

**SWINBURNE UNIVERSITY OF TECHNOLOGY**

COS40007 – Artificial Intelligence for Engineering

Portfolio Assessment-1: “Hello Machine Learning for Engineering”

Student Name: Nguyen Cuong Nhat

Student ID: 104178590

Instructor: Dr. Pham Thi Kim Dung

Submission Date: 22/9/2024

# I. Dataset

This dataset contains water quality measurements aimed at assessing the potability of water, indicating whether it is suitable for human consumption. It includes features such as pH level, hardness, dissolved solids, chloramine concentration, sulfate levels, and other key water quality parameters. The target variable, "Potability," is binary, with 1 indicating potable water and 0 indicating non-potable water. The primary goal is to use these attributes to predict water safety, making the dataset suitable for a supervised binary classification task. This dataset offers valuable insights for water quality assessment and decision-making in public health and environmental management.

As an AI major student, I chose this dataset because it presents a real-world application of machine learning in a critical area: water safety. Predicting water potability is a key challenge for public health, and applying AI models to assess water quality attributes can significantly aid in making informed decisions for water treatment and ensuring safe consumption. This dataset allows me to explore and develop supervised classification models, enhancing my understanding of AI techniques while contributing to an impactful environmental and societal issue.

# II. Exploratory data analysis (EDA) summary

## Data understanding

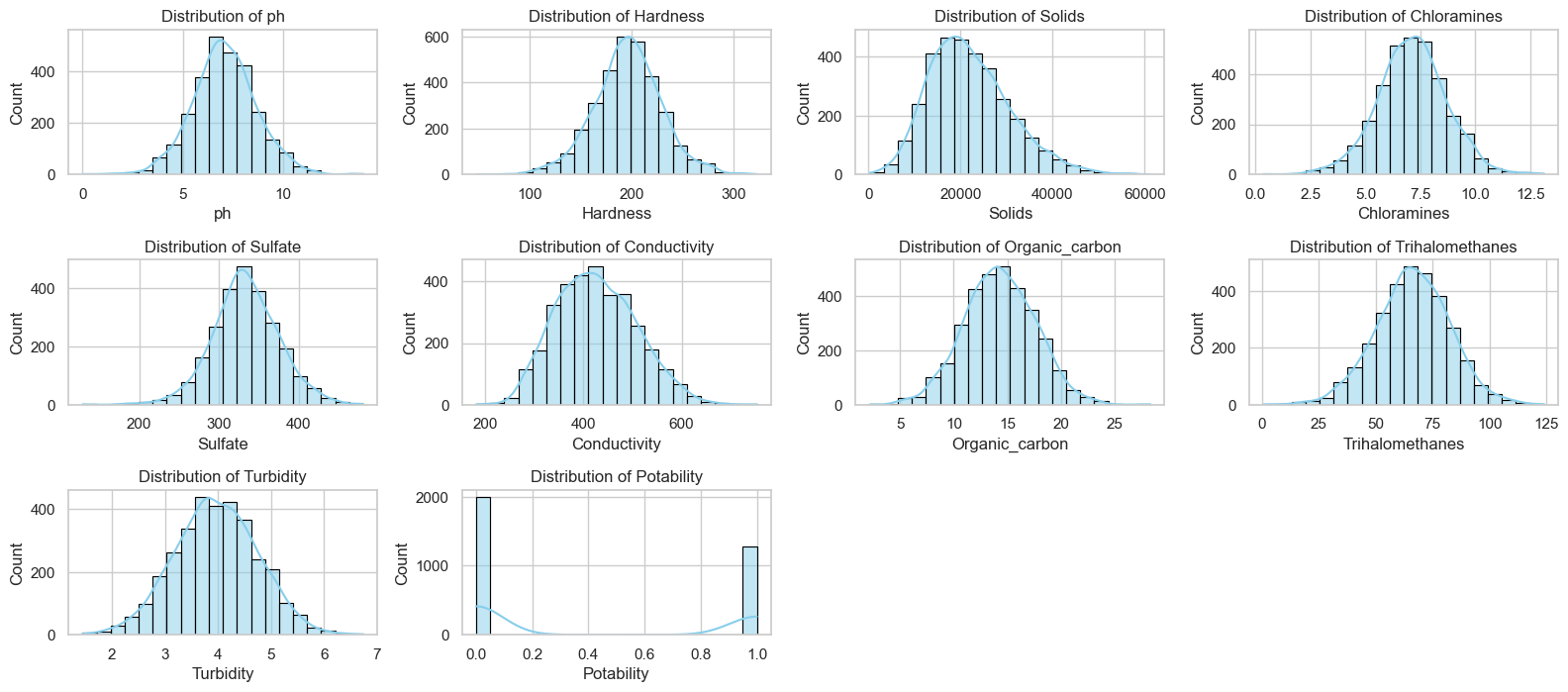
The dataset consists of 3,276 rows and 10 columns, each representing a water sample with various attributes related to water quality. The columns include:

* pH: The pH level of the water.
* Hardness: Water hardness, a measure of mineral content.
* Solids: Total dissolved solids in the water.
* Chloramines: Chloramines concentration in the water.
* Sulfate: Sulfate concentration in the water.
* Conductivity: Electrical conductivity of the water.
* Organic\_carbon: Organic carbon content in the water.
* Trihalomethanes: Trihalomethanes concentration in the water.
* Turbidity: Turbidity level, a measure of water clarity.
* Potability: Target variable; indicates water potability with values 1 (potable) and 0 (not potable).

These attributes are all continuous variables represented as `float64` data types. The target variable, "Potability," is a binary `int64` column that classifies water samples as either potable (1) or non-potable (0).

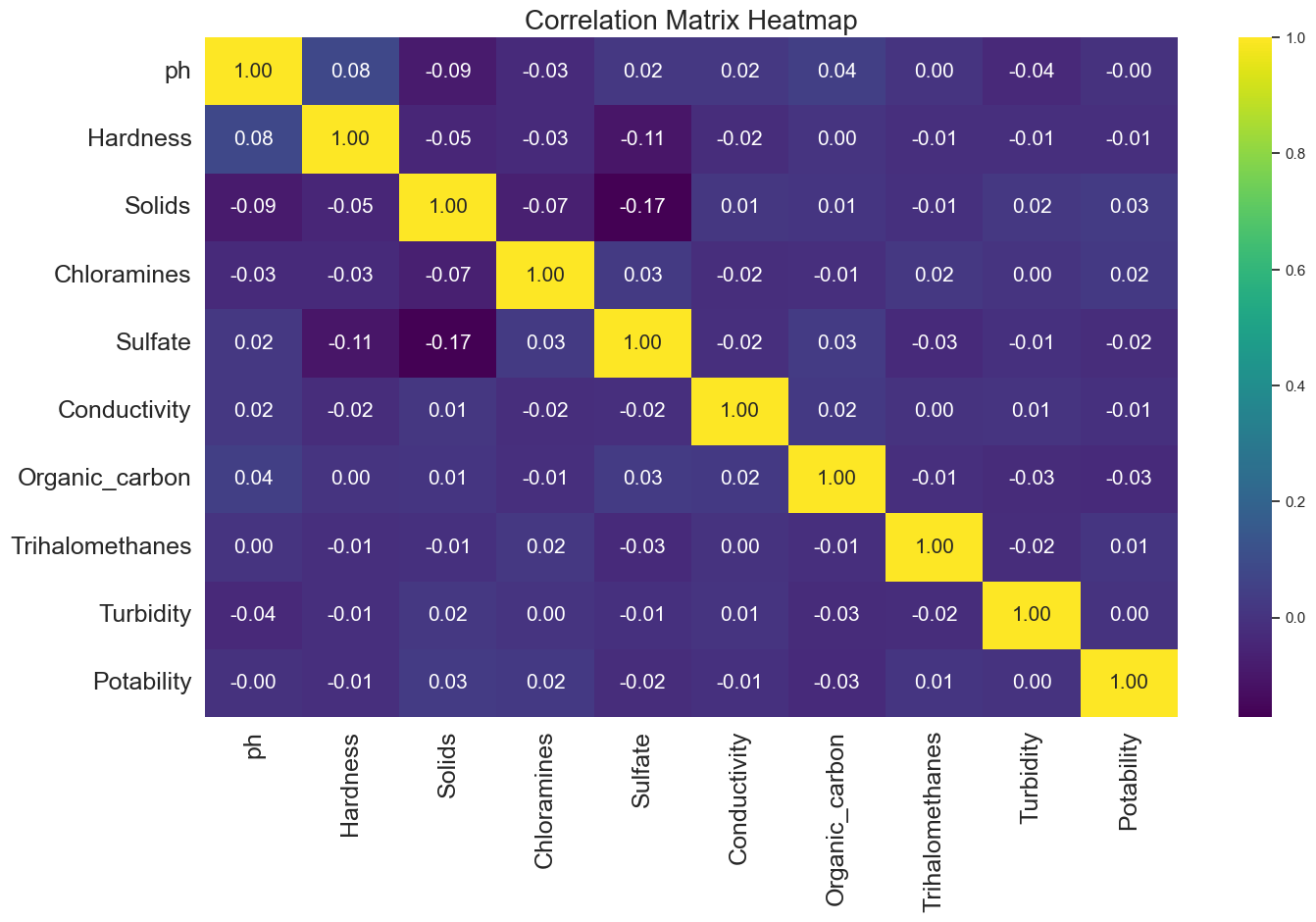
There are 9 features in the dataset with `float64` types, and 1 target variable with `int64`. This structure provides a solid foundation for a binary classification task, where machine learning models can be applied to predict whether a water sample is suitable for consumption based on the given features.

