DATA ANALYSIS WITH COGNOS

Group 2

Project 11 – WATER ANALYSIS



COLLEGE CODE:5113

TEAM 10

TEAM LEADER: INIYAN P – au511321104032

TEAM MATES:

KAMALESH L - au511321104039

DHANUSH M – au511321104021

MUKESH C - au511321104057

KUGAN K - au511321104044

WATER QUALITY ANALYSIS

PHASE 2:INNOVATION

Water quality analysis involves testing and evaluating the characteristics of water to determine its safety and suitability for various purposes. It includes assessing parameters such as pH, dissolved oxygen, turbidity, and the presence of contaminants like bacteria, chemicals, and heavy metals. This analysis helps in ensuring that water meets the required standards for drinking, industrial use, and environmental protection.

**Introduction:**

Being able to provide enough fresh drinking water is a core requirement. Within the climate change debate, one of the largest challenges is ensuring enough freshwater to survive. Water quality is a big concern that impacts all the specifies. Only about three percent of Earth’s water is freshwater. Of that, only 1.2 percent can be used as drinking water, with the remainder locked up in glaciers, ice caps, and permafrost, or buried deep in the ground. Using a data-driven approach to assess the features that impact the water quality could greatly improve our understanding of what makes water drinkable.

We will seek to find hidden insights with data analysis techniques using pandas and numpy. For the data visualizations, the matplotlib and seaborn libraries will be used.

**Goal:**

The goal of water quality analysis using data analytics is to assess and monitor the quality of water in various environmental settings, such as natural bodies of water, industrial processes, drinking water supplies, and wastewater treatment systems. This analysis aims to ensure the safety of water for human consumption, protect aquatic ecosystems, and maintain overall environmental health. Data analytics plays a crucial role in achieving this goal by providing insights, patterns, and predictions from large datasets related to water quality.

**Data set:**

For this piece of analysis, the Water Quality dataset has been taken from Kaggle¹.

Dataset link:

<https://www.kaggle.com/datasets/adityakadiwal/water-potability/>

## About dataset

The water\_potability.csv file contains water quality metrics for 3276 different water bodies.

**1. pH value:** PH is an important parameter in evaluating the acid–base balance of water. It is also the indicator of acidic or alkaline condition of water status. WHO has recommended a maximum permissible limit of pH from 6.5 to 8.5. The current investigation ranges were 6.52–6.83 which are in the range of WHO standards.

**2. Hardness:** Hardness is mainly caused by calcium and magnesium salts. These salts are dissolved from geologic deposits through which water travels. The length of time water is in contact with hardness producing material helps determine how much hardness there is in raw water. Hardness was originally defined as the capacity of water to precipitate soap caused by Calcium and Magnesium.

**3. Solids (Total dissolved solids - TDS):** Water has the ability to dissolve a wide range of inorganic and some organic minerals or salts such as potassium, calcium, sodium, bicarbonates, chlorides, magnesium, sulfates etc. These minerals produced an unwanted taste and diluted color in the appearance of water. This is the important parameter for the use of water. The water with high TDS value indicates that water is highly mineralized. The Desired limit for TDS is 500 mg/l and maximum limit is 1000 mg/l which is prescribed for drinking purpose.

**4. Chloramines:** Chlorine and chloramine are the major disinfectants used in public water systems. Chloramines are most commonly formed when ammonia is added to chlorine to treat drinking water. Chlorine levels up to 4 milligrams per liter (mg/L or 4 parts per million (ppm)) are considered safe in drinking water.

**5. Sulfate:** Sulfates are naturally occurring substances that are found in minerals, soil, and rocks. They are present in ambient air, groundwater, plants, and food. The principal commercial use of sulfate is in the chemical industry. Sulfate concentration in seawater is about 2,700 milligrams per liter (mg/L). It ranges from 3 to 30 mg/L in most freshwater supplies, although much higher concentrations (1000 mg/L) are found in some geographic locations.

**6. Conductivity:** Pure water is not a good conductor of electric current rather’s a good insulator. Increase in ions concentration enhances the electrical conductivity of water. Generally, the amount of dissolved solids in water determines the electrical conductivity. Electrical conductivity (EC) actually measures the ionic process of a solution that enables it to transmit current. According to WHO standards, EC value should not exceeded 400 μS/cm.

**7. Organic\_carbon:** Total Organic Carbon (TOC) in source waters comes from decaying natural organic matter (NOM) as well as synthetic sources. TOC is a measure of the total amount of carbon in organic compounds in pure water. According to the US EPA < 2 mg/L as TOC in treated / drinking water, and < 4 mg/Lit in source water which is use for treatment.

