**Project :-**

Underground water quality distribution

Team members: -

1. Avinash Kumar B22ME014
2. Aditya Pote B22ME005
3. Shardul Date B22ME077
4. Yash Golani B22ME072
5. Rohan Gupta B22MT037

**Prof.: - Avinash Sharma**

**Mentor: - Palaskar Adarsh Mahesh**

Project Collab link : - https://colab.research.google.com/drive/1FlOwU2HajFNw0KLvXmpdyQ6s\_31RYnYX?usp=sharing#scrollTo=gzwMLHUp2p9P

Report

We have given three 3 datasets of years 2018,2019,2020 containing the information of parameters that a water can be classified on the basis of these parameters into different classes---one for water irrigation and other for livestock.

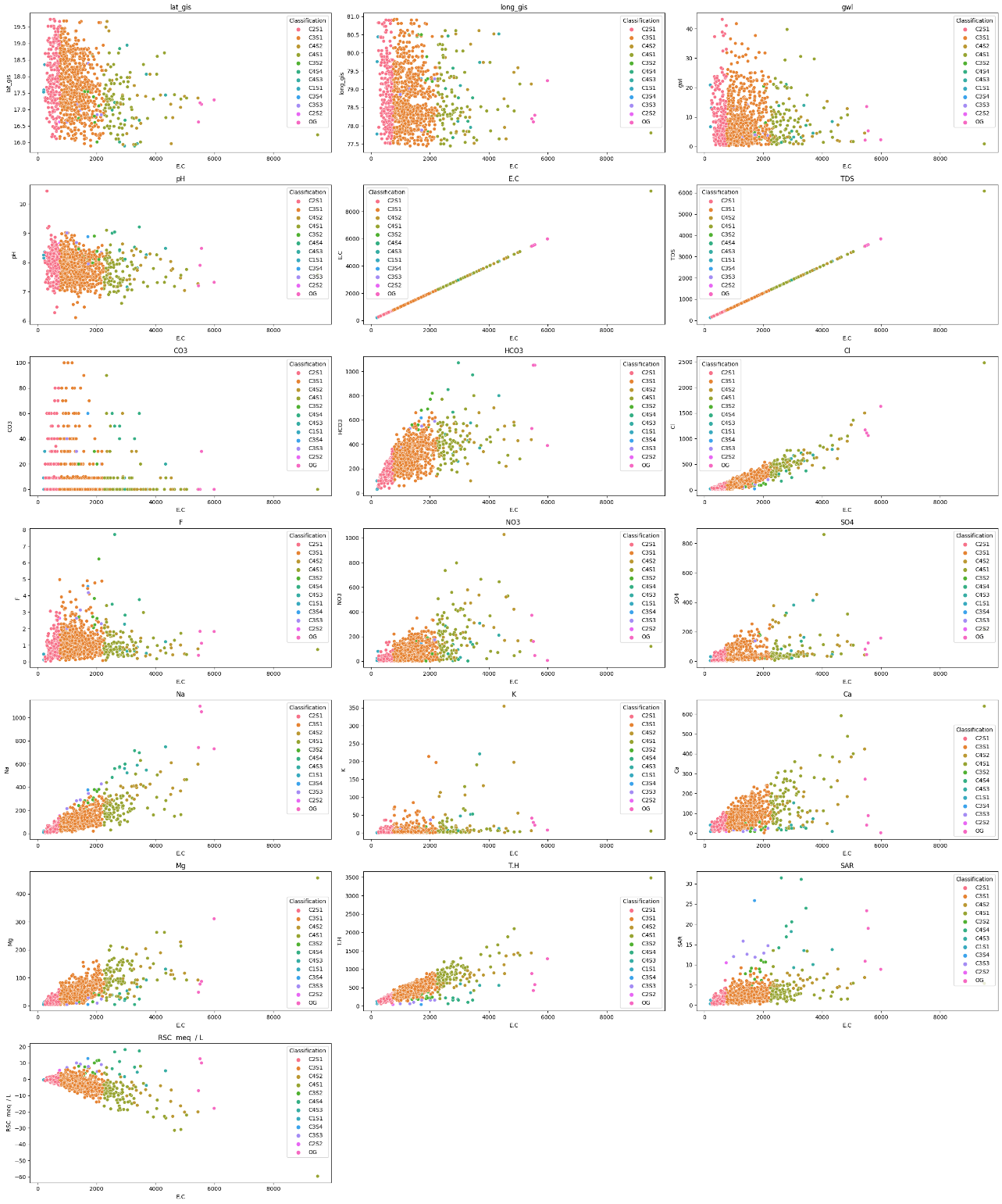
Data preprocessing: -

* Imported all the libraries that would need during the model training and evaluating it.
* Loaded all the dataset using pandas library (pd.read\_csv).
  + - Checked each file carefully to make sure there is no mistake. For example, taking the first dataset (2018\_csv file) named as df\_1. Checked there: -
      * + Its 1st 5 rows
        + Missing value
        + Name column
        + Unnamed column
        + Similar column
      * After checking these: -
        + 3-missing values in gwl column
      * Similarly, in 2nd dataset(df\_2): -
        + Different column names
        + 160 missing values in CO\_-2 column
      * Similarly, in 3rd dataset(df\_3): -
        + 1 unnamed column
        + 3 missing values in gwl column
* All the missing values are filled with their respecting column mean.
* Different column names have been changed
* Unnamed column has been dropped.
* After going though these processes, all the datasets are combined to each other and the combined dataset is named as ‘df’.
* Once again, repeated those processes if there is still any mistake.
* For exploratory data analysis, checked the number of classes in Classification column, and got there 13 classes-11 + 2 classes (O.G, OG) which contains those entities are not in those of 11 classes but both O.G and OG are same classes that’s why O.G is replaced by OG. And then created a numcol list containing all the columns having numerical values.
* During EDA, found a float value in the pH column and which has been fixed it.

