Московский государственный технический университет имени Н. Э. Баумана

Факультет «Информатика и системы управления»

Кафедра ИУ5

Отчёт по
лабораторной работе № 6
по курсу Технологии машинного обучения

Подготовила:

Калашникова Анастасия

Группа ИУ5-64Б

1. 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.

2. Выполнение

Текст программы:

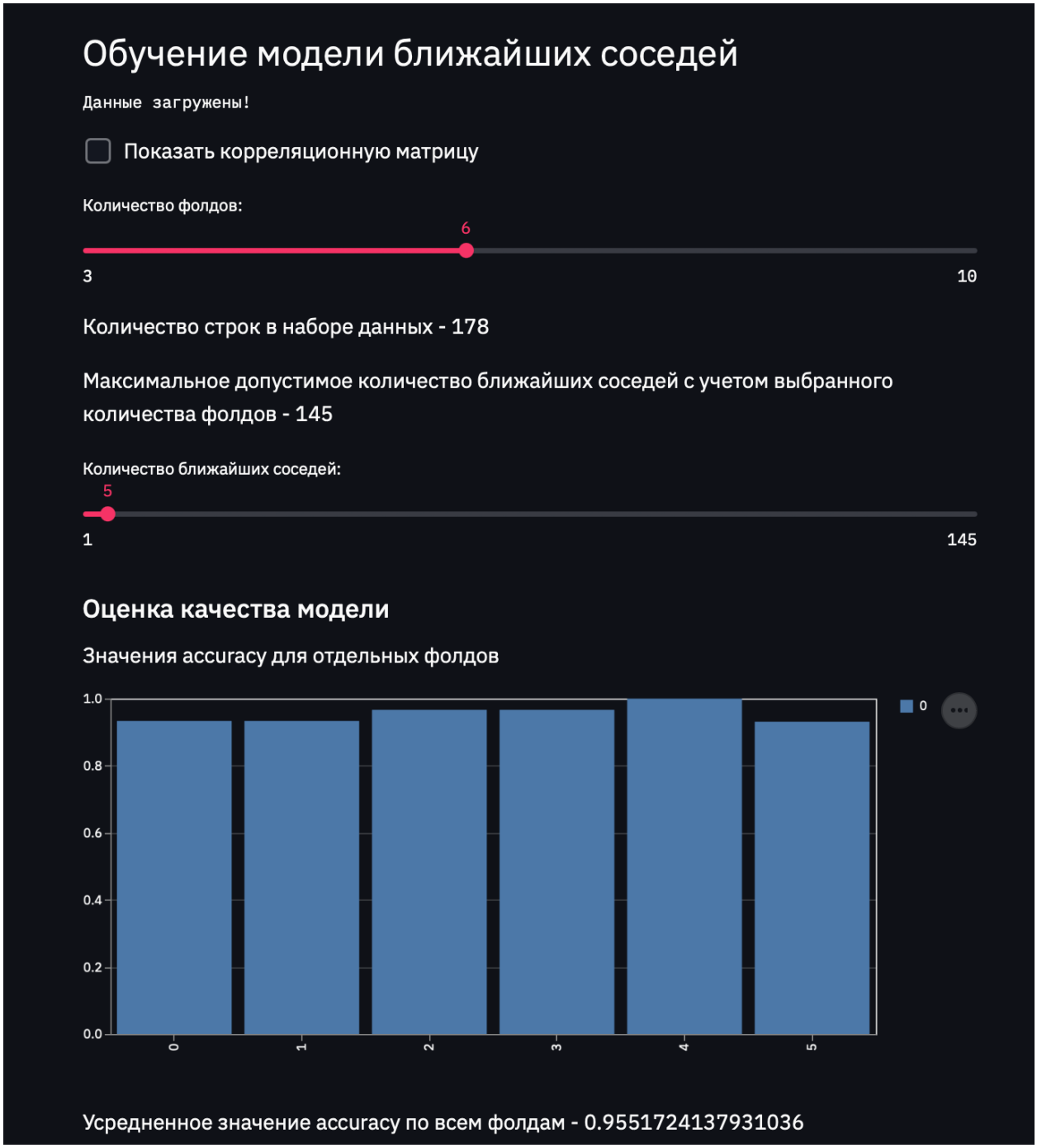
```
import streamlit as st
import seaborn as sns
import pandas as pd
import numpy as np
from sklearn.datasets import *
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import cross_val_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.model_selection import LeavePOut
from sklearn.model_selection import GridSearchCV
import matplotlib.pyplot as plt
```

```
@st.cache
def load_data():
    '''
    Загрузка данных
    '''
    wine = load_wine()
    data = pd.DataFrame(data=np.c_[wine['data']], columns=wine['feature_names'])
    sc = MinMaxScaler()
    wine_sc = sc.fit_transform(wine.data)
    return wine_sc, wine.target, data.shape[0], data

st.header('Обучение модели ближайших соседей')
data_load_state = st.text('Загрузка данных...')
data_X, data_Y, data_len, data = load_data()
data_load_state.text('Данные загружены!')

if st.checkbox('Показать корреляционную матрицу'):
    fig1, ax = plt.subplots(figsize=(10, 5))
    sns.heatmap(data.corr(), annot=True, fmt='.2f')
    st.pyplot(fig1)

cv_slider = st.slider('Количество фолдов:', min_value=3, max_value=10,
value=5, step=1)

# Вычислим количество возможных ближайших соседей
rows_in_one_fold = int(data_len / cv_slider)
```

```
allowed_knn = int(rows_in_one_fold * (cv_slider - 1))

st.write('Количество строк в наборе данных - {}'.format(data_len))

st.write('Максимальное допустимое количество ближайших соседей с учетом выбранного количества фолдов - {}'.format(

    allowed_knn))


cv_knn = st.slider('Количество ближайших соседей:', min_value=1, max_value=allowed_knn, value=5, step=1)


scores = cross_val_score(KNeighborsClassifier(n_neighbors=cv_knn),

                         data_X, data_Y, scoring='accuracy', cv=cv_slider)


st.subheader('Оценка качества модели')

st.write('Значения accuracy для отдельных фолдов')

st.bar_chart(scores)

st.write('Усредненное значение accuracy по всем фолдам - {}'.format(np.mean(scores)))


if st.checkbox('Рассчитать гиперпараметр автоматически:'):

    X_train, X_test, Y_train, Y_test = train_test_split(data_X, data_Y, test_size=0.3, random_state=1)

    n_range = np.array(range(1, allowed_knn, 1))

    tuned_params = [{'n_neighbors': n_range}]


    # lpo = LeavePOut(2)

    clf_gs = GridSearchCV(KNeighborsClassifier(), tuned_params, cv=cv_slider, scoring='accuracy')
```

```
clf_gs.fit(X_train, Y_train)
param = clf_gs.best_params_['n_neighbors']
score_best = clf_gs.best_score_
st.write('Лучшее значение параметра - {}'.format(param))
st.write('Лучшее значение метрики - {}'.format(score_best))
```

Экранные формы с примерами выполнения программы:



Усредненное значение accuracy по всем фолдам - 0.9551724137931036

☑ Рассчитать гиперпараметр автоматически:

Лучшее значение параметра - 14

Лучшее значение метрики - 0.9841269841269842