



การคาดการณ์ค่าความต้องการออกซิเจนในทางเคมี  
โดยข้อมูลคุณภาพน้ำของแม่น้ำในประเทศเกาหลีใต้

Chemical oxygen demand prediction  
based on the south Korean river water quantity.

เสนอ

ผศ.ดร. พาพิศ วงศ์ชัยสุวัฒน์

จัดทำโดย

นายคุณานนต์ สุรสร 6110502634 หมู่ 1

ภาควิชาวิศวกรรมอุตสาหการ

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์

ปีการศึกษา 2566

## คำนำ

รายงานเล่มนี้เป็นส่วนหนึ่งของโครงการในวิชาระบบสารสนเทศเพื่อการจัดการสำหรับวิศวกรเป็นรายงานเกี่ยวกับการหาโมเดลสำหรับคาดการณ์ค่าความต้องการออกซิเจนในทางเคมีที่เป็นค่าสำหรับการวัดคุณภาพน้ำและเป็นรายงานเพื่อการศึกษาเกี่ยวกับการเขียนโปรแกรม การเรียนรู้ของเครื่อง (Machine Learning) และ การสร้างโมเดลสำหรับการเรียนรู้ของเครื่อง

โครงการนี้นำเอาข้อมูลคุณภาพของแม่น้ำในประเทศเกาหลีใต้มาเป็นข้อมูลในการศึกษาเพื่อสร้างโมเดลและหาโมเดลที่สามารถคาดการณ์ค่าได้โดยมีความคลาดเคลื่อนน้อยที่สุด ในโครงการนี้นำเอาค่าความต้องการออกซิเจนในทางเคมี (Chemistry Oxygen Demand) เป็นค่าที่ใช้สำหรับวัดคุณภาพน้ำจากการใช้ออกซิเจนของสารเคมีในแหล่งน้ำเพื่อทำให้เกิดปฏิกิริยาทางเคมีของสารเคมี การคาดการณ์ค่าความต้องการออกซิเจนในทางเคมีใช้ตัวแปรจำนวน 17 ตัวแปรในการคาดการณ์โดยพิจารณาโมเดลและพารามิเตอร์เพื่อให้ได้โมเดลและพารามิเตอร์ที่สามารถคาดการณ์ค่าได้คลาดเคลื่อนน้อยที่สุด

ผู้จัดทำขอขอบคุณอาจารย์พาพิศ วงศ์ชัยวัฒน์สำหรับคำแนะนำในการทำโครงการและการปรับปรุงแก้ไขโครงการดังกล่าวให้เหมาะสม ผู้จัดทำคาดหวังว่ารายงานเล่มนี้จะเป็นประโยชน์ในการศึกษาเกี่ยวกับการเขียนโปรแกรม การเรียนรู้ของเครื่อง (Machine Learning) การสร้างโมเดลสำหรับการเรียนรู้ของเครื่อง และ การศึกษาเกี่ยวกับปัญญาประดิษฐ์ (Artificial Intelligence) ในระดับการศึกษาสูงกว่านี้ หากรายงานเล่มนี้มีข้อผิดพลาดแต่ประการใด ผู้จัดทำขออภัยมา ณ ที่นี้

นายคุณานนต์ สุรสร

ผู้จัดทำ

## สารบัญ

เนื้อหา	หน้า
บทนำ.....	1
1. ที่มาและความสำคัญ.....	1
2. วัตถุประสงค์.....	1
3. ขอบเขต.....	2
ทฤษฎีที่เกี่ยวข้อง.....	3
1. การวัดคุณภาพน้ำ.....	3
2. โมเดลและพารามิเตอร์.....	3
3. การประเมินโมเดล.....	6
วิธีการดำเนินงาน.....	9
1. การค้นหาชุดข้อมูลสำหรับการศึกษา.....	9
2. ลักษณะของชุดข้อมูลคุณภาพน้ำของแม่น้ำน่านกตง.....	9
3. ตัวแปรของชุดข้อมูลคุณภาพแม่น้ำสายก่อนการปรับปรุงชุดข้อมูล.....	10
4. การจัดการชุดข้อมูล.....	12
5. การปรับปรุงชุดข้อมูลและการหาความสัมพันธ์ระหว่างตัวแปร.....	13
6. การหาโมเดลโดยวิธี Hyperparameter Tuning.....	19
ผลการวิเคราะห์ข้อมูล.....	23
1.การประเมินโมเดลหลักโดยใช้ข้อมูลทดสอบของแม่น้ำสายเดียว.....	23
2.การปรับปรุงโมเดล.....	25
3.การหาโมเดลของ GridsearchCV.....	27

เนื้อหา	หน้า
สรุปผลการดำเนินงานและข้อเสนอแนะ.....	30
1. ข้อสรุปผล.....	30
2. การนำไปใช้.....	30
3. ข้อเสนอแนะ.....	31
เอกสารอ้างอิง.....	32

## บทนำ

### 1. ที่มาและความสำคัญ

แหล่งน้ำบนผิวดินเช่น แม่น้ำ ทะเลสาบ ลำธาร คลองเป็นแหล่งน้ำที่ประกอบด้วยสิ่งมีชีวิตเช่นปลาน้ำจืด สัตว์น้ำ จุลินทรีย์และ ตะกอนหรือแร่ธาตุหลายประเภทไหลมาตามน้ำ ในปัจจุบันพื้นที่ของเมืองมีการขยายตัวมากขึ้นทำให้แหล่งน้ำหลายแห่งเป็นส่วนหนึ่งของพื้นที่เมืองด้วยการนำน้ำจืดจากแหล่งน้ำดังกล่าวไปทำเป็นน้ำประปาสำหรับใช้ในชุมชนและปล่อยน้ำจากชุมชนไหลกลับเข้าแหล่งน้ำ

การปล่อยน้ำจากชุมชนไหลกลับแหล่งน้ำ ในบางกรณีเป็นการปล่อยน้ำโดยไม่ผ่านการบำบัดก่อนปล่อยลงแหล่งน้ำทำให้น้ำที่ปล่อยมีสารเคมีเช่น โลหะหนัก สารอินทรีย์ น้ำมัน ไขมัน สารซักฟอก และ ธาตุอาหาร สารเคมีที่กล่าวมาทำให้ปริมาณที่ออกซิเจนที่จุลินทรีย์ต้องใช้มีมากขึ้น มีการเพิ่มจำนวนของพืชผิวน้ำในแหล่งน้ำทำให้พืชผิวน้ำบดบังแสงอาทิตย์ส่องลงไปใต้น้ำทำให้เกิดน้ำเสียขึ้นในแหล่งน้ำ

สภาวะมลพิษทางน้ำส่งผลกระทบต่อพื้นที่ที่มีการใช้น้ำจากแหล่งน้ำเช่น ชุมชน โรงงาน และ พื้นที่ทางการเกษตรไม่สามารถนำน้ำมาทำเป็นน้ำประปาได้ และ ภาครัฐต้องดำเนินการในหลายมาตรการเพื่อฟื้นฟูแหล่งน้ำเช่นการกำจัดพืชผิวน้ำ การเก็บขยะออกจากแหล่งน้ำ

ปัจจัยสำคัญที่ส่งผลกระทบต่อเกิดการเกิดน้ำเสียประกอบด้วยสารเคมีที่ปล่อยลงแหล่งน้ำ ปริมาณการใช้ออกซิเจนของสิ่งมีชีวิต โครงการจัดทำขึ้นเพื่อคาดการณ์ค่า Chemical Oxygen Demand (COD) ของแหล่งน้ำโดยใช้โมเดลที่สร้างขึ้นด้วยโปรแกรมคอมพิวเตอร์และคาดการณ์ค่า COD อ้างอิงจากชุดข้อมูลของแม่น้ำในประเทศเกาหลีใต้

### 2. วัตถุประสงค์

เพื่อหาโมเดลสำหรับคาดการณ์ค่าความต้องการใช้ออกซิเจนในทางเคมี (Chemical Oxygen Demand) และ โมเดลดังกล่าวสามารถคาดการณ์ค่าแม่นยำและค่าความคลาดเคลื่อนน้อยที่สุด

### 3. ขอบเขต

1. ชุดข้อมูลที่ใช้ในการศึกษาเป็นชุดข้อมูลของแม่น้ำในประเทศไทยได้

## ทฤษฎีที่เกี่ยวข้อง

### 1. การวัดคุณภาพน้ำ

ในแหล่งน้ำนั้นมีคุณภาพที่แตกต่างกันขึ้นกับสภาพทางกายภาพของแหล่งน้ำและปริมาณสารเคมีภายในแหล่งน้ำ น้ำเป็นสิ่งจำเป็นสำหรับมนุษย์เพราะคนนำน้ำมาใช้ในการบริโภค ในโรงงานอุตสาหกรรมมีการนำน้ำไปใช้ในกระบวนการผลิตเช่นการใช้น้ำสำหรับต้มสร้างไอน้ำเพื่อไปปั่นเครื่องกำเนิดไฟฟ้าให้เกิดกระแสไฟฟ้า

การประเมินคุณภาพน้ำนั้นมีหลายการประเมินเพื่อเป็นการตัดสินใจว่าแหล่งน้ำดังกล่าวสามารถนำไปใช้ในการอุปโภคบริโภคและใช้ในด้านอุตสาหกรรมโดยการประเมินคุณภาพน้ำมีการประเมินดังนี้

1. Biochemical Oxygen Demand เป็นค่าที่ใช้ประเมินปริมาณออกซิเจนที่แบคทีเรียและสิ่งมีชีวิตใช้ในการบริโภค
2. Chemical Oxygen Demand เป็นค่าที่ใช้ประเมินปริมาณออกซิเจนที่ทำให้สารเคมีในน้ำออกซิไดซ์ไปเป็นผลิตภัณฑ์ที่เป็นสารอนินทรีย์
3. pH เป็นค่าที่วัดความเป็นกรดเบสของน้ำที่นำมาศึกษา
4. Dissolved Oxygen เป็นค่าที่วัดปริมาณออกซิเจนที่ละลายในน้ำเพราะออกซิเจนเป็นสิ่งจำเป็นสำหรับสิ่งมีชีวิตในแหล่งน้ำเช่น พืช สัตว์ จุลินทรีย์ เป็นต้น

### 2. โมเดลและพารามิเตอร์

โมเดลเป็นสิ่งที่ใช้วัดผลลัพธ์ที่ขึ้นกับตัวแปรนำเข้ามาฟังก์ชันในลักษณะตัวเลขสำหรับโมเดลที่เป็น Regression Model ตัวแปรนำเข้ามาในลักษณะข้อความ ภาพ และ ตัวแปรนำเข้าในลักษณะอื่นนอกเหนือสำหรับโมเดลที่เป็น Classification Model ในการสร้างโมเดลนั้นต้องพิจารณาตัวแปรนำเข้า

พารามิเตอร์ที่ใช้สำหรับโมเดลเป็นเหมือนสิ่งที่กำหนดลักษณะของฟังก์ชันให้เป็นไปตามที่พารามิเตอร์กำหนด ในโครงงานนี้มีโมเดลที่ทำการศึกษาดังต่อไปนี้

## 2.1 Linear Regression มีเงื่อนไขการกำหนดลักษณะพารามิเตอร์ดังนี้

- 1) `fit_intercept`: *bool, default=True*
- 2) `copy_X`: *bool, default=True*
- 3) `n_jobs`: *int, default=None*
- 4) `positive`: *bool, default=False*

## 2.2 Ridge Linear Regression มีเงื่อนไขการกำหนดลักษณะพารามิเตอร์ดังนี้

- 1) `alpha`: *{float, ndarray of shape (n\_targets,)}, default=1.0*
- 2) `fit_intercept`: *bool, default=True*
- 3) `copy_X`: *bool, default=True*
- 4) `max_iter`: *int, default=None*
- 5) `tol`: *float, default=1e-4*
- 6) `solver`: *{'auto', 'svd', 'cholesky', 'lsqr', 'sparse\_cg', 'sag', 'saga', 'lbfgs'}, default='auto'*
- 7) `positive`: *bool, default=False*
- 8) `random_state`: *int, RandomState instance, default=None*

## 2.3 Lasso Linear Regression มีเงื่อนไขการกำหนดลักษณะพารามิเตอร์ดังนี้

- 1) `alpha`: *float, default=1.0*
- 2) `fit_intercept`: *bool, default=True*
- 3) `precompute`: *bool or array-like of shape (n\_features, n\_features), default=False*
- 4) `copy_X`: *bool, default=True*
- 5) `max_iter`: *int, default=1000*
- 6) `tol`: *float, default=1e-4*
- 7) `warm_start`: *bool, default=False*



8) positive: *bool, default=False*

9) random\_state: *int, RandomState instance, default=None*

10) selection: *{'cyclic', 'random'}, default='cyclic'*

## 2.4 Support Vector Regression มีเงื่อนไขการกำหนดลักษณะพารามิเตอร์ดังนี้

1) kernel: *{'linear', 'poly', 'rbf', 'sigmoid', 'precomputed'} or callable, default='rbf'*

2) degree: *int, default=3*

3) gamma: *{'scale', 'auto'} or float, default='scale'*

4) coef0: *float, default=0.0*

5) tol: *float, default=1e-3*

6) C: *float, default=1.0*

7) epsilon: *float, default=0.1*

8) shrinking: *bool, default=True*

9) cache\_size: *float, default=200*

10) verbose: *bool, default=False*

11) max\_iter: *int, default=-1*

## 2.5 K-Neighbors Regression มีเงื่อนไขการกำหนดลักษณะพารามิเตอร์ดังนี้

1) n\_neighbors: *int, default=5*

2) weights: *{'uniform', 'distance'}, callable or None, default='uniform'*

3) algorithm: *{'auto', 'ball\_tree', 'kd\_tree', 'brute'}, default='auto'*

4) leaf\_size: *int, default=30*

5) p: *float, default=2*

6) metric: *str or callable, default='minkowski'*

7) metric\_params: *dict, default=None*

8) `n_jobs: int, default=None`

## 2.6 Decision Tree Regression

1) `criterion: {"squared_error", "friedman_mse", "absolute_error", "poisson"}, default="squared_error"`

2) `splitter: {"best", "random"}, default="best"`

3) `max_depth: int, default=None`

4) `min_samples_split: int or float, default=2`

5) `min_samples_leaf: int or float, default=1`

6) `min_weight_fraction_leaf: float, default=0.0`

7) `max_features: int, float or {"auto", "sqrt", "log2"}, default=None`

8) `random_state: int, RandomState instance or None, default=None`

9) `max_leaf_nodes: int, default=None`

10) `min_impurity_decrease: float, default=0.0`

11) `ccp_alpha: non-negative float, default=0.0`

## 3. การประเมินโมเดล

โมเดลที่โครงการคาดหวังเป็นโมเดลที่สามารถคาดการณ์ค่าได้แม่นยำและมีความคลาดเคลื่อนน้อยในการประเมินโมเดลมี 3 การประเมินเช่น

### 3.1 Mean Absolute Error

เป็นการประเมินโมเดลโดยใช้ผลรวมค่าสัมบูรณ์ของผลต่างระหว่างค่าที่คาดการณ์ได้และค่าที่เก็บข้อมูลได้จริง

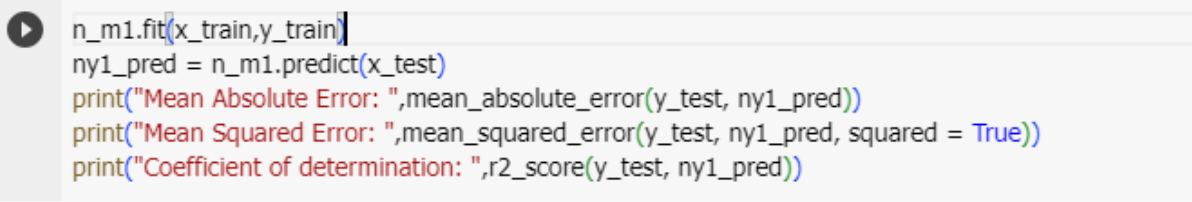
$$MAE = \frac{\sum_{i=1}^n |y_i - \hat{y}_i|}{n}$$

$y_i$  เป็นค่าจริงของข้อมูลลำดับที่  $i$

$\hat{y}_i$  เป็นค่าคาดการณ์ไว้ของข้อมูลลำดับที่  $i$

$n$  เป็นค่าจำนวนของข้อมูลที่เก็บไว้

ในการคำนวณค่า Mean Absolute Error โดยใช้คอมพิวเตอร์จะใช้คำสั่งสำหรับหาค่า MAE ดังนี้  
`mean_absolute_error(y_test,y_pred)` ตามภาพที่ 1 ค่า Mean Absolute Error ที่มีค่าน้อยแสดงว่า  
 ผลต่างระหว่างค่าที่คาดการณ์ได้และค่าที่เก็บข้อมูลได้จริงมีค่าน้อยมากดังนั้นโมเดลดังกล่าวสามารถคาดการณ์  
 ค่าได้แม่นยำ



```

n_m1.fit(x_train,y_train)
ny1_pred = n_m1.predict(x_test)
print("Mean Absolute Error: ",mean_absolute_error(y_test, ny1_pred))
print("Mean Squared Error: ",mean_squared_error(y_test, ny1_pred, squared = True))
print("Coefficient of determination: ",r2_score(y_test, ny1_pred))

```

Mean Absolute Error: 0.7987065697497872  
 Mean Squared Error: 1.217500835963755  
 Coefficient of determination: 0.8079131947702844

ภาพที่ 1 การใช้ชุดคำสั่งหา Mean Absolute Error

### 3.2 Mean Squared Error

เป็นการประเมินโมเดลโดยใช้ผลรวมค่าที่แตกต่างยกกำลังสองของค่าที่คาดการณ์ได้และค่าที่เก็บ  
 ข้อมูลได้จริง

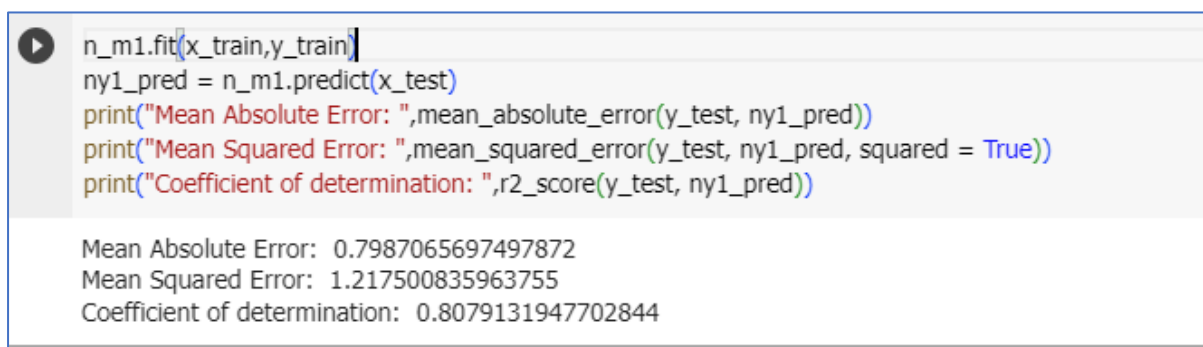
$$MSE = \frac{\sum_{i=1}^n (y_i - \hat{y})^2}{n}$$

$y_i$  เป็นค่าจริงของข้อมูลลำดับที่  $i$

$\hat{y}_i$  เป็นค่าคาดการณ์ไว้ของข้อมูลลำดับที่  $i$

$n$  เป็นค่าจำนวนของข้อมูลที่เก็บไว้

ในการคำนวณค่า Mean Squared Error โดยใช้คอมพิวเตอร์จะใช้คำสั่งสำหรับหาค่า MAE ดังนี้  
`mean_squared_error(y_test,y_pred)` ตามภาพที่ 2 ค่า Mean Squared Error ที่มีค่าน้อยแสดงว่าผลต่าง  
 ระหว่างค่าที่คาดการณ์ได้และค่าที่เก็บข้อมูลได้จริงมีค่าน้อยมากดังนั้นโมเดลดังกล่าวสามารถคาดการณ์ค่าได้  
 แม่นยำ



```

n_m1.fit(x_train,y_train)
ny1_pred = n_m1.predict(x_test)
print("Mean Absolute Error: ",mean_absolute_error(y_test, ny1_pred))
print("Mean Squared Error: ",mean_squared_error(y_test, ny1_pred, squared = True))
print("Coefficient of determination: ",r2_score(y_test, ny1_pred))

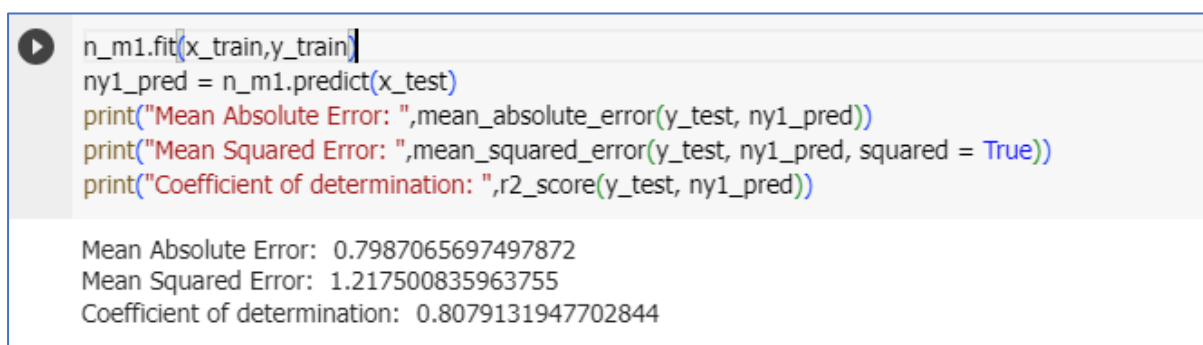
```

Mean Absolute Error: 0.7987065697497872  
 Mean Squared Error: 1.217500835963755  
 Coefficient of determination: 0.8079131947702844

ภาพที่ 2 การใช้ชุดคำสั่งหา Mean Squared Error

### 3.3 Coefficient of Determination

ในการหาค่า Coefficient of Determination โดยใช้คอมพิวเตอร์จะใช้คำสั่งสำหรับหาค่า MAE ดังนี้  
`r2_score(y_test,y_pred)` ตามภาพที่ 3 ค่า Coefficient of Determination ที่มีค่าเป็น 1 หมายความว่า  
 โมเดลนั้นสามารถคาดการณ์ได้ดีตามตัวแปรที่มีในโมเดลดังกล่าว



```

n_m1.fit(x_train,y_train)
ny1_pred = n_m1.predict(x_test)
print("Mean Absolute Error: ",mean_absolute_error(y_test, ny1_pred))
print("Mean Squared Error: ",mean_squared_error(y_test, ny1_pred, squared = True))
print("Coefficient of determination: ",r2_score(y_test, ny1_pred))

```

Mean Absolute Error: 0.7987065697497872  
 Mean Squared Error: 1.217500835963755  
 Coefficient of determination: 0.8079131947702844

ภาพที่ 3 การใช้ชุดคำสั่งหา Coefficient of Determination

## การดำเนินงาน

### 1. การค้นหาชุดข้อมูลสำหรับการศึกษา

ในการศึกษาเรื่องการคาดการณ์ค่า Chemical oxygen demand ใช้ชุดข้อมูลของบริการข้อมูลทางสถิติของเกาหลี (Korean Statistical Information Service) เป็นเว็บไซต์ในภาพที่ 4 เว็บไซต์นี้เก็บข้อมูลทางสถิติของประเทศเกาหลีได้เกี่ยวกับเศรษฐกิจ สิ่งแวดล้อม สังคม ประชากร เป็นต้น

KOSIS

Statistics Korea

Statistical List : Statistical Data

My History

Statistics Search

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

go top

ภาพที่ 4 เว็บไซต์ของบริการข้อมูลทางสถิติของเกาหลี

### 2. ลักษณะของชุดข้อมูลคุณภาพน้ำของแม่น้ำนังกอง

- 1) ชุดของข้อมูลมาจาก Korean Statistical Information Service (KOSIS)
- 2) แม่น้ำนังกองเป็นแม่น้ำสายหนึ่งในประเทศเกาหลีใต้ยาว 510 กิโลเมตรมีต้นกำเนิดจากภูเขาแทแบกและไหลออกสู่ทะเลญี่ปุ่นที่เมืองปูซาน
- 3) ชุดข้อมูลคุณสมบัติของน้ำในแม่น้ำนังกองประกอบด้วยข้อมูลคุณสมบัติของน้ำที่แม่น้ำนังกองในช่วงเดือนมิถุนายนถึงเดือนสิงหาคมของปี 2017 ถึง 2020 ที่เป็นฤดูร้อนในประเทศเกาหลีใต้

## 4) ในชุดข้อมูลมีคอลัมน์ดังนี้

ตารางคอลัมน์ของชุดข้อมูลคุณสมบัติของน้ำที่แม่น้ำน่านกวด

คอลัมน์ของชุดข้อมูลคุณสมบัติของน้ำที่แม่น้ำน่านกวด		
Biochemical Oxygen Demand (BOD)	Dissolved Oxygen (DO)	Chemical Oxygen Demand (COD)
Water Temperature	Hydrogen ion Concentration	Suspended Solids (SS)
Total Nitrogen (T-N)	Total Phosphorus (T-S)	Fecal Coliform Bacteria
Coliform Bacteria	Electrical Conductivity	Anionic Surfactant (ABS)
Phenol	Cadmium	Hydrargyrum (Hg)
Lead (Pb)	6 가크롬 Cr+6 (Hexavalent chromium Cr+6)	Arsenic (As)
Hydrargyrum (Hg)	Nitrate Nitrogen (NO <sub>3</sub> -N)	Ammonia Nitrogen (NH <sub>3</sub> -N)
Phosphate phosphorus (PO <sub>4</sub> -P)	Chlorophyll a	Dissolved Total Nitrogen (DTN)
Dissolved Total Phosphorus (DTP)	Copper (Cu)	TOC
Antimony	Flow	

## 3. ตัวแปรของชุดข้อมูลคุณภาพแม่น้ำสายก่อนการปรับปรุงชุดข้อมูล

ในชุดข้อมูลของบริการข้อมูลทางสถิติของเกาหลี (Korean Statistical Information Service) มีตัวแปร 38 ตัวแปรประกอบด้วยข้อมูลตัวเลขเกี่ยวกับข้อมูลทางกายภาพของแหล่งน้ำและการวัดปริมาณสารเคมีของแม่น้ำที่มีการเก็บข้อมูล

1. ข้อมูลทางกายภาพของแหล่งน้ำเป็นข้อมูลเกี่ยวกับการวัดสภาพของแหล่งน้ำและการเก็บข้อมูลคุณภาพน้ำจากจุดที่มีการเก็บข้อมูลในแต่ละจุดโดยมีตัวแปร 12 ตัวแปรดังนี้

ตารางแสดงตัวแปรของข้อมูลทางกายภาพของแหล่งน้ำ

1. By water system (1)	2. By water system (2)	3. By water system (3)
4. Water Temperature	5. Fecal Coliform Bacteria	6. Coliform Bacteria
7. Electrical Conductivity (EC)	8. Anionic Surfactant (ABS)	9. Chlorophyll a
10. Flow	11. Month	12. Year

2. ข้อมูลการวัดปริมาณสารเคมีเป็นข้อมูลเกี่ยวกับการวัดปริมาณสารเคมีในแหล่งน้ำในจุดที่มีการเก็บข้อมูลแต่ละจุดมีตัวแปร 26 ตัวแปรดังนี้

ตารางแสดงตัวแปรของข้อมูลการวัดปริมาณสารเคมี

1. Biochemical Oxygen Demand (BOD)	2. Dissolved Oxygen (DO)	3. Chemical Oxygen Demand (COD)
4. Hydrogen ion Concentration	5. Suspended Solids (SS)	6. Total Nitrogen(T-N)
7. Total Phosphorus (T-P)	8. phenol	9. Cadmium (Cd)
10. Cyanogen (CN)	11. Lead (Pb)	12. Hexavalent chromium Cr+6
13. Arsenic (As)	14. Hydrargyrum (Hg)	15. Nitrate Nitrogen (NO <sub>3</sub> -N)
16. Ammonia Nitrogen (NH <sub>3</sub> -N)	17. Phosphate-phosphorus (PO <sub>4</sub> -P)	18. Dissolved Total Nitrogen (DTN)
19. Dissolved Total Phosphorus (DTP)	20. TOC	21. Antimony
22. Organic Phosphorus	23. Polychlorinated Biphenyl (PCB)	24. Tetrachloroethylene (PCE)
25. Depth of water	26. DHEP	

#### 4.การจัดการชุดข้อมูล

- 1) ชุดข้อมูลมี 2173 แถว และ 35 คอลัมน์โดยนำข้อมูลคุณภาพน้ำในเดือนมิถุนายนถึงเดือนสิงหาคมระหว่างปี 2017 – 2020
- 2) ชุดข้อมูลทั้งหมดถูกรวมโดยใช้ pd.concat() ตามภาพที่ 5
- 3) ชุดข้อมูลมีการแสดงผลทางสถิติเช่นจำนวนชุดข้อมูล ค่าเฉลี่ย ส่วนเบี่ยงเบนมาตรฐาน ค่าต่ำสุด ค่าเปอร์เซ็นต์ไทล์ที่ 25 ค่าเปอร์เซ็นต์ไทล์ที่ 50 ค่าเปอร์เซ็นต์ไทล์ที่ 75 และ ค่าสูงสุดตามภาพที่ 6

```
[3] df_June2017 = pd.read_csv('/content/drive/MyDrive/Code_MIS/June 2017.csv')
df_July2017 = pd.read_csv('/content/drive/MyDrive/Code_MIS/July 2017.csv')
df_August2017 = pd.read_csv('/content/drive/MyDrive/Code_MIS/August 2017.csv')
df_June2018 = pd.read_csv('/content/drive/MyDrive/Code_MIS/June 2018.csv')
df_July2018 = pd.read_csv('/content/drive/MyDrive/Code_MIS/July 2018.csv')
df_August2018 = pd.read_csv('/content/drive/MyDrive/Code_MIS/August 2018.csv')
df_June2019 = pd.read_csv('/content/drive/MyDrive/Code_MIS/June 2019.csv')
df_July2019 = pd.read_csv('/content/drive/MyDrive/Code_MIS/July 2019.csv')
df_August2019 = pd.read_csv('/content/drive/MyDrive/Code_MIS/August 2019.csv')
df_June2020 = pd.read_csv('/content/drive/MyDrive/Code_MIS/June 2020.csv')
df_July2020 = pd.read_csv('/content/drive/MyDrive/Code_MIS/July 2020.csv')
df_August2020 = pd.read_csv('/content/drive/MyDrive/Code_MIS/August 2020.csv')

[4] frames = [df_June2017,df_July2017,df_August2017,df_June2018,df_July2018,df_August2018,df_June2019,df_July2019,df_August2019,df_June2020,df_July2020,df_August2020]

[5] raw_dataset = pd.concat(frames)
```

ภาพที่ 5 ชุดข้อมูลสำหรับการศึกษา

	Biochemical Oxygen Demand(BOD) (mg/l)	Dissolved Oxygen(DO) (mg/l)	Chemical Oxygen Demand(COD) (mg/l)	Water Temperature (Celcius)	Hydrogen ion Concentration (pH)	Suspended Solids(SS) (mg/l)	Total Nitrogen(T- N) (mg/l)	Total Phosphorus(T- P) (mg/l)	Fecal Coliform Bacteria (Fecal Coliform Bacteria/100ml)	Coliform Bacteria (Coliform Bacteria/100ml)
count	2173.000000	2173.000000	2173.000000	2173.000000	2173.000000	2173.000000	2173.000000	2173.000000	2173.000000	2.173000e+03
mean	1.666636	12.506121	5.986838	25.348780	7.912103	10.894202	2.215174	0.059618	1756.492407	3.755906e+04
std	1.089332	178.965458	2.477643	3.201597	0.491789	17.831841	1.339178	0.057193	5348.233037	9.579292e+04
min	0.100000	4.000000	1.000000	10.000000	6.400000	0.200000	0.282000	0.000000	0.000000	0.000000e+00
25%	0.900000	8.000000	4.300000	23.500000	7.600000	3.600000	1.413000	0.026000	40.000000	2.000000e+03
50%	1.400000	8.600000	5.800000	25.500000	7.900000	6.700000	1.957000	0.046000	740.000000	2.400000e+04
75%	2.100000	9.300000	7.300000	27.500000	8.200000	12.600000	2.585000	0.074000	1756.000000	3.755900e+04
max	8.900000	8351.000000	27.200000	36.000000	11.300000	438.000000	13.632000	0.832000	156000.000000	1.975000e+06

8 rows x 28 columns

ภาพที่ 6 ค่าทางสถิติของชุดข้อมูล



## 5. การปรับปรุงชุดข้อมูลและการหาความสัมพันธ์ระหว่างตัวแปร

### 3.1 การปรับปรุงชุดข้อมูล

#	Column	Non-Null Count	Dtype
0	By Water System(1)	2173 non-null	object
1	By Water System(2)	2173 non-null	object
2	By Water System(3)	2173 non-null	object
3	Biochcmical Oxygen Demand(BOD) (mg/l)	2173 non-null	float64
4	Dissolved Oxygen(DO) (mg/l)	2173 non-null	float64
5	Chemical Oxygen Demand(COD) (mg/l)	2173 non-null	float64
6	Water Temperature (Celcius)	2173 non-null	float64
7	Hydrogen ion Concentration (pH)	2173 non-null	float64
8	Suspended Solids(SS) (mg/l)	2173 non-null	float64
9	Total Nitrogen(T-N) (mg/l)	2173 non-null	float64
10	Total Phosphorus(T-P) (mg/l)	2173 non-null	float64
11	Fecal Coliform Bacteria (Fecal Coliform Bacteria/100ml)	1503 non-null	float64
12	Coliform Bacteria (Coliform Bacteria/100ml)	1503 non-null	float64
13	Electrical Conductivity(EC) (umhos/cm)	2173 non-null	int64
14	Anionic Surfactant(ABS) (mg/l)	640 non-null	float64
15	phenol (mg/l)	1470 non-null	float64
16	Cadmium(Cd) (mg/l)	668 non-null	float64
17	Hydrargyrum(Hg) (mg/l)	644 non-null	float64
18	Lead(Pb) (mg/l)	668 non-null	float64
19	Hexavalent chromium Cr+6 (mg/l)	668 non-null	float64
20	Arsenic(As) (mg/l)	668 non-null	float64
21	Hydrargyrum(Hg) (mg/l).1	665 non-null	float64
22	Nitrate Nitrogen(NO3-N) (mg/l)	1612 non-null	float64
23	Ammonia Nitrogen(NH3-N) (mg/l)	1611 non-null	float64
24	Phosphate-phosphorus(PO4-P) (mg/l)	1618 non-null	float64
25	Chlorophyll a (mg/m^3)	1618 non-null	float64
26	Dissolved Total Nitrogen(DTN) (mg/l)	1618 non-null	float64
27	Dissolved Total Phosphorus(DTP) (mg/l)	1618 non-null	float64
28	TOC (mg/l)	2163 non-null	float64
29	Antimony (mg/l)	668 non-null	float64
30	Flow (m^3/sec)	1078 non-null	float64
31	Month	2173 non-null	object
32	Year	2173 non-null	int64
33	Organic Phosphorus (mg/l)	104 non-null	float64
34	Polychlorinated Biphenyl(PCB) (mg/l)	104 non-null	float64

ภาพที่ 7 จำนวนข้อมูลในชุดข้อมูลแบ่งตามคอลัมน์

1) จากภาพที่ 7 ในคอลัมน์ทั้งหมดของชุดข้อมูลนี้ มีบางคอลัมน์ไม่ได้มีจำนวนข้อมูล 2173 ข้อมูล และบางคอลัมน์ไม่ได้มีการเก็บข้อมูลในบางเดือนเช่นคอลัมน์ Organic Phosphorus และ Polychlorinated Biphenyl ที่มีการเก็บข้อมูลในเดือนกรกฎาคมของปี 2017 ถึง 2020 เท่านั้น

2) ค่าที่เป็น NaN ในบางคอลัมน์ถูกแทนที่ด้วยค่าเฉลี่ยของค่าทั้งหมดของคอลัมน์แต่ละคอลัมน์ เพราะในชุดข้อมูลนี้เป็นชุดข้อมูลที่ศึกษาแม่น้ำสายเดียวในหลายจุดตามภาพที่ 8

```
main_dataset['Fecal Coliform Bacteria (Fecal Coliform Bacteria/100ml)'].fillna(int(main_dataset['Fecal Coliform Bacteria (Fecal Coliform Bacteria/100ml)'].mean()), inplace=True)
main_dataset['Coliform Bacteria (Coliform Bacteria/100ml)'].fillna(int(main_dataset['Coliform Bacteria (Coliform Bacteria/100ml)'].mean()), inplace=True)
main_dataset['Anionic Surfactant(ABS) (mg/l)'].fillna(int(main_dataset['Anionic Surfactant(ABS) (mg/l)'].mean()), inplace=True)
main_dataset['phenol (mg/l)'].fillna(int(main_dataset['phenol (mg/l)'].mean()), inplace=True)
main_dataset['Cadmium(Cd) (mg/l)'].fillna(int(main_dataset['Cadmium(Cd) (mg/l)'].mean()), inplace=True)
main_dataset['Hydrargyrum(Hg) (mg/l)'].fillna(int(main_dataset['Hydrargyrum(Hg) (mg/l)'].mean()), inplace=True)
main_dataset['Lead(Pb) (mg/l)'].fillna(int(main_dataset['Lead(Pb) (mg/l)'].mean()), inplace=True)
main_dataset['Hexavalent chromium Cr+6 (mg/l)'].fillna(int(main_dataset['Hexavalent chromium Cr+6 (mg/l)'].mean()), inplace=True)
main_dataset['Arsenic(As) (mg/l)'].fillna(int(main_dataset['Arsenic(As) (mg/l)'].mean()), inplace=True)
main_dataset['Nitrate Nitrogen(NO3-N) (mg/l)'].fillna(int(main_dataset['Nitrate Nitrogen(NO3-N) (mg/l)'].mean()), inplace=True)
main_dataset['Ammonia Nitrogen(NH3-N) (mg/l)'].fillna(int(main_dataset['Ammonia Nitrogen(NH3-N) (mg/l)'].mean()), inplace=True)
main_dataset['Phosphate-phosphorus(PO4-P) (mg/l)'].fillna(int(main_dataset['Phosphate-phosphorus(PO4-P) (mg/l)'].mean()), inplace=True)
main_dataset['Chlorophyll a (mg/m^3)'].fillna(int(main_dataset['Chlorophyll a (mg/m^3)'].mean()), inplace=True)
main_dataset['Dissolved Total Nitrogen(DTN) (mg/l)'].fillna(int(main_dataset['Dissolved Total Nitrogen(DTN) (mg/l)'].mean()), inplace=True)
main_dataset['Dissolved Total Phosphorus(DTP) (mg/l)'].fillna(int(main_dataset['Dissolved Total Phosphorus(DTP) (mg/l)'].mean()), inplace=True)
main_dataset['TOC (mg/l)'].fillna(int(main_dataset['TOC (mg/l)'].mean()), inplace=True)
main_dataset['Antimony (mg/l)'].fillna(int(main_dataset['Antimony (mg/l)'].mean()), inplace=True)
main_dataset['Flow (m^3/sec)'].fillna(int(main_dataset['Flow (m^3/sec)'].mean()), inplace=True)
```

ภาพที่ 8 การเติมค่าเฉลี่ยของแต่ละคอลัมน์แทนค่าที่เป็น NaN

3) คอลัมน์ Organic Phosphorus และ Polychlorinated Biphenyl ไม่ได้ถูกนำไปใช้ในการวิเคราะห์เพราะข้อมูลของคอลัมน์มีข้อมูลในเดือนกรกฎาคมของปี 2017 ถึง 2020

4) คอลัมน์ Year มีชนิดตัวแปรเป็น INT เปลี่ยนเป็น Object เพราะไม่ได้ถูกนำมาใช้ในการคำนวณ

5) จากข้อที่กล่าวมาเมื่อดำเนินการแล้ว จำนวนค่าในคอลัมน์มีทั้งหมด 2173 ค่าทุกคอลัมน์

ชุดข้อมูลคุณภาพน้ำมีข้อมูลเกี่ยวกับลักษณะทางกายภาพของแหล่งน้ำและปริมาณสารเคมีจำนวน คอลัมน์ 36 คอลัมน์ ในแม่น้ำบางสายมีการเก็บข้อมูลชุดข้อมูลไม่ได้มีการเก็บข้อมูล 36 คอลัมน์และข้อมูลที่เก็บได้มีจำนวนน้อยทำให้ไม่สามารถนำชุดข้อมูลไปใช้ในการประมวลผลได้

การเก็บข้อมูลคุณภาพน้ำในช่วงเดือนมิถุนายนถึงเดือนสิงหาคมของปี 2017 ถึง 2020 พบว่าชุดข้อมูลของแม่น้ำนกกดมีคอลัมน์ 35 คอลัมน์ ชุดข้อมูลของแม่น้ำยงขันมีคอลัมน์ 34 คอลัมน์ ชุดข้อมูลของแม่น้ำฮั่นมีคอลัมน์ 36 คอลัมน์ และ ชุดข้อมูลของแม่น้ำกิมมีคอลัมน์ 33 คอลัมน์ บางคอลัมน์ของชุดข้อมูลแม่น้ำบางคอลัมน์มีข้อมูลหายไปและไม่ครบตามที่กำหนด

เพื่อเป็นการปรับปรุงชุดข้อมูลให้มีข้อมูลครบตามที่กำหนดและสามารถนำไปใช้ในการประมวลผลได้ จึงมีการดำเนินการปรับปรุงชุดข้อมูล 3 วิธีการดังนี้

#### 1. การตัดคอลัมน์ที่มีจำนวนข้อมูลน้อยมาก

31 Organic Phosphorus (mg/l)	70 non-null	float64
32 Polychlorinated Biphenyl(PCB) (mg/l)	57 non-null	float64
33 Tetrachloroethylene(PCE) (mg/l)	13 non-null	float64
34 Depth of water (m)	1 non-null	float64
35 DEHP (mg/l)	13 non-null	float64

ภาพที่ 9 คอลัมน์ที่ 31 ถึง 35 ของชุดข้อมูลคุณภาพน้ำแม่น้ำฮั่น

ชุดข้อมูลของแม่น้ำบางชุดข้อมูลมีจำนวนข้อมูลน้อยมากทำให้ไม่สามารถนำไปทำการประมวลผลได้ เช่นในกรณีของข้อมูลคุณภาพน้ำของแม่น้ำฮั่นมีคอลัมน์บางคอลัมน์ที่ข้อมูลมีน้อยมากเช่น ในภาพที่ 9 คอลัมน์ที่ 31 Organic Phosphorus (mg/l) มีข้อมูล 70 ข้อมูลคอลัมน์ที่ 34 Depth of water (m) มีข้อมูล 1 ข้อมูล คอลัมน์ที่ 35 DEHP มีข้อมูล 13 ข้อมูลจากข้อมูลที่ควรมีในคอลัมน์ 1139 ข้อมูล ด้วยเหตุผลนี้จึงได้ตัดคอลัมน์ดังกล่าวออกไปจากชุดข้อมูลโดยใช้ชุดคำสั่ง drop() ตามภาพที่ 10

```
main_dataset1.drop(['Anionic Surfactant(ABS) (mg/l)',
                    'Cadmium(Cd) (mg/l)',
                    'Lead(Pb) (mg/l)',
                    'Hexavalent chromium Cr+6 (mg/l)',
                    'Arsenic(As) (mg/l)',
                    'Antimony (mg/l)',
                    'Flow (m^3/sec)'], axis=1, inplace=True)
```

ภาพที่ 10 การตัดคอลัมน์ด้วยชุดคำสั่ง drop()

#### 2. การเติมค่าแทนข้อมูลที่ไม่ได้ระบุไว้ในคอลัมน์

9 Total Nitrogen(T-N) (mg/l)	1139 non-null	float64
10 Total Phosphorus(T-P) (mg/l)	1139 non-null	float64
11 Fecal Coliform Bacteria (Fecal Coliform Bacteria/100ml)	983 non-null	float64
12 Coliform Bacteria (Coliform Bacteria/100ml)	983 non-null	float64
13 Electrical Conductivity(EC) (umhos/cm)	1139 non-null	float64

ภาพที่ 11 คอลัมน์ที่ 9 ถึง 13 ของชุดข้อมูลคุณภาพน้ำแม่น้ำฮั่น

ในบางคอลัมน์มีข้อมูลไม่ครบตามจำนวนเช่นในกรณีของข้อมูลคุณภาพน้ำของแม่น้ำอันมีคอลัมน์ที่มีข้อมูลหายไปเช่น ภาพที่ 11 คอลัมน์ที่ 11 Coliform Bacteria (Fecal Coliform Bacteria/100ml) มีข้อมูล 983 ข้อมูล และ คอลัมน์ที่ 12 Fecal Coliform Bacteria (Fecal Coliform Bacteria/100ml) มีข้อมูล 983 ข้อมูลจากข้อมูลที่ควรมีในคอลัมน์ 1139 ข้อมูลจึงได้ดำเนินการเติมค่าที่หายไปของข้อมูลโดยใช้ชุดคำสั่ง fillna() ในภาพที่ 12

```
main_dataset1['Fecal Coliform Bacteria (Fecal Coliform Bacteria/100ml)'].fillna(int(main_dataset1['Fecal Coliform Bacteria (Fecal Coliform Bacteria/100ml)'].mean()), inplace=True)
main_dataset1['Coliform Bacteria (Coliform Bacteria/100ml)'].fillna(int(main_dataset1['Coliform Bacteria (Coliform Bacteria/100ml)'].mean()), inplace=True)
main_dataset1['Nitrate Nitrogen(NO3-N) (mg/l)'].fillna(int(main_dataset1['Nitrate Nitrogen(NO3-N) (mg/l)'].mean()), inplace=True)
main_dataset1['Ammonia Nitrogen(NH3-N) (mg/l)'].fillna(int(main_dataset1['Ammonia Nitrogen(NH3-N) (mg/l)'].mean()), inplace=True)
main_dataset1['Phosphate-phosphorus(PO4-P) (mg/l)'].fillna(int(main_dataset1['Phosphate-phosphorus(PO4-P) (mg/l)'].mean()), inplace=True)
main_dataset1['Chlorophyll a (mg/m^3)'].fillna(int(main_dataset1['Chlorophyll a (mg/m^3)'].mean()), inplace=True)
main_dataset1['Dissolved Total Nitrogen(DTN) (mg/l)'].fillna(int(main_dataset1['Dissolved Total Nitrogen(DTN) (mg/l)'].mean()), inplace=True)
main_dataset1['Dissolved Total Phosphorus(DTP) (mg/l)'].fillna(int(main_dataset1['Dissolved Total Phosphorus(DTP) (mg/l)'].mean()), inplace=True)
main_dataset1['TOC (mg/l)'].fillna(int(main_dataset1['TOC (mg/l)'].mean()), inplace=True)
```

ภาพที่ 12 การเติมค่าด้วยชุดคำสั่ง fillna()

ชุดข้อมูลจำนวน 38 คอลัมน์ที่ผ่านการตัดคอลัมน์ที่มีจำนวนข้อมูลน้อยมากและการเติมค่าแทนข้อมูลที่ไม่ได้ระบุไว้ในคอลัมน์เหลือจำนวนคอลัมน์ 18 คอลัมน์ของภาพที่ 13 แบ่งออกเป็น 17 ตัวแปรต้นในลำดับที่ 0 ถึง 16 และ 1 ตัวแปรตามในลำดับที่ 17

#	Column	Non-Null Count	Dtype
0	Biochemical Oxygen Demand(BOD) (mg/l)	2173 non-null	float64
1	Dissolved Oxygen(DO) (mg/l)	2173 non-null	float64
2	Water Temperature (Celcius)	2173 non-null	float64
3	Hydrogen ion Concentration (pH)	2173 non-null	float64
4	Suspended Solids(SS) (mg/l)	2173 non-null	float64
5	Total Nitrogen(T-N) (mg/l)	2173 non-null	float64
6	Total Phosphorus(T-P) (mg/l)	2173 non-null	float64
7	Fecal Coliform Bacteria (Fecal Coliform Bacteria/100ml)	2173 non-null	float64
8	Coliform Bacteria (Coliform Bacteria/100ml)	2173 non-null	float64
9	Electrical Conductivity(EC) (umhos/cm)	2173 non-null	float64
10	Nitrate Nitrogen(NO3-N) (mg/l)	2173 non-null	float64
11	Ammonia Nitrogen(NH3-N) (mg/l)	2173 non-null	float64
12	Phosphate-phosphorus(PO4-P) (mg/l)	2173 non-null	float64
13	Chlorophyll a (mg/m^3)	2173 non-null	float64
14	Dissolved Total Nitrogen(DTN) (mg/l)	2173 non-null	float64
15	Dissolved Total Phosphorus(DTP) (mg/l)	2173 non-null	float64
16	TOC (mg/l)	2173 non-null	float64
17	Chemical Oxygen Demand(COD) (mg/l)	2173 non-null	float64

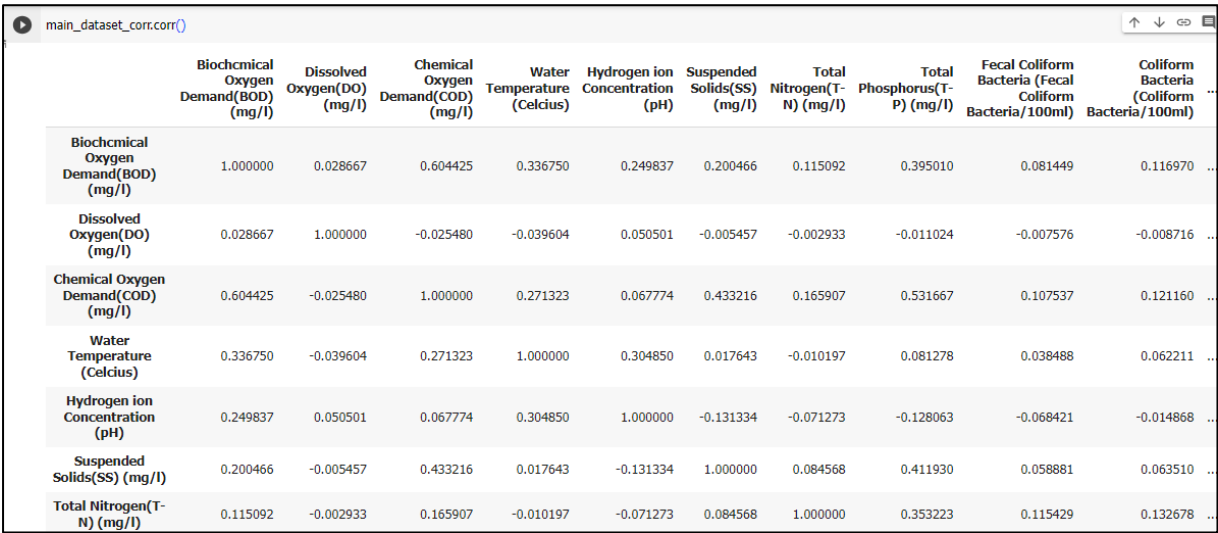
ภาพที่ 13 ชุดข้อมูลจำนวน 18 คอลัมน์

### 3.2 การหาความสัมพันธ์ระหว่างตัวแปร

1) คอลัมน์ By Water System (1), By Water System (2), By Water System (3), Month และ Year ถูกเอาออกไปเพราะชนิดของข้อมูลเป็น Object ไม่สามารถนำมาคำนวณหา Correlation Matrix

2) ชุดข้อมูลที่มีค่าเป็นตัวเลขถูกนำมาคำนวณเพื่อสร้าง Correlation Matrix ด้วยคำสั่ง `corr()` ผลลัพธ์ที่ได้คือ ตาราง Correlation Matrix ของตัวแปรหนึ่งสัมพันธ์กับตัวแปรอื่นทั้งหมดตามภาพที่ 14

3) ในบางกรณีสามารถหาค่าสหสัมพันธ์ของคอลัมน์หนึ่งเทียบกับคอลัมน์ทั้งหมดของภาพที่ 14 เป็นการเปรียบเทียบค่าสหสัมพันธ์ของ Chemical Oxygen Demand (COD) ว่ามีค่าสหสัมพันธ์กับคอลัมน์อื่นตามภาพที่ 15



The screenshot shows a Jupyter Notebook interface with a code cell containing `main_dataset_corr()` and its output, a Correlation Matrix. The matrix is a square table where each row and column represents a different water quality parameter, and the cells contain the Pearson correlation coefficient between them. The diagonal elements are all 1.000000, indicating perfect self-correlation. The parameters included are Biochemical Oxygen Demand (BOD), Dissolved Oxygen (DO), Chemical Oxygen Demand (COD), Water Temperature, Hydrogen ion Concentration (pH), Suspended Solids (SS), Total Nitrogen (T-N), Total Phosphorus (T-P), Fecal Coliform Bacteria, and Coliform Bacteria.

	Biochemical Oxygen Demand(BOD) (mg/l)	Dissolved Oxygen(DO) (mg/l)	Chemical Oxygen Demand(COD) (mg/l)	Water Temperature (Celcius)	Hydrogen ion Concentration (pH)	Suspended Solids(SS) (mg/l)	Total Nitrogen(T-N) (mg/l)	Total Phosphorus(T-P) (mg/l)	Fecal Coliform Bacteria (Fecal Coliform Bacteria/100ml)	Coliform Bacteria (Coliform Bacteria/100ml)
Biochemical Oxygen Demand(BOD) (mg/l)	1.000000	0.028667	0.604425	0.336750	0.249837	0.200466	0.115092	0.395010	0.081449	0.116970
Dissolved Oxygen(DO) (mg/l)	0.028667	1.000000	-0.025480	-0.039604	0.050501	-0.005457	-0.002933	-0.011024	-0.007576	-0.008716
Chemical Oxygen Demand(COD) (mg/l)	0.604425	-0.025480	1.000000	0.271323	0.067774	0.433216	0.165907	0.531667	0.107537	0.121160
Water Temperature (Celcius)	0.336750	-0.039604	0.271323	1.000000	0.304850	0.017643	-0.010197	0.081278	0.038488	0.062211
Hydrogen ion Concentration (pH)	0.249837	0.050501	0.067774	0.304850	1.000000	-0.131334	-0.071273	-0.128063	-0.068421	-0.014868
Suspended Solids(SS) (mg/l)	0.200466	-0.005457	0.433216	0.017643	-0.131334	1.000000	0.084568	0.411930	0.058881	0.063510
Total Nitrogen(T-N) (mg/l)	0.115092	-0.002933	0.165907	-0.010197	-0.071273	0.084568	1.000000	0.353223	0.115429	0.132678

ภาพที่ 14 Correlation Matrix

[23] COD_corr	
Biochcmical Oxygen Demand(BOD) (mg/l)	0.604425
Dissolved Oxygen(DO) (mg/l)	-0.025480
Chemical Oxygen Demand(COD) (mg/l)	1.000000
Water Temperature (Celcius)	0.271323
Hydrogen ion Concentration (pH)	0.067774
Suspended Solids(SS) (mg/l)	0.433216
Total Nitrogen(T-N) (mg/l)	0.165907
Total Phosphorus(T-P) (mg/l)	0.531667
Fecal Coliform Bacteria (Fecal Coliform Bacteria/100ml)	0.107537
Coliform Bacteria (Coliform Bacteria/100ml)	0.121160
Electrical Conductivity(EC) (umhos/cm)	0.327931
Anionic Surfactant(ABS) (mg/l)	-0.006816
phenol (mg/l)	NaN
Cadmium(Cd) (mg/l)	-0.054729
Hydrargyrum(Hg) (mg/l)	NaN
Lead(Pb) (mg/l)	0.015038
Hexavalent chromium Cr+6 (mg/l)	-0.024140
Arsenic(As) (mg/l)	-0.064583
Nitrate Nitrogen(NO3-N) (mg/l)	0.033126
Ammonia Nitrogen(NH3-N) (mg/l)	0.302631
Phosphorate-phosphorus(PO4-P) (mg/l)	0.383133
Chlorophyll a (mg/m <sup>3</sup> )	0.506554
Dissolved Total Nitrogen(DTN) (mg/l)	0.167644
Dissolved Total Phosphorus(DTP) (mg/l)	0.405044
TOC (mg/l)	0.832273
Antimony (mg/l)	0.025927
Flow (m <sup>3</sup> /sec)	0.092017
dtype: float64	

ภาพที่ 15 ค่าสหสัมพันธ์ของ Chemical Oxygen Demand (COD) เทียบกับคอลัมน์อื่น

4) จากภาพที่ 15 คอลัมน์ Chemical Oxygen Demand (COD) มีค่าสหสัมพันธ์สูงกับคอลัมน์ TOC หรือ Total Organic Carbon ข้อสรุปคือ ค่า Total Organic Carbon มีผลทำให้ค่า Chemical Oxygen Demand (COD) เพิ่มขึ้น

5) คอลัมน์ Chemical Oxygen Demand (COD) มีค่าสหสัมพันธ์ปานกลางกับคอลัมน์ BOD หรือ Biochemical Oxygen Demand ข้อสรุปคือ ค่า Biochemical Oxygen Demand มีผลทำให้ค่า Chemical Oxygen Demand (COD) เพิ่มขึ้น

6) คอลัมน์ Chemical Oxygen Demand (COD) มีค่าสหสัมพันธ์ปานกลางกับคอลัมน์ T-P หรือ Total Phosphorus ข้อสรุปคือ ค่า Total Phosphorus มีผลทำให้ค่า Chemical Oxygen Demand (COD) เพิ่มขึ้น

7) คอลัมน์ของชุดข้อมูลจำนวนมากมีค่าสหสัมพันธ์น้อยกับคอลัมน์ Chemical Oxygen Demand (COD)

8) ตัวแปรที่ผ่านการปรับปรุงชุดข้อมูลมี 18 ตัวแปร

## 6. การหาโมเดลโดยวิธี Hyperparameter Tuning

โมเดลที่ใช้ในการคาดการณ์ค่าความต้องการออกซิเจนทางเคมีเป็นโมเดลที่มีลักษณะเป็น Regression model สาเหตุมาจากตัวแปรต้นและตัวแปรตามมีลักษณะเป็นตัวเลขเพื่อหาว่าโมเดลใดสามารถนำมาใช้ในการคาดการณ์ค่าดังกล่าว ในโปรเจกต์นี้จะพิจารณาลักษณะโมเดลที่จะนำไปใช้ในการคาดการณ์ 6 โมเดลดังนี้

- 1.Linear Regression
- 2.Ridge Regression
- 3.Lasso Regression
- 4.Support Vector Regression
- 5.K-Nearest neighbor regression
- 6.Decision Tree Regression

การใช้ Hyperparameter tuning จะมีการกำหนดพารามิเตอร์ของโมเดล 6 ประเภทและใช้ชุดคำสั่งหาพารามิเตอร์ที่เหมาะสมสำหรับการคาดการณ์ค่าความต้องการออกซิเจนทางเคมีโดยมี Mean absolute error เป็นตัวกำหนดตามภาพที่ 16

ตารางพารามิเตอร์สำหรับโมเดลของ 6 โมเดลโดยใช้ข้อมูลคุณภาพน้ำของแม่น้ำน่าน

โมเดล	พารามิเตอร์
1. Linear Regression	1. fit_intercept: [True,False] 2. n_jobs: [None,1,5,10] 3. positive: [True,False]
2. Ridge Linear Regression	1. alpha: [0.01,0.1,1.0,10.0] 2. fit_intercept: [True,False] 3. random_state: [None,1,42]
3. Lasso Linear Regression	1. alpha: [0.01,0.1,1.0,10.0] 2. fit_intercept: [True,False] 3. random_state: [None,1,42]
4. Support Vector Regression	1. C: [0.1,1,10] 2. degree: [3,6,9]



	3. epsilon: [0.01,0.1,0.2] 4. gamma: ['scale','auto']
5. K-Neighbor Regressor	1. n_neighbors: [3,5,7] 2. weights: ['uniform','distance'] 3. algorithm: ['auto','ball_tree', 'kd_tree','brute'] 4. p: [1,2]
6. Decision Tree Regressor	1. max_depth: [None,4] 2. splitter: ['best','random'] 3. min_samples_split: [2,3,4] 4. min_samples_leaf: [2,3,4] 5. random_state: [None,1,42]

```

results2 = []

for model_info2 in models2:
    model2 = model_info2['model']
    param_grid2 = model_info2['param_grid']
    grid_search2 = GridSearchCV(model2, param_grid2,cv= 5,scoring='neg_mean_absolute_error')
    grid_search2.fit(x_train, y_train)

    best_model2 = grid_search2.best_estimator_
    y_pred2 = best_model2.predict(x_test)
    score_model2 = mean_absolute_error(y_test, y_pred2)

    results2.append({
        'Model': model_info2['name'],
        'Best Parameters': grid_search2.best_params_,
        'Mean Absolute Error': score_model2
    })

# Display the results
results2_df = pd.DataFrame(results2)
results2_df