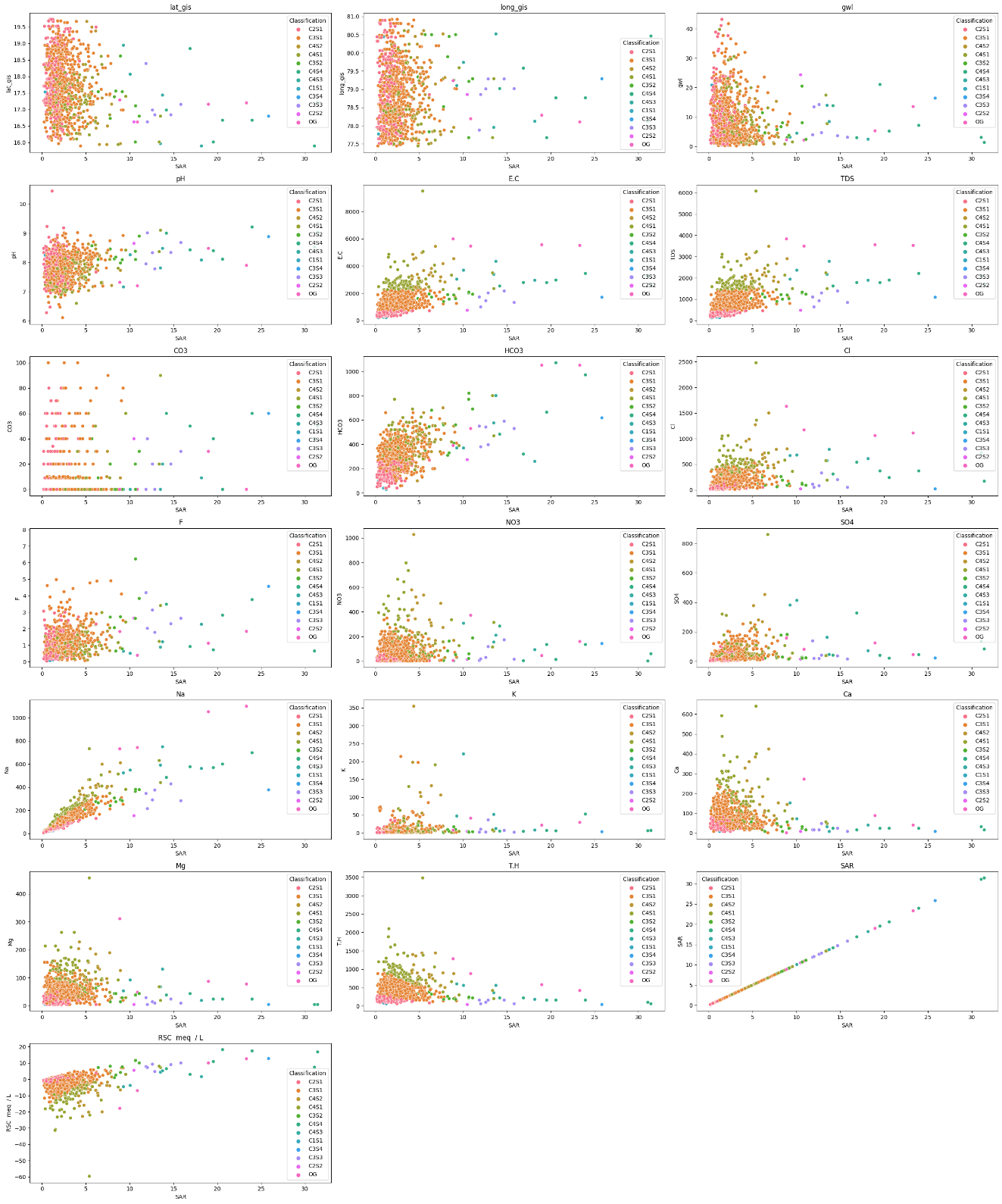
Exploratory Data Analysis: -

* In the numcol, there are two important columns- one electrical conductivity (E.C) and other Sodium adsorption ratio (SAR).
* There are a number of scatterplots between E.C vs others and SAR vs others.
  + In E.C vs others, E.C vs TDS has a linear line forming structure because of the fact --Conductivity (EC) and total dissolved solids (TDS) are water quality parameters, which are used to describe salinity level. These two parameters are correlated and usually expressed by a simple equation: TDS = k EC (in 25 °C)
* There are a number of histograms- one group of before normalizing the data and other after normalizing the data.

