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»

Прогнозирование конечных свойств новых материалов (композиционных материалов)

Слушатель Костромина Ольга Сергеевна

Москва
2022

Содержание

Введение

Композиционные материалы – это искусственно созданные материалы, состоящие из нескольких других с четкой границей между ними. Композиты обладают теми свойствами, которые не наблюдаются у компонентов по отдельности. При этом композиты являются монолитным материалом, т. е. компоненты материала неотделимы друг от друга без разрушения конструкции в целом. Яркий пример композита – железобетон. Бетон прекрасно сопротивляется сжатию, но плохо растяжению. Стальная арматура внутри бетона компенсирует его неспособность сопротивляться сжатию, формируя тем самым новые, уникальные свойства. Современные композиты изготавливаются из других материалов: полимеры, керамика, стеклянные и углеродные волокна, но данный принцип сохраняется. У такого подхода есть и недостаток: даже если мы знаем характеристики исходных компонентов, определить характеристики композита, состоящего из этих компонентов, достаточно проблематично. Для решения этой проблемы есть два пути: физические испытания образцов материалов или прогнозирование характеристик. Суть прогнозирования заключается в симуляции представительного элемента объема композита, на основе данных о характеристиках входящих компонентов (связующего и армирующего компонента). Созданные прогнозные модели помогут сократить количество проводимых испытаний, а также пополнить базу данных материалов возможными новыми характеристиками материалов и цифровыми двойниками новых композитов.

1. Аналитическая часть

1.1 Постановка задачи

Объектом исследования является кейс с данными о начальных свойствах компонентов композиционных материалов (количество связующего, наполнителя, температурный режим отверждения и т.д.), основанный на реальных производственных задачах Центра НТИ «Цифровое материаловедение: новые материалы и вещества» (структурное подразделение МГТУ им. Н.Э. Баумана).

Цель работы: спрогнозировать ряд конечных свойств получаемых композиционных материалов.

Необходимо решить следующие задачи.

1) Изучить теоретические основы и методы решения поставленной задачи.

2) Провести разведочный анализ предложенных данных. Необходимо нарисовать гистограммы распределения каждой из переменной, диаграммы ящика с усами, попарные графики рассеяния точек. Необходимо также для каждой колонке получить среднее, медианное значение, провести анализ и исключение выбросов, проверить наличие пропусков.

3) Провести предобработку данных (удаление шумов, нормализация и т.д.).

4) Обучить нескольких моделей для прогноза модуля упругости при растяжении и прочности при растяжении. При построении модели необходимо 30% данных оставить на тестирование модели, на остальных происходит 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 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. Оформить файл README.

1.2 Описание используемых методов

Лассо (Lasso()) — это линейная модель, которая оценивает разреженные коэффициенты. Это полезно в некоторых контекстах из-за своей тенденции отдавать предпочтение решениям с меньшим количеством ненулевых коэффициентов, эффективно уменьшая количество функций, от которых зависит данное решение. По этой причине лассо и его варианты являются фундаментальными для области сжатого зондирования. При определенных условиях он может восстановить точный набор ненулевых коэффициентов. Математически он состоит из линейной модели с добавленным членом регуляризации.

Метод случайного леса (RandomForestRegressor()) – универсальный алгоритм машинного обучения, суть которого состоит в использовании ансамбля решающих деревьев. Само по себе решающее дерево предоставляет крайне невысокое качество классификации, но из-за большого их количества результат значительно улучшается. Цель ансамблевых методов – объединить прогнозы нескольких базовых оценок, построенных с заданным алгоритмом обучения, чтобы улучшить обобщаемость / надежность по сравнению с одной оценкой. В методе случайного леса каждое дерево в ансамбле строится из выборки, взятой с заменой (то есть выборкой начальной загрузки) из обучающей выборки. Кроме того, при разбиении каждого узла во время построения дерева наилучшее разбиение находится либо по всем входным характеристикам, либо по случайному подмножеству размера max_features.

Назначение этих двух источников случайности – уменьшить дисперсию оценки леса.

Градиентный бустинг (GradientBoostingRegressor()) представляет собой ансамбль деревьев решений. В основе данного алгоритма лежит итеративное обучение деревьев решений с целью минимизировать функцию потерь. Благодаря особенностям деревьев решений градиентный бустинг способен работать с категориальными признаками, справляться с нелинейностями.

Принцип, лежащий в основе метода k-ближайших соседей (KNeighborsRegressor()), состоит в том, чтобы найти предопределенное количество обучающих выборок, ближайших по расстоянию к новой точке, и предсказать метку по ним. Количество выборок может быть заданной пользователем константой (обучение k-ближайшего соседа) или изменяться в зависимости от локальной плотности точек (обучение соседей на основе радиуса). Расстояние, как правило, может быть любой метрической мерой: стандартное евклидово расстояние является наиболее распространенным выбором.

Основная идея метода опорных векторов (SVR()) – перевод исходных векторов в пространство более высокой размерности и поиск разделяющей гиперплоскости с наибольшим зазором в этом пространстве. Две параллельных гиперплоскости строятся по обеим сторонам гиперплоскости, разделяющей классы. Разделяющей гиперплоскостью будет гиперплоскость, создающая наибольшее расстояние до двух параллельных гиперплоскостей. Алгоритм основан на допущении, что чем больше разница или расстояние между этими параллельными гиперплоскостями, тем меньше будет средняя ошибка классификатора.

1.3 Разведочный анализ данных

Разведочный анализ данных – это предварительное исследование исходных данных, в ходе которого вычисляется статистика и ищутся в данных аномалии, шаблоны или взаимосвязи. Короче говоря, это попытка выяснить, что нам

могут сказать данные. Выводы могут быть интересными сами по себе, или они могут способствовать выбору модели, помогая решить, какие признаки мы будем использовать.

Разведочный анализ данных был проведен над датасетом (рисунок 1), полученным путем объединения двух датасетов по индексу (тип объединения INNER). Описательная статистика для каждой колонки исследовательского датасета, включающая в себя информацию о числе значений, среднем значении, стандартном отклонении, минимальном значении, верхнем значении первого квартиля, медианном значении, верхнем значении третьего квартиля, максимальном значении, представлена на рисунке 2.

	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, град	Шаг нашивки	Плотность нашивки
0	1.857143	2030.000000	738.736842	30.000000	22.267857	100.000000	210.000000	70.000000	3000.000000	220.000000	0.0	4.000000	57.000000
1	1.857143	2030.000000	738.736842	50.000000	23.750000	284.615385	210.000000	70.000000	3000.000000	220.000000	0.0	4.000000	60.000000
2	1.857143	2030.000000	738.736842	49.900000	33.000000	284.615385	210.000000	70.000000	3000.000000	220.000000	0.0	4.000000	70.000000
3	1.857143	2030.000000	738.736842	129.000000	21.250000	300.000000	210.000000	70.000000	3000.000000	220.000000	0.0	5.000000	47.000000
4	2.771331	2030.000000	753.000000	111.860000	22.267857	284.615385	210.000000	70.000000	3000.000000	220.000000	0.0	5.000000	57.000000
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1018	2.271346	1952.087902	912.855545	86.992183	20.123249	324.774576	209.198700	73.090961	2387.292495	125.007669	90.0	9.076380	47.019770
1019	3.444022	2050.089171	444.732634	145.981978	19.599769	254.215401	350.660830	72.920827	2360.392784	117.730099	90.0	10.565614	53.750790
1020	3.280604	1972.372865	416.836524	110.533477	23.957502	248.423047	740.142791	74.734344	2662.906040	236.606764	90.0	4.161154	67.629684
1021	3.705351	2066.799773	741.475517	141.397963	19.246945	275.779840	641.468152	74.042708	2071.715856	197.126067	90.0	6.313201	58.261074
1022	3.808020	1890.413468	417.316232	129.183416	27.474763	300.952708	758.747882	74.309704	2856.328932	194.754342	90.0	6.078902	77.434468

1023 rows × 13 columns

Рисунок 1. Исследовательский датасет

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

Рисунок 2. Описательная статистика исходных данных

Построим гистограммы распределения и функцию плотности распределения по каждой переменной (рисунок 3). Видим, что все переменные, кроме угла нашивки, имеют распределение, близкое к нормальному.

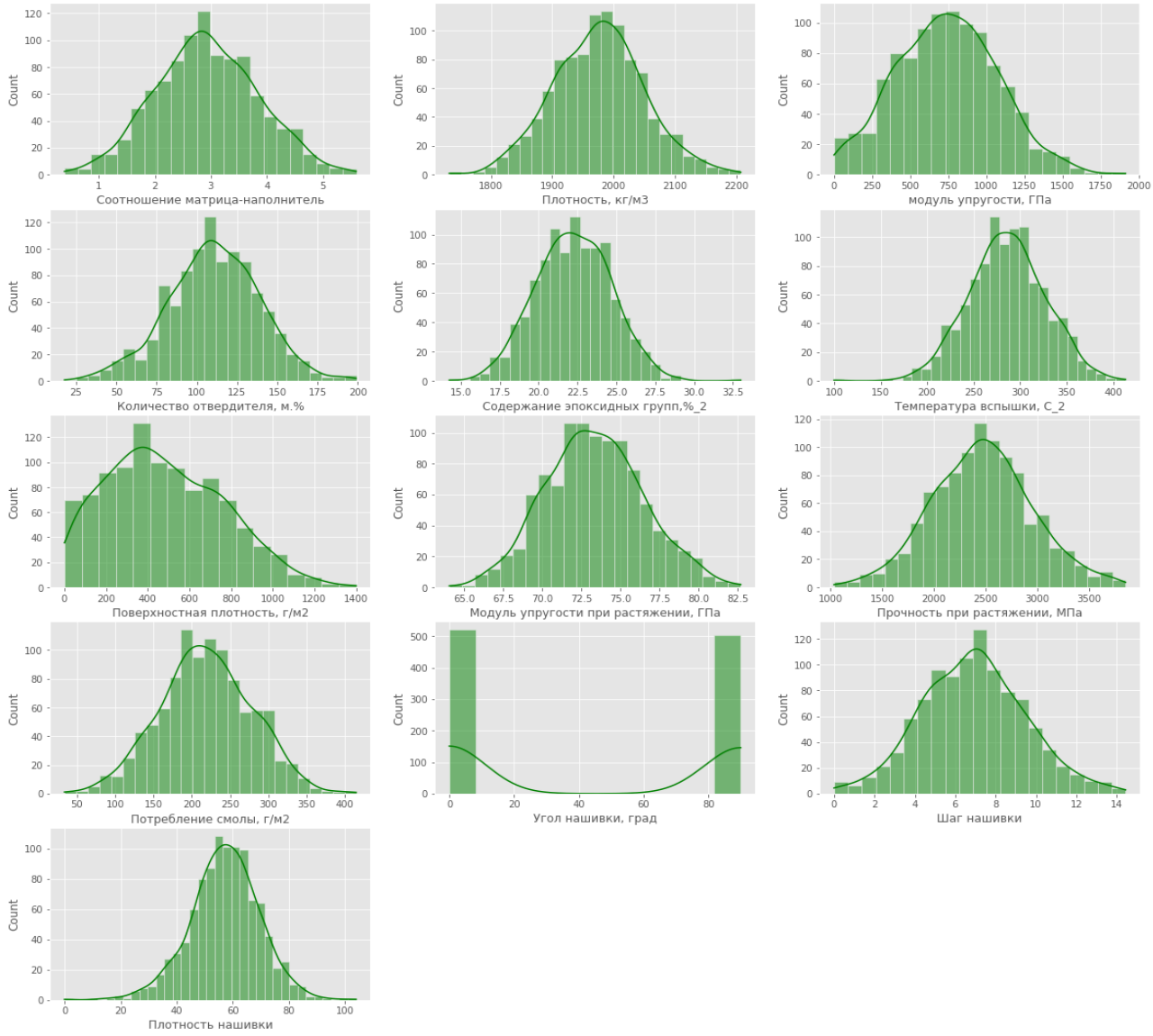


Рисунок 3. Гистограммы распределения

Для обнаружения выбросов в исходных данных воспользуемся самым простым визуальным способом – построим диаграммы размаха, или «ящики с усами» (рисунок 4). Скучковавшиеся окружности на этих графиках – это и есть выбросы, избавляются от которых с помощью квантилей.

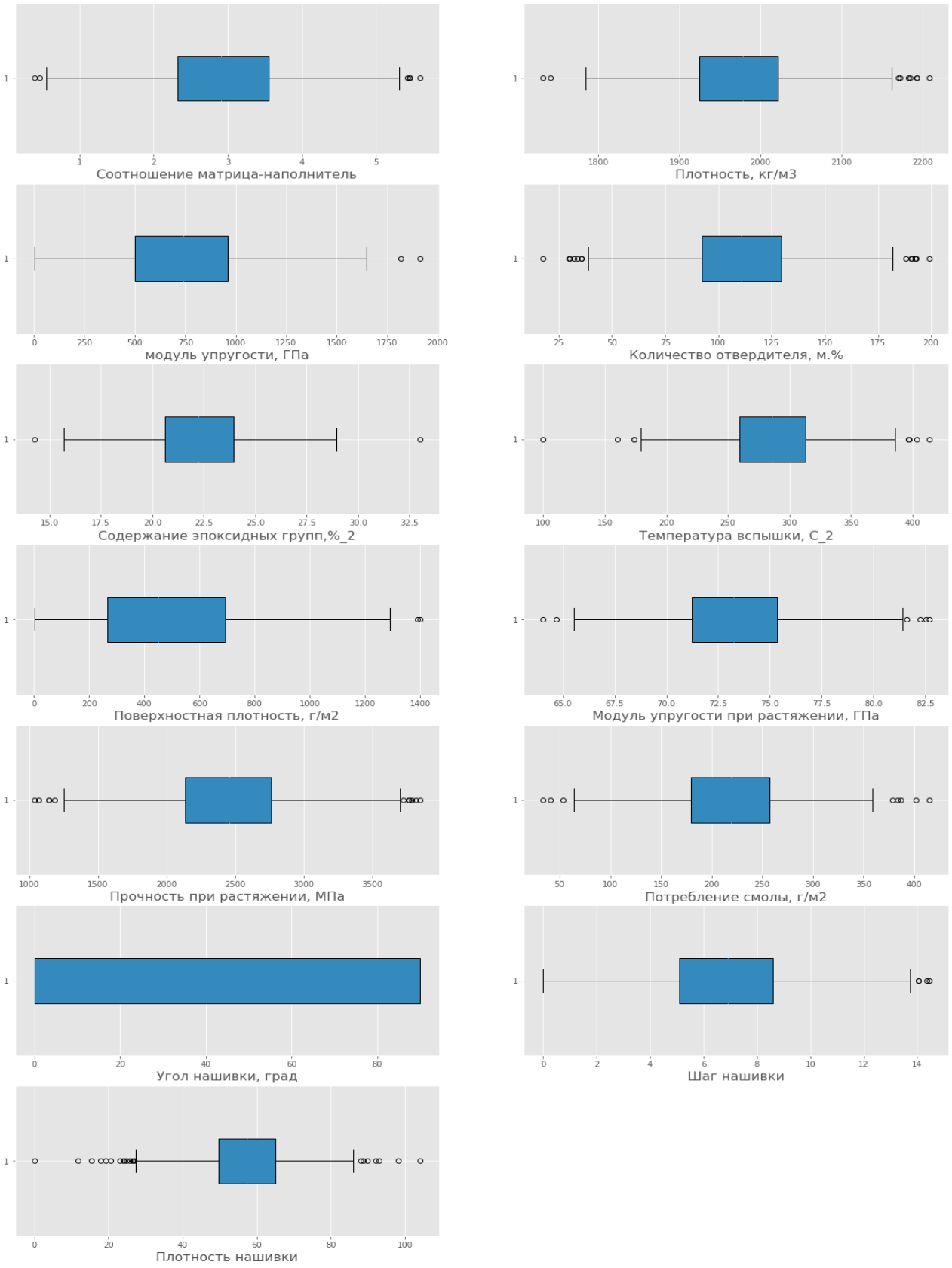


Рисунок 4. «Ящики с усами»

Информацию о том, как связаны между собой переменные исследуемого датасета, даст визуализация матрицы рассеяний (рисунок 5) и матрицы корреляций (рисунок 6). Матрица рассеяний показывает все попарные диаграммы рассеяния и позволяет быстро получить грубое представление о том, какие переменные коррелируют.

Посмотрим на матрицу корреляций, в которой элемент в i-ой строке и j-ом столбце означает корреляцию между i-ой и j-ой переменными. Обнаруживается близкая к нулю попарная корреляция между переменными, что свидетельствует о нелинейной связи между ними.

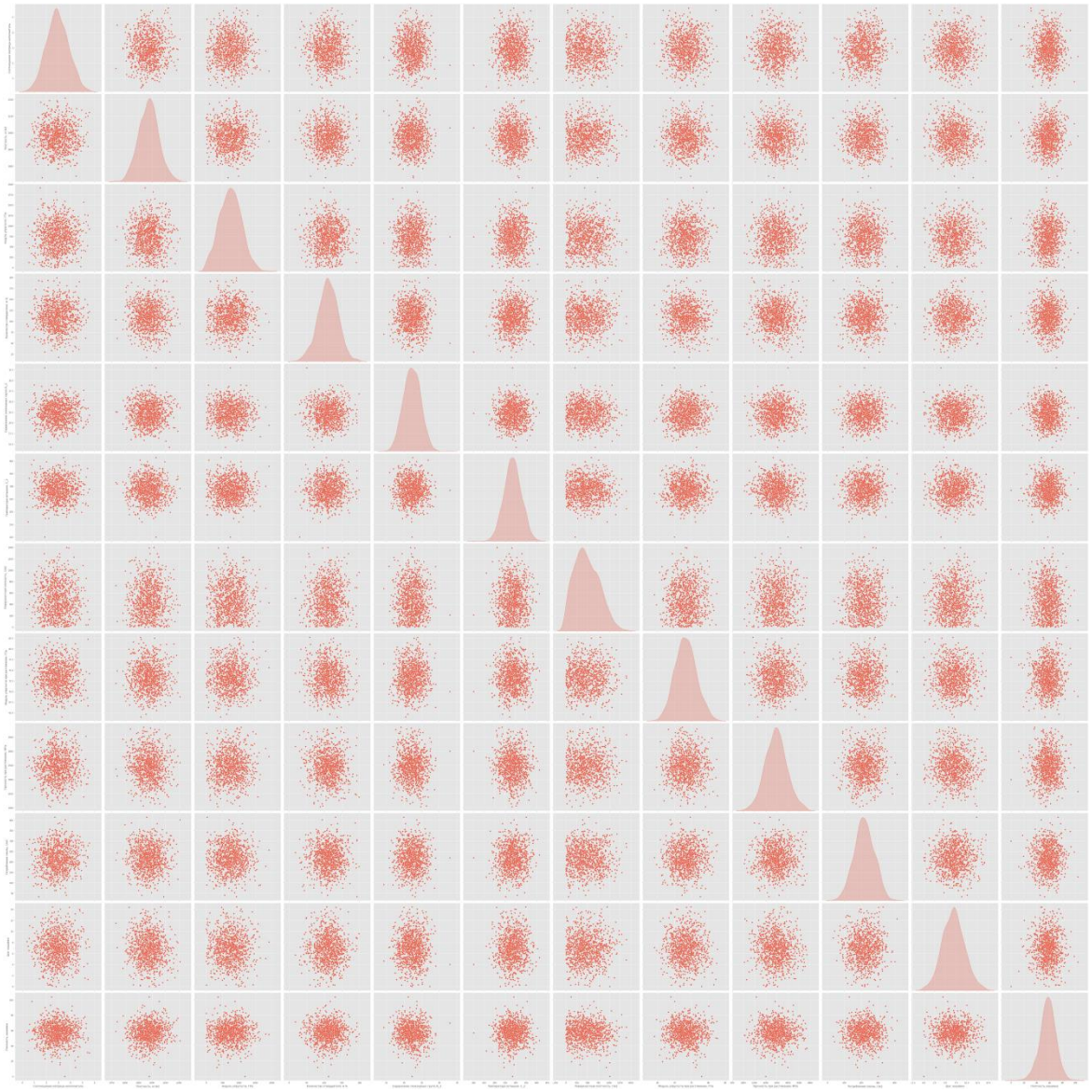


Рисунок 5. Матрица рассеяний

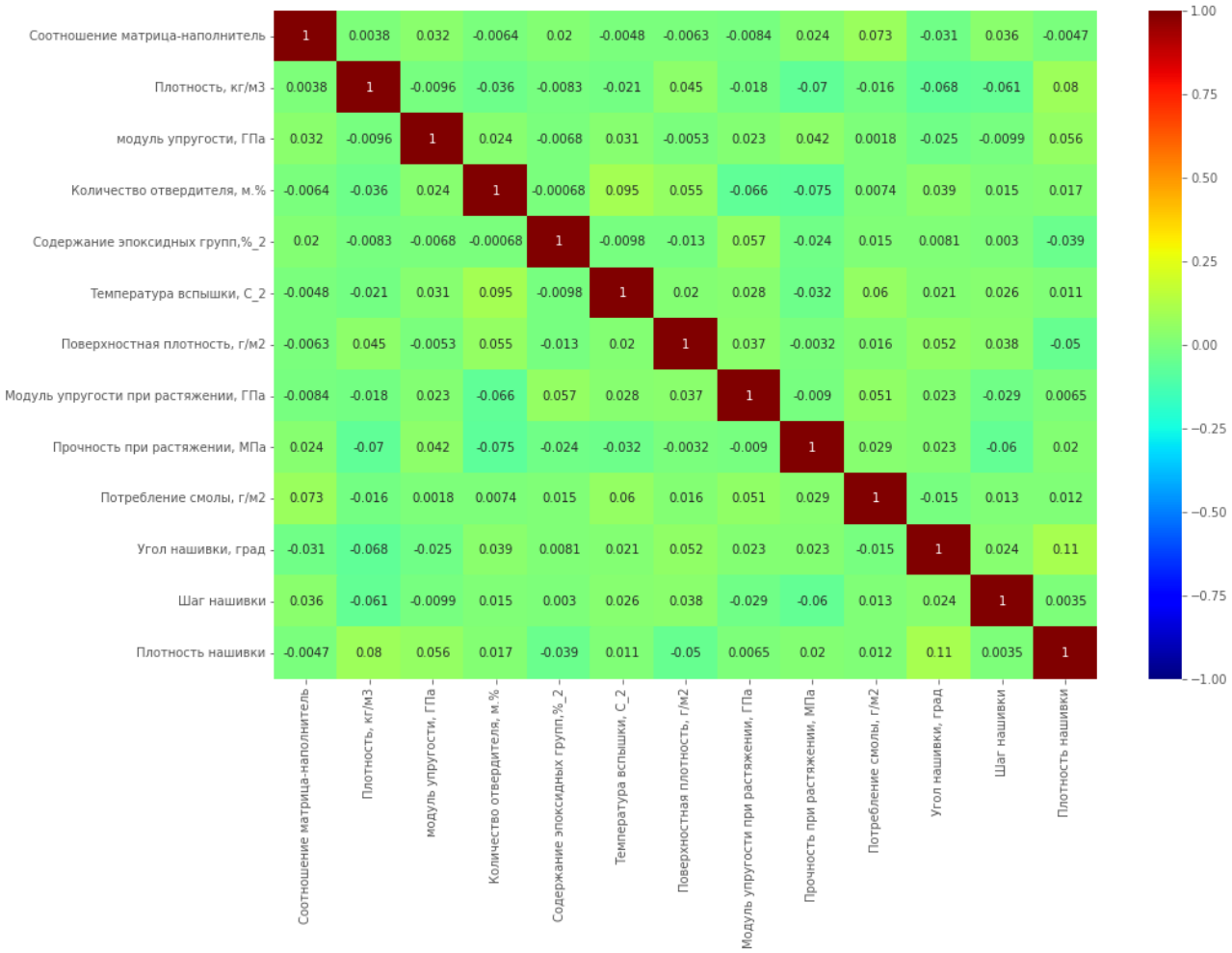


Рисунок 6. Отображение взаимосвязей между переменными с помощью тепловой карты

2. Практическая часть

2.1 Предобработка данных

Предобработка данных состоит из трех этапов: анализ данных на пропуски и дубликаты, удаление выбросов и нормализация данных.

Пропусков, как и дубликатов в нашем датасете не обнаружено.

Удаление выбросов происходит на основе межквартильного расстояния.

```
Соотношение матрица-наполнитель          6
Плотность, кг/м3                         9
модуль упругости, ГПа                    2
Количество отвердителя, м.%              14
Содержание эпоксидных групп,%_2          2
Температура вспышки, С_2                 8
Поверхностная плотность, г/м2            2
Модуль упругости при растяжении, ГПа     6
Прочность при растяжении, МПа            11
Потребление смолы, г/м2                  8
Угол нашивки, град                       0
Шаг нашивки                              4
Плотность нашивки                        21
dtype: int64
```

Рисунок 7. Количество выбросов в каждом столбце

После удаления выбросов построим «ящики с усами» (рисунок 8). Убеждаемся в том, что выбросы отсутствуют.

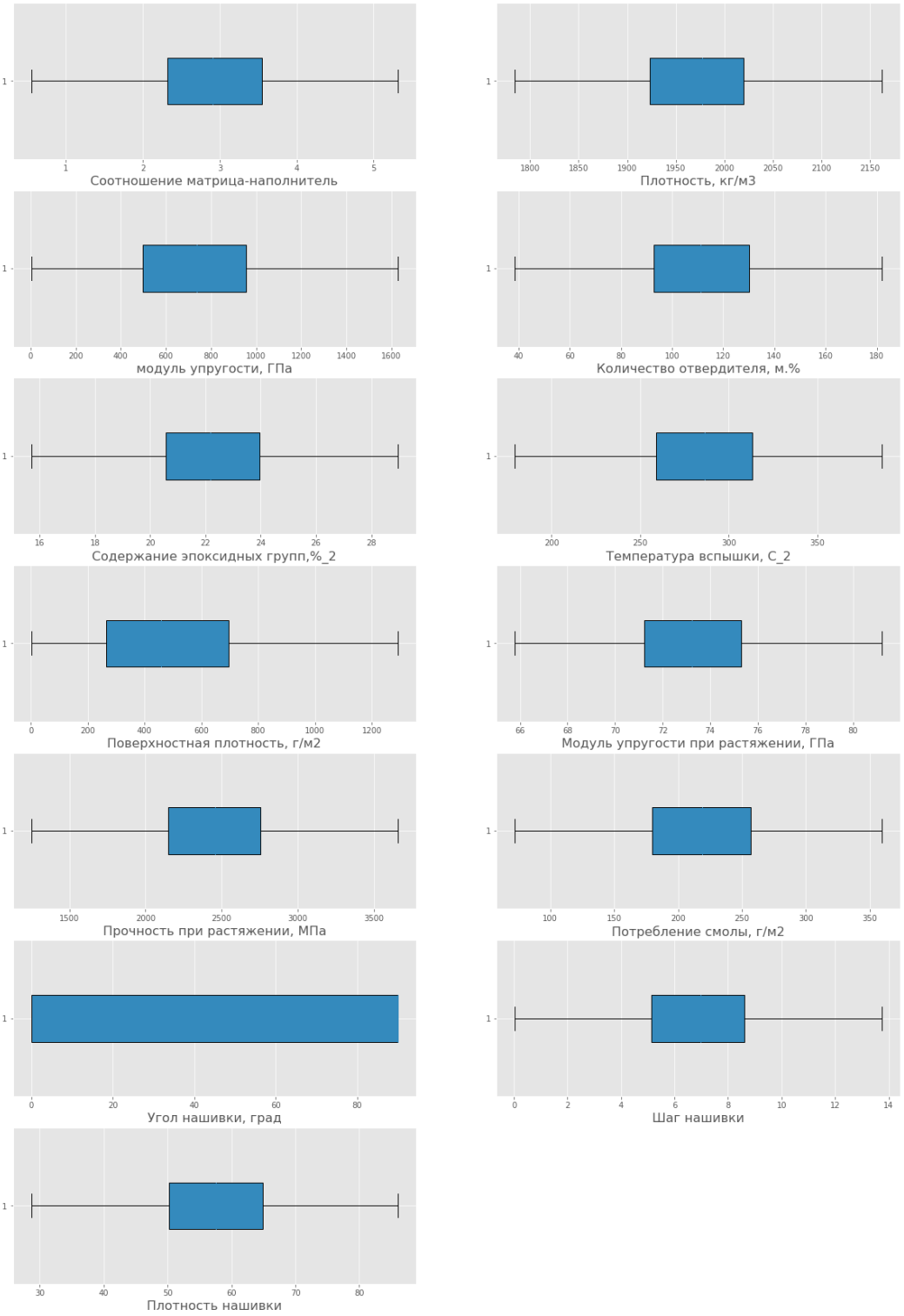


Рисунок 8. «Ящики с усами» после удаления выбросов

Далее проводим нормализацию обработанного датасета с помощью метода MinMaxScaler. Результат нормализации представлен на рисунке 9. Теперь у каждого признака минимальное значение равно 0, а максимальное 1, о чем свидетельствует описательная статистика нормализованного датасета (рисунок 10).

	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, град	Шаг нашивки	Плотность нашивки
0	0.274768	0.651097	0.452951	0.079153	0.607435	0.509164	0.162230	0.272962	0.727777	0.514688	0.0	0.289334	0.546433
1	0.274768	0.651097	0.452951	0.630983	0.418887	0.583596	0.162230	0.272962	0.727777	0.514688	0.0	0.362355	0.319758
2	0.466552	0.651097	0.461725	0.511257	0.495653	0.509164	0.162230	0.272962	0.727777	0.514688	0.0	0.362355	0.494123
3	0.465836	0.571539	0.458649	0.511257	0.495653	0.509164	0.162230	0.272962	0.727777	0.514688	0.0	0.362355	0.546433
4	0.424236	0.332865	0.494944	0.511257	0.495653	0.509164	0.162230	0.272962	0.727777	0.514688	0.0	0.362355	0.720799
...	...	...	...	...	...	...	...	...	...	...	...	...	...
917	0.361662	0.444480	0.560064	0.337550	0.333908	0.703458	0.161609	0.473553	0.472912	0.183151	1.0	0.660014	0.320103
918	0.607674	0.704373	0.272088	0.749605	0.294428	0.362087	0.271207	0.462512	0.461722	0.157752	1.0	0.768759	0.437468
919	0.573391	0.498274	0.254927	0.501991	0.623085	0.334063	0.572959	0.580201	0.587558	0.572648	1.0	0.301102	0.679468
920	0.662497	0.748688	0.454635	0.717585	0.267818	0.466417	0.496511	0.535317	0.341643	0.434855	1.0	0.458245	0.516112
921	0.684036	0.280923	0.255222	0.632264	0.888354	0.588206	0.587373	0.552644	0.668015	0.426577	1.0	0.441137	0.850430

922 rows × 13 columns

Рисунок 10. Нормализованный с помощью MinMaxScaler датасет

	count	mean	std	min	25%	50%	75%	max
Соотношение матрица-наполнитель	922.0	0.499412	0.187858	0.0	0.371909	0.495189	0.629774	1.0
Плотность, кг/м3	922.0	0.502904	0.188395	0.0	0.368184	0.511396	0.624719	1.0
модуль упругости, ГПа	922.0	0.451341	0.201534	0.0	0.305188	0.451377	0.587193	1.0
Количество отвердителя, м.%	922.0	0.506200	0.186876	0.0	0.378514	0.506382	0.638735	1.0
Содержание эпоксидных групп,%_2	922.0	0.490578	0.180548	0.0	0.366571	0.488852	0.623046	1.0
Температура вспышки, С_2	922.0	0.516739	0.190721	0.0	0.386228	0.516931	0.646553	1.0
Поверхностная плотность, г/м2	922.0	0.373295	0.217269	0.0	0.204335	0.354161	0.538397	1.0
Модуль упругости при растяжении, ГПа	922.0	0.487343	0.196366	0.0	0.353512	0.483718	0.617568	1.0
Прочность при растяжении, МПа	922.0	0.503776	0.188668	0.0	0.373447	0.501481	0.624299	1.0
Потребление смолы, г/м2	922.0	0.507876	0.199418	0.0	0.374647	0.510143	0.642511	1.0
Угол нашивки, град	922.0	0.510846	0.500154	0.0	0.000000	1.000000	1.000000	1.0
Шаг нашивки	922.0	0.503426	0.183587	0.0	0.372844	0.506414	0.626112	1.0
Плотность нашивки	922.0	0.503938	0.193933	0.0	0.376869	0.504310	0.630842	1.0

Рисунок 11. Описательная статистика нормализованного датасета

2.2 Разработка и обучение модели

Разработка и обучение моделей машинного обучения производится для двух выходных свойств композиционного материала: модуль упругости при растяжении и прочность при растяжении.

