3. 데이터 EDA

• INDEX

- 데이터 EDA
  - 데이터 시각화
- 모델 학습
- 성능 평가
- 성능 검증
- 비지도 학습
  - 군집화
  - 차원축소
- 참고

데이터 EDA

데이터 EDA

서울 18반 마이스터 트랙
이재윤

탐색적 데이터 분석(EDA, Exploratory Data Analysis)

- 데이터를 본격적으로 분석하기 전에 여러 각도에서 관찰하고 이해하는 과정
- 데이터에 대한 깊은 이해를 얻을 수 있으며, 모든 프로젝트의 첫 과정!
- 중요성
  1. 분석의 방향성 제시
     - 데이터의 특징을 파악하여 어떤 분석 기법을 적용하고, 어떤 모델을 구축할 지 결정하는 데 도움을 줌
  2. 모델의 성능 향상
     - 데이터의 문제점을 미리 파악하고 해결함으로써 모델의 정확성과 안정성을 높일 수 있음
  3. 새로운 인사이트 발견
     - 예상치 못한 데이터의 패턴이나 관계를 발견하여, 새로운 인사이트로 활용할 수 있음

서울 18반 마이스터 트랙
이재윤

탐색적 데이터 분석(EDA) 진행 순서

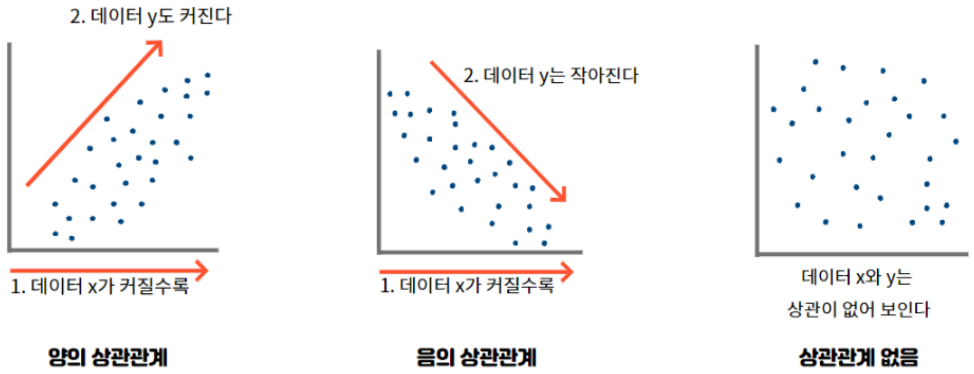
❖ 주의! 순서를 작성하긴 했지만, 이 과정은 항상 한 방향으로만 진행되는 것이 아닌, 각 단계를 오가면서 반복적으로 진행

1. 데이터를 불러오고, 기본 정보를 확인 (타입, 결측치, 통계량 등)
2. 데이터 정제 및 전처리
   - 결측치, 이상치, 중복값을 식별하고 처리하고, 올바르지 않은 데이터 타입 수정
   - 분석에 사용되지 않을 변수(열) 제거
3. 개별 변수 분석
   - 변수의 분포와 특징을 이해
4. 변수 간 관계 분석
   - 변수들 사이의 관계, 패턴, 상관관계를 파악
5. 분석 결과 정리

서울 18반 마이스터 트랙
이재윤

[참고] 상관관계

- 두 변수가 함께 변화하는 관련성을 의미
- 하나의 변수가 증가/감소할 때, 다른 변수가 어떤 일정한 관련성을 띄며 함께 변화하는지를 나타내는 지표
- 일반적으로 -1에서 1 사이의 값을 가지며, 이 값을 상관 계수(Correlation Coefficient)라고 함
  - 1. 양의 상관관계 (1에 가까움): 한 변수가 증가할 때 다른 변수도 증가하는 경향
  - 2. 음의 상관관계 (-1에 가까움): 한 변수가 증가할 때 다른 변수는 감소하는 경향
  - 3. 상관관계 없음 (0에 가까움): 두 변수 사이에 뚜렷한 선형 관계가 없음



❖ 상관관계가 높다고 해서 원인과 결과(인과관계)를 의미하는 것은 아님

데이터 시각화

서울 18반 마이스터 트랙
이재윤

데이터 시각화

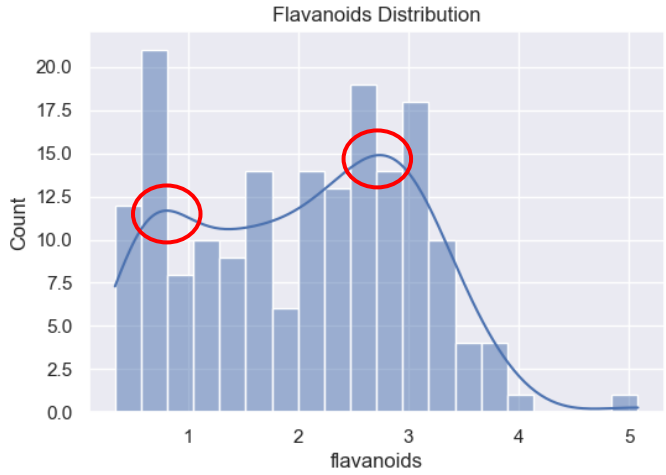
- 숫자로 가득한 표를 보는 것보다는 그래프를 보는 것이 데이터 패턴을 훨씬 직관적으로 파악할 수 있음
- Matplotlib 과 Seaborn 을 활용하여 데이터를 시각화
- **Matplotlib**
  - 파이썬 시각화의 근간이 되는 라이브러리
  - 그래프의 세부적인 요소, 크기, 레이아웃을 설정할 수 있음
- **Seaborn**
  - Matplotlib을 더 예쁘고 통계적으로 의미 있는 그래프로 쉽게 만들어주는 라이브러리
  - Matplotlib을 기반으로 동작함

➢ **Matplotlib**을 기반으로 제목/축/라벨 등을 추가하고, **Seaborn**으로 데이터를 그래프로 그릴 예정

서울 18반 마이스터 트랙
이재윤

단변수 분석 - 히스토그램(Histogram)

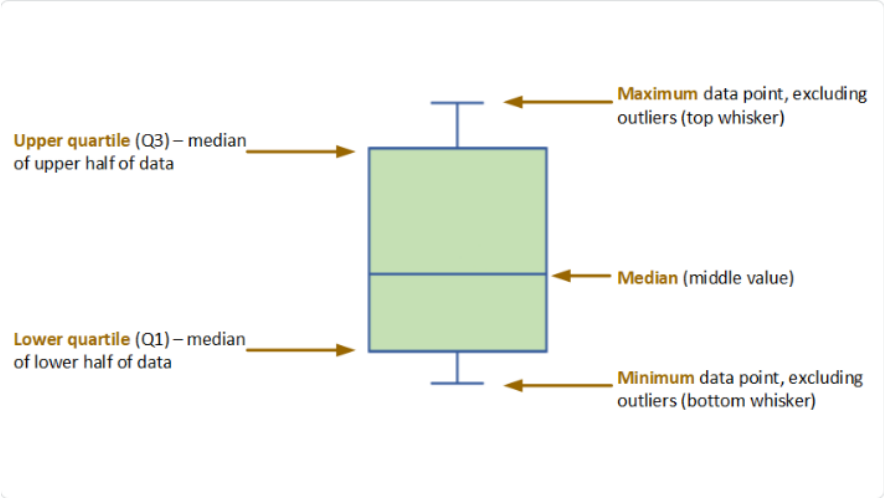
- 특정 수치형 변수 하나의 데이터 분포를 파악하기 위해서 사용
- 데이터의 전체 범위를 여러 개의 동일한 구간(bin)으로 나눈 뒤, 각 구간에 속하는 데이터의 개수를 막대 그래프로 표현
- 장점
  - 직관적으로 분포를 파악할 수 있음
  - 이상치 탐지가 수월
- 단점
  - 구간(bin) 설정에 민감
- 가령 이 그래프를 살펴보면, 봉우리가 2군데가 있는데, 이는 두 개의 서로 다른 하위 그룹이 데이터에 섞여있다는 의미
  - ex) 초등학교에 있는 모든 사람의 키를 그래프로 그릴 경우
    - 초등학생들의 평균 키를 중심으로 한 '초등학생 그룹'의 봉우리
    - 선생님들의 평균 키를 중심으로 한 '선생님 그룹'의 봉우리
    - 결국 서로 다른 특성을 가진 두 그룹의 데이터가 합쳐지면, 그래프에는 각각의 그룹을 대표하는 봉우리가 생김

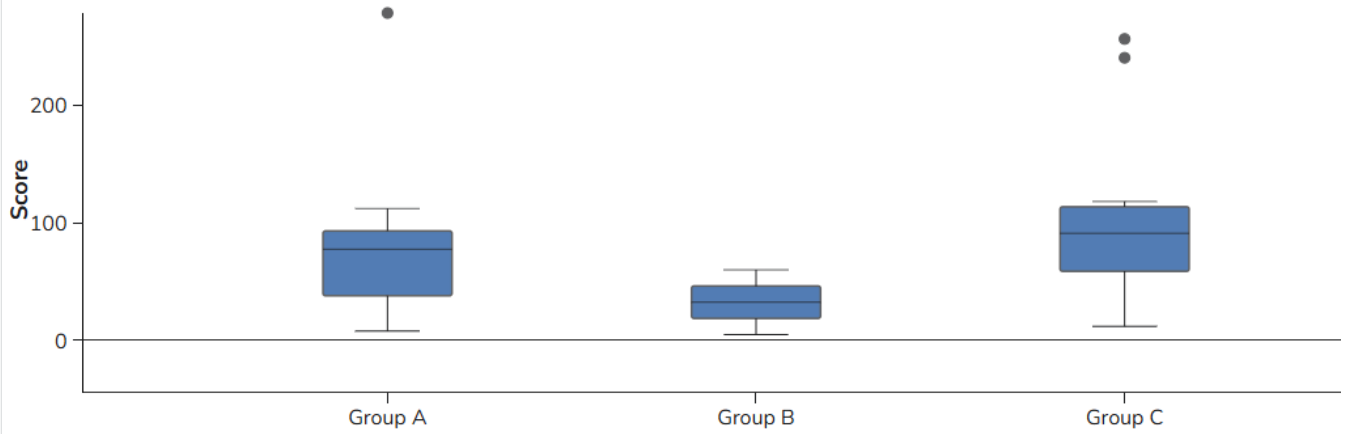


서울 18반 마이스터 트랙
이재윤

단변수 분석 - 박스 플롯(Bot Plot)

- 수치형 데이터의 분포와 이상치를 한눈에 파악하기 위해 사용되는 그래프
- 데이터의 사분위수(25%, 50%, 75%)를 이용하여 전체 데이터가 어떻게 퍼져있는지를 간결하게 요약
- 다른 값들에 비해 유독 크거나 작은 값, "이상치"를 찾는 데 매우 유용
- 데이터 개수를 파악하기 어렵고, 데이터의 구체적인 분포 형태를 알기는 어려움



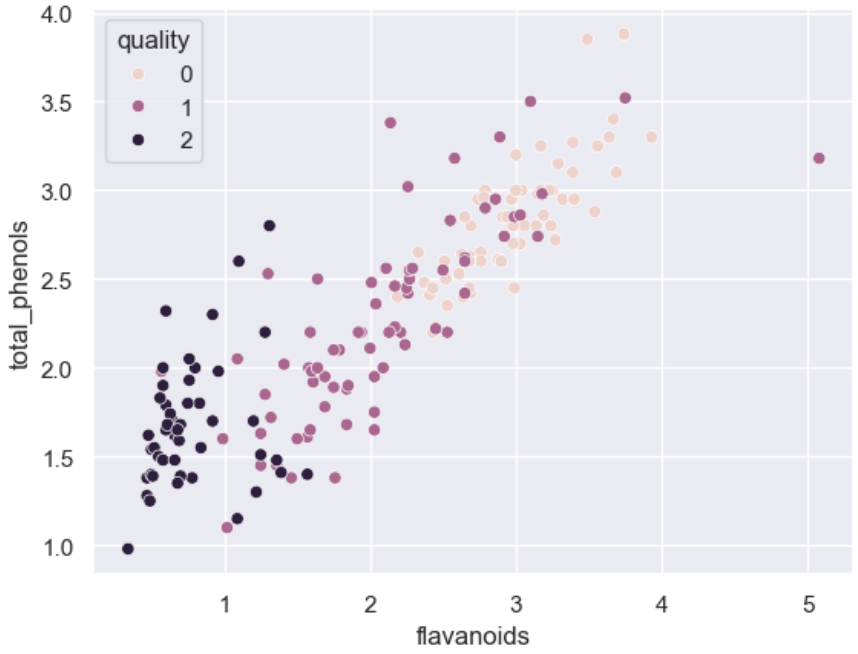


https://www.cdc.gov/cove/data-visualization-types/box-and-whiskers-plot.html

서울 18반 마이스터 트랙
이재윤

다변수 분석 - 산점도(Scatter Plot)

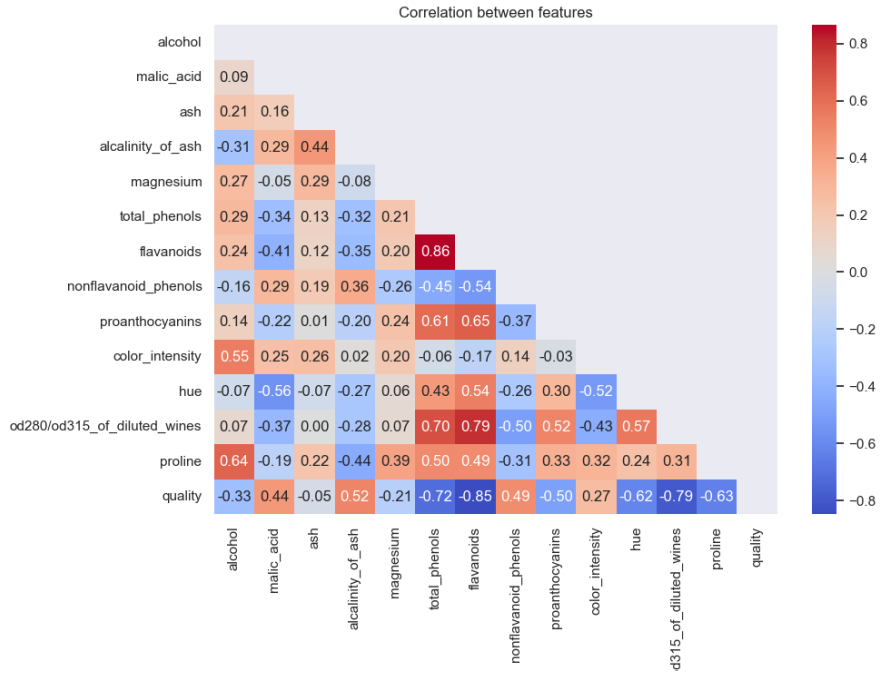
- 두 개의 수치형 변수 사이의 관계를 파악하기 위해 점을 찍어 만드는 그래프
- 두 변수가 어떤 관계를 갖고 있는 지 직관적으로 파악할 수 있음
- 장점
  - 패턴이나 이상치를 쉽게 발견할 수 있음
- 단점
  - 데이터가 너무 많으면 점들이 서로 겹쳐서 분포를 제대로 파악하기 어려움
  - 상관관계를 보여줄 뿐, 인과관계를 증명하지 않음



서울 18반 마이스터 트랙
이재윤

다변수 분석 - 히트맵(Heatmap)

- 여러 숫자형 변수들 간의 크기를 색상으로 변환하여 보여주는 것
- 주로 상관관계 분석에 많이 사용 됨
- 수 많은 변수 간의 관계를 한 번에 살펴보고, 상관관계가 높은 변수 쌍을 찾아내기 유용
- 장점
  - 직관적이고, 다변량 관계를 파악할 수 있음
- 단점
  - 정확한 수치를 파악하기 어려움
  - 변수가 너무 많으면 히트맵이 지나치게 커지고 복잡해져서 한눈에 파악하기 어려워짐

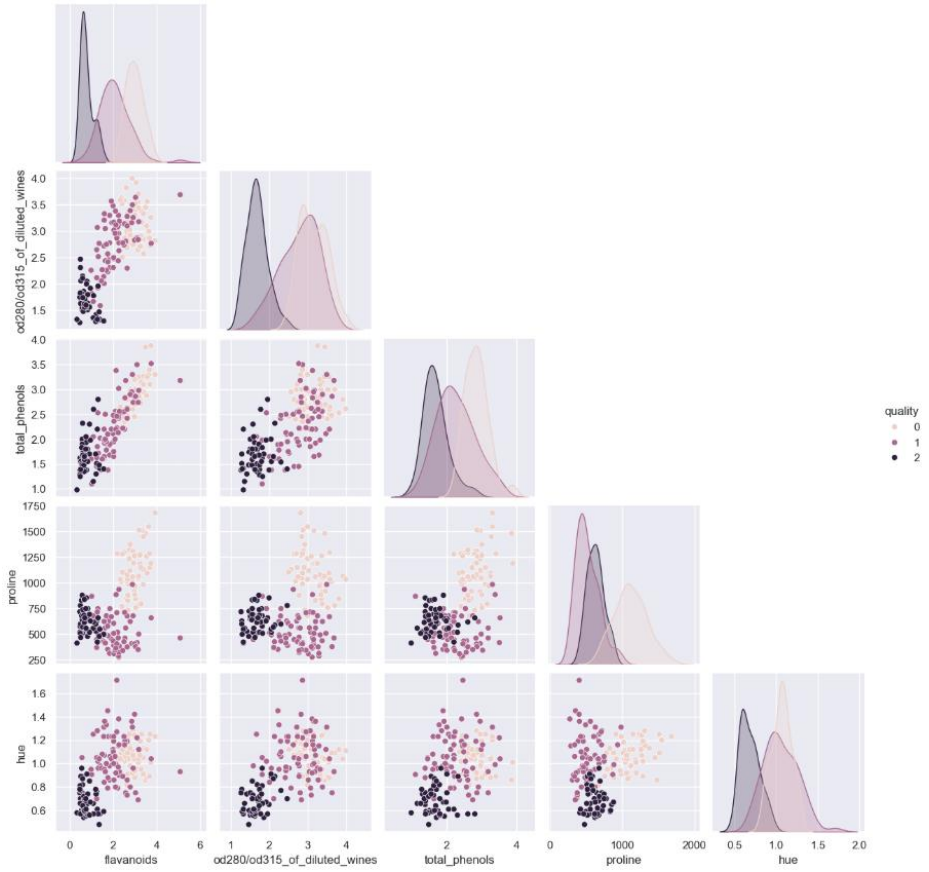


서울 18반 마이스터 트랙
이재윤

다변수 분석 - 페어 플롯(Pair Plot)

- 여러 변수들 간의 관계를 한 번에 보여주는 그래프
- 변수들 사이의 관계와 분포를 빠르게 훑어볼 때 좋은 도구
- 대각선(분포도)
  - 각 성분이 어떻게 분포되어있는 지를 보여줌
- 대각선 외(산점도)
  - 두 성분 사이의 관계를 보여주는 산점도(Scatter Plot)
- 장점
  - 짧은 코드로 모든 변수 간의 관계와 분포를 동시에 탐색할 수 있음
- 단점
  - 변수 개수가 많아지면 계산이 오래 걸림
  - 변수가 10개를 넘어가면 너무 복잡해짐

  => 중요하다고 생각되는 몇 개의 변수만 선택해서 그리는 것이 일반적



모델 학습

모델 학습

EDA를 통해 데이터 분석이 끝났으니, 모델에 데이터를 학습시켜 보자!

