



준지도학습(SSL)



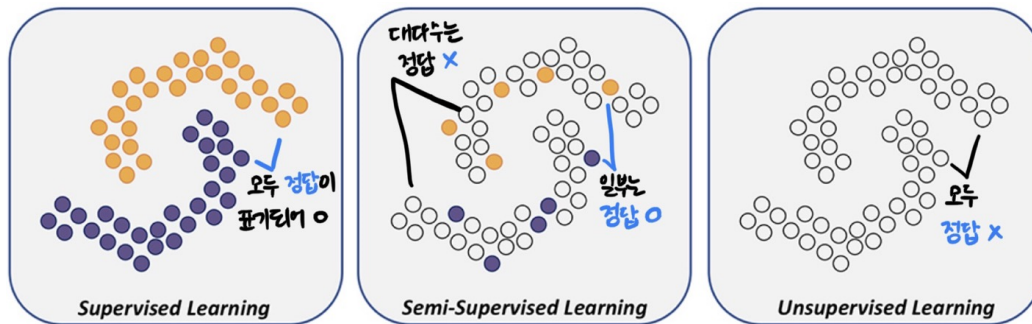
Reference

- [사이킷런 User Guide](#)
- [Semi-supervised learning 방법론 소개](#) ← 개괄적인 내용
- [A Beginner's Guide to Semi-Supervised Learning](#) ← 유도 학습
- [그래프 기반 준지도학습](#) ← 변환 학습
- ChatGPT(사랑해♥)

0. 개요

0-1. 머신러닝 방법론

- 크게 **지도학습**(supervised learning)과 **비지도학습**(unsupervised learning)으로 나뉨
 - 구분 기준) 학습 데이터의 라벨(label)의 유무
 - **지도학습**
 - 정답이 되는 데이터(labeled data)를 학습시켜 새로운 입력(input) 데이터가 무엇인지 예측하는 것을 목표로 함
 - 분류(classification), 회귀(regression)
 - **비지도학습**
 - 정해진 정답 없이 주어진 입력 데이터 간의 숨어있는 구조를 찾아 비슷한 데이터끼리 묶는 것을 목표로 함
 - 군집화(clustering)
- 준지도학습은 지도학습과 비지도학습의 방법론을 **모두** 활용하는 학습 방법
 - 레이블이 있는 샘플을 사용하여 모델을 지도(supervise)하는 동시에 레이블이 없는 샘플을 사용하여 데이터의 구조나 특성을 학습
 - 학습 정확도를 크게 향상시키기 위해 학습 시 소량의 레이블이 지정된(labeled) 데이터와 대량의 레이블이 지정되지 않은(unlabeled) 데이터를 결합하는 기계 학습의 한 분야



0-2. 준지도학습(semi-supervised learning)

- train 데이터에서 일부 샘플에 레이블이 지정되지 **않은** 상황
- `sklearn.semi_supervised` 의 준지도학습 estimators는 레이블이 지정되지 않은 데이터를 추가적으로 활용하여 기존 데이터 분포의 모양을 더 잘 파악하고 새로운 샘플로 더 잘 일반화할 수 있음
- 레이블이 지정된 데이터의 양이 매우 **적고** 레이블이 지정되지 않은 데이터의 양이 **많을** 때 활용할 수 있는 학습 방법
 - 학습 데이터에 수동으로 레이블을 지정(데이터 레이블링)하는 데 드는 비용을 줄이고 오늘날의 디지털 세계에서 찾을 수 있는 레이블이 지정되지 않은 방대한 양의 데이터를 활용할 수 있다는 부분에서 유리
- y의 레이블이 지정되지 **않은** 항목
 - `fit()` 을 사용하여 모델을 학습할 때 레이블이 지정되지 않은 곳에 레이블이 지정된 데이터와 함께 식별자를 할당하는 것이 중요
 - 사용하는 식별자: **-1(integer)**
 - 문자열(string) 레이블의 경우 문자열과 정수를 모두 포함할 수 있도록 y의 dtype이 `object` 여야 함
- 데이터 세트의 **분포**에 대한 가정이 필요
 - 레이블이 지정되지 않은 데이터를 사용하려면 데이터의 기본 분포와 어떤 관계가 있어야 함
 - 준지도학습 알고리즘은 아래의 가정 중 적어도 하나를 활용

