In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

In [2]:

```
data=pd.read_csv("water_potability.csv")
```

In [3]:

```
data.head()
```

Out[3]:

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon
0	NaN	204.890455	20791.318981	7.300212	368.516441	564.308654	10.379783
1	3.716080	129.422921	18630.057858	6.635246	NaN	592.885359	15.180013
2	8.099124	224.236259	19909.541732	9.275884	NaN	418.606213	16.868637
3	8.316766	214.373394	22018.417441	8.059332	356.886136	363.266516	18.436524
4	9.092223	181.101509	17978.986339	6.546600	310.135738	398.410813	11.558279

In [4]:

```
data.shape
```

Out[4]:

```
(3276, 10)
```

In [5]:

```
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)
memory usage: 256.1 KB
```

In [6]:

```
data.isnull().sum() #yes we have empyt block in the dataset
```

Out[6]:

```
ph                 491
Hardness             0
Solids               0
Chloramines          0
Sulfate            781
Conductivity         0
Organic_carbon       0
Trihalomethanes    162
Turbidity            0
Potability           0
dtype: int64
```

Preprocessing

In [7]:

```
# total rows=3276
data.count()
# ph,Sulfate,Trihalomethanes have less value
```

Out[7]:

```
ph                 2785
Hardness           3276
Solids             3276
Chloramines        3276
Sulfate            2495
Conductivity       3276
Organic_carbon     3276
Trihalomethanes    3114
Turbidity          3276
Potability         3276
dtype: int64
```

In [8]:

```
data.mean()
```

Out[8]:

```
ph                     7.080795
Hardness             196.369496
Solids             22014.092526
Chloramines            7.122277
Sulfate              333.775777
Conductivity         426.205111
Organic_carbon        14.284970
Trihalomethanes       66.396293
Turbidity              3.966786
Potability             0.390110
dtype: float64
```

In [9]:
```
data.describe()
```

Out[9]:

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic
count	2785.000000	3276.000000	3276.000000	3276.000000	2495.000000	3276.000000	327
mean	7.080795	196.369496	22014.092526	7.122277	333.775777	426.205111	1
std	1.594320	32.879761	8768.570828	1.583085	41.416840	80.824064	3
min	0.000000	47.432000	320.942611	0.352000	129.000000	181.483754	2
25%	6.093092	176.850538	15666.690297	6.127421	307.699498	365.734414	1
50%	7.036752	196.967627	20927.833607	7.130299	333.073546	421.884968	1
75%	8.062066	216.667456	27332.762127	8.114887	359.950170	481.792304	1
max	14.000000	323.124000	61227.196008	13.127000	481.030642	753.342620	2

In [10]:
```
data.fillna(data.mean(),inplace=True)
```

In [11]:
```
data #you can find empty blocks are filled with mean of respective columns
```

Out[11]:

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbo
0	7.080795	204.890455	20791.318981	7.300212	368.516441	564.308654	10.37978
1	3.716080	129.422921	18630.057858	6.635246	333.775777	592.885359	15.18001
2	8.099124	224.236259	19909.541732	9.275884	333.775777	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
...	...	...	...	...	...	...	
3271	4.668102	193.681735	47580.991603	7.166639	359.948574	526.424171	13.89441
3272	7.808856	193.553212	17329.802160	8.061362	333.775777	392.449580	19.90322
3273	9.419510	175.762646	33155.578218	7.350233	333.775777	432.044783	11.03907
3274	5.126763	230.603758	11983.869376	6.303357	333.775777	402.883113	11.16894
3275	7.874671	195.102299	17404.177061	7.509306	333.775777	327.459760	16.14036

3276 rows × 10 columns

In [12]:

```
# now check the null value
data.isnull().sum()
#you can see none of column has empty value
```

Out[12]:

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

In [13]:

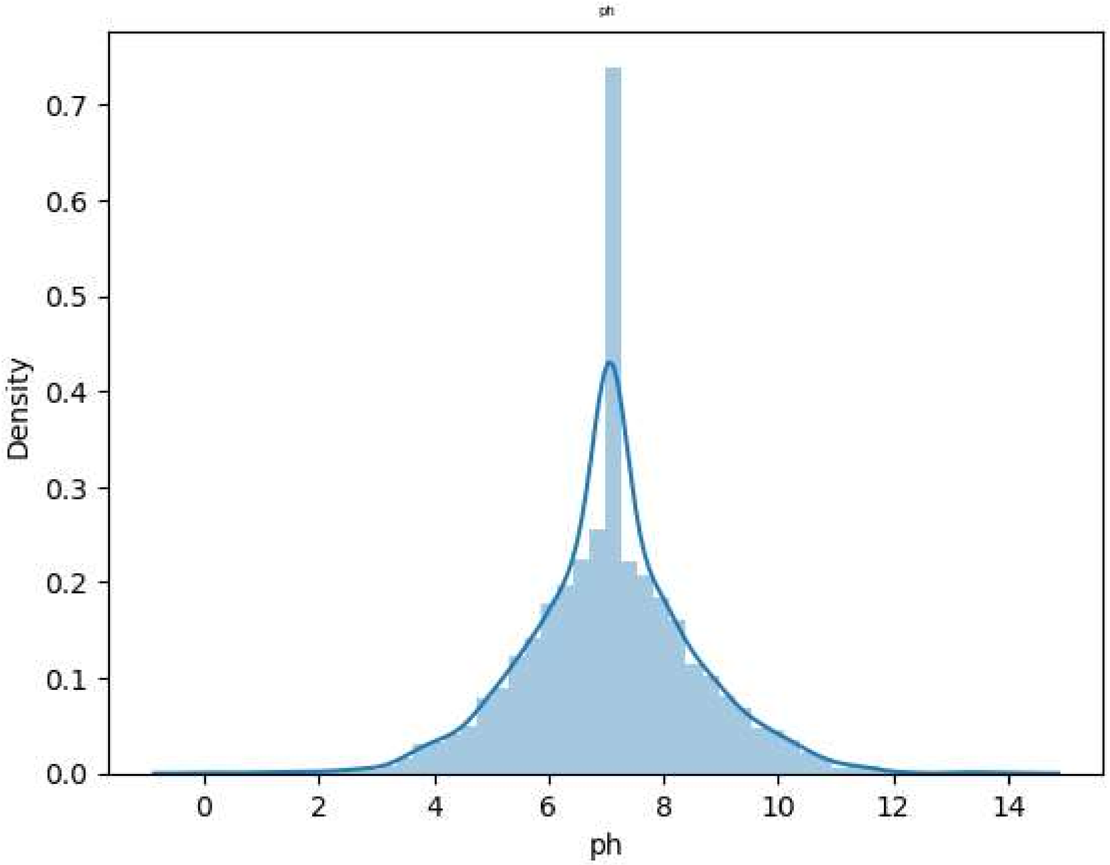
```
# checking for the duplicate value
data.duplicated().sum() #it means no duplicate values are present
```

Out[13]:

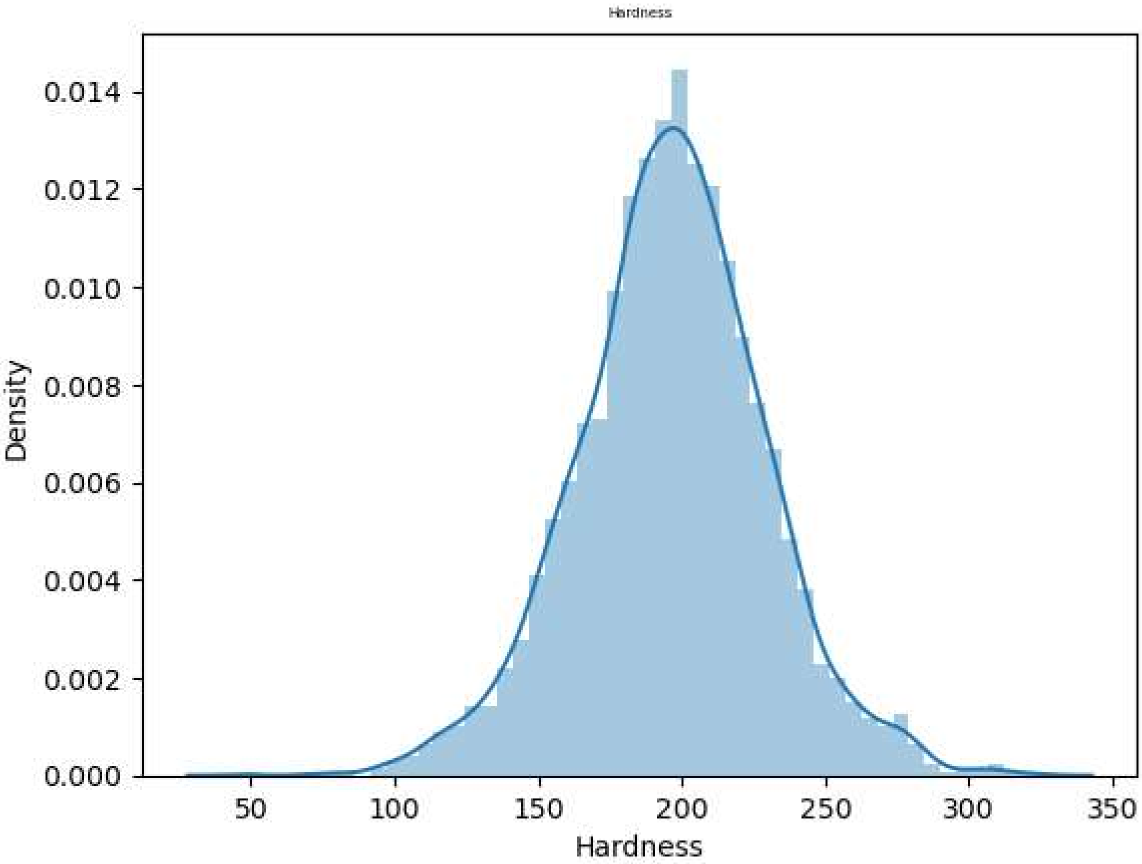
```
0
```

In [14]:
```
# Observing the nature of data distribution
for i in data.columns:
    sns.distplot(data[i])
    plt.title(i,size=5)
    plt.show()
# we can observe that all data is normaly distributed, So we can use it direct
```

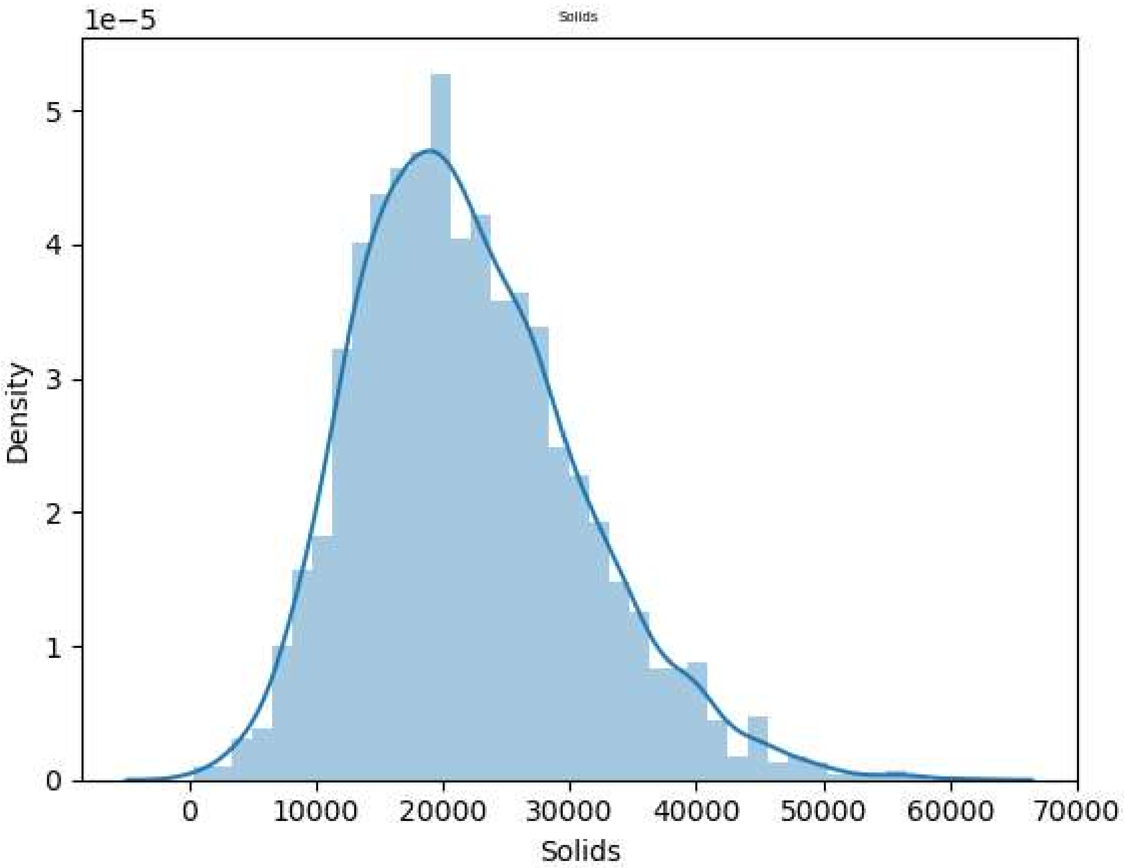
```
C:\Users\A to Z Infosys\anaconda3\lib\site-packages\seaborn\distributions.py:
2619: FutureWarning: `distplot` is a deprecated function and will be removed
in a future version. Please adapt your code to use either `displot` (a figure
-level function with similar flexibility) or `histplot` (an axes-level functi
on for histograms).
  warnings.warn(msg, FutureWarning)
```



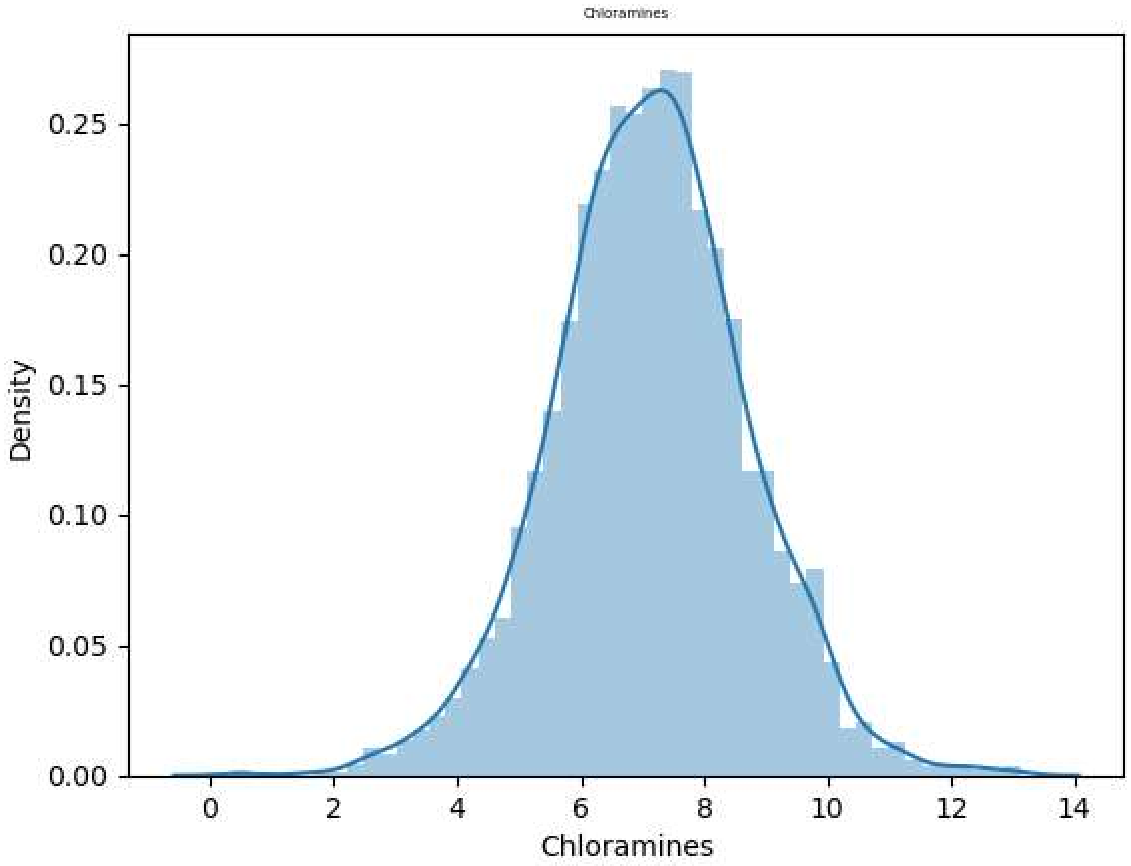
```
C:\Users\A to Z Infosys\anaconda3\lib\site-packages\seaborn\distributions.py:
2619: FutureWarning: `distplot` is a deprecated function and will be removed
in a future version. Please adapt your code to use either `displot` (a figure
-level function with similar flexibility) or `histplot` (an axes-level functi
on for histograms).
  warnings.warn(msg, FutureWarning)
```



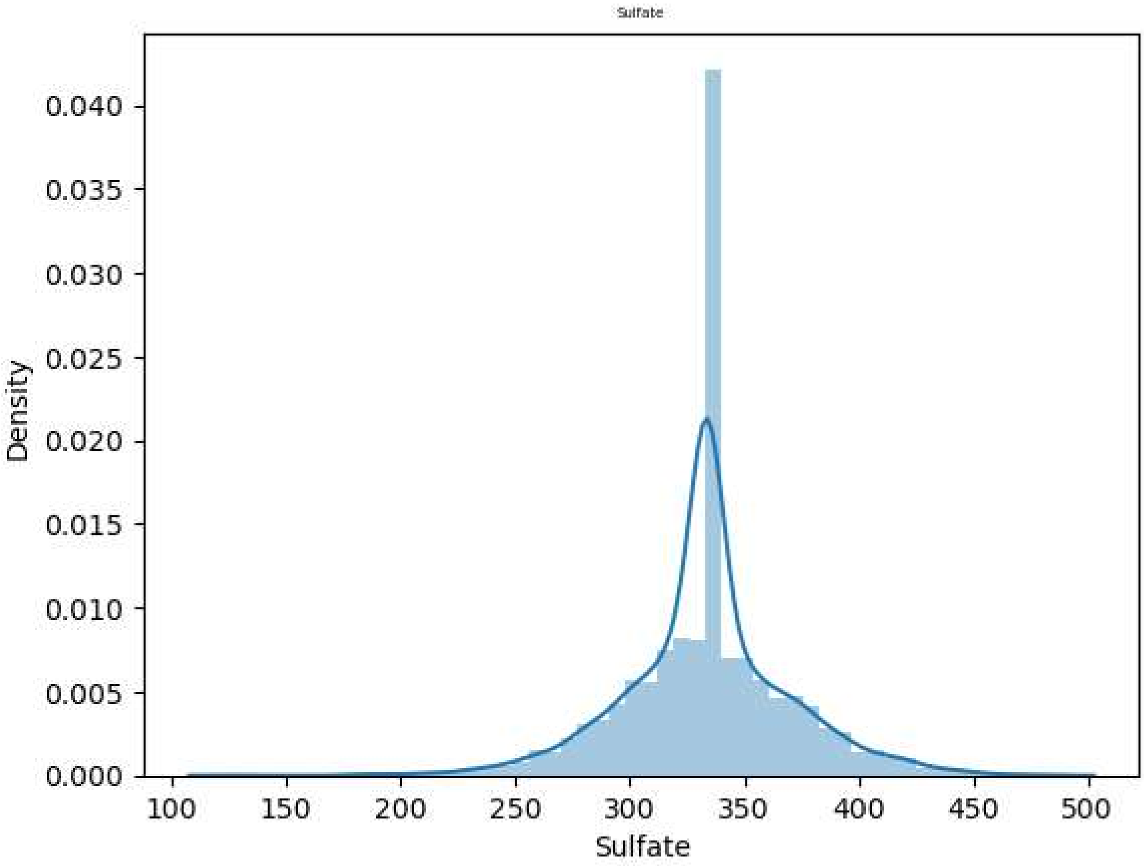
```
C:\Users\A to Z Infosys\anaconda3\lib\site-packages\seaborn\distributions.py:
2619: FutureWarning: `distplot` is a deprecated function and will be removed
in a future version. Please adapt your code to use either `displot` (a figure
-level function with similar flexibility) or `histplot` (an axes-level functi
on for histograms).
  warnings.warn(msg, FutureWarning)
```



