ONLINE HACKATHON

FRESHWATER QUALITY PREDICTION

Team info

Pranav Dev Khindria RajatGoyal

Problem Statement:

"Freshwater is one of our most vital and scarce natural resources, making up just 3% of the earth's total water volume. It touches nearly every aspect of our daily lives, from drinking, swimming, and bathing to generating food, electricity, and the products we use every day. Access to a safe and sanitary water supply is essential not only to human life, but also to the survival of surrounding ecosystems that are experiencing the effects of droughts, pollution, and rising temperatures. Participants interested in this theme shall use the dataset mentioned below to check water suitability for consumption."

Dataset provided:

Kaggle

GitHub:

Project

Analysis Of Given Data:

The given data consisted of 5,956,842 rows and 23 columns, with there being 22 feature columns and one target column.

Feature_columns:

-pH
-Minerals(Chloride, Sulfate, Iron, Chlorine, Zinc, Fluoride, Nitrate, Lead, Manganese, Copper)
- Color
- Turbidity
- Odor
- Conductivity
- Total dissolved solids(TDS)
- Source
- Water and air temperatures
- Month, date, and time of the day the sample was collected

The data can be further divided into numerical and categorical columns as well

Categorical Features:

- **Colors**: Colorless, Near Colorless, Faint yellow, Light yellow, Yellow
- **Sources**: Lake, River, Ground, Spring, Stream, Aquifer, Reservoir, Well
- **Months**: All months of the year (Jan. to Dec.)
- **Days**: 1-31
- **Time of the day**: 0-23

The remaining are numerical columns

The data had a lot of NaN values per column, which had to be imputed/cleaned with the distribution of NaN values per column given as:

Percentage of NaN values in 'index': 0.0%

Percentage of NaN values in 'ph': 1.95%

Percentage of NaN values in 'iron': 0.67%

Percentage of NaN values in 'nitrate': 1.77%

Percentage of NaN values in 'chloride': 2.95%

Percentage of NaN values in 'lead': 0.45%

Percentage of NaN values in 'zinc': 2.62%

Percentage of NaN values in 'color': 0.1%

Percentage of NaN values in 'turbidity': 0.84%

Percentage of NaN values in 'fluoride': 3.18%

Percentage of NaN values in 'copper': 3.35%

Percentage of NaN values in 'odor': 3.0%

Percentage of NaN values in 'sulfate': 3.31%

Percentage of NaN values in 'conductivity': 2.75%

Percentage of NaN values in 'chlorine': 0.97%

Percentage of NaN values in 'manganese': 1.84%

Percentage of NaN values in 'total dissolved solids': 0.03%

Percentage of NaN values in 'source': 1.48%

Percentage of NaN values in 'water temperature': 2.82%

Percentage of NaN values in 'air temperature': 0.5%

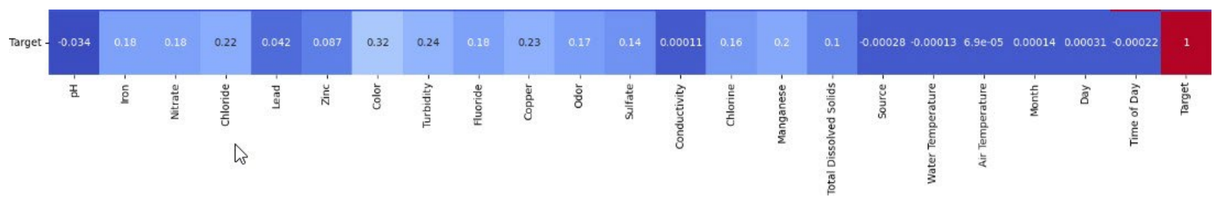
Percentage of NaN values in 'month': 1.61%

