**A Project report on**

**WATER QUALITY PREDICTION USING**

**MACHINE LEARNING**

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

**BACHELOR OF TECHNOLOGY**

**In**

**COMPUTER SCIENCE AND ENGINEERING**

***Submitted by***

**P.MRUDUKAR 19B91A05I4**

***Under the Guidance of***

**Sri V. DILIP KUMAR**

Assistant Professor



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**SRKR ENGINEERING COLLEGE (A)**

Chinna Amiram, Bhimavaram, West Godavari Dist., A.P.

[2022-2023]

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**SRKR ENGINEERING COLLEGE (A)**

Chinna Amiram, Bhimavaram, West Godavari Dist., A.P.

[2022-2023]



**BONAFIDE CERTIFICATE**

This is to certify that the project work entitled **“WATER QUALITY PREDICTION USING MACHINE LEARNING”** is the bonafide work of PANDRAKA MRUDUKAR (19B91A05I4), who carried out the project work under my supervision in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering.

**SUPERVISOR HEAD OF THE DEPARTMENT**

Sri V. DILIP KUMAR Dr V. CHANDRA SEKHAR

Assistant Professor Professor & HOD

**SELF DECLARATION**

We hereby declare that the project work entitled “Water Quality Prediction using Machine Learning” is a genuine work carried out by us in B.Tech., (Computer Science and Engineering) at SRKR Engineering College(A), Bhimavaram and has not been submitted either in part or full for the award of any other degree or diploma in any other institute or University.

**P. Mrudukar 19B91A05I4**

# **ABSTRACT**

The degradation of natural water resources is occurring at an alarming rate, making it one of the most critical and concerning issues confronting humanity. The effects of unclean water are far-reaching, impacting every aspect of life. It is possible to effectively combat the effects of water contamination if data is analyzed and water quality can be predicted in advance. Thereupon, it is currently pivotal to develop a model that predicts the quality of water resources and technology can be used to its advantage. Machine learning is a prominent technology for predictions in this day and age, so the Water Quality Index (WQI) can perhaps be calculated by applying its tools. The algorithms like Polynomial regression, Support vector regression, Ridge and Lasso regression, Elastic Net regression are used to predict the WQI. Observed results suggest that Polynomial regression and Ridge and Lasso regression can be implemented properly for practical applications, while Elastic Net regression predicts WQI more accurately.

**TABLE OF CONTENTS**

**CHAPTER CHAPTER NAME PAGE NO**

**Chapter - 1 INTRODUCTION 1**

**Chapter - 2 LITERATURE SURVEY 3**

**Chapter - 3 PROBLEM STATEMENT 11**

**Chapter - 4 SYSTEM REQUIREMENTS SPECIFICATION 12**

4.1 PURPOSE 12

4.2 SCOPE 12

4.3 EXISTING SYSTEM 12

4.4 PROPOSED SYSTEM 14

4.5 REQUIREMENT ANALYSIS 14

**Chapter - 5 METHODOLOGY 16**

5.1 SYSTEM ARCHITECTURE 16

5.2 DATASET 18

5.3 MACHINE LEARNING ALGORIHTMS 19

5.4 EVALUATION METRICS 35

**Chapter - 6 IMPLEMENTATION 37**

6.1 DATA EXTRACTION 37

6.2 DATA PREPROCESSING 37

6.3 TRAINING THE MODELS 41

6.4 MODEL EVALUATION 43

6.5 WQI PREDICTION 43

**Chapter - 7 RESULT ANALYSIS 44**

**Chapter - 8 CONCLUSION 52**

**APPENDIX(A) SAMPLE CODE 53**

**APPENDIX(B) UML DESIGN 63**

**REFERENCES 67**

# **LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **TABLE.NO** | **NAME OF THE TABLE** | **PAGE NO** |
| 4.1 | Categorization of water quality | 13 |
| 6.1 | Relative weights of water quality parameters | 39 |
| 7.1 | Results of Evaluation metrics | 44 |

# **LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Fig.NO** | **NAME OF THE TABLE** | **PAGE NO** |
| 5.1 | System architecture | 16 |
| 5.2 | Architecture of Linear regression | 20 |
| 5.3 | Architecture of Polynomial regression | 22 |
| 5.4 | Architecture of Support vector regression | 25 |
| 5.5 | Architecture of Lasso regression | 27 |
| 5.6 | Architecture of Ridge regression | 30 |
| 5.7 | Architecture of Elastic Net regression | 32 |
| 7.1 | Validation curve of Linear regression | 45 |
| 7.2 | Validation curve of Support vector regression | 45 |
| 7.3 | Validation curve of Ridge regression | 46 |
| 7.4 | Validation curve of LASSO regression | 46 |
| 7.5 | Validation curve of Elastic Net regression | 47 |
| 7.6 | Model Vs R2 Score | 47 |
| 7.7 | Model Vs MSE | 48 |
| 7.8 | Model Vs MAE | 48 |
| 7.9 | Model Vs EVS | 49 |
| 7.10 | Model Vs Max Error | 49 |

**Chapter 1**

# **INTRODUCTION**

Water pollution is a significant environmental problem that affects the health and well-being of humans, animals, and the ecosystem. Water pollution occurs when harmful substances contaminate water bodies, degrading water quality and rendering it extremely poisonous to humans jeopardizing our health. According to the United Nations, around 80% of wastewater is discharged into the environment without treatment, contributing to water pollution. The death caused by unsafe water each year is greater than war and all other forms of violence combined.

Meanwhile, 2.5% of the Earth's water is freshwater, meaning the concentration of dissolved salts is less. Of this freshwater, about 68.7% is locked up in ice and glaciers, while another 30.1% is stored underground in aquifers. Only about 0.3% of the Earth's freshwater is readily available in rivers, lakes, and streams for human use.

The United Nations predicts that by 2050, the world's population will reach 9.7 billion, with two-thirds of people living in cities which may put even greater pressure on already limited freshwater resources, leading to water scarcity, pollution, and other environmental and social challenges. Action must be taken now to manage water resources sustainably, promote conservation and efficiency, and develop new technologies to predict the contamination of pollutants in water bodies to take further action.

The developed and developing countries are more focused on urbanization but water resources being contaminated due to numerous industrial and human activities are blinded yet the after effects are transparent. For the durable existence of mankind, both present and upcoming generations, it is the responsibility of countries worldwide to conserve water bodies and maintain efficient WQI. According to the UN, waterborne diseases cause the death of more than 1.5 million people every year, much greater than deaths caused by accidents, crimes, and terrorism combined. Without a proper plan, urban development in India causes frequent environmental problems or increases human waste and activity, polluting the environment. A proper plan is undoubtedly necessary after experiencing the effects of change in the environment because of the increase in human waste.

Central Pollution Control Board (CPCB) in association with State Pollution Control Boards (SPCBs) in the States and Pollution Control Committees (PCCs) has set up a National Water Quality Monitoring Network (NWMP) to access the quality of water of various water resources and suppress the pollution in water bodies. The government of India has released the Water Quality Affected Data from 2003 to 2021.

The deterioration of water resources is mainly due to contamination. It is very significant to forecast the water quality of upcoming years with the help of the existing data by analyzing the water quality of previous years. The goal of this study is to develop efficient machine learning models to predict the water quality index of upcoming years based on the collected data.

Predicting the level of water pollution can help authorities take timely actions to prevent or mitigate the impact of pollution. Machine learning (ML) can be used to develop models that predict the level of water pollution based on various parameters. The goal of a water pollution prediction project in ML is to develop a model that can predict the level of pollution in a water body based on historical data.

Overall, a water pollution prediction project in ML can help authorities take timely actions to prevent or mitigate the impact of pollution, which can improve the health and well-being of humans, animals, and the environment. Water pollution prediction is an important problem that can be addressed using machine learning (ML) techniques.

**Chapter 2**

# **LITERATURE SURVEY**

The degradation of water quality has alerted researchers and subject matter experts. As a result, several studies and research developments were conducted over years to improve the quality of water. It is essential to scrutinize the previous works in order to gain a better understanding of the current problem and its future development. The survey of past research works is summarized as follows:

Kosha.A.shah, et al., proposed “Evaluation of water quality index for river Sabarmati, Gujarat, India”. an approach to calculating the Water Quality Index (WQI) by utilizing six water quality parameters, namely D.O, pH, B.O.D, electrical conductivity, nitrate nitrogen, and total coliform. These parameters were measured at three stations from 2005 to 2008 along with the Sabarmati river basin. The evaluation of WQI in this study primarily employs the weighted arithmetic mean method, which has the advantage of incorporating data from multiple water quality parameters into a mathematical equation. This equation rates the health of the water body with a number. The study's findings reveal a degradation in water quality as the river flows from rural to urbanized areas [1].

Amir Hamzeh Haghiabi et al., “Water quality prediction using machine learning methods.” proposes the artificial intelligence techniques for estimating the parameters of water quality of Tireh River located in the south-west of Iran. The techniques used in this paper are Artificial Neural network (ANN), Group Method of data handling (GMDH) and Support vector machine (SVM). This study was carried out by using the features such as pH, SO4, Ca, Na, Cl, Mg, HCO3 etc., The results indicated that SVM shows best performance, accuracy of ANN is acceptable for practical purposes and the lowest accuracy of models was related to GMDH [2].

Fitore Muharemi, et al., “Machine learning approaches for anomaly detection of water quality on a real-world data set”. proposed the ML approaches that can be used for anomaly detection of Water quality. Several ML models were applied to water quality data, including Logistic Regression, Linear Discriminant Analysis, Support Vector Machines, Artificial Neural Network, Deep Neural Network (DNN), Recurrent Neural Network (RNN), and Long Short-Term Memory (LSTM) to accurately detect significant changes in water quality. The study findings indicated that all the algorithms were susceptible to vulnerabilities, with SVM, ANN, and logistic regression models showing slightly less vulnerability, while DNN, RNN, and LSTM models were found to be highly vulnerable [3].

Sayed Babak Haji Seyed Asodollah, et al., proposed “River water quality index prediction and uncertainty analysis: A comparative study of machine learning models.” the study that examines the effectiveness of three machine learning models in predicting the Water Quality Index (WQI) of the Lam Tsuen River in Hong Kong using water quality parameters measured on a monthly basis. The models include Support Vector Regression (SVR), Decision Tree Regression (DTR), and a novel ensemble Extra Tree Regression (ETR) algorithm. The study considers several input parameter combinations. The findings suggest that ETR using all ten inputs produces more precise results compared to SVR and DTR [4].

Tianan Deng et al., said “Machine learning based marine water quality prediction for coastal hydro-environment management” in which Support vector machine (SVM) and artificial neural network (ANN) are used to predict the probable HAB (Harmful algal blooms). The study's findings and results show that SVM performs better than all ANN models at predicting water quality, but at the expense of computational efficiency since it considers nonlinear correlations between inputs and outputs [5].

Hongfang Lu et al., proposed “Hybrid decision tree-based machine learning models for short-term water quality prediction”in which two hybrid models that utilize extreme gradient boosting (XGBoost) and random forest (RF) as basic models to predict six water quality indicators: water temperature, dissolved oxygen, pH value, specific conductance, turbidity, and fluorescent dissolved organic matter. The models incorporate a data denoising technique called complete ensemble empirical mode decomposition with adaptive noise (CEEMDAN). The study found that CEEMDAN-RF produces the most accurate predictions for temperature, dissolved oxygen, and specific conductance, while CEEMDAN-XGBoost performs best in predicting pH value, turbidity, and FDOM [6].

Donya Dezfooli, et al., proposed “Classification of water quality status based on minimum quality parameters” in which three models, namely probabilistic neural network (PNN), k-nearest neighbor, and support vector machine (SVM), for classifying the water quality of the Karoon River using three assessment criteria: error rate, error value, and accuracy. The study evaluated the performance of these models by removing some of the water quality assessment parameters. The results showed that all three models produced similar outcomes when no parameter was removed. However, when some quality factors were removed, PNN was found to be the best model, achieving 90.70% accuracy, 9.30% error rate, and an error value of 4, by using only turbidity, faecal coliform, and total solids to classify water quality [7].

Ali Najah Ahmeda, et al., proposed “Machine learning methods for better water quality prediction” Adaptive Neuro-Fuzzy Inference System (ANFIS), Radial Basis Function Neural Networks (RBF-ANN), and Multi-Layer Perceptron Neural Networks (MLP-ANN) according to historical water-quality parametric data. In order to evaluate the impacts on the model, three evaluation techniques or assessment processes have been used. The first assessment process is dependent on the partitioning of the neural network connection weights that ascertains the significance of every input parameter in the network. On the other hand, the second and third assessment processes ascertain the most effectual input that has the potential to construct the models using a single and a combination of parameters, respectively. During these processes, two scenarios were introduced. Scenario 1 constructs a prediction model for water quality parameters at every station, while Scenario 2 develops a prediction model on the basis of the value of the same parameter at the previous station (upstream). The results for Scenario 2 was observed to be more adequate than Scenario 1 [8].

Duie Tien Bui, et al., said “Improving prediction of water quality indices using novel hybrid machine-learning algorithms” the effectiveness of four standalone (RF, M5P, RT, and REPT) and 12 hybrid data-mining algorithms (hybrids of the standalones with bagging, CVPS, and RFC) for predicting monthly WQI in a humid environment in northern Iran. The modeling process revealed that fecal coliform concentration was the most important determinant of WQI. This was followed in order of importance by BOD, NO− 3, DO, EC, COD, PO2− 4, Turbidity, TS, and pH [9].

Yafra Khan, et al., proposed “Predicting and Analyzing Water Quality using Machine Learning: A Comprehensive Model” which presents a water quality prediction model that uses Artificial Neural Network (ANN) and time-series analysis to predict water quality factors. The study utilized water quality historical data from 2014, recorded at 6-minute intervals. The performance of the model was evaluated using Mean-Squared Error (MSE), Root Mean-Squared Error (RMSE), and Regression Analysis. The proposed model, which included ANN-NAR, was found to be reliable, producing improved prediction accuracy. The study revealed that the proposed model had the lowest MSE of 3.7x10-4 for turbidity and the best Regression value of 0.99 for Specific Conductance [10].

Kangyang Chen, et al., said “Comparative analysis of surface water quality prediction performance and identification of key water parameters using different machine learning models based on big data” to compare the water quality prediction performance of 10 learning models (7 traditional and 3 ensemble models) using big data (33,612 observations) from the major rivers and lakes in China from 2012 to 2018, based on the precision, recall, F1-score, weighted F1-score, and explore the potential key water parameters for future model prediction. Results showed that the bigger data could improve the performance of learning models in prediction of water quality [11].

Md Galal Uddin, et al., proposed “Robust machine learning algorithms for predicting coastal water quality index”. This study examined the performance of eight commonly used algorithms, including Random Forest (RF), Decision Tree (DT), K-Nearest Neighbor (KNN), Extreme Gradient Boosting (XGBoosting), Extra Tree, Support Vector Machine (SVM), Linear Regression, and Gaussian Naïve Bayes, in predicting the coastal water quality index (WQI) at Cork Harbor. The evaluation of model performance was based on several metrics such as Root Mean Squared Error (RMSE), Mean Squared Error (MSE), Mean Absolute Error (MAE), R-squared (R2), and Percentage Relative Error Index (PREI). The results suggested that the DT, Extra Tree, and XGB models are effective, robust, and able to significantly reduce model uncertainty in predicting WQIs. The findings of this study can aid in reducing model uncertainty and optimizing the WQM-WQI model architecture for predicting WQI values, with a 95% confidence interval for the predicted values. However, it is important to note that this study only compared eight general algorithms, and the results are more applicable to coastal regions [12].

Furqan Rustam, et al., proposed “An artificial Neural network model for water quality and water consumption prediction”. This study introduces a neural network architecture that is simple yet effective in predicting both water quality and water consumption. To achieve this, the study employs several machine learning algorithms, such as Decision Tree, Random Forest, Logistic Regression, Support Vector Classifier, AdaBoost, and other deep learning models. The study uses performance metrics such as accuracy, precision, recall, and F1 Score to evaluate the models. Two datasets are used to test the proposed approach. The results show that the proposed approach achieves 0.96 accuracy in water quality prediction, outperforming existing studies. Additionally, a 0.99 R² score is obtained for water consumption prediction, which outperforms state-of-the-art approaches [13].

Gaganjot kaur kang, et al., proposed “Data-driven Water Quality Analysis and Prediction: A Survey”. This paper examines the current state of research on water quality assessment and prediction, and compares the various big data analytics approaches and prediction models used in this field. Among the models discussed are Pearson's coefficient Analysis, Fast fuzzy C-mean clustering, GIS, and Antcolony algorithm, while the machine learning models analyzed include ANN, RBFN (radial basis function network), Decision tree, and Improved decision tree. Results indicate that ANN has the lowest MSE (3.7\*10-4) and outperforms other models in terms of prediction accuracy [14].

Sani Isah Abba, et al., proposed “Implementation of data intelligence models coupled with ensemble machine learning for prediction of water quality index” in which ,back propagation neural network (BPNN) and adaptive neuro-fuzzy inference system (ANFIS), support vector regression (SVR), and one multilinear regression (MLR) are considered for the prediction of water quality index (WQI) at three stations. The WQ parameters including dissolved oxygen (DO), pH, biological oxygen demand (BOD), ammonia (NH3), temperature (T), and WQI were obtained from the Central Pollution Control Board (CPCB). The findings showed that the developed data intelligence models were capable of predicting the WQI at the three stations, with NNE providing the best modelling results. The minimum values for root mean square (RMS) were found to range between 0.1213 and 0.4107, 0.003 and 0.0367, and 0.002 and 0.0272 for Nizamuddin, Palla, and Udi (Chambal), respectively. In particular, ANFIS-M3, BPNN-M4, and BPNN-M3 were found to enhance performance with respect to absolute error by 41%, 4%, and 3%, respectively, compared to other models for Nizamuddin, Palla, and Udi (Chambal) stations [15].

Isah Abba, et al., proposed “Estimation of water quality index using artificial intelligence” In this study, Artificial Intelligence techniques and a Multi Linear Regression model were used to estimate the Water Quality Index (WQI). The accuracy and performance of the models were evaluated using performance measures such as Mean Square Error, Root Mean Squared Error and Determination Coefficient. Three models, namely ANN, ANFIS, and MLR, were proposed for the estimation of the WQI of the river. The results showed that the artificial intelligence-based models (ANN and ANFIS) performed better than the conventional MLR model, with up to a 10% improvement in the verification phase. Although the performance of ANN was slightly better than ANFIS, both models outperformed MLR in estimating the WQI [16].

Luka Grbčić et al., proposed “Coastal water quality prediction based on machine learning with feature interpretation and spatial temporal analysis”, which focuses on maintaining a routine monitoring of E.Coli in 15 beaches. This study involves the analysis of the temporal data in predicting the E.Coli bacteria in coastal water.The findings show that Catboost algorithm performs best to predict E.Coli with an R2  value of 0.71 and 0.69 [17].

Wang.X et al., proposed “Evaluation of water quality based on a machine learning water quality index for the Ebinur lake watershed, China” to identify the threats to water quality. The study proposes a model for estimating and assessing the Water Quality Index (WQI) by combining a machine learning algorithm, remote sensing spectral indices (difference index, DI; ratio index, RI; and normalized difference index, NDI), and fractional derivatives methods. To evaluate the accuracy of the model, the study uses various performance measures such as R2, RMSE, SD, RPD, and geographically weighted regression (GWR) model [18].

Ali Omran Al-Sulttani et al., “Proposition of new ensemble Data-Intelligence models for surface Water Quality Prediction”. This study was aimed to develop and compare five different ensemble machine learning (ML) models for predicting monthly biochemical oxygen demand (BOD) values of the Euphrates River, Iraq. The models included Quantile Regression Forest (QRF), Random Forest (RF), Radial Support Vector Machine (SVM), Stochastic Gradient Boosting (GBM), and Gradient Boosting Machines (GBM\_H2O). Results indicated that the QRF model outperformed the other models. Moreover, the developed integrative PCA-QRF model exhibited a much better performance compared to standalone models and those integrated with GA. These findings suggest that the QRF model may be useful for predicting BOD values in the Euphrates River, Iraq [19].

Charmaine Jerome et al., proposed “Evaluation of water quality index and its impact on the quality of life in an industrial area in Bangalore, South India”, which proposes a water quality index to determine the suitability of groundwater for drinking purposes. The Tiwari and Mishra (1985) index was utilized to calculate the water quality index (WQI) in this study. Nine water quality parameters, including pH, alkalinity, total dissolved solids (TDS), total hardness, calcium (Ca) hardness, magnesium (Mg) hardness, nitrate, chloride, and sulfate, were taken into consideration for WQI computation. The weight of each parameter, denoted by Wi, was determined based on its weightage using the formula WQI = (Ʃ qi Wi) / (ƩWi). The relationship between WQI and quality of life was investigated using Pearson correlation [20].

Md.Galal Uddin et al., “A review of water quality index models and their use for assessing surface water quality”. proposes a comparative study of the most commonly used WQI models, including the different model components, structures and applications. It mainly focuses on parameterization of the WQI models, the techniques used to decide on the sub-indices, parameter weighting values, index aggregation functions and the sources of uncertainty. Twenty-one different WQI models were identified and reviewed in this study. Some of the models are CCME (Canadian Council of Ministers of the Environment) WQI, NSF(National sanitation foundation)WQI, Horton WQI, SRDD-WQI,BCWQI etc., The results show that 82% of the water quality models applications have been used to assess river water quality [21].

Tingting Xu et al., proposed "A predictive model of recreational water quality based on adaptive synthetic sampling algorithms and machine learning" to help resolve the issue of unbalanced datasets in water quality models. This paper provides a comprehensive analysis of the development and validation of a machine learning-based predictive model for recreational water quality, with a focus on the use of adaptive synthetic sampling techniques to improve the accuracy and reliability of the model. This paper utilizes four machine learning techniques: k-mean nearest neighbour, boosting decision trees, support vector machines, and multilayer perceptron artificial neural networks. The models except support vector machine all provide good predictions with relatively high sensitivity (around 75%) and accuracy (over 90%), suggesting that more sophisticated model training that involves artificial data can be used to predict both compliance and exceedance conditions more effectively [22].

Vinod Kothari et al., proposed "Correlation of various water quality parameters and water quality index of districts of Uttarakhand" which investigates the association between various water quality parameters and the water quality index (WQI) in Uttarakhand, India. The study scrutinizes the water quality data of 24 sites located across the state. It encompasses parameters such as pH, electrical conductivity, turbidity, total dissolved solids, and several metals and ions. The findings of the study reveal that there is a positive correlation between the WQI and pH, calcium, and magnesium. In contrast, there is a negative correlation between the WQI and turbidity, total dissolved solids, chloride, and sulfate. The study emphasizes the importance of continuous monitoring and assessment of water quality conditions using multiple parameters and indices. This will ensure effective management of water resources in the area[23].

V.Geethanjali et al., "Indian Water Pollution Monitoring and Forecasting for anomaly with Fail-Safe Wireless Sensor Networks using Machine learning techniques" proposes linear regression algorithm to predict the annual water quality index on the Indian Water quality data. This study suggests weighted arithmetic mean model to estimate the water quality index. The results shows the accuracy of the linear regression model as 0.81 [24].

Md. Mehedi Hassan et al., "Efficient Prediction of Water Quality Index (WQI) using Machine Learning Algorithms" proposes the techniques that intends to categorize a dataset of water quality in various places of India based on the several parameters like dissolved oxygen (DO), total coliform (TC), biological oxygen demand (BOD), Nitrate, pH, and electric conductivity (EC) based upon several machine learning techniques such as RF, NN, MLR, SVM, and BTM. The findings of this paper showed that Nitrate, PH, conductivity, DO, TC, and BOD are the key qualities that contribute to the orderly classification of water quality [25].

Umair Ahmed et al., proposed "Eﬃcient Water Quality Prediction Using Supervised Machine Learning”. This paper explores a series of supervised machine learning algorithms to approximate the WQI and the WQC. The proposed methodology of this paper employs four input parameters namely pH, temperature, total dissolved solids , and turbidity. The results of this study indicates that gradient boosting and polynomial regression predict the WQI most efficiently whereas multi-layer perceptron (MLP) classifies the WQC most efficiently [26].

The review of previous works shows that improvement in water quality can be done by predicting water quality and considering preventive measures in case of degradation. It can be seen that water quality monitoring and prediction is done by using several machine learning models. A few models are Extra tree regression, Gradient boosting, artificial neural networks, and Random Forest, etc., It can also be observed that several water quality indices are developed to find the composite effect of the water parameters on the quality of water.

**Chapter 3**

# **PROBLEM STATEMENT**

The quality of the water can be determined using the water quality index. Water quality index is a numerical value which exhibits the composite effect of the water quality parameters. It is essential to forecast the water quality index to mitigate the water quality issues. This study aims to estimate the Water quality index (WQI) using the weighted arithmetic mean method for the given years in the dataset and predict the water quality index in advance by using different machine learning algorithms. The predicted values can be used to analyze the increase or decrease of WQI and take necessary measures to prevent the deteriorating nature of water quality.

**Chapter 4**

# **SYSTEM REQUIREMENTS SPECIFICATION**

**4.1 PURPOSE**

It is essential to ensure that the water is portable for human consumption to prevent harm to human and environmental health. The purpose of this model is to anticipate the changes in water quality over years, allowing the required measures to be taken to prevent pernicious effects. ML models are used to analyze the water quality using a range of variables such as pH, temperature, BOD etc., and make predictions of the water quality index in the following years.

**4.2 SCOPE**

It is estimated that every day almost 40 million liters of wastewater goes into rivers and other water bodies. The water bodies in India are polluted due to Urbanization and it is true that people have died due to consuming contaminated water in India.

It is estimated that millions of Indian people suffer from waterborne diseases every year. Poor water quality can lead to a vast number of deadly waterborne diseases such as cholera, dysentery, and typhoid. These diseases can be fatal and can cause severe illness that can be life-threatening for vulnerable populations such as young, elderly and weak immune people. It is alarming that it is essential to make sound decisions on managing water quality in the future. The prediction of the water quality index of upcoming years can be used to alert us on ongoing, emerging problems and to determine the standards of water which helps in anticipating if the water is portable.

Therefore, this project provides the scope to take precautions to avoid poor water quality by predicting WQI for the upcoming years

**4.3 EXISTING SYSTEM**

In existing system, the water quality index is evaluated using standard WQI calculation method in which the numerical value is then multiplied by a weighting factor that is relative to the significance of the test to water quality. The existing system categorizes the water quality based on the WQI as shown in Table 4.1:

|  |  |
| --- | --- |
| Value of WQI | Quality of Water |
| 90-100 | Excellent |
| 70-90 | Good |
| 50-70 | Medium |
| 25-50 | Bad |
| 0-25 | Very bad |

**Table 4.1 Categorization of water quality**

Several models like Random Forest, Decision Tree, Linear Regression, Multi-layer perceptron, LSTM, Artificial Neural Networks, and other deep learning models are used in the existing system. The water quality index is estimated using pH, Dissolved Oxygen, TDS, Hardness, B.O.D etc., After the calculation of the WQI, several models are applied on the dataset to predict the WQI. Some of the existing models also classify the Quality of water based on the WQI.

**4.3.1 Existing system disadvantages:**

The disadvantages identified in the existing system are:

* Water quality index is predicted by taking several features as input.
* It requires pre-processing steps like feature selection to select the important features.
* Some of the existing models have high computational speed.
* Existing systems does not forecast the WQI value. They either predict the WQI with many features as independent variables or classify the water nature as potable or not.

**4.4 PROPOSED SYSTEM**

The existing systems consists of prediction of the water quality index using several water quality parameters and classification of water quality but not the forecasting of water quality index. CPCB initiatives mostly work on the monitoring of water. There must be a system that predicts the water quality in advance to take any water quality deterioration prevention measures. The proposed model overcomes this issue of forecasting the annual water quality index. The prediction of WQI in advance is done by evaluating the WQI using the weighted arithmetic mean method at several water monitoring stations in India and by using the machine learning algorithms.

**4.5 REQUIREMENTS ANALYSIS:**

**4.5.1 Functional requirements**

* Choosing the dataset consisting of required water quality parameters for WQI.
* Pre-processing the data by data type conversion, filling missing values and feature construction.
* Calculating WQI under feature construction step.
* Splitting the data set into a training set and testing set.
* Train the data set using different ML algorithms to make a model. Checking the efficiency of the models on the testing set.
* Start predicting the WQI of particular year with best model obtained.

**4.5.2 Non-functional requirements**

* **Availability:** The application for predicting the water quality index for an year will be available in all the systems that have python installed in them.
* **Performance**: Performance is calculated in terms of the prediction of the model. Accuracy is used to measure the performance of the model.

**4.5.3 Software requirements**

* **Python**: Python is a deciphered, significant level, broadly useful programming language.
* **Flask:** Flask is a a web application framework written in python, in simple terms it helps end users interact with your python code (in this case our ML models) directly from their web browser without needing any libraries, code files.
* **Jupyter Notebook:** Jupyter Notebook is a web-based interactive computing platform. The notebook combines live code, equations, narrative text, visualizations.

**4.5.4 Hardware requirements**

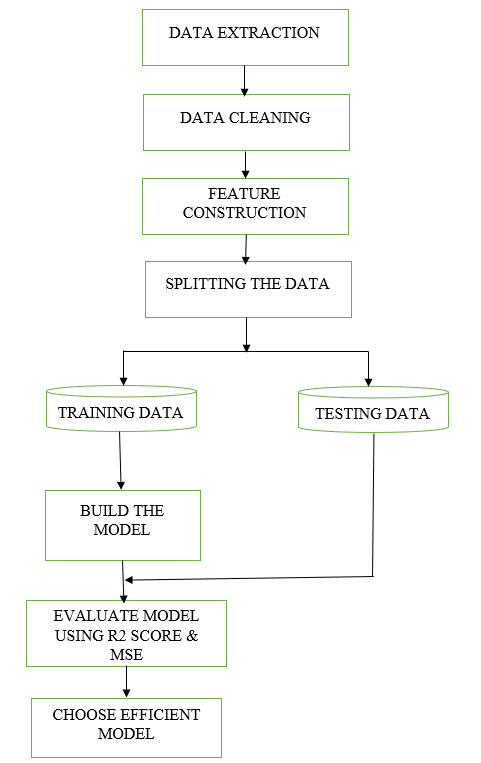
1. RAM: 4GB/8GB/16GB
2. Processor: Intel Core i3/i5/i9
3. Hard Disk: 500GB/1TB/2TB
4. OS: Windows 7 or higher
5. High speed internet

**Chapter 5**

# **METHODOLOGY**

**5.1 SYSTEM ARCHITECTURE**

The flow chart that demonstrates the architecture of proposed system is shown in Fig 5.1:

****

**Fig 5.1 System architecture**

**System architecture Description:**

1. Data Extraction: This step involves collection of data from CPCB website. The data is in the form of PDF’s for each year which are to be downloaded. The collected data is in a messy format with different column names. This data is to be converted into excel format and it is necessary to ensure that the columns are identical before merging the data with the data from Kaggle.
2. Data Cleaning: The collected data contains the values in object format which are to be converted into floating values before checking for the null values. The parameters of the water change according to the locations. Hence, filling the null values with central tendency measures may not provide accurate results. The MICE Forest algorithm is used to fill the null values.
3. Feature Construction: This step includes constructing the features year and water quality index. The water quality index is calculated using the weighted arithmetic mean which gives the composite effect of the parameters in the water. The relative weights and sub-index values required for evaluating the water quality index are taken according to the CPCB standards.
4. Splitting the data: Splitting the data is crucial in supervised learning technique. The data can be divided into training and testing data. The most frequently used splitting ratio are 8:2 or 7:3. The training data is used to train and build a machine learning model. The splitting of the data can be done by utilizing train\_test\_split() method which takes the dependent and independent variables and the test\_size as attributes and returns 4 parameters.
5. Build the Model: Model is built using the training data and the machine learning algorithms. The algorithms used are Linear regression, Polynomial regression, Support vector regression, LASSO regression, Ridge regression and Elastic net regression.
6. Evaluating Model: Evaluating model includes testing the model’s results on the testing data and comparing the results to the actual values. The efficiency of the model can be checked by using the evaluation metrics. R2 score and Mean squared error (MSE) are used to evaluate the model. R2 score indicates how accurate the model can perform and the mean squared error shows the error percentage between the observed and expected values of the data.
7. Choose efficient model: The efficient model is considered to build the website and predict the water quality index value.

Forecasting the water quality index helps in identifying the measures that need to be taken to prevent deterioration of the water. Several machine learning algorithms can be developed to mitigate the issue of water quality deterioration by predicting the annual water quality index.

**5.2 DATASET**

The dataset consists of different features. They are Station code, Location, State, pH, Dissolved oxygen, B.O.D, Conductivity, Temperature, Fecal Coliform, Total coliform, Nitrate and Nitrite.

The feature called Year is added to the data extracted from Kaggle and Central Pollution control board. Another feature called water quality index is evaluated using the weighted arithmetic mean method.

**5.2.1 Feature definitions**

* pH: It is used to define how acidic or basic the solution (water) is.
* Dissolved oxygen (D.O): It represents the amount of oxygen dissolved in the water.
* Biochemical oxygen demand (B.O.D): It is a measure of the amount of D.O required to remove waste organic matter from water in the process of decomposition by aerobic bacteria.
* Electrical Conductivity: Degree to which the water can conduct electricity.
* Total coliform: Total coliforms include bacteria that are found in water that has been influenced by other wastes.
* Nitrate: Nitrate is a nitrogen oxo anion formed by loss of a proton from nitric acid.

**5.2.2 Water Quality Index**

Water quality index represents the quality of the water in the form of a numerical value for any intended purpose. WQI is calculated to show the composite influence of the parameters that were considered while evaluating the water quality index. The parameters that can be taken into account are pH, Dissolved oxygen, Electrical conductivity, Biochemical oxygen demand, Total coliform, Nitrate and nitrite, etc.,

Several techniques were introduced by the experts to calculate the water quality index. A few of them include the Oregon technique, the Horton Index method, the National Sanitation foundation method, the Canadian Council of Ministers of the Environment method, and the Weighted arithmetic method. The choice of method for water quality index evaluation depends upon the location and also the number of parameters considered. In India, weighted arithmetic mean method is mostly in use as it reflects the influence of a greater number of water quality parameters when compared to other calculation methods.

The formula for calculating the weighted arithmetic mean water quality index is:

WQI) / )

where,

N is number of considered parameters

is sub-index value of a parameter

is relative weight of the parameter

**5.3 MACHINE LEARNING ALGORITHMS**

**5.3.1 Linear regression:**

Linear regression is a machine learning algorithm that estimates how a model follows the linear relationship between the dependent and independent variables. It fits a linear model to minimize the residual sum of squares between the targets in the dataset, and the targets predicted by the linear approximation.

Linear regression can be mathematically represented as:

where:

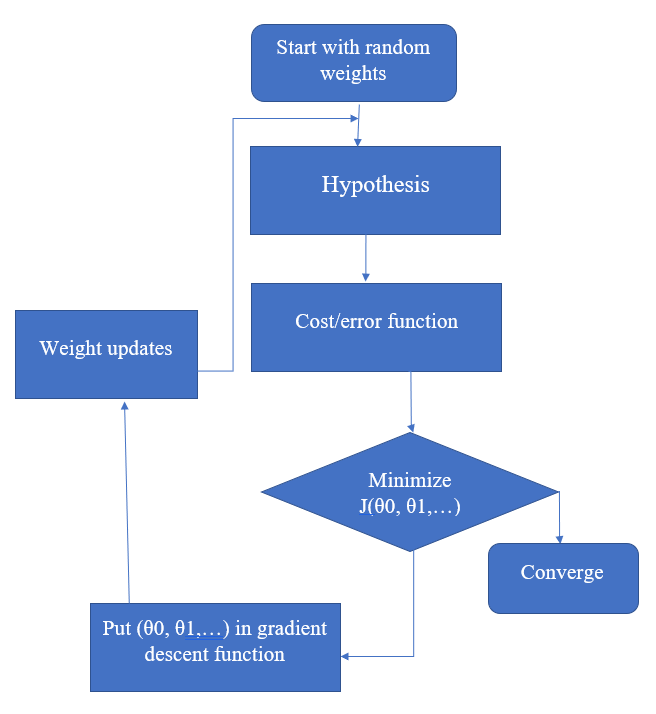
yi: Dependent variable

β0: Intercept

β1: Slope

xi: Independent variable

εi: Random Error



**Fig 5.2 Architecture of linear regression**

**Working of Linear regression:**

* Data gathering: Data is collected for the independent variable(s) and dependent variable.
* Data Preprocessing: The data is preprocessed by removing any missing or erroneous values, normalizing the data if needed, and splitting it into training and testing sets.
* Parameter initialization: The coefficients and the bias term (intercept) are initialized to small random values.
* Defining the cost function: The cost function is defined, which measures the difference between the predicted output and the actual output. The cost function generally used is the Mean Squared Error (MSE).
* Defining the learning rate: A random learning rate is chosen to determine how quickly the algorithm learns. A small learning rate will result in slower convergence, while a large learning rate may cause overshooting.
* Model training: The model is trained using the training set to update the parameters using gradient descent. Gradient descent is an optimization algorithm that minimizes the cost function by iteratively adjusting the parameters in the direction of the negative gradient.
* Model evaluation: The performance of the model is evaluated using the testing set by calculating the MSE.
* Prediction: Once the model is trained, it can be used to make predictions on new data.

**Algorithm of Linear regression:**

**Step-1**: Import essential libraries.

**Step-2**: Read the data using the pandas read\_csv() or read\_excel().

**Step-3**: Pre-process the data.

**Step-4**: Split the data into training and testing set in either 8:2 ratio or 7:3 ratio. This can be done by using train\_test\_split().

**Step-5**: Fit the model with the training data by defining the essential parameters.

**Step-6**: Evaluate the efficiency of the model using the evaluation metrics and make required predictions.

**Methods used:**

*sklearn.linear\_model.LinearRegression(fit\_intercept,copy\_X,n\_jobs,positive)*

Here,

* *fit\_intercept* accepts boolean values which tells whether the intercept of the model should be calcilated or not.
* *copy\_X* accepts Boolean values and describes whether X value should be copied or overwritten.
* *n\_jobs* is an integer value which is used to set the number of jobs that can be used for computation.
* *positive* accepts boolean value and is used when the coefficients are required to be positive.

**5.3.2 Polynomial regression:**

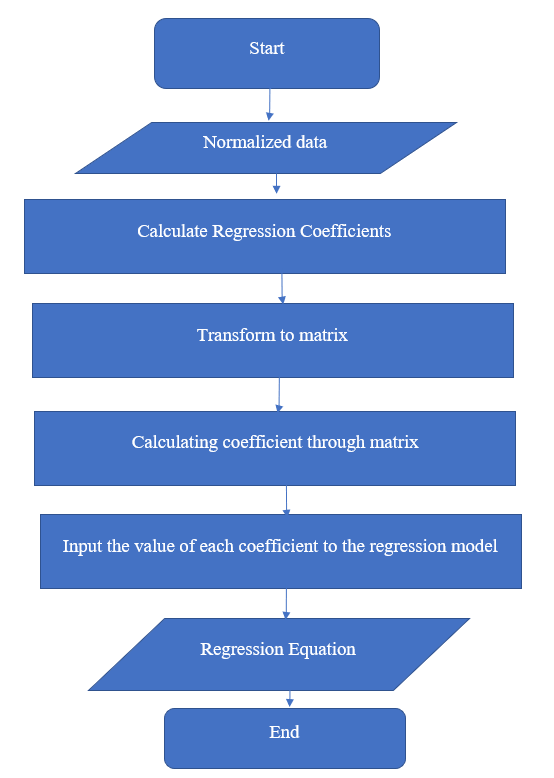
Polynomial regression resembles linear regression but the difference is that linear regression applies to linearly separable data while polynomial regression fits non-linear data. It models the relationship between the independent and dependent variables to an nth degree polynomial. The relationship between the independent and dependent variable, with the help of a polynomial equation, turns out as a curvilinear graph.

Mathematically polynomial regression can be represented as:

Where:

i=1,2,….n

k:level of the equation required



**Fig 5.3 Architecture of Polynomial regression**

**Working of Polynomial regression:**

* Data collection: Data is collected for the dependent and independent variables.
* Determining the degree of polynomial: The degree of the polynomial for the model is decided based on the complexity of the data and the level of accuracy required.
* Designing the matrix: The matrix of size n x m is designed where n is the number of data points and m is the degree of the polynomial plus one. The matrix contains the powers of x from 0 to m-1.
* Fitting the model: Linear regression is used to fit the polynomial model to the data which involves finding the coefficients that minimizes the sum of the squared errors between the predicted and actual values of dependent variable.
* Model evaluation: The model is evaluated by calculating the mean squared error (MSE) and R-squared value. The MSE measures the average squared difference between the predicted and actual values of y. The R-squared value indicates how well the model fits the data, with a value of 1 indicating a perfect fit.
* Predictions: The evaluated model can be used to predict new values of the dependent variable.

**Algorithm of polynomial regression:**

**Step-1**: Import essential libraries.

**Step-2**: Read the data and perform data pre-processing on the dataset.

**Step-3**: Split the data into training and testing sets in either 8:2 or 7:3 ratio. This can be done by using train\_test\_split().

**Step-4**: Built the polynomial equation by giving the required degree to the PolynomialFeatures class.

**Step-5**: Build the model by fitting the polynomial equation into the Linear regression class.

**Step-6**: Evaluate the model's efficiency using the evaluation metrics and make predictions.

**Methods used:**

*sklearn.preprocessing.PolynomialFeatures(degree,interaction\_only,include\_bias,order)*

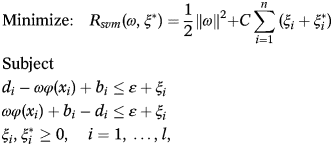
Here,

* *degree* is an integer value that describes the maximum degree the polynomial features accept.
* *interaction\_only* accepts boolean valuewhich produces interaction features if set true.
* *include\_bias* accepts boolean value which indicates whether bias can be included or not.
* *order* accepts ‘C’ or ‘F’ where F is faster to compute.

**5.3.3 Support vector machine:**

The support vector machine algorithm is a dynamic algorithm which can solve a wide range of problems such as linear and non-linear problems, binary, binomial along with classification and regression problems using SVM classifier and SVM regressor respectively. SVM uses the concept of margins and minimises the chances of model overfitting. The goal of this algorithm is to create a hyperplane which can categorise the new data point into the correct category by segregating n-dimensional space in classes.

Consider the following mathematical expression:



Here,

ω  is a normal vector.

 is the regularization factor.

C is the error penalty factor.

b is a bias.

ɛ is the error function.

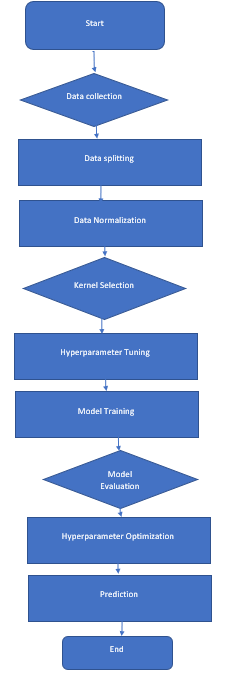
xi is the input vector.

di is the target value.

l is the number of elements in the training data set.

φ(xi) is a feature space.

 and  are upper and lower excess deviations.



**Fig 5.4 Architecture of Support vector regression**

**Working of Support vector regression:**

* Data Collection: Data is gathered for the dependent and the independent variables.
* Data Normalization: The data is normalized to ensure that all the features have the same scale.
* Kernel selection: A kernel function is chosen to map the data into a higher dimensional space.
* Hyperparameter selection: The values for the hyperparameters of the model are selected, including the regularization parameter C and the kernel function parameters.
* Creating SVR model: An SVR model is created using the chosen kernel function and hyperparameters.
* Model training: SVR model is trained using the training data.
* Model evaluation: The model performance of the model is evaluated using mean squared error or coefficient of determination.
* Predictions: Once the model is trained and evaluated, it can be used to make required predictions.

**Algorithm of support vector regression:**

**Step-1**: Import essential libraries.

**Step-2**: Read the data using read\_csv() or read\_excel().

**Step-3**: Pre-process the data by handling missing values, normalizing the features and splitting the data into training and testing sets.

**Step-4**: Select a kernel function. Most commonly used kernel functions are linear, polynomial, radial basis function and sigmoid.

**Step-5**: Train the model using the SVR(). SVR() takes many arguments, some of them are kernel, gamma etc.,

**Step-6**: Evaluate the model using the evaluation metrics and make required predictions.

**Methods used:**

*Sklearn.svm.SVR(kernel,degree,gamma,coef0,tol,C,epsilon,cache\_size,shrinking,verbose,max\_iter)*

Here,

* *kernel* accepts linear, poly, rbf, sigmoid, precomputed.
* *degree* accepts integer value and it must be positive.
* *gamma* accepts floating values or ‘scale’,’auto’.
* *coef0* is floating value and significant only for ‘poly’ and ‘sigmoid’.
* *tol* accepts float values which indicates the tolerance for stopping criterion.
* *max\_iter* is used to keep hard limit on iterations within solver.

**5.3.4 LASSO Regression:**

LASSO regression is a regularized model that attempts to minimize the loss function by adding a penalty term to it. The penalty term in lasso regression can be calculated by multiplying the lambda value by the absolute weight of the individual feature. LASSO regression performs L1 regularization and helps to reduce the underfitting of the data.

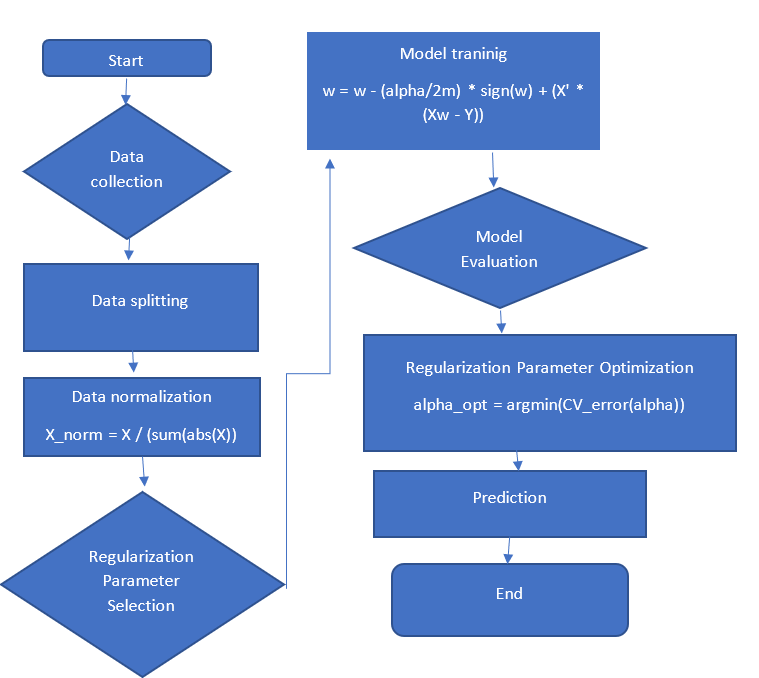
Cost function = Loss + λ \* Σ ||w||

Here,

Loss is sum of squared residual

λ is penalty

w is slope of the curve



**Fig 5.5 Architecture of LASSO regression**

**Working of LASSO regression:**

* Data Collection: Data is gathered for the dependent and the independent variables.
* Data Normalization: The data is normalized to ensure that all the features have the same scale.
* Regularization parameter selection: The value for the regularization parameter λ is selected. ‘λ’ controls the strength of the penalty term.
* Creating LASSO regression model: A LASSO regression model is created using the chosen λ value.
* Model training: LASSO model is trained using the training data.
* Model evaluation: The model performance of the model is evaluated using mean squared error or coefficient of determination.
* Predictions: Once the model is trained and evaluated, it can be used to make required predictions.

**Algorithm of LASSO regression:**

**Step-1**: Import essential libraries.

**Step-2**: Load the data using functions in pandas library and analyse the data.

**Step-3**: Pre-process the data by handling missing values, normalizing the features and splitting the data into training and testing sets.

**Step-4**: Fit the model on the training data by using LassoCV().

**Step-5**: Evaluate the efficiency of the model by using the evaluation metrics like MAE, RMSE,R2 score etc.,

**Methods used:**

*sklearn.linear\_model.LassoCV(eps,n\_alphas,alphas,fit\_intercept,precompute,mmax\_iter,tol,copy\_X,cv,verbose,n\_jobs,positive,random\_state,selection)*

Here,

* *eps* indicates the length of the path.
* *n\_alphas* accepts integer values and indicates numberof alphas along regularization path.
* *alphas* is a list of alphas where to compute the models.
* *fit\_intercept* accepts boolean values which tells whether the intercept of the model should be calcilated or not.
* *cv* accepts integer values and determines the cross-validation splitting strategy.
* *copy\_X* accepts Boolean values and describes whether X value should be copied or overwritten.
* *n\_jobs* is an integer value which is used to set the number of jobs that can be used for computation.
* *positive* accepts boolean value and is used when the coefficients are required to be positive.
* *max\_iter* is the maximum number of iterations.
* *tol* accepts floating value and indicates the tolerance for the optimization.
* *verbose* is the amount of verbosity.
* *selection* is set to either ‘random’ or ‘cyclic’.

**5.3.5 Ridge Regression:**

Ridge regression also known as L2 regularization is a regularized model that attempts to minimize the loss function by adding a penalty term to it. The penalty term in ridge regression can be calculated by multiplying the lambda value by the squared weight of the individual feature.Ridge regression prevents the overfitting of the model.

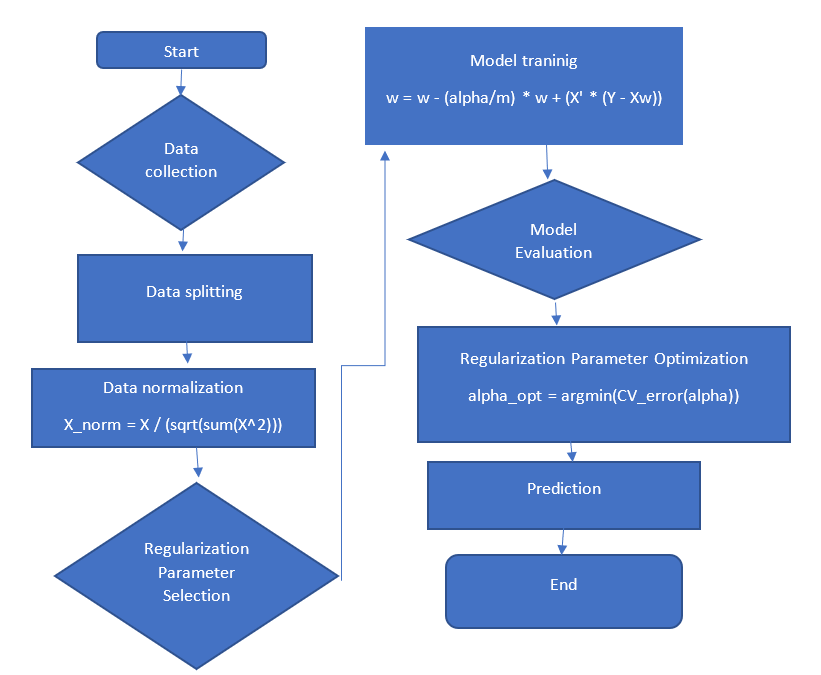
Cost function = Loss + λ \* Σ ||w||2

Here,

Loss is sum of squared residual

λ is penalty

w is slope of the curve



**Fig 5.6 Architecture of Ridge regression**

**Working of Ridge regression:**

* Data Collection: Data is gathered for the dependent and the independent variables.
* Data Normalization: The data is normalized to ensure that all the features have the same scale.
* Regularization parameter selection: The value for the regularization parameter λ is selected. ‘λ’ controls the strength of the penalty term.
* Creating Ridge regression model: A ridge regression model is created using the chosen λ value.
* Model training: Ridge model is trained using the training data.
* Model evaluation: The model performance of the model is evaluated using mean squared error or coefficient of determination.
* Predictions: Once the model is trained and evaluated, it can be used to make required predictions.

**Algorithm of ridge regression:**

**Step-1**: Import essential libraries.

**Step-2**: Load the data using read\_csv() or read\_excel() functions.

**Step-3**: Pre-process the data by handling missing values, normalizing the features and splitting the data into training and testing sets and split it into train and test sets in either 8:2 ratio or 7:3 ratio.

**Step-4**: Fit the model using the RidgeCV() function from sklearn.

**Step-5**: Evaluate the efficiency of the model by using the evaluation metrics like MAE, RMSE,R2 score etc., and use the trained model to make predictions.

**Methods used:**

*sklearn.linear\_model.RidgCVe(alphas,fit\_intercept*, *scoring*, *cv*, *gcv\_mode*, *store\_cv\_values*, *alpha\_per\_target)*

Here,

* *alpha* accepts float and ndarrays which is used to control the regularization strength. It is the constant that multiplies the L2 term.
* *fit\_intercept* accepts boolean values which tells whether the intercept of the model should be calcilated or not.
* *scoring* takes string values. If None, the negative mean squared error if cv is ‘auto’ or None (i.e. when using leave-one-out cross-validation), and r2 score otherwise.
* *cv* accepts integer values and determines the cross-validation splitting strategy.
* *gcv\_mode* indicates which strategy to use when performing Leave-one-out cross validation.
* *store\_cv\_values* is a Boolean value and indicates whether the cross validation values shold be stored or not.
* *alpha\_per\_target* accepts Boolean value nad describes whether to to optimize alpha value.

**5.3.6 Elastic Net Regression:**

Elastic net is the combination of linear regression with lasso and ridge. It is a regularized linear regression that combines L1 and L2 penalty functions.During training it adds regularized penalties to loss function.

elastic\_net\_penalty = (alpha \* l1\_penalty) + ((1 – alpha) \* l2\_penalty)

where,

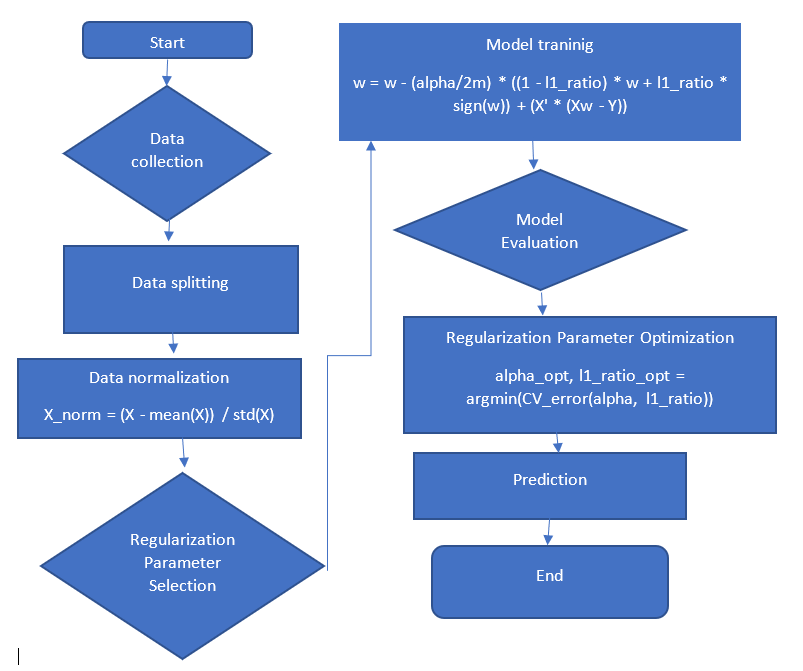
l1\_penalty= Loss + λ \* Σ ||w||

l2\_penalty= Loss + λ \* Σ ||w||2

Loss is sum of squared residual

λ is penalty

w is slope of the curve



**Fig 5.7 Architecture of Elastic net regression**

**Working of Elastic net regression:**

* Data Collection: Data is gathered for the dependent and the independent variables.
* Data Normalization: The data is normalized to ensure that all the features have the same scale.
* Regularization parameter selection: The value for the regularization parameter and λ is selected. ‘λ’ and ‘’ controls the strength of the L1 and L2 penalty terms.
* Creating Elastic net regression model: An elastic net regression model is created using the chosen and λ value.
* Model training: Elastic net regression model is trained using the training data.
* Model evaluation: The model performance of the model is evaluated using mean squared error or coefficient of determination.
* Predictions: Once the model is trained and evaluated, it can be used to make required predictions.

**Algorithm of elastic net regression:**

**Step-1**: Import essential libraries.

**Step-2**: Load the data using read\_csv() or read\_excel() functions.

**Step-3**: Pre-process the data by handling missing values, normalizing the features and splitting the data into training and testing sets and split it into train and test sets in either 8:2 ratio or 7:3 ratio.

**Step-4**: Fit the model using the ElasticNetCV() function from sklearn by choosing the required l1\_ratio.

**Step-5**: Evaluate the efficiency of the model by using the evaluation metrics like MAE, RMSE,R2 score etc., and use the trained model to make predictions.

**Methods used:**

*sklearn.linear\_model.ElasticNetCV(l1\_ratio,eps,n\_alphas,fit\_intercept,max\_iter,tol,cv,copy\_X,verbose,n\_jobs,positive,random\_state,selection)*

Here,

* *l1\_ratio* is a floating value that indicates the penalty of l1 regularization.
* *eps* indicates the length of the path.
* *n\_alphas* accepts integer values and indicates numberof alphas along regularization path.
* *alphas* is a list of alphas where to compute the models.
* *fit\_intercept* accepts boolean values which tells whether the intercept of the model should be calcilated or not.
* *cv* accepts integer values and determines the cross-validation splitting strategy.
* *copy\_X* accepts Boolean values and describes whether X value should be copied or overwritten.
* *n\_jobs* is an integer value which is used to set the number of jobs that can be used for computation.
* *positive* accepts boolean value and is used when the coefficients are required to be positive.
* *max\_iter* is the maximum number of iterations.
* *tol* accepts floating value and indicates the tolerance for the optimization.
* *verbose* is the amount of verbosity.
* *selection* is set to either ‘random’ or ‘cyclic’.

**5.3.7 MICE Forest:**

It stands for Multiple Imputation by chained equation. It imputes missing data using LightGBM in an iterative method. It can impute categorical and numeric data without much setup. This is used to fill the null values in the dataset.

**Working of MICE Forest:**

* Input: Data with missing values, number of imputations (m), and a random forest algorithm.
* Random forest creation: For each variable with missing values, create a random forest model using the observed values of that variable and the other variables as predictors.
* Imputation: For each variable with missing values, impute the missing values using the corresponding random forest model. Repeat this process m times to create m imputed datasets.
* Estimation: For each imputed dataset, use the completed data to estimate the model parameters of interest.
* Combining Final Estimates: Combine the m estimates of the model parameters using Rubin's rules to obtain the final estimates and their standard errors.

**Algorithm of MICE Forest:**

1.Fill in missing values from random draws of non-missing data

2.For each iteration:

3. For each variable v with missing values:

4. Optional:Subset data where v was originally non missing

5. Train model v~X where X are the other variables in the dataset

6. Do one of:

7. 1) Replace missing values with predictions from model

8. 2) Replace missing values using mean matching

9. End

10.End

**5.4 EVALUATION METRICS**

**5.4.1 R-Squared**

R-Squared (R2 Score) represents how close the data points are to the fitted line. It is one minus the ratio of sum of squares of residuals of regression model and total sum of squares of errors.

R2 =1- ((

where,

is actual value

is predicted value

is the mean of observed data

N is total number of data points

**5.4.2 Mean squared Error**

Mean squared error represents the mean of squared difference between the predicted and actual value of the target feature.

MSE = (1/N) \*

where,

is actual value

is predicted value

N is total number of data points

**5.4.3 Mean Absolute Error**

Mean absolute error is a metric used to measure the difference between predicted and actual values in a dataset. It is the average of the absolute differences between the predicted and actual values.

MAE=( \*(

where,

n is the total number of data points in the dataset

yi is the predicted value

xi is the actual value

**5.4.4 Max Error**

Max error also known as maximum absolute error, is a metric used to measure the worst-case error or the largest difference between the predicted and actual values in a dataset.

Max Error = max (|yi – xi|)

where,

yi is the predicted value

xi is the actual value

**5.4.5 Explained variance score**

Explained Variance Score (EVS)is a metric used to measure the proportion of variance in the target variable that is explained by the model.

EVS = 1 - (Var(y\_true - y\_pred) / Var(y\_true))

where,

y\_true is the true values of the target variable

y\_pred is the predicted values of the target variable

Var() is the variance of the differences between the true and predicted values.

**Chapter 6**

# **IMPLEMENTATION**

The proposed methodology has been divided into following steps. The steps are further explained in detail.

1. Data Extraction
2. Data Pre-processing
3. Data type conversion
4. Handling missing values
5. Feature Construction
6. Training the models
7. Model Evaluation
8. WQI prediction

**6.1 DATA EXTRACTION**

The data is extracted from two sources. They are:

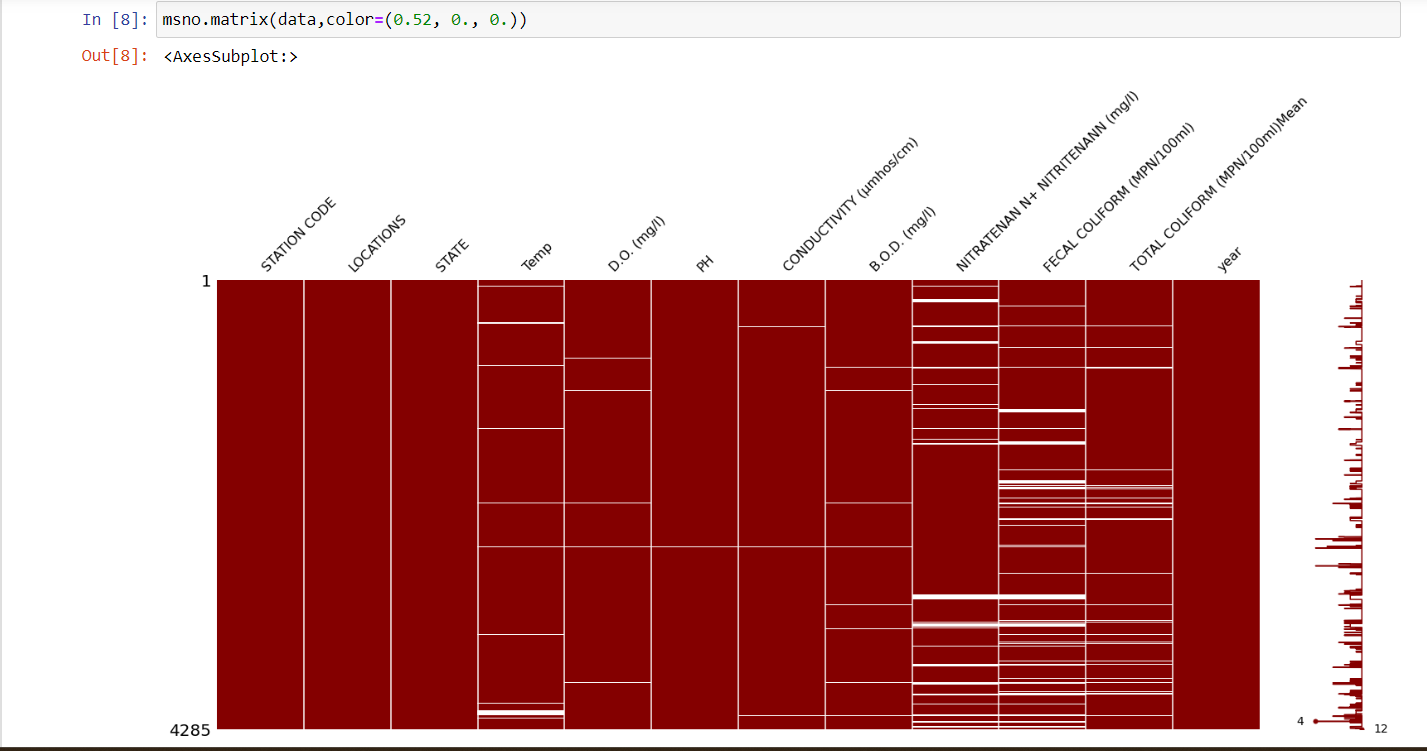
* Kaggle: The Indian water quality dataset from Kaggle contains data from the year 2003 to 2014.
* Central Pollution Control Board website: The data of the years 2015 to 2021 is collected from CPCB website of India in the form of PDF’s. The PDFs are then converted to excel formats.

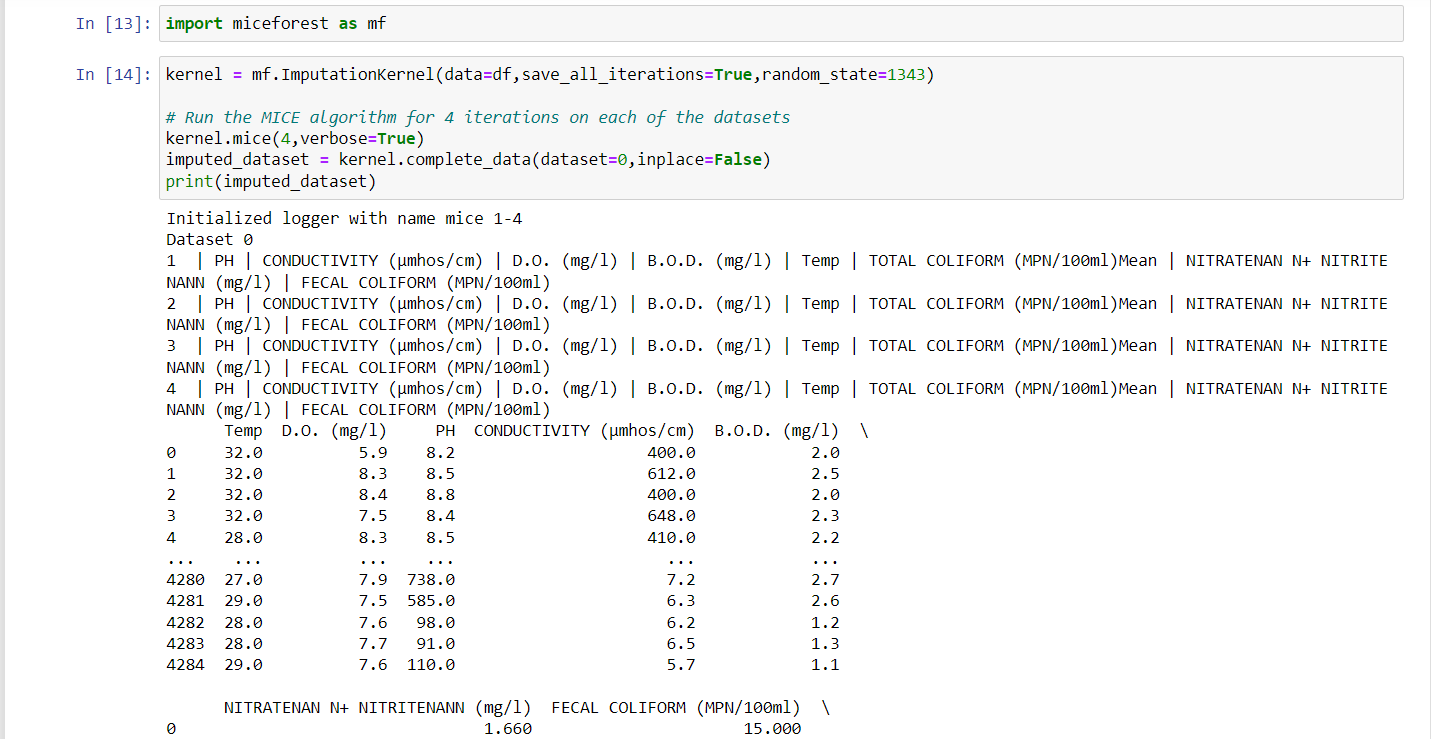
The extracted datasets are then merged to form a single dataset in excel format. The dataset contains several water parameters like pH, dissolved oxygen, B.O.D etc.,

**6.2 DATA PRE-PROCESSING**

1. Data type conversion: The data types of the features in the extracted dataset is ‘object.’ These features are transformed to ‘float64’ data type.



1. Handling missing values: The missing values in the data are imputed using MICE forest. This is done by using the mice() function in the miceforest library. The mice function is computed on the kernel dataset by giving iteration as parameter.



1. Feature Construction: Feature construction refers to addition of a new feature to the dataset. Here the feature constructed is WQI.

WQI stands for water quality index. WQI represents the quality of water in the form of a number in a range of 0-100. The WQI is calculated using the following formula:

WQI) / )

WQI calculation steps:

Step-1: Assign the sub-index (qi) value according to the Indian water quality standards.

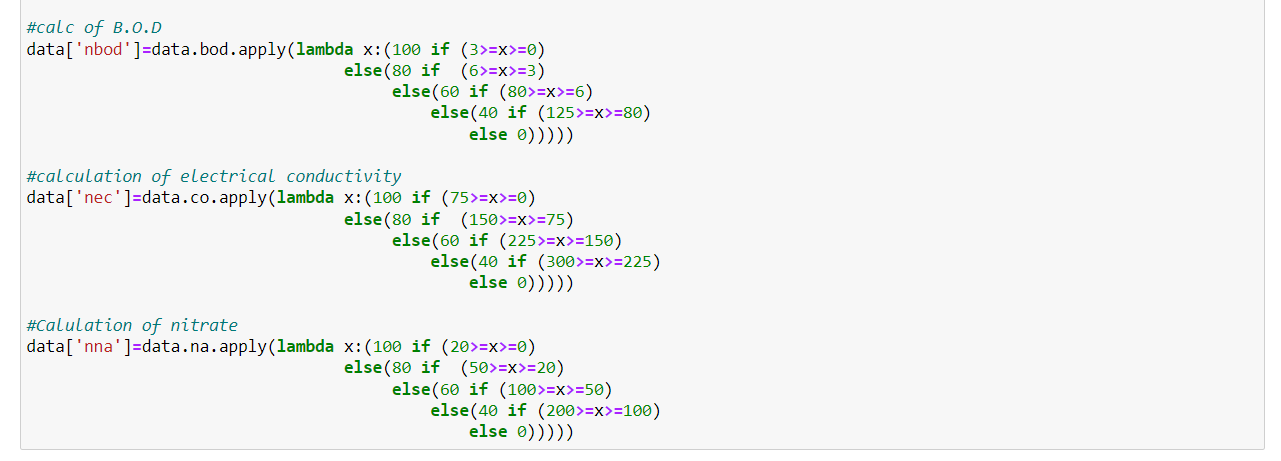
|  |  |
| --- | --- |
| Feature name | Unit weight |
| pH | 0.165 |
| DO | 0.281 |
| TC | 0.281 |
| BOD | 0.234 |
| EC | 0.009 |
| NN | 0.028 |

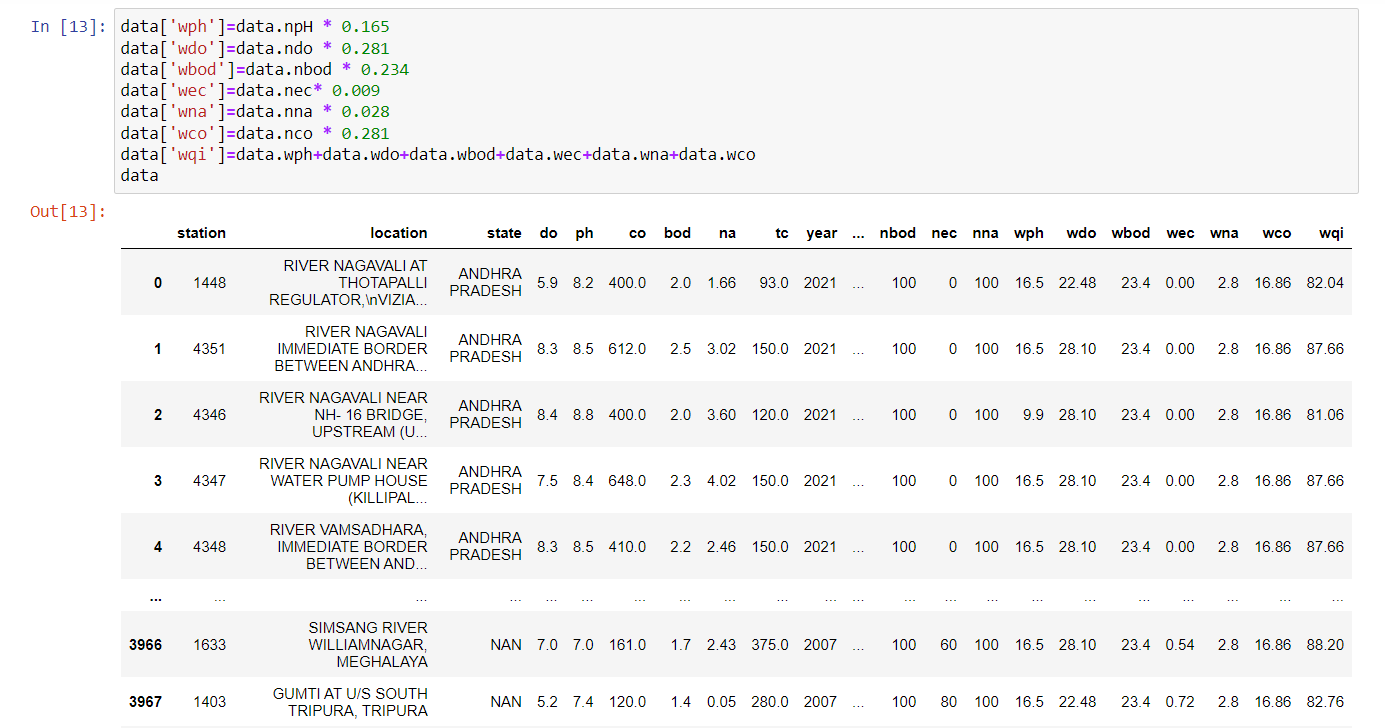
Step-2: Multiply sub-index and relative weight (wi).

**Table 6.1 Relative weights of water quality parameters**

Step-3: Mean of sum of wiqi values is considered as the water quality index for each year.

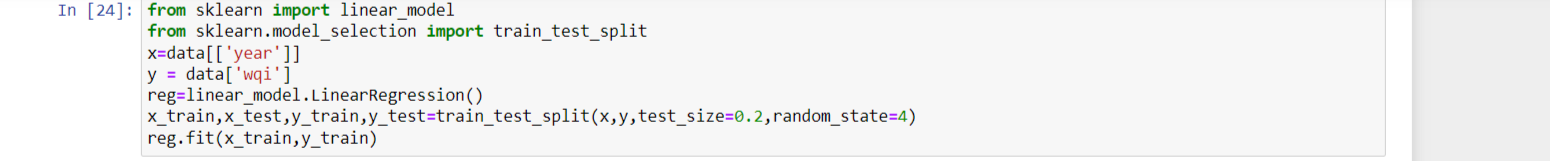
****

****

****

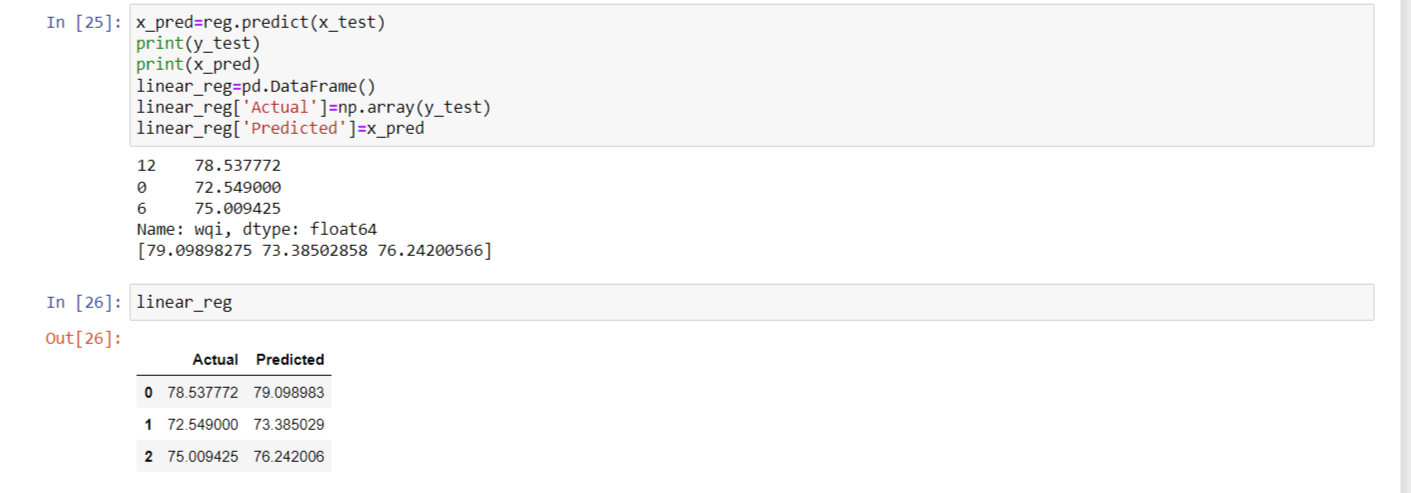
**6.3 Training the models**

The initial step in training the models is splitting the pre-processed data into training and testing data.This step can be done by using train\_test\_split(). The considered parameters of train\_test\_split() are years in array format, wqi values in array format, requires size of the test data(test\_size) and random\_state=4 so that the records in train data and test data are fixed for every iteration.



The training data is used to build the model. The algorithms used to train the dataset are Linear regression, Polynomial regression, Support vector regression, Ridge regression, Lasso regression and Elastic net regression. The training of the models is done as shown below.

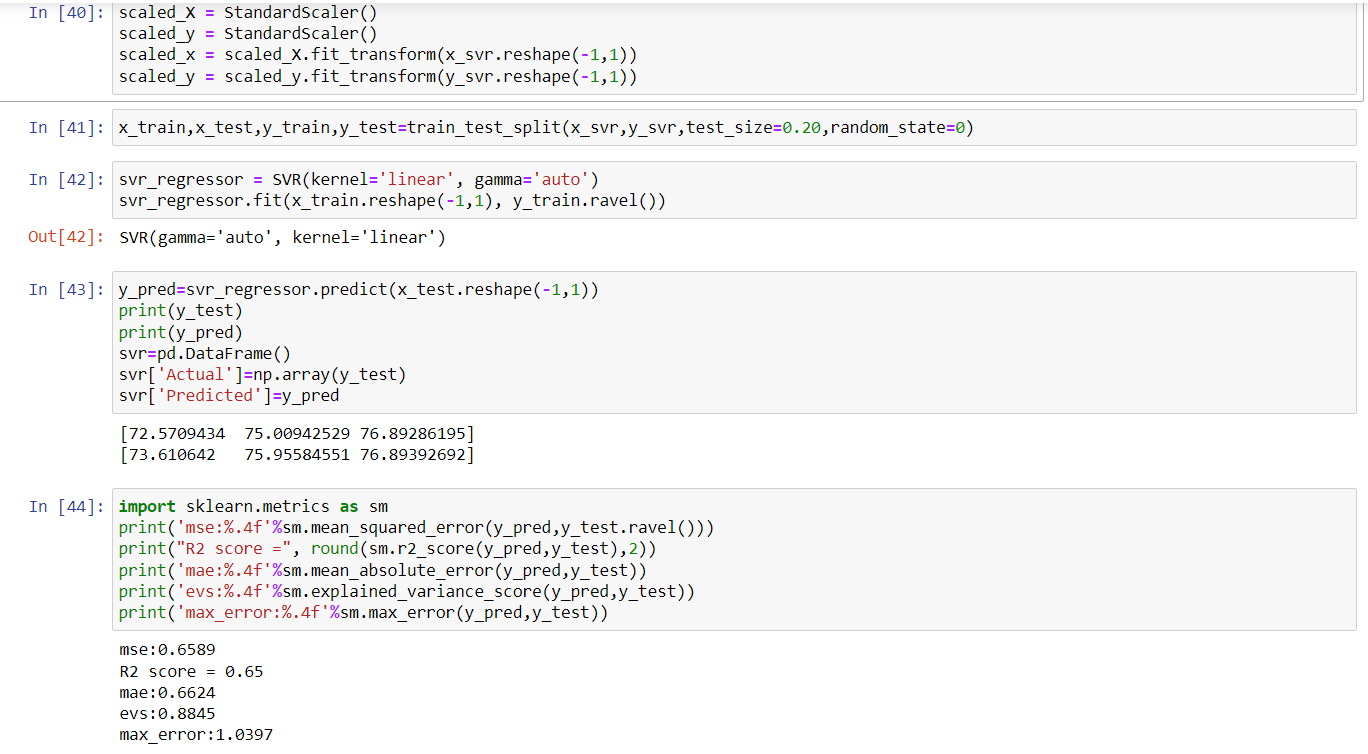
1.Linear Regression

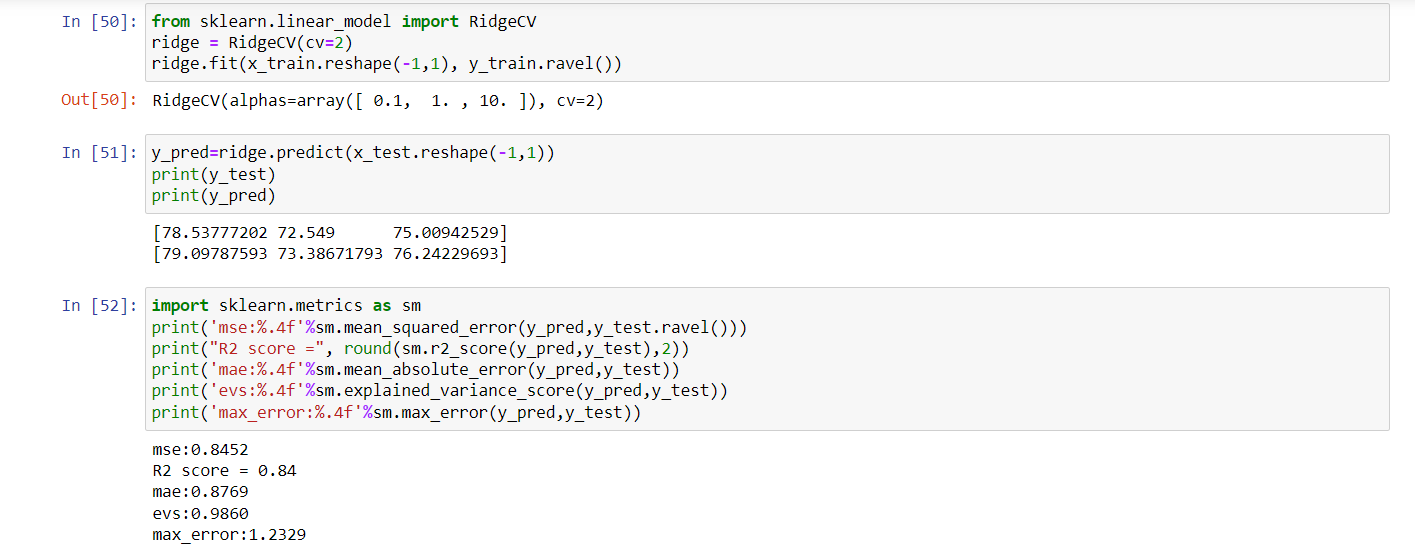


2.Polynomial Regression



3.Support vector regression



4.Ridge Regression

5.Lasso Regression



6.Elastic Net Regression



**6.4 Model Evaluation**

Model evaluation is used to understand the performance of the model. R2 score, MSE, MAE, Explained variance score and Max error evaluation metrics are used to identify the efficient model. The model with high R2 score and low MSE can be considered efficient. The R2 score and MSE are imported from sklearn.metrics.

The model evaluation is done as follows:

1.Predict the values using the trained model.

2.Calculate the evaluation scores using the predicted values and actual values.

**6.5 WQI Prediction**

The model with high R2 score and low MSE is used to predict the WQI in this paper. The accurate model is saved using the pickle library and a website is created using flask. WQI value is predicted by giving the year value as input by the user. Upon entering the year value, the application shows the predicted water quality index, the percentage increase or decrease in the water quality index from the previous year and the purposes for which the water can be used.

**Chapter 7**

# **RESULT ANALYSIS**

The models used for training are Linear regression, Polynomial regression, Support vector regression, Ridge regression, Lasso regression and Elastic net regression. After training and testing the metrics, R2 score, mean squared Error, Mean absolute error, max error and explained variance score are used to evaluate the models. The result analysis of the models is as follows:

* Support vector regression is not suitable for the prediction of water quality index.
* The performance of Linear regression and Ridge regression is approximately similar.
* Polynomial regression performs better than the regularization models.
* Elastic Net regression provides most accurate results compared to the other models and the error is also very low.

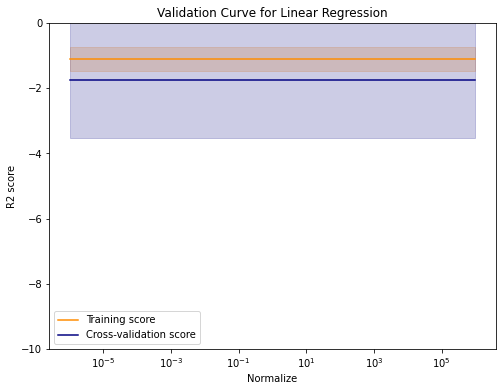
The results of the models are shown in the following Table 7.1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Model | R2 Score | MSE | MAE | EVS | Max error |
| Linear regression | 0.844826 | 0.844385 | 0.876607 | 0.986043 | 1.232580 |
| Polynomial regression | 0.867802 | 0.797668 | 0.799403 | 0.973711 | 1.362558 |
| Support vector regression | 0.654424 | 0.658895 | 0.662395 | 0.884548 | 1.039699 |
| Ridge regression | 0.844533 | 0.845153 | 0.876898 | 0.985982 | 1.232872 |
| Lasso regression | 0.844226 | 0.845958 | 0.877202 | 0.985918 | 1.233176 |
| Elastic net regression | 0.887058 | 0.624138 | 0.783399 | 0.998114 | 0.927783 |

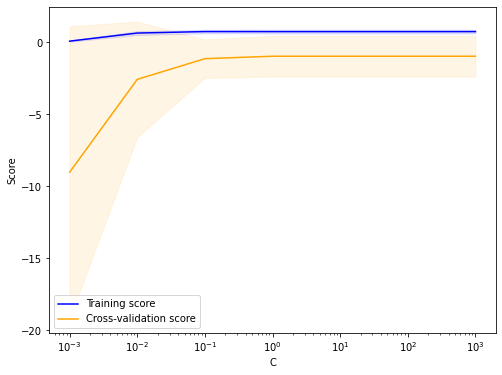
**Table 7.1 Results of evaluation metrics**

A validation curve is a graphical representation of a machine learning model's performance measures, including accuracy, precision, recall, and F1 score, among others. The plot displays the performance metrics against a hyperparameter that governs the model's complexity, revealing how the model's performance changes with the hyperparameter's value. The primary aim of a validation curve is to determine the hyperparameter's optimal value, which yields the model's maximum performance.

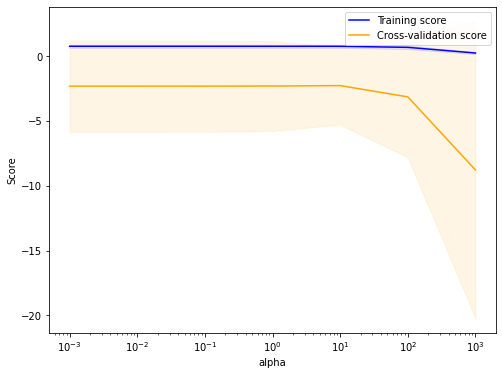
The validation curves for the models are shown below



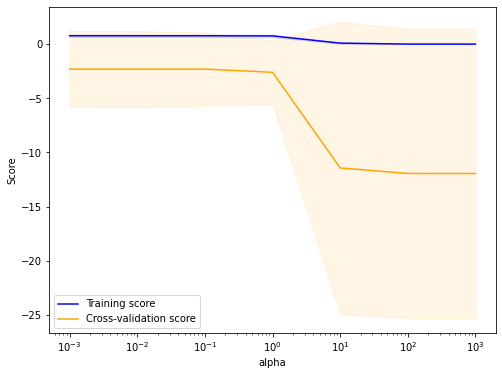
**Fig. 7.1 Validation curve of Linear regression**



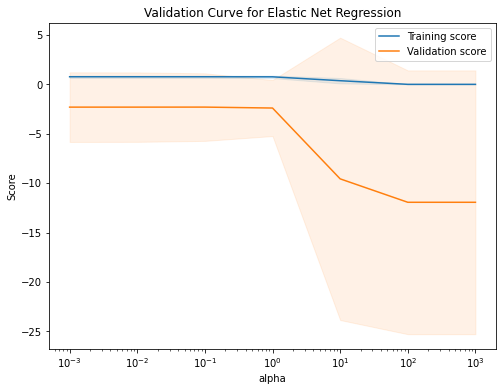
**Fig. 7.2 Validation curve of Support vector regression**



**Fig. 7.3 Validation curve of Ridge regression**

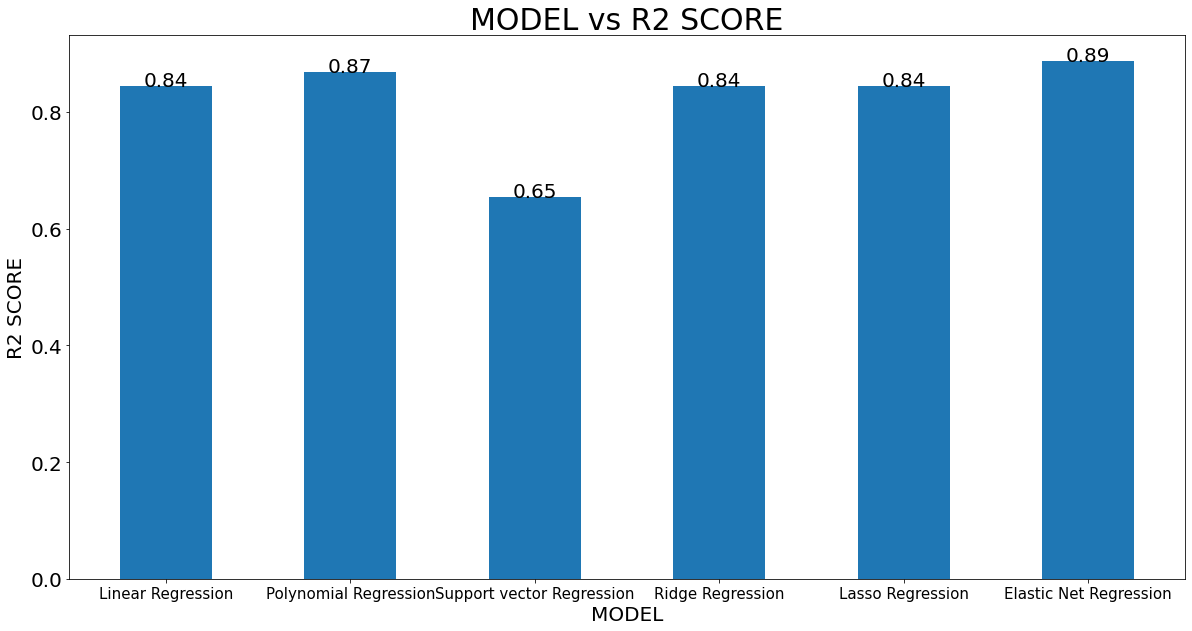
****

**Fig. 7.4 Validation curve of LASSO regression**

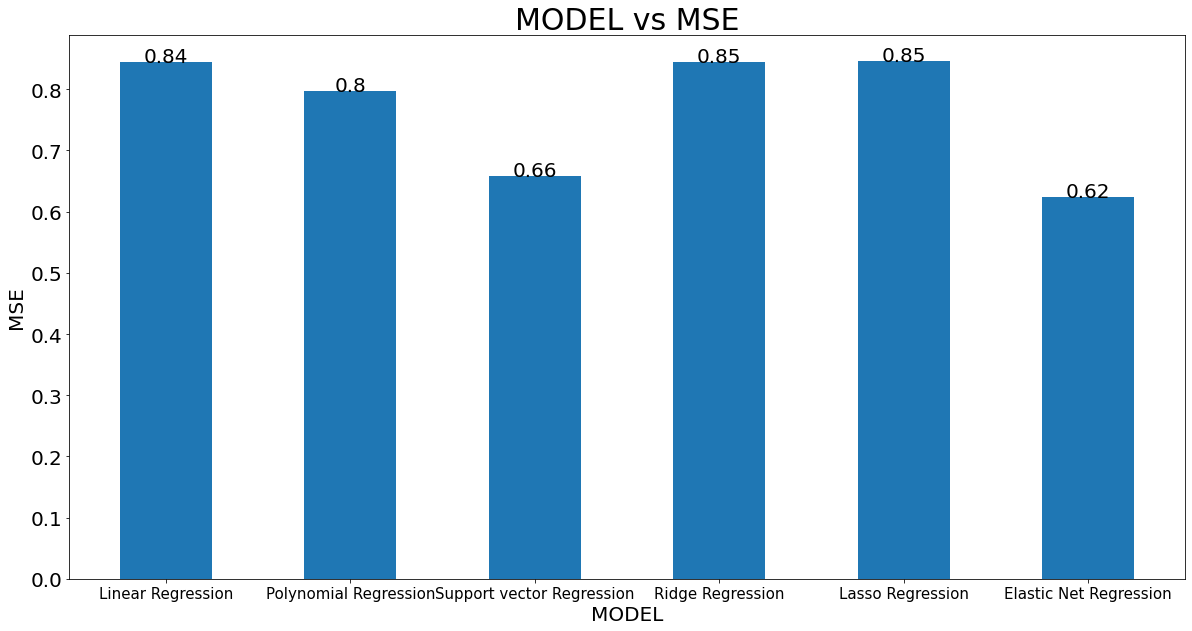
****

**Fig. 7.5 Validation curve of Elastic net regression**

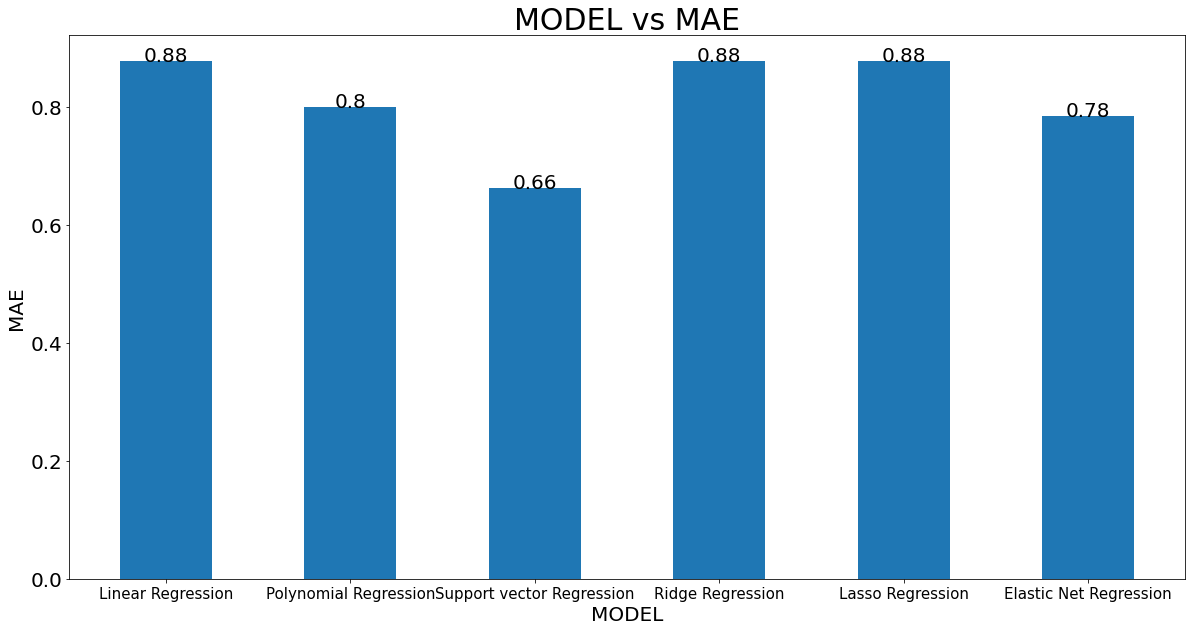
The findings of evaluation metrics are plotted as a bar graph and are shown below:



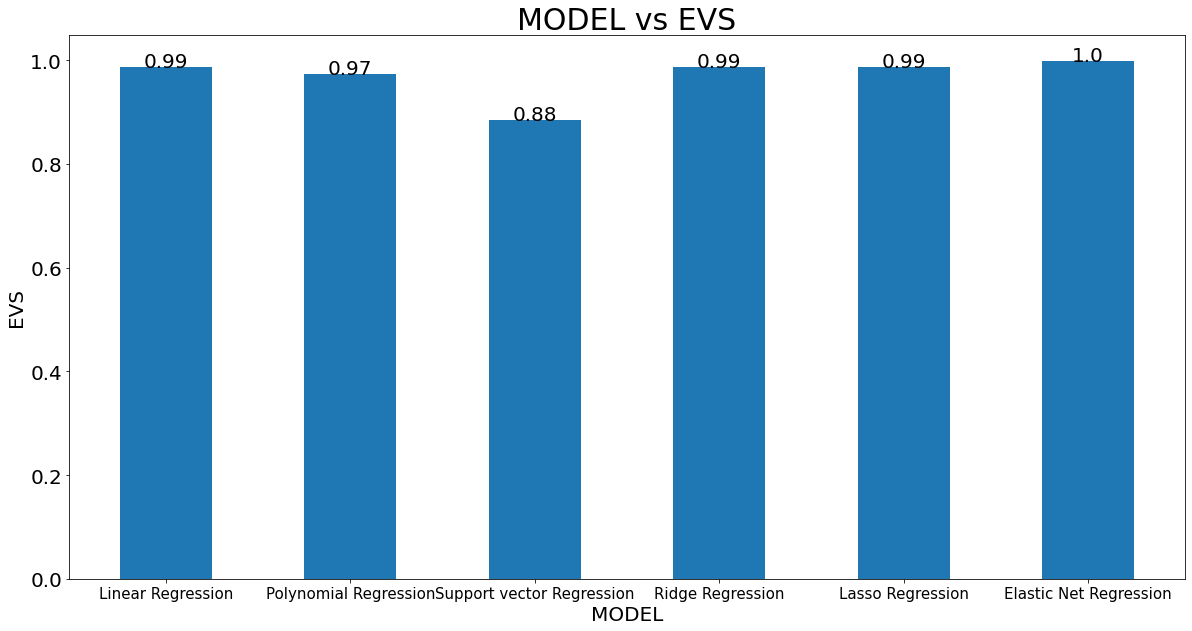
**Fig. 7.6 Model Vs R2 Score**



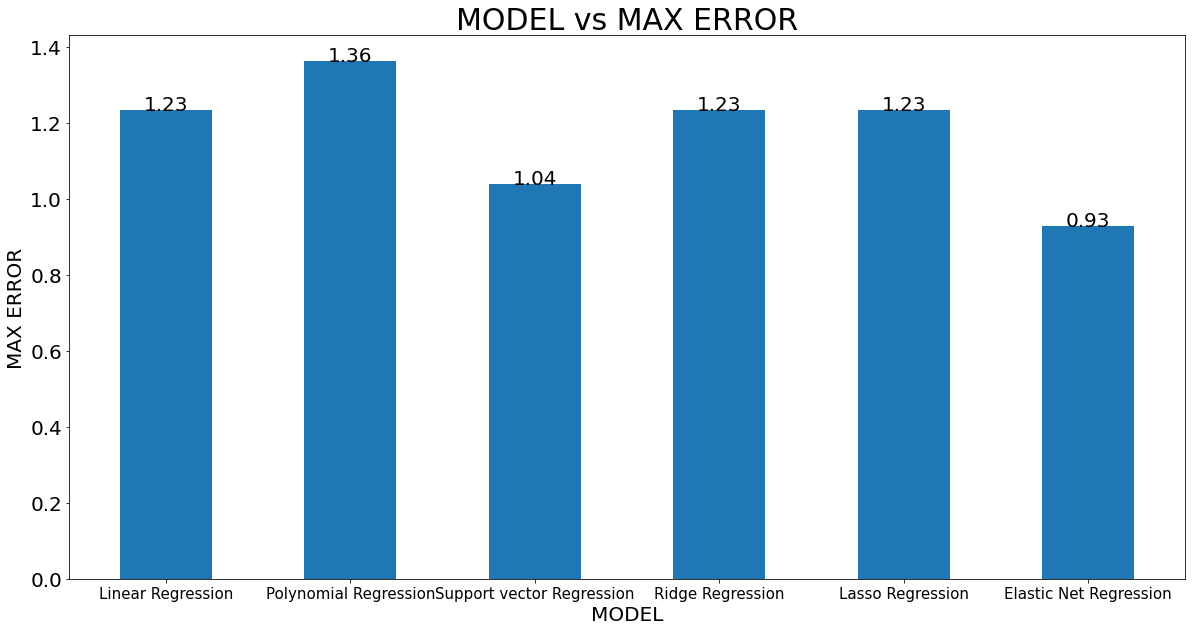
**Fig. 7.7 Model Vs MSE**



**Fig. 7.8 Model Vs MAE**



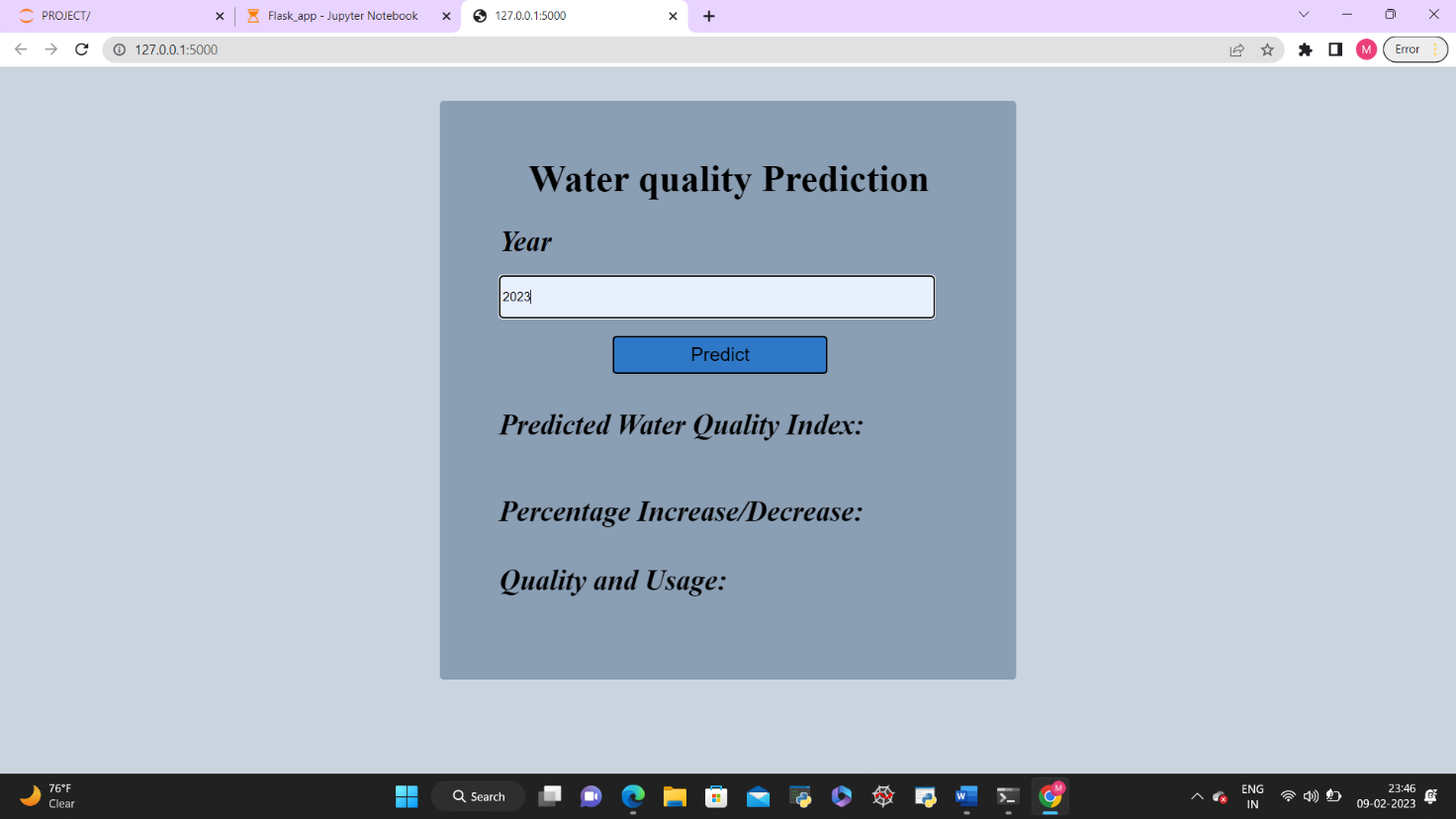
**Fig. 7.9 Model Vs EVS**

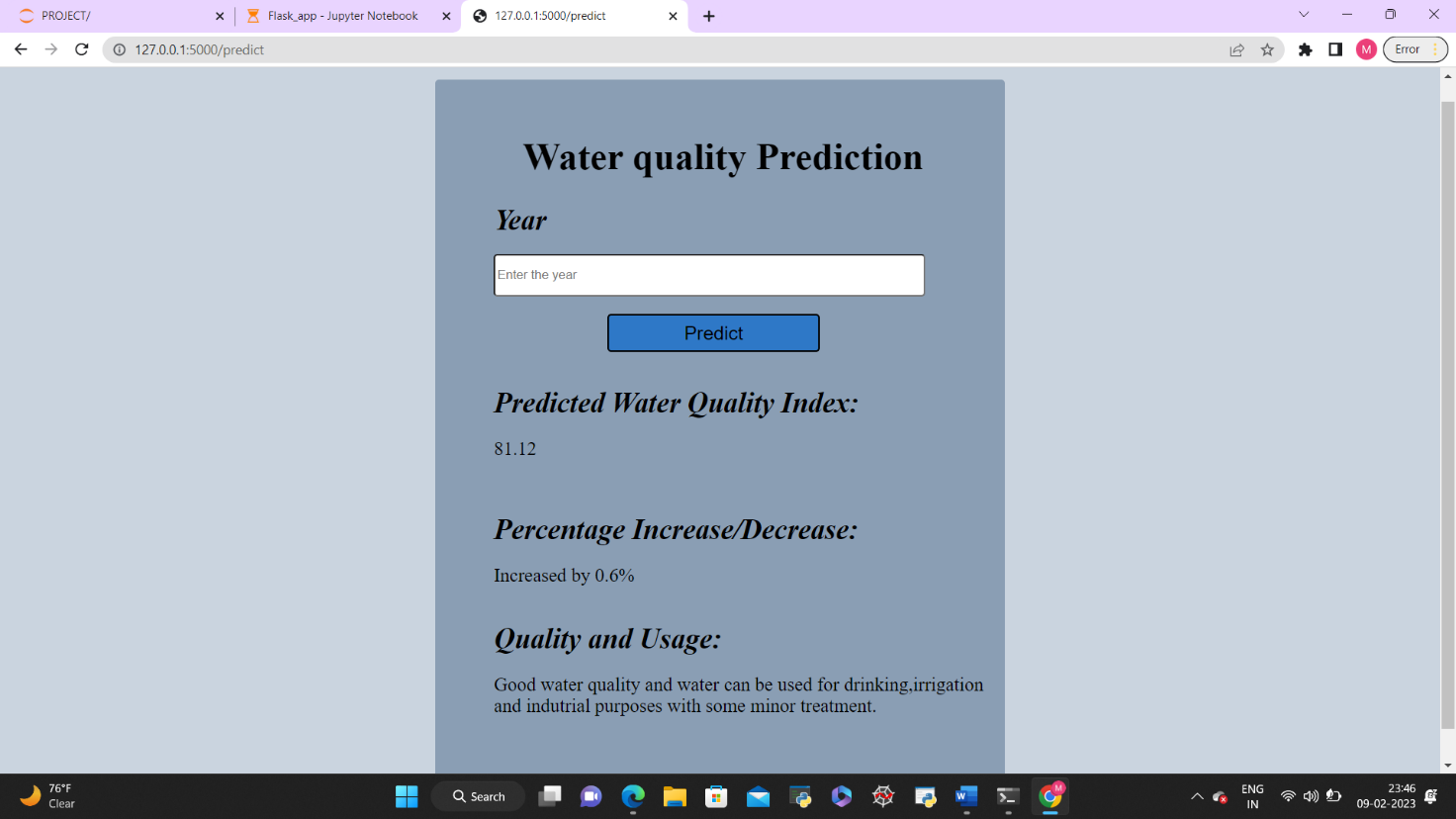


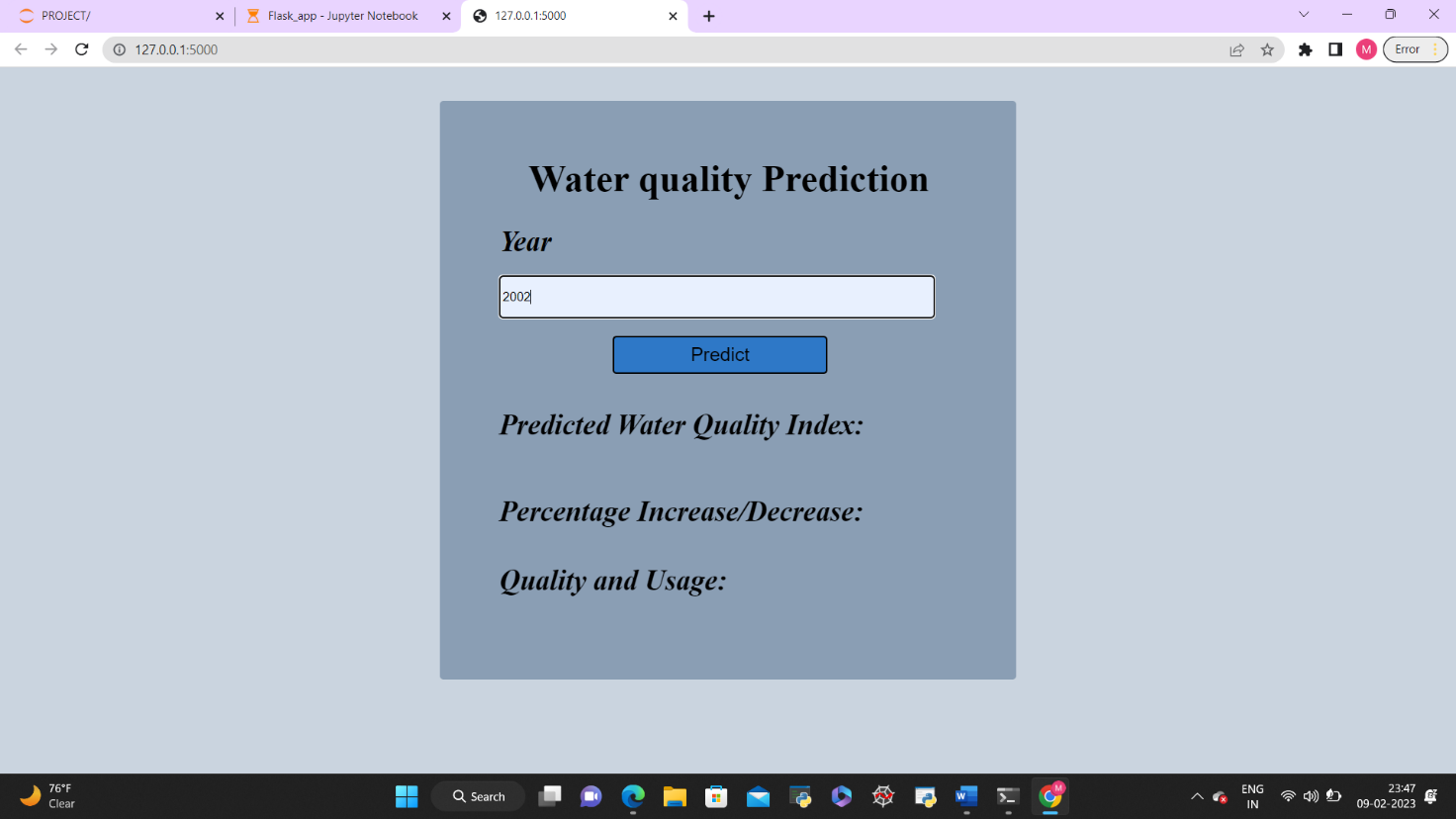
**Fig. 7.10 Model Vs Max error**

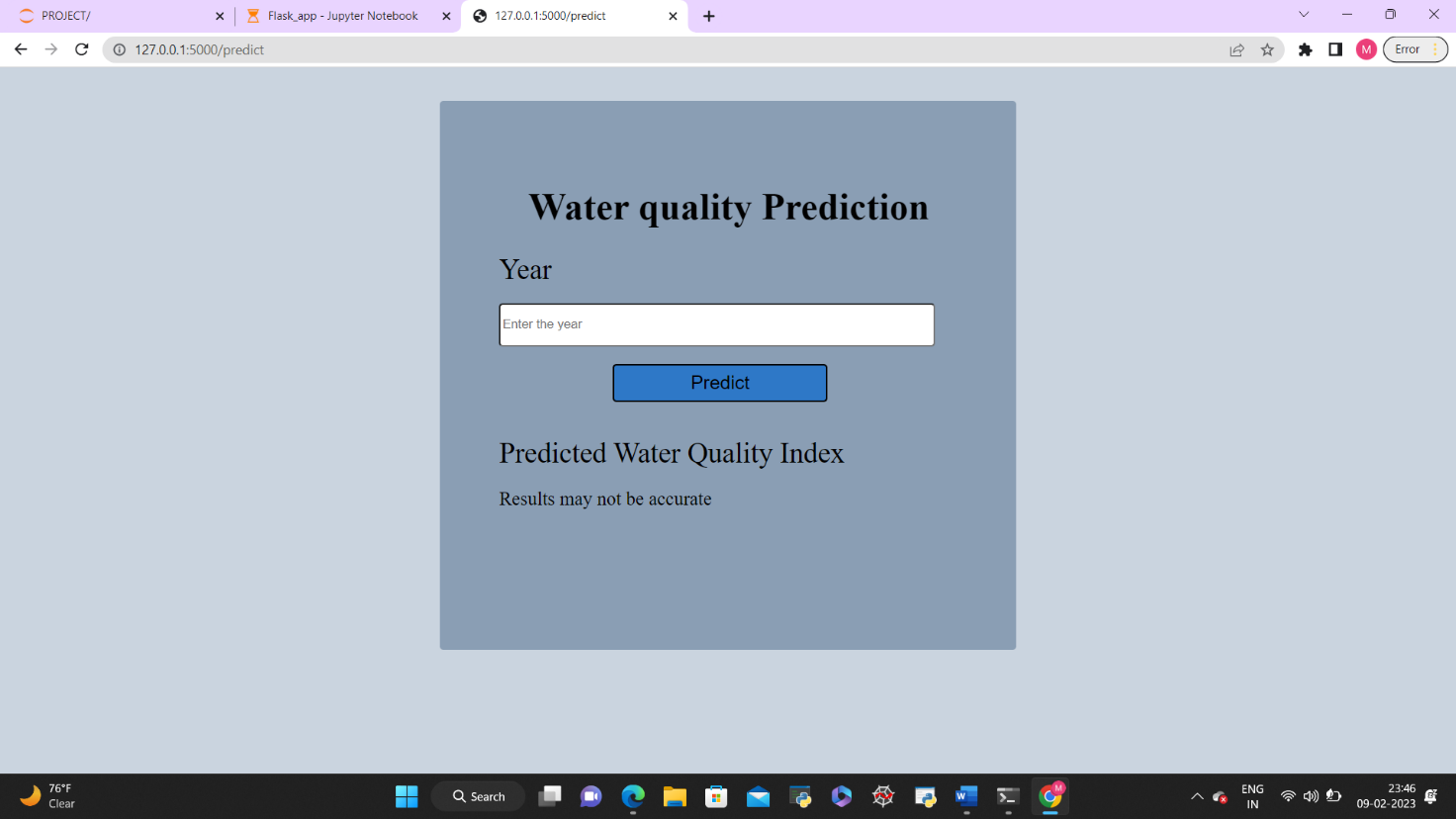
The results of evaluation metric scores reveal that the elastic net regression performs better when compared to the other models.It can be observed that ridge and LASSO regularization models perform approximately similar.

The elastic net regression is deployed using flask and an interface has been created to predict the water quality index when the year value is given as an input. The snapshots of interface are shown below.









**Chapter 8**

# **CONCLUSION**

The results of this study substantiate the use of machine learning algorithms to forecast the water quality index of India. Except Support vector regression all the algorithms provide accurate prediction with more than 80% efficiency. Elastic Net regression is used to develop the interface as it is found that it is more efficient than other algorithms for the proposed system. The developed interface provides the water quality index, the change in the water quality index compared to previous year and the activities for which the water can be used. This study can be further elaborated by extending the dataset using the data of the future years.

# **APPENDIX(A): SAMPLE CODE**

*#Importing required libraries*

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import sklearn.metrics as sm

data=pd.read\_excel(r"C:\Users\91939\OneDrive\Desktop\2021.xlsx")

data.dtypes

*#Changing the data types*

data['Temp']=pd.to\_numeric(data['Temp'],errors='coerce')

data['D.O.(mg/l)']=pd.to\_numeric(data['D.O. (mg/l)'],errors='coerce')

data['PH']=pd.to\_numeric(data['PH'],errors='coerce')

data['B.O.D.(mg/l)']=pd.to\_numeric(data['B.O.D.(mg/l)'],errors='coerce')

data['CONDUCTIVITYµmhos/cm)']=pd.to\_numeric(data['CONDUCTIVITY (µmhos/cm)’],

errors='coerce')

data['NITRATENAN N+ NITRITENANN (mg/l)']=pd.to\_numeric(data['NITRATENAN N+ NITRITENANN (mg/l)'],errors='coerce')

data['TOTAL COLIFORM (MPN/100ml)Mean']=pd.to\_numeric(data['TOTAL COLIFORM (MPN/100ml)Mean'],errors='coerce')

data['FECAL COLIFORM (MPN/100ml)']=pd.to\_numeric(data['FECAL COLIFORM (MPN/100ml)'],errors='coerce')

data.dtypes

data.shape

*#Imputation of missing values*

pip install missingno

import missingno as msno

msno.matrix(data,color=(0.52, 0., 0.))

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

null\_values

pip install miceforest

import miceforest as mf

df=data[['Temp','D.O. (mg/l)','PH','CONDUCTIVITY (µmhos/cm)','B.O.D. (mg/l)','NITRATENAN N+ NITRITENANN (mg/l)','FECAL COLIFORM (MPN/100ml)','TOTAL COLIFORM (MPN/100ml)Mean']]

kernel = mf.ImputationKernel(data=df,save\_all\_iterations=True,random\_state=1343)

*# Run the MICE algorithm for 3 iterations on each of the datasets*

kernel.mice(3,verbose=True)

completed\_dataset = kernel.complete\_data(dataset=0, inplace=False)

print(completed\_dataset)

d=pd.DataFrame(completed\_dataset)

d1=data[['STATION CODE','LOCATIONS','STATE']]

df=pd.concat([d1,d],axis=1)

df['year']=data['year']

data=df

*#Considering required values from the dataset*

start=0

end=3971

station=data.iloc [start:end ,0]

location=data.iloc [start:end ,1]

state=data.iloc [start:end ,2]

do= data.iloc [start:end ,4].astype(np.float64)

ph = data.iloc[ start:end,5]

co = data.iloc [start:end ,6].astype(np.float64)

bod = data.iloc [start:end ,7].astype(np.float64)

na= data.iloc [start:end ,8].astype(np.float64)

tc=data.iloc [start:end ,10].astype(np.float64)

year=data.iloc[start:end,11]

data.head()

data=pd.concat([station,location,state,do,ph,co,bod,na,tc,year],axis=1)

data.columns=['station','location','state','do','ph','co','bod','na','tc',

'year']

*#Calculation of PH*

data['npH']=data.ph.apply(lambda x: (100 if (8.5>=x>=7)

else(80 if (8.6>=x>=8.5) or (6.9>=x>=6.8)

else(60 if (8.8>=x>=8.6) or (6.8>=x>=6.7)

else(40 if (9>=x>=8.8) or (6.7>=x>=6.5)

else 0)))))

*#Calculation of Dissolved Oxygen*

data['ndo']=data.do.apply(lambda x:(100 if (x>=6)

else(80 if (6>=x>=5.1)

else(60 if (5>=x>=4.1)

else(40 if (4>=x>=3)

else 0)))))

*#Calculation of Total Coliform*

data['nco']=data.tc.apply(lambda x:(100 if (5>=x>=0)

else(80 if (50>=x>=5)

else(60 if (500>=x>=50)

else(40 if (10000>=x>=500)

else 0)))))

*# Calculation of B.O.D*

data['nbod']=data.bod.apply(lambda x:(100 if (3>=x>=0)

else(80 if (6>=x>=3)

else(60 if (80>=x>=6)

else(40 if (125>=x>=80)

else 0)))))

*#Calculation of electrical conductivity*

data['nec']=data.co.apply(lambda x:(100 if (75>=x>=0)

else(80 if (150>=x>=75)

else(60 if (225>=x>=150)

else(40 if (300>=x>=225)

else 0)))))

*#Calculation of nitrate*

data['nna']=data.na.apply(lambda x:(100 if (20>=x>=0)

else(80 if (50>=x>=20)

else(60 if (100>=x>=50)

else(40 if (200>=x>=100)

else 0)))))

*#Calculating weighted averages*

data['wph']=data.npH \* 0.165

data['wdo']=data.ndo \* 0.281

data['wbod']=data.nbod \* 0.234

data['wec']=data.nec\* 0.009

data['wna']=data.nna \* 0.028

data['wco']=data.nco \* 0.281

data['wqi']=data.wph+data.wdo+data.wbod+data.wec+data.wna+data.wco

data

*#Calculation overall wqi for each year*

agr=data.groupby('year')['wqi'].mean()

agr.head()

agr

data=agr.reset\_index(level=0,inplace=False)

data = data[np.isfinite(data['wqi'])]

data.head()

*#Scatter plot of data points*

x=data[['year']]

y = data['wqi']

plt.scatter(x,y)

plt.show()

*#Plot of data points*

import matplotlib.pyplot as plt

data=data.set\_index('year')

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

plt.show()

data=data.reset\_index(level=0,inplace=False)

data

*#Creating lists to store results*

results=[]

names=[]

errors=[]

*#Linear regression*

from sklearn import linear\_model

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

x=data[['year']]

y = data['wqi']

reg=linear\_model.LinearRegression()

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2,random\_state=4)

reg.fit(x\_train,y\_train)

x\_pred=reg.predict(x\_test)

print(y\_test)

print(x\_pred)

*#Evaluation Metrics*

print('mse:%.4f'%sm.mean\_squared\_error(x\_pred,y\_test))

print("R2 score =", round(sm.r2\_score(x\_pred,y\_test), 2))

names.append('Linear Regression')

results.append(sm.r2\_score(x\_pred,y\_test))

errors.append(sm.mean\_squared\_error(x\_pred,y\_test))

#Plotting the actual and predicted results

x\_axis=x

y\_axis=y

y1\_axis=reg.predict(x)

plt.scatter(x\_axis,y\_axis)

plt.plot(x\_axis,y1\_axis,color='r')

plt.title("linear regression")

plt.show()

*#Polynomial regression*

from sklearn.preprocessing import PolynomialFeatures

poly\_reg= PolynomialFeatures(degree = 2)

x\_poly = poly\_reg.fit\_transform(x\_train)

reg.fit(x\_poly,y\_train)

y\_pred\_pr = reg.predict(poly\_reg.fit\_transform(x\_test))

*#Evaluation Metrics*

print('mse:%.2f'%sm.mean\_squared\_error(y\_pred\_pr,y\_test))

print("R2 score =", round(sm.r2\_score(y\_pred\_pr,y\_test), 2))

names.append('Polynomial Regression')

results.append(sm.r2\_score(y\_pred\_pr,y\_test))

errors.append(sm.mean\_squared\_error(y\_pred\_pr,y\_test))

*#Plotting polynomial regression*

x\_axis=x

y\_axis=y

y1\_axis=reg.predict(poly\_reg.fit\_transform(x))

plt.scatter(x\_axis,y\_axis)

plt.plot(x\_axis,y1\_axis,color='r')

plt.title("Polynomial regression")

plt.show()

*#Support vector regression*

scaled\_X = StandardScaler()

scaled\_y = StandardScaler()

scaled\_x = scaled\_X.fit\_transform(x.reshape(-1,1))

scaled\_y = scaled\_y.fit\_transform(y.reshape(-1,1))

svr\_regressor = SVR(kernel='linear', gamma='auto')

svr\_regressor.fit(x\_train.reshape(-1,1), y\_train.ravel())

y\_pred=svr\_regressor.predict(x\_test.reshape(-1,1))

print(y\_test)

print(y\_pred)

*#Evaluation metrics*

print('mse:%.4f'%sm.mean\_squared\_error(y\_pred,y\_test.ravel()))

print("R2 score =", round(sm.r2\_score(y\_pred,y\_test),2))

names.append('Support vector Regression')

results.append(sm.r2\_score(y\_pred,y\_test.ravel()))

errors.append(sm.mean\_squared\_error(y\_pred,y\_test))

*#Plotting Support vector regression*

plt.scatter(x, y, color='teal', edgecolors='black', label='Train')

plt.plot(x,svr\_regressor.predict(x.reshape(-1,1)), color='orange', label='SVR')

plt.title('Simple Vector Regression')

plt.legend()

plt.show()

*#Ridge Regression*

from sklearn.linear\_model import RidgeCV

ridge = RidgeCV()

ridge.fit(x\_train.reshape(-1,1), y\_train.ravel())

y\_pred=ridge.predict(x\_test.reshape(-1,1))

print(y\_test)

print(y\_pred)

*#Evaluation metrics*

print('mse:%.4f'%sm.mean\_squared\_error(y\_pred,y\_test.ravel()))

print("R2 score =", round(sm.r2\_score(y\_pred,y\_test),2))

names.append('Ridge Regression')

results.append(sm.r2\_score(y\_pred,y\_test))

errors.append(sm.mean\_squared\_error(y\_pred,y\_test))

names.append('Ridge Regression')

results.append(sm.r2\_score(y\_pred,y\_test))

errors.append(sm.mean\_squared\_error(y\_pred,y\_test))

*#Plotting Ridge regression*

plt.scatter(x,y,color='teal',edgecolors='black',label='Actual observation points')

plt.plot(x, ridge.predict(x.reshape(-1,1)), color='orange', label='Ridge regressor')

plt.title('Ridge Regression')

plt.legend()

plt.show()

*#Lasso regression*

from sklearn.linear\_model import LassoCV

lasso = LassoCV()

lasso.fit(x\_train.reshape(-1,1), y\_train.ravel())

y\_pred=lasso.predict(x\_test.reshape(-1,1))

print(y\_test)

print(y\_pred)

*#Evaluation metrics*

print('mse:%.4f'%sm.mean\_squared\_error(y\_pred,y\_test.ravel()))

print("R2 score =", round(sm.r2\_score(y\_pred,y\_test),2))

names.append('Lasso Regression')

results.append(sm.r2\_score(y\_pred,y\_test))

errors.append(sm.mean\_squared\_error(y\_pred,y\_test))

*#Plotting Lasso regression*

plt.scatter(x\_l, y\_l, color='teal', edgecolors='black', label='Actual observation points')

plt.plot(x\_l, lasso.predict(x\_l.reshape(-1,1)), color='orange', label='LASSO regressor')

plt.title('LASSO Regression')

plt.legend()

plt.show()

*#Elastic Net regression*

from sklearn.linear\_model import ElasticNetCV

elasticNet = ElasticNetCV(cv=3) #l1\_ratio=0.6

elasticNet.fit(x\_train.reshape(-1,1), y\_train.ravel())

y\_pen=elasticNet.predict(x\_test.reshape(-1,1))

print(y\_pen)

print(y\_test)

*#Evaluation metrics*

print('mse:%.2f'%sm.mean\_squared\_error(y\_pen,y\_test.ravel()))

print("R2 score =", round(sm.r2\_score(y\_pen,y\_test),2))

names.append('Elastic Net Regression')

results.append(sm.r2\_score(y\_pen,y\_test))

errors.append(sm.mean\_squared\_error(y\_pen,y\_test))

*#Plotting Elastic Net regression*

plt.scatter(x, y, color='teal', edgecolors='black', label='Train')

plt.plot(x,elasticNet.predict(x.reshape(-1,1)),color='orange',label='ElasticNet regressor')

plt.title('ElasticNet Regression')

plt.legend()

plt.show()

*#Plotting results*

Results=pd.DataFrame()

Results['Model']=names

Results['R2 Score']=results

Results['Mean squared error']=errors

plt.barh(names, results)

plt.ylabel("Model")

plt.xlabel("R2 Score")

plt.title("Model vs R2 score")

plt.show()

plt.barh(names, errors)

plt.ylabel("Model")

plt.xlabel("MSE")

plt.title("Model vs MSE")

plt.show()

*#Saving Model*

import pickle

with open('water\_pred\_model.pkl', 'wb') as files:

pickle.dump(elasticNet, files)

*#Creating interface using flask*

import numpy as np

from flask import Flask, request, render\_template

import pickle

##from flask\_ngrok import run\_with\_ngrok

app = Flask(\_\_name\_\_)

#run\_with\_ngrok(app)

model = pickle.load(open('water\_pred\_model.pkl', 'rb'))

@app.route('/')

def home():

return render\_template('Pro2.html')

@app.route('/predict',methods=['POST'])

def predict():

input = [float(x) for x in request.form.values()]

prev\_year=[np.array([input[0]-1])]

final\_input = [np.array(input)]

prev\_pred=model.predict(prev\_year)

prediction = model.predict(final\_input)

percentage=((prediction-prev\_pred)/(prev\_pred))\*100

p\_i\_d=''

if(percentage>0):

p\_i\_d='Increased by {}%'.format(round(percentage[0],1))

elif(percentage==0):

p\_i\_d='Remained Constant'

else:

p\_i\_d='Decreased by {}%'.format(round(percentage[0],1))

n=prediction

result=''

if(n>90):

result="Excellent water quality and water can be used for drinking,irrigation and indutrial purposes."

elif(n<=90 and n>70):

result="Good water quality and water can be used for drinking,irrigation and indutrial purposes with some minor treatment."

elif(n<=70 and n>50):

result= "Medium water quality and water can be used for irrigation and industrial purposes with some minor treatment."

elif(n<=50 and n>25):

result="Bad water quality and water can be used for irrigation purposes with some minor treatment."

else:

result="Very poor water quality and water needs a proper treatment for any kind of usage."

if(input[0]<2007 or input[0]>=2024):

return render\_template('Pro.html',prediction\_text='Results may not be accurate')

return render\_template('Pro2.html',prediction\_text="{}".format(round(prediction[0],2)),Per=p\_i\_d,QNU=result)

if \_\_name\_\_ == "\_\_main\_\_":

app.debug=True

app.run(use\_reloader=False)

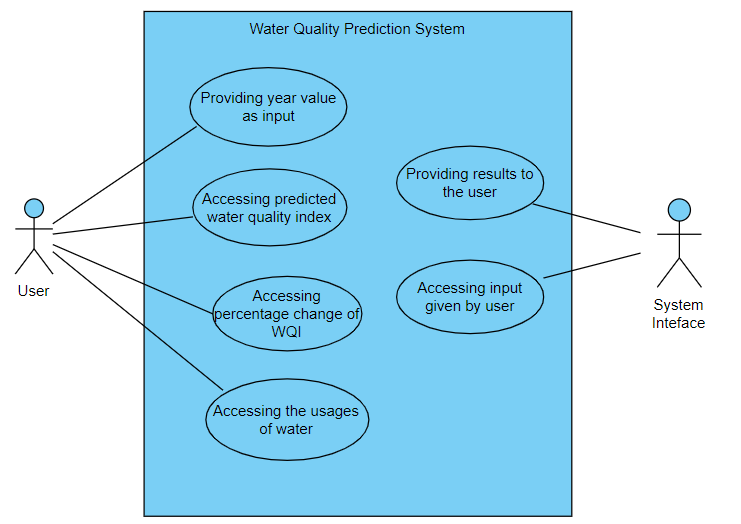
# **APPENDIX(B): UML DESIGN**

**UML DIAGRAMS:**

UML is to define a standard way to visualize the way a system has been designed and it is similar to blueprints used in other fields of engineering. It is a general purpose, development modelling language in the field of software engineering that is intended to provide a standard way to visualize the design of a system.

**B.1 Use case Diagram**

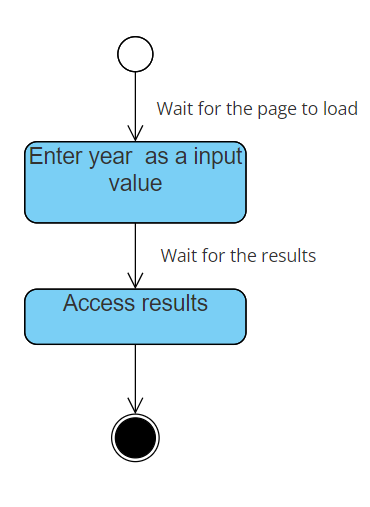
A use case is a list of steps that defines interaction between an actor and the system. The functionality of a system is described from the point of view of a user. The use case diagram is shown in Fig B.1

****

**Fig B.1 Use case diagram**

**B.2 State chart Diagram:**

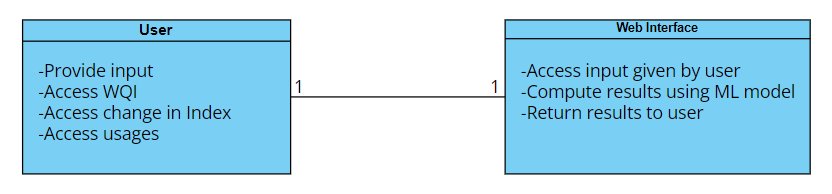
State diagrams are used to give an abstract description of the behavior of a system. This behavior is analyzed and represented by a series of events that can occur in one or more possible states. The state chart diagram is shown in Fig B.2

****

**Fig B.2 State chart diagram**

**B.3 Class Diagram**

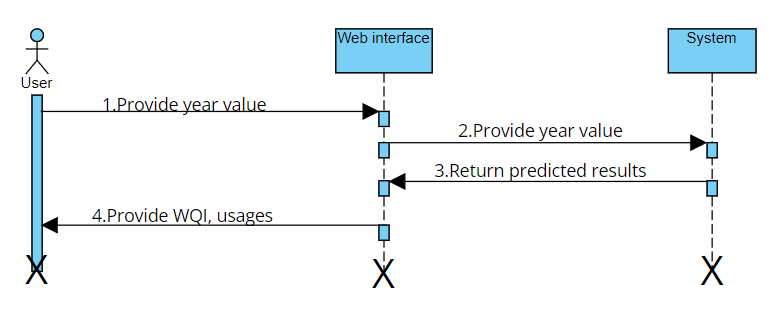
Class Diagram is a static structure in UML that is utilized to depict the structure of a system by presenting the classes within the system, along with their attributes, operations and the connections among the objects. The class diagram is shown in Fig B.3

****

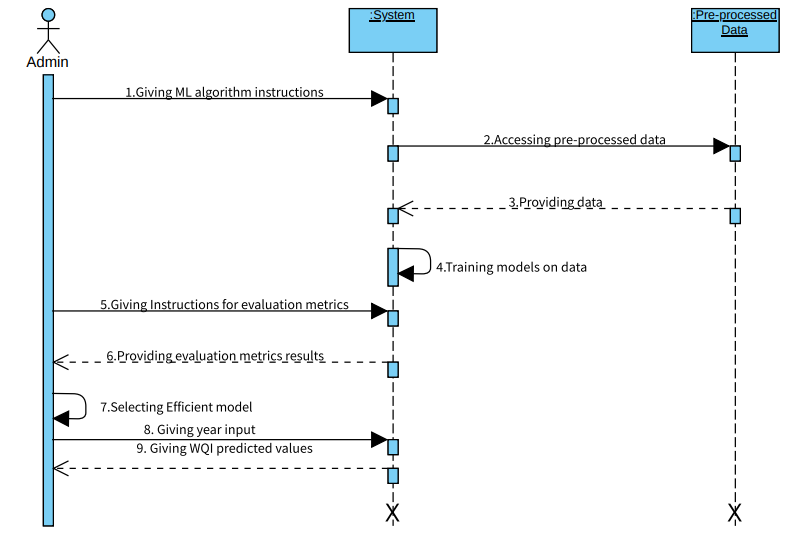
**Fig B.3 Class Diagram**

**B.4 Sequence Diagram:**

A sequence diagram or system sequence diagram (SSD) shows [process](https://en.wikipedia.org/wiki/Process_(computing)) interactions arranged in time sequence in the field of [software engineering](https://en.wikipedia.org/wiki/Software_engineering). The sequence diagram is shown in Fig B.4 and Fig B.5

****

**Fig B.4 Sequence Diagram of user’s interaction**

****

**Fig B.5 Sequence Diagram for Training the models**

# **REFERENCES**

[1] Shah, K.A., Joshi, G.S.,” Evaluation of water quality index for River Sabarmati, Gujarat, India”, Applied Water Science, vol 7, pp. 1349–1358 ,2017.

[2] Amir Hamzeh Haghiabi, Ali Heidar Nasrolahi and Abbas Parsaie, “Water quality prediction using machine learning methods”, IWA Publishing 2018 Water Quality Research Journal,vol. 53, issue.1, pp.3-13, 2018.

[3]Muharemi, F., D. Logofătu, and F. Leon. 2019. “Machine Learning Approaches for Anomaly Detection of Water Quality on a Real-world Data Set”, Journal of Information and Telecommunication, vol 3, pp.294–307,2019.

[4] Seyed Babak Haji Seyed Asadollah, Ahmad Sharafati, Davide Motta, Zaher Mundher Yaseen ,“River Water Quality Index prediction and uncertainty analysis: a comparative study of machine learning models”, Journal of Environmental Chemical Engineering,vol 9,issue 1, pp.1-47,2020.

[5] Tianan Deng, Kwok-Wing Chau, Huan-Feng Duan,”Machine learning based marine water quality prediction for coastal hydro-environment management”, Journal of Environmental Management,vol.284,pp.1-14, 2021.

[6] Hongfang Lu,Xin Ma, “Hybrid decision tree-based machine learning models for short-term water quality prediction”, Chemosphere,vol.249,pp.1-12, 2020.

[7] Donya Dezfooli, Seyed‑Mohammad Hosseini‑Moghari, Kumars Ebrahimi, Shahab Araghinejad, “Classification of water quality status based on minimum quality parameters: application of machine learning techniques” Springer International Publishing AG,vol 4,pp.311-324, 2018.

[8] Ali Najah Ahmed, Faridah Binti Othman , Haitham Abdulmohsin Afan , Rusul Khaleel Ibrahim , Chow Ming Fai , Md Shabbir Hossain , Mohammad Ehteram , Ahmed Elshafie, ”Machine learning methods for better water quality prediction”, Journal of Hydrology, vol.578,pp.1-18,2019.

[9] Duie Tien Bui ,Khabat Khosravi, John Tiefenbacher, Hoang Nguyen, Nerantzis Kazakis, “Improving prediction of water quality indices using novel hybrid machine-learning algorithms”, Science of total Environment,vol.721,pp.1-15,2020.

[10] Yafra Khan,Chai Soo See., “Predicting and Analyzing Water Quality using Machine Learning: A Comprehensive Model”, University Malaysia Sarawak,vol 1,pp.1-6,2016.

[11] Kangyang Chen, Hexia Chen, Chuanlong Zhou, Yichao Huang, Xiangyang Qi , Ruqin Shen , Fengrui Liu, Min Zuo, Xinyi Zou, Jinfeng Wang, Yan Zhang, Da Chen, Xingguo Chen, Yongfeng Deng, Hongqiang Ren, “Comparative analysis of surface water quality prediction performance and identification of key water parameters using different machine learning models based on big data”, Water Research,vol.171,pp.1-10,2020.

[12] Md Galal Uddin , Stephen Nash , Mir Talas Mahammad Diganta ,Azizur Rahman , Agnieszka I. Olbert ,” Robust machine learning algorithms for predicting coastal water quality index”, Journal of Environmental Management, [vol 321](https://www.sciencedirect.com/journal/journal-of-environmental-management/vol/321/suppl/C),pp .1-16, 2022.

[13] Furqan Rustam , Abid Ishaq, Sayyida Tabinda Kokab , Isabel de la Torre Diez , Juan Luis Vidal Mazón , Carmen Lili Rodríguez and Imran Ashraf , “An Artificial Neural Network Model for Water Quality and Water Consumption Prediction”,MDPI,vol 14,pp.1-24,2022.

[14] G. Kang, J. Z. Gao and G. Xie, "Data-driven water quality analysis and prediction: A survey", Proc. 3rd IEEE Int. Conf. Big Data Computing Service and Application, vol 1, pp. 224-232, 2017.

[15] Sani Isah Abba1 & Quoc Bao Pham, Gaurav Saini, Nguyen Thi Thuy Linh, Ali Najah Ahmed, Meriame Mohajane, Mohammadreza Khaledian, Rabiu Aliyu Abdulkadir, Quang-Vu Bach, “Implementation of data intelligence models coupled with ensemble machine learning for prediction of water quality index”, Environmental Science and Pollution Research, vol 27, pp 41527-41539,2020.

[16]Muhammad Sani Gaya,Sani Isah Abba,Aliyu Muhammad Abdu,Abubakar Ibrahim Tukur,Mubarak Auwal Saleh,Parvaneh Esmaili,Norhaliza Abdul Whab,”Estimation of water quality index using artificial intelligence approaches and multi-linear regression”,International Journal of Artificial Intelligence,vol 9,pp.126-134,March 2020.

[17]Luka Grbčića, Siniša Družetaa, Goran Maušab, Tomislav Lipićh , Darija Vukić Lušićc, Marta Alvira , Ivana Lučina, Ante Sikiricaa, Davor Davidovići , Vanja Travašc, Daniela Kalafatovićc, Kristina Pikeljj , Hana Fajkovićj , Toni Holjevića , Lado Kranjčevića,” Coastal Water Quality Prediction Based on Machine Learning with Feature Interpretation and Spatio–temporal Analysis”, Environmental Modelling and software, vol 155,pp 1-15, 2022.

[18]Wang, X., Zhang, F. & Ding, “ Evaluation of water quality based on a machine learning algorithm and water quality index for the Ebinur Lake Watershed”, China. Scientific Reports, vol 7, pp 1-18, 2017.

[19] Ali Omran Al-Sulttani, Mustafa Al-Mukhtar, Ali B Roomi, Aitazaz Farooque, Khaled Mohamed Khedher, Zaher Mundher Yaseen, “Proposition of New Ensemble Data-Intelligence Models for Surface Water Quality Prediction”,IEEE Access, vol 1, pp 1-15,2017.

[20]Charmaine Jerome and Anitha Pius,”Evaluation of water quality index and its impact on the quality of life in an industrial area in Bangalore, South India “, American journal of scientific and industrial research,, vol 1pp 1-9,2010.

[21] Md. Galal Uddin, Stephen Nash, Agnieszka I. Olbert,”A review of water quality index models and their use for assessing surface water quality”,Ecological Indicators,vol 122,pp .1-22,2021.

[22] Xu, T., Coco, G., & Neale, M.,”A predictive model of recreational water quality based on adaptive synthetic sampling algorithms and machine learning”, Water Research, pp 1-11,2020.

[23] Vinod Kothari, Suman Vij, SuneshKumar Sharma, Neha Gupta,”Correlation of various water quality parameters and water quality index of districts of Uttarakhand,Environmental and Sustainability Indicators”,vol 9,pp 1-8,2021

[24] V. Geethanjali,N.L. Anbarivan,”Indian Water Pollution Monitoring and Forecasting for Anomaly with Fail-Safe Wireless Sensor Networks using Machine learning techniques ”, International Research Journal of Engineering and Technology ,vol 6, pp. 771-781 ,2019.

[25] Md. Mehedi Hassan, Md. Mahedi Hassan, Laboni Akter, Md. Mushfiqur Rahman, Sadika Zaman, Khan Md. Hasib,Nusrat Jahan, Raisun Nasa Smrity, Jerin Farhana, M. Raihan, Swarnali Mollick, "Efficient Prediction of Water Quality Index (WQI) Using Machine Learning Algorithms",Atlantis Press,vol 1,pp. 1-12,2021.

[26] Umair Ahmed, Raﬁa Mumtaz 1, Hirra Anwar, Asad A. Shah, Rabia Irfan and JoséGarcía-Nieto, "Eﬃcient Water Quality Prediction Using Supervised Machine Learning",MDPI,vol 11, pp. 1-14,2019.

