Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Московский государственный технический университет имени Н.Э. Баумана» (МГТУ им. Н.Э. Баумана)

Лабораторная работа 6
по курсу «Технологии машинного обучения»

Выполнил
студент группы ИУ5-64
XXX

Москва, 2021

Цель работы

изучение возможностей демонстрации моделей машинного обучения с помощью веб-приложений.

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.

Код

```python
#!/usr/bin/env python
import streamlit as st
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler, StandardScaler, LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error, median_absolute_error, mean_absolute_percentage_error
from sklearn.ensemble import BaggingRegressor, RandomForestRegressor, AdaBoostRegressor, GradientBoostingRegressor

st.header('Датасет')
main_status = st.text('')
read_state = st.text('Чтение датасета...')
data = pd.read_csv("../2/melbourne_housing.csv")
columns_and_types = {
  "Rooms": np.int64,
  "Type": None,
  "Price": np.int64,
  "Distance": np.float64,
  "Postcode": np.int64,
  "Bedroom2": np.int64,
  "Bathroom": np.int64,
  "Car": np.int64,
  "Landsize": np.float64,
  "BuildingArea": np.float64,
  "YearBuilt": np.int64,
  "Lattitude": np.float64,
  "Longtitude": np.float64,
  "Propertycount": np.int64,
}
data = data[list(columns_and_types.keys())]
data.dropna(axis=0, how='any', inplace=True)
data = data.astype({k: v for k,v in columns_and_types.items() if v is not None})
type_encoder = LabelEncoder()
```

```
data["Type"] = type_encoder.fit_transform(data["Type"])
read_state.text('Датасет загружен!')

st.subheader('head:')
st.write(data.head())

test_size = st.sidebar.slider("test_size", 0.1, 0.9, value = 0.3)
n_estimators = st.sidebar.slider("n_estimators", 1, 50, value=5)
random_state = st.sidebar.slider("random_state", 1, 100, value=10)

target_option = st.sidebar.selectbox('Target:', data.columns)
feature_cols = []
st.sidebar.subheader('Features:')
for col in data.columns:
  cb = st.sidebar.checkbox(col, value=True)
  if cb:
    feature_cols.append(col)

scaler = st.sidebar.radio("Мастабирование", ("Нет", "MinMaxScaler", "StandardScaler"))
if scaler == 'MinMaxScaler':
  sc2 = MinMaxScaler()

  for col in data.columns:
    if col != target_option:
      data[col] = sc2.fit_transform(data[[col]])

  st.subheader('Мастабирование:')
  st.write(data.head())
elif scaler == 'StandardScaler':
  sc2 = StandardScaler()

  for col in data.columns:
    if col != target_option:
      data[col] = sc2.fit_transform(data[[col]])

  st.subheader('Мастабирование:')
  st.write(data.head())

main_status.text('В процессе обучения...')
data_X = data.loc[:, [x for x in feature_cols if x != target_option]]
data_Y = data.loc[:, target_option]
data_X_train, data_X_test, data_y_train, data_y_test = train_test_split(
  data_X,
  data_Y,
  test_size=test_size,
  random_state=1,
)

bc = BaggingRegressor(n_estimators=n_estimators, oob_score=True, random_state=random_state)
bc.fit(data_X_train, data_y_train)

rfc = RandomForestRegressor(n_estimators=n_estimators, oob_score=True, random_state=random_state)
rfc.fit(data_X_train, data_y_train)

gbc = GradientBoostingRegressor(random_state=random_state)
gbc.fit(data_X_train, data_y_train)

main_status.text("Обучено!")

metrics = [r2_score, mean_absolute_error, mean_squared_error, median_absolute_error,
mean_absolute_percentage_error]
metr = [i.__name__ for i in metrics]
metrics_ms = st.sidebar.multiselect("Метрики", metr)

methods = [bc, rfc, gbc]
md = [i.__class__.__name__ for i in methods]
methods_ms = st.sidebar.multiselect("Методы обучения", md)

selMethods = []
for i in methods_ms:
  for j in methods:
    if i == j.__class__.__name__:
      selMethods.append(j)

selMetrics = []
for i in metrics_ms:
  for j in metrics:
    if i == j.__name__:
      selMetrics.append(j)

st.header('Оценка')
```

```
115 for name in selMetrics:
116   st.subheader(name.__name__)
117   for func in selMethods:
118     y_pred = func.predict(data_X_test)
119 
120     fig, ax = plt.subplots()
121     ax.plot(data_X_test, data_y_test, 'b.')
122     ax.plot(data_X_test, y_pred, 'r.')
123     st.pyplot(fig)
124 
125     st.text("{} - {}".format(func.__class__.__name__, name(y_pred, data_y_test)))
```

Результат работы