**RAINFALL PREDICTION USING ARIMA AND LINEAR REGRESSION**

**A PROJECT REPORT**

***Submitted by***

**SUPRIYA S 211418104275**

**SURYA P 211418104278**

**SWETHA N 211418104284**

***in partial fulfillment for the award of the degree of***

**BACHELOR OF ENGINEERING**

***in***

**COMPUTER SCIENCE AND ENGINEERING**



**PANIMALAR ENGINEERING COLLEGE**

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

**MAY 2022**

**PANIMALAR ENGINEERING COLLEGE**

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

**BONAFIDE CERTIFICATE**

Certified that this project report **“RAINFALL PREDICTION USING ARIMA AND LINEAR REGRESSION”** is the bonafide work of **SUPRIYA S [REGISTER NO: 211418104275], SURYA P [REGISTER NO: 211418104278], SWETHA N [REGISTER NO: 211418104284]”** who carried out the project work under **Dr.K.SANGEETHA,M.E., Ph.D.** supervision.

**SIGNATURE SIGNATURE**

**Dr.S.MURUGAVALLI,M.E.,Ph.D., Dr.K.SANGEETHA,M.E., Ph.D.,**

**HEAD OF THE DEPARTMENT SUPERVISOR**

**ASSOCIATE PROFESSOR**

DEPARTMENT OF CSE, DEPARTMENT OF CSE,

PANIMALAR ENGINEERING COLLEGE, PANIMALAR ENGINEERING COLLEGE,

NASARATHPETTAI, NASARATHPETTAI,

POONAMALLEE, POONAMALLEE,

CHENNAI-600 123. CHENNAI-600 123.

Certified that the above mentioned students were examined in End Semester project viva-voice held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**DECLARATION BY THE STUDENT**

We, Supriya S (211418104275), Surya P (211418104278), Swetha N (211418104284) hereby declare that this project report titled “Rainfall Prediction Using Arima And Linear Regression”, under the guidance of Dr.K.Sangeetha, M.E., Ph.D., is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

**1.SUPRIYA S**

**2.SURYA P**

**3.SWETHA N**

**ACKNOWLEDGEMENT**

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our Directors **Tmt.C.VIJAYARAJESWARI**, **Dr.C.SAKTHI KUMAR,M.E.,Ph.D** and **Dr.SARANYASREE SAKTHI KUMAR B.E.,M.B.A.,Ph.D.,** for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr. S.MURUGAVALLI , M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank my **Dr.K.SANGEETHA ,M.E., Ph.D.,** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

**SUPRIYA S**

**SURYA P**

**SWETHA N**

**ABSTRACT**

Rainfall is the greatest of nature's gifts for our daily life, as well as the most important climate factor affecting human lives with farmers and agricultural complex systems. Rainfall forecasting is critical because excessive and irregular rainfall can have numerous consequences, such as crop destruction and property damage, so a better forecasting model is required for early warning that can limit risks to life and property while also better managing agricultural farms. Time series data have been used extensively in classical statistics. The proposed methodology predicts annual rainfall by time series ARIMA model and Linear Regression a machine learning algorithm. Time series data have been used extensively in classical statistics. The ARIMA has been trained to produce excellent outcomes. The ARIMA model demonstrated greater accuracy in all seasonal and yearly rains. To offer a solid prediction, this method, like time series ARIMA, requires a strict assumption of stationarity. We use real data from the Indian government website and Kaggle to compare model quality in ARIMA using different evaluation metrics. As a result, the ARIMA model accurately forecasts rainfall with less error, and the resultant model can be used to forecast rainfall for future years.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT** | v |
|  | **LIST OF TABLES** | viii |
|  | **LIST OF FIGURES** | viii |
|  | **LIST OF ABBREVIATIONS** | Ix |
| **1.** | **INTRODUCTION** | 1 |
|  | 1.1 Overview | 2 |
|  | 1.2 Problem Definition | 3 |
| **2.** | **LITERATURE SURVEY** | 5 |
| **3.** | **SYSTEM ANALYSIS** | 15 |
|  | 3.1 Existing System | 16 |
|  | 3.2 Proposed system | 16 |
|  | 3.3 Feasibility Study | 17 |
|  | 3.4 Hardware Environment | 20 |
|  | 3.5 Software Environment | 20 |
| **4.** | **SYSTEM DESIGN** | 21 |
|  | 4.1 Data Flow Diagram | 22 |
|  | 4.2 UML Diagrams | 24 |
| **5.** | **SYSTEM ARCHITECTURE** | 29 |
|  | 5.1 Module Design Specification | 30 |
|  | 5.2 Algorithms | 36 |
| **6.** | **SYSTEM IMPLEMENTATION** | 38 |
|  | 6.1 Client-side coding | 39 |
|  | 6.2 Server-side coding | 42 |
| **7.** | **PERFORMANCE ANALYSIS** | 54 |
|  | 7.1 Results and Discussion | 55 |
|  | 7.2 Performance Analysis | 56 |
| **8.** | **CONCLUSION** | 60 |
|  | 8.1 Conclusion and Future Enhancements | 61 |
|  | **APPENDICES** | 62 |
|  | A.1 Sample Screens. | 63 |
|  | **REFERENCES** | 65 |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **Table No** | **Table Description** | **Page No** |
| 5.1 | Data samples | 31 |
| 5.2 | Dataset features | 32 |
| 5.3 | Required features | 32 |
| 7.1 | Regression performance | 55 |
| 7.2 | ARIMA performance | 57 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Fig No** | **Figure Description** | **Page No** |
| 4.1 | Level 0 data flow diagram | 22 |
| 4.2 | Level 1 data flow diagram | 23 |
| 4.3 | Level 2 data flow diagram | 23 |
| 4.4 | Use case diagram | 24 |
| 4.5 | Class diagram | 25 |
| 4.6 | Sequence diagram | 26 |
| 4.7 | Activity diagram | 27 |
| 5.1 | System architecture | 30 |
| 5.2 | ARIMA flow diagram | 34 |
| 7.1 | ARIMA summary | 55 |
| 7.2 | R-squared percentage | 56 |
| 7.3 | Maximum temperature | 57 |
| 7.4 | Minimum temperature | 58 |
| A.1 | Dataset file | 61 |
| A.2 | Current weather | 61 |
| A.3 | Historic data | 62 |
| A.4 | Add location | 62 |
| A.5 | Prediction | 63 |
| A.6 | Regression | 63 |
| A.7 | ARIMA | 64 |

**LIST OF ABBREVIATIONS**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **ABBREVIATIONS** | **EXPANSION** |
| 1 | ARIMA | Auto-Regressive Integrated Moving Average |
| 2 | ACF | Autocorrelation Function |
| 3 | BDTR | Boosted Decision Tree Regression |
| 4 | BLR | Bayesian Linear Regression |
| 5 | DFR | Decision Forest Regression |
| 6 | DT | Decision Tree |
| 7 | PACF | Partial Autocorrelation Function |
| 8 | NNR | Neural Network Regression |
| 9 | MLR | Multiple Linear Regression |
| 10 | LR | Linear Regression |
| 11 | ML | Machine Learning |
| 12 | SARIMA | Seasonal Auto Regressive Integrated Moving Average |
| 13 | MSE | Mean Squared Error |
| 14 | RMSE | Root Mean Squared Error |
| 15 | DRCF | Dynamic Regional Combined short-term rainfall Forecasting approach |
| 16 | MLP | Multi-layer Perceptron |
| 17 | PCA | Principal Component Analysis |
| 18 | KNN | K Nearest Neighbor |
| 19 | SVM | Support Vector Machine |
| 20 | SVR | Support Vector Regression |
| 21 | ANN | Artificial Neural Network |
| 22 | RBF | Radial Basis Function |
| 23 | MAE | Mean Absolute Error |
| 24 | GRU | Gate Recurrent Unit |
| 25 | RF | Random Forest |
| 26 | XGB | Extreme Gradient Boosting |
| 27 | VAE | variable auto-encoder |
| 28 | NumPy | NUMeric Python |
| 29 | SciPy | SCIentific Python |
| 30 | SymPy | SYMbolic Python |
| 31 | Matplotlib | PLOTtingLIBrary |
| 32 | Pytest | Python TESTing |
| 33 | WSGI | Web Server Gateway Interface |
| 34 | LOC | Lines of Code |
| 35 | DFD | Data Flow Diagram |
| 36 | UML | Unified Modeling Language |
| 37 | CSV | Comma Separated Value |
| 38 | NAN | Not A Number |
| 39 | AR | Auto Regressive |
| 40 | MA | Moving Average |
| 41 | AIC | Akaike Information Criteria |
| 42 | MAD | Median Absolute Deviation |

**INTRODUCTION**

**CHAPTER 1**

**INTRODUCTION**

The most essential component in today's world condition is the prediction of rainfall. Unforeseen weather calamities cause significant losses to both humans and the country's economy. Flooding is caused by unexpectedly heavy rainfall, which has far-reaching consequences such as crop destruction and property damage. For the past few years, unanticipated shifts in the rainfall pattern have been a big challenge for the meteorological department. Accuracy in rainfall forecast is critical for countries like India, whose core occupation is agriculture and on which the country's economy is heavily reliant. Agriculture is vital to India's well-being. Agriculture's success is based on rainfall. It also aids in the conservation of water resources. Rainfall information in the past helps farmers better manage their crops, leading to economic growth in the country. As a result, a better forecasting model is required for early warning that can reduce hazards to life and property.

* 1. **OVERVIEW**

The rainfall analysis aids in the prevention and reduction of disasters such as droughts and floods. We can simply do it here by analysing historical rainfall data and forecasting rainfall for future seasons. According to the needs, we may apply a variety of approaches such as classification and regression, and we can also calculate the error between the real and predicted values as well as the accuracy. Because different methodologies yield varying degrees of accuracy, it is critical to select the appropriate algorithm and model in accordance with the requirements.

The machine learning technology was used to predict rainfall. Machine learning allows systems to learn and improve from experience without being designed by humans. The development of the machine learning idea has greatly simplified data analysis and prediction. Machine learning does not necessitate an understanding of the physical mechanisms that govern the atmosphere, but rather analyses past data to forecast future data. As a result, this procedure could be utilized to forecast weather. Rainfall data is time-series data that fluctuates as the climate and seasons change. The goal of this project is to predict rainfall using Linear Regression (LR) Auto-Regressive Integrated Moving Average (ARIMA) approaches. The parameters of the LR equation are obtained from the dataset, and the variables are derived from the dataset via correlation. ARIMA is an effective statistical technique for predicting rainfall modelling time series. The ARIMA model development technique includes iterative stages of discovery, estimation, diagnosis, and forecasting. Once chosen, the model is used to forecast monthly or seasonal rainfall series.

Timeseries forecasting is important for making predictions and informed strategic decisions, as well as predicting future rainfall using previously observed data. Time series analysis is a common approach to multivariate statistical analysis. It is used to forecast rainfall based on rainfall time series features. Exploration of time series offers numerous options for discovering, describing, and modelling climate inconsistencies and impacts. Modelling can be done using historical weather data collected by meteorological stations located throughout India.

* 1. **PROBLEM DEFINITION**

Rainfall prediction is a vital part of weather forecasting. Observing environmental shifts, particularly climatic variations, is a critical component of building the future. Predicting rainfall accurately is one of the most difficult issues faced by meteorologists all around the world. Researchers have used several ways to assess various meteorological features. Predicting rainfall is a scientific and technological application that predicts the amount of rain that will fall over a specific area. The most important issue is to accurately predict rainfall for active use of rainfall for water resources, crops, water resource planning, and agricultural objectives.

Rainfall is the major input to the river basin, influencing stream water capacity, particularly during severe rain events. Furthermore, one of the primary goals of climate change research is to determine whether there is a significant change in the occurrence and frequency of heavy rainfall events. Prediction of precipitation is useful in preventing flooding, which saves lives and property. Forecasting rainfall is difficult for meteorologists due to fluctuations in the time and amount of precipitation.

Rainfall forecasting is a difficult task, and the results must be precise. There are numerous hardware devices available for predicting rainfall based on weather parameters such as temperature, humidity, and pressure. Because traditional approaches are inefficient, we can achieve accurate results by employing machine learning algorithms. One of the most essential problems in hydrological studies has been the accuracy level of the ML models used in projecting rainfall based on historical data. An accurate ML forecasting model could provide early warnings of severe weather, assisting in the prevention of natural disasters and destruction. As a result, ML algorithms capable of forecasting rainfall with acceptable precision and minimizing error in the dataset of projected rainfall from climate change models with expected observable rainfall are required.

**LITERATURE SURVEY**

**CHAPTER 2**

**LITERATURE SURVEY**

**2.1 Prediction of Short-Time Rainfall Based on Deep Learning.[5]**

**AUTHOR -** Dechao Sun, Jiali Wu, Hong Huang, Renfang Wang, Feng Liang, and Hong Xinhua.

**YEAR –** 2021

Short-term heavy rain is a type of severe and severe weather that suddenly threatens human health and property safety. This work offers a convolutional 3D GRU (Conv3D-GRU) model to forecast future rainfall intensity in the region, which is superior to machine learning vision in terms of accuracy. Radar echo map extrapolation technology is the main technology for rainfall distribution now. According to the recognized echo maps, the intensity of the echo distribution and the moving speed of the echo body and the direction (such as rainfall) are determined. echo in the timeline is coded and determined using GRU. Finally, a trained model is used to predict radar echo maps for the next 1-2 hours. The algorithm successfully extracts temporary and local aspects of radar echo maps, reduces the error between anticipated and actual rainfall, and improves the accuracy of short-term rainfall predictions, according to test findings. Echo maps, reduce the accuracy of short-term rainfall. the error between the predicted value and actual rainfall, and improved the accuracy of short-term rain forecast.

**MERITS -** The experimental results show that the algorithm can effectively extract the temporal and spatial features of radar echo maps, reduce the error between the predicted value and the real value of rainfall, and improve the accuracy of short-term rainfall prediction.

**DEMERITS -** The features like meteorological features, temperature, wind field are not included.

**2.2Rainfall Prediction Using Machine Learning Algorithms for the Various Ecological Zones of Ghana.[7]**

**AUTHOR -** Nana Kofi Ahoi Appiah-Badu, Yaw Marfo Missah, Leonard K. Amekudzi, Najim Ussiph, Twum Frimpong, And Emmanuel Ahene.