0-3. 준지도학습의 기본 가정

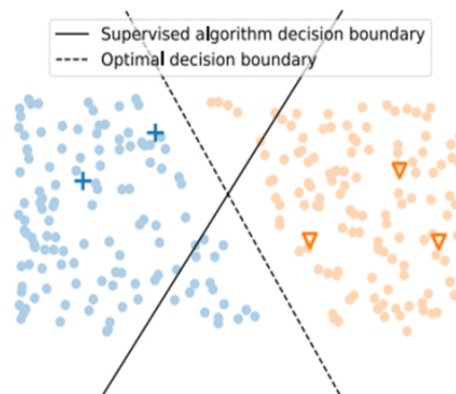
▼ a) 연속성/평활 가정(Continuity / Smoothness assumption)

- x와 x'가 가까우면, y와 y'도 충분히 가까워야 한다.
 - 입력 데이터가 서로 비슷하면 출력 레이블 또한 서로 비슷함

- x_1 이 x_2 와 가깝고, x_2 와 x_3 가 가깝다면 x_1 이 x_3 와 가깝지 않더라도 x_3 의 레이블이 x_1 과 같을 것이라고 기대(x_1 : labeled/ x_2, x_3 : unlabeled)
- 일반적으로 지도학습에서도 가정되며 기하학적으로 간단한 의사 결정 경계에 대한 선호를 산출
- 준지도학습의 경우, 평활 가정은 저밀도 영역의 결정 경계에 대한 선호도를 추가적으로 산출하기에 서로 가깝지만 다른 클래스에 속하는 소수의 포인트가 존재

▼ b) 저차원 가정(Low-density assumption)

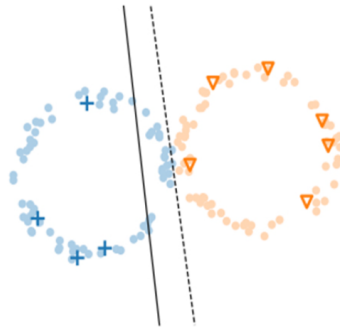
- 모델의 결정 경계가 데이터의 확률 밀도가 높은 곳을 지나지 않는다.
- 표본 공간에서 밀도가 **높은** 영역은 일반적으로 특정 클래스에 속함
 - 두 클래스를 구분하는 결정 경계는 데이터가 거의 **없는** 곳에 존재해야 함
 - 데이터가 많은 지역에 결정 경계를 놓는다면 가까운 데이터는 같은 레이블을 가진다는 smoothness 가정을 위반



▼ c) 매니폴드 가정(Manifold assumption)

- 고차원의 입력 데이터가 저차원 공간에서 **특정한 구조**(manifold)를 따라 놓여 있다.
 - 동일한 저차원의 매니폴드에서 주성분 분석(PCA)과 같은 방법을 사용하여 데이터 포인트를 저차원 공간으로 투영하는 것을 의미 → 동일한 레이블이 지정됨
 - 이 경우 레이블이 지정된 데이터와 레이블이 지정되지 않은 데이터를 모두 사용하여 매니폴드를 학습하면 차원의 저주(curse of dimensionality)를 피할 수 있음
 - 이후 매니폴드에 정의된 **거리**와 **밀도**를 사용하여 학습을 진행할 수 있음
- 직접 모형화하기엔 힘들지만 자유도(degree of freedom)가 몇개밖에 없는 일부 공정에서 고차원 데이터가 생성될 때 실용적

- 예시) 사람의 목소리는 몇 개의 목소리 주름에 의해 조절되고, 다양한 얼굴 표정의 이미지는 몇 개의 근육에 의해 조절됨
- 해당 경우 가능한 모든 음향파 또는 이미지의 공간이 아닌 생성 문제의 자연 공간(특정 공간)에서 거리와 부드러움을 각각 고려하는 것이 권장됨



