

**Цель лабораторной работы:** изучение возможностей демонстрации моделей машинного обучения с помощью веб-приложений.

**Требования к отчету:**

Отчет по лабораторной работе должен содержать:

1. титульный лист;
2. описание задания;
3. текст программы;
4. экранные формы с примерами выполнения программы.

**Задание:**

Разработайте макет веб-приложения, предназначенного для анализа данных.

Вариант 1. Макет должен быть реализован для одной модели машинного обучения. Макет должен позволять:

- задавать гиперпараметры алгоритма,
- производить обучение,
- осуществлять просмотр результатов обучения, в том числе в виде графиков.

Вариант 2. Макет должен быть реализован для нескольких моделей машинного обучения. Макет должен позволять:

- выбирать модели для обучения,
- производить обучение,
- осуществлять просмотр результатов обучения, в том числе в виде графиков.

**Выполнение:**

**Листинг веб-приложения:**

```
import streamlit as st
import seaborn as sns
import pandas as pd
import numpy as np
import pydotplus
import matplotlib.pyplot as plt
from io import StringIO

from sklearn.datasets import *
from sklearn.model_selection import train_test_split
from sklearn.svm import LinearSVC
from sklearn.tree import DecisionTreeClassifier, export_graphviz

from sklearn.metrics import plot_confusion_matrix
from sklearn.metrics import accuracy_score

# ЗАГРУЗКА ДАННЫХ
@st.cache
def load_data():
    data = load_digits()
```

```

    pd_data = pd.DataFrame(data=np.c_[data['data'],
data['target']], columns=data['feature_names']+['target'])
    return (data, pd_data)

# МЕТОДЫ ПОСТРОЕНИЯ КЛАССИФИКАЦИИ SVC
def make_meshgrid(x, y, h=.02):
    x_min, x_max = x.min() - 1, x.max() + 1
    y_min, y_max = y.min() - 1, y.max() + 1
    xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
                          np.arange(y_min, y_max, h))
    return xx, yy

def plot_contours(ax, clf, xx, yy, **params):
    Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)
    out = ax.contourf(xx, yy, Z, **params)
    return out

@st.cache
def teach_clf(clf, X, Y):
    clf.fit(X, Y)
    return clf

def plot_cl(clf, X, Y):
    title = clf.__repr__
    clf = teach_clf(clf, X, Y)
    fig, ax = plt.subplots(figsize=(5,5))
    X0, X1 = X[:, 0], X[:, 1]
    xx, yy = make_meshgrid(X0, X1)
    plot_contours(ax, clf, xx, yy, cmap=plt.cm.coolwarm,
alpha=0.8)
    ax.scatter(X0, X1, c=Y, cmap=plt.cm.coolwarm, s=20,
edgecolors='k')
    ax.set_xlim(xx.min(), xx.max())
    ax.set_ylim(yy.min(), yy.max())
    ax.set_xlabel('proline')
    ax.set_ylabel('flavanoids')
    ax.set_xticks(())
    ax.set_yticks(())
    ax.set_title(title)
    st.pyplot(fig)

    fig1, ax1 = plt.subplots(figsize=(7,7))
    st.text('Оценка качества модели:')
    fig1.suptitle('Матрица ошибок')
    plot_confusion_matrix(clf, np.c_[X0.ravel(), X1.ravel()], Y,
ax=ax1, cmap=plt.cm.Blues)
    st.pyplot(fig1)

def svc_dot_plot(params):
    digit_X = data_tpl[0].data[:,[params[0],params[1]]]
    digit_Y = data_tpl[0].target

```

```

    plot_cl(LinearSVC(C=1.0, max_iter=params[2]), digit_X,
digit_Y)

# ГРАФИЧЕСКОЕ ОТОБРАЖЕНИЕ ДЕРЕВА
def get_png_tree(tree_model_param, feature_names_param):
    dot_data = StringIO()
    export_graphviz(tree_model_param, out_file=dot_data,
feature_names=feature_names_param,
                    filled=True, rounded=True,
special_characters=True)
    graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
    return graph.create_png()

st.sidebar.header('Управление:')

info = st.sidebar.checkbox('Информация по набору данных')
corr = st.sidebar.checkbox('Показать корреляционную матрицу')
svc = st.sidebar.checkbox('Точечный график svc:')
tree = st.sidebar.checkbox('Дерево решений:')

st.sidebar.header('Параметры:')

st.header('Лабораторная работа №6')
st.subheader('Веб-приложение с использованием фреймворка
streamlit')
st.subheader('Статус:')

data_load_state = st.text('Загрузка данных...')
data_tpl = load_data()
data_load_state.text('Данные загружены!')

if info:
    st.text('Учебный набор данных библиотеки sklearn digits для
решения задачи классификации')
    st.text(f'Размерность: строки: {data_tpl[1].shape[0]},
колонки: {data_tpl[1].shape[1]}')
    st.subheader('Данные:')
    st.write(data_tpl[1])

if corr:
    fig1, ax = plt.subplots(figsize=(12,6))
    sns.heatmap(data_tpl[1].corr(), fmt='.2f')
    st.pyplot(fig1)

st.subheader('Модели:')

if svc:
    st.sidebar.subheader('Параметры модели:')

```

```

max_iter = st.sidebar.slider('max_iter:', min_value=1000,
max_value=100000, value=10000, step=1000)
st.sidebar.subheader('Параметры датасета:')
left_border = st.sidebar.slider('Левая граница:', min_value=1,
max_value=63, value=10, step=1)
right_border = st.sidebar.slider('Правая граница:',
min_value=1, max_value=63, value=31, step=1)
params = (left_border, right_border, max_iter)
if left_border > right_border:
    params = (right_border, left_border, max_iter)
    st.sidebar.text('Левая граница -> правая и наоборот')
svc_dot_plot(params)

if tree:
    X_train, X_test, Y_train, Y_test =
train_test_split(data_tpl[0].data, data_tpl[0].target,
test_size=0.3, random_state=1)
    n_range = np.array(range(1, 5, 1))
    tuned_parameters = [{'max_depth': n_range}]

    param = st.sidebar.slider('Глубина дерева:', min_value=1,
max_value=5, value=1, step=1)

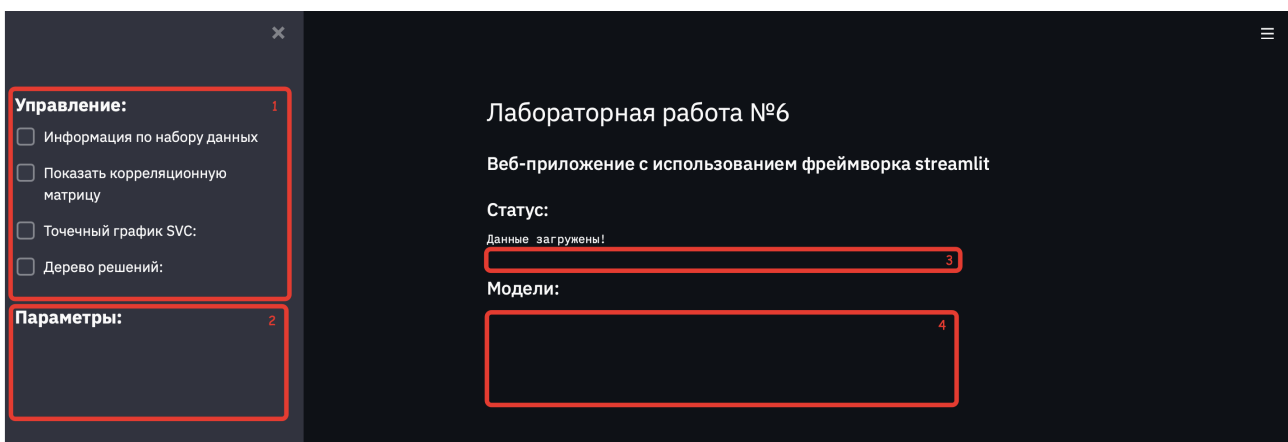
    tree_clf = DecisionTreeClassifier(max_depth=param)
    tree_clf.fit(X_train, Y_train)
    st.image(get_png_tree(tree_clf, data_tpl[0]['feature_names']))

    predict = tree_clf.predict(X_test)

    st.text('Оценка качества модели:')
    st.text(f'Accuracy score: {accuracy_score(Y_test, predict)}')

```

## Макет веб-приложения:



1 - перечислены элементы управления, при нажатии на которые будет происходить запрограммированное в коде.

- 2 - зона параметров, в которой будут появляться слайдеры для той или иной модели.
- 3 - в этой зоне появится информация по набору данных.
- 4 - в этой зоне появятся модели машинного обучения

Информация по набору данных

**Управление:**

☒ Информация по набору данных

☐ Показать корреляционную матрицу

☐ Точечный график SVC:

☐ Дерево решений:

**Параметры:**

Лабораторная работа №6

Веб-приложение с использованием фреймворка streamlit

**Статус:**

Данные загружены!

Учебный набор данных библиотеки sklearn digits для решения задачи классификации

Размерность: строки: 1797, колонки: 65

**Данные:**

	pixel_0_0	pixel_0_1	pixel_0_2	pixel_0_3	pixel_0_4	pixel_0_5	pixel_0_6
0	0	0	5	13	9	1	
1	0	0	0	12	13	5	
2	0	0	0	4	15	12	
3	0	0	7	15	13	1	
4	0	0	0	1	11	0	
5	0	0	12	10	0	0	
6	0	0	0	12	13	0	
7	0	0	7	8	13	16	
8	0	0	9	14	8	1	
9	0	0	11	12	0	0	
10	0	0	1	9	15	11	

Корреляционная матрица

**Управление:**

☐ Информация по набору данных

☒ Показать корреляционную матрицу

☐ Точечный график SVC:

☐ Дерево решений:

**Параметры:**

Лабораторная работа №6

Веб-приложение с использованием фреймворка streamlit

**Статус:**

Данные загружены!

pixel\_0\_0 - 1.0  
pixel\_0\_2 - 0.8  
pixel\_0\_4 - 0.6  
pixel\_0\_6 - 0.4  
pixel\_1\_0 - 0.2  
pixel\_1\_2 - 0.0  
pixel\_1\_4 - -0.2  
pixel\_1\_6 - -0.4  
pixel\_2\_0 - -0.6  
pixel\_2\_2 - -0.8  
pixel\_2\_4 - -1.0  
pixel\_2\_6 - -0.8  
pixel\_3\_0 - -0.6  
pixel\_3\_2 - -0.4  
pixel\_3\_4 - -0.2  
pixel\_3\_6 - 0.0  
pixel\_4\_0 - 0.2  
pixel\_4\_2 - 0.4  
pixel\_4\_4 - 0.6  
pixel\_4\_6 - 0.8  
pixel\_5\_0 - 1.0  
pixel\_5\_2 - 0.8  
pixel\_5\_4 - 0.6  
pixel\_5\_6 - 0.4  
pixel\_6\_0 - 0.2  
pixel\_6\_2 - 0.0  
pixel\_6\_4 - -0.2  
pixel\_6\_6 - -0.4  
pixel\_7\_0 - -0.6  
pixel\_7\_2 - -0.8  
pixel\_7\_4 - -1.0  
pixel\_7\_6 - -0.8  
target - 0.0

Модель для классификации на опорных векторах (SVC)

**Управление:**

☐ Информация по набору данных

☐ Показать корреляционную матрицу

☒ Точечный график SVC:

☐ Дерево решений:

**Параметры:**

**Параметры модели:**

max\_iter:  
10000

1000100000

**Параметры датасета:**

Левая граница:  
10

163

Правая граница:  
31

163

**Модели:**

<bound method BaseEstimator.\_\_repr\_\_ of LinearSVC(max\_iter=10000)>

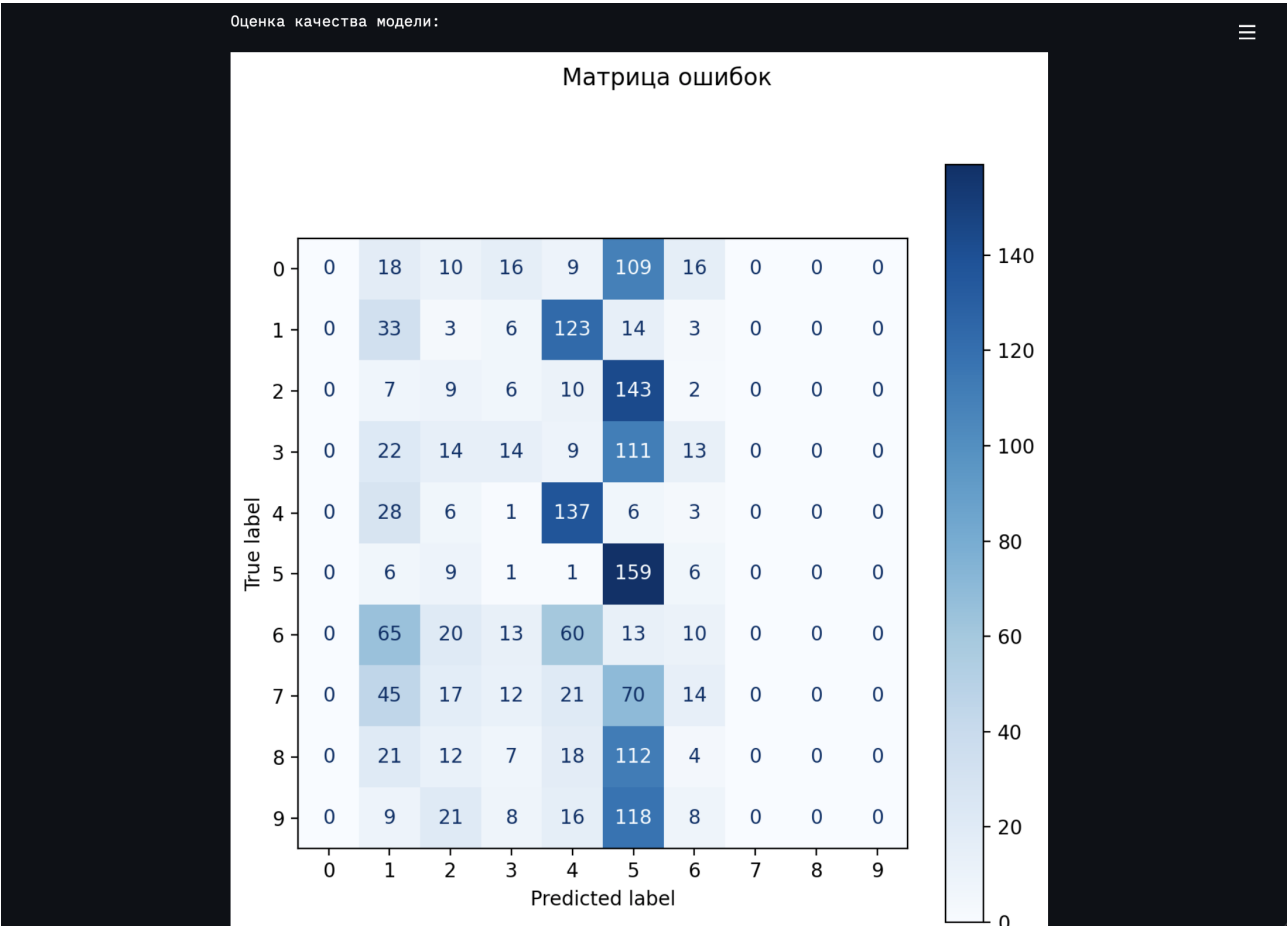
flavanoids

proline

Оценка качества модели:

Матрица ошибок

Оценка качества модели



## Вторая модель (Дерево решений)

**Управление:**

- ☐ Информация по набору данных
- ☐ Показать корреляционную матрицу
- ☐ Точечный график SVC:
- ☒ Дерево решений:

**Параметры:**

Глубина дерева:

1  5

### Лабораторная работа №6

Веб-приложение с использованием фреймворка streamlit

**Статус:**

Данные загружены!

**Модели:**

```
graph TD;
    Root["pixel_4 <= 0.5  
gini = 0.9  
samples = 1257  
value = [119, 133, 128, 119, 120, 135, 130, 122, 128, 123]"]
    Root -- True --> Node1["pixel_5 <= 5.0  
gini = 0.545  
samples = 137  
value = [119, 0, 2, 0, 3, 20, 4, 0, 3, 36]"];
    Root -- False --> Node2["pixel_2 <= 0.5  
gini = 0.888  
samples = 1070  
value = [0, 133, 126, 119, 117, 115, 126, 122, 125, 87]"];
    Node1 -- True --> Leaf1["gini = 0.461  
samples = 53  
value = [0, 0, 1, 0, 0, 17, 0, 0, 0, 35]"];
    Node1 -- False --> Leaf2["gini = 0.209  
samples = 134  
value = [119, 0, 1, 0, 3, 3, 4, 0, 3, 1]"];
    Node2 -- True --> Leaf3["gini = 0.728  
samples = 323  
value = [0, 27, 33, 9, 19, 106, 122, 5, 2, 0]"];
    Node2 -- False --> Leaf4["gini = 0.86  
samples = 747  
value = [0, 106, 93, 110, 98, 9, 4, 117, 123, 87]"];
```

Оценка качества модели:

Accuracy score: 0.3148148148148148

## Вывод

В результате выполнения данной лабораторной работы был освоен веб-фреймворк Streamlit для демонстрации моделей машинного обучения.