**YEAR -** 2021

Due to climate change and variability, predicting accurate rainfall patterns has become increasingly challenging. Differentiating algorithms' efficiency in rainfall forecasts has increased. The findings propose that different categorization methods be used to anticipate rainfall in different locations of Ghana. Separation algorithms include Decision Tree (DT), Random Forest (RF), Multilayer Perceptron (MLP), Extreme Gradient Boosting (XGB), and K-Nearest Neighbor (KNN). From 1980 to 2019, the Ghana Meteorological Agency provided the data, which covers a number of climatic parameters. In a range of equilibrium training and data processing circumstances, the effectiveness of classification algorithms was evaluated using accuracy, memory, f1 scores, and timeliness. The Decision Tree is consistently presented as the fastest in terms of modelling time, whereas the MLP is consistently presented as the most time-consuming. However, the K-Nearest neighbour has done the worst in all areas in all training and testing standards that require further investigation.

**MERITS -** Decision Tree is consistently portrayed as the fastest, whereas MLP used the most run time.

**DEMERITS** - K-Nearest Neighbour performed worst in all zones on all training and testing ratios.

**2.3 Rainfall Prediction using Machine Learning & Deep Learning Techniques.[3]**

**AUTHOR -**Cmak Zeela Basha, Nagulla Bhavana, Ponduru Bhavya, Sowmya V.

**YEAR –** 2020

Rainfall forecasting using machine learning techniques is quite accurate. Some of the more common machine learning approaches are the ARIMA Model (Auto-Regressive Integrated Moving Average), Artificial Neural Network, Reversing Objects, Vector Support Machine, and Editing Maps. Similar to linear and non-linear models, two of the most extensively used models forecast seasonal rainfall. The ARIMA model is the first model. Rain forecasting can be done using Back Propagation NN, Cascade NN, or Layer Recurrent Network when utilizing the Artificial Neural Network (ANN). Biological Neural Networks are similar to Artificial Neural Networks. The primary assumption is to acquire concrete and perhaps costly information while following the principles of information and relationships. Synthetic NN is a promising component in this big field. Rain is anticipated by utilizing in-depth reading methods in this paper. Multilayer Perceptron and Auto-Encoder were two in-depth reading algorithms used. Auto-Encoders are responsible for predicting time series by performing an output feature. Comparing the current structure with other state systems. The results aim that according to the MSE and RMSE, our proposed facilities exceed the remaining routes. Accuracy can be measured with MSE and RMSE in comparison with other models. This study examines the various types of rainfall prediction systems as well as the issues that can arise when employing different rainfall forecast methods. Artificial Neural Networks outperform all other methods because of their indirect relationship to rain data sets and ability to learn from the past.

**MERITS -** Artificial Neural Network makes a superior solution to all approaches available.

**DEMERITS -** It will be difficult to predict small changes in climate.

**2.4Prediction Of Rainfall Using Machine Learning Techniques.[6]**

**AUTHOR -** Moulana Mohammed, Roshitha Kolapalli, Niharika Golla, Siva Sai Maturi.

**YEAR –** 2020

The forecast model is used to forecast rain. The first step is to transform the data into the appropriate format for testing. Then, to see the variety of rain patterns, conduct a detailed data analysis. The forecast rainfall by splitting the data into a training and test set, then comparing and analyzing the results using a range of machine learning techniques (MLR, SVR, and so on) and computational methods. SVR is expected to assist the customer in overcoming challenges such as important feature distribution, information geometry, and general model issue over measurement in this project, which focuses on rainfall estimations. The SVR display is used to determine the bit rate resolution. In direct and indirect interactions, the author describes soft feet for the usage of a straight piece and an RBF. As expected, the author forecasts that SVR will outperform MLR. Because MLR cannot detect non-linearity in the data set, SVR is effective in these situations. They also looked at MLR and SVR models Total Maximum Error (MAE) to see how well they worked. In actuality, the SVR adjusted model produces the best outcomes.

**MERITS** - This project is to offer non-experts easy access to the techniques, and approaches utilized in the sector of precipitation prediction and provide a comparative study among the various machine learning techniques.

**DEMERITS -** The main challenge is to build a model for long term rainfall prediction.

**2.5 Rain Prediction Using Rule-Based Machine Learning Approach.[10]**

**AUTHOR -** Uchamad Taufiq Anwar, Saptono Nugrohadi, Vita Tantriyati, Vikky Aprelia Windarni.

**YEAR -** 2020

The rain forecast model is very useful for human activities. This study aims to develop a rain forecast model using a data mining method using historical data. This study aims to develop a rainfall forecasting model using a legal machine learning method using weather history data. Testing using the J48 method resulted in up to 77.8% accuracy in the training model and provided 86% accurate predictor results when tested compared to actual 2020 weather data. The model also showed that the features that predict the rainfall most are moderate humidity. (RH\_avg), followed by low temperature (Tn). The high accuracy achieved by the J48 method is consistent with other studies that said that the Decision Tree model is better compared to other predictable results. viewing. This result has given us a better understanding of the rainy season and the model may be used for a number of purposes such as agriculture and transport.

**MERITS -** The experiment using the J48 method resulted in up to 77.8% accuracy in the training model and gave accurate prediction results of 86% when tested against actual weather data in 2020.

**DEMERITS -** A small change in the data can result in a major change in the structure of the decision tree, which can convey a different result from what users will get in a normal event.

**2.6 Rainfall forecasting model using machine learning methods: Case study Terengganu, Malaysia.[11]**

**AUTHOR -** Wanie M. Ridwan, Michelle Sapitang, Awatif Aziz,Khairul Faizal Kushiar, Ali Najah Ahmed, Ahmed El-Shafie.

**YEAR –** 2020

Unexpected rainfall due to climate change can cause either overflow or drying in a water storage area. In this study, many models and methods were present a application for predicting rain data in Tasik Kenyir, Terengganu. The study looked at how to build and compare many Machine Learning models (ML), as well as how to predict rain using two different methodologies, in diverse situations and time frames. The data involved in this study include to measure rainfall at 10 stations in the study area using Thiessen polygon to measure station area and proposed rainfall. Bayesian Linear Regression (BLR), Boosted Deception Tree Decline (BDTR), Decision Forest Retreat (DFR), and Neural Network Depression are the four ML techniques employed in the prediction model (NNR). The study also includes two methodologies for predicting precipitation rain: autocorrelation function (ACF) and expectancies error rain forecasting. This study is critical for bettering water management, particularly in Kenyir. The lake's actual location.

**MERITS -** Forecasting Rainfall Using Autocorrelation Function shows high accuracy.

**DEMERITS -** Accurate rainfall prediction might be achieved by proposing hybrid machine learning algorithms and with the inclusion of different climate change scenarios.

**2.7 Weather Forecasting Using Machine Learning Algorithm.[8]**

**AUTHOR -** Nitin Singh, Saurabh Chaturvedi, Shamim Akhter.

**YEAR -** 2019

Developing a remote weather forecast system is the main reason for this work. Data statistics and machine learning algorithms, such as random forest classification, are used to predict weather conditions. In this paper, an inexpensive and practical solution for weather forecasting is developed. This paper introduces rain predictions using real-time data on temperature, humidity, and pressure using a variety of sensors. It works by making multiple decision trees while training a set of data and outputting a separate tree separation mode. In the proposed system, the app is built on the Raspberry Pi 3 B. This application receives real-time data from temperature and humidity sensors to predict rainfall today. The main goal of this project is to use the Python machine learning idea on the Raspberry Pi board to create a low-cost, dependable, and effective weather forecasting software.

**MERITS -** In this paper, a low-cost and portable solution for weather prediction is devised.

**DEMERITS** - The size of the test set is 1835, out of which the number 1491 represents the count of correct predictions that rain will not happen and 122 is the number of correct predictions that rain will happen.

**2.8 Rainfall Prediction Using Machine Learning.[3]**

**AUTHOR -** Arnav Garg, Himanshu Pandey.

**YEAR –** 2019

As global warming increases visibility and rainfall become a major problem for countries that do not have access to appropriate technology and that if done properly can help them with a few goals such as farming, health, drinking, and much more. And for this purpose the author predicts next year's rain using SVR, SVM, and KNN machine learning algorithms and compares the results obtained with each algorithm. Although all algorithms namely SVM, SVR, and KNN are equally useful, in rainfall prediction SVM and SVR are much more accurate than KNN, and although SVR provides almost complete when considering all the confusing and biological biases they conclude by predicting rain SVM. is the best of the three and the best way to use it is to make the range of predicted values ​​very high and low by adding bias to the model.

**MERITS -** Though all the algorithms i.e. SVM, SVR and KNN are equally useful, for rainfall prediction SVM and SVR are more accurate than KNN.

**DEMERITS -** Even though SVR provides near perfect if we consider all the anomalies and biases found in nature.

* 1. **Short-term Rainfall Forecasting Using Multi-layer Perceptron.[9]**

**AUTHOR –** Pengcheng Zhang, Yangyang Jia, Jerry Gao, Wei Song, Hareton Leung.

**YEAR –** 2018

To improve the overall accuracy of the short-term rain forecast, this paper proposes a novel solution called the Dynamic Regional Combined short-term rainfall Forecasting approach (DRCF) using the Multi-layer Perceptron (MLP). First, Principal Component Analysis (PCA) is used to reduce the size of thirteen materials, which acts as an MLP input. Second, a greedy algorithm is used to determine the composition of the MLP. Highly visible features play an important role in the flow of rain systems. Surface factors also cause different rainfall. The Dynamic Regional Combined (DRCF) short-term rainfall prediction model is proposed in this paper using the Multi-Layer Perceptron to handle the aforementioned (MLP). The model input includes five high-altitude features and eight additional features from the target area and surrounding areas. Over the next three hours, outputs are aimed at local rainfall. The study compared the DRCF to space models and other machine learning approaches using data from 56 real-world meteorology stations in China. Test results show that DRCF exceeds existing methods in both threat points (TS) and root mean square error (RMSE).

**MERITS –** To solve the clutter interference which is caused by the extension of the perception range, DRCF is enhanced with several dynamic strategies

**DEMERITS -** The forecasting interval of DRCF is limited to 3 hours and the greedy algorithm can only decide the optimal structure of local MLP, but cannot get the global optimal structure**.**

**2.10 Time series analysis model to forecast rainfall for Allahabad region.[2]**

**AUTHOR -** Anosh Graham and Ekta Pathak Mishra

**YEAR –** 2017

The box-jerkins seasonal ARIMA model is used to model and predict rainfall in this research. Rainy weather is critical to India's agricultural practices and crop production; in fact, this season accounts for the majority of the country's yearly rainfall. The small variation in time and the amount of heavy rainfall have the potential to have an impact on agricultural production. Rainfall is a very important climate factor that affects agriculture. Monthly rainfall forecasts play a vital role in planning and managing agricultural systems and water resources. The rainfall and temperature forecast is made using the central regressive integrated motion system in the middle. Modelling and forecasting rainfall are done in a mathematical way based on the autoregressive integrated moving average (ARIMA). The coefficient of equilibrium (R2) and root mean square were used to measure the performance of the accepted models (RMSE). The results show that the ARIMA model of the season provides a consistent satisfactory prediction of rainfall limits on a monthly basis.

**MERITS -** The seasonal ARIMA model provide consistent and satisfactory predictions for rainfall parameters on a monthly scale.

**DEMERITS-** The accuracy of these predictions can be increased in the future by using these predicted values for missing values**.**

**SYSTEM ANALYSIS**

**CHAPTER 3**

**SYSTEM ANALYSIS**

**3.1 EXISTING SYSTEM**

Time series analysis is utilized to handle problems like drought, flood, and other agricultural difficulties, and a machine learning model is used to anticipate the future values of Tamil Nadu Monsoon rainfall. In Tamil Nadu, statistical methods such as the Seasonal Arima model are used to forecast rainfall time series. The rainfall prediction model exhibited the stationarity of the time series flow and analyzed the seasonal correlogram using the SARIMA (Seasonal Auto-Regressive Integrated Moving Average). This model's performance is measured using measures such as Mean Squared Error (MSE) and Root Mean Squared Error (RMSE). The model that was developed could be used to forecast Monsoon rainfall.

**DISADVANTAGE**

* A large number of studies have gone into determining how rainfall influences agriculture.
* However, the majority of these researchers necessitate extensive, sophisticated data that is not readily available.

**3.2 PROPOSED SYSTEM**

The proposed methodology evaluated India's rainfall data and forecasted future rainfall using linear regression and the ARIMA algorithm. Rainfall data were obtained from the Indian government's official website and Kaggle. Rainfall measurements in India have been taken for over a decade. Technical indicators, fundamental analysis, Fourier transformations, linear regression, and the ARIMA model were all used in the proposed approach to identifying features. All of the seasonal and annual rainfall forecasts were more accurate with the ARIMA model. We analyze the model quality in ARIMA based on numerous factors using real data from the Indian government website and Kaggle.

**3.3 FEASIBILITY STUDY**

**3.3.1 TECHNICAL FEASIBILITY**

This project is about predicting rainfall. The project's time series analysis model is useful for anticipating future data characteristics. Machine Learning methods and approaches used in the research will be beneficial, with a high accuracy score and a good forecast of rainfall. The system is built with the Python programming language and the Flask framework, which includes libraries, modules, and tools for creating web applications.

**3.3.1.1Why PYTHON is used?**

Python enables developers to be more productive and confident in the software they create. Python's simplicity and consistency, as well as access to excellent libraries and frameworks for AI and machine learning (ML), flexibility, platform freedom, and a large community, make it the best choice for machine learning and AI applications. These factors contribute to the language's overall appeal.

Implementing machine learning algorithms can be difficult and time-consuming. To enable developers to come up with the greatest coding solutions, it's critical to have a well-structured and well-tested environment.

Python frameworks and libraries are used by programmers to reduce development time. A software library is a collection of pre-written code that programmers can utilize to tackle common programming challenges. Python has a large number of libraries for artificial intelligence and machine learning because to its robust technology stack. Here are a few examples:

o For machine learning, Keras, TensorFlow, and Scikit-learn

o NumPy for data analysis and high-performance scientific computing

o Advanced computing using SciPy

o Pandas for data analysis in general

**3.3.1.2 REQUIREMENTS**

The requirements that need to be installed in this project are listed below,

* Click - A click is a Python tool that allows you to create command-line interfaces with as little code as possible. The "Command Line Interface Creation Kit" is what it's called. It's incredibly customizable, but it also comes with decent defaults.
* Flask - Flask is a WSGI web application framework that is lightweight. It's built to make getting started simple and quick, with the flexibility to scale up to more sophisticated projects. It started out as a basic wrapper for Werkzeug and Jinja and has since grown to become one of the most popular Python web application frameworks.
* Flask-SQLAlchemy - Flask-SQLAlchemy is a Flask extension that adds SQLAlchemy support to your application. It seeks to make using SQLAlchemy with Flask easier by providing useful defaults and extra utilities that make common tasks easier.
* Itsdangerous - To verify that a token has not been tampered with, data is cryptographically signed. The serialization of data can be customized. As needed, data is compressed. While loading a token, a timestamp can be added and confirmed automatically.
* Jinja2 - Jinja is a templating engine that is quick, expressive, and extendable. The template has placeholders that allow you to write code in Python syntax. After that, data is supplied to the template in order to render the final document.
* MarkupSafe - MarkupSafe creates a text object that is safe to use in HTML and XML because it escapes characters. Ones with particular meanings are replaced to seem like the original characters. This protects against injection attacks, allowing untrusted user input to be shown on a page safely.
* Werkzeug – Werkzeug is a WSGI web application library with a lot of features. It started as a simple collection of WSGI application utilities and has evolved into one of the most powerful WSGI utility libraries.

