PAK AUSTRIA FACHHOCHSCHULE: INSTITUTE OF APPLIED SCIENCES AND TECHNOLOGY
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEEING

Machine Learning Lab

COMP-240L

Project Report

School of Computing Sciences

Prepared by:

Syed Hasnain Ali

Ubaid Ullah

Submitted to:

Ms. Sana Sikandar

Dr. Kamran Ullah

TABLE OF CONTENTS

Water-Quality-Analysis-Using-Machine-Learning

Abstract:

Determining water quality and potability holds critical importance for human health. Water quality measurement is extremely vital and drinking water that does not comply with quality standards can directly negatively affect human health. This study focuses on assessing water potability through machine learning algorithms. Our analysis involved eight diverse machine learning models: Logistic Regression, Decision Tree, KNeighbours, Gaussian Naive Bayes, Support Vector Machine (SVM), and Random Forest. The dataset consisted of 3276 records and 10 features that could affect water potability. These features include pH levels, hardness, solids, chloramines, sulfate, conductivity, organic carbon, trihalomethanes, turbidity, and potability status. We divided the dataset into 33% for testing and 66% for training. SVM achieved the highest accuracy rate of 66.08%, while the K-Nearest Neighbors (KNN) model reached the lowest accuracy rate of 57.56%. These results show the importance of using strong machines learning algorithms such as predicting water safety.

Introduction:

Access to safe drinking water is a fundamental human right and a critical need for all individuals. Ensuring water quality involves understanding various factors that affect potability and identifying contaminants that may render water unsafe for consumption. The Water Quality Analysis Using Machine Learning project aims to develop a predictive model to classify water samples as safe or unsafe for consumption based on key water quality indicators. This project will utilize machine learning techniques to analyze data on water quality and predict water potability.

Ensuring safe water quality is vital to prevent health risks globally, with water-related issues accounting for a significant number of deaths annually. Employing early detection methods is crucial to mitigate these risks. Utilizing machine learning algorithms offers a promising avenue for early prediction of water quality issues.

In this project, we assessed the efficacy of eight diverse machine learning models to predict water quality parameters. The models encompassed Logistic Regression, Decision Tree, KNeighbours, Gaussian Naive Bayes, Support Vector Machine (SVM), and Random Forest. The dataset consisted of 3276 records and 10 features such as pH levels, dissolved solids, turbidity, chlorine levels, conductivity, organic carbon, temperature, nitrates, phosphates, and water potability status. Training and testing of these models were conducted using a 66/33 train/test split. The analysis revealed that the SVM model yielded the highest accuracy, reaching 66.08%. On the other hand, K-Nearest Neighbors displayed the lowest accuracy, achieving 57.56 %. These results suggest the potential of machine learning in forecasting water quality, SVM as a potentially optimal model for this purpose.

We would like to give information about the algorithms we use in our project.

1.1. Logistic Regression:

Logistic regression is a supervised machine learning algorithm primarily employed for binary classification tasks. It utilizes a logistic function, or sigmoid function, to generate probability values between 0 and 1. For instance, in a scenario with two classes (Class 0 and Class 1), if the logistic function output exceeds a predefined threshold (typically 0.5), the instance is assigned to Class 1; otherwise, it belongs to Class 0. This method is an extension of linear regression tailored for classification problems.

One of its core functions is predicting a categorical dependent variable by leveraging independent variables. Consequently, the anticipated output must be discrete or categorical, such as Yes/No, 0/1, or true/false. However, instead of producing precise binary values, logistic regression yields probabilities ranging between 0 and 1, reflecting the likelihood of a data point belonging to a specific class.

Comparatively, logistic regression shares similarities with linear regression, yet their usage differs significantly. While linear regression tackles regression problems, logistic regression is geared towards solving classification challenges. Rather than fitting a linear regression line, logistic regression employs an "S"-shaped logistic function, which yields predictions predominantly as 0 or 1.

The sigmoid curve generated by the logistic function signifies the probability of an outcome, like determining if cells are cancerous or non-cancerous based on specific characteristics or whether a mouse is obese depending on its weight. Its significance in machine learning lies in its capability to furnish probabilities and classify new data, accommodating both continuous and discrete datasets.

