Busan science high school

2023 Ocean ICT Festival
2023 BOIF

C 13

QR 코드 영역
QR 삽입 후
테두리 삭제

Youtube 영상 QR

흡착제의 성질을 활용한 가장 효율적인 방류시간과 장소 제시

아지인 2502 김아진
2503 김지인

(1) 탐구동기

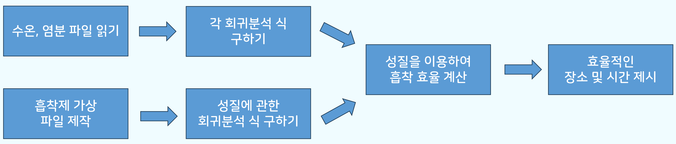
최근 해양오염의 심각성이 높아지고 있다. 해양오염을 일으키는 물질에는 미세플라스틱, 중금속 등 여러 종류가 있고, 이를 흡착하여 제거하는 것에 관한 연구가 활발히 진행중이다. 연구가 진행되었을 때에 온도, 염분에 따른 효율을 알려주는 프로그램을 제작하고자 하였다.

(2) 이론적배경

-중금속

중금속이란, 주기율표 상의 아래쪽에 주로 위치하고 있으며 비중이 4 이상인 무거운 원소를 말한다. 이는 하천, 대기 등 여러 방법을 통해 지속적으로 해양으로 유입되었다. 하지만 최근 해양쓰레기의 양이 많아지면서 퇴적물 내에 포함된 여러 중금속이 다시 녹아 나오는 용출 현상이 발생하며 해양 속 중금속의 양이 늘어나고 있다. 중금속은 체내에 흡수, 축적되게 되면 중금속 중독을 일으키며 폐암과 같은 여러가지 질병의 원인이 될 수 있다.

(3) 알고리즘 및 작동 원리



(4) 파이썬 코드 및 출력 화면

[가상의 흡착제 성질, 회귀분석 코드(요약)]

	A	B	C	D	E	F	G
1	salinity	temperature	absorption				
2	30	-20	5.2				
3	30	-19	5.3				
4	30	-18	5.3				
5	30	-17	5.6				
6	30	-16	5.8				
7	30	-15	6				
8	30	-14	6.3				
9	30	-13	6.5				
10	30	-12	6.7				
11	30	-11	6.9				

```
In [6]: filename='sal_temp_abs.csv'
        data=df.read_csv(filename)
        data.head()
```

Out[6]:

	salinity	temperature	absorption
0	30	-20	5.2
1	30	-19	5.3
2	30	-18	5.3
3	30	-17	5.6
4	30	-16	5.8

```
In [12]: data=np.loadtxt('sal_temp_abs.csv',delimiter=',',skiprows=1,encoding='UTF8')
         X=data[:,0:2]
         Y=data[:,2]

In [13]: from sklearn.linear_model import LinearRegression
         from sklearn.metrics import mean_squared_error
         from sklearn.model_selection import train_test_split

In [14]: x_train, x_test, y_train, y_test=train_test_split(X,Y,test_size=0.2,random_state=6)
```

```
In [30]: poly = PolynomialFeatures(6)
         poly = poly.fit_transform(X)
         x_train, x_test, y_train, y_test=train_test_split(poly,Y,test_size=0.2,random_state=5)
         print('변인 개수:',len(x_train[0]))

         lrmodel=LinearRegression()
         lrmodel.fit(x_train,y_train)

         predicted_y_train=lrmodel.predict(x_train)
         mse_train=mean_squared_error(y_train,predicted_y_train)
         print("MSE on train data: %f"%mse_train)

         predicted_y_test=lrmodel.predict(x_test)
         mse_test=mean_squared_error(y_test,predicted_y_test)
         print("MSE on train data: %f"%mse_test)


         r2_train = r2_score(y_train, predicted_y_train)
         r2_test = r2_score(y_test, predicted_y_test)
         print("train의 결정계수:%f"%r2_train)
         print("test의 결정계수:%f"%r2_test)
```

```
변인 개수: 28
MSE on train data: 0.051954
MSE on train data: 0.180972
train의 결정계수:0.990571
test의 결정계수:0.985669
```

```
In [1]: from sklearn.preprocessing import StandardScaler
        import matplotlib.pyplot as plt
        import numpy as np
        from sklearn.linear_model import LinearRegression
        from sklearn.metrics import mean_squared_error
        from sklearn.model_selection import train_test_split
        from sklearn.preprocessing import PolynomialFeatures

        data=np.loadtxt('sal_temp_abs.csv',delimiter=',',skiprows=1)

        X=data[:,0:2]
        Y=data[:,2]
        std_X=StandardScaler().fit_transform(X)
```

```
In [42]: poly = PolynomialFeatures(8)
         poly = poly.fit_transform(std_X)
         x_train, x_test, y_train, y_test=train_test_split(poly,Y,test_size=0.2,random_state=5)
         print('변인 개수:',len(x_train[0]))

         lrmodel=LinearRegression()
         lrmodel.fit(x_train,y_train)

         predicted_y_train=lrmodel.predict(x_train)
         mse_train=mean_squared_error(y_train,predicted_y_train)
         print("MSE on train data: %f"%mse_train)

         predicted_y_test=lrmodel.predict(x_test)
         mse_test=mean_squared_error(y_test,predicted_y_test)
         print("MSE on train data: %f"%mse_test)


         r2_train = r2_score(y_train, predicted_y_train)
         r2_test = r2_score(y_test, predicted_y_test)
         print("train의 결정계수:%f"%r2_train)
         print("test의 결정계수:%f"%r2_test)
```

```
변인 개수: 45
MSE on train data: 0.033104
MSE on train data: 0.065216
train의 결정계수:0.993992
test의 결정계수:0.990744
```