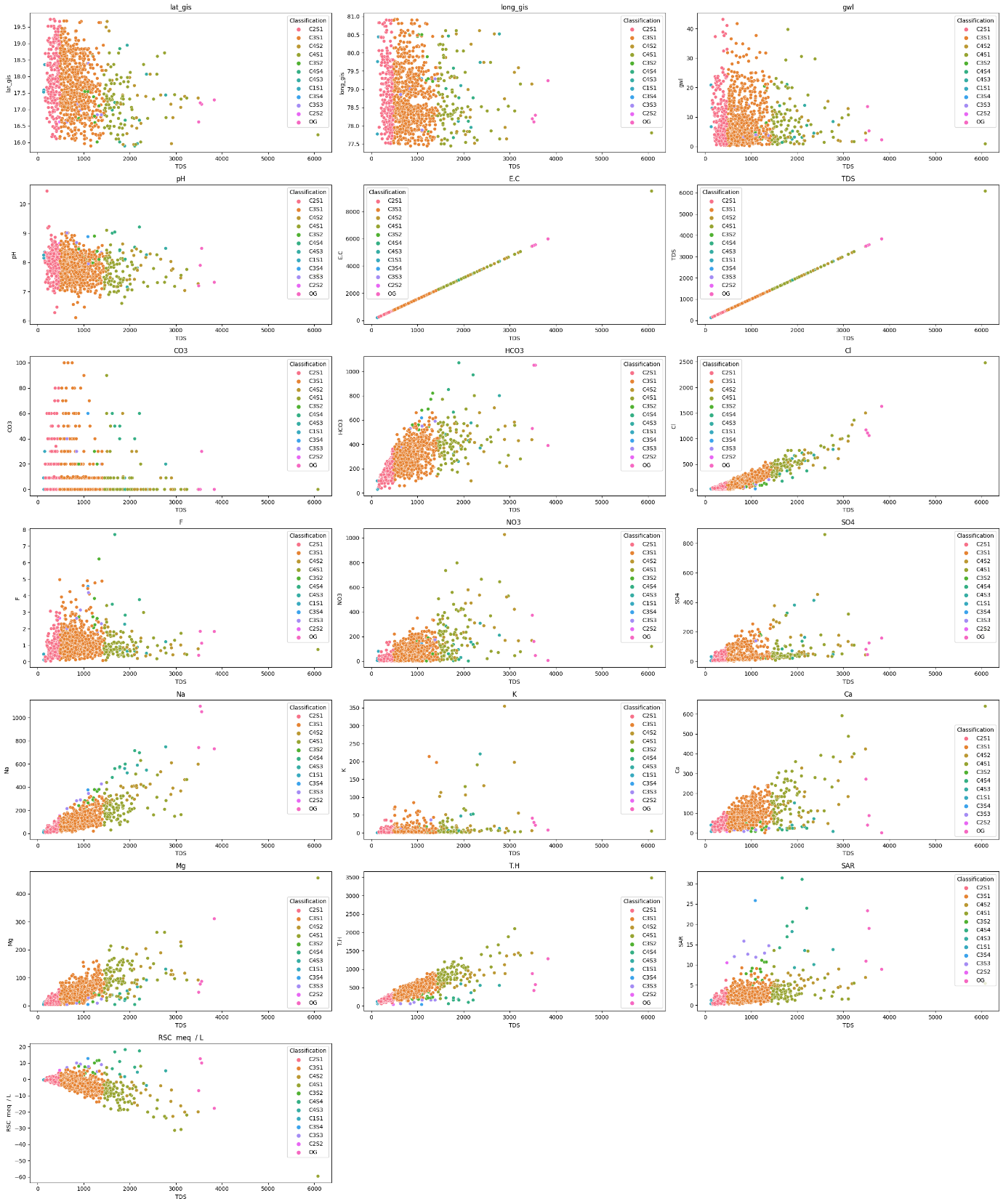
**E.C vs Others**



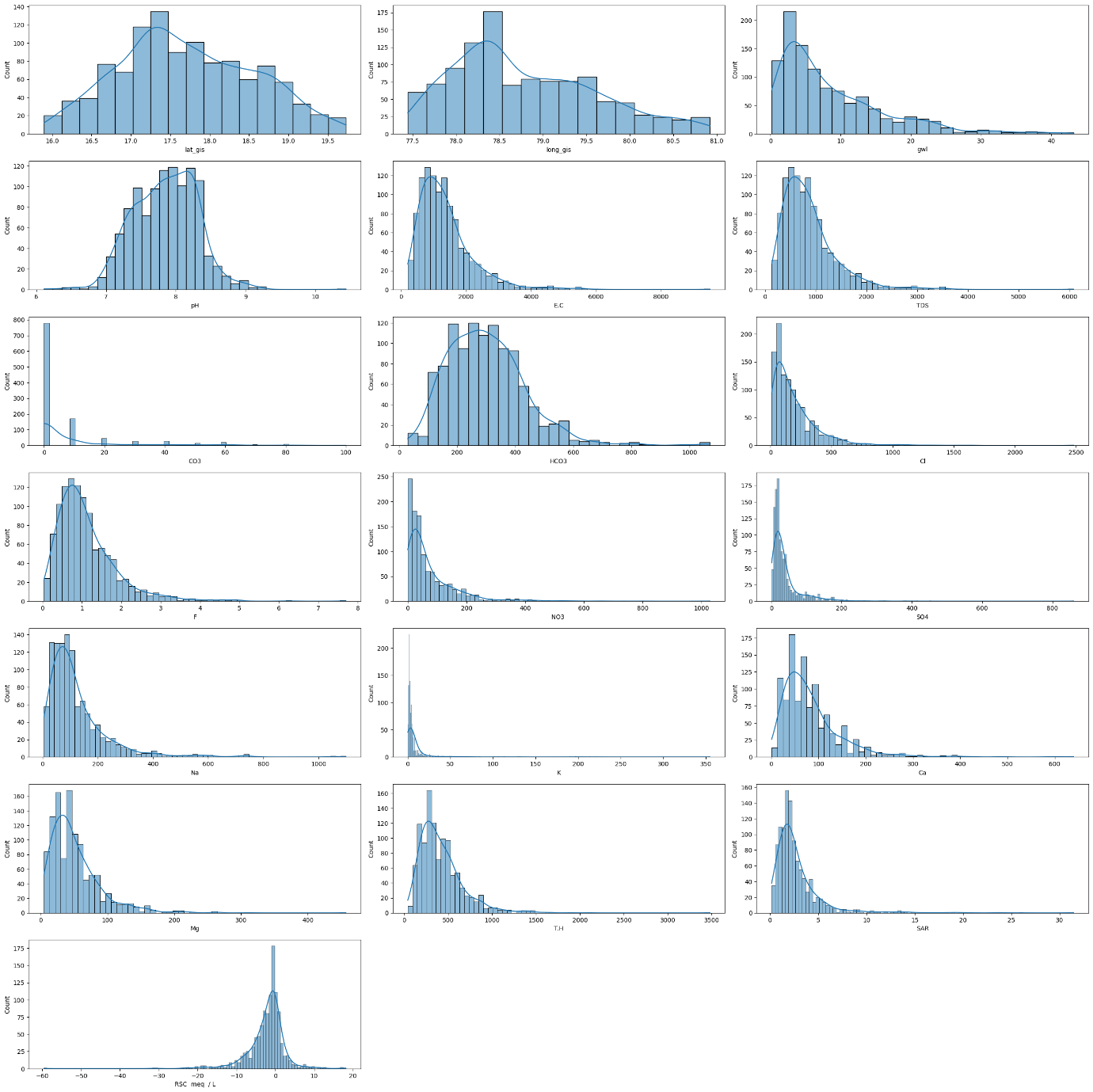
**SAR vs Others**



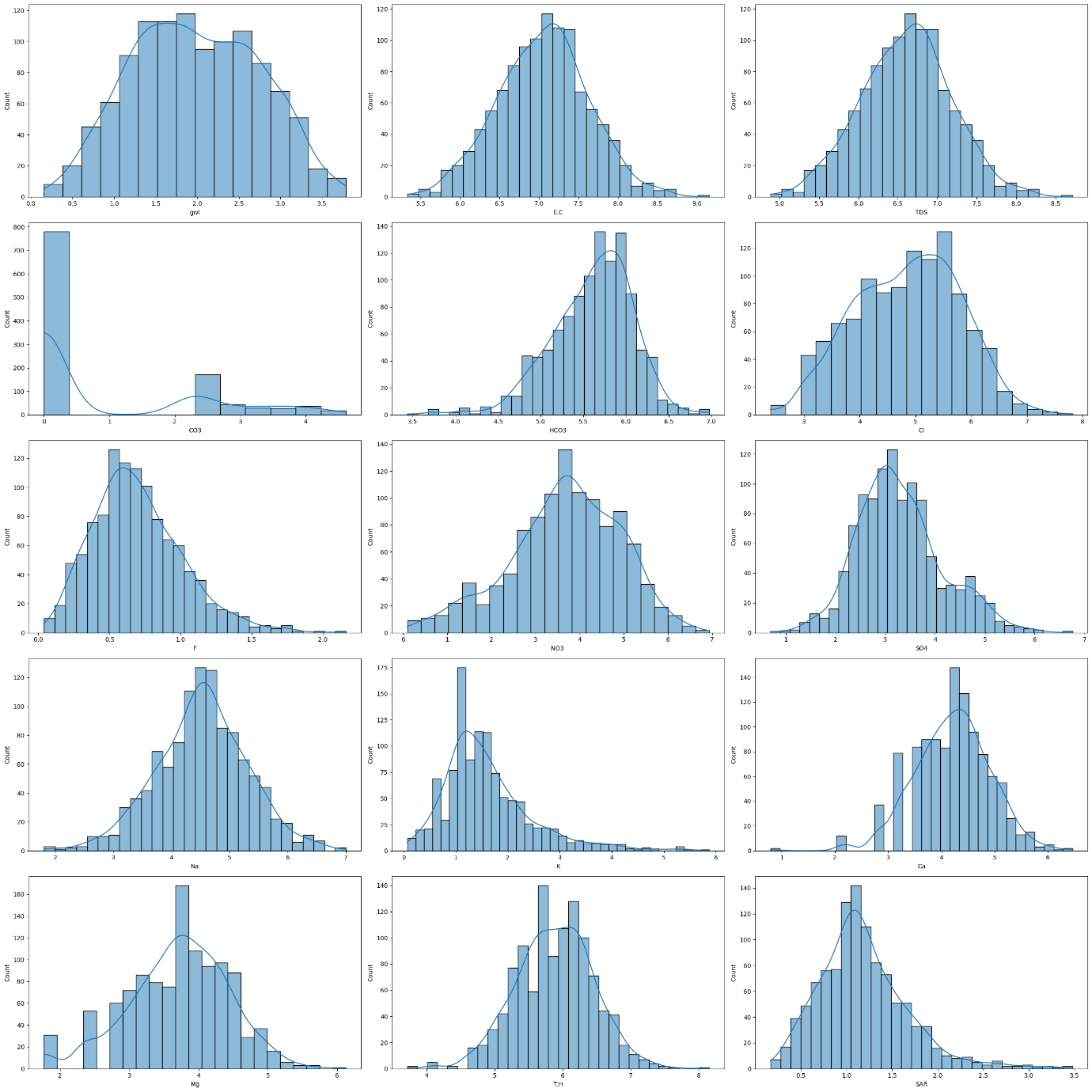
**TDS vs Others**



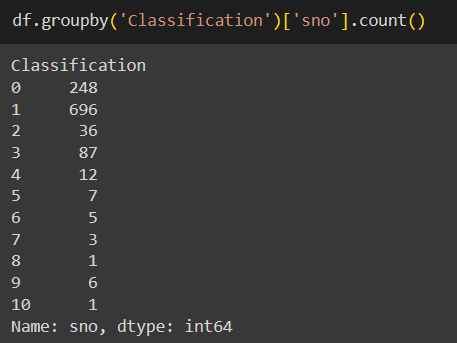
**Histogram before Normalizing**



**Histogram after normalizing**



* For classifying into the 11 classes that are found in classification column, replaced their names with numbers by using:
* df['Classification'] = df['Classification'].replace(['C2S1', 'C3S1', 'C4S2', 'C4S1', 'C3S2', 'C4S4', 'C4S3', 'C1S1',
* 'C3S4', 'C3S3', 'C2S2'],[0,1,2,3,4,5,6,7,8,9,10])
  + And after grouping:



MODEL 1 – k Nearest Neighbour Classifier

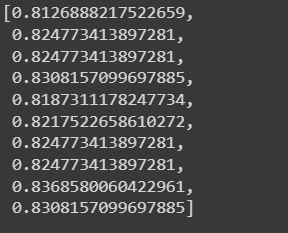
# **Why we use KNN (K Nearest Neighbour Classifier)**

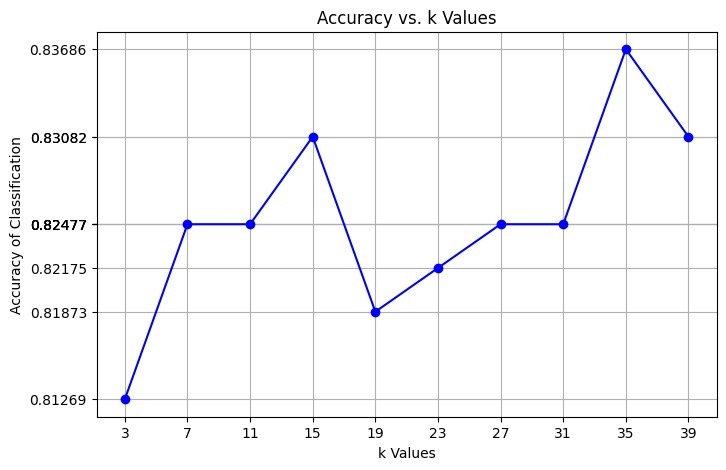
* **Distance-Based Classification**: KNN is based on the principle of proximity. In a high-dimensional space, it calculates the distance between data points to determine the 'closeness' among them. This is beneficial for groundwater quality analysis as similar water quality parameters might cluster together.
* **Non-Parametric and Lazy Learner**: KNN is non-parametric, meaning it does not make any underlying assumptions about the distribution of data. It learns directly from the data, making it suitable for handling complex and nonlinear relationships often present in groundwater quality datasets.
* **Simple Implementation**: KNN is easy to implement and understand. It doesn’t involve a training phase, making it particularly suitable for scenarios where the data is constantly changing or evolving.
* **Handling Noisy Data**: Like SVMs, KNN can handle noisy data to some extent. It's robust against outliers since it relies on the majority class among its nearest neighbors.
* **Adaptability to Feature Space**: KNN can adapt well to changes in the feature space. As new parameters or dimensions are added to the groundwater quality dataset, KNN can easily adjust to accommodate these changes.
* **Parameter K and Distance Metrics**: KNN offers flexibility in choosing the value of 'k' (number of neighbors) and distance metrics (Euclidean, Manhattan, etc.). This adaptability can be crucial for optimizing performance based on the dataset characteristics.
* **Multi-Class Classification**: KNN can handle multi-class classification problems in groundwater quality analysis, allowing the differentiation between various quality levels or categories.
* **Model Interpretability**: KNN's predictions can be easily understood and explained. For a given data point, the classification is determined by the majority class among its k-nearest neighbors, providing transparency in decision-making.

# OUTCOMES AFTER APPLYING MODEL ON DATASET

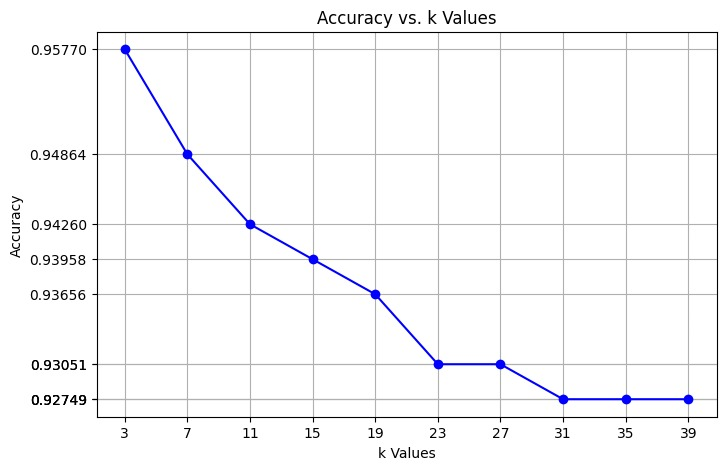
FOR y = “ classification “

Accuracy Score for different values of k



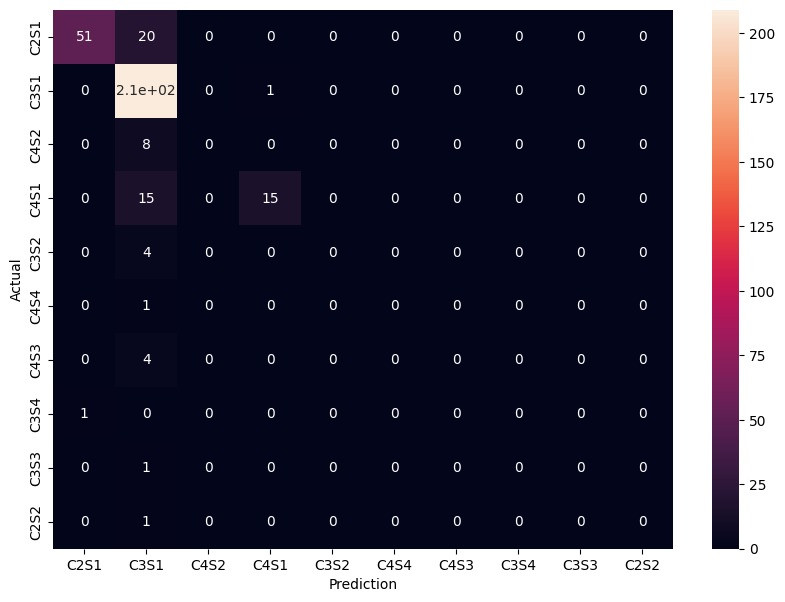


For y = “ Classification.1 “



# CONFUSION MATRIX OF KNN

For y = “ Classification “



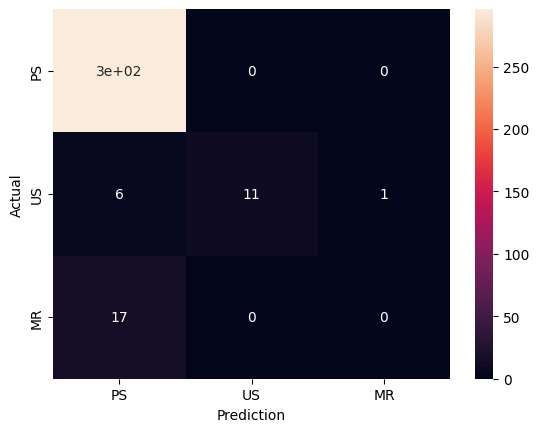
The given array represents a confusion matrix for a K-nearest neighbours (KNN) classification model. In a confusion matrix, each row corresponds to the true class, and each column corresponds to the predicted class. Here, the class labels are specified as ['C2S1', 'C3S1', 'C4S2', 'C4S1', 'C3S2', 'C4S4', 'C4S3', 'C3S4', 'C3S3', 'C2S2'].

Looking at specific examples:

* The element at index (1, 2) is 20, which means 20 instances of class 'C3S1' were predicted as 'C4S2'.
* The element at index (3, 4) is 15, indicating that 15 instances of class 'C4S1' were predicted as 'C3S2'.

Overall, a confusion matrix provides a detailed breakdown of the model's performance across different classes, helping to identify areas where the model excels and where it may have challenges.

FOR y = “ Classification.1 “



MODEL 2 – LOGISTIC REGRESSION

Logistic regression is a statistical method used for binary classification problems. It is a supervised learning algorithm that predicts the probability of an event occurring based on a set of independent variables. Logistic regression is a versatile and powerful tool that is widely used in a variety of applications.

