ABSTRACT

The project involves analyzing water quality data to assess the suitability of water for specific purposes, such as drinking. The objective is to identify potential issues or deviations from regulatory standards and determine water potability based on various parameters. This project includes defining analysis objectives, collecting water quality data, designing relevant visualizations, and building a predictive model.

1. **Define Analysis Objectives:**

- Clearly articulate the specific goals and objectives of the analysis. For example, you may want to assess if the water meets regulatory standards for drinking purposes, agricultural use, recreational activities, etc.

2. **Collect Water Quality Data:**

- Gather relevant data from reliable sources. This may include information on chemical, physical, biological, and radiological parameters. Ensure the data covers a sufficient time period and is representative of the area in question.

3. **Data Preprocessing:**

- Clean and prepare the data for analysis. This involves tasks like handling missing values, outliers, and normalizing or standardizing data if necessary.

4. **Exploratory Data Analysis (EDA):**

- Perform exploratory data analysis to gain insights into the dataset. This could involve summary statistics, visualizations, and correlation analysis to understand the relationships between different variables.

5. **Regulatory Standards and Guidelines:**

- Familiarize yourself with the relevant regulatory standards and guidelines for water quality. This will be crucial for assessing the data against established benchmarks.

6. **Data Visualization:**

ABSTRACT

- Create visualizations to present the water quality data effectively. This could include plots, charts, and maps to highlight any deviations from standards.

7. **Model Building:**

- Depending on the complexity of your project, you may want to build a predictive model. This could involve using techniques like regression, classification, or machine learning algorithms to predict water quality based on specific parameters.

8. **Model Evaluation and Validation:**

- Assess the performance of your predictive model using appropriate metrics. Cross-validation and other techniques may be used to ensure the model's reliability.

9. **Interpret Results:**

- Interpret the results of your analysis in the context of your defined objectives. Highlight any potential issues, deviations from standards, or areas of concern.

10. **Recommendations and Reporting:**

- Based on your analysis, provide recommendations for any necessary actions or interventions. This could include suggestions for water treatment, policy changes, or further monitoring.

11. **Documentation:**

- Document your methodology, data sources, analysis techniques, and results. This will be important for transparency, replication, and future reference.

12. **Communication:**

- Present your findings in a clear and understandable manner, tailored to your target audience. This could involve creating reports, presentations, or visual dashboards.

ABSTRACT

Remember to stay updated with any changes in regulatory standards and consider consulting with experts in the field of water quality if needed. Additionally, ethical considerations and environmental impact should also be taken into account throughout the project.

NAAN MUDHALVAN PROJECT PHASE 2: **INNOVATION**

PROJECT TITLE: **WATER QUALITY ANALYSIS**

BY: **SANTHOSH V**

DATA ANALYTICS OF WATER QUALITY ANALYSIS

VALUABLE INNOVATION STEPS:

STEP1: DATA COLLECTIONS

- Review the initial design concept to ensure it aligns with the identified problem.
- Gather feedback from stakeholders and subject matter experts for improvements.
- Incorporate necessary changes to enhance the design's effectiveness.

STEP2: CLEANING DATA

- Clean and reprocess the data to remove outliers, errors, and inconsistencies.
- Ensure data quality before analysis.
- Transformation data into proper format for further processes

STEP3: EVALUATE AND ANAYLSIS

Machine Learning Models:

- Train machine learning models to predict water quality based on chemical components present in water.
- This can help pinpoint potential sources of water quality.

Cluster Analysis:

- Use clustering algorithms to group similar noise patterns together.
- This can help identify areas with distinct noise characteristics.

STEP4: DATA VISUALIZATION

- Create interactive maps and visualizations to communicate water quality patterns.
- Finding components present on water such as PH, sodium, carbon, hydrogen etc..

STEP5: DISCRIBE RESULT (COMMUNICATOIN)

- Result is predicted by using appropriate calculation method using statistic evaluation as per our data collections.
- Finally result of water quality is predicted.

WATER QUALITY ANALYSIS DESIGN

NAAN MUDHALVAN PROJECT PHASE 3: **Development part 1**

PROJECT TITLE: **WATER QUALITY ANALYSIS**

BY: **SANTHOSH V**

DATA ANALYTICS OF WATER QUALITY ANALYSIS

Preprocess the given data sets and visualize :

Give Data sets before cleaning processes:

	A	B	C	D	E	F	G	H	I	J
1	ph	Hardness	Solids	Chloramir	Sulfate	Conductiv	Organic_c	Trihalome	Turbidity	Potability
2		204.8905	20791.32	7.300212	368.5164	564.3087	10.37978	86.99097	2.963135	0
3	3.7160801	129.4229	18630.06	6.635246		592.8854	15.18001	56.32908	4.500656	0
4	8.0991242	224.2363	19909.54	9.275884		418.6062	16.86864	66.42009	3.055934	0
5	8.3167659	214.3734	22018.42	8.059332	356.8861	363.2665	18.43652	100.3417	4.628771	0
6	9.0922235	181.1015	17978.99	6.5466	310.1357	398.4108	11.55828	31.99799	4.075075	0
7	5.5840866	188.3133	28748.69	7.544869	326.6784	280.4679	8.399735	54.91786	2.559708	0
8	10.223862	248.0717	28749.72	7.513408	393.6634	283.6516	13.7897	84.60356	2.672989	0
9	8.6358487	203.3615	13672.09	4.563009	303.3098	474.6076	12.36382	62.79831	4.401425	0
10		118.9886	14285.58	7.804174	268.6469	389.3756	12.70605	53.92885	3.595017	0
11	11.180284	227.2315	25484.51	9.0772	404.0416	563.8855	17.92781	71.9766	4.370562	0
12	7.3606401	165.5208	32452.61	7.550701	326.6244	425.3834	15.58681	78.74002	3.662292	0
13	7.9745216	218.6933	18767.66	8.110385		364.0982	14.52575	76.48591	4.011718	0
14	7.1198244	156.705	18730.81	3.606036	282.3441	347.715	15.92954	79.50078	3.445756	0
15		150.1749	27331.36	6.838223	299.4158	379.7618	19.37081	76.51	4.413974	0
16	7.4962322	205.345	28388	5.072558		444.6454	13.22831	70.30021	4.777382	0
17	6.3472718	186.7329	41065.23	9.629596	364.4877	516.7433	11.53978	75.07162	4.376348	0
18	7.0517858	211.0494	30980.6	10.0948		315.1413	20.39702	56.6516	4.268429	0
19	9.18156	273.8138	24041.33	6.90499	398.3505	477.9746	13.38734	71.45736	4.503661	0
20	8.9754643	279.3572	19460.4	6.204321		431.444	12.88876	63.82124	2.436086	0
21	7.3710503	214.4966	25630.32	4.432669	335.7544	469.9146	12.50916	62.79728	2.560299	0
22		227.435	22305.57	10.33392		554.8201	16.33169	45.38282	4.133423	0
23	6.660212	168.2837	30944.36	5.858769	310.9309	523.6713	17.88424	77.04232	3.749701	0
24		215.9779	17107.22	5.60706	326.944	436.2562	14.18906	59.85548	5.459251	0
25	3.9024757	196.9032	21167.5	6.996312		444.4789	16.60903	90.18168	4.528523	0
26	5.4003018	140.7391	17266.59	10.05685	328.3582	472.8741	11.25638	56.93191	4.824786	0
27	6.5144151	198.7674	21218.7	8.670937	323.5963	413.2905	14.9	79.84784	5.200885	0

