Prediction of water potability : A Comparison of Decision Tree and Random Forest

Description and motivation of the problem

Water is essential in all living things. Everyone may neglect it, but drinking clean water is a necessity.It is important that everyone have right access to clean water without contamination. Polluted water causes serious waterborne illnesses and poses a threat to human health[1]. Every year almost 3,575,000 people are died due to water-related diseases[2]. Machine learning methods have been applied to predict the quality of water and answer all these problems

• Solve the binary classification problem task of predicting potable water quality based on a dataset from the Kaggle website using two classification models which are Decision Tree (DT) and Random Forest (RF).
• Analyse and compare the performance of both models from predicting the potability of water (water is safe for drinking or not) from the water quality dataset.
• comparing the results with a previous study obtained by Osim Kumar Pal[2] for random forest using a similar dataset and using confusion metrics to assess model performance.

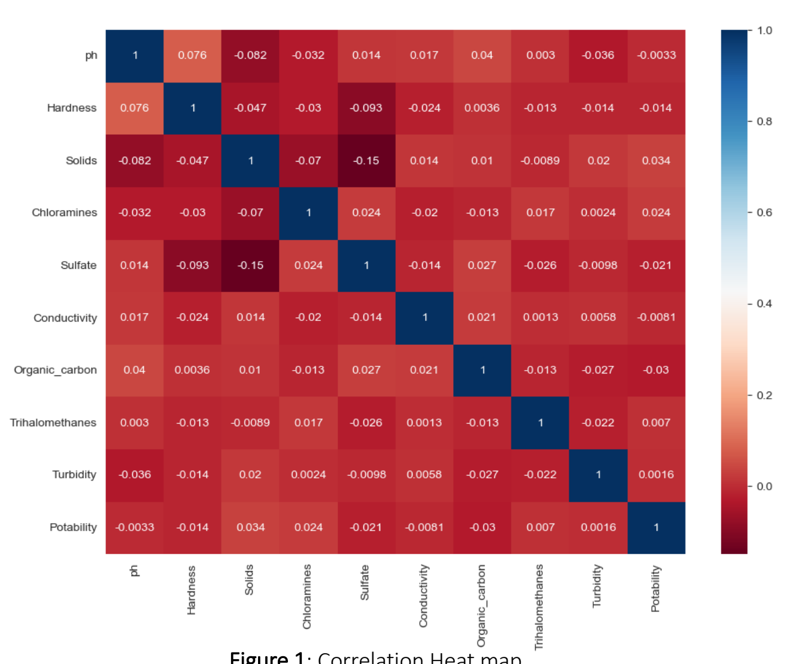
Figure 1: Correlation Heat map

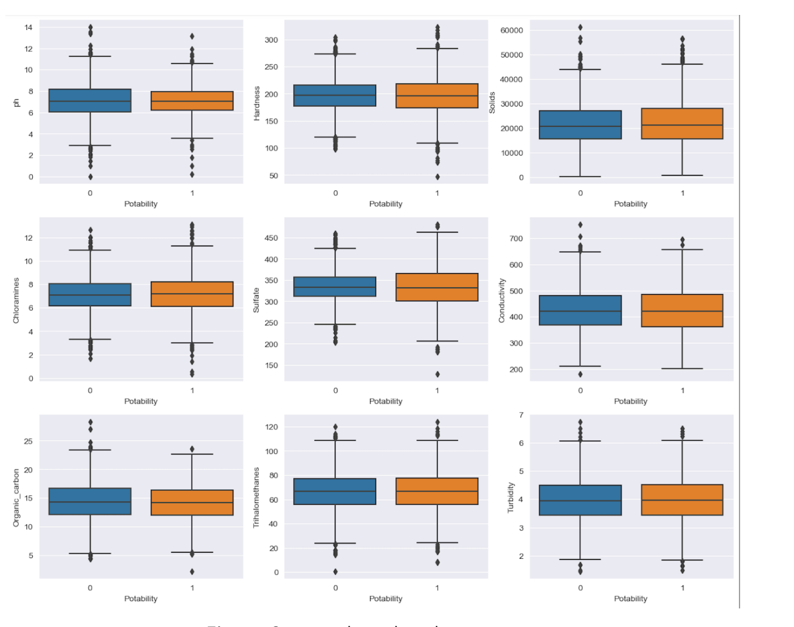
Figure 2: Box plots distribution

Categories	Max	Min	Mean	Std
pH	14.00	0.00	7.08	1.47
Hardness	323.12	47.43	196.37	32.88
Solids	61227.19	320.94	22014.09	8768.57
Choramines	13.13	0.35	7.12	1.58
Sulfate	481.03	129.00	333.78	36.14
Conductiity	753.34	181.48	426.21	80.82
Organic Carbon	28.30	2.20	14.28	3.30
Trihalomethanes	124.00	0.73	66.39	15.77
Turbidity	6.74	1.45	3.97	0.78
Potability	1.00	0.00	0.39	0.48

Table 1: All categories

Parameters	Unit	Standards
pH		6.5-8.5
Hardness	mg/L	300
Solids (TDS)	ppm	<20000
Chloramines	mg/L	<4
Sulfate	mg/L	<250
Conductivity	µS/cm	<400
Organic Carbon	ppm	<25
Trihalomethanes	µg/liter	<37
Turbidity	NTU	<5

Table 2: Drinkable water quality standards

Dataset Description & Exploratory Analysis

• This project used a water quality dataset from the Kaggle website, containing details about components and minerals in water such as pH, hardness, solids, chloramines, sulfate, conductivity, organic carbon, trihalomethanes, turbidity and potability.
• The original dataset consists of 3,276 rows and 10 columns. There are 10 features in this dataset as mentioned but, in this coursework, we will use potability as a target class for our prediction and 9 predictors left used for training our models.
• The target class contains binary numbers 1 and 0 which is Indicates if water is safe for human consumption where 1 means potable (safe for drinking) and 0 means not potable (unsafe for drinking)