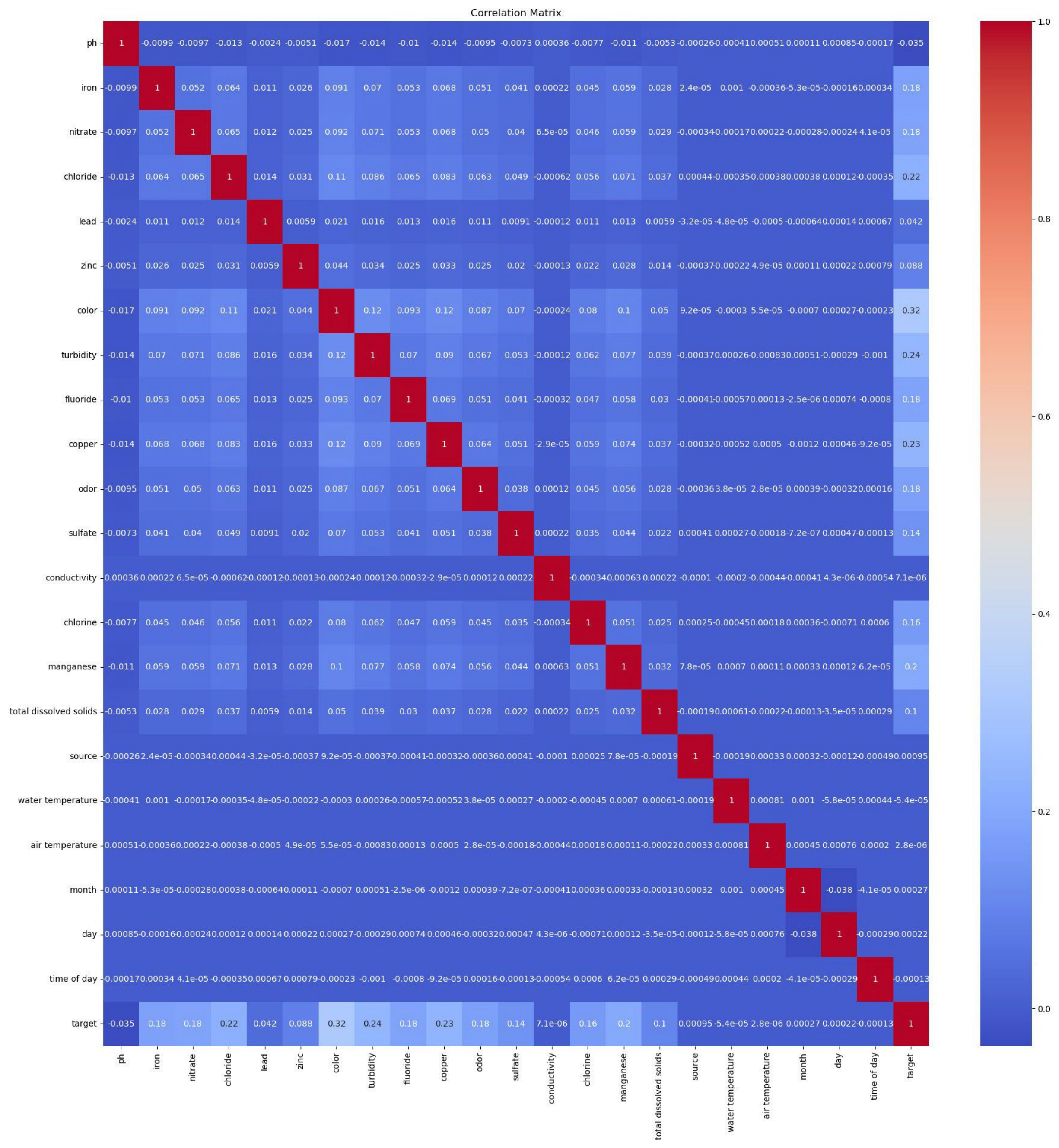
Percentage of NaN values in 'day': 1.67%