Moreover, logistic regression proves versatile in classifying observations across diverse data types. It adeptly identifies the most influential variables crucial for effective classification, contributing to its widespread use in various domains.

1.2. Decision Tree:

A decision tree represents a hierarchical tree-like structure in which each internal node signifies a feature or attribute, branches illustrate the decision rules based on those features, and the leaf nodes convey the output or prediction of the algorithm. Its versatility makes it a valuable tool in supervised machine learning, catering to both classification and regression problems. Due to its intuitive representation and ability to handle complex datasets, the decision tree algorithm holds significant prominence in various applications within the field of machine learning.

During the training phase, the decision tree algorithm operates by recursively segmenting the training data into subsets based on attribute values. This segmentation occurs through selecting the optimal attribute that maximizes information gain or minimizes impurity, measured using metrics like entropy or Gini impurity. The goal is to effectively partition the data, reducing randomness or impurity within each subset. This process continues until a stopping criterion is met, which could be reaching a defined maximum depth for the tree, or a minimum number of samples required to split a node.

The strength of decision trees lies in their ability to handle non-linear relationships between features and target variables. Furthermore, decision trees are fundamental components of ensemble methods like Random Forest. In Random Forest, multiple decision trees are constructed using different subsets of the training data and features. The combination of predictions from these trees through a voting or averaging mechanism results in improved accuracy and robustness, making Random Forest one of the most powerful and widely used algorithms in machine learning.

1.3. K-Nearest Neighbors (KNN):

The K-Nearest Neighbors (KNN) algorithm is a fundamental method used in machine learning for both classification and regression tasks. It predicts the label or value of a new data point by considering its K closest neighbors in the training dataset. This supervised learning algorithm finds extensive application in pattern recognition, data mining, and intrusion detection.

One of the key advantages of KNN is its non-parametric nature, which means it does not assume any specific

distribution of the data. Unlike some other algorithms like Gaussian Mixture Models (GMM), KNN does not impose constraints based on data distribution. Instead, it relies on prior training data to classify coordinates into groups based on attributes, making it applicable in various real-life scenarios.

KNN's popularity lies in its simplicity and ease of implementation in machine learning. It can handle both numerical and categorical data without requiring assumptions about the underlying data distribution, offering flexibility across different types of datasets. Additionally, its non-parametric approach makes it less sensitive to outliers compared to other algorithms, enhancing its robustness in practical applications.

The algorithm operates by identifying the K nearest neighbors to a given data point using distance metrics like Euclidean distance. Subsequently, the classification or value of the data point is determined by the majority vote or average of these K neighbors. This adaptability to different patterns and reliance on local data structure for predictions highlights the significance of KNN in various machine learning applications.

1.4. Gaussian Naïve Bayes:

Gaussian Naive Bayes stands as a machine learning classification technique rooted in a probabilistic approach. It assumes a normal distribution (also termed Gaussian distribution) for each class. This assumption means that the parameters associated with the classes follow a bell-shaped curve, emphasizing independence among parameters in predicting the output variable.

The model's underlying principle assumes that each parameter possesses an independent capability to predict the output variable, facilitating the prediction of the probability of a dependent variable being classified into each group.

During the prediction phase, Gaussian Naive Bayes combines the predictions for all parameters, ultimately yielding a probability for the dependent variable to belong to each group. The final classification is then assigned to the group with higher probability.

Gaussian distribution, synonymous with normal distribution, is a statistical model characterizing the distributions of continuous random variables. Its defining trait is the bell-shaped curve, showcasing the probability density function. The crucial characteristics of the normal distribution include the mean (μ) and standard deviation (σ). The mean represents the average value within the distribution, while the standard deviation indicates the "width" or dispersion of the distribution around the mean, signifying how much the values deviate from the mean value.

1.5. Support Vector Machines (SVM):

Support Vector Machines (SVM) represent a supervised learning algorithm used in machine learning to address classification and regression tasks, excelling particularly in binary classification problems where data elements are grouped into two categories.