• There are some missing values in some features, in this research, we also tried various methods to deal with missing values whether by filling with mean, filling with 0 or removing the row that contains a missing value. We have found that filling with mean values is a method that may not produce the best results but is the most reasonable. The reason we will explain in the section on experiments.
• Table 1 represents summary statistics about max, min, mean and standard deviation of all features in the dataset compare with potable water quality standards from the world health organization in table 2. The dataset tended to be slightly higher than the standard values given because this information contains large amounts of non-drinkable water.
• The correlation heatmap from figure1 illustrates the correlation either with labels or with inner variables. There shows a weakly correlation between some features such as solids and chloramines affecting the potability of water slightly.
• The box plots distribution figure 2 shows the distribution of each feature and outliers. The distribution shows that the pH of drinkable water should be between 7.0 and lower variance than non-drinkable water. However, most of the boxplots show very similar distribution. This makes it impossible to definitively determine what are the indicators that make water not drinkable.
• We tried to use PCA which is unsupervised machine learning as a part of pre-processing before training the supervised model to find the important feature and reduce the dimension for feature selection. However, according to the correlation heatmap, the relationship between each feature is quite weak leading to poor performance of PCA to reduce features[10].
• Figure 4 show the distribution of all features in this dataset most of the feature follows the Gaussian distribution. However,to enchanch the performance and accuracy of our model we need to normalise this dataset because each variable still has a different scale. The result shall discuss in the next section.
• The pie chart in figure 3 provides information about potable water by the percentage which the majority which accounted for 60.99% is not safe for drinking only 39.01% is drinkable water.
•As a result, of previous observations. The distribution of the target class in the dataset is an imbalance (0 non-drinkable: 1998, 1 drinkable: 1278). There is a higher proportion of non-drinkable water than drinkable water as we can see from the pie chart in figure 3. Imbalance target class may affect the result in terms of accuracy when using this data training model classifier[9].

