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 Новицкий Павел Романович

Москва, 2022

**Содержание**

Содержание 2

Введение 3

1. Аналитическая часть 4

1.1. Постановка задачи 4

1.2. Описание используемых методов 6

1.3. Разведочный анализ данных 11

2. Практическая часть 13

2.1. Предобработка данных 16

2.2. Разработка и обучение модели 16

2.3. Тестирование модели 20

2.4. Нейронная сеть 20

2.5. Разработка приложения 22

2.6. Репозиторий и результаты 23

Заключение 24

Библиографический список 25

Приложения **Ошибка! Закладка не определена.**

**Введение**

Пояснительная записка подготовлена в рамках выпускной квалификационной работы по курсу «Data Science» на тему «Прогнозирование конечных свойств новых материалов (композиционных материалов)».

Композиционные материалы представляют собой сложные химические соединения. Они по-настоящему являются материалами будущего, хотя данный метод использовался ещё с времён Месопотамии и Древнего Египта, когда для постройки жилищ использовалась смесь глины и соломы, а первый «осмысленный» композиционный материал – железобетон – был создан ещё в конце XIX века, но используется до сих пор. Таким образом, первыми «композиционными материалами» была просто смесь двух или более компонентов, которые буквально вручную соединялись в один.

Тем не менее, использование лучших характеристик различных материалов получило широкое развитие и вышло на новый уровень – применение наукоёмких технологий и химических расчётов. Это позволило комбинировать высокую прочность с лёгкостью, упругость с высокой жароустойчивостью и так далее. Также это позволяет создавать синтетические материалы, постепенно уходя от использования натуральных, которые характеризуются сложностью их добычи и конечностью использования. Таким образом, композиционные материалы могут не только превосходить натуральные по своим характеристикам, но и во многих случаях иметь меньшею стоимость.

В наше время большинство композиционных материалов уникальны, так как создаются под определённые задачи заказчика. В то же время, это создаёт новую трудность, заключающуюся в выборе необходимого материала из предложенного множества или создания нового. В связи с этим заказчиком была предложена задача по предсказанию практических свойств материала, опираясь на некоторые его характеристики.

Для решения поставленной задачи изучены теоретические основы предметной области, изучены материалы, представленные заказчиком.

Проведены исследования и анализ предоставленных заказчиком данных с использованием методов, изученных на курсе «Data Science».

**1. Аналитическая часть**

**1.1. Постановка задачи**

Постановка задачи предоставлена заказчиком:

**Описание:**

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).

**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.

**Датасет** со свойствами композитов.

Число входных и выходных переменных:

**Входные переменные:**

1. Угол нашивки, град
2. Шаг нашивки
3. Плотность
4. Соотношение матрица-наполнитель
5. Плотность, кг/м3
6. Модуль упругости, Гпа
7. Количество отвердителя, м
8. Содержание эпоксидных групп,%\_2
9. Температура вспышки, С\_2
10. Поверхностная плотность, г/м2
11. Модуль упругости при растяжении, Гпа
12. Прочность при растяжении, Мпа
13. Потребление смолы, г/м2

**Выходные переменные:**

В зависимости от решаемой задачи выходными переменными становятся:

* Модуль упругости при растяжении, Гпа,
* Прочность при растяжении, Мпа,

При этом, становясь выходной переменной, соответствующая переменная исключается из входных данных.

**1.2. Описание используемых методов**

Задачи решались с использованием следующих методов и библиотек Python:

* 1. NumPy - это общедоступный модуль для python, который предоставляет общие математические и числовые операции в виде пре-скомпилированных, быстрых функций, обеспечивает функционал, который можно сравнить с функционалом MatLab. NumPy (Numeric Python) предоставляет базовые методы для манипуляции с большими массивами и матрицами.
  2. Matplotlib — библиотека на языке программирования [Python](https://ru.wikipedia.org/wiki/Python" \o "Python) для визуализации данных [двумерной (2D) графикой](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D0%BA%D0%B0#%D0%94%D0%B2%D1%83%D0%BC%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D0%BA%D0%B0_(2D)) ([3D графика](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D0%BA%D0%B0#%D0%A2%D1%80%D1%91%D1%85%D0%BC%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D0%BA%D0%B0) также поддерживается). Получаемые изображения могут быть использованы в качестве иллюстраций в публикациях.
  3. [Pandas](http://pandas.pydata.org/) - данный пакет делает Python мощным инструментом для анализа данных. Пакет дает возможность строить сводные таблицы, выполнять группировки, предоставляет удобный доступ к табличным данным.
  4. Scikit-learn (sklearn) — это один из наиболее широко используемых пакетов Python для Data Science и Machine Learning. Он содержит функции и алгоритмы для машинного обучения: классификации, прогнозирования или разбивки данных на группы.
  5. [Seaborn](https://seaborn.pydata.org/) — это библиотека для создания статистических графиков на Python. Она основывается на [matplotlib](https://matplotlib.org/) и тесно взаимодействует со [структурами данных pandas](https://livecodestream.dev/post/how-to-work-with-pandas-in-python/).
  6. TensorFlow — открытая [программная библиотека](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) для [машинного обучения](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), разработанная компанией [Google](https://ru.wikipedia.org/wiki/Google_(%D0%BA%D0%BE%D0%BC%D0%BF%D0%B0%D0%BD%D0%B8%D1%8F)" \o "Google (компания)) для решения задач построения и тренировки [нейронной сети](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D0%BA%D1%83%D1%81%D1%81%D1%82%D0%B2%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BD%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C) с целью автоматического нахождения и классификации образов, достигая качества человеческого восприятия.
  7. Math  - является самым базовым математическим модулем Python. Охватывает основные математические операции, такие как сумма, экспонента, модуль и так далее. Эта библиотека не используется при работе со сложными математическими операциями, такими как умножение матриц.

