Project Proposal

Freshwater Potability Assessment & Prediction

Hardi Raval

Agenda

Introduction

Research Questions?

Data Source

Data Pre-Processing [Missing values, Class imbalance, correlation, Outlier Removal]

Data Analysis [Confusion matrix & Algorithm selection]

Evaluation methods and metrics [Accuracy, Precision, Recall, F1-score]

QA

Introduction

- Freshwater is essential because there's not much of it, but we need it for everything.
- It's vital for drinking, cleaning, growing food, and making electricity.
- Clean water keeps us healthy and supports all living things.
- We use data to check if water is safe to use.
- Protecting freshwater is crucial for us and the environment.

What are the Goals ?

Build

Build a model to reliably predict whether water is safe for drinking.

Understand

Identify the factors that affect water quality, including safety and portability.

Educate

Educate people about effects of different resources in water reservoir.

Research Questions?

What factors determine whether water is safe for drinking, and how do they interact?

How do things like pH, iron, and other substances affect whether water is safe to drink?

Can we develop a reliable model to predict water safety based on key chemical composition?

How can predictive modeling complement existing regulatory frameworks?

Data Source: kaggle

Water Quality Prediction Dataset:

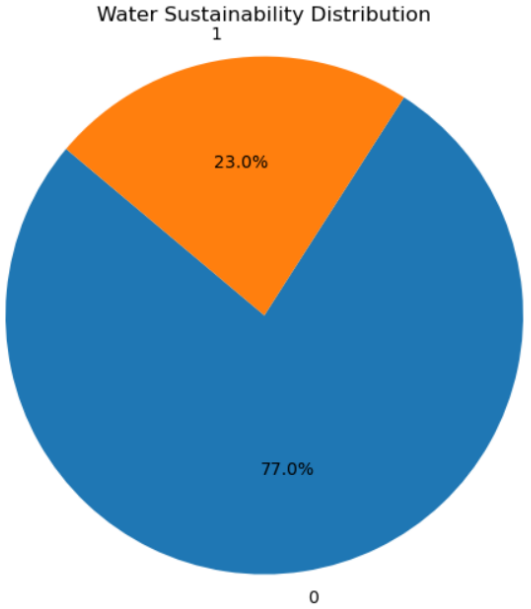
https://www.kaggle.com/datasets/deepikaarikesavan/water-quality

Having clean water to drink is really important for staying healthy, and it's something everyone should have.

Dataset Details:

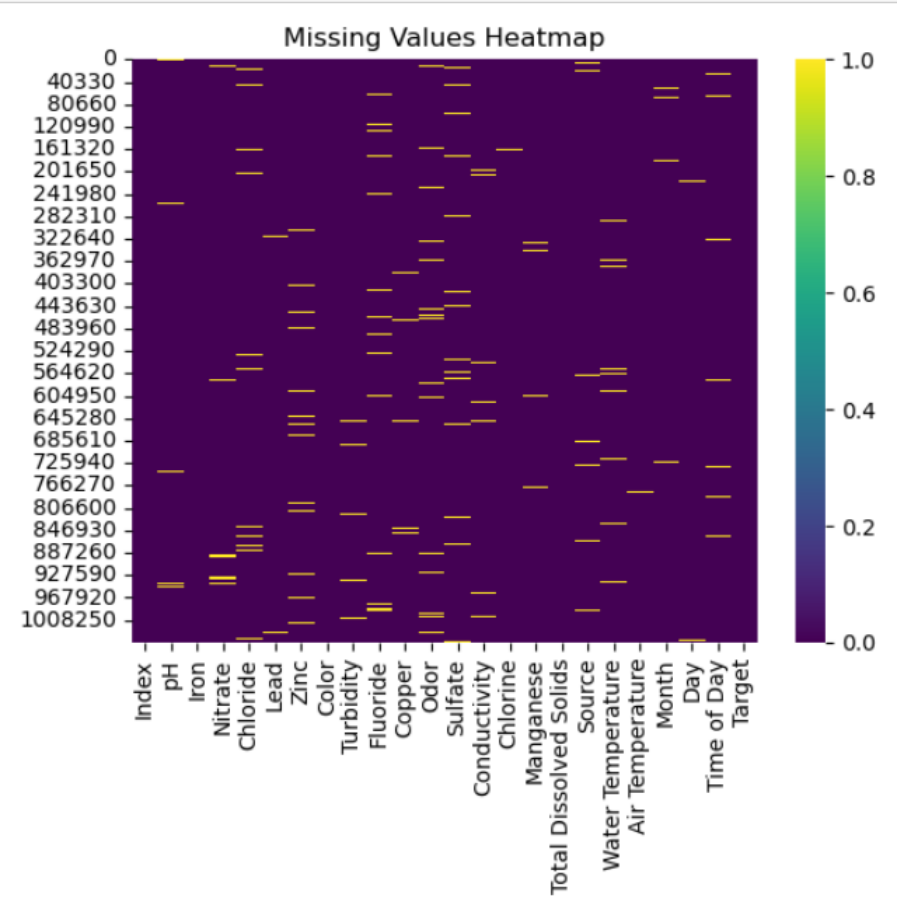
- Number of Columns: 24
  - Numerical Data: 21
  - Categorical Data: 3
- Number of Rows: 1048574

Water sustainability	**No. of Records** *(out of 10,48,574)*
Not sustainable **(Target:0)**	8,07,841 **(77.0%)**
Sustainable **(Target:1)**	2,40,734 **(23.0%)**



