МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

по курсу

«Data Science»

Тема: «Прогнозирование конечных свойств новых материалов (композиционных материалов)»

Слушатель Бойко Татьяна Сергеевна

Москва, 2023

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. Они могут быть созданы из различных типов материалов, таких как полимеры, металлы, керамика и углеродные материалы, и могут иметь различные формы, включая листы, волокна, маты и пены. Основным преимуществом композиционных материалов является их высокая прочность и легкость, что делает их особенно полезными в авиационной, автомобильной и аэрокосмической промышленности. Они также обладают высокой коррозионной стойкостью, устойчивостью к высоким температурам и могут иметь высокую степень гибкости и устойчивости к ударам. 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, но данный принцип сохраняется. Ниже перечислены некоторые примеры композиционных материалов:

1. Стеклопластик - это композитный материал, который состоит из стекловолокна, пропитанного эпоксидной смолой. Он обладает высокой прочностью, легкостью и стойкостью к коррозии и используется в автомобильной, судостроительной и аэрокосмической промышленности.

2. Углеродные волокна - это материалы, состоящие из тонких волокон углерода, которые связаны вместе с помощью эпоксидной смолы. Они имеют высокую прочность и легкость и широко используются в производстве автомобилей, самолетов, велосипедов и спортивных товаров.

3. Композитные пены - это материалы, состоящие из полимерных материалов, таких как полиуретан, вспененные с добавлением различных добавок, таких как алюминий или карбонат кальция. Они широко используются в производстве лодок, плотов, ветряных турбин и других изделий, которые требуют высокой прочности и легкости.

4. Композитные материалы на основе арамида - это материалы, состоящие из волокон арамида, таких как Кевлар, связанных вместе с помощью эпоксидной смолы. Они обладают высокой прочностью и стойкостью к ударам и используются в производстве защитной одежды, бронежилетов и других изделий.

Это только несколько примеров композитных материалов, которые могут использоваться в различных отраслях промышленности.

Однако, производство композитных материалов является сложным и требует специального оборудования и навыков. 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 (включая различные методы анализа, такие как рентгеновская дифракция, электронная микроскопия и термический анализ) и прогнозирование характеристик. Суть прогнозирования заключается в симуляции представительного элемента объема композита, на основе данных о характеристиках входящих компонентов (связующего и армирующего компонента).

В прогнозировании могут использоваться компьютерное моделирование и симуляции для оценки свойств материалов на основе их структуры и компонентов. Это позволяет исследовать свойства материалов на молекулярном уровне и улучшить их характеристики.

Также может быть использовано машинное обучение и анализ данных для поиска связей между структурой материала и его свойствами, что позволяет оптимизировать процесс разработки и производства новых материалов.

Независимо от метода, прогнозирование конечных свойств новых материалов является ключевым фактором при их разработке и может привести к созданию материалов с улучшенными свойствами и более широким спектром применения.

Машинное обучение и анализ данных могут быть использованы для поиска связей между структурой композиционных материалов и их свойствами. Этот подход может помочь в определении оптимальных параметров производства и формирования структуры материала для достижения желаемых свойств.

Для этого необходимо создать модель, которая может обучиться на данных о структуре и свойствах материалов, и затем использовать эту модель для прогнозирования свойств новых материалов на основе их структуры.

Примерами методов машинного обучения, которые могут использоваться для поиска связей между структурой композиционных материалов и их свойствами, являются регрессионный анализ, глубокое обучение, методы кластеризации и методы обработки изображений.

В данной работе на входе имеются данные о начальных свойствах компонентов композиционных материалов (количество связующего, наполнителя, температурный режим отверждения и т. д.). На выходе необходимо спрогнозировать ряд конечных свойств получаемых композиционных материалов. Кейс основан на реальных производственных задачах Центра НТИ «Цифровое

материаловедение: новые материалы и вещества» (структурное подразделение МГТУ им. Н.Э. Баумана).

Актуальность: созданные прогнозные модели помогут сократить количество проводимых испытаний, а также пополнить базу данных материалов возможными новыми характеристиками материалов, и цифровыми двойниками новых композитов. Традиционно разработка композитных материалов является долгосрочным процессом, так как из свойств отдельных компонентов невозможно рассчитать конечные свойства композита. Для достижения 4 определенных характеристик требуется большое количество различных комбинированных тестов, что делает насущной задачу прогнозирования успешного решения, снижающего затраты на разработку новых материалов.

1. Аналитическая часть

1.1. Постановка задачи

Для исследовательской работы были даны 2 файла: X_bp.xlsx (с данными о параметрах, состоящий из 1023 строк и 10 столбцов данных) и X_nup.xlsx (данными нашивок, состоящий из 1040 строк и 3 столбцов данных). Для разработки моделей по прогнозу модуля упругости при растяжении, прочности при растяжении и соотношения матрица-наполнитель нужно объединить 2 файла. Объединение по типу INNER, поэтому часть информации (17 строк таблицы X_nup.xlsx), не имеющая соответствующих строк в таблице X_bp.xlsx, будет удалена.

```
# Объединить таблицу по индексу
df = pd.merge(df1, df2, on='Unnamed: 0', how='inner')
df.head()
```

	Unnamed: 0	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	н
0	0	1.857143	2030.0	738.736842	30.00	22.267857	100.000000	210.0	70.0	3000.0	220.0	
1	1	1.857143	2030.0	738.736842	50.00	23.750000	284.615385	210.0	70.0	3000.0	220.0	
2	2	1.857143	2030.0	738.736842	49.90	33.000000	284.615385	210.0	70.0	3000.0	220.0	
3	3	1.857143	2030.0	738.736842	129.00	21.250000	300.000000	210.0	70.0	3000.0	220.0	

Рисунок 1 – Пример начала работы с датасетами

Также необходимо провести разведочный анализ данных, нарисовать гистограммы распределения каждой из переменной, диаграммы boxplot (ящик с усами), попарные графики рассеяния точек.

```
df.hist(figsize = (20,20), color = "Pink")
plt.show()
```

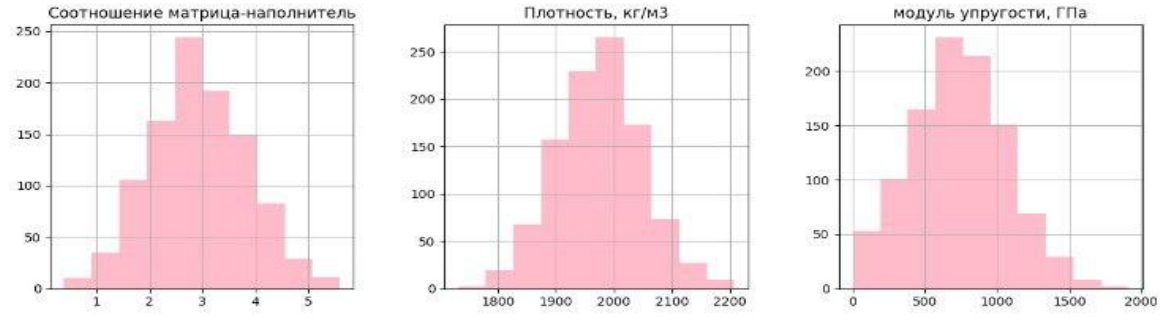

Рисунок 2 – Гистограммы распределения переменных

Для каждой колонки получить среднее, медианное значение, провести анализ и исключение выбросов, проверить наличие пропусков; сделать предобработку: удалить шумы и выбросы, сделать нормализацию и стандартизацию. Обучить несколько моделей для прогноза модуля упругости при растяжении и прочности при растяжении. Написать нейронную сеть, которая будет рекомендовать соотношение матрица-наполнитель. Разработать приложение с графическим интерфейсом, которое будет выдавать прогноз соотношения «матрица-наполнитель». Оценить точность модели на тренировочном и тестовом датасете. Создать репозиторий в GitHub и разместить код исследования. Оформить файл README.

```
ax = fig2.add_subplot(133)
sns.boxplot(y = df['Плотность нашивки'], color='#F5B9FC')
```

```
<AxesSubplot: ylabel='Плотность нашивки'>
```

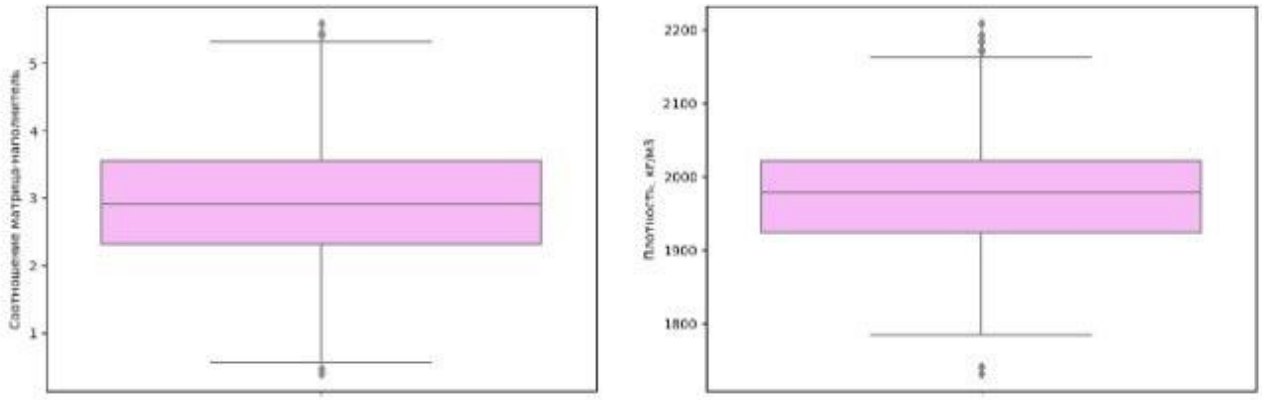

Рисунок 3 – Диаграммы переменных – “ящики с усами” (Boxplot)

1.2. Описание используемых методов

Данная задача относится к задачам регрессии. Машинное обучение с учителем в задачах регрессии используется для нахождения связи между независимыми переменными и зависимой переменной с целью предсказания значения последней по заданным значениям первых. В основе такого обучения лежат алгоритмы, которые пытаются найти оптимальную функцию, связывающую эти переменные. Целью обучения является нахождение наилучшей функции, которая минимизирует ошибку предсказания.

