모델 평가와 성능 향상

July 26, 2023

목차

- 교차 검증
- 그리드 탐색
- 평가 지표와 측정

모델평가 개요

- 지도학습과 비지도 학습이론 학습
  - 다양한 머신러닝 알고리즘
- **모델평가 과정**
  - 데이터셋을 훈련 데이터와 테스트 데이터로 분리 – train_test_split
  - 모델 학습 - fit
  - 모델 평가 – score
    - score 메서드는 정확히 분류된 샘플의 비율을 계산

```
from sklearn.datasets import make_blobs
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split

X, y = make_blobs(random_state=0)

X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=0)

logreg = LogisticRegression().fit(X_train, y_train)
print("테스트 세트 점수: {:.2f}".format(logreg.score(X_test, y_test)))
# 테스트 세트 점수: 0.88
```

- **데이터를 훈련 세트와 테스트 세트로 나누는 이유**
  - 새로운 데이터에 모델이 얼마나 잘 일반화되는지 측정

모델이 훈련세트에 잘 맞는 것보다, 학습 과정에 없던 데이터에 대해 예측을 얼마나 잘 하느냐가 중요

평가 방법의 확장

- 1. 안정적인 일반화 성능 측정 방법인 교차 검증
- 2. score 메서드가 제공하는 정확도와 R^2값 이외의 분류와 회귀 성능을 측정하는 방법

교차검증

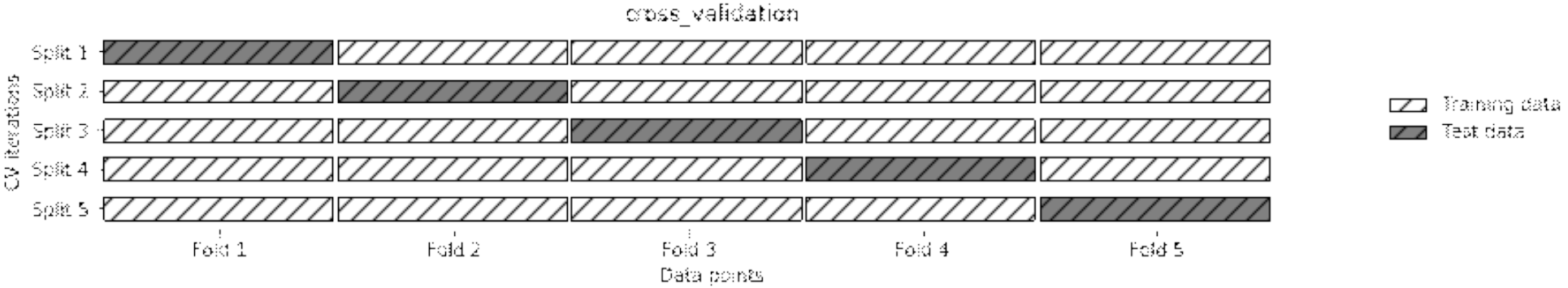
➢ 교차검증(cross-validation)

- 일반화 성능을 측정하기 위해 훈련 세트와 데이터 세트로 나누는 것보다 더 안정적이고 뛰어난 통계적 평가 방법
- 데이터를 여러 번 반복해서 나누고 여러 모델을 학습
- 종류
  - K-겹 교차검증
  - 계층별 k-겹 교차검증(k-fold cross-validation)
    - ➔ k는 특정 숫자인데 보통 5또는 10을 사용
  - 임의 분할 교차검증
  - 그룹별 교차검증
- Sklearn 에서는 cross_val_score 함수 및 KFold, StratifiedKFold, GroupKFold 클래스 등 제공

K-겹 교차검증

- 5-겹 교차 검증에서의 데이터 분할
  - 첫 번째 모델은
    - 첫 번째 폴드를 테스트로 사용, 나머지 2~5까지를 훈련 세트로 사용하여 학습
  - 두 번째 모델은
    - 두 번째 폴드를 테스트로 사용, 나머지 1,3~5까지를 훈련 세트로 사용하여 학습

mglearn.plots.plot_cross_validation()



[5-겹 교차 검증에서의 데이터 분할]

K-겹 교차검증

- **붓꽃 데이터셋을 로지스틱회귀로 교차검증**
  - Cross_val_score 함수 매개변수는 평가모델, 훈련 데이터, 타겟 레 이블임(기본 3-겹 교차검증)

```
from sklearn.model_selection import cross_val_score
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression

iris = load_iris()
logreg = LogisticRegression()

scores = cross_val_score(logreg, iris.data, iris.target)  #모델,훈련 데이터, 타깃 레이블
print("교차 검증 점수: {}".format(scores))
# 교차 검증 점수: [0.961 0.922 0.958]
```

- cross_val_score의 **기본값은 3-겹 교차** 검증 정확도 **값이 3개가 반환**
- 폴드의 수는 cv 매개변수를 사용해서 변경

K-겹 교차검증

- **붓꽃 데이터셋을 로지스틱회귀로 교차검증**
- 𝕞𝕏🖬인 교차 검증

```
scores = cross_val_score(logreg, iris.data, iris.target, cv=5)
print("교차 검증 점수: {}".format(scores))
# 교차 검증 점수: [1.    0.967 0.933 0.9   1.   ]
```