[염분 회귀분석 코드(요약)]

	A	B	C	D	E	F	G
1	salinity	temperatu	altitude				
2	30	-20	34.5				
3	30	-19	46.3				
4	30	-18	66.3				
5	30	-17	69.5				
6	30	-16	68.5				
7	30	-15	35.4				
8	30	-14	33.4				
9	30	-13	27.5				
10	30	-12	56.4				
11	30	-11	34.5				
12	30	-10	54.3				
13	30	-9	63.4				
14	30	-8	25.6				
15	30	-7	47.5				

```
In [9]: import matplotlib.pyplot as plt
        import numpy as np
        import pandas as df

In [17]: from sklearn.linear_model import LinearRegression
         filename='C:/Users/bio10/Downloads/temp_date_lat.csv'
         data=df.read_csv(filename)
         data.head()
```

Out[17]:

	YMD	LO	LA	SRFCLYR_WTPR	SRC
0	01-Dec-1970 00:00:00	0.5	73.5	1.28	COBE-SST2
1	01-Jan-1970 00:00:00	0.5	72.5	0.47	COBE-SST2
2	01-Feb-1970 00:00:00	0.5	72.5	0.51	COBE-SST2
3	01-Mar-1970 00:00:00	0.5	72.5	0.73	COBE-SST2
4	01-Apr-1970 00:00:00	0.5	72.5	1.18	COBE-SST2

```
In [41]: from sklearn.linear_model import LinearRegression
         filename='C:/Users/bio10/Desktop/salinity.csv'
         data=df.read_csv(filename)
         data.head()
```

Out[41]:

	salinity	temperature	altitude
0	30	-20	34.5
1	30	-19	46.3
2	30	-18	66.3
3	30	-17	69.5
4	30	-16	68.5

```
In [41]: from sklearn.linear_model import LinearRegression
         filename='C:/Users/bio10/Desktop/salinity.csv'
         data=df.read_csv(filename)
         data.head()
```

Out[41]:

	salinity	temperature	altitude
0	30	-20	34.5
1	30	-19	46.3
2	30	-18	66.3
3	30	-17	69.5
4	30	-16	68.5

```
In [47]: data=np.loadtxt('C:/Users/bio10/Desktop/salinity.csv',delimiter=',',skiprows=1,encoding='UTF8')
         X=data[:,0:2]
         Y=data[:,2]

In [53]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

Out[53]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)

```
In [54]: from sklearn.linear_model import LinearRegression
         from sklearn.metrics import mean_squared_error
         from sklearn.model_selection import train_test_split

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

Out[54]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)

```
In [52]: import pandas as pd
         import numpy as np
         from sklearn.linear_model import LinearRegression

         df = pd.DataFrame({'x': [1, 2, 3], 'y': [1,2,3]})

         X = df['x'].values.reshape(-1, 1)
         y = df['y'].values.reshape(-1, 1)
         model = LinearRegression().fit(X, y)

         print('Coefficients:', model.coef_)
         print('Intercept:', model.intercept_)
```

```
Coefficients: [[1.]]
Intercept: [6.66133815e-16]
```

[수온 회귀분석 코드(요약)]

```
In [4]: import numpy as np
        import matplotlib.pyplot as plt
        from sklearn.linear_model import LinearRegression
        import pandas as df
        filename='temp_date_lat.csv'
        data=df.read_csv(filename)
        data.head()
```

Out[4]:

	YMD	LO	LA	SRFCLYR_WTPR	SRC
0	01-Dec-1970 00:00:00	0.5	73.5	1.28	COBE-SST2
1	01-Jan-1970 00:00:00	0.5	72.5	0.47	COBE-SST2
2	01-Feb-1970 00:00:00	0.5	72.5	0.51	COBE-SST2
3	01-Mar-1970 00:00:00	0.5	72.5	0.73	COBE-SST2
4	01-Apr-1970 00:00:00	0.5	72.5	1.18	COBE-SST2

[흡착제 성질을 이용한 최적 조건 출력]

```
In [56]: import pandas as pd
         import numpy as np
         from sklearn.linear_model import LinearRegression

         # 시간대와 흡착제 제거 효율 데이터 생성 (임의의 예시 데이터)
         time_periods = np.array([1, 2, 3, 4, 5])  # 시간대
         removal_efficiency = np.array([85, 92, 88, 95, 90])  # 제거 효율 (%)


         df = pd.DataFrame({'Time': time_periods, 'Efficiency': removal_efficiency})


         X = df['Time'].values.reshape(-1, 1)
         y = df['Efficiency'].values.reshape(-1, 1)
         model = LinearRegression().fit(X, y)


         print('Coefficients:', model.coef_)
         print('Intercept:', model.intercept_)


         best_time = model.predict([[0]])
         print('Best Time for Maximum Removal Efficiency:', best_time[0][0], 'hours')
```

```
Coefficients: [[1.3]]
Intercept: [86.]
Best Time for Maximum Removal Efficiency: 86.1 hours
```

(5) 기대효과

- 환경의 조건 (기온, 위치, 시간) 등에 따른 수온과 염분을 예측할 수 있을 것이다.
- 해양 속 중금속을 제거할 수 있는 흡착제가 개발되었을 시, 예측한 수온과 염분에 따라 어떤 시간대에 흡착제를 방류해야 가장 효과적이게 작용할 수 있는지 알아낼 수 있다.
- 중금속을 효과적으로 제거함으로서 해양오염과 이로 인해 인간에게 미치는 악영향을 최대한 줄일 수 있다.