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 Левин Евгений Владимирович

Москва, 2023

Оглавление

[Введение 3](#_Toc133676474)

[1. Аналитическая часть 5](#_Toc133676475)

[1.1. Постановка задачи 5](#_Toc133676476)

[1.2. Описание используемых методов 10](#_Toc133676477)

[1.2.1 Линейная регрессия 10](#_Toc133676478)

[1.2.3 Метод опорных векторов для регрессии 11](#_Toc133676479)

[1.2.4 Метод k-ближайших соседей 12](#_Toc133676480)

[1.2.5 Случайный лес 13](#_Toc133676481)

[1.2.6 Адаптивный бустинг 14](#_Toc133676482)

[1.2.7 Градиентный бустинг 15](#_Toc133676483)

[1.2.8 Нейронная сеть 16](#_Toc133676484)

[1.3. Разведочный анализ данных 16](#_Toc133676485)

[1.3.1 Ход решения задачи 17](#_Toc133676486)

[1.3.2 Препроцессинг 18](#_Toc133676487)

[1.3.3 Поиск гиперпараметров по сетке 19](#_Toc133676488)

[1.3.4 Метрики качества моделей 19](#_Toc133676489)

[2. Практическая часть 20](#_Toc133676490)

[2.1. Разбиение и предобработка данных 20](#_Toc133676491)

[2.2 Разработка и обучение моделей для прогнозирования модуля упругости при растяжении. Тестирование моделей. 20](#_Toc133676492)

[2.3 Разработка нейронной сети для прогнозирования соотношения матрица-наполнитель 23](#_Toc133676493)

[2.4. Разработка приложения 26](#_Toc133676505)

[2.5. Создание удаленного репозитория 28](#_Toc133676506)

[Заключение 29](#_Toc133676507)

[Библиографический список 30](#_Toc133676508)

# Введение

* 1. **Тема работы**: прогнозирование конечных свойств новых материалов (композиционных материалов).

1. 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.
2. Яркий пример композита - железобетон. Бетон прекрасно сопротивляется сжатию, но плохо растяжению. Стальная арматура внутри бетона компенсирует его неспособность сопротивляться сжатию, формируя тем самым новые, уникальные свойства.
3. Современные композиты изготавливаются из других 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.
4. 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. Кейс основан на реальных производственных задачах Центра НТИ «Цифровое материаловедение: новые материалы и вещества» (структурное подразделение МГТУ им. Н.Э. Баумана).
5. **Актуальность**: Созданные прогнозные модели помогут сократить количество проводимых испытаний, а также пополнить базу данных материалов возможными новыми характеристиками материалов, и цифровыми двойниками новых композитов.

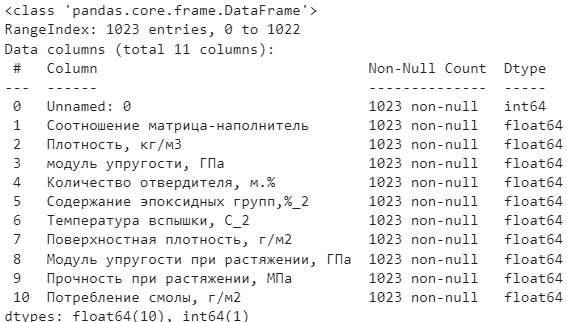
# 1. Аналитическая часть

## 1.1. Постановка задачи

От специалистов в предметной области получен датасет, состоящий из двух файлов: X\_bp (составляющая из базальтопластика) и Х\_nup (составляющая из углепластика).

На рисунке 1 мы видим, что файл X\_bp содержит:

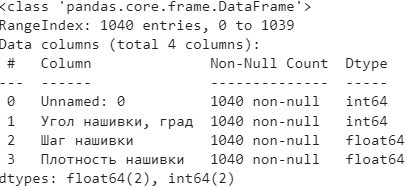
* признаков: 10 и индекс;
* строк: 1023.



Рисунок

На рисунке 2 мы видим, что файл X\_nup содержит:

* признаков: 3 и индекс;
* строк: 1040.



Рисунок

В файле X\_bp мы видим, что признак «Прочность при растяжении» измеряется в мегапаскалях (МПа), в то время как признаки «Модуль упругости» и «Модуль упругости при растяжении» в гигапаскалях (ГПа). Для удобства переведём мегапаскали в гигапаскали.

По условию задачи файлы нужно объединить с типом INNER по индексу. После объединения часть строк из файла X\_nup была отброшена, как и колонка с индексом не несущая никакой информации.