Percentage of NaN values in 'time of day': 1.92%

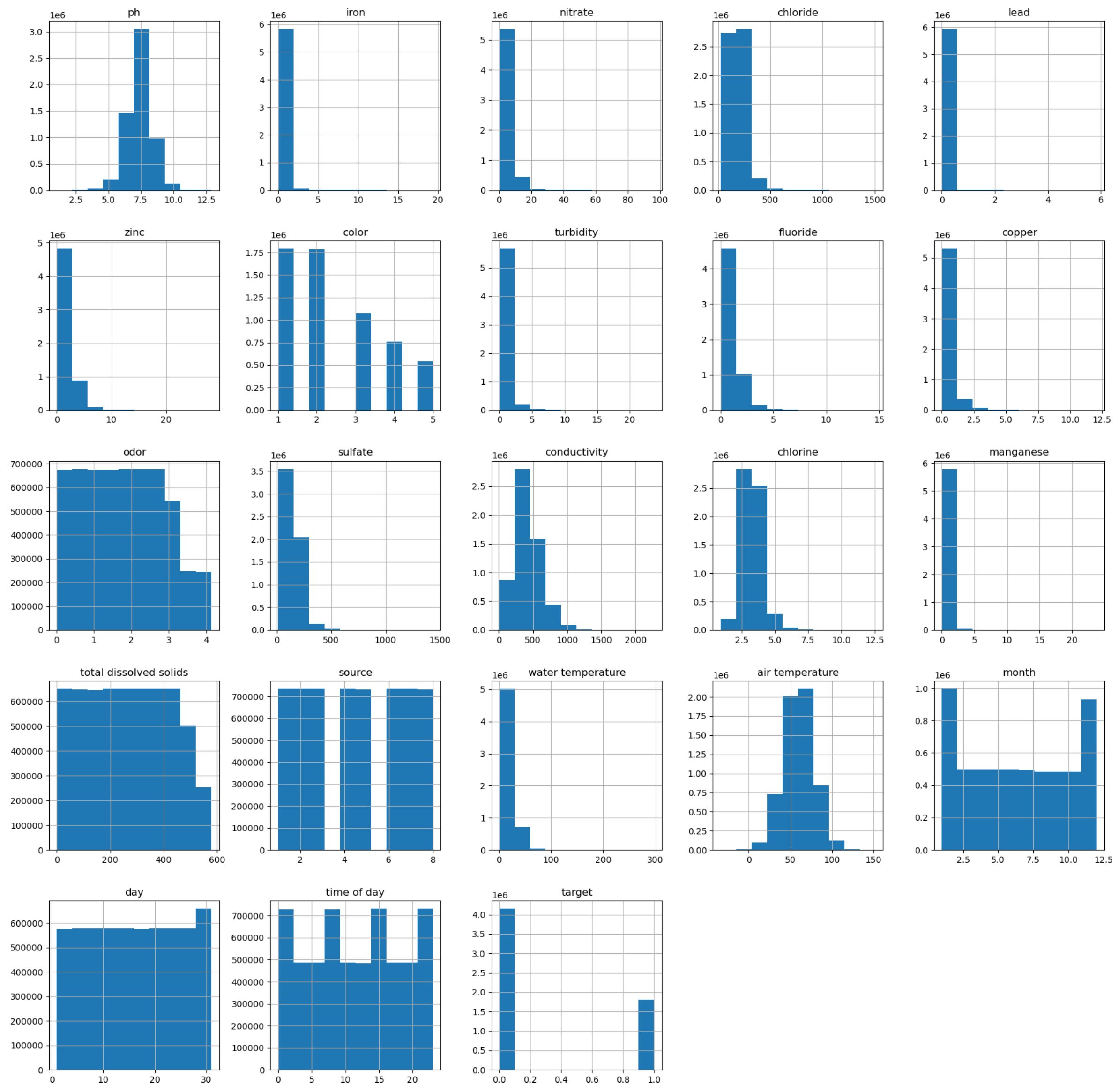
Percentage of NaN values in 'target': 0.0%

Dropping nan values would have resulted in removal of around **1.9 million rows** which would've caused a loss of data of approximately ***33%*** of the dataset. To prevent such a significant data loss, we decided to impute the values after evaluating the importance of each feature according to the correlation matrix.





But we couldn’t judge the importance of features solely on the basis of the correlation matrix and decided to analyze the distribution of each feature accordingly to make a proper judgment since we know that ph is an important factor when considering the quality of water. Still, the matrix didn’t show much correlation of ph with the target.