Handling Missing Values

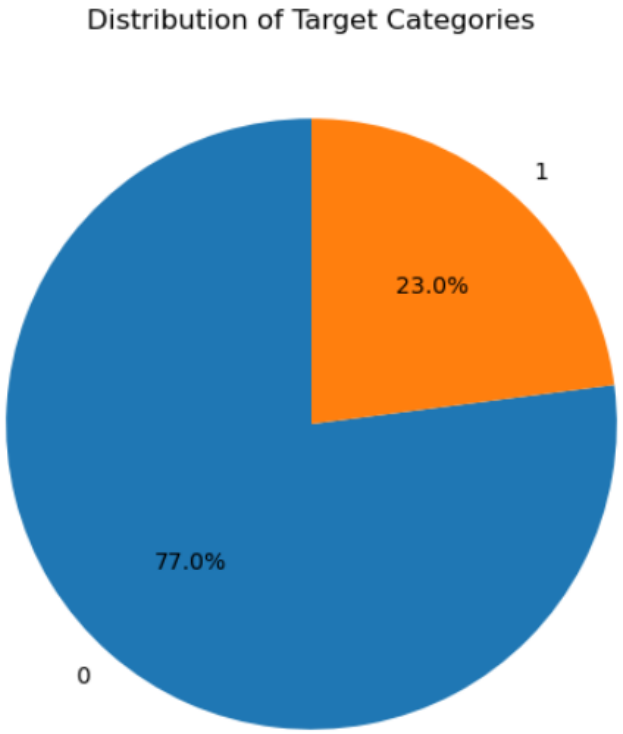
- Categorical Columns: **Fill missing values with mode (most frequent category).**
- Quantitative Columns**: Replace missing values with median.**
- Temporal Columns**: Use specialized methods like interpolation or forward/backward filling.**

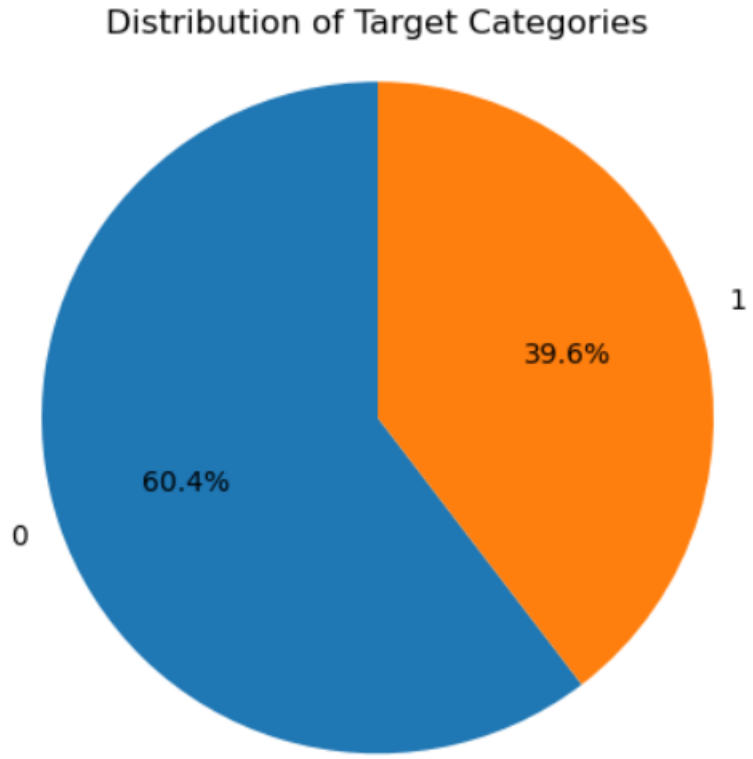



```
Missing Values:
 Index                    0.0
pH                        0.0
Iron                      0.0
Nitrate                   0.0
Chloride                  0.0
Lead                      0.0
Zinc                      0.0
Color                     0.0
Turbidity                 0.0
Fluoride                  0.0
Copper                    0.0
Odor                      0.0
Sulfate                   0.0
Conductivity              0.0
Chlorine                  0.0
Manganese                 0.0
Total Dissolved Solids    0.0
Source                    0.0
Water Temperature         0.0
Air Temperature           0.0
Month                     0.0
Day                       0.0
Time of Day               0.0
Target                    0.0
```

Class Imbalance

- Original class distribution: **77% and 23%.**
- Implemented **Down sampling** to address class imbalance.
- Down sampling involved reducing instances in the majority class.
- After down sampling, class distribution became **60.4% and 39.6%.**
- This approach ensures a more balanced representation of classes.



