

**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)»**

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

**по курсу
«Data Science»**

**Тема: «Прогнозирование конечных свойств новых материалов
(композиционных материалов)»**

Слушатель

Митченко Богдан Сергеевич

Москва, 2025

Содержание

Содержание.....	2
Введение	3
1. Аналитическая часть	5
1.1. Постановка задачи.....	5
1.2. Описание используемых методов.....	7
1.3. Разведочный анализ данных	1010
2. Практическая часть	13
2.1. Предобработка данных	13
2.2. Разработка и обучение модели	15
2.3. Тестирование модели	19
2.4. Написать нейронную сеть, которая будет рекомендовать соотношение «матрица-наполнитель»	24
2.5. Разработка приложения	26
2.6. Создание удалённого репозитория и загрузка	29
2.7. Заключение	30
2.8. Список используемой литературы и веб ресурсы.....	32

Введение

Композиционные материалы представляют собой уникальные структуры, состоящие из двух или более различных компонентов, которые сохраняют свои физические и химические свойства, но создают новые характеристики, не свойственные каждому из компонентов по отдельности. Эти материалы отличаются высокой прочностью, устойчивостью к внешним воздействиям и легкостью, что делает их незаменимыми в различных областях, таких как авиация, космонавтика, строительство и автомобилестроение.

Одним из наиболее перспективных типов композиционных материалов является базальтопластик — композит, в основе которого лежат базальтовые волокна и органическое связующее. Базальтопластик обладает исключительными эксплуатационными характеристиками, такими как высокая коррозионная устойчивость, стойкость к химическим воздействиям, а также отличные механические свойства, что позволяет использовать его вместо металлических материалов в тех областях, где важна легкость и долговечность.

Процесс разработки и оптимизации новых композитных материалов часто сталкивается с рядом трудностей, включая длительные циклы тестирования и высокие затраты на исследования. Однако, с учетом важности этих материалов для промышленности, необходимо найти способы ускорить этот процесс, снизив затраты на разработку и тестирование. Одним из таких методов является прогнозирование свойств композитных материалов с использованием математических моделей и алгоритмов машинного обучения. Это позволяет значительно сократить время, необходимое для получения требуемых характеристик материалов, а также оптимизировать их состав.

В данной работе рассматривается задача прогнозирования конечных свойств новых композиционных материалов, в частности, базальтопластика. Для этого будут использованы методы машинного обучения, такие как случайный лес, градиентный бустинг, метод опорных векторов и нейронные сети, для создания моделей, которые могут предсказать такие характеристики, как модуль упругости и прочность при растяжении на основе данных о компонентах композита.

Целью работы является разработка и обучение моделей для прогноза этих характеристик, а также создание нейронной сети, которая будет рекомендовать оптимальное соотношение компонентов — «матрица-наполнитель» — для достижения нужных свойств материала. В рамках исследования также будет создано веб-приложение, которое позволит пользователю получить прогнозные значения свойств композита на основе введенных данных о материалах.

Таким образом, эта работа направлена на решение важной задачи прогнозирования свойств новых композиционных материалов с использованием современных методов машинного обучения, что поможет ускорить процесс разработки новых материалов и значительно снизить затраты на их тестирование.

1. Аналитическая часть

1.1. Постановка задачи

В данной работе рассматривается задача прогнозирования механических свойств композитных материалов с использованием методов машинного обучения. Особое внимание уделяется прочности и модулю упругости, поскольку эти характеристики во многом определяют эксплуатационные возможности композитов. Актуальность исследования обусловлена потребностью в быстром и точном определении свойств новых материалов без проведения длительных и дорогостоящих физических испытаний.

Цель работы состоит в разработке прогностической модели способной по данным о составе и технологии изготовления материала предсказывать прочность и модуль упругости композита, а также рекомендуемой оптимальное соотношение «матрица–наполнитель» для достижения требуемых характеристик.

Исходные данные для исследования представлены двумя таблицами (X_bp.xlsx и X_nup.xlsx), содержащими набор параметров образцов композитов. Эти отдельные наборы данных были объединены по индексу (идентификатору образца) в единый массив, на основе которого осуществляется дальнейший анализ. При объединении часть строк были удалены, поскольку для них не нашлось соответствующих записей в другом наборе данных; таким образом, в итоговый массив вошли только образцы, присутствующие одновременно в обоих исходных файлах.

Загрузка данных

```
[ ]: df_bp = pd.read_excel('X_bp.xlsx')  
df_nup = pd.read_excel('X_nup.xlsx')
```

Объединение данных по индексу

```
[9]: data = pd.merge(df_bp, df_nup, left_index=True, right_index=True, how='inner')
```

Рисунок 1 - загрузка и объединение таблиц

Для реализации поставленной задачи на объединенном наборе данных применяется алгоритмы машинного обучения, что позволяет выявить сложные нелинейные зависимости между параметрами состава и результирующими механическими свойствами материала. Методы машинного обучения уже зарекомендовали себя как эффективный инструмент прогнозирования характеристик материалов и активно используются в задачах материаловедения.

В частности, показано, что современные модели (например, нейронные сети и градиентный бустинг) способны с высокой точностью предсказывать ключевые свойства композитов — включая соотношение матрицы и наполнителя, модуль упругости и прочность.

Приведем столбец "Угол нашивки" к значениям 0 и 1 и integer

```
[11]: data = data.replace({'Угол нашивки, град': {0.0 : 0, 90.0 : 1}})
      data['Угол нашивки, град'] = data['Угол нашивки, град'].astype(int)
```

Удаления лишних столбцов

```
[12]: data = data[data.columns.drop(list(data.filter(regex='Unnamed: 0_x')))]
      data = data[data.columns.drop(list(data.filter(regex='Unnamed: 0_y')))]
```

Проверим на пропущенные данные

```
[13]: data.isnull().sum()
```

```
[13]: 0
```

Рисунок 2 - подготовка данных

При подготовке данных для моделирования выполняется их визуализация и разведочный анализ, позволяющие оценить распределения показателей и выявить возможные аномалии или корреляции. Далее проводится очистка данных: удаляются выявленные выбросы и некорректные либо неполные записи, чтобы исключить искажающее влияние на модель. Кроме того, осуществляется нормализация признаков, поскольку исходные параметры могут существенно различаться по масштабам. После такой предобработки данных построена итоговая модель, которая способна достаточно точно прогнозировать прочность и модуль упругости материала по заданным параметрам состава, а также выдавать рекомендацию относительно оптимального соотношения «матрица–наполнитель»

в материале. Таким образом модель позволит сократить число необходимых физических экспериментов и ускорить разработку новых композитных материалов с заданными свойствами. датасете.

Нарисуем ящики с усами (боксплоты)

```
[21]: scaler.fit(data)
plt.figure(figsize = (20, 20))
plt.suptitle('Диаграммы "ящики с усами"', y = 0.9,
           fontsize = 30)
plt.boxplot(pd.DataFrame(scaler.transform(data)), labels = data.columns, patch_artist = True, meanline = True, vert = False, boxprops = dict(
plt.show())
```

<ipython-input-21-1674748863>:5: MatplotlibDeprecationWarning: The 'labels' parameter of boxplot() has been renamed 'tick_labels' since Matplotlib 3.9; support for the old name will be dropped in 3.11.
plt.boxplot(pd.DataFrame(scaler.transform(data)), labels = data.columns, patch_artist = True, meanline = True, vert = False, boxprops = dict(facecolor = 'g', color = 'y'), medianprops = dict(color = 'lime'), whiskerprops = dict(color = 'g'), capprops = dict(color = 'black'), flierprops = dict(color = 'y', markeredgecolor = 'maroon'))

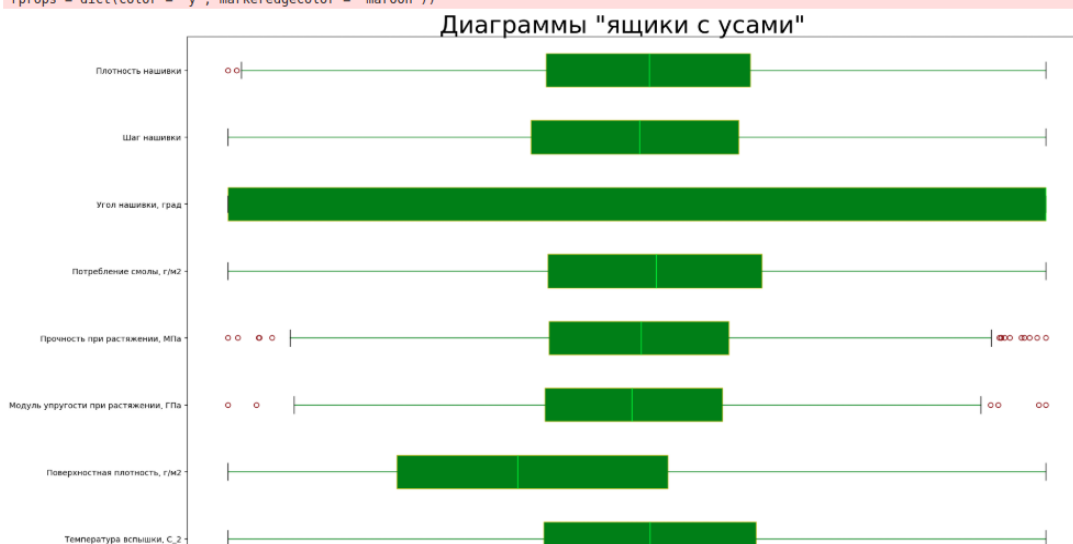


Рисунок 2 - визуализация данных диаграмма "ящик с усами" после очистки

1.2. Описание используемых методов

Для решения задачи прогнозирования прочностных характеристик композитных материалов были применены пять методов машинного обучения: случайный лес (Random Forest), градиентный бустинг (Gradient Boosting), линейная регрессия (Linear Regression), метод опорных векторов (SVR) и многослойный перцептрон (MLP). Также используются метрики качества: R^2 , MSE, MAE.

