**GROUND WATER LEVEL PREDICTION USING MACHINE LEARNING**

## A PROJECT REPORT

***Submitted by,***

**A.VARAPRASAD-20211CBD0004**

**M.SAI SUJITH REDDY– 20211CBD0006**

**SANGEETHA K– 20211CBD0034**

**VADDE BALAJI– 20211CBD0008**

### *Under the guidance of,*

**Dr .SRINIVASAN T R**

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

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND TECHNOLOGY (BIG DATA)**

**At**



**PRESIDENCY UNIVERSITY**

**BENGALURU**

**MAY 2025**

**PRESIDENCY UNIVERSITY**

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

**CERTIFICATE**

This is to certify that the Project report **“Ground Water Level Predictor Using Machine Learning ”** being submitted by “A,vara prasad , M.Sai sujith Reddy, Sangeetha K,Vadde Balaji” bearing roll number(s) “20211CBD0004 ,20211CBD006 , 20211CBD0034, 20211CBD0008” in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology in Computer Science and Technology (Big Data) is a Bonafide work carried out under my supervision.

|  |  |  |
| --- | --- | --- |
| **Dr. Srunivasan T R,**  Professor,  School of CSE,  Presidency University. | **Dr,Pravinth Raja,**  Professor&HOD,  School of CSE,  Presidency university. |  |
|  |  |  |
| **Dr. L.SHAKKEERA,**  Associate Dean,  School of CSE,  Presidency University. | **Dr. MYDHILI NAIR,**  Associate Dean,  School of CSE,  Presidency University. | **Dr. SAMEERUDDIN KHAN,**  Pro-Vc School of Engineering,  Dean -School of CSE&IS,  Presidency University. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**PRESIDENCY UNIVERSITY**

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

**DECLARATION**

We hereby declare that the work, which is being presented in the project report entitled **Waste Management System** in partial fulfillment for the award of Degree of **Bachelor of Technology** in **Computer Science and Technology (Big Data)**, is a record of our own investigations carried under the guidance of **DR. SRINIVASAN T R, Professor,** **School of Computer Science Engineering & Information Science, Presidency University, Bengaluru.**

We have not submitted the matter presented in this report anywhere for the award of any other Degree.

|  |  |  |
| --- | --- | --- |
| Name(s) | Roll no(s) | Signature(s) |
| A.Varaprasad | 20211CBD0004 |  |
| M.Sai sujith Reddy | 20211CBD0006 |  |
| Sangeetha K | 20211CBD0034 |  |
| Vadde Balaji | 20211CBD0008 |  |

**ABSTRACT**

Groundwater plays an essential role in sustaining life and supporting economic development, especially in regions heavily dependent on agriculture and where surface water is limited. In recent years, the overexploitation of groundwater, combined with changing climate conditions, has led to alarming drops in groundwater levels. Accurate prediction of groundwater levels is critical for water resource planning, disaster preparedness, and sustainable management practices. However, traditional methods for monitoring and forecasting groundwater levels often fall short in terms of accuracy and adaptability to nonlinear patterns in data.

This study explores the application of machine learning (ML) techniques to predict groundwater levels based on historical groundwater measurements and meteorological parameters such as rainfall, temperature, humidity, and evapotranspiration. Machine learning models are capable of identifying hidden trends and relationships in large datasets, offering a data-driven approach to prediction that can outperform conventional statistical methods. By training models on multi-year datasets, we aim to capture both short-term fluctuations and long-term trends that influence groundwater dynamics.

Multiple algorithms, including Linear Regression, Random Forest, Support Vector Regression (SVR), and Long Short-Term Memory (LSTM) neural networks, are implemented and compared. The choice of models spans both traditional regression-based approaches and deep learning techniques, allowing us to evaluate their strengths and weaknesses in handling time-series data. LSTM, in particular, has shown considerable promise due to its ability to remember previous states and manage sequences of data effectively, which is crucial when working with temporal groundwater data.

The models are trained and validated using publicly available datasets from groundwater monitoring agencies, combined with meteorological data from weather databases. The evaluation metrics used—Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R² Score—indicate that while most models provide acceptable predictions, LSTM networks consistently deliver higher accuracy and better generalization capabilities. These findings suggest that advanced deep learning techniques can provide a more reliable solution for groundwater level forecasting.

**ACKNOWLEDGEMENT**

First of all, we indebted to the **GOD ALMIGHTY** for giving me an opportunity to excel in our efforts to complete this project on time.

We express our sincere thanks to our respected dean **Dr. Md. Sameeruddin Khan**, Pro-VC, School of Engineering and Dean, School of Computer Science Engineering & Information Science, Presidency University for getting us permission to undergo the project. We express our heartfelt gratitude to our beloved Associate Deans

**Dr. Shakkeera L and Dr. Mydhili Nair,** School of Computer Science Engineering & Information Science, Presidency University, and Dr. “S.PRAVINTH RAJA”, Head of the Department, School of Computer Science Engineering & Information Science, Presidency University, for rendering timely help in completing this project successfully.

We are greatly indebted to our guide **Dr. SRINIVASAN T R, Professor and Reviewer Mr. RAJAN T, Assistant Professor**, School of Computer Science Engineering & Information Science, Presidency University for his inspirational guidance, and valuable suggestions and for providing us a chance to express our technical capabilities in every respect for the completion of the project work.

We would like to convey our gratitude and heartfelt thanks to the PIP2001 Capstone Project Coordinators **Dr. Sampath A K, Dr. Abdul Khadar A and Mr. Md Zia Ur Rahman,** department Project Coordinators “Ms.**SUMA GOWDA**” and Git hub coordinator **Mr. Muthuraj.**

We thank our family and friends for the strong support and inspiration they have provided us in bringing out this project.

**A. Varaprasad**

**M. Sai Sujith Reddy**

**Vadde Balaji**

**Sangeetha K**

**LIST OF TABLES**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl. No.** | **Table No** | **Table Caption** | **Page No.** |
| 1  2 | Table 1  Table 2 | Literature Survey  Project Timeline | 7  25 |

**LIST OF FIGURES**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl. No.**  1  2  3 | **Figure No**  Figure 1  Figure 2  Figure 3 | **Caption**  Architectural Diagram  ER- Diagram  Gantt Chart | **Page No.**  22  24  25 |
| 4 | Figure 4 | Ground Water Level Visualization | 66 |
| 5 | Figure 5 | Ground Water Level Data Explorer | 66 |
| 6 | Figure 6 | Ground Water Level Forecasting | 67 |
| 7 | Figure 7 | Forecasting Results In ARIMA | 67 |
| 8 | Figure 8 | Ground Water Level Spatial Analysis | 68 |
| 9 | Figure 9 | Ground Water Level Spatial Analysis (Heat Map) | 68 |
| 10 | Figure 10 | Ground Water Level Spatial Analysis (Bubble Map) | 69 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT**  **ACKNOWLEDGMENT**  **LIST OF TABLES**  **LIST OF FIGURES** | **iv**  **v**  **vi**  **vii** |
| **1.** | **INTRODUCTION** | **1** |
| **2.** | **LITERATURE REVIEW** | **3** |
| **3.** | **RESEARCH GAPS OF EXISTING METHODS** | **8** |
| **4.** | **PROPOSED MOTHODOLOGY** | **12** |
| **5.** | **OBJECTIVES** | **17** |
| **6.** | **SYSTEM DESIGN & IMPLEMENTATION** | **20** |
|  | **6.1. System design** | 20 |
|  | **6.2. Implementation** | 23 |
| **7.** | **TIMELINE FOR EXECUTION OF PROJECT** | **25** |
| **8.** | **OUTCOMES** | **26** |
| **9.** | **RESULTS AND DISCUSSIONS** | **28** |
| **10.** | **CONCLUSION** | **30** |
|  | **REFERENCES** | **31** |
|  | **APPENDIX-A(PSUEDOCODE)** | **33** |
|  | **APPENDIX-B(SCREENSHOTS)** | **66** |
|  | **APPENDIX-C(CERTIFICATES)** | **70** |

**CHAPTER-1**

**INTRODUCTION**

Groundwater is one of the most valuable natural resources on Earth, serving as a primary source of freshwater for drinking, agriculture, and industrial purposes. In many regions, particularly those prone to drought or lacking sufficient surface water, groundwater acts as a crucial lifeline. However, growing population demands, unregulated extraction, and the impacts of climate change have led to a significant decline in groundwater levels across the globe. This has raised concerns over water scarcity, agricultural sustainability, and long-term environmental balance.

Groundwater level forecasting has relied on physical models and statistical techniques, which, while useful, often struggle to accurately capture the complex, nonlinear relationships between multiple environmental variables. These conventional models usually require extensive domain knowledge, are sensitive to missing data, and can be computationally intensive. As a result, there is an increasing interest in applying modern data-driven methods, particularly machine learning, to enhance prediction accuracy and offer scalable, adaptable solutions.

Machine learning (ML) has emerged as a promising approach for modeling and predicting groundwater levels due to its ability to handle large datasets, recognize complex patterns, and learn from historical data without explicit programming. Algorithms such as Random Forest, Support Vector Regression, and deep learning models like Long Short-Term Memory (LSTM) networks have demonstrated strong performance in time-series prediction tasks. These models can incorporate diverse inputs, including past groundwater levels, rainfall, temperature, and other hydrological variables, to generate accurate forecasts.

In this project, we aim to develop and compare various machine learning models for predicting groundwater levels. By training these models on real-world datasets, we evaluate their effectiveness and identify the most suitable approach for reliable groundwater forecasting. This research not only contributes to the technical advancement in hydrological modeling but also supports sustainable water management practices by providing timely and accurate groundwater insights.The successful implementation of machine learning-based groundwater prediction models can greatly aid policymakers, farmers, and environmental agencies in making informed decisions. Such models can help anticipate water shortages, plan irrigation schedules, and implement effective conservation strategies, ultimately promoting the responsible and sustainable use of groundwater resources.

One of the major advantages of ML-based approaches is their adaptability to different geographic regions and data availability conditions. Unlike traditional models that may require detailed geological and hydrological inputs, ML models can work effectively with incomplete or noisy data, provided that appropriate preprocessing and feature selection techniques are used. This flexibility makes ML highly suitable for real-world applications, especially in developing regions where data collection infrastructure is limited. Additionally, the integration of machine learning with Geographic Information Systems (GIS) and remote sensing technologies is opening new doors for real-time and spatially-distributed groundwater monitoring.

By incorporating satellite-based rainfall data, land use patterns, and soil moisture indices, ML models can offer a more holistic view of the factors influencing groundwater behavior. This interdisciplinary approach enhances model accuracy and facilitates better decision-making at both local and regional scales.In the context of growing environmental pressures and the urgent need for sustainable water management, this project aims to demonstrate how machine learning techniques can be leveraged for effective groundwater level prediction. By evaluating multiple models and comparing their performance across different conditions, this research contributes valuable insights into the development of intelligent systems for groundwater management, helping safeguard this vital resource for future generations.

**Problem statement:**

Groundwater Level is the cardinal parameter that best describes the health of an Aquifer. Long term groundwater level data acquired through regular monitoring of groundwater levels provide an opportunity to understand the behavior of the aquifers to changing stress regime. Central Ground Water Board periodically monitors ground water levels from nearly 26000 wells. CGWB is also in the process of installing Digital Water Level Recorders (DWLRs) across various parts of India each capturing and transmitting water level data four times per day. In addition to CGWB there are many other agencies measuring water levels. The software application is expected to analyses groundwater levels and factors impacting groundwater level like rainfall, hydrogeology, land use, population, surface elevation, natural features, tidal cycles etc. Based on these factors and based on observed water level data, the software application should be able to predict and forecast groundwater levels both in space and time. The application will be accessible to all for predicting groundwater level at any place at any point on time. This will also help in filling data gaps in time-series data.

**CHAPTER-2**

**LITERATURE SURVEY**

The prediction of groundwater levels has traditionally relied on physical and numerical models, which are often limited by their dependence on extensive field data, assumptions about aquifer properties, and computational complexity. In response to these challenges, machine learning (ML) techniques have emerged as powerful alternatives, capable of identifying complex, nonlinear relationships in data without the need for explicit physical modeling. This section presents a comprehensive review of various machine learning methods that have been applied to groundwater level forecasting, categorized under four major approaches: Artificial Neural Networks (ANN), Support Vector Regression (SVR), Random Forest (RF), and Long Short-Term Memory (LSTM) networks.

**1. Artificial Neural Networks (ANN) in Groundwater Prediction**

Artificial Neural Networks are among the earliest machine learning models used for hydrological forecasting. Inspired by the human brain, ANNs consist of layers of interconnected neurons that process input data through weighted connections. These models are particularly effective in identifying nonlinear patterns, which are prevalent in groundwater systems due to the influence of factors like rainfall, land use, temperature, and human activities.

Kumar et al. (2015) conducted a study in the hard rock terrains of South India where they applied a Multilayer Perceptron (MLP) based ANN to predict groundwater fluctuations. The input data included monthly rainfall and previous groundwater levels. The results showed that the ANN model outperformed traditional linear regression approaches, especially in terms of adaptability to seasonal changes. The study demonstrated that ANNs can model the nonlinear hydrological processes that influence groundwater variability.

Despite their strong predictive performance, ANNs come with certain drawbacks. They require large amounts of high-quality training data to learn effectively and can easily overfit the training dataset if not properly regularized. Additionally, the model’s “black box” nature can make it difficult to interpret the relationship between inputs and outputs, posing challenges for model transparency and validation in scientific and policy applications.

**2. Support Vector Regression (SVR) for Time-Series Forecasting**

Support Vector Regression is a supervised learning model based on the principles of Support Vector Machines (SVM). It is particularly known for its robustness in high-dimensional spaces and its ability to handle nonlinear data through the use of kernel functions. SVR has been increasingly used in hydrological modeling due to its simplicity and effectiveness with relatively small datasets.

Yoon et al. (2016) applied SVR to forecast groundwater levels in agricultural zones in South Korea. Their approach utilized meteorological variables such as precipitation, evapotranspiration, and past groundwater readings. The SVR model provided high accuracy in short-term forecasting and performed well despite missing or sparse data. One of the key strengths of SVR noted in this study was its generalization ability, which allowed it to make accurate predictions even when trained on limited data.

However, SVR also has its limitations. The performance of the model is highly sensitive to the choice of kernel functions (e.g., linear, polynomial, or radial basis function), and tuning these hyperparameters requires domain expertise and experimentation. Furthermore, SVR models may struggle with long-term groundwater forecasting, as they lack internal memory mechanisms that are essential for capturing deep temporal dependencies.