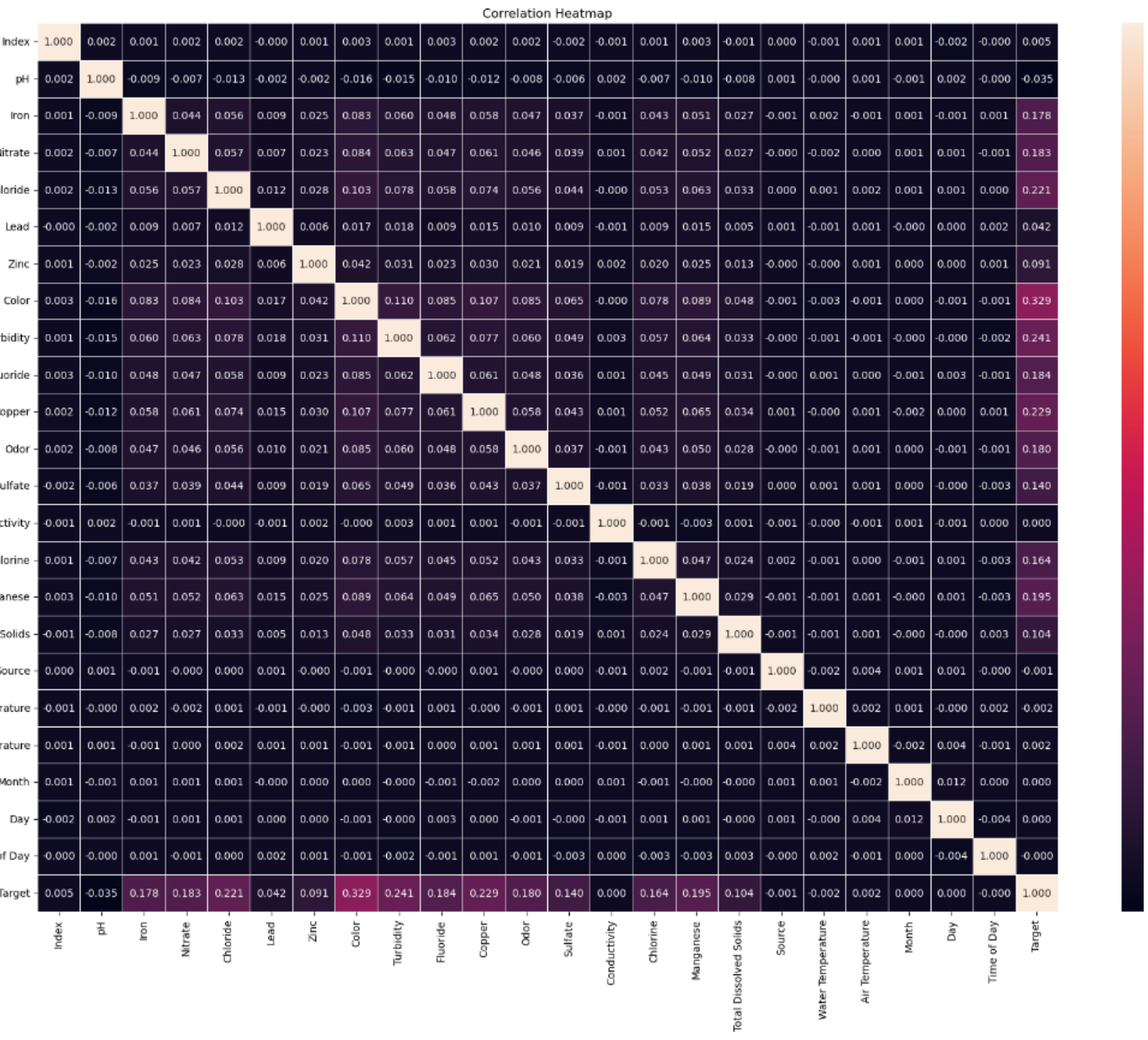


Label Encoding for Categorical Data

- Converted 'Source', 'Color', and 'Month' values to numerical equivalents.
- Utilized **Label Encoding** to assign unique numerical labels to categorical data.

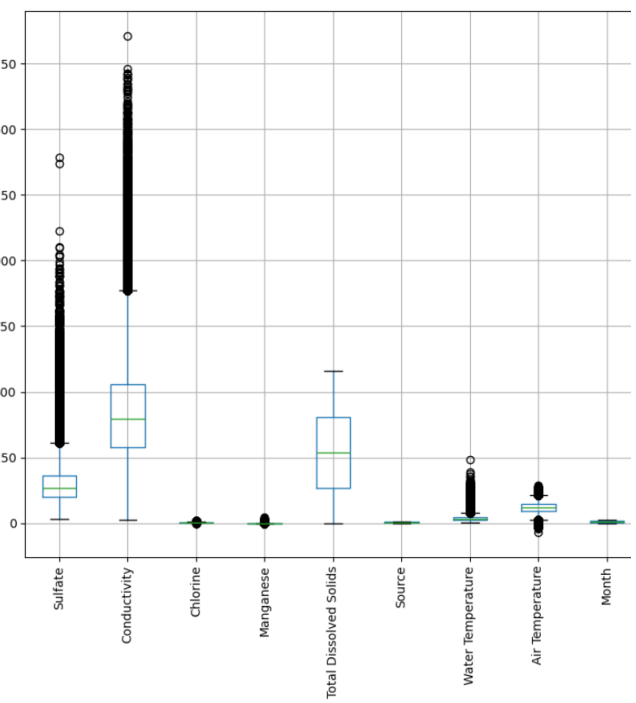
Lead	Zinc	Color	Turbidity	Fluoride	...	Chlorine	Manganese	Total Dissolved Solids	Source	Water Temperature	Air Temperature	Month	Day	Time of Day	Target
710000e-52	3.434827	Colorless	0.022683	0.607283	...	3.708178	2.270000e-15	332.118789	Well	16.467383	43.493324	January	29.0	4.0	0
850000e-94	1.245317	Faint Yellow	0.019007	0.622874	...	3.292038	8.020000e-07	284.641984	Lake	15.348981	71.220586	November	26.0	16.0	0
290000e-76	0.528280	Light Yellow	0.319956	0.423423	...	3.560224	7.007989e-02	570.054094	River	11.643467	44.891330	January	31.0	8.0	0
000000e-176	4.027879	Near Colorless	0.166319	0.208454	...	3.516907	2.468295e-02	100.043838	Ground	10.092392	60.843233	April	1.0	21.0	0
170000e-132	3.807511	Light Yellow	0.004867	0.222912	...	3.177849	3.296139e-03	168.075545	Spring	15.249416	69.336671	June	29.0	7.0	0

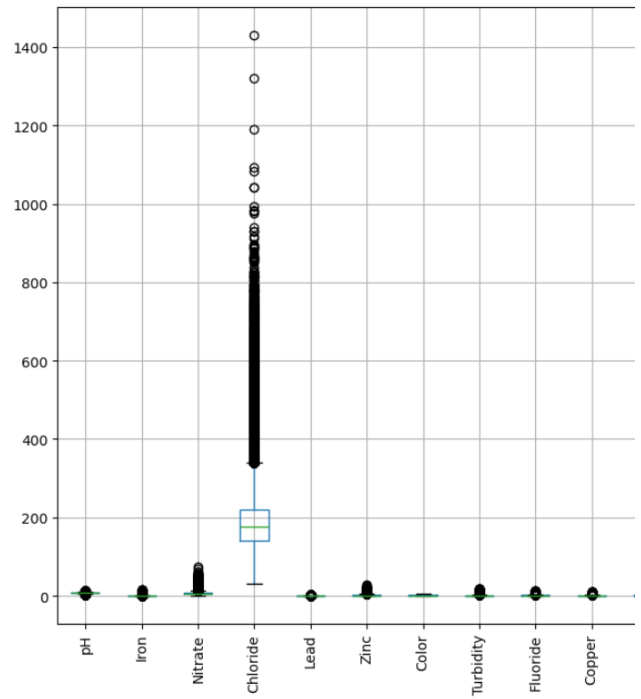
Lead	Zinc	Color	Turbidity	Fluoride	...	Chlorine	Manganese	Total Dissolved Solids	Source	Water Temperature	Air Temperature	Month	Day	Time of Day	Target
3.340000e-13	1.978032	2	0.248652	1.691318	...	5.966624	1.152750e-02	52.369446	5	10.404332	42.508011	3	12.0	23.0	1
4.860000e-39	1.128556	3	0.000030	0.765429	...	2.635055	1.672900e-04	232.941036	4	12.522236	63.289036	6	4.0	1.0	1
1.890000e-48	2.848238	0	0.081378	1.610560	...	3.265975	7.470000e-12	330.241630	3	15.644198	84.323390	4	28.0	16.0	1
1.920000e-28	0.400466	0	0.154662	0.070338	...	3.287618	2.900000e-07	80.026017	5	18.005467	55.267615	3	21.0	9.0	1
1.440000e-13	1.923391	3	0.010267	1.878288	...	3.505797	1.630000e-08	67.661410	4	48.176181	58.447806	4	11.0	17.0	1