Прежде всего, делим нормализованный датасет на входные и выходные данные. Затем разделяем выходные данные на обучающую и тестовую выборки (в соотношении 70:30).

Для свойства модуля упругости при растяжении разработаны и обучены следующие три модели:

- модель на основе градиентного бустинга (GradientBoostingRegressor());

- случайный лес (RandomForestRegressor());

- линейная модель Лассо (Lasso()).

Для свойства прочности при растяжении разработаны и обучены следующие четыре модели:

- линейная модель Лассо (Lasso());

- модель k-ближайших соседей (KNeighborsRegressor());

- модель на основе метода опорных векторов (SVR());

- модель на основе градиентного бустинга (GradientBoostingRegressor()).

Для каждой модели и каждого признака задавалась сетка гиперпараметров, по которым происходила оптимизация модели. Оптимизация подбора гиперпараметров модели производилась с помощью выбора по сетке и перекрестной проверки. Найденные оптимальные гиперпараметры подставлялись в модель, и производилось обучение модели на тренировочных данных.

Настройки гиперпараметров производились случайным поиском с 10-блочной перекрёстной проверкой.

1. Задаём сетку гиперпараметров.
2. Случайно выбираем комбинацию гиперпараметров (RandomizedSearchCV из Scikit-Learn)
3. Создаём модель с использованием этой комбинации.
4. Оцениваем результат работы модели с помощью k-блочной перекрёстной проверки (GridSearchCV из Scikit-Learn).
5. Решаем, какие гиперпараметры дают лучший результат.

2.3 Тестирование модели

После обучения моделей была проведена оценка точности этих моделей на обучающей и тестовых выборках. В качестве параметра оценки модели была

выбрана средняя абсолютная ошибка (MAE). Для наибольшей наглядности результатов прогнозирования модели, были построены диаграммы рассеяния тестовых данных (реальных значений) и спрогнозированных данных, а также графики плотности спрогнозированных и реальных значений и гистограммы погрешности.

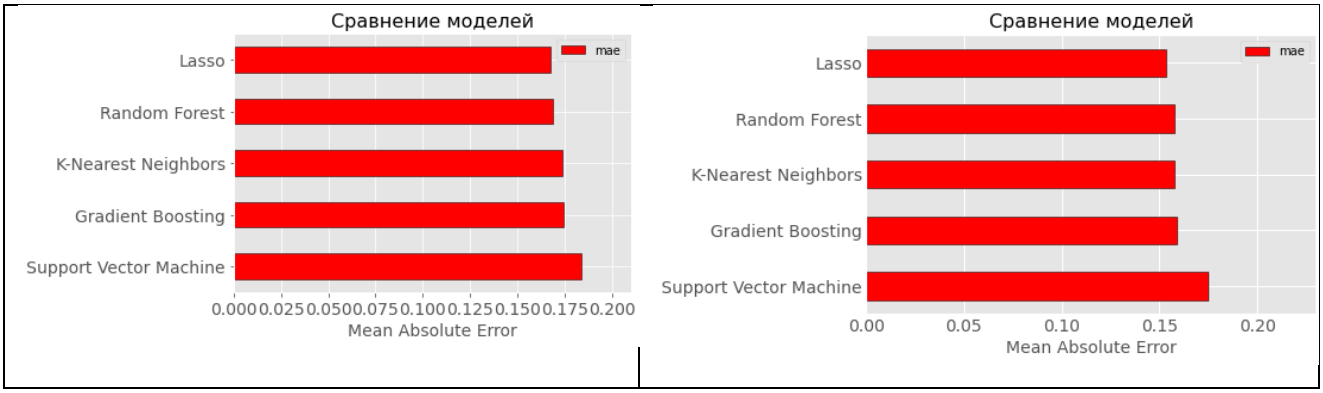
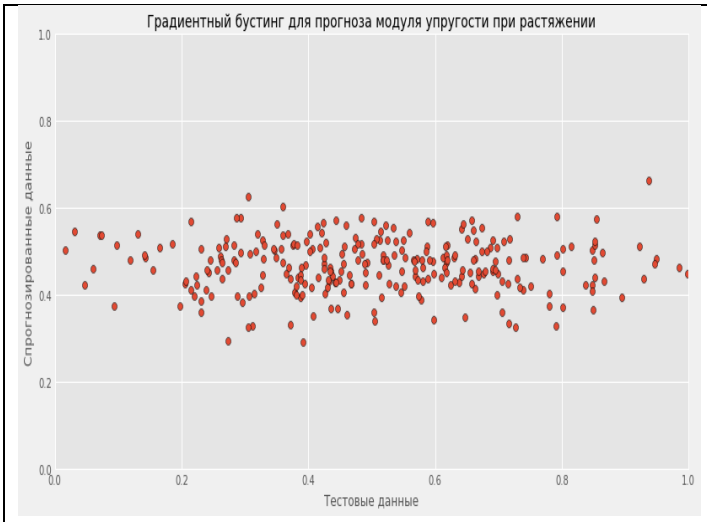


Рисунок 12. Сравнение оценок моделей со стандартными параметрами для прогнозирования свойств модуля упругости при растяжении (слева) и прочности при растяжении (справа)

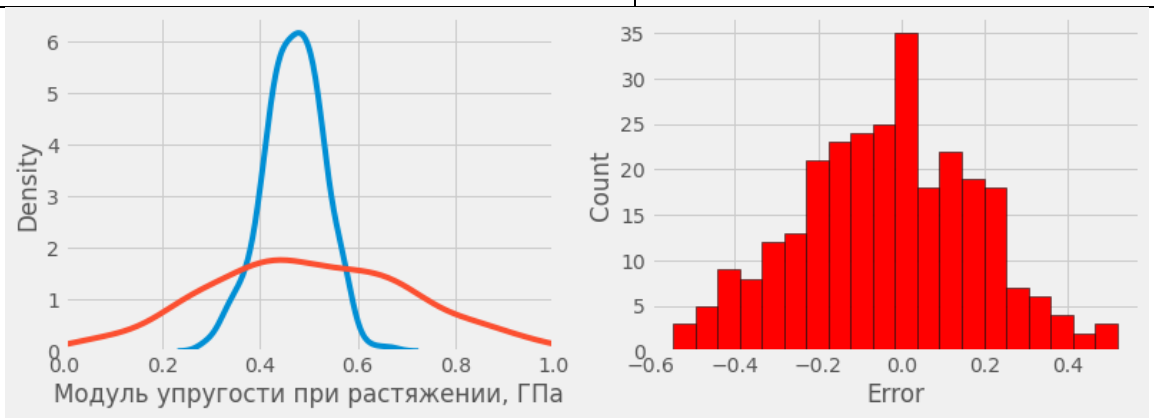
Результаты работы модели на основе градиентного бустинга для прогнозирования свойства модуля упругости при растяжении представлены ниже.

```
[ ]  # Модель с параметрами по умолчанию
     default_model_gbr1 = GradientBoostingRegressor(random_state = 42).fit(x_train1, y_train1)
     default_model_gbr1

     GradientBoostingRegressor(random_state=42)

[ ]  # Настроенная наилучшая модель
     final_model_gbr1 = grid_search_gbr1.best_estimator_
     final_model_gbr1

     GradientBoostingRegressor(loss='lad', max_depth=15, max_features='auto',
                               min_samples_split=10, random_state=42)

[ ]  default_pred_gbr1 = default_model_gbr1.predict(x_test1)
     final_pred_gbr1 = final_model_gbr1.predict(x_test1)

     print('Средняя абсолютная ошибка модели со стандартными параметрами на тестовом наборе: MAE = %0.4f.' % mae(y_test1, default_pred_gbr1))
     print('Средняя абсолютная ошибка настроенной модели на тестовом наборе: MAE = %0.4f.' % mae(y_test1, final_pred_gbr1))

     Средняя абсолютная ошибка модели со стандартными параметрами на тестовом наборе: MAE = 0.1747.
     Средняя абсолютная ошибка настроенной модели на тестовом наборе: MAE = 0.1728.
```



	Тестовые данные	Спрогнозированные данные
319	0.387108	0.395278
377	0.742719	0.487172
538	0.364855	0.448914
296	0.435807	0.456200
531	0.468100	0.426032
...	...	...
420	0.359362	0.475689
133	0.446885	0.369484
490	0.609638	0.440140
558	0.372351	0.439003
363	0.392277	0.291119

277 rows × 2 columns



Результаты работы модели случайный лес для прогнозирования свойства модуля упругости при растяжении представлены ниже.

```
# Модель с параметрами по умолчанию
default_model_rfr1 = RandomForestRegressor().fit(x_train1, y_train1)
default_model_rfr1
```

