Progressive Education Society's

Modern College of Arts, Science And Commerce, Pune, 411005

T.Y. B.Sc FINAL YEAR 2020-21

Department of Statistics

A PROJECT ON

"STATISTICAL ANALYSIS OF WATER QUALITY USING PREDICTIVE MODELING AND MACHINE LEARNING TECHNIQUES"

Submitted by:

Dnyaneshwar Darekar

Sunny Bhilare

Amar Kadam

Kunal Thosar

Mahesh Kabadi

Guided by:

Prof. Sagar Khandagale.

MODERN COLLEGE SHIVAJINAGAR(AUTONOMOUS)
DEPARTMENT OF STATISTICS
2021-22

Progressive Education Society's
MODERN COLLEGE OF ARTS, SCIENCE, & COMMERCE

Shivajinagar, Pune-411005.

CERTIFICATE

This is to certify that Mr. Dnyaneshwar Darekar

Mr.Sunny Bhilare

Mr.Kunal Thosar

Mr. Amar Kadam

Mr. Mahesh Kabadi

Of class T.Y.B.Sc. Statistics

Has

Satisfactorily completed project on

"Statistical Analysis of Water Quality using Predictive Modeling and Machine Learning Techniques"

for the academic year 2022-2023.

Sagar Khandagale
Preject Guide

Dr. P.G. Dixit
Head of the Department

INDEX

ACKNOWLEDGEMENT

Any project completed successfully offers a great sense of achievement and satisfaction. The project would remain incomplete if the people who made it possible and whose guidance and encouragement go without mention.

We wish to express our cofound gratitude and indebtedness to our estimated guide Sagar Khandagale Sir, Department of Statistics MODERN COLLEGE OF ARTS, SCIENCE& COMMERCE, COLLEGE, PUNE-05 for her expert guidance and scholarly supervision, endless motivation, encouragement throughout the execution of this work.

We are also grateful to the faculties of the department of the statistics MODERN COLLEGE OF ARTS, SCIENCE & COMMERCE, COLLEGE, PUNE-05, those who have directly and indirectly rendered their valuable contribution while preparing our project. We are honoured to express our deep sense of gratitude for principal Dr. R. S. ZUNJARRAO, who has always set an example of hard work and destination to all students.

Finally, we would like to extend a deep appreciation to all those associated with this project for having shared a genuine desire to make a positive contribution to address the challenges associated with every element of this project.

MOTIVATION

Water affect the very crucial role in our life. Drink quality of water is also very important. The polluted water affects environment greatly and leads to many waterborne diseases such as Cholera, Diarrhoea , Typhoid, Hepatitis and various skin diseases. In village we need to check water quality every time when supply the water to villagers.

Motivation behind the project was to check the water quality. Main motive of this project is to classifies the water which is drinkable or not. So that we provide quality of water to the peoples. The treated water is analysed in the lab or any small place before it is supply, for that parameters like ph, Hardness, Solids, Chloramines, Sulfate, Conductivit, Organic_carbon, Trihalomethanes, Turbidity which have standard set of values . For these values we are testing a hypothesis for all parameters so that we can conclude the given water sample(population) is potable or not and lastly we are doing model fitting in which we are searching for best fitting model with which we can estimate(classify) potability for given data.

INTRODUCTION

We can't decide water quality by our naked eyes. To check the quality of water we use an automated tool using machine learning based classification techniques is proposed in the paper. Different classifiers are used to check whether the given sample of water is drinkable or not and the results of those classifiers are decide for water quality. There are lot of classification model like Naïve bayes, Logistic regression, KNN, SVM etc. However, experimental results indicate that ensemble classifiers are the best classification to check the water quality. This project uses data provided from Kaggle. This data contains components which are in the water.

This project follows five stages. The five stages adopted for thisproject are –

1. Problem Definition (Project Overview, Project statement and Metrics)
2. Data Collection
3. Data cleaning, exploring and pre-processing
4. Modeling
5. Evaluating

Terminology

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 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.

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.

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 un-wanted taste and diluted color in 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. Desirable limit for TDS is 500 mg/l and maximum limit is 1000 mg/l which prescribed for drinking purpose.

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.

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.

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.

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 US EPA < 2 mg/L as TOC in treated / drinking water, and < 4 mg/Lit in source water which is use for treatment.

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.

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.

Potability:

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

Hypothesis Testing-

Hypothesis testing is one of the most important concepts in Statistics which is heavily used by Statisticians, Machine Learning Engineers, and Data Scientists. In hypothesis testing, Statistical tests are used to check whether the null hypothesis is rejected or not rejected. These Statistical tests assume a null hypothesis of no relationship or no difference between groups.

Parametric and Non-Parametric Test-

Parametric tests are those tests for which we have prior knowledge of the population distribution (i.e. normal), or if not then we can easily approximate it to a normal distribution which is possible with the help of the Central Limit Theorem.

In **Non-Parametric tests**, we don't make any assumption about the parameters for the given population or the population we are studying. In fact, these tests don't depend on the population.

As our project aim is to check whether water is drinkable or not i.e The given water is potable or not potable. So for testing this claim we have to perform the hypothesis testing for each parameter **(ph, Hardness, Solids, Chloramines, Sulfate, Conductivit, Organic_carbon, Trihalomethanes,Turbidity)**.

ph	**6.5<PH<8.5**
Hardness	
Solids	
Chloramines	
Sulfate	
Conductivity	
Organic_carbon	
Trihalomethanes	
Turbidity	

CLAIM: If all the parameters value are in the specified range then we can conclude that the water is potable, otherwise it is not potable.

So for checking this claim we perform the hypothesis testing for a given sample data. The first step to proceed by police testing we want to check whether the given sample is coming from normal population or not.

Let's check the normality of each parameter by Shapiro test.

Import Data set:

```
library(readxl)

waterqualitydata <-
read_excel("C:/Users/ASUS/Desktop/waterqualitydata.xlsx")

View(waterqualitydata)
```

Shapiro-Wilk normality test (l.o.s.=5%)

To test: H_0 = PH is normally distributed.

H_1 = PH is not normally distributed.

```
> shapiro.test(waterqualitydata$ph)


Shapiro-Wilk normality test


data:  waterqualitydata$ph

W = 0.99587, p-value = 5.727e-07



> shapiro.test(waterqualitydata$Hardness)


Shapiro-Wilk normality test


data:  waterqualitydata$Hardness

W = 0.99597, p-value = 9.61e-08

> shapiro.test(waterqualitydata$Solids)


Shapiro-Wilk normality test
```

```
data:  waterqualitydata$Solids
W = 0.97773, p-value < 2.2e-16
> shapiro.test(waterqualitydata$Chloramines)

Shapiro-Wilk normality test

data:  waterqualitydata$Chloramines
W = 0.99677, p-value = 1.818e-06

> shapiro.test(waterqualitydata$Sulfate)

Shapiro-Wilk normality test

data:  waterqualitydata$Sulfate
W = 0.99602, p-value = 3.467e-06
> shapiro.test(waterqualitydata$Conductivity)

Shapiro-Wilk normality test

data:  waterqualitydata$Conductivity
W = 0.99297, p-value = 1.494e-11
> shapiro.test(waterqualitydata$Organic_carbon)

Shapiro-Wilk normality test

data:  waterqualitydata$Organic_carbon
```

```
W = 0.99952, p-value = 0.6251

> shapiro.test(waterqualitydata$Trihalomethanes)


Shapiro-Wilk normality test


data:  waterqualitydata$Trihalomethanes

W = 0.99886, p-value = 0.03479

> shapiro.test(waterqualitydata$Turbidity)


Shapiro-Wilk normality test


data:  waterqualitydata$Turbidity

W = 0.9997, p-value = 0.9336
```

Conclusion:

As p-value for each test is less than 0.05 so we reject null hypothesis at 5% l.o.s.

By seeing the output of shapiro test we can easily conclude that the data does not follow normal distribution. So we should go with corresponding non parametric test.

There exist a suitable non parametric test for checking median(median is measure of central tendency for non parametric test) value which is known as one sample Wilcoxon signed rank test.

The following is the information about the test.

Wilcoxon's Signed Rank Test:

It is one of the non-parametric tests used to test the location of a population based on a sample of data or to compare the locations of two populations using two samples. The sign test for location utilizes only the signs of difference of observations from hypothesized median (or the difference of observations in the pairs) without considering the magnitude of the difference. If the information regarding magnitude is available then a test procedure that takes into account the size and the relative magnitude of the differences as well, is expected to give a better performance. Wilcoxon’s signed rank test is based on this consideration. However, the better performance is obtained at the cost of additional assumption of symmetry of the population about true median.

Testing Problem:

Suppose $X_1, \ldots$, Xn is a random sample of size n from the distribution of random variable X. Let $F_x(.)$ be the distribution function and M be the median of X.it is required to test the hypothesis.

$H_0 : M = M_0$ against one of the alternatives,

1) $H_1 : M > M_0$
2) $H_1 : M < M_0$
3) $H_1 : M \neq M_0$

Assumptions:

1. $F_x(.)$ is continuous
2. $F_x(.)$ is symmetric about M.

Test Statistic:

Let T^+ = sum of positive ranks

T^- = sum of negative ranks.

Note that, T^+ and T^- both are non-negative numbers and

$$T^+ + T^- = \sum_{i=1}^{n} \frac{n(n+1)}{2}$$

Under H_0, the distributions of T^+ and T^- are identical and each distribution is symmetric about the common mean n(n+1)/4. So, any one of the T^+ or T^- can be used as the test statistic.

R code:

```
> a=wilcox.test(waterqualitydata$ph,mu=7.04,alternative ="greater")
> a

Wilcoxon signed rank test with continuity correction

data:  waterqualitydata$ph
V = 1985957, p-value =  4.407e-08

alternative hypothesis: true location is less than 7.04

> a=wilcox.test(waterqualitydata$Hardness,mu=196.98,alternative ="less")
> a

Wilcoxon signed rank test with continuity correction

data:  waterqualitydata$Hardness
V = 2642329, p-value = 1.2e-07
alternative hypothesis: true location is less than 196.98

> a=wilcox.test(waterqualitydata$Solids,mu=20927.83,alternative ="less")
> a

Wilcoxon signed rank test with continuity correction

data:  waterqualitydata$Solids
V = 2894362, p-value = 3.2e-04
```

```
alternative hypothesis: true location is less than 20927.83

> a=wilcox.test(waterqualitydata$Chloramines,mu=7.13,alternative ="less")
> a

Wilcoxon signed rank test with continuity correction

data:  waterqualitydata$Chloramines
V = 2668379, p-value = 1.062e-05
alternative hypothesis: true location is less than 7.13

> a=wilcox.test(waterqualitydata$Sulfate,mu=333.07,alternative ="less")
> a

Wilcoxon signed rank test with continuity correction

data:  waterqualitydata$Sulfate
V = 1580798, p-value = 4.52e-2
alternative hypothesis: true location is less than 333.07

> a=wilcox.test(waterqualitydata$Conductivity,mu=421.88,alternative ="less")
> a

Wilcoxon signed rank test with continuity correction
```

```
data:  waterqualitydata$Conductivity
V = 2773451, p-value = 2.342e-03

alternative hypothesis: true location is less than 421.88

> a=wilcox.test(waterqualitydata$Organic_carbon,mu=14.22,alternative ="less")
> a

Wilcoxon signed rank test with continuity correction

data:  waterqualitydata$Organic_carbon
V = 2737315, p-value = 3.122e-09

alternative hypothesis: true location is less than 14.22

> a=wilcox.test(waterqualitydata$Trihalomethanes,mu=66.62,alternative ="less")
> a

Wilcoxon signed rank test with continuity correction

data:  waterqualitydata$Trihalomethanes
V = 2407313, p-value = 4.122e-11

alternative hypothesis: true location is less than 66.62
> a=wilcox.test(waterqualitydata$Turbidity,mu=3.96,alternative ="less")
> a
```

```
Wilcoxon signed rank test with continuity correction


data:  waterqualitydata$Turbidity

V = 2712644, p-value = 3.12e-04


alternative hypothesis: true location is less than 3.96
```

As all the null hypothesis is rejected for the given parameters, we can conclude that all the parameters are in suitable range. But in future the decision may or may not be same as it is in present because it depends on the given sample.

Note: Suppose in the future if we get similar data and we want to Check the given water is potable or not we can use appropriate machine learning model for checking purpose.

Benefits of using model over the hypothesis:

The hypothesis is possible if and only the given sample is considerably large. sometimes it is very costly to get the large sample but if you have given only one data point we can't use the hypothesis but we can use the model to get idea about the census.

Scope of using the machine learning models in day to day life

We can easily see that in summer season some villages face the water problem. Sometimes water provided to them maybe collected from river or from lake are from some well which is not tested chemically whether it is potable or not because by the naked eyes we can't figure out the water as potable or not.

So if we have provided the parameters value it will be very difficult for human being to check each value in the parameter space and give conclusion about the sample. Sometimes it will reject the sample even it satisfy all the require conditions. So to increase the efficiency of work we use the machine learning models.

MACHING LEARNING:

What is Machine Learning?

Machine learning (ML) is basically the study of computer algorithms that can improve automatically through experience and by the use of past data. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as training data, in order to make decisions and test its accuracy with the help of test data. Machine learning algorithms are used in a wide variety of applications, such as in medicine, email filtering, speech recognition, computer vision, etc.

Nowadays the demand of statistics in machine learning is increasing day by day. In models, Statistical methods are required in the preparation of train data and test data and also to check the accuracy of the models.

This includes:

- Outlier detection.
- Missing value imputation.
- Data sampling.
- Data scaling.
- Variable encoding.

This all can be done in machine learning by applying the proper statistical tools.

Why we use it?

The response variable of our data was in the form of classification type. So we classify our data in two groups namely potable water or non potable water as like a binary variable.

Potable water=1

Non potable water=0

There are also some classification models that are used in machine learning.

Example of those models are

1.Logistic Regression.

2.K-Nearest Neighbor

3.Support Vector Machines

4.Kernel SVM

5.Naive Bayes

6.Decision Tree Classification

7.Random Forest Classification

8.ANN

9.CNN

we want to develop a model that can predict the values of potability. Our focus is on both accuracy of the predictions and interpretability of the model.

Therefore we have choose the models that suits our data best. We will evaluate three different models covering the complexity spectrum.

1.Logistic Regession.

2.K-Nearest Neighbors.

3.Decision tree

To head-start the ML process, the cleaning of data is must.

