EURON 2주차 예습과제_정윤지

Chapter 02: 사이킷런으로 시작하는 머신러닝

01. 사이킷런 소개와 특징

- 파이썬 머신러닝 라이브러리 중 가장 많이 사용되는 라이브러리(가장 쉽고 효율적인 개발 라이브러리를 제공)

특징

- 가장 쉽고 파이썬스러운 API를 제공
- 다양한 알고리즘과 개발을 위한 편리한 프레임워크와 API 제공
- 실전 환경에서 검증 완료, 많은 환경에서 사용되는 성숙한 라이브러리

02. 첫 번째 머신러닝 만들어보기 - 붓꽃 품종 예측하기

- 붓꽃 데이터 세트로 품종을 분류
- 붓꽃 데이터 세트는 꽃잎의 길이, 너비 등을 기반으로 꽃의 품종을 예측하기 위한 것
- 분류는 대표적인 지도학습
- 지도학습은 학습을 위한 다양한 피처와 분류 결정값인 레이블 데이터로 모델을 학습한 뒤, 별도의 테스트 데이터 세트에서 미지의 레이블을 예측함
- 학습을 위한 데이터 세트를 학습 데이터 세트, 예측 성능 평가를 위한 데이터 세트를 테스트 데이터 세트라고 함
- load_iris()를 이용하여 데이터 세트를 생성하고, ML의 알고리즘인 의사 결정 트리 알고리즘을 활용
- 학습 데이터 세트와 테스트 데이터 세트를 반드시 구분해야 하는데, train_test_split() 함수를 사용
- test_size =0.2 로 입력 파라미터를 설정하면 전체 데이터 중 테스트 데이터를 20% 할당함
- 데이터 구분 이후, 의사결정트리를 이용, DecisionTreeClassifier를 객체로 생성
- 예측 결과를 기반으로 의사결저트리 기반의 DecisionTreeClassifier의 예측 성능을 평가: 정확도를 지표로
- 예측한 붓꽃의 품종과 실제 테스트 데이터 세트의 붓꽃 품종이 얼마나 일치하는지 확인
- 정확도 측정을 위해 accuracy_score() 함수를 이용

붓꽃 데이터 세트 분류를 위한 프로세스

- 데이터 세트 분리: 학습 데이터와 테스트 데이터로 분리
- 모델 학습: 학습 데이터 기반 ML 알고리즘을 적용해 모델을 학습
- 예측 수행: 학습된 ML 모델을 이용해 테스트 데이터 분류(예측)
- 평가: 예측 결과와 테스트 데이터의 실제 결과를 비교해 ML 모델 성능을 평가

03. 사이킷런 기반 프레임워크 익히기

Estimator 이해 및 fit(), predict() 메서드

- 사이킷런은 ML 모델 학습을 위해 fit(), 학습 모델의 예측을 위해 predict() 메서드를 제공

- 사이킷런은 이 두 가지 만으로 지도학습의 두 축인 분류와 회귀의 알고리즘을 구현
- 분류 알고리즘을 구현한 클래스를 Classifier, 회귀 알고리즘을 구현한 클래스를 Regressor로 지칭(둘을 합쳐 Estimator 클래스)
- 비지도학습인 차원 축소, 클러스터링, 피처 추출 등을 구현한 클래스는 fit()과 transform()을 적용(입력 데이터 형태에 맞추어 데이터를 변환하기 위한 사전 구조를 맞추는 작업)
- fit()과 transform()을 개별적으로 적용하는 것과 fit_transform()을 한 번에 적용하는 것은 다름

사이킷런의 주요 모듈

예제 데이터

- sklearn.datesets: 사이킷런에 내장되어 예제로 제공하는 데이터 세트

피처 퍼리

- sklearn.preprocessing: 데이터 전처리에 필요한 다양한 가공 기능 제공(문자열을 숫자로 인코딩, 정규화 등)
- sklearn.feature_selection: 알고리즘에 영향을 미치는 피처를 우선 순위대로 셀렉션 작업을 수행하는 기능 제공
- sklearn.feature_extraction: 텍스트 데이터나 이미지 데이터의 벡터화된 피처를 추출하는 데 사용, 텍스트 데이터의 피처 추출은 text 모듈에, 이미지 데이터의 피처 추출은 image 모듈에 지원 API가 있음

피처 처리 & 차원 축소

- sklearn.decomposition: 차원 축소와 관련한 알고리즘을 지원하는 모듈, PCA NMF 등을 통해 차원 축소 기능 수행이 가능

데이터 분리, 검증 & 파라미터 튜닝

- sklearn.model_selection: 교차 검증을 위한 학습용/테스트 용 분리, 그리드 서치로 최적 파라미터 추출 등의 API 제공

평가

- sklearn.metrics: 분류, 회귀, 클러스터링, 페어와이즈에 대한 다양한 성능 측정 방법 제공(Accuracy 등)