```

ภาพที่ 16 ชุดคำสั่งสำหรับหาพารามิเตอร์ของโมเดล

ชุดคำสั่งจะหาพารามิเตอร์ที่มาจากพารามิเตอร์ที่กำหนดไว้และแสดงผลออกมาเป็นพารามิเตอร์ของโมเดลและค่า Mean absolute error ของโมเดลได้ผลลัพธ์ดังนี้



ตารางแสดงค่า Mean absolute error ของ 6 โมเดลโดยใช้ข้อมูลคุณภาพน้ำของแม่น้ำน่าน

โมเดล	พารามิเตอร์	Mean absolute error
Linear Regression	'fit_intercept': False 'n_jobs': None 'positive': False	0.7987
Ridge Linear Regression	'alpha': 10.0 'fit_intercept': False 'random_state': None	0.7788
Lasso Linear Regression	'alpha': 0.01 'fit_intercept': False 'random_state': None	0.7812
Support Vector Regression	'C': 10 'degree': 3 'epsilon': 0.01 'gamma': 'auto'	1.7267
K-Neighbors Regressor	'algorithm': 'auto' 'n_neighbors': 7 'p': 1 'weights': 'distance'	1.5473
Decision Tree Regressor	'max_depth': None, 'min_samples_leaf': 4 'min_samples_split': 2 'random_state': None 'splitter': 'random'	0.8459

จากตารางให้ผลลัพธ์ว่า Ridge Linear Regression ให้ค่า Mean absolute error น้อยที่สุดเมื่อเปรียบเทียบกับค่า Mean absolute error ของโมเดลอื่นนั้นหมายความว่า หากใช้โมเดลประเภท Ridge Linear Regression ในการคาดการณ์ค่า COD โมเดลสามารถคาดการณ์ค่าที่พยากรณ์เปรียบเทียบกับค่าจริงได้ค่าคลาดเคลื่อนของค่าที่พยากรณ์และค่าจริงน้อย

เพื่อเป็นการศึกษาว่าโมเดล 6 ประเภทที่ใช้ข้อมูลคุณภาพน้ำของแม่น้ำน่านักดงสามารถคาดการณ์ค่า COD โดยใช้ Mean absolute error และ Mean squared error ในโปรเจกต์นี้จะพิจารณาโมเดลโดยนำเอาโมเดลดังกล่าวไปทดสอบกับข้อมูลของแม่น้ำสายอื่น

## ผลการวิเคราะห์ข้อมูล

### 1.การประเมินโมเดลหลักโดยใช้ข้อมูลทดสอบของแม่น้ำสายเดียว

โมเดลที่ผ่านกระบวนการ Hyperparameter Tuning และใช้ข้อมูลฝึกหัดคุณภาพน้ำของแม่น้ำน้กตงไปทำการคาดการณ์ค่า COD ของชุดข้อมูลทดสอบของแม่น้ำน้กตง ชุดข้อมูลของแม่น้ำยองชั้น ชุดข้อมูลของแม่น้ำฮั่น และ ชุดข้อมูลของแม่น้ำกิมที่ได้ผลลัพธ์ในค่า Mean Absolute Error และ Mean Squared Error

```
ifdm1 = LinearRegression(fit_intercept = False, n_jobs = None, positive = False)
ifdm2 = Ridge(alpha = 10.0, fit_intercept = False, random_state = None)
ifdm3 = Lasso(alpha = 0.01, fit_intercept = False, random_state = None)
ifdm4 = SVR(C = 10, degree = 3, epsilon = 0.01, gamma = 'auto')
ifdm5 = KNeighborsRegressor(algorithm = 'auto', n_neighbors = 7, p = 1, weights = 'distance')
ifdm6 = DecisionTreeRegressor(max_depth = None, min_samples_leaf = 4,
                               min_samples_split = 4, random_state = None, splitter = 'random')
```

ภาพที่ 17 โมเดลหลัก

ตารางการประเมินโมเดลหลักโดยใช้ข้อมูลทดสอบคุณภาพของแม่น้ำน้กตง

โมเดล	Mean Absolute Error	Mean Squared Error
1. Linear Regression	0.7987	1.2175
2. Ridge Linear Regression	0.7788	1.1566
3. Lasso Linear Regression	0.7812	1.1606
4. Support Vector Regression	1.7267	5.7188
5. K-Neighbor Regressor	1.5473	4.7114
6. Decision Tree Regressor	0.8264	1.4836

จากตารางการประเมินโมเดลหลักโดยใช้ข้อมูลทดสอบคุณภาพของแม่น้ำน้กตงได้ข้อสรุปว่า Ridge Linear Regression ให้ค่า Mean Absolute Error และ Mean Squared Error น้อยที่สุด

ตารางการประเมินโมเดลหลักโดยใช้ข้อมูลคุณภาพของแม่น้ำยงชั้น

โมเดล	Mean Absolute Error	Mean Squared Error
1. Linear Regression	1.1237	6.6095
2. Ridge Linear Regression	1.0395	4.5149
3. Lasso Linear Regression	1.0515	4.8160
4. Support Vector Regression	2.5100	11.4438
5. K-Neighbor Regressor	2.5191	13.0675
6. Decision Tree Regressor	1.4637	5.0913

จากตารางการประเมินโมเดลหลักโดยใช้ข้อมูลคุณภาพของแม่น้ำยงชั้นได้ข้อสรุปว่า Ridge Linear Regression ให้ค่า Mean Absolute Error และ Mean Squared Error น้อยที่สุด

ตารางการประเมินโมเดลหลักโดยใช้ข้อมูลคุณภาพของแม่น้ำฮั่น

โมเดล	Mean Absolute Error	Mean Squared Error
1. Linear Regression	0.9347	1.8513
2. Ridge Linear Regression	0.9398	1.7709
3. Lasso Linear Regression	0.9388	1.7764
4. Support Vector Regression	2.5352	8.6404
5. K-Neighbor Regressor	2.0270	6.4447
6. Decision Tree Regressor	1.1488	2.4453

จากตารางการประเมินโมเดลหลักโดยใช้ข้อมูลคุณภาพของแม่น้ำฮั่นได้ข้อสรุปว่า Linear Regression ให้ค่า Mean Absolute Error และ Ridge Linear Regression ให้ค่า Mean Squared Error น้อยที่สุด

ตารางการประเมินโมเดลหลักโดยใช้ข้อมูลคุณภาพของแม่น้ำกิม

โมเดล	Mean Absolute Error	Mean Squared Error
1. Linear Regression	2.0113	9.9762

โมเดล	Mean Absolute Error	Mean Squared Error
2. Ridge Linear Regression	1.9444	9.8418
3. Lasso Linear Regression	1.9692	10.1276
4. Support Vector Regression	2.4173	11.1578
5. K-Neighbor Regressor	2.6398	13.3086
6. Decision Tree Regressor	1.8136	6.6792

จากตารางการประเมินโมเดลหลักโดยใช้ข้อมูลคุณภาพของแม่น้ำกิมได้ข้อสรุปว่า Decision Tree Regressor ให้ค่า Mean Absolute Error และ Ridge Linear Regression ให้ค่า Mean Squared Error น้อยที่สุด

ในตารางการประเมินโมเดลหลักสำหรับข้อมูลทดสอบของแม่น้ำนกกตง ข้อมูลของแม่น้ำยองชั้น ข้อมูลของแม่น้ำฮัน และ ข้อมูลของแม่น้ำกิมพบว่าโมเดลให้ความคลาดเคลื่อนของค่าคาดการณ์เทียบกับค่าที่เก็บข้อมูลไว้สูงมากเมื่อประเมินค่า Mean Squared Error

การทดสอบชุดข้อมูลคุณภาพน้ำของแม่น้ำกิมและแม่น้ำยองชั้น สามารถสรุปได้ว่าโมเดลดังกล่าว ค่าคาดการณ์ค่า COD คลาดเคลื่อนไปจากค่า COD ที่เก็บไว้สำหรับข้อมูลคุณภาพน้ำของแม่น้ำนกกตงเท่านั้น

## 2.การปรับปรุงโมเดล

เพื่อเป็นการปรับปรุงโมเดลให้สามารถคาดการณ์ค่า COD จึงทำการปรับปรุงโมเดลโดยนำเอาชุดข้อมูลของแม่น้ำสี่สายมาใช้แทนข้อมูลของแม่น้ำสายเดียว และมี 4 โมเดลที่ถูกใช้ในการพิจารณาเช่น Linear Regression, Ridge Linear Regression, Lasso Linear Regression และ Decision Tree Regressor

ส่วนโมเดล Support Vector Regression และ K-Neighbor Regressor ไม่ถูกนำมาพิจารณาเพราะสองโมเดลมีค่า Mean Squared Error สูงมากเมื่อเปรียบเทียบกับค่า Mean Squared Error ของสี่โมเดลก่อนหน้า

```
nifdm1 = LinearRegression(fit_intercept = False, n_jobs = None, positive = True)
nifdm2 = Ridge(alpha = 0.01, fit_intercept = False, random_state = None)
nifdm3 = Lasso(alpha = 0.1, fit_intercept = False, random_state = None)
nifdm4 = DecisionTreeRegressor(max_depth = None, min_samples_leaf = 4,
                               min_samples_split = 2, random_state = None, splitter = 'random')
```

ภาพที่ 18 โมเดลใหม่

ตารางการประเมินโมเดลหลักที่ใช้ข้อมูลของแม่น้ำ 1 สาย

โมเดล	โมเดลหลักที่ใช้ข้อมูลของแม่น้ำ 1 สาย	
	Mean Absolute Error	Mean Squared Error
1. Linear Regression	0.7987	1.2175
2. Ridge Linear Regression	0.7788	1.1566
3. Lasso Linear Regression	0.7812	1.1606
4. Decision Tree Regressor	0.8264	1.4836

จากตารางการประเมินโมเดลหลักที่ใช้ข้อมูลของแม่น้ำ 1 สายได้ข้อสรุปว่า Ridge Linear Regression ให้ค่า Mean Absolute Error และ Decision Tree Regressor ให้ค่า Mean Squared Error น้อยที่สุด

