The classification of water quality for consumption using Machine Learning

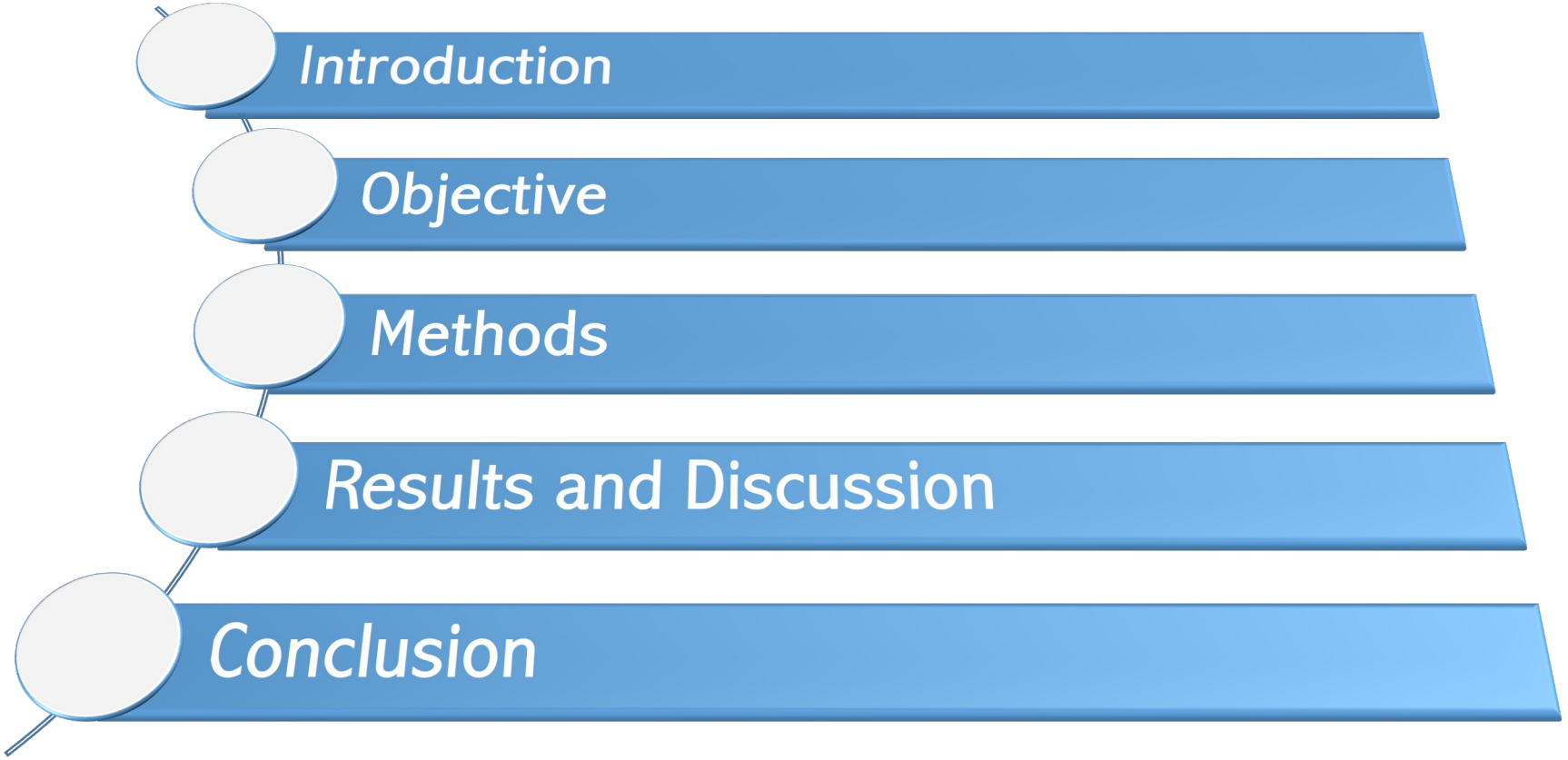
10th December 2023

6538135 Ms.Suganya Photawin

Advisor: Asst. Prof. Tanasanee Phienthrakul, Ph.D. and Asst. Prof. Vasin Suttichaya, Ph.D.

Mahidol University
Wisdom of the Land

Agenda



Mahidol University
Wisdom of the Land

Introduction

The quality of drinkable water is crucial for human health and must be free from various contaminants. The use of Machine Learning (ML) in water quality assessment enables rapid and accurate analysis, saving time and continuously monitoring changes in water quality to prevent potential health risks to consumers.

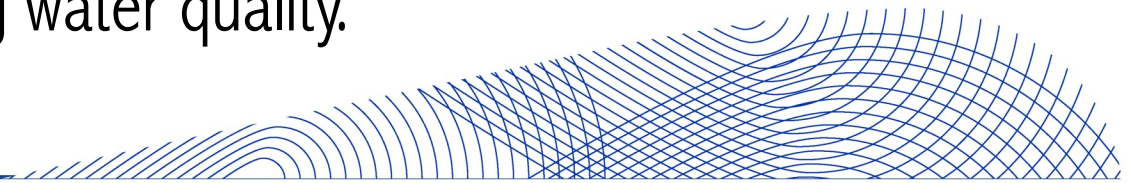
Mahidol University
Wisdom of the Land

Objective

1. **Water Quality Assessment:** Examine and analyze important parameters related to water quality assessment to determine if the water is of sufficient quality to be suitable for drinking.