## Observation

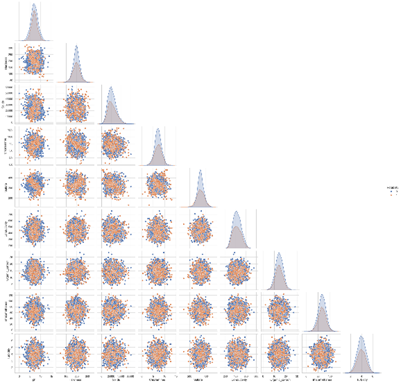


*Figure 1: Distribution of each feature in the dataset*

* **Feature Distributions**:
* Most features, such as ph, Hardness, Solids, Chloramines, Sulfate, Conductivity, Organic\_carbon, Trihalomethanes, and Turbidity, have approximately normal distributions. This suggests that the data is relatively balanced without strong skewness or extreme outliers for these features.
* However, the Potability target feature is imbalanced, with the majority of the water samples being classified as non-potable (label 0).

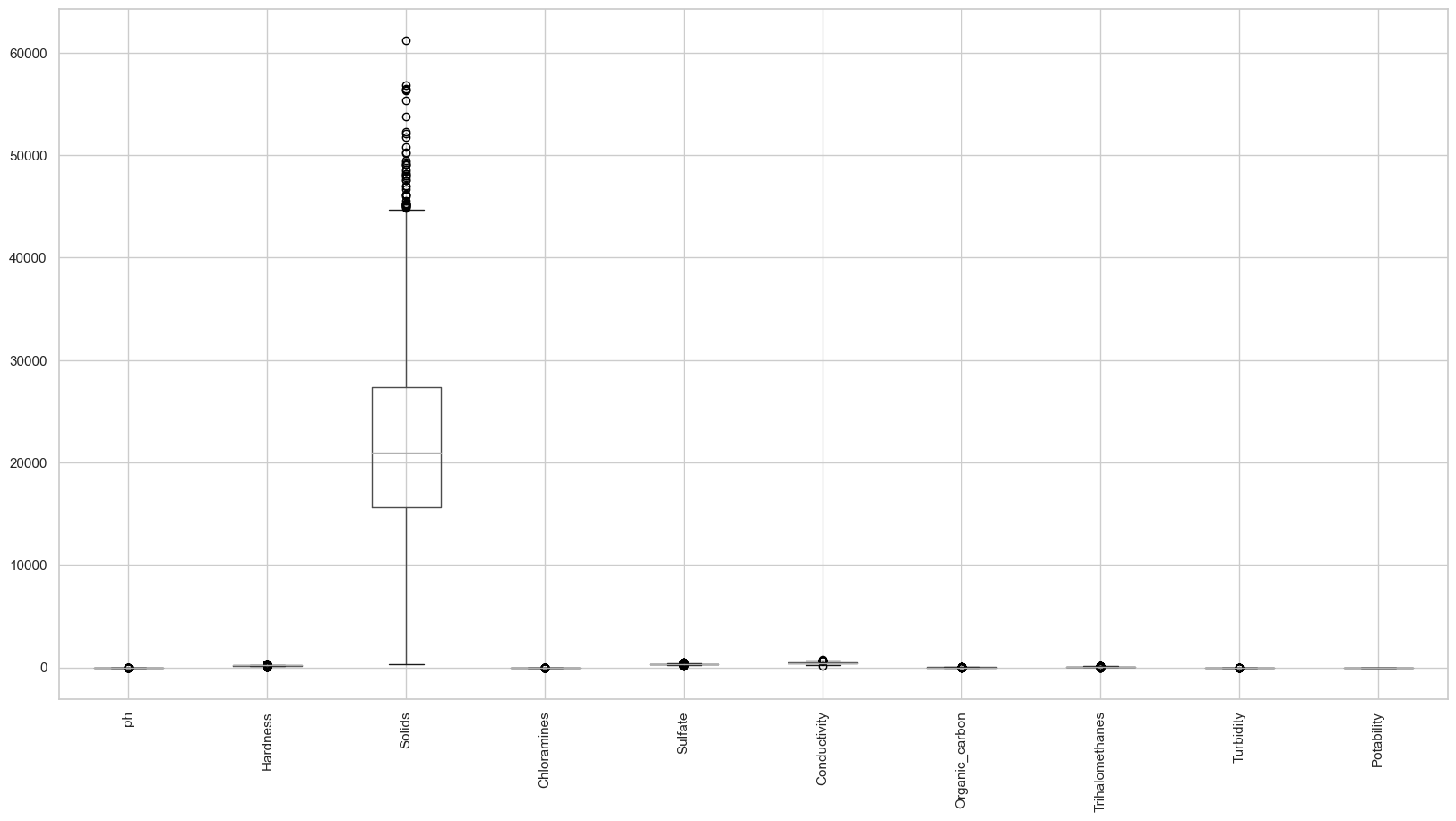


*Figure 2: Correlation heatmap*



*Figure 3: Pairplot to visualize relationships between numerical columns*

* **Correlations**: Most features show weak correlations with the target variable, "Potability." None of the features exhibit strong positive