Для решения этой задачи были рассмотрены различные методы (некоторые из них применены):

− Линейная регрессия (Linear regression);

– Лассо регрессия (Lasso);

– Гребневая регрессия (Ridge Regression) ;
– Эластичная сеть (Elastic Net);
– Метод k-ближайших соседей (k-nearest neighbors, KNN);
– Дерево решений (Decision Tree);
– Случайный лес (Random Forest);
– Градиентный бустинг (Gradient Boosting);
– Адаптивный бустинг (AdaBoost);
– Стохастический градиентный спуск (SGD);
– Метод опорных векторов (Support Vector Machines, SVM);
– Многослойный перцептрон (англ. Multi-Layer Perceptron, MLP).

Линейная регрессия (Linear regression) — это алгоритм машинного обучения, основанный на контролируемом обучении, рассматривающий зависимость между одной входной и выходными переменными. Это один из самых простых и широко используемых инструментов статистического моделирования. Он определяет зависимость переменных с помощью линии соответствия. Идея линейной регрессии состоит в том, чтобы построить линейную функцию, которая наилучшим образом описывает зависимость между независимыми и зависимыми переменными. Линейная регрессия предполагает, что между зависимой и независимыми переменными существует линейная связь, т. е. изменение значения независимой переменной на единицу приводит к изменению значения зависимой переменной на фиксированную величину. Для построения модели линейной регрессии используется метод наименьших квадратов (МНК), который заключается в минимизации суммы квадратов разностей между фактическими значениями зависимой переменной и предсказанными значениями.

Достоинства метода: простота реализации; интерпретируемость (позволяет наглядно оценить влияние каждого признака на результат); вычислительная эффективность (имеет линейную сложность, что делает ее вычислительно эффективной для больших объемов данных); хорошее качество при правильной настройке гиперпараметров и использовании правильных признаков.

Недостатки метода: ограничена линейной зависимостью между признаками и целевой переменной, что может привести к недостаточной точности модели в некоторых случаях; чувствительна к выбросам в данных; если признаки сильно коррелируют между собой, то линейная регрессия может давать неустойчивые и неточные результаты (проблема мультиколлинеарности); требует предварительной обработки данных, чтобы избавиться от пропущенных значений, выбросов и других аномалий в данных.

Лассо регрессия (Lasso) – это метод регрессии, который используется для отбора признаков и снижения размерности данных. Он является разновидностью линейной регрессии, которая также минимизирует сумму квадратов отклонений (RSS, англ. Residual Sum of Squares – дословно “сумма квадратов остатков”) между фактическими и прогнозируемыми значениями целевой переменной. Однако, в отличие от обычной линейной регрессии, лассо регрессия также добавляет штраф (L1) за абсолютные значения коэффициентов регрессии.

В частности, лассо регрессия минимизирует функцию потерь, которая состоит из двух частей: суммы квадратов отклонений и суммы абсолютных значений коэффициентов регрессии, умноженных на коэффициент регуляризации. Коэффициент регуляризации позволяет балансировать между двумя целями: минимизацией RSS и минимизацией суммы абсолютных значений коэффициентов. В результате, лассо регрессия предоставляет решение с набором коэффициентов регрессии, которые могут быть нулевыми, что означает, что соответствующие

признаки были исключены из модели. Таким образом, лассо регрессия позволяет выбирать наиболее важные признаки и устранять шумовые или неинформативные признаки, что уменьшает шансы на переобучение и улучшает обобщающую способность модели.

Достоинства метода: позволяет автоматически отбирать наиболее значимые признаки, что может улучшить качество модели и ускорить ее обучение; уменьшает вероятность переобучения при наличии большого количества признаков; позволяет работать с большим количеством признаков, даже если они коррелируют между собой.

Недостатки метода: если все признаки взаимосвязаны между собой, то LASSO может выбрать только один из них, что может привести к потере информации; в результате регуляризации L1 могут быть отброшены некоторые важные признаки, если их коэффициенты в модели слишком малы; может давать нестабильные результаты при наличии шумовых признаков или признаков с очень маленькими весами.

Гребневая регрессия (Ridge Regression) – это метод регуляризации линейной регрессии, который помогает уменьшить переобучение модели и улучшить ее обобщающую способность. Гребневая регрессия добавляет к функции потерь модели штрафную функцию L2, которая представляет собой сумму квадратов весов признаков, умноженных на коэффициент регуляризации α. Таким образом, функция потерь в гребневой регрессии определяется следующим образом:

*Loss function = Sum of squares of residuals + α * Sum of squares of coefficients*

где α - коэффициент регуляризации, который контролирует величину штрафа.

Добавление L2-штрафа к функции потерь приводит к уменьшению весов признаков, но не исключает признаки из модели полностью, как это делает L1-штраф в лассо регрессии.

Достоинства метода: уменьшает переобучение модели и улучшает ее обобщающую способность; позволяет использовать все признаки в модели, но с уменьшенной значимостью; работает хорошо в случае, когда признаки взаимозависимы между собой.

Недостатки метода: не исключает признаки из модели, что может привести к увеличению шума в данных; не подходит для отбора наиболее значимых признаков; требует настройки коэффициента регуляризации α, который может быть сложно подобрать.

Эластичная сеть (Elastic Net) – это метод регрессии, который комбинирует в себе два метода регуляризации - L1 (лассо) и L2 (гребневая). Он был разработан для устранения недостатков каждого метода и получения лучших результатов в задачах регрессии. Этот метод помогает справиться с проблемой мультиколлинеарности в данных и улучшает обобщающую способность модели. L1 регуляризация использует сумму модулей коэффициентов признаков, что приводит к отбору признаков, а L2 регуляризация использует сумму квадратов коэффициентов признаков, что приводит к сжатию коэффициентов к нулю. ElasticNet регрессия комбинирует эти два метода регуляризации, используя линейную комбинацию L1 и L2 штрафов.

В ElasticNet регрессии есть два параметра: alpha и l1_ratio. Параметр alpha контролирует общую интенсивность регуляризации, а параметр l1_ratio контролирует отношение между L1 и L2 регуляризацией.

Достоинства метода: устранение недостатков методов L1 (lasso) и L2 (ridge) регуляризации, позволяя получить более универсальную и точную модель; способность решать задачу отбора признаков и устранять мультиколлинеарность признаков; подходит для работы с большим количеством признаков, так как L1-регуляризация способствует разреженности модели и отбору признаков;

имеется возможность настройки параметров alpha и l1_ratio для получения оптимальной модели.

Недостатки метода: возможно, что при наличии только небольшого числа важных признаков, ElasticNet не сможет найти наилучшую модель и будет переобучаться; необходимость подбора оптимальных значений параметров alpha и l1_ratio, что может потребовать дополнительной работы; ElasticNet может быть более сложным и менее интерпретируемым методом, чем простые линейные модели; если данные имеют ярко выраженную структуру и зависимость между признаками и целевой переменной линейна, более простые методы могут давать лучшие результаты.

Метод k-ближайших соседей (k-nearest neighbors, KNN) – это простой и популярный алгоритм классификации и регрессии в машинном обучении, который используется для предсказания значения целевой переменной на основе близости к ближайшим объектам в обучающем наборе данных. В случае классификации, KNN использует голосование большинства среди k ближайших соседей для определения класса нового объекта. В случае регрессии, KNN использует среднее значение k ближайших соседей для предсказания значения целевой переменной.

Достоинства метода: прост для понимания и реализации; не требует предварительного обучения на большом количестве данных, что делает его полезным в задачах, где данные быстро меняются; универсальность: может использоваться для работы с различными типами данных, включая числовые, категориальные и текстовые данные.

Недостатки метода: чувствителен к выбросам и шуму в данных, что может существенно снизить точность классификации или регрессии; вычислительная сложность: KNN требует постоянного расчета расстояний между объектами, что может быть очень вычислительно сложным при больших наборах данных и большом количестве признаков; проблема выбора параметра k: выбор параметра k

является важным шагом в использовании метода KNN и может существенно влиять на результаты.

Дерево решений (Decision Tree) – это алгоритм машинного обучения, который строит дерево, представляющее собой последовательность решающих правил, используемых для классификации или регрессии данных. Дерево решений представляет собой древовидную структуру, в которой каждый узел представляет собой тест на один из признаков, а каждая ветвь соответствует одному из возможных значений этого признака. Нижние узлы дерева соответствуют терминальным узлам, в которых принимается решение о классификации или регрессии данных. Дерево решений является одним из вариантов решения регрессионной задачи в том случае, когда зависимость в данных не имеет очевидной корреляции.

Достоинства: простота интерпретации: дерево решений может быть легко интерпретировано и объяснено, так как оно представляет собой последовательность простых логических правил; может работать с нечисловыми данными, такими как категориальные или текстовые данные; быстрота работы: может быть быстро построено и использовано для классификации или регрессии больших объемов данных.

Недостатки: наклонность к переобучению: дерево решений может иметь тенденцию к переобучению, особенно если оно имеет слишком много узлов и слишком мало данных для обучения; неустойчивость к шуму; проблема выбора гиперпараметров: выбор оптимальных гиперпараметров (таких как глубина дерева или критерий разделения) может быть сложной задачей, особенно для больших и сложных деревьев.

Случайный лес (Random Forest) — это алгоритм машинного обучения, который использует ансамбль деревьев решений для классификации или регрессии

данных. Основная идея случайного леса заключается в построении множества деревьев решений на разных случайных подвыборках данных и признаков, а затем объединении их прогнозов для получения более точного результата. Процесс построения случайного леса выглядит следующим образом:

1. Из общего набора данных случайным образом выбирается подвыборка.
2. Из этой подвыборки случайным образом выбираются признаки для каждого дерева.
3. Для каждого дерева из подвыборки строится дерево решений с использованием выбранных признаков.
4. Деревья объединяются в ансамбль, и результаты прогнозирования каждого дерева усредняются или принимается решение путем голосования.

Достоинства: высокая точность; устойчивость к переобучению: случайный лес обычно не склонен к переобучению, так как использует множество деревьев решений, каждое из которых обучено на разных случайных подвыборках данных; способность обрабатывать большие объемы данных с высокой скоростью.

Недостатки: сложность интерпретации: объединение прогнозов множества деревьев решений делает случайный лес менее интерпретируемым, чем отдельное дерево решений; затратность на обучение: требуется обучение большого количества деревьев решений на разных подвыборках данных; неустойчивость к шуму.

Градиентный бустинг (Gradient Boosting) - это метод машинного обучения, который используется для решения задач классификации и регрессии. Он основан на идее последовательного построения ансамбля слабых моделей (обычно решающих деревьев), которые корректируют ошибки предыдущих моделей, путем обучения каждой новой модели на остатках (разница между истинными значениями и предсказаниями) предыдущей модели. Каждая новая модель в

ансамбле старается предсказать остатки предыдущих моделей, используя градиентный спуск для минимизации функции потерь.