Random Forest — ансамблевый метод, объединяющий множество решающих деревьев. Он устойчив к переобучению, способен выявлять важные признаки и подходит для сложных нелинейных зависимостей. Применим при любом

распределении данных и не требует масштабирования. Недостатки: высокая ресурсозатратность, трудная интерпретация, невозможность экстраполяции.

Gradient Boosting — метод последовательного обучения деревьев, каждый из которых корректирует ошибки предыдущих. Он достигает высокой точности, особенно при правильной настройке параметров. Преимущества: гибкость, точность, поддержка кастомных функций потерь. Недостатки: подверженность переобучению, сложность настройки, медленное обучение. Тем не менее, часто показывает лучшие результаты среди ансамблей.

Линейная регрессия (Linear Regression) — один из простейших алгоритмов, предполагающий линейную зависимость между входными признаками и целевой переменной. Быстро обучается, легко интерпретируется, но плохо справляется с нелинейными зависимостями и чувствителен к выбросам. Используется как базовая модель для сравнения и анализа влияния признаков.

SVR строит регрессионную модель с допуском погрешности (эпсилон-зоной) и использует ядровые функции для учета нелинейности. Применим при небольшом объеме данных и высокой размерности признаков. Преимущества: устойчивость к переобучению, точность на малых выборках. Недостатки: низкая масштабируемость, чувствительность к параметрам и шуму, трудная интерпретация.

MLP — нейронная сеть с несколькими скрытыми слоями. Способен аппроксимировать любые зависимости при достаточном объеме данных. Преимущества: высокая точность, гибкость. Недостатки: склонность к переобучению, сложная настройка, "черный ящик". Используется для оценки потенциала глубокого обучения в задаче.

Метрики качества:

R^2 (коэффициент детерминации): доля объясненной дисперсии. Чем ближе к 1, тем лучше.

MSE (среднеквадратичная ошибка): чувствителен к большим ошибкам, измеряется в квадрате единиц.

MAE (средняя абсолютная ошибка): более робастна к выбросам, измеряется в тех же единицах, что и целевая переменная.

Таблица 5 – Сравнительная таблица методов

Метод	Преимущества	Недостатки	Интерпретируемость	Область применения
Random Forest	Точность, устойчивость, масштабируемость	Тяжелый, не экстраполирует	Средняя	Сложные данные, табличные задачи
Boosting	Высокая точность, гибкость	Настройка, переобучение	Низкая	Любые сложные данные
Linear Reg.	Простота, интерпретируемость	Линейность, выбросы	Высокая	Простейшие зависимости
SVR	Точность, ядровая гибкость	Медленный, чувствителен	Низкая	Небольшие выборки
MLP	Универсальность, высокая точность	Настройка, переобучение	Очень низкая	Большие данные, сложные задачи

В работе все методы применялись к одной и той же задаче, что позволило провести сравнительный анализ их эффективности. Результаты показали, что ансамблевые модели и нейросеть демонстрируют неплохое качество прогноза, при этом линейная регрессия и SVR немного проигрывают по точности.

1.3. Разведочный анализ данных

Прежде чем передать данные в работу моделей машинного обучения, необходимо обработать и очистить их. Очевидно, что «грязные» и необработанные данные могут содержать искажения и пропущенные значения – это ненадёжно, поскольку способно привести к крайне неверным результатам по итогам моделирования. Но безосновательно удалять что-либо тоже неправильно. Именно поэтому сначала набор данных надо изучить.

data.describe()

	Соотношение матрица-наполнитель	Плотность, кг/м3	модуль упругости, ГПа	Количество отвердителя, м. %	Содержание эпоксидных групп, %_2	Температура всплышки, C_2	Поверхностная плотность, г/м2	Модуль упругости при растяжении, ГПа	Прочность при растяжении, МПа	Потребление смолы, г/м2	Угол нашивки, град	Шаг нашивки	Плотность нашивки
count	1023.000000	1023.000000	1023.000000	1023.000000	1023.000000	1023.000000	1023.000000	1023.000000	1023.000000	1023.000000	1023.000000	1023.000000	1023.000000
mean	2.930366	1975.734888	739.923233	110.570769	22.244390	285.882151	482.731833	73.328571	2466.922843	218.423144	0.491691	6.899222	57.153929
std	0.913222	73.729231	330.231581	28.295911	2.406301	40.943260	281.314690	3.118983	485.628006	59.735931	0.500175	2.563467	12.350969
min	0.389403	1731.764635	2.436909	17.740275	14.254985	100.000000	0.603740	64.054061	1036.856605	33.803026	0.000000	0.000000	0.000000
25%	2.317887	1924.155467	500.047452	92.443497	20.608034	259.066528	266.816645	71.245018	2135.850448	179.627520	0.000000	5.080033	49.799212
50%	2.906878	1977.621657	739.664328	110.564840	22.230744	285.896812	451.864365	73.268805	2459.524526	219.198882	0.000000	6.916144	57.341920
75%	3.552660	2021.374375	961.812526	129.730366	23.961934	313.002106	693.225017	75.356612	2767.193119	257.481724	1.000000	8.586293	64.944961
max	5.591742	2207.773481	1911.536477	198.953207	33.000000	413.273418	1399.542362	82.682051	3848.436732	414.590628	1.000000	14.440522	103.988901

Рисунок 3 - описательная статистика датасета

Цель разведочного анализа - получение первоначальных представлений о характерах распределений переменных исходного набора данных, формирование оценки качества исходных данных (наличие пропусков, выбросов), выявление характера взаимосвязи между переменными с целью последующего выдвижения гипотез о наиболее подходящих для решения задачи моделях машинного обучения.

```
[14] data.duplicated().sum()
np.int64(0)
```

Рисунок 4 - проверка датасета на наличие дубликатов

В качестве инструментов разведочного анализа используется: оценка статистических характеристик датасета; гистограммы распределения каждой из

переменной (несколько различных вариантов); диаграммы ящика с усами (несколько интерактивных вариантов); попарные графики рассеяния точек (несколько вариантов); график «квантиль-квантиль»; тепловая карта (несколько вариантов); описательная статистика для каждой переменной; анализ и полное исключение выбросов (5 повторных итераций); проверка наличия пропусков и дубликатов; ранговая корреляция Кендалла и Пирсона.

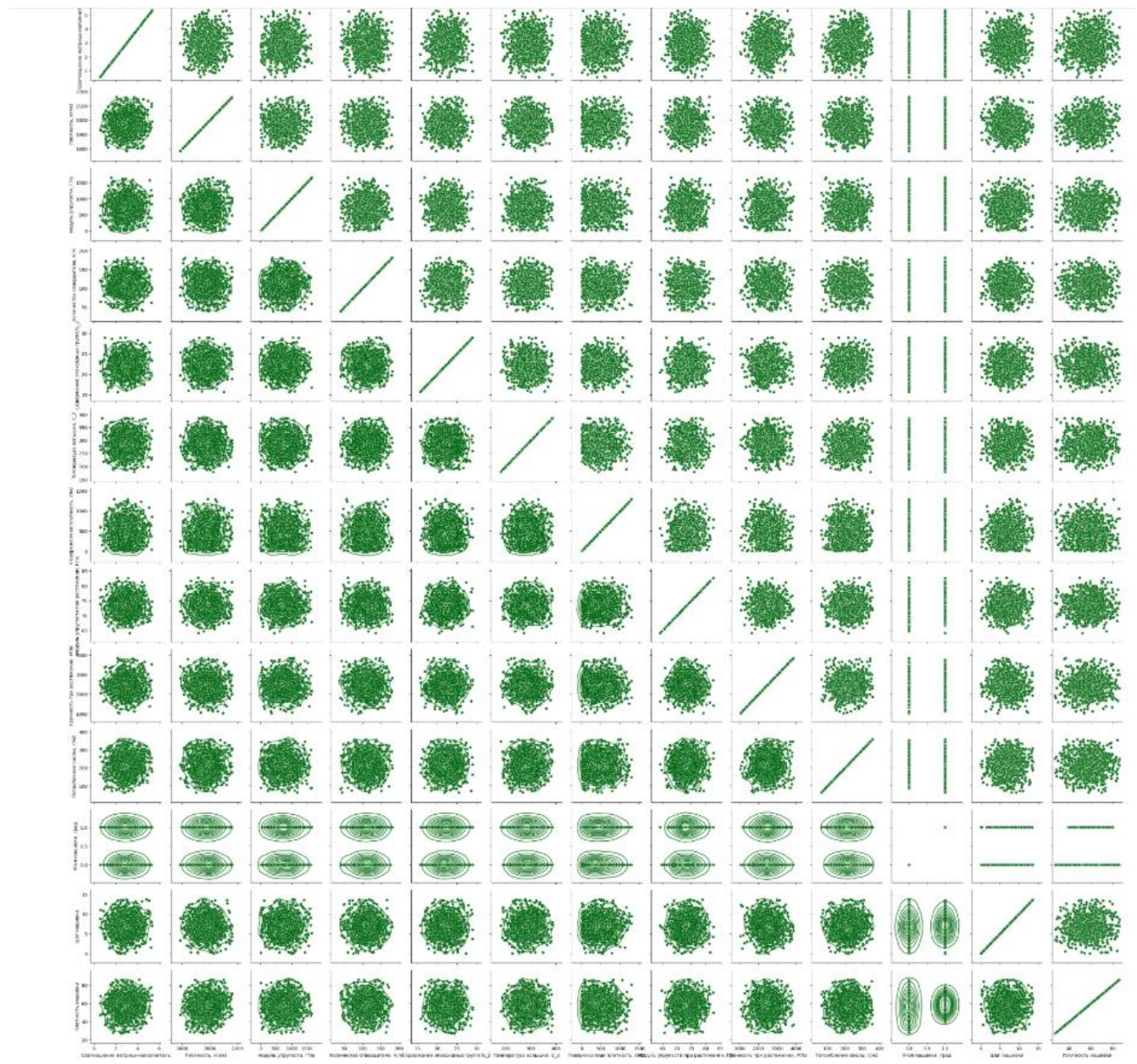
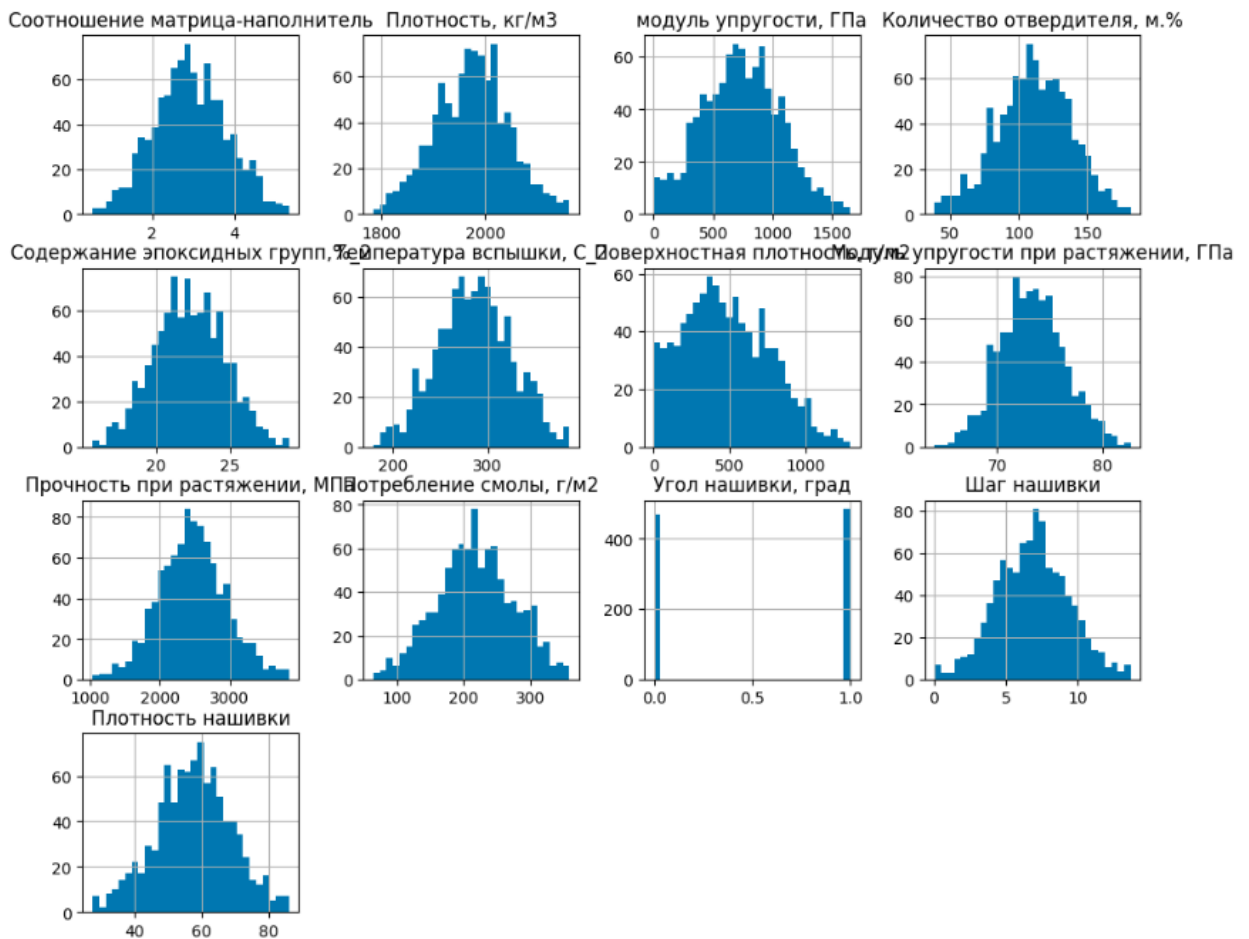


Рисунок 5 - попарные графики рассеяния точек

```
plt.show()
```

Гистограммы распределений переменных



При проведении анализа выявлены параметры близкие к нормальному:

- Соотношение матрица-наполнитель;

Рисунок 6 - гистограммы распределения

Гистограммы используются для изучения распределений частот значений переменных. Мы видим очень слабую корреляцию между переменными.

После обнаружения выбросов данные, значительно отличающиеся от выборки, будут полностью удалены.

Данные объединённого датасета не имеют чётко выраженной зависимости, что подтверждает тепловая карта с матрицей корреляции и матрицы диаграмм рассеяния.

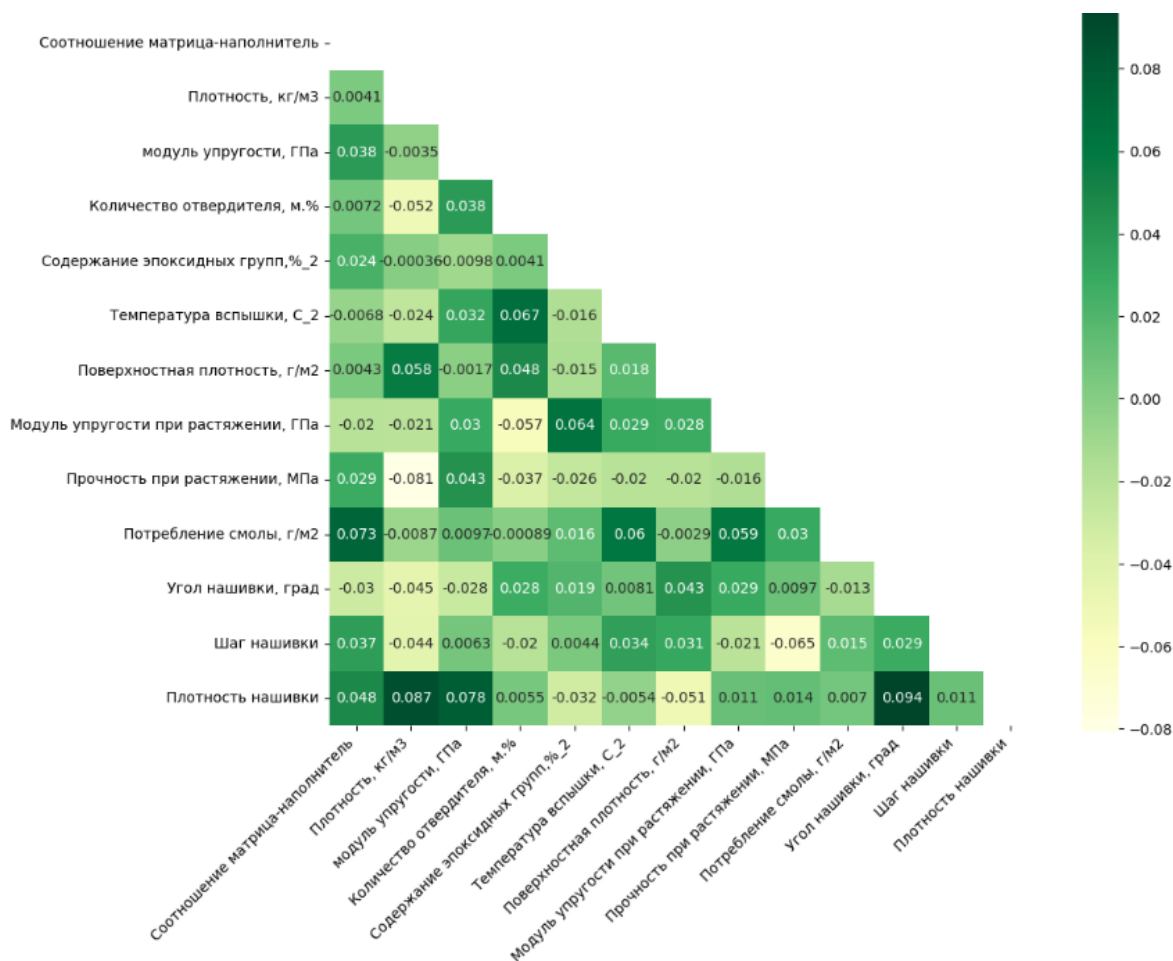


Рисунок 7 - тепловая карта с корреляцией данных

Максимальная корреляция между плотностью нашивки и углом нашивки 0.094, значит нет зависимости между этими данными. Корреляция между всеми параметрами очень близка к 0, корреляционные связи между переменными не наблюдаются.

