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 Андреева Елена Анатольевна

Москва, 2022

Оглавление

Введение

Тема данной выпускной квалификационной работы – прогнозирование конечных свойств новых композиционных материалов.

Композиционные материалы — это любые материалы, сделанные из более чем одной составляющей, поэтому вокруг нас много композиционных материалов, которые мы используем, не задумываясь об их принадлежности к классу композитов. Современные композитные материалы обычно состоят из двух компонентов: наполнителя и матрицы.

Внедрение композиционных материалов обусловлено стремлением использовать их преимущества по сравнению с традиционно используемыми металлами и сплавами. Примеры композита – железобетон (сочетание стали арматуры и камня бетона), древесноволокнистая плита ДВП (сочетание древесной основы – щепы и полимерного связующего).

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

Целью данной работы является разработка эффективной модели прогнозирования конечных свойств новых композиционных материалов на основе данных об их составе и структуре. Для достижения этой цели в работе рассматриваются различные подходы к анализу данных и методы машинного обучения, а также проводится сравнительный анализ результатов работы различных моделей.

В ходе работы были использованы данные о композиционных материалах, содержащие информацию об их составе и структуре, а также о конечных свойствах, которые необходимо было прогнозировать. Были

проведены эксперименты с различными алгоритмами машинного обучения, в том числе с использованием нейронных сетей, деревьев решений, метода опорных векторов и случайных лесов. Созданные прогнозные модели помогут сократить количество проводимых испытаний, а также пополнить базу данных материалов возможными новыми характеристиками материалов, и цифровыми двойниками новых композитов.

Результаты работы могут быть использованы в дальнейших исследованиях в области материаловедения, а также могут быть применены в производственных условиях для оптимизации процесса разработки новых материалов с требуемыми свойствами.

1. Аналитическая часть

1.1. Постановка задачи

Для исследовательской работы были даны 2 файла: X_bp.xlsx (с данными о параметрах базальтопластика, состоящий из 1024 строки и 11 столбцов) и X_nup.xlsx (данными нашивок углепластика, состоящий из 1041 строки и 4 столбцов).

```
In [41]: #загружаем первый датасет (базальтопластик), смотрим размерность, затем название столбцов
         df_bp = pd.read_excel('C:\Data Science\ВКР\Андреева\Датасет\X_bp.xlsx')
         df_bp.shape
Out[41]: (1023, 11)

In [42]: #удаляем первый неинформативный столбец
         df_bp = pd.read_excel('C:\Data Science\ВКР\Андреева\Датасет\X_bp.xlsx')
         df_bp.drop(['Unnamed: 0'], axis=1, inplace=True)

In [43]: #Посмотрим на первые 5 строк первого датасета и убедимся, что первый столбец удалился
         df_pb.head()
```

Out[43]:

	Соотношение матрица-наполнитель	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	Потребление смолы, г/м2
0	1.857143	2030.0	738.736842	30.00	22.267857	100.000000	210.0	70.0	3000.0	220.0
1	1.857143	2030.0	738.736842	50.00	23.750000	284.615385	210.0	70.0	3000.0	220.0
2	1.857143	2030.0	738.736842	49.90	33.000000	284.615385	210.0	70.0	3000.0	220.0
3	1.857143	2030.0	738.736842	129.00	21.250000	300.000000	210.0	70.0	3000.0	220.0
4	2.771331	2030.0	753.000000	111.86	22.267857	284.615385	210.0	70.0	3000.0	220.0

```
In [44]: # Проверим размерность первого файла
         df_pb.shape
Out[44]: (1023, 10)
```

Рисунок 1 – пример начала работы с файлом X_bp.xlsx

```
In [46]: # загружаем второй датасет (углепластик)
         df_nup = pd.read_excel('C:\Data Science\ВКР\Андреева\Датасет\X_nup.xlsx')
         df_nup.shape
Out[46]: (1040, 4)

In [47]: #Удаляем первый неинформативный столбец
         df_nup.drop(['Unnamed: 0'], axis=1, inplace=True)

In [48]: #Посмотрим на первые 5 строк второго датасета и убедимся, что и здесь не нужный первый столбец удален
         df_nup.head()
```

Out[48]:

	Угол нашивки, град	Шаг нашивки	Плотность нашивки
0	0	4.0	57.0
1	0	4.0	60.0
2	0	4.0	70.0
3	0	5.0	47.0
4	0	5.0	57.0

```
In [49]: #проверим размерность второго файла
         df_nup.shape
Out[49]: (1040, 3)
```

Рисунок 2 - пример начала работы с файлом X_nup.xlsx

Известно, что файлы требуют объединения с типом INNER по индексу. После объединения часть строк из файла X_nup была отброшена. И дальнейшие

исследования проводим с объединенным датасетом, содержащим 13 признаков и 1023 строк или объектов.

```
In [50]: #Мы знаем, что эти два датасеты отличаются объемом строк.
         #Наша задача собрать исходные данные файлы в один, единый набор данных.
         #По условию задачи объединяем их по типу INNER.
         df = df_bp.merge(df_nup, left_index = True, right_index = True, how = 'inner')
         df.head().T
```

Out[50]:

	0	1	2	3	4
Соотношение матрица-наполнитель	1.857143	1.857143	1.857143	1.857143	2.771331
Плотность, кг/м3	2030.000000	2030.000000	2030.000000	2030.000000	2030.000000
модуль упругости, ГПа	738.736842	738.736842	738.736842	738.736842	753.000000
Количество отвердителя, м.%	30.000000	50.000000	49.900000	129.000000	111.860000
Содержание эпоксидных групп,%_2	22.267857	23.750000	33.000000	21.250000	22.267857
Температура вспышки, С_2	100.000000	284.615385	284.615385	300.000000	284.615385
Поверхностная плотность, г/м2	210.000000	210.000000	210.000000	210.000000	210.000000
Модуль упругости при растяжении, ГПа	70.000000	70.000000	70.000000	70.000000	70.000000
Прочность при растяжении, МПа	3000.000000	3000.000000	3000.000000	3000.000000	3000.000000
Потребление смолы, г/м2	220.000000	220.000000	220.000000	220.000000	220.000000
Угол нашивки, град	0.000000	0.000000	0.000000	0.000000	0.000000
Шаг нашивки	4.000000	4.000000	4.000000	5.000000	5.000000
Плотность нашивки	57.000000	60.000000	70.000000	47.000000	57.000000

```
In [51]: #Проверяем размер полученного файла.
         df.shape
         # Итоговый датасет имеет 13 столбцов и 1023 строки. Соответственно, 17 строк из таблицы X_nup было отброшено,т.е часть данных уд
Out[51]: (1023, 13)
```

Рисунок 3 – пример объединения датасетов с типом объединения INNER

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

Затем провести разведочный анализ данных, нарисовать гистограммы распределения каждой из переменной, диаграммы ящика с усами, попарные графики рассеяния точек. По ним видно, что все признаки, кроме «Угол нашивки», имеют нормальное распределение и принимают неотрицательные значения. «Угол нашивки» принимает значения: 0, 90.

Нам известно, что датасет был предварительно подготовлен, поэтому отсутствие пропусков не удивило. В сырых данных пропуски и значения некорректных типов как правило присутствуют.

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

Попарные графики рассеяния точек приведены на рисунке 4.

