|  |  |  |
| --- | --- | --- |
| Защищено:  Гапанюк Ю.Е.    "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г. |  | Демонстрация:  Гапанюк Ю.Е.  "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г. |

**Отчет по лабораторной работе № 6 по курсу**

**Технологии машинного обучения**

**ГУИМЦ**

#### Тема работы: " Создание веб-приложения для демонстрации моделей машинного обучения "

18

(количество листов)

Вариант № **3**

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| студент группы ИУ5Ц-82Б | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (подпись) |
| Чиварзин А.Е. | "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г. |

Москва, МГТУ - 2022

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**СОДЕРЖАНИЕ**

[1. Описание задания 3](#_Toc104311857)

[2. Текст программы 4](#_Toc104311858)

[3. Результаты выполнения программы 9](#_Toc104311859)

# Описание задания

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.

Для разработки рекомендуется использовать следующие (или аналогичные) фреймворки:

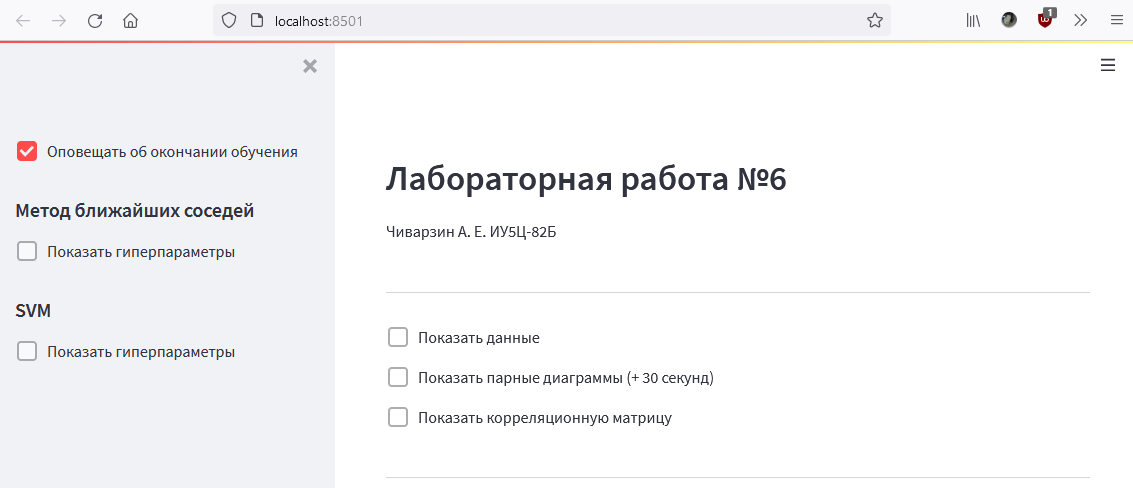
* [streamlit](https://streamlit.io/)
* [gradio](https://www.gradio.app/)
* [dash](https://plotly.com/dash/)

# Текст программы

import os  
import subprocess  
  
import matplotlib.pyplot as plt  
import numpy as np  
import pandas as pd  
import seaborn as sns  
import streamlit as st  
from sklearn.model\_selection import GridSearchCV, train\_test\_split  
from sklearn.neighbors import KNeighborsRegressor  
from sklearn.preprocessing import LabelEncoder  
from sklearn.preprocessing import MinMaxScaler  
from sklearn.svm import SVR  
  
  
def about():  
 st.header('Лабораторная работа №6')  
 st.write('Чиварзин А. Е. ИУ5Ц-82Б')  
 st.markdown('---------------------------------------------------------------------')  
  
  
@st.cache  
def load\_data():  
 *'''  
 Загрузка данных  
 '''* data\_original = pd.read\_csv('data/cwurData.csv', sep=",")  
 return data\_original  
  
  
@st.cache  
def delete\_NULLs(data\_in):  
 data\_out = data\_in.copy()  
 *# Удаление строк, содержащих пустые значения* data\_out = data\_out.dropna(axis=0, how='any')  
 return data\_out  
  
  
@st.cache  
def preprocess\_data(data\_in):  
 *'''  
 Масштабирование и кодирование признаков, функция возвращает X и y для кросс-валидации  
 '''* data\_out = data\_in.copy()  
 le = LabelEncoder()  
 institution\_le = le.fit\_transform(data\_out['institution'])  
 le\_country = LabelEncoder()  
 country\_le = le\_country.fit\_transform(data\_out['country'])  
 data\_digit = data\_out.copy()  
 data\_digit["institution"] = institution\_le  
 data\_digit['country'] = country\_le  
  