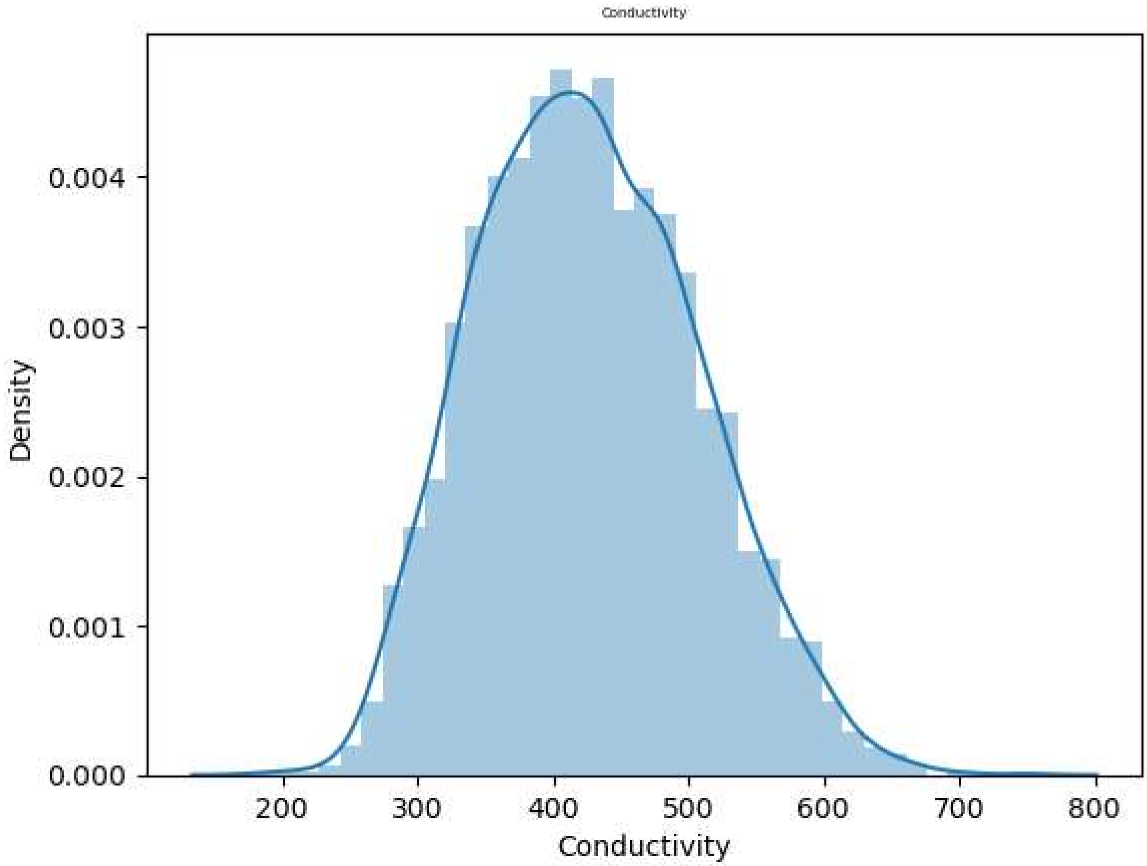
```
C:\Users\A to Z Infosys\anaconda3\lib\site-packages\seaborn\distributions.py:
2619: FutureWarning: `distplot` is a deprecated function and will be removed
in a future version. Please adapt your code to use either `displot` (a figure
-level function with similar flexibility) or `histplot` (an axes-level functi
on for histograms).
  warnings.warn(msg, FutureWarning)
```



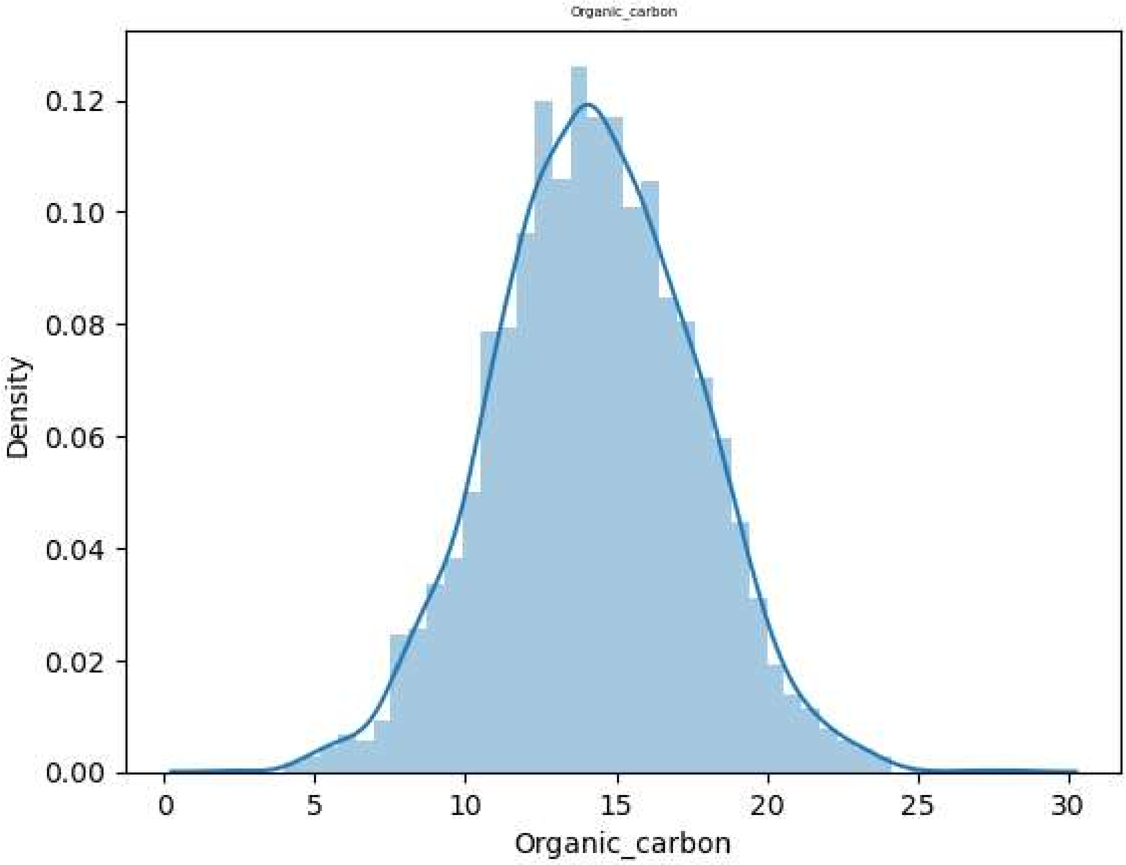
```
C:\Users\A to Z Infosys\anaconda3\lib\site-packages\seaborn\distributions.py:
2619: FutureWarning: `distplot` is a deprecated function and will be removed
in a future version. Please adapt your code to use either `displot` (a figure
-level function with similar flexibility) or `histplot` (an axes-level functi
on for histograms).
  warnings.warn(msg, FutureWarning)
```



