머신러닝 - 평가

2021

0. 개요

❖ 성능 평가 지표의 종류

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

1. 정확도(Accuracy)

❖ 정의

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

❖ 왜곡하는 경우

- 불균형한 레이블 값 분포에서 ML 모델의 성능을 평가하는 경우
- 예)
  - 100개의 데이터, 90개의 레이블이 0, 10개의 레이블이 1인 경우
  - 예측을 무조건 0로 하는 경우 → 90%의 정확도

1. 정확도(Accuracy)

❖ 왜곡하는 사례

- 타이타닉의 경우 – 단순히 Sex 피쳐만 보고 예측해도 상당한 정확도를 가짐

```
import numpy as np
from sklearn.base import BaseEstimator

class MyDummyClassifier(BaseEstimator):
    # fit( ) 메소드는 아무것도 학습하지 않음.
    def fit(self, X, y=None):
            pass

    # predict( ): 단순히 Sex feature가 1이면 0, 그렇지 않으면 1로 예측함.
    def predict(self, X):
        pred = np.zeros((X.shape[0], 1))
        for i in range(X.shape[0]):
            if X['Sex'].iloc[i] == 1:
                pred[i] = 0
            else:
                pred[i] = 1

        return pred
```

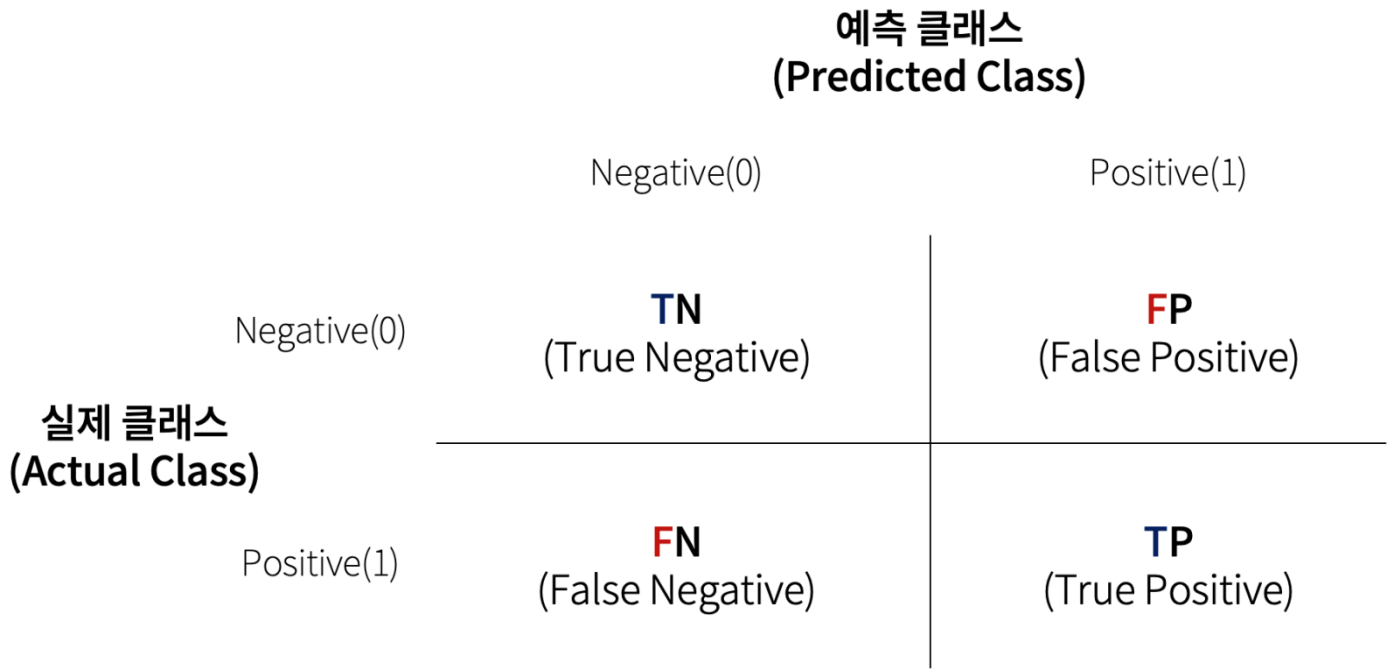
1. 정확도(Accuracy)