The pH value's limited impact on the target variable can be attributed to its alignment with standard guidelines, falling within the 6.5-8.5 range. Relying solely on the correlation matrix would have been misleading, emphasizing the need for a more comprehensive analysis.

Cleaning the Data:

Then, after some further analysis, we decided to drop the following columns in the **fresh_water_visual.ipynb**:

Source, time of day, month, air temperature, Month, day

As they didn't provide any significant value in predicting the quality of fresh water since sources were all fresh water sources and other were time statistics

We decided to use **Intel DevCloud** for further computation which was provided to us by the intel oneAPI Hackathon team.

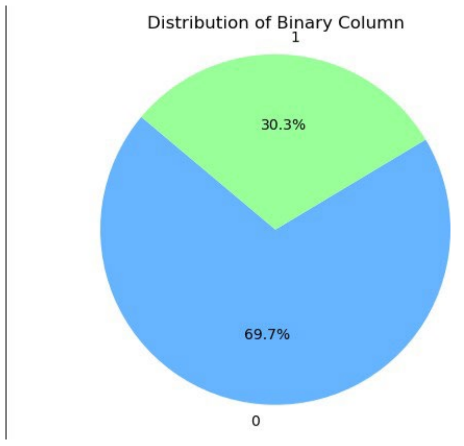
The values in remaining numerical columns were imputed using mean and median values accordingly to what seem fit as per the column's distribution as follows:

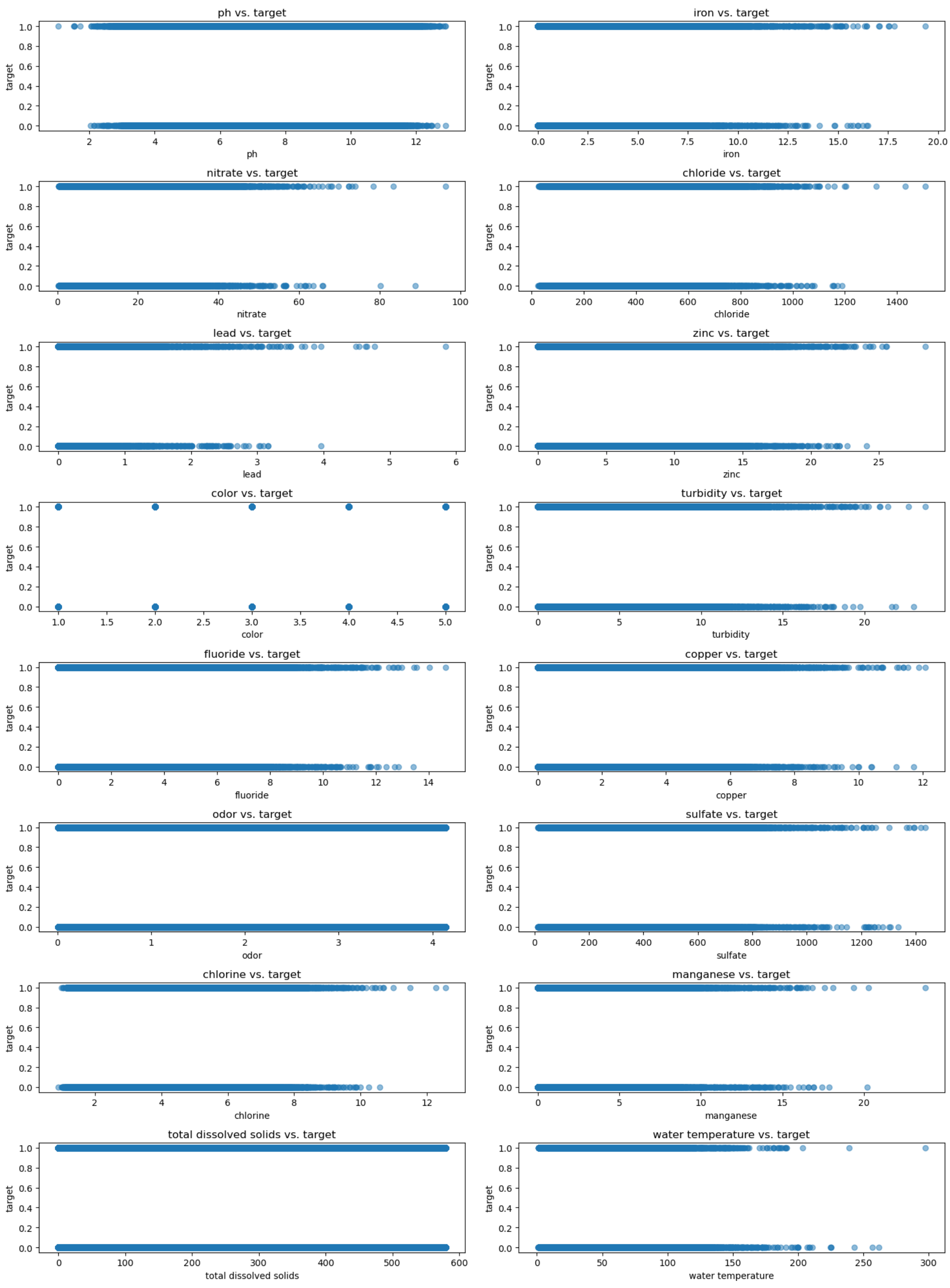
Column	Method
pH	Median
Iron	Mean
Nitrate	Median
Chloride	Median
Lead	Mean
Zinc	Median
Turbidity	Mean
Fluoride	Median
Copper	Median
Odor	Median
Sulfate	Median
Conductivity	Median
Chlorine	Median
Manganese	Mean
Total Dissolved Solids	Median
Water temperature	Median

