Usama Arif roll no 14

In [173]:

```
import pandas as pd
import math
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix,f1_score
```

In [174]:

```
dataset=pd.read_csv('water_potability.csv')
dataset.head()
```

Out[174]:

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbo
0	NaN	204.890455	20791.318981	7.300212	368.516441	564.308654	10.37978
1	3.716080	129.422921	18630.057858	6.635246	NaN	592.885359	15.18001
2	8.099124	224.236259	19909.541732	9.275884	NaN	418.606213	16.86863
3	8.316766	214.373394	22018.417441	8.059332	356.886136	363.266516	18.43652
4	9.092223	181.101509	17978.986339	6.546600	310.135738	398.410813	11.55827

In [176]:

```
dataset['ph'].fillna(dataset['ph'].mean(),inplace=True)
dataset['Sulfate'].fillna(dataset['Sulfate'].mean(),inplace=True)
dataset['Trihalomethanes'].fillna(dataset['Trihalomethanes'].mean(),inplace=True)
```

In [177]:

```
dataset.isna().sum()
```

Out[177]:

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

In [178]:

```
1 dataset.describe()
```

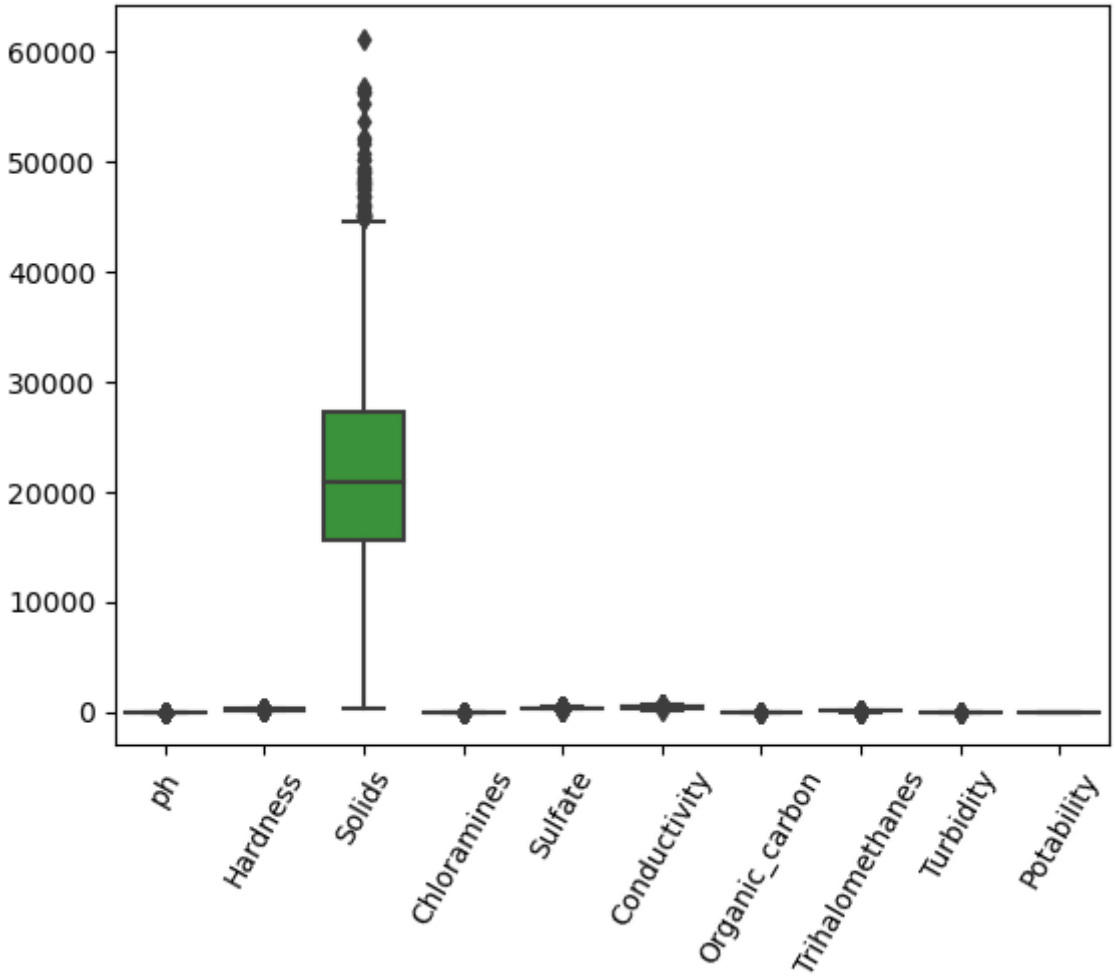
Out[178]:

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Orga
count	3276.000000	3276.000000	3276.000000	3276.000000	3276.000000	3276.000000	3
mean	7.080795	196.369496	22014.092526	7.122277	333.775777	426.205111	
std	1.469956	32.879761	8768.570828	1.583085	36.142612	80.824064	
min	0.000000	47.432000	320.942611	0.352000	129.000000	181.483754	
25%	6.277673	176.850538	15666.690297	6.127421	317.094638	365.734414	
50%	7.080795	196.967627	20927.833607	7.130299	333.775777	421.884968	
75%	7.870050	216.667456	27332.762127	8.114887	350.385756	481.792304	
max	14.000000	323.124000	61227.196008	13.127000	481.030642	753.342620	

visualization

In [179]:

```
sns.boxplot(data=dataset)
plt.xticks(rotation = 60);
```

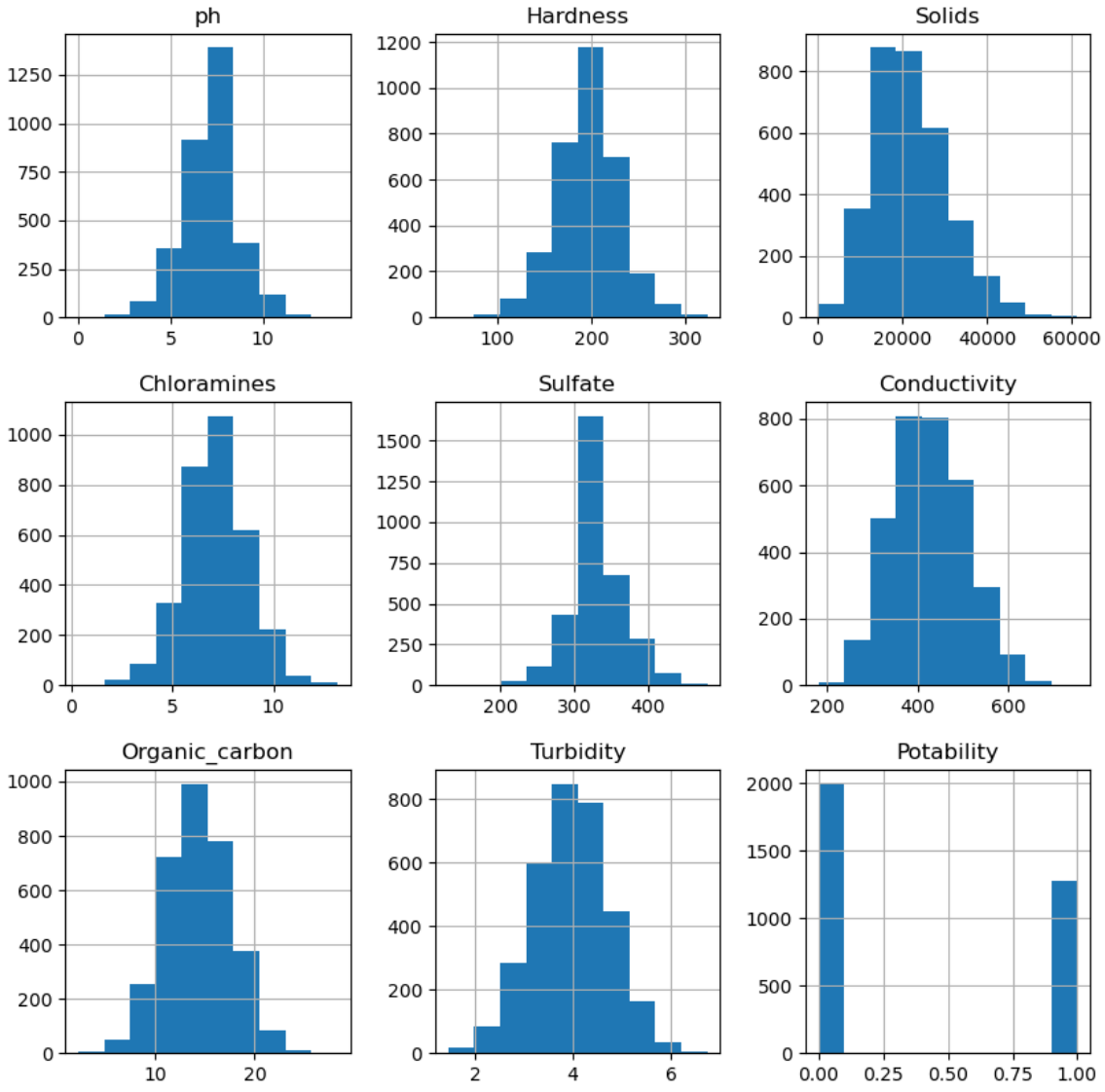


In [87]:

```
1 dataset.hist(figsize=(10,10))
```

Out[87]:

```
array([[<Axes: title={'center': 'ph'}>,
        <Axes: title={'center': 'Hardness'}>,
        <Axes: title={'center': 'Solids'}>],
       [<Axes: title={'center': 'Chloramines'}>,
        <Axes: title={'center': 'Sulfate'}>,
        <Axes: title={'center': 'Conductivity'}>],
       [<Axes: title={'center': 'Organic_carbon'}>,
        <Axes: title={'center': 'Turbidity'}>,
        <Axes: title={'center': 'Potability'}>]], dtype=object)
```



In [180]:

```
x=dataset.drop('Potability',axis=1)
y=dataset['Potability']
scaler=StandardScaler()
features=['ph', 'Hardness', 'Solids', 'Chloramines', 'Sulfate', 'Conductivity',
       'Organic_carbon', 'Trihalomethanes', 'Turbidity']
transformer=ColumnTransformer([('features',scaler,features)])
x_transformed=transformer.fit_transform(x)
x_transformed


```

Out[180]:

```