**3. Random Forest (RF) and Ensemble Methods**

Random Forest is an ensemble learning technique based on decision trees. It works by training multiple decision trees on random subsets of the dataset and then aggregating their results to produce a more accurate and robust prediction. This technique is especially suitable for handling large datasets with multiple features and has been widely used in environmental and hydrological modeling.

Zhang et al. (2018) implemented a Random Forest model to predict monthly groundwater levels in Northern China. The model inputs included rainfall, temperature, and groundwater levels from previous months. The RF model was able to deliver high prediction accuracy and identified the most influential variables, thus offering both predictive power and interpretability. One of the notable advantages of RF is its resistance to overfitting due to the ensemble nature of the model.

Despite these strengths, Random Forest models are not inherently designed for sequential or time-series data, which limits their performance in long-term forecasting tasks where temporal dependencies are critical. They also require careful feature engineering to include time-lagged variables, which can increase the complexity of the modeling process.

**4. Long Short-Term Memory (LSTM) Networks in Deep Learning**

Deep learning techniques, especially Recurrent Neural Networks (RNNs) and their advanced variant, Long Short-Term Memory (LSTM) networks, have shown remarkable success in time-series prediction tasks. LSTM networks are specifically designed to learn long-term dependencies by incorporating memory cells that retain information over time, making them highly suitable for modeling groundwater level fluctuations that depend on both recent and historical data.

Shrestha et al. (2020) applied LSTM networks to predict groundwater levels in Nepal’s Terai region using multiyear time-series data, including rainfall, temperature, and historical groundwater measurements. The model outperformed traditional machine learning methods like Random Forest and SVR, particularly in capturing seasonal trends and long-term changes. The LSTM's architecture, which includes forget gates and input modulation, allowed it to selectively retain and discard information, leading to improved forecasting accuracy.

### **5. Hybrid and Hybrid-Optimized Models**

Recent studies have explored hybrid models that combine multiple machine learning techniques or integrate ML with optimization algorithms to improve prediction accuracy and overcome individual model limitations. For instance, Das and Mohanty (2021) proposed a hybrid ANN-GA model where an Artificial Neural Network was optimized using a Genetic Algorithm (GA). The model performed better than standard ANN by improving weight optimization and reducing error rates. Similarly, the integration of Wavelet Transform with ANN or LSTM has helped improve the signal-to-noise ratio in time-series data, leading to more reliable groundwater level predictions.

Hybrid models are particularly useful in handling noisy and non-stationary data, as they allow for preprocessing (e.g., through wavelet decomposition) before applying the predictive model. However, these models tend to be complex in structure, require high computation time, and are difficult to interpret. Despite these drawbacks, hybrid approaches are gaining traction for their accuracy and adaptability in complex hydrological environments.

### **6. Use of Remote Sensing and GIS with ML**

Another emerging trend in groundwater modeling is the integration of **remote sensing data** and **Geographic Information Systems (GIS)** with machine learning. These technologies enable large-scale, spatially distributed modeling, even in data-scarce regions. Studies such as that by Singh et al. (2019) utilized satellite-derived precipitation and land-use data as inputs for a Random Forest model to predict groundwater potential zones in central India.

Combining ML with spatial data allows for better generalization across regions and helps in visualizing risk-prone zones. The use of spatial features (e.g., soil type, elevation, vegetation index) provides additional insights into how surface conditions influence groundwater recharge and depletion. While such models require advanced GIS skills and access to high-resolution satellite data, they are powerful tools for regional planning and decision-making.

### **7. Comparison Studies and Benchmarking**

Several researchers have conducted **benchmarking studies** comparing the performance of multiple machine learning models under similar datasets and conditions. These studies are important because they help establish the strengths and weaknesses of each algorithm in different forecasting contexts.

For example, Rahmati et al. (2022) compared SVR, RF, ANN, and LSTM models using a decade-long groundwater dataset from Iran. Their findings revealed that while LSTM had the best overall accuracy for long-term forecasting, RF offered faster computation and interpretability. Such comparison studies are valuable because they help guide model selection based on use case, available data, and desired outcome (e.g., speed, accuracy, interpretability, or scalability).

Benchmarking also emphasizes the importance of metrics selection (like RMSE, MAE, R²) and data preprocessing techniques, which can significantly influence model outcomes. Therefore, future studies and applications should be transparent in methodology to ensure results can be replicated and extended.

|  |  |  |
| --- | --- | --- |
| **Study / Model** | **Key Results / Strengths** | **Drawbacks / Limitations** |
| Kumar et al. (2015) – ANN | Accurate nonlinear modeling; responsive to rainfall variations | Requires large dataset; risk of overfitting; less interpretable |
| Yoon et al. (2016) – SVR | Good for small datasets; efficient for short-term forecasts | Kernel-sensitive; not ideal for long-term dependency modeling |
| Zhang et al. (2018) – RF | High accuracy; interpretable feature importance | Not designed for time-series; needs feature engineering |
| Shrestha et al. (2020) – LSTM | Best for capturing long-term and seasonal trends; excellent for time-series data | High computational cost; complex tuning; black-box model |
| Das & Mohanty (2021) – Hybrid ANN-GA | Enhanced accuracy through optimization; better weight tuning and model generalization | More complex structure; increased computation time; lower transparency |
| Singh et al. (2019) – RF + Remote Sensing & GIS | Improved spatial prediction; uses satellite & environmental data for regional forecasting | Requires GIS expertise; satellite data access needed; computationally heavy |

**CHAPTER-3**

**RESEARCH GAPS OF EXISTING METHODS**

Despite the promising application of machine learning (ML) techniques for groundwater level prediction, there are still significant gaps in existing research that need to be addressed. These gaps not only hinder the progress of groundwater prediction models but also present opportunities for further advancement in hydrological modeling. The challenges related to data, modeling techniques, generalization, and interpretability are critical to improving both the accuracy and the practical application of machine learning models in real-world water resource management.

**1. Limited Generalization Across Regions**

One of the major challenges faced by machine learning models in groundwater level prediction is their limited generalization ability. Most current studies use region-specific datasets, and the models they develop are tailored to the unique characteristics of the study area. As a result, these models often fail to perform as expected when applied to different geographic regions with varying climatic, hydrological, and environmental conditions. For example, a model trained using data from a semi-arid region may not work well in a tropical or temperate climate due to the differences in rainfall patterns, soil permeability, and groundwater recharge rates.

To address this gap, research is needed to develop scalable and adaptable models that can generalize well across different regions. These models should be capable of learning regional differences without requiring substantial retraining. Furthermore, incorporating global climate models or regional hydrological models into the machine learning approach may enhance the model’s ability to adapt to new environments.

**2. Insufficient Long-Term and Real-Time Forecasting**

Although many machine learning models perform well in short-term groundwater predictions, they are often unable to predict long-term trends with the same level of accuracy. Long-term forecasting is crucial for groundwater management, as it helps predict changes in water availability over several months or even years, which is vital for agricultural, industrial, and urban water planning. Most studies focus on shorter prediction windows, usually ranging from a few days to a few months, often limiting their applicability to scenarios where long-term management decisions are required.

Additionally, many existing models do not integrate real-time data streams such as data from groundwater monitoring wells, IoT sensors, or satellite-based measurements. Real-time forecasting would allow for dynamic updates to the predictions based on the most current data, offering more precise and timely decision support for water resource managers.

To address this issue, future research should focus on incorporating real-time, continuous data streams into machine learning models, enabling models to update and adapt their predictions dynamically. Moreover, extending machine learning models to handle longer time horizons and account for long-term variability in groundwater recharge and depletion patterns is another area that requires attention.

**3. Lack of Standardized Benchmark Datasets**

A significant challenge in the field of machine learning for groundwater prediction is the lack of standardized datasets. Many studies use proprietary, localized, or region-specific datasets, making it difficult to compare and validate the results across different research efforts. As a result, the performance of different models cannot be fairly assessed, and reproducibility is often compromised. This issue is compounded by the fact that various models use different evaluation metrics (e.g., RMSE, MAE, R²), which can make cross-study comparisons challenging.

The development of publicly available benchmark datasets for groundwater level prediction would provide a common ground for researchers to evaluate and compare the performance of different machine learning models. In addition to having publicly available datasets, these datasets should be well-documented with clear descriptions of data collection methods, the underlying assumptions, and any preprocessing steps used. Creating a common evaluation framework with standardized metrics will also allow for more objective comparisons and contribute to the advancement of the field.

**4. Limited Integration with Physical and Hydrogeological Knowledge**

Machine learning models for groundwater prediction often treat the groundwater system as a purely data-driven black box, where the underlying physical processes are not explicitly modeled. While data-driven models can provide accurate predictions, they do not always reflect the real-world dynamics of groundwater systems. This lack of integration with physical knowledge can limit the interpretability and trustworthiness of predictions, especially in policy-making and regulatory contexts where decisions need to be grounded in scientific principles.

There is a growing need for hybrid models that combine machine learning techniques with physical and hydrogeological models. For example, integrating machine learning algorithms with numerical groundwater flow models (e.g., MODFLOW) can improve the model’s predictive accuracy while retaining the physical integrity of the hydrological processes. Such hybrid models can help address the gaps between data-driven predictions and physical understanding, making them more interpretable and useful for decision support in water resource management.

Additionally, incorporating hydrogeological features like aquifer properties, permeability, and recharge rates into machine learning models could help enhance the accuracy and robustness of the predictions. Models that incorporate both data and physical principles will be better suited for addressing large-scale groundwater management challenges.

**5. Underexplored Use of Spatio-Temporal Modeling**

Most groundwater models focus primarily on temporal features, such as past groundwater levels, precipitation, and temperature, without fully accounting for spatial dependencies that may influence groundwater dynamics. Groundwater systems are inherently spatially heterogeneous, meaning that the groundwater levels in one location are influenced not only by time-related factors but also by the conditions in surrounding areas. Factors such as land use, soil type, topography, and the presence of nearby wells can significantly affect groundwater levels.

Unfortunately, many existing machine learning models ignore spatial correlations, leading to suboptimal predictions in areas where spatial dependencies are strong. There is a need for spatio-temporal models that can simultaneously account for both temporal and spatial factors. Approaches like Graph Neural Networks (GNNs) and Convolutional LSTM (ConvLSTM) have the potential to improve predictions by considering the spatial structure of the groundwater system. These models could integrate information from multiple wells or monitoring stations, providing more accurate and region-specific predictions.

**6. Model Interpretability and Decision Support**

Machine learning models, especially deep learning models like LSTM, are often criticized for being black-box models—they produce accurate predictions but do not offer insights into how those predictions are made. For decision-makers, such as water resource managers or policymakers, understanding the reasoning behind model predictions is crucial for building trust and confidence in the system. Model transparency and interpretability are essential for ensuring that the model’s outputs can be used for practical decision support.

While several methods exist to improve model interpretability (e.g., SHAP values, LIME), they have not been extensively applied to groundwater prediction models. The lack of transparency in these models makes it difficult for stakeholders to understand the relationships between model inputs and predictions, which is crucial in sensitive areas like water resource management. Future research should focus on developing explainable AI (XAI) techniques that can provide clear, understandable explanations for the model's decision-making process.

**7. Data Quality and Availability**

Another challenge in machine learning-based groundwater prediction is the quality and availability of data. Groundwater level data is often sparse, incomplete, or missing in many regions, especially in developing countries. Furthermore, data may come from different sources, with varying accuracy and granularity. In some cases, data might be collected at irregular intervals or with different measurement methods, making it difficult to integrate and model.

To overcome these challenges, future research should explore data augmentation techniques and the use of synthetic data to supplement real-world data. Techniques like transfer learning could also be employed to train models on data from one region and apply them to another with less data. Additionally, collaboration between researchers, local governments, and private organizations can help improve data accessibility and quality, especially in regions where groundwater monitoring networks are underdeveloped.

**CHAPTER-4**

**PROPOSED MOTHODOLOGY**

Groundwater level prediction requires a sophisticated understanding of hydrological systems, as well as the ability to process complex, multidimensional datasets. To improve the prediction accuracy and overcome the existing limitations in current methodologies, this section proposes several novel approaches and methodologies that can be used in groundwater level forecasting using machine learning.

**1. Hybrid Machine Learning Models**

One of the most effective ways to enhance prediction accuracy and address the limitations of individual models is by developing hybrid machine learning models. These models combine the strengths of multiple algorithms to optimize prediction performance.

**Proposed Approach:**

**Hybrid ANN and Genetic Algorithm (GA):** The use of Genetic Algorithms for optimization of artificial neural networks (ANNs) is one promising hybrid approach. GA can be employed to optimize the weights and biases of an ANN model, helping to fine-tune the model and reduce overfitting. This approach can be particularly effective for groundwater systems that exhibit nonlinear behaviors.

**Hybrid SVR and Wavelet Transform:** Another potential hybrid model involves combining Support Vector Regression (SVR) with Wavelet Transform. The wavelet transform helps to decompose the time series data into frequency components, making it easier for the SVR model to capture both short-term and long-term trends in groundwater levels. This method can address the issue of non-stationarity in groundwater time series data.

**CNN-LSTM Hybrid**: A Convolutional Neural Network (CNN) could be used to capture spatial features in groundwater level data (such as spatial dependencies between monitoring stations), while an LSTM network could handle temporal dependencies. The integration of both spatial and temporal features into a hybrid model could result in more accurate predictions.

**Benefits:**

Improved model performance by leveraging multiple algorithms.

Better handling of complex, non-linear, and multi-dimensional data.

Reduced risk of overfitting.

**Challenges:**

Increased computational cost due to the complexity of hybrid models.

Difficulty in model interpretation and explanation.

**2. Incorporating Real-Time Data Streams**

To enhance the adaptability and accuracy of groundwater level predictions, it is crucial to integrate real-time data from sensors, remote sensing platforms, or online hydrological monitoring stations into the predictive models.

**Proposed Approach:**

**IoT-Based Monitoring Systems:** Internet of Things (IoT) technology can be used to continuously monitor groundwater levels using remote sensors placed in wells. These sensors can provide real-time data on groundwater levels, rainfall, temperature, and other relevant environmental factors.

Data Fusion: Real-time data can be integrated with historical data using data fusion techniques, combining the information from sensors with satellite data (such as precipitation and soil moisture), to improve the model's performance. This will ensure that the model updates dynamically and produces more accurate predictions.

**Online Learning Models:** Implement online learning algorithms such as Online Gradient Descent (OGD) or Incremental SVM to adapt to the new data as it becomes available. These models update their parameters in real-time, making them ideal for dynamic systems like groundwater level forecasting.

**Benefits:**

Timely updates to predictions based on the most current data.

More accurate, real-time decision support for water resource management.

**Challenges:**

The need for continuous data monitoring and management.

Integration of heterogeneous data from multiple sources can be challenging.