 sc1 = MinMaxScaler()  
 sc1\_data = sc1.fit\_transform(data\_digit[['broad\_impact']])  
 sc2 = MinMaxScaler()  
 sc2\_data = sc2.fit\_transform(data\_digit[['institution']])  
 sc3 = MinMaxScaler()  
 sc3\_data = sc3.fit\_transform(data\_digit[['country']])  
 sc4 = MinMaxScaler()  
 sc4\_data = sc4.fit\_transform(data\_digit[['national\_rank']])  
 sc5 = MinMaxScaler()  
 sc5\_data = sc5.fit\_transform(data\_digit[['quality\_of\_education']])  
 sc6 = MinMaxScaler()  
 sc6\_data = sc6.fit\_transform(data\_digit[['alumni\_employment']])  
 sc7 = MinMaxScaler()  
 sc7\_data = sc7.fit\_transform(data\_digit[['quality\_of\_faculty']])  
 sc8 = MinMaxScaler()  
 sc8\_data = sc8.fit\_transform(data\_digit[['publications']])  
 sc9 = MinMaxScaler()  
 sc9\_data = sc9.fit\_transform(data\_digit[['influence']])  
 sc10 = MinMaxScaler()  
 sc10\_data = sc10.fit\_transform(data\_digit[['citations']])  
 sc11 = MinMaxScaler()  
 sc11\_data = sc11.fit\_transform(data\_digit[['broad\_impact']])  
 sc12 = MinMaxScaler()  
 sc12\_data = sc12.fit\_transform(data\_digit[['patents']])  
 sc13 = MinMaxScaler()  
 sc13\_data = sc13.fit\_transform(data\_digit[['score']])  
 sc14 = MinMaxScaler()  
 sc14\_data = sc14.fit\_transform(data\_digit[['year']])  
  
 data\_normal = data\_digit.copy()  
 data\_normal['world\_rank'] = sc1\_data  
 data\_normal['institution'] = sc2\_data  
 data\_normal['country'] = sc3\_data  
 data\_normal['national\_rank'] = sc4\_data  
 data\_normal['quality\_of\_education'] = sc5\_data  
 data\_normal['alumni\_employment'] = sc6\_data  
 data\_normal['quality\_of\_faculty'] = sc7\_data  
 data\_normal['publications'] = sc8\_data  
 data\_normal['influence'] = sc9\_data  
 data\_normal['citations'] = sc10\_data  
 data\_normal['broad\_impact'] = sc11\_data  
 data\_normal['patents'] = sc12\_data  
 data\_normal['score'] = sc13\_data  
 data\_normal['year'] = sc14\_data  
 data\_out = data\_normal  
 return pd.DataFrame(data\_out).drop(['world\_rank'], axis=1), data\_out['world\_rank']  
  
  
*#######################################################################################################################*def KNN(x, y):  
 reg\_gs = GridSearchCV(KNeighborsRegressor(), tuned\_parameters, cv=cv\_slider, scoring='neg\_median\_absolute\_error')  
 reg\_gs.fit(x, y)  
  
 st.subheader('Оценка качества модели')  
  
 st.write('Лучшее значение параметров - {}'.format(reg\_gs.best\_params\_))  
  
 *# Изменение качества на тестовой выборке в зависимости от К-соседей* fig1 = plt.figure(figsize=(7, 5))  
 ax = plt.plot(n\_range, reg\_gs.cv\_results\_['mean\_test\_score'])  
 plt.xlabel('Количество соседей')  
 st.pyplot(fig1)  
  
  
def SVM(X: pd.DataFrame, Y):  
 if X.shape[1] == 0:  
 st.write('Ни один столбец не выбран')  
 else:  
 gp = st.selectbox('Гиперпараметр для построения', X.columns)  
 X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, random\_state=2022, test\_size=0.1)  
 svr = SVR(kernel='linear')  
 svr.fit(X\_train, Y\_train)  
 pred\_y = svr.predict(X\_test)  
 fig2 = plt.figure(figsize=(7, 5))  
 plt.scatter(X\_test[gp], Y\_test, marker='s', label='Тестовая выборка')  
 plt.scatter(X\_test[gp], pred\_y, marker='o', label='Предсказанные данные')  
 plt.legend(loc='lower right')  
 plt.xlabel('рейтинг за широкое влияние')  
 plt.ylabel('Целевой признак')  
 st.pyplot(fig2)  
  
  
about()  
use\_msg = True  
if os.name == 'nt':  
 if st.sidebar.checkbox('Оповещать об окончании обучения', value=True):  
 use\_msg = True  
 else:  
 use\_msg = False  
  
data = load\_data()  
data\_no\_null = delete\_NULLs(data)  
  
if st.checkbox('Показать данные'):  
 st.write(data\_no\_null)  
  
if st.checkbox('Показать парные диаграммы (+ 30 секунд)'):  
 st.pyplot(sns.pairplot(data\_no\_null, height=5))  
 st.write('Разглядеть не получится. Известный баг: https://github.com/streamlit/streamlit/issues/796')  
  
if st.checkbox('Показать корреляционную матрицу'):  
 fig1, ax = plt.subplots(figsize=(10, 5))  
 sns.heatmap(data\_no\_null.corr(), annot=True, fmt='.2f')  
 st.pyplot(fig1)  
  