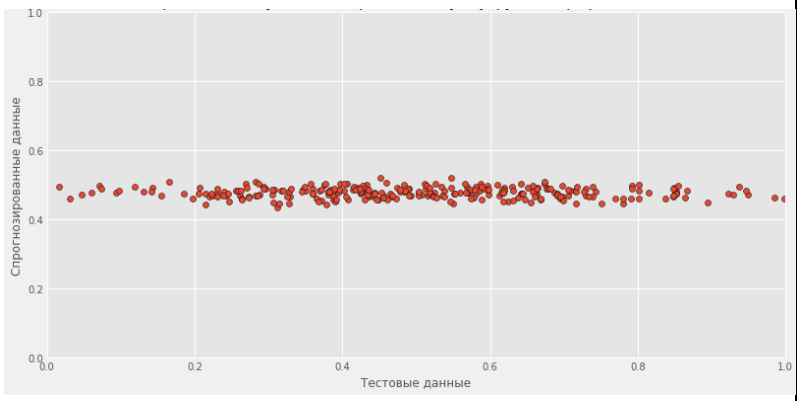
```
RandomForestRegressor()
```

```
# Настроенная наилучшая модель
final_model_rfr1 = grid_search_rfr1.best_estimator_
final_model_rfr1
```

```
RandomForestRegressor(criterion='mae', max_depth=4, max_features='log2',
                      min_samples_leaf=8, min_samples_split=4, n_estimators=300,
                      random_state=42)
```

```
default_pred_rfr1 = default_model_rfr1.predict(x_test1)
final_pred_rfr1 = final_model_rfr1.predict(x_test1)

print('Средняя абсолютная ошибка модели со стандартными параметрами на тестовом наборе: MAE = %0.4f.' % mae(y_test1, default_pred_rfr1))
print('Средняя абсолютная ошибка настроенной модели на тестовом наборе: MAE = %0.4f.' % mae(y_test1, final_pred_rfr1))
```

```
Средняя абсолютная ошибка модели со стандартными параметрами на тестовом наборе: MAE = 0.1677.
Средняя абсолютная ошибка настроенной модели на тестовом наборе: MAE = 0.1677.
```



	Тестовые данные	Спрогнозированные данные
319	0.387108	0.459092
377	0.742719	0.480743
538	0.364855	0.461990
296	0.435807	0.465367
531	0.468100	0.467930
...	...	...
420	0.359362	0.473950
133	0.446885	0.459269
490	0.609638	0.501269
558	0.372351	0.488084
363	0.392277	0.459069

277 rows × 2 columns

Результаты работы линейной модели Лассо для прогнозирования свойства модуля упругости при растяжении представлены ниже.

```
# Модель с параметрами по умолчанию
default_model_ls1 = Lasso().fit(x_train1, y_train1)
default_model_ls1
```

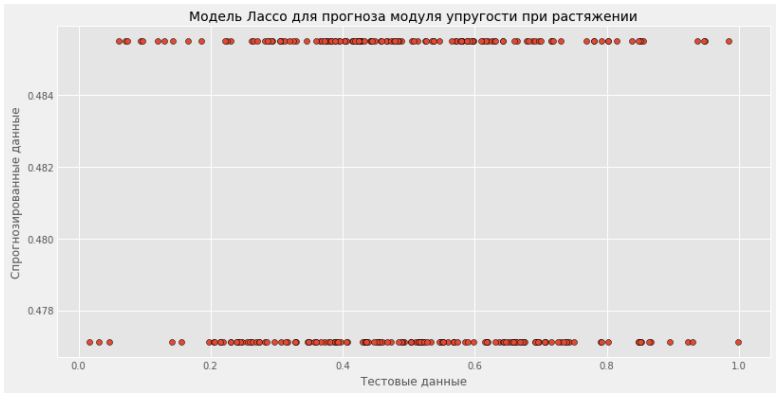
```
Lasso()
```

```
# Настроенная наилучшая модель
final_model_ls1 = grid_search_ls1.best_estimator_
final_model_ls1
```

```
Lasso(alpha=0.004)
```

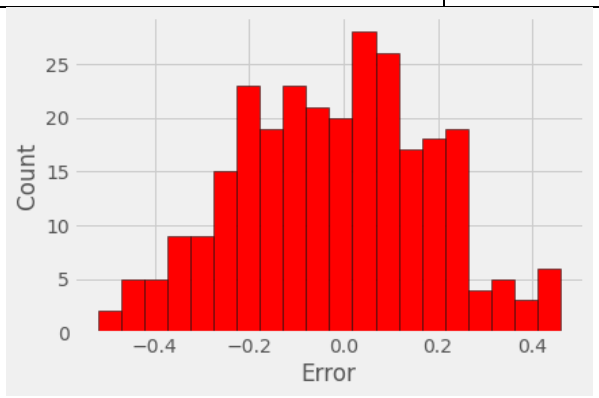
```
default_pred_ls1 = default_model_ls1.predict(x_test1)
final_pred_ls1 = final_model_ls1.predict(x_test1)

print('Средняя абсолютная ошибка модели со стандартными параметрами на тестовом наборе: MAE = %0.4f.' % mae(y_test1, default_pred_ls1))
print('Средняя абсолютная ошибка настроенной модели на тестовом наборе: MAE = %0.4f.' % mae(y_test1, final_pred_ls1))
```

```
Средняя абсолютная ошибка модели со стандартными параметрами на тестовом наборе: MAE = 0.1674.
Средняя абсолютная ошибка настроенной модели на тестовом наборе: MAE = 0.1678.
```



	Тестовые данные	Спрогнозированные данные
319	0.387108	0.477110
377	0.742719	0.477110
538	0.364855	0.485502
296	0.435807	0.477110
531	0.468100	0.485502
...	...	...
420	0.359362	0.477110
133	0.446885	0.477110
490	0.609638	0.485502
558	0.372351	0.485502
363	0.392277	0.477110

277 rows × 2 columns



▾ Сравнение моделей

```
# Сравнение моделей с подобранными параметрами
models = [Lasso(alpha=0.004),
          GradientBoostingRegressor(loss='lad', max_depth=15, max_features='auto',
                                    min_samples_split=10, random_state=42),
          RandomForestRegressor(criterion='mae', max_depth=12, max_features='sqrt',
                                min_samples_split=4, n_estimators=500)
          ]

print('Средняя абсолютная ошибка настроенной модели Lasso на тестовом наборе: MAE = %0.4f.' % mae(y_test1, final_pred_ls1))
print('Средняя абсолютная ошибка настроенной модели GradientBoostingRegressor на тестовом наборе: MAE = %0.4f.' % mae(y_test1, final_pred_gbr1))
print('Средняя абсолютная ошибка настроенной модели RandomForestRegressor на тестовом наборе: MAE = %0.4f.' % mae(y_test1, final_pred_rfr1))
```

```
Средняя абсолютная ошибка настроенной модели Lasso на тестовом наборе: MAE = 0.1678.
Средняя абсолютная ошибка настроенной модели GradientBoostingRegressor на тестовом наборе: MAE = 0.1728.
Средняя абсолютная ошибка настроенной модели RandomForestRegressor на тестовом наборе: MAE = 0.1677.
```

Результаты работы линейной модели Лассо для прогнозирования свойства прочности при растяжении представлены ниже.

```
# Модель с параметрами по умолчанию
default_model_ls2 = Lasso().fit(x_train2, y_train2)
default_model_ls2
```

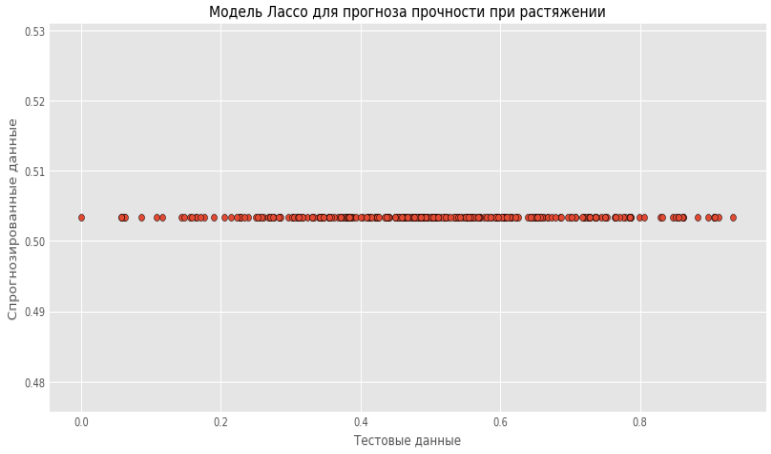
```
Lasso()
```

```
# Настроенная наилучшая модель
final_model_ls2 = grid_search_ls2.best_estimator_
final_model_ls2
```

```
Lasso(alpha=0.004)
```

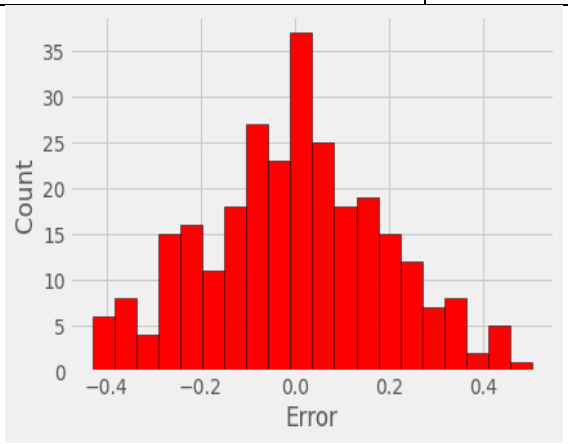
```
default_pred_ls2 = default_model_ls2.predict(x_test2)
final_pred_ls2 = final_model_ls2.predict(x_test2)

print('Средняя абсолютная ошибка модели со стандартными параметрами на тестовом наборе: MAE = %0.4f.' % mae(y_test2, default_pred_ls2))
print('Средняя абсолютная ошибка настроенной модели на тестовом наборе: MAE = %0.4f.' % mae(y_test2, final_pred_ls2))
```

```
Средняя абсолютная ошибка модели со стандартными параметрами на тестовом наборе: MAE = 0.1534.
Средняя абсолютная ошибка настроенной модели на тестовом наборе: MAE = 0.1534.
```



	Тестовые данные	Спрогнозированные данные
319	0.381499	0.503339
377	0.605408	0.503339
538	0.708160	0.503339
296	0.438781	0.503339
531	0.061865	0.503339
...	...	...
420	0.854766	0.503339
133	0.347529	0.503339
490	0.503825	0.503339
558	0.568094	0.503339
363	0.510845	0.503339

277 rows × 2 columns



Результаты работы модели на основе метода k-ближайших соседей для прогнозирования свойства прочности при растяжении представлены ниже.

```
# Модель с параметрами по умолчанию
default_model_knr2 = KNeighborsRegressor().fit(x_train2, y_train2)
default_model_knr2
```

```
KNeighborsRegressor()
```

```
# Настроенная наилучшая модель
final_model_knr2 = grid_search_knr2.best_estimator_
final_model_knr2
```

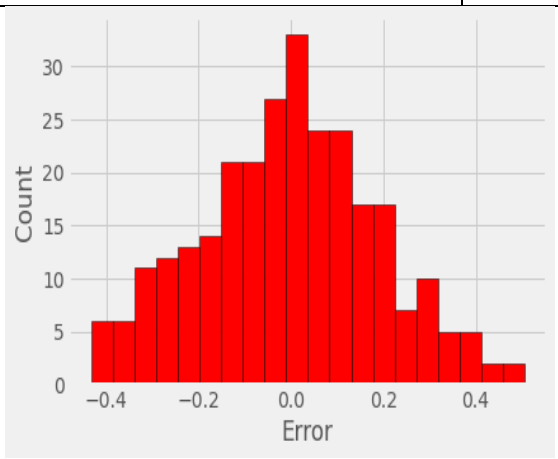
```
KNeighborsRegressor(n_neighbors=70, weights='distance')
```

```
default_pred_knr2 = default_model_knr2.predict(x_test2)
final_pred_knr2 = final_model_knr2.predict(x_test2)

print('Средняя абсолютная ошибка модели со стандартными параметрами на тестовом наборе: MAE = %0.4f.' % mae(y_test2, default_pred_knr2))
print('Средняя абсолютная ошибка настроенной модели на тестовом наборе: MAE = %0.4f.' % mae(y_test2, final_pred_knr2))
```

```
Средняя абсолютная ошибка модели со стандартными параметрами на тестовом наборе: MAE = 0.1666.
Средняя абсолютная ошибка настроенной модели на тестовом наборе: MAE = 0.1541.
```

	Тестовые данные	Спрогнозированные данные
319	0.381499	0.497051
377	0.605408	0.495601
538	0.708160	0.492720
296	0.438781	0.443859
531	0.061865	0.472416
...	...	...
420	0.854766	0.516820
133	0.347529	0.509669
490	0.503825	0.515138
558	0.568094	0.540368
363	0.510845	0.524114

277 rows × 2 columns



Результаты работы модели на основе метода опорных векторов для прогнозирования свойства прочности при растяжении представлены ниже.

```
# Модель с параметрами по умолчанию
default_model_svr2 = SVR().fit(x_train2, y_train2)
default_model_svr2
```

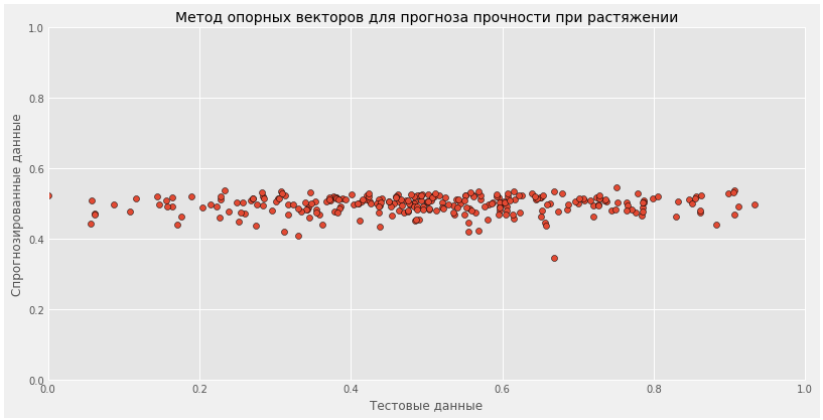
```
SVR()
```

```
# Настроенная наилучшая модель
final_model_svr2 = grid_search_svr2.best_estimator_
final_model_svr2
```

```
SVR(C=0.0001, degree=5, gamma=0.9, kernel='poly')
```

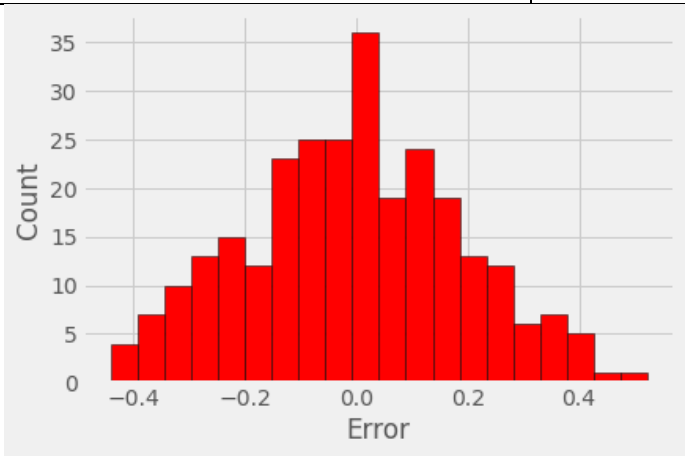
```
default_pred_svr2 = default_model_svr2.predict(x_test2)
final_pred_svr2 = final_model_svr2.predict(x_test2)

print('Средняя абсолютная ошибка модели со стандартными параметрами на тестовом наборе: MAE = %0.4f.' % mae(y_test2, default_pred_svr2))
print('Средняя абсолютная ошибка настроенной модели на тестовом наборе: MAE = %0.4f.' % mae(y_test2, final_pred_svr2))
```

```
Средняя абсолютная ошибка модели со стандартными параметрами на тестовом наборе: MAE = 0.1663.
Средняя абсолютная ошибка настроенной модели на тестовом наборе: MAE = 0.1541.
```



	Тестовые данные	Спрогнозированные данные
319	0.381499	0.518916
377	0.605408	0.493429
538	0.708160	0.509172
296	0.438781	0.436584
531	0.061865	0.471816
...	...	...
420	0.854766	0.515011
133	0.347529	0.499941
490	0.503825	0.486532
558	0.568094	0.534347
363	0.510845	0.528162

277 rows × 2 columns



Результаты работы модели на основе градиентного бустинга для прогнозирования свойства прочности при растяжении представлены ниже.

```
# Модель с параметрами по умолчанию
default_model_gbr2 = GradientBoostingRegressor(random_state = 42).fit(x_train2, y_train2)
default_model_gbr2
```

```
GradientBoostingRegressor(random_state=42)
```

```
# Настроенная наилучшая модель
final_model_gbr2 = grid_search_gbr2.best_estimator_
final_model_gbr2
```

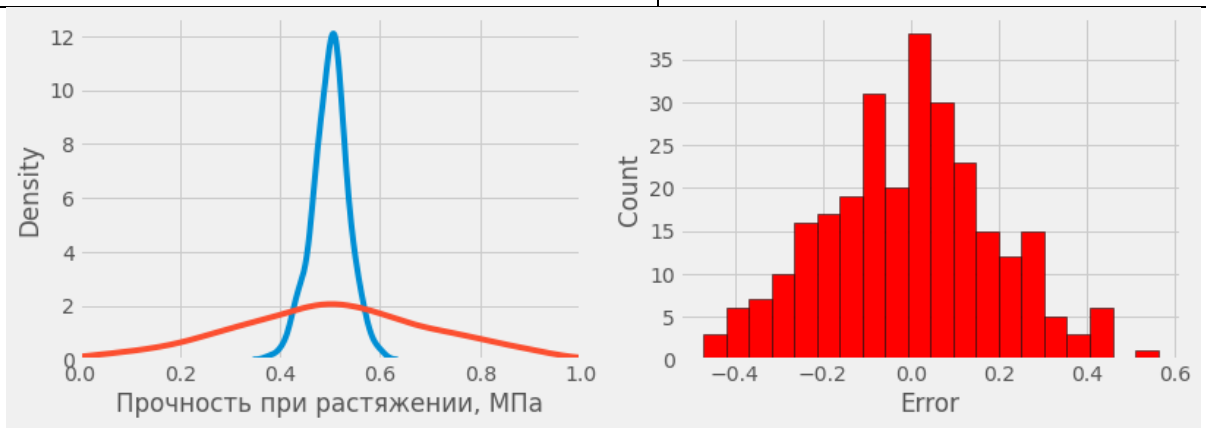
```
GradientBoostingRegressor(loss='lad', max_depth=2, max_features='sqrt',
                          min_samples_leaf=4, min_samples_split=10,
                          random_state=42)
```

```
default_pred_gbr2 = default_model_gbr2.predict(x_test2)
final_pred_gbr2 = final_model_gbr2.predict(x_test2)

print('Средняя абсолютная ошибка модели со стандартными параметрами на тестовом наборе: MAE = %0.4f.' % mae(y_test2, default_pred_gbr2))
print('Средняя абсолютная ошибка настроенной модели на тестовом наборе: MAE = %0.4f.' % mae(y_test2, final_pred_gbr2))
```

```
Средняя абсолютная ошибка модели со стандартными параметрами на тестовом наборе: MAE = 0.1590.
Средняя абсолютная ошибка настроенной модели на тестовом наборе: MAE = 0.1559.
```

	Тестовые данные	Спрогнозированные данные
319	0.381499	0.477776
377	0.605408	0.508165
538	0.708160	0.511722
296	0.438781	0.510172
531	0.061865	0.490830
...	...	...
420	0.854766	0.434579
133	0.347529	0.516971
490	0.503825	0.486106
558	0.568094	0.532512
363	0.510845	0.402278

277 rows × 2 columns



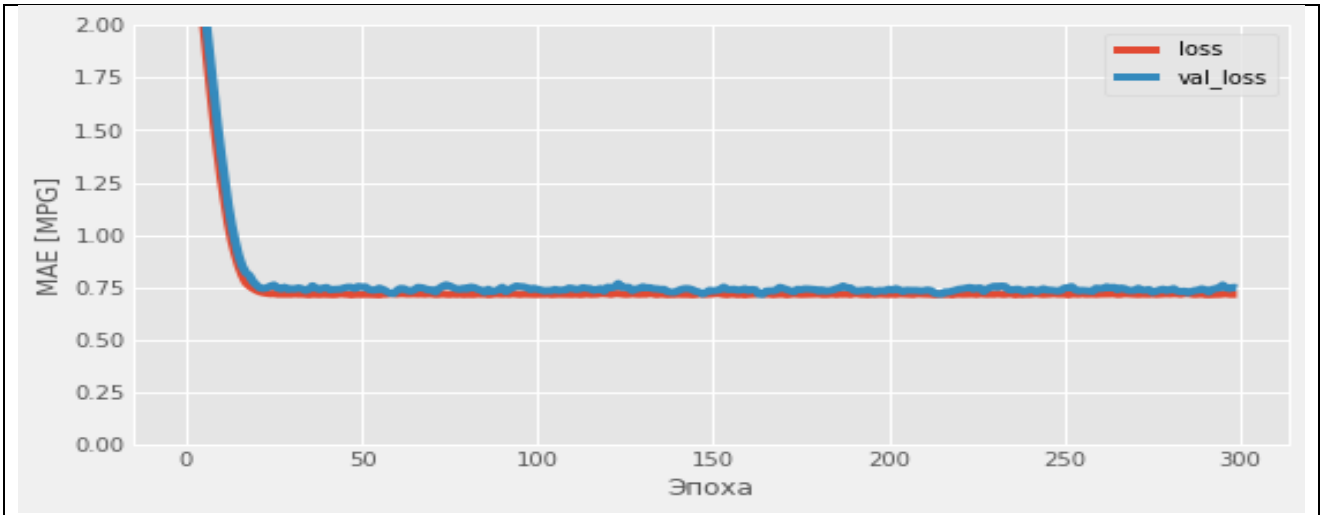
Сравнение моделей

```
[ ]  # Сравнение моделей с подобранными параметрами
     models = [Lasso(alpha=0.004),
               KNeighborsRegressor(n_neighbors=70, weights='distance'),
               SVR(C=0.0001, degree=5, gamma=0.9, kernel='poly'),
               GradientBoostingRegressor(loss='lad', max_depth=2, max_features='sqrt',
                                         min_samples_leaf=4, min_samples_split=10,
                                         random_state=42)
               ]

     print('Средняя абсолютная ошибка настроенной модели Lasso на тестовом наборе: MAE = %0.4f.' % mae(y_test2, final_pred_ls2))
     print('Средняя абсолютная ошибка настроенной модели KNeighborsRegressor на тестовом наборе: MAE = %0.4f.' % mae(y_test2, final_pred_knr2))
     print('Средняя абсолютная ошибка настроенной модели SVR на тестовом наборе: MAE = %0.4f.' % mae(y_test2, final_pred_svr2))
     print('Средняя абсолютная ошибка настроенной модели GradientBoostingRegressor на тестовом наборе: MAE = %0.4f.' % mae(y_test2, final_pred_gbr2))

     Средняя абсолютная ошибка настроенной модели Lasso на тестовом наборе: MAE = 0.1534.
     Средняя абсолютная ошибка настроенной модели KNeighborsRegressor на тестовом наборе: MAE = 0.1541.
     Средняя абсолютная ошибка настроенной модели SVR на тестовом наборе: MAE = 0.1541.
     Средняя абсолютная ошибка настроенной модели GradientBoostingRegressor на тестовом наборе: MAE = 0.1559.
```