- 보통 교차 검증의 정확도를 간단하게 나타내려면 평균을 사용

```
print("교차 검증 평균 점수: {:.2f}".format(scores.mean()))
# 교차 검증 평균 점수: 0.96
```

- **교차 검증 평균값 모델의 정확도가 대략 96%일 것으로 기대**

교차검증

➢ 교차검증 장점

- 테스트 세트에는 모델별로 각 샘플이 정확하게 한번씩 들어가므로 모델의 일반화 성능을 보장

  데이터를 여러 겹으로 나누면 모델이 훈련 데이터에 얼마나 민감한지 알 수 있음
- 분할을 한 번 했을 때보다 데이터를 더 효과적으로 사용할 수 있음
  - Train_test_split 은 항상 훈련세트 75%, 테스트세트 25%만을 사용하지만 5겹은 80%/20%, 10겹은 90%/10% 로 조절 가능

➢ 단점

- 연산 비용이 늘어남
- 모델을 k개 만들어야 하므로 하나의 모델보다 k배 느림

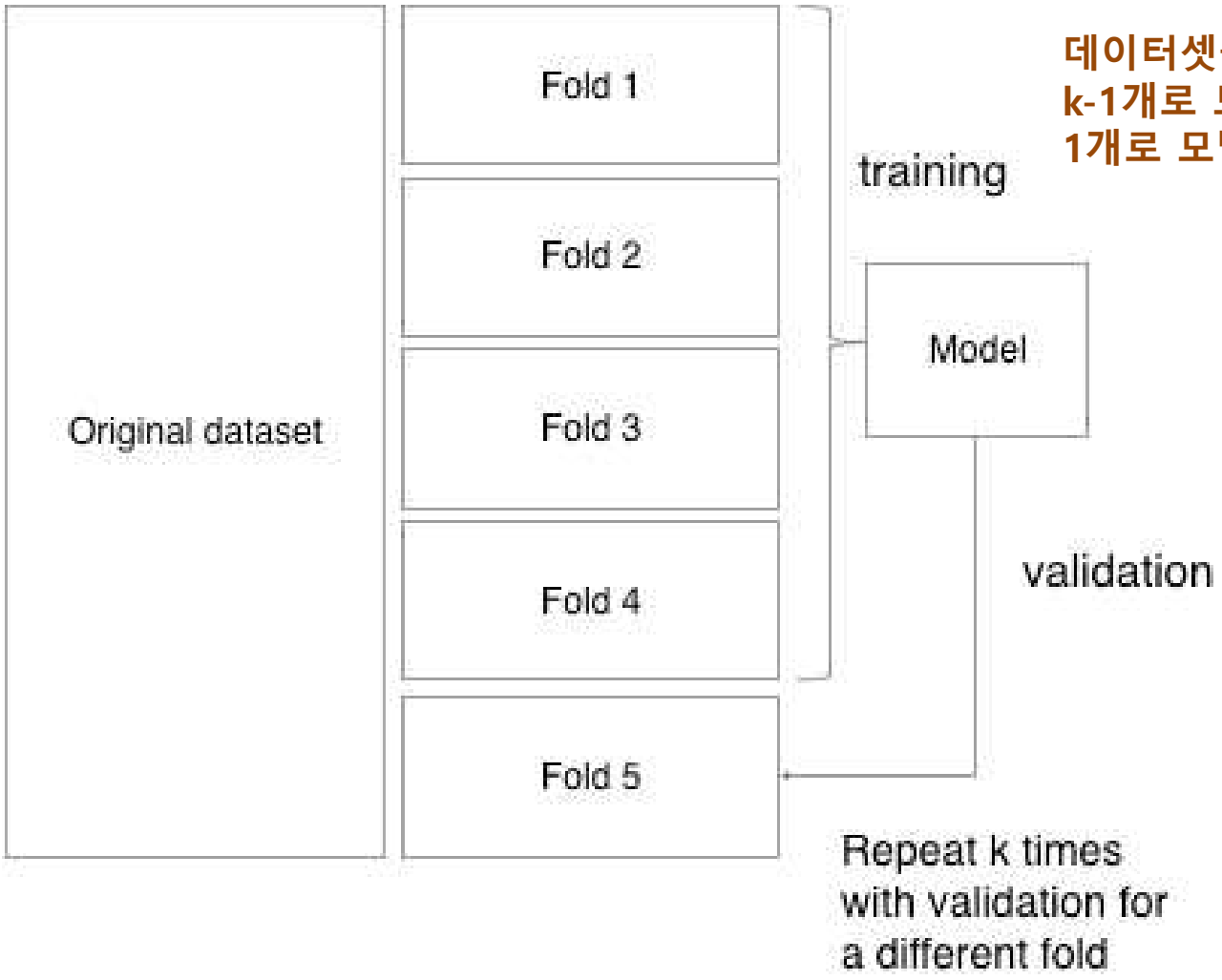
계층별 k-겹 교차 검증

- K-겹 교차검증의 문제점
  - 출력 클래스 비율이 일정하지 않으면 제대로 학습되지 않음
  - 붓꽃 데이터 출력 사례
  - 출력 클래스가 1/3씩이므로 3-겹 교차검증의 정확도는 0이 됨

```
from sklearn.datasets import load_iris
iris = load_iris()
print("Iris 레이블:\n{}".format(iris.target))
Iris 레이블:
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]]
```

- 이 데이터에 3-겹 교차 검증을 적용한다고 생각
  - 첫 번째 폴드는 클래스 0만 가지고 있으므로 정확도는 0이 된다.
  - 두 번째, 세 번째도 같은 방법으로 정확도는 0이 된다.

교차 검증(Cross-Validation)



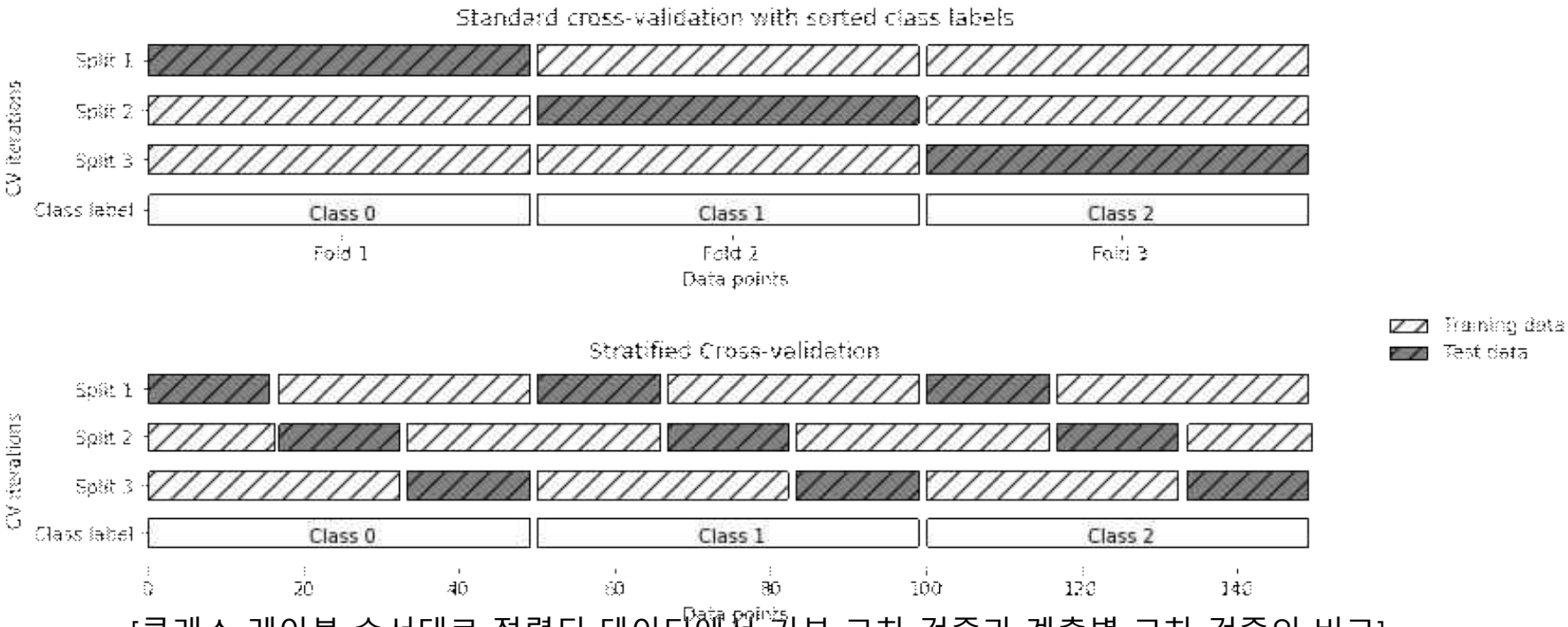
데이터셋을 k개로 쪼개어
k-1개로 모델을 학습하고,
1개로 모델을 검증합니다.

계층별 k-겹 교차 검증

➢ 계층별 k-겹 교차검증 방식

- 일반/계층별 k-겹 교차검증 비교

```
mglearn.plots.plot_stratified_cross_validation()
```



[클래스 레이블 순서대로 정렬된 데이터에서 기본 교차 검증과 계층별 교차 검증의 비교]

- 90%가 클래스 A, 10%가 클래스 B인 데이터라면, 계층별 교차 검증은 각 폴드에 9:1 비율 만듬
- **분류기의 일반화 성능을 측정 k-겹 교차 검증보다 계층별 k-겹 교차 검증을 사용하는 것이 좋다.**

교차 검증 상세 옵션

➢ 교차검증 분할기(Splitter)

- Cross_val_score 함수의 cv 매개변수에 전달

- **교차검증 분할기(Splitter)**
  - 계층별 폴드 대신 샘플 순서를 섞는 방법(shuffle=True)도 가능
- **LOOCV(Leave-one-out cross-validation)**
  - 테스트 폴드에 단 하나의 샘플만 포함