ตารางการประเมินโมเดลหลักที่ใช้ข้อมูลของแม่น้ำ 4 สาย

โมเดล	โมเดลหลักที่ใช้ข้อมูลของแม่น้ำ 4 สาย	
	Mean Absolute Error	Mean Squared Error
1. Linear Regression	0.9307	1.9189
2. Ridge Linear Regression	0.9290	1.8957
3. Lasso Linear Regression	0.9323	1.8846
4. Decision Tree Regressor	1.0807	2.5159

จากตารางการประเมินโมเดลหลักที่ใช้ข้อมูลของแม่น้ำ 4 สายได้ข้อสรุปว่า Ridge Linear Regression ให้ค่า Mean Absolute Error และ Lasso Linear Regression ให้ค่า Mean Squared Error น้อยที่สุด

ตารางการประเมินโมเดลใหม่ที่ใช้ข้อมูลของแม่น้ำ 4 สาย

โมเดล	โมเดลใหม่ที่ใช้ข้อมูลของแม่น้ำ 4 สาย	
	Mean Absolute Error	Mean Squared Error
1. Linear Regression	0.9392	1.9433
2. Ridge Linear Regression	0.9307	1.9187
3. Lasso Linear Regression	0.9393	1.9001
4. Decision Tree Regressor	1.0686	2.6133

จากตารางการประเมินโมเดลหลักที่ใช้ข้อมูลของแม่น้ำ 4 สายได้ข้อสรุปว่า Ridge Linear Regression ให้ค่า Mean Absolute Error และ ค่า Mean Squared Error น้อยที่สุด

### 3.การหาโมเดลของ GridsearchCV

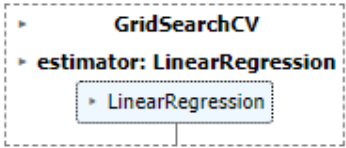
เป็นการหาโมเดลโดยใช้ GridsearchCV เพื่อเปรียบเทียบกับวิธีการหาโมเดลโดยวิธี Hyperparameter Tuning และ GridsearchCV โดยใช้ค่าความคลาดเคลื่อนของโมเดลเช่น Mean Absolute Error เป็นสิ่งที่ค้นหาโมเดล ในการประเมินโมเดลใช้ Mean Absolute Error และ Mean Squared Error เพื่อเปรียบเทียบกลุ่มโมเดลของสองวิธีการ

โมเดลที่หาได้จาก GridsearchCV มาจาก best estimator ที่แสดงผลว่าโมเดลหนึ่งที่ถูกเลือกและพารามิเตอร์ของโมเดลดังกล่าว

```

✓ [239] ifdx_train, ifdx_test, ifdy_train, ifdy_test = train_test_split(x_ifd, y_ifd, test_size = 0.2, random_state = 42)
✓ [240] nparam_grid = {'fit_intercept': [True,False], 'n_jobs': [None,1,5,10], 'positive': [True,False]}
✓ [241] grid_model1 = LinearRegression()
✓ [242] grid_search = GridSearchCV(grid_model1, nparam_grid, cv=5, scoring='neg_mean_absolute_error')
✓ [243] grid_search.fit(ifdx_train, ifdy_train)

```



ภาพที่ 19 การหาโมเดลของ GridsearchCV

```

[244] best_params = grid_search.best_params_
      best_estimator = grid_search.best_estimator_

[245] best_params
      {'fit_intercept': False, 'n_jobs': None, 'positive': True}

[246] best_estimator
      LinearRegression
      LinearRegression(fit_intercept=False, positive=True)

[288] gifdm1 = LinearRegression(fit_intercept = False, positive = True)

[289] gifdm1.fit(ifdx_train, ifdy_train)
      gifdm1_pred = gifdm1.predict(ifdx_test)
      print("Mean Absolute Error: ", mean_absolute_error(ifdy_test, gifdm1_pred))
      print("Mean Squared Error: ", mean_squared_error(ifdy_test, gifdm1_pred, squared = True))

      Mean Absolute Error: 0.9392536338509349
      Mean Squared Error: 1.9433995609003576

```

ภาพที่ 20 การหาโมเดลของ GridsearchCV



```
#LinearRegression(fit_intercept = False,positive = True)
#Ridge(alpha=0.01, fit_intercept=False)
#gifdm3 = Lasso(alpha=0.1, fit_intercept=False)
#gifdm4 = DecisionTreeRegressor(min_samples_leaf=4, splitter='random')
```

ภาพที่ 21 โมเดลของ GridsearchCV

ตารางการประเมินโมเดลของ GridsearchCV ที่ใช้ข้อมูลของแม่น้ำ 4 สาย

โมเดล	โมเดลของ GridsearchCV ที่ใช้ข้อมูลของแม่น้ำ 4 สาย	
	Mean Absolute Error	Mean Squared Error
1. Linear Regression	0.9392	1.9433
2. Ridge Linear Regression	0.9307	1.9187
3. Lasso Linear Regression	0.9393	1.9001
4. Decision Tree Regressor	1.0922	2.6041

จากตารางการประเมินโมเดลหลักที่ใช้ข้อมูลของแม่น้ำ 4 สายได้ข้อสรุปว่า Ridge Linear Regression ให้ค่า Mean Absolute Error และ ค่า Mean Squared Error น้อยที่สุด

## ข้อสรุปผลและการนำไปใช้

### 1. ข้อสรุปผล

1. ในการพิจารณาโมเดลของHyperparameter Tuning สำหรับการคาดการณ์ Chemical Oxygen Demand มีการประเมินโมเดลด้วย Mean absolute error, Mean squared error และ Coefficient of determination ทำให้ได้ข้อสรุปว่าโมเดลที่เหมาะสมสำหรับการคาดการณ์ควรใช้ชุดข้อมูลของแม่น้ำสาย และ มีโมเดลหลักที่สามารถนำไปใช้ในการคาดการณ์เช่น Lasso Linear Regression , Ridge Linear Regression และ Linear Regression

2. Mean squared error ของโมเดลหลักจากวิธีการHyperparameter Tuning มีค่าน้อยกว่า Mean squared error ของโมเดล GridsearchCV ดังนั้นการคาดการณ์ค่า COD ควรใช้โมเดลหลักจากวิธีการ Hyperparameter Tuning เช่น Lasso Linear Regression Ridge Linear Regression และ Linear Regression เพื่อให้ค่า COD ที่คาดการณ์แม่นยำมากที่สุด การเลือกโมเดลไปใช้ขึ้นกับค่าความคลาดเคลื่อนในการคาดการณ์ที่ยอมรับได้ของหน่วยงานภาครัฐและเอกชน

### 2. การนำไปใช้

โมเดลที่ใช้สำหรับการคาดการณ์ค่า COD สามารถนำไปใช้โดยมีตัวแปรต้น 17 ตัวแปรและตัวแปรตาม 1 ตัวแปรเป็นตัวแปรเกี่ยวกับคุณภาพน้ำที่เป็นประโยชน์ต่อภาครัฐในการศึกษาและตรวจสอบคุณภาพน้ำของแม่น้ำเพราะในประเทศไทยมีแม่น้ำหลายสายซึ่งเป็นส่วนสำคัญในด้านเศรษฐกิจเช่น การเกษตรกรรม การประมง และ การอุปโภคบริโภคในครัวเรือน

โมเดลที่มี 18 ตัวแปรทำให้ภาครัฐสามารถเก็บข้อมูลในตัวแปรดังกล่าวได้เพื่อลดเวลาในการเก็บข้อมูลตามจำนวนตัวแปรที่มากขึ้น และ ภาครัฐสามารถนำข้อมูลที่มีไปใช้ในการประเมินคุณภาพน้ำและคาดการณ์ค่า COD ของแม่น้ำแต่ละสายได้

โมเดลเป็นประโยชน์ต่อภาคเอกชนเช่น โรงงานอุตสาหกรรม โรงไฟฟ้า โรงแรม และ องค์กรทางธุรกิจ อื่นๆที่ต้องการนำน้ำไปใช้ในองค์กรเพื่อการบริหาร การผลิต การบำบัดน้ำเสียเพื่อนำน้ำกลับแหล่งน้ำธรรมชาติ

เพื่อให้การใช้น้ำของภาคเอกชนเป็นไปตามมาตรฐานที่กำหนดเพื่อเป็นการรักษาสิ่งแวดล้อมเช่น การปล่อยน้ำของโรงงานกลับสู่แหล่งน้ำธรรมชาติ

### 3. ข้อเสนอแนะ

1. การเก็บข้อมูลของแม่น้ำสายอื่นเพื่อให้โมเดลสามารถคาดการณ์ค่า COD แม่น้ำย้าขึ้น
2. การเก็บข้อมูลจากชุดข้อมูลของฤดูร้อนเป็นชุดข้อมูลใน 1 ปีเพราะแต่ละประเทศมีฤดูกาลที่แตกต่างกันเช่น ประเทศไทยมี 3 ฤดูกาล ประเทศเกาหลีใต้มี 4 ฤดูกาลทำให้โมเดลสามารถคาดการณ์ค่า COD เมื่อนำไปใช้กับข้อมูลของแม่น้ำสายอื่นและไม่มีข้อจำกัดการใช้โมเดลสำหรับในฤดูกาลที่มีการเก็บข้อมูล
3. การปรับปรุงโมเดลให้สามารถคาดการณ์ค่า COD ได้ในอนาคตที่อ้างอิงจากชุดข้อมูลที่มีอยู่ในปัจจุบัน
4. พิจารณาเรื่องการเก็บข้อมูล 18 ตัวแปรว่าการเก็บข้อมูลให้น้อยกว่า 18 ตัวแปรสามารถทำได้หรือไม่เพื่อลดเวลาในการเก็บข้อมูลเพราะแหล่งเก็บข้อมูลมีการเผยแพร่ชุดข้อมูลคุณภาพน้ำที่มีตัวแปรแตกต่างกันโดยต้องคำนึงถึงปัญหา Overfitting และ Underfitting เมื่อนำโมเดลไปใช้

## เอกสารอ้างอิง

เอกสารประกอบการเรียนการสอนวิชาการระบบสารสนเทศเพื่อการจัดการสำหรับวิศวกรเรื่อง Machine Learning ของผศ.ดร. พาทิศ วงศ์ชัยสุวัฒน์

<https://www.neonics.biz/water-quality/water-quality-measurement/>

<https://kosis.kr/eng/>

<https://scikit-learn.org/stable/index.html>