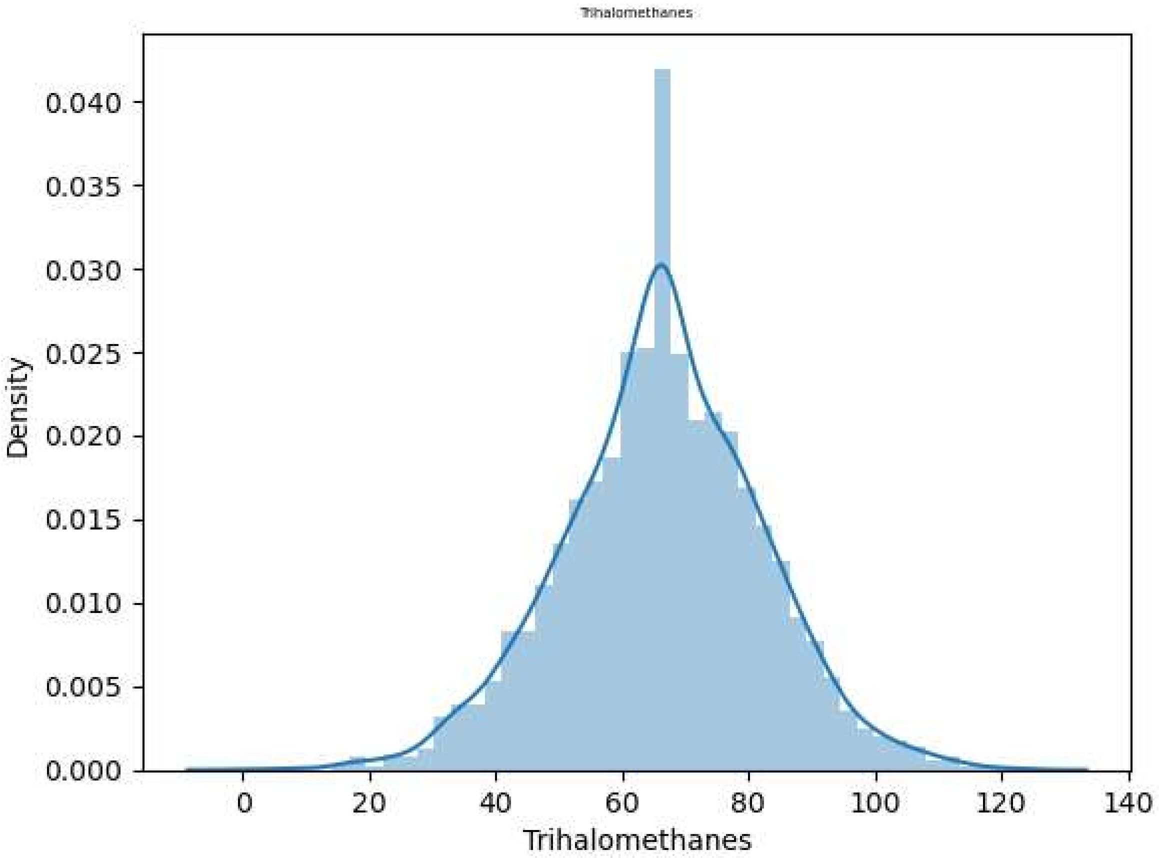
```
C:\Users\A to Z Infosys\anaconda3\lib\site-packages\seaborn\distributions.py:
2619: FutureWarning: `distplot` is a deprecated function and will be removed
in a future version. Please adapt your code to use either `displot` (a figure
-level function with similar flexibility) or `histplot` (an axes-level functi
on for histograms).
  warnings.warn(msg, FutureWarning)
```



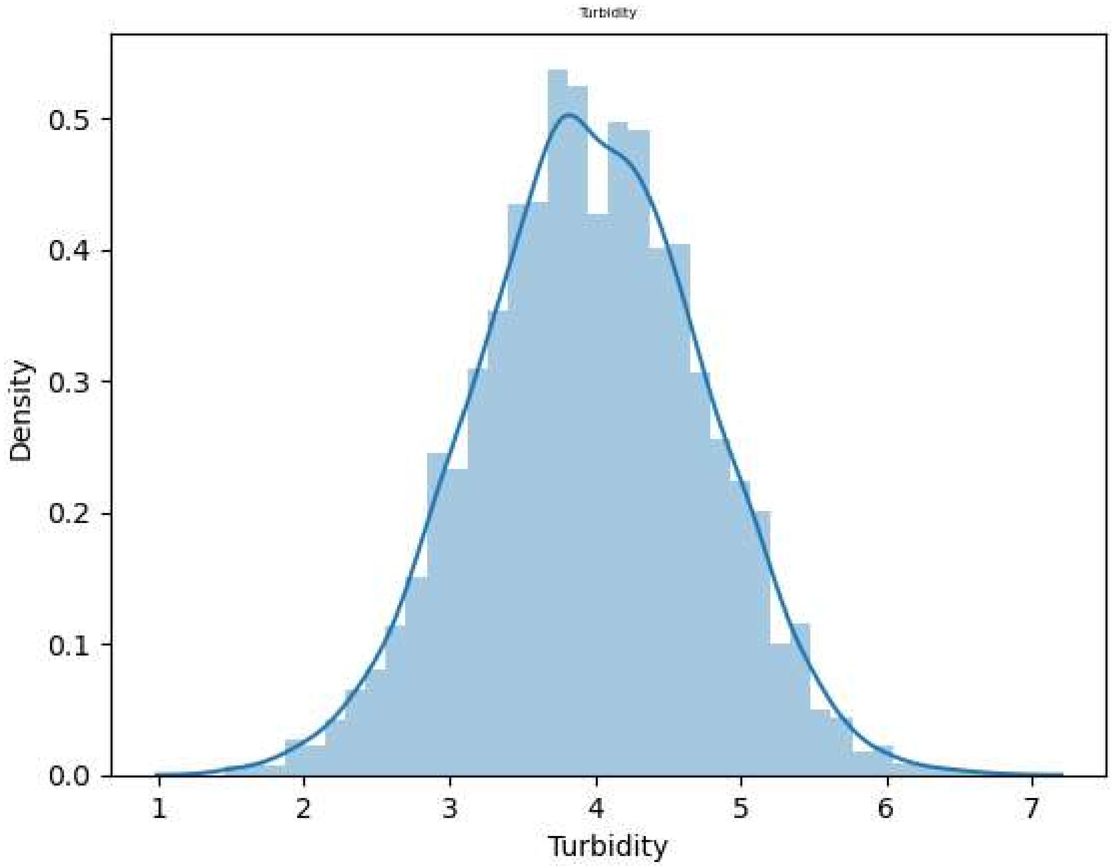
```
C:\Users\A to Z Infosys\anaconda3\lib\site-packages\seaborn\distributions.py:
2619: FutureWarning: `distplot` is a deprecated function and will be removed
in a future version. Please adapt your code to use either `displot` (a figure
-level function with similar flexibility) or `histplot` (an axes-level functi
on for histograms).
  warnings.warn(msg, FutureWarning)
```



