6. Python과 R 연동

6.1 Python, R 연동 개요

Python, R 연동 노드는 사용자가 작성한 Python 또는 R 스크립트를 실행할 수 있는 기능을 제공한다. 이를 통해 ECMiner™에서 제공하지 않는 연산이나 분석 라이브러리를 자유롭게 수행할 수 있다. 실행방법은 로컬 경로에서 Python, R 실행 환경과 연동하여 입력 데이터에 대해 스크립트를 실행하고 결과를 ECMiner™로 불러오는 방식이다.

■ 연동 노드 입출력 형식

Python, R 연동 노드는 ECMiner™의 다른 노드들과 마찬가지로 입력과 출력이 모두 데이터 표 형태로 연결되는 표준 입출력 구조를 가진다. 이러한 구조를 위해 사용자가 작성한 코드 앞뒤에 자동으로 삽입되는 내부 스크립트가 존재한다. 이 스크립트를 통해 입력 데이터는 ECMiner™의 입력노드에서 자동으로 csv 파일로 생성되고, “ecmData”라는 이름의 데이터프레임 형태로 Python 또는 R 환경에 전달된다. 결과 데이터 또한 데이터프레임 형태로 ECMiner™에 반환되어 표 형태의 출력 노드 데이터로 변환된다.

■ 상단 툴바

Python, R 연동 노드의 스크립트를 작성할 때 사용되는 기본 기능이다.

툴바	기능	설명
	실행 취소	마지막 작업 실행 취소
	다시 실행	이전에 실행 취소된 작업 다시 실행
	줄 번호 표시	코드 줄 번호 표시 또는 숨김
	들여쓰기 표시	코드 들여쓰기 수준을 가이드라인 형태로 표시
	특수 문자 표시	공백, 탭, 줄바꿈 등의 특수 문자를 화면에 표시
	자동 줄 바꿈	코드가 편집기 창 너비를 넘어간 경우, 자동으로 줄을 바꿔 표시
	코드 축소	코드 블록(함수, 클래스 등)을 접거나 펼침
	찾기	스크립트 내의 특정 단어나 문장 검색
	다음 찾기	검색된 단어의 다음 위치로 이동
	이전 찾기	검색된 단어의 이전 위치로 이동
	열/줄로 이동	특정 행(줄 번호)이나 열로 커서 이동
	책갈피 설정	현재 줄의 책갈피 설정 또는 해제

6.2 Python 연동

6.2.1 Python 기초

Python 연동 노드를 실행하기 위해서는 로컬 경로에 Python 실행 환경이 설치되어 있어야 한다. 로컬에 Python이 설치되어 있다면 ECMiner™은 해당 환경을 인식하여 스크립트를 실행할 수 있다.

■ 가상환경

가상환경(Virtual Environment)은 로컬 환경 내에 가상으로 만들어진 독립된 Python 실행 환경이다. 프로젝트별로 가상환경을 구성하면, 필요한 라이브러리와 Python 버전을 개별적으로 관리할 수 있다. 이를 통해 패키지 간 버전 충돌을 방지하고 각 프로젝트에 필요한 라이브러리를 자유롭게 설치·운영할 수 있도록 한다. 따라서 프로젝트를 독립적으로 관리하기 위해 가상환경을 생성하는 것을 권장하며, Python 연동 노드에서도 해당 가상환경을 지정하여 실행할 수 있다.

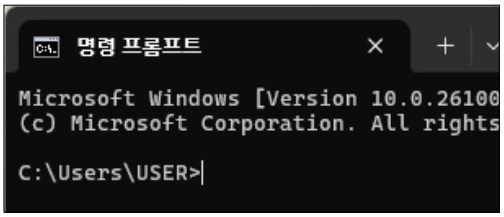
■ 가상환경 생성 방법

Python 3에는 기본적으로 venv 모듈이 포함되어 있어 이를 이용하여 손쉽게 가상환경을 만들 수 있다. 또한, 데이터 분석 환경에서는 conda 명령어를 사용하기도 한다. ECMiner™는 WindowOS 기반으로 구동되므로, 아래 절차에 따라 간편하게 가상환경을 만들 수 있다.

