

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

ПО КУРСУ “Data Science”

Тема: Прогнозирование конечных свойств новых композиционных материалов

Цель исследования: построение моделей прогнозирования следующих параметров: «модуль упругости при растяжении»; «прочность при растяжении»; «соотношение матрица-наполнитель»

Итог работы: Разработка приложения, которое будет выдавать прогноз параметра «соотношение матрица-наполнитель».

Слушатель: Савельев Петр Олегович

Введение

На входе имеются данные о начальных свойствах компонентов композиционных материалов (количество связующего наполнителя, температурный режим отверждения и т.д.).

Данные представлены в виде двух таблиц X_bp.xlsx и X_pip.xlsx

На выходе необходимо спрогнозировать ряд конечных свойств получаемых композиционных материалов.

Созданные прогнозные модели помогут сократить количество проводимых испытаний, а также пополнить базу данных материалов возможными новыми характеристиками материалов, и цифровыми двойниками новых композитов.

План работ

- 1.Импорт библиотек, загрузка датасетов, объединение таблиц
- 2.Проведение разведочного анализа
- 3.Визуализация
- 4.Предобработка данных удаление выбросов
- 5.Разработка моделей прогноза прочности при растяжении модуля упругости при растяжении построение моделей, поиск лучшей модели
- 6.Разработка нейронной сети для рекомендации соотношения матрица-наполнитель
7. Заключение

Загрузка и обработка данных

1. Импортирование библиотек
2. Загрузка файлов
3. Проверка размерностей
4. Объединение по индексу
5. Удаление лишних колонок

```
# загружаем файлы
```

```
df1 = pd.read_excel('./Data/X_bp.xlsx')
df2 = pd.read_excel('./Data/X_nup.xlsx')
# смотрим размерность
df1.shape, df2.shape
```

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

```
# объединяем датасеты
```

```
df = pd.merge(df1, df2, how = 'inner')
df.drop(['Unnamed: 0'], axis=1, inplace=True)
# 13 столбцов и 1023 строки
df.shape
```

```
(1023, 13)
```

```
# устранием строки без индекса
df.dropna(inplace = True)
df.head()
```

Соотношение матрица-наполнитель	Плотность, кг/м3	модуль упругости, ГПа	Количество отвердителя, м.%	Содержание эпоксидных групп,%_2
---------------------------------	------------------	-----------------------	-----------------------------	---------------------------------

0	1.857143	2030.0	738.736842	30.00	22
1	1.857143	2030.0	738.736842	50.00	23
2	1.857143	2030.0	738.736842	49.90	33

Соотношение матрица-наполнитель	Плотность, кг/м3	модуль упругости, ГПа	Количество отвердителя, м.%	Содержание эпоксидных групп,%_2	Температура вспышки, С_2	Поверхностная плотность, г/м2
0	1.857143	2030.0	738.736842	30.00	22.267857	100.000000
1	1.857143	2030.0	738.736842	50.00	23.750000	284.615385
2	1.857143	2030.0	738.736842	49.90	33.000000	284.615385
3	1.857143	2030.0	738.736842	129.00	21.250000	300.000000
4	2.771331	2030.0	753.000000	111.86	22.267857	284.615385

```
df.shape, df.columns
```

```
((1023, 13),
Index(['Соотношение матрица-наполнитель', 'Плотность, кг/м3',
       'модуль упругости, ГПа', 'Количество отвердителя, м.%',
       'Содержание эпоксидных групп,%_2', 'Температура вспышки, С_2',
       'Поверхностная плотность, г/м2', 'Модуль упругости при растяжении, ГПа',
       'Прочность при растяжении, МПа', 'Потребление смолы, г/м2',
       'Угол нашивки, град', 'Шаг нашивки', 'Плотность нашивки'],
      dtype='object'))
```

```
print(df.shape)
df.head(1)
```

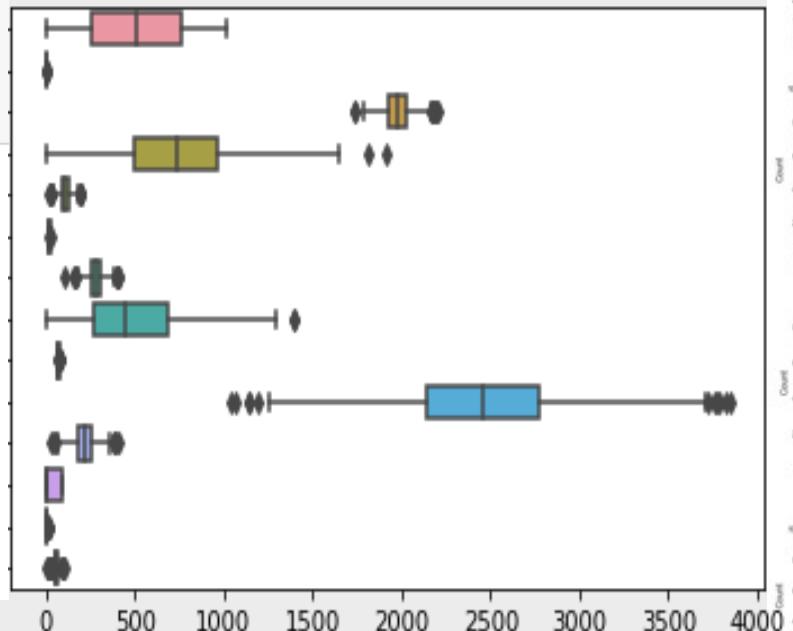
```
(1023, 13)
```

Проводим разведочный анализ

```
# Делаем подсчет уникальных значений по столбцам
```

