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 Pro»

Слушатель Идрисов Вадим Шамилевич

Москва, 2023

[1.Аналитическая часть 1](#_Toc148001083)

[1.2 Постановка задачи. 1](#_Toc148001084)

[1.2 Методы 2](#_Toc148001085)

[Метод случайного леса (Random Forest) 2](#_Toc148001086)

[Метод дерева решений (Decision Tree) 4](#_Toc148001087)

[Нейронные сети в Sequential Model 6](#_Toc148001088)

[1.3 Методы для анализа данных 8](#_Toc148001089)

[2.Практическая часть 14](#_Toc148001090)

[2.1. Предобработка данных. 14](#_Toc148001091)

[Машинное обучение для Модуль упругости при растяжении Прочность при растяжении 16](#_Toc148001092)

[Нейронная сеть (Соотношение матрица-наполнитель). 20](#_Toc148001093)

[Приложение 23](#_Toc148001094)

[3.Обсуждение 29](#_Toc148001096)

[4.Заключение 29](#_Toc148001097)

[библиографический список 30](#_Toc148001098)

# Аналитическая часть

## 1.2 Постановка задачи.

Для начала опишем постановку задачи. В данном случае, имеется набор данных, состоящий из 1023 наблюдений и 10 признаков (переменных). Однако, одной из целей работы является разработка модели, способной прогнозировать определенный параметр материала (выходной показатель) на основе значений этих 10 признаков. Не менее важно обратить внимание на характеристики самих данных. Сюда включается обнаружение выбросов (необычных значений) и работа с пропущенными данными (если таковые имеются). Очистка данных от выбросов и заполнение пропусков, если это необходимо, представляют собой важную часть предварительной обработки данных .

* Соотношение матрица-наполнитель (Matrix-Filler Ratio): Этот признак представляет собой отношение материала матрицы к наполнителю в композите.
* Плотность кг/м3 (Density kg/m3): Он измеряет массу на единицу объема материала, что является важной характеристикой в материаловедении.
* Модуль упругости, ГПа (Elastic Modulus, GPa): Это мера жесткости материала и его способности возвращаться в исходную форму при деформации.
* Количество отвердителя, м.% (Hardener Content, %): Этот признак указывает на процент отвердителя в композите.
* Содержание эпоксидных групп, %\_2 (Epoxy Group Content, %): Он измеряет процент эпоксидных групп в материале, что важно для адгезивов и покрытий.
* Температура вспышки, С\_2 (Flash Point Temperature, °C): Это самая низкая температура, при которой пары вещества могут загореться в присутствии открытого пламени или искры.
* Поверхностная плотность, г/м2 (Surface Density, g/m2): Этот признак измеряет массу материала на единицу площади.
* Модуль упругости при растяжении, ГПа (Tensile Elastic Modulus, GPa): Это способность материала выдерживать растягивающее воздействие без разрыва.
* Прочность при растяжении, МПа (Tensile Strength, MPa): Он измеряет максимальное напряжение, которое материал может выдержать при растяжении.
* Потребление смолы, г/м2 (Resin Consumption, g/m2): Этот признак указывает на количество смолы, используемое на единицу площади.

Эти признаки являются ключевыми для понимания и предсказания свойств материалов, особенно в приложениях, где механические характеристики, прочность и состав играют важную роль [5-7].

## 1.2 Методы

### Метод случайного леса (Random Forest)

Случайный лес - это мощный ансамблевый метод машинного обучения, который широко используется для задач регрессии и классификации. Он базируется на идее комбинирования нескольких деревьев решений для получения более точных и устойчивых прогнозов. Давайте рассмотрим его ключевые аспекты:

Принцип работы:

* Случайный лес создает ансамбль деревьев решений, где каждое дерево обучается на случайной подвыборке данных.
* Важное свойство случайного леса заключается в использовании бутстрэп выборки, что означает, что каждое дерево обучается на различном подмножестве данных с возвращением.
* Каждое дерево обучается независимо, и прогнозы всех деревьев комбинируются для получения окончательного прогноза.

Преимущества:

* Высокая точность прогнозов: За счет комбинирования прогнозов нескольких деревьев случайного леса, этот метод способен предсказывать целевую переменную с высокой точностью.
* Устойчивость к переобучению: Случайный лес борется с проблемой переобучения, так как каждое дерево обучается на разных подмножествах данных.
* Устойчивость к выбросам: Ансамбль деревьев смягчает влияние выбросов.

Недостатки:

* Интерпретируемость: Случайный лес менее интерпретируем, чем одно дерево решений.
* Подгонка под данные: В некоторых случаях случайный лес может "переобучиться" под обучающие данные и потребовать аккуратного настройки параметров.

Области применения:

* Прогнозирование: Случайный лес успешно применяется в задачах прогнозирования, включая предсказание стоимости, временных рядов, и других непрерывных переменных.
* Классификация: Этот метод также широко используется в задачах классификации, таких как определение категорий, диагностика заболеваний и многие другие.

Априорные предпосылки к работоспособности:

* Наличие достаточного объема данных: Для эффективной работы случайного леса требуется значительное количество данных.
* Подбор параметров: Некоторые параметры, такие как количество деревьев и максимальная глубина деревьев, требуют тщательной настройки для достижения лучших результатов.

Случайный лес представляет собой мощный инструмент для решения задач машинного обучения и может быть полезным в вашем исследовании для прогнозирования параметра материала на основе заданных признаков.

### Метод дерева решений (Decision Tree)

Дерево решений - это метод машинного обучения, который позволяет принимать решения путем пошагового разделения данных на подгруппы на основе значений признаков. Этот метод обладает следующими характеристиками :

Принцип работы:

* Дерево решений представляет собой иерархическую структуру, в которой каждый узел представляет собой решение по определенному признаку.
* Процесс построения дерева начинается с корневого узла, который делит данные на две или более подгруппы в зависимости от значения выбранного признака.
* Этот процесс рекурсивно повторяется для каждой подгруппы, пока не будет достигнут критерий остановки (например, максимальная глубина дерева или минимальное количество наблюдений в узле).

Преимущества:

* Интерпретируемость: Деревья решений легко интерпретировать, их логика принятия решений понятна.
* Простота визуализации: Деревья могут быть визуализированы, что облегчает понимание их работы.
* Работа с категориальными данными: Деревья решений могут обрабатывать как количественные, так и категориальные данные.

Недостатки:

* Неустойчивость к изменениям в данных: Малейшие изменения в данных могут привести к значительным изменениям в структуре дерева.
* Склонность к переобучению: Деревья решений могут легко переобучиться на обучающих данных, что ухудшает их способность к обобщению на новые данные.

Области применения:

* Классификация: Деревья решений широко используются для задач классификации, таких как определение категорий или качественных характеристик.
* Регрессия: Они также применяются в задачах регрессии, где целью является предсказание количественных переменных.
* Принятие решений: Деревья решений могут использоваться для принятия решений в бизнесе и инженерии.

Априорные предпосылки к работоспособности:

* Выбор признаков: Выбор правильных признаков играет ключевую роль в успехе дерева решений. Некорректный выбор признаков может привести к низкой производительности.
* Обработка переобучения: Требуется аккуратный контроль переобучения, например, путем настройки параметров дерева (глубины, критериев остановки).

Дерево решений - это простой и интерпретируемый метод, который может быть полезен для анализа данных и принятия решений.

### Нейронные сети в Sequential Model

Нейронные сети представляют собой мощный класс алгоритмов машинного обучения, вдохновленных функционированием человеческого мозга. Ваше исследование включает использование нейронных сетей в рамках модели Sequential Model. Рассмотрим ключевые аспекты этого метода :

Принцип работы:

* Sequential Model представляет собой структуру нейронной сети, в которой слои располагаются последовательно друг за другом. Каждый слой содержит нейроны, и информация передается от входного слоя к выходному в одном направлении.
* Нейроны в сети связаны весами, которые настраиваются в процессе обучения. Нейронные сети пытаются выучить оптