Кан Владислав

AIN-2-22

1. Загрузите датасеты классификации из библиотеки skikit-learn:

In [3]:
```
from sklearn.datasets import load_digits, load_wine, load_breast_cancer


digits = load_digits()
print("Размер данных (digits):", digits.data.shape)
print("Целевые классы (digits):", digits.target_names)


wine = load_wine()
print("Размер данных (wine):", wine.data.shape)
print("Целевые классы (wine):", wine.target_names)


breast_cancer = load_breast_cancer()
print("Размер данных (breast_cancer):", breast_cancer.data.shape)
print("Целевые классы (breast_cancer):", breast_cancer.target_names)
```

```
Размер данных (digits): (1797, 64)
Целевые классы (digits): [0 1 2 3 4 5 6 7 8 9]
Размер данных (wine): (178, 13)
Целевые классы (wine): ['class_0' 'class_1' 'class_2']
Размер данных (breast_cancer): (569, 30)
Целевые классы (breast_cancer): ['malignant' 'benign']
```

Опираясь на приведенные выше примеры, выполните следующие задачи, указанными выше датасетами:

2. Изучите данные

In [16]:
```
import pandas as pd

# Изучение датасета digits
digits_df = pd.DataFrame(digits.data)
digits_df['target'] = digits.target
print(digits_df.head())

# Изучение датасета wine
wine_df = pd.DataFrame(wine.data)
wine_df['target'] = wine.target
print(wine_df.head())

# Изучение датасета cancer
```

```
cancer_df = pd.DataFrame(cancer.data)
cancer_df['target'] = cancer.target
print(cancer_df.head())
```

```
     0    1    2     3     4     5    6    7    8    9  ...   55   56   57  \
0  0.0  0.0  5.0  13.0   9.0   1.0  0.0  0.0  0.0  0.0  ...  0.0  0.0  0.0
1  0.0  0.0  0.0  12.0  13.0   5.0  0.0  0.0  0.0  0.0  ...  0.0  0.0  0.0
2  0.0  0.0  0.0   4.0  15.0  12.0  0.0  0.0  0.0  0.0  ...  0.0  0.0  0.0
3  0.0  0.0  7.0  15.0  13.0   1.0  0.0  0.0  0.0  8.0  ...  0.0  0.0  0.0
4  0.0  0.0  0.0   1.0  11.0   0.0  0.0  0.0  0.0  0.0  ...  0.0  0.0  0.0

    58    59    60    61   62   63  target
0  6.0  13.0  10.0   0.0  0.0  0.0       0
1  0.0  11.0  16.0  10.0  0.0  0.0       1
2  0.0   3.0  11.0  16.0  9.0  0.0       2
3  7.0  13.0  13.0   9.0  0.0  0.0       3
4  0.0   2.0  16.0   4.0  0.0  0.0       4

[5 rows x 65 columns]
       0     1     2     3      4     5     6     7     8     9    10    11  \
0  14.23  1.71  2.43  15.6  127.0  2.80  3.06  0.28  2.29  5.64  1.04  3.92
1  13.20  1.78  2.14  11.2  100.0  2.65  2.76  0.26  1.28  4.38  1.05  3.40
2  13.16  2.36  2.67  18.6  101.0  2.80  3.24  0.30  2.81  5.68  1.03  3.17
3  14.37  1.95  2.50  16.8  113.0  3.85  3.49  0.24  2.18  7.80  0.86  3.45
4  13.24  2.59  2.87  21.0  118.0  2.80  2.69  0.39  1.82  4.32  1.04  2.93

       12  target
0  1065.0       0
1  1050.0       0
2  1185.0       0
3  1480.0       0
4   735.0       0
       0      1       2       3        4        5       6        7       8  \
0  17.99  10.38  122.80  1001.0  0.11840  0.27760  0.3001  0.14710  0.2419
1  20.57  17.77  132.90  1326.0  0.08474  0.07864  0.0869  0.07017  0.1812
2  19.69  21.25  130.00  1203.0  0.10960  0.15990  0.1974  0.12790  0.2069
3  11.42  20.38   77.58   386.1  0.14250  0.28390  0.2414  0.10520  0.2597
4  20.29  14.34  135.10  1297.0  0.10030  0.13280  0.1980  0.10430  0.1809

         9  ...     21      22      23      24      25      26      27  \
0  0.07871  ...  17.33  184.60  2019.0  0.1622  0.6656  0.7119  0.2654
1  0.05667  ...  23.41  158.80  1956.0  0.1238  0.1866  0.2416  0.1860
2  0.05999  ...  25.53  152.50  1709.0  0.1444  0.4245  0.4504  0.2430
3  0.09744  ...  26.50   98.87   567.7  0.2098  0.8663  0.6869  0.2575
4  0.05883  ...  16.67  152.20  1575.0  0.1374  0.2050  0.4000  0.1625

       28       29  target
0  0.4601  0.11890       0
1  0.2750  0.08902       0
2  0.3613  0.08758       0
3  0.6638  0.17300       0
4  0.2364  0.07678       0

[5 rows x 31 columns]
```

Исследование данных для набора рукописных цифр (load_digits)

In [5]:
```
print("Описание набора Digits:\n", digits.DESCR)
print("Признаки для набора Digits (первые 5 строк):\n", digits.data[:5])
print("Целевые метки для набора Digits (первые 5 меток):\n", digits.target[:5])
```

```
print(f"Размер данных (digits): {digits.data.shape}")
print(f"Размер целевых меток (digits): {digits.target.shape}")
print(f"Уникальные классы для набора Digits: {set(digits.target)}")
import numpy as np
unique, counts = np.unique(digits.target, return_counts=True)
print(f"Распределение классов (digits): {dict(zip(unique, counts))}")
```

Описание набора Digits:
 .. _digits_dataset:

Optical recognition of handwritten digits dataset

Data Set Characteristics:

:Number of Instances: 1797
:Number of Attributes: 64
:Attribute Information: 8x8 image of integer pixels in the range 0..16.
:Missing Attribute Values: None
:Creator: E. Alpaydin (alpaydin '@' boun.edu.tr)
:Date: July; 1998

This is a copy of the test set of the UCI ML hand-written digits datasets
https://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits

The data set contains images of hand-written digits: 10 classes where
each class refers to a digit.

Preprocessing programs made available by NIST were used to extract
normalized bitmaps of handwritten digits from a preprinted form. From a
total of 43 people, 30 contributed to the training set and different 13
to the test set. 32x32 bitmaps are divided into nonoverlapping blocks of
4x4 and the number of on pixels are counted in each block. This generates
an input matrix of 8x8 where each element is an integer in the range
0..16. This reduces dimensionality and gives invariance to small
distortions.

For info on NIST preprocessing routines, see M. D. Garris, J. L. Blue, G.
T. Candela, D. L. Dimmick, J. Geist, P. J. Grother, S. A. Janet, and C.
L. Wilson, NIST Form-Based Handprint Recognition System, NISTIR 5469,
1994.

.. dropdown:: References

  - C. Kaynak (1995) Methods of Combining Multiple Classifiers and Their
    Applications to Handwritten Digit Recognition, MSc Thesis, Institute of
    Graduate Studies in Science and Engineering, Bogazici University.
  - E. Alpaydin, C. Kaynak (1998) Cascading Classifiers, Kybernetika.
  - Ken Tang and Ponnuthurai N. Suganthan and Xi Yao and A. Kai Qin.
    Linear dimensionalityreduction using relevance weighted LDA. School of
    Electrical and Electronic Engineering Nanyang Technological University.
    2005.
  - Claudio Gentile. A New Approximate Maximal Margin Classification
    Algorithm. NIPS. 2000.

Признаки для набора Digits (первые 5 строк):
 [[0. 0. 5. 13. 9. 1. 0. 0. 0. 0. 13. 15. 10. 15. 5. 0. 0. 3.
  15. 2. 0. 11. 8. 0. 0. 4. 12. 0. 0. 8. 8. 0. 0. 5. 8. 0.
   0. 9. 8. 0. 0. 4. 11. 0. 1. 12. 7. 0. 0. 2. 14. 5. 10. 12.
   0. 0. 0. 0. 6. 13. 10. 0. 0. 0.]
 [0. 0. 0. 12. 13. 5. 0. 0. 0. 0. 0. 11. 16. 9. 0. 0. 0. 0.
   3. 15. 16. 6. 0. 0. 0. 7. 15. 16. 16. 2. 0. 0. 0. 0. 1. 16.
  16. 3. 0. 0. 0. 0. 1. 16. 16. 6. 0. 0. 0. 0. 1. 16. 16. 6.
   0. 0. 0. 0. 0. 11. 16. 10. 0. 0.]
 [0. 0. 0. 4. 15. 12. 0. 0. 0. 0. 3. 16. 15. 14. 0. 0. 0. 0.
   8. 13. 8. 16. 0. 0. 0. 0. 1. 6. 15. 11. 0. 0. 0. 1. 8. 13.
  15. 1. 0. 0. 0. 9. 16. 16. 5. 0. 0. 0. 0. 3. 13. 16. 16. 11.

```
   5.  0.  0.  0.  0.  3. 11. 16.  9.  0.]
 [ 0.  0.  7. 15. 13.  1.  0.  0.  0.  8. 13.  6. 15.  4.  0.  0.  0.  2.
   1. 13. 13.  0.  0.  0.  0.  0.  2. 15. 11.  1.  0.  0.  0.  0.  0.  1.
  12. 12.  1.  0.  0.  0.  0.  0.  1. 10.  8.  0.  0.  0.  8.  4.  5. 14.
   9.  0.  0.  0.  7. 13. 13.  9.  0.  0.]
 [ 0.  0.  0.  1. 11.  0.  0.  0.  0.  0.  0.  7.  8.  0.  0.  0.  0.  0.
   1. 13.  6.  2.  2.  0.  0.  0.  7. 15.  0.  9.  8.  0.  0.  5. 16. 10.
   0. 16.  6.  0.  0.  4. 15. 16. 13. 16.  1.  0.  0.  0.  0.  3. 15. 10.
   0.  0.  0.  0.  0.  2. 16.  4.  0.  0.]]
Целевые метки для набора Digits (первые 5 меток):
 [0 1 2 3 4]
Размер данных (digits): (1797, 64)
Размер целевых меток (digits): (1797,)
Уникальные классы для набора Digits: {np.int64(0), np.int64(1), np.int64(2), np.i
nt64(3), np.int64(4), np.int64(5), np.int64(6), np.int64(7), np.int64(8), np.int6
4(9)}
Распределение классов (digits): {np.int64(0): np.int64(178), np.int64(1): np.int6
4(182), np.int64(2): np.int64(177), np.int64(3): np.int64(183), np.int64(4): np.i
nt64(181), np.int64(5): np.int64(182), np.int64(6): np.int64(181), np.int64(7): n
p.int64(179), np.int64(8): np.int64(174), np.int64(9): np.int64(180)}
```

Исследование данных для набора вин (load_wine)

In [6]:
```python
print("Описание набора Wine:\n", wine.DESCR)
print("Признаки для набора Wine (первые 5 строк):\n", wine.data[:5])
print("Целевые метки для набора Wine (первые 5 меток):\n", wine.target[:5])
print(f"Размер данных (wine): {wine.data.shape}")
print(f"Размер целевых меток (wine): {wine.target.shape}")
print(f"Уникальные классы для набора Wine: {set(wine.target)}")
unique_wine, counts_wine = np.unique(wine.target, return_counts=True)
print(f"Распределение классов (wine): {dict(zip(unique_wine, counts_wine))}")
```

Описание набора Wine:
 .. _wine_dataset:

Wine recognition dataset

Data Set Characteristics:

:Number of Instances: 178
:Number of Attributes: 13 numeric, predictive attributes and the class
:Attribute Information:
    - Alcohol
    - Malic acid
    - Ash
    - Alcalinity of ash
    - Magnesium
    - Total phenols
    - Flavanoids
    - Nonflavanoid phenols
    - Proanthocyanins
    - Color intensity
    - Hue
    - OD280/OD315 of diluted wines
    - Proline
    - class:
        - class_0
        - class_1
        - class_2

:Summary Statistics:

	Min	Max	Mean	SD
Alcohol:	11.0	14.8	13.0	0.8
Malic Acid:	0.74	5.80	2.34	1.12
Ash:	1.36	3.23	2.36	0.27
Alcalinity of Ash:	10.6	30.0	19.5	3.3
Magnesium:	70.0	162.0	99.7	14.3
Total Phenols:	0.98	3.88	2.29	0.63
Flavanoids:	0.34	5.08	2.03	1.00
Nonflavanoid Phenols:	0.13	0.66	0.36	0.12
Proanthocyanins:	0.41	3.58	1.59	0.57
Colour Intensity:	1.3	13.0	5.1	2.3
Hue:	0.48	1.71	0.96	0.23
OD280/OD315 of diluted wines:	1.27	4.00	2.61	0.71
Proline:	278	1680	746	315

:Missing Attribute Values: None
:Class Distribution: class_0 (59), class_1 (71), class_2 (48)
:Creator: R.A. Fisher
:Donor: Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)
:Date: July, 1988

This is a copy of UCI ML Wine recognition datasets.
https://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data

The data is the results of a chemical analysis of wines grown in the same
region in Italy by three different cultivators. There are thirteen different

```
measurements taken for different constituents found in the three types of
wine.

Original Owners:

Forina, M. et al, PARVUS -
An Extendible Package for Data Exploration, Classification and Correlation.
Institute of Pharmaceutical and Food Analysis and Technologies,
Via Brigata Salerno, 16147 Genoa, Italy.

Citation:

Lichman, M. (2013). UCI Machine Learning Repository
[https://archive.ics.uci.edu/ml]. Irvine, CA: University of California,
School of Information and Computer Science.

.. dropdown:: References

    (1) S. Aeberhard, D. Coomans and O. de Vel,
    Comparison of Classifiers in High Dimensional Settings,
    Tech. Rep. no. 92-02, (1992), Dept. of Computer Science and Dept. of
    Mathematics and Statistics, James Cook University of North Queensland.
    (Also submitted to Technometrics).

    The data was used with many others for comparing various
    classifiers. The classes are separable, though only RDA
    has achieved 100% correct classification.
    (RDA : 100%, QDA 99.4%, LDA 98.9%, 1NN 96.1% (z-transformed data))
    (All results using the leave-one-out technique)

    (2) S. Aeberhard, D. Coomans and O. de Vel,
    "THE CLASSIFICATION PERFORMANCE OF RDA"
    Tech. Rep. no. 92-01, (1992), Dept. of Computer Science and Dept. of
    Mathematics and Statistics, James Cook University of North Queensland.
    (Also submitted to Journal of Chemometrics).

Признаки для набора Wine (первые 5 строк):
 [[1.423e+01 1.710e+00 2.430e+00 1.560e+01 1.270e+02 2.800e+00 3.060e+00
  2.800e-01 2.290e+00 5.640e+00 1.040e+00 3.920e+00 1.065e+03]
 [1.320e+01 1.780e+00 2.140e+00 1.120e+01 1.000e+02 2.650e+00 2.760e+00
  2.600e-01 1.280e+00 4.380e+00 1.050e+00 3.400e+00 1.050e+03]
 [1.316e+01 2.360e+00 2.670e+00 1.860e+01 1.010e+02 2.800e+00 3.240e+00
  3.000e-01 2.810e+00 5.680e+00 1.030e+00 3.170e+00 1.185e+03]
 [1.437e+01 1.950e+00 2.500e+00 1.680e+01 1.130e+02 3.850e+00 3.490e+00
  2.400e-01 2.180e+00 7.800e+00 8.600e-01 3.450e+00 1.480e+03]
 [1.324e+01 2.590e+00 2.870e+00 2.100e+01 1.180e+02 2.800e+00 2.690e+00
  3.900e-01 1.820e+00 4.320e+00 1.040e+00 2.930e+00 7.350e+02]]
Целевые метки для набора Wine (первые 5 меток):
 [0 0 0 0 0]
Размер данных (wine): (178, 13)
Размер целевых меток (wine): (178,)
Уникальные классы для набора Wine: {np.int64(0), np.int64(1), np.int64(2)}
Распределение классов (wine): {np.int64(0): np.int64(59), np.int64(1): np.int64(7
1), np.int64(2): np.int64(48)}
```

Исследование данных для набора по раку груди (load_breast_cancer)

In [7]:
```
print("Описание набора Breast Cancer:\n", breast_cancer.DESCR)
print("Признаки для набора Breast Cancer (первые 5 строк):\n", breast_cancer.dat
```

```
print("Целевые метки для набора Breast Cancer (первые 5 меток):\n", breast_cance
print(f"Размер данных (breast cancer): {breast_cancer.data.shape}")
print(f"Размер целевых меток (breast cancer): {breast_cancer.target.shape}")
print(f"Уникальные классы для набора Breast Cancer: {set(breast_cancer.target)}"
unique_breast, counts_breast = np.unique(breast_cancer.target, return_counts=Tru
print(f"Распределение классов (breast cancer): {dict(zip(unique_breast, counts_b
```

Описание набора Breast Cancer:
 .. _breast_cancer_dataset:

Breast cancer wisconsin (diagnostic) dataset

Data Set Characteristics:

:Number of Instances: 569

:Number of Attributes: 30 numeric, predictive attributes and the class

:Attribute Information:
    - radius (mean of distances from center to points on the perimeter)
    - texture (standard deviation of gray-scale values)
    - perimeter
    - area
    - smoothness (local variation in radius lengths)
    - compactness (perimeter^2 / area - 1.0)
    - concavity (severity of concave portions of the contour)
    - concave points (number of concave portions of the contour)
    - symmetry
    - fractal dimension ("coastline approximation" - 1)

    The mean, standard error, and "worst" or largest (mean of the three
    worst/largest values) of these features were computed for each image,
    resulting in 30 features. For instance, field 0 is Mean Radius, field
    10 is Radius SE, field 20 is Worst Radius.

    - class:
            - WDBC-Malignant
            - WDBC-Benign

:Summary Statistics:

	Min	Max
radius (mean):	6.981	28.11
texture (mean):	9.71	39.28
perimeter (mean):	43.79	188.5
area (mean):	143.5	2501.0
smoothness (mean):	0.053	0.163
compactness (mean):	0.019	0.345
concavity (mean):	0.0	0.427
concave points (mean):	0.0	0.201
symmetry (mean):	0.106	0.304
fractal dimension (mean):	0.05	0.097
radius (standard error):	0.112	2.873
texture (standard error):	0.36	4.885
perimeter (standard error):	0.757	21.98
area (standard error):	6.802	542.2
smoothness (standard error):	0.002	0.031
compactness (standard error):	0.002	0.135
concavity (standard error):	0.0	0.396
concave points (standard error):	0.0	0.053
symmetry (standard error):	0.008	0.079
fractal dimension (standard error):	0.001	0.03
radius (worst):	7.93	36.04
texture (worst):	12.02	49.54

```
perimeter (worst):                    50.41  251.2
area (worst):                         185.2  4254.0
smoothness (worst):                   0.071  0.223
compactness (worst):                  0.027  1.058
concavity (worst):                    0.0    1.252
concave points (worst):               0.0    0.291
symmetry (worst):                     0.156  0.664
fractal dimension (worst):            0.055  0.208
==================================== ====== ======

:Missing Attribute Values: None

:Class Distribution: 212 - Malignant, 357 - Benign

:Creator:  Dr. William H. Wolberg, W. Nick Street, Olvi L. Mangasarian

:Donor: Nick Street

:Date: November, 1995

This is a copy of UCI ML Breast Cancer Wisconsin (Diagnostic) datasets.
https://goo.gl/U2Uwz2

Features are computed from a digitized image of a fine needle
aspirate (FNA) of a breast mass.  They describe
characteristics of the cell nuclei present in the image.

Separating plane described above was obtained using
Multisurface Method-Tree (MSM-T) [K. P. Bennett, "Decision Tree
Construction Via Linear Programming." Proceedings of the 4th
Midwest Artificial Intelligence and Cognitive Science Society,
pp. 97-101, 1992], a classification method which uses linear
programming to construct a decision tree.  Relevant features
were selected using an exhaustive search in the space of 1-4
features and 1-3 separating planes.

The actual linear program used to obtain the separating plane
in the 3-dimensional space is that described in:
[K. P. Bennett and O. L. Mangasarian: "Robust Linear
Programming Discrimination of Two Linearly Inseparable Sets",
Optimization Methods and Software 1, 1992, 23-34].

This database is also available through the UW CS ftp server:

ftp ftp.cs.wisc.edu
cd math-prog/cpo-dataset/machine-learn/WDBC/

.. dropdown:: References

  - W.N. Street, W.H. Wolberg and O.L. Mangasarian. Nuclear feature extraction
    for breast tumor diagnosis. IS&T/SPIE 1993 International Symposium on
    Electronic Imaging: Science and Technology, volume 1905, pages 861-870,
    San Jose, CA, 1993.
  - O.L. Mangasarian, W.N. Street and W.H. Wolberg. Breast cancer diagnosis and
    prognosis via linear programming. Operations Research, 43(4), pages 570-577,
    July-August 1995.
  - W.H. Wolberg, W.N. Street, and O.L. Mangasarian. Machine learning techniques
    to diagnose breast cancer from fine-needle aspirates. Cancer Letters 77 (199
4)
    163-171.
```

```
Признаки для набора Breast Cancer (первые 5 строк):
 [[1.799e+01 1.038e+01 1.228e+02 1.001e+03 1.184e-01 2.776e-01 3.001e-01
  1.471e-01 2.419e-01 7.871e-02 1.095e+00 9.053e-01 8.589e+00 1.534e+02
  6.399e-03 4.904e-02 5.373e-02 1.587e-02 3.003e-02 6.193e-03 2.538e+01
  1.733e+01 1.846e+02 2.019e+03 1.622e-01 6.656e-01 7.119e-01 2.654e-01
  4.601e-01 1.189e-01]
 [2.057e+01 1.777e+01 1.329e+02 1.326e+03 8.474e-02 7.864e-02 8.690e-02
  7.017e-02 1.812e-01 5.667e-02 5.435e-01 7.339e-01 3.398e+00 7.408e+01
  5.225e-03 1.308e-02 1.860e-02 1.340e-02 1.389e-02 3.532e-03 2.499e+01
  2.341e+01 1.588e+02 1.956e+03 1.238e-01 1.866e-01 2.416e-01 1.860e-01
  2.750e-01 8.902e-02]
 [1.969e+01 2.125e+01 1.300e+02 1.203e+03 1.096e-01 1.599e-01 1.974e-01
  1.279e-01 2.069e-01 5.999e-02 7.456e-01 7.869e-01 4.585e+00 9.403e+01
  6.150e-03 4.006e-02 3.832e-02 2.058e-02 2.250e-02 4.571e-03 2.357e+01
  2.553e+01 1.525e+02 1.709e+03 1.444e-01 4.245e-01 4.504e-01 2.430e-01
  3.613e-01 8.758e-02]
 [1.142e+01 2.038e+01 7.758e+01 3.861e+02 1.425e-01 2.839e-01 2.414e-01
  1.052e-01 2.597e-01 9.744e-02 4.956e-01 1.156e+00 3.445e+00 2.723e+01
  9.110e-03 7.458e-02 5.661e-02 1.867e-02 5.963e-02 9.208e-03 1.491e+01
  2.650e+01 9.887e+01 5.677e+02 2.098e-01 8.663e-01 6.869e-01 2.575e-01
  6.638e-01 1.730e-01]
 [2.029e+01 1.434e+01 1.351e+02 1.297e+03 1.003e-01 1.328e-01 1.980e-01
  1.043e-01 1.809e-01 5.883e-02 7.572e-01 7.813e-01 5.438e+00 9.444e+01
  1.149e-02 2.461e-02 5.688e-02 1.885e-02 1.756e-02 5.115e-03 2.254e+01
  1.667e+01 1.522e+02 1.575e+03 1.374e-01 2.050e-01 4.000e-01 1.625e-01
  2.364e-01 7.678e-02]]
Целевые метки для набора Breast Cancer (первые 5 меток):
 [0 0 0 0 0]
Размер данных (breast cancer): (569, 30)
Размер целевых меток (breast cancer): (569,)
Уникальные классы для набора Breast Cancer: {np.int64(0), np.int64(1)}
Распределение классов (breast cancer): {np.int64(0): np.int64(212), np.int64(1):
np.int64(357)}
```

Типы данных для наборов. Статистическая информация для наборов

In [8]:

```python
import pandas as pd

digits_df = pd.DataFrame(digits.data, columns=[f'pixel_{i}' for i in range(digit
wine_df = pd.DataFrame(wine.data, columns=wine.feature_names)
breast_cancer_df = pd.DataFrame(breast_cancer.data, columns=breast_cancer.featur

print("Статистическая информация для набора Digits:\n", digits_df.describe())
print("Статистическая информация для набора Wine:\n", wine_df.describe())
print("Статистическая информация для набора Breast Cancer:\n", breast_cancer_df.
print("Типы данных для набора Digits:\n", digits_df.dtypes)
print("Типы данных для набора Wine:\n", wine_df.dtypes)
print("Типы данных для набора Breast Cancer:\n", breast_cancer_df.dtypes)
```

Статистическая информация для набора Digits:

	pixel_0	pixel_1	pixel_2	pixel_3	pixel_4
count	1797.0	1797.000000	1797.000000	1797.000000	1797.000000
mean	0.0	0.303840	5.204786	11.835838	11.848080
std	0.0	0.907192	4.754826	4.248842	4.287388
min	0.0	0.000000	0.000000	0.000000	0.000000
25%	0.0	0.000000	1.000000	10.000000	10.000000
50%	0.0	0.000000	4.000000	13.000000	13.000000
75%	0.0	0.000000	9.000000	15.000000	15.000000
max	0.0	8.000000	16.000000	16.000000	16.000000

	pixel_5	pixel_6	pixel_7	pixel_8	pixel_9	...
count	1797.000000	1797.000000	1797.000000	1797.000000	1797.000000	...
mean	5.781859	1.362270	0.129661	0.005565	1.993879	...
std	5.666418	3.325775	1.037383	0.094222	3.196160	...
min	0.000000	0.000000	0.000000	0.000000	0.000000	...
25%	0.000000	0.000000	0.000000	0.000000	0.000000	...
50%	4.000000	0.000000	0.000000	0.000000	0.000000	...
75%	11.000000	0.000000	0.000000	0.000000	3.000000	...
max	16.000000	16.000000	15.000000	2.000000	16.000000	...

	pixel_54	pixel_55	pixel_56	pixel_57	pixel_58
count	1797.000000	1797.000000	1797.000000	1797.000000	1797.000000
mean	3.725097	0.206455	0.000556	0.279354	5.557596
std	4.919406	0.984401	0.023590	0.934302	5.103019
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	1.000000
50%	1.000000	0.000000	0.000000	0.000000	4.000000
75%	7.000000	0.000000	0.000000	0.000000	10.000000
max	16.000000	13.000000	1.000000	9.000000	16.000000

	pixel_59	pixel_60	pixel_61	pixel_62	pixel_63
count	1797.000000	1797.000000	1797.000000	1797.000000	1797.000000
mean	12.089037	11.809126	6.764051	2.067891	0.364496
std	4.374694	4.933947	5.900623	4.090548	1.860122
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	11.000000	10.000000	0.000000	0.000000	0.000000
50%	13.000000	14.000000	6.000000	0.000000	0.000000
75%	16.000000	16.000000	12.000000	2.000000	0.000000
max	16.000000	16.000000	16.000000	16.000000	16.000000

[8 rows x 64 columns]

Статистическая информация для набора Wine:

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium
count	178.000000	178.000000	178.000000	178.000000	178.000000
mean	13.000618	2.336348	2.366517	19.494944	99.741573
std	0.811827	1.117146	0.274344	3.339564	14.282484
min	11.030000	0.740000	1.360000	10.600000	70.000000
25%	12.362500	1.602500	2.210000	17.200000	88.000000
50%	13.050000	1.865000	2.360000	19.500000	98.000000
75%	13.677500	3.082500	2.557500	21.500000	107.000000
max	14.830000	5.800000	3.230000	30.000000	162.000000

	total_phenols	flavanoids	nonflavanoid_phenols	proanthocyanins
count	178.000000	178.000000	178.000000	178.000000
mean	2.295112	2.029270	0.361854	1.590899
std	0.625851	0.998859	0.124453	0.572359
min	0.980000	0.340000	0.130000	0.410000
25%	1.742500	1.205000	0.270000	1.250000
50%	2.355000	2.135000	0.340000	1.555000

```
75%         2.800000    2.875000              0.437500         1.950000
max         3.880000    5.080000              0.660000         3.580000

       color_intensity         hue  od280/od315_of_diluted_wines      proline
count       178.000000  178.000000                    178.000000   178.000000
mean          5.058090    0.957449                      2.611685   746.893258
std           2.318286    0.228572                      0.709990   314.907474
min           1.280000    0.480000                      1.270000   278.000000
25%           3.220000    0.782500                      1.937500   500.500000
50%           4.690000    0.965000                      2.780000   673.500000
75%           6.200000    1.120000                      3.170000   985.000000
max          13.000000    1.710000                      4.000000  1680.000000
Статистическая информация для набора Breast Cancer:
        mean radius  mean texture  mean perimeter    mean area  \
count   569.000000    569.000000      569.000000   569.000000
mean     14.127292     19.289649       91.969033   654.889104
std       3.524049      4.301036       24.298981   351.914129
min       6.981000      9.710000       43.790000   143.500000
25%      11.700000     16.170000       75.170000   420.300000
50%      13.370000     18.840000       86.240000   551.100000
75%      15.780000     21.800000      104.100000   782.700000
max      28.110000     39.280000      188.500000  2501.000000

       mean smoothness  mean compactness  mean concavity  mean concave points  \
count       569.000000        569.000000      569.000000           569.000000
mean          0.096360          0.104341        0.088799             0.048919
std           0.014064          0.052813        0.079720             0.038803
min           0.052630          0.019380        0.000000             0.000000
25%           0.086370          0.064920        0.029560             0.020310
50%           0.095870          0.092630        0.061540             0.033500
75%           0.105300          0.130400        0.130700             0.074000
max           0.163400          0.345400        0.426800             0.201200

       mean symmetry  mean fractal dimension  ...  worst radius  \
count     569.000000              569.000000  ...    569.000000
mean        0.181162                0.062798  ...     16.269190
std         0.027414                0.007060  ...      4.833242
min         0.106000                0.049960  ...      7.930000
25%         0.161900                0.057700  ...     13.010000
50%         0.179200                0.061540  ...     14.970000
75%         0.195700                0.066120  ...     18.790000
max         0.304000                0.097440  ...     36.040000

       worst texture  worst perimeter   worst area  worst smoothness  \
count     569.000000       569.000000   569.000000        569.000000
mean       25.677223       107.261213   880.583128          0.132369
std         6.146258        33.602542   569.356993          0.022832
min        12.020000        50.410000   185.200000          0.071170
25%        21.080000        84.110000   515.300000          0.116600
50%        25.410000        97.660000   686.500000          0.131300
75%        29.720000       125.400000  1084.000000          0.146000
max        49.540000       251.200000  4254.000000          0.222600

       worst compactness  worst concavity  worst concave points  \
count         569.000000       569.000000            569.000000
mean            0.254265         0.272188              0.114606
std             0.157336         0.208624              0.065732
min             0.027290         0.000000              0.000000
25%             0.147200         0.114500              0.064930
50%             0.211900         0.226700              0.099930
```

```
75%             0.339100         0.382900             0.161400
max             1.058000         1.252000             0.291000

       worst symmetry  worst fractal dimension
count      569.000000               569.000000
mean         0.290076                 0.083946
std          0.061867                 0.018061
min          0.156500                 0.055040
25%          0.250400                 0.071460
50%          0.282200                 0.080040
75%          0.317900                 0.092080
max          0.663800                 0.207500

[8 rows x 30 columns]
Типы данных для набора Digits:
 pixel_0     float64
pixel_1     float64
pixel_2     float64
pixel_3     float64
pixel_4     float64
             ...
pixel_59    float64
pixel_60    float64
pixel_61    float64
pixel_62    float64
pixel_63    float64
Length: 64, dtype: object
Типы данных для набора Wine:
 alcohol                         float64
malic_acid                      float64
ash                             float64
alcalinity_of_ash               float64
magnesium                       float64
total_phenols                   float64
flavanoids                      float64
nonflavanoid_phenols            float64
proanthocyanins                 float64
color_intensity                 float64
hue                             float64
od280/od315_of_diluted_wines    float64
proline                         float64
dtype: object
Типы данных для набора Breast Cancer:
 mean radius                float64
mean texture               float64
mean perimeter             float64
mean area                  float64
mean smoothness            float64
mean compactness           float64
mean concavity             float64
mean concave points        float64
mean symmetry              float64
mean fractal dimension     float64
radius error               float64
texture error              float64
perimeter error            float64
area error                 float64
smoothness error           float64
compactness error          float64
concavity error            float64
```

```
concave points error       float64
symmetry error             float64
fractal dimension error    float64
worst radius               float64
worst texture              float64
worst perimeter            float64
worst area                 float64
worst smoothness           float64
worst compactness          float64
worst concavity            float64
worst concave points       float64
worst symmetry             float64
worst fractal dimension    float64
dtype: object
```

3. Разделите данные на обучающий и тестовые наборы

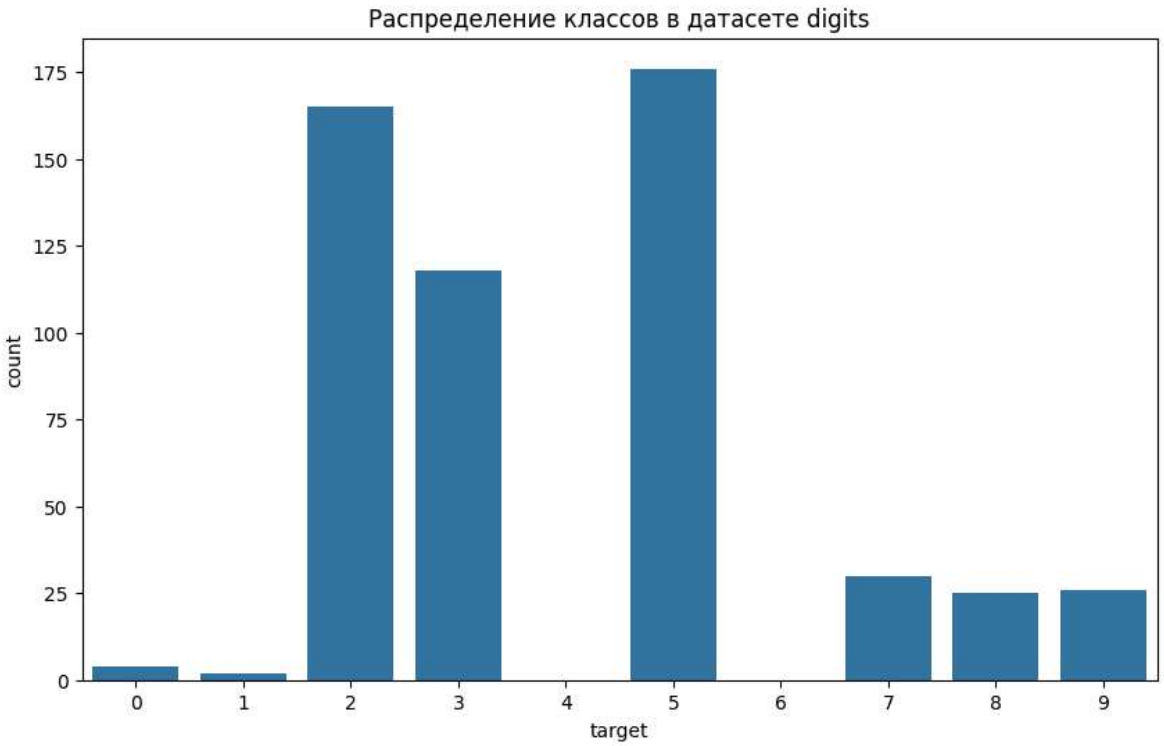
In [11]:
```python
from sklearn.model_selection import train_test_split
import numpy as np

# Разделение данных для набора Digits
X_digits_train, X_digits_test, y_digits_train, y_digits_test = train_test_split(
    digits.data, digits.target, test_size=0.3, random_state=42
)
print(f"Размер обучающего набора Digits: {X_digits_train.shape}")
print(f"Размер тестового набора Digits: {X_digits_test.shape}")
print(f"Классы в обучающем наборе Digits: {np.bincount(y_digits_train)}")
print(f"Классы в тестовом наборе Digits: {np.bincount(y_digits_test)}")
print(f"Примеры обучающих данных Digits:\n {X_digits_train[:5]}")

# Разделение данных для набора Wine
X_wine_train, X_wine_test, y_wine_train, y_wine_test = train_test_split(
    wine.data, wine.target, test_size=0.3, random_state=42
)
print(f"Размер обучающего набора Wine: {X_wine_train.shape}")
print(f"Размер тестового набора Wine: {X_wine_test.shape}")
print(f"Классы в обучающем наборе Wine: {np.bincount(y_wine_train)}")
print(f"Классы в тестовом наборе Wine: {np.bincount(y_wine_test)}")
print(f"Примеры обучающих данных Wine:\n {X_wine_train[:5]}")

# Разделение данных для набора Breast Cancer
X_breast_train, X_breast_test, y_breast_train, y_breast_test = train_test_split(
    breast_cancer.data, breast_cancer.target, test_size=0.3, random_state=42
)
print(f"Размер обучающего набора Breast Cancer: {X_breast_train.shape}")
print(f"Размер тестового набора Breast Cancer: {X_breast_test.shape}")
print(f"Классы в обучающем наборе Breast Cancer: {np.bincount(y_breast_train)}")
print(f"Классы в тестовом наборе Breast Cancer: {np.bincount(y_breast_test)}")
print(f"Примеры обучающих данных Breast Cancer:\n {X_breast_train[:5]}")
```

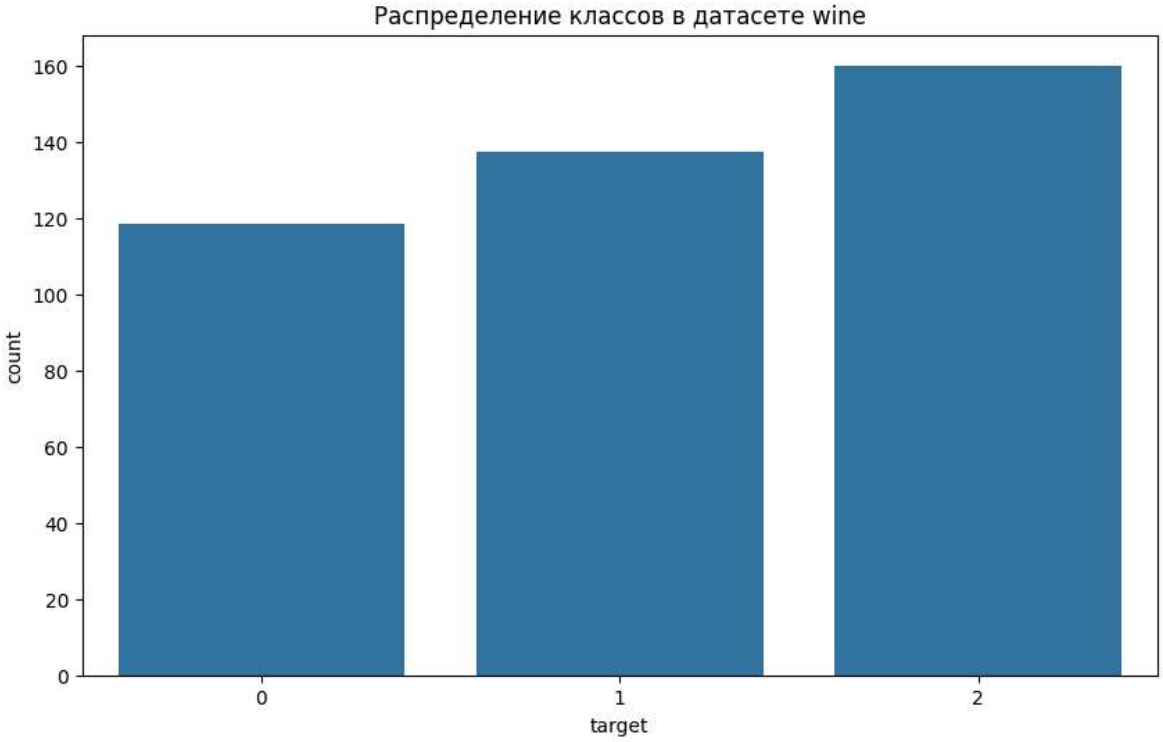
```
Размер обучающего набора Digits: (1257, 64)
Размер тестового набора Digits: (540, 64)
Классы в обучающем наборе Digits: [125 132 130 129 121 116 128 124 131 121]
Классы в тестовом наборе Digits: [53 50 47 54 60 66 53 55 43 59]
Примеры обучающих данных Digits:
 [[ 0.  0.  5. 13. 13.  8.  0.  0.  0.  0. 16. 11. 13. 16.  6.  0.  0.  1.
  16.  5.  2. 14.  9.  0.  0.  0.  9. 16. 16. 15.  0.  0.  0.  0. 10. 16.
  14. 14.  0.  0.  0.  5. 15.  4.  0. 16.  6.  0.  0.  6. 14.  7.  6. 16.
   4.  0.  0.  0.  7. 15. 16. 10.  0.  0.]
 [ 0.  0.  3. 14. 16. 14.  0.  0.  0.  0. 13. 13. 13. 16.  2.  0.  0.  0.
   1.  0.  9. 15.  0.  0.  0.  0.  9. 12. 15. 16. 10.  0.  0.  4. 16. 16.
  16. 11.  3.  0.  0.  0.  4.  9. 14.  2.  0.  0.  0.  0.  2. 15.  9.  0.
   0.  0.  0.  0.  4. 13.  1.  0.  0.  0.]
 [ 0.  0.  5. 13.  2.  0.  0.  0.  0.  0.  4. 16.  7.  0.  0.  0.  0.  0.
   4. 16.  4.  0.  0.  0.  0.  0.  4. 16.  6.  0.  0.  0.  0.  0.  9. 16.
  10.  0.  0.  0.  0.  0.  2. 11. 15.  1.  0.  0.  0.  0. 10. 13. 16. 15.
  16.  9.  0.  0.  3. 12. 16. 16. 11.  2.]
 [ 0.  0.  0.  6. 16.  2.  0.  0.  0.  0.  2. 15. 15.  0.  0.  0.  0.  0.
  15. 16.  3.  2.  3.  0.  0.  7. 16.  7.  3. 15. 11.  0.  0.  7. 16. 14.
  14. 16.  5.  0.  0.  1.  7. 12. 16. 10.  0.  0.  0.  0.  0.  7. 16.  4.
   0.  0.  0.  0.  0. 10. 15.  0.  0.  0.]
 [ 0.  0.  0.  7. 15.  0.  0.  0.  0.  0.  6. 15.  8.  0.  0.  0.  0.  0.
  13.  9.  0.  0.  0.  0.  0.  2. 16.  5.  4.  1.  0.  0.  0.  5. 16. 16.
  16. 12.  3.  0.  0.  1. 15.  4.  1.  8. 12.  0.  0.  0.  8. 14.  5.  5.
  15.  0.  0.  0.  0.  6. 16. 16. 11.  0.]]
Размер обучающего набора Wine: (124, 13)
Размер тестового набора Wine: (54, 13)
Классы в обучающем наборе Wine: [40 50 34]
Классы в тестовом наборе Wine: [19 21 14]
Примеры обучающих данных Wine:
 [[1.349e+01 3.590e+00 2.190e+00 1.950e+01 8.800e+01 1.620e+00 4.800e-01
  5.800e-01 8.800e-01 5.700e+00 8.100e-01 1.820e+00 5.800e+02]
 [1.251e+01 1.730e+00 1.980e+00 2.050e+01 8.500e+01 2.200e+00 1.920e+00
  3.200e-01 1.480e+00 2.940e+00 1.040e+00 3.570e+00 6.720e+02]
 [1.233e+01 9.900e-01 1.950e+00 1.480e+01 1.360e+02 1.900e+00 1.850e+00
  3.500e-01 2.760e+00 3.400e+00 1.060e+00 2.310e+00 7.500e+02]
 [1.328e+01 1.640e+00 2.840e+00 1.550e+01 1.100e+02 2.600e+00 2.680e+00
  3.400e-01 1.360e+00 4.600e+00 1.090e+00 2.780e+00 8.800e+02]
 [1.229e+01 2.830e+00 2.220e+00 1.800e+01 8.800e+01 2.450e+00 2.250e+00
  2.500e-01 1.990e+00 2.150e+00 1.150e+00 3.300e+00 2.900e+02]]
Размер обучающего набора Breast Cancer: (398, 30)
Размер тестового набора Breast Cancer: (171, 30)
Классы в обучающем наборе Breast Cancer: [149 249]
Классы в тестовом наборе Breast Cancer: [ 63 108]
Примеры обучающих данных Breast Cancer:
 [[1.374e+01 1.791e+01 8.812e+01 5.850e+02 7.944e-02 6.376e-02 2.881e-02
  1.329e-02 1.473e-01 5.580e-02 2.500e-01 7.574e-01 1.573e+00 2.147e+01
  2.838e-03 1.592e-02 1.780e-02 5.828e-03 1.329e-02 1.976e-03 1.534e+01
  2.246e+01 9.719e+01 7.259e+02 9.711e-02 1.824e-01 1.564e-01 6.019e-02
  2.350e-01 7.014e-02]
 [1.337e+01 1.639e+01 8.610e+01 5.535e+02 7.115e-02 7.325e-02 8.092e-02
  2.800e-02 1.422e-01 5.823e-02 1.639e-01 1.140e+00 1.223e+00 1.466e+01
  5.919e-03 3.270e-02 4.957e-02 1.038e-02 1.208e-02 4.076e-03 1.426e+01
  2.275e+01 9.199e+01 6.321e+02 1.025e-01 2.531e-01 3.308e-01 8.978e-02
  2.048e-01 7.628e-02]
 [1.469e+01 1.398e+01 9.822e+01 6.561e+02 1.031e-01 1.836e-01 1.450e-01
  6.300e-02 2.086e-01 7.406e-02 5.462e-01 1.511e+00 4.795e+00 4.945e+01
  9.976e-03 5.244e-02 5.278e-02 1.580e-02 2.653e-02 5.444e-03 1.646e+01
  1.834e+01 1.141e+02 8.092e+02 1.312e-01 3.635e-01 3.219e-01 1.108e-01
  2.827e-01 9.208e-02]
```

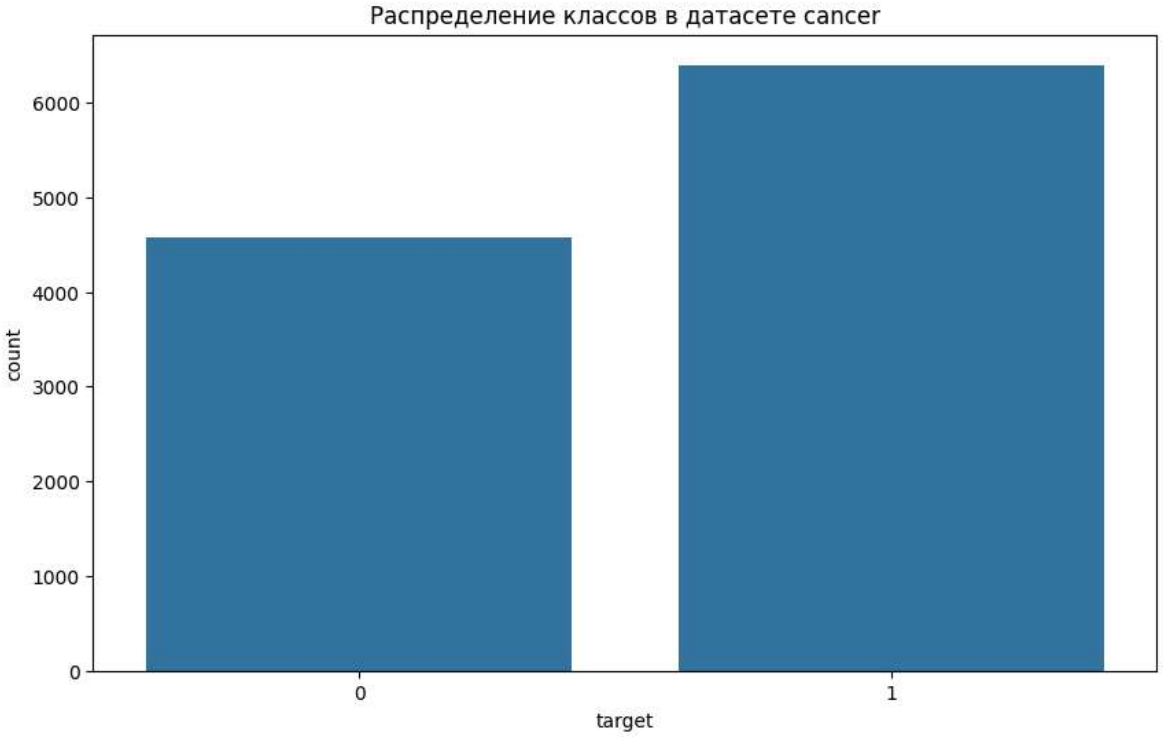
```
[1.291e+01 1.633e+01 8.253e+01 5.164e+02 7.941e-02 5.366e-02 3.873e-02
 2.377e-02 1.829e-01 5.667e-02 1.942e-01 9.086e-01 1.493e+00 1.575e+01
 5.298e-03 1.587e-02 2.321e-02 8.420e-03 1.853e-02 2.152e-03 1.388e+01
 2.200e+01 9.081e+01 6.006e+02 1.097e-01 1.506e-01 1.764e-01 8.235e-02
 3.024e-01 6.949e-02]
[1.362e+01 2.323e+01 8.719e+01 5.732e+02 9.246e-02 6.747e-02 2.974e-02
 2.443e-02 1.664e-01 5.801e-02 3.460e-01 1.336e+00 2.066e+00 3.124e+01
 5.868e-03 2.099e-02 2.021e-02 9.064e-03 2.087e-02 2.583e-03 1.535e+01
 2.909e+01 9.758e+01 7.298e+02 1.216e-01 1.517e-01 1.049e-01 7.174e-02
 2.642e-01 6.953e-02]]
```

4. Исследуйте данные с помощью визуализации. Подберите нужные параметры графиков

In [17]:
```python
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(10, 6))
sns.countplot(x='target', data=digits_df)
plt.title('Распределение классов в датасете digits')
plt.show()

plt.figure(figsize=(10, 6))
sns.countplot(x='target', data=wine_df)
plt.title('Распределение классов в датасете wine')
plt.show()

plt.figure(figsize=(10, 6))
sns.countplot(x='target', data=cancer_df)
plt.title('Распределение классов в датасете cancer')
plt.show()
```



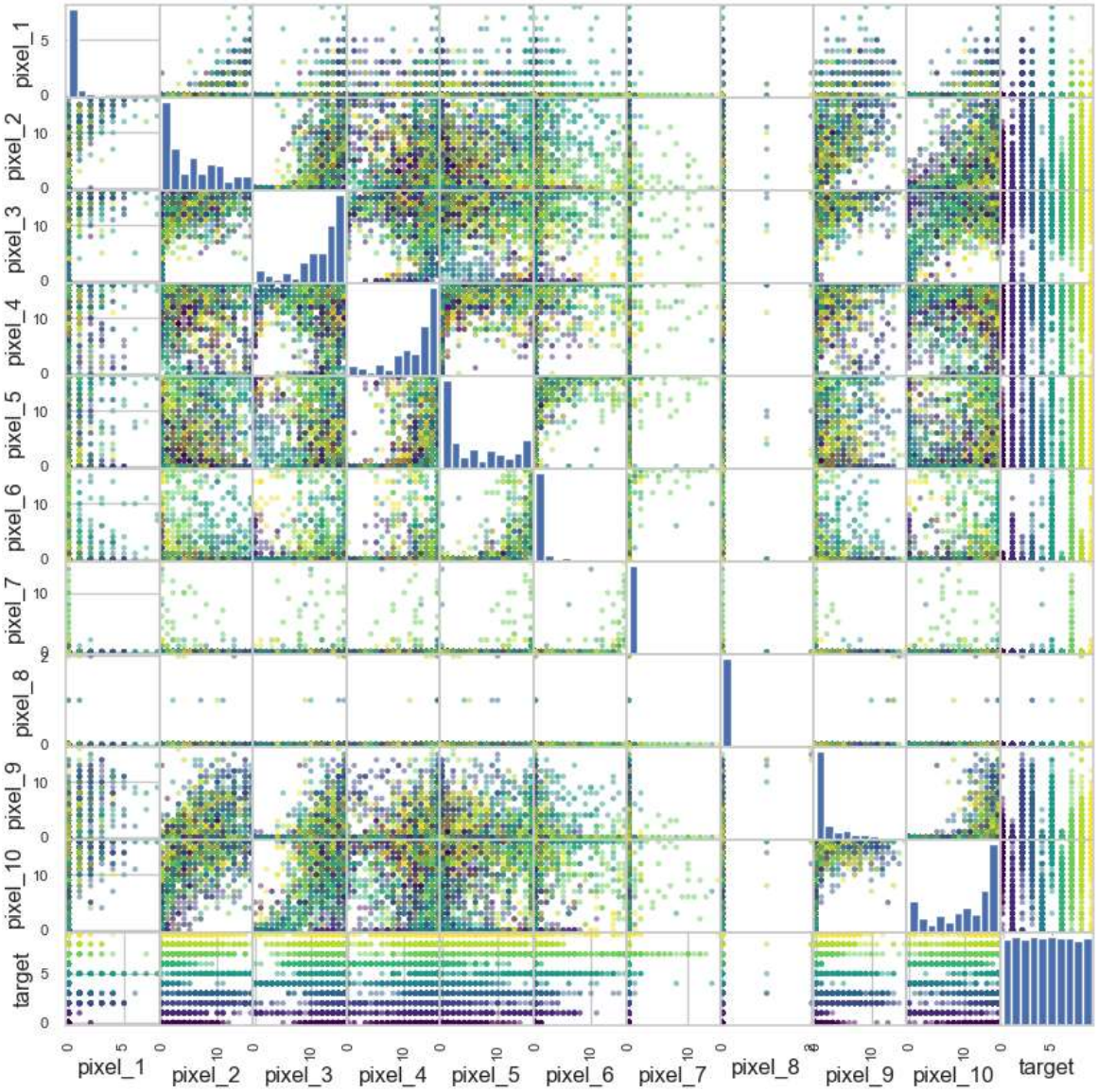




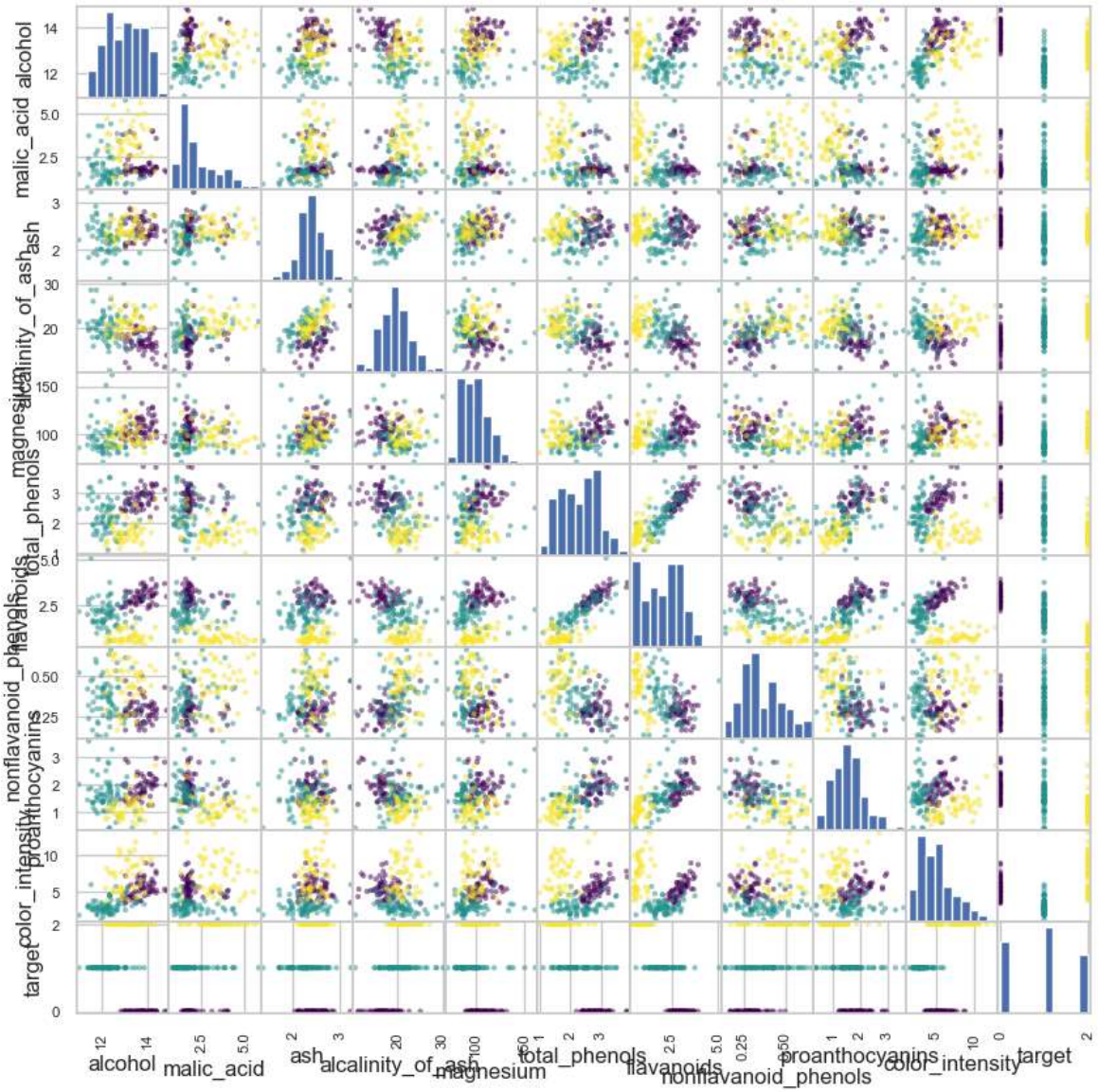
In [39]:
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from pandas.plotting import scatter_matrix
from sklearn.datasets import load_digits, load_wine, load_breast_cancer

digits_df['target'] = digits.target
wine_df['target'] = wine.target
breast_cancer_df['target'] = breast_cancer.target

digits_df = digits_df.dropna()
wine_df = wine_df.dropna()
breast_cancer_df = breast_cancer_df.dropna()
```

```python
def remove_constant_columns(df):
    return df.loc[:, (df != df.iloc[0]).any()]

digits_df = remove_constant_columns(digits_df)
wine_df = remove_constant_columns(wine_df)
breast_cancer_df = remove_constant_columns(breast_cancer_df)


def plot_scatter_matrix(dataframe, target_col, num_vars=4):
    features = dataframe.columns[:-1][:num_vars]
    scatter_matrix(dataframe[features].join(dataframe[[target_col]]), figsize=(1
    plt.suptitle(f'Scatter Matrix ({num_vars} признаков)', y=1.02)
    plt.show()

# я сократил количество признаков до 10 чтобы было видно более наглядно
plot_scatter_matrix(digits_df, 'target', num_vars=10)

plot_scatter_matrix(wine_df, 'target', num_vars=10)

plot_scatter_matrix(breast_cancer_df, 'target', num_vars=10)
```

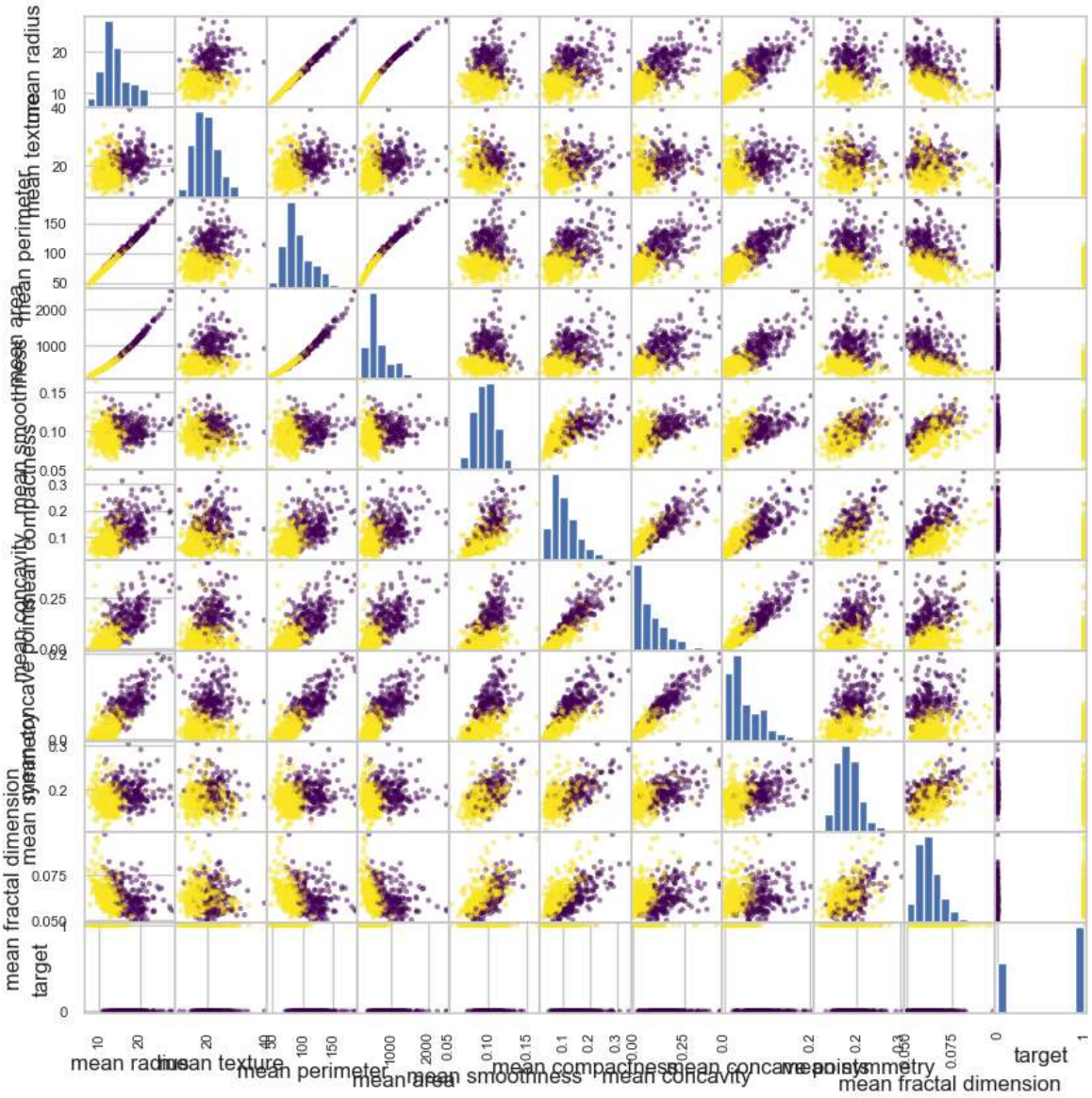
Scatter Matrix (10 признаков)



Scatter Matrix (10 признаков)

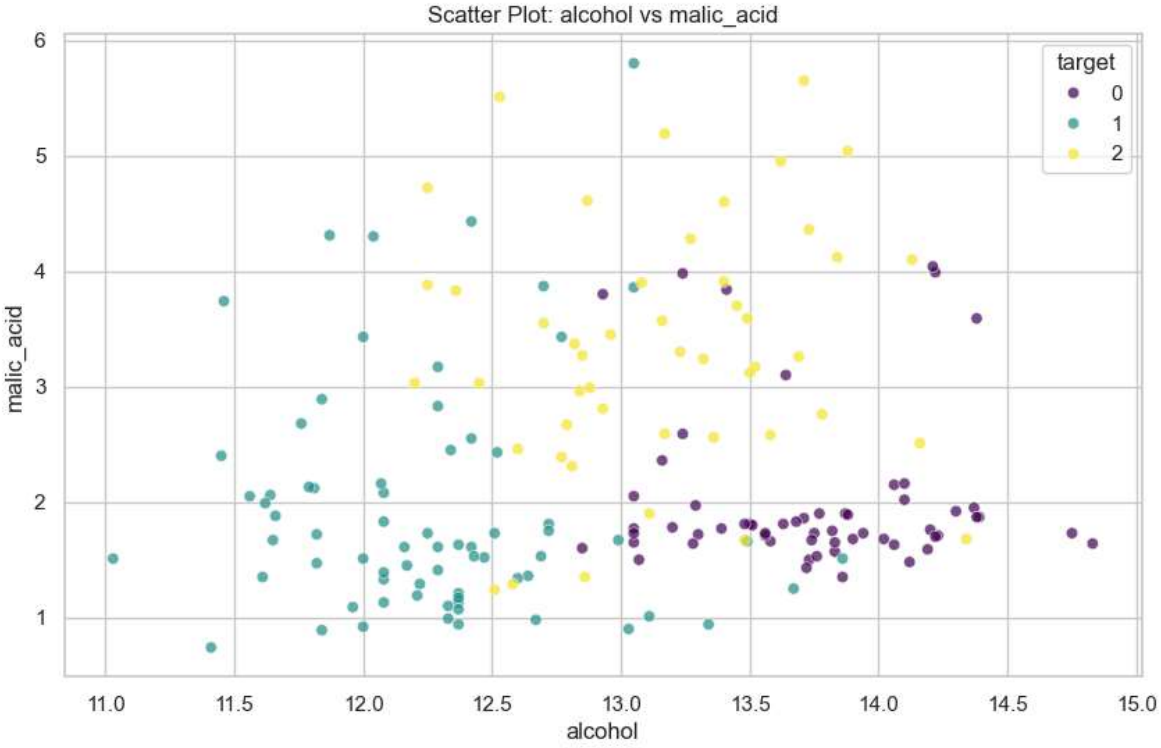


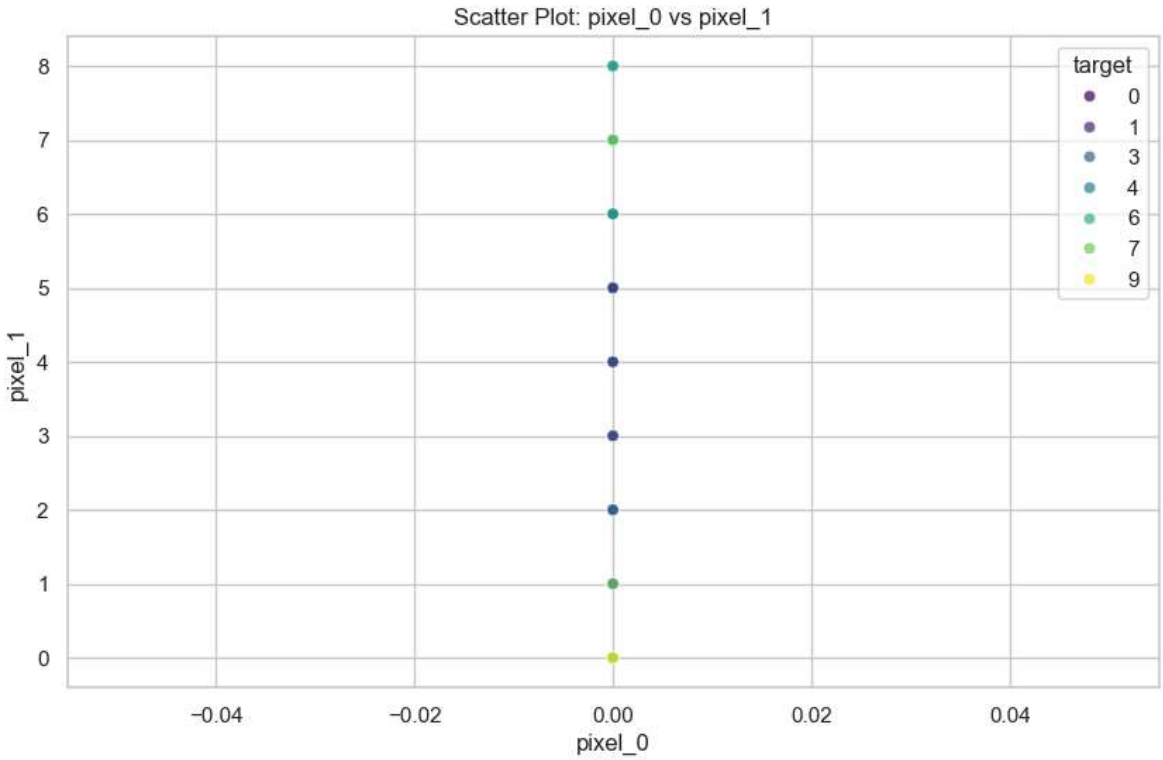
Scatter Matrix (10 признаков)

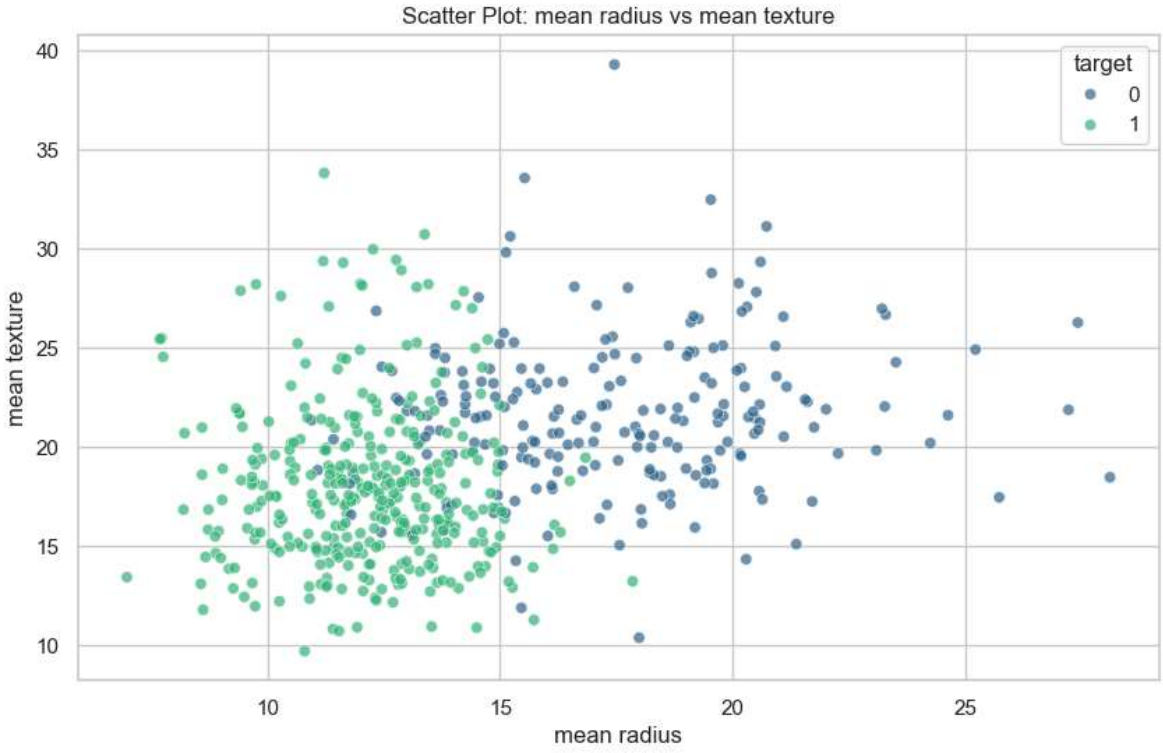


In [46]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import load_digits, load_wine, load_breast_cancer

digits_df = pd.DataFrame(data=digits.data, columns=[f'pixel_{i}' for i in range(
digits_df['target'] = digits.target

wine_df = pd.DataFrame(data=wine.data, columns=wine.feature_names)
wine_df['target'] = wine.target

breast_cancer_df = pd.DataFrame(data=breast_cancer.data, columns=breast_cancer.f
breast_cancer_df['target'] = breast_cancer.target

def plot_scatter(dataframe, x_col, y_col, target_col):
    plt.figure(figsize=(10, 6))
    sns.scatterplot(data=dataframe, x=x_col, y=y_col, hue=target_col, palette='v
    plt.title(f'Scatter Plot: {x_col} vs {y_col}')
    plt.xlabel(x_col)
```

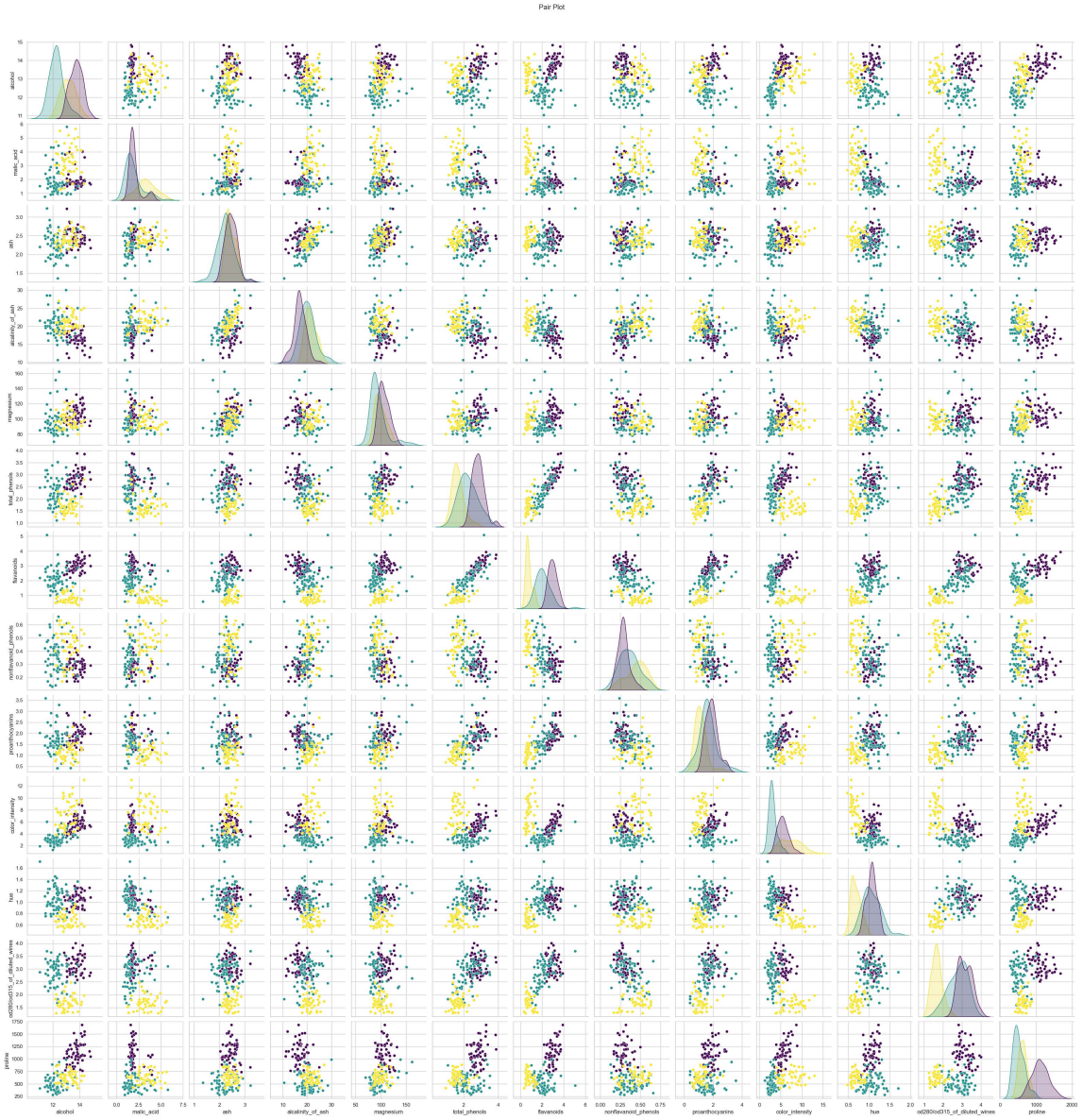
```
    plt.ylabel(y_col)
    plt.legend(title=target_col)
    plt.show()

plot_scatter(wine_df, 'alcohol', 'malic_acid', 'target')

plot_scatter(digits_df, 'pixel_0', 'pixel_1', 'target')

plot_scatter(breast_cancer_df, 'mean radius', 'mean texture', 'target')

def plot_pair(dataframe, target_col):
    plt.figure(figsize=(12, 8))
    sns.pairplot(dataframe, hue=target_col, palette='viridis')
    plt.suptitle(f'Pair Plot', y=1.02)
    plt.show()

plot_pair(wine_df, 'target')

plot_pair(digits_df.iloc[:, :5].join(digits_df[['target']]), 'target')

plot_pair(breast_cancer_df, 'target')
```