Достоинства градиентного бустинга: высокая точность предсказаний; работа с различными типами данных, включая категориальные и числовые признаки; возможность работы с пропущенными данными, без необходимости заполнения пропусков; масштабируемость: градиентный бустинг может быть масштабирован для работы с большими объемами данных.

Недостатки градиентного бустинга: требовательность к вычислительным ресурсам: может потребовать значительного объема вычислительных ресурсов, особенно при использовании большого количества слабых моделей и/или больших объемов данных; неустойчивость к выбросам; чувствительность к переобучению: может переобучаться, если используется слишком много деревьев или если данные содержат много шума; необходимость настройки гиперпараметров для достижения лучшей производительности, таких как количество деревьев, глубина деревьев и скорость обучения.

Адаптивный бустинг (AdaBoost) – это алгоритм машинного обучения, который используется для классификации и регрессии. Он основан на идее комбинирования нескольких слабых моделей (например, деревьев решений) для создания более сильной модели. AdaBoost и градиентный бустинг используют похожий принцип комбинации нескольких слабых моделей для создания более сильной модели, но их подход к этому отличается. AdaBoost обучает слабые модели последовательно, при этом каждый следующий классификатор фокусируется на тех объектах, которые были неправильно классифицированы предыдущими моделями. Этот процесс повторяется многократно, с каждым разом увеличивая веса тех объектов, которые были классифицированы неправильно. На выходе из алгоритма каждая модель имеет свой вес, который зависит от ее

точности на обучающих данных. Затем все модели комбинируются в единую модель, взвешивая их весами, чтобы получить итоговую модель, которая дает наиболее точный результат.

Достоинства AdaBoost: высокая точность классификации на сложных данных; нет необходимости в предварительной настройке гиперпараметров; может использоваться для любых типов моделей, которые могут обучаться на обучающих данных.

Недостатки AdaBoost: чувствителен к выбросам в данных; требует больше вычислительных ресурсов, чем простые алгоритмы машинного обучения, такие как линейная регрессия или деревья решений; склонен к переобучению на небольших выборках данных.

Стохастический градиентный спуск (SGD) – это метод оптимизации, используемый для обучения моделей машинного обучения, таких как линейные модели и нейронные сети. Он относится к классу методов градиентного спуска, которые минимизируют функцию потерь, связанную с моделью. В отличие от классического градиентного спуска, который вычисляет градиент функции потерь для всего обучающего набора данных, SGD вычисляет градиент только для одного случайно выбранного наблюдения (экземпляра) из обучающего набора данных. Это делает SGD быстрее и более эффективным для обучения на больших наборах данных. В SGD для каждого наблюдения градиент вычисляется и используется для обновления весов модели, чтобы минимизировать функцию потерь. Этот процесс повторяется для каждого наблюдения в обучающем наборе данных в течение нескольких эпох (итераций). SGD имеет несколько гиперпараметров, которые могут быть настроены, чтобы улучшить процесс обучения, включая размер шага (learning rate), количество эпох и размер пакета (batch size) - количество случайно

выбранных наблюдений, используемых для вычисления градиента на каждой итерации.

Достоинства SGD: обрабатывает обучающие данные по одному случайному примеру за раз, что позволяет обучать модели быстрее, чем с использованием обычного градиентного спуска; также снижается потребление памяти по сравнению с обычным градиентным спуском, который требует хранения всех данных в памяти; SGD может пропустить локальный минимум, который может быть достигнут с помощью обычного градиентного спуска; SGD может дать лучшую сходимость для некоторых задач, особенно для задач с большим количеством признаков.

Недостатки SGD: может быть шумным и неустойчивым, так как градиенты для каждого примера являются стохастическими и могут изменяться на каждой итерации; может потребовать большое количество эпох для достижения сходимости, особенно если выбрано слишком большое значение для размера пакета (batch size); имеет несколько гиперпараметров, которые могут сильно влиять на процесс обучения и могут потребовать тщательной настройки, таких как размер шага (learning rate), количество эпох и размер пакета (batch size); не гарантирована глобальная сходимость: SGD не гарантирует достижение глобального минимума, как обычный градиентный спуск, особенно если функция потерь не выпуклая.

Метод опорных векторов (Support Vector Machines, SVM) – это алгоритм машинного обучения, который используется для задач классификации, регрессии и обнаружения выбросов. Он основан на поиске гиперплоскости, которая наилучшим образом разделяет классы в пространстве признаков. Гиперплоскость – это просто подмножество пространства, которое имеет на одну меньшую размерность, чем пространство, в котором оно находится. Гиперплоскость

выбирается таким образом, чтобы она находилась максимально далеко от ближайших точек обоих классов (или значений целевой переменной). Точки данных, которые расположены ближе всего к гиперплоскости, называются опорными векторами. В задачах регрессии SVM стремится построить модель, которая бы предсказывала значения целевой переменной на основе входных признаков. Для использования SVM в задачах регрессии необходимо выбрать два класса точек, расположенных выше и ниже линии регрессии, которую нужно построить. Затем SVM строит гиперплоскость, которая наилучшим образом разделяет два класса точек, и таким образом формирует линию регрессии. Для предсказания значения для нового объекта SVM смотрит на расстояние между этим объектом и линией регрессии и предсказывает значение в соответствии с этим расстоянием.

Достоинства метода: SVM хорошо работает с многомерными данными и может обрабатывать большие наборы данных; имеет параметр регуляризации, который позволяет контролировать уровень переобучения; может использовать ядерные функции, которые позволяют ему обрабатывать нелинейные данные; может давать хорошие результаты на новых данных, которые не использовались при обучении.

Недостатки метода: чувствительность к выбросам; необходимость настройки параметров, таких как выбор ядра и настройка параметров регуляризации, которые могут сильно влиять на производительность модели и требуют тщательной настройки; интерпретация модели SVM может быть сложной из-за нелинейного преобразования признаков и использования ядерных функций; обучение модели SVM может быть вычислительно сложным для больших наборов данных, особенно при использовании нелинейных ядерных функций.

Многослойный перцептрон (Multi-Layer Perceptron, MLP) – это название для архитектуры нейронной сети и для алгоритма машинного обучения, который

использует эту архитектуру. Принцип работы многослойного перцептрона в задачах регрессии заключается в том, что сначала данные подаются на входной слой сети, затем проходят через один или несколько скрытых слоев, где в каждом нейрон вычисляет своё значение на основе входных данных и весов, которые соединяют нейроны в этом слое. Значения каждого нейрона в скрытом слое передаются на следующий скрытый слой или на выходной слой, где происходит финальный расчёт и выдаётся конечный результат - числовое значение, соответствующее прогнозируемой переменной. Обучение многослойного перцептрона в задачах регрессии заключается в определении оптимальных значений весов нейронов, которые минимизируют среднеквадратическую ошибку между прогнозируемыми значениями и фактическими значениями. Для этого используются различные методы оптимизации, такие как стохастический градиентный спуск (SGD) или его модификации.

Достоинства метода: может моделировать сложные нелинейные зависимости между входными и выходными данными; хорошо справляется с задачами классификации и регрессии, когда данные имеют нелинейную структуру; возможность использования множества функций активации, что позволяет адаптировать модель к разным типам задач; гибкость настройки параметров сети, например, количество скрытых слоев и количество нейронов в каждом слое, что позволяет достичь высокой точности предсказания.

Недостатки метода: подбор оптимальных параметров может быть трудоемким процессом, требующим много времени и вычислительных ресурсов; может быть склонен к переобучению, особенно когда количество нейронов в сети слишком большое или когда данных недостаточно; может быть неэффективным для решения задач, в которых данные имеют линейную структуру, например, когда прогнозируемая переменная линейно зависит от входных данных; не всегда легко

интерпретировать результаты работы сети, то есть понять, каким образом было принято решение и по каким признакам.

Используемые метрики качества моделей

Коэффициент детерминации, также известный как R-квадрат, является мерой качества прогнозной модели в регрессионном анализе. Он описывает, насколько хорошо линейная модель подходит для описания вариации зависимой переменной (той, которую мы пытаемся предсказать) на основе независимых переменных (тех, которые мы используем для предсказания). Коэффициент детерминации может принимать значения от 0 до 1. Значение 0 означает, что модель не объясняет вариацию зависимой переменной, а значение 1 означает, что модель идеально предсказывает зависимую переменную на основе независимых переменных. Если же R-квадрат равен 0,5, модель объясняет лишь 50 процентов дисперсии данных. Чем ближе коэффициент детерминации к 1, тем лучше модель подходит для описания вариации зависимой переменной на основе независимых переменных. Отрицательные значение коэффициента детерминации означают плохую объясняющую способность модели. Однако, не следует использовать R-квадрат как единственную меру качества модели, так как он может быть занижен в случае наличия выбросов или нелинейной зависимости между переменными.

MAE (Mean Absolute Error) показывает среднее абсолютное отклонение прогнозов модели от фактических значений. Для расчета MAE сначала на каждом примере данных вычисляется абсолютная разница между прогнозом модели и фактическим значением. Затем все эти разницы усредняются, чтобы получить единственное число, которое и будет показывать среднюю ошибку модели на данном наборе данных.

MAE определяется формулой:

*MAE = (1/n) * ∑|y_pred - y_true|,*

где y_pred - прогноз модели, y_true - фактическое значение, n - количество примеров в наборе данных.

Чем меньше значение MAE, тем лучше модель способна прогнозировать значения целевой переменной. MAE позволяет оценить точность модели в абсолютных единицах измерения целевой переменной.

MSE (Mean Squared Error) (средняя квадратичная ошибка) принимает значениях в тех же единицах, что и целевая переменная. Чем ближе к нулю MSE, тем лучше работают предсказательные качества модели.

1.3. Разведочный анализ данных

Прежде чем передать данные в работу моделей машинного обучения, необходимо обработать и очистить их. Необработанные данные могут содержать искажения и пропущенные значения и способны привести к неверным результатам.