**8. Trihalomethanes:** THMs are chemicals which may be found in water treated with chlorine. The concentration of THMs in drinking water varies according to the level of organic material in the water, the amount of chlorine required to treat the water, and the temperature of the water that is being treated. THM levels up to 80 ppm is considered safe in drinking water.

**9. Turbidity:** The turbidity of water depends on the quantity of solid matter present in the suspended state. It is a measure of light emitting properties of water and the test is used to indicate the quality of waste discharge with respect to colloidal matter. The mean turbidity value obtained for Wondo Genet Campus (0.98 NTU) is lower than the WHO recommended value of 5.00 NTU.

**10. Potability:** Indicates if water is safe for human consumption where 1 means Potable and 0 means Not potable.

**Methods and Algorithms**:

A jupyter notebook instance with Python code was used for processing.

### Following are the list of algorithms that are used in this notebook.

* Logistic regression
* Decision tree
* Random tree
* SVM
* Adaboost

## **Understanding the data:**

Firstly, we need to understand the data that we are working with. As the file format is a csv file, the standard pandas import statement using read\_csv will be used.

# Import the dataset for review as a DataFrame

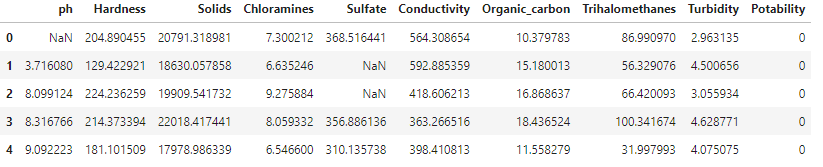
df = pd.read\_csv("../input/water-potability/water\_potability.csv")

# Review the first five observations

df.head()

Having imported the data, the code assigns the variable df with the DataFrame output results from the pandas method.

As with any dataset that you will process, reviewing a sample of records will help you to gain comfort. A DataFrame has a large number of methods associated with it, with the pandas API a great resource to use. Within the API a head method can be used. Output 1.1 shows the first 5 rows of the DataFrame by default. In order to produce a larger number of rows to be displayed a numeric value would be required inside the parenthesis. Two alternatives could be applied to sample the DataFrame with i) sample (df.sample()) selecting random rows from the index, or ii) tail (df.tail()) selecting the last n rows from the index.



Output 1.1 First five record details from the DataFrame

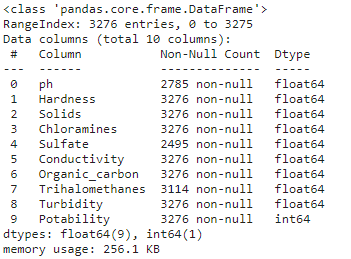
When running any method, the parenthesis is included after the method name allowing the Python interpreter to produce the result.

Displaying the memory of a DataFrame can be a common task, particularly when memory constraints are involved. An example is where the dataset to import is potentially larger than the memory available within the Python session. By using the pandas library a DataFrame is created in-memory so users should understand what memory can be used when performing these processing steps.

# Display information about the DataFrame - contains memory details

df.info(memory\_usage="deep")

The code above can be used as a method to display output 1.2. With the inclusion of the keyword memory\_usage, the Python interpreter is forced to do a deeper search to understand the memory usage that is displayed below. A default option would perform a general search to understand, so if accuracy in your assessment is required then ensure that the keyword phrase from above is applied.



Output 1.2 Provides an overview of the features and details of memory usage

From the results shown in output 1.2, it can show a range of details, from the column names and data types, to also confirming the class of the variable and number of non-null values. We can see that 3,276 rows are shown within the entire table. However, for the column Sulfate, there are only 2,495 non-null values present. Therefore, a number of missing values can be reviewed to understand if there is a pattern for these missing entries with other columns.

**Code generation for prediction:**

Start the water quality analysis task by importing the necessary

Python libraries and the dataset:

import numpy as np

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

%matplotlib inline

import plotly.express as px

import warnings

warnings.filterwarnings('ignore')

df.info

gives the various information properties for potability.

