Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.

water_quality_predict.ipynb

File Edit View Insert Runtime Tools Help

Share Gemini

Commands + Code + Text Run all Reconnect

```
[ ] Start coding or generate with AI.
```

Load Data

```
import pandas as pd

df = pd.read_csv('https://raw.githubusercontent.com/DinukaSanjana/Water-Quality-Monitoring/refs/heads/main/water_potability_csv.csv')
df
```

	ph	Conductivity	Turbidity	Potability
0	3.716080	592.885359	4.500656	0
1	8.099124	418.606213	3.055934	0
2	8.316766	363.266516	4.628771	0
3	9.092223	398.410813	4.075075	0
4	5.584087	280.467916	2.559708	0
...	...	...	...	...
2780	4.668102	526.424171	4.435821	1
2781	7.808856	392.449580	2.798243	1
2782	9.419510	432.044783	3.298875	1
2783	5.126763	402.883113	4.708658	1
2784	7.874671	327.459761	2.309149	1

2785 rows × 4 columns

Next steps: Generate code with df View recommended plots New interactive sheet

Data Separation

```
[ ] y = df['Potability']
    y
```

	Potability
0	0
1	0
2	0
3	0
4	0
...	...
2780	1
2781	1
2782	1
2783	1
2784	1

2785 rows × 1 columns

dtype: int64

```
[ ] x = df.drop('Potability', axis=1)
    x
```

	ph	Conductivity	Turbidity
0	3.716080	592.885359	4.500656
1	8.099124	418.606213	3.055934
2	8.316766	363.266516	4.628771
3	9.092223	398.410813	4.075075
4	5.584087	280.467916	2.559708
...	...	...	...
2780	4.668102	526.424171	4.435821
2781	7.808856	392.449580	2.798243
2782	9.419510	432.044783	3.298875
2783	5.126763	402.883113	4.708658
2784	7.874671	327.459761	2.309149

2785 rows × 3 columns

Next steps: Generate code with x View recommended plots New interactive sheet

```
[ ] y = df['Potability']
    y
```

	Potability
0	0
1	0
2	0
3	0
4	0
...	...
2780	1
2781	1
2782	1
2783	1
2784	1

2785 rows × 1 columns

dtype: int64

```
[ ] from sklearn.model_selection import train_test_split

    x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=100)
```

```
[ ] x_test
```

	ph	Conductivity	Turbidity

2699	6.641174	421.417352	3.154701
2166	5.870616	427.182531	3.728785
2767	6.683368	307.725009	5.208061
442	5.608745	596.076368	4.644212
2185	5.946161	499.937502	4.903632
...	...	...	...
785	5.700785	359.506553	3.978141
2039	5.916930	428.832746	3.157560
2089	5.097786	445.562644	4.829323
709	7.440825	452.995293	2.496343
1532	6.530796	490.644138	3.181635

557 rows × 3 columns

Next steps: Generate code with x_test | View recommended plots | New interactive sheet

Model Building

Linear Regression

```
[ ] from sklearn.linear_model import LinearRegression

    lr = LinearRegression()
    lr.fit(x_train, y_train)
```

```
LinearRegression
```

Apply the model to make a prediction

```
[ ] y_lr_train_pred = lr.predict(x_train)
    y_lr_test_pred = lr.predict(x_test)
```

```
[ ] print(y_lr_train_pred, y_lr_test_pred)
```

Show hidden output

```
[ ] y_lr_test_pred
```

Show hidden output

Evaluate model performance

```
[ ] from sklearn.metrics import mean_squared_error, r2_score

    lr_train_mea = mean_squared_error(y_train, y_lr_train_pred)
    lr_train_r2 = r2_score(y_train, y_lr_train_pred)

    lr_test_mea = mean_squared_error(y_test, y_lr_test_pred)
    lr_test_r2 = r2_score(y_test, y_lr_test_pred)
```

```
[ ] print('LR MSE (Train) : ' ,lr_train_mea)
    print('LR R2 (Train) : ' ,lr_train_r2)
    print('LR MSE (Test) : ' ,lr_test_mea)
    print('LR R2 (Test) : ' ,lr_test_r2)
```

```
LR MSE (Train) :  0.23525205378758154
LR R2 (Train) :  0.00026416410235607923
LR MSE (Test) :  0.2557692254971047
LR R2 (Test) :  -0.0292107190823764 9
```

```
[ ] results = pd.DataFrame(['Linear Regression', lr_train_mea, lr_train_r2, lr_test_mea, lr_test_r2]).transpose()
```

```
[ ] results
```

	0	1	2	3	4
0	Linear Regression	0.235252	0.000264	0.255769	-0.029211

```
[ ] Start coding or generate with AI.
```