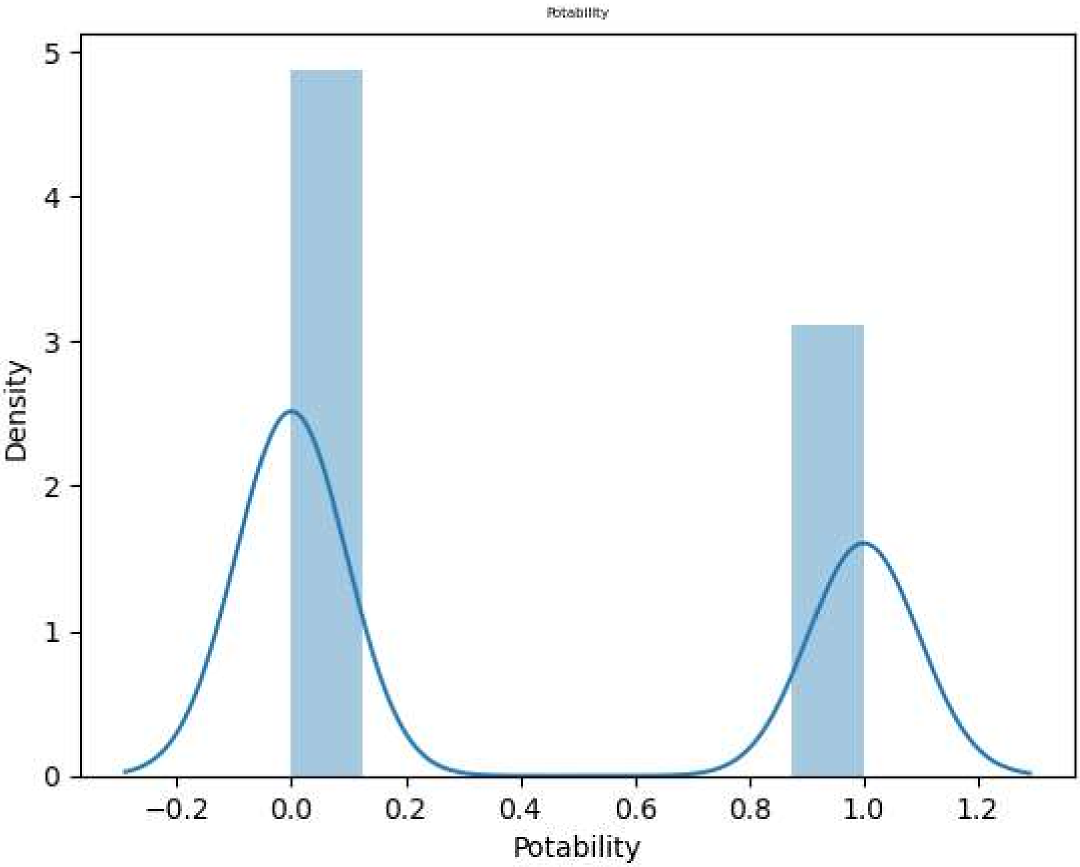
```
C:\Users\A to Z Infosys\anaconda3\lib\site-packages\seaborn\distributions.py:
2619: FutureWarning: `distplot` is a deprecated function and will be removed
in a future version. Please adapt your code to use either `displot` (a figure
-level function with similar flexibility) or `histplot` (an axes-level functi
on for histograms).
  warnings.warn(msg, FutureWarning)
```



```
C:\Users\A to Z Infosys\anaconda3\lib\site-packages\seaborn\distributions.py:
2619: FutureWarning: `distplot` is a deprecated function and will be removed
in a future version. Please adapt your code to use either `displot` (a figure
-level function with similar flexibility) or `histplot` (an axes-level functi
on for histograms).
  warnings.warn(msg, FutureWarning)
```



```
C:\Users\A to Z Infosys\anaconda3\lib\site-packages\seaborn\distributions.py:
2619: FutureWarning: `distplot` is a deprecated function and will be removed
in a future version. Please adapt your code to use either `displot` (a figure
-level function with similar flexibility) or `histplot` (an axes-level functi
on for histograms).
  warnings.warn(msg, FutureWarning)
```

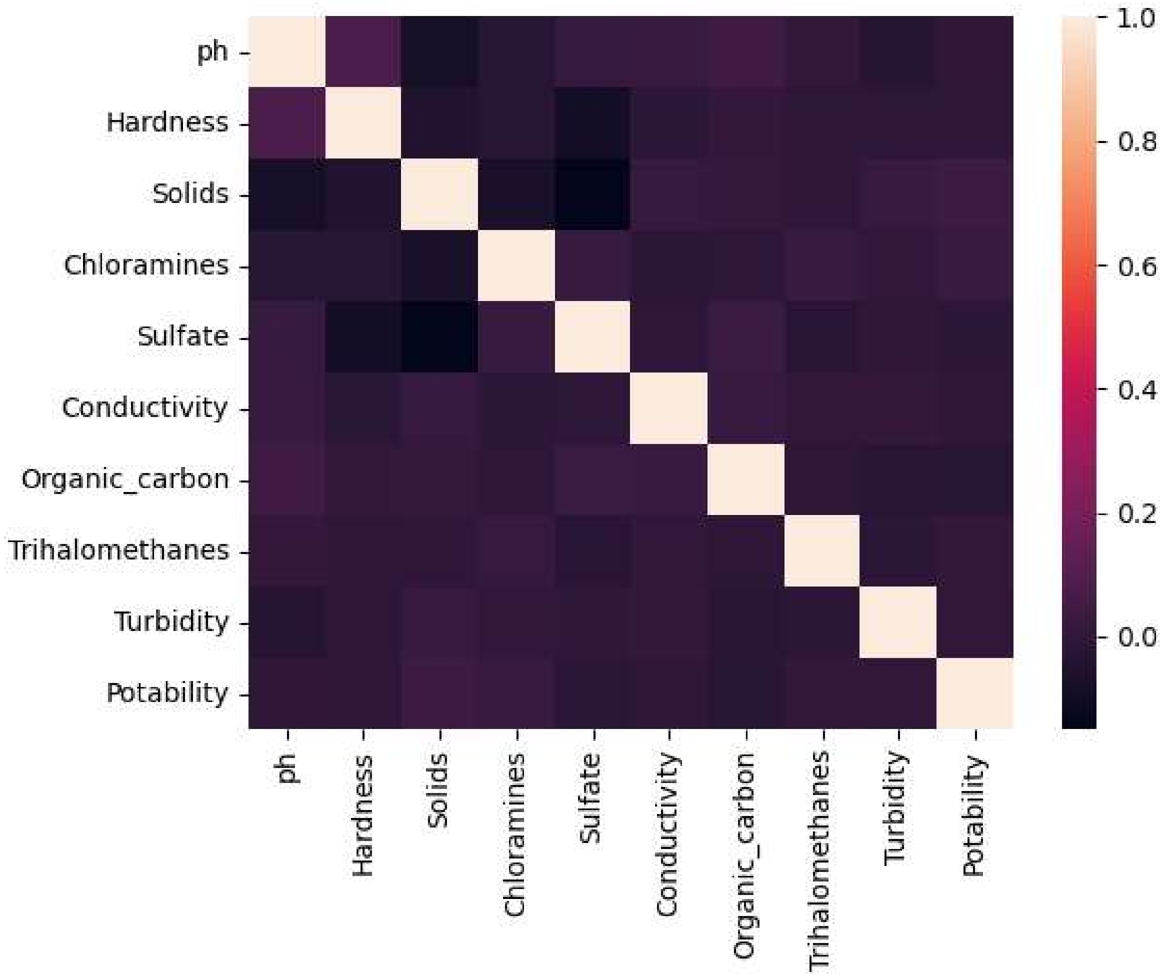

Potability
Density
0
1
2
3
4
5
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2
Potability

In [15]:

```
# checking the correlation of all the parameter,
# if correlation value is high than we can keep any one attribute
sns.heatmap(data.corr())
# most of the colors are darker means our data is less corelated with each oth
```

Out[15]:

```
<AxesSubplot:>
```



In [16]:

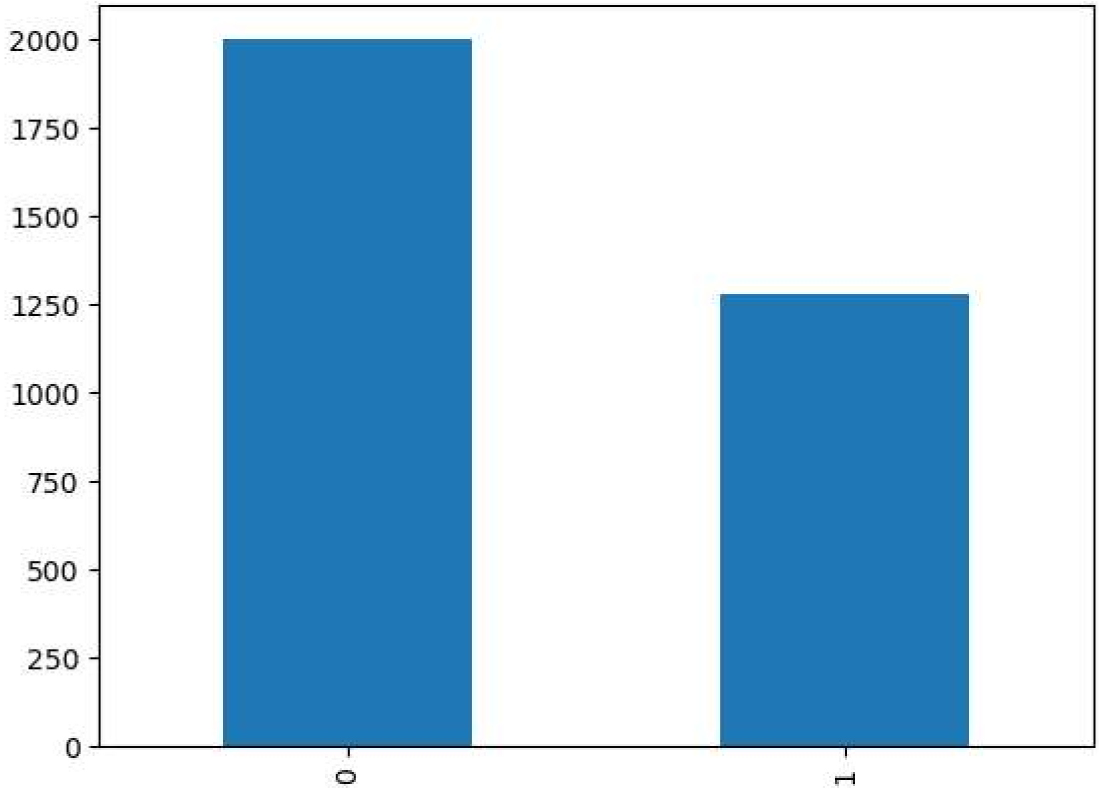
```
# Now checking the distribution of target column
data['Potability'].value_counts()
```

Out[16]:

```
0    1998
1    1278
Name: Potability, dtype: int64
```

In [17]:

```
data['Potability'].value_counts().plot(kind='bar')
plt.xticks(rotation='vertical')
plt.show()
```



In [18]:

```
x=data.drop(columns=['Potability'])
```

In [19]:

```
y=data['Potability']
```

In [20]:

```
# scaler=StandardScaler()
# x=scaler.fit_transform(x)
```

In [21]:

```
x
```

Out[21]:

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbo
0	7.080795	204.890455	20791.318981	7.300212	368.516441	564.308654	10.37978
1	3.716080	129.422921	18630.057858	6.635246	333.775777	592.885359	15.18001
2	8.099124	224.236259	19909.541732	9.275884	333.775777	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
...	...	...	...	...	...	...	
3271	4.668102	193.681735	47580.991603	7.166639	359.948574	526.424171	13.89441
3272	7.808856	193.553212	17329.802160	8.061362	333.775777	392.449580	19.90322
3273	9.419510	175.762646	33155.578218	7.350233	333.775777	432.044783	11.03907
3274	5.126763	230.603758	11983.869376	6.303357	333.775777	402.883113	11.16894
3275	7.874671	195.102299	17404.177061	7.509306	333.775777	327.459760	16.14036

3276 rows × 9 columns

Resampling of data (optinal)

In [22]:

```
y.value_counts()
```

Out[22]:

```
0    1998
1    1278
Name: Potability, dtype: int64
```

In [23]:

```
# Now we are doing data oversampling- means it make the equal number of sample
# this is optional step
from imblearn import over_sampling
from imblearn.over_sampling import RandomOverSampler
```

In [26]:

```
ros=RandomOverSampler(random_state=0)
x_resampled,y_resampled=ros.fit_resample(x,y)
```

In [27]:

```
# x_resampled['Potability'].value_counts()
```

In [28]:
```
x_resampled
```

Out[28]:

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbo
0	7.080795	204.890455	20791.318981	7.300212	368.516441	564.308654	10.37978
1	3.716080	129.422921	18630.057858	6.635246	333.775777	592.885359	15.18001
2	8.099124	224.236259	19909.541732	9.275884	333.775777	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
...	...	...	...	...	...	...	
3991	5.913755	175.326062	12044.624691	8.368785	347.880372	380.967166	12.53082
3992	7.672910	152.878305	22989.351184	6.231913	343.439017	401.140879	14.25268
3993	7.080795	129.883297	8906.975623	6.827901	327.551789	525.224717	17.65741
3994	7.226593	207.832229	22097.413223	5.862717	354.708382	353.654046	10.40105
3995	7.080795	182.317256	30430.211752	6.151217	350.448584	479.957076	16.57656

3996 rows × 9 columns

In [29]:
```
y_resampled
```

Out[29]:
```
0       0
1       0
2       0
3       0
4       0
       ..
3991    1
3992    1
3993    1
3994    1
3995    1
Name: Potability, Length: 3996, dtype: int64
```

In [30]:
```
r_data=pd.concat([x_resampled,y_resampled],axis=1) #our data is ready with res
```

In [31]:
```
# r_data.duplicated().sum()
# as it adds the duplicate value we won't use sampling
```

Data splitting and Model Trainig

In [32]:

```
# Now it time to start Model training and we are using four models
# i)Random Forest
# ii)DecisionTree
# iii)AdaBoost
# iv)Naive Bayes
#v) Logistic Regression
#vi) K Nearest Neighbour
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, confusion_matrix

# models
from sklearn.ensemble import RandomForestClassifier,AdaBoostClassifier
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.neighbors import KNeighborsClassifier
```

In [33]:

```
All_model=[

    ("Random Forest",RandomForestClassifier(random_state=10)),
#     ("Linear Regression",LinearRegression()),
    ("Decision Tree", DecisionTreeClassifier(random_state=42)),
    ("Naive Bayed",  GaussianNB()),
     ("Ada Boost",AdaBoostClassifier(random_state=42)),
    ("K Neighbour Classifier",KNeighborsClassifier(n_neighbors=2))
]
```

In [45]:

```
# X=data.drop(columns='Potability')
X=r_data.drop(columns='Potability')
# Y=data['Potability']
Y=r_data['Potability']
```

In [46]:
```
X
```

Out[46]:
```
            ph    Hardness        Solids  Chloramines     Sulfate  Conductivity  Organic_carbo
   0  7.080795  204.890455  20791.318981     7.300212  368.516441    564.308654       10.3797
   1  3.716080  129.422921  18630.057858     6.635246  333.775777    592.885359       15.18001
   2  8.099124  224.236259  19909.541732     9.275884  333.775777    418.606213       16.8686
   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
 ...       ...         ...           ...          ...         ...           ...
3991  5.913755  175.326062  12044.624691     8.368785  347.880372    380.967166       12.53082
3992  7.672910  152.878305  22989.351184     6.231913  343.439017    401.140879       14.2526
3993  7.080795  129.883297   8906.975623     6.827901  327.551789    525.224717       17.6574
3994  7.226593  207.832229  22097.413223     5.862717  354.708382    353.654046       10.4010
3995  7.080795  182.317256  30430.211752     6.151217  350.448584    479.957076       16.5765

3996 rows × 9 columns
```

In [47]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, rando
```

In [48]:
```
# Creating new dataframe to store the accuracy wrt to model used
accuracy_table=pd.DataFrame(columns=["Algo", "Accuracy"])
```

In [49]:

```
for i,j in All_model:
    print(j)
    print(i)
    j.fit(X_train,Y_train)
    predictions=j.predict(X_test)
    acc=accuracy_score(Y_test, predictions)
    temp= {"Algo": i, "Accuracy": acc}
    accuracy_table=accuracy_table.append(temp, ignore_index=True)
```