Для решения задач регрессии были использованы два метода машинного обучения:

* 1. Линейная регрессия (Linear regression) — модель зависимости переменной x от одной или нескольких других переменных (факторов, регрессоров, независимых переменных) с линейной функцией зависимости. Линейная регрессия относится к задаче определения «линии наилучшего соответствия» через набор точек данных и стала простым предшественником нелинейных методов, которые используют для обучения нейронных сетей.
  2. Случайный лес (Random forest) -  алгоритм [машинного обучения](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), предложенный [Лео Брейманом](https://ru.wikipedia.org/wiki/%D0%91%D1%80%D0%B5%D0%B9%D0%BC%D0%B0%D0%BD,_%D0%9B%D0%B5%D0%BE) и [Адель Катлер](https://en.wikipedia.org/wiki/Adele_Cutler), заключающийся в использовании комитета (ансамбля) [решающих деревьев](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%BE_%D0%BF%D1%80%D0%B8%D0%BD%D1%8F%D1%82%D0%B8%D1%8F_%D1%80%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D0%B9). Алгоритм сочетает в себе две основные идеи: метод [бэггинга](https://ru.wikipedia.org/wiki/%D0%91%D1%8D%D0%B3%D0%B3%D0%B8%D0%BD%D0%B3" \o "Бэггинг) Бреймана, и [метод случайных подпространств](https://en.wikipedia.org/wiki/Random_subspace_method), предложенный [Тин Кам Хо](https://en.wikipedia.org/wiki/Tin_Kam_Ho). Алгоритм применяется для задач классификации, регрессии и кластеризации. Основная идея заключается в использовании большого [ансамбля](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D1%81%D0%B0%D0%BC%D0%B1%D0%BB%D1%8C_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D0%BE%D0%B2_(%D0%BE%D0%B1%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BC%D0%B0%D1%88%D0%B8%D0%BD)) [решающих деревьев](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%BE_%D0%BF%D1%80%D0%B8%D0%BD%D1%8F%D1%82%D0%B8%D1%8F_%D1%80%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D0%B9), каждое из которых само по себе даёт очень невысокое качество классификации, но за счёт их большого количества результат получается хорошим.
  3. Метод k-ближайших соседей (k Nearest Neighbors, или kNN) – популярный алгоритм классификации, который используется в разных типах задач машинного обучения. Наравне с деревом решений это один из самых понятных подходов к классификации. В данном методе используется евклидова метрика (евклидово расстояние, или же Euclidean distance) – метрика в евклидовом пространстве, расстояние между двумя точками евклидова пространства, вычисляемое по теореме Пифагора. Другими словами, это наименьшее возможное расстояние между точками A и B, где каждый элемент изначально рассматривается как отличный от всех остальных.

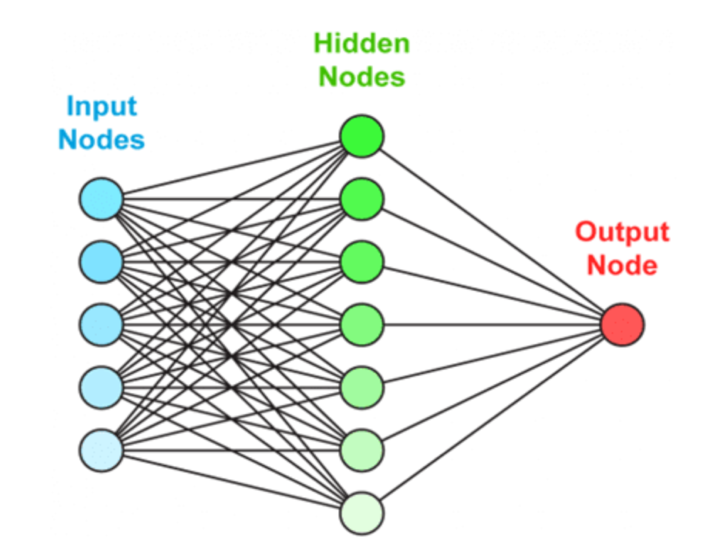
Также в ходе работы была разработана нейронная сеть Искусственная нейронная сеть (ИНС) – это математическая модель, а также её программное воплощение, построенная по принципу организации и функционирования биологических нейронных сетей, т.е. сетей нервных клеток живого организма. Любая нейросеть — это набор нейронов и связей между ними. Нейрон лучше всего представлять просто как функцию с кучей входов и одним выходом. Задача нейрона — взять числа со своих входов, выполнить над ними функцию и отдать результат на выход.