- 모델이 주어진 모든 데이터를 학습한다면, 새로운 데이터에 대해서 항상 좋은 결과를 응답할까?
- 예를 들어,
  - 모의고사 문제와 정답 심지어 오타까지 외운 학생이 과연 수능을 잘 볼 수 있을까?
  - 잘볼수있다!
  - 하지만 정말못풀수도있다!
  - 모의고사 문제에만 익숙해져서, 실전 수능에서는 문제풀이 능력이 현저하게 떨어질 수 있다.

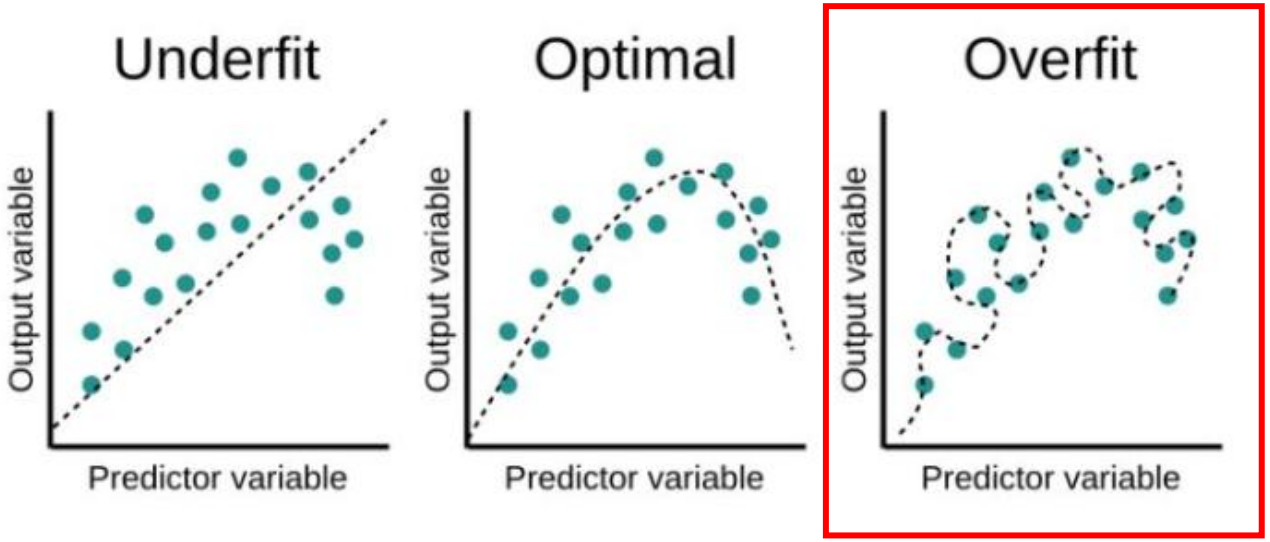
- 마찬가지로, 모델 또한 주어진 데이터에만 너무 '과하게' 최적화되어,
  훈련(기존 데이터)에서는 좋은 성능을 보이지만, 실제 테스트(새로운 데이터)에서는 성능이 나빠지는 현상

➢ 이것이 "**과적합(Overfitting)**"

모델 학습

과적합(Overfiting)

- 머신러닝 모델이 훈련 데이터(Training Data)에만 너무 익숙해져서, 데이터의 중요한 패턴을 넘어 사소한 잡음(noise)까지 전부 외워버리는 현상
- 이미 학습한 훈련 데이터에서는 좋은 성능을 보이지만, 테스트 데이터(새로운 데이터)에서는 낮은 성능을 보임
- 일반화를 하지 못하고, 훈련 데이터에 따라 예측이 크게 흔들림



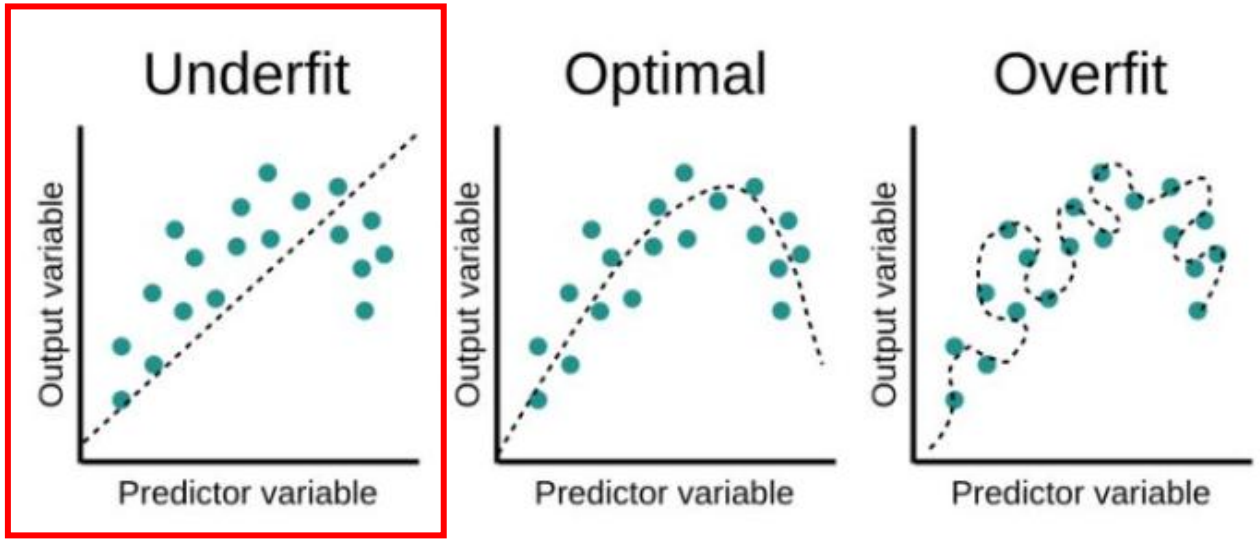
https://wikidocs.net/152777

서울 18반 마이스터 트랙
이재윤

과소적합(Underfiting)

- 모델이 너무 단순해서 훈련 데이터의 핵심적인 패턴조차 제대로 학습하지 못한 상태
- 모델의 성능은 훈련 데이터에서도 낮고, 테스트 데이터에서도 낮음

➢ 그러면, 모델이 과적합(Overfiting) 혹은 과소적합(Underfiting) 됐는 지는 어떻게 알까?

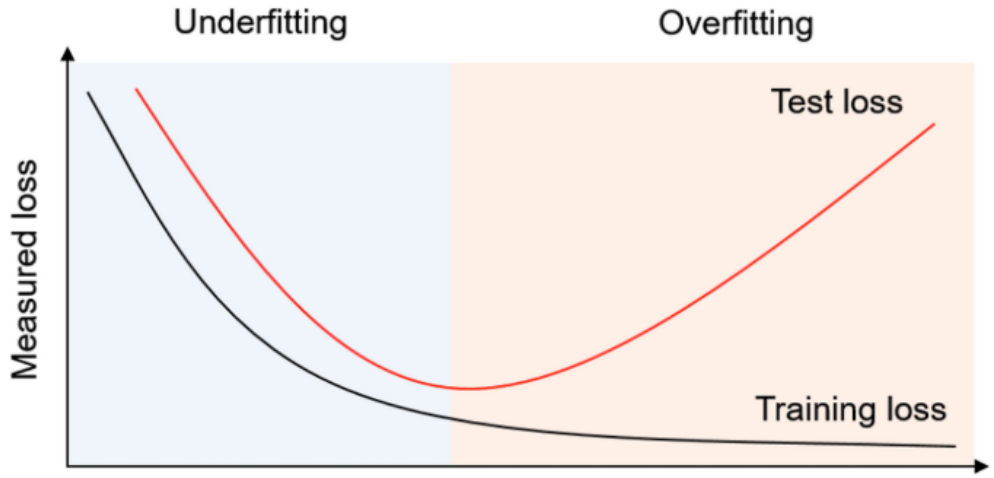


https://wikidocs.net/152777

모델 학습

데이터 분리

- 주어진 데이터을 훈련 데이터와 테스트 데이터로 분리
- 훈련 데이터: 모델을 학습시키는 용도로 활용
- 테스트 데이터: 모델이 얼마나 새로운 데이터를 잘 예측하는지 일반화 성능을 최종 평가하는 용도
  (학습된 모델이 과적합(Overfitting) 됐는 지 확인)
- 일반적으로 7:3, 8:2 (훈련:테스트)

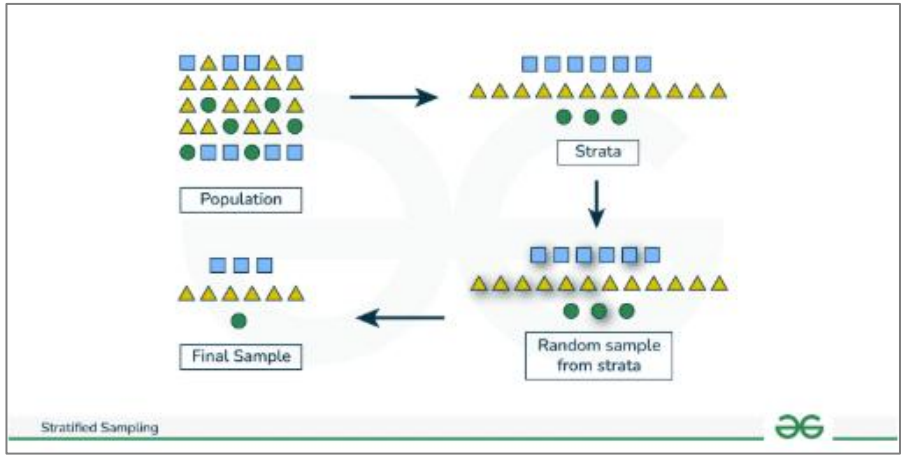


서울 18반 마이스터 트랙
이재윤

데이터 분리 - 코드

- train_test_split(): 주어진 데이터을 훈련 데이터와 테스트 데이터로 분리하는 함수
- test_size=0.3 : 전체 데이터 중 30%를 테스트용, 나머지 70%를 훈련용으로 나누겠다는 의미
- random_state=42: 이 숫자를 고정해놓으면 코드를 다시 시작해도 동일한 랜덤값이 배정 됨 (아무 값이나 가능)
- stratify=y
  - 원본 데이터의 정답 비율을 분할된 데이터셋에 동일한 비율로 나눠주는 옵션
  - 훈련 데이터에 '정상' 메일만 들어가고, 테스트 데이터에 '스팸' 메일만 들어가는 것을 방지할 수 있음
  - 분류 문제에서는 필수

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.3,
                                                    random_state=42,
                                                    stratify=y)
```



https://www.geeksforgeeks.org/machine-learning/how-to-implement-stratified-sampling-with-scikit-learn/

모델 학습

데이터 분리를 했으니 이제 진짜 학습 시작??

- 아니요
- 이제 데이터를 분리했을 뿐, 데이터 전처리를 아직 진행하지 않았음
- 결측치, 임베딩, 스케일링 작업이 필요

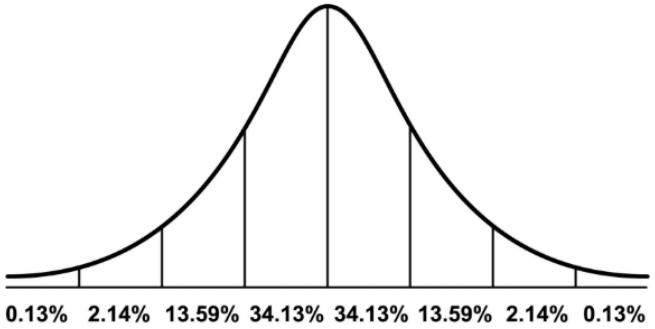
스케일링

- 각 특성의 단위를 맞춰서 모델이 공평하게 학습하도록 만드는 과정
- 마치 키(180cm)와 몸무게(70kg)를 비슷한 범위의 값으로 조정
- 대표적인 방법으로 표준화와 정규화가 있음

서울 18반 마이스터 트랙
이재윤

[복습] 스케일링 - **표준화**

- 모델에게 '나이' 와 '연봉' 데이터를 그대로 주면 어떻게 될까?
  - ➢ 값의 범위가 훨씬 큰 '연봉'을 더 중요한 정보로 착각할 수 있음
- 제각각인 데이터의 단위를 공평하게 맞춰주는 작업
- 데이터의 평균을 0, 표준편차를 1로 변환
  - 평균 0 : 모든 데이터를 0을 기준으로 맞춤
  - 표준편차 1 : 각 데이터가 평균으로부터 얼마나 떨어져 있는지(표준편차)를 기준으로 데이터의 스케일을 조절
- 특징
  - 이상치에 상대적으로 덜 민감하고, 정규분포를 따를 때 효과적
    (*__이상치__: 다른 값들에 비해 극단적으로 다른 값)
  - 모든 특성에 공평한 영향력을 부여
  - 경사 하강법, SVM, 거리 기반 알고리즘의 성능 향상



서울 18반 마이스터 트랙
이재윤

스케일링 - 표준화 - 코드

- 훈련 데이터의 표준화를 위한 평균과 표준편차를 구하고, 데이터 표준화 진행

❖ 주의) 테스트 데이터의 표준화는 반드시 "훈련 데이터의 평균/표준편차"를 활용해야 함!!!!!

- 테스트 데이터는 아직 학습하지 않은 미래의 데이터
- 테스트 데이터의 평균/표준편차를 scaler 에 반영한다면, 정답을 미리 훔쳐본 것과 같아짐
- 이렇게 유출된 정보로 학습한 모델은 실제 현실에서 제대로 된 성능이 나오지 않음

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()

# fit: 훈련 데이터의 평균과 표준편차 계산
# transform: 데이터를 표준화
scaler.fit(X_train)
X_train_norm = scaler.transform(X_train)

# or
# fit_transform: fit과 transform을 한 번에 수행
X_train_norm = scaler.fit_transform(X_train)

# 테스트 데이터는 훈련 데이터의 통계량(평균, 표준편차)을 사용하여 변환
X_test_norm = scaler.transform(X_test)
```

성능 평가

서울 18반 마이스터 트랙
이재윤

성능 평가

- 선형 회귀 모델은 MSE(Mean Squared Error, 평균 제곱 오차)를 활용해서 모델을 평가
- 분류 모델은 어떻게 평가할까?
  - 모델의 예측 값과 실제 정답을 비교해서, 모델이 얼마나 정답을 잘 맞혔는 지 "객관적인 숫자"로 확인하는 과정
  - 평가 지표
    - 정확도(Accuracy)
    - 정밀도(Precision)
    - 재현율(Recall)
    - 조화 평균(F1-Score)

혼동 행렬 활용
(Confusion Matrix)

예측(y_pred)	정답(y_test)	결과
1	1	O
0	0	O
0	1	X
1	1	O

서울 18반 마이스터 트랙
이재윤

분류 모델 성능 평가 - 정확도(Accuracy) (1/2)

- (맞은 예측 개수) / (전체 예측 개수)
- 가장 이해하기 쉽고, 직관적인 성능 평가 지표
- 예시) 스팸 메일 분류
  - 100개의 이메일 중에서 실제 스팸 메일 10개와 정상 메일 90개가 있다.
  - 모델의 예측 결과
    - 스팸 10개 중 8개를 스팸으로 맞춤
    - 정상 90개 중 85개를 정상으로 맞춤
  - 모델이 올바르게 예측한 총 개수는 93개이므로, 정확도는 93 / 100 = 93 %

```
from sklearn.metrics import accuracy_score

accuracy = accuracy_score(y_test, y_pred)
```

➢ 하지만, 정확도가 아무리 높아도 안심할 수 없음

서울 18반 마이스터 트랙
이재윤

분류 모델 성능 평가 - 정확도(Accuracy) (2/2)

- 하지만, 정확도가 아무리 높아도 안심할 수 없음
- 예) 암 진단
  - 100명의 환자 중 정상 99명, 암 환자 1명이 있다.
  - 모든 환자를 전부 '정상'이라고 예측하는 모델이라고 하면, 이 모델은 99명의 정상을 맞추고, 1명의 암 환자를 틀렸음.
  - 정확도는 99% 이지만, 정작 우리가 찾아야 할 1명의 암 환자를 놓쳤기 때문에
  - 이 모델은 아무 쓸모가 없음

- 이처럼 불균형 데이터에서는 정확도를 성능 평가 지표로 사용하기 어려움

서울 18반 마이스터 트랙
이재윤

분류 모델 성능 평가 - 혼동 행렬(Confusion Matrix) (1/4)

- 분류 모델의 성능을 평가할 때 사용하는 가장 기본적이면서도 강력한 도구
- 어떤 실수를 어떻게 저질렀는지를 한눈에 보여주는 상세한 지표
- Positive: 우리가 찾으려는 대상(ex: 암, 스팸)
- Negative: 그 외의 대상(ex: 정상)
- 구성 요소
  - True Positive(TP)
  - True Negative(TN)
  - False Positive(FP)
  - False Negative(FN)