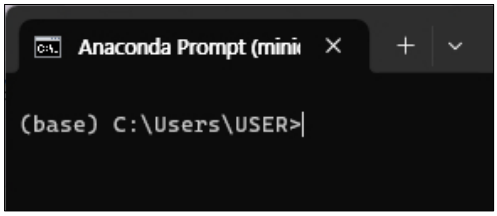
(1) 명령 창 실행

시작메뉴에서 명령 프롬프트 또는 Anaconda Prompt를 실행한다.

<명령 프롬프트>



<Anaconda Prompt>



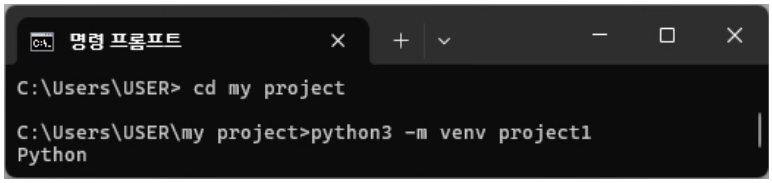
(2) 가상환경 생성

작업하고 있는 프로젝트 파일 디렉토리에 가상환경을 생성한다. 아래 명령어를 실행하면 설정한 가상환경 이름으로 디렉토리가 생성되고, 그 안에 가상환경이 구성된다.

- venv

```
cd ~/{프로젝트 디렉토리}
python3 -m venv {가상환경이름}
```

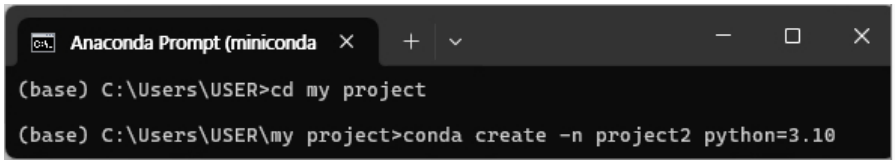
(예시)



- conda

```
cd ~/{프로젝트 디렉토리}
conda create -n {가상환경이름} python={파이썬버전}
```

(예시)



(3) 가상환경 활성화

가상환경을 사용하려면 아래 명령어를 통해 가상환경을 활성화한 후 작업을 시작해야 한다. 이때, 명령어 실행 위치는 앞 단계에서 생성된 가상환경 디렉토리 내부여야 한다.

- venv

```
source {가상환경이름}/Scripts/activate
```

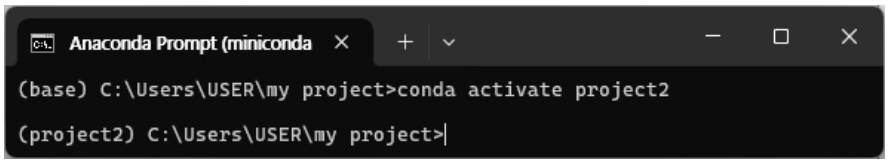
- conda (운영체제 공통)

```
conda activate {가상환경이름}
```

명령어 실행 후 프롬프트에 가상환경 이름이 표시되면 활성화가 완료된 것이다.

```
({가상환경이름}) C:\Users>
```

(예시)



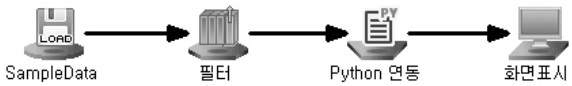
(4) 가상환경 비활성화

가상환경을 종료하려면 아래 명령어를 입력한다. MacOS와 WindowsOS 모두 동일하다.

```
deactivate
```

6.2.2 Python 연동 노드

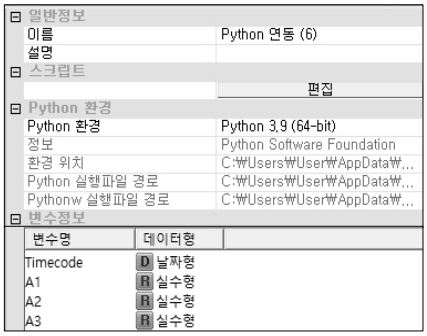
Python 연동 노드에 입력 노드를 연결하여 데이터를 읽어올 수 있다. 아래 스트림과 같이 파일 입력 노드와 전처리 노드들을 연결하여 Python 연동 노드와 함께 작업이 가능하다.



Python 연동 노드를 실행하기 위해서는 노드 속성창에서 Python 환경을 선택한다. 로컬에 Python 패키지와 가상환경이 설치되어 있으면 ECMiner™에서 자동으로 인식되며,
작업을 실행하고자 하는 가상환경을 선택한다. 속성창의 편집버튼을 눌러 파이썬 코드를 작성하여 확인을 누르면 변수 정보에 변수들이 표시되며, Python 스크립트 실행 후 최종적으로 ecmData DataFrame 내에 존재하는 변수들이다.

■ Python 연동 노드

- 옵션 선택



① ‘Python 환경’ 선택
스크립트를 실행하고자 하는 Python 환경 선택
환경 선택 시 Python 실행파일 경로 등이 자동으로 입력됨

② 스크립트 ‘편집’ 적용
Python 코드를 작성

- 스크립트 편집기

Python 연동 노드를 더블클릭하거나 노드 속성창의 편집 버튼을 누르면 아래와 같이 스크립트 편집기 창이 나타난다.

실행하고자 하는 Python 스크립트를 편집기에 작성한 후, 오른쪽 하단의 확인 버튼을 누르면 적용된다. 왼쪽 하단의 가져오기 버튼을 누르면, 사전에 작성되어 있는 Python script file(*.py)을 가져올 수 있다.

Python 연동 노드의 입출력 구조를 유지하기 위해 삽입된 전후 코드는 아래와 같다.
(1) 입력 데이터 불러오기

```
# -*- coding: cp949 -*-
import datetime
import numpy as np
import pandas as pd
pd.set_option('display.max_columns', None)
# 입력노드로부터 생성된 임시 csv 파일로부터 ecmData DataFrame 생성.
ecmData = pd.read_csv("%temporary_input_csv_file%", encoding = 'CP949')
```

(2) 출력 데이터 내보내기

```
ecmData.to_csv("%temporary_output_csv_file%", index = False, encoding =
'CP949')
```

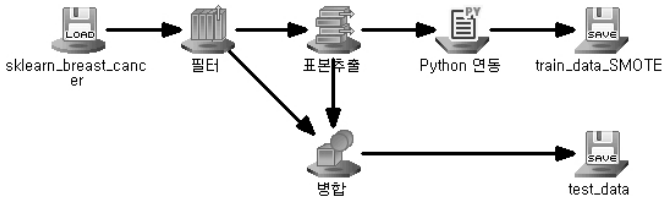
위의 자동 코드는 ECMiner™에서 Python 데이터 입출력 형식으로 자동 처리된다.

6.2.3 Python 연동 예제

Python 연동 노드를 활용하여 데이터 불균형 문제를 보정하고자 한다. 예제에서는 sklearn 라이브러리에 포함된 유방암(breast cancer) 데이터를 사용하며, 데이터 중 일부 주요 변수만 선택하여 학습용(train)과 검증용(test) 데이터를 생성한다.

SMOTE(Synthetic Minority Over-sampling Technique)는 불균형 데이터에서 소수 클래스(minority class) 범주 데이터를 인공적으로 생성하여 클래스 간 균형을 맞추는 오버샘플링 기법이다. Python에서 비대칭 데이터 라벨 문제를 해결하기 위해 사용되는 방법이다. 해당 라이브러리를 사용하여 학습 데이터의 대표성을 높이고, 모델이 한쪽 클래스에 치우쳐 학습되는 문제를 완화한 데이터를 생성한다.

본 예제에서는 모델링에 사용할 학습용과 검증용 데이터를 분할하기 위하여 아래와 같은 스트림을 실행한다.



■ 데이터 (sklearn_breast_cancer.csv)

breast cancer 데이터는 종양의 세포 특성을 기반으로 악성(Malignant)과 양성(Benign)로 분류된 데이터로, 각 샘플은 여러 측정 지표(예: 반경, 질감, 면적, 대칭성 등)를 포함한다.

mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	target
17,99000	10,38000	122,80000	1,001,00000	0,11840	0,27760	0,30010	0,14710	0,24190	0
20,57000	17,77000	132,90000	1,326,00000	0,08474	0,07864	0,08690	0,07017	0,18120	0
19,69000	21,25000	130,00000	1,203,00000	0,10960	0,15990	0,19740	0,12790	0,20690	0

■ 데이터 전처리

- 변수 선택

필터 노드를 이용해 전체 데이터에서 아래 다섯 개의 변수들만 선택한다.

사용 컬럼	설명
mean radius	세포 반경의 평균값
mean texture	세포 표면 질감의 평균값
mean area	세포 면적의 평균값
mean symmetry	세포 대칭성의 평균값
target	종양의 악성(0) 또는 양성(1) 여부

- 데이터 분할

표본추출 노드를 이용해 임의추출 방법으로 전체 데이터의 80%를 학습용 데이터를 선택한다. 그 후, 병합 노드의 Anti-Join 병합 방법을 통해 학습용 데이터 포함되지 않은 데이터로 테스트 데이터를 추출한다.

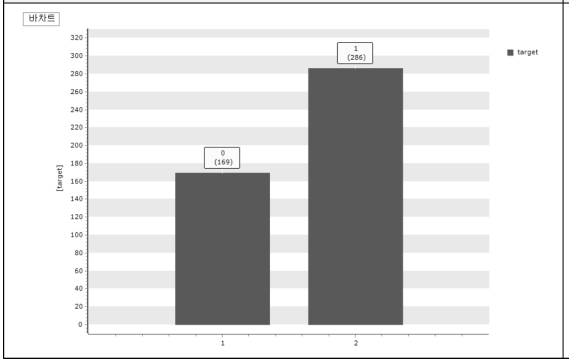
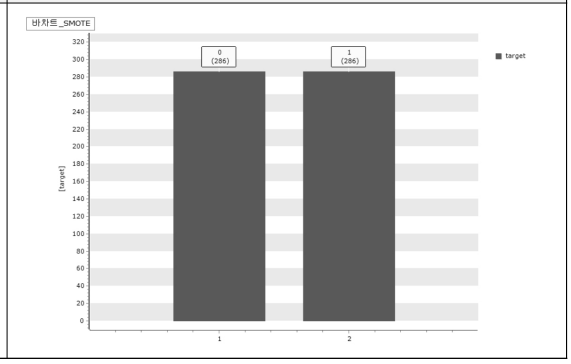
■ 학습용 데이터 오버샘플링

Python 연동 노드에서 SMOTE 기법을 적용하여 train 데이터의 클래스 불균형을 보정한다. 스크립트 편집기를 열어 아래와 같이 코드를 작성한 뒤 확인 버튼을 눌러 수행한다.

```
# 모듈 미설치 시 !pip install imbalanced-learn 명령어로 설치
from imblearn.over_sampling import SMOTE

# ECMiner™의 입력 노드에서 전달된 ecmData DataFrame 사용
X = ecmData.drop('target', axis=1)
y = ecmData['target']

# SMOTE 적용
sm = SMOTE(random_state=42)
X_resampled, y_resampled = sm.fit_resample(X, y)

# 결과를 DataFrame 형식으로 다시 재구성
ecmData = pd.concat([pd.DataFrame(X_resampled, columns=X.columns),
                     pd.DataFrame(y_resampled, columns=['target'])], axis=1)
```

■ 실행 결과

SMOTE 기법을 적용하기 전과 후의 클래스별 데이터 개수 변화를 바차트로 시각화하여 보여준다. 오른쪽 그래프는 SMOTE를 적용하여 두 클래스의 데이터 개수가 동일하게 조정된 균형 잡힌 학습용 데이터이 생성된 것을 확인할 수 있다.

SMOTE 기법 적용 전	SMOTE 기법 적용 후
	

■ 결과 저장

Python 연동 노드의 출력 결과(균형 보정된 train 데이터)는 파일 출력 노드로 연결하여 csv 파일 형태로 저장한다. 이때 검증용 데이터도 함께 저장하여, 이후 모델링 단계에서 활용할 수 있다.

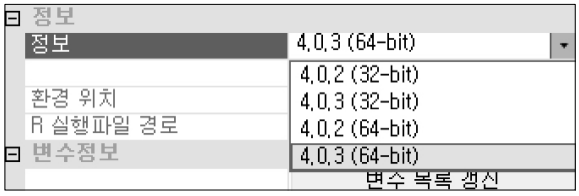
6.3 R 연동

6.3.1 R 기초

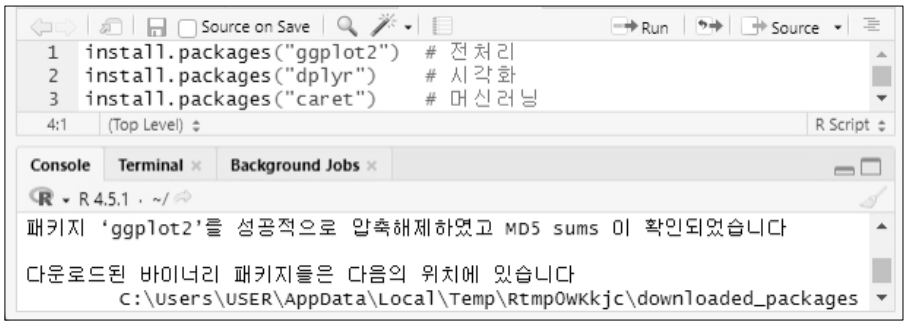
R 연동은 로컬에 설치된 R 실행 환경과 연동하여 ECMiner™ 내부 데이터에 대한 R 스크립트를 수행할 수 있다. R 연동은 ECMiner™에서 제공하지 않는 분석 모듈, 라이브러리의 활용이 가능하게 한다.

■ R 환경 구성

R 프로그램이 로컬에 설치되어 있어야 하며, ECMiner™은 시스템 내 R 환경을 자동으로 인식한다.



R 연동시 필요한 추가 패키지가 있다면, R 환경에 별도로 설치해야 한다. 다음은 R studio에서 'ggplot2', 'dplyr', 'caret' 패키지를 설치하는 예시이다.



6.3.2 R 연동 노드

R 연동 노드에 입력 노드를 연결하여 데이터를 불러들일수 있다. 아래와 같이 파일 입력 노드와 전처리 노드들을 연결하여 R 연동 노드와 함께 작업이 가능하다.

R 연동을 위해서는 노드 속성창에서 R 환경을 선택한다. 로컬에 R 패키지와 가상환경이 설치되어 있으면 ECMiner™에서 자동으로 인식되며, 실행하고자 하는 R 버전을 선택한다.속성창의 편집버튼을 눌러 파이썬 코드를 작성하여 확인을 누르면 변수 정보에 변수들이 표시되며, R 스크립트 실행 후 최종적으로 ecmData DataFrame 내에 존재하는 변수들이다.

■ R 연동 노드

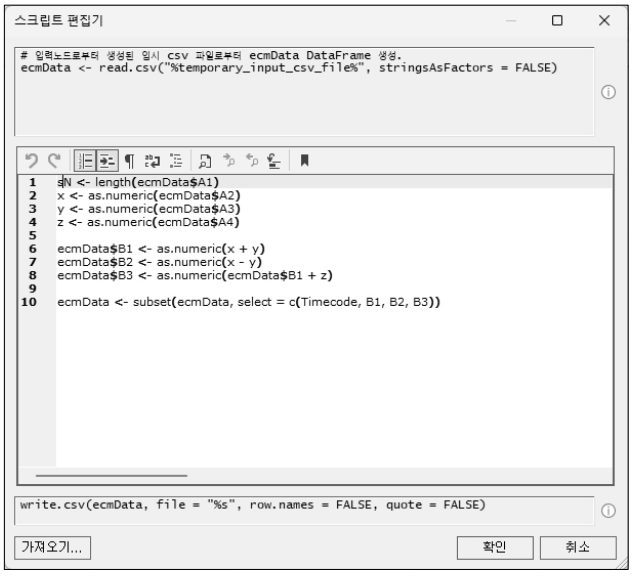
- 옵션 선택

① ‘정보’ 선택
스크립트를 실행하고자 하는 R 환경을 선택

② 스크립트 ‘편집’ 적용
R 코드를 작성

- 스크립트 편집기

R 연동 노드를 더블클릭하거나 노드 속성창의 편집 버튼을 누르면 아래와 같이 스크립트 편집기 창이 나타난다.



실행하고자 하는 R 스크립트를 편집기에 작성한 후, 오른쪽 하단의 확인 버튼을 누르면 적용된다. 왼쪽 하단의 가져오기 버튼을 누르면, 사전에 작성되어있는 R script file(*R)을 가져올 수 있다.

R 연동 노드의 입출력 구조를 유지하기 위해 삽입된 전후 코드는 아래와 같다.

(1) 입력 데이터 불러오기

```
# 입력노드로부터 생성된 임시 csv 파일로부터 ecmData DataFrame 생성.
ecmData <- read.csv("%temporary_input_csv_file%", stringsAsFactors = FALSE)
```

(2) 출력 데이터 내보내기

```
write.csv(ecmData, file = "%s", row.names = FALSE, quote = FALSE)
```

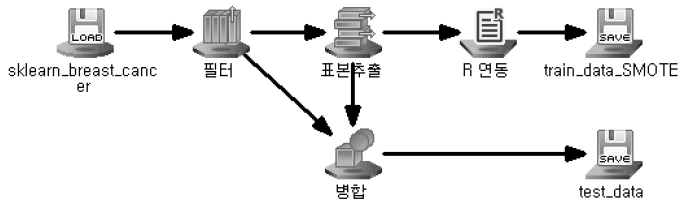
위의 자동 코드들은 ECMiner™가 R과의 데이터 입출력 형식을 자동으로 처리한다.

6.3.3 R 연동 예제

R 연동 노드를 활용하여 데이터 불균형 문제를 보정하고자 한다. 예제에서는 Python 연동노드에 동일하게 사용한 유방암(breast cancer) 데이터를 사용하며, 데이터 중 일부 주요 변수만 선택하여 학습용(train)과 검증용(test) 데이터를 생성한다.

R의 smotefamily 라이브러리를 통해 SMOTE 기법을 적용할 수 있다. 해당 라이브러리를 사용하여 학습 데이터의 대표성을 높이고, 모델이 한쪽 클래스에 치우쳐 학습되는 문제를 완화한 데이터를 생성한다.

본 예제에서는 모델링에 사용할 학습용과 검증용 데이터를 분할하기 위하여 아래와 같은 스트림을 실행한다.



■ 데이터 (sklearn_breast_cancer.csv)

breast cancer 데이터는 종양의 세포 특성을 기반으로 악성(Malignant)과 양성(Benign)로 분류된 데이터로, 각 샘플은 여러 측정 지표(예: 반경, 질감, 면적, 대칭성 등)를 포함한다.

mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	target
17,99000	10,38000	122,80000	1,001,00000	0,11840	0,27760	0,30010	0,14710	0,24190	0
20,57000	17,77000	132,90000	1,326,00000	0,08474	0,07864	0,08690	0,07017	0,18120	0
19,69000	21,25000	130,00000	1,203,00000	0,10960	0,15990	0,19740	0,12790	0,20690	0

■ 데이터 전처리

- 변수 선택

필터 노드를 이용해 전체 데이터에서 아래 다섯 개의 변수들만 선택한다.

사용 컬럼	설명
mean radius	세포 반경의 평균값
mean texture	세포 표면 질감의 평균값
mean area	세포 면적의 평균값
mean symmetry	세포 대칭성의 평균값
target	종양의 악성(0) 또는 양성(1) 여부

- 데이터 분할

표본추출 노드를 이용해 임의추출 방법으로 전체 데이터의 70%를 학습용 데이터를 선택한다. 그 후, 병합 노드의 Anti-Join 병합 방법을 통해 학습용 데이터 포함되지 않은 데이터로 검증 데이터를 추출한다.

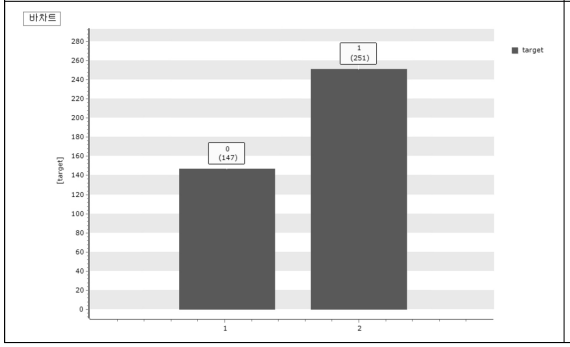
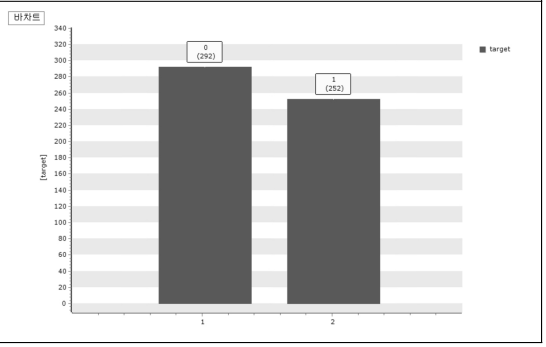
■ 학습용 데이터 오버샘플링

R 연동 노드에서 SMOTE 기법을 적용하여 train 데이터의 클래스 불균형을 보정한다. 스크립트 편집기를 열어 아래와 같이 코드를 작성한 뒤 확인 버튼을 눌러 수행한다.

```
# 패키지 설치/로딩 - 모듈 미설치 시 install.packages("smotefamily")로 설치
library(smotefamily)

# 1. 훈련데이터 준비 : 종속/독립변수 분리
X <- ecmData[, setdiff(names(ecmData), "target")]
target <- as.factor(ecmData$target)
X_num <- model.matrix(~ . - 1, data = X) # dummy 인코딩

# 2. SMOTE 적용
set.seed(123)
smote_out <- smotefamily::SMOTE(
  X = data.frame(X_num),
  target = target,
  K = 5,            # 소수 클래스의 이웃 수
  dup_size = 1     # 오버샘플링 강도(=100%)
)

# 3. 결과를 ecmData로 변환
out <- smote_out$data
out$target <- out$class
out$class <- NULL
ecmData <- out
```

■ 실행 결과

SMOTE 기법을 적용하기 전과 후의 클래스별 데이터 개수 변화를 바차트로 시각화하여 보여준다. 오른쪽 그래프는 SMOTE를 적용하여 두 클래스의 데이터 개수가 유사한 비율로 조정되어 보다 균형 잡힌 학습용 데이터이 생성된 것을 확인할 수 있다.

SMOTE 기법 적용 전	SMOTE 기법 적용 후
	

■ 결과 저장

R 연동 노드의 출력 결과(균형 보정된 train 데이터)는 파일 출력 노드로 연결하여 csv 파일 형태로 저장한다. 이때 검증용 데이터도 함께 저장하여, 이후 모델링 단계에서 활용할 수 있다.