Типовая нейросеть состоит из трех компонентов:

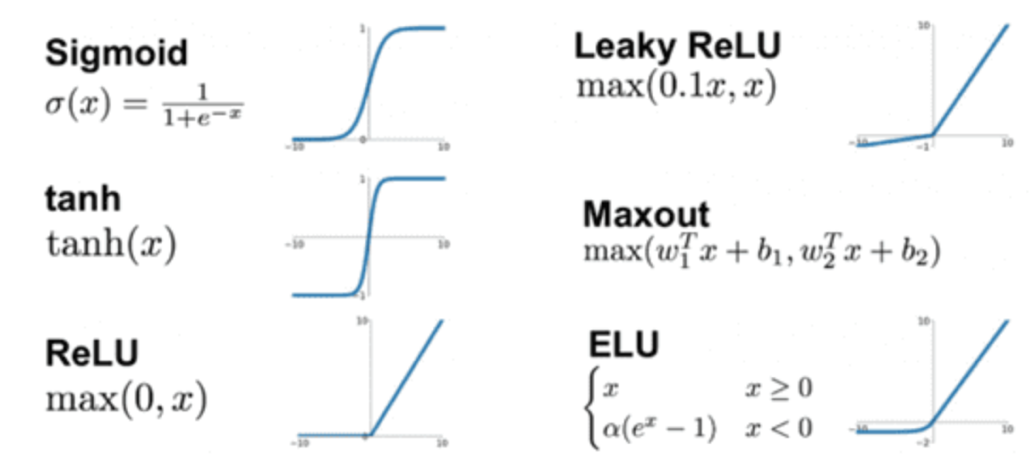
входной слой, нейроны которого принимают исходный вектор значений, кодирующий входной сигнал и передают его на последующий слой, усилив или ослабив;

промежуточные слои, которые выполняют основные вычислительные операции;

выходной слой, нейроны которого представляют собой выходы сети и иногда также могут производить какие-либо вычисления.

****

Каждый нейрон предыдущего слоя передает сигналы нейронам последующего методом прямого или обратного распространения ошибки по синоптическим связям с весовыми коэффициентами.

****

Выход каждого нейрона является результатом его нелинейной непрерывной и монотонной функции активации: сигмоиды, синусоиды, Гауссовой, ступенчатой и подобных им (рис. 3). Функция активации определяет зависимость сигнала на выходе нейрона от взвешенной суммы сигналов на его входах. Благодаря ее нелинейности нейросети с достаточно небольшим числом узлов и слоев могут решать довольно сложные задачи прогнозирования, распознавания и классификации. Для каждого типа решаемой задачи характерна своя функция активации.

**1.3. Разведочный анализ и предобработка данных**

Для анализа дата сета, в котором представлены данные о начальных свойствах компонентов композиционных материалов (количество связующего, наполнителя, температурный режим отвередения и т.д.) обратимся к возможностям библиотеки Pandas. С помощью данной библиотеки можно можно группировать таблицы, очищать и изменять данные, вычислять параметры и делать выборки. Данные были предоставлены в виде двух файлов EXCEL. Загрузить файлы можно с помощью команды

x\_bp = pd.read\_excel(r'C:\my\_path\X\_bp.xlsx', index\_col=0)

x\_nup = pd.read\_excel(r'C:\my\_path\X\_nup.xlsx', index\_col=0)

Данные для решения поставленных задач представлены в виде двух датасетов разной длины. Посмотреть, как они выглядит, можно с помощью метода .head(), который по умолчанию выводит пять первых строк. Получили 2 таблицы:

Таблица 1

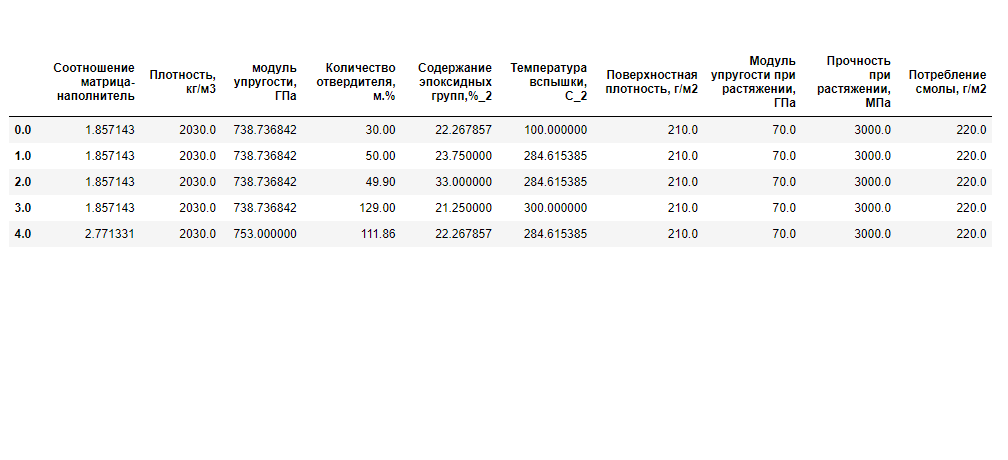


Таблица 2



С помощью команды x\_nup.info(), x\_bp.info() была выведена информация по каждому из датафреймов. Итого размерность датафрейма x\_bp составляет 1023 строки и 10 стобцова, датафрейма x\_nup составляет 1040 строк и 3 столбца.

Следующим шагом необходимо объединить два датафрейма в один. Объединение было предложено сделать по индексу тип объединения INNER. Используем следующую команду:

df1 = x\_bp.merge(x\_nup, how = 'inner', left\_index = True, right\_index=True)

Таким образом, для решения задач датасеты были предварительно объединены и приведены к одной длине в 1023 строки. Данные одного датасета были после объединения обрезаны 17 строк., что составило 1,6% данных и явилось не существенным.

Чтобы получить первичное представление о статистических характеристиках нашего объединенного датасета, достаточно команды df1.describe()

Таблица 3

Изображение выглядит как стол

Автоматически созданное описание

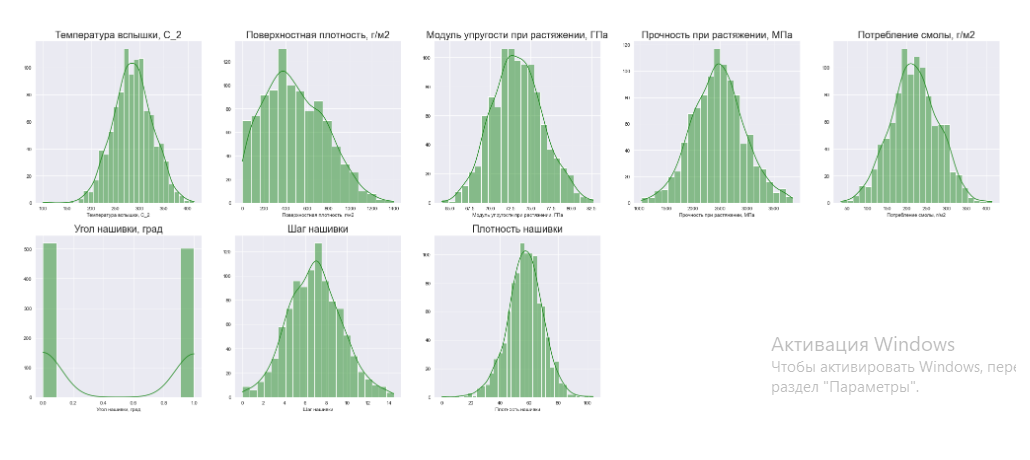
С помощью df1.info() мы увидим сколько значений в каждом столбце (в нашем случае в столбцах нет пропущенных значений) и формат данных.

Также из описательной статистики мы видим, что в столбце «Угол нашивки» присутствуют только два значения: «90» и «0». Таким образом, мы можем сразу же преобразовать значения и привести их к 0 и 1 до нормализации основного массива данных.

Следующим шагом были построены гистограммы плотности распределения значений каждой переменной. Гистограммы отображают числовые величины на одной оси и переменные категории на другой. Они позволяют увидеть значения параметров для каждой категории. Для большинства характеристик распределение близко к нормальному. Исключение составляют:

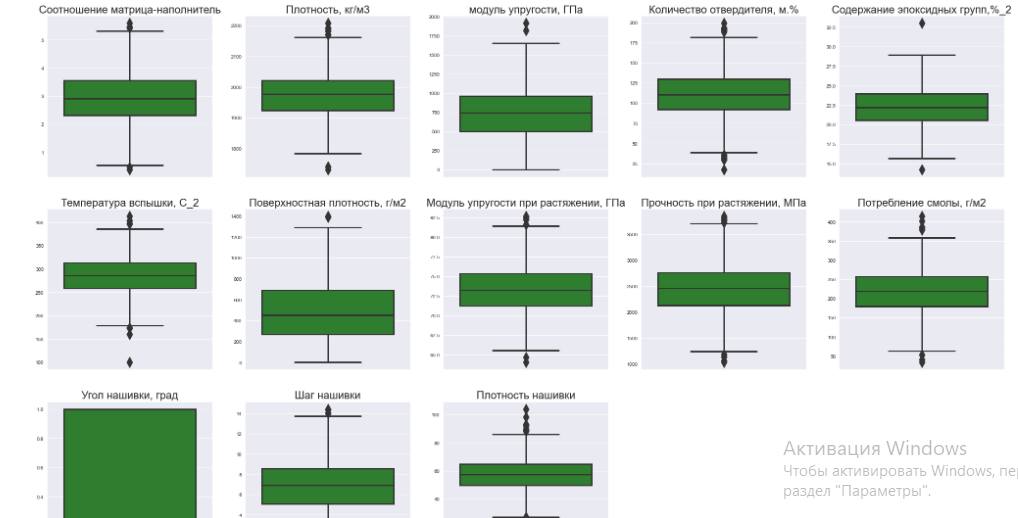
* - поверхностная плотность, г/м2 - нормальное распределение со смещением влево, что свидетельствует о большем количестве измерений с меньшим показателем поверхностной плотности.
* Угол нашивки, град - дискретное распределение, так как колонка содержит всего два уникальных значения. Выборочные результаты представлены на рисунке 1.