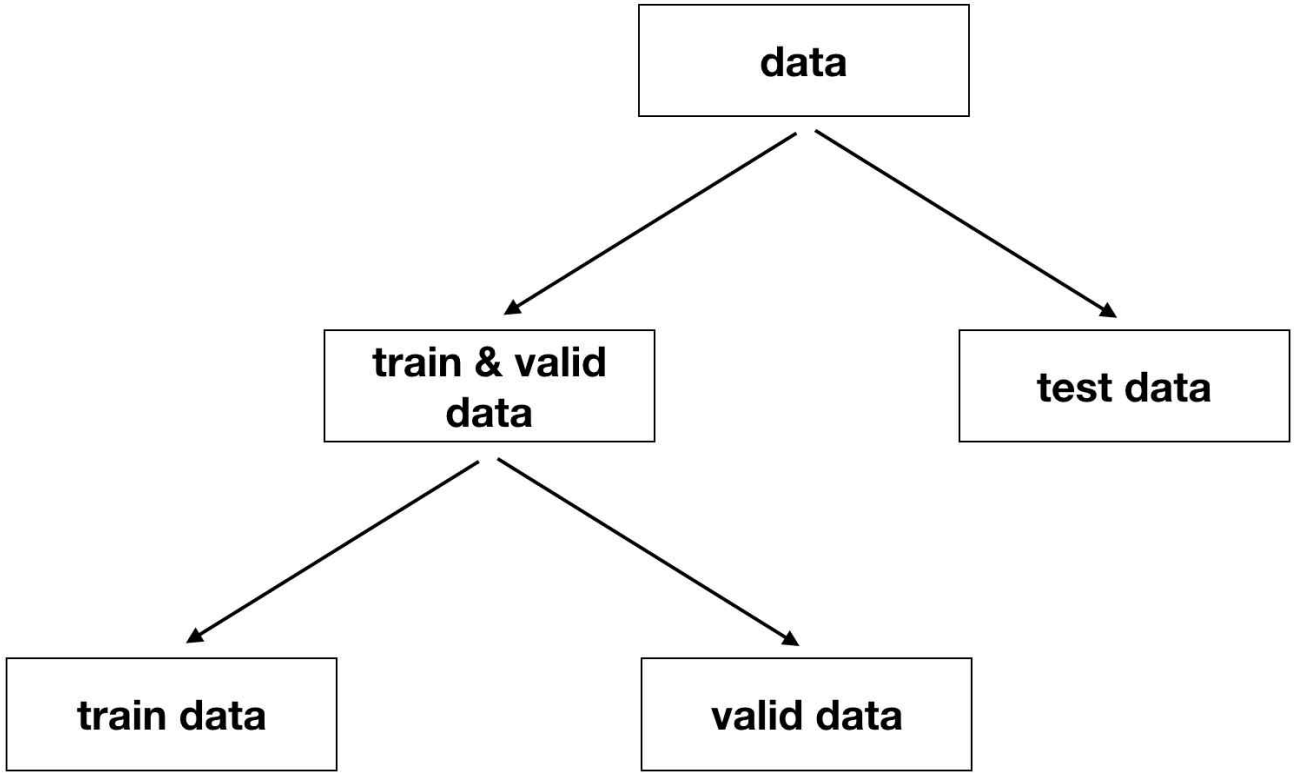
임의 분할 교차 검증

➢ 임의 분할 교차검증

- Train_size 만큼의 데이터로 훈련 세트를 만들고 test_size만큼의 데 이터로 테스트 세트를 n_splits 만큼 만들도록 분할
- Sklearn 에서는 ShuffleSplit 클래스 제공
- 10개의 데이터셋으로 나누고 5개는 훈련 세트, 2개는 테스트 세트 로 4번 반복하는 사례

교차 검증(Cross-Validation)

- Train Data : 모델을 학습하는데 사용하는 데이터
  - (모델이 알고 있는 학습할 데이터)
- Valid Data : 학습한 모델의 성능을 검증하는 데이터
  - (모델이 모르는 학습하지 않을 데이터, 모델 검증에 사용하는 데이터)
- Test Data : 학습한 모델로 예측할 데이터
  - (모델이 모르는 예측할 데이터)



5.3 평가 지표와 측정

➢ 평가 지표 선택 기준

- 최종 목표를 기억하라 – 응용의 고차원적 목표인 비즈니스 지표
  - 어떤 머신러닝 응용에서 특정알고리즘을 선택하여 나타난 결과를 비즈니스 임팩트(business impact)라고 함

➢ 이진 분류의 평가 지표에서 고려사항

- 에러의 종류
  - 암 진단 사례에서 양성(암), 음성(건강) 테스트 시에 거짓 양성(false positive) 와 거짓 음성(false negative) 오류 고려
  - 거짓 음성은 최대한 피해야 하며 거짓 양성은 중요도가 낮음
- 불균형(unbalanced) 데이터 셋
  - 두 종류의 오류(거짓 양성과 거짓 음성) 중 하나가 다른 것보다 훨씬 많을 때 더 중요

