Guided Project 과정

[SK] SKADA 인증 Practitioner 시험 대비 문제 풀이

프로젝트 이해

SKADA Practitioner 실습 문제 구성

탐색적 데이터 분석 (EDA)

데이터 전처리

머신러닝 모델 적용

머신러닝 모델 고도화

심화 문제

수강 목표

목표 1

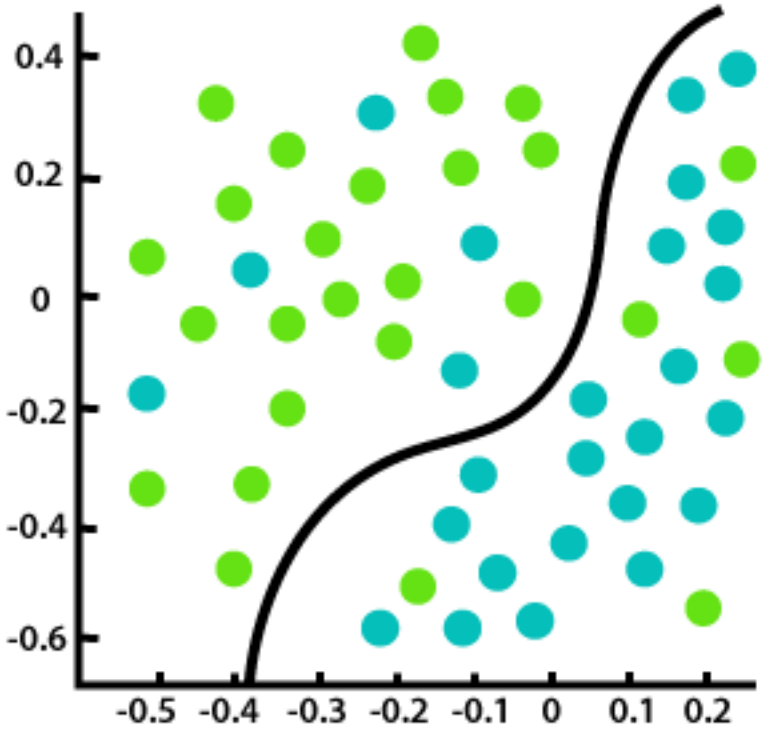
- 머신러닝을 활용하여 데이터를 분석하는 방법론을 이해할 수 있습니다.

목표 2

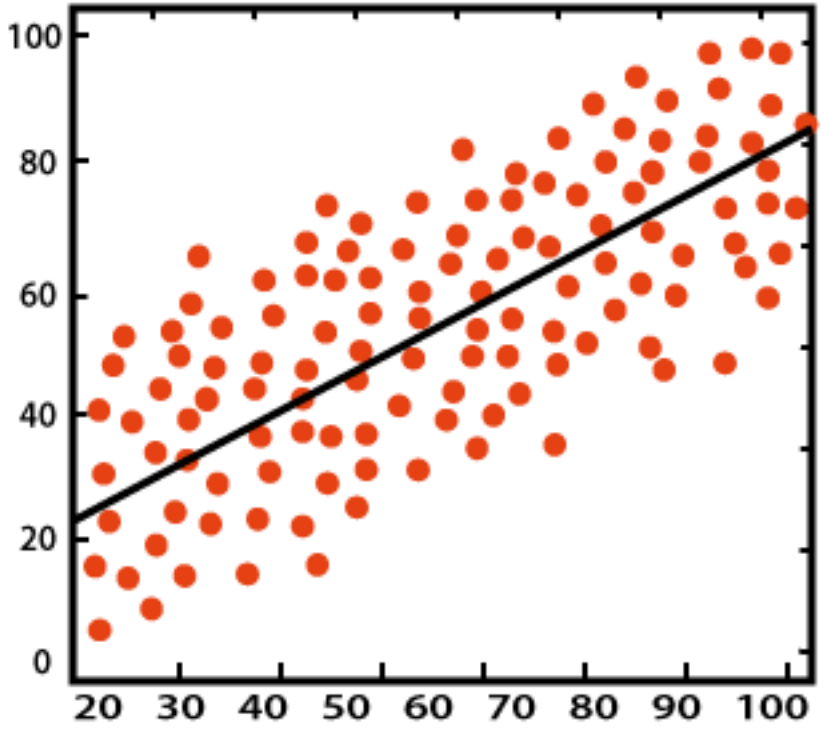
- 머신러닝 회귀, 분류 문제 해결 시 고려사항을 이해합니다.

01. 프로젝트 이해

머신러닝 회귀, 분류



프로젝트1: 분류 분석



프로젝트2: 회귀 분석

01. 프로젝트 이해

프로젝트 1: 반도체 제조 공정 이상 분류 모델 개발

분류 분석에 해당

	0	1	2	3	4	6	7	8	9	10
0	2989.82	2434.00	2180.0556	1031.0669	0.7565	104.7056	0.1226	1.5865	0.0080	-0.0216
1	3017.53	2524.09	2201.0667	880.2317	1.4148	106.5478	0.1211	1.3720	-0.0005	0.0052
2	3032.89	2500.81	2183.4333	1582.5646	1.3601	99.0267	0.1240	1.4615	-0.0034	-0.0042
3	3037.92	2361.50	2210.9778	1572.4698	1.0204	106.2089	0.1222	1.5251	0.0468	-0.0010
4	2982.67	2541.55	2173.4889	1145.7970	0.9402	104.0556	0.1247	1.3762	-0.0206	-0.0104
...	...	...	...	...	...	...	...	...	...	...
1632	2871.62	2530.11	2175.2556	1022.1660	1.2833	100.6222	0.1250	1.4679	-0.0033	-0.0022
1633	3244.74	2422.00	2208.5222	1838.7054	1.1571	95.2056	0.1249	1.5575	0.0049	-0.0207
1634	3118.63	2478.88	2170.5444	921.0605	1.4390	104.5300	0.1226	1.4385	-0.0029	0.0099
1635	2988.39	2493.72	2206.4000	982.5452	1.1853	116.8167	0.1228	1.5279	-0.0094	0.0001
1636	3052.98	2515.51	2172.8111	969.3436	1.2736	102.7367	0.1243	1.4065	-0.0037	-0.0058

01. 프로젝트 이해

반도체 제조 공정 모니터링을 위한 데이터셋 소개

UCI Secom Dataset

- 반도체 제조 공정에서 다양한 센서와 측정 지점에서 수집된 신호들로 구성
- 1567개의 관측치가 있으며 591개의 feature로 구성되어 있음
  - 관측치에 비해 feature의 수가 많은 fat 데이터셋으로 볼 수 있음

- 타겟 변수는 정상(0), 비정상(1)으로 기록되어 있음
  - 비정상 데이터는 157개로써 전체 데이터셋의 10% 수준

	0	1	2	3	4	6	7	8	9	10	...	580	581	582	583	584	585	586	587	588	589
0	2989.82	2434.00	2180.0556	1031.0669	0.7565	104.7056	0.1226	1.5865	0.0080	-0.0216	...	NaN	NaN	0.4952	0.0136	0.0041	2.7510	0.0104	0.0083	0.0024	79.8045
1	3017.53	2524.09	2201.0667	880.2317	1.4148	106.5478	0.1211	1.3720	-0.0005	0.0052	...	NaN	NaN	0.4998	0.0097	0.0026	1.9495	0.0328	0.0235	0.0068	71.5333
2	3032.89	2500.81	2183.4333	1582.5646	1.3601	99.0267	0.1240	1.4615	-0.0034	-0.0042	...	0.0055	114.4127	0.4961	0.0124	0.0036	2.4896	0.0155	0.0177	0.0055	114.4127
3	3037.92	2361.50	2210.9778	1572.4698	1.0204	106.2089	0.1222	1.5251	0.0468	-0.0010	...	NaN	NaN	0.4965	0.0186	0.0042	3.7541	0.0335	0.0084	0.0030	25.1494
4	2982.67	2541.55	2173.4889	1145.7970	0.9402	104.0556	0.1247	1.3762	-0.0206	-0.0104	...	NaN	NaN	0.4949	0.0146	0.0033	2.9448	0.0137	0.0326	0.0108	237.4625
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1632	2871.62	2530.11	2175.2556	1022.1660	1.2833	100.6222	0.1250	1.4679	-0.0033	-0.0022	...	0.0037	39.4516	0.5012	0.0131	0.0037	2.4087	0.0283	0.0112	0.0037	39.4516
1633	3244.74	2422.00	2208.5222	1838.7054	1.1571	95.2056	0.1249	1.5575	0.0049	-0.0207	...	NaN	NaN	0.5027	0.0184	0.0038	3.5629	0.0189	0.0059	0.0017	31.0252
1634	3118.63	2478.88	2170.5444	921.0605	1.4390	104.5300	0.1226	1.4385	-0.0029	0.0099	...	NaN	NaN	0.4994	0.0160	0.0047	3.2946	0.0075	0.0112	0.0041	150.3448
1635	2988.39	2493.72	2206.4000	982.5452	1.1853	116.8167	0.1228	1.5279	-0.0094	0.0001	...	0.0039	0.0000	0.4972	0.0154	0.0035	3.3992	-0.0006	0.0118	0.0039	0.0000
1636	3052.98	2515.51	2172.8111	969.3436	1.2736	102.7367	0.1243	1.4065	-0.0037	-0.0058	...	0.0059	52.7014	0.5081	0.0158	0.0037	3.4106	0.0302	0.0159	0.0059	52.7014

01. 프로젝트 이해

프로젝트 2: 전기차 배터리 사용량 예측 모델 개발

자료 출처: SKT, 서울시 자율주행차 길 넓힌다 – SK텔레콤 뉴스룸 (sktelecom.com)

- 측정 값: 외기 온도, 실내 온도, 고도, 속도, 전압, 전류, 잔여 배터리
- 예측 대상: 배터리 사용량
- 회귀 분석에 해당

01. 프로젝트 이해

전기차 운행 모니터링

	Time [s]	Velocity [km/h]	Elevation [m]	Throttle [%]	Motor Torque [Nm]	Longitudinal Acceleration [m/s^2]	Regenerative Braking Signal	Battery Voltage [V]	Battery Current [A]	Battery Temperature [°C]	...	AirCon Power [kW]	Heater Signal	Heater Voltage [V]	Heater Current [A]	Ambient Temperature [°C]	Coolant Temperature Heatercore [°C]	Requested Coolant Temperature [°C]
0	0.0	0.0	574.0	0.0	0.0	-0.03	0.0	391.4	-2.20	21.0	...	0.4	1	0	0	25.5	0	0
1	0.1	0.0	574.0	0.0	0.0	0.00	0.0	391.4	-2.21	21.0	...	0.4	1	0	0	25.5	0	0
2	0.2	0.0	574.0	0.0	0.0	-0.01	0.0	391.4	-2.26	21.0	...	0.4	1	0	0	25.5	0	0
3	0.3	0.0	574.0	0.0	0.0	-0.03	0.0	391.4	-2.30	21.0	...	0.4	1	0	0	25.5	0	0
4	0.4	0.0	574.0	0.0	0.0	-0.03	0.0	391.4	-2.30	21.0	...	0.4	1	0	0	25.5	0	0

- 주행기록은 크게 네 가지 종류의 피처로 구성
  - 환경 관련 피처: ambient temperature(외기 온도), elevation(고도) 등
  - 자동차 관련 피처: velocity(속도), throttle(가속 조절 장치), regenerative braking(회생 제동) 등
  - 배터리 관련 피처: voltage(전압), current(전류), SoC(state of charge, 충전량) 등
  - 난방 회로 관련 피처: cabin temperature(실내온도), coolant temperature(냉각수 온도), heating power(난방출력) 등

프로젝트 이해

SKADA Practitioner 실습 문제 구성

탐색적 데이터 분석 (EDA)

데이터 전처리

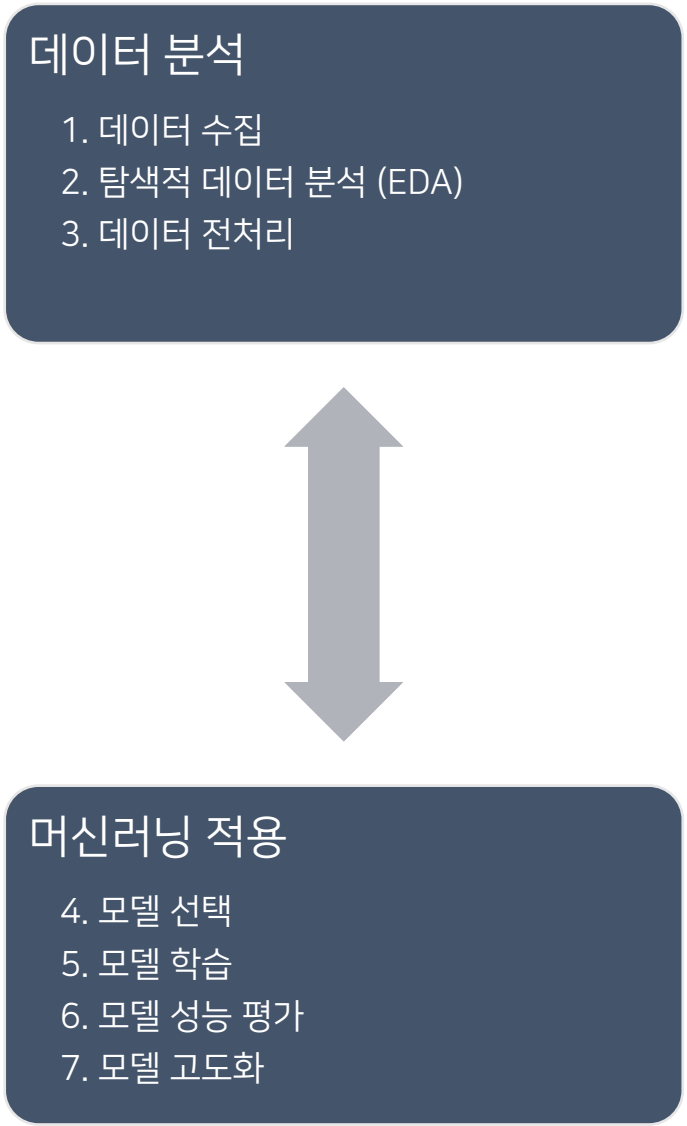
머신러닝 모델 적용

머신러닝 모델 고도화

심화 문제

02. SKADA Practitioner 실습 문제 구성

머신러닝 프로젝트 진행 과정



일반적인 머신러닝 프로젝트는 크게 **데이터 분석 단계와 머신러닝 적용 단계**로 나눌 수 있다.

보통 1~7의 과정으로 진행된다.

데이터 분석 과정에서는 데이터를 수집하고 분석한 후, 데이터와 Task에 맞춰 전처리를 진행한다.

머신러닝 적용 과정에서는 테스크와 전처리한 데이터를 바탕으로 모델을 선택, 학습, 평가를 진행한다.

필요에 따라 다양한 기법을 적용하여 머신러닝 모델을 고도화 할 수 있다.

02. SKADA Practitioner 실습 문제 구성

SKADA Practitioner 실습 문제

데이터 분석

1. 데이터 수집
2. 탐색적 데이터 분석 (EDA)
3. 데이터 전처리

머신러닝 적용

4. 모델 선택
5. 모델 학습
6. 모델 성능 평가
7. 모델 고도화

SKADA Practitioner 실습 문제는 일반적으로 1. 데이터 수집 과정까지 완료된 상황이 주어지며, 수집된 데이터를 **프로그램에 불러오는** 문제가 주어질 수 있다.

[문제 1-1]

[데이터 설명 1-1]

현장 담당자로부터 전달 받은 데이터의 상세 정보는 다음과 같다.

- 제조 공정명 : 반도체 제조 공정
- 수집 장비 : 반도체 제조 설비 내 센서 데이터
- 데이터셋 구조 : 테이블 형식(Tabular)로, 총 474개의 칼럼과 1637개의 샘플로 구성되어 있다.
- 데이터셋 저장 상태: feature와 label은 각각 `dataset/feature_data` 와 `dataset/label_data` 폴더에 총 아홉개씩의 파일로 저장되어 있다.

SKADA Practitioner 실습 문제에서 제공된 데이터

또한, 2. 탐색적 데이터 분석 및 3. 데이터 전처리 과정에 대한 가이드라인을 제공한다.

[문제 1-1] 아래 내용에 따라 데이터 전처리를 수행하시오. (6점)

- 각 주행기록 (이름이 Trip으로 시작하는 파일)을 살펴보고, 모든 주행기록에 **공통으로** 존재하는 열만 남기고 나머지 열은 제거한다.
- 모든 주행기록에 **공통으로** 존재하는 열의 이름들을 **selected_cols에 저장한다.** (열 이름의 예시: 'Elevation [m]', 'SoC [%]' 등)

SKADA Practitioner 실습 문제에서 제공된 데이터 전처리 가이드라인.

프로젝트 이해

SKADA Practitioner 실습 문제 구성

탐색적 데이터 분석 (EDA)

데이터 전처리

머신러닝 모델 적용

머신러닝 모델 고도화

심화 문제

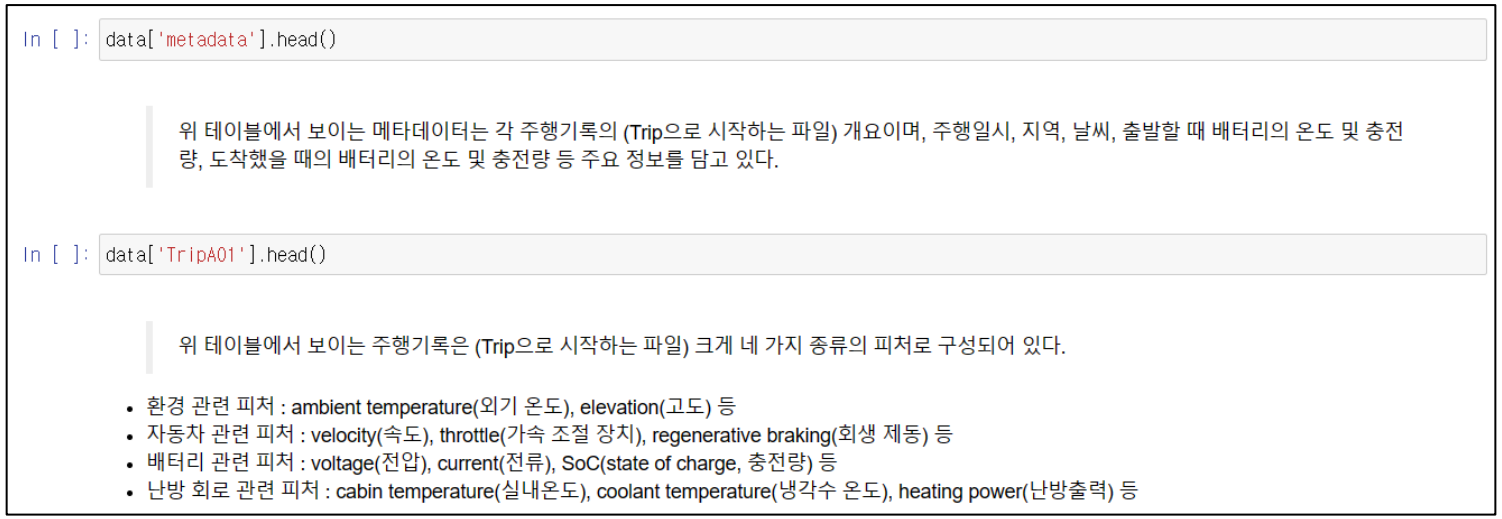
03. 탐색적 데이터 분석 (EDA)

탐색적 데이터 분석 (EDA) 이란?

데이터를 분석하여 데이터의 특징, 패턴, 상관 관계 등을 확인하는 과정.

데이터의 특징을 파악하고 필요한 전처리 기법과 모델을 선택하는데 도움을 줄 수 있다.