ML 알고리즘

- sklearn.ensemble: 앙상블 알고리즘 제공(랜덤 포레스트, 에이다 부스트, 그래디언트 부스팅 등)
- sklearn.liner_model: 선형 회귀, 릿지, 라쏘 및 로지스틱 회귀 등 회귀 관련 알고리즘을 지원, SGD 관련 알고리즘 지원
- sklearn.naive_bayes: 나이브 베이즈 알고리즘 제공, 가우시간 NB, 다항분포 NB
- sklearn.neighbors: 최근접 이웃 알고리즘 제공(K-NN 등)
- sklearn.svm: 서포트 벡터 머신 알고리즘 제공
- sklearn.tree: 의사 결정 트리 알고리즘 제공

- sklearn.cluster: 비지도 클러스터링 알고리즘 제공(K 평균, 계층형, DBSCAN 등)

유틸리티
- Sklearn.pipeline: 피러 처리 등의 변환과 ML 알고리즘 학습과 예측을 함께 묶어 실행 가능한 유틸리티 제공

일반적으로 머신러닝 모델을 구축하는 주요 프로세스는 피처의 가공, 변경, 추출을 수행하는 피처 처리, ML 알고리즘 학습/예측 수행, 모델 평가의 단계를 반복적으로 수행하는 것

내장된 예제 데이터 세트
- 분류나 회귀를 연습하기 위한 예제용도의 데이터 세트와 분류나 클러스터링을 위해 표본 데이터로 생성될 수 있는 데이터 세트로 나뉨

분류나 회귀 연습용 예제 데이터
- datasets.load_boston(): 회귀 용도, 미국 보스턴의 집 피처들과 가격에 대한 데이터 세트
- datasets.load_breast_cancer(): 분류 용도, 위스콘신 유방암 피처들과 악성/음성 레이블 데이터 세트
- datasets.load_diabetes(): 회귀 용도, 당뇨 데이터 세트
- datasets.load_digits(): 분류 용도, 0에서 9까지 숫자의 이미지 픽셀 데이터 세트
- datasets.load_iris(): 분류 용도, 붓꽃에 대한 피처를 가진 데이터 세트

fetch 계열의 명령은 처음부터 저장돼 있지 않고, 인터넷에서 내려받아 홈 디렉터리 아래의 scikit_learn_data 라는 서브 디렉터리에 저장한 후 추후 불러들이는 데이터(최초 사용 시 인터넷 연결이 없으면, 사용 불가)
- fetch_covtype(): 회귀 분석용 토지 조사 자료
- fetch_20newsgroups(): 뉴스 그룹 텍스트 자료
- fetch_olivetti_faces(): 얼굴 이미지 자료
- fetch_lfw_people(): 얼굴 이미지 자료
- fetch_lfw_pairs(): 얼굴 이미지 자료
- fetch_rcv1(): 로이터 뉴스 말뭉치
- fetch_mldata(): ML 웹사이트에서 다운로드

분류와 클러스터링을 위한 표본 데이터 생성기
- datasets.make.classifications(): 분류를 위한 데이터 세트를 생성, 높은 상관도나 불필요 속성 등의 노이즈 효과를 위한 데이터를 무작위로 생성
- datasets.make_blobs(): 클러스터링을 위한 데이터 세트를 무작위 생성, 군집 지정 개수에 따라 여러 가지 클러스터링을 위한 데이터 세트를 만들어 줌

사이킷런 내장 데이터 세트의 개별 키
- data: 피처의 데이터 세트를 가리킴, 넘파이 배열

- target: 분류 시 레이블 값, 회귀일 때는 숫자 결과 값 데이터 세트
- target_names: 개별 레이블의 이름을 나타냄, 넘파이 배열 또는 파이썬 리스트
- feature_names: 피처의 이름을 나타냄, 넘파이 배열 또는 파이썬 리스트
- DESCR: 데이터 세트에 대한 설명과 각 피처의 설명을 나타냄, 스트링 타입

피처의 데이터 값을 반환 받기 위해 내장 데이터 세트 API를 호출한 후, KEY 값을 지정
load_iris() API의 반환 결과는 sklearn, until, Bunch 클래스이며 Bunch 클래스는 파이썬 딕셔너리 자료형과 유사
데이터 기는 피처들의 데이터 값을 가리키며, 데이터 세트가 딕셔너리 형태이기 때문에 피처 데이터 값을 추출하기 위해서는 데이터 세트를 이용

