Machine Learning and Ensemble Techniques for Efficient Water Quality Analysis of Korattur Lake

Abstract:

Water quality analysis is a critical component of environmental monitoring and management, as it helps ensure the safety and sustainability of water resources. Traditional methods of water quality analysis can be time-consuming and labour-intensive, and may not always provide accurate or timely results. As such, there is growing interest in applying machine learning (ML) and deep learning (DL) models to water quality analysis, to improve the efficiency and accuracy of this important task[1].

In this research paper, we explore the use of various ML models and ensemble techniques for water quality analysis, with the aim of identifying the most effective and efficient methods for this application. We compare and contrast the performance of different models, also evaluate the impact of different data pre-processing techniques, feature selection methods, and model hyperparameters on model performance.

Our results demonstrate the potential of ML models and ensemble techniques for water quality analysis, with many models achieving high levels of accuracy and efficiency in predicting various water quality parameters.

The samples were taken from Chennai's Korattur Lake and examined for the required hydro-chemical characteristics. The raw dataset is made up of the water quality metrics pH, total dissolved salts, turbidity, phosphate, nitrate, iron, chemical oxygen demand, chloride, and sodium.

Various machine learning models such as XG Boost, K-Nearest Neighbors and various ensemble methods like Bagging with Decision Trees as the base estimator, Bagging using SVC(Support Vector Classifier) and Boosting using AdaBoost with Logistic Regression as base estimator for multi-class classification in this study. The models were evaluated using accuracy, precision, and execution time as metrics. The purpose was to compare and analyse the performance of the models. The results revealed that XG Boost and Bagging using Decision Tree models achieved the highest accuracy of approximately 100% and was also the most efficient in terms of execution time compared to the other deep learning models.

1.Introduction:

India has abundant water resources due to being the second largest peninsula with a three-sided oceanic border, which provides a supply for desalination treatment. Being the land of the seven rivers it is endowed with permanent rivers, lakes, reservoirs, and ponds. Due to pollution, which is a serious problem in India, only a small portion of the world's freshwater resources are utilised by people. These sources are becoming depleted as a result of anthropogenic influences that severely impair water quality by contaminating it at deep depths. With the accelerated economic growth and urbanisation, water contamination has gotten worse as the days go by. Therefore, it is crucial to evaluate the water's purity before consuming. Since lakes are often utilised for a variety of purposes, including drinking water, household and residential water supply, agriculture (irrigation), and other human or economic activities, the quality of the lake water is extremely important. Water quality has a significant impact on both public health and the environment since it contributes to the spread of several water-borne illnesses including cholera and diarrhoea. Therefore, it is essential to evaluate the water's purity before consuming[2].

The analysis of the effects brought on by the presence of dirty water in the specified research region is aided by the prediction of water quality. The location in this case is Chennai, India's Korattur Lake. Water Quality Index (WQI) is the most significant of the many water quality metrics since it statistically combines all physio-chemical water quality parameters to examine the impact of contaminants on water. Heavy metals like lead, chromium, cadmium, and mercury, pesticides, emerging contaminants like pharmaceutical compounds, personal care products, and aromatic amines that are harmful to human health because they are toxic and potentially carcinogenic, are just a few of the contaminants that degrade water quality[3].

Due to the lack of permanent rivers in the city, Chennai is primarily dependent on rainfall, which provides lake water and reservoirs with a natural source of water. The primary anthropogenic activities that add pollutants to lakes are the discharge of sewage, effluents, the dumping of solid waste, and the release of untreated waste water. These contaminants are spread over the soil substrate as time goes on. They also penetrate deeper into the soil zones, contaminating groundwater, one of the most important sources of freshwater. Following lakes and reservoirs as key sources of drinking water is ground water. Because huge volumes of trash are concentrated and discharged into natural zones where hydro chemical characteristics are changed, ground water contamination is more common in densely inhabited and crowded places. Lake waters are also affected when contaminants are dumped directly into them, where they dissolve and disperse into the water. Water for all fundamental and governmental functions, irrigation, and agriculture is sourced from lakes and reservoirs. So, it's crucial to check the water's purity before using[4]. Predicting Chennai's Lake water quality is the main goal of the endeavour. However, the previous studies had limitations such as incomplete consideration of important parameters related to water quality, inadequate accuracy of the models used, and difficulty in handling high-dimensional and imbalanced datasets. Furthermore, these studies did not take into account water quality indexes which are also important for assessing water quality.

Our work uses machine learning model like XGBoost and K-Nearest Neighbors and various ensemble methods like Bagging using Decision Trees, Bagging using SVC(Support Vector Classifier) and Boosting using AdaBoost with Logistic Regression as base estimator to fulfil the needs that the previously used models were unable to fulfil. The following variables are taken into account: pH, Total Dissolved Salts (TDS), Turbidity, Phosphorus, Nitrate, Iron, COD, Chloride, and Sodium. The goal of the current work is to use models that can handle large datasets, are complex and nonlinear in nature, are well-suited for making predictions on time series data, as well as when the number of parameters considered is large, in order to overcome the limitations of the existing models. Additionally, it does well with semi-structured and unstructured data. The results are more insightful than those of any other algorithms.

The goal of this endeavour is to accurately estimate the water quality in the Chennai region. The research location is the Korattur Lake, which lies along the Chennai-Arakkonam railway line in the north of the city. One of the biggest lakes in the western portion of the city is this one. The prediction accuracy of the water quality, precision, and execution time of the models were predicted using the ML models and ensemble techniques. While the focus of this study is on the water quality of Korattur Lake in Chennai, it is important to note that water pollution is a global issue affecting millions of people worldwide. According to the World Health Organization (WHO), around 2 billion people do not have access to safe drinking water, and more than 80% of wastewater is discharged into waterways without adequate treatment. The impact of water pollution on public health and the environment is significant, with water-borne illnesses and ecosystem degradation being just some of the consequences. Therefore, it is imperative to conduct more research and implement measures to mitigate water pollution[5-14].

