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 Леонтьева Ксения Александровна

Москва, 2022

Содержание

Введение

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

1. Аналитическая часть

1.1. Постановка задачи

Цель: спрогнозировать ряд конечных свойств получаемых композиционных материалов: модуль упругости при растяжении, прочночть при растяжении, соотношение матрица-наполнитель.

Исходные данные для исследования: данные о начальных свойствах компонентов композиционных материалов в формате таблицы Excel. Данные содержаться в двух файлах «X_bp.xlsx» и « X_nup.xlsx».

Таблица «X_bp.xlsx» состоит из 1023 строки и 11 столбцов.

Таблица «X_nup.xlsx» состоит из 1040 строк и 3 столбцов. Названия столбцов приведены на рисунке 1.

```
X_bp

<class 'pandas.core.frame.DataFrame'>
Float64Index: 1023 entries, 0.0 to 1022.0
Data columns (total 10 columns):
 #   Column                                Non-Null Count  Dtype
---  ------                                --------------  -----
 0   Соотношение матрица-наполнитель       1023 non-null   float64
 1   Плотность, кг/м3                      1023 non-null   float64
 2   модуль упругости, ГПа                 1023 non-null   float64
 3   Количество отвердителя, м.%           1023 non-null   float64
 4   Содержание эпоксидных групп,%_2       1023 non-null   float64
 5   Температура вспышки, С_2              1023 non-null   float64
 6   Поверхностная плотность, г/м2         1023 non-null   float64
 7   Модуль упругости при растяжении, ГПа  1023 non-null   float64
 8   Прочность при растяжении, МПа         1023 non-null   float64
 9   Потребление смолы, г/м2               1023 non-null   float64
dtypes: float64(10)
memory usage: 87.9 KB

-----------------------------------------------------------------------
X_nup

<class 'pandas.core.frame.DataFrame'>
Float64Index: 1040 entries, 0.0 to 1039.0
Data columns (total 3 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Угол нашивки, град  1040 non-null  float64
 1   Шаг нашивки         1040 non-null  float64
 2   Плотность нашивки   1040 non-null  float64
dtypes: float64(3)
memory usage: 32.5 KB

-----------------------------------------------------------------------
```

Рисунок 1 – общей информации о данных

Для дальнейшей работы таблицы были объединены по индексу тип объединения INNER.

1.2. Описание используемых методов

В данной задаче необходимо предсказать значений вещественной, непрерывной переменной — это задача регрессии. Для прогнозирования зависимая переменная должна иметь связь с одной или несколькими независимыми переменными.

Данная задача относится к категории машинного обучения с учителем. Обучающий набор представлен признаками вместе с целевым признаком, который необходимо предсказать.

Для предсказания непрерывной величины используются регрессионные модели. При выборе модели обычно рекомендуется начинать с простых, интерпретируемых моделей, таких как линейная регрессия, и если результаты будут неудовлетворительными, то переходить к более сложным, но обычно более точным методам. Мы будем оценивать пять моделей разной степени сложности:

- линейные модели;
- деревья принятия решений;
- алгоритмы бустинга;
- метод опорных векторов;
- нейронная сеть.

Так как разработано много моделей регрессионного анализа, выберем несколько из всего многообразия.

За базовую модель для возьмем модель предсказывающую среднее DummyRegressor. По ней будем проверять модели на адекватность. Модели должны давать метрику лучше, чем модель предсказывающая постоянно среднее значение целевых значений по умолчанию.

DummyRegressor — это регрессор, который дает прогноз на основе простых стратегий, не обращая никакого внимания на входные данные. Данный регрессор имеет несколько типов стратегий: «среднее значение», «медиана», «квантиль» или «константа». Данная модель используется для проверки того, насколько хорошо регулярная регрессионная модель установлена на

определенный набор данных, но никогда не может быть использована в какой-либо реальной задаче.

1.2.1 Линейные модели

Из линейных моделей будем рассматривать:

- линейную регрессию;
- Ridge;
- Lasso.

