

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

Гутенев Евгений Олегович

Москва, 2022

Содержание

1. Аналитическая часть	4
1.1. Постановка задачи	4
1.2. Описание используемых методов	7
1.2.1. Линейная регрессия	7
1.2.2. Лассо	8
1.2.3. Метод опорных векторов для регрессии	8
1.2.4. Метод k-ближайших соседей.....	9
1.2.5. Деревья решений	10
1.2.6. Случайный лес	11
1.2.7. Градиентный бустинг.....	12
1.2.8. Нейронная сеть	13
1.3. Разведочный анализ данных	14
2. Практическая часть	17
2.1. Предобработка данных	17
2.1.1. Для прогнозирования модуля упругости при растяжении	17
2.1.2. Для прогнозирования прочности при растяжении	18
2.1.3. Для прогнозирования соотношения матрица-наполнитель	19
2.2. Разработка и обучение моделей.....	20
2.2.1. Для прогнозирования модуля упругости при растяжении	20
2.2.2. Для прогнозирования прочности при растяжении	23
2.3. Тестирование моделей	26
2.4. Разработка нейросети для прогнозирования соотношения матрица-наполнитель	27
2.5. Разработка приложения.....	32
2.6. Создание удаленного репозитория	32
Заключение	33
Библиографический список	34
Приложение А. Скриншот разработанного приложения	35

Введение

Композиционные материалы – искусственно созданные материалы, состоящие из двух или более неоднородных и нерастворимых друг в друге компонентов, соединяемых между собой физико-химическими связями.

Одним из компонентов композиционных материалов является арматура, или наполнитель, обеспечивающие необходимые механические характеристики материала, а другим компонентом – матрица, обеспечивающая совместную работу армирующих элементов. Свойства композиционных материалов зависят от состава компонентов, количественного соотношения и прочности связи между ними. Комбинируя объемное содержание компонентов, можно в зависимости от назначения получить композиции с необходимыми специальными свойствами. Композиционные материалы имеют высокую удельную и усталостную прочность, жесткость и износстойкость. Из них можно изготовить уникальные по эксплуатационным свойствам размеростабильные конструкции.

Композиционные материалы являются весьма перспективными конструкционными материалами для множества отраслей машиностроения. В свою очередь, осуществить поддержку контроля качества и инновационных исследований в производственном секторе могут технологии машинного обучения.

Учитывая широкое распространение и применение композиционных материалов, имеющуюся потребность в создании новых специфических конструкций, а также высокую стоимость их производства, связанную прежде всего с проведением большого количества испытаний для получения необходимых свойств, видится целесообразным использование принципов машинного обучения для сокращения времени и затрат на создание определенного конечного продукта. Алгоритмы машинного обучения могут помочь исследователям анализировать, как незначительные изменения основных характеристик улучшают абразивность и долговечность вновь созданного материала. Эти предложения могут положительно повлиять на производственный процесс.

1. Аналитическая часть

1.1. Постановка задачи

Предмет исследования - композитные материалы с матрицей из базальтопластика и нашивками из углепластика. Имеется датасет, содержащий данные о начальных свойствах компонентов композиционных материалов. Требуется разработать модели, прогнозирующие значения ряда основных свойств композитных материалов, а также создать приложение, которое будет выдавать прогноз по одной из них.

Датасет состоит из двух файлов: X_bp (матрица из базальтопластика, содержит 1 индекс, 10 признаков, 1023 строк) и X_nip (наполнитель из углепластика, содержит 1 индекс, 3 признака, 1040 строк). После объединения данных файлов по индексу типом INNER (по условию) дальнейшее исследование будет проводиться с датасетом, состоящим из 13 признаков и 1023 строк.

Описание признаков датасета приведено в таблице 1. Все признаки имеют тип float64. Пропуски в данных отсутствуют. Все признаки, кроме «Угол нашивки», являются непрерывными, количественными. «Угол нашивки» принимает только два значения.

Таблица 1 — Описание признаков

Название	Тип данных	Ненулевые значения	Уникальные значений
Соотношение матрица-наполнитель	float64	1023	1014
Плотность, кг/м3	float64	1023	1013
модуль упругости, ГПа	float64	1023	1020
Количество отвердителя, м.%	float64	1023	1005
Содержание эпоксидных групп,%_2	float64	1023	1004
Температура вспышки, С_2	float64	1023	1003
Поверхностная плотность, г/м2	float64	1023	1004
Модуль упругости при растяжении, ГПа	float64	1023	1004

Прочность при растяжении, МПа	float64	1023	1004
Потребление смолы, г/м ²	float64	1023	1003
Угол нашивки, град	float64	1023	2
Шаг нашивки	float64	1023	989
Плотность нашивки	float64	1023	988

Гистограммы и диаграммы распределения значений признаков приведены на рисунке 1. Практически все признаки имеют нормальное распределение, принимают неотрицательные значения. «Угол нашивки» имеет значения 0 и 90.

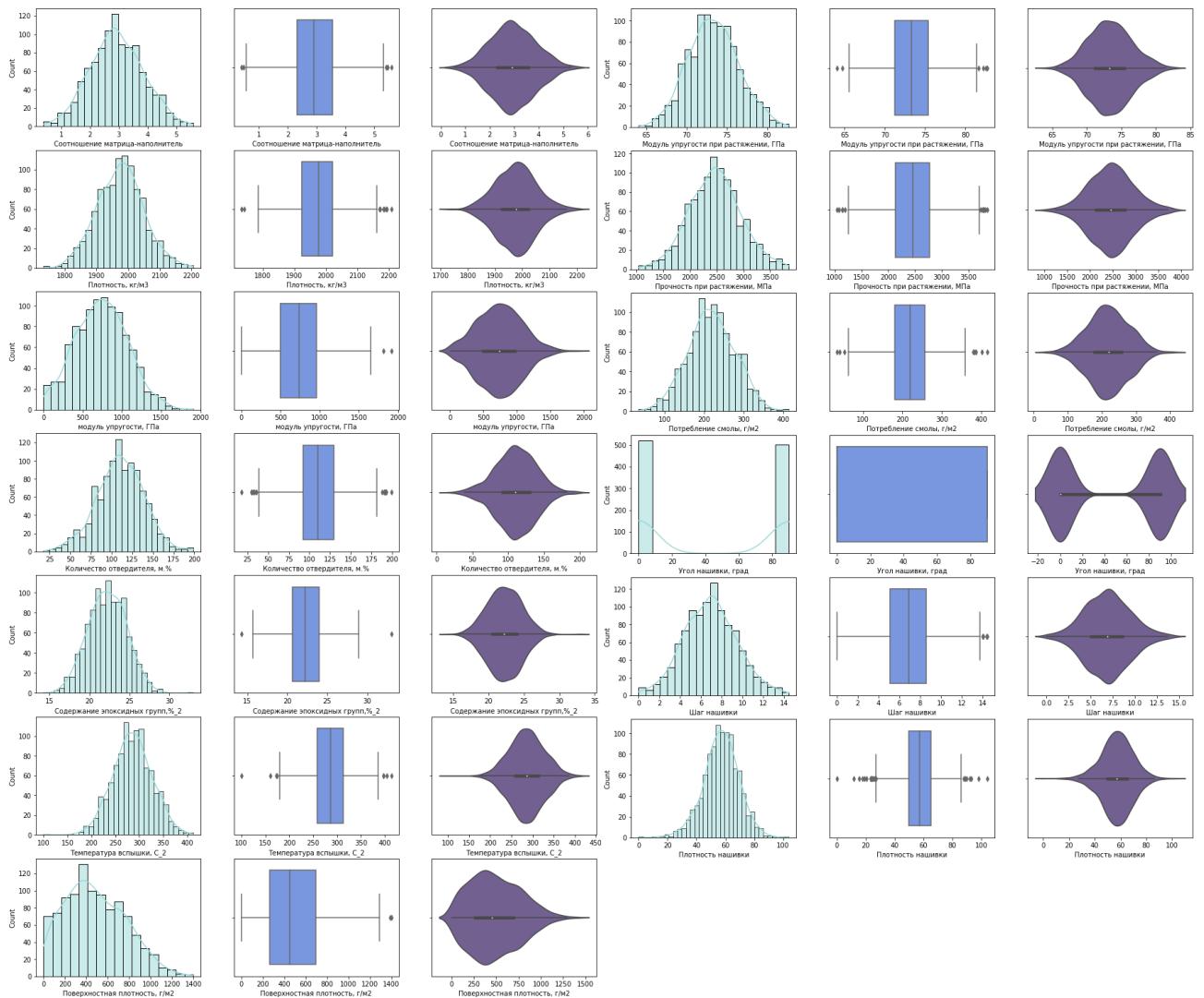


Рисунок 1 - Гистограммы и диаграммы распределения значений признаков

Далее, на рисунке 2, представлена описательная статистика датасета. Попарные графики рассеяния точек приведены на рисунке 3.

