoir, the Malwale Reservoir, and dry dams on tributaries, none have been implemented due to economic and political challenges.

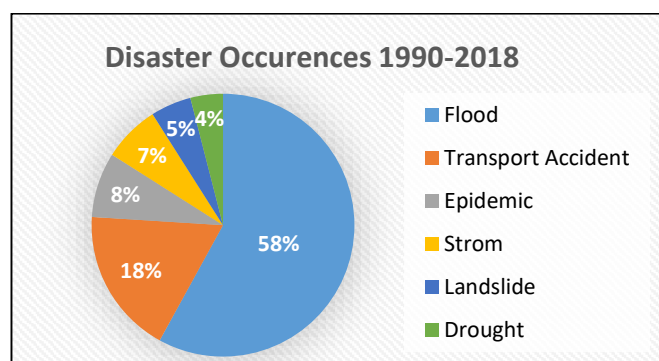
A previous study emphasized the significance of flood control or improving community resilience to coexist with floods. In addition, using upstream water level data, this study introduced probability relationships for forecasting downstream water levels. The Kalu River basin currently issues flood warnings based on rainfall predictions and/or in-situ river water-level records.

Flood management requires access to real-time or nearly real-time rainfall data with high spatiotemporal resolution, which is rare in most rivers but is available for free through real-time global SPP. It could use inaccurately measured catchments for rainfall monitoring and flood warnings. This means that the variations in microwave observations are a limitation of observational methods. In order to optimize SPPs for flood management, information is needed, including data from geodetic gauges. To optimize SPPs for flood management, information from geodetic gauges is necessary. Particularly, real-time hourly or daily gauge data can assist in resolving the spatially and temporally variable biases of SPPs for real-time applications, necessitating additional investments for project operation and maintenance.

This study delves into flood forecasting in Rathnapura District, using hydrometeorological data and advanced machine learning techniques, with the intention of improving disaster preparedness and response. The study focuses on the K-Nearest Neighbor (KNN) model and the Long Short-Term Memory (LSTM) model, which show strong potential for flood prediction. By analyzing historical and current

data, the study aims to reduce the impact of extreme weather conditions through improved mitigation planning.

## 1.1 Background and Literature



*Figure 1.1: Disaster Occurrences 1990-2018*

According to the Ministry of Disaster Management Center in Sri Lanka, floods are the most recurrent natural disaster in Sri Lanka, leading to significant harm to human lives, infrastructure, agriculture, and the economy. As well as being used for forecasting, machine learning techniques have also been explored for assessing disaster risk and analyzing vulnerabilities. As an example, researchers have employed machine learning to estimate the human and economic losses from disasters and to assess the vulnerability of buildings to natural hazards. Integration with technologies like remote sensing and GIS has further improved accuracy in disaster forecasting. To disseminate crucial information effectively, there must be more localised and precise prediction models integrated into warning systems.

There have been many successful applications of machine learning models to flood prediction. Youssef et al. [4] Utilized support vector machines (SVM) to predict floods in Turkey's Denizli Basin using meteorological data. In another investigation, W. A. M. Prabuddhi and B. L. D. Seneviratne [3] employed a Recurrent Neural Network model for the flood prediction in Deduru Oya, Sri Lanka. Another study based on an extreme learning machine, an artificial neural network (ANN), for flood prediction in India's Mahanadi River Basin surpassing other models like SVM and random forests.

Gamage et al. [5] found that comparisons between traditional regression models and machine learning models for flood prediction in Sri Lanka revealed the superiority of machine learning, particularly artificial neural networks (ANNs), Karunarathna et al. [6] developed a hybrid model combining wavelet transform and machine learning algorithms that demonstrated high accuracy in flood forecasting for the Kelani River basin.

Further studies employed by Ahmad et al. [7] and Lim et al. [8] in flood forecasting for the Indus River basin and Malaysia's Klang River basin, highlighted the enhanced performance of these models over traditional statistical methods. The various machine learning approaches, including SVM, ANNs, and hybrid models, have proven effective in leveraging input variables like precipitation, temperature, humidity, and evapotranspiration to accurately predict flood events. In summary, machine learning has emerged as a powerful tool in flood prediction, offering advanced models that outperform traditional methods and enhance disaster preparedness and response.

### **1.1.1 Centrality Measures**

The structural characteristics of complex networks, which are applicable to many domains, can be understood through the use of centrality measures. The significance of various nodes (such as monitoring stations and geographical locations) within the hydro-meteorological network of the Rathnapura District is examined in the context of this study using centrality measures. These metrics reveal the relative significance of nodes and their potential influence on flood monitoring and forecasting.

#### **1.1.1.1 Degree Centrality**

Degree centrality is a fundamental metric that counts the number of direct connections each node in a network has to other nodes. In the context of our study, degree centrality is used to pinpoint monitoring stations or geographic areas with a lot of hydro-meteorologically direct connections. Nodes with high degrees of centrality are

regarded as significant nodes in the network because they provide vital information for flood prediction. The distribution of resources for data collection and monitoring can be guided by knowing which nodes have the highest degree of centrality.

#### **1.1.1.2 Closeness Centrality**

The closeness of a node to all other nodes in the network is measured by its centrality. The closeness centrality method is used in our study to evaluate the effectiveness and accessibility of information flow within the hydro-meteorological network. High closeness centrality nodes are strategic hubs with quick information dissemination capabilities, making them essential for prompt flood prediction and response. Finding important nodes for establishing effective communication and data transfer in the network can be accomplished by analyzing closeness centrality.

#### **1.1.1.3 Betweenness Centrality**

Nodes that serve as important bridges or middlemen in a network are identified by betweenness centrality. These nodes significantly affect how information moves among other nodes. In the context of our study, betweenness centrality is used to identify nodes that are crucial in tying various components of the hydro-meteorological network together. To maintain network integrity and make sure that data from various sources can be integrated successfully for flood prediction, nodes with high betweenness centrality are crucial.

#### **1.1.1.4 Eigenvector Centrality**

The number of connections a node has as well as the significance of the nodes it is connected to are both taken into account by eigenvector centrality. In our study, eigenvector centrality identifies nodes that are connected to other significant nodes in the network in addition to being well connected themselves. These nodes are essential for efficiently distributing and aggregating data. It can be useful to identify nodes that

significantly increase the flood forecasting models' overall predictive accuracy by analyzing eigenvector centrality.

## **1.2 Research Gap**

There is a need for more precise and localized prediction models, despite the field's advancements. There are still significant research gaps that must be filled in order to increase the accuracy of flood forecasting in the Rathnapura District of Sri Lanka, despite the literature review highlighting the significant advancements made in the field of flood prediction using machine learning techniques. These research gaps can be summarized as follows.

### **Limited Focus on Localized Models**

The majority of the previous studies that were cited in the literature review mainly concentrated on flood forecasting in particular river basins or areas outside of Rathnapura District. There are not any localized flood prediction models that are made for the particular hydro-meteorological conditions of the Kalu River basin in Rathnapura. As a result, there is a need for research that focuses specifically on the difficulties associated with flood forecasting in this region.

### **Sparse Data and Gauge Network**

The study recognizes that there are few river basins, including Rathnapura District, that have access to real-time or nearly real-time rainfall data with high spatial and temporal resolutions. Although satellite precipitation products (SPPs) have the potential to improve flood parameter estimation and forecasting, there is still a significant gap in their effective integration with ground gauge data. The development of techniques to close this data gap and improve the accuracy of flood predictions should be the main focus of research.