❖ 왜곡하는 사례

- 타이타닉 사례 예측 → 78.8% 정확도

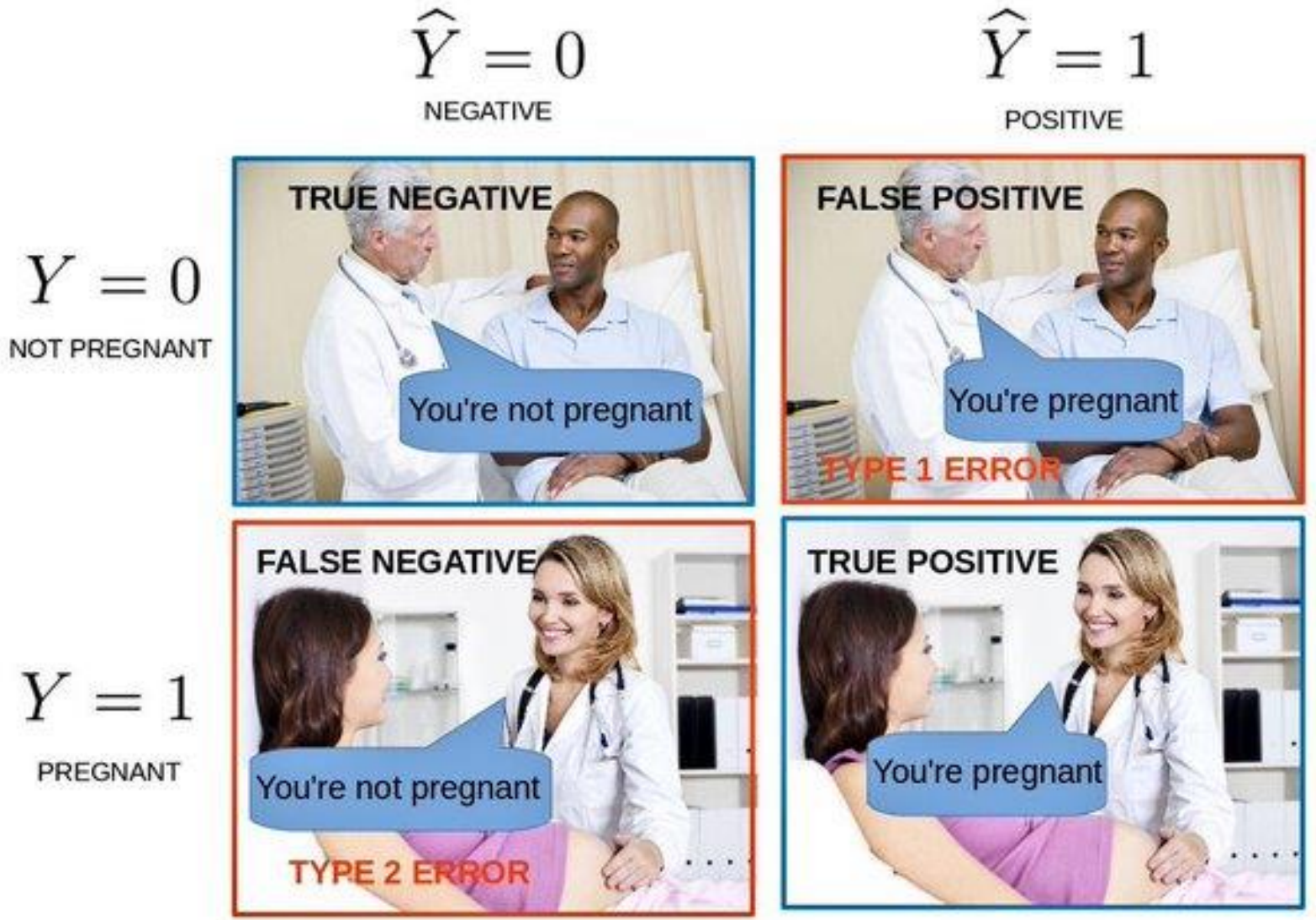
```
from sklearn.model_selection import train_test_split

# 원본 데이터를 재로딩, 데이터 가공, 학습데이터/테스트 데이터 분할.
titanic_df = pd.read_csv('../00.data/titanic/train.csv')
y_titanic_df = titanic_df['Survived']
X_titanic_df= titanic_df.drop('Survived', axis=1)
X_titanic_df = transform_features(X_titanic_df)
X_train, X_test, y_train, y_test=train_test_split(X_titanic_df, y_titanic_df,
                                                  test_size=0.2, random_state=0)

# 앞에서 생성한 Dummy Classifier를 이용하여 학습/예측/평가 수행.
myclf = MyDummyClassifier()
myclf.fit(X_train, y_train)

mypredictions = myclf.predict(X_test)

from sklearn.metrics import accuracy_score
pred = accuracy_score(y_test, mypredictions)
print(f'Dummy Classifier의 정확도: {pred:.4f}')
```