In this part we will begin building your project by loading:

```
import pandas as pd

file_path = "/content/water_potability.csv"

data = pd.read_csv(file_path)

file_path = "/content/water_potability.csv" data = pd.read_csv(file_path)

print(data.head())
```

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

   Organic_carbon  Trihalomethanes  Turbidity  Potability
0       10.379783        86.990970   2.963135           0
1       15.180013        56.329076   4.500656           0
2       16.868637        66.420093   3.055934           0
3       18.436524       100.341674   4.628771           0
4       11.558279        31.997993   4.075075           0
```

```
selected_columns=data[['ph','Hardness','Solids','Chloramines']]

print(selected_columns)
```

```
          ph    Hardness        Solids  Chloramines
0        NaN  204.890455  20791.318981     7.300212
```

1	3.716080	129.4229	18630.0578	6.635246
2	8.099124	224.2362	19909.5417	9.275884
3	8.316766	214.3733	22018.4174	8.059332
4	9.092223	181.1015	17978.9863	6.546600
…	…	…	…	…
3271	4.668102	193.6817	47580.9916	7.166639
3272	7.808856	193.5532	17329.8021	8.061362
3273	9.419510	175.7626	33155.5782	7.350233
3274	5.126763	230.6037	11983.8693	6.303357
3275	7.874671	195.102299	17404.177061	7.509306

[3276 rows x 4 columns]

In this part we begin with the preprocessing odd dataset:

```
import pandas as pd

from sklearn.model_selection import train_test_split from sklearn.preprocessing import  StandardScaler

from sklearn.preprocessing import LabelEncoder

data  =  pd.read_csv('/content/water_potability.csv')

data['ph'].fillna(data['ph'].mean(),  inplace=True)

label_encoder = LabelEncoder()

data['Hardness']  =  label_encoder.fit_transform(data['Hardness'])

X  =  data.drop('Potability',  axis=1) y  =  data['Potability']

X  =  data.drop('Potability',  axis=1) y  =  data['Potability']



scaler = StandardScaler()

X_train = scaler.fit_transform(X_train) X_test = scaler.transform(X_test)
```

In this part we begins with the exploratory data analysis:

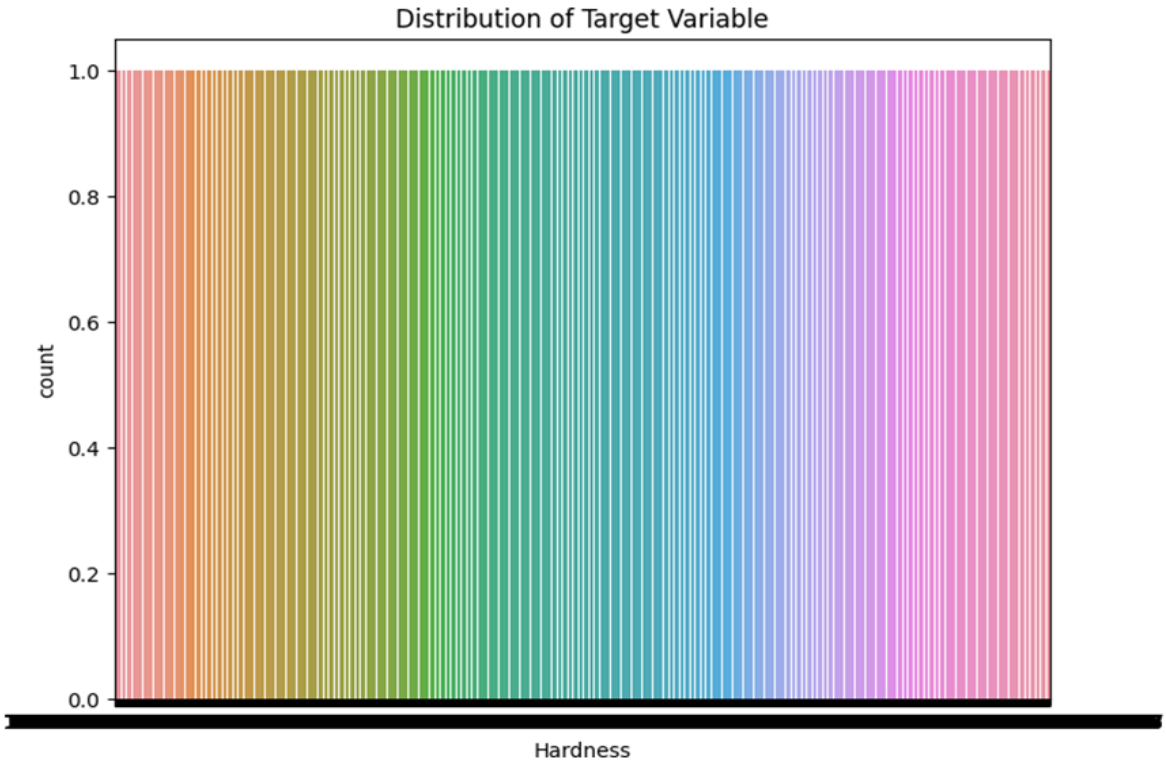
```
import pandas as pd

 import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

data  =  pd.read_csv('/content/water_potability.csv')

print("First 5 rows of the dataset:") print(data.head())


print(data.describe())
```

	ph	Hardness	Solids	Chloramines	Sulfate\
count	2785.000000	3276.000000	3276.000000	3276.000000	2495.000000
mean	7.080795	196.369496	22014.092526	7.122277	333.775777
std	1.594320	32.879761	8768.570828	1.583085	41.416840
min	0.000000	47.432000	320.942611	0.352000	129.000000
25%	6.093092	176.850538	15666.690297	6.127421	307.699498
50%	7.036752	196.967627	20927.833607	7.130299	333.073546
75%	8.062066	216.667456	27332.762127	8.114887	359.950170
max	14.000000	323.124000	61227.196008	13.127000	481.030642

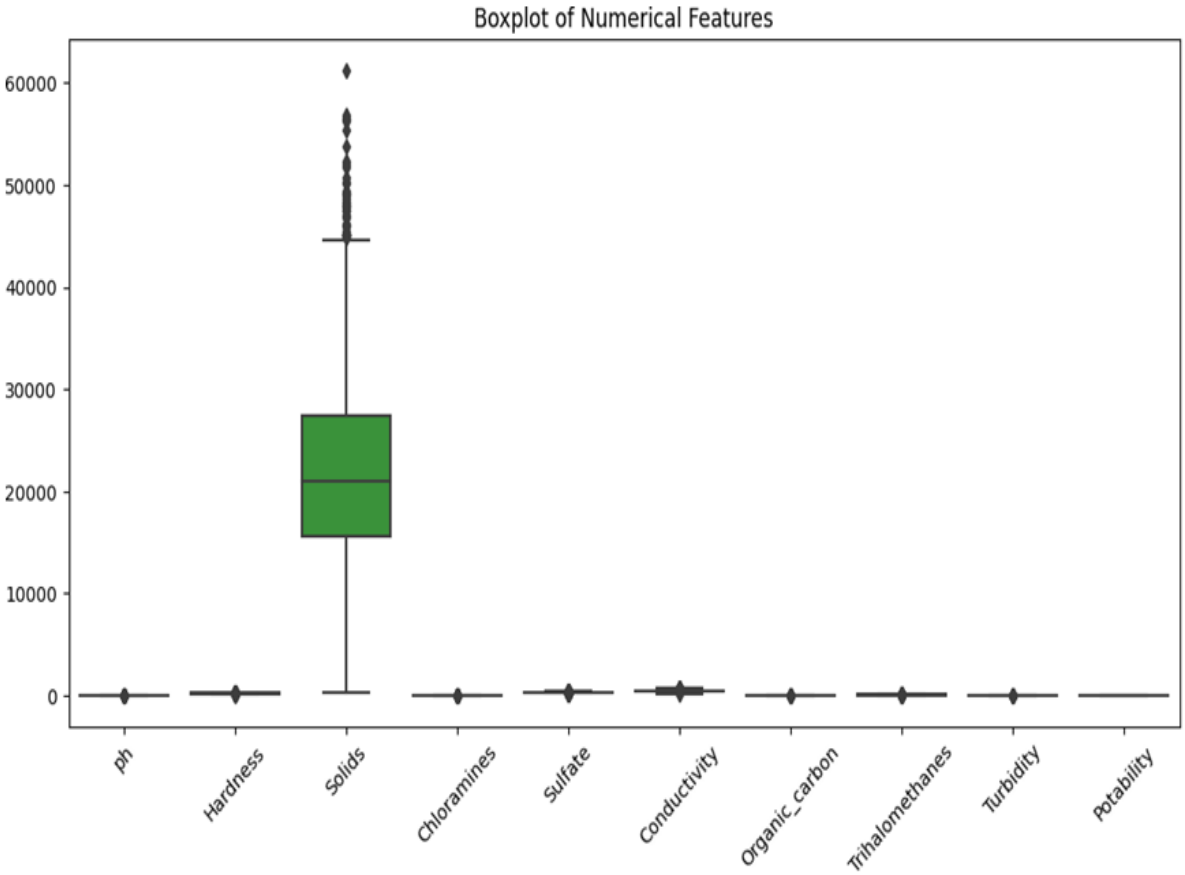
	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
count	3276.00000	3276.00000	3114.00000	3276.00000	3276.00000

	0	0	0	0	0
mean	426.205111	14.284970	66.396293	3.966786	0.390110
std	80.824064	3.308162	16.175008	0.780382	0.487849
min	181.483754	2.200000	0.738000	1.450000	0.000000
25%	365.734414	12.065801	55.844536	3.439711	0.000000
50%	421.884968	14.218338	66.622485	3.955028	0.000000
75%	481.792304	16.557652	77.337473	4.500320	1.000000
max	753.342620	28.300000	124.000000	6.739000	1.000000

```
plt.figure(figsize=(8, 6))

sns.countplot(x='Hardness', data=data)

plt.title('Distribution of Target Variable')

plt.show()
```



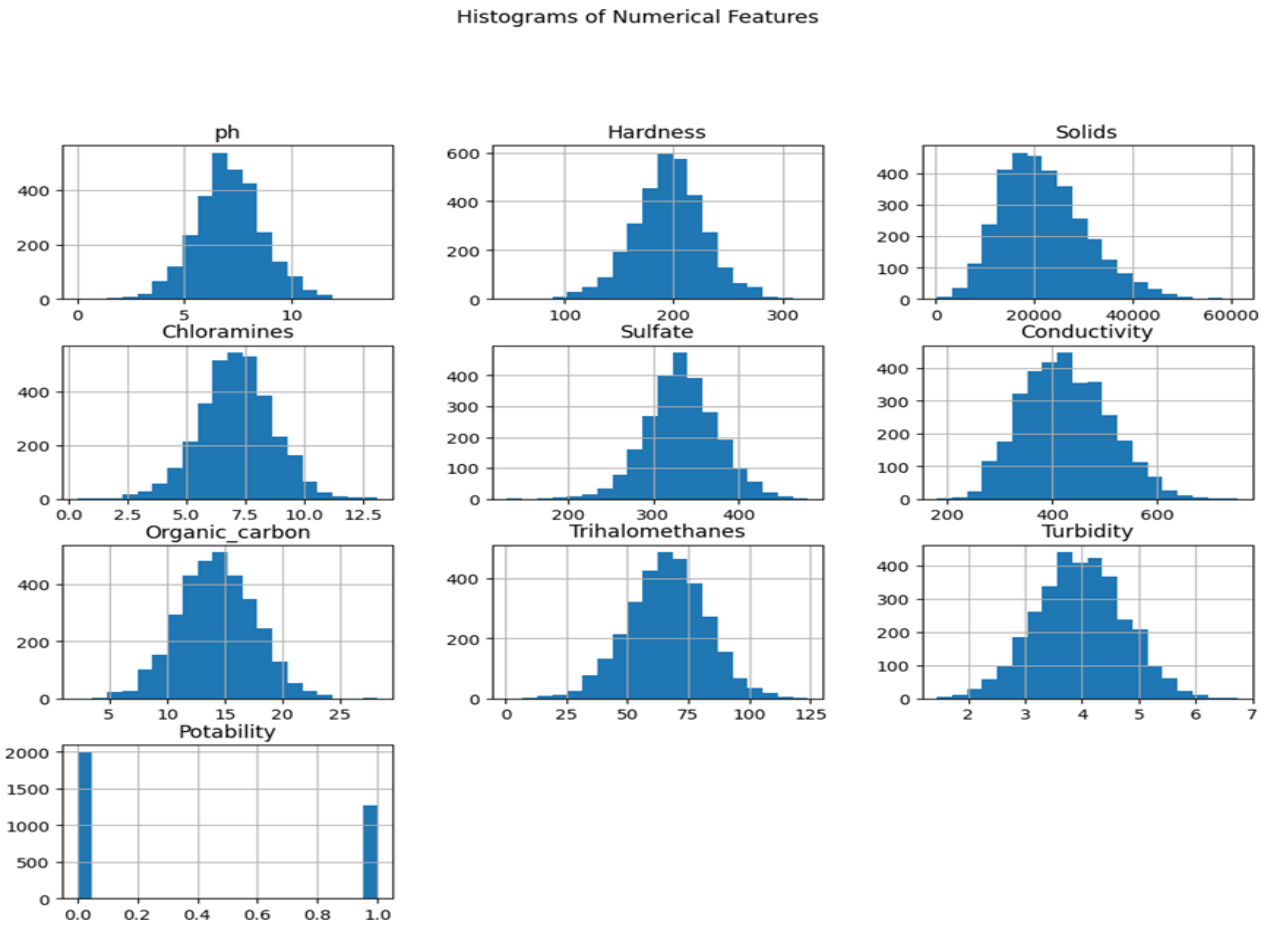

```
plt.figure(figsize=(12, 6))

sns.boxplot(data=data.select_dtypes(include=['float64', 'int64']))

plt.title('Boxplot of Numerical Features')

plt.xticks(rotation=45)

 plt.show()
```



data.select_dtypes(include=['float64', 'int64']).hist(bins=20, figsize=(12, 10))

plt.suptitle("Histograms of Numerical Features", y=1.02)

plt.show()



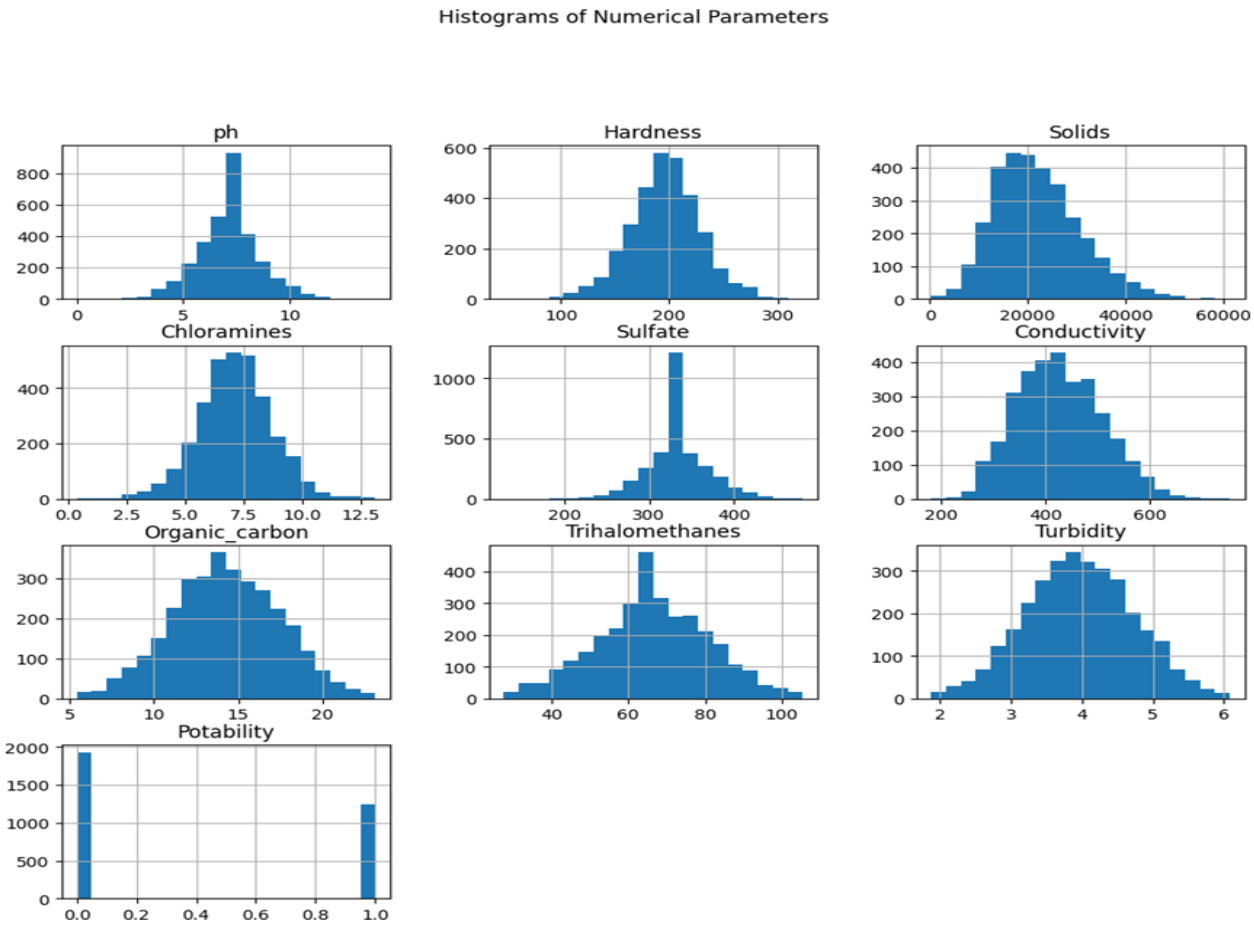
for column in numeric_columns:

data = handle_outliers_iqr(data, column)

Visualize Parameter Distributions

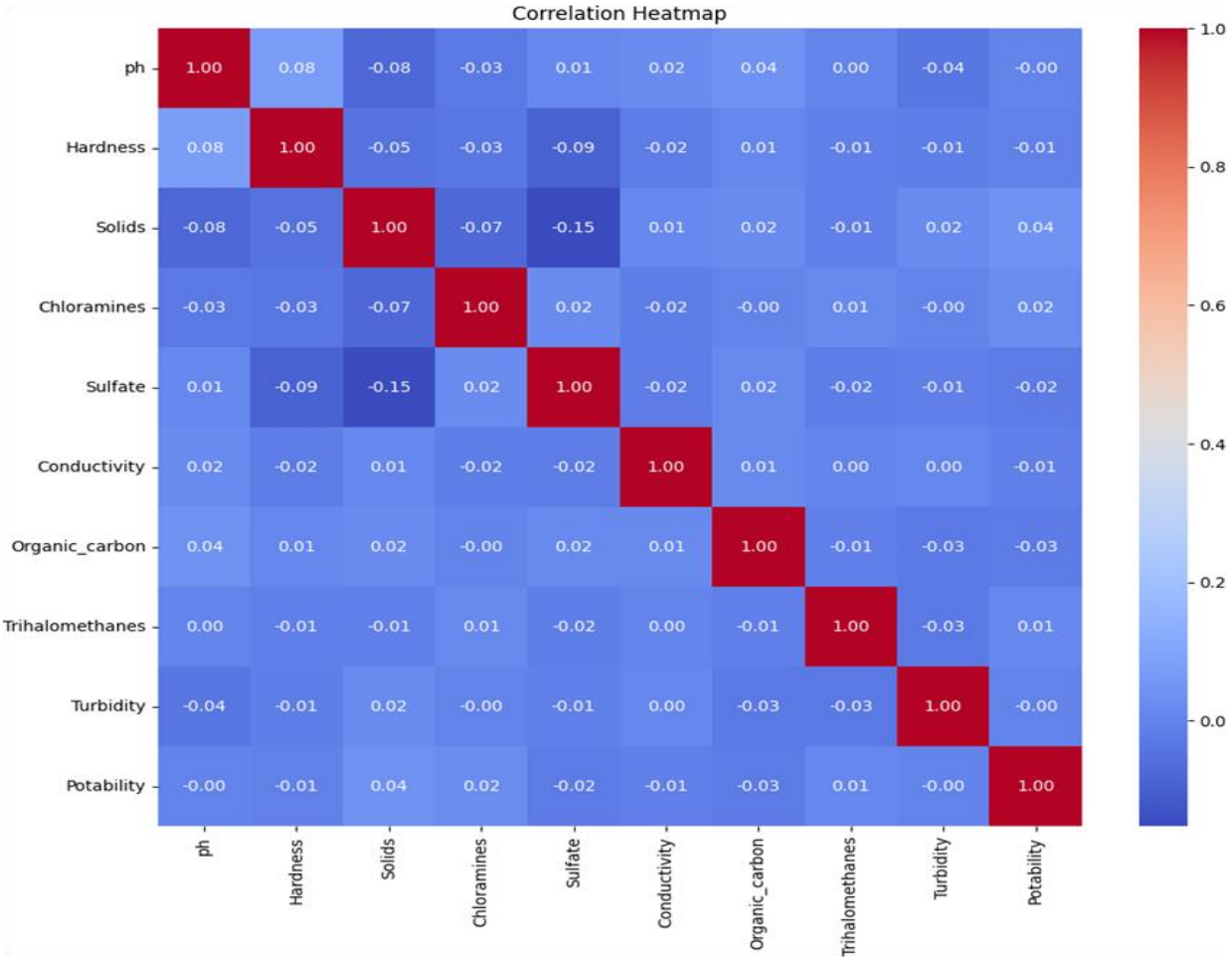
data.select_dtypes(include=['float64', 'int64']).hist(bins=20, figsize=(12, 10)) plt.suptitle("Histograms of Numerical Parameters", y=1.02)

plt.show()



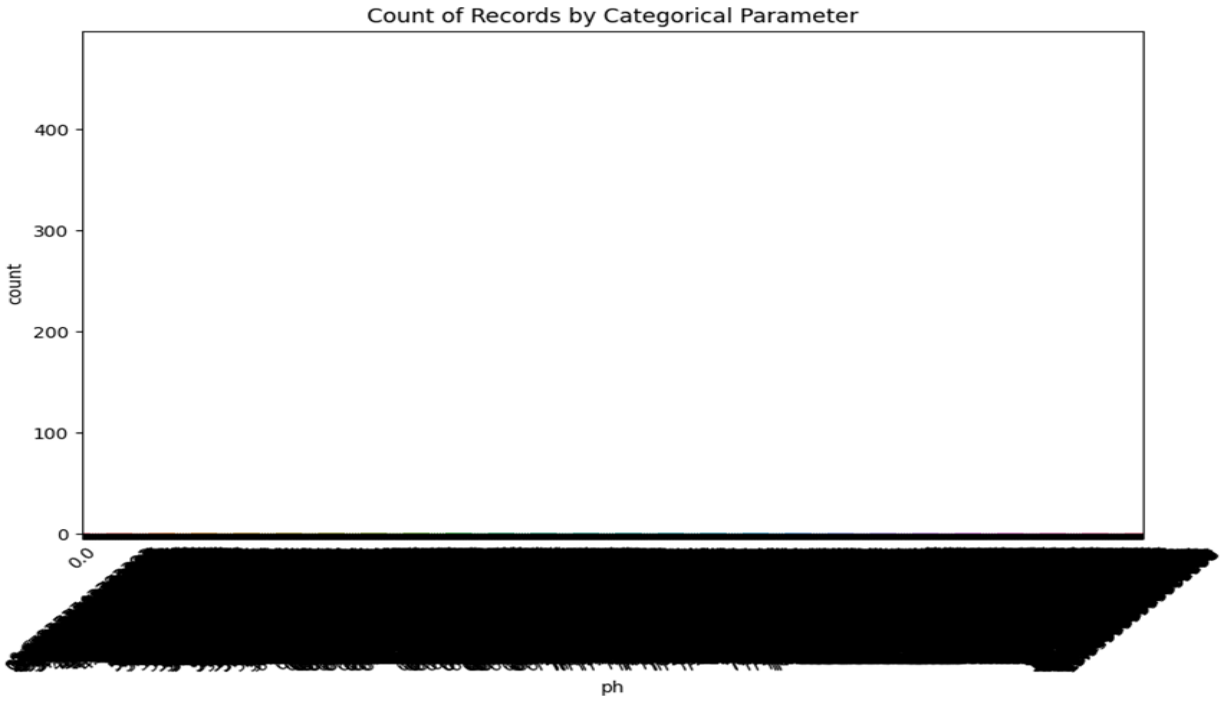
Visualize Correlations

```
correlation_matrix = data.corr() # Plot correlation heatmap

plt.figure(figsize=(12,  10))

sns.heatmap(correlation_matrix,  annot=True,  cmap='coolwarm',  fmt=".2f") plt.title('Correlation

Heatmap')

plt.show()
```



Identify Potential Deviations from Standards

```
plt.figure(figsize=(10, 6)) sns.countplot(x='ph', data=data)

plt.title('Count of Records by Categorical Parameter')

plt.xticks(rotation=45)

plt.show()
```


Count of Records by Categorical Parameter
400
300
count
200
100
0
0.0
ph

NAAN MUDHALVAN PROJECT PHASE 4: **Development part 2**

PROJECT TITLE: **WATER QUALITY ANALYSIS**

DATA ANALYTICS OF WATER QUALITY ANALYSIS

creating visualizations and building a predictive model:

```
import pandas as pd

file_path = "/content/water_potability.csv"

data = pd.read_csv(file_path)

file_path = "/content/water_potability.csv" data = pd.read_csv(file_path)

print(data.head())
```

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

     Organic_carbon  Trihalomethanes  Turbidity  Potability
0         10.379783        86.990970   2.963135           0
1         15.180013        56.329076   4.500656           0
2         16.868637        66.420093   3.055934           0
3         18.436524       100.341674   4.628771           0
4         11.558279        31.997993   4.075075           0
```

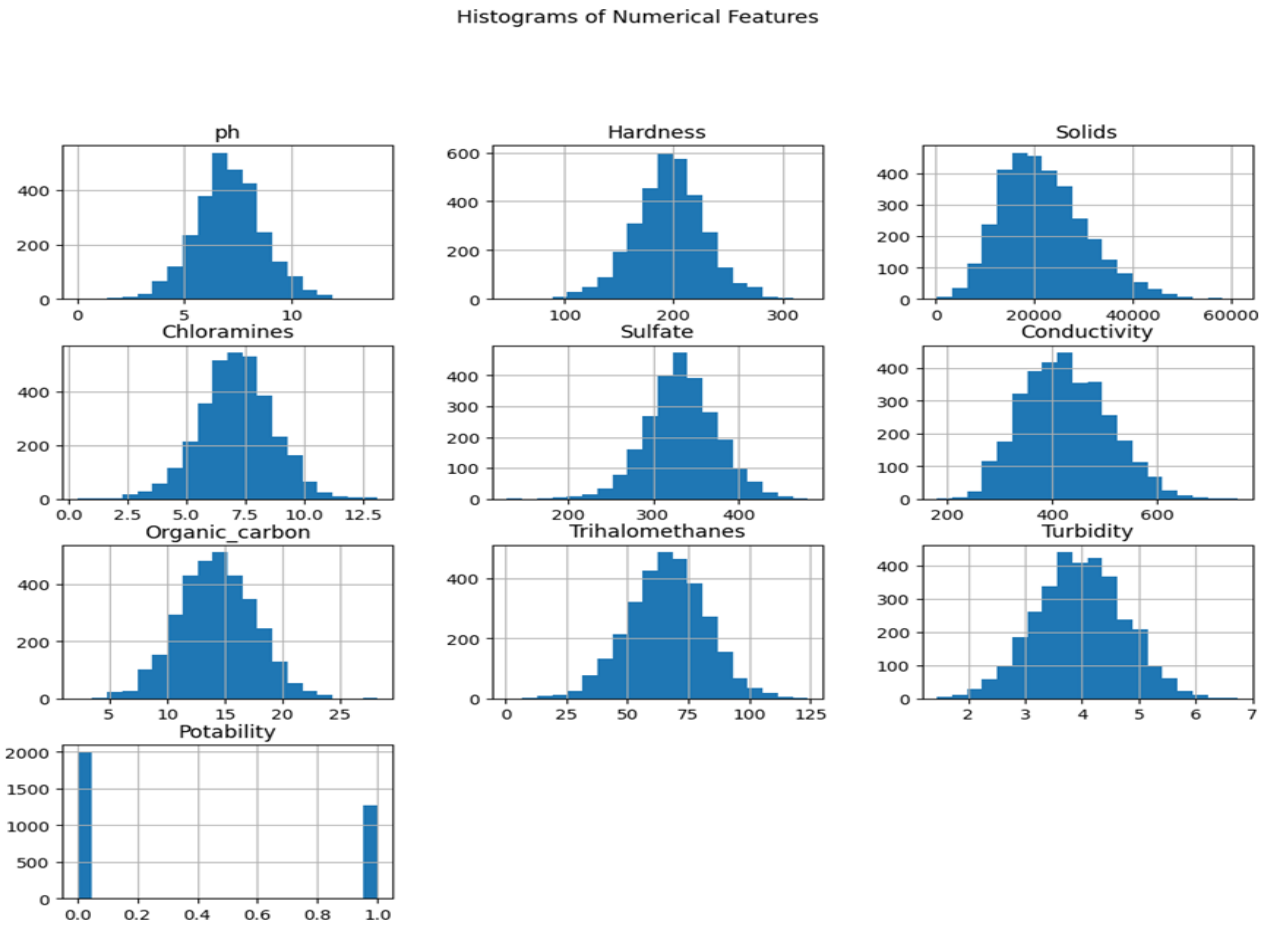
```
selected_columns=data[['ph','Hardness','Solids','Chloramines']]

print(selected_columns)
```

	ph	Hardness	Solids	Chloramines
0	NaN	204.890455	20791.318981	7.300212
1	3.716080	129.422921	18630.057858	6.635246
2	8.099124	224.236259	19909.541732	9.275884
3	8.316766	214.373394	22018.417441	8.059332
4	9.092223	181.101509	17978.986339	6.546600
…	…	…	…	…
3271	4.668102	193.681735	47580.991603	7.166639
3272	7.808856	193.553212	17329.802160	8.061362
3273	9.419510	175.762646	33155.578218	7.350233
3274	5.126763	230.603758	11983.869376	6.303357
3275	7.874671	195.102299	17404.177061	7.509306

[3276 rows x 4 columns]

HISTOGRAMS

```
data.select_dtypes(include=['float64', 'int64']).hist(bins=20, figsize=(12, 10))

plt.suptitle("Histograms of Numerical Features", y=1.02)

plt.show()
```



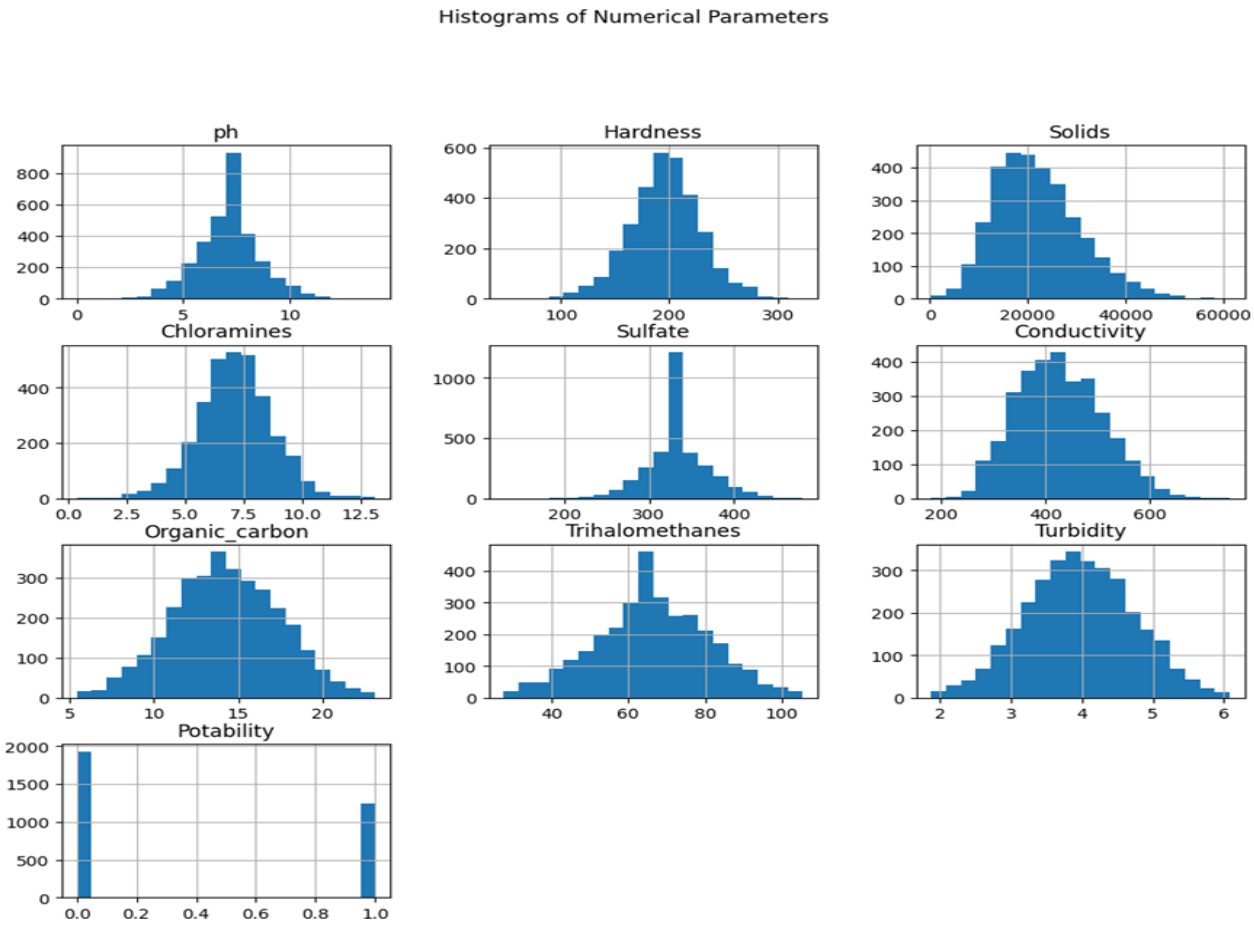
for column in numeric_columns:

data = handle_outliers_iqr(data, column)

Visualize Parameter Distributions

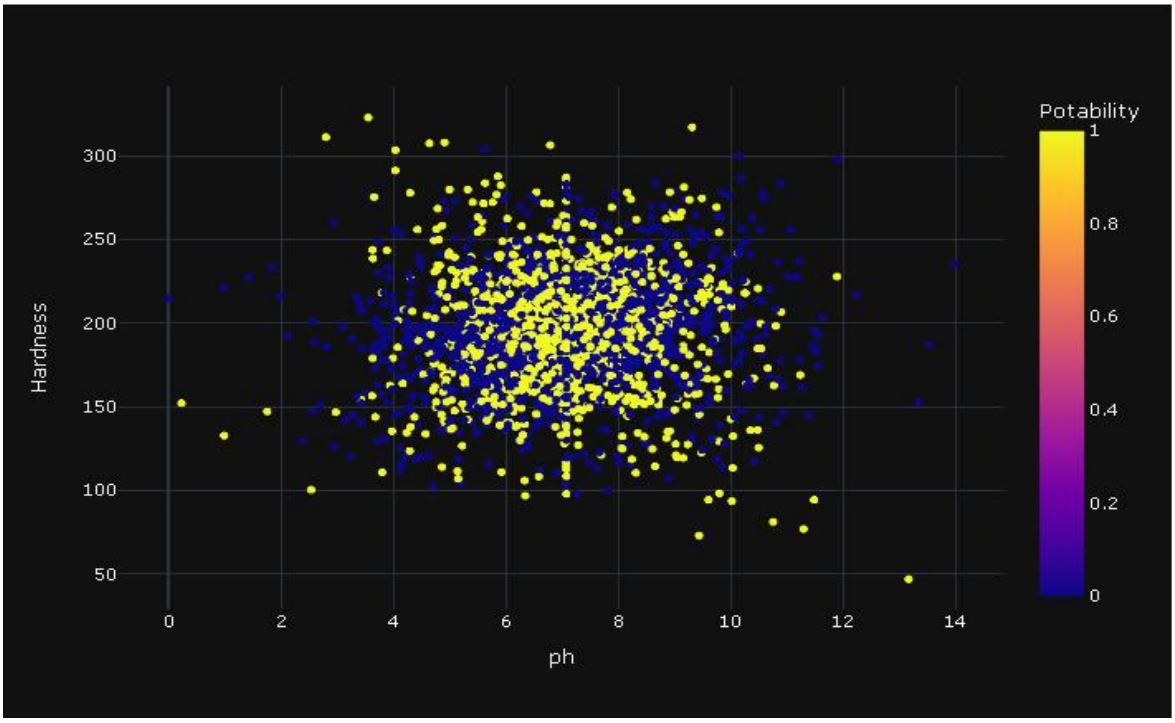
data.select_dtypes(include=['float64', 'int64']).hist(bins=20, figsize=(12, 10)) plt.suptitle("Histograms of Numerical Parameters", y=1.02)

plt.show()

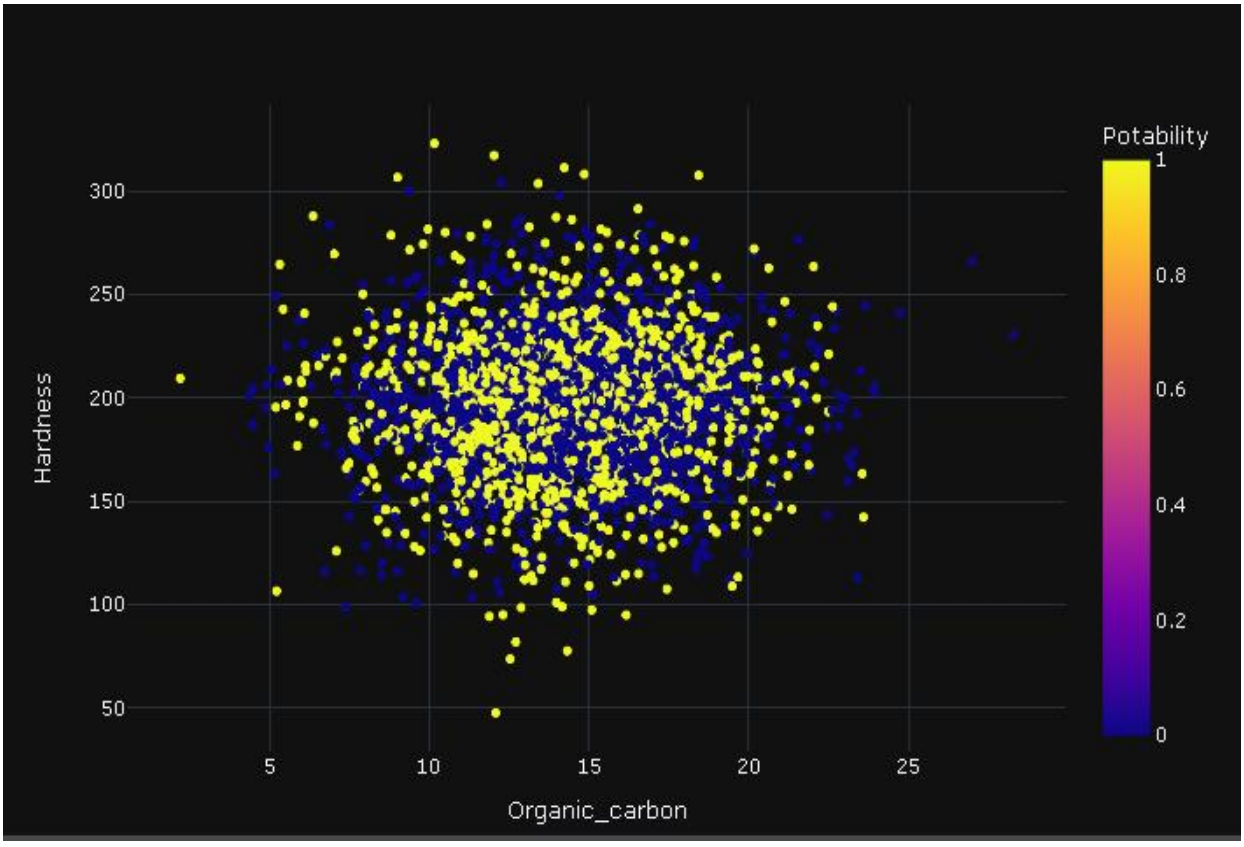


SCATTER PLOTS

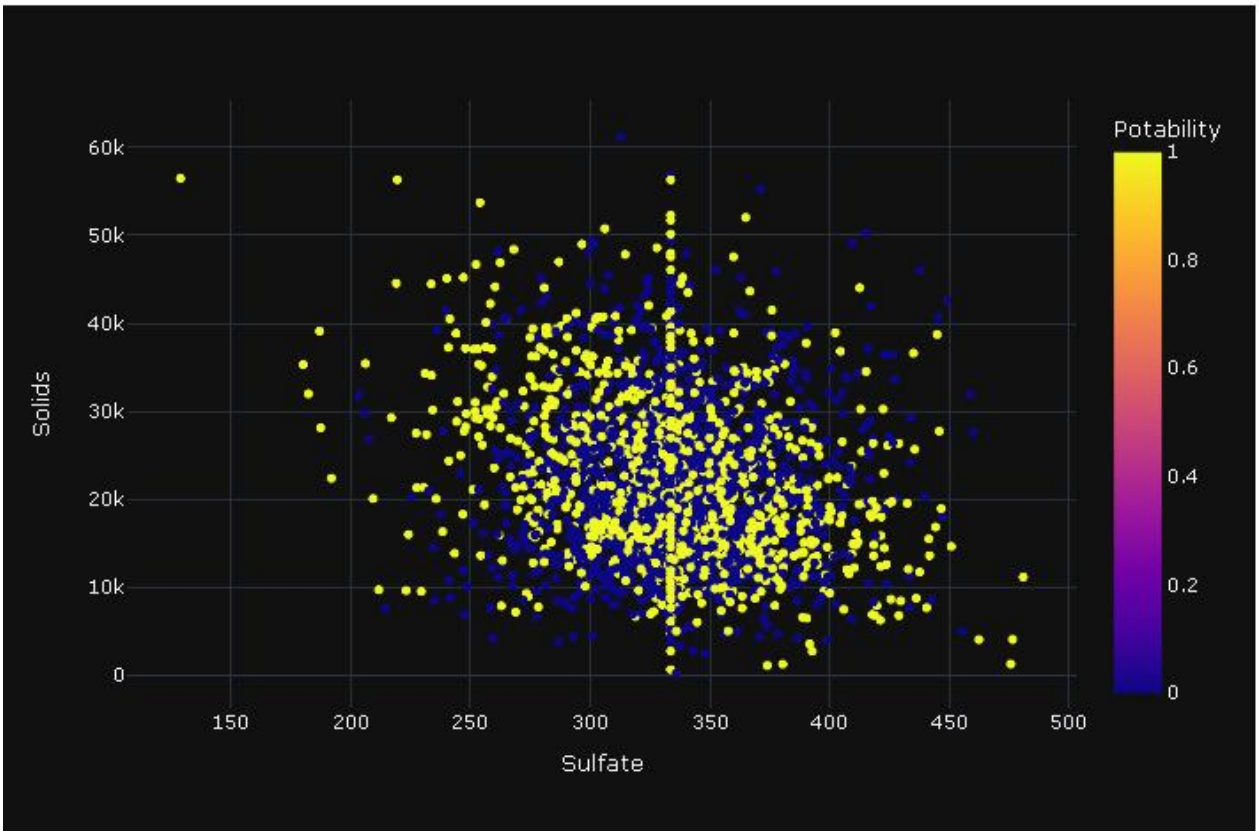
```
fig=px.scatter(data,x="ph",y="Hardness",color="Potability",template="plotl
y_dark")

fig.show()
```




```
fig = px.scatter(data,x ="Organic_carbon",y="Hardness",color=

"Potability",template="plotly_dark")

fig.show()
```



```
fig = px.scatter(data,x ="Sulfate",y="Solids",color=
"Potability",template="plotly_dark")
fig.show()
```



CORRELATION MATRIX

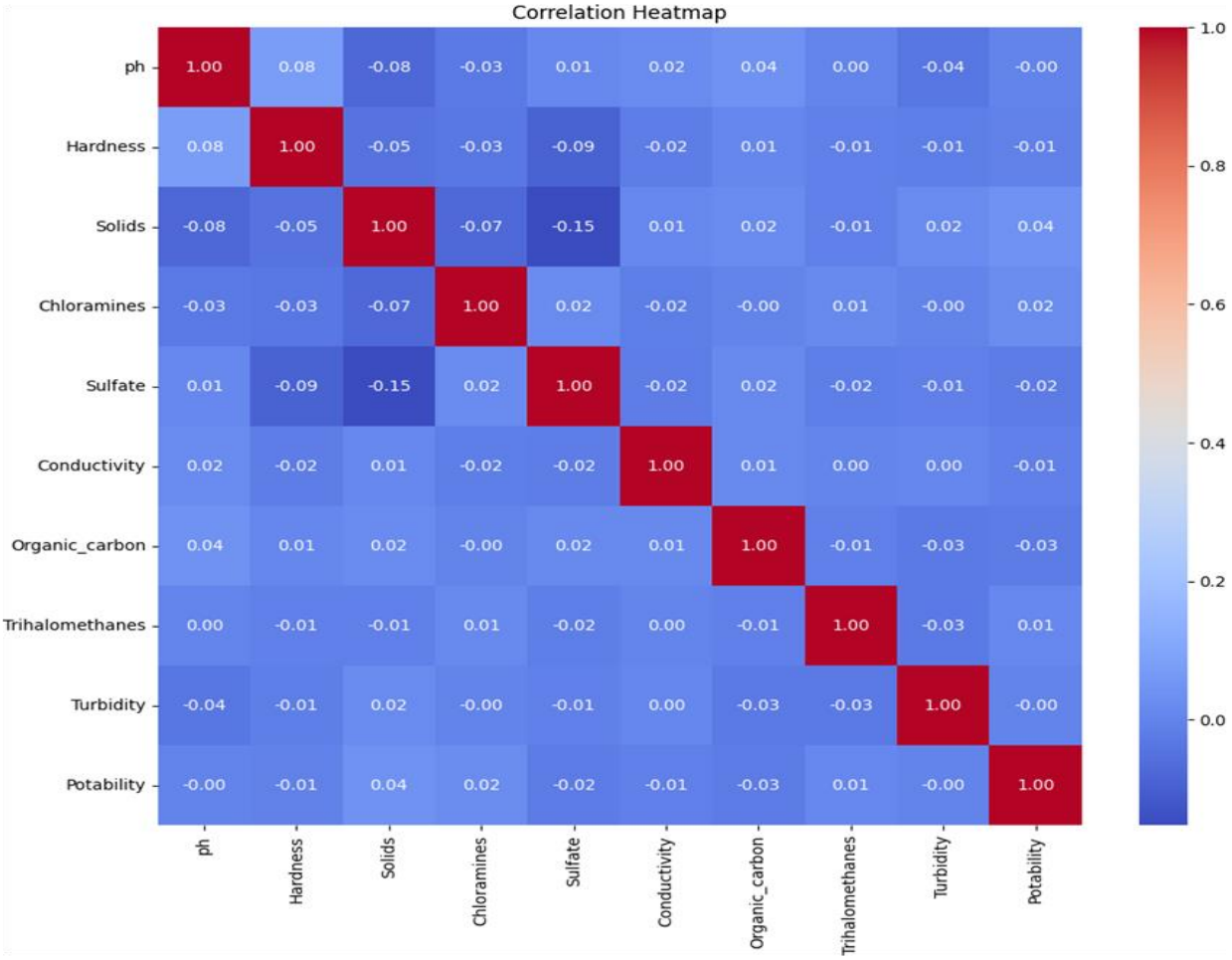
Visualize Correlations

correlation_matrix = data.corr() # Plot correlation heatmap

plt.figure(figsize=(12, 10))

sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f") plt.title('Correlation Heatmap')

plt.show()



Model Training

Decision Tree:

```
from sklearn.tree import DecisionTreeClassifier

dt = DecisionTreeClassifier(criterion= 'entropy', min_samples_split= 3,)

dt.fit(X_train,Y_train)
```

OUT:

DecisionTreeClassifier

```
DecisionTreeClassifier(criterion='entropy', min_samples_split=3)
```

```
Y_test
```

```
2541    0
2605    0
330     1
515     0
400     1
       ..
482     0
2970    0
50      0
839     0
374     1
Name: Potability, Length: 656, dtype: int64
```

```
Y_prediction=dt.predict(X_test)
```

```
from sklearn.metrics import accuracy_score, confusion_matrix
```

```
accuracy_score(Y_prediction,Y_test)
```

PREDICTION VALUE

```
59.756097560975604 %
```

Model Optimization /Hyper Parameter Tuning

```
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import RepeatedStratifiedKFold
```

```
dt= DecisionTreeClassifier()
criterion = ["gini","entropy"]
```

```
splitter = ['best','random ']

min_samples_split=range (1,10)

parameters = dict(criterion=criterion,splitter= splitter,

min_samples_split= min_samples_split)

cv= RepeatedStratifiedKFold(n_splits = 5,random_state=101)

grid_search_cv_dt= GridSearchCV(estimator=dt,

param_grid=parameters,scoring='accuracy',cv=cv)

print(grid_search_cv_dt.best_params_)

{'criterion': 'entropy', 'min_samples_split': 9, 'splitter': 'best'}

prediction_grid=grid_search_cv_dt.predict(X_test)

accuracy_score(Y_test,prediction_grid)*100
```

OUPUT

```
59.756097560975604 %
```