회귀의 평가지표

➢ 회귀의 평가지표

- 분류와 비슷하게 사용 가능
  - 타겟을 과대 예측한 것 대비 과소 예측한 것 분석
- 대부분의 응용은 score 메소드의 R2 계산으로 충분
- 가끔 평균 제곱 에러 또는 평균 절댓값 에러를 사용하여 비즈니스 결정

분류(Classification) 성능 평가 지표

- 정확도(Accuracy)
- 오차행렬(Confusion Matrix)
- 정밀도(Precision)
- 재현율(Recall)
- F1 스코어
- ROC AUC

모델 성능 평가

- **Accuracy**
  - 가장 많이 쓰이는것 : 모든 tp를 전부 더하고 전체 갯수로 나눈것
- **정밀도(Precision)**
  - TP / (TP + FP)
- **재현율(Recall)**
  - TP /(TP+ FN)
- **F1 score**
  - 데이터가 밸런스하지 않을 때 잘 예측함
  - 정밀도도 중요하고 재현율도 중요한데 둘 중 무엇을 쓸지 고민
  - 이 두 값을 조화평균 내서 수치로 나타낸 지표

1. Accuracy(정확도)

➢ 모든 데이터에 대해 클래스 라벨을 얼마나 잘 맞췄는지를 계산

Idx	y	y_hat
1	1	1
2	1	1
3	1	0
4	1	1
5	1	1
6	0	0
7	0	0
8	0	1
9	0	1
10	0	0

Error = 3
Error rate
= Misclassification rate = 3/10 = 0.3
Accuracy = 1 – misclassification rate = 0.7

2. Confusion Matrix(혼동행렬)

- 정확도로는 분류 모델의 평가가 충분하지 않을 수 있습니다. 예를 들어, 병이 있는 사람을 병이 없다고 판단하는 경우 Risk 가 높기 때문에 모델의 목적에 맞게 분류 모델을 평가하여야 합니다. 이때 사용되는 것이 Confusion Matrix 입니다.

2. Confusion Matrix(혼동행렬)

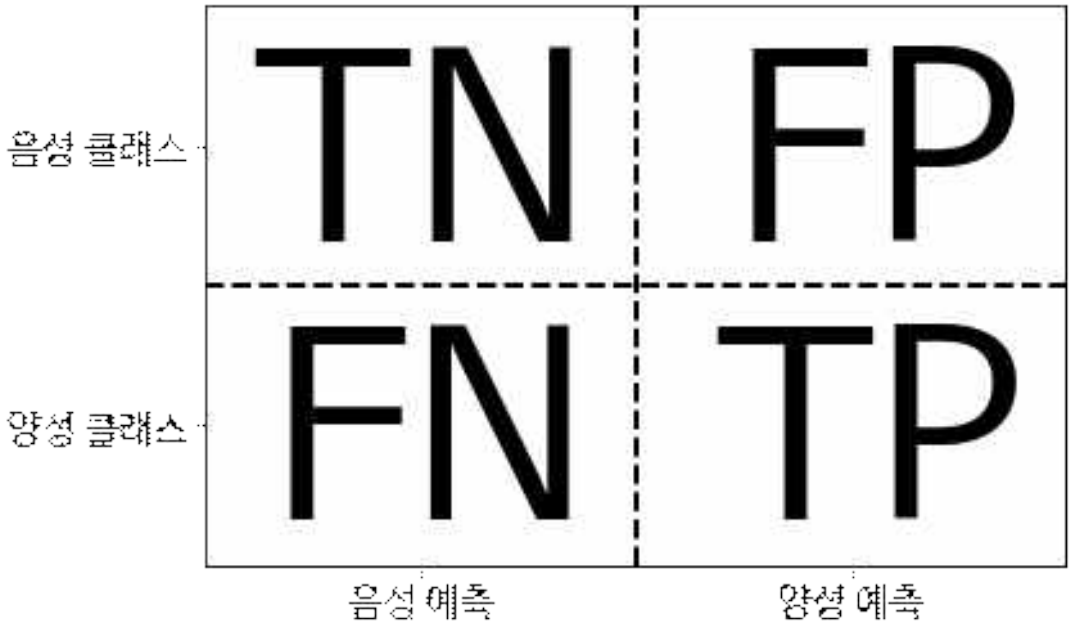
- 오차 행렬(confusion matrix)은 이진 분류 평가 결과를 나타낼 때 가장 널리 사용하는 방법 중 하나다.

  오차 행렬의 대각 행렬은 정확히 분류된 경우다. 다른 항목은 하 클래스의 샘플들이 다르 클래스로 잘못 분류된 경우가 얼마나 많은지를 알려준다. 이들을 축약해 FP, FN, TP(True Positive), TN(True Negative) 이라고 쓴다.

2. Confusion Matrix(혼동행렬)

- **정확도(Accuracy)**
  - (TN+TP) /(TN + FP + FN + TP)
- **정밀도(Precision)**
  - TP / (FP + TP)
- **재현율(Recall)**
  - TP / (FN + TP)
  - 암환자, 보험사기

2. Confusion Matrix(혼동행렬)



- TP (true positive) ➔ 맞는 것을 올바르게 예측한 것
- TN (true negative) 틀린 것을 올바르게 예측한 것
- FP (false positive) ➔ 틀린 것을 맞다고 잘못 예측한 것
- FN (false negative) ➔ 맞는 것을 틀렸다고 잘못 예측한 것

2. Confusion Matrix(혼동행렬)

	A	B	C	D
A	1	0	0	0
B	0	1	0	1
C	0	0	1	0
D	0	0	0	2

- 혼동 행렬은 얼마나 잘못(혹은 올바로) 예측했는지, 얼마나 '혼동 스러운지 ' 보여주는 행렬이다.

Data is balanced-accuracy

predictions →

	A	B	C	D
A	10	0	0	0
B	0	5	3	2
C	0	1	8	1
D	0	1	0	9

(10 + 5 + 8 + 9) / 40 = **0.8**

predictions →

	A	B	C	D
A	8	2	0	0
B	1	7	0	2
C	0	0	9	1
D	2	3	0	5

(8 + 7 + 9 + 5) / 40 = **0.725**

Data is imbalanced-accuracy? → f1 score

predictions →

	A	B	C	D
A	100	80	10	10
B	0	9	0	1
C	0	1	8	1
D	0	1	0	9

(100 + 9 + 8 + 9) / 230 = **0.547**

predictions →

	A	B	C	D
A	198	2	0	0
B	7	1	0	2
C	0	8	1	1
D	2	3	4	1

(198 + 1 + 1 + 1) / 230 = **0.87**

모델1,2의 정확도

0.98	A	B	C	D
A	995	5	0	0
B	8	0	1	1
C	10	0	0	1
D	0	1	9	0

모델 1 : 정확도 = (955 + 0 + 0 + 0) / 1030 = 96.6%

	A	B	C	D
A	700	100	100	100
B	0	9	1	0
C	0	0	9	1
D	0	1	0	9

모델 2 : 정확도 = (700 + 9 + 9 + 9) / 1030 = 70.5%

- **데이터 상에서 불균일 하게 분포된 경우 F1 점수를 사용하면 정확도 보다 나은 성능 평가 비교가 가능합니다.**

모델 성능 평가

- **모델 성능 평가**
  - TP (true positive)
  - TN (true negative)
  - 
  - FN (false negative)

    Accuracy
  - 가장 많이 쓰이는것 : 모든 tp를 전부 더하고 전체 갯수로 나눈것
- **정밀도(Precision)**
  - TP / (TP + FP)
- **재현율(Recall)**
  - TP /(TP+ FN)
- **F1 score**
  - 데이터가 밸런스 하지 않을 때 잘 예측함
  - 정밀도도 중요하고 재현율도 중요한데 둘 중 무엇을 쓸지 고민
  - 이 두 값을 조화평균 내서 수치로 나타낸 지표

모델1-F1 score

0.98	A	B	C	D	재현율
A	995	5	0	0	0.99
B	8	0	1	1	0
C	10	0	0	1	0
D	0	1	9	0	0
정밀도	0.98	0	0	0	

평균 정밀도 = (0.98 + 0 + 0 + 0) /4 = 0.245

평균 재현율 = (0.99 + 0 + 0+ 0)/4 = 0.2475

F1 점수= 2* 0.245 * 0.245 / (0.245 + 0.245)
= 0.121275 / 0.4905
= 0.246

모델2-F1 score

	A	B	C	D	재현율
A	700	100	100	100	0.7
B	0	9	1	0	0.9
C	0	0	9	1	0.9
D	0	1	0	9	0.9
정밀도	1	0.08	0.08	0.08	

평균 정밀도 = (1 + 0.08 + 0.08 + 0.08) / 4 = 0.31

평균 재현율 = (0.7 + 0.9 + 0.9 + 0.9) / 4 = 0.85

F1 점수= 2 * 0.31 * 0.85 / (0.31 + 0.85)
= 0.527 / 1.16
= 0.454

- **레이블이 불균일하게 분포된 경우 F1 점수는 한쪽 레이블에 치우치지 않는 레이블의 전체적인 성능에 대해 올바르게 평가하는 것을 확인할 수 있습니다.**

사이킷런의 정밀도, 재현율

- 정밀도는 precision_score()
- 재현율은 recall_score() 제공

업무에 따른 재현율과 정밀도의 상대적 중요도

- 재현율이 상대적으로 더 중요한 지표인 경우는 실제 Positive 양성인 데이터 예측을 Negative로 잘못 판단하게 되면 업무상 큰 영향이 발생하는 경우 : 암진단, 금융사기 판별

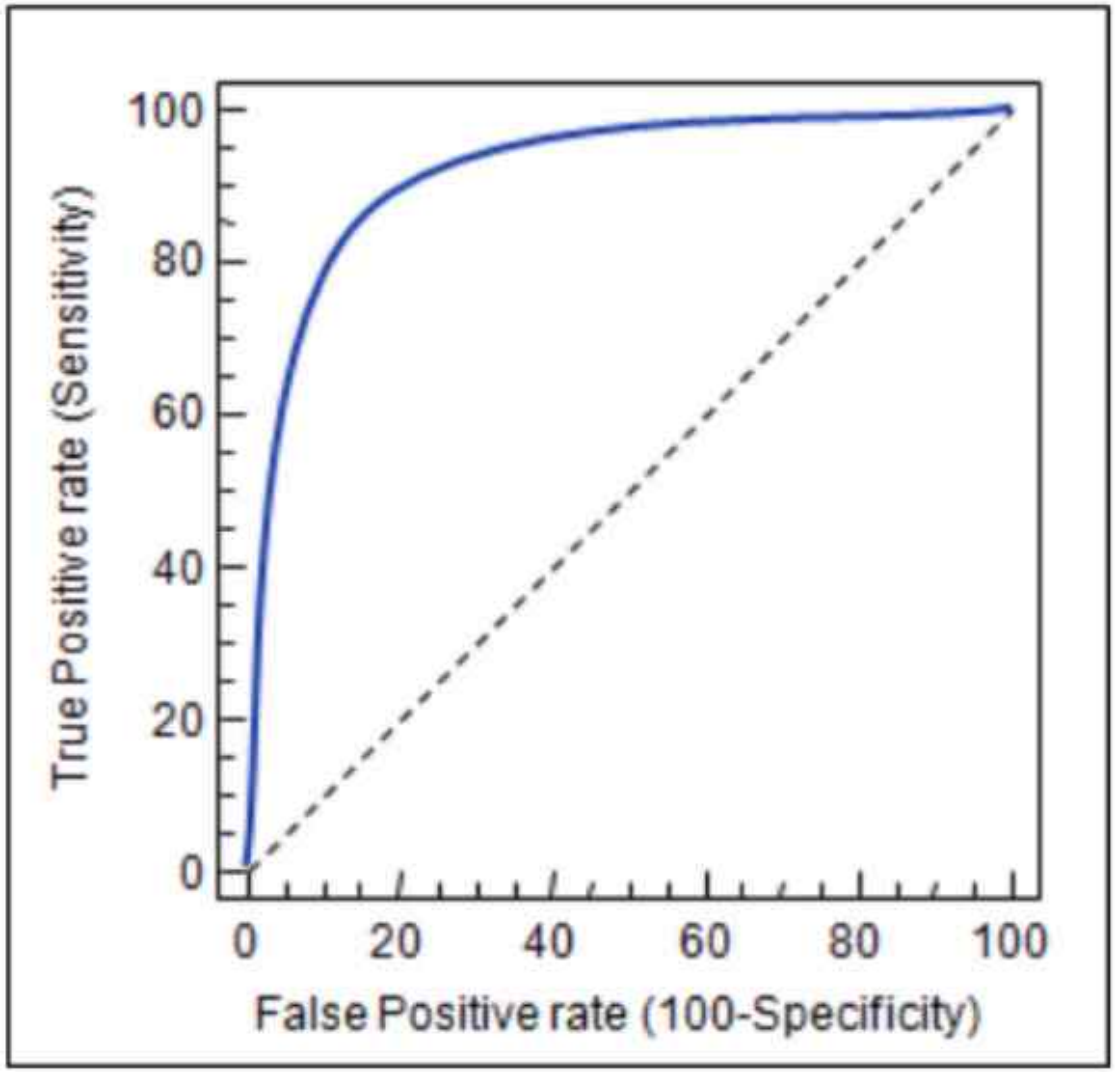
- 정밀도가 상대적으로 더 중요한 지표인 경우는 실제 Negative 음성인데 데이터 예측을 Positive 양성으로 잘못 판단하게 되면 업무상 큰 영향이 발생하는 경우 : 스팸메일

➢ 정밀도-재현율 곡선

- 모델의 분류 작업을 결정하는 임계값을 바꾸는 것은 해당 분류기 의 정밀도와 재현율의 상충 관계를 조정하는 일
- 양성 샘플의 실수(FN)을 10%보다 작게 하여 90% 이상 재현율
  - 임계값은 비즈니스 목표에 따라 결정됨 – 운영 포인트(operating point)
  - 이 임계값을 유지하면서 적절한 정밀도를 내는 모델을 만드는 일이 어려움
- 이럴 때 정밀도-재현율 곡선(precision-recall curve)를 그려 정밀도와 재현율의 모든 점을 살펴봄
- Sklearn.metrics 모듈에서 정밀도-재현율 곡선 함수 제공

3. ROC 곡선과 AUC

- ROC Curve(Receiver-Operation Characteristic curve, 수신자 조작 특성 곡선)과 이에 기반한 AUC 스코어는 이진 분류의 예측 성능 측정에서 중요하게 사용되는 지표
- 의학 분야에서 많이 사용되지만, 머신러닝의 이진 분류 모델의 예측 성능을 판단하는 중요한 평가 지표
- 민감도와 특이도가 서로 어떤 관계를 가지며 변하는지를 2차원 평면상에 표현
- ROC Curve가 그려지는 곡선을 의미하고, AUC(Area Under Curve)는 ROC Curve의 면적을 뜻합니다. AUC 값이 1에 가까울수록 좋은 모델을 의미합니다.