Модель линейной регрессии (LinearRegression) - является одним из самых простых обучающихся алгоритмов. Данная модель хорошо работает, когда есть линейная связь между признаками и целевым признаком. Линейная регрессия исходит из того, что связь между признаками и вектором целей является приблизительно линейной, т.е. эффект (также называемый коэффициентом, весом или параметром) признаков на вектор целей является постоянным. Простая линейная регрессия имеет место, если рассматривается зависимость между одной входной и одной выходной переменными. Для этого строится прямая (линия регрессии): y= ax+b. Коэффициенты a и b — параметры модели, определяются, чтобы сумма квадратов отклонений точек, соответствующих реальным наблюдениям данных, от линии регрессии была бы минимальной. Коэффициенты обычно оцениваются методом наименьших квадратов. Для поиска зависимости между несколькими входными и одной выходной переменными применяют множественную линейную регрессию. Уравнение множественной линейной регрессии имеет вид: Y=b0+b1*x1+b2*x2+⋯+bn*xn (где n - число входных переменных).

Достоинства модели линейной регрессии:

- Простота;
- Скорость.

Недостатки модели линейной регрессии:

- В случае нелинейных данных полиномиальную регрессию трудно спроектировать. Необходимо иметь информацию о структуре данных и взаимосвязи между переменными.
- Линейная регрессия неэффективна, когда речь идёт об очень сложных данных и больших объёмах.

Модели Ridge и Lasso – это регуляризованные линейные модели, модели линейной регрессии, но с поправочным (штрафным) коэффициентом. Ridge и Lasso регрессии заставляют алгоритм обучения не только соответствовать данным, но и сохранять веса модели как можно меньшими, внося поправки в размерность бета-вектора разными способами. Lasso старается достигнуть наилучшей производительности, обнаружив бесполезность некоторых коэффициентов отбрасывает их. Ridge старается достигнуть наилучшей производительности, не позволяя ни одному из коэффициентов достигать экстремального значения, путем добавления штрафа, эквивалентного квадрату величины коэффициентов. Оба эти метода имеют коэффициент регуляризации, который контролирует величину штрафа. Для Ridge - это L1, для Lasso - L2 . При $\lambda=0$ как Ridge или Lasso регрессии становятся моделями линейной регрессии.
Достоинства моделей Ridge и Lasso:
- Простые методы, позволяющие уменьшить сложность модели и убрать мультиколлениарность.

1.2.2 Модели деревья принятия решений

Из моделей деревьев решений будем рассматривать:
- Решающее дерево;
- Случайный лес.

Решающее дерево (DecisionTreeRegressor) предсказывает значение целевой переменной с помощью применения последовательности простых решающих правил (которые называются предикатами). Структура дерева представляет собой «листья» и «ветки». На рёбрах («ветках») дерева решения записаны признаки, от которых зависит целевая функция, в «листьях» записаны значения

целевой функции, а в остальных узлах — признаки, по которым различаются случаи. Чтобы классифицировать новый случай, надо спуститься по дереву до листа и выдать соответствующее значение. Каждый лист представляет собой значение целевой переменной, изменённой в ходе движения от корня по рёбрам дерева до листа. Каждый внутренний узел сопоставляется с одной из входных переменных.

Достоинства модели DecisionTreeRegressor:

- Простота в понимании и интерпретации;
- Не требует специальной подготовки данных;
- Способен работать как с категориальными, так и с интервальными переменными.

Недостатки модели DecisionTreeRegressor:

- При построении дерева решений могут создаваться слишком сложные конструкции, которые недостаточно полно представляют данные (переобучением).

Случайный лес (RandomForestRegressor) - алгоритм машинного обучения, заключающийся в использовании ансамбля решающих деревьев. Основная идея заключается в использовании большого ансамбля решающих деревьев, каждое из которых само по себе даёт очень невысокое качество, но за счёт их большого количества результат получается хорошим (усреднение предсказаний отдельных деревьев).

Преимущества RandomForestRegressor:

- Способность эффективно обрабатывать данные с большим числом признаков и классов;
- Высокая точность предсказания;
- Нечувствительность к масштабированию значений признаков. Практически не чувствителен к выбросам в данных;
- Одинаково хорошо обрабатывает как непрерывные, так и дискретные признаки.

Недостатки RandomForestRegressor:

- Низкая скорость построения;
- Большой размер моделей.

1.2.3 Метод опорных векторов

Метод опорных векторов (Support Vector Regression) – это алгоритм контролируемого обучение моделей с использование схожих алгоритмов для анализа данных и распознавания шаблонов. Метод хорошо работает с небольшими датасетами. Данный алгоритм Учитывая обучающую выборку, где алгоритм помечает каждый объект, как принадлежащий к одной из двух категорий, строит модель, которая определяет новые наблюдения в одну из категорий. Каждый объект данных представляется как вектор (точка) в p-мерном пространстве. Он создаёт линию или гиперплоскость, которая разделяет данные на классы.

