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 Васягин Игорь Евгеньевич

Москва, 2023

Оглавление

Введение

Композиционный материал или композитный материал - 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/матрицы и наполнитель/наполнители, последние 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.

1. Аналитическая часть

1.1. Постановка задачи

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).

Дано: датасет из 2-х файлов X_bp.xlsx X_nup.xlsx со свойствами композитов.

Требуется:

1) Объединение делать по индексу тип объединения INNER.
2) Провести разведочный анализ предложенных данных.
3) Провести предобработку данных (удаление шумов, нормализация и т.д.).
4) Обучить нескольких моделей для прогноза:
   А) Модуля упругости при растяжении.
   Б) Прочности при растяжении.

   При построении модели необходимо 30% данных оставить на тестирование модели, на остальных произвести обучение моделей. При построении моделей провести поиск гиперпараметров модели с помощью поиска по сетке с перекрестной проверкой, количество блоков равно 10.
5) Написать нейронную сеть, которая будет рекомендовать соотношение:
   А) Матрица-наполнитель.
6) Разработать приложение с графическим интерфейсом или интерфейсом командной строки, которое будет выдавать прогноз, полученный в задании 4 или 5 (один или два прогноза, на выбор учащегося).
7) Оценить точность модели на тренировочном и тестовом датасете.
8) Создать репозиторий в GitHub / GitLab и разместить там код исследования.

1.2. Входящие данные

Датасет состоит из 13 признаков, 1023 строки.

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

Рисунок 1 – Датасет

Исходя из поставленной задачи, распределим признаки на входные и выходные значения. Другими словами получаем модель для обучения с учителем.

Модель	Входные параметры	Выходные параметры, Прогноз
1	Соотношение матрица-наполнитель Плотность, кг/м3 Модуль упругости, ГПа Количество отвердителя, м.% Содержание эпоксидных групп,%_2 Температура вспышки, С_2 Поверхностная плотность, г/м2 Потребление смолы, г/м2 Угол нашивки, град Шаг нашивки Плотность нашивки	Модуль упругости при растяжении, Гпа Прочность при растяжении, МПа
2	Плотность, кг/м3 Модуль упругости, ГПа Количество отвердителя, м.% Содержание эпоксидных групп,%_2 Температура вспышки, С_2 Поверхностная плотность, г/м2 Модуль упругости при растяжении, ГПа Прочность при растяжении, МПа	Соотношение матрица-наполнитель

	Потребление смолы, г/м2 Угол нашивки, град Шаг нашивки Плотность нашивки	

Таблица 1 – Признаки модели

1.3. Описание используемых методов

Данная задача в рамках классификации категорий машинного обучения относится к машинному обучению с учителем и традиционно это задача регрессии.

Цель любого алгоритма обучения с учителем — определить функцию потерь и минимизировать её, соответственно, для получения наилучшего результата были применены следующие методы.

1. Linear regression - линейная регрессия
2. Dummy - Дамми
3. LASSO - Лассо
4. Ridge - гребневая регрессия
5. Elastic Net - Эластичная регрессия
6. K-Neighbors Regressor - к-ближайших соседей
7. Decision Tree Regressor - дерево решений
8. Random Forest - случайный лес
9. Gradient Boosting Regressor - градиентный бустинг
10. SVR - метод топорных векторов
11. Нейронная сеть

1.3.1. Линейная регрессия

Линейная регрессия предполагает линейное отношение между свободной переменной и целевой переменной. При наличии нескольких входных переменных процедура называется множественной линейной регрессией. Алгоритм линейной регрессии используется, если метки непрерывны. Один из

простейших способов выполнения вычислений основан на применении так называемого обычного метода наименьших квадратов.

Плюсы алгоритма: алгоритм прост в реализации и хорошо работает вне зависимости от размера набора данных. Не требует высоких вычислительных мощностей. За счет регуляризации можно снизить риск переобучения

Минусы алгоритма: возможны ситуации недообучения, когда модель машинного обучения не может правильно собрать данные. Обычно это происходит, когда функция гипотезы не может хорошо соответствовать данным. Поскольку линейная регрессия предполагает линейную связь между входными и выходными переменными, она не может должным образом соответствовать сложным наборам данных, из-за чего модель имеет низкую точность. Регрессия чувствительна к выбросам.

1.3.2. Дамми

Фиктивная переменная качественная переменная, принимающая значения 0 и 1, включаемая в эконометрическую модель для учёта влияния качественных признаков и событий на объясняемую переменную. При этом фиктивные переменные позволяют учесть влияние не только качественных признаков, принимающих два значения, но и несколько возможных. В этом случае добавляются несколько фиктивных переменных. Фиктивная переменная может быть также индикатором принадлежности наблюдения к некоторой подвыборке. Последнее можно использовать для обнаружения структурных изменений.

1.3.3. Регрессия по методу Лассо

Операция наименьшего абсолютного сокращения и выбора. Как следует из названия, LASSO использует метод «усадки», в котором определяются коэффициенты, которые уменьшаются по направлению к центральной точке в качестве среднего значения. Процедура лассо поощряет использование простых разреженных моделей (то есть моделей с меньшим количеством параметров). Этот конкретный тип регрессии хорошо подходит для моделей,

демонстрирующих высокий уровень мультиколлинеарности, или когда нужно автоматизировать определенные части выбора модели, такие как выбор переменных/исключение параметров.

1.3.4. Гребневая регрессия

Усовершенствование Линейной регрессии с повышенной устойчивостью к ошибкам, с ограничениями на коэффициенты регрессии для получения наиболее приближенного к реальности результата.

Плюсы алгоритма: результат гораздо проще интерпретировать.

Применяется метод для борьбы с переизбыточностью данных, когда независимые переменные коррелируют друг с другом.

1.3.5. Эластичная регрессия

Включает в себя термины регуляризации как L-1, так и L-2. Это дает преимущества регрессии Лассо и Гребневой. Было установлено, что он обладает предсказательной способностью лучше, чем у Лассо, хотя все еще выполняет выбор функций. Поэтому получается лучшее из обоих методов, выполняя выбор функции Лассо с выбором группы объектов Ридж.

Лассо, Гребневая и Эластичная сеть - это модификации обычной линейной регрессии наименьших квадратов, которые используют дополнительные штрафы, чтобы сохранить значения коэффициента небольшими и упростить модель.

Лассо полезно для выбора функций, когда набор данных имеет функции с плохой предсказательной силой.

Регрессия гребня полезна для группового эффекта, при котором коллинеарные элементы могут быть выбраны вместе.

Эластичная регрессия сочетает в себе регрессию Лассо и Гребневую, что потенциально приводит к модели, которая является простой и прогнозирующей.

1.3.6. Метод k-ближайших соседей

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.

Плюсы: Алгоритм прост и легко реализуем, не чувствителен к выбросам, нет необходимости строить модель, настраивать несколько параметров или делать дополнительные допущения. Алгоритм универсален и его можно использовать для обоих типов задач: классификации и регрессии

Минусы: алгоритм работает значительно медленнее при увеличении объема выборки, предикторов или независимых переменных. Из аргумента выше следуют большие вычислительные затраты во время выполнения. Всегда нужно определять оптимальное значение k.

1.3.7. Деревья решений

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

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

В узлах находятся решающие правила и производится проверка соответствия примеров этому правилу. В результате проверки множество примеров, попавших в узел, разбивается на два подмножества: удовлетворяющие правилу и не удовлетворяющие ему. Затем к каждому подмножеству вновь применяется правило и процедура рекурсивно повторяется пока не будет достигнуто некоторое условие остановки алгоритма. В последнем узле проверка и разбиение не производится и он объявляется листом.

В листе содержится не правило, а подмножество объектов, удовлетворяющих всем правилам ветви, которая заканчивается данным листом. Для классификации — это класс, ассоциируемый с узлом, а для регрессии — соответствующий листу интервал целевой переменной.