SKADA 문제에서는 탐색적 데이터 분석 과정이 문제로 출제될 수 있다.
또한 탐색적 데이터 분석 결과를 바탕으로 **데이터 전처리 코드를** 작성하는 문제가 주어질 수 있다.
필요에 따라서 수험자 각자의 탐색적 데이터 분석 과정을 바탕으로 주어진 문제들을 해결할 수 있다.



In [ ]:
```
data['metadata'].head()
```

> 위 테이블에서 보이는 메타데이터는 각 주행기록의 (Trip으로 시작하는 파일) 개요이며, 주행일시, 지역, 날씨, 출발할 때 배터리의 온도 및 충전량, 도착했을 때의 배터리의 온도 및 충전량 등 주요 정보를 담고 있다.

In [ ]:
```
data['TripA01'].head()
```

> 위 테이블에서 보이는 주행기록은 (Trip으로 시작하는 파일) 크게 네 가지 종류의 피처로 구성되어 있다.

- 환경 관련 피처 : ambient temperature(외기 온도), elevation(고도) 등
- 자동차 관련 피처 : velocity(속도), throttle(가속 조절 장치), regenerative braking(회생 제동) 등
- 배터리 관련 피처 : voltage(전압), current(전류), SoC(state of charge, 충전량) 등
- 난방 회로 관련 피처 : cabin temperature(실내온도), coolant temperature(냉각수 온도), heating power(난방출력) 등

SKADA 기출 문항 예시

03. 탐색적 데이터 분석 (EDA)

탐색적 데이터 분석 (EDA) – 데이터 확인

데이터를 열람하여 데이터의 기초적인 특성을 확인하는 과정.

[문제 1-2]

[데이터 설명 1-2]

현장 담당자로부터 전달 받은 데이터의 상세 정보는 다음과 같다.

- 제조 공정명 : 반도체 제조 공정
- 수집 장비 : 반도체 제조 설비 내 센서 데이터
- 데이터셋 구조 : 테이블 형식(Tabular)로, 총 474개의 칼럼과 1637개의 샘플로 구성되어 있다.

Feature 변수 `x` 와 label 변수 `y` 가 갖고 있는 각 **칼럼의 이름**과 의미는 다음과 같다.

- ***변수*** `x`
  0~473: 반도체 제조 설비 내 센서 데이터로 측정된 익명화된 정보
- ***변수*** `y`
  Pass/Fail: 공정 과정 중 이상 샘플 테스트 통과 여부 (0: 통과, 1: 실패)

SKADA 기출 문항 예시

03. 탐색적 데이터 분석 (EDA)

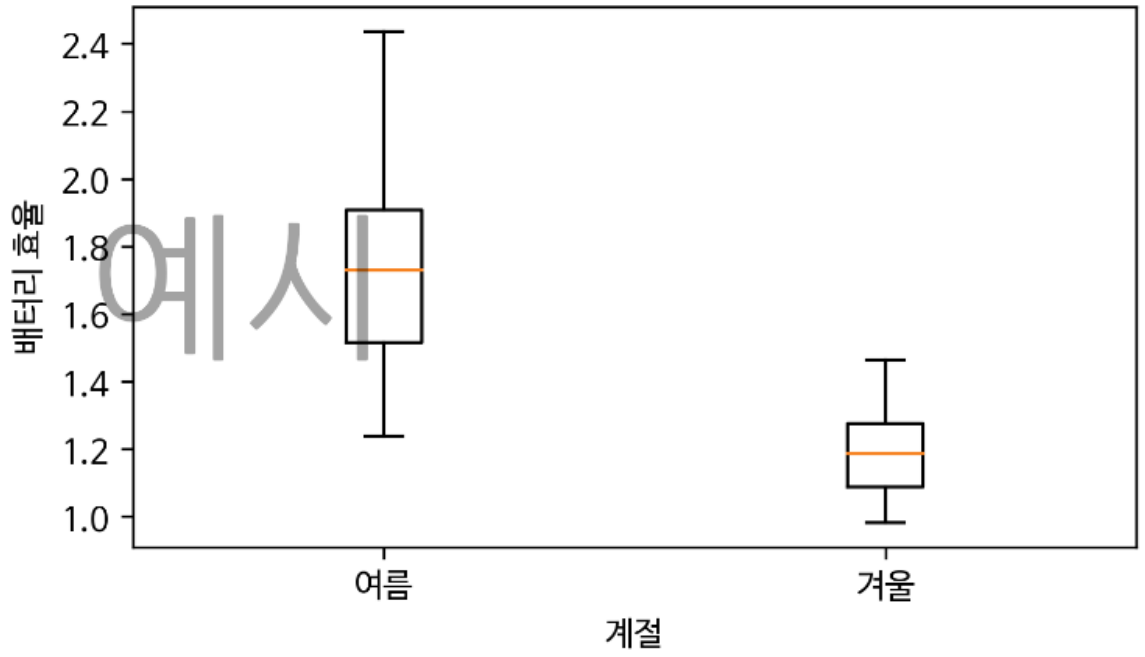
탐색적 데이터 분석 (EDA) – 상관관계 분석

- 분석하고자 하는 변수 간의 상관관계를 분석하는 과정.
- 입력 Feature 간의 상관관계를 분석하여 불필요한 특성을 제거할 수도 있고,
  입력 Feature와 목표 Feature 간의 상관관계를 분석하여 모델링 전략을 수립할 수도 있다.

- SKADA 문제에서는 필요한 상관관계 분석 상황이 주어지고, 이를 구현하는 문제가 출제될 수 있다.

[문제 2-2] 계절별 배터리 효율을 비교하는 박스 플롯 (box plot)을 그리는 코드를 완성하시오. (6점)

- 여름철 주행기록은 TripA로 시작하는 파일에, 겨울철은 TripB로 시작하는 파일에 기술되어 있다.
- 여름철 배터리 효율 수치와, 겨울철 배터리 효율 수치를 각각 **summer_r, winter_r 리스트에 저장한다** `(List[float])` **형식.**
- 리스트를 완성하면 주어진 코드에 의해 아래 예시와 똑같은 박스 플롯이 그려진다.



SKADA 기출 문항 예시

03. 탐색적 데이터 분석 (EDA)

탐색적 데이터 분석 (EDA) – 이상치, 결측치 처리

- 이상치 (Outlier) : 다른 데이터와 크게 다른 값을 나타내는 데이터. 잘못 측정된 데이터일 가능성이 높다.
  - 일반적으로 제거하며, 다른 데이터의 평균, 최빈값 등 통계치로 대체할 수 있다.
  - 이상 탐지 (Anomaly Detection) Task일 경우 이상치를 찾는 것이 목적이므로 이상치 처리에 신중해야 한다.
- 결측치 (Missing Value): 기록되지 않아 데이터 값이 없는 데이터
  - 제거하거나 다른 데이터의 평균, 최빈값 등 통계치로 대체할 수 있다.

- SKADA 문제에서는 이상치, 결측치 처리 상황 및 처리 기법이 주어지고, 이를 구현하는 문제가 출제될 수 있다.
  - 이상치, 결측치 처리는 데이터 전처리 과정에서 수행하기도 한다.

[문제 3-1번 상황설명]

머신러닝을 활용해 본격적인 분석에 들어가 보기로 한다.

모델 학습을 준비하던 중 주행기록에 데이터값이 없는, 결측치가 존재한다는 사실을 발견했다.

다행히 대부분의 자료에는 한두 개의 결측치만 존재했지만, 어떤 기록은 결측치가 너무 많아 자료로 활용할 수 없을 정도였다.

그래서 결측치를 효과적으로 처리하여 데이터의 완성도를 높이기로 했다.

[문제 3-1] 아래 내용을 따라, 결측치를 채우거나 특정 주행 기록을 분석 대상에서 제외하시오. (6점)

- 기본적으로 결측치는 직전 행의 값으로 채우도록 한다 (직전 행의 값이 5라면, 결측치를 5로 채운다).
- 만약 결측치를 채우기 위해 사용할 직전 행이 없다면, 직후 행의 값으로 채운다 (보통 가장 첫 행에 해당한다).
- 결측치가 5개 이상인 열이 하나라도 존재하는 주행기록은 분석 대상에서 제외하기 위해, 주행기록의 이름(파일 이름)을 *removed_keys*에 추가한다

SKADA 기출 문항 예시

프로젝트 이해

SKADA Practitioner 실습 문제 구성

탐색적 데이터 분석 (EDA)

데이터 전처리

머신러닝 모델 적용

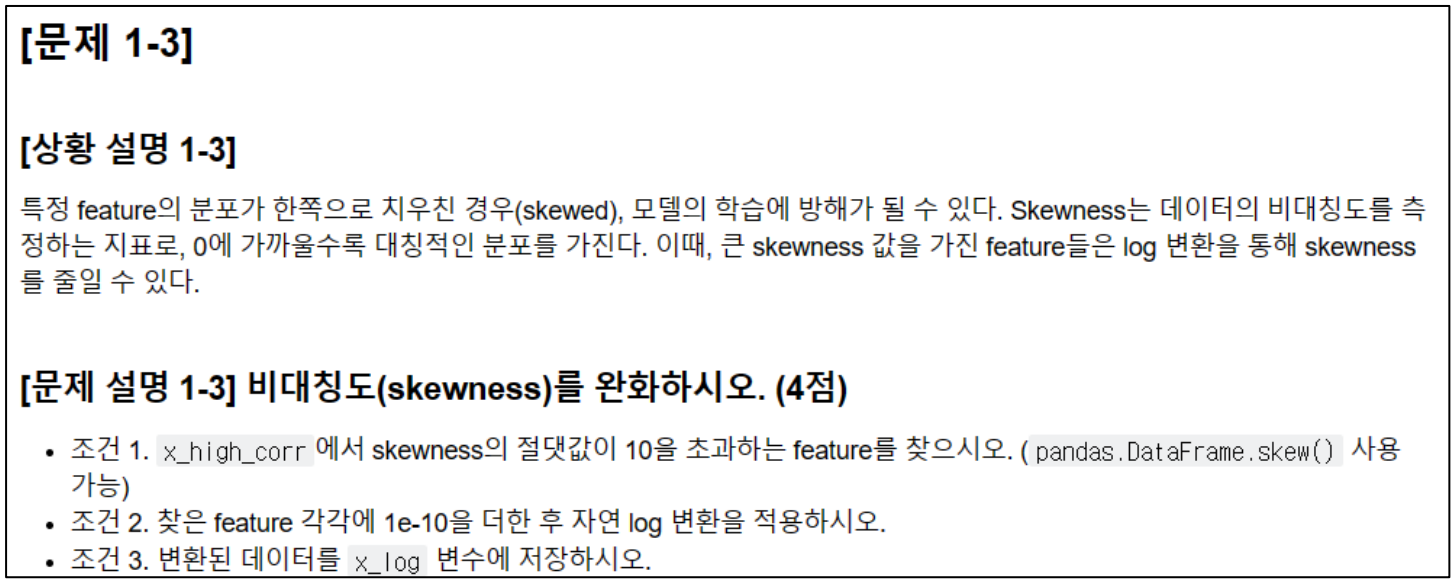
머신러닝 모델 고도화

심화 문제

04. 데이터 전처리

데이터 전처리란?

