

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего
образования
«Омский государственный технический университет»

Факультет информационных технологий и компьютерных систем
Кафедра «Прикладная математика и фундаментальная информатика»

РАСЧЕТНО-ГРАФИЧЕСКАЯ РАБОТА

по дисциплине Машинное обучение
на тему Разработка дашборда для инференса моделей и анализа данных

Студента(ки) Романовой Виктории Витальевны
фамилия, имя, отчество полностью

Курс 2 Группа МО-211

Направление 02.03.03. Математическое обеспечение и
администрирование информационных систем
код, наименование

Руководитель ассистент
должность, ученая степень, звание
Гуненков М. Ю. 05.06.2023
фамилия, инициалы, дата, подпись

Выполнил 05.06.2023
дата, подпись студента(ки)

Омск 2023

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	3
1. Описание модели	5
2. Создание дашборда	6
ЗАКЛЮЧЕНИЕ	10
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ	11
Приложение А. Результат работы приложения.....	12
Приложение Б. Исходный код.....	19

ВВЕДЕНИЕ

Машинное обучение (*ML*) — это направление искусственного интеллекта (ИИ), сосредоточенное на создании систем, которые обучаются и развиваются на основе получаемых ими данных. Искусственный интеллект — это широкий термин, который включает в себя компьютерные системы, имитирующие человеческий интеллект. Машинное обучение и ИИ часто идут бок о бок, и термины иногда используются взаимозаменяемо, но, строго говоря, это не одно и то же. Разница состоит в том, что машинное обучение всегда подразумевает использование ИИ, однако ИИ не всегда подразумевает машинное обучение.

Сегодня компьютеры работают бок о бок с человеком. Каждый раз, когда мы пользуемся банковскими услугами, делаем покупки в Интернете или общаемся в соцсетях, алгоритмы машинного обучения помогают сделать это взаимодействие удобнее, эффективнее и безопаснее. Машинное обучение и связанные с ним технологии быстро развиваются: их сегодняшние возможности только вершина айсберга.

В машинном обучении мы используем следующие библиотеки:

1. *NumPy* — библиотека с открытым исходным кодом с поддержкой многомерных массивов (включая матрицы) и высокоуровневых математических функций, предназначенных для работы с многомерными массивами.
2. *Matplotlib* — библиотека для визуализации данных. В ней строятся двумерные и трехмерные графики.
3. *SciPy* — библиотека с открытым исходным кодом, предназначенная для выполнения научных и инженерных расчётов.
4. *Pandas* — программная библиотека для обработки и анализа данных. Предоставляет специальные структуры данных и операции для манипулирования числовыми таблицами и временными рядами.

5. *Sklearn* — библиотека, включающая алгоритмы и инструменты для задач классификации, регрессии и кластеризации, а так же методы оценки производительности модели машинного обучения
6. *Pickle* — модуль для сериализации и десериализации объектов
7. *Streamlit* — фреймворк *Python* с открытым исходным кодом, используемый для развертывания моделей машинного обучения в веб-приложениях

1. Описание модели

Нами была выбрана модель градиентного бустинга для классификации объектов. Данная модель дала наилучший показатель детерминации 0,88.

Для начала мы предобработка данные стандартным способом – удалили дубликаты, категориальные признаки заменили на числовые, устранили пропущенные значения. Затем разбили датасет на обучающую и тестовую выборки, провели балансировку классов с помощью *RandomUnderSampler()* и стандартизацию с помощью *StandardScaler()*.

Затем была использована модель градиентного бустинга для классификации *GradientBoostingClassifier()* из библиотеки *sklearn*. С помощью команд *fit* и *predict* обучили нашу модель и сделали предсказания.

Затем нужно было сериализовать модель для дальнейшего использования в дашборде. Для этого мы использовали библиотеки *pickle*. Обученную модель сериализуем с помощью *pickle.dump()*.

Данные есть, модель обучена и сохранена, можно приступать к визуализации.