or negative relationships with "Potability." The heatmap confirms this, as the correlation values are close to zero. This indicates that each feature has limited linear impact on water potability classification.

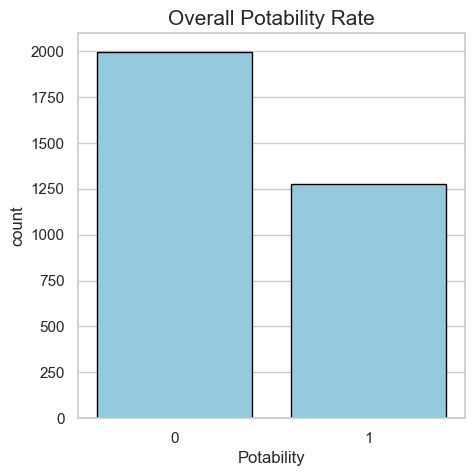


*Figure 4: Box plot to visualize outliers*

* **Outliers**: There are notable outliers in features such as ph, Hardness, Sulfate, Trihalomethanes, and Solids based on the boxplots. These outliers might require special treatment during modeling, such as capping or transformation.
* Given the weak correlations and presence of outliers, it might be beneficial to explore non-linear transformations or combinations of features (such as feature engineering) to improve the prediction accuracy. Feature scaling may also be useful, given the differences in scales across the features like Solids and Conductivity.