1. 아래 예시와 같이 데이터의 분포가 한쪽으로 치우쳤거나, 데이터의 형태가 불규칙적인 경우 등 모델이 데이터를 해석하는데 방해가 되는 요소를 완화
   - 예시: 비대칭도 완화, 샘플링 등
2. 데이터를 모델의 특성에 맞게 재구성하여 모델이 데이터를 더 잘 해석할 수 있도록 하는 과정
   - 예시: 정규화 및 표준화, 인코딩, 범주화 등



[문제 1-3]

[상황 설명 1-3]

특정 feature의 분포가 한쪽으로 치우친 경우(skewed), 모델의 학습에 방해가 될 수 있다. Skewness는 데이터의 비대칭도를 측정하는 지표로, 0에 가까울수록 대칭적인 분포를 가진다. 이때, 큰 skewness 값을 가진 feature들은 log 변환을 통해 skewness를 줄일 수 있다.

[문제 설명 1-3] 비대칭도(skewness)를 완화하시오. (4점)

- 조건 1. `x_high_corr` 에서 skewness의 절댓값이 10을 초과하는 feature를 찾으시오. (`pandas.DataFrame.skew()` 사용 가능)
- 조건 2. 찾은 feature 각각에 1e-10을 더한 후 자연 log 변환을 적용하시오.
- 조건 3. 변환된 데이터를 `x_log` 변수에 저장하시오.

SKADA 기출 문항 예시

04. 데이터 전처리

특성 엔지니어링

- 확인한 데이터에서 불필요한 데이터나 중복된 데이터를 제거하는 과정.
- 효과적인 분석을 위해 데이터를 보다 유용하게 만드는 과정을 의미한다.

[상황 설명 1-2]

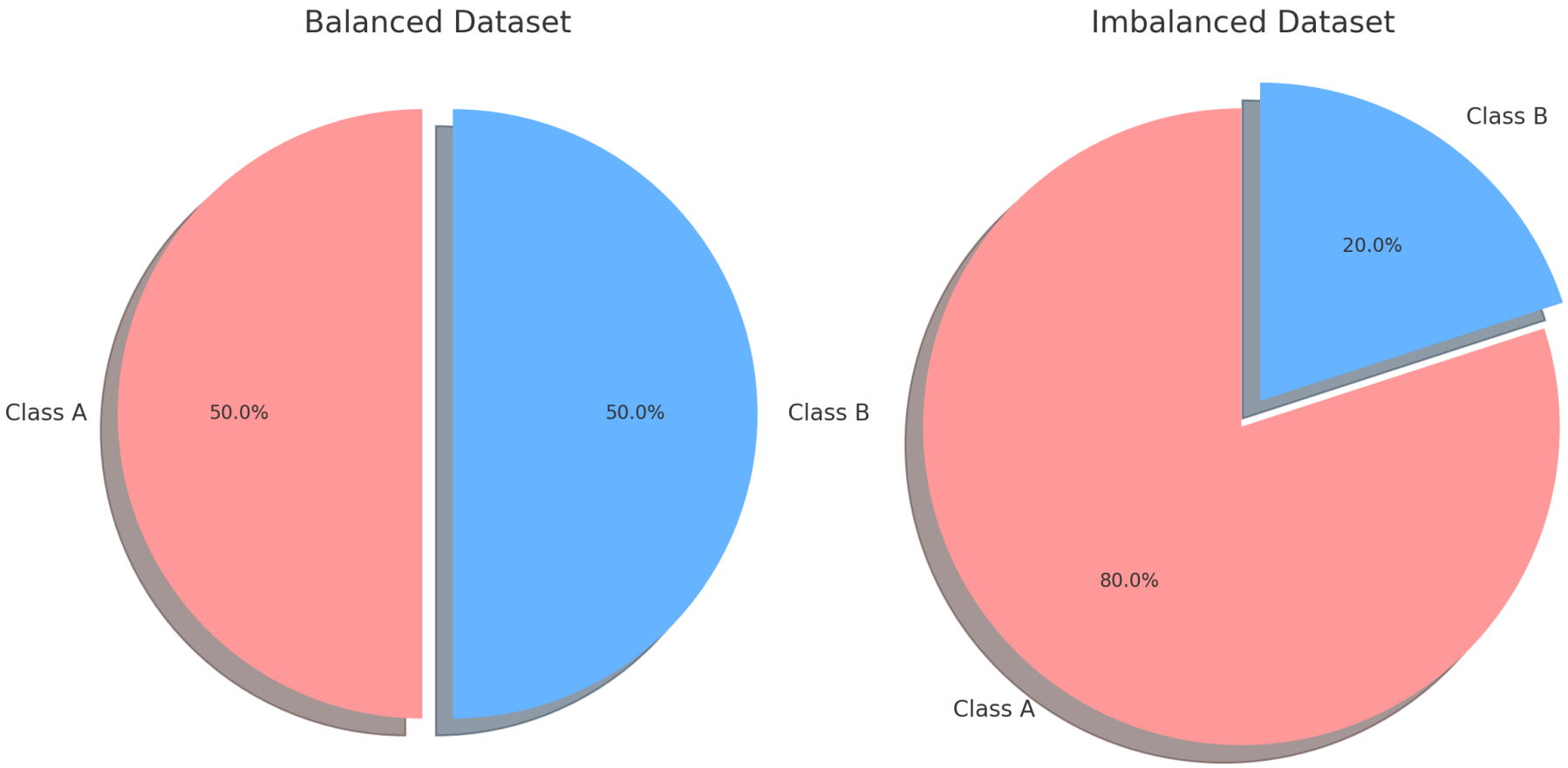
가장 먼저, 불러온 데이터를 이용하여 feature engineering을 수행한다. 이 데이터셋은 관측 샘플 수에 비해 많은 feature를 갖고 있는 데이터셋으로써, 효과적인 분석을 위해 아래와 같은 feature engineering 작업이 필수적이다.

- Feature selection: 중요하지 않은 feature를 제거하여 모델의 정확성을 향상시키고, 오버피팅을 방지한다.
- Feature scaling: feature의 스케일을 조정하여 모델의 학습을 안정화시킨다.
- Missing value 처리: Missing value를 처리하여 데이터의 완결성을 높인다.

SKADA 기출 문항 예시

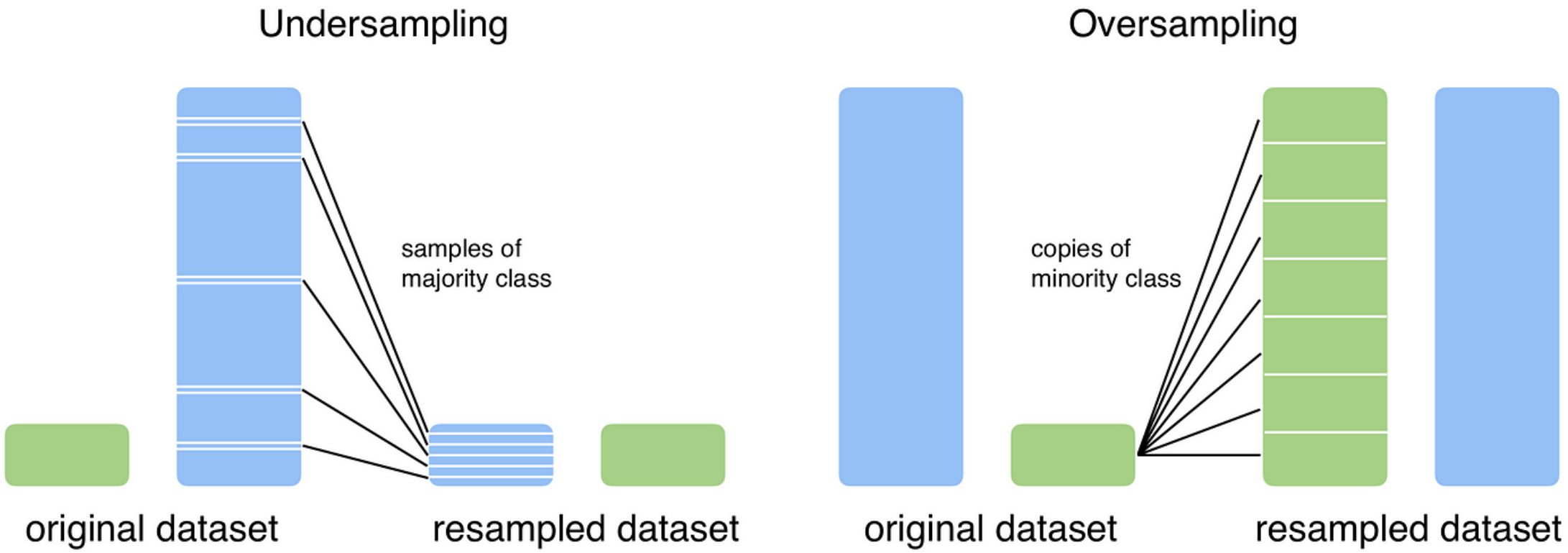
04. 데이터 전처리

Imbalanced dataset 문제 소개



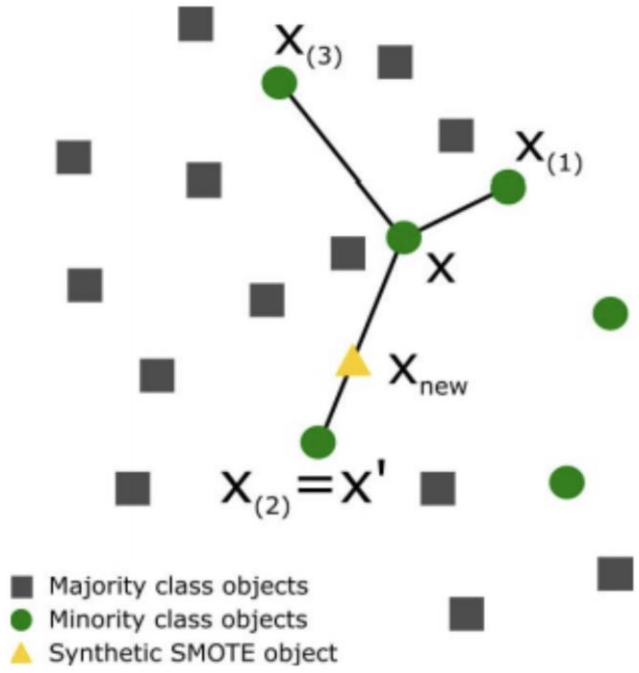
04. 데이터 전처리

Random sampling을 이용한 Imbalance 문제 해결



04. 데이터 전처리

SMOTE 알고리즘에 대한 이해



SMOTE 동작 과정:

1. **Minority Class Sample 선택**: 소수 클래스에서 랜덤하게 하나의 인스턴스를 선택합니다.
2. **k-Nearest Neighbors 계산**: 선택된 인스턴스와 가장 가까운 거리에 있는 k개의 소수 클래스 샘플을 찾습니다.
3. **Synthetic Sample 생성**: 선택된 인스턴스와 k개의 이웃 중 하나 사이의 선형 보간을 통하여 새로운 인스턴스를 생성합니다.

[문제 설명 3-1] 데이터 불균형을 처리하시오. (4점)

- 조건 1. SMOTE를 활용하여 불균형을 처리한다. (imblearn 라이브러리 사용 가능)
- 조건 2. 결과 변수명은 `x_train_pca`, `y_train`에서 `x_resampled`, `y_resampled`로 변경한다.

SKADA 기출 문항 예시

04. 데이터 전처리

모델에 따른 데이터 전처리

모델의 종류나 코드 구현체 등에 따라 효과적으로 데이터를 분석하기 위해 데이터를 전처리 할 수 있다.

- 데이터를 분류할 때 딥러닝 / 머신러닝 계열 모델을 활용한다면 데이터 정규화 과정이 필요할 수 있다.
- 수치 변수 (Numerical variable)만 인풋으로 사용될 수 있는 모델을 사용할 경우 범주화 변수 (Categorical variable)을 원 핫 벡터 또는 임베딩 모델 사용 등이 수반되어야 한다.
- 아래 상황에서는 머신러닝 모델에서 수치 연산 알고리즘에 numpy 라이브러리로 구현되어 있으므로 데이터도 그에 맞춰서 변환해줘야 한다.

[문제 4번 상황설명]

머신러닝 모델을 학습시키고 테스트하기 위해서는, 그에 맞는 데이터셋이 준비되어야 한다.

앞의 세 문제 (문제 1, 2, 3) 을 해결하지 못했더라도 뒤에 등장할 문제를 풀어볼 기회를 주기 위해, 미리 모든 전처리가 된 파일을 제공한다.

아래의 데이터 프레임 - train_x, train_y, test_x, test_y - 은 문제 1, 2, 3의 전처리 과정들을 수행한 후, selected_keys에 포함된 주행기록을 3:1의 비율로 학습용과 테스트용으로 분할한 결과이다.

또한 머신러닝에 사용되는 수치 연산 알고리즘은 numpy 라이브러리를 기반으로 하므로, pandas dataframe 형태의 데이터를 numpy 형식으로 변환하였다.

```
loaded = np.load('./example/preprocessed_data.npz')
train_x, train_y = loaded['train_x'], loaded['train_y'].ravel()
test_x, test_y = loaded['test_x'], loaded['test_y'].ravel()
```

SKADA 기출 문항 예시

프로젝트 이해

SKADA Practitioner 실습 문제 구성

탐색적 데이터 분석 (EDA)

데이터 전처리

머신러닝 모델 적용

머신러닝 모델 고도화

심화 문제

05. 머신러닝 모델 적용

머신러닝 모델 적용 과정

머신러닝 모델을 설계, 구현하고 Task에 적용하는 과정.

SKADA 시험에서는 대부분 모델 종류가 주어지고, 주어진 모델을 전처리한 데이터에 구현 및 적용하는 문제가 주어진다.

실제 현업에서 많이 사용되는 Boosting 계열 모델이 출제된 경우가 있음.

[문제 4-1] 아래 내용에 따라, Gradient Boosting 모델을 학습하는 코드를 작성하시오. (5점)

- 주어진 sklearn의 GradientBoosting 클래스를 (*my_model* 변수)을 사용해서 학습을 진행한다.
- *train_x*, *train_y* 를 학습 데이터로 사용한다.
- 학습된 모델로 예측치 *train_p*와 *test_p*를 계산한다. 이때, *train_p* 와 *test_p* 는 각각 *train_x* 와 *test_x*를 학습된 모델로 예측한 결과다.

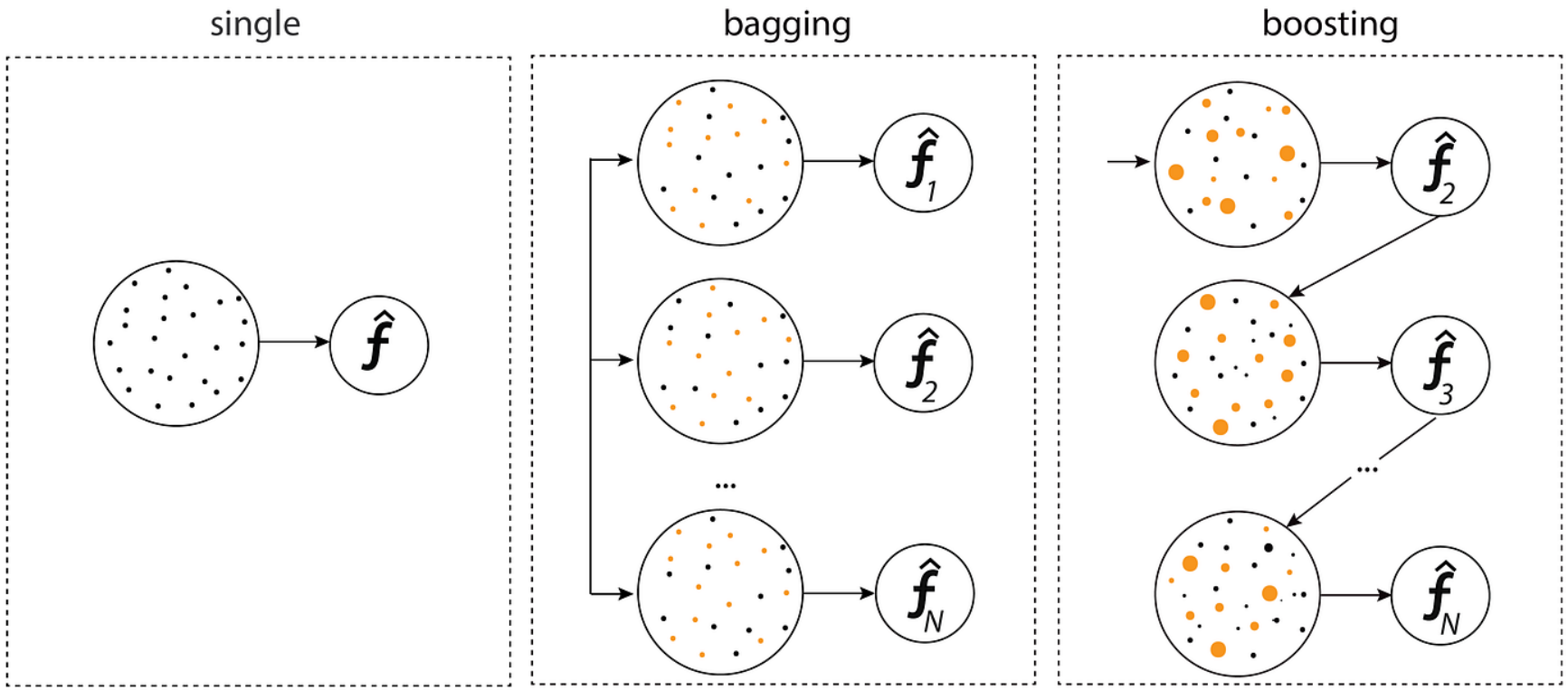
```
# 평가를 위한 랜덤 시드 고정, 절대 수정 금지
seed = 3064

### 문제 4-2 문제 풀이 시작 ###
my_model = GradientBoostingRegressor(random_state = seed)


####### answer #######

# ... 빈칸 채우기

# train_p   = ...
# test_p    = ...

####### answer #######
```

SKADA 기출 문항 예시

05. 머신러닝 모델 적용

Boosting 알고리즘 소개

- 여러 모델 (예: Decision tree)을 순차적으로 학습시키는 Ensemble 방법
- 각 모델은 이전에 학습된 모델이 잘못 예측한 데이터에 집중하여 학습이 진행
- 반면 Bagging은 여러 모델을 병렬적으로 학습시킴