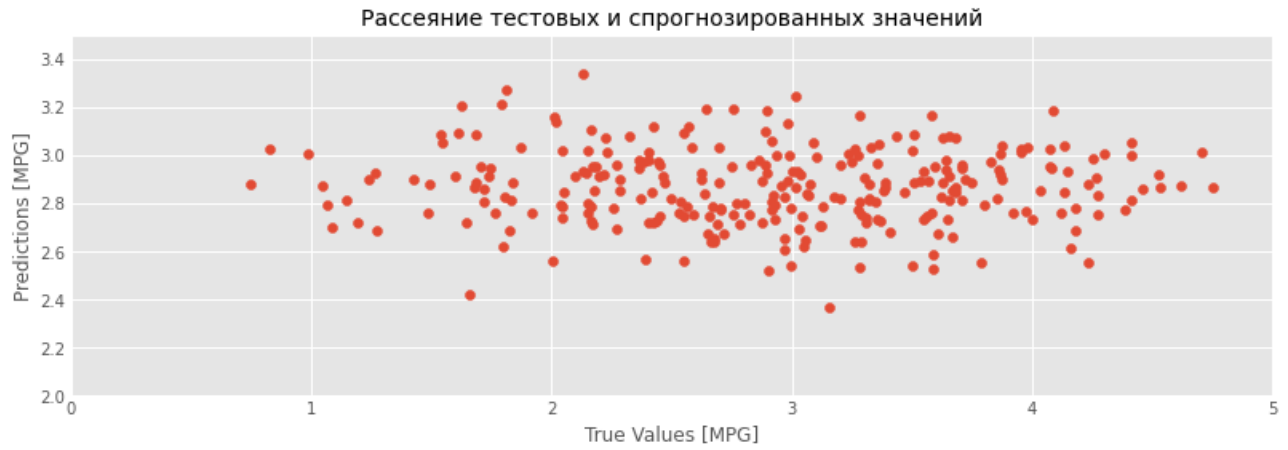
2.4 Нейронная сеть

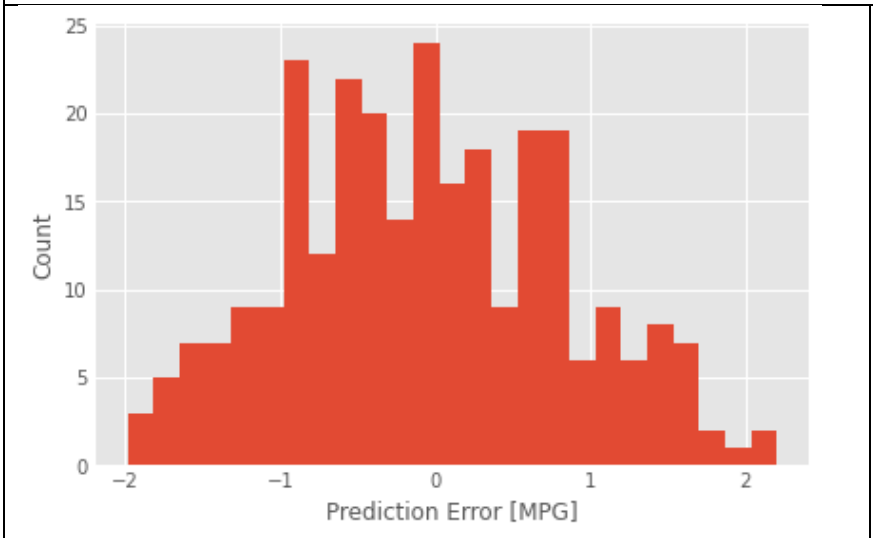
Для рекомендации соотношения «матрица-наполнитель» разработана и обучена линейная модель и многослойный персептрон с помощью библиотеки Keras.

Результаты работы линейной модели



Средняя абсолютная ошибка: 0.7294415831565857.





	Тест	Прогноз
360	2.097010	2.911542
427	4.073344	2.952605
611	1.647649	2.720194
336	3.391009	2.889096
604	3.578784	3.164991
...	...	...
479	2.424798	2.719007
149	1.916843	2.760573
561	3.011978	3.248160
632	2.622594	2.901837
411	1.273273	2.689256

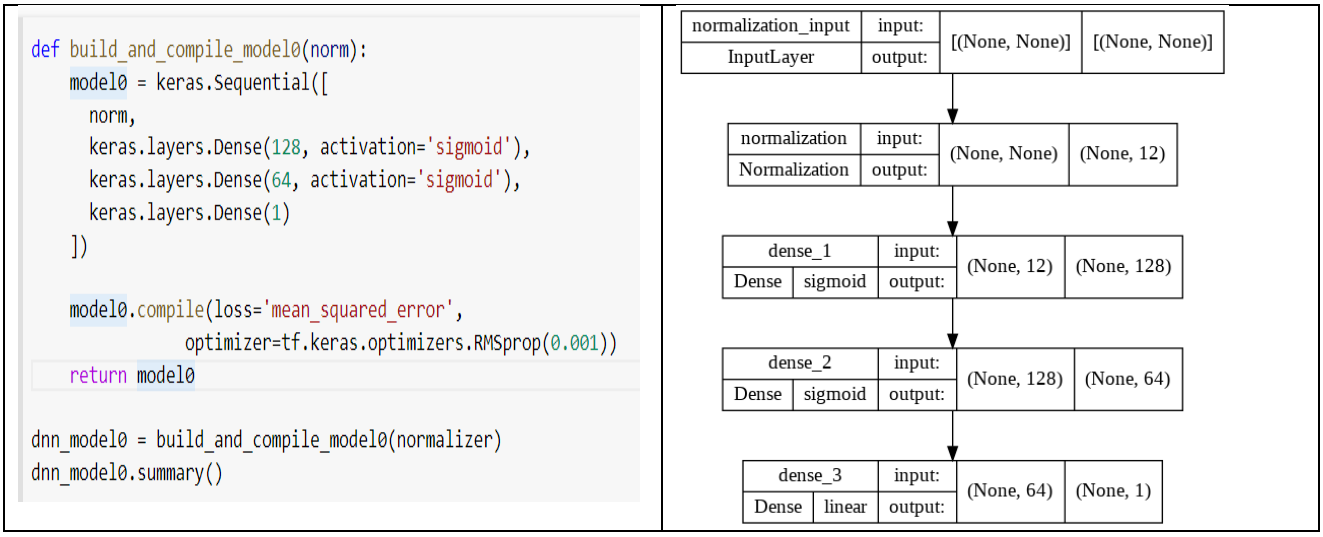
277 rows × 2 columns

Первая модель многослойного персептрона состоит из трех скрытых уровней. Первый уровень содержит 64 нейрона, последующие скрытые уровни содержат

64 и 1 нейрон. Снижение числа нейронов на каждом уровне сжимает информацию, которую сеть обработала на предыдущих уровнях.

Скрытые уровни нейронной сети трансформируются функциями активации, которые вносят в систему нелинейность.

Для первого эксперимента была выбрана функция активации сигмоида.

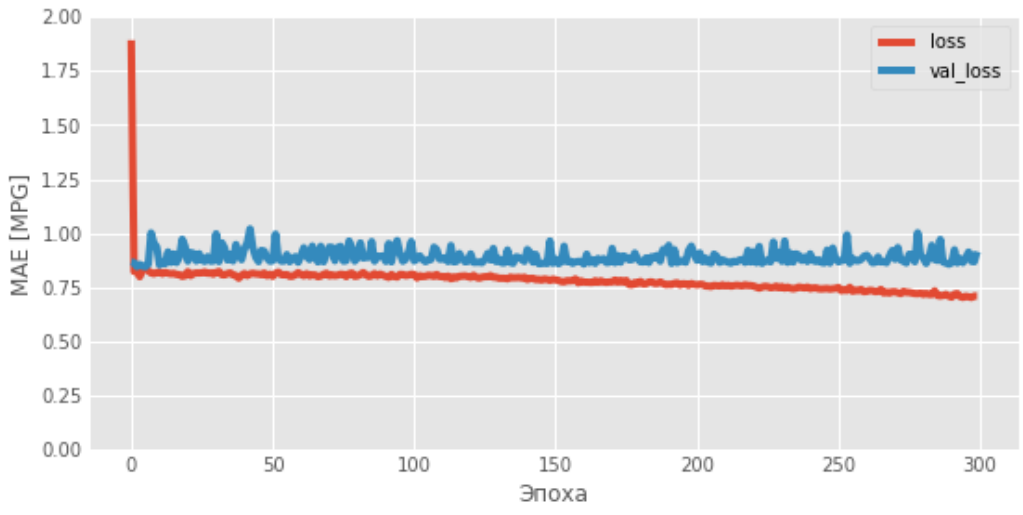


```
def build_and_compile_model0(norm):
    model0 = keras.Sequential([
      norm,
      keras.layers.Dense(128, activation='sigmoid'),
      keras.layers.Dense(64, activation='sigmoid'),
      keras.layers.Dense(1)
    ])

    model0.compile(loss='mean_squared_error',
                optimizer=tf.keras.optimizers.RMSprop(0.001))
    return model0

dnn_model0 = build_and_compile_model0(normalizer)
dnn_model0.summary()
```

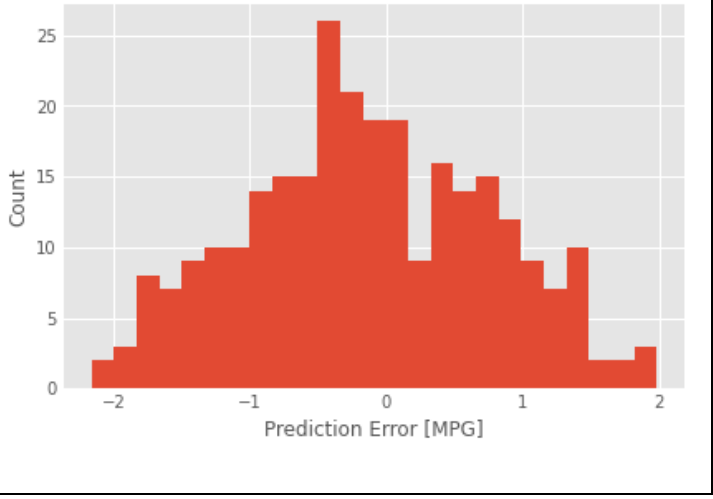

Рисунок 13. Архитектура первого многослойного персептрона

Для оценки отклонения между прогнозами сети и реальными результатами в ходе обучения использовалась функция средней квадратичной ошибки.