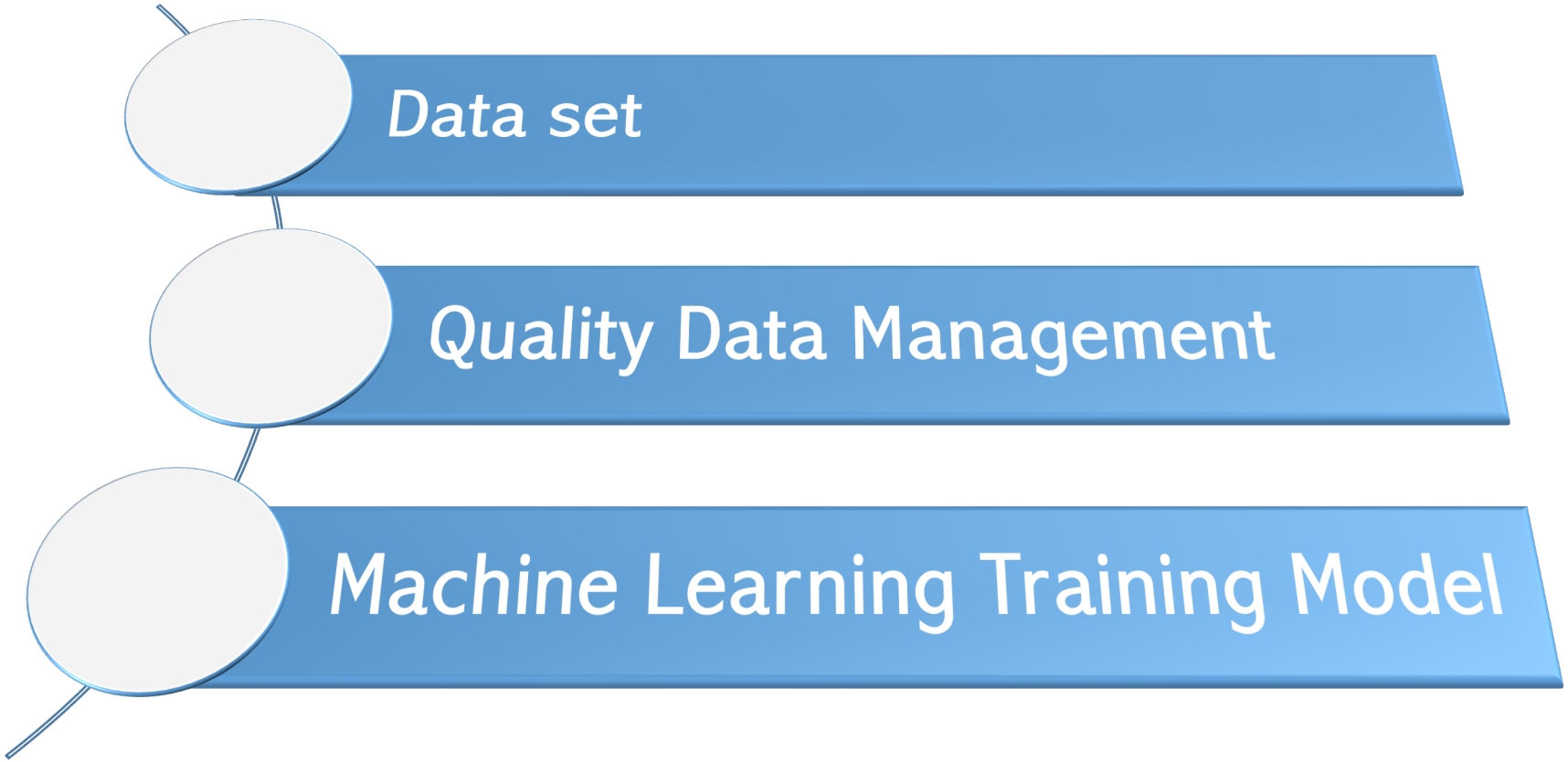
2. **Quality Data Management:** Develop and improve the process of managing high-quality data to suit the application of Machine Learning.

3. **Application of Machine Learning:** Apply Machine Learning algorithms to develop accurate and reliable methods for classifying water quality.



Mahidol University
Wisdom of the Land

Methods



Data Set

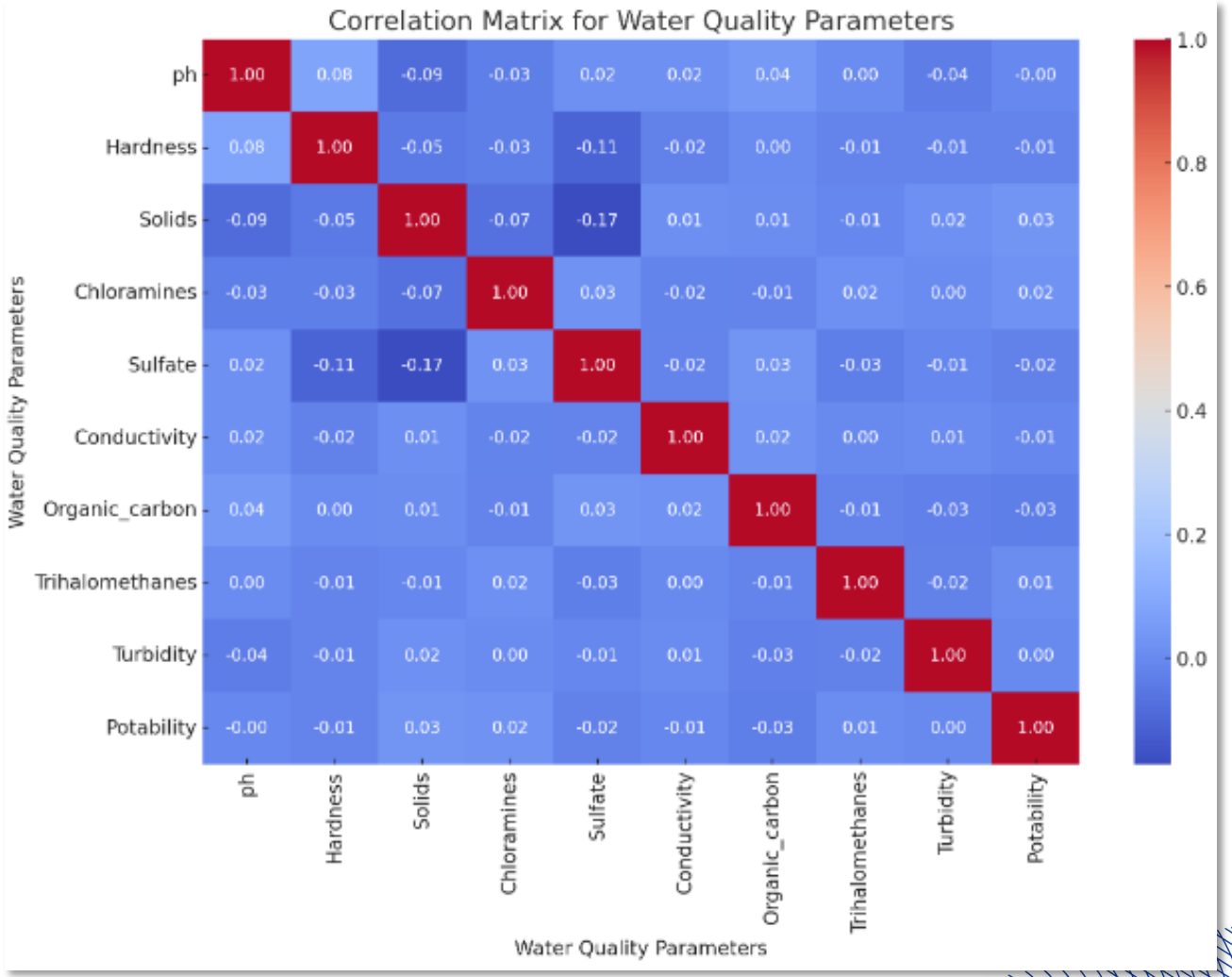
The dataset obtained from the Kaggle website, accessible at https://www.kaggle.com/datasets/adityakadiwal/water-potability,

```
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
None
Shape of the Dataset = (3276, 10)
```

