COMP5279N Programming for Data Science

Coursework 3: Data Analysis Project

Water Potability

- Kwok Kiu Ho, 21100880@stu.ucn.ac.uk
- Ka Ho Lai, 21100880@stu.ucn.ac.uk
- Hi-shih Reddy Yang, s21100880@stu.ucn.ac.uk
- Bhargavan Jr, s21100880@stu.ucn.ac.uk

Project Plan

About the Data

Data is one of the most important part of a project, in our analysis we are going to use the dataset called Water Potability. This dataset is an commonly used dataset from Kaggle and contains 10 variables that affect the water quality which is determining the quality of drinking water, presented by Kaggle.

- **Dataset Overview:** The dataset contains 3,276 rows and 10 columns with 9 numerical variables (features) and one 10 categorical variable (target). Each row represents a water sample, and the columns represent both the chemical and physical properties of the water, such as pH, hardness, solids, chloramines, sulfate, conductivity, organic carbon, trihalomethanes, and turbidity, along with a binary indicator (0 or 1) of whether the sample is potable (safe to drink) or not.
- **Potential Applications:** This dataset is commonly used for exploratory data analysis, feature engineering, and machine learning classification tasks to predict water potability based on these chemical and physical properties.

Attributes/Variables

- **pH value:** The pH of water is a measure of how acidic or alkaline it is, ranging from 0 to 14. It can affect the taste of water and is a key indicator for water quality. Extreme pH values can make water corrosive or unsafe to drink. The recommended pH range for drinking water is between about 6.5 and 8.5. (WHO guidelines)
- **Hardness:** Hardness was initially described as water's ability to precipitate soap due to its Calcium and Magnesium content. It primarily measures the concentration of dissolved calcium and magnesium ions. Hard water can affect the taste and scale formation in pipes and appliances, but is generally considered safe to drink.
- **Solids (Total dissolved solids - TDS):** It high TDS indicates that more minerals and salts are present in the water. It is a measure of the combined content of all inorganic and organic substances dissolved in water.
- **Chloramines:** Free and combined chlorine and bromine are added to water supplies under **EPA/WHO** rules (4.0 ppm/mg/L and 5.0 ppm/mg/L) to eliminate bacteria and viruses. Chlorine is a disinfectant commonly added to water to kill pathogens and prevent waterborne diseases. Chloramines are a group of compounds formed when chlorine reacts with ammonia in water. They are used as an alternative to chlorine for disinfection, sometimes preferred because they last longer in the water and can be more stable than chlorine.
- **Sulfate:** Sulfate is a natural substance found in most water sources. Its concentration can vary significantly depending on the geological origin of the water and the presence of industrial pollution. The recommended maximum level in drinking water is typically around 250 mg/L (WHO guidelines). High sulfate levels can cause a laxative effect and give water a bitter taste.
- **Organic Carbon:** Total Organic Carbon (TOC) is a source water is derived from both natural organic matter (NOM) and anthropogenic (synthetic) sources. The presence of organic carbon is a concern because it can react with disinfectants like chlorine to form harmful disinfection by-products (DBPs) such as trihalomethanes (THMs).
- **Trihalomethanes:** THMs are compounds that can be found in chlorinated water. THM concentrations in drinking water vary depending on the amount of organic matter in the source water, the dose of chlorine used for disinfection, and the water temperature. The presence of THMs is a health concern due to their potential carcinogenic effects.
- **Turbidity:** The amount of solid stuff present in the suspended state determines how turbid the water is. The term is used to describe the quality of water discharged or supplied in relation to drinking water and assesses the high-working capabilities of water treatment plants. The recommended maximum turbidity for drinking water is typically 5 NTU (Nephelometric Turbidity Units).
- **Potability:** Indicates whether water is safe for human consumption, with 0 indicating undrinkable and 1 indicating drinkable.

Attributes used:
some examples from the dataset:
pH, Hardness, Solids, Chloramines, Sulfate, Conductivity, Organic_carbon, Trihalomethanes, Turbidity, Potability

Project Aim and Objectives

According to a study by the United Nations, 2.2 billion people reside in water-stressed areas, and unless we work together to avert this **Sustainable Development Goal 6 (SDG 6)** challenge, more people will be impacted as climate change advances in the coming decades.

Access to clean drinking water is crucial for health, environmental human right, and a component of successful health protection strategies.

Testing accurate chemical and microbial parameters, such as pH and turbidity, is the objective behind water.

- Identification of the key elements that affect the potability of the water
- Exploration of the relationships between different water quality parameters
- Investigating the impact of outliers on the model's performance and determine if they represent a genuine phenomenon or noise
- Implementing a machine learning model capable of categorizing the input information and predicting water potability
- Evaluating the model's performance using appropriate metrics (accuracy, precision, recall, F1-score, confusion matrix) to ensure its reliability in classifying the potability of water depending on the input parameters.

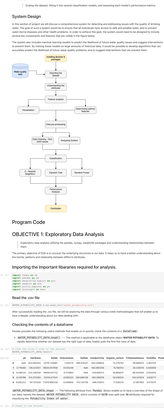
Specific questions

- **OBJECTIVE 1**
  - Exploratory data analysis utilizing the pandas, numpy, matplotlib packages and understanding relationships between variables
- **OBJECTIVE 2**
  - Visualizing and examining the variables to identify the factors that have the major influence on the potability index.
- **OBJECTIVE 3**
  - Creating the most effective technique for pre-processing and extracting the key attributes required for better output of the project.
- **OBJECTIVE 4**
  - Scaling the dataset, fitting it into several classification models, and assessing each model's performance metrics.

System Design

In this section, we will provide an extensive comprehensive system for collecting and addressing issues with the quality of drinking water. The goal of such a system would be to ensure that all individuals have access to safe and potable water, and to prevent waterborne diseases and other health concerns. The system would also have the capacity to monitor and control water quality in real-time, to ensure that any contaminants are detected and dealt with promptly.

The system also includes machine learning models to predict the likelihood of future water quality issues and suggest interventions or preventive measures based on historical data and current trends. The system would also have a user-friendly interface that provides stakeholders, such as water utilities and regulatory agencies, with actionable insights and suggested interventions that can prevent issues.



Program Code

OBJECTIVE 1: Exploratory Data Analysis

- Exploratory data analysis utilizing the pandas, numpy, matplotlib packages and understanding relationships between variables

The primary objective of EDA is to provide the preliminary structures to our data. It helps us to have a better understanding about the trends, patterns and relationship between variables.

Importing the Important libraries required for analysis.

```
%matplotlib inline
%pandas_setup -df -ts
%load_ext autotime
%load_ext rpy2.ipython
%load_ext pretty_jsonl
%load_ext jupyterlab_spreadsheets
```

Read the .csv file

```
water_potability_data = pd.read_csv('./water_potability.csv')
```

The dataset is in the form of a .csv file, we will be importing the data through various data methodologies that will enable us to have a better understanding about our data wrangling.

Checking the contents of a dataframe

```
water_potability_data.shape
```

```
(3276, 10)
```

```
water_potability_data.info()
```

```
<class pandas.core.frame.DataFrame>
RangeIndex: 3276 entries, 0 to 3275
Data columns (total 10 columns):
 ph               2785 non-null   float64
 Hardness         3276 non-null   float64
 Solids           3276 non-null   float64
 Chloramines      3276 non-null   float64
 Sulfate          2495 non-null   float64
 Conductivity     3276 non-null   float64
 Organic_carbon   3276 non-null   float64
 Trihalomethanes  3114 non-null   float64
 Turbidity        3276 non-null   float64
 Potability       3276 non-null   int64
dtypes: float64(9), int64(1)
memory usage: 256.1 KB
```

```
water_potability_data.head(5)
```