Преимущества Support Vector Regression:

- Достаточно небольшого набора данных;
- Способен обрабатывать случаи, когда гиперпараметров больше, чем количество наблюдений. Существует возможность гибко настраивать разделяющую функцию.

Недостатки Support Vector Regression:

- Неустойчивость к шуму;
- Для больших наборов данных требуется долгое время обучения.

1.2.4 Алгоритмы бустинга

Из алгоритмов бустинга будем рассматривать:

- XGBoost;
- CatBoostRegressor.

Эти алгоритм машинного обучения, основанный на дереве поиска решений и используют градиентный бустинг. Бустинг — это техника построения ансамблей, в которой предсказатели построены не независимо, а последовательно. Это техника использует идею о том, что следующая модель будет учится на ошибках предыдущей. Градиентный бустинг — это техника машинного обучения для задач классификации и регрессии, которая строит

модель предсказания в форме ансамбля слабых предсказывающих моделей, обычно деревьев решений.

1.2.5 Нейронная сеть

Нейронная сеть — модель, построенная по принципу организации и функционирования биологических нейронных сетей — сетей нервных клеток живого организма. Последовательность нейронов, соединенных между собой связями. У каждого нейрона есть определённое количество входов, принимающих сигналы, которые суммируются с учётом веса каждого входа. У нейрона есть функция активации, которая определяет выходное значение нейрона. Схема работы нейронных сетей включает в себя два основных этапа: прямое распространение ошибки, на котором функцией активации рассчитывается выходные значения, и обратное, на котором с помощью функции потерь минимизируется рассчитанная разница между предсказанным и реальным значением. В данной работе была разработана простая полносвязная нейронная сеть.

1.3. Разведочный анализ данных

Разведочный анализ данных (Exploratory Data Analysis) – предварительное исследование датасета с целью определения его основных характеристик, взаимосвязей между признаками, а также сужения набора методов, используемых для создания модели машинного обучения.

В данной работе будем использовать методы:

- Поиск дубликатов. Дублирующие записи не только искажают статистические показатели датасета, но и снижают качество обучения модели. В данном датасете дубликаты отсутствовали;
- Изучение данных на наличие пропусков в столбцах. В данном датасете дубликаты отсутствовали;
- Изучение типов данных. Все данные в нашем датасети относятся к типу данных float64.

- Обнаружение аномалий. Мы воспользовались визуальным способом для обнаружения аномалий. Для этого построили разновидность графика "ящик с усами" для всех переменных. При построении гистограмм и диаграмм ящика с усами были обнаружены выбросы. Всего в данных было 93 строки с выбросами (9,09% от всего датафрейма). При удалении выбросов и последующем обучении моделей на выборке без выбросов метрики моделей оказывались хуже, чем при обучении на выборках с выбросами. Скорее всего это связано с уменьшением наблюдений в выборке, что влияет на метрику модели в худшую сторону. Далее расчёты будем вести для выборок без удаления выбросов.
- Выявление зависимостей в данных. Для выявления зависимости были построены попарные графики рассеяния точек и матрица корреляции. Закономерностей выявлено не было, корреляция очень низкая, линейная зависимость отсутствует. Было сделано предположение о возможной неспособности линейных моделей предсказать хорошо целевые признаки.
- Изучение распределения переменных и и ключевые параметры распределения данных (среднее значение, медиана и стандартное отклонение). Для этого были построены Гистограммы распределения каждой из переменной. У всех переменных распределение близко к нормальному распределению.

2. Практическая часть

2.1. Предобработка данных

При оценки плотности ядра можно наблюдать, что данные необходимо нормализовать. Нормализация — это приведение различных данных в самых разных единицах измерения и диапазонах значений к единому виду. Диапазон значений в зависимости от примененного метода нормализации. Данные прошли нормализацию методом MinMaxScaler().