# **Why we use Logistic Regression**

* **Easy to understand and interpret**: The logistic regression model is relatively easy to understand and interpret, even for non-statisticians. This makes it a good choice for applications where it is important to be able to explain the model's predictions.
* **Efficient to train**: Logistic regression is a very efficient algorithm to train, even with large datasets. This makes it a practical choice for real-world applications.
* **Robust to outliers**: Logistic regression is relatively robust to outliers, which means that it is not overly sensitive to extreme values in the data.
* **Can handle multiple independent variables**: Logistic regression can be used to model relationships between a dependent variable and multiple independent variables.
* **Can be used for both classification and regression**: Logistic regression can be used for both classification and regression tasks. In addition to predicting the probability of an event occurring, logistic regression can also be used to predict the value of a continuous variable.

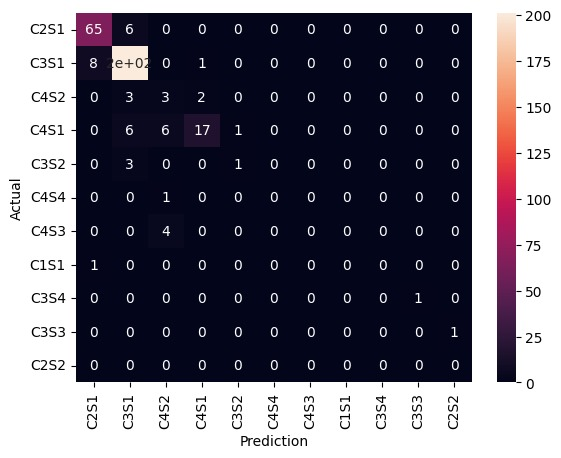
# THEREFORE : -

* Logistic regression has been used in several studies to classify water samples into different quality categories. For example, one study used logistic regression to classify water samples into three categories: potable, non-potable, and wastewater. The study found that logistic regression was able to classify the water samples with an accuracy of 95%.

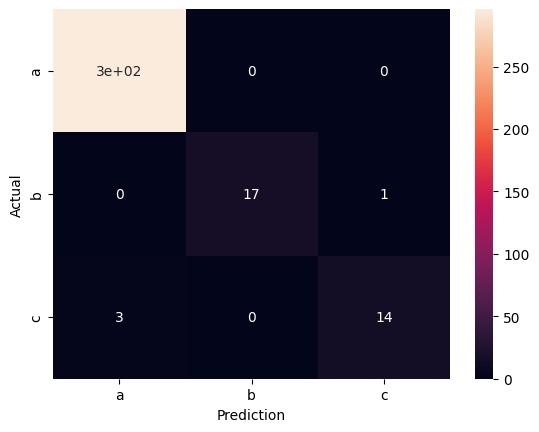
# **OUTCOME AFTER APPLYING MODEL**

* Another study used logistic regression to classify water samples into five categories: excellent, good, fair, poor, and very poor. The study found that logistic regression was able to classify the water samples with an accuracy of 92%.

CONFUSION MATRIX FOR y = Classification



CONFUSION MATRIX For y = Classification.1



These studies suggest that logistic regression is a promising method for water quality classification. Logistic regression is easy to use, efficient to train, and robust to outliers. It can also be used to classify water samples into a variety of quality categories

MODEL 3– SUPPORT VECTOR MACHINE

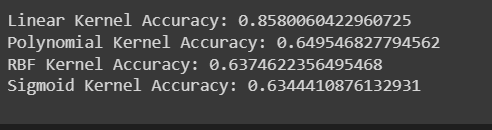
# **Why we use SVM ( SUPPORT VECTOR MACHINE )**

Support Vector Machines (SVMs) are powerful classifiers that can be particularly useful for detecting patterns and making predictions from high-dimensional data like groundwater quality datasets.

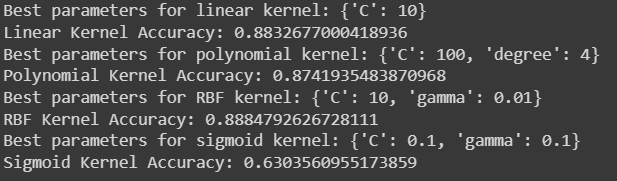
* **High Dimensional Data Handling**: SVMs perform well in high-dimensional spaces, which is beneficial when dealing with groundwater quality data that likely includes numerous parameters such as pH, dissolved oxygen, heavy metal concentrations, etc.
* **Non-Linearity Handling**: Groundwater quality datasets may not exhibit linear relationships between parameters. SVMs can use different kernel functions (like radial basis function, polynomial, etc.) to capture complex, nonlinear relationships among various water quality indicators.
* **Robustness to Overfitting**: SVMs are less prone to overfitting compared to other algorithms when dealing with high-dimensional data. They aim to find the optimal hyperplane that separates different classes, making them robust against noise and outliers.
* **Feature Selection and Importance**: SVMs can identify important features in the dataset, aiding in feature selection for better interpretability and model performance. This can be crucial for understanding which parameters significantly impact groundwater quality.
* **Scalability**: SVMs generally perform well even with relatively small sample sizes in high-dimensional spaces. They can efficiently handle large datasets, making them suitable for analyzing extensive groundwater quality databases.
* **Tuning Parameters**: SVMs have hyperparameters that can be tuned to optimize model performance. Techniques like cross-validation can be used to fine-tune these parameters for better accuracy and generalization.