Рисунок 1



Для поиска выбросов построены диаграммы Ящика с усами. Выборочные результаты представлена на рисунке 2.

Рисунок 2



Как показано на рисунке 2, в выборке присутствует незначительное количество выбросов. Тем не менее, данные выбросы могут помешать исследованию, а также затруднить работу нейросети, поэтому исследователем было принято решение избавиться от выбросов.

for col in df.columns:

q75,q25 = np.percentile(df.loc[:,col],[75,25])

intr\_qr = q75-q25

max = q75+(1.5\*intr\_qr)

min = q25-(1.5\*intr\_qr)

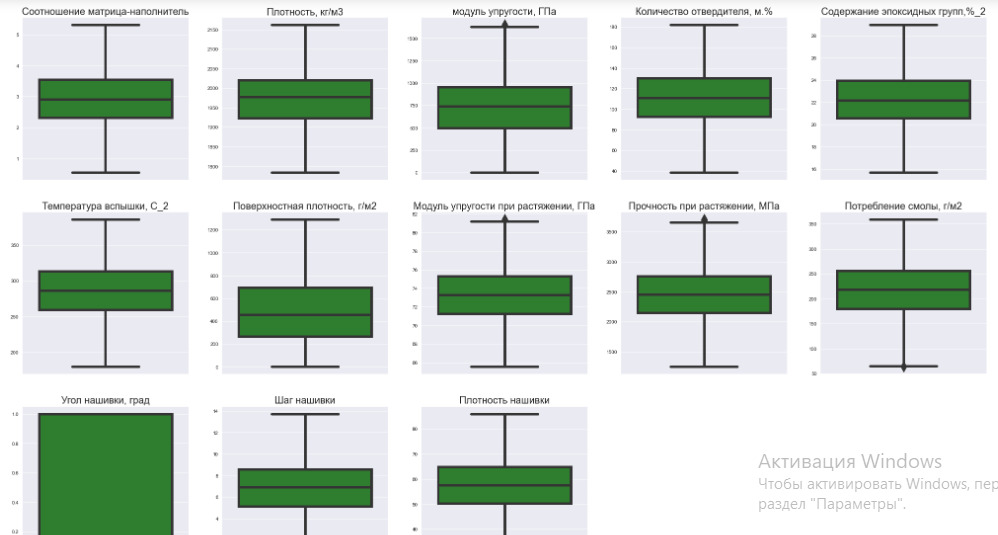
df.loc[df[col] < min,col] = np.nan

df.loc[df[col] > max,col] = np.nan

df = df.dropna(axis=0)

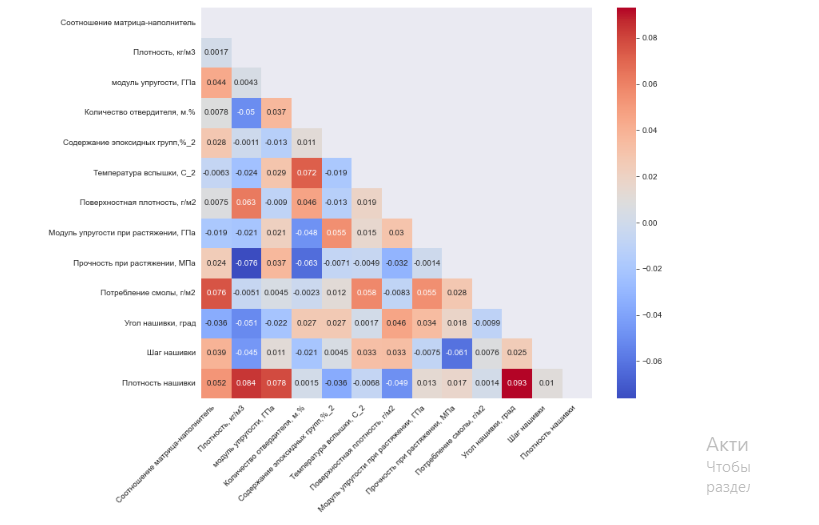
Была повторно построена гистограмма ящик с усами, чтобы проверить результат очистки. Результат представлен на рисунке 3.

