1. Feature Selection

```
# 1、Importing algorithm libraries

import pandas as pd

from xgboost import XGBRegressor

from sklearn.ensemble import RandomForestRegressor

from sklearn.model_selection import train_test_split

from sklearn.metrics import mean_squared_error

from sklearn.metrics import mean_absolute_error

from sklearn.metrics import r2_score

from math import sqrt

import matplotlib.pyplot as plt

import numpy as np

data1 = pd.read_excel('CoRE MOF-7feature.xlsx')

data1

# 2、Correlation diagram of features and adsorption properties mapping

import seaborn as sns

plt.figure(figsize=(8,6))

data1 = pd.DataFrame(data1)

sns.heatmap(data1.corr()

                    ,annot=True

                ,linewidths=0.2

                    ,center=0

                #,xticklabels=['$Vol$', r'$ρ$', '$GCD$', '$PLD$', '$LCD$',    '$Sa$',
r'$\phi$','${N_{isobutene}}$','$APS$', '$R$']

                #,yticklabels=['$Vol$', r'$ρ$', '$GCD$', '$PLD$', '$LCD$',    '$Sa$',
r'$\phi$','${N_{isobutene}}$','$APS$', '$R$']

                ,xticklabels=['Vol', r'$ρ$', 'GCD', 'PLD', 'LCD',    'Sa',
r'$\phi$','${N_{isobutene}}$','APS', 'R']

                ,yticklabels=['Vol', r'$ρ$', 'GCD', 'PLD', 'LCD',    'Sa',
```

```
r'$\phi$','${N_{isobutene}}$','APS', 'R']
                ,cbar_kws={'label': 'correlation coefficient'})
plt.savefig("corr-heat-7f.jpg",dpi=300)
plt.show()
```

2. Model Selection (Ridge regression as an example)

(1)

```
from sklearn.model_selection import train_test_split
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as cm
#machine learning
from sklearn.linear_model import Ridge, LinearRegression, Lasso,ElasticNet
from sklearn.linear_model import LassoCV
from sklearn.linear_model import RidgeCV
from sklearn.linear_model import ElasticNetCV
from sklearn.linear_model import BayesianRidge
from sklearn import svm
from sklearn.ensemble import RandomForestClassifier
from sklearn.neural_network import MLPRegressor
from xgboost import XGBRegressor
# Criteria
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import r2_score
```

(2)

```
data = pd.read_excel(' CoRE MOF.xlsx'
                        ,index_col=0 # Data indexing automatically ignored
                          )
```

```
# data =data.astype(int)
data = pd.DataFrame(data)
(3)
x = data.iloc[:,0:5]
x = np.matrix(x)
y1 = data.iloc[:,-1].values.reshape(-1,1)
den = data.iloc[:,-6].values.reshape(-1,1)

x1train, x1test, y1train, y1test = train_test_split(x,y1,test_size=0.2, random_state=0)
x2train, x2den, y2train, y2den = train_test_split(x,den,test_size=0.2, random_state=0)
print(x1train.shape,x1test.shape,y1train.shape,y1test.shape)
(4)
data.describe()
(5)
from math import sqrt
alpharange = np.logspace(-3,3,400,base=10)
Ridge1_ = RidgeCV(alphas=alpharange,cv=5).fit(x1train,y1train)
Ridge1_ = Ridge(alpha = Ridge1_.alpha_).fit(x1train,y1train)
y1_predict_rr = Ridge1_.predict(x1test)

mse_predict_N_ri = mean_squared_error(y1test, y1_predict_rr)
mae_predict_N_ri = mean_absolute_error(y1test, y1_predict_rr)
R2_N_ri = r2_score(y1test,y1_predict_rr)
rmse_predict_N_ri = sqrt(mse_predict_N_ri)
print('Uptakes，Ridge：
',mse_predict_N_ri,mae_predict_N_ri,rmse_predict_N_ri,R2_N_ri)
(6)
font = {'family' : 'serif',
        'color'    : 'black',
```

```
        'weight' : 'normal',
        'size'   : 11,
        }
plt.scatter(y1test,y1_predict_rr, c=y2den, cmap='rainbow',alpha = 0.5)
plt.xlabel('CBMC Simulated isobutene adsorption capacity(mmol/g)',fontdict = font)
plt.ylabel('ML predicted isobutene adsorption capacity(mmol/g)',fontdict = font)
plt.title('Ridge regression',fontdict = font)
plt.tick_params(labelsize=11)
plt.plot([0,y1_predict_rr.max()],[0,y1_predict_rr.max()],linewidth = '1.5',color = 'r')
plt.savefig("N-Ridge regression.png", bbox_inches='tight',dpi=2000)
plt.show()
```