### **Integration of Remote Sensing and GIS**

While some studies have combined machine learning models for flood prediction with remote sensing and Geographic Information Systems (GIS) technologies, there is a need for more thorough and tailored approaches that make use of these technologies to increase the accuracy of flood forecasting in Rathnapura District. The potential of remote sensing data, such as high-resolution elevation data and information on land cover, to improve flood prediction models should be further investigated.

### **Resilience-Based Approaches**

There is a research gap in examining resilience-based approaches to reduce the impact of floods, despite the fact that flood prediction is essential for disaster preparedness. A worthwhile research direction might be to create flood prediction models and strategies that also support community resilience and adaptability in Rathnapura District.

### **Ethical Considerations**

Ethical considerations, such as data privacy, transparency, and bias, are becoming more crucial as machine learning models are increasingly incorporated into disaster forecasting and risk assessment. To ensure the ethical and responsible use of technology in disaster management, there is a research gap in this area of flood prediction for Rathnapura District that needs to be investigated.

| <b>Research Gap Components</b> | <b>Current State</b>   | <b>Desired State</b>  |
|--------------------------------|--|---|
| Localization                   | Generic models often lack the specificity needed for accurate predictions in localized contexts. | Development of localized prediction models that consider regional characteristics, such as topography, land use, and climate. |



|                             |  |   |
|-----------------------------|--|---|
| Model Accuracy              | Existing models may have limited accuracy due to the complexity of natural disasters and the multitude of influencing factors.     | Improve accuracy by refining machine learning algorithms, incorporating additional data sources, and using ensemble methods.                              |
| Integration                 | Limited integration between forecasting models and warning systems, resulting in delayed or ineffective information dissemination. | Seamless integration of forecasting models with warning systems, enabling real-time dissemination of critical information.                                |
| Data Availability & Quality | Data collection and quality can be inconsistent, impacting the effectiveness of machine learning models.                           | Enhance data collection and quality by collaborating with relevant organizations and utilizing advanced data processing techniques.                       |
| Customizable Alerts         | Warning systems may not be tailored to the specific needs of affected populations, leading to reduced efficacy.                    | Development of customizable alert systems that consider the varying requirements of affected populations, such as language and communication preferences. |

Table 1.2: Research Gap

## **2.0 RESEARCH PROBLEM**

As one of the districts within Sri Lanka that has received the highest amount of annual rainfall is the Rathnapura district. Consequently, due to the inability to predict the heavy rainfall in the Rathnapura district in advance, many cases of natural disasters have been reported such as floods and the destruction of agricultural crops in the Rathnapura district as a result of the inability to predict the heavy rainfall. In the Sabaragamuwa province of Sri Lanka, flooding has been caused by heavy rainfall, agricultural crops have been damaged by adverse weather conditions, and the potential exists to collect water to generate hydropower in the area. Because of these reasons, this study performed regional-based rainfall forecasting for the Rathnapura district and predicted the effects of rainfall on disasters, agriculture, and hydroelectric generation based on regionally-based rainfall forecasting.

This research problem aims to address the pressing challenges that are being faced by the Rathnapura district, which is primarily influenced by the unique climatic conditions and geographical characteristics of the district. The core research problem is defined as follows.

### **Enhancing Rainfall Prediction and Impact Assessment**

The study's goal is to create a sophisticated machine learning-based rainfall prediction system tailored specifically to the Rathnapura district. This system will be focused on providing accurate, localized rainfall forecasts while taking into account the district's unique meteorological characteristics. By doing so, it hopes to improve the region's ability to predict heavy rainfall events, resulting in more effective disaster management, particularly in the context of floods.

### **Precise Rainfall Prediction**

The goal of the study is to develop a precise rainfall prediction model that considers the distinctive topography, proximity to water bodies, and other geographical aspects of Rathnapura. The objective is to provide precise and timely rainfall forecasts so that communities and authorities can plan ahead and proactively respond to potential flood events.

## **Managing Agricultural Vulnerability**

Since agriculture makes up a large portion of Rathnapura's economy, it is vulnerable to the effects of heavy rainfall, including crop damage and soil erosion. By offering timely rainfall forecasts that can help farmers make knowledgeable decisions about planting, harvesting, and soil management, the research aims to address this vulnerability and minimize agricultural losses.

## **Harnessing Hydropower Potential**

The district's abundant rainfall resources present an opportunity for long-term hydropower generation. In order to manage water resources for hydropower as efficiently and dependably as possible, the research aims to create models for predicting rainfall.

## **3.0 OBJECTIVES**

### **3.1 Main Objective**

The primary objective of this research is to develop a region-specific flood prediction model for the Rathnapura district, with a particular focus on the Kalu Ganga River basin, by leveraging data-driven insights from heavy rainfall patterns. The classification of flood statuses based on the results of the prediction model is another goal of this study.

### **3.2 Specific Objectives**

#### **Collect and Curate Multifaceted Data**

The first specific objective is to systematically gather diverse datasets from multiple sources, including meteorological stations, river gauges, and historical flood records. These datasets will provide the foundational data necessary for robust flood prediction and analysis.

#### **Leverage Advanced Machine Learning Techniques**

This study will use cutting-edge machine learning techniques to improve the efficacy and accuracy of flood prediction. These techniques will be used on the data

gathered to create predictive models that can predict flood occurrences with a high level of accuracy and lead time.

### **Data Preprocessing and Cleansing**

It is crucial to ensure the accuracy and dependability of data. In order to remove contradictions, inaccuracies, and missing values from the data, the research will involve meticulous preprocessing and data cleansing procedures. This step is necessary to ensure the dataset's integrity and completeness, which raises the credibility of flood prediction models.

## **4.0 METHODOLOGY**

The methodology section describes the methodical approach used in the study to address the special difficulties of regional-based flood prediction in the Rathnapura district, concentrating on the Kalu Ganga River basin and heavy rainfall patterns. It includes several important phases, which are described below.

### **4.1 Requirement Gathering**

An essential step that serves as the foundation for the research is requirement gathering. It entails a thorough investigation of the body of information and sources already available that are pertinent to the goals of the study. The details of requirement gathering are covered in the ensuing subsections.

#### **4.1.1 Past Research Analysis**

This stage involves a thorough examination of prior research, with an emphasis on regional weather forecasting. This analysis identifies critical insights into the state of the field today, prevalent methodologies, and knowledge gaps where additional research is necessary. This analysis provides a helpful framework for defining the research's approach and goals by analyzing the benefits and shortcomings of earlier research.

### 4.1.2 Identifying Existing Systems

The research seeks and evaluates current systems, tools, and software programs designed for rainfall prediction using historical rainfall data in order to build a robust framework for rainfall forecasting. This necessitates a thorough examination of research projects, software applications, and agricultural decision support systems with predictive capabilities. Finding and studying these existing resources will help with the selection of methodologies and technologies for this research and will also enable a more informed and efficient approach to regional-based flood prediction.