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

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

Недостаток деревьев решений - склонность переобучаться. Переобучение в случае дерева решений — это точное распознавание примеров, участвующих в обучении и полная несостоятельность на новых данных. В худшем случае, дерево будет большой глубины и сложной структуры, а в каждом листе будет только один объект. Для решения этой проблемы используют разные критерии остановки алгоритма. Простота метода — одновременно преимущество и недостаток. Из-за этого применение дерева решений ограниченно. Алгоритм не подходит для решения задач с более сложными зависимостями.

1.3.8. Деревья решений

Алгоритм создает несколько деревьев решений, которые объединяются для более точного прогноза.

Логика модели случайного леса заключается в том, что несколько отдельных деревьев решений работают гораздо лучше в группе, чем по отдельности. При использовании Random Forest для классификации каждое дерево дает классификацию или «голосование». Лес выбирает классификацию с большинством «голосов». При использовании Random Forest для регрессии лес выбирает среднее значение результатов всех деревьев.

Ключевым моментом здесь является тот факт, что корреляция между отдельными моделями, то есть между деревьями решений, составляющими более крупную модель случайного леса низка (или отсутствует). В то время как отдельные деревья решений могут давать ошибки, большая часть группы будет правильной, что приведет к общему результату в правильном направлении.

1.3.9. Градиентный бустинг

Градиентный бустинг - 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.

Плюсы: алгоритм работает с любыми функциями потерь, предсказания в среднем лучше, чем у других алгоритмов, самостоятельно справляется с пропущенными данными.

Минусы: алгоритм крайне чувствителен к выбросам и при их наличии будет тратить огромное количество ресурсов на эти моменты. Однако стоит отметить, что использование MAE вместо MSE значительно снижает влияние выбросов на модель.

Модель будет склонна к переобучению при слишком большом количестве деревьев. Данная проблема присутствует в любом алгоритме, связанном с деревьями и исправляется правильной настройкой параметров.

1.3.10. Метод опорных векторов

Метод опорных векторов - 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.

Основная идея метода 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 выборки любого класса. Максимально близкие объекты разных классов определяют опорные вектора. Если в исходном пространстве объекты линейно неразделимы, то выполняется переход в пространство большей размерности.

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

Плюсы: хорошо работает с пространством признаков большого размера, хорошо работает с данными небольшого объема.

Минусы: неустойчивость к шуму, выбросы в обучающих данных становятся опорными объектами-нарушителями и напрямую влияют на построение разделяющей гиперплоскости.

1.3.11. Нейронная сеть

Представляет собой вычислительные системы, основанные на биологических нейронных сетях, которые составляют мозг животных.

Нейронная сеть основана на наборе соединенных блоков или узлов, называемых искусственными нейронами, которые в общих чертах моделируют нейроны биологического мозга. Каждое соединение, подобно синапсам в биологическом мозге, может передавать сигнал другим нейронам.

Искусственный нейрон получает сигналы, затем обрабатывает их и может передавать сигналы нейронам, подключенным к нему. "Сигнал" при соединении представляет собой действительное число, а выходные данные каждого нейрона вычисляются с помощью некоторой нелинейной функции от суммы его входных данных. Соединения называются ребрами. Нейроны и ребра обычно имеют вес, который корректируется по мере обучения. Вес увеличивает или уменьшает силу сигнала при соединении. Нейроны могут иметь пороговое значение, такое, что сигнал отправляется, только если совокупный сигнал пересекает это пороговое значение.

Как правило, нейроны объединены в слои. Разные слои могут выполнять разные преобразования на своих входных данных. Сигналы передаются от

первого слоя (входного слоя) к последнему слою (выходному слою), возможно, после многократного обхода слоев.

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

1.4. Разведочный анализ данных

Для дальнейшей передачи Датасета в процесс машинного обучения, необходимо провести анализ на выбросы, содержание различных ошибок и пропущенных значений.

Уникальные значения

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

Содержание нулевых значений

```
Соотношение матрица-наполнитель          0
Плотность, кг/м3                         0
Модуль упругости, ГПа                    0
Количество отвердителя, м.%              0
Содержание эпоксидных групп,%_2          0
Температура вспышки, С_2                 0
Поверхностная плотность, г/м2            0
Модуль упругости при растяжении, ГПа     0
Прочность при растяжении, МПа            0
Потребление смолы, г/м2                  0
Угол нашивки, град                       0
Шаг нашивки                              0
Плотность нашивки                        0
dtype: int64
```

Рисунок 2 – Значения признаков

Нулевые значения в данных отсутствуют.

Из 13 признаков один является категориальным – «Угол нашивки, град».

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

Рисунок 3 – Описание признаков

В качестве инструментов разведочного анализа используется: оценка статистических характеристик датасета; гистограммы распределения каждой из переменной; диаграммы boxplot (ящика с усами); попарные графики рассеивания точек; тепловая карта; описательная статистика для каждой переменной; анализ и полное исключение выбросов.

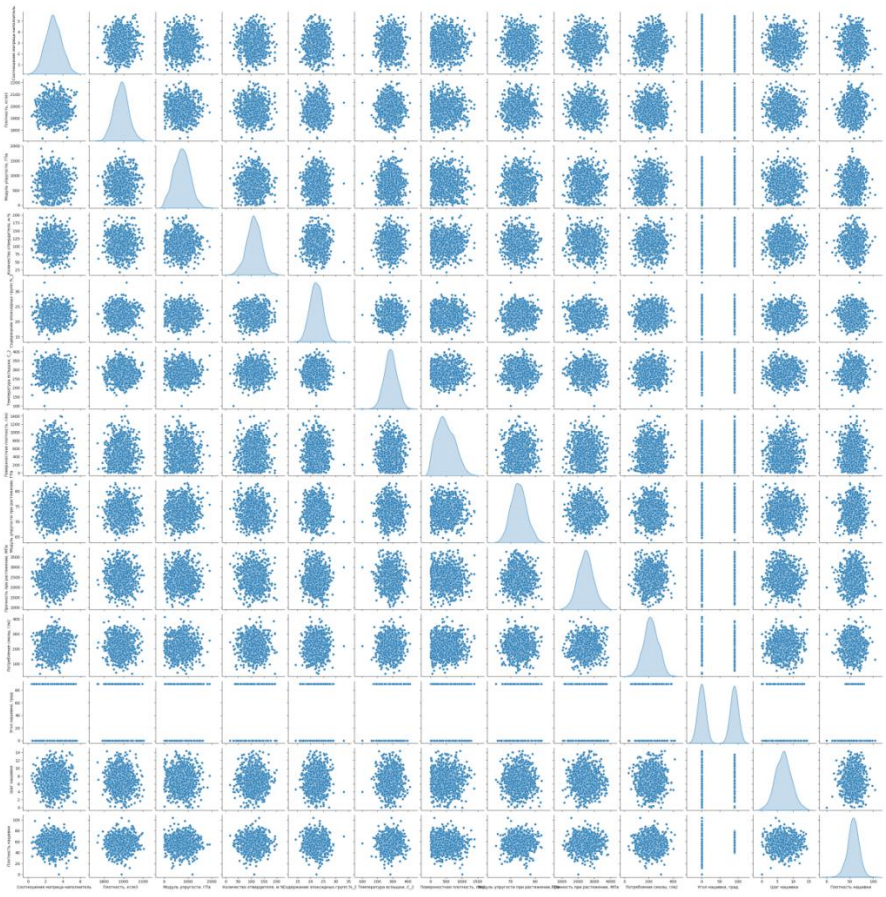


Рисунок 4 – Попарный график рассеивания данных

Графики попарного рассеивания позволяют нам увидеть возможную взаимосвязь между любыми двумя переменными соответствующего набора данных.

На попарных графиках распределения не видно корреляции между признаками. Единственная зависимость, которую можно отметить – это меньшая дисперсия значений Плотности нашивки при 90 градусов Угла нашивки, по сравнению со значением 0 градусов.

