4장 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.

**5.2 [방법] 인공신경망 예제**

**Q1.** 인공신경망을 활용하기 위한 라이브러리를 조사하라.

**A1.** 인공신경망 학습을 위한 다양한 라이브러리가 존재한다.

|  |
| --- |
| Library(‘neuralnet’) |

|  |
| --- |
| Library(‘H2O’) |

|  |
| --- |
| Library(‘keras’) |

**Q2.** 인공신경망 학습에는 다양한 하이퍼파라미터가 선행적으로 결정되야한다. 이들 중 모델 정확도와 직접적으로 연관있는 하이퍼파라미터 중 활성화 함수에대해 설명하라.

**A2.** Activation function은 인공신경망의 node를 활성화시키는 활성화 함수로, input된 가중치 합을 출력 신호로 변환하는 함수이다. 인공 신경망에서 이전 레이어에 대한 가중 합의 크기에 따라 활성 여부가 결정된다. 목적 및 역할에따라 선택적으로 사용된다.

**5.3 [응용] 인공 신경망 기반 촉매 반응 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/

**Q1**. 데이터를 Jupyter환경으로부터 불러오고 데이터를 확인하라.

**A1.** 다음과 같은 code를 사용하여 불러올 수 있다.

|  |
| --- |
| WGSR <- read.csv(file.choose(), header = T) |

* ‘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(WGSR) |

* ‘view()’ 함수를 사용하여 데이터를 확인할 수 있다.

|  |
| --- |
| WGSR\_re <- WGSR |

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

**Q2.** 인공지능 학습을 위해서 데이터의 단위를 무시한 상대적 영향력을 파악할 필요가 있다. 촉매 반응 데이터를 표준화하라. 또한, 데이터의 결측치를 제거하라.

**A2.** 다음과 같은 절차를 통해 표준화 및 결측치 제거가 가능하다.

|  |
| --- |
| WGSR\_scale <- scale(WGSR\_re) |

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

|  |
| --- |
| WGSR\_scale <- data.frame(replace(WGSR\_scale, is.na(WGSR\_scale), 0) |

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

**Q3.** 인공신경망 기반 예측 모델의 정확도를 검증하기 위해 데이터를 모델 학습 데이터와 모델 검증 데이터로 분할하라.

**A3.** 다음과 같은 절차를 통해 데이터의 분할이 가능하다.

|  |
| --- |
| smp\_size <- floor(0.7 \* nrow(WGSR\_scale))  train\_ind <- sample(seq\_len(nrow(WGSR\_scale)), size = smp\_size) |

* ‘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.

|  |
| --- |
| neural\_train <- WGSR\_scale[train\_ind,]  neural\_test <- WGSR\_scale[-train\_ind,] |

* 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.

|  |
| --- |
| name\_col <- c(colnames(WGSR\_scale))  name\_col <- name\_col[1:38] |

* 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.

**Q4.** 인공 신경망 학습을 시도하고 정확도를 검증하라.

**A4.** 다음과 같은 절차를 통해 인공 신경망 학습을 시도할 수 있다.

|  |
| --- |
| vari\_name <- as.formula(paste(‘CO.Conversion ~’,  paste(name\_col,  collapse = ‘+’))) |

* ‘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(7)  WGSR\_scale\_neural <- neuralnet(vari\_name, neural\_train,  hidden = c(34, 10, 8, 8, 1)) |

* ‘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.

|  |
| --- |
| WGSR\_scale\_neural$result.matrix |

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

|  |
| --- |
| pred\_result <- data.frame(compute(WGSR\_scale\_neural, neural\_test[,1:38]))  pred\_result\_train <- data.frame(compute(  WGSR\_scale\_neural, neural\_train[,1:38]) |

* ‘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.

|  |
| --- |
| Pred\_train\_unscaled <- data.frame(mean(WGSR$CO.Conversion) +  pred\_result\_train$net.result \* sd(WGSR$CO.Conversion))  neural\_train\_unscaled <- data.frame(mean(WGSR$CO.Conversion) +  neural\_train$net.result \* sd(WGSR$CO.Conversion)) |
| Pred\_result\_unscaled <- data.frame(mean(WGSR$CO.Conversion) +  pred\_result$net.result \* sd(WGSR$CO.Conversion))  neural\_test\_unscaled <- data.frame(mean(WGSR$CO.Conversion) +  neural\_test$net.result \* sd(WGSR$CO.Conversion)) |

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

|  |
| --- |
| rsq(neural\_train\_unscaled[,1], pred\_train\_unscaled[,1])  Metrics::mse(neural\_train\_unscaled[,1], pred\_train\_unscaled[,1])  Metrics::rmse(neural\_train\_unscaled[,1], pred\_train\_unscaled[,1]) |
| rsq(neural\_test\_unscaled[,1], pred\_ test\_unscaled[,1])  Metrics::mse(neural\_ test\_unscaled[,1], pred\_ test\_unscaled[,1])  Metrics::rmse(neural\_ test\_unscaled[,1], pred\_ test\_unscaled[,1]) |

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

**학습 결과**

* 학습 내용

고성능 수성화 전이 반응 촉매 개발을 위한 인공 신경망 기반 예측 방법론 익히기.

* 학습 결과 확인하기

인공 신경망 알고리즘의 활용 방법 및 예측 모델 학습을 위한 데이터 구조화 익히기.

* 학습 결과 응용하기

본 장의 학습내용에 기반해 촉매 성능의 예측과 예측 알고리즘의 정확도 향상시키기.