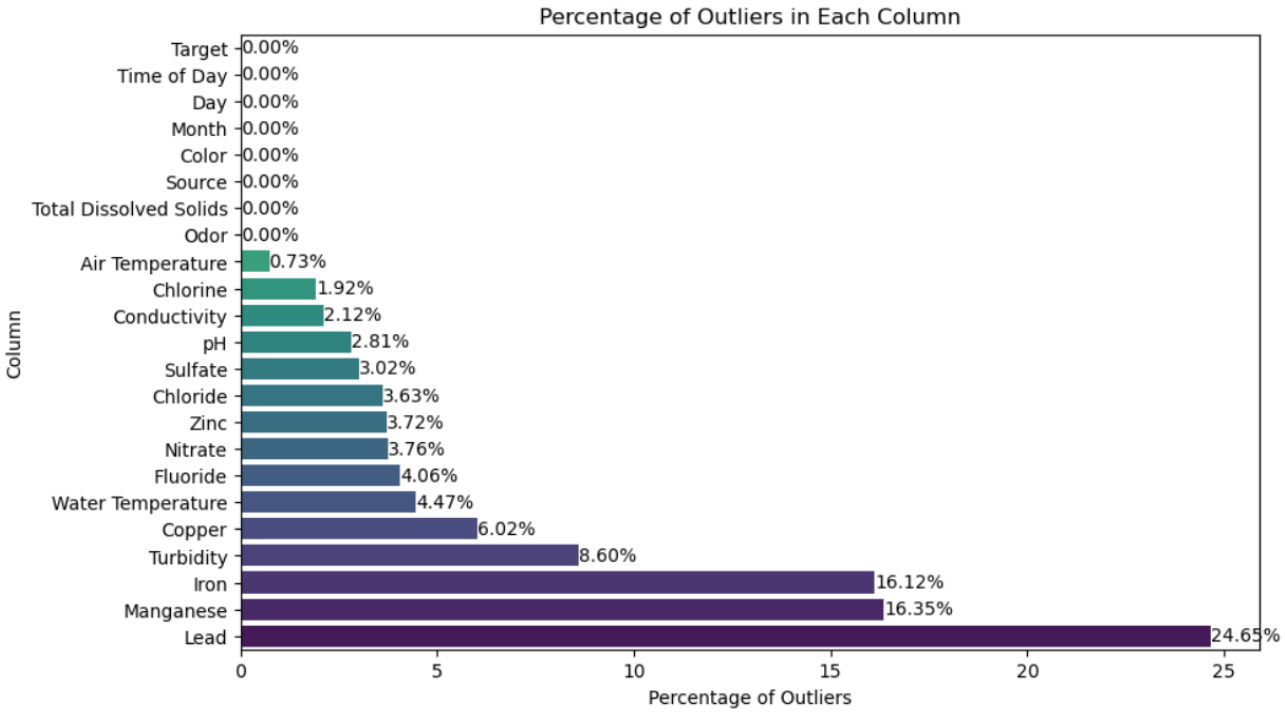


Correlation Heatmap

- **Correlation Analysis Results**
  - No significant correlation between target label and features.
  - Highlights the need for further in-depth analysis.
  - Indicates potential hidden relationships requiring exploration.