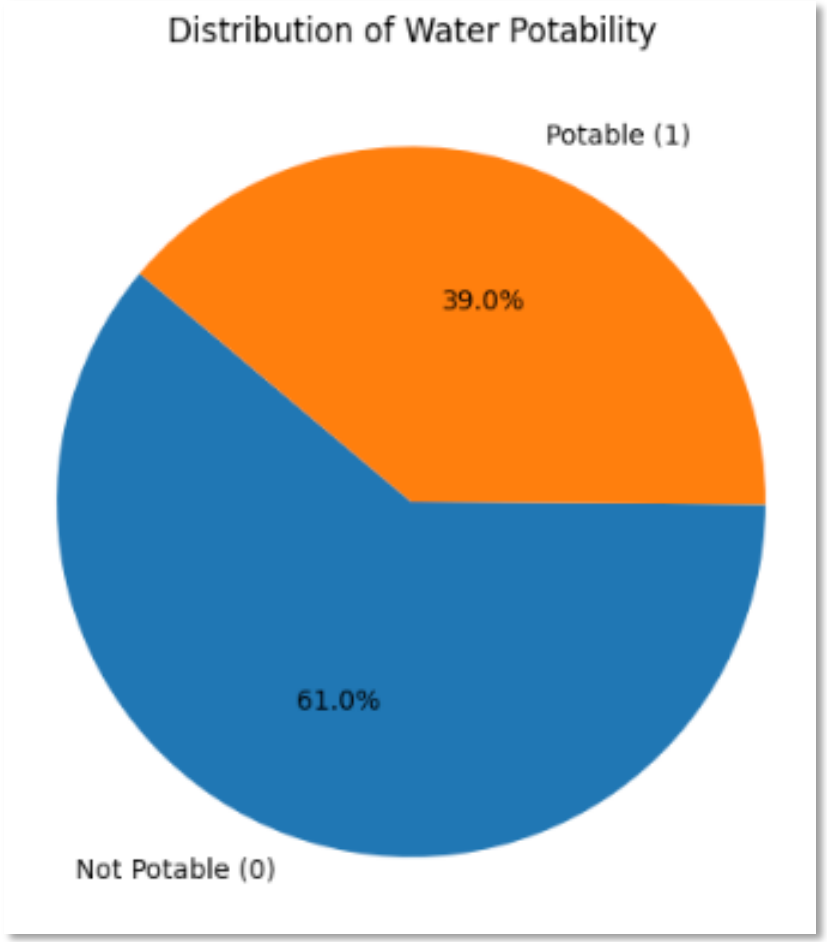
Data Set

1. ph: pH of water (0 to 14).
2. Hardness: Capacity of water to precipitate soap in mg/L.
3. Solids: Total dissolved solids in ppm. Or mg/L.
4. Chloramines: Amount of Chloramines in ppm.
5. Sulfate: Amount of Sulfates dissolved in mg/L.
6. Conductivity: Electrical conductivity of water in µS/cm.
7. Organic carbon: Amount of organic carbon in ppm.
8. Trihalomethanes: Amount of Trihalomethanes in µg/L.
9. Turbidity: Measure of light emitting property of water in NTU.
10. Potability: Indicates if water is safe for human consumption. Potable -1 and Not potable -0

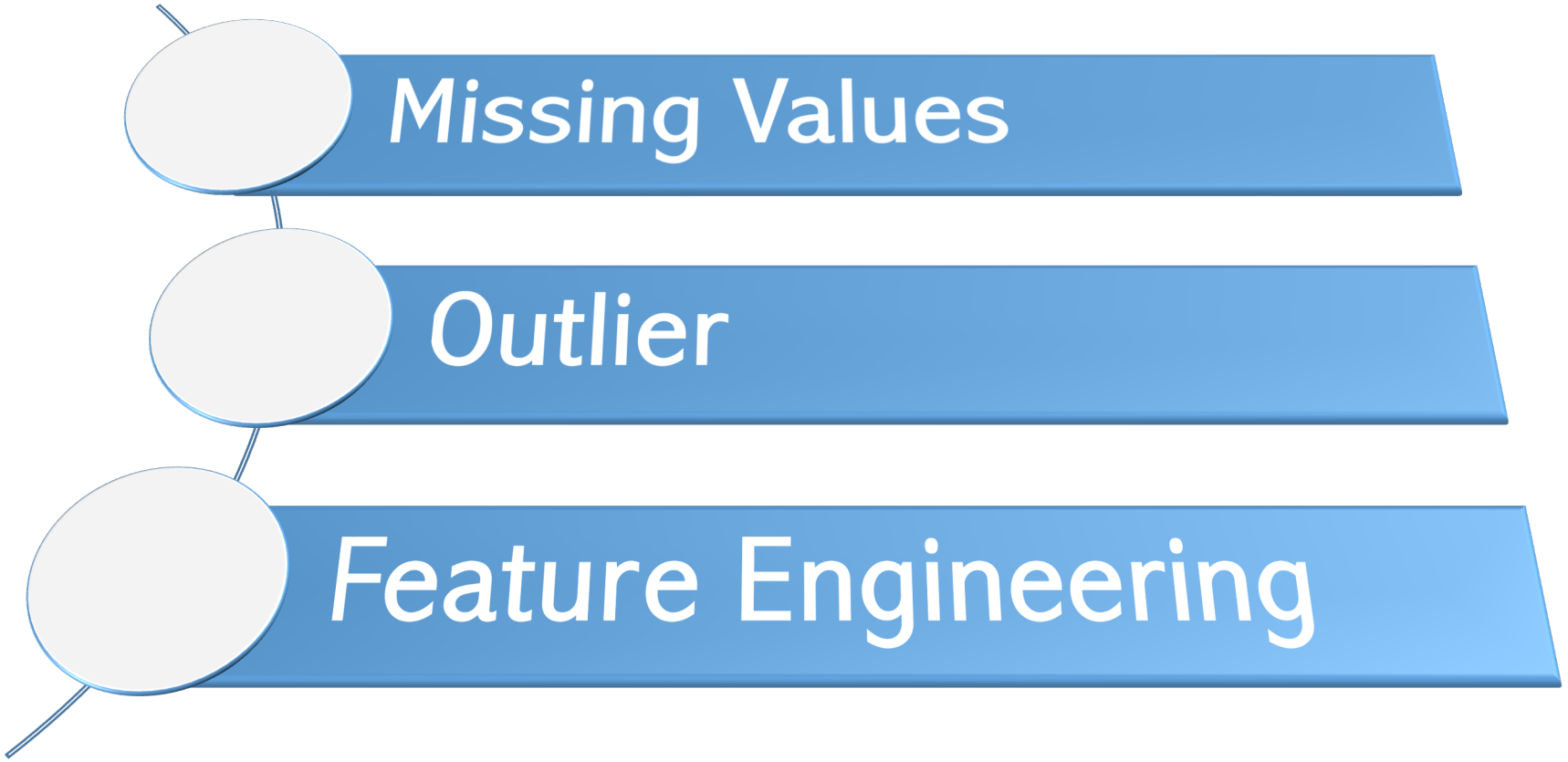
Data Set


Correlation Matrix for Water Quality Parameters
ph 1.00 0.08 -0.09 -0.03 0.02 0.02 0.04 0.00 -0.04 -0.00
Hardness 0.08 1.00 -0.05 -0.03 -0.11 -0.02 0.00 -0.01 -0.01 -0.01
Solids -0.09 -0.05 1.00 -0.07 -0.17 0.01 0.01 -0.01 0.02 0.03
Chloramines -0.03 -0.03 -0.07 1.00 0.03 -0.02 -0.01 0.02 0.00 0.02
Sulfate 0.02 -0.11 -0.17 0.03 1.00 -0.02 0.03 -0.03 -0.01 -0.02
Conductivity 0.02 -0.02 0.01 -0.02 -0.02 1.00 0.02 0.00 0.01 -0.01
Organic_carbon 0.04 0.00 0.01 -0.01 0.03 0.02 1.00 -0.01 -0.03 -0.03
Trihalomethanes 0.00 -0.01 -0.01 0.02 -0.03 0.00 -0.01 1.00 -0.02 0.01
Turbidity -0.04 -0.01 0.02 0.00 -0.01 0.01 -0.03 -0.02 1.00 0.00
Potability -0.00 -0.01 0.03 0.02 -0.02 -0.01 -0.03 0.01 0.00 1.00
Water Quality Parameters
ph
Hardness
Solids
Chloramines
Sulfate
Conductivity
Organic_carbon
Trihalomethanes
Turbidity
Potability
Water Quality Parameters
1.0
0.8
0.6
0.4
0.2
0.0

