Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»

(МГТУ им. Н.Э. Баумана)

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Технологии машинного обучения

Отчет по лабораторной работе № 6

Создание веб-приложения для демонстрации моделей машинного обучения.

Студент:

группы ИУ5-64Б

Ведьгун Е.А.

2021 г.

Москва

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.

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

@st.cache
def load_data(df):
    # Сформируем DataFrame
    df_load = pd.DataFrame(data=np.c_[df['data']],
                           columns=df['feature_names'])
    sc = MinMaxScaler()
    df_sc = sc.fit_transform(df.data)
    return df_sc, df.target, df_load.shape[0], df_load

st.header('Выберите датасет')
db = st.radio('',('iris','wine','breast_cancer'))
if db == 'iris':
    df = load_iris()
elif db == 'wine':
    df = load_wine()
elif db == 'breast_cancer':
    df = load_breast_cancer()

st.header('Обучение модели ближайших соседей')

data_load_state = st.text('Загрузка данных...')
data_X, data_Y, data_len, data  = load_data(df)
data_load_state.text('Данные загружены!')

st.write(data.head())

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)))
```

Выберите датасет

- (•) iris
- ( ) wine
- ( ) breast_cancer

Обучение модели ближайших соседей

Данные загружены!

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1000	3.5000	1.4000	0.2000
1	4.9000	3	1.4000	0.2000
2	4.7000	3.2000	1.3000	0.2000
3	4.6000	3.1000	1.5000	0.2000
4	5	3.6000	1.4000	0.2000

Количество фолдов:

5

3 10

Количество строк в наборе данных - 150

Максимальное допустимое количество ближайших соседей с учетом выбранного количества фолдов - 120

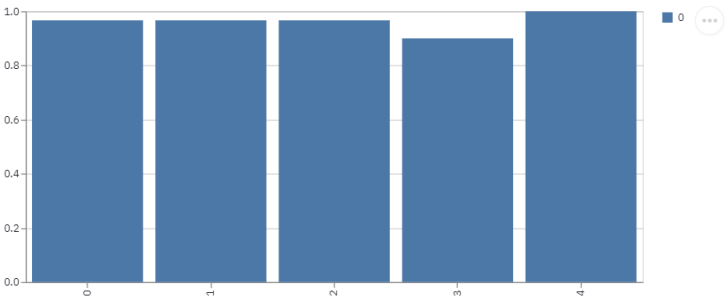
Количество ближайших соседей:

5

1 120

Оценка качества модели

Значения accuracy для отдельных фолдов



Усредненное значение accuracy по всем фолдам - 0.96