2. Создание дашборда

С дашбордами мы сталкиваемся каждый день. Это наглядный и эффективный инструмент маркетологов, бизнес-аналитиков, менеджеров проектов, *HR*-специалистов и учёных

Дашборд простыми словами — это интерактивная аналитическая панель, графический интерфейс. Смысл в том, что на одном экране расположены все ключевые метрики, показатели цели или процессов. С помощью этих метрик можно выявить и проанализировать тренды и изменения. Мы сталкиваемся с дашбордами каждый день. Приборная панель в автомобиле или графики активности в приложении фитнес-браслета — всё это дашборды.

Приступим к его реализации.

Для создания дашборда мы используем библиотеку *streamlit*. Для начала организуем функции загрузки датасет и нашей модели (рис.1).

```
def load_data():
    df = pd.read_csv(data)
    return df

def load_test_data():
    test_data_2 = pd.read_csv(data)
    test_data = test_data_2.drop(['dangerouse'], axis = 1)
    return test_data    #X

def load_model(path_to_file):
    with open(path_to_file, 'rb') as path_to_file:
        model = pickle.load(path_to_file)
    return model
```

Рисунок 1 — Функции загрузки

Заголовок получится с помощью *st.title()*, а обычный текст прописывается через *st.write()*.

Так же сделаем боковую панель с помощью `st.sidebar()`. Там же пропишем страницы, что у нас будут и дополнительную информацию (рис.2).

```
st.sidebar.title("Меню")

page = st.sidebar.selectbox(
    "Выберите страницу",
    ["Описание данных", "Запрос к модели"])

st.sidebar.info("здесь представлен дашборд о космических объектах")
st.sidebar.info("нажав кнопку ниже, вы можете просмотреть весь датасет на первой странице")
```

Рисунок 2 — *Sidebar*

Приступим к наполнению первой странице. На первой странице мы будем выводить название, датасет целиком, описание датасета и вывод того количества строк данных, которое укажет пользователь (рис.3).

```
if page == "Описание данных":
    st.title("Описание данных")

    show_about = st.checkbox('Показать описание датасета')
    if show_about == True:
        st.markdown("""
        этот датасет состоит из 27419 строк и 8 столбцов. Включает в себя описание различных свойств космических объектов. Рассмотрим
        * est_diameter_min - минимальный диаметр объекта,
        * est_diameter_max - максимальный диаметр объекта,
        * relative_velocity - относительная скорость движения,
        * miss_distance - расстояние до объекта,
        * absolute_magnitude - абсолютная величина,
        * dangerouse - представляет объект опасность или нет
        """)

    show_data = st.sidebar.checkbox('Отобразить датасет')
    if show_data == True:
        st.markdown(
            """## Здесь вы можете посмотреть все данные""")
        st.write(test_data)

    number = st.number_input('Введите количество строк для отображения', 2)
    st.write('Вы выбрали показ на', number, 'строк')
    st.markdown("""## Здесь вы можете посмотреть все данные""")
    st.write(df.head(number))
```

Рисунок 3 — Первая страница

Приступим к наполнению второй страницы с запросами. Пользователю будет доступно 3 запроса: показать коэффициент детерминации, предсказать значения, показать графики распределение данных (рис.4).

```
elif page == "Запрос к модели":
    st.title("Запрос к модели")
    request = st.selectbox(
        "Выберите запрос",
        ["SCORE", "Предсказанные значения", "Графики"]
    )
```

Рисунок 4 — Запросы

Коэффициент детерминации просто выводим на экран (рис.5).

```
if request == "SCORE":
    st.header("Коэффициент детерминации: score = 0.87")
```

Рисунок 5 — Коэффициент детерминации

Приступаем к предсказанным значениям (рис.6).

```
elif request == "Предсказанные значения":
    number = st.number_input('Введите количество строк для отображения', 2)

    st.write("Первые", number)
    first_pred = model.predict(X_test)
    #st.write(first_pred[:number])
    dange = 0
    for item in first_pred[:number]:
        if item == 1:
            st.write(f"{item:.2f}", "представляет угрозу")
            dange += 1
        elif item == 0:
            st.write(f"{item:.2f}", "не представляет угрозу")
    st.write(dange, "объектов представляют угрозу")

    show_com = st.checkbox('Показать комментарий')
    if show_com == True:
        st.write("Данный набор данных был обучен с помощью модели градиентного бустинга")
```

Рисунок 6 — Предсказания

Через *predict* получаем предсказание модели и выводим то число предсказаний, которое указал пользователь. Так же прописываем интерпретацию выведенного значения (их у нас всего 2).

