|  |  |
| --- | --- |
|  | **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 \_\_\_\_\_\_\_\_\_\_\_\_\_\_ ИУ (Информатика и системы управления)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ИУ5 (Системы обработки информации и управления)\_\_\_\_\_\_\_

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОЙ РАБОТЕ***

***НА ТЕМУ:***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***Создание веб-приложения для анализа и \_\_\_\_\_\_***

***\_\_\_\_\_визуализации данных с использованием\_\_\_\_\_\_ \_\_\_\_\_\_\_\_методов машинного обучения \_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

Студентка ИУ5-61Б **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** Андреев А.В.

(Группа) (Подпись, дата) (Фамилия И.О.)

Руководитель курсового проекта **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  Гапанюк Ю.Е.

(Подпись, дата) (Фамилия И.О.)

*Москва, 2021 г.*

**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)**

УТВЕРЖДАЮ

Заведующий кафедрой \_\_\_\_\_\_\_\_\_\_

(Индекс)

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

(И.О.Фамилия)

« \_\_\_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 20 \_\_\_\_ г.

**ЗАДАНИЕ**

**на выполнение курсовой работы**

по дисциплине \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студентка группы ИУ5-61Б Андреев Александр Владимирович

(Фамилия, имя, отчество)

Тема курсового проекта Создание веб-приложения для анализа и визуализации данных

\_\_\_\_\_с использованием методов машинного обучения \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

Направленность КР (учебная, исследовательская, практическая, производственная, др.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_учебная\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Источник тематики (кафедра, предприятие, НИР) \_\_\_\_\_\_\_\_\_\_\_\_\_\_кафедра\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

График выполнения работы: 25% к \_\_\_ нед., 50% к \_\_\_ нед., 75% к \_\_ нед., 100% к \_\_\_ нед.

***Задание*** \_\_\_\_\_\_провести предобработку выбранного набора данных, обучить модели машинного обучения, оценить метрики качества для каждой модели, подобрать соответствующие гиперпараметры, выбрать лучшие алгоритмы для данного датасета; обернуть всё в интерактивное веб-приложение \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

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

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

***Оформление курсовой работы:***

Расчетно-пояснительная записка на \_\_\_\_\_ листах формата А4.

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

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

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

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

Дата выдачи задания « \_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.

**Руководитель курсовой работы**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

**Содержание**

[Введение 4](#_Toc73563181)

[Задание 4](#_Toc73563182)

[Средства реализации 5](#_Toc73563183)

[Листинг 5](#_Toc73563185)

[Выводы 16](#_Toc73563186)

[Список источников информации 16](#_Toc73563187)

# Введение

В рамках курса «Технологии машинного обучения» текущего семестра мы постепенно наращивали свои знания и проходили все стадии разработки продукта, связанного с анализом данных: разведочный анализ данных, предобработку признаков, обучение моделей, применений метрик качества. Были рассмотрены различные алгоритмы машинного обучения, в том числе и ансамблевые модели. Шаг за шагом мы собирали наработки в рамках лабораторных работ, и вот пришло время аккумулировать всё сделанное за семестр в курсовой работе.

# Задание

На выбранном наборе данных:

* произвести разведочный анализ данных, визуализировать признаки (все или некоторые) таким образом, чтобы прояснить структуру данных;
* разделить признаки на независимые фичи и целевую переменную;
* провести всю необходимую предобработку данных (масштабирование числовых признаков, заполнение пропущенных значений, кодирование категориальных признаков);
* провести корреляционный анализ (корреляционная матрица и/или тепловая карта), сделать соответствующие выводы о независимых признаках, сильно коррелирующих между собой или с целевой переменной;
* выбрать на основании предыдущих пунктов те признаки, которые войдут в модели;
* разделить набор данных на обучающую и тестовую выборки;
* определиться с моделями машинного обучения и метриками оценки качества, подходящими для конкретной задачи (классификации или регрессии);
* обучить моделей на данных тестовой выборки без подбора гиперпараметров;
* обучить моделей на всех данных с использованием стратегий кросс-валидации с подбором гиперпараметров и выявлением оптимальных значений;
* сравнить значений метрик качества для двух последних пунктов;
* в веб-приложении сделать возможным задавать гиперпараметры для каждого алгоритма и наблюдать за изменением метрик (в том числе в виде графиков);
* сравнить полученные вручную результаты с pipeline’ом библиотеки AutoML;
* сформулировать выводы о качестве обученных моделей.

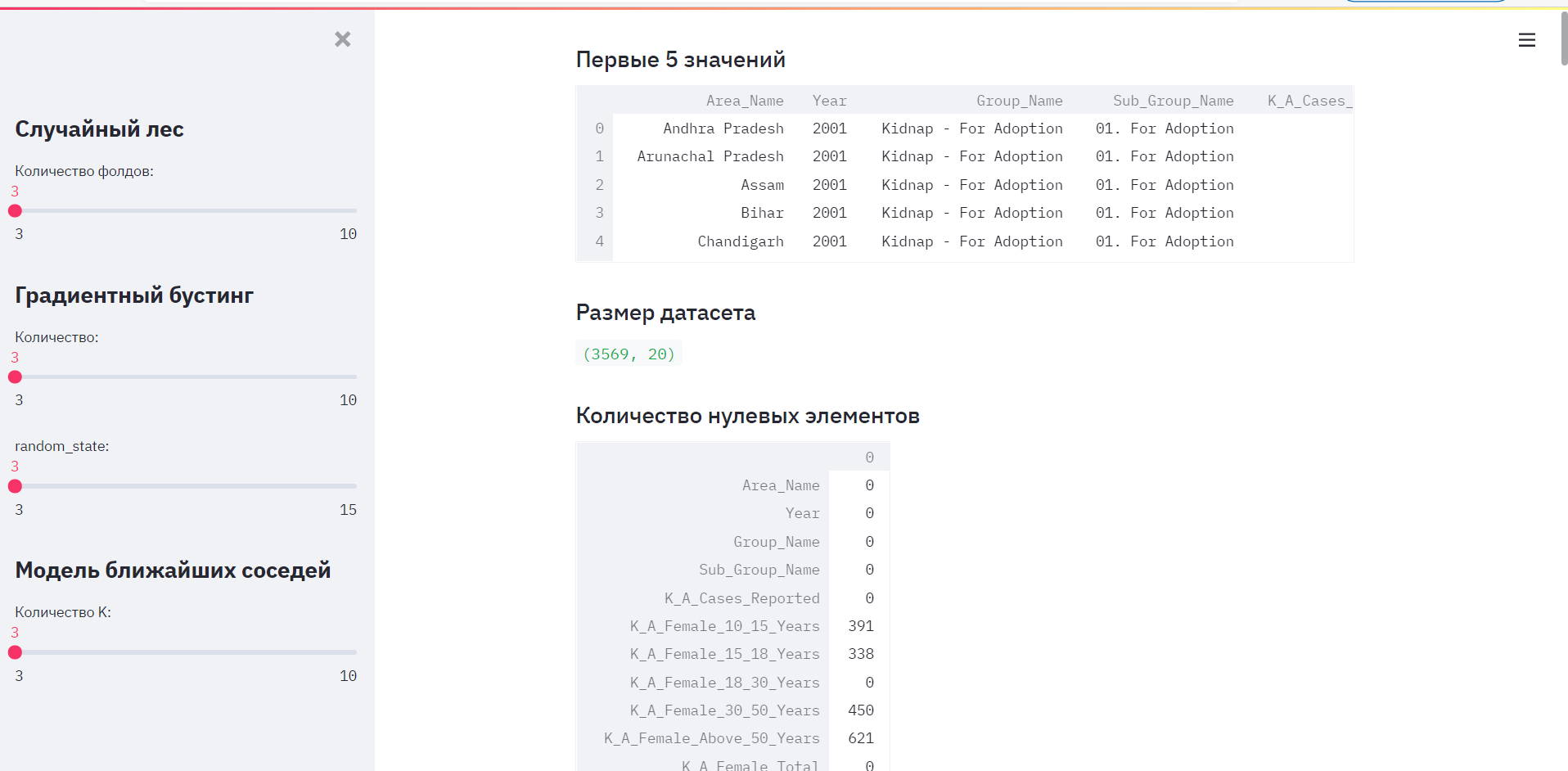
# Средства реализации

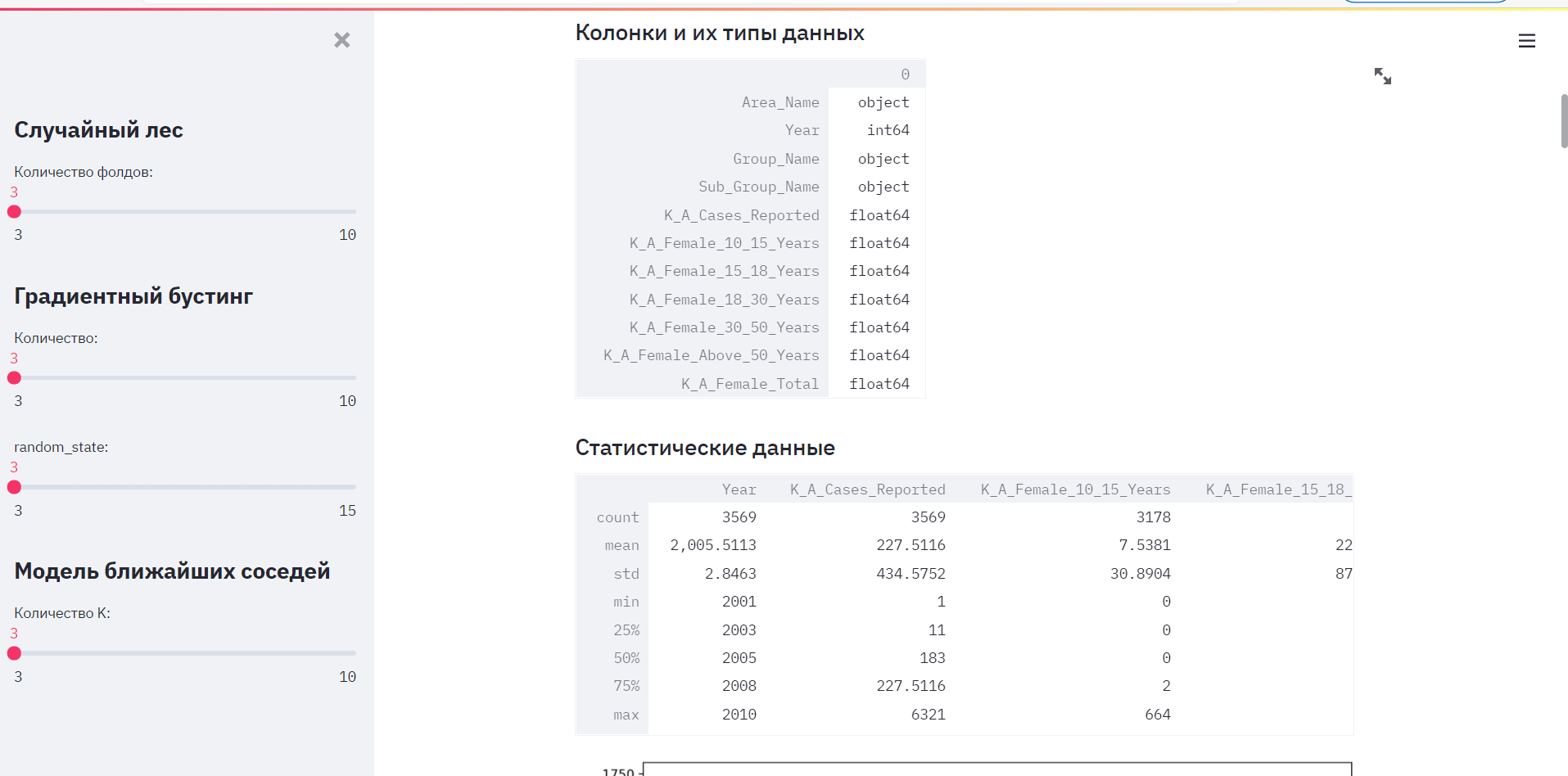
Приложение реализовано на языке программирования Python с использованием веб-фреймворка для задач машинного обучения Streamlit, а также библиотек для работы с данными Pandas, Numpy, Scipy, sklearn.