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
0	NaN	204.890455	20791.318981	7.300212	NaN	568.304671	14.284970	64.846964	2.559708	0
1	3.716080	129.422921	18630.057858	6.635246	NaN	592.885359	15.180013	56.329076	4.500656	0
2	8.099124	224.236259	19909.541732	9.275884	NaN	418.606213	16.868637	66.420093	3.055934	0
3	8.316766	214.373394	22018.417441	8.059332	356.886136	363.266516	18.436524	100.341674	4.628771	0
4	9.092223	181.101509	17978.986339	6.546600	310.135738	398.410813	11.558279	31.997993	4.075075	0

```
water_potability_data.describe(percentiles=[0.25,0.50,0.75,0.90,0.95,0.99]).T
```

Key Observations

- NEPOR_PERCENTILES_DEFAULT dataset has 3276 records across 10 attributes.
- Only ph and Sulfate have missing values (NaN).
- ph has 391 missing values and Sulfate has 781 missing values.
- Attributes such as ph, Sulfate and Trihalomethanes might have missing values.

```
water_potability_data.describe(percentiles=[0.25,0.50,0.75,0.90,0.95,0.99]).T
```

	count	mean	std	min	25%	50%	75%	90%	95%	99%	max
ph	2785.0	7.080795	1.594320	0.000000	6.093092	7.036752	8.062066	8.772776	9.094379	9.732000	14.000000
Hardness	3276.0	196.369496	32.879761	47.432000	176.850538	196.967627	216.667456	232.099000	241.000000	258.000000	323.124000
Solids	3276.0	22014.092526	8768.570828	320.942611	15666.690297	20927.833607	27332.762127	37332.190502	43122.109609	56488.672413	61227.196008
Chloramines	3276.0	7.122277	1.583085	0.352000	6.127421	7.130299	8.114887	8.895000	9.323000	10.390000	15.480000
Sulfate	2495.0	333.775777	41.416840	129.000000	307.699498	333.073546	359.948574	390.683993	404.000000	436.000000	481.030642
Conductivity	3276.0	426.205111	80.824064	181.483754	365.734414	421.884968	481.792304	563.881481	591.643671	753.342620	753.342620
Organic_carbon	3276.0	14.284970	3.308162	2.200000	12.065762	14.218338	16.557652	18.309337	19.927089	22.337333	28.300000
Trihalomethanes	3114.0	66.396293	16.175008	8.577013	55.844536	66.622485	77.337473	87.990327	94.852030	104.011326	124.000000
Turbidity	3276.0	3.966786	0.780769	1.450000	3.439711	3.955028	4.500656	5.052000	5.370000	6.099000	6.739000
Potability	3276.0	0.390110	0.487849	0.000000	0.000000	0.000000	1.000000	1.000000	1.000000	1.000000	1.000000

Observations

- Following both NEPOR's values can not roughly say that the data is normally distributed for every attributes/features that are numerical in nature.
- Most of the attributes have extremely high values which are treated as OUTLIERS. We can understand from the difference between the 75% and max values.
- The ultimate observation is that we have a problem with the data, especially outliers. Outliers are the most important thing to be taken into consideration.

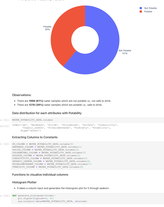
Checking missing values in our dataset with graphical representation

Missing values are a typical issue in many real-world datasets. They can stem from a variety of machine learning models and/or data collection errors. Many machine learning models find this, and we discuss it today, the existence of missing values in our **DF**.

```
water_potability_data.isnull().sum.sort_values(ascending=False).to_frame(name='missing_data').T
```

	missing_data
Sulfate	781
ph	491
Trihalomethanes	162
Hardness	0
Solids	0
Chloramines	0
Conductivity	0
Organic_carbon	0
Turbidity	0
Potability	0

```
sns.heatmap(water_potability_data.isnull(),cmap='viridis')
```



- **Key observations**
  - In our dataset NEPOR_PERCENTILES_DEFAULT, 3 attributes consists of missing values.
  - Sulfate has the highest number of missing values, followed by ph, and Trihalomethanes. Only the type of the three columns is **float** in nature.
  - **ph** has 491 missing values
  - **Sulfate** has 781 missing values
  - **Trihalomethanes** has 162 null values.

