

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»**

Слушатель

Лапшин Олег Васильевич

Москва, 2022

## Содержание

Введение	3
1 Аналитическая часть	5
1.1 Постановка задачи	5
1.2 Описание используемых методов	13
1.2.1 Линейная регрессия	13
1.2.2 Гребневая (Ridge) регрессия	14
1.2.3 Метод k-ближайших соседей	15
1.2.4 Деревья решений	15
1.2.5 Градиентный бустинг	16
1.2.6 Нейронная сеть	17
1.3 Разведочный анализ данных	18
1.3.1 План решения задачи	19
1.3.2 Препроцессинг	20
1.3.3 Перекрестная проверка	21
1.3.4 Метрики качества моделей	21
2 Практическая часть	22
2.1 Разбиение и предобработка данных	22
2.1.1 Для прогнозирования модуля упругости при растяжении	22
2.2 Разработка и обучение моделей для прогнозирования	23
2.3 Разработка нейронной сети для прогнозирования соотношения матрица-наполнитель	25
2.4. Разработка приложения	27
2.5. Создание удаленного репозитория	28
Заключение	29
Библиографический список	31
Приложение А	33

## Введение

Тема данной работы - Прогнозирование конечных свойств новых материалов (композиционных материалов).

Композиционными называются материалы, в которых имеет место сочетание двух (или более) химически разнородных компонентов (фаз) с четкой границей раздела между ними. Это неоднородные по химическому составу и структуре материалы.

Структура композиционных материалов представляет собой матрицу (основной компонент), содержащую в своем объеме или армирующие элементы, часто называемые наполнителем. Матрица и наполнитель разделены границей (поверхностью) раздела. Наполнитель равномерно распределен в матрице и имеет заданную пространственную ориентацию.

Композиционные материалы характеризуются совокупностью свойств, не присущих каждому в отдельности взятому компоненту. За счет выбора армирующих элементов, варьирования их объемной доли в матричном материале, а также размеров, формы, ориентации и прочности связи по границе «матрица-наполнитель», свойства композиционных материалов можно регулировать в значительных пределах.

Возможно получить композиты с уникальными эксплуатационными свойствами. Этим обусловлено широкое применение композиционных материалов в различных областях техники. Области применения композиционных материалов многочисленны; кроме авиационно-космической, ракетной и других специальных отраслей техники, композиционные материалы могут быть успешно применены в энергетическом турбостроении, в автомобильной промышленности - для деталей двигателей и кузовов автомашин; в машиностроении - для корпусов и деталей машин; в химической промышленности - для автоклавов, цистерн, аппаратов сернокислотного производства, ёмкостей для хранения и перевозки нефтепродуктов и др.

Учитывая такое широкое распространение и высокую потребность в новых материалах, тема данной работы является очень актуальной.

Стоимость производства композитного материала высока. Зная характеристики компонентов, невозможно рассчитать свойства композита. Значит для получения заданных свойств требуется большое количество испытаний различных комбинаций. Сократить время и затраты на создание определенного материала могла бы помочь система поддержки производственных решений, построенная на принципах машинного обучения.

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

## 1 Аналитическая часть

### 1.1 Постановка задачи

От специалистов в заданной предметной области был получен набор данных далее по тексту именуется как датасет, содержащий данные о производственных параметрах и свойствах некоторого композита. Требуется обучить несколько моделей для прогноза модуля упругости при растяжении и прочности при растяжении. При построении моделей необходимо 30% данных оставить на тестирование модели, а на остальных 70% данных производить обучение моделей. Также при построении моделей необходимо провести поиск гиперпараметров модели с помощью поиска по сетке с перекрестной проверкой, количество блоков равно десяти.

Написать нейронную сеть, которая будет рекомендовать соотношение матрица-наполнитель.

Так же требуется разработать приложение, делающее удобным использование данных моделей специалистом предметной области.

Датасет состоит из двух файлов: X\_br.xlsx и X\_nup.xlsx.

Файл X\_br.xlsx имеет размерность (1023, 10) и содержит: десять параметров, индекс и одну тысячу двадцать три строки.

Файл X\_nup.xlsx имеет размерность (1040, 3) и содержит: три параметра, индекс и одну тысячу сорок строк.

По условиям задания файлы требуют объединения с типом INNER по индексу.

После преобразования вышеуказанных файлов в pandas DataFrame объединения часть семнадцать строк (менее 2%) из файла X\_nup.xlsx была отброшена.

Дальнейшие исследования проводим с объединенным датасетом, содержащим тринадцать параметров и одну тысячу двадцать три строки.

Описание признаков объединенного датасета приведено в таблице 1.

Все признаки имеют тип float64, то есть вещественный. Пропусков в данных нет. Все признаки, кроме “Угол нашивки”, являются непрерывными, количественными. “Угол нашивки” принимает только два значения и в идеале должен рассматриваться как категориальный признак.

Таблица 1 Описание признаков датасета

Название	Тип данных	Непустые значения	Уникальные значения
Соотношение матрица-наполнитель	float64	1023	1014
Плотность, кг/м3	float64	1023	1013
модуль упругости, ГПа	float64	1023	1020
Количество отвердителя, м.%	float64	1023	1005
Содержание эпоксидных групп,%_2	float64	1023	1004
Температура вспышки, С_2	float64	1023	1003
Поверхностная плотность, г/м2	float64	1023	1004
Модуль упругости при растяжении, ГПа	float64	1023	1004
Прочность при растяжении, МПа	float64	1023	1004
Потребление смолы, г/м2	float64	1023	1003
Угол нашивки, град	float64	1023	2
Шаг нашивки	float64	1023	989
Плотность нашивки	float64	1023	988

Гистограммы распределения переменных и диаграммы “ящик с усами” приведены на рисунках 1 и 2. По ним видно, что все признаки, кроме “Угол нашивки”, имеют нормальное распределение и принимают неотрицательные значения. “Угол нашивки” принимает значения: 0, 90.

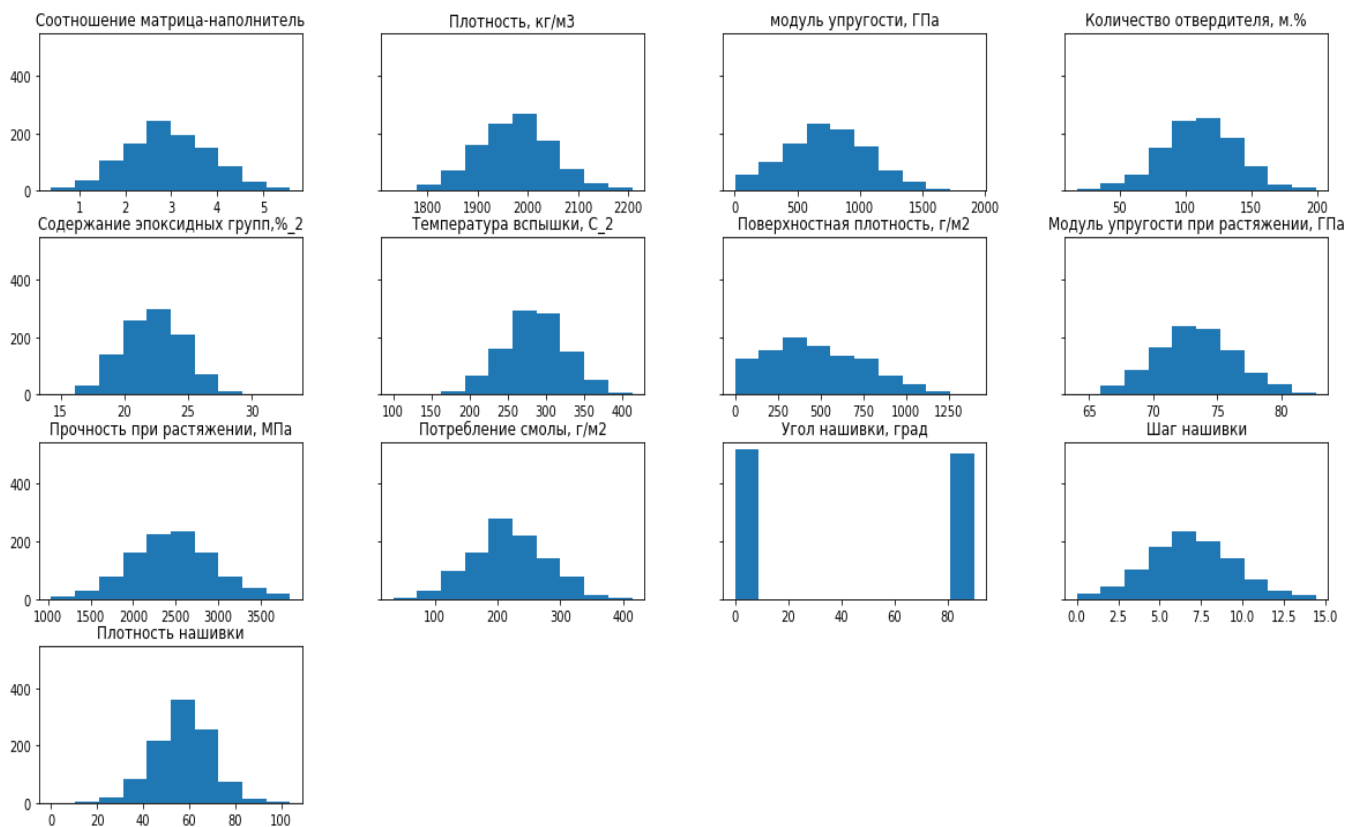


Рисунок 1 Гистограммы распределения переменных

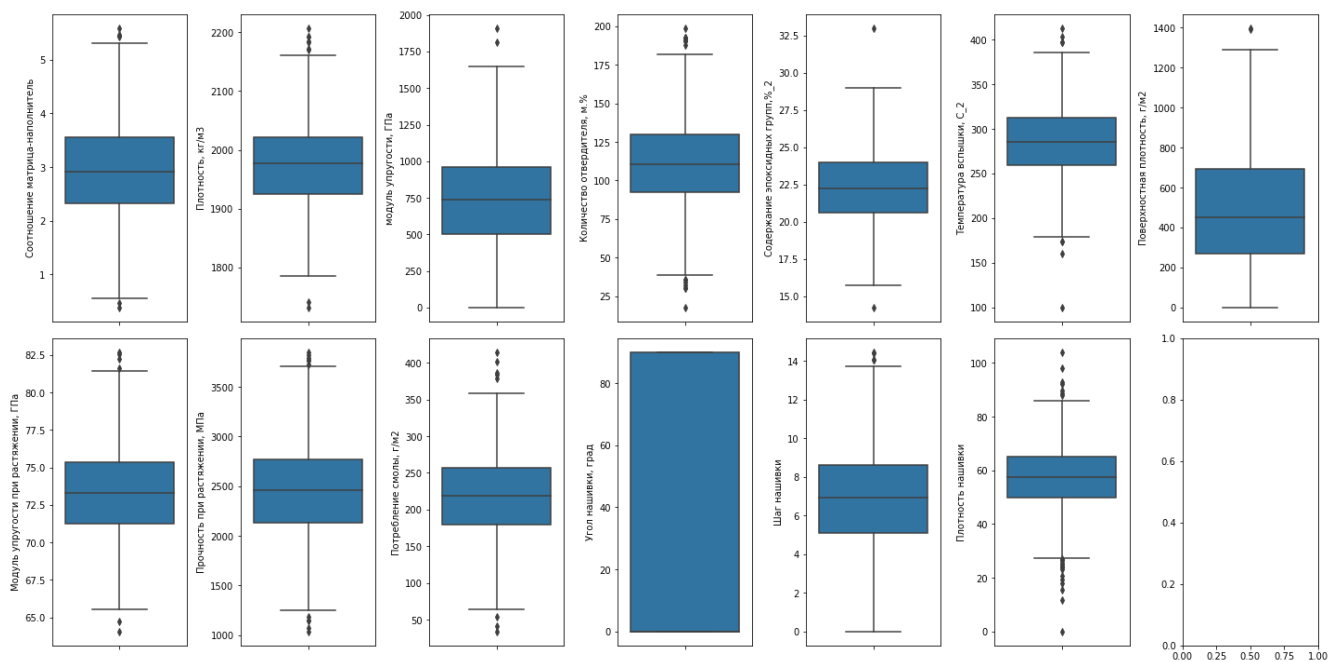


Рисунок 2 “Ящик с усами”

Отличительной чертой датасета является отсутствие пропусков, что может “сказать” о первоначальной обработке данных и их подготовке. В сырых данных пропуски и значения некорректных типов как правило присутствуют.

Так же нас интересует описательная статистика датасета. Она представлена в таблице 2. Она в численном виде отражает то, что мы видим на гистограммах.

Попарные графики рассеяния точек приведены на рисунке 3 и 4.

По графикам рассеяния мы видим, что некоторые точки отстоят далеко от общего облака. Так визуально выглядят выбросы - аномальные, некорректные значения данных, выходящие за пределы допустимых значений признака.

Таблица 2 Описательная статистика признаков датасета

	mean	std	min	max	median
Соотношение матрица-наполнитель	2.9304	0.9132	0.3894	5.5917	2.9069
Плотность, кг/м3	1975.73	73.7292	1731.76	2207.77	1977.62
модуль упругости, ГПа	739.923	330.231	2.4369	1911.53	739.664
Количество отвердителя, м.%	110.570	28.2959	17.7403	198.953	110.564
Содержание эпоксидных групп,%_2	22.2444	2.4063	14.2550	33.0000	22.2307
Температура вспышки, С_2	285.882	40.9433	100.000	413.273	285.896
Поверхностная плотность, г/м2	482.731	281.314	0.6037	1399.54	451.864
Модуль упругости при растяжении, ГПа	73.3286	3.1190	64.0541	82.6821	73.2688
Прочность при растяжении, МПа	2466.92	485.628	1036.85	3848.43	2459.52
Потребление смолы, г/м2	218.423	59.7359	33.8030	414.590	219.198



Угол нашивки, град	44.2522	45.0158	0.0000	90.0000	0.0000
Шаг нашивки	6.8992	2.5635	0.0000	14.4405	6.9161
Плотность нашивки	57.1539	12.3510	0.0000	103.988	57.3419

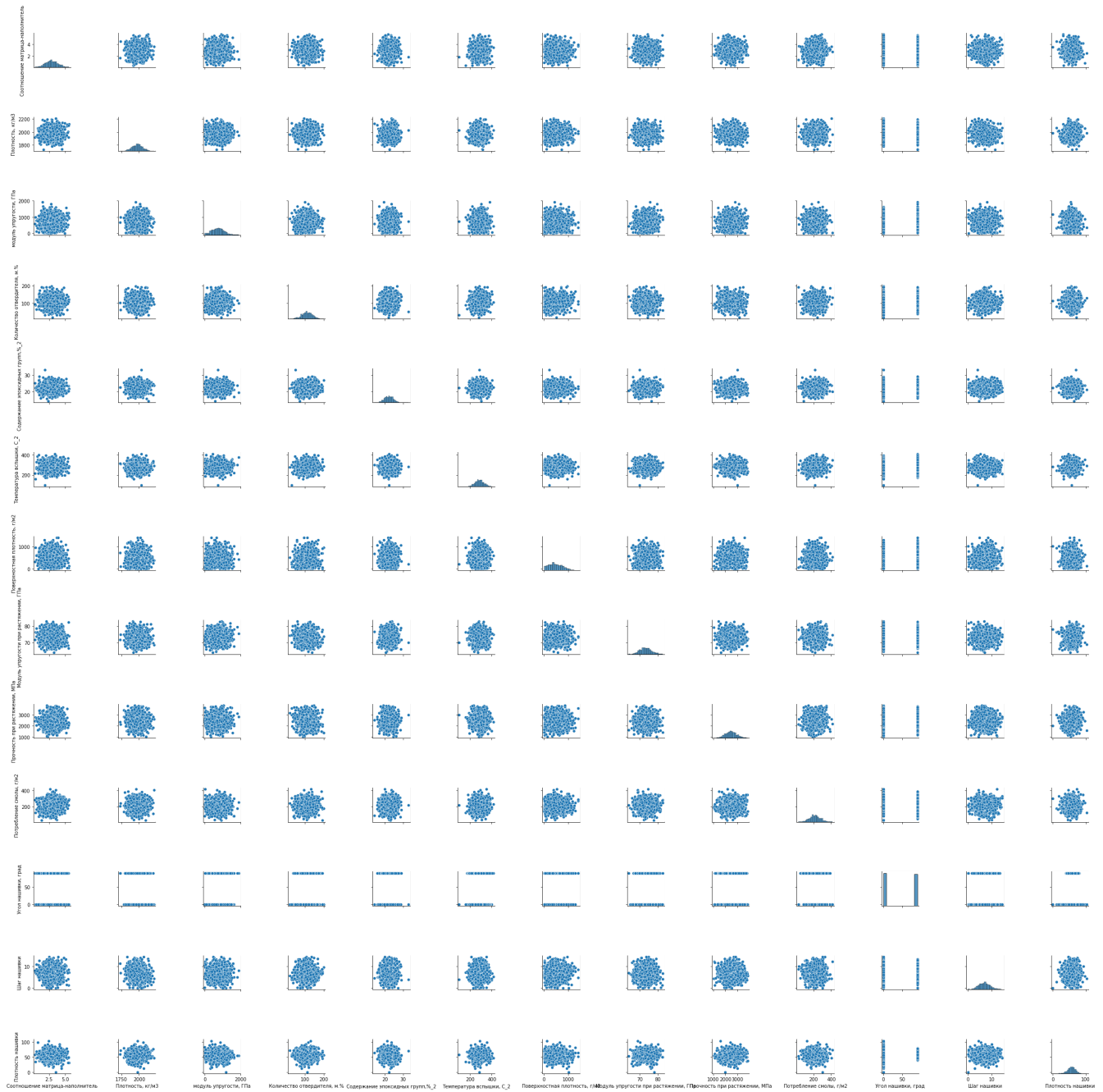
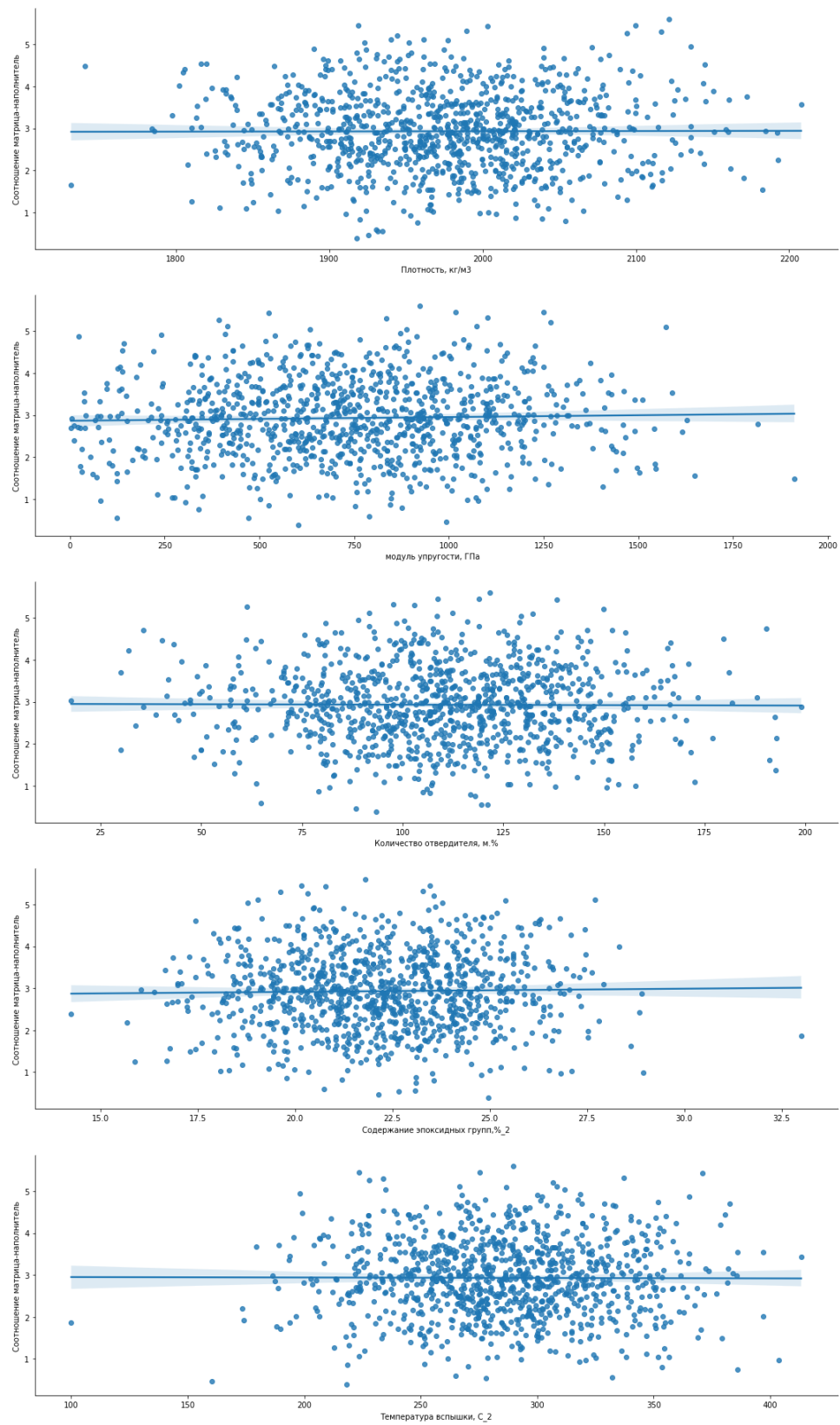
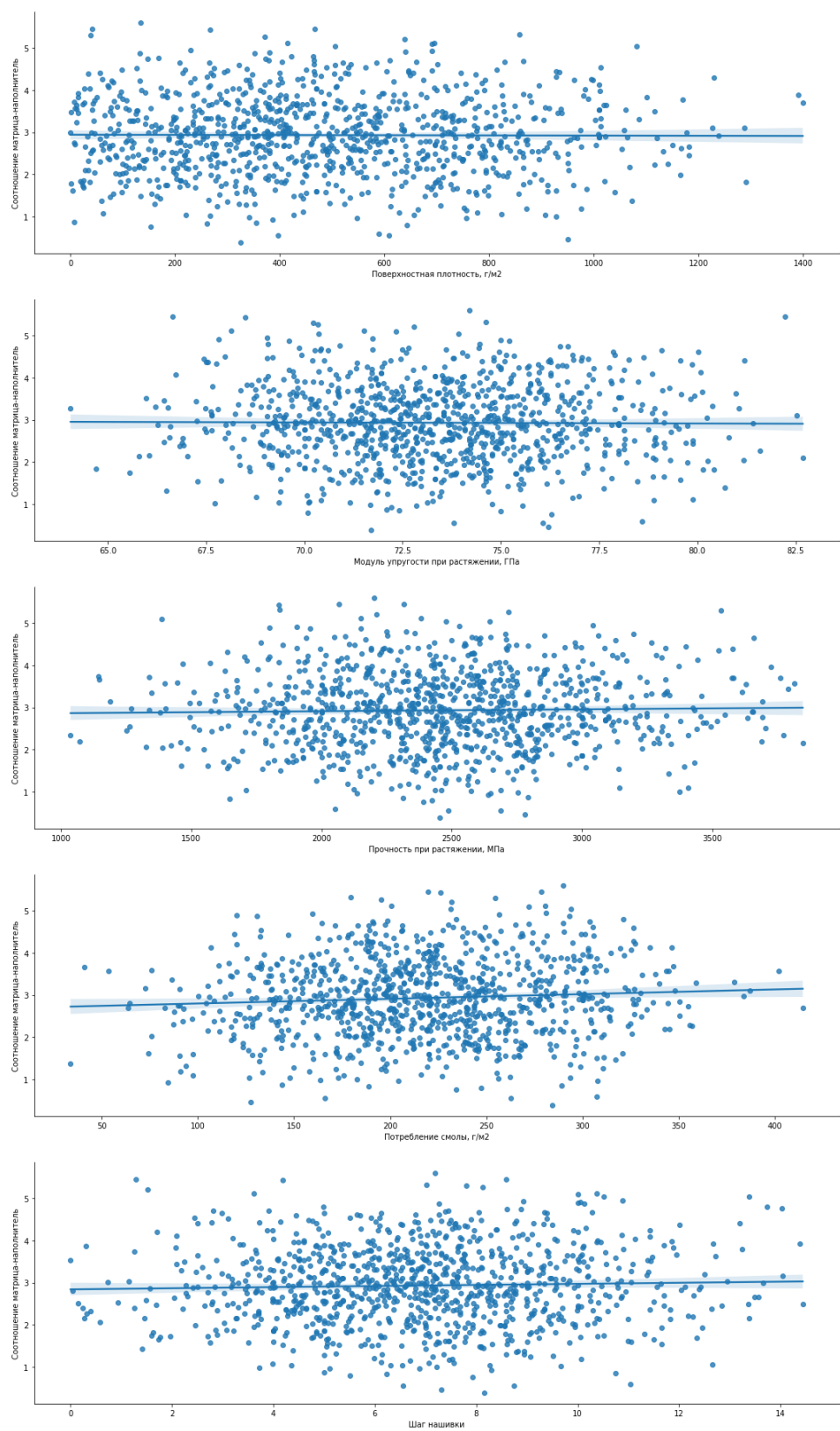


Рисунок 3 Попарные графики рассеяния точек





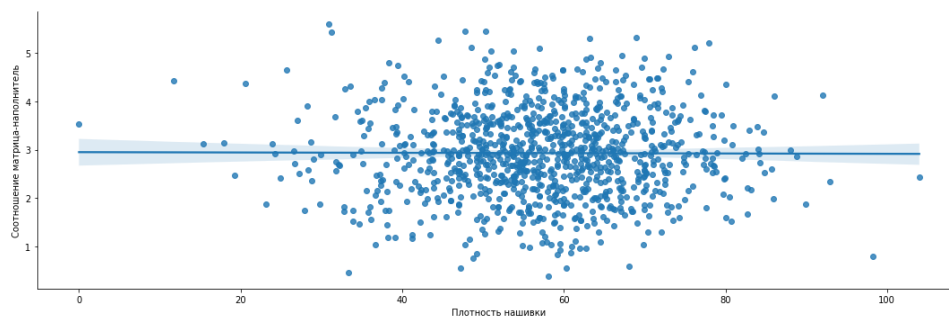


Рисунок 4 Попарные графики рассеяния точек

Применив метод межквартильных расстояний для удаления выбросов в параметрах с нормальным распределением, было обнаружено:

"Соотношение матрица-наполнитель" кол-во выбросов = 0.59%

"Плотность, кг/м<sup>3</sup>" кол-во выбросов = 0.88%

"модуль упругости, ГПа" кол-во выбросов = 0.20%

"Количество отвердителя, м.%" кол-во выбросов = 1.37%

"Содержание эпоксидных групп, %\_2" кол-во выбросов = 0.20%

"Температура вспышки, С\_2" кол-во выбросов = 0.78%

"Поверхностная плотность, г/м<sup>2</sup>" кол-во выбросов = 0.20%

"Модуль упругости при растяжении, ГПа" кол-во выбросов = 0.59%

"Прочность при растяжении, МПа" кол-во выбросов = 1.08%

"Потребление смолы, г/м<sup>2</sup>" кол-во выбросов = 0.78%

"Угол нашивки, град" кол-во выбросов = 0.00%

"Шаг нашивки" кол-во выбросов = 0.39%

"Плотность нашивки" кол-во выбросов = 2.05%

и удалено 69 выбросов.

После этого осталось в датасете осталось 954 строк и 13 параметров график распределения отображен на графике 5

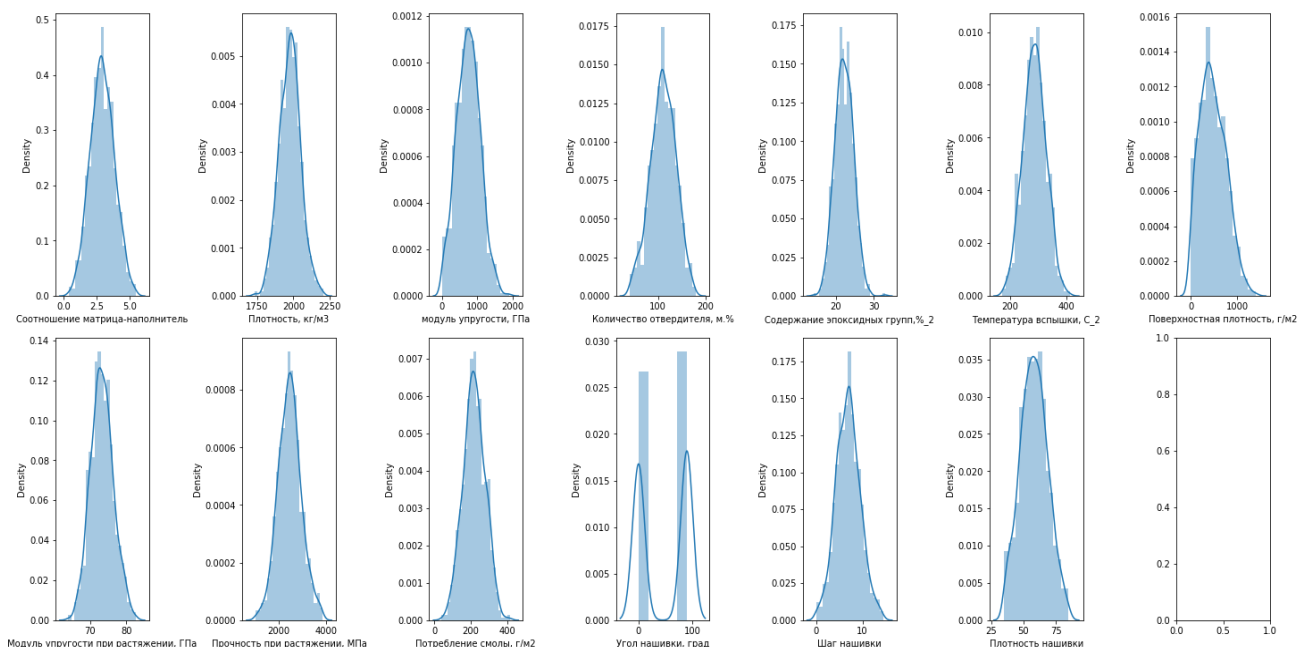


График 5 График распределения

По условиям задания нашими целевыми переменными являются:

- Модуль упругости при растяжении, ГПа;
- Прочность при растяжении, МПа;
- Соотношение матрица-наполнитель.

## 1.2 Описание используемых методов

Предсказание значений вещественной, непрерывной переменной - это задача регрессии. Эта зависимая переменная должна иметь связь с одной или несколькими независимыми переменными, называемых также предикторами или регрессорами. Регрессионный анализ помогает понять, как “типичное” значение зависимой переменной изменяется при изменении независимых переменных.

В настоящее время разработано много методов регрессионного анализа. Например, простая и множественная линейная регрессия. Эти модели являются параметрическими в том смысле, что функция регрессии определяется конечным числом неизвестных параметров, которые оцениваются на основе данных.

### 1.2.1 Линейная регрессия

Простая линейная регрессия имеет место, если рассматривается зависимость между одной входной и одной выходной переменными. Для этого определяется уравнение регрессии (1) и строится соответствующая прямая, известная как линия регрессии.

$$y=ax+by=ax+b \quad (1)$$

Коэффициенты  $a$  и  $b$ , называемые также параметрами модели, определяются таким образом, чтобы сумма квадратов отклонений точек, соответствующих реальным наблюдениям данных, от линии регрессии была бы минимальной. Коэффициенты обычно оцениваются методом наименьших квадратов.

Если ищется зависимость между несколькими входными и одной выходной переменными, то имеет место множественная линейная регрессия.

Соответствующее уравнение имеет вид (2).

$$Y=b_0+b_1*x_1+b_2*x_2+\dots+b_n*x_n \quad (2)$$

где  $n$  - число входных переменных.

Линейная регрессия - первый тщательно изученный метод регрессионного анализа. Его главное достоинство - простота. Такую модель можно построить и рассчитать даже без мощных вычислительных средств. Простота является и главным недостатком этого метода. Тем не менее, именно с линейной регрессии целесообразно начать подбор подходящей модели.

На языке python линейная регрессия реализована в `sklearn.linear_model.LinearRegression`.

### 1.2.2 Гребневая (Ridge) регрессия

Ридж-регрессия или гребневая регрессия - это один из методов понижения размерности. Часто его применяют для борьбы с переизбыточностью данных, когда независимые переменные коррелируют друг с другом (т.е. имеет место мультиколлинеарность).

Следствием этого является плохая обусловленность матрицы и неустойчивость оценок коэффициентов регрессии. Оценки, например, могут иметь неправильный знак или значения, которые намного превосходят те, которые приемлемы из физических или практических соображений.

Применение гребневой регрессии нередко оправдывают тем, что это практический приём, с помощью которого при желании можно получить меньшее значение среднего квадрата ошибки.

Метод стоит использовать, если:

- сильная обусловленность;
- сильно различаются собственные значения или некоторые из них близки к нулю;
- в матрице есть почти линейно зависимые столбцы.

Эти методы реализованы в `sklearn.linear_model.Ridge`.

### 1.2.3 Метод k-ближайших соседей

Это метод классификации, который адаптирован для регрессии - метод k-ближайших соседей (k Nearest Neighbors). Он относит объекты к классу, которому принадлежит большинство из k его ближайших соседей в многомерном пространстве признаков.

Число k - это количество соседних объектов в пространстве признаков, которые сравниваются с классифицируемым объектом. Иными словами, если k=10, то каждый объект сравнивается с 10-ю соседями. Метод широко применяется в технологиях Data Mining.

В процессе обучения алгоритм просто запоминает все векторы признаков и соответствующие им метки классов. При работе с реальными данными, т.е. наблюдениями, метки класса которых неизвестны, вычисляется расстояние между вектором нового наблюдения и ранее запомненными.

Затем выбирается k ближайших к нему векторов, и новый объект относится к классу, которому принадлежит большинство из них.

Несмотря на свою относительную алгоритмическую простоту, метод показывает хорошие результаты. Главным его недостатком является высокая вычислительная трудоемкость, которая увеличивается квадратично с ростом числа обучающих примеров.

Этот метод - пример непараметрической регрессии.

Он реализован в `sklearn.neighbors.KNeighborsRegressor`.

### 1.2.4 Деревья решений

Деревья решений (Decision Trees) - еще один непараметрический метод, применяемый и для классификации, и для регрессии. Деревья решений используются в самых разных областях человеческой деятельности и представляют собой иерархические древовидные структуры, состоящие из правил вида «Если ..., то ...».

Решающие правила автоматически генерируются в процессе обучения на обучающем множестве путем обобщения обучающих примеров. Поэтому их называют индуктивными правилами, а сам процесс обучения — индукцией деревьев решений.

Дерево состоит из элементов двух типов: узлов (node) и листьев (leaf).

В узлах находятся решающие правила и производится проверка соответствия примеров этому правилу. В результате проверки множество примеров, попавших в узел, разбивается на два подмножества: удовлетворяющие правилу и не удовлетворяющие ему. Затем к каждому подмножеству вновь

применяется правило и процедура рекурсивно повторяется пока не будет достигнуто некоторое условие остановки алгоритма. В последнем узле проверка и разбиение не производится и он объявляется листом.

В листе содержится не правило, а подмножество объектов, удовлетворяющих всем правилам ветви, которая заканчивается данным листом.

Для классификации - это класс, ассоциируемый с узлом, а для регрессии - соответствующий листу интервал целевой переменной.

При формировании правила для разбиения в очередном узле дерева необходимо выбрать атрибут, по которому это будет сделано. Общее правило для классификации можно сформулировать так: выбранный атрибут должен разбить множество наблюдений в узле так, чтобы результирующие подмножества содержали примеры с одинаковыми метками класса, а количество объектов из других классов в каждом из этих множеств было как можно меньше. Для этого были выбраны различные критерии, например, теоретико-информационный и статистический.

Для регрессии критерием является дисперсия вокруг среднего. Минимизируя дисперсию вокруг среднего, мы ищем признаки, разбивающие выборку таким образом, что значения целевого признака в каждом листе примерно равны.

Огромное преимущество деревьев решений в том, что они легко интерпретируемы, понятны человеку. Они могут использоваться для извлечения правил на естественном языке. Еще преимущества — высокая точность работы, нетребовательность к подготовке данных.

Недостаток деревьев решений - склонность переобучаться. Переобучение в случае дерева решений - это точное распознавание примеров, участвующих в обучении, и полная несостоятельность на новых данных. В худшем случае, дерево будет большой глубины и сложной структуры, а в каждом листе будет только один объект. Для решения этой проблемы используют разные критерии остановки алгоритма.

Деревья решений реализованы в `sklearn.tree.DecisionTreeRegressor`.

#### 1.2.5 Градиентный бустинг

Градиентный бустинг (GradientBoosting) - еще один представитель ансамблевых методов.

В отличие от случайного леса, где каждый базовый алгоритм строится независимо от остальных, бустинг воплощает идею последовательного построения линейной комбинации алгоритмов. Каждый следующий алгоритм старается уменьшить ошибку предыдущего.



Чтобы построить алгоритм градиентного бустинга, нам необходимо выбрать базовый алгоритм и функцию потерь или ошибки (loss). Loss-функция – это мера, которая показывает, насколько хорошо предсказание модели соответствует данным. Используя градиентный спуск и обновляя предсказания, основанные на скорости обучения (learning rate), ищем значения, на которых loss минимальна.

Бустинг, использующий деревья решений в качестве базовых алгоритмов, называется градиентным бустингом над решающими деревьями. Он отлично работает на выборках с “табличными”, неоднородными данными и способен эффективно находить нелинейные зависимости в данных различной природы. На настоящий момент это один из самых эффективных алгоритмов машинного обучения. Благодаря этому он широко применяется во многих конкурсах и промышленных задачах. Он проигрывает только нейросетям на однородных данных (изображения, звук и т. д.).

Из недостатков алгоритма можно отметить только затраты времени на вычисления и необходимость грамотного подбора гиперпараметров.

Реализация градиентного бустинга возможна из библиотеки `sklearn - sklearn.ensemble.GradientBoostingRegressor`.

### 1.2.6 Нейронная сеть

Нейронная сеть - это последовательность нейронов, соединенных между собой связями. Структура нейронной сети пришла в мир программирования из биологии. Вычислительная единица нейронной сети - нейрон или персептрон.

У каждого нейрона есть определённое количество входов, куда поступают сигналы, которые суммируются с учётом значимости (веса) каждого входа.

Смещение - это дополнительный вход для нейрона, который всегда равен 1 и, следовательно, имеет собственный вес соединения.

Так же у нейрона есть функция активации, которая определяет выходное значение нейрона. Она используется для того, чтобы ввести нелинейность в нейронную сеть. Примеры активационных функций: `relu`, `sigmoid`, `softmax` и т.п.

У полносвязной нейросети выход каждого нейрона подается на вход всем нейронам следующего слоя. У нейросети имеется:

- входной слой - его размер соответствует входным параметрам;

- скрытые слои - их количество и размерность определяем специалист;

- выходной слой - его размер соответствует выходным параметрам.

Прямое распространение - это процесс передачи входных значений в нейронную сеть и получения выходных данных, которые называются прогнозируемым значением.

Прогнозируемое значение сравниваем с фактическим с помощью функции потерь. В методе обратного распространения ошибки градиенты (производные

значений ошибок) вычисляются по значениям весов в направлении, обратном прямому распространению сигналов. Значение градиента вычитают из значения веса, чтобы уменьшить значение ошибки. Таким образом происходит процесс обучения. Обновляются веса каждого соединения, чтобы функция потерь минимизировалась.

Для обновления весов в модели используются различные оптимизаторы.

Количество эпох показывает, сколько раз выполнялся проход для всех примеров обучения.

Нейронные сети применяются для решения задач регрессии, классификации, распознавания образов и речи, компьютерного зрения и других. На настоящий момент это самый мощный, гибкий и широко применяемый инструмент в машинном обучении.

### 1.3 Разведочный анализ данных

Цель разведочного анализа данных - выявить закономерности в данных. Для корректной работы большинства моделей желательна сильная зависимость выходных переменных от входных и отсутствие зависимости между входными переменными.

На рисунках 3 и 4 графики попарного рассеяния точек. По форме “облаков точек” невозможно выявить зависимостей, которые станут основой работы моделей. Помочь выявить связь между признаками может матрица корреляции, приведенная на рисунке 6.

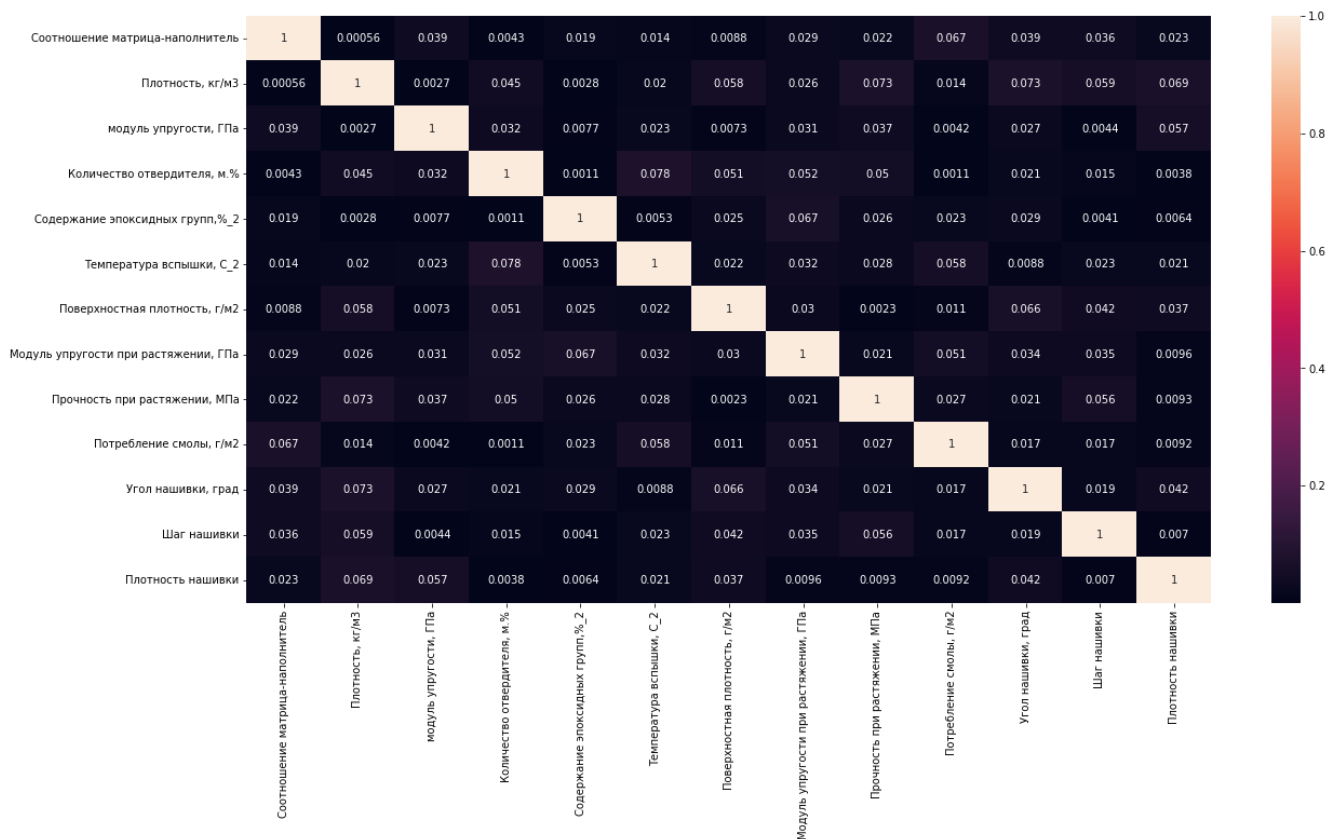


Рисунок 6 Тепловая карта корреляции

По тепловой карте корреляции можно увидеть, что все коэффициенты корреляции близки к нулю и означает отсутствие линейной зависимости между признаками.

Незначительными признаками корреляции обладают лишь:

- Температура вспышки, С\_2 - Количество отвердителя, м. %
- Плотность, кг/м3 - Прочность при растяжении, МПа
- Плотность, кг/м3 - Угол нашивки, град

### 1.3.1 План решения задачи

План решения каждой из задач и построения оптимальной модели будет следующим:

- разделить данные на тренировочную и тестовую выборки. В задании указано, что на тестирование оставить 30% данных;
- выполнить препроцессинг, то есть подготовку исходных данных;
- выбрать базовую модель для определения нижней границы качества предсказания. Использу базовую модель, возвращающую среднее

значение целевого признака. Лучшая модель по своим характеристикам должна быть лучше базовой;

- взять несколько моделей с гиперпараметрами по умолчанию, и используя перекрестную проверку, посмотреть их метрики на тренировочной выборке;
- подобрать для этих моделей гиперпараметры с помощью поиска по сетке с перекрестной проверкой, количество блоков равно 10;
- сравнить метрики моделей после подбора гиперпараметров и выбрать лучшую;
- получить предсказания лучшей и базовой моделей на тестовой выборке, сделать выводы;
- сравнить качество работы лучшей модели на тренировочной и тестовой выборке.

### 1.3.2 Препроцессинг

Цель препроцессинга, или предварительной обработки данных - обеспечить корректную работу моделей.

Его необходимо выполнять после разделения на тренировочную и тестовую выборку, как будто мы не знаем параметров тестовой выборки (минимум, максимум, матожидание, стандартное отклонение).

Препроцессинг для категориальных и количественных признаков выполняется по-разному.

Категориальный признак один - “Угол нашивки, град”. Он принимает значения 0 и 90. Модели отработают лучше, если мы превратим эти значения в 0 и 1 с помощью LabelEncoder или OrdinalEncoder.

Вещественных количественных признаков у нас большинство. Проблема вещественных признаков в том, что их значения лежат в разных диапазонах, в разных масштабах. Это видно в таблице 2. Необходимо провести одно из двух возможных преобразований:

- нормализацию - приведение в диапазон от 0 до 1 с помощью MinMaxScaler;
- стандартизацию - приведение к матожиданию 0, стандартному отклонению 1 с помощью StandartScaler.

В работе будет использована нормализация и MinMaxScaler.

Выходные переменные останутся не измененными.

### 1.3.3 Перекрестная проверка

Для обеспечения статистической устойчивости метрик модели используем перекрестную проверку или кросс-валидацию. Чтобы ее реализовать, выборка разбивается на необходимое количество раз на тестовую и валидационную. Модель обучается на тестовой выборке, затем выполняется расчет метрик качества на валидационной. В качестве результата мы получаем средние метрики качества для всех валидационных выборок. Перекрестную проверку реализует функция `cross_validate` из `sklearn`.

#### 1.3.4 Метрики качества моделей

Существует множество различных метрик качества, применимых для регрессии. В этой работе используется:

RMSE (Root Mean Squared Error) или корень из средней квадратичной ошибки принимает значения в тех же единицах, что и целевая переменная. Метрика использует возведение в квадрат, поэтому хорошо обнаруживает грубые ошибки, но сильно чувствительна к выбросам;

MAE (Mean Absolute Error) - средняя абсолютная ошибка так же принимает значения в тех же единицах, что и целевая переменная;

MAPE (Mean Absolute Percentage Error) или средняя абсолютная процентная ошибка — безразмерный показатель, представляющий собой взвешенную версию MAE.

## 2 Практическая часть

### 2.1 Разбиение и предобработка данных

#### 2.1.1 Для прогнозирования модуля упругости при растяжении

Признаки датасета были разделены на входные и выходные, а строки - на тренировочное и тестовое множество. Размерности полученных наборов данных показаны на рисунке 7. Описательная статистика входных признаков до и после предобработки показана на рисунке 8. Описательная статистика выходного признака показана на рисунке 9.

```
x_train: (667, 12) y_train: (667, 1)
x_test: (287, 12) y_test: (287, 1)
```

Рисунок 7 Размерности тренировочного и тестового множеств

	min	max	mean	std
0	0.389403	5.455566	2.930646	0.904203
1	1731.764635	2207.773481	1975.747669	73.820992
2	2.436909	1911.536477	743.484597	330.231615
3	40.304806	179.645962	110.688614	26.781546
4	14.254985	33.000000	22.208086	2.421878
5	173.484920	413.273418	286.559313	40.450441
6	0.603740	1399.542362	481.074070	278.814346
7	1036.856605	3848.436732	2467.771681	481.224022
8	41.048278	414.590628	218.328594	59.498348
9	0.000000	90.000000	46.792453	44.987872
10	0.037639	14.440522	6.917423	2.572487
11	35.005121	84.840888	58.059589	10.465658

	min	max	mean	std
0	0.031185	1.000000	0.505959	0.175854
1	0.000000	0.967488	0.511649	0.157760
2	0.000996	0.851482	0.389434	0.171242
3	0.000000	0.980428	0.508197	0.193462
4	0.000000	1.000000	0.426132	0.129633
5	0.002039	1.000000	0.474961	0.168977
6	0.000761	1.000000	0.346048	0.200115
7	0.000000	1.000000	0.503434	0.174262
8	0.060602	1.000000	0.477176	0.156139
9	0.000000	1.000000	0.515742	0.500127
10	0.000000	0.995552	0.470768	0.180465
11	0.000000	1.000000	0.473552	0.215424

Рисунок 8 Описательная статистика входных признаков до и после предобработки для задачи с параметром Модуль упругости при растяжении, ГПа

Аналогичным образом были подготовлены данные для параметров (признаков) “Прочность при растяжении, МПа” и “Соотношение матрица-наполнитель”.

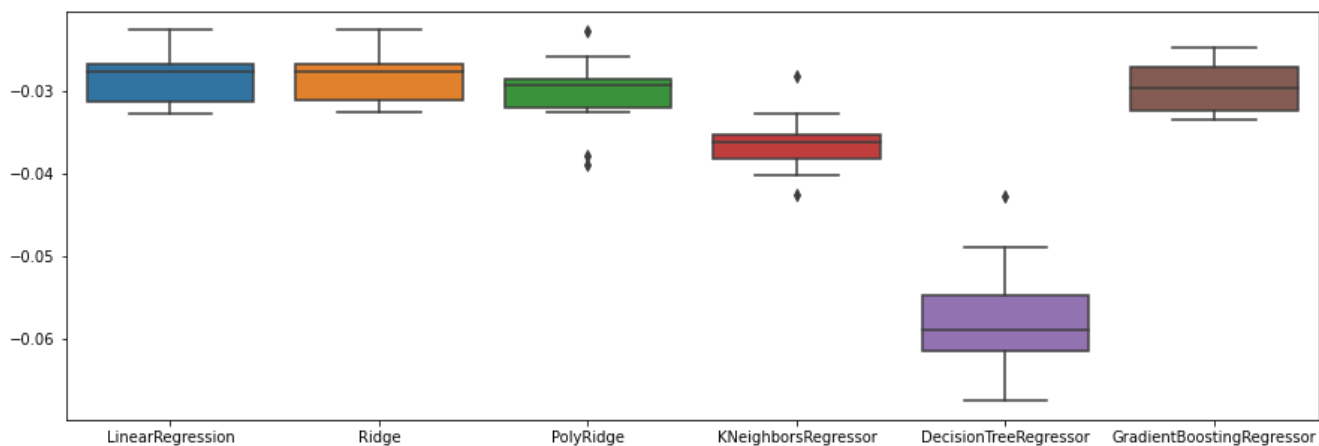
## 2.2 Разработка и обучение моделей для прогнозирования

Для подбора лучшей модели для выполнения этой задачи были выбраны следующие модели:

LinearRegression - линейная регрессия;  
Ridge - гребневая регрессия;  
KneighborsRegressor - метод ближайших соседей;  
DecisionTreeRegressor - деревья решений;  
GradientBoosting - градиентный бустинг,  
описанные в разделе 1.2.

Метрики работы выбранных моделей с гиперпараметрами по умолчанию, полученные на тестовом множестве, приведены на рисунках 9 и 10.

Ни одна из выбранных мной моделей не оказалась подходящей для предоставленных данных.



	LinearRegression	Ridge	PolyRidge	KNeighborsRegressor	DecisionTreeRegressor	GradientBoostingRegressor
0	-0.027368	-0.027314	-0.028680	-0.036453	-0.048984	-0.027950
1	-0.029411	-0.029362	-0.038978	-0.042571	-0.062243	-0.031969
2	-0.032681	-0.032670	-0.037773	-0.040280	-0.062273	-0.033579
3	-0.031824	-0.031780	-0.032665	-0.036422	-0.059673	-0.032422
4	-0.027056	-0.026993	-0.028455	-0.035373	-0.067604	-0.027373
5	-0.027973	-0.027918	-0.029938	-0.035367	-0.059443	-0.031220
6	-0.022654	-0.022637	-0.022830	-0.028270	-0.057153	-0.024795
7	-0.031897	-0.031843	-0.030194	-0.038765	-0.058517	-0.033444
8	-0.024399	-0.024430	-0.028706	-0.036104	-0.054112	-0.027095
9	-0.026571	-0.026562	-0.025804	-0.032760	-0.042749	-0.026685

Рисунок 9 Метрики работы выбранных моделей



Обычная линейная регрессия методом наименьших квадратов  
MAE: 0.13271867359870188  
MSE: 0.02794886397265417  
RMSE: 0.16717913737262247

Линейный метод наименьших квадратов с регуляризацией l2  
MAE: 0.13263628212128165  
MSE: 0.027938115050893025  
RMSE: 0.16714698636497466

PolyRidge  
MAE: 0.13543274634462302  
MSE: 0.028756498486995827  
RMSE: 0.16957741148807476

Regression based on k-nearest neighbors ...  
MAE: 0.15030780504373686  
MSE: 0.03414234603445326  
RMSE: 0.18477647586869184

Дерево решений ...  
MAE: 0.17860060477088957  
MSE: 0.05176681084076325  
RMSE: 0.22752320945513063

Gradient Boosting ...  
MAE: 0.13792449178645536  
MSE: 0.030205009756914706  
RMSE: 0.17379588532791768

Рисунок 10 Метрики работы выбранных моделей

## 2.3 Разработка нейронной сети для прогнозирования соотношения матрица-наполнитель

В задании для соотношения матрица-наполнитель необходимо построить нейросеть.

Строю нейронную сеть с помощью класса `keras.Sequential` со следующими параметрами:

- входной слой: 12;
- выходной слой: 1;
- скрытых слоев: 6;
- нейронов на каждом скрытом слое 64;
- активационная функция скрытых слоев: `relu`;
- оптимизатор: `Adam`;

- loss-функция: MeanAbsolutePercentageError, и архитектурой, отображенной на рисунке 11:

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 64)	832
dropout (Dropout)	(None, 64)	0
dense_1 (Dense)	(None, 64)	4160
dropout_1 (Dropout)	(None, 64)	0
dense_2 (Dense)	(None, 64)	4160
dropout_2 (Dropout)	(None, 64)	0
dense_3 (Dense)	(None, 64)	4160
dropout_3 (Dropout)	(None, 64)	0
dense_4 (Dense)	(None, 64)	4160
dropout_4 (Dropout)	(None, 64)	0
dense_5 (Dense)	(None, 64)	4160
dropout_5 (Dropout)	(None, 64)	0
dense_6 (Dense)	(None, 1)	65

=====  
Total params: 21,697  
Trainable params: 21,697  
Non-trainable params: 0  
=====

Рисунок 11 Архитектура нейронной сети

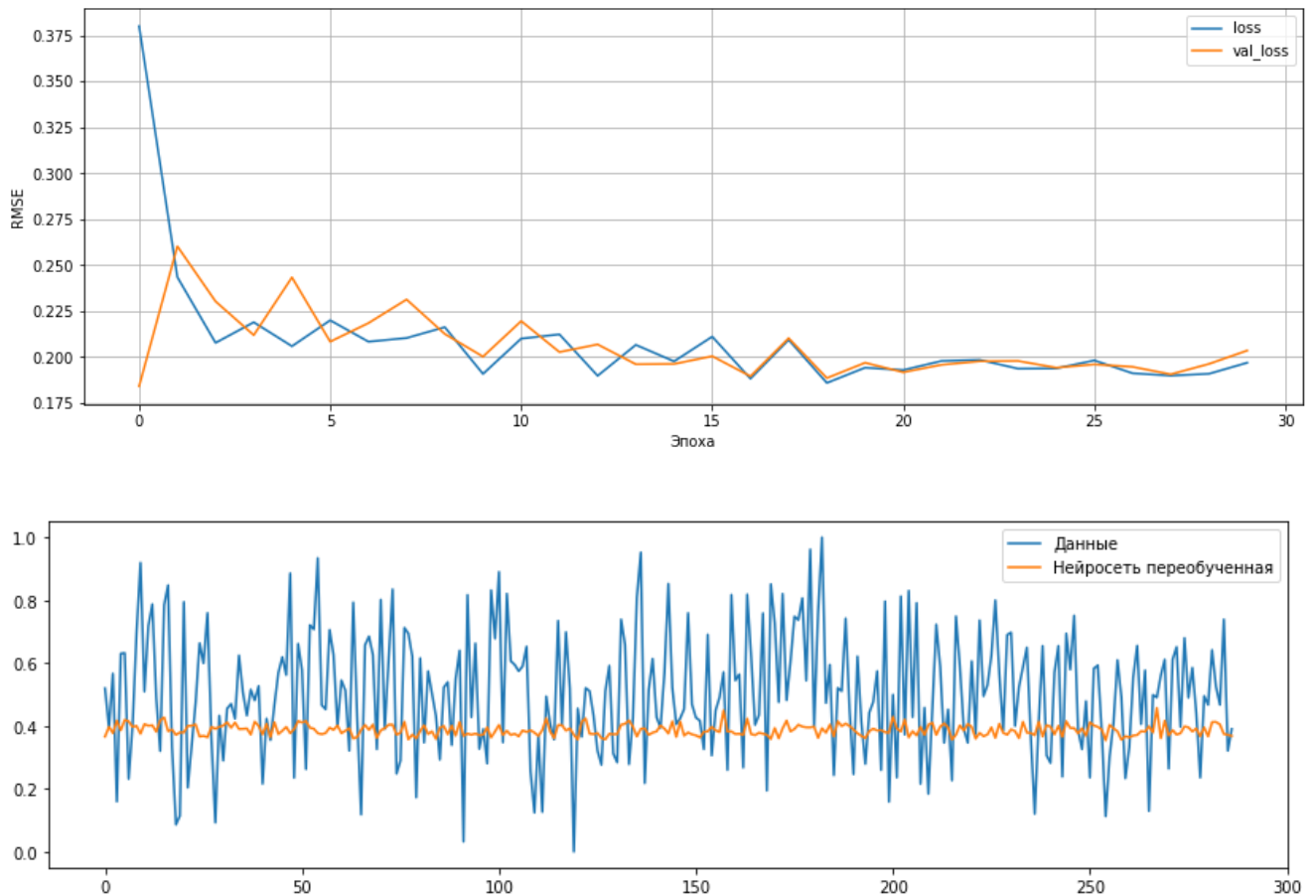


Рисунок 12 Работа нейронной сети

Визуализация результатов показывает, что нейросеть из библиотеки tensorflow старалась подстроиться к данным.

Для борьбы с переобучением были добавлены Dropout-слои с параметром 0.2.

#### 2.4. Разработка приложения

Несмотря на то, что пригодных к внедрению моделей получить не удалось, можно разработать функционал приложения. Возможно, дальнейшие исследования позволят построить качественную модель и внедрить ее в готовое приложение.

В приложении необходимо реализовать следующие функции:

- выбор целевой переменной для предсказания;
- ввод входных параметров;
- проверка введенных параметров;
- загрузка сохраненной модели, получение и отображение прогноза выходных параметров.

Веб-приложение реализовано с помощью языка Python и фреймворка Flask.

Скриншоты разработанного веб-приложения приведены в приложении А.

## 2.5. Создание удаленного репозитория

Для данного исследования был создан удаленный репозиторий на GitHub, который находится по адресу <https://github.com/olelap/FQW-DS-BMSTU-2022>. На него были загружены результаты работы: исследовательский notebook, код приложения.

## Заключение

В ходе выполнения данной работы я постарался пройти практически весь Dataflow pipeline, рассмотрели большую часть операций и задач, которые приходится выполнять специалисту по работе с данными.