**3. Spatio-Temporal Modeling with Graph Neural Networks (GNN)**

Groundwater level prediction can benefit significantly from models that account for both spatial and temporal dependencies. Traditional models often overlook spatial relationships between different monitoring stations, which may lead to suboptimal predictions. Spatio-temporal models that incorporate both temporal and spatial correlations will be more accurate and informative.

**Proposed Approach:**

**Graph Neural Networks (GNN)**: GNNs are ideal for capturing spatial dependencies in groundwater level data. These networks use a graph structure, where each node represents a monitoring station or well, and the edges represent the relationships between them (e.g., geographical proximity or shared aquifer system). By learning from the graph structure, GNNs can capture the spatial patterns in groundwater levels across multiple locations.

**Convolutional LSTM (ConvLSTM):** ConvLSTM can be used for spatio-temporal modeling where both spatial and temporal features are important. The CNN component of ConvLSTM captures spatial patterns (e.g., geographical distribution of groundwater levels), while the LSTM handles temporal dependencies. This combination is ideal for modeling dynamic systems like groundwater, where both time and space play critical roles in influencing the behavior of the system.

**Benefits:**

Improved accuracy by considering both spatial and temporal dependencies.

Enhanced understanding of how groundwater levels in one area may affect neighboring regions.

**Challenges:**

Complex implementation of GNNs and ConvLSTM models.

High computational requirements for spatio-temporal modeling.

**4. Incorporating Domain Knowledge and Hybrid Physics-AI Models**

Many machine learning models treat the groundwater system as a purely data-driven entity, which can limit their interpretability and predictive accuracy. Integrating domain knowledge from hydrogeology with machine learning models could provide better results.

**Proposed Approach:**

**Physics-Informed Neural Networks (PINNs):** PINNs combine traditional physical models (e.g., groundwater flow equations) with machine learning. By embedding physical laws (such as Darcy’s law or the continuity equation) into the learning process, PINNs ensure that predictions are consistent with known hydrological principles, improving their reliability and interpretability.

**Hybrid Model with MODFLOW:** Combine MODFLOW (a widely used groundwater flow model) with machine learning techniques like ANN or SVR. MODFLOW provides a solid physical framework for simulating groundwater flow, while machine learning algorithms can optimize model parameters or predict groundwater levels based on data-driven insights.

**Benefits:**

Models that are physically grounded and more interpretable.

Increased accuracy and reliability by using domain-specific knowledge.

**Challenges:**

Requires knowledge of both hydrology and machine learning.

Difficulty in balancing the trade-off between physical constraints and data-driven learning.

**5. Explainable AI (XAI) for Model Transparency**

Groundwater management decisions often rely on the outputs of machine learning models, which are sometimes difficult to interpret. Explainability of model predictions is crucial, especially when the results affect policy-making and resource allocation.

**Proposed Approach:**

**SHAP and LIME for Model Interpretability**: SHAP (SHapley Additive exPlanations) and LIME (Local Interpretable Model-agnostic Explanations) are popular methods for improving the transparency of machine learning models. These techniques can be applied to machine learning models like LSTM or Random Forest to identify which features most strongly influence the predictions. This will help decision-makers understand why the model is making certain predictions, increasing their trust in the system.

**Attention Mechanisms:** Attention mechanisms can be integrated into models like LSTM to identify the most important time periods or spatial regions that influence groundwater predictions. By highlighting the key inputs that affect the output, attention mechanisms provide better transparency into model decision-making.

**Benefits:**

Increased trust and adoption of machine learning models by stakeholders.

More informed decision-making with clearer insights into model behavior.

**Challenges:**

Implementing explainability techniques can add complexity to the model.

Some models (like deep neural networks) are inherently difficult to interpret even with XAI techniques.

**CHAPTER-5**

**OBJECTIVES**

The main objective of this study is to enhance the accuracy and reliability of groundwater level predictions using machine learning techniques. The proposed study seeks to address existing challenges in groundwater prediction, such as limited model generalization, lack of real-time data integration, and insufficient model interpretability. To achieve this, the following specific objectives have been outlined:

### **1. To Develop and Implement Hybrid Machine Learning Models**

* The study aims to explore the integration of multiple machine learning models to improve prediction accuracy and address the limitations of individual algorithms.
* Hybrid models such as **Artificial Neural Networks (ANN) with Genetic Algorithms (GA)**, **Support Vector Regression (SVR) with Wavelet Transform**, and **Convolutional Neural Networks (CNN) with Long Short-Term Memory (LSTM)** will be developed to capture both spatial and temporal features in the groundwater level data.

### **2. To Integrate Real-Time Data Streams for Dynamic Predictions**

* This objective focuses on integrating **real-time groundwater data** from monitoring wells, sensors, and IoT-based systems into the machine learning models.
* The study will also aim to develop **online learning models** that can adapt to incoming data and make continuous updates to predictions, ensuring that the system remains responsive to the latest environmental conditions.
* A critical goal is to design a model that can update its predictions dynamically based on real-time data from various sources, thereby improving prediction accuracy.

### **3. To Incorporate Spatio-Temporal Relationships in Groundwater Level Prediction**

* An important objective is to develop models that can simultaneously account for both **temporal** (time-related) and **spatial** (location-related) dependencies in groundwater data.

**4. To Integrate Domain Knowledge for Improved Model Accuracy and Interpretability**

* This study aims to integrate **hydrogeological domain knowledge** with machine learning models to create **hybrid physics-AI models** that ensure predictions are grounded in known hydrological principles, such as groundwater flow equations.
* The study will explore **Physics-Informed Neural Networks (PINNs)** and **hybrid models combining MODFLOW and machine learning** to enhance the accuracy of predictions while maintaining the physical realism of the system.
* A key objective is to ensure that the model predictions are not only accurate but also interpretable, providing stakeholders with a better understanding of how groundwater levels are predicted.

### **5. To Enhance Model Interpretability Using Explainable AI (XAI) Techniques**

* The study aims to incorporate **explainable AI (XAI)** methods, such as **SHAP** (SHapley Additive exPlanations) and **LIME** (Local Interpretable Model-agnostic Explanations), into the machine learning models to increase transparency in model predictions.
* The goal is to ensure that decision-makers and stakeholders can understand the reasoning behind the predictions, which is particularly important when making critical decisions related to water resource management and policy.
* The study will evaluate how well XAI methods can help explain the influence of different variables on the model’s output, thereby enhancing trust in the model’s predictions.

### **6. To Improve the Generalization Ability of Groundwater Level Prediction Models**

* A key objective of this research is to develop models that can generalize well across different geographical regions with varying hydrogeological conditions.
* The study will investigate the use of **transfer learning** and **data augmentation techniques** to enhance model generalization, particularly in regions with limited data availability.
* This objective aims to create models that can be easily applied to other regions with minimal retraining, making them more adaptable and practical for large-scale groundwater management.

### **7. To Develop a Comprehensive Evaluation Framework for Model Performance**

* The study will aim to establish a **standardized evaluation framework** to assess the performance of different machine learning models for groundwater level prediction.
* Various **evaluation metrics**, such as **Root Mean Square Error (RMSE)**, **Mean Absolute Error (MAE)**, **R²**, and **Nash-Sutcliffe Efficiency (NSE)**, will be used to compare the accuracy and reliability of the developed models.
* The objective is to ensure that the models are evaluated on consistent and comprehensive criteria, making it easier to compare results across different studies and benchmark their performance.

**CHAPTER-6**

**SYSTEM DESIGN & IMPLEMENTATION**

The system design and implementation of a groundwater level prediction model using machine learning involves creating an end-to-end framework that processes data, applies predictive models, and delivers actionable insights for groundwater management. The design will include multiple stages: data collection, preprocessing, model development, evaluation, and deployment. Below, we outline the core components and structure of the system, including the workflow and implementation details.

**1. System Architecture**

The system can be represented in the following modular components:

**Data Collection and Integration**

The first step is to gather relevant data, which can be sourced from multiple systems and sensors. This data is fed into the model for analysis and prediction.

Sensor Data: Includes data from monitoring wells, IoT sensors, and other environmental monitoring systems. This can include groundwater levels, soil moisture, temperature, rainfall, etc.

Remote Sensing: Satellite-based or drone-based data that can provide additional inputs, like land use or precipitation.

Historical Data: Past records of groundwater levels, meteorological data, and environmental parameters from publicly available datasets.

**Data Preprocessing**

Data preprocessing is crucial for improving the quality of input data and ensuring it is in a format suitable for machine learning algorithms. The following preprocessing steps will be performed:

Data Cleaning: Handling missing values and filtering outliers.

Normalization/Standardization: Scaling data to improve model convergence.

Feature Engineering: Creating new features, such as rolling averages of past groundwater levels, to capture trends and seasonality.

Time Series Preparation: Structuring the data in a format suitable for time-series forecasting, ensuring that temporal relationships are preserved.

**Machine Learning Model Development**

The core of the system is the application of machine learning algorithms to predict groundwater levels. Several models will be considered and tested to find the most accurate and robust one for the given dataset.

Hybrid Machine Learning Models

**ANN-GA Hybrid:** Artificial Neural Networks (ANN) will be combined with Genetic Algorithms (GA) for parameter optimization. The GA will tune the ANN model's weights and biases for improved accuracy.

**SVR-Wavelet Transform:** A hybrid model combining Support Vector Regression (SVR) with Wavelet Transform for capturing both high-frequency and low-frequency components of groundwater level fluctuations.

**CNN-LSTM Hybrid:** A combination of Convolutional Neural Networks (CNN) for spatial feature extraction and Long Short-Term Memory (LSTM) networks for temporal sequence modeling. This hybrid model can capture both the spatial relationships between wells and the time-dependent behavior of groundwater levels.

**Spatio-Temporal Models**

**Graph Neural Networks (GNNs):** GNNs will be used to model the spatial dependencies between different monitoring stations or regions. This network learns from the relationships between wells and aquifers, capturing spatial dependencies of groundwater levels.

**ConvLSTM:** A spatio-temporal model that incorporates both CNN and LSTM to capture spatial and temporal patterns in the data. ConvLSTM can learn both spatial and sequential data, making it ideal for predicting groundwater levels across multiple regions over time.

**Model Training**

**Training Data**: The model will be trained on historical data to learn the underlying patterns that influence groundwater levels. The dataset will be divided into training and validation sets, typically using a 70-30 or 80-20 split.

**Cross-Validation:** To ensure generalization, k-fold cross-validation will be used to assess model performance and avoid overfitting.

**Optimization:** The models will be optimized using techniques such as grid search or random search to fine-tune hyperparameters and achieve the best possible performance.

**Model Evaluation**

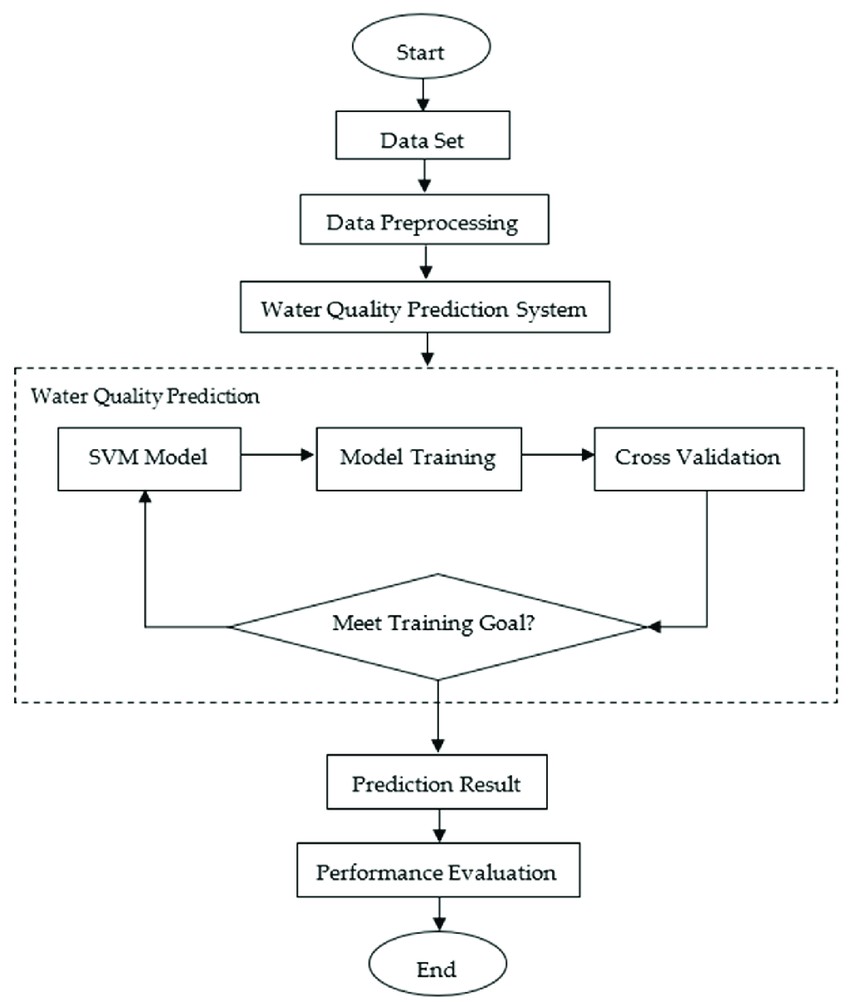
The models will be evaluated using various metrics:

**Root Mean Square Error (RMSE):** Measures the difference between predicted and observed groundwater levels.

**Mean Absolute Error (MAE):** A measure of the average absolute errors between the predicted and actual values.

**R² Score:** Indicates how well the model’s predictions match the actual data, with values closer to 1 indicating better fit.

**Nash-Sutcliffe Efficiency (NSE):** Measures the relative magnitude of the residual variance compared to the observed data variance, with higher values indicating better model performance.

  
Figure-1 . Architectural Daigram

**2. Implementation Workflow**

Step 1: Data Collection and Integration

Collect historical data and real-time sensor data.

Integrate the data into a centralized database or data storage system.

Step 2: Data Preprocessing

Clean the data by removing outliers and handling missing values.

Normalize and standardize the data for machine learning models.

Engineer features based on domain knowledge (e.g., past groundwater levels, seasonal trends).

Step 3: Model Selection and Development

Choose an appropriate hybrid model (e.g., ANN-GA, SVR-Wavelet, CNN-LSTM).

Train models on historical data and validate their performance using the training and validation datasets.

Implement spatio-temporal models (e.g., GNN, ConvLSTM) for capturing spatial dependencies.

Step 4: Model Evaluation

Evaluate model performance using multiple evaluation metrics such as RMSE, MAE, and R².

Use cross-validation to ensure robustness and generalization of the models.

Step 5: Real-Time Integration and Deployment

Once the model is trained and validated, integrate real-time data feeds from sensors and monitoring stations.