Цель разведочного анализа - получение первоначальных представлений о характерах распределений переменных исходного набора данных, формирование оценки качества исходных данных (наличие пропусков, выбросов), выявление характера взаимосвязи между переменными с целью последующего выдвижения гипотез о наиболее подходящих для решения задачи моделях машинного обучения.

```
df.describe().T
```

	count	mean	std	min	25%	50%	75%	max
Соотношение матрица-наполнитель	1023.0	2.930366	0.913222	0.389403	2.317887	2.906878	3.552660	5.591742
Плотность, кг/м3	1023.0	1975.734888	73.729231	1731.764635	1924.155467	1977.621657	2021.374375	2207.773481
модуль упругости, ГПа	1023.0	739.923233	330.231581	2.436909	500.047452	739.664328	961.812526	1911.536477
Количество отвердителя, м.%	1023.0	110.570769	28.295911	17.740275	92.443497	110.564840	129.730366	198.953207
Содержание эпоксидных групп,%_2	1023.0	22.244390	2.406301	14.254985	20.608034	22.230744	23.961934	33.000000
Температура вспышки, С_2	1023.0	285.882151	40.943260	100.000000	259.066528	285.896812	313.002106	413.273418
Поверхностная плотность, г/м2	1023.0	482.731833	281.314690	0.603740	266.816645	451.864365	693.225017	1399.542362
Модуль упругости при растяжении, ГПа	1023.0	73.328571	3.118983	64.054061	71.245018	73.268805	75.356612	82.682051
Прочность при растяжении, МПа	1023.0	2466.922843	485.628006	1036.856605	2135.850448	2459.524526	2767.193119	3848.436732
Потребление смолы, г/м2	1023.0	218.423144	59.735931	33.803026	179.627520	219.198882	257.481724	414.590628
Угол нашивки, 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 4 – Описательная статистика датасета

В качестве инструментов разведочного анализа используется: оценка статистических характеристик датасета; гистограммы распределения каждой из переменной; диаграммы ящика с усами; попарные графики рассеяния точек; тепловая карта; описательная статистика для каждой переменной; анализ и полное исключение выбросов (3 повторных итерации); проверка наличия пропусков и дубликатов; коэффициент корреляции признаков.

```
# сумма выбросов по каждому столбцу

df_clean = df.copy()
print(df_clean.shape)
for name in df_clean.columns:
    outlier = boxplot_stats(df_clean[name])
    High = outlier[0]['whishi']
    Low = outlier[0]['whislo']
    print ('Количество выбросов в столбце', name, ': ', len(outlier[0]['fliers']))
    df_clean = df_clean[~((df_clean[name] < Low) | (df_clean[name] > High))]
print(df_clean.shape)
```

```
(1023, 13)
Количество выбросов в столбце Соотношение матрица-наполнитель :  6
Количество выбросов в столбце Плотность, кг/м3 :  9
Количество выбросов в столбце модуль упругости, ГПа :  2
Количество выбросов в столбце Количество отвердителя, м.% :  14
Количество выбросов в столбце Содержание эпоксидных групп,%_2 :  2
Количество выбросов в столбце Температура вспышки, С_2 :  6
Количество выбросов в столбце Поверхностная плотность, г/м2 :  2
Количество выбросов в столбце Модуль упругости при растяжении, ГПа :  5
Количество выбросов в столбце Прочность при растяжении, МПа :  14
Количество выбросов в столбце Потребление смолы, г/м2 :  5
Количество выбросов в столбце Угол нашивки, град :  0
Количество выбросов в столбце Шаг нашивки :  4
Количество выбросов в столбце Плотность нашивки :  22
(932, 13)
```

Рисунок 5 – Начальное количество выбросов

Гистограммы используются для изучения распределений частот значений переменных. Мы видим очень слабую корреляцию между переменными.

После обнаружения выбросов данные, значительно отличающиеся от выборки, будут полностью удалены. Для расчета этих данных мы будем использовать методы трех сигм и межквартильного диапазона.

```
# выбросы после очистки датасета
print(df_clean.shape)
plt.figure(figsize=(15, 30))
i=1
for name in df_clean.columns:
    plt.subplot(5,3,i)
    sns.boxplot(y=df_clean[name], color = 'pink')
    outlier = boxplot_stats(df_clean[name])
    print ('Количество выбросов в столбце', name, ': ', len(outlier[0]['fliers']))
    i +=1
print(df_clean.shape)
```

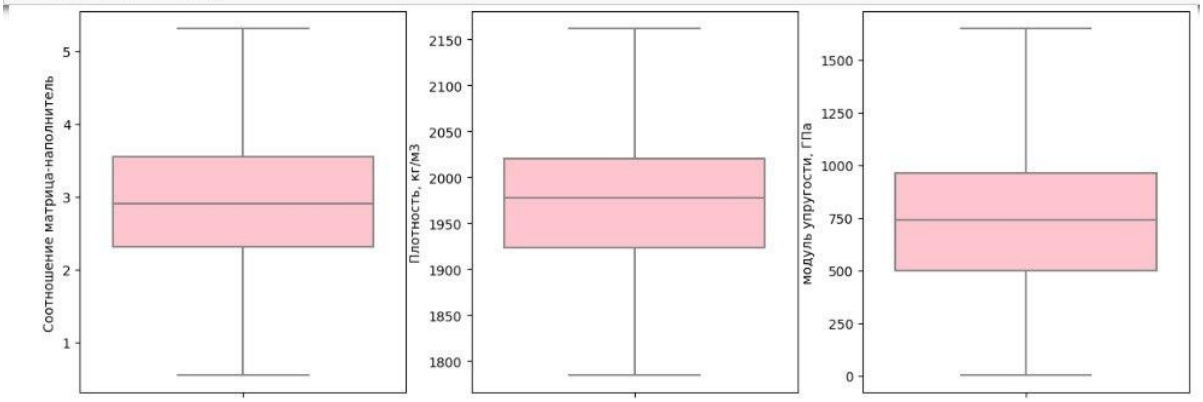

Рисунок 6 – Boxplot после очистки датасета

Данные объединенного датасета не имеют чётко выраженной зависимости, что подтверждает тепловая карта с матрицей корреляции.

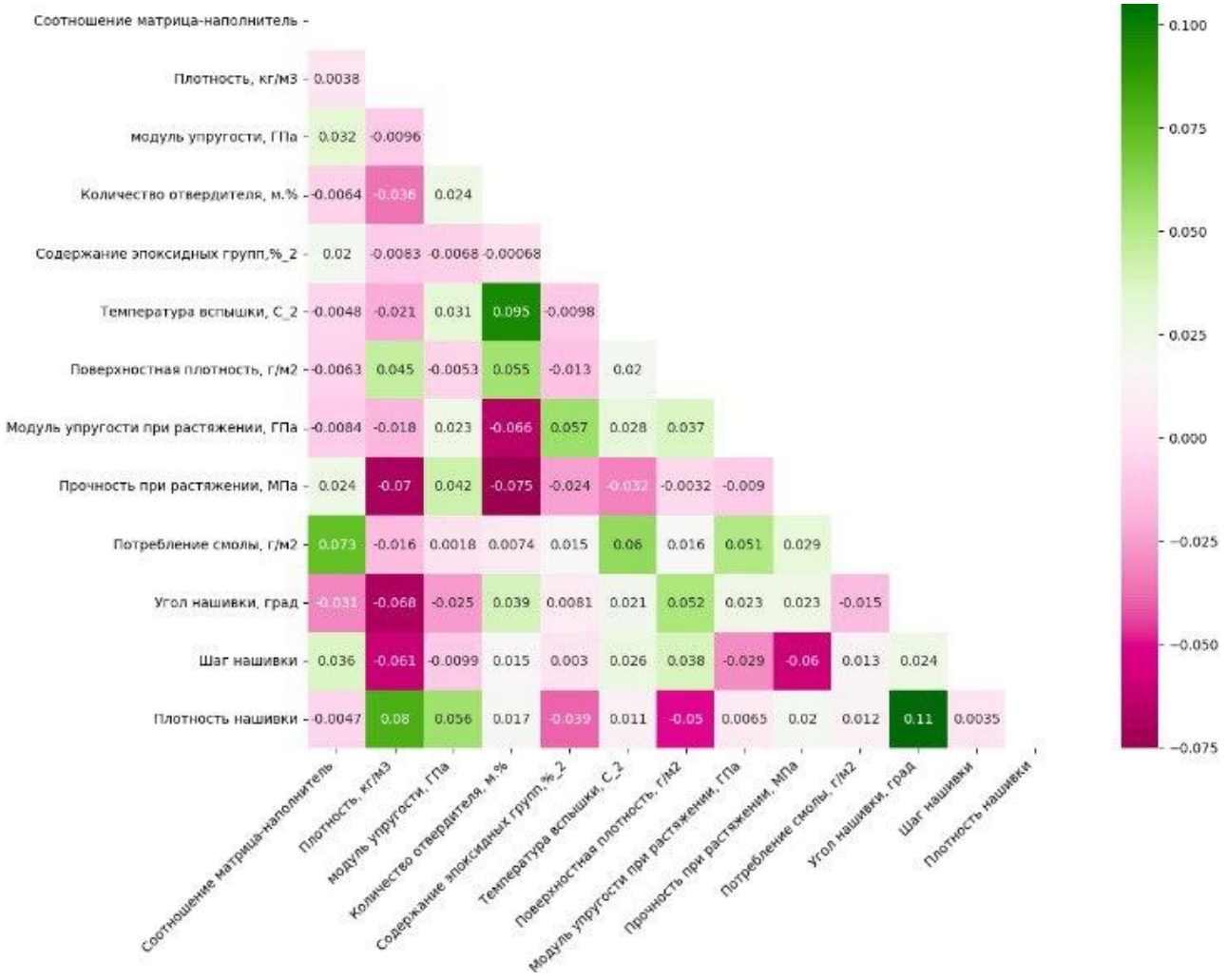

Рисунок 7 – Тепловая карта с корреляцией данных

Максимальная корреляция между плотностью нашивки и углом нашивки 0.11, значит зависимости между этими данными нет. Корреляция между всеми параметрами близка к 0, корреляционные связи между переменными не наблюдаются.

Гистограммы показывают нормальное распределение, за исключением признака Угол нашивки, который имеет всего два значения 0 и 90 градусов. Данный столбец мы преобразуем в числа 0 и 1 с помощью кодировщика LabelEncoder.