st.markdown('--------------------------------------------------------------------------------------------------')  
  
st.sidebar.header('Метод ближайших соседей') *#########################################################################*do\_analize\_KNN = False *# По-умолчанию отключим анализ данных, чтобы не ждать 15 секунд - 5 ммнут!!!*if st.sidebar.checkbox('Показать гиперпараметры'):  
 cv\_slider = st.sidebar.slider('Количество фолдов:', min\_value=2, max\_value=10, value=3, step=1)  
 step\_slider = st.sidebar.slider('Шаг для соседей:', min\_value=1, max\_value=50, value=10, step=1)  
 *# Количество записей* data\_len = data.shape[0]  
 data\_no\_null\_len = data\_no\_null.shape[0]  
 *# Вычислим количество возможных ближайших соседей* rows\_in\_one\_fold = int(data\_no\_null\_len / cv\_slider)  
 allowed\_knn = int(rows\_in\_one\_fold \* (cv\_slider - 1))  
 st.write('Количество строк в наборе данных до очистки строк - {}'.format(data\_len))  
 st.write('Количество строк в наборе данных после очистки строк - {}'.format(data\_no\_null\_len))  
 st.write('Максимальное допустимое количество ближайших соседей с учётом выбранного количества фолдов - {}'.format(allowed\_knn))  
 *# Подбор гиперпараметра* n\_range\_list = list(range(1, allowed\_knn, step\_slider))  
 n\_range = np.array(n\_range\_list)  
 st.write('Возможные значения соседей - {}'.format(n\_range))  
 tuned\_parameters = [{'n\_neighbors': n\_range}]  
  
 if st.sidebar.checkbox('Выполнять обучение'):  
 do\_analize\_KNN = True  
 else:  
 do\_analize\_KNN = False  
 st.sidebar.write(  
 '⚠ Устанавливайте этот флажок только после окончательной установки гиперпараметров\n'  
 'После установки флажка рекоммендуется идти пить чай на 30 секунд - 30 минут...')  
  
st.sidebar.header('SVM') *#############################################################################################*do\_analize\_SVM = False  
show\_SVM\_params = False  
if st.sidebar.checkbox('Показать гиперпараметры '):  
 show\_SVM\_params = True  
 if st.sidebar.checkbox('Выполнять обучение'):  
 do\_analize\_SVM = True  
 else:  
 do\_analize\_SVM = False  
else:  
 show\_SVM\_params = False  
  
data\_X, data\_y = preprocess\_data(data\_no\_null)  
  
if do\_analize\_KNN:  
 KNN(data\_X, data\_y)  
  
if show\_SVM\_params:  
 columns = st.multiselect('Столбцы для построения модели', data\_X.columns)  
 if do\_analize\_SVM:  
 SVM(data\_X[columns], data\_y)  
  
*# Выводим пользователю сообщение об окончании работы скрипта (на редакции Enterprise multi-session оно полноэкранное)  
# Работает только на Windows*if do\_analize\_KNN or do\_analize\_SVM:  
 if os.name == 'nt' and use\_msg:  
 cmd\_output = subprocess.check\_output(['cmd', '/c chcp 65001 & msg \* Модель успешно построена!'])  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 pass