Комментарий: модель обучена с помощью Градиентного бустинга (*GradientBoostingClassifier*) из библиотеки *sklearn*.

Последний запрос включает в себя показ графиков (рис.7).

```
elif request == "Графики":
    parametr = st.selectbox("Выберите параметр для графика:",
                           ['relative_velocity', 'miss_distance', 'absolute_magnitude'])
    arg = parametr
    argument = df[arg]
    fig, ax = plt.subplots()
    ax.hist(argument, bins=20)
    st.pyplot(fig)
```

Рисунок 7 — Графики

Пользователь может указать распределение какого признака он хочет увидеть, и ему продемонстрируется график выбранного признака. Графики строятся с помощью библиотеки *matplotlib*.

ЗАКЛЮЧЕНИЕ

В рамках задания были предобработаны данные, создана и обучена модель машинного обучения. Так же был создан дашборд для наглядного представления полученных результатов. Было создано несколько страниц, на которых выведены данные, предсказания модели машинного обучения и визуализированы данные с помощью графиков. Так же были изучены модуль *pickle* и фреймворк *streamlit*.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Документация Streamlit. URL: <https://streamlit.io/> (дата обращения: 03.06.2023).
2. Сохранение и восстановление моделей. URL: https://rukovodstvo.net/posts/id_1322/ (дата обращения: 03.06.2023).
3. Создание интерактивных панелей с Streamlit и Python. URL: <https://proglib.io/p/sozdanie-interaktivnyh-paneley-s-streamlit-i-python-2021-06-21> (дата обращения: 01.06.2023).
4. Машинное обучение в streamlit: делаем это понятным для бизнеса. URL: <https://habr.com/ru/companies/otus/articles/701412/> (дата обращения: 04.06.2023).
5. Что такое машинное обучение. URL: [https://www.oracle.com/cis/artificial-intelligence/machine-learning/what-is-machine-learning/#:~:text=%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5%20%D0%BE%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5%20\(ML\)%20%E2%80%94%D1%8D%D1%82%D0%BE,%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D1%8B%D0%B5%20%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B%2C%20%D0%B8%D0%BC%D0%B8%D1%82%D0%B8%D1%80%D1%83%D1%8E%D1%89%D0%B8%D0%B5%20%D1%87%D0%B5%D0%BB%D0%BE%D0%B2%D0%B5%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9%20%D0%B8%D0%BD%D1%82%D0%B5%D0%BB%D0%BB%D0%B5%D0%BA%D1%82](https://www.oracle.com/cis/artificial-intelligence/machine-learning/what-is-machine-learning/#:~:text=%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5%20%D0%BE%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5%20(ML)%20%E2%80%94%D1%8D%D1%82%D0%BE,%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D1%8B%D0%B5%20%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B%2C%20%D0%B8%D0%BC%D0%B8%D1%82%D0%B8%D1%80%D1%83%D1%8E%D1%89%D0%B8%D0%B5%20%D1%87%D0%B5%D0%BB%D0%BE%D0%B2%D0%B5%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9%20%D0%B8%D0%BD%D1%82%D0%B5%D0%BB%D0%BB%D0%B5%D0%BA%D1%82) (дата обращения: 04.06.2023).
6. Что такое дашборды, какие они бывают и как их строить. URL: <https://practicum.yandex.ru/blog/chto-takoe-dashbord/> (дата обращения: 04.06.2023).

Приложение А

(обязательное)

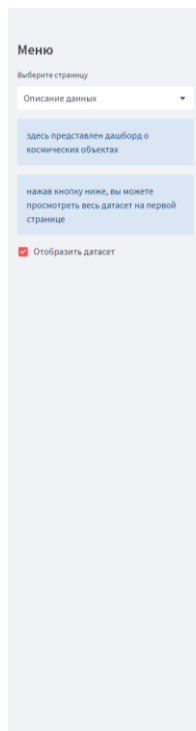
Результат работы приложения.

Первая страница (рис.А.1).