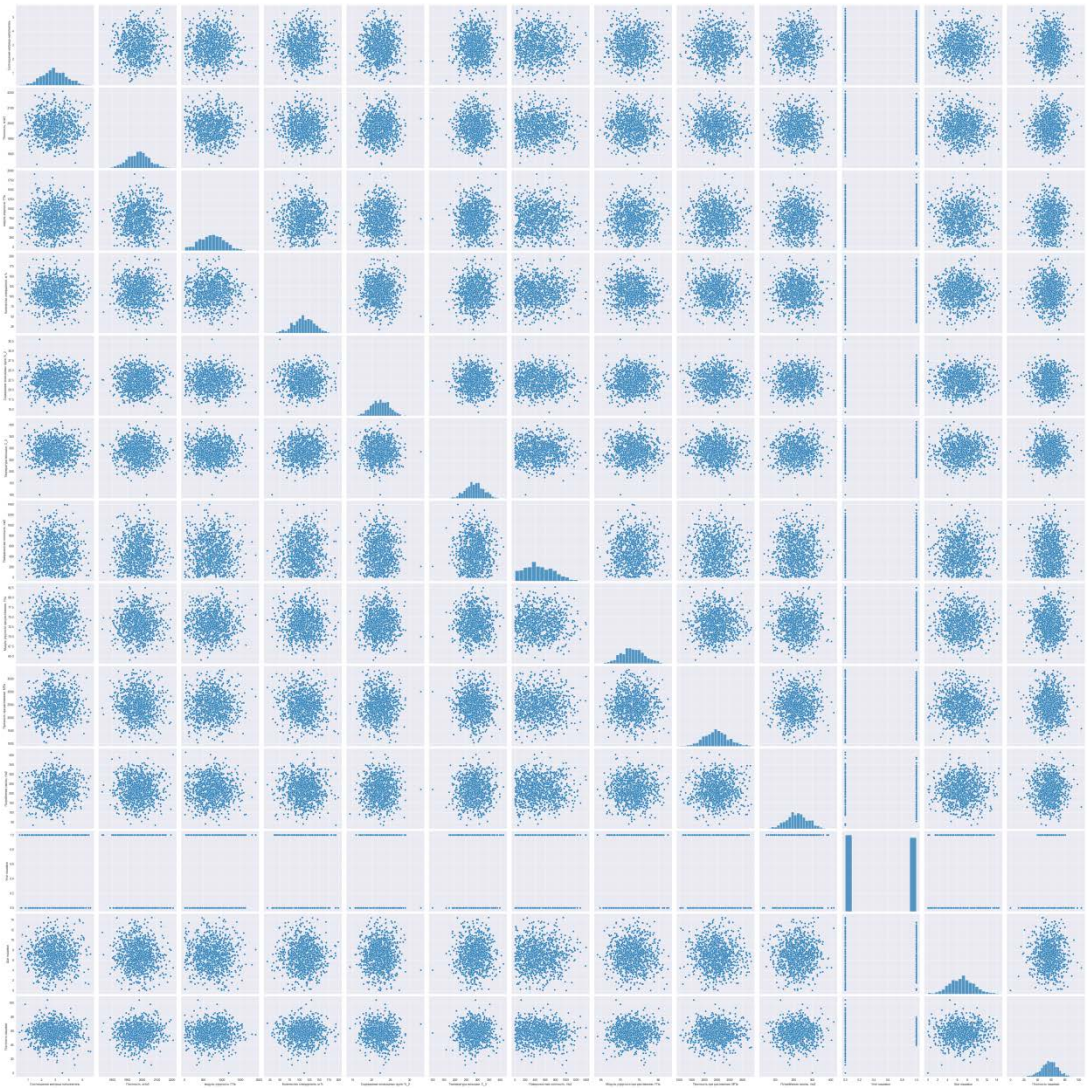

Рисунок 4 - Попарные графики рассеяния точек

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

Есть следующие методы выявления выбросов для признаков с нормальным распределением:

- метод 3-х сигм;
- метод межквартильных расстояний.

Применив эти методы на нашем датасете было найдено:

- методом 3-х сигм — 24 выброса;
- методом межквартильных расстояний — 93 выброса.

Пример выбросов на гистограмме распределения и диаграмме «ящик с усами» приведен на рисунке 5.

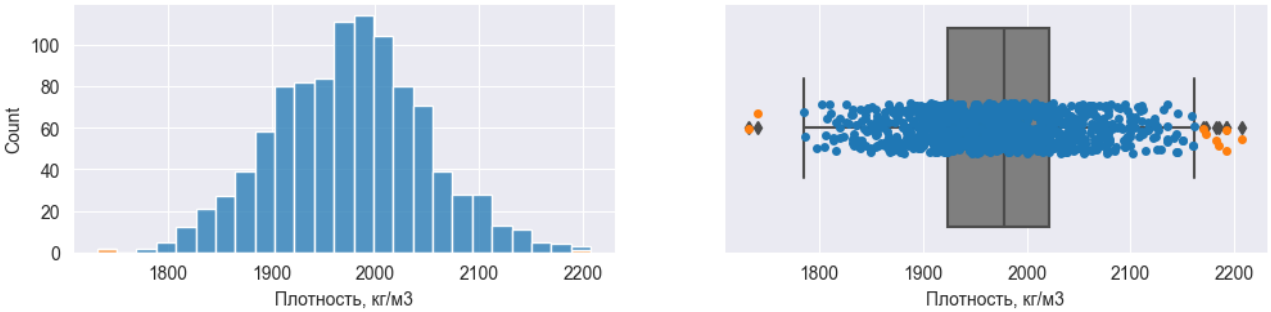

Рисунок 5 – пример выбросов

Поскольку известно, что датасет очищен от явного шума, следует применить метод 3-х сигм как более деликатный, чтобы не потерять значимые данные. Значения, определенные как выбросы, удаляем. После этого осталось в датасете осталось 1000 строк и 13 признаков-переменных.

В задании целевыми переменными указаны:

- модуль упругости при растяжении, Гпа;
- прочность при растяжении, МПа;
- соотношение матрица-наполнитель.

1.2. Описание используемых методов

Для решения данной задачи регрессии было использовано несколько методов машинного обучения:

- Lasso - Лассо (Лассо-регрессия);
- LinearRegression - Линейная регрессия;
- Ridge - Гребневая регрессия;
- DecisionTreeRegressor - Регрессионное дерево решений;
- RandomForestRegressor - Случайный лес регрессии;
- ElasticNetCV – Эластичная сеть регрессии;
- SVR - Метод опорных векторов для регрессии;
- BayesianRidge - Байесовская линейная регрессия;

- KernelRidge - Ядерная регрессия.

Лассо (Лассо-регрессия) — это метод регуляризации линейной регрессии, который использует L1-регуляризацию. Он работает путем добавления штрафа за сумму абсолютных значений коэффициентов регрессии, что приводит к сокращению некоторых коэффициентов до нуля, что позволяет выполнять отбор признаков.

Линейная регрессия — это простой метод машинного обучения для построения линейной модели, которая моделирует связь между входными признаками и выходными целевыми значениями. Он работает путем минимизации суммы квадратов ошибок между прогнозируемыми и реальными значениями.

Гребневая регрессия (Ridge) — это метод регуляризации линейной регрессии, который использует L2-регуляризацию. Он работает путем добавления штрафа за квадраты значений коэффициентов регрессии, что ограничивает их значения и уменьшает переобучение.

Дерево решений регрессии (Decision Tree Regression) — это метод нелинейной модели, который использует дерево решений для моделирования зависимости между признаками и целевой переменной. Дерево решений представляет собой структуру данных, которая состоит из узлов и листьев. Узлы представляют собой разделение признаков на две или более ветви, а листья — это конечные значения, которые являются предсказанными значениями для данного наблюдения. Дерево решений строится по обучающей выборке путем рекурсивного разделения на подгруппы, каждый раз выбирая признак, который максимально снижает дисперсию (или уменьшает ошибку).

Случайный лес регрессии — это метод ансамблирования множества деревьев решений, чтобы получить более сильную модель. Он работает путем построения множества деревьев решений, которые каждое используют подмножество признаков и данных, и затем усредняет результаты прогнозирования всех деревьев.

Эластичная сеть регрессии (ElasticNetCV) - метод регрессии, который является комбинацией Лассо-регрессии и Гребневой регрессии. Он минимизирует сумму квадратов ошибок и включает в функционал потерь штрафы как на L1-, так и на L2-нормы коэффициентов регрессии. Это позволяет улучшить результаты в случаях, когда в данных присутствуют коррелированные признаки.

Метод опорных векторов для регрессии (SVR) – это метод регрессии, который использует максимальный зазор между опорными векторами для построения линейной или нелинейной модели регрессии. Он минимизирует сумму квадратов ошибок и добавляет ограничения на значения коэффициентов регрессии в зависимости от расстояний до опорных векторов.

Байесовская линейная регрессия (Bayesian Ridge) - это метод регрессии, который использует байесовский подход для оценки коэффициентов регрессии и шума в данных. Он минимизирует сумму квадратов ошибок и моделирует коэффициенты регрессии как случайные переменные с нормальным распределением.

Ядерная регрессия (Kernel Ridge) — это метод регрессии, который использует ядерные функции для построения нелинейной модели регрессии. Он минимизирует сумму квадратов ошибок и добавляет штраф на сложность модели за счет добавления к функционалу потерь L2-нормы коэффициентов регрессии.