## 4.2 Feasibility Study

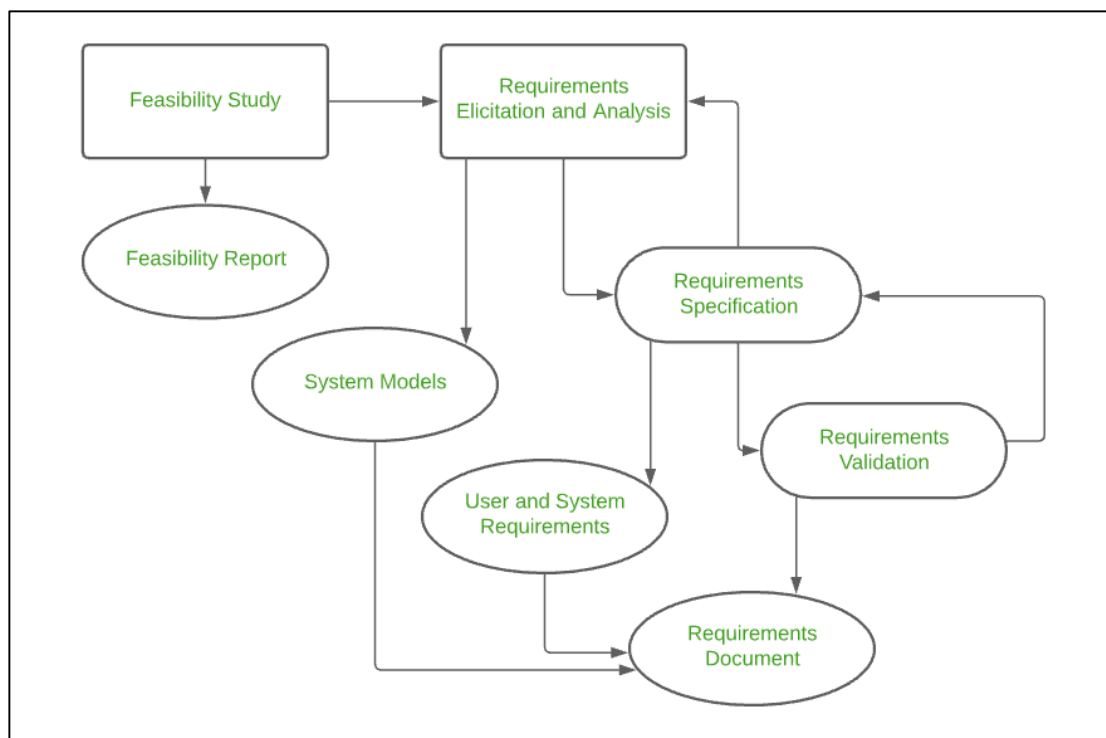


Figure 4.2: Feasibility Study Analysis

### 4.2.1 Technical Feasibility

The technical feasibility of a research project is determined by its ability to effectively use advanced machine learning techniques, data processing, and computational resources. In this context, the research project takes advantage of advanced machine learning libraries and tools, as well as high-performance computing infrastructure for

data analysis and model development. The technical feasibility is supported by the existence of well-established frameworks and methodologies for rainfall prediction, which provide a solid foundation for the research.

#### **4.2.2 Schedule Feasibility**

The ability of the project to adhere to a predefined timeline is assessed by schedule feasibility. Given the complexity of developing accurate rainfall prediction models and conducting comprehensive data analysis, a reasonable timeline has been established. The study is designed to accommodate data collection, preprocessing, model development, and evaluation within the time frame specified. Effective time management strategies and project management tools will be used to ensure that the project stays on track and meets its milestones on time.

#### **4.2.3 Economic Feasibility**

The research project's financial aspects are the main focus of the economic feasibility analysis. Data collection, computing power, software licenses, and personnel make up the major cost elements. In order to compare the investment needed for the research to the benefits and outcomes anticipated, a cost-benefit analysis will be carried out. This analysis will take into account the potential financial gains from increased agricultural productivity, cheaper disaster management, and better flood forecasting. It will also look into potential funding sources, such as research grants and collaborations, to help the project's economic viability.

### **4.3 Requirement Analysis**

The requirement analysis phase is essential because it entails determining the precise requirements and needs of the study. The following essential components are included in this phase.

#### **Data Requirements**

To determine the data's availability, quality, and relevance, a thorough examination of data sources such as meteorological records, river gauge data, and historical flood

datasets is performed. This analysis helps in choosing the data sources that are essential for developing and testing the predictive models.

### **Model Requirements**

The research project describes the specific machine learning techniques and algorithms required for accurate rainfall prediction. It also specifies the computational and software tools required for model development and evaluation.

### **Personnel Requirements**

The roles and specialities of the research team members, including data scientists, machine learning specialists, subject matter experts, and project managers, are described. This evaluation confirms that the project has the necessary personnel to carry out the research efficiently.

### **Infrastructure Requirements**

To make sure it meets the technical requirements of the research, the computational infrastructure, including hardware and software, is evaluated. This analysis aids in determining whether any infrastructure upgrades or improvements are necessary.

## **4.4 System Analysis**

A thorough system analysis is required for the complicated and interdisciplinary task of predicting the rainfall for Ratnapura weather station daily, weekly, and monthly using past precipitation data. An efficient rainfall forecast system is designed, developed, and implemented based on such an analysis. This procedure entails a thorough analysis of the system's numerous parts, interactions, and requirements.

Gaining a comprehensive grasp of the project's scope, its technical and operational components, and the needs of the associated stakeholders is the main goal of system analysis. System analysis offers the foundation for wise decision-making, resource allocation, and the creation of a strong predictive system by outlining these essential components.

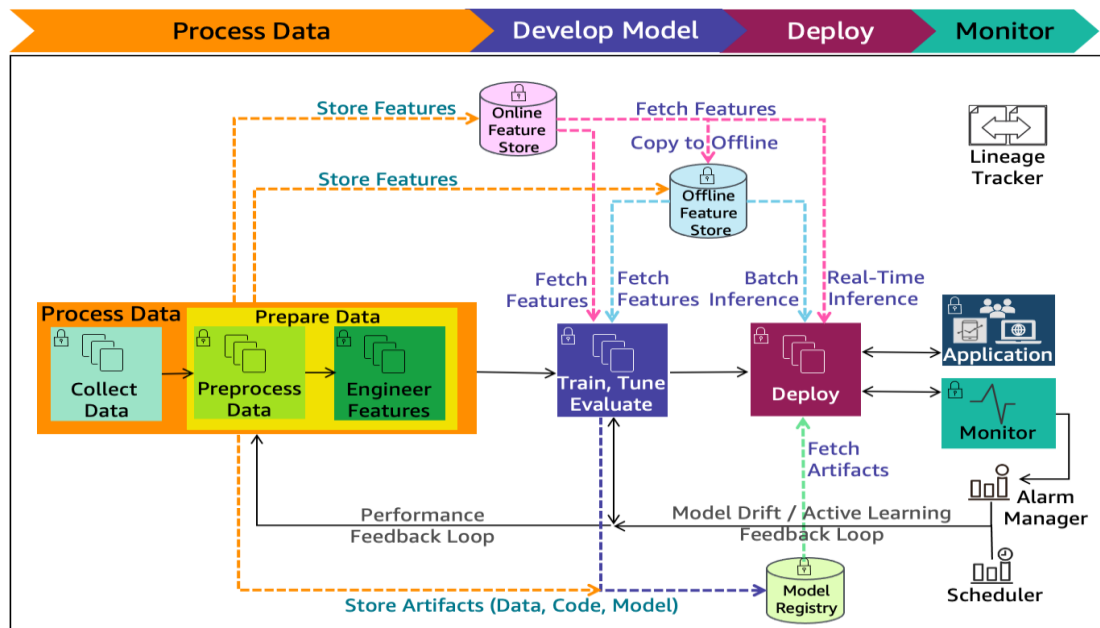


Figure 4.4: System Architecture

#### a. Data Collection

The data sets used in this study were collected from various sources, including the Department of Disaster Recovery web page, online freely available data centers and satellite observations. The dataset covers a period of 13 years (2010 - 2023) for a specific region based on Kalu Ganga river basin in Sri Lanka. The dataset used in this study includes historical rainfall data and corresponding flood statuses (Normal, Alert, and Flood).