2. 오차 행렬(Confusion Matrix)



- TN: 예측값을 0으로 예측했고, 실제값 역시 0인 경우
- FP: 예측값을 1로 예측했고, 실제값은 0인 경우
- FN: 예측값을 0으로 예측했고, 실제값은 1인 경우
- TP: 예측값을 1로 예측했고, 실제값 역시 0인 경우

2. 오차 행렬(Confusion Matrix)



출처: https://twitter.com/bearda24

2. 오차 행렬(Confusion Matrix)

❖ MyFakeClassifier 사례

```
from sklearn.metrics import confusion_matrix

# 예측 결과인 fakepred와 실제 결과인 y_test의 Confusion Matrix출력
confusion_matrix(y_test, fakepred)
```

		예측 클래스	
		Negative	**Positive**
실제 클래스	**Negative**	**TN** 예측: Negative(7이 아닌 Digit) 405개 실제: Negative(7이 아닌 Digit)	**FP** 예측: Positive(Digit 7) 0 실제: Negative(7이 아닌 Digit)
	Positive	**FN** 예측: Negative(7이 아닌 Digit) 45개 실제: Positive(Digit 7)	**TP** 실제: Positive(Digit 7) 0 실제: Positive(Digit 7)

3. 정밀도와 재현율

❖ 정밀도(Precision)

- **Precision = TP / (FP + TP)**
- Positive 예측 성능을 더욱 정밀하게 측정하기 위한 평가 지표
- 실제 Negative 음성 데이터를 Positive 양성으로 잘못 판단하게 되면 업무상 큰 영향이 발생하는 경우