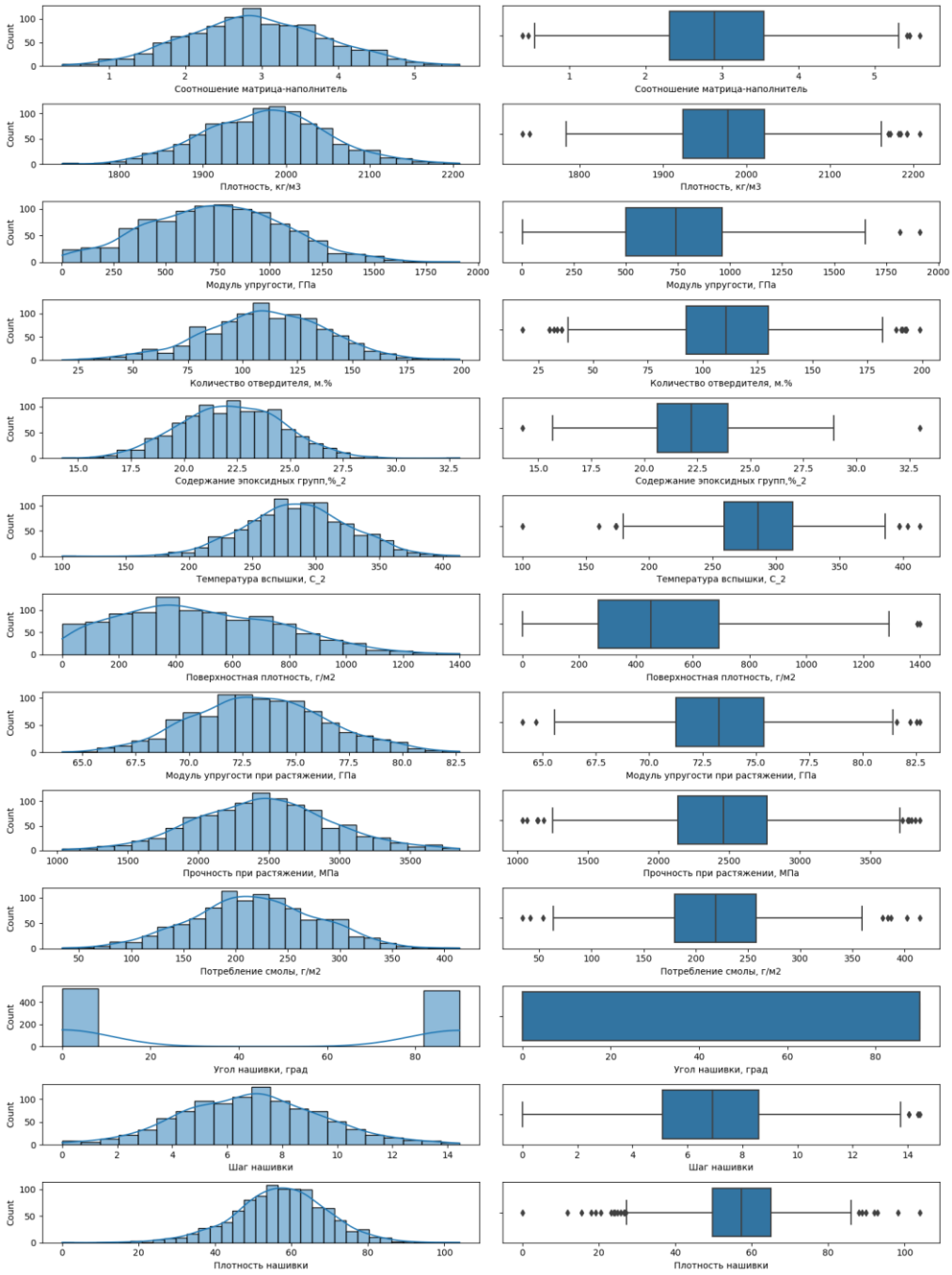


Рисунок 5 – График распределения, «Ящик с усами»

Box-plot показывает наличие выбросов в данных. График слева – нормальное распределение по Гауссу.

Вынесем «Ящик с усами» и кривые на отдельные графики.

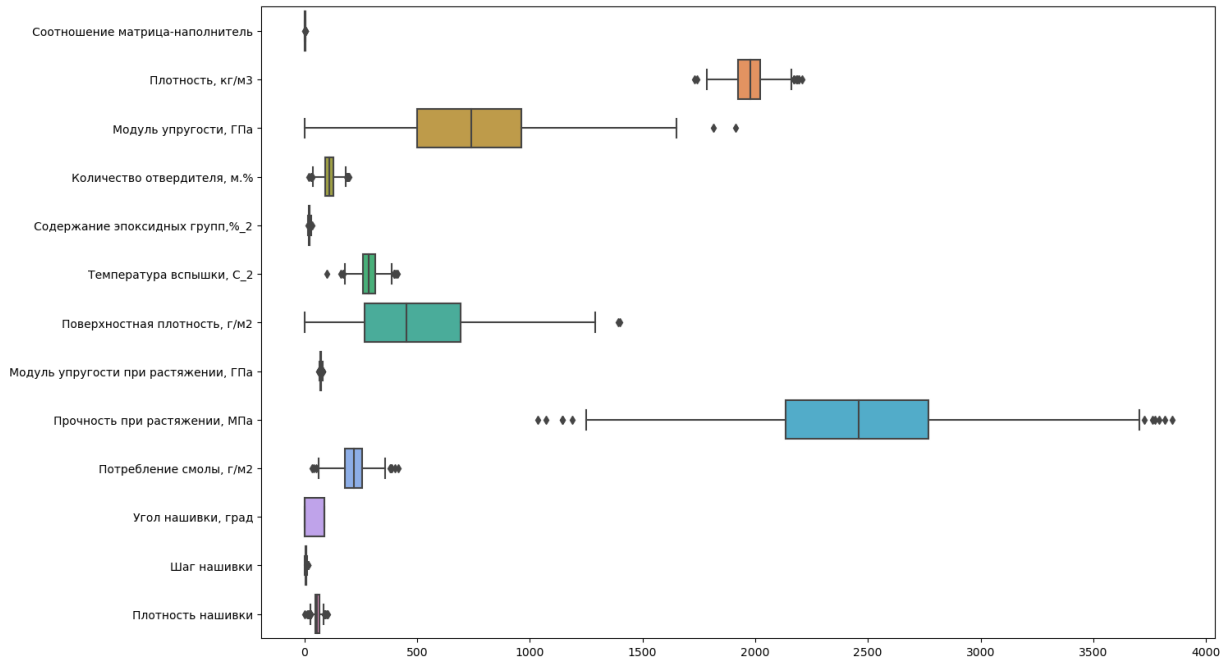


Рисунок 6 – «Ящик с усами»

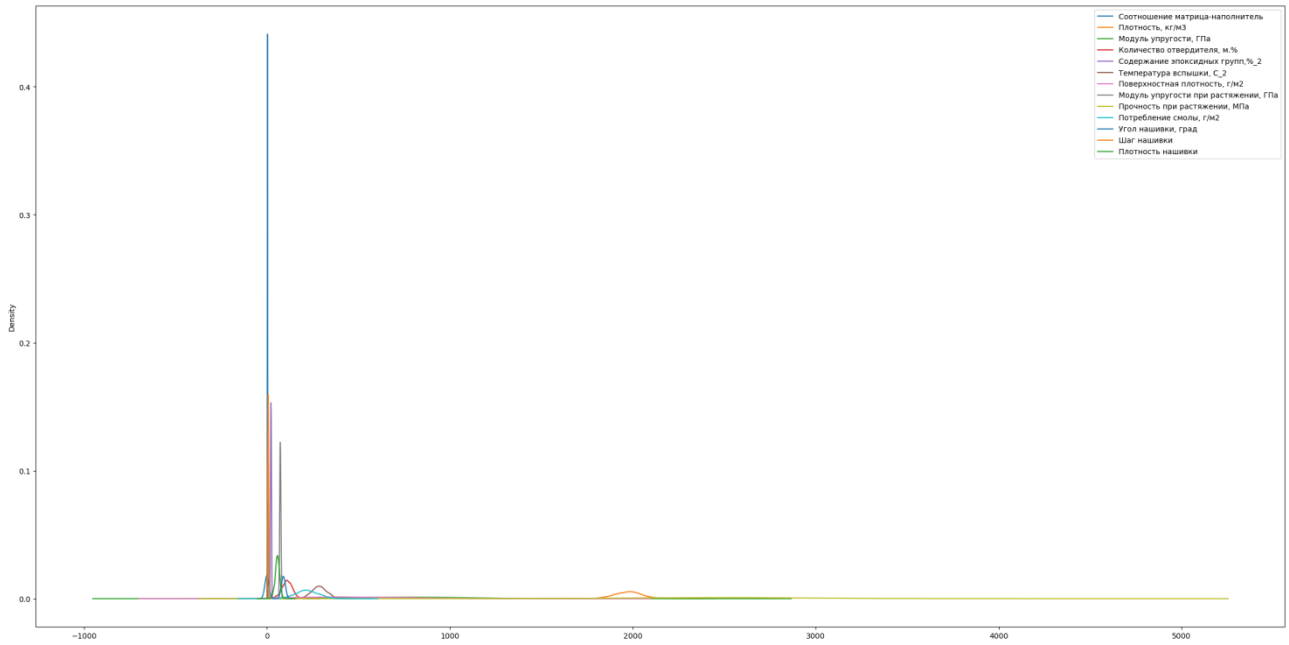


Рисунок 7 – Нормальное распределение

На графиках четко видны выбросы и значительный разброс значений по признакам.

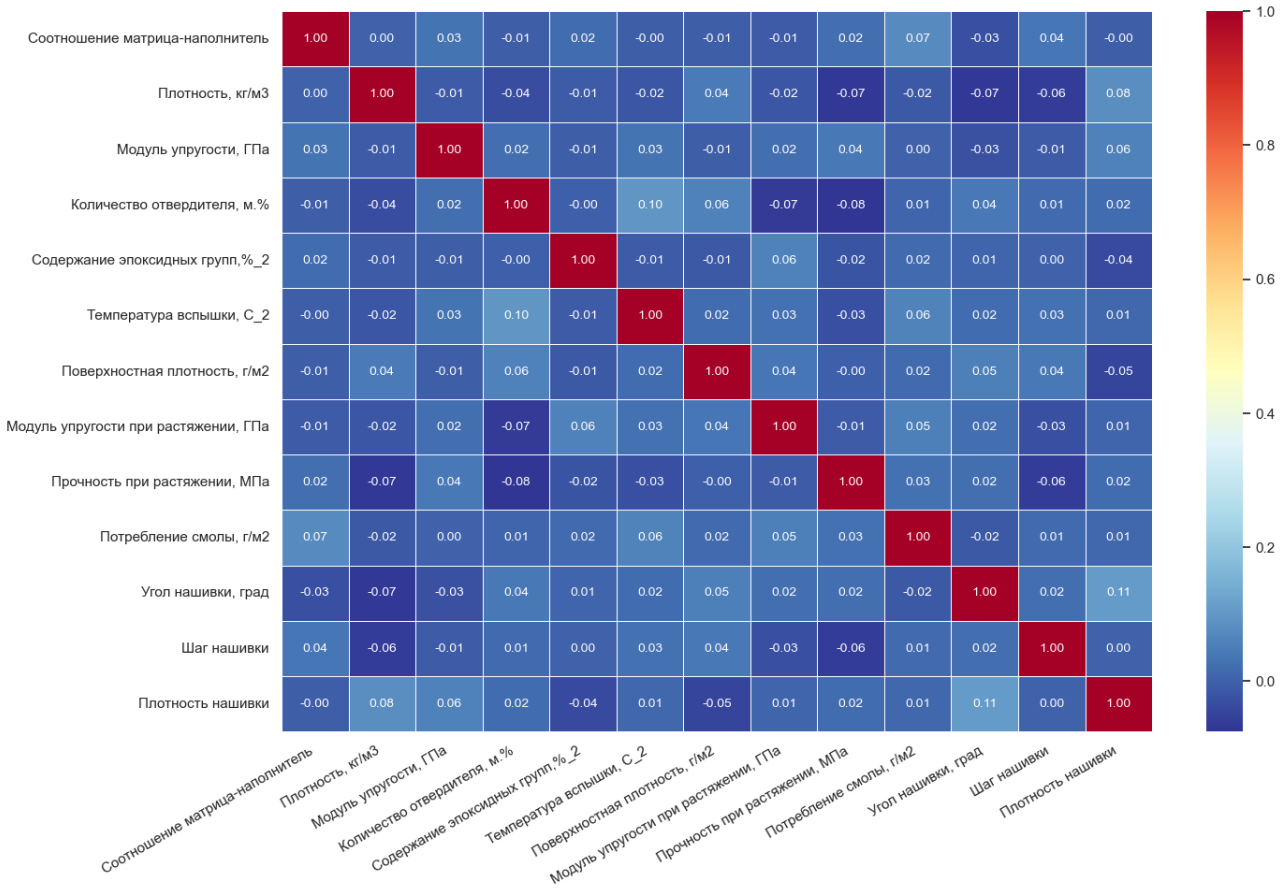


Рисунок 8 – Тепловая карта

Тепловая карта показывает отсутствие корреляции между признаками и целевыми переменными. Значения близки к нулю, связи между переменными не наблюдаются.

2. Практическая часть

2.1. Предобработка данных

В качестве подготовки к дальнейшей обработке данных используем метод межквартильного размаха IQR для очистки от выбросов, а также стандартизируем данные методом StandartScaler.

	count	mean	std	min	25%	50%	75%	max
Соотношение матрица-наполнитель	1023.0	-4.102290e-17	1.000489	-2.783776	-0.671007	-0.025733	0.681760	2.915695
Плотность, кг/м3	1023.0	8.510625e-16	1.000489	-3.310622	-0.699921	0.025603	0.619318	3.148712
Модуль упругости, ГПа	1023.0	3.777797e-16	1.000489	-2.234332	-0.726742	-0.000784	0.672249	3.549589
Количество отвердителя, м.%	1023.0	-5.013910e-17	1.000489	-3.282308	-0.640946	-0.000210	0.677447	3.125033
Содержание эпоксидных групп,%_2	1023.0	-3.429428e-17	1.000489	-3.321825	-0.680362	-0.005674	0.714119	4.471955
Температура вспышки, С_2	1023.0	4.692673e-16	1.000489	-4.542215	-0.655266	0.000358	0.662703	3.112932
Поверхностная плотность, г/м2	1023.0	6.565835e-17	1.000489	-1.714677	-0.767897	-0.109779	0.748614	3.260615
Модуль упругости при растяжении, ГПа	1023.0	-9.653405e-17	1.000489	-2.975023	-0.668350	-0.019172	0.650543	3.000355
Прочность при растяжении, МПа	1023.0	7.937606e-16	1.000489	-2.946217	-0.682074	-0.015242	0.618616	2.846190
Потребление смолы, г/м2	1023.0	-2.799976e-16	1.000489	-3.092116	-0.649770	0.012992	0.654174	3.285517
Угол нашивки, град	1023.0	-2.757434e-15	1.000489	-0.983518	-0.983518	-0.983518	1.016758	1.016758
Шаг нашивки	1023.0	-2.967649e-16	1.000489	-2.692680	-0.710007	0.006604	0.658443	2.943275
Плотность нашивки	1023.0	4.688332e-17	1.000489	-4.629749	-0.595768	0.015228	0.631112	3.793863

Рисунок 9 – Описание датасета после стандартизации

Используя стандартизацию, мы центрируем столбцы признаков по среднему значению 0, чтобы столбцы признаков принимали форму нормального распределения, что упрощает изучение весов. Кроме того, стандартизация сохраняет полезную информацию о выбросах и делает алгоритм менее чувствительным к ним.