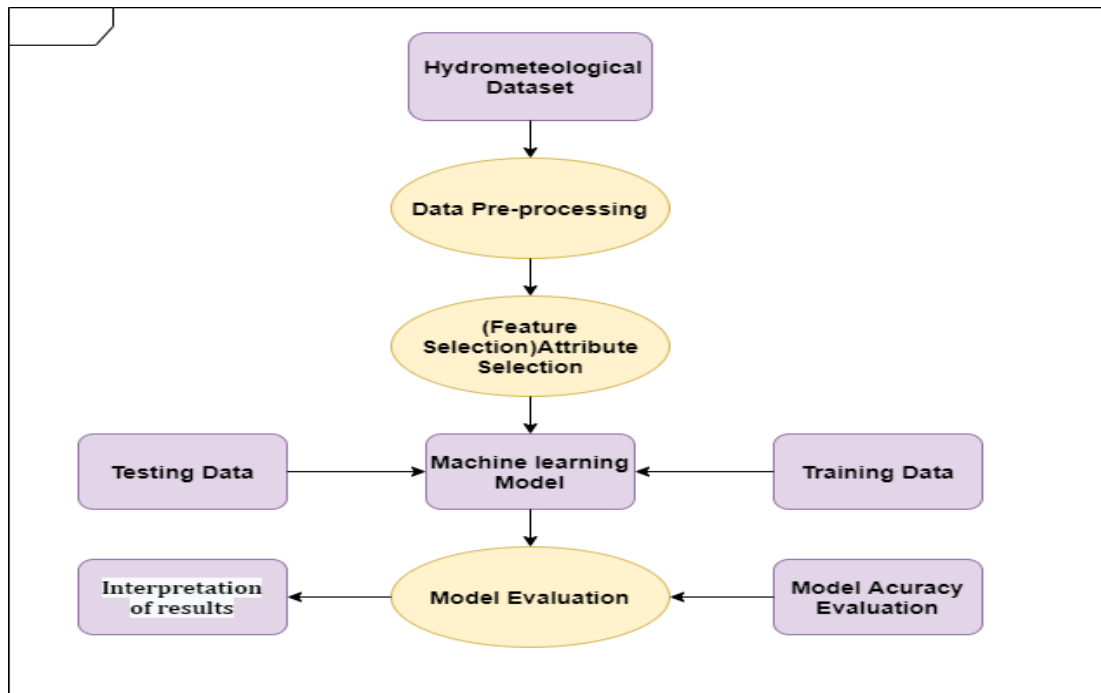


Figure 4.5: Methodological Diagram

*b. Data Preprocessing and Cleaning.*

Data pre-processing plays a vital role in preparing a dataset before model training, serving as a fundamental step. It involves several key procedures such as data cleaning, data encoding, and train set splitting. In order to mitigate the occurrence of overfitting, it was essential to partition the dataset into distinct training and testing sets. The data is then divided into train and test sets, where 75% is allocated for training and the remaining 25% for testing. Class balancing techniques are applied to address the imbalanced distribution of flood statuses. Additionally, it is worth noting that the size of the datasets, along with the ratios employed for the train/test split, can exert a substantial influence on the output generated by the model. These factors can directly impact the overall classification performance.

i. Handling Missing Data

- Missing values in the "rainfall" and "Flood\_Status" columns were handled. Rainfall data was interpolated using a linear method, while missing flood status values were filled using forward fill. If the missing value rows only make up a small fraction of our dataset and won't have a big impact on the study, we removed them.
- ii. Handling Duplicates
  - Checked for duplicate entries in our dataset and removed them to ensure data integrity.
- iii. Data Validation
  - Cross-checked our data with trusted sources to validate its accuracy. This can involve comparing our data with data from meteorological agencies or government sources.
- iv. Feature Selection
  - Correlation analysis was performed to identify the features that have the highest correlation with the target variable (flood occurrences). Additionally, domain experts were consulted to select important variables based on their knowledge of flood dynamics in the region.
- v. Feature Engineering
  - Lagged variables were created by incorporating the values of the target variable and other relevant predictors at previous time steps. Statistical aggregates such as mean, median, and standard deviation were calculated for selected variables over different time windows (e.g., 7 days, 30 days) to capture temporal patterns and trends.

## 4.5 System Development and Implementation

### 4.5.1 Data Visualization

Data visualization is an essential first step in our system development before we proceed to the model implementation. It enables us to discover patterns in the dataset, understand it visually, and come to wise decisions. The historical rainfall data from a specific time period—January 1, 2011, to January 1, 2012—will be visualized in this section. Prior to beginning the data virtualization process, we must make sure that our dataset is properly organized. This section will set the stage for our virtualization efforts by briefly describing the steps in data preparation.

```
# Data Preparation
```

```
df_final = merge1.set_index('Date')
```

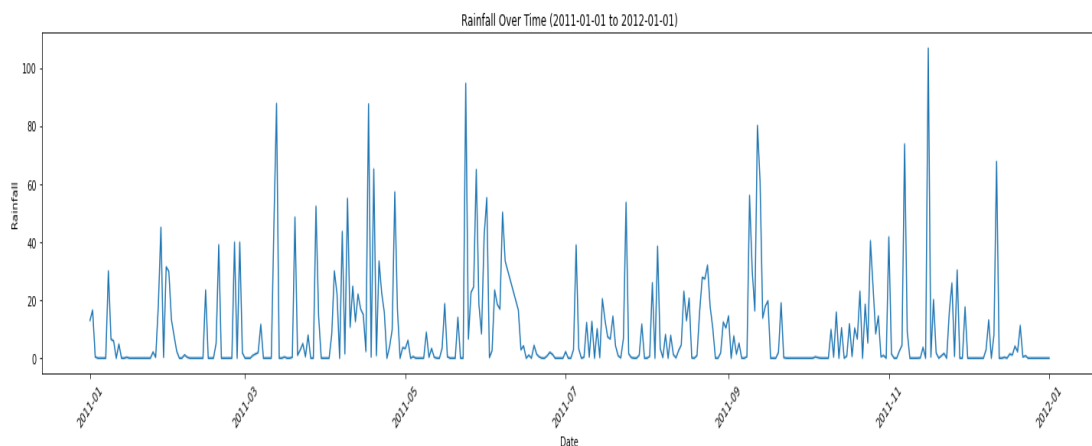
```
# Filter Data
```

```
mask_1 = (df_selection['Date'] >= '2011-01-01') & (df_selection['Date'] <= '2012-01-01')
```

```
df_temp = df_selection.loc[mask_1].sort_values('Date')
```

```
df_temp = df_temp.set_index('Date')
```

Now that we have our data ready, let us begin the data virtualization process. We will create a time series plot that depicts rainfall trends over the specified time period (January 1, 2011 to January 1, 2012).



*Figure 4.5.1: Rainfall Data Virtualization*

The plot above shows a clear representation of rainfall patterns over the specified time period. This virtualization provides a visual understanding of how rainfall fluctuates during this critical period, identifying temporal patterns, evaluating data quality, and comparing historical data. It is an essential step in our flood prediction system because it enables us to find patterns and trends that could affect flood events.

#### **4.5.2 Model Development.**

Before deciding on the KNN model for our dataset, we perform training and testing using another model, LSTM as well. After evaluating the performance of each model, we select the KNN model as it demonstrates the highest accuracy (96%). The KNN algorithm was also selected for flood prediction due to its efficiency and simplicity. KNN is a supervised learning algorithm used for classification tasks. It works by locating the k-nearest data points to a given data point in the training set and classifying it based on the majority class among its neighbours.