Рисунок 3



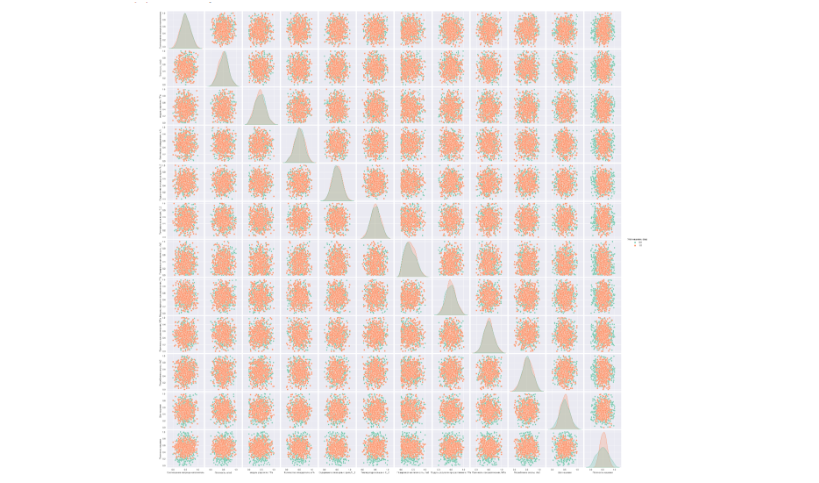
Далее для исследования данных были построены попарные графики точек рассеяния, а также тепловая карта коэффициентов корреляции.

Рисунок 4



Максимальная корреляция между Плотностью нашивки и углом нашивки и составляет 0.093, что говорит об отсутствии зависимости между этими данными. Корреляция между всеми параметрами очень близка к 0, что говорит об отсутствии корреляционных связей между переменными.

Рисунок 5



Одним из выводов может быть следующим. Представленный детасет был предобработан заказчиком и оставшиеся переменные являются независимыми и каждая из них несет свою функцию при влиянии на композитный материал.

На этом разведочный анализ данных был завершен и исследователь перешел к предобработке данных.

**2. Практическая часть**

**2.1. Разработка и обучение модели**

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

* Модель k ближайших соседей
* Модель линейной регрессии
* Модель случайный лес.

Для обучения модели k ближайших соседей была использована функция:

KNeighborsRegressor

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

В качестве параметров были заданы:

'weights' : ['uniform', 'distance'],

'algorithm' : ['auto', 'ball\_tree', 'kd\_tree', 'brute']

В качестве метрик оценки качества был выбран R2

Для обучения моделей линейной регрессии была использована функция:

LinearRegression()

В качестве метрик оценки качества были выбраны MAE и R2.

Для обучения моделей линейной регрессии была использована функция:

RandomForestRegressor

Для обучения моделей случайный лес были настроены гиперпараметры модели:

Количество деревьев:

'n\_estimators' : range(10, 1000, 10)

В качестве критериев были выбраны следующие:

'criterion' : ['squared\_error', 'absolute\_error', 'poisson'],

Глубина дерева (количество ветвлений или уровней деревьев:

'max\_depth' : range(1, 7),

'min\_samples\_split' : range(20, 50, 5),

'min\_samples\_leaf' : range(2, 8),

Так как данные модели плохо справлялись с поставленной задачей была произведена попытка построения многослойного перцептрона с помощью библиотеки sklearn.

Многослойный персептрон - частный случай персептрона Розенблатта, в котором один алгоритм обратного распространения ошибки обучает все слои. Название по историческим причинам не отражает особенности данного вида персептрона, то есть не связано с тем, что в нём имеется несколько слоёв. ). Особенностью является наличие более чем одного обучаемого слоя (как правило — два или три). Необходимость в большом количестве обучаемых слоёв отпадает, так как теоретически единственного скрытого слоя достаточно, чтобы перекодировать входное представление таким образом, чтобы получить линейную разделимость для выходного представления.

Для каждой модели были определены ошибки:

Таблица 4

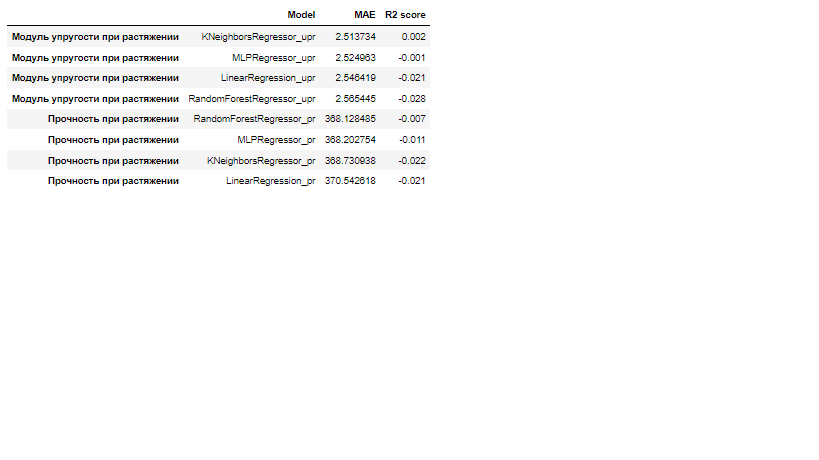


Рисунок 6

Средняя абсолютная ошибка моделей прогноза Модуля упругости при растяжении

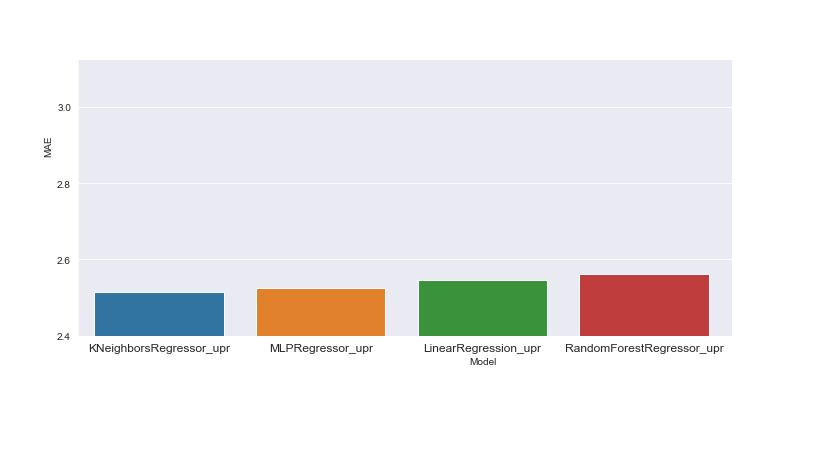
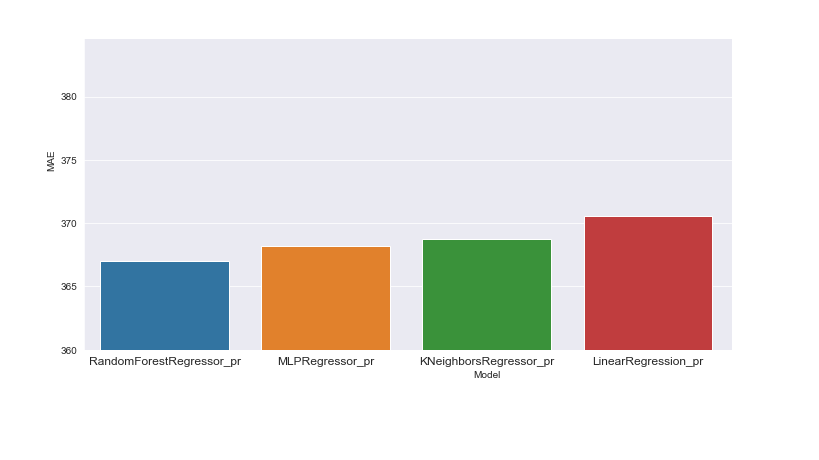


Рисунок 7

Средняя абсолютная ошибка моделей прогноза Модуля прочности при растяжении



Так как данные плохо коррелированы, а также учитывая малый опыт практического применения моделей обучения исследователем, модели показывают очень плохие результаты обучения.

**2.2. Нейронная сеть**

Для построения модели для рекомендательной системы была построена нейронная сеть с помощью библиотеки tensorflow.

На вход подавались нормализованные данные, модель включала один входной слой с активационной функцией LeakyRelu и 128 нейронов. Далее были настроены четыре основных слоя:

-один слой с 128 нейронами

-один слой с 64 нейронами

-один слой с 32 нейронами

-один слой с 16 нейронами

И далее выходной слой с 1 нейроном, так как на выходе выводится одно значение для введенных данных.

Для оптимизации был применен стохастический оптимизатор градиентного спуска с помощью функции optimizers.SGD.

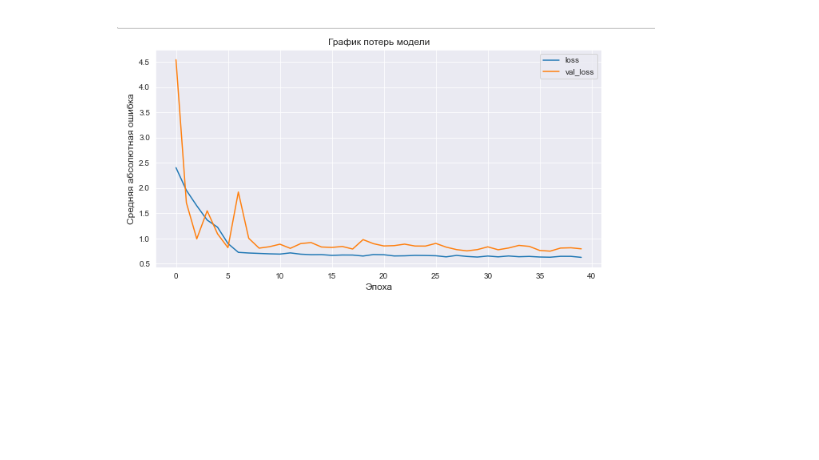
А также для оценки качества модели был применена метрика MАE – средняя абсолютная ошибка.

