ML,DL

[ML/DL] python 을 통한 분류(classification) 성능평가지표 사용법(Accuracy,Precision,Recall,F1 Scroe)

2020. 11. 2. 02:49 수정 삭제 공개

분류(classification) 성능평가지표

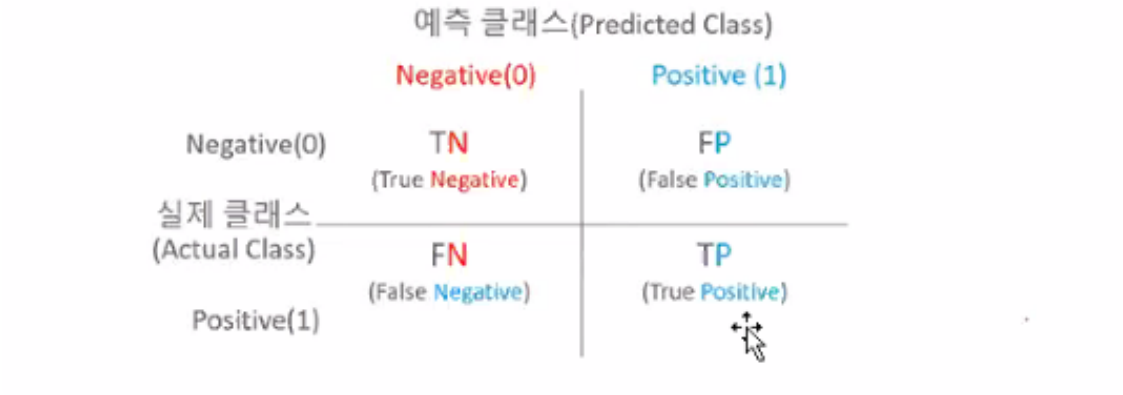
#metrics 서브 패키지

confusion_matrix(answer, prediction) == 오차 행렬

=> 오차 행렬은 어떠한 유형의 예측 오류가 발생하고 있는지를 함께 나타내는 지표이다. 즉 분류의 성능을 평가하는 행렬이다.

오차 행렬(Confusion Matrix)

오차 행렬은 이진 분류의 예측 오류가 얼마인지와 더불어 어떠한 유형의 예측 오류가 발생하고 있는지를 함께 나타내는 지표 입니다



이진 분류표

제품을 생산하는 제조공장에서 품질 테스트를 실시하여 불량품을 찾아내고 불량품을 공장으로 돌려보낸다(recall)

품질 테스트의 결과가 양성 -> 불량품 예측한 것이고

음성 -> 정상제품이라고 예측한 것이다.

#TP : 불량품(P)을 불량품(P)으로 정확하게 예측 - True

#TN : 정상제품(N)을 정상제품(N)으로 정확하게 예측 - True

#FP : 정상제품(N)을 불량품(P)이라고 예측 - False

#FN : 불량품(P)을 정상제품(N)이라고 예측 - False

```
y_true = [1,0,1,1,0,1]
y_pred = [0,0,1,1,0,1]
confusion_matrix(y_true,y_pred)
```

```
array([[2, 0],
       [1, 3]], dtype=int64)
```

2 => 0을 0으로 예측한 것 2 개

0 => 0을 1로 예측한 것

1 => 1을 0으로 예측한 것

1 => 1을 1로 예측한 것

 # 예를 통한 오차 행렬 정리

		predict		
		Positive	Negative	합계
acutal	Positive	900	54	954
	Negative	100	87	187
	합계	1000	141	1141

		predict	
		Positive	Negative
acutal	Positive	True Positive	False Negative
	Negative	False Positive	True Negative

TP의 경우 1이라고 예측했는데 실제도 1(정답)
TN의 경우 0이라고 예측했는데 실제도 0(정답)
FP의 경우 1이라고 예측했는데 실제로는 0(오답)
FN의 경우 0이라고 예측했는데 실제로는 1(오답)

맞게 검출한 비율	Accuracy	**T P + T N / (T P + T N + F P + F N)**	0.8650307
P로 예측한 것 중 실제 P의 비율	Precision	**T P / T P + F P**	0.9
실제 P를 P로 예측	Sensitivity = Recall	**T P / T P + F N**	0.9433962
실제 N을 N로 예측	Specificity	**T N / F P + T N**	0.4652406
오류율	Error	**F N + F P / T P + F N + F P + T N**	0.1349693
정밀도와 재현율의 평균	F1 score	**2 * (precision * recall) / (precision + recall)**	0.9211873