**3.3.2 ECONOMICAL FEASIBILITY**

This study is being conducted to determine the system's economic impact on the organization. The cost is determined by the hardware and software requirements of our project. The COCOMO model is used to estimate the project's cost.

Total number of lines of code (LOC) = 1300

KLOC=1300/1000=1.3

Effort = 2.4\*( =>3.161 *person-month*

Development time = 2.5 => 3.871 *months*

Person required =3.161/3.871 =>0.816 *person*

Productivity = 1.3/3.161 => 0.411 *KLOC/ person-month*

P=411 LOC/person-month

**3.3.3 SOCIAL FEASIBILITY**

The purpose of the study is to determine the user's level of acceptance of the system. This includes teaching the user how to utilize the technology effectively. The user must accept the system as a need rather than feel threatened by it. The methods used to educate and familiarize the user with the system are totally responsible for the level of acceptance by the users. Rainfall is one of the most difficult and unpredictable chores that has a major impact on human culture. Forecasting that is accurate and timely can help to reduce human and financial losses. Therefore, this project is socially feasible.

**3.4 HARDWARE REQUIREMENT**

* RAM - 8 GB
* Software - python idle
* Hard Disk - 1 tb
* Processor - I5 and above

**3.4 SOFTWARE REQUIREMENT:**

* Operating System - Windows 10
* Front End - CSS
* Language - Python(3.10) Version.
* Server - Flask.

**SYSTEM DESIGN**

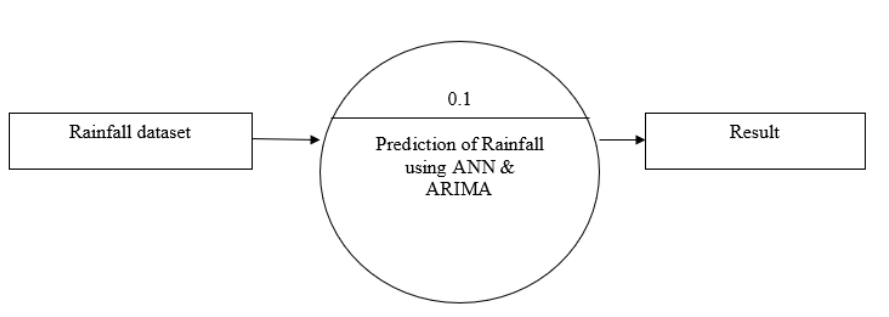
**CHAPTER 4**

**SYSTEM DESIGN**

**4.1 DATA FLOW DIAGRAM**

A data flow (DFD) diagram is a graphical representation of a "flow" of data through an information system, which modifies its process characteristics. DFD is often used as the first step in creating an overview of the system without going into too much detail, which can be explained later.

**4.1.1 LEVEL 0**

Level 0 describes the whole process of the project. We use a rain data set as an input. The system will use the ARIMA and LR algorithm to predict rain and predict future outcomes.

**Regression &**

Fig. No. 4.1 Level 0 data flow diagram

**4.1.2 LEVEL 1**

Level 1 describes the preliminary consideration and process of extracting a project feature. Rainfall data is provided as input and the data is processed in advance. The pre-processing phase involves extracting records with empty values. Once the database has been cleared in the pre-processing phase, we must prepare it to include the feature removal process. The original database contains unwanted information that is not required for model training so, by using the feature removal process only the required information is extracted from the database.

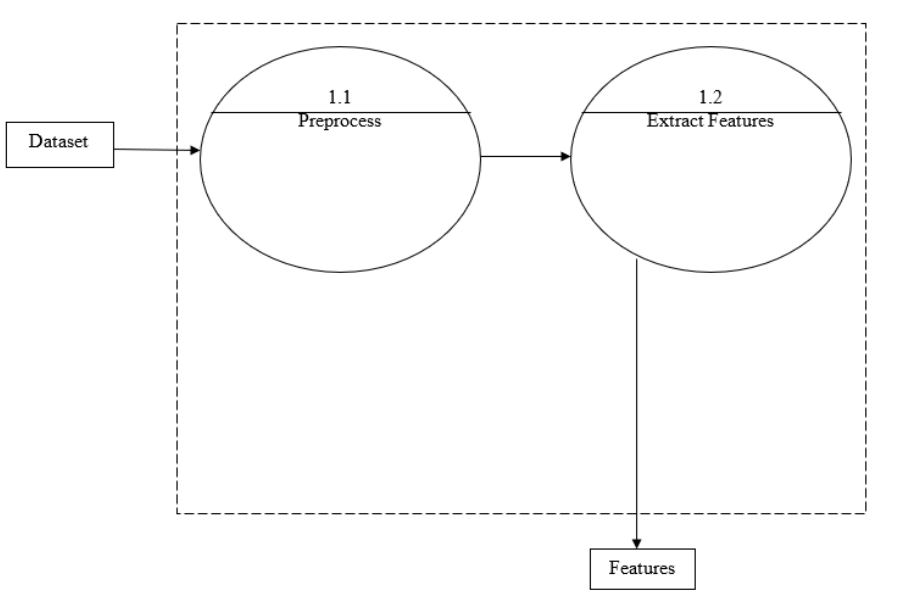


Fig. No. 4.2 Level 1 data flow diagram

**4.1.3 LEVEL 2**

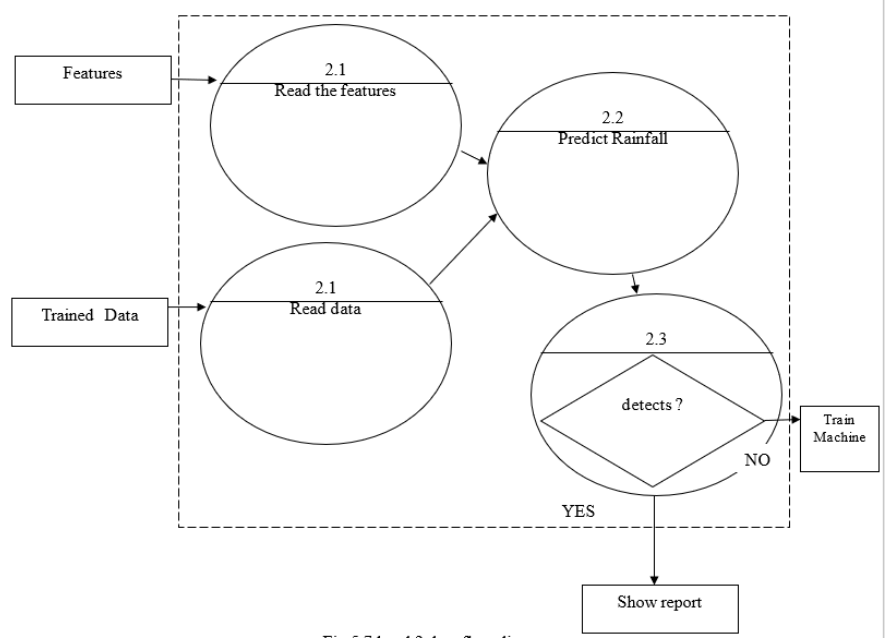
Level 2 describes a more detailed process which is the flow of project work. Once the database is trained it will read feature-based data and user data and predict rainfall. By using the appropriate model, you will get the prediction result.

Fig. No. 4.3 Level 2 data flow diagram

**4.2 UML DIAGRAM**

**4.2.1 USE CASE**

A simple user interface diagram represents the user interaction with a system that shows the relationship between the user and the different operating conditions in which the user is involved. The application case diagram can identify different types of system users and different operating conditions and will often be associated with other types of diagrams. Terms of use are represented by circles or ellipses.

There are two characters in the project manager and program. First, add a data set by the administrator and the system reads the database and processes the data. After pre-processing the domain data element occurs using ARIMA and retrieval. Finally, predict the rain and confirm the forecast result.

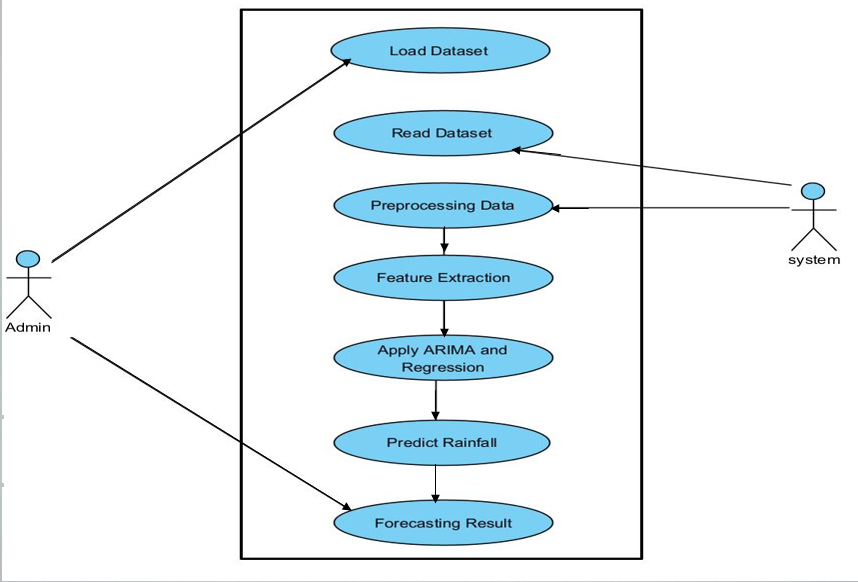


Fig. No. 4.4 Use case diagram

**4.2.2 CLASS DIAGRAM**

In the Unified Modeling Language (UML), a class diagram is a form of static structural diagram that shows the system's classes, attributes, actions (or methods), and relationships among objects. A static diagram is a class diagram. It depicts an application's static view. A class diagram is used not only for visualizing, describing, and documenting many parts of a system but also for creating executable code for a software program A class diagram depicts a class's attributes and operations, as well as the system's limitations.

The class diagram has two classes: admin and system. The load dataset method and the view result method are both parts of the admin class. Read, pre-process, train, and predict rainfall are all part of the system class.

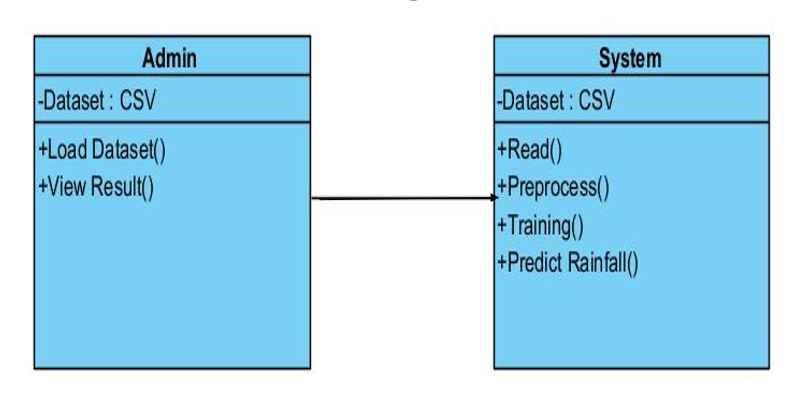


Fig. No. 4.5 Class diagram

**4.2.3 SEQUENCE DIAGRAM**

The UML Sequence Diagrams flowchart shows how the tasks are performed. They capture the interaction between objects in a shared space. Sequential diagrams focus on time and show the order of communication using the direct axis of the drawing to represent the time and messages sent and received. There are two characters in this project: admin and system. After uploading the database, the data is read and processed in advance. Feature extract is used to remove a feature. Released information is trained and predictable. The result was predictable.

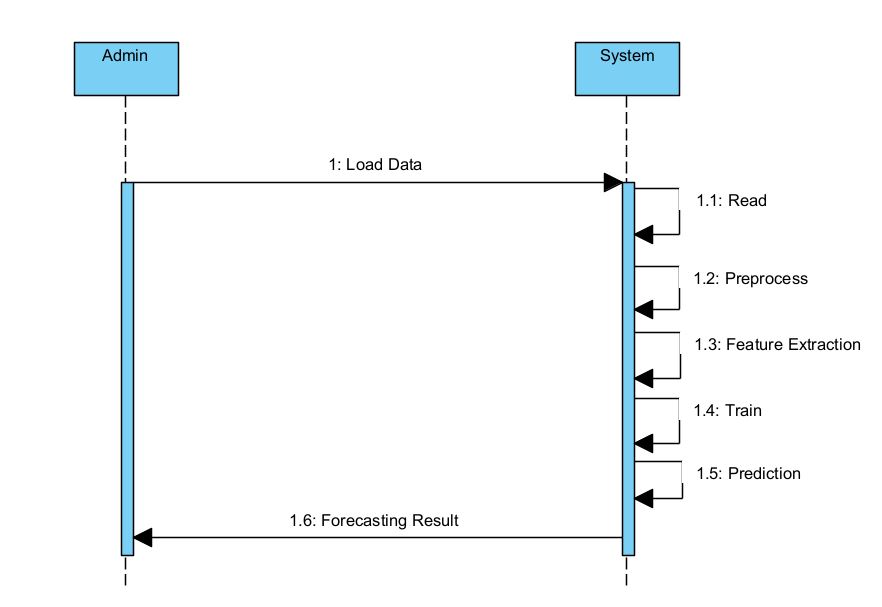


Fig. No. 4.6 Sequence diagram

**4.2.4 ACTIVITY DIAGRAM**

Activity diagrams are widely used as a flowchart showing system functions. Function diagrams are different from flowcharts in that they have additional features. Branches, parallel flow, and swimming pools are examples of these additional skills. Another important diagram in the UML to describe the dynamic features of the system is the function diagram. An activity diagram is a flow chart that shows the movement of information from one to another. The action can be defined as a system activity. From action to action, a flow of control is expressed.