Since missing data can directly impact our model's accuracy, we will be imputing the missing values/rows the data before fitting it into our machine learning models.

Results - Objective 1

- Make powerfully relevant has **OTHER** distributed across **30** unique attributes.
- Out of 30 attributes 8 are in **Float64** and the remaining are in **int64**. Number of attributes in **Float64** are more than **int64**.
- There are more columns (**ph**, **SULFATE64**, and **TRIHALOMETHANE**) that contain missing values.
- Most of the attributes have extremely high values which are treated as **OUTLIERS**. We can understand from the difference between the **75%** and **max** values.
- The ultimate observation is that we have a problem with the data, especially **outliers**. Outliers are the most important thing to be taken into consideration.

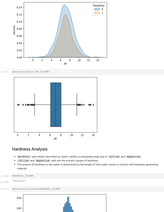
Checking missing values in our dataset with graphical representation

Missing values are a typical issue in many real-world datasets. They can stem from a variety of machine learning models and/or data collection errors. Many machine learning models find this, and we discuss it today, the existence of missing values in our **DF**.

```
water_potability_data.isnull().sum.sort_values(ascending=False).to_frame(name='missing_data').T
```

	missing_data
Sulfate	781
ph	491
Trihalomethanes	162
Hardness	0
Solids	0
Chloramines	0
Conductivity	0
Organic_carbon	0
Turbidity	0
Potability	0

```
sns.heatmap(water_potability_data.isnull(),cmap='viridis')
```



- **Key observations**
  - In our dataset NEPOR_PERCENTILES_DEFAULT, 3 attributes consists of missing values.
  - Sulfate has the highest number of missing values, followed by ph, and Trihalomethanes. Only the type of the three columns is **float** in nature.
  - **ph** has 491 missing values
  - **Sulfate** has 781 missing values
  - **Trihalomethanes** has 162 null values.

Since missing data can directly impact our model's accuracy, we will be imputing the missing values/rows the data before fitting it into our machine learning models.

Results - Objective 1

- Make powerfully relevant has **OTHER** distributed across **30** unique attributes.
- Out of 30 attributes 8 are in **Float64** and the remaining are in **int64**. Number of attributes in **Float64** are more than **int64**.
- There are more columns (**ph**, **SULFATE64**, and **TRIHALOMETHANE**) that contain missing values.
- Most of the attributes have extremely high values which are treated as **OUTLIERS**. We can understand from the difference between the **75%** and **max** values.
- The ultimate observation is that we have a problem with the data, especially **outliers**. Outliers are the most important thing to be taken into consideration.

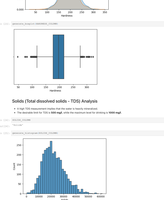
OBJECTIVE 2: Visualization

- Visualizing and examining the variables to identify the factors that has the major influence on the potability index.

Potability Analysis

- Indicates whether water is safe for human consumption.
- 1 indicates drinkable and 0 indicates undrinkable.

```
%%R -i NEPOR_PERCENTILES_DEFAULT -w 1000 -h 500
NEPOR_PERCENTILES_DEFAULT$Potability <- factor(NEPOR_PERCENTILES_DEFAULT$Potability)
ggplot(NEPOR_PERCENTILES_DEFAULT, aes(x=Potability, fill=Potability))+
  geom_bar(stat="count")+
  ggtitle("Pie Chart of Potability Feature")+
  theme(plot.title=element_text(size=20))+
  coord_polar(theta="y")+
  theme_void()+
  scale_fill_manual(values=c("#ff0000","#00ff00"))+
  geom_text(aligned=TRUE,stat="count",position=position_stack(vjust=0.5),size=5)
```

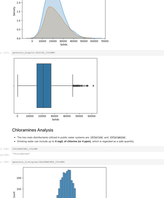


Observations

- There is **1998** **NOT** potable samples which are not potable, i.e. not safe to drink.
- There is **1278** **POTABLE** samples which are potable, i.e. safe to drink.

Data distribution for each attributes/features

```
%%R -i NEPOR_PERCENTILES_DEFAULT -w 1000 -h 1000
NEPOR_PERCENTILES_DEFAULT$Potability <- factor(NEPOR_PERCENTILES_DEFAULT$Potability)
ggplot(gather(NEPOR_PERCENTILES_DEFAULT, cols, value), aes(x=value, fill=Potability))+
  geom_histogram(bins=30)+
  facet_wrap(~cols, scales="free")
```



Extracting Columns to Constants

```
%% rm -rf NEPOR_PERCENTILES_DEFAULT_COLUMNS
%%writefile NEPOR_PERCENTILES_DEFAULT_COLUMNS.txt
ph,Hardness,Solids,Chloramines,Sulfate,Conductivity,Organic_carbon,Trihalomethanes,Turbidity,Potability
%%cat NEPOR_PERCENTILES_DEFAULT_COLUMNS.txt
%%rm -rf NEPOR_PERCENTILES_DEFAULT_COLUMNS.sh
%%writefile NEPOR_PERCENTILES_DEFAULT_COLUMNS.sh
export NEPOR_PERCENTILES_DEFAULT_COLUMNS=`cat NEPOR_PERCENTILES_DEFAULT_COLUMNS.txt`
echo $NEPOR_PERCENTILES_DEFAULT_COLUMNS
%%chmod +x NEPOR_PERCENTILES_DEFAULT_COLUMNS.sh
%%./NEPOR_PERCENTILES_DEFAULT_COLUMNS.sh
```

Functions to visualize individual columns

Histogram Plotter

- Creates a count chart and generates the mean and std for it through seaborn.

```
%%NEPOR_PERCENTILES_DEFAULT_COLUMNS.sh
%%rm -rf NEPOR_PERCENTILES_DEFAULT_HISTPLOT.sh
%%writefile NEPOR_PERCENTILES_DEFAULT_HISTPLOT.sh
export NEPOR_PERCENTILES_DEFAULT_HISTPLOT="sns.histplot(data=NEPOR_PERCENTILES_DEFAULT,x=f'{NEPOR_PERCENTILES_DEFAULT_COLUMNS[0]}',hue='Potability',kde=True,bins=30,palette='Set1')"
%%chmod +x NEPOR_PERCENTILES_DEFAULT_HISTPLOT.sh
%%./NEPOR_PERCENTILES_DEFAULT_HISTPLOT.sh
```

Kernel Distribution Estimation (KDE) Plotter

- Creates a count chart and generates the mean and std for it through seaborn.

```
%%NEPOR_PERCENTILES_DEFAULT_HISTPLOT.sh
%%rm -rf NEPOR_PERCENTILES_DEFAULT_KDEPLOT.sh
%%writefile NEPOR_PERCENTILES_DEFAULT_KDEPLOT.sh
export NEPOR_PERCENTILES_DEFAULT_KDEPLOT="sns.kdeplot(data=NEPOR_PERCENTILES_DEFAULT,x=f'{NEPOR_PERCENTILES_DEFAULT_COLUMNS[0]}',hue='Potability',fill=True)"
%%chmod +x NEPOR_PERCENTILES_DEFAULT_KDEPLOT.sh
%%./NEPOR_PERCENTILES_DEFAULT_KDEPLOT.sh
```