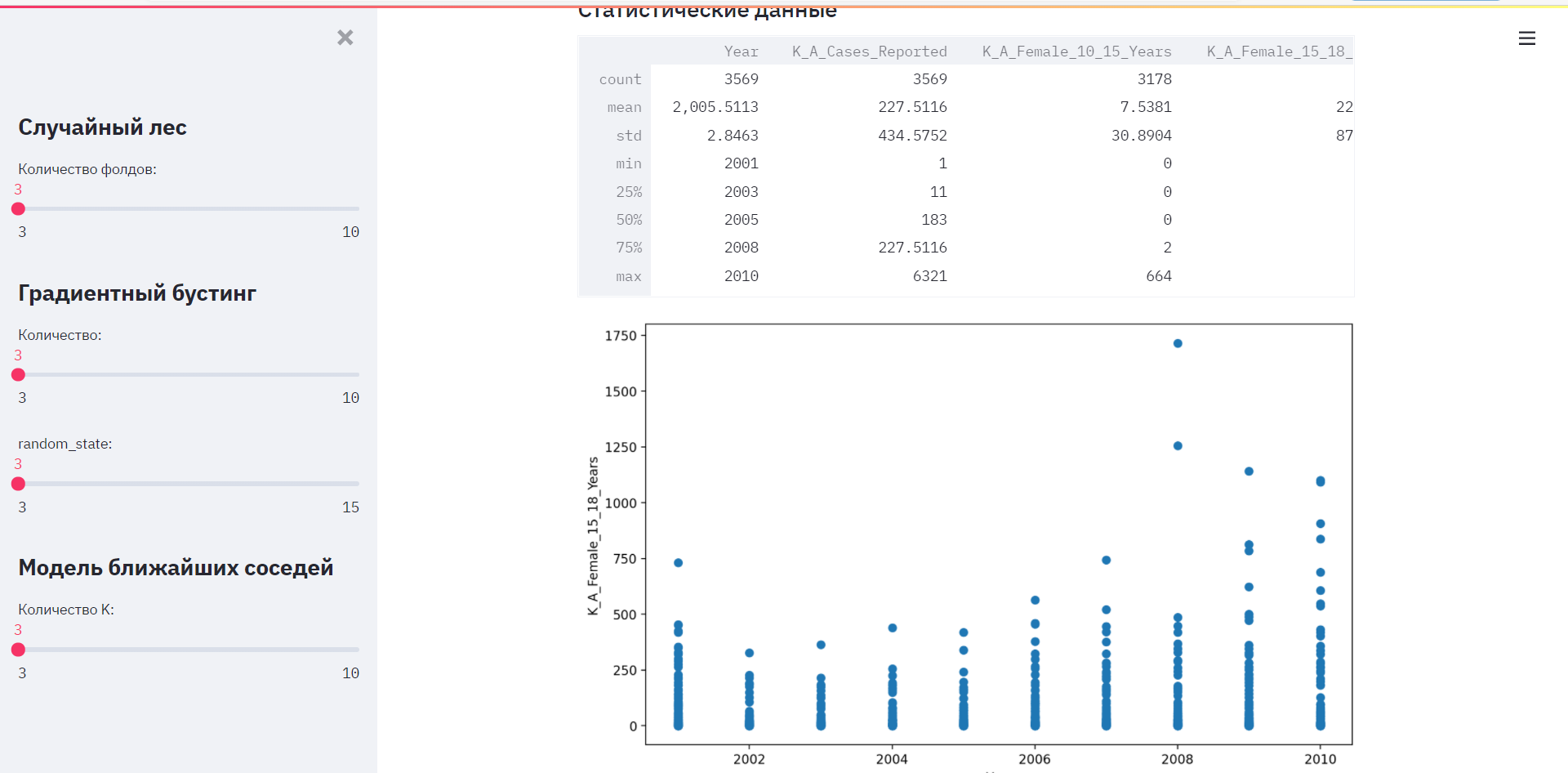
# Листинг

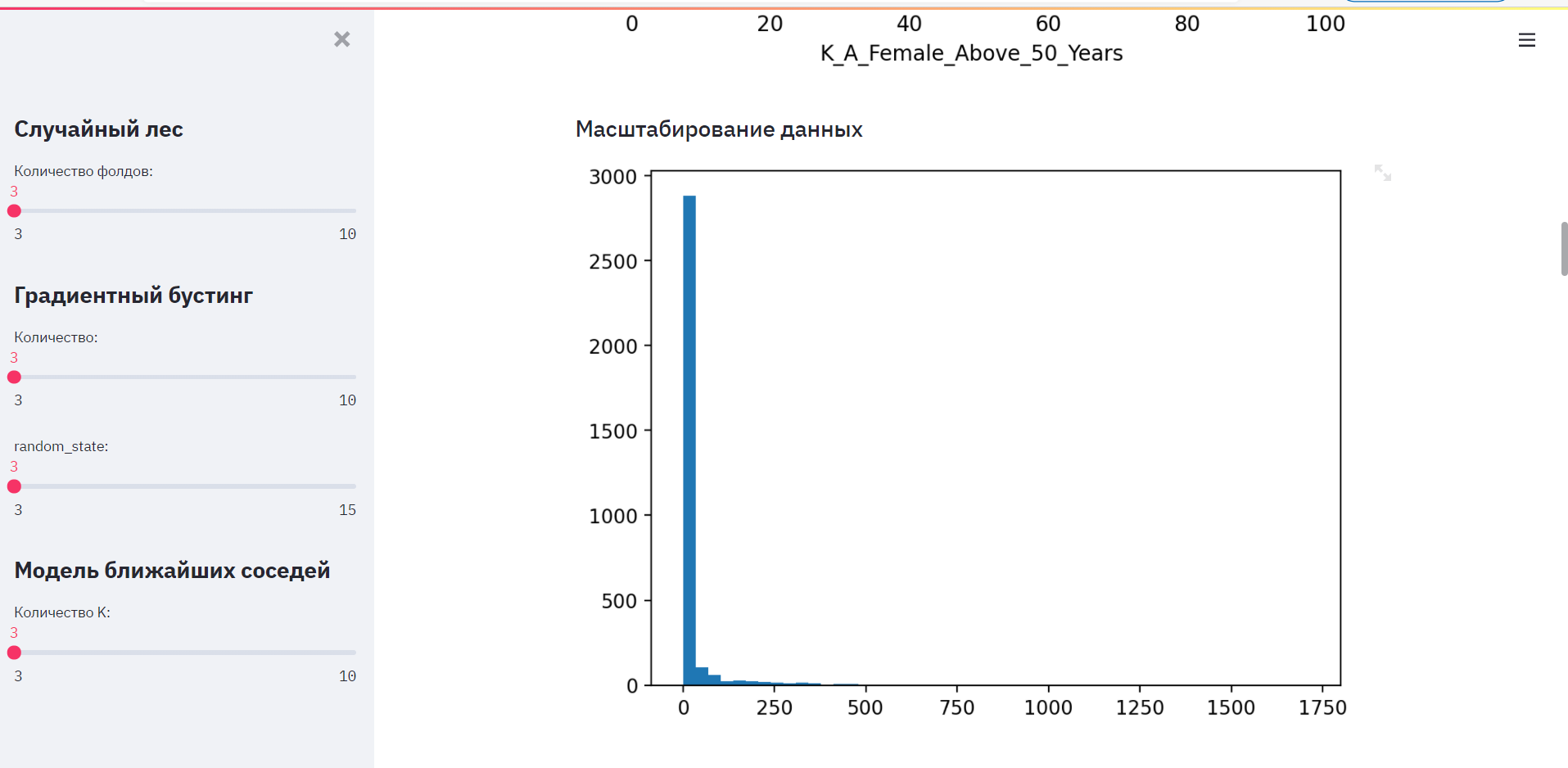
import streamlit as st  
import seaborn as sns  
import pandas as pd  
import numpy as np  
import plotly.figure\_factory as ff  
from sklearn.preprocessing import MinMaxScaler  
from sklearn.model\_selection import cross\_val\_score  
from sklearn.model\_selection import GridSearchCV  
from sklearn.neighbors import KNeighborsRegressor, KNeighborsClassifier  
from sklearn.ensemble import RandomForestRegressor  
import matplotlib.pyplot as plt  
from sklearn.model\_selection import train\_test\_split  
from sklearn.svm import SVC, LinearSVC  
from sklearn.ensemble import GradientBoostingRegressor  
from sklearn.preprocessing import StandardScaler, MinMaxScaler, StandardScaler, Normalizer  
from sklearn.linear\_model import LinearRegression  
from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, median\_absolute\_error, r2\_score  
from sklearn.neighbors import KNeighborsRegressor  
from sklearn import tree  
import re  
  
def load\_data():  
 *'''  
 Загрузка данных  
 '''* data = pd.read\_csv('data/39\_Specific\_purpose\_of\_kidnapping\_and\_abduction.csv')  
 return data  
  
  
@st.cache  
def preprocess\_data(data\_in):  
 *'''  
 Масштабирование признаков, функция возвращает X и y для кросс-валидации  
 '''* data\_out = data\_in.copy()  
 # Числовые колонки для масштабирования  
 scale\_cols = ['K\_A\_Female\_18\_30\_Years', 'K\_A\_Female\_Total', 'K\_A\_Male\_Total']  
 new\_cols = []  
 sc1 = MinMaxScaler()  
 sc1\_data = sc1.fit\_transform(data\_out[scale\_cols])  
 for i in range(len(scale\_cols)):  
 col = scale\_cols[i]  
 new\_col\_name = col + '\_scaled'  
 new\_cols.append(new\_col\_name)  
 data\_out[new\_col\_name] = sc1\_data[:, i]  
 X = data\_out[new\_cols]  
 Y = data\_out['K\_A\_Cases\_Reported']  
 # Чтобы в тесте получилось низкое качество используем только 0,5% данных для обучения  
 X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, Y, train\_size=0.8, test\_size=0.2, random\_state=1)  
 return X\_train, X\_test, y\_train, y\_test, X, Y  
  
  
data = load\_data()  
  
data['K\_A\_Female\_18\_30\_Years'] = data['K\_A\_Female\_18\_30\_Years'].replace(0,np.nan)  
data['K\_A\_Female\_18\_30\_Years'] = data['K\_A\_Female\_18\_30\_Years'].fillna(data['K\_A\_Female\_18\_30\_Years'].mean())  
  
data['K\_A\_Female\_Total'] = data['K\_A\_Female\_Total'].replace(0,np.nan)  
data['K\_A\_Female\_Total'] = data['K\_A\_Female\_Total'].fillna(data['K\_A\_Female\_Total'].mean())  
  
data['K\_A\_Male\_Total'] = data['K\_A\_Male\_Total'].replace(0,np.nan)  
data['K\_A\_Male\_Total'] = data['K\_A\_Male\_Total'].fillna(data['K\_A\_Male\_Total'].mean())  
  
data['K\_A\_Cases\_Reported'] = data['K\_A\_Cases\_Reported'].replace(0,np.nan)  
data['K\_A\_Cases\_Reported'] = data['K\_A\_Cases\_Reported'].fillna(data['K\_A\_Cases\_Reported'].mean())  
  
st.sidebar.header('Случайный лес')  
n\_estimators\_1 = st.sidebar.slider('Количество фолдов:', min\_value=3, max\_value=10, value=3, step=1)  
  
st.sidebar.header('Градиентный бустинг')  
n\_estimators\_2 = st.sidebar.slider('Количество:', min\_value=3, max\_value=10, value=3, step=1)  
random\_state\_2 = st.sidebar.slider('random\_state:', min\_value=3, max\_value=15, value=3, step=1)  
  
st.sidebar.header('Модель ближайших соседей')  
n\_estimators\_3 = st.sidebar.slider('Количество K:', min\_value=3, max\_value=10, value=3, step=1)  
  
# Первые пять строк датасета  
st.subheader('Первые 5 значений')  
st.write(data.head())  
  
st.subheader('Размер датасета')  
st.write(data.shape)  
  
st.subheader('Количество нулевых элементов')  
st.write(data.isnull().sum())  
  
st.write(data['Group\_Name'].value\_counts())  
  
st.subheader('Колонки и их типы данных')  
st.write(data.dtypes)  
  
st.subheader('Статистические данные')  
st.write(data.describe())  
  
fig, ax = plt.subplots(figsize=(10, 6))  
ax.scatter(x=data['Year'], y=data['K\_A\_Female\_15\_18\_Years'])  
plt.xlabel("Year")  
plt.ylabel("K\_A\_Female\_15\_18\_Years")  
st.pyplot(fig)  
  
f1, ax = plt.subplots()  
sns.boxplot(x=data['K\_A\_Female\_Above\_50\_Years'])  
st.pyplot(f1)  
  
st.subheader('Масштабирование данных')  
f, ax = plt.subplots()  
plt.hist(data['K\_A\_Female\_15\_18\_Years'], 50)  
plt.show()  
st.pyplot(f)  
  
st.subheader('Показать корреляционную матрицу')  
fig1, ax = plt.subplots(figsize=(10, 5))  
sns.heatmap(data.corr(), annot=True, fmt='.2f')  
st.pyplot(fig1)  
  
X\_train, X\_test, Y\_train, Y\_test, X, Y = preprocess\_data(data)  
forest\_1 = RandomForestRegressor(n\_estimators=n\_estimators\_1, oob\_score=True, random\_state=10)  
forest\_1.fit(X, Y)  
Y\_predict = forest\_1.predict(X\_test)  
  
  
st.subheader('RandomForestRegressor')  
st.subheader('Средняя абсолютная ошибка:')  
st.write(mean\_absolute\_error(Y\_test, Y\_predict))  
st.subheader('Средняя квадратичная ошибка:')  
st.write(mean\_squared\_error(Y\_test, Y\_predict))  
st.subheader('Median absolute error:')  
st.write(median\_absolute\_error(Y\_test, Y\_predict))  
st.subheader('Коэффициент детерминации:')  
st.write(r2\_score(Y\_test, Y\_predict))  
  
fig1 = plt.figure(figsize=(7, 5))  
ax = plt.scatter(X\_test['K\_A\_Female\_Total\_scaled'], Y\_test, marker='o', label='Тестовая выборка')  
plt.scatter(X\_test['K\_A\_Female\_Total\_scaled'], Y\_predict, marker='.', label='Предсказанные данные')  
plt.legend(loc='lower right')  
plt.xlabel('K\_A\_Female\_Total\_scaled')  
plt.ylabel('strength')  
plt.plot(n\_estimators\_1)  
st.pyplot(fig1)  
  
st.subheader('Нахождение лучшего случайного леса')  
  
params2 = {  
 'n\_estimators': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 25, 50, 75, 100],  
}  
  
grid\_2 = GridSearchCV(estimator=RandomForestRegressor(oob\_score=True, random\_state=10),  
 param\_grid=params2,  
 scoring='neg\_mean\_squared\_error',  
 cv=3,  
 n\_jobs=-1)  
grid\_2.fit(X, Y)  
  