Data Set


Distribution of Water Potability
Potable (1)
39.0%
61.0%
Not Potable (0)

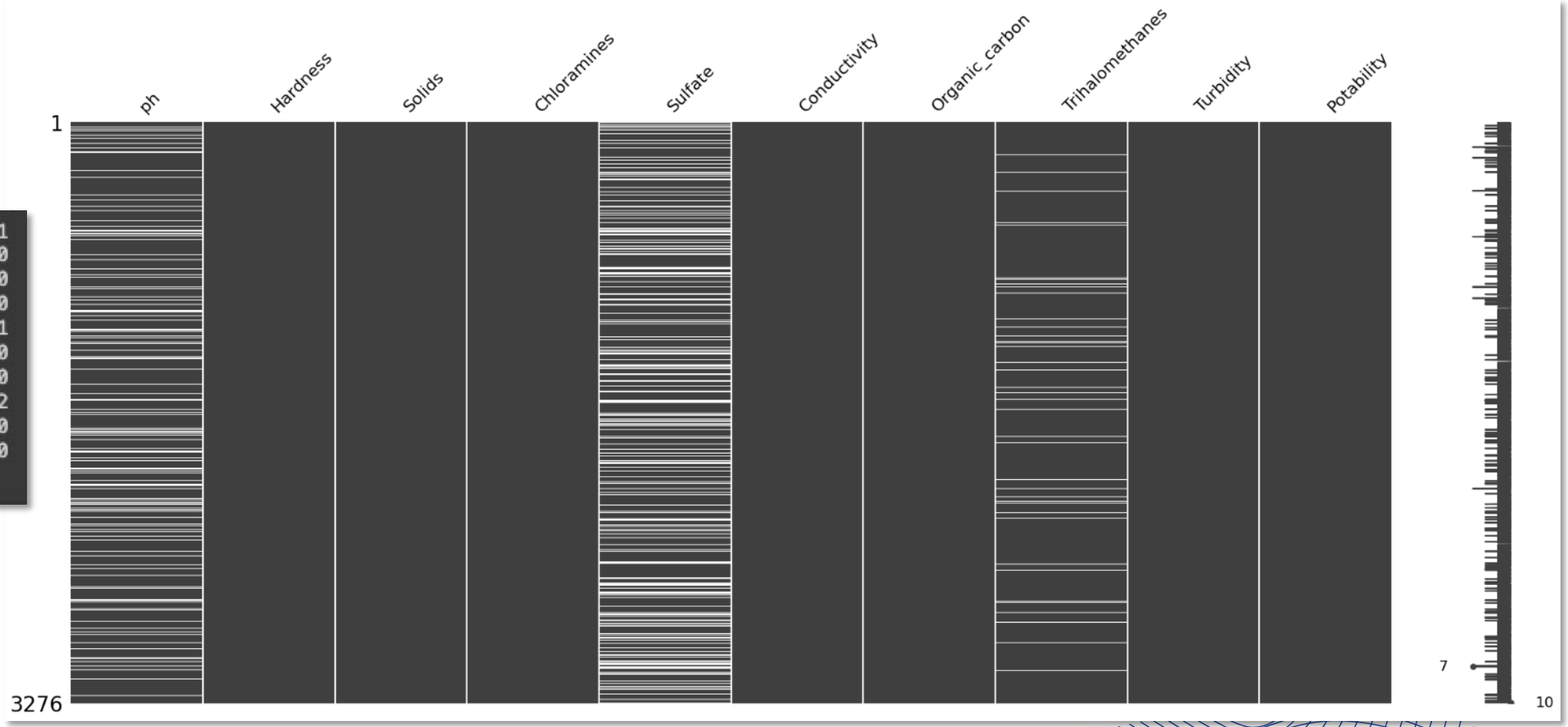
Quality Data Management



Missing Values

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

dtype: int64



Missing Values

- Replace missing value with 0
- Delete all data in a row when missing values are found.
- Replace Missing Value with condition

Replace missing value with 0

```
df.fillna(0, inplace=True)
```

```
ph                0
Hardness          0
Solids            0
Chloramines       0
Sulfate           0
Conductivity      0
Organic_carbon    0
Trihalomethanes   0
Turbidity         0
Potability        0
dtype: int64
```

Delete all data in a row when missing values are found.

Before

```
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
None
```

```
df = df.dropna()
```

After

```
Int64Index: 2011 entries, 3 to 3271
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   ph               2011 non-null   float64
 1   Hardness         2011 non-null   float64
 2   Solids           2011 non-null   float64
 3   Chloramines      2011 non-null   float64
 4   Sulfate          2011 non-null   float64
 5   Conductivity     2011 non-null   float64
 6   Organic_carbon   2011 non-null   float64
 7   Trihalomethanes  2011 non-null   float64
 8   Turbidity        2011 non-null   float64
 9   Potability       2011 non-null   int64
dtypes: float64(9), int64(1)
memory usage: 172.8 KB
None
```

Remain 2011 row (Missing 1265 = 38.61%)

Replace Missing Value with condition (ph)

ตารางมาตรฐาน pH น้ำดื่มในประเทศไทย

พารามิเตอร์	กรมอนามัย (1)	อย.(2)	สมอ.(3)
คุณภาพน้ำทางกายภาพ/ฟิสิกส์			
- ความเป็นกรด-ด่าง(pH)	อยู่ระหว่าง 6.5-8.5	อยู่ระหว่าง 6.5-8.5	อยู่ระหว่าง 6.5-8.5
- ความขุ่น (turbidity)	ไม่เกิน 5 NTU	ไม่เกิน 5 ซิลิกาสเกล	ไม่เกิน 5 NTU
- สี (Colour)	ไม่เกิน 15 หน่วย แพลทินัม-โคบอลต์	ไม่เกิน20ฮาเซนยูนิต	ไม่เกิน 5 หน่วย แพลทินัม-โคบอลต์
- กลิ่น	ไม่กำหนด	ต้องไม่มีกลิ่น แต่ไม่รวมถึงกลิ่นคลอรีน	ไม่กำหนด

(6.8 + 8.5) / 2

```
mean_ph_non_potable = df[df['Potability'] == 0]['ph'].mean()
if ((df['Potability'][x] == 0)) : df['ph'][x] = mean_ph_non_potable
elif ((df['Potability'][x] == 1) ) : df['ph'][x] = 7.5
```

```
Mean ph non potable : 7.0854
```