The dataset is loaded and preprocessed in the activity diagram. Preprocessed data is used for feature extraction. Rainfall is predicted using this model, which includes ARIMA and regression.

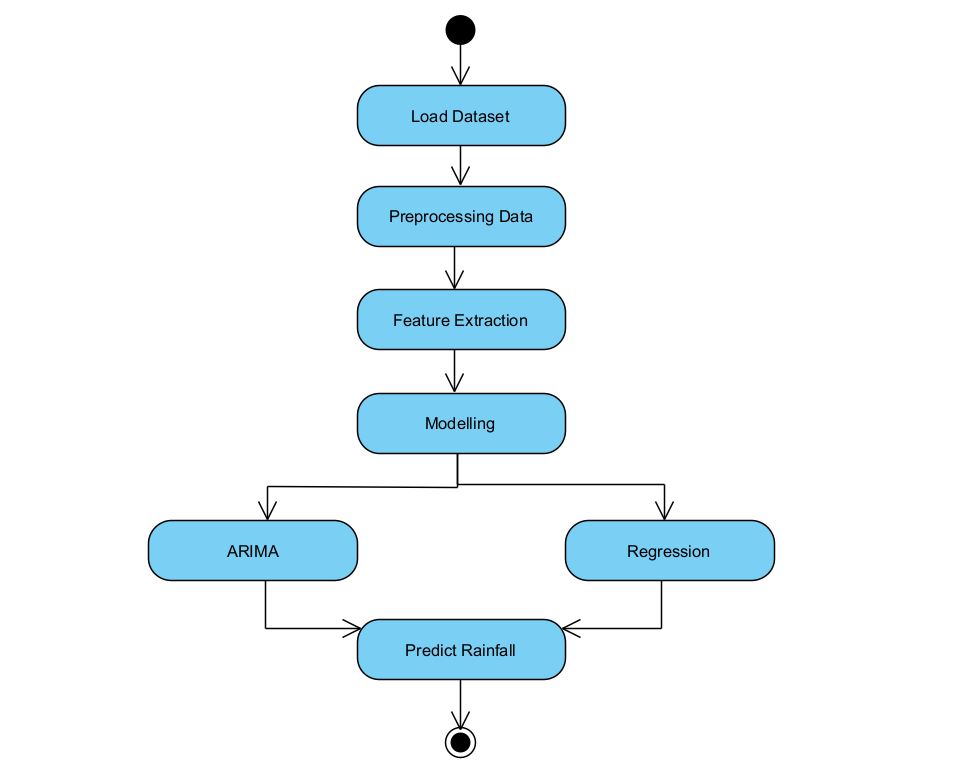


Fig. No. 4.7 Activity diagram

**SYSTEM ARCHITECTURE**

**CHAPTER 5**

**SYSTEM ARCHITECTURE**

Rainfall information is saved in CSV or Excel format. The monthly amount is included in the database. There may be empty values, negative values, or errors in the database. The dataset is cleaned during the pre-processing step. Previous processing methods include the removal of incomplete records. At this stage, we should create a database containing non-NAN (Not A Number) values; if the database already has NAN numbers, they must be replaced with 0 or lowered. We have to fix the cleaned database to go into the feature removal process once it is found.

If you have a large database and need to reduce the number of resources without sacrificing any important or related information, feature removal can help. Because the original database in this project contains undesirable data that isn't needed for model training, the feature removal technique is used to extract only the necessary data.

Thereafter, the database is entered into the ARIMA model. ARIMA is a mathematical analysis model that uses time-series data to predict future values based on previous results. Supervised Learning uses linear regression as a machine learning approach. It carries out the task of regression. A regression model is a numerical value based on an independent variable that predicts the objective.

Following forecasts, the often occurring itemset The frequent item can be retrieved by searching for it in the dataset using the frequent itemset procedure. Temperature, humidity, and rainfall datasets are all part of the frequently used item. Using parameters such as temperature, humidity, and rainfall dataset, the overall future rainfall accuracy is forecasted and validated.

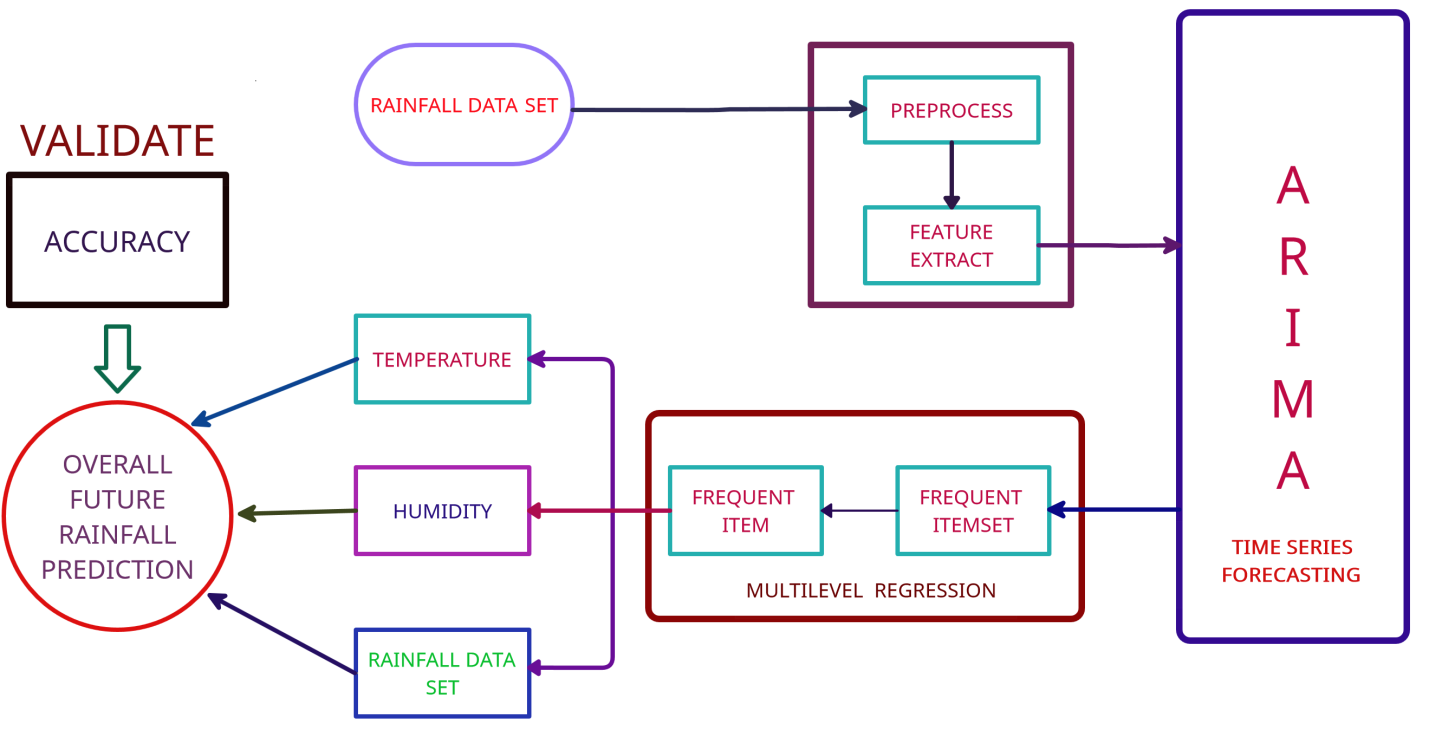
****

Fig. No. 5.1 System architecture

**5.1 MODULE DESIGN SPECIFICATION**

**5.1.1 MODULES**

* Data Collection.
* Data Preprocessing.
* Feature Extraction.
* Linear Regression.
* ARIMA (Auto-Regressive Integrated Moving Average)
* Ensemble Methods and Bagging.

**5.1.1.1 DATA COLLECTION**

Data.gov.in is used to collect random real-time data that is converted to a fixed format. The most important function of all machine learning projects is data collection. Because the data we provide algorithms is input. As a result, the efficiency and accuracy of the algorithms are determined by the accuracy and quality of the data obtained.

Table. No. 5.1 Data samples

|  |  |  |  |
| --- | --- | --- | --- |
|  | FILE FORMAT | DATA COUNT | SOURCE |
| Rainfall | CSV | 44728 | Kaggle |

There are 21 features available in the dataset like time, precip intensity, precip Intensitymax, precip probability, temperature min, temperature max, apparent temperature min, apparent temperature max, latitude, longitude, precip type, time., etc for the prediction of rainfall.

**5.1.1.2 DATA PREPROCESSING**

Data processing is the process of converting raw data into an unintelligible format. There may be empty values, negative values, or errors in the database. During the pre-processing phase, the database is deleted. The pre-processing process involves deleting incomplete records. As a result, it produces consistent and reliable data, which improves the efficiency of training data for analysis and allows for more accurate decision-making. The bfill () method is used in our project to process input and complete empty values.

**bfill()**

The bfill () method replaces NULL values ​​with the following line values. It is used to replenish non-database values. It will fill any NAN values ​​in the pandas data framework.

**5.1.1.3 FEATURE EXTRACTION**

If we have a large database and need to reduce the number of resources without sacrificing any important or related information, removing the feature becomes helpful. The original database contains data that is not required for model training. As a result, the feature removal method only extracts the required information from the original database.

Table. No. 5.2 Dataset features

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| State | Latitude | Longitude | TempMin | TempMax | Precip  Intensity | Precip  Probability |
| Karnataka | 15.31 | 75.71 | 18.62 | 30.26 | 0.0062 | 0.07 |
| Haryana | 29.23 | 76.43 | 12.74 | 28.78 | 0.0034 | 0.02 |
| Tamil Nadu | 11.05 | 78.38 | 23.98 | 32.32 | 0.0149 | 0.09 |
| Kerala | 10.85 | 76.27 | 22.92 | 32.43 | 0.1591 | 0.41 |

Table. No. 5.3 Required features

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| State | TempMin | TempMax | Precip  Intensity | Precip  Probability |
| Karnataka | 18.62 | 30.26 | 0.0062 | 0.07 |
| Haryana | 12.74 | 28.78 | 0.0034 | 0.02 |
| Tamil Nadu | 23.98 | 32.32 | 0.0149 | 0.09 |
| Kerala | 22.92 | 32.43 | 0.1591 | 0.41 |

**5.1.1.4 LINEAR REGRESSION**

The regression algorithm is a method of predicting end results. It is the link between features and results. Algorithms are designed to detect the relationship between the individual elements and the results. A simple algorithm is a linear regression algorithm. The main purpose of the algorithm is to reduce the difference in height. The goal of dropping a line is to produce the type of numbers as the output. It is a way of predicting future results. The linebacker indicates the link between the frequency values ​​of the S and T variables.

A vertical line is a line that connects the prediction with the target variables. This indicates two types of connection. There is a positive linear relationship where the target variable increases on the y axis and the predictive variable rise to the x-axis. There is a positive line correlation where the target variable decreases on the y-axis and the predictive variable decreases on the x-axis.

The best fit line is the one with the least amount of error between predicted and true values; it is derived using least squares regression and gradient descent. There are two forms of linear regression: multiple linear regression and Simple linear regression is a type of regression that uses only one predictor variable to predict the value of a numerical target variable. It works with only one predictor variable. Multiple linear regression is a type of regression in which more than one predictor variable is used to predict the value of a numerical target variable. It works with more than one predictor variable.

**5.1.1.5 ARIMA**

ARIMA is a summary of the Auto-Regressive Integrated Moving Average, which is a predictor of time series. A timeline is a set of observations made over a period of time. There are two types of time series: univariant and multivariant. A fixed time series contains values ​​from one variation, while a different time series contains values ​​derived from several variables.

The p, d, and q parameters in ARIMA are used to reduce the difference between actual values ​​and forecasts. We will use the grid search to improve the model by trying all the possible values ​​and finding the lowest error. If the price range is high, this process takes longer.

AR stands for Auto-Regressive Process, which defines current values ​​and is based on its previous value of p. The order of the AR process is P. The current deviation from the previous q deviation is called the MA process. The order of the MA process is Q.

The timeline must be stable, and we must make it stable if prices are not available. As a result, the parameter d = 1 needs to be set. Because it eliminates the growing trend, this is known as the first order split.

ACF represents the function of automatic integration. The log is used to set the contact area. The point of correlation between these two variants is determined by the Partial Auto Correlation Function (PACF).

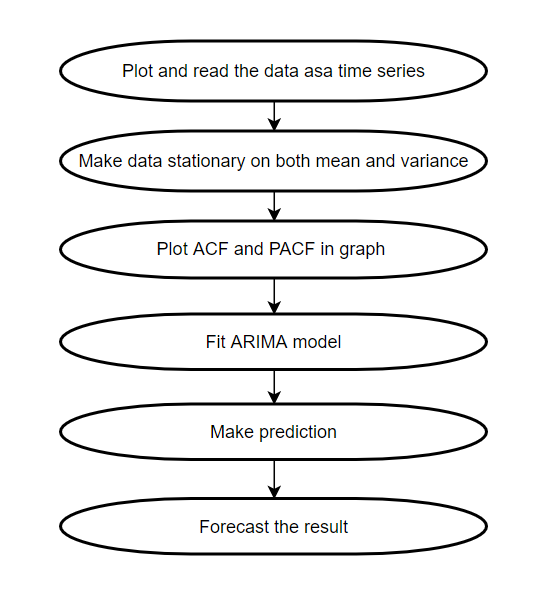


Fig. No. 5.2 ARIMA flow diagram

**STEPS IN ARIMA**

**Step 1:** Creating Models

This stage carefully selects a standard ARIMA formulation to model the rainfall data. This is done by carefully inspecting and selecting the most important characteristics of rainfall intensity and other weather parameters. Precipitation and rainfall are the data sources for this project.

**Step 2:** Identifying the Model

For the rainfall data, a trial model must be identified. To begin, the original rainfall data must be transformed in order to render the underlying mechanism stationary. The autocorrelation function (ACF) or unit root test can be used to check the data in this stage. Additional lag residual and lag dependent tests were performed using partial ACF.

**Step 3:** Estimation of Parameters

The parameters of the model's elements must be evaluated after the model's elements have been determined. Assuming the data are observations of a stationary time series, good parameter estimators can be derived (Step 2). If a Moving Average (MA) pattern is found, the next step is to use maximum likelihood or least square estimate to complete the optimization process.

**Step 4:** Validation of Hypotheses

If the model's assumptions are confirmed, proceed to Step 4, otherwise, return to Step 2 to enhance the model. The model assumptions made in Step 1 are validated using a diagnosis verification. This diagnosis checks whether the leftover hypotheses are correct.

