**Project Development Phase**
**Model Performance Test**

| | |
|---|---|
| Date | 18 November 2022 |
| Team ID | PNT2022TMID28091 |
| Project Name | Efficient Water Quality Analysis & Prediction using Machine Learning |
| Maximum Marks | 10 Marks |

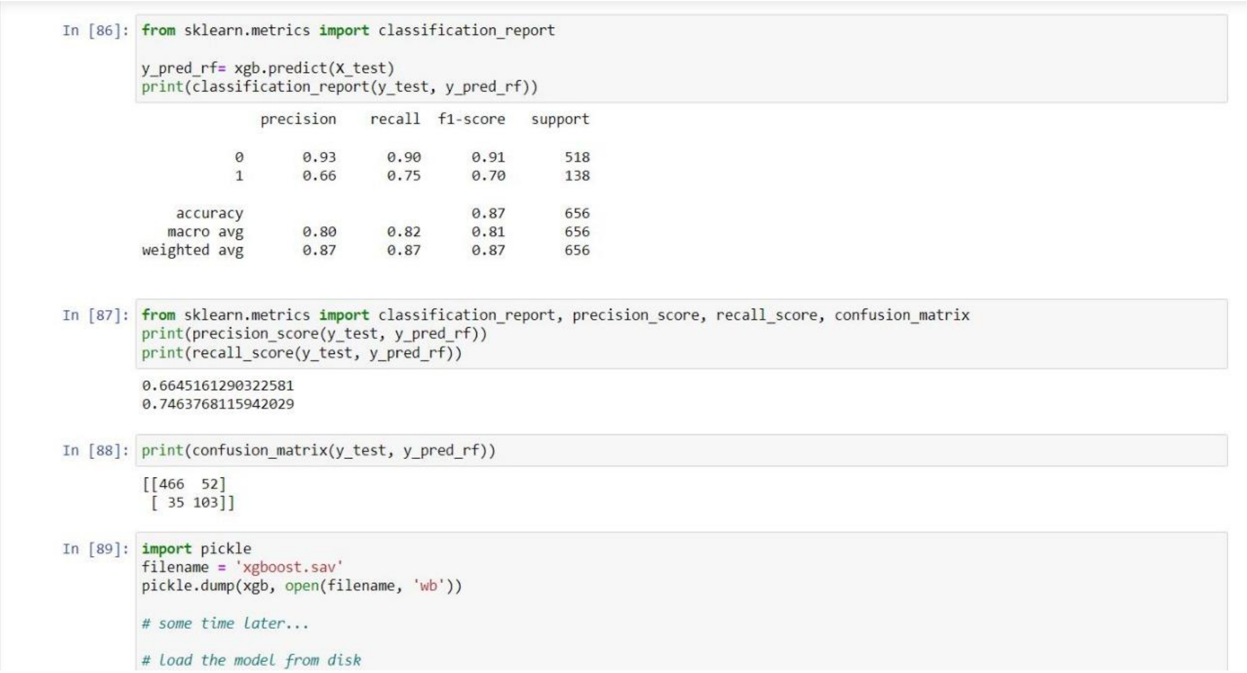
**Model Performance Testing:**

Project team shall fill the following information in model performance testing template

# ***Metrics***

## Classification Model and Regression Model:

| | |
|---|---|
| Logistic Regression | 0.79 |
| K Nearest Neighbors | 0.78 |
| Decision Tree | 0.86 |
| Random Forest | 0.89 |
| AdaBoost | 0.88 |
| Bagging Classifier | 0.89 |
| XGBoost | 0.87 |

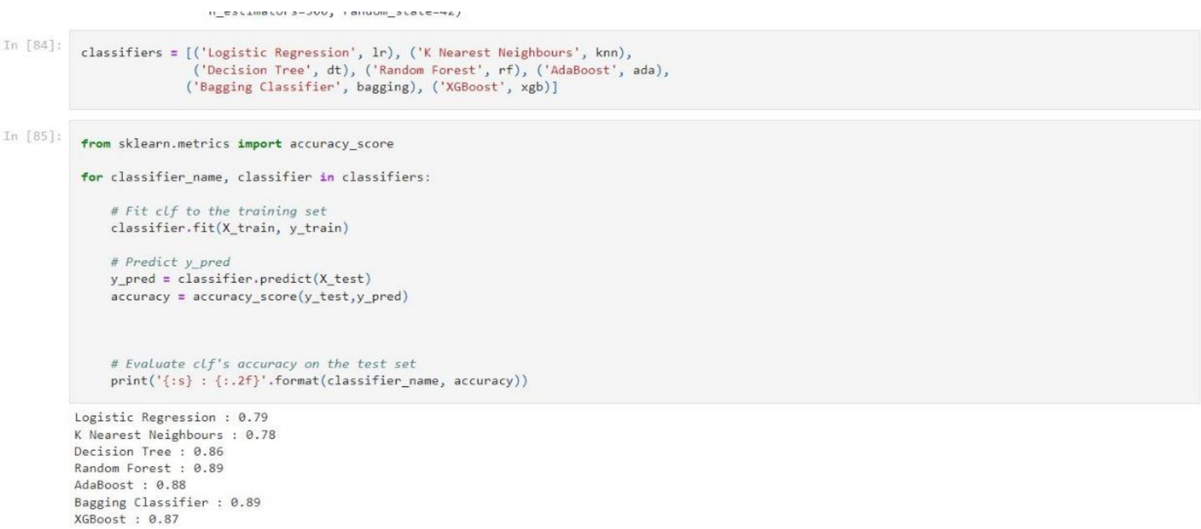


```
In [86]: from sklearn.metrics import classification_report

         y_pred_rf= xgb.predict(X_test)
         print(classification_report(y_test, y_pred_rf))

                       precision    recall  f1-score   support

                    0       0.93      0.90      0.91       518
                    1       0.66      0.75      0.70       138

             accuracy                           0.87       656
            macro avg       0.80      0.82      0.81       656
         weighted avg       0.87      0.87      0.87       656
```

```
In [87]: from sklearn.metrics import classification_report, precision_score, recall_score, confusion_matrix
         print(precision_score(y_test, y_pred_rf))
         print(recall_score(y_test, y_pred_rf))

         0.6645161290322581
         0.7463768115942029
```

```
In [88]: print(confusion_matrix(y_test, y_pred_rf))

         [[466  52]
          [ 35 103]]
```

```
In [89]: import pickle
         filename = 'xgboost.sav'
         pickle.dump(xgb, open(filename, 'wb'))

         # some time later...

         # load the model from disk
```



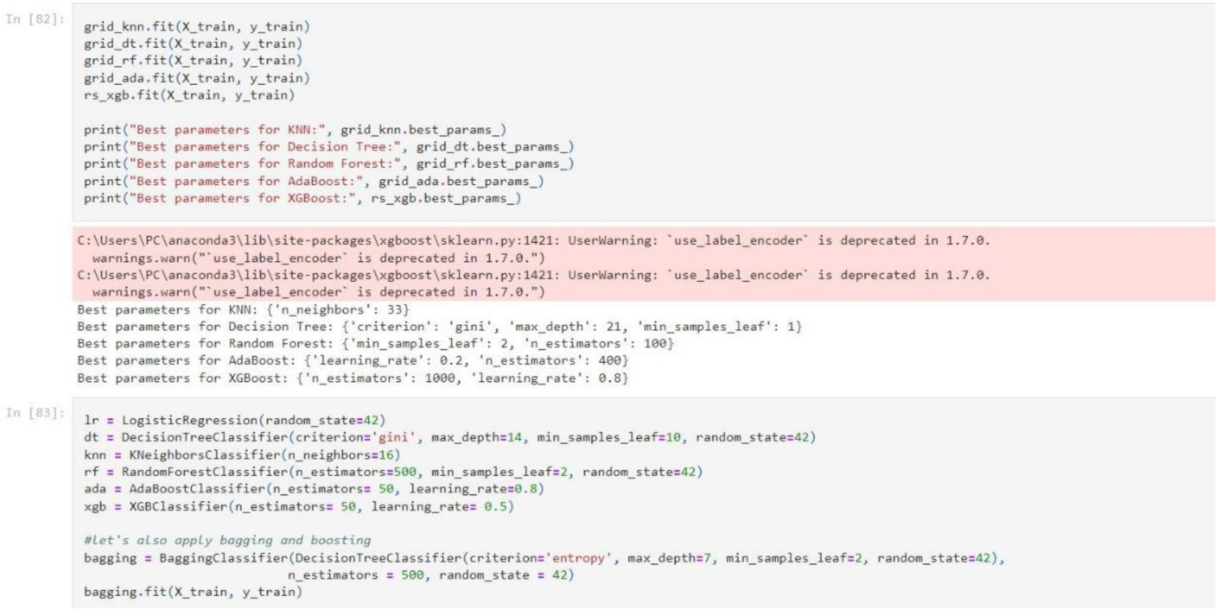
```
                      n_estimators=500, random_state=42)
```

```
In [84]: classifiers = [('Logistic Regression', lr), ('K Nearest Neighbours', knn),
                        ('Decision Tree', dt), ('Random Forest', rf), ('AdaBoost', ada),
                       ('Bagging Classifier', bagging), ('XGBoost', xgb)]
```

```
In [85]: from sklearn.metrics import accuracy_score

         for classifier_name, classifier in classifiers:

             # Fit clf to the training set
             classifier.fit(X_train, y_train)

             # Predict y_pred
             y_pred = classifier.predict(X_test)
             accuracy = accuracy_score(y_test,y_pred)



             # Evaluate clf's accuracy on the test set
             print('{:s} : {:.2f}'.format(classifier_name, accuracy))

         Logistic Regression : 0.79
         K Nearest Neighbours : 0.78
         Decision Tree : 0.86
         Random Forest : 0.89
         AdaBoost : 0.88
         Bagging Classifier : 0.89
         XGBoost : 0.87
```

# ***Tune the Model***

## Validation Method –



```
In [82]:
grid_knn.fit(X_train, y_train)
grid_dt.fit(X_train, y_train)
grid_rf.fit(X_train, y_train)
grid_ada.fit(X_train, y_train)
rs_xgb.fit(X_train, y_train)

print("Best parameters for KNN:", grid_knn.best_params_)
print("Best parameters for Decision Tree:", grid_dt.best_params_)
print("Best parameters for Random Forest:", grid_rf.best_params_)
print("Best parameters for AdaBoost:", grid_ada.best_params_)
print("Best parameters for XGBoost:", rs_xgb.best_params_)
```

```
C:\Users\PC\anaconda3\lib\site-packages\xgboost\sklearn.py:1421: UserWarning: `use_label_encoder` is deprecated in 1.7.0.
  warnings.warn("`use_label_encoder` is deprecated in 1.7.0.")
C:\Users\PC\anaconda3\lib\site-packages\xgboost\sklearn.py:1421: UserWarning: `use_label_encoder` is deprecated in 1.7.0.
  warnings.warn("`use_label_encoder` is deprecated in 1.7.0.")
Best parameters for KNN: {'n_neighbors': 33}
Best parameters for Decision Tree: {'criterion': 'gini', 'max_depth': 21, 'min_samples_leaf': 1}
Best parameters for Random Forest: {'min_samples_leaf': 2, 'n_estimators': 100}
Best parameters for AdaBoost: {'learning_rate': 0.2, 'n_estimators': 400}
Best parameters for XGBoost: {'n_estimators': 1000, 'learning_rate': 0.8}
```

```
In [83]:
lr = LogisticRegression(random_state=42)
dt = DecisionTreeClassifier(criterion='gini', max_depth=14, min_samples_leaf=10, random_state=42)
knn = KNeighborsClassifier(n_neighbors=16)
rf = RandomForestClassifier(n_estimators=500, min_samples_leaf=2, random_state=42)
ada = AdaBoostClassifier(n_estimators= 50, learning_rate=0.8)
xgb = XGBClassifier(n_estimators= 50, learning_rate= 0.5)

#let's also apply bagging and boosting
bagging = BaggingClassifier(DecisionTreeClassifier(criterion='entropy', max_depth=7, min_samples_leaf=2, random_state=42),
                            n_estimators = 500, random_state = 42)
bagging.fit(X_train, y_train)
```