```
RandomForestClassifier(random_state=10)
Random Forest

C:\Users\A to Z Infosys\AppData\Local\Temp\ipykernel_14280\1411955508.py:8: F
utureWarning: The frame.append method is deprecated and will be removed from
pandas in a future version. Use pandas.concat instead.
  accuracy_table=accuracy_table.append(temp, ignore_index=True)
C:\Users\A to Z Infosys\AppData\Local\Temp\ipykernel_14280\1411955508.py:8: F
utureWarning: The frame.append method is deprecated and will be removed from
pandas in a future version. Use pandas.concat instead.
  accuracy_table=accuracy_table.append(temp, ignore_index=True)
C:\Users\A to Z Infosys\AppData\Local\Temp\ipykernel_14280\1411955508.py:8: F
utureWarning: The frame.append method is deprecated and will be removed from
pandas in a future version. Use pandas.concat instead.
  accuracy_table=accuracy_table.append(temp, ignore_index=True)

DecisionTreeClassifier(random_state=42)
Decision Tree
GaussianNB()
Naive Bayed
AdaBoostClassifier(random_state=42)
Ada Boost
KNeighborsClassifier(n_neighbors=2)
K Neighbour Classifier

C:\Users\A to Z Infosys\AppData\Local\Temp\ipykernel_14280\1411955508.py:8: F
utureWarning: The frame.append method is deprecated and will be removed from
pandas in a future version. Use pandas.concat instead.
  accuracy_table=accuracy_table.append(temp, ignore_index=True)
C:\Users\A to Z Infosys\AppData\Local\Temp\ipykernel_14280\1411955508.py:8: F
utureWarning: The frame.append method is deprecated and will be removed from
pandas in a future version. Use pandas.concat instead.
  accuracy_table=accuracy_table.append(temp, ignore_index=True)
```

In [50]:

```
accuracy_table.sort_values(by='Accuracy')
```

Out[50]:

	Algo	Accuracy
2	Naive Bayed	0.55875
3	Ada Boost	0.55875
4	K Neighbour Classifier	0.60125
1	Decision Tree	0.70125
0	Random Forest	0.7775

In [51]:
```
# if do the same using StanderScaler
scaler = StandardScaler()
X = scaler.fit_transform(X)
```

In [52]:
```
X
```

Out[52]:
```
array([[-0.00391605,  0.25956782, -0.15738462, ..., -1.18098616,
         1.28842112, -1.28102328],
       [-2.33332377, -2.01696775, -0.40100535, ...,  0.27773111,
        -0.63944346,  0.67326663],
       [ 0.70107841,  0.8431487 , -0.25677993, ...,  0.79087833,
        -0.00497148, -1.16307046],
       ...,
       [-0.00391605, -2.00308016, -1.49700636, ...,  1.03057479,
        -2.1574769 , -0.16378381],
       [ 0.09702074,  0.34830866, -0.01015964, ..., -1.17452155,
         0.022948  ,  0.46539736],
       [-0.00391605, -0.4213699 ,  0.92912642, ...,  0.70212205,
        -0.49193949,  0.71834676]])
```

In [42]:

```python
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, rando

accuracy_table2=pd.DataFrame(columns=["Algo2", "Accuracy2"])

for i,j in All_model:
    print(j)
    print(i)
    j.fit(X_train,Y_train)
    predictions=j.predict(X_test)
    acc=accuracy_score(Y_test, predictions)
    temp= {"Algo2": i, "Accuracy2": acc}
    accuracy_table2=accuracy_table2.append(temp, ignore_index=True)
```

```
RandomForestClassifier(random_state=10)
Random Forest

C:\Users\A to Z Infosys\AppData\Local\Temp\ipykernel_14280\2942780521.py:12:
FutureWarning: The frame.append method is deprecated and will be removed from
pandas in a future version. Use pandas.concat instead.
  accuracy_table2=accuracy_table2.append(temp, ignore_index=True)
C:\Users\A to Z Infosys\AppData\Local\Temp\ipykernel_14280\2942780521.py:12:
FutureWarning: The frame.append method is deprecated and will be removed from
pandas in a future version. Use pandas.concat instead.
  accuracy_table2=accuracy_table2.append(temp, ignore_index=True)
C:\Users\A to Z Infosys\AppData\Local\Temp\ipykernel_14280\2942780521.py:12:
FutureWarning: The frame.append method is deprecated and will be removed from
pandas in a future version. Use pandas.concat instead.
  accuracy_table2=accuracy_table2.append(temp, ignore_index=True)

DecisionTreeClassifier(random_state=42)
Decision Tree
GaussianNB()
Naive Bayed
AdaBoostClassifier(random_state=42)
Ada Boost
KNeighborsClassifier(n_neighbors=2)
K Neighbour Classifier

C:\Users\A to Z Infosys\AppData\Local\Temp\ipykernel_14280\2942780521.py:12:
FutureWarning: The frame.append method is deprecated and will be removed from
pandas in a future version. Use pandas.concat instead.
  accuracy_table2=accuracy_table2.append(temp, ignore_index=True)
C:\Users\A to Z Infosys\AppData\Local\Temp\ipykernel_14280\2942780521.py:12:
FutureWarning: The frame.append method is deprecated and will be removed from
pandas in a future version. Use pandas.concat instead.
  accuracy_table2=accuracy_table2.append(temp, ignore_index=True)
```

In [43]:
```
accuracy_table2.sort_values(by='Accuracy2')
```

Out[43]:

	Algo2	Accuracy2
1	Decision Tree	0.577744
3	Ada Boost	0.620427
2	Naive Bayed	0.631098
4	K Neighbour Classifier	0.660061
0	Random Forest	0.679878

In [57]:
```
# Next part we do Hyper Paramenter Tuning to improve our efficiency
param_grid_rfc = {"min_samples_split": [2, 3, 10],
                  "min_samples_leaf": [1, 3, 10],
                  "n_estimators" :[100, 200, 500],
                  "random_state": [42]}

grid_rfc = GridSearchCV(RandomForestClassifier(), param_grid_rfc, scoring="acc

grid_rfc.fit(X_train, Y_train)
```

```
---------------------------------------------------------------------------
-
_RemoteTraceback                          Traceback (most recent call las
t)
_RemoteTraceback:
"""
Traceback (most recent call last):
  File "C:\Users\A to Z Infosys\anaconda3\lib\site-packages\joblib\externa
ls\loky\process_executor.py", line 428, in _process_worker
    r = call_item()
  File "C:\Users\A to Z Infosys\anaconda3\lib\site-packages\joblib\externa
ls\loky\process_executor.py", line 275, in __call__
    return self.fn(*self.args, **self.kwargs)
  File "C:\Users\A to Z Infosys\anaconda3\lib\site-packages\joblib\_parall
el_backends.py", line 620, in __call__
    return self.func(*args, **kwargs)
  File "C:\Users\A to Z Infosys\anaconda3\lib\site-packages\joblib\paralle
l.py", line 288, in __call__
    return [func(*args, **kwargs)
  File "C:\Users\A to Z Infosys\anaconda3\lib\site-packages\joblib\paralle
```

In [55]:
```
rfc_params = grid_rfc.best_params_
rfc = RandomForestClassifier(**rfc_params)
rfc.fit(X_train, Y_train)
predictions = rfc.predict(X_test)
score = accuracy_score(Y_test, predictions)
print("Accuracy Score:", score)
```

```
Accuracy Score: 0.78
```

In [ ]: