

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

Москва, 2023

Содержание

Введение.....	3
1. Аналитическая часть.....	4
1.1. Постановка задачи.....	4
1.2. Описание используемых методов.....	7
1.3. Разведочный анализ данных.....	10
2. Практическая часть.....	20
2.1. Предобработка данных.....	20
2.2. Разработка и обучение модели.....	24
2.3. Тестирование модели.....	29
2.4. Нейронная сеть.....	31
2.5. Разработка приложения.....	37
2.6. Создание репозитория.....	40
Заключение.....	41
3. Список используемой литературы.....	42

Введение

Композиционные материалы – это искусственно созданные материалы, состоящие из нескольких других с четкой границей между ними. Композиты обладают теми свойствами, которые не наблюдаются у компонентов по отдельности. При этом композиты являются монолитным материалом, т.е. компоненты материала неотделимы друг от друга без разрушения конструкции в целом. Яркий пример композита – железобетон. Бетон прекрасно сопротивляется сжатию, но плохо изгибу и растяжению. Стальная арматура внутри бетона компенсирует его неспособность сопротивляться изгибу, формируя тем самым новые, уникальные свойства. Современные композиты изготавливаются из других материалов: полимеры, керамика, стеклянные и углеродные волокна, но данный принцип сохраняется. У такого подхода есть и недостаток: даже зная характеристики исходных компонентов, определить характеристики композита, состоящего из этих компонентов, достаточно проблематично. Для решения этой проблемы есть два пути: физические испытания образцов материалов, или прогнозирование характеристик. Суть прогнозирования заключается в симуляции представительного элемента объема композита, на основе данных о характеристиках входящих компонентов (связующего и армирующего компонента).

1. Аналитическая часть

1.1 Постановка задачи

На входе имеются данные о начальных свойствах компонентов композиционных материалов (плотность, количество отвердителя, содержание эпоксидных групп и т.д.). На выходе необходимо спрогнозировать ряд конечных свойств получаемых композиционных материалов, а именно:

- модуль упругости при растяжении, ГПа;
- прочность при растяжении, МПа.

Созданные прогнозные модели помогут сократить количество проводимых испытаний, а также пополнить базу данных материалов возможными новыми характеристиками материалов, и цифровыми двойниками новых композитов.

Также необходимо написать нейронную сеть, которая будет рекомендовать соотношение матрица-наполнитель.

Данные для решения поставленной задачи представлены в виде двух датасетов, представленных в виде файлов с именами X_bp.xlsx и X_nup.xlsx формата Microsoft Excel. Датасеты имеют различное количество строк, а именно: 1023 и 1040 соответственно. Так же различаются входные переменные (Рисунок 1, 2).

```
# Отобразим первые 5 строк датасета
data_bp.head()
```

Соотношение матрица-наполнитель	Плотность, кг/м3	модуль упругости, ГПа	Количество отвердителя, м.%	Содержание эпоксидных групп, %_2	Температура вспышки, С_2	Поверхностная плотность, г/м2	Модуль упругости при растяжении, ГПа	Прочность при растяжении, МПа	Потребление смолы, г/м2	
0	1.857143	2030.0	738.736842	30.00	22.267857	100.000000	210.0	70.0	3000.0	220.0
1	1.857143	2030.0	738.736842	50.00	23.750000	284.615385	210.0	70.0	3000.0	220.0
2	1.857143	2030.0	738.736842	49.90	33.000000	284.615385	210.0	70.0	3000.0	220.0
3	1.857143	2030.0	738.736842	129.00	21.250000	300.000000	210.0	70.0	3000.0	220.0
4	2.771331	2030.0	753.000000	111.86	22.267857	284.615385	210.0	70.0	3000.0	220.0

```
# Смотрим количество строк и столбцов (переменных) в датасете 'X_bp'
data_bp.shape
```

(1023, 10)

Рисунок 1 – Обзор датасета «X_bp.xlsx»

```
# Отобразим первые 5 строк датасета  
data_nup.head()
```

	Угол нашивки, град	Шаг нашивки	Плотность нашивки
0	0	4.0	57.0
1	0	4.0	60.0
2	0	4.0	70.0
3	0	5.0	47.0
4	0	5.0	57.0

```
# Смотрим количество строк и столбцов (переменных) в датасете 'X_nup'  
data_nup.shape
```

(1040, 3)

Рисунок 2 – Обзор датасета «X_nup.xlsx»

Для последующего решения задачи производим объединение датасетов X_bp.xlsx и X_nup.xlsx в один по “индексу” с помощью метода .merge, тип объединения INNER.

Получаем датасет имеющий следующие входные переменные (факторы):

- 1) соотношение матрица-наполнитель;
- 2) плотность, кг/м³;
- 3) модуль упругости, Гпа;
- 4) количество отвердителя, м.%;
- 5) содержание эпоксидных групп, %_2;
- 6) температура вспышки, С_2;
- 7) поверхностная плотность, г/м²;
- 8) модуль упругости при растяжении, Гпа;
- 9) прочность при растяжении, Мпа;
- 10) потреблении смолы, г/м²;
- 11) угол нашивки, град;
- 12) шаг нашивки;
- 13) плотность нашивки.

В зависимости от решаемой задачи выходными переменными становятся:

- модуль упругости при растяжении, Гпа;
- прочность при растяжении, Мпа;
- соотношение матрица-наполнитель.

При этом, становясь выходной переменной, соответствующая переменная исключается из входных данных.

1.2 Описание используемых методов

Для решения поставленной задачи применим методы машинного обучения.

Машинное обучение (англ. *machine learning*, ML) – класс методов искусственного интеллекта, характерной чертой которых является не прямое решение задачи, а обучение за счёт применения решений множества сходных задач. Для построения таких методов используются средства математической статистики, численных методов, математического анализа, методов оптимизации, теории вероятностей, теории графов, различные техники работы с данными в цифровой форме.

Решение задачи машинного обучения состоит из следующих основных этапов: формулирование проблемы, сбор и предобработка данных, разработка и построение модели машинного обучения, тестирование, оценка качества, постобработка результатов, внедрение эксплуатацию

Прогнозирование конечных свойств композиционных материалов является задачей регрессии в машинном обучении.

Задача регрессии в машинном обучении – это предсказание одного параметра (Y) по известному параметру X , где X – набор параметров, характеризующий наблюдение.

Рассмотрим некоторые наиболее распространенные методы машинного обучения для решения задач регрессии.

Метод линейной регрессии (LinearRegression) – это модель линейной регрессии, которая оценивает взаимосвязь между несколькими независимыми переменными (признаками) и одной зависимой переменной.

Используемая в статистике регрессионная модель зависимости одной (объясняемой, зависимой) переменной «у» от другой или нескольких других

переменных (факторов, регрессоров, независимых переменных) « x » с линейной функцией зависимости.

Линейная регрессия использует метод, известный как обычный метод наименьших квадратов, чтобы найти наиболее подходящее уравнение регрессии.

Преимущества:

- хорошо изучен, быстр и прост в реализации;
- легко объясним и интерпретируем;
- может работать на очень больших выборках.

Недостатки:

- моделирует только прямые линейные зависимости;
- плохо работает в задачах, в которых зависимость ответов от признаков сложная, нелинейная;
- выбросы оказывают огромное влияние.

Метод решающих деревьев (DecisionTreeRegressor) – также называют деревом классификации или регрессионным деревом. Структура дерева представляет собой «листья» и «ветки». На рёбрах («ветках») дерева решения записаны признаки, от которых зависит целевая функция, в «листьях» записаны значения целевой функции, а в остальных узлах – признаки, по которым различаются случаи. Чтобы классифицировать новый случай, надо спуститься по дереву до листа и выдать соответствующее значение. Цель состоит в том, чтобы создать модель, которая предсказывает значение целевой переменной на основе нескольких переменных на входе. Каждый лист представляет собой значение целевой переменной, изменённой в ходе движения от корня по рёбрам дерева до листа. Каждый внутренний узел сопоставляется с одной из входных переменных. Алгоритм вычисляет информационный прирост для каждой характеристики и выбирает ту, которая дает наивысшее значение.

Преимущества:

- прост для понимания и интерпретации, деревья можно визуализировать;

- не требует специальной подготовки данных (нормализация, добавления фиктивных переменных, удаления пропущенных данных);
- способен обрабатывать как числовые, так и категориальные данные (другие методы обычно специализируются на анализе наборов данных, которые имеют только один тип переменной);
- позволяет оценить модель при помощи статистических тестов, это даёт возможность оценить надёжность модели;
- хорошо работает даже в том случае, если были нарушены первоначальные предположения, включённые в модель;
- использует модель белого ящика. Если данная ситуация наблюдаема в модели, объяснение условия легко объясняется булевой логикой. В отличие от этого, в модели чёрного ящика (например, в искусственной нейронной сети) результаты могут быть более сложными для интерпретации.

Недостатки:

- в процессе построения дерева решений могут создаваться слишком сложные конструкции (деревья), которые недостаточно полно представляют данные. Данную проблему называют переобучением. Для того, чтобы её избежать, необходимо использовать метод «регулирования глубины дерева»;
- есть концепции, которые трудно изучить, потому что деревья решений не выражают их легко, например, XOR, проблемы с четностью или мультиплексором;
- предсказания деревьев решений не являются ни гладкими, ни непрерывными, а кусочно-постоянными приближениями. Поэтому они не очень хороши для экстраполяции;
- деревья решений могут быть нестабильными, поскольку небольшие отклонения в данных могут привести к созданию совершенно другого дерева. Эта проблема устраняется путем использования деревьев решений в ансамбле;
- проблема обучения оптимального дерева решений является NP-полной при нескольких аспектах оптимальности и даже для простых концепций. Следовательно, практические алгоритмы обучения дерева решений основаны

на эвристических алгоритмах, таких как жадный алгоритм, где локально оптимальные решения принимаются на каждом узле. Такие алгоритмы не могут гарантировать возврат глобально оптимального дерева решений. Это может быть смягчено путем обучения нескольких деревьев в обучаемом ансамбле, где функции и образцы выбираются случайнным образом с заменой.

Метод случайный лес (Random Forest) – алгоритм машинного обучения, предложенный Лео Брейманом и Адель Катлер, заключающийся в использовании комитета (ансамбля) решающих деревьев. Алгоритм сочетает в себе две основные идеи: метод бэггинга Бреймана, и метод случайных подпространств, предложенный Тин Кам Хо. Алгоритм применяется для задач классификации, регрессии и кластеризации. Основная идея заключается в использовании большого ансамбля решающих деревьев, каждое из которых само по себе даёт очень невысокое качество классификации, но за счёт их большого количества результат получается хорошим.

Преимущества:

- имеет высокую точность предсказания, не требует тщательной настройки параметров, хорошо работает со значениями по умолчанию;
- способен эффективно обрабатывать данные с большим числом признаков и классов;
- одинаково хорошо обрабатываются как непрерывные, так и дискретные признаки. Существуют методы построения деревьев по данным с пропущенными значениями признаков;
- нечувствительность к масштабированию (в том числе к любым монотонным преобразованиям) значений признаков;
- высокая параллелизуемость и масштабируемость.

Недостатки:

- требуется значительный объем вычислительных ресурсов, большой размер получающихся моделей.

Метод К ближайших соседей (KneighborsRegressor) – алгоритм классификации, который используется в разных типах задач машинного обучения. На интуитивном уровне суть метода проста: посмотри на соседей вокруг, какие из них преобладают, таковыми ты и являешься.

В случае использования метода для регрессии, объекту присваивается среднее значение по k ближайшим к нему объектам, значения которых уже известны. Регрессия на основе соседей может использоваться в тех случаях, когда метки данных являются непрерывными, а не дискретными переменными. Метка, присвоенная точке запроса, вычисляется на основе среднего значения меток ее ближайших соседей. Базовая регрессия ближайших соседей использует одинаковые веса: то есть каждая точка в локальной окрестности вносит одинаковый вклад в классификацию точки запроса. При некоторых обстоятельствах может быть выгодно взвешивать точки таким образом, чтобы близлежащие точки вносили больший вклад в регрессию, чем удаленные точки.

Несмотря на свою простоту, метод К ближайших соседей успешно решает большое количество задач классификации и регрессии, включая рукописные цифры и сцены спутниковых изображений. Будучи непараметрическим методом, он часто оказывается успешным в ситуациях классификации, когда граница принятия решения очень нерегулярна.

Преимущества:

- алгоритм прост и легко реализуем;
- универсален, можно использовать для обоих типов задач: классификации и регрессии;
- не чувствителен к выбросам;

Недостатки:

- алгоритм работает медленнее при увеличении объема выборки, предикторов или независимых переменных
- требуется значительный объем вычислительных ресурсов;
- обязательно нужно определять оптимальное значение k .

Метод опорных векторов (SVR) англ. support vector machine – один из наиболее популярных методов обучения, который применяется для решения задач классификации и регрессии. Основная идея метода заключается в построении гиперплоскости, разделяющей объекты выборки оптимальным способом. Алгоритм работает в предположении, что чем больше расстояние (зазор) между разделяющей гиперплоскостью и объектами разделяемых классов, тем меньше будет средняя ошибка классификатора

Регрессия опорных векторов – это контролируемая модель обучения, которую можно использовать для выполнения как линейной, так и нелинейной регрессии. Цель применения линейной регрессии – минимизировать ошибку между прогнозом и данными. Однако цель применения регрессии опорных векторов к набору данных – убедиться, что ошибки не превышают пороговое значение. В SVR мы помещаем как можно больше экземпляров между строками, ограничивая при этом нарушение полей.