Рисунок А.1 — Первая страница

Укажем «Отобразить датасет», «отобразить описание» (рис. А.2).



NEO

Это дашборд про космические объекты

Описание данных

☒ Показать описание датасета

этот датасет состоит из 27419 строк и 8 столбцов. Включает в себя описание различных свойств космических объектов. Рассмотрим данные:

- `est_diameter_min` - минимальный диаметр объекта,
- `est_diameter_max` - максимальный диаметр объекта,
- `relative_velocity` - относительная скорость движения,
- `miss_distance` - расстояние до объекта,
- `absolute_magnitude` - абсолютная величина,
- `dangerouse` - представляет объект опасностью или нет

Здесь вы можете посмотреть все данные

	<code>est_diameter_min</code>	<code>relative_velocity</code>	<code>miss_distance</code>	<code>absolute_magnitude</code>	
0	0.016	56,014.0785	1,024,332.9981	26.1	
1	0.0305	7,864.3481	32,681,860.6268	24.7	
2	0.0555	55,257.5445	65,386,361.8441	23.4	
3	0.0193	41,531.4047	12,607,957.9881	25.7	
4	0.1395	67,639.3945	71,305,897.3204	21.4	
5	0.0462	18,933.8758	28,470,829.526	23.8	
6	0.116	43,184.4027	34,184,167.4813	21.8	
7	0.0211	38,064.8023	26,511,535.9537	25.5	
8	0.2211	33,736.86	56,706,016.8526	20.4	
9	0.2424	56,188.2156	61,754,410.9384	20.2	

Введите количество строк для отображения

2

Вы выбрали показ на 2 строк

Здесь вы можете посмотреть все данные

<code>est_diameter_min</code>	<code>relative_velocity</code>	<code>miss_distance</code>	<code>absolute_magnitude</code>	<code>dangerouse</code>
-------------------------------	--------------------------------	----------------------------	---------------------------------	-------------------------

Рисунок А.2

Увеличим число строк для отображения (рис. А.3, рис. А.4).

NEO

Это дашборд про космические объекты

Описание данных

☐ Показать описание датасета

Введите количество строк для отображения

5

Вы выбрали показ на 5 строк

Здесь вы можете посмотреть все данные

	<code>est_diameter_min</code>	<code>relative_velocity</code>	<code>miss_distance</code>	<code>absolute_magnitude</code>	<code>dangerouse</code>
0	0.016	56,014.0785	1,024,332.9981	26.1	0
1	0.0305	7,864.3481	32,681,860.6268	24.7	0
2	0.0555	55,257.5445	65,386,361.8441	23.4	0
3	0.0193	41,531.4047	12,607,957.9881	25.7	0
4	0.1395	67,639.3945	71,305,897.3204	21.4	0

Рисунок А.3 — Число строк для отображения

NEO

Это дашборд про космические объекты

Описание данных

☐ Показать описание датасета

Введите количество строк для отображения

18

Вы выбрали показ на 18 строк

Здесь вы можете посмотреть все данные

	est_diameter_min	relative_velocity	miss_distance	absolute_magnitude	dangerouse
0	0.016	56,014.0785	1,024,332.9981	26.1	0
1	0.0305	7,864.3481	32,681,860.6268	24.7	0
2	0.0555	55,257.5445	65,386,361.8441	23.4	0
3	0.0193	41,531.4047	12,607,957.9881	25.7	0
4	0.1395	67,639.3945	71,305,897.3204	21.4	0
5	0.0462	18,933.8758	28,470,829.526	23.8	0
6	0.116	43,184.4027	34,184,167.4813	21.8	0
7	0.0211	38,064.8023	26,531,535.9537	25.5	0
8	0.2211	33,736.86	56,706,616.8326	20.4	0
9	0.2424	56,188.2156	61,754,410.9184	20.2	0

Рисунок А.4 — Увеличение числа строк

Переходим на вторую страницу (рис.А.5).

Меню

Выберите страницу

Описание данных

Описание данных

Запрос к модели

нажав кнопку ниже, вы можете
просмотреть весь датасет на первой
странице

☐ Отобразить датасет

Рисунок А.5 — Переход на вторую страницу

Коэффициент детерминации (первый запрос) (рис. А.6).