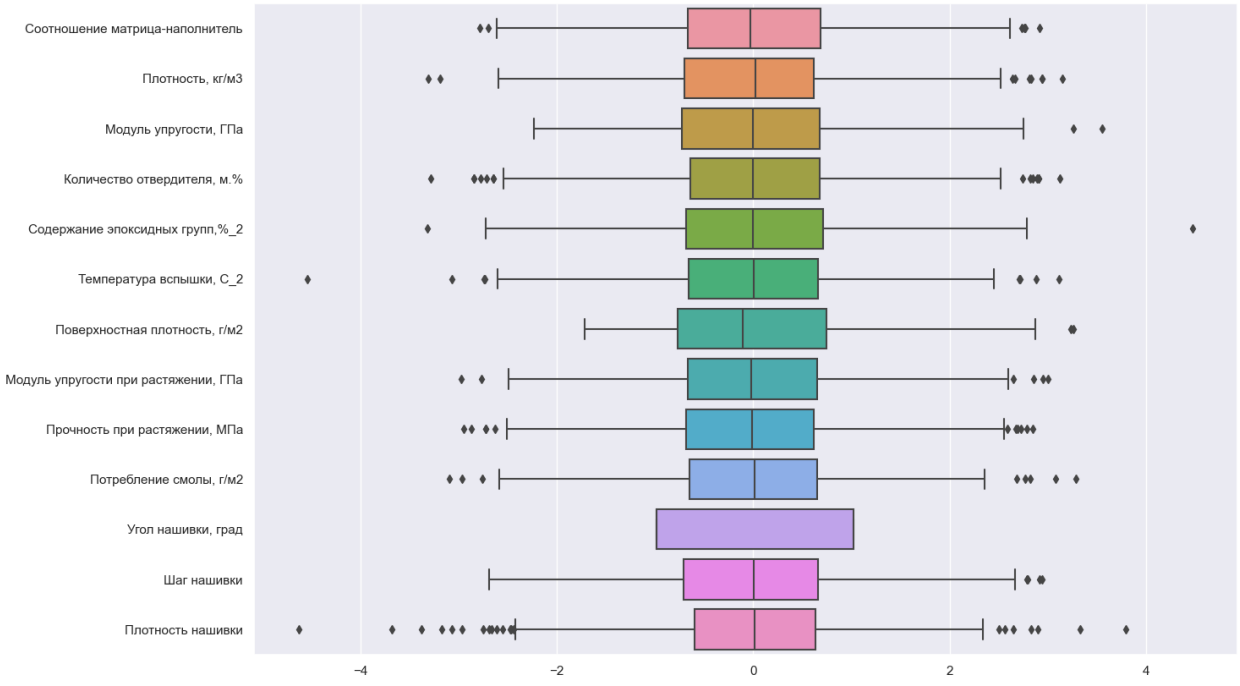


Рисунок 10 - «Ящик с усами» после стандартизации, до удаления выбросов

Далее удаляем выбросы методом межквартильного размаха.

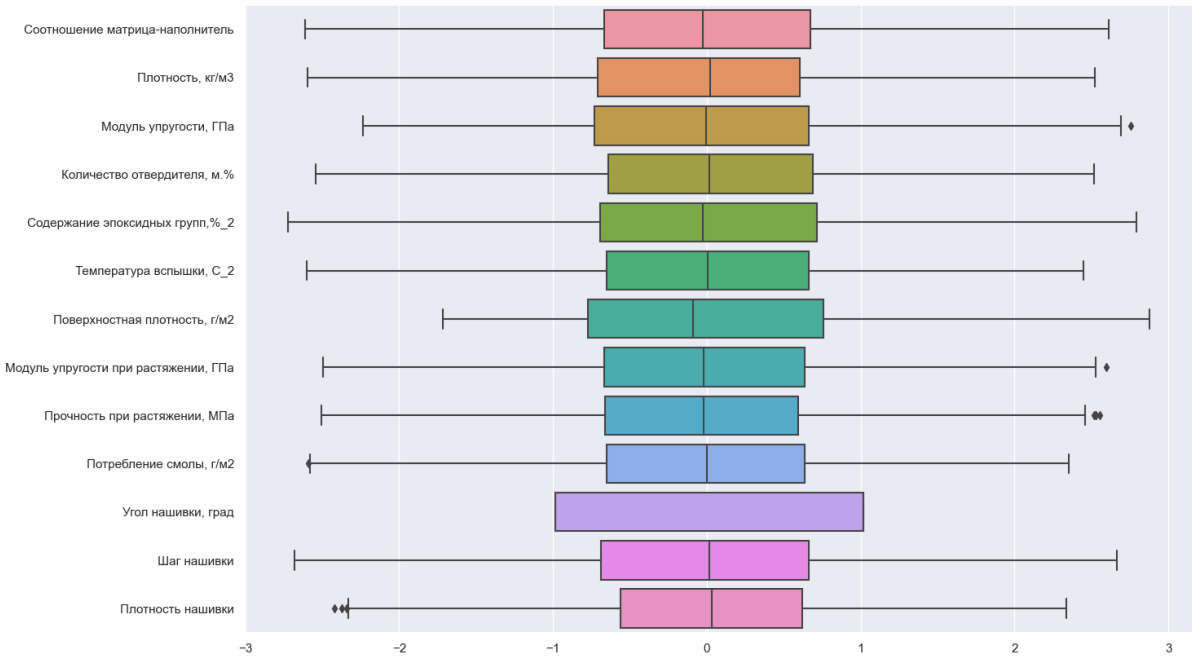


Рисунок 11 - «Ящик с усами» после стандартизации и удаления выбросов

Результат очистки от выбросов виден на графике.

Нормализуем данные методом Normalizer().

Смотрим обновленный очищенный Датасет:

	count	mean	std	min	25%	50%	75%	max
Соотношение матрица-наполнитель	936.0	-0.002047	0.278720	-0.736970	-0.202569	-0.008203	0.199412	0.652068
Плотность, кг/м3	936.0	-0.005779	0.272050	-0.702335	-0.204961	0.006913	0.189967	0.725274
Модуль упругости, ГПа	936.0	-0.002458	0.281938	-0.649405	-0.217683	-0.001201	0.198212	0.713835
Количество отвердителя, м.%	936.0	0.007026	0.270797	-0.722259	-0.187585	0.005521	0.205621	0.734216
Содержание эпоксидных групп,%_2	936.0	-0.005639	0.281418	-0.696394	-0.207710	-0.007297	0.210553	0.667124
Температура вспышки, С_2	936.0	0.001436	0.273659	-0.793328	-0.195383	0.001091	0.194493	0.750782
Поверхностная плотность, г/м2	936.0	-0.002983	0.283842	-0.630006	-0.238900	-0.029166	0.213642	0.760946
Модуль упругости при растяжении, ГПа	936.0	-0.002247	0.275719	-0.672087	-0.203252	-0.006188	0.196761	0.709015
Прочность при растяжении, МПа	936.0	0.001053	0.269270	-0.652666	-0.203522	-0.005199	0.189479	0.752336
Потребление смолы, г/м2	936.0	-0.004237	0.273329	-0.691941	-0.195633	-0.000239	0.192672	0.747426
Угол нашивки, град	936.0	0.017126	0.307042	-0.600179	-0.281021	0.218917	0.302881	0.545564
Шаг нашивки	936.0	-0.000481	0.278545	-0.716617	-0.204390	0.005955	0.203298	0.704588
Плотность нашивки	936.0	0.007557	0.257474	-0.665016	-0.165191	0.010254	0.187391	0.656576

Рисунок 12 – Очищенный и нормализованный датасет

На этом подготовка Датасета к дальнейшему обучению закончена.

Сделана копия Датасета для последующей обработки нейросетью.

2.2. Метрики качества моделей

Список и описание метрик качества моделей регрессии:

R2: коэффициент детерминации. Показывает, насколько условная дисперсия модели отличается от дисперсии реальных значений Y. Если этот коэффициент близок к 1, то условная дисперсия модели достаточно мала и весьма вероятно, что модель неплохо описывает данные. Если же коэффициент R-квадрат сильно меньше, например, меньше 0.5, то, с большой долей уверенности модель не отражает реальное положение вещей.

MSE: метрика, которая сообщает нам среднеквадратичную разницу между прогнозируемыми значениями и фактическими значениями в наборе данных. Чем ниже MSE, тем лучше модель соответствует набору данных.

