**III.인공지능 기반 물질 개발 및 거동 분석**

1. 수성 가스 전이 반응 촉매 분석

최근 수소가 환경 친화적인 차세대 에너지 전달체로 부각되어 수소 생산을 위한 효율적인 기술이 요구되는 실정이다. 수성 가스 전이 반응 (Water gas shift reaction, WGSR)은 고순도 수소를 생산하는 기술로, 미래 청정에너지 생산을 위한 기술로 각광을 받고있다. WGSR은 합성가스 내의 일산화탄소와 수증기가 반응하여 수소와 이산화탄소를 생성하는 반응으로, 아래와 같은 반응식을 따른다.

고효율의 WGSR 공정 설계에는 고성능 WGSR 촉매 개발이 중요하다. 촉매 성능은 활성 (전환율, 선택도) 및 안정성 (기계적 내구성, 열적 안정성)으로 대표되고, 현재 촉매 성능 개선을 위한 촉매 물질의 특성에 대한 연구가 활발히 진행되고 있다. 하지만, 촉매 성능은 촉매 재료와 운전 조건에 따른 복잡한 현상으로 실험단계에서 예측하는 것은 상당히 어렵다.

그러므로, 고성능 촉매 설계를 위해서 촉매 성능을 예측할 정확하고 정밀한 방법론이 요구되는 실정이다. 만약, 촉매의 성능을 예측할 수 있다면 촉매 개발과 공정 적용의 긴 R&D 생애주기를 효과적으로 줄일 수 있다. 본 실습에서 제안한 인공신경망은 고차원 데이터를 처리할 수 있는 알고리즘으로, 복잡계 해석을 위한 유용한 방법론이다.

본 실습에서는 인공신경망 기반 고성능 WGSR 촉매 예측을 목적으로한다.

**예제 1: 인공 신경망을 이용한 수성 가스 전이 반응 촉매 예측**

예제는 R 4. 0. 2 프로그래밍 언어를 기준으로 Rstudio 개발환경에서 작성되었다. 예제 실습을 위해 Anaconda 프로그램을 아래 URL로부터 다운로드할 수 있다.

https://www.anaconda.com/

* **데이터 링크**

(WGSR 데이터 링크 삽입)



* ‘read.csv’ 함수를 사용하여 데이터 파일이 저장된 장소를 직접 찾아 ‘WSGR’ 이름으로 데이터를 불러온다. 해당 데이터는 column 이름이 이미 존재하는 데이터로 ‘header = T’ arg를 통해 이를 밝힌다.



* ‘view()’ 함수를 사용하여 데이터를 확인할 수 있다.



* ‘WGSR’ 데이터의 손상을 막기위해 ‘WGSR\_re’ 데이터를 생성하여 사용한다.

**실습: 인공신경망 기반 수성 가스 전이 반응 촉매 예측**



* ‘scale()’ 함수를 사용하여 ‘WGSR\_re’ 데이터를 전처리한다.



* ‘replace()’ 함수를 사용하여 ‘WGSR\_scale’ 데이터의 공백값 (‘NA’)를 제거하도록 한다. 이때, ‘is.na’ arg를 활용해 해당 데이터의 공백값만을 인식시킬 수 있다.



* ‘floor()’ 함수를 사용하여 ‘WGSR\_scale’ 데이터의 행 개수의 70%에 해당되는 값을 생성한다. 이 값은 ‘smp\_size’로 명명하여, 후에 무작위 데이터 추출에 사용한다.
* ‘sample()’ 함수를 사용해서 ‘WGSR\_scale’ 데이터로부터 행 번호를 무작위 추출을 진행한다. ‘size’ arg를 ‘smp\_size’로 설정하여 총 70% 데이터를 추출한다. ‘seq\_len()’ 함수를 통해 1부터 임의의 지정된 숫자까지 순차 데이터를 생성할 수 있다.



* 임의로 추출된 행 번호 (‘train\_ind’)를 사용해서 ‘WGSR\_scale’ 데이터의 행들을 추출한다. 70%가 학습에 사용할 데이터로 ‘neural\_train’ 으로 명명한다. ‘train\_ind’ 행 번호를 제외한 데이터는 검증에 사용될 데이터이므로 ‘neural\_test’로 명명한다.



* 인공신경망 학습을 위해 변수 이름을 지정하는 과정이 필요하다.
* ‘WGSR\_scale’ 데이터의 열 이름을 추출하여 ‘name\_col’로 명명한다. 해당 ‘name\_col’의 38번까지가 독립변수이므로 해당 부분을 추출하여 ‘name\_col’을 재 정의한다.



* ‘CO.conversion’은 데이터의 종속변수에 해당한다. ‘paste()’함수를 사용해서 ‘name\_col’의 변수들을 ‘+’ 기호로 연결된 일련의 문장으로 만든다. 그리고 ‘CO.conversion~’과 결합한다.
* 해당 문자열을 ‘as.formula’ 함수를 통해 수식으로 인식시키고, ‘vari\_name’으로 명명한다.



* ‘set.seed()’를 ‘7’로 지정하여 계산의 무작위성을 고정한다.
* ‘neuralnet()’ 함수를 사용해서 인공신경망 모델을 구축한다. Arg로 ‘vari\_name’을 사용해 데이터의 독립 변수 및 종속 변수 관계를 지정하고, ‘neural\_train’ 데이터를 사용한다는 것을 명시한다. ‘hidden’ arg를 통해 인공신경망의 구조를 설정한다. 구조는 다섯개의 은닉층과 각 층에 해당하는 node인 34-10-8-8-1로 구성된다.
* 해당 인공신경망을 ‘WGSR\_scale\_neural’로 명명한다.



* ‘$result.matrix’를 사용하면 ‘WGSR\_scale\_neural’의 weight, bias, 그리고 학습 결과를 확인할 수 있다.

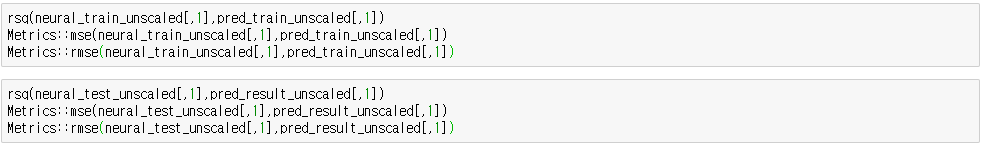


* ‘compute’ 함수를 통해 ‘WGSR\_scale\_neural’모델을 사용할 수 있다.
* ‘neural\_test’ 데이터의 독립변수 부분을 지정하여 예측해보도록 한다.
* 마찬가지로 ‘neural\_train’도 예측하여 학습 정확도를 확인해보도록 한다.





* 예측된 데이터는 scale된 값이므로 unscale할 필요가 있다. 따라서, 수식을 역전시켜 예측된 학습 데이터와 검증 데이터의 원본 값을 추출하도록 한다.



* 정확도 파악을 위해 각 unscale된 데이터의 R2, MSE, RMSE 값을 확인한다.

1. 이온성 액체의 무한 희석 활성도 계수 추정

이온성 액체 (Ionic liquids, ILs)는 구조를 사용목적에 따라 선택적으로 합성 및 사용할 수 있는 designer material로, 흔히 탄화수소 추출 공정에 이용한다. 이온성 액체는 독특한 물리/화학적 특성 (전기화학적 안정성, 높은 이온전도성 등)을 갖아, 기존의 유기 용매 기반의 추출 공정을 대체할 친환경적인 대안으로 각광을 받고있다.

이온성 액체의 거동 원리는 보통 묽은 용액에서의 활동도 계수 (Infinite diluent activity coefficient, IDAC)에 기반해 이해됐다. 이를 계산하기 위한 방법으로 UNIFAC, COSMO-RS, 혹은 Abraham과 같은 mechanistic model들이 사용됐다. 하지만, 활동도 계수 계산을 위한 mechanistic model들은 낮은 정확도와 한정된 데이터를 기반으로 작동한다는 단점이 명확하다. 또한, 수많은 용매와 용질 사이의 활동도 계수 실험은 상당한 시간과 예산을 필요로 하는 작업으로, 현재의 이온성 액체 개발과 적용은 비효율적인 상태에 머물러있다.

인공신경망 (Artificial neural networks)은 데이터 기반 학습을 통한 복잡계 현상의 예측 모델을 생성하는 인공지능 알고리즘이다. 인공신경망은 고차원의 데이터 학습에 높은 성능을 보여주고 있고, 오늘날 다양한 물질 개발 및 공정 개선 연구에 활용되고 있다.

본 실습에서는 이온성 액체의 물리/화학적 특성 데이터 베이스를 활용하여, 인공신경망을 통한 활동도 계수 예측모델을 개발한다.

**예제 1: 인공 신경망을 이용한 ILs 활동도 예측**

예제는 Python 3.5 프로그래밍 언어를 기준으로 Jupyter 개발환경에서 작성되었다. 예제 실습을 위해 Anaconda 프로그램을 아래 URL로부터 다운로드할 수 있다.

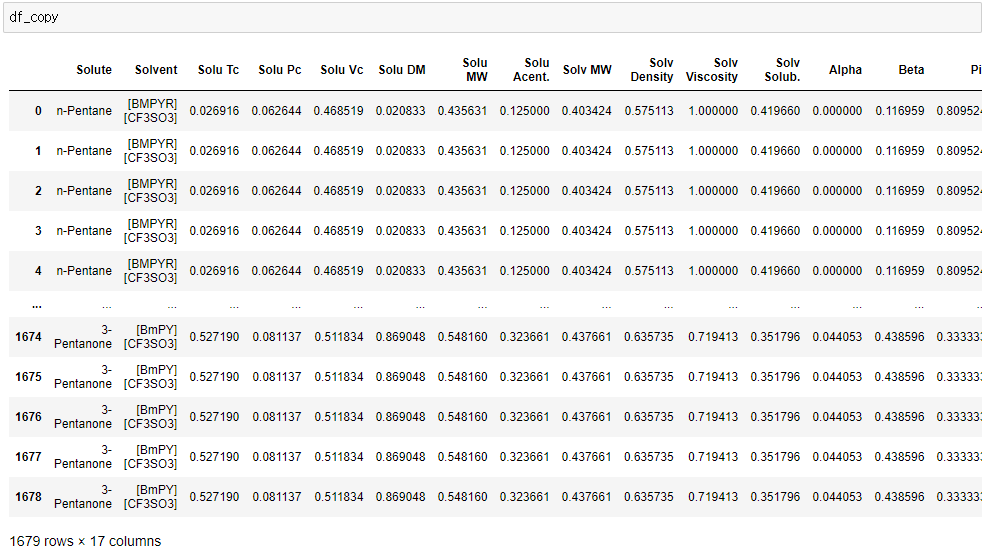
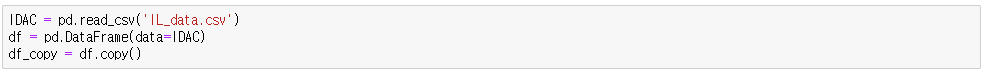
https://www.anaconda.com/

* **데이터 링크**

(ILs 데이터 링크 삽입)



* ‘pandas’ package는 데이터 편집을 위한 도구로, python 언어에서 매우 유용하게 사용된다. OSN 데이터 불러오기 및 데이터 편집 (컬럼 및 이름 설정)을 위해 본 실습에서 사용한다.
* ‘numpy’ package는 수학적 기능이 탑재된 도구로, 난수 발생과 대수적 계산을 위해 사용된다. 본 실습에서는 데이터의 형태 변환 및 벡터 계산에 사용한다.
* ‘matplotlib.pyplot’ package는 시각화를 위한 그래프 도구로, 다양한 그래프를 그리기 위해 사용된다.
* ‘pandas’ package를 import하고 ‘pd’로 축약해 사용한다.
* ‘numpy’ package를 import하고 ‘np’로 축약해 사용한다.
* ‘matplotlib.pyplot’ package를 import하고 ‘plt’로 축약해 사용한다.



* ‘pd’를 사용하여 Jupyter 환경에 있는 ‘IL\_data.csv’ 데이터를 Jupyter script 환경에 불러온다..
* ‘IL\_data.csv’는 ‘pd’를 활용, 데이터 프레임 형태인 df로 정의한다.
* ‘.copy’ 함수를 사용하여 ‘df’ 데이터의 복사본인 ‘df\_copy’를 만든다. 이 과정은 원본 데이터의 회손을 피하기위해 실행된다..

**실습: 인공신경망 기반 ILs 활동도 계수 예측**



* ‘random’ module은 무작위 숫자 생성 등 무작위 관련 함수를 제공한다. 본 실습에서 훈련 및 검증 데이터셋 생성을 위해 사용한다.
* ‘random’ module을 import하고 ‘rn’으로 축약해 사용한다.



* ‘sklearn’ package는 계산을 위한 유용한 함수를 다수 내장하고 있다. ‘sklearn.preprocessing’의 ‘MinMaxScaler’ 함수를 import한다.



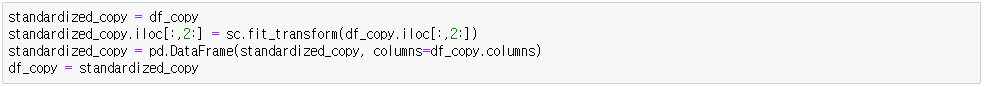
* ‘sklearn.model\_selection’의 ‘train\_test\_split’ 함수를 import하도록 한다. ‘train\_test\_split’ 함수는 모델 학습 및 검증을 위한 모델 학습 데이터와 검증 데이터를 분할하기 위해 사용한다.
* ‘sklearn.metrics’의 ‘mean\_square\_error’ 함수와 ‘r2\_score’ 함수를 import하도록 한다. ‘mean\_square\_error’와 ‘r2\_score’는 모델 검증을 위한 정확도 지표를 내장하고 있는 함수이다.



* ‘keras’는 ‘TensorFlow’와 같은 머신러닝 백앤드 엔진 플랫폼을 지원하는 라이브러리 이자 API (Application programming interface)이다.
* ‘keras’로부터 ‘Sequential’, ‘optimizer’, ‘metrics’ 함수들을 import한다.
* ‘tensorflow.keras’ 함수로부터 ‘backend’ import하고 ‘K’로 축약한다.
* ‘keras.layers’로부터 ‘Dense’, ‘BatchNormalization’, ‘Dropout’, ‘Activation’ 함수들을 import한다.
* ‘keras.optimizers’로부터 ‘Adam’ 함수를 import한다.
* ‘keras.initializers’로부터 ‘lecun\_normal’ 함수를 import한다.



* ‘MinMaxScaler()’ 함수를 ‘sc’로 명명하여 사용.



* ‘sc.fit\_transform’ 함수를 사용하여 ‘df\_copy’ 데이터의 numerical value만 (3열부터 17열)을 정규화한다.
* 정규화한 데이터는 ‘standardized\_copy’로 명명하고 데이터 프레임 형태로 변환한다.
* ‘standardized\_copy’ 데이터를 ‘df\_copy’ 로 명명한다.



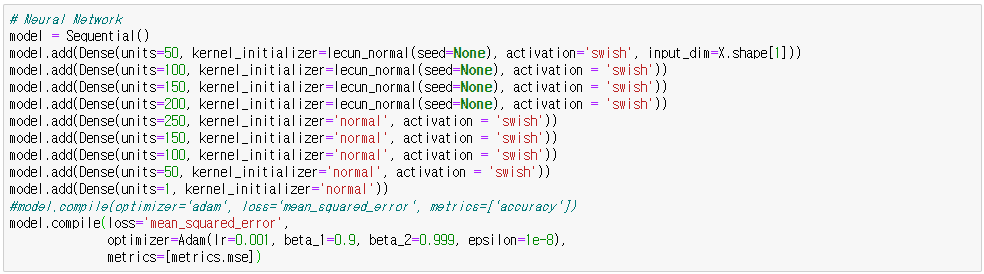
* ‘df\_copy’ 데이터의 독립변수 부분인 2열부터 16열까지를 추출하여 ‘X’로 명명한다.
* ‘df\_copy’ 데이터의 종속변수 부분인 ‘in IDAC’을 추출하여 ‘Y’로 명명한다.



* ‘tf’의 ‘.random.set\_seed’ 함수를 사용하여 데이터 추출의 무작위성을 고정시킨다.
* 마찬가지로 인공신경망 학습의 무작위성을 고정시키기위해 ‘randomstate’를 생성하고 차후 학습에 사용한다.



* ‘train\_test\_split’ 함수를 사용하여 학습데이터와 검증데이터를 생성한다.
* ‘X\_train’과 ‘Y\_train’은 각각 학습데이터의 독립변수와 종속변수 부분이고 마찬가지로 ‘X\_test’와 ‘Y\_test’는 각각 검증데이터의 독립변수와 종속변수 부분이다.
* 본 실습에서는 학습에 전체 데이터의 70%를 할당할 것이므로, ‘test\_size’ arg를 ‘.3’으로 설정한다. 모델 학습의 재현성을 위해 데이터 추출의 무작위성을 고정한다 (‘random\_state’ arg).



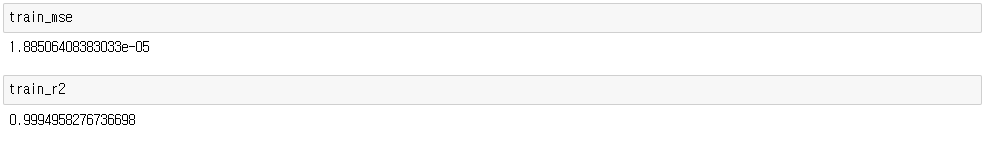
* ‘sequential()’ 함수를 사용하여 인공신경망 모델을 생성한다. 모델은 ‘model’로 명명하고 인공신경망 층은 ‘.add’를 통해 추가할 수 있다.
* 일반적인 신경망 모델을 사용하기 위해 ‘Dense’ 기능을 넣고, 각 노드를 지정한다.
* Activation argument는 인공신경망의 활성화 함수에 해당하며, Input shape argument는 데이터의 column 차원이다.
* 본 실습에선 ‘swish’를 활성화 함수로 사용한다.
* ‘.compile’ 함수를 통해 학습의 optimizer와 loss function을 설정할 수 있다.
* 본 실습에서 사용된 Optimizer는 adam optimizer이고 loss function으로 MSE (Mean square error)를 사용한다.



* ‘.fit’ 함수를 사용하여 모델 학습을 진행한다.
* 모델 학습에 ‘X\_train’와 ‘y\_train’ 데이터를 사용한다. ‘batch\_size’ arg를 통해 한번에 10개 데이터셋을 고려하도록 한다.
* 또한, ‘epochs’ arg를 통해 반복 학습의 주기를 1000으로 지정한다.
* ‘verbose’ arg는 모델 학습의 진행상황을 표현하는 것으로, 0으로 설정하여 진행상황을 표시하지 않는 것으로 지정한다.



* 학습이 끝나면, ‘.predict’ 함수를 사용해서 ‘X\_train’ 데이터를 예측한다. 예측된 결과값은 ‘y\_train\_pred’로 명명한다.
* ‘mean\_squared\_error’와 ‘r2\_score’함수를 사용해서 ‘y\_train’과 ‘y\_train\_pred’의 정확도를 확인한다.



* 학습의 정확도를 확인한다.



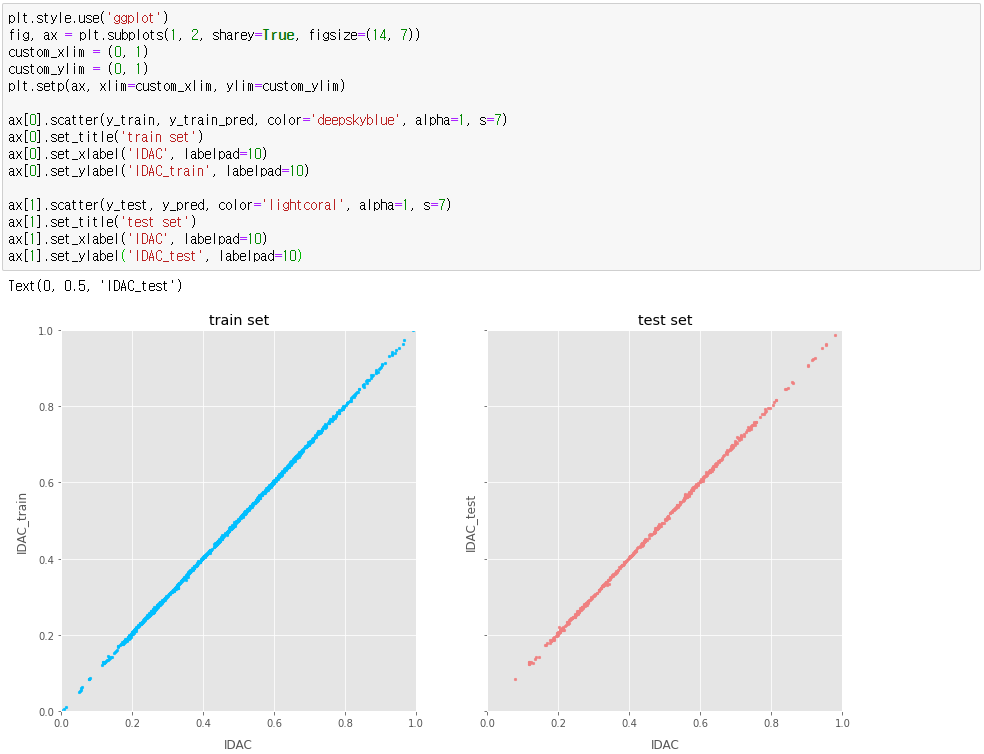
* 마찬가지로 ‘mean\_squared\_error’와 ‘r2\_score’함수를 사용해서 ‘y\_test’과 ‘y\_test\_pred’의 정확도를 확인한다.



* 학습의 정확도를 확인한다.



* 학습의 정확도를 확인한다.



* ‘plt’를 사용하여 정확도를 시각화한다.