МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Московский государственный технический университет имени Н.Э. Баумана

(национальный исследовательский университет)»

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**по курсу**

«Data Science»

**Тема:** «**Прогнозирование конечных свойств новых материалов**

**(композиционных материалов)»**

Слушатель Алексеева Александра Евгеньевна

# 

Москва, 2022

# **Содержание**

[Содержание 2](#_Toc122703389)

[Введение 3](#_Toc122703390)

[1. Аналитическая часть 4](#_Toc122703391)

[1.1. Постановка задачи 4](#_Toc122703392)

[1.2. Описание используемых методов 7](#_Toc122703393)

[1.3. Разведочный анализ данных 18](#_Toc122703394)

[2. Практическая часть 21](#_Toc122703395)

[2.1. Разведочный анализ и предобработка данных 21](#_Toc122703396)

[2.2. Разработка и обучение модели 35](#_Toc122703397)

[2.3. Тестирование модели 39](#_Toc122703398)

[2.4. Создание нейронной сети для прогнозирования 43](#_Toc122703399)

[2.5. Создание удалённого репозитория 50](#_Toc122703400)

[2.6. Заключение 50](#_Toc122703401)

[2.7. Приложения 51](#_Toc122703402)

# **Введение**

**Композиционный материа́л** (КМ), сокращённо **компози́т** — многокомпонентный материал, изготовленный (человеком или природой) из двух или более компонентов с существенно различными физическими и/или химическими свойствами, которые, в сочетании, приводят к появлению нового материала с характеристиками, отличными от характеристик отдельных компонентов и не являющимися простой их суперпозицией. В составе композита принято выделять матрицу и наполнитель, последний выполняют функцию армирования (по аналогии с арматурой в таком композиционном строительном материале, как железобетон). В качестве наполнителей композитов как правило выступают углеродные или стеклянные волокна, а роль матрицы играет полимер. Сочетание разных компонентов позволяет улучшить характеристики материала и делает его одновременно лёгким и прочным. При этом отдельные компоненты остаются таковыми в структуре композитов, что отличает их от смесей и затвердевших растворов. Варьируя состав матрицы и наполнителя, их соотношение, ориентацию наполнителя, получают широкий спектр материалов с требуемым набором свойств. Многие композиты превосходят традиционные материалы и сплавы по своим механическим свойствам и в то же время они легче. Использование композитов обычно позволяет уменьшить массу конструкции при сохранении или улучшении её механических характеристик.

Поскольку конечные свойства композитов зависят от многих факторов – свойств входящих материалов, технологии их обработки, кроме того, один и тот же набор входящих данных не определяет однозначно то или иное конечное свойство – задача однозначного получения тех или иных конечных свойств достаточно актуальна.

Одним из вариантов решения этой задачи является прогнозирование конечных свойств композитов как наименее трудозатратный способ. Суть прогнозирования состоит в создании прогностической модели, способной с заданной точностью определить то или иное конечное свойство по набору входящих свойств.

Данная работа является попыткой создать наиболее оптимальную модель для прогнозирования следующих конечных свойств композитов:

* Прочность при растяжении;
* Модуль упругости при растяжении;
* Соотношение «матрица-наполнитель».

Данная задача прогнозирования является задачей регрессии в машинном обучении.

Для прогнозирования первых двух свойств в работе применялись различные методы регрессии, для прогнозирования оптимального значения соотношения «матрица-наполнитель» применялся метод нейронной сети.

Для удобства использования одной из созданных прогностических моделей на практике в рамках данной работы создано приложение.

1. **Аналитическая часть**
   1. **Постановка задачи**

Цель работы: разработка модели для прогноза модуля упругости при растяжении, прочности при растяжении и соотношения «матрица-наполнитель».

Для достижения поставленной цели разбиваем работу на следующие этапы:

* Предобработка данных;
* Разведочный анализ данных;
* Разработка и обучение модели;
* Тестирование модели.

Для выполнения работы даны два файла:

* X\_bp.xlsx (параметры базальтопластика): 1023 строк, 11 колонок;

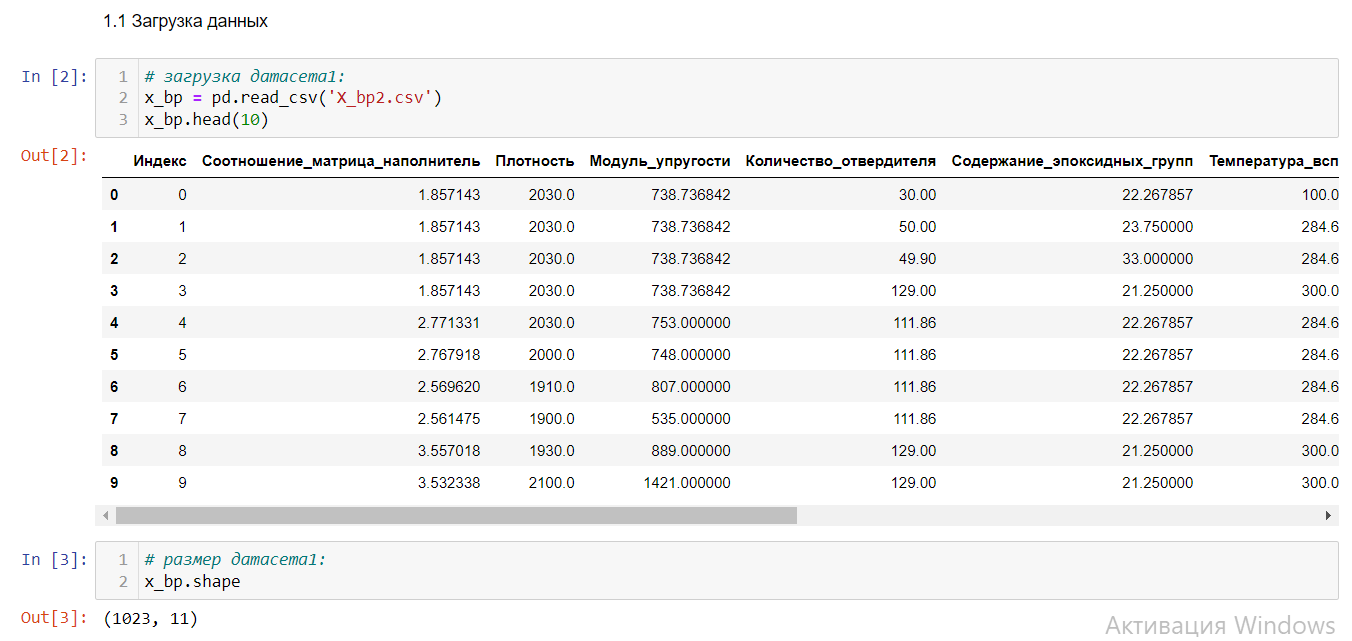


Рисунок 1 - получение данных файла X\_bp.xlsx

* X\_nup.xlsx (нашивки углепластика): 1040 строк, 4 колонки;

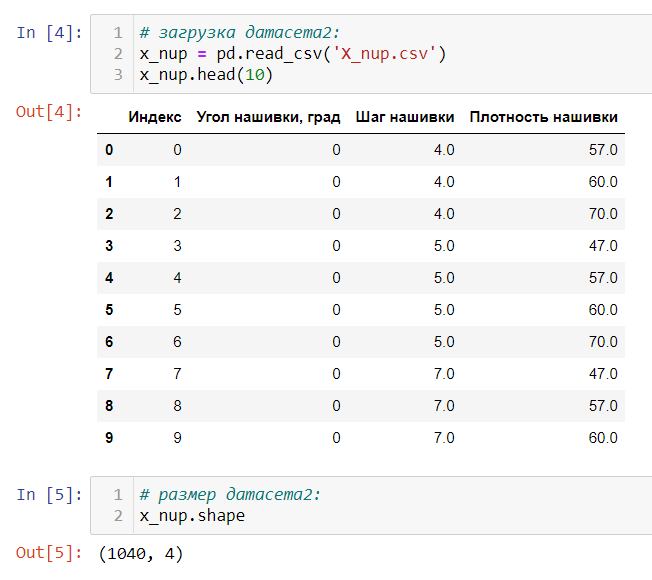


Рисунок 2 - получение данных файла X\_nup.xlsx

Выводы: Во втором файле количество строк меньше, чем в первом. Используем внутреннее соединение таблиц этих двух датасетов для того, чтобы получить только те данные, которые есть как в первом датасете, так и во втором.

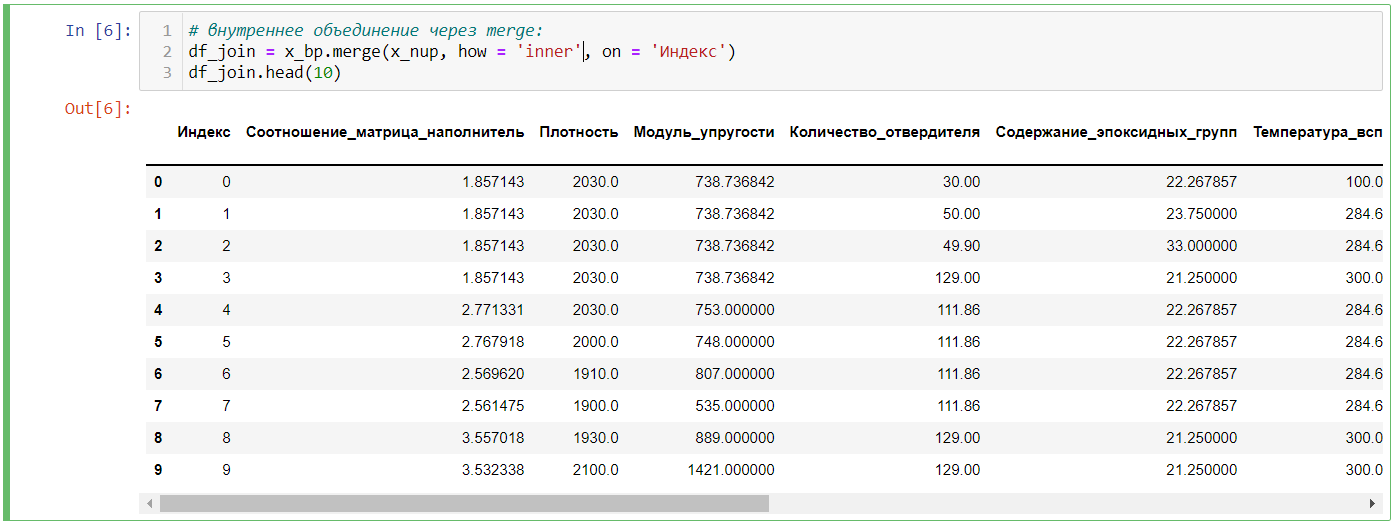


Рисунок 3 - объединение датасетов в один

Выводы: после внутреннего соединения 17 строк датасета x\_nup не вошли в результирующий датасет df\_join, поскольку они не найдены по заданному условию в датасете x\_bp. Кроме того, в результирующем датасете имеем столбец "Индекс", который не несет полезной информации в рамках решаемой задачи, поэтому необходимо его удалить.

После удаления снова смотрим объем данных:

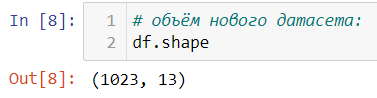


Рисунок 4 - объем данных после удаления неинформативных столбцов

В целях организации работы предполагается пользоваться планом работ (см. Приложение 1).

Проверяем данные на наличие пропусков и смотрим типы данных:

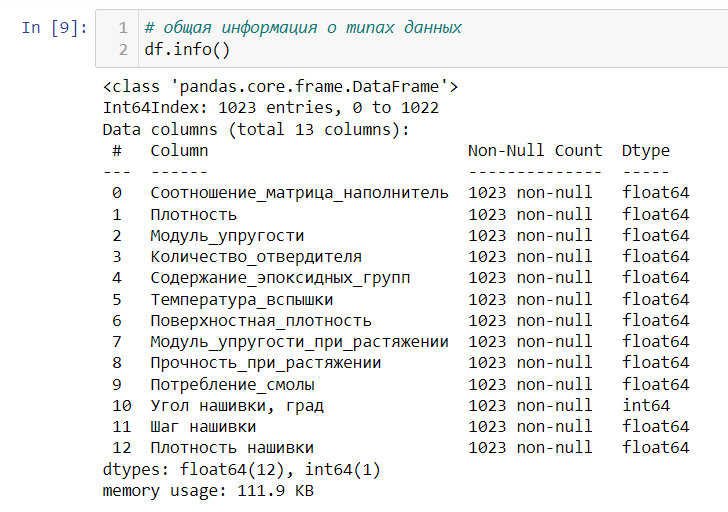


Рисунок 5 - общая информация о датасете

Выводы: 1) все колонки, кроме "Угол нашивки, град", имеют формат числа с плавающей запятой объемом 64 бита, качественных значений нет; 2) пропуски отсутствуют, поэтому действия по удалению пропусков производить не требуется.

Полученный датасет является готовым для этапа разведочного анализа данных.

* 1. **Описание используемых методов**

Поставленная задача регрессии в соответствии с классификацией категорий задач машинного обучения относится к задаче машинного обучения с учителем.

Основная **цель** **машинного** **обучения** заключается в том, что можно представить реальность, используя математическую функцию, которую алгоритм не знает заранее, но которую он может определить после просмотра некоторых данных — всегда в форме парных входов и выходов. То есть каждый алгоритм **машинного** **обучения** построен на модифицируемой математической функции.

Выбор алгоритма строго зависит от размера и структуры данных. Таким образом, выбор правильного алгоритма может быть неясен до тех пор, пока мы не проверим возможные варианты и не наткнёмся на ошибки.

Эффективность того или иного алгоритма определяется, в основном, значением функции потерь – чем меньше потери, тем точнее предсказывает алгоритм. Исходя из этих соображений в работе использовались различные по своей сути алгоритмы, зарекомендовавшие себя как наиболее эффективные в части решения задач регрессии:

* Линейная регрессия;
* Случайный лес;
* К-ближайших соседей;
* Дерево решений;
* Метод опорных векторов;
* Градиентный бустинг.

**Линейная регрессия (Linear Regression)**.

Линейная регрессия – один из наиболее известных и понятных алгоритмов в статистике и машинном обучении.

Линейную регрессию можно представить в виде уравнения, которое описывает прямую, наиболее точно показывающую взаимосвязь между входными переменными Х и выходными переменными Y. Для составления этого уравнения нужно найти определённые коэффициенты B для входных переменных.

Например: Y = B0 + B1 \* X

Зная X, мы должны найти Y, и цель линейной регрессии заключается в поиске значений коэффициентов B0 и B1.

Перед применением линейной регрессии рекомендуется: убрать похожие (коррелирующие) переменные и избавьтесь от шума в данных, если это возможно. Линейная регрессия — быстрый и простой алгоритм, который хорошо подходит в качестве первого алгоритма для изучения.

Линейная регрессия - отличный инструмент для анализа взаимосвязей между переменными, но она не рекомендуется для большинства практических приложений, потому что она чрезмерно упрощает реальные проблемы, предполагая линейную взаимосвязь между переменными.

В данной работе линейная регрессия применяется больше в качестве альтернативного метода, нежели метода для практического решения задачи.

**Случайный лес (Random Forest).**

Это метод коллективного обучения для классификации, регрессии и других задач, который работает путем построения множества деревьев решений во время обучения. В задачах классификации результатом случайного леса является класс, выбранный большинством деревьев. Для задач регрессии возвращается среднее или среднее предсказание отдельных деревьев. Леса случайных решений соответствуют привычке деревьев решений перестраиваться под их обучающий набор. Случайные леса обычно превосходят деревья решений, но их точность ниже, чем у деревьев с градиентным усилением.

В алгоритме случайного леса для всех выборок из тренировочных данных строятся деревья решений. При построении деревьев для создания каждого узла выбираются случайные признаки. В отдельности полученные модели не очень точны, но при их объединении качество предсказания значительно улучшается.

Преимущества алгоритма:

* Универсальность: алгоритм можно использовать как для задач регрессии, так и для задач классификации.
* Производительность: способность обрабатывать большие наборы данных с высокой размерностью.
* Повышение точности модели и предотвращение проблем переобучения.

Недостатки алгоритма:

* Не так эффективен при решении задач регрессии, как при решении задач классификации.
* Для более точного прогноза требуется больше деревьев, что приводит к более медленной модели.
* Не подходит для анализа и описания взаимосвязей переменных.

В данной работе алгоритм выбран с надежной на то, что с нашей задачей регрессии он справится.

**К-ближайших соседей (K-Nearest Neighbour).**

 Алгоритм k-ближайших соседей (k-NN) представляет собой непараметрический метод контролируемого обучения. Он используется как для классификации, так и для регрессии. В обоих случаях входные данные состоят из k ближайших обучающих примеров в наборе данных. Результат зависит от того, используется ли k-NN для классификации или регрессии: в регрессии k-NN результатом является значение свойства для объекта. Это значение является средним из значений k ближайших соседей.

Соседи берутся из набора объектов, для которых известно значение свойства объекта (для регрессии *k*-NN). Это можно рассматривать как обучающий набор для алгоритма, хотя никакого явного шага обучения не требуется.

Особенностью алгоритма *k*-NN является то, что он чувствителен к локальной структуре данных. Для повышения точности алгоритма, как правило, требуется нормализация обучающих данных.

Преимущества метода:

* Простота использования полученных результатов.
* Решения не уникальны для конкретной ситуации, возможно их использование для других случаев.
* Целью поиска является не гарантированно верное решение, а лучшее из возможных.

Недостатки метода:

* Данный метод не создает каких-либо моделей или правил, обобщающих предыдущий опыт, – в выборе решения они основываются на всем массиве доступных исторических данных, поэтому невозможно сказать, на каком основании строятся ответы.
* Существует сложность выбора меры "близости" (метрики). От этой меры главным образом зависит объем множества записей, которые нужно хранить в памяти для достижения удовлетворительной классификации или прогноза. Также существует высокая зависимость результатов классификации от выбранной метрики.
* Чувствителен к масштабу данных, а также к неинформативным признакам.
* Для применения алгоритма необходимо, чтобы метрическая близость объектов совпадала с их семантической близостью, чего не всегда просто добиться.
* При использовании метода возникает необходимость полного перебора обучающей выборки при распознавании, следствие этого - вычислительная трудоемкость.

Типичные задачи данного метода - это задачи небольшой размерности по количеству классов и переменных.

В данной работе алгоритм выбран из соображений, что негативное влияние большинства недостатков метода удастся нивелировать за счет правильной подготовки данных и за счет нахождения оптимальных гиперпараметров алгоритма.

**Дерево решений (Decision Tree)**

Дерево решений представляет собой иерархическую древовидную структуру, состоящую из правила вида «Если …, то ...». За счет обучающего множества правила генерируются автоматически в процессе обучения.

В отличие от нейронных сетей, деревья как аналитические модели проще, потому что правила генерируются на естественном языке за счет обобщения множества отдельных наблюдений (обучающих примеров), описывающих предметную область. В обучающем множестве для примеров должно быть задано целевое значение, так как деревья решений — модели, создаваемые на основе обучения с учителем.

Дерево решений — метод представления решающих правил в определенной иерархии, включающей в себя элементы двух типов — узлов (node) и листьев (leaf). Узлы включают в себя решающие правила и производят проверку примеров на соответствие выбранного атрибута обучающего множества.

Простой случай: примеры попадают в узел, проходят проверку и разбиваются на два подмножества:

* примеры, которые удовлетворяют установленное правило;
* примеры, которые не удовлетворяют установленное правило.

Далее к каждому подмножеству снова применяется правило, процедура повторяется. Это продолжается, пока не будет достигнуто условие остановки алгоритма. Последний узел, когда не осуществляется проверка и разбиение, становится листом.

Лист определяет решение для каждого попавшего в него примера. Для дерева классификации — это класс, ассоциируемый с узлом, а для дерева регрессии — соответствующий листу модальный интервал целевой переменной. В листе содержится не правило, а подмножество объектов, удовлетворяющих всем правилам ветви, которая заканчивается этим листом.

Пример попадает в лист, если соответствует всем правилам на пути к нему. К каждому листу есть только один путь. Таким образом, пример может попасть только в один лист, что обеспечивает единственность решения.

Основная задача при построении дерева решений — последовательно и рекурсивно разбить обучающее множество на подмножества с применением решающих правил в узлах. Процесс разбиения продолжают до того, пока все узлы в конце ветвей не станут листами.

Узел становится листом в двух случаях:

* естественным образом — когда он содержит единственный объект или объект только одного класса;
* после достижения заданного условия остановки алгоритм — например, минимально допустимое число примеров в узле или максимальная глубина дерева.

**Преимущества метода:**

* Формируют четкие и понятные правила классификации/ регрессии. То есть деревья решений хорошо и быстро интерпретируются;
* Способны генерировать правила в областях, где специалисту трудно формализовать свои знания;
* Легко визуализируются, то есть могут «интерпретироваться» не только как модель в целом, но и как прогноз для отдельного тестового субъекта (путь в дереве);
* Быстро обучаются и прогнозируют;
* Не требуется много параметров модели;
* Поддерживают как числовые, так и категориальные признаки;

**Недостатки метода:**

* Деревья решений чувствительны к шумам во входных данных. Небольшие изменения обучающей выборки могут привести к глобальным корректировкам модели, что скажется на смене правил классификации и интерпретируемости модели.
* Разделяющая граница имеет определенные ограничения, из-за чего дерево решений по качеству классификации уступает другим методам.
* Возможно переобучение дерева решений, из-за чего приходится прибегать к методу «отсечения ветвей», установке минимального числа элементов в листьях дерева или максимальной глубины дерева.
* Сложный поиск оптимального дерева решений: это приводит к необходимости использования эвристики типа жадного поиска признака с максимальным приростом информации, которые в конечном итоге не дают 100-процентной гарантии нахождения оптимального дерева.

В данной работе алгоритм выбран из-за подхода к поиску зависимостей путем применения правил естественного языка, что отличает его от других методов. Негативное влияние большинства недостатков метода предполагается нивелировать за счет правильной подготовки данных и за счет нахождения оптимальных гиперпараметров алгоритма.

**Метод опорных векторов (Support Vector Machine).**

В основе метода опорных векторов для задач регрессии или регрессии опорных векторов (SVR) лежит поиск гиперплоскости, при которой риск в многомерном пространстве будет минимальным. По сравнению с традиционной регрессионной моделью SVR оценивает коэффициенты путем минимизации квадратичных потерь. Так, если прогнозное значение попадает в область гиперплоскости, то потери равны нулю. В противном случае – разности прогнозного и фактического значений.

Регрессия опорных векторов поддерживает все интересные свойства машин опорных векторов. Учитывая точки данных, алгоритм пытается найти кривую. Однако вместо того, чтобы кривая действовала как граница принятия решения в задаче классификации, в SVR найдено соответствие между некоторым вектором и **положением** на кривой. А опорные векторы участвуют в поиске наиболее близкого соответствия между точками данных и фактической функцией, которая ими представлена. Интуитивно, когда мы максимизируем расстояние между **опорными векторами** до регрессирующей кривой, мы приближаемся к фактической кривой (потому что в статистических выборках всегда присутствует некоторый шум). Из этого также следует, что мы можем отбросить все векторы, которые не являются опорными векторами, предполагая, что они являются статистическими выбросами. Результатом действия алгоритма является регрессирующая функция.

В задачах регрессии метод опорных векторов не очень популярен из-за того, что подходит не под любую задачу. Однако в некоторых случаях он может оказаться эффективнее других методов.

Особое преимущество метода SVM для решения задач восстановления зависимостей составляет его применения для нелинейных зависимостей.

Преимущества метода:

* хорошо работает с пространством признаков большого размера;
* хорошо работает с данными небольшого объема;
* алгоритм максимизирует разделяющую полосу, которая, как подушка безопасности, позволяет уменьшить количество ошибок классификации;
* так как алгоритм сводится к решению задачи квадратичного программирования в выпуклой области, то такая задача всегда имеет единственное решение (разделяющая гиперплоскость с определенными гиперпараметрами алгоритма всегда одна).

Недостатки метода:

* долгое время обучения (для больших наборов данных);
* неустойчивость к шуму: выбросы в обучающих данных становятся опорными объектами-нарушителями и напрямую влияют на построение разделяющей гиперплоскости;
* не описаны общие методы построения ядер и спрямляющих пространств, наиболее подходящих для конкретной задачи в случае линейной неразделимости классов. Подбирать полезные преобразования данных – искусство.

В данной работе алгоритм выбран из-за возможности использования для нелинейных зависимостей.

**Градиентный бустинг ()**

**Градиентный бустинг** — это техника машинного обучения для задач классификации и регрессии, которая строит модель предсказания в форме ансамбля слабых предсказывающих моделей, обычно деревьев решений. Финальная модель является линейной композицией этих слабых предсказывающих моделей.

Градиентный бустинг использует следующие инструменты:

* Дерево решений, которое является моделью регрессии, связывающей значения целевой переменной с различными независимыми переменными посредством многократных двоичных разделений;
* Метод бустинга, представляющий собой адаптивный метод объединения нескольких простых моделей, таких как деревья принятия решений, для расчета более точных прогнозов.

Алгоритм комбинирует эти инструменты итеративно, выполняя следующие шаги:

1. Дерево решений формируется в соответствии с историческими данными.
2. Если отклонения в целевой переменной (например, истории продаж) объясняются не полностью, рассчитываются необъясненные остаточные ошибки, и в следующей итерации определяется новое дерево решений, которое объясняет отклонения остатков.
3. Этот процесс выполняется несколько раз, так что все меньше и меньше остаточных ошибок используется в качестве данных ввода в дополнительных деревьях.
4. Деревья принятия решений объединяются в окончательную модель, где они имеют определенный вес. Прогноз представляет взвешенную сумму прогнозов отдельных деревьев.

Преимущества алгоритма:

* Алгоритм можно использовать также в случаях, когда между предикторами может наблюдаться несколько зависимостей.
* Этот алгоритм может обрабатывать множество потенциальных предикторных переменных.
* Отсутствует необходимость в преобразованиях ввода, таких как применение логарифмической функции к некоторым независимым переменным.
* Нерелевантные предикторные переменные определяются автоматически и не влияют на прогноз.
* Этот алгоритм менее чувствителен к резко выделяющимся значениям, чем другие алгоритмы.

Недостатки алгоритма:

* Невысокая достоверность по сравнению с другими методами;
* Древовидная структура зависит от данных обучения, поэтому алгоритм не может дать хорошие прогнозы для случаев, когда будущие значения независимых переменных в значительной степени отличаются от соответствующих прошлых значений.
* Он часто требует множества вычислений, что может увеличить время выполнения.

В данной работе этот алгоритм был выбран из-за эффективной работы со множеством предикторных переменных, между которыми может быть несколько зависимостей, чего нет в других алгоритмах.

* 1. **Разведочный анализ данных**

Разведочный анализ данных, Exploratory Data Analysis (EDA) применяется для нахождения связей между переменными в ситуациях, когда отсутствуют (или недостаточны) априорные представления о природе этих связей. Как правило, при разведочном анализе учитывается и сравнивается большое число переменных, а для поиска закономерностей используются самые разные методы. Вычислительные методы разведочного анализа данных включают основные статистические методы, а также более сложные, специально разработанные методы многомерного анализа, предназначенные для отыскания закономерностей в многомерных данных. К основным методам разведочного статистического анализа относится процедура анализа распределений переменных (например, чтобы выявить переменные с несимметричным или негауссовым распределением, в том числе и бимодальные), просмотр корреляционных матриц с целью поиска коэффициентов, превосходящих по величине определенные пороговые значения, или анализ многовходовых таблиц частот.

Разведочный анализ данных проводится на начальных этапах проекта и имеет целью выявление: основных характеристик набора данных (датасета), взаимосвязей признаков (переменных), а также – сужения набора методов, используемых для создания модели машинного обучения.

В данной работе разведочный анализ проведен по следующим этапам:

* Проверка рабочего датасета на дубликаты и пропуски;
* Получение среднего значения и медианы для каждой переменной, анализ описательной статистики;
* Анализ распределения значений переменных;
* Анализ статистической зависимости переменных;
* Анализ аномалий и выбросов

В процессе анализа применялись визуальные и табличные способы вывода информации.

Анализ распределения значений переменных проводился посредством гистограммы распределения, являющейся графическим методом исследования характера рассеяния значений случайной величины. Основная решаемая с помощью этого метода задача – определение закона распределения значений переменных датасета. Результат исследования оказывает основное влияние на выбор того или иного алгоритма обучения.

Анализ статистической зависимости проводился посредством диаграмм попарного рассеяния точек (матрицы диаграмм рассеяния), которая показывает наличие или отсутствия корреляции между двумя переменными и, таким образом, дает качественное представление о характере корреляционной зависимости пар переменных. Количественная оценка степени статистической зависимости, которая является уточнением качественного показателя, производилась посредством вычисления значений следующих коэффициентов корреляции:

* **коэффициент ранговой корреляции Кендалла**: служит для измерения порядковой связи между двумя измеряемыми величинами и часто используется в качестве тестовой статистики при проверке статистической гипотезы, чтобы установить, можно ли считать две переменные статистически зависимыми. Этот тест является непараметрическим, поскольку он не основывается на каких-либо предположениях относительно распределений X или Y или распределения (X,Y). Связь между признаками можно признать статистически значимой, если значения коэффициентов ранговой корреляции Кендалла больше 0,5;
* **коэффициент линейной корреляции Пирсона**: применяется для исследования взаимосвязи двух переменных, измеренных в метрических шкалах на одной и той же выборке. Он позволяет определить, насколько пропорциональная изменчивость двух переменных. Коэффициент корреляции Пирсона чувствителен к выбросам. Одно аномальное значение может существенно исказить коэффициент. Поэтому перед проведением анализа следует проверить и при необходимости удалить выбросы. Если связь между признаками слабая или нелинейная, то значение коэффициента будет около 0.

По этим коэффициентам мы можем проверить, есть ли ранговая или линейная корреляционная связь между парами признаков (переменных).

Анализ аномалий и выбросов проводился посредством метода «Ящик с усами», который позволяет наглядно оценить основные характеристики распределения и выбросов.

На основании проведенного анализа определены задачи предобработки данных.

1. **Практическая часть**
   1. **Разведочный анализ и предобработка данных**

**Проверка рабочего датасета на дубликаты и пропуски**.

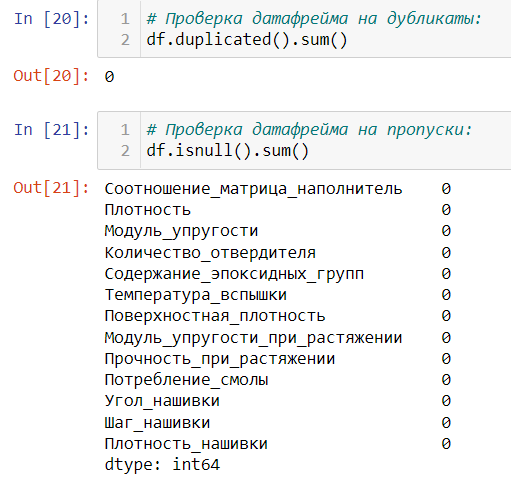


Рисунок 6 - проверка рабочего датасета на дубликаты и пропуски

Выводы: дубликатов и пропусков не обнаружено, можно приступать к статистическому анализу данных.

**Получение среднего значения и медианы для каждой переменной, анализ описательной статистики.**

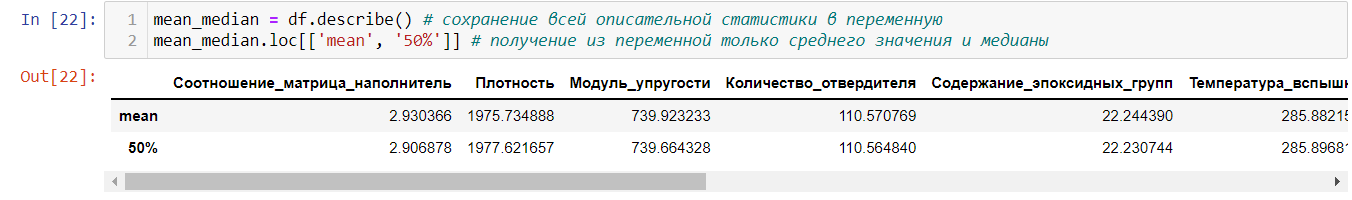


Рисунок 7 - получение среднего значения и медианы

Выводы: среднее и медианное значения каждой переменной практически совпадают, либо незначительно отклоняются друг от друга, что говорит в целом о нормальном характере распределения значений этих переменных. Исключение составляет переменная Поверхностная\_плотность, где отклонение между средним и медианой составляет около 6,4% среднего значения.

Произведем описательную статистику, которая позволит получить следующие данные:

* count - количество значений;
* mean - среднее значение;
* std - стандартное отклонение;
* min – минимум;
* 25% - верхнее значение первого квартиля;
* 50% - медиана;
* 75% - верхнее значение третьего квартиля;
* max – максимум

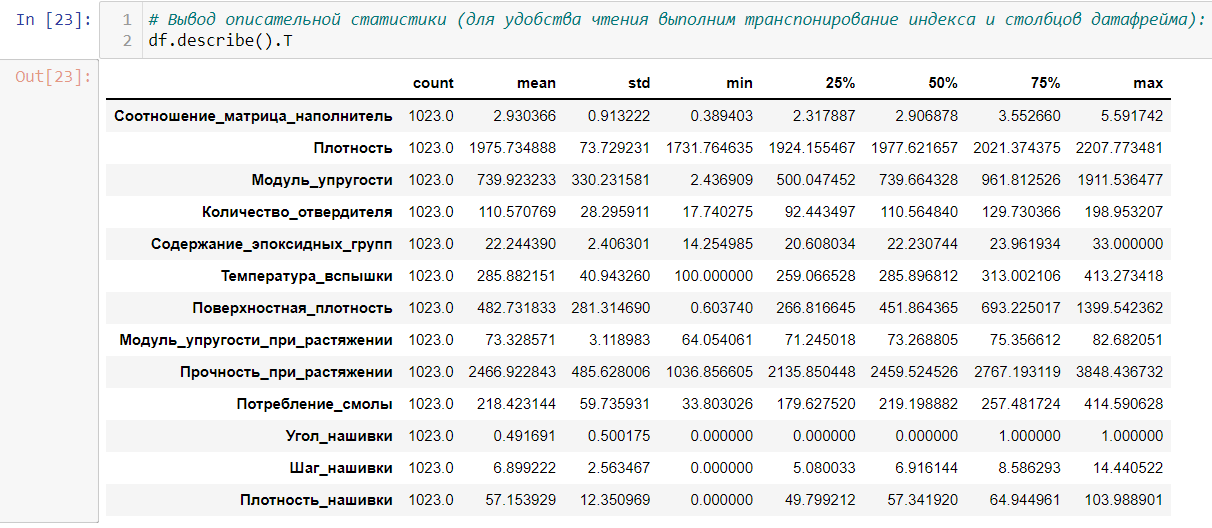


Рисунок 8 - получение описательной статистики

Выводы: Поскольку средние значения признаков имеют разный порядок, кроме того не входят в диапазон [0,1], это говорит о низкой сопоставимости их значений. В целях повышения сопоставимости данных необходимо масштабирование данных (приведение данных к заданному диапазону). Выбор алгоритма масштабирования будем производить позже.

**Анализ распределения значений переменных**.

Для определения характера распределения годится простая гистограмма, не требующая для построения много времени:

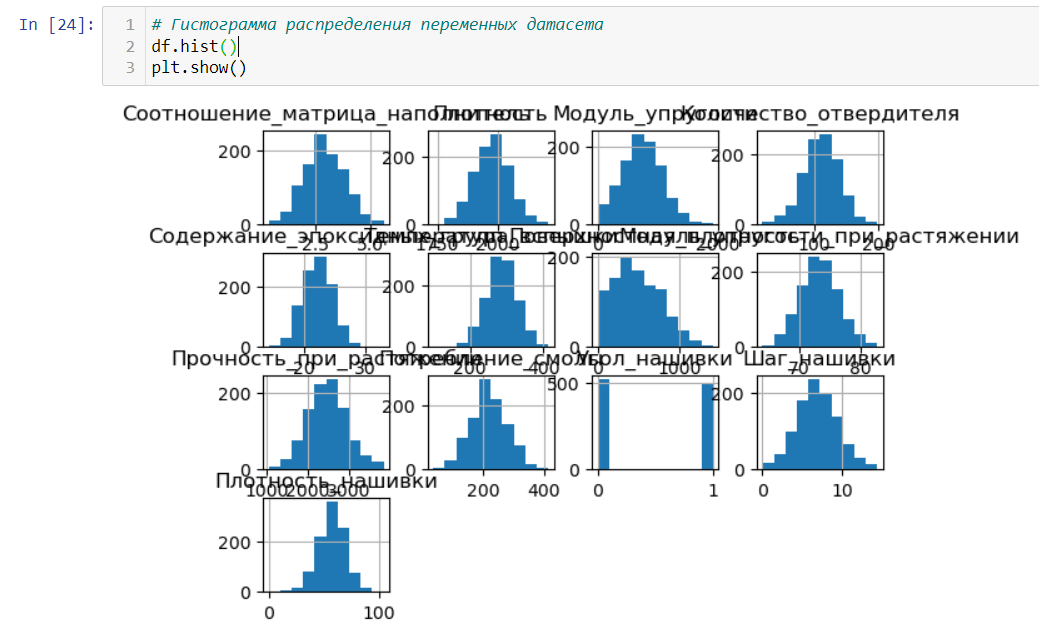


Рисунок 9 - гистограмма распределения

Такие гистограммы также можно построить отдельно для каждой переменной.

Выводы: целевые переменные распределены по нормальному закону с некоторой асимметрией, также как и большинство других переменных (все, кроме угла нашивки, поскольку там всего два возможных значения - 0 и 90).

**Анализ статистической зависимости переменных.**

Для наглядной оценки характера статистической зависимости между парами переменных строим простую матрицу диаграмм рассеяния:

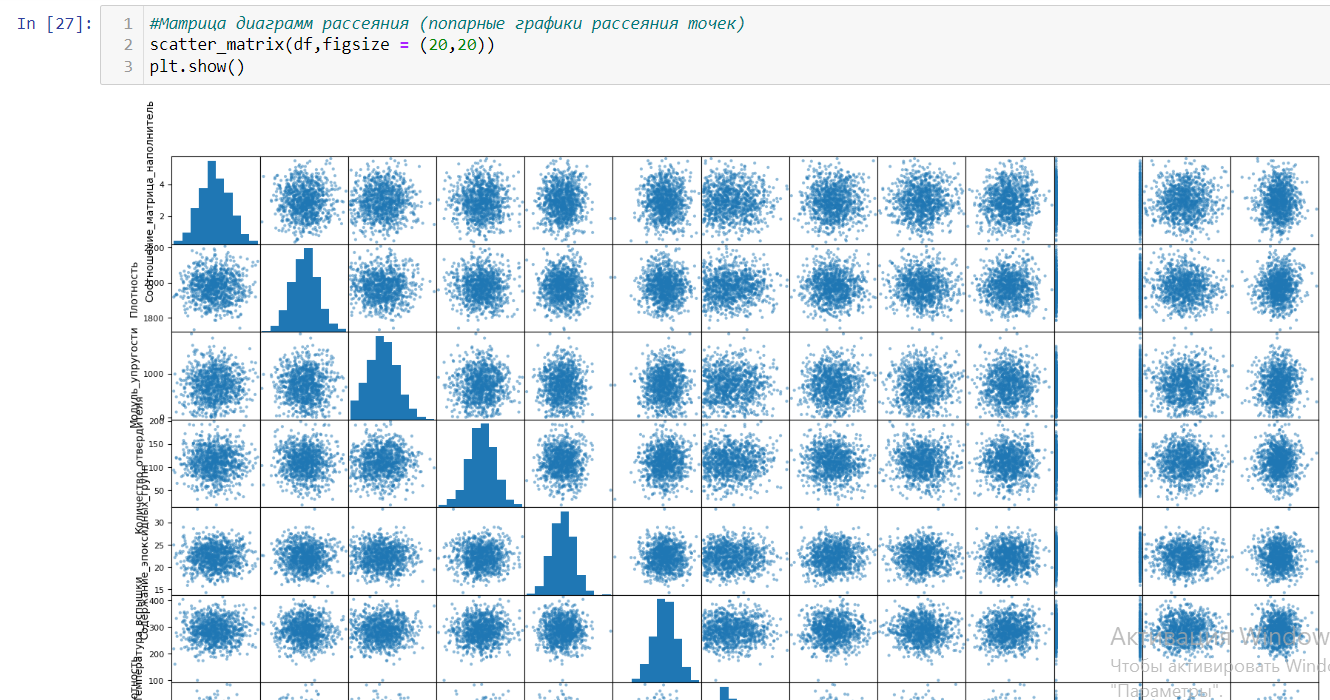


Рисунок 9 - матрица диаграмм рассеяния

Выводы: Ни у одной пары признаков не наблюдается корреляционной зависимости.

Проводим уточнение по коэффициентам корреляции:

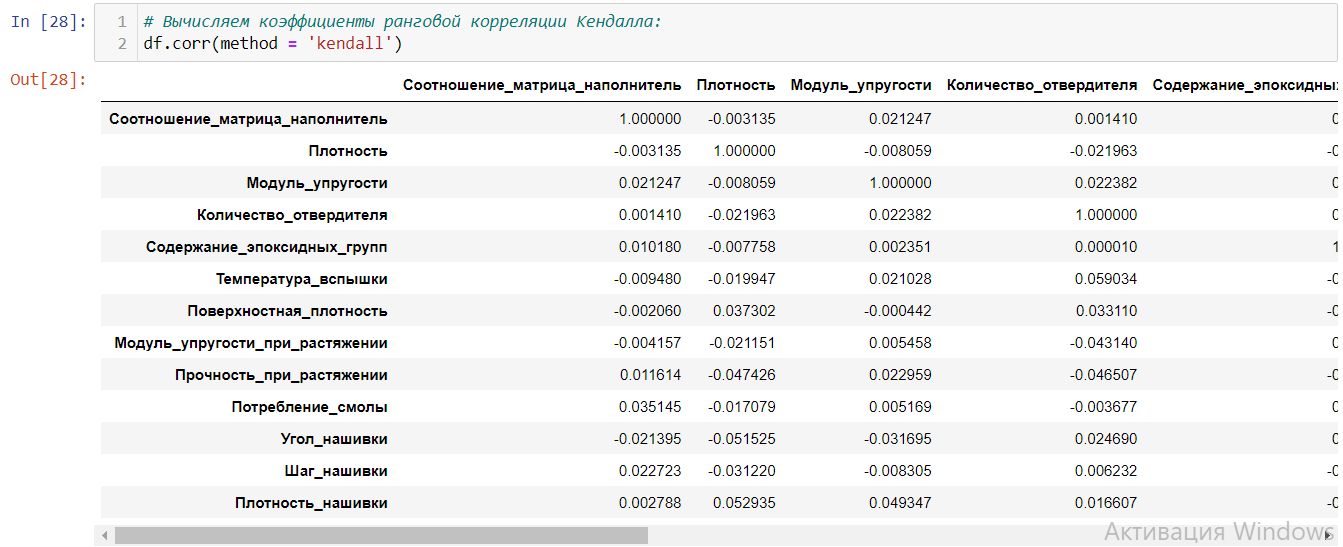


Рисунок 10 – коэффициент ранговой корреляции Кендалла

Выводы: по результатам применения метода ранговой корреляции Кендалла нулевая гипотеза об отсутствии статистической зависимости (ранговой корреляции) между парами переменных подтверждается, поскольку ни одна пара (кроме пар, образованных собственными значениями признака) не имеет значения по модулю, большее 0,5. Кроме того, все значения по модулю менее 0,1.

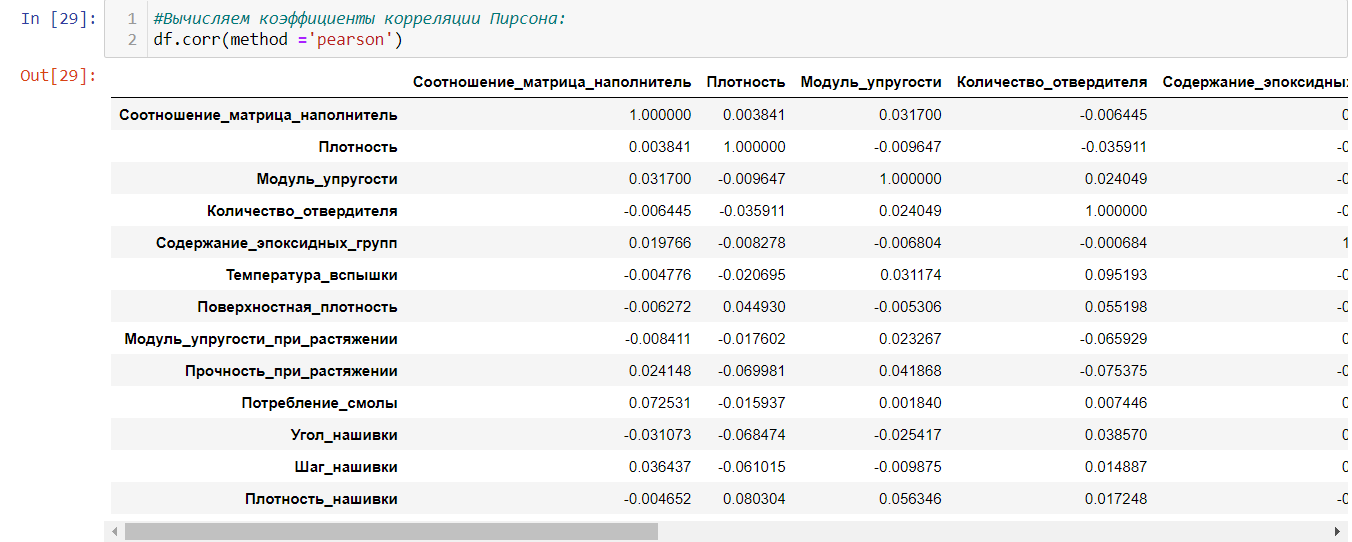


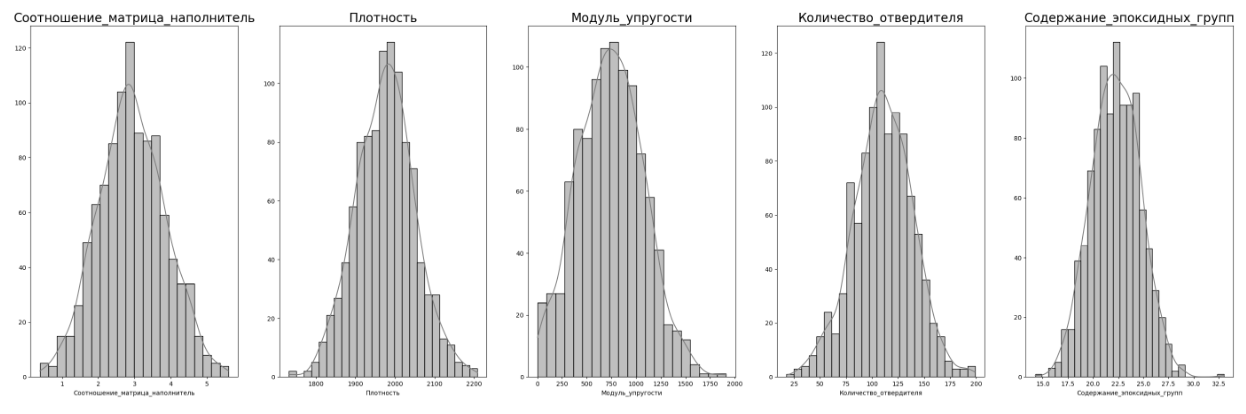
Рисунок 11 – коэффициент линейной корреляции Пирсона

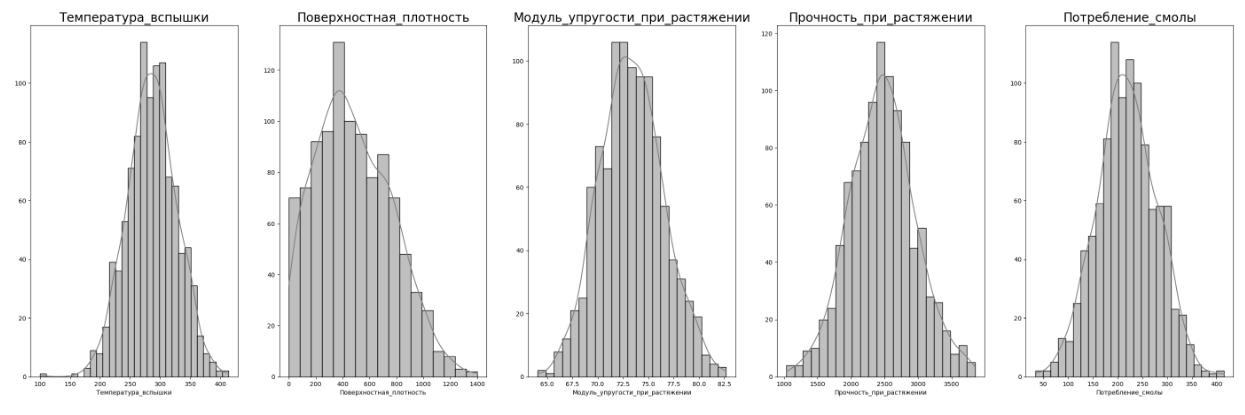
Выводы: по результатам применения метода линейной корреляции Пирсона нулевая гипотеза об отсутствии линейной статистической зависимости (линейной корреляции) между парами переменных подтверждается, поскольку значения коэффициентов всех пар (кроме пар, образованных собственными значениями признака) по модулю близки к нулю (не превышают 0,1). Поскольку можно утверждать, что отсутствует только линейная зависимость, то возможно, что имеет место быть нелинейная зависимость между переменными. Кроме того, входные данные не проверялись на наличие выбросов, поэтому данные выводы не могут быть однозначными.

**Анализ аномалий и выбросов.**

Для предварительной визуальной оценки воспользуемся вторым, более детальным, вариантом гистограммы распределения:







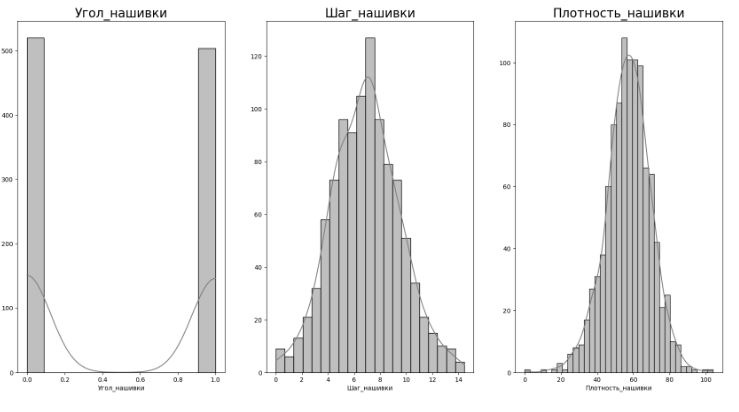


Рисунок 12 – второй вариант гистограммы распределения

Выводы: наблюдаются выбросы у признаков: Плотность, Содержание эпоксидных групп, Температура вспышки, Плотность нашивки. Имеет смысл более детально проанализировать выбросы по этим признакам посредством метода "ящик с усами".

Для более детального анализа распределения и выбросов используем метод Ящик с усами", который служит для визуализации распределения в выборке и поиска объектов-выбросов в данных. Поскольку текущие переменные сильно отличаются по прядку значений друг от друга и не нормированы, то они имеют низкую сопоставимость, что мешает вывести диаграммы "ящик с усами" по всем переменным на один график. Поэтому строим диаграммы для каждой переменной отдельно, используя данные графиков только для оценки (идентификации) выбросов, поскольку остальные показатели распределения были детально получены ранее. Для экономии времени используем простые средства визуализации (на рисунке 13 – пример по одной переменной, фактически этот метод применялся к каждой переменной):

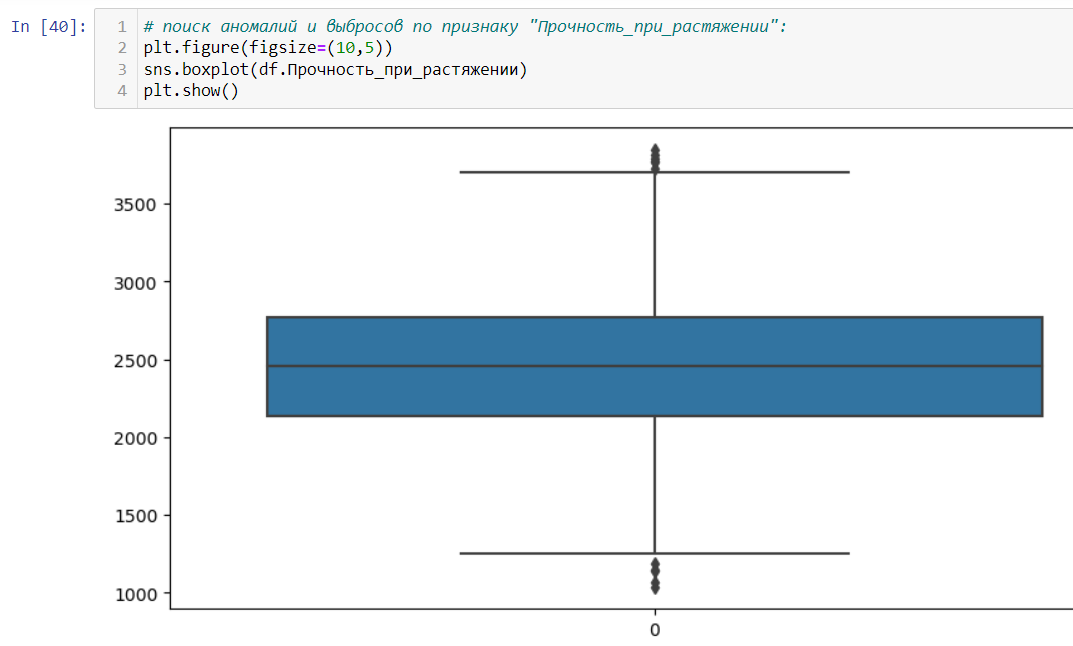


Рисунок 13 – пример построения «ящика с усами»

Выводы: выбросы визуализируются практически у всех переменных, имеет смысл вывести некоторую аналитическую сводку по выбросам.

Посмотрим количество и процент выбросов по признакам:

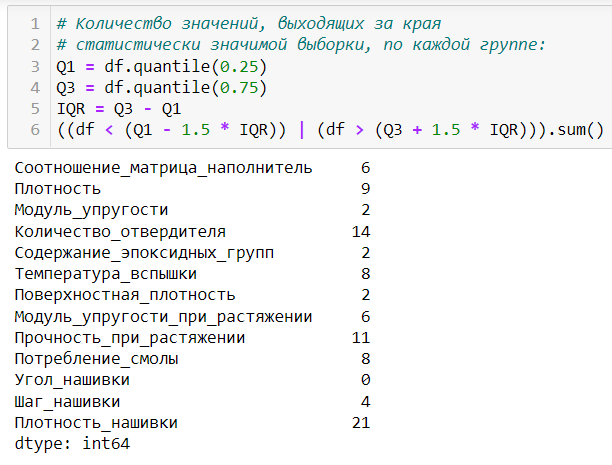
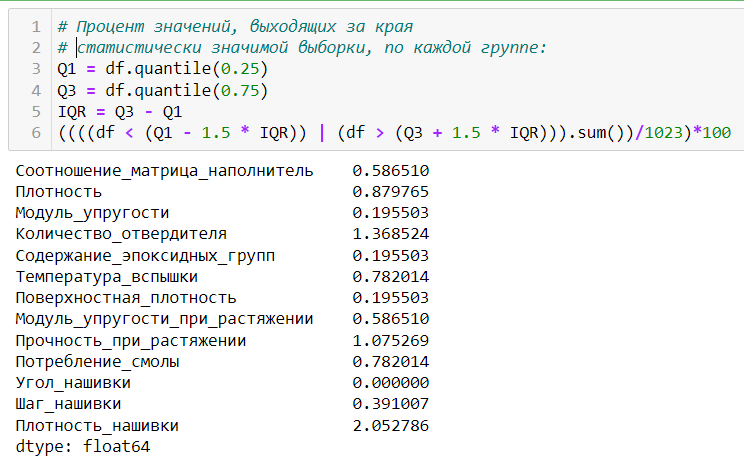
 

Рисунок 14 – аналитическая сводка по выбросам

Выводы: 1) Значения, выходящие за края статистически значимой выборки (границы усов), наблюдаются у всех переменных, кроме переменной "Угол нашивки" (не рассматривалась на графике ящик с усами"), поскольку она имеет всего два известных значения. 2) Поскольку свойства композитного материала не являются простой суперпозицией входящих в его состав компонентов, то значения, превышающие границы усов, возможно, относятся к аномалиям (которые также нужно учитывать для прогноза целевых признаков). 3) Наибольшее число отклонений имеют следующие переменные:

* Количество\_отвердителя: 14 значений из 1023 (что составляет 1,37 %);
* Прочность\_при\_растяжении: 11 значений из 1023 (что составляет 1,07 %);
* Плотность\_нашивки: 21 значение из 1023 (что составляет 2,05 %).

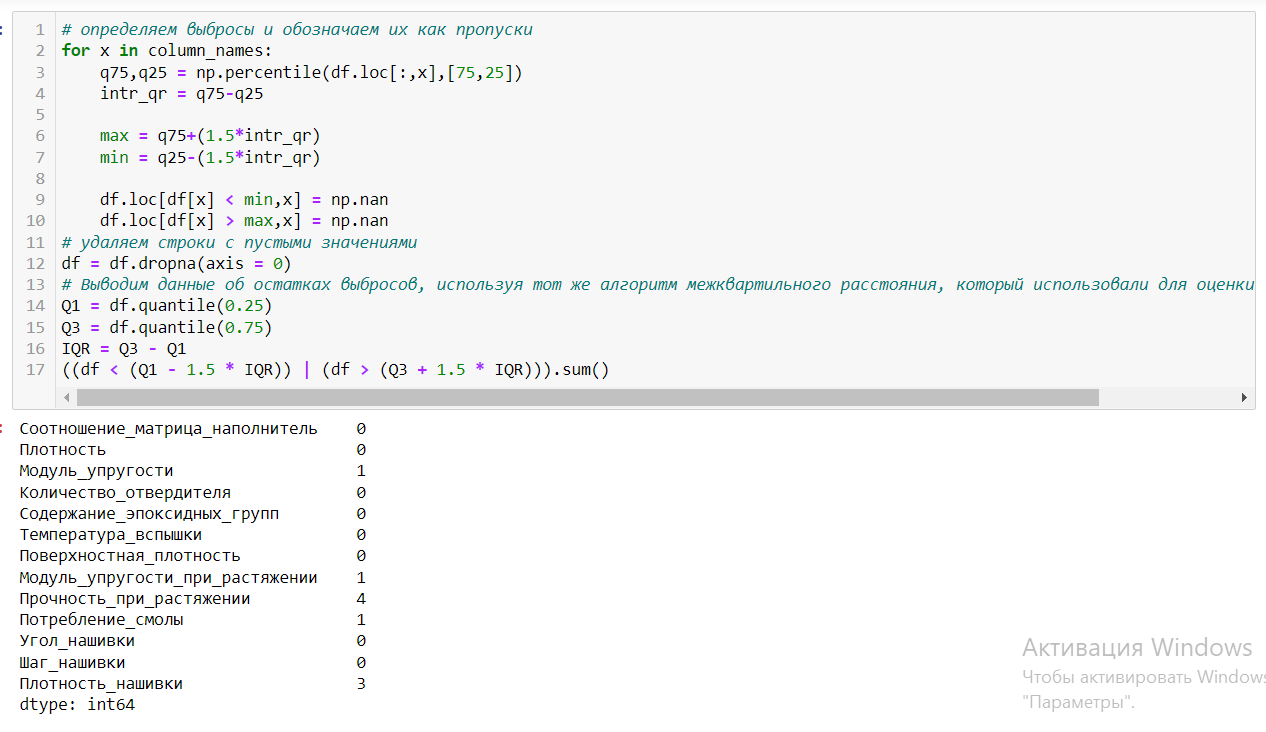
4) Наибольший размах в значении отклонений имеют следующие переменные:

* Плотность (около 50% длины правого уса)
* Содержание\_эпоксидных\_групп (около 75% длины правого уса) Потребление\_смолы (около 60% длины правого уса)
* Плотность\_нашивки (около 120% длины левого уса);

5) Общее число выбросов: 93.

Переходим к циклу "исключение выбросов - анализ очищенных от выбросов данных - проверка на наличие остатков выбросов - повторное исключение выбросов". Определяем выбросы как все значения, выходящие за края статистически значимой выборки. Есть несколько способов поиска выбросов данных. Все приведённые способы будут работать, поскольку выборка распределена нормально. Для удаления выбросов применяем метод межквартильных расстояний. В случае, если этот метод не даст необходимых результатов, можно попробовать другие методы.

Удаление выбросов производим в цикле, для каждой строки определяя выброс и обозначая его как пропуск:



Выводы: пустых значений нет, однако выбросы ещё остались, хотя их число значительно снизилось. Размер данных для работы сократился на 97 строк, что составляет 9,5% исходных данных. При изначально небольшой выборке потеря около 10% данных может привести к недостаточности данных для корректного обучения. Возможно, эффективнее было бы решение заменить значения выбросов на средние по переменной.

Поскольку число оставшихся выбросов незначительно, проанализируем их значения для оценки того, насколько наличие таких выбросов критично для решения задачи.

Для визуализации размаха оставшихся выбросов воспользуемся методом «ящик с усами» так же, как и ранее.

Выводы: поскольку оставшиеся выбросы находятся либо на границе усов, либо выходят за эти границы незначительно и, кроме того, количество выбросов крайне мало по сравнению с нормальными значениями переменных, принимается решение не исключать оставшиеся выбросы. Проведём разведочный анализ на очищенных данных в целях принятия решения о возможности перехода к масштабированию данных.

Повторный разведочный анализ проводим теми же средствами, что и первичный.

По результатам проведенного анализа имеем следующие выводы:

* 1. В целом после удаления выбросов линейная корреляция по-прежнему не наблюдается. Однако, значения несколько изменились, хотя, как и ранее, ни одно из значений не превышает по модулю величину 0,1. Поэтому эти изменения можно в рамках решаемой задачи принять как допустимые.
  2. Средние и медианные значения несколько изменились по сравнению с данными до исключения выбросов. По некоторым переменным средние значения приблизились к медианным больше, чем до исключения выбросов (особенно - по переменной Поверхностная\_плотность, где отклонение медианы от среднего снизилось с 6,4 до 4,6 процентов от среднего значения). Медиана угла нашивки перешла со значения 0 на значение 1, поскольку среднее стало несколько превышать величину 0,5 (ранее среднее значение было чуть меньше 0,5 и медиана принимала значение 0). Принимаем такие изменения как допустимые и переходим к проверке данных для масштабирования.
  3. пропусков нет, число строк ожидаемо уменьшилось, угол нашивки после преобразований стал иметь тип float вместо integer. Поэтому необходимо посмотреть, какие значения сейчас располагаются в этой колонке. Поскольку имеем по-прежнему только два значения - 0 и 1, поэтому преобразуем float64 обратно в integer.

Переходим к масштабированию данных.

Масштабирование применяется в целях нивелирования степени влияния переменных на результат анализа при машинном обучении, а также - в целях сопоставимости данных между переменными. Нивелирование степени влияния той или иной переменной необходимо из соображений того, что входные условия решаемой задачи не предполагают расстановки приоритетов между переменными (т.е. все переменные предполагаются равносильными по степени своего влияния на целевые данные). Поскольку на предыдущем этапе мы выяснили, что все переменные, за исключением угла нашивки (значения которой распределены по биномиальному закону), имеют нормальное или близкое к нормальному, распределение, то для масштабирования этих признаков можно применить следующие методы: а) Нормализация посредством алгоритма MinMaxScaler(); б) Стандартизация посредством алгоритма StandardScaler().

Предварительно проводим оценку плотности ядра (KDE):

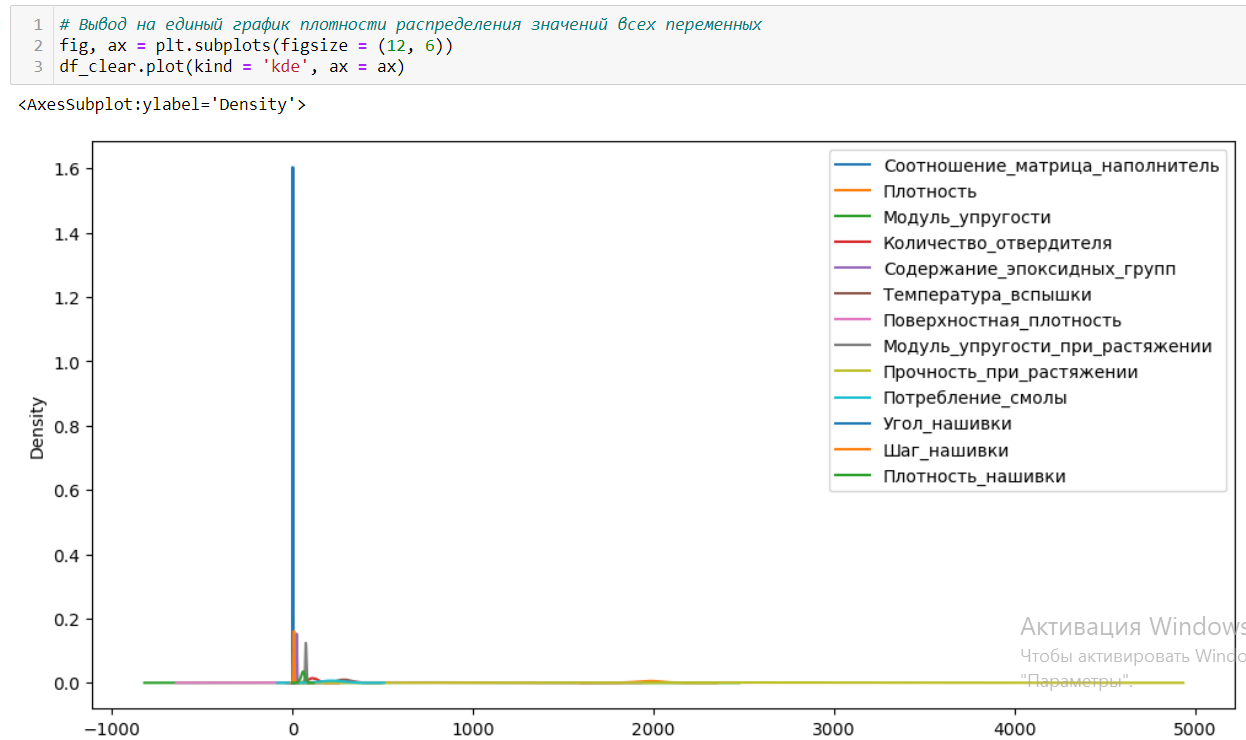


Рисунок 15 – график плотности распределения переменных

Вывод: поскольку данные находятся в разных диапазонах, не сопоставимых друг с другом. Поэтому данные необходимо масштабировать для приведения их в сопоставимый вид.

Нормализация данных с помощью MinMaxScaler:

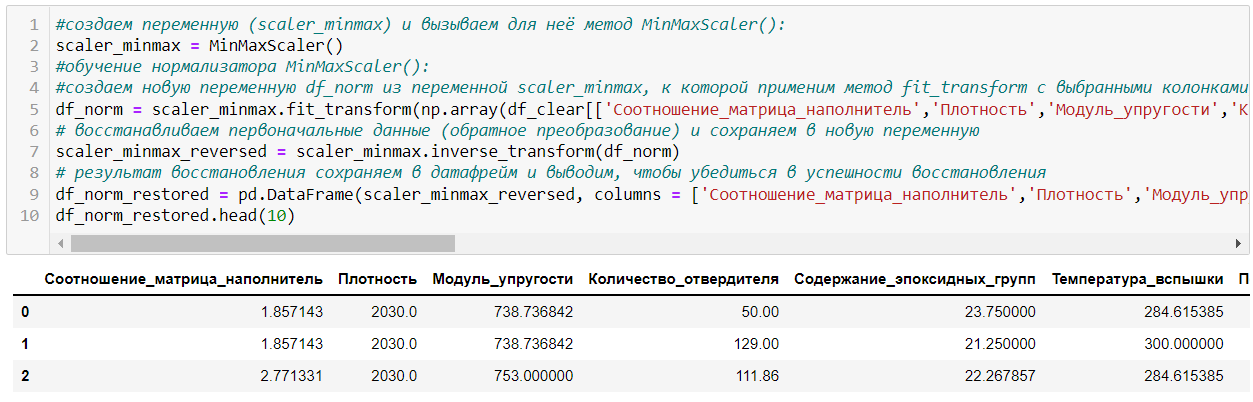
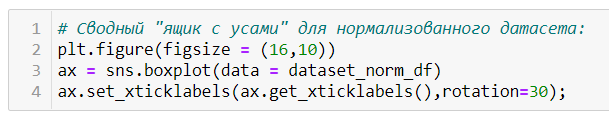


Рисунок 16 – применение метода MinMaxScaler

Вызываем описательную статистику для нормализованного датасета и по результатам имеем следующие выводы: после нормализации все средние значения входят в диапазон [0,1], что говорит о сопоставимости признаков в целях анализа их значений.

Выводим на единый график «ящики с усами» всех переменных для оценки распределения их значений:



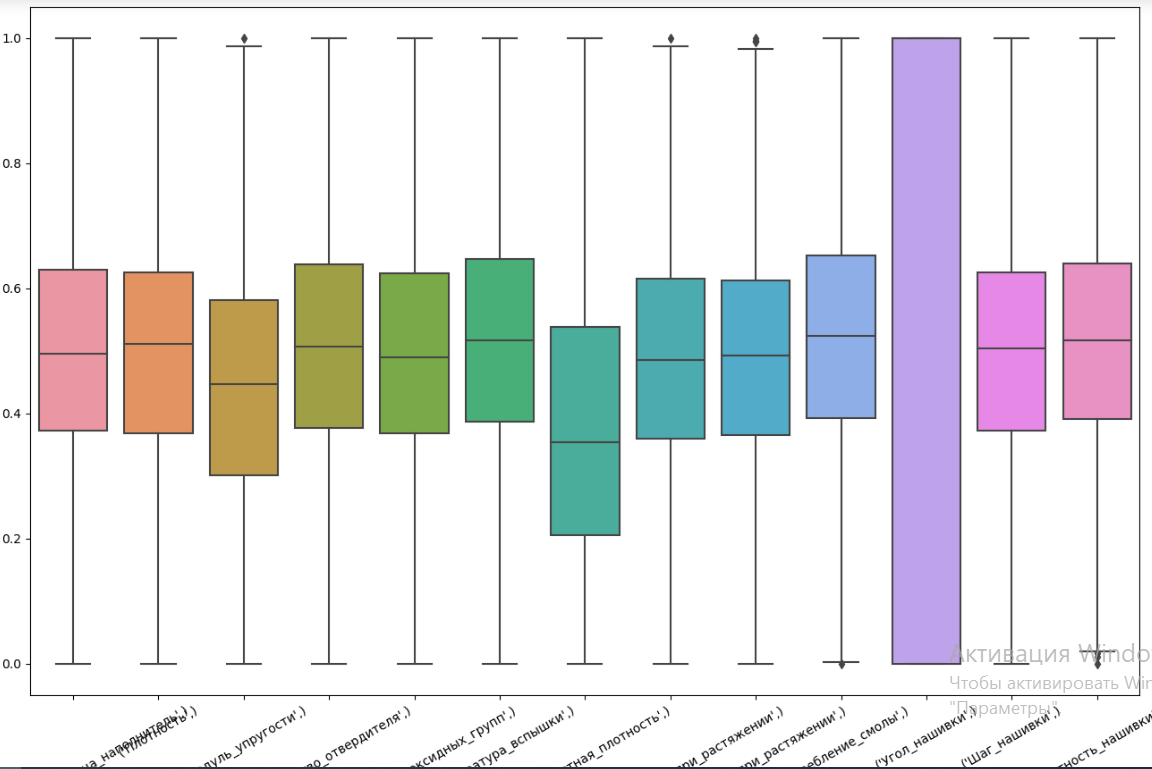


Рисунок 17 – «Ящик с усами» для всех переменных

Выводы: наблюдаем все данные строго в пределах интервала [0,1], медианы имеют разные значения.

Поскольку мы применили метод, предназначенный для нормализации нормально распределенных величин, для величины с биномиальным распределением (Угол нашивки), следует посмотреть, не изменилось ли распределение этой переменной в результате нормализации. По результатам имеем следующие выводы: характер распределения значений переменной не изменился, поэтому дополнительных действий для этой переменной не производим. Предполагаем, что метод нормализации MinMaxScaler не меняет характер распределения величин. Кроме того, диапазон значений также не изменился.

Стандартизация данных с помощью StandardScaler: стандартизацию проводим аналогичным образом.

По результатам проведенной стандартизации имеем следующие выводы:

* 1. данные сопоставимы и отцентрированы по среднему значению 0 со стандартным отклонением 1, чтобы столбцы объектов имели те же параметры, что и стандартное нормальное распределение. Можно посмотреть, как выглядит распределение переменных стандартизованного датасета на графике "ящик с усами".
  2. На графике «Ящик с усами» наблюдаем IQR строго в пределах интервала [-1,1], медианы стремятся к нулевому значению.

Выводы в целом по масштабированию: данные преобразованы и приведены к одному порядку. Поэтому применяемые методы подходит в целях предпроцессинга данных и подготовки данных для алгоритмов машинного обучения.

Сохраняем эти два датасета во внешние файлы для дальнейшего использования.

* 1. **Разработка и обучение модели**

Разработка и обучение моделей машинного обучения осуществлялась для прогнозирования значений двух (целевых) переменных: «Прочность при растяжении» и «Модуль упругости при растяжении». Для каждой из переменных обучение проведено со стандартным набором гиперпараметров алгоритмов обучения, после чего произведен поиск оптимальных параметров и выбор наилучшей комбинации гиперпараметров модели.

Поиск наилучших гиперпараметров произведен методом поиска по сетке с перекрестной проверкой, количество блоков равно 10.

Гиперпараметры модели - это такие параметры, которые задаются на входе работы алгоритма и определяют вариативность работы алгоритма. Метод поиска по сетке рассматривает некоторые комбинации гиперпараметров и выбирает ту, которая дает более низкий балл ошибки. Этот метод особенно полезен, когда есть только некоторые гиперпараметры для оптимизации. Однако он уступает другим методам взвешенно-случайного поиска, когда модель машинного обучения становится более сложной. По мнению автора работы, ни одна из моделей не является сложной, поэтому для поиска наилучших гиперпараметров выбран метод поиска по сетке. В работе применение метода поиска по сетке реализовано посредством модуля GridSearchCV из библиотеки Scikit Learn.

Этапы разработки и обучения:

* 1. Определение входных и целевых данных:

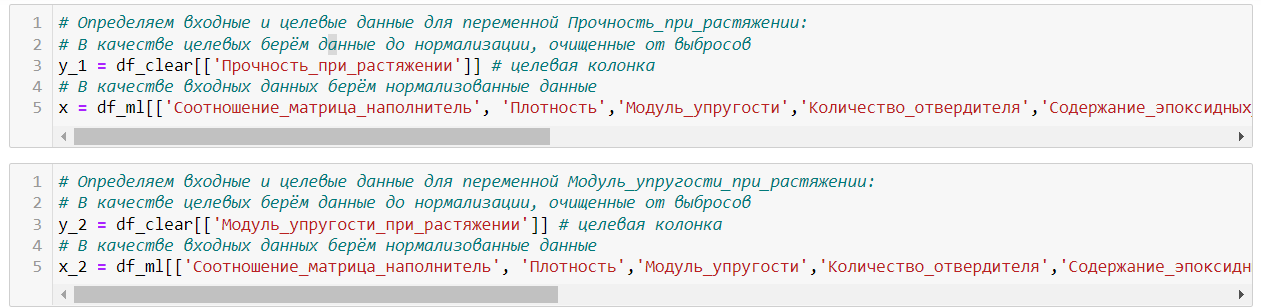


Рисунок 18 – Набор данных для обучения и тестирования

* 1. Разделение данных на обучающие и тестовые – 30% использованы для тестирования, 70% использованы для обучения:

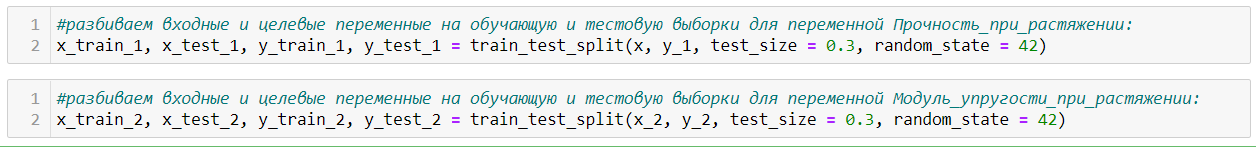
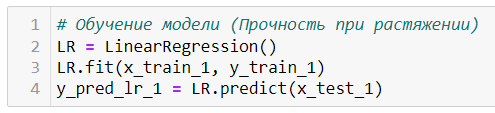


Рисунок 18 – Разделение данных на обучающие и тестовые

* 1. Обучение моделей:
* Линейная регрессия (LinearRegression):



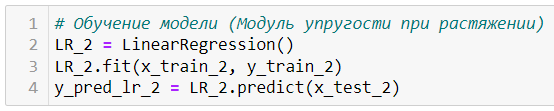


Рисунок 19 – Обучение модели LinearRegression

* Случайный лес (Random Forest):

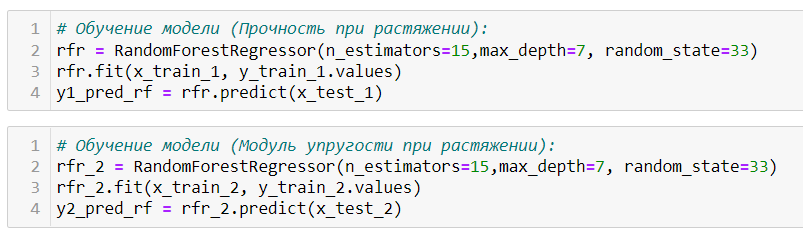


Рисунок 20 – Обучение модели RandomForestRegressor

* К-ближайших соседей (KNeighborsRegressor):

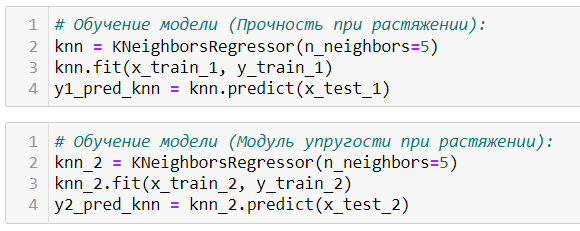


Рисунок 20 – Обучение модели KNeighborsRegressor

* Дерево решений ():

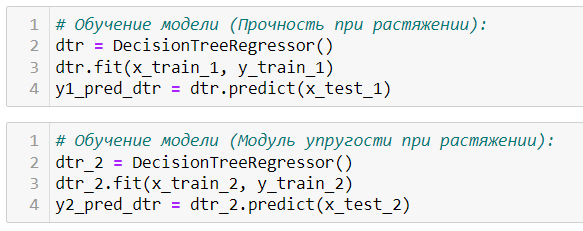


Рисунок 21 – Обучение модели DecisionTreeRegressor

* Метод опорных векторов (SVR):

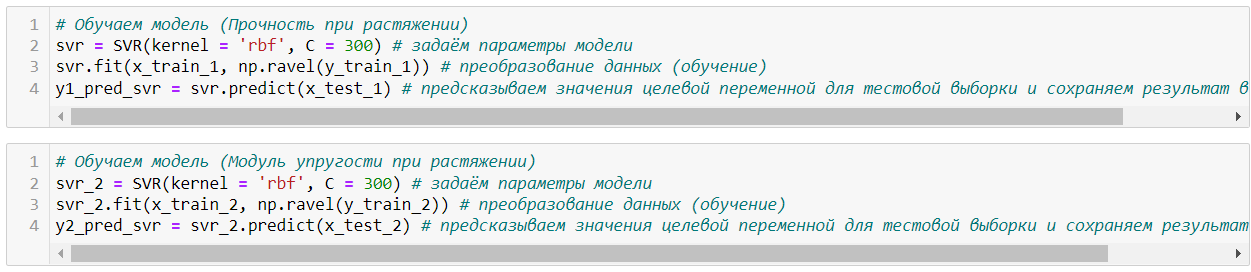


Рисунок 22 – Обучение модели SVR

* Градиентный бустинг (GradientBoostingRegressor):

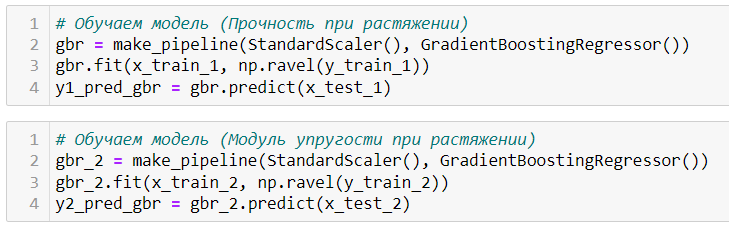


Рисунок 23 – Обучение модели GradientBoostingRegressor

* 1. Выбор наилучших гиперпараметров моделей (на примере модели RandomForestRegressor):

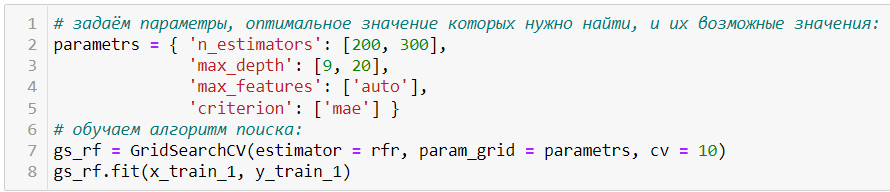


Рисунок 24 – Обучение алгоритма поиска

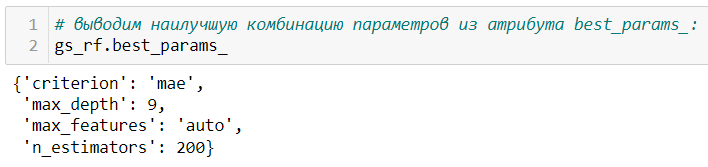


Рисунок 25 – Вывод наилучшей комбинации гиперпараметров

* 1. Подстановка результатов поиска в модель (переобучение модели с новыми гиперпараметрами):

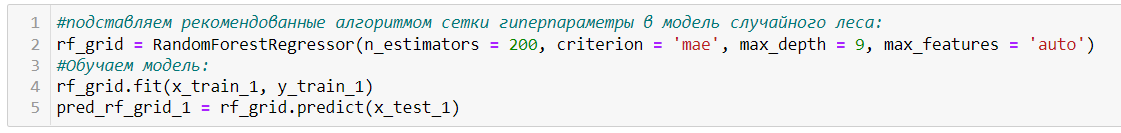


Рисунок 26 – Переобучение модели с новыми параметрами

* 1. Сравнение точности модели со стандартными и оптимальными гиперпараметрами (на примере модели RandomForestRegressor для целевой переменной «Прочность при растяжении»):

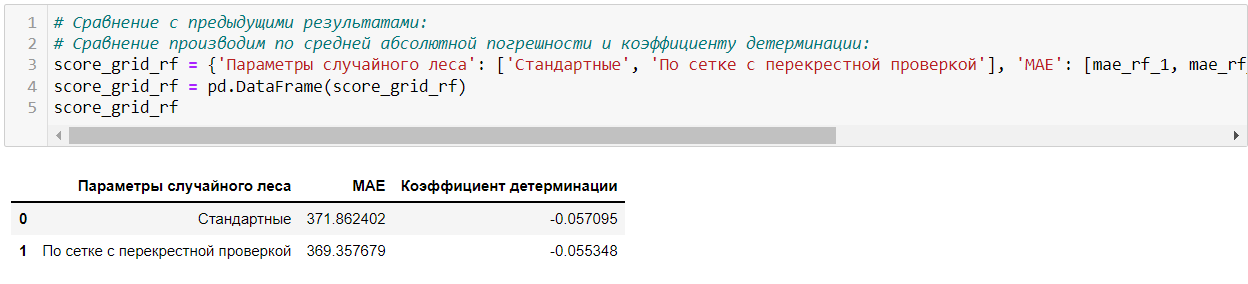


Рисунок 27 – Сравнение точности модели до и после оптимизации

* 1. **Тестирование модели**

Оценка моделей произведена на основании следующих метрик потерей модели (пример использования метрик в работе – см Рисунок 28):

* Средняя абсолютная ошибка модели (MAE) – показывает среднюю ошибку модели и используется как для оценки производительности модели, так и для сравнительной оценки моделей;
* Средняя квадратическая ошибка модели (MSE) - показывает средний квадрат ошибки модели и используется для оценки производительности модели в качестве уточняющей метрики;
* Коэфициент детерминации модели (R2) – используется как для оценки качества подбора уравнения регрессии, так и для сравнительной оценки моделей.

Определение метрик произведено для каждой модели и каждой целевой переменной (ниже на Рисунке 28 – на примере переменной «Прочность при растяжении» и модели LinearRegression):

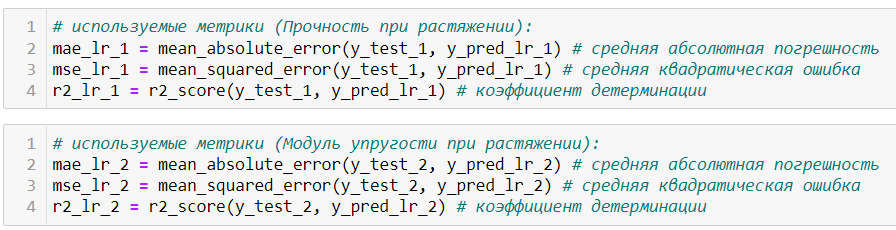
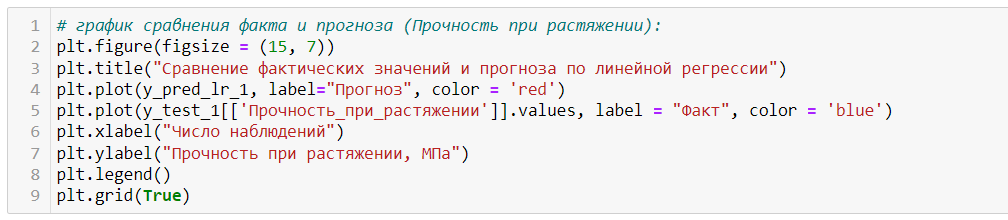


Рисунок 28 – Определение метрик для оценки модели

Также для визуальной оценки точности моделей для каждой модели и целевой переменной построены графики сравнения прогнозных значений с фактическими (на примере модели LinearRegression и переменной «Прочность при растяжении):



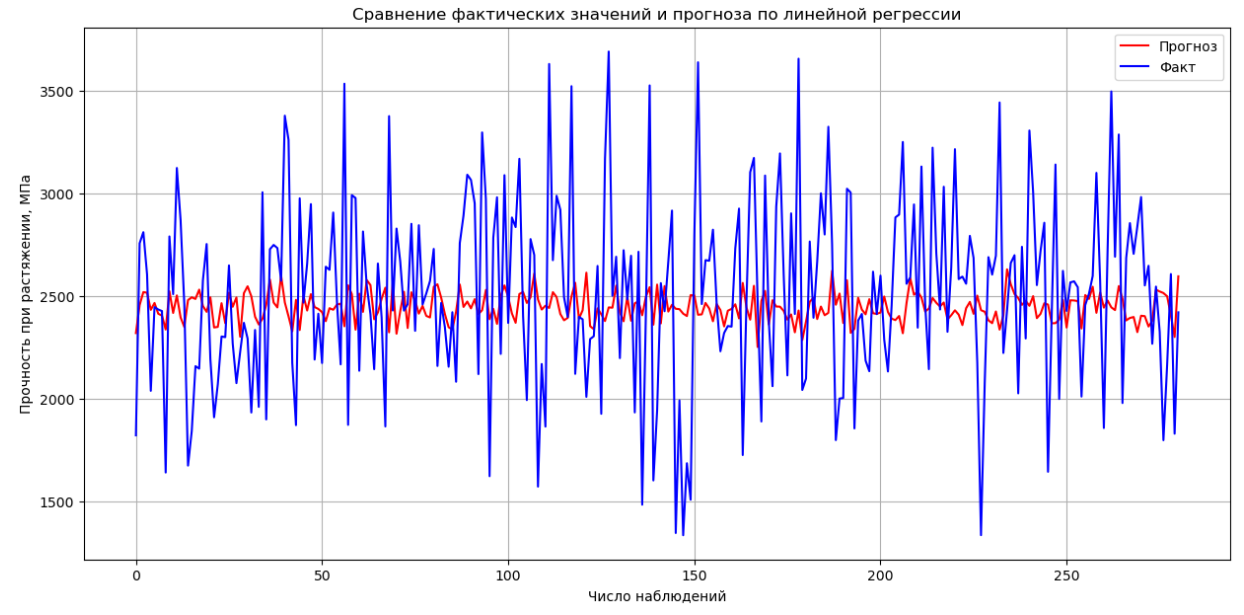


Рисунок 29 – Визуализация результатов прогноза в сравнении с фактом

Сравнительные результаты оценок всех моделей для каждой целевой переменной представлены в табличном виде (на примере целевой переменной «Прочность при растяжении»):

* Для моделей со стандартными гиперпараметрами:

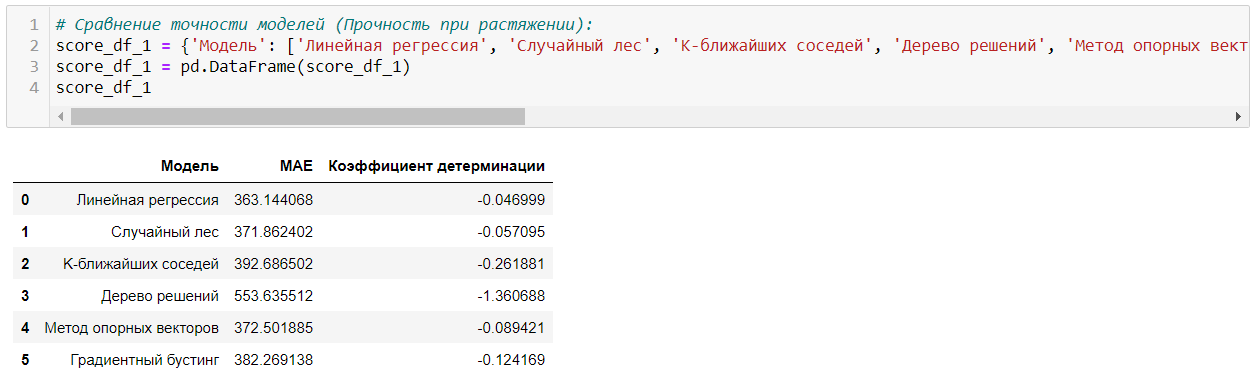
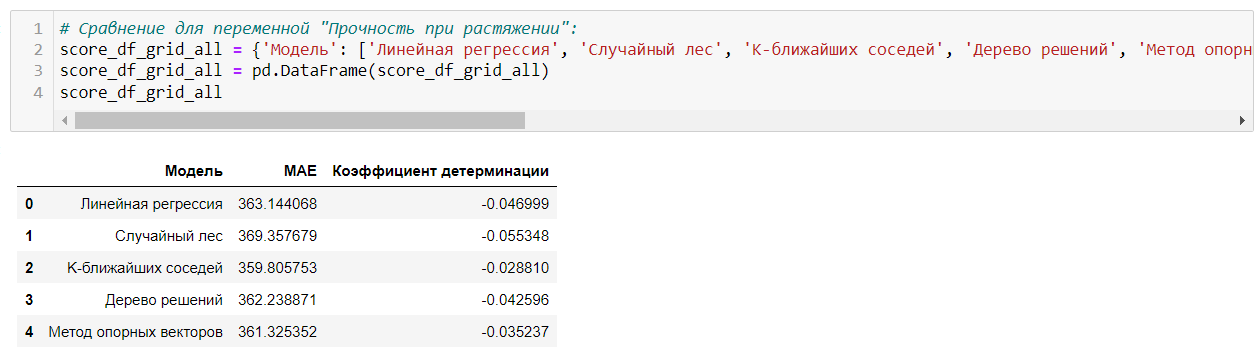


Рисунок 30 – Сравнение результатов оценок точности моделей (стандартные гиперпараметры)

* Для моделей с оптимальными гиперпараметрами:



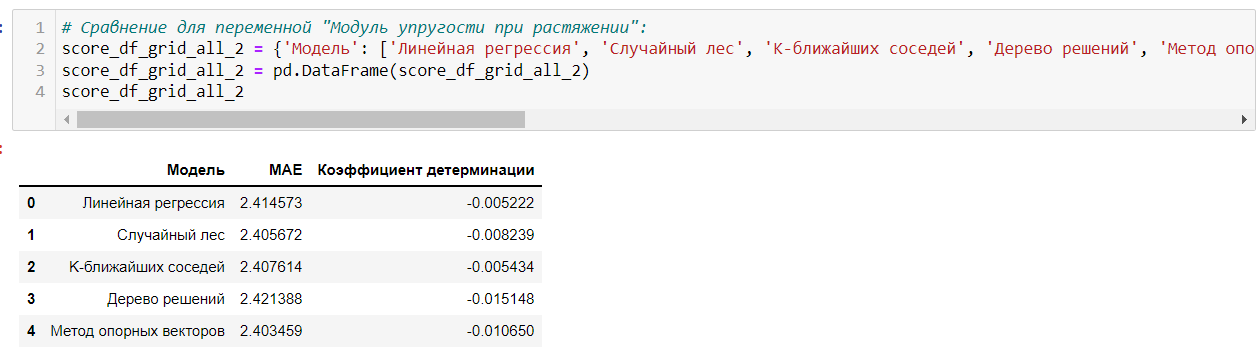


Рисунок 31 – Сравнение результатов оценок точности моделей (оптимальные гиперпараметры)

На основании данных сравнительных таблиц можно сделать следующие выводы:

* + 1. Все модели имеют высокие значения MAE (как до, так и после оптимизации гиперпараметров), что говорит об их низкой точности;
    2. Все модели имеют отрицательный коэффициент детерминации (как до, так и после оптимизации гиперпараметров), что говорит о том, что модель даёт неоднозначные результаты и предсказывает хуже, чем результаты модели, выдающей среднее значение;
    3. Подбор оптимальных гиперпараметров методом сетки с перекрестной проверкой несколько снизил потери, но в целом результата не дал;
    4. Наилучший результат для прогнозирования целевой переменной «Прочность при растяжении» до оптимизации гиперпараметров показала модель «Линейная регрессия», а после оптимизации – «К-ближайших соседей». Но поскольку все модели имеют схожие значения оценок и, кроме того, все оценки показывают на их низкую точность и низкую детерминированность, то нет уверенности, что текущая лучшая модель сохранит своё первенство при повторных итерациях.