# III. Class labelling for target variable / developing ground truth data

Because our target variable “Portability” is already a categorical variable, we can skip the step for class labelling for target variable and developing ground truth data.

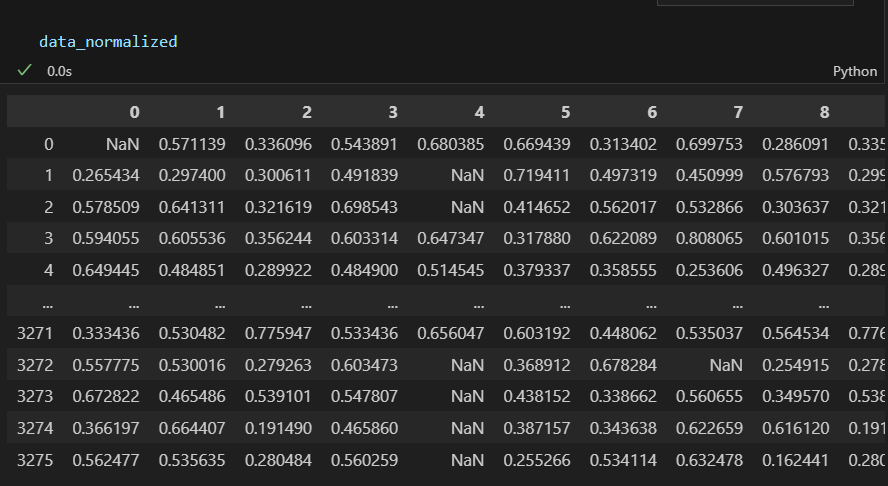


*Figure 5: Count the number of labels in the target feature*

# IV. Feature engineering and Feature selection

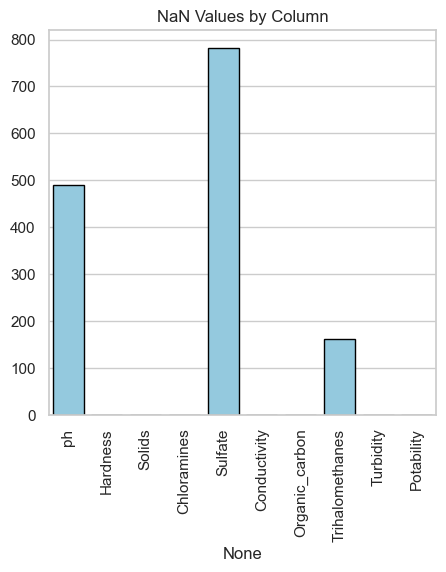
## Data normalization

MinMax scaling was applied to all features to ensure they are on a similar scale, which is important for certain machine learning model. Scaling helps to prevent features with larger ranges from is proportionately influencing the model's predictions.