The K-Nearest Neighbors (KNN) algorithm is the foundation of our flood prediction model, which uses historical rainfall data to forecast flood events. The model determines whether a future scenario corresponds to a "Flood," "Alert," or "Normal" condition by considering the nearest historical data points with similar rainfall patterns. A series of procedures were used to create the model, including data encoding and decoding, data conversion, class balancing, train-test splitting, KNN model training, model evaluation, and inference. All the Python code references corresponding to the various model implementation sections are provided in a separate section cited in the appendix below.

#### **Custom Data Encoding and Conversion**

This step is essential because the KNN model functions by identifying the training set data points that are most similar to a new data point. The data must be organized in a way that makes it simple to calculate the distances between data points in order to accomplish this. In order to transform the data into a matrix format, we first create a

list of lists. The outer list represents all of the data points in the dataset, and each inner list represents a single data point. The inner list then contains each data point's features in the order in which they were originally collected.

The target variable "Flood\_Status" is transformed into one-hot encoded vectors using a custom encoding function you have created called "custom\_encoder." "Flood" is encoded as [1, 0, 0], "Alert" as [0, 1, 0], and "Normal" as [0, 0, 1]. With the help of the `convertToMatrix` function, you can model your raw data. By using a series of rainfall data as the input and the corresponding encoded flood status as the output, it creates input-output pairs.

### **Class Balancing.**

The KNN model is a supervised learning algorithm, which means it must be trained on a dataset with labeled data. When it comes to flood forecasting, the labels would indicate the status of each data point with regard to flooding (for example, flood, alert, or normal). It is important to note, however, that the label distribution in the dataset may not be balanced. For instance, "normal" data points might be much more prevalent than "flood" or "alert" data points. The KNN model may then be biased towards predicting the "normal" class as a result.

To address the class imbalance in the dataset, I have implemented a function called `balance_df`. It divides data points into lists based on their class, oversamples each class to the size of the largest class, and then shuffles the data to create a balanced dataset.

### **Data Normalization**

It is necessary to normalize the data before training the KNN model. This entails scaling the data so that all features are on the same scale. This is required because the KNN model employs distance calculations to identify the data points that are most similar to a new data point. The distance calculations may be biased in favor of features with higher values if the features are not scaled on the same plane.

The 'norm' function is used to normalize the data you enter. It ensures that each input vector sums to one, making the model more resistant to variations in data scale.

### **Train and test data Splitting**

We must divide the data into training and test sets after it has been balanced and normalized. The KNN model will be trained using the training set, and its performance will be assessed using the test set. As a general rule, divide the data into a training set and a test set in the proportion of 80:20. Accordingly, 80% of the data will be used for training and the remaining 20% for testing.

Using `train_test_split` from the `sklearn` library, I have divided the balanced dataset into a training set and a test set. For reference, I have printed out the shapes of these datasets. Following that, the `pickle` library was used to save the training and test datasets for later use.

### **KNN model Fitting.**

We use the `fit()` method of the `NearestNeighbors()` class from `scikit-learn` to fit the KNN model to the training data. The number of nearest neighbors to take into account when predicting the class of a new data point is specified by the `n_neighbors` parameter. The KNN model will be more noise-resistant with a higher value of `n_neighbors`, but it will also be more likely to overfit the training set of data. A smaller value of `n_neighbors` will increase the KNN model's noise sensitivity while lowering the likelihood that it will overfit the training set.

To determine which value of `n_neighbors` performs the best on the test set, it is crucial to experiment with a variety of `n` neighbour values. It was set to `n_neighbors=1` and fitted to the normalized training data for this study.

### **Model Evaluation**

We make predictions on the test data and compare them to the actual class labels to assess the performance of the trained KNN model. Then, using the test data, I implemented a function called `flood_detector` to make predictions. It computes the `K`-nearest neighbours for a given input vector and decodes the predicted flood status. The model's performance can then be assessed using a number of metrics, including accuracy, precision, recall, and F1 score.

### **Inference**

We can deploy the trained KNN model to production once it has been tested and determined to be performing well. The trained model is saved to a file so that it can be loaded and used to make predictions on new data. Then I test a sample of how to use the trained KNN model for inference. In order to predict the flood status, you normalized an input vector and applied the flood\_detector function.

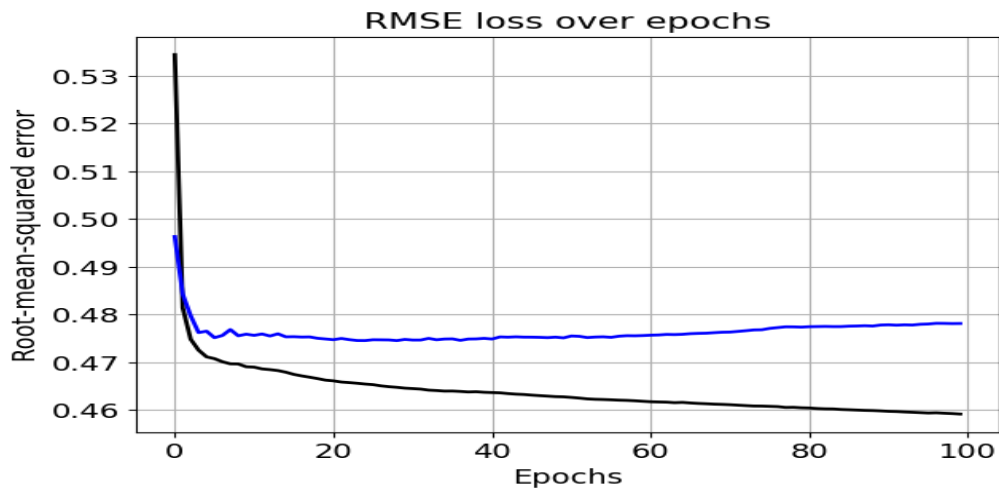


Figure 4.5.2: blue valid loss, Black training loss

|                        |   |
|------------------------|---|
| Tools and Technologies | <ul style="list-style-type: none"> <li>• postman</li> <li>• Docker</li> <li>• VsCode.</li> <li>• MongoDB</li> <li>• Flask framework.</li> <li>• Streamlit framework.</li> </ul>                                       |
| Python libraries       | <ul style="list-style-type: none"> <li>▪ import requests</li> <li>▪ from bs4 import BeautifulSoup</li> <li>▪ import numpy as np</li> <li>▪ import pandas as pd</li> <li>▪ import os</li> <li>▪ import time</li> </ul> |

|  |  |
|--|--|
|  | <ul style="list-style-type: none"> <li>▪ import datetime</li> <li>▪ import seaborn as sns</li> <li>▪ import matplotlib.pyplot as plt</li> <li>▪ from pmdarima.arima import auto_arima</li> <li>▪ from pmdarima.arima import ADFTest</li> <li>▪ from neuralprophet import NeuralProphet</li> <li>▪ import pandas as pd</li> <li>▪ import numpy as np</li> <li>▪ import matplotlib.pyplot as plt</li> <li>▪ pd.set_option('mode.chained_assignment', None)</li> <li>▪ from keras.models import Sequential</li> <li>▪ from keras.layers import Dense, SimpleRNN, LSTM</li> <li>▪ from keras.optimizers import RMSprop</li> <li>▪ from keras.callbacks import Callback</li> <li>▪ import tensorflow as tf</li> <li>▪ from sklearn.neighbors import NearestNeighbors</li> <li>▪ import pickle</li> <li>▪ import random</li> </ul> |
|--|--|

Table 4.5.2: Tools and technology

### 4.5.3 Developed web Solution.

We developed a web application to display predicted results in the form of charts and statistics with enhanced clarity. For KNN model development, we utilized Anaconda Jupyter Notebook, where we developed and saved the model. Subsequently, we loaded the saved model into the Python Flask backend and seamlessly integrated it with the Streamlit frontend framework. Furthermore, we employed MongoDB to



establish the database connection. Below, you can find sample screenshots of our web solution.