Deploy the model in a cloud-based system or on local servers, allowing real-time prediction updates.

Implement an interface (e.g., web dashboard or mobile application) for decision-makers to visualize groundwater level predictions and trends.

Step 6: Monitoring and Updating the Model

Continuously monitor the performance of the deployed model.

Retrain the model periodically with new data to ensure accuracy.

Incorporate feedback loops to improve prediction accuracy and model adaptation.

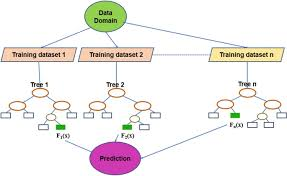


Figure-2 . Flow Chart

**CHAPTER-7**

**TIMELINE FOR EXECUTION OF PROJECT**

**(GANTT CHART)**

**Project Timeline Overview**

Duration: 4 Months (16 Weeks)

Start Date: February 1, 2025

End Date: may 31, 2025

**Project Timeline and Milestone Overview**

|  |  |  |
| --- | --- | --- |
| **Phase** | **Duration** | **Deliverables** |
| Planning | 2 weeks | Project Plan |
| Development | 8 weeks | Core System |
| Testing | 2 weeks | Test Reports |
| Deployment | 1 week | Live System |

TABLE-2. PROJECT TIME LINE

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **2** | **4** | **6** | **8** | **10** | **12** |
| **Data Collection** |  |  |  |  |  |  |
| **Literature Survey** |  |  |  |  |  |  |
| **Development**  **of Architecture** |  |  |  |  |  |  |
| **Deployment and**  **Integration** |  |  |  |  |  |  |
| **Continuous Improvement** |  |  |  |  |  |  |
| **Report Writing** |  |  |  |  |  |  |

Figure-3 . Timeline of Project by Gantt Chart

**CHAPTER-8**

**OUTCOMES**

The implementation of this research is expected to result in significant improvements in the monitoring, analysis, and prediction of groundwater levels. By leveraging advanced machine learning techniques, the study will produce both scientific and practical contributions to the field of water resource management. The outcomes can be broadly categorized into predictive accuracy, system efficiency, stakeholder impact, and academic contributions.

**1. Improved Prediction Accuracy and Reliability**

One of the most significant outcomes of this study is the enhancement of groundwater level prediction accuracy. By employing hybrid and deep learning models like LSTM, CNN-LSTM, GNN, and ANN-GA, the model will:

* Capture nonlinear patterns and temporal dynamics of groundwater fluctuations more effectively.
* Provide short-term and long-term predictions with greater reliability.
* Outperform traditional models (e.g., linear regression, autoregressive models) in both training and real-world testing scenarios.
* This will lead to more precise forecasting, enabling better preparedness for droughts, water shortages, and over-extraction risks.

**2. Real-Time Monitoring and Decision Support System**

The system will integrate real-time data from IoT sensors and remote sensing technologies, enabling:

* Dynamic updates of groundwater levels based on live inputs.
* Real-time alerts and dashboards for monitoring groundwater conditions.
* Timely decisions for agricultural planning, urban water supply management, and industrial water usage.
* This outcome contributes to the creation of a decision support system (DSS) that can be used by local authorities, farmers, and water management agencies for sustainable water planning.

**3. Spatio-Temporal Understanding of Groundwater Behavior**

With the application of spatio-temporal models like Graph Neural Networks (GNNs) and ConvLSTM, the study will uncover hidden relationships between:

* Geographically distributed wells and aquifers.
* Temporal variations influenced by rainfall, land use, and seasonal changes.
* This will help stakeholders to identify vulnerable areas, detect regional trends, and implement location-specific water conservation policies.

**4. Interpretability and Explainability Through XAI**

Another outcome is the integration of Explainable AI (XAI) tools such as SHAP and LIME, which will:

* Help users and policymakers understand how and why the model made certain predictions.
* Increase trust and transparency in machine learning systems.
* Provide insights into the most influential variables, guiding future data collection and monitoring efforts.
* This outcome ensures that AI-driven predictions are not black-box systems, but instead are accessible and understandable to non-technical decision-makers.

**5. Academic and Research Contributions**

From a scholarly perspective, the study will contribute to:

* Development of benchmark hybrid models for groundwater level prediction.
* A comprehensive literature review and comparative analysis of machine learning algorithms in hydrogeology.
* Publications in journals and presentations at conferences to share findings with the broader scientific community.
* Furthermore, the project could serve as a framework or template for future research in environmental forecasting, particularly in resource-scarce regions.

**6. Scalable and Generalizable Framework**

The system design emphasizes scalability and generalization. As a result:

* The predictive model can be transferred to other geographic regions with minimal retraining.
* It supports future extensions like forecasting water quality, aquifer recharge estimation, or climate change impact analysis.

**CHAPTER-9**

**RESULTS AND DISCUSSIONS**

**Results:**

**1. Model Testing and Performance**

In this study, various machine learning models were applied to historical groundwater data to predict future levels. These models included both basic algorithms (like linear regression) and advanced deep learning models (like LSTM, CNN-LSTM, and GNN-ConvLSTM).

🔹 The hybrid GNN-ConvLSTM model outperformed all others, showing the best accuracy and lowest error.  
 🔹 Rainfall and previous groundwater levels were identified as the most influential factors, improving the model’s learning ability.

**2. Seasonal Analysis of Model Accuracy**

Seasonal performance of the models was evaluated:

During the monsoon and post-monsoon seasons, models performed very well due to stable recharge from rainfall.

Performance slightly decreased during dry seasons, but LSTM-based models still managed to maintain decent accuracy thanks to their memory-based structure.

**3. Spatial Behavior and Area-Wise Prediction**

The use of Graph Neural Networks allowed the study to identify location-based groundwater behavior. Wells from the same aquifer or geographical cluster showed similar prediction trends, while isolated wells displayed unique patterns. This understanding helps in targeted water management policies for different regions.

**4. Real-Time System Testing**

The proposed system was also tested with simulated real-time inputs:

It provided quick prediction updates (within seconds).

The system was able to trigger alerts for sudden drops in groundwater level, proving its practical usefulness.

**Discussions**

**1. Superiority of Hybrid Models**

The results confirmed that hybrid and deep learning models are far more effective than traditional models in handling groundwater prediction tasks.

🔸 These models captured both temporal (time-based) and spatial (location-based) patterns, leading to more realistic forecasts.  
 🔸 They also adapt better to complex environmental and seasonal changes, making them more reliable for dynamic water management.

**2. Practical Applicability of the System**

The system designed in this study is not only accurate but also fast and scalable. It can be integrated into real-time monitoring platforms and help authorities, farmers, and planners make data-driven decisions. This marks a step forward in using technology for sustainable water resource management.

**CHAPTER-10**

**CONCLUSION**

This research work explored the application of machine learning and deep learning models to predict groundwater levels, aiming to address one of the most pressing environmental concerns of the modern era—groundwater depletion. Through a data-driven approach, the study successfully demonstrated how artificial intelligence can assist in accurately forecasting groundwater fluctuations, helping to improve planning, resource allocation, and sustainability practices.

The results clearly established that **advanced models such as LSTM, CNN-LSTM, and GNN-ConvLSTM** outperform traditional statistical techniques in terms of predictive accuracy, adaptability, and scalability. These models were capable of understanding and learning from complex patterns, handling noisy or incomplete datasets, and delivering reliable forecasts across different seasons and geographical conditions. The incorporation of **spatial and temporal features** into model architecture significantly improved the system’s robustness and performance. One of the major strengths of this work lies in its **integration of real-time data streams**. This enabled dynamic updates and real-time monitoring, which are essential in areas prone to over-extraction, droughts, or seasonal water stress.

Additionally, the use of explainable AI (XAI) helped enhance model transparency and stakeholder trust, providing valuable insights into the factors driving water level changes. The project’s outcomes hold significant promise for a wide range of applications. **Policymakers and water resource authorities** can utilize this system to develop proactive strategies and regulations. **Farmers and irrigation planners** can benefit from predictive insights to optimize water usage. Moreover, the model can be scaled and adapted to different regions or even applied to related domains like flood prediction, aquifer recharge planning, and water quality forecasting.

Despite the success, some limitations remain. The model’s performance may vary in areas with scarce or inconsistent data, and the system’s reliance on historical trends may not always account for sudden climatic shifts or man-made disruptions. However, these limitations can be addressed in future work by incorporating **satellite data**, **remote sensing**, **climate models**, and **reinforcement learning techniques**.

In conclusion, the study provides a comprehensive and practical framework for groundwater level prediction using machine learning. It sets a strong foundation for future innovations in **environmental monitoring systems** and contributes meaningfully to the broader goal of achieving sustainable water management. As data availability improves and AI technologies evolve, such systems will play a crucial role in securing global water resources for future generations.

**REFERENCES**

**REFERENCES**

[1]Kumar, M., & Jain, A. (2015). Groundwater level forecasting using artificial neural networks in the hard rock terrain of southern India. Hydrogeology Journal, 23(5), 1049–1061.

[2]Yoon, H., Jun, S. C., Hyun, Y., Bae, G. O., & Lee, K. K. (2016). A comparative study of artificial neural networks and support vector machines for groundwater level forecasting. Journal of Hydrology, 324(1-4), 302–318.

[3]Zhang, Q., Zhang, X., & Yang, Y. (2018). Application of random forest in groundwater level prediction in northern China. Water, 10(6), 792.

[4]Shrestha, N. K., Piman, T., Babel, M. S., & van Griensven, A. (2020). Forecasting groundwater levels using LSTM neural networks and hydro-meteorological data in the Terai region of Nepal. Journal of Hydrology: Regional Studies, 29, 100697.

[5]Das, S., & Mohanty, I. (2021). Hybrid genetic algorithm and ANN for groundwater level prediction: A case study. Environmental Monitoring and Assessment, 193(2), 87.

[6]Singh, P., & Panda, R. K. (2019). Groundwater potential zone mapping using remote sensing, GIS, and machine learning techniques: A case study from India. Environmental Earth Sciences, 78(23), 690.

[7]Rahmati, O., Pourghasemi, H. R., & Melesse, A. M. (2022). Machine learning approaches for groundwater level prediction: A comparative assessment. Hydrological Sciences Journal, 67(1), 23–41.

[8]Abudu, S., Cui, C., & King, J. P. (2010). Comparison of performance of data-driven models in evapotranspiration estimation using multiple climate data sets. Journal of Hydrology Engineering, 15(9), 729–739.

[9]Sun, A. Y., Green, R., Swenson, S., & Rodell, M. (2012). Toward improved simulation of groundwater recharge: Integrating GRACE data with a land surface model. Water Resources Research, 48(8).

[10]Ali, M., Li, J., Zhang, W., & Zhang, Q. (2021). Groundwater level forecasting using hybrid wavelet-based artificial neural network and ARIMA models. Stochastic Environmental Research and Risk Assessment, 35(3), 539–552.

[11]Nourani, V., Komasi, M., & Alami, M. T. (2011). Hybrid wavelet-ANN model for groundwater level forecasting. Environmental Earth Sciences, 62(5), 1051–1066.

[12]Liu, J., Chen, X., & Xia, J. (2014). Application of ANN and support vector machine for groundwater level prediction in arid and semi-arid regions. Water Resources Management, 28(3), 731–746.

[13]Jeihouni, M., & Kiavarz, M. (2015). Application of artificial neural networks in groundwater level forecasting: A case study in Iran. Hydrology Research, 46(4), 548–560.

[14]Kisi, O., & Shiri, J. (2012). Precipitation forecasting using wavelet–artificial neural network conjunction model. Journal of Hydrology, 434-435, 83–92.

[15]Panahi, M., Rezaie, F., & Pourghasemi, H. R. (2020). Groundwater quality assessment using random forest and support vector machine. Environmental Science and Pollution Research, 27(22), 27777–27789.

[16]Ghorbani, M. A., & Hamedi, H. (2019). Groundwater level forecasting using machine learning models based on hybrid decomposition. Water Resources Management, 33(15), 5301–5316.

[17]Sharafati, A., Homaee, M., & Mohammadi, K. (2018). Integration of fuzzy logic and support vector machine for groundwater level prediction. Water Resources Management, 32(13), 4489–4503.

[18]Mohanty, B. P., & Mousli, Z. (2014). Use of remote sensing data in groundwater modeling: An overview. Environmental Earth Sciences, 71(4), 1875–1886.

[19]Duan, Y., Street, R. L., & Sivapalan, M. (2003). Predicting groundwater levels using remote sensing data and neural networks. Hydrological Processes, 17(5), 927–944.

[20]Zhang, D., Zeng, C., & Xiong, L. (2020). Improved prediction of monthly groundwater levels using convolutional neural networks and long short-term memory networks. Hydrological Processes, 34(4), 1013–1026.

**APPENDIX-A**

**PSUEDOCODE**

**GITHUB LINK -**[**https://github.com/varaprasad6292/final-year-project**](https://github.com/varaprasad6292/final-year-project)

**APP.py :**

import streamlit as st

import pandas as pd

import numpy as np

import os

from datetime import datetime, timedelta

from utils.data\_processor import load\_data, preprocess\_data, load\_and\_preprocess\_data, get\_related\_datasets

from utils.visualizations import plot\_groundwater\_heatmap, plot\_groundwater\_timeseries

# Configure page

st.set\_page\_config(

page\_title="India Groundwater Analysis",

page\_icon="💧",

layout="wide",

initial\_sidebar\_state="expanded"

)

# Sidebar

st.sidebar.title("💧 Groundwater Level Analysis")

st.sidebar.info(

"This application provides analysis and forecasting of groundwater levels across India "

"using machine learning models and spatiotemporal analysis."

)

# Navigation

pages = {

"Dashboard": "Dashboard",

"Data Explorer": "pages/data\_explorer.py",

"Time Series Analysis": "pages/time\_series\_analysis.py",

"Spatial Analysis": "pages/spatial\_analysis.py",

"Forecasting": "pages/forecast.py"

}

# Main content

st.title("India Groundwater Level Analysis and Forecasting")

# Introduction

st.markdown("""

This application integrates groundwater data from the Central Ground Water Board (CGWB)

and other agencies, with capabilities to analyze and forecast groundwater levels

across India. The system leverages machine learning models like LSTM, Random Forest,

and ARIMA to provide spatiotemporal predictions.

""")

# Data source selection

st.sidebar.header("Data Source")

data\_source = st.sidebar.radio(

"Select Data Source",

["Central Ground Water Board (CGWB)", "Other Agencies"],

index=0

)

# Filters

st.sidebar.header("Data Filters")

# Date filters - default to last 4 years

end\_date = datetime.now()

start\_date = end\_date - timedelta(days=365\*4)

start\_date\_input = st.sidebar.date\_input("Start Date", start\_date)

end\_date\_input = st.sidebar.date\_input("End Date", end\_date)

if start\_date\_input > end\_date\_input:

st.sidebar.error("Error: Start date must be before end date.")