3. Model Building and Separation Property Prediction for G-MOFs (Taking isobutene loading as a label for example)

```
# 1、Importing algorithm libraries
import pandas as pd
from xgboost import XGBRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import r2_score
from math import sqrt
import matplotlib.pyplot as plt
import numpy as np
#2、Importing and viewing data
data = pd.read_excel('804-CoREMOF.xlsx')
data
# 3、Dividing the above data into labels and features
```

```
y = data.iloc[:,6].values
x = data.iloc[:,1:6]
print(y)
print(x)
# 4、Modeling with XGBoost model
x = np.matrix(x)
x1train, x1test, y1train, y1test = train_test_split(x,y,test_size=0.2,
random_state=1000)
xgbr = XGBRegressor(
        eta=0.1,
        gamma=0.2,
        max_depth=10,
        n_estimators=100,
        min_child_weight=0.5,
        colsample_bytree=0.9,
        subsample=0.8,

        #tree_method='gpu_hist',
        seed=42)
# 5、Scoring and outputting the training and test sets
y1_predict_train = xgbr.fit(x1train,y1train).predict(x1train) # Scoring the training set
mse_predict_train = mean_squared_error(y1train, y1_predict_train) # Calculating
MSE
mae_predict_train = mean_absolute_error(y1train, y1_predict_train) # Calculating
MAE
rmse_predict_train = sqrt(mse_predict_train) # Calculating RMSE
R2_train = r2_score(y1train,y1_predict_train) # Calculating R2
print('Train set Score: MSE({:.6f}), MAE({:.6f}), RMSE({:.6f}),
R2({:.6f})'.format(mse_predict_train,mae_predict_train,rmse_predict_train,R2_train))
```

```
# Output the above scores
y1_predict = xgbr.fit(x1train,y1train).predict(x1test) # Scoring the test set
mse_predict = mean_squared_error(y1test, y1_predict)
mae_predict = mean_absolute_error(y1test, y1_predict)
R2 = r2_score(y1test,y1_predict)
rmse_predict = sqrt(mse_predict)
print('Test set Score: MSE({:.6f}), MAE({:.6f}), RMSE({:.6f}),
R2({:.6f})'.format(mse_predict,mae_predict,rmse_predict,R2))
# 6、Draw and save a diagram
plt.scatter(y1train.tolist(),y1_predict_train.tolist(),alpha=0.4,edgecolors = 'b',label =
'Train set')
plt.scatter(y1test.tolist(),y1_predict.tolist(),alpha=0.4,edgecolors = 'r',label = 'Test set')
plt.plot([0,25],[0,25],linewidth = '1.0',color = 'r')
plt.xlabel('CBMC simulated isobutene loading (mmol/g)')
plt.ylabel('XGBoost predicted isobutene loading (mmol/g)')
plt.title('XGBoost regression')
plt.legend()
plt.savefig('isobutene-804-5f.jpg',dpi=2000)
plt.ylim(ymin = 0, ymax=25)
plt.xlim(xmin = 0, xmax=25)
plt.show()
# 7、Feature importances
xgbr.feature_importances_
Weight = pd.DataFrame(xgbr.feature_importances_)
Weight.to_csv('importance to isobutene loading.csv',index=False)
Weight
# 8、Importing data of G-MOFs
dataset1 = pd.read_csv('G-MOF-oms-fina.csv')
dataset1
```

```
# 9
x3 = dataset1.iloc[:,1:6]
x3
# 10
x3 = np.matrix(x3)
y_predictG = xgbr.fit(x1train,y1train).predict(x3)
dfG = pd.DataFrame(y_predictG)
dfG.columns = ['XGBoost predicted isobutene loading (mmol/g)']
df2 = pd.concat([pd.DataFrame(dataset1),dfG],axis=1)
df2.to_excel('G-MOF-predict-isobutene.xlsx',index=False)
df2
```

