

Лабораторная работа №7. Метод опорных векторов и логистическая регрессия в решении задач машинного обучения.

Часть 2. Логистическая регрессия.

Используемый набор данных: [banknote authentication](https://archive.ics.uci.edu/ml/datasets/banknote+authentication)
(<https://archive.ics.uci.edu/ml/datasets/banknote+authentication>)

In [1]:

```
from IPython.display import display
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, roc_curve, 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/00267/data_banknote_authentication.txt"
fileName = "dataset/data_banknote_authentication.txt"
downloadFile(url, fileName)
```

Опишем заголовки колонок. Для удобства используем сокращенные названия:

- **Variance** - Variance of Wavelet Transformed image.
- **Skewness** - Skewness of Wavelet Transformed image.
- **Curtosis** - Curtosis of Wavelet Transformed image.
- **Entropy** - Entropy of image.

In [3]:

```
headers = ["Variance", "Skewness", "Curtosis", "Entropy", "Class"]  
data = pd.read_csv(fileName, names=headers)  
data["Class"] = data["Class"].astype("category")  
data.sample(40)
```

Out[3]:

	Variance	Skewness	Curtosis	Entropy	Class
907	-1.39070	-1.37810	2.305500	-0.021566	1
247	2.05970	-0.99326	5.211900	-0.293120	0
566	2.64150	7.58600	-0.285620	-1.667700	0
770	0.34340	0.12415	-0.287330	0.146540	1
1174	-3.20510	-0.14279	0.975650	0.045675	1
637	3.30040	7.08110	-1.325800	0.222830	0
944	-2.12100	-0.05588	1.949000	1.353000	1
512	2.66480	10.75400	-3.399400	-4.168500	0
1371	-2.54190	-0.65804	2.684200	1.195200	1
416	4.34830	11.10790	-4.085700	-4.253900	0
650	0.21084	9.43590	-0.094543	-1.859000	0
73	4.16540	-3.44950	3.643000	1.087900	0
1106	-0.12690	-1.15050	-0.951380	0.578430	1
877	-5.12160	-5.31180	10.384600	-1.061200	1
107	-1.85840	7.88600	-1.664300	-1.838400	0
409	0.64295	7.10180	0.349300	-0.413370	0
1296	-3.59330	0.22968	0.712600	-0.333200	1
99	-2.74190	11.40380	2.539400	-5.579300	0
276	4.05200	-0.16555	0.453830	0.512480	0
1273	-2.36750	-0.43663	1.692000	-0.430180	1
769	-0.89409	3.19910	-1.821900	-2.945200	1
137	5.03200	8.20260	-2.625600	-1.034100	0
599	3.92920	-2.91560	2.212900	0.308170	0
1257	-1.89740	3.50740	-1.784200	-3.849100	1
344	3.59820	7.13070	-1.303500	0.212480	0
364	5.78230	5.57880	-2.408900	-0.056479	0
430	1.56910	6.34650	-0.182800	-2.409900	0
1128	-2.18020	3.37910	-1.225600	-2.662100	1
983	-2.03360	-1.40920	1.158200	0.365070	1
1032	0.16358	-3.35840	1.374900	1.356900	1
1047	1.06370	3.69570	-4.159400	-1.937900	1
11	3.98990	-2.70660	2.394600	0.862910	0
418	-1.25760	1.58920	7.007800	0.424550	0
166	1.11660	8.64960	-0.962520	-1.811200	0
117	2.10800	6.79550	-0.170800	0.490500	0
393	3.78310	10.05260	-3.886900	-3.736600	0
645	2.92330	6.04640	-0.111680	-0.586650	0

	Variance	Skewness	Curtosis	Entropy	Class
507	4.60140	5.62640	-2.123500	0.193090	0
300	0.32920	-4.45520	4.571800	-0.988800	0
334	3.92940	1.41120	1.807600	0.897820	0

In [4]:

```
display(data.isna().sum())
display(data.describe())
```

```
Variance      0
Skewness      0
Curtosis      0
Entropy       0
Class         0
dtype: int64
```

	Variance	Skewness	Curtosis	Entropy
count	1372.000000	1372.000000	1372.000000	1372.000000
mean	0.433735	1.922353	1.397627	-1.191657
std	2.842763	5.869047	4.310030	2.101013
min	-7.042100	-13.773100	-5.286100	-8.548200
25%	-1.773000	-1.708200	-1.574975	-2.413450
50%	0.496180	2.319650	0.616630	-0.586650
75%	2.821475	6.814625	3.179250	0.394810
max	6.824800	12.951600	17.927400	2.449500

Пропусков в данных нет.

Подготовим данные и обучим модель.

In [5]:

```
X = data.drop(columns=["Class"]).copy()
y = data["Class"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=0)
y_pred = LogisticRegression().fit(X_train, y_train).predict(X_test)
```

Оценим получившуюся классификацию.

