**EVALUATING AND CLASSIFYING WATER QUALITY USING MACHINE LEARNING**

### A PROJECT REPORT

***Submitted by***

##### ANANDHA LAKSHMI P [REGISTER NO:211418104019]

**GOWTHAMI D [REGISTER NO:211418104071]**

**HARITHA R [REGISTER NO:211418104080]**

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

***of***

**BACHELOR OF ENGINEERING**

***in***

**COMPUTER SCIENCE AND ENGINEERING**



## PANIMALAR ENGINEERING COLLEGE

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

**MAY 2022**

# **PANIMALAR ENGINEERING COLLEGE**

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

**BONAFIDE CERTIFICATE**

Certified that this project report **“EVALUATING AND CLASSIFYING WATER QUALITY USING MACHINE LEARNING”**is the bonafide work of“**ANANDHALAKSHMIP[211418104019],GOWTHAMID[211418104071],HARITHA R[21141810480]”** who carried out the project work under my supervision.

**SIGNATURE SIGNATURE Dr.S.MURUGAVALLI,M.E.,Ph.D., Mrs. D.JENNIFER,M.E.,**

**HEAD OF THE DEPARTMENT SUPERVISOR**

DEPARTMENT OF CSE, DEPARTMENT OF CSE,

PANIMALAR ENGINEERING PANIMALAR ENGINEERING COLLEGE, COLLEGE,

NASARATHPETTAI, NASARATHPETTAI,

POONAMALLEE, POONAMALLEE,

CHENNAI-600 123. CHENNAI-600 123.

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

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**DECLARATION BY THE STUDENT**

We,**ANANDHALAKSHMIP[211418104019],GOWTHAMID[211418104071],HARITHA R[21141810480]** hereby declare that this project report titled “**EVALUATING AND CLASSIFYING WATER QUALITY USING MACHINE LEARNING**”,under the guidance of **Mrs. D.JENNIFER, M.E.,** is the orginial work done by us and we have not plagiarized or submitted to any other degree in any university by us.

**1.ANANDHA LAKSHMI P**

**2.GOWTHAMI D**

**3.HARITHA R**

**ACKNOWLEDGEMENT**

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

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

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

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

We would like to thank my **Project Guide** **Mrs.D. JENNIFER,M.E.,** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

##### ANANDHA LAKSHMI P

**GOWTHAMI D**

**HARITHA R**

**ABSTRACT**

Water pollution can be described as one of the most dangerous threats that the humanity ever faced. Great damage is caused to crops ,animals, forests etc. To prevent this problem we have to predict water quality from pollutants using machine learning techniques. Hence, water quality evaluation and prediction has become an important research area. The aim is to investigate machine learning based techniques for water quality forecasting by predicting results in best accuracy. Our analysis provides a comprehensive guide to sensitivity analysis of model parameters with regard to performance in prediction of water quality by accuracy calculation. To propose a machine learning-based method to accurately predict the Water Quality Index value by prediction results in the form of best accuracy from comparing supervised classification machine learning algorithms. Additionally, to compare and discuss the performance of various machine learning algorithms from the given water supply department dataset with evaluation classification report. Precision, Recall, and F1 Score are used to determine the success of the proposed machine learning algorithm technique.

**TABLE OF CONTENTS:**

**S.NO TITLE PG.NO**

**ABSTRACT** v

**LIST OF FIGURES** viii

**LIST OF ABBREVIATIONS** viii

**1 INTRODUCTION**

1.1 Overview 1

1.2 Problem Definition 1

**2 LITERATURE SURVEY** 3

**3 SYSTEM ANALYSIS** 8

3.1 Existing System 9

3.2 Proposed System 9

3.3 Feasibility Study 10

3.4 Hardware Environment 12

3.5 Software Environment 12

**4 SYSTEM DESIGN** 13

4.1 ER Diagram 14

4.2 Data flow diagram 14

4.3 UML Diagrams 15

**5 SYSTEM ARCHITECTURE**  17

5.1Module Design Specification 18

5.2 Algorithms 23

**6 SYSTEM IMPLEMENTATION** 29

6.1 Server side coding 30

6.2 Client side coding 52

**7 PERFORMANCE ANALYSIS** 54

7.1 Results & Discussion 55

7.2 Performance Analysis 55

**8 CONCLUSION AND FUTURE ENHANCEMENTS** 61

**APPENDICES**

A.1 Sample Screens 62

**REFERENCES** 64

**LIST OF FIGURES**

**FIG NO FIGURE DESCRIPTION PG NO**

4.1 ER- DIAGRAM 14

4.2 DATA FLOW DIAGRAM 14

4.3 USE CASE DIAGRAM 15

4.4 CLASS DIAGRAM 15

4.5 ACTIVITY DIAGRAM 16

4.6 SEQUENCE DIAGRAM 16

5.1 SYSTEM ARCHITECTURE 18

5.2 MODULE DIAGRAM OF PRE-PROCESSING 19

5.3 MODULE DIAGRAM OF DATA VISUALISATION 21

A.1 OUTPUT SCREEN 62

A.2 ENTERING VALUE 62

A.3 PREDICTING WATER QUALITY 63

**LIST OF ABBREVIATIONS**

**S.NO ABBREVIATION EXPANSION**

1 ML Machine Learning

2 WQM Water Quality Monitoring

3 WQI Water Quality Index

4 LR Logistic Regression

5 RF Random Forest

6 DT Decision Tree

7 SVM Support Vector Machine

8 PCB Physical-chemical-Biological

9 TP True Positive

10 TN True Negative

11 FP False Positive

12 FN False Negative

**CHAPTER- 1**

**1.INTRODUCTION**

**1.1 OVERVIEW**

One of the biggest fears of the world is water pollution. The of water needs to be monitored in real time to supply a safe drinking water. In the 21st century a lot of inventions have been developed but at the same time global warming ,pollutions are also being formed, which has lead to unsafe drinking water for the worlds population. Nowadays, water quality monitoring in real time faces challenges because of growing population, and limited water resources, growing population, etc. Therefore better methods need to be developed to predict the water quality whether it is safe or unsafe for drinking.

**1.2 PROBLEM DEFINITION**

Water pollution can be described as one of the most dangerous threats that the humanity ever faced. Great damage is caused to crops ,animals, forests etc. To prevent this problem we have to predict water quality from pollutants using machine learning techniques. Hence, water quality evaluation and prediction has become an important research area. The aim is to investigate machine learning based techniques for water quality forecasting by predicting results in best accuracy. Our analysis provides a comprehensive guide to sensitivity analysis of model parameters with regard to performance in prediction of water quality by accuracy calculation. To propose a machine learning-based method to accurately predict the Water Quality Index value by prediction results in the form of best accuracy from comparing supervised classification machine learning algorithms. Additionally, to compare and discuss the performance of various machine learning algorithms from the given water supply department dataset with evaluation classification report. Precision, Recall, and F1 Score are used to determine the success of the proposed machine learning algorithm technique.

**CHAPTER - 2**

**2**. **LITERATURE SURVEY**

**1.Title:** Comparison of Water Quality Classification Models using Machine Learning

**Author**: Neha Radhakrishnan; Anju S Pillai

**Year**: July 2021

**Published in**: Fifth International Conference on Communication and Electronics Systems

**Observation:** In this paper, machine learning methods such as SVM, Decision Tree, and Nave Bayes are used to compare water quality classification models. pH, DO, BOD, and electrical conductivity are among factors examined when determining water quality. The water quality was determined using only four factors. Only three methods are compared, resulting in lower accuracy. As a result, the decision tree algorithm was discovered to be a better classification model, with a 98.50 percent accuracy.

**2. Title: Water quality prediction and classification based on principal component regression and gradient boosting classifier**

**Year:** June2021

**Author:** Md.Saikat islam khan

**Published in :** Journal of King Saud University-Computer and Information Science

**Observation:** Gradient Boosting Regression, Multiple Linear Regression, and Support Vector Regression are three machine learning techniques used in this work to compare water quality classification models. The data set is separated into two parts: 75 percent for training and 25% for testing. WQI is calculated using nine parameters (PH, DO, TDS, Chloride, COD, EC, SS, Turbidity, Alkalinity). As a result, Support Vector Regression produces excellent results.

**3.Title: Prediction of groundwater quality indices using Machine Learning algorithms**

**Year:** Dec 2021

**Authors:** Hemant Raheja, Arun Goel, Mahesh Pal

**Published in:** International journal of machine learning and computing **Observation:**The number of parameters considered in this study was 12. The data set is separated into two parts: 75 percent for training and 25% for testing. Two indices are used to calculate water quality. Entropy water quality index and Water quality index are the two. EWQI is efficient as a result of two indexes. Gradient Boosting Model, DNN, and XGBoost are the algorithms employed in this paper. Some user-defined parameter is required by this algorithm. Correlation coefficient, Root mean square error, and Index of agreement are examples of output parameters. As a result, DNN generates great accuracy.

**4.Title: Research on water quality prediction method based on AE-LSTM**

**Author :** Huizinga Zhang, Kimie Jinx

**Year** : October 2021

**Published in:** 5th International Conference on Automation, Control Engineering

**Observation:**  A water quality parameter prediction technique based on automatic encoder (AE) spatial property reduction and long and short time memory (LSTM) neural networks is proposed. The Lang Fang Water Quality Automatic Observation Station data collection is used to assess the tactic's efficacy. The strategy is discovered to have higher forecast accuracy and toughness by estimating total phosphorus (TP) and total atomic number 7 (TN) concentrations. Results, The prediction model with AE-LSTM input had a higher prediction impact and accuracy than the prediction model with LSTM input, and it could effectively forecast water quality parameters.

**5.Title: Water quality analysis in a lake using deep learning methodology**

**Year**: Aug 2020

**Author**: P.Senthil Kumar, Prasannamedha G, Soumya K

**Published in**: International journal of environmental analytical chemistry

**Observation:** Korattur Lake in Chennai was used to collect samples. This paper's parameter was 9. (PH ,TDS ,COD ,Nitrate ,Iron ,Sodium ,Phosphate ,Turbidity ,chloride ). Artificial Neural Networks, Recurrent Neural Networks, and Long Short Term Memory are the algorithms employed. As a result, LSTM has the highest accuracy, around 94%.

**6. Title: Predicting Water Quality Parameters Using Machine Learning**

**Author**: Nikhil M Ragi; Ravishankar Holla; G Manju

**Year** : March 2020

**Published in:** 4th International Conference on Recent Trends on Electronics, Information, Communication Technology (RTEICT)

**Observation:** ANN (Artificial Neural Network) is used. This method eliminates the use of chemicals in the evaluation of water quality indicators and is also cost effective. This paper presents a transient methodology for predicting unknown parameters such as pH, chloride, and sulphate values using well-known parameters such as pH scale, electrical physical phenomena, and TDS, as well as the Levenberg-Marquardt algorithmic programme, which aids in the classification of water bodies for various applications. In forecasting chloride, total-hardness, sulphate, and total alkalinity, the accuracy was 83.94 percent, 87.9%, 81.736 percent, and 79.48 percent, respectively.

**7.Title:** Ground Water Quality Prediction using Machine Learning Algorithms

**Author :** S.Vijay & Dr.K.Kamaraj

**Year:** March 01, 2019

**Published in:** International Journal of Research and Analytical Reviews

**Observation:** It examines the physico-chemical properties of ground water quality in the Vellore district towns of Ranipet, Arcot, and Walljah Pet. For the objective of investigating the quality of groundwater, water samples were obtained from various designated bore wells. In the Vellore district, there are two major types of water contamination: high and low. Water quality metrics like PH, TDS, EC, Chloride, Sulphate, Nitrate, Carbonate, Bicarbonate, metal ions, and trace elements have all been calculated. Water quality prediction algorithms employing Machine Learning classifier algorithm C5.0, Nave Bayes, and Random Forest as leaner. With accuracy and classification error, Nave Bayes and Random Forest generated better results.

**8.Title:** Predictive Analysis of Water Quality Parameters using Deep Learning

**Author**: Archana Solanki, Himanshu Agrawal, Kanchan Khare

**Year**: September 2019

**Published in**: International journal of computer application

**Observation**: When compared to other techniques, deep learning algorithms that use unsupervised learning provide proper results. Stack denoising autoencoder, deep belief network, regression, and multilayer perception are examples of algorithms. The results show that toughness can be achieved using a denoising autoencoder and a deep belief network, as well as successfully managing information variability. The benefit of unsupervised learning algorithms is assessed using metrics such as mean absolute error and mean square error to examine the prediction error rate. Area unit pH scale, dissolved oxygen, and turbidity were employed as parameters. Modules include data collecting (krishna stream), data preparation, and modelling.

**CHAPTER-3**

**3. SYSTEM ANALYSIS**

**3.1 EXISTING SYSTEM**

The sensor technology for WQM has improved. The cost-effective sensorised tools have been implemented which measures the PCB values are available and are deployed on boats ,buoys ,ships. The connected sensors technology has been a bridging solution for the disconnect between data quality, data gathering and data analysis and enhances the global data inter-comparability. They has reviewed the sensor deployment strategies, key deployment strategies, emerging methods for data analysis.

EXISTING SYSTEM DRAWBACKS

* Sensor technology is being used
* It is expensive to deploy
* No machine learning concept is used
* No accuracy is found

**3.2 PROPOSED SYSTEM:**

The proposed method is to build a machine learning model for water quality. Data collecting include gathering historical data on water quality. Data mining is a technique used for processing enormous data in the domain If discovered before treatment, the water can save lives . Machine learning concept is applied in this system which reduces the manual effort to make a better model which is error less. Data analysis and data visualization are being done on the dataset . Then machine learning algorithms are applied on the dataset. After applying the algorithms each one produces different accuracy. Finally the algorithm with highest accuracy is selected and deployed.

**Advantages:**

* Machine learning concept is used.
* Various algorithms are used.
* Highest accuracy is attained

**3.3 FEASIBILITY STUDY**

**3.3.1 SOCIAL FEASIBILITY**

Water pollution is one of the greatest threats to the humanity which destroys animals, plants, crops ,environment etc. To prevent this problem various machine learning techniques have to be developed to predict the water quality. The aim is to predict the water quality using various ML algorithms and predict their highest accuracy. Hence this system predicts the water quality and predicts whether the water is safe to drink or not.

**3.3.2 TECHNICAL FEASIBILITY**

**The Requirements of this system are:**

Operating System : Windows 10

Tool : Anaconda with Jupyter Notebook

Processor : intel i5

Hard disk : 500 GB

RAM : 8 GB

* **Python :**
* It's a multipurpose language. It's utilised for a wide range of purposes, including software development and machine learning apps.
* It is simple to learn
* It has a legible syntax.

• Python's features include flexibility, readability, scalability, portability, and speed.

* Flask: Flask is a Python-based microweb framework. The flask framework is used to deploy this system.
* As a result, this system is technically viable.
* **USED PACKAGES:**
* **sklearn:**
* Sklearn is a machine learning library for Python that includes many machine learning methods.
* **NumPy:**
* It's a python module that provides quick math functions for calculations.

It's utilised to read information.**NumPy:**

* It is a numeric python module which provides fast maths functions for calculations.
* It is used to read data in numpy arrays and for manipulation purpose.
* **Pandas**
* Pandas can read and write a variety of files, and data frames make data processing simple.
* **Matplotlib:**
* Data visualisation is an effective tool for identifying trends in a dataset.
* Data frames make it simple to manipulate data.

**3.3.3 ECONOMIC FEASIBILITY**

* Total number of lines of code (LOC)=1200K
* KLOC=1200/1000=1.2
* Effort = 2.4 (1.2)1.05 =>2.906 person-month
* Development time = 2.5(2.906)0.38=>3.747months
* Average staff size=2.906/3.747 =>0.775 person
* Productivity = 1.2/2.906 > 0.412 KLOC/person-month
* Hence this system is economically feasible.

**3.4 HARDWARE REQUIREMENTS:**

Processor : intel i5

Hard disk : 500 GB

RAM : 8 GB

**3.5 SOFTWARE REQUIREMENTS:**

Operating System : Windows 10

Tool : Anaconda with Jupyter Notebook

**CHAPTER - 4**

**4. SYSTEM DESIGN**

**4.1 E-R DIAGRAM**

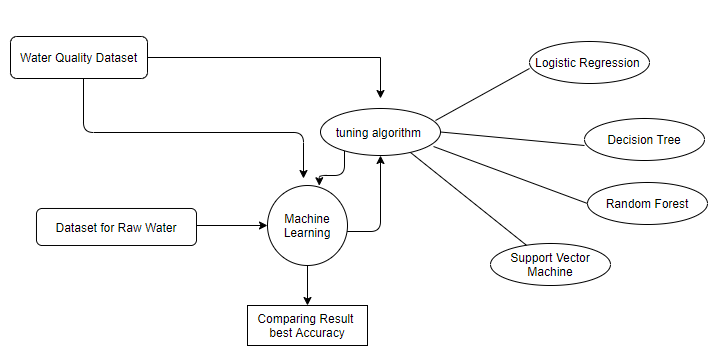


FIG 4.1 E-R DIAGRAM

**4.2 DATA FLOW DIAGRAM:**

Source Data

Data Processing and Cleaning

Testing Dataset

Training Dataset

Best Model by Accuracy

Classification ML Algorithms

Finding Water quality

FIG 4.2 DATA FLOW DIAGRAM

**4.3 USE CASE DIAGRAM:**

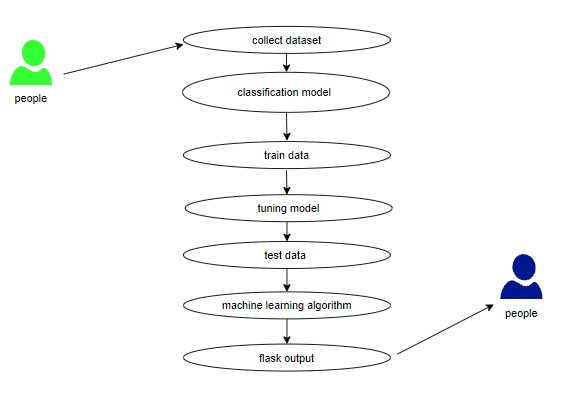


FIG 4.3 USE CASE DIAGRAM

**4.4 CLASS DIAGRAM:**

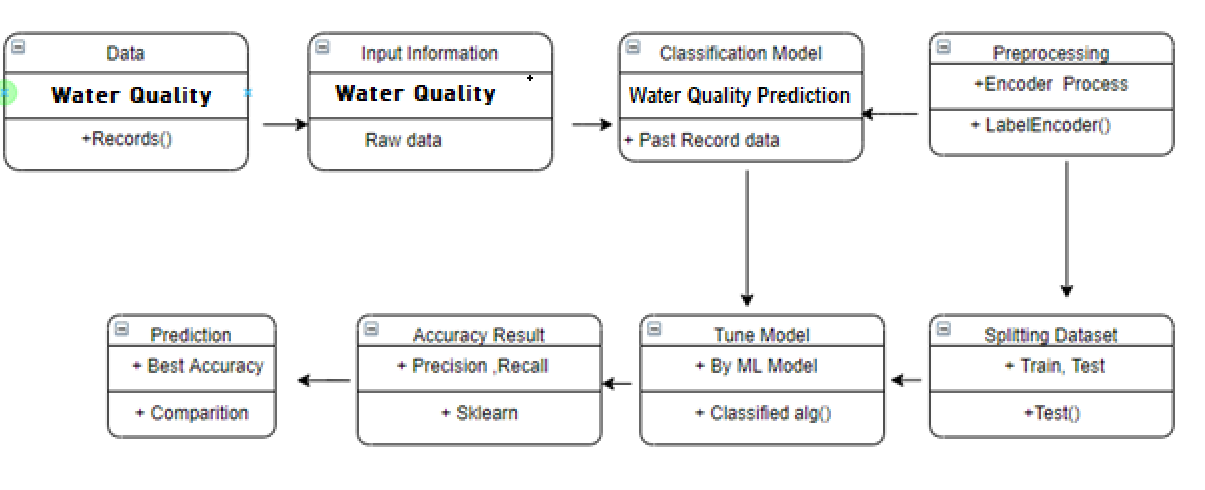


FIG 4.4 CLASS DIAGRAM

**4.5 ACTIVITY DIAGRAM:**

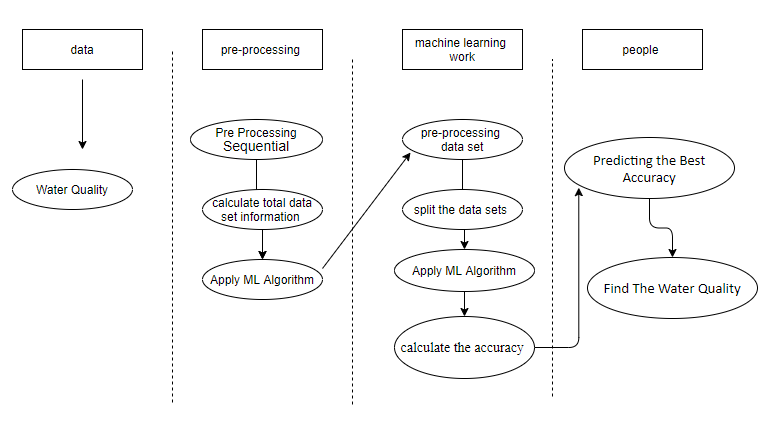


FIG 4.5 ACTIVITY DIAGRAM

**4.6 SEQUENCE DIAGRAM:**

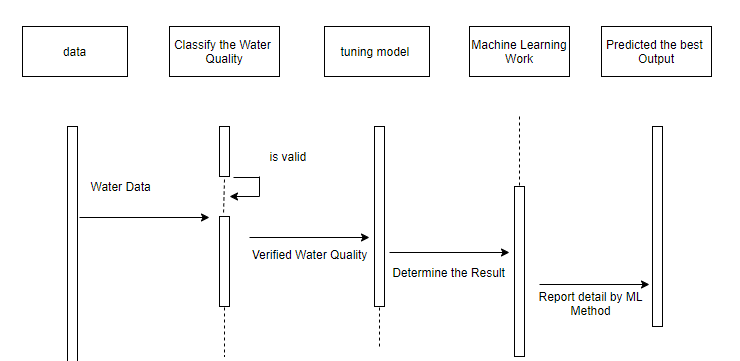


FIG 4.6 SEQUENCE DIAGRAM

**CHAPTER -5**

**5. SYSTEM ARCHITEC****TURE**

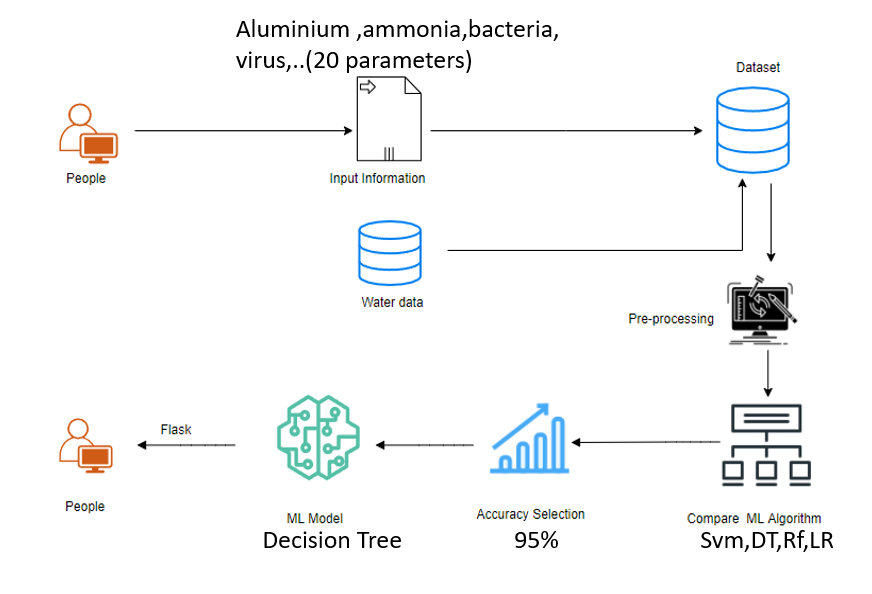


FIG 5.1 SYSTEM ARCHITECTURE

Water pollution is one of the greatest fears of the world. So as to provide a safe drinking water there is a need to predict the quality of water in real time. But, since the real time water monitoring faces challenges due to growing pollution limited water resources etc. There is a need to develop new and better technologies to predict the water quality parameters.

**5.1 MODULE DESIGN SPECIFICATION**

**5.1.1 DATA PRE-PROCESSING**

Validation techniques in ML is used to get the error rate which is close to true error rate pf the dataset. If the volume of the data is too large the validation techniques are not needed. This is concerned with locating the missing or duplicate value.Datatype is found-that is whether integer or float. Fine tuning of model hyper parameters are finetuned by ML engineers. Data gathering, data analysis, and the process of dealing with data content, quality, and organisation can be time-consuming. Understanding data and its properties is done during the data identification step, which aids in the selection of an algorithm to build the model.

A number data is cleaned by pythons pandas library, It focuses specifically on biggest data cleaning task , missing values, it spends less time cleaning the data and more time exploring and modelling.

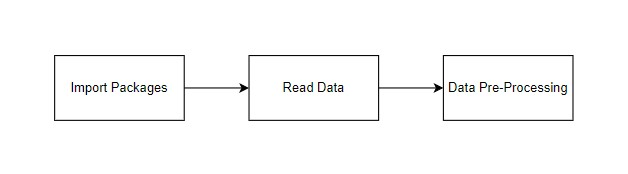
**MODULE DIAGRAM:**

FIG 5.2 MODULE DIAGRAM

**5.1.2 DATA VALIDATION OR DATA CLEANING PROCESS**

The specified dataset is being loaded. The library packages are being imported. Variables are identified by data shape and type, and missing values and duplicate values are evaluated. A validation dataset is a sample of data kept back from training your model that is used to measure model skill when tweaking models and techniques for making the most of validation and test datasets while evaluating your models. Drop the column and rename the dataset. The steps for cleaning data vary depending on the dataset. The main purpose is to find and fix problems so that data may be used for better decision-making and analytics.

**5.1.3 DATA VISUALISATION**

It is an important skill in machine learning and applied statistics. Statistics focuses on estimating data and quantitative descriptions. Data visualisation provides qualitative comprehension tools that are useful for comprehending datasets, exploring datasets, and spotting outliers, patterns, and corrupt data, among other things. it is used to express the key relationship in charts plots .

Expressing the data in visual form makes it more understandable. Visualising data samples is an important skill in both applied ML and applied statistics. It will discover the different plots that will be known while visualising data in python and how to use them for better understanding of data.

* How to use line plots to visualise time series data and bar charts to visualise categorical variables.

• How to summarise data distributions with histograms and box plots

Pre-processing refers to the adjustments made to our data before it is fed into the algorithm.. Data pre-processing is used to clean the raw dataset . In other words, In other words, anytime data is acquired from various sources, it is obtained in raw format, which makes analysis impossible.. The data has to be in better manner for achieving better results. Some Machine Learning models require data in a specific format; for example, the Random Forest(RF) algorithm does not tolerate null values. To implement RF algorithm null values have to be eliminated from the original raw dataset. Dataset should be formatted in such a way that more than one machine learning and deep learning algorithms are executed using the same dataset.

**False Positives (FP)** occur when the actual class is not the same as the projected class.

**False Negatives (FN)** are situations in which the real class is yes but the expected class is no.

**True Positives (TP**) are accurately predicted positive values, indicating that the value of the actual class and the value of the projected class are both yes.

**True Negatives (TN):** These are correctly predicted negative values, indicating that the actual class value is no and the predicted class value is also no.

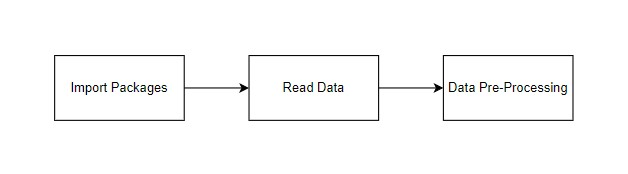
**MODULE DIAGRAM:** 

FIG 5.3 MODULE DIAGRAM

**5.1.4 COMPARING ALGORITHMS**

It is necessary to compare the performance of different machine learning algorithms which creates a test harness to compare different machine learning algorithms in python using scikit-learn. This test harness is used as a template to build your own ML problems and is able to add more different algorithms to compare. Different models have different performance characteristics. Resampling methods like cross validation is used to estimate accuracy of each model on unseen data. This estimate is used to choose one or two best models. For a new dataset it is good idea to visualize the data using different techniques to look at the data from different perspectives. The same idea applies to model selection. Different visualisation methods are used to show variance, average accuracy and other properties of the model accuracies. Each algorithm is tested in the same way on the same data, which can be accomplished by requiring each algorithm to be tested with the same test harness. In this system 4 different algorithms are compared:

* Logistic Regression
* Decision Tree
* Random Forest
* Support Vector Machine

Machine Learning Model is built using install Scikit-Learn libraries. This library package has to do pre-processing, linear model with logistic regression method, cross validating by K-Fold method, ensemble with random forest method and tree with decision tree classifier. Then it splits the data into train set and test set then finally predicting the result by comparing accuracy.

**5.1.5 PREDICTING RESULT BY ACCURACY**

To predict a value, the logistic regression process uses a linear equation with independent predictors. The anticipated value ranges from negative infinity to positive infinity. Logistic regression is a high-accuracy model that is created by comparing the best accuracy

True Positive Rate(TPR) = TP / (TP + FN)

False Positive rate(FPR) = FP / (FP + TN)

**Accuracy**: Accuracy is the percentage of right predictions out of a total number of forecasts.

Accuracy = (TP + TN) / (TP + TN + FP + FN)

It's only the proportion of accurately anticipated observations to total observations. Only when the dataset is symmetric and the values of false negative and false positive are almost equal is accuracy a good measure.

**Precision:** Precision refers to the percentage of positive predictions that are correct.

Precision = TP / (TP + FP)

The ratio of accurately anticipated positive observations to total predicted positive observations is known as precision. The low false positive rate is related to high precision. We have a precision of 0.788, which is rather good.

**Recall:** Recall that the proportion of positive observed values that was correctly predicted is the proportion of positive observed values that was correctly predicted.

Recall = TP / (TP + FN)

Recall(Sensitivity) Yes, recall (Sensitivity) is the ratio of accurately predicted positive observations to all observations in the actual class.

**F1 Score** The weighted average of Precision and Recall is the F1 Score. As a result, this score considers both false positives and false negatives. When the class distribution is unequal, F1 is frequently more valuable than accuracy. When false positives and false negatives have equivalent costs, accuracy works well. If the costs of a false negative and a false positive are significantly different, it's wiser to consider both Precision and Recall.

**General Formula:**

F- Measure = 2TP / (2TP + FP + FN)

**F1-Score Formula:**

F1 Score = 2\*(Recall \* Precision) / (Recall + Precision)

**5.2 ALGORITHM:**

**5.2.1 LOGISTIC REGRESSION:**

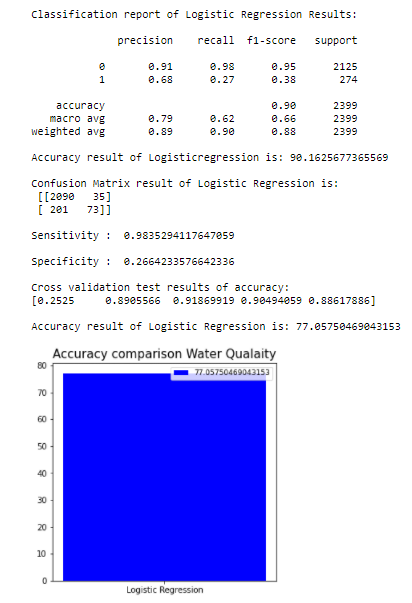
It's a statistical technique for analysing a data set with one or more independent factors that influence the outcome. A dichotomous variable is used to assess the outcome (in which there are only two possible outcomes). The purpose of logistic regression is to identify the best-fitting model to represent the relationship between a set of independent (predictor or explanatory) factors and a dichotomous feature of interest (dependent variable = response or outcome variable). A Machine Learning classification approach called logistic regression is used to predict the likelihood of a categorical dependent variable. The dependent variable in logistic regression is a binary variable that comprises data coded as 1 (yes, success, etc.) or 0 (no) (no, failure, etc.). 

FIG 5.4 LOGISTIC REGRESSION RESULT

**5.2.2 DECISION TREE:**

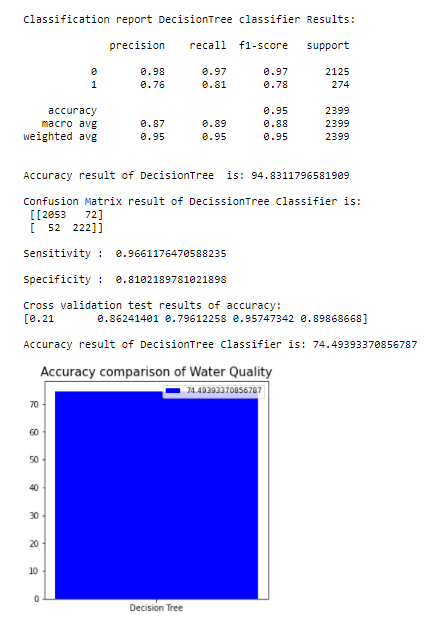
Decision tree analysis is a prophetic modelling fashion that can be used in a variety of situations. An algorithmic strategy that can resolve the dataset in multitudinous ways grounded on different conditions can be used to produce decision trees. The most important algorithms in the sphere of supervised algorithms are decision trees. They can be used for bracket as well as retrogression. Decision bumps, where the data is resolve, and leaves, where we get the result, are the two main ingredientsof a tree.

FIG 5.5 DECISION TREE RESULT

IMPORTANT TERMS IN DECISION TREE

The root node symbolises the total population or sample, which is then separated into two or more homogeneous groups.

Decision Knot - When a sub-node splits into several sub-nodes, it's referred to as a decision knot.

Bumps that do not resolve are referred to as Leaf or Terminal knots.

Cutting - Pruning Pruning is the process of removing sub-nodes from a decision knot. Splitting in the opposite direction, as it were.

Branch/Sub-Tree A branch or sub-tree is a portion of the overall tree.

Knot between parent and child A parent knot of sub-nodes is a knot that is divided into sub-nodes, whereas sub-nodes are the children of a parent knot.

**5.2.3 RANDOM FOREST:**

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model. "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

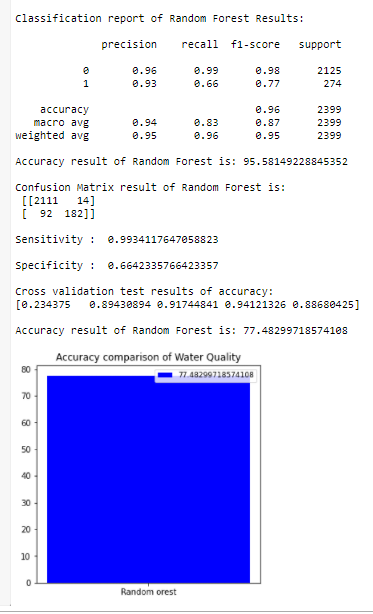


FIG 5.6 RANDOM FOREST RESULT

**5.2.4 SUPPORT VECTOR MACHINE:**

An SVM training algorithm creates a model that assigns new exemplifications to one of two orders, making it anon-probabilistic double direct classifier, given a series of training exemplifications that are collectively designated as belonging to one of two orders. The thing of using SVMs is to detect the stylish line in two confines or the stylish hyperplane in further than two confines to help us in classifying our space. The largest periphery, or the maximum distance between data points of both classes, is used to find the hyperplane (line).

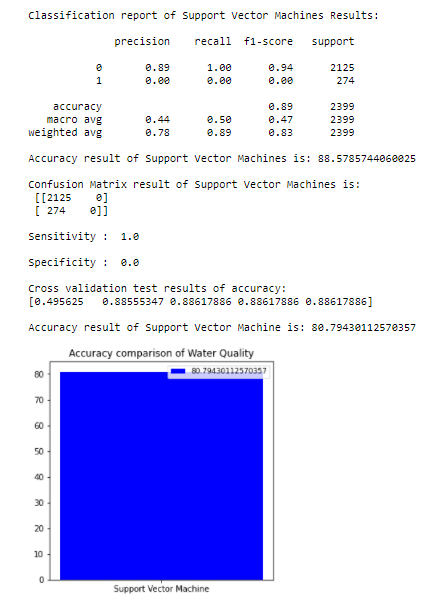


FIG 5.7 SUPPORT VECTOR MACHINE RESULT

**CHAPTER - 6**

**6 .SYSTEM IMPLEMENTATION**:

**6.1 SERVER SIDE CODING**

*#import package*

**import** pandas **as** pd

**import** numpy **as** ny

*#read dataset*

data **=** pd**.**read\_csv('water.csv')

*# view head*

data**.**head()

*#view tail*

data**.**tail()

*# shape of the project*

data**.**shape

*# size ofthe data*

data**.**size

*#columns*

data**.**columns

data**.**isnull()

data**.**isnull()**.**sum()

data**.**dropna()

*# information about dataset*

data**.**info()

data**.**duplicated()

data**.**duplicated()**.**sum()

*# describe*

data**.**describe()

data['arsenic']**.**nunique()

data['barium']**.**nunique()

data["uranium"]**.**nunique()

*# check the unique values*

data["aluminium"]**.**nunique()

data["viruses"]**.**nunique()

data["mercury"]**.**nunique()

data["silver"]**.**unique()

data["mercury"]**.**unique()

data["arsenic"]**.**unique()

data["uranium"]**.**unique()

data["chloramine"]**.**unique()

data["aluminium"]**.**unique()

data**.**head()

*#min max values*

print("minimum values of aluminium :", data["aluminium"]**.**min())

print("mean values of alumimum :",data["aluminium"]**.**mean())

print("maximum values of aluminium :", data["aluminium"]**.**max())

*#min max values*

print("minimum values of aluminium :", data["chloramine"]**.**min())

print("mean values of alumimum :",data["chloramine"]**.**mean())

print("maximum values of aluminium :", data["chloramine"]**.**max())

*#min max values*

print("minimum values of aluminium :", data["copper"]**.**min())

print("mean values of alumimum :",data["copper"]**.**mean())

print("maximum values of aluminium :", data["copper"]**.**max())

In [ ]:

*#min max values*

print("minimum values of aluminium :", data["flouride"]**.**min())

print("mean values of alumimum :",data["flouride"]**.**mean())

print("maximum values of aluminium :", data["flouride"]**.**max())

*#min max values*

print("minimum values of aluminium :", data["lead"]**.**min())

print("mean values of alumimum :",data["lead"]**.**mean())

print("maximum values of aluminium :", data["lead"]**.**max())

*#min max values*

print("minimum values of aluminium :", data["nitrates"]**.**min())

print("mean values of alumimum :",data["nitrates"]**.**mean())

print("maximum values of aluminium :", data["nitrates"]**.**max())

*#min max values*

print("minimum values of aluminium :", data["radium"]**.**min())

print("mean values of alumimum :",data["radium"]**.**mean())

print("maximum values of aluminium :", data["radium"]**.**max())

data**.**corr()

data**.**corr()**.**describe()

data["mercury"]**.**value\_counts()

pd**.**Categorical(data["uranium"])**.**describe()

data**.**info()

data**.**columns

**from** sklearn.preprocessing **import** LabelEncoder

le **=** LabelEncoder()

var\_mod **=** (['aluminium', 'ammonia', 'arsenic', 'barium', 'cadmium', 'chloramine',

'chromium', 'copper', 'flouride', 'bacteria', 'viruses', 'lead',

'nitrates', 'nitrites', 'mercury', 'perchlorate', 'radium', 'selenium',

'silver', 'uranium', 'is\_safe'])

**for** i **in** var\_mod:

data[i] **=** le**.**fit\_transform(data[i])**.** astype(int)

data**.**head()

# Model-2:

**import** pandas **as** pd

**import** numpy **as** np

**import** matplotlib.pyplot **as** plt

**import** seaborn **as** sns

**import** warnings

warnings**.**filterwarnings('ignore')

data **=** pd**.**read\_csv("water.csv")

data**.**head()

data["is\_safe"]**.**unique()

data["is\_safe"]**.**unique()

data[data["is\_safe"]**==**'#NUM!']

data **=** data**.**drop(data[data["is\_safe"]**==**'#NUM!']**.**index)

data["is\_safe"]**.**unique()

data["is\_safe"] **=** data["is\_safe"]**.**astype(float)

data**.**info()

data**.**columns

data**.**isnull()**.**sum()

data **=**data**.**dropna()

plt**.**figure(figsize**=** (10,8))

sns**.**heatmap(data**.**isnull())

plt**.**show()

plt**.**figure(figsize**=**(9,6))

sns**.**scatterplot(x**=**data['aluminium'],y**=**data["barium"])

plt**.**show()

plt**.**figure(figsize **=** (14,8))

sns**.**stripplot(x **=** data["cadmium"], y **=** data["chromium"])

data**.**columns

plt**.**figure(figsize**=** (12,8))

plt**.**subplot(2,2,1)

data["cadmium"]**.**plot(kind**=**'density')

plt**.**subplot(2,2,2)

data["copper"]**.**plot(kind**=**'density')

plt**.**subplot(2,2,3)

data["viruses"]**.**plot(kind**=**'density')

plt**.**subplot(2,2,4)

data["chromium"]**.**plot(kind**=**'density')

plt**.**show()

data**.**hist(figsize**=**(20,35),layout **=**(19,3) )

plt**.**show()

plt**.**figure(figsize**=**(12,4))

plt**.**subplot(1,2,1)

plt**.**title("mercury")

sns**.**boxplot(data["mercury"])

plt**.**subplot(1,2,2)

plt**.**title("perchlorate")

sns**.**boxplot(data["perchlorate"])

plt**.**show()

*#Propagation by variable*

**def** PropByVar(data, variable):

dataframe\_pie **=** data[variable]**.**value\_counts()

ax **=** dataframe\_pie**.**plot**.**pie(figsize**=**(10,10), autopct**=**'%1.2f%%', fontsize **=** 12)

ax**.**set\_title(variable **+** ' \n', fontsize **=** 15)

**return** np**.**round(dataframe\_pie**/**data**.**shape[0]**\***100,2)

PropByVar(data, 'is\_safe')

plt**.**figure(figsize**=**(15,12))

sns**.**heatmap(data**.**corr(), annot **=True**)

plt**.**show()

data**.**columns

**from** sklearn.preprocessing **import** LabelEncoder

le **=** LabelEncoder()

var **=** ['aluminium', 'ammonia', 'arsenic', 'barium', 'cadmium', 'chloramine',

'chromium', 'copper', 'flouride', 'bacteria', 'viruses', 'lead',

'nitrates', 'nitrites', 'mercury', 'perchlorate', 'radium', 'selenium',

'silver', 'uranium', 'is\_safe']

**for** i **in** var:

data[i] **=** le**.**fit\_transform(data[i])

data**.**head()

Module 3 : Performance measurements of Logistic regression

*#import library packages*

**import** pandas **as** p

**import** matplotlib.pyplot **as** plt

**import** seaborn **as** s

**import** numpy **as** n

*#Load given dataset*

data **=** p**.**read\_csv("water.csv")

**import** warnings

warnings**.**filterwarnings('ignore')

data**.**head(5)

data**.**tail(5)

data**.**isnull()**.**sum()

data **=** data**.**dropna()

data**.**shape

data**.**duplicated()**.**sum()

data**.**info()

data["is\_safe"]**.**unique()

data[data["is\_safe"]**==**'#NUM!']

data **=** data**.**drop(data[data["is\_safe"]**==**'#NUM!']**.**index)

data["is\_safe"]**.**unique()

data**.**describe()

data**.**corr()

df **=** data

df**.**columns

*#According to the cross-validated MCC scores, the random forest is the best-performing model, so now let's evaluate its performance on the test set.*

**from** sklearn.metrics **import** confusion\_matrix, classification\_report, matthews\_corrcoef, cohen\_kappa\_score, accuracy\_score, average\_precision\_score, roc\_auc\_score

X **=** data**.**drop(labels**=**'is\_safe', axis**=**1)

*#Response variable*

y **=** data**.**loc[:,'is\_safe']

*#We'll use a test size of 30%. We also stratify the split on the response variable, which is very important to do because there are so few fraudulent transactions.*

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

X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(X, y, test\_size**=**0.3, random\_state**=**0, stratify**=**y)

# Logistic Regression :

**from** sklearn.metrics **import** accuracy\_score, confusion\_matrix

**from** sklearn.linear\_model **import** LogisticRegression

**from** sklearn.model\_selection **import** cross\_val\_score

logR**=** LogisticRegression()

logR**.**fit(X\_train,y\_train)

predictLR **=** logR**.**predict(X\_test)

print("")

print('Classification report of Logistic Regression Results:')

print("")

print(classification\_report(y\_test,predictLR))

x **=** (accuracy\_score(y\_test,predictLR)**\***100)

print('Accuracy result of Logisticregression is:', x)

print("")

cm2**=**confusion\_matrix(y\_test,predictLR)

print('Confusion Matrix result of Logistic Regression is:\n',cm2)

print("")

sensitivity2 **=** cm2[0,0]**/**(cm2[0,0]**+**cm2[0,1])

print('Sensitivity : ', sensitivity2 )

print("")

specificity2 **=** cm2[1,1]**/**(cm2[1,0]**+**cm2[1,1])

print('Specificity : ', specificity2)

print("")

accuracy **=** cross\_val\_score(logR, X, y, scoring**=**'accuracy')

print('Cross validation test results of accuracy:')

print(accuracy)

*#get the mean of each fold*

print("")

print("Accuracy result of Logistic Regression is:",accuracy**.**mean() **\*** 100)

LR**=**accuracy**.**mean() **\*** 100

**def** graph():

**import** matplotlib.pyplot **as** plt

data**=**[LR]

alg**=**"Logistic Regression"

plt**.**figure(figsize**=**(5,5))

b**=**plt**.**bar(alg,data,color**=**("b"))

plt**.**title("Accuracy comparison Water Qualaity",fontsize**=**15)

plt**.**legend(b,data,fontsize**=**9)

graph()

TN **=** cm2[1][0]

FN **=** cm2[0][0]

TP **=** cm2[1][1]

FP **=** cm2[0][1]

print("True Positive :",TP)

print("True Negative :",TN)

print("False Positive :",FP)

print("False Negative :",FN)

print("")

TPR **=** TP**/**(TP**+**FN)

TNR **=** TN**/**(TN**+**FP)

FPR **=** FP**/**(FP**+**TN)

FNR **=** FN**/**(TP**+**FN)

print("True Positive Rate :",TPR)

print("True Negative Rate :",TNR)

print("False Positive Rate :",FPR)

print("False Negative Rate :",FNR)

print("")

PPV **=** TP**/**(TP**+**FP)

NPV **=** TN**/**(TN**+**FN)

print("Positive Predictive Value :",PPV)

print("Negative predictive value :",NPV)

cm2**=**confusion\_matrix(y\_test, predictLR)

print('Confusion matrix-LR:')

print(cm2)

s**.**heatmap(cm2**/**n**.**sum(cm2), annot**=True**, cmap **=** 'Blues', annot\_kws**=**{"size": 16}, fmt**=**'.2%',)

plt**.**show()

# Module 4 : Performance measurements of DecisionTree:

*#import library packages*

**import** pandas **as** p

**import** matplotlib.pyplot **as** plt

**import** seaborn **as** s

**import** numpy **as** n

*#Load given dataset*

data **=** p**.**read\_csv("water.csv")

**import** warnings

warnings**.**filterwarnings('ignore')

data**.**head(5)

data**.**tail(5)

data**.**isnull()**.**sum()

data **=** data**.**dropna()

data**.**shape

data**.**duplicated()**.**sum()

data["is\_safe"]**.**unique()

data[data["is\_safe"]**==**'#NUM!']

data **=** data**.**drop(data[data["is\_safe"]**==**'#NUM!']**.**index)

data["is\_safe"]**.**unique()

data**.**info()

data**.**describe()

data**.**corr()

df **=** data

df**.**columns

*#According to the cross-validated MCC scores, the random forest is the best-performing model, so now let's evaluate its performance on the test set.*

**from** sklearn.metrics **import** confusion\_matrix, classification\_report, matthews\_corrcoef, cohen\_kappa\_score, accuracy\_score, average\_precision\_score, roc\_auc\_score

X **=** data**.**drop(labels**=**'is\_safe', axis**=**1)

*#Response variable*

y **=** data**.**loc[:,'is\_safe']

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

X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(X, y, test\_size**=**0.3, random\_state**=**0, stratify**=**y)

# DecisionTree:

**from** sklearn.metrics **import** accuracy\_score, confusion\_matrix

**from** sklearn.tree **import** DecisionTreeClassifier

**from** sklearn.model\_selection **import** cross\_val\_score

DT**=**DecisionTreeClassifier()

DT**.**fit(X\_train,y\_train)

predictDT **=** DT**.**predict(X\_test)

print("")

print('Classification report DecisionTree classifier Results:')

print("")

print(classification\_report(y\_test,predictDT))

print("")

x **=** (accuracy\_score(y\_test,predictDT)**\***100)

print('Accuracy result of DecisionTree is:', x)

print("")

cm2**=**confusion\_matrix(y\_test,predictDT)

print('Confusion Matrix result of DecissionTree Classifier is:\n',cm2)

print("")

sensitivity2 **=** cm2[0,0]**/**(cm2[0,0]**+**cm2[0,1])

print('Sensitivity : ', sensitivity2 )

print("")

specificity2 **=** cm2[1,1]**/**(cm2[1,0]**+**cm2[1,1])

print('Specificity : ', specificity2)

print("")

accuracy **=** cross\_val\_score(DT, X, y, scoring**=**'accuracy')

print('Cross validation test results of accuracy:')

print(accuracy)

*#get the mean of each fold*

print("")

print("Accuracy result of DecisionTree Classifier is:",accuracy**.**mean() **\*** 100)

dt**=**accuracy**.**mean() **\*** 100

**def** graph():

**import** matplotlib.pyplot **as** plt

data**=**[dt]

alg**=**"Decision Tree"

plt**.**figure(figsize**=**(5,5))

b**=**plt**.**bar(alg,data,color**=**("b"))

plt**.**title("Accuracy comparison of Water Quality",fontsize**=**15)

plt**.**legend(b,data,fontsize**=**9)

graph()

TN **=** cm2[1][0]

FN **=** cm2[0][0]

TP **=** cm2[1][1]

FP **=** cm2[0][1]

print("True Positive :",TP)

print("True Negative :",TN)

print("False Positive :",FP)

print("False Negative :",FN)

print("")

TPR **=** TP**/**(TP**+**FN)

TNR **=** TN**/**(TN**+**FP)

FPR **=** FP**/**(FP**+**TN)

FNR **=** FN**/**(TP**+**FN)

print("True Positive Rate :",TPR)

print("True Negative Rate :",TNR)

print("False Positive Rate :",FPR)

print("False Negative Rate :",FNR)

print("")

PPV **=** TP**/**(TP**+**FP)

NPV **=** TN**/**(TN**+**FN)

print("Positive Predictive Value :",PPV)

print("Negative predictive value :",NPV)

cm2**=**confusion\_matrix(y\_test, predictDT)

print('Confusion matrix-DT:')

print(cm2)

s**.**heatmap(cm2**/**n**.**sum(cm2), annot**=True**, cmap **=** 'Blues', annot\_kws**=**{"size": 16}, fmt**=**'.2%',)

plt**.**show()

# Module 5 : Performance measurements of Random Forest algorithms

*#import library packages*

**import** pandas **as** p

**import** matplotlib.pyplot **as** plt

**import** seaborn **as** s

**import** numpy **as** n

*#Load given dataset*

data **=** p**.**read\_csv("water.csv")

**import** warnings

warnings**.**filterwarnings('ignore')

data**.**head(5)

data**.**tail(5)

data**.**isnull()**.**sum()

data **=** data**.**dropna()

data**.**shape

data**.**duplicated()**.**sum()

data["is\_safe"]**.**unique()

data[data["is\_safe"]**==**'#NUM!']

data **=** data**.**drop(data[data["is\_safe"]**==**'#NUM!']**.**index)

data["is\_safe"]**.**unique()

data**.**info()

data**.**describe()

data**.**corr()

df **=** data

df**.**columns

*#According to the cross-validated MCC scores, the random forest is the best-performing model, so now let's evaluate its performance on the test set.*

**from** sklearn.metrics **import** confusion\_matrix, classification\_report, matthews\_corrcoef, cohen\_kappa\_score, accuracy\_score, average\_precision\_score, roc\_auc\_score

X **=** data**.**drop(labels**=**'is\_safe', axis**=**1)

*#Response variable*

y **=** data**.**loc[:,'is\_safe']

*#We'll use a test size of 30%. We also stratify the split on the response variable, which is very important to do because there are so few fraudulent transactions.*

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

X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(X, y, test\_size**=**0.3, random\_state**=**0, stratify**=**y)

# Random Forest:

**from** sklearn.ensemble **import** RandomForestClassifier

**from** sklearn.metrics **import** accuracy\_score, confusion\_matrix

**from** sklearn.model\_selection **import** cross\_val\_score

rfc **=** RandomForestClassifier()

rfc**.**fit(X\_train,y\_train)

predictR **=** rfc**.**predict(X\_test)

print("")

print('Classification report of Random Forest Results:')

print("")

print(classification\_report(y\_test,predictR))

x **=** (accuracy\_score(y\_test,predictR)**\***100)

print('Accuracy result of Random Forest is:', x)

print("")

cm1**=**confusion\_matrix(y\_test,predictR)

print('Confusion Matrix result of Random Forest is:\n',cm1)

print("")

sensitivity1 **=** cm1[0,0]**/**(cm1[0,0]**+**cm1[0,1])

print('Sensitivity : ', sensitivity1 )

print("")

specificity1 **=** cm1[1,1]**/**(cm1[1,0]**+**cm1[1,1])

print('Specificity : ', specificity1)

print("")

accuracy **=** cross\_val\_score(rfc, X, y, scoring**=**'accuracy')

print('Cross validation test results of accuracy:')

print(accuracy)

*#get the mean of each fold*

print("")

print("Accuracy result of Random Forest is:",accuracy**.**mean() **\*** 100)

RFC**=**accuracy**.**mean() **\*** 100

**def** graph():

**import** matplotlib.pyplot **as** plt

data**=**[RFC]

alg**=**"Random orest"

plt**.**figure(figsize**=**(5,5))

b**=**plt**.**bar(alg,data,color**=**("b"))

plt**.**title("Accuracy comparison of Water Quality")

plt**.**legend(b,data,fontsize**=**9)

graph()

TN **=** cm1[0][1]

FN **=** cm1[1][1]

TP **=** cm1[0][0]

FP **=** cm1[1][0]

print("True Positive :",TP)

print("True Negative :",TN)

print("False Positive :",FP)

print("False Negative :",FN)

print("")

TPR **=** TP**/**(TP**+**FN)

TNR **=** TN**/**(TN**+**FP)

FPR **=** FP**/**(FP**+**TN)

FNR **=** FN**/**(TP**+**FN)

print("True Positive Rate :",TPR)

print("True Negative Rate :",TNR)

print("False Positive Rate :",FPR)

print("False Negative Rate :",FNR)

print("")

PPV **=** TP**/**(TP**+**FP)

NPV **=** TN**/**(TN**+**FN)

print("Positive Predictive Value :",PPV)

print("Negative predictive value :",NPV)

cm2**=**confusion\_matrix( predictR,y\_test)

print('Confusion matrix-RF:')

print(cm2)

s**.**heatmap(cm2**/**n**.**sum(cm2), annot**=True**, cmap **=** 'Blues', annot\_kws**=**{"size": 16},fmt**=**'.2%')

plt**.**show()

# Module 6 : Performance measurements of SVM

*#import library packages*

**import** pandas **as** p

**import** matplotlib.pyplot **as** plt

**import** seaborn **as** sns

**import** numpy **as** n

*#Load given dataset*

data **=** p**.**read\_csv("water.csv")

**import** warnings

warnings**.**filterwarnings('ignore')

data**.**head(5)

data**.**isnull()**.**sum()

data **=** data**.**dropna()

data**.**duplicated()**.**sum()

data["is\_safe"]**.**unique()

data[data["is\_safe"]**==**'#NUM!']

data **=** data**.**drop(data[data["is\_safe"]**==**'#NUM!']**.**index)

data["is\_safe"]**.**unique()

data**.**info()

df **=** data

df**.**columns

*#According to the cross-validated MCC scores, the random forest is the best-performing model, so now let's evaluate its performance on the test set.*

**from** sklearn.metrics **import** confusion\_matrix, classification\_report, matthews\_corrcoef, cohen\_kappa\_score, accuracy\_score, average\_precision\_score, roc\_auc\_score

X **=** data**.**drop(labels**=**'is\_safe', axis**=**1)

*#Response variable*

y **=** data**.**loc[:,'is\_safe']

*#We'll use a test size of 30%. We also stratify the split on the response variable, which is very important to do because there are so few fraudulent transactions.*

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

X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(X, y, test\_size**=**0.3, random\_state**=**0, stratify**=**y)

# svc

**from** sklearn.svm **import** SVC

**from** sklearn.metrics **import** accuracy\_score, confusion\_matrix

**from** sklearn.model\_selection **import** cross\_val\_score

s **=** SVC()

s**.**fit(X\_train,y\_train)

predicts **=** s**.**predict(X\_test)

print("")

print('Classification report of Support Vector Machines Results:')

print("")

print(classification\_report(y\_test,predicts))

x **=** (accuracy\_score(y\_test,predicts)**\***100)

print('Accuracy result of Support Vector Machines is:', x)

print("")

cm2**=**confusion\_matrix(y\_test,predicts)

print('Confusion Matrix result of Support Vector Machines is:\n',cm2)

print("")

sensitivity1 **=** cm2[0,0]**/**(cm2[0,0]**+**cm2[0,1])

print('Sensitivity : ', sensitivity1 )

print("")

specificity1 **=** cm2[1,1]**/**(cm2[1,0]**+**cm2[1,1])

print('Specificity : ', specificity1)

print("")

accuracy **=** cross\_val\_score(s,X, y, scoring**=**'accuracy')

print('Cross validation test results of accuracy:')

print(accuracy)

*#get the mean of each fold*

print("")

print("Accuracy result of Support Vector Machine is:",accuracy**.**mean() **\*** 100)

S**=**accuracy**.**mean() **\*** 100

**def** graph():

**import** matplotlib.pyplot **as** plt

data**=**[S]

alg**=**"Support Vector Machine"

plt**.**figure(figsize**=**(5,5))

b**=**plt**.**bar(alg,data,color**=**("b"))

plt**.**title("Accuracy comparison of Water Quality")

plt**.**legend(b,data,fontsize**=**9)

graph()

TN **=** cm2[1][0]

FN **=** cm2[0][0]

TP **=** cm2[1][1]

FP **=** cm2[0][1]

print("True Positive :",TP)

print("True Negative :",TN)

print("False Positive :",FP)

print("False Negative :",FN)

print("")

TPR **=** TP**/**(TP**+**FN)

TNR **=** TN**/**(TN**+**FP)

FPR **=** FP**/**(FP**+**TN)

FNR **=** FN**/**(TP**+**FN)

print("True Positive Rate :",TPR)

print("True Negative Rate :",TNR)

print("False Positive Rate :",FPR)

print("False Negative Rate :",FNR)

print("")

PPV **=** TP**/**(TP**+**FP)

NPV **=** TN**/**(TN**+**FN)

print("Positive Predictive Value :",PPV)

print("Negative predictive value :",NPV)

cm2**=**confusion\_matrix(y\_test, predicts)

print('Confusion matrix-SVM:')

print(cm2)

sns**.**heatmap(cm2**/**n**.**sum(cm2), annot**=True**, cmap **=** 'Blues', annot\_kws**=**{"size": 16}, fmt**=**'.2%',)

plt**.**show()

**6.2 CLIENT SIDE CODING**

import numpy as np

from flask import Flask, request, jsonify, render\_template

import pickle

import joblib

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

model = joblib.load('dt.pkl')

@app.route('/')

def home():

return render\_template('index.html')

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

def predict():

'''

For rendering results on HTML GUI

'''

int\_features = [(x) for x in request.form.values()]

final\_features = [np.array(int\_features)]

print(final\_features)

prediction = model.predict(final\_features)

output = prediction[0]

print(output)

if output == '0':

output ='water is not safe'

elif output =='1':

output = 'Water is safe'

return render\_template('index.html', prediction\_text='WATER QUALITY {}'.format(output))

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

app.run(host="localhost", port=8012)

**CHAPTER - 7**

**7. PERFORMANCE ANALYSIS**

**7.1 RESULTS AND DISCUSSIONS**

The proposed model was used to prognosticate the water quality. For this purpose the dataset mentioned before was used to estimate the model performance to find out the stylish one, comparisons with RF, LR, SVC, DT were displayed. The dataset was resolve into 70 and 30for training and testing purposes independently. For the results performance of 4 algorithms were compared in which Decision Tree has attained the loftiest delicacy of 96 when compared to 3 other algorithms. Eventually the system is stationed as a web operation. This system predicts the water quality status as safe or unsafe for drinking.

**7.2 PERFORMANCE ANALYSIS**

Evaluation Criteria are used to measure the quality of the machine literacy model. Assessing machine literacy models or algorithms is essential for any design. By using evaluation criteria, we can insure that the model is operating rightly and optimally.

It's veritably important to use multiple evaluation criteria to estimate your model.

This is because a model may perform well using one dimension from one

evaluation metric, but may perform inadequately using another dimension from

another evaluation metric. There are numerous different types of evaluation criteria

available to test a model. They're confusion matrix, delicacy, perfection, recall,

F1- score, false positive rate, Receiver driver characteristics wind (ROC),

Precision-Recall (PR) wind, Logarithmic loss, mean absolute error and root

mean squared error.

In this design, we include confusion matrix and bracket delicacy.

A confusion matrix gives us a matrix as affair and describes the complete

performance of the model.

Bracket of a test dataset produces four issues – true positive, false

positive, true negative, and false negative.

True positive (TP) correct positive vaticination

False positive (FP) incorrect positive vaticination

True negative( TN) correct negative vaticination

False negative (FN) incorrect negative vaticination

Bracket delicacy is the rate of the number of correct prognostications to the total number of input samples, which is generally what we relate to when we use the term delicacy. The stylish delicacy is1.0, whereas the worst is0.0. It can also be calculated by 1 – ERR.

Delicacy is calculated as the total number of two correct prognostications

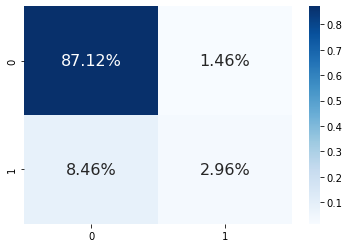
(TP TN) divided by the total number of a dataset (P N).

Delicacy = (TP TN)/ (TP TN FN FP)

Delicacy = (TP TN)/ (P N)

**7.2.1 CONFUSION MATRIX:**

LOGISTIC REGRESSION:



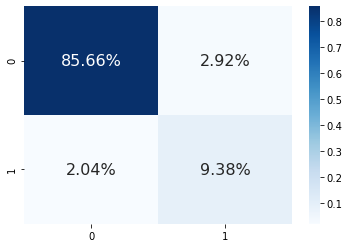
True Positive Rate : 0.03285515964831097

True Negative Rate : 0.8529411764705882

False Positive Rate : 0.14705882352941177

False Negative Rate : 0.967144840351689

DECISION TREE:



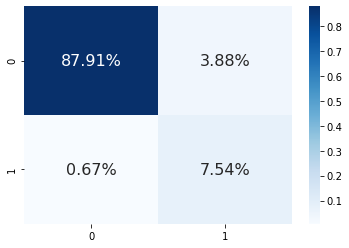
True Positive Rate : 0.09868421052631579

True Negative Rate : 0.4117647058823529

False Positive Rate : 0.5882352941176471

False Negative Rate : 0.9013157894736842

RANDOM FOREST:



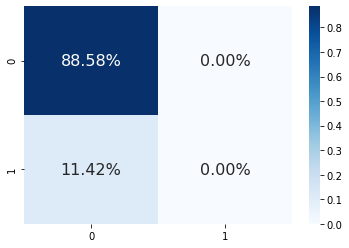
True Positive Rate : 0.9209606986899563

True Negative Rate : 0.14678899082568808

False Positive Rate : 0.8532110091743119

False Negative Rate : 0.07903930131004366

SUPPORT VECTOR MACHINE:



True Positive Rate : 0.0

True Negative Rate : 1.0

False Positive Rate : 0.0

False Negative Rate : 1.0

**7.2.2 ACCURACY**

|  |  |
| --- | --- |
| METHOD | ACCURACY |
| Logistic regression | 90% |
| Decision Tree | 95% |
| Random Forest | 95% |
| Support Vector Machine | 88% |

The above table shows Decision Tree and Random Forest has maximum accuracy of 95% hence decision tree is deployed used flask framework.

**CHAPTER - 8**

**8. CONCLUSION AND FUTURE ENHANCEMENT**

In this system the analytical process started from data cleaning and processing , finding the missing values, exploratory analysis of data ,applying different algorithms and finally model building and evaluation and finding the highest accuracy. The best accuracy on public test set is the highest accuracy score which will be found out. This application is helpful in predicting Water Quality status. As a future enhancement Water Quality prediction will be done using a using AI model. This process can be automated by showing the prediction result in web application or desktop application. The work can be optimized and implemented in Artificial Intelligence environment.

**APPENDICES**

**A.1 SAMPLE SCREENS**

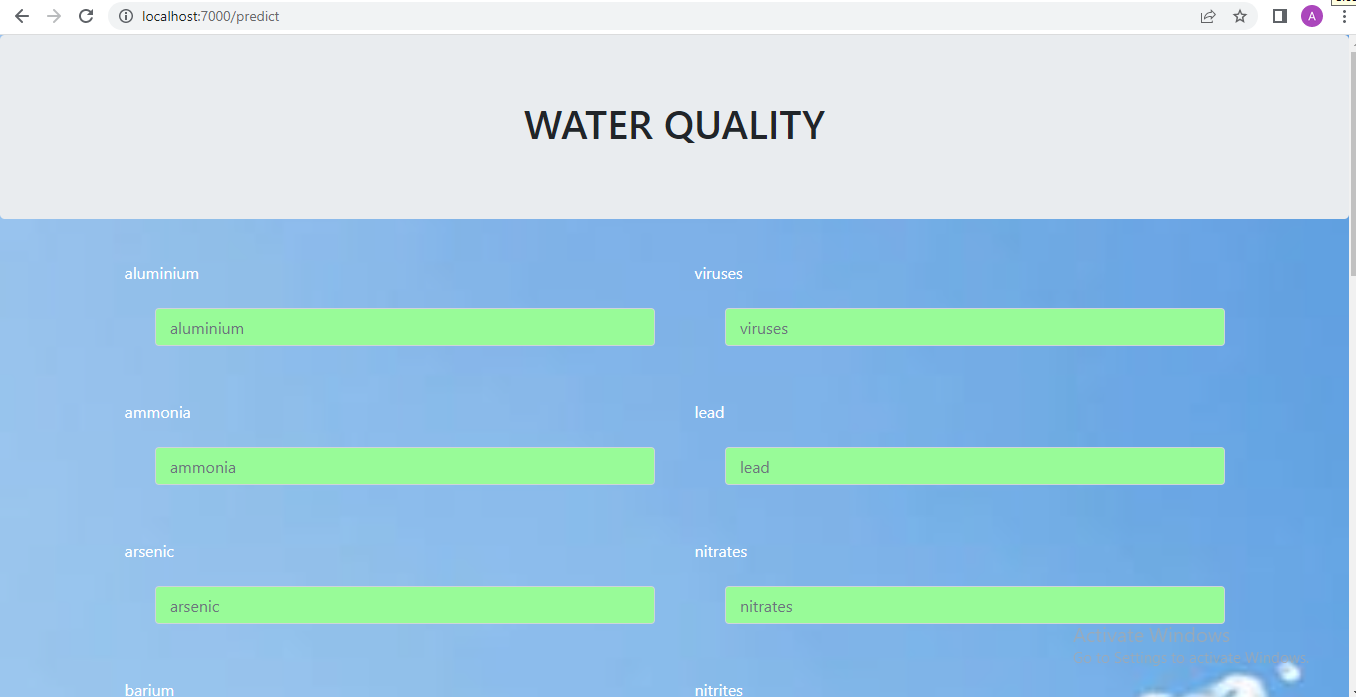


FIG A.1 OUTPUT SCREEN

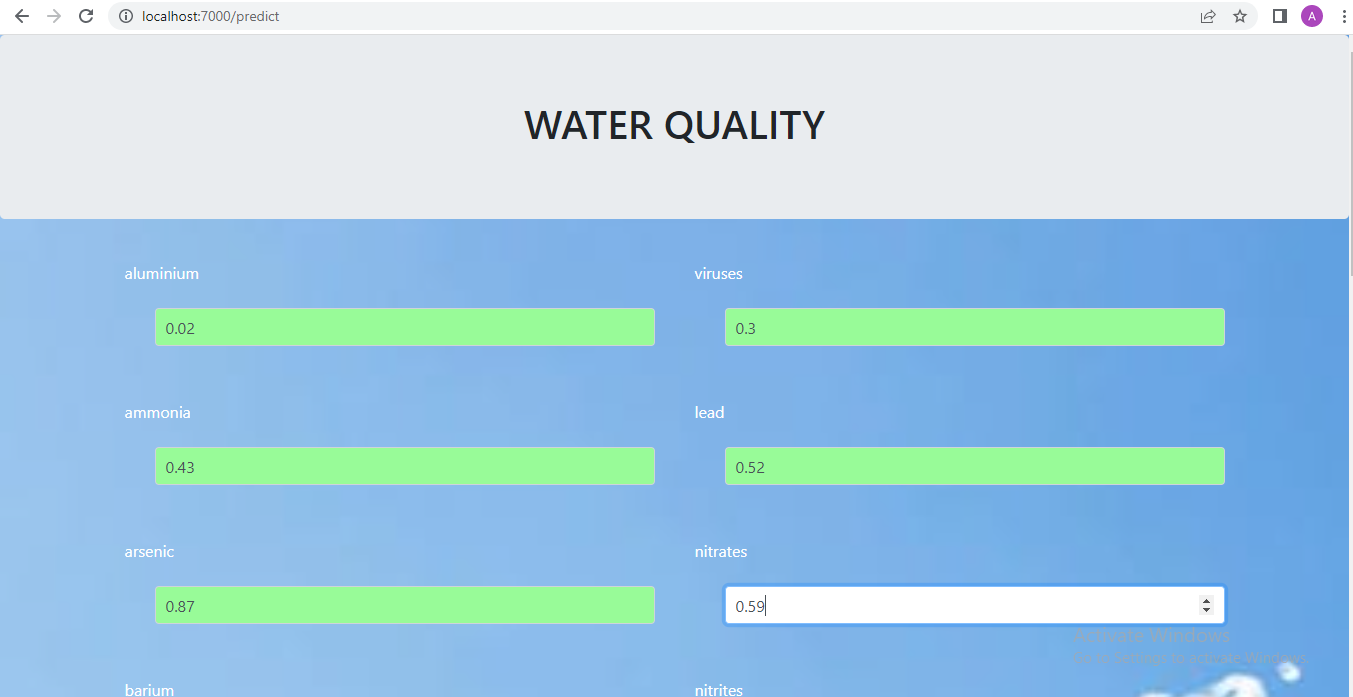


FIG A.2 ENTERING VALUES OF 20 PARAMETERS

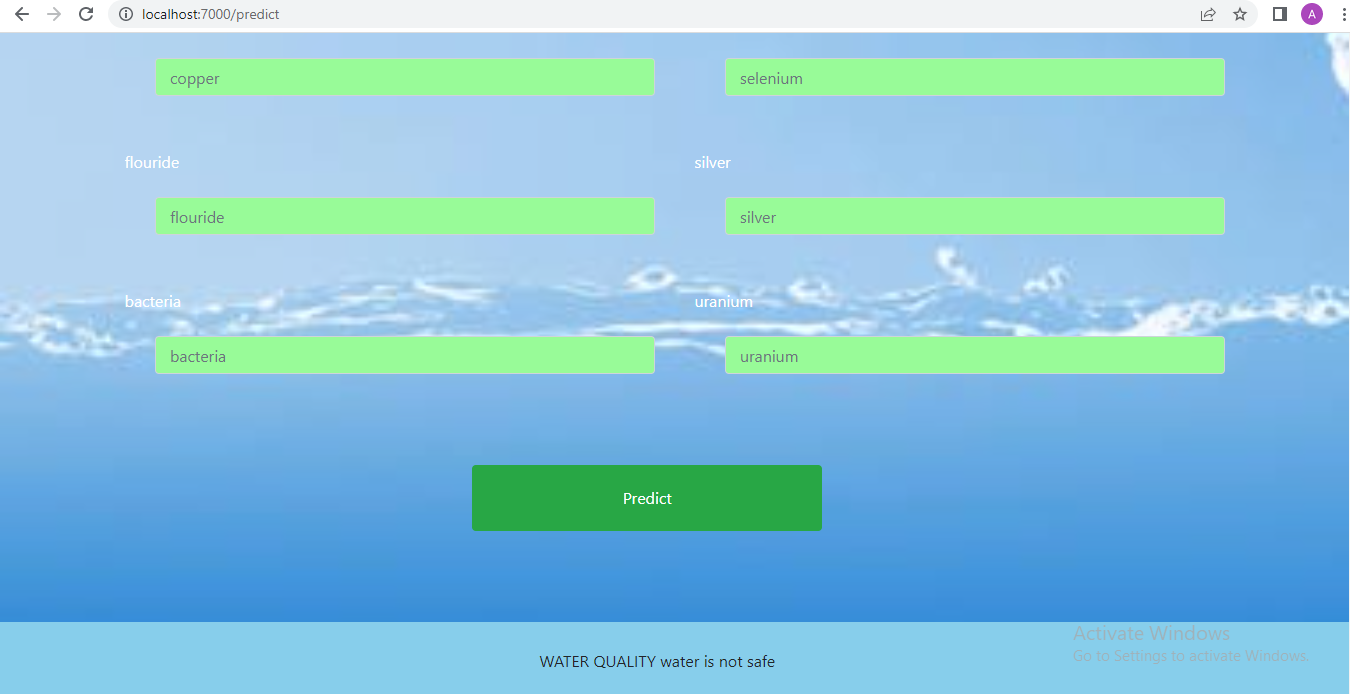


FIG A.3 PREDICTING WATER QUALITY

|  |  |
| --- | --- |
|  |  |

**REFERENCES:**

[1] Archana Solanki, Himanshu Agrawal, Kanchan Khare ,”Predictive Analysis of Water Quality Parameters using Deep Learning”, International journal of computer application,Volume 125 no 9, page 29-34, Year 2019

[2] Mr.M.Anbuchezhian, Dr.R.Venkataraman, Mrs.V.Kumuthavalli,”Water Quality Analysis and Prediction using Machine Learning Algorithms”, International journal of emerging technologies and innovative Research, Volume 5 no 11,page 455-462 , Year 2018

[3]Neha Radhakrishnan; Anju S Pilla, “Comparison of Water Quality Classification Models using Machine Learning”, Fifth International Conference on Communication and Electronics Systems ,page 1183-1188, Year 2020

[4]Msiska slam khan,”Water quality prediction and classification based on principal component regression” , Journal of King Saud University-Computer and Information Science, page 1-9, Year 2021

[5]Hemant Raheja,Arun Goel, Mahesh Pal,”Prediction of groundwater quality indices using Machine Learning algorithms”, International journal of machine learning and computing,Volume17 no , page 337-351, Year 2021

[6]Huiqing Zhang, Kemei Jin,”Research on water quality prediction method based on AE-LSTM”, 5th International Conference on Automation, Control and Robotics Engineering, page 602-606, Year 2021

[7]P.Senthil kumar, Prasannamedha G, Soumya k, “Water quality analysis in a lake using deep learning methodology”, International journal of environmental analytical chemistry, page 1-16, Year 2020

[8]Nikhil M Ragi; Ravishankar Holla; G Manju,”Predicting Water Quality Parameters Using Machine Learning”, 4th International Conference on Recent Trends on Electronics, Information, Communication Technology (RTEICT), Year 2020

[9]Theyazn H. H Aldhyani, Mohammed Al-Yaari, Hasan Alkahtani and Mashael Maashi ,”Water Quality Prediction Using Artificial Intelligence Algorithm”, International journal of applied bionics and biomechanics ,page 1-12, Year 2020

[10] S.Vijay & Dr.K.Kamaraj, “Ground Water Quality Prediction using Machine Learning Algorithms”, International Journal of Research and Analytical Reviews, Volume 6 no 1,page 743x-749x, Year 2019