In [25]:

```
import numpy as np
import pandas as pd
from sklearn import datasets
from sklearn import linear_model
from sklearn.cluster import KMeans
from sklearn import model_selection
from sklearn import metrics
from pandas import DataFrame
from matplotlib import pylab as plt

from copy import copy
import random
import sys
import time

# Загружаем набор данных Ирисы:
iris = datasets.load_iris()

iris_frame = DataFrame(iris.data)
iris_frame.columns = iris.feature_names
iris_frame['target'] = iris.target
iris_frame['name'] = iris_frame.target.apply(lambda x : iris.target_names[x])
#iris_frame
```

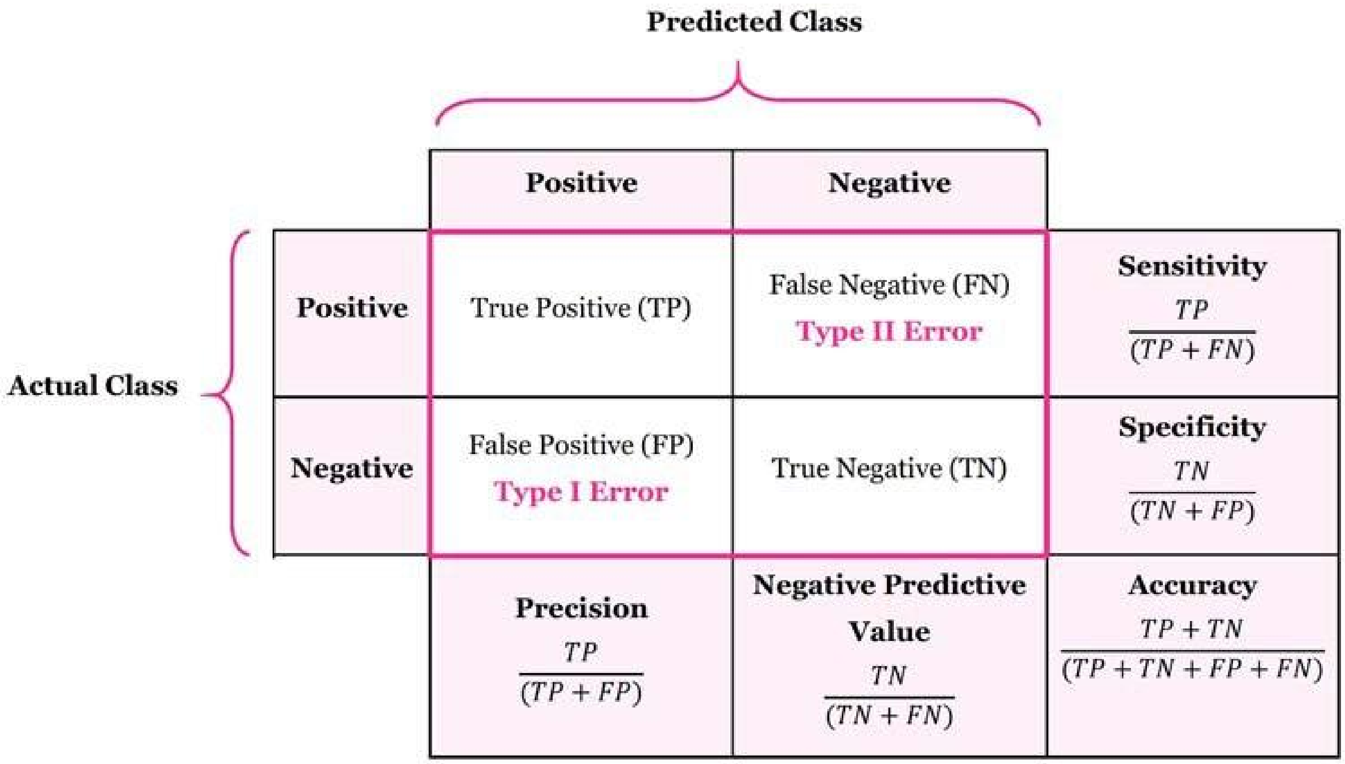
In [26]:

```
iris_frame['petal_area'] = 0.0
for k in range(len(iris_frame['petal length (cm)'])):
    iris_frame['petal_area'][k] = iris_frame['petal length (cm)'][k] * iris_fram
e['petal width (cm)'][k]


s1 = iris_frame[iris_frame['target'] == 2]
s1 = s1.replace(2, 1)
s2 = iris_frame[iris_frame['target'] == 1]
s2 = s2.replace(1, 0)
s3 = iris_frame[iris_frame['target'] == 3]
s3 = s2.replace(0, 0)
binary = pd.concat([s1, s2, s3])

features = np.array(iris_frame.columns)
features = np.append(features[:4], features[-1])
X_train, X_test, y_train, y_test = model_selection.train_test_split(binary[featu
res],

binary[['target']],

test_size = 0.2,

random_state = 0)
X_train.shape, y_train.shape, X_test.shape, y_test.shape
```