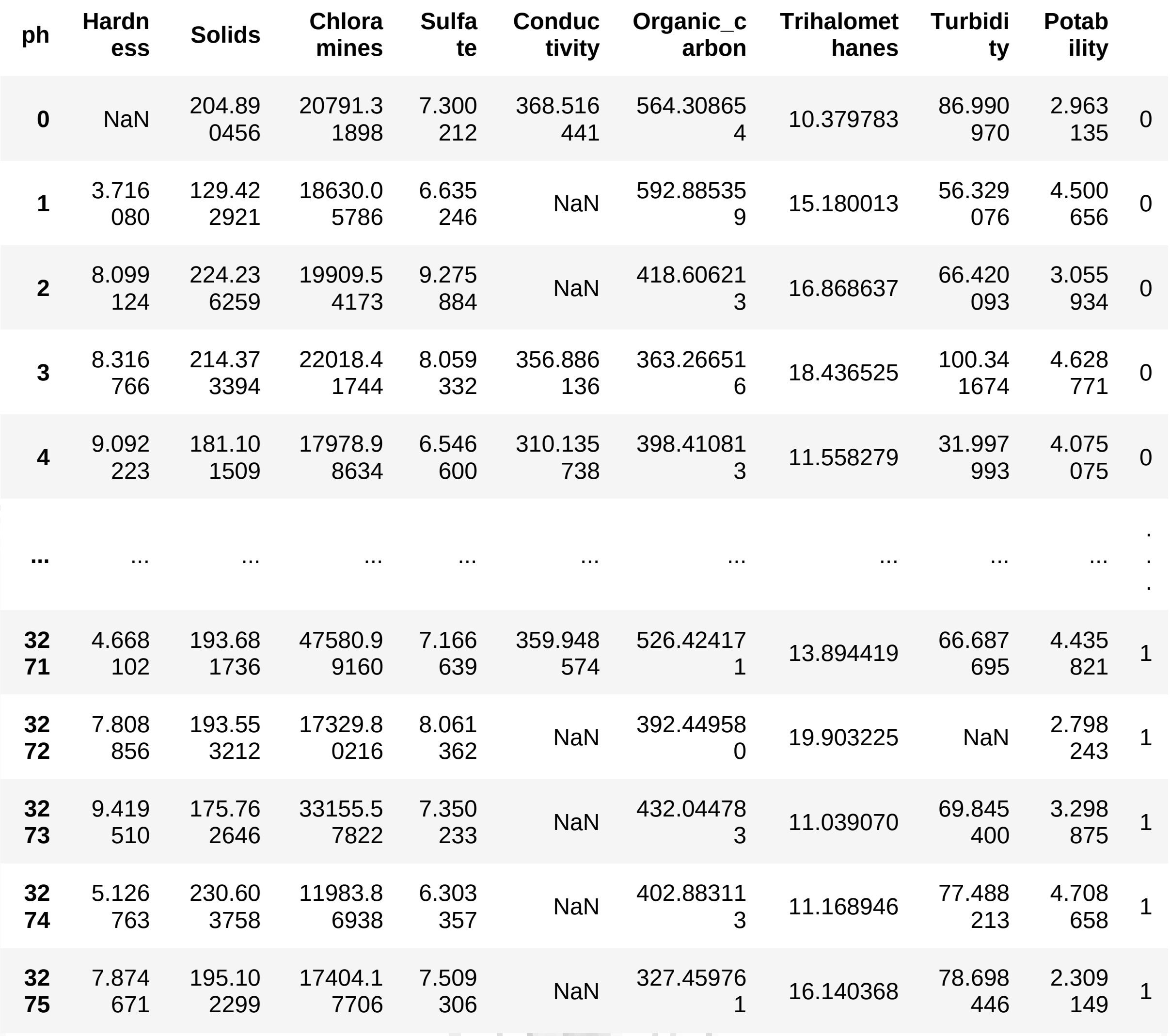
Why data cleaning is important?

To reduce the errors and to increase the efficiency of model we need to clean our data.

Lets clean our data set using python:

Codes for cleaning data :

```
import pandas as pd   #to import and analyse data
import numpy as np    #to work with array -mathematical operations

import matplotlib.pyplot as plt    #data visualization and graphical plotting

import seaborn as sns;sns.set()     #data visualisation and exploratory data analysis

import math #mathematical calculations

data= pd.read_csv(r'C:\Users\ASUS\Desktop\Pandas\water_potability data for project.csv')

                                                   #importing data in csv format data
```



ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability	
0	NaN	204.890456	20791.31898	7.300212	368.516441	564.308654	10.379783	86.990970	2.963135	0
1	3.716080	129.422921	18630.05786	6.635246	NaN	592.885359	15.180013	56.329076	4.500656	0
2	8.099124	224.236259	19909.54173	9.275884	NaN	418.606213	16.868637	66.420093	3.055934	0
3	8.316766	214.373394	22018.41744	8.059332	356.886136	363.266516	18.436525	100.341674	4.628771	0
4	9.092223	181.101509	17978.98634	6.546600	310.135738	398.410813	11.558279	31.997993	4.075075	0
...	...	...	...	...	...	...	...	...	...	...
3271	4.668102	193.681736	47580.99160	7.166639	359.948574	526.424171	13.894419	66.687695	4.435821	1
3272	7.808856	193.553212	17329.80216	8.061362	NaN	392.449580	19.903225	NaN	2.798243	1
3273	9.419510	175.762646	33155.57822	7.350233	NaN	432.044783	11.039070	69.845400	3.298875	1
3274	5.126763	230.603758	11983.86938	6.303357	NaN	402.883113	11.168946	77.488213	4.708658	1
3275	7.874671	195.102299	17404.17706	7.509306	NaN	327.459761	16.140368	78.698446	2.309149	1

3276 rows × 10 columns

data.shape # rows,coloumb

```
(3276, 10)
```

```
# Data Cleaning
```

```
data.info()
<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)


data.isnull().sum()
ph                 491
Hardness             0
Solids               0
Chloramines          0
Sulfate            781
Conductivity         0
Organic_carbon       0
Trihalomethanes    162
Turbidity            0
Potability           0
dtype: int64


data.fillna(data.mean(),inplace=True)    # Fill null v
alues by it's average value.
Data
```

ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability	
0	7.080795	204.890456	20791.31898	7.300212	368.516441	564.308654	10.379783	86.990970	2.963135	0
1	3.716080	129.422921	18630.05786	6.635246	333.775777	592.885359	15.180013	56.329076	4.500656	0
2	8.099124	224.236259	19909.54173	9.275884	333.775777	418.606213	16.868637	66.420093	3.055934	0
3	8.316766	214.373394	22018.41744	8.059332	356.886136	363.266516	18.436525	100.341674	4.628771	0
4	9.092223	181.101509	17978.98634	6.546600	310.135738	398.410813	11.558279	31.997993	4.075075	0
...	...	...	...	...	...	...	...	...	...	...

ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability	
3271	4.668102	193.681736	47580.99160	7.166639	359.948574	526.424171	13.894419	66.687695	4.435821	1
3272	7.808856	193.553212	17329.80216	8.061362	333.775777	392.449580	19.903225	66.396293	2.798243	1
3273	9.419510	175.762646	33155.57822	7.350233	333.775777	432.044783	11.039070	69.845400	3.298875	1
3274	5.126763	230.603758	11983.86938	6.303357	333.775777	402.883113	11.168946	77.488213	4.708658	1
3275	7.874671	195.102299	17404.17706	7.509306	333.775777	327.459761	16.140368	78.698446	2.309149	1

3276 rows × 10 columns

```
data.isnull().sum()   #checking null values
ph                   0
Hardness             0
Solids               0
Chloramines          0
Sulfate              0
Conductivity         0
Organic_carbon       0
Trihalomethanes      0
Turbidity            0
Potability           0
dtype: int64
```

As data cleaning is done so we can move further.