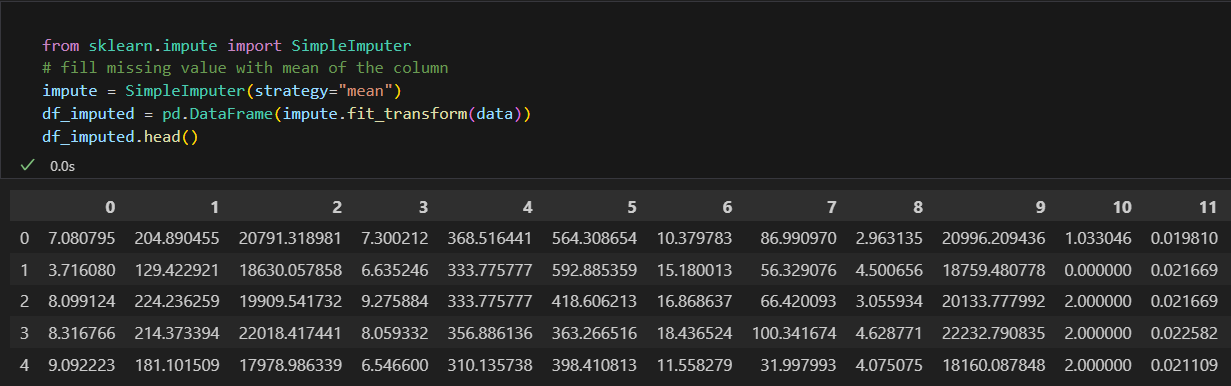
*Figure 6: Data after been normalized*

## Handle missing data and outliers



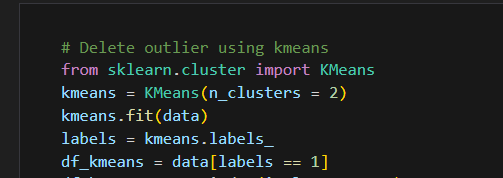
*Figures 7: NaN values count for each column*

I also replace missing values (NaNs) with the mean value of the corresponding columns. This process, known as imputation, is necessary because many machine learning models cannot work with missing data directly and ensures that missing data does not lead to biased or incomplete results.



*Figure 8: Data without missing value*

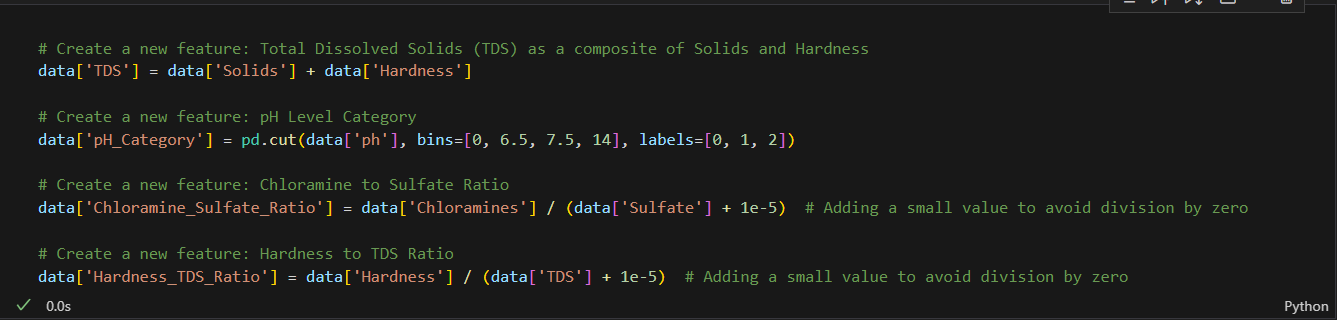
The outlier is also deleted using Kmeans. I use Kmeans to cluster each feature into 2 cluster. Therefore, 1 cluster is normal data, and the other is cluster containing outlier. And I just remove this cluster to ensure our data now no longer has outlier value that can affect the performance of the model.



*Figure 9: use Kmeans to filter missing value*

## Create new composite features

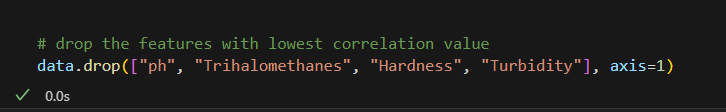
I have created interaction terms that may provide better predictive power, especially between features with weak correlations. Also, the pH value is binned to transform into categorical variable.



*Figure 10: Create composite features*

## Feature selection

Correlation analysis: based on the correlation map above, I will remove 4 features that has the lowest correlation with our target variable “Portability” (features that have the correlation value closest to 0)



*Figure 11: Drop features that have weak correlation value*

This selection process reduced the feature set to the most relevant variables, improving model efficiency and reducing the risk of high bias.