		예측클래스	
		Positive	Negative
실제클래스	Positive	TP (True Positive)	FN (False Negative)
	Negative	FP (False Positive)	TN (True Negative)

https://leehah0908.tistory.com/19

서울 18반 마이스터 트랙
이재윤

분류 모델 성능 평가 - 혼동 행렬(Confusion Matrix) (1/4)

- True Positive(TP)
  - 실제 Positive인 것을 모델이 Positive 로 정확히 예측한 것
  - 암 환자(Positive)를 암(Positive) 이라고 올바르게 진단
  - 정답
- True Negative(TN)
  - 실제 Negative인 것을 모델이 Negative 로 정확히 예측한 것
  - 정상인(Negative)을 정상(Negative)이라고 올바르게 진단
  - 정답

		예측클래스	
		Positive	Negative
실제클래스	Positive	TP (True Positive)	FN (False Negative)
	Negative	FP (False Positive)	TN (True Negative)

서울 18반 마이스터 트랙
이재윤

분류 모델 성능 평가 - 혼동 행렬(Confusion Matrix) (1/4)

- False Positive(FP)
  - 실제 Negative 인 것을 모델이 Positive 로 잘못 예측한 것
  - 정상인(Negative)을 암(Positive) 이라고 잘 못 진단
  - 모델의 섣부른 판단
- False Negative(FN)
  - 실제 Positive 인 것을 모델이 Negative 로 잘못 예측한 것
  - 암 환자(Positive)을 정상(Negative)이라고 잘못 진단
  - 모델이 놓침

		예측클래스	
		Positive	Negative
실제클래스	Positive	TP (True Positive)	FN (False Negative)
	Negative	FP (False Positive)	TN (True Negative)

https://leehah0908.tistory.com/19

서울 18반 마이스터 트랙
이재윤

분류 모델 성능 평가 - 혼동 행렬(Confusion Matrix) (1/4)

- 100명의 환자를 진단하는 암 예측 모델이 있다고 가정 (실제 암 환자: 10명, 실제 정상인: 90명)
- 모델의 예측 결과
  - 실제 암 환자 10명 중 8명을 암으로 예측(쎄=8)
  - 실제 암 환자 10명 중 2명을 '정상'으로 놓침(FN=2)
  - 실제 정상인 90명 중 5명을 '암'으로 오진(FP=5)
  - 실제 정상인 90명 중 85명을 '정상'으로 예측(TN=85)

		예측 클래스	
		Positive	Negative
실제 클래스	Positive	TP (True Positive)	FN (False Negative)
	Negative	FP (False Positive)	TN (True Negative)

	예측: 암 (Positive)	예측: 정상 (Negative)	합계 (실제)
실제: 암	TP = 8	FN = 2	10
실제: 정상	FP = 5	TN = 85	90
합계 (예측)	13	87	100

https://leehah0908.tistory.com/19

서울 18반 마이스터 트랙
이재윤

분류 모델 성능 평가 - 정밀도(Precision)

- 모델이 “Positive”라고 예측한 것들 중에서 진짜 “Positive”한 것들의 비율
- 모델이 “암” 이라고 예측한 것들 중에서 진짜 “암”이었던 비율
- 모델의 예측이 얼마나 정교한지를 나타냄
- 정밀도 = TP / (TP + FP) = 8 / (8 + 5) = 61.5%

- 모델의 예측을 “진짜라고 믿을 수 있어야 할 때” 유용

		예측 클래스	
		Positive	Negative
실제 클래스	Positive	TP (True Positive)	FN (False Negative)
	Negative	FP (False Positive)	TN (True Negative)

	예측: 암 (Positive)	예측: 정상 (Negative)	합계 (실제)
실제: 암	TP = 8	FN = 2	10
실제: 정상	FP = 5	TN = 85	90
합계 (예측)	13	87	100

https://leehah0908.tistory.com/19

서울 18반 마이스터 트랙
이재윤

분류 모델 성능 평가 - 재현율(Recall)

- 실제 "Positive" 중에서 모델이 "Positive"라고 맞춘 비율
- 실제 "암" 환자들 중에서 모델이 "암"이라고 맞춘 비율
- 모델이 찾아야 할 것을 얼마나 빠짐없이 잘 찾아냈는지를 나타냄
- 재현율 = TP / (TP + FN) = 8 / (8 + 2) = 80%

- 모델이 "하나라도 놓치면 안 될 때" 유용

		예측 클래스	
		Positive	Negative
실제 클래스	Positive	TP (True Positive)	FN (False Negative)
	Negative	FP (False Positive)	TN (True Negative)

	예측: 암 (Positive)	예측: 정상 (Negative)	합계 (실제)
실제: 암	TP = 8	FN = 2	10
실제: 정상	FP = 5	TN = 85	90
합계 (예측)	13	87	100

https://leehah0908.tistory.com/19

서울 18반 마이스터 트랙
이재윤

분류 모델 성능 평가 - 조화 평균(F1-Score)

- 정밀도와 재현율을 모두 고려하고, 균형 잡힌 성능을 나타낼 때 사용
- 클래스 불균형이 심할 때, 정확도를 대체할 수 있는 훌륭한 성능 지표
- 일반적으로 정밀도를 높이면 재현율이 떨어지고, 재현율을 높이면 정밀도가 떨어지는 경향이 있음(Trade-off)
- 정밀도 = TP / (TP + FP) = 8 / (8 + 5) = 61.5%
- 재현율 = TP / (TP + FN) = 8 / (8 + 2) = 80%
- F1-score = 69.5%

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

	예측: 암 (Positive)	예측: 정상 (Negative)	합계 (실제)
실제: 암	TP = 8	FN = 2	10
실제: 정상	FP = 5	TN = 85	90
합계 (예측)	13	87	100

https://encord.com/glossary/f1-score-definition/

성능 검증

성능 검증

평가한 점수가 정말 신뢰성이 있을까?

- 데이터를 어떻게 나눴는지 / 하이퍼파라미터를 어떻게 설정했는 지에 따라 점수가 달라질 수 있음
  - 하이퍼 파라미터: 사람이 외부에서 직접 설정하는 값 (ex: 학습률)
- 어떤 방식이 좋은 성적을 받을 수 있는 지를 검증하자!
  - 학습률을 0.1로 설정하고 검증, 학습률을 0.2로 설정하고 검증, …
  - 제일 높은 성적을 받는 방식을 채택해서 성능 평가
- 검증을 테스트 데이터로 진행하면, “데이터 유출(Data Leakage)” 발생
- 검증 방식 종류
  - 검증 데이터(단순 분할 방식)
  - 교차 검증(Cross-Validation)

성능 검증

검증 데이터(Validation Data)

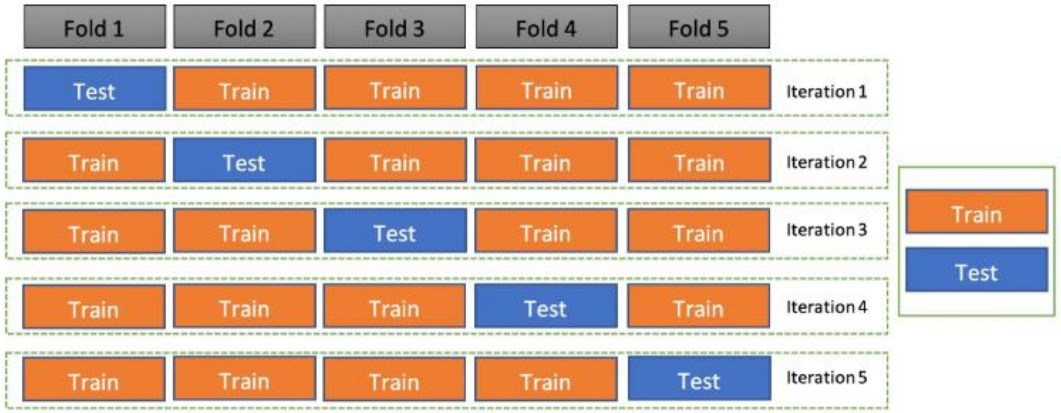
- 모델을 훈련시키는 과정에서, 어떤 훈련 방법이 가장 좋은지 확인하고, 최고의 모델을 선택하기 위해 사용되는 데이터
- 예를 들어,
  - **동일한 학습데이터**에 대해서 모델이 A 방법으로 학습하고, **동일한 검증 데이터**로 평가 => 85점
  - **동일한 학습데이터**에 대해서 모델이 B 방법으로 학습하고, **동일한 검증 데이터**로 평가 => 70점
  - **동일한 학습데이터**에 대해서 모델이 C 방법으로 학습하고, **동일한 검증 데이터**로 평가 => 95점
  - 모델은 가장 점수가 높은 "B 방법"으로 학습을 진행하고, 이후 테스트데이터로 평가 진행
- 즉, 최고의 학습 방법을 구분하기 위한 테스트데이터라고 생각하면 됨
- 검증 데이터는 학습 용도로써 활용되지 않으며, 학습 데이터에서 따로 분리
- 검증 데이터가 없이 테스트 데이터로만 성능을 자꾸 확인하면, 테스트 데이터에만 잘 맞는 모델이 만들어짐
  => 정보 유출(Data Leakage) (이후에 학습 예정)

성능 검증

교차 검증(Cross-Validation)

- 훈련 데이터를 여러 개(K개)의 덩어리(Fold)로 나눈 뒤, 각 Fold가 돌아가면서 한 번씩 '검증 데이터'역할을 하는 것
- 검증 데이터를 일회성으로 사용했을 때의 불안정성을 극복하기 위해, 여러 개의 검증 세트를 체계적으로 만들어 평가하는 방법
- 마치 수능 전에 여러 번의 모의고사를 보는 것과 같음

구분	검증 데이터 (단순 분할)	교차 검증 (K-Fold)
목적	동일 (하이퍼파라미터 튜닝, 중간 성능 측정)	동일
'연습 시험지'	고정된 1개	여러 개(K개)가 돌아가며 역할 수행
신뢰도	데이터 분할에 따라 결과가 좌우될 수 있음	더 안정적이고 신뢰도 높은 성능 추정 가능
비용	계산이 빠름	모델을 여러 번 학습시켜 시간이 더 걸림



https://velog.io/@pppanghyun/6.-%EA%B5%90%EC%B0%A8-%EA%B2%80%EC%A6%9DCross-Validation

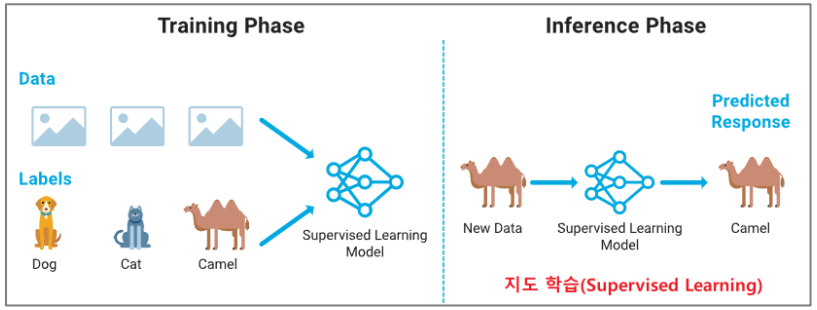
비지도 학습

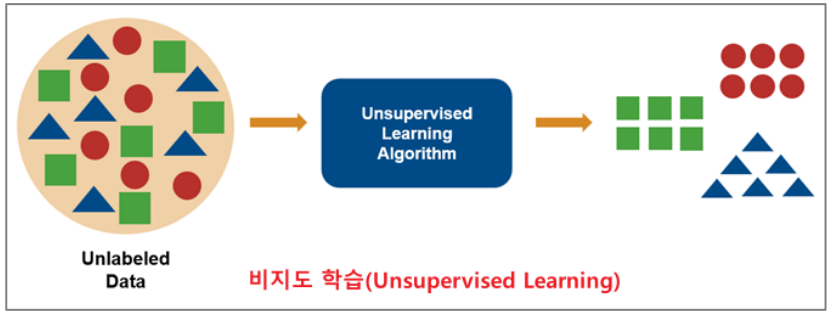
비지도 학습

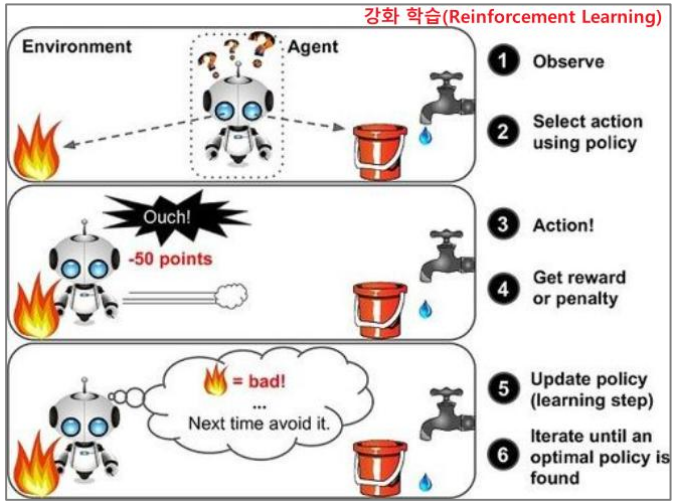
서울 18반 마이스터 트랙
이재윤

머신러닝

- 컴퓨터에게 데이터를 주고, 그 데이터 안에서 규칙(패턴)을 스스로 학습하게 하는 방식
- 학습 종류
  - 지도 학습(Supervised Learning)
    - 정답지가 있는 데이터를 학습시키는 방법
    - 의료 진단, 스팸 필터링, 주가/부동산 가격 예측 등에 활용
  - **비지도 학습(Unsupervised Learning)**
    - 정답이 없는 데이터의 숨겨진 구조나 규칙을 찾는 방법
    - 추천 시스템, 이상 거래 탐지, 고객 세분화 등에 활용
  - 강화 학습(reinforcement Learning)
    - '상' 과 '벌' 을 통해서 최적의 행동 방식을 학습하는 방법 (알파고)
    - 로봇 제어, 자율 주행, 게임 AI 등에 활용
- 일반적으로 비지도 -> 지도 -> 강화 학습으로 복합적인 학습으로 결과물 생성





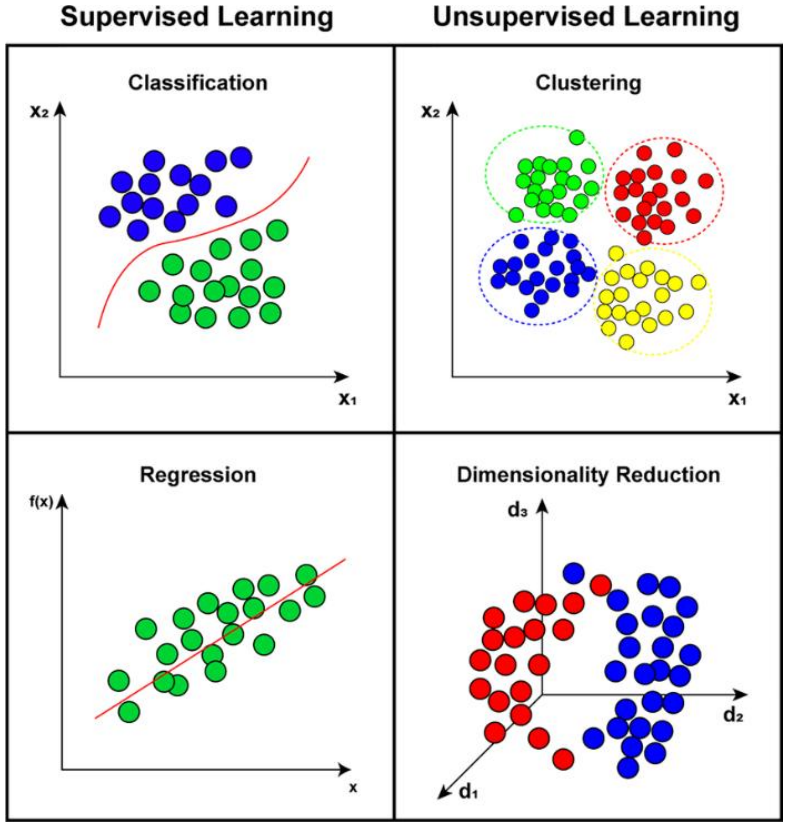


https://www.mathworks.com/discovery/supervised-learning.html?s_tid=srchtitle_site_search_1_supervised+learning
https://www.mathworks.com/discovery/unsupervised-learning.html https://marutitech.com/businesses-reinforcement-learning/

비지도 학습

비지도 학습(UnSupervised Learning)

- 정답이 없는 데이터의 숨겨진 구조나 규칙을 찾는 방법
- 수백 장의 동물 사진을 주고 "비슷한 것끼리 묶어 봐" 라고 하는 것
- 비지도 학습 유형
  - 군집화(Clustering)
    - 비슷한 데이터끼리 그룹으로 묶는 것
  - 차원 축소(Dimensionality Reduction)
    - 데이터의 복잡성(차원)을 줄여 핵심만 남기는 것
  - 연관(Association)
    - 데이터 안에서 특정 항목들이 얼마나 자주 함께 발생하는 지를 분석하여 규칙이나 관계를 찾아내는 기법



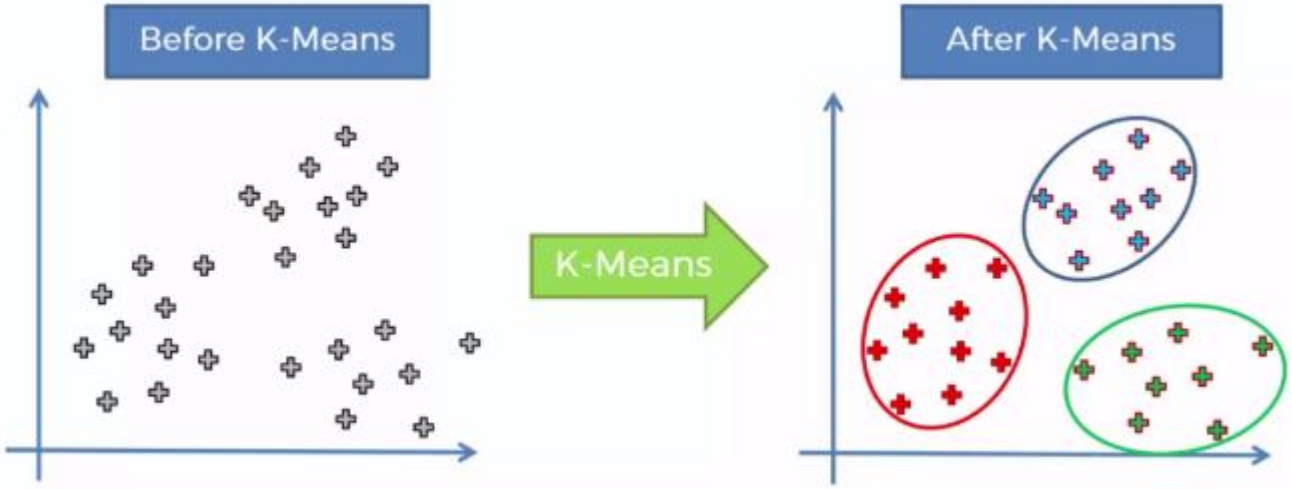
https://www.researchgate.net/figure/Representation-of-main-categories-on-machine-learning-state-of-art-divided-between_fig4_359454245

군집화(Clustering)

서울 18반 마이스터 트랙
이재윤

군집화(Clustering)

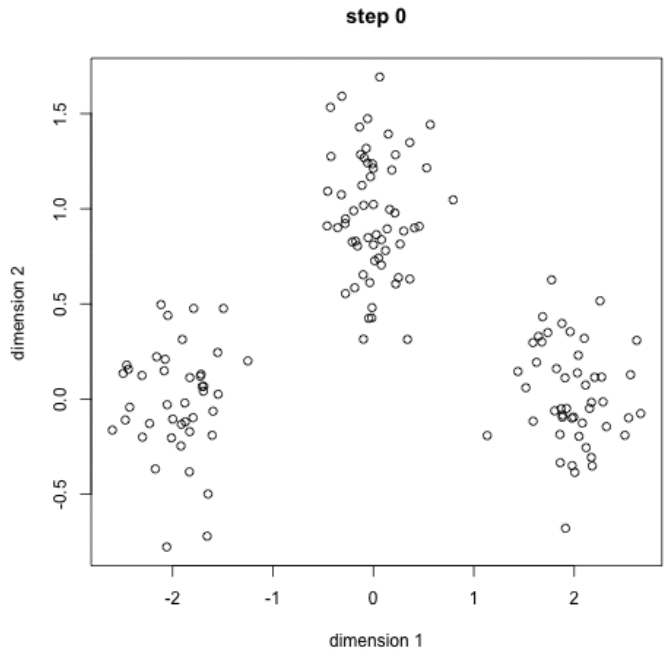
- 정답이 없는 데이터들을 유사한 특징을 가진 것끼리 하나의 그룹(Cluster)으로 묶는 작업
  - 예) 구매 패턴이 비슷한 고객 나누기, 이미지 분할, 비슷한 주제의 문서를 분류
- 정답이 없어도 되기 때문에 레이블링 작업이 불필요
- 정답이 없기 때문에 사람이 파악하지 못한 숨겨진 패턴을 찾을 수 있음
- 군집화 알고리즘
  - K-평균(K-Means)
  - 계층적 군집(Hierarchical)



서울 18반 마이스터 트랙
이재윤

K-Means (1/3)

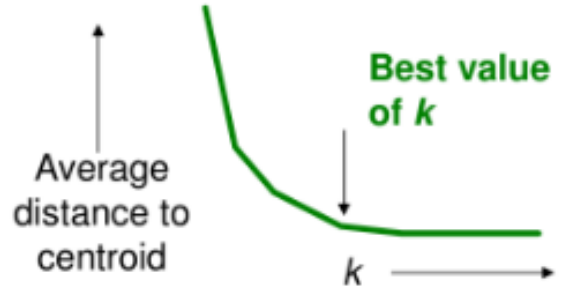
- 데이터들이 가장 가까운 중심점(Centroid)을 기준으로 뭉치도록 그룹을 형성하도록 하는 알고리즘
- 동작 방식
  1. K개의 중심점을 데이터 공간에 랜덤으로 배치
  2. 모든 데이터는 자신과 가장 가까운 중심점과 그룹을 형성 (K개의 그룹이 생성됨)
  3. 각 그룹의 평균을 계산하고, 그 지점을 다시 중심점으로 설정
  4. 새로 생성된 중심점을 기준으로 2~3번 과정을 "더 이상 변화가 없을 때까지" 반복



서울 18반 마이스터 트랙
이재윤

K-Means (2/3)

- 이상치에 민감하고, 초기 중심점의 위치에 따라 결과가 달라질 수 있음
- 군집의 개수(K)를 직접 지정해야 함
  - 너무 작게 잡으면, 다른 성격이 묶이고
  - 너무 크게 잡으면, 같은 그룹이 여러 개로 쪼개질 수 있음
- 엘보우 방법(Elbow Method)
  - 적절한 K 값을 선택하는 방법
  - K를 1부터 10까지 점차 늘려가면서 "**이너셔**"를 계산
    - **이너셔**
      - 각 데이터 포인트가 자신이 속한 클러스터의 중심점에서 얼마나 떨어져 있는지, 그 거리의 총합
      - 이너셔가 작을수록 데이터들이 잘 뭉쳐있음을 나타냄
  - 이너셔가 급격하게 감소하다가 완만해지는 지점이 "최적의 K"가 될 확률이 가장 높은 후보



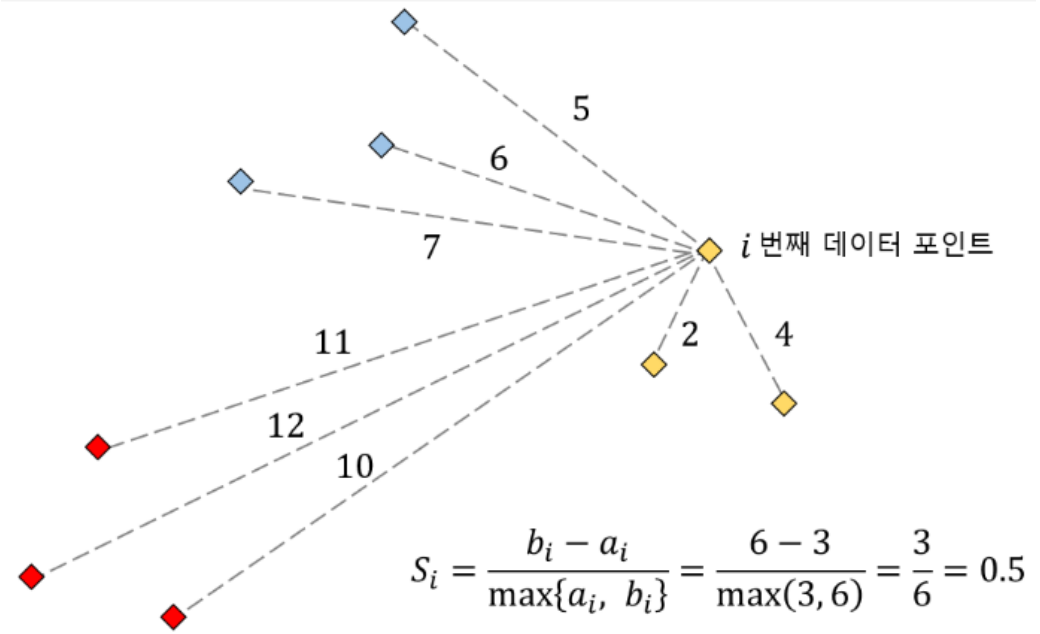
[엘보우 기법]

https://movefast.tistory.com/307

서울 18반 마이스터 트랙
이재윤

K-Means (3/3)

- 군집이 잘 뭉쳤는 지는 어떻게 평가할까?
- 실루엣 스코어(Silhouette Coefficient)
  - 각 데이터들이 다른 그룹과의 가까움에 따라서 점수를 매김

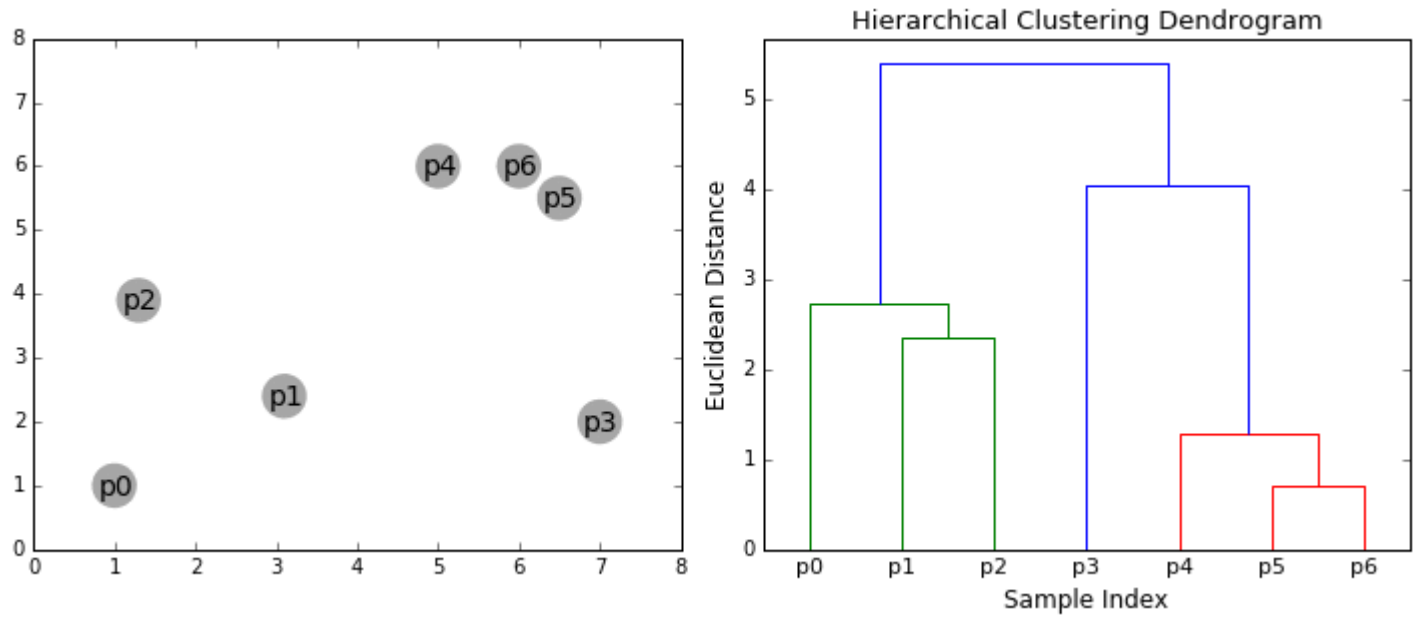


https://losskatsu.github.io/machine-learning/silhouette-score/#5-%EC%8B%A4%EB%A3%A8%EC%97%A3-%EC%8A%A4%EC%BD%94%EC%96%B4-%EA%B3%84%EC%82%B0-%EA%B3%BC%EC%A0%95

서울 18반 마이스터 트랙
이재윤

계층적 군집(Hierarchical Clustering)

- K-Means 와 달리 K 값을 미리 정하지 않아도 되는 군집
- 데이터 간의 거리를 바탕으로 가까운 것부터 순서대로 묶어 나가면서 나무 형태의 구조를 만들어가는 방식



https://dashee87.github.io/data%20science/general/Clustering-with-Scikit-with-GIFs/

차원 축소

차원 축소

데이터를 분석할 때, 데이터의 특징이 매우 많다면 모델이 더 똑똑해질까?

- No! "차원의 저주(Curse of Dimensionality)"
- 차원의 저주
  - 데이터의 차원(특징의 개수)이 증가할수록, 데이터를 분석하는 데 필요한 공간의 부피가 기하급수적으로 커져 발생하는 문제
  - 문제점
    - 불필요한 정보(노이즈)가 많아져 모델의 예측 성능이 떨어질 수 있음
    - 모델이 너무 복잡해져 과적합(Overfitting)이 될 가능성이 높아짐
    - 계산할 것이 너무 많아져 모델 학습에 많은 시간이 걸림
    - 3차원을 넘어서면서 시각화하기 어려움

➢ 데이터의 중요한 정보는 최대한 유지하면서, 불필요한 특징을 제거하자 => "차원 축소(Dimensionality Reduction)"

차원 축소

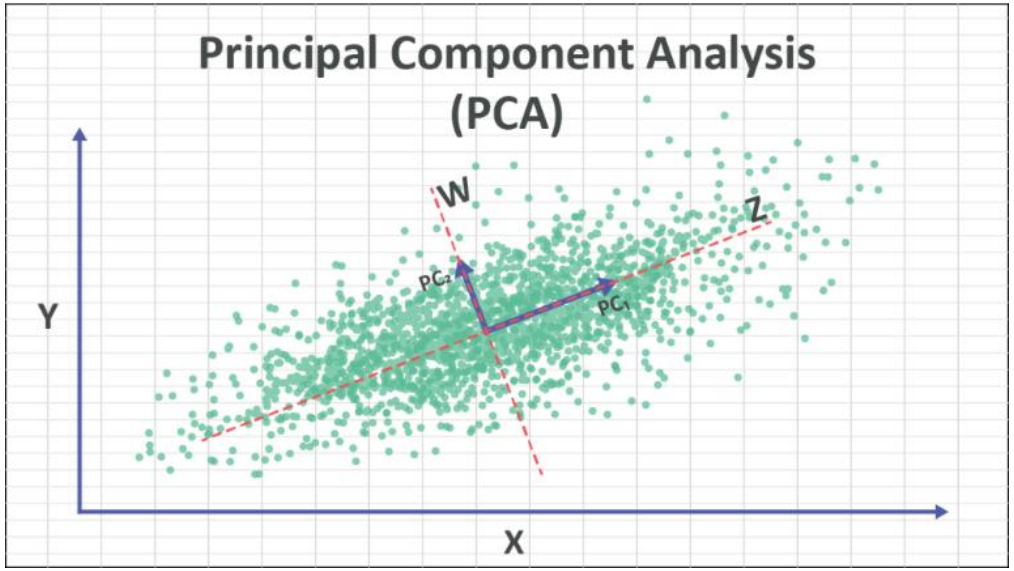
차원 축소(Dimensionality Reduction)

- 데이터의 중요한 정보는 최대한 유지하면서, 서로 관련이 높거나 불필요한 특징들을 제거하거나 압축하여 데이터의 차원을 줄이는 기술
- 두 가지 방법
  - 특성 선택
    - 여러 특징 중에서 가장 중요하다고 생각되는 몇 개의 특징만 골라내는 방식
    - ex) 100개 특징 중 상관관계가 높은 10개만 선택
  - 특성 추출
    - 기존 특징들을 새로운 축으로 투영하여, 더 적은 수의 새로운 특징으로 압축하는 방식
    - PCA(주성분 분석)

차원 축소

PCA(Principal Component Analysis)

- 데이터가 가장 넓게 퍼져있는(분산이 가장 큰) 방향으로 새로운 축을 찾는 기술
- 새로운 축이 바로 데이터의 정보를 가장 잘 설명하는 제1 주성분(Principal Component)
- 예를 들어) 럭비공을 2D 로 표현한다면 ?
  - 럭비공이 가장 길어 보이는 방향에서 찍어야 함
  - 주성분은 이런 데이터를 정보를 가장 많이 포함하는 첫 번째 축
- 주성분과 직각을 이루는 축이 제2 주성분

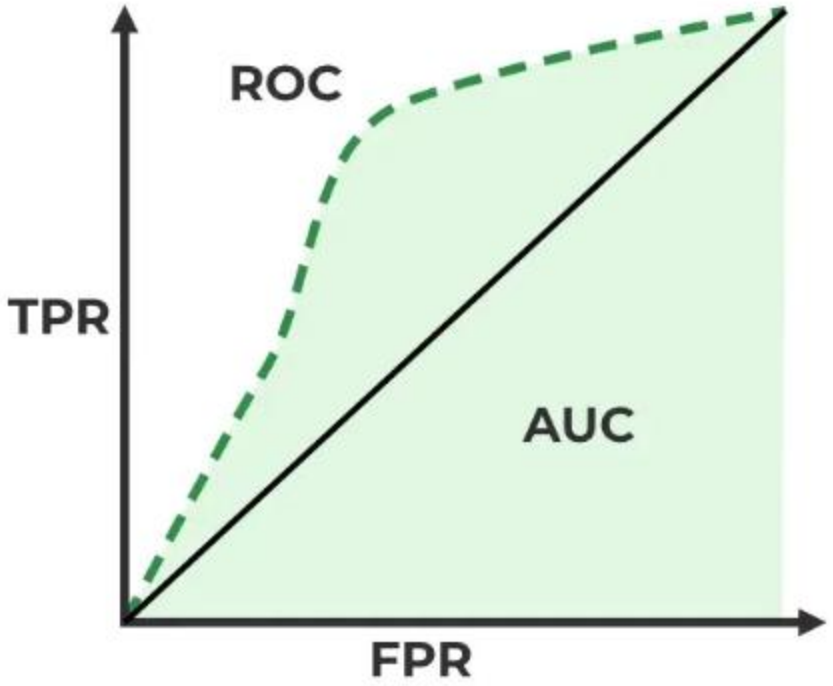


참고

서울 18반 마이스터 트랙
이재윤

[심화] ROC Curve

- 다양한 임계값 변화에 따라 성능이 어떻게 달라지는지를 시각적으로 보여주는 그래프
  - 임계값: 분류의 기준이며, 예를 들면 0.6을 넘을 경우 Positive, 0.8을 넘을 경우 Positive를 설정하는 값
- FPR
  - 실제 정상(Negative)인 데이터 중에서 모델이 "암(Positive)"라고 잘못 예측한 비율
  - 멀쩡한 사람을 암 환자로 오진할 확률 (0에 가까울수록 좋음)
- TPR
  - 실제 "암"인 데이터 중에서 모델이 "암"이라고 정확히 예측한 비율
  - 실제 암 환자를 암 환자로 정확히 찾아낼 비율(1에 가까울수록 좋음)
- AUC
  - ROC Curve 그래프의 아래쪽 면적
  - 1에 가까울수록 성능이 완벽에 가까우며, 0.5에 가까울수록 랜덤 추측의 수준을 의미
  - 일반적으로 AUC 값이 0.8 이상이면 좋은 모델



https://www.geeksforgeeks.org/machine-learning/auc-roc-curve/

다음 시간에
만나요!