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 3276 entries, 0 to 3275

Data columns (total 10 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 ph 2785 non-null float64

1 Hardness 3276 non-null float64

2 Solids 3276 non-null float64

3 Chloramines 3276 non-null float64

4 Sulfate 2495 non-null float64

5 Conductivity 3276 non-null float64

6 Organic\_carbon 3276 non-null float64

7 Trihalomethanes 3114 non-null float64

8 Turbidity 3276 non-null float64

9 Potability 3276 non-null int64

dtypes: float64(9), int64(1)

memory usage: 256.1 KB

|  | ph | Hardness | Solids | Chloramines | Sulfate | Conductivity | Organic\_carbon | Trihalomethanes | Turbidity | Potability |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| count | 2785.000000 | 3276.000000 | 3276.000000 | 3276.000000 | 2495.000000 | 3276.000000 | 3276.000000 | 3114.000000 | 3276.000000 | 3276.000000 |
| mean | 7.080795 | 196.369496 | 22014.092526 | 7.122277 | 333.775777 | 426.205111 | 14.284970 | 66.396293 | 3.966786 | 0.390110 |
| std | 1.594320 | 32.879761 | 8768.570828 | 1.583085 | 41.416840 | 80.824064 | 3.308162 | 16.175008 | 0.780382 | 0.487849 |
| min | 0.000000 | 47.432000 | 320.942611 | 0.352000 | 129.000000 | 181.483754 | 2.200000 | 0.738000 | 1.450000 | 0.000000 |
| 25% | 6.093092 | 176.850538 | 15666.690297 | 6.127421 | 307.699498 | 365.734414 | 12.065801 | 55.844536 | 3.439711 | 0.000000 |
| 50% | 7.036752 | 196.967627 | 20927.833607 | 7.130299 | 333.073546 | 421.884968 | 14.218338 | 66.622485 | 3.955028 | 0.000000 |
| 75% | 8.062066 | 216.667456 | 27332.762127 | 8.114887 | 359.950170 | 481.792304 | 16.557652 | 77.337473 | 4.500320 | 1.000000 |
| max | 14.000000 | 323.124000 | 61227.196008 | 13.127000 | 481.030642 | 753.342620 | 28.300000 | 124.000000 | 6.739000 | 1.000000 |

*#unstacking to see correaltion*

corr = df.corr()

c1 = corr.abs().unstack()

c1.sort\_values(ascending = False)[12:24:2]

Hardness Sulfate 0.106923

ph Solids 0.089288

Hardness ph 0.082096

Solids Chloramines 0.070148

Hardness Solids 0.046899

ph Organic\_carbon 0.043503

dtype: float64

ax = sns.countplot(x = "Potability",data= df, saturation=0.8)

plt.xticks(ticks=[0, 1], labels = ["Not Potable", "Potable"])

plt.show()



:

x = df.Potability.value\_counts()

labels = [0,1]

print(x)

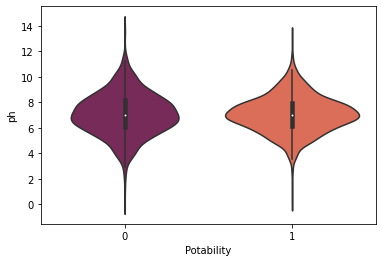
0 1998

1 1278

Name: Potability, dtype: int64

sns.violinplot(x='Potability', y='ph', data=df, palette='rocket')

<AxesSubplot:xlabel='Potability', ylabel='ph'>



## Using Logistic Regression:

Logistic Regression is particularly useful in estimating the vulnerability of aquifers,which are underground layers of water bearing permeable rock from which groundwater can be extracted.

The logistic regression models relate the probability of a contaminant concentration exceeding a threshold concentration to a set of possible influencing variables.

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import confusion\_matrix, accuracy\_score, classification\_report

*# Creating model object*

model\_lg = LogisticRegression(max\_iter=120,random\_state=0, n\_jobs=20)

*# Training Model*

model\_lg.fit(X\_train, y\_train)

LogisticRegression(max\_iter=120, n\_jobs=20, random\_state=0)

*# Making Prediction*

pred\_lg = model\_lg.predict(X\_test)

*# Calculating Accuracy Score*

lg = accuracy\_score(y\_test, pred\_lg)

print(lg)

0.6284658040665434

print(classification\_report(y\_test,pred\_lg))

precision recall f1-score support

0 0.63 1.00 0.77 680

1 0.00 0.00 0.00 402

accuracy 0.63 1082

macro avg 0.31 0.50 0.39 1082

weighted avg 0.39 0.63 0.49 1082

*# confusion Maxtrix*

cm1 = confusion\_matrix(y\_test, pred\_lg)

sns.heatmap(cm1/np.sum(cm1), annot = True, fmt= '0.2%', cmap = 'Reds')

<AxesSubplot:>