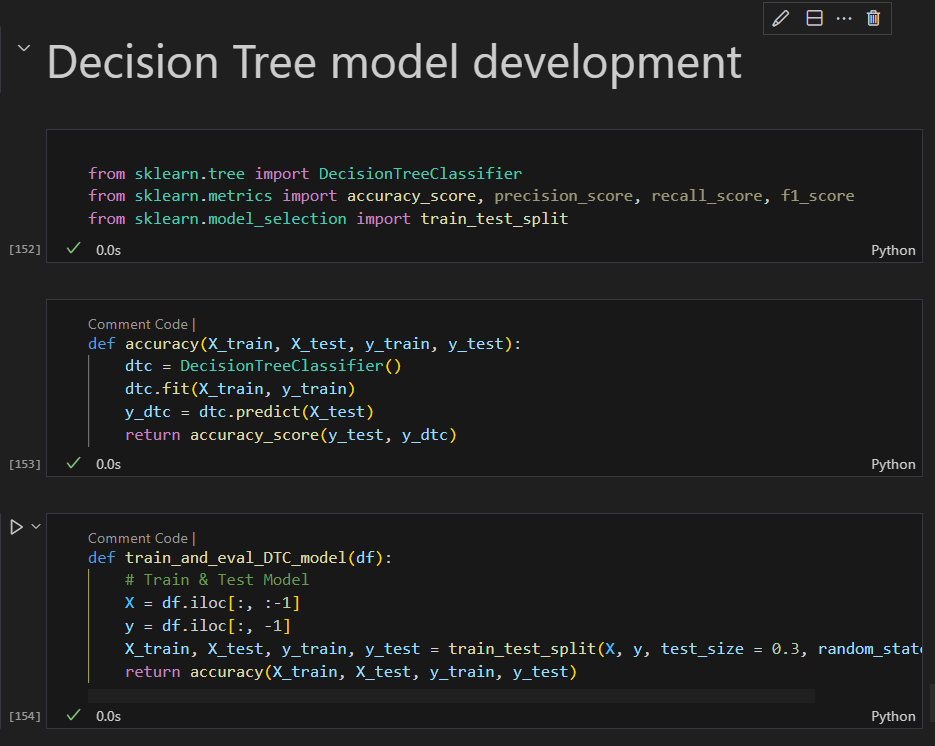
# V. Decision tree model development

A Decision Tree is a supervised machine learning model used for both classification and regression tasks. It splits the data into subsets based on feature values, creating a tree-like structure where each internal node represents a decision on a feature, each branch represents an outcome, and each leaf node represents a class label (for classification) or a predicted value (for regression).

I have created and evaluated decision tree models with 5 different set of features came up from the summary of EDA:

* Raw\_feature.csv (all features without normalisation and without composite features)
* Raw\_feature\_normalized.csv (all features with normalisation and without composite features)
* Engineered\_feature.csv (all features without normalisation and containing composite features)
* Engineered\_feature\_normalized.csv (all features with normalisation and containing composite features)
* Selected\_feature.csv (selected features with normalisation)

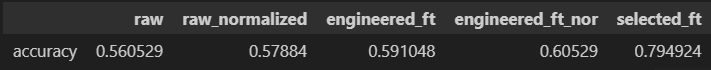
(Note: result can be slightly different when re-train the model)



*Figure 12: Decision tree model*

The data is split 70% for train and 30% for test. The evaluation metric I used is accuracy. It is a commonly used evaluation metric for decision tree models, especially for classification tasks. It measures the proportion of correctly predicted instances (both true positives and true negatives) out of the total number of instances in the dataset.

# VI. Experiments/Discussion



*Figure 13: Final comparison table*

The table above shows the accuracy of Decision Tree model under different feature processing scenarios. Here is a brief summary of the observation:

* The trend shows that preprocessing (such as normalization, feature engineering, and feature selection) leads to noticeable improvements in model performance.
* Feature selection has the most significant impact on accuracy, suggesting that not all features contribute equally to the model's performance, and removing irrelevant or redundant features can greatly enhance the result.
* While normalization helps in both the raw and engineered feature sets, the major improvements come from more advanced techniques like feature engineering and selection.

# VII. Conclusion

The Decision Tree model's performance improves as more advanced feature processing techniques are applied. Starting with the raw features, the accuracy was low, but normalization and feature engineering gradually improved results. The most substantial improvement came from feature selection, suggests that carefully selecting and engineering features plays a crucial role in enhancing model accuracy, indicating that not all features are equally important for prediction. Therefore, feature engineering and selection should be prioritized when optimizing model performance.

# VIII. Appendix

Colab notebook: https://colab.research.google.com/drive/1KyI7oQdW47FXBtH5npKUsw-6f1ow4f0t?usp=sharing