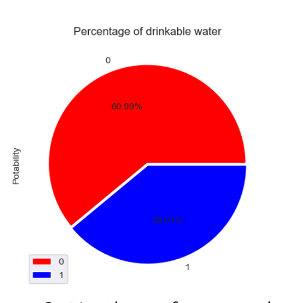

Figure 3: Pie Chart of target value

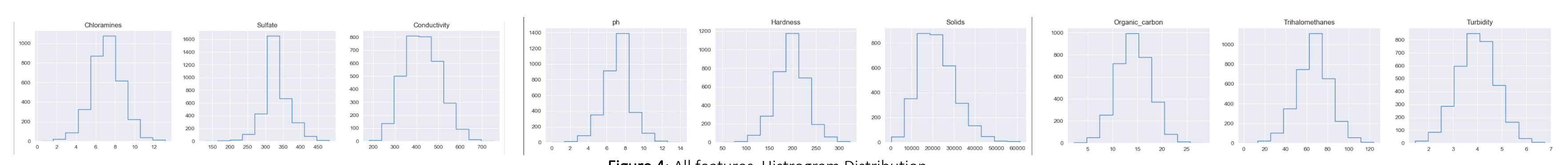

Figure 4: All features Histrogram Distribution

The summary of implementing model with their advantages and disadvantages

Decision Tree (DT)

• A decision tree is a non-parametric supervised learning algorithm. Using a graphical representation of all the possible solutions to a decision based on certain conditions.
• It is widely used for classification and regression problems and become a standout of the most well-known machine learning methods for classification problems[13]. Numerous fields apply this method to solve their problem and propose in many ways.
• Decision trees also form the basis for Random Forests which are probabilistic and state-of-the-art in computer vision.
• Decision tree consists of a root node, branches, internal nodes, and leaf nodes. Starting with a root node which is the first split decision from the data features and each branch of the tree represents a possible decision that must be filled to move on to the next branch until it ends in a leaf where there are no other branches[3].

Advantages
✓ Flexibility of model can use in both classification and regression task
✓ The concept of the model is simple to interpret, understand, and visualize.
✓ Little effort is required for data preparation. Because decision trees are resilient to outliers and missing values, they require less cleaning of data than other algorithms
✓ Can handle both numerical and categorical data. Can also handle multi-output problems but the attribute generated require to be categorical.
✓Require a lower time for training model compared with other methods

Disadvantages
! Usually outperformed by many other machine learning approaches such as naïve bayes, logistic regression including random forest.
! Decision tree does not work well with continuous numerical variables.
! High Variance, model is going to change quickly when chaging in training data.
! Can create over the complex tree that does not perform well with different datasets leading to overfitting of the data which can give a poor result when apply to the full dataset but limiting the tree depth can avoid this problem

Random Forest (RF)

•First introduced by Leo Breiman in 2001[5]. It is a powerful and famous supervised learning method used for classification and regression tasks developed from the decision tree, most used and popular among many researchers and students.
• Random forest based on multiple decision trees It constructs classification trees from several samples and uses their majority vote for classification and average for regression[2].
•Single decision tree sometimes tends to learn the training data too well, resulting in poor prediction performance on unseen data. Random forests can correct decision trees' habit of overfitting to their training set.
• Random forest uses Bagging to divide the dataset for training to a different tree. In MATLAB, there are useful functions to generate Random Forests such as treebagger and fitcensemble. It provides stable predictions, and the accuracy is better than performing a single decision tree.

Advantages
✓ Run efficiently and perform well with highly accurate predictions for large data without pre-processing and with little tuning.
✓ Reducing the possibility of overfitting by building multiple trees.
✓ Few hyperparameters for tunning
✓ Similarly to decision trees, random Forests can handle linear and non-linear relationships well.
✓ Better accuracy than other classification algorithms
✓ Provide information about which features are important for each datasets

Disadvantages
! Random Forests are not easily interpretable.
! Require a long time for optimal parameter and training model
! Random forest is like a black box algorithm due to few parameters make we have very little control over what the model does.
! A large number of trees can make the model too slow and ineffective for real time prediction
! If the proportion of relevant features is small. Random forests are likely to perform poorly because there is a lower chance that relevant features are used to produce trees.