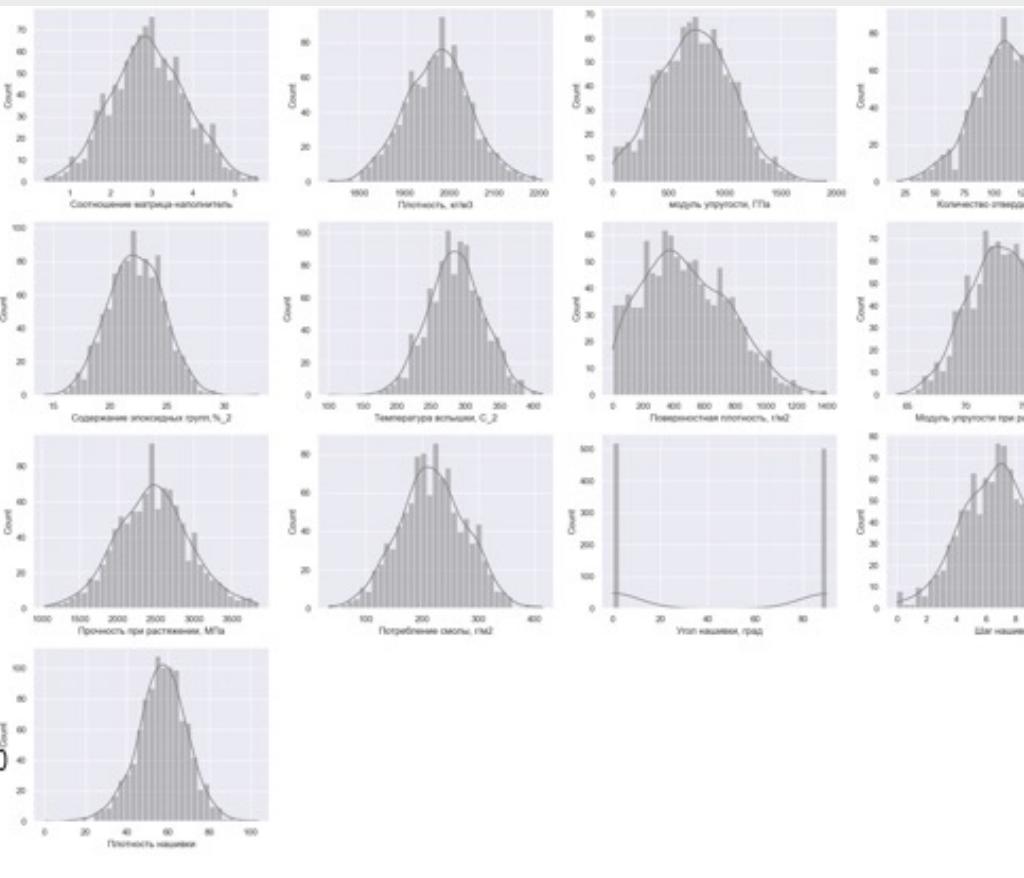
```
df.unique()
```

Соотношение матрица–наполнитель	1014
Плотность, кг/м ³	1013
модуль упругости, ГПа	1020
Количество отвердителя, м.%	1005
Содержание эпоксидных групп,%_2	1004
Температура вспышки, С_2	1003
Поверхностная плотность, г/м ²	1004
Модуль упругости при растяжении, ГПа	1004
Прочность при растяжении, МПа	1004
Потребление смолы, г/м ²	1003
Угол нашивки, град	2
Шаг нашивки	989
Плотность нашивки	988
dtype: int64	



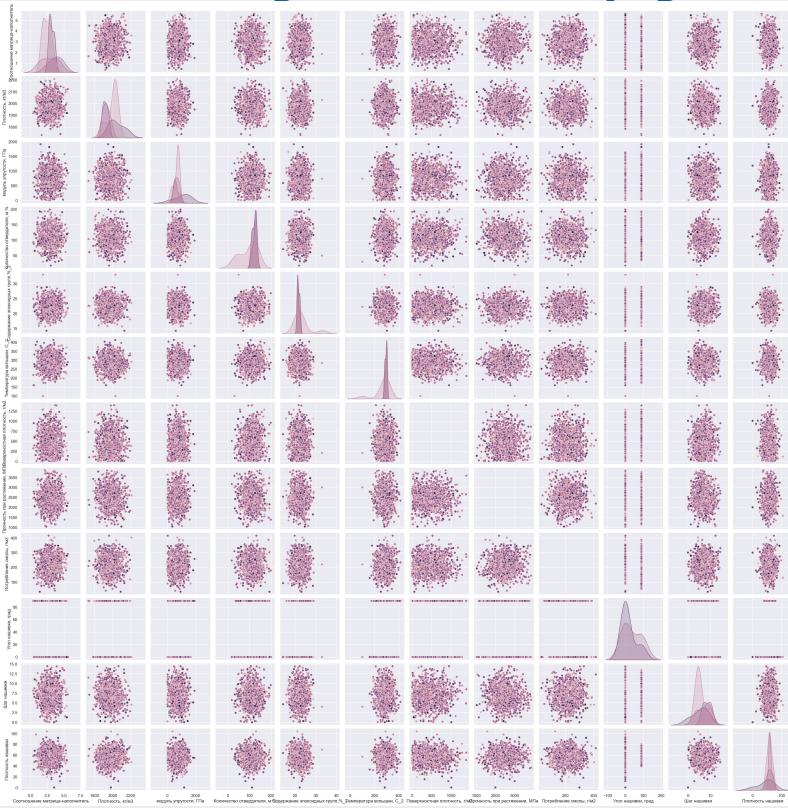
```
df.isna().sum() # подсчет пропущенных значений
```