04. Model Selection 모듈 소개
사이킷런의 model_selection 모듈은 학습 데이터와 테스트 데이터 세트를 분리하거나 교차 검증 분할 및 평가, Estimator의 하이퍼 파라미터를 튜닝하기 위한 다양한 함수와 클래스를 제공

학습/테스트 데이터 세트 분리 - train_test_split()
- 학습 데이터 세트로 학습하고, 예측하면 학습한 학습 데이터 세트를 기반으로 예측했기 때문에 문제가 발생
- 예측을 수행하는 데이터 세트는 학습을 수행한 학습용 데이터 세트가 아니라 전용의 테스트 데이터 세트여야 함
- 사이킷런의 train_test_spilt()를 통해 원본 데이터 세트에서 학습 및 테스트 데이터 세트를 쉽게 분리할 수 있음

파라미터 종류
- test_size: 전체 데이터에서 테스트 데이터 세트 크기를 얼마로 샘플링할 것인가를 결정(디폴트 25%)
- train_size(): 전체 데이터에서 학습용 데이터 세트 크기를 얼마로 샘플링 할 것인가를 결정
- shuffle: 데이터를 분리하기 전에 데이터를 미리 섞을 것인지 결정, 데이터 분산으로 효율적인 학습 및 데이터 세트를 만드는 데 사용(디폴트 True)
- random_state: 호출할 때마다 동일한 학습/테스트 용 데이터 세트를 생성하기 위해 주어지는 난수의 값, train_test_spilt()는 호출 시 무작위로 데이터를 분리하므로 지정하지 않으면 수행할 때마다 다른 학습/테스트 용 데이터를 생성함
- train_test_spilt() 의 반환 값은 튜플 형태

학습을 위한 데이터의 양을 일정 수준 이상으로 보장하는 것도 중요하지만, 학습된 모델에 대해 다양한 데이터를 기반으로 예측 성능을 평가해보는 것도 중요

교차 검증
- 학습 데이터와 별도의 테스트 데이터를 사용하는 방법 역시 과적합(모델이 학습 데이터에 과도하게 최적화 되어 실제 예측을 다른 데이터로 수행할 경우 예측 성능이 과도하게 떨어지는 것)에 취약한 약점을 가질 수 있음
- 과적합 문제의 개선을 위해 교차 검증을 이용해 다양한 학습, 평가를 진행
- 교차 검증은 데이터 편증을 막기 위해 별도의 여러 세트로 구성된 학습 데이터 세트와 검증 데이터 세트에서 학습과 평가를 수행하는 것, 각 세트에서 수행한 평가 결과에 따라 하이퍼 파라미터 튜닝 등의 모델 최적화를 더욱 손쉽게 할 수 있음

- 대부분의 ML 모델의 성능 평가는 교차 검증 기반으로 1차 평가를 진행, 최종적으로 테스트 데이터 세트에 적용해 평가하는 프로세스
- ML에 사용되는 데이터 세트를 세분화하여 학습, 검증, 테스트 데이터 세트로 나눌 수 있음
- 테스트 데이터 세트 외에 별도의 검증 데이터 세트를 두어, 최종 평가 이전에 학습된 모델을 다양하게 평가할 수 있음

K 폴드 교차 검증
- 가장 보편적으로 사용되는 교차 검증 기법
- K개의 데이터 폴드 세트를 만들어, K번 만큼 각 폴드 세트에 학습과 검증 평가를 반복적으로 수행
- 학습 데이터 세트와 검증 데이터 세트를 점진적으로 변경하면서 마지막 K번째까지 학습과 검증을 수행
- K개의 예측 평가를 구한 후, 이를 평균내 K폴드 평가 결과로 반영
- 사이킷런에서는 K폴드 교차 검증 프로세스를 구현하기 위해 KFold와 StratifiedKFold 클래스를 제공

Stratified K 폴드
- 불균형한 분포도를 가진 레이블(결정 클래스) 데이터 집합을 위한 K 폴드 방식
- 불균형한 분포도를 가진 레이블 데이터 집합은 특정 레이블 값이 특이하게 많거나 매우 적어서 값의 분포가 한쪽으로 치우치는 것을 말함
- K 폴드가 레이블 데이터 집합이 원본 데이터 집합의 레이블 분포를 학습 및 테스트 세트에 제대로 분배하지 못하는 경우의 문제를 해결

교차 검증을 보다 간편하게 - cross_val_score()
- 사이킷런은 교차 검증을 편리하게 수행할 수 있도록 하는 API를 제공
- 대표적인 API가 cross_val_score()
- K Fold로 데이터를 학습하고 예측하는 코드는 폴드 세트를 설정하고 for 루프에서 반복으로 학습 및 테스트 데이터의 인덱스를 추출한 뒤 반복적으로 학습과 예측을 수행하고 예측 성능을 반환하는데, cross_val_score()는 이런 일련의 과정을 한꺼번에 수행해주는 API임
- cross_val_score()는 CV로 지정된 횟수만큼 scoring 파라미터로 지정된 평가 지표로 평가 결괏값을 배열로 반환, 일반적으로 이를 평균해 평가 수치로 사용
- 내부에서 Estimator를 학습, 예측, 평가시켜주므로 간단한 교차 검증 수행이 가능

GridSaerchCV - 교차 검증과 최적 하이퍼 파라미터 튜닝을 한 번에
- 사이킷런은 GridSaerchCV API를 통해 알고리즘에 사용되는 하이처 파라미터를 순차적으로 입력하면서 편리하게 최적의 파라미터를 도출할 수 있는 방안을 제공
- GridSaerchCV는 교차 검증을 기반으로 하이퍼 파라미터의 최적 값을 찾게 해줌(데이터 세트를 cross-validation을 위한 학습/데이터 세트로 자동으로 분할한 뒤 하이퍼 파라미터 그리드에 기술된 모든 파라미터를 순차적으로 적용해 최적의 파라미터를 찾을 수 있게 함

- 사용자가 튜닝하고자 하는 여러 종류의 하이퍼 파라미터를 다양하게 테스트하며 최적의 파라미터를 편리하게 찾아주지만, 동시에 순차적으로 파라미터를 테스트하므로 수행시간이 상대적으로 오래 걸림

GridSaerchCV 클래스의 생성자로 들어가는 주요 파라미터
- param_gird: key 리스트 값을 가지는 딕셔너리가 주어짐. 사용될 여러 파라미터 값을 지정
- scoring: 예측 성능을 측정할 평가 방법을 지정, 사이킷런의 성능 평가 지표를 지정하는 문자열로 지정하나, 별도의 성능 평가 지표 함수도 지정할 수 있음
- cv: 교차 검증을 위해 분할되는 학습/데이터 세트의 개수를 지정
- refit: 디폴트가 True 이며 True로 생성 시 가장 최적의 하이퍼 파라미터를 찾은 뒤 입력된 estimator 객체를 해당 하이퍼 파라미터로 재학습

주요 칼럼 별 의미
- params: 수행할 때마다 적용된 개별 하이퍼 파라미터 값을 나타냄
- rank_test_score: 하이퍼 파라미터별로 성능이 좋은 score 순위를 나타냄(1이 가장 뛰어난 순위, 이때의 파라미터가 최적의 하이퍼 파라미터)
- mean_test_score: 개별 하이퍼 파라미터별로 CV의 폴딩 테스트 세트에 대해 총 수행한 평가 평균값

GridSaerchCV 객체의 fit()을 수행하면 최고 성능을 나타낸 하이퍼 파라미터의 값과 그 때의 평가 결과 값이 각각 best_params_, best_score_ 속성에 기록됨

05. 데이터 전처리
- ML 알고리즘은 데이터에 기반하고 있으므로 어떤 데이터를 입력으로 가지느냐에 따라 결과가 달라질 수 있음
- 사이킷런의 ML 알고리즘은 결손값(Null)을 허용하지 않음
- Null 값을 어떻게 처리하는 지는 경우에 따라 다음(Null 값의 수가 적으면 피처의 평균값 등으로 대체할 수 있지만, Null 값이 일정 수준 이상이면 결정이 힘듦, 단순 평균값으로 대체할 경우 예측 왜곡이 심할 수 있음)
- 사이킷런의 머신러닝 알고리즘은 문자열 값을 입력값으로 허용하지 않음 -> 모든 문자열 값은 인코딩되어 숫자형으로 변환해야 함(문자열 피처는 일반적으로 카테고리형 피처와 텍스트형 피처를 의미함)

데이터 인코딩
- 머신러닝을 위한 대표적인 인코딩 방식은 레이블 인코딩과 원-핫 인코딩이 있음

레이블 인코딩
- 카테고리 피처를 코드형 숫자 값으로 변환
- 사이킷런의 레이블 인코딩은 LabelEncoder 클래스로 구현
- LabelEncoder를 객체로 생성한 후 fit()과 transform()을 호출해 레이블 인코딩을 수행
- 간단하게 문자열 값을 숫자형 카테고리 값으로 변환

