Лабораторная работа №4. Оценка качества моделей машинного обучения.

Часть 1. Задача бинарной классификации. ¶

Используемый набор данных: [Breast Cancer Wisconsin (Diagnostic) (https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29)](https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Diagnostic%29)

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report
from sklearn.metrics import roc_curve
from sklearn.metrics import roc_auc_score
import os
import requests

%matplotlib inline

pd.options.display.max_columns = None
```

In [2]:

```
def downloadFile(url, filePath):
    if not os.path.exists(filePath):
        req = requests.get(url)
        f = open(filePath, "wb")
        f.write(req.content)
        f.close

url = "https://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsi
n/"
downloadFile(url + "/wdbc.data", "dataset/wdbc.data")
downloadFile(url + "/wdbc.names", "dataset/wdbc.names")
```

In [3]:

```
headers = ["ID", "Diagnosis", "Radius Mean", "Texture Mean", "Perimeter Mean", "Area Mean", "Smoothness Mean",
           "Compactness Mean", "Concavity Mean", "Concave points Mean", "Symmetry Mean", "Fractal dimension Mean",
           "Radius SE", "Texture SE", "Perimeter SE", "Area SE", "Smoothness SE", "Compactness SE", "Concavity SE",
           "Concave points SE", "Symmetry SE", "Fractal dimension SE", "Radius Worst", "Texture Worst", "Perimeter Worst",
           "Area Worst", "Smoothness Worst", "Compactness Worst", "Concavity Worst", "Concave points Worst",
           "Symmetry Worst", "Fractal dimension Worst"]
data = pd.read_csv("dataset/wdbc.data", names=headers)
data = data.astype({"Diagnosis": "category"})
data.sample(40)
```

Out[3]:

	ID	Diagnosis	Radius Mean	Texture Mean	Perimeter Mean	Area Mean	Smoothness Mean	Compactness Mean	C
388	903011	B	11.270	15.50	73.38	392.0	0.08365	0.11140	
184	873885	M	15.280	22.41	98.92	710.6	0.09057	0.10520	
116	864726	B	8.950	15.76	58.74	245.2	0.09462	0.12430	
73	859983	M	13.800	15.79	90.43	584.1	0.10070	0.12800	
39	855138	M	13.480	20.82	88.40	559.2	0.10160	0.12550	
12	846226	M	19.170	24.80	132.40	1123.0	0.09740	0.24580	
332	897132	B	11.220	19.86	71.94	387.3	0.10540	0.06779	
494	914102	B	13.160	20.54	84.06	538.7	0.07335	0.05275	
103	862980	B	9.876	19.40	63.95	298.3	0.10050	0.09697	
260	887549	M	20.310	27.06	132.90	1288.0	0.10000	0.10880	
16	848406	M	14.680	20.13	94.74	684.5	0.09867	0.07200	
277	8911670	M	18.810	19.98	120.90	1102.0	0.08923	0.05884	
356	9010259	B	13.050	18.59	85.09	512.0	0.10820	0.13040	
513	915940	B	14.580	13.66	94.29	658.8	0.09832	0.08918	
444	9110127	M	18.030	16.85	117.50	990.0	0.08947	0.12320	
473	9113846	B	12.270	29.97	77.42	465.4	0.07699	0.03398	
186	874217	M	18.310	18.58	118.60	1041.0	0.08588	0.08468	
142	869218	B	11.430	17.31	73.66	398.0	0.10920	0.09486	
313	893988	B	11.540	10.72	73.73	409.1	0.08597	0.05969	
181	873593	M	21.090	26.57	142.70	1311.0	0.11410	0.28320	
242	883852	B	11.300	18.19	73.93	389.4	0.09592	0.13250	
471	9113816	B	12.040	28.14	76.85	449.9	0.08752	0.06000	
159	871149	B	10.900	12.96	68.69	366.8	0.07515	0.03718	
561	925311	B	11.200	29.37	70.67	386.0	0.07449	0.03558	
61	858981	B	8.598	20.98	54.66	221.8	0.12430	0.08963	
341	898143	B	9.606	16.84	61.64	280.5	0.08481	0.09228	
520	917092	B	9.295	13.90	59.96	257.8	0.13710	0.12250	
25	852631	M	17.140	16.40	116.00	912.7	0.11860	0.22760	
139	868871	B	11.280	13.39	73.00	384.8	0.11640	0.11360	
543	922296	B	13.210	28.06	84.88	538.4	0.08671	0.06877	
542	921644	B	14.740	25.42	94.70	668.6	0.08275	0.07214	
322	894855	B	12.860	13.32	82.82	504.8	0.11340	0.08834	
68	859471	B	9.029	17.33	58.79	250.5	0.10660	0.14130	
135	868202	M	12.770	22.47	81.72	506.3	0.09055	0.05761	
507	91544002	B	11.060	17.12	71.25	366.5	0.11940	0.10710	
180	873592	M	27.220	21.87	182.10	2250.0	0.10940	0.19140	

	ID	Diagnosis	Radius Mean	Texture Mean	Perimeter Mean	Area Mean	Smoothness Mean	Compactness Mean	C
427	90745	B	10.800	21.98	68.79	359.9	0.08801	0.05743	
179	873586	B	12.810	13.06	81.29	508.8	0.08739	0.03774	
91	861799	M	15.370	22.76	100.20	728.2	0.09200	0.10360	
525	91805	B	8.571	13.10	54.53	221.3	0.10360	0.07632	

In [4]:

```
data.isna().sum()
```

Out[4]:

```
ID                         0
Diagnosis                  0
Radius Mean                0
Texture Mean               0
Perimeter Mean             0
Area Mean                  0
Smoothness Mean            0
Compactness Mean           0
Concavity Mean             0
Concave points Mean        0
Symmetry Mean              0
Fractal dimension Mean     0
Radius SE                  0
Texture SE                 0
Perimeter SE               0
Area SE                    0
Smoothness SE              0
Compactness SE             0
Concavity SE               0
Concave points SE          0
Symmetry SE                0
Fractal dimension SE       0
Radius Worst               0
Texture Worst              0
Perimeter Worst            0
Area Worst                 0
Smoothness Worst           0
Compactness Worst          0
Concavity Worst            0
Concave points Worst       0
Symmetry Worst             0
Fractal dimension Worst    0
dtype: int64
```

Пропусков в данных нет.

Подготовим данные для классификации: выберем признаки и метки и сформируем тренировочные и тестовые наборы.

In [5]:

```
X = data.drop(columns=["ID", "Diagnosis"]).copy()
y = data["Diagnosis"].copy().cat.codes

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=2
7)
```

Создадим классификатор, обучим его, а затем выполним классификацию.

In [6]:

```
dtc = DecisionTreeClassifier()
dtc.fit(X_train, y_train)
y_pred = dtc.predict(X_test)
```

Оценим получившуюся классификацию.

In [7]:

```
print(classification_report(y_test, y_pred))
```

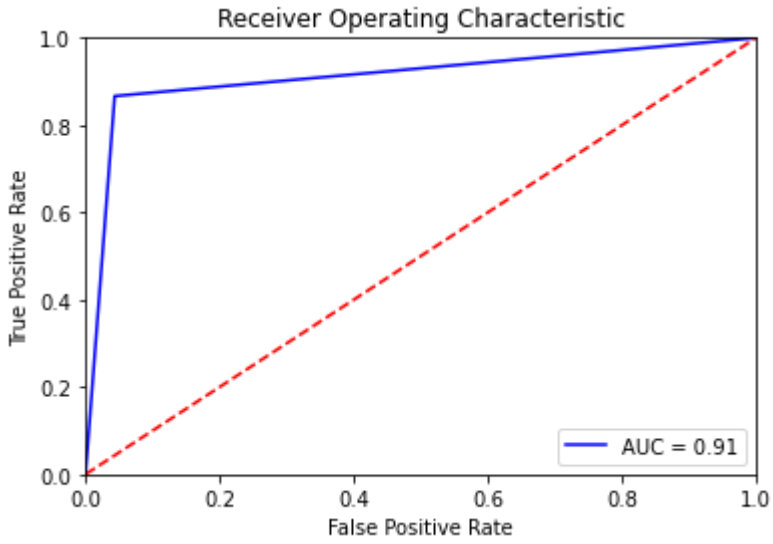
```
              precision    recall  f1-score   support

           0       0.92      0.96      0.94        69
           1       0.93      0.87      0.90        45

    accuracy                           0.92       114
   macro avg       0.92      0.91      0.92       114
weighted avg       0.92      0.92      0.92       114
```

In [8]:

```
fpr, tpr, _ = roc_curve(y_test, y_pred)
auc = roc_auc_score(y_test, y_pred)
```

In [9]:

```
plt.title('Receiver Operating Characteristic')
plt.plot(fpr, tpr, 'b', label = "AUC = %0.2f"%auc)
plt.legend(loc = 'lower right')
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0, 1])
plt.ylim([0, 1])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```



Высокое значение AUC говорит о качественной классификации.