After imputation of the above,the nan values of **'Color'** were dropped as it consisted of only 5739 rows with nan values which didn't provide any significant change.

We then plotted violin plots of each column with respect to target column to get a sense of distribution as target is a binary column with 0 and 1 with 1 being drinkable and 0 being unsafe to drink.

The violin plots of the following are in the **fresh_water_visual.ipynb**



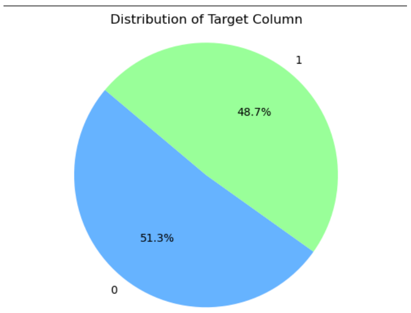


A scatter graph was also plotted to see how extreme the outliers were. With some features having very extreme outliers.

We then used adasyn (Adaptive Synthetic Sampling Approach for Imbalanced Learning) to balance out the dataset. After using adasyn, the rows increased to 8089744 to balance out the dataset and the ratio is as follows:

With all the analysis and preprocessing, the final dataset was ready which will be used to train models to predict the quality of fresh water

After applying adasyn, the dataset has been balanced and this is the distribution.



The Processed Dataset

Training The Model:

We aimed to have a model with as much precision as possible without overfitting, so we tried and tested different models. We compared them based on the resultant f1 score.

Given the dataset's size, we found the Intel® AI Analytics Toolkit (by one API) to be a valuable resource for streamlining our problem-solving approach. This toolkit, designed to enhance data science and analytics workflows on Intel® architectures, leverages OneAPI libraries for optimized computations. By seamlessly integrating its extensive package offerings into our Python codebase, we were able to enhance efficiency and reduce computational overhead significantly.

Key Intel AI Analytics Toolkit packages utilized in our project include:

Intel Extensions for Scikit-learn i.e. sklearnex and XGBoost

Intel Extension For Tensorflow i.e. itex

Given the complexity of the tabular data, we initially employed simpler machine learning models for swift results, before transitioning to more intricate machine learning and deep learning models. The dataset was partitioned into a 80:20 ratio for training, and testing respectively.
We also made a function for evaluation of different models

The following models were implemented for testing to find out the best model for our case:

- Logistic regression
- Random Forest Classifier
- XGBoost
- DNN

More about the functionality of these models can be found in **evaluating_different_models.ipynb**

Model	Time without oneApi	Time with oneApi	% improvement	Accuracy	f1 score
DNN	3047.19	2497.7	21.99983985	0.851	0.857
Logistic Regression	327.7	252.08	29.9984132	0.792	0.778
Random Forest Classifier	328.11	226.28	45.00176772	0.914	0.917
XGBoost	239.83	175.06	36.99874329	0.897	0.899

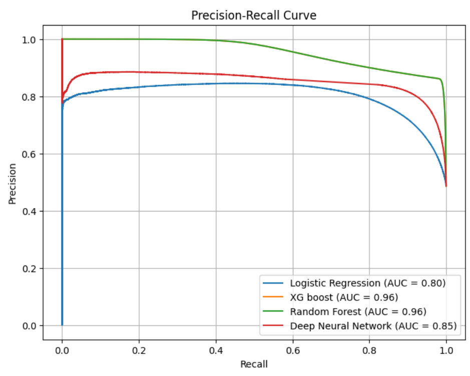
The least time taken to train was by **XGBoost,** but it had an accuracy of only 0.897 and f1 score of 0.899

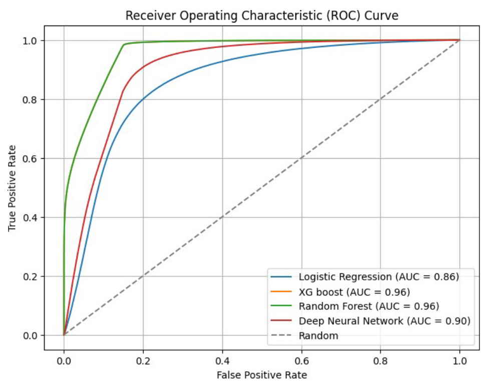
The most accurate was the **Random Forest Classifier,** with an accuracy of 0.914 and f1 score of 0.917, but the model is too big to run on low-tier hardware.

Why we used f1 score and AUC as metric to judge model performance.

We chose the F1 score and AUC-ROC score as evaluation metrics for our binary classification model. The F1 score balances precision and recall, crucial for our binary target. AUC-ROC assesses the model's ability to distinguish between classes. Intel OneAPI facilitated efficient metric calculation. These metrics are vital in gauging model performance, especially in imbalanced datasets or with varying error costs.

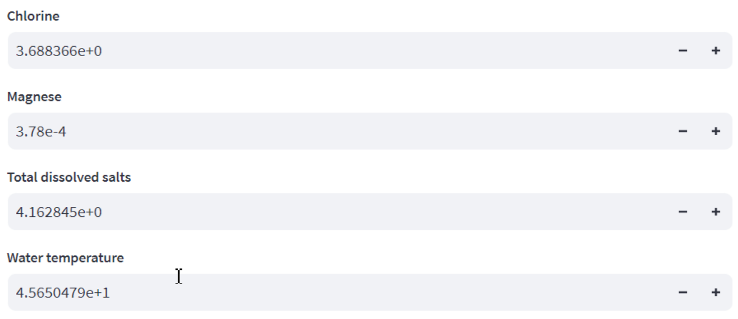
These are the comparison curves after model evaluation.





Challenges and Solutions:

We encountered a challenge in implementing a model capable of imputing missing values in a dataset based on partial input from users. Unfortunately, the process proved to be computationally intensive, leading to delays due to limited resources. However, we're actively working towards overcoming this constraint, as we're committed to providing a seamless service for this purpose.
At present, our program allows users to input various parameters, and based on this information, our model assesses whether the water is potable or not. This prediction is made with a high degree of accuracy, thanks to the Deep Neural Network (DNN) model we've employed.
In our development process, we opted for the DNN model over the Random Forest Classifier. This decision was influenced by the size of the Random Forest model, which posed challenges when attempting to integrate it with third-party software like Streamlit due to its substantial memory footprint (approximately 4.55GB).
Looking ahead, we're dedicated to refining and expanding our capabilities to offer even more robust and efficient solutions for water quality assessment.


Chlorine
3.688366e+0

Magnese
3.78e-4

Total dissolved salts
4.162845e+0

Water temperature
4.5650479e+1

We would also like to work on a research paper related to this and explore this domain.