Replace Missing Value with condition (Sulfate)

```
print('Conditonal Statements to fill in the Missing Values of Sulfate Column')
print("\n")
print('if Potability = 0')
condition_1_mean_sulfate = df[(df['Potability'] == 0)][['Sulfate']].mean()
print("Sulfate : {:.4f}".format(float(condition_1_mean_sulfate)))

print("\n")
print('if Potability = 1')
condition_2_mean_sulfate = df[(df['Potability'] == 1)][['Sulfate']].mean()
print("Sulfate : {:.4f}".format(float(condition_2_mean_sulfate)))
```

```
if Potability = 0
Sulfate : 334.5643


if Potability = 1
Sulfate : 332.5670
```

Replace Missing Value with condition (Trihalomethanes)

```python
# Calculate the mean of 'Trihalomethanes' for each group of 'Potability'
mean_trihalomethanes_potable = df[df['Potability'] == 1]['Trihalomethanes'].mean()
mean_trihalomethanes_not_potable = df[df['Potability'] == 0]['Trihalomethanes'].mean()

# Replace missing 'Trihalomethanes' values for the potable group
df.loc[(df['Potability'] == 1) & (df['Trihalomethanes'].isnull()), 'Trihalomethanes'] = mean_trihalomethanes_potable

# Replace missing 'Trihalomethanes' values for the non-potable group
df.loc[(df['Potability'] == 0) & (df['Trihalomethanes'].isnull()), 'Trihalomethanes'] = mean_trihalomethanes_not_potable

# Print the mean values for each 'Potability' group
print('Mean Trihalomethanes for Potable water:', mean_trihalomethanes_potable)
print('Mean Trihalomethanes for Non-Potable water:', mean_trihalomethanes_not_potable)
```

```
Mean Trihalomethanes for Potable water: 66.53968374070116
Mean Trihalomethanes for Non-Potable water: 66.30355527306088
```

Outlier

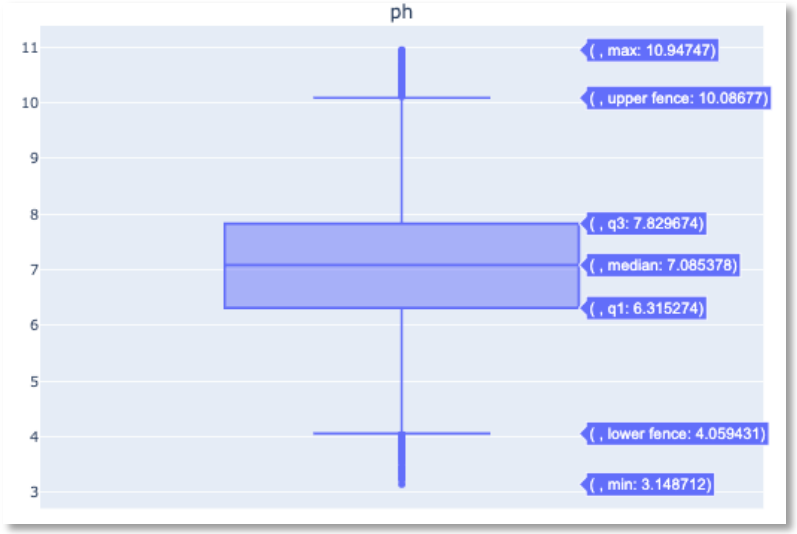
Tukey's Rule is a statistical method used to detect outliers in a dataset. This method utilizes the interquartile range (IQR), which is the difference between the 25th percentile (Q1) and the 75th percentile (Q3), to establish a range for considering potential outliers.
The IQR measures the spread of the middle 50% of the data. Tukey's Rule defines the boundaries for outliers by multiplying the IQR by 1.5 and then adding or subtracting this value from Q1 and Q3 to find the normal range.
According to Tukey's Rule:

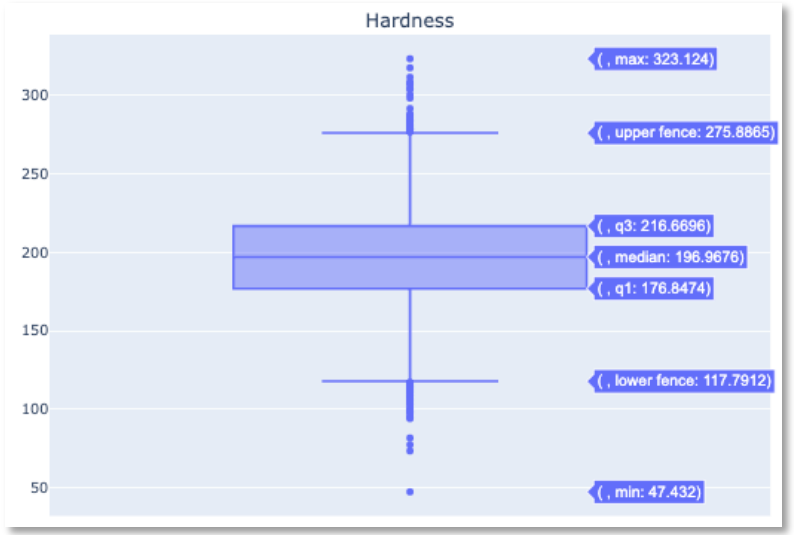
- Any value below Q1 - 1.5 * IQR or
- Any value above Q3 + 1.5 * IQR

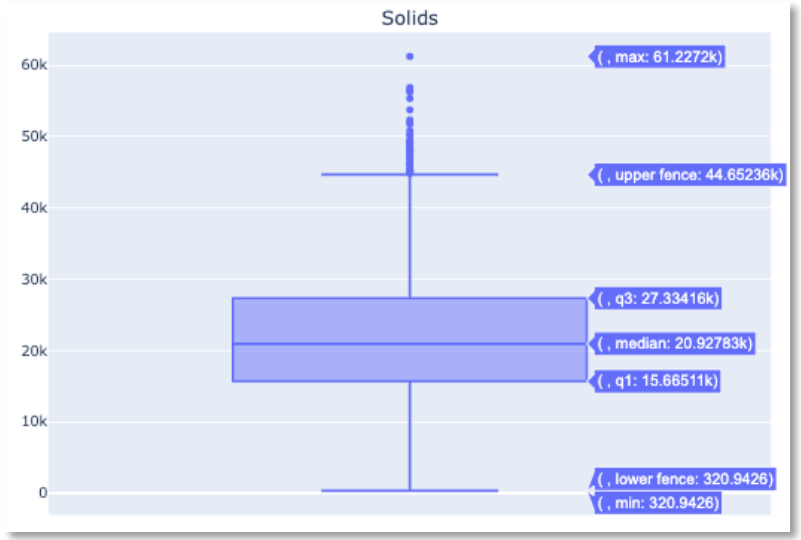
is considered an outlier. This method effectively identifies values that lie outside the main distribution of data and is commonly used in data analysis to decide which values should be considered part of the dataset or excluded.

```
Q1 = df[column].quantile(0.25)
Q3 = df[column].quantile(0.75)
IQR = Q3 - Q1
outlier_lower = Q1 - 1.5 * IQR
outlier_upper = Q3 + 1.5 * IQR
```