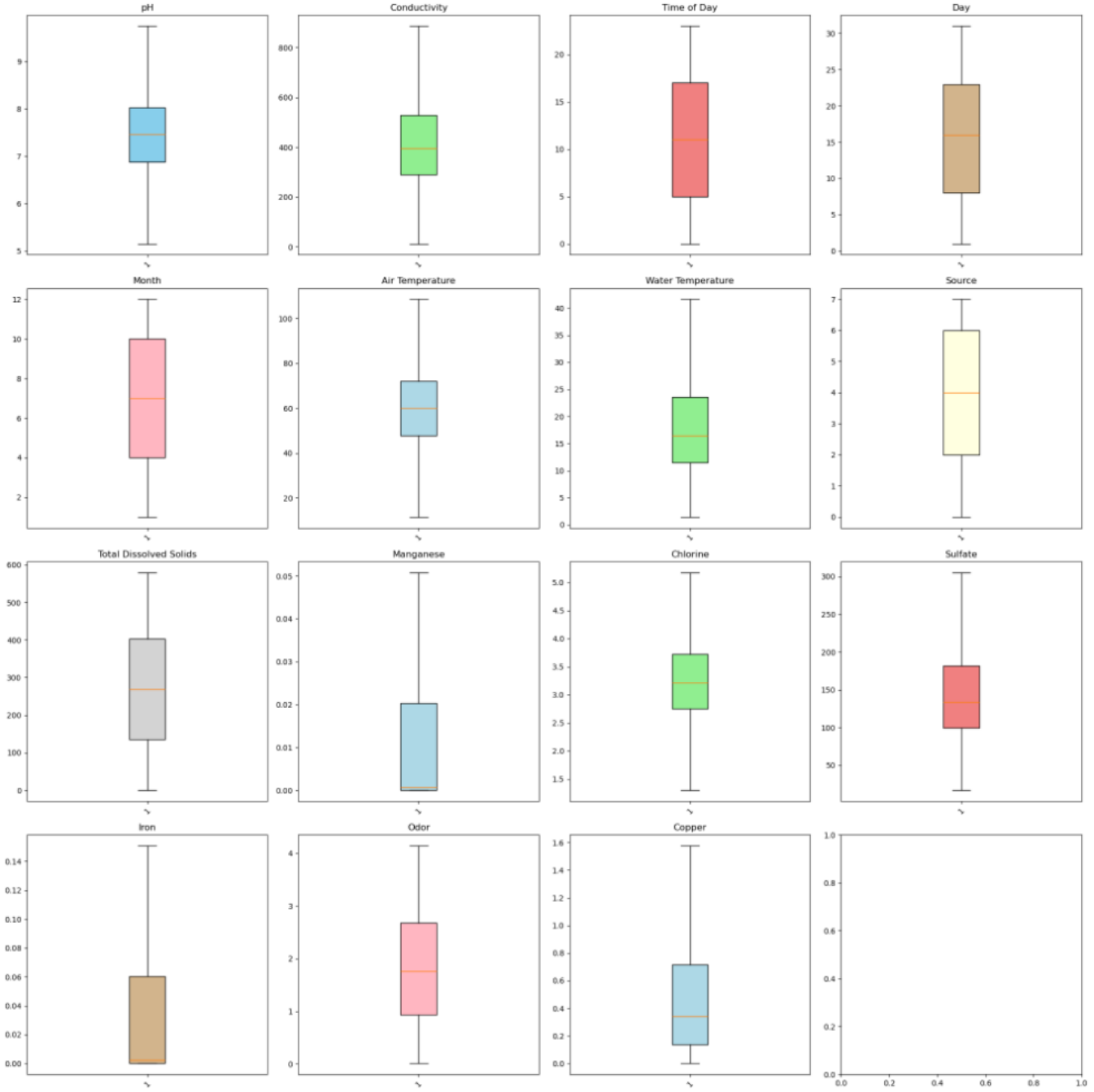


Outlier Detection and Analysis

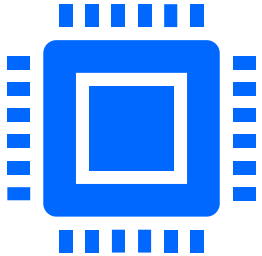
- Identified outliers using boxplot visualization.
- Calculated percentage of outliers for each column.
- "Lead" column exhibits the highest percentage of outliers.
- Further investigation warranted to understand implications and potential data quality issues.

Outlier Removal

- Excluded non-predictive columns.
- Used Interquartile Range (IQR) method.
- Clipped outliers within 1.5 times IQR.
- Visualized top features' outlier removal



Data Preprocessing for Model Training

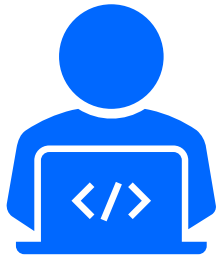


Feature Scaling:

Utilized StandardScaler from sklearn.preprocessing.

Applied feature scaling to training and testing sets.

Ensured uniform scale across features for improved model performance.



Data Splitting:

Used train_test_split function from sklearn.model_selection.

Split data into training and testing sets.

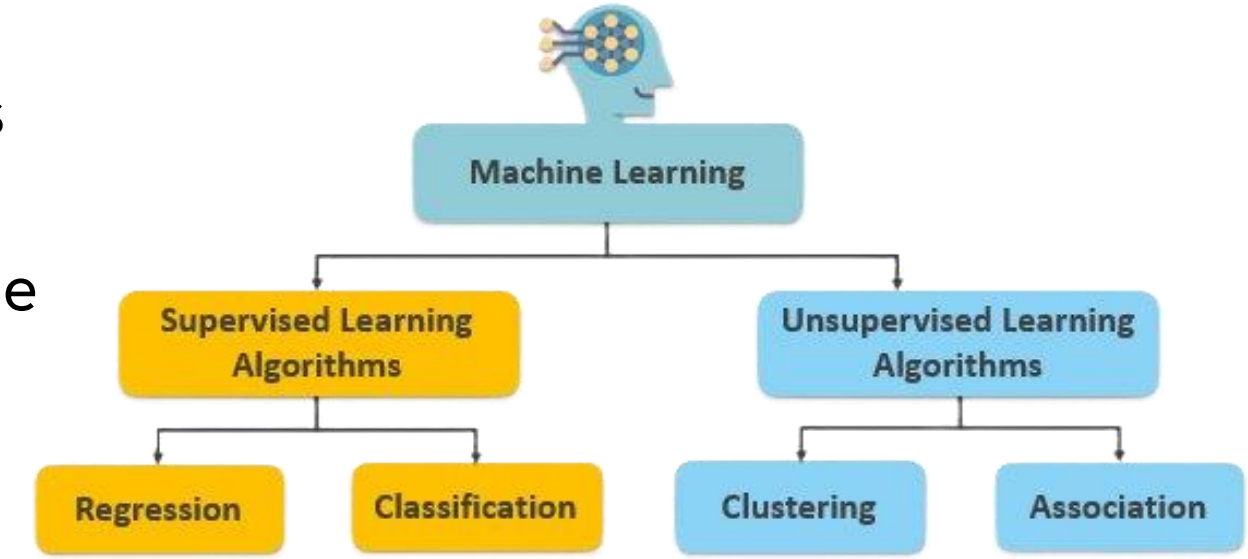
Allocated 70% of data for training and 30% for testing.

Shuffled data to ensure randomness.

Set random state for reproducibility.