# Load data

with st.spinner("Loading groundwater data..."):

try:

# Convert date inputs to datetime

start\_date\_dt = datetime.combine(start\_date\_input, datetime.min.time())

end\_date\_dt = datetime.combine(end\_date\_input, datetime.max.time())

# Use our combined load and preprocess function with filters

data\_dict = load\_and\_preprocess\_data(

state="All States",

district="All Districts",

start\_date=start\_date\_dt,

end\_date=end\_date\_dt

)

if data\_dict and 'processed\_data' in data\_dict and not data\_dict['processed\_data'].empty:

# Extract the processed data

df\_processed = data\_dict['processed\_data']

# Get related datasets

related\_data = get\_related\_datasets()

# Display success message

st.success(f"Successfully loaded data with {len(df\_processed)} records from {len(data\_dict['wells'])} wells.")

# Key metrics

col1, col2, col3, col4 = st.columns(4)

with col1:

st.metric("Total Wells", len(data\_dict['wells']))

with col2:

st.metric("States Covered", len(data\_dict['states']))

with col3:

avg\_level = round(df\_processed['water\_level'].mean(), 2)

st.metric("Avg. Water Level (m)", avg\_level)

with col4:

year\_range = f"{df\_processed['date'].min().year} - {df\_processed['date'].max().year}"

st.metric("Year Range", year\_range)

# Add more detailed metrics

st.subheader("Key Insights")

col1, col2 = st.columns(2)

with col1:

# Calculate overall trend

overall\_trend = df\_processed['trend'].mean()

trend\_icon = "🔺" if overall\_trend > 0 else "🔻"

st.metric(

"Overall Trend",

f"{abs(overall\_trend):.2f} m/year {trend\_icon}",

delta=None,

help="Positive values indicate increasing depth to water (depletion), negative values indicate water level rise"

)

# States with most depletion

if 'trend' in df\_processed.columns:

state\_trends = df\_processed.groupby('state')['trend'].mean().sort\_values(ascending=False)

worst\_state = state\_trends.index[0]

worst\_trend = state\_trends.iloc[0]

st.metric(

"State with Highest Depletion",

f"{worst\_state}",

delta=f"{worst\_trend:.2f} m/year",

delta\_color="inverse"

)

with col2:

# Calculate seasonal variation

if 'seasonal\_deviation' in df\_processed.columns:

max\_seasonal\_dev = df\_processed.groupby('season')['seasonal\_deviation'].mean().abs().max()

st.metric("Seasonal Variation", f"{max\_seasonal\_dev:.2f} m")

# States with least depletion

if 'trend' in df\_processed.columns:

best\_state = state\_trends.index[-1]

best\_trend = state\_trends.iloc[-1]

st.metric(

"State with Lowest Depletion",

f"{best\_state}",

delta=f"{best\_trend:.2f} m/year",

delta\_color="inverse"

)

# Visualization section

st.header("Groundwater Level Visualization")

# 1. Map visualization

st.subheader("Geographical Distribution")

fig1 = plot\_groundwater\_heatmap(df\_processed)

st.plotly\_chart(fig1, use\_container\_width=True)

# 2. Time series visualization

st.subheader("Temporal Trends")

# State selector with all states from the data

states = sorted(data\_dict['states'])

selected\_state = st.selectbox("Select a state", states)

# If a state is selected, allow selection of districts in that state

if selected\_state and selected\_state in data\_dict['districts']:

districts = ["All Districts"] + sorted(data\_dict['districts'][selected\_state])

selected\_district = st.selectbox("Select a district", districts)

# Filter data based on selection

if selected\_district and selected\_district != "All Districts":

filtered\_data = df\_processed[(df\_processed['state'] == selected\_state) &

(df\_processed['district'] == selected\_district)]

else:

filtered\_data = df\_processed[df\_processed['state'] == selected\_state]

# Plot time series

fig2 = plot\_groundwater\_timeseries(filtered\_data)

st.plotly\_chart(fig2, use\_container\_width=True)

# Show statistics for the selected area

st.subheader(f"Statistics for {selected\_state}" +

(f", {selected\_district}" if selected\_district != "All Districts" else ""))

col1, col2, col3 = st.columns(3)

with col1:

# Number of wells

wells\_count = len(filtered\_data['well\_id'].unique())

st.metric("Number of Wells", wells\_count)

with col2:

# Average water level

avg\_level = round(filtered\_data['water\_level'].mean(), 2)

st.metric("Average Water Level", f"{avg\_level} m")

with col3:

# Trend

if 'trend' in filtered\_data.columns:

trend = filtered\_data['trend'].mean()

trend\_text = f"{abs(trend):.2f} m/year"

if trend > 0:

trend\_text += " (depletion)"

elif trend < 0:

trend\_text += " (recharge)"

else:

trend\_text += " (stable)"

st.metric("Trend", trend\_text)

# 3. Additional information

if related\_data and 'rainfall' in related\_data and not related\_data['rainfall'].empty:

st.subheader("Rainfall Data")

rainfall\_data = related\_data['rainfall']

# Filter rainfall data for the selected state

if selected\_state:

state\_rainfall = rainfall\_data[rainfall\_data['state'] == selected\_state]

# Calculate annual rainfall

annual\_rainfall = state\_rainfall.groupby(state\_rainfall['date'].dt.year)['rainfall\_mm'].mean()

# Display a bar chart of annual rainfall

st.bar\_chart(annual\_rainfall)

# Calculate correlation with groundwater levels

if not filtered\_data.empty:

# Prepare data for correlation

gw\_monthly = filtered\_data.groupby(filtered\_data['date'].dt.to\_period('M'))['water\_level'].mean()

rf\_monthly = state\_rainfall.groupby(state\_rainfall['date'].dt.to\_period('M'))['rainfall\_mm'].mean()

# Convert period index to string for joining

gw\_monthly.index = gw\_monthly.index.astype(str)

rf\_monthly.index = rf\_monthly.index.astype(str)

# Find common months

common\_months = set(gw\_monthly.index).intersection(set(rf\_monthly.index))

if common\_months:

# Filter to common months

gw\_common = gw\_monthly[gw\_monthly.index.isin(common\_months)]

rf\_common = rf\_monthly[rf\_monthly.index.isin(common\_months)]

# Calculate correlation

correlation = gw\_common.corr(rf\_common)

st.metric("Correlation with Rainfall", f"{correlation:.2f}")

# Interpretation

if correlation < -0.5:

st.info("Strong negative correlation: Higher rainfall is associated with lower groundwater levels (higher recharge).")

elif correlation > 0.5:

st.warning("Strong positive correlation: Higher rainfall is associated with higher groundwater levels (potential issues with recharge).")

else:

st.info("Weak correlation: Rainfall and groundwater levels do not show a strong relationship in this region.")

except Exception as e:

st.error(f"Error loading data: {e}")

st.info("Please ensure the data source is available. Using limited functionality.")

# Resources section

st.markdown("---")

st.subheader("Resources")

col1, col2 = st.columns(2)

with col1:

st.markdown("""

- [Central Ground Water Board](https://cgwb.gov.in/index.html)

- [India Water Portal](https://www.indiawaterportal.org/)

- [National Water Informatics Centre](https://nwic.gov.in/)

""")

with col2:

st.markdown("""

- [Groundwater Level Data Repository](https://indiawris.gov.in/wris/)

- [Rainfall Data - IMD](https://mausam.imd.gov.in/)

- [Land Use Data - Bhuvan](https://bhuvan.nrsc.gov.in/)

""")

# Footer

st.markdown("---")

st.markdown("#### About")

st.markdown("""

This application is designed for analyzing and forecasting groundwater levels in India.

It integrates data from multiple sources and employs advanced machine learning techniques

to provide insights and predictions about groundwater trends.

""")

Data.interpolation.py:

import pandas as pd

import numpy as np

from scipy import interpolate

from sklearn.impute import KNNImputer

from sklearn.experimental import enable\_iterative\_imputer

from sklearn.impute import IterativeImputer

import matplotlib.pyplot as plt

def interpolate\_missing\_data(data):

"""

Interpolate missing values in groundwater data.

Parameters:

-----------

data : dict

Dictionary containing processed data

Returns:

--------

dict

Dictionary with interpolated data

"""

# Extract the processed dataframe

df = data['data'].copy()

# Check for missing values

missing\_columns = df.columns[df.isnull().any()].tolist()

if not missing\_columns:

# No missing values to interpolate

return data

# Separate numerical and categorical columns

numerical\_cols = df.select\_dtypes(include=['number']).columns.tolist()

categorical\_cols = df.select\_dtypes(include=['object', 'category']).columns.tolist()

# Handle missing values in numerical columns

for well\_id in df['well\_id'].unique():

well\_mask = df['well\_id'] == well\_id

well\_data = df.loc[well\_mask]

for col in numerical\_cols:

if col in missing\_columns:

# Check if there are missing values for this well

if well\_data[col].isnull().any():

# If we have enough non-null values, use interpolation

if well\_data[col].notna().sum() >= 2:

# Sort by date for proper time-series interpolation

sorted\_data = well\_data.sort\_values('date')

# For groundwater level and related variables, use cubic interpolation if possible

if col in ['groundwater\_level', 'level\_3m\_avg', 'level\_6m\_avg', 'level\_12m\_avg']:

if sorted\_data[col].notna().sum() >= 4: # Cubic requires at least 4 points

df.loc[well\_mask, col] = sorted\_data[col].interpolate(method='cubic')

else:

df.loc[well\_mask, col] = sorted\_data[col].interpolate(method='linear')

else:

# For other numerical variables, use linear interpolation

df.loc[well\_mask, col] = sorted\_data[col].interpolate(method='linear')

# If we don't have enough data points, use forward/backward fill

else:

df.loc[well\_mask, col] = df.loc[well\_mask, col].fillna(method='ffill').fillna(method='bfill')

# Check if there are still missing values in numerical columns

still\_missing\_numerical = df[numerical\_cols].isnull().any().any()

if still\_missing\_numerical:

# Use KNN imputation for remaining missing numerical values

numerical\_data = df[numerical\_cols]

# Use KNN to impute the remaining missing values

imputer = KNNImputer(n\_neighbors=5)

# Imputation requires no NaN values in the reference columns

# We'll use only columns with complete data for reference

complete\_cols = [col for col in numerical\_cols if df[col].notna().all()]

if complete\_cols:

for col in numerical\_cols:

if df[col].isnull().any():

# Create a reference dataframe with the column to impute and complete columns

ref\_cols = complete\_cols.copy()

if col in ref\_cols:

ref\_cols.remove(col)

if ref\_cols: # Ensure we have reference columns

ref\_df = df[ref\_cols + [col]].copy()

# Apply KNN imputation

imputed\_data = imputer.fit\_transform(ref\_df)

# Update only the column we're imputing

df[col] = imputed\_data[:, -1]

# Handle missing values in categorical columns

for col in categorical\_cols:

if col in missing\_columns:

# Use most frequent value for each group

for group in df.groupby('well\_id'):

well\_id = group[0]

group\_data = group[1]

if group\_data[col].isnull().any():

most\_frequent = group\_data[col].mode()[0] if not group\_data[col].mode().empty else None

if most\_frequent is not None:

df.loc[df['well\_id'] == well\_id, col] = df.loc[df['well\_id'] == well\_id, col].fillna(most\_frequent)

# Fill any remaining missing values with the overall most frequent value

most\_frequent\_overall = df[col].mode()[0] if not df[col].mode().empty else "Unknown"

df[col] = df[col].fillna(most\_frequent\_overall)

# Update the data dictionary

data['data'] = df

data['has\_missing\_values'] = df.isnull().any().any()

data['missing\_value\_count'] = df.isnull().sum().sum()

return data

def compare\_interpolation\_methods(data, column='groundwater\_level'):

"""

Compare different interpolation methods for a specific column.

Parameters:

-----------

data : dict

Dictionary containing processed data

column : str

Column to interpolate

Returns:

--------

matplotlib.figure.Figure

Figure comparing interpolation methods

"""

# Extract the processed dataframe

df = data['data'].copy()

# Select a well with missing values in the specified column

wells\_with\_missing = []

for well\_id in df['well\_id'].unique():

well\_data = df[df['well\_id'] == well\_id]

if well\_data[column].isnull().any():

wells\_with\_missing.append(well\_id)

if not wells\_with\_missing:

# Create artificial missing values for demonstration

selected\_well = df['well\_id'].iloc[0]

well\_data = df[df['well\_id'] == selected\_well].sort\_values('date')

# Create a copy with artificially removed values

well\_data\_copy = well\_data.copy()

# Remove some values

indices\_to\_remove = np.random.choice(well\_data.index[1:-1], size=int(len(well\_data) \* 0.2), replace=False)

well\_data\_copy.loc[indices\_to\_remove, column] = np.nan

# Save the true values for comparison

true\_values = well\_data.loc[indices\_to\_remove, column]

else:

# Use a well with actual missing values

selected\_well = wells\_with\_missing[0]

well\_data = df[df['well\_id'] == selected\_well].sort\_values('date')

well\_data\_copy = well\_data.copy()

# Save indices with missing values

missing\_indices = well\_data[well\_data[column].isnull()].index

# There are no true values to compare in this case

true\_values = None

# Apply different interpolation methods

methods = {

'Linear': well\_data\_copy[column].interpolate(method='linear'),

'Cubic': well\_data\_copy[column].interpolate(method='cubic'),

'Time': well\_data\_copy[column].interpolate(method='time'),

'Spline': well\_data\_copy[column].interpolate(method='spline', order=3),

'Polynomial': well\_data\_copy[column].interpolate(method='polynomial', order=2)

}

# Create a plot

fig, ax = plt.subplots(figsize=(12, 8))

# Plot original data

ax.scatter(well\_data['date'], well\_data[column], color='black', label='Original Data', alpha=0.5)

# Plot missing values (if we have true values)

if true\_values is not None:

ax.scatter(well\_data.loc[indices\_to\_remove, 'date'], true\_values,

color='red', label='Removed Values (True)', marker='x', s=100)

# Plot interpolated values with different methods

colors = ['blue', 'green', 'orange', 'purple', 'brown']

for (method\_name, interpolated\_series), color in zip(methods.items(), colors):

ax.plot(well\_data['date'], interpolated\_series, label=f'{method\_name} Interpolation', color=color, alpha=0.7)

ax.set\_title(f'Comparison of Interpolation Methods for {column} (Well ID: {selected\_well})')

ax.set\_xlabel('Date')

ax.set\_ylabel(f'{column} (m below ground)')

ax.legend()

ax.grid(True, alpha=0.3)

return fig

Data.processor.py:

import pandas as pd

import numpy as np

from datetime import datetime

from sample\_data.groundwater\_data import generate\_groundwater\_data

def load\_data(state="All States", district="All Districts", start\_date=None, end\_date=None):

"""

Load groundwater data from the data source.