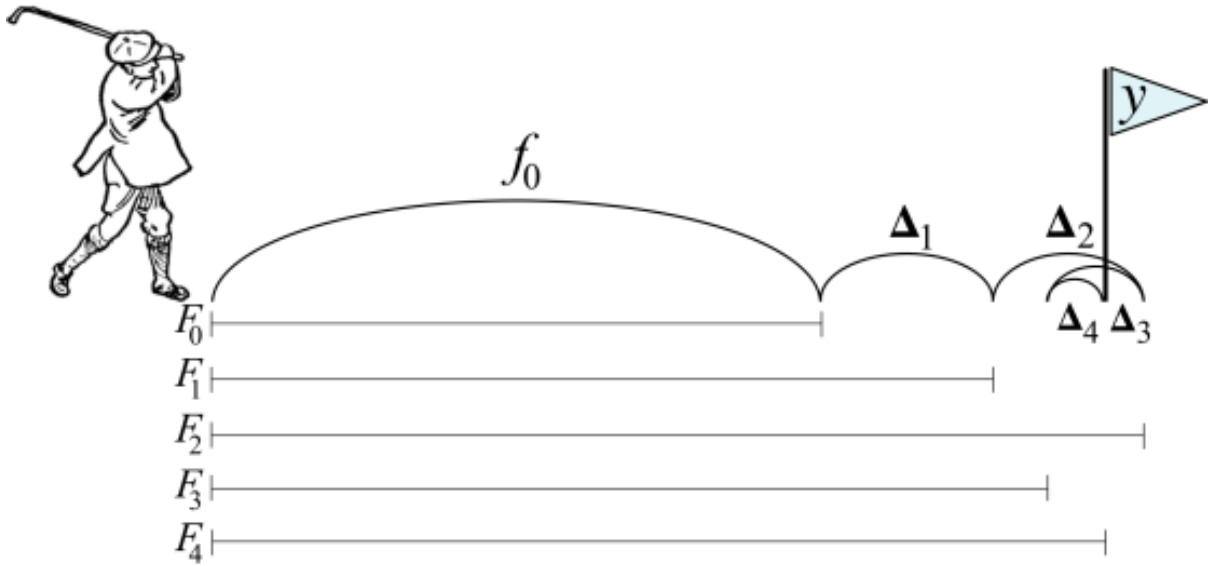


05. 머신러닝 모델 적용

Gradient Boosting Machine (GBM) 소개

- Gradient boosting = Boosting with gradient descent

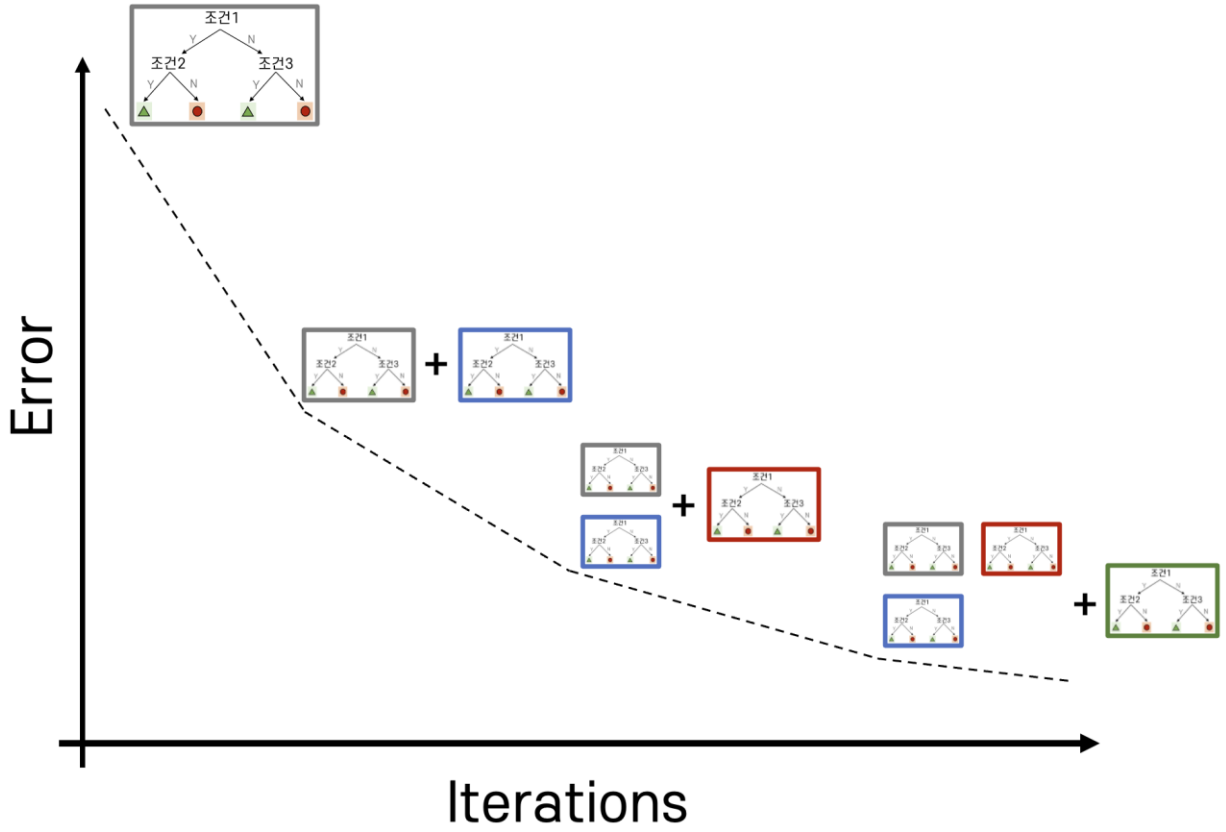
직관적 이해
- 골프에서 샷을 칠 때, 공은 바로 홀에 들어가지는 않지만 점점 가까워짐
- 다음 샷은 이전 샷의 실수를 바로잡아 목표에 더 가까이 가게 함
- GBM은 이전 모델의 오류를 바로잡는 새로운 모델을 하나씩 추가함 → 모델의 성능을 점진적으로 향상



05. 머신러닝 모델 적용

Gradient Boosting Machine (GBM)

- 최종 GBM모델은 지금까지 학습된 모델들의 합으로 구성
  - $\mathrm{GBM}(X) = F_1(X) + F_2(X) + F_3(X) + \cdots + F_T(X)$
- 모델 $F_i(X)$가 추가될 때 마다 모델의 학습 데이터에 대한 에러는 점점 작아짐



05. 머신러닝 모델 적용

XGBoost 소개

- Decision tree를 base learner로 사용한 GBM 모델의 대표 라이브러리
  - 링크: https://xgboost.ai/

- GBM에 여러가지 기능을 추가하여 모델을 최적화
  - 빠른 학습 속도를 위해 Tree 구성 시 병렬 처리 기능을 구현
  - Overfitting을 방지하고자 L_1, L_2 정규화와 같은 여러 정규화 기법을 추가적으로 활용
  - 그 외 결측치 자동 처리, 교차 검증 등의 기능이 내장

→ GBM보다 계산 속도 및 예측 정확도 측면에서 모두 더 좋은 성능을 보임

- Kaggle 대회에서 우승 또는 상위 랭크를 차지한 많은 솔루션들에 XGBoost가 포함

05. 머신러닝 모델 적용

XGBoost의 파라미터에 대한 이해

Hyperparameter	Description
learning_rate (eta)	Controls the step size at each iteration to prevent overfitting. Lower value requires more trees.
n_estimators	Number of boosting rounds or trees to build. Too many may lead to overfitting.
max_depth	Maximum depth of a tree. Increasing this value makes the model more complex.
min_child_weight	Minimum sum of instance weight needed in a child. Higher values prevent learning specific relations.
subsample	Fraction of training data sampled for each tree. Prevents overfitting, but too low can cause underfitting.
colsample_bytree	Fraction of features sampled for each tree. Similar to max_features in RandomForest.
gamma (min_split_loss)	Minimum loss reduction required for a further partition on a leaf node. Higher values make the model conservative.
reg_lambda	L2 regularization term on weights. Increasing it makes the model more conservative.
reg_alpha	L1 regularization term on weights. Helps in preventing overfitting.

프로젝트 이해

SKADA Practitioner 실습 문제 구성

탐색적 데이터 분석 (EDA)

데이터 전처리

머신러닝 모델 적용

머신러닝 모델 고도화

심화 문제

06. 머신러닝 모델 고도화

머신러닝 모델을 고도화 하여 Task에 대한 성능을 향상 시키는 과정이다.
하이퍼파라미터 튜닝과 여러 개의 모델을 한번에 사용하는 Ensemble 알고리즘 구현 등 다양한 기법을 적용할 수 있다.

SKADA 시험에서는 대부분 머신러닝 모델 고도화 기법이 하나 혹은 여럿 주어지고, 가이드를 따라 해당 기법에 대한 코드를 구현하는 문제가 주어진다.

이전 시험에서는 하이퍼파라미터 튜닝과 Stacking이라는 앙상블 알고리즘 구현이 출제되었다.

[문제 5번 상황설명]

학습된 모델이 잘 최적화 되어있는가를 검토해 볼 시간이다.

개발한 모델을 개선하기 위해 하이퍼파라미터 튜닝을 시도 해 보기로 했다.

[문제 5] 하이퍼파라미터 튜닝을 통해 가장 우수한 Gradient Boosting 모델을 선택하는 코드를 완성하시오. (10점)

- 문제 4 의 코드를 재활용할 수 있다.
- 채점을 위해 random_state을 미리 정의된 *seed* 로 설정한다.
- 코드에서 정의한 하이퍼라미터 목록인 *MAX_DEPTH*, *LEARNING_RATE*, *N_ESTIMATORS* 의 모든 조합을 기록한 리스트인 *hyperparameters* 대해 grid search를 이용한 튜닝을 수행한다.
- 각 모델은 *train_x*로 학습한 뒤, *test_x*에 대한 mae를 측정하고, 이 수치가 가장 낮은 모델을 선택한다.
- 선택된 모델의 하이퍼파라미터 *best_hyperparameter* 를 구하는 것이 최종 목표이다. 이는 튜플(Tuple) 형식으로 `Tuple[int, float, int]` (max_depth, learning_rate, n_estimator) 세 개의 값들을 저장한다.