Таблица 1 - Сравнительная таблица с указанием априорных предпосылок к работоспособности каждого метода

Метод машинного обучения	**Априорные предпосылки**
Лассо (Лассо-регрессия)	Наличие признаков, которые не являются важными для модели; мультиколлинеарность между признаками.
Линейная регрессия	Линейная зависимость между входными

	признаками и выходными целевыми значениями
Гребневая регрессия (Ridge)	Мультиколлинеарность между признаками; необходимость сокращения размерности признаков
Регрессионное дерево решений	Нелинейная зависимость между входными признаками и выходными целевыми значениями
Случайный лес регрессии	Наличие множества слабых моделей, которые могут быть объединены в более сильную модель
Эластичная сеть регрессии (ElasticNetCV)	Эффективна в случаях, когда в данных присутствуют коррелированные признаки.
Метод опорных векторов для регрессии (SVR)	Нелинейная зависимость между входными признаками и выходными целевыми значениями; необходимость нахождения гиперплоскости, которая разделяет данные на две части с наибольшим зазором
Байесовская линейная регрессия (Bayesian Ridge)	Эффективна при работе с данными, когда нужно моделировать неопределенность в коэффициентах регрессии и шуме данных
Ядерная регрессия (Kernel Ridge)	Нелинейная зависимость между входными признаками и выходными целевыми значениями; необходимость применения ядерной функции для построения нелинейных границ

1.3. Разведочный анализ данных

Разведочный анализ данных — это анализ основных свойств данных, нахождение в них общих закономерностей, распределений и аномалий, построение начальных моделей, зачастую с использованием инструментов визуализации.

В данном проекте были использованы следующие методы разведочного анализа данных:

- Описательная статистика данного датасета.
- Визуальный анализ гистограмм

- Визуальный анализ диаграмм размаха («ящик с усами»)
- Проверка нормальности распределения по критерию Пирсона
- Анализ попарных графиков рассеяния переменных
- Корреляционный анализ с целью поиска коэффициентов

Для просмотра статистических данных для в каждого столбца используется метод describe( ). Этот метод показывает количество строк в столбце - count, среднее значение столбца - mean, столбец стандартное отклонение - std, минимальные (min) и максимальные (max) значения, а также границу каждого квартиля - 25%, 50% и 75%. Любые значения NaN автоматически пропускаются.

```
#Изучим описательную статистику наших данных:
# max, min, квартили, медиана, стандартное отклонение, среднее значение, посмотрим на основные параметры анализа данных
df.describe().T
```

	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	1023.0	0.491691	0.500175	0.000000	0.000000	0.000000	1.000000	1.000000
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 6 - Описательная статистика

Проверим датасет на наличие пропущенных данных. Как видно на рисунке 7, пропущенные данные отсутствуют.

```
: a = df.isnull()
  a.sum()
```

```
: 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                            0
  Шаг нашивки                             0
  Плотность нашивки                       0
  dtype: int64
```

```
: # Пропущенных данных нет = нулевых значений нет, очистка не требуется
```

Рисунок 7 - Наличие пропущенных данных

Для просмотра количества дубликатов в датасете используем метод duplicated().

```
# Проверим датасет на наличие дубликатов. Дубликаты отсутствуют.
df.duplicated().sum()
```

```
0
```

Рисунок 8 - Проверка на наличие дубликатов

Для оценки величины и характера разброса данных построены гистограммы распределения для каждого из признаков.

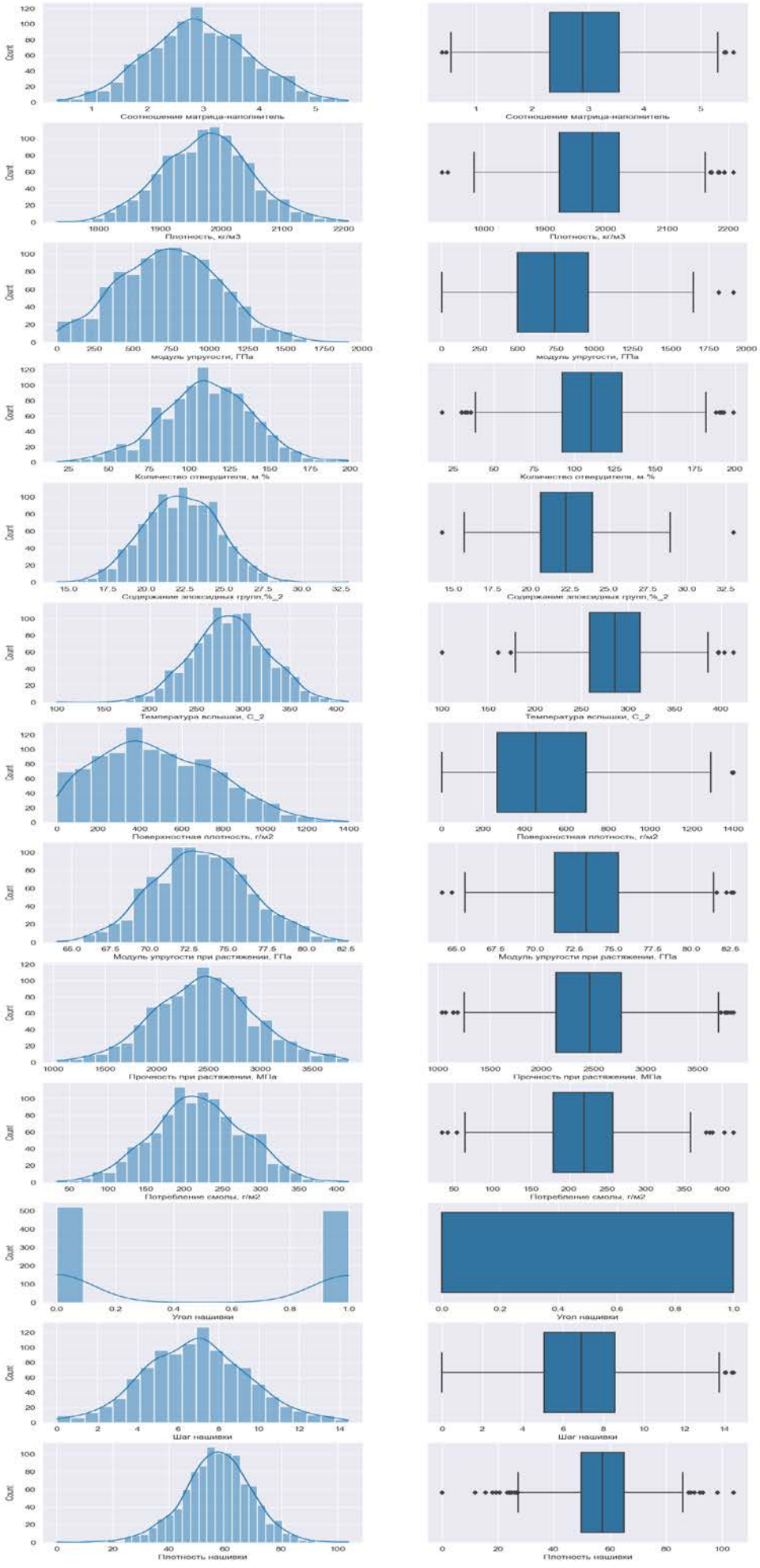
Count
Соотношение матрица-наполнитель
Плотность, кг/м3
модуль упругости, ГПа
Количество отвердителя, м %
Содержание эпоксидных групп,%_2
Температура вспышки, С_2
Поверхностная плотность, г/м2
Модуль упругости при растяжении, ГПа
Прочность при растяжении, МПа
Потребление смолы, г/м2
Угол нашивки
Шаг нашивки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

Рисунок 9 - Гистограммы распределения и «ящики с усами»

Гистограммы показывают что в основном распределения данных близки к нормальному, за исключением столбца с данными “Угол нашивки, град”, который представлен двумя значениями.

Для проверки датасета на наличие выбросов был проведён визуальный анализ диаграмм размаха. Диаграммы размаха показывают медиану, верхний квартиль, нижний квартиль, межквартильный размах, выбросы. На «ящиках с усами» хорошо заметны наличия выбросов в данном датасете.

Так же был проведён анализ попарных графиков рассеяния переменных. Попарный график помогает нам визуализировать распределение отдельных переменных, а также взаимосвязи между двумя переменными. Это отличный метод определения тенденций между переменными для последующего анализа. Если точки на графике расположены вдоль прямой линии, это может указывать на линейную связь между переменными. Если точки расположены в случайном порядке на графике, это может указывать на отсутствие связи между переменными или на то, что связь между переменными сложна и не может быть описана простой линейной связью. Также попарный график рассеяния может помочь идентифицировать выбросы, что являются отдельными наблюдениями, которые выделяются на графике и могут оказывать большое влияние на результаты анализа.

График оценки плотности показывает, как вероятность распределена по значению непрерывной случайной величины. Он может использоваться для визуализации формы распределения, а также для получения информации о вероятности различных значений этой случайной величины.

График оценки плотности может использоваться для выявления выбросов, определения типа распределения, сравнения распределений между собой, оценки вероятности событий и принятия решений на основе статистических данных.

Оценка плотности ядра показывает, что значения переменных находятся в разных диапазонах поэтому требуется нормализация данных.

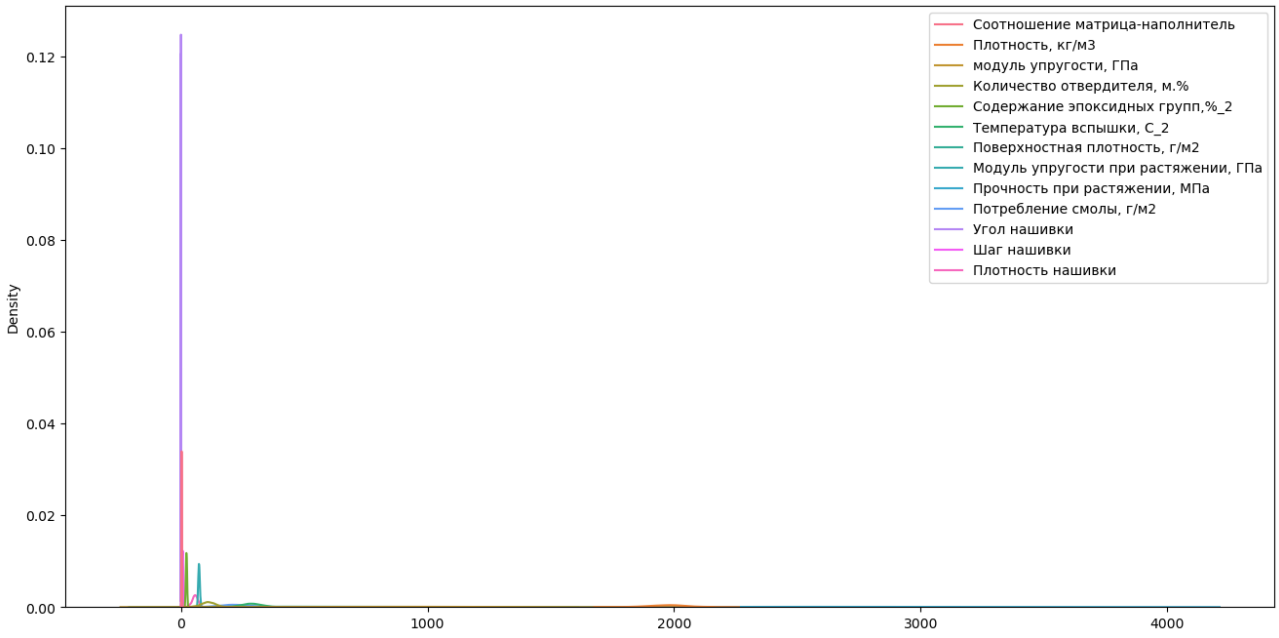

Рисунок 10 - График оценки плотности распределения

Для визуализации коэффициентов корреляции и определения наличия между переменными зависимости, была построена тепловая карта коэффициентов корреляции методом Пирсона.

Корреляция Пирсона — это статистический показатель, который используется для измерения степени линейной связи между двумя переменными. Она измеряет силу и направление линейной связи между двумя непрерывными переменными.

Коэффициент корреляции Пирсона принимает значения от -1 до 1. Значение -1 указывает на полную обратную линейную связь, значение 1 указывает на полную прямую линейную связь, а значение 0 указывает на отсутствие линейной связи между переменными.

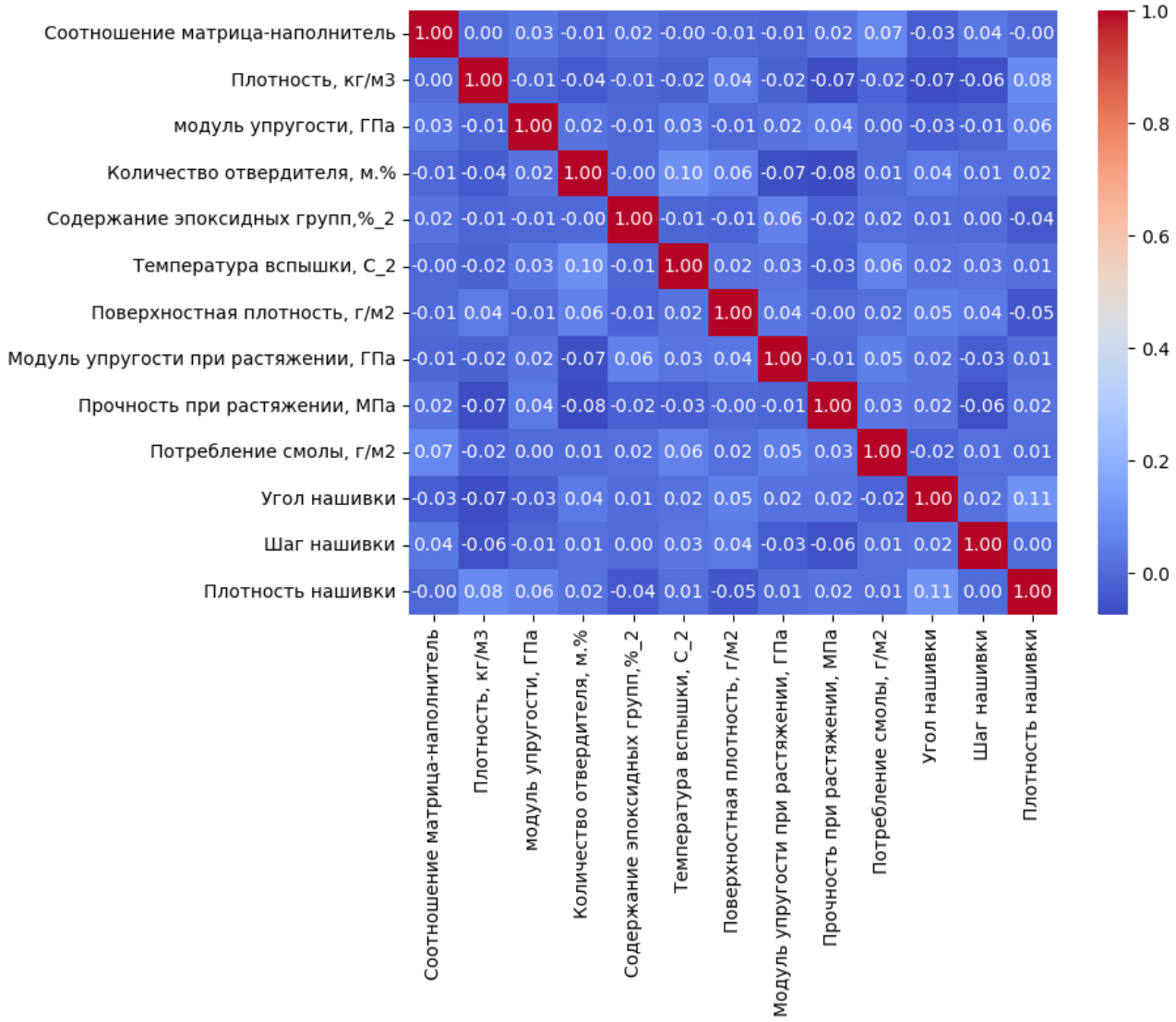

Рисунок 11 - Тепловая карта

Корреляции между переменными практически не наблюдается.

2. Практическая часть

2.1 Предобработка данных

Для удаления выбросов в датасете был выбран метод исключения выбросов с помощью правила трёх сигм.

Правило трех сигм — это статистический метод для определения выбросов в наборе данных. Этот метод основан на стандартном отклонении данных от среднего значения. Согласно правилу трех сигм, большинство данных должны находиться в пределах трех стандартных отклонений от среднего значения. Используя эту концепцию, можно определить, какие данные являются выбросами.

```
In [204]: # Удаляем выбросы в датасете с помощью метода трёх сигм
          outliers = pd.DataFrame(index=df1.index) # Создание пустого датафрейма для записи выбросов
          for column in df1: # запускаем цикл по каждому столбцу датафрема
              zscore = (df1[column] - df1[column].mean()) / df1[column].std() # рассчитывание Z-оценки для каждого столбца
              outliers[column] = (zscore.abs() > 3) #определяем выбросы с помощью абсолютного значения Z-оценки > 3
          df1 = df1[outliers.sum(axis=1)==0] # фильтруем, оставляя только строки без выбросов
          df1.shape

Out[204]: (1000, 13)

In [205]: # Сохраним очищенные данные
          df1.to_excel('C:\Data Science\BKP\data_cleaned.xlsx')
```

Рисунок 12 - Удаление выбросов

Очищенный от выбросов датасет содержит 1000 строк. С помощью диаграмм размаха проверяем на сколько хорошо были удалены выбросы в датасете.

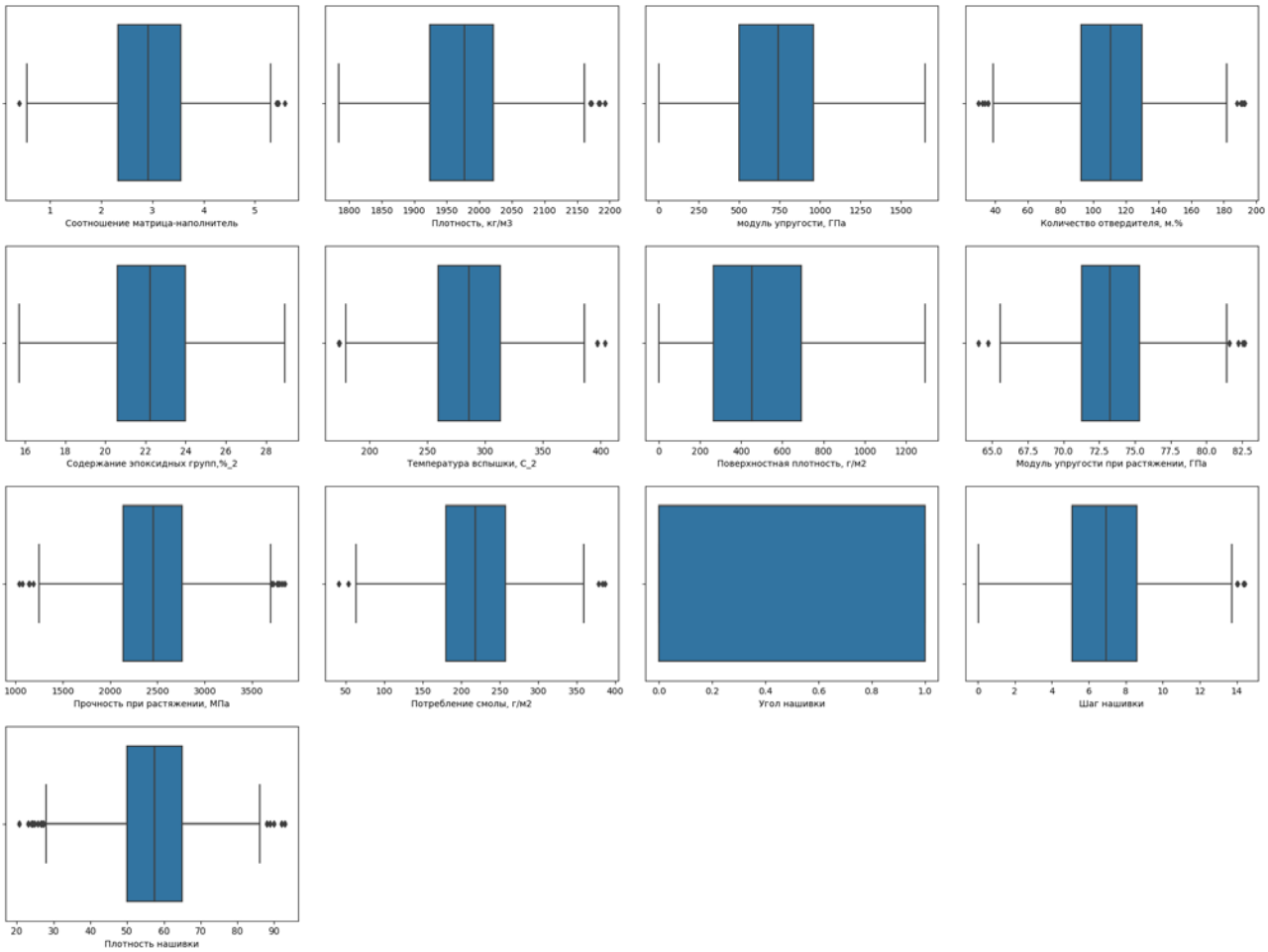

Рисунок 13 - Диаграмма рассеивания

Так как признак «Угол нашивки» представлен только двумя значениями: “0” и “90”, то эти значения были заменены на 0 и 1.

```
#Заменим категориальные признаки в столбце 'Угол нашивки, град' на значения 0 и 1
df_1 = df_1.replace({'Угол нашивки, град':{0.0:0, 90.0:1}})
df_1.head()
```

Рисунок 14 - Замена категориальных признаков

Для приведения всех признаков в датасете к одинаковой шкале была сделана нормализация с помощью метода MinMaxScaler из библиотеки sklearn.

Нормализация — это процесс приведения данных к определенному масштабу или диапазону значений. Нормализация может использоваться для приведения различных признаков в датасете к единому масштабу, чтобы они имели одинаковый вклад в результаты анализа или моделирования.

Нормализация данных является важным шагом в подготовке данных для машинного обучения и может улучшить результаты алгоритмов, повысить их скорость и устойчивость к шуму.

Нормализация (MinMaxScaler) является одним из методов нормализации данных. Этот метод используется для масштабирования признаков в диапазоне от 0 до 1. Он работает путем пересчета значений признаков на основе их минимального и максимального значений в наборе данных.

Формула для MinMax-нормализации следующая:

$$x\_scaled = (x - x\_min) / (x\_max - x\_min)$$

где:

x_scaled - отмасштабированное значение признака;

x - оригинальное значение признака;

x_min - минимальное значение признака в наборе данных;

x_max - максимальное значение признака в наборе данных.

Процесс нормализации MinMaxScaler включает в себя следующие шаги:

- Определение минимального и максимального значений признака в наборе данных.
- Применение формулы MinMax-нормализации для каждого значения признака.
- Получение отмасштабированных значений признаков в диапазоне от 0 до 1.

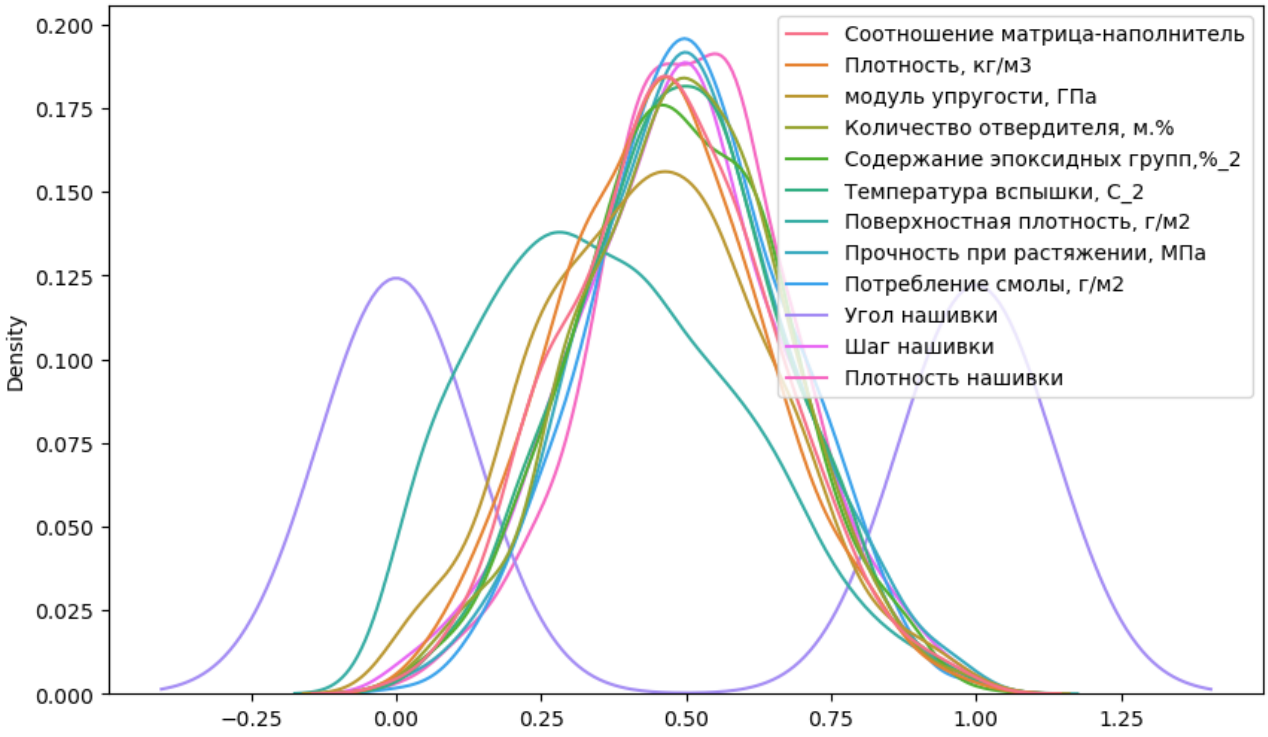

Рисунок 15 - График оценки плотности распределения после нормализации

Описательная статистика тренировочной выборки до нормализации.