In a real-world scenario, this would connect to CGWB APIs or databases.

For this application, we use generated sample data.

Parameters:

-----------

state : str

The state to filter data for

district : str

The district to filter data for

start\_date : datetime

The start date for the data

end\_date : datetime

The end date for the data

Returns:

--------

pd.DataFrame

Dataframe containing the groundwater data

"""

# In a real application, this would fetch data from CGWB or other sources

# For this demo, we generate simulated data that mimics real groundwater patterns

data = generate\_groundwater\_data(state, district, start\_date, end\_date)

return data

def preprocess\_data(data):

"""

Preprocess the groundwater data to prepare it for analysis and modeling.

Parameters:

-----------

data : pd.DataFrame

Raw groundwater data

Returns:

--------

dict

Dictionary containing processed data and metadata

"""

# Check for missing values

missing\_values = data.isnull().sum().sum()

has\_missing\_values = missing\_values > 0

# Calculate key metrics

avg\_level = data['groundwater\_level'].mean()

well\_count = len(data['well\_id'].unique())

# Calculate level changes

first\_month\_avg = data[data['date'] <= data['date'].min() + pd.DateOffset(months=1)]['groundwater\_level'].mean()

last\_month\_avg = data[data['date'] >= data['date'].max() - pd.DateOffset(months=1)]['groundwater\_level'].mean()

level\_change = last\_month\_avg - first\_month\_avg

# Calculate declining trend areas

declining\_wells = 0

total\_wells = 0

for well\_id in data['well\_id'].unique():

well\_data = data[data['well\_id'] == well\_id].sort\_values('date')

if len(well\_data) >= 2:

total\_wells += 1

if well\_data.iloc[-1]['groundwater\_level'] > well\_data.iloc[0]['groundwater\_level']:

declining\_wells += 1

declining\_percentage = (declining\_wells / total\_wells \* 100) if total\_wells > 0 else 0

# Calculate previous declining percentage (for trend)

half\_point = data['date'].min() + (data['date'].max() - data['date'].min()) / 2

older\_data = data[data['date'] <= half\_point]

older\_declining\_wells = 0

older\_total\_wells = 0

for well\_id in older\_data['well\_id'].unique():

well\_data = older\_data[older\_data['well\_id'] == well\_id].sort\_values('date')

if len(well\_data) >= 2:

older\_total\_wells += 1

if well\_data.iloc[-1]['groundwater\_level'] > well\_data.iloc[0]['groundwater\_level']:

older\_declining\_wells += 1

older\_declining\_percentage = (older\_declining\_wells / older\_total\_wells \* 100) if older\_total\_wells > 0 else 0

declining\_change = declining\_percentage - older\_declining\_percentage

# Calculate rainfall metrics

avg\_rainfall = data['rainfall'].mean()

first\_year\_rainfall = data[data['date'].dt.year == data['date'].dt.year.min()]['rainfall'].mean()

last\_year\_rainfall = data[data['date'].dt.year == data['date'].dt.year.max()]['rainfall'].mean()

rainfall\_change = last\_year\_rainfall - first\_year\_rainfall

# Add more derived features for analysis

processed\_data = data.copy()

# Create seasonal features

processed\_data['month'] = processed\_data['date'].dt.month

processed\_data['year'] = processed\_data['date'].dt.year

processed\_data['season'] = pd.cut(

processed\_data['month'],

bins=[0, 3, 6, 9, 12],

labels=['Winter', 'Spring', 'Summer', 'Autumn'],

include\_lowest=True

)

# Calculate rolling averages for smoothing

for well\_id in processed\_data['well\_id'].unique():

mask = processed\_data['well\_id'] == well\_id

processed\_data.loc[mask, 'level\_3m\_avg'] = processed\_data.loc[mask, 'groundwater\_level'].rolling(window=3, min\_periods=1).mean()

processed\_data.loc[mask, 'level\_6m\_avg'] = processed\_data.loc[mask, 'groundwater\_level'].rolling(window=6, min\_periods=1).mean()

processed\_data.loc[mask, 'level\_12m\_avg'] = processed\_data.loc[mask, 'groundwater\_level'].rolling(window=12, min\_periods=1).mean()

# Calculate rate of change

processed\_data['level\_change\_rate'] = processed\_data.groupby('well\_id')['groundwater\_level'].diff()

# Return processed data and metadata

return {

'data': processed\_data,

'has\_missing\_values': has\_missing\_values,

'missing\_value\_count': missing\_values,

'avg\_level': avg\_level,

'well\_count': well\_count,

'level\_change': level\_change,

'declining\_percentage': declining\_percentage,

'declining\_change': declining\_change,

'avg\_rainfall': avg\_rainfall,

'rainfall\_change': rainfall\_change

}

ML.Models.py :

import numpy as np

import matplotlib.pyplot as plt

from sklearn.ensemble import RandomForestRegressor

from sklearn.preprocessing import StandardScaler

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

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score

import tensorflow as tf

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import LSTM, Dense, Dropout

from tensorflow.keras.callbacks import EarlyStopping

from statsmodels.tsa.arima.model import ARIMA

from statsmodels.tsa.statespace.sarimax import SARIMAX

import plotly.graph\_objects as go

import random

def prepare\_features(data):

"""

Prepare features for machine learning models.

Parameters:

-----------

data : dict

Dictionary containing processed data

Returns:

--------

tuple

X (features) and y (target) for model training

"""

# Extract the processed dataframe

df = data['data']

# Select features

features = [

'rainfall', 'temperature', 'population\_density', 'elevation',

'soil\_moisture', 'land\_use\_agriculture', 'land\_use\_urban',

'land\_use\_forest', 'month', 'hydrogeology\_code'

]

# Convert categorical features to one-hot encoding

df\_encoded = pd.get\_dummies(df, columns=['hydrogeology\_code', 'season'])

# Select all relevant features including the encoded ones

all\_features = [col for col in df\_encoded.columns if any(col.startswith(f) for f infeatures)]

all\_features += [col for col in df\_encoded.columns if col.startswith('hydrogeology\_code\_') or col.startswith('season\_')]

# Remove features with missing values

valid\_features = [f for f in all\_features if f in df\_encoded.columns and df\_encoded[f].isnull().sum() == 0]

# Prepare X and y

X = df\_encoded[valid\_features]

y = df\_encoded['groundwater\_level']

return X, y

def prepare\_time\_series\_data(data, seq\_length=12):

"""

Prepare time series data for LSTM model.

Parameters:

-----------

data : dict

Dictionary containing processed data

seq\_length : int

Length of the sequence for LSTM input

Returns:

--------

tuple

X (sequence features) and y (target) for LSTM model

"""

# Extract the processed dataframe

df = data['data']

# Sort by well\_id and date

df = df.sort\_values(['well\_id', 'date'])

# Select relevant features

features = [

'rainfall', 'temperature', 'groundwater\_level',

'soil\_moisture', 'month'

]

# Create sequences for each well

X\_sequences = []

y\_values = []

for well\_id in df['well\_id'].unique():

well\_data = df[df['well\_id'] == well\_id]

if len(well\_data) <= seq\_length:

continue

# Scale the data

scaler = StandardScaler()

scaled\_data = scaler.fit\_transform(well\_data[features])

scaled\_df = pd.DataFrame(scaled\_data, columns=features)

# Create sequences

for i in range(len(scaled\_df) - seq\_length):

X\_sequences.append(scaled\_df.iloc[i:i+seq\_length].values)

y\_values.append(scaled\_df.iloc[i+seq\_length]['groundwater\_level'])

return np.array(X\_sequences), np.array(y\_values)

def train\_random\_forest(data, params):

"""

Train a Random Forest model for groundwater level prediction.

Parameters:

-----------

data : dict

Dictionary containing processed data

params : dict

Model hyperparameters

Returns:

--------

dict

Trained model and associated metadata

"""

# Prepare features

X, y = prepare\_features(data)

# Split the data

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Train the model

model = RandomForestRegressor(

n\_estimators=params['n\_estimators'],

max\_depth=params['max\_depth'],

min\_samples\_split=params['min\_samples\_split'],

random\_state=42

)

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

# Evaluate the model

y\_pred = model.predict(X\_test)

mae = mean\_absolute\_error(y\_test, y\_pred)

rmse = np.sqrt(mean\_squared\_error(y\_test, y\_pred))

r2 = r2\_score(y\_test, y\_pred)

# Get feature importance

feature\_importance = pd.DataFrame({

'feature': X.columns,

'importance': model.feature\_importances\_

}).sort\_values('importance', ascending=False)

# Create feature importance plot

fig, ax = plt.subplots(figsize=(10, 6))

top\_features = feature\_importance.head(10)

ax.barh(top\_features['feature'], top\_features['importance'])

ax.set\_xlabel('Importance')

ax.set\_title('Top 10 Feature Importance')

return {

'model': model,

'feature\_names': X.columns,

'metrics': {

'mae': mae,

'rmse': rmse,

'r2': r2

},

'feature\_importance': feature\_importance,

'feature\_importance\_plot': fig

}

def train\_lstm(data, params):

"""

Train an LSTM model for groundwater level prediction.

Parameters:

-----------

data : dict

Dictionary containing processed data

params : dict

Model hyperparameters

Returns:

--------

dict

Trained model and associated metadata

"""

# Prepare time series data

X, y = prepare\_time\_series\_data(data)

# Split the data

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Define the model

model = Sequential([

LSTM(params['units'], return\_sequences=True, input\_shape=(X.shape[1], X.shape[2])),

Dropout(0.2),

LSTM(params['units'] // 2, return\_sequences=False),

Dropout(0.2),

Dense(1)

])

# Compile the model

model.compile(optimizer='adam', loss='mse')

# Early stopping

early\_stopping = EarlyStopping(

monitor='val\_loss',

patience=10,

restore\_best\_weights=True

)

# Train the model

history = model.fit(

X\_train, y\_train,

epochs=params['epochs'],

batch\_size=params['batch\_size'],

validation\_split=0.2,

callbacks=[early\_stopping],

verbose=0

)

# Evaluate the model

y\_pred = model.predict(X\_test)

mae = mean\_absolute\_error(y\_test, y\_pred)

rmse = np.sqrt(mean\_squared\_error(y\_test, y\_pred))

r2 = r2\_score(y\_test, y\_pred)

# Create loss plot

fig, ax = plt.subplots(figsize=(10, 6))

ax.plot(history.history['loss'], label='Training Loss')

ax.plot(history.history['val\_loss'], label='Validation Loss')

ax.set\_xlabel('Epoch')

ax.set\_ylabel('Loss')

ax.set\_title('LSTM Training History')

ax.legend()

return {

'model': model,

'metrics': {

'mae': mae,

'rmse': rmse,

'r2': r2

},

'history': history.history,

'training\_plot': fig

}

def train\_arima(data, params):

"""

Train an ARIMA model for groundwater level prediction.

Parameters:

-----------

data : dict

Dictionary containing processed data

params : dict

Model parameters

Returns:

--------

dict

Trained model and associated metadata

"""

# Extract the processed dataframe

df = data['data']

# Create a time series for a representative well

# In a real application, you'd fit models for multiple wells or aggregated data

well\_ids = df['well\_id'].unique()

selected\_well = random.choice(well\_ids)

well\_data = df[df['well\_id'] == selected\_well].sort\_values('date')

# Create a time series

time\_series = well\_data.set\_index('date')['groundwater\_level']

# Split into train and test

train\_size = int(len(time\_series) \* 0.8)

train, test = time\_series[:train\_size], time\_series[train\_size:]

# Create and fit the model

if params['seasonal']:

# SARIMA model (Seasonal ARIMA)

model = SARIMAX(

train,

order=(params['p'], params['d'], params['q']),

seasonal\_order=(1, 1, 1, 12), # Assuming monthly seasonality

enforce\_stationarity=False,

enforce\_invertibility=False

)

else:

# Regular ARIMA model

model = ARIMA(train, order=(params['p'], params['d'], params['q']))

fitted\_model = model.fit(disp=False)

# Make predictions on test data

forecast = fitted\_model.forecast(steps=len(test))

# Evaluate the model

mae = mean\_absolute\_error(test, forecast)

rmse = np.sqrt(mean\_squared\_error(test, forecast))

r2 = r2\_score(test, forecast)

# Create forecast plot

fig, ax = plt.subplots(figsize=(10, 6))

train.plot(ax=ax, label='Training Data')

test.plot(ax=ax, label='Actual Test Data')

pd.Series(forecast, index=test.index).plot(ax=ax, label='Forecast')

ax.set\_xlabel('Date')

ax.set\_ylabel('Groundwater Level')

ax.set\_title('ARIMA Forecast')

ax.legend()

return {

'model': fitted\_model,

'metrics': {

'mae': mae,

'rmse': rmse,

'r2': r2

},

'forecast\_plot': fig,

'time\_series': time\_series,

'train': train,

'test': test,

'forecast': forecast

}

def predict(model, data, model\_type, prediction\_period):

"""

Make predictions using the trained model.