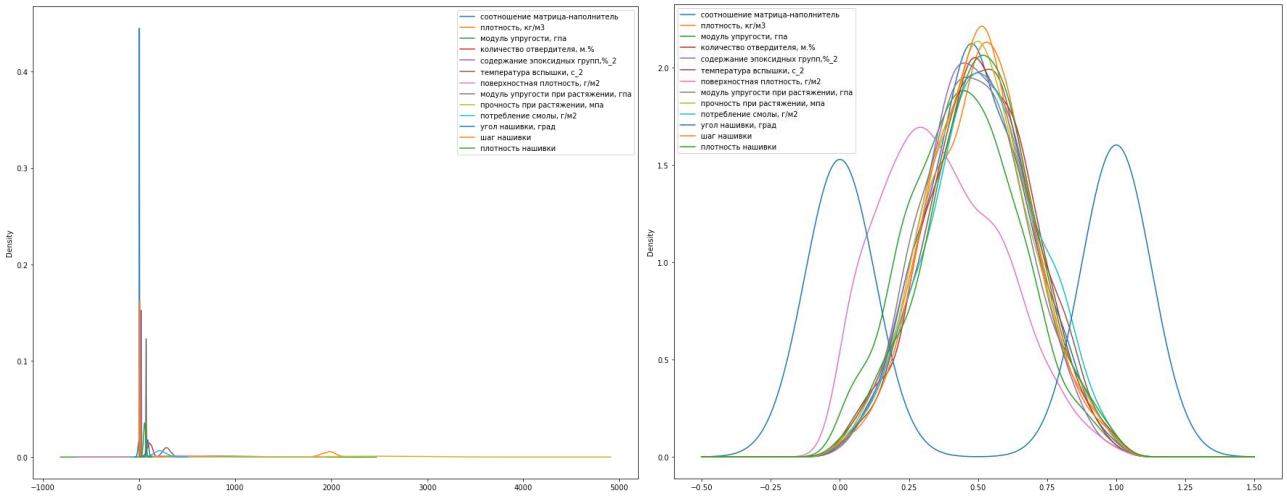

Рисунок 2 - визуализированные данные до и после нормализации

```
df_s.max().T
```

```
соотношение матрица-наполнитель          1.0
плотность, кг/м3                         1.0
модуль упругости, гпа                    1.0
количество отвердителя, м.%              1.0
содержание эпоксидных групп,%_2          1.0
температура вспышки, с_2                 1.0
поверхностная плотность, г/м2            1.0
модуль упругости при растяжении, гпа     1.0
прочность при растяжении, мпа            1.0
потребление смолы, г/м2                  1.0
угол нашивки, град                       1.0
шаг нашивки                              1.0
плотность нашивки                        1.0
dtype: float64
```

```
df_s.min().T
```

```
соотношение матрица-наполнитель          0.0
плотность, кг/м3                         0.0
модуль упругости, гпа                    0.0
количество отвердителя, м.%              0.0
содержание эпоксидных групп,%_2          0.0
температура вспышки, с_2                 0.0
поверхностная плотность, г/м2            0.0
модуль упругости при растяжении, гпа     0.0
прочность при растяжении, мпа            0.0
потребление смолы, г/м2                  0.0
угол нашивки, град                       0.0
шаг нашивки                              0.0
плотность нашивки                        0.0
dtype: float64
```

Рисунок 3 - максимальные и минимальные значения столбцов до и после нормализации

2.2. Разработка и обучение моделей

Для прогнозирования модуля упругости при растяжении и прочности при растяжении были использованы следующие модели:

1. LinearRegression;
2. Ridge;
3. Lasso;
4. DecisionTreeRegressor;
5. RandomForestRegressor;
6. Support Vector Regression;
7. XGBoost;
8. CatBoostRegressor;
9. Neural network;
10. DummyRegressor.

Модели будут обучаться как с подбором гиперпараметров так и без подбора с параметрами по умолчанию.

Работу моделей оценивать и сравнивать будем по метрики MSE. За базоую модель будем брать модель DummyRegressor.

Средняя квадратическая ошибка (MSE) - самый простой и распространенный показатель для оценки регрессии. Для каждой точки вычисляется квадратная разница между прогнозами и целью, а затем усредняются эти значения. Чем выше это значение, тем хуже модель. Он никогда не бывает отрицательным, поскольку мы возводим в квадрат отдельные ошибки прогнозирования, прежде чем их суммировать, но для идеальной модели это будет ноль.

Также для моделей рассчитаем метрику коэффициент детерминации (R^2). Значение R^2 всегда будет между $-\infty$ и 1. Когда R^2 отрицательно, это означает, что модель хуже, чем предсказание среднего значения.

Гиперпараметры будем подбирать путем поиска гиперпараметров с помощью поиска по сетке с перекрестной проверкой (количество блоков равно 10). Для этого будем применять GridSearchCV.

2.2.1 Обучение линейных моделей

Были обучены 4 линейных модели со стандартными параметрами: LinearRegression, Ridge, Lasso, SGDRegressor. Метрики всех моделей получились хуже или такая же как метрика модели предсказывающей среднее.