**Step 5:** Prediction

Forecasting is now possible with the model. Predict future values of daily rainfall data using the model from Step 3.

**5.1.1.6 ENSEMBLE METHOD AND BAGGING**

In a given database, the merging method creates multiple guessing models, which are then integrated into the final guessing model. Since then, many other integration methods have been developed, but wrapping, enhancing, and predicting the ARIMA time series is the most widely used method of integration, and many studies have shown that these methods can improve modelling performance.

Packing creates several bootstrap data in the original database, creates a predictive model for each bootstrap data in a consistent manner, and integrates models to create the final model. A database obtained from a randomly modified sample that is the same size as the original database is called "bootstrap data" in this context.

**5.2 ALGORITHM**

**5.2.1 LINEAR REGRESSION**

**STEPS:**

Step 1: Read the dataset

Step 2: Clean up the data in preprocessing stage.

Step 3: Perform the linear regression analysis.

Step 4: Check for homoscedasticity

Step 5: Visualize the results with a graph

Step 6: Report the results.

**5.2.2 ARIMA**

**STEPS:**

Step 1: Read the dataset

Step 2: Fill in any missing values in our time series using the bfill() method.

Step 3: Selection of ARIMA Time Series Model Parameters

Step 4: Define the p, d, and q parameters

Step 5: Modeling an ARIMA Time Series

Step 6: Confirming Forecast accuracy

Step 7: Creating and Visualizing Forecasts

**SYSTEM IMPLEMENTATION**

**CHAPTER 6**

**SYSTEM IMPLEMENTATION**

**6.1 CLIENT-SIDE CODING**

{% extends "base.html" %}

{% block content %}

<div>

<form action = "{{url\_for('prediction')}}" method = "POST">

Select location for prediction:<br>

<select name="city" class="form-control" id="city" required>

{% for row in list\_of\_locations %}

<option value="{{ row['place'] }}">{{ row['place'] }}</option>

{% endfor %}

</select>

<br>

<center><button class="w3-bar-item w3-button w3-teal" type="submit" name ="btn" value="regression">Predict with Regression</button>

<!--<button class="w3-bar-item w3-button w3-teal" type="submit" name ="btn" value="neuralnetwork">Predict with Neural Network</button>-->

<button class="w3-bar-item w3-button w3-teal" type="submit" name ="btn" value="arima" >Predict with ARIMA</button></center>

</form>

<div>

<br>

{% if method!=NULL %}

{% if method=='regression' %}

<center>

<div class="row">

{%for r in results%}

<div class="column">

<div class="card">

{% if r[0]=='temperatureMin' %}

<h3>Minimum Temperature in C</h3>

{%elif r[0]=='temperatureMax'%}

<h3>Maximum Temperature in C</h3>

{%elif r[0]=='precipIntensity'%}

<h3>Precipitation Intensity</h3>

{% endif %}

<strong><p>Value: {{r[1] }}</p>

<p>Explained Variance: {{r[2] }}</p>

<p>The Mean Absolute Error: {{r[3] }}</p>

<p>The Median Absolute Error: {{r[4] }}</p></strong>

</div>

</div>

{% endfor %}

</div>

{% endif %}

{% if method=='arima' %}

<div class="row">

{%for r in results%}

<div class="column">

<div class="card">

{% if r[0]=='temperatureMin' %}

<h3>Minimum Temperature in C</h3>

{%elif r[0]=='temperatureMax'%}

<h3>Maximum Temperature in C</h3>

{%elif r[0]=='precipIntensity'%}

<h3>Precipitation Intensity</h3>

{% endif %}

<strong>

<p>1: {{r[1][0] }}</p>

<p>2: {{r[1][1] }}</p>

<p>3: {{r[1][2] }}</p>

<p>4: {{r[1][3] }}</p>

<p>5: {{r[1][4] }}</p>

<p>6: {{r[1][5] }}</p>

<p>7: {{r[1][6] }}</p>

</strong>

</div>

</div>

{% endfor %}

</div>

<br>

<div class="row">

<div class="column">

<h4>Fig.1 - Minimum Temperature</h4>

<img src={{ url1 }} alt="Chart" height="390" width="550">

</div>

<div class="column">

<h4>Fig.2 - Maximum Temperature</h4>

<img src={{ url2 }} alt="Chart" height="390" width="550">

</div>

<div class="column">

<h4>Fig.3 - Precipitation Intensity</h4>

<img src={{ url3 }} alt="Chart" height="390" width="550">

</div>

</div>

{% endif %}

{% endif %}

</div>

</div>

{% endblock %}

`

**6.2 SERVER-SIDE CODING**

**ARIMA**

import warnings

import itertools

import pandas as pd

import numpy as np

import statsmodels.api as sm

import matplotlib.pyplot as plt

plt.style.use('fivethirtyeight')

import requests

import json

import sys

import os

import pyodbc

import datetime

import pandas as pd

import collections

import time

from datetime import datetime, timedelta

import os

df = pd.read\_csv("weather.csv").set\_index("time")

city = str(input("Enter city: "))

#city = "Karnataka"

df = df.loc[df['city'] == city]

df.index = pd.to\_datetime(df.index)

#df.sort\_values("time")

predictfeature = str(input("Enter feature to predict: "))

#predictfeature = "temperatureMax"

feature = [predictfeature]

data = df[feature]

print(data)

y = data

# The 'MS' string groups the data in buckets by start of the month

if predictfeature == 'precipIntensity':

y = y.precipIntensity.resample('d').mean()

elif predictfeature == 'precipIntensityMax':

y = y.precipIntensityMax.resample('d').mean()

elif predictfeature == 'precipProbability':

y = y.precipProbability.resample('d').mean()

elif predictfeature == 'temperatureMin':

y = y.temperatureMin.resample('d').mean()

elif predictfeature == 'temperatureMax':

y = y.temperatureMax.resample('d').mean()

else:

print("Invalid feature")

exit(1)

# The term bfill means that we use the value before filling in missing values

y = y.fillna(y.bfill())

print(y)

temp = y.head(1)

print(temp)

temp = np.array(temp.index)

print("temp = " , temp)

lastdate = ''

for i in temp:

t = str(i).split("T")

t = t[0]

t = t.split("-")

t= datetime(int(t[0]),int(t[1]),int(t[2]))

lastdate = t

y.plot(figsize=(15, 6))

plt.show()

p = d = q = range(0, 2)

# Generate all different combinations of p, q and q triplets

pdq = list(itertools.product(p, d, q))

# Generate all different combinations of seasonal p, q and q triplets

seasonal\_pdq = [(x[0], x[1], x[2], 12) for x in list(itertools.product(p, d, q))]

#print('Examples of parameter combinations for Seasonal ARIMA...')

#print('SARIMAX: {} x {}'.format(pdq[1], seasonal\_pdq[1]))

#print('SARIMAX: {} x {}'.format(pdq[1], seasonal\_pdq[2]))

#print('SARIMAX: {} x {}'.format(pdq[2], seasonal\_pdq[3]))

#print('SARIMAX: {} x {}'.format(pdq[2], seasonal\_pdq[4]))

warnings.filterwarnings("ignore") # specify to ignore warning messages

for param in pdq:

for param\_seasonal in seasonal\_pdq:

try:

mod = sm.tsa.statespace.SARIMAX(y,

order=param,

seasonal\_order=param\_seasonal,

enforce\_stationarity=False,

enforce\_invertibility=False)

results = mod.fit()

#print('ARIMA{}x{}12 - AIC:{}'.format(param, param\_seasonal, results.aic))

except:

continue

mod = sm.tsa.statespace.SARIMAX(y,

order=(1, 1, 1),

seasonal\_order=(0, 1, 1, 12),

enforce\_stationarity=True,

enforce\_invertibility=False)

results = mod.fit()

pred = results.get\_prediction(start=pd.to\_datetime(lastdate), dynamic=False)

pred\_ci = pred.conf\_int()

print(pred.predicted\_mean)

y\_forecasted = pred.predicted\_mean

y\_truth = y[lastdate:]

# Compute the mean square error

mse = ((y\_forecasted - y\_truth) \*\* 2).mean()

print('The Mean Squared Error of our dynamic forecasts is {}'.format(round(mse, 2)))

pred\_dynamic = results.get\_prediction(start=pd.to\_datetime(lastdate), dynamic=True, full\_results=True)

pred\_dynamic\_ci = pred\_dynamic.conf\_int()

#ax = y['2017':].plot(label='observed', figsize=(20, 15))

#pred\_dynamic.predicted\_mean.plot(label='Dynamic Forecast', ax=ax)

print(pred\_dynamic.predicted\_mean)

# Extract the predicted and true values of our time series

y\_forecasted = pred\_dynamic.predicted\_mean

y\_truth = y[lastdate:]

# Compute the mean square error

mse = ((y\_forecasted - y\_truth) \*\* 2).mean()

print('The Mean Squared Error of our forecasts is {}'.format(round(mse, 2)))

# Get forecast 500 steps ahead in future

pred\_uc = results.get\_forecast(steps=900)

print(pred\_uc.predicted\_mean)

# Get confidence intervals of forecasts

pred\_ci = pred\_uc.conf\_int()

ax = y.plot(label='observed', figsize=(20, 15))

pred\_uc.predicted\_mean.plot(ax=ax, label='Forecast')

ax.fill\_between(pred\_ci.index,

pred\_ci.iloc[:, 0],

pred\_ci.iloc[:, 1], color='k', alpha=.25)

ax.set\_xlabel('Date')

ax.set\_ylabel(predictfeature)

plt.legend()

plt.show()

**LINEAR REGRESSION**

#import matplotlib

#import matplotlib.pyplot as plt

import numpy as np

import pandas as pd

import statsmodels.api as sm

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_absolute\_error, median\_absolute\_error

from sklearn.model\_selection import train\_test\_split

import requests, json

import collections

import time

import datetime

import os

def get\_target\_date():

"""Return target date 1000 days prior to current date."""

current\_date = datetime.now()

target\_date = current\_date - timedelta(days=1000)

return target\_date

def derive\_nth\_day\_feature(df, feature, N):

nth\_prior\_measurements = df[feature].shift(periods=N)

col\_name = f'{feature}\_{N}'

df[col\_name] = nth\_prior\_measurements

features = [

'time', 'precipIntensity', 'precipIntensityMax',

'precipProbability',

'temperatureMin', 'temperatureMax',

'apparentTemperatureMin',

'apparentTemperatureMax',

]

df = pd.read\_csv("weather.csv").set\_index("time")

city = str(input("Enter city: "))

df = df.loc[df['city'] == city]

df.dropna()

print(df.sort\_values("time"))

print(df.columns)

nextday = datetime.datetime.today()

nextday += datetime.timedelta(days=1)

temp = str(nextday).split(" ")[0]

temp = (temp).split("-")

temp = datetime.datetime(int(temp[0]),int(temp[1]),int(temp[2]))

nextday = temp

print(nextday)

record = [[nextday,'','','','','','','']]

newdf = pd.DataFrame(record, columns=features).set\_index('time')

print(newdf)

df.index = pd.to\_datetime(df.index)

newdf.index = pd.to\_datetime(newdf.index)

features = [

'precipIntensity', 'precipIntensityMax',

'precipProbability',

'temperatureMin', 'temperatureMax',

'apparentTemperatureMin',

'apparentTemperatureMax',

]

data = df[features]

data = data.sort\_values(by=['time'])

data = data.resample('d').mean().dropna(how='all')

#print("Edited database with no dublicates \n", data)

data = data.append(newdf)

df = data

# target measurement of mean temperature

predictfeature = str(input("Enter feature to predict: "))

ft = [predictfeature]

#print(tmp[feature][1])

# a list representing Nth prior measurements of feature

# notice that the front of the list needs to be padded with N

# None values to maintain the constistent rows length for each N

for feature in features:

if feature != 'time':

for N in range(1, 4):

derive\_nth\_day\_feature(df, feature, N)

print("Dataframe with nth day features: " , df)

to\_remove = [

feature for feature in features

if feature not in ft

]

#print(to\_remove)

# make a list of columns to keep

to\_keep = [col for col in df.columns if col not in to\_remove]

#print(to\_keep)

# select only the columns in to\_keep and assign to df

df = df[to\_keep]

df = df.apply(pd.to\_numeric, errors='coerce')

#print(df.info())

# Call describe on df and transpose it due to the large number of columns

spread = df.describe().T

# precalculate interquartile range for ease of use in next calculation

IQR = spread['75%'] - spread['25%']

# create an outliers column which is either 3 IQRs below the first quartile or

spread['outliers'] = (spread['min'] <(spread['25%'] -(3 \* IQR))) | (spread['max'] > (spread['75%'] + 3 \* IQR))

#print(spread)

#print(spread.iloc[spread.outliers,])

#print("Current: ",df)

trial = df.loc[nextday]

#print("Testing dataset: " , trial)

df = df.dropna()

#print(df)

df\_corr = df.corr()[[predictfeature]].sort\_values(predictfeature)

#print(df\_corr)

df\_corr\_fil = df\_corr[abs(df\_corr[predictfeature]) > 0.30]

#print(df\_corr\_fil)

unwanted = [predictfeature]

predictors = df\_corr\_fil.index.tolist()

predictors = [i for i in predictors if i not in unwanted]

print("Predictors: ", predictors)

df2 = df[[predictfeature] + predictors]

trial = trial[[predictfeature] + predictors]

X = df2[predictors]

trial = trial[predictors]

y = df2[predictfeature]

alpha = 0.05

# Add a constant to the predictor variable set to represent the Bo intercept

X = sm.add\_constant(X)

#print("Testing dataset: ", trial)

#print("X dataset: ", X)

def stepwise\_selection(X,

y,

initial\_list=predictors,

threshold\_out=alpha,

verbose=True):

included = list(initial\_list)

#print("Initial list : ", initial\_list)

while True:

#print("List:", included)

changed = False

model = sm.OLS(y,X[included]).fit()

# use all coefs except intercept

pvalues = model.pvalues.iloc[1:]

#print("Values: ", pvalues)

worst\_pval = pvalues.max() # null if pvalues is empty

if worst\_pval > threshold\_out:

changed = True

worst\_feature = pvalues.idxmax()

#print("Worst Feature:", worst\_feature)

included.remove(worst\_feature)

#print("List:", included)

if verbose:

print('Drop {:30} with p-value {:.6}'.format(worst\_feature, worst\_pval))

if not changed:

break

return included

result = stepwise\_selection(X, y)

print('Resulting features:')

print(result)

X = X[result]

trial=trial[result]

#print("X: ", X)

#print("Testing: ", trial)

model = sm.OLS(y, X).fit()

print(model.summary())

X\_train, X\_test, y\_train, y\_test = train\_test\_split(

X, y, test\_size=0.2, random\_state=12)

regressor = LinearRegression()

regressor.fit(X\_train, y\_train)

prediction = regressor.predict(X\_test)

#print("X\_test : " , X\_test)

#print("Prediction: ", prediction)

trial = [trial]

print(trial)

predicttest = regressor.predict(trial)

print("Prediction of testing: ", predicttest)

print('The Explained Variance: %.2f' % regressor.score(X\_test, y\_test))

print('The Mean Absolute Error: %.2f degrees celcius' % mean\_absolute\_error(

y\_test, prediction))

print('The Median Absolute Error: %.2f degrees celcius' %

median\_absolute\_error(y\_test, prediction))

**PERFORMANCE ANALYSIS**

**CHAPTER 7**

**PERFORMANCE ANALYSIS**

**7.1 RESULT AND DISCUSSION**

Rainfall forecasting is an attempt to predict future rainfall patterns as well as possible weather conditions. The weather conditions are calculated using parameters like temperature, pressure, humidity, sunrise and sunset hours, latitude, longitude, precipitation, wind speed, and data set length.

The parameters taken into account for experimental purposes include minimum temperature, maximum temperature, precipitation intensity, and precipitation probability. The rainfall dataset for this projection comes from Data.gov.in, a free open web-based data source. Construction of the Arima model for the time series is used to forecast rainfall in India.

Developed the Arima forecasting model in Python using the several packages such as pandas, numpy, and others. Regression Analysis and Auto Regressive Moving Average statistical processes are used to predict and model outcomes.Correlation plots and time-series plots are used to visualise time-series data. In India, statistical methods such as the Arima model and linear regression are used to forecast rainfall time series.

The R-squared value (R2), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) are used to analyze the model's performance (RMSE).

The results clearly show that the ARIMA model accurately forecasts Rainfall with less error, and that the derived model may be used to forecast rainfall for future years.

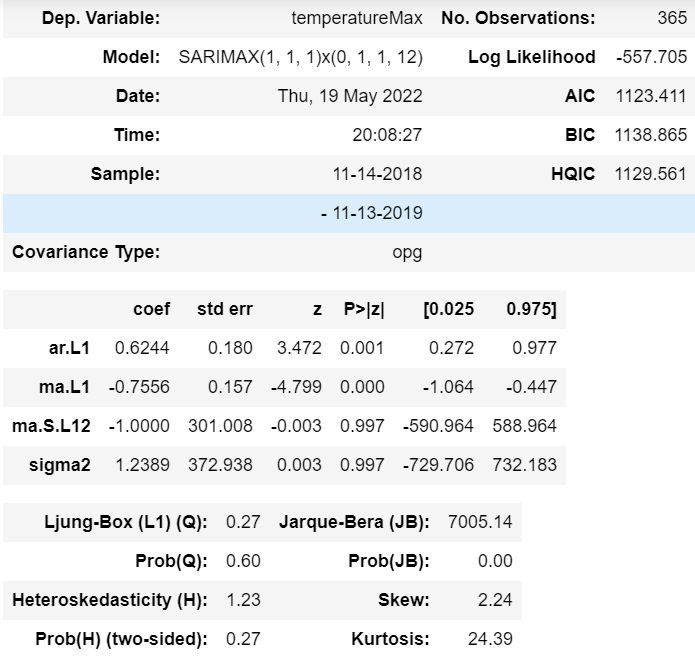


Fig No. 7.1 ARIMA summary

**7.2 PERFORMANCE ANALYSIS**

**7.2.1 LINEAR REGRESSION**

In linear regression, the accuracy of the project is predicted based on the R-squared value. The R-squared value is always between 0 and 1. If the R-squared value is nearly equal to 1 means the regression model fits the observation. R-Squared value nearly 100% means the model is fit and performed well.

R-squared = sum of squares due to regression

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

The total sum of squares

Table. No. 7.1 Regression Performance

|  |  |  |
| --- | --- | --- |
| Features | R-Squared | R-Squared % |
| TemperatureMin | 0.999 | 99.9 |
| TemperatureMax | 0.996 | 99.6 |
| PrecipIntensity | 0.643 | 64.3 |

Fig No. 7.2 R-squared percentage

**7.2.2 ARIMA**

**Mean Squared Error (MSE)**

The average of the squares of the mistakes is the mean squared error. This indicates that the average of the sums of the squares of each difference between the estimated and true values is returned.

MSE formula = (1/n) \* Σ(actual – forecast)2

**Root Mean Squared Error (MSE)**

The root mean squared error, or RMSE, is the most used metric for evaluating linear regression model performance. The primary concept is to compare the model's predictions to actual observed data to see how bad/wrong they are. As a result, a high RMSE is "bad," whereas a low RMSE is "excellent."

RMSE formula = √ (Σ(actual – forecast)2/n)

**Error Percentage**

First, calculate the mean squared error of this model. And the root means squared error for this particular model. Also, the value of the root mean squared should be very smaller than the mean squared error**.** In this case, we can see the average error is going to be roughly 5.26/27.68 \*100=19% of the actual value.

Table. No. 7.2 ARIMA performance

|  |  |  |  |
| --- | --- | --- | --- |
|  | MSE | RMSE | Error % |
| Observed | 27.68 | 5.26 | 19% |
| Forecasted | 1058.85 | 32.54 | 3% |

Here in this project, the forecasted error is very low compared to the observed error so this model is best fit for rainfall prediction.

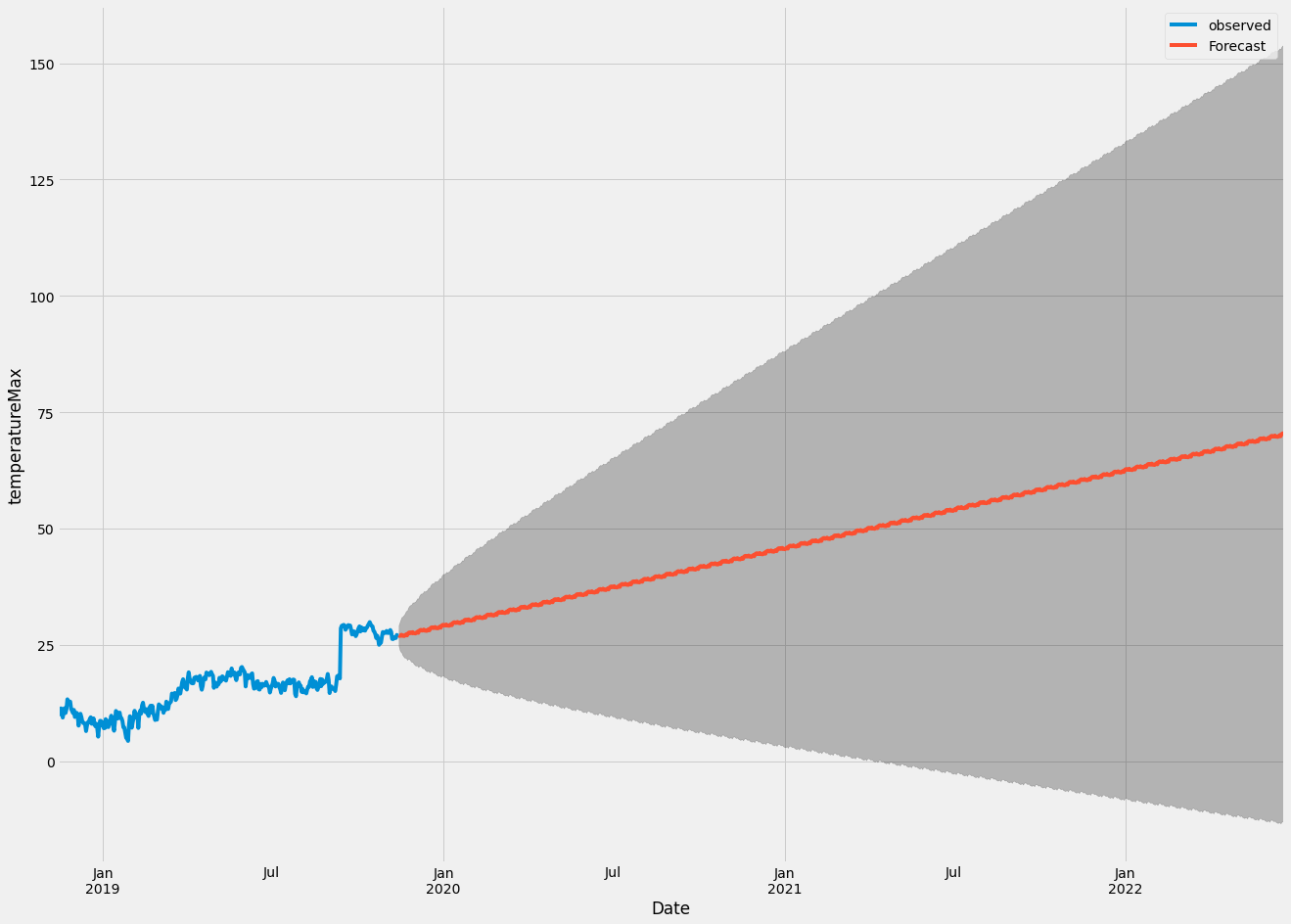
****

Fig No. 7.3 Maximum temperature

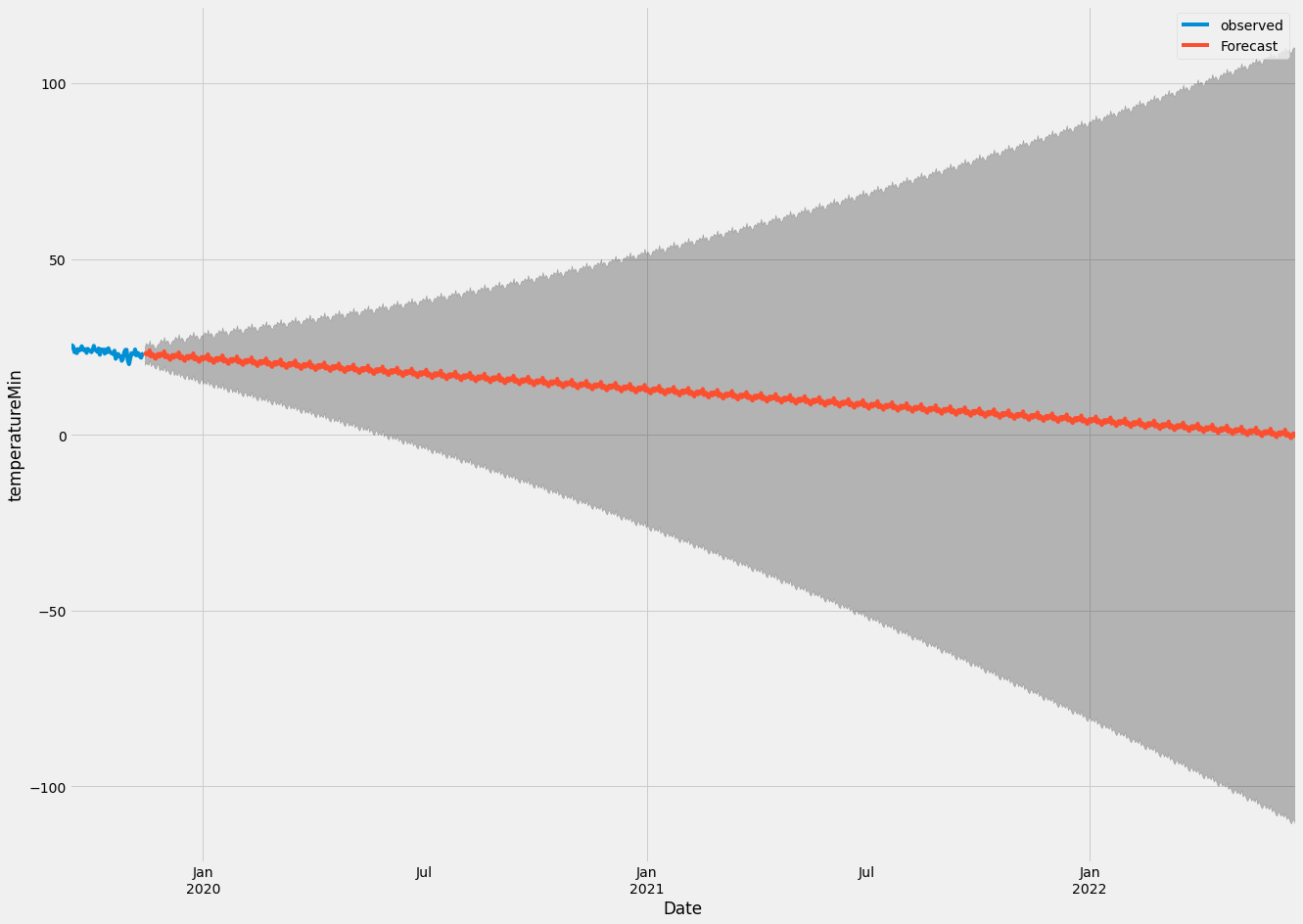
****

Fig No. 7.4 Minimum temperature

**CONCLUSION**

**CHAPTER 8**

**CONCLUSION**

**8.1 CONCLUSION & FUTURE ENHANCEMENT**

The proposed study introduces a rainfall recommendations system that uses ARIMA and Linear Regression to provide computer-assisted results. Depending on the data set used in a particular area, the model focuses on the variety of rainfall and its production in each area, as well as the weather and seasonal patterns. The algorithms used for the recommendation function with ARIMA can be achieved when it rains during the rainy season, so relationships between parameters (such as optimal temperature, rainfall, wind speed, humidity, soil availability, and seed varieties required), rainfall, and region have been established. investigated and demonstrated.

The response is measurable and may be used to calculate the recommended rainfall in additional provinces in the same way as the method. This work can be continuously improved to avoid the problem of inequality in production and demand by adding humidity and airspeed to all regions, which will result in a more accurate recommendation. Rainfall, irrigation, and other features may be added to the system to improve its output. In addition, the recommendation can be modified to warn of timely rainfall data over the course of a particular season and to suggest the types of fertilizers or nutrients needed in the soil for the crop to thrive and produce its superior ARIMA accuracy.

**APPENDICES**

**A.1 SAMPLE SCREENS**

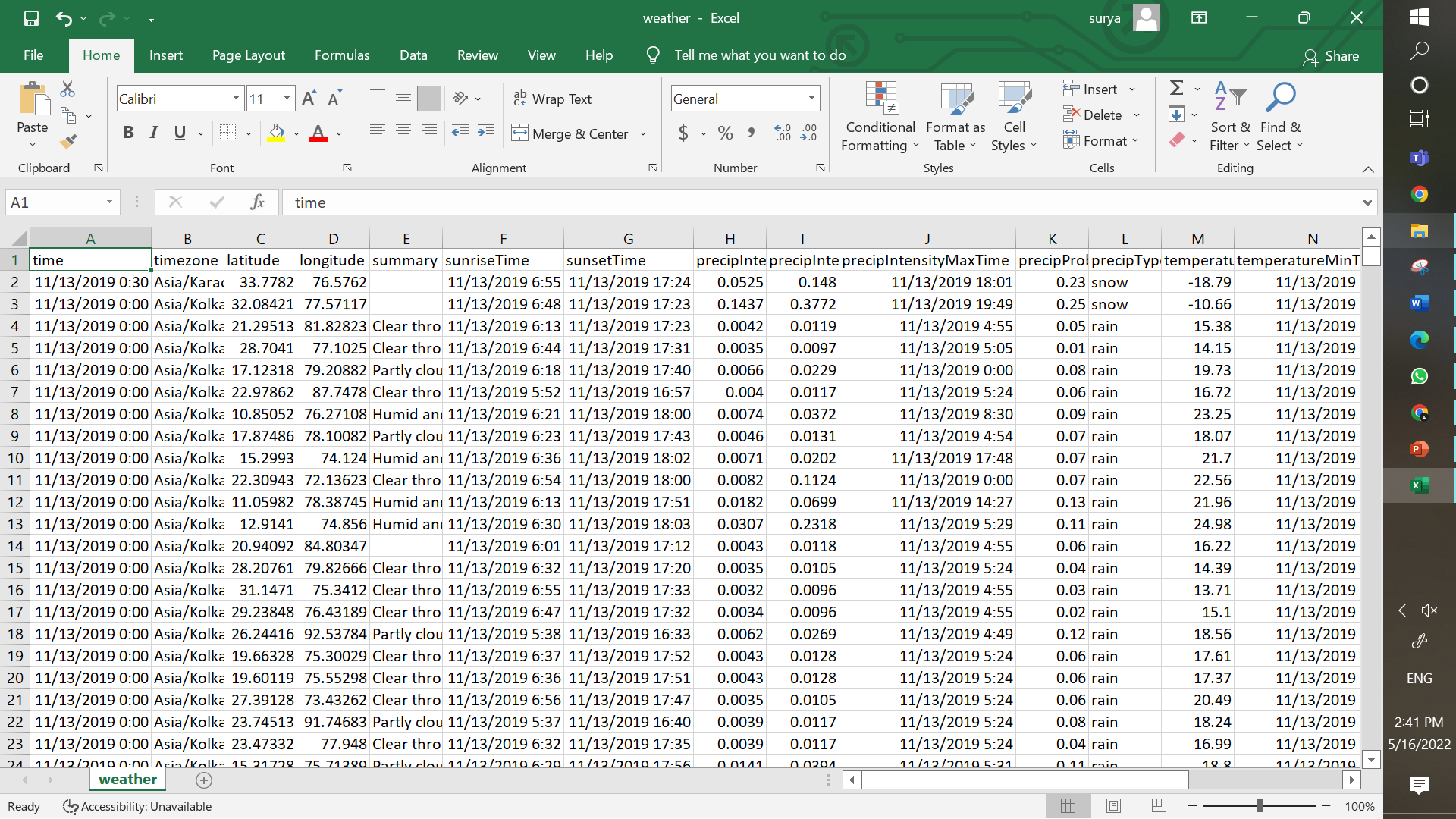


Fig No. A.1 Dataset file

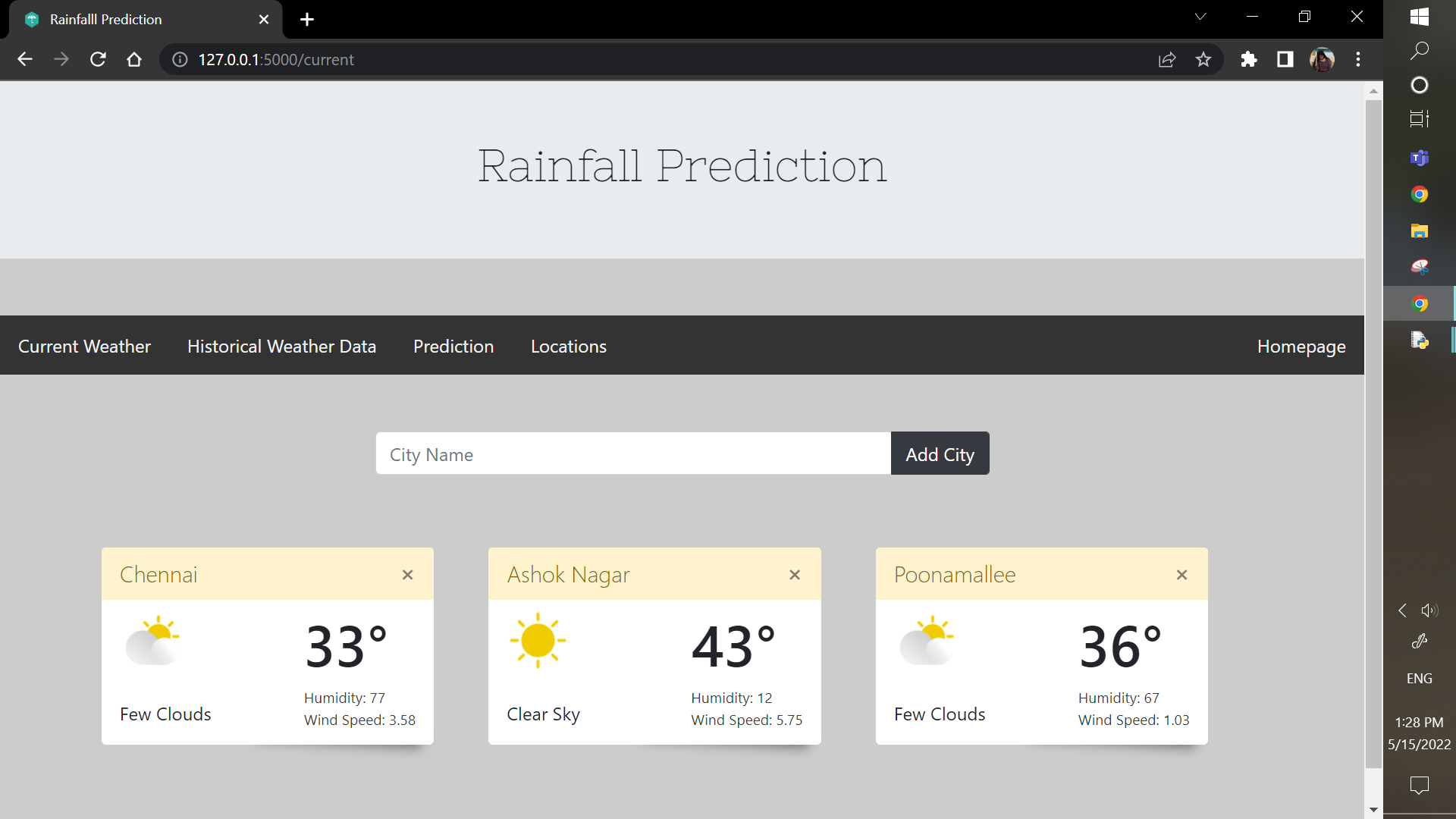
****

Fig No. A.2 Current weather

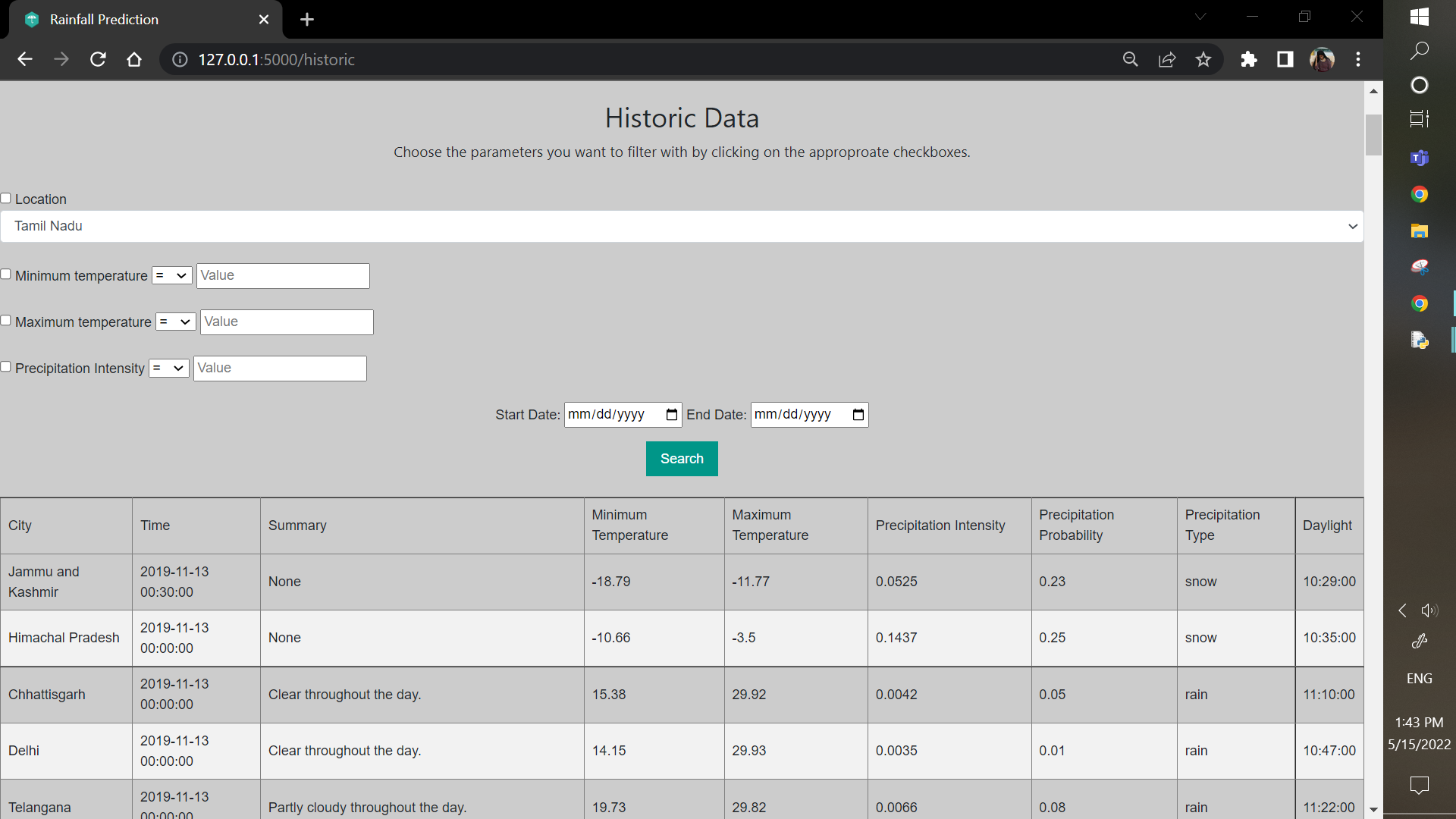


Fig No. A.3 Historic data

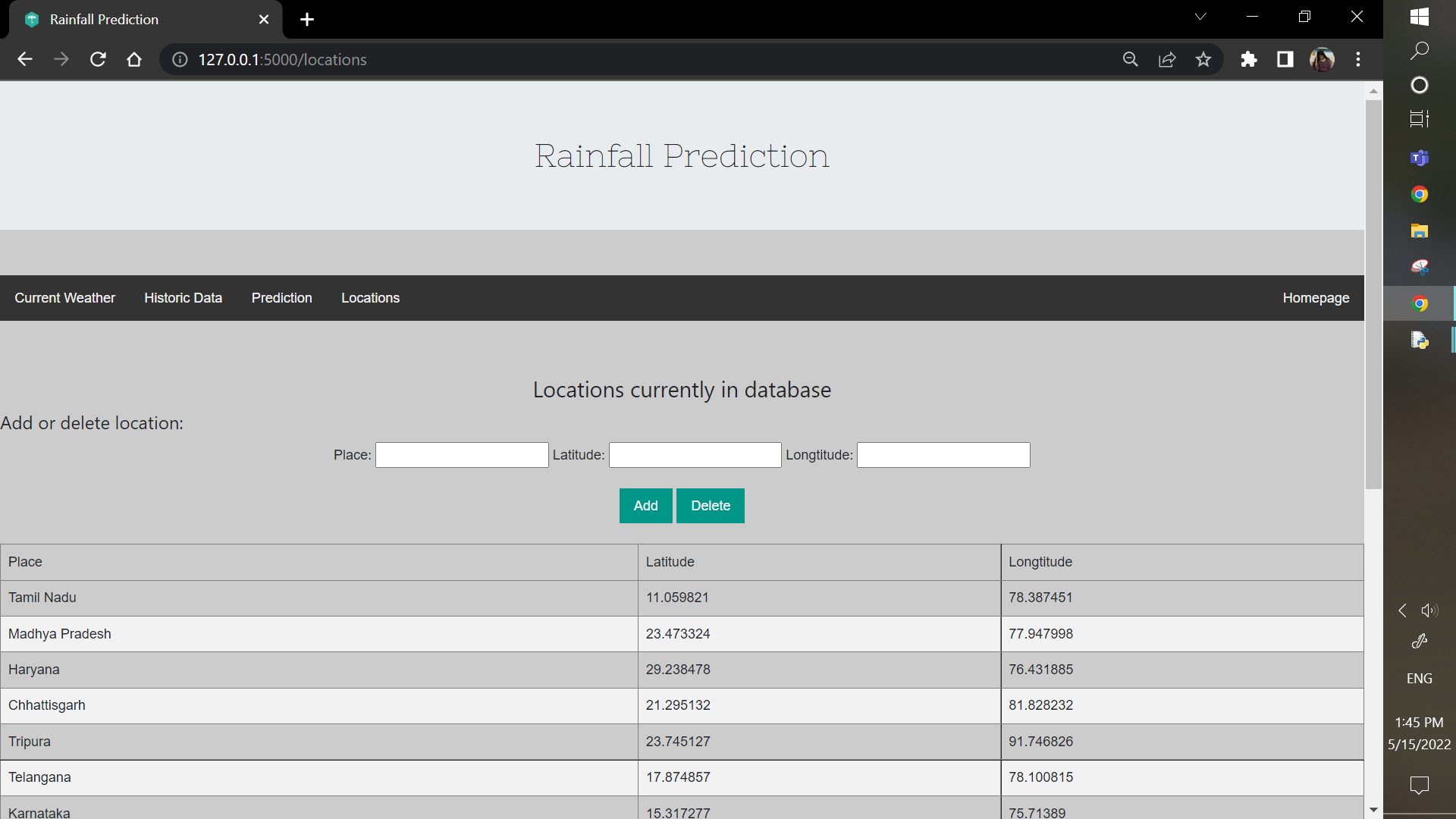


Fig No. A.4 Add location

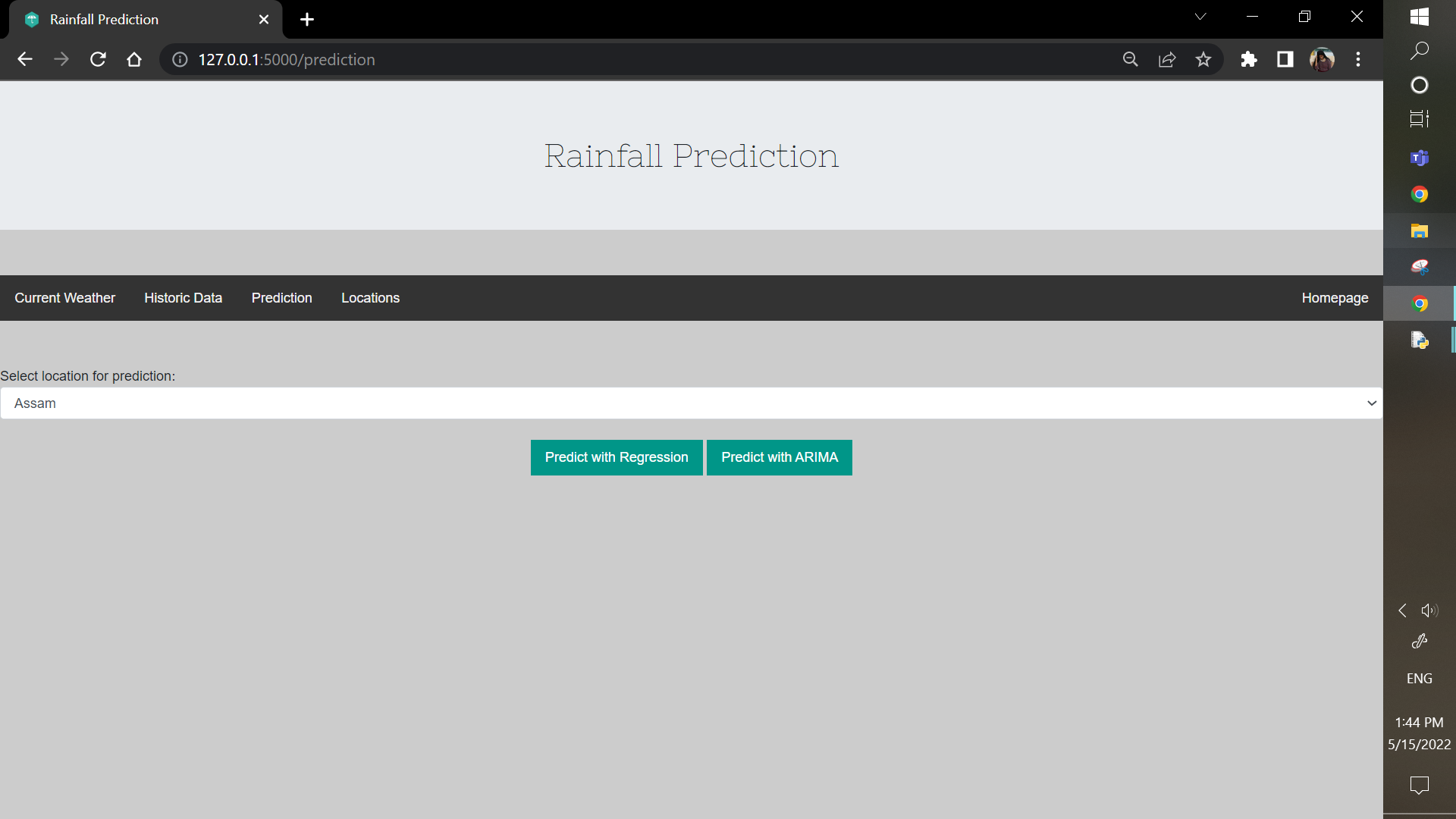


Fig No. A.5 Prediction

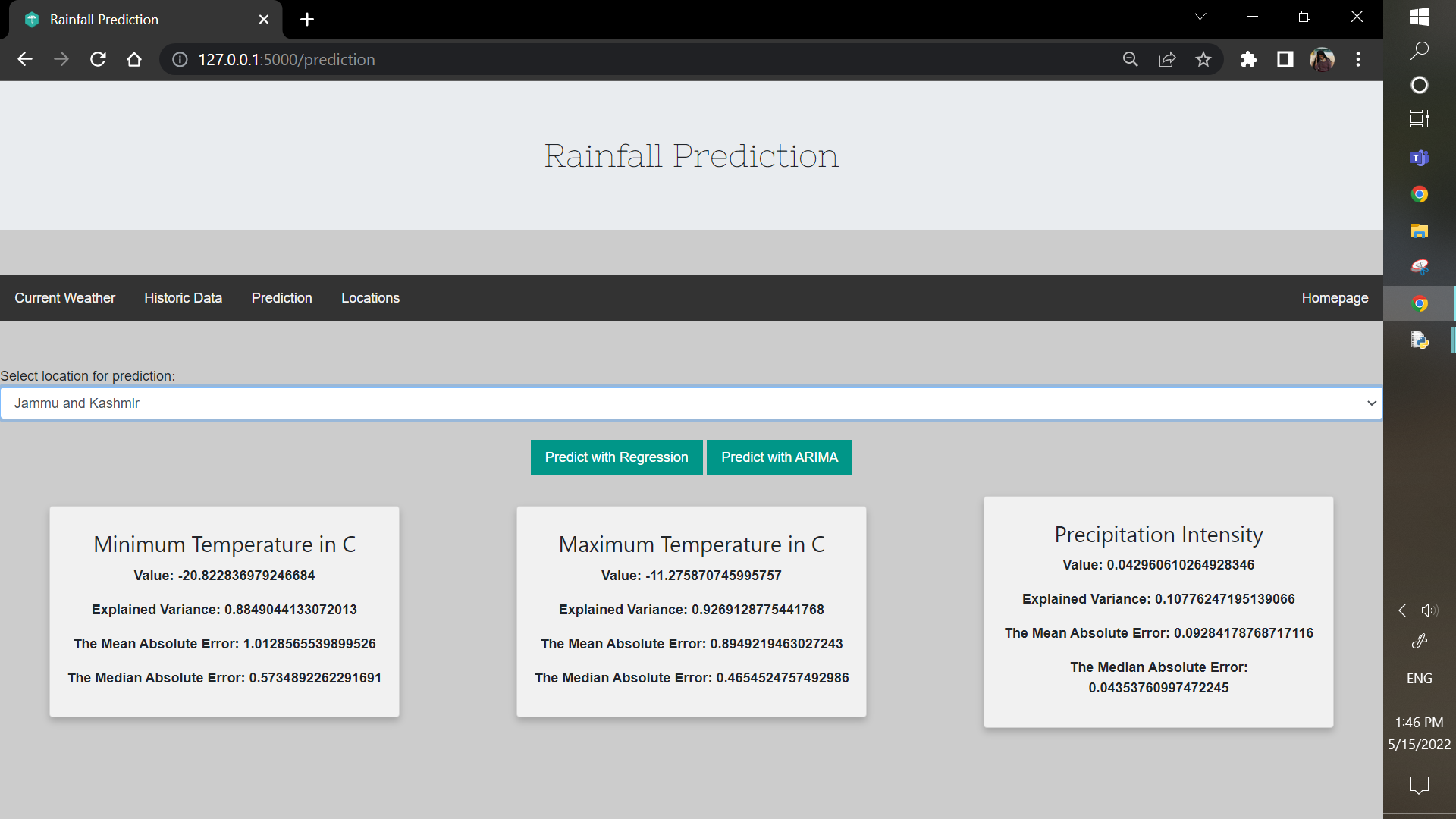


Fig No. A.6 Regression

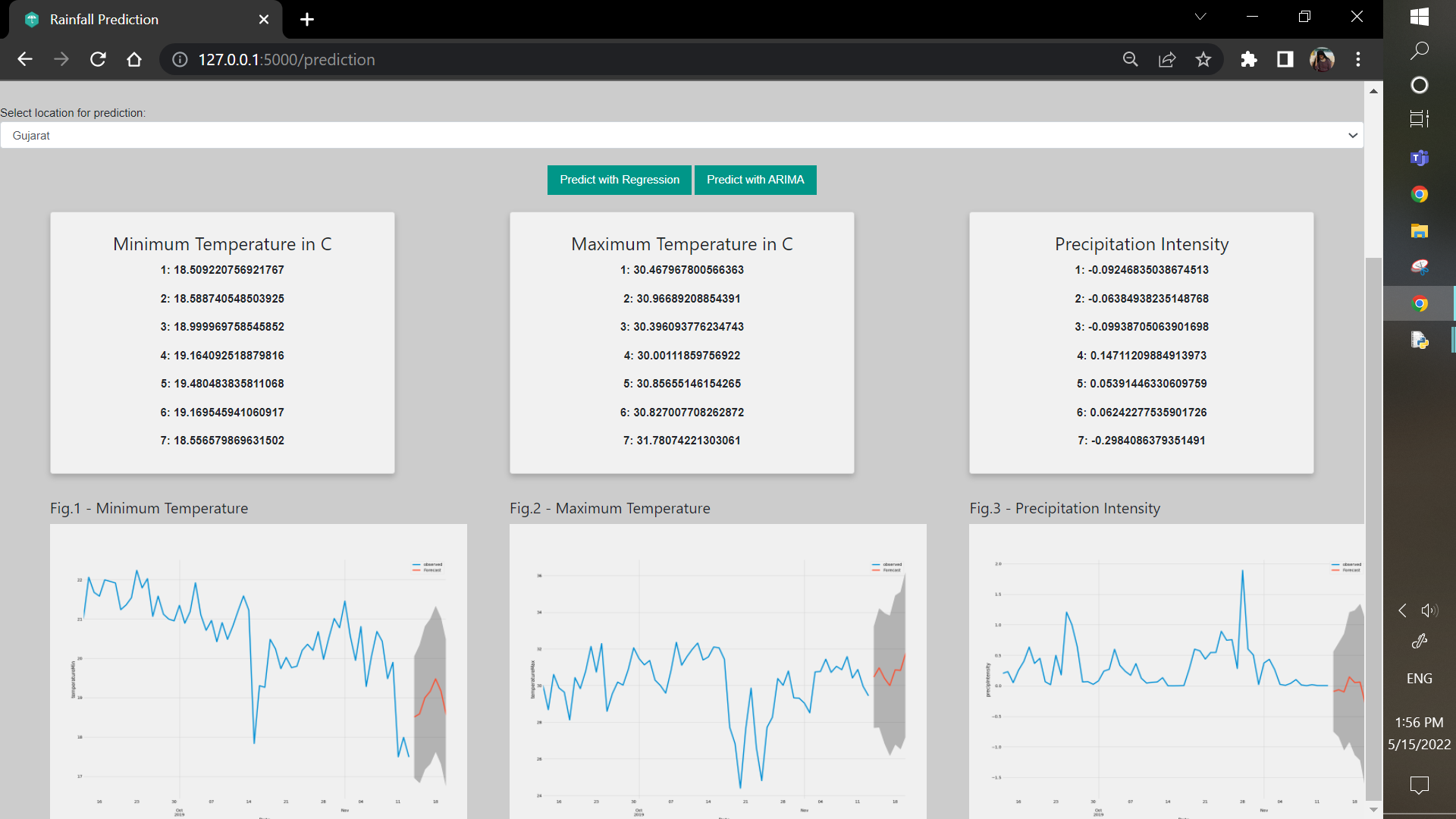


Fig No. A.7 ARIMA

**REFERENCES**

1. U.Ashwini, K.Kalaivani, K.Ulagapriya and A.Saritha (2021),“Time Series Analysis based Tamilnadu Monsoon Rainfall Prediction using Seasonal ARIMA”, IEEE.
2. Anosh Graham, Ekta Pathak Mishra (2017), “Time series analysis model to forecast rainfall for Allahabad region”, Journal of Pharmacognosy and Phytochemistry.
3. Arnav Garg, Himanshu Pandey (2019), “Rainfall Prediction Using Machine Learning”, International Journal of Innovative Science and Research Technology, Volume 4, Issue 5,.
4. CMAK Zeelan Basha, Nagulla Bhavana, Ponduru Bhavya, Sowmya V (2020), “Rainfall Prediction Using Machine Learning & Deep Learning Techniques “, International Conference on Electronics and Sustainable Communication Systems.