Соотношение матрица–наполнитель	0
Плотность, кг/м ³	0
модуль упругости, ГПа	0
Количество отвердителя, м.%	0
Содержание эпоксидных групп,%_2	0
Температура вспышки, С_2	0
Поверхностная плотность, г/м ²	0
Модуль упругости при растяжении, ГПа	0
Прочность при растяжении, МПа	0
Потребление смолы, г/м ²	0
Угол нашивки, град	0
Шаг нашивки	0
Плотность нашивки	0
dtype: int64	

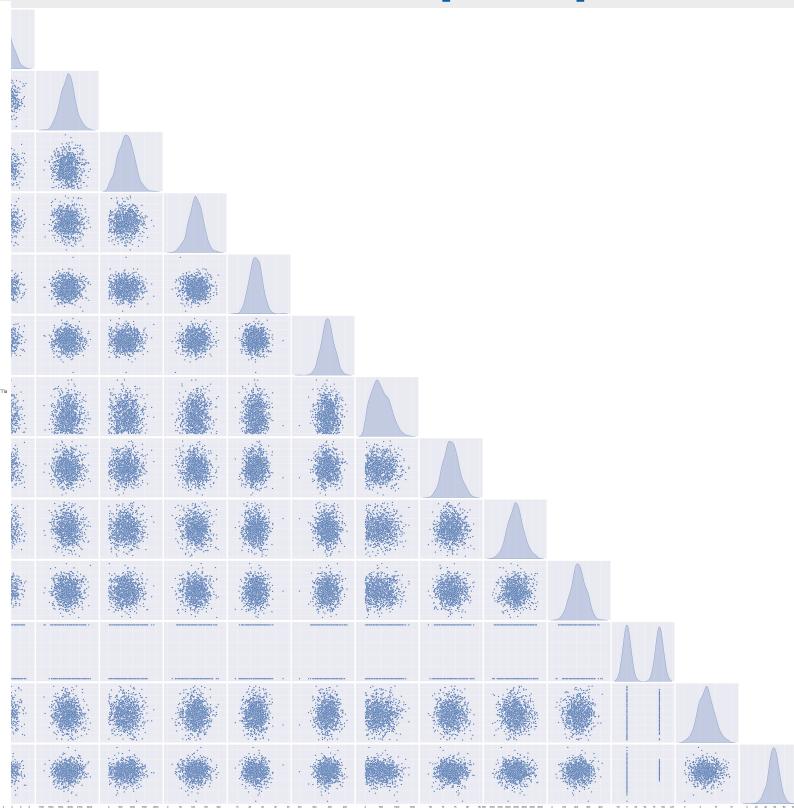


Используем график ящик с усами Boxplot с помощью него мы сможем посмотреть компактное изображение одномерного распределения вероятностей. Такой вид диаграммы показывает медиану (среднее), нижний и верхний квартили и минимальное или максимальное значение выборки, также можно посмотреть выбросы. Как видно на гистограммах, данные стремятся к нормальному распределению практически везде, кроме угла нашивки, имеющим только 2 значения.

Визуализируем данные. Графики.



Модуль упругости при растяжении, ГПа



Прочность при растяжении, МПа"

Данные объединенного датасета не имеют выраженной зависимости, что подтверждает тепловая карта с матрицей корреляции и матрицы диаграмм рассеяния.

Максимальная корреляция между "Плотностью нашивки" и "Углом нашивки" - 0.11, значит нет зависимости между этими данными. Корреляция между всеми параметрами близка к 0.

Смотрим выбросы распределение

Соотношение матрица-наполнитель

Плотность, кг/м3

модуль упругости, ГПа

Количество отвердителя, м.%

Содержание эпоксидных групп, %_2

Температура вспышки, С_2

Поверхностная плотность, г/м2

Модуль упругости при растяжении, ГПа

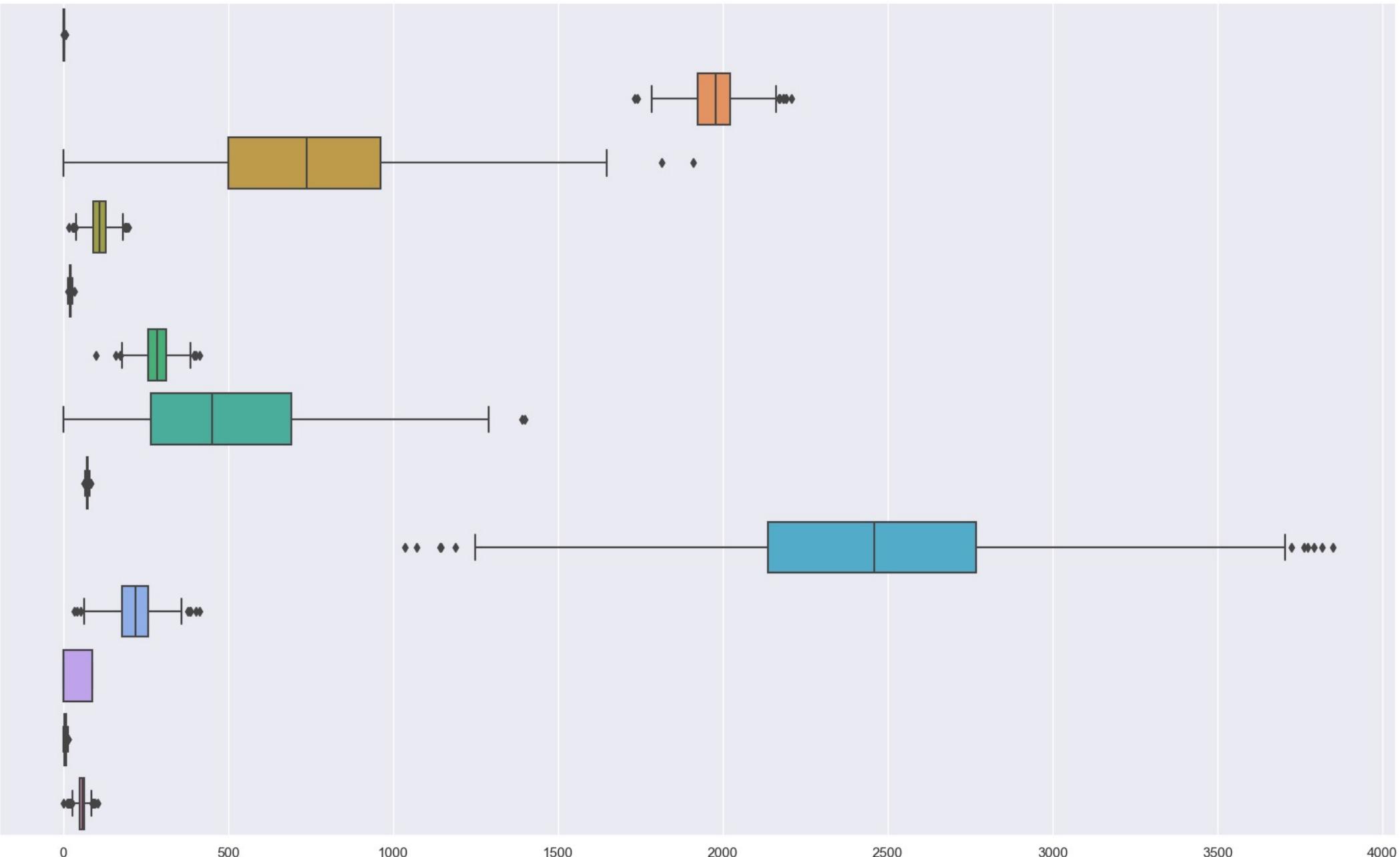
Прочность при растяжении, МПа

Потребление смолы, г/м2

Угол нашивки, град

Шаг нашивки

Плотность нашивки



Удаляем выбросы межквартильным интервалом

Приводим столбец к 0 и 1 Угол нашивки, град

```
# Приводим столбец к 0 и 1
df = df.replace({'Угол нашивки, град': {0.0 : 0, 90.0 : 1}}
df.iloc[:,10] = df.iloc[:,10].astype(int)
df.iloc[:,10]
```

```
0      0
1      0
2      0
3      0
4      0
..
1018    1
1019    1
1020    1
1021    1
1022    1
Name: Угол нашивки, град, Length: 1023, dtype: int64
```

```
# убираем выбросы

dropen = ["Соотношение матрица-наполнитель",
          "Плотность, кг/м3",
          "модуль упругости, ГПа",
          "Количество отвердителя, м.%",
          "Содержание эпоксидных групп,%_2",
          "Температура вспышки, С_2",
          "Поверхностная плотность, г/м2",
          "Модуль упругости при растяжении, ГПа",
          "Прочность при растяжении, МПа",
          "Потребление смолы, г/м2",
          "Шаг нашивки",
          "Плотность нашивки"]

for i in dropen:
    q75, q25 = np.percentile(df.loc[:,i], [75,25])
    intr_qr = q75 - q25
    max = q75 + (1.5 * intr_qr) # выбрана размашика 1,5
    min = q25 - (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)
```

остальные методы применять не
стал, чтобы не терять данные. От
аномальных избавится удалось.



Сравнение двух методов удаления выбросов

- удаление методом межквартильных расстояний или Зсигм

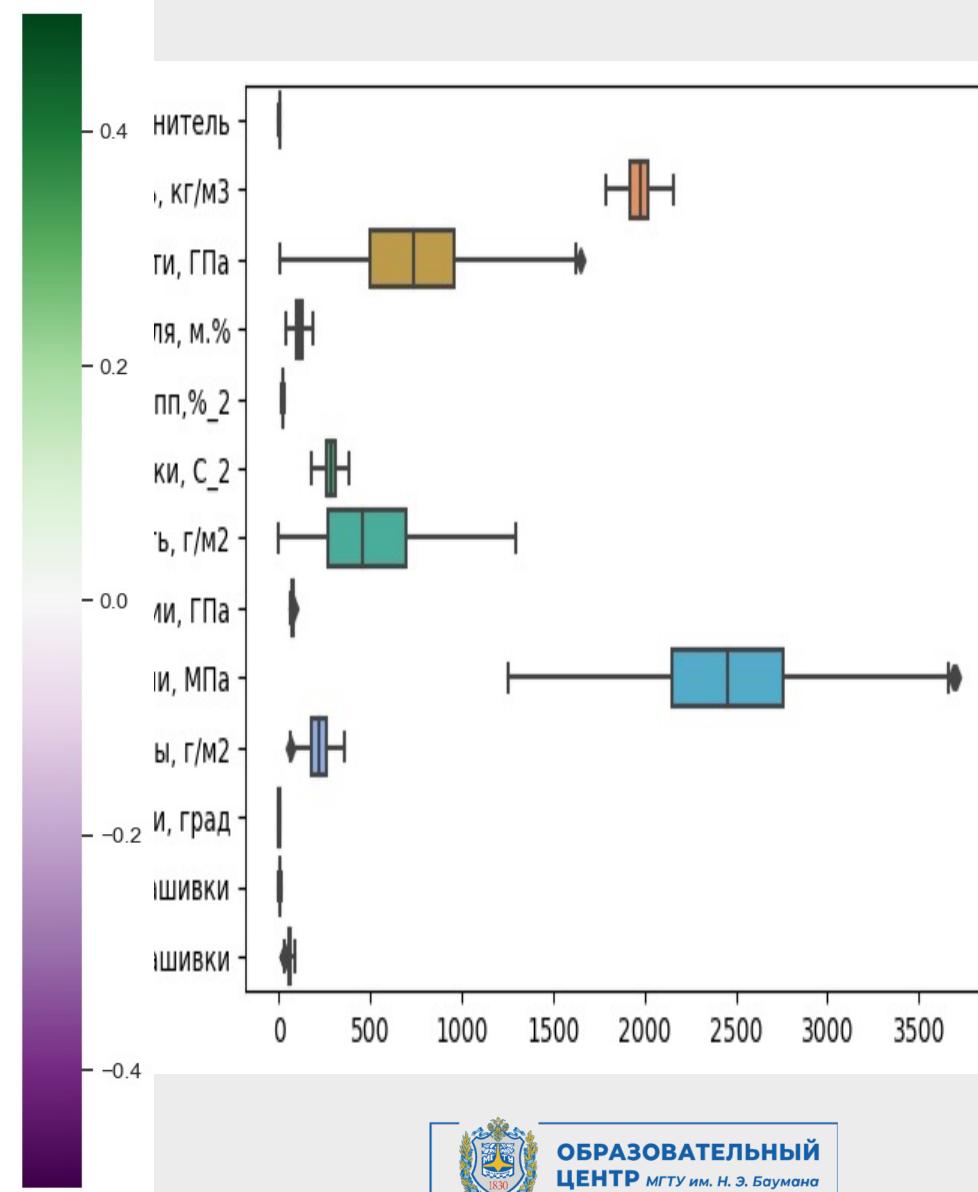
```
Соотношение матрица-наполнитель 3s : 0
Плотность, кг/м3 3s : 3
модуль упругости, ГПа 3s : 2
Количество отвердителя, м.% 3s : 2
Содержание эпоксидных групп,%_2 3s : 2
Температура вспышки, С_2 3s : 3
Поверхностная плотность, г/м2 3s : 2
Модуль упругости при растяжении, ГПа 3s : 0
Прочность при растяжении, МПа 3s : 0
click to expand output; double click to hide output
Потребление смолы, г/м2 : 3
Угол нашивки, град 3s : 0
Шаг нашивки 3s : 0
Плотность нашивки 3s : 7
Выбросы: 24
```

Межквартильные расстояния

```
Соотношение матрица-наполнитель : 6
Плотность, кг/м3 : 9
модуль упругости, ГПа : 2
Количество отвердителя, м.% : 14
Содержание эпоксидных групп,%_2 : 2
Температура вспышки, С_2 : 8
Поверхностная плотность г/м2 : 2
o expand output; double click to hide output
Прочность при растяжении, ГПа : 6
Потребление смолы, г/м2 : 8
Угол нашивки, град : 0
Шаг нашивки : 4
Плотность нашивки : 21
Выбросы: 93
```

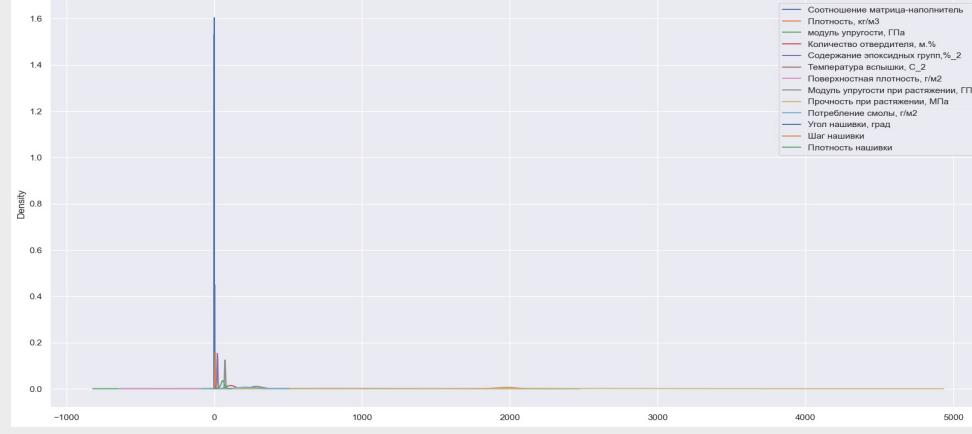
Приемлемый результат дал нам метод межквартильных интервалов его мы и использовали далее на следующем слайде.

После удаления выбросов

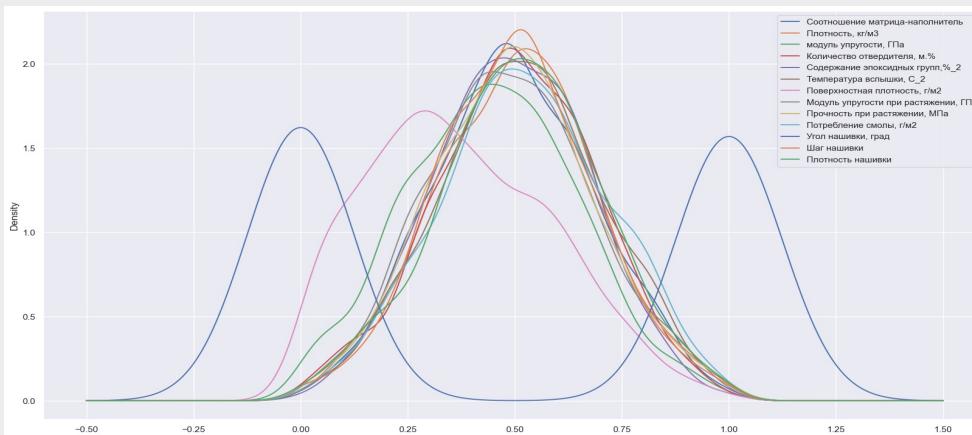


Нормализация данных

Оценка плотности ядра до нормализации



Оценка плотности ядра после нормализации

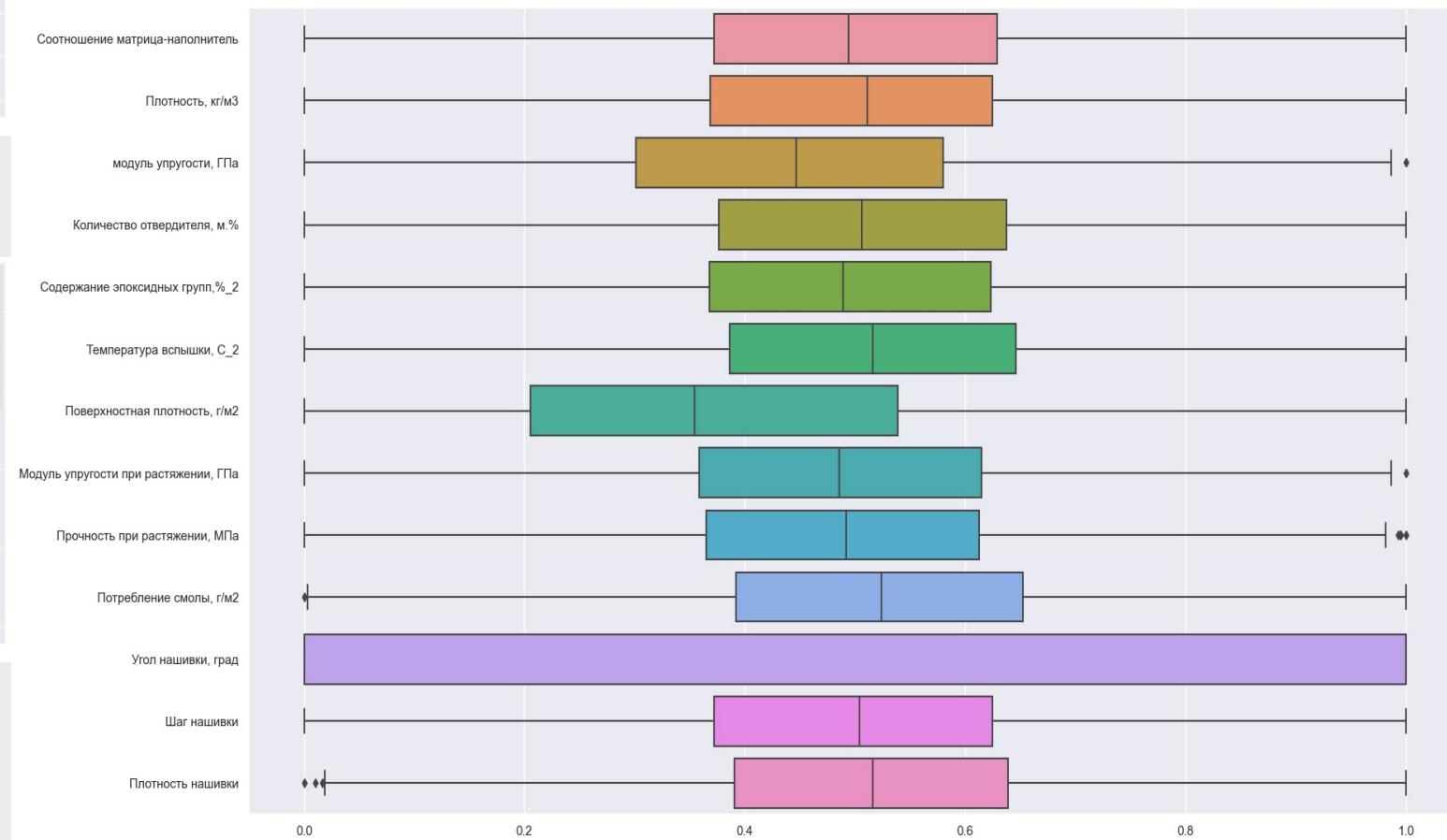


Нормализуем значения. Для этого применим
MinMaxScaler() Данные от 0 до 1 распределились

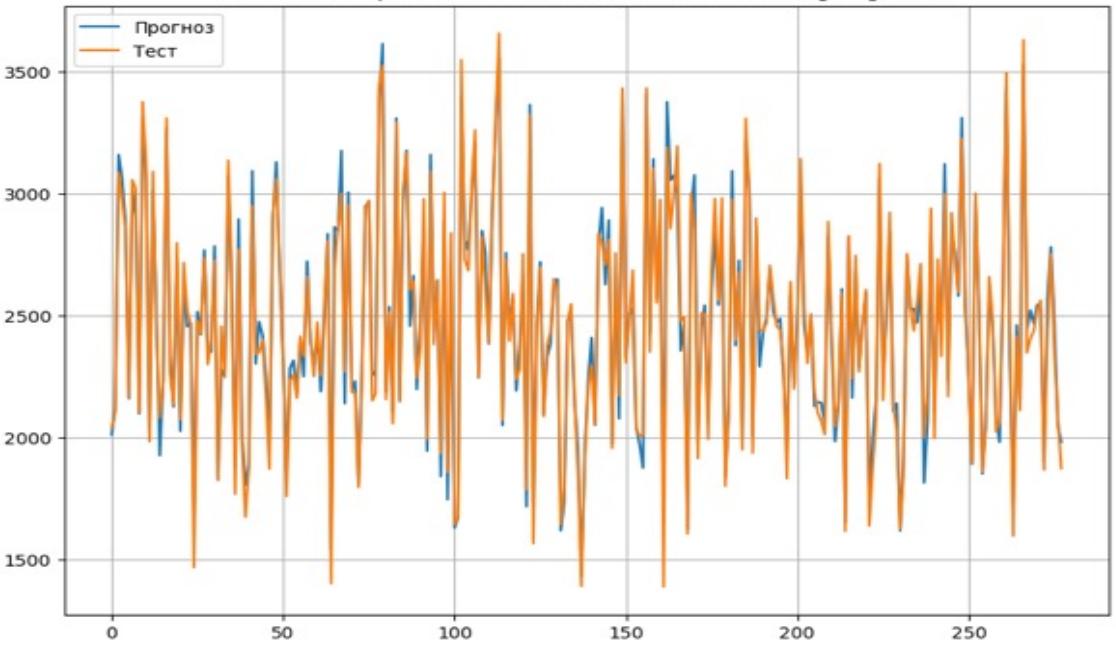
#Нормализация *MinMaxScaler()*

```
scaler = preprocessing.MinMaxScaler()  
col = df_n.columns  
result = scaler.fit_transform(df_n)  
minmax = pd.DataFrame(result, columns = col)  
minmax.describe()  
df_n.head()
```

Посмотрим на Boxplot

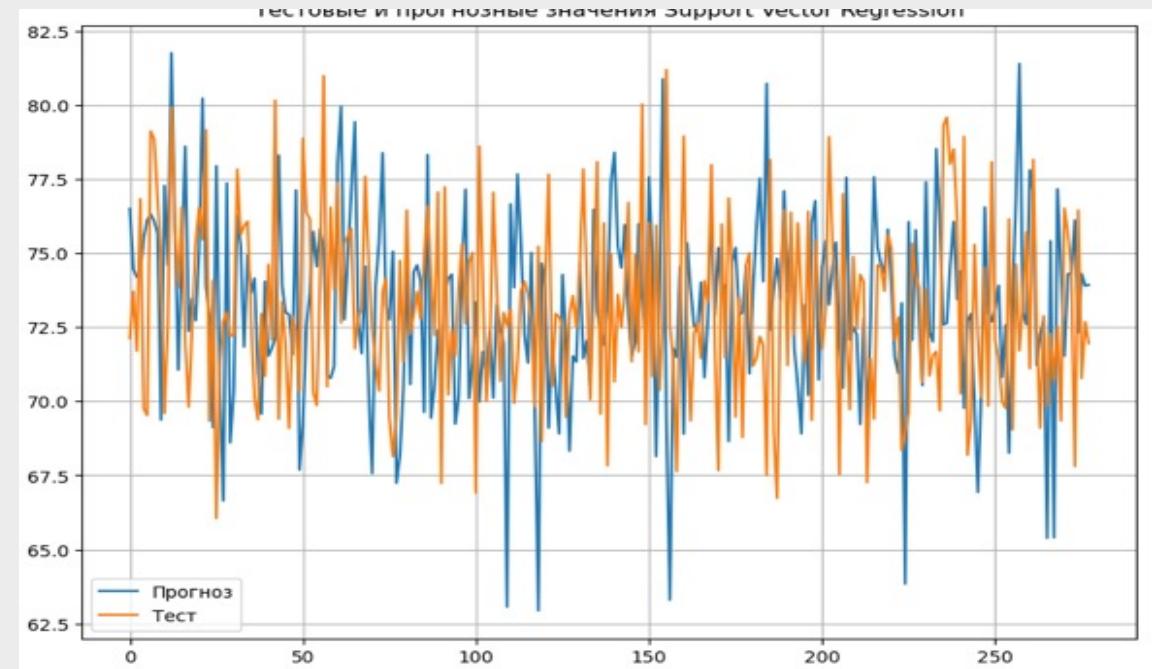


Разработка моделей



Модуль упругости при растяжении, Гпа

лучший метод для прочности при растяжении,
метод градиентного бустинга



Прочность при растяжении, Мпа

Лучший метод для модуля упругости при растяжении –
метод опорных векторов.

При разработке моделей были выполнены следующие этапы:

- разделение нормализованных данных на обучающую и тестовую выборки (в соотношении 70 на 30);
- проверка моделей на нормализованных данных;
- Оценка моделей по метрике MSE, R2.

- К-ближайших соседей
- метод опорных векторов
- линейная регрессия
- случайный лес
- Граюиентный бустинг

Разработка нейронной сети

```
#Сформируем входы и выход для модели
tv = data['Соотношение матрица-наполнитель']
tr_v = data.loc[:, data.columns != 'Соотношение матрица-наполнитель']

#Разбиваем выборки на обучающую и тестовую
x_train, x_test, y_train, y_test = train_test_split(tr_v, tv, test_size=0.2)

#Нормализуем данные
x_train_n = tf.keras.layers.Normalization(axis=-1)
x_train_n.adapt(np.array(x_train))

model_NN.summary()
# Обучим модель

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

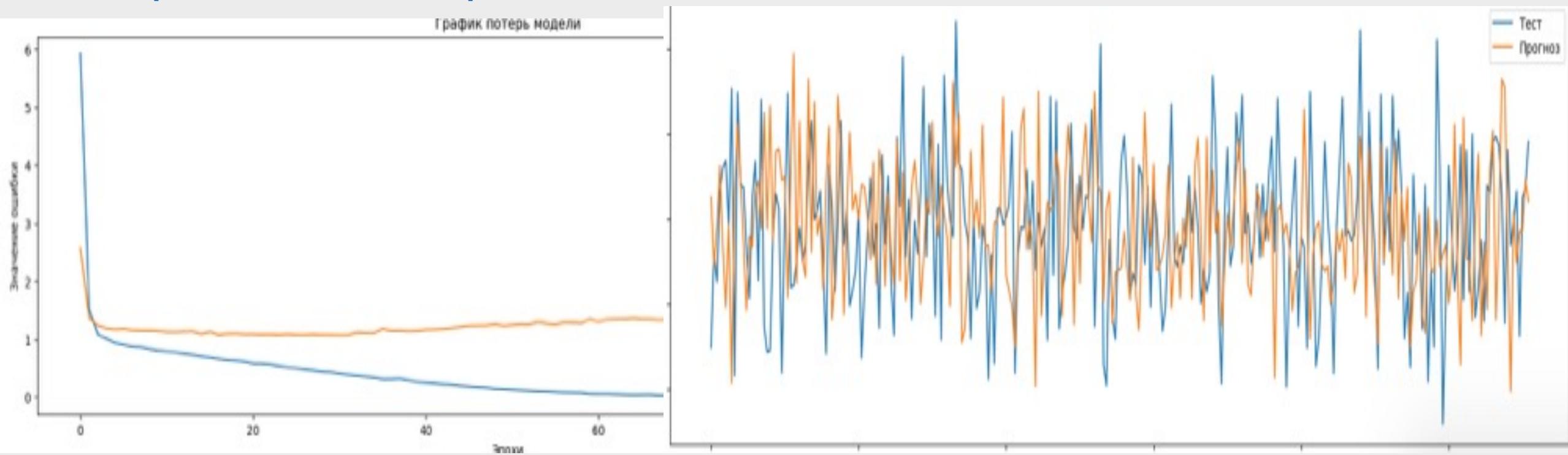
```
model_NN = Sequential([
    x_train_n,
    Dense(128, activation='relu'),
    Dense(64, activation='relu'),
    Dense(16, activation='relu'),
    Dense(3, activation='relu'),
    Dense(3, activation='relu'),
    Dense(1)
])
model_NN.compile(optimizer = tf.keras.optimizers.Adam(0.001), loss='mean_squared_error')
```

```
# 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]
```

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

Разработка нейронной сети



В процессе создания нейронная сеть были определены параметры подбором

Далее было проведены обучение и оценка модели, построены необходимые графики.

Потери на диаграмме быстро уменьшаются в течение первых двадцати эпох. Для тестового набора потери уменьшаются с той же скоростью, что и для обучающего набора. Это означает, что полученная модель нам не подходит далее ошибка растет

Заключение

В процессе выполнения данной работы получилось сделать следующие выводы:

- распределение полученных данных близко к нормальному
- Лоэффициенты корреляции между парами признаков стремятся к нулю
- Часть используемых моделей смогла дать приблизительные прогнозы
- Лучшая модель предсказаний метод опорных векторов и метод градиентного бустинга
- из свойств материалов невозможно с большой точностью определить соотношение «матрица – наполнитель».

Из вышесказанного можно сделать вывод, для того чтобы определить на сколько верны полученные решения необходимо собрать дополнительные данные и провести дополнительные исследования более глубоко вникнуть с помощью специалистов в данной области (консультирование/сотрудничество) Есть подозрение что данные разбавлены или полностью синтетические.



edu.bmstu.ru

+7 495 182-83-85

edu@bmstu.ru

Москва, Госпитальный переулок ,
д. 4-6, с.3