SPRINT 1 –DATA COLLECTION AND DATA PREPROCESSING

1.DATA COLLECTION :

We can collect the dataset from, https://www.kaggle.com/datasets/anbarivan/indian-water-quality-data?resource=download.

2.DATA PREPROCESSING:

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

- Getting the dataset
- Importing libraries
- Importing datasets
- Analyzing the data
- Finding Missing Data
- Encoding Categorical Data
- Splitting dataset into training and test set
- Feature scaling

GETTING THE DATASET:

We can collect the dataset from, https://www.kaggle.com/datasets/anbarivan/indian-water-quality-data?resource=download

IMPORTING LIBRARIES:

import pandas as pd

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

IMPORTING DATASETS:

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

ANALYSING THE DATA:

```
df.head();
df.describe();
df.shape
df.info();
```

FINDING MISSING DATA:

```
df.isnull().any();
df.isnull().sum();
for feature in df.columns:
   if df[feature].isnull().sum()>0:
      print(f"{feature} : {round(df[feature].isnull().mean(),4)*100}%")
-------Fill missing values with median
for feature in df.columns:
   df[feature].fillna(df[feature].median() , inplace = True)
------- find dublicate rows in dataset
duplicate = df[df.duplicated()]
duplicate
----------removing outliers
```

```
Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3 - Q1
print(IQR)
```

ENCODING CATEGORICAL DATA:

```
sns.countplot(x='Potability',data=df )
df["Potability"].value_counts()
print(f"0 : {round(1998 /3276 * 100 , 2)}")
print(f"1 : {round(1278 /3276 * 100 , 2)}")


---------- Correlation
plt.figure(figsize=(25,25))
ax = sns.heatmap(df.corr(), cmap = "coolwarm", annot=True, linewidth=2)
```

SPLITTING DATASET INTO TESTING AND TRAINING:

```
---------splitting data into x and y
X = df.iloc[: , : -1]
y = df.iloc[ : , -1]


----------split dataset into train and test
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3,
random_state= 5)
```

FEATURE SCALING:

```
from sklearn.preprocessing import StandardScaler
```

```
sc = StandardScaler()

X_train_final = sc.fit_transform(X_train)

X_test_final = sc.transform(X_test)

from sklearn.metrics import confusion_matrix, classification_report,
accuracy_score

----------Random Forest Classifier

from sklearn.ensemble import RandomForestClassifier

rf_classifier = RandomForestClassifier(n_estimators = 20, criterion = 'entropy',
class_weight = "balanced_subsample",random_state = 51)

rf_classifier.fit(X_train_final, y_train)

y_pred = rf_classifier.predict(X_test_final)

accuracy_score(y_test, y_pred)

print(classification_report(y_test, y_pred))


----------XGBoost Classifier

from xgboost import XGBClassifier

xgb_classifier = XGBClassifier(random_state=0)

xgb_classifier.fit(X_train_final, y_train)

y_pred_xgb = xgb_classifier.predict(X_test_final)

accuracy_score(y_test, y_pred_xgb)


print(classification_report(y_test, y_pred_xgb))
```