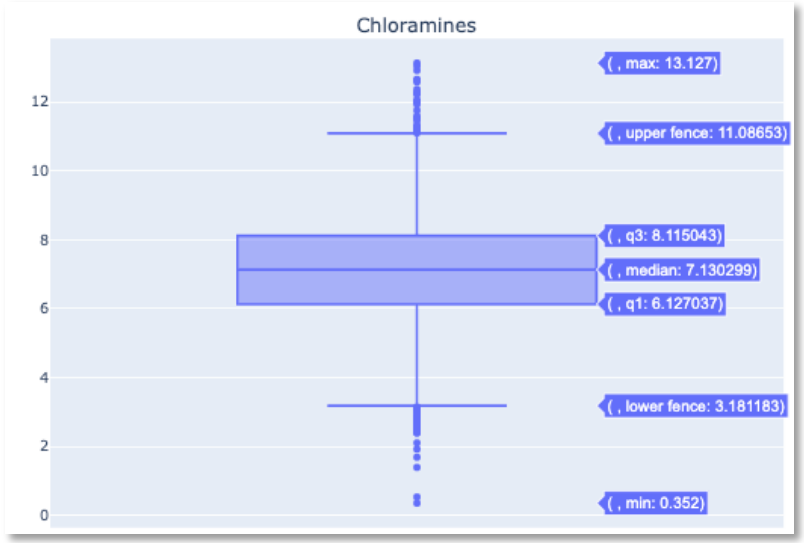
Outlier (1)

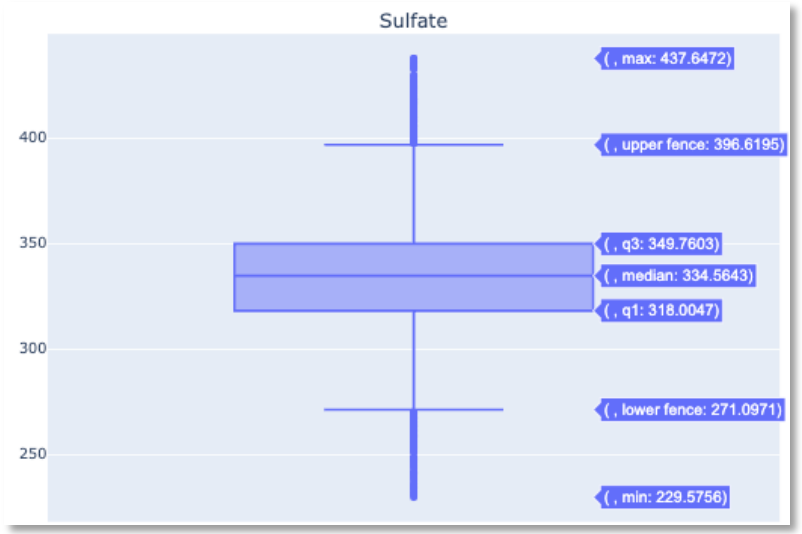

ph
11
10
9
8
7
6
5
4
3
(, max: 10.94747)
(, upper fence: 10.08677)
(, q3: 7.829674)
(, median: 7.085378)
(, q1: 6.315274)
(, lower fence: 4.059431)
(, min: 3.148712)

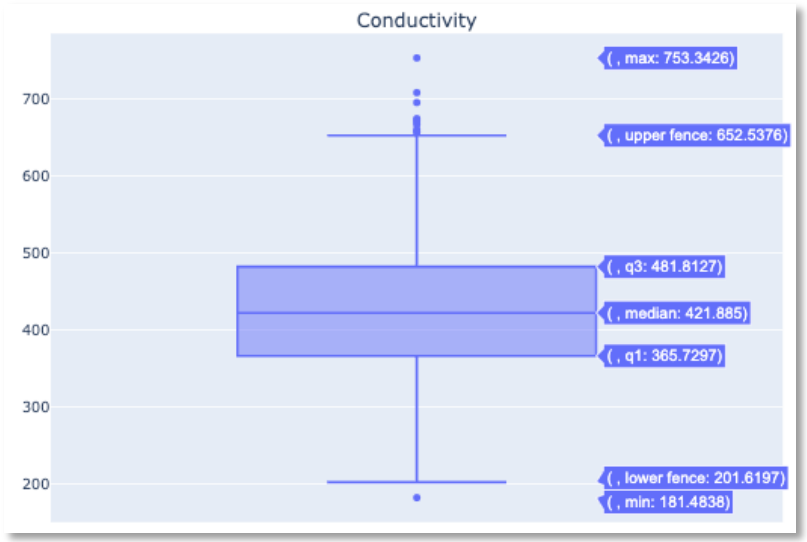

Hardness
300
250
200
150
100
50
(, max: 323.124)
(, upper fence: 275.8865)
(, q3: 216.6696)
(, median: 196.9676)
(, q1: 176.8474)
(, lower fence: 117.7912)
(, min: 47.432)


Solids
60k
50k
40k
30k
20k
10k
0
(, max: 61.2272k)
(, upper fence: 44.65236k)
(, q3: 27.33416k)
(, median: 20.92783k)
(, q1: 15.66511k)
(, lower fence: 320.9426)
(, min: 320.9426)

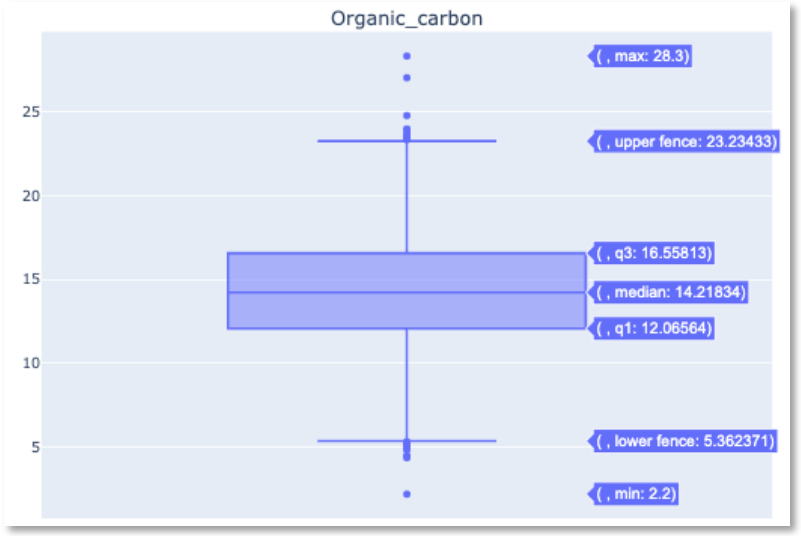
Outlier (2)

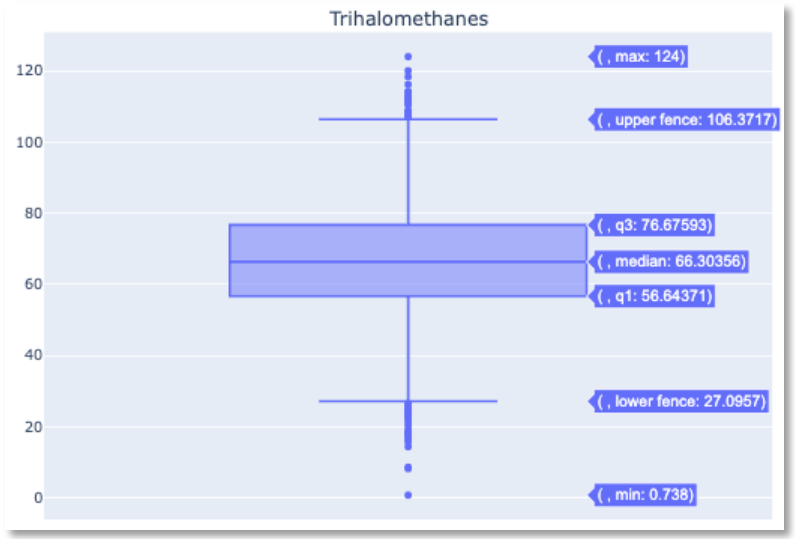

Chloramines
(, max: 13.127)
(, upper fence: 11.08653)
(, q3: 8.115043)
(, median: 7.130299)
(, q1: 6.127037)
(, lower fence: 3.181183)
(, min: 0.352)
12
10
8
6
4
2
0