Hypothesis statement

• Random forest aggregates the simplicity of decision trees[5]. Trying to reduce variance by training on a random different set of data. Therefore, RF typically performs better than DT across the same dataset.
• It is expected that both models will perform significantly better than a random guess when predicting water potability.
• Random forest takes a longer time for training the model and optimising hyperparameters compared with the decision tree but both models will perform well and operates easily on a large dataset.
• We compare our result with Mftnakrsu[12] who found that RF consistently outperforms DT in both with and without optimise hyperparameter. We also compare the results with a previous study obtained by Osim Kumar Pal[2] for random forests using a similar dataset.

Methodology

1. Handling missing value by imputing mean value into missing data, the reason will explain in the analysis section. Data pre-processing and feature selection, interpretation dataset before comparing model.
2. Split the dataset into a train set and test set (80:20). The test data remains unseen to models until the end. For the training set due to the features being different scales, we apply normalisation to change them into the same scale before train model.
3 Train both models without optimising hyperparameters and evaluate by a confusion matrix. Calculate recall, precision, F1 score and accuracy[6]. This will provide information about how good or bad the performance model is and compare after optimising hyperparameters.
4. Hyperparameters were chosen using brute force and optimize function for the decision tree and Bayesians optimization and optimize function for random forest. Finding the best hyperparameters that give the lowest loss on the training dataset.
5. Predict the unseen test set mentioned in step 2. The error is measured by the confusion matrix along with other scores such as recall, precision, F1 score and accuracy. Interpreting the result and making a conclusion for the analysis & evaluation section.

Experimental results, parameter choices and feature selection:

Decision Tree

• After splitting the dataset into a train set and test set, we normalise the data into the same scale before fitting the decision tree model using fitctree function in MATLAB that referance from the *standard CART* algorithm[11] to create decision trees which can return a fitted binary classification decision tree based on our training set features.
• fitctree has three hyperparameters to adjust to control the depth of the tree[4] that is MaxNumSplits, MinLeafSize and MinParentSize. Reducing the complexity and depth of the tree can prevent overfitting.
•According to figure 5 adjusting the minimum sample leaf size can increase the performance of our model. A smaller leaf size possibly makes the model more prone to capturing noise in train data. With a larger leaf size, the tree will stop growing after a few splits leading to the inaccuracy of the model. Therefore, in order to find the appropriate range, we generate a graph setting of values from 10 through 100 that represent the minimum number of observations per leaf node.
• Another hyperparameter that we are going to optimise is the maximum number of decision split in each tree. The lower value means the less complex decision tree.
• For the decision tree we used two methods for optimising hyperparameters the first one is manual *brute force* search. Using for-loop to go through all possible hyperparameters until the best parameter is found. Second is using optimizeHyperparameters function to minimise the CV loss for the decision tree by varying the hyperparameters 30 times over different values of hyperparameters[4].
• The optimization function figure 6, 7 searched over the minimum number of leaf nodes and the maximum number of decision split, The output is the decision tree classifier with the minimum estimated cross-validation loss.

The final choice of parameters

•The performance matrix in figure 8 is a measure of the model's accuracy in making predictions on the training dataset. Both methods that we used to optimize hyperparameters increased the performance of the model around by 9%.
Brute Force: MinLeafSize: 18, MaxNumSplits: 33, train accuracy = 0.675
Optimise function: MinLeafSize: 22, MaxNumSplits: 18, train accuracy = 0.6768
• To conclude, choosing the best hyperparameter: minimum sample leaf size is 22 observations per leaf and the maximum number of decision split is 22 times.

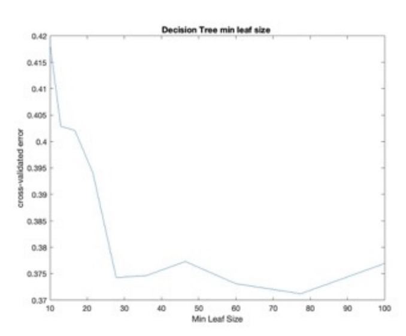

Figure 5: MinLeafSize observation

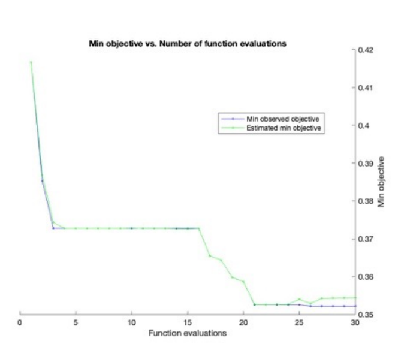

Figure 6: Optimise Hyperparameters

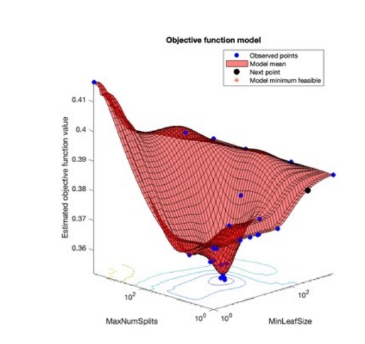

Figure 7: Colleration between hyperparameters

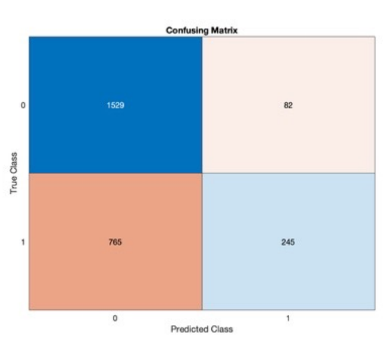

Figure 8: Decision Tree Confusion Matrix

Random Forest

• After dividing the dataset into a train set and test set and normalising the data into the same scale, we fit the model by a bagging or bootstrapping algorithm with random predictor selections at each split[5] to divide the data for training among the different trees.
•MATLAB provide a built-in function called treebagger and fitcensemble to create a Random forest which can return an ensemble of bagged decision trees based on our training dataset.
• Hyperparameters that can control depth in Random forests consist of the number of trees in the forest, the maximum depth of the trees, and a minimum number of samples per leaf node similar to the decision tree.
•From Figures 9 and 10 increasing the number of trees in a forest tree affects the accuracy of the model using cross-validating giving comparable estimates to those of the independent set.
•For the Random Forest we also used two methods for optimising hyperparameters the first one is Bayesian Optimization for hyperparameter tuning. The second is using the optimizeHyperparameters function to minimise the CV loss for the decision tree by varying the hyperparameters 30 times over different values of hyperparameters.
• Using automatic hyperparameter optimization (figure) find hyperparameters that minimize five-fold CV loss in which the number of objective function evaluations has been set at 30. Then select the hyperparameter optimization options to optimize two parameters which are the minimum number of observations per leaf node and the number of trees in the forest.

The final choice of parameters

Bayesian optimization by treebag method produced a better result in terms of accuracy. However, treebag function does not have cross-Val to prevent overfitted and when looking at the training results of the training set, the prediction model was too accurate and conflicts with the results from the test data. Therefore, we choose a fitcensemble method that gives us more control over the implementation and validates the result which we can ensure that it is not prone to overfitting.
• In the end, we choose a minimum sample leaf size is 2 observations per leaf and the number of trees in the forest is 472 trees as a final hyperparameter.