피마 인디언 데이터 분석하기

피마 인디언 옛 모습

미국 남서부에 살고 있는 피마 인디언은 1950년대까지만 해도 비만인이 한 사람도 없는 민족 지금은 전체 부족의 60%가 당뇨, 80%가 비만으로 고통받고 있습니다.

피마인디언 당뇨병 예측

- 피마 인디언 당뇨병 데이터 세트를 이용해 당뇨병 여부를 판단하는 머신러닝 예측 모델을 수립하고, 평가 지표를 적용
- kaggle.com
- uciml
- https://www.kaggle.com/uciml/pima-indians-diabetes-database

피마 인디언 데이터 분석하기

- 비만이 유전 및 환경, 모두의 탓이라는 것을 증명하는 좋은 사례가 바로 미국 남서부에 살고 있는 피마 인디언의 사례

속성 / 클래스

	정보 1	정보 2	정보 3	…	정보 8	당뇨병 여부
1번째 인디언	6	148	72	…	50	1
2번째 인디언	1	85	66	…	31	0
3번째 인디언	8	183	64	…	32	1
…	…	…	…	…	…	…
768번째 인디언	1	93	70	…	23	0

샘플

표 피마 인디언 데이터의 샘플, 속성, 클래스 구분

피마 인디언 데이터 분석하기

- 샘플 수: 768
- 속성: 8
  - 정보 1 (pregnant): 과거 임신 횟수
  - 정보 2 (plasma): 포도당 부하 검사 2시간 후 공복 혈당 농도(mm Hg)
  - 정보 3 (pressure): 확장기 혈압(mm Hg)
  - 정보 4 (thickness): 삼두근 피부 주름 두께(mm)
  - 정보 5 (insulin): 혈청 인슐린(2-hour, mu U/ml)
  - 정보 6 (BMI): 체질량 지수(BMI, weight in kg/(height in $m)^2$)
  - 정보 7 (pedigree): 당뇨병 가족력
  - 정보 8 (age): 나이
- 클래스: 당뇨(1), 당뇨 아님(0)

추천 시스템 (Recommendation)

July 26, 2023

추천(Recommendation)

- 아마존 등과 같은 전자상거래 업체로부터 넷플릭스, 유투브, 애플 뮤직 등 콘텐츠 포털까지 추천 시스템을 통해 사용자의 취향을 이해하고 맞춤 상품과 콘텐츠 제공해 조금이라도 오랫동안 자기 사이트에 고객이 머무르게 하기 위해 전력을 기울이고 있다.

추천 시스템 방식

콘텐츠 기반 필터링
Content Based Filtering

협업 필터링
Collaborative Filtering

추천 시스템은 이들 방식중 1가지를 선택하거나 이들을 결합하여 hybrid 방식으로 사용

(예: Content Based + Collaborative Filtering)

관련연구 및 제안 사항

- **정보의 홍수 속에서 각 개인의 선호도에 따라 적절한 콘텐츠를 추천해 주는 서비스에 매력을 느낄 것이다.**
  - 콘텐츠 : 온라인 쇼핑몰-물품, 책이나 비디오-창작물, 매일 먹는 음식의 종류
- **관련 연구**

구분	사용자/아이템 기반의 협업 필터링	콘텐츠 기반의 추천
내용	▪ 나와 선호도가 유사한 사용자들을 기반으로 내가 접하지 않았던 아이템들에 대한 선호도를 예측하는 기법	▪ 평소 자주 접하던 아이템을 분석하여 유사한 아이템들을 추천하는 방법 ▪ 특정 콘텐츠와 유사한 성질을 가지는 콘텐츠를 검색- 전통적인 정보 검색에 근거
장점	▪ 예상치 못한 아이템들을 추천 받을 수 있다.	▪ 사용자가 선호하는 아이템들을 선별하여 추천해 주기 때문에 일정수준 이상의 사용자 경험(UX) 만족도 보장
단점	▪ 선호도 측정을 위한 데이터를 축적해야함 ▪ 새로운 사용자나 콘텐츠에 대해 추천 결과에 반영시킬 수 없는 콜드 스타트 문제	▪ 같은 주제를 가지는 뉴스만 추천 : 피로도와 지루함을 쉽게 느낌

콘텐츠 기반 필터링 – Contents Based Filtering

➢ 프로세스

- 콘텐츠에 대한 여러 텍스트 정보들을 피쳐 벡터화
- 코사인 유사도로 콘텐츠별 유사도 계산
  콘텐츠 별로 가중 평점을 계산
- 유사도가 높은 콘텐츠 중에 평점이 좋은 콘텐츠 순으로 추천

협업 필터링의 유형

- **최근접 이웃 기반(Nearest Neighbor)**
  - 사용자 기반(User-user CF)
  - 아이템 기반(Item-item CF)

- **잠재 요인 기반(Latent Factor)**
  - 행렬 분해 기반(Matrix Factorization)

Questions or Comments?