Box Plotter

- Creates a boxplot chart and generates the std and mean for it through seaborn.

```
%%NEPOR_PERCENTILES_DEFAULT_KDEPLOT.sh
%%rm -rf NEPOR_PERCENTILES_DEFAULT_BOXPLOT.sh
%%writefile NEPOR_PERCENTILES_DEFAULT_BOXPLOT.sh
```

PH Analysis

- The **pH** column is an essential metric in determining its **SUITABILITY** for human
- It can also indicate if the water is **ACIDIC** or **STRICKEN**.
- The World Health Organization (WHO) recommends a pH level within acceptable ranges.
- The current investigation ranges around **6.5 to 8.5**, which aligns with the WHO guidelines range.

```
%%% PH_COLUMN
%%%
```

```
%%NEPOR_PERCENTILES_DEFAULT_BOXPLOT.sh
```



```
%%NEPOR_PERCENTILES_DEFAULT_KDEPLOT.sh
```

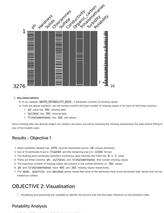
```
%%NEPOR_PERCENTILES_DEFAULT_HISTPLOT.sh
```

Hardness Analysis

- **Hardness** was initially described as water's ability to precipitate soap due to its Calcium and Magnesium content.
- Calcium and magnesium salts are the primary sources of hardness.
- The ability to precipitate soap is a key characteristic in the context of hard water, commonly in contact with traditional cleaning products.

```
%%% HARDNESS_COLUMN
%%%
```

```
%%NEPOR_PERCENTILES_DEFAULT_BOXPLOT.sh
```



```
%%NEPOR_PERCENTILES_DEFAULT_KDEPLOT.sh
```



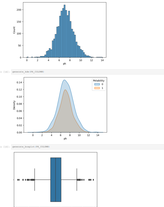
```
%%NEPOR_PERCENTILES_DEFAULT_HISTPLOT.sh
```

Solids (Total dissolved solids - TDS) Analysis

- It high TDS measurement implies that the water is heavily chlorinated.
- The recommended limit is **500 mg/l** while the maximum level for drinking is **1000 mg/l**

```
%%% SOLIDS_COLUMN
%%%
```

```
%%NEPOR_PERCENTILES_DEFAULT_BOXPLOT.sh
```



```
%%NEPOR_PERCENTILES_DEFAULT_KDEPLOT.sh
```

```
%%NEPOR_PERCENTILES_DEFAULT_HISTPLOT.sh
```

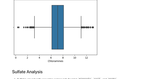

Chloramines Analysis

- The main disinfectant in drinking water is chlorine and chloramines are utilized in some locations.
- Drinking water has levels as high **4 mg/L or chlorine or 4 ppm**, which is regarded as a safe quantity.

```
%%% CHLORAMINES_COLUMN
%%%
```

```
%%NEPOR_PERCENTILES_DEFAULT_BOXPLOT.sh
```

```
In [18]: sns.boxplot(x_train[CONTINUOUS_COLUMNS]
```



```
In [50]: # DT Classifier
         dt = DecisionTreeClassifier(criterion='gini')
         cross_score = cross_val_score(dt,X_train,y_train,cv=5)
         print("%s: %f " % ('DT', cross_score.mean()))
```

```
In [51]: # Decision Tree Tuning
         dt = DecisionTreeClassifier()
         from sklearn.model_selection import GridSearchCV
         grid = {'max_depth': [1,2,3,4,5,6,7,8,9,10],
                 'max_features': ['auto', 'sqrt', 'log2'],
                 'min_samples_leaf': [1,2,3,4,5],
                 'min_samples_split': [1,2,3,4,5,6,7,8,9,10]
                 }
         dt_cv = GridSearchCV(dt, grid, cv=5)
         dt_cv.fit(X_train,y_train)
```

```
Out[51]: GridSearchCV(cv=5, error_score=nan,
                      estimator=DecisionTreeClassifier(ccp_alpha=0.0, class_weight=N
                                                       criterion='gini', max_depth=N
                                                       max_features=None, max_leaf_no
                                                       min_impurity_decrease=0.0,
                                                       min_impurity_split=None,
                                                       min_samples_leaf=1,
                                                       min_samples_split=2,
                                                       min_weight_fraction_leaf=0.0,
                                                       presort='deprecated',
                                                       random_state=None,
                                                       splitter='best'),
                      iid='deprecated', n_jobs=None,
                      param_grid={'max_depth': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
                                  'max_features': ['auto', 'sqrt', 'log2'],
                                  'min_samples_leaf': [1, 2, 3, 4, 5],
                                  'min_samples_split': [1, 2, 3, 4, 5, 6, 7, 8, 9,
                                                        10]},
                      pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
                      scoring=None, verbose=0)
```

```
In [52]: # Decision Tree Tuning
         dt = DecisionTreeClassifier(criterion='gini')
         cross_score = cross_val_score(dt,X_train,y_train,cv=5)
         print("%s: %f " % ('DT', cross_score.mean()))
```

```
DT: 0.777778
```