	count	mean	std	min	25%	50%	75%	max	median
Соотношение матрица-наполнитель	1023.000	2.930	0.913	0.389	2.318	2.907	3.553	5.592	2.907
Плотность, кг/м3	1023.000	1975.735	73.729	1731.765	1924.155	1977.622	2021.374	2207.773	1977.622
модуль упругости, ГПа	1023.000	739.923	330.232	2.437	500.047	739.664	961.813	1911.536	739.664
Количество отвердителя, м.%	1023.000	110.571	28.296	17.740	92.443	110.565	129.730	198.953	110.565
Содержание эпоксидных групп,%_2	1023.000	22.244	2.406	14.255	20.608	22.231	23.962	33.000	22.231
Температура вспышки, С_2	1023.000	285.882	40.943	100.000	259.067	285.897	313.002	413.273	285.897
Поверхностная плотность, г/м2	1023.000	482.732	281.315	0.604	266.817	451.864	693.225	1399.542	451.864
Модуль упругости при растяжении, ГПа	1023.000	73.329	3.119	64.054	71.245	73.269	75.357	82.682	73.269
Прочность при растяжении, МПа	1023.000	2466.923	485.628	1036.857	2135.850	2459.525	2767.193	3848.437	2459.525
Потребление смолы, г/м2	1023.000	218.423	59.736	33.803	179.628	219.199	257.482	414.591	219.199
Угол нашивки, град	1023.000	44.252	45.016	0.000	0.000	0.000	90.000	90.000	0.000
Шаг нашивки	1023.000	6.899	2.563	0.000	5.080	6.916	8.586	14.441	6.916
Плотность нашивки	1023.000	57.154	12.351	0.000	49.799	57.342	64.945	103.989	57.342

Рисунок 2 - Описательная статистика

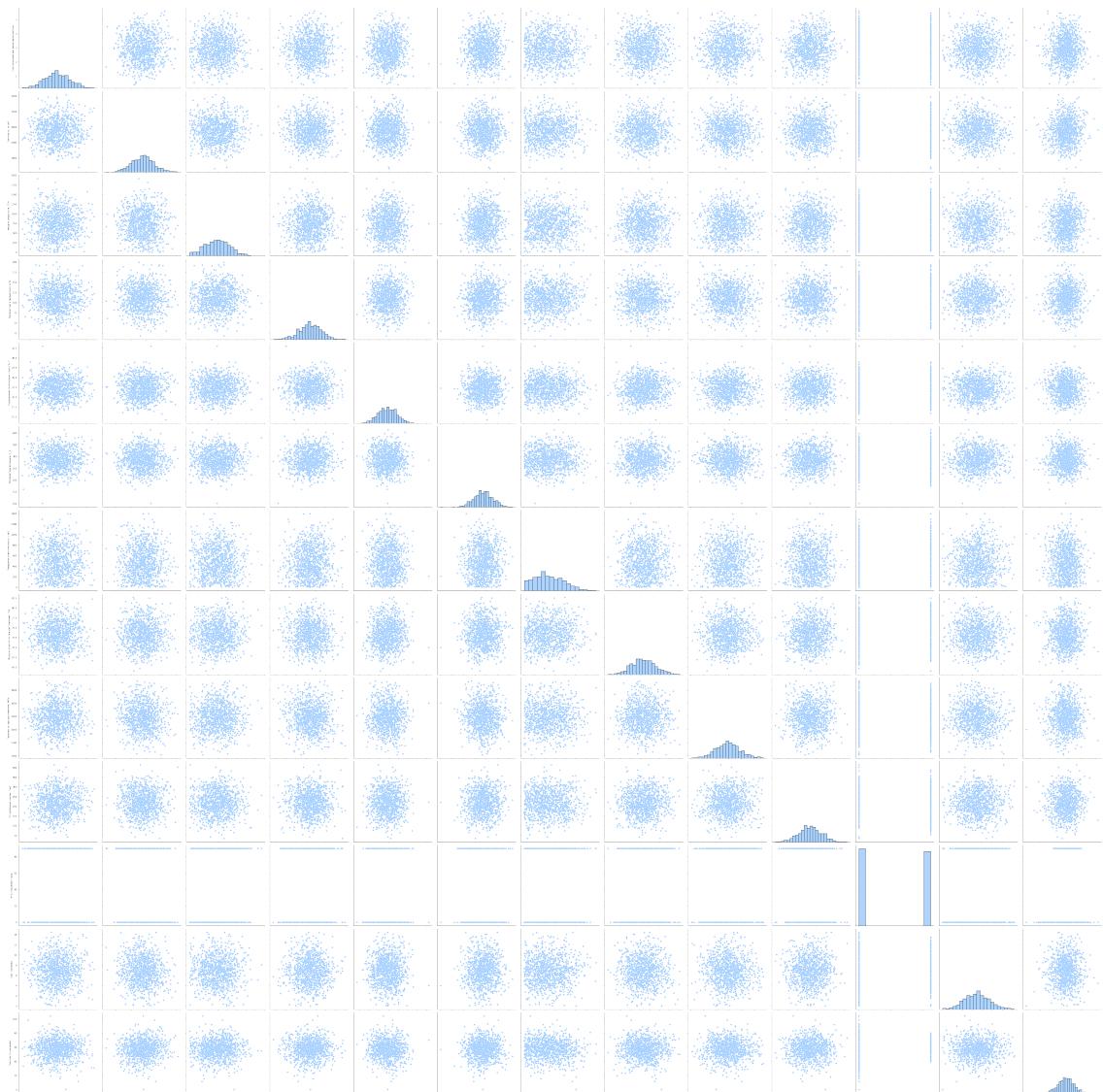


Рисунок 3 - Попарные графики рассеяния точек

На рисунке 3 мы видим, что ряд точек выделяется из общей выборки. Поэтому на следующем этапе необходимо выявить выбросы и исключить их из датасета. Так, воспользовавшись методом трех сигм, получаем 24 выброса. После удаления строк с полученными аномальными значениями признаков датасет будет состоять из 1000 строк.

В качестве выходных переменных будут следующие признаки:

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

1.2. Описание используемых методов

Прогнозирование значений вещественной непрерывной переменной на основе выборки объектов с различными признаками является задачей регрессии. Данная зависимая переменная должна иметь связь с одной или несколькими независимыми переменными. На текущий момент существует множество методов регрессионного анализа.

1.2.1. Линейная регрессия

Простая линейная регрессия имеет место, если рассматривается зависимость между одной входной и одной выходной переменными. Для этого определяется уравнение регрессии (1) и строится соответствующая прямая, известная как линия регрессии.

$$y = ax + b \quad (1)$$

Коэффициенты a и b , называемые также параметрами модели, определяются таким образом, чтобы сумма квадратов отклонений точек, соответствующих реальным наблюдениям данных, от линии регрессии была бы минимальной. Коэффициенты обычно оцениваются методом наименьших квадратов.

Если ищется зависимость между несколькими входными и одной выходной переменными, то имеет место множественная линейная регрессия. Соответствующее уравнение имеет вид (2).

$$Y = b_0 + b_1 * x_1 + b_2 * x_2 + \dots + b_n * x_n, \quad (2)$$

где n - число входных переменных.

Очевидно, что в данном случае модель будет описываться не прямой, а гиперплоскостью. Коэффициенты уравнения множественной линейной регрессии подбираются таким образом, чтобы минимизировать сумму квадратов отклонения реальных точек данных от этой гиперплоскости.

Линейная регрессия — первый тщательно изученный метод регрессионного анализа. Его главное достоинство — простота. Такую модель можно построить и рассчитать без использования мощных вычислительных средств. Простота является и главным недостатком этого метода. Тем не менее, именно с линейной регрессии целесообразно начать подбор подходящей модели.

Данный метод реализован в `sklearn.linear_model.LinearRegression`.

1.2.2. Лассо

Метод регрессии Лассо (Lasso, Least Absolute Shrinkage and Selection Operator) — это вариация линейной регрессии, специально адаптированная для данных, которые имеют сильную корреляцию признаков друг с другом.

Лассо использует сжатие коэффициентов и тем самым пытается уменьшить сложность данных, искривляя пространство, на котором они лежат. В этом процессе данный метод автоматически помогает устраниить или исказить сильно коррелированные и избыточные функции в методе с низкой дисперсией. Регрессия лассо использует регуляризацию L1, то есть взвешивает ошибки по их абсолютному значению.

Регуляризация позволяет интерпретировать модели. Если коэффициент стал равен 0, значит данный входной признак не является значимым.

Данный метод реализован в `sklearn.linear_model.Lasso`.

1.2.3. Метод опорных векторов для регрессии

Метод опорных векторов (SupportVectorMachine, SVM) — один из наиболее популярных методов машинного обучения. Он создает гиперплоскость или

набор гиперплоскостей в многомерном пространстве, которые могут быть использованы для решения задач классификации и регрессии. Чаще всего он применяется в постановке бинарной классификации.

Основная идея заключается в построении гиперплоскости, разделяющей объекты выборки оптимальным способом. Максимально близкие объекты разных классов определяют опорные вектора. Если в исходном пространстве объекты линейно неразделимы, то выполняется переход в пространство большей размерности.

Решается задача оптимизации. Для вычислений используется ядерная функция, получающая на вход два вектора и возвращающая меру сходства между ними: линейная, полиномиальная и гауссовская (rbf).

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

Преимущество метода — его хорошая изученность. В качестве недостатков можно выделить чувствительность к выбросам и отсутствие интерпретируемости.

Вариация метода для регрессии называется SVR (SupportVector-Regression).

Данный метод реализован в `sklearn.svm.SVR`.

1.2.4. Метод k-ближайших соседей

Метод k-ближайших соседей (k-NearestNeighbors) является еще одним методом классификации, который адаптирован для регрессии. На интуитивном уровне суть метода проста: посмотри на соседей вокруг, какие из них преобладают, таковым ты и являешься.

В случае использования метода для регрессии, объекту присваивается среднее значение по k-ближайшим к нему объектам, значения которых уже известны.

Для реализации метода необходима метрика расстояния между объектами. Используется, например, евклидово расстояние для количественных при-

знаков или расстояние Хэмминга для категориальных. Этот метод — пример непараметрической регрессии.

Данный метод реализован в `sklearn.neighbors.KneighborsRegressor`.

1.2.5. Деревья решений

Деревья решений (DecisionTrees) - еще один непараметрический метод, применяемый как для решения задач классификации, так и задач регрессии. Деревья решений используются в самых разных сферах деятельности и представляют собой иерархические древовидные структуры, состоящие из правил вида «Если ..., то ...».

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 и листьев. 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 правило 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 производятся 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.

При формировании правила для разбиения в очередном узле дерева необходимо выбрать атрибут, по которому это будет сделано. Общее правило для классификации можно сформулировать так: выбранный атрибут должен разбить множество наблюдений в узле так, чтобы результирующие подмножества содержали примеры с одинаковыми метками класса, а количество объектов из

других классов в каждом из этих множеств было как можно меньше. Для этого были выбраны различные критерии, например, теоретико-информационный и статистический.

Для регрессии критерием является дисперсия вокруг среднего. Минимизируя дисперсию вокруг среднего, происходит поиск признаков, разбивающих выборку таким образом, что значения целевого признака в каждом листе примерно равны.

Огромное преимущество деревьев решений в том, что они легко интерпретируются и понятны. Они могут использоваться для извлечения правил на естественном языке, имеют высокую 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.

Данный метод реализован в `sklearn.tree.DecisionTreeRegressor`.

1.2.6. Случайный лес

Случайный лес (RandomForest) — представитель ансамблевых методов.

Если точность дерева решений оказалось недостаточной, мы можем множество моделей собрать в коллектив. Формула итогового решателя (3) — это усреднение предсказаний отдельных деревьев.

$$a(x) = \frac{1}{N} \sum_{i=1}^N b_i(x) \quad (3),$$

где N — количество деревьев, i — счетчик для деревьев, b — решающее дерево, x — сгенерированная нами на основе данных выборка.

Для определения входных данных каждому дереву используется метод случайных подпространств. Базовые алгоритмы обучаются на различных подмножествах признаков, которые выделяются случайным образом.

Преимуществами случайного леса являются высокая точность предсказания, отсутствие склонности к переобучению и чувствительности к выбросам в данных, оптимальная обработка как непрерывных, так и дискретных признаков, данных с большим числом признаков, высокая параллелизуемость и масштабируемость. Из недостатков можно отметить, что построение модели занимает большее количество времени и возможную потерю интерпретируемости.

Данный метод реализован в `sklearn.ensemble.RandomForestRegressor`.

1.2.7. Градиентный бустинг

Градиентный бустинг (GradientBoosting) — еще один представитель ансамблевых методов.

В отличие от случайного леса, где каждый базовый алгоритм строится независимо от остальных, бустинг воплощает идею последовательного построения линейной комбинации алгоритмов. Каждый следующий алгоритм старается уменьшить ошибку предыдущего. Для того, чтобы построить алгоритм градиентного бустинга, необходимо выбрать базовый алгоритм и функцию потерь или ошибки (loss). Loss-функция — это мера, которая показывает насколько хорошо предсказание модели соответствуют данным. Используя градиентный спуск и обновляя предсказания, основанные на скорости обучения (learning rate), происходит поиск значения, на которых loss минимальна.

Бустинг, использующий деревья решений в качестве базовых алгоритмов, называется градиентным бустингом над решающими деревьями. Он отлично работает на выборках с табличными, неоднородными данными и способен эффективно находить нелинейные зависимости в данных различной природы. На настоящий момент это один из самых эффективных алгоритмов машинного обучения. Благодаря этому он широко применяется во многих тематических соревнованиях и промышленных задачах. Он проигрывает только нейросетям на

однородных данных (изображения, звук и т. д.). Из недостатков алгоритма можно отметить затраты времени на вычисления и необходимость грамотного подбора гиперпараметров.

Данный метод реализован в `sklearn.ensemble.GradientBoostingRegressor`.

1.2.8. Нейронная сеть

Нейронная сеть — это последовательность нейронов, соединенных между собой связями. Структура нейронной сети пришла в мир программирования из биологии. Вычислительная единица нейронной сети — нейрон или персепtron.

У каждого нейрона есть определённое количество входов, куда поступают сигналы, которые суммируются с учётом значимости (веса) каждого входа. Смещение — это дополнительный вход для нейрона, который всегда равен 1 и, следовательно, имеет собственный вес соединения. Также у нейрона есть функция активации, которая определяет выходное значение нейрона. Она используется для того, чтобы ввести нелинейность в нейронную сеть. Примеры активационных функций: `relu`, `sigmoid`, `tahn`, `softmax`.

У полносвязной нейросети выход каждого нейрона подается на вход всем нейронам следующего слоя. У нейросети имеется входной слой (его размер соответствует входным параметрам), скрытые слои (их количество и размерность определяется специалистом по машинному обучению), выходной слой (его размер соответствует выходным параметрам).

Прямое распространение — это процесс передачи входных значений в нейронную сеть и получения выходных данных, которые называются прогнозируемым значением.

Прогнозируемое значение сравниваем с фактическим с помощью функции потери. В методе обратного распространения ошибки градиенты (производные значений ошибок) вычисляются по значениям весов в направлении, обратном прямому распространению сигналов. Значение градиента вычитают из значения веса, чтобы уменьшить значение ошибки. Таким образом происходит

процесс обучения. Обновляются веса каждого соединения, чтобы функция потеря минимизировалась.

Для обновления весов в модели используются различные оптимизаторы. Количество эпох показывает, сколько раз выполнялся проход для всех примеров обучения. 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.

Будем использовать методы из `sklearn.neural_network.MLPRegressor` и `tensorflow.keras.Sequential`.

1.3. Разведочный анализ данных

Для корректной работы большинства моделей важна сильная зависимость выходных переменных от входных и отсутствие зависимости между входными переменными. График попарного рассеяния точек не помог выявить связь между имеющимися признаками. Воспользуемся матрицей корреляции, изображенной на рисунке 4.