Этот поток операций и задач включает:

- изучение теоретических методов анализа данных и машинного обучения;
- изучение основ предметной области, в которой решается задача;
- извлечение и трансформацию данных (был предоставлен готовый набор данных);
- проведение разведочного анализа данных статистическими методами;
- DataMining - извлечение признаков из датасета и их анализ;
- разделение имеющихся, в нашем случае размеченных, данных на обучающую и тестовую выборки;
- выполнение предобработки (препроцессинга) данных для обеспечения корректной работы моделей;
- построение аналитического решения. Это включает выбор алгоритма решения и модели, сравнение различных моделей, подбор гиперпараметров модели;
- визуализация модели и оценка качества аналитического решения;
- сохранение моделей;
- разработка и тестирование приложения для поддержки принятия решений специалистом предметной области, которое использовало бы найденную модель.

Внедрение решения и приложения в эксплуатацию пока затронуты не были.

Дальнейшие возможные пути решения этой задачи могли бы быть:

- углубиться в изучение нейросетей, попробовать различные архитектуры, параметры обучения и т.д.;
- провести отбор признаков разными методами. Испробовать методы уменьшения размерности, например метод главных компонент;
- после уменьшения размерности градиентный бустинг может улучшить свои результаты. Также есть большой простор для подбора гиперпараметров для этого метода;
- проконсультироваться у экспертов в предметной области. Возможно, они могли бы поделиться знаниями, необходимыми для решения задачи.



## Библиографический список

- 1 Композиционные материалы: учебное пособие для вузов / Д. А. Иванов, А. И. Ситников, С. Д. Шляпин ; под редакцией А. А. Ильина. — Москва : Издательство Юрайт, 2019 — 253 с. — (Высшее образование). — Текст : непосредственный.
- 2 Силен Дэви, Мейсман Арно, Али Мохамед. Основы Data Science и Big Data. Python и наука о данных. — СПб.: Питер, 2017. — 336 с.: ил.
- 3 ГрасД. Data Science. Наука о данных с нуля: Пер. с англ. - 2-е изд., перераб. и доп. - СПб.: БХВ-Петербург, 2021. - 416 с.: ил.
- 4 Документация по языку программирования python: — Режим доступа: <https://docs.python.org/3.8/index.html>.
- 5 Документация по библиотеке numpy: — Режим доступа: <https://numpy.org/doc/1.22/user/index.html#user>.
- 6 Документация по библиотеке pandas: — Режим доступа: [https://pandas.pydata.org/docs/user\\_guide/index.html#user-guide](https://pandas.pydata.org/docs/user_guide/index.html#user-guide).
- 7 Документация по библиотеке matplotlib: — Режим доступа: <https://matplotlib.org/stable/users/index.html>.
- 8 Документация по библиотеке seaborn: — Режим доступа: <https://seaborn.pydata.org/tutorial.html>.
- 9 Документация по библиотеке sklearn: — Режим доступа: [https://scikit-learn.org/stable/user\\_guide.html](https://scikit-learn.org/stable/user_guide.html).
- 10 Документация по библиотеке keras: — Режим доступа: <https://keras.io/api/>.
- 11 Руководство по быстрому старту в flask: — Режим доступа: <https://flask-russian-docs.readthedocs.io/ru/latest/quickstart.html>.
- 12 Loginom Вики. Алгоритмы: — Режим доступа: <https://wiki.loginom.ru/algorithms.html>.
- 13 Andre Ye. 5 алгоритмов регрессии в машинном обучении, о которых вам следует знать: — Режим доступа: <https://habr.com/ru/company/vk/blog/513842/>.
- 14 Alex Maszański. Метод k-ближайших соседей (k-nearest neighbour): — Режим доступа: <https://proglab.io/p/metod-k-blizhayshih-sosedey-k-nearest-neighbour-2021-07-19>.
- 15 Yury Kashnitsky. Открытый курс машинного обучения. Тема 3. Классификация, деревья решений и метод ближайших соседей: — Режим доступа: <https://habr.com/ru/company/ods/blog/322534/>.
- 16 Yury Kashnitsky. Открытый курс машинного обучения. Тема 5. Композиции: бэггинг, случайный лес: — Режим доступа: <https://habr.com/ru/company/ods/blog/324402/>.

17 Alex Maszański. Машинное обучение для начинающих: алгоритм случайного леса (Random Forest): – Режим доступа: <https://proglib.io/p/mashinnoe-obuchenie-dlya-nachinayushchih-algoritm-sluchaynogo-l-esa-random-forest-2021-08-12>.

18 Alex Maszański. Решаем задачи машинного обучения с помощью алгоритма градиентного бустинга: – Режим доступа: <https://proglib.io/p/reshaem-zadachi-mashinnogo-obucheniya-s-pomoshchyu-algoritma-gradientnogo-bustinga-2021-11-25>.

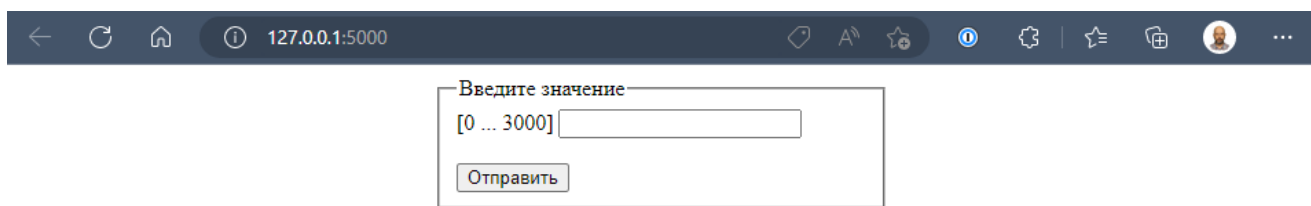


## Приложение А

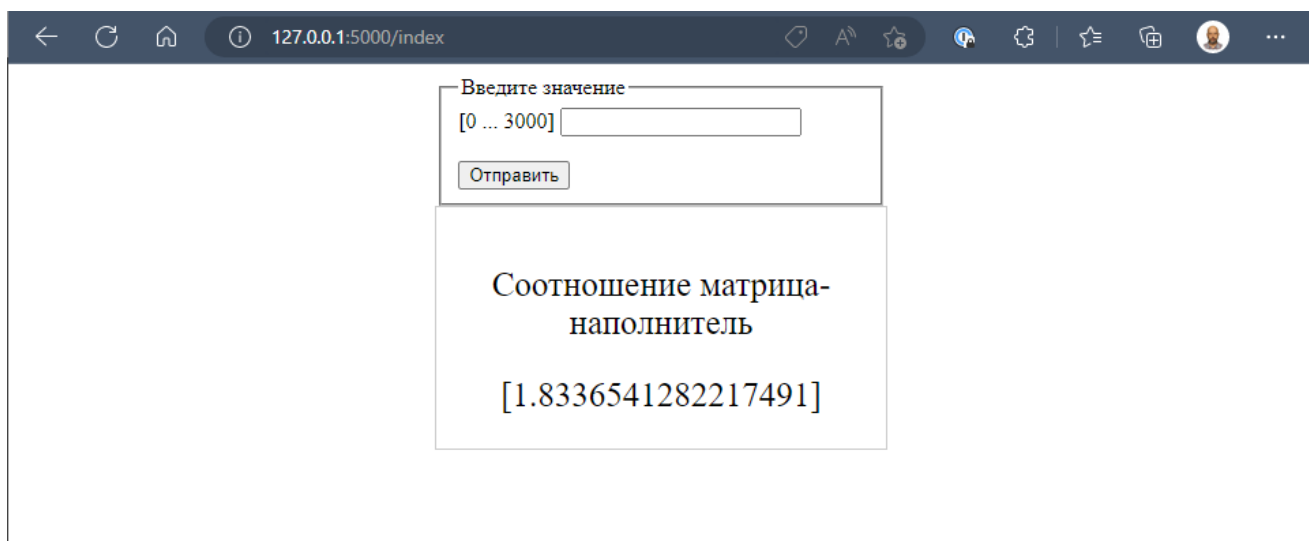
Скриншоты веб-приложения, иллюстрирующие его работу, приведены на рисунках.

Реализована функция выбора целевой переменной для предсказания соотношения матрица-наполнитель.

При проверке введенных параметров значения не могут быть пустыми, должны быть вещественными, не могут содержать некорректных символов и должны соответствовать допустимому диапазону.



A screenshot of a web browser window. The address bar shows the URL `127.0.0.1:5000`. The page content consists of a form with the label "Введите значение" (Enter value) above a text input field. To the left of the input field is the range "[0 ... 3000]". Below the input field is a button labeled "Отправить" (Send).



A screenshot of a web browser window. The address bar shows the URL `127.0.0.1:5000/index`. The page content shows the same input form as the previous screenshot. Below the form, there is a large box containing the text "Соотношение матрица-наполнитель" (Matrix-filler ratio) and the numerical value "[1.8336541282217491]".