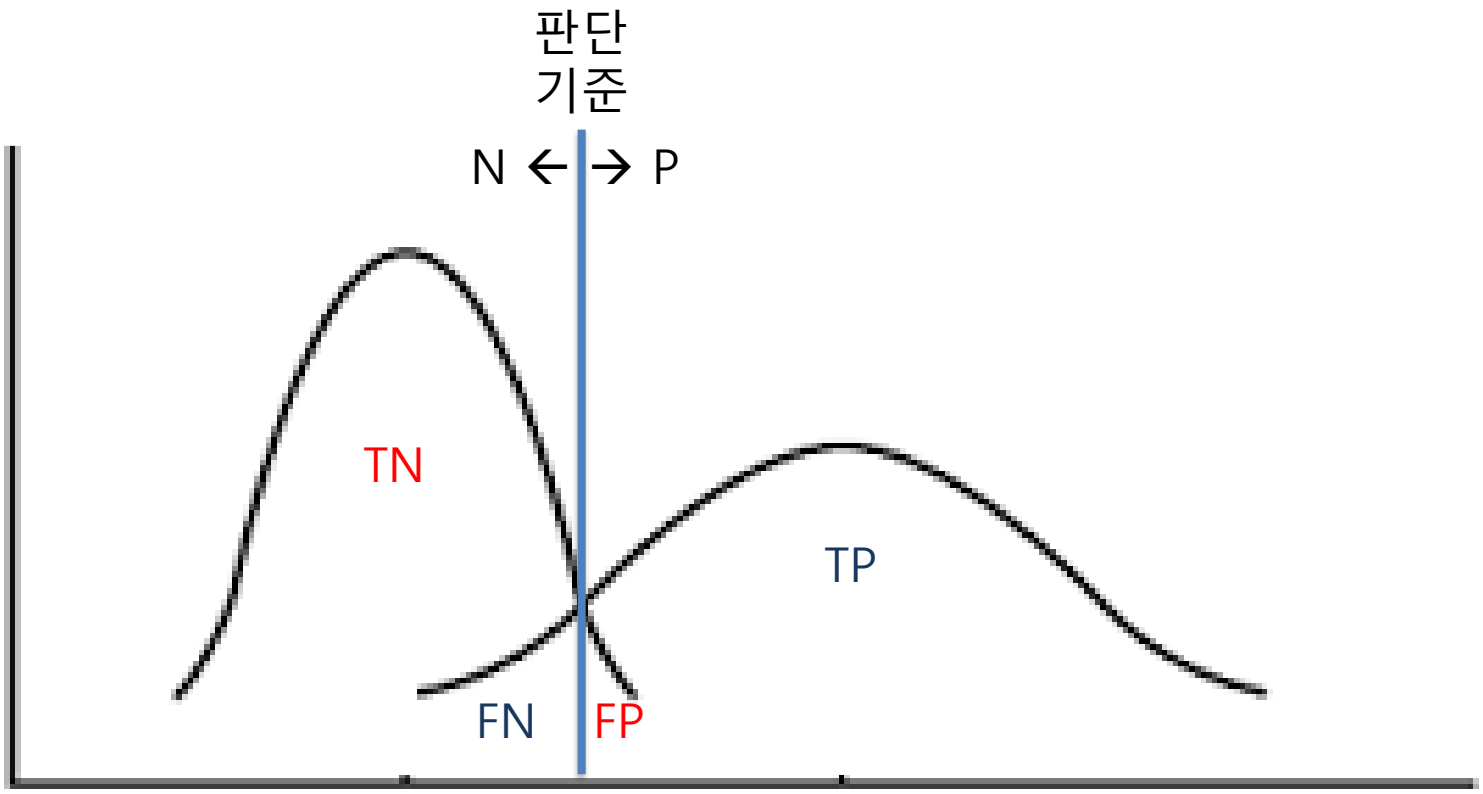
  예) 판사의 유죄 판결 (무죄 추정의 원칙)

❖ 재현율(Recall)

- **Recall = TP / (FN + TP)**
- 민감도(Sensitivity) 또는 TPR(True Positive Rate)라고도 불림
- 실제 Positive 양성 데이터를 Negative 음성으로 잘못 판단하게 되면 업무상 큰 영향이 발생하는 경우

  예) 의사의 암 진단 (추후에 정밀 진단을 통해 암 진위여부를 판단할 수 있음)

3. 정밀도와 재현율



정밀도: 내가 내린 양성 판단 중에 … (유죄 판결)

재현율: 실제 있는 양성 중에 … (암 1차 검사)