In [215]:
```
# Описательная статистика выборки train до нормализации данных
X_upr_train.describe().T
```

Out[215]:

	count	mean	std	min	25%	50%	75%	max
Соотношение матрица-наполнитель	700.0	2.943860	0.902194	0.547391	2.334089	2.920689	3.547394	5.591742
Плотность, кг/м3	700.0	1972.286516	73.148332	1784.482245	1919.833477	1970.707511	2020.188578	2192.738783
модуль упругости, ГПа	700.0	738.627618	326.130594	2.436909	498.031611	738.847005	955.961892	1649.415706
Количество отвердителя, м.%	700.0	112.119243	28.056458	33.624187	92.841533	112.146169	131.073351	192.851702
Содержание эпоксидных групп,%_2	700.0	22.179055	2.335087	15.695894	20.602284	22.151350	23.886426	28.907470
Температура вспышки, С_2	700.0	286.449560	40.645101	173.973907	258.714220	286.396906	313.695527	403.652861
Поверхностная плотность, г/м2	700.0	481.805877	278.253589	1.668002	266.978731	463.506297	687.180113	1288.691844
Прочность при растяжении, МПа	700.0	2469.109198	493.531741	1071.123751	2135.886086	2455.974462	2782.111990	3848.436732
Потребление смолы, г/м2	700.0	216.838475	58.108052	41.048278	177.036759	216.541911	255.408701	386.903431
Угол нашивки	700.0	0.495714	0.500339	0.000000	0.000000	0.000000	1.000000	1.000000
Шаг нашивки	700.0	6.880379	2.590968	0.037639	5.137770	6.906595	8.586121	14.033215
Плотность нашивки	700.0	57.403269	12.036623	20.571633	49.642226	57.300796	65.159797	92.963492

Рисунок 16 - Описательная статистика до нормализации

Описательная статистика тренировочной выборки после нормализации.

```
In [216]: # Описательная статистика выборки train после нормализации данных
          X_upr_train_norm.describe().T
```

Out[216]:

	count	mean	std	min	25%	50%	75%	max
Соотношение матрица-наполнитель	700.0	0.475080	0.178852	0.0	0.354198	0.470486	0.594725	1.0
Плотность, кг/м3	700.0	0.460015	0.179172	0.0	0.331535	0.456148	0.577349	1.0
модуль упругости, ГПа	700.0	0.446995	0.198017	0.0	0.300911	0.447128	0.578954	1.0
Количество отвердителя, м.%	700.0	0.492974	0.176204	0.0	0.371904	0.493143	0.612012	1.0
Содержание эпоксидных групп,%_2	700.0	0.490718	0.176745	0.0	0.371371	0.488621	0.619951	1.0
Температура вспышки, С_2	700.0	0.489708	0.176965	0.0	0.368951	0.489479	0.608334	1.0
Поверхностная плотность, г/м2	700.0	0.373061	0.216199	0.0	0.206143	0.358842	0.532634	1.0
Прочность при растяжении, МПа	700.0	0.503359	0.177701	0.0	0.383379	0.498630	0.616059	1.0
Потребление смолы, г/м2	700.0	0.508277	0.168013	0.0	0.393195	0.507419	0.619798	1.0
Угол нашивки	700.0	0.495714	0.500339	0.0	0.000000	0.000000	1.000000	1.0
Шаг нашивки	700.0	0.488922	0.185128	0.0	0.364410	0.490795	0.610799	1.0
Плотность нашивки	700.0	0.508781	0.166270	0.0	0.401573	0.507366	0.615928	1.0

Рисунок 17 - Описательная статистика после нормализации

2.2. Разработка обучение и тестирование модели

Для решения поставленной задачи данные были разделены на обучающую и тестовую выборки в соотношении 70/30. Для каждой модели был создан словарь с гиперпараметрами и с помощью поиска по сетке с перекрестной проверкой были найдены лучшие гиперпараметры для каждой модели. Оценка качества работы каждой из моделей выполнялась с помощью вычисления коэффициента детерминации (R^2), среднеквадратичной ошибки (RMSE) и средней абсолютной ошибка (MAE).

Для прогнозирования модуля упругости при растяжении и прочности при растяжении были использованы следующие методы машинного обучения:

- Лассо-регрессия (Lasso);
- Линейная регрессия (LinearRegression);
- Гребневая регрессия (Ridge);
- Регрессионное дерево решений (DecisionTreeRegressor);
- Случайный лес регрессии (RandomForestRegressor);
- Эластичная сеть регрессии (ElasticNetCV);
- Метод опорных векторов для регрессии (SVR);

- Байесовская линейная регрессия (BayesianRidge);
- Ядерная регрессия (KernelRidge).

Коэффициент детерминации, имеющий отрицательные значения близкие к нулю, говорит о том, что результат использования моделей не точнее использования для прогноза среднего значения прогнозируемого параметра.

Наилучший результат для “Модуль упругости при растяжении” с подобранными гиперпараметрами на тренировочной выборке показала модель KernelRidge.

Таблица 2 - Показатели моделей на тренировочной выборке, предсказывающих модуль упругости при растяжении.

	R2	**RMSE**	**MAE**
Lasso	-0.019376	-3.126837	-2.510495
LinearRegression	-0.020283	-3.126736	-2.502399
Ridge	-0.018656	-3.124360	-2.501499
DecisionTreeRegressor	-1.157518	-4.548800	-3.686873
RandomForestRegressor	-0.088072	-3.228611	-2.571452
ElasticNetCV	-0.014429	-3.118477	-2.501795
SVR	-0.039026	-3.153494	-2.514281
BayesianRidge	-0.019112	-3.125729	-2.508368
KernelRidge	-7.406742	-8.908562	-7.108328

Все модели для расчёта “ модуля упругости при растяжении” на тестовой выборке показали неудовлетворительные результаты, коэффициент детерминации у всех моделей показал отрицательное значение. Лучшие показатели на тестовой выборке показывает модель BayesianRidge и ElasticNetCV.

Таблица 3 - Показатели моделей на тестовой выборке, предсказывающих модуль упругости при растяжении.

	R2	RMSE	MAE
Lasso	-0.014721	-3.118528	-2.502596
LinearRegression	-0.020283	-3.126736	-2.502399
Ridge	-0.017370	-3.122490	-2.500862
DecisionTreeRegressor	-0.025911	-3.136856	-2.512126
RandomForestRegressor	-0.030986	-3.143604	-2.517188
ElasticNetCV	-0.013461	-3.117018	-2.498687
SVR	-0.016112	-3.119945	-2.485916
BayesianRidge	-0.013430	-3.116968	-2.499407
KernelRidge	-0.012441	-3.115224	-2.492515

Единственная модель для расчёта “Прочности при растяжении” на тренировочной выборке, у которой коэффициент детерминации стал положительным - стала модель SVR (Метод опорных векторов).

Таблица 4 - Показатели моделей на тренировочной выборке, предсказывающих “Прочности при растяжении”.

	R2	RMSE	MAE
Lasso	-0.013231	-490.990325	-390.660135
LinearRegression	-0.016506	-491.738397	-391.673424
Ridge	-0.015202	-491.443440	-391.275036
DecisionTreeRegressor	-1.050187	-695.606763	-555.861941
RandomForestRegressor	-0.044060	-498.430036	-396.407725
ElasticNetCV	-0.022025	-493.294205	-392.198521

SVR	-0.021077	-493.114512	-390.527482
BayesianRidge	-0.022544	-493.399019	-391.082112
KernelRidge	-0.472360	-591.528506	-471.941020

Все модели для расчёта “Прочности при растяжении” на тестовой выборке показали неудовлетворительные результаты, коэффициент детерминации у всех моделей показал отрицательное значение.

Лучшие показатели на предсказанных значениях показали BayesianRidge и ElasticNetCV.

Таблица 5 - Показатели моделей на тестовой выборке, предсказывающих “Прочности при растяжении”.

	R2	**RMSE**	**MAE**
Lasso	-0.011016	-490.494300	-389.793813
LinearRegression	-0.016506	-491.738397	-391.673424
Ridge	-0.014197	-491.217869	-390.996241
DecisionTreeRegressor	-0.052261	-500.563981	-397.438065
RandomForestRegressor	-0.007580	-489.809585	-389.693131
ElasticNetCV	-0.022941	-493.539259	-391.022774
SVR	0.003436	-486.965371	-385.813515
BayesianRidge	-0.022944	-493.539871	-391.010971
KernelRidge	-0.011806	-490.780486	-389.554120

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

Для решения поставленной задачи по рекомендации значения параметра “Соотношение матрица-наполнитель” были написаны нейросети с помощью MLPRegressor из библиотеки sklearn и с помощью Sequential из библиотеки Keras.

Для работы с нейросетью датасет разделен на новые тестовую и обучающую выборку в соотношении 70/30, с целевым параметром «Соотношение матрица/наполнитель».