st.write(grid\_2.best\_params\_)  
  
forest\_3 = RandomForestRegressor(n\_estimators=4, oob\_score=True, random\_state=5)  
forest\_3.fit(X, Y)  
Y\_predict3 = forest\_3.predict(X\_test)  
st.subheader('Средняя абсолютная ошибка:')  
st.write(mean\_absolute\_error(Y\_test, Y\_predict3))  
st.subheader('Средняя квадратичная ошибка:')  
st.write(mean\_squared\_error(Y\_test, Y\_predict3))  
st.subheader('Median absolute error:')  
st.write(median\_absolute\_error(Y\_test, Y\_predict3))  
st.subheader('Коэффициент детерминации:')  
st.write(r2\_score(Y\_test, Y\_predict3))  
  
fig1 = plt.figure(figsize=(7, 5))  
ax = plt.scatter(X\_test['K\_A\_Female\_Total\_scaled'], Y\_test, marker='o', label='Тестовая выборка')  
plt.scatter(X\_test['K\_A\_Female\_Total\_scaled'], Y\_predict3, marker='.', label='Предсказанные данные')  
plt.legend(loc='lower right')  
plt.xlabel('K\_A\_Female\_Total\_scaled')  
plt.ylabel('strength')  
plt.plot(n\_estimators\_1)  
st.pyplot(fig1)  
  
st.subheader('Градиентный бустинг')  
  
grad = GradientBoostingRegressor(n\_estimators=n\_estimators\_2, random\_state=random\_state\_2)  
grad.fit(X\_train, Y\_train)  
Y\_grad\_pred = grad.predict(X\_test)  
st.subheader('Средняя абсолютная ошибка:')  
st.write(mean\_absolute\_error(Y\_test, Y\_grad\_pred))  
st.subheader('Средняя квадратичная ошибка:')  
st.write(mean\_squared\_error(Y\_test, Y\_grad\_pred))  
st.subheader('Median absolute error:')  
st.write(median\_absolute\_error(Y\_test, Y\_grad\_pred))  
st.subheader('Коэффициент детерминации:')  
st.write(r2\_score(Y\_test, Y\_grad\_pred))  
  
fig2 = plt.figure(figsize=(7, 5))  
ax = plt.scatter(X\_test['K\_A\_Female\_Total\_scaled'], Y\_test, marker='o', label='Тестовая выборка')  
plt.scatter(X\_test['K\_A\_Female\_Total\_scaled'], Y\_grad\_pred, marker='.', label='Предсказанные данные')  
plt.legend(loc='lower right')  
plt.xlabel('K\_A\_Female\_Total\_scaled')  
plt.ylabel('strength')  
plt.plot(random\_state\_2)  
st.pyplot(fig2)  
  
st.subheader('Нахождение лучшего////')  
  
params = {  
 'n\_estimators': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 25, 50, 75, 100],  
 'max\_features': [0.2, 0.3, 0.4, 0.6, 0.8, 0.9, 1.0],  
 'min\_samples\_leaf': [0.01, 0.04, 0.06, 0.08, 0.1]  
}  
  
grid\_gr = GridSearchCV(estimator=GradientBoostingRegressor(random\_state=10),  
 param\_grid=params,  
 scoring='neg\_mean\_squared\_error',  
 cv=3,  
 n\_jobs=-1)  
grid\_gr.fit(X\_train, Y\_train)  
st.write(grid\_gr.best\_params\_)  
  
grad1 = GradientBoostingRegressor(n\_estimators=100, max\_features=0.8, min\_samples\_leaf=0.01, random\_state=10)  
grad1.fit(X\_train, Y\_train)  
Y\_grad\_pred1 = grad1.predict(X\_test)  
  
st.subheader('Средняя абсолютная ошибка:')  
st.write(mean\_absolute\_error(Y\_test, Y\_grad\_pred1))  
st.subheader('Средняя квадратичная ошибка:')  
st.write(mean\_squared\_error(Y\_test, Y\_grad\_pred1))  
st.subheader('Median absolute error:')  
st.write(median\_absolute\_error(Y\_test, Y\_grad\_pred1))  
st.subheader('Коэффициент детерминации:')  
st.write(r2\_score(Y\_test, Y\_grad\_pred1))  
  
fig1 = plt.figure(figsize=(7, 5))  
ax = plt.scatter(X\_test['K\_A\_Female\_Total\_scaled'], Y\_test, marker='o', label='Тестовая выборка')  
plt.scatter(X\_test['K\_A\_Female\_Total\_scaled'], Y\_grad\_pred1, marker='.', label='Предсказанные данные')  
plt.legend(loc='lower right')  
plt.xlabel('K\_A\_Female\_Total\_scaled')  
plt.ylabel('strength')  
plt.plot(n\_estimators\_1)  
st.pyplot(fig1)  
  
st.subheader('Построение линейной регрессии')  
  
Lin\_Reg = LinearRegression().fit(X\_train, Y\_train)  
  
lr\_y\_pred = Lin\_Reg.predict(X\_test)  
  
fig3 = plt.figure(figsize=(7, 5))  
plt.scatter(X\_test['K\_A\_Female\_Total\_scaled'], Y\_test, marker='s', label='Тестовая выборка')  
plt.scatter(X\_test['K\_A\_Female\_Total\_scaled'], lr\_y\_pred, marker='o', label='Предсказанные данные')  
plt.legend(loc='lower right')  
plt.xlabel('K\_A\_Female\_Total\_scaled')  
plt.ylabel('strength')  
plt.show()  
st.pyplot(fig3)  
  
st.subheader('Tree')  
  
clf = tree.DecisionTreeRegressor()  
clf = clf.fit(X\_train, Y\_train)  
  
lr\_y\_pred = clf.predict(X\_test)  
  
fig5 = plt.figure(figsize=(7, 5))  
plt.scatter(X\_test['K\_A\_Female\_Total\_scaled'], Y\_test, marker='s', label='Тестовая выборка')  
plt.scatter(X\_test['K\_A\_Female\_Total\_scaled'], lr\_y\_pred, marker='o', label='Предсказанные данные')  
plt.legend(loc='lower right')  
plt.xlabel('K\_A\_Female\_Total\_scaled')  
plt.ylabel('strength')  
plt.show()  
st.pyplot(fig5)  
  
st.subheader('Модель ближайших соседей для произвольного гиперпараметра K')  
  
Regressor\_5NN = KNeighborsRegressor(n\_neighbors = n\_estimators\_3)  
Regressor\_5NN.fit(X\_train, Y\_train)  
  
lr\_y\_pred = Regressor\_5NN.predict(X\_test)  
  
fig6 = plt.figure(figsize=(7, 5))  
plt.scatter(X\_test['K\_A\_Female\_Total\_scaled'], Y\_test, marker='s', label='Тестовая выборка')  
plt.scatter(X\_test['K\_A\_Female\_Total\_scaled'], lr\_y\_pred, marker='o', label='Предсказанные данные')  
plt.legend(loc='lower right')  
plt.xlabel('K\_A\_Female\_Total\_scaled')  
plt.ylabel('strength')  
plt.show()  
st.pyplot(fig6)

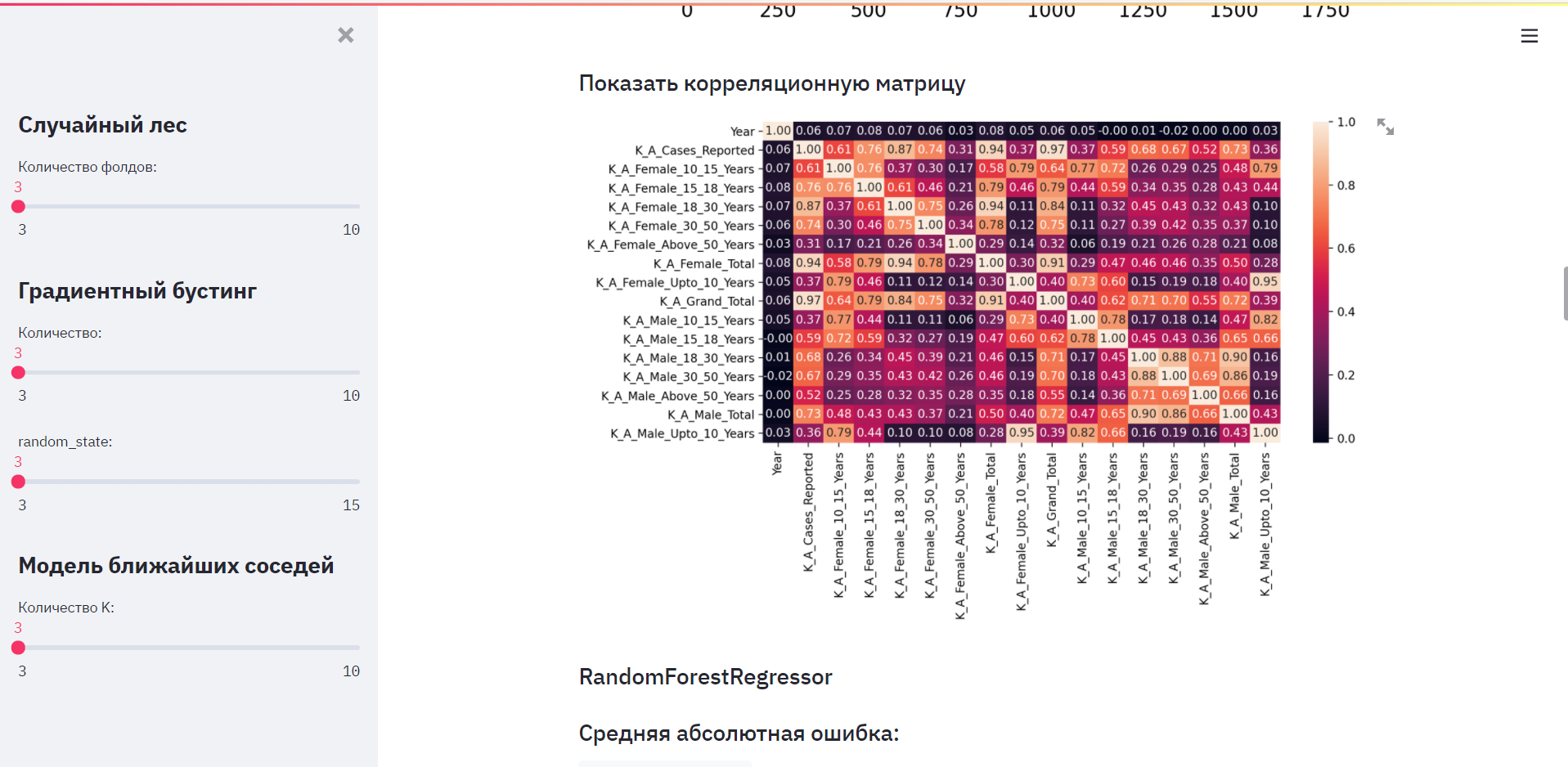
**Результат выполнения**

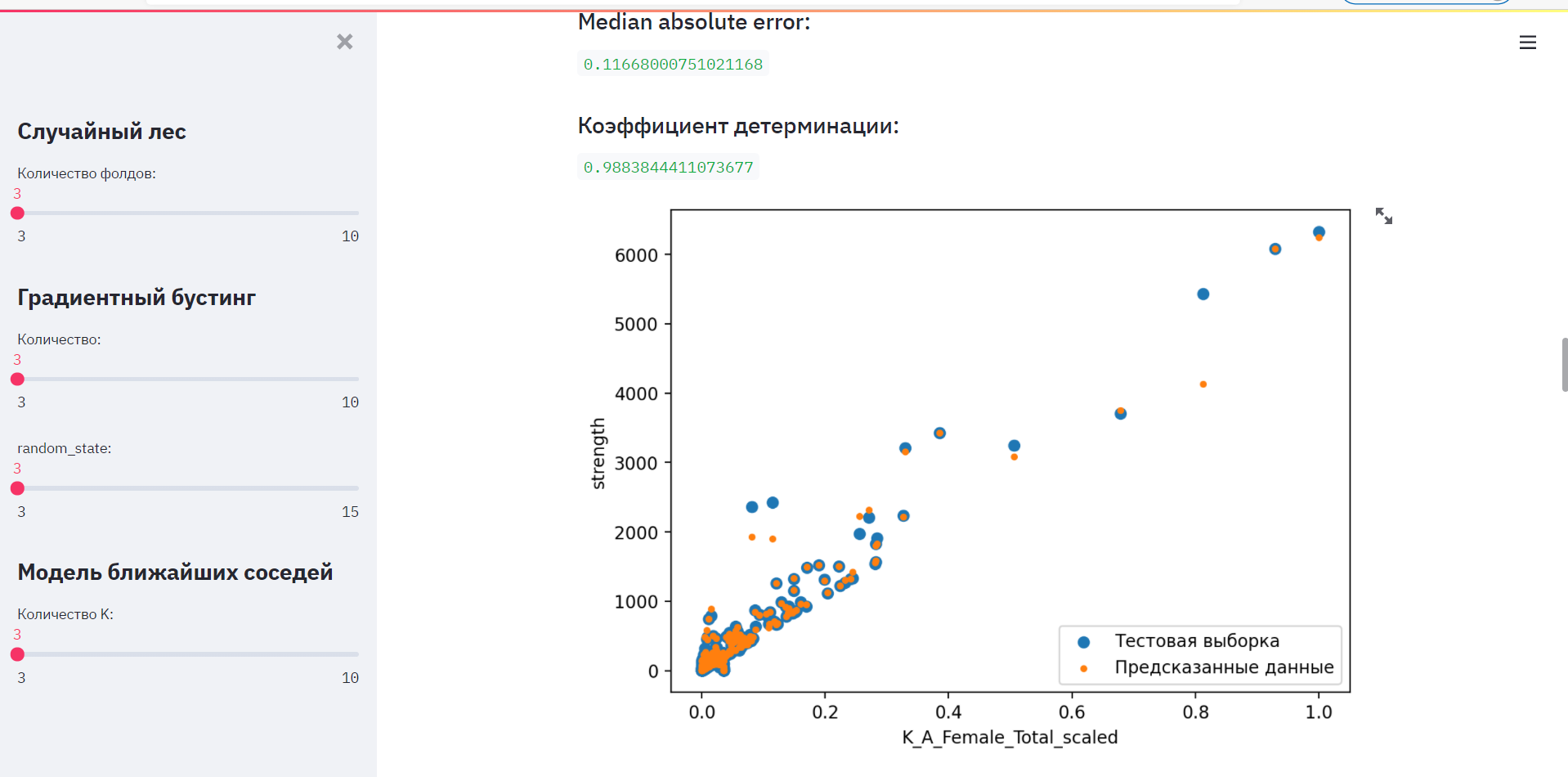
****

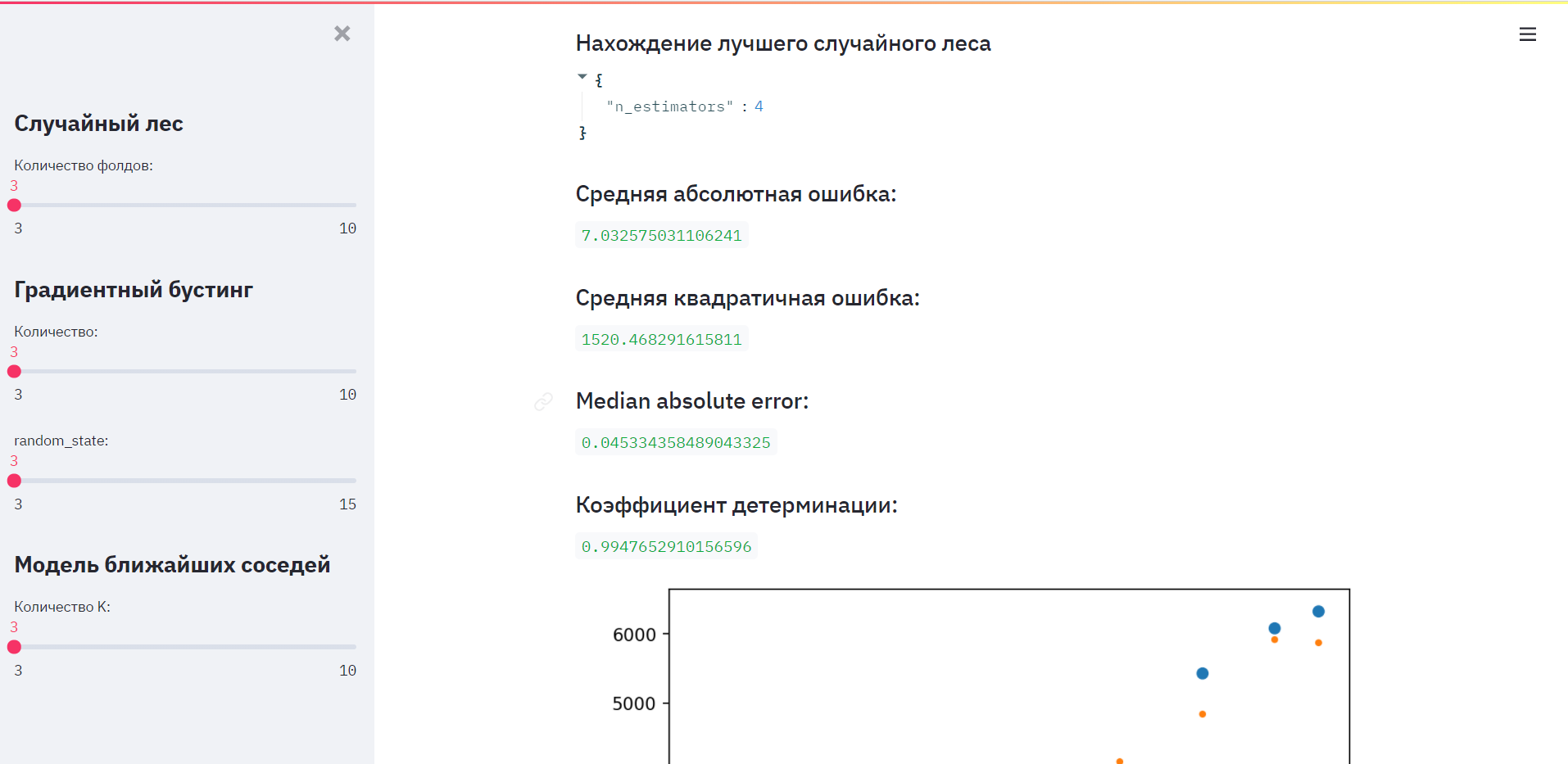
****

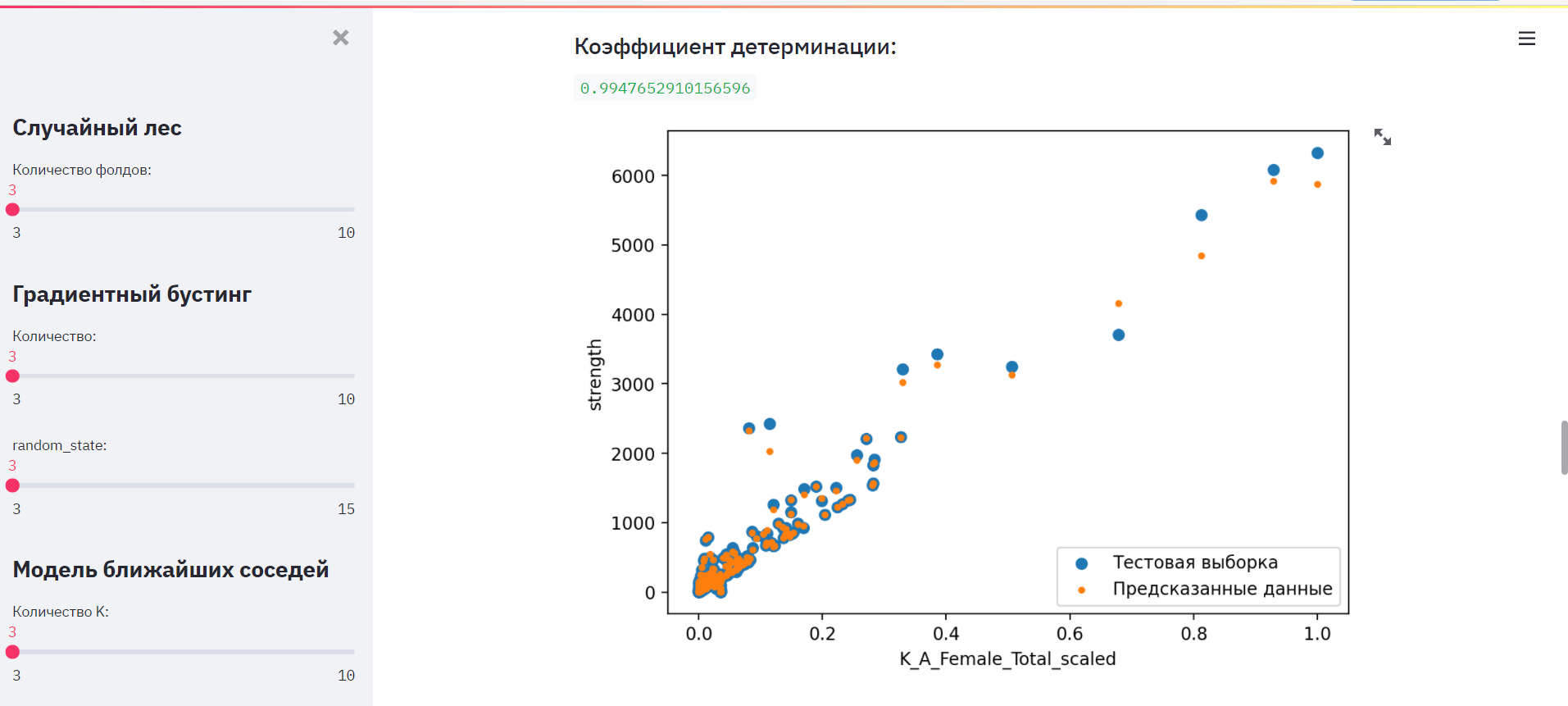
****

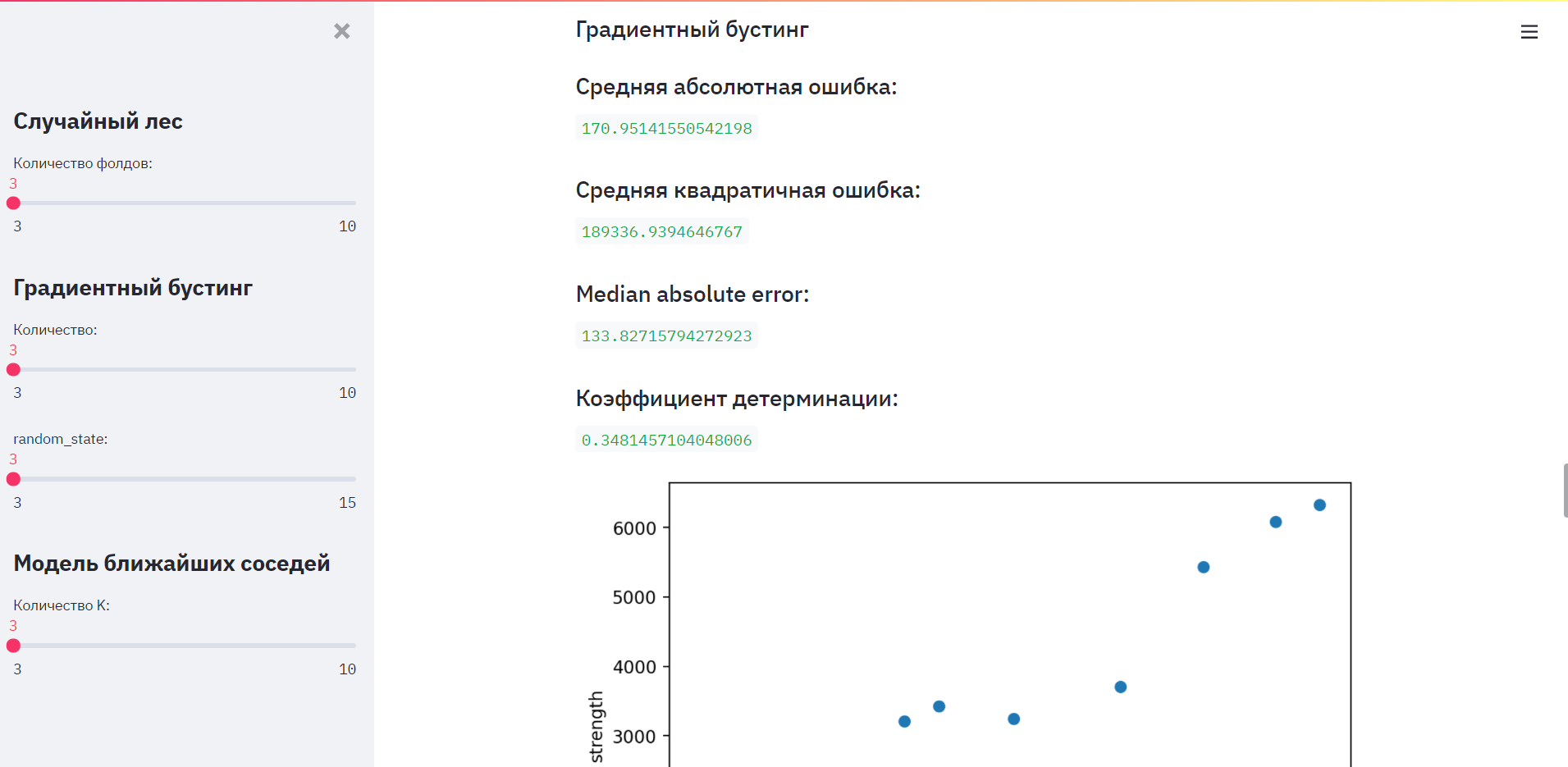
****

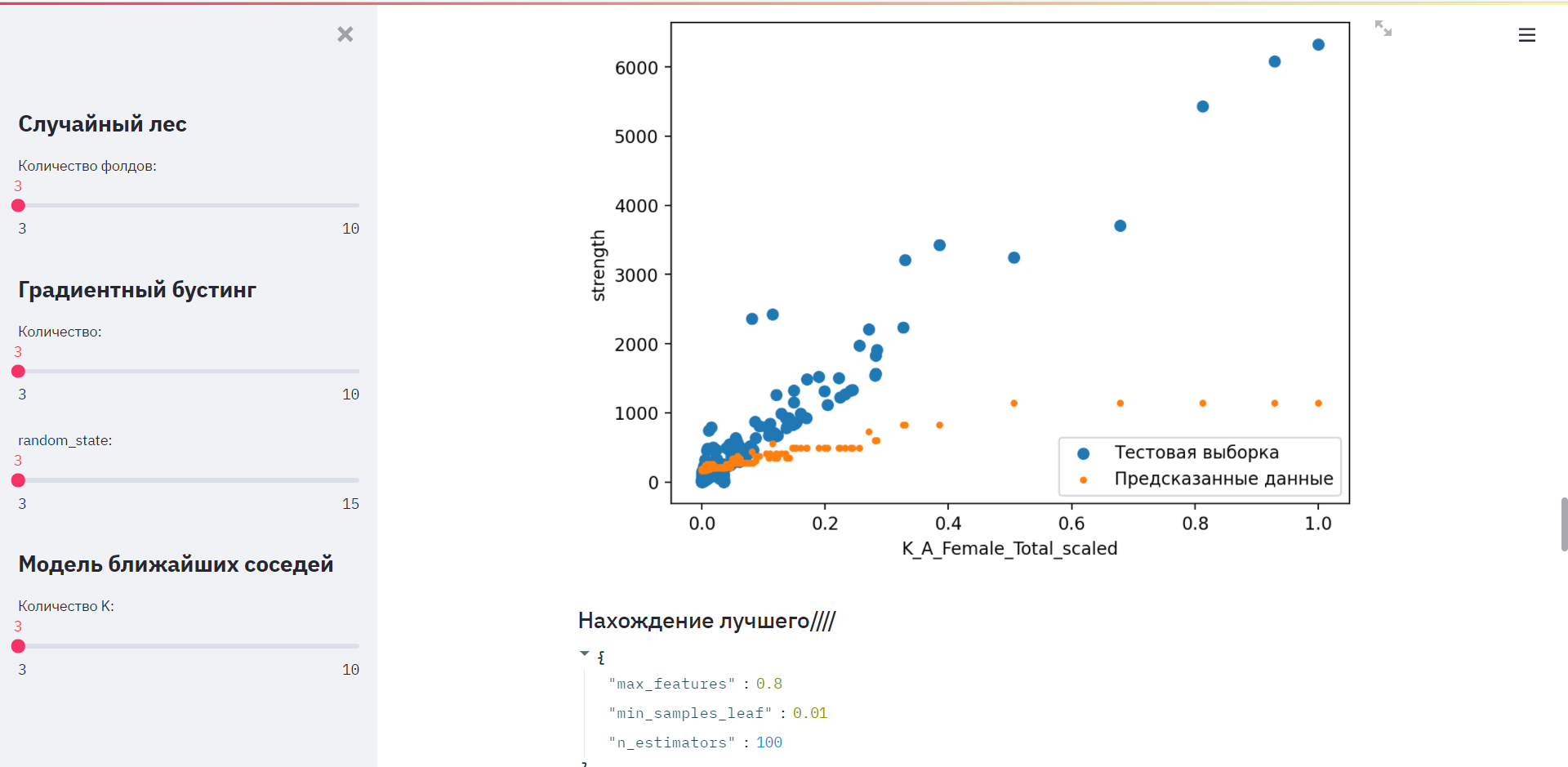
****

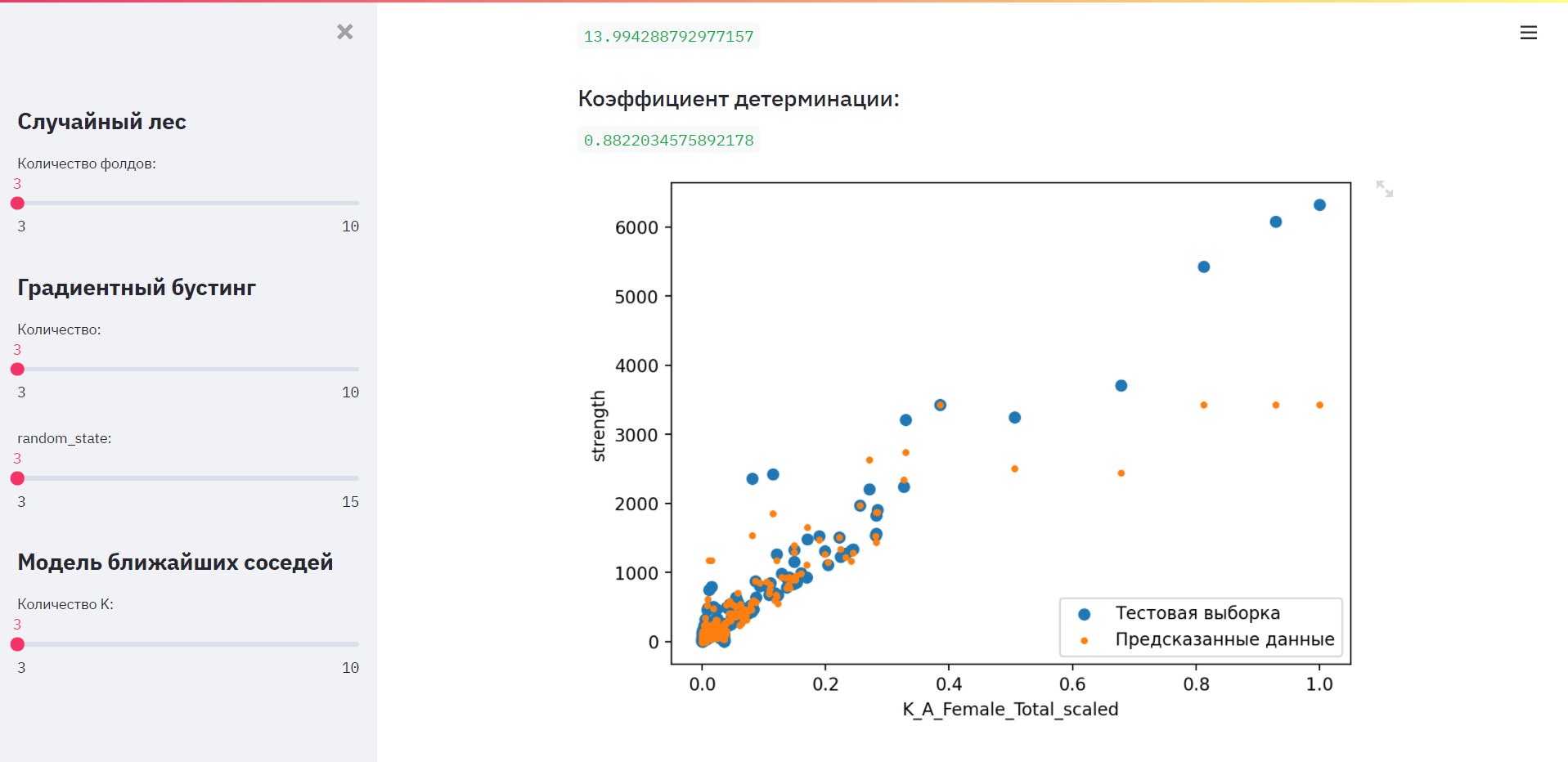
****

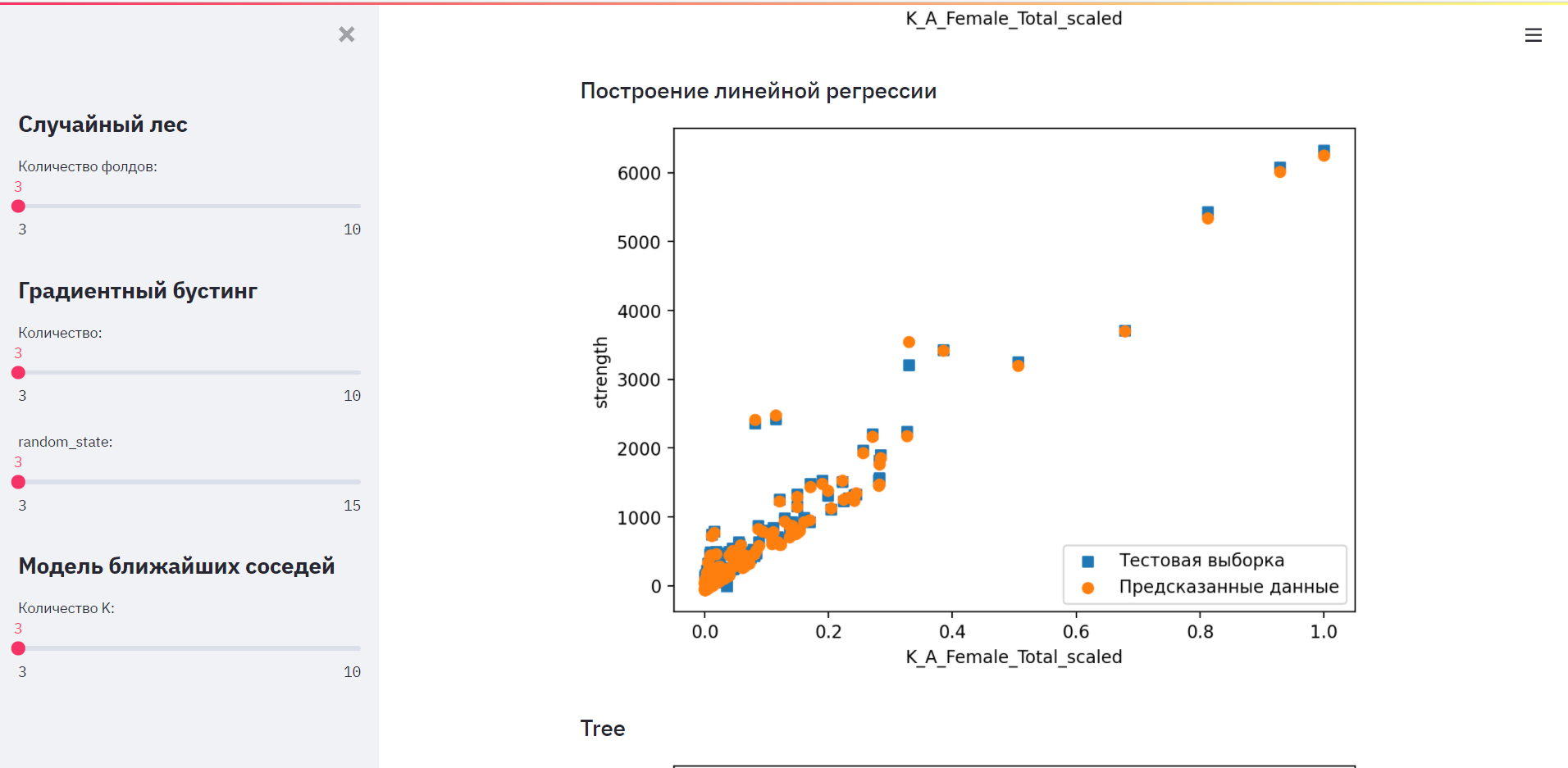
****

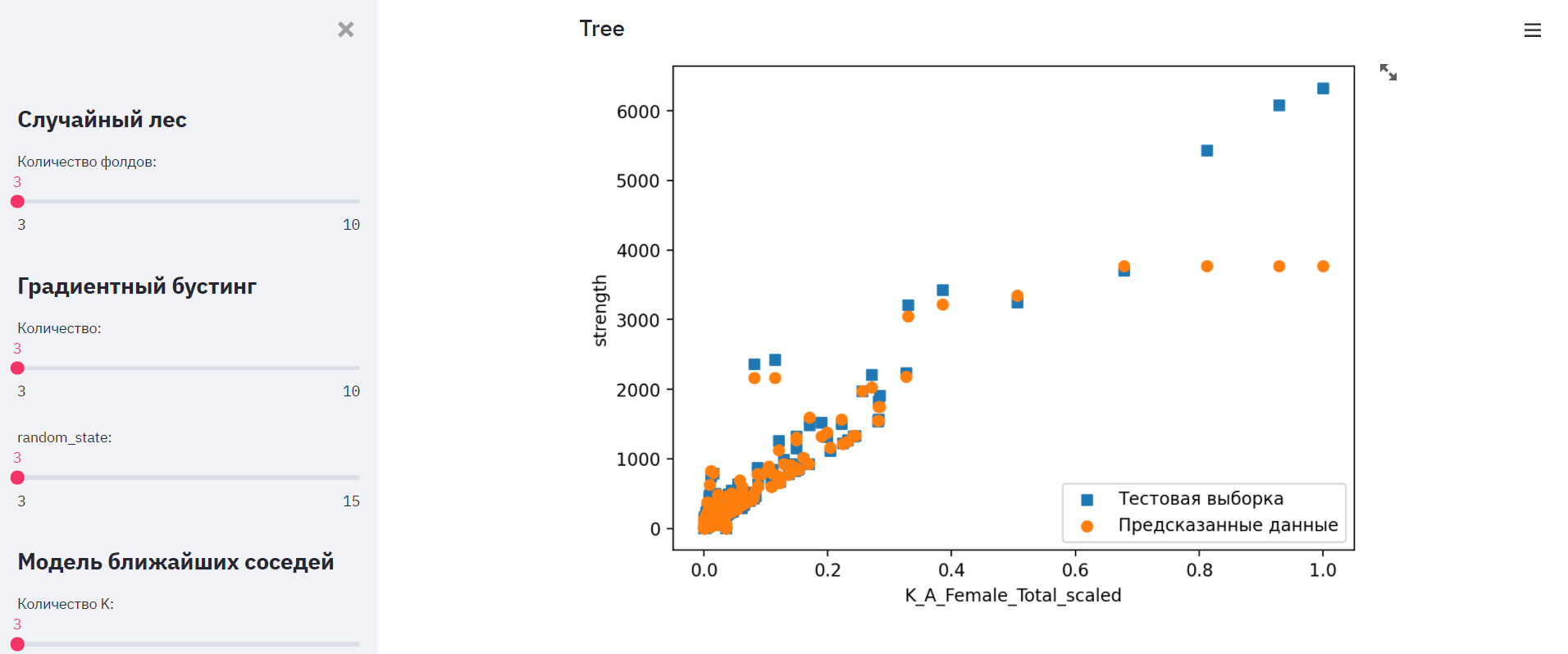
****

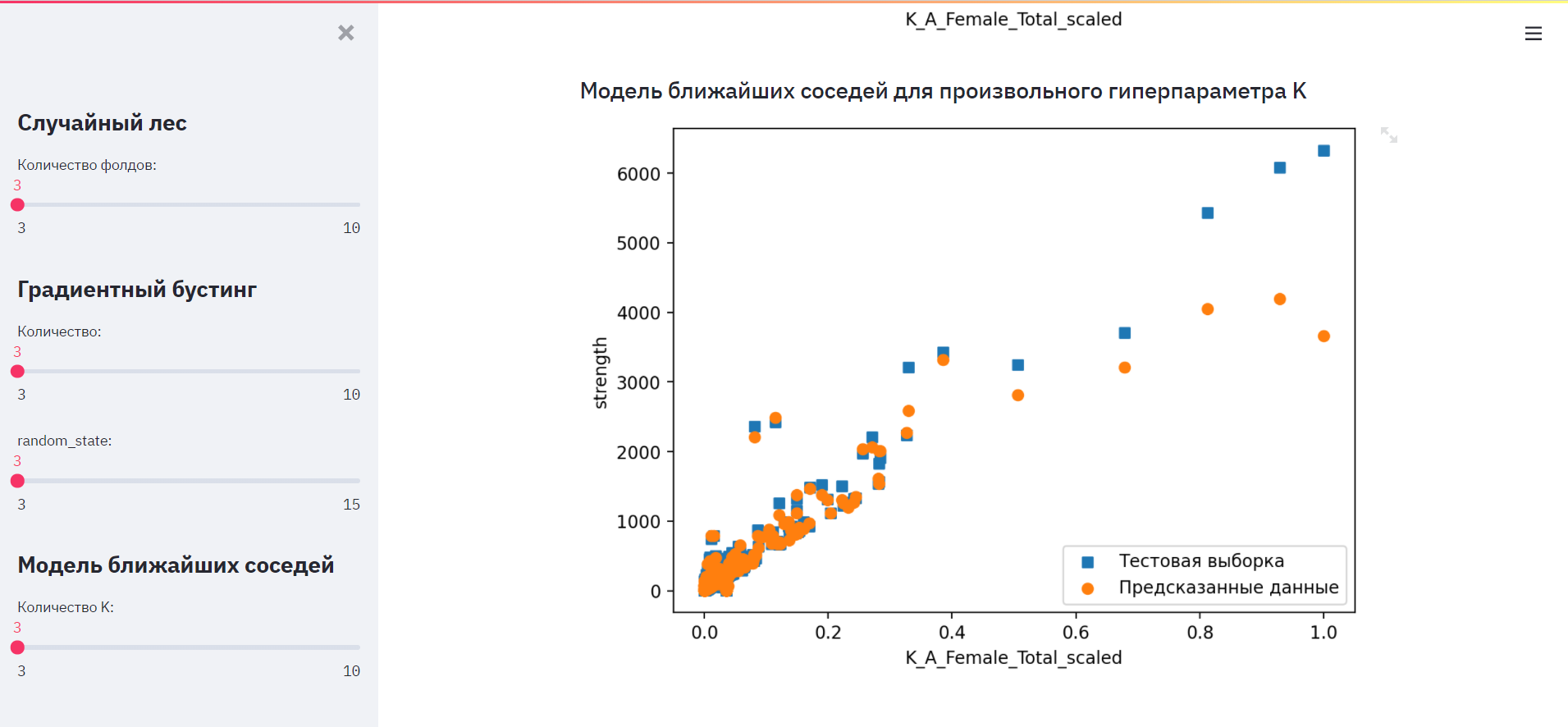
****

****

****

****

****

****

# Список источников информации

1. Репозиторий курса по Технологиям машинного обучения [Электронный ресурс] <https://github.com/ugapanyuk/ml_course_2021/wiki/COURSE_TMO>
2. Документация библиотеки AutoML TPOT [Электронный ресурс] https://github.com/EpistasisLab/tpot
3. Блог Александра Дьяконова. Ансамбли в машинном обучении [Электронный ресурс] <https://dyakonov.org/2019/04/19/>
4. Блог Александра Дьяконова. Градиентный бустинг [Электронный ресурс] https://dyakonov.org/2017/06/09/