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 Pro»

Слушатель Бобрович Татьяна Александровна

Ижевск, 2024

Содержание

Введение

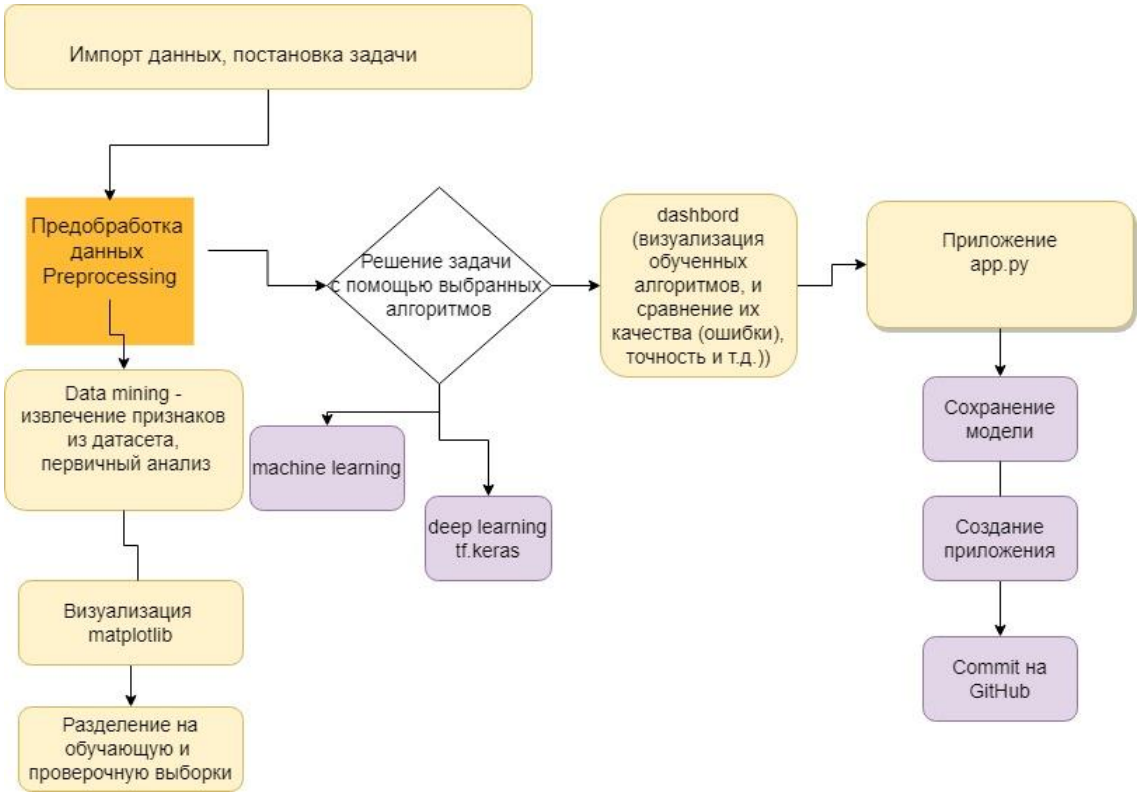
Тема: Прогнозирование конечных свойств новых материалов (композиционных материалов).

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. Современные композиты изготавливаются из разных материалов: полимеры, керамика, стеклянные и углеродные волокна, но данный принцип сохраняется. У такого подхода есть и недостаток: даже если мы знаем 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).

На входе имеются данные о начальных свойствах компонентов композиционных материалов (количество связующего, наполнителя, температурный режим отверждения и т.д.). На выходе необходимо спрогнозировать ряд конечных свойств получаемых композиционных материалов.

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

1 Блок-схема общего пайплайна работ



1. Аналитическая часть

1.1. Постановка задачи

В данной работе исследуется данные о характеристиках компонентов, используемых для получения композиционных материалов и о свойствах самого композиционного материала.

Требуется разработать модели, прогнозирующие значения некоторых свойств в зависимости от остальных.

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

Для решения поставленной задачи получен датасет, состоящий из двух файлов: X_bp и X_nup

Файл X_bp содержит:

– признаков: 10 и индекс;

– строк: 1023.

Файл X_nup содержит:

– признаков: 3 и индекс;

– строк: 1040.

Известно, что файлы требуют объединения с типом INNER по индексу. После объединения часть строк из файла X_nup была отброшена. И дальнейшие исследования проводим с объединенным датасетом, содержащим 13 признаков и 1023 строк или объектов.

Смотрим размерность методом shape. Видим, что в датасетах разное количество строк: 1023 и 1040.

Файлы требуют объединения с типом INNER по индексу. Тип объединения – INNER. Означает, что объединяются только те значения, которые можно найти в обеих таблицах.

После объединения часть строк из файла X_nup была отброшена. И дальнейшие исследования проводим с объединенным датасетом, содержащим 13 признаков и 1023 строк или объектов.

Описание признаков объединенного датасета приведено в таблице 1. Все признаки имеют тип float64, int64 то есть вещественный. Пропусков в данных нет. Всепризнаки, кроме «Угол нашивки», являются непрерывными, количественными.

«Угол нашивки» принимает только два значения и будет рассматриваться как категориальный признак.

Подсчет уникальных значений по столбцам методом data.nunique() показывает, что почти все значения уникальны.

Таблица 1 — Описание признаков дата сета

Название	Файл	Тип данных	Непустых значений	Уникальных значений
Соотношение матрица-наполнитель	X_bp	float64	1023	1014
Плотность, кг/м3	X_bp	float64	1023	1013
модуль упругости, ГПа	X_bp	float64	1023	1020
Количество отвердителя, м.%	X_bp	float64	1023	1005
Содержание эпоксидных групп,%_2	X_bp	float64	1023	1004
Температура вспышки, С_2	X_bp	float64	1023	1003
Поверхностная плотность, г/м2	X_bp	float64	1023	1004
Модуль упругости при растяжении, ГПа	X_bp	float64	1023	1004
Прочность при растяжении, МПа	X_bp	float64	1023	1004
Потребление смолы, г/м2	X_bp	float64	1023	1003
Угол нашивки, град	X_nup	int64	1023	2
Шаг нашивки	X_nup	float64	1023	989
Плотность нашивки	X_nup	float64	1023	988

Гистограммы распределения переменных и диаграммы «ящик с усами» приведены на рисунке 1.1 и рисунке 1.2. По ним видно, что все признаки, кроме «Угол нашивки», имеют нормальное распределение и принимают неотрицательные значения. «Угол нашивки» принимает значения: 0, 90.

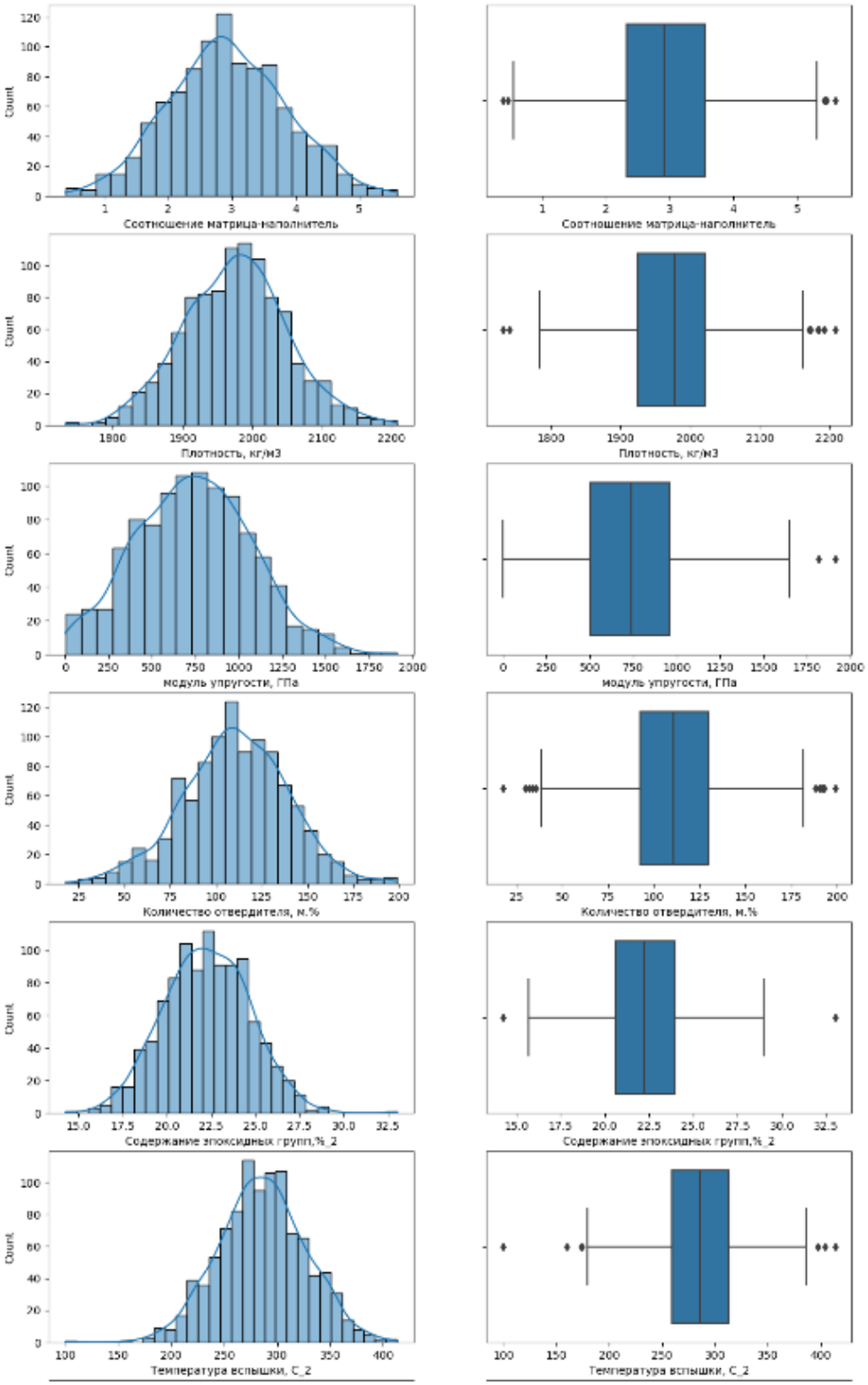


Рисунок 1.1 - Гистограммы распределения переменныхи диаграммы «ящик с усами»

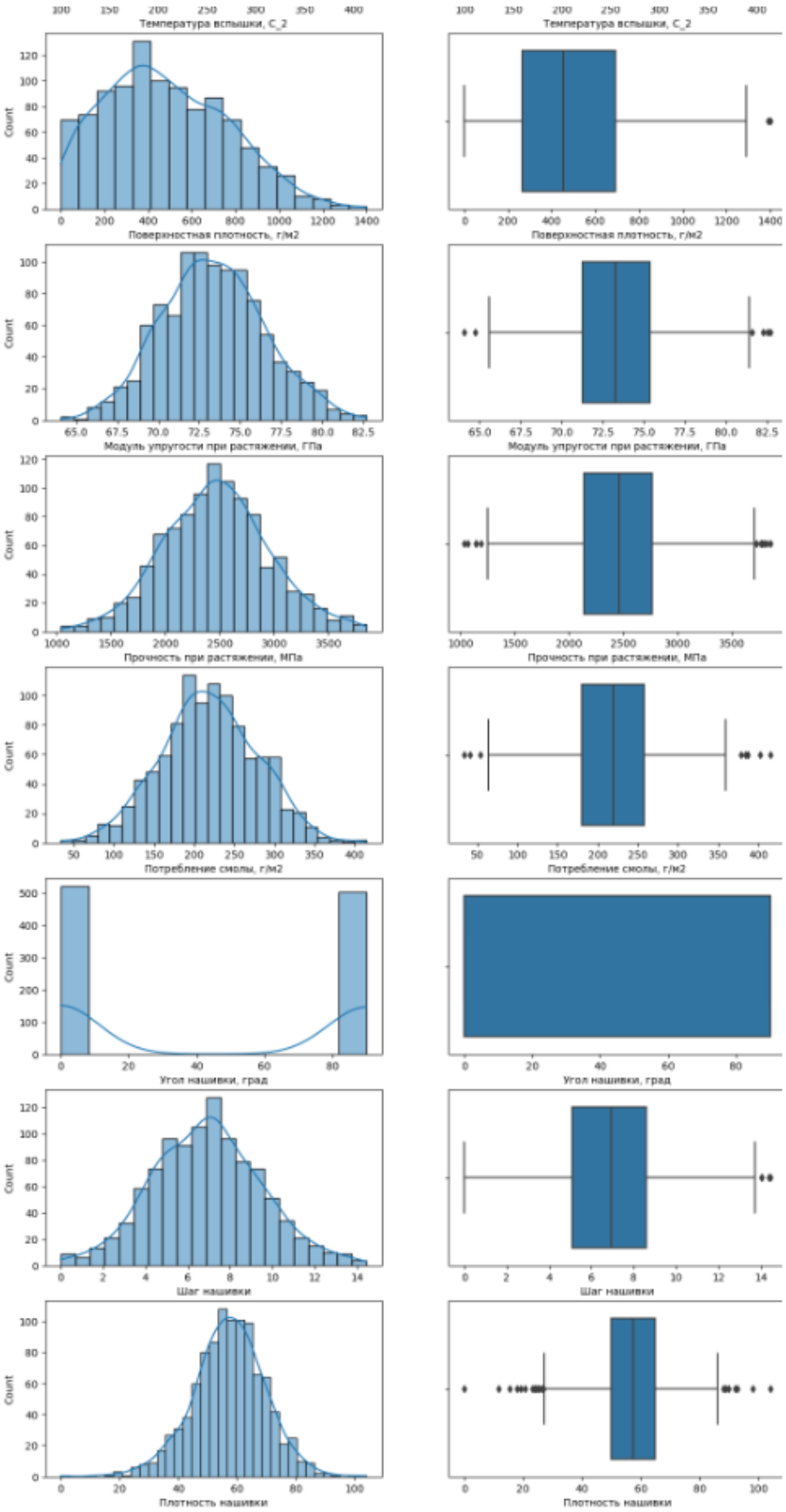


Рисунок 1.2 - Гистограммы распределения переменныхи диаграммы «ящик с усами»

В датасете отсутствуют пропуски данных, поэтому полагаю, что он был предварительно подготовлен.

Методом data.describe() смотрим описательную статистику (Рисунок 2). Метод показывает количество значений, среднее значение, стандартное отклонение, минимальное значение, 25-50-75% перцентили и максимальное значение. Она в численном виде отражает то, что мы видим на гистограммах.

	count	mean	std	min	25%	50%	75%	max
Соотношение матрица-наполнитель	1023.0	2.930366	0.913222	0.389403	2.317887	2.906878	3.552660	5.591742
Плотность, кг/м3	1023.0	1975.734888	73.729231	1731.764635	1924.155467	1977.621657	2021.374375	2207.773481
модуль упругости, ГПа	1023.0	739.923233	330.231581	2.436909	500.047452	739.664328	961.812526	1911.536477
Количество отвердителя, м.%	1023.0	110.570769	28.295911	17.740275	92.443497	110.564840	129.730366	198.953207
Содержание эпоксидных групп,%_2	1023.0	22.244390	2.406301	14.254985	20.608034	22.230744	23.961934	33.000000
Температура вспышки, С_2	1023.0	285.882151	40.943260	100.000000	259.066528	285.896812	313.002106	413.273418
Поверхностная плотность, г/м2	1023.0	482.731833	281.314690	0.603740	266.816645	451.864365	693.225017	1399.542362
Модуль упругости при растяжении, ГПа	1023.0	73.328571	3.118983	64.054061	71.245018	73.268805	75.356612	82.682051
Прочность при растяжении, МПа	1023.0	2466.922843	485.628006	1036.856605	2135.850448	2459.524526	2767.193119	3848.436732
Потребление смолы, г/м2	1023.0	218.423144	59.735931	33.803026	179.627520	219.198882	257.481724	414.590628
Угол нашивки, град	1023.0	44.252199	45.015793	0.000000	0.000000	0.000000	90.000000	90.000000
Шаг нашивки	1023.0	6.899222	2.563467	0.000000	5.080033	6.916144	8.586293	14.440522
Плотность нашивки	1023.0	57.153929	12.350969	0.000000	49.799212	57.341920	64.944961	103.988901

Рисунок 2 — Описательная статистика признаков датасета

По всем признакам есть существенный разброс значений.

Проверяем количество пропусков в сумме по столбцам методом data.isnull(). sum(). Пропусков нет.

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. Графики рассеивания приведены на Рисунке 3.

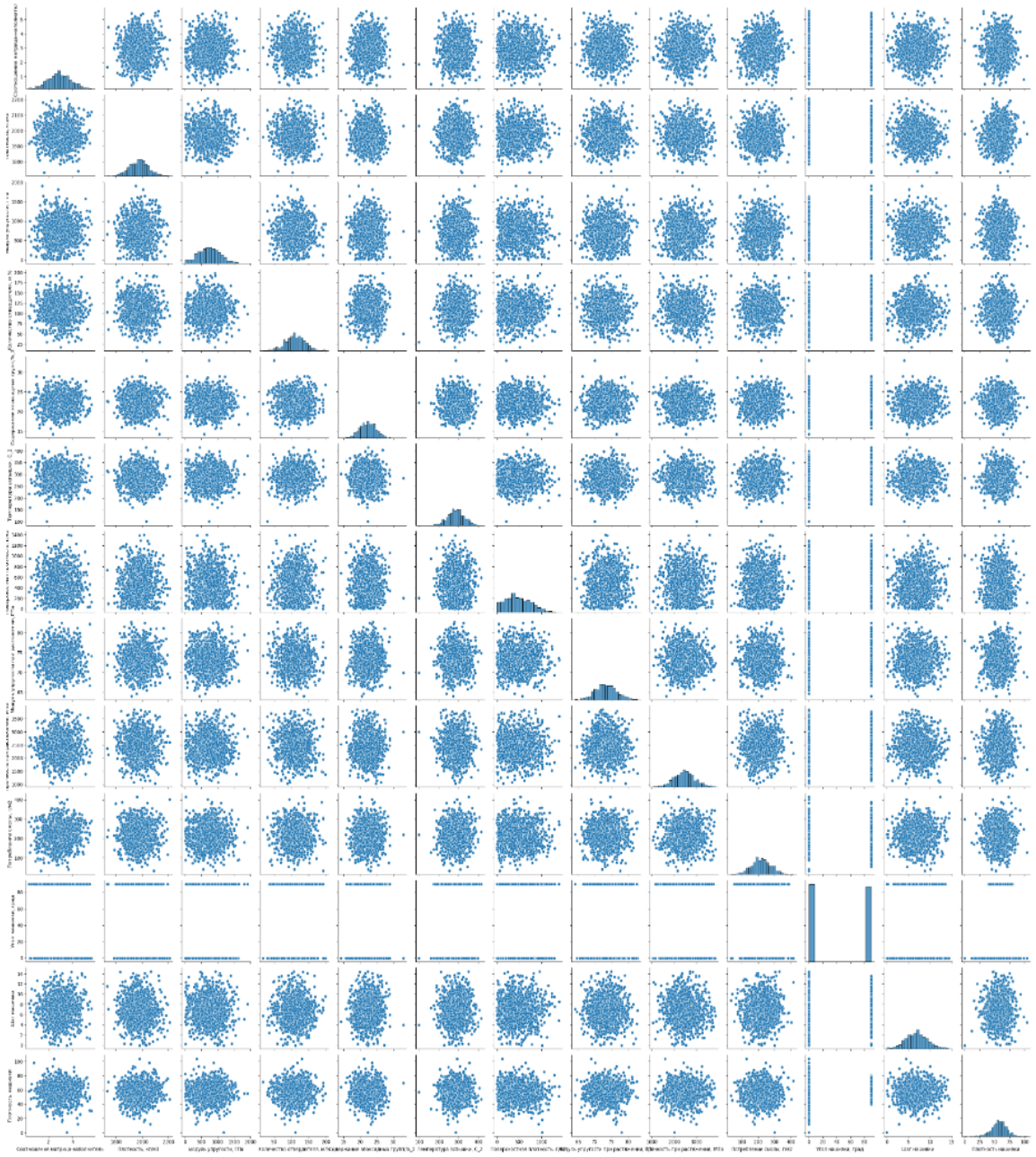


Рисунок 3 - Графики рассеяния точек

Существуют такие методы выявления выбросов для признаков с нормальным распределением:

– метод 3-х сигм;

– метод межквартильных расстояний.

Применив эти методы на нашем датасете было найдено:

– методом 3-х сигм — 24 выброса;

– методом межквартильных расстояний — 93 выброса.

Пример выбросов на гистограмме распределения и диаграмме «ящик сусами» приведен на рисунках 4.1 .

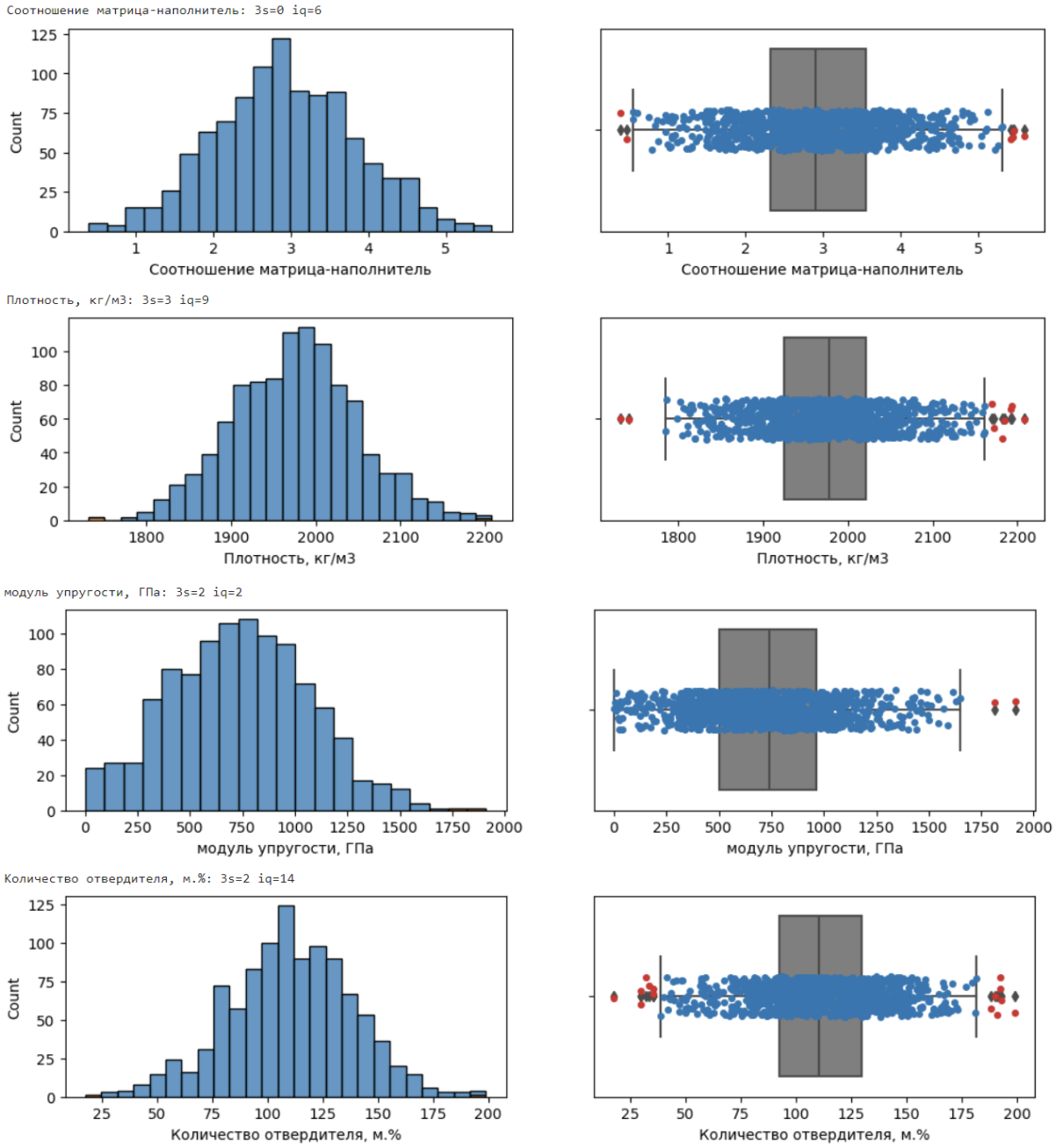


Рисунок 4.1. Примеры выбросов

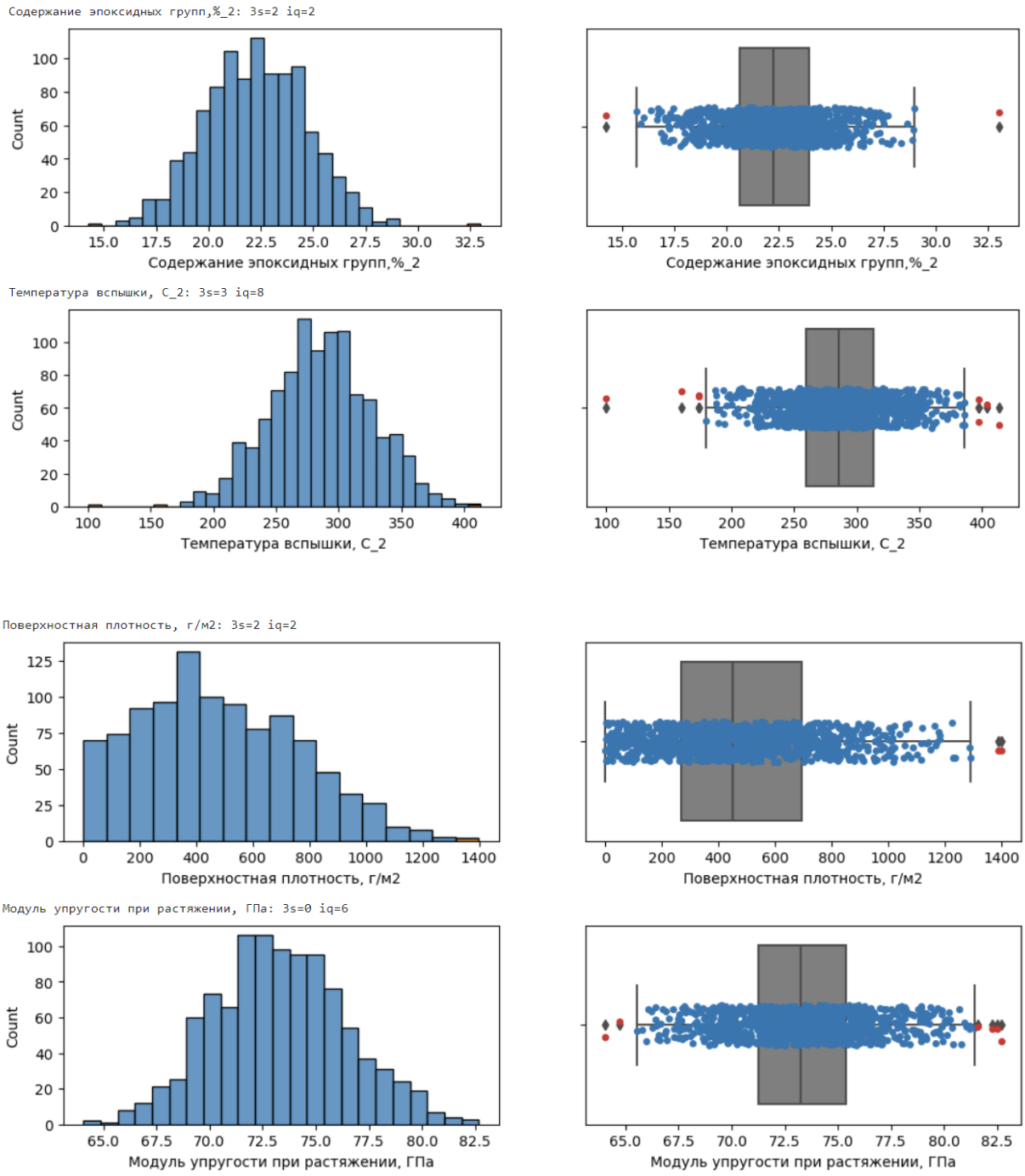


Рисунок 4.2. Примеры выбросов

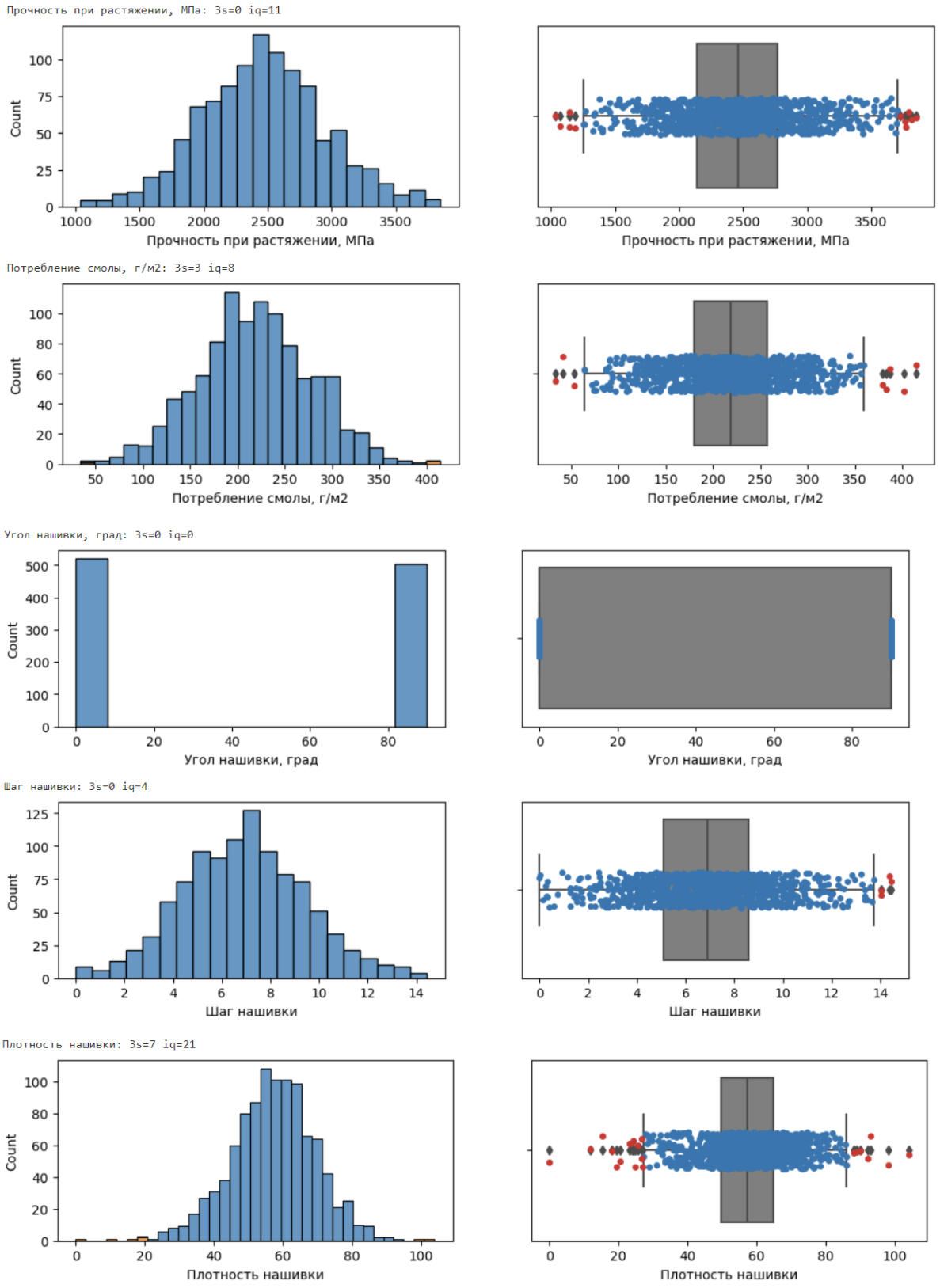


Рисунок 4.3. Примеры выбросов

Поскольку у признаков распределение нормальное, можно применить метод 3-х сигм как более деликатный, чтобы не потерять значимые данные. Значения, определенные как выбросы, удаляем. После этого осталось в датасете осталось 1000 строк и 13 признаков-переменных.

В задании целевыми переменными указаны:

- 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.

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

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).

$$Y = b0 + b1 * x1 + b2 * x2 + \cdots + bn * xn,$$

(2)

где n - число входных переменных.

Очевидно, что в данном случае модель будет описываться не прямой, а гиперплоскостью. Коэффициенты уравнения множественной линейной регрессии подбираются так, чтобы минимизировать сумму квадратов отклонения реальных точек данных от этой гиперплоскости.

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

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

1.2.2 Метод опорных векторов для регрессии

Метод опорных векторов (Support Vector Machine, SVM) — один из наиболее популярных методов машинного обучения. Он создает гиперплоскость или набор гиперплоскостей в многомерном пространстве, которые могут быть использованы для решения задач классификации и регрессии.

Чаще всего он применяется в постановке бинарной классификации.

Основная идея заключается в построении гиперплоскости, разделяющей объекты выборки оптимальным способом. Интуитивно, хорошее разделение достигается за счет гиперплоскости, которая имеет самое большое расстояние до

ближайшей точки обучающей выборке любого класса. Максимально близкие объекты разных классов определяют опорные вектора.

Если в исходном пространстве объекты линейно неразделимы, то выполняется переход в пространство большей размерности.

Решается задача оптимизации.

Для вычислений используется ядерная функция, получающая на вход два вектора и возвращающая меру сходства между ними:

- линейная;
- полиномиальная;
- гауссовская (rbf).

Эффективность метода опорных векторов зависит от выбора ядра, параметров ядра и параметра C для регуляризации.

Преимущество метода — его хорошая изученность.

Недостатки:

– чувствительность к выбросам;
– отсутствие интерпретируемости.

Вариация метода для регрессии называется SVR (Support Vector Regression).

В python реализацию SVR можно найти в sklearn.svm.SVR.

1.2.3 Метод k-ближайших соседей

Еще один метод классификации, который адаптирован для регрессии - метод k-ближайших соседей (k Nearest Neighbors). На интуитивном уровне суть метода проста: посмотри на соседей вокруг, какие из них преобладают, таковым ты и являешься.

В случае использования метода для регрессии, объекту присваивается среднее значение по k ближайшим к нему объектам, значения которых уже известны.

Для реализации метода необходима метрика расстояния между объектами. Используется, например, евклидово расстояние для количественных признаков

или расстояние Хэмминга для категориальных.

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 — соответствующийлисту интервал целевой переменной.

При формировании правила для разбиения в очередном узле дерева необходимо выбрать атрибут, по которому это будет сделано. Общее правило 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 наестественном 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 толькоодин объект. Для решения этой проблемы используют разные критерии остановки алгоритма.

Деревья решений реализованы в sklearn.tree.DecisionTreeRegressor.

1.2.5 Случайный лес

Случайный лес (RandomForest) — представитель ансамблевых методов. Если точность дерева решений оказалось недостаточной, мы можем множество моделей собрать в коллектив.

Формула итогового решателя (3) — это усреднение предсказаний отдельных деревьев.

$$a(x) = \frac{1}{N}\sum_{i=1}^{N} b_i(x) \quad (3),$$

где

N – количество деревьев;

i – счетчик для деревьев;

b – решающее дерево;

x – сгенерированная нами на основе данных выборка.

Для определения входных данных каждому дереву используется метод случайных подпространств. Базовые алгоритмы обучаются на различных подмножествах признаков, которые выделяются случайным образом.

Преимущества случайного леса:

- высокая точность предсказания;
- редко переобучается;
- практически не чувствителен к выбросам в данных;
- одинаково хорошо обрабатывает как непрерывные, так и дискретные признаки, данные с большим числом признаков;
- высокая параллелизуемость и масштабируемость.

Из недостатков можно отметить, что его построение занимает больше времени. Так же теряется интерпретируемость.

Метод реализован в sklearn.ensemble.RandomForestRegressor.

1.2.6 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 соответствуют 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.

В этой работе я использую реализацию градиентного бустинга из библиотеки sklearn — sklearn.ensemble.GradientBoostingRegressor. Хотя существуют и другие реализации, некоторые из которых более мощные, например, XDGBoost.

1.2.7 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, сигмоида.

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 потери. 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 рисунке 2 мы видели график попарного рассеяния точек. По форме «облаков точек» мы не заметили зависимостей, которые станут основой работы моделей. Помочь выявить связь между признаками может матрица корреляции,

приведенная на рисунке 5.

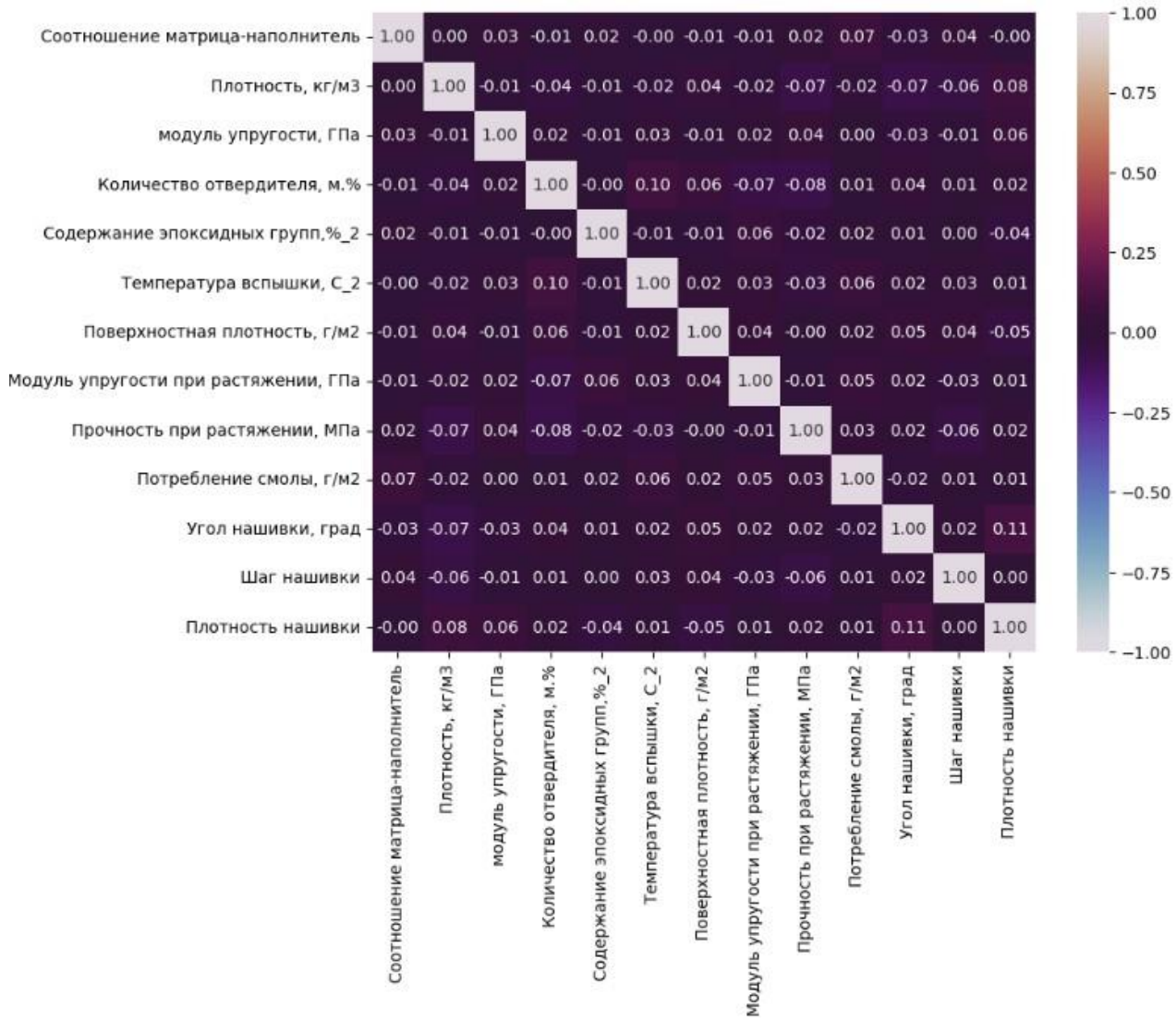


Рисунок 5 — Матрица корреляции

По матрице корреляции мы видим, что все коэффициенты корреляции близки к нулю, что означает отсутствие линейной зависимости между признаками.

1.3.1 Выбор признаков

Статистическими методами мы зависимостей между признаками не обнаружили.

Можно предположить, что признаки делятся на:

– свойства матрицы;

– свойства наполнителя;
– свойства смеси и производственного процесса;
– свойства готового композита.

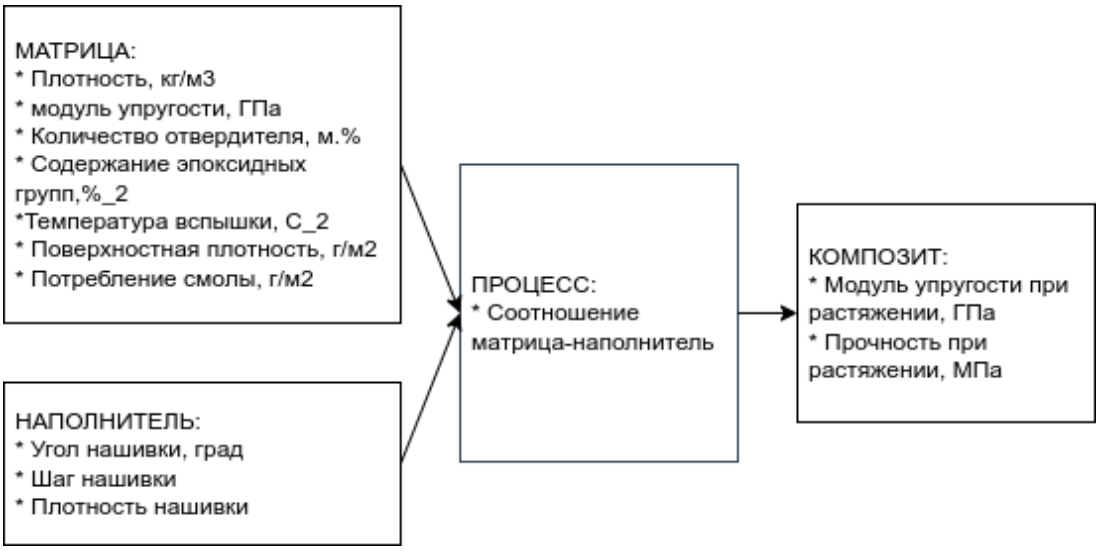


Рисунок 6 — Группы признаков
с точки зрения предметной области

Таким образом, в нашем случае целевые признаки имеют зависимости вида (4), (5), (6).

$$\text{модуль упругости при растяжении}(\text{композит}) = f(\text{матрица}, \text{наполнитель}, \text{процесс}) \quad (4)$$

$$\text{прочность при растяжении}(\text{композит}) = f(\text{матрица}, \text{наполнитель}, \text{процесс}) \quad (5)$$

$$\text{соотношение} - \text{матрица} - \text{наполнитель}(\text{процесс}) = f(\text{матрица}, \text{наполнитель}, \text{композит}) \quad (6)$$

Для каждого из целевых признаков построим отдельную модель, следовательно, решим 3 отдельные задачи.

1.3.2 Ход решения задачи

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

1. разделить данные на тренировочную и тестовую выборки. В задании указано, что на тестирование оставить 30% данных;
2. выполнить препроцессинг, то есть подготовку исходных данных;
3. взять несколько моделей с гиперпараметрами по умолчанию, а некоторые модели взять с гиперпараметрами, подобранными с помощью GridSearchCV, и используя перекрестную проверку, посмотреть их метрики на тренировочной выборке;
4. сравнить метрики моделей после подбора гиперпараметров и выбрать лучшую;
5. получить предсказания моделей на тестовой выборке,сделать выводы;

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. Т.к. у этого признака всего два значения, то при нормализации они примут значения 0 и 1, поэтому можно не использовать 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. Т.к. наши данные распределены

нормально, то можно использовать любой метод преобразований:

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

Использую нормализацию MinMaxScaler.

Выходные переменные никак не изменяю.

1.3.3 Поиск гиперпараметров по сетке

Поиск гиперпараметров по сетке реализует класс GridSearchCV из sklearn. Он получает модель и набор гиперпараметров, поочередно передает их в модель, выполняет обучение и определяет лучшие комбинации гиперпараметров.

1.3.4 Метрики качества моделей

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

1. RMSE (Root Mean Squared Error) или корень из средней квадратичной ошибки принимает значениях в тех же единицах, что и целевая переменная. Метрика использует возведение в квадрат, поэтому хорошо обнаруживает грубые ошибки, но сильно чувствительна к выбросам;
2. MAE (Mean Absolute Error) - средняя абсолютная ошибка так же принимает значениях в тех же единицах, что и целевая переменная;
3. MSE – показатель среднеквадратичной ошибки (Mean Squared Error).

MAE, MSE, RMSE эти метрики надо минимизировать.

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 6. Описательная статистика входных признаков после предобработки показана на рисунке 7. Описательная статистика выходного признака показана на рисунке 8.

```
X1_train_full (700, 11) y1_train (700,)
X1_test_full (300, 11) y1_test (300,)
```

Рисунок 6 - Размерности тренировочного и тестового множеств после разбиения для 1-й задачи

	count	mean	std	min	25%	50%	75%	max
Соотношение матрица-наполнитель	700.0	0.485931	0.171753	0.000000	0.370964	0.482614	0.598634	0.973824
Плотность, кг/м3	700.0	0.472639	0.178456	0.003805	0.341250	0.475188	0.585166	1.000000
модуль упругости, ГПа	700.0	0.442705	0.201107	0.000000	0.300116	0.443514	0.576671	0.986997
Количество отвердителя, м.%	700.0	0.500208	0.170812	0.000000	0.389062	0.499809	0.619464	1.000000
Содержание эпоксидных групп,%_2	700.0	0.492359	0.177694	0.014011	0.371262	0.489801	0.622879	1.000000
Температура вспышки, С_2	700.0	0.491495	0.176607	0.002124	0.369007	0.490614	0.613700	1.000000
Поверхностная плотность, г/м2	700.0	0.362326	0.213716	0.000000	0.202486	0.334003	0.517914	0.997948
Потребление смолы, г/м2	700.0	0.517680	0.171437	0.065453	0.401857	0.517418	0.632832	1.000000
Угол нашивки, град	700.0	0.490000	0.500257	0.000000	0.000000	0.000000	1.000000	1.000000
Шаг нашивки	700.0	0.476447	0.177032	0.000000	0.344539	0.473847	0.594983	1.000000
Плотность нашивки	700.0	0.509360	0.163700	0.000000	0.407037	0.508219	0.618271	1.000000

Рисунок 7 - Описательная статистика входных признаков тренировочного множества после предобработки для 1-й задачи

```
count    700.000000
mean      73.216471
std        3.117332
min       64.054061
25%       71.244758
50%       73.015881
75%       75.231680
max       82.682051
Name: Модуль упругости при растяжении, ГПа, dtype: float64
```

Рисунок 8 - Описательная статистика выходного признака тренировочного множества для 1-й задачи

2.1.2 Для прогнозирования прочности при растяжении

Признаки датасета были разделены на входные и выходные, а строки - на тренировочное и тестовое множество. Размерности полученных наборов данных показаны на рисунке 9. Описательная статистика входных признаков до и после предобработки показана на рисунке 10. Описательная статистика выходного признака показана на рисунке 11.

```
X2_train_full (700, 11) y2_train (700,)
X2_test_full (300, 11) y2_test (300,)
```

Рисунок 9 - Размерности тренировочного и тестового множеств после разбиения для 2-й задачи

	count	mean	std	min	25%	50%	75%	max
Соотношение матрица-наполнитель	700.0	0.485931	0.171753	0.000000	0.370964	0.482614	0.598634	0.973824
Плотность, кг/м3	700.0	0.472639	0.178456	0.003805	0.341250	0.475188	0.585166	1.000000
модуль упругости, ГПа	700.0	0.442705	0.201107	0.000000	0.300116	0.443514	0.576671	0.986997
Количество отвердителя, м.%	700.0	0.500208	0.170812	0.000000	0.389062	0.499809	0.619464	1.000000
Содержание эпоксидных групп,%_2	700.0	0.492359	0.177694	0.014011	0.371262	0.489801	0.622879	1.000000
Температура вспышки, С_2	700.0	0.491495	0.176607	0.002124	0.369007	0.490614	0.613700	1.000000
Поверхностная плотность, г/м2	700.0	0.362326	0.213716	0.000000	0.202486	0.334003	0.517914	0.997948
Потребление смолы, г/м2	700.0	0.517680	0.171437	0.065453	0.401857	0.517418	0.632832	1.000000
Угол нашивки, град	700.0	0.490000	0.500257	0.000000	0.000000	0.000000	1.000000	1.000000
Шаг нашивки	700.0	0.476447	0.177032	0.000000	0.344539	0.473847	0.594983	1.000000
Плотность нашивки	700.0	0.509360	0.163700	0.000000	0.407037	0.508219	0.618271	1.000000

Рисунок 10 - Описательная статистика входных признаков тренировочного множества после предобработки для 2-й задачи

```
count     700.000000
mean     2471.491430
std       480.707711
min      1036.856605
25%      2146.936034
50%      2465.763343
75%      2755.169485
max      3848.436732
Name: Прочность при растяжении, МПа, dtype: float64
```

Рисунок 11 - Описательная статистика выходного признака тренировочного множества для 2-й задачи

2.1.3 Для прогнозирования соотношения матрица-наполнитель

Признаки датасета были разделены на входные и выходные, а строки - на тренировочное и тестовое множество. Размерности полученных наборов данных показаны на рисунке 12. Описательная статистика входных признаков до и после предобработки показана на рисунке 13. Описательная статистика выходного признака показана на рисунке 14.

```
X3_train_full (700, 12) y3_train (700,)
X3_test_full (300, 12) y3_test (300,)
```

Рисунок 12 - Размерности тренировочного и тестового множеств после разбиения для 3-й задачи

	count	mean	std	min	25%	50%	75%	max
Плотность, кг/м3	700.0	0.472639	0.178456	0.003805	0.341250	0.475188	0.585166	1.000000
модуль упругости, ГПа	700.0	0.442705	0.201107	0.000000	0.300116	0.443514	0.576671	0.986997
Количество отвердителя, м.%	700.0	0.500208	0.170812	0.000000	0.389062	0.499809	0.619464	1.000000
Содержание эпоксидных групп,%_2	700.0	0.492359	0.177694	0.014011	0.371262	0.489801	0.622879	1.000000
Температура вспышки, С_2	700.0	0.491495	0.176607	0.002124	0.369007	0.490614	0.613700	1.000000
Поверхностная плотность, г/м2	700.0	0.362326	0.213716	0.000000	0.202486	0.334003	0.517914	0.997948
Модуль упругости при растяжении, ГПа	700.0	0.491863	0.167347	0.000000	0.386016	0.481094	0.600044	1.000000
Прочность при растяжении, МПа	700.0	0.510259	0.170974	0.000000	0.394824	0.508222	0.611156	1.000000
Потребление смолы, г/м2	700.0	0.517680	0.171437	0.065453	0.401857	0.517418	0.632832	1.000000
Угол нашивки, град	700.0	0.490000	0.500257	0.000000	0.000000	0.000000	1.000000	1.000000
Шаг нашивки	700.0	0.476447	0.177032	0.000000	0.344539	0.473847	0.594983	1.000000
Плотность нашивки	700.0	0.509360	0.163700	0.000000	0.407037	0.508219	0.618271	1.000000

Рисунок 13 - Описательная статистика входных признаков тренировочного м-ва после предобработки для 3-й задачи

```
count    700.000000
mean       2.917381
std        0.893515
min        0.389403
25%        2.319283
50%        2.900123
75%        3.503701
max        5.455566
Name: Соотношение матрица-наполнитель, dtype: float64
```

Рисунок 14 - Описательная статистика выходного признака для 3-й задачи

2.2 Разработка и обучение моделей для прогнозирования модуля упругости при растяжении

Для подбора лучшей модели для этой задачи я взяла следующие модели:

- Линейная регрессия;
- Метод опорных векторов для регрессии;
- Метод k-ближайших соседей;
- Деревья решений;
- Градиентный бустинг;
- Случайный лес.

Ни одна из выбранных мной моделей не оказалась оптимальной для наших данных.

Метрики работы выбранных моделей, полученные с помощью перекрестной проверки на тестовом множестве, приведены на рисунке 15

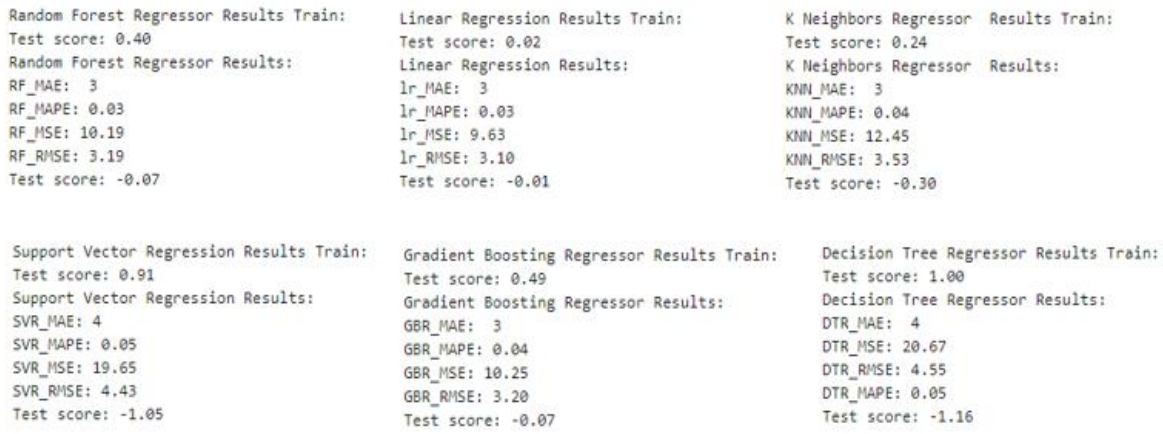


Рисунок 15 — Результаты моделей

Все модели крайне плохо описывают исходные данные – по всем моделям MAE приближена к стандартному отклонению.

	Перрессор	MAE
0	RandomForest	2.578553
1	Linear Regression	2.510989
2	KNeighbors	2.803555
3	Support Vector	3.505371
4	GradientBoosting	2.586746
5	DecisionTree	3.506381
6	RandomForest1_GridSearchCV	2.537475
7	KNeighbors1_GridSearchCV	2.768574
8	DecisionTree1_GridSearchCV	2.527006

Рисунок 16 — Результаты моделей с гиперпараметрами

Решить данную задачу на этом этапе не удалось. По нескольким моделям были подобраны гиперпараметры,но оценки существенно не изменились.

После обучения моделей была проведена оценка точности этих моделей на обучающей и тестовых выборках. В качестве параметра оценки модели использовалась средняя квадратическая ошибка (MSE). Результат неудовлетворительный.

2.3 Для прогнозирования прочности при растяжении

Для подбора лучшей модели для этой задачи я взяла следующие модели:
- Линейная регрессия;
- Метод опорных векторов для регрессии;
- Метод k-ближайших соседей;
- Деревья решений;
- Градиентный бустинг;
- Случайный лес.

Метрики работы выбранных моделей, полученные с помощью перекрестной проверки на тестовом множестве, приведены на рисунке 16.

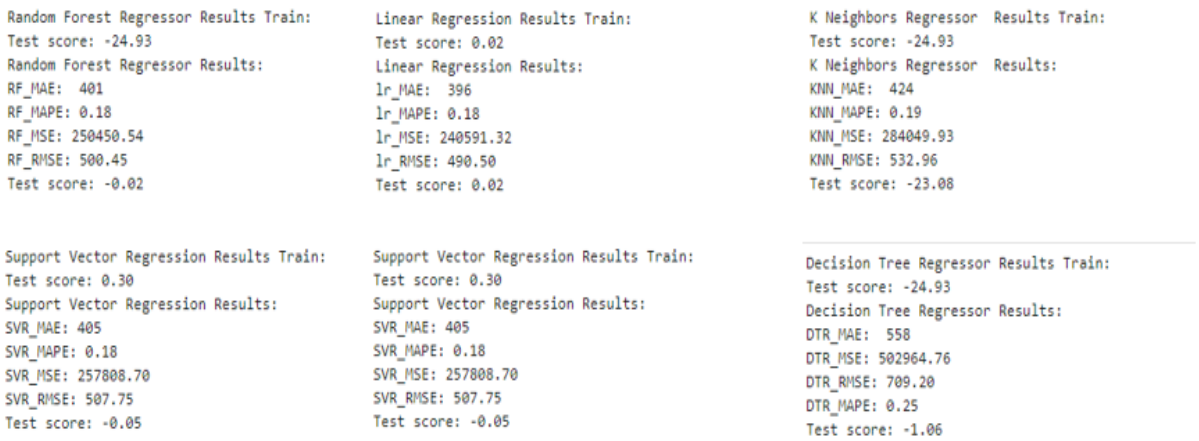


Рисунок 17 — Результаты моделей

Ни одна из выбранных мной моделей не соответствует данным.

Результат исследования отрицательный. Не удалось получить модели, которая могла бы оказать помощь в принятиирешений специалисту предметной области.

2.4 Разработка нейронной сети для прогнозирования соотношения

матрица-наполнитель

По заданию для соотношения матрица-наполнитель необходимо построить нейросеть.

2.4.1 MLPRegressor из библиотеки sklearn

Строю нейронную сеть с помощью класса MLPRegressor следующей архитектуры:

– слоев: 8;

– нейронов на каждом слое: 24;

– активационная функция: relu;

– оптимизатор: adam;

– пропорция разбиения данных на тестовые и валидационные: 30%;

– ранняя остановка, если метрики на валидационной выборке не улучшаются;

– количество итераций: 5000.

Нейросеть обучилась за 826 мс и 105 итерации. График обучения приведенна рисунке 17.

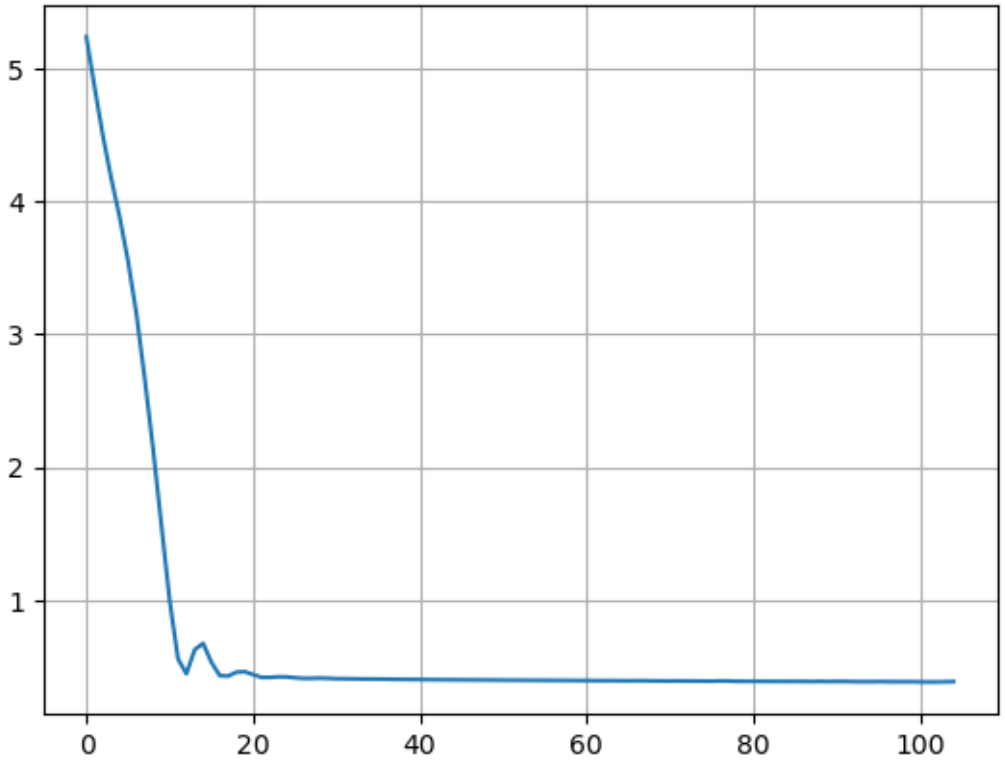


Рисунок 17 — График обучения MLPRegressor

Итоги работы модели:

```
MAE3: 0.7694991587628883
MSE3: 0.9045963312234566
RMSE3: 0.9511026922595985
```

2.4.2 Нейросеть из библиотеки tensorflow

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

– входной слой для 12 признаков;

– пакетная нормализация BatchNormalization

– оптимизатор: Adam;

– loss-функция: MeanAbsoluteError.

– Архитектура нейросети приведена на рисунке 18

Layer (type)	Output Shape	Param #
dense_4 (Dense)	(None, 24)	312
batch_normalization_1 (BatchNormalization)	(None, 24)	96
dense_5 (Dense)	(None, 8)	200
dense_6 (Dense)	(None, 8)	72
dense_7 (Dense)	(None, 1)	9

Рисунок 18 — Архитектура нейросети в виде summary

Запускаю обучение нейросети со следующими

параметрами:

– пропорция разбиения данных на тестовые и валидационные: 30%;

– количество эпох: 100.

График обучения приведен на рисунке 19

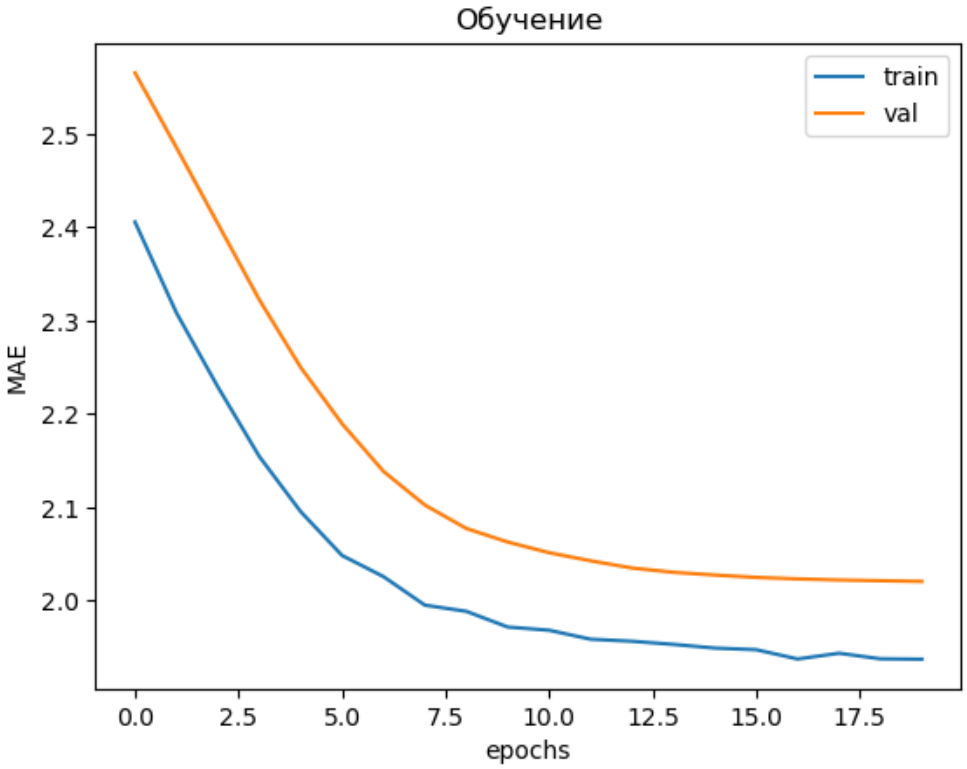


Рисунок 19 — График обучения нейросети

У нейросети показатели для тестовой выборки сильнее отличаются в худшую сторону от показателей тренировочной. Это говорит о том, что она не нашла закономерностей, а стала учить данные из тестовой выборки. Возможно, требуется более тщательное и грамотное построение архитектуры нейронной сети, чтобы получить лучший результат. Но сейчас задача далека от решения.

Наша модель работает не точнее среднего, и бесполезна для применения в реальных условиях.

2.6. 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.

Решено разработать веб-приложение с помощью языка Python, фреймворка Flask и шаблонизатора Jinja.

Эту задачу получилось решить. Скриншоты разработанного веб-приложения приведены в приложении А.

2.7. Создание удаленного репозитория

Для данного исследования был создан удаленный репозиторий на GitHub, который находится по адресу https://github.com/Tatianaizh/composite-material.git. Нанего были загружены результаты работы: исследовательский notebook, код приложения.

Заключение

В ходе выполнения данной работы мы прошли 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 транформацию данных. Здесь нам был предоставлен готовый набор данных, поэтому через трудности работы с разными источниками ипарсингом данных мы еще не соприкоснулись;

– 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.

В этой работе мы имели дело не с учебными наборами данных, которые дают хорошо изученные решения, а с реальной производственной задачей. И к

сожалению, не смогли поставленную задачу решить — не получили моделей, которые бы описывали закономерности предметной области. Я проделала максимум исследований, которые возможно было провести в очень короткий промежуток времени, применила большую часть знаний, полученных в ходе прохождения курса.

Возможные причины неудачи:

– нечеткая постановка задачи, отсутствие дополнительной информации о зависимости признаков с точки зрения физики процесса. Незначимые признаки являются для модели шумом, и мешают найти зависимость целевых от значимых входных признаков;

– исследование предварительно обработанных данных. Возможно, на "сырых", не предобработанных данных можно было бы получить более качественные модели, воспользовавшись другими методами очистки и подготовки;

– мой недостаток знаний и опыта. Нейросети являются самым современным подходам к решению такого рода задач. Они способны находить скрытые и нелинейные зависимости в данных. Но выбор оптимальной архитектуры нейросети является неочевидной задачей.

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. Так же 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.

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.

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://proglib.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-lesa-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.