Out[26]:

```
((120, 5), (120, 1), (30, 5), (30, 1))
```

CONFUSION MATRIX

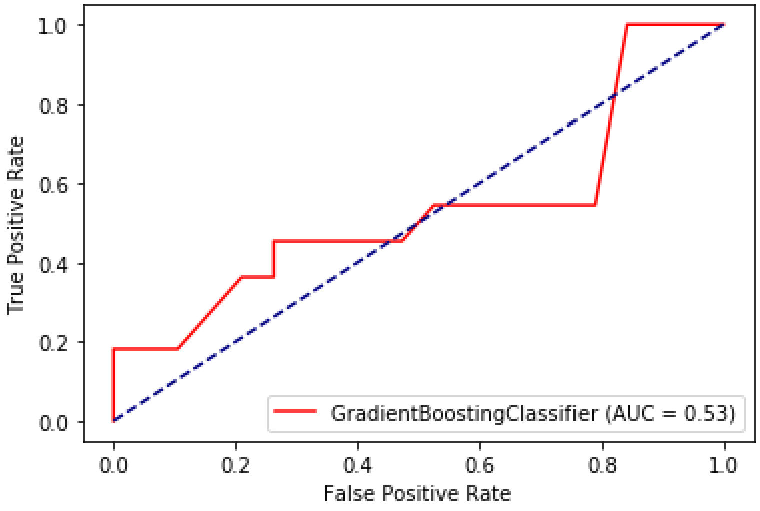


		Predicted Class		
		Positive	Negative	
Actual Class	Positive	True Positive (TP)	False Negative (FN) Type II Error	Sensitivity $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) Type I Error	True Negative (TN)	Specificity $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	Negative Predictive Value $\frac{TN}{(TN+FN)}$	Accuracy $\frac{TP+TN}{(TP+TN+FP+FN)}$