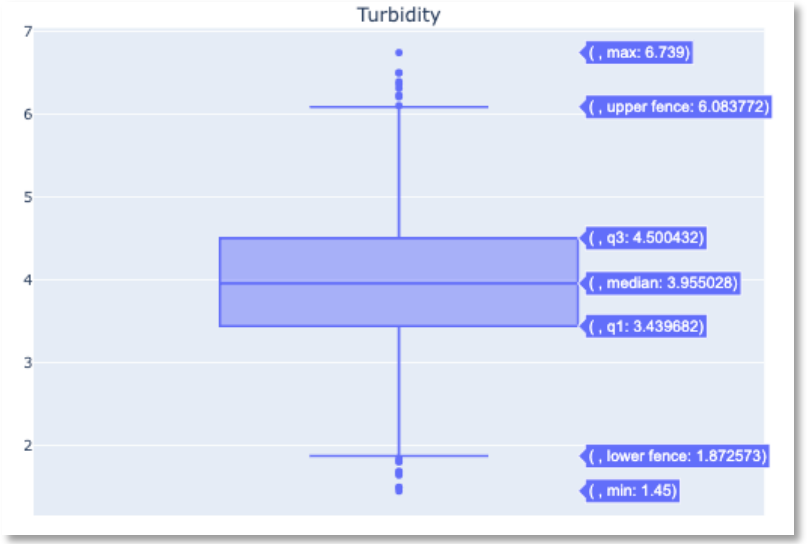

Sulfate
(, max: 437.6472)
(, upper fence: 396.6195)
(, q3: 349.7603)
(, median: 334.5643)
(, q1: 318.0047)
(, lower fence: 271.0971)
(, min: 229.5756)
400
350
300
250


Conductivity
(, max: 753.3426)
(, upper fence: 652.5376)
(, q3: 481.8127)
(, median: 421.885)
(, q1: 365.7297)
(, lower fence: 201.6197)
(, min: 181.4838)
700
600
500
400
300
200

Outlier (3)


Organic_carbon
(, max: 28.3)
(, upper fence: 23.23433)
(, q3: 16.55813)
(, median: 14.21834)
(, q1: 12.06564)
(, lower fence: 5.362371)
(, min: 2.2)
25
20
15
10
5


Trihalomethanes
(, max: 124)
(, upper fence: 106.3717)
(, q3: 76.67593)
(, median: 66.30356)
(, q1: 56.64371)
(, lower fence: 27.0957)
(, min: 0.738)
120
100
80
60
40
20
0


Turbidity
(, max: 6.739)
(, upper fence: 6.083772)
(, q3: 4.500432)
(, median: 3.955028)
(, q1: 3.439682)
(, lower fence: 1.872573)
(, min: 1.45)
7
6
5
4
3
2

Feature Engineering

```
df['Hardness_Sulfate'] = df['Hardness'] * df['Sulfate']

df['Chloramines_OrganicCarbon'] = df['Chloramines'] * df['Organic_carbon']

df['Solids_Sulfate_ratio'] = df.apply(lambda row: row['Solids'] / row['Sulfate']
                                      if row['Sulfate'] > 0 else None, axis=1)
df['ph_Hardness'] = df['ph'] * df['Hardness']
```

Hardness_Sulfate	Chloramines_OrganicCarbon	Solids_Sulfate_ratio	ph_Hardness
75505.501517	75.774616	56.418973	1451.726415
43300.287457	100.723120	55.684538	480.945936
75021.444822	156.471513	59.508867	1816.117313
76506.892200	148.586079	61.695917	1782.893330
56166.050134	75.667432	57.971347	1646.615390

Machine Learning Training Model

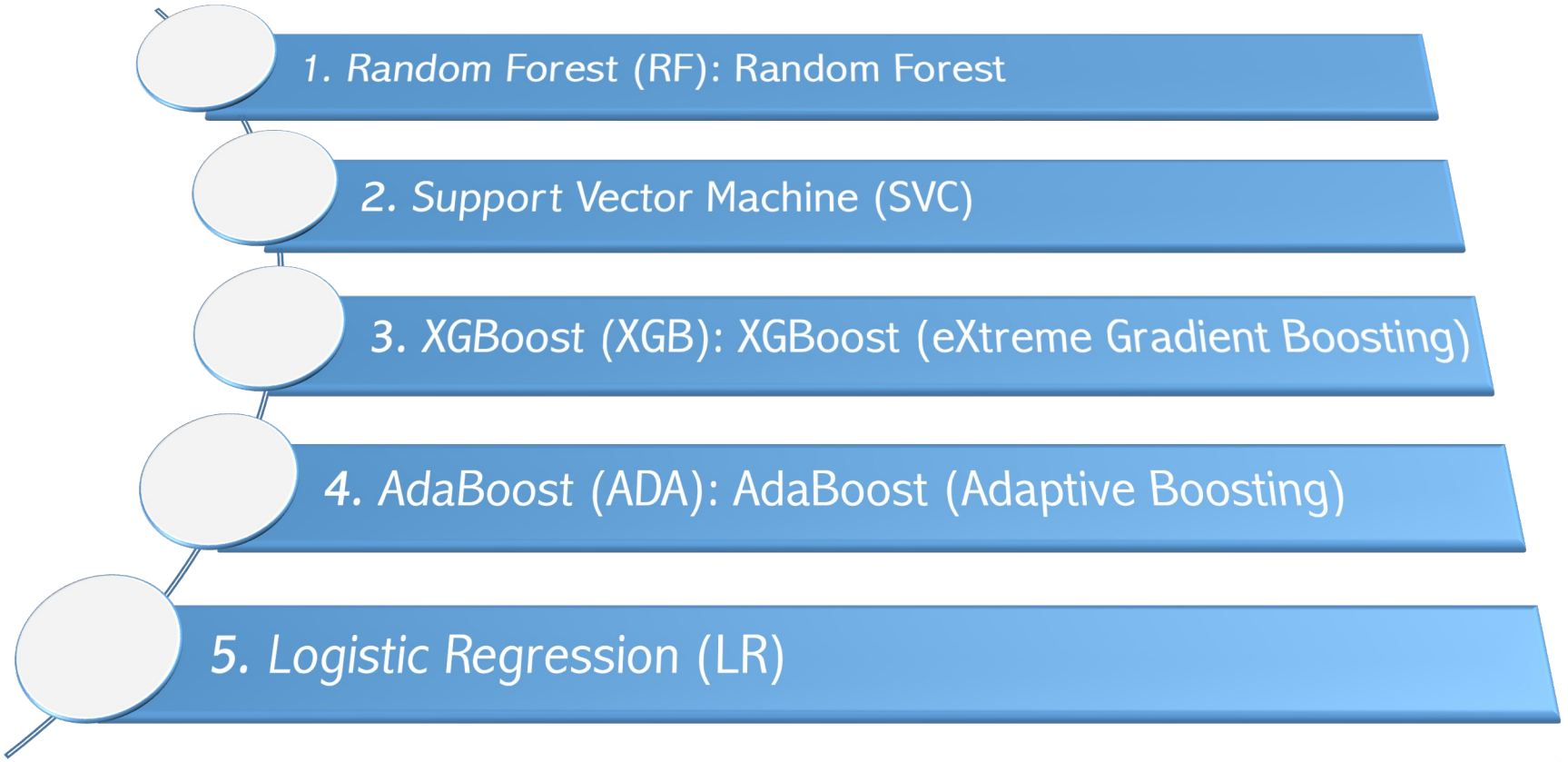
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=101)
```