재현율(recall_socre)이 상대적으로 더 중요한 지표는 실제 positive 양성인 데이터 예측을 Negative로 잘못 판단하게 되면 업무상 큰 영향이 발생하는 경우
ex) 암 진단, 금융사기 판별

정밀도(precision_score)가 상대적으로 더 중요한 지표인 경우는 실제 Negative 음성인데 데이터 예측을 positive양성으로 잘못 판단하게 되면 업무상 큰 영향이 발생하는 경우 :
ex) 스팸메일

예시)

```
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings('ignore')


### fit() 메서드는 아무 것도 수행하지 않고, predict()는 Sex 피처가 1이면 0, 그렇지 않으면 1로 예측
from sklearn.base import BaseEstimator

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

```
    # predict 메서드는 단순히 Sex 피처가 1이면 0, 아니면 1로 예측
    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

titanic = pd.read_csv('titanic_train.csv')
titanic.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

```
titanic.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
PassengerId    891 non-null int64
Survived       891 non-null int64
Pclass         891 non-null int64
Name           891 non-null object
Sex            891 non-null object
Age            714 non-null float64
SibSp          891 non-null int64
Parch          891 non-null int64
Ticket         891 non-null object
Fare           891 non-null float64
Cabin          204 non-null object
Embarked       889 non-null object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.6+ KB
```

```
titanic_label = titanic['Survived']
titanic_label.head()
titanic_feature_df = titanic.drop(['Survived'],axis=1)
titanic_feature_df.head()
```

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

```
## Null 처리 함수
def fillna(df):
```

```
    df['Age'].fillna(df['Age'].mean(), inplace=True)
    df['Cabin'].fillna('N', inplace=True)
    df['Embarked'].fillna('N', inplace=True)
    df['Fare'].fillna(0, inplace=True)
    return df

## 머신러닝에 불필요한 피처 제거
def drop_features(df):
    df.drop(['PassengerId', 'Name', 'Ticket'], axis=1, inplace=True)
    return df

## Label Encoding 수행
def format_features(df):
    df['Cabin'] = df['Cabin'].str[:1]
    features = ['Cabin', 'Sex', 'Embarked']
    for feature in features:
        le = LabelEncoder()
        le.fit(df[feature])
        df[feature] = le.transform(df[feature])
    return df

## 앞에서 실행한 Data Preprocessing 함수 호출
def transform_features(df):
    df = fillna(df)
    df = drop_features(df)
    df = format_features(df)
    return df


titanic_feature_df = transform_features(titanic_feature_df)


X_train, X_test, y_train, y_test = train_test_split(titanic_feature_df,titanic_label,test_size=.2,random_state=10)


dummy_model = MyDummyClassifier()


dummy_model.fit(X_train,y_train)

y_pred = dummy_model.predict(X_test)
print('accuracy {}'.format(accuracy_score(y_test,y_pred)))
```

```
accuracy 0.8212290502793296
```

```
from sklearn.metrics import recall_score, precision_score
```

```
def display_eval(y_test,y_pred):
    confusion = confusion_matrix(y_test,y_pred)
    accuracy = accuracy_score(y_test,y_pred)
    precision = precision_score(y_test,y_pred)
    recall = recall_score(y_test,y_pred)
    print(confusion)
    print('*'*50)
    print()
    print('정확도:{},정밀도 : {}, 재현율:{}'.format(accuracy,precision,recall))



from sklearn.linear_model import LogisticRegression
# 로지스틱 회귀
lr_model = LogisticRegression()
lr_model.fit(X_train,y_train)
prediction = lr_model.predict(X_test)
display_eval(y_test,prediction)
```

```
[[104  13]
 [ 17  45]]
**************************************************

정확도:0.8324022346368715,정밀도 : 0.7758620689655172, 재현율:0.7258064516129032
```

```
print('accuracy : ',(104+45)/(104+13+17+45))
print('recall : ',(47)/(47+15))
print('precision : ',(45)/(47+16))
```

```
accuracy :  0.8324022346368715
recall :  0.7580645161290323
precision :  0.7142857142857143
```

```
# [실습] - 유방암 관련 데이터 - 정확, 재현율(실제 P를 N으로 예측해서는 안된다.)
# 재현율은 실제 양성을 양성으로 예측한 비율이므로 높을수록 좋은 성능모형이라 판단 할 수 있다.
```

```
from sklearn.datasets import load_breast_cancer
from sklearn.ensemble import RandomForestClassifier

from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings('ignore')


cancer = load_breast_cancer()
print(type(cancer))
```

```
<class 'sklearn.utils.Bunch'>
```

```
cancer.keys()
```

```
dict_keys(['data', 'target', 'target_names', 'DESCR', 'feature_names', 'filename'])
```

```
cancer_df = pd.DataFrame(data=cancer.data, columns=cancer.feature_names)

cancer_df.head()
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst radius	worst texture	worst perimeter	worst area	wors smoothnes
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	25.38	17.33	184.60	2019.0	0.162
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	24.99	23.41	158.80	1956.0	0.123
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	23.57	25.53	152.50	1709.0	0.144
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	14.91	26.50	98.87	567.7	0.209
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	22.54	16.67	152.20	1575.0	0.137