Для подбора лучших параметров для MLPRegressor был использован метод GridSearchCV из библиотеки sklearn.

```
# Воспользуемся GridSearchCV из библиотеки sklearn для подбора лучших параметров для MLPRegressor.
param_list = {"hidden_layer_sizes": [(12, 12, 12), (8, 12), (24, 24), (24, 24, 24), (8, 8), (16, 16)],
              "activation": ["identity", "logistic", "tanh", "relu"],
              "solver": ["lbfgs", "sgd", "adam"],
              "alpha": [0.00005, 0.0001, 0.0005, 0.005, 0.05],
              }
model_matrix = MLPRegressor()

grid_search_matrix2 = GridSearchCV(model_matrix, param_list, cv=5, verbose=1, n_jobs=-1)
grid_search_matrix2.fit(X_matrix_train_norm.values, y_matrix_train)
best_params = grid_search_matrix2.best_params_
best_score = grid_search_matrix2.best_score_
print(f'Best params: {best_params}')
print(f'Best score: {best_score}')
```

```
Fitting 5 folds for each of 360 candidates, totalling 1800 fits
Best params: {'activation': 'tanh', 'alpha': 0.005, 'hidden_layer_sizes': (12, 12, 12), 'solver': 'sgd'}
Best score: 0.0019243149088536526
```

Рисунок 18 - Подбор гиперпараметров

Преимущества многослойного перцептрона:

- Возможность изучать нелинейные модели.
- Возможность изучения моделей в режиме реального времени.
- К недостаткам многослойного персептрона (MLP) можно отнести:
- MLP со скрытыми слоями имеют невыпуклую функцию потерь, когда существует более одного локального минимума. Поэтому разные инициализации случайных весов могут привести к разной точности проверки.
- MLP требует настройки ряда гиперпараметров, таких как количество скрытых нейронов, слоев и итераций.
- MLP чувствителен к масштабированию функций.

Архитектура и параметры нейронной сети многослойного персептрона MLPRegressor:

- Последовательная модель нейронной сети.

- Модель состоит из трёх скрытых слоев, в каждом слое по 12 нейронов и выходного слоя с одним нейроном.
- Функция активации слоев выбран гиперболический тангенс (tanh). В качестве оптимизатора нейронной сети используется SGD.
- (стохастический градиентный спуск) с импульсом ускорения = 0.5.
- В нейронной сети используется функция ранней остановки обучения.
- Параметр регуляризации L2 (регуляризация весов), который уменьшает переобучение модели - 0.005.
- Максимальное количество итераций для обучения модели – 500.
- Для валидации будет использовано 30% обучающих данных.

```
# Настраиваем нейросеть по параметрам GridSearchCV
model_matrix3_1 = MLPRegressor(
    hidden_layer_sizes = (12, 12, 12),
    activation = 'tanh',
    solver='sgd',
    alpha = 0.005,
    max_iter=500,
    early_stopping = True,
    validation_fraction = 0.3,
    random_state=42,
    verbose=True
)
```

Рисунок 19 - Настройка нейронной сети

В процессе обучения нейронной сети из библиотеки Keras сработала ранняя остановка обучения на 17-ой эпохе обучения, т.к. в течение 10-ти эпох не наблюдалось улучшения результата потерь на валидационной выборке.

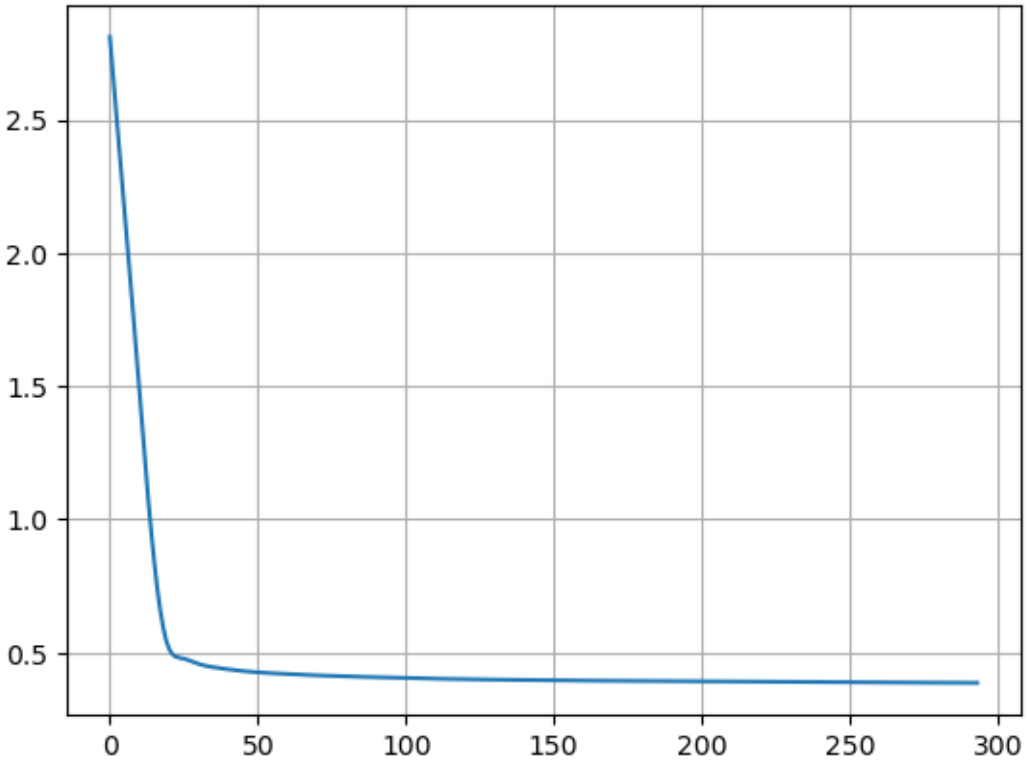

Рисунок 20 - График ошибки

Коэффициент детерминации для MLPRegressor без подобранных параметров: -0.037495.

Среднеквадратичная ошибка для MLPRegressor без подобранных параметров: 0.940855.

График прогнозных данных, полученных с помощью нейронной сети MLPRegressor.

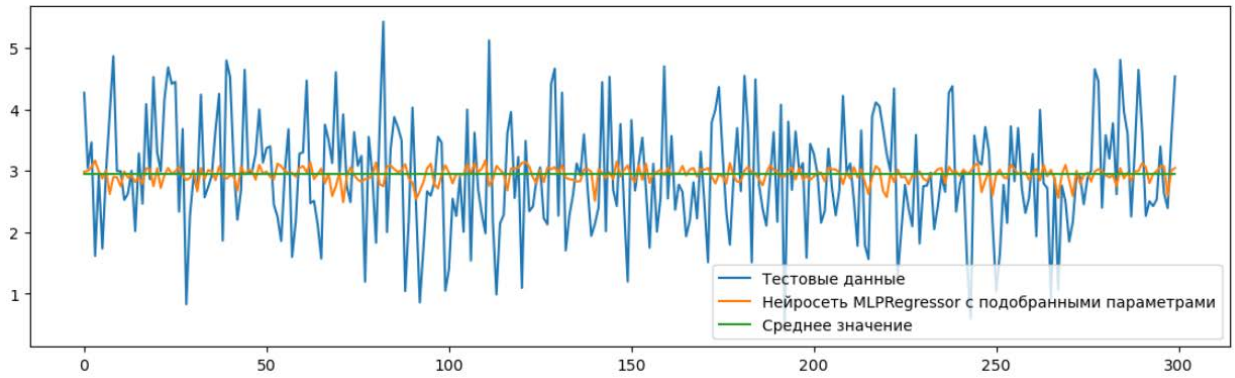

Рисунок 21 - График прогнозных данных

Модель нейронной сети, созданной с помощью MLPRegressor показала неудовлетворительный результат. Коэффициент детерминации, имеющий значение близкое к нулю, говорит о том, что результат использования нейронной сети не точнее использования для прогноза среднего значения прогнозируемого параметра.