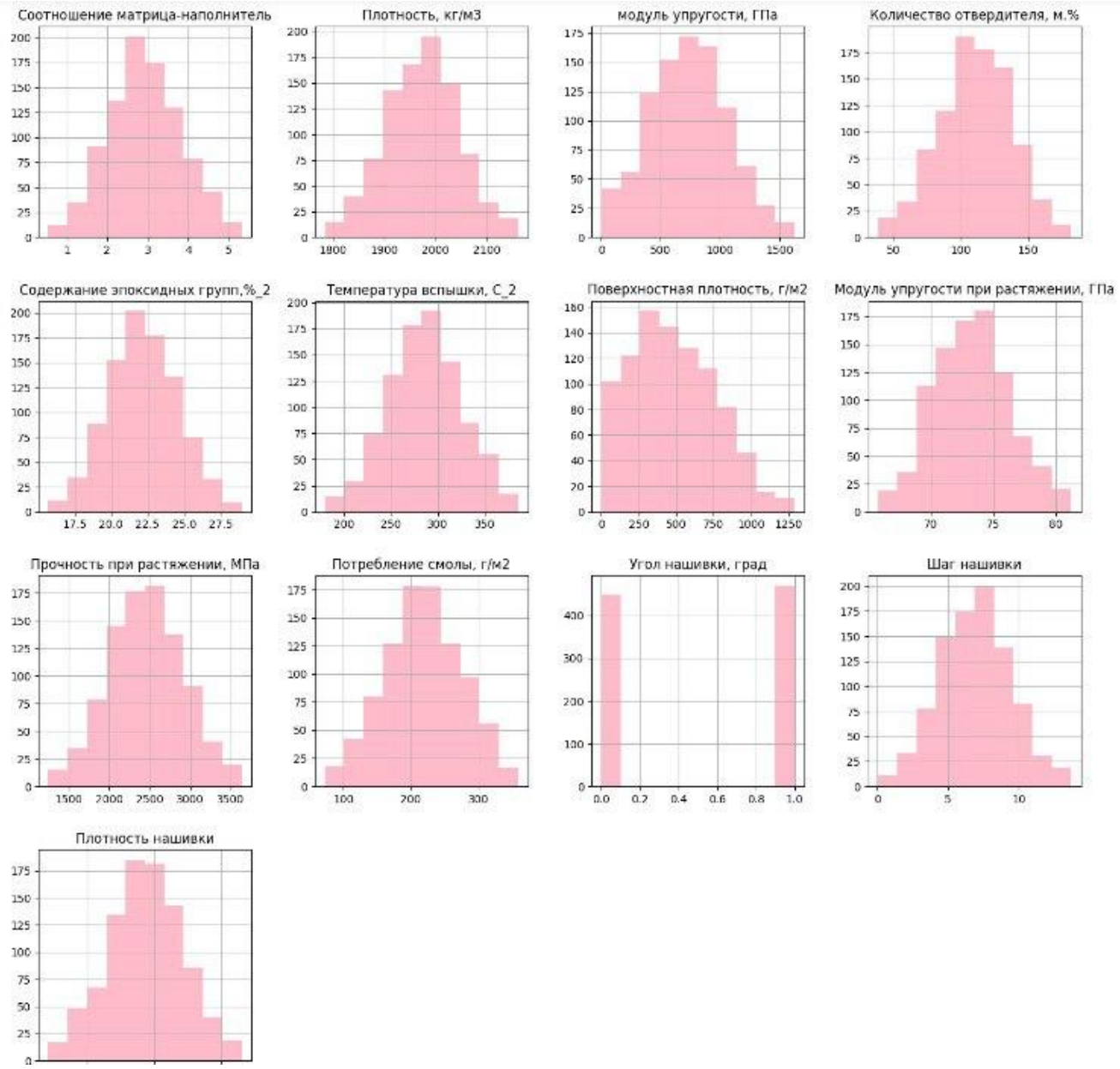

Рисунок 8 – Гистограммы распределения очищенного датасета

На попарных графиках рассеяния также не видно корреляции между признаками.

```
<seaborn.axisgrid.PairGrid at 0x7fbf90c0b3a0>
```

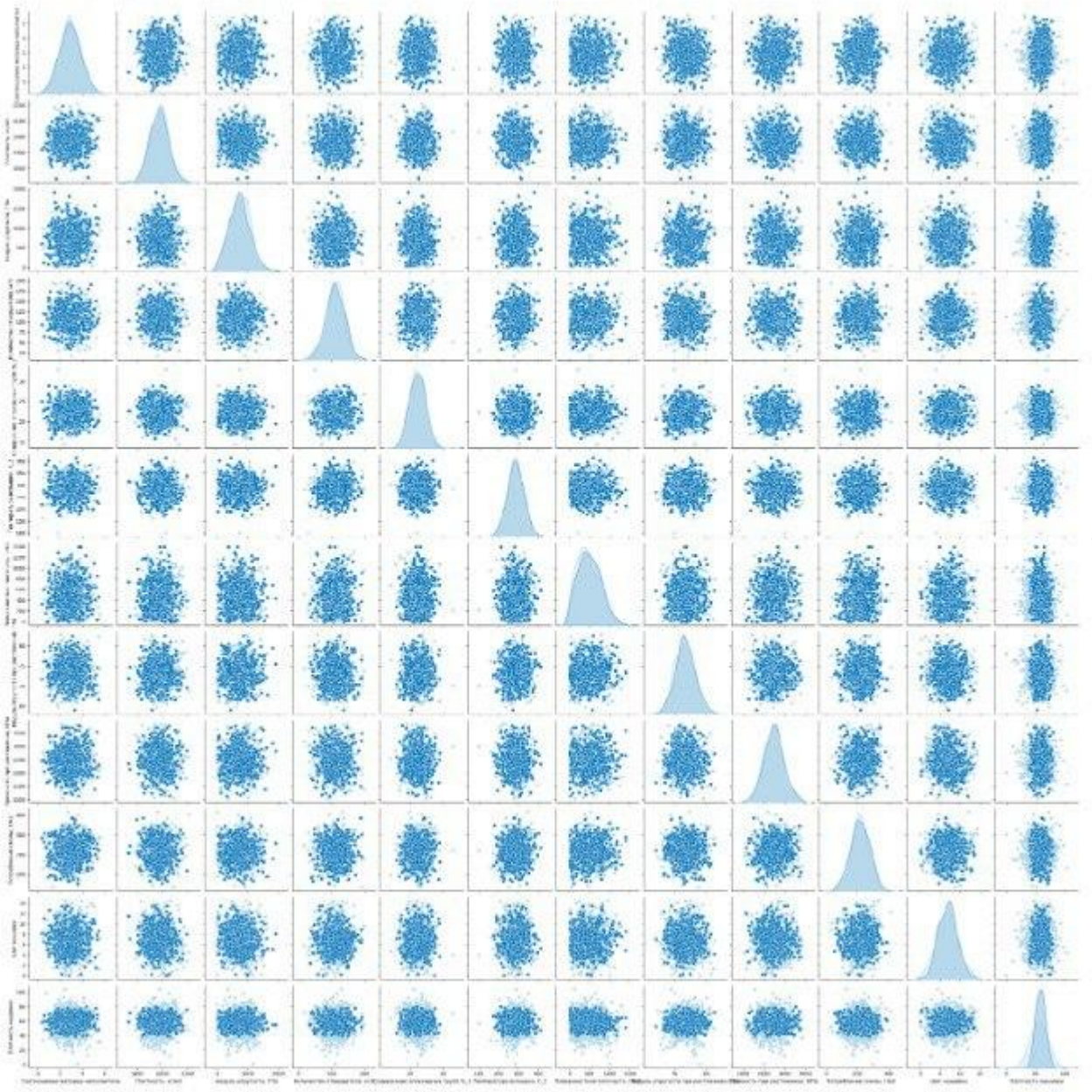

Рисунок 8 – Попарные графики рассеяния точек

2. Практическая часть

2.1. Предобработка данных

По условиям задания нормализуем значения. Для этого применим MinMaxScaler().

2.2. Разработка и обучение моделей

Разработка и обучение моделей машинного обучения осуществлялась для двух выходных параметров: «Прочность при растяжении» и «Модуль упругости при растяжении» отдельно. Для решения были использованы некоторые из методов, описанные выше.

```
knr = KNeighborsRegressor()
knr_params = {'n_neighbors': range(1, 200),
              'weights': ['uniform', 'distance'],
              'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute']}
GSCV_knr_e = GridSearchCV(knr, knr_params, n_jobs=-1, cv=10)
GSCV_knr_e.fit(X_train_e, y_train_e)
GSCV_knr_e.best_params_
print(f'Лучшая модель на кросс-валидации с параметрами {GSCV_knr_e.best_params_} и результатом {GSCV_knr_e.best_score_:.4f}')
```

```
Лучшая модель на кросс-валидации с параметрами {'algorithm': 'auto', 'n_neighbors': 137, 'weights': 'uniform'} и результатом -
0.0232
```

```
GSCV_knr_s = GridSearchCV(knr, knr_params, n_jobs=-1, cv=10)
GSCV_knr_s.fit(X_train_s, y_train_s)
GSCV_knr_s.best_params_
print(f'Лучшая модель на кросс-валидации с параметрами {GSCV_knr_s.best_params_} и результатом {GSCV_knr_s.best_score_:.4f}')
```

```
Лучшая модель на кросс-валидации с параметрами {'algorithm': 'auto', 'n_neighbors': 93, 'weights': 'uniform'} и результатом -
0.0102
```

Рисунок 9 – Поиск гиперпараметров по сетке

Порядок разработки модели для каждого параметра и для каждого выбранного метода можно разделить на следующие этапы: разделение нормализованных данных на обучающую и тестовую выборки (в соотношении 70 на 30); обучение моделей; сравнение моделей по метрике MAE и MSE; поиск гиперпараметров, по которым будет происходить оптимизация модели, с помощью выбора по сетке и перекрёстной проверки. Оценка полученных результатов работы моделей. В качестве параметра оценки выбран также коэффициент детерминации (R-квадрат, R2).

```
rfr_params = {'n_estimators': [150, 200, 250, 300],
              'max_depth': [5, 20],
              'criterion': ['squared_error', 'absolute_error', 'friedman_mse', 'poisson'] }
GSCV_rfr_e = GridSearchCV(rfr, rfr_params, n_jobs=-1, cv=10)
GSCV_rfr_e.fit(X_train_e, y_train_e)
GSCV_rfr_e.best_params_
print(f'Лучшая модель на кросс-валидации с параметрами {GSCV_rfr_e.best_params_} и результатом {GSCV_rfr_e.best_score_:.4f}')
```

```
Лучшая модель на кросс-валидации с параметрами {'criterion': 'absolute_error', 'max_depth': 5, 'n_estimators': 200} и результа
том -0.0623
```

```
Результат: Лучшая модель на кросс-валидации с параметрами {'criterion': 'absolute_error', 'max_depth': 5, 'n_estimators': 200}
и результатом -0.0623
```

```
GSCV_rfr_s = GridSearchCV(rfr, rfr_params, n_jobs=-1, cv=10)
GSCV_rfr_s.fit(X_train_s, y_train_s)
GSCV_rfr_s.best_params_
print(f'Лучшая модель на кросс-валидации с параметрами {GSCV_rfr_s.best_params_} и результатом {GSCV_rfr_s.best_score_:.4f}')
```

```
Лучшая модель на кросс-валидации с параметрами {'criterion': 'absolute_error', 'max_depth': 5, 'n_estimators': 150} и результа
том -0.0329
```

```
Результат: Лучшая модель на кросс-валидации с параметрами {'criterion': 'absolute_error', 'max_depth': 5, 'n_estimators': 150}
и результатом -0.0329
```

```
best_model_rfr_e = GSCV_rfr_e.best_estimator_
best_model_rfr_s = GSCV_rfr_s.best_estimator_
```

```
print(best_model_rfr_e)

print(f'R2-score RFR для Модуля упругости при растяжении, МПа: {best_model_rfr_e.score(X_test_e, y_test_e).round(3)}')
```

```
RandomForestRegressor(criterion='absolute_error', max_depth=5, n_estimators=200)
R2-score RFR для Модуля упругости при растяжении, МПа: -0.004
```

Рисунок 10 – Модель RandomForestRegressor

2.3. Тестирование модели

Модели после настройки гиперпараметров показали результат немного лучше. В результате все модели показали примерно одинаковый результат: ошибка MAE примерно равна стандартному отклонению, значения R2 находятся около нуля или отрицательны, то есть все модели предсказывают результат, сопоставимый со средним значением. Можно считать, что все примененные модели не справились с задачей, результат неудовлетворительный.

	MSE train	MSE test	R2 train	R2 test	MAE train	MAE test	MSE train	MSE test	R2 train	R2 test	MAE train	MAE test
RandomForestRegressor	6,9005	9,2088	0,2483	-0,0145	2,1325	2,4773	141389,7205	224732,5402	0,2756	0,0013	300,0945	384,4820
RandomForestRegressor_best	6,8874	9,1510	0,2498	-0,0082	2,0923	2,4501	150521,6349	226238,0916	0,2288	-0,0054	305,7443	384,2891
RandomForestRegressor_norm	0,0290	0,0384	0,2502	-0,0054	0,1357	0,1588	0,0061	0,0403	0,8226	-0,0193	0,0623	0,1625
Lasso	9,1087	9,0485	0,0078	0,0031	2,4409	2,4471	191764,4221	224528,2006	0,0175	0,0022	348,1372	384,1102
Lasso_best	9,1642	9,0960	0,0017	-0,0021	2,4453	2,4543	191887,4816	223679,5537	0,0169	0,0060	348,3619	383,0815
Lasso_norm	0,0387	0,0383	0,0000	-0,0009	0,1587	0,1591	0,0343	0,0395	0,0000	-26,9966	0,1469	0,1607
DecisionTreeRegressor	0,0000	18,7589	1,0000	-1,0667	0,0000	3,5058	0,0000	412531,9984	1,0000	-0,8332	0,0000	505,6368
DecisionTreeRegressor_best	8,6243	9,7378	0,0606	-0,0728	2,3382	2,5229	184036,1737	229794,8330	0,0571	-0,0212	332,8734	386,5286
DecisionTreeRegressor_norm	0,0363	0,0410	0,0606	-0,0728	0,1517	0,1637	0,0323	0,0403	0,0571	-0,0212	0,1395	0,1620
KNeighborsRegressor_best	9,0767	9,2401	0,0113	-0,0180	2,4322	2,4806	190223,9471	225364,1789	0,0254	-0,0015	346,2832	385,7366
KNeighborsRegressor_norm	0,0385	0,0382	0,0055	0,0010	0,1584	0,1591	0,0326	0,0406	0,0488	-0,0276	0,1435	0,1636
SGDRegressor_best	16,4760	15,3740	-0,7947	-0,6938	3,2687	3,2256	216378,0788	263885,3112	-0,1086	-0,1727	368,6177	417,0691
SGDRegressor_norm	0,0387	0,0383	-0,0002	-0,0019	0,1588	0,1591	0,0344	0,0396	-0,0031	-0,0013	0,1473	0,1611
Ridge_best	9,0739	9,0440	0,0116	0,0036	2,4419	2,4490	191839,1980	223790,0397	0,0172	0,0055	348,3967	383,4161
Ridge_norm	0,0382	0,0382	0,0116	0,0011	0,1582	0,1593	0,0338	0,0394	0,0136	0,0026	0,1460	0,1608

Рисунок 11 – Оценка результатов работы моделей

2.4. Нейронная сеть для рекомендации «Соотношение матрица-наполнитель»

Загружаем очищенный датасет, для X удаляем целевой столбец «Соотношение матрица-наполнитель» и сохраняем его в y. Далее разбиваем на обучающую и тестовую выборку. Затем делаем нормализацию входных данных и преобразуем их в np.array.

```
target_matrix = df_matrix['Соотношение матрица-наполнитель']
train_matrix = df_matrix.drop(['Соотношение матрица-наполнитель', 'Unnamed: 0'], axis=1)
```

```
target_matrix.head(2)
```

```
0    1.857143
1    1.857143
Name: Соотношение матрица-наполнитель, dtype: float64
```

```
train_matrix.head(2)
```

	Плотность, кг/м3	модуль упругости, ГПа	Количество отвердителя, м.%	Содержание эпоксидных групп,%_2	Температура вспышки, С_2	Поверхностная плотность, г/м2	Модуль упругости при растяжении, ГПа	Прочность при растяжении, МПа	Потребление смолы, г/м2	Угол нашивки, град	Шаг нашивки	Плотно наши
0	2030.0	738.736842	50.0	23.75	284.615385	210.0	70.0	3000.0	220.0	0	4.0	
1	2030.0	738.736842	129.0	21.25	300.000000	210.0	70.0	3000.0	220.0	0	5.0	

```
x_train, x_test, y_train, y_test = train_test_split(train_matrix, target_matrix, test_size = 0.3, random_state = 17)
```

```
# нормализуем входные данные и преобразуем в np.array
x_train_n = tf.keras.layers.Normalization(axis =-1)
x_train_n.adapt(np.array(x_train))
```

Рисунок 12 – Разбиение и нормализация данных для нейронной сети

Создаем архитектуру нейронной сети и запускаем обучение. Оценивая результаты меняем параметры нейросети: количество нейронов, функции активации, количество слоев.

```
model = Sequential(x_train_n)

model.add(Dense(128))
model.add(BatchNormalization())
model.add(LeakyReLU())
model.add(Dense(64))
model.add(BatchNormalization())
model.add(LeakyReLU())
model.add(Dense(64))
model.add(BatchNormalization())
model.add(LeakyReLU())
model.add(Dense(32))
model.add(BatchNormalization())
model.add(LeakyReLU())
model.add(Dense(32))
model.add(BatchNormalization())
model.add(LeakyReLU())
model.add(Dense(1))
model.add(Activation(activation='elu'))
```

```
model.compile(optimizer=tf.optimizers.SGD(learning_rate=0.01,
                                          momentum=0.9,
                                          nesterov=False),
              loss='mean_absolute_error')
```

```
%%time
history = model.fit(x_train, y_train,
    batch_size = 64,
    epochs=40,
    verbose=1,
    validation_split = 0.2
    )
```

Рисунок 13 – Код нейросети

Все нейросети показали схожий результат с ошибкой MAE чуть меньшей, чем среднее отклонение.

```
model_loss_plot(history)
```

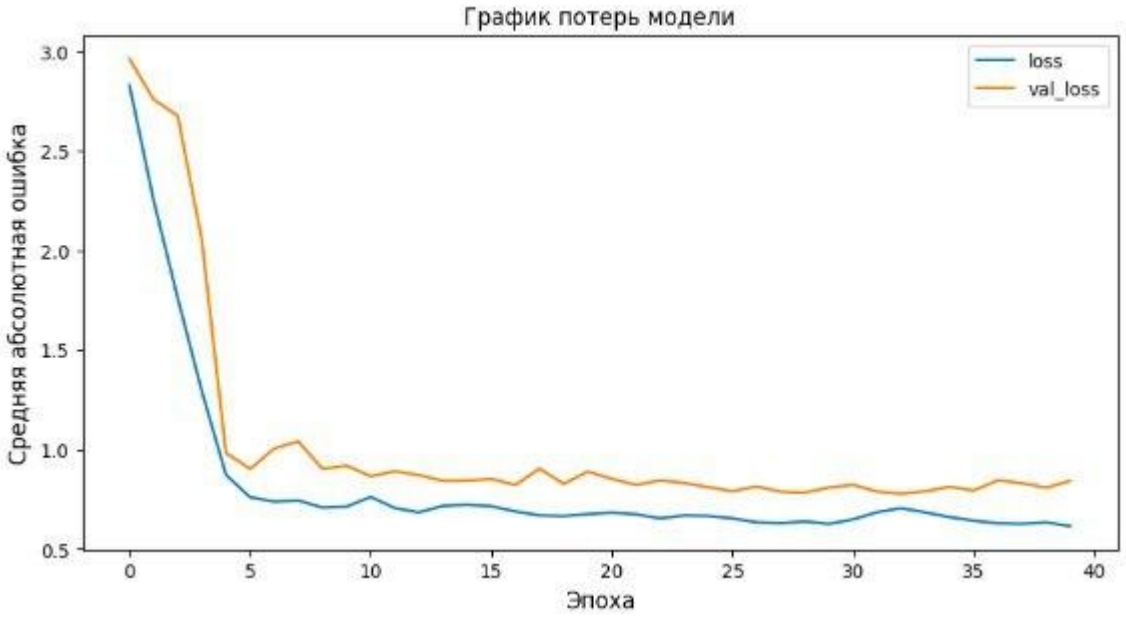

Рисунок 14 – График потерь моделей

2.5. Разработка приложения

Создание приложения для расчета параметра «Соотношение матрица-наполнитель». Данное приложение – это основной файл Flask, папка templates с шаблоном html-страницы, папка models с сохранёнными моделями model1 и model2.

31 lines (24 sloc) | 1021 Bytes Raw Blame

```python
import flask
from flask import render_template
import tensorflow as tf
from tensorflow import keras
import numpy as np

app = flask.Flask(__name__, template_folder = 'templates')

def prediction(param1, param2, param3, param4, param5, param6, param7, param8, param9, param10, param11, param12):
    model = tf.keras.models.load_model('../models/model2/')
    prediction = model.predict([param1, param2, param3, param4, param5, param6, param7, param8, param9, param10, param11, param12])
    return prediction[0][0]

@app.route('/', methods=['POST', 'GET'])

def main():
    params_list = []
    result = ''
    if flask.request.method == 'GET':
        return render_template('main.html')

    if flask.request.method == 'POST':
        for i in range(1,13,1):
            param = flask.request.form.get(f'param{i}')
            params_list.append(float(param))

        result = prediction(*params_list)
    return render_template('main.html', result=result)

if __name__ == '__main__':
    app.run()
```

Рисунок 15 - Код приложения

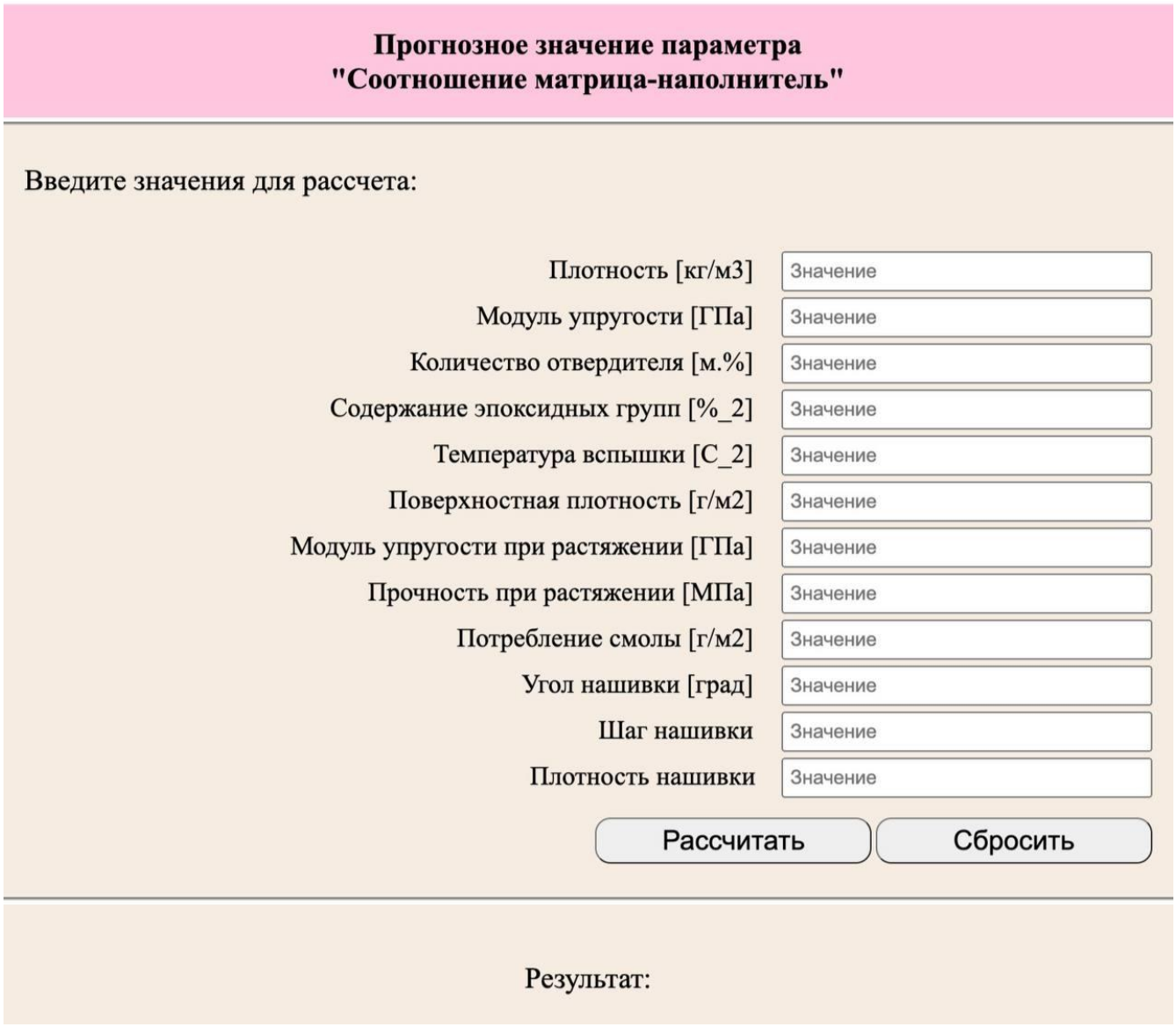

Рисунок 16 – Форма для ввода параметров

На выходе пользователь получает результат прогноза для значения параметра «Соотношение матрица – наполнитель».

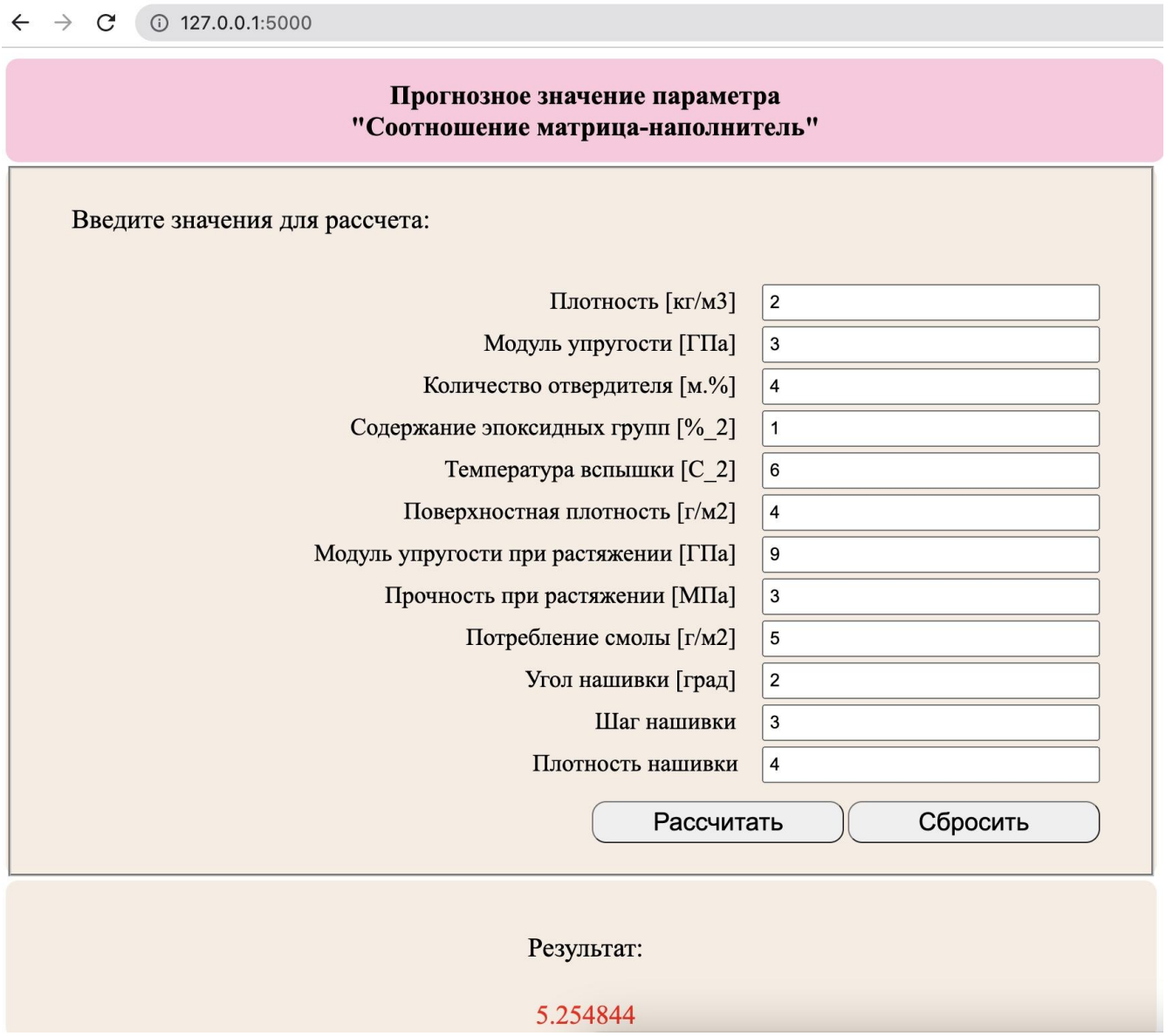

Рисунок 17 – Результат расчёта «Соотношение матрица – наполнитель»

2.6. Создание удаленного репозитория и загрузка

Репозиторий был создан на github.com по адресу: https://github.com/t-448/composite-vkr

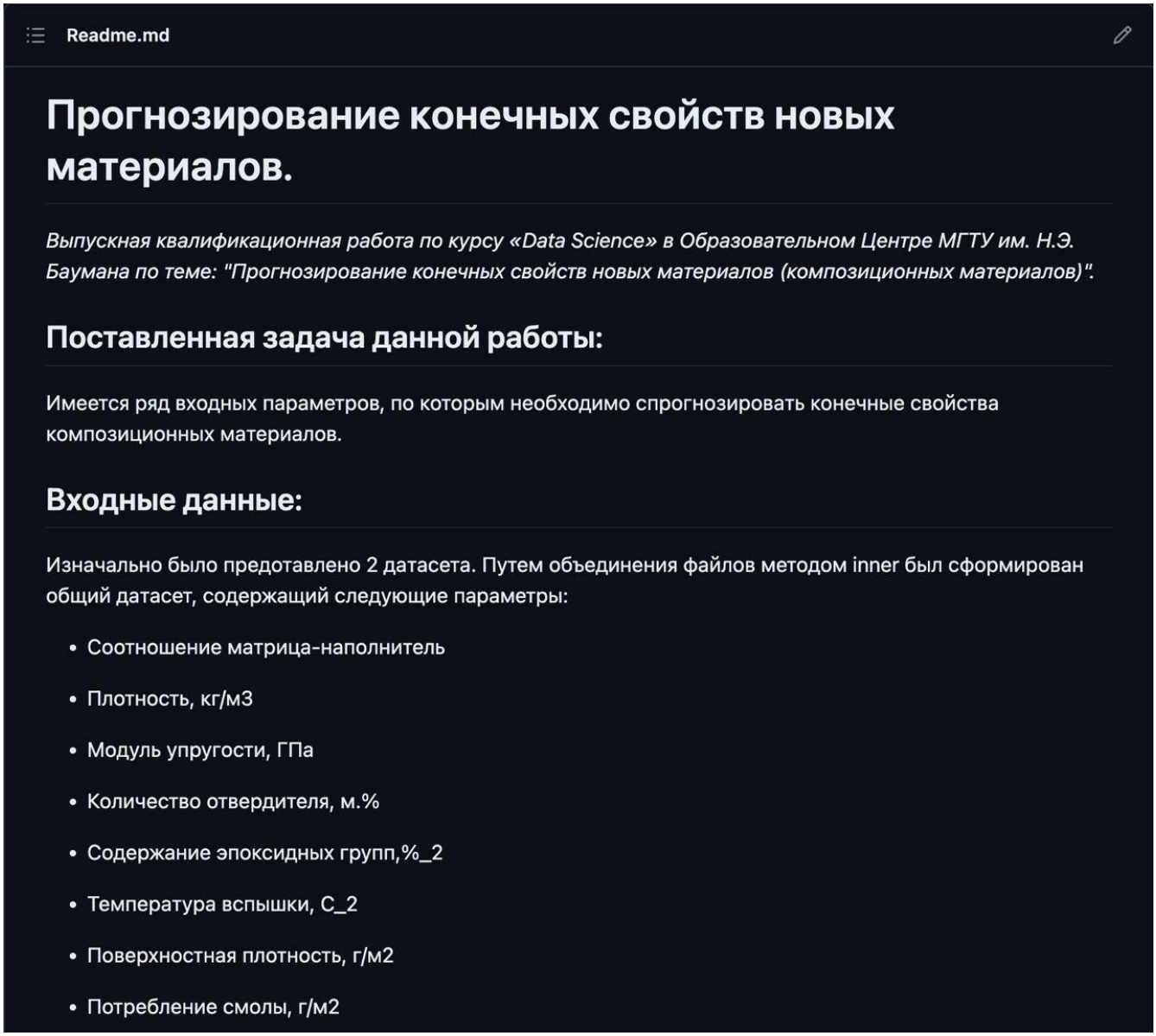

Рисунок 18 – Часть файла Readme.md

Заключение

Данное исследование демонстрирует значимые ограничения в предсказании свойств композитных материалов на основе имеющихся данных. Несмотря на близкое к нормальному распределение данных в объединенном датасете, наблюдается стремление коэффициентов корреляции между признаками к нулю, что не позволяет получить достоверные прогнозы. Модели регрессии, используемые в этом исследовании, также не проявили высокой эффективности в прогнозировании свойств композитов, наблюдается неудовлетворительный уровень результатов. Среди прочих выводов, следует отметить, что не удалось установить связь между свойствами материалов и соотношением матрицы и наполнителя. Такой вывод, однако, не говорит о том, что прогнозирование характеристик композитных материалов на основе данного набора данных является невозможным. Скорее всего, проблема связана с недостатком вводных данных, использованными подходами, инструментами и методами, а также необходимостью дополнительных исследований, включая консультации экспертов. Для улучшения результатов требуются дополнительные вводные данные, полученные в результате математических преобразований, а также активное сотрудничество междисциплинарной команды ученых. В целом, необходимо учитывать, что предсказание характеристик композиционных материалов без углубленного изучения материаловедения и проведения экспериментального анализа свойств этих материалов не является решением удовлетворительного уровня. Для того, чтобы улучшить модели и повысить точность прогнозирования, необходимо применять производные методы и включать в анализ более широкий набор показателей для выявления дополнительных взаимосвязей между признаками. С учетом отсутствия корреляции между признаками, можно сделать вывод о том, что задача прогнозирования характеристик композиционных материалов при использовании текущих алгоритмов является трудной и может

потребовать улучшенных подходов и сбора более полной и релевантной информации.

Список литературы

1 Агапова, Е. С. (2018). Применение методов машинного обучения для построения квантово-химических моделей. Известия высших учебных заведений. Химия и химическая технология, 61(5), 47-53.

2 Брыков, А. Н., & Логинов, А. С. (2019). Прогнозирование свойств новых материалов с использованием методов машинного обучения. Научно-технический вестник информационных технологий, механики и оптики, 19(2), 249-255.

3 Волков, А. Г., Горелова, М. Н., & Камкина, Н. В. (2021). Моделирование свойств новых материалов методами машинного обучения. Вестник Московского университета. Серия 2: Химия, 62(3), 190-195.

4 Губанова, Е. Р., & Давыдова, Ю. А. (2018). Применение методов машинного обучения для предсказания свойств материалов. Материалы конференции "Молодые ученые России: вектор науки", 1-5.

5 Дроздова, Е. А. (2019). Моделирование свойств материалов методами машинного обучения. Известия Волгоградского государственного технического университета, 23(2), 82-88.

6 Косых, М. С., & Козлов, А. А. (2021). Применение машинного обучения для прогнозирования свойств материалов. Экологическая безопасность и устойчивое развитие, 5(1), 63-73.

7 Максимов, Ю. М., & Жуков, С. Ю. (2019). Машинное обучение для расчета термодинамических свойств веществ. Журнал физической химии, 93(4), 517-527.

8 Плетнева, М. В., Пащенко, А. С., & Жуков, С. Ю. (2018). Использование методов машинного обучения для предсказания термодинамических свойств неорганических соединений. Известия высших учебных заведений. Химия и химическая технология, 61(1), 71-80.

9 Раймонд, Ж., Гаренц, А., & Хуперт, К. (2019). Применение методов машинного обучения для исследования свойств материалов на основе металлических кластеров. Журнал физической химии, 93(12), 2049-2057.

10 Саввина, Е. А. (2021). Прогнозирование свойств новых материалов с помощью методов машинного обучения. Научный журнал КубГАУ, 2(163), 1-8.

11 Санников, И. А., & Дороганов, В. В. (2019). Прогнозирование свойств материалов методами машинного обучения. Журнал технической физики, 89(5), 12-20.

12 Сметанин, А. Н. (2020). Моделирование свойств материалов с помощью методов машинного обучения. Труды Московского физико-технического института, 12(1), 37-42.

13 Токмакова, М. Ю., & Гришкин, С. И. (2019). Прогнозирование свойств новых материалов с использованием методов машинного обучения. Журнал прикладной химии, 92(8), 1183-1189.

14 Хлыстова, Л. А. (2021). Применение методов машинного обучения для прогнозирования свойств материалов. Журнал высшей школы и прикладной химии, 94(5), 413-419.

15 Хохлова, М. А., & Александрова, А. С. (2020). Моделирование свойств материалов с помощью машинного обучения. Вестник Российского университета дружбы народов. Серия: Математика, информатика, физика, 28(2), 185-194.

16 Amsler, C., Frenkel, D., & Zhu, Q. (2020). Machine learning in materials science. Annual Review of Materials Research, 50, 1-

17 Chen, C., & Wong, D. S. (2015). Machine learning applied to the prediction of physical properties of molecular crystals. Journal of Chemical Information and Modeling, 55(5), 1021-1032.

18 Choudhary, K., Kalish, I., & Singh, A. (2018). Machine learning-assisted materials discovery: an overview. npj Computational Materials, 4(1), 1-15.

19 Faber, F. A., Lindmaa, A., von Lilienfeld, O. A., & Armiento, R. (2016). Crystal structure representations for machine learning models of formation energies. International Journal of Quantum Chemistry, 116(10), 742-749.

20 Gómez-Bombarelli, R., Aguilera-Iparraguirre, J., Hirzel, T. D., Duvenaud, D., Maclaurin, D., Blood-Forsythe, M. A., ... & Aspuru-Guzik, A. (2018). Design of efficient molecular organic light-emitting diodes by a high-throughput virtual screening and experimental approach. Nature Materials, 17(3), 216-223.

21 Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press.

22 Hautier, G., Fischer, C. C., Jain, A., Mueller, T., & Ceder, G. (2010). Finding nature's missing ternary oxide compounds using machine learning and density functional theory. Chemistry of Materials, 22(12), 3762-3767.

23 Kalinin, S. V., Sumpter, B. G., & Archibald, R. K. (2015). Big-deep-smart data in imaging for guiding materials design. Nature Materials, 14(10), 973-980.

24 Löfberg, A., & Eriksson, L. (2015). Quantitative structure-property relationships of corrosion inhibitors using random forests. Journal of Chemical Information and Modeling, 55(9), 1882-1892.

25 Mannodi-Kanakkithodi, A., Pilania, G., Huan, T. D., Lookman, T., & Ramprasad, R. (2016). Machine learning strategy for accelerated design of polymer dielectrics. Scientific Reports, 6(1), 1-11.

26 Meredig, B., Agrawal, A., Kirklin, S., Saal, J. E., Doak, J. W., Thompson, A., & Wolverton, C. (2014). Combinatorial screening for new materials in unconstrained composition space with machine learning. Physical Review B, 89(9), 094104.

27 Pilania, G., Wang, C., Jiang, X., Rajasekaran, S., & Ramprasad, R. (2017). Accelerating materials property predictions using machine learning. Scientific Reports, 7(1), 1-8.

28 Pyzer-Knapp, E. O., Suh, C., & Gómez-Bombarelli, R. (2015). Towards fast computational screening in soft matter: a heuristic approach for the discovery of molecular organic light-emitting diodes. Journal of Materials Chemistry C, 3(39), 10182-10190.

29 Raccuglia, P., Elbert, K. C., Adler, P. D., Falk, C., Wenny, M. B., Mollo, A., ... & Green, W. H. (2016). Machine-learning-assisted materials discovery using failed experiments. Nature, 533(7601), 73-76.

30 Ramakrishnan, R., Hartmann, M., Tapavicza, E., & von Lilienfeld, O. A. (2015). Electronic spectra from TDDFT and machine learning in chemical space. Journal of Chemical Physics, 143(8), 084111.

31 Rasmussen, C. E., & Williams, C. K. (2006). Gaussian processes for machine learning. MIT Press.

32 Sorkun, H. B., & Avci, D. (2020). Prediction of sound absorption coefficient of polyurethane foam-based composites using machine learning algorithms. Journal of Sandwich Structures & Materials, 22(1), 214-237.

33 Sudre, C. H., Murray, B., Varsavsky, T., Graham, M. S., Penfold, R. S., Bowyer, R. C., Wolf, J. (2020). Attributes and predictors of long COVID. Nature Medicine, 27(4), 626-631.

34 Ward, L., Liu, R., Krishna, A., Hegde, V. I., Agrawal, A., Choudhary, A., Wolverton, C. (2016). Including crystal structure attributes in machine learning models of formation energies via Voronoi tessellations. Physical Review B, 94(9), 094110.

35 Kaggle. https://www.kaggle.com/

36 TensorFlow. https://www.tensorflow.org/

37 Keras. https://keras.io/

38 PyTorch. https://pytorch.org/

39 Scikit-learn. https://scikit-learn.org/

40 SciPy. https://www.scipy.org/

41 GitHub. https://github.com/

...