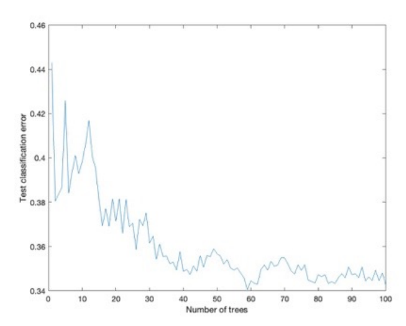

Figure 9: Number of tree in forest

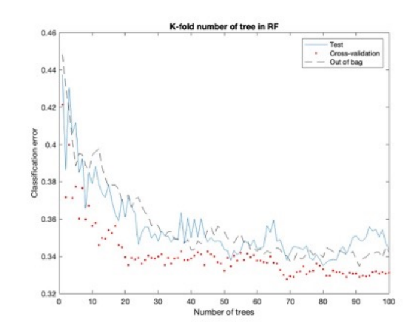

Figure 10:CV-out of bag number of tree

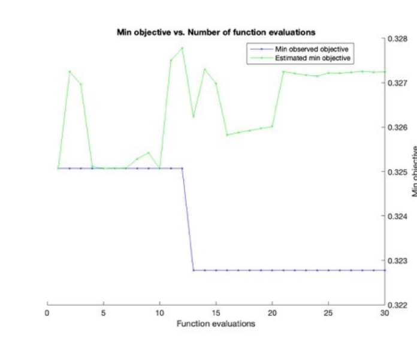

Figure 11: RF optimise Hyperparameter

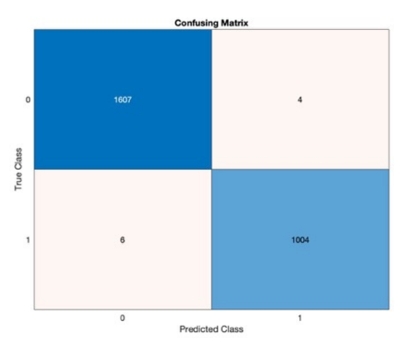

Figure 12: Random Forest Confusion Matrix

Future Work:

• While working with the unbalanced dataset, consider using methods like SMOTE or ADASYN to rebalance the classes. Overcome unbalanced dataset to improve the performance of the model.
• Using other classification approaches such as Naïve Bayes or Logistic Regression etc. and trying to add more processes for data wranglings such as transformation or feature selection instead of just only filling the mean value.
• Increase the data set scale. Improve the model performance by gathering more data, especially on the potable water (1) samples to handle the cost of misclassification.
• Using other methods to optimise and find the best hypermeter such as grid search.

Analysis and Evaluation of results:

• There are several methods for handling missing values. Water quality is very sensitive data[8] that we should not impute 0 value in each row. However, removing missing rows results in even fewer data available for the training model as we know DT & RF perform significantly improve by training with a large sample. Therefore, imputing by mean values will be suitable for this project that requires comparing the full performance between two models.
• From the previous observation, the correlation heatmap shows there is a weak correlation between all the features leading to poor performance when performing PCA for feature selection. That is why we do not have to remove any features from the dataset.
• While working with different scales of features in the dataset, machine learning models are likely to prioritize larger-scale variables to handle this we apply normalization to the dataset. Converting all features with different scales to the default scale[3]before training model.
• DT achieve a significant increase of 9% in both tuning methods after tuning the hyperparameters. However, RF improvement by 4% after adjusting the hyperparameters shows that the RF model still performs well with little tunning. The automatic hyperparameter optimization function in MATLAB was used along to search and compare the performance of different hyperparameters to find the most suitable hyperparameters that minimize loss. After select the best hyperparameters, we calculate the confusion matrix to evaluate the performance in each model on training set the result can see in figure 8 and 12.
•Our experiment shows that the RF outperform DT in both the training set and test set to forecast the quality of water which is supported our hypothesis statement. The result can see in table 3 the best RF and DT models after performing hyperparameter tuning are slightly improved and if we compare these two models on the test set, we can see that RF perform slightly better than DT by 4% of accuracy which is consistent with the previous result produce by Mftnakrsu[12].
• Although the RF model achieves superior accuracy, it requires an amount of processing time (33.27s) for training the model and tuning hyperparameters if the model has a large number of trees in the forest compared with DT (0.06s) which is much more effective and faster to deal with the real-time prediction.
• Relying on only accuracy score as an output metric may not be sufficient for making a conclusion. We also consider looking at other scoring metrics[6] such as precision, recall and F1 score illustrate in table 3. In terms of precision and recall, RF also produces better scores than DT by 0.07 and 0.05 respectively when performing on the test dataset.
• From the ROC graph in figure 15, the test AUC of RF is marginally higher than DT (even without tunning hyperparameter RF still have a higher performance in both train and test sample). As a baseline, the classifier model that creates a curve line closer to the top left corner is likely to have a higher performance. In contrast, if the curve is near the 45-degree diagonal of the ROC space that means our model has less accuracy and is not different from random guess. Due to measuring the same subject which is the test set sample, it can claim that the random forest has a better performance compared to the decision tree.
• From the confusion matrixes in the figure 16 and 17, it can be seen that the RF handles the cost of misclassification better than the DT, especially in answering the proportion of drinkable water (1).
•Most of the correct classification is the undrinkable water (0) labels in both models. This is probably due to there being a number of learning labels which result in having more confidence to identify that class. Nevertheless, the accuracy of the model when predicting drinkable water (1) is still low in both models due to an unbalanced dataset, making most machine learning models ignore the minority class leading to poor performance results.
• Finally, the table of water quality modelling results below shows the F1-score which is a performance metric for classification and is calculated from the summary of precision and recall. We can see that the RF f1-score is much higher than the DT by 5%. Hence, this can confirm our previous assumption that random forests have higher performance than a decision tree.

Water quality modeling result		
Decision tree	Model	Random Forest
0.6768	Train set accuracy	0.6913
0.0689	evaluation time	33.2734
0.6122	Test set accuracy	0.6534
0.5977	precision	0.6608
0.5479	recall	0.5989
0.5717	F1 score	0.6283