To use the machine learning model the basic assumptions is that there should no multicollinearity between the regressor. So in our data type let X1, X2, X3, X4, X5, X6, X7, X8, X9 be ph, Hardness, Solids, Chloramines, Sulfate, Conductivit, Organic_carbon, Trihalomethanes, Turbidity respectively. These are the regressor in our data which affects the value of the response variable. So to check the multicollinearity between the regressors we use the Heat map as statistical tool.

HEAT MAP

What is heat map?

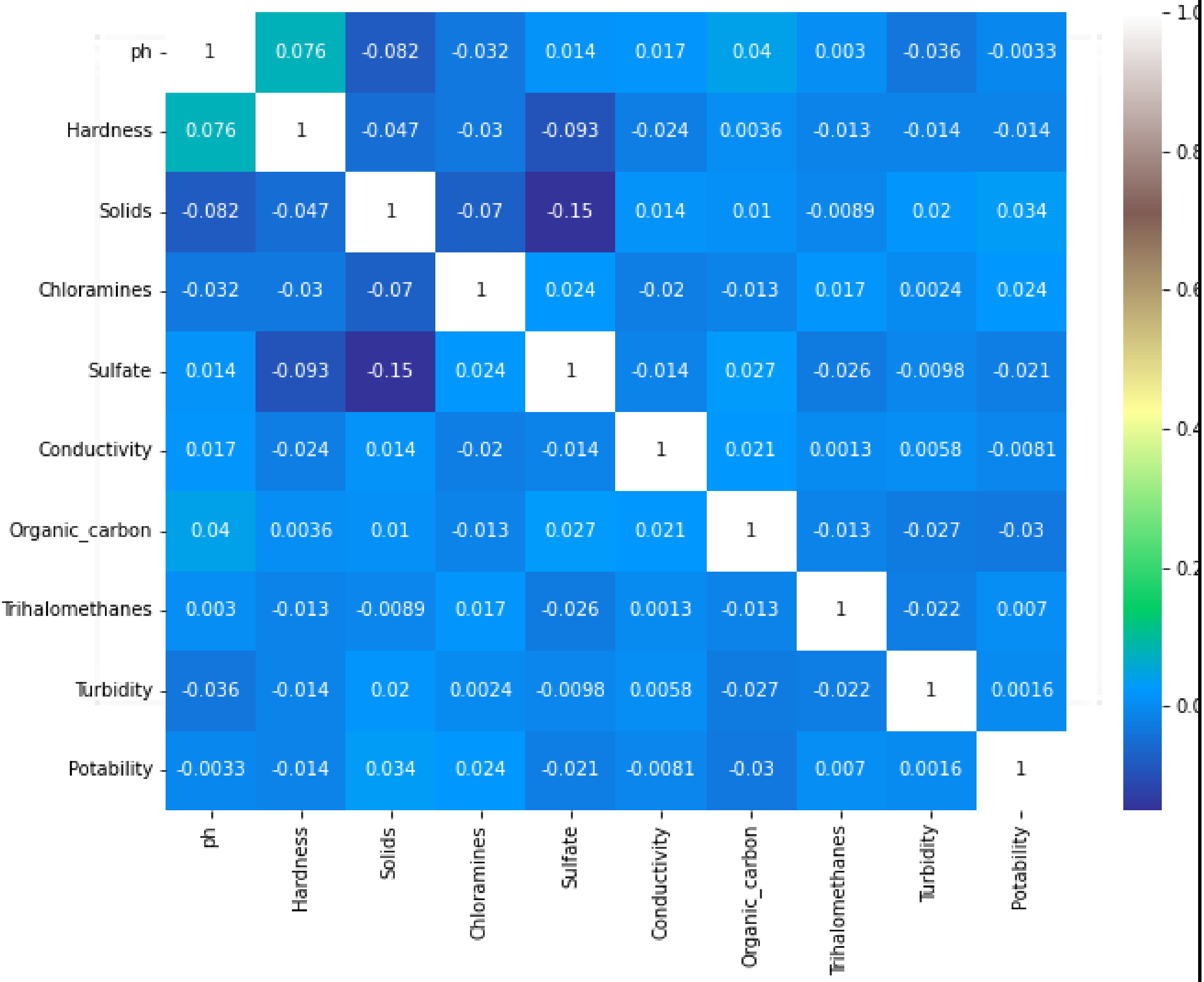
A heatmap is basically the representation of two dimensional information (data) with the help of colours . It gives warm-to-cool colour spectrum to show which parts of a data has the most attention. We use Heatmap as a correlation matrix .In heatmap correlation matrix, both the axis has same variables and we check the correlation between them by using it .The dark colour represent the positive correlation and the medium light colour gives no correlation between the variable.

As it gives visual as well as numerical value to check the correlation . The values in the cell indicate the strength of the relationship, with positive values indicating a positive relationship and negative values indicating a negative relationship. In addition, correlation plots can be used to identify outliers and to detect linear and nonlinear relationships. The color-coding of the cells makes it easy to identify relationships between variables at a glance.

Check the multicollinearity between the regressors:

By using python we plot the heatmap for our data. The respective commands are as follows.

```
#correlation using heatmap
sns.heatmap(data.corr(),annot=True,cmap='terrain')
fig=plt.gcf()
fig.set_size_inches(11,8)
plt.show()
```



Conclusion of heat map:

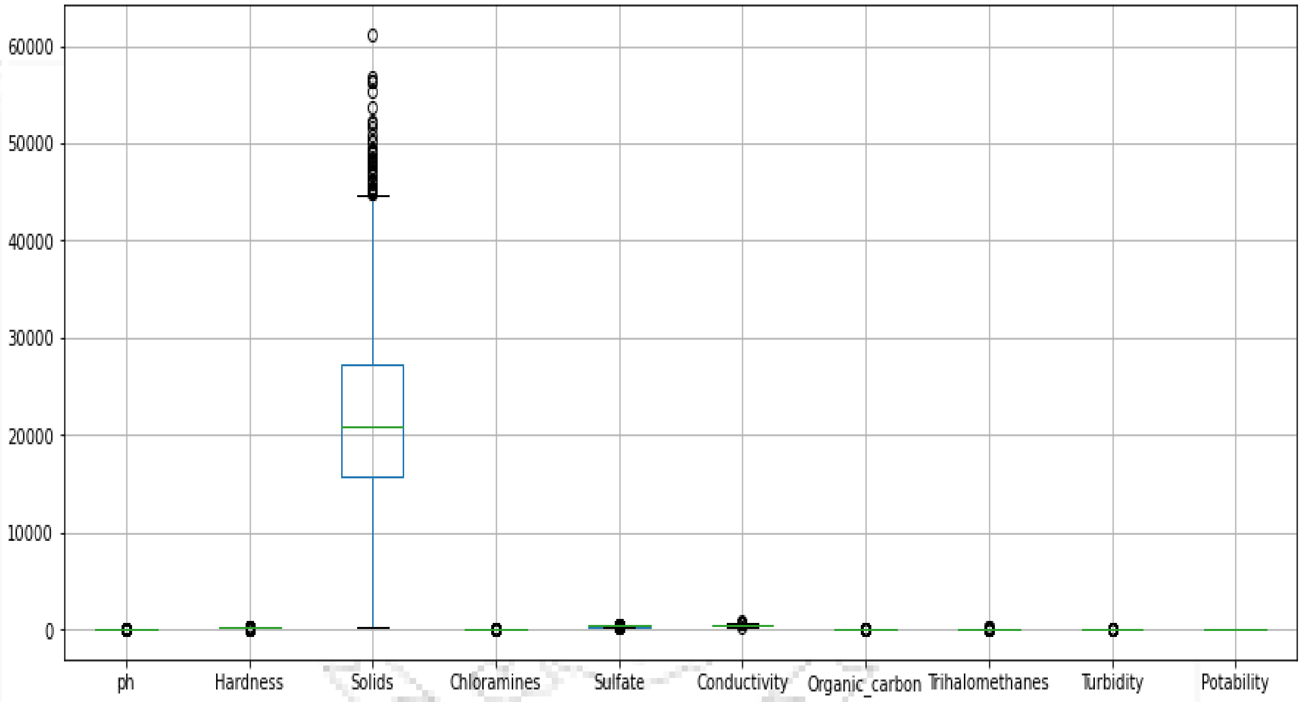
As the correlation coefficient are neglible, we can conclude that the parameters are uncorrelated.

# Checking outliers using boxplot:

ata.boxplot(figsize=(15,6))

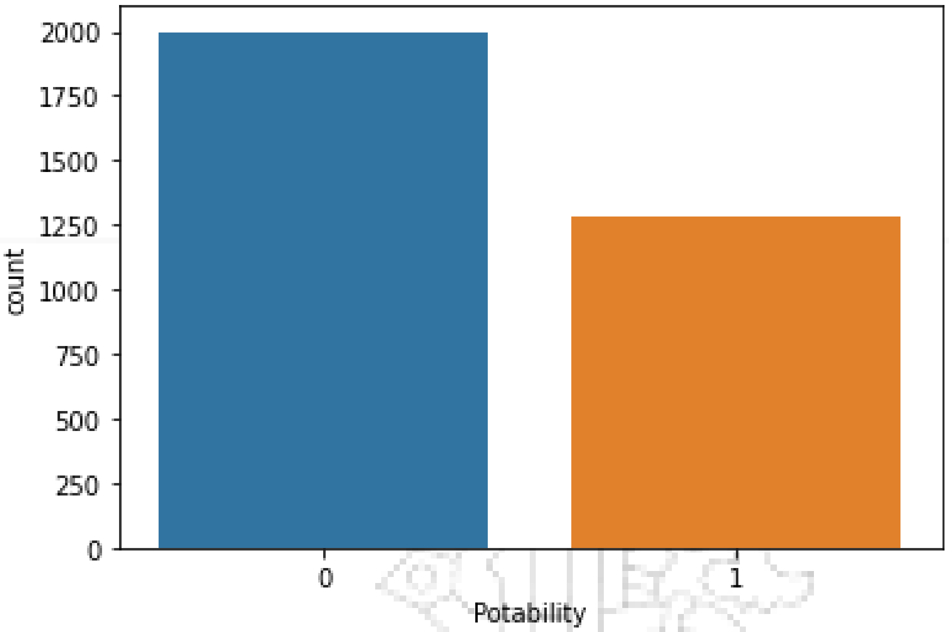
plt.show()

BOX PLOT



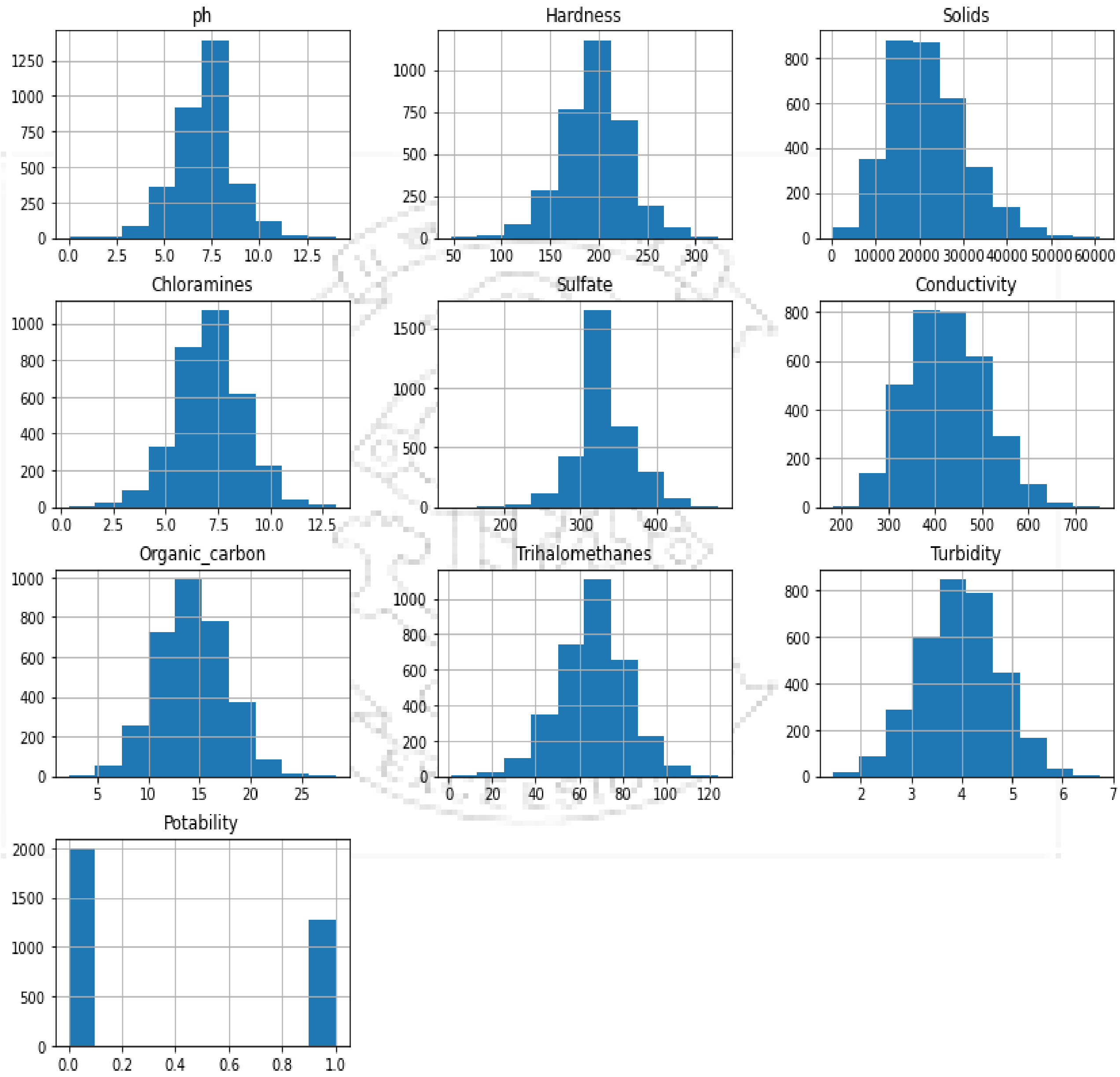
# We are not removing outliers because they may decide the quality of water.

```
#countplot for potability
sns.countplot(data['Potability'])
plt.show()
```



#graphical representation of parameters using histogram.

```
data.hist(figsize=(14,12))

plt.show()
```

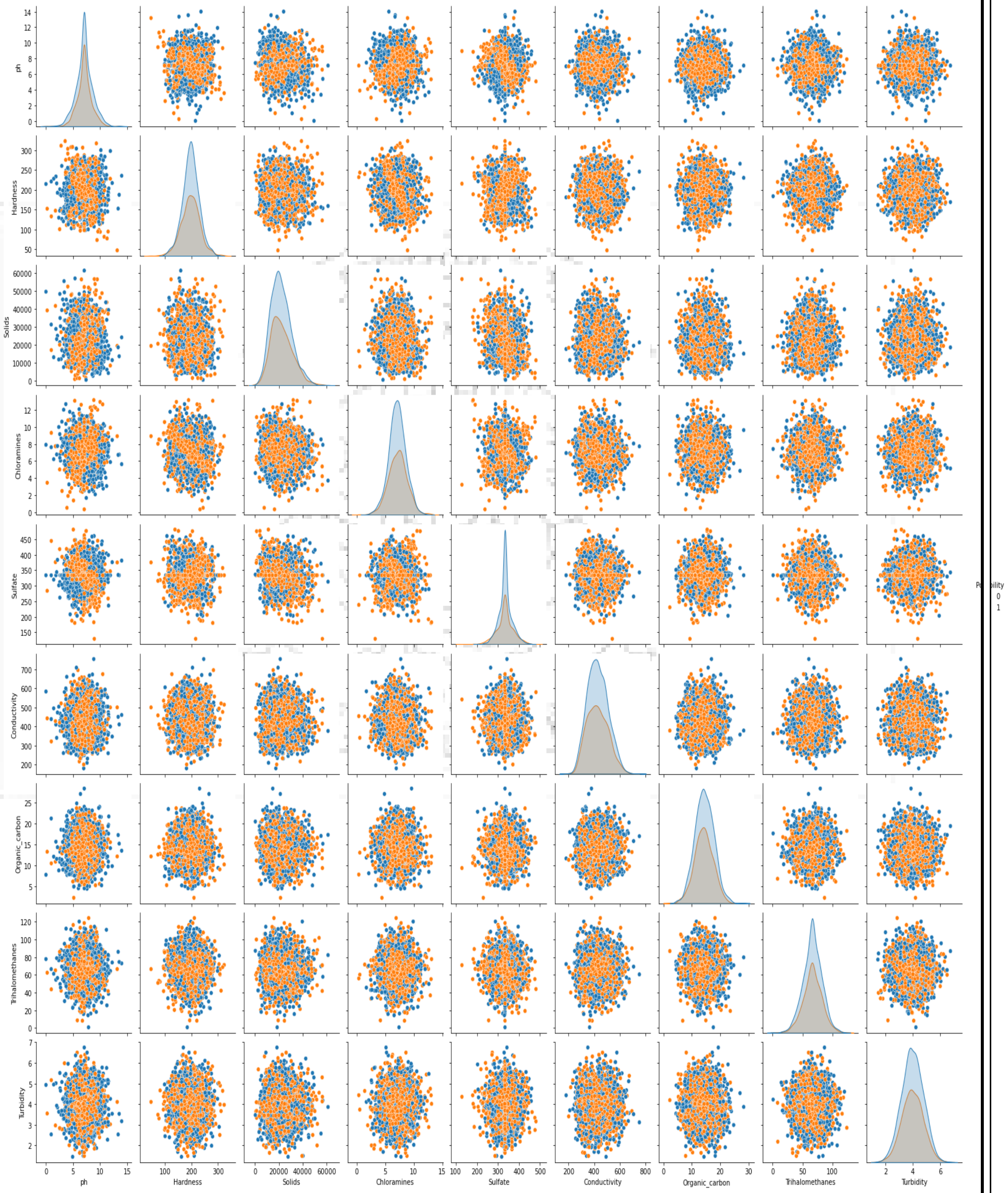


Conclusion:

From the above graph, we concluded that our data don’t follow normal distribution. By using Shapiro-Wilk test also it was confirmed.

#graphical representation of relationship between the parameters using pairplots.

```
sns.pairplot(data,hue='Potability')

plt.show()
```



Decision Tree.

What is decision tree?

Decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource. It is Supervised Machine learning algorithm which uses set of rules to make decisions. It is one of the classification algorithms which uses rule-based approach.

For example: Planning the next vacation which depends on various factors such as time, no. of members, budget.

It can perform both classification and regression tasks so referred as CART algorithm (Classification and Regression Tree).

Intuition: Need of use of dataset features to create YES/NO type questions (In our case water potability)

until we isolate all data points belonging to each class.

Model characteristics:

1)Fewer the splits more the accuracy.

2)Algorithm assigns only one class to each leaf node.

3)It picks best split to minimize loss function on basis of purity – "GINI Impurity"

$$G=\sum_{k=1}^{c} \mathrm{P}(1-\mathrm{P})$$

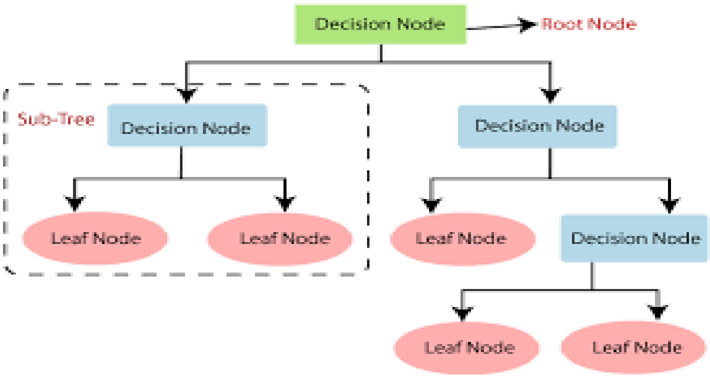
4)Uses greedy approach

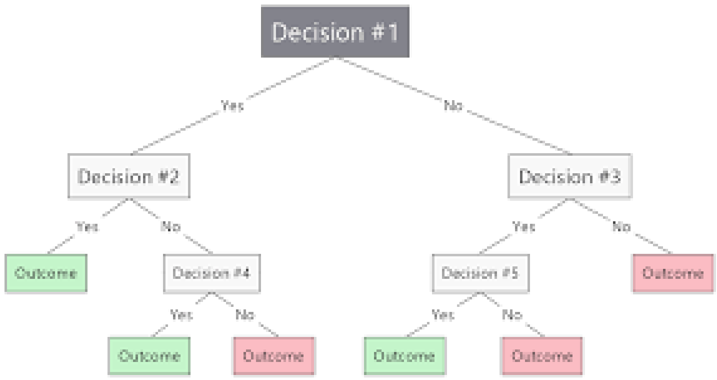
5)It can be linearized into decision rules

6)It should be paralled by a probability model as a choice model

7)Descriptive means for calculating conditional probabilities.

8)Categorical variable decision tree.





Python – Code

#Importingm dataset:

```
data= pd.read_csv(r'C:\Users\ASUS\Desktop\Pandas\water_potability data for project.csv')
```

#Split data set into train and test set:

```
X=data.drop('Potability',axis=1)      ( # Inpute Variable)

X

Y=data['Potability']        (# Targer Varrable)

Y

from sklearn.model_selection import train_test_split

X_train , X_test , Y_train , Y_test =
train_test_split(X,Y,test_size=0.2,shuffle=True,random_state=0)
```

X_train

#Model fitting decision tree:

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy_score,confusion_matrix,precision_score

data=DecisionTreeClassifier(criterion= 'gini', min_samples_split= 10, splitter= 'best')

data.fit(X_train,Y_train)

#Prediction for test dataset:

```
#Prediction for test dataset
prediction=data.predict(X_test)
prediction
array([0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0,
       0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0,
       0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0,
       0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0,
       0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1,
       1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0,
       1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1,
       1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0,
       1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1,
       0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0,
       0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0,
       0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1,
       0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1,
       0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0,
       1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1,
       1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1,
       1, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0,
       0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0,
       1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0,
       0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1,
       1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0,
       1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0,
       1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0], dtype=int64)
```

#Accuracy:

accuracy_score(prediction,Y_test)

*print('accuracy_score:',accuracy_score(prediction,Y_test)*100,'%')*

print("feature importances:\n{}".format(data.feature_importances_))

#Confusion matrix:

confusion_matrix(prediction,Y_test)

#Model Evaluation:

# Confusion matrix

```
array([[281, 134],
       [131, 110]], dtype=int64)
```

#Accuracy:

```
accuracy_score: 59.60365853658537 %
```

#Conclusion:

Accuracy rate for fitting model given by Dicesion tree is 59.60% of our data.

Advantage :

1. Simple to understand and to interpret.
2. It can handle both numerical as well as categorical data.

Disadvantages:

1)Unstable: Change sensitive

2)Relatively inaccurate

3)Bias in favour of attributes with more level

4) Calculations can get very complex

K-Nearest Neighbor classifier (k-NN):

What is K-NN?

K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique. It was 1st used for classification task by Fix and Hodges in 1951.K-NN is a **non-parametric algorithm**, which means it does not make any assumption on underlying data. It maps an Input to an Output based on example of Input-Output pairs. i.e it stores all the available data and classifies a new data point based on the similarity.

- Euclidean distance-

$$d(x,\ y)\ =\ \sqrt{\sum_{i=1}^{n}(y_i - x_i)^2}$$

-
- In K NN classification, the output is the class membership. An object is classified by majority votes of its neighbors, with the object being assigned to the class most common among its k nearest (k is positive integer, typically small). If k=1, then the object is simply assign to the class of that single nearest neighbor.
- In k-NN regression, the output is the property value for the object. This value is the average of the values of its k-nearest neighbors.
- k-NN is a type of instance based learning where the function is only approximated locally and all computation is deferred until classification. The k-NN algorithm is among the simplest of all machine learning algorithms.
- Both for classification and regression, a useful technique can be to assign weight to the neighbors, so that the nearer neighbors contribute more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight 1/d, where d is the distance to the neighbor.

- The particularity of the k-NN algorithm is that it is sensitive to the local structure of the data. The algorithm is not be confused with k-means, another popular machine learning technique.
- One of the simple procedures that can be used for classification is the Nearest Neighbor (NN) rule. It classifies as sample based on the category of its nearest neighbor. When large sample are involved it can be shown that this rule has probability of error which is less than twice the optimum error. Hence there is less than twice the probability of error compare to any decision rule. The nearest neighbor based classifier used some or all the patterns available in the training set to classify a test pattern. This classifier essentially involves finding the similarity between the patterns in the training set.
- Among the various methods of supervised statistical pattern recognization, Nearest Neighbor rule achieves consistently high performance, without the priori assumption about the distribution from which the training examples are drawn. Sample is classified by calculating the distance to the nearest training case K-NN classifier extends this idea by taking the k-nearest point and assigning the class pf majority. It is common to select k small and odd to breakties. It can be also given by square root of sample. Larger k-value helps reduce the effects of noisy point within training data set. And the choice of k is often perform through cross validation.

Python – Code

#Importingm dataset:

```
data= pd.read_csv(r'C:\Users\ASUS\Desktop\Pandas\water_potability data for
project.csv')
```

#Split data set into train and test set:

```
X=data.drop('Potability',axis=1)      ( # Inpute Variable)

X

Y=data['Potability']                  (# Targer Varrable)

Y

from sklearn.model_selection import train_test_split


X_train , X_test , Y_train , Y_test =
train_test_split(X,Y,test_size=0.2,shuffle=True,random_state=0)


X_train
```

#Model fitting and prediction for KNN:

```
from sklearn.neighbors import KNeighborsClassifier


knn=KNeighborsClassifier(metric='euclidean',n_neighbors=22)

knn.fit(X_train,Y_train)
```

#Prediction for test dataset:

```
prediction_knn=knn.predict(X_test)

prediction_knn
```

```
array([1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
       0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0,
       0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1,
       0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], dtype=int64)
```

#Accuracy:

accuracy_knn=accuracy_score(Y_test,prediction_knn)*100

print('accuracy_score:',accuracy_knn,'%')

#Confusion matrix:

confusion_matrix(prediction,Y_test)

#Model Evaluation:

# Confusion matrix

```
array([[277, 131],
       [135, 113]], dtype=int64)
```

#Accuracy:

```
accuracy_score: 60.97560975609756 %
```

#Conclusion:

Accuracy rate for fitting model given by Dicesion tree is 60.98% of our data.

Advantages:

1. It is simple to implement
2. It is robust to noisy training data
3. It can be more effective if the training data is large

Disadvantages:

1. It is simple to implement
2. It is robust to noisy training data
3. It can be more effective if the training data is large

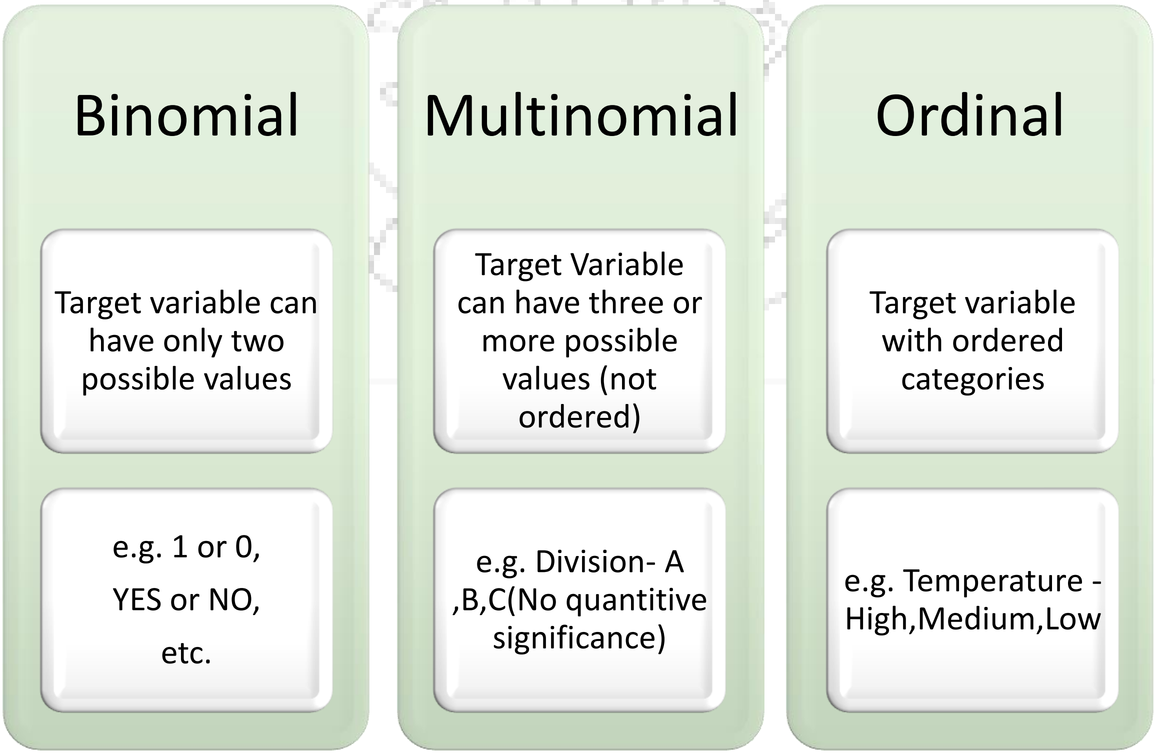
Logistic Regression Model

What is logistic model?

It is a statistical method which is used to Predict a "binary output such as Yes or No (in our case 1 or o). Logistic regression model predicts dependent variable of data using regressors which are independent.

It is basically a supervised classification algorithm use in classification problems. As in linear regression, it is assume that the data follows linear function similarly logistic model builds a regression model to predict the probability that given data entry belongs to Category numbered as "1" OR “0”

Types of the Logistic Regression:



Binomial

Target variable can have only two possible values

e.g. 1 or 0,

YES or NO,

etc.

Multinomial

Target Variable can have three or more possible values (not ordered)

e.g. Division- A ,B,C(No quantitive significance)

Ordinal

Target variable with ordered categories

e.g. Temperature - High,Medium,Low

Assumptions:

1. Absence of Multicollinearity- one of the most important assumptions.
2. The dependent variable must be dichotomous.

Why this model?

As in our data, response variable is in the form of binary type and also there is no collinearity between the regressors (Using heatmap we can observed) , hence we have use this model for testing quality of water i.e whether it is potable or not.

Model of Logistic regression:

1. Y= E(YIx) + ε
2. Y = Π(x) + ε

Where, ε is Bernoulli random variable with

a. E(ε)=0

b. var(ε) = π(x)(1-π(x))

$$\pi(x) = \frac{e^{\beta0+\beta1X1+\beta2X2+\beta3X3+\beta4X4+\beta5X5+\beta6X6+\beta7X7+\beta8X8+\beta9X9}}{1+e^{\beta0+\beta1X1+\beta2X2+\beta3X3+\beta4X4+\beta5X5+\beta6X6+\beta7X7+\beta8X8+\beta9X9}}$$

Where , β_1 ,β_2, β_3, $\beta_{4,}$ β_5, β_6, β_7, $\beta_{8,}$ β_9 are regression coefficients and variables are

POTABILITY	Y
PH	X_1
Hardness	X_2
Solids	X_3

Chloramines	X_4
Sulfate	X_5
Conductivity	X_6
Organic_carbon	X_7
Trihalomethanes	X_8
Turbidity	X_9

Logistic model considers probability using which we are going to allocate new observation to specify class.For this purpose the threshold probability is decided and by default it is consider as P=0.5

Python – Code

#Importingm dataset:

```
data= pd.read_csv(r'C:\Users\ASUS\Desktop\Pandas\water_potability data for
project.csv')
```

#Split data set into train and test set:

```
X=data.drop('Potability',axis=1)      ( # Inpute Variable)

X

Y=data['Potability']                  (# Targer Varrable)

Y

from sklearn.model_selection import train_test_split


X_train , X_test , Y_train , Y_test =
train_test_split(X,Y,test_size=0.2,shuffle=True,random_state=0)
```

```
X_train
```

#Model fitting and prediction for Logistic Regression Model:

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(X_train, Y_train)
```

#Prediction for test dataset:

```
prediction=data.predict(X_test)
```

#Accuracy:

```
test_acc = accuracy_score(Y_test,prediction)
print("The Accuracy for Test Set is {}".format(test_acc*100),'%')
```

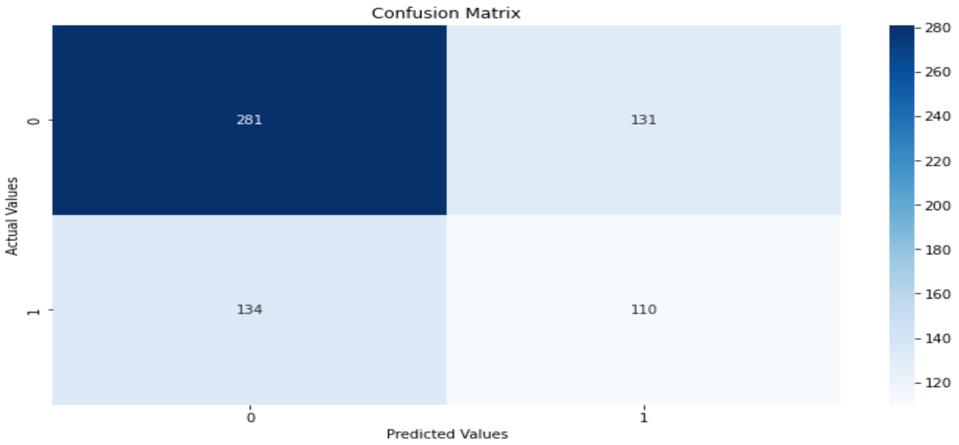
#Confusion matrix:

```
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
print(classification_report(Y_test,prediction))


cm=confusion_matrix(Y_test,prediction)
plt.figure(figsize=(12,6))
plt.title("Confusion Matrix")
sns.heatmap(cm, annot=True,fmt='d', cmap='Blues')
plt.ylabel("Actual Values")
plt.xlabel("Predicted Values")
plt.savefig('confusion_matrix.png')
```

#Model Evaluation:

```
# Confusion matrix
```

```
precision    recall  f1-score   support

           0       0.68      0.68      0.68       412
           1       0.46      0.45      0.45       244

    accuracy                           0.60       656
   macro avg       0.57      0.57      0.57       656
weighted avg       0.60      0.60      0.60       656
```



#Accuracy:

```
The Accuracy for Test Set is 59.60365853658537 %
```

#Conclusion:

Accuracy rate for fitting model given by Dicesion tree is 59.60% of our data.

Conclusion

Our project includes the understanding of the Machine Learning and its basic types. The classification models were used to analyse the water quality. The supervised classification models namely decision tree, KNN model and Logistic Model were fitted to our sample data of 3276 sample points. The water quality analysis is based on the parameters present in it, which are ph, Hardness, Solids, Chloramines, Sulfate, Conductiviti, Organic_carbon, Trihalomethanes, Turbidity there standard ranges were provided by WHO and lab reports. Of the three models that were fitted to this data, KNN model proved to be the best fit with accuracy of 60.97 %. With this accuracy, it concludes that our data is correct fitted.

Among all the three models logistic regression classifier has highest accuracy due to the very less miss classification. Hence we can say that logistic regression model is best to our data.

References

BOOKS

1. Data Mining Concepts and Techniques (Third Edition) by Jiawei Han , Micheline Kamber, Jian Pei

2.Fundamentals of Python Programming by Richard L.Halterman.

LINKS:

1. https://www.kaggle.com/datasets/dnyanadarekar/water-potability

2.https://www.analyticsvidhya.com/blog/2021/06/hypothesis-testing-parametric-and-non-parametric-tests-in-statistics/

3.https://www.javatpoint.com/machine-learning

4.https://www.wikipedia.org/

5. https://github.com/python

6. https://www.analyticsvidhya.com/blog/2020/11/popular-classification-models-for-machine-learning/

7. https://www.kdnuggets.com/2020/01/decision-tree-algorithm-explained.html