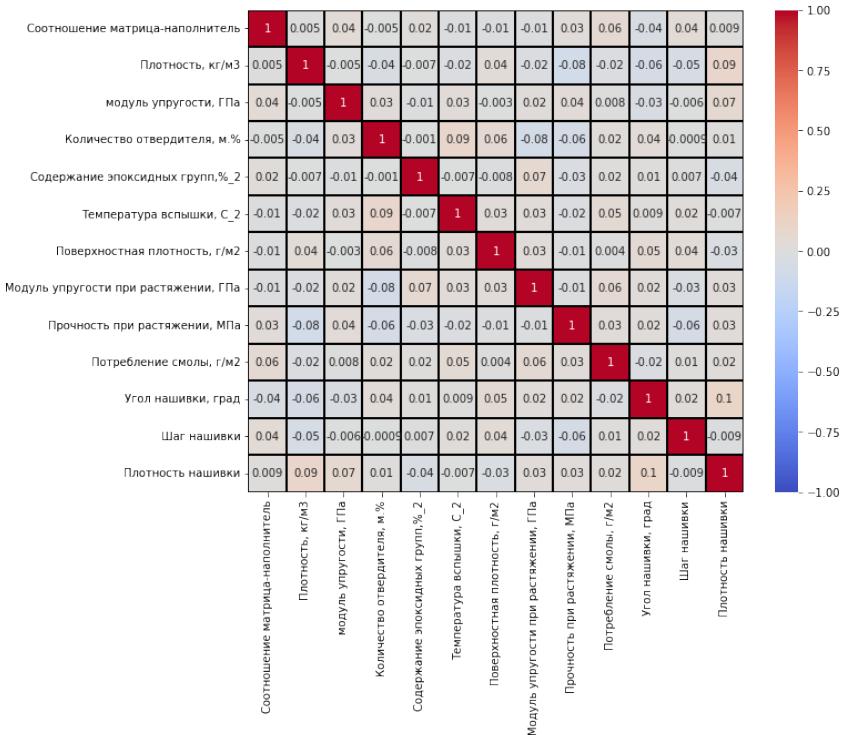


Рисунок 4 — Матрица корреляции

Коэффициенты корреляции, близкие к 0, показывают отсутствие линейной зависимости между признаками. Максимальная корреляция наблюдается между признаками «Плотность нашивки» и «Угол нашивки» и составляет 0.1. Таким образом, статистические методы не помогли найти искомые зависимости между признаками.

В связи с тем, что никакой конкретной информации относительно распределения признаков по группам по условию не дано, определим несколько зависимостей. Так, целевые признаки будут иметь следующие зависимости:

1. «Модуль упругости при растяжении, Гпа» - зависит от всех имеющихся в датасете признаков, за исключением признака «Прочность при растяжении, МПа»;
2. «Прочность при растяжении, МПа» - зависит от всех имеющихся в датасете признаков, за исключением признака «Модуль упругости при растяжении, Гпа»;
3. «Соотношение матрица-наполнитель» - зависит от всех имеющихся в датасете признаков, без исключений.

Для каждого из 3 целевых признаков построим отдельную модель для прогноза.

Процесс построения оптимальной модели для прогнозирования каждого из целевых признаков будет включать разделение данных на тренировочную и тестовую выборки (по условию, на тестирование необходимо оставить 30% данных), подготовку исходных данных, выбор базовой модели для определения нижней границы качества предсказания, выбор моделей с гиперпараметрами по умолчанию и сравнение их метрик на тренировочной выборке, выбор моделей с гиперпараметрами с помощью поиска по сетке с перекрестной проверкой (по условию, количество блоков равно 10), сравнение метрик моделей после подбора гиперпараметров и выбор лучшей, получение предсказания базовой и лучшей из моделей на тестовой выборке, сравнение качества работы лучшей модели на тренировочной и тестовой выборках.

На этапе предварительной обработки данных необходимо обеспечить корректную работу моделей. Данный процесс выполняется после разделения на тренировочную и тестовую выборки.

Многие алгоритмы машинного обучения работают лучше или сходятся быстрее, когда функции находятся в относительно одинаковом масштабе или близки к нормальному распределению. Поэтому нам будет необходимо выполнить нормализацию входных данных. При этом для категориального признака «Угол нашивки, град», принимающего всего 2 значения (0 и 90), используем `OrdinalEncoder`, а для остальных признаков применим `MinMaxScaler`, что приведет соответствующие значения в диапазон от 0 до 1.

Предварительную обработку данных будет необходимо также повторить в приложении. В связи с этим реализуем предварительную обработку с помощью `ColumnTransformer`. Значения выходных переменных при этом мы не трансформируем.

Для обеспечения статистической устойчивости метрик модели используем перекрестную проверку с помощью функции `cross_validate` (библиотека `sklearn`). При этом, выборка разбивается определенное количество раз на тестовую и валидационную. Модель обучается на тестовой выборке, затем выполняется расчет метрик качества на валидационной. В результате мы получаем средние метрики качества для всех валидационных выборок.

Поиск гиперпараметров по сетке реализуем с помощью класса `GridSearchCV` со встроенной перекрестной проверкой (библиотека `sklearn`), который поможет выполнить обучение и определить лучшие комбинации гиперпараметров.

При построении оптимальной модели для прогнозирования каждого из целевых признаков будем также использовать ряд метрик качества, применимых для задач регрессии:

- R2 (или коэффициент детерминации) измеряет долю дисперсии, объясненную моделью, в общей дисперсии целевой переменной. Фактически, данная мера качества — это нормированная среднеквадратичная ошибка. Если она близка к единице, то модель хорошо объясняет данные, если же она близка к

нулю, то прогнозы сопоставимы по качеству с константным предсказанием. R² в норме принимает положительные значения. Этую метрику надо максимизировать. Отрицательное значение коэффициента детерминации означает плохую объясняющую способность модели.

- RMSE (Root Mean Squared Error, корень из средней квадратичной ошибки) принимает значения в тех же единицах, что и целевая переменная. Грубые ошибки становятся заметнее за счет того, что ошибку прогноза мы возводим в квадрат. Чувствительна к выбросам.

- MAE (Mean Absolute Error, средняя абсолютная ошибка) принимает значения в тех же единицах, что и целевая переменная. Каждая ошибка вносит свой вклад в MAE пропорционально абсолютному значению ошибки.

- max error (или максимальная ошибка) данной модели в единицах измерения целевой переменной.

2. Практическая часть

2.1. Разбиение и предобработка данных

2.1.1. Для прогнозирования модуля упругости при растяжении

Признаки датасета разделены на входные и выходные, а строки - на тренировочную и тестовую выборки. Размерности новых множеств, полученные в процессе преобразования с помощью функции train_test_split(), выглядят следующим образом:

- X1_train: (700, 11), y1_train: (700, 1);
- X1_test: (300, 11), y1_test: (700, 1).

Описательная статистика входных признаков до и после предобработки изображена на рисунке 5, выходного признака - на рисунке 6.

	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
count	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000
min	0.54739	1784.48225	2.43691	33.62419	15.69589	173.97391	1.66800	41.04828	0.00000	0.03764	20.57163
max	5.59174	2192.73878	1649.41571	192.85170	28.90747	403.65286	1288.69184	386.90343	90.00000	14.03322	92.96349
mean	2.94386	1972.28652	738.62762	112.11924	22.17906	286.44956	481.80588	216.83848	44.61429	6.88038	57.40327
std	0.90219	73.14833	326.13059	28.05646	2.33509	40.64510	278.25359	58.10805	45.03052	2.59097	12.03662
	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	Шаг нашивки	Плотность нашивки	Угол нашивки, град
count	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000
min	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
max	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
mean	0.47508	0.46002	0.44699	0.49297	0.49072	0.48971	0.37306	0.50828	0.48892	0.50878	0.49571
std	0.17885	0.17917	0.19802	0.17620	0.17675	0.17696	0.21620	0.16801	0.18513	0.16627	0.50034

Рисунок 5 - Описательная статистика входных признаков до и после предобработки для прогнозирования модуля упругости при растяжении

```

count 700.00000
min 64.05406
max 82.23760
mean 73.39876
std 3.12858

```

Рисунок 6 - Описательная статистика выходного признака для прогнозирования модуля упругости при растяжении

2.1.2. Для прогнозирования прочности при растяжении

Признаки датасета разделены на входные и выходные, а строки - на тренировочную и тестовую выборки. Размерности новых множеств выглядят следующим образом:

- X2_train: (700, 11), y2_train: (700, 1);
- X2_test: (300, 11), y2_test: (700, 1).

Описательная статистика входных признаков до и после предобработки изображена на рисунке 7, выходного признака - на рисунке 8.