NEO

Это дашборд про космические объекты

Запрос к модели

Выберите запрос

SCORE

Коэффициент детерминации: score = 0.87

Рисунок А.6 — Коэффициент детерминации

Предсказанные значения (второй запрос) (рис. А.7).

NEO

Это дашборд про космические объекты

Запрос к модели

Выберите запрос

Предсказанные значения

Введите количество строк для отображения

2

Первые 2

1.00 представляет угрозу

0.00 не представляет угрозу

1 объектов представляют угрозу

☒ Показать комментарий

Данный набор данных был обучен с помощью модели градиентного бустинга

Рисунок А.7 — Предсказание

Можем увеличить число значений (рис. А.8).

NEO

Это дашборд про космические объекты

Запрос к модели

Выберите запрос

Предсказанные значения

Введите количество строк для отображения

28

Первые 28

1.00 представляет угрозу
1.00 представляет угрозу
0.00 не представляет угрозу
1.00 представляет угрозу
0.00 не представляет угрозу
0.00 не представляет угрозу
0.00 не представляет угрозу
0.00 не представляет угрозу
0.00 не представляет угрозу
0.00 не представляет угрозу
0.00 не представляет угрозу
0.00 не представляет угрозу
1.00 представляет угрозу
0.00 не представляет угрозу
1.00 представляет угрозу
0.00 не представляет угрозу
0.00 не представляет угрозу
1.00 представляет угрозу
0.00 не представляет угрозу
0.00 не представляет угрозу

Рисунок А.8 — Увеличение числа предсказаний

Графики (третий запрос) (рис.А.9, рис.А.10, рис.А.11).

NEO

Это дашборд про космические объекты

Запрос к модели

Выберите запрос

Графики

Выберите параметр для графика:

relative_velocity

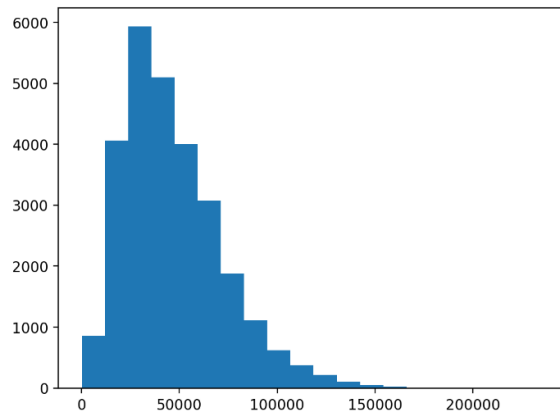


Рисунок А.9 — Первый график

NEO

Это дашборд про космические объекты

Запрос к модели

Выберите запрос

Графики

Выберите параметр для графика:

absolute_magnitude

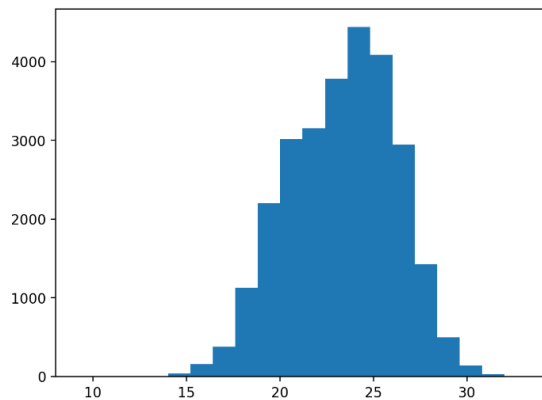


Рисунок А.10 — Второй график

NEO

Это дашборд про космические объекты

Запрос к модели

Выберите запрос

Графики

Выберите параметр для графика:

miss_distance

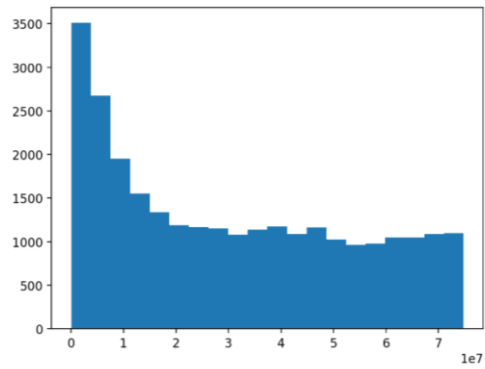


Рисунок А.11 — Третий график

Приложение Б (обязательное)