Table 3: Summaries model result

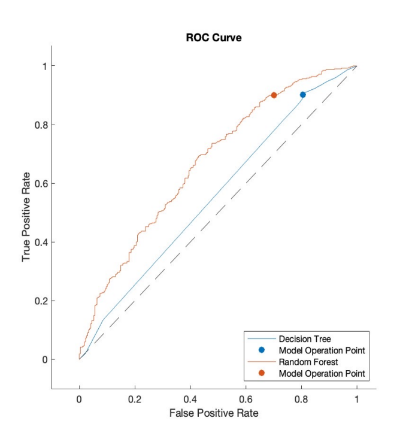

Figure 15: ROC curve DT vs RF

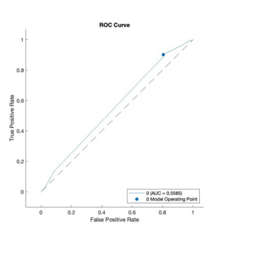

Figure 13: DT ROC curve

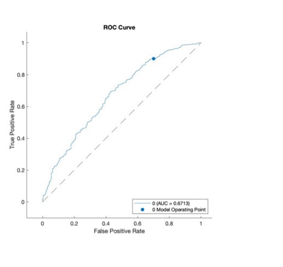

Figure 14: RF ROC curve

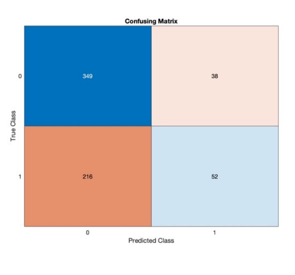

Figure 16: DT confusion matrix test

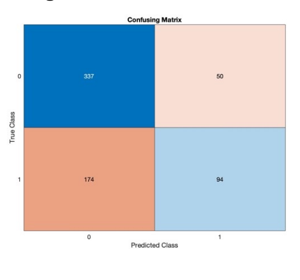

Figure 17: RF confusion matrix test

Lessons Learned

• Random forest is a combination of multiple decision trees which can increase the accuracy of the model without issues of overfitting. Even if it comes with a longer processing time than the decision tree.
• RF perform well without much tuning in exchange for longer computational time. As well DT increases in the number of samples leading to complex trees and overfitting. Hence, it is important to control tree depth in both models.
• Target class imbalance (0,1) leads to inefficient model performance due to RF & DT model tending to ignore minority class, which can make the model inaccuracy.
• Tuning parameters can help reach optimal model performance However, we should beware when tuning hyperparameters because there are a lot of parameters and methods to implement, leading to overfitting and working well only with training data.

Reference

[1] Deaths from Dirty Water. (no date) *The world counts*. Available at: https://www.theworldcounts.com/challenges/planet-earth/freshwater/deaths-from-dirty-water/story (Accessed: November 26, 2022).
[2] Pal, O. K. (2022). *'The Quality of Drinkable Water using Machine Learning Techniques'*. Department of Electrical & Electronics Engineering, American International University-Bangladesh, Bangladesh y, 9, doi: https://dx.doi.org/10.22161/ijaers.96.2
[3] Haq, M. I. K., Ramadhan, F. D., Az-Zahra, F., Kurniawati, L. & Helen, A. (2021). *'Classification of Water Potability Using Machine Learning Algorithms'*. International Conference on Artificial Intelligence and Big Data Analytics, 2021, pp. 1-5, doi: 10.1109/ICAIBDA53487.2021.9689727.
[4] MathWorks. (2020). Improving Classification Trees and Regression Trees - MATLAB & Simulink - MathWorks United Kingdom. Uk.Mathworks.com. https://www.mathworks.com/help/stats/improving-classification-trees-and-regression-trees.html
[5] Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5–32. https://doi.org/10.1023/a:1010933404324
[6] Huilgol, P. (2019) Accuracy vs. F1-Score, Analytics Vidhya. Available at: https://medium.com/analytics-vidhya/accuracy-vs-f1-score-6258237beca2
[7] MathWorks. (2020). Test Ensemble Quality - MATLAB & Simulink - MathWorks United Kingdom. Uk.Mathworks.com. https://uk.mathworks.com/help/stats/methods-to-evaluate-ensemble-quality.html
[8] Semparudhi. (2021) WATER POTABILITY PREDICTION WITH MACHINE LEARNING, medium.com. Available at: https://parudhi.medium.com/water-potability-prediction-with-machine-learning-8eea74c2970
[9] Breiman, L., Chen C & Liaw A. (2004). Using Random Forests to learn imbalanced Data. Department of Statistics, UC Berkeley, 666
[10] *principal component analysis* (no date) *OriginLab Corporation - Data Analysis and Graphing Software - 2D graphs, 3D graphs, Contour Plots, Statistical Charts, Data Exploration, Statistics, Curve Fitting, Signal Processing, and Peak Analysis*. Available at: https://www.originlab.com/doc/Origin-Help/PrincipleComp-Analysis#:~:text=PCA%20should%20be%20used%20mainly,0.3%2C%20PCA%20will%20not%20help. (Accessed: November 25, 2022).
[11] Breiman, L., J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees.* Boca Raton, FL: Chapman & Hall, 1984.
[12] Mftnakrsu (2022) *Water Quality Eda, Random Forest & Decision tree*, *Kaggle*. Kaggle. Available at: https://www.kaggle.com/code/mftnakrsu/water-quality-eda-random-forest-decision-tree (Accessed: November 28, 2022).
[13] B. Charbuty and A. M. Abdulazeez, "Classification based on decision tree algorithm for machine learning," *Journal of Applied Science and Technology Trends*, vol. 2, no. 01, pp. 20–28, 2021.