	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
count	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000
min	0.54739	1784.48225	2.43691	33.62419	15.69589	173.97391	1.66800	41.04828	0.00000	0.03764	20.57163
max	5.59174	2192.73878	1649.41571	192.85170	28.90747	403.65286	1288.69184	386.90343	90.00000	14.03322	92.96349
mean	2.94386	1972.28652	738.62762	112.11924	22.17906	286.44956	481.80588	216.83848	44.61429	6.88038	57.40327
std	0.90219	73.14833	326.13059	28.05646	2.33509	40.64510	278.25359	58.10805	45.03052	2.59097	12.03662
	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	Шаг нашивки	Плотность нашивки	Угол нашивки, град
count	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000
min	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
max	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
mean	0.47508	0.46002	0.44699	0.49297	0.49072	0.48971	0.37306	0.50828	0.48892	0.50878	0.49571
std	0.17885	0.17917	0.19802	0.17620	0.17675	0.17696	0.21620	0.16801	0.18513	0.16627	0.50034

Рисунок 7 - Описательная статистика входных признаков до и после предобработки для прогнозирования прочности при растяжении

count	700.00000
min	1071.12375
max	3848.43673
mean	2469.10920
std	493.53174

Рисунок 8 - Описательная статистика выходного признака для прогнозирования прочности при растяжении

2.1.3. Для прогнозирования соотношения матрица-наполнитель

Признаки датасета разделены на входные и выходные, а строки - на тренировочную и тестовую выборки. Размерности новых множеств выглядят следующим образом:

- X3_train: (700, 11), y3_train: (700, 1);
- X3_test: (300, 11), y3_test: (700, 1).

Описательная статистика входных признаков до и после предобработки изображена на рисунке 9, выходного признака - на рисунке 10.

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
700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000
1784.48225	2.43691	33.62419	15.69589	173.97391	1.66800	64.05406	1071.12375	41.04828	0.00000	0.03764	20.57163
2192.73878	1649.41571	192.85170	28.90747	403.65286	1288.69184	82.23760	3848.43673	386.90343	90.00000	14.03322	92.96349
1972.28652	738.62762	112.11924	22.17906	286.44956	481.80588	73.39876	2469.10920	216.83848	44.61429	6.88038	57.40327
73.14833	326.13059	28.05646	2.33509	40.64510	278.25359	3.12858	493.53174	58.10805	45.03052	2.59097	12.03662

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	Шаг нашивки	Плотность нашивки	Угол нашивки, град
700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000	700.00000
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
0.46002	0.44699	0.49297	0.49072	0.48971	0.37306	0.51391	0.50336	0.50828	0.48892	0.50878	0.49571
0.17917	0.19802	0.17620	0.17675	0.17696	0.21620	0.17206	0.17770	0.16801	0.18513	0.16627	0.50034

Рисунок 9 - Описательная статистика входных признаков до и после предобработки для прогнозирования соотношения матрица-наполнитель

```

count    700.00000
min     0.54739
max     5.59174
mean    2.94386
std     0.90219

```

Рисунок 10 - Описательная статистика выходного признака для прогнозирования соотношения матрица-наполнитель

2.2. Разработка и обучение моделей

2.2.1. Для прогнозирования модуля упругости при растяжении

Для подбора оптимальной модели для прогнозирования модуля упругости при растяжении будем использовать следующие модели:

- LinearRegression (линейная регрессия, раздел 1.2.1);
- Lasso (Лассо-регрессия, раздел 1.2.2);
- SVR (метод опорных векторов, раздел 1.2.3);
- KNeighborsRegressor (метод k-ближайших соседей, раздел 1.2.4);
- DecisionTreeRegressor (деревья решений, раздел 1.2.5);

- RandomForestRegressor (случайный лес, раздел 1.2.6);
- GradientBoostingRegressor (градиентный бустинг, раздел 1.2.7).

Для определения нижней границы качества рабочей модели будем использовать DummyRegressor из библиотеки sklearn (стратегия использования - с прогнозированием среднего значения обучающей выборки). Данный регрессор полезен в качестве простого базиса для сравнения с другими (реальными) регрессорами.

Метрики работы отобранных моделей с гиперпараметрами по умолчанию, полученные с помощью перекрестной проверки на тренировочной выборке, отображены на рисунке 11.

	R2	RMSE	MAE	max_error
DummyRegressor	-0.019376	-3.126837	-2.510495	-7.798105
LinearRegression	-0.018532	-3.123936	-2.502366	-8.098392
Lasso	-0.019376	-3.126837	-2.510495	-7.798105
SVR	-0.039011	-3.153628	-2.514390	-8.311025
KNeighborsRegressor	-0.227280	-3.424991	-2.724152	-8.703162
DecisionTreeRegressor	-1.034156	-4.403633	-3.589790	-11.822403
GradientBoostingRegressor	-0.098848	-3.242846	-2.593079	-8.632489
RandomForestRegressor	-0.075323	-3.208305	-2.555245	-8.380008

Рисунок 11 - Результативность моделей с гиперпараметрами по умолчанию

Коэффициент детерминации R2 близок к 0 для линейных моделей (LinearRegression и Lasso), а также метода опорных векторов (SVR). Практически все используемые метрики используемых моделей находятся на одном уровне с метриками базовой модели DummyRegressor. Методы k-ближайших соседей и Дерево решений показали еще более низкие результаты. Алгоритм Случайный лес и Градиентный бустинг отработали лучше, чем 2 вышеупомянутых метода, но несколько хуже, чем линейные модели. Так, ни одна из выбранных моделей не оказалась подходящей для прогнозирования выходных переменных.

После выполнения подбора гиперпараметров по сетке с перекрестной проверкой, получили новые метрики, значения которых приведены на рисунке 12.

	R2	RMSE	MAE	max_error
Lasso(alpha=0.01)	-0.013670	-3.116889	-2.501700	-7.985118
SVR(C=0.5)	-0.021235	-3.127617	-2.501344	-8.154217
KNeighborsRegressor(n_neighbors=91)	-0.009123	-3.110492	-2.497003	-7.900511
DecisionTreeRegressor(max_depth=2, max_features=5, min_samples_split=3, random_state=42)	-0.024121	-3.135466	-2.496037	-8.292868
GradientBoostingRegressor(learning_rate=0.01, loss='absolute_error', min_samples_split=10, random_state=42)	-0.014217	-3.118909	-2.496307	-7.994517
RandomForestRegressor(bootstrap=False, criterion='absolute_error', max_features=1, n_estimators=50, random_state=42)	-0.023475	-3.134199	-2.484176	-8.293257

Рисунок 12 — Результативность моделей после подбора гиперпараметров

Таким образом, проведенный подбор гиперпараметров несколько улучшил прогнозирование всех используемых моделей. Однако, выбранные модели все также не могут найти зависимости. Лучшая модель из представленных - KNeighborsRegressor, по показателям опередила линейные модели, однако дает коэффициент детерминации близкий к нулю, незначительно более высокий, чем DummyRegressor. Подбор параметров для алгоритмов Градиентный бустинг и Случайный лес положительно повлиял на результативность данных моделей, вместе с тем, этот процесс оказался самым затратным по времени.

На рисунке 13 приведена визуализация прогнозирования модели KNeighborsRegressor на тестовой выборке.

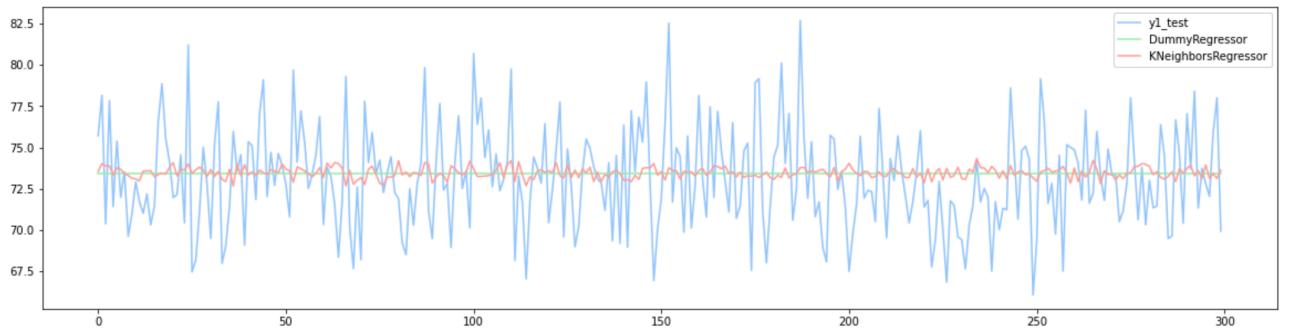


Рисунок 13 — Визуализация прогнозирования модели KNeighborsRegressor

Результативность работы моделей KNeighborsRegressor и DummyRegressor на тестовом множестве и их сравнение отображены на рисунке 14. Полученная модель с подобранными гиперпараметрами работает несколько

хуже базовой. Отрицательное значение коэффициента детерминации означает плохую объясняющую способность выбранной модели.