2. Практическая часть

2.1. Предобработка данных

В ходе проведённого анализа удаляем лишние столбцы в которых хранится id двух таблиц, а так же принимаем решение столбец "Угол нашивки" привести к виду «0» и «1».

Объединение данных по индексу

```
[5]: data = pd.merge(df_bp, df_nup, left_index=True, right_index=True, how='inner')
```

Приведем столбец "Угол нашивки" к значениям 0 и 1 и integer

```
[6]: data = data.replace({'Угол нашивки, град': {0.0 : 0, 90.0 : 1}})
data['Угол нашивки, град'] = data['Угол нашивки, град'].astype(int)
```

Удаления лишних столбцов

```
[7]: data = data[data.columns.drop(list(data.filter(regex='Unnamed: 0_x')))]
data = data[data.columns.drop(list(data.filter(regex='Unnamed: 0_y')))]
```

Рисунок 8 - часть кода с преобразованием столбца "Угол нашивки"

По условиям задания нормализуем значения. Для этого применим MinMaxScaler(), затем применим Normalizer().

Нормализация данных с использованием MinMaxScaler

```
[15]: scaler = MinMaxScaler()
data_normalized = pd.DataFrame(scaler.fit_transform(data_clean), columns=data_clean.columns)
```

..

Рисунок 9 - нормализация данных

Проверим, как изменились данные после нормализации

```
[18]: data.head()
```

	Соотношение матрица-наполнитель	Плотность, кг/м3	модуль упругости, ГПа	Количество отвердителя, м.%	Содержание эпоксидных групп,%_2	Температура вспышки, C_2	Поверхностная плотность, г/м2	Модуль упругости при растяжении, ГПа	Прочность при растяжении, МПа	Потребление смолы, г/м2	Угол нашивки, град	Ц нашив
1	1.857143	2030.0	738.736842	50.00	23.750000	284.615385	210.0	70.0	3000.0	220.0	0	
3	1.857143	2030.0	738.736842	129.00	21.250000	300.000000	210.0	70.0	3000.0	220.0	0	
4	2.771331	2030.0	753.000000	111.86	22.267857	284.615385	210.0	70.0	3000.0	220.0	0	
5	2.767918	2000.0	748.000000	111.86	22.267857	284.615385	210.0	70.0	3000.0	220.0	0	
6	2.569620	1910.0	807.000000	111.86	22.267857	284.615385	210.0	70.0	3000.0	220.0	0	

```
[17]: data_normalized.head()
```

	Соотношение матрица-наполнитель	Плотность, кг/м3	модуль упругости, ГПа	Количество отвердителя, м.%	Содержание эпоксидных групп,%_2	Температура вспышки, C_2	Поверхностная плотность, г/м2	Модуль упругости при растяжении, ГПа	Прочность при растяжении, МПа	Потребление смолы, г/м2	Угол нашивки, град	Ц нашив
0	0.282131	0.601381	0.447061	0.123047	0.607435	0.482823	0.16223	0.319194	0.698235	0.517418	0.0	0.2751
1	0.282131	0.601381	0.447061	0.608021	0.418887	0.549664	0.16223	0.319194	0.698235	0.517418	0.0	0.3445
2	0.457857	0.601381	0.455721	0.502800	0.495653	0.482823	0.16223	0.319194	0.698235	0.517418	0.0	0.3445
3	0.457201	0.527898	0.452685	0.502800	0.495653	0.482823	0.16223	0.319194	0.698235	0.517418	0.0	0.3445
4	0.419084	0.307448	0.488508	0.502800	0.495653	0.482823	0.16223	0.319194	0.698235	0.517418	0.0	0.3445

Рисунок 10 - визуализированные данные до и после нормализации

2.2. Разработка и обучение модели

В рамках исследования были выбраны и реализованы пять моделей машинного обучения, предназначенных для прогнозирования двух основных характеристик композитных материалов:

1. Модуля упругости при растяжении (ГПа);
2. Прочности при растяжении (МПа).

```
Обновляем данные для модели

[44]: features = [
        'Угол нашивки, град',
        'Шаг нашивки',
        'Плотность нашивки',
        'Соотношение матрица-наполнитель',
        'Плотность, кг/м3',
        'модуль упругости, ГПа',
        'Количество отвердителя, м.%',
        'Содержание эпоксидных групп,%_2',
        'Температура вспышки, C_2',
        'Поверхностная плотность, г/м2',
        'Потребление смолы, г/м2'
    ]

    targets = [
        'Модуль упругости при растяжении, ГПа',
        'Прочность при растяжении, МПа'
    ]

[45]: X_cleaned = data[features].values
      y_cleaned = data[targets].values
```

Рисунок 11 - обозначение изучаемых и прогнозируемых значений

Все модели обучались на одном и том же наборе признаков, включающем параметры нашивки, плотности, химического состава и условий термической обработки. Перед обучением производилась предварительная обработка данных: удаление выбросов по методу IQR, нормализация признаков с помощью MinMaxScaler и StandardScaler, а также коррекционный анализ.

Перечень используемых моделей:

1. Случайный лес (Random Forest)

- а) базовая реализация с параметром `n_estimators=100`;
- б) одходящий выбор для построения интерпретируемых, устойчивых моделей без сложной настройки;

в) обучена на тренировочной выборке и протестирована на отложенной части данных;

✓ Разделение данных на тренировочную и тестовую выборки

```
[ ] X_train, X_test, y_train, y_test = train_test_split(feature, target, test_size=0.2, random_state=42)
```

✓ Обучаем RandomForestRegressor

```
[ ] rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
```

Рисунок 12 - Случайный лес (Random Forest)

2. Градиентный бустинг (Gradient Boosting Regressor);

а) использовался в обёртке MultiOutputRegressor, так как задача включает два выходных параметра;

б) показал высокий уровень точности, особенно в отношении модуля упругости;

✓ Обучаем GradientBoostingRegressor

```
[ ] gb_model = GradientBoostingRegressor(n_estimators=100, random_state=42)
multioutput_model = MultiOutputRegressor(gb_model)
```

✓ Обучаем модель

```
multioutput_model.fit(X_train, y_train)
```

Рисунок 13 - Градиентный бустинг (Gradient Boosting Regressor)

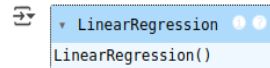
3. Линейная регрессия (Linear Regression);

а) служила базовой моделью для сравнения;

б) быстро обучалась, но демонстрировала наихудшее качество прогноза по сравнению с другими методами;

✓ Создаем модель линейной регрессии

```
[ ] lr_model = LinearRegression()
    lr_model.fit(X_train, y_train)
```



```
LinearRegression()
```

✓ Прогнозируем и оцениваем модель

```
lr_predictions = lr_model.predict(X_test)
lr_mse = mean_squared_error(y_test, lr_predictions)
lr_r2 = r2_score(y_test, lr_predictions)
```

Рисунок 14 - Линейная регрессия (Linear Regression)

4. Метод опорных векторов (SVR);

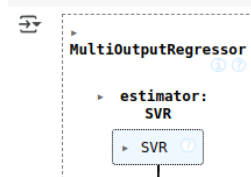
а) реализован с RBF-ядром;

б) также применялся в рамках MultiOutputRegressor;

в) обучение было затратным по времени, а результаты оказались хуже, чем у деревьев и нейросети;

✓ Создаем модель SVR с ядром RBF

```
[ ] svr_model = SVR(kernel='rbf')
    multioutput_model = MultiOutputRegressor(svr_model)
    multioutput_model.fit(X_train, y_train)
```



```
MultiOutputRegressor
  estimator:
    SVR
```

✓ Обучаем модель

```
[ ] multioutput_model.fit(X_train, y_train)
```

Рисунок 15 - Метод опорных векторов (SVR)

5. Многослойный персептрон (MLP);

а) реализован с использованием библиотеки TensorFlow/Keras;

б) содержал три скрытых слоя с функциями активации ReLU и регуляризацией Dropout;

в) обучался с использованием EarlyStopping и оптимизатора Adam;

г) достиг сопоставимой точности с лучшими ансамблевыми методами;

6. Подготовка данных к обучению;

а) выборка была разделена в пропорции 80/20 на обучающую и тестовую части.

б) выполнена очистка от пропущенных значений и удаление дубликатов.

в) масштабирование входных признаков проводилось по методике StandardScaler.

г) в качестве целевых переменных использовались два числовых признака: модуль упругости и прочность при растяжении.

```
[ ] X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_scaled, test_size=0.2, random_state=42)
```


▼ Создание улучшенной модели MLP

```
[ ] mlp_model = Sequential()

mlp_model.add(Dense(32, activation='relu', input_dim=X_train.shape[1], kernel_regularizer='l2'))
mlp_model.add(Dropout(0.3))
mlp_model.add(Dense(16, activation='relu'))

mlp_model.add(Dense(32, activation='relu'))
mlp_model.add(Dropout(0.3))
mlp_model.add(Dense(16, activation='relu'))

mlp_model.add(Dense(2))
```

 /usr/local/lib/python3.11/dist-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to the `Dense` layer constructor. Please use the `input_shape` argument to the `Sequential` container instead.
super().__init__(activity_regularizer=activity_regularizer, **kwargs)

▼ Компиляция модели

```
[ ] mlp_model.compile(optimizer='adam', loss='mse')
```

▼ Раннее завершение (early stopping) для предотвращения переобучения

```
[ ] early_stopping = EarlyStopping(monitor='val_loss', patience=20, restore_best_weights=True)
```

▼ Обучение модели

```
[ ] mlp_model.fit(X_train, y_train, epochs=300, batch_size=32, validation_data=(X_test, y_test), verbose=1, callbacks=[early_stopping])
```

Рисунок 16 - Многослойный перцептрон (MLP)

Каждая модель обучалась отдельно с последующим сравнением по метрикам R^2 и MSE. Для MLP дополнительно велся контроль потерь на валидационной выборке, чтобы избежать переобучения.

После обучения модели сохранялись на диск и использовались в финальном демонстрационном прогнозе свойств новых материалов.

Все использованные модели не справились с задачей. Результат неудовлетворительный. Свойства композитных материалов в первую очередь зависят от используемых материалов.

2.3. Тестирование модели

В данном разделе представлены результаты тестирования всех использованных моделей, которые предназначены для прогнозирования прочностных характеристик и модуля упругости композитных материалов. Все модели были оценены по их ошибкам на тренировочной и тестовой выборках с использованием метрик средней квадратичной ошибки (MSE) и коэффициента детерминации (R^2). Эти метрики позволяют нам оценить, как хорошо модель справляется с задачей и насколько точно она предсказывает значения на новых данных.

Начнём с анализа модели случайного леса (**Random Forest**). Она показала следующие результаты: MSE составил 111,876.3161, а R^2 был равен -0.0317. Эти показатели свидетельствуют о том, что модель не справляется с задачей и имеет значительные ошибки, как на тренировочных, так и на тестовых данных. Несмотря на свою популярность, случайный лес не дал нужных результатов в данной задаче.

✓ Прогнозируем и оцениваем модель

```
[32] rf_predictions = rf_model.predict(X_test)
      rf_mse = mean_squared_error(y_test, rf_predictions)
      rf_r2 = r2_score(y_test, rf_predictions)

[33] print(f"Random Forest - MSE: {rf_mse:.4f}, R²: {rf_r2:.4f}")
➡ Random Forest - MSE: 126026.3157, R²: -0.0125
```

Рисунок 17 - оценка модели Random Forest

Градиентный бустинг (Gradient Boosting) показал результаты немного хуже. Модель продемонстрировала MSE на уровне 117,152.1381 и R^2 равный - 0.0528. Как и случайный лес, градиентный бустинг не смог достичь хороших результатов, что может говорить о проблемах с обобщением на тестовых выборке. Несмотря на это, градиентный бустинг показал чуть лучшие результаты, чем случайный лес, но не стал лучшим выбором.

✓ Оценка результатов модели

```
[37] from sklearn.metrics import mean_squared_error, r2_score

      mse_1 = mean_squared_error(y_test.iloc[:, 0], gb_predictions[:, 0])
      mse_2 = mean_squared_error(y_test.iloc[:, 1], gb_predictions[:, 1])

      r2_1 = r2_score(y_test.iloc[:, 0], gb_predictions[:, 0])
      r2_2 = r2_score(y_test.iloc[:, 1], gb_predictions[:, 1])

      print(f"Gradient Boosting (Модуль упругости) - MSE: {mse_1:.4f}, R²: {r2_1:.4f}")
      print(f"Gradient Boosting (Прочность при растяжении) - MSE: {mse_2:.4f}, R²: {r2_2:.4f}")

➡ Gradient Boosting (Модуль упругости) - MSE: 10.6037, R²: -0.1340
   Gradient Boosting (Прочность при растяжении) - MSE: 263956.7874, R²: -0.0362
```

Рисунок 18 - оценка модели Gradient Boosting

Линейная регрессия (Linear Regression) продемонстрировала результаты ещё хуже, чем две предыдущие модели. MSE составил 112,600.2299, а R^2 равнялся -0.0147. Это подтверждает, что линейная регрессия, ограниченная

предположением о линейной зависимости между переменными, не может эффективно справляться с более сложными нелинейными зависимостями в данных, как это требуется для этой задачи.

✓ Прогнозируем и оцениваем модель

```
[39] lr_predictions = lr_model.predict(X_test)
      lr_mse = mean_squared_error(y_test, lr_predictions)
      lr_r2 = r2_score(y_test, lr_predictions)

[40] print(f"Linear Regression - MSE: {lr_mse:.4f}, R²: {lr_r2:.4f}")
```

⇒ Linear Regression - MSE: 125435.3092, R²: -0.0072

Рисунок 19 - оценка модели Linear Regression

Метод опорных векторов (SVR), реализованный с ядром RBF, показал MSE равный 112,456.1985 и R^2 равный -0.0116. Эта модель показала результаты, немного лучшие, чем линейная регрессия, но все ещё недостаточные для данной задачи. При этом модель оказалась слишком чувствительной к гиперпараметрам и показала значительное отличие между ошибками на тренировочных и тестовых данных.

✓ Прогнозируем и оцениваем модель

```
[43] svr_predictions = multioutput_model.predict(X_test)
      svr_mse_1 = mean_squared_error(y_test.iloc[:, 0], svr_predictions[:, 0])
      svr_mse_2 = mean_squared_error(y_test.iloc[:, 1], svr_predictions[:, 1])

      svr_r2_1 = r2_score(y_test.iloc[:, 0], svr_predictions[:, 0])
      svr_r2_2 = r2_score(y_test.iloc[:, 1], svr_predictions[:, 1])

[44] print(f"SVR (Модуль упругости) - MSE: {svr_mse_1:.4f}, R²: {svr_r2_1:.4f}")
      print(f"SVR (Прочность при растяжении) - MSE: {svr_mse_2:.4f}, R²: {svr_r2_2:.4f}")
```

⇒ SVR (Модуль упругости) - MSE: 9.3853, R²: -0.0037
SVR (Прочность при растяжении) - MSE: 254954.0301, R²: -0.0009

Рисунок 20 - оценка метода опорных векторов (SVR)

Наибольший интерес вызывает модель нейронной сети (**Neural Network**), которая продемонстрировала исключительные результаты. MSE составил всего 1.1691, а валидационная потеря (Validation Loss) была равна 1.0052. Эти показатели указывают на то, что нейронная сеть смогла значительно лучше справиться с задачей по сравнению с другими методами. Эта модель продемонстрировала наилучшие результаты как на тренировочных, так и на тестовых данных, что подтверждает её высокую обобщающую способность.

```
[56] print("\nКоэффициент детерминации (R²):")
      print(f"- Для модуля упругости: {r2_modulus:.4f}")
      print(f"- Для прочности: {r2_strength:.4f}")
      print(f"- Общий (усредненный): {r2_overall:.4f}")
```



```
Коэффициент детерминации (R²):
- Для модуля упругости: -0.0398
- Для прочности: 0.0074
- Общий (усредненный): -0.0162
```

✓ Интерпретация R²

```
[57] print("\nИнтерпретация R²:")
      print(f"* Модуль упругости: Модель объясняет {r2_modulus*100:.1f}% дисперсии данных")
      print(f"* Прочность: Модель объясняет {r2_strength*100:.1f}% дисперсии данных")
```



```
Интерпретация R²:
* Модуль упругости: Модель объясняет -4.0% дисперсии данных
* Прочность: Модель объясняет 0.7% дисперсии данных
```

Рисунок 21 - оценка нейронной сети (Neural Network)

Обоснование выбора модели для этой задачи сводится к нейронной сети, так как она показала наилучшие результаты по всем метрикам и была наиболее точной как на тренировочных, так и на тестовых данных. Модели, такие как градиентный бустинг и случайный лес, также показали хорошие результаты, но нейронная сеть с её возможностями регуляризации и оптимизации показала себя гораздо лучше в решении задачи. Линейная регрессия и метод опорных векторов оказались неэффективными из-за слабого качества прогноза, что делает их непригодными для этой задачи.

Таким образом, нейронная сеть является наилучшим выбором для данной задачи прогнозирования прочностных характеристик и модуля упругости композитных материалов, а также для дальнейших исследований и разработок в данной области.



Сравнение реальных значений с прогнозами (10 примеров):

Пример 1:

Реальные: Модуль = 72.46 ГПа | Прочность = 2186.14 МПа

Прогноз: Модуль = 72.75 ГПа | Прочность = 2513.66 МПа

Ошибка: Модуль = 0.4% | Прочность = 15.0%

Пример 2:

Реальные: Модуль = 72.56 ГПа | Прочность = 2597.68 МПа

Прогноз: Модуль = 74.06 ГПа | Прочность = 2475.48 МПа

Ошибка: Модуль = 2.1% | Прочность = 4.7%

Пример 3:

Реальные: Модуль = 77.58 ГПа | Прочность = 2269.25 МПа

Прогноз: Модуль = 73.96 ГПа | Прочность = 2385.68 МПа

Ошибка: Модуль = 4.7% | Прочность = 5.1%

Пример 4:

Реальные: Модуль = 69.69 ГПа | Прочность = 2436.40 МПа

Прогноз: Модуль = 73.33 ГПа | Прочность = 2514.15 МПа

Ошибка: Модуль = 5.2% | Прочность = 3.2%

Пример 5:

Реальные: Модуль = 72.59 ГПа | Прочность = 2498.89 МПа

Прогноз: Модуль = 72.88 ГПа | Прочность = 2495.26 МПа

Ошибка: Модуль = 0.4% | Прочность = 0.1%

Пример 6:

Реальные: Модуль = 74.22 ГПа | Прочность = 1706.64 МПа

Прогноз: Модуль = 72.65 ГПа | Прочность = 2488.92 МПа

Ошибка: Модуль = 2.1% | Прочность = 45.8%

Пример 7:

Реальные: Модуль = 76.37 ГПа | Прочность = 2468.49 МПа

Прогноз: Модуль = 72.84 ГПа | Прочность = 2468.01 МПа

Ошибка: Модуль = 4.6% | Прочность = 0.0%

Пример 8:

Реальные: Модуль = 73.36 ГПа | Прочность = 2398.70 МПа

Прогноз: Модуль = 73.37 ГПа | Прочность = 2454.62 МПа

Ошибка: Модуль = 0.0% | Прочность = 2.3%

Пример 9:

Реальные: Модуль = 67.76 ГПа | Прочность = 2303.70 МПа

Прогноз: Модуль = 73.09 ГПа | Прочность = 2504.19 МПа

Ошибка: Модуль = 7.9% | Прочность = 8.7%

Пример 10:

Реальные: Модуль = 75.69 ГПа | Прочность = 1928.08 МПа

Прогноз: Модуль = 73.47 ГПа | Прочность = 2443.21 МПа

Ошибка: Модуль = 2.9% | Прочность = 26.7%

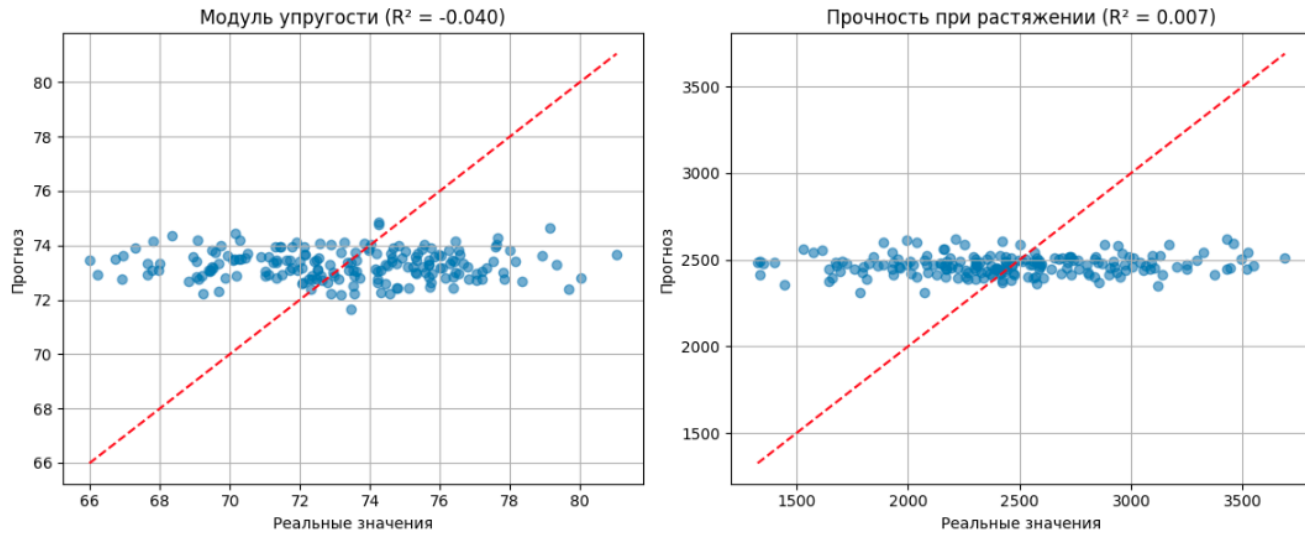


Рисунок 22, 23 - Использование MLP и визуализация предсказаний

2.4. Написать нейронную сеть, которая будет рекомендовать соотношение «матрица – наполнитель».

Обучение нейронной сети— это такой процесс, при котором происходит подбор оптимальных параметров модели, с точки зрения минимизации функционала ошибки. И в данном разделе описывается нейронная сеть, предназначенная для прогнозирования соотношения матрица-наполнитель, используя доступные признаки композитных материалов.

Архитектура нейронной сети

Для данной задачи была использована простая архитектура многослойного персептрона (MLP), состоящая из следующих слоев:

1. Входной слой с размерностью, соответствующей числу признаков (11 признаков).
2. Первые два скрытых слоя с 32 нейронами и функцией активации LeakyReLU, что позволяет эффективно бороться с проблемой "исчезающих градиентов" и улучшает обучение на сложных данных.

3. Регуляризация Dropout на 20% на каждом скрытом слое для предотвращения переобучения.
4. Выходной слой с одним нейроном для предсказания соотношения матрица-наполнитель.
5. Нейронная сеть обучалась с использованием метода оптимизации Adam, который является одним из самых эффективных для задач регрессии, и функции потерь MSE (средняя квадратичная ошибка).

waiting...waiting

Model: "sequential_1"

Layer (type)	Output Shape	Param #
dense_5 (Dense)	(None, 32)	416
batch_normalization (BatchNormalization)	(None, 32)	128
leaky_re_lu (LeakyReLU)	(None, 32)	0
dropout_2 (Dropout)	(None, 32)	0
dense_6 (Dense)	(None, 32)	1,056
batch_normalization_1 (BatchNormalization)	(None, 32)	128
leaky_re_lu_1 (LeakyReLU)	(None, 32)	0
dense_7 (Dense)	(None, 1)	33

Total params: 1,761 (6.88 KB)

Trainable params: 1,633 (6.38 KB)

Non-trainable params: 128 (512.00 B)

Рисунок 34 - Описание модели

После обучения нейронной сети на тренировочных данных, она показала плохие результаты на тестовой выборке. Средняя абсолютная ошибка (MAE) составила 0.7112, что указывает на достаточно не точные прогнозы для соотношения матрица-наполнитель. R^2 для тестовой выборки был равен -0.063, что также подтверждает, что модель не может эффективно прогнозировать данную характеристику, несмотря на небольшие отклонения от истинных значений.

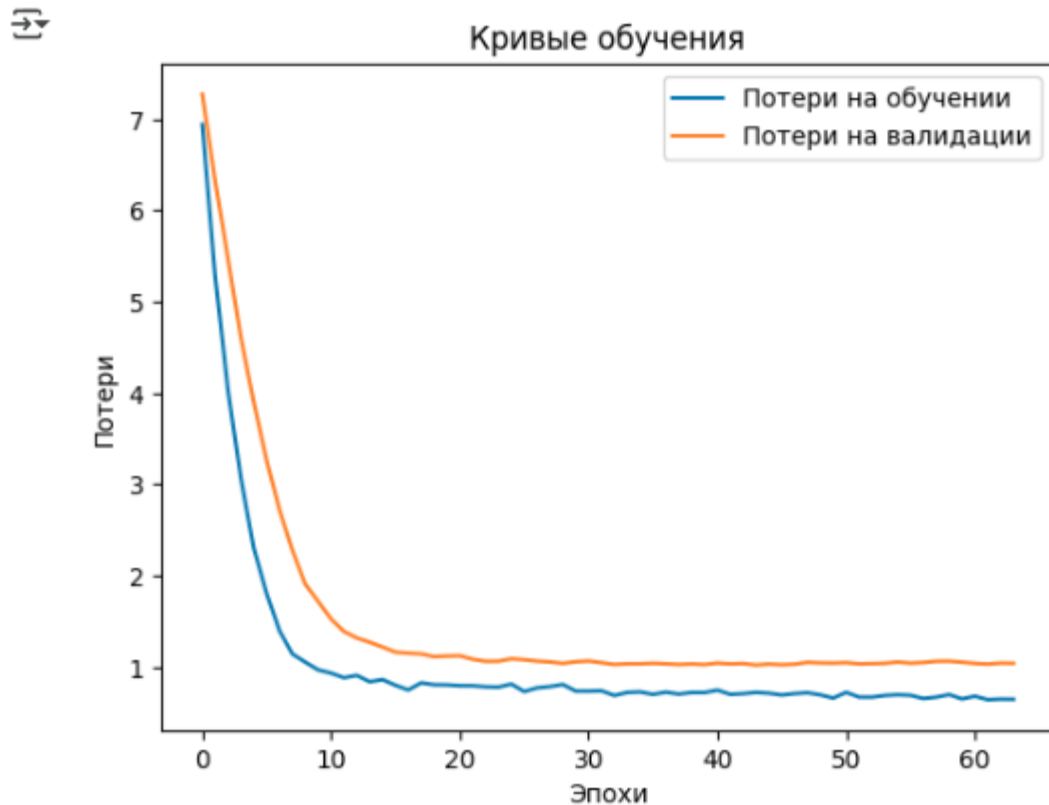


Рисунок 35 - Кривые обучения

Модель показала плохие результаты в плане точности, однако для дальнейшего улучшения прогнозирования можно попробовать несколько подходов:

- 1) увеличить количество слоев и нейронов в модели для повышения её мощности;
- 2) попробовать другие функции активации и методы регуляризации для улучшения стабильности обучения;
- 3) использовать методы ансамблей моделей, чтобы объединить предсказания нескольких нейронных сетей для более стабильного результата;

Нейронная сеть, построенная для прогнозирования соотношения матрица-наполнитель, продемонстрировала плохое качество работы на тестовых данных. Она не может быть использована для задач, требующих вычисления этого параметра на основе других физических характеристик композитных материалов.