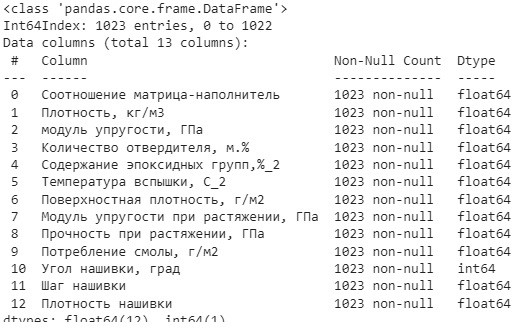
Дальнейшие исследования проводим с объединенным датасетом, содержащим 13 признаков и 1023 записи.

Рисунок 3

На рисунке 3 видно, что все признаки, кроме «Угол нашивки» имеют тип float64, то есть вещественный. Пропусков в данных нет.

На рисунке 4 видно, что признак «Угол нашивки» имеет только 2 уникальных значения и мог бы быть рассмотрен как категориальный. Но от специалистов в предметной области получена информация, что данный признак теоретически может принимать и другие значения в градусах. Поэтому менять его значения на 0 и 1 мы не будем.

Рисунок 4

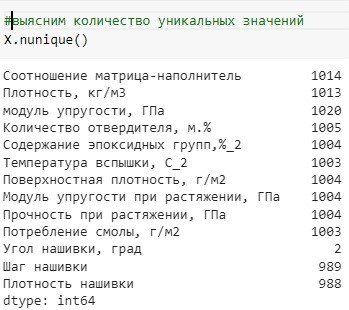
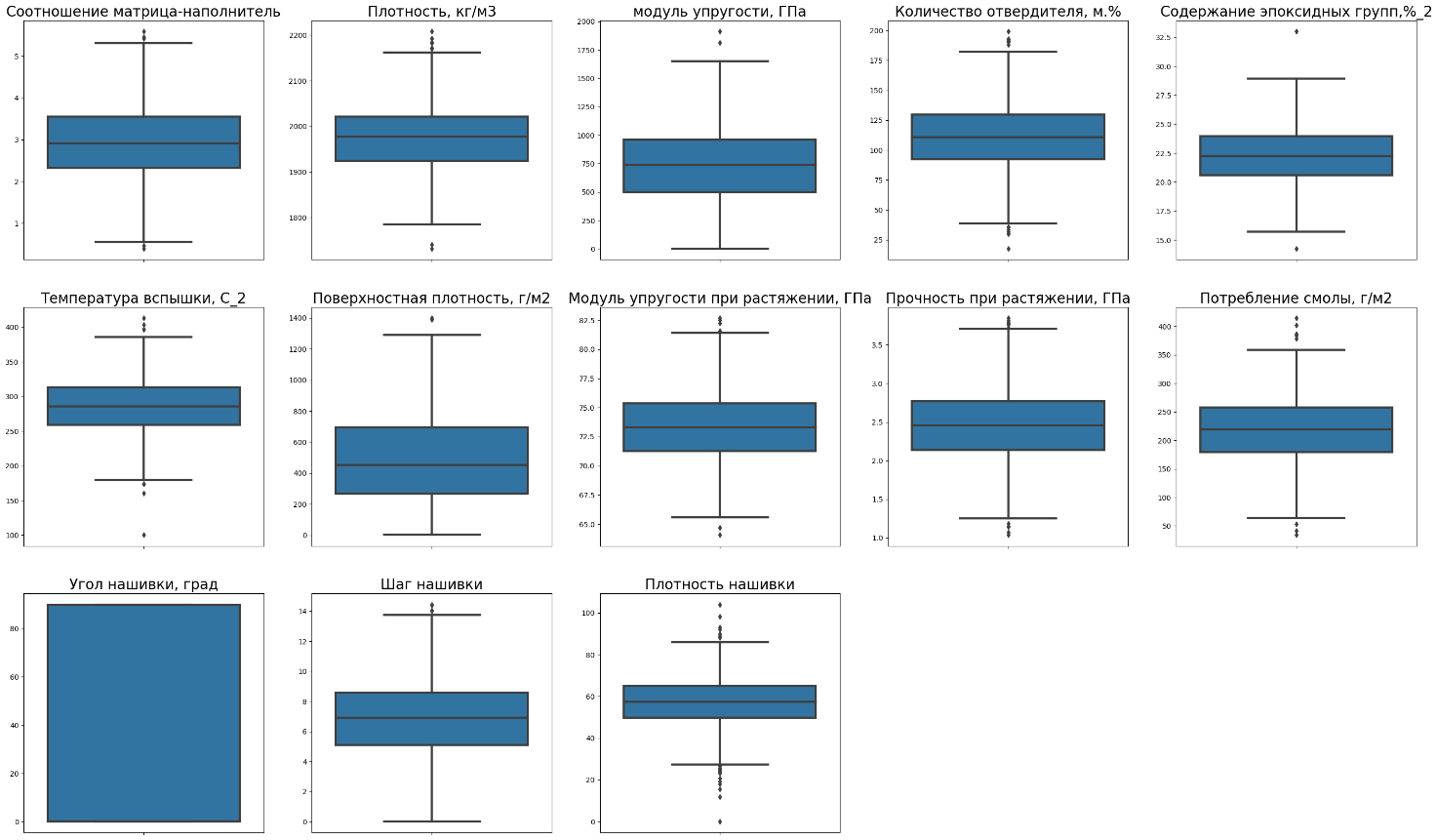