4. F1 스코어

- 재현율과 정밀도의 조화 평균
- 정밀도와 재현율이 어느 한쪽으로 치우치지 않는 경우 높은 수치를 나타냄

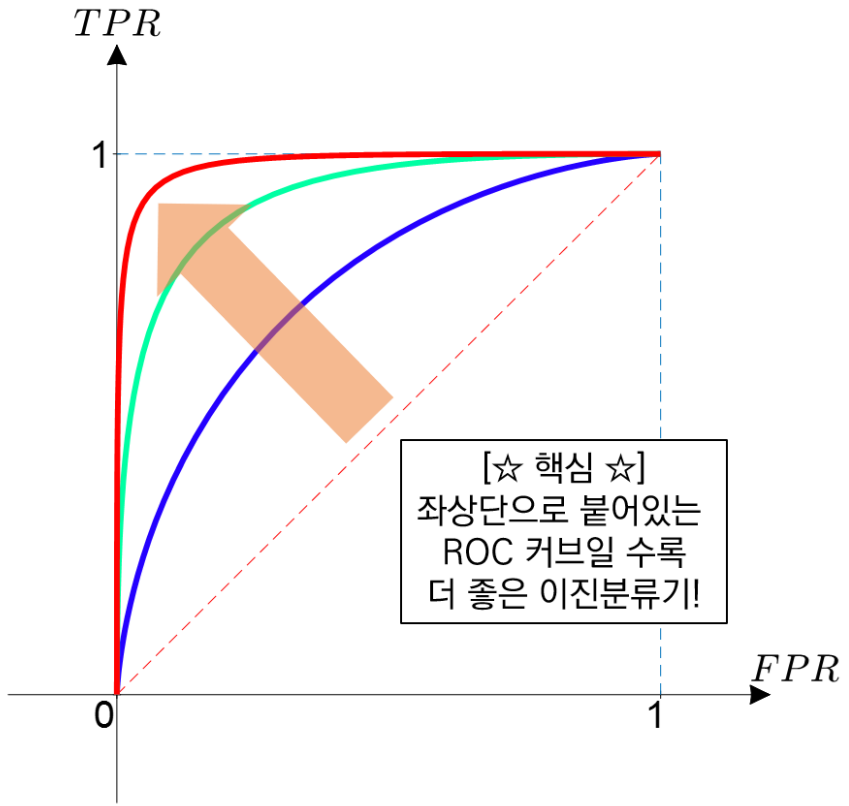
$$F1 = \frac{2}{\frac{1}{recall} + \frac{1}{precision}} = 2 \times \frac{precision \cdot recall}{precision + recall}$$

```
from sklearn.metrics import f1_score
f1 = f1_score(y_test, pred)
print('F1 스코어: {0:.4f}'.format(f1))
```

5. ROC 곡선과 AUC

❖ ROC(Receiver Operation Characteristic) 곡선

- 수신자 판단 곡선
- FPR(False Positive Rate)가 변할 때 TPR(True Positive Rate)이 어떻게 변하는지를 나타내는 곡선
- TNR(True Negative Rate) - 특이성
- FPR = 1 - TNR
- roc_curve() API 제공



❖ AUC(Area Under Curve)

- ROC 곡선 밑의 면적
- 1에 가까울수록 좋은 수치

6. 피마 인디언 당뇨병 예측 사례

- 북아메리카 피마 지역 원주민의 Type-2 당뇨병 결과 데이터
- 캐글 사이트(https://www.kaggle.com/uciml/pima-indians-diabetes-database)
- Feature 설명
  - Pregnancies: 임신 횟수
  - Glucose: 포도당 부하 검사 수치
  - BloodPressure: 혈압
  - SkinThickness: 팔 삼두근 뒤쪽의 피하지방 측정값
  - Insulin: 혈청 인슐린
  - BMI: 체질량 지수(체중/키)
  - DiabetesPedigreeFunction: 당뇨 내력 가중치 값
  - Age: 나이
  - Outcome: 클래스 결정 값