Сложность подбора времени более чем квадратична с количеством выборок, что затрудняет масштабирование до наборов данных, содержащих более пары 10000 выборок.

Преимущества:

- эффективен в пространствах с высокой размерностью;
- эффективен в случаях, когда количество измерений больше, чем количество выборок;
- использует подмножество обучающих точек в функции принятия решений (называемых опорными векторами), поэтому он также экономит память;
- универсальность: для функции принятия решения могут быть указаны различные функции ядра. Предоставляются общие ядра, но также возможно указать пользовательские ядра;
- хорошо адаптируется;
- хорошо работает для нелинейных задач;
- отсутствие предвзятости к объекту, отличающемуся от других.

Недостатки:

- если количество функций намного больше, чем количество выборок чувствителен к чрезмерной подгонке при выборе функций ядра, термин регуляризации имеет решающее значение;
- SVM напрямую не предоставляют оценки вероятности, они рассчитываются с использованием дорогостоящей пятикратной перекрестной проверки;
- требуется масштабирование данных;
- труден для интерпретации.

Метрики оценки качества прогнозирования

Для оценки точности и качества работы выбранных моделей прогнозирования и нейронных сетей используются такие показатели, как средняя абсолютная ошибка (MAE), средняя квадратичная ошибка (MSE), коэффициент детерминации(R^2)

MAE – измеряет среднюю абсолютную ошибку прогнозов. Для каждой точки вычисляется разница между прогнозами и целью, а затем усредняются эти значения. Чем дальше это значение от нуля, тем хуже модель. Показатель принимает так же нулевое значение, что может интерпретироваться как обучение идеальной модели, но и как, то, что сумма положительных и отрицательных значений прогнозов в итоге дала ноль.

MSE – измеряет среднюю квадратичную ошибку прогнозов. Для каждой точки вычисляется квадратная разница между прогнозами и целью, а затем усредняются эти значения. Чем выше это значение, тем хуже модель. Он никогда не бывает отрицательным, поскольку отдельные ошибки прогнозирования возводятся в квадрат, прежде чем их суммировать, но для идеальной модели это будет ноль.

R^2 – коэффициент детерминации, или R-квадрат, является еще одним показателем, который мы можем использовать для оценки модели, и он тесно связан с MSE, но имеет преимущество в том, что не имеет значения, являются ли выходные значения очень большими или очень маленькими, R^2 всегда будет

между $-\infty$ и 1. Коэффициент детерминации для модели с константой принимает значения от 0 до 1. Чем ближе значение коэффициента к 1, тем сильнее зависимость. При оценке регрессионных моделей это интерпретируется как соответствие модели данным. Для приемлемых моделей предполагается, что коэффициент детерминации должен быть хотя бы не меньше 50 %. Модели с коэффициентом детерминации выше 80 % можно признать достаточно хорошими. Значение коэффициента детерминации 1 означает функциональную зависимость между переменными. Когда R^2 отрицательно, это означает, что модель хуже, чем предсказание среднего значения.

1.3 Разведочный анализ данных

Разведочный анализ данных – один из первых и определяющих шагов в обработке данных. Разведочный анализ данных означает изучение данных для получения из них практической информации. Он включает в себя анализ и обобщение массивных наборов данных, часто в форме диаграмм и графиков.

Цель разведочного анализа – получение первоначальных представлений о характеристах 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.

В качестве инструментов анализа и визуализации данных используются оценка статистических характеристик датасета; гистограммы распределения каждого признака; диаграммы ящика с усами; попарные графики рассеяния точек; квантиль-квантильный график; тепловая карта взаимной корреляции признаков; проверка наличия пропусков и дубликатов.

Проведем разведочный анализ данных.

Информация о датасете представлена на рисунке 3. Практически все переменные имеют значения float64 (число с плавающей точкой), за исключением переменной "Угол нашивки, град" имеющую значение int64

(целые числа), строчные значения (str) отсутствуют, соответственно, отсутствует необходимость их категорирования. Пропусков не имеется. Очистка не требуется. Объединенный датасет имеет 1023 строки.

```
# Выведем информацию о датасете
data.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 1023 entries, 0 to 1022
Data columns (total 13 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Соотношение матрица-наполнитель    1023 non-null   float64
 1   Плотность, кг/м3                  1023 non-null   float64
 2   модуль упругости, ГПа            1023 non-null   float64
 3   Количество отвердителя, м.%       1023 non-null   float64
 4   Содержание эпоксидных групп,%_2  1023 non-null   float64
 5   Температура вспышки, С_2          1023 non-null   float64
 6   Поверхностная плотность, г/м2      1023 non-null   float64
 7   Модуль упругости при растяжении, ГПа 1023 non-null   float64
 8   Прочность при растяжении, МПа       1023 non-null   float64
 9   Потребление смолы, г/м2            1023 non-null   float64
 10  Угол нашивки, град                1023 non-null   int64  
 11  Шаг нашивки                      1023 non-null   float64
 12  Плотность нашивки                1023 non-null   float64
dtypes: float64(12), int64(1)
memory usage: 111.9 KB
```

Рисунок 3 – Информация о датасете «data»

На рисунке 4 представлено содержание уникальных значений в датасете. В основном в каждом столбце содержатся только уникальные значения, но в столбце "Угол нашивки, град" всего 2 значения.

```
# Смотрим уникальные значения
data.unique()

Соотношение матрица-наполнитель      1014
Плотность, кг/м3                      1013
модуль упругости, ГПа                 1020
Количество отвердителя, м.%          1005
Содержание эпоксидных групп,%_2      1004
Температура вспышки, С_2              1003
Поверхностная плотность, г/м2         1004
Модуль упругости при растяжении, ГПа 1004
Прочность при растяжении, МПа        1004
Потребление смолы, г/м2               1003
Угол нашивки, град                   2
Шаг нашивки                         989
Плотность нашивки                   988
dtype: int64
```

Рисунок 4 – Содержание уникальных значений в переменных датасета

Из описательной статистики на рисунке 5 видно, что диапазон разброса минимальных и максимальных значений переменных достаточно велик, минимальное значение в датасете равно «0», максимальное равно «3848»

```
# Выведем описательную статистику  
data.describe().round(2).T
```

	count	mean	std	min	25%	50%	75%	max
Соотношение матрица-наполнитель	1023.0	2.93	0.91	0.39	2.32	2.91	3.55	5.59
Плотность, кг/м3	1023.0	1975.73	73.73	1731.76	1924.16	1977.62	2021.37	2207.77
модуль упругости, ГПа	1023.0	739.92	330.23	2.44	500.05	739.66	961.81	1911.54
Количество отвердителя, м.%	1023.0	110.57	28.30	17.74	92.44	110.56	129.73	198.95
Содержание эпоксидных групп,%_2	1023.0	22.24	2.41	14.25	20.61	22.23	23.96	33.00
Температура вспышки, С_2	1023.0	285.88	40.94	100.00	259.07	285.90	313.00	413.27
Поверхностная плотность, г/м2	1023.0	482.73	281.31	0.60	266.82	451.86	693.23	1399.54
Модуль упругости при растяжении, ГПа	1023.0	73.33	3.12	64.05	71.25	73.27	75.36	82.68
Прочность при растяжении, МПа	1023.0	2466.92	485.63	1036.86	2135.85	2459.52	2767.19	3848.44
Потребление смолы, г/м2	1023.0	218.42	59.74	33.80	179.63	219.20	257.48	414.59
Угол нашивки, град	1023.0	44.25	45.02	0.00	0.00	0.00	90.00	90.00
Шаг нашивки	1023.0	6.90	2.56	0.00	5.08	6.92	8.59	14.44
Плотность нашивки	1023.0	57.15	12.35	0.00	49.80	57.34	64.94	103.99

Рисунок 5 – Описательная статистика датасета

Для отображения характера распределения переменных в датасете построим гистограммы распределения (Рисунок 6).

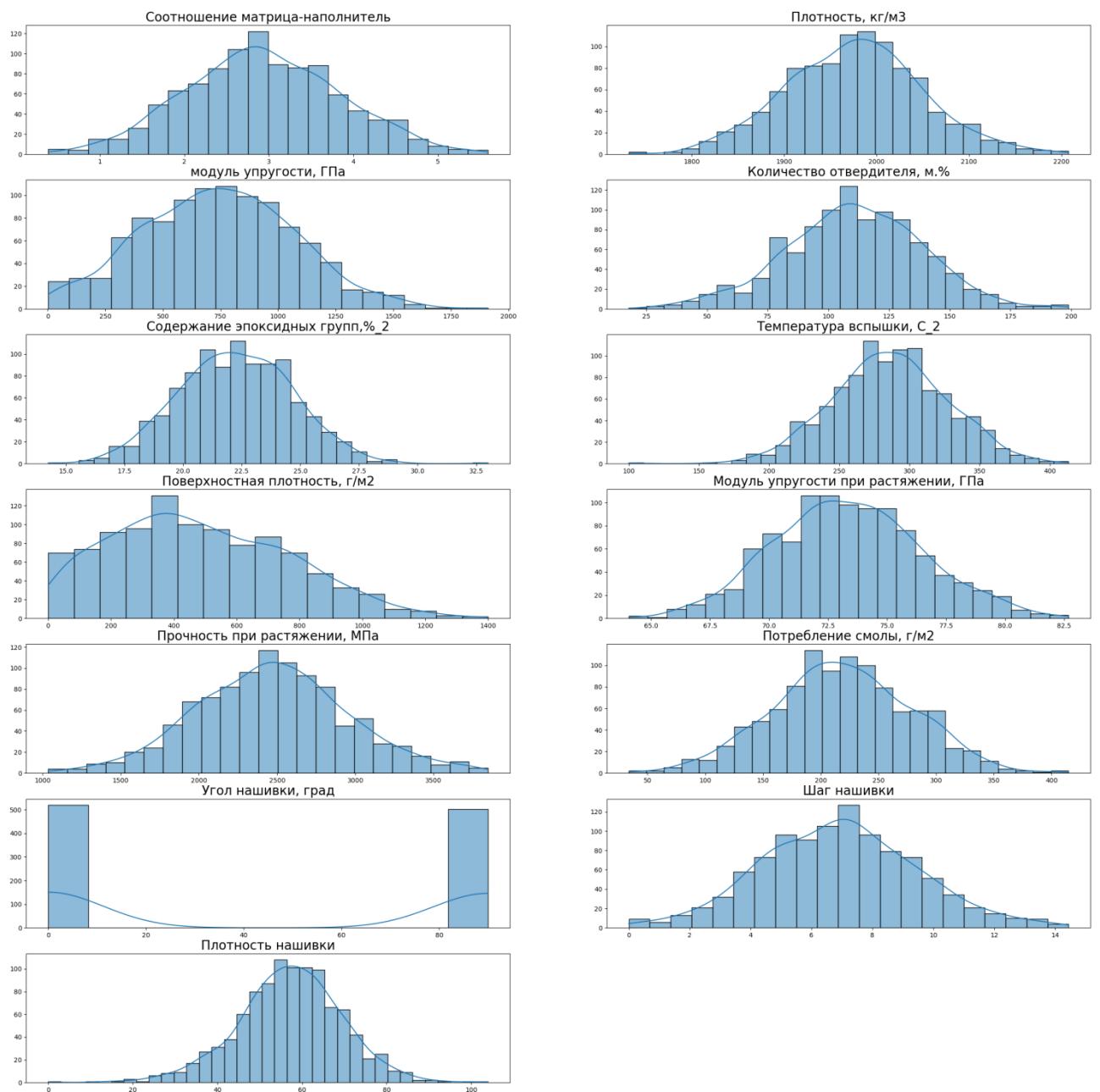


Рисунок 6 – Гистограммы распределения для каждой переменной датасета

По гистограммам распределения видно, что распределение величин близко к нормальному для большей части переменных, за исключением поверхности плотности – смещением влево и угла нашивки – дискретная величина, график оказался не показателен.

Построим диаграммы типа boxplot («Ящик с усами»), для исследуемого датасета (Рисунок 7).

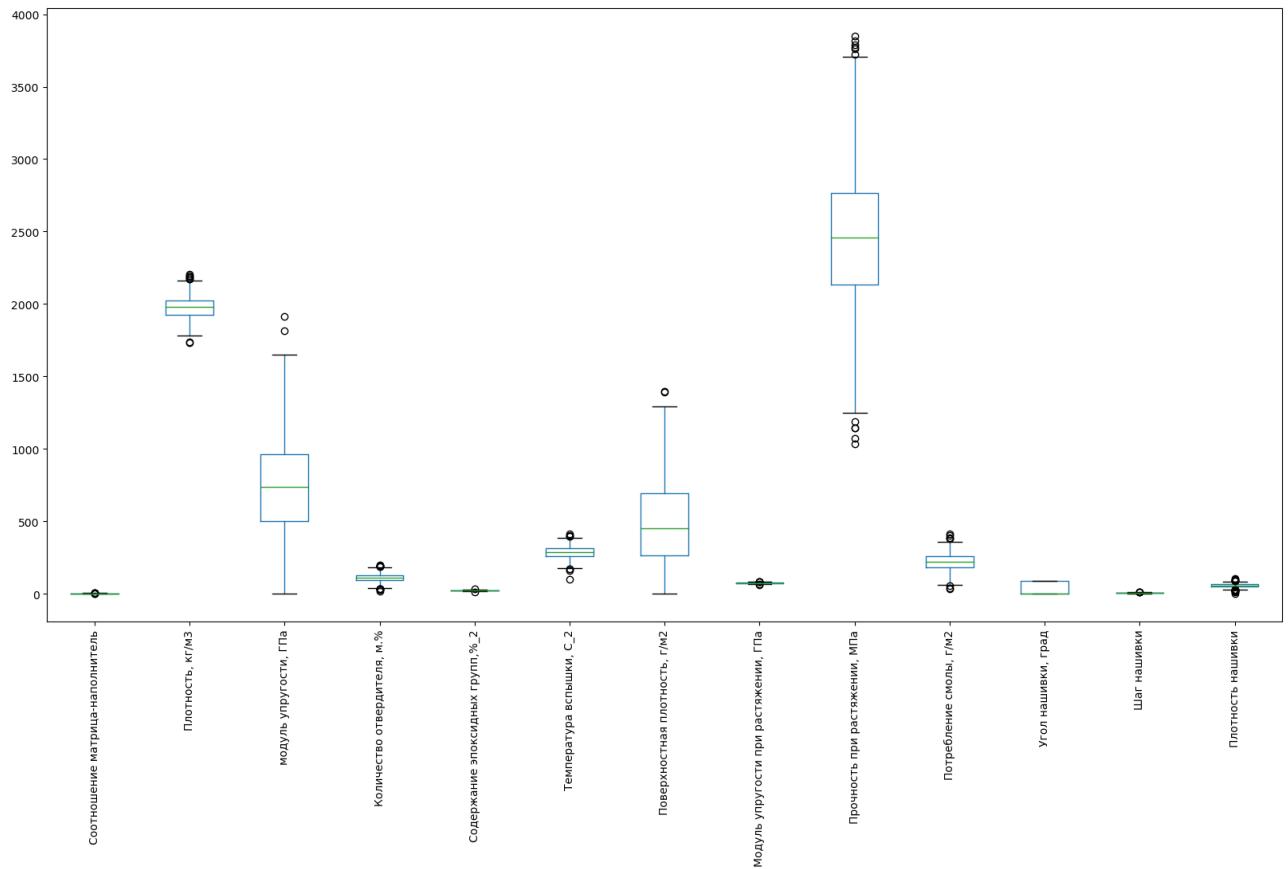


Рисунок 7 – Диаграммы boxplot («Ящик с усами») для датасета

Из диаграммы Boxplot («Ящик с усами») видно, что порядок значений переменных различается в разы.

Построив диаграммы «Ящик с усами» для каждой переменной датасета (Рисунок 8), видно, что практически у всех переменных имеются выбросы, кроме “Угол нашивки, град”, так как данная переменная принимает дискретные значения и диаграмма «Ящик с усами» для этой переменной не показательна.

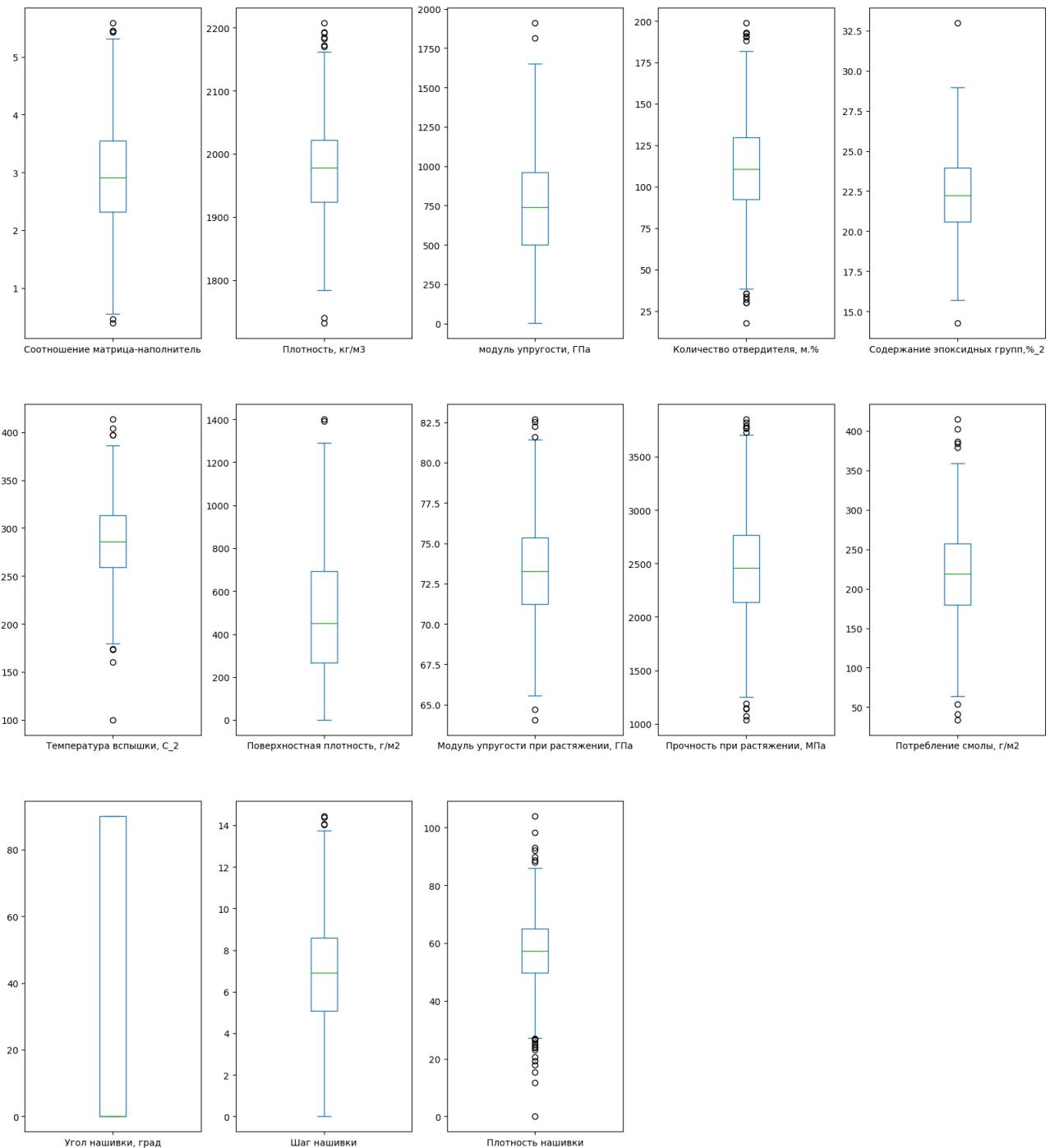


Рисунок 8 – Диаграммы Boxplot («Ящик с усами») для каждой переменной

В целях выявления зависимостей между переменными построим тепловую карту коэффициентов корреляции (Рисунок 9). Коэффициенты корреляции показывают слабую зависимость между переменными.

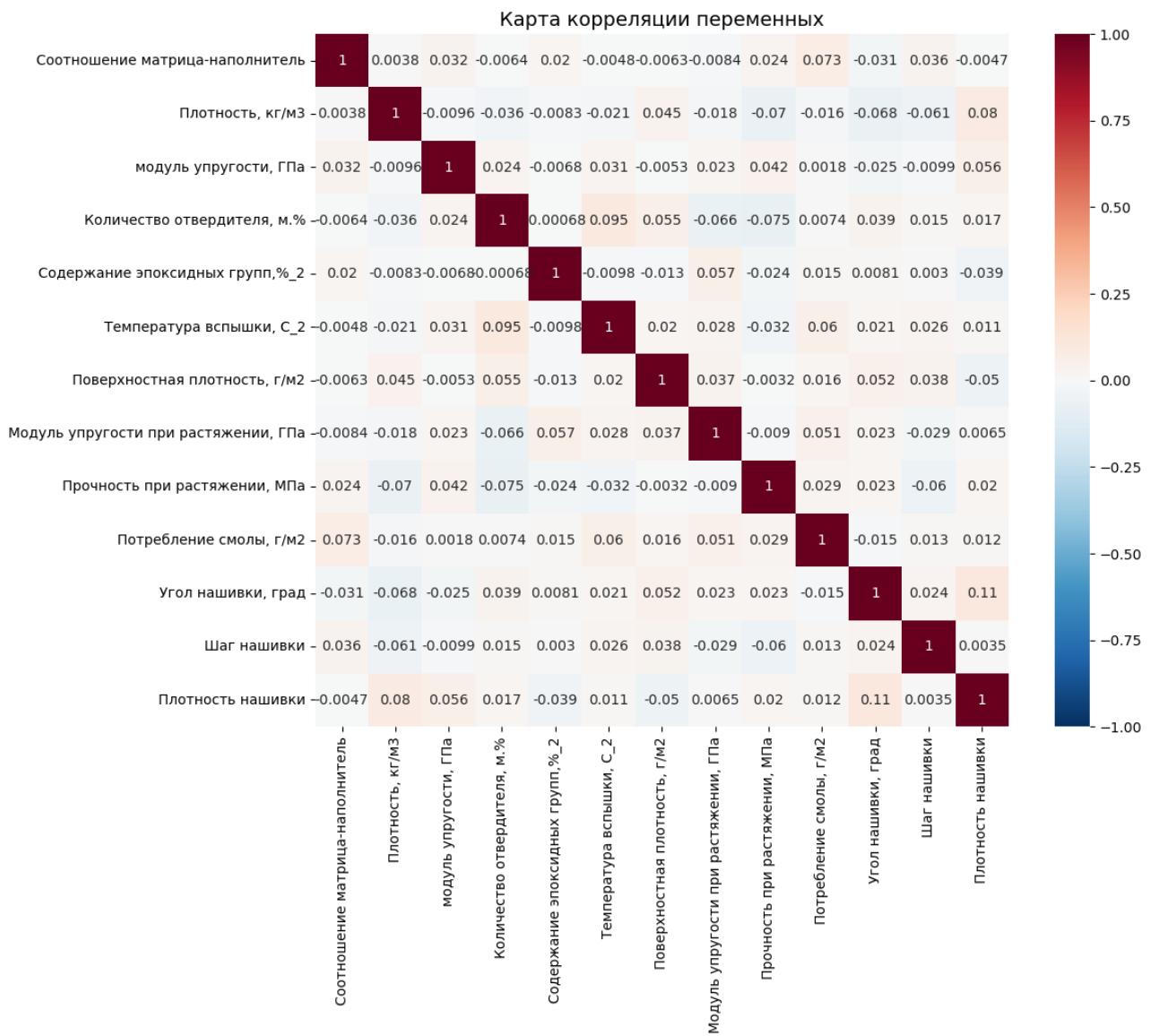


Рисунок 9 – Тепловая карта коэффициентов корреляции

Также можно построить попарные графики рассеивания (Рисунок 10), на которых также наблюдаются выбросы и отсутствие каких-либо выраженных зависимостей между переменными.

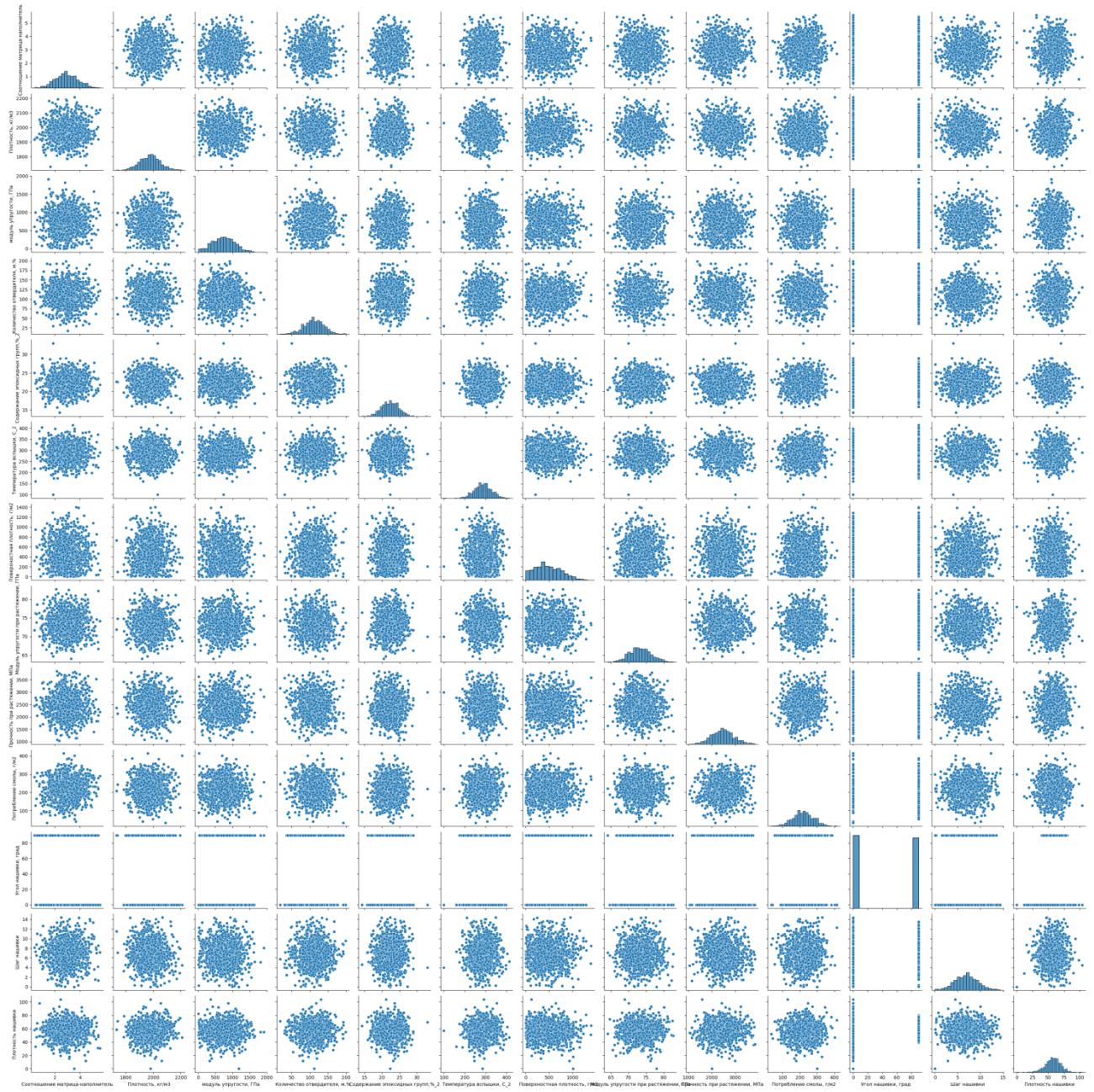


Рисунок 10 – Попарные графики рассеивания

2. Практическая часть

2.1 Предобработка данных

Так как в процессе анализа датасета в переменных наблюдались выбросы, выполним их очистку. Выбросы – это данные, которые существенно отличаются от других наблюдений. Они могут соответствовать реальным отклонениям, но могут быть и просто ошибками.

Удаление выбросов выполним с помощью межквартильных диапазонов

$$IQR = \text{Quartile3} - \text{Quartile1}$$

Для выявления выбросов для каждой переменной определяются значения, находящиеся выше и ниже нормального диапазона наборов данных, т.е. за пределами верхней и нижней границы:

- верхняя граница = $Q3 + 1.5 * IQR$;
- нижняя граница = $Q1 - 1.5 * IQR$.

В результате самые сильные выбросы удалены, размерность очищенного датасета составляет 936 на 13. Повторное формирование диаграммы boxplot «Ящик с усами» наглядно показывает снижение выбросов (Рисунок 11).

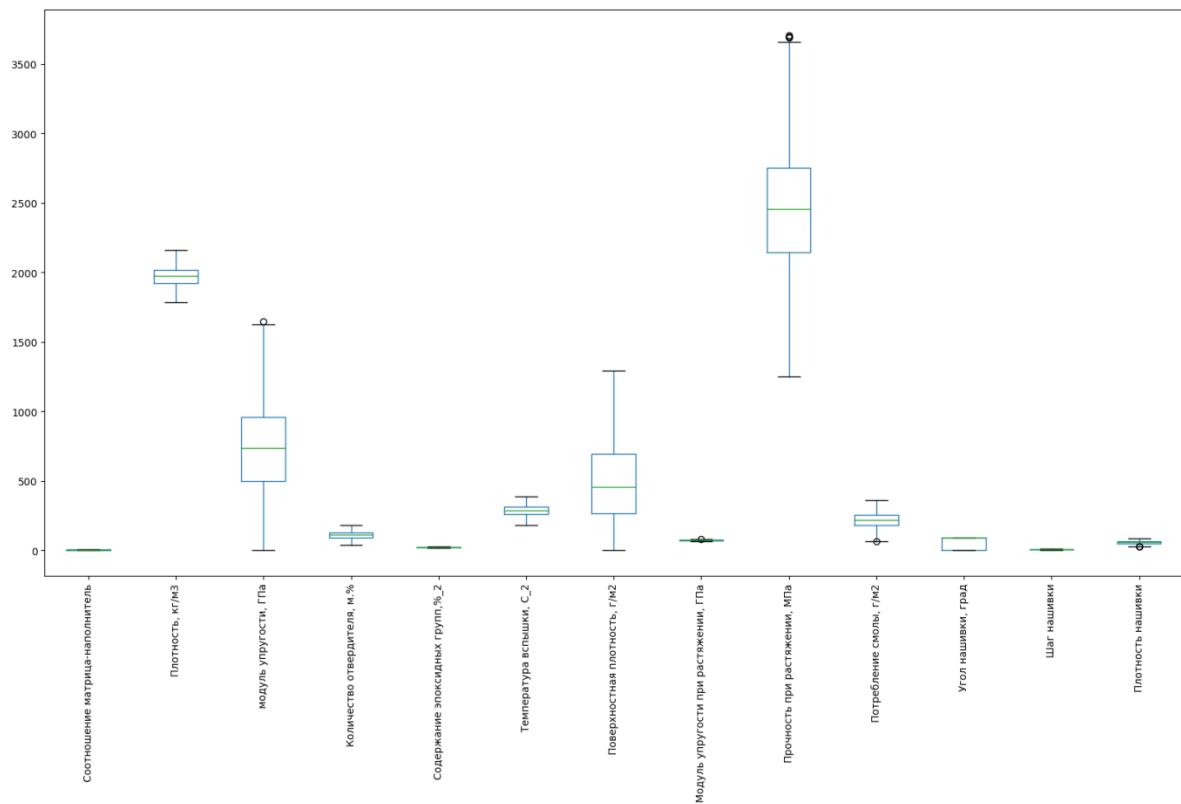


Рисунок 11 – Диаграммы boxplot после удаления выбросов

Построив график распределения для каждой переменной, убедимся в необходимости нормализации данных (Рисунок 12).

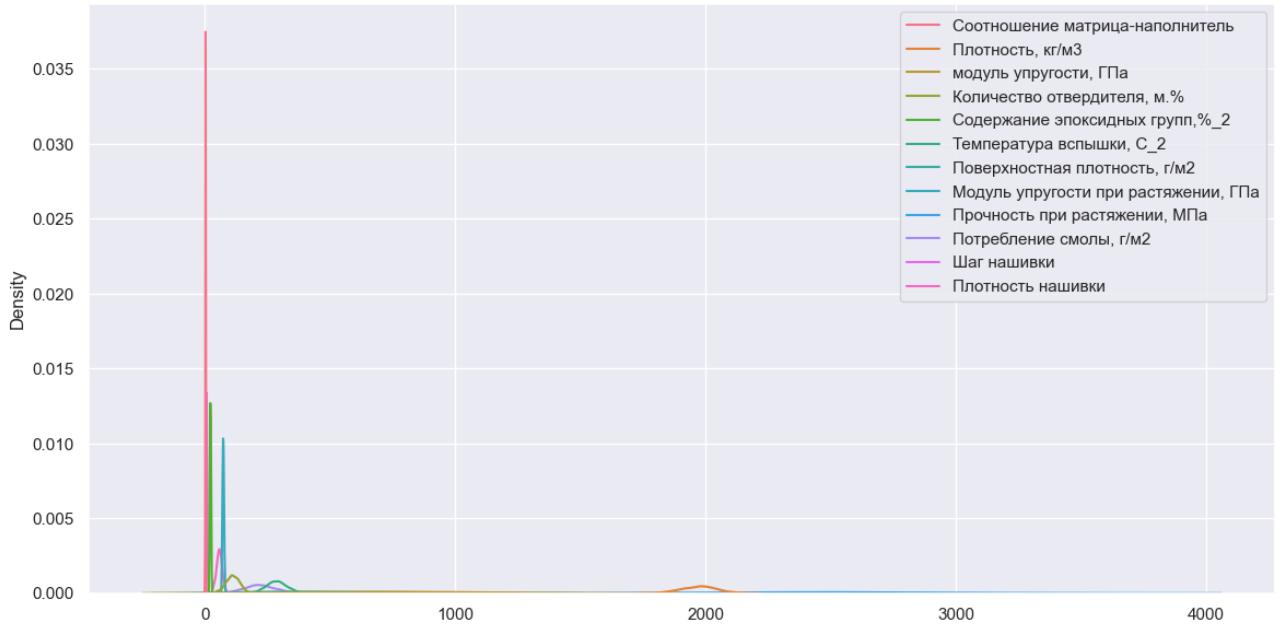


Рисунок 12 – График оценки плотности ядра датасета

Нормализация – это преобразование данных к некоторым безразмерным единицам. Иногда в рамках заданного диапазона, например, [0..1] или [-1..1]. Иногда с какими-то заданным свойством, как, например, стандартным отклонением равным 1.

Ключевая цель нормализации – приведение различных данных в самых разных единицах измерения и диапазонах значений к единому виду, который позволит сравнивать их между собой или использовать для расчёта схожести объектов.

Нормализацию данных проведем при помощи метода `MinMaxScaler()` с заданным диапазоном, по умолчанию от 0 до 1. Результат нормализации представлен на рисунках 13, 14

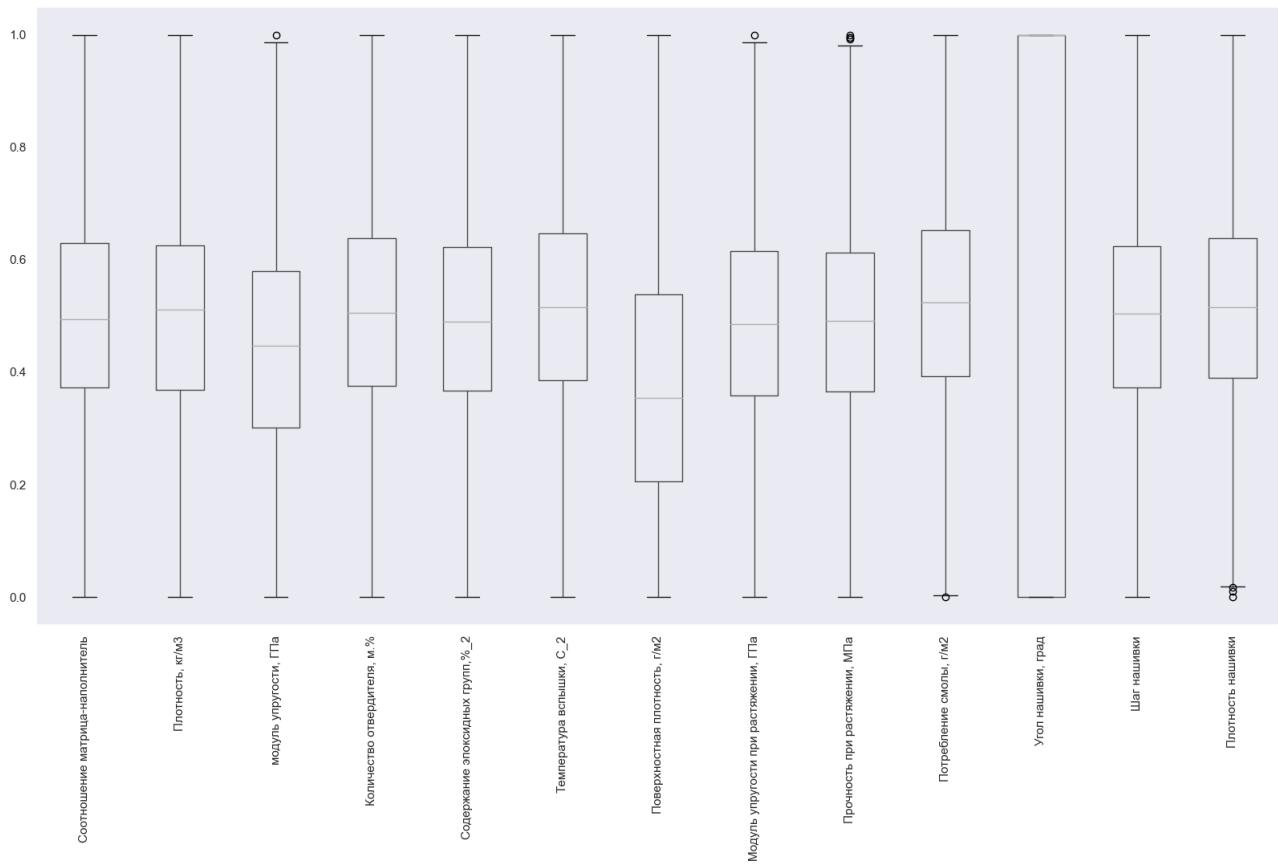


Рисунок 13 – Диаграммы boxplot после нормализации

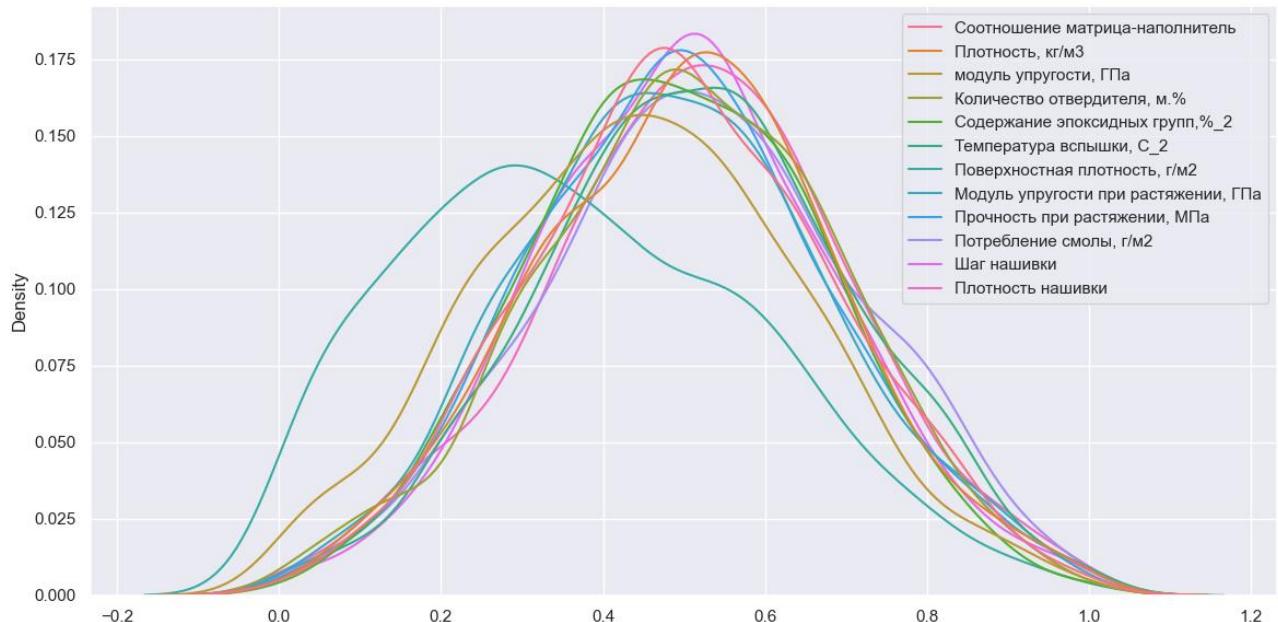


Рисунок 14 – График оценки плотности ядра датасета после нормализации

2.2 Разработка и обучение модели

В ходе работы обучим модели для прогноза “Модуля упругости при растяжении” и “Прочности при растяжении”. При построении модели 30%

данных оставим на тестирование модели, остальные данные выделим для обучения моделей. При построении моделей проведем поиск гиперпараметров модели с помощью поиска по сетке с перекрестной проверкой, количество блоков равно 10.

Характеристики качества пригодности моделей прогнозирования описывают, насколько достоверно, выбранная в качестве генератора прогноза, модель описывает ретроспективу исследуемого явления. Чем точнее построенная модель объясняла прошлое, тем больше вероятность того, что она будет удачно предсказывать будущее. Надежность моделей прогнозирования оценивается путем сравнения фактических и предсказанных значений. Эта разница позволяет проверить, применима ли к конкретным данным рассматриваемая модель и те предположения, на которых она основана. Основными оценочными характеристиками качества регрессионной модели используем такие показатели, как средняя абсолютная ошибка (MAE), средняя квадратичная ошибка (MSE), коэффициент детерминации (R^2).

2.2.1 Прогноз модуля упругости при растяжении

Метод линейной регрессии (LinearRegression): по результатам поиска оптимальных гиперпараметров выявлены основные настройки для получения наилучшего результата при применении метода `LinearRegression(fit_intercept=True, positive=True)`. С данными настройками модель была обучена на тренировочной выборке. Результат на тренировочной выборке: $R^2: 0.009$.

Метод решающих деревьев (DecisionTreeRegressor): по результатам поиска оптимальных гиперпараметров выявлены основные настройки для получения наилучшего результата при применении метода `DecisionTreeRegressor(criterion='poisson', max_depth=2, splitter='random')`. С данными настройками модель была обучена на тренировочной выборке. Результат на тренировочной выборке: $R^2: 0.004$.

Метод случайного леса (RandomForestRegressor): по результатам поиска оптимальных гиперпараметров выявлены основные настройки для получения

наилучшего результата при применении метода RandomForestRegressor (max_features= 'sqrt', max_depth= 1, n_estimators= 50, random_state= 42). С данными настройками модель была обучена на тренировочной выборке. Результат на тренировочной выборке: R^2 : 0.015.

Метод К ближайших соседей (KneighborsRegressor): по результатам поиска оптимальных гиперпараметров выявлены основные настройки для получения наилучшего результата при применении метода KneighborsRegressor (algorithm= 'auto', n_neighbors= 10, weights= 'uniform'). С данными настройками модель была обучена на тренировочной выборке. Результат на тренировочной выборке: R^2 : 0.07.

Метод опорных векторов (SVR): по результатам поиска оптимальных гиперпараметров выявлены основные настройки для получения наилучшего результата при применении метода SVR (gamma= 'auto', kernel= 'poly', shrinking= True). С данными настройками модель была обучена на тренировочной выборке. Результат на тренировочной выборке: R^2 : 0.023.

2.2.2 Прогноз прочности при растяжении

Метод линейной регрессии (LinearRegression): по результатам поиска оптимальных гиперпараметров выявлены основные настройки для получения наилучшего результата при применении метода LinearRegression(fit_intercept= True, positive= True). С данными настройками модель была обучена на тренировочной выборке. Результат на тренировочной выборке: R^2 : 0.004.

Метод решающих деревьев (DecisionTreeRegressor): по результатам поиска оптимальных гиперпараметров выявлены основные настройки для получения наилучшего результата при применении метода DecisionTreeRegressor (criterion= 'poisson', max_depth= 2, splitter= 'random'). С данными настройками модель была обучена на тренировочной выборке. Результат на тренировочной выборке: R^2 : 0.024.

Метод случайного леса (RandomForestRegressor): по результатам поиска оптимальных гиперпараметров выявлены основные настройки для получения

наилучшего результата при применении метода RandomForestRegressor (`max_features= 'sqrt'`, `max_depth= 1`, `n_estimators= 100`, `random_state= 42`). С данными настройками модель была обучена на тренировочной выборке. Результат на тренировочной выборке: R^2 : 0.027.

Метод К ближайших соседей (`KNeighborsRegressor`): по результатам поиска оптимальных гиперпараметров выявлены основные настройки для получения наилучшего результата при применении метода `KNeighborsRegressor` (`algorithm= 'brute'`, `n_neighbors= 10`, `weights= 'distance'`). С данными настройками модель была обучена на тренировочной выборке. Результат на тренировочной выборке: R^2 : 0.99.

Метод опорных векторов (`SVR`): по результатам поиска оптимальных гиперпараметров выявлены основные настройки для получения наилучшего результата при применении метода `SVR` (`gamma= 'auto'`, `kernel= 'poly'`, `shrinking= True`). С данными настройками модель была обучена на тренировочной выборке. Результат на тренировочной выборке: R^2 : 0.031.

2.3 Тестирование модели

2.3.1 Прогноз модуля упругости при растяжении

Метод линейной регрессии (`LinearRegression`).

Результат на тестовой выборке: R^2 : -0.005, MAE: 0.151, MSE: 0.034. Графическое представление точности предсказания на рисунке 15.

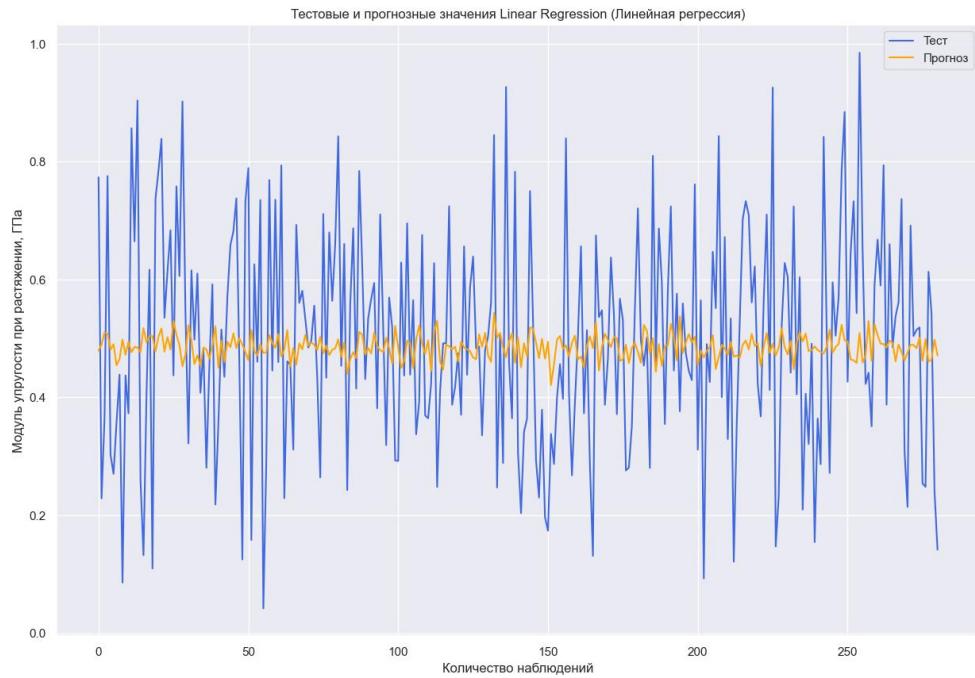


Рисунок 15 – Точность метода LinearRegression

Метод решающих деревьев (DecisionTreeRegressor).

Результат на тестовой выборке: $R^2: -0.014$, MAE: 0.152, MSE: 0.035.

Графическое представление точности предсказания на рисунке 16.

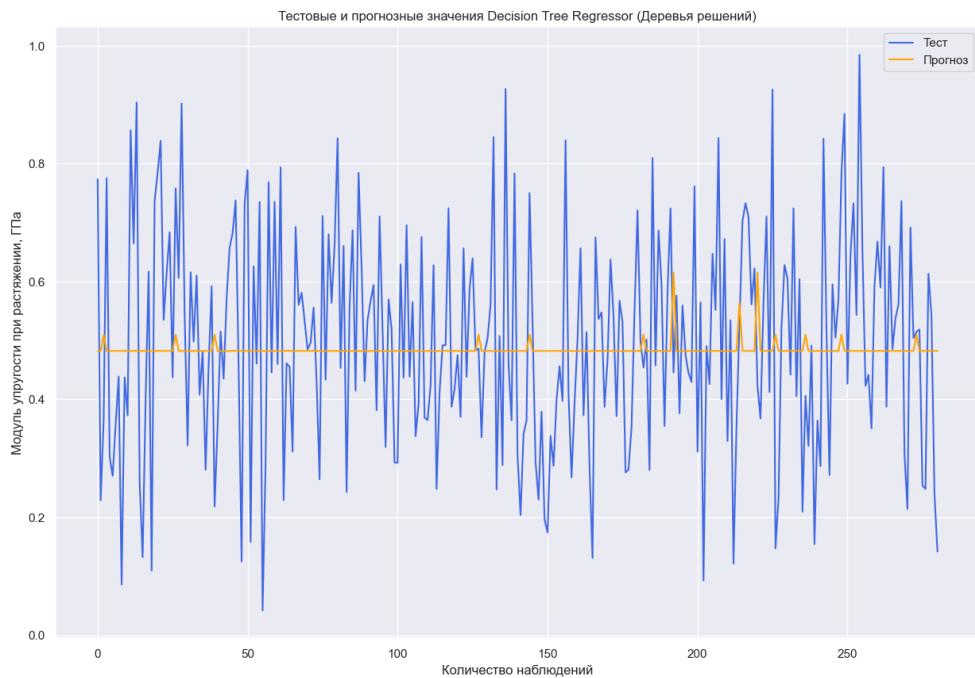


Рисунок 16 – Точность метода DecisionTreeRegressor

Метод случайного леса (RandomForestRegressor).

Результат на тестовой выборке: R^2 : -0.005, MAE: 0.150, MSE: 0.034.

Графическое представление точности предсказания на рисунке 17.

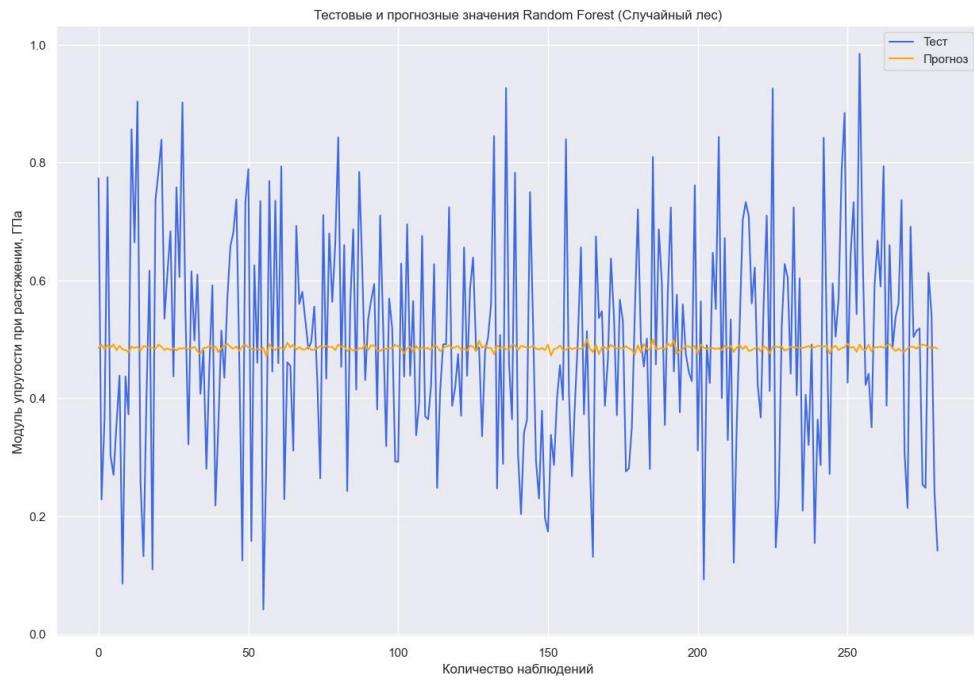


Рисунок 17 – Точность метода RandomForestRegressor

Метод К ближайших соседей (KNeighborsRegressor).

Результат на тестовой выборке: R^2 : -0.102, MAE: 0.158, MSE: 0.038.

Графическое представление точности предсказания на рисунке 18.

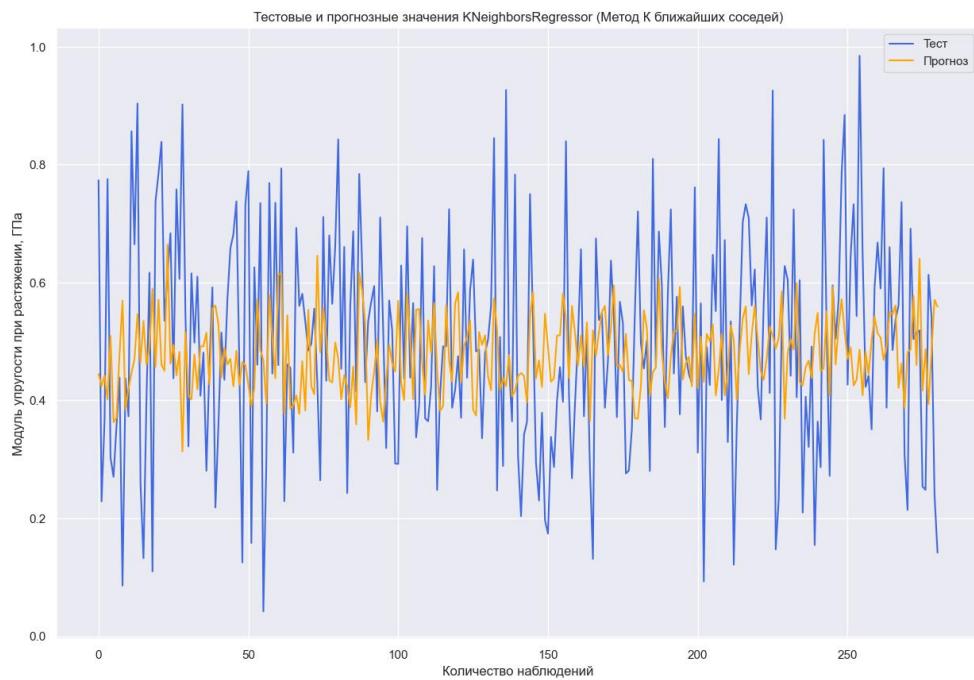


Рисунок 18 – Точность метода KNeighborsRegressor

Метод опорных векторов (SVR).

Результат на тестовой выборке: R^2 : 0.01, MAE: 0.150, MSE: 0.034.

Графическое представление точности предсказания на рисунке 19.

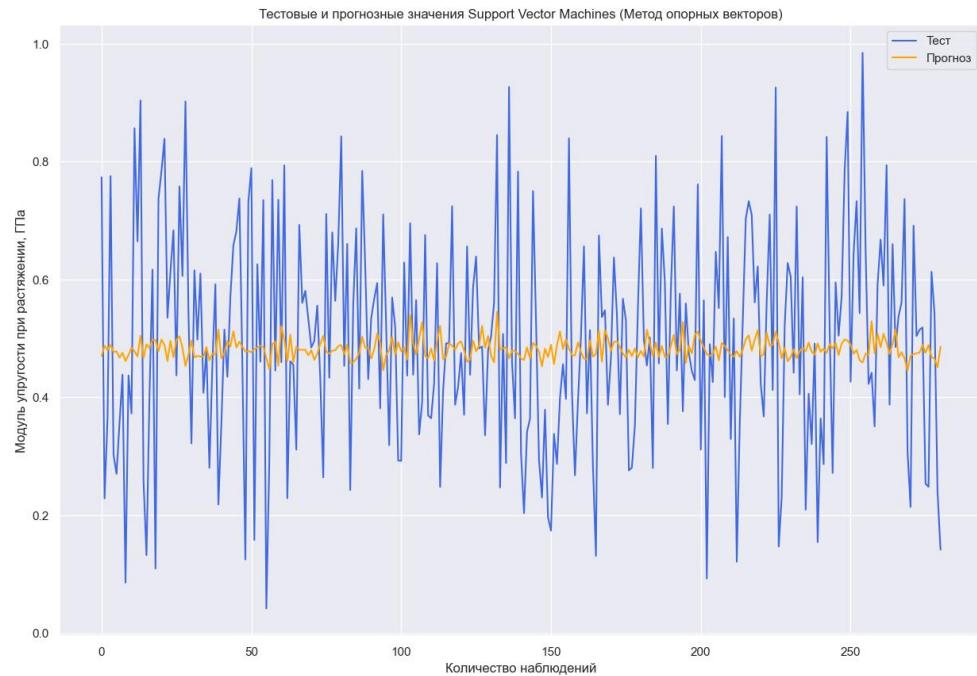


Рисунок 19 – Точность метода SVR

Построим сводную таблицу 1 результатов тестирования использованных моделей, сравнивать будем по средней абсолютной ошибке (MAE), средней квадратичной ошибке (MSE) и коэффициенту детерминации (R^2).

Таблица 1 – Сравнение обученных моделей

Модуль упругости при растяжении				
Метод машинного обучения	MAE	MSE	R^2	R^2 (train)
LinearRegression (Линейная регрессия)	0.151824	0.034828	-0.005818	0.009843
DecisionTreeRegressor (Регрессионное дерево решений)	0.152394	0.035123	-0.014331	0.004219
RandomForestRegressor (Случайный лес регрессии)	0.150918	0.034644	-0.000508	0.015890
KneighborsRegressor (Метод К ближайших соседей)	0.158387	0.038174	-0.102440	0.070250
SVR (Метод опорных векторов)	0.150919	0.034270	0.010285	0.023628

Из сводной таблицы видно, что при предсказании модуля упругости при растяжении не удалось приблизится к идеальному результату более, чем до предсказания среднего значения. Коэффициент детерминации близкий к нулю, а тем более отрицательное его значение свидетельствует о низком качестве модели и отсутствии линейных связей. MAE и MSE показывает высокие показатели, что так же подтверждает отсутствие линейных связей.

2.3.2 Прогноз прочности при растяжении

Метод линейной регрессии (LinearRegression).

Результат на тестовой выборке: $R^2: -0.043$, MAE: 0.147, MSE: 0.035.

Графическое представление точности предсказания на рисунке 20.

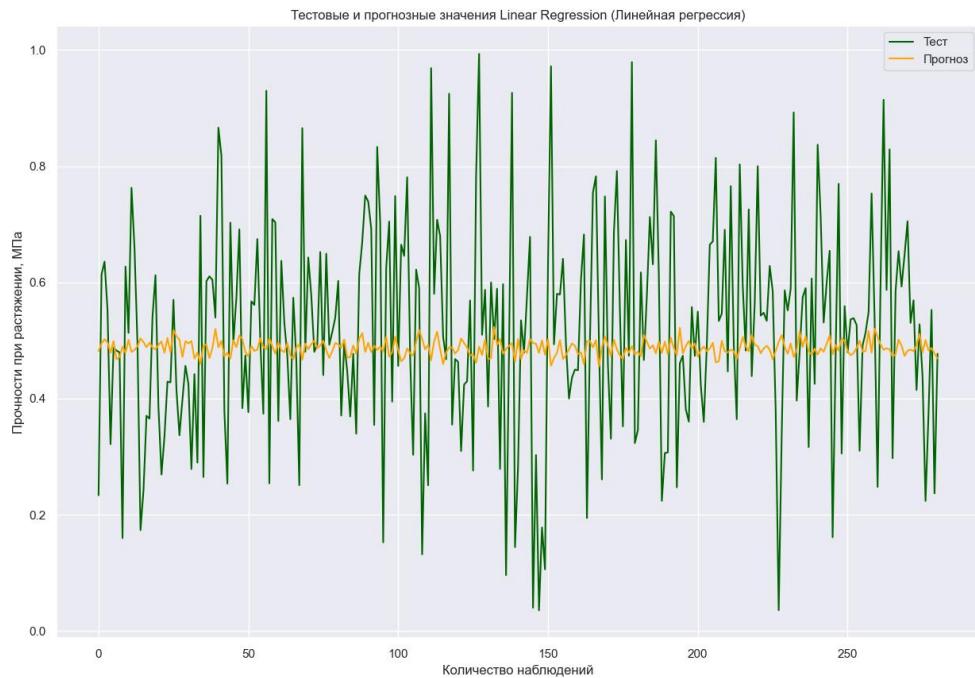


Рисунок 20 – Точность метода LinearRegression

Метод решающих деревьев (DecisionTreeRegressor).

Результат на тестовой выборке: $R^2: -0.054$, MAE: 0.148, MSE: 0.035.

Графическое представление точности предсказания на рисунке 21.

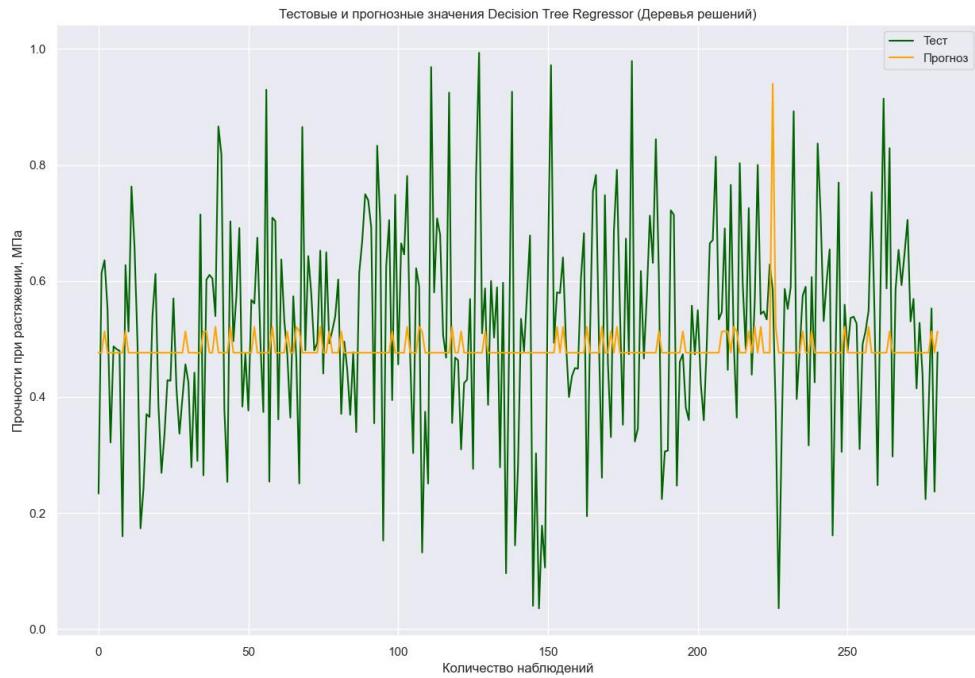


Рисунок 21 – Точность метода DecisionTreeRegressor

Метод случайного леса (RandomForestRegressor).

Результат на тестовой выборке: $R^2: -0.032$, MAE: 0.147, MSE: 0.034.

Графическое представление точности предсказания на рисунке 22.

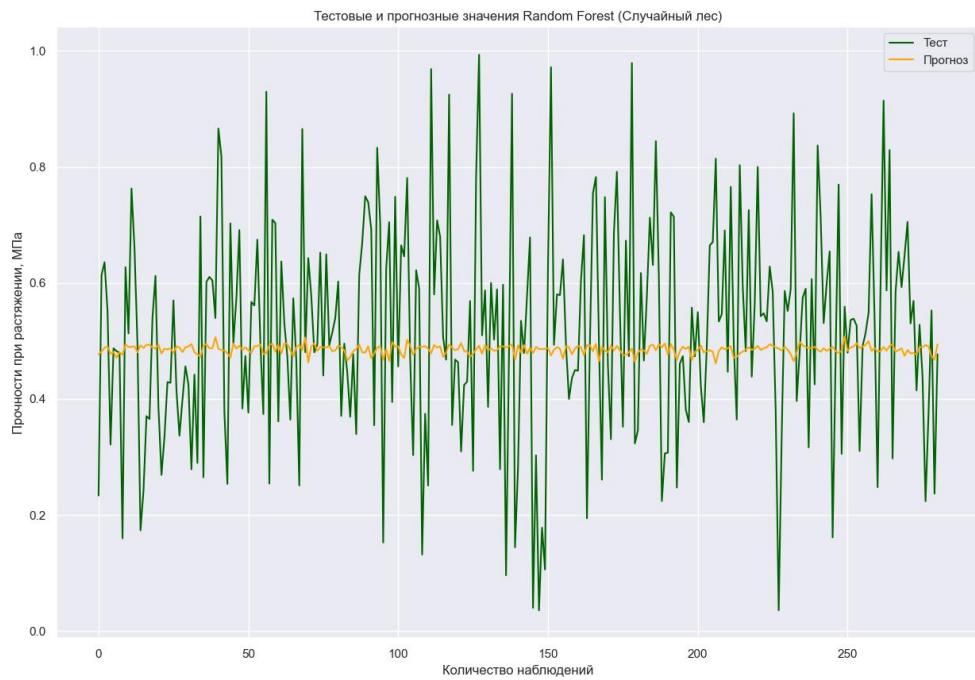


Рисунок 22 – Точность метода RandomForestRegressor

Метод К ближайших соседей (KneighborsRegressor).

Результат на тестовой выборке: R^2 : -0.147, MAE: 0.153, MSE: 0.038.

Графическое представление точности предсказания на рисунке 23.

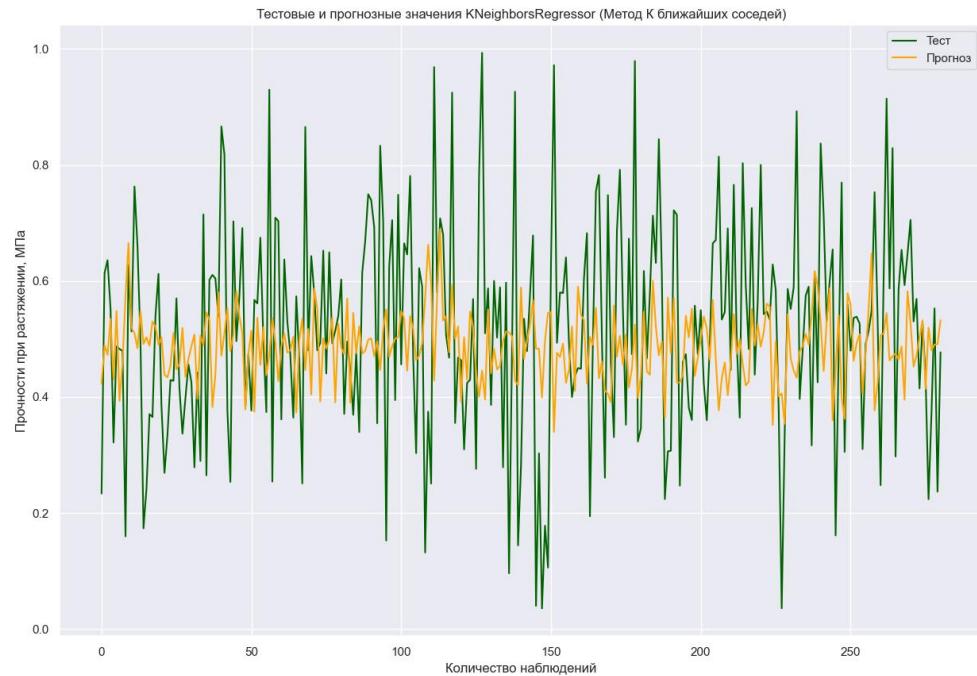


Рисунок 23 – Точность метода KneighborsRegressor

Метод опорных векторов (SVR).

Результат на тестовой выборке: R^2 : -0.038, MAE: 0.147, MSE: 0.035.

Графическое представление точности предсказания на рисунке 24.

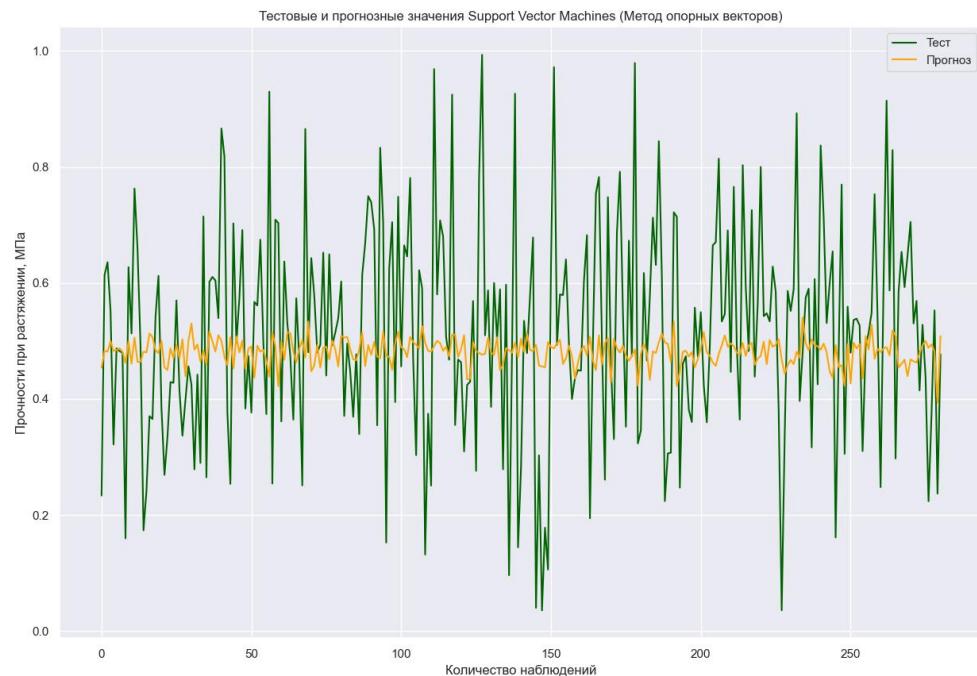


Рисунок 24 – Точность метода SVR

Построим сводную таблицу 2 результатов тестирования использованных моделей, сравнивать будем по средней абсолютной ошибке (MAE), средней квадратичной ошибке (MSE) и коэффициенту детерминации (R^2).

Таблица 2 – Сравнение обученных моделей

Прочность при растяжении				
Метод машинного обучения	MAE	MSE	R^2	R^2 (train)
LinearRegression (Линейная регрессия)	0.147422	0.035332	-0.043577	0.004432
DecisionTreeRegressor (Регрессионное дерево решений)	0.148127	0.035708	-0.054684	0.024868
RandomForestRegressor (Случайный лес регрессии)	0.147014	0.034962	-0.032641	0.027347
KneighborsRegressor (Метод К ближайших соседей)	0.153163	0.038854	-0.147594	0.999999
SVR (Метод опорных векторов)	0.147598	0.035166	-0.038648	0.031339

Из сводной таблицы видно, что при предсказании прочности при растяжении не удалось приблизится к идеальному результату более, чем до предсказания среднего значения. Коэффициент детерминации близкий к нулю, а тем более отрицательное его значение свидетельствует о низком качестве модели и отсутствии линейных связей. MAE и MSE показывает высокие показатели, что так же подтверждает отсутствие линейных связей.

2.4 Разработка нейронной сети для рекомендации соотношения матрица-наполнитель

Для построения нейронных сетей используем библиотеку Keras.

Для рекомендации соотношения матрица-наполнитель обучим несколько искусственных нейронных сетей. Применим модель нейронной сети Sequential. Модель Sequential представляет собой линейную структуру, в которой слои добавляются один за другим. Каждый слой принимает выходные данные предыдущего слоя в качестве входных данных. Таким образом, в модели Sequential нет возможности объединять несколько входов или выходов.

Данные в нейросеть подаются нормализованные в диапазоне от 0 до 1, датасет разделен на тестовую (30%) и обучающую (70%) выборки, с целевым параметром «Соотношение матрица-наполнитель».

Первая нейросеть (Рисунок 25).

Первый слой - это входной слой Dense(), который имеет 128 нейронов, функцию активации 'relu' и количество входных параметров, соответствующее размерности массива X_matr_train. Затем добавляются скрытые слои Dense() с функцией активации 'relu' и разными количествами нейронов: 64, 32, 16, 8, 4. Последний слой Dense() имеет 2 нейрона и функцию активации 'relu'. Выходной слой Dense() имеет один нейрон и не имеет функции активации.

```
# Создадим нейронную сеть для предсказания параметра "Соотношение матрица-наполнитель"
# Создадим модель помошью класса Sequential, который предполагает, что выходы одного слоя идут на следующий слой
model1 = Sequential()
# Добавим входной слой
model1.add(Dense(128, input_dim=X_matr_train.shape[1], activation='relu'))
# Добавим скрытые слои
model1.add(Dense(64, activation='relu'))
model1.add(Dense(32, activation='relu'))
model1.add(Dense(16, activation='relu'))
model1.add(Dense(8, activation='relu'))
model1.add(Dense(4, activation='relu'))
model1.add(Dense(2, activation='relu'))
# Добавим выходной слой
model1.add(Dense(1))
# Компилируем модель
model1.compile(loss='mse', optimizer='adam', metrics=['mae'])
# Выведем информацию о модели
model1.summary()
```

Рисунок 25 – Сводные данные первой нейросети

Обучение проводилось пакетами данных по 100 записей в течение 60 эпох, размер валидационной выборки 20%. Процесс обучения показан на рисунке 26.

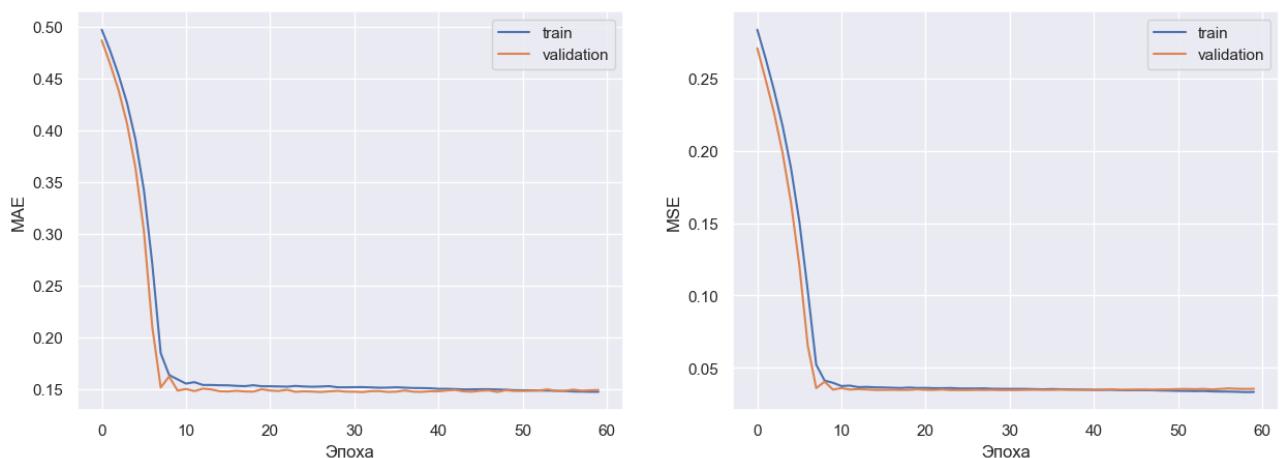


Рисунок 26 – Обучение первой нейросети

Результат на тестовой выборке: MAE: 0.149729, MSE: 0.033838, R²: -0.009423. Графическое представление точности предсказания на рисунке 27.

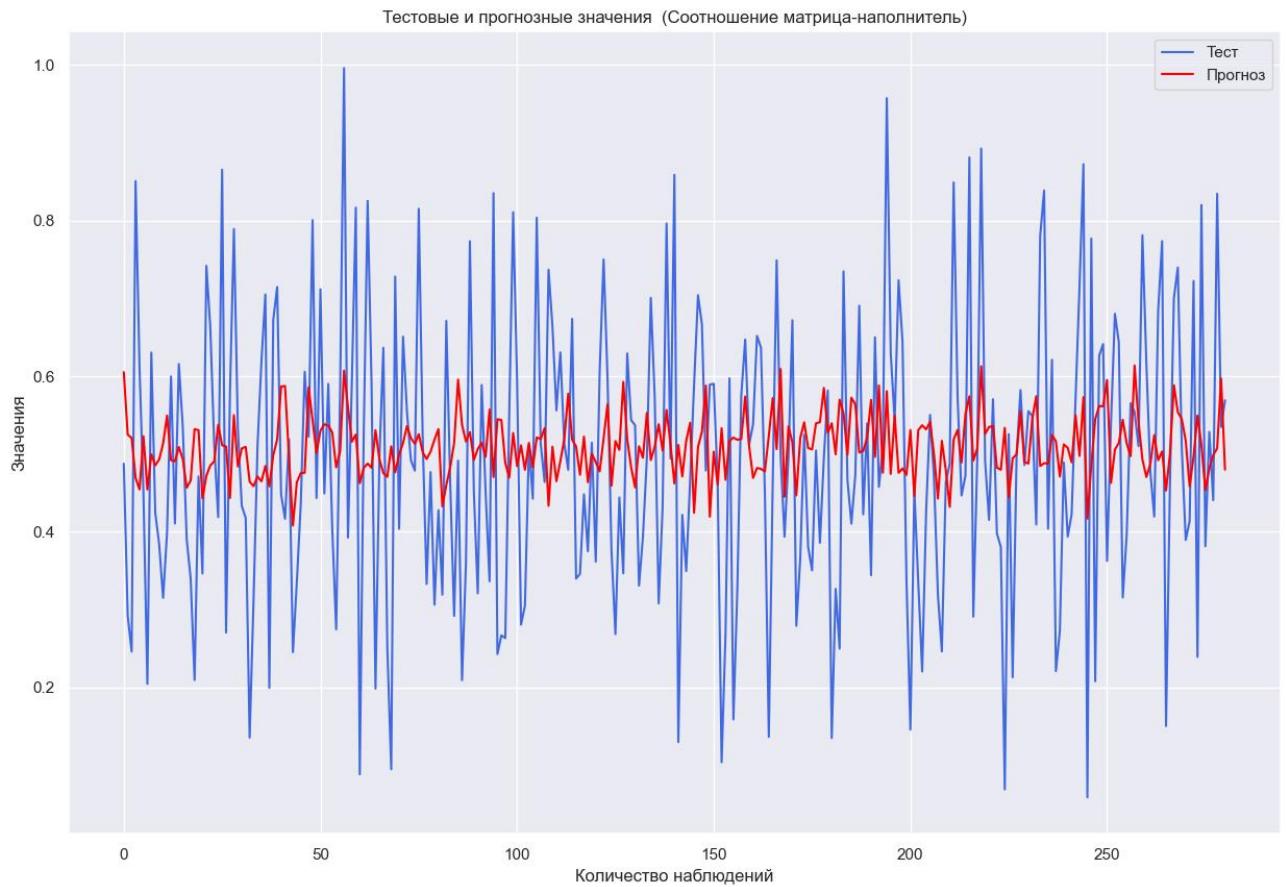


Рисунок 27 – График прогнозных значений первой нейросети

Вторая нейросеть (Рисунок 28).

В этом варианте добавлены слои Dropout с параметрами 0.12, который используется для прореживания (случайного отключения) 12% нейронов на данных слоях, чтобы предотвратить переобучение. Активационная функция на внутренних слоях "relu", на выходном слое "tanh".

```
# Создадим ещё одну модель, добавим прореживание Dropout

model2 = Sequential()
# Добавим входной слой
model2.add(Dense(64, input_dim=X_matr_train.shape[1], activation='relu'))
model2.add(Dropout(0.12))
model2.add(Dense(32, activation='relu'))
model2.add(Dropout(0.12))
model2.add(Dense(16, activation='relu'))
model2.add(Dropout(0.12))
model2.add(Dense(8, activation='tanh'))
# Добавим выходной слой
model2.add(Dense(1))
# Компилируем модель
model2.compile(loss='mse', optimizer='adam', metrics=['mae'])
# Выведем информацию о модели
model2.summary()
```

Рисунок 28 – Сводные данные второй нейросети

Обучение проводилось пакетами данных по 32 записи в течение 60 эпох, размер валидационной выборки 20%. Процесс обучения показан на рисунке 29.

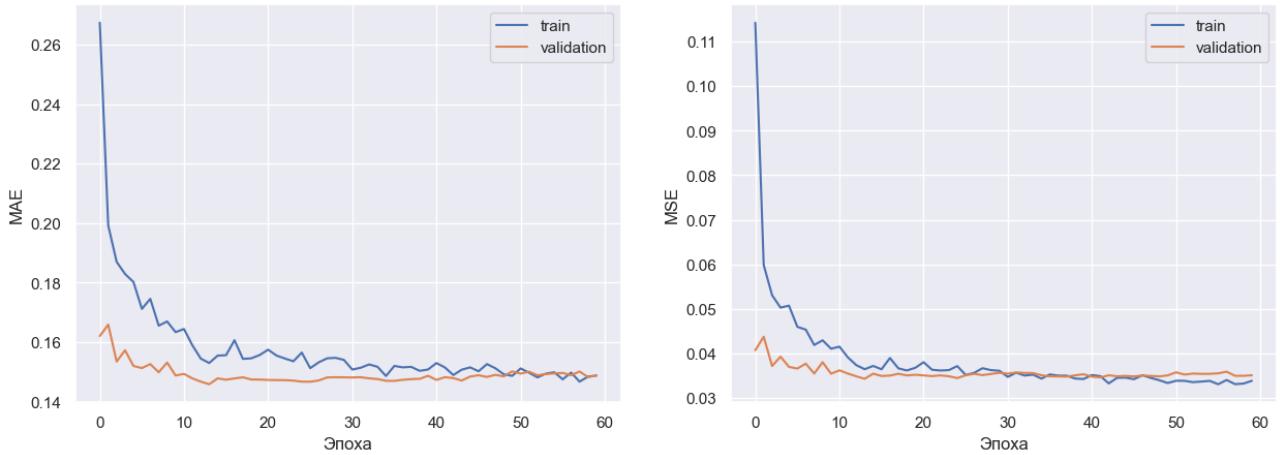


Рисунок 29 – Обучение второй нейросети

Результат на тестовой выборке: МАЕ: 0.150222, MSE: 0.035252, R^2 : -0.051625. Графическое представление точности предсказания на рисунке 30.

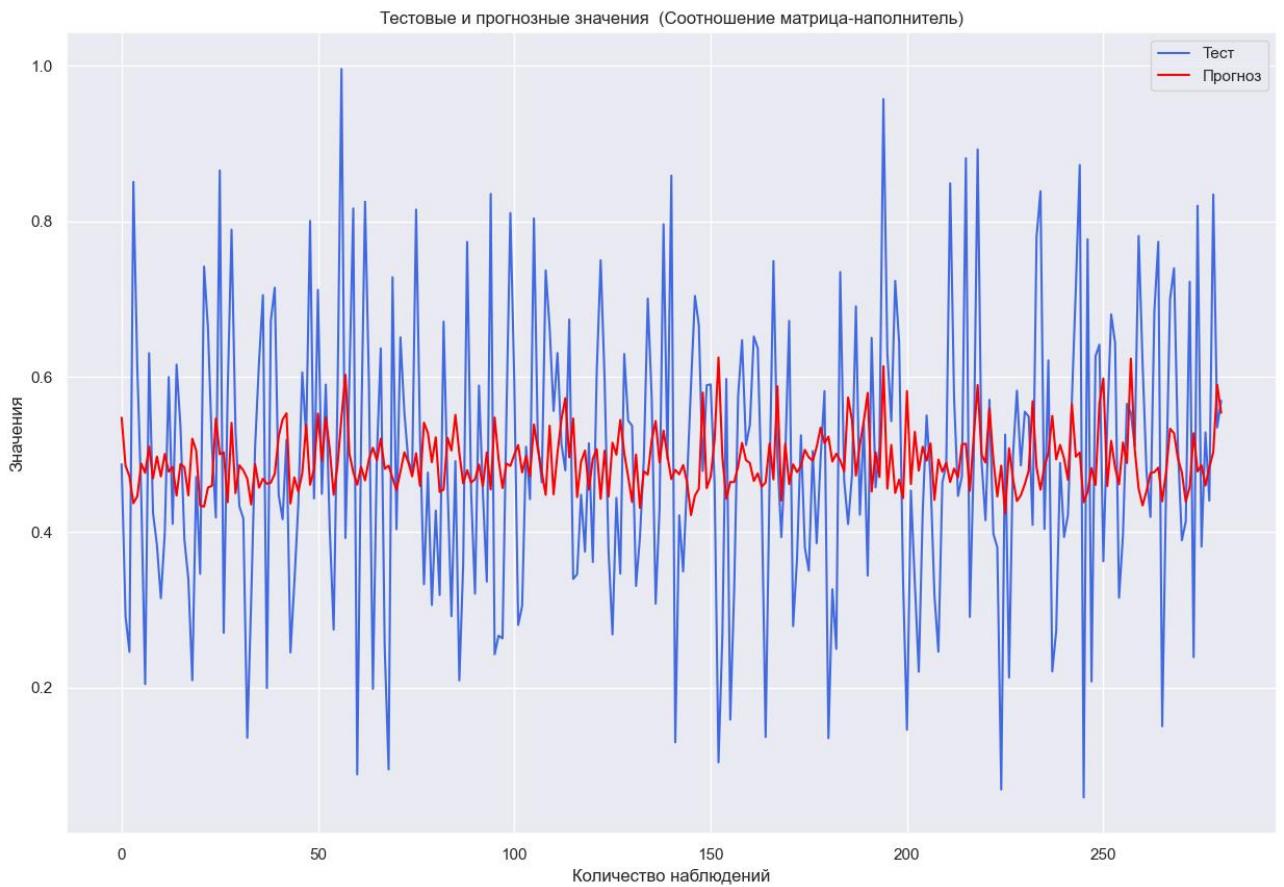


Рисунок 30 – График прогнозных значений второй нейросети

Модели последовательных нейронных сетей, созданных с помощью библиотеки Keras показали неудовлетворительный результат. Коэффициенты детерминации, имеющие значения близкие к нулю, говорят о том, что результат использования нейронных сетей не точнее использования для прогноза среднего значения прогнозируемого параметра.

Таблица 3 - Результаты работы нейронных сетей для предсказания

Соотношение матрица-наполнитель				
Нейронная сеть	MAE	MSE	R ²	R ² (train)
Последовательная нейросеть (Keras)	0.149729	0.033838	-0.009423	0.062134
Последовательная нейросеть с Dropout (Keras)	0.150222	0.035252	-0.051625	0.087347

2.5 Разработка приложения

Веб-приложение для прогноза соотношения «Матрица-наполнитель». написано на языке программирования Python использованием библиотеки Flask, для написания шаблонов страниц был использован язык разметки HTML.

Для запуска приложения необходимо:

- 1) Запустить среду разработки PyCharm;
- 2) Запустить приложение командой «python app.py»;
- 3) Перейти по сгенерированной ссылке;
- 4) Нажать на кнопку «Спрогнозировать значение матрица-наполнитель»;
- 5) Ввести значение для каждого параметра;
- 6) Нажать кнопку «Рассчитать».

2.6 Создание удаленного репозитория и загрузка результатов работы на него

Адрес репозитория на Github (с загруженным ноутбуком и файлами работы):
https://github.com/bqm7/VKR_Data

Заключение

Цель проекта заключалась в создании моделей прогнозирования, которые помогут сократить количество проводимых испытаний и пополнить базу данных материалов. В ходе данной работы был решён ряд задач, связанных с обработкой и анализом данных.

Была проведена предобработка данных, включающая проверку на выбросы и пропуски, а также нормализацию. Разведочный анализ показал, что распределение данных в объединенном датасете близко к нормальному, однако коэффициенты корреляции между параметрами очень низкие, и очевидные зависимости, которые могли бы помочь в построении эффективной модели, отсутствуют. Были применены и обучены модели машинного обучения с разными параметрами, протестировано несколько вариантов настроек нейронных сетей, а также создано веб-приложение для прогноза «соотношения матрица-наполнитель».

Базовые модели, простой подбор гиперпараметров и базовые модели нейросети не показали высокой точности в прогнозировании целевых результатов. Возможно, решение лежит через получение дополнительных данных или через глубокое изучение взаимосвязей между параметрами и поиск сложных закономерностей.

Список использованной литературы

- 1) Андерсон, Карл. Аналитическая культура. От сбора данных до бизнес-результатов / Карл Андерсон ; пер. с англ. Юлии Константиновой, 2017. — 336 с.
- 2) Андрей Бурков. Инженерия машинного обучения / пер. с англ. А.А. Слинкина. – М.: ДМК Пресс, 2022 — 306 с.
- 3) Глубокое обучение. Самый краткий и понятный курс / Джон Д. Келлехер; / пер. с англ. М.А. Райтман – Москва: Эксмо, 2022 – 160 с.
- 4) Библиотека Keras - инструмент глубокого обучения. Реализация нейронных сетей с помощью библиотек Theano и TensorFlow / пер. с англ. Слинкин А. А. - М.: ДМК Пресс, 2018. - 294 с.
- 5) Силен Дэви, Мейсман Арно, Али Мохамед. Основы Data Science и Big Data. Python и наука о данных. – СПб.: Питер, 2017. – 336 с.: ил.
- 6) Язык программирования Python – Режим доступа:
<https://www.python.org/>. (дата обращения 03.03.2023)
- 7) Библиотека Pandas – Режим доступа: <https://pandas.pydata.org/> (дата обращения 03.03.2023)
- 8) Библиотека NumPy – Режим доступа: <https://numpy.org/> (дата обращения: 03.03.2023).
- 9) Библиотека Matplotlib – Режим доступа: <https://matplotlib.org/>. (дата обращения 18.03.2023)
- 10) Библиотека Seaborn – Режим доступа: <https://seaborn.pydata.org/>. (дата обращения 03.03.2023)
- 11) Библиотека Sklearn – Режим доступа: <https://scikit-learn.org/stable/> (дата обращения 03.03.2023)
- 12) Библиотека Tensorflow - Режим доступа: <https://www.tensorflow.org/>. (дата обращения 03.03.2023)