RMSE (Root Mean Squared Error): метрика, которая сообщает нам квадратный корень из средней квадратичной разницы между прогнозируемыми значениями и фактическими значениями в наборе данных. Чем ниже RMSE, тем лучше модель соответствует набору данных.

MAE (Mean Absolute Error): средняя абсолютная ошибка. Рассчитывается как среднее абсолютных разностей между наблюдаемым и предсказанным значениями.

MAPE (Mean Absolute Percentage Error): средняя абсолютная процентная ошибка - безразмерный показатель, представляющий собой взвешенную версию.

2.3. Разработка и обучение моделей «Модуль упругости при растяжении, ГПа» и «Прочность при растяжении, МПа»

2.3.1. Обработка параметров

Согласно заданию делим Датасет на тестовые и тренировочные значения методом train_test_split в соответствии 70:30.

```
X = X_normalize.drop(['Модуль упругости при растяжении, ГПа', 'Прочность при растяжении, МПа'], axis=1)
y = X_normalize[['Модуль упругости при растяжении, ГПа', 'Прочность при растяжении, МПа']]
```

```
X.shape, y.shape
```

```
((936, 11), (936, 2))
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=14)
```

```
print('X_train: ', X_train.shape, '  y_train:  ', y_train.shape)
print('X_test:  ', X_test.shape,  '  y_test:   ', y_test.shape)
```

```
X_train:  (655, 11)   y_train:   (655, 2)
X_test:   (281, 11)   y_test:    (281, 2)
```

```
# Запись в XLSX
X_percent.to_excel('X_percent.xlsx')
```

```
y_test_mod = y_test.drop('Прочность при растяжении, МПа', axis=1)
y_test_end = y_test.drop('Модуль упругости при растяжении, ГПа', axis=1)

y_train_mod = y_train.drop('Прочность при растяжении, МПа', axis=1)
y_train_end = y_train.drop('Модуль упругости при растяжении, ГПа', axis=1)
```

```
print('y_train_mod: ', y_train_mod.shape, '  y_train_end:  ', y_train_end.shape)
print('y_test_mod:  ', y_test_mod.shape,  '  y_test_end:   ', y_test_end.shape)
```

```
y_train_mod:  (655, 1)   y_train_end:   (655, 1)
y_test_mod:   (281, 1)   y_test_end:    (281, 1)
```

Рисунок 13 – Деление датасета на тренировочные и тестовые выборки

2.3.2. Разработка и обучение моделей

В качестве поиска лучших параметров для используемых регрессоров, применил метод GridSearchCV(). Для старта перечислил все основные гиперпараметры.

```
# стартовые настройки регрессоров для поиска лучших гиперпараметров

param_linear = {}

param_dummy = {'strategy': ['mean', 'median','quantile'],
               'quantile': [0.005, 0.01, 0.05, 0.1, 0.25, 0.5, 1]}

param_lasso = {'alpha' : [0.005, 0.01, 0.05, 0.1, 0.25, 0.5, 1],
               'selection' : ['cyclic', 'random']}

param_ridge = {'alpha': [0.1, 0.5, 1.0],
               'solver': ['auto', 'svd', 'cholesky', 'lsqr', 'sparse_cg', 'sag', 'saga']}

param_elastic = {'alpha' : [0.01, 0.015, 0.02, 0.025, 0.05, 0.5, 1.0, 10.0],
                 'selection' : ['cyclic', 'random']}

param_neighbor = {'n_neighbors': range (28, 42, 1),
                  'weights': ['uniform', 'distance'],
                  'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute']
                 }

param_decision = {'criterion': ['squared_error', 'friedman_mse', 'absolute_error'],
                  'splitter': ['best', 'random'],
                  'max_depth': range (5, 15, 1),
                  'min_samples_leaf': range (2, 4, 1),
                  'min_samples_split': range (2, 4, 1),
                  'max_features': range(1, 11, 1)
                 }
```

```
param_random = {'criterion': ['squared_error', 'friedman_mse', 'absolute_error'],
                'max_depth': range (1, 3, 1),
                'min_samples_leaf': range (2, 6, 1),
                'min_samples_split': range (2, 10, 1),
                'n_estimators': range (6, 12, 1)
               }

param_gradient = {'loss': ['squared_error', 'absolute_error', 'huber', 'quantile'],
                  'criterion': ['friedman_mse', 'squared_error'],
                  'max_features': ['sqrt', 'log2'],
                  'max_depth': range (1, 5, 1)
                 }

param_svr = {'kernel': ['linear', 'poly', 'rbf', 'sigmoid'],
             'gamma': ['scale', 'auto']}
```

Рисунок 14 – Настройки регрессоров

Итогом запуска цикла по поиску гиперпараметров стало составление списка для двух признаков.

	name	feature	best_parameters
1	LINEAR	модуль	LinearRegression()
2		прочность	LinearRegression()
3	DUMMY	модуль	DummyRegressor(quantile=0.005)
4		прочность	DummyRegressor(quantile=0.005)
5	LASSO	модуль	Lasso(alpha=0.01)
6		прочность	Lasso(alpha=0.005)
7	RIDGE	модуль	Ridge(solver='lsqr')
8		прочность	Ridge(solver='lsqr')
9	ELASTIC	модуль	ElasticNet(alpha=0.015)
10		прочность	ElasticNet(alpha=0.01)
11	K-NEIGHBORS	модуль	KNeighborsRegressor(n_neighbors=41)
12		прочность	KNeighborsRegressor(n_neighbors=39, weights='distance')
13	DECISION TREE	модуль	DecisionTreeRegressor(max_depth=5, max_features=1, min_samples_leaf=3, random_state=14, splitter='random')
14		прочность	DecisionTreeRegressor(max_depth=7, max_features=1, min_samples_leaf=2, random_state=14, splitter='random')
15	RANDOM FOREST	модуль	RandomForestRegressor(max_depth=1, min_samples_leaf=2, n_estimators=9, random_state=14)
16		прочность	RandomForestRegressor(max_depth=1, min_samples_leaf=2, n_estimators=6, random_state=14)
17	GRADIENT BOOSTING	модуль	GradientBoostingRegressor(loss='absolute_error', max_depth=1, max_features='sqrt', random_state=14)
18		прочность	GradientBoostingRegressor(max_depth=1, max_features='sqrt', random_state=14)
19	SVR	модуль	SVR(gamma='auto', kernel='poly')
20		прочность	SVR(gamma='auto', kernel='poly')

Рисунок 15 – Список лучших параметров для регрессоров

2.3.3. Тестирование моделей

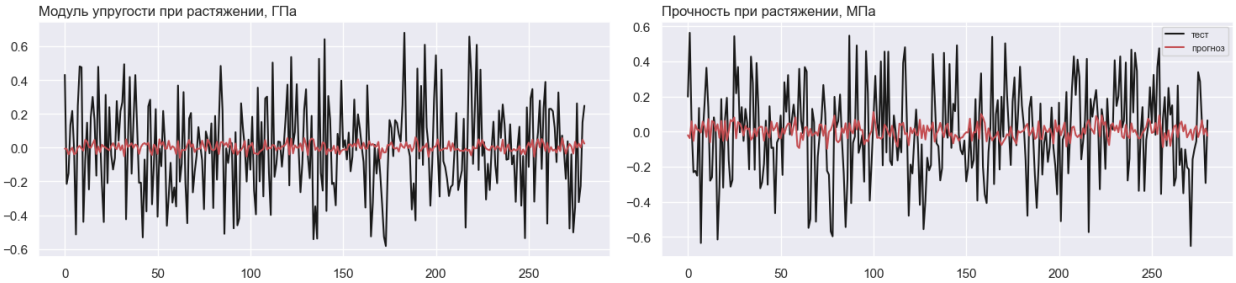
Тестируем обе модели по лучшим параметрам. Выводим на печать.

Тестовые и прогнозные значения: LINEAR

Тестовые и прогнозные значения: DUMMY

Тестовые и прогнозные значения: LASSO

Тестовые и прогнозные значения: RIDGE

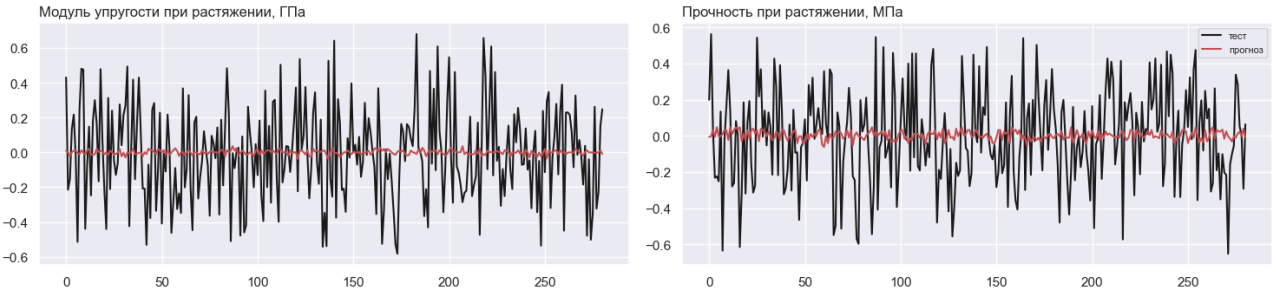


Тестовые и прогнозные значения: ELASTIC

Тестовые и прогнозные значения: K-NEIGHBORS

Тестовые и прогнозные значения: DECISION TREE

Тестовые и прогнозные значения: RANDOM FOREST



Тестовые и прогнозные значения: GRADIENT BOOSTING

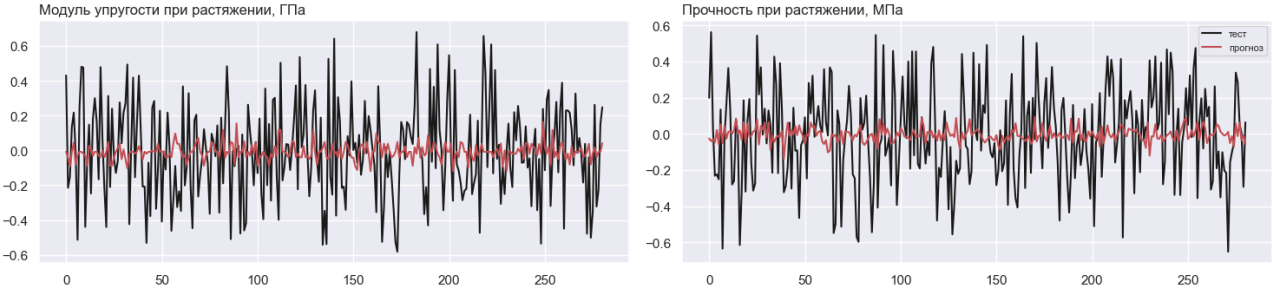


Рисунок 16 – Графики соответствия прогноза тестовой выборки

2.3.4. Ошибки

	name	feature	mse	rmse	mae	r2	mape
1	LINEAR	модуль	0.072562	0.269373	0.217889	0.019288	1.352428
2		прочность	0.071976	0.268284	0.217090	-0.031393	9.434725
3	DUMMY	модуль	0.074034	0.272091	0.220546	-0.000602	1.002576
4		прочность	0.069822	0.264239	0.213066	-0.000531	6.574755
5	LASSO	модуль	0.074034	0.272091	0.220546	-0.000602	1.002576
6		прочность	0.069822	0.264239	0.213066	-0.000531	6.574755
7	RIDGE	модуль	0.072577	0.269401	0.217926	0.019085	1.346937
8		прочность	0.071897	0.268136	0.216948	-0.030259	9.296989
9	ELASTIC	модуль	0.074034	0.272091	0.220546	-0.000602	1.002576
10		прочность	0.069822	0.264239	0.213066	-0.000531	6.574755
11	K-NEIGHBORS	модуль	0.074317	0.272611	0.221565	-0.004426	1.445657
12		прочность	0.072537	0.269328	0.215942	-0.039435	183.634952
13	DECISION TREE	модуль	0.074954	0.273778	0.221138	-0.013047	1.360824
14		прочность	0.070986	0.266432	0.215070	-0.017206	21.991708
15	RANDOM FOREST	модуль	0.073929	0.271900	0.220563	0.000807	1.122600
16		прочность	0.071029	0.266512	0.214664	-0.017816	68.978934
17	GRADIENT BOOSTING	модуль	0.076182	0.276012	0.223623	-0.029645	1.687428
18		прочность	0.071058	0.266568	0.216150	-0.018241	2.355036
19	SVR	модуль	0.073975	0.271984	0.220236	0.000186	1.096575
20		прочность	0.069821	0.264236	0.212943	-0.000504	18.598072

Рисунок 17 – Ошибки моделей

Визуально ни одна модель не отработала хорошо. Список полученных ошибок говорит о том же.

2.4. Нейросеть для рекомендации «Соотношение матрица-наполнитель»

2.4.1. Обработка параметров

За основу взят подготовленный заранее подготовленный Датасет (п. 2.1.1).

Делим Датасет на тестовые и тренировочные значения методом train_test_split в соответствии 70:30.

```
X = X_neuro.drop(['Соотношение матрица-наполнитель'], axis=1)
y = X_neuro[['Соотношение матрица-наполнитель']]
```

```
x_train, x_val, y_train, y_val = train_test_split(X, y, test_size=0.3, random_state=14)
```

```
y_val.describe().T
```

	count	mean	std	min	25%	50%	75%	max
Соотношение матрица-наполнитель	281.0	2.896197	0.918636	0.596783	2.28456	2.862056	3.502063	5.314144

```
print(x_train.shape, x_val.shape)
print()
print(y_train.shape, y_val.shape)
```

```
(655, 12) (281, 12)

(655, 1) (281, 1)
```

Рисунок 18 – Деление датасета на тестовую и валидационную выборки

2.4.2. Разработка и обучение модели

Для поиска лучших параметров настройки нейросети был взят инструмент Tuner из библиотеки Keras Tensor Flow.

Настраиваем Tuner на поиск параметров. В том числе, по оптимальному количеству слоев модели.

```python
def build_model(hp):

    model = keras.Sequential()

#    model.add(layers.Flatten())
    model.add(LayerNormalization())

    for i in range(hp.Int('num_layers', 1, 5)):
        model.add(layers.Dense(units=hp.Int(f'units_{i}', min_value=4,
                                            max_value=12, step=2),
                               activation=hp.Choice('activation', ['relu', 'tanh']),
                               )
                  )

    if hp.Boolean('dropout'):
        model.add(layers.Dropout(rate=0.25))

    model.add(layers.Dense(1,))

    optimizer = hp.Choice('optimizer', ['sgd','rmsprop','adam'])

    learning_rate = hp.Choice('learning_rate', values = [.1, .01, .001, .0001])

    model.compile(optimizer=optimizer,
                  loss='mean_squared_error',
                  metrics=['mean_squared_error'],
                 )

    return model
```

Рисунок 19 – Настройка Тюнера

После определения области поиска подготавливаем классы тюнера для запуска. Выбираем BayesianOptimization, RandomSearch и Hyperband.

```python
Bayes = kt.BayesianOptimization(hypermodel=build_model, objective='val_loss',
                                max_trials=50, executions_per_trial=3, overwrite=True)

Random = kt.RandomSearch(hypermodel=build_model, objective='val_loss',
                         max_trials=50, executions_per_trial=3, overwrite=True)

Hyperband = kt.Hyperband(hypermodel=build_model, objective='val_loss',
                         max_epochs=50, executions_per_trial=3, overwrite=True)
```