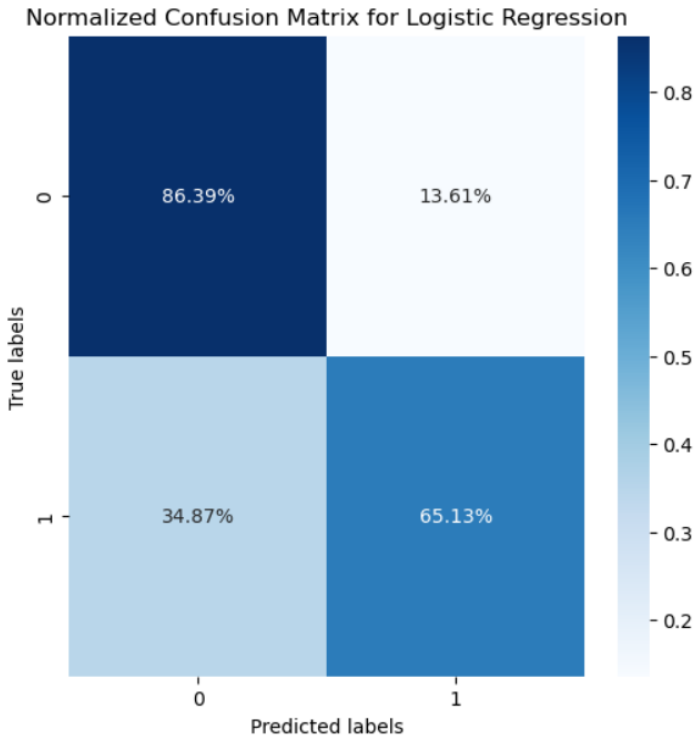
Algorithm selection

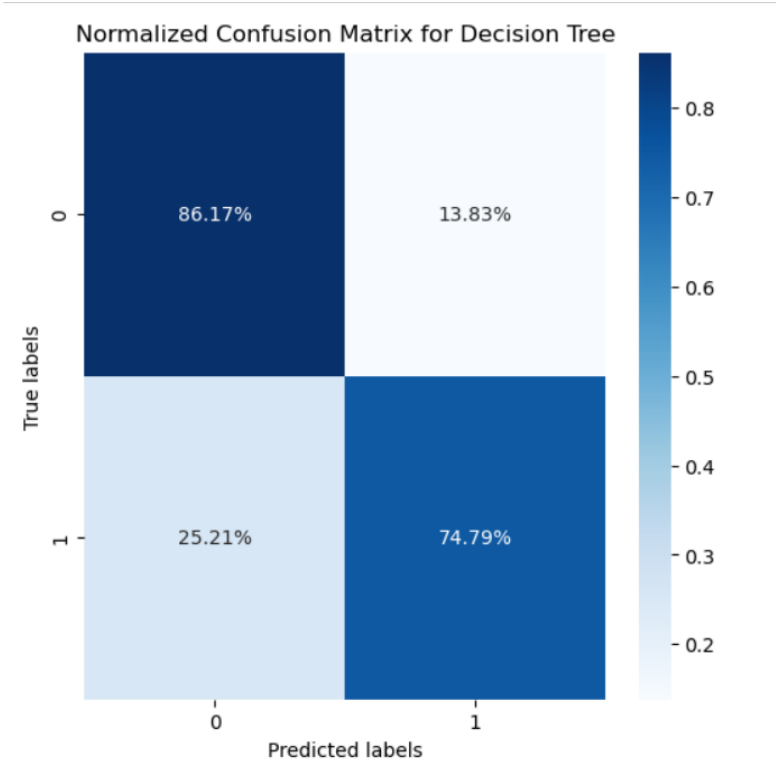
- Since we have class label available , problem belongs to supervised learning.
- Output of the target variable is either 0(not potable to drink) or 1(potable to drink):classification technique is used



Confusion Matrix

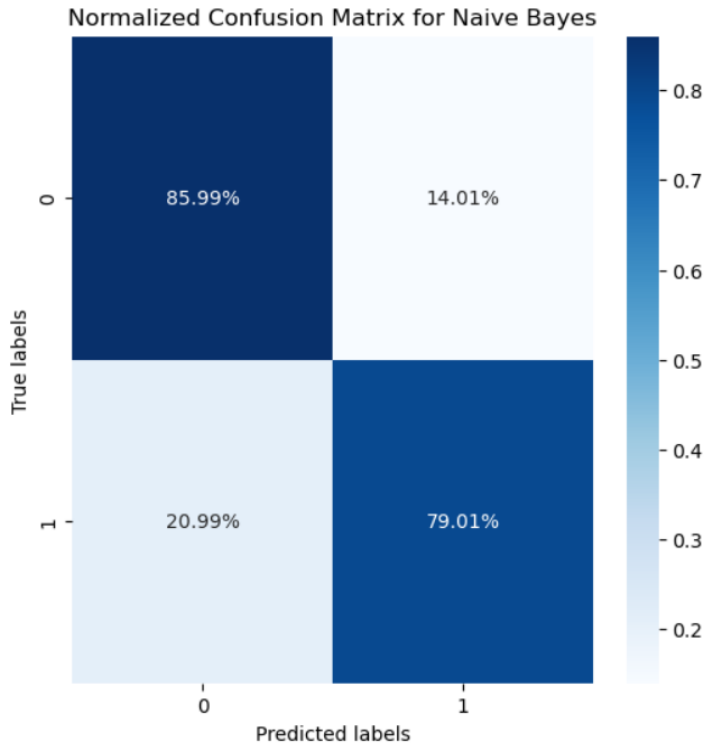
- After performing Logistic Regression, which exhibited significant misclassification, we switched to Decision Tree for further analysis. Here's what we observed:
- **True Negative: 86%**
- **True Positive : 74%**
- **Reduction in FN error from 34% to 25%**
- We also run other model to reduce the False Positive and False Nagative value.

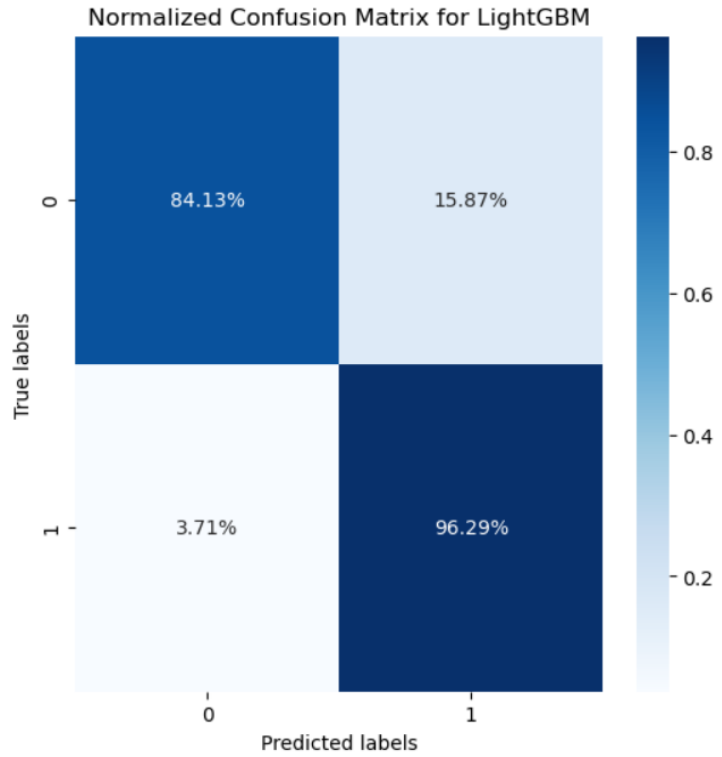




Confusion Matrix

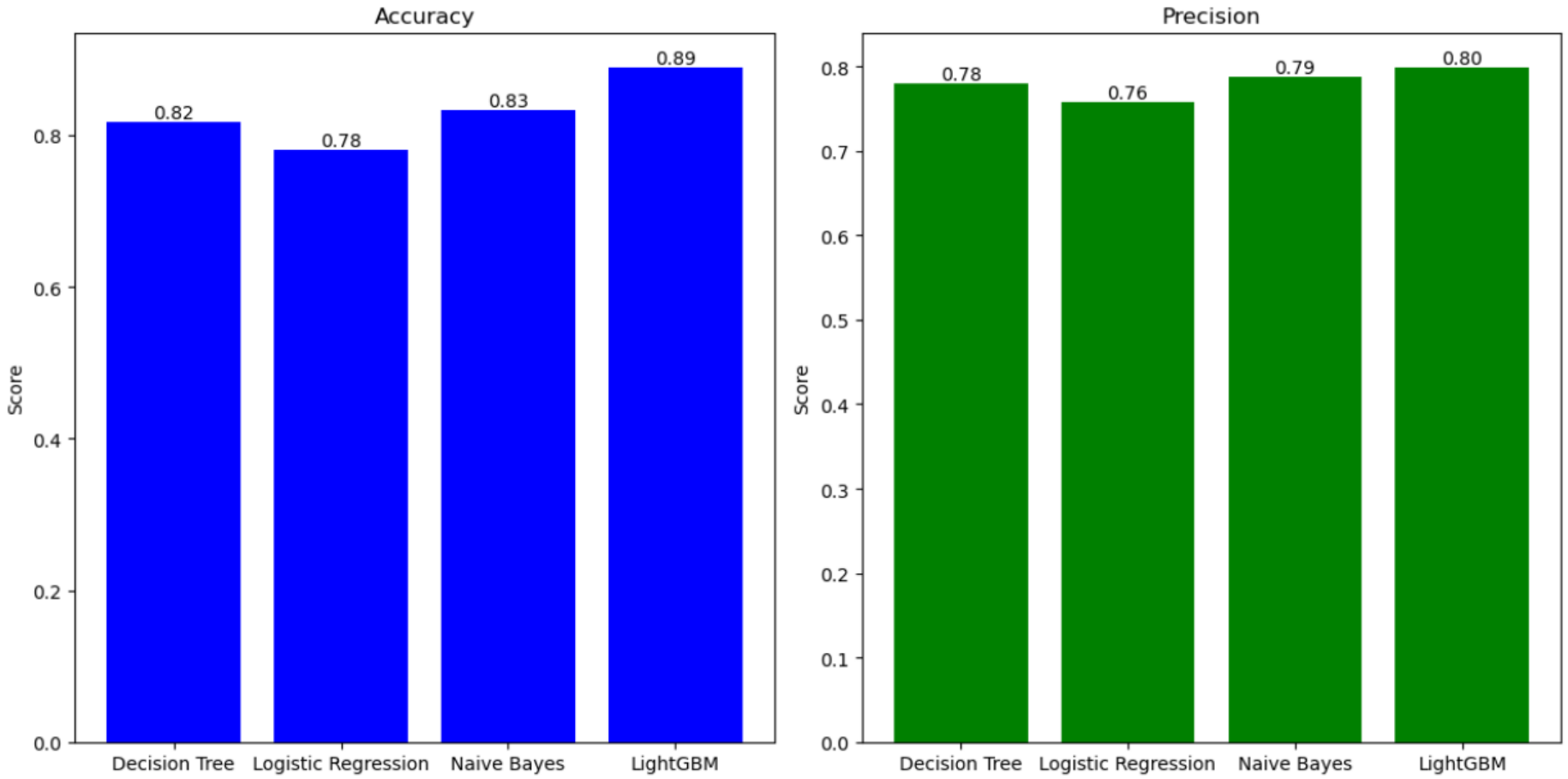
- Naive Bayes yielded a false negative error of 20% and false positive error of 14%.
- LightGBM demonstrated superior performance, with a reduced false negative of 3.71% .
- LightGBM's performance surpass Naive Bayes model, making it the preferred model due to its ability to significantly reduce false negatives and increase true positives..