Вышеперечисленные параметры являются гиперпараметрами настраиваемой модели.

Модель обучалась на 40 эпохах. Большее количество эпох не дало лучшего результата. Датасет не является очень большим и модель может обучаться на меньшем количестве эпох.

Результаты обучения модели характеризуются изменением средней абсолютной ошибки, представленном на рисунке:

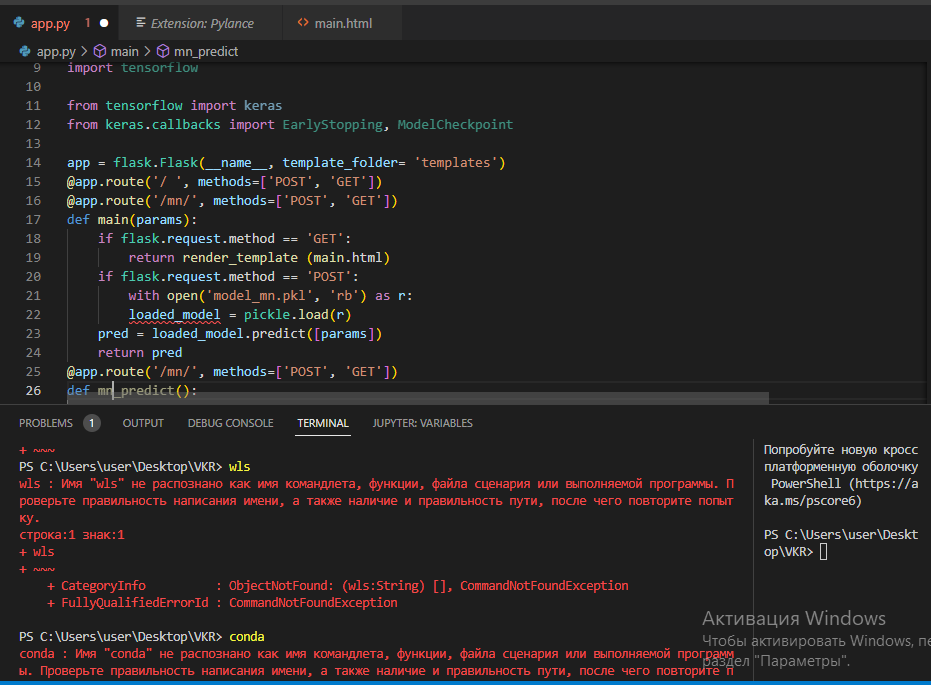
Рисунок 8



**2.3. Разработка приложения**

Также была предпринята попытка разработки веб-приложения Flask в среде разработки Visual Studio. Однако ввиду недостаточной компетентности исследователя попытка оказалась неудачной.

Рисунок 9



**2.6. Репозиторий и результаты**

Создан репозиторий в GitLab, где размещен код исследования и файл с пояснительной запиской.

Страница слушателя: <https://gitlab.com/pavelnovithsky98/>

Созданный репозиторий: <https://gitlab.com/pavelnovithsky98/my-pathetic-try/-/tree/main>

**Заключение**

По результатам исследования были изучены теоретические основы методов машинного обучения, изучены основные библиотеки Python, как одного из основных инструментов для работы аналитика данных.

На основании практической задачи были апробированы методы машинного обучения и построения моделей для решения задач аналитики.

Результаты построения и обучения моделей не дали положительного результата, но позволили приобрести опыт по выбору модели для решения задач регрессии, опыт по настройке таких моделей, опыт по оценке качества моделей и расчета различных метрик, характеризующих качество построенной модели, позволили оценить временной фактор на проведение подобного рода исследования. Полученные в ходе проведения исследования данные так же позволили определить вектор дальнейшего познания глубин машинного обучения для дальнейшего применения их на практике.

**Библиографический список**

* 1. Документация по библиотеке numpy: – Режим доступа: <https://numpy.org/doc/1.22/user/index.html#user>.
  2. Документация по библиотеке pandas: – Режим доступа: <https://pandas.pydata.org/docs/user_guide/index.html#user-guide>.
  3. Документация по библиотеке matplotlib: – Режим доступа: <https://matplotlib.org/stable/users/index.html>.
  4. Документация по библиотеке seaborn: – Режим доступа: <https://seaborn.pydata.org/tutorial.html>.
  5. Документация по библиотеке sklearn: – Режим доступа: <https://scikit-learn.org/stable/user_guide.html>.
  6. Документация по библиотеке keras: – Режим доступа: <https://keras.io/api/>.