Рисунок 20 – Используемые классы тюнеров

Запускаем Tuner и получаем лучшие параметры.

```
Best Hyperparameters

---------------------
Bayesian Optimization
---------------------
{'num_layers': 3, 'units_0': 10, 'activation': 'relu', 'dropout': False, 'optimizer': 'rmsprop', 'learning_rate': 0.01, 'units_1': 10, 'units_2': 12, 'units_3': 8, 'units_4': 8}

---------------------
Random Search
---------------------
{'num_layers': 2, 'units_0': 12, 'activation': 'relu', 'dropout': False, 'optimizer': 'adam', 'learning_rate': 0.0001, 'units_1': 8, 'units_2': 12, 'units_3': 12, 'units_4': 6}

---------------------
Hyperband
---------------------
{'num_layers': 3, 'units_0': 6, 'activation': 'tanh', 'dropout': True, 'optimizer': 'sgd', 'learning_rate': 0.001, 'units_1': 4, 'units_2': 6, 'units_3': 10, 'units_4': 12, 'tuner/epoch
s': 17, 'tuner/initial_epoch': 6, 'tuner/bracket': 3, 'tuner/round': 2, 'tuner/trial_id': '0040'}
```

```
---------------------
Bayesian Optimization
---------------------
Results summary
Results in .\untitled_project
Showing 1 best trials
Objective(name="val_loss", direction="min")

Trial 06 summary
Hyperparameters:
num_layers: 3
units_0: 10
activation: relu
dropout: False
optimizer: rmsprop
learning_rate: 0.01
units_1: 10
units_2: 12
units_3: 8
units_4: 8
Score: 0.833955983320872
None
```

```
---------------------
Random Search
---------------------
Results summary
Results in .\untitled_project
Showing 1 best trials
Objective(name="val_loss", direction="min")

Trial 46 summary
Hyperparameters:
num_layers: 2
units_0: 12
activation: relu
dropout: False
optimizer: adam
learning_rate: 0.0001
units_1: 8
units_2: 12
units_3: 12
units_4: 6
Score: 0.8341850241025289
None
```

```
---------------------
Hyperband
---------------------
Results summary
Results in .\untitled_project
Showing 1 best trials
Objective(name="val_loss", direction="min")

Trial 0047 summary
Hyperparameters:
num_layers: 3
units_0: 6
activation: tanh
dropout: True
optimizer: sgd
learning_rate: 0.001
units_1: 4
units_2: 6
units_3: 10
units_4: 12
tuner/epochs: 17
tuner/initial_epoch: 6
tuner/bracket: 3
tuner/round: 2
tuner/trial_id: 0040
Score: 0.8319942951202393
None
```

Рисунок 21 –Лучшие параметры для моделей

2.4.3. Тестирование моделей

Тестируем модели по лучшим параметрам. Выводим на печать.

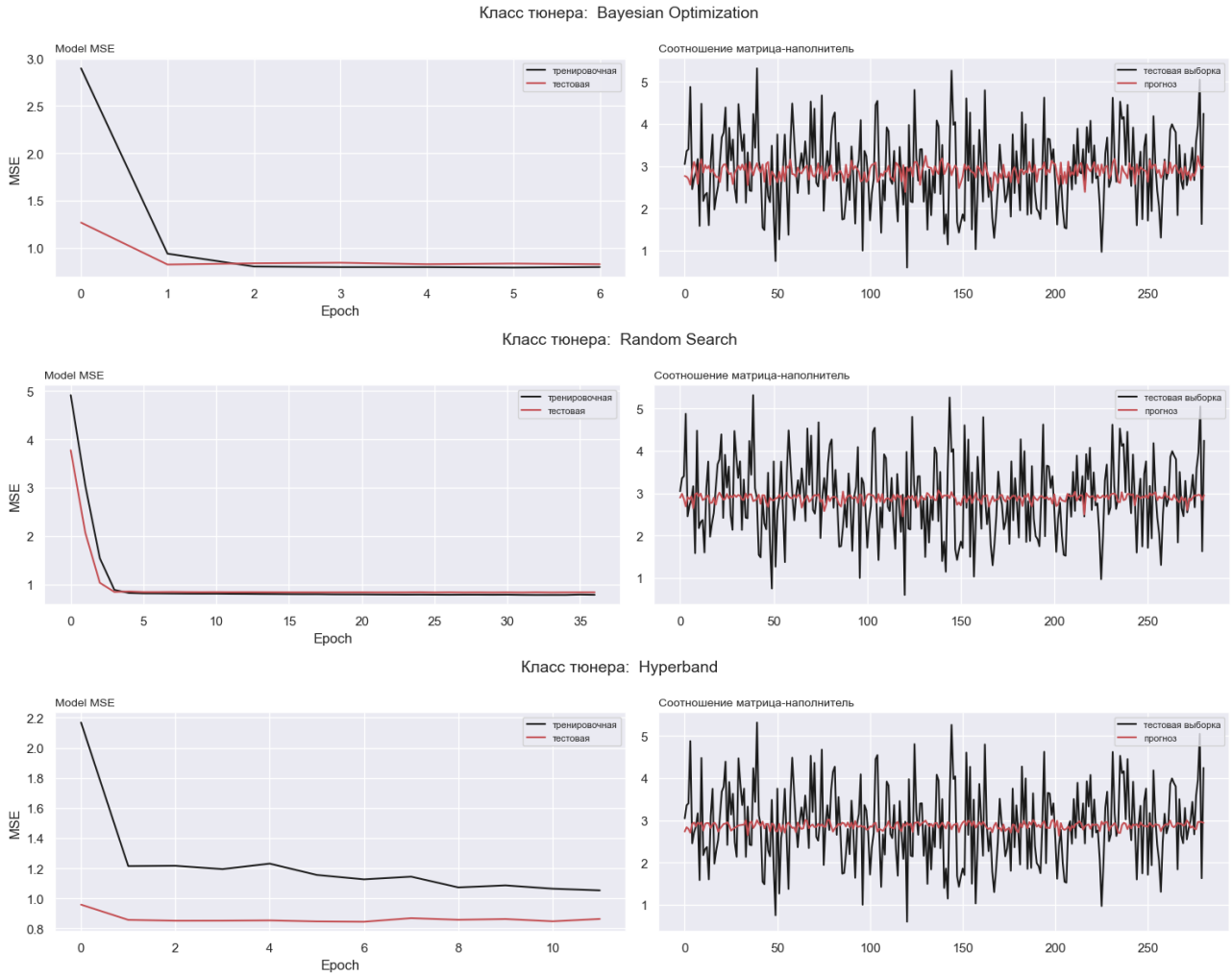


Рисунок 22 – График ошибок по эпохам и предсказание на тестовых данных

2.4.4. Ошибки

	name	mse	rmse	mae	r2	mape
1	Bayesian Optimization	0.824209	0.907859	0.735155	0.019836	0.314291
2	Random Search	0.835548	0.914083	0.741446	0.006351	0.318499
3	Hyperband	0.844739	0.919097	0.741930	-0.004579	0.317525

Рисунок 23 – Ошибки моделей нейросети

Как и в случае с регрессиями, визуально, ни одна модель нейросети не отработала хорошо. Этот вывод подтверждает список полученных ошибок.

Заключение

В ходе выполнения выпускной работы были выполнены все задачи предложенного пайплайна.

Проведенная исследовательская работа позволяет сделать некоторые основные выводы по теме:

1) Распределение данных соответствует нормальному.
2) Корреляция между парами признаков стремится к нулю.
3) Использованные при разработке моделей методы не позволили получить достоверные прогнозы.
4) Применённые модели регрессии и нейросеть не показали эффективности в прогнозировании свойств композитов.

Список используемых в работе ресурсов

1. Документация по библиотеке Keras: https://keras.io/api/
2. Документация по библиотеке Matplotlib: https://matplotlib.org/stable/plot_types/index
3. Документация по библиотеке Numpy: https://numpy.org/doc/1.22/user/index.html#user
4. Документация по библиотеке Pandas: https://pandas.pydata.org/docs/user_guide/index.html#user-guide
5. Документация по библиотеке Scikit-learn: https://scikit-learn.org/stable/user_guide.html
6. Документация по библиотеке Seaborn: https://seaborn.pydata.org/tutorial.html
7. Документация по библиотеке Tensorflow: https://www.tensorflow.org/?hl=ru
8. Документация по языку программирования Python: https://docs.python.org/3.9/