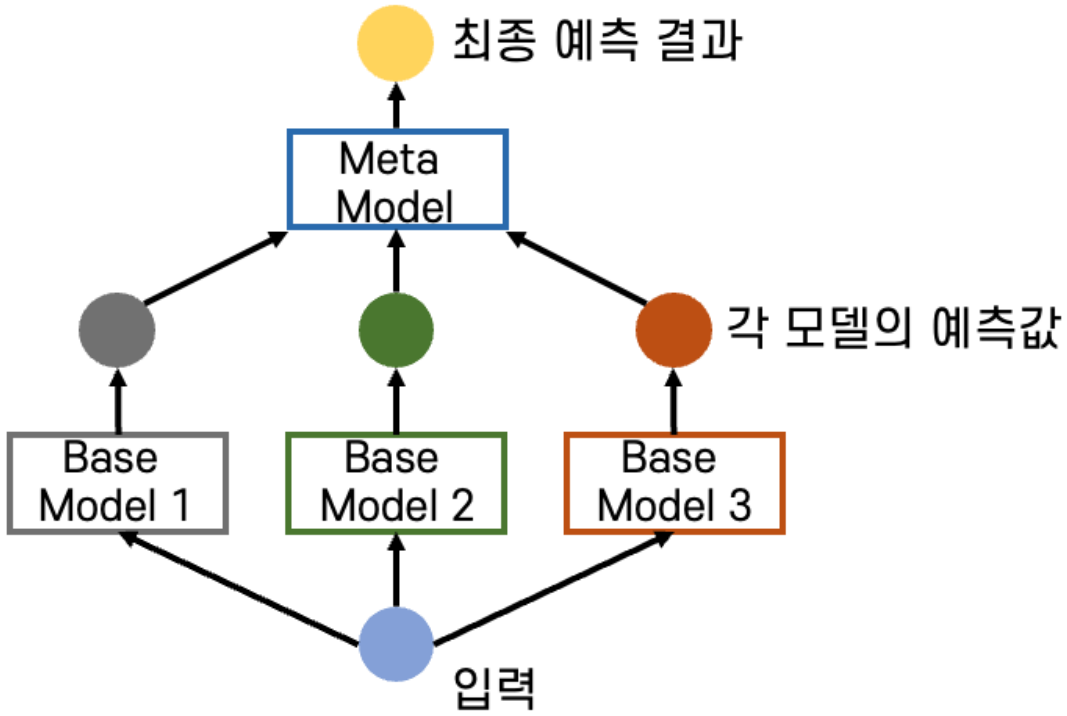
SKADA 기출 문항 예시

```
In [ ]: # 평가를 위한 랜덤 시드 고정, 절대 수정 금지
seed = 3064
MAX_DEPTH = [2, 3, 4]
LEARNING_RATE = [0.1, 0.05, 0.01]
N_ESTIMATORS = [60, 80, 100, 120]

best_hyperparameter = None
best_model = None
best_mae = 1000000

### Q5 문제 풀이 시작 ###
hyperparameters = list(itertools.product(MAX_DEPTH, LEARNING_RATE, N_ESTIMATORS))

for max_depth, lr, n_estimators in hyperparameters:

    ####### answer #######

    raise NotImplementedError # 정답 코드 작성 후 제거

    ####### answer #######


#####################

print('[info] Hyperparameter tuning finished')
print('[info] Best hyperparameters: {}'.format(best_hyperparameter))
print('[info] Best test mae: {}'.format(best_mae))
```

06. 머신러닝 모델 고도화

Stacking 알고리즘이란?

- 여러 Base model의 결과를 Meta model이 활용하여 최종 예측을 하는 방법
  - Base model: SVM, GBM , XGBoost, Neural network등을 이용하여 독립적으로 학습된 모델
  - Meta model: Base model들의 예측 결과를 학습 데이터로 사용하여 최종 예측 진행



06. 머신러닝 모델 고도화

Stacking 알고리즘이란?

- 실제 출제된 고도화 기법 문항

[이론 설명 4-1]

스태킹(Stacking)은 여러 개의 모델의 예측 결과를 입력으로 사용하여 새로운 메타 모델을 학습시키는 앙상블 기법이다. 베이스 모델은 원본 데이터로부터 예측을 수행하며, 메타 모델은 베이스 모델들의 예측 결과를 바탕으로 최종 예측을 수행한다. 이 방법은 다양한 모델의 강점을 결합하여 전체적인 성능을 향상시키는 데 도움을 준다.

[문제 설명 4-1] 스태킹 모델을 학습하시오. (6점)

- 조건 1. 베이스 모델로 열개의 XGBoost classifier를 사용한다. 모든 모델의 하이퍼파라미터는 아래와 같이 설정한다:
  - gamma=0.5
  - learning_rate=0.1
  - max_depth=4
  - n_estimators=20
- 조건 2. 각 베이스 모델의 `random_state`는 41~50 중 서로 다른 값을 사용한다.
- 조건 3. scikit-learn 라이브러리에 구현되어 있는 stacking classifier를 활용한다.
- 조건 4. 메타 모델로는 Logistic regression을 사용한다.
- 조건 5. StackingClassifier 객체의 변수 이름은 `stacking_clf`로 설정한다.

※ 모델이 정상적으로 학습하는 경우 **5분** 이내에 학습이 완료되는 것을 확인 할 수 있다.

```
# ============================================================ #
#         START OF YOUR CODE (DO NOT DELETE/MODIFY THIS LINE)        #
# ============================================================ #
#
#                이 곳에서 코드를 수정 및 작성하시오.
#
# ============================================================ #
#          END OF YOUR CODE (DO NOT DELETE/MODIFY THIS LINE)         #
# ============================================================ #


y_pred_stack = stacking_clf.predict(x_resampled)
accuracy = accuracy_score(y_resampled, y_pred_stack)
print(f"Stacking Model Accuracy: {accuracy:.4f}")
```

SKADA 기출 문항 예시

06. 머신러닝 모델 고도화

Stacking 알고리즘의 의미

- 여러 모델을 예측에 활용한다면, 그 출력을 어떻게 조합할까?

- 여러 모델의 출력을 조합하는 대표적인 방법은 평균과 가중 평균이 존재
  - 평균: $\hat{y} = \frac{1}{N}\sum_{i=1}^{N} f_i(X)$
  - 가중 평균: $\hat{y} = \sum_{i=1}^{N} \alpha_i f_i(X)$

- Stacking은 여러 모델의 출력을 조합할 때 Meta model이 최적 조합을 학습
  - Stacking: $\hat{y} = g(f_1(X), f_2(X), \ldots, f_N(X))$

[문제 4-2]

[문제 설명 4-2] 메타 모델의 최적의 하이퍼파라미터를 찾으시오. (6점)

- 조건 1. 스태킹 모델의 메타 모델의 하이퍼파라미터를 찾는다.
- 조건 2. `GridSearchCV` 함수를 사용한다. (`sklearn.model_selection.GridSearchCV`)
- 조건 3. `param_grid` 변수에 저장된 값들을 하이퍼파라미터 후보로 사용한다.
- 조건 4. 하이퍼파라미터 선택 기준은 ROC-AUC가 높은 값을 기준으로 한다.
- 조건 5. 3-fold cross validation을 사용한다.
- 조건 6. `GridSearchCV` 객체의 변수 이름은 `grid_search` 로 설정한다.

SKADA 기출 문항 예시

프로젝트 이해

SKADA Practitioner 실습 문제 구성

탐색적 데이터 분석 (EDA)

데이터 전처리

머신러닝 모델 적용

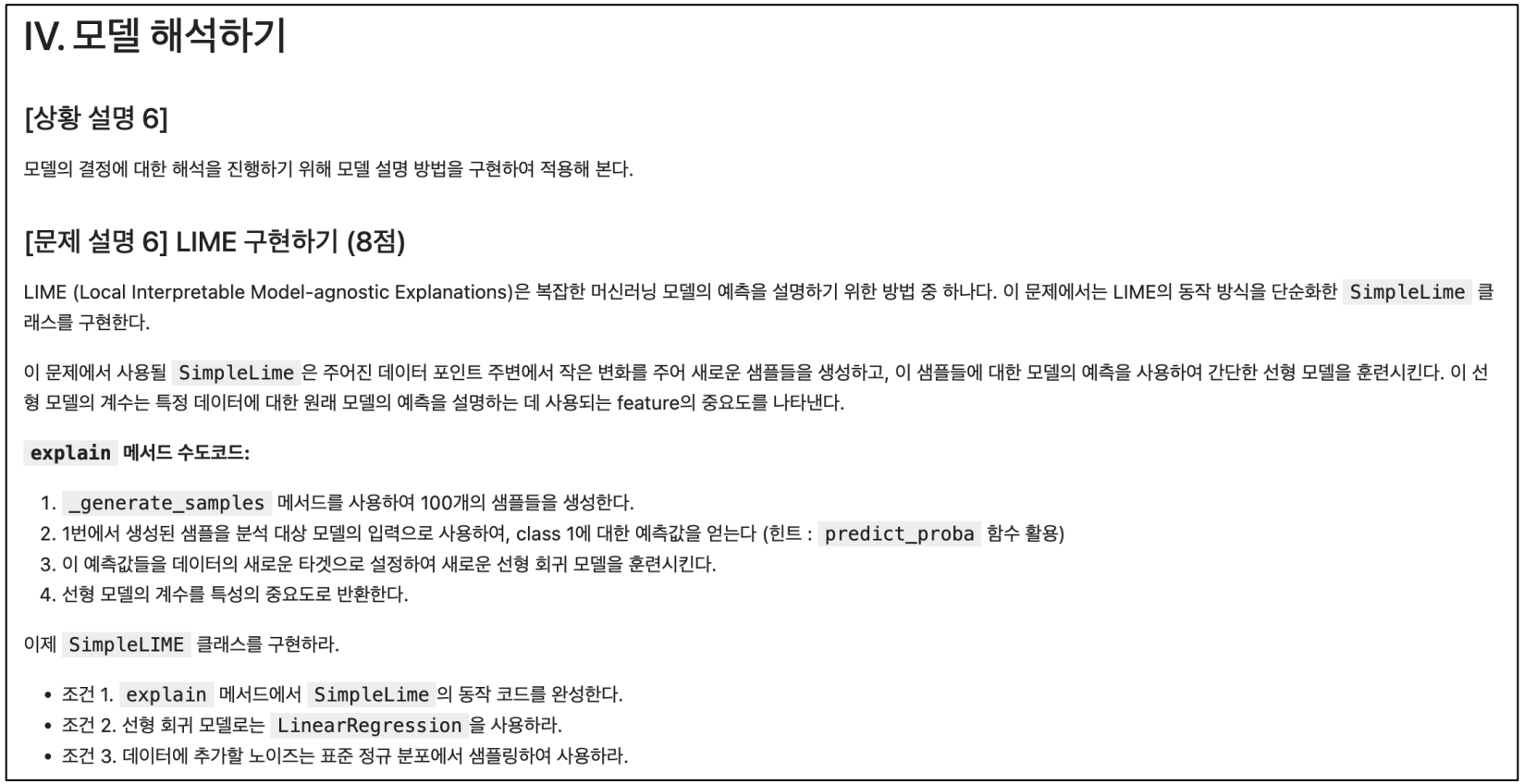
머신러닝 모델 고도화

심화 문제

07. 심화 문제

SKADA 시험의 마지막 문제는 앞서 진행한 과정을 바탕으로 상대적으로 어려운 문제가 주어지며,
보통 모델의 결과를 분석하고 해석하는 문제가 주로 출제된다.