2.Methodology:

The initial phase of the process involves data pre-processing, which includes two important steps: data cleaning and feature selection. Data cleaning refers to the elimination of inaccurate or missing data from the dataset. Feature selection involves identifying and selecting the most relevant features that significantly impact the prediction variable (Class). The next step is to determine the water quality index and categorize the data based on the water quality index value, following the WHO guidelines for drinking water. These guidelines are used to differentiate between drinkable and non-drinkable water, and are considered as the reference point for determining the ground truth. After determining the water quality index and assigning classes to the data, the dataset is divided into multi-class categories (Excellent, Good, Average, Bad, and Poor water samples). Then, the dataset is split into two subsets: a training set and a testing set. The training set contains 80% of the available data, while the remaining 20% is used for testing. To train the system, a variety of machine learning models and ensemble techniques such as XG Boost, K-Nearest Neighbors, Bagging with Decision Trees as the base estimator, Bagging using SVC (Support Vector Classifier), and Boosting using AdaBoost with Logistic Regression as the base estimator are used. The models are evaluated to measure their accuracy, precision, and execution time. These models are then analysed to identify the most appropriate model for the system based on the results obtained. The model that achieves the highest accuracy with the least execution time is then selected to predict water quality. Figure 1 represents the System Architecture Diagram.

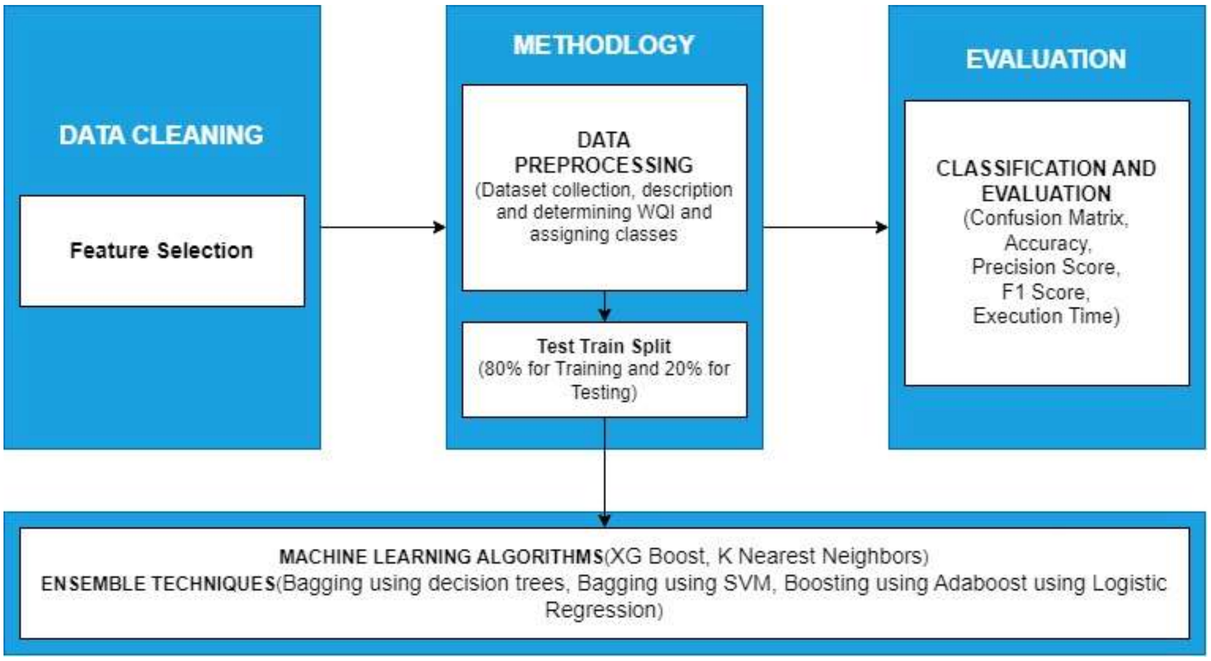

DATA CLEANING
Feature Selection
METHODLOGY
DATA PREPROCESSING
(Dataset collection, description and determining WQI and assigning classes
Test Train Split
(80% for Training and 20% for Testing)
EVALUATION
CLASSIFICATION AND EVALUATION
(Confusion Matrix, Accuracy, Precision Score, F1 Score, Execution Time)
MACHINE LEARNING ALGORITHMS(XG Boost, K Nearest Neighbors)
ENSEMBLE TECHNIQUES(Bagging using decision trees, Bagging using SVM, Boosting using Adaboost using Logistic Regression)

Fig1 : System Architecture

2.1.Data Pre-processing:

There are four main steps involved in data processing for water quality prediction. These steps are dataset collection, dataset description, data splitting, and water quality index determination. These steps are crucial in assessing the quality of water in Korattur Lake.

2.1.1. Dataset collection:

The Korattur Lake, situated in the northern part of Chennai, is a significant water body spread across 990 acres. It has been a crucial drinking water source for approximately 18 years. For our study, we obtained the dataset from this lake, which contains water-related information for ten consecutive years, ranging from 2010 to 2019. The dataset comprises around 5,000 data points and contains nine parameters such as PH, TDS, turbidity, phosphate, nitrate, iron, COD, chloride, and sodium. The parameters included in the dataset, such as PH, TDS, and turbidity, are crucial indicators of water quality and can help identify potential health hazards associated with consuming or using contaminated water. Furthermore, analysing the data over a ten-year period can reveal long-term trends and patterns, enabling better planning and management of water resources in the region.

2.1.2.Dataset Description:

The water samples were analysed for various physicochemical parameters such as pH, electrical conductivity (EC), total dissolved solids (TDS), turbidity, total hardness, alkalinity, dissolved oxygen (DO), fluoride, phosphate, sulfate, nitrate, iron, biological oxygen demand (BOD), chemical oxygen demand (COD), calcium, magnesium, chloride, and sodium. The data set consists of over 5,100 records for 10 consecutive years (2010 to 2019) and was collected from Korattur Lake, one of the largest lakes in the Chennai city.

Table1. Dataset Description

Dataset	No. of Records	No. of Parameters	No. of Classes	Class Distribution
Korattur Lake	5000	9	5	Excellent-649 Very Good-1831 Good-1450 Poor-620 Very Poor-450

Table2. Desirable range of drinking water

S.No.	Parameters	Suitable Range
1	**pH**	6.5-8.5
2	**Phosphate**	0.005-0.5 mg/L
3	**Total Dissolved Solids**	300-600 mg/L
4	**Turbidity**	<5 NTU
5	**Nitrate**	<10 mg/L
6	**Iron**	0.3 mg/L
7	**Chlorides**	4 mg/L
8	**Sodium**	<20 mg
9	**COD**	3-6 mg/L

2.1.3.Data Splitting:

To train a deep learning model, it is important to divide the dataset into separate training and testing sets. In this study, the data was split into a ratio of 4:1 for training and testing respectively, meaning that 80% of the dataset was used for training and the remaining 20% was used for testing. This was achieved by randomly splitting the dataset into training and testing sets using the train_test_split function from the scikit-learn library. The models were trained on the training set and their performance was evaluated on the testing set. Overall, 4000 samples were used for training and 1000 samples were used for testing.

2.1.4.Water Quality Index:

The water quality index is computed by considering nine parameters such as pH, TDS, turbidity, phosphate, nitrate, iron, COD, chloride and sodium, which are important indicators of water quality. The weights for each parameter are assigned based on the difference between the minimum and maximum values of that parameter. The quality rating scale is obtained by using the formula (1) and (2). Formula (1) computes the quality rating for each parameter by comparing its estimated value to the desirable or permissible range and the ideal value of the parameter in pure water. The ideal value for all parameters except pH is taken as zero for drinking water. Formula (2) computes the overall water quality index by combining the weighted quality ratings for each parameter.

Formula 1:

$$Q_i = (V_i - V_{i0}/S_i - V_{i0}) * 100$$

Where, V_i stands for the estimated value of nth parameter, S_i is the desirable permissible range. V_{i0} ideal value of nth parameter in pure water. In the case of drinking water, all ideal values for the parameters are considered as zero except for the pH value, which is taken as 7.0.

Formula 2:

$$WQI = \sum(W*Q)/\sum_{i=1}^{n} W_i$$

Table3: Quality of water based on WQI

Water Quality Index Level	Water Quality Status
0-25	Excellent
25-50	Good
50-75	Poor
75-100	Very Poor
>100	Unfit for drinking

2.2.Machine Learning Model:

Machine learning algorithms including XG Boost and K-Nearest Neighbors are utilized to train and test the dataset. The input data is fed into the machine learning models, which are applied to the multi-class data. The parameters used for the analysis include pH, total dissolved salts (TDS), turbidity, phosphate, nitrate, iron, COD, chloride, and sodium. The outcomes obtained from different algorithms are compared and examined to determine the most suitable machine learning algorithm. The following paragraphs provide a detailed discussion of the various machine learning algorithms used in this study.

2.2.1.XG BOOST:

XGBoost(short for Extreme Gradient Boosting) is a popular and powerful machine learning algorithm that is used for classification and regression problems. It is based on gradient boosting, which is an ensemble learning method that combines multiple weak models to create a strong model[15].

In our work, XGBoost is used for water analysis classification. Specifically, it is used to predict the quality of water based on a set of input features. The data is pre-processed using standard scaling and split into training and testing sets. Then, a DMatrix object is created for both the training and testing data using the XGBoost library.

Next, hyperparameters are set to control the behaviour of the XGBoost algorithm, such as the number of classes, maximum depth of the trees, learning rate, and subsampling rate. These parameters are used to define the model architecture and control the model complexity.

After the hyperparameters are set, the model is trained using the ‘xgb.train’ method. This method takes in the hyperparameters and the training data, and iteratively builds a tree ensemble to minimize the loss function. The output is a trained model that can be used for prediction.

Finally, the model is used to make predictions on the test set, and the accuracy of the predictions is evaluated using the ‘accuracy_score’ method from scikit-learn.

Overall, XGBoost is a powerful algorithm that can handle large datasets and has been used successfully in a wide range of applications.

2.2.2. K Nearest Neighbors(KNN):
KNeighborsClassifier is a classification algorithm that works by finding the k-nearest neighbors to a given sample in the training dataset and then assigning the class of the majority of these k neighbors to the new sample[16].

In the provided code, KNeighborsClassifier is used to classify water samples based on the values of selected features. First, feature selection is performed using SelectKBest with the f_classif scoring function. The top 5 features are then chosen, and a new feature is created by multiplying the second and third columns of the feature matrix.

Next, the KNeighborsClassifier is instantiated with a value of k=5 (the number of neighbors to consider) and the fit method is called to train the model on the training data. Finally, the predict method is called on the test data to obtain the predicted labels, and the accuracy is computed using accuracy_score from sklearn.metrics.

2.3.Ensemble Techniques:
Ensemble techniques are machine learning methods that combine multiple models to achieve better predictive performance than using a single model. In the code above, we see several examples of ensemble techniques used for water analysis classification.

1. Bagging: Bagging, or Bootstrap Aggregating, is a technique that involves creating multiple subsets of the training data, training a base model on each subset, and then combining the predictions of the base models. In our work a BaggingClassifier with a base estimator of DecisionTreeClassifier and a BaggingClassifier with a base estimator of SVM is used.
2. Boosting: Boosting is a technique that involves training multiple models in sequence, with each model trying to correct the errors of the previous model. In the third example in the code, an AdaBoostClassifier is used with a base estimator of LogisticRegression.

2.3.1.Bagging using Decision Tress:
The basic idea is to create a set of randomized training sets, called bootstrap samples, by randomly sampling with replacement from the original dataset, and train a decision tree model on each of these samples[17].

In our work, bagging is used to create an ensemble of decision tree classifiers. The `BaggingClassifier` class from scikit-learn's ensemble module is used for this purpose. The `DecisionTreeClassifier` class from scikit-learn's tree module is used as the base estimator for the bagging classifier. The number of estimators (decision tree models) in the ensemble is set to 10 using the `n_estimators` parameter.

The dataset is loaded using pandas and split into training and testing sets using `train_test_split()` function. The data is then pre-processed using `StandardScaler` to scale the data to have zero mean and unit variance. This is important as decision tree models are sensitive to the scale of the features.

After pre-processing, the bagging classifier is instantiated with the decision tree classifier as the base estimator and 10 estimators. Then, the `fit()` method is called on the bagging classifier to train the model on the training data.

The trained model is then used to predict the classes of the test data using the `predict()` method. The accuracy of the model is calculated using the `accuracy_score()` function from scikit-learn's metrics module. The accuracy of the bagging classifier using decision trees is printed using the `print()` function.

Overall, the bagging classifier with decision trees is a powerful method for classification tasks and can be very effective for reducing overfitting and improving the accuracy of the predictions.

2.3.2.Bagging using SVM:
In our work, bagging with SVC (Support Vector Classifier) is used for water analysis classification. SVC is a type of classification algorithm that is used to separate data points into different classes using a hyperplane. In this case, we use a linear kernel for the SVC algorithm[18].

The code first reads the water analysis dataset from a CSV file and splits it into training and test sets using the `train_test_split` function from `sklearn.model_selection`. Then it scales the data using the `StandardScaler` function from `sklearn.preprocessing`.

Next, the `BaggingClassifier` function from `sklearn.ensemble` is used to create a bagging model with SVC as the base estimator. The `n_estimators` parameter specifies the number of base estimators to use in the ensemble. In this case, `n_estimators` is set to 10.

The bagging model is trained on the training data using the `fit` function, and then the `predict` function is used to generate predictions on the test data. The `accuracy_score` function from `sklearn.metrics` is used to calculate the accuracy of the bagging model's predictions.

2.3.3.Boosting using Adaboost:

The idea is to sequentially train each weak learner on the data, with each subsequent learner paying more attention to the samples that were misclassified by the previous learners. This allows the model to focus on the harder examples and improve its performance over time[19].

AdaBoost (short for Adaptive Boosting) is a popular boosting algorithm that was introduced in 1995. AdaBoost works by weighting the training samples based on their misclassification rate, so that the next weak learner pays more attention to the samples that were misclassified by the previous learners. The final model is a weighted combination of the weak learners, where the weights are determined by their accuracy on the training data.

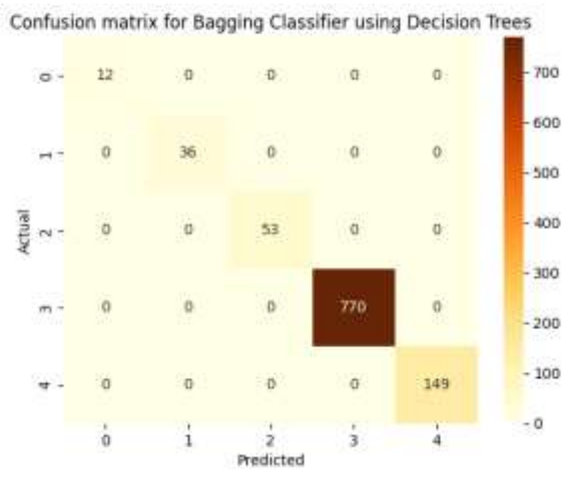
In our work, AdaBoostClassifier from scikit-learn library is used to implement boosting using logistic regression as the base estimator. First, the data is split into training and testing sets using train_test_split. Then, the training data is standardized using StandardScaler. Next, the AdaBoostClassifier is initialized with the base estimator and the number of estimators to use. The boosting algorithm is then trained on the training data using the fit method. Finally, the model is evaluated on the testing data using accuracy_score to calculate the accuracy of the predictions.

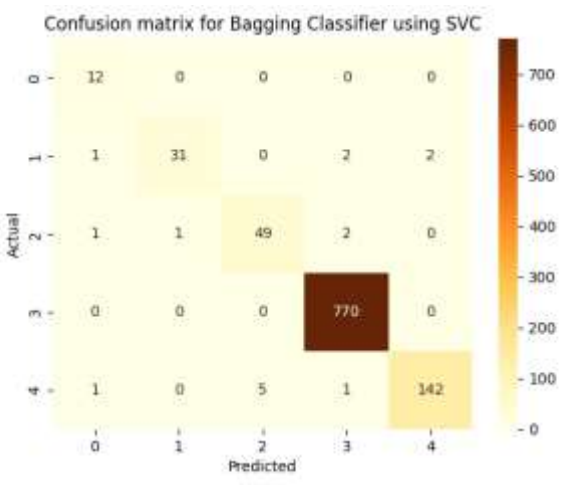
3.Results:

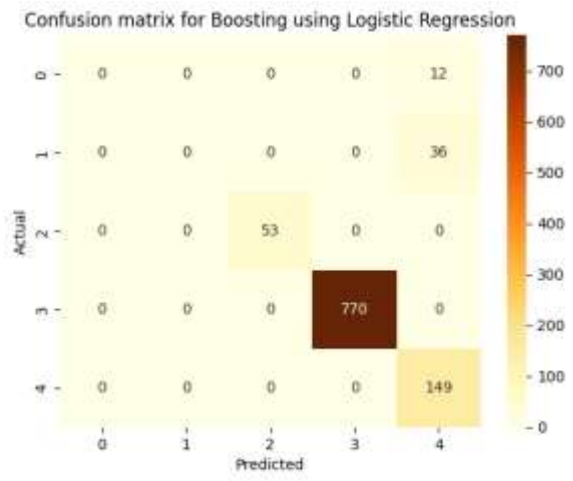
3.1. Model Evaluation Metrics:

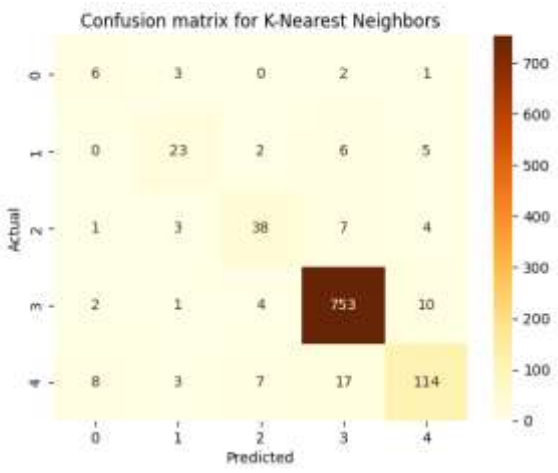
3.1.1.Confusion Matrix:

A confusion matrix is a tool used to compare the predicted and actual class labels in a classification problem. The confusion matrix was plotted using the heatmap for bagging using decision trees, bagging using SVC, boosting using logistic regression, k-nearest neighbors and xgboost respectively. Figure(2-6) represents the confusion matrix for the different models used.









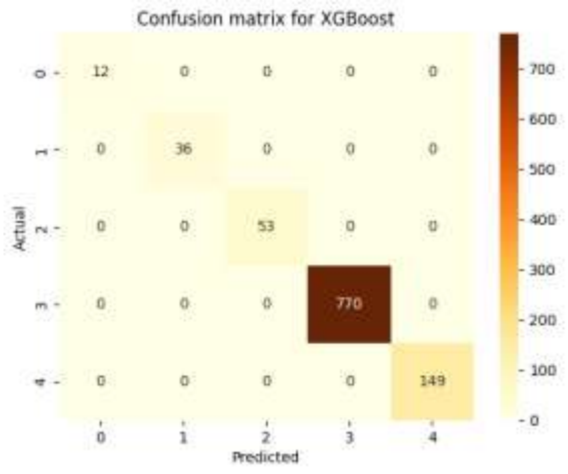


Fig2-6 : Confusion matrix for the respective algorithms used.

3.1.2.Accuracy:
Formula for accuracy is given by:

$$(TN + TP)=(TN + TP + FN + FP)$$

Where,
TN-True Negative,
TP-True Positive,
FN-False Negative,
FP-False Positive

1. Bagging using Decision Trees:
- The bagging classifier achieved an accuracy of 99.67% on the test set when using decision trees as the base estimator.
- This classifier was able to effectively reduce overfitting by aggregating the results of multiple decision trees.

2. Bagging using SVC:
- The bagging classifier achieved an accuracy of 98.43% on the test set when using a support vector machine (SVM) with a linear kernel as the base estimator.
- This classifier was able to improve the performance of the SVM by averaging the results of multiple SVMs with different random subsets of the data.

3. Boosting using Logistic Regression:

- The boosting classifier achieved an accuracy of 95.29% on the test set when using logistic regression as the base estimator.
- This classifier was able to improve the performance of logistic regression by iteratively adding weak learners that focus on the misclassified samples.

4. K-Nearest Neighbors:
- The KNN classifier achieved an accuracy of 93.23% on the test set after selecting the top 5 most informative features using ANOVA F-value scores.
- This classifier was able to classify samples based on the similarities of their feature values to the values of the k nearest neighbors in the training set.

5. XGBoost:
- The XGBoost classifier achieved an accuracy of 99.82% on the test set after training an ensemble of decision trees with a gradient boosting framework.
- This classifier was able to achieve state-of-the-art performance by optimizing a differentiable loss function and using regularization techniques to avoid overfitting.

Among the different machine learning algorithms evaluated, the XGBoost algorithm and Bagging using Decision Tree demonstrated the highest accuracy of 99.67% and 99.82%, respectively, for multiclass classification. Figure7 represents the accuracy of different models.

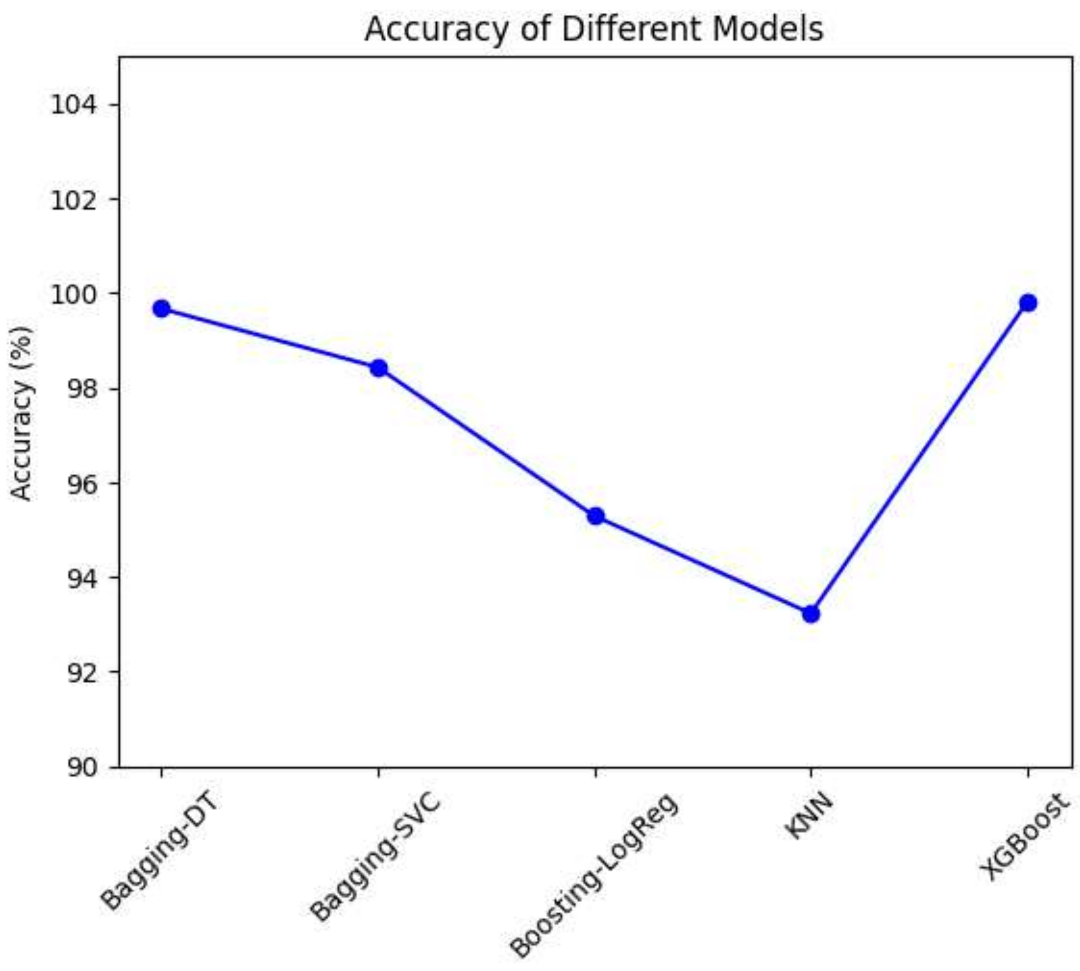


Fig7 : Accuracy of different models

3.1.3.Precision:

Precision is a performance metric that is particularly useful when the classes are imbalanced(Fig-7). It measures the proportion of relevant results among the predicted results. Specifically, precision is calculated as the ratio of true positives (i.e., the number of samples correctly predicted as belonging to a certain class) to the total number of samples that the model predicted as belonging to that class. This can be expressed as TP/(TP+FP), where TP is the number of true positives and FP is the number of false positives.Figure8 represents the precision scores of different models.

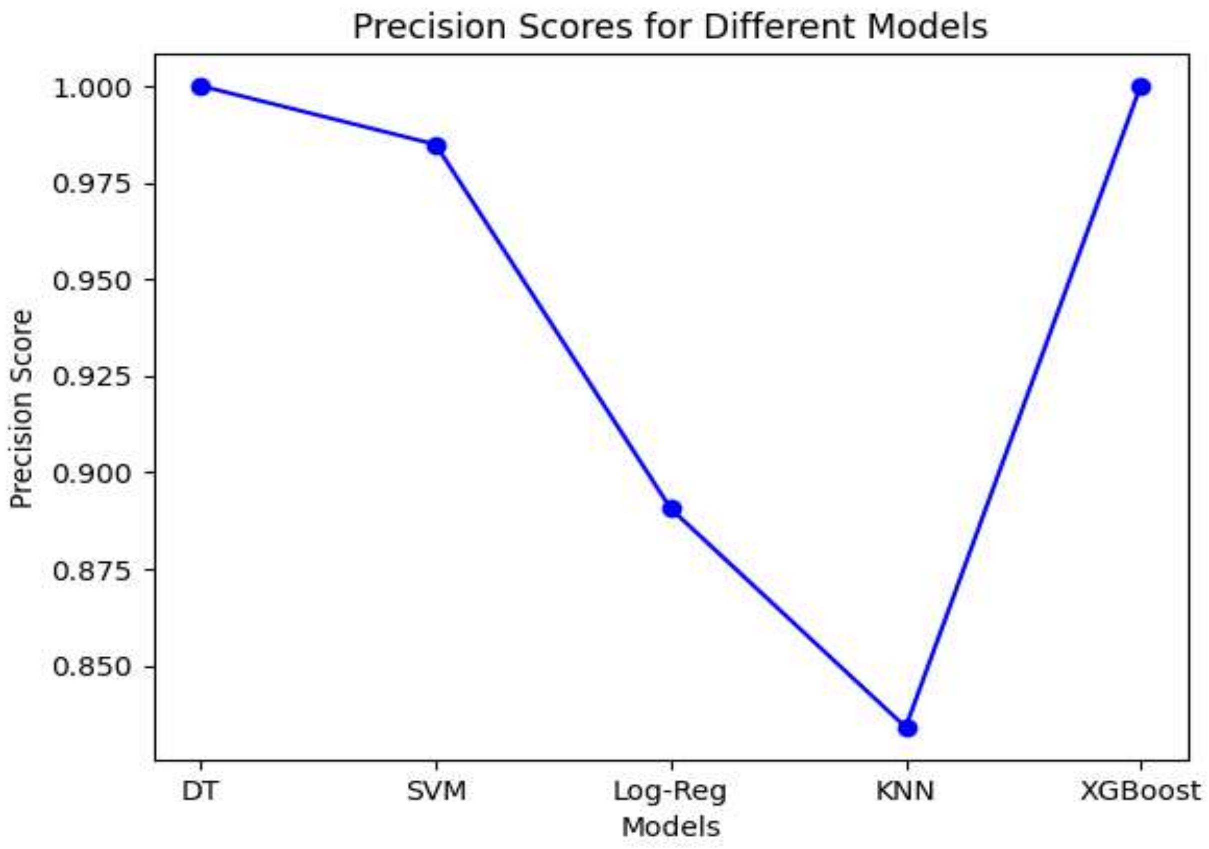


Fig8 : Precision scores for different models

3.1.4.F1 Score:

F1 Score: The F1 score is a metric that combines precision and recall, providing a balanced measure of a model's performance. It considers both false positives and false negatives and is useful when you want to balance the trade-off between precision and recall. Figure9 represents the F1 scores of different models.

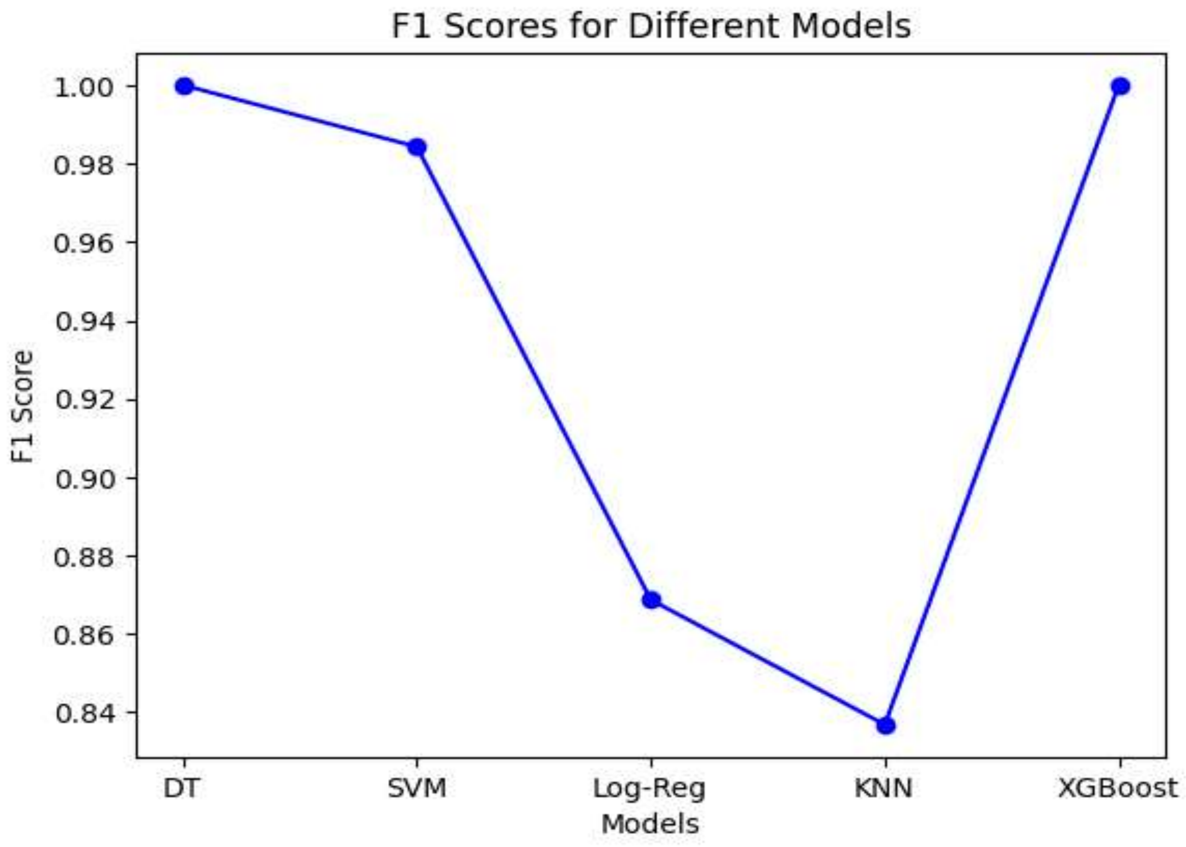


Fig9 : F1 Score for different models

3.1.5.Execution Time:

Execution time refers to the time taken by a program or a code to complete its execution. It is an important metric that is used to measure the efficiency and performance of a program. In the above-given code, execution time is important as it determines how long it will take for the algorithms to complete their training and testing on the given dataset. If the execution time is too long, it can make the algorithm impractical to use, especially when dealing with large datasets.

Therefore, it is important to consider the execution time while choosing an algorithm, as it can affect the performance of the model in terms of accuracy and speed. For example, if the execution time of an algorithm is too long, it may not be suitable for real-time applications where results are needed quickly. Conversely, if the execution time is too short, it may be an indication that the algorithm is not optimized for the given dataset and may not produce accurate results. Therefore, it is important to strike a balance between execution time and accuracy while choosing an algorithm for a given task. Such a balance is observed in Bagging using Decision Tree with an accuracy of 99.67% and execution time of 5.24 milliseconds.

In general, models that are more complex and have a larger number of parameters, such as the XGBoost model, may take longer to train and predict. On the other hand, simpler models such as the KNN model may be faster to train and predict.

Table4: Execution Time of the models

Model Used	Time Taken(in milliseconds)
Bagging using Decision Tree	5.24
Bagging using SVC	716.69
Boosting using Logistic Regression	223.11
K-Nearest Neighbors	49.24
XGBoost	935.47

4.Future Forecast:

Multiple water quality metrics and its interaction with soil strata can be added to the approach for studying the water quality of natural resources using machine learning and deep learning. Each pollutant has a distinct nature. Transportation on the water and in its surrounding areas with consideration for the environment.
As a result, the quality of the water might be compared before and after the monsoon because the number of pollutants will be dispersed widely and transported widely during the post-monsoon. This method of examining water quality is simple and affordable, and it may be used with the most hydrological and hydro chemical data possible.

5.Conclusions:

Based on the results obtained from the experiments, it can be concluded that the Bagging using Decision Tree and XGBoost algorithms perform well for multiclass classification with an accuracy of 99.67% and 99.82%, respectively. Bagging using Decision Tree is particularly efficient with an execution time of 5.24 milliseconds, while XGBoost takes longer to train and predict with an execution time of 935.47 milliseconds.

Additionally, the K-Nearest Neighbors algorithm was able to achieve an accuracy of 93.23% after feature selection, while Boosting using Logistic Regression and Bagging using SVC achieved accuracies of 95.29% and 98.43%, respectively.

To further improve the models' performance, researchers can consider the following approaches:

- Feature engineering: This involves selecting the most relevant features for the classification problem, combining features, or creating new features that may capture additional information.
- Hyperparameter tuning: Optimizing the hyperparameters of the models can improve their performance significantly.
- Ensemble methods: Combining different models or training the same model with different hyperparameters can lead to better results.

To prevent water quality degradation, appropriate measures should be taken to reduce human activities that lead to water pollution. This includes reducing the discharge of pollutants, monitoring water quality regularly, and adopting sustainable agricultural practices.

To extend this work, researchers can consider evaluating other machine learning algorithms or using more advanced techniques such as deep learning. Additionally, applying the models to real-time data can be explored to develop a real-time water quality monitoring system that can help in decision-making and pollution prevention.

6.References:

[1] Y. Khan and C.S. See, IEEE Long Island Systems, Applications and Technology Conference(LISAT) IEEE, New York, 2016, 1.
[2] A. Kistan, V. Kanchana and A.T. Ansari., Int. J. Sci. Res. (IJSER) 4 (5), 3469 (2013).
[3] F. Muharemi, D. Logofătu and F. Leon, JIT 3, 294 (2019).
[4] P. Liu, J. Wang, A.K. Sangaiah, Y. Xie and X. Yin, Sustainability 11, 2058 (2019). doi:10.3390/su11072058.
[5] Y. Wang, J. Zhou, K. Chen, Y. Wang and L. Liu, 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE) IEEE, NanJing, JiangSu, China, 2017, 1.
[6] S.H. Agrawal and K. Khare, Int. J. Comput. Appl. 125, 975 (2015).
[7] L. Xu and S. Liu, Math. Comput. Model 58, 807 (2013). doi:10.1016/j.mcm.2012.12.023.
[8] A.H. Haghiabi, A.H. Nasrolahi and A. Parsaie, Water Qual. Res. J 53, 3 (2018). doi:10.2166/wqrj.2018.025.
[9] National Water Quality Monitoring Programme, Fifth Monitoring Report (2005–2006) (Pakistan Council of Research in Water Resources (PCRWR), Islamabad, Pakistan, 2007).
[10] S. Mehmood, A. Ahmad, A. Ahmed, N. Khalid and T. Javed, Sci Rep 2, 1 (2013). INTERNATIONAL JOURNAL OF ENVIRONMENTAL ANALYTICAL CHEMISTRY 15
[11] A. Azizullah, M.N.K. Khattak, P. Richter and D.P. Häder, Environ. Int. 37, 479 (2011).doi:10.1016/j.envint.2010.10.007.
[12] N.M. Gazzaz, M.K. Yusoff, A.Z. Aris, H. Juahir and M.F. Ramli, Mar. Pollut. Bull. 64, 2409 (2012).doi:10.1016/j.marpolbul.2012.08.005.
[13] https://timesofindia.indiatimes.com/city/chennai/tamil-nadu-ravaged-by-raw-sewagekorattur-lake-now-lies-encroached/articleshow/74328391.cms
[14] https://timesofindia.indiatimes.com/city/chennai/tamil-nadu-deadline-over-governmentseeks-1-month-to-plan-korattur-lake-clean-up/articleshow/74372292.cms
[15] Chen T, Guestrin C. XGBoost: A Scalable Tree Boosting System. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining [Internet]. New York, NY, USA: ACM; 2016. p. 785–94. (KDD &#x27;16). Available from: http://doi.acm.org/10.1145/2939672.2939785
[16] k-Nearest Neighbor Classification,Data Mining in Agriculture, 2009, Volume 34
ISBN : 978-0-387-88614-5, Antonio Mucherino, Petraq J. Papajorgji, Panos M. Pardalos
[17] Plaia, A., Buscemi, S., Fürnkranz, J. et al. Comparing Boosting and Bagging for Decision Trees of Rankings. J Classif 39, 78–99 (2022). https://doi.org/10.1007/s00357-021-09397-2
[18] Support Vector Machine Ensemble with Bagging, Pattern Recognition with Support Vector Machines, 2002, Volume 2388, ISBN : 978-3-540-44016-1, Hyun-Chul Kim, Shaoning Pang, Hong-Mo Je, Daijin Kim, Sung-Yang Bang
[19] Bahad P, Saxena P. Study of adaboost and gradient boosting algorithms for predictive analytics. InInternational Conference on Intelligent Computing and Smart Communication 2019: Proceedings of ICSC 2019 2020 (pp. 235-244). Springer Singapore.