Results - Objective 4

- The **Decision Tree** & **Random** neighbours classification models are identified that the accuracy is **84%**.
- For the **SVC** model that is a logistic is being set to **20**, which provides a good balance between sensitivity and robustness.
- Additionally, we achieve **84.0%** with the model and obtained an accuracy of **78%** which was less accurate than a **DNN** model.
- While accuracy using the **Decision** was not as good as the **minimum number of samples** required to split an internal node was **2** and **Gini** was the function to measure the quality of a split.
- Finally we used **random forest** classification model on our dataset, and obtained an accuracy of **80%**.
- Finally, we used **Decision Tree** classification model on our dataset, and obtained an accuracy of **78%**.
- Now, we have the **Random Forest** model as the most effective model for our dataset.
- Which is the popular we see also conclusion that **random forests model** is the prime **is overfitting**.

Conclusion

Achievements

- In order to develop an efficient model that can identify whether a particular water sample is safe for drinking we successfully performed an EDA utilizing pandas, numpy, matplotlib packages and tried to understand how different features affect our target variable potability.
- We visualized and examined each attribute such as **pH**, **hardness**, **Solids** etc. to identify which factors had the major influence on the potability index and further scaled the dataset before training our models.
- Further, before training the model we identified the presence of **NaN** values and **dropped** that might raise the accuracy. Also, we used **Standard Scaler** to scale our features before training our models.
- We trained our data on three different kinds models namely **DNN**, **Decision Tree** and **Random Forest**, and evaluated them on our test dataset/unseen data where we identified the **Random Forest** performed efficiently and gave us the maximum accuracy.

Limitations

- The major limitation of the dataset is the size of the available data. The dataset has just **3276** records of different water bodies which were based on **3276** unique sources after data cleaning.
- The data set is highly imbalanced, which is a major challenge for the classifier used to train our models.
- It would be better, if we could have derived **new features** from the existing one and tested the accuracy of the model.
- For study would have been more accurate if our dataset included more parameters such as **dissolved oxygen**, **pesticide**, **mercury** etc.
- We could have been able to determine models based water quality if our dataset contained a **location parameter**, since it would be suitable for future studies.

Future Work

- Improving the training data: The quality and diversity of the training data can have a significant impact on the performance of the model. It is important to ensure that the training data is representative of the real-world data that the model will be used to classify and that it includes a wide variety of samples, which will ensuring that the data is representative of the wide variety of water samples that the model will be used to classify.
- The model could be used in other applications such as to identify potential sources of contamination in water supplies, or to monitor the effectiveness of water treatment processes over time.
- Investigating the effects of climate change on water quality: Climate change is having a significant impact on water quality and availability in many parts of the world, and it is important to understand the potential impacts of climate change on water quality and develop strategies for incorporating this information into the model to improve its accuracy.
- Developing a mobile app or a web interface for the model: This would make it easier for citizens to monitor drinking water and access information about the water quality in their area.