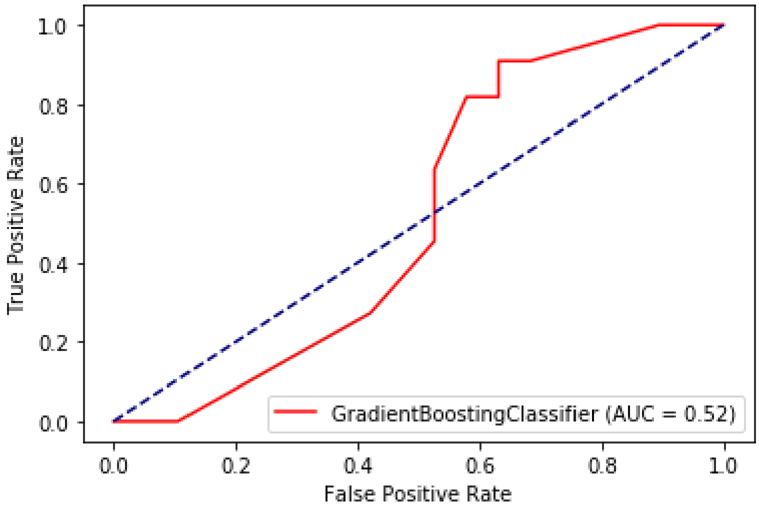
In [29]:

```
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.metrics import plot_roc_curve, confusion_matrix
from sklearn.metrics import roc_auc_score, accuracy_score, f1_score, classificat
ion_report

# hide all warnings!
import warnings
warnings.filterwarnings('ignore')

# work with each feature... for comapring
for feature in features:
    # prepare fitable format for data
    print(f'= = = = = = = = = FEATURE = {feature} = = = = = = = = =', end='\n\n'
)
    X_tr = np.array(X_train[feature]).reshape((len(X_train),1))
    y_te = np.array(y_test).reshape(1, len(y_test))[0]
    X_te = np.array(X_test[feature]).reshape((len(X_test),1))

    #fit model
    model = GradientBoostingClassifier()
    model.fit(X_tr,y_train)

    # predictions
    predictions_model = model.predict(X_te)
    predictions_model = np.array(predictions_model).tolist()

    # get vals
    f1 = round(f1_score(predictions_model, y_te, average='macro')*100,3)
    ROC_AUC = round(roc_auc_score(predictions_model, y_test, average='macro'), 3
)

    # evaluate by metrics f1-score and ROC_AUC
    print(f'f1 score = \t{f1}%')
    print(f'ROC-AUC score = {ROC_AUC}',end='\n\n')

    # in print 0 - positive, 1 - negative
    print(pd.DataFrame(confusion_matrix(y_test, predictions_model), columns=['PR
pos', 'PR neg']), end='\n\n')

    # visualise features and choose the best one
    svc_disp = plot_roc_curve(model, X_te, y_test, color='red')
    print(classification_report(y_test, predictions_model))
    plt.plot([0, 1], [0, 1], color='navy', linestyle='--')
    plt.show()
    predictions_model
    print('= ='*20, end='\n\n')
```

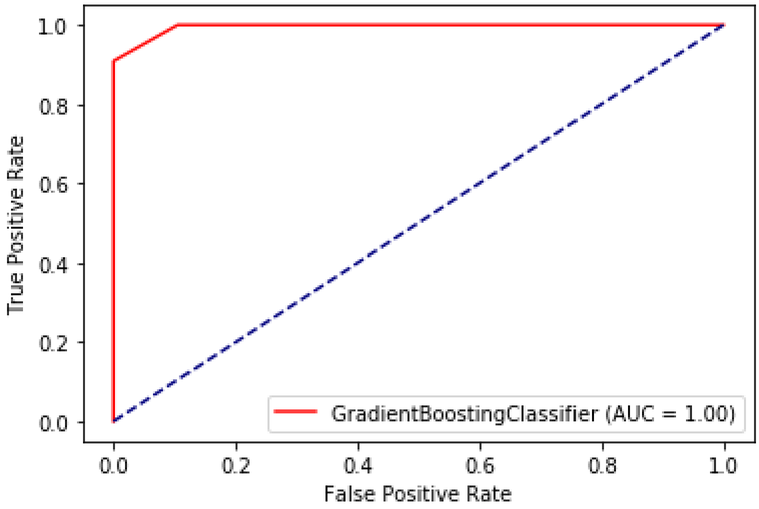
```
= = = = = = = = = FEATURE = sepal length (cm) = = = = = = = = =

f1 score =      57.638%
ROC-AUC score = 0.591

   PR pos  PR neg
0      15       4
1       7       4

              precision    recall  f1-score   support

           0       0.68      0.79      0.73        19
           1       0.50      0.36      0.42        11

    accuracy                           0.63        30
   macro avg       0.59      0.58      0.58        30
weighted avg       0.62      0.63      0.62        30
```