5. Dechao Sun, Jiali Wu, Hong Huang, Renfang Wang , Feng Liang, and Hong Xinhua (2021), “Prediction of Short-Time Rainfall Based on Deep Learning”, Volume 2021, 8 pages.
6. Moulana Mohammed, Roshitha Kolapalli, Niharika Golla, Siva Sai Maturi (2020), "Prediction Of Rainfall Using Machine Learning Techniques", international journal of scientific & technology research, volume 9, issue 01.
7. Nana kofi ahoi appiah-badu, yaw marfo missah, leonard k. amekudzi, najim ussiph, twum frimpong, and emmanuel ahene,(2021), “Rainfall Prediction Using Machine Learning Algorithms for the Various Ecological Zones of Ghana”, IEEE, volume 10.
8. Nitin Singh, Saurabh Chaturvedi, Shamim Akhter (2019),“Weather Forecasting Using Machine Learning Algorithm”, IEEE.
9. Pengcheng Zhang, Yangyang Jia, Jerry Gao, Wei Song, Hareton Leung,(2018) “Short-term Rainfall Forecasting Using Multi-layer Perceptron”, IEEE.
10. Uchamad Taufiq Anwar, Saptono Nugrohadi, Vita Tantriyati, Vikky Aprelia Windarni,(2020) “Rain Prediction Using Rule-Based Machine Learning Approach”, Advance Sustainable Science, Engineering and Technology.
11. Wanie M. Ridwan, Michelle Sapitang, Awatif Aziz, Khairul Faizal Kushiar, Ali Najah Ahmed, Ahmed El-Shafie (2020),“Rainfall forecasting model using machine learning methods”, Ain Shams Engineering Journal.