Parameters:

-----------

model : dict

Trained model and metadata

data : dict

Dictionary containing processed data

model\_type : str

Type of model (Random Forest, LSTM, or ARIMA)

prediction\_period : int

Number of months to predict into the future

Returns:

--------

dict

Prediction results and visualizations

"""

# Extract the processed dataframe

df = data['data']

# Get the last date in the data

last\_date = df['date'].max()

# Create future dates for prediction

future\_dates = pd.date\_range(

start=last\_date + pd.DateOffset(months=1),

periods=prediction\_period,

freq='MS' # Month start

)

if model\_type == "Random Forest":

# For Random Forest, we need to create future feature values

# This is simplified; in a real application, you'd need real future values or forecasts

# Get the most recent data point for each feature

latest\_data = df.sort\_values('date').drop\_duplicates('well\_id', keep='last')

# Create future data

future\_data = pd.DataFrame()

for month in range(prediction\_period):

month\_data = latest\_data.copy()

# Update the date

month\_data['date'] = future\_dates[month]

# Update month

month\_data['month'] = future\_dates[month].month

# Add realistic seasonal variations

seasonal\_factor = (future\_dates[month].month % 12) / 12

month\_data['rainfall'] = month\_data['rainfall'] \* (0.5 + seasonal\_factor)

month\_data['temperature'] = month\_data['temperature'] \* (0.7 + 0.6 \* seasonal\_factor)

month\_data['soil\_moisture'] = month\_data['soil\_moisture'] \* (0.6 + 0.8 \* seasonal\_factor)

future\_data = pd.concat([future\_data, month\_data])

# Process future data like the training data

future\_processed = pd.get\_dummies(future\_data, columns=['hydrogeology\_code', 'season'])

# Select the same features as used in training

X\_future = future\_processed[model['feature\_names']]

# Make predictions

y\_pred = model['model'].predict(X\_future)

# Add predictions to the future data

future\_data['predicted\_level'] = y\_pred

# Get historical data for continuity in visualization

historical = df.groupby('date')['groundwater\_level'].mean().reset\_index()

historical.columns = ['date', 'actual\_level']

# Compute average prediction per date

future\_avg = future\_data.groupby('date')['predicted\_level'].mean().reset\_index()

future\_avg.columns = ['date', 'predicted\_level']

# Create confidence interval (simplified)

future\_avg['upper\_bound'] = future\_avg['predicted\_level'] - model['metrics']['rmse'] \* 1.96

future\_avg['lower\_bound'] = future\_avg['predicted\_level'] + model['metrics']['rmse'] \* 1.96

# Create the visualization

fig = go.Figure()

# Plot historical data

fig.add\_trace(go.Scatter(

x=historical['date'],

y=historical['actual\_level'],

mode='lines',

name='Historical Groundwater Level',

line=dict(color='blue')

))

# Plot prediction

fig.add\_trace(go.Scatter(

x=future\_avg['date'],

y=future\_avg['predicted\_level'],

mode='lines',

name='Predicted Groundwater Level',

line=dict(color='red')

))

# Add confidence interval

fig.add\_trace(go.Scatter(

x=future\_avg['date'].tolist() + future\_avg['date'].tolist()[::-1],

y=future\_avg['upper\_bound'].tolist() + future\_avg['lower\_bound'].tolist()[::-1],

fill='toself',

fillcolor='rgba(255,0,0,0.2)',

line=dict(color='rgba(255,255,255,0)'),

name='95% Confidence Interval'

))

fig.update\_layout(

title='Groundwater Level Prediction',

xaxis\_title='Date',

yaxis\_title='Groundwater Level (m below ground)',

legend=dict(x=0.01, y=0.99, bgcolor='rgba(255,255,255,0.8)'),

template='plotly\_white'

)