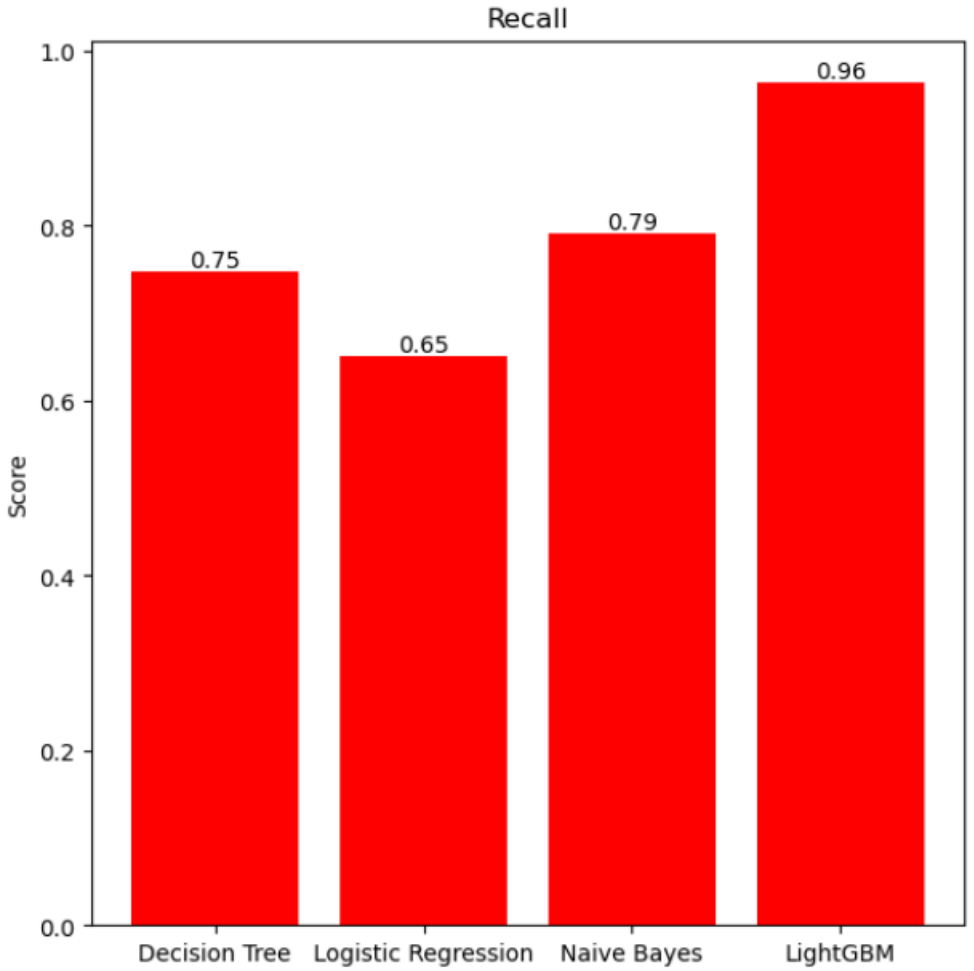
Model Comparison

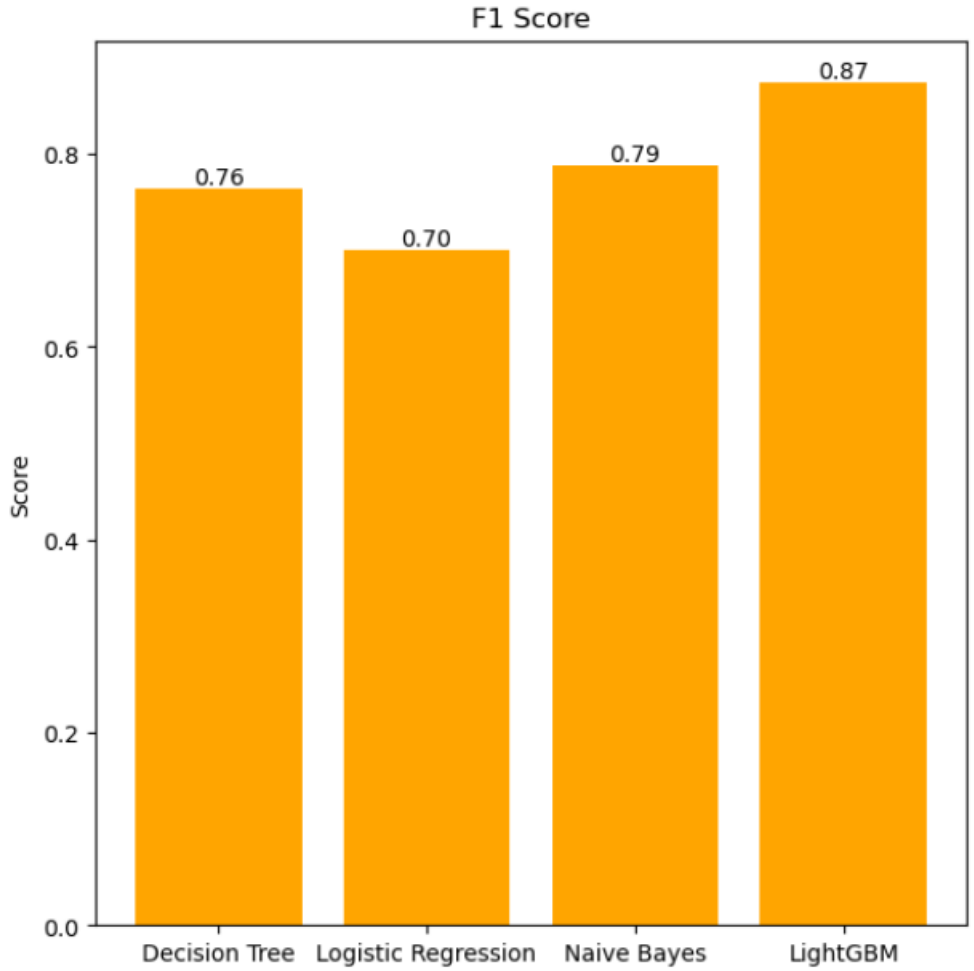
- We had done model training on Decision Tree, Logistic Regression, Naive Bayes, LightGBM



Model Comparison

- We found that LightGBM achieved the highest recall score and an impressive F1 score of 0.87, demonstrating its superior performance compared to other models.





Cross validation

- We used training data to generate multiple mini train-test split, to tune our model. Used cv=5

```
# Train and evaluate Naive Bayes with cross-validation
naive_bayes = GaussianNB()
nb_cv_scores = cross_val_score(naive_bayes, X_train, Y_train, cv=5)
# 5-fold cross-validation
naive_bayes.fit(X_train, Y_train)
nb_accuracy = accuracy_score(Y_test, naive_bayes.predict(X_test))
nb_precision = precision_score(Y_test, naive_bayes.predict(X_test))
nb_recall = recall_score(Y_test, naive_bayes.predict(X_test))
nb_f1 = f1_score(Y_test, naive_bayes.predict(X_test))
```

Model Selection

- Even after performing cross validation on our chosen algorithms, we got highest accuracy with lightGBM.
- LightGBM stands out with an accuracy of 0.89 and precision of 0.80, demonstrating superior performance compared to other models.
- Therefore, we have chosen lightGBM as our final model to predict water potability

Future Work

Temporal Analysis: If time series data is available, exploring changes in water quality over time could yield insights into trends and seasonal variations.

Geospatial analysis: If location data is available, we aim to perform geospatial analysis to identify water quality patters over different regions.


THANK YOU