array([[-6.04313345e-16,  2.59194711e-01, -1.39470871e-01, ...,
        -1.18065057e+00,  1.30614943e+00, -1.28629758e+00],
       [-2.28933938e+00, -2.03641367e+00, -3.85986650e-01, ...,
         2.70597240e-01, -6.38479983e-01,  6.84217891e-01],
       [ 6.92867789e-01,  8.47664833e-01, -2.40047337e-01, ...,
         7.81116857e-01,  1.50940884e-03, -1.16736546e+00],
       ...,
       [ 1.59125368e+00, -6.26829230e-01,  1.27080989e+00, ...,
        -9.81329234e-01,  2.18748247e-01, -8.56006782e-01],
       [-1.32951593e+00,  1.04135450e+00, -1.14405809e+00, ...,
        -9.42063817e-01,  7.03468419e-01,  9.50797383e-01],
       [ 5.40150905e-01, -3.85462310e-02, -5.25811937e-01, ...,
         5.60940070e-01,  7.80223466e-01, -2.12445866e+00]])
```

Training the data

In [ ]:

```

```

In [181]:

```
model=KNeighborsClassifier(n_neighbors=57,p=2,metric='minkowski')
x_train,x_test,y_train,y_test=train_test_split(x_transformed,y,test_size=0.2,random_
model.fit(x_train,y_train);
```

making predictions

In [182]:

```
y_preds=model.predict(x_test)
y_preds
```

Out[182]:

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

In [183]:

```
predicted_data=pd.DataFrame(y_preds,columns=['prediction'])
actual=pd.DataFrame({'y_test':y_test}).reset_index(drop=True)
pre_vs_act=pd.concat([actual,predicted_data],axis=1)

pre_vs_act.head(10)
```

Out[183]:

	y_test	prediction
0	0	0
1	1	0
2	0	0
3	0	0
4	1	0
5	1	1
6	0	0
7	0	0
8	0	0
9	0	0

Model evaluation

In [184]:

```
f1_score=f1_score(y_test,y_preds)
f1_score

```

Out[184]:

```
0.25082508250825086
```

In [185]:

```
model.score(x_test,y_test)
```

Out[185]:

```
0.6539634146341463
```

In [ ]:

```

```