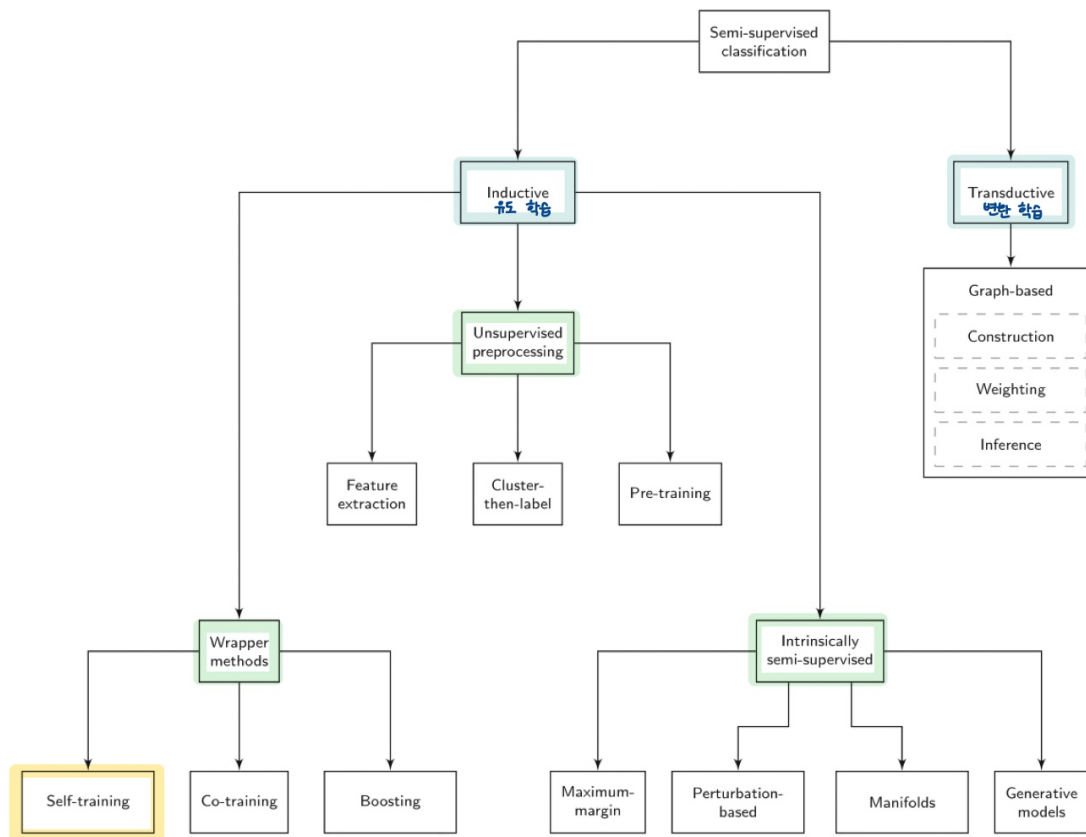
▼ +) 군집 가정(Cluster assumption)

- 데이터들이 같은 클러스터에 속하면 해당 데이터들은 같은 클래스에 속한다.
- 위의 세 가지 가정을 모두 **일반화**한 가정
 - 입력 공간 상에서 가까운 것들을 클러스터로 보는 경우: smoothness assumption
 - 확률 밀도가 높은 지역의 데이터 포인트를 클러스터로 보는 경우: low-density assumption
 - 저차원 manifold 상의 데이터 포인트를 클러스터로 보는 경우: manifold assumption
- 데이터는 **이산형(discrete)** 군집을 형성하는 경향이 있으며, 동일한 군집에 있는 점들은 다른 군집에 있는 점들에 비해 레이블을 공유할 가능성이 더 높음
 - 라벨을 공유하는 데이터는 여러 군집에 분산될 수 있음
 - 데이터 포인트(라벨링된 것과 레이블링되지 않은 것 모두)가 의미 있게 군집화될 수 없는 경우, 준지도 학습 방법이 지도 학습 방법에 비해 성능이 개선되는 것은 불가능하다는 것을 시사

1. 준지도학습 방법론



Scikit-Learn에서 제공되는 기능 위주로 정리



- 먼저 모델의 학습 원리에 따라 크게 **유도 학습**과 **변환 학습**으로 나눌 수 있음
- **유도 학습(Inductive Learning)**
 - 일반화된 함수를 학습하여 전체 입력 공간에 적용
 - 특정(훈련) 사례를 통해 학습한 뒤 전체 입력 공간에 대한 조건으로 일반화
 - 이용 가능한 훈련 데이터 세트의 도움을 받아 일반화된 기능을 학습하려고 노력
 - 일반화된 함수는 논리적으로는 참일 수 있지만, 샘플 공간의 모든 데이터 포인트에 대해 실질적으로 참인지 아닌지는 확신할 수 없음
- **변환 학습(Transductive Learning)**
 - 우리가 정말로 필요로 하는 구체적인 답을 얻는 데 초점을 두는 학습 방법
 - ex) 유사한 데이터 포인트 사이의 연결을 통해 그래프를 형성
 - 특정 훈련 예제에서 규칙을 생성한 다음 테스트 예제에 적용하는 것
 - 완전한 도메인 기반 학습 → 입력 샘플이 다른 경우에는 잘 작동하지 않음
 - 학습(train) 및 테스트(test) 단계가 필요하지 않으며 전체 입력 공간에 대한 분류기(classifier)를 구축하지 x

- 특히 작은 레이블이 있는 데이터셋과 대량의 레이블이 없는 데이터셋이 주어진 경우에 유용

→ 모든 레이블이 없는 데이터에 대한 예측을 수행하는 것이 아닌 주로 특정 관심 데이터에

대한 예측을 수행하는 방식으로 작동

✓ Inductive & Transductive Learning 차이점

1-1. 유도 학습

- 훈련(train) 단계에서 모델을 구축하고, 이후 새로운 데이터 포인트의 레이블(label)을 예측(predict)

- 예제

- 데이콘 코드 공유_Self-Training에 대한 간단한 예시

→ 주석에 용어가 조금 모호한 부분들이 존재하는 것 같음

(코드 리뷰 같이 해보면 좋을 듯함)

a) Wrapper Method

- 레이블이 지정된 데이터로 모델을 훈련하고, 그 훈련된 모델을 반복적으로 사용하여 레이블이 지정되지 않은 데이터에 대한 의사 레이블(가짜 레이블 / pseudo label)을 생성하는 방법

→ 의사 레이블이 생성된 데이터를 포함한 데이터셋으로 모델을 다시 훈련하며, 모든 레이블이 지정되지 않은 데이터에 대해 의사 레이블이 생성될 때까지 해당 과정을 반복

- 레이블링 방법에 따라 3가지로 구분됨

▼ 자체 훈련(self-training)

- 단일 분류기 학습기를 사용하는 래퍼 메서드
- 수행 단계
 1. 레이블이 지정(labeled)된 데이터로 분류기(classifier)를 훈련
 2. 훈련된 모델로 레이블이 지정되지 않은(unlabeled) 데이터를 분류
 3. 확실하게 분류된 의사 레이블(pseudo-labels)을 레이블이 지정된 데이터와 함께 사용하여 분류기를 다시 훈련
 4. 레이블이 지정되지 않은 데이터가 없을 때까지 단계 2와 3을 반복
- 전체 데이터셋을 다시 훈련하는 것을 피하기 위해 분류기를 점진적으로 학습시킬 수 있음

- 개별 데이터 포인트를 대상으로 목적 함수를 최적화
- 반복적인 의사 레이블링 접근법으로도 알려져 있음
- 래퍼 메서드 패러다임에서 약간 벗어나는 방식

- `sklearn.semi_supervised.SelfTrainingClassifier` 로 구현 가능
- Self Training은 추가 레이블링 과정에서 오답을 누적할 수 있음을 주의해야 함
 - 결과를 평가할 때에는 추가 레이블링된 데이터가 포함된 전체 데이터에 대한 평가를 진행하는 것이 좋음

▼ [사이킷런 User Guide\(밑에 번역해 둬\)](#)

- 야로프스키의 알고리즘을 기반으로 함
 - 주어진 지도 학습 분류기가 준지도 학습 분류기 역할을 할 수 있어 레이블이 지정되지 않은 데이터에 대해서도 학습할 수 있음
- `SelfTrainingClassifier` 는 `predict_proba` 를 구현하는 모든 분류기 (classifier)와 함께 호출할 수 있으며, 매개 변수 `base_classifier` 로 전달
 - 각 반복(iteration)에서 `base_classifier` 는 레이블이 지정되지 않은 샘플의 레이블을 예측하고 레이블이 지정된 데이터 세트에 이러한 레이블의 하위 집합을 추가하는 작업을 수행
- 하위 집합의 선택은 선택 기준에 따라 결정됨
 - 예측 확률에 대한 `threshold` 를 사용하거나 예측 확률에 따라 `k_best` 표본을 선택하여 수행할 수 있음
- 각 표본에 레이블이 지정된 iteration 뿐만 아니라 최종 적합(fit)에 사용되는 레이블 또한 속성으로 사용할 수 있음
 - `max_iter` 파라미터
 - 루프(반복)가 최대로 실행되는 횟수를 지정
 - `None` 으로 설정하면 모든 샘플에 라벨이 있거나 해당 반복에서 새 샘플이 선택되지 않을 때까지 알고리즘이 반복됨

⚠ Self-Training 분류기를 사용하는 경우 **분류기 보정(calibration)**이 중요

▼ 예시 코드

```
from sklearn.datasets import make_classification
from sklearn.linear_model import LogisticRegression
from sklearn.semi_supervised import SelfTrainingClassifier

### 데이터 생성
```

```

# 레이블이 있는 샘플 생성
X_labeled, y_labeled = make_classification(n_samples=100, n_features=10,
                                          n_informative=5, n_classes=2,
                                          random_state=42)

# 레이블이 없는 샘플 생성
X_unlabeled, _ = make_classification(n_samples=900, n_features=10,
                                    n_informative=5, n_classes=2,
                                    random_state=42)

# 테스트 데이터 생성
X_test, y_test = make_classification(n_samples=200, n_features=10,
                                    n_informative=5, n_classes=2,
                                    random_state=42)

#### Self Training
# 기본 분류기 생성
base_classifier = LogisticRegression()
# Self Training 분류기 생성
self_training_model = SelfTrainingClassifier(base_classifier)

# 초기 모델 학습
self_training_model.fit(X_labeled, y_labeled)
# 모델 업데이트
self_training_model.fit(X_unlabeled)

# 학습된 모델을 사용하여 새로운 데이터의 레이블을 예측
y_pred = self_training_model.predict(X_test)

```

▼ 공동 훈련(co-training)

- 여러 개의 지도학습(supervised learning) 분류기(classifier)에 대한 self-training의 확장
- 두 개의 분류기를 함께 훈련
 - 한 분류기에서 가장 확실하게 예측된(predicted) 레이블을 다른 분류기의 의사 레이블(pseudo-label) 데이터로 활용
- 다중 뷰 공동 훈련(Multi-view co-training)
 - 훈련 데이터를 완전히 다른 시각에서 분류기를 훈련하는 것을 의미
 - ex) Blum 등은 1998년에 대학 웹 페이지의 분류를 위해 "웹 페이지 텍스트"와 "웹 페이지로의 링크에서의 앵커 텍스트"를 외부 소스로부터 사용하여 두 개의 분류기를 구성하는 것을 제안
- 단일 뷰 공동 훈련(Single-view co-training)
 - 일반적으로 앙상블 방법으로 적용
 - ex) 준지도 학습에서의 RandomForest는 동일한 레이블이 지정된 데이터로 여러 개의 트리를 훈련한 다음, 다른 트리의 모든 공동 예측(joint

prediction)의 의사 레이블을 나머지 트리의 레이블된 데이터 포인트로 사용하는 방식으로 적용

▼ 부스팅(boosting)

- 탐욕적인(greedy) 방식으로 가중(weighted) 앙상블 분류기를 구성
- 각 기저 학습기(base learner)는 이전 학습기(또는 이전의 출력)에 영향을 받으며, 오분류 된 데이터 포인트에 더 큰 가중치가 할당됨
- 최종 예측은 기저 분류기의 예측들의 선형 결합(linear combination)으로 얻어짐
- 의사 레이블링 된(pseudo-labeled) 데이터를 각 학습 단계 이후에 도입함으로써 준지도 학습 설정으로 쉽게 확장될 수 있음
 - self-training과 co-training의 의사 레이블링 접근법은 부스팅 방법으로 쉽게 확장시킬 수 있음
 - **SSMBoost**, **ASSEMBLE**, **SemiBoost**, **RegBoost** 등 여러 부스팅 방법이 존재하며, 레이블이 지정되지 않은 데이터셋을 지도학습 분류기에 학습시키는 것을 가능하게 해 줌

b) 비지도 전처리 방법(Unsupervised Preprocessing Method)

- 생략

c) 내재적 준지도 학습 방법(Intrinsically Semi-supervised Method)



- 생략

1-2. 변환 학습



원본 글 중 필요한 부분만 요약(세부적인 원리 등은 생략)

- **그래프 기반(Graph-based)** 준지도 학습
 - 기본 가정: 그래프가 주어지고 해당 그래프는 레이블이 있는(labeled) 데이터와 없는 데이터(unlabeled)를 모두 포함하고 있음
 - 데이터는 **node**로, 유사성(유사도 가중치, Similarity Weight)은 **edge**로 표현
 - 가정) 강한 간선(→ edge)으로 연결되어 있으면 해당 노드(→ 데이터)들은 같은 레이블을 가질 것이다.

 <p>not similar</p>	 <p>'indirectly' similar with stepping stones</p>
--	---

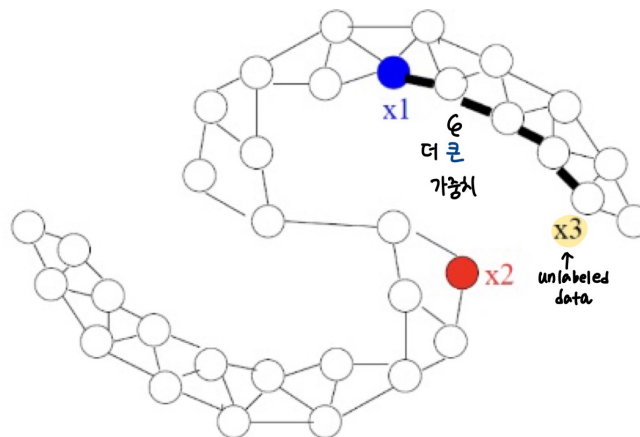
- 상대적인 유사도 계산을 위해 두 가지를 활용함

- **K-Nearest Neighbor 그래프**

- k개까지 둘 사이의 거리를 0에서 1 사이로 표시
- 객체 기준 그래프 → 밀도가 고려되지 못함

- **epsilon 반경 그래프**

- 특정 노드 기준으로 일정 cut-off 이상의 노드들을 연결
- 모든 객체가 연결되지 않아도 ok ⇒ 임의의 i와 j가 연결된다는 보장이 x
→ epsilon 그래프를 만든 후 레이블이 propagation 될 수 있도록 해줌



⇒ x3은 x1과의 가중치가 크기에 블루 레이블을 가질 가능성이 더 크다.

- 수행 단계

1. 초기 단계

- 레이블이 있는 데이터셋을 사용하여 초기 모델을 훈련
- 이 모델은 주어진 데이터에 대한 예측을 수행할 수 있음

2. 예측 단계

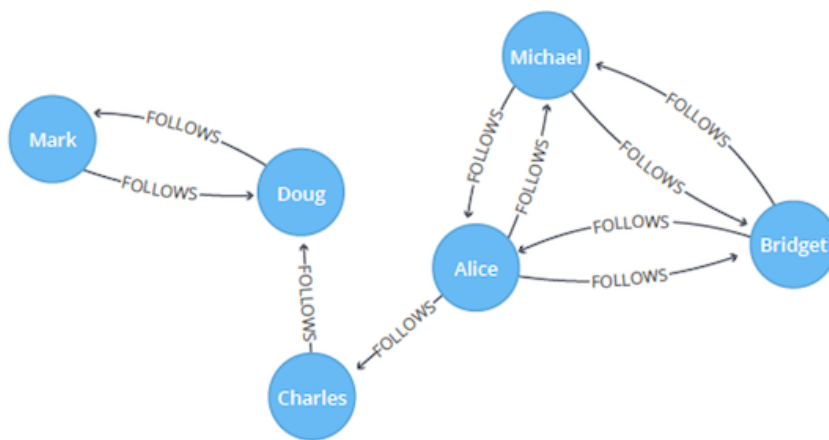
- 레이블이 없는 데이터에 대한 예측을 수행

- 예측은 초기 모델을 사용하여 이루어지며, 주로 주어진 데이터 포인트의 속성과 주변 데이터 포인트의 속성을 비교하여 예측을 수행

3. 업데이트 단계

- 예측된 레이블을 사용하여 모델을 업데이트
 - 레이블이 없는 데이터에 대한 예측이 개선될 수 있음
- 주로 예측된 레이블을 새로운 데이터셋에 추가하여 레이블이 있는 데이터셋을 확장하고, 모델을 재훈련하는 방식으로 업데이트를 수행

a) 라벨 전파(Label Propagation)



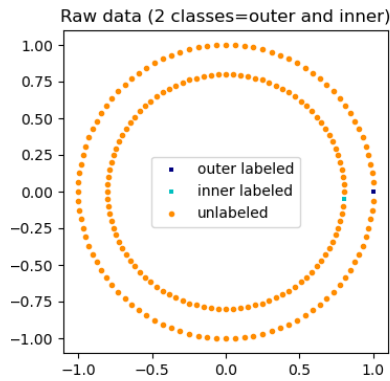
Graph Model

트위터 팔로우를 그래프로 표현한 것

- 레이블이 있는 샘플과 레이블이 없는 샘플을 합쳐서 그래프의 노드로 표현하고, 노드 간의 연결을 통해 정보를 전파하는 방식
- 레이블이 있는 샘플을 기반으로 그래프를 구성한 후, 레이블이 없는 샘플의 레이블을 주변 노드의 레이블을 참고하여 추정
 - 이 과정을 반복하여 모든 샘플에 대한 레이블을 추정하고, 모델을 훈련하는 방법

레이블이 지정되지 않은 관측치의 구조는 클래스 구조와 일치

→ 클래스 레이블은 레이블이 지정되지 않은 관측치의 훈련(train) 세트로 전파될 수 있음



- 해당 모델에서 가능한 작업들
 - 분류 작업
 - 대체(alternative) 차원 공간에 데이터를 투영하는 **커널 방법(kernel method)**
- **사이킷런이 제공하는 라벨 전파 모델**
 - **LabelPropagation**
 - **LabelSpreading**

⇒ 둘 다 입력 데이터 세트의 모든 항목에 대한 **유사성 그래프(similarity graph)**를 구성하며 작동

▼ 차이점

1) 레이블 분포에 대한 클램핑 효과에 대한 수정 사항

- 클램핑을 활용하면 알고리즘에서 실제 ground labeled data(우리가 정한 정답, 참값)가 지정된 데이터의 가중치를 어느 정도 변경할 수 있음
- **LabelPropagation** 알고리즘은 입력 라벨의 하드 클램핑을 수행 → **$\alpha = 0$** 을 의미
- 클램핑 인자는 완화될 수 있음
 - ex> **$\alpha = 0.2$** : 원래 레이블 분포의 80%를 항상 유지하지만 알고리즘은 분포의 신뢰도를 20% 이내로 변경

2) 그래프의 유사도 행렬

- **LabelPropagation**: 수정을 거치지 않은 데이터로 구성된 원시 유사성 행렬 활용
- **LabelSpreading**: 정규화 특성이 있는 손실 함수를 최소화
 - 정규화 특성은 노이즈에 더 강함

- 원본 그래프의 수정된 버전에서 반복됨
- 정규화된 그래프 라플라스 행렬을 계산하여 edge 가중치를 정규화
- Spectral clustering에도 활용됨
- 라벨 전파 모델의 **커널 함수(kernel method)**
 - **rbf**
 - $k(x_1, x_2) = \exp(-\gamma \|x_1 - x_2\|^2, \gamma > 0)$
 - γ 은 **gamma** 라는 키워드로 설정 가능
 - 메모리에서 밀집행렬(dense matrix)로 표현되는 완전 연결 그래프 (fully connected graph)를 생성
 - 행렬은 매우 클 수 있으며, 알고리즘의 각 반복에 대해 전체 행렬 곱셈 계산을 수행하는 비용(cost)과 결합하면 실행 시간이 엄청나게 길어질 수 있음
 - **knn**
 - $1[x' \in kNN(x)]$
 - k 는 **n_neighbors** 라는 키워드로 설정 가능
 - 실행 시간을 획기적으로 줄일 수 있는 훨씬 더 메모리 친화적인 희소 행렬(sparse matrix)을 생성
 - 커널 선택은 알고리즘의 **확장성과 성능**에 모두 영향을 미침

▼ 예시 코드

```
from sklearn.semi_supervised import LabelPropagation
from sklearn.datasets import make_classification

### 레이블이 있는 샘플 생성
X_labeled, y_labeled = make_classification(n_samples=100, n_features=10,
                                         n_informative=5, n_classes=2,
                                         random_state=42)

### 레이블이 없는 샘플 생성
X_unlabeled, _ = make_classification(n_samples=900, n_features=10,
                                     n_informative=5, n_classes=2,
                                     random_state=42)

### 전체 레이블 데이터 생성
X = np.concatenate([X_labeled, X_unlabeled])
y = np.concatenate([y_labeled, np.full(len(X_unlabeled), -1)])

### 준지도학습 모델 생성 및 훈련
lp_model = LabelPropagation() # 라벨 전파
```

```
lp_model.fit(X, y)

# 레이블이 없는 샘플에 대한 예측
unlabeled_predictions = lp_model.predict(X_unlabeled)
```

- 예제

- [Medium](#)
- [machinelearningmastery](#)

2. 전체적인 모델링 프로세스

2-1. 데이터 불러오기

- 필요한 데이터를 불러와 데이터프레임으로 저장
 - train: 학습용 파일
 - test: 최종 예측을 위한 파일
 - sample_submission: test 데이터를 통해 예측한 결과값을 저장하여 대회에 결과를 제출하기 위한 파일

2-2. 데이터 전처리/ EDA

- 변수 선택/가공
- 결측치 처리
- 변수의 자료형(`dtype`) 확인
- 범주형 변수 처리
 - Label Encoding
 - One-hot Encoding
- 이상치 처리

2-3. 준지도학습



모델링을 두 번 수행한다고 이해하면 편할 것 같습니다.

- 1) 의사 라벨링을 위한 모델링
- 2) 최종 예측을 위한 모델링

Step 0. 데이터 분리

- 레이블이 있는 데이터와 없는 데이터 분리
- X, y 분리

Step 1. Base Model 모델링

- 의사 라벨링(pseudo labeling)을 위한 기본 분류기인 Base Model 모델링
 - 모델 객체 생성
 - 학습/예측/평가
- 라벨이 있는(labeled) 데이터만 가지고 수행
- 하이퍼 파라미터 튜닝
 - `GridSearchCV`, `RandomizedSearchCV`, 또는 `Bayesian optimization` 등의 방법을 사용하여 최적의 하이퍼파라미터를 탐색
 - 또는 `PyCaret` 등의 AutoML 방법을 활용할 수 있음
 - 주어진 데이터와 문제에 맞는 하이퍼 파라미터 조합 찾기

⇒ 일반적인 분류 모델링 과정과 동일



변환 학습(= 그래프 기반) 방법의 경우 레이블이 있는 데이터와 레이블이 없는 데이터 간의 **연결 관계(neighborhood)**를 활용하여 레이블을 전파하는 방식

- Label Propagation 방식의 경우 Base Model을 생성할 필요가 x

Step 2. 의사 라벨링(Pseudo Labeling)

- Base model의 생성이 완료되면 해당 모델을 이용하여 레이블이 없는 데이터에 대한 의사 레이블을 생성
 - 대체적으로 모델이 예측한 값을 활용
 - ⇒ `model.predict(X_unlabeled)`
 - 주로 확률 기반으로 생성되지만 다른 방법을 통해 생성하는 것도 가능
- 라벨링이 완료된 의사 레이블을 기존의 레이블이 있는 데이터에 추가하여 새로운 학습용 데이터를 구성
 - 이렇게 구성된 데이터로 모델을 다시 학습
 - 해당 과정들을 반복하며 라벨이 있는 데이터를 확장시킴

- 최종 학습용 데이터 가공
 - 레이블이 **있는** 데이터가 많아진 상태
 - 필요에 따라 해당 과정 이후 레이블이 없는 데이터를 삭제하거나 결측치를 최빈값으로 채우는 등의 결측치 처리를 수행

2-4. 최종 예측을 위한 모델링

- 라벨링이 완료된 데이터를 최종 학습용 데이터로 활용하여 최종 예측을 위한 모델을 모델링
- 하이퍼 파라미터 튜닝 또한 기존의 분류 모델과 동일하게 수행할 수 있음

2-5. 최종 예측

- 최종 모델을 활용하여 test 데이터에 대한 예측 수행