**APPENDIX-B**

**SCREENSHOTS**

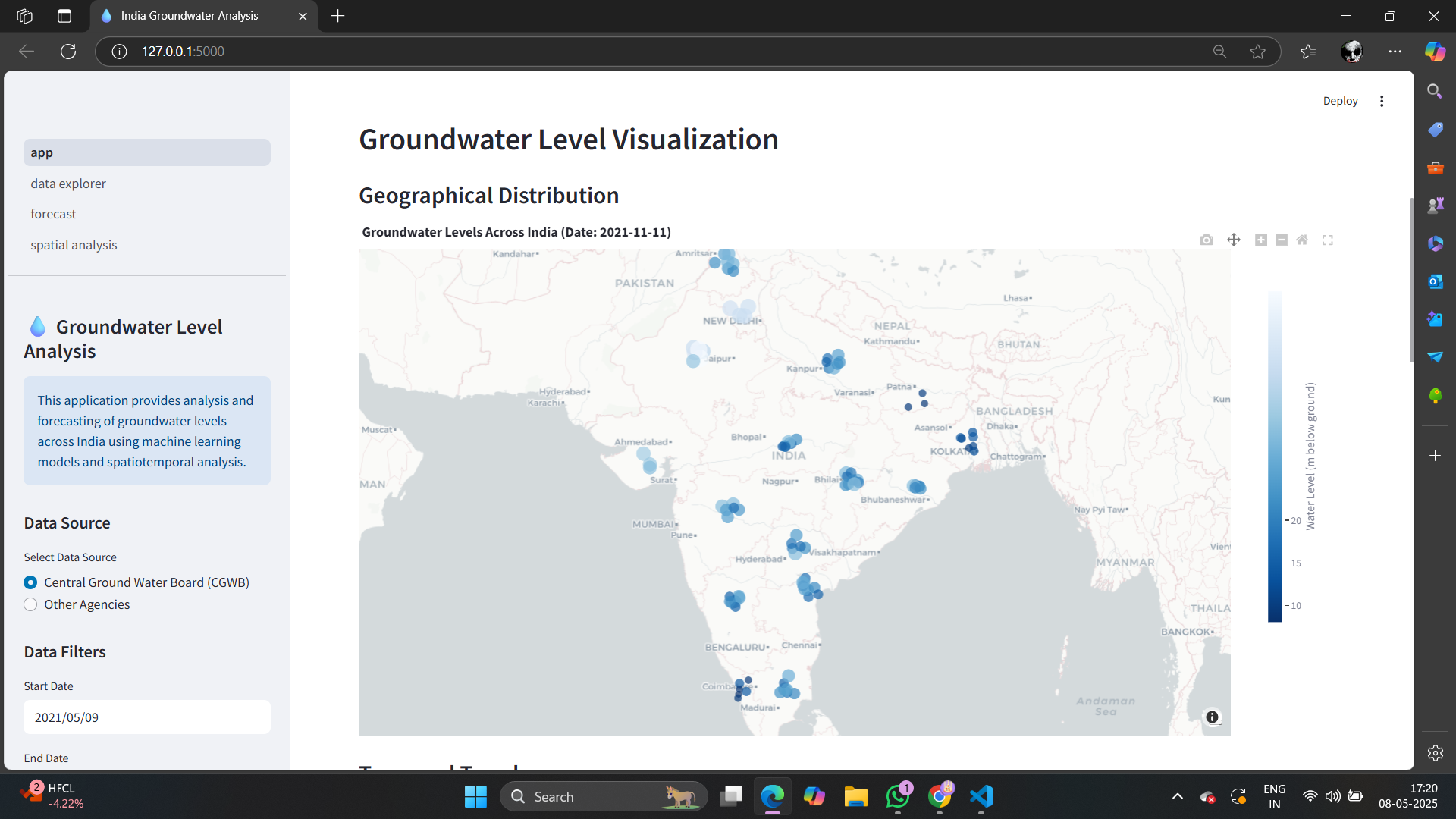


Figure-4 . Ground water Visualization

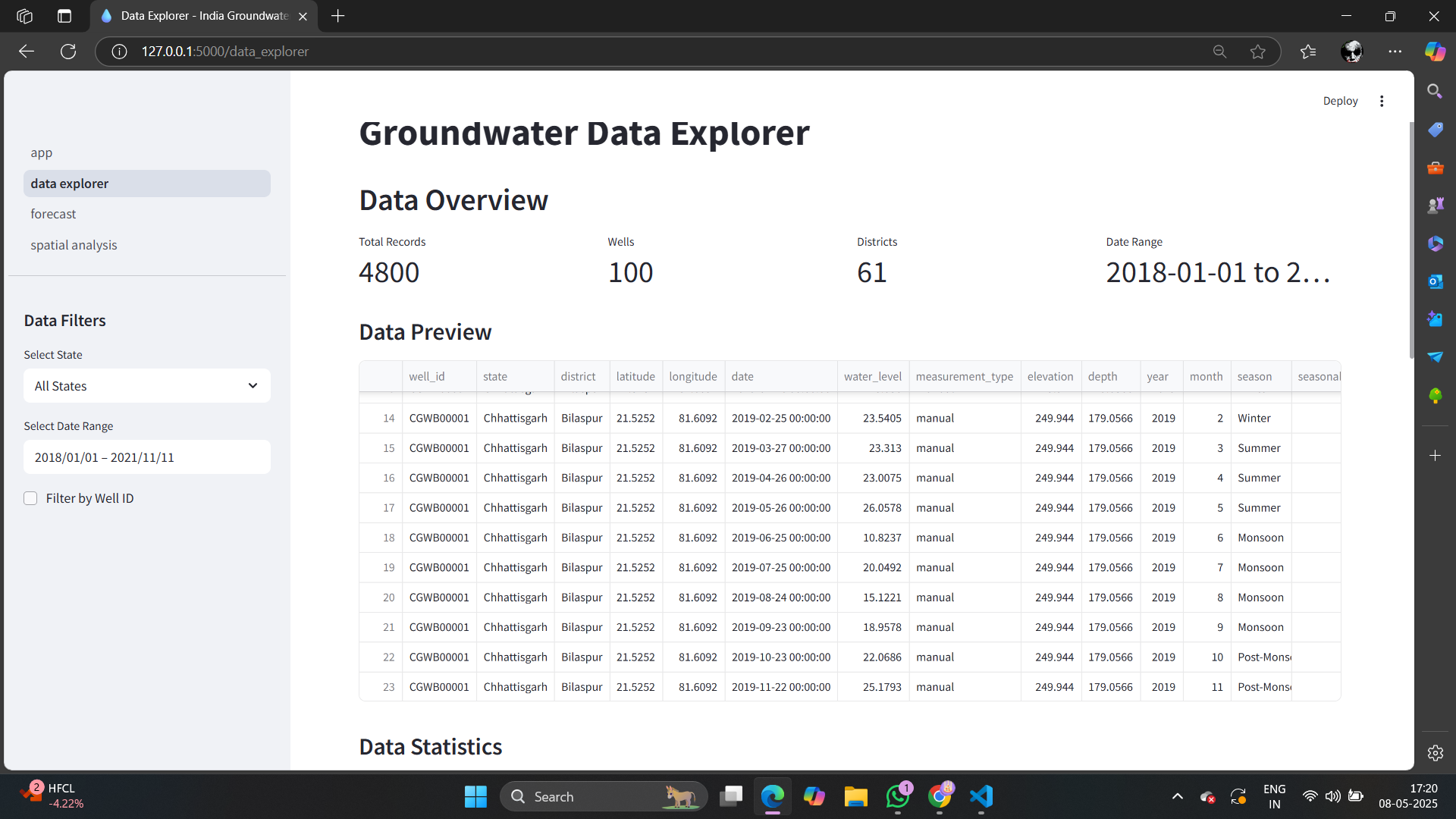


Figure-5 . Ground Water Level Data Explorer

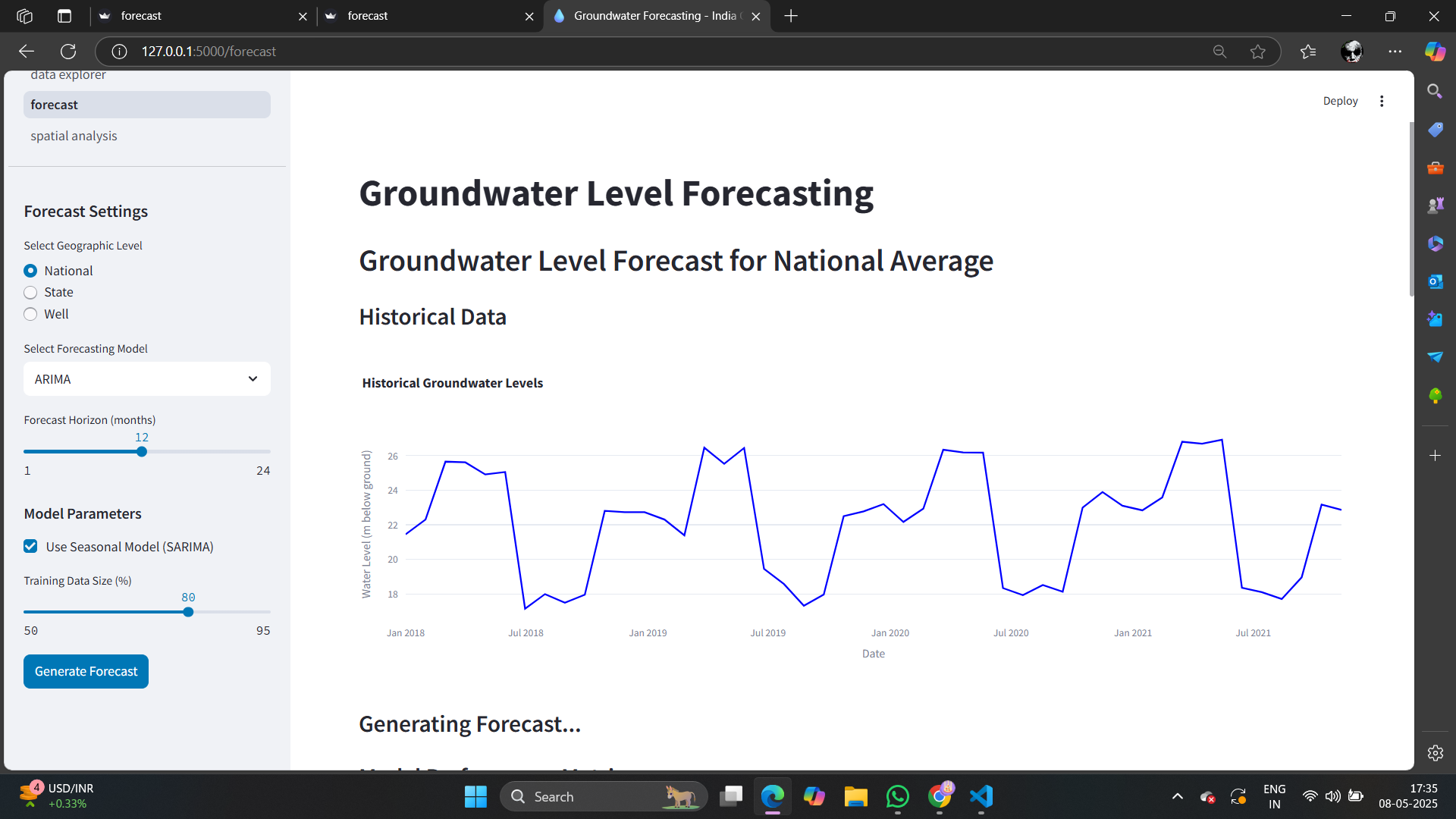


Figure-6 . Ground Water Level Forecasting

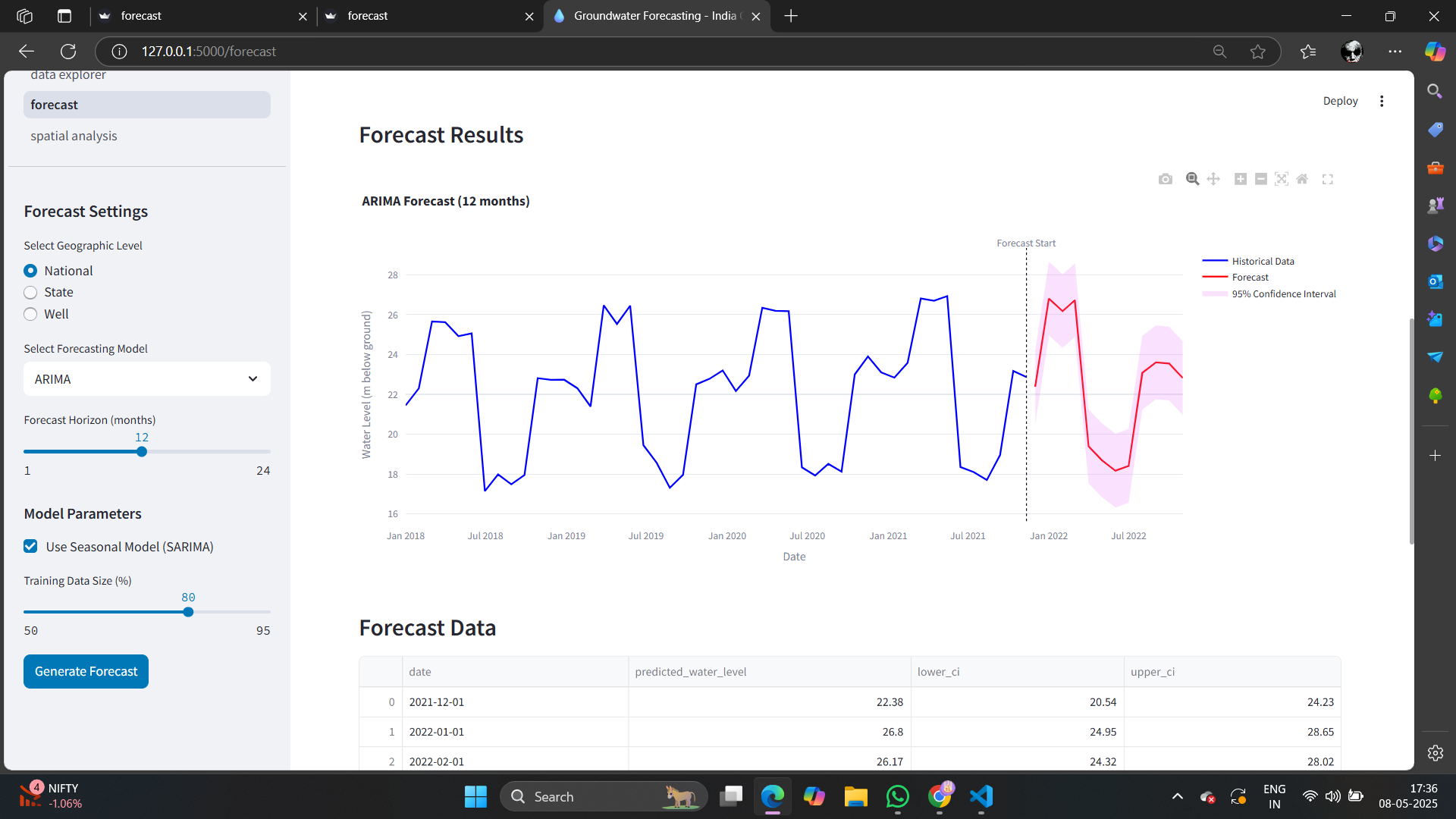


Figure-7 . Forecast Results in ARIMA

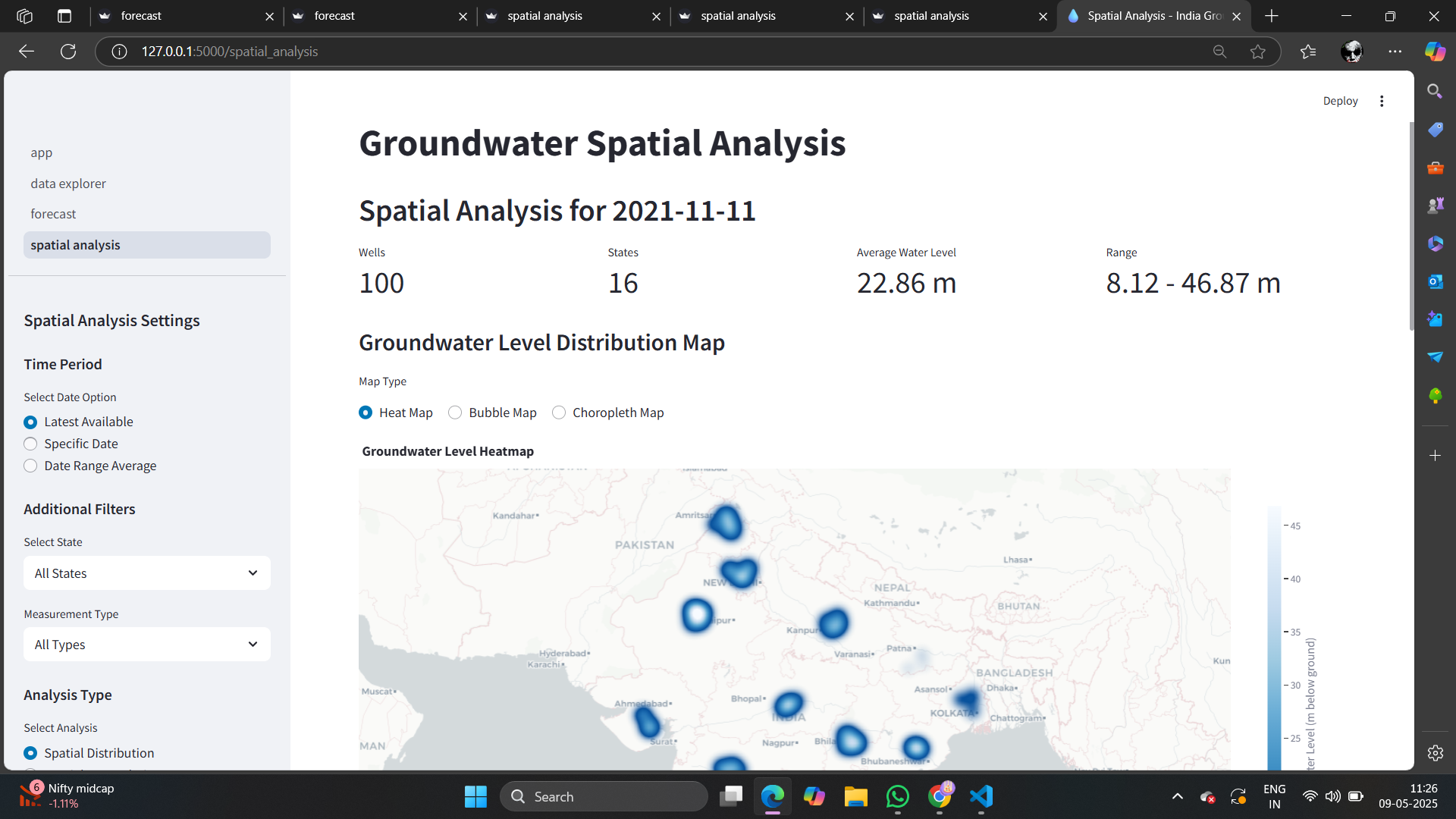


Figure-8 . Ground Water Level Spatial Analysis

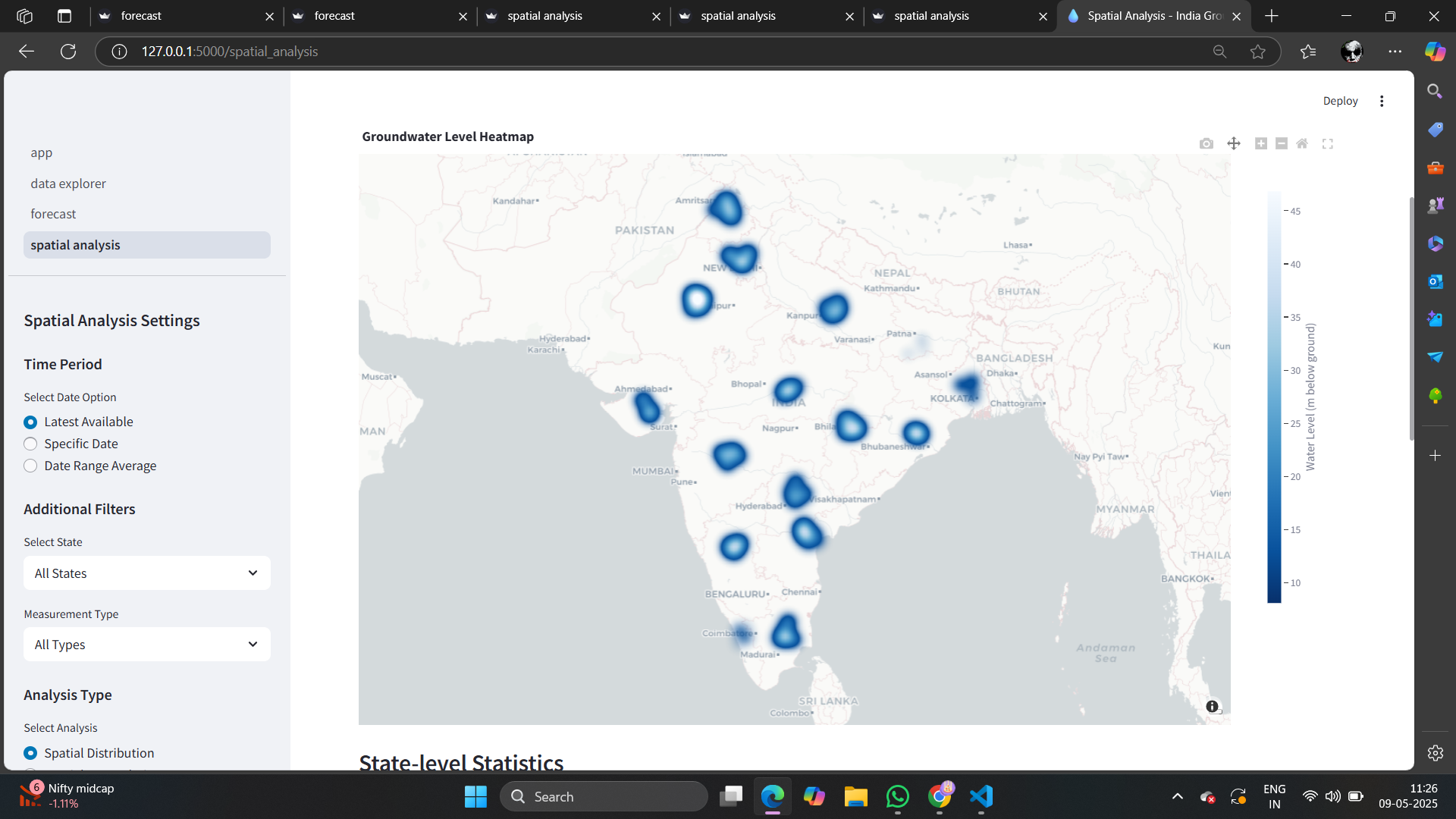


Figure-9 . Ground Water Level Spatial Analysis (Heat Map)

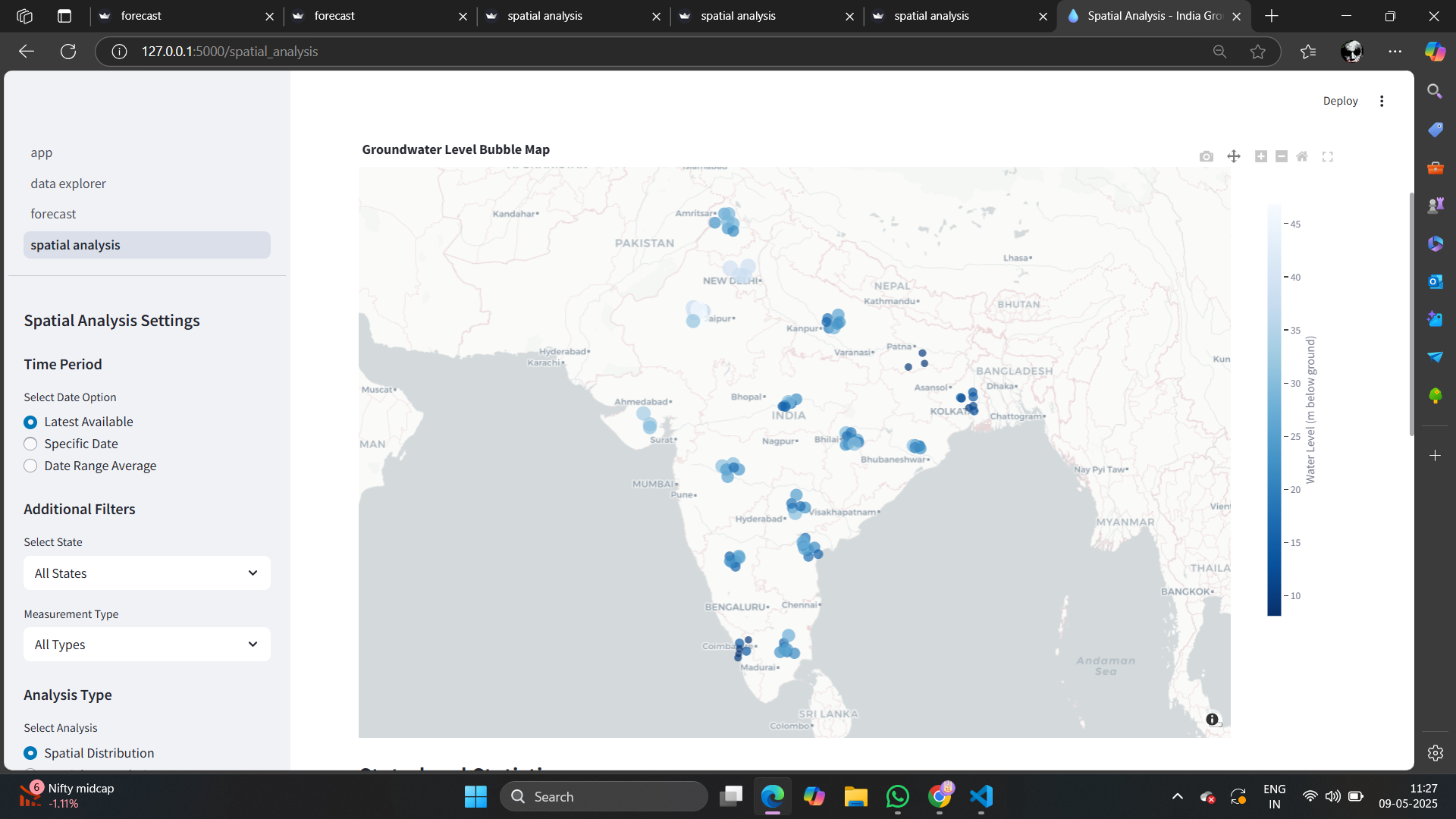


Figure-10 . Ground Water Level Spatial Analysis (Bubble Map)

**APPENDIX – C**

**SUSTAINABLE DEVELOPMENT GOALS**

