DEVELOPMENT PHASE PART 2

WATER QUALITY ANALYSIS

Date	29-10-2023
Team ID	714
Project Name	Water Quality Analysis

Table Of Content:

1. Introduction:

In the development part 2 the project is Continued building the analysis by creating visualizations and building a predictive model. Using visualization libraries like Matplotlib, Seaborn the histograms, scatter plots, and correlation matrices were created. A predictive model is built to determine water potability based on water quality parameters.

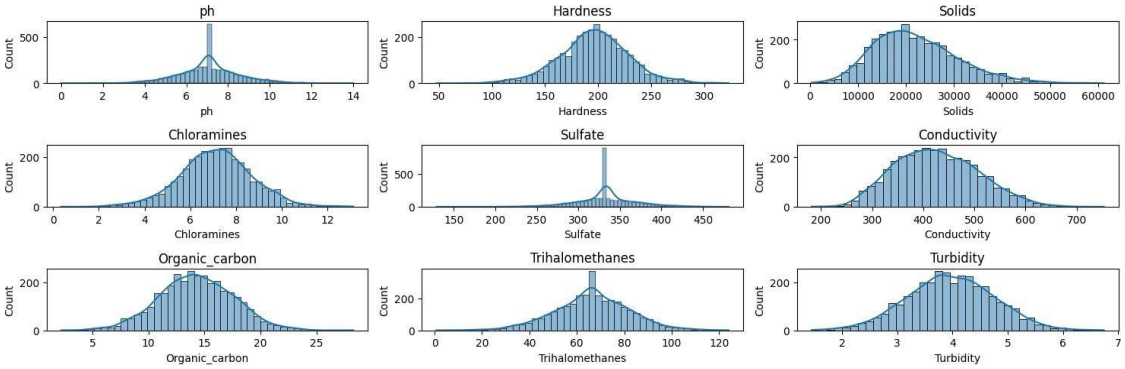
2. Data Preprocessing:

In the previous phase the Data processing which is essential in data analysis to increase data quality. Data processing is described as “the collection and manipulation of data components to produce meaningful information.” Through meticulous handling of missing values, dynamic feature scaling, and real-time outlier detection, the dataset attained a level of precision essential for accurate predictions. The data preprossing is done by using Jupyter notebook .

3. Data visualization:

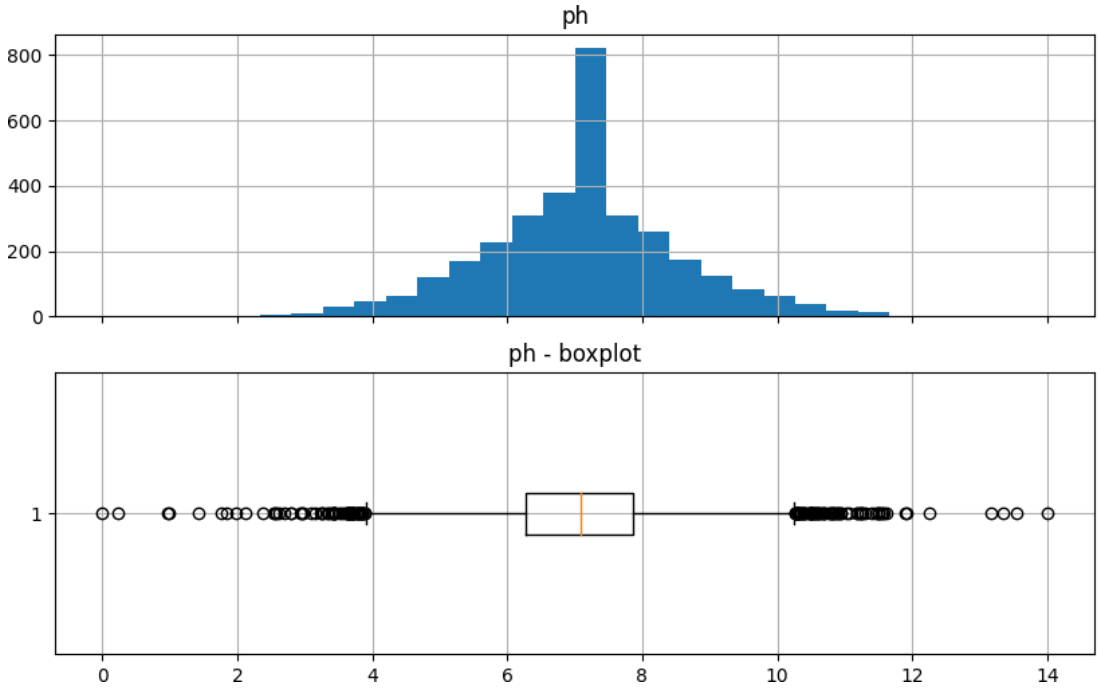
3.1. Histogram & Distribution.

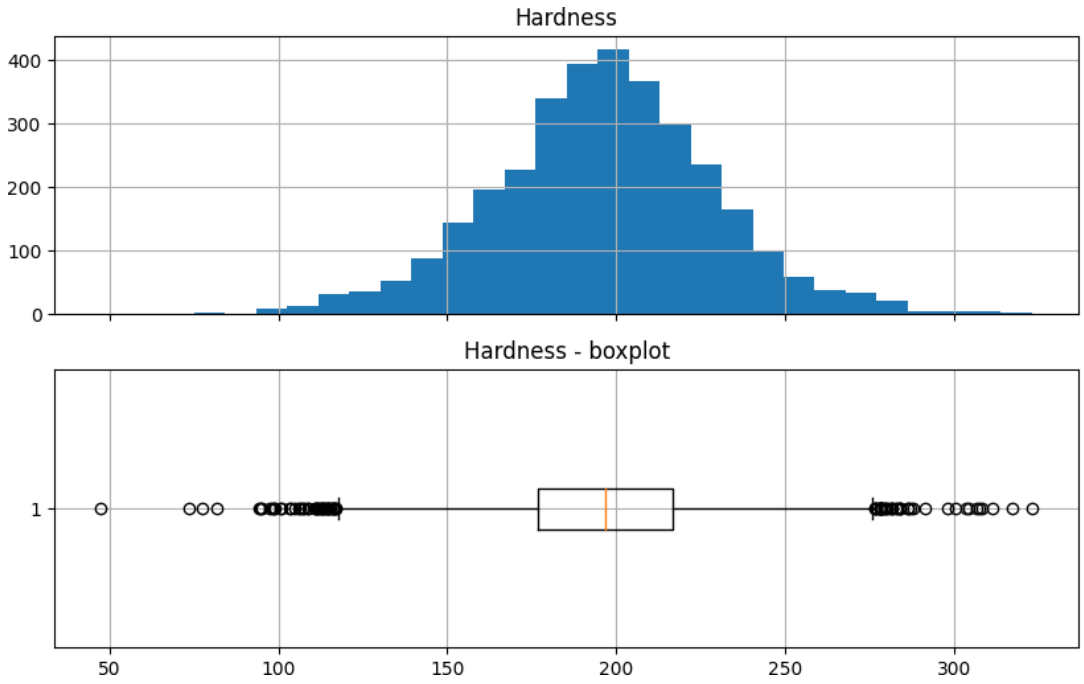
```
def show_distributions(columns: list, data: pd.DataFrame, nrows: int = 1,
ncols: int = 3):
    # This function creates distribution subplots.
    fig, axes = plt.subplots(nrows=nrows, ncols=ncols, figsize=(15, 5))
    axes = axes.ravel()
    for index, column in enumerate(columns):
        sns.histplot(data[column], kde=True, ax=axes[index])
        axes[index].set_title(column)

        # Adjust layout
    plt.tight_layout()
    plt.show()
show_distributions(data.columns[:-1], data,3,3)
```

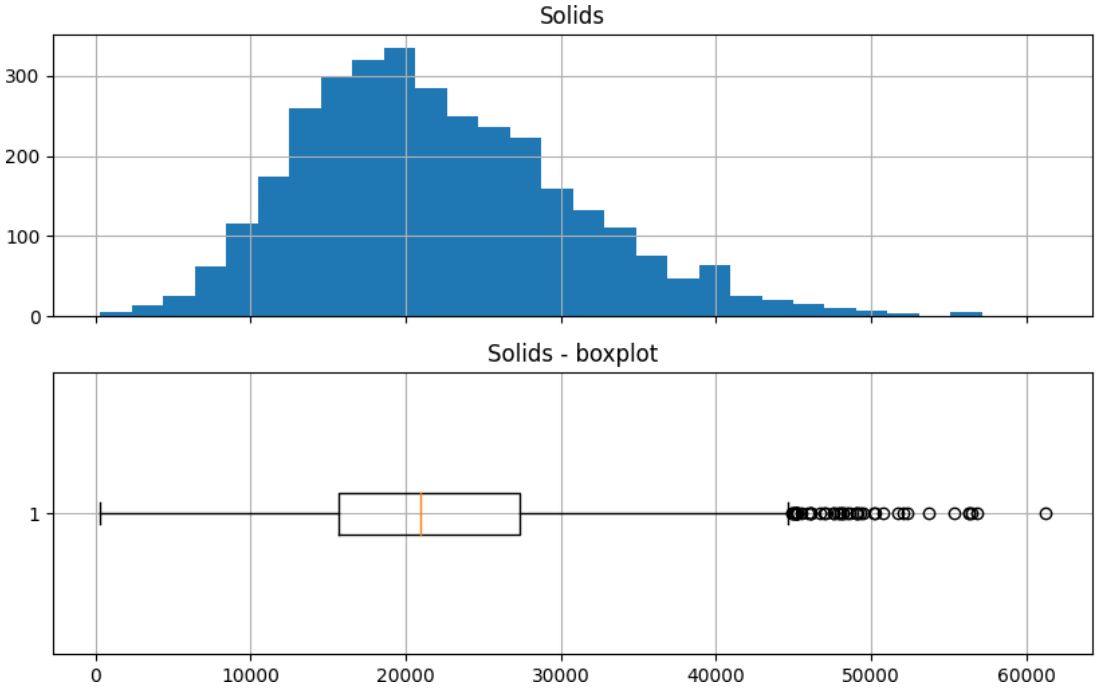


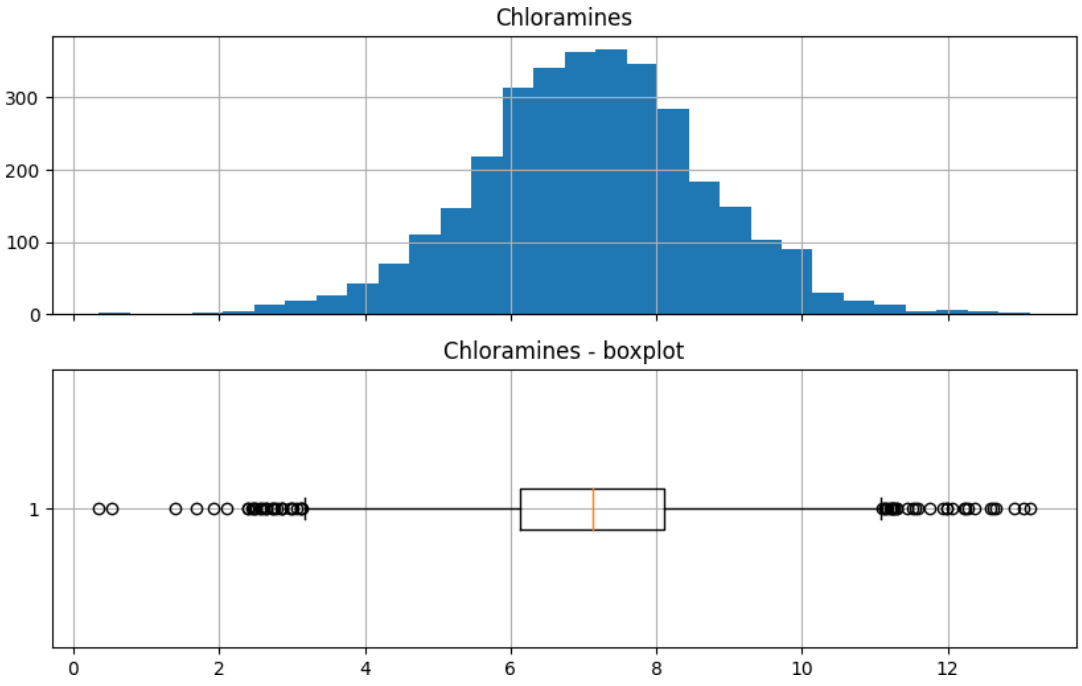
3.2. Boxplot

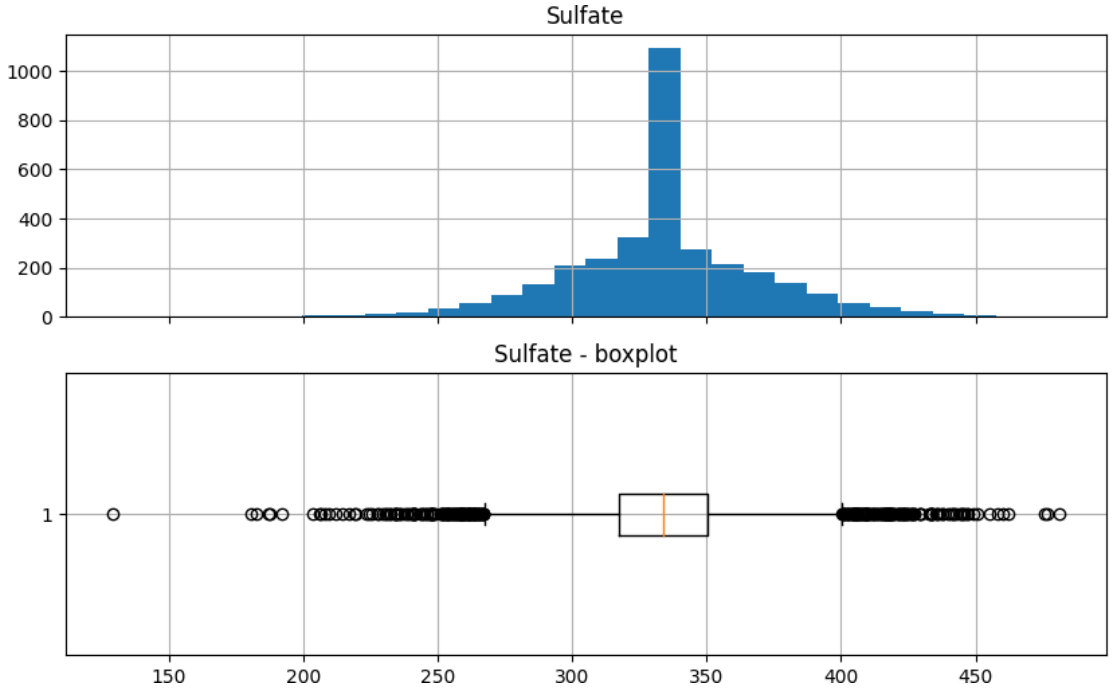
```
features_num = ['ph', 'Hardness', 'Solids', 'Chloramines', 'Sulfate',
                'Conductivity', 'Organic_carbon', 'Trihalomethanes',
                'Turbidity']
for f in features_num:
    fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(10,6), sharex=True)
    ax1.hist(data[f], bins=30)
    ax1.grid()
    ax1.set_title(f)
    # for boxplot we need to remove the NaNs first
    feature_wo_nan = data[~np.isnan(data[f])][f]
    ax2.boxplot(feature_wo_nan, vert=False)
    ax2.grid()
    ax2.set_title(f + ' - boxplot')
    plt.show()
```

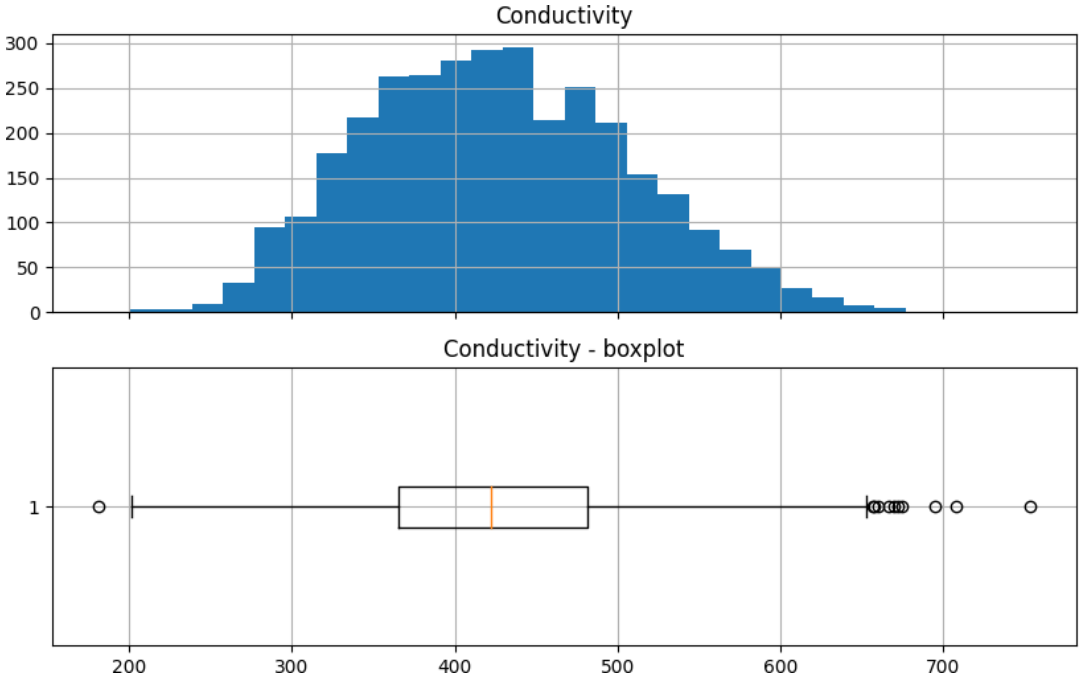

ph
800
600
400
200
0
ph - boxplot
1
0
2
4
6
8
10
12
14

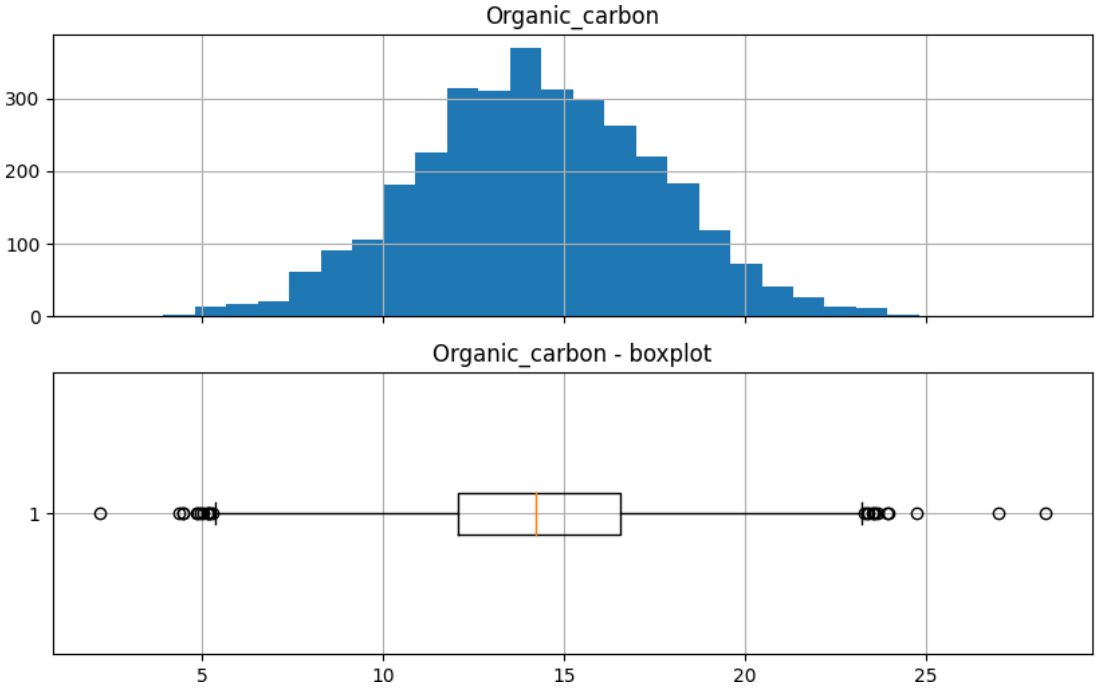

Hardness
400
300
200
100
0
Hardness - boxplot
1
50
100
150
200
250
300

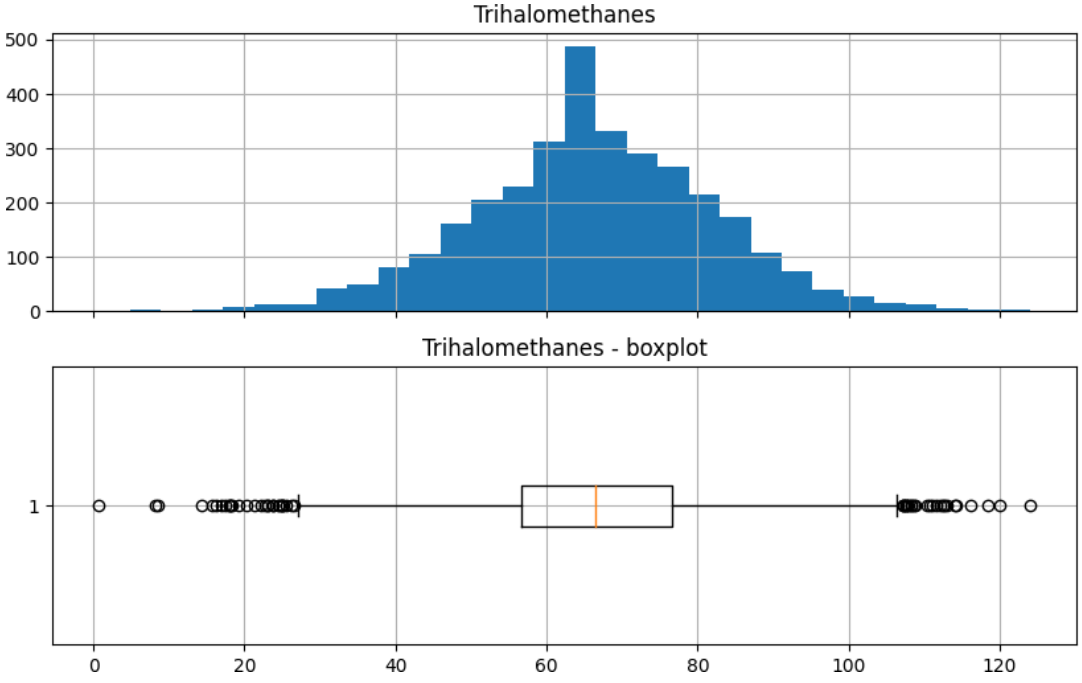

Solids
300
200
100
0
Solids - boxplot
1
0
10000
20000
30000
40000
50000
60000

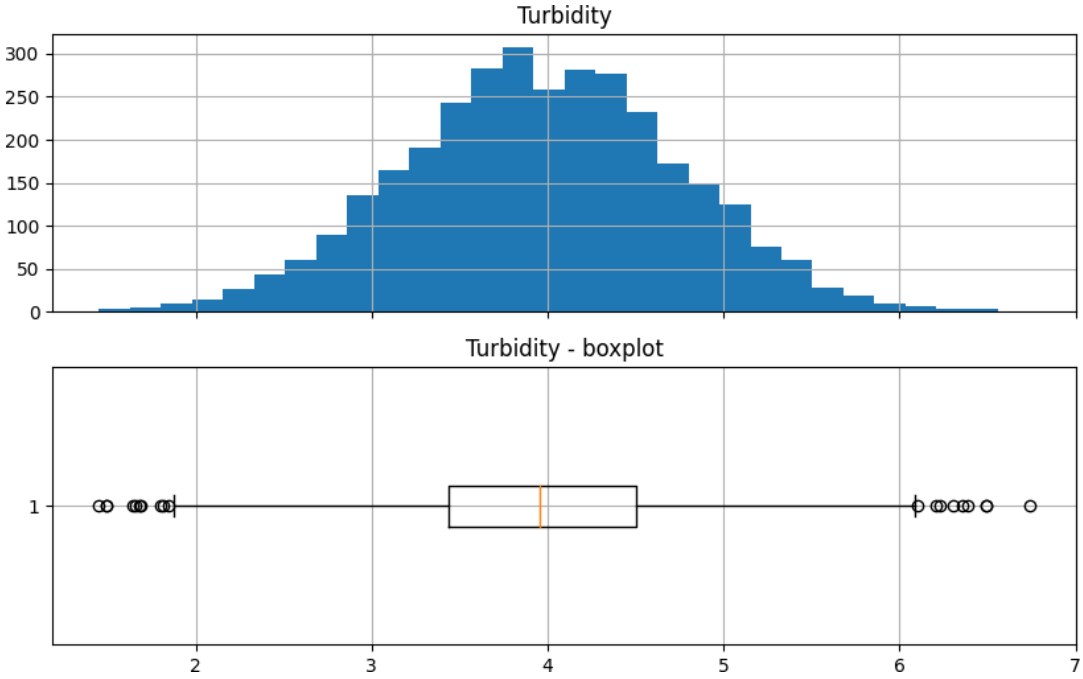

Chloramines
300
200
100
0
Chloramines - boxplot
1
0
2
4
6
8
10
12


Sulfate
1000
800
600
400
200
0
Sulfate - boxplot
1
150
200
250
300
350
400
450


Conductivity
300
250
200
150
100
50
0
Conductivity - boxplot
1
200
300
400
500
600
700


Organic_carbon
300
200
100
0
Organic_carbon - boxplot
1
5
10
15
20
25


Trihalomethanes
500
400
300
200
100
0
Trihalomethanes - boxplot
1
0
20
40
60
80
100
120

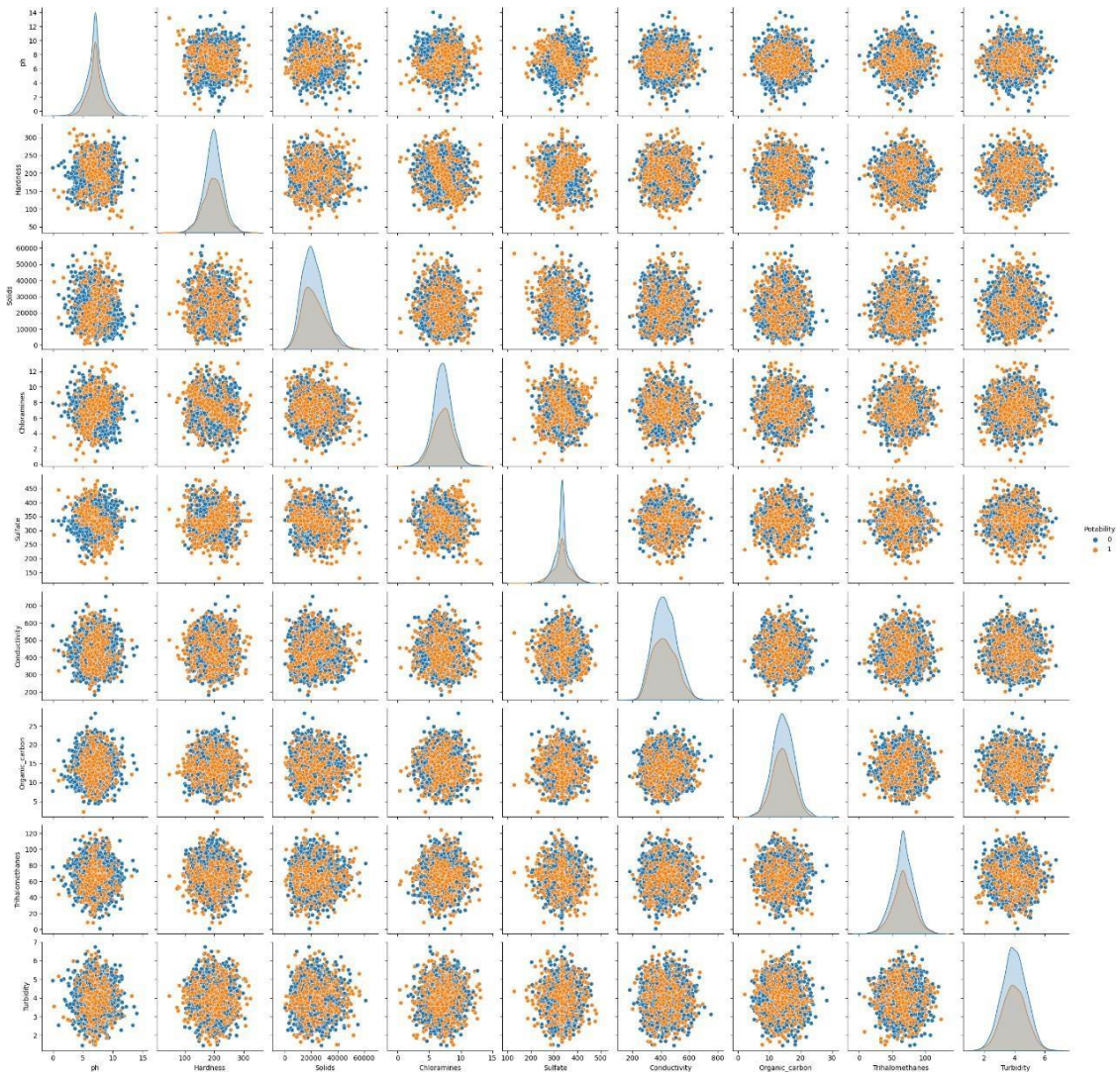

Turbidity
300
250
200
150
100
50
0

Turbidity - boxplot
1
2
3
4
5
6
7

3.3. Scatter Plots.

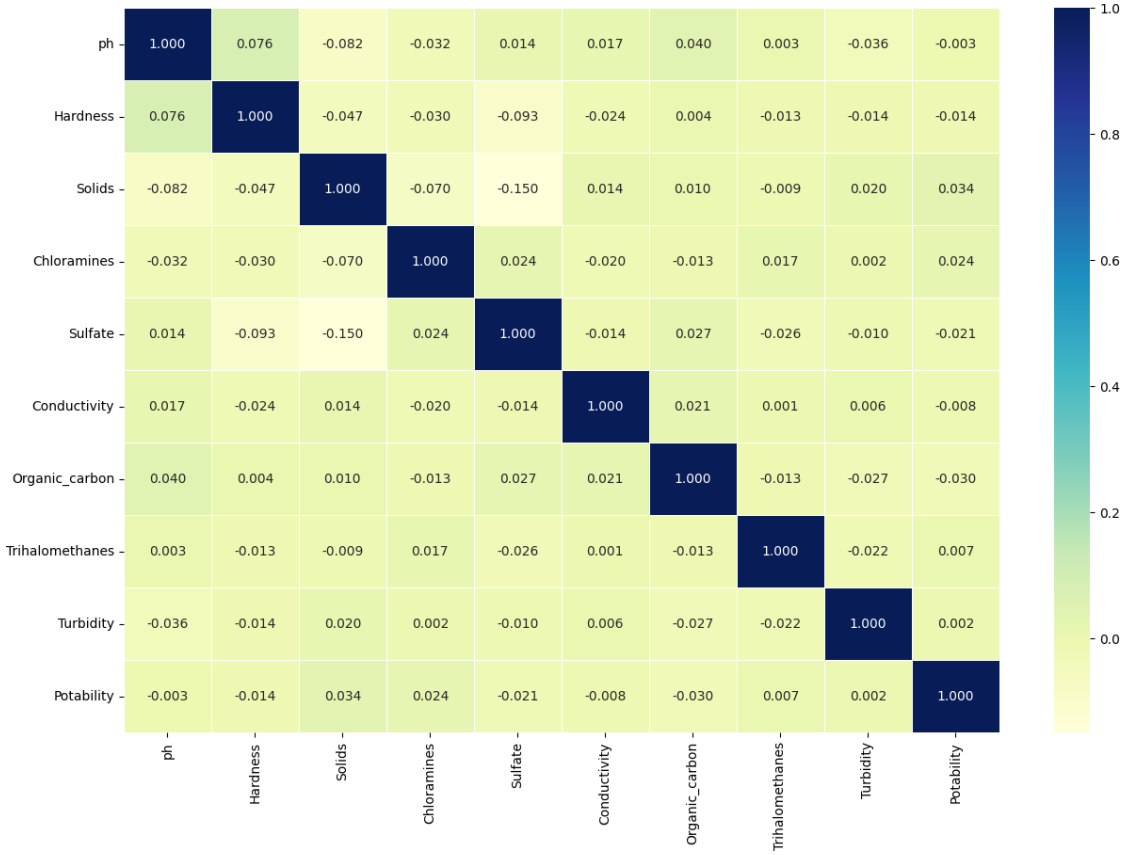
```
sns.pairplot(data=data,hue="Potability")
```

```
<seaborn.axisgrid.PairGrid at 0x7e0550701060>
```



3.4. Correlation Heatmap.

```
corr_mat = data.corr()
fig, ax = plt.subplots(figsize=(15,10))
ax =
sns.heatmap(corr_mat,annot=True,linewidths=0.5,fmt='.3f',cmap='YlGnBu')
```



4. Data Splitting.

```
sm = SMOTE(random_state=42)

X, y = data[data.columns[:-1]], data["Potability"]
X, y = sm.fit_resample(X, y)
from sklearn.preprocessing import Normalizer, StandardScaler
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size =
0.25)
```

#5. Predictive Model.

```
models = [RandomForestClassifier()]
pipelines = {}
for model in models:
    model_name = str(model._class_).split(".")[-1].split("'")[0]
    pipe = Pipeline([
        ("scaler", StandardScaler()),  # Preprocessing step
        ("classifier", model)  # Classifier step
    ])
    pipelines[model_name] = pipe


for name,pipe in pipelines.items():
    print(f"Training {name}")
    scores = cross_val_score(pipe, X_train, y_train, cv = 5, scoring =
"accuracy")
    print(f"Mean Score {scores.mean()} -- Std {scores.std()} -- Min
```

```
{scores.min()} -- Max {scores.max()}")
    pipe.fit(X_train, y_train)
```

```
Training RandomForestClassifier
Mean Score 0.6740100166944908 -- Std 0.01986394725623316 -- Min
0.6427378964941569 -- Max 0.7028380634390651
```

5.1. Hyperparameter tuning the RandomForest model.

```
param_grid = {
    "criterion": ["gini", "entropy", "log_loss"],
    'n_estimators': [10, 20, 30, 40,50],
    'max_depth': [5, 10, 20, 30, 50],
}

rf_classifier = RandomForestClassifier(random_state = 42)
scorer = make_scorer(accuracy_score)
grid_search = GridSearchCV(
    rf_classifier, param_grid, scoring=scorer, cv=5, verbose = 1
)
grid_search.fit(X_train, y_train)
best_rf = grid_search.best_estimator_

best_predictions =  best_rf.predict(X_test)
best_accuracy = accuracy_score(best_predictions, y_test)

print("Best Accuracy Score:", best_accuracy)
```

```
Fitting 5 folds for each of 75 candidates, totalling 375 fits
Best Accuracy Score: 0.6866866866866866
```

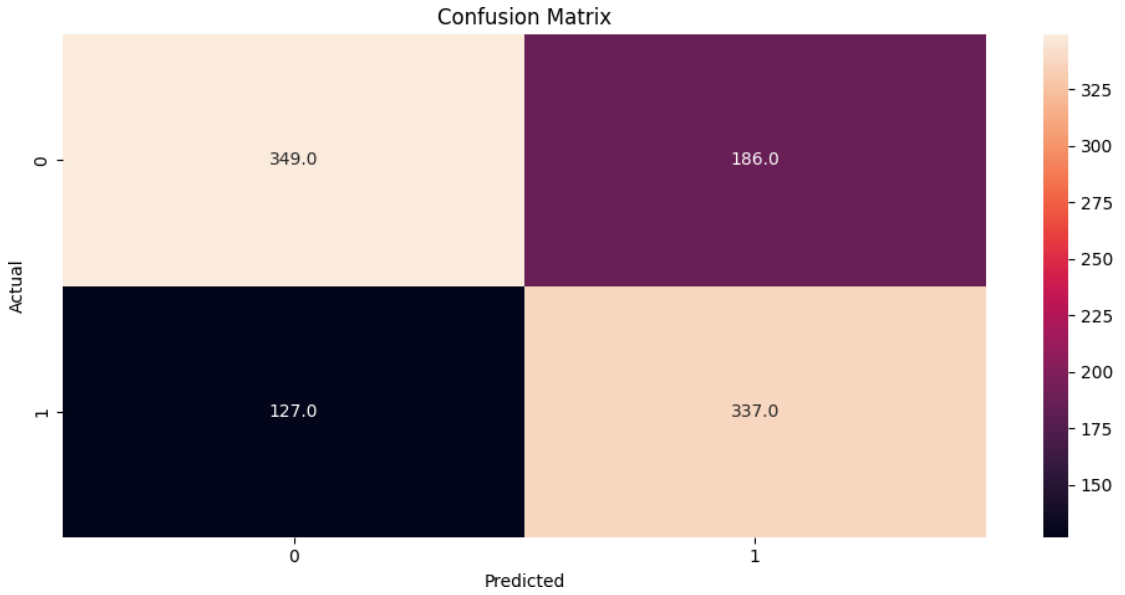
5.2. RandomForest model Accuracy Score.

```
accuracy_score(best_rf.predict(X_test),y_test)
```

```
0.6866866866866866
```

5.3. Confusion Matrix

```
plt.figure(figsize=(10,5))
sns.heatmap(confusion_matrix(best_rf.predict(X_test),y_test), annot =
True,fmt='.1f')
plt.ylabel("Actual")
plt.xlabel("Predicted")
plt.title("Confusion Matrix")
plt.tight_layout()
```



6.Conclusion:

Thus, in this document the project is Continue building the analysis by creating visualizations and building a predictive model. Using visualization libraries like Matplotlib, Seaborn the histograms, scatter plots, and correlation matrices were created. The Random Forest Classifier, trained on pre-processed water potability data, demonstrates robust performance with an accuracy of approximately 74%. Despite complexities in the dataset, the model showcases reliable predictive power, making it a suitable choice for potability prediction.