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 Савельев Петр Олегович

Москва, 2022

Содержание

Введение

Композитный материал (далее КМ) — многокомпонентный материал, изготовленный (человеком или природой) из двух или более компонентов с существенно различными физическими и/или химическими свойствами, которые, в сочетании, приводят к появлению нового материала с характеристиками, отличными от характеристик отдельных компонентов и не являющимися простой суперпозицией. В составе композита принято выделять матрицу/матрицы и наполнитель/наполнители, последние выполняют функцию армирования (по аналогии с арматурой в таком композиционном строительном материале, как железобетон). В качестве наполнителей композитов как правило выступают углеродные или стеклянные волокна, а роль матрицы играет полимер. Сочетание разных компонентов позволяет улучшить характеристики материала и делает его одновременно лёгким и прочным. При этом отдельные компоненты остаются таковыми в структуре композитов, что отличает их от смесей и затвердевших растворов. Изменяя состав матрицы и наполнителя, их соотношение, ориентацию наполнителя, получают широкий спектр материалов с требуемым набором свойств. Многие композиты превосходят обычные материалы и сплавы по своим механическим свойствам. Использование композитов обычно позволяет уменьшить массу конструкции при сохранении или улучшении её механических характеристик.

По типу материала матрицы КМ могут быть:

1. полимерные (термопласты, реактопласты, смеси);
2. металлические (в том числе материалы, получаемые методами порошковой металлургии, и сплавы, состоящие из макронеоднородных фаз);
3. неорганические (неорганические полимеры, минералы, углерод, керамика);
4. комбинированные (полиматричные).

Базальт – горная порода, составляющая 30% земной коры, его запасы не исчерпаемы. Базальты образуются в результате затвердевания расплава силикатной магмы.

Базальтопластик - композитный материал на основе базальтовых волокон и органического связующего вещества. Базальтопластик конкурирует с металлическими изделиями, и превосходит их по щелочной, коррозионной устойчивости.

Области использования композиционных материалов практически ничем не ограничены. И сегодня КМ нашли применение во многих отраслях промышленности.

Оптимизация состава базируется на экспериментальном определении тех или иных характеристик композита путем испытания образцов разного состава. Объем экспериментальных работ зависит от количества определяемых характеристик и точности их оценки.

1. Аналитическая часть

1.1 Постановка задачи

Для анализа нам даны два файла X_bp.xlsx и X_nup.xlsx, загружаем и смотрим размерность полученных датасетов, в первом 1023 строки и 11 столбцов во втором 1040 строк и 4 столбца, смотрим размерность, на рисунке 1.

```
[ ]  # загружаем файлы
     df1 = pd.read_excel('X_bp.xlsx')
     df2 = pd.read_excel('X_nup.xlsx')
     # сморим размерность
     df1.shape,df2.shape

     ((1023, 11), (1040, 4))
```

Рисунок 1.

Задача состоит в том, что нам нужно разработать модели для прогноза модуля упругости при растяжении, прочности при растяжении и соотношения матрица-наполнитель. Мы имеем два файла представленные на рисунке 1. Для дальнейшей обработки данных и ее исследования нам необходимо объединить имеющиеся два датасета в один, у нас один датасет больше другого. Устраняем строки без индекса (часть информации не имеет строк в таблице свойств используемых компонентов композитов и соотношений).

Анализируем таблицу смотрим какие признаки присутствуют выведем описательную статистику получим среднее значение, сделаем подсчет уникальных значений по столбцам, проверим на дубликаты, проверим данные на выбросы, пропуски. Также удалим шумы.

Проведем разведочный анализ данных, нарисуем гистограммы распределения каждой из переменной, также отрисуем диаграммы ящика с усами и попарные

графики рассеяния и визуализируем данные с помощью матрицы до удаления выбросов. Удалим методом межквартильных растояний и посмотрим на ящик с усами после проделанных манипуляций. Проведем нормализацию, стандартизацию, определим входные и выходные переменные

Для нас важно подготовить/нормализовать/стандартизировать данные таким образом чтобы они давали разным алгоритмам машинного обучения наилучшую возможность решения нашей задачи, для того чтобы наша модель имела возможность работать со всеми признаками и чтобы эти признаки были в одном диапазоне. Приступим к написанию, обучению моделей для прогноза упругости при растяжении и прочности при растяжении и нейросети для прогноза соотношения «матрица-наполнитель». Проведем оценку проделанной работы. Напишем readme.txt. Создадим репозиторий в гитхаб и выложим код проведенной работы.

1.2 Описание используемых методов

Задача которую нам надо решить в рамках классификации, классификация в машинном обучении относится к алгоритмам контролируемого обучения, обучения с учителем это также это задача регрессии.

В результате работы были применены следующие методы: Линейная регрессия, Регрессия k-ближайших соседей, SVR метод опорных векторов, Случайный лес.

Линейная регрессия в машинном обучении с учителем. Алгоритму для обучения необходимо указать как входные данные, так и заранее подготовленные выходные данные. Все вместе это называется обучающей выборкой. Преимущества линейной регрессии в том, что она весьма проста в реализации, но надо учитывать, что выбросы могут оказывать огромное влияние на регрессию.

k-ближайших соседей. Идея заключается в том, что классифицируемый образ относится к классу к которому принадлежит его ближайший сосед. Определяется **k** ближайший сосед к **x** образов и образ **x** зачисляется в тот класс к которому относится наибольшее число образов, входящих в эту группу. Плюсы данного алгоритма в том, что алгоритм прост и его легко понять, тривиальное обучение модели на новых данных, работает с любым количеством категорий в задачах класификации, модель принимает два параметра **k** и метрика расстояния, которой хотели бы воспользоваться обычно это эвклидово расстояние и имеет низкую чувствительность к выбросам. Минусы в том, что высокая стоимость вычисления, т.к. вам требуется обработать весь объем данных, работает не так хорошо с категорическими параметрами.

SVR метод опорных векторов заключается в поиске плоскости регрессии, чтобы все данные в коллекции были наиболее близки к плоскости. Это алгоритм обучения с учителем, использующихся для задач классификации и

регрессионного анализа, это контролируемое обучение моделей с использование схожих алгоритмов для анализа данных и распознавания шаблонов. Каждый объект данных представляется как вектор (точка) в p-мерном пространстве. Он создаёт линию или гиперплоскость, которая разделяет данные на классы. Плюсы данного метода в том, что достаточно небольшого набора данных эффективен при большом количестве гиперпараметров. Существует возможность гибко настраивать разделяющую функцию. Алгоритм максимизирует разделяющую полосу, которая, позволяет уменьшить количество ошибок классификации. Недостатки метода в том, что у метода неустойчивость к шуму, и шум напрямую влияет на построение разделяющей гиперплоскости, что не дает получить хороший результат. Требует много времени на обучение.

Случайный лес - это алгоритм классификации, состоящий из многих деревьев решений (ансамбль решающих деревьев), и это алгоритм машинного обучения с учителем. Он использует бэггинг и случайность признаков при построении каждого отдельного дерева, чтобы попытаться создать некоррелированный лес, прогноз которого точнее, чем у любого отдельного дерева. Плюсы метода в том, что имеет высокую точность предсказания, на большинстве задач будет лучше линейных алгоритмов; точность сравнима с точностью бустинга, практически не чувствителен к выбросам, не чувствителен к масштабированию (и вообще к любым монотонным преобразованиям) значений признаков, не требует тщательной настройки параметров, хорошо работает «из коробки». Минусы, в отличие от одного дерева, результаты случайного леса сложнее интерпретировать, алгоритм работает хуже многих линейных методов, когда в выборке очень много разреженных признаков, алгоритм склонен к переобучению на некоторых задачах, особенно на зашумленных данных, для данных, включающих категориальные переменные с различным количеством

уровней, случайные леса предвзяты в пользу признаков с большим количеством уровней: когда у признака много уровней, дерево будет сильнее подстраиваться именно под эти признаки, так как на них можно получить более высокое значение оптимизируемого функционала (типа прироста информации).

Градиентный бустинг - основная идея градиентного бустинга: строятся последовательно несколько базовых классификаторов, каждый из которых как можно лучше компенсирует недостатки предыдущих. Финальный классификатор является линейной композицией этих базовых классификаторов.

Достоинства метода: новые алгоритмы учатся на ошибках предыдущих; наблюдения выбираются на основе ошибки; прост в настройке темпа обучения и применения; легко интерпретируем.

Недостатки метода: необходимо тщательно выбирать критерии остановки, иначе это может привести к переобучению; наблюдения с наибольшей ошибкой появляются чаще; менее гибок чем нейронные сети.

1.3 Разведочный анализ данных

Разведочный анализ является обязательной процедурой в результате которой - получим представления о характере распределения данных, оценим качество исходных данных на наличие пропусков, выбросов. При разведочном анализе учитывается и сравнивается большое число признаков, закономерностей. Трудно работать с данными, не понимая, что они из себя представляют, стоит ли что то удалять или нет, пропущенные данные или искаженные, безосновательно удалять что-либо не в коем случае не стоит. Именно поэтому сначала набор данных надо изучить, загрузим и выведем некоторые статистики.

```
In [6]: df.shape,df.columns

Out[6]: ((1023, 14),
         Index(['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 растяжен
        ии, ГПа',
                'Прочность при растяжении, МПа', 'Потребление смолы, г/м2',
                'Угол нашивки, град', 'Шаг нашивки', 'Плотность нашивки'],
               dtype='object'))
```

Рисунок 2.

В разведочном анализе применяются оценка характеристик датасета, гистограммы распределения, диаграммы ящика с усами, попарные графики рассеяния точек, тепловая карта, анализ и удаление выбросов, пропусков, дубликатов.

```
In [8]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1023 entries, 0 to 1022
Data columns (total 13 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
 10  Угол нашивки, град                     1023 non-null   int64  
 11  Шаг нашивки                            1023 non-null   float64
 12  Плотность нашивки                      1023 non-null   float64
dtypes: float64(12), int64(1)
memory usage: 111.9 KB
```

Рисунок 3. Посмотрим информацию о размерности данных.

```
df.describe()
```

:

	Соотношение матрица-наполнитель	Плотность, кг/м3	модуль упругости, ГПа	Количество отвердителя, м.%	Содержание эпоксидных групп,%_2	Температура вспышки, С_2	Пове пл
count	1023.000000	1023.000000	1023.000000	1023.000000	1023.000000	1023.000000	1
mean	2.930366	1975.734888	739.923233	110.570769	22.244390	285.882151	
std	0.913222	73.729231	330.231581	28.295911	2.406301	40.943260	
min	0.389403	1731.764635	2.436909	17.740275	14.254985	100.000000	
25%	2.317887	1924.155467	500.047452	92.443497	20.608034	259.066528	
50%	2.906878	1977.621657	739.664328	110.564840	22.230744	285.896812	
75%	3.552660	2021.374375	961.812526	129.730366	23.961934	313.002106	
max	5.591742	2207.773481	1911.536477	198.953207	33.000000	413.273418	1

Рисунок 4. Описательная статистика.

```
#  Делаем подсчет уникальных значений по столбцам
df.nunique()
```

```
Соотношение матрица-наполнитель          1014
Плотность, кг/м3                         1013
модуль упругости, ГПа                    1020
Количество отвердителя, м.%              1005
Содержание эпоксидных групп,%_2          1004
Температура вспышки, С_2                 1003
Поверхностная плотность, г/м2            1004
Модуль упругости при растяжении, ГПа     1004
Прочность при растяжении, МПа            1004
Потребление смолы, г/м2                  1003
Угол нашивки, град                          2
Шаг нашивки                               989
Плотность нашивки                         988
dtype: int64
```

Рисунок 5. Сделаем подсчет уникальных значений по столбцам.

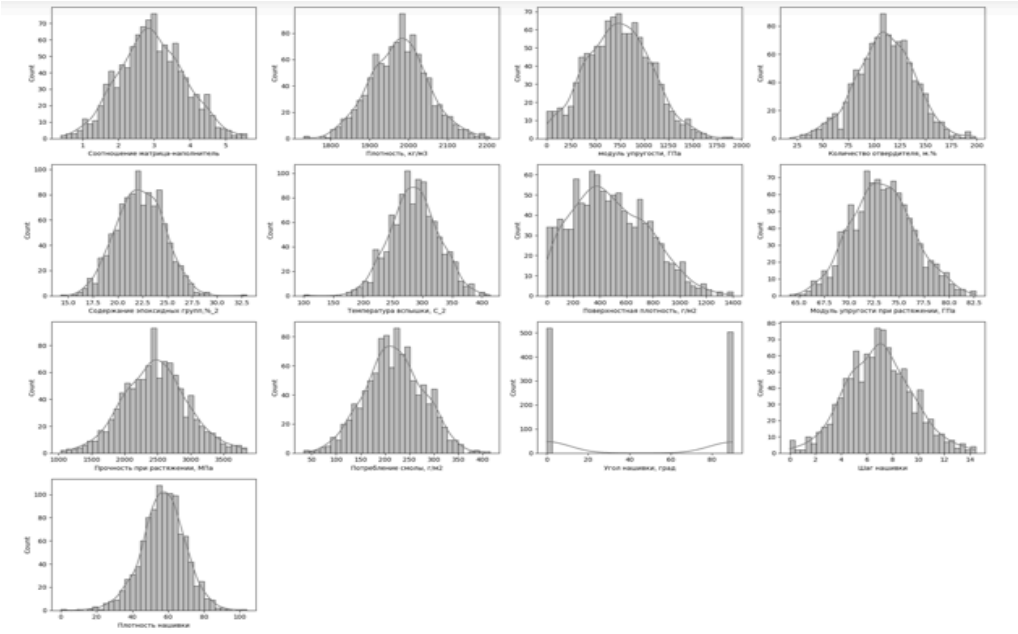


Рисунок 6. Гистограмма

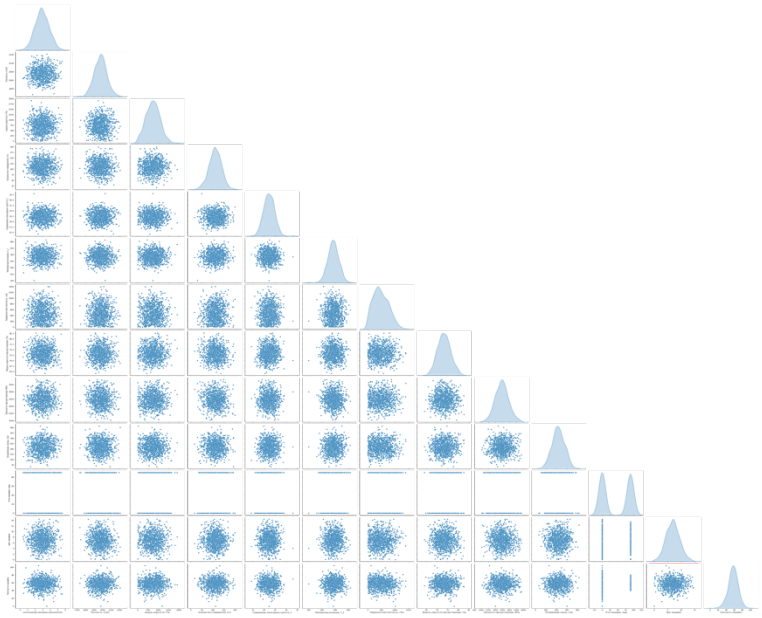


Рисунок 7. Попарные графики рассеяния точек. Наблюдаем отсутствие кореляции. Зависимость между показателями не линейная(метод линейной регрессии судя по увиденному не подходит).

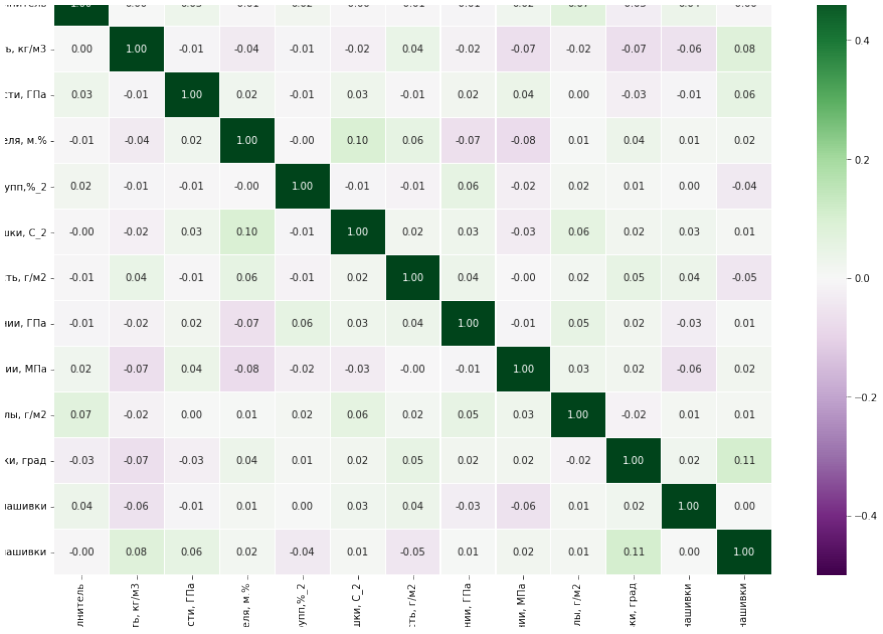
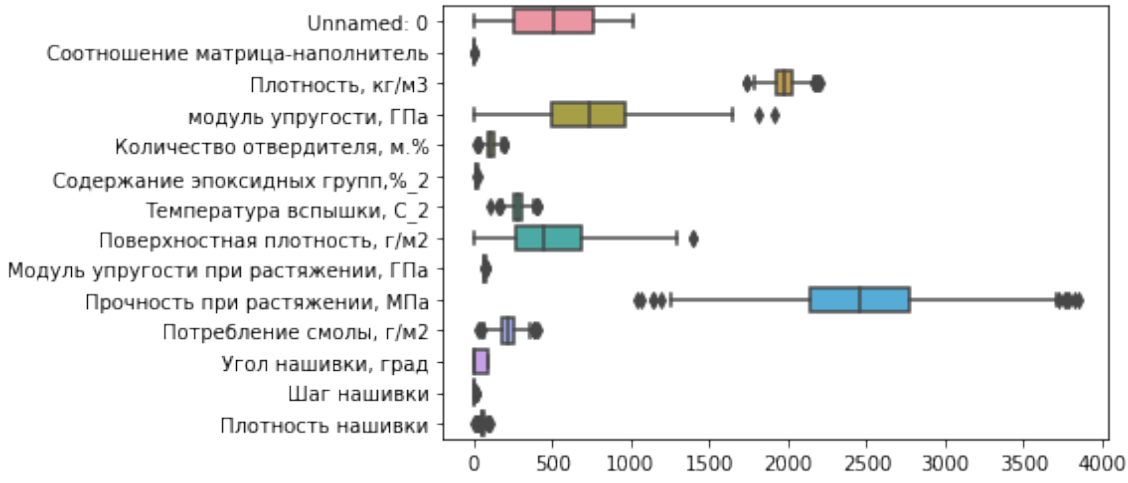


Рисунок 8. Визуализация корреляционной матрицы, тепловая карта.

Корреляция между всеми параметрами очень близка к нулю (Рисунок 8), максимальная корреляция между плотностью нашивки и углом нашивки и составляет 0.11. Все говорит об отсутствии корреляционных связей между переменными.

Далее используем график ящик с усами Boxplot (Рисунок 9) с помощью него мы сможем посмотреть компактное изображение одномерного распределения вероятностей. Такой вид диаграммы показывает медиану (среднее), нижний и верхний квартили и минимальное или максимальное значение выборки, также мо жно посмотреть выбросы.

Рисунок 8. Смотрим выбросы



Выбросы присутствуют (Рисунок 9)

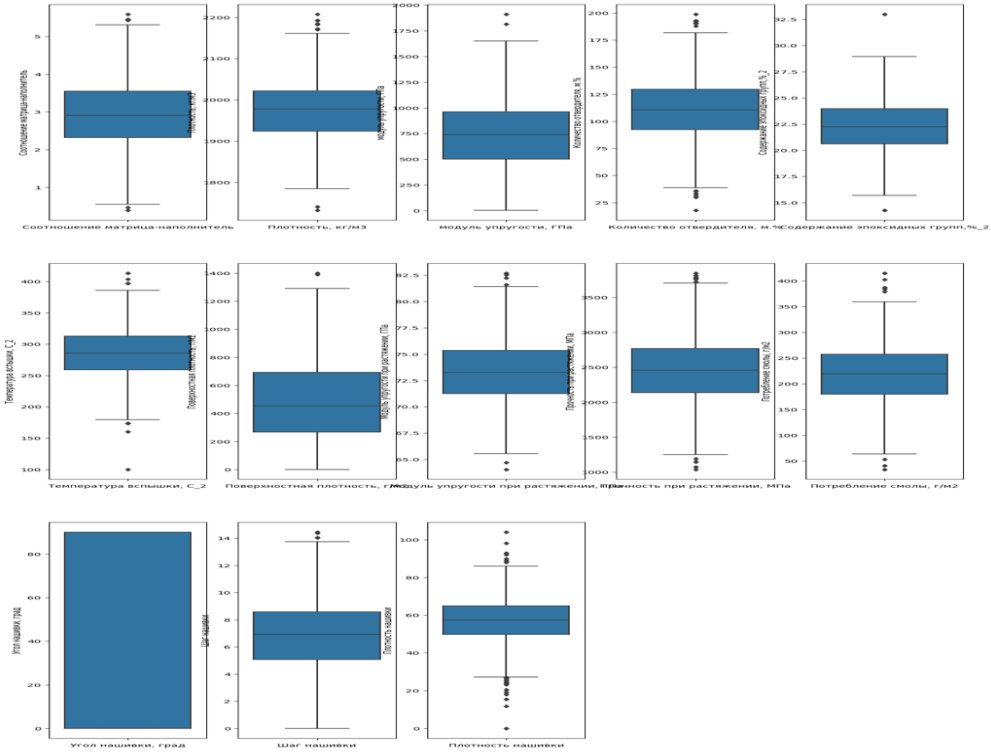


Рисунок 10 График ящиков с усами.

После анализа данных были обнаружены выбросы которые будем удалять методом межквартильных растояний IQR. IQR испоьзуется компаниями в качестве показателя уровня их дохода, также он используется для выявления выбросов и аномалий. Также может указывать на асиметрию набора данных.

На этом наш анализ заканчивается переходим к предобработке данных.

2 Практическая часть

2.1 Предобработка данных

Если наши данные будут не в лучшем формате то мы не сможем оценить как наши признаки влияют на целевую переменную необходимо подготовить данные таким образом чтобы они давали нашим алгоритмам обучения наилучшую возможность для решения задач наша модель при обучении должна иметь возможность работать с одинаковыми признаками и иметь один диапазон. Если данные в одном порядке то необходимости к нормализации нет. Нормализация это такой процесс когда мы делаем преобразование числовых значений как правило вдиапазоне от ноля до единицы, при этом мы основываемся на максимальном и минимальном значении. Мы будем применять межквартильный интервал он определяется как разница между 75м и 25м процентилями данных.

Удаляем обнаруженные выбросы методом межквартильных интервалов.

Рисунок 11.

```
# Чистим через цикл
for i in dropen:
    qvan_75, qvan_25 = np.percentile(df.loc[:,i], [75,25])
    intr_qr = qvan_75 - qvan_25
    max = qvan_75 + (1.5 * intr_qr) # выбрана размашка 1,5
    min = qvan_25 - (1.5 * intr_qr)
    df.loc[df[i] < min, i] = np.nan
    df.loc[df[i] > max, i] = np.nan

df_n = df.dropna(axis=0)
df_n.shape
```

Смотрим на график ящик с усами после удаления выбросов и тут же видим асиметрию данных Рисунок 12.

```
plt.show()
```

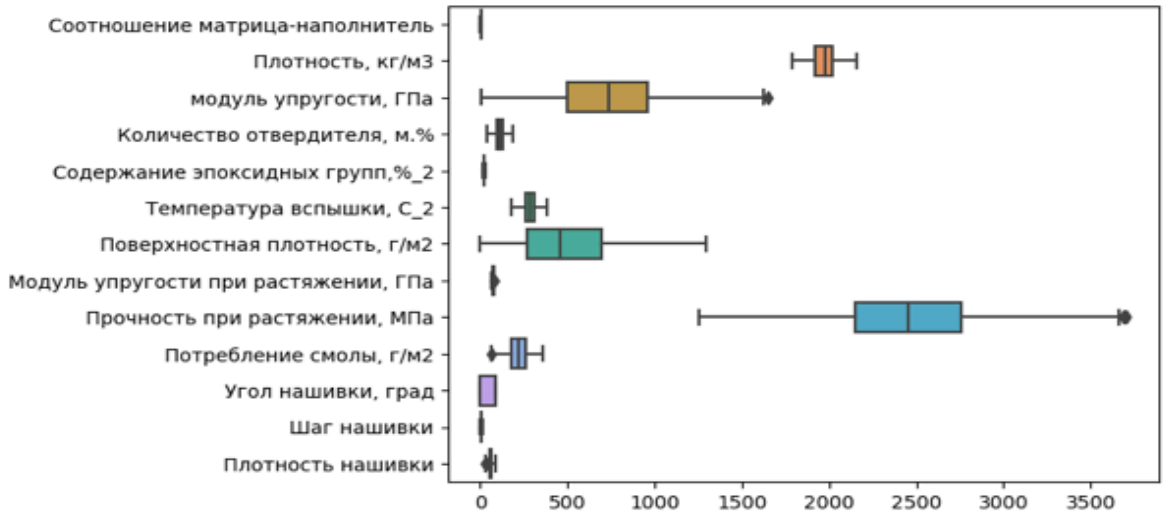


Рисунок 12.

Выбросы практически ушли дальше очищать не будем так можем утерять часть данных которые являются редкими/уникальными признаками, явные выбросы которые кординально выбиваются из общего увиденного в выборке мы убрали которые могли быть ошибкой ввода данных (аномалией). Сделаем оценку плотности ядра рисунок 13.

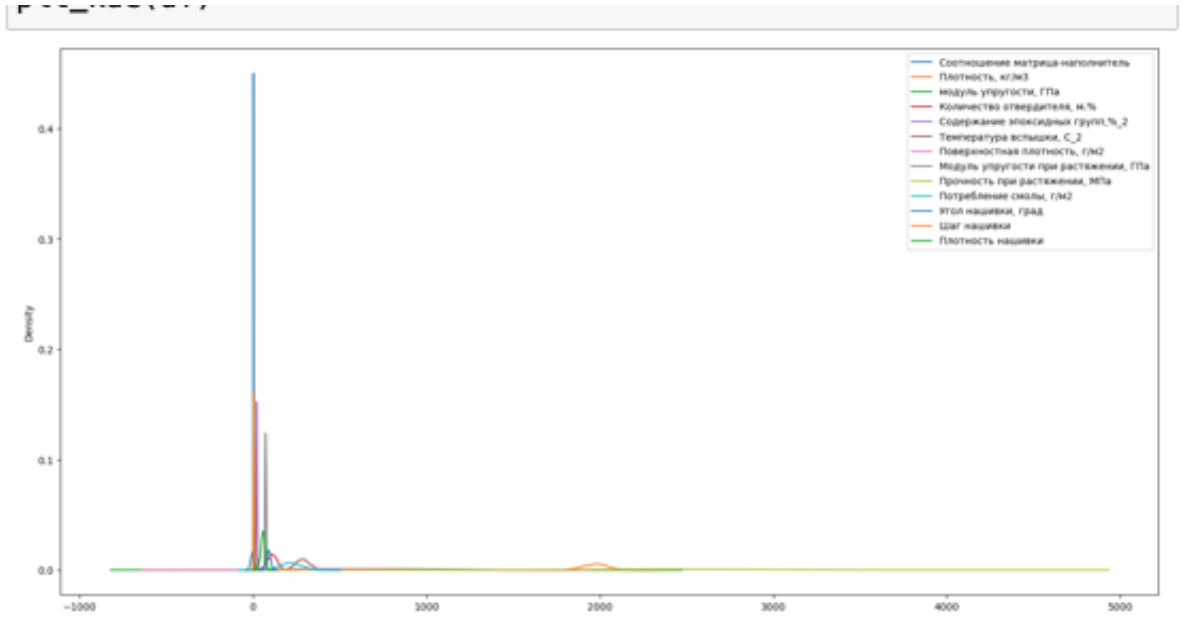


Рисунок 13.

Нормализуем данные пррименяем MinMaxScaler. так как многие модели машинного обучения показывают лучшие результаты, когда значения признаков лежат в одинаковых диапазонах, а в исходных данных это не так, то используем MinMaxScaler (значения по умолчанию min=0, max=1)

, которая преобразует значения числовых признаков таким образом, что они будут лежать в диапазоне от 0 до 1.

```
#Нормализация MinMaxScaler()
scaler = preprocessing.MinMaxScaler()
col = data_norm.columns
result = scaler.fit_transform(data_norm)
minmax = pd.DataFrame(result, columns = col)
```

Описательная статистика характеристик после нормализации на рисунке 14.

	count	mean	std	min	25%	50%	75%	max
Unnamed: 0	936.0	0.51	0.29	0.0	0.25	0.52	0.76	1.0
Соотношение матрица-наполнитель	936.0	0.50	0.19	0.0	0.37	0.49	0.63	1.0
Плотность, кг/м3	936.0	0.50	0.19	0.0	0.37	0.51	0.62	1.0
модуль упругости, ГПа	936.0	0.45	0.20	0.0	0.30	0.45	0.58	1.0
Количество отвердителя, м.%	936.0	0.50	0.19	0.0	0.38	0.51	0.64	1.0
Содержание эпоксидных групп,%_2	936.0	0.49	0.18	0.0	0.37	0.49	0.62	1.0
Температура вспышки, С_2	936.0	0.52	0.19	0.0	0.39	0.52	0.65	1.0
Поверхностная плотность, г/м2	936.0	0.37	0.22	0.0	0.21	0.35	0.54	1.0
Модуль упругости при растяжении, ГПа	936.0	0.49	0.19	0.0	0.36	0.49	0.62	1.0
Прочность при растяжении, МПа	936.0	0.50	0.19	0.0	0.37	0.49	0.61	1.0
Потребление смолы, г/м2	936.0	0.52	0.20	0.0	0.39	0.52	0.65	1.0
Угол нашивки, град	936.0	0.51	0.50	0.0	0.00	1.00	1.00	1.0
Шаг нашивки	936.0	0.50	0.18	0.0	0.37	0.50	0.62	1.0
Плотность нашивки	936.0	0.51	0.19	0.0	0.39	0.52	0.64	1.0

Рисунок 14.

Нормализации — приведение различных данных в самых разных единицах измерения и диапазонах значений к единому виду, сделанно

Оценим ящик с усами на рисунке 15. Также видим диапазон от 0-1.

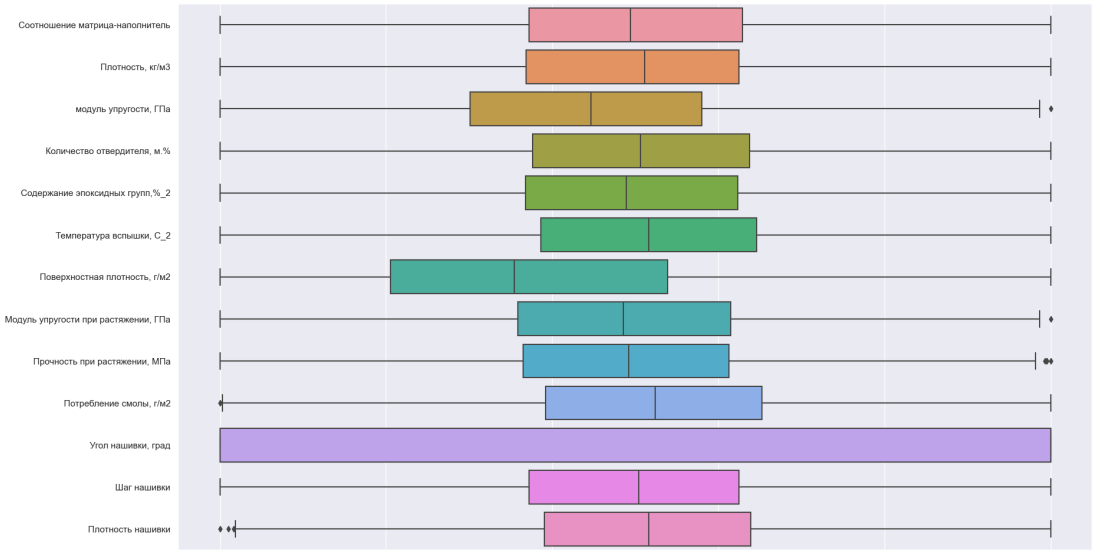


Рисунок 15.

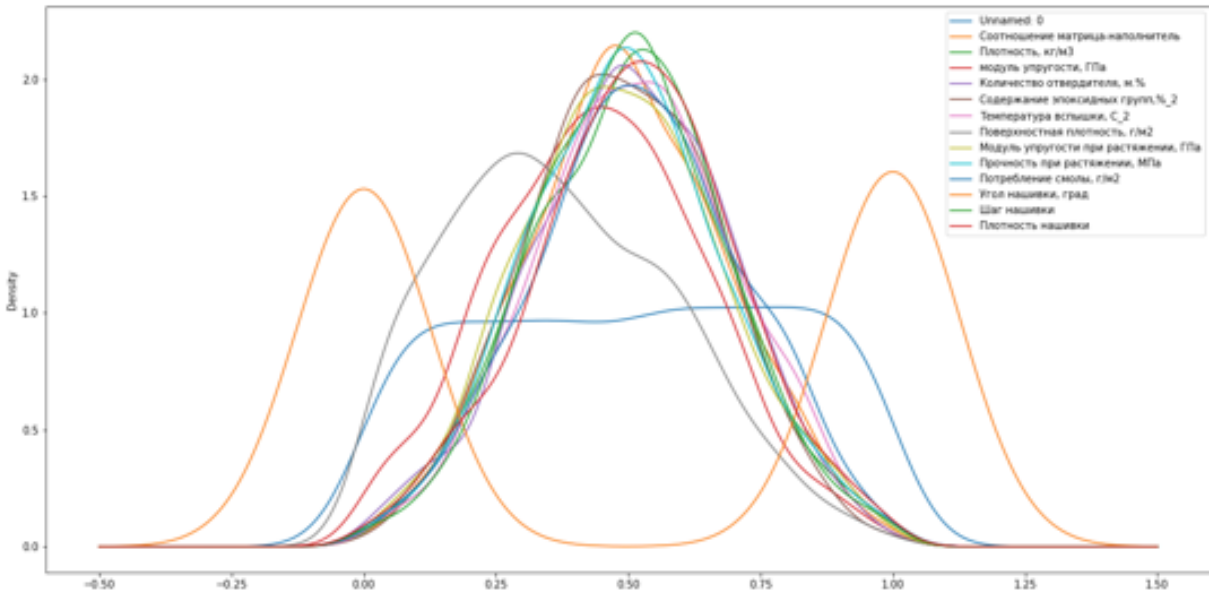


Рисунок 16 Визуализируем данные плотности ядра после нормализации в сравнении с рисунком 13. Видим явные отличия, данный график говорит о том, что все получилось довольно не плохо.

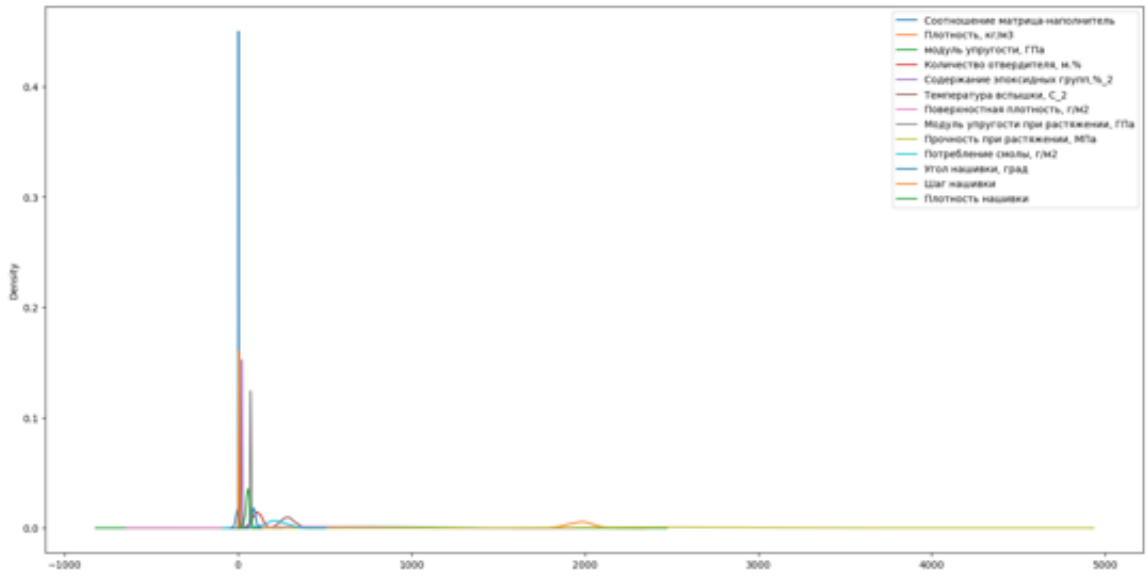


Рисунок 13. До нормализации.

2.2Разработка и обучение модели

В процессе обучения моделей проведена оценка точности этих моделей на выборках тестовой и обучающей. Разработка и обучение моделей машинного обучения согласно методами выбранными в пункте 1.2. Строим модели для прогноза модуля упругости при растяжении и прочности при растяжении. Загружаем данные из файла сохранения

```
df_n = pd.read_excel('./data/X_cln.xlsx')
df_n.shape
```

Определяем входы и выходы для модели, разделяем датасет на тестовую и обучающую выборки рисунок 14.

```
# Разбиваем на тестовую, тренировочную выборки (Прочность при растяж
X_train_one, X_test_one, y_train_one, y_test_one = train_test_split(
    normal.loc[:, normal.columns != 'Прочность при растяжении, МПа']
    data_norm[['Прочность при растяжении, МПа']],
       test_size = 0.3,
    random_state = 42)
```

Рисунок 14.

Далее применяем следующие модели: линейная регрессия, метод опорных векторов, k-ближайших соседей, случайный лес, градиентный бустинг.

2.3Тестирование модели

Выводить оценки будем по

R2 коффициент детерминации, MAE средняя абсолютная ошибка и MSE

Линейная регрессия

```
#Метод линейной регрессии
lr = LinearRegression()
lr.fit(X_train_one, y_train_one)
y_pred_lr = lr.predict(X_test_one)
mae_lr = mean_absolute_error(y_pred_lr, y_test_one)
R2_lr = r2_score(y_pred_lr, y_test_one)
```

```
Результат:
Score: 0.97
lr_MAE: 64.60
lr_MAPE: 0.03
Test score: 0.97
```

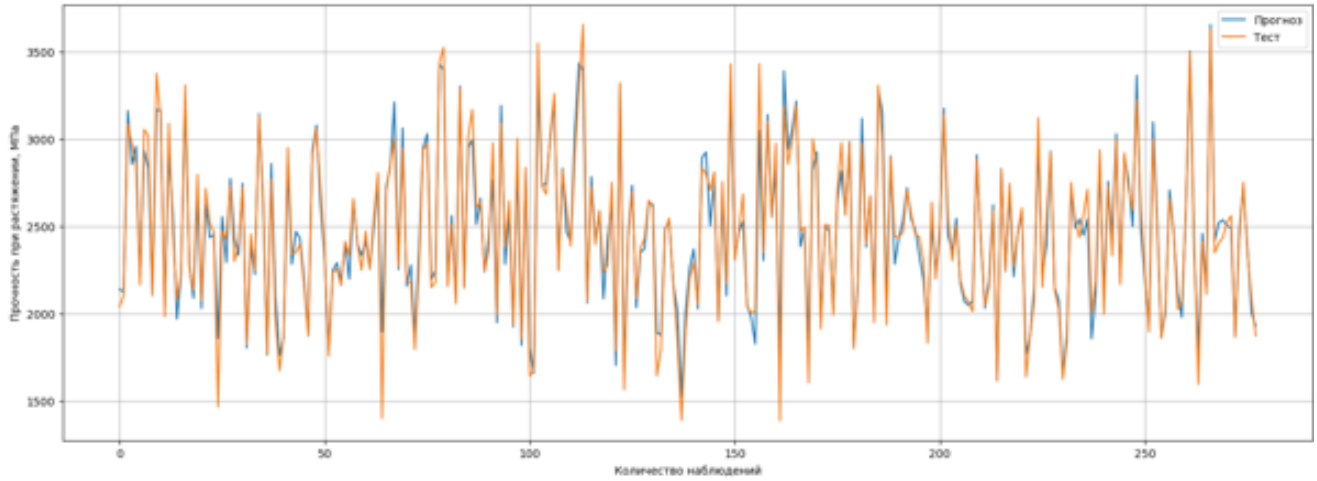


График на рисунке 15 На удивление справился по R2 на 96 процентов

model	target	MSE	R2
Метод линейной регрессии	Прочность при растяжении, МПа	6940.273546	0.967326

k-ближайших соседей

```
# Метод K ближайших соседей
knr = KNeighborsRegressor(n_neighbors=5)
knr.fit(X_train_one, y_train_one)
y_pred_knn = knr.predict(X_test_one)
mae_knr = mean_absolute_error(y_pred_knn, y_test_one)
R2_knr = r2_score(y_pred_knn, y_test_one)
```

Результат работы модели. Рисунок 16.

```
K Neighbors Regressor  Results Train:
Score: 0.95
K Neighbors Regressor  Results:
KNN_MAE:  106
KNN_MAPE: 0.05
Test score: 0.92
```

Рисунок16

model	target	MSE	R2
Метод к ближайших соседей	Прочность при растяжении, МПа	17722.988965	0.916561

Метод опорных векторов

```
svr = make_pipeline(StandardScaler(), SVR(kernel = 'rl
#обучаем модель
svr.fit(X_train_one, np.ravel(y_train_one))
#вычисляем коэффициент детерминации
y_pred_svr=svr.predict(X_test_one)
mae_svr = mean_absolute_error(y_pred_svr, y_test_one)
R2_svr = r2_score(y_pred_svr, y_test_one)
```

Результат работы метода на рисунке 17.

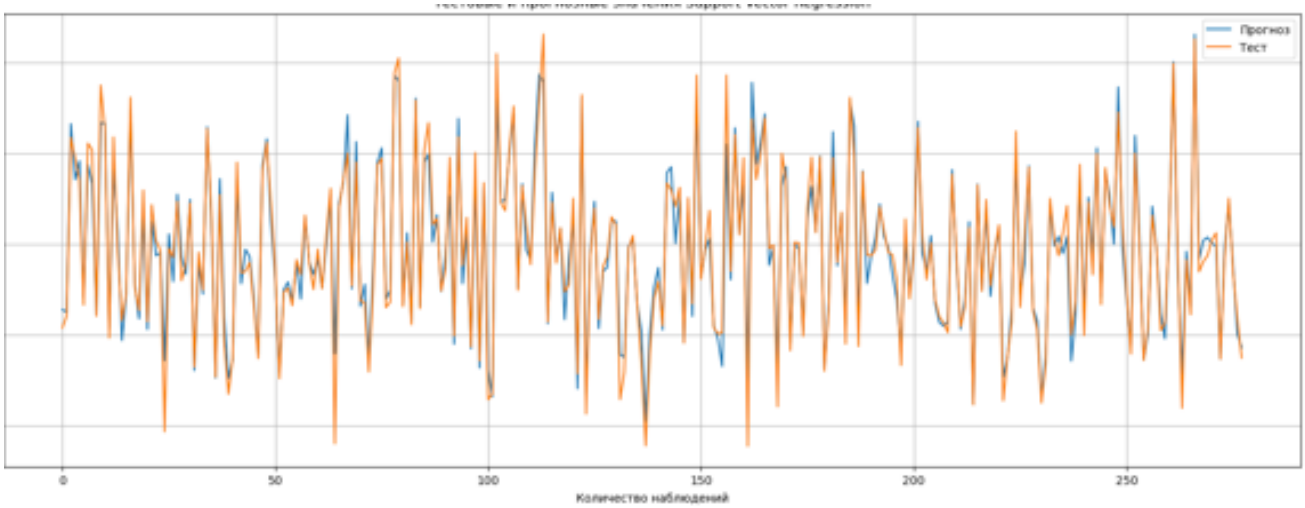


График на рисунке 17 Модуль упругости и Прочность при растяжении

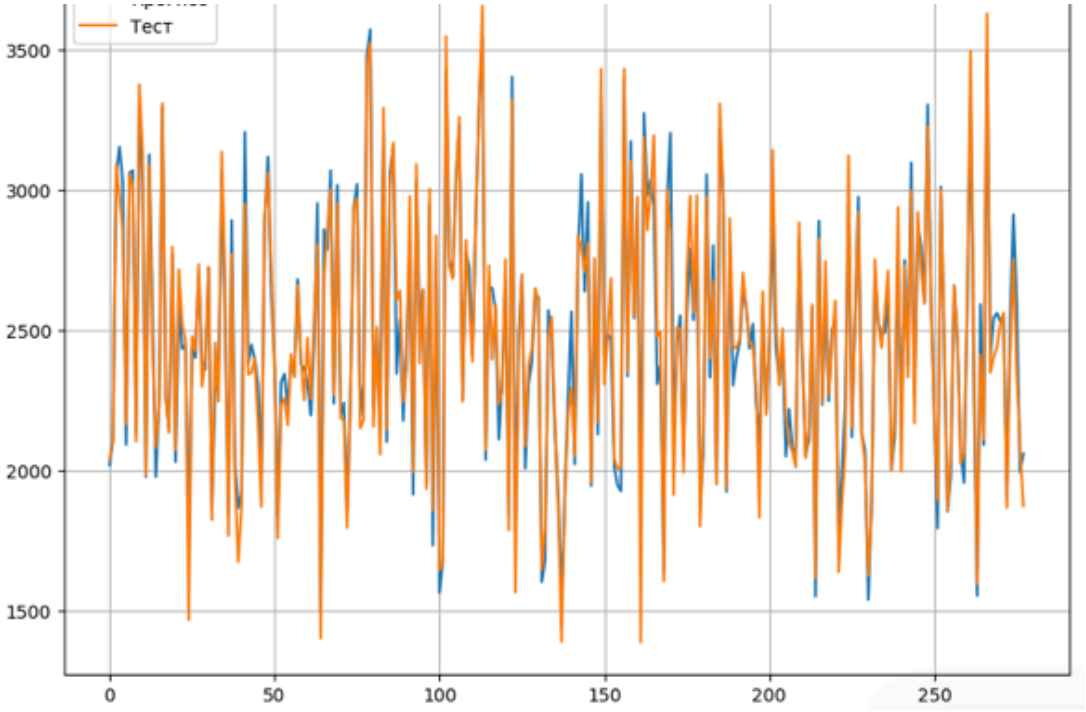
model	target	MSE	R2
Метод опорных векторов	Прочность при растяжении, МПа	9551.443529	0.955032

```
Результат:
Score: 0.99
SVR_MAE: 69.61
SVR_MAPE: 0.03
Test score: 0.96
MAE for mean target:  371.
```

Случайный лес

```
rfr = RandomForestRegressor(n_estimators=15,max_depth=7,
rfr.fit(X_train_one, y_train_one.values)
y_pred_forest = rfr.predict(X_test_one)
mae_rfr = mean_absolute_error(y_pred_forest, y_test_one)
R2_rfr = r2_score(y_pred_forest, y_test_one)
```

model	target	MSE	R2
Метод случайного леса	Прочность при растяжении, МПа	10217.704986	0.951896



Результат работы представлен на рисунке 18.

Градиентный бустинг

```
#Метода градиентного бустинга
gbr = make_pipeline(StandardScaler(), GradientBoostingRegressor())
gbr.fit(X_train_one, np.ravel(y_train_one))
y_pred_gbr = gbr.predict(X_test_one)
mae_gbr = mean_absolute_error(y_pred_gbr, y_test_one)
R2_gbr = r2_score(y_pred_gbr, y_test_one)
```

Результаты

```
Результат:
Score: 0.99
GBR_MAE:  62
GBR_MAPE: 0.03
Test score: 0.97
```

	model	target	MSE	R2
0	Метод опорных векторов	Прочность при растяжении, МПа	9551.443529	0.955032
1	Метод случайного леса	Прочность при растяжении, МПа	10217.704986	0.951896
2	Метод линейной регрессии	Прочность при растяжении, МПа	6940.273546	0.967326
3	Метод градиентного бустинга	Прочность при растяжении, МПа	6212.086851	0.970754

Оценки собраны в датасет ошибок и приведена ниже на рисунке 19.

Модели дали желаемый результат ниже 90 процентов по R2 не падала учитывая что даже 0, 5 не всегда плохо.

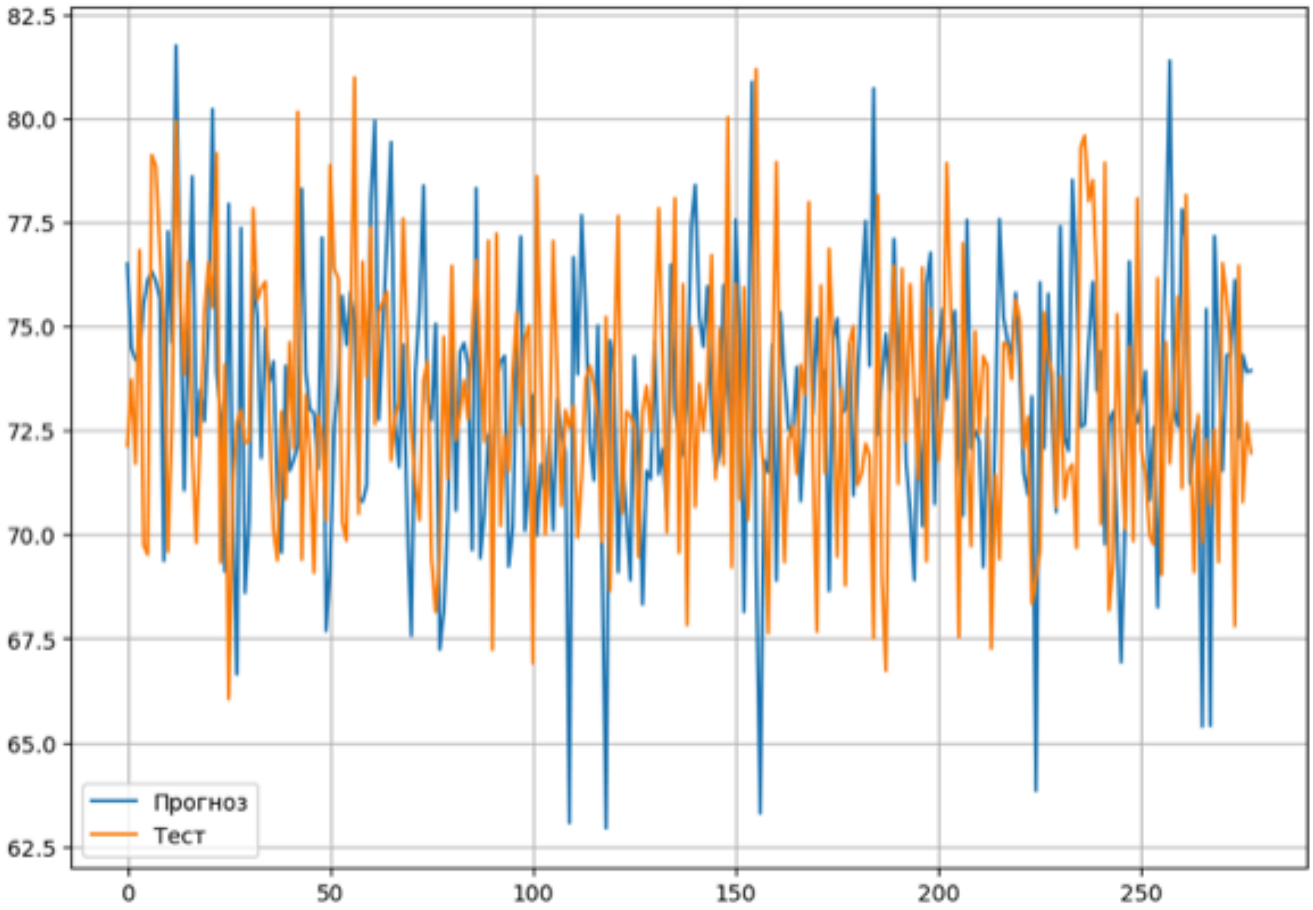
А теперь посмотрим на модели для модуля упругости при растяжении, первоначально модели обучались не соло а вместе, что не дало по какимто причинам хороших результатов поэтому было решение обучить этот модуль с отдельно, разбиваем выборку. Используем подбор гипер параметров.

Метод опорных векторов

```
#Метод опорных векторов
svr2 = make_pipeline(StandardScaler(), SVR(kernel = 'rbf', C = 500.0,

#обучаем модель
svr2.fit(X_train_two, np.ravel(y_train_two))
```

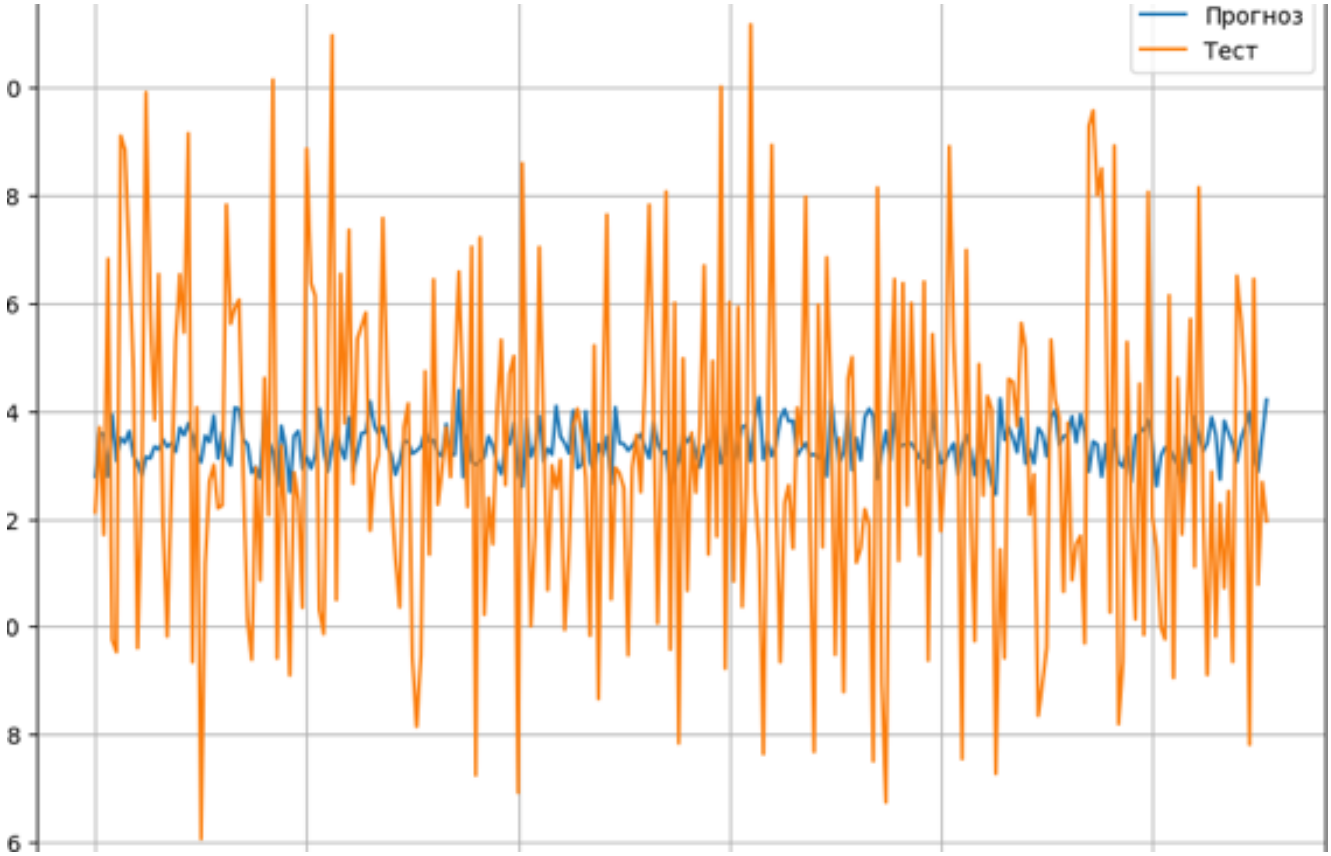
```
SVR_MAE: 3.44617
SVR_MAPE: 0.04724
SVR_MAPE: 0.04724
Train core: 0.90
Test score: -1.03
MAE for mean target:  2.4458905302866936
```



Метод линейной регрессии обучение и результаты с графиком

```
lr2 = LinearRegression()
lr2.fit(X_train_two, y_train_two)
y_pred_lr2 = lr2.predict(X_test_two)
```

Результаты.



Метод случайного леса подбираем гипер параметры для метода.

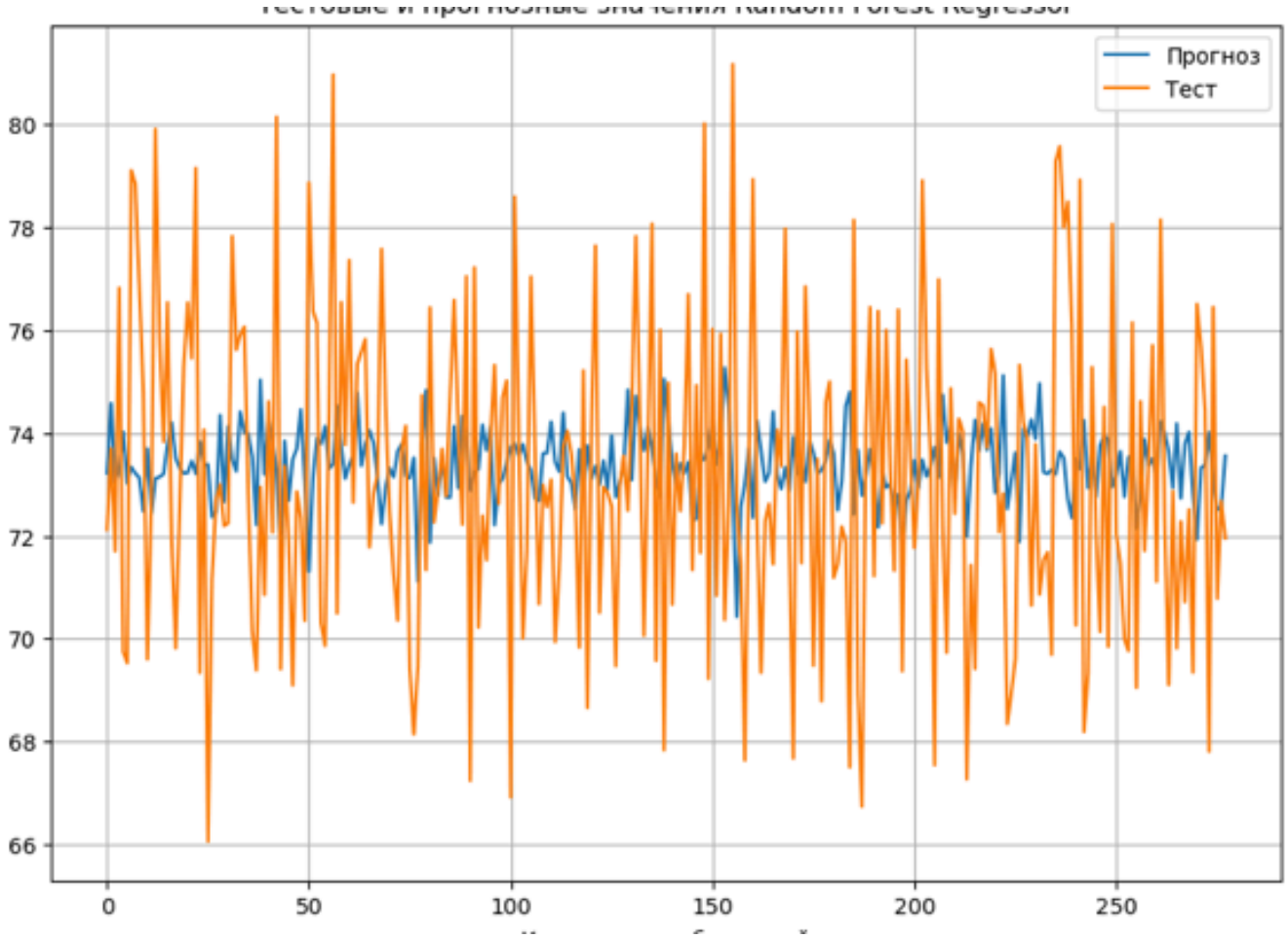
```
RandomForestRegressor(criterion='mse', max_depth=15, n_estimators=300,
                      random_state=33)
R2-score RFR для прочности при растяжении, МПа: 0.962
```

Обучаем метод и смотрим результат.

```
rfr2 = RandomForestRegressor(n_estimators = 15,max_depth = 7, random_
rfr2.fit(X_train_two, y_train_two.values)
y_pred_forest2 = rfr2.predict(X_test_two)
```

```
Результат:
Score: 0.44
RF_MAE: 2.5469516458
RF_MAPE: 0.0348912694
Test score: -0.0919659356
```

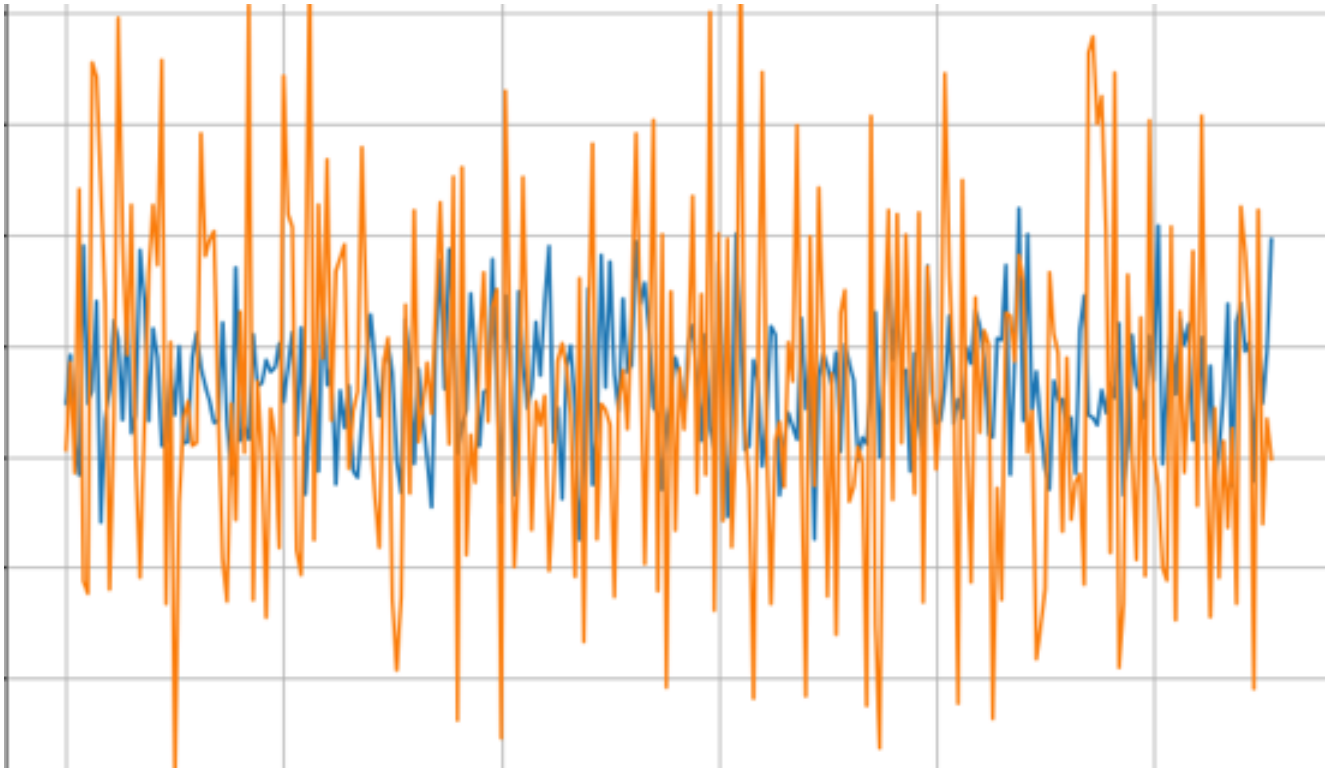
Результаты представлены на графике ниже



Метод К ближайших соседей результаты на рисунке ниже.

```
# Метод К ближайших соседей
knr2 = KNeighborsRegressor(n_neighbors=5)
knr2.fit(X_train_two, y_train_two)
y_pred_knn2 = knr2.predict(X_test_two)
```

График обучения результат.



По итогу обучения было сделано сравнение всех методов с сортировкой по результативности с оценками.

	Регрессор	MAE	R2
3	Градиентный бустинг	62.409764	0.968839
2	Линейная регрессия	64.595424	0.964526
6	Случайный лес_Grid	67.714998	0.961933
0	Опорные вектора	69.613765	0.949222
1	Случайный лес	75.724864	0.948571
5	K ближайшие соседи_Grid	103.377777	0.921272
4	K ближайшие соседи	106.144166	0.892177

2.4Написать нейронную сеть, которая будет рекомендовать соотношение матрица.

В результате проведенной работы была создана нейро сеть при помощи библиотеки TensorFlow. Пять скрытых слоев, конфигурации: x_train_n на входе первый слой, далее скрытые слой 128, далее 64 и последний скрытый 16, 3, 3, на выходе 1 нейрон. Активационная функция сконфигурирована данным образом, см на рисунке 20.

```
model_NN = Sequential([
    x_train_n,
    Dense(128, activation='relu'),
    Dense(64, activation='relu'),
    Dense(16, activation='relu'),
    Dense(3, activation='relu'),
    Dense(3, activation='relu'),
    Dense(1)
    ])
model_NN.compile(optimizer = tf.ke
loss = 'mean_squared_error', metri
```

Рисунок 20.

Перед входом в нейронку данные были нормализованы. И разбиты на выборки. Подача данных в сеть. И результат на рисунке 21.

```
model_NN.summary()
# Обучим модель

model_hist1 = model_NN.fit(
    x_train,
    y_train,
    epochs = 100,
    verbose = 1,
    validation_split = 0.2)
```

```
Model Results:
Model_MAE:  1
Model_MAPE: 0.39
Test score: 1.36
```

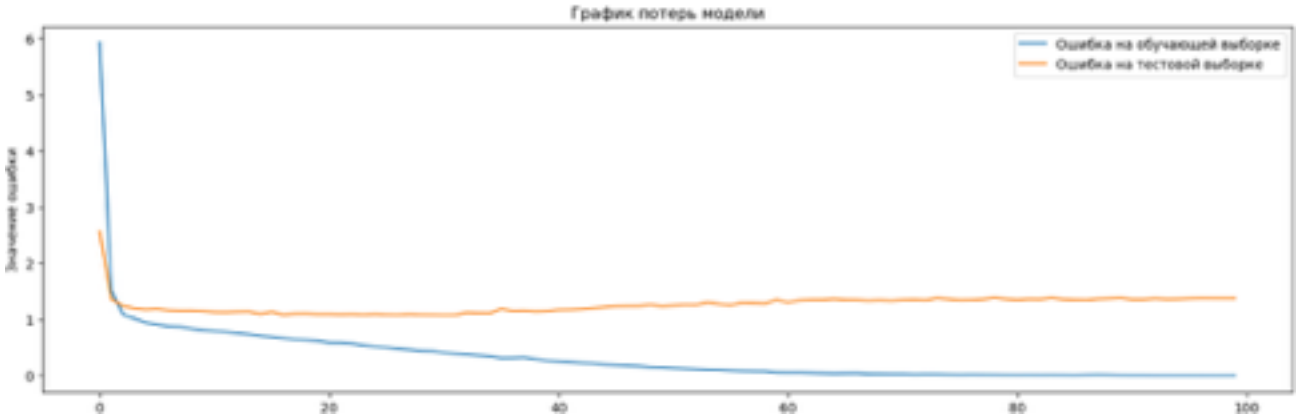
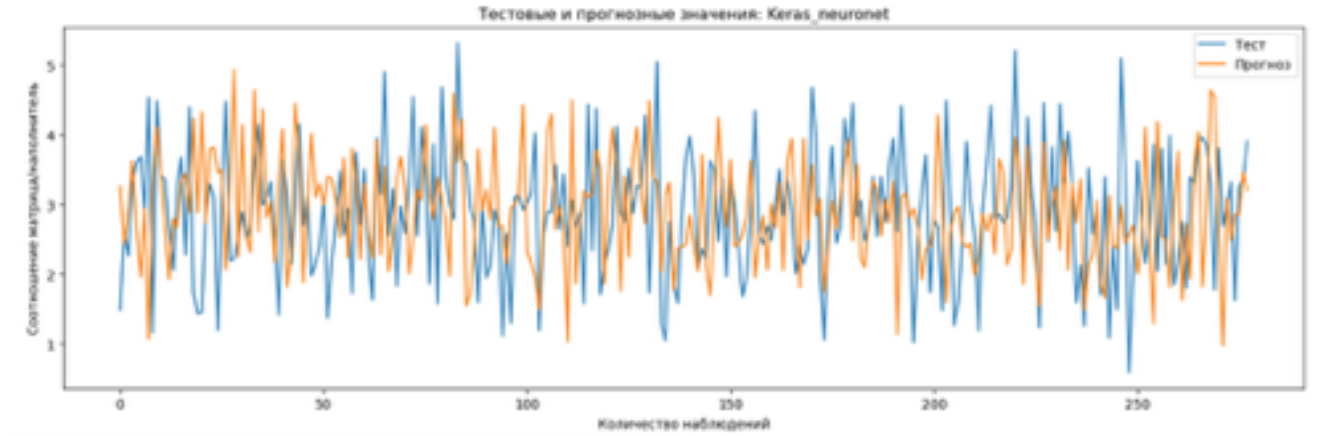


Рисунок 21. Графики отклогнений показывают рост ошибок сеть не эфективная прогнозы будут не точные. Посмотрим оценку MSE




```
# оценка модели MSE
model_NN.evaluate(x_test, y_test, verbose = 1)
```

```
9/9 [==============================] - 0s 2ms/step - loss:
1.3590 - root_mean_squared_error: 1.1658

[1.358999490737915, 1.1657613515853882]
```

Было использовано разное количество настроек но все же не достаточно чтобы рекомендовать точные прогнозы. При обучении моделей было замечено что на тестовой выборке нейронная сеть ведет себя не состоятельно.

2.5Разработка приложения

Приложение находится в конце основного ноутбука имеет консольное управление ввода рандомного числа для просмотра расчета модуля упругости при растяжении и прочность при растяжении предсказанную машинным обучением.

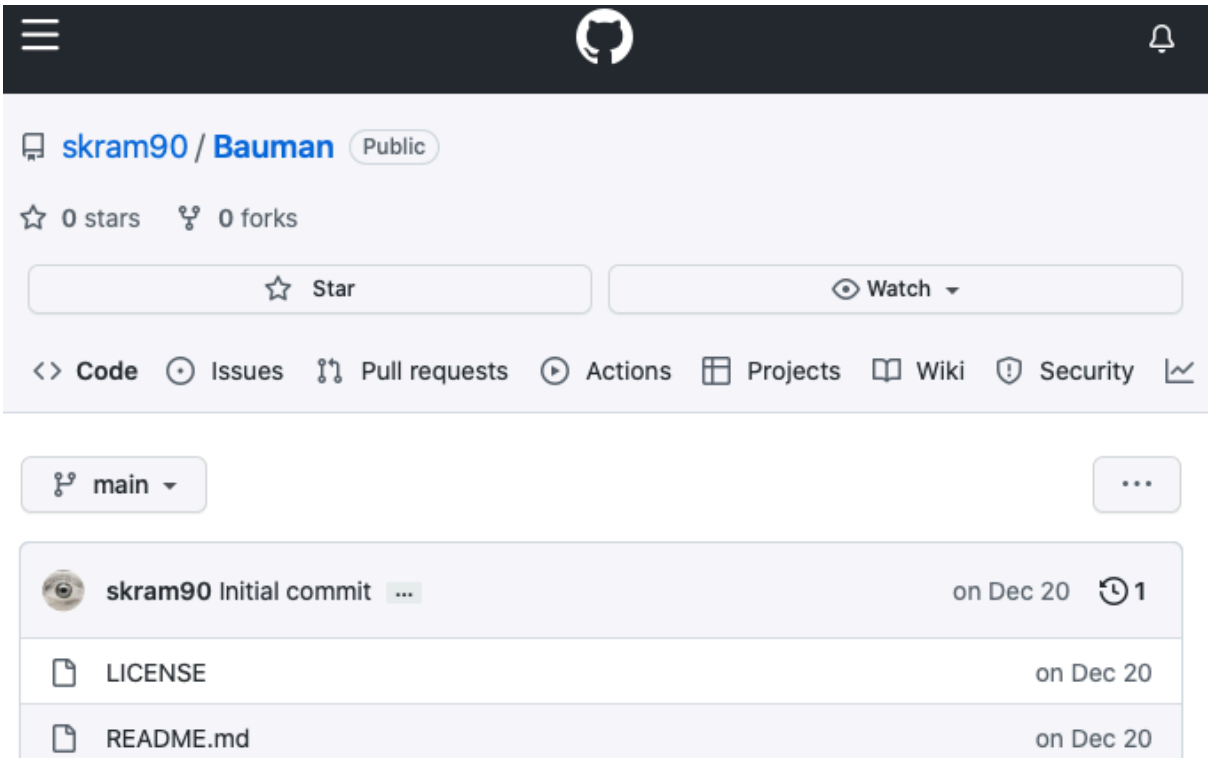
```
print("Ноль завершит работу программы 1-start, 0-exit")

for i in range(len(df_n)):
    df.loc[i, 'wq'] = i*5
while True:
    s = input("1-start, 0-exit")
    if s == '0':
        break
    if s !=0:
        n = input("1-start, 0-exit")
        x = np.expand_dims(X_train_one.iloc[n], axis=0)
        res = svr.predict(x)
        print(res)
        y_train_one.iloc[n]
```

```
Ноль завершит работу программы
```

2.6. Создание удаленного репозитория

Репозиторий был создан на github.com по адресу: https://github.com/skram90/Bauman



2.7. Заключение

Машинное обучение в задачах моделей прогнозирования – сложный процесс, требующий не только навыков программирования, но и профессиональных знаний в сфере работы с композитными материалами, в такой рааботе необходим контакт со специалистами в данной сфере для консультаций.

В ходе работы была произведена подробная опись и анализ датасета; построено множество графиков; осуществлено разбиение данных на обучающую и тестовую выборки. Для реализации моделей машинного обучения и поиска гиперпараметров были задействованы несколько алгоритмов: метод К ближайших соседей, линейная регрессия, деревья решений, опорные вектора, случайный лес. Были составлены отчеты, оценивающие качество проводимого обучения.

Было представлено сравнение результатов оценок работы моделей, графики и диаграммы, позволяющие оценить итоги проведенного обучения. Обучена и разработана нейронная сеть. В процессе выполнения данной работы получилось сделать следующие выводы. Распределение полученных данных в объединенном датасете близко к нормальному, коэффициенты корреляции между парами признаков стремятся к нулю, использованные модели не позволяют получить достаточно достоверные прогнозы, распределение полученных данных в объединенном датасете близко к нормальному коэффициенты корреляции между парами признаков стремятся к нулю использованные модели не всегда позволяют получить достоверные прогнозы лучшая метрика для прочности при растяжении, метод градиентного бустинга лучшая метрика для модуля упругости при растяжении – метод опорных векторов.

Из этого можно сделать вывод, для того чтобы определить на сколько верны полученные решения нужно дополнять датасет недостоющими данными, а для этого нужно основательно погрузится в проблематику данного направления, а именно собрать команду из специалистов по данной теме для консультаций и сбора информации.

2.9.Библиографический список

1. Devpractice Team. Python. Визуализация данных. Matplotlib. Seaborn. Mayavi. - devpractice.ru. 2020. - 412 с.: ил.
2. Гафаров, Ф.М., Галимянов А.Ф. Искусственные нейронные сети и приложения: учеб. пособие /Ф.М. Гафаров, А.Ф. Галимянов. – Казань: Издательство Казанского университета, 2018. – 121 с.
3. Джулли, Пал: Библиотека Keras - инструмент глубокого обучения / пер. с англ. А. А. Слинкин.- ДМК Пресс, 2017. – 249 с.
4. Грас, Джоэл. Data Science. Наука о данных с нуля: Пер. с англ. - 2-е изд., перераб. и доп. - СПб.: БХВ-Петербург, 2021. - 416 с.:

5. Д. Рутковская, М. Пилиньский, Л. Рутковский Нейронные сети, генетические алгоритмы и нечеткие системы. - М.: Горячая Линия - Телеком. - 2013. - 384 с. ISBN: 978-5-9912-0320-3
6. Д. Фостер Генеративное глубокое обучение. Творческий потенциал нейронных сетей. - СПб.: Питер. - 2020. - 336 с. - ISBN: 978-5-4461-1566-2
7. Андерсон, Карл Аналитическая культура. От сбора данных до бизнес-результатов / Карл Андерсон ; пер. с англ. Юлии Константиновой ; [науч. ред. Руслан Салахиев]. — М. : Манн, Иванов и Фербер, 2017. — 336 с
8. С. Николенко, А. Кадурин, Е. Архангельская Глубокое обучение. Погружение в мир нейронных сетей. - СПб.: Питер. - 2020. - 480 с.
9. Д. Фостер Генеративное глубокое обучение. Творческий потенциал нейронных сетей. - СПб.: Питер. - 2020. - 336 с. - ISBN: 978-5-4461-1566-2