```
models =[("LR", LogisticRegression(max_iter=1000)),("SVC", SVC()),('KNN',KNeighborsClassifier(n_neighbors=10)),
         ("DTC", DecisionTreeClassifier()),("GNB", GaussianNB()),
        ("SGDC", SGDClassifier()),("Perc", Perceptron()),("NC",NearestCentroid()),
        ("Ridge", RidgeClassifier()),("NuSVC", NuSVC()),("BNB", BernoulliNB()),
         ('RF',RandomForestClassifier()),('ADA',AdaBoostClassifier()),
        ('XGB',GradientBoostingClassifier()),('PAC',PassiveAggressiveClassifier())]
```

Machine Learning Training Model

	Version1	Version2	Version3	Version4	Version5
RF	0.660	0.706	0.784	0.787	0.823
SVC	0.649	0.720	0.687	0.639	0.698
XGB	0.611	0.672	0.785	0.814	0.825
ADA	0.544	0.583	0.772	0.776	0.813
LR	0.307	0.541	0.807	0.307	0.699

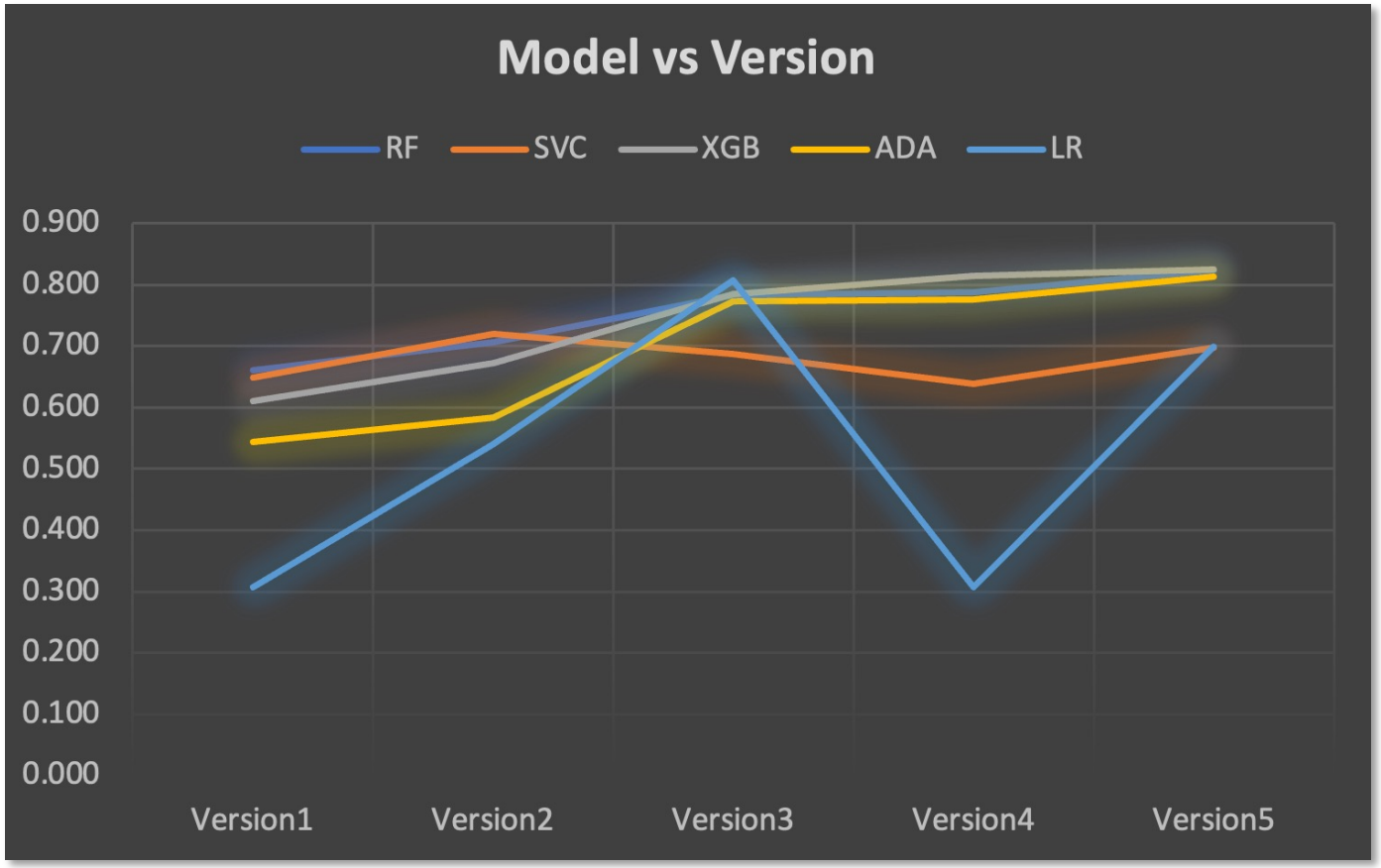
Machine Learning Training Model



Results and Discussion

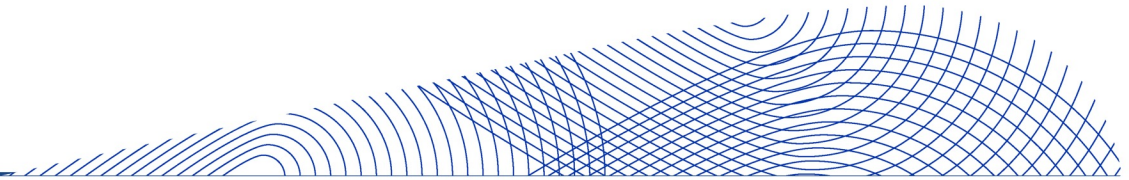
Version	01	02	03	04	05
Best accuracy	0.660	0.720	0.807	0.814	0.825
Model	RF	SVC	LR	XGB	XGB
Replace missing value with 0					
Delete all data in a row when missing values are found.					
Replace Missing Value with condition					
Replace Outlier with condition					
Add Feature Engineering					

Results and Discussion



Conclusion

1. Effective Data Management Enhances Model Performance
2. Importance of Specific Data Management Techniques
3. Impact of Removing Rows with Missing Values
4. Model-Specific Responses to Data Management
5. Testing and Evaluation Are Crucial
6. Selecting Appropriate Techniques for Dataset and Model



Thank You