4. Data Mining Key Genes (Take CoRE MOFs database as an example)

```
# 1、Importing algorithm libraries
import pandas as pd
from xgboost import XGBRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import r2_score
from math import sqrt
import matplotlib.pyplot as plt
import numpy as np
#2、Importing and viewing data
data = pd.read_csv('CoREMOFs-genes-N-S-Cw-R-APS.csv')
data
# 3、Dividing the above data into labels and features
y = data.iloc[:,440].values
```

```
x = data.iloc[:,1:435]
print(y)
print(x)
# 4、Modeling with XGBoost model
x = np.matrix(x)
x1train, x1test, y1train, y1test = train_test_split(x,y,test_size=0.2,
random_state=1000)
xgbr = XGBRegressor(
        eta=0.1,
        gamma=0.2,
        max_depth=10,
        n_estimators=100,
        min_child_weight=0.5,
        colsample_bytree=0.9,
        subsample=0.8,

        #tree_method='gpu_hist',
        seed=42)
# 5、Scoring and outputting the training and test sets
y1_predict_train = xgbr.fit(x1train,y1train).predict(x1train) # Scoring the training set
mse_predict_train = mean_squared_error(y1train, y1_predict_train) # Calculating
MSE
mae_predict_train = mean_absolute_error(y1train, y1_predict_train) # Calculating
MAE
rmse_predict_train = sqrt(mse_predict_train) # Calculating RMSE
R2_train = r2_score(y1train,y1_predict_train) # Calculating R2
print('Train set Score: MSE({:.6f}), MAE({:.6f}), RMSE({:.6f}),
R2({:.6f})'.format(mse_predict_train,mae_predict_train,rmse_predict_train,R2_train))
# Output the above scores
```

```
y1_predict = xgbr.fit(x1train,y1train).predict(x1test) # Scoring the test set
mse_predict = mean_squared_error(y1test, y1_predict)
mae_predict = mean_absolute_error(y1test, y1_predict)
R2 = r2_score(y1test,y1_predict)
rmse_predict = sqrt(mse_predict)
print('Test set Score: MSE({:.6f}), MAE({:.6f}), RMSE({:.6f}),
R2({:.6f})'.format(mse_predict,mae_predict,rmse_predict,R2))
# 6、Draw and save a diagram
plt.scatter(y1train.tolist(),y1_predict_train.tolist(),alpha=0.4,edgecolors = 'b',label =
'Train set')
plt.scatter(y1test.tolist(),y1_predict.tolist(),alpha=0.4,edgecolors = 'r',label = 'Test set')
plt.plot([0,45],[0,45],linewidth = '1.0',color = 'r')
plt.xlabel('CBMC simulated APS')
plt.ylabel('XGBoost predicted APS')
plt.title('XGBoost regression')
plt.legend()
plt.savefig('CoREMOFs-APS',dpi=2000)
plt.ylim(ymin = 10, ymax=122)
plt.xlim(xmin = 10, xmax=122)
plt.show()
# 7、
xgbr.feature_importances_
Weight = pd.DataFrame(xgbr.feature_importances_)
Weight.to_csv('CoRE MOFs-genes- importance-APS.csv',index=False)
Weight
```