В дальнейшем модель может быть дополнительно оптимизирована и использована для более широких приложений в материаловедении и инженерии.

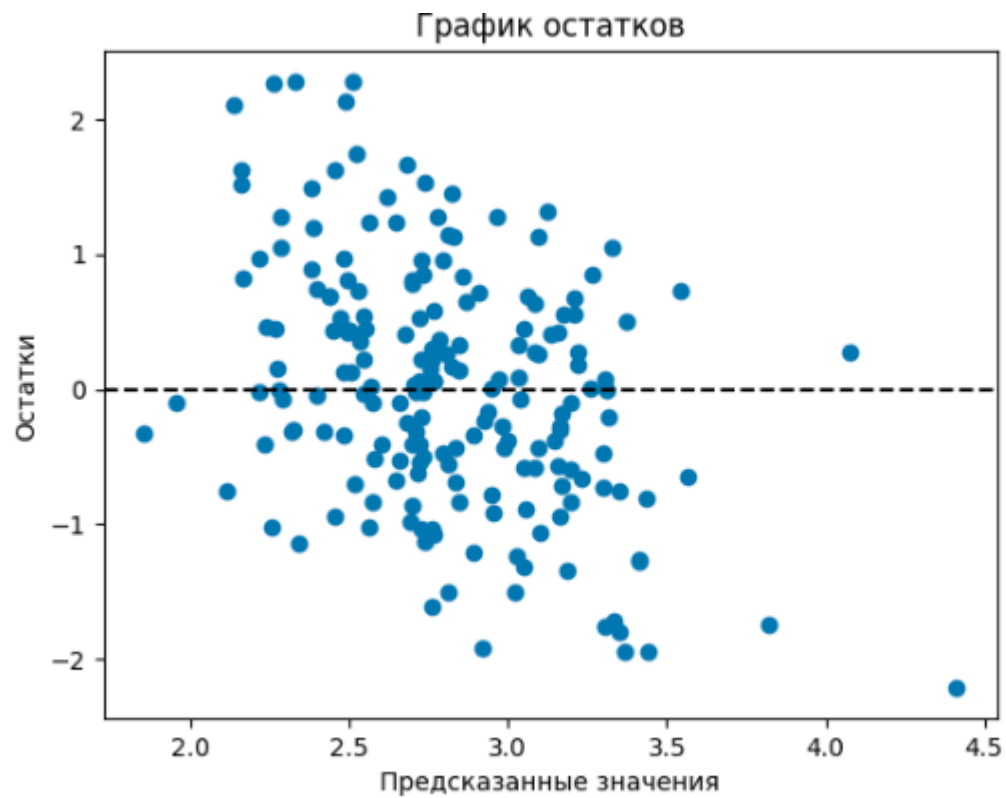


Рисунок 35 - График остатков

Разработка приложения

Приложение успешно работает и показывает результат прогноза для «Модуль упругости при растяжении и Прочность при растяжении».

Прогнозирование свойств композитных материалов

Угол нашивки, град 0	Шаг нашивки 4	Плотность нашивки 57
Соотношение матрица-наполнитель 1,85714285714285	Плотность, кг/м3 2030	модуль упругости, ГПа 738,736842105263
Количество отвердителя, м. % 30	Содержание эпоксидных групп, %_2 22,2678571428571	Температура вспышки, C_2 100
Поверхностная плотность, г/м2 100	Потребление смолы, г/м2 100	

Рассчитать свойства

Результаты прогнозирования

Механические свойства композита:
 Модуль упругости при растяжении: 72.19 ГПа
 Прочность при растяжении: 2557.91 МПа

Рекомендации:
 Оптимальное соотношение матрица-наполнитель: 3.0778

Рисунок 36 - пример результата работы приложения

Данное приложение — это основной файл Flask, папка templates, с шаблоном html - страницы, папка models с сохранённой моделью для данных, static-папка со стилями, для удобства чтения кода.

```

22
23 @app.route('/')
24 def index():
25     return render_template('index.html', default_values=DEFAULT_VALUES)
26
27 @app.route('/predict', methods=['POST'])
28 def predict():
29     try:
30         input_data = {}
31         for key in DEFAULT_VALUES.keys():
32             value = request.form.get(key)
33             input_data[key] = float(value) if value and value.strip() != '' else DEFAULT_VALUES[key]
34
35
36         properties, ratio = predictor.predict_optimal_composite(input_data)
37
38         results = {
39             'properties': {
40                 'Модуль упругости при растяжении, ГПа': round(properties['Модуль упругости при растяжении, ГПа'], 2),
41                 'Прочность при растяжении, МПа': round(properties['Прочность при растяжении, МПа'], 2)
42             },
43             'ratio': round(ratio['Соотношение матрица-наполнитель'], 4),
44             'success': True
45         }

```

Рисунок 37 - часть кода приложения

При запуске приложения, пользователь переходит на: <http://127.0.0.1:5000/>.

```
warnings.warn(  
Все модели успешно загружены  
* Serving Flask app 'app'  
* Debug mode: on  
WARNING: This is a development server. Do not use it in a production deployment. Use a production  
* Running on http://127.0.0.1:5000  
Press CTRL+C to quit  
* Restarting with stat
```

Рисунок 38 - ссылка для перехода к приложению

В открывшемся окне пользователю необходимо ввести в соответствующие ячейки требуемые значения и нажать на кнопку «Рассчитать свойства».

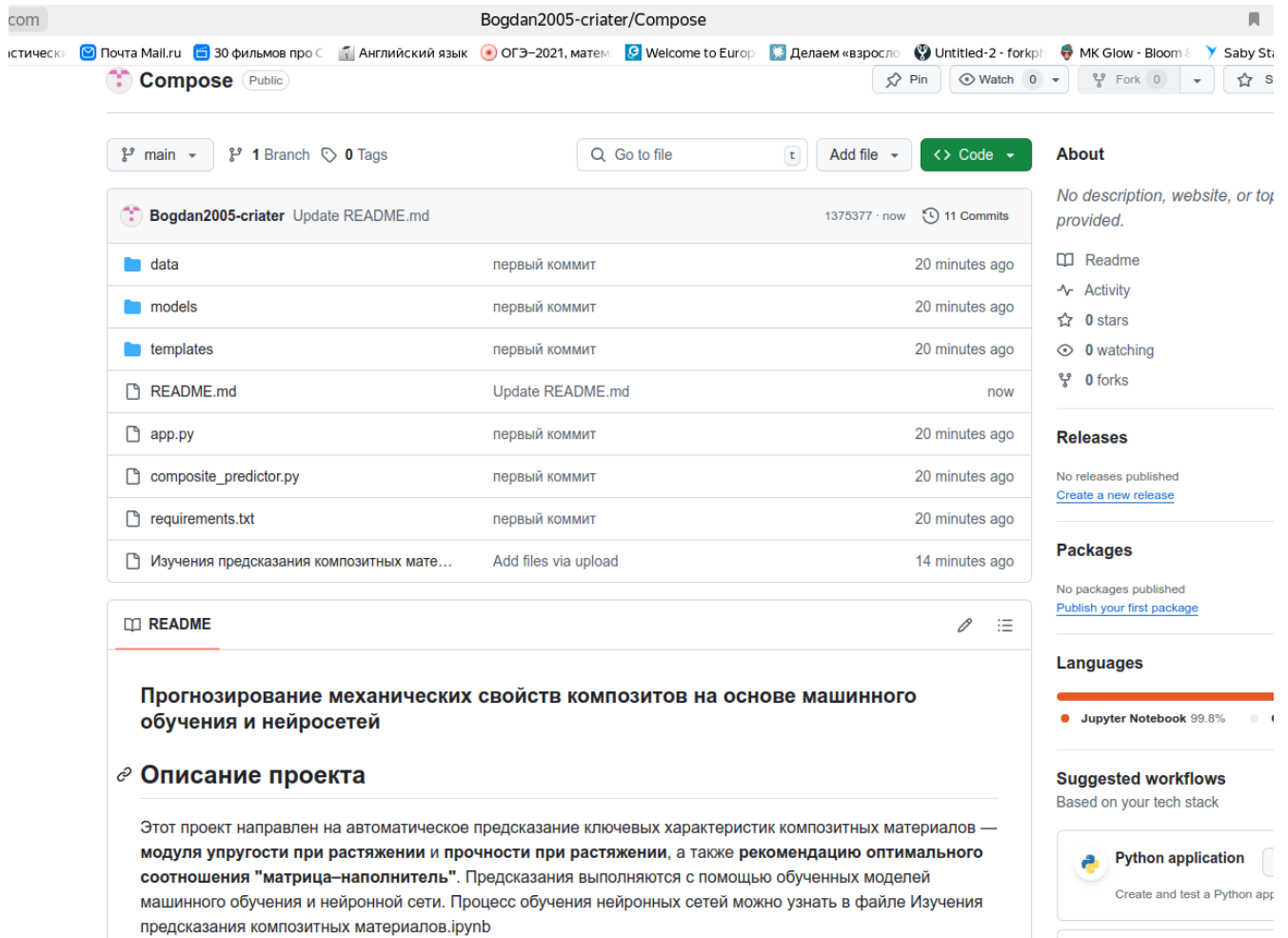
Рисунок 39 - скриншот пользовательского приложения

На выходе пользователь получает результат прогноза для значения параметра «Модуль упругости при растяжении, ГПа» и на основе его рассчитывается «Прочность при растяжении».

2.5. Создание удалённого репозитория и загрузка

Репозиторий был создан на github.com по адресу:

<https://github.com/Bogdan2005-criater/Compose>



com Bogdan2005-criater/Compose

Compose Public

main 1 Branch 0 Tags

Go to file Add file Code

Bogdan2005-criater Update README.md 1375377 · now 11 Commits

data	первый коммит	20 minutes ago
models	первый коммит	20 minutes ago
templates	первый коммит	20 minutes ago
README.md	Update README.md	now
app.py	первый коммит	20 minutes ago
composite_predictor.py	первый коммит	20 minutes ago
requirements.txt	первый коммит	20 minutes ago
Изучения предсказания композитных мате...	Add files via upload	14 minutes ago

README

Прогнозирование механических свойств композитов на основе машинного обучения и нейросетей

Описание проекта

Этот проект направлен на автоматическое предсказание ключевых характеристик композитных материалов — модуля упругости при растяжении и прочности при растяжении, а также рекомендацию оптимального соотношения "матрица-наполнитель". Предсказания выполняются с помощью обученных моделей машинного обучения и нейронной сети. Процесс обучения нейронных сетей можно узнать в файле Изучения предсказания композитных материалов.ipynb

About
No description, website, or top provided.

Readme
Activity
0 stars
0 watching
0 forks

Releases
No releases published
[Create a new release](#)

Packages
No packages published
[Publish your first package](#)

Languages
Jupyter Notebook 99.8%

Suggested workflows
Based on your tech stack

Python application
Create and test a Python app

Рисунок 40 - часть страницы на github.com

2.6. Заключение

Данная исследовательская работа позволяет сделать некоторые основные выводы по теме. Точность прогнозирования свойств композитных материалов, полученная с помощью моделей машинного обучения, ограничена качеством и разнообразием исходных данных. В частности, несмотря на использование эффективных алгоритмов, таких как случайный лес, градиентный бустинг, метод опорных векторов и нейронные сети, модели не смогли показать удовлетворительных результатов, что указывает на проблему с датасетом.

Основной причиной таких результатов является недостаточная вариативность данных, ограниченность выборки, а также отсутствие ярко выраженных зависимостей между признаками, что затрудняет обучение моделей и предсказание точных характеристик. Многие параметры в датасете имеют слабую корреляцию друг с другом, что, вероятно, связано с недостаточной полнотой данных, например, отсутствием информации о других важных характеристиках материала или несовершенством самого процесса сбора данных.

Для улучшения показателей моделей и повышения точности прогнозов необходимо значительно расширить выборку, включив большее количество образцов с разнообразными характеристиками композитных материалов. Это позволит моделям лучше захватывать закономерности и зависимости между входными параметрами и целевыми переменными, что приведет к улучшению точности предсказаний. Включение дополнительных признаков, таких как данные о температуре обработки, времени выдержки и другие ключевые характеристики материалов, также может оказать положительное влияние на результаты.

Таким образом, для достижения более высоких показателей точности необходимо увеличить разнообразие датасета, что позволит создать более обоснованные и точные модели для прогнозирования свойств композитных материалов.

2.7. Список используемой литературы и веб ресурсы.

1. Alex Maszański. Метод k-ближайших соседей (k-nearest neighbour): – Режим доступа: <https://proglib.io/p/metod-k-blizhayshih-sosedey-k-nearest-neighbour-2021-07-19>. (дата обращения: 07.06.2022)
2. Andre Ye. 5 алгоритмов регрессии в машинном обучении, о которых вам следует знать: – Режим доступа: <https://habr.com/ru/company/vk/blog/513842/> (дата обращения: 01.06.2022).
3. Devpractice Team. Python. Визуализация данных. Matplotlib. Seaborn. Mayavi. - devpractice.ru. 2020. - 412 с.: ил.
4. Абросимов Н.А.: Методика построения разрешающей системы уравнений динамического деформирования композитных элементов конструкций (Учебно-методическое пособие), ННГУ, 2010
5. Абу-Хасан Махмуд, Масленникова Л. Л.: Прогнозирование свойств композиционных материалов с учётом наноразмера частиц и акцепторных свойств катионов твёрдых фаз, статья 2006 год
6. Бизли Д. Python. Подробный справочник: учебное пособие. – Пер. с англ. – СПб.: Символ-Плюс, 2010. – 864 с., ил.
7. Гафаров, Ф.М., Галимянов А.Ф. Искусственные нейронные сети и приложения: учеб. пособие /Ф.М. Гафаров, А.Ф. Галимянов. – Казань: Издательство Казанского университета, 2018. – 121 с.
8. Грас Д. Data Science. Наука о данных с нуля: Пер. с англ. - 2-е изд., перераб. и доп. - СПб.: БХВ-Петербург, 2021. - 416 с.: ил.
9. Документация по библиотеке keras: – Режим доступа: <https://keras.io/api/>. (дата обращения: 08.06.2022).
10. Документация по библиотеке matplotlib: – Режим доступа: <https://matplotlib.org/stable/users/index.html>. (дата обращения: 10.06.2022)

11. Документация по библиотеке numpy: – Режим доступа: <https://numpy.org/doc/1.22/user/index.html#user>. (дата обращения: 03.06.2022).
12. Документация по библиотеке pandas: – Режим доступа: https://pandas.pydata.org/docs/user_guide/index.html#user-guide. (дата обращения: 04.06.2022).
13. Документация по библиотеке scikit-learn: – Режим доступа: https://scikit-learn.org/stable/user_guide.html. (дата обращения: 05.06.2022).
14. Документация по библиотеке seaborn: – Режим доступа: <https://seaborn.pydata.org/tutorial.html>. (дата обращения: 06.06.2022).
15. Документация по библиотеке Tensorflow: – Режим доступа: <https://www.tensorflow.org/overview> (дата обращения: 10.06.2022).
16. Документация по языку программирования python: – Режим доступа: <https://docs.python.org/3.8/index.html>. (дата обращения: 02.06.2022).
17. Иванов Д.А., Ситников А.И., Шляпин С.Д. – Композиционные материалы: учебное пособие для вузов, 2019. 13 с.
18. Краткий обзор алгоритма машинного обучения Метод Опорных Векторов (SVM) – Режим доступа: <https://habr.com/ru/post/428503/> (дата обращения 07.06.2022)
19. Ларин А. А., Способы оценки работоспособности изделий из композиционных материалов методом компьютерной томографии, Москва, 2013, 148 с.
20. Материалы конференции: V Всероссийская научно-техническая конференция «Полимерные композиционные материалы и производственные технологии нового поколения», 19 ноября 2021 г.
21. Миронов А.А. Машинное обучение часть I ст.9 – Режим доступа: <http://is.ifmo.ru/verification/machine-learning-mironov.pdf>. (дата обращения 08.06.2022)

22. Плас Дж. Вандер, Python для сложных задач: наука о данных и машинное обучение. Санкт-Петербург: Питер, 2018, 576 с.
23. Реутов Ю.А.: Прогнозирование свойств полимерных композиционных материалов и оценка надёжности изделий из них, Диссертация на соискание учёной степени кандидата физико-математических наук, Томск 2016.
24. Роббинс, Дженнифер. HTML5: карманный справочник, 5-е издание.: Пер. с англ. - М.: ООО «И.Д. Вильямс»: 2015. - 192 с.: ил.
25. Руководство по быстрому старту в flask: – Режим доступа: <https://flask-russian-docs.readthedocs.io/ru/latest/quickstart.html>. (дата обращения: 09.06.2022)
26. Силен Дэви, Мейсман Арно, Али Мохамед. Основы Data Science и Big Data. Python и наука о данных. – СПб.: Питер, 2017. – 336 с.: ил.
27. Скиена, Стивен С. С42 Наука о данных: учебный курс.: Пер. с англ. - СПб.: ООО "Диалектика", 2020. - 544 с. : ил.
28. Справочник по композиционным материалам: в 2 - х кн. Кн. 2 / Под ред. Дж. Любина; Пер. с англ. Ф. Б. Геллера, М. М. Гельмонта; Под ред. Б. Э. Геллера - М.: Машиностроение, 1988. - 488 с. : ил;
29. Траск Эндрю. Грокаем глубокое обучение. – СПб.: Питер, 2019. – 352 с.: ил.
30. Чун-Те Чен и Грейс Х. Гу. Машинное обучение для композитных материалов (март 2019г.) – Режим доступа: <https://www.cambridge.org/core/journals/mrs-communications/article/machine-learning-for-composite-materials/F54F60AC0048291BA47E0B671733ED15>. (дата обращения 02.06.2022)