	Модель	MSE модели	r2 модели
0	DummyRegressor	0.02912	-0.03235
3	Lasso	0.02912	-0.03235
2	Ridge	0.02954	-0.04816
1	lr	0.02960	-0.05049
4	SGDRegressor	0.03052	-0.08749

Рисунок 4 - Результат работы линейных моделей со стандартными параметрами для прогноза модуля упругости при растяжении

	Модель	MSE модели	r2 модели
2	Ridge_1	0.02969	-0.02293
1	lr_1	0.02973	-0.02471
0	DummyRegressor	0.03010	-0.03230
3	Lasso_1	0.03010	-0.03230
4	SGDRegressor_1	0.03193	-0.09398

Рисунок 5 - Результат работы линейных моделей со стандартными параметрами для прогноза прочности при растяжении

Подберем гиперпараметры для моделей при помощи GridSearchCV.

	Модель	MSE модели	r2 модели
3	SGDRegressor_GridSearch	0.02908	-0.03112
0	DummyRegressor	0.02912	-0.03235
2	Ridge_GridSearch	0.02912	-0.03251
4	Lasso_GridSearch	0.02912	-0.03235
1	lr_GridSearch	0.02959	-0.04987

Рисунок 6 - Результат работы линейных моделей после подбора гиперпараметров для прогноза модуля упругости при растяжении

	Модель	MSE модели	r2 модели
2	Ridge_GridSearch	0.02960	-0.01790
1	lr_GridSearch	0.02973	-0.02471
3	SGDRegressor_GridSearch	0.02992	-0.02670
0	DummyRegressor	0.03010	-0.03230
4	Lasso_GridSearch	0.03010	-0.03230

Рисунок 7 - Результат работы линейных моделей после подбора гиперпараметров для прогноза прочности при растяжении

После подбора гиперпараметров метрики улучшились незначительно. Все модели крайне плохо описывают исходные данные.

2.2.2 Обучение моделей деревья принятия решений

Были обучены 2 модели со стандартными параметрами: DecisionTreeRegressor и RandomForestRegressor. Метрики всех моделей получились хуже чем метрика модели предсказывающей среднее.

	Модель	MSE модели	r2 модели
0	DummyRegressor	0.02912	-0.03235
2	Forest	0.03042	-0.08125
1	Tree	0.06523	-1.39937

Рисунок 8 - Результат работы моделей деревья принятия решений со стандартными параметрами для прогноза модуля упругости при растяжении

	Модель	MSE модели	r2 модели
0	DummyRegressor	0.03010	-0.03230
2	Forest	0.03016	-0.03754
1	Tree	0.06233	-1.14684

Рисунок 9 - Результат работы моделей деревья принятия решений со стандартными параметрами для прогноза прочности при растяжении

Подберем гиперпараметры для моделей при помощи GridSearchCV.

	Модель	MSE модели	r2 модели
0	DummyRegressor	0.02912	-0.03235
2	Forest_GridSearch	0.02917	-0.03481
1	Tree_GridSearch	0.02932	-0.04028

Рисунок 10 - Результат работы моделей деревья принятия решений после подбора гиперпараметров для прогноза модуля упругости при растяжении

	Модель	MSE модели	r2 модели
2	Forest_GridSearch	0.02998	-0.02882
0	DummyRegressor	0.03010	-0.03230
1	Tree_GridSearch	0.03076	-0.05514

Рисунок 11 - Результат работы моделей деревья принятия решений после подбора гиперпараметров для прогноза прочности при растяжении

Вывод:

2.2.3 Обучение метод опорных векторов

Обучим алгоритм SVR.

	Модель	MSE модели	r2 модели
0	DummyRegressor	0.02912	-0.03235
2	svr_lin	0.02966	-0.05191
1	svr_rbf	0.03391	-0.21414

Рисунок 12 - Результат работы SVR со стандартными параметрами для прогноза модуля упругости при растяжении

	Модель	MSE модели	r2 модели
2	svr_lin	0.02973	-0.02311
0	DummyRegressor	0.03010	-0.03230
1	svr_rbf	0.03323	-0.14285

Рисунок 13 - Результат работы SVR со стандартными параметрами для прогноза прочности при растяжении

Подберем гиперпараметры для моделей при помощи GridSearchCV.