- 일괄적인 숫자 값으로 변환되면서 몇몇 ML 알고리즘에는 이를 적용할 경우 예측 성능이 떨어지는 경우 발생(숫자 값의 경우 크고 작음에 대한 특성이 작용하기 때문)
- 이러한 특성 때문에 레이블 인코딩은 선형회귀와 같은 ML 알고리즘에는 적용하지 않아야 함
- 트리 계열의 ML 알고리즘은 숫자의 특성을 반영하지 않으므로, 레이블 인코딩도 문제가 없음

원-핫 인코딩(One-Hot Encoding)
- 레이블 인코딩의 문제점을 해결하기 위한 인코딩 방식
- 피처 값의 유형에 따라 새로운 피처를 추가해, 고유 값에 해당하는 칼럼에만 1을 표시, 나머지 칼럼에 0을 표시
- 행 형태로 되어 있는 피처의 고유 값을 열 형태로 차원 변환한 뒤, 고유 값에 해당하는 칼럼에만 1을 표시
- 사이킷런에서 OneHotEncoder 클래스로 변환이 가능
- toarray() 메서드를 이용해 밀집 행렬로 변환하는 과정이 필요

피처 스케일링과 정규화
- 서로 다른 변수의 값 범위를 일정한 수준으로 맞추는 작업
- 대표적인 방법으로 표준화, 정규화가 있음
- 표준화: 데이터의 치퍼 각각이 평균이 0이고, 분산이 1인 가우시안 정규분포를 가진 값으로 변환하는 것

$$x_i\_new = \frac{x_i - mean(x)}{stdev(x)}$$

- 정규화: 서로 다른 피처의 크기를 통일하기 위해 크기를 변환해주는 개념, 개별 데이터의 크기를 모두 똑같은 단위로 변경하는 것

$$x_i\_new = \frac{x_i - \min(x)}{\max(x) - \min(x)}$$

- 사이킷런의 전처리에서 제공하는 모듈과 일반적인 정규화는 차이가 있음

StandardScaler
- 사이킷런에서 제공하는 대표적인 피처 스케일링 클래스
- 표준화를 쉽게 지원하기 위한 클래스
- 개별 피처를 평균이 0이고 분산이 1인 값으로 변환
- 사이킷런에서 구현한 RBF 커널을 이용하는 서포츠 백터 머신이나 선형 회귀, 로지스틱 회귀는 데이터가 가우시안 분포를 가지고 있다고 가정하고 구현되었기 때문에 사전에 표준화를 적용하는 것은 예측 성능 향상에 중요한 요소가 될 수 있음

MinMaxScaler

- 사이킷런에서 제공하는 대표적인 피처 스케일링 클래스
- 데이터 값을 0과 1 사이의 범위 값으로 변환(음수 값이 있으면 -1에서 1값으로 변환)

학습 데이터와 테스트 데이터의 스케일링 변환 시 유의점
- StandardScaler 나 MinMaxscaler 와 같은 Scaler 객체를 이용해 데이터의 스케일링 변환 시 fit(), transform(), fit_transform() 메서드를 이용
- Scaler 객체를 이용해 학습 데이터 세트로 fit()과 transform()을 적용하면 테스트 데이터 세트로는 다시 fit()을 수행하지 않고 학습 데이터 세트로 fit()을 수행한 결과를 이용해 transform() 변환을 적용해야 함
- 학습 데이터와 테스트 데이터의 fit(), transform(), fit_transform()을 이용해 스케일링 변환 시 가능하다면 전체 데이터의 스케일링 변환을 적용한 뒤 학습과 테스트 데이터로 분리해야 하며, 테스트 데이터 변환 시 fit() 이나 fit_transform()을 적용하지 않고 학습 데이터로 이미 fit() 된 Scaler 객체를 이용해 transform()으로 변환해야 함