Архитектура и параметры нейронной сети из библиотеки Keras:

- Последовательная модель (Sequential) нейронной сети.
- Модель состоит из трёх скрытых слоев (Dense), с количеством нейронов, в которых равно 8, 16, 24 и выходного слоя с одним нейроном, так же в нейронной сети есть три слоя регуляризации (Dropout) со значениями (0.2, 0.3, 0.5).
- Функция активации слоев выбран гиперболический тангенс (tanh).
- В качестве оптимизатора нейронной сети используется SGD (стохастический градиентный спуск) со скоростью обучения 0,005, функцией потерь "среднеквадратическая ошибка" и метрикой оценки качества "среднеквадратичная ошибка".
- В нейронной сети используется функция ранней остановки обучения, если в течение пяти эпох не наблюдается улучшения потерь на валидационной выборке.
- Количество эпох обучения равно 100
- Для валидации будет использовано 30% обучающих данных.

```
# Создадим последовательную модель с слоем Dropout.
model_matrix_tf3 = tf.keras.Sequential([X_matrix_train_n_k, layers.Dense(8, activation='tanh'),
                                                            layers.Dropout(0.2),
                                                            layers.Dense(16, activation='tanh'),
                                                            layers.Dropout(0.3),
                                                            layers.Dense(24, activation='tanh'),
                                                            layers.Dropout(0.5),
                                                            layers.Dense(1)
                                        ])

# Компиляция модели с оптимизатором, функцией потерь и метриками
model_matrix_tf3.compile(optimizer = tf.keras.optimizers.SGD(0.005), loss = 'mean_squared_error', metrics = [tf.keras.metrics.Roo


# Создание callback-функции для остановки обучения, если значение потерь на валидационных данных не улучшится в течение 5 эпох
early_stopping_callback = keras.callbacks.EarlyStopping(monitor='val_loss', patience=5)

# Посмотрим на архитектуру модели
model_matrix_tf3.summary()
```

Рисунок 22 - Нейронная сеть

```
%%time
#Обучение нейросети
history = model_matrix_tf3.fit(
  X_matrix_test,
  y_matrix_train,
  epochs=100,
  validation_split=0.3,
  verbose=1,
  callbacks=[early_stopping_callback])
```

Рисунок 23. Обучение нейронной сети

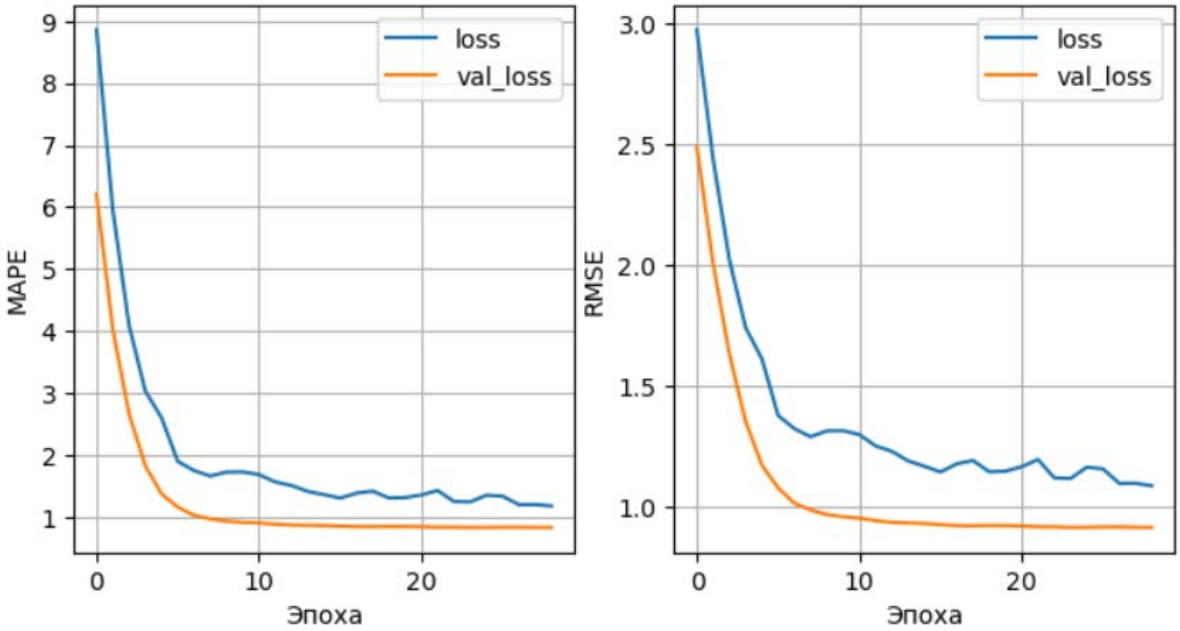

Рисунок 24 - График ошибки

Коэффициент детерминации для последовательной нейросети со слоем Dropout: -0.048021

Среднеквадратичная ошибка для последовательной нейросети со слоем Dropout: 0.945616

График прогнозных значений, полученных с помощью последовательной нейронной сети из библиотеки Keras.

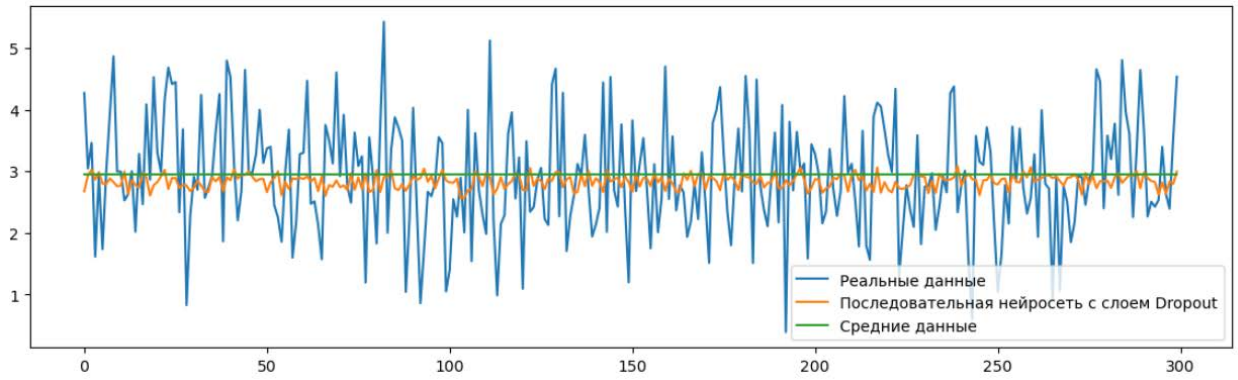

Рисунок 25 - График прогнозных значений

Модель последовательной нейронной сети, созданной с помощью библиотеки Keras показала неудовлетворительный результат. Коэффициент детерминации, имеющий значение близкое к нулю, говорит о том, что результат использования нейронной сети не точнее использования для прогноза среднего значения прогнозируемого параметра.

Таблица 6 - Результаты работы нейронных сетей для предсказания “Соотношение матрица-наполнитель”

	R2	**RMSE**
MLPRegressor без подобранных гиперпараметров	-0.037495	0.940855
MLPRegressor с подобранными гиперпараметрами	-0.019444	0.932634
Последовательная нейросеть (Keras)	-0.064893	0.953197
Последовательная нейросеть (Keras) с callback	-0.085778	0.962499
Последовательная нейросеть (Keras) с Dropout	-0.048021	0.945616

2.4 Создание удаленного репозитория и загрузка результатов работы на него

Ссылка на репозиторий в GitHub:

https://github.com/Zefir07/BMTSU_DS_AndreevaEA_Komposits.git

Заключение

В данной работе не удалось разработать эффективную модель прогнозирования конечных свойств новых композиционных материалов на основе данных об их составе и структуре, точность предсказанных значений практически не превосходило среднее значения.

Тем не менее получен существенный практический опыт по анализу, визуализации и предобработке данных, созданию нейронных сетей и моделей машинного обучения.

Список использованной литературы

1) Композиционные материалы: Справочник /Под. ред. В.В. Васильева, Ю.М.Тарнопольского. –М.: Машиностроение, 1990. –512 с.

2) Библиотека Keras - инструмент глубокого обучения. Реализация нейронных сетей с помощью библиотек Theano и TensorFlow / пер. с англ. Слинкин А. А. - М.: ДМК Пресс, 2018. - 294 с.

3) Силен Дэви, Мейсман Арно, Али Мохамед. Основы Data Science и Big Data. Python и наука о данных. – СПб.: Питер, 2017. – 336 с.: ил.

4) Платформа scikit-learn [Электронный ресурс]: – Режим доступа: https://scikit-learn.org/stable/ (дата обращения: 15.04.2023).

5) Библиотека Seaborn- Режим доступа: https://seaborn.pydata.org/. (дата обращения 16.04.2023)

6) Язык программирования Python- Режим доступа: https://www.python.org/. (дата обращения 29.03.2023)

7) Библиотека Pandas – Режим доступа: https://pandas.pydata.org/ (дата обращения 31.03.2023)

8) Библиотека Sklearn – Режим доступа: https://scikit-learn.org/stable/ (дата обращения 15.04.2023)

9) Библиотека Pandas- Режим доступа: https://pandas.pydata.org/. (дата обращения 30.03.2023)

10) Библиотека Matplotlib- Режим доступа: https://matplotlib.org/. (дата обращения 01.04.2023)

11) Библиотека Tensorflow: Режим доступа: https://www.tensorflow.org/. (дата обращения 05.04.2023)

12) https://habr.com/ru/companies/otus/articles/4429 - Режим доступа: (дата обращения 19.04.2023)