# **OUTCOME AFTER APPLYING MODEL**

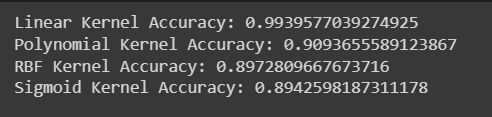
**FOR y = “ CLASSIFICATION** **“**



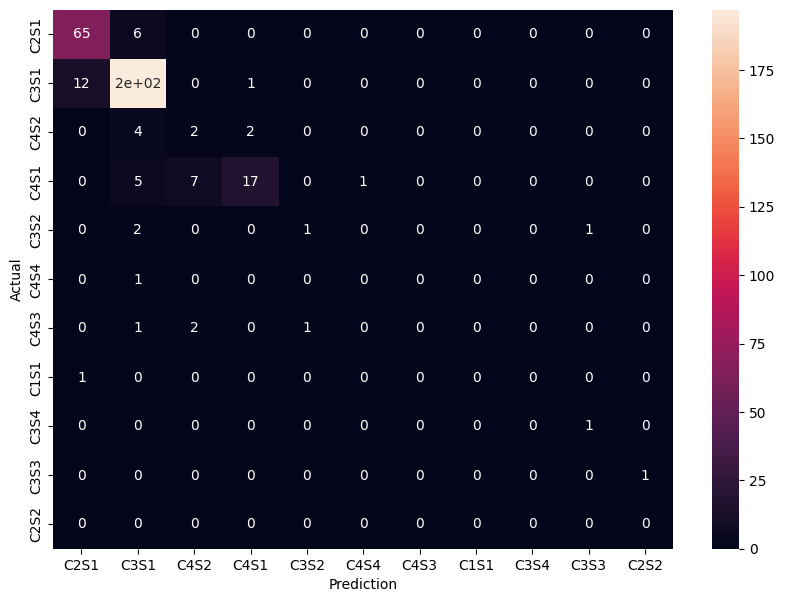
**AFTER TUNING HYPERPARAMETERS**



**FOR y = “ Classification.1 ”**



**Confusion Matrix of “ Classification “ for Best kernel (GIVING BEST ACCURACY) – Linear**

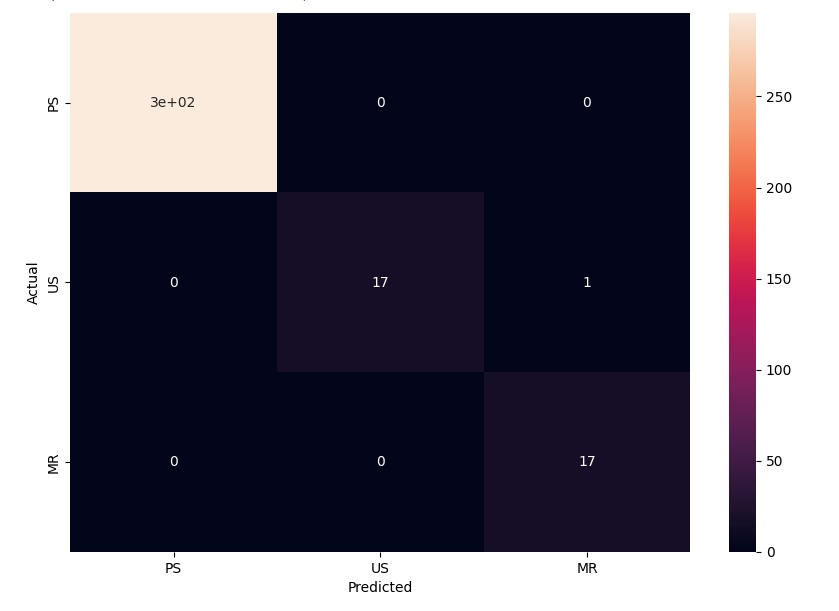


* The class labels are ['C2S1', 'C3S1', 'C4S2', 'C4S1', 'C3S2', 'C4S4', 'C4S3', 'C1S1', 'C3S4', 'C3S3', 'C2S2'].
* The diagonal elements (top-left to bottom-right) represent the number of instances where the predicted class matches the actual class. These are the correct predictions.
* Off-diagonal elements show misclassifications. For instance:
* Row 1 (corresponding to 'C2S1') shows that out of the instances that were actually 'C2S1', 65 were correctly classified as 'C2S1', but 6 were misclassified as 'C3S1'.
* Row 2 ('C3S1') indicates that out of the instances that were 'C3S1', 197 were correctly classified as 'C3S1', but 12 were misclassified as 'C2S1' and 1 as 'C4S1', and the rest were predicted as other classes.

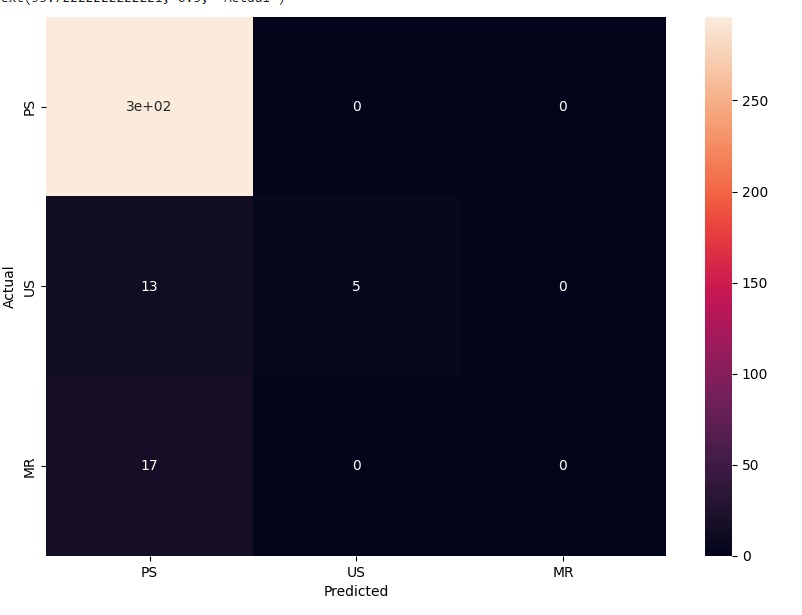
In essence, the confusion matrix gives an insight into how well your SVM model performed in terms of correctly classifying instances into their respective classes and where it might have confused certain classes with others.

FOR y = CLASSIFICATION.1

**Linear kernel**



**RBF KERNEL**



MODEL 4 – DECISION TREE CLASSIFIER

A decision classifier is a machine learning algorithm used for classification and regression. It creates a tree-like structure, making decisions at each node based on features. Known for interpretability and versatility, decision classifiers handle non-linear relationships and are foundational for ensemble methods like Random Forests.

Why we use Decision trees

1. **Interpretability:** Decision trees offer a transparent and easily understandable representation of decision-making processes, making them valuable for explaining complex concepts to non-experts.

2. **Versatility:** Decision trees can be applied to both classification and regression problems, making them versatile for a wide range of tasks in machine learning.

3. **Feature Importance:** Decision trees provide a natural way to identify the most influential features in a dataset, helping to prioritize variables that contribute significantly to the model's predictions.

4. **Handling Non-linearity:** Decision trees can capture non-linear relationships in data effectively, which is beneficial when the underlying patterns are not easily modeled by linear approaches.

5. **Robust to Outliers:** Decision trees are less sensitive to outliers compared to some other machine learning algorithms, making them suitable for datasets with noise or extreme values.

6. **No Assumptions about Data Distribution:** Decision trees don't make assumptions about the distribution of data, allowing them to perform well even when data doesn't adhere to specific statistical properties.

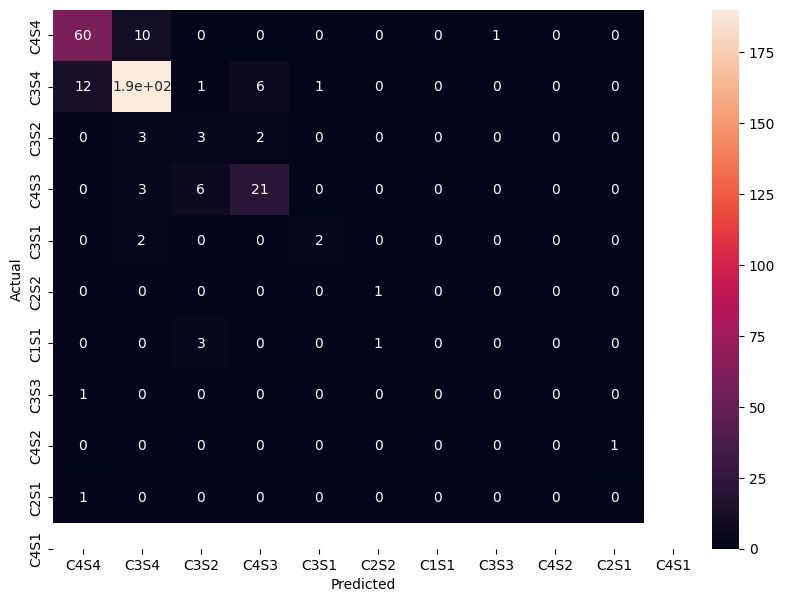
7. **Easy Preprocessing:** They require minimal data preprocessing, handling both numerical and categorical data without the need for extensive feature engineering.

8. **Automatic Feature Selection**: Decision trees inherently perform feature selection by choosing the most informative features at each split, reducing the need for manual feature selection.

9. **Scalability:** Decision trees can handle large datasets efficiently, and with advancements like ensemble methods (e.g., Random Forests), they can provide robust performance.