	R2	RMSE	MAE	max_error
DummyRegressor	-0.007651	-3.082670	-2.479138	-9.283290
KNeighborsRegressor	-0.012047	-3.089388	-2.490524	-8.909661

Рисунок 14 - Метрики работы лучшей модели на тестовом множестве

2.2.2. Для прогнозирования прочности при растяжении

Для подбора оптимальной модели для прогнозирования модуля упругости при растяжении будем использовать следующие модели:

- LinearRegression (линейная регрессия, раздел 1.2.1);
- Lasso (Лассо-регрессия, раздел 1.2.2);
- SVR (метод опорных векторов, раздел 1.2.3);
- KNeighborsRegressor (метод k-ближайших соседей, раздел 1.2.4);
- DecisionTreeRegressor (деревья решений, раздел 1.2.5);
- RandomForestRegressor (случайный лес, раздел 1.2.6);
- GradientBoostingRegressor (градиентный бустинг, раздел 1.2.7).

Для определения нижней границы качества рабочей модели будем также использовать DummyRegressor из библиотеки sklearn (стратегия использования - с прогнозированием среднего значения обучающей выборки). Данный регрессор полезен в качестве простого базиса для сравнения с другими (реальными) регрессорами.

Метрики работы отобранных моделей с гиперпараметрами по умолчанию, полученные с помощью перекрестной проверки на тренировочной выборке, отображены на рисунке 15.

	R2	RMSE	MAE	max_error
DummyRegressor	-0.022944	-493.539876	-391.010975	-1281.791709
LinearRegression	-0.014804	-491.329446	-391.262712	-1305.947015
Lasso	-0.011470	-490.565450	-390.216167	-1301.382442
SVR	-0.020978	-493.091014	-390.520034	-1278.945635
KNeighborsRegressor	-0.197623	-533.145743	-424.397300	-1408.337527
DecisionTreeRegressor	-1.097452	-700.282012	-561.980002	-1784.349498
GradientBoostingRegressor	-0.033926	-496.051787	-398.082126	-1274.138037
RandomForestRegressor	-0.036206	-496.315183	-393.991316	-1325.856363

Рисунок 15 — Результативность моделей с гиперпараметрами по умолчанию

Коэффициент детерминации R2 практически всех моделей также близок к 0 и принимает отрицательное значение. Метод Дерево решений снова показал самый низкий результат. Линейные методы и SVR показали лучшие результаты. Метрики алгоритмов LinearRegression и Lasso оказались даже несколько лучше, чем у DummyRegressor. Немного хуже линейных моделей отработали методы Дерево решений, Градиентный бустинг и Случайный лес.

После выполнения подбора гиперпараметров по сетке с перекрестной проверкой, получили новые метрики, значения которых приведены на рисунке 16.

	R2	RMSE	MAE	max_error
Lasso(alpha=1)	-0.011470	-490.565450	-390.216167	-1301.382442
SVR(C=0.01, degree=5, kernel='poly')	0.001648	-487.500183	-386.090173	-1287.643454
KNeighborsRegressor(n_neighbors=67)	-0.015582	-491.711317	-389.708036	-1286.796521
DecisionTreeRegressor(criterion='poisson', max_depth=3, max_features=10, min_samples_split=3, random_state=42, splitter='random')	-0.014634	-491.520067	-388.357164	-1282.857828
GradientBoostingRegressor(loss='absolute_error', n_estimators=75, random_state=42)	0.002556	-487.325938	-386.488031	-1250.025205
RandomForestRegressor(bootstrap=False, criterion='poisson', max_depth=4, max_features=1, n_estimators=50, random_state=42)	-0.015398	-491.694360	-388.959919	-1278.428575

Рисунок 16 — Результативность моделей после подбора гиперпараметров

Проведенный подбор гиперпараметров улучшил прогнозирование всех используемых моделей. Лучшие модели из представленных - Градиентный бустинг и SVR, по показателям опередили линейные модели. При этом R2 данных моделей не только находятся выше нижней границы качества рабочей модели, но также еще принимают положительные значения. Несмотря на достаточно

продолжительный по времени процесс подбора гиперпараметров, алгоритм GradientBoostingRegressor положительно повлиял на результативность данной модели.

На рисунке 17 приведена визуализация прогнозирования модели GradientBoostingRegressor на тестовой выборке. Результаты прогнозирования модели Градиентный бустинг с подобранными гиперпараметрами достаточно далеки от значений выходных переменных. Однако, результаты выглядят на порядок лучше, чем те, что были получены с помощью алгоритма KNeighborsRegressor для прогнозирования модуля упругости при растяжении.

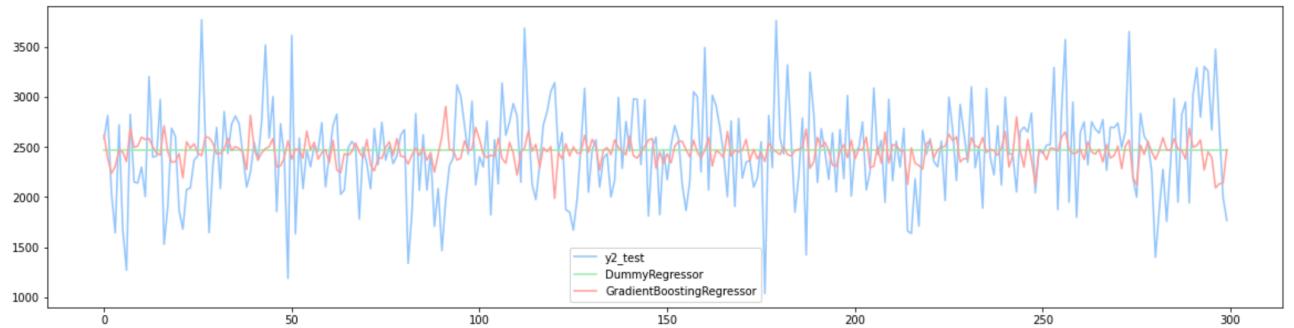


Рисунок 17 — Визуализация прогнозирования модели GradientBoostingRegressor

Результативность работы моделей GradientBoostingRegressor и DummyRegressor на тестовом множестве и их сравнение отображены на рисунке 18. Теперь полученная модель с подобранными гиперпараметрами работает хуже базовой, а R2 вновь принимает отрицательное значение коэффициента детерминации, что означает плохую объясняющую способность выбранной модели.

	R2	RMSE	MAE	max_error
DummyRegressor	-0.000928	-464.631542	-363.886617	-1432.252593
GradientBoostingRegressor	-0.083697	-483.460637	-377.439793	-1385.473615

Рисунок 18 - Метрики работы лучшей модели на тестовом множестве

2.3. Тестирование моделей

Сравним ошибку каждой из отобранных моделей на тренировочной и тестовой выборках. Так, лучшей моделью для прогнозирования модуля упругости при растяжении выбрана KNeighborsRegressor (`n_neighbors=91`). Метрики качества ее работы отображены на рисунке 19.

	R2	RMSE	MAE	max_error
Модуль упругости / train	0.027593	-3.082905	-2.469592	-9.220760
Модуль упругости / test	-0.012047	-3.089388	-2.490524	-8.909661

Рисунок 19 - Метрики работы лучшей модели для прогнозирования модуля упругости при растяжении

R2 выбранного алгоритма имеет лучший показатель при прогнозировании на тренировочной выборке. В связи с тем, что построенная модель лучше объясняет примеры из обучающей выборки, но несколько хуже работает на примерах, не участвовавших в обучении, возможно имеет место эффект переобучения. Однако, даже на тренировочной выборке модель не нашла искомых закономерностей во входных переменных.

Если модуль упругости при растяжении лежит в диапазоне от 64.05 до 82.24, то алгоритм KNeighborsRegressor делает предсказание с точностью ± 8.91 (значение `max_error`). Таким образом, данный метод не будет полезен для применения в реальных условиях.

Лучшей моделью для прогнозирования прочности при растяжении выбрана GradientBoostingRegressor (`loss='absolute_error'`, `n_estimators=75`). Метрики качества ее работы отображены на рисунке 20.

	R2	RMSE	MAE	max_error
Прочность при растяжении / train	0.249254	-427.318062	-315.244670	-1346.296578
Прочность при растяжении / test	-0.083697	-483.460637	-377.439793	-1385.473615

Рисунок 20 - Метрики работы лучшей модели для прогнозирования прочности при растяжении

Как и в случае с предыдущей моделью, коэффициент детерминации алгоритма Градиентный бустинг имеет лучший показатель при прогнозировании на тренировочной выборке, а значит имеется основание предполагать, что модель переобучилась. На тренировочной выборке модель также не нашла искомых закономерностей во входных переменных.

Если модуль прочности при растяжении лежит в диапазоне от 1071.12 до 3848.44, то алгоритм GradientBoostingRegressor делает предсказание с точностью ± 1385.47 (значение max_error). Таким образом, данный метод также не будет полезен для применения в реальных условиях.

2.4. Разработка нейронной сети для прогнозирования соотношения матрица-наполнитель

Построим нейронную сеть для прогнозирования соотношения матрица-наполнитель. Для определения нижней границы качества модели нейросети будем также использовать DummyRegressor (стратегия использования - с прогнозированием среднего значения обучающей выборки).

Для начала построим нейронную сеть с помощью класса MLPRegressor (библиотека sklearn). Архитектура модели многослойного персептрана представлена на рисунке 21.

```
model3_NN = MLPRegressor(hidden_layer_sizes = (24, 48, 48, 48, 24, 24),
                         activation = 'relu',
                         solver='adam',
                         max_iter=400,
                         early_stopping = True,
                         validation_fraction = 0.3,
                         random_state=42,
                         verbose=True
)
```

Рисунок 21 - Архитектура модели нейросети MLPRegressor

Нейронная сеть обучилась за 51 итерацию. График обучения модели приведен на рисунке 22. Визуализация прогнозирования модели приведена на рисунке 23.

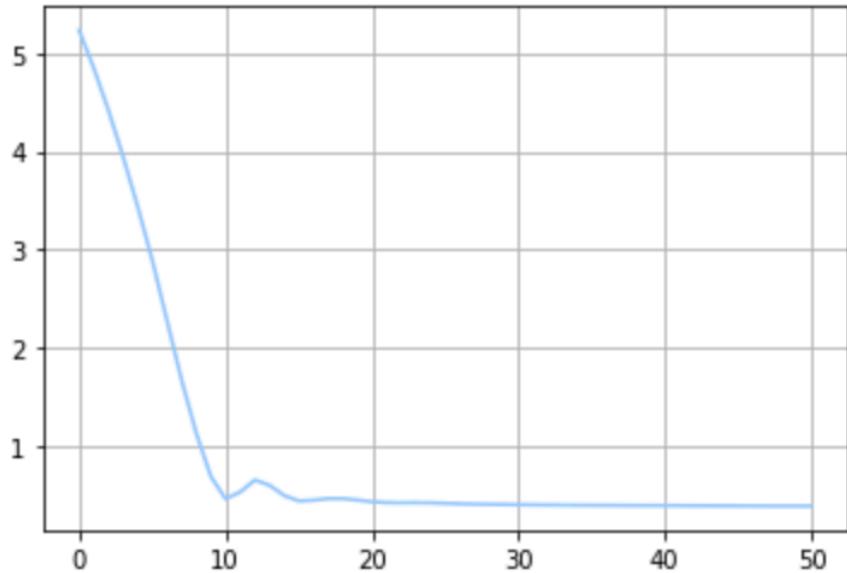


Рисунок 22 — График обучения нейросети MLPRegressor

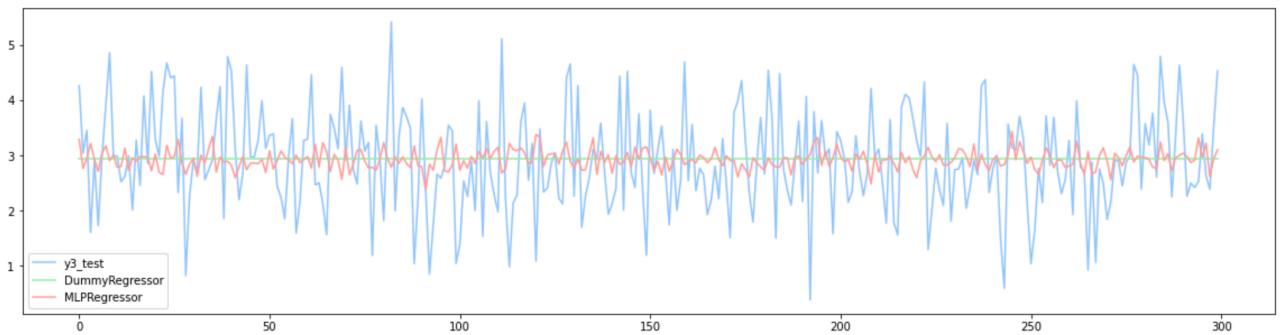


Рисунок 23 - Визуализация прогнозирования нейросети MLPRegressor

Результативность работы моделей нейронной сети и DummyRegressor на тестовом множестве и их сравнение отображены на рисунке 24. Полученные значения говорят о том, что модель MLPRegressor работает несколько хуже базовой и имеет плохую объясняющую способность.

	R2	RMSE	MAE	max_error
DummyRegressor	-0.000744	-0.924041	-0.739327	-2.554458
MLPRegressor	-0.043173	-0.943426	-0.753366	-2.633232

Рисунок 24 — Метрики работы нейросети MLPRegressor на тестовом множестве

Далее построим нейронную сеть с помощью класса keras.Sequential (библиотека tensorflow). Параметры новой нейронной сети будут включать входной и выходной слои (для 12 и 1 признаков соответственно), 10 скрытых слоев (от 24 до 48 нейронов на каждом слое), активационную функцию скрытых слоев relu, оптимизатор Adam, loss-функцию MeanAbsoluteError. Кроме того, обучение нейросети будет состоять из 100 эпох, для проверочной выборки будем использовать 30% имеющихся в тренировочной выборке переменных. Архитектура данной нейронной сети представлена на рисунке 25.

Layer (type)	Output Shape	Param #
<hr/>		
dense_1 (Dense)	(None, 24)	312
dense_2 (Dense)	(None, 24)	600
dense_3 (Dense)	(None, 24)	600
dense_4 (Dense)	(None, 36)	900
dense_5 (Dense)	(None, 36)	1332
dense_6 (Dense)	(None, 36)	1332
dense_7 (Dense)	(None, 36)	1332
dense_8 (Dense)	(None, 24)	888
dense_9 (Dense)	(None, 48)	1200
dense_10 (Dense)	(None, 24)	1176
out (Dense)	(None, 1)	25
<hr/>		
Total params:	9,697	
Trainable params:	9,697	
Non-trainable params:	0	

Рисунок 25 — Архитектура нейронной сети Sequential

График обучения данной модели нейросети представлен на рисунке 26.

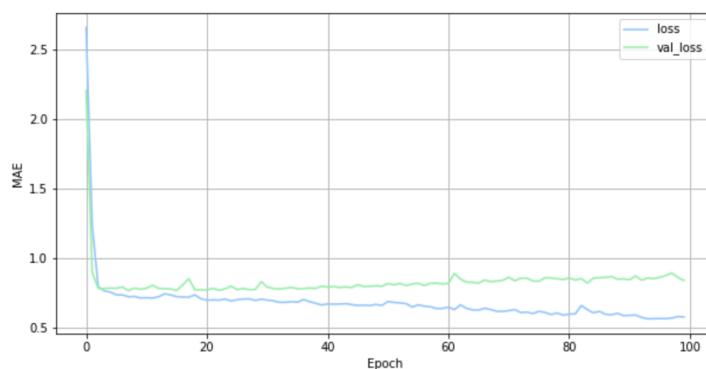


Рисунок 26 - График обучения нейронной сети Sequential

Данные на графике говорят о том, что нейронная сеть начинает переобучаться (значение loss на обучающем наборе данных уменьшается, а на валидационном - растет с раннего этапа обучения). Так, для улучшения качества нейросети воспользуемся возможностью остановить обучение модели ранней остановкой. Для этого возьмем нейросеть с идентичной архитектурой, но воспользуемся методом EarlyStopping, который отслеживает значение val_loss. График обучения оптимизированной нейронной сети представлен на рисунке 27.

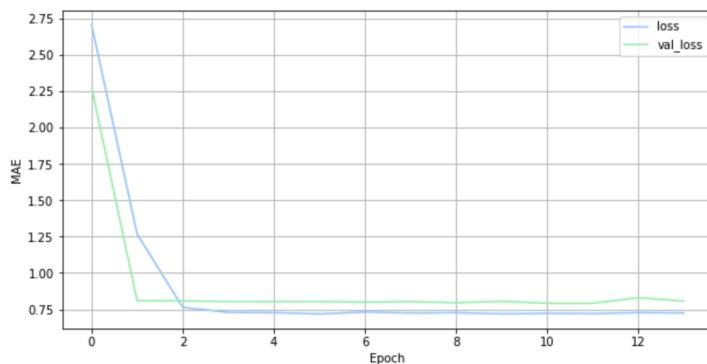


Рисунок 27 - График обучения нейронной сети Sequential с применением EarlyStopping

Попробуем также применить метод добавления в архитектуру нейросети Dropout-слоев, который поможет достичь эффекта уменьшения степени специализации отдельных нейронов и повышения качества обучения при сохранении общего числа нейронов сети. Снова построим модель идентичной архитектуры, и добавим после первого, третьего, пятого, седьмого и девятого скрытых слоев Dropout-слой со значением параметра равным 0.2 (всего 5). График обучения вновь собранной сети представлен на рисунке 28.

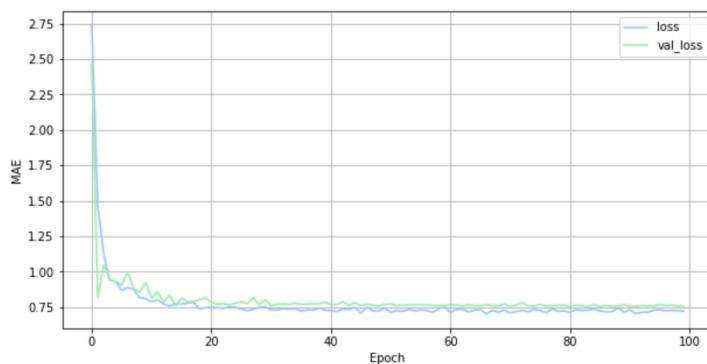


Рисунок 28 — График обучения нейронной сети Sequential с Dropout-слоями

Результативность работы моделей собранных нейронных сетей и DummyRegressor на тестовом множестве и их сравнение отображены на рисунке 29.

	R2	RMSE	MAE	max_error
DummyRegressor	-0.000744	-0.924041	-0.739327	-2.554458
Sequential	-0.254183	-1.034452	-0.837282	-3.580135
Sequential with EarlyStopping	-0.069018	-0.955042	-0.761271	-2.711501
Sequential with DropoutLayers	0.000587	-0.923427	-0.735780	-2.554991

Рисунок 29 - Метрики работы моделей нейросетей Sequential на тестовом множестве

Самая оптимальная модель из представленных, обладающая лучшей обобщающей способностью и меньшими значениями ошибок на тестовом множестве - нейронная сеть Sequential с Dropout-слоями. При этом R2 данной модели не только находится выше нижней границы качества рабочей модели, но также еще и принимают положительные значения.

Таким образом, лучшей моделью для прогнозирования соотношения матрица-наполнитель выбрана Sequential с Dropout-слоями. Сравним ошибку данной модели на тренировочной и тестовой выборках. Метрики качества ее работы отображены на рисунке 30.

	R2	RMSE	MAE	max_error
Соотношение матрица-наполнитель / train	-0.002292	-0.902582	-0.720672	-2.707118
Соотношение матрица-наполнитель / test	0.000587	-0.923427	-0.735780	-2.554991

Рисунок 30 - Метрики работы модели нейросети Sequential с Dropout-слоями для прогнозирования соотношения матрица-наполнитель

2.5. Разработка приложения

Веб-приложение будет разработано в редакторе исходного кода Visual Studio Code с помощью языка Python и фреймворка Flask. Данное приложение должно будет выдавать прогноз соотношения матрица-наполнитель.

Функционал приложения будет включать ввод входных переменных, отображение рекомендательных инструкций по заполнению корректных значений признаков, частичную верификацию введенных параметров (проверка введенных параметров на вещественность), загрузку сохраненной модели (нейросеть Sequential с Dropout-слоями), получение и отображение прогноза выходной переменной.

2.6. Создание удаленного репозитория

В рамках выполнения данного исследования на GitHub был создан репозиторий (адрес: https://github.com/Eu9EN3/vkr_). На нем были размещены результаты работы: код исследования, код приложения, разработанные модели, пояснительная записка и презентация (в электронном виде).

Заключение

Проведенное исследование строилось на базе кейса, основанного на реальных производственных задачах Центра НТИ «Цифровое материаловедение: новые материалы и вещества» (структурное подразделение МГТУ им. Н.Э. Баумана). В процессе решения задачи анализа имеющихся данных были применены полученные в ходе прохождения курса знания, однако, эффективная модель для прогноза ряда конечных свойств получаемых композиционных материалов так и не была найдена.

В качестве возможных путей решения данной задачи, которые смогут значительно повлиять на улучшение объясняющей способности используемых алгоритмов машинного обучения, можно выделить получение более детальной информации об имеющихся зависимостях между признаками датасета, необходимость применения ряда других методов предобработки данных, более глубокий подход к поиску гиперпараметров и подбору архитектуры для построения нейросетей и параметров их обучения, проведение консультаций со специалистами по машинному обучению и экспертами в предметной области, а также получение большего опыта в решении реальных бизнес-задач.

Библиографический список

1. Эндрю Траск. Грокаем глубокое обучение. - СПб.: Питер, 2019. - 352 с.
2. Любанович Билл. Простой Python. Современный стиль программирования. 2-е изд. - СПб.: Питер, 2021. - 592 с.
3. Дауни Аллен. Основы Python. Научитесь думать как программист. - Москва: Манн, Иванов и Фербер, 2021. - 304 с.
4. Верон Орельен. Прикладное машинное обучение с помощью Scikit-Learn и TensorFlow: концепции, инструменты и техники для создания интеллектуальных систем. - СПб.: ООО «Альфа-книга», 2018. - 688 с.
5. Силен Дэви, Мейсман Арно, Али Мохамед. Основы Data Science и Big Data. Python и наука о данных. – СПб.: Питер, 2017. – 336 с.
6. Грас Д. Data Science. Наука о данных с нуля: Пер. с англ. - 2-е изд., перераб. и доп. - СПб.: БХВ-Петербург, 2021. - 416 с.
7. Документация по языку программирования Python: – <https://docs.python.org/3.8/index.html>.
8. Документация по библиотеке numpy: – <https://numpy.org/doc/1.22/user/index.html#user>.
9. Документация по библиотеке pandas: – https://pandas.pydata.org/docs/user_guide/index.html#user-guide.
10. Документация по библиотеке sklearn: – https://scikit-learn.org/stable/user_guide.html.
11. Документация по библиотеке matplotlib: – <https://matplotlib.org/stable/users/index.html>.
12. Документация по библиотеке seaborn: – <https://seaborn.pydata.org/tutorial.html>.
13. Документация по библиотеке keras: – <https://keras.io/api/>.
14. Руководство по быстрому старту в flask: – <https://flask-russian-docs.readthedocs.io/ru/latest/quickstart.html>.
15. Полное руководство по линейной регрессии в Scikit-Learn: - <https://pythonru.com/uroki/linear-regression-sklearn>.

Приложение А. Скриншот разработанного приложения

На рисунке 31 отображено основное окно веб-приложения, демонстрирующее имеющийся функционал, после осуществления ввода корректных значений признаков и получения прогноза соотношения матрица-наполнитель.

① 127.0.0.1:5000

Прогнозирование соотношения матрица-наполнитель

Плотность, кг/м3 (введите значение от 1700 до 2300)

Модуль упругости, ГПа (введите значение от 2 до 2000)

Количество отвердителя, м.% (введите значение от 17 до 200)

Содержание эпоксидных групп,%_2 (введите значение от 14 до 34)

Температура вспышки, С_2 (введите значение от 100 до 414)

Поверхностная плотность, г/м2 (введите значение от 0 до 1400)

Модуль упругости при растяжении, ГПа (введите значение от 64 до 83)

Прочность при растяжении, МПа (введите значение от 1036 до 3849)

Потребление смолы, г/м2 (введите значение от 33 до 414)

Угол нашивки, град (введите значение - 0 или 90)

Шаг нашивки (введите значение от 0 до 15)

Плотность нашивки (введите значение от 0 до 104)

Соотношение матрица-наполнитель - [2.8658552]

Рисунок 31 - Результат работы модели для прогнозирования соотношения матрица-наполнитель