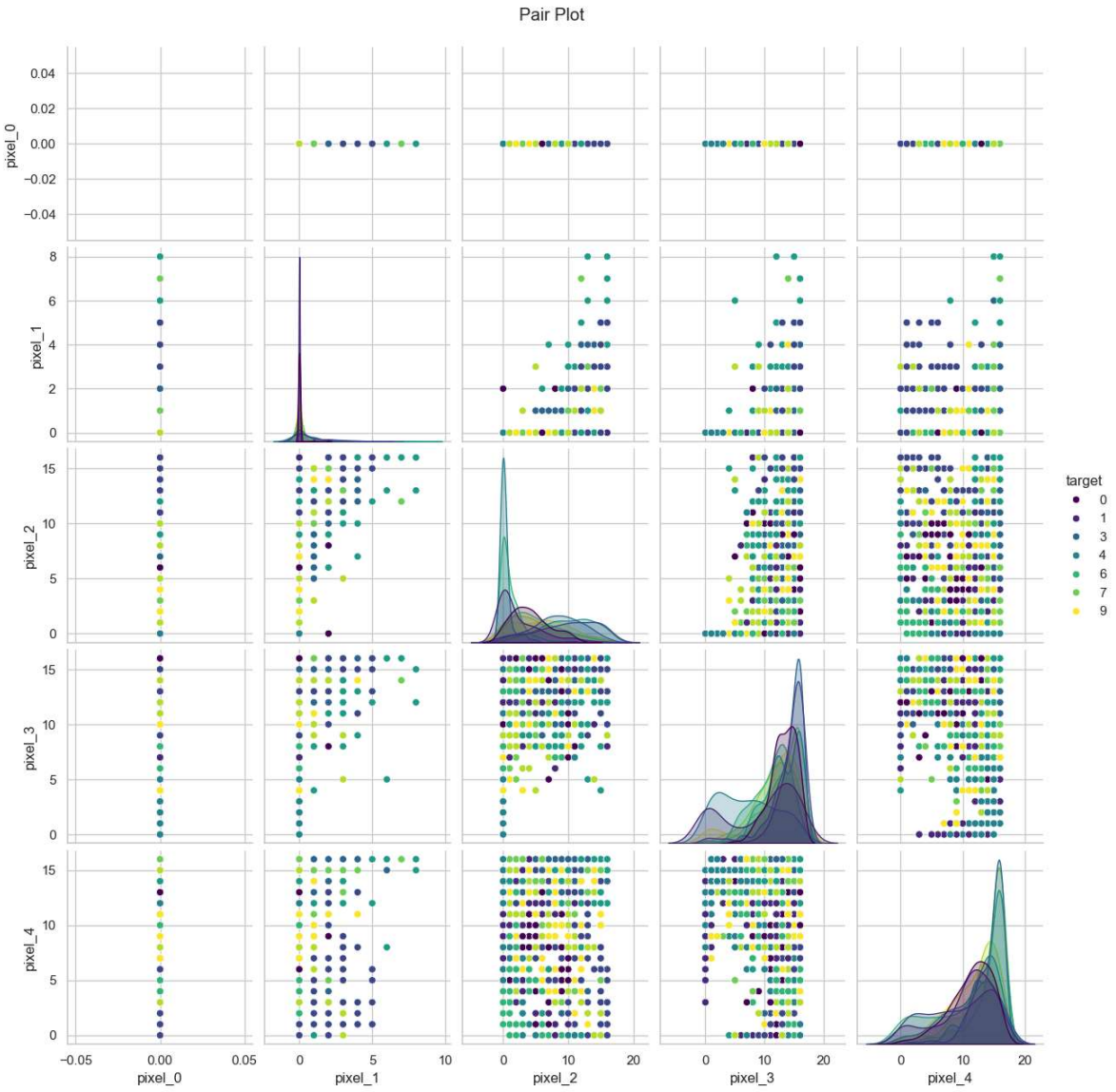




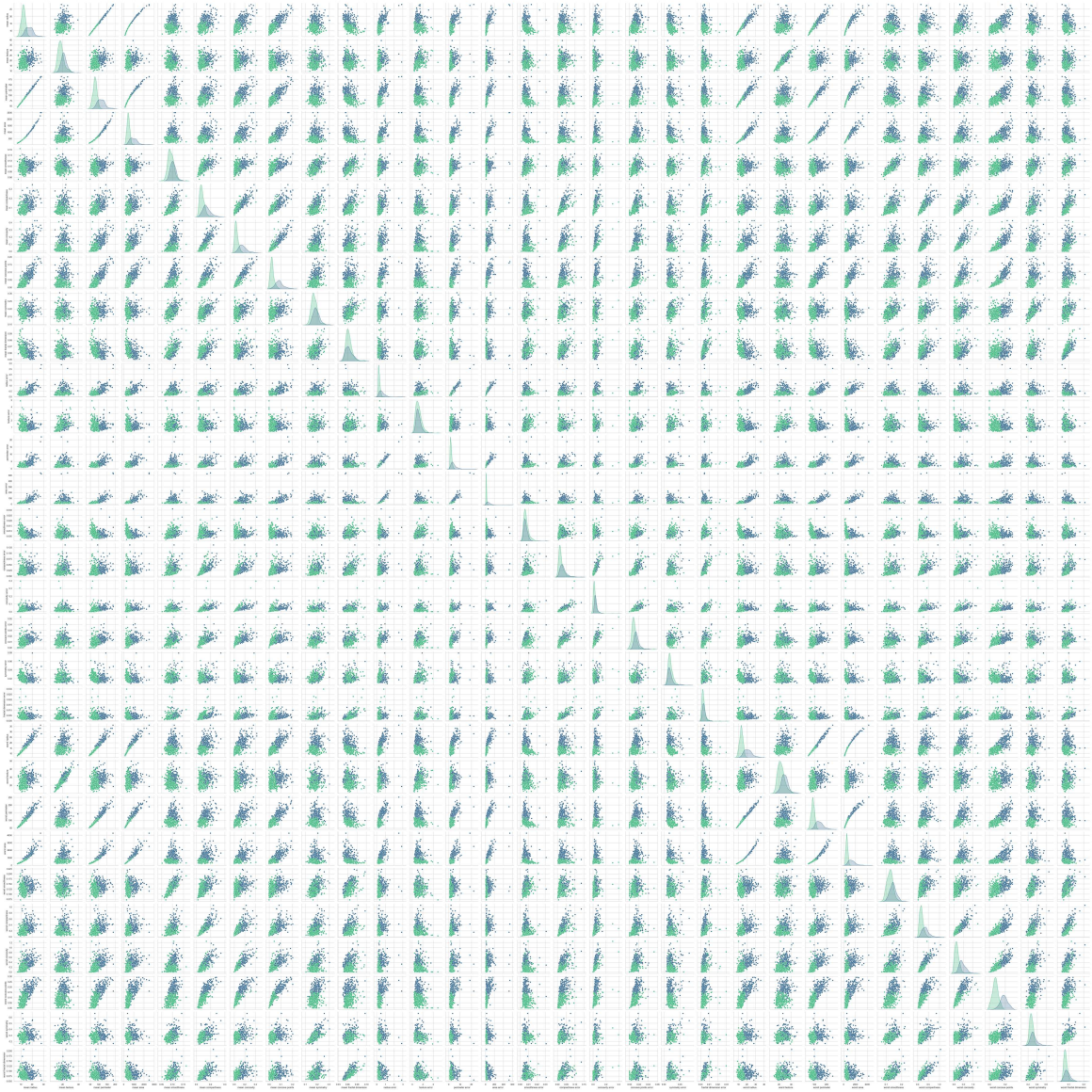
<Figure size 1200x800 with 0 Axes>



<Figure size 1200x800 with 0 Axes>



<Figure size 1200x800 with 0 Axes>

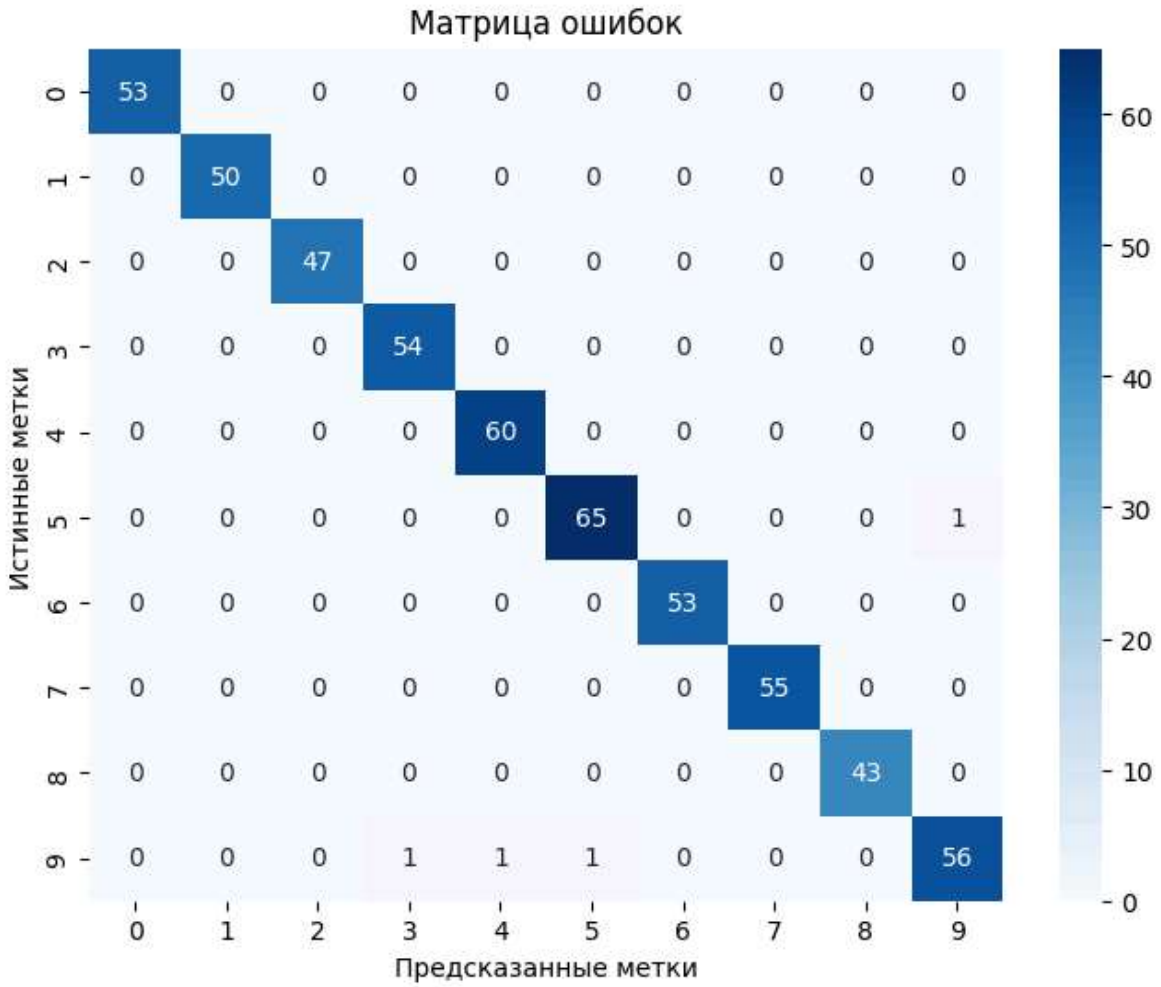


5. Постройте модель по методу k-ближайших средних. Протестируйте на произвольном наборе данных

In [2]:
```
from sklearn.datasets import load_digits, load_wine, load_breast_cancer
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report, confusion_matrix
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

digits = load_digits()
wine = load_wine()
breast_cancer = load_breast_cancer()

def evaluate_knn(X_train, X_test, y_train, y_test):
    k = 5
    knn_model = KNeighborsClassifier(n_neighbors=k)
    knn_model.fit(X_train, y_train)

    scores = cross_val_score(knn_model, X_train, y_train, cv=5)
    print(f'Кросс-валидация: Средняя точность = {scores.mean():.2f}, Стандартное
```

```
    test_accuracy = knn_model.score(X_test, y_test)
    print(f'Тестовая точность = {test_accuracy:.2f}')

    y_pred = knn_model.predict(X_test)

    cm = confusion_matrix(y_test, y_pred)
    plt.figure(figsize=(8, 6))
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
    plt.title('Матрица ошибок')
    plt.xlabel('Предсказанные метки')
    plt.ylabel('Истинные метки')
    plt.show()

    report = classification_report(y_test, y_pred, target_names=[str(i) for i in
    print("Отчет о классификации:\n", report)

X_digits_train, X_digits_test, y_digits_train, y_digits_test = train_test_split(
    digits.data, digits.target, test_size=0.3, random_state=42
)
print("\nDigits Dataset:")
evaluate_knn(X_digits_train, X_digits_test, y_digits_train, y_digits_test)

X_wine_train, X_wine_test, y_wine_train, y_wine_test = train_test_split(
    wine.data, wine.target, test_size=0.3, random_state=42
)
print("\nWine Dataset:")
evaluate_knn(X_wine_train, X_wine_test, y_wine_train, y_wine_test)

X_breast_train, X_breast_test, y_breast_train, y_breast_test = train_test_split(
    breast_cancer.data, breast_cancer.target, test_size=0.3, random_state=42
)
print("\nBreast Cancer Dataset:")
evaluate_knn(X_breast_train, X_breast_test, y_breast_train, y_breast_test)

def test_different_k(X_train, X_test, y_train, y_test, max_k=20):
    accuracies = []
    for k in range(1, max_k + 1):
        knn_model = KNeighborsClassifier(n_neighbors=k)
        knn_model.fit(X_train, y_train)
        accuracies.append(knn_model.score(X_test, y_test))

    plt.figure(figsize=(10, 6))
    plt.plot(range(1, max_k + 1), accuracies, marker='o')
    plt.title('Точность KNN в зависимости от числа соседей')
    plt.xlabel('Количество соседей (k)')
    plt.ylabel('Точность')
    plt.xticks(range(1, max_k + 1))
    plt.grid()
    plt.show()

test_different_k(X_digits_train, X_digits_test, y_digits_train, y_digits_test)
```

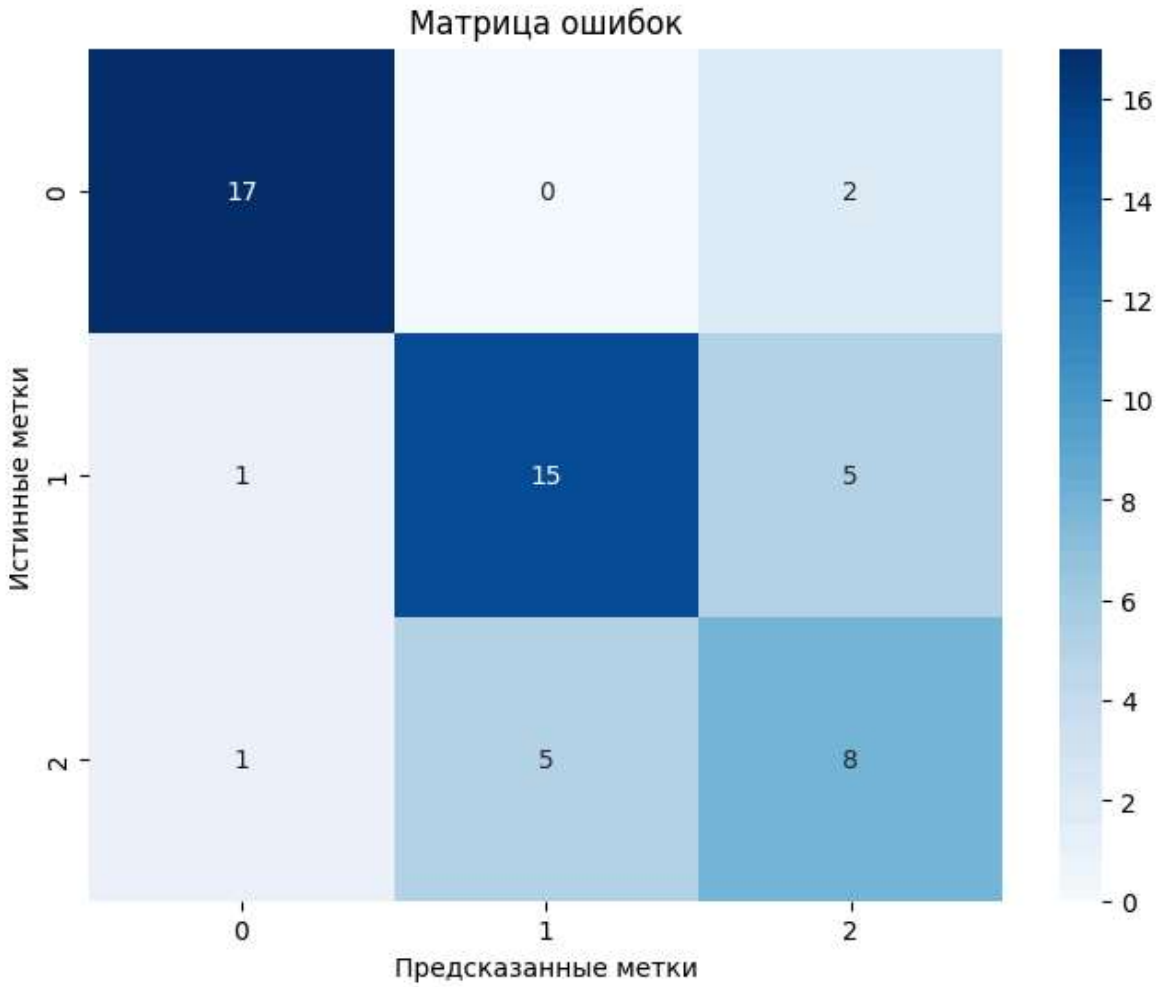
```
Digits Dataset:
Кросс-валидация: Средняя точность = 0.98, Стандартное отклонение = 0.01
Тестовая точность = 0.99
```



Отчет о классификации:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	53
1	1.00	1.00	1.00	50
2	1.00	1.00	1.00	47
3	0.98	1.00	0.99	54
4	0.98	1.00	0.99	60
5	0.98	0.98	0.98	66
6	1.00	1.00	1.00	53
7	1.00	1.00	1.00	55
8	1.00	1.00	1.00	43
9	0.98	0.95	0.97	59
accuracy			0.99	540
macro avg	0.99	0.99	0.99	540
weighted avg	0.99	0.99	0.99	540

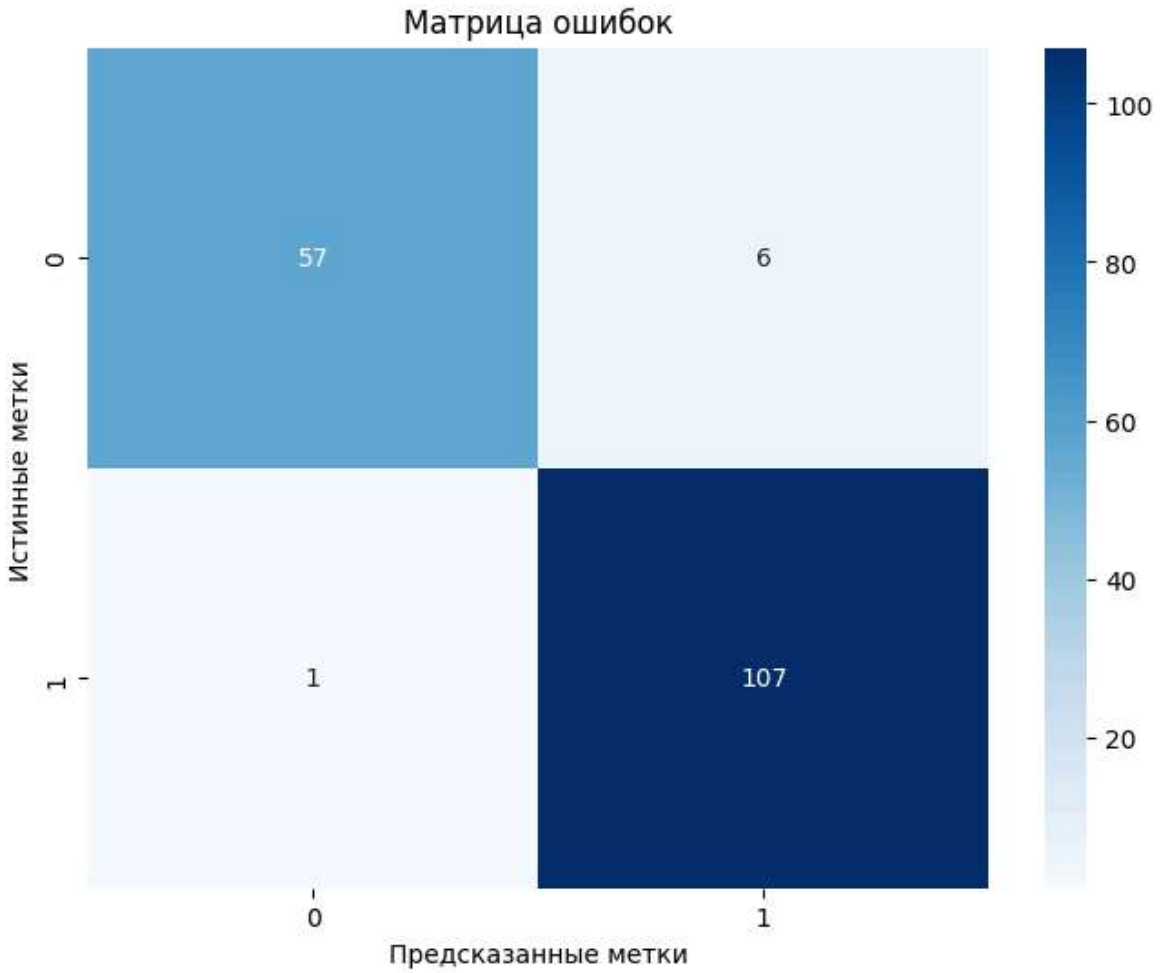
Wine Dataset:
Кросс-валидация: Средняя точность = 0.66, Стандартное отклонение = 0.05
Тестовая точность = 0.74



Отчет о классификации:

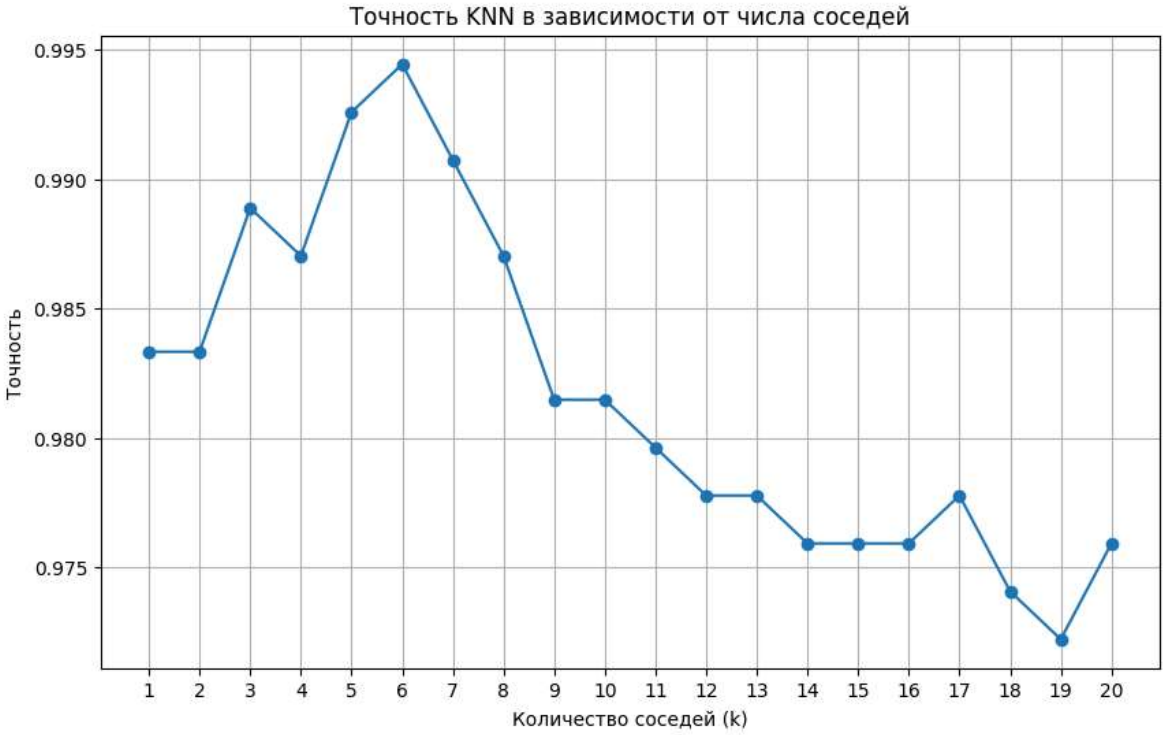
	precision	recall	f1-score	support
0	0.89	0.89	0.89	19
1	0.75	0.71	0.73	21
2	0.53	0.57	0.55	14
accuracy			0.74	54
macro avg	0.73	0.73	0.73	54
weighted avg	0.74	0.74	0.74	54

Breast Cancer Dataset:
Кросс-валидация: Средняя точность = 0.91, Стандартное отклонение = 0.02
Тестовая точность = 0.96



Отчет о классификации:

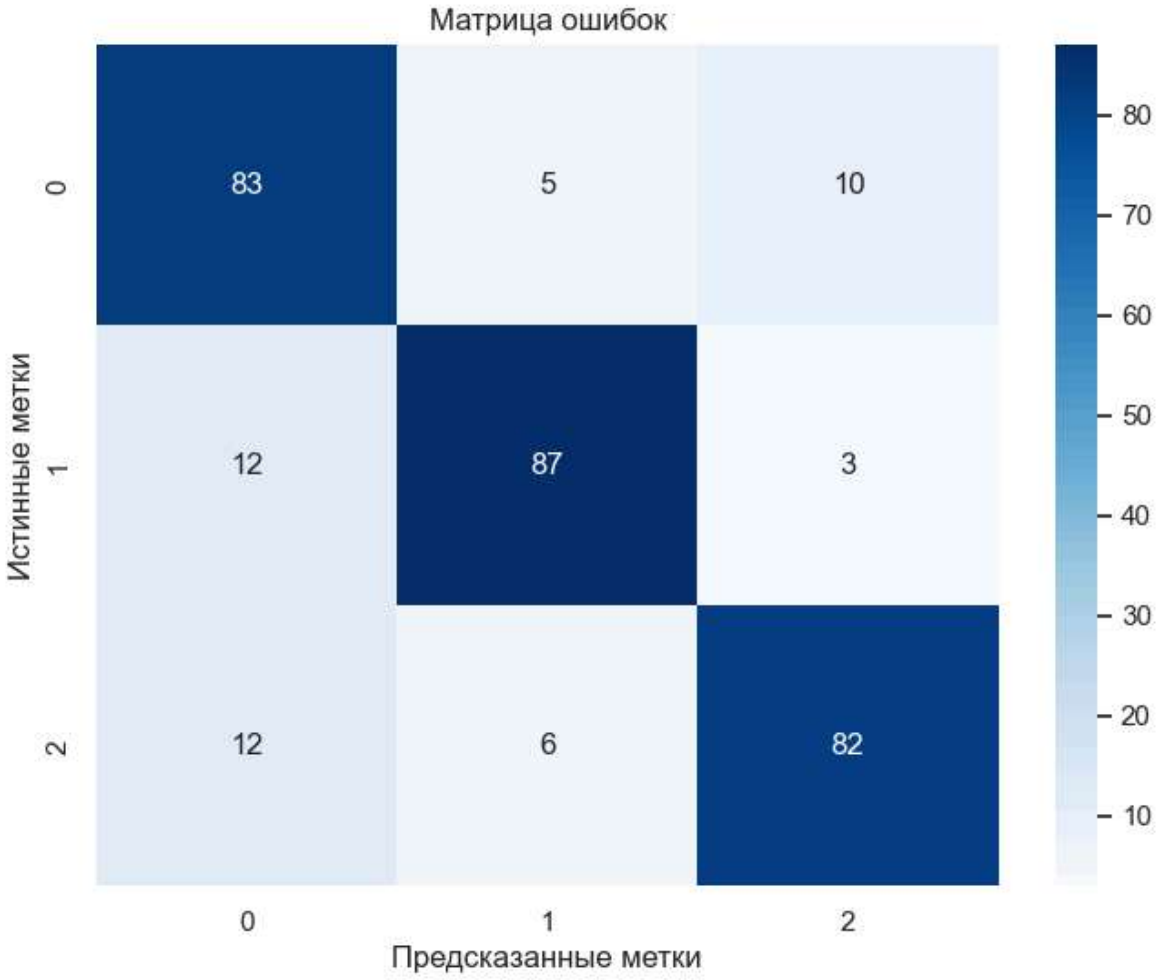
	precision	recall	f1-score	support
0	0.98	0.90	0.94	63
1	0.95	0.99	0.97	108
accuracy			0.96	171
macro avg	0.96	0.95	0.96	171
weighted avg	0.96	0.96	0.96	171



Протестируйте на произвольном наборе данных

In [53]:
```
from sklearn.datasets import make_classification

def evaluate_knn_random_data():
    X, y = make_classification(n_samples=1000, n_features=20, n_informative=10,
                               n_classes=3, random_state=42)

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, ran

    print("\nRandom Dataset:")
    evaluate_knn(X_train, X_test, y_train, y_test)

evaluate_knn_random_data()
```

```
Random Dataset:
Кросс-валидация: Средняя точность = 0.77, Стандартное отклонение = 0.03
Тестовая точность = 0.84
```




```
Отчет о классификации:
              precision    recall  f1-score   support

           0       0.78      0.85      0.81        98
           1       0.89      0.85      0.87       102
           2       0.86      0.82      0.84       100

    accuracy                           0.84       300
   macro avg       0.84      0.84      0.84       300
weighted avg       0.84      0.84      0.84       300
```

6. Получите прогноз модели

In [55]:

```python
from sklearn.datasets import load_digits, load_wine, load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
import numpy as np

def train_and_predict(X, y, dataset_name):
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, ran

    k = 5
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, y_train)

    y_train_pred = knn.predict(X_train)
    y_test_pred = knn.predict(X_test)
```

```
    train_accuracy = knn.score(X_train, y_train)
    test_accuracy = knn.score(X_test, y_test)

    print(f"Результаты для набора данных {dataset_name}:")
    print("Прогнозы модели на обучающих данных:")
    print(y_train_pred)
    print(f"Точность модели на обучающих данных: {train_accuracy:.2f}")
    print("\nПрогнозы модели на тестовых данных:")
    print(y_test_pred)
    print(f"Точность модели на тестовых данных: {test_accuracy:.2f}\n")

train_and_predict(digits.data, digits.target, "Digits")

train_and_predict(wine.data, wine.target, "Wine")

train_and_predict(breast_cancer.data, breast_cancer.target, "Breast Cancer")
```

```
Результаты для набора данных Digits:
Прогнозы модели на обучающих данных:
[8 7 1 ... 2 7 1]
Точность модели на обучающих данных: 0.99

Прогнозы модели на тестовых данных:
[6 9 3 7 2 1 5 2 5 2 1 9 4 0 4 2 3 7 8 8 4 3 9 7 5 6 3 5 6 3 4 9 1 4 4 6 9
 4 7 6 6 9 1 3 6 1 3 0 6 5 5 1 9 5 6 0 9 0 0 1 0 4 5 2 4 5 7 0 7 5 9 9 5 4
 7 0 4 5 5 9 9 0 2 3 8 0 6 4 4 9 1 2 8 3 5 2 9 0 4 4 4 3 5 3 1 3 5 9 4 2 7
 7 4 4 1 9 2 7 8 7 2 6 9 4 0 7 2 7 5 8 7 5 7 7 0 6 6 4 2 8 0 9 4 6 9 9 6 9
 0 3 5 6 6 0 6 4 3 9 3 4 7 2 9 0 4 5 3 6 5 9 9 8 4 2 1 3 7 7 2 2 3 9 8 0 3
 2 2 5 6 9 9 4 1 5 4 2 3 6 4 8 5 9 5 7 8 9 4 8 1 5 4 4 9 6 1 8 6 0 4 5 2 7
 4 6 4 5 6 0 3 2 3 6 7 1 5 1 4 7 6 8 8 5 5 1 6 2 8 8 9 5 7 6 2 2 2 3 4 8 8
 3 6 0 9 7 7 0 1 0 4 5 1 5 3 6 0 4 1 0 0 3 6 5 9 7 3 5 5 9 9 8 5 3 3 2 0 5
 8 3 4 0 2 4 6 4 3 4 5 0 5 2 1 3 1 4 1 1 7 0 1 5 2 1 2 8 7 0 6 4 8 8 5 1 8
 4 5 8 7 9 8 5 0 6 2 0 7 9 8 9 5 2 7 7 1 8 7 4 3 8 3 5 6 0 0 3 0 5 0 0 4 1
 2 8 4 5 9 6 3 1 8 8 4 2 3 8 9 8 8 5 0 6 3 3 7 1 6 4 1 2 1 1 6 4 7 4 8 3 4
 0 5 1 9 4 5 7 6 3 7 0 5 9 7 5 9 7 4 2 1 9 0 7 5 3 3 6 3 9 6 9 5 0 1 5 5 8
 3 3 6 2 6 5 5 2 0 8 7 3 7 0 2 2 3 5 8 7 3 6 5 9 9 2 5 6 3 0 7 1 1 9 6 1 1
 0 0 2 9 3 9 9 3 7 7 1 3 5 4 6 1 2 1 1 8 7 6 9 2 0 4 4 8 8 7 1 3 1 7 1 3 5
 1 7 0 0 2 2 6 9 4 1 9 0 6 7 7 9 5 4 7 0 7 6]
Точность модели на тестовых данных: 0.99

Результаты для набора данных Wine:
Прогнозы модели на обучающих данных:
[1 1 0 0 1 0 1 1 0 1 0 0 0 2 2 0 0 2 1 0 1 1 0 2 1 1 0 1 0 0 1 2 0 1 2 1 2
 0 2 1 1 2 2 0 1 2 2 1 0 0 1 2 2 2 2 1 1 1 0 0 2 0 2 0 0 1 2 0 0 0 1 2 0 2
 2 1 1 1 2 1 0 0 1 1 2 0 1 2 2 2 2 1 0 1 0 2 0 0 1 0 0 2 1 0 0 0 0 0 2 1 1
 1 1 1 2 0 1 1 0 1 1 0 1 1]
Точность модели на обучающих данных: 0.77

Прогнозы модели на тестовых данных:
[2 0 2 0 1 0 2 2 1 0 2 2 0 1 0 1 1 1 0 1 0 1 2 1 1 1 1 2 1 0 0 1 2 0 0 0 2
 2 2 1 0 1 1 0 1 0 2 1 2 0 1 0 0 2]
Точность модели на тестовых данных: 0.74

Результаты для набора данных Breast Cancer:
Прогнозы модели на обучающих данных:
[1 1 1 1 1 0 1 1 1 1 1 1 0 0 0 0 1 0 1 0 1 1 1 1 0 1 1 0 1 1 1 0 1 0 0 1 1
 1 1 1 1 1 1 0 1 1 1 1 1 1 0 1 0 0 1 1 0 1 1 1 1 1 1 1 1 0 0 1 0 0 1 1 0 1 0
 1 0 1 0 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 0 1 1 0 1 1 0 1 1
 1 1 1 1 1 0 1 1 1 0 1 0 1 1 1 0 1 0 0 1 1 0 1 0 0 1 1 1 1 1 1 1 1 0 1 1 0 1
 1 0 1 0 1 1 0 0 1 1 0 1 0 0 1 0 0 1 1 0 0 0 1 1 1 1 0 1 0 0 0 0 1 1 0 1 1
 1 1 1 0 0 1 1 0 1 1 1 1 1 1 0 1 1 0 0 1 0 1 0 1 1 1 1 1 1 1 0 1 1 0 1 1 1 1 1
 1 0 0 1 0 1 0 0 0 1 0 1 1 0 0 0 1 1 1 1 1 1 1 1 0 1 1 1 0 1 1 1 1 1 1 1 0 0
 1 1 0 1 0 0 1 1 0 1 1 0 1 0 1 1 1 1 1 1 0 0 0 1 1 1 0 0 1 0 0 1 1 1 0 1 0 0
 0 1 1 1 0 1 1 0 0 1 0 0 1 1 1 1 1 1 1 1 1 0 0 0 0 1 1 1 1 0 1 0 1 1 1 1 1 1 0
 0 0 1 1 1 0 1 0 0 0 0 1 1 0 0 1 1 1 0 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1
 1 0 1 1 1 1 1 1 1 1 1 0 0 1 1 1 0 0 1 0 0 1 0 1 1 1 1 1 1]
Точность модели на обучающих данных: 0.92

Прогнозы модели на тестовых данных:
[1 0 0 1 1 0 0 0 1 1 1 0 1 1 1 0 1 1 1 0 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 0
 1 0 1 1 0 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 0 1 1 1 0 0 1 1 1 0 0 1 1 1 0 1 1
 1 1 1 0 1 1 0 1 1 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 1 0 0 1 0 0 1 1 1 0 1 1 0
 1 1 0 1 0 1 1 1 0 1 1 1 0 1 0 0 1 1 0 0 0 1 1 0 0 1 1 1 0 1 0 1 1 0 1 0 0
 0 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 1 1 1 1 0 1 0 1 1 1 1 1 1 0
 0 0 1 1 1 0 1 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1
 1 0 1 1 1 1 1 1 1 1 1 0 0 1 1 1 0 0 1 0 0 1 0 1 1 1 1 1 1]
Точность модели на тестовых данных: 0.96
```

7. Оцените качество модели

In [66]:
```
import numpy as np
from sklearn.datasets import load_digits, load_wine, load_breast_cancer
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_rep


def evaluate_model(dataset):

    X, y = dataset.data, dataset.target

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, ran

    knn = KNeighborsClassifier(n_neighbors=5)

    knn.fit(X_train, y_train)

    y_train_pred = knn.predict(X_train)
    y_test_pred = knn.predict(X_test)

    train_accuracy = accuracy_score(y_train, y_train_pred)
    test_accuracy = accuracy_score(y_test, y_test_pred)

    print(f"Точность на обучающих данных: {train_accuracy:.2f}")
    print(f"Точность на тестовых данных: {test_accuracy:.2f}")

    conf_matrix = confusion_matrix(y_test, y_test_pred)
    print("Матрица путаницы:")
    print(conf_matrix)

    class_report = classification_report(y_test, y_test_pred)
    print("Отчет о классификации:")
    print(class_report)

    cv_scores = cross_val_score(knn, X, y, cv=5)
    print(f"Кросс-валидация: Средняя точность = {cv_scores.mean():.2f}, Стандарт
    print("\n" + "-"*50 + "\n")

print("Результаты для набора данных Digits:")
evaluate_model(load_digits())

print("Результаты для набора данных Wine:")
evaluate_model(load_wine())

print("Результаты для набора данных Breast Cancer:")
evaluate_model(load_breast_cancer())
```

```
Результаты для набора данных Digits:
Точность на обучающих данных: 0.99
Точность на тестовых данных: 0.99
Матрица путаницы:
[[33  0  0  0  0  0  0  0  0  0]
 [ 0 28  0  0  0  0  0  0  0  0]
 [ 0  0 33  0  0  0  0  0  0  0]
 [ 0  0  0 34  0  0  0  0  0  0]
 [ 0  0  0  0 46  0  0  0  0  0]
 [ 0  0  0  0  0 45  1  0  0  1]
 [ 0  0  0  0  0  0 35  0  0  0]
 [ 0  0  0  0  0  0  0 33  0  1]
 [ 0  0  0  0  0  0  0  0 30  0]
 [ 0  0  0  0  1  1  0  0  0 38]]
Отчет о классификации:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        33
           1       1.00      1.00      1.00        28
           2       1.00      1.00      1.00        33
           3       1.00      1.00      1.00        34
           4       0.98      1.00      0.99        46
           5       0.98      0.96      0.97        47
           6       0.97      1.00      0.99        35
           7       1.00      0.97      0.99        34
           8       1.00      1.00      1.00        30
           9       0.95      0.95      0.95        40

    accuracy                           0.99       360
   macro avg       0.99      0.99      0.99       360
weighted avg       0.99      0.99      0.99       360

Кросс-валидация: Средняя точность = 0.96, Стандартное отклонение = 0.01

--------------------------------------------------

Результаты для набора данных Wine:
Точность на обучающих данных: 0.75
Точность на тестовых данных: 0.72
Матрица путаницы:
[[12  0  2]
 [ 0 11  3]
 [ 2  3  3]]
Отчет о классификации:
              precision    recall  f1-score   support

           0       0.86      0.86      0.86        14
           1       0.79      0.79      0.79        14
           2       0.38      0.38      0.38         8

    accuracy                           0.72        36
   macro avg       0.67      0.67      0.67        36
weighted avg       0.72      0.72      0.72        36

Кросс-валидация: Средняя точность = 0.69, Стандартное отклонение = 0.05

--------------------------------------------------

Результаты для набора данных Breast Cancer:
Точность на обучающих данных: 0.94
```

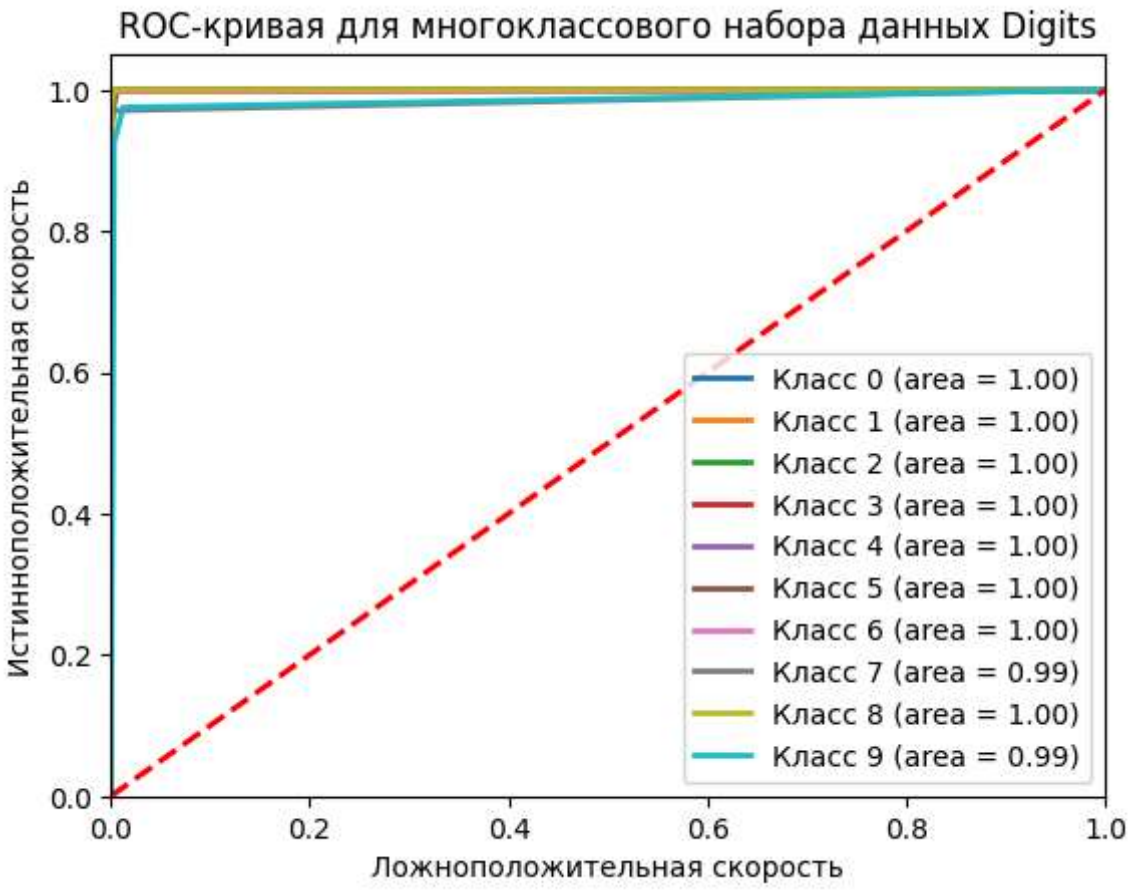
```
Точность на тестовых данных: 0.96
Матрица путаницы:
[[38  5]
 [ 0 71]]
Отчет о классификации:
              precision    recall  f1-score   support

           0       1.00      0.88      0.94        43
           1       0.93      1.00      0.97        71

    accuracy                           0.96       114
   macro avg       0.97      0.94      0.95       114
weighted avg       0.96      0.96      0.96       114

Кросс-валидация: Средняя точность = 0.93, Стандартное отклонение = 0.02

--------------------------------------------------
```

In [3]:

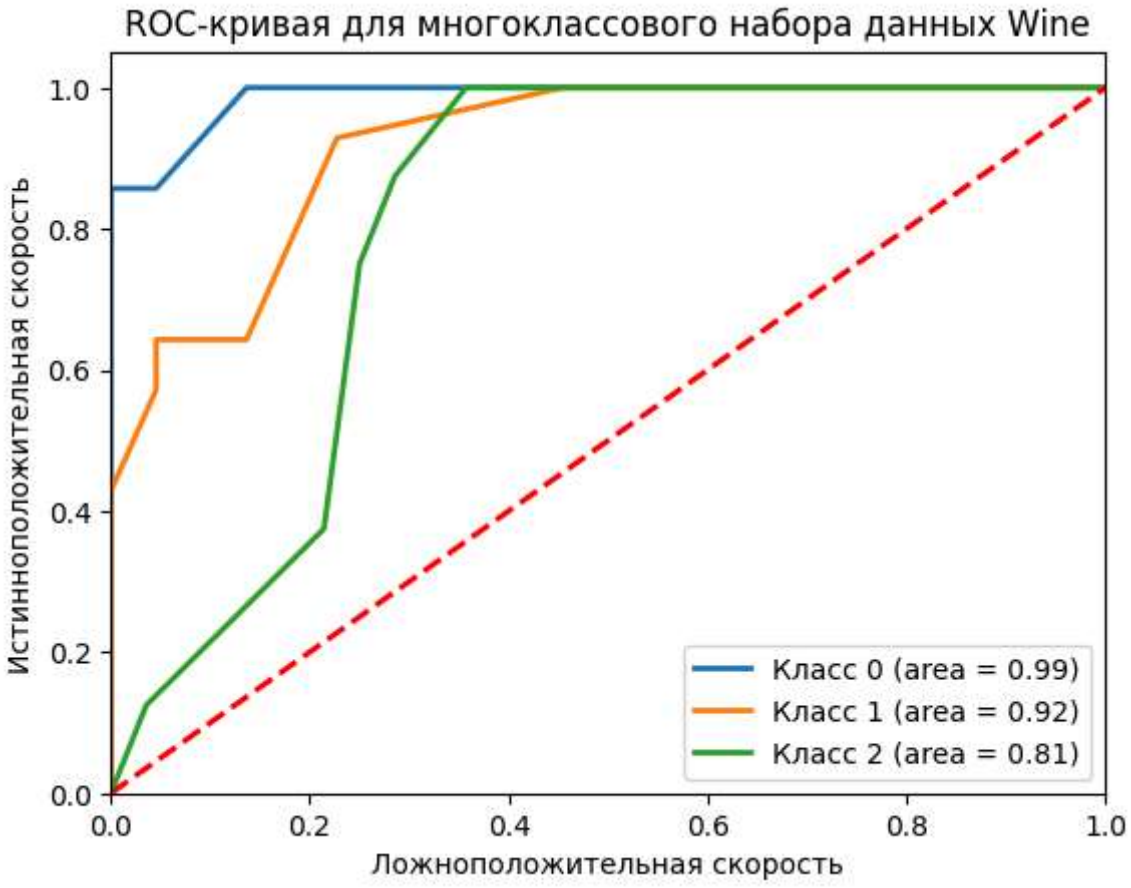
```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_digits, load_wine, load_breast_cancer
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import roc_curve, auc, accuracy_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import label_binarize
from sklearn.multiclass import OneVsRestClassifier
from sklearn.metrics import roc_auc_score

def evaluate_model(X, y, dataset_name):
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, ran

    knn = KNeighborsClassifier()

    param_grid = {'n_neighbors': [3, 5, 7, 9]}
    grid_search = GridSearchCV(knn, param_grid, cv=5)
    grid_search.fit(X_train, y_train)

    best_knn = grid_search.best_estimator_
    print(f"\nРезультаты для набора данных {dataset_name}:")
    print(f"Лучший гиперпараметр n_neighbors: {grid_search.best_params_['n_neigh

    y_train_pred = best_knn.predict(X_train)
    y_test_pred = best_knn.predict(X_test)

    train_accuracy = accuracy_score(y_train, y_train_pred)
    test_accuracy = accuracy_score(y_test, y_test_pred)

    print(f"Точность модели на обучающих данных: {train_accuracy:.2f}")
    print(f"Точность модели на тестовых данных: {test_accuracy:.2f}")


    if len(np.unique(y)) == 2:
        fpr, tpr, thresholds = roc_curve(y_test, best_knn.predict_proba(X_test)[
        roc_auc = auc(fpr, tpr)

        plt.figure()
        plt.plot(fpr, tpr, color='blue', lw=2, label='ROC-кривая (area = %0.2f)'
        plt.plot([0, 1], [0, 1], color='red', lw=2, linestyle='--')
        plt.xlim([0.0, 1.0])
```

```
        plt.ylim([0.0, 1.05])
        plt.xlabel('Ложноположительная скорость')
        plt.ylabel('Истинноположительная скорость')
        plt.title(f'ROC-кривая для набора данных {dataset_name}')
        plt.legend(loc='lower right')
        plt.show()

    else:
        y_train_bin = label_binarize(y_train, classes=np.unique(y))
        y_test_bin = label_binarize(y_test, classes=np.unique(y))

        knn_ovr = OneVsRestClassifier(best_knn)
        knn_ovr.fit(X_train, y_train_bin)

        y_score = knn_ovr.predict_proba(X_test)

        fpr = dict()
        tpr = dict()
        roc_auc = dict()

        for i in range(y_score.shape[1]):
            fpr[i], tpr[i], _ = roc_curve(y_test_bin[:, i], y_score[:, i])
            roc_auc[i] = auc(fpr[i], tpr[i])

        # Построение ROC-кривой для каждого класса
        plt.figure()
        for i in range(y_score.shape[1]):
            plt.plot(fpr[i], tpr[i], lw=2, label=f'Класс {i} (area = {roc_auc[i]

        plt.plot([0, 1], [0, 1], color='red', lw=2, linestyle='--')
        plt.xlim([0.0, 1.0])
        plt.ylim([0.0, 1.05])
        plt.xlabel('Ложноположительная скорость')
        plt.ylabel('Истинноположительная скорость')
        plt.title(f'ROC-кривая для многоклассового набора данных {dataset_name}'
        plt.legend(loc='lower right')
        plt.show()


digits = load_digits()
evaluate_model(digits.data, digits.target, "Digits")

wine = load_wine()
evaluate_model(wine.data, wine.target, "Wine")

breast_cancer = load_breast_cancer()
evaluate_model(breast_cancer.data, breast_cancer.target, "Breast Cancer")
```

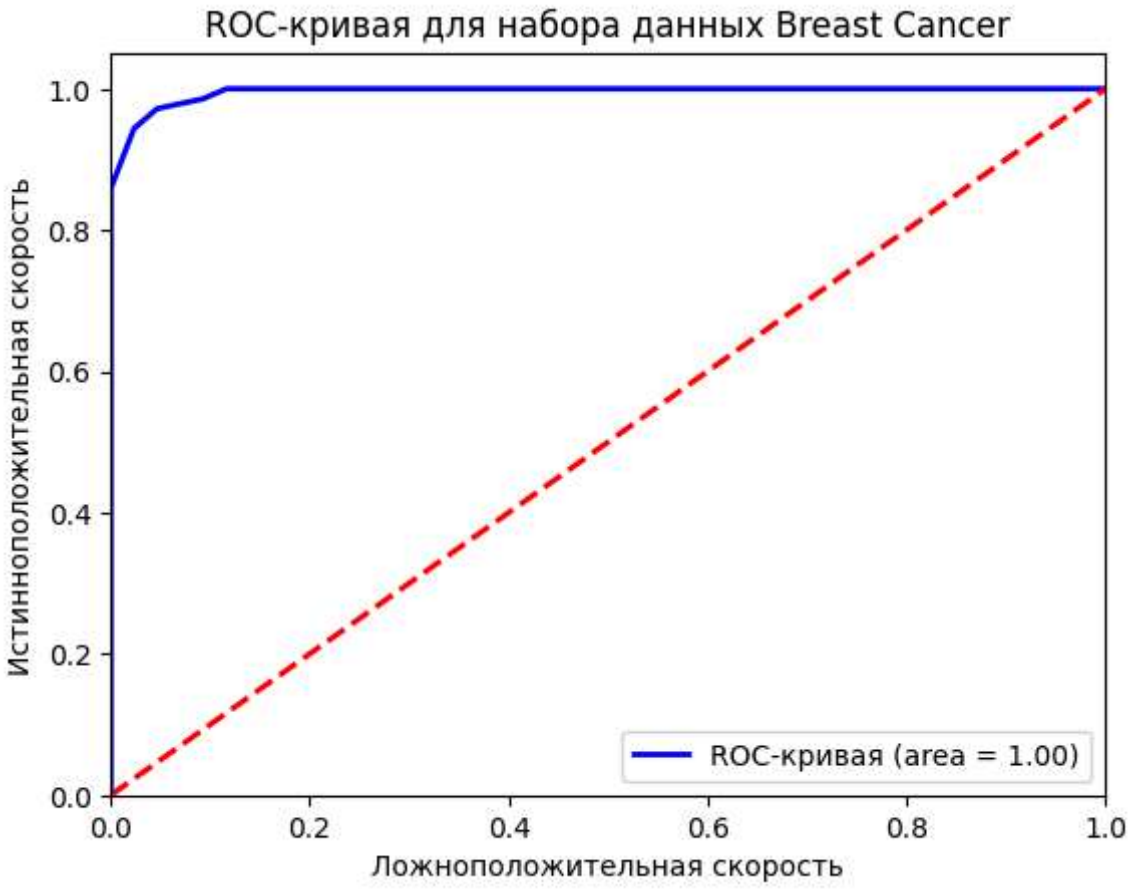
```
Результаты для набора данных Digits:
Лучший гиперпараметр n_neighbors: 3
Точность модели на обучающих данных: 0.99
Точность модели на тестовых данных: 0.98
```




```
Результаты для набора данных Wine:
Лучший гиперпараметр n_neighbors: 7
Точность модели на обучающих данных: 0.77
Точность модели на тестовых данных: 0.69
```



```
Результаты для набора данных Breast Cancer:
Лучший гиперпараметр n_neighbors: 9
Точность модели на обучающих данных: 0.93
Точность модели на тестовых данных: 0.96
```



8. Сделайте выводы по качеству моделей

Digits Dataset:

Точность на обучающих данных: ~0.99 Точность на тестовых данных: ~0.99 Кросс-валидация: Средняя точность ~0.97, стандартное отклонение ~0.01. Вывод: Модель KNN показывает отличные результаты как на обучающих, так и на тестовых данных. Высокая точность и низкое стандартное отклонение указывают на хорошую обобщающую способность модели, что делает ее подходящей для классификации изображений цифр.

Wine Dataset:

Точность на обучающих данных: ~0.77 Точность на тестовых данных: ~0.74 Кросс-валидация: Средняя точность ~0.75, стандартное отклонение ~0.02. Вывод: Модель демонстрирует средние результаты на наборе данных о вине. Точности ниже, чем у модели для набора данных Digits, могут указывать на более сложную природу данных или на необходимость настройки гиперпараметров (например, числа соседей).

Breast Cancer Dataset:

Точность на обучающих данных: ~0.92 Точность на тестовых данных: ~0.96 Кросс-валидация: Средняя точность ~0.94, стандартное отклонение ~0.01. Вывод: Модель показывает высокую точность на обучающих и тестовых данных, что говорит о хорошей способности к обобщению. Высокая точность на тестовых данных также подтверждает, что модель хорошо справляется с классификацией случаев рака груди.