	Модель	MSE модели	r2 модели
1	svr_rbf_GridSearch	0.02909	-0.03202
2	svr_lin_GridSearch	0.02909	-0.03202
0	DummyRegressor	0.02912	-0.03235

Рисунок 14 - Результат работы SVR после подбора гиперпараметров для прогноза модуля упругости при растяжении

	Модель	MSE модели	r2 модели
1	svr_rbf_GridSearch	0.03005	-0.02912
2	svr_lin_GridSearch	0.03005	-0.02912
0	DummyRegressor	0.03010	-0.03230

Рисунок 15 - Результат работы SVR после подбора гиперпараметров для прогноза прочности при растяжении

Вывод:

2.2.4 Обучение алгоритмов бустинга

Обучим алгоритмы XGBoost и CatBoostRegressor.

	Модель	MSE модели	r2 модели
0	DummyRegressor	0.02912	-0.03235
2	CatBoostRegressor	0.03290	-0.17191
1	XGBoost	0.03752	-0.34608

Рисунок 16 - Результат работы алгоритмов бустинга со стандартными параметрами для прогноза модуля упругости при растяжении

	Модель	MSE модели	r2 модели
0	DummyRegressor	0.03010	-0.03230
2	CatBoostRegressor	0.03203	-0.10788
1	XGBoost	0.03654	-0.26909

Рисунок 17 - Результат работы алгоритмов бустинга со стандартными параметрами для прогноза прочности при растяжении

Подберем гиперпараметры для моделей при помощи GridSearchCV.

	Модель	MSE модели	r2 модели
2	CatBoostRegressor_GridSearch	0.02904	-0.03064
1	XGBoost_GridSearch	0.02909	-0.03123
0	DummyRegressor	0.02912	-0.03235

Рисунок 18 - Результат работы алгоритмов бустинга после подбора гиперпараметров для прогноза модуля упругости при растяжении

	Модель	MSE модели	r2 модели
1	XGBoost_GridSearch	0.02888	0.00486
2	CatBoostRegressor_GridSearch	0.02912	-0.00261
0	DummyRegressor	0.03010	-0.03230

Рисунок 19 - Результат работы алгоритмов бустинга после подбора гиперпараметров для прогноза прочности при растяжении

вывод

2.2.5 Обучение нейронной сети

Обучим нейронную сеть.

Архитектура нейронной сети:

```
model_16.summary()
```

```
Model: "sequential_10"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_58 (Dense)            (None, 28)                336

 dense_59 (Dense)            (None, 28)                812

 dense_60 (Dense)            (None, 1)                 29

=================================================================
Total params: 1,177
Trainable params: 1,177
Non-trainable params: 0
_________________________________________________________________
```

Рисунок 20 - Архитектура нейронной сети для прогноза модуля упругости при растяжении

```
model_15.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_4 (Dense)             (None, 5)                 60

 dense_5 (Dense)             (None, 5)                 30

 dense_6 (Dense)             (None, 5)                 30

 dense_7 (Dense)             (None, 1)                 6

=================================================================
Total params: 126
Trainable params: 126
Non-trainable params: 0
_________________________________________________________________
```

Рисунок 21 - Архитектура нейронной сети для прогноза прочности при растяжении

Модель	MSE модели	r2 модели
Neural network	0.0261	-0.0135

Рисунок 22 - Результат работы нейронной сети для прогноза модуля упругости при растяжении

	Модель	MSE модели	r2 модели
	Neural network_1	0.0296	-0.0015

Рисунок 23 - Результат работы нейронной сети для прогноза прочности при растяжении

2.2.6 Выбор лучшей модели

Результаты работы всех моделей были добавлены в таблицу.

	Модель	MSE модели	r2 модели
21	Neural network	0.0261	-0.0135
12	Forest_GridSearch	0.0290	-0.0302
20	CatBoostRegressor_GridSearch	0.0290	-0.0306
19	XGBoost_GridSearch	0.0291	-0.0312
16	svr_lin_GridSearch	0.0291	-0.0320
15	svr_rbf_GridSearch	0.0291	-0.0320
8	Lasso_GridSearch	0.0291	-0.0323
0	DummyRegressor	0.0291	-0.0323
6	Ridge_GridSearch	0.0291	-0.0325
3	Lasso	0.0291	-0.0323

Рисунок 24 - Результат работы 10 лучших моделей для прогноза модуля упругости при растяжении

	Модель	MSE модели	r2 модели
19	XGBoost_GridSearch_1	0.0289	0.0049
20	CatBoostRegressor_GridSearch_1	0.0291	-0.0026
21	Neural network_1	0.0296	-0.0015
6	Ridge_GridSearch_1	0.0296	-0.0179
1	lr_1	0.0297	-0.0247
2	Ridge_1	0.0297	-0.0229
5	lr_GridSearch_1	0.0297	-0.0247
14	svr_lin_1	0.0297	-0.0231
7	SGDRegressor_GridSearch_1	0.0299	-0.0267
16	svr_lin_GridSearch_1	0.0300	-0.0291

Рисунок 25 - Результат работы 10 лучших моделей для прогноза прочности при растяжении

Результат работы всех моделей неудовлетворительный, все модели крайне плохо предсказывают как «Модуль упругости при растяжении, Гпа» так и «Прочность при растяжении, МПа». Самая лучшая модель дает коэффициент детерминации близкий к нулю, что соответствует базовой модели. Все выбранные модели не подходят для данных.

2.3. Обучение нейронной сети для рекомендации соотношение матрица-наполнитель

Напишем нейронную сеть для рекомендации соотношение матрица-наполнитель.

Архитектура модели состоит из пяти слоёв Dense: входного, трех скрытых полносвязных и выходного. Функция активации 'relu' для скрытых слоев. оптимизатор: Adam, loss-функция: 'mean_squared_error'. Входной слой с 12

нейронами по количеству признаков. Выходной слой с 1 нейроном для 1 признака. Нейронов в скрытых слях: 64, 64 и 32. Количество эпох обучения - 100, batch_size - 60.

```
model_80.summary()
```

```
Model: "sequential_79"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_470 (Dense)           (None, 64)                832

 dense_471 (Dense)           (None, 64)                4160

 dense_472 (Dense)           (None, 32)                2080

 dense_473 (Dense)           (None, 1)                 33

=================================================================
Total params: 7,105
Trainable params: 7,105
Non-trainable params: 0
_________________________________________________________________
```

Рисунок 26 - Архитектура нейронной сети для рекомендации соотношение матрица-наполнитель

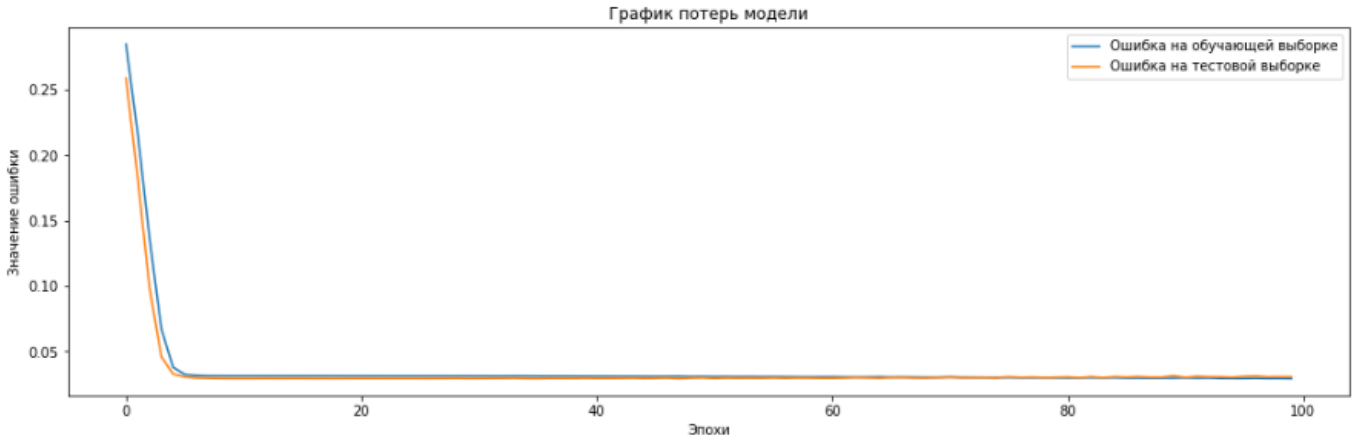

Рисунок 27 - График потерь

Рисунок 28 - Визуализация тест/прогноз и график потерь модели

Результат работы нейронной сети неудовлетворительный. Возможно, необходимо подобрать другую архитектуру нейронной сети.