실제 시험에서는 구현한 머신러닝 모델을 사람이 이해할 수 있도록 해석하는 문제가 주어진 적이 있다.



IV. 모델 해석하기

[상황 설명 6]

모델의 결정에 대한 해석을 진행하기 위해 모델 설명 방법을 구현하여 적용해 본다.

[문제 설명 6] LIME 구현하기 (8점)

LIME (Local Interpretable Model-agnostic Explanations)은 복잡한 머신러닝 모델의 예측을 설명하기 위한 방법 중 하나다. 이 문제에서는 LIME의 동작 방식을 단순화한 `SimpleLime` 클래스를 구현한다.

이 문제에서 사용될 `SimpleLime`은 주어진 데이터 포인트 주변에서 작은 변화를 주어 새로운 샘플들을 생성하고, 이 샘플들에 대한 모델의 예측을 사용하여 간단한 선형 모델을 훈련시킨다. 이 선형 모델의 계수는 특정 데이터에 대한 원래 모델의 예측을 설명하는 데 사용되는 feature의 중요도를 나타낸다.

`explain` 메서드 수도코드:

1. `_generate_samples` 메서드를 사용하여 100개의 샘플들을 생성한다.
2. 1번에서 생성된 샘플을 분석 대상 모델의 입력으로 사용하여, class 1에 대한 예측값을 얻는다 (힌트 : `predict_proba` 함수 활용)
3. 이 예측값들을 데이터의 새로운 타겟으로 설정하여 새로운 선형 회귀 모델을 훈련시킨다.
4. 선형 모델의 계수를 특성의 중요도로 반환한다.

이제 `SimpleLIME` 클래스를 구현하라.

- 조건 1. `explain` 메서드에서 `SimpleLime`의 동작 코드를 완성한다.
- 조건 2. 선형 회귀 모델로는 `LinearRegression`을 사용하라.
- 조건 3. 데이터에 추가할 노이즈는 표준 정규 분포에서 샘플링하여 사용하라.

SKADA 기출 문항 예시

07. 심화 문제

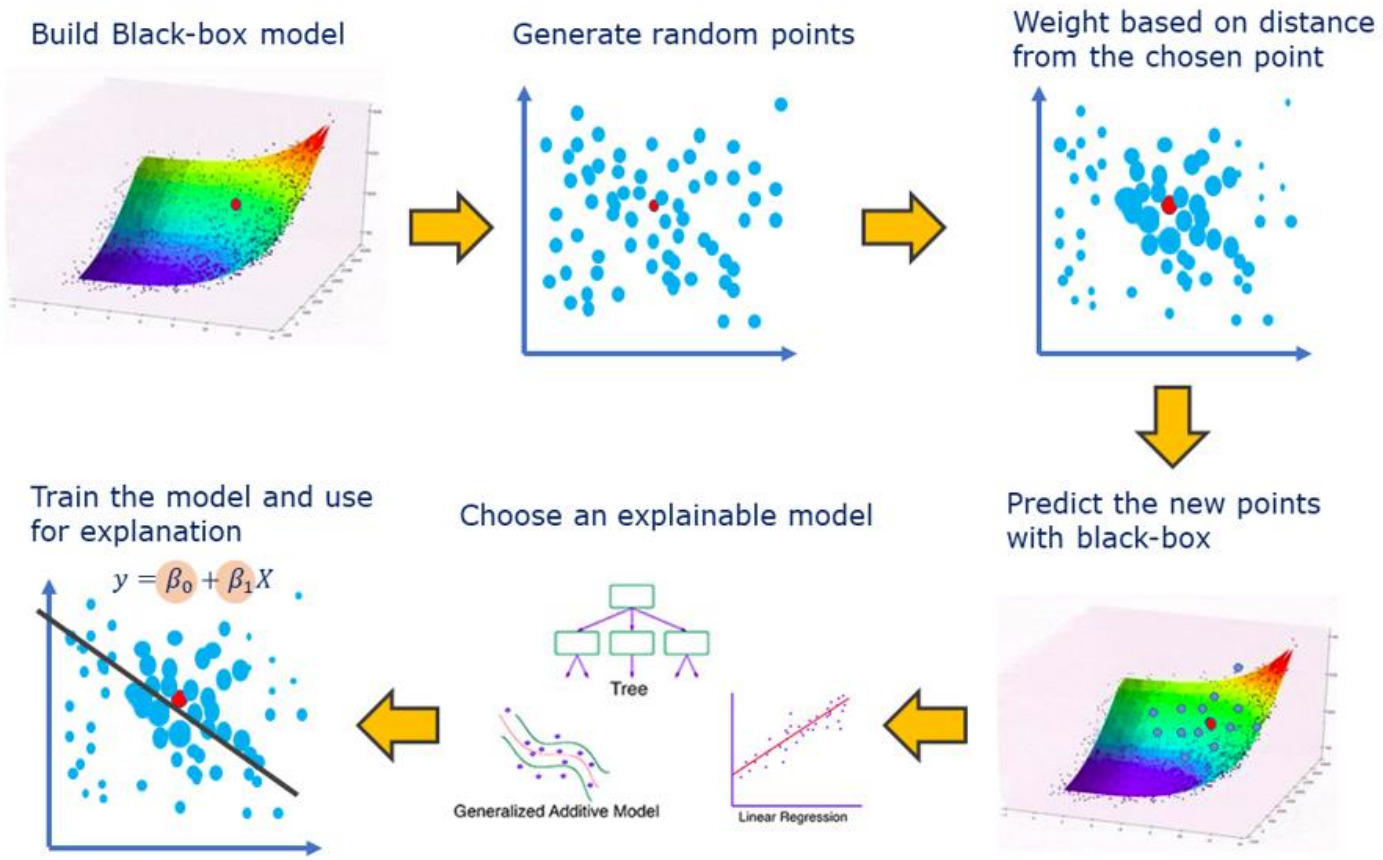
처음 보는 알고리즘이더라도 수험자들이 문제를 해결할 수 있도록 충분한 설명이 제공되며, 그러한 부분만 구현을 요구한다.

```python
class SimpleLIME:
    def __init__(self, model: Any, num_samples: int = 100):
        """
        Initialize the SimpleLIME.

        Parameters:
        - model: The black-box model we want to explain.
        - num_samples: Number of perturbed samples to generate.
        """
        self.model = model
        self.num_samples = num_samples

    def _generate_samples(self, data_point: np.ndarray) -> np.ndarray:
        """
        Generate perturbed samples around a given data point.

        Parameters:
        - data_point: The data point around which to generate samples.

        Returns:
        - Perturbed samples.
        """
        noise = np.random.normal(loc=0, scale=1, size=(self.num_samples, data_point.shape[0]))
        samples = data_point + noise

        return samples

    def explain(self, data_point: np.ndarray) -> np.ndarray:
        """
        Explain the prediction of a given data point.

        Parameters:
        - data_point: The data point to explain.

        Returns:
        - feature_importances: Feature importances.
        """
```

SKADA 기출 문항 예시

07. 심화 문제

실제 시험에서는 LIME 알고리즘을 단순화한 버전인 SimpleLIME을 구현하였다.



SKADA 기출 문항 예시

07. 심화 문제

실제 시험에서는 PDP 알고리즘이 출제되었다.

이를 위해 머신러닝 모델 분석 기법 중 하나인 PDP(partial dependence plot)를 활용해 보기로 했다.

PDP는 특정 변수를 제외한 나머지 변수들이 결과에 영향을 미칠 수 없도록 한 후, 특정변수의 결과에 대한 영향력을 분석하는 기법이다.

이제 PDP를 통해 배터리 성능을 분석하여, S사와 Z사 양사에 보고할 내용을 찾아내 보자.

[문제 6] 아래의 설명을 따라 PDP를 계산하는 메소드인 *compute_pdp* 를 완성하시오. (8점)

1. 전체 n 개의 데이터 중, i 번째 데이터를 (x_i, y_i) 라고 하자. 여기서, x_i와 y_i는 각각 i 번째 입력과 출력 데이터이고, 입력 데이터는 총 k 개의 피쳐(열)로 이뤄져 있다. 다시 말해, $x_i = (x_{i1}, \ldots, x_{ik})$ 이다.
2. 이 데이터로 학습된 모델 f와 f가 계산한 예측치 p_i를 다음과 같이 표현하자: $p_i = f(x_{i1}, \ldots x_{ik}) = f(x_i)$.
3. i 번째 입력 데이터의 j 번째 피쳐(열)의 값을 m 번째 입력 데이터의 j 번째 피쳐(열)의 값으로 바꾼 $z_{imj} = (x_{i1}, \ldots, x_{mj}, \ldots, x_{ik})$ 를 정의하자.
4. m 과 j 는 고정한 채, $i = \{1, \ldots, n\}$에 대하여 $f(z_{imj})$ 를 계산하고, 이 값의 평균을 q_{mj}라고 하자.
5. 모든 $m = \{1, \ldots, n\}$에 대하여 (x_{mj}, q_{mj}) 를 계산하고, 이를 2차원 평면에 시각화하는 방식으로 PDP를 그린다.

SKADA 기출 문항 예시