# Результаты выполнения программы

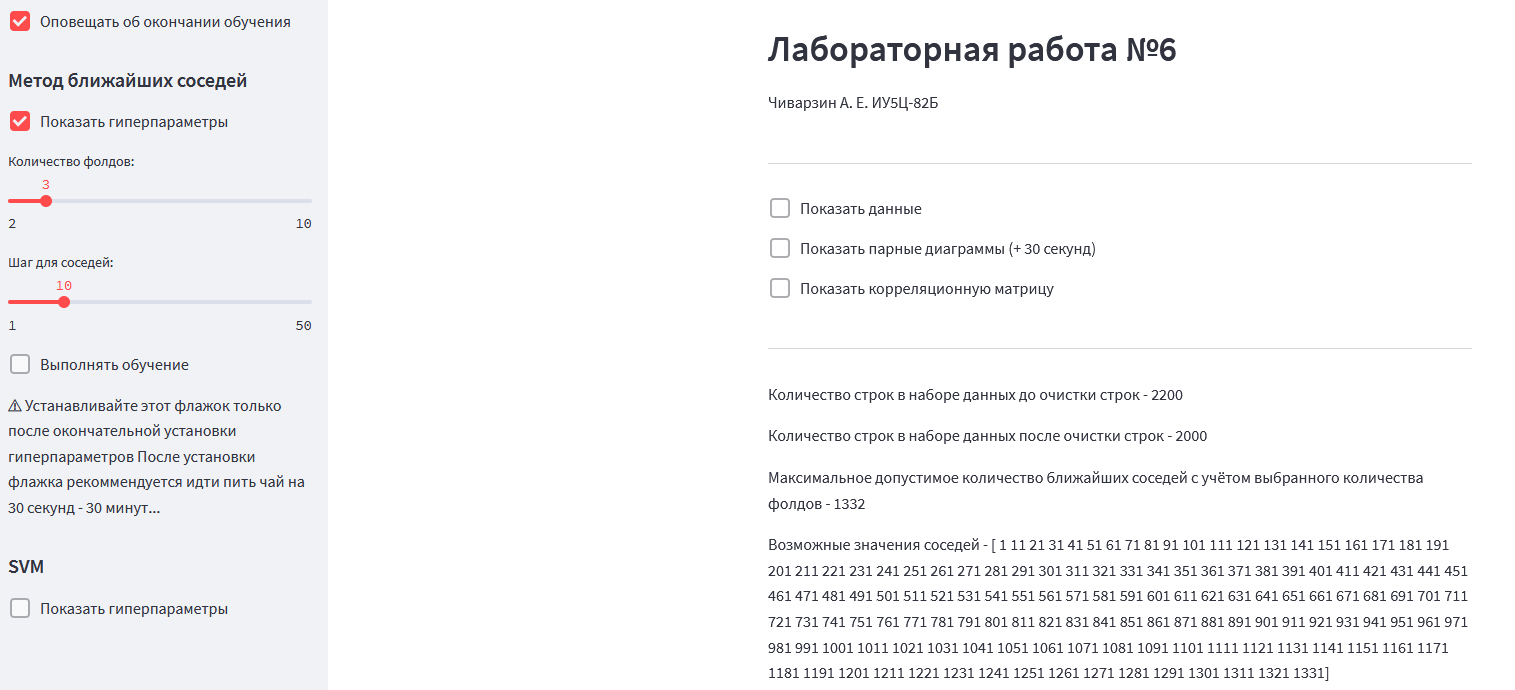
**Вид приложения после запуска**



Левая панель содержит выбор модели машинного обучения. На правой панели можно вывести исходные данные, парные диаграммы и показать корреляционную матрицу.

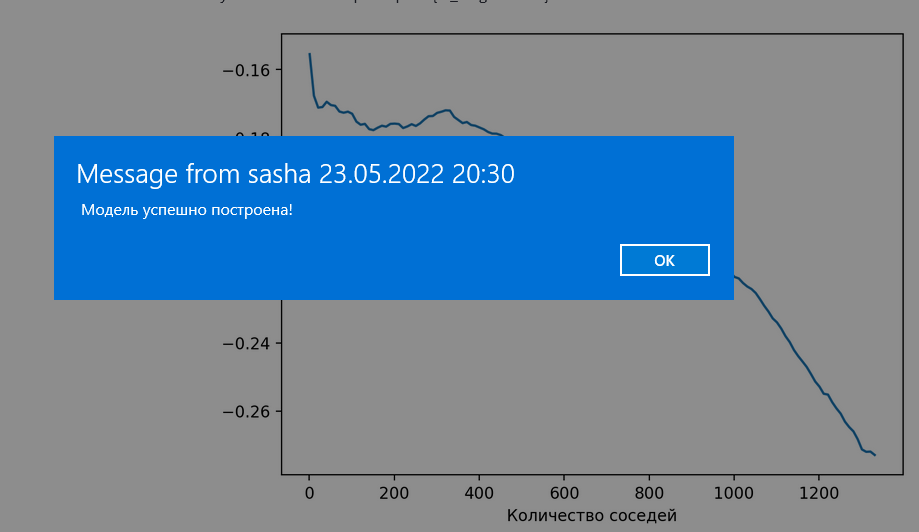
На левой панели можно включить или отключить показ сообщения об успешном завершении обучения, что полезно при

**Ближайшие соседи**



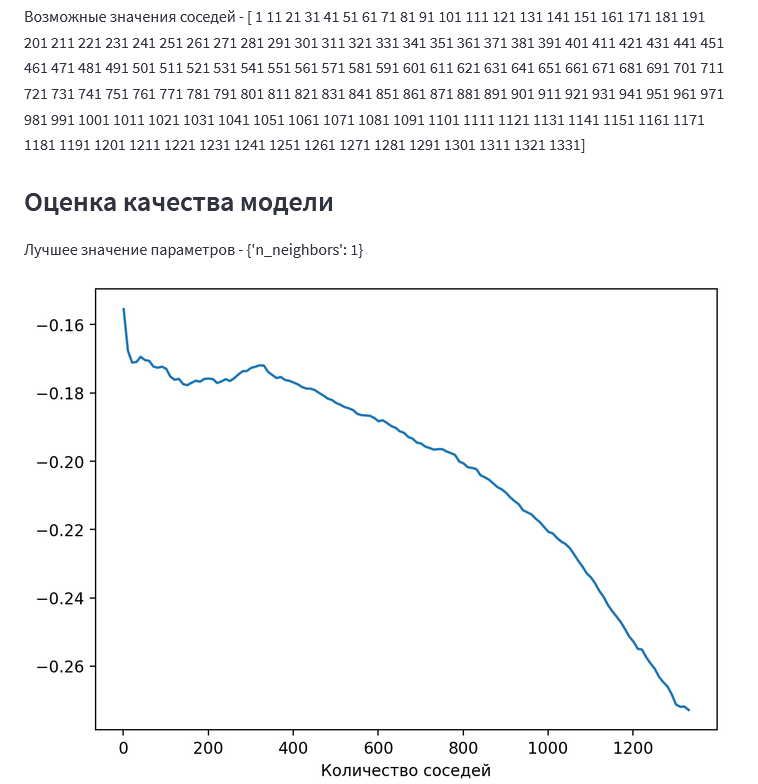
Поле включения флажка «Показать гиперпараметры» на левой панели появляется выбор гиперпараметров для KNN, а на правой панели выводятся информация, такая как список из количества соседей, ….