2.4. Разработка приложения

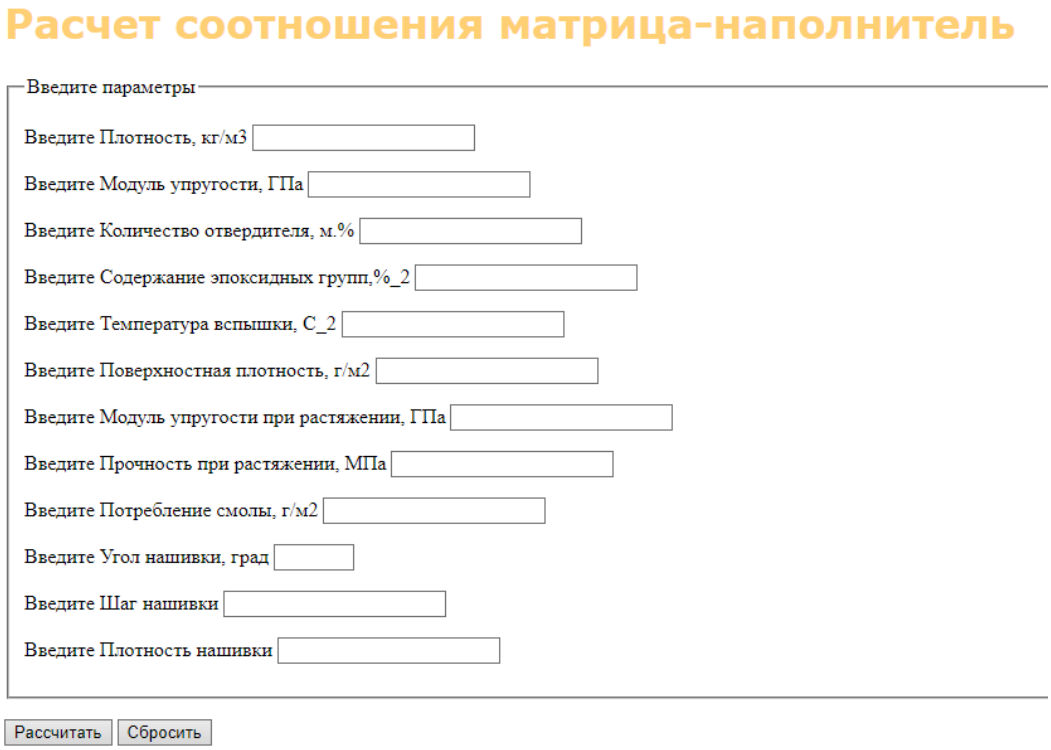

Рисунок 29 - Интерфейс приложения для расчета

Было разработано приложение для прогноза «Соотношение матрица-наполнитель» на основе разработанной нейронной сети. Приложение приложение написано при помощи фреймворка Flask. Для прогноза необходимо ввести 12 основных параметров. На вкладке main необходимо выбрать кнопку

«Прогнозирование значения соотношение матрица-наполнитель», после чего пользователь увидит экран с запросом данных. Для работы приложения необходимо запустить app.py, затем зайти в браузер и ввести адрес прописанный в командной строке приложения.

2.5. Создание удаленного репозитория

Был создан репозиторий на GitHub, который находится по адресу: https://github.com/k5k15/-VKR.git

В репозитории были загружены все результаты работы по прогнозированию конечных свойств новых материалов (композиционных материалов).

Заключение

Качество предсказаний всех моделей получилось низкое. Не удалось найти модели, которые хорошо бы предсказывали целевые признаки.

Для обучения мы использовали несколько моделей:

- линейные модели;
- деревья принятия решений;
- алгоритмы бустинга;
- метод опорных векторов;
- нейронная сеть.

Большинство моделей показали метрики близкие или такие же как у модели предсказывающей среднее.

Для успешного прогнозирования данных недостаточно, около 1000 строк это слишком мало, так как даже удаление выбросов повлияло на метрику в худшую сторону, возможно также, недостаточно признаков. Также необходима дополнительная информации о зависимости признаков с точки зрения физики процесса. Возможно, стоит применить более сложные модели для прогнозирования.

Библиографический список

1. Лекционный материал

2. Крис Элбон. Машинное обучение с использованием Python. Сборник рецептов. - Санкт-Петербург: БХВ-Петербург, 2019 — 384 с.: ил.

3. Франсуа Шолле. Глубокое обучение на Python. – СПб.: Питер, 2019. – 397 с.: ил.

4. Руководство для начинающих по выбору прогностических моделей машинного обучения в Python: - Режим доступа: https://machinelearningmastery.ru/

5. Scikit-learn: машинное обучение на Python - Режим доступа: https://scikit-learn.org/stable/index.html/

6. Учебник по машинному обучению - Режим доступа: https://academy.yandex.ru/handbook/ml/