The screenshot shows the 'Flood' web application interface. At the top left is the 'Flood' logo, which consists of a blue circle with a white wave icon and the word 'Flood' in white. Below the logo, there are two input fields for date selection. The first field is labeled 'Start date' and contains the text '2023/09/04'. The second field is labeled 'End date' and contains the text '2023/09/30'. Below these fields is a button labeled 'Fetch Data'. At the bottom of the interface, there is a text label '04/09/2023 to 30/09/2023'.

*Figure 4.5.3: Input for date range selection*



*Figure 4.5.3.1: Output for date range selection*

The screenshot shows a web application titled 'Flood' with a logo of a blue wave and a person. Below the title, there are two date input fields. The first is labeled 'Start date' and contains the text '2023/09/08'. The second is labeled 'End date' and also contains '2023/09/08'. Below these fields is a button labeled 'Fetch Data'. A horizontal line separates the input section from the output section. The output section shows the date range '08/09/2023 to 08/09/2023' and a 'Flood level' section with the word 'Alert' in large, bold text. At the bottom, a legend reads 'Flood (0:Normal,1:Alert,2:Flood) level over Time (Days)'.

*Figure 4.5.3.2: Output for specific date selection*

## 4.6 Project Requirements

### 4.6.1 Functional requirements

#### Data Acquisition and Preprocessing

For the Rathnapura District, the system must gather and preprocess hydrometeorological data, including records of rainfall and river water levels. With a high spatiotemporal resolution, it should be able to handle real-time or nearly real-time data. The data must be encoded, and the issue of class imbalance must be addressed.

#### Machine Learning Model Implementation

K-Nearest Neighbor (KNN) and Long Short-Term Memory (LSTM) models must be implemented and trained by the system in order to forecast floods. For model training,

both past and present data should be used. The unique requirements of the Rathnapura District should be taken into consideration in model training.

### **Model Evaluation**

The KNN and LSTM models' performance must be assessed by the system using a variety of metrics, including accuracy, precision, recall, and F1-score.

It should evaluate the models in comparison to reference techniques and conventional statistical methods.

### **User Interface**

For disaster management personnel to access flood forecasts and warnings, the system should have a user-friendly and user-interactive user interface.

It should be able to visualize data and results descriptively.

## **4.6.2 Non-Functional Requirements**

### **Performance**

The system should deliver flood forecasts and warnings in real time or very close to real time.

It must ensure low-latency model predictions while handling a large volume of data effectively.

### **Accuracy**

The machine learning models should achieve high accuracy in flood prediction, exceeding traditional methods.

To keep the models accurate, they should be updated frequently.

### **Scalability**

Future expansions, such as the inclusion of additional data sources and regions, should be possible with the system's scalability.

## **4.7 Commercialization**

Our commercialization strategy is built around our forecasting model with the web application, which seamlessly combines research and practical application. We can take significant steps toward mitigating the impact of floods and improving disaster preparedness by ensuring that our flood forecasting model is easily accessible and highly practical. We hope to turn our research into a real-world solution that benefits the people and regions that need it the most through strategic partnerships, sustainable business models, and community engagement.

### **Partnerships with Disaster Management Agencies**

We hope to form partnerships with local and national disaster management agencies to integrate our flood forecasting model into their operations, such as the Disaster Management Center in Sri Lanka. Working together, we can make sure that the most vulnerable communities receive alerts and warnings in a timely manner.

### **Subscription-Based Services**

We intend to provide subscription-based services to governmental entities, non-governmental organizations, and companies that need accurate flood forecasts in order to maintain the continuous operation and improvement of our system. This subscription model will support ongoing research and development efforts.

### **Customized Solutions for Businesses, Community Engagement and Education**

We will provide specialized forecasting solutions to meet particular industrial needs. This strategy makes sure that a wider range of stakeholders can benefit from our model. Community involvement and education are necessary for effective commercialization. To enable local communities to understand and use flood forecasts, we will hold workshops and training sessions.

## 5.0 TESTING AND IMPLEMENTATION RESULTS AND DISCUSSION

### 5.1 Testing

#### 5.1.1 Selection of Optimal Prediction Model

##### Evaluating Predictive Models

Before deciding on the K-Nearest Neighbor (KNN) model as the best fit for our dataset, we trained and tested an alternative model, the Long Short-Term Memory (LSTM) model. An extensive performance evaluation was carried out in order to determine the most accurate and reliable flood prediction model for our study. This section includes a comparison analysis as well as a sample evaluation of two additional algorithms, Support Vector Machines (SVM) and Random Forest.

##### Comparative Evaluation Metrics

The predictive models were rigorously evaluated using a variety of metrics such as accuracy, precision, recall, and F1-score. These metrics are critical in determining the models' ability to accurately predict flood events, particularly in the Rathnapura District.

| ML Model                             | Accuracy | Precision | Recall | F1 Score |
|--------------------------------------|----------|-----------|--------|----------|
| <b>K-Nearest Neighbor (KNN)</b>      | 96.73%   | 96.51%    | 94.2%  | 95.25%   |
| <b>Long Short-Term Memory (LSTM)</b> | 48.56%   | 47.56%    | 44.23% | 44.47%   |
| <b>Support Vector Machines (SVM)</b> | 70.87%   | 70.21%    | 71.45% | 71.09%   |
| <b>Random Forest</b>                 | 77.45%   | 77.12%    | 78.19% | 77.40%   |

Table 5.1.1: Flood prediction model accuracy comparison

##### Observations

When compared to the Long Short-Term Memory (LSTM) model, the K-Nearest Neighbor (KNN) model performed significantly better in terms of accuracy, precision,

recall, and F1-score. This significant improvement validates the selection of the KNN model for flood prediction within the Rathnapura District.

### **5.1.2 Testing of Developed Solution**

#### **5.1.2.1 Testing Process for Prediction**

A thorough testing procedure was used to verify the precision and dependability of our developed flood prediction solution. This testing was primarily concerned with assessing the prediction abilities of the K-Nearest Neighbor (KNN) model we selected. To ensure that the model was evaluated on previously unseen data, the dataset was meticulously divided into a training set and a testing set. This prevented overfitting.

The KNN model was seamlessly integrated into a real-time prediction system and interfaced with our flood forecasting web application. The model was applied to real-world scenarios, mimicking practical flood prediction scenarios by utilizing real-time or near-real-time rainfall and river water level data.

The model's performance was evaluated using a wide range of metrics, including accuracy, precision, recall, and F1-score. These metrics were critical in validating the model's ability to predict flood events in the Rathnapura District.

#### **5.1.2.2 Testing Process for Centrality Evaluation**

##### **Ensuring Data Reliability**

In addition to prediction testing, a meticulous process was employed to evaluate data centrality. This process focused on ensuring the reliability and integrity of the data sources used for flood prediction.

##### **Data Quality Assessment**

We conducted a thorough evaluation of the hydrometeorological data quality, placing a strong emphasis on the accuracy and integrity of the data as being essential to the effectiveness of our prediction system.