The fundamental goal of an SVM is to identify the optimal line, known as the decision boundary or hyperplane in high-dimensional feature spaces, that effectively separates data points belonging to different classes. This separation aims to maximize the margin, which signifies the distance between the hyperplane and the closest data points of each category, enabling clear distinction between classes.

SVMs are instrumental in analyzing intricate data that cannot be linearly separated. Nonlinear SVMs achieve this by employing a mathematical technique that transforms data into higher-dimensional space, facilitating boundary identification.

The working principle of SVMs revolves around transforming input data into a higher-dimensional feature space, enhancing the ability to ascertain a linear separation or classify the dataset more accurately. To achieve this transformation, SVMs utilize a kernel function. Instead of explicitly calculating transformed space coordinates, the kernel function enables implicit computation of dot products between transformed feature vectors, bypassing costly computations for extreme cases.

SVMs accommodate both linearly separable and non-linearly separable data through various kernel functions like linear, polynomial, or radial basis function (RBF) kernels. These kernels empower SVMs to capture intricate data relationships and patterns effectively.

During training, SVMs employ a mathematical formulation to pinpoint the optimal hyperplane in the higher-dimensional kernel space. This hyperplane's significance lies in maximizing the margin between different class data points while minimizing classification errors.

The choice of kernel function significantly influences SVM performance by mapping data from the original feature space to the kernel space. Optimal kernel function selection depends on data characteristics, emphasizing the critical role of the chosen kernel in SVM algorithm performance.

1.6. Random Forest:

The Random Forest Algorithm has garnered immense popularity due to its ease of use and versatility in addressing classification and regression challenges within machine learning. Its robustness lies in the capability to effectively manage intricate datasets while mitigating the risk of overfitting, rendering it an invaluable tool across various predictive tasks.

A standout feature of the Random Forest Algorithm is its proficiency in handling diverse types of data, including both continuous variables (common in regression problems) and categorical variables (typical in classification tasks). This versatility contributes to its effectiveness in performing well across both classification and regression tasks, making it a preferred choice in diverse applications.

This tutorial aims to delve into the inner workings of the Random Forest Algorithm, offering insights into its functionality and implementation in a classification task. By understanding how Random Forest operates, users can harness its strengths to effectively address classification challenges and leverage its adaptability for robust predictive modeling.

2. MATERIALS AND METHODS (or EXPERIMENTAL):

The dataset utilized in this study comprises records gathered from various sources, encompassing information on water quality parameters and potability status. It includes 3276 entries with 10 features known to influence water potability, such as pH levels, hardness, solids, chloramines, sulfate, conductivity, organic carbon, trihalomethanes, turbidity, and the status of water potability. The dataset was preprocessed to handle missing values and outliers before the modeling phase.

This study assessed the performance of eight diverse machine learning models to predict water potability. The models considered were Logistic Regression, Decision Tree, K-Nearest Neighbors, Gaussian Naive Bayes, Support Vector Machine (SVM), and Random Forest. Each model was implemented using widely adopted libraries in Python, such as Scikit-learn.

Before training the models, the dataset underwent preprocessing steps, including feature scaling, and splitting the dataset into training and testing subsets. Feature scaling involves standardizing numerical attributes to

ensure uniform scales across all features. The dataset split into 33% for testing and 66% for training purposes using a stratified sampling technique.

The machine learning models were trained on the training dataset and evaluated using the test dataset. Performance metrics such as accuracy, precision, recall, and F1-score were computed to assess each model's predictive capability. Hyperparameter tuning was performed using cross-validation to optimize model performance.

Statistical analyses were conducted using Python libraries to derive insights into feature importance and correlation among variables. Visualization techniques including histograms, correlation matrices, and feature importance plots were employed to enhance the understanding of the dataset's characteristics and the models' behavior.

Models	**Accuracy Score**	**Codes**	**Confusion Matrix**
Logistic Regression	60.11%	*Figure 1*	*Figure 2*
Decision Tree	62.62%	*Figure 3*	*Figure 4*
K-Nearest Neighbors	57.56%	*Figure 5*	*Figure 6*
Gaussian Naive Bayes	61.39%	*Figure 7*	*Figure 8*
Support Vector Machine (SVM)	66.08%	*Figure 9*	*Figure 10*
Random Forest	62.84%	*Figure 11*	*Figure 12*

Table 1: Models used in predicting water quality, their accuracy rates, confusion matrix codes, and visuals.

2.1. Figures:

```
# Logistic Regression
model_lg = LogisticRegression()
model_lg.fit(X_train, y_train)
y_pred_lg = model_lg.predict(X_test)
cm_lg = confusion_matrix(y_test, y_pred_lg)
print("Confusion Matrix - Logistic Regression:")
print(cm_lg)
```

Figure 1

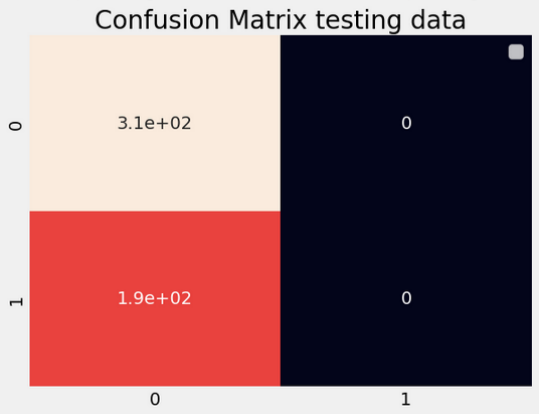

Figure 2

```
# Decision Tree Classifier
model_dt = DecisionTreeClassifier()
model_dt.fit(X_train, y_train)
y_pred_dt = model_dt.predict(X_test)
cm_dt = confusion_matrix(y_test, y_pred_dt)
print("Confusion Matrix - Decision Tree Classifier:")
print(cm_dt)
```

Figure 3

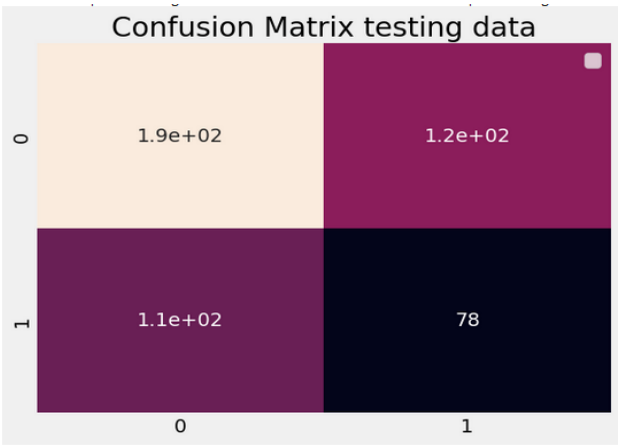

Figure 4

```python
# KNeighbours
model_kn = KNeighborsClassifier()
model_kn.fit(X_train, y_train)
y_pred_kn = model_kn.predict(X_test)
cm_kn = confusion_matrix(y_test, y_pred_kn)
print("Confusion Matrix - KNeighbours:")
print(cm_kn)
```

Figure 5

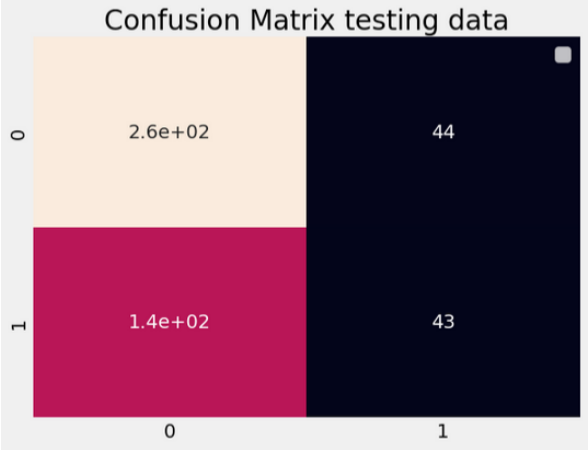

Figure 6

```python
# Gaussian Naive Bayes
model_nb = GaussianNB(var_smoothing=best_params_nb['var_smoothing'])
model_nb.fit(X_train, y_train)
y_pred_nb = model_nb.predict(X_test)
cm_nb = confusion_matrix(y_test, y_pred_nb)
print("Confusion Matrix - Gaussian Naive Bayes:")
print(cm_nb)
```