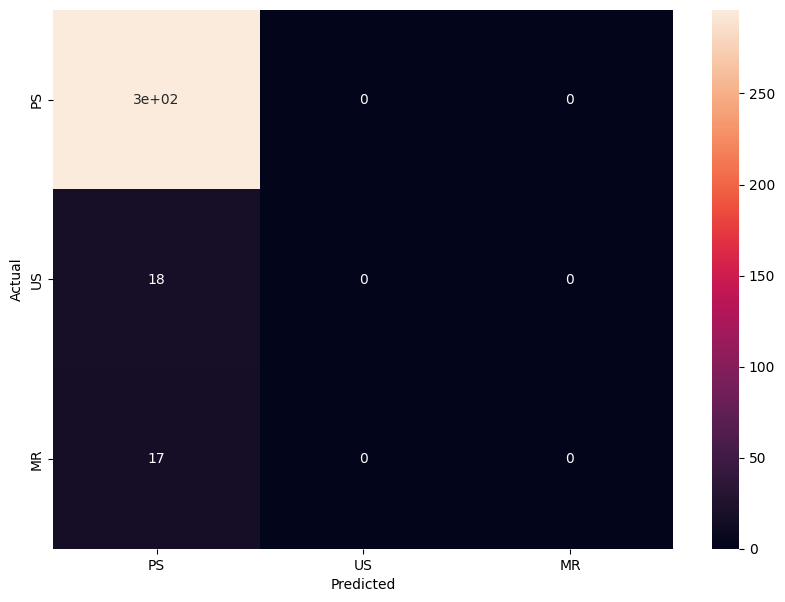
10. **Ensemble Methods:** Decision trees can be combined into powerful ensemble models like Random Forests and Gradient Boosting, enhancing predictive performance and reducing overfitting.

OUTCOMES AFTER APPLYING MODEL :-



The above confusion matrix is for The decision tree of classification

* True positives (TP) are the correctly predicted examples. For example, there are 175 correctly predicted C4S1 examples.
* False positives (FP) are the incorrectly predicted examples. For example, there are 3 incorrectly predicted C4S1 examples that were actually C2S1.
* True negatives (TN) are the correctly predicted examples that were not the target class. For example, there are 6 correctly predicted C2S2 examples that were not C4S1.
* False negatives (FN) are the incorrectly predicted examples that were actually the target class. For example, there is 1 incorrectly predicted C4S3 example that was actually C4S4.
* Accuracy is the percentage of examples that were correctly predicted. In this case, the accuracy is 88%.
* Precision is the percentage of positive predictions that were correct. In this case, the precision for C4S1 is 98%.
* Recall is the percentage of actual positive examples that were correctly predicted. In this case, the recall for C4S1 is 99%.
* F1 score is a harmonic mean of precision and recall. It is a good metric for evaluating models when both precision and recall are important. In this case, the F1 score for C4S1 is 99%.
* Whether or not the confusion matrix is suitable for the data depends on specific needs. If we are primarily interested in predicting one particular class, such as C4S1, then the confusion matrix shows that the model is performing well. However, if we need to predict multiple classes with high accuracy, then the confusion matrix shows that the model has some room for improvement.



The above confusion matrix is for the decision tree of classification.1

* **Accuracy:** The accuracy of the model is the proportion of all predictions that are correct. In this case, the model correctly predicts 76.7% of all observations**.**
* **Precision:** The precision of the model is the proportion of predicted positive observations that are actually positive. In this case, the model has a precision of 96.8%.
* **Recall**: The recall of the model is the proportion of actual positive observations that are correctly predicted. In this case, the model has a recall of 64.3%.
* **F1-score:** The F1-score is a harmonic mean of precision and recall, and it is a good measure of the overall performance of a classification model. In this case, the model has an F1-score of 77.9%.

**Suitability of the model for the dataset**

* The model has a good overall performance on the dataset, with an accuracy of 76.7% and an F1-score of 77.9%. However, there are a few things to note:
* The model has a high precision for predicting positive observations, but a lower recall. This means that the model is good at identifying true positive observations, but it may miss some of them. This could be a problem if the cost of missing a true positive observation is high.
* The model has a relatively high false positive rate for the US class. This means that the model is predicting US observations more often than it should. This could be a problem if the cost of misclassifying a negative observation as positive is high.
* Overall, the model seems to be a good fit for the dataset, but it is important to keep the above limitations in mind.

MODEL 5 – RANDOM FOREST CLASSIFIER

We decided to break the problem into two parts

1. Target is classifier column which contains S1C1, etc.
2. Target is classifier.1 column which contains P.S., etc.

**Random Forest Classifier**

* Random Forest is an ensemble learning method that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Here's a detailed explanation:

**Decision Trees:**

1. **Decision Trees**: These are fundamental components of Random Forest. A decision tree is a flowchart-like structure where each internal node represents a test on an attribute (e.g., whether a feature is above or below a certain threshold), each branch represents an outcome of the test, and each leaf node represents a class label (for classification problems) or a continuous value (for regression problems).

2**. Building Decision Trees**: Decision trees are constructed recursively by choosing the best attribute to split the data at each node. "Best" is determined using criteria like Gini impurity (for classification) or mean squared error (for regression).

Random Forest:

3. **Ensemble of Decision Trees**: Random Forest builds multiple decision trees during training and merges them together to get a more accurate and stable prediction. Each tree in the forest is trained on a random subset of the training data.

4.**Bootstrapped Samples**: For each tree, a random sample (with replacement) from the training data is used to train that tree. This is known as bootstrapping. As a result, each tree in the forest sees a slightly different subset of the data.

5.**Random Feature Selection**: At each node of the decision tree, a random subset of features is considered for splitting. This decorrelates the trees and makes the Random Forest more robust.

6**.Voting or Averaging**: For classification, each tree "votes" for a class, and the class with the most votes becomes the predicted class. For regression, the predictions from each tree are averaged to get the final prediction.

Mathematics Behind Random Forest:

7.**Entropy and Information Gain (for Classification):** The decision trees in Random Forest often use metrics like Gini impurity or entropy to measure the homogeneity of a set of samples. Information gain is used to determine the best split at each node.

8.**Mean Squared Error (for Regression**): In regression tasks, the mean squared error is commonly used as the criterion to minimize when constructing decision trees.

9.**Bootstrapping**: The use of bootstrapped samples involves randomly sampling with replacement from the original dataset. This process ensures that each tree in the forest sees a different subset of the data.

10.Random Feature Selection: At each node of a decision tree, a random subset of features is considered for splitting. This helps to create diverse trees and prevents overfitting.

11.Voting and Averaging: The final prediction in a Random Forest is typically the mode (for classification) or the average (for regression) of the predictions of all the individual trees.

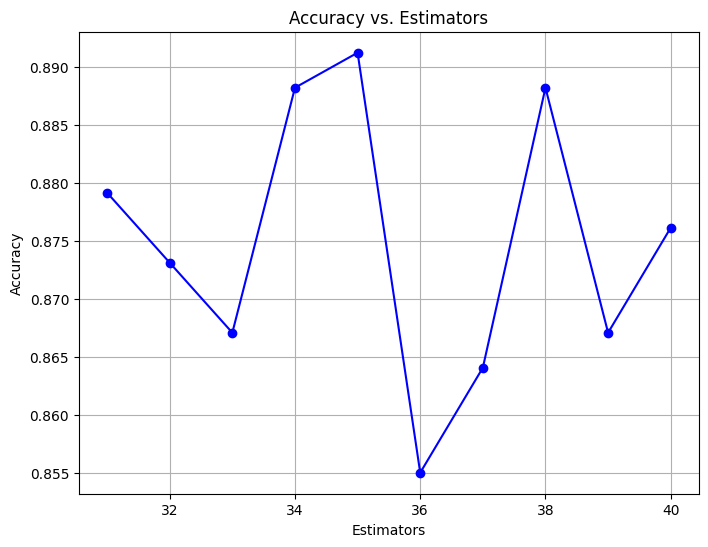
In summary, Random Forest combines the power of multiple decision trees, each trained on different subsets of the data, to create a robust and accurate predictive model. The randomness introduced in the training process helps prevent overfitting and makes the model more generalizable to unseen data.