Исходный код

```
import streamlit as st
import pandas as pd
import pickle
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from imblearn.under_sampling import RandomUnderSampler

data = ('Neo_dash_new.csv')

def load_data():
    df = pd.read_csv(data)
    return df

def load_test_data():
    test_data_2 = pd.read_csv(data)
    test_data = test_data_2.drop(['dangerouse'], axis = 1)
    return test_data

def load_model(path_to_file):
    with open(path_to_file, 'rb') as path_to_file:
        model = pickle.load(path_to_file)
    return model

def processing():
    df = load_data()
    y = df['dangerouse']
    X = load_test_data()
    underSampler = RandomUnderSampler(sampling_strategy='majority')
    X_under_sample, y_under_sample = underSampler.fit_resample(X, y)
    X_train, X_test, y_train, y_test = train_test_split(X_under_sample,
y_under_sample, test_size=0.2, random_state=4)
    scaler = StandardScaler().fit(X_train)
    X_train = scaler.transform(X_train)
    X_test = scaler.transform(X_test)

    return X_train, X_test, y_train, y_test

def main():
    name = 'GradBoost.pkl'
    df = load_data()
    test_data = load_test_data()
    model = load_model(name)
    X_train, X_test, y_train, y_test = processing()
```

```

st.title("NEO")
st.write("""Это дашборд про космические объекты""")

st.sidebar.title("Меню")

page = st.sidebar.selectbox(
    "Выберите страницу",
    ["Описание данных", "Запрос к модели"])

st.sidebar.info("здесь представлен дашборд о космических объектах")
st.sidebar.info("нажав кнопку ниже, вы можете просмотреть весь датасет на
первой странице")

if page == "Описание данных":
    st.title("Описание данных")

    show_about = st.checkbox('Показать описание датасета')
    if show_about == True:
        st.markdown("""
        этот датасет состоит из 27419 строк и 8 столбцов. Включает в себя
описание различных свойств космических объектов. Рассмотрим данные:
        * est_diameter_min - минимальный диаметр объекта,
        * est_diameter_max - максимальный диаметр объекта,
        * relative_velocity - относительная скорость движения,
        * miss_distance - расстояние до объекта,
        * absolute_magnitude - абсолютная величина,
        * dangerous - представляет объект опасность или нет
        """)

    show_data = st.sidebar.checkbox('Отобразить датасет')
    if show_data == True:
        st.markdown(
            "#### Здесь вы можете посмотреть все данные")
        st.write(test_data)

2)
number = st.number_input('Введите количество строк для отображения',

st.write('Вы выбрали показ на', number, 'строк')
st.markdown("#### Здесь вы можете посмотреть все данные")
st.write(df.head(number))

elif page == "Запрос к модели":
    st.title("Запрос к модели")
    request = st.selectbox(
        "Выберите запрос",
        ["SCORE", "Предсказанные значения", "Графики"]
    )
    if request == "SCORE":
        st.header("Коэффициент детерминации: score = 0.87")

    elif request == "Предсказанные значения":
        number = st.number_input('Введите количество строк для
отображения', 2)

```

```

st.write("Первые", number)
first_pred = model.predict(X_test)
dange = 0
for item in first_pred[:number]:
    if item == 1:
        st.write(f"{item:.2f}", "представляет угрозу")
        dange += 1
    elif item == 0:
        st.write(f"{item:.2f}", "не представляет угрозу")
st.write(dange, "объектов представляют угрозу")

show_com = st.checkbox('Показать комментарий')
if show_com == True:
    st.write("Данный набор данных был обучен с помощью модели
градиентного бустинга")

elif request == "Графики":
    parametr = st.selectbox("Выберите параметр для графика:",
                            ['relative_velocity', 'miss_distance',
'absolute_magnitude'])
    arg = parametr
    argument = df[arg]
    fig, ax = plt.subplots()
    ax.hist(argument, bins=20)
    st.pyplot(fig)

if __name__ == "__main__":
    main()

```