В качестве оптимизатора использовался RMSprop-оптимизатор, алгоритм которого похож на метод градиентного спуска с импульсом.



Средняя абсолютная ошибка: 0.8168711066246033.



	Тест	Прогноз
360	2.097010	2.698929
427	4.073344	2.787695
611	1.647649	2.989741
336	3.391009	2.654714
604	3.578784	2.083353
...	...	...
479	2.424798	2.508657
149	1.916843	2.745739
561	3.011978	2.896571
632	2.622594	2.435316
411	1.273273	2.652678

277 rows × 2 columns

Вторая модель многослойного персептрона состоит из пяти скрытых уровней. Первый уровень содержит 256 нейронов, последующие скрытые уровни содержат 192, 128, 64 и 1 нейрон. Была выбрана функция активации relu. Добавили Dropout – метод борьбы с переобучением нейронной сети.

```python
def build_and_compile_model1(norm):
    model1 = keras.Sequential([
      norm,
      keras.layers.Dense(256, activation='relu'),
      keras.layers.Dropout(0.8),
      keras.layers.Dense(192, activation='relu'),
      keras.layers.Dropout(0.8),
      keras.layers.Dense(128, activation='relu'),
      keras.layers.Dropout(0.8),
      keras.layers.Dense(64, activation='relu'),
      keras.layers.Dropout(0.8),
      keras.layers.Dense(1)
    ])

    model1.compile(loss='mean_absolute_error',
                optimizer=tf.keras.optimizers.Adam(learning_rate=0.001))
    return model1
```

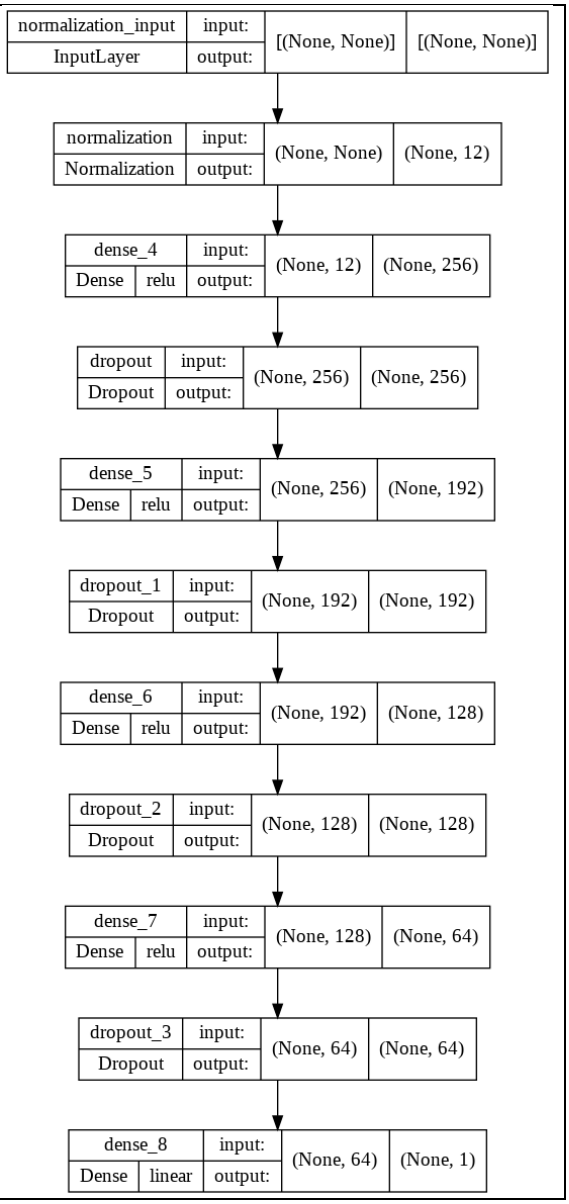
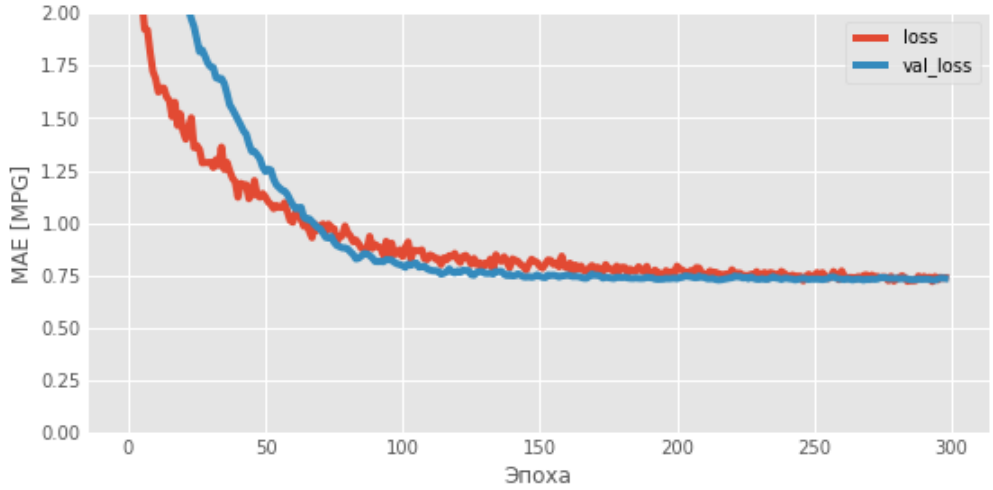


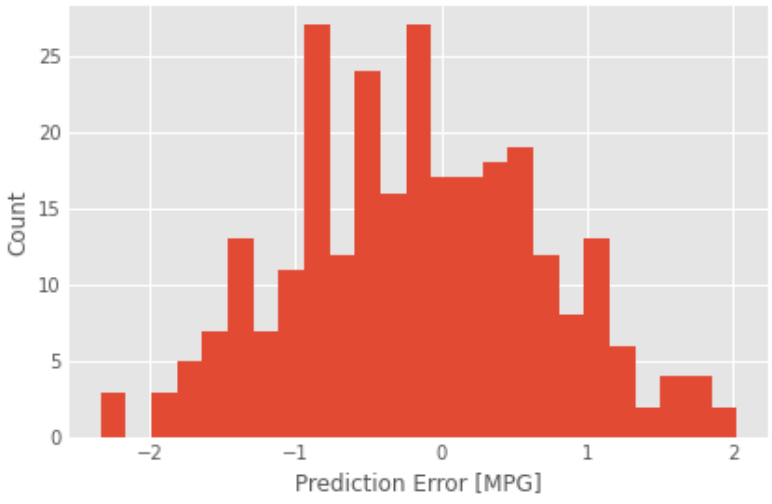
Рисунок 13. Архитектура второго многослойного персептрона

Для оценки отклонения между прогнозами сети и реальными результатами в ходе обучения использовалась функция средней абсолютной ошибки.

В качестве оптимизатора использовался Adam.



Средняя абсолютная ошибка: 0.7274526953697205.



	Тест	Прогноз
360	2.097010	2.771945
427	4.073344	2.771527
611	1.647649	2.772239
336	3.391009	2.771825
604	3.578784	2.772597
...	...	...
479	2.424798	2.771859
149	1.916843	2.772062
561	3.011978	2.771788
632	2.622594	2.772402
411	1.273273	2.771854

277 rows × 2 columns

2.5 Разработка приложения

Приложение для прогнозирования конечных свойств композиционных материалов не разработано.

2.6 Создание удаленного репозитория

Страница удаленного репозитория создана на GitHub.
Адрес страницы: https://github.com/Kostromina90/bmstu_DS_Kostromina_OS

Заключение

Несмотря на большую проделанную работу, результаты которой лишь частично представлены в отчете, построенные и обученные модели не решают поставленных задач прогнозирования модуля упругости при растяжении и прочности при растяжении композиционных материалов. Все модели не удовлетворительно описывают исходные данные. Построенные и обученные нейронные сети также не справились с задачей рекомендации соотношения матрица-наполнитель.

Приложение для прогнозирования конечных свойств композиционных материалов не разработано.

Список литературы

1. Грас Д. Data Science. Наука о данных с нуля. Пер. с англ. – 2-е изд., перераб. и доп. – СПБ.: БХВ-Петербург, 2022.
2. Постолит А. В. Основы искусственного интеллекта в примерах на Python. Самоучитель. – СПБ.: БХВ-Петербург, 2022.
3. Джорши Практик. Искусственный интеллект с примерами на Python. Пер. с англ. – СПБ.: ООО «Диалектика», 2019.
4. Введение в статистическое обучение с примерами на языке R / Джеймс Г. [и др.]. – 2-е изд., испр. – Москва : ДМК Пресс, 2018.
5. Документация по языку программирования python: – Режим доступа: https://docs.python.org/3.8/index.html.
6. Документация по библиотеке pandas: – Режим доступа: https://pandas.pydata.org/docs/user_guide/index.html#user-guide.
7. Документация по библиотеке numpy: – Режим доступа: https://numpy.org/doc/1.22/user/index.html#user.
8. Документация по библиотеке matplotlib: – Режим доступа: https://matplotlib.org/stable/users/index.html
9. Документация по библиотеке seaborn: – Режим доступа: https://seaborn.pydata.org/tutorial.html.
10. Документация по библиотеке sklearn: – Режим доступа: https://scikitlearn.org/stable/user_guide.html.
11. Делаем проект по машинному обучению на Python. Часть 1: https://habr.com/ru/company/nix/blog/425253/
12. Делаем проект по машинному обучению на Python. Часть 2: https://habr.com/ru/company/nix/blog/425907/