```
cancer_df['target'] = cancer.target
cancer_df.head()cancer_df['target'] = cancer.targetcancer_df.head()
```

```
cancer_df['target'] = cancer.target
cancer_df.head()
```

mean ctness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area	worst smoothness	worst compactness	worst concavity	worst concave points	worst symmetry	worst fractal dimension	target
.27760	0.3001	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0	0.1622	0.6656	0.7119	0.2654	0.4601	0.11890	0
.07864	0.0869	0.07017	0.1812	0.05667	...	23.41	158.80	1956.0	0.1238	0.1866	0.2416	0.1860	0.2750	0.08902	0
.15990	0.1974	0.12790	0.2069	0.05999	...	25.53	152.50	1709.0	0.1444	0.4245	0.4504	0.2430	0.3613	0.08758	0
.28390	0.2414	0.10520	0.2597	0.09744	...	26.50	98.87	567.7	0.2098	0.8663	0.6869	0.2575	0.6638	0.17300	0
.13280	0.1980	0.10430	0.1809	0.05883	...	16.67	152.20	1575.0	0.1374	0.2050	0.4000	0.1625	0.2364	0.07678	0

```
# 분류 학습기 생성
# 팍습 및 평가(교차검증)
# 평가지표에 대한 평균값
# accuracy, precision, recall

X_train , X_test , y_train, y_test = train_test_split(cancer_data , cancer_labe ,
                                    test_size=0.2,
                                    random_state=20)



lr = LogisticRegression()
lr.fit(X_train,y_train)
lr_pred = lr.predict(X_test)
print( '예측 정확도 : {0:.2f}'.format( accuracy_score(y_test, lr_pred)))


from sklearn.model_selection import cross_val_score, cross_validate,KFold
from sklearn.metrics import make_scorer
fold = KFold(n_splits=20,random_status=1,shuffle=True)

scoring = {
   'accuracy' :,
   'precision' : ,
   'recall' :
}
result = cross_validate(lr,cancer_data,cancer_labe,scoring=scoring , cv=fold)
print(result.key())
```

```
dict_keys(['fit_time', 'score_time', 'test_accuracy', 'test_precision', 'test_recall', 'test_f1_score'])
```

이렇게 result의 key값을 보면 여러 값들이 들어가 있다.

```
# 평가지표에 대한 평균값을 구해보자
print('accuracy',np.round(result['test_accuracy'].mean(),2))
print('precision',np.round(result['test_precision'].mean(),2))
```

```
print('recall',np.round(result['test_recall'].mean(),2))
print('f1_score',np.round(result['test_f1_score'].mean(),2))
```

```
accuracy 0.95
precision 0.96
recall 0.97
f1_score 0.96
```

이 값을 key를 통해 값을 빼면 위와 같이 나오게 된다.

'ML,DL' 카테고리의 다른 글

[ML/DL] python 으로 구현하는 ROC곡선과 AUC

[ML/DL] 정밀도와 재현율의 트레이드 오프 정의와 구현

[ML/DL] python 을 통한 분류(classification) 성능평가지표 사용법(Accuracy,P...

[ML/DL] python 을 통한 교차검증 (k -Fold , stratifiedkFold)

[ML/DL] python 을 통한 결측값 확인 및 결측치 처리 방법

[ML/DL] 데이터 인코딩 - Label Encoding / One-hot Encoding/ dummies

Accuracy F1 Score precision recall 분류 성능평가지표 성능평가지표

나아무늘보

혼자 끄적끄적하는 블로그 입니다.