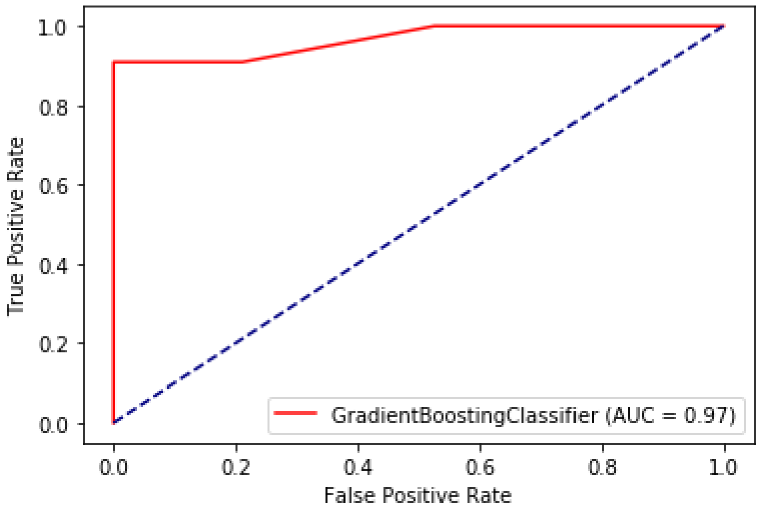
```
= == == == == == == == == == == == == == == == == == == == =

= = = = = = = = = FEATURE = sepal width (cm) = = = = = = = = =

f1 score =      36.17%
ROC-AUC score = 0.304

   PR pos  PR neg
0      17       2
1      11       0

              precision    recall  f1-score   support

           0       0.61      0.89      0.72        19
           1       0.00      0.00      0.00        11

    accuracy                           0.57        30
   macro avg       0.30      0.45      0.36        30
weighted avg       0.38      0.57      0.46        30
```



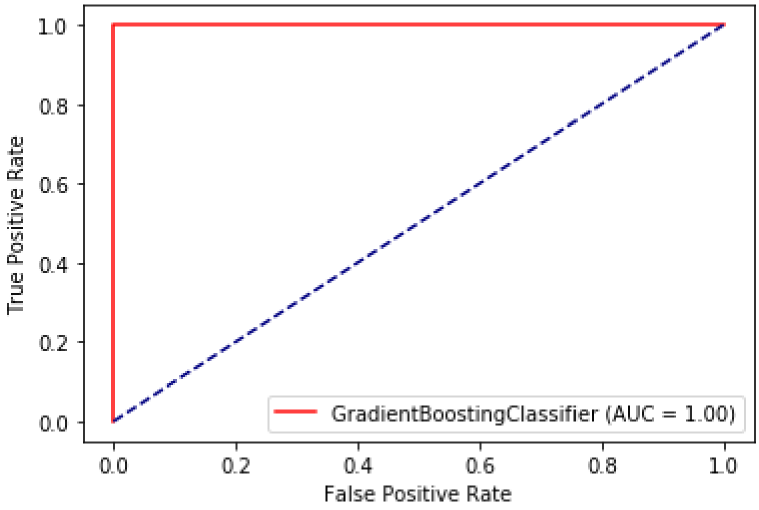
```
= == == == == == == == == == == == == == == == == == == == == =

= = = = = = = = = FEATURE = petal length (cm) = = = = = = = = =

f1 score =      96.337%
ROC-AUC score = 0.975

   PR pos  PR neg
0      19       0
1       1      10

              precision    recall  f1-score   support

           0       0.95      1.00      0.97        19
           1       1.00      0.91      0.95        11

    accuracy                           0.97        30
   macro avg       0.97      0.95      0.96        30
weighted avg       0.97      0.97      0.97        30
```



```
= == == == == == == == == == == == == == == == == == == == =

= = = = = = = = = FEATURE = petal width (cm) = = = = = = = = =

f1 score =      96.337%
ROC-AUC score = 0.975

   PR pos  PR neg
0      19       0
1       1      10

              precision    recall  f1-score   support

           0       0.95      1.00      0.97        19
           1       1.00      0.91      0.95        11

    accuracy                           0.97        30
   macro avg       0.97      0.95      0.96        30
weighted avg       0.97      0.97      0.97        30
```



```
= == == == == == == == == == == == == == == == == == == == =

= = = = = = = = = FEATURE = petal_area = = = = = = = = =

f1 score =      96.337%
ROC-AUC score = 0.975

   PR pos  PR neg
0      19       0
1       1      10

              precision    recall  f1-score   support

           0       0.95      1.00      0.97        19
           1       1.00      0.91      0.95        11

    accuracy                           0.97        30
   macro avg       0.97      0.95      0.96        30
weighted avg       0.97      0.97      0.97        30
```



= =

Feature 'petal_area' is the best one!

+ very interesting example xD https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html (https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html)