**Обучение KNN**

****

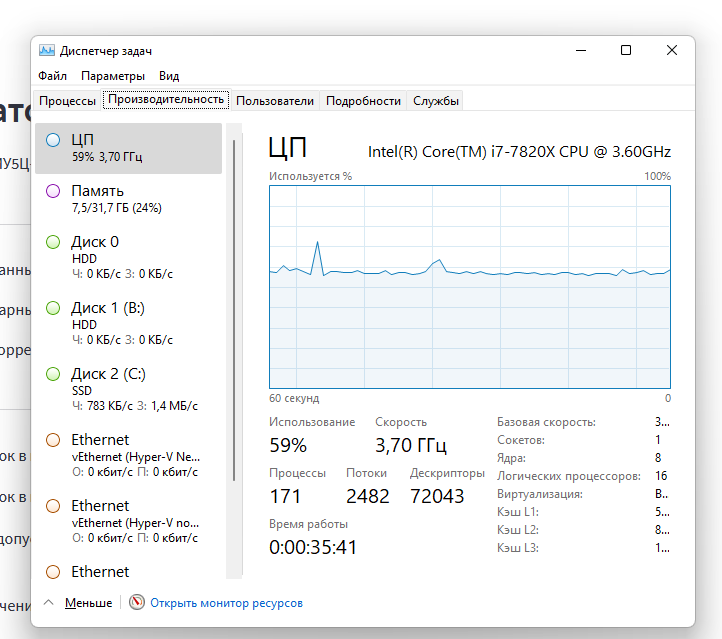
После установки флажка «Выполнять обучение» было выполнено обучение модели, которое длилось 21 секунду, после чего было получено полноэкранное сообщение. (Если на ОС не активна роль RDS, то будем маленькое окно с тем же текстом)

На следующей странице отчёта представлен полученный график (без перекрытия).

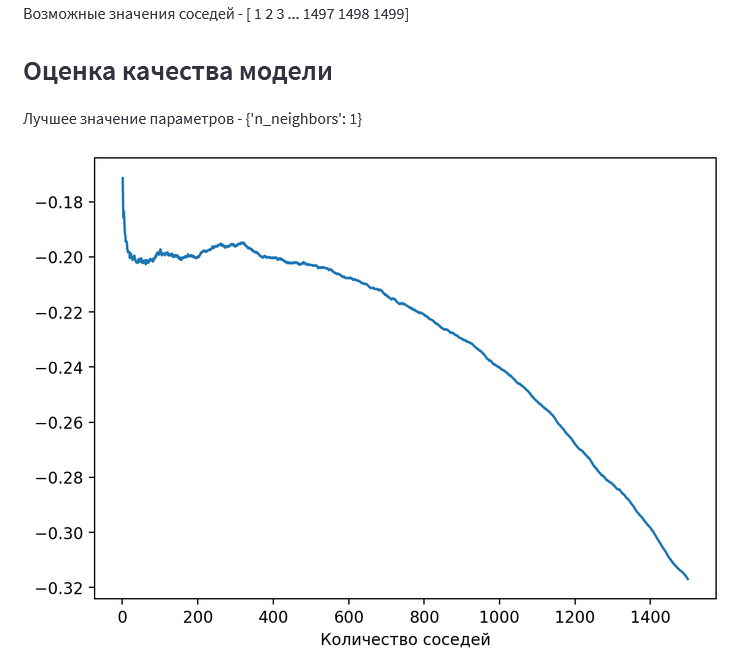


Если выставить шаг соседей = 1 и количество фолдов = 4, то придётся ждать 4 минуты 30 секунд.

При этом в диспетчере задач ЦП будет загружен на 59% при этом на машине всего один активный сеанс (console), отключённых сеансов тоже нет.



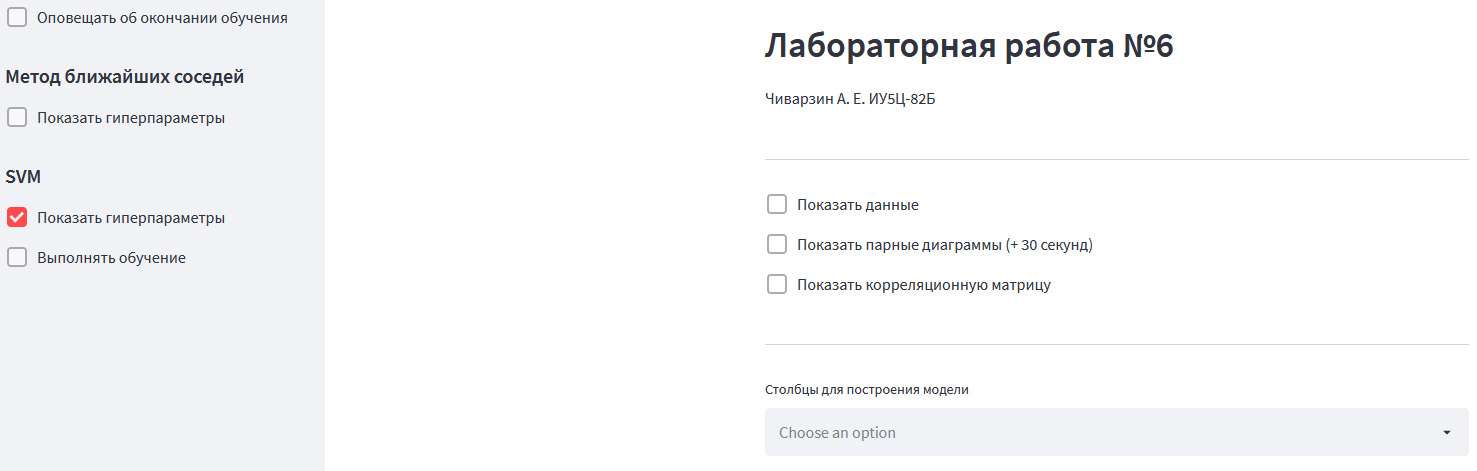
Ниже представлен получившийся график



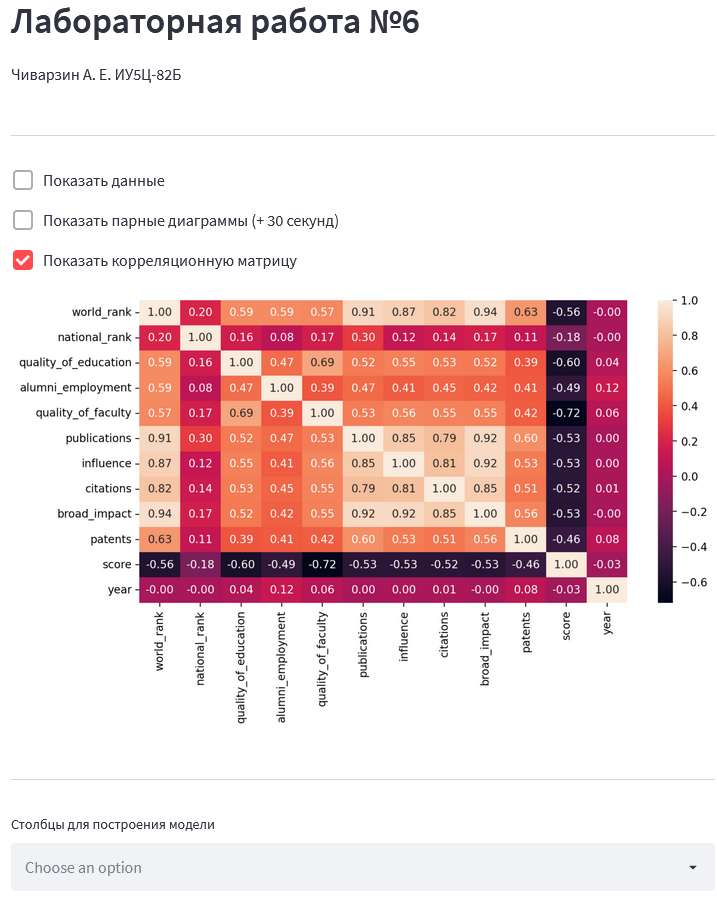
**SVM**

Были сняты все флажки в категории «Метод ближайших соседей». Оповещение об окончании обучения также отключено из-за отсутствия длительных ожиданий.

После установки флажка «Показать гиперпараметры» в SVM приложение предложет выбрать столбцы для построения модели:



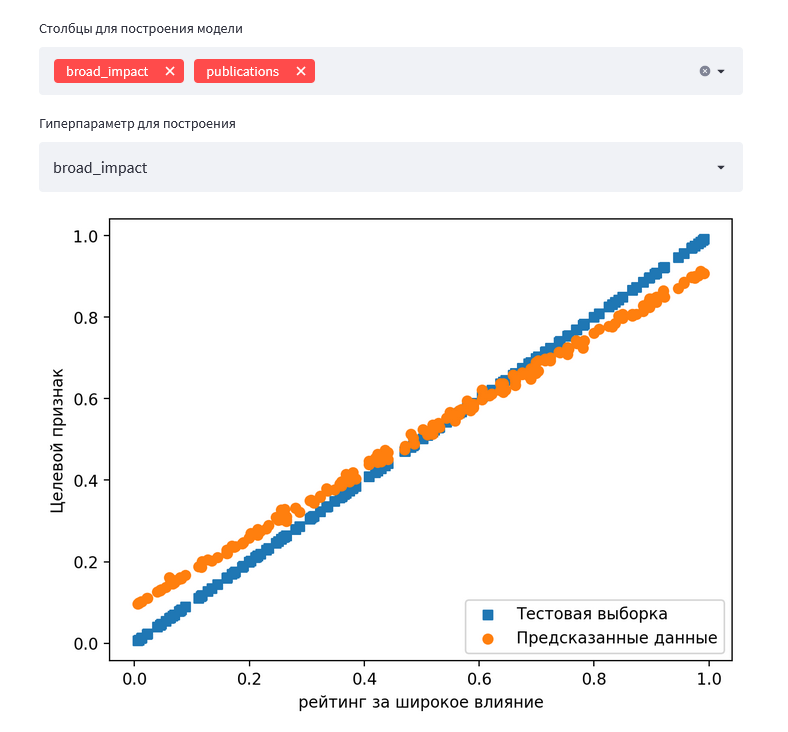
Для того, чтобы сделать более логичный выбор, в приложении имеется возможность отобразить корреляционную матрицу. Для этого установим флажок «Показать корреляционную матрицу».



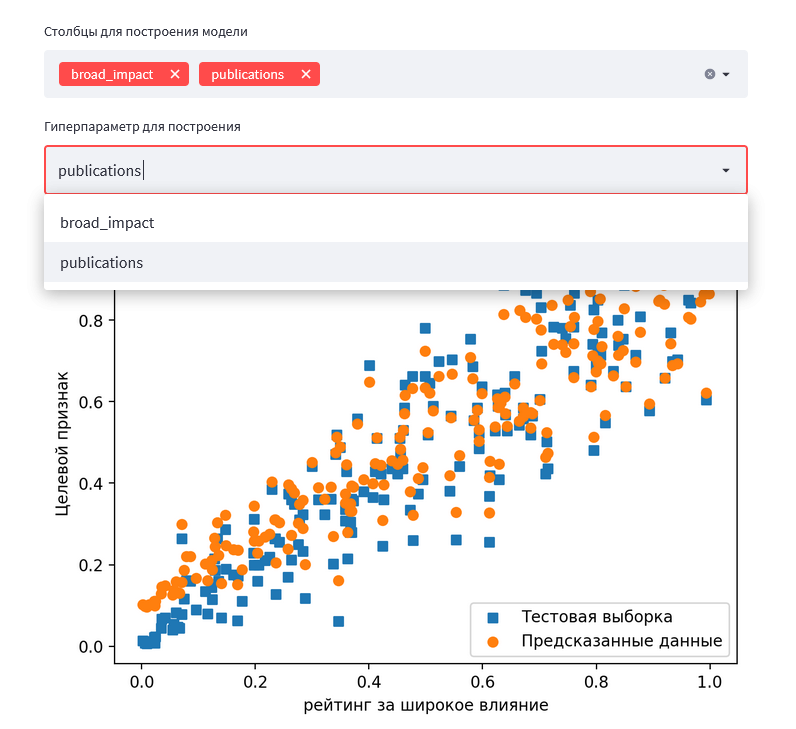
Как видно из матрицы, более всего с целевым признаком «world\_rank» коррелируют broad\_impact и publications. Отключим вывод матрицы и построим модель по этим признакам.



После установки параметров отметим флажок «Выполнять обучение».

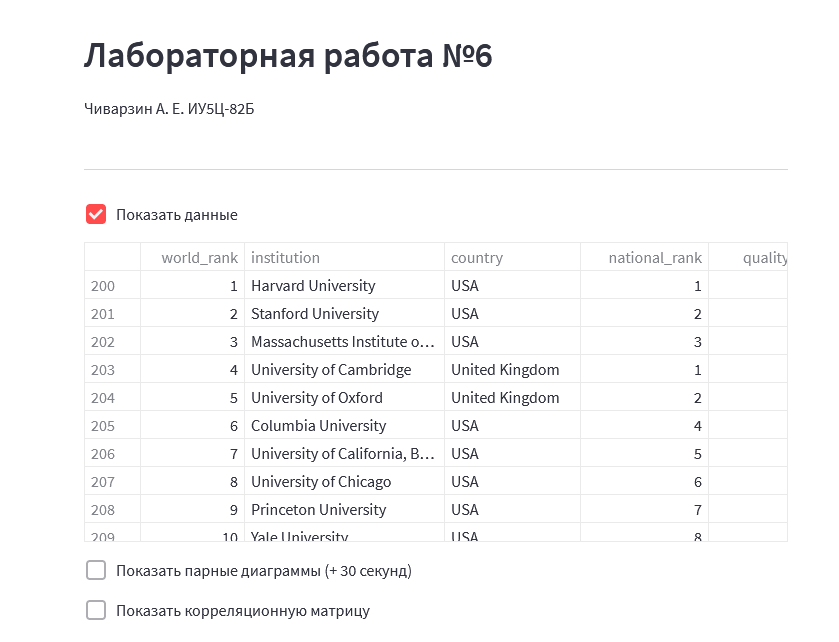


Имеется возможность изменить гиперпараметр построения

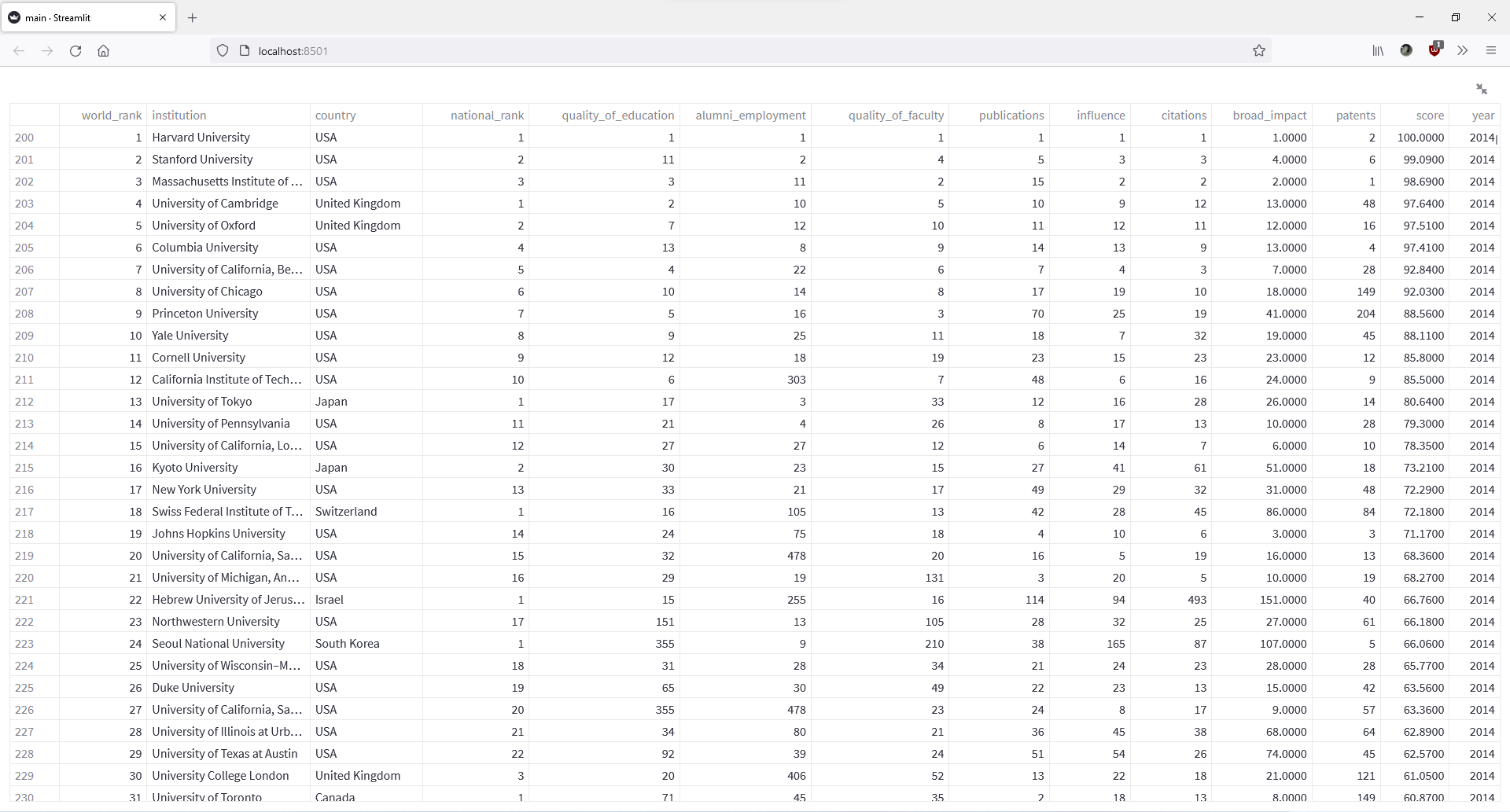


**Вывод исходных данных**

Были сняты все флажки и установлен флажок «Показать данные»



Поддерживается возможность открыть таблицу почти на всё окно браузера

.

**Вывод парных диаграмм**

Были сняты все флажки и установлен флажок «Показать парные лмаграммы»