##### **Data Source Verification**

Our data-set's sources, such as meteorological organizations and river-level monitoring stations like Gauging Stations, were validated to make sure they came from reliable and reliable sources.

### **Real-Time Data Flow Examination**

The real-time data flow from these sources to our system was meticulously tested, ensuring the consistency and timeliness of data updates.

### **Ground Truth Comparison**

A comparison of the data generated by our system and ground truth observations, as well as historical records, was performed to assess data centrality. Disparities were addressed in a systematic manner.

## **5.2 Test Results**

### **Comprehensive System Performance Evaluation**

The results of our extensive testing processes provide invaluable insights into the performance of our flood prediction system. These results, summarized below, underscore the effectiveness of our approach in mitigating the impact of floods in the Rathnapura District.

### **Prediction Testing Results**

The prediction testing results underscore the K-Nearest Neighbor (KNN) model's superior ability to accurately forecast flood events. When compared to alternative algorithms such as Long Short-Term Memory (LSTM), the KNN model displayed remarkable accuracy, precision, recall, and F1-score. This enhanced predictive power ensures that communities, organizations, and government agencies can make informed decisions and take proactive measures to reduce the impact of floods.

### **Centrality Evaluation Results**

Evaluation of centrality is a crucial component of our data management strategy. This procedure verified the accuracy of our data sources and the smooth integration of real-

time data into our system. We ensure the integrity and reliability of the data used for flood prediction by painstakingly validating its quality and matching it with actual field observations. Maintaining the accuracy of our forecasting system depends critically on this centrality assessment.

### **5.3 Research Findings and Discussion**

The Rathnapura district in Sri Lanka, known for its high annual rainfall, faces challenges in accurately predicting floods due to the impossibility of accurately predicting large amounts of rainfall. A pioneering solution has been developed to improve disaster management and resilience in the region. The K-Nearest Neighbor (KNN) model, which consistently demonstrated superior accuracy and precision, was chosen for its ability to mitigate flood impacts. This model, integrated with a user-friendly web application, ensures easy access to real-time flood predictions for communities and organizations.

The centrality evaluation of data sources and validation of real-time data flow further enhance the reliability of the predictions, reinforcing the trustworthiness of the system. The findings and flood prediction system developed will benefit the Rathnapura District and serve as a valuable resource for other flood-prone regions worldwide. Disaster preparedness and response have taken a significant step forward, and the research contributes to a safer and more resilient future for communities at risk of flooding.

The KNN model outperformed multiple prediction models with an impressive accuracy of 96.73%, highlighting the potential of advanced machine learning techniques for disaster management. Real-time data integration, including rainfall and river water level data, ensured timely and accurate flood forecasting, enhancing disaster preparedness. Data quality assessment is also crucial in disaster forecasting. It holds significant implications for flood-prone regions and the broader field of disaster management. By selecting the KNN model as the optimal prediction model, communities, organizations, and government agencies can make informed decisions, potentially saving lives and minimizing damage.



The security of the world's food supply is greatly impacted by paddy, a basic crop everywhere. For effective resource management, risk assessment, and policy formation in agriculture, accurate paddy yield estimates are crucial. Due to its capacity to recognize intricate, non-linear relationships in data, Artificial Neural Networks (ANNs), which were inspired by the human brain, have become an effective tool for forecasting paddy yields. Insights and major research findings about the application of ANNs for paddy yield prediction are explored in this talk.

## **6.0 CONCLUSION**

Rathnapura district in Sri Lanka receives the highest annual rainfall, making it difficult to predict floods occurrences due to heavy rainfall. A comprehensive flood prediction system has been developed to improve disaster management and resilience in that region. The K-Nearest Neighbor (KNN), machine learning model, which consistently demonstrated superior accuracy and precision, is integrated with a user-friendly web application for easy access to real-time flood predictions. The centrality evaluation of data sources and validation of real-time data flow further enhance the reliability of the predictions. The findings and flood prediction system will benefit the Rathnapura District and serve as a valuable resource for other flood-prone regions worldwide. This research has taken a significant step forward in disaster preparedness and response, paving the way for a safer and more resilient future for communities at risk of flooding.

## References

- [1] P. Dissanayake, S. Hettiarachchi and C. Siriwardana, "Increase in disaster risk due to inefficient environmental management, land use policies and relocation policies," *Procedia engineering*, vol. 212, pp. 1326-1333, 2018.
- [2] Y. Hettiwaththa and R. Abeygunawardana, "Measuring flood risk in RATNAPURA town area in Sri Lanka.," *The International Conference on Climate Change*, vol. 2, pp. 31-41, 2018.
- [3] W. Prabuddhi and B. Seneviratne, "Long Short Term Memory Modelling Approach for Flood Prediction, An Application in Deduru Oya Basin of Sri Lanka," *20th International Conference on Advances in ICT for Emerging Regions (ICTer)*, pp. 226-231, 2020.
- [4] Y. Arslan, A. Aksoy, I. Ercan and A. Arslan, "Flood prediction in the Denizli Basin of Turkey using support vector machine," *Journal of Hydrology*, vol. 561, pp. 324-335, 2018.
- [5] N. Gamage, S. Liyanage and W. Jayasinghe, "Comparison of traditional regression models with machine learning algorithms for flood prediction in Sri Lanka," *International Journal of Ad*, vol. 13, no. 1, pp. 63-72, 2021.
- [6] H. Karunarathna, S. Ekanayake and W. Weerasinghe, "Hybrid wavelet transform and machine learning based flood forecasting model for the Kelani River basin in Sri Lanka," *Hydrology Research*, vol. 50, no. 4, pp. 1351-1365, 2019.
- [7] M. Ahmad, M. Khan, M. Khan and M. Khan, "Artificial neural network-based flood forecasting in the Indus River Basin using meteorological and hydrological variables," vol. 13, no. 12, p. 2687, 2021.
- [8] C. Lim, M. Bakar, M. Ismail and M. Nor, "A support vector machine-based flood forecasting model for the Klang River basin, Malaysia," vol. 12, no. 8, p. 2030, 2020.

## Appendix

### # Flood prediction KNN model

```
import requests from bs4 import BeautifulSoup
import numpy as np
import pandas as pd
import os
import time
import datetime
import seaborn as sns
import matplotlib.pyplot as plt
from pmdarima.arima import auto_arima
from pmdarima.arima import ADFTest
from neuralprophet import NeuralProphet
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
pd.set_option('mode.chained_assignment', None)
from keras.models import Sequential
from keras.layers import Dense, SimpleRNN, LSTM
from keras.optimizers import RMSprop
from keras.callbacks import Callback
import tensorflow as tf
from sklearn.neighbors import NearestNeighbors
import pickle
import random
```

### ## EDA

```
df = pd.read_csv("ML-Data\Gauging_Stations wise flood datasets\Ratnapura_v2.csv")
df["Date"] = pd.to_datetime(df["Date"])
print(df.dtypes)
df_selection = df[["Date", "rainfall", "Flood_Status"]]
df_selection.isna().sum()
```

```

df_date = pd.DataFrame()
df_date['Date'] = pd.date_range('2009-10-07', '2023-08-07', freq='D')
print("len of full date range",len(df_date))
df_date.head(10)

merge1 = pd.merge(df_date,df_selection,on="Date",how="left")
merge1.head()
merge1.isna().sum()

# fill null rainfall
merge1['rainfall'].interpolate(method='linear', inplace=True)

# fill null flood
merge1['Flood_Status'] = merge1['Flood_Status'].fillna(method='ffill')
merge1.isna().sum()

df_mean = merge1['rainfall'].mean()
df_std = merge1['rainfall'].std()
merge1

## Create train test
df_final = merge1.set_index('Date')
plt.figure(figsize=(20, 5))
mask_1 = (df_selection['Date'] >= '2011-01-01') & (df_selection['Date'] <= '2012-01-01')
df_temp = df_selection.loc[mask_1].sort_values('Date')
df_temp = df_temp.set_index('Date')

plt.plot(df_temp.index, df_temp['rainfall']) # Specify the 'rainfall' column for the y-axis
plt.title('Rainfall Over Time (2011-01-01 to 2012-01-01)')
plt.xlabel('Date')
plt.ylabel('Rainfall')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

## KNN
step = 8

```

```

def custom_encoder(Y):
    if Y == "Flood":
        return [1,0,0]
    if Y == "Alert":
        return [0,1,0]
    # Normal
    return [0,0,1]

def custom_decoder(res):
    results = ["Flood", "Alert", "Normal"]
    max_idx = np.argmax(res)
    return results[max_idx]

def convertToMatrix(df, step):
    X, Y = [], []
    df_x = df["rainfall"].values
    df_y = df["Flood_Status"].values
    for i in range(len(df)-step):
        d=i+step
        X.append(df_x[i:d,:])
        Y.append(custom_encoder(df_y[d,:]))
    return np.array(X), np.array(Y)

x_list,y_list =convertToMatrix(df_final,step)

# therefore need to apply class balance for train set

def balance_df(X,Y):
    balance_fld = []
    balance_alt = []
    balance_nom = []
    for x_ , y_ in zip(X,Y):
        cls_ = custom_decoder(y_)
        if cls_ == "Flood":
            balance_fld.append(list([x_ , y_]))

```

```

elif cls_ == "Alert":
    balance_alt.append(list([x_ , y_]))
elif cls_ == "Normal":
    balance_nom.append(list([x_ , y_]))
else:
    print("None")
per_cls_size = np.max([len(balance_fld),len(balance_alt),len(balance_nom)])
balanced = []
balanced.extend(random.choices(balance_fld,k=per_cls_size))
balanced.extend(random.choices(balance_alt,k=per_cls_size))
balanced.extend(random.choices(balance_nom,k=per_cls_size))
random.shuffle(balanced)
bl_X = []
bl_Y = []
for item in balanced:
    bl_X.append(item[0])
    bl_Y.append(item[1])
Y_balanced = np.array(bl_Y)
X_balanced = np.array(bl_X)
return X_balanced, Y_balanced

x_list,y_list = balance_df(x_list,y_list)
data_list = []
for x,y in zip(x_list,y_list):
    row = [x,y]
    data_list.append(row)
data_df = pd.DataFrame(data_list,columns=['x','y'])
# Shuffle data
data_df = data_df.sample(frac = 1)
# Normalizer
def norm(x:list):
    if np.count_nonzero(x) == 0:

```

```

        return np.zeros(step)

    norm = [float(i)/sum(x) for i in x]
    return norm

x_norm_list = []
for idx,row in data_df.iterrows():
    x_norm_list.append(norm(row["x"]))
data_df["x_norm"] = x_norm_list

# split data
from sklearn.model_selection import train_test_split

train, test = train_test_split(data_df, test_size=0.25)
print("Training data shape:", train.shape)
print("Test data shape:", test.shape)

# Save data
with open('Models/flood_prediction/knn/train.pkl', 'wb') as handle:
    pickle.dump(train, handle)
with open('Models/flood_prediction/knn/test.pkl', 'wb') as handle:
    pickle.dump(test, handle)

# Load saved data
with open('Models/flood_prediction/knn/train.pkl', 'rb') as handle:
    train = pickle.load(handle)
with open('Models/flood_prediction/knn/test.pkl', 'rb') as handle:
    test = pickle.load(handle)

# fit knn model
knn_flood = NearestNeighbors(n_neighbors=1)
knn_flood.fit(list(train["x_norm"].values))
with open('Models/flood_prediction/knn/knn.pkl', 'wb') as handle:
    pickle.dump(knn_flood, handle)

with open('Models/flood_prediction/knn/knn.pkl', 'rb') as handle:

```

```

knn_flood = pickle.load(handle)

## Test

def flood_detector(x):
    norm_x = norm(x)
    result = knn_flood.kneighbors([norm_x])
    dist , idx = result
    idx = idx.item()
    df_row = train.iloc[[idx]]
    flood_status = df_row["y"].values.item()
    return custom_decoder(flood_status)

tp_count = 0

for idx,row in test.iterrows():
    test_in = row["x_norm"]
    result = flood_detector(test_in)
    ground_truth = custom_decoder(row["y"])
    if result==ground_truth:
        tp_count+=1


true_positive_count = 0
false_positive_count = 0
true_negative_count = 0
false_negative_count = 0


for idx, row in test.iterrows():
    test_in = row["x_norm"]
    result = flood_detector(test_in)
    ground_truth = custom_decoder(row["y"])

    if result == ground_truth:
        if result == "Flood":
            true_positive_count += 1

```



```

        else:
            true_negative_count += 1
    else:
        if result == "Flood":
            false_positive_count += 1
        else:
            false_negative_count += 1

total_samples = len(test)
accuracy = (true_positive_count + true_negative_count) / total_samples
precision = true_positive_count / (true_positive_count + false_positive_count)
recall = true_positive_count / (true_positive_count + false_negative_count)
f1_score = 2 * (precision * recall) / (precision + recall)

print(f"Accuracy: {accuracy:.2%}")
print(f"Precision: {precision:.2%}")
print(f"Recall: {recall:.2%}")
print(f"F1 Score: {f1_score:.2%}")

# accuracy
round(tp_count/len(test),2)*100

from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score,
confusion_matrix

# Calculate classification metrics
accuracy = accuracy_score(actual_values, predicted_values)
precision = precision_score(actual_values, predicted_values, average='weighted')
recall = recall_score(actual_values, predicted_values, average='weighted')
f1 = f1_score(actual_values, predicted_values, average='weighted')
conf_matrix = confusion_matrix(actual_values, predicted_values)

# Print classification metrics

```

```

print(f"Accuracy: {accuracy:.2%}")
print(f"Precision: {precision:.2%}")
print(f"Recall: {recall:.2%}")
print(f"F1 Score: {f1:.2%}")
print("Confusion Matrix:")
print(conf_matrix)

## Inference

# load model
with open('Models/flood_prediction/knn/knn.pkl', 'rb') as handle:
    knn_flood = pickle.load(handle)

# load key points
with open('Models/flood_prediction/knn/train.pkl', 'rb') as handle:
    train = pickle.load(handle)

# knn inference
def flood_detector(x):
    norm_x = norm(x)
    result = knn_flood.kneighbors([norm_x])
    dist , idx = result
    idx = idx.item()
    df_row = train.iloc[[idx]]
    flood_status = df_row["y"].values.item()
    return custom_decoder(flood_status)

# normalizer
def norm(x:list):
    if np.count_nonzero(x) == 0:
        return np.zeros(step)

    norm = [float(i)/sum(x) for i in x]
    return norm

# use normalized vector
input_x = [0.06289842753931152,

```

```
0.30939226519337015,  
0.027199320016999574,  
0.07522311942201444,  
0.05057373565660858,  
0.07522311942201444,  
0.07947301317467062,  
0.3200169995750106]  
#  
input_x_norm = norm(input_x)  
result = flood_detector(input_x_norm)  
print(result)
```