06. 사이킷런으로 수행하는 타이타닉 생존자 예측
- 실습

07. 정리
- 사이킷런은 파이썬 계열의 대표적인 머신러닝 패키지로, 머신러닝 알고리즘 제공, 직관적인 API 프레임워크, 다양한 모듈을 지원
- 머신러닝 애플리케이션은 데이터의 가공 및 변환 과정의 전처리 작업, 데이터를 학습 데이터와 테스트 데이터로 분리하는 데이터 세트 분리 작업을 거친 후 학습 데이터를 기반으로 머신러닝 알고리즘을 적용해 모델을 학습
- 학습된 모델을 기반으로 테스트 데이터에 대한 예측을 수행, 예측된 결과 값을 실제 결과 값과 비교해, 머신러닝 모델에 대한 평가를 수행하는 방식으로 구성
- 데이터 전처리 작업은 오류 데이터의 보정이나 Null 값 처리 등의 다양한 데이터 클렌진 작업, 레이블 인코딩이나 원-핫 인코딩과 같은 인코딩 작업, 데이터의 스케일링/정규화 작업 등으로 머신러닝 알고리즘이 최적으로 수행되도록 데이터를 사전처리
- 머신러닝 모델은 학습 데이터 세트로 학습, 별도의 테스트 데이터 세트로 평가되어야 함
- 테스트 데이터 세트에만 치우친 머신러닝 모델을 극복하기 위해 교차 검증을 진행하며 다양한 클래스와 함수를 제공

Chapter 03: 평가

분류의 성능 평가 지표
- 정확도
- 오차행렬
- 정밀도

- 재현율
- F1 스코어
- ROC AUC

분류의 결정 클래스 값 종류의 유형에 따라 긍정/부정과 같은 2개의 결과 값만을 가지는 이진 분류와 여러 개의 결정 클래스 값을 가지는 멀티 분류로 나뉨

01. 정확도

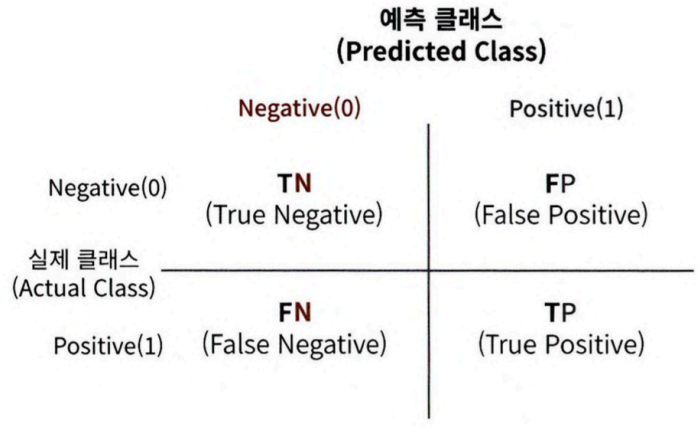
실제 데이터에서 예측 데이터가 얼마나 같은지를 판단하는 지표

$$\text{정확도(Accuracy)} = \frac{\text{예측 결과가 동일한 데이터 건수}}{\text{전체 예측 데이터 건수}}$$

- 직관적으로 모델 예측 성능을 나타내는 평가 지표
- 이진 분류의 경우 데이터의 구성에 따라 ML 모델의 성능 왜곡이 가능(정확도 수치만으로 성능을 평가하지 않음)
- 한계점 극복을 위해 여러 가지 분류 지표와 함께 적용하여 모델 성능 평가가 필요

02. 오차 행렬

- 성능 지표로 활용되는 오차행렬은 학습된 분류 모델이 예측을 수행하면서 얼마나 헷갈리고 있는지 보여줌
- 이진 분류의 예측 오류가 얼마인지와 더불어 어떠한 유형의 예측 오류가 발생하고 있는지를 함께 나타내는 지표



- 4분면 행렬에서 실제 레이블 클래스 값과 예측 레이블 클래스 값이 어떠한 유형을 가지고 매핑되는지를 나타냄
- TN: 예측 값을 negative 값 0 으로 예측했고 실제 값 역시 negative 값 0
- FP: 예측 값을 positive 값 1로 예측했는데 실제 값은 negative 값 0
- FN: 예측 값을 negative 값 0으로 예측했는데 실제 값은 positive 값 0
- TP: 예측 값을 positive 값 1로 예측했는데 실제 값 역시 positive 값 1
- 사이킷런은 오차 행렬을 구하기 위해 confusion_matrix() API를 제공

- TP,TN,FP,TN 값은 Classifier 성능의 여러 면모를 판단할 수 있는 기반 정보를 제공(이 값을 조합해 정확도, 정밀도, 재현율 값을 알 수 있음)
- 일반적으로 불균형한 레이블 클래스를 가지는 이진 분류 모델에서는 많은 데이터 중에서 중점적으로 찾아야 하는 매우 적은 수의 결과 값에 positive를 설정해 1 값을 부여하고, 그렇지 않은 경우 negative로 0 값을 부여하는 경우가 다수
- 불균형한 이진 분류 데이터 세트에서는 positive 데이터 건수가 매우 작기 때문에 데이터에 기반한 ML 알고리즘은 positive보다는 negative로 예측 정확도가 높아지는 경향이 발생(불균형한 데이터 세트에서 정확도만으로는 모델 신뢰도가 떨어질 수 있음)

03. 정밀도와 재현율
- Positive 데이터 세트의 예측 성능에 초점을 맞춘 평가 지표

$$\text{정밀도} = TP / (FP + TP)$$

$$\text{재현율} = TP / (FN + TP)$$

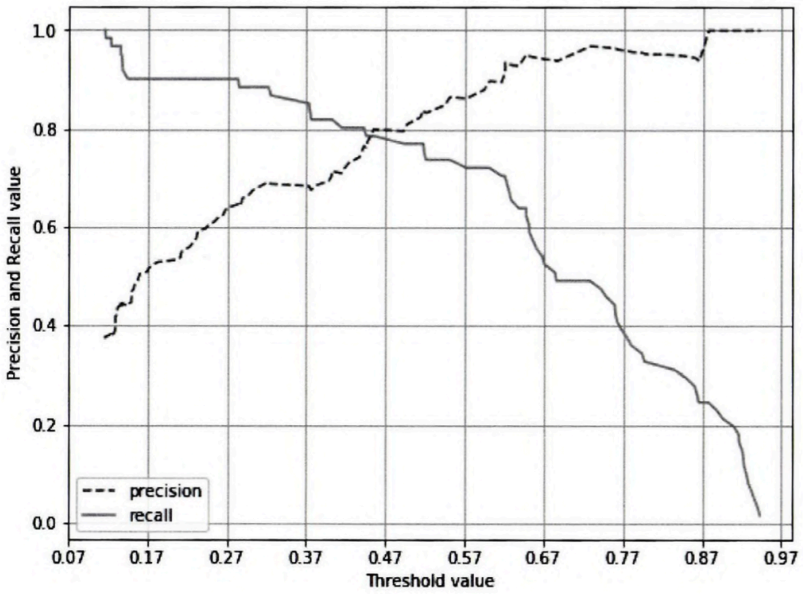
정밀도
- 예측을 positive로 한 대상 중 예측과 실제 값이 positive로 일치한 데이터의 비율
- 예측 성능을 더욱 정밀하게 측정하기 위한 평가 지표로 양성 예측도라고도 불림
- 실제 negative 음성인 데이터 예측을 positive 양성으로 잘못 판단하게 되면 업무 상 큰 영향이 발생하는 경우 중요 지표

재현율
- 실제 값이 positive인 대상 중 예측과 실제 값이 positive로 일치한 데이터의 비율
- 민감도 또는 TPR 이라고도 불림
- 실제 positive 양성 데이터를 negative로 잘못 판단하게 되면 업무 상 큰 영향이 발생하는 경우 중요 지표
- 재현율이 정밀도보다 상대적으로 중요한 업무가 많음

정밀도/재현율 트레이드오프
- 분류하려는 업무의 특성상 정밀도 또는 재현율이 특별히 강조돼야 할 경우 분류의 결정 임계 값을 조정해 정밀도 또는 재현율의 수치를 높일 수 있음
- 상호 보완적인 평가 지표이기 때문에 한 쪽을 강제로 높이면 다른 하나의 수치는 떨어지기 쉬운데, 이를 정밀도/재현율의 트레이드오프라고 부름
- 사이킷런의 분류 알고리즘은 예측 데이터가 특정 레이블에 속하는지를 계산하기 위해 먼저 개별 레이블 별로 결정 확률을 구한 후, 예측 확률이 큰 레이블 값으로 예측하게 됨
- 사이킷런은 개별 데이터 별로 예측 확률을 반환하는 메서드인 predict_proba()를 제공

- predict_proba() 메서드는 학습이 완료된 사이킷런 classifier 객체에서 호출 가능하며 테스트 치퍼 데이터 세트를 파라미터로 입력해주면 테스트 피처 레코드의 개별 클래스 예측 확률을 반환
- predict() 메서드: predict_proba() 호출 결과로 반환된 배열에서 분류 결정 임계 값보다 큰 값이 들어있는 칼럼의 위치를 받아서 최종적으로 에측 클래스를 결정하는 API
- 사이킷런은 분류 결정 임계 값을 조절해 정밀도와 재현율의 성능 수치를 상호 보완적으로 조정할 수 있음
- Positive 예측 값이 많아지면 상대적으로 재현율 값이 높아짐(양성 위주 예측으로 인해 실제 양성을 음성으로 예측하는 횟수가 상대적으로 줄어들기 때문)
- 사이킷런은 precision_recall_curve() API를 제공

입력 파라미터	**y_true**: 실제 클래스값 배열 (배열 크기= [데이터 건수]) **probas_pred**: Positive 칼럼의 예측 확률 배열 (배열 크기= [데이터 건수])
반환 값	**정밀도**: 임곗값별 정밀도 값을 배열로 반환 **재현율**: 임곗값별 재현율 값을 배열로 반환



- 임계 값이 낮을수록 많은 수의 양성 예측으로 인해 재현율 값이 극도로 높아지고 정밀도 값이 극도로 낮아짐

정밀도와 재현율의 맹점

- 임계 값을 변경함에 따라 정밀도와 재현율의 수치가 변경되는데, 이러한 변경은 업무 환경에 맞게 두 개의 수치를 상호 보완할 수 있는 수준에서 적용돼야 함
- 단순히 하나의 성능 지표 수치를 높이기 위한 수단으로 사용되는 것을 경계해야 함

정밀도가 100%가 되는 방법

- 확실한 기준이 되는 경우에만 positive로 에측하고 나머지는 모두 negative로 예측

재현율이 100%가 되는 방법

- 모든 환자를 positive로 예측

04. F1 스코어

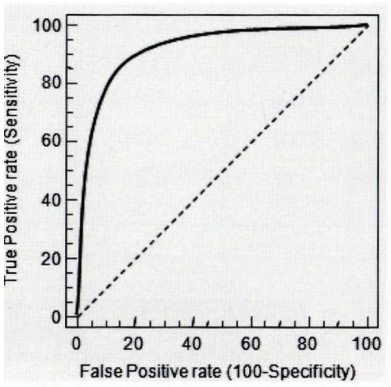
- 정밀도와 재현율을 결합한 지표
- 어느 한 쪽으로 치우치지 않는 수치를 나타낼 때 상대적으로 높은 값을 가짐

$$F1 = \frac{2}{\frac{1}{recall} + \frac{1}{precision}} = 2 * \frac{precision * recall}{precision + recall}$$

- F1_score() API를 제공

05. ROC 곡선과 AUC

- 이진 분류의 예측 성능 측정에서 중요하게 사용되는 지표
- FPR이 변할 때 TPR(재현율=민감도)이 어떻게 변하는지를 나타내는 곡선
- TNR(특이성): 민감도에 대응하는 지표
- 민감도는 실제 값 양성이 정확하게 예측되어야 하는 수준을 나타내며, 특이성은 실제 값 음성이 정확히 예측돼야 하는 수준을 나타냄



〈 ROC 곡선 예시 〉

- 곡선이 가운데 직선에 가까울수록 성능이 떨어지는 것이며 멀어질수록 성능이 뛰어남
- 사이킷런은 ROC 곡선을 구하지 위해 roc_curve() API를 제공
- roc_curve()의 주요 입력 파라미터와 반환 값

입력 파라미터	**y_true**: 실제 클래스 값 array (array shape = [데이터 건수]) **y_score**: predict_proba( )의 반환 값 array에서 Positive 칼럼의 예측 확률이 보통 사용됨. array, shape = [n_samples]
반환 값	**fpr**: fpr 값을 array로 반환 **tpr**: tpr 값을 array로 반환 **thresholds**: threshold 값 array

- 일반적으로 ROC 곡선 자체는 FPR 과 TPR의 변화 값을 보는 데 이용

- 분류의 성능 지표로 사용되는 것은 ROC 곡선 면적에 기반한 AUC 값으로 결정
- AUC: ROC 곡선 밑의 면적을 구한 것으로, 일반적으로 1에 가까울수록 좋은 수치(보통의 분류는 0.5 이상의 AUC 값을 가짐)

06. 피마 인디언 당뇨병 예측
- 실습

07. 정리
- 이진 분류의 레이블 값이 불균형하게 분포될 경우 단순히 예측 결과와 실제 결과가 일치하는 지표인 정확도만으로는 머신러닝 모델의 예측 성능을 평가할 수 없음
- 오차 행렬은 negative 와 positive 값을 가지는 실제 클래스 값과 예측 클래스 값이 True 와 False에 따라 TN, FP, FN, TP 로 매핑되는 4분면 행렬을 기반으로 예측 성능을 평가
- 정밀도와 재현율은 positive 데이터 세트의 예측 성능에 초점을 맞는 지표
- F1 스코어는 정밀도와 재현율을 결합한 평가 지표, 정밀도와 재현율이 어느 한 쪽으로 치우치지 않을 때 높은 지표 값을 가지게 됨
- ROC-AUC는 일반적으로 이진 분류의 성능 평가를 위해 가장 많이 사용되는 지표
- AUC 값은 ROC 곡선 밑의 면적을 구한 것으로 일반적으로 1에 가까울수록 좋은 수치