In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_score , recall_score, roc_auc_score
from sklearn.metrics import f1_score, confusion_matrix, precision_recall_curve,roc_curve
```

In [41]:

```
df = pd.read_pickle('titanic_.pkl')
```

In [82]:

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split

y = df['Survived']
X = df.drop('Survived',axis =1)

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.2, random_state=10)
dt_clf = DecisionTreeClassifier()
dt_clf.fit(X_train,y_train)
pred = dt_clf.predict(X_test)
print('정확도: {0:.4f}'.format(accuracy_score(y_test,pred)))
print()
```

```
정확도: 0.8547
```

In [70]:

```
from sklearn.linear_model import LogisticRegression


y = df['Survived']
X = df.drop('Survived',axis =1)

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.2, random_state=11)
lr_clf = LogisticRegression()
lr_clf.fit(X_train,y_train)
pred = lr_clf.predict(X_test)
print('정확도: {0:.4f}'.format(accuracy_score(y_test,pred)))
```

```
정확도: 0.8659
```

In [95]:

```
from sklearn.ensemble import RandomForestClassifier

y = df['Survived']
X = df.drop('Survived',axis =1)

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.2, random_state=11)
rf_model = RandomForestClassifier()
rf_model.fit(X_train,y_train)
pred = rf_model.predict(X_test)
print('정확도: {0:.4f}'.format(accuracy_score(y_test,pred)))
```

```
정확도: 0.8380
```

## 모델별 accuracy,confusion matrix,precision, recall, roc auc score 평가

In [94]:

```
# dt

from sklearn.preprocessing import Binarizer
from sklearn.metrics import roc_auc_score

y = df['Survived']
X = df.drop('Survived',axis =1)

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.2, random_state=10)
dt_clf = DecisionTreeClassifier()
dt_clf.fit(X_train,y_train)
pred = dt_clf.predict(X_test)
print('정확도: {0:.4f}'.format(accuracy_score(y_test,pred)))
print()

def get_clf_eval(y_test,pred):
    confusion = confusion_matrix(y_test,pred)
    accuracy = accuracy_score(y_test,pred)
    precision = precision_score(y_test,pred)
    recall = recall_score(y_test,pred)
    f1 = f1_score(y_test,pred)
    roc_score=roc_auc_score(y_test,pred)
    print('오차행렬')
    print(confusion)
    print()
    print('정확도:{0:.4f}, 정밀도:{1:.4f}, 재현율:{2:.4f}, ₩
    f1 score:{3:.4f}, ROC AUC:{4:.4f}'.format(accuracy,precision,recall,f1, roc_score))



get_clf_eval(y_test, pred)
```

```
정확도: 0.8547

오차행렬
[[110   7]
 [ 19  43]]

정확도:0.8547, 정밀도:0.8600, 재현율:0.6935,     f1 score:0.7679, ROC AUC:0.8169
```

In [93]:

```
#lr

from sklearn.linear_model import LogisticRegression


y = df['Survived']
X = df.drop('Survived',axis =1)

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.2, random_state=11)
lr_clf = LogisticRegression()
lr_clf.fit(X_train,y_train)
pred = lr_clf.predict(X_test)
print('정확도: {0:.4f}'.format(accuracy_score(y_test,pred)))
print()

get_clf_eval(y_test, pred)
```

```
정확도: 0.8659

오차행렬
[[108  10]
 [ 14  47]]

정확도:0.8659, 정밀도:0.8246, 재현율:0.7705,     f1 score:0.7966, ROC AUC:0.8429
```

In [92]:

```
#rf
from sklearn.ensemble import RandomForestClassifier

y = df['Survived']
X = df.drop('Survived',axis =1)

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.2, random_state=11)
rf_model = RandomForestClassifier()
rf_model.fit(X_train,y_train)
pred = rf_model.predict(X_test)
print('정확도: {0:.4f}'.format(accuracy_score(y_test,pred)))

print()

get_clf_eval(y_test, pred)
```

```
정확도: 0.8324

오차행렬
[[109   9]
 [ 21  40]]

정확도:0.8324, 정밀도:0.8163, 재현율:0.6557,     f1 score:0.7273, ROC AUC:0.7897
```

## 하이퍼파라미터 튜닝

In [91]:

```
from sklearn.model_selection import GridSearchCV


X_train, X_test, y_train, y_test = train_test_split(X,y,
                                                    test_size=0.2, random_state=11)
dt_clf = DecisionTreeClassifier()
parameters = {'max_depth':[1,2,3], 'min_samples_split':[2,3]}

grid_dtree = GridSearchCV(dt_clf, param_grid=parameters, cv=3, refit=True)
grid_dtree.fit(X_train,y_train)

print('GridSearchCV 최적 파라미터:',grid_dtree.best_params_)
print('GridSearchCV 최고 정확도: {0:4f}'.format(grid_dtree.best_score_))

estimator = grid_dtree.best_estimator_
pred = estimator.predict(X_test)
print('테스트 데이터 세트 정확도: {0:.4f}'.format(accuracy_score(y_test,pred)))
```

```
GridSearchCV 최적 파라미터: {'max_depth': 3, 'min_samples_split': 2}
GridSearchCV 최고 정확도: 0.810434
테스트 데이터 세트 정확도: 0.8771
```