Рисунок 4

На рисунке 5 приведены диаграммы «Ящик с усами» для каждого признака. Видим, что в датасете присутствуют выбросы данных



Рисунок

Также для всех признаков построим парный график (Рисунок 6). Каких-то зависимостей на нём не видно. Все признаки, кроме «Угла нашивки» имеют нормальное распределение.

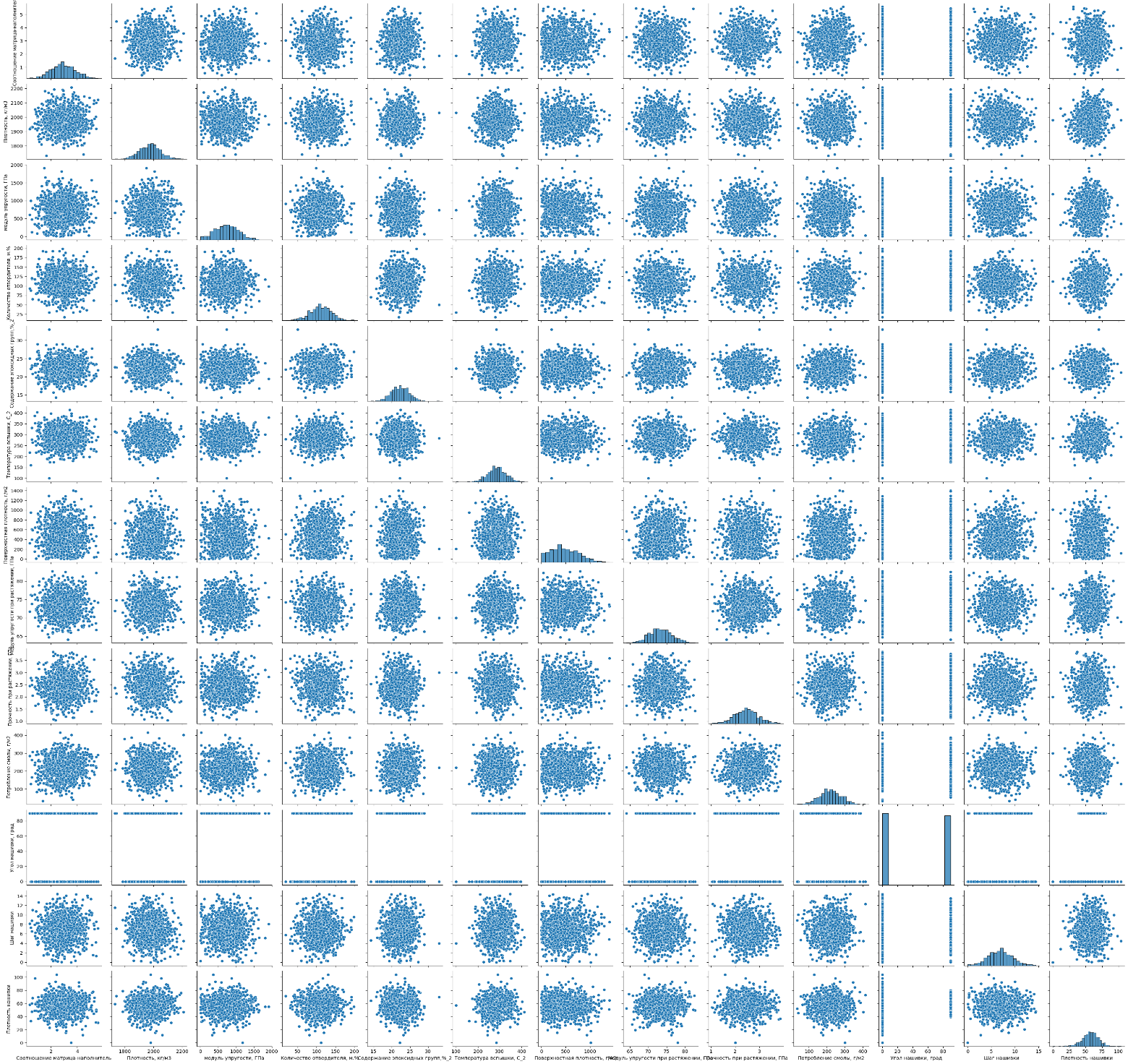
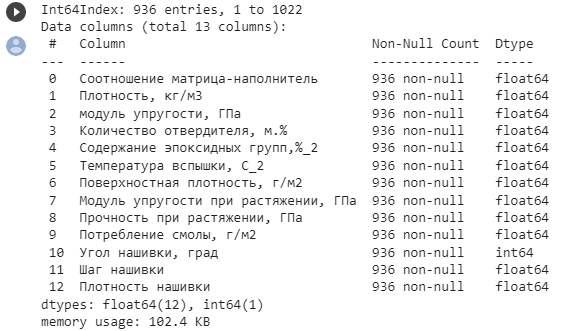


Рисунок 6

Имеющиеся выбросы выявим методом межквартильных расстояний. После их удаления в датасете осталось 936 записей.



Рисунок

В задании целевыми переменными указаны:

* модуль упругости при растяжении, Гпа;
* прочность при растяжении, ГПа;
* 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.

(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).

(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 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.

## 1.2.3 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 С 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.

## 1.2.4 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.

## 1.2.5 Случайный лес

Случайный лес - это алгоритм машинного обучения, который используется для решения задач классификации и регрессии. Он работает путем комбинирования нескольких деревьев решений в одну сильную модель.

Принцип работы Случайного леса заключается в том, что он создает множество деревьев решений, каждое из которых обучается на разных подмножествах данных и с использованием разных наборов признаков. Таким образом, каждое дерево решений в лесу имеет свои сильные и слабые стороны, и их комбинация позволяет получить более точные прогнозы.

При построении деревьев решений Случайный лес использует метод бутстрэпа, который позволяет создавать случайные подмножества данных для обучения каждого дерева. Кроме того, для каждого дерева выбирается случайный набор признаков, что позволяет уменьшить корреляцию между деревьями и улучшить точность прогнозов.

Одним из главных преимуществ Случайного леса является его способность к обработке больших объемов данных и к достижению высокой точности классификации. Кроме того, он может использоваться с различными типами моделей обучения и не требует настройки множества гиперпараметров.

Однако, у Случайного леса есть и некоторые недостатки. Например, он может быть медленным в обучении, особенно если используются большие наборы данных и сложные модели обучения. Кроме того, Случайный лес может быть чувствителен к выбросам в данных, что может привести к переобучению модели.

Метод реализован в sklearn.ensemble.RandomForestRegressor.

## 1.2.6 Адаптивный бустинг

AdaBoost - это алгоритм машинного обучения, который используется для решения задач классификации и регрессии. Он работает путем комбинирования нескольких слабых моделей обучения, таких как решающие деревья, в одну сильную модель.

Принцип работы AdaBoost заключается в том, что он последовательно обучает несколько слабых моделей на разных подмножествах данных. На каждой итерации алгоритм назначает больший вес неправильно классифицированным объектам, чтобы следующая модель сосредоточилась на них. Таким образом, AdaBoost фокусируется на объектах, которые сложно классифицировать, и пытается улучшить точность классификации.

В конце, все модели комбинируются в одну сильную модель, которая может давать более точные прогнозы. Комбинация моделей происходит путем взвешенного голосования, где каждая модель имеет свой вес, зависящий от ее точности.

Одним из главных преимуществ AdaBoost является его способность к обработке больших объемов данных и к достижению высокой точности классификации. Кроме того, он может использоваться с различными типами моделей обучения и не требует настройки множества гиперпараметров.

Однако, у AdaBoost есть и некоторые недостатки. Например, он может быть чувствителен к выбросам в данных, что может привести к переобучению модели. Кроме того, AdaBoost может быть медленным в обучении, особенно если используются сложные модели обучения.

## 1.2.7 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.

## 1.2.8 Нейронная сеть

Нейронная сеть - это последовательность нейронов, соединенных между собой связями, которая использует структуру, вдохновленную биологией. Каждый нейрон имеет определенное количество входов, куда поступают сигналы, которые суммируются с учетом значимости каждого входа. Смещение - это дополнительный вход для нейрона, который всегда равен 1 и имеет собственный вес соединения. Функция активации определяет выходное значение нейрона и вводит нелинейность в нейронную сеть.

Прямое распространение - это процесс передачи входных значений в нейронную сеть и получения выходных данных, которые называются прогнозируемым значением. Прогнозируемое значение сравнивается с фактическим с помощью функции потери. В методе обратного распространения ошибки градиенты вычисляются по значениям весов в направлении, обратном прямому распространению сигналов. Значение градиента вычитается из значения веса, чтобы уменьшить значение ошибки. Таким образом происходит процесс обучения.

Для обновления весов в модели используются различные оптимизаторы. Нейронные сети применяются для решения задач регрессии, классификации, распознавания образов и речи, компьютерного зрения и других. На сегодняшний день 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 6 показан график попарного рассеяния точек. Мы не обнаружили зависимостей по форме 'облаков точек', которые могут служить основой для работы моделей. На рисунке 8 представлена матрица корреляции, из которой видно, что корреляции между признаками очень слабы.

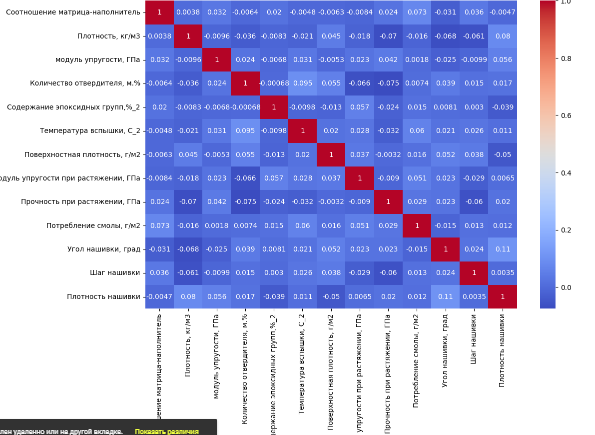


Рисунок 8

.

## 1.3.1 Ход решения задачи

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

* Разделение данных на тренировочную и тестовую выборки. Для этого будет использоваться условие задачи, которое указывает оставить 30% данных для тестирования модели.
* Подготовка исходных данных, включая предобработку данных, обработку пропущенных значений, масштабирование признаков и т.д., чтобы данные были готовы для обучения моделей.
* Выбор базовой модели для предсказания. Базовая модель будет выбрана из нескольких доступных моделей, с использованием гиперпараметров по умолчанию.
* Оценка метрик моделей на тренировочной выборке с использованием кросс-валидации. Для этого будет проведена перекрестная проверка с 10 блоками для каждой модели, чтобы оценить их производительность.
* Подбор гиперпараметров для моделей с использованием поиска по сетке и кросс-валидации. Будут рассмотрены различные значения гиперпараметров моделей для выбора наилучших комбинаций.
* Сравнение метрик моделей после подбора гиперпараметров и выбор лучшей модели на основе этих метрик. Будет проанализировано, какая модель показывает наилучшие результаты на тренировочной выборке.
* Получение предсказаний лучшей модели и базовой модели на тестовой выборке. Будут сделаны выводы о производительности и эффективности лучшей модели в сравнении с базовой моделью на тестовых данных.

Таким образом, весь процесс включает разделение данных, подготовку данных, выбор моделей, оценку метрик, подбор гиперпараметров, сравнение моделей и получение предсказаний на тестовой выборке для сделания окончательных выводов о наилучшей модели для решения поставленной задачи.

## 1.3.2 Препроцессинг

Препроцессинг (или предварительная обработка) данных - это процесс подготовки данных перед применением моделей машинного обучения. Он включает в себя различные техники и методы для очистки, преобразования, масштабирования или извлечения признаков из исходных данных, чтобы обеспечить их корректное использование в моделях машинного обучения. Препроцессинг выполняется с целью улучшения качества моделей, устранения шумов, упрощения данных и облегчения анализа и интерпретации результатов. Препроцессинг является важным шагом в практике машинного обучения, так как качество входных данных существенно влияет на качество и точность моделей. Его необходимо выполнять после разделения на тренировочную и тестовую выборку, как будто мы не знаем параметров тестовой выборки (минимум, максимум, матожидание, стандартное отклонение).

Вещественных количественных признаков у нас большинство. Проблема вещественных признаков в том, что их значения лежат в разных диапазонах, в разных масштабах. Необходимо провести масштабирование— приведение в диапазон от 0 до 1 с помощью MinMaxScaler.

## 1.3.3 Поиск гиперпараметров по сетке

Класс GridSearchCV из библиотеки scikit-learn (sklearn) представляет собой инструмент для выполнения поиска по сетке (Grid Search) гиперпараметров модели машинного обучения. Он автоматически перебирает различные комбинации заданных гиперпараметров модели, выполняет обучение моделей с каждой комбинацией на тренировочной выборке, и оценивает их производительность с использованием встроенной перекрестной проверки (cross-validation)..

## 1.3.4 Метрики качества моделей

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

* R2 или коэффициент детерминации измеряет долю дисперсии, объясненную моделью, в общей дисперсии целевой переменной.
* 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;
* MAE (Mean Absolute Error) - средняя абсолютная ошибка так же принимает значениях 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

По условию первой части задачи целевые признаки – это модуль упругости при растяжении и прочность при растяжении. Для выбора модели выделим пока один признак – модуль упругости при растяжении. Разделим датасет на тренировочную и тестовую выборки. По условию задачи тестовая выборка составляет 30%, тренировочная – 70%. С помощью MinMaxScaler приводим признаки к одному масштабу. Выбрав модель и подобрав гиперпараметры, применим эту модель и для прогнозирования целевой переменной «Прочность при растяжении»

Во второй части задачи предсказываем значения переменной «Соотношение матрица-наполнитель» с помощью нейронной сети. Перед этим мы также разделим датасет на выборки и отмасшатабируем данные.

## 2.2 Разработка и обучение моделей для прогнозирования модуля упругости при растяжении. Тестирование моделей.

Для подбора лучшей были взяты следующие модели:

* LinearRegression - линейная регрессия
* SVR - метод опорных векторов
* KneighborsRegressor - метод ближайших соседей
* GradientBoosting – градиентный бустинг
* AdaBoostRegressor – адаптивный бустинг
* RandomForestRegressor - случайный лес

В качестве метрики возьмём коэффициент детерминации. Коэффициент детерминации (R^2) используется для оценки качества предсказаний модели регрессии и измеряет долю объясненной вариации зависимой переменной моделью. R^2 принимает значения от 0 до 1, где 0 означает, что модель не объясняет вариацию целевой переменной, а 1 означает, что модель объясняет всю вариацию целевой переменной.

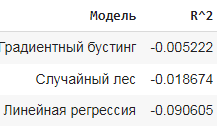


Рисунок 9.

Три модели выдали близкий к нулю, но отрицательный коэффициент детерминации. Это говорит о том, что модели работают даже чуть хуже, чем простое усреднение. Для улучшения результатов попробуем использовать модуль GridSearchCV из библиотеки sklearn для автоматического подбора оптимальных гиперпараметров моделей регрессии.

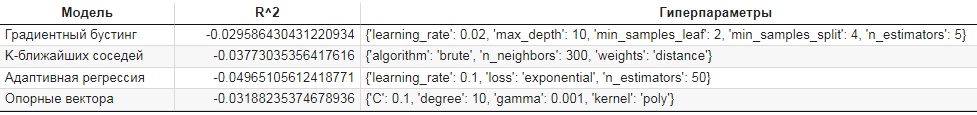


Рисунок 10.

Видим, что заметных улучшений нет.

Теперь применим модель регрессии для прогнозирования двух целевых переменных («Прочность при растяжении» и «Модуль упругости при растяжении»). Перед этим выделим эти два признака из общего ранее очищенного датасета, и ещё раз проведём разбиение на тренировочную и тестовую выборку с фиксированным случайным состоянием (random\_state). В качестве возьмём модель градиентного бустинга с ранее подобранными гиперпараметрами. Значения метрик «Средняя абсолютная ошибка», «Средняя квадратичная ошибки» и «Коэффициент детерминации» представлены на рисунках 11-12.



Рисунок 11

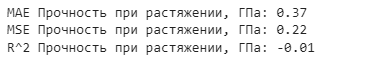


Рисунок 12

Теперь визуализируем наши результаты на диаграммах «BoxPlot» и «ScatterPlot»:

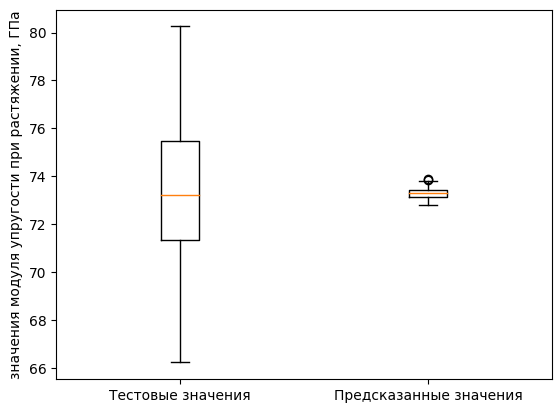
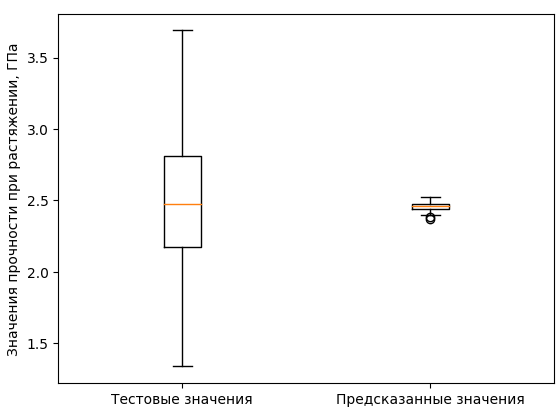


Рисунок 13-14. Результаты на диаграмме BoxPlot

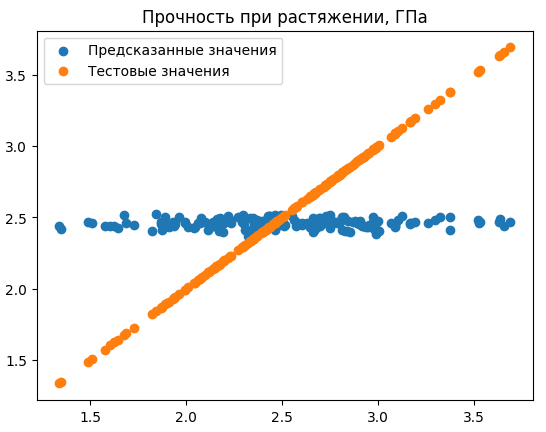
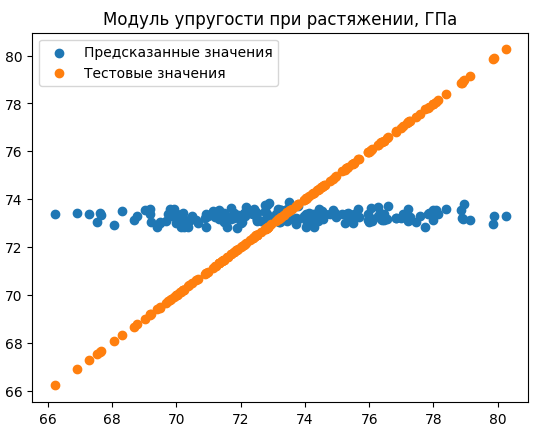


Рисунок 15-16. Результаты на диаграмме ScatterPlot

На рисунках 13-16 чётко видно, что предсказанные нашей моделью значения сосредоточены около среднего значения признака.

Подводя итог, мы можем сказать, что результаты неудовлетворительны. Подбор гиперпараметров также не помог решить поставленную задачу. Ни одна из выбранных моделей не оказалась подходящей для наших данных. Специалисту в предметной области эти модели в работе не помогут.

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

## Для решения задачи регрессии мы разработаем три многослойных перцептронных нейронных сети. Для этого мы используем библиотеку scikit-learn и оптимизатор Optuna.

## Для подготовки данных мы используем функцию train\_test\_split, которая разделяет данные на тренировочный и тестовый наборы, и объект StandardScaler, который производит стандартизацию данных.

## Затем используем библиотеку Optuna для оптимизации гиперпараметров. Внутри функции objective для оптимизации мы генерируем гиперпараметры для каждой нейронной сети, такие как количество слоев, количество нейронов в каждом слое, функция активации, скорость обучения и оптимизатор.

## Далее создаем три объекта MLPRegressor, каждый из которых представляет многослойную перцептронную нейронную сеть для решения задачи регрессии. Каждый объект инициализируется с помощью гиперпараметров, сгенерированных ранее, и обучается на тренировочном наборе данных. Затем каждая модель используется для предсказания на тестовом наборе данных.

## Для первой модели мы вычисляем функцию потерь с помощью mean\_absolute\_error, которая сравнивает истинные значения целевой переменной с предсказанными значениями. Для второй модели мы используем mean\_squared\_error, а для третьей - коэффициент детерминации (R^2).

## Для каждой модели мы создаем объект Optuna study, который используем для оптимизации параметров нейронной сети. Функция create\_study принимает аргумент direction, который указывает, нужно ли минимизировать или максимизировать функцию потерь. В нашем случае мы минимизируем среднюю абсолютную ошибку для первой модели, среднеквадратичную ошибку для второй модели и максимизируем коэффициент детерминации для третьей модели.

## После того, как study создан, мы вызываем его метод optimize, который запускает оптимизацию. В данном случае число испытаний равно 100.

## После завершения оптимизации получаем набор оптимальных гиперпараметров для каждой модели, которые используются для создания и обучения новых моделей с помощью MLPRegressor. Затем каждая модель используется для предсказания на тестовом наборе данных.

* 1. Наконец, с помощью библиотеки Matplotlib строим диаграммы с истинными значениями целевой переменной и предсказанными значениями на тестовом наборе данных для каждой модели. Результаты работы представлены на рисунках

## 

Рисунок 17

## 

## 

Рисунок 18

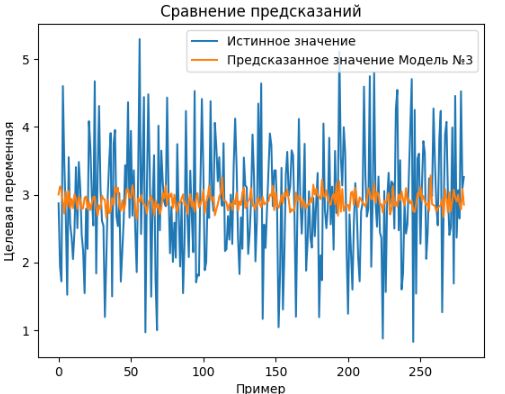


Рисунок 19

## 2.4. Разработка приложения

По условию задачи, сохранённую модель нейронной сети нужно использовать в приложении, которое будет предсказывать значение целевой переменной «Расчёт соотношения матрица-наполнитель»

Веб-приложение разработано с помощью языка Python, фреймворка Flask и шаблонизатора Jinja. Для демонстрации его работы используем инструмент ngrok, который позволяет создавать временные туннели для локальных веб-серверов.

Ngrok создает защищенное соединение между локальным сервером и интернетом, что позволит тестировать и демонстрировать наше веб-приложение в режиме реального времени. На рисунке 20 представлен скриншот интерфейса нашего приложения

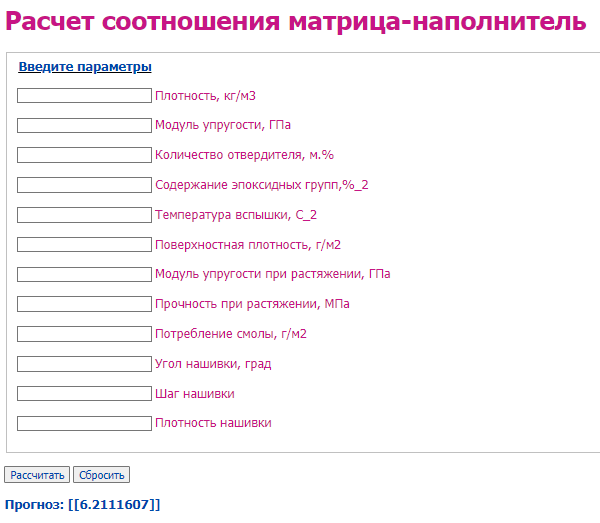


Рисунок 20

## 2.5. Создание удаленного репозитория

Для данного исследования был создан удаленный репозиторий на GitHub, который находится по адресу <https://github.com/Evgeny-Lyovin/VKR-repository/tree/main>. На него были загружены результаты работы: исследовательский notebook, код приложения, презентация, пояснительная записка

## Заключение

В рамках работы на тему "Прогнозирование конечных свойств композиционных материалов" были изучены теоретические основы и методы решения задачи. С помощью библиотеки pandas\_profiling составлен отчёт и был проведен разведочный анализ предложенных данных, анализ и исключение выбросов, а также проверку наличия пропусков. Затем данные были предварительно обработаны, включая удаление шумов и нормализацию.

* Было обучено несколько моделей для прогнозирования модуля упругости при растяжении прочности при растяжении и проведен поиск гиперпараметров с помощью GreedSearchCV.
* Была разработана нейронная сеть, которая рекомендует соотношение матрица-наполнитель и выполнен поиск гиперпараметров с помощью библиотеки Optuna
* Кроме того, было создано приложение, которое использует ранее обученную и сохраненную модель для выдачи прогнозов.

# Библиографический список

1 Документация по языку программирования python: – Режим доступа: <https://docs.python.org/3.8/index.html>.

2 Документация по библиотеке numpy: – Режим доступа: <https://numpy.org/doc/1.22/user/index.html#user>.

3 Документация по библиотеке pandas: – Режим доступа: <https://pandas.pydata.org/docs/user_guide/index.html#user-guide>.

4 Документация по библиотеке matplotlib: – Режим доступа: <https://matplotlib.org/stable/users/index.html>.

5 Документация по библиотеке seaborn: – Режим доступа: <https://seaborn.pydata.org/tutorial.html>.

6 Документация по библиотеке sklearn: – Режим доступа: <https://scikit-learn.org/stable/user_guide.html>.

7 Документация по библиотеке keras: – Режим доступа: <https://keras.io/api/>.

8 Руководство по быстрому старту в flask: – Режим доступа: <https://flask-russian-docs.readthedocs.io/ru/latest/quickstart.html>

9 ГрасД. Г77. Data Science. Наука о данных с нуля: Пер. с англ. - 2-е изд., перераб. и доп. - СПб.: БХВ-Петербург, 2021. - 416 с.