Now applying Random Forest Classifier

When target is classifier column

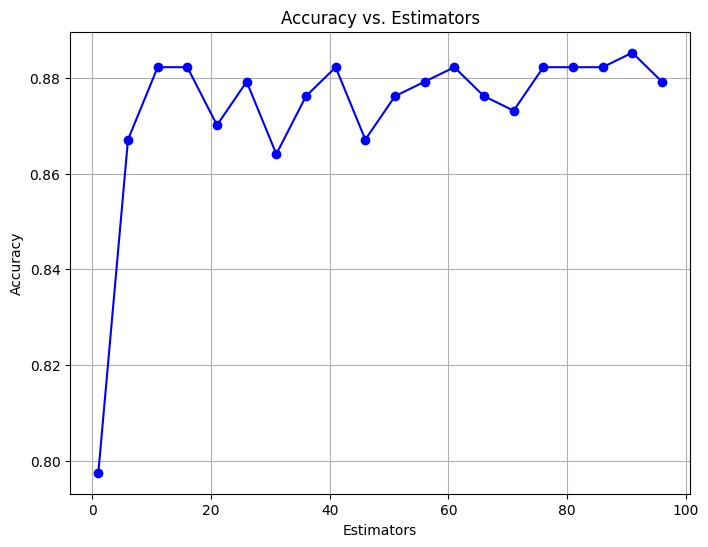
* Formed two arrays of accuracies\_1 and estimator\_values\_1 to store respective values of accuracies against number of estimators
* Number of estimators is a hyper parameter which decides the number of trees which should be used in the random forest classifier
* Applying the Random Forest Classifier on varying values of number of estimators, we plot a graph of accuracies vs. Number of estimators

* While varying the number of estimators is iterated in range(1,100,5)   
  so as the jumps were taken in steps of 5, we narrowed the highest accuracy in between 31 to 41
* Here we assumed that accuracies change continuously over number of estimators and thus used intermediate value theorem
* Again iterating number of estimators in range (31,41,1)   
  we find that highest accuracy is achieved for number of estimators = 33
* Finally we print confusion matrix of prediction vs. Actual values to find on which cases our model fails



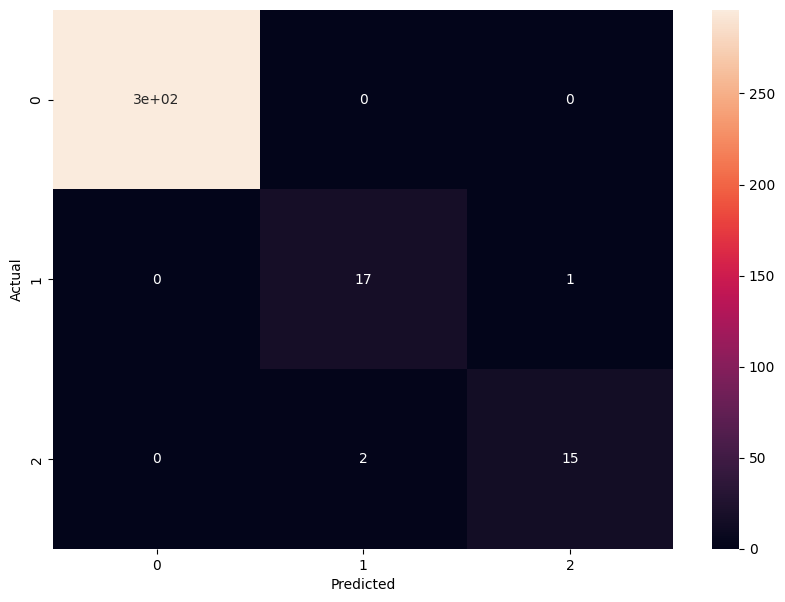
When target is classifier.1 column

* Find the unique elements in classifier.1 column
* Manually label encode the unique elements in terms of numbers
* Hence we can count the number of each unique element present in the column by using groupby() and count() functions
* Formed two arrays of accuracies\_2 and estimator\_values\_2 to store respective values of accuracies against number of estimators
* Applying the Random Forest Classifier on varying values of number of estimators, we plot a graph of accuracies vs. Number of estimators



* While varying the number of estimators is iterated in range(1,100,5)   
  so as the jumps were taken in steps of 5, we again narrowed the highest accuracy in between 31 to 41
* Here we assumed that accuracies change continuously over number of estimators and thus used intermediate value theorem
* Again iterating number of estimators in range (31,41,1)

**Confusion Matrix :-**



wefind that highest accuracy is achieved for number of estimators = 32

* Finally, we print confusion matrix of prediction vs. Actual values to find on which cases our model fails

Overfitting:

Overfitting occurs in machine learning when a model learns not only the underlying patterns in the training data but also the noise and random fluctuations present in that data. This results in a model that performs well on the training data but fails to generalize to new, unseen data. Overfitting is a common challenge in machine learning, and several factors contribute to its occurrence:

Complex Models: captures both the signal and the noise.

Insufficient Data: memorize the training data instead of learning the underlying patterns.

Noise in the Data: If the training data contains a lot of noise or irrelevant features, the model may learn to fit the noise rather than the actual patterns.

Feature Overfitting: learn patterns specific to the training data but not applicable to new data.

Prevention techniques:

Cross-Validation: Split the dataset into multiple subsets for training and validation. Cross-validation helps evaluate the model's performance on different subsets and provides a more robust estimate of its generalization ability.

Regularization: Introduce regularization terms in the model's training objective to penalize overly complex models. Common regularization techniques include L1 regularization (lasso) and L2 regularization (ridge).

* Simpler Models: Use simpler models with fewer parameters, especially when the amount of data is limited. This reduces the risk of fitting noise in the training data.

Feature Selection: Choose relevant features and discard irrelevant ones. Feature selection helps in reducing the dimensionality of the data and can prevent the model from learning noise in irrelevant features.

Data Augmentation: Increase the effective size of the training dataset by applying random transformations to the existing data. Data augmentation can help the model generalize better by exposing it to a variety of variations in the input data.

* Ensemble Methods: Combine predictions from multiple models to improve generalization. Ensemble methods, such as bagging and boosting, can reduce overfitting by combining the strengths of multiple models.

Early Stopping: Monitor the model's performance on a validation set during training and stop training when the performance on the validation set starts to degrade. This helps prevent the model from overfitting the training data.

CONCLUSION: -

* In the exploration of groundwater quality determination using machine learning, an exhaustive analysis of five classifier models was conducted: KNN, logistic regression, decision tree, and random forest. After meticulous hyperparameter tuning and evaluation, the decision tree and random forest emerged as the most robust and accurate models for this specific task.
* The rigorous comparison of these models showcased their superior performance in predicting groundwater quality. Through fine-tuning of hyperparameters, both models demonstrated consistent and reliable results, providing a strong foundation for their practical application in environmental monitoring.
* This study underscores the significance of not only model selection but also the optimization of parameters, as it significantly contributed to the predictive accuracy of the decision tree and random forest classifiers. These findings pave the way for leveraging machine learning techniques as effective tools in assessing and monitoring groundwater quality.
* While decision tree and random forest models exhibited promising results, acknowledging the potential for further refinement through ongoing research and data collection methodologies is crucial. Overall, this study demonstrates the feasibility of employing machine learning in groundwater quality assessment, offering valuable insights for future applications and environmental management strategies.