Возможные причины таких результатов и меры по их улучшению:

* Наличие ошибок или неполнота исходных данных для моделирования. В этом случае необходимо понять, в какой момент возникли эти ошибки – присутствовали в исходных файлах либо появились в процессе подготовки и обработки данных для передачи в модель;
* Неудачная модель регрессии – логика или параметры. В этом случае необходимо либо использовать другую модель, либо экспериментировать с набором параметров;
* Неподходящие методы нормализации: возможно, стоит попробовать, например, Normalizer;
* Неподходящий набор входных гиперпараметров для поиска наилучших либо неподходящий алгоритм поиска.
  1. **Создание нейронной сети для прогнозирования**

**соотношения «матрица – наполнитель».**

Прогнозирование целевой переменной «Соотношение ‘матрица – наполнитель’» в данной работе произведено посредством искусственной нейронной сети.

Характерным отличием нейросетей от других вычислительных моделей является их ориентация на биологические принципы, благодаря чему они обладают следующими качествами:

* массовый параллелизм;
* распределённое представление информации и вычисления;
* способность к обучению и обобщению;
* адаптивность;
* свойство контекстуальной обработки информации;
* толерантность к ошибкам;
* низкое энергопотребление.

Правила работы нейросетей не программируются, а вырабатываются в процессе обучения, что обеспечивает адаптивность этой модели к изменениям входных сигналов и шуму.

Решение задачи выполнено посредством последовательной модели Sequential из пакета Keras библиотеки глубокого обучения Tensorflow.   
Создание нейронной сети произведено по следующим этапам:

1. Подготовка данных для обучения:

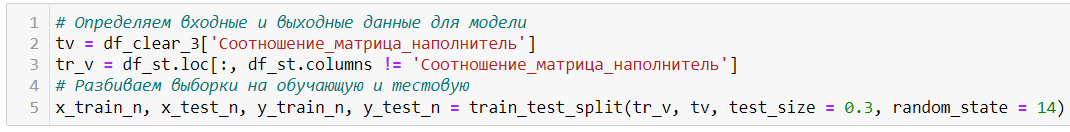


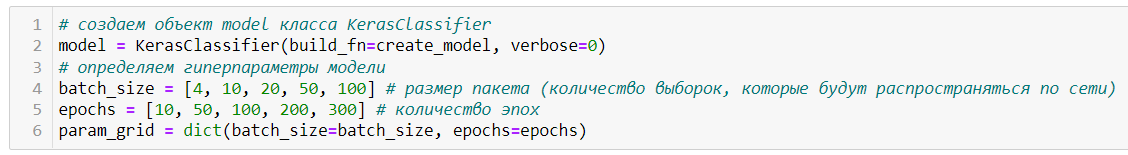
Рисунок 32 – Подготовка данных для обучения нейросети

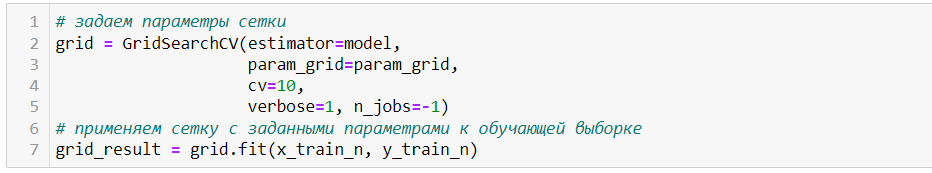
1. Задание функции модели. Используем технику выпадения - технику, при которой случайно выбранные нейроны игнорируются во время тренировки. Это означает, что их вклад в активацию нижестоящих нейронов временно удален на прямом проходе, и любые обновления веса не применяются к нейрону на обратном проходе:

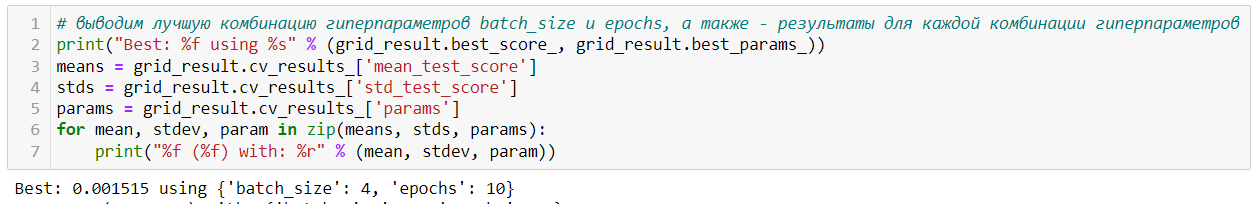


Рисунок 33 – Определение функции модели

1. Настройка оценщика модели и поиск оптимальных параметров. Для создания оценщика модели применен метод KerasClassifier. Поиск оптимальных параметров производим с помощью поиска по сетке GridSearch с количеством блоков = 10. Поскольку поиск по сетке выполняется достаточно долго и есть риск аварийного прерывания кода в случае перегрузки из-за большого числа комбинаций, будем производить поиск по сетке последовательно, для каждого одного-двух гиперпараметров отдельно, подставляя найденные значения в следующий поиск (на Рисунке 34 – часть поиска для демонстрации алгоритма):







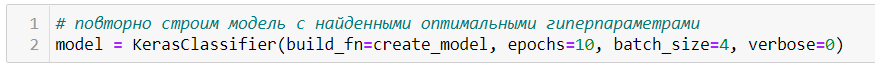


Рисунок 34 – Поиск оптимальных гиперпараметров

После последовательного поиска наилучших гиперпараметров имеем следующие результаты: Наилучшей комбинацией значений гиперпараметров является: «размер пакета» = 4; «число эпох» = 10; «оптимизатор» = Adadelta; «слои» = ([32, 8, 3]); «функция активации» = sigmoid; «коэффициент отсева» = 0,0.

1. Выполняем построение модели с подобранными оптимальными гиперпараметрами:



Рисунок 35 – Построение окончательной модели

1. Обучение нейросети:

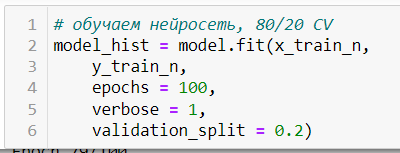
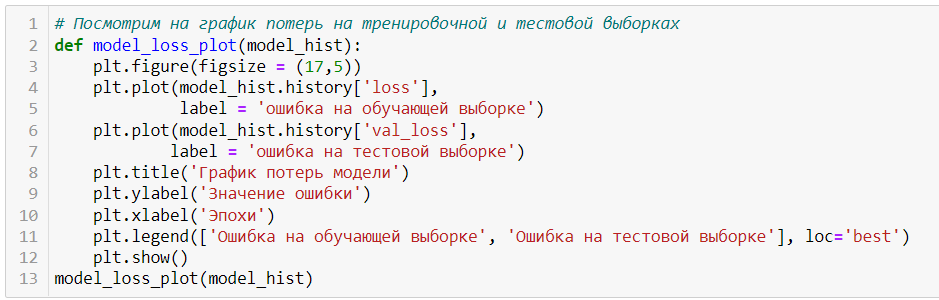


Рисунок 36 – Обучение нейросети

1. Составление графика потерь:



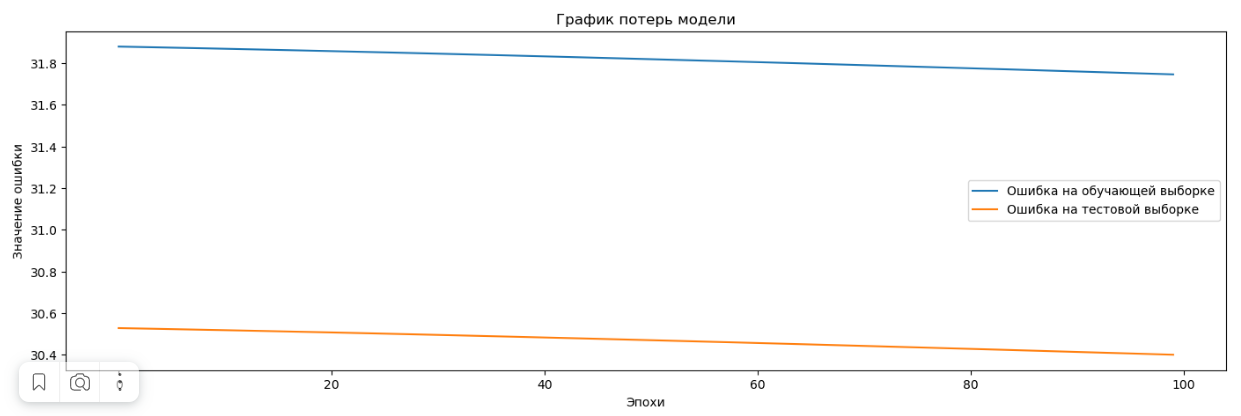
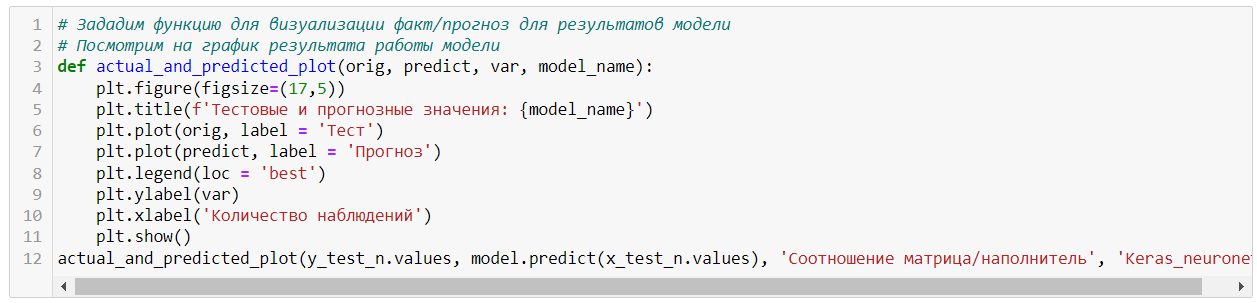


Рисунок 37 – График потерь модели

Выводы на основании графика: в случае числа эпох, равного 10, потери не снижаются ни на тестовых данных, ни на обучающих. В случае числа эпох, равного 100 (как на Рисунке 37), потери модели снижаются незначительно как на тестовых, так и на обучающих данных.

1. Построение графиков для сравнения результатов прогноза нейросетью с фактическими данными:



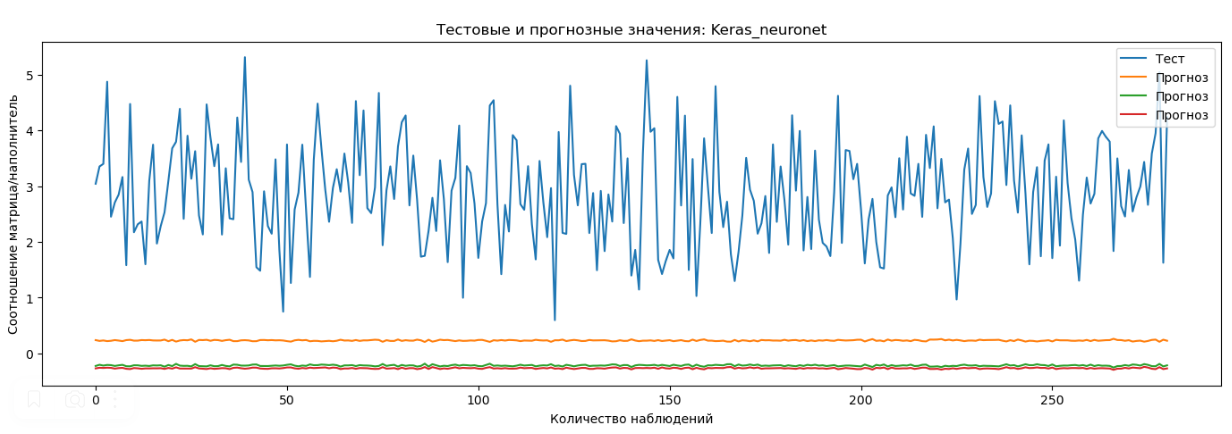


Рисунок 38 – Графики тестовых и прогнозных значений

Выводы по результатам оценки созданной нейросети: модель нейросети имеет абсолютно нулевую точность (т.е. не имеет никаких совпадений с фактическими значениями).

Возможные причины таких результатов и меры по их улучшению:

* Наличие ошибок или неполнота исходных данных для моделирования. В этом случае необходимо понять, в какой момент возникли эти ошибки – присутствовали в исходных файлах либо появились в процессе подготовки и обработки данных для передачи в модель;
* Неудачная функция модели – логика или параметры. В этом случае необходимо создать новую функцию модели;
* Неподходящий оценщик: возможно, стоит попробовать, например, KerasRegression;
* Неподходящий набор входных гиперпараметров для поиска наилучших.

**Разработка приложения**

Приложение для созданной модели прогнозирования свойства «Прочность при растяжении» композитных материалов создано в оболочке Jupiter Notebook.

Приложение достаточно примитивно и не является конечным для предложения его пользователю, поэтому предполагается в дальнейшем дополнять его таким образом, чтобы сделать его более удобным и привлекательным для пользователей.