Figure 7

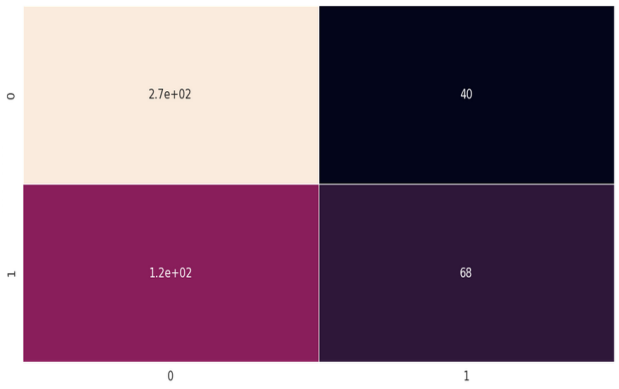

Figure 8

```
# SVM
model_svm = SVC(kernel='rbf', random_state=42)
model_svm.fit(X_train, y_train)
y_pred_svm = model_svm.predict(X_test)
cm_svm = confusion_matrix(y_test, y_pred_svm)
print("Confusion Matrix - SVM:")
print(cm_svm)
```

Figure 9

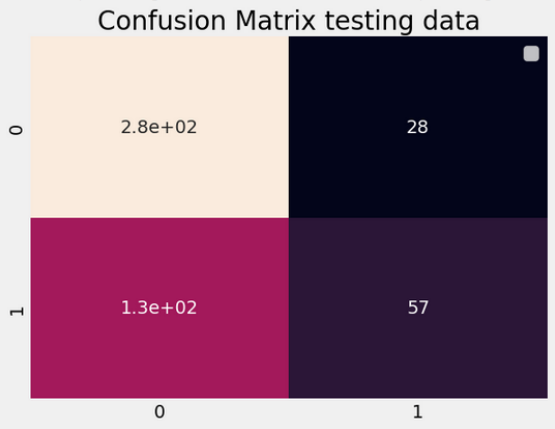

Figure 10

```
[ ] from sklearn.ensemble import RandomForestClassifier
    from sklearn.metrics import accuracy_score
```

```
# create the model
model_rf = RandomForestClassifier(n_estimators=500, oob_score=True, random_state=100)


# fitting the model
model_rf=model_rf.fit(x_train, y_train)

y_pred = model_rf.predict(x_test)
print(accuracy_score(y_test,y_pred))
```

Figure 11

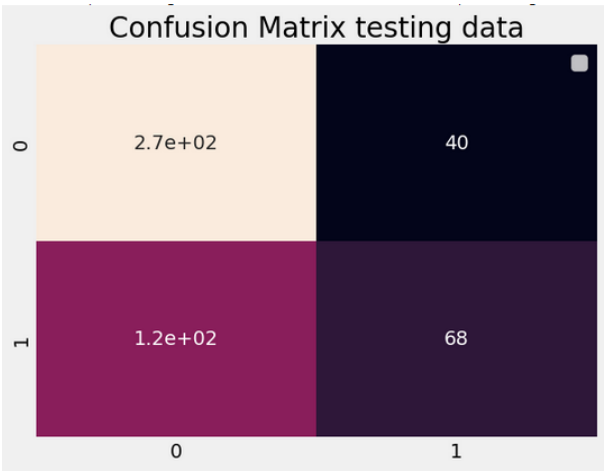

Figure 12

2.2. Accuracy Comparison Between Different Models on Same Dataset:

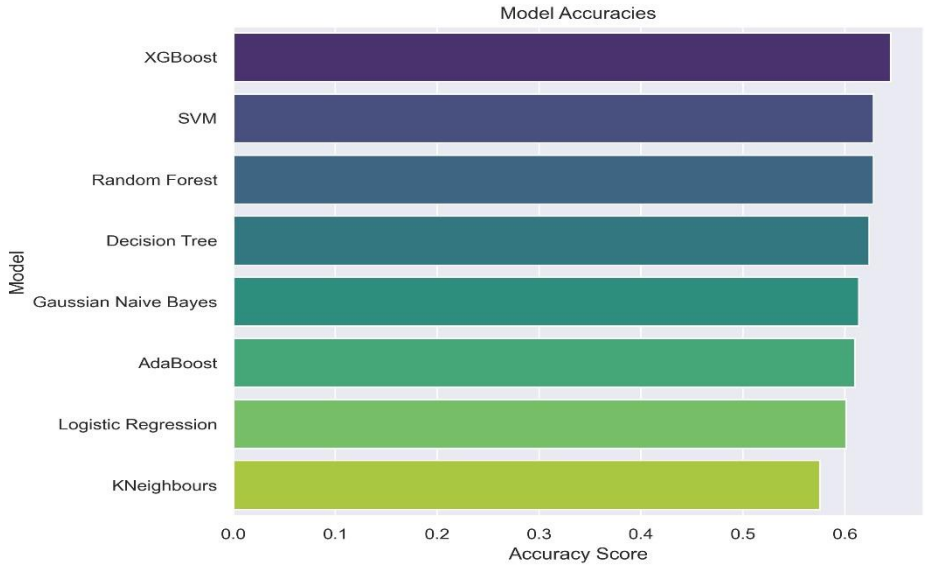

Figure 13

2.3. Feature Importance In SVM:

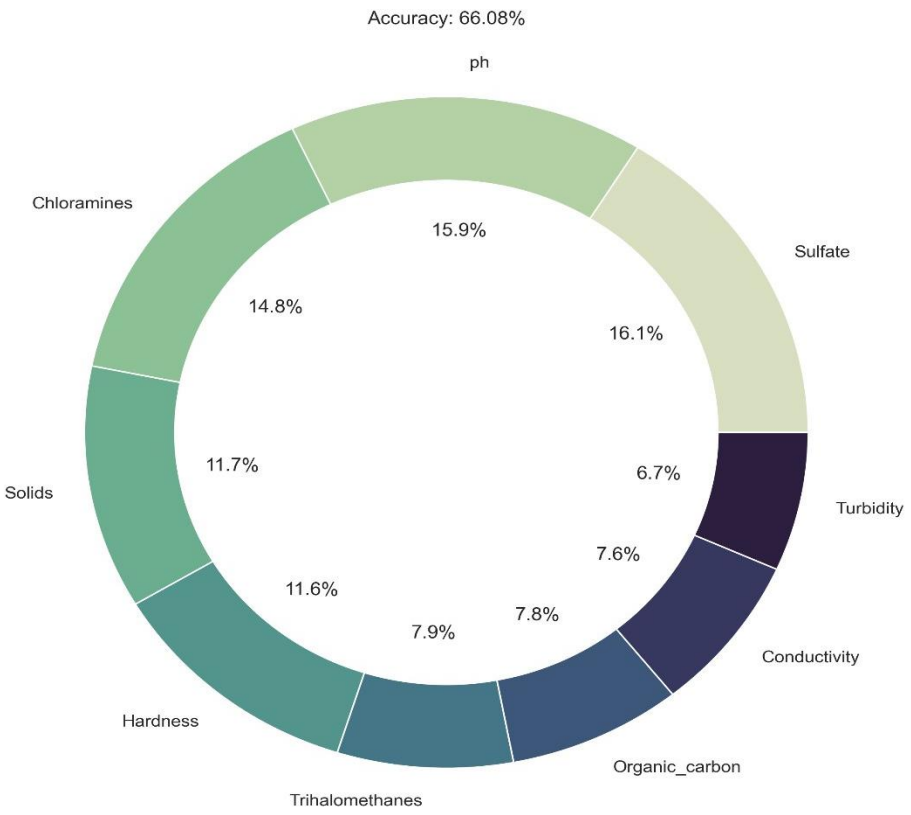

Figure 14

3. Results:

The Logistic Regression model has an accuracy of 60.11%, the Decision Tree model has an accuracy of 62.62%, the K-Nearest Neighbors (KNN) model has an accuracy of 57.56%, the AdaBoost model has an accuracy of 60.98%, the Gaussian Naïve Bayes model has an accuracy of 61.39%, the Support Vector Machines (SVM) model has an accuracy of 66.08%, the Random Forest model has an accuracy of 62.84%. So, the Support Vector Machines (SVM) model has the highest accuracy, with a score of 66.08% in *Figure 13*. The weight ratios of the features in the Support Vector Machines (SVM) model with the highest accuracy rate are illustrated in *Figure 14.*

4. Getting Too High Values in True Positive and False Negative:

The values in the confusion matrix (**2.7×10^2**etc.) represent the predictions made by the model, specifically True Positives, False Positives, True Negatives, and False Negatives. The reason for high values in **True Positives** (correct predictions for one class) and **False Negatives** (misclassified actual positives) could be attributed to several factors in dataset and model.

4.1 Interpreting High True Positives and False Negatives:

- **High True Positives (e.g., 2.7×10^2etc)**:
    - Indicates the model is correctly predicting a large number of cases for the most common class

(e.g., dot balls). This is expected in datasets with a dominant class.

- **High False Negatives (e.g., 1.2×10^2etc)**:
    - Suggests the model is failing to predict positives correctly for some classes, likely due to an imbalance or inability to capture class-specific features.

4.2 Reasons for High True Positives and False Negatives:

4.1.1 Class Imbalance:

- Some datasets often have imbalanced distributions for attributes (e.g., hardness of water or pH compared to other locations). If the majority class (e.g., 0) dominates the data, the model predicts that class will be more often, leading to higher counts in True Positives for the dominant class and False Negatives for the minority classes.

4.1.2 Model Bias Toward Majority Class:

- A model like Random Forest may prioritize predicting the majority class (e.g., dot balls) due to its frequency. While it performs well in the majority class (high True Positives), it fails to detect the minority classes (high False Negatives).

4.1.3 Threshold for Binary Classification:

- If the threshold for classification (default 0.5 for probability-based models) is not optimal, it may classify most samples as the majority class, inflating True Positives and False Negatives.

4.1.4 Limited Model Complexity:

- If the model is not complex enough to capture subtle patterns (e.g., interactions between features), it might fail to distinguish between certain classes, leading to misclassifications.

4.1.5 Lack of Critical Features:

- Water Quality is influenced by several factors like source, location, and surrounding conditions. If these features are missing, the model may struggle to generalize, causing more errors for certain outcomes.

4.1.6 Overfitting or Underfitting:

- An overfit model might memorize patterns in the training data but perform poorly on unseen data, leading to misclassifications.
- An underfit model lacks the capacity to learn the necessary decision boundaries, inflating errors like False Negatives.

5. Discussion:

The results of this study suggest that machine learning models may be useful in predicting whether water is drinkable. Among the eight models tested, the Support Vector Machines (SVM) model had the highest accuracy with a score of 66.08%. This suggests that the Support Vector Machines (SVM) model may be the most suitable model for predicting water quality. However, considering that our world is rich in water resources, we can infer that the dataset used is relatively small and cannot represent the entire population. Therefore, it is important to note that the results of this study should be interpreted with caution. Further studies with larger and more diverse data sets are needed to confirm these findings.

6. Conclusion:

To sum up, this study demonstrated the potential of machine learning algorithms to help predict whether

water is drinkable by measuring water quality. Eight different models were evaluated on a dataset containing 3276 records and 10 features. The results showed that the Support Vector Machines (SVM) model had the highest accuracy with a score of 66.08%, while K-Nearest Neighbors has the lowest accuracy with a score of 57.56%. These findings suggest that machine learning models may be useful in predicting whether water is drinkable, and that Support Vector Machines (SVM)model may be the most suitable model for this purpose. Further research is needed to confirm these results and determine the most effective ways to apply these models in the clinical setting.