Суть приложения заключается в следующем:

1. Предварительно загружается сохраненная ранее модель:

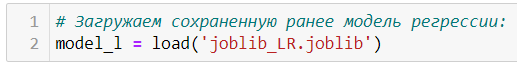
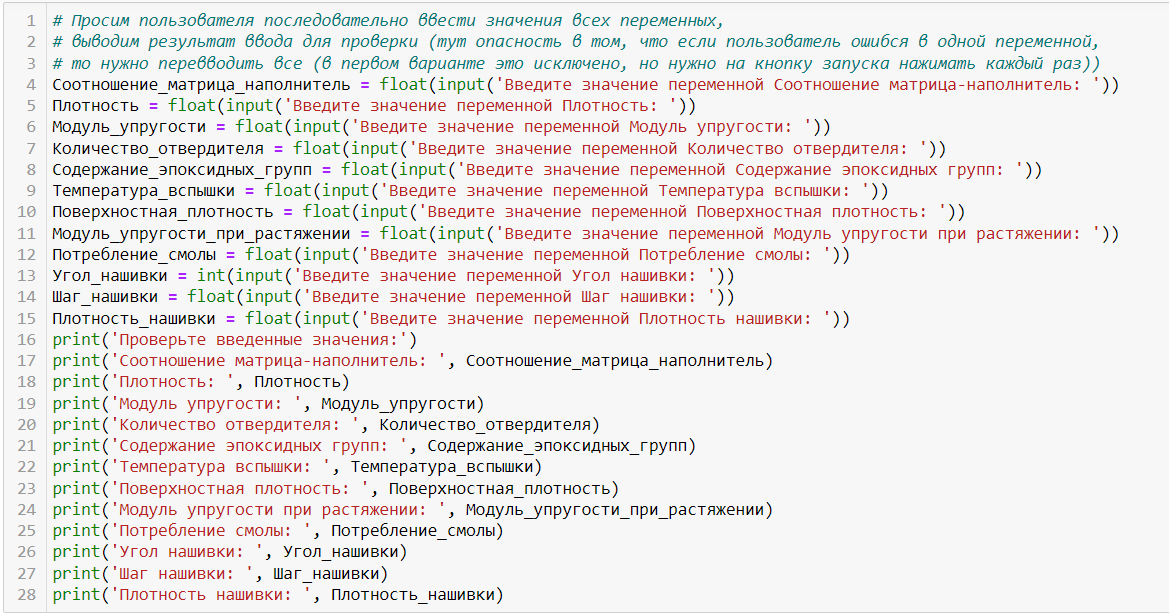


Рисунок 39 – Загрузка сохраненной ранее модели

1. Пользователь вводит последовательно значения входных переменных, после чего система предлагает ему проверить введенные значения:





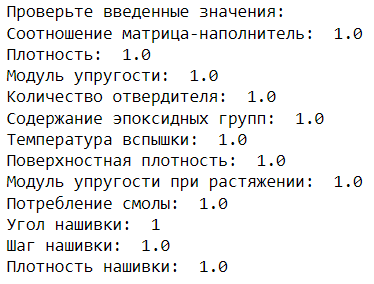
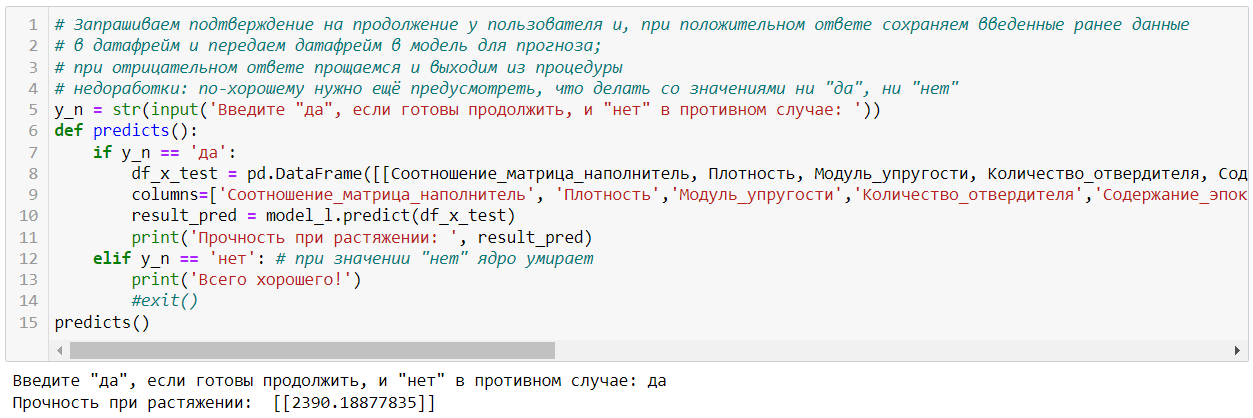


Рисунок 40 – Ввод и проверка введенных значений

1. Далее, у пользователя запрашивается подтверждение или отказ от продолжения работы и, в зависимости от введенного пользователем значения (да/нет), запускается один из вариантов поведения системы: если «да» – то прогноз значения целевой переменной производится и результат выводится на экран, если «нет» – на экран выводится сообщение «Всего хорошего!» и никаких других действий не производится:



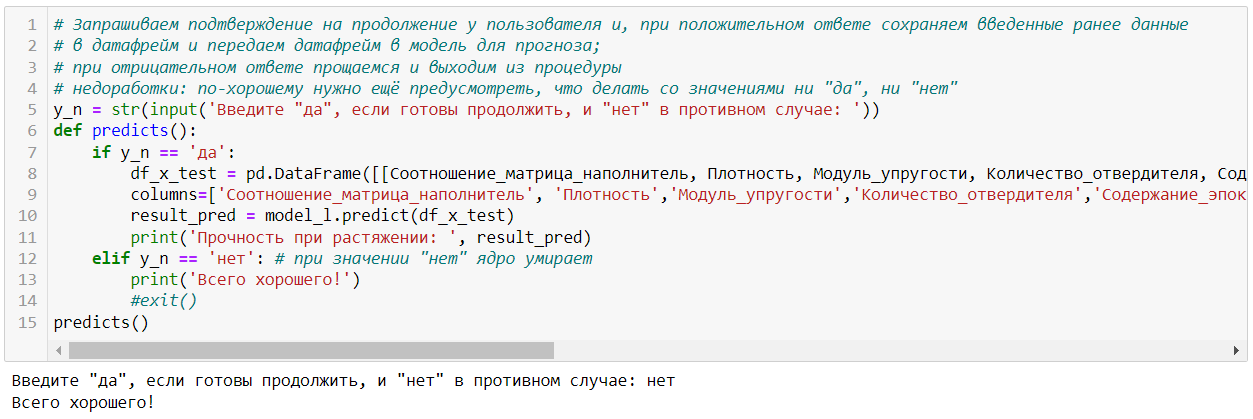


Рисунок 41 – Выполнение прогноза

* 1. **Создание удалённого репозитория**

Для предоставления общего доступа к ВКР на веб-сервисе github.com:

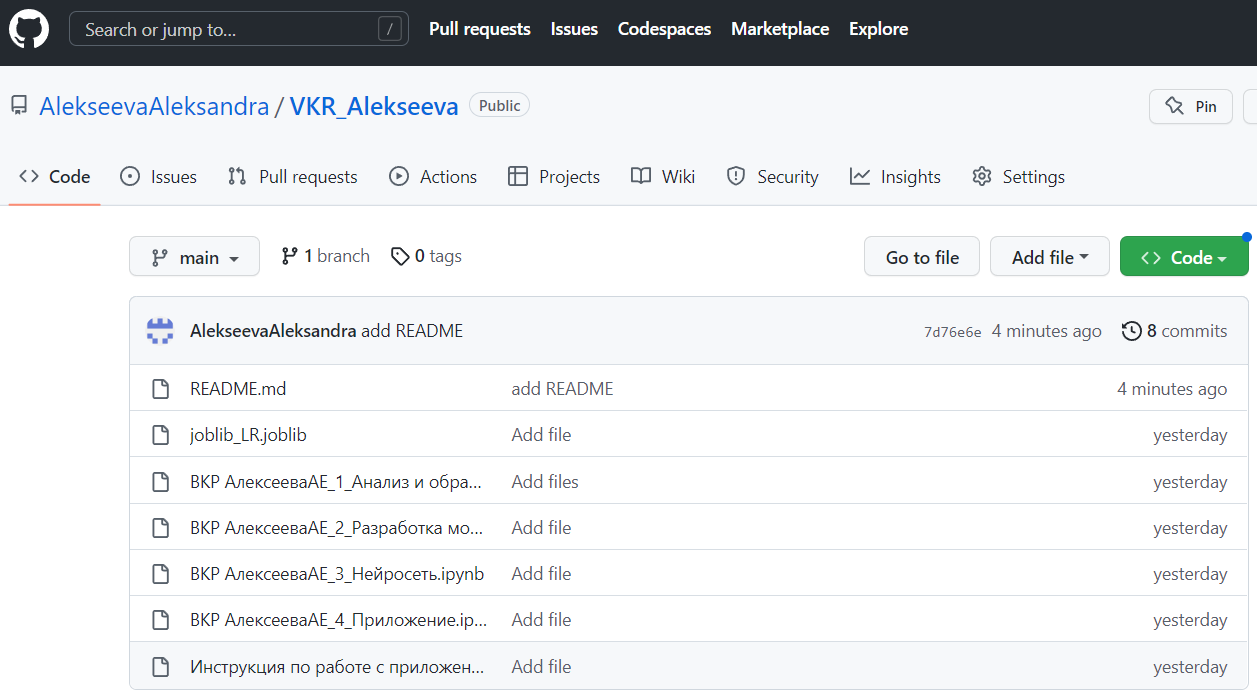


Рисунок 42 – репозиторий с данными ВКР

Репозиторий доступен по ссылке:

<https://github.com/AlekseevaAleksandra/VKR_Alekseeva/tree/main>

* 1. **Заключение**

Созданные модели прогнозирования – всего лишь малая часть из большого множества вариантов решения задачи прогнозирования, причем самые простые варианты. В данной работе на практике были применены самые распространенные методы и способы решения задачи. Несмотря на то, что разработанные варианты решений не эффективны, их всё же следует учитывать при последующих итерациях решения задач прогнозирования с тем, чтобы не допускать их повторного применения, экономя тем самым время.

Для того, чтобы решить поставленную задачу, необходимо разработать некий алгоритм исследования, который позволит за минимально число шагов найти оптимальную модель регрессии, а именно: разработка ветвей алгоритма (например, если в определенный момент имеем неудовлетворительные результаты, то не идём до конца, а переходим к другой ветке (новая модель или новый вариант текущей модели), начальных значений (параметров модели), условий перехода от одной ветви к другой (точка и критерий перехода) и т.д.

Для составления такого алгоритма потребуется совместная работа группы специалистов (не только специалистов DataScience, но и, например, специалистов по производству композитов, конечные свойства которых прогнозируются; возможно, специалистов по статистике для усиления математического аппарата). В случае набора команды из специалистов разных областей связующим звеном для совместной работы всех членов команды должен стать специалист DataScience.

Возможно, для решения задачи не достаточно только лишь того арсенала средств, которые предоставляет статистика и машинное обучение, и имеет смысл комбинировать эти методы с принципиально другими подходами: разбить задачу на классы по методам обработки и снова объединять после обработки, пересмотреть сбор первичных данных (число входящих признаков), на каких-то этапах проводить физический эксперимент (лабораторные исследования), где-то использовать теоретические основы физики и т.п.

* 1. **Приложения**

Приложение 1: Список используемой литературы и веб ресурсы

* + - 1. Елена Капаца, статья MinMaxScaler, апрель 2022г: <https://www.helenkapatsa.ru/minmaxscaler/>
      2. 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>.
      3. Andre Ye. 5 алгоритмов регрессии в машинном обучении, о которых вам сле-дует знать: – Режим доступа: <https://habr.com/ru/company/vk/blog/513842>
      4. Документация по библиотеке keras: <https://keras.io/api/>
      5. Документация по библиотеке matplotlib: [https://matplotlib.org/stable/users/index.html](https://matplotlib.org/stable/users/index.html%20)
      6. Документация по библиотеке numpy:  <https://numpy.org/doc/1.22/user/index.html#user>
      7. Документация по библиотеке pandas: <https://pandas.pydata.org/docs/user_guide/index.html#user-guide>
      8. Документация по библиотеке scikit-learn: <https://scikit-learn.org/stable/user_guide.html>
      9. Документация по библиотеке seaborn: <https://seaborn.pydata.org/tutorial.html>
      10. Документация по библиотеке Tensorflow: <https://www.tensorflow.org/overview>
      11. Документация по языку программирования python:  <https://docs.python.org/3.8/index.html>
      12. Краткий обзор алгоритма машинного обучения Метод Опорных Векторов (SVM): <https://habr.com/ru/post/428503/>
      13. Материалы с сайта Wikipedia.org: <https://ru.wikipedia.org/wiki/Заглавная_страница>

Приложение 2: План работы

1. Загрузка и обработка исходных данных
   1. Загрузка данных
   2. Объединение исходных данных в один датасет
   3. Удаление лишних (неинформативных) столбцов
   4. Анализ данных по типам значений и обработка значений при необходимости
   5. Сохранение результатов обработки исходных данных как рабочего датасета в файл
2. Разведочный анализ данных
   1. Проверка рабочего датасета на дубликаты и пропуски
   2. Получение среднего значения и медианы для каждой переменной, анализ описательной статистики
   3. Анализ распределения значений переменнных
   4. Анализ статистической зависимости переменных (матрица диаграмм рассеяния, коэфициенты корреляции)
3. Предобработка данных
   1. Анализ аномалий и выбросов в рабочем датасете (метод "Ящик с усами")
   2. Цикл "исключение выбросов - анализ очищенных от выбросов данных - проверка на наличие остатков выбросов - повторное исключение выбросов"
   3. Проведение разведочного анализа на очищенных данных
   4. Сохранение очищенного от выбросов датасета в файл
4. Нормализация и стандартизация данных очищенного датасета
   1. Оценка плотности ядра (KDE)
   2. Нормализация данных с помощью MinMaxScaler. Оценка результатов нормализации
   3. Стандартизация с помощью StandardScaler. Оценка результатов стандартизации
   4. Сохранение отмасштабированных данных в файл.
5. Разработка и обучение прогнозных моделей
   1. Подготовка данных для обучения для каждой переменной
   2. Определение входных и выходных данных
   3. Разделение данных на обучающую и тестовую выборки
   4. Проверка обучающей и тестовой выборок на случайность
   5. Обучение прогнозных моделей посредством выбранных алгоритмов и оценка работы алгоритмов для каждой переменной:
      1. Алгоритм "Линейная регрессия"
      2. Алгоритм "Случайный лес"
      3. Алгоритм "К-ближайших соседей"
      4. Алгоритм "Дерево решений"
      5. Алгоритм "Метод опорных векторов"
      6. Алгоритм "Градиентный бустинг"
   6. Сравнение работы алгоритмов
6. Поиск и оценка оптимальных гиперпараметров и модели:
   1. Поиск и оценка оптимальных гиперпараметров моделей для каждой переменной
   2. Поиск модели с наилучшими гиперпараметрами
   3. Сравнение работы алгоритмов после оптимизации гиперпараметров. Выбор лучшей модели
7. Разработка нейронной сети для рекомендации соотношения матрица-наполнитель
   1. Разделение данных на обучающие и тестовые
   2. Описание функции модели
   3. Поиск наилучшей архитектуры и оптимальных гиперпараметров
   4. Применение оптимальных гиперпараметров для построения окончательной модели и обучение модели
   5. Оценка потерь модели и точности модели на тестовых и обучающих данных.
8. Разработка приложения, составление инструкции к приложению

Приложение 3: Инструкция к приложению для модели прогнозирования конечного свойства «Прочность при растяжении» композитных материалов

ВНИМАНИЕ! *Для использования приложения необходим язык Python, а также – среда разработки JupiterNotebook или Colab, в котрой необходимо открыть файл .* *ipynb, содержащий приложение. Также в этой среде должен быть размещен файл .joblib, содержащий модель регрессии.*

В файле приложения работа сводится к следующему:

1. Импорт библиотек (запуск первой ячейки):



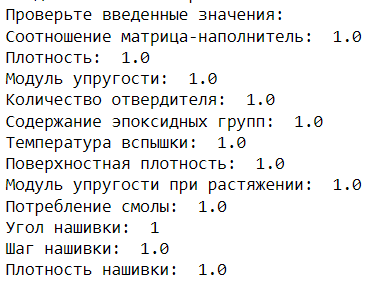
1. Загрузка модели из файла, размещенного в среде разработки (запуск второй ячейки):



1. Последовательный ввод входных данных для прогнозирования (значения входных свойств композитов) (запуск третьей ячейки):



После нажатия кнопки запуска необходимо ввести с клавиатуры значение каждого свойства. Ввод данных каждого свойства завершается нажатием кнопки Enter на клавиатуре, после завершения ввода последнего свойства производится завершение исполнения кода в ячейке и на экран выводится результат ввода:



Введенные данные необходимо проверить и перейти к запуску следующей ячейки.

ВНИМАНИЕ! *Текущая версия приложения не предполагает корректировку введенных значений после завершения ввода по отдельным свойствам, т.е. корректируются значения всех свойств повторным вводом данных последовательно по каждому свойству.*

1. Прогнозирование и получение результатов прогноза (запуск четвёртой ячейки). Перед началом прогнозирования необходимо в поле ввода вести одно из двух возможных значений – «да» или «нет»:

* «да» вводится в случае, если необходимо произвести прогнозирование;
* «нет» вводится в случае, если по каким-то причинам нужно отказаться от прогнозирования.

ВНИМАНИЕ! *Текстовые значения «да» и «нет» вводятся строго без кавычек и только строчными буквами:*



По завершении ввода (нажатием кнопки Enter после ввода текста) в зависимости от введенного текста на экран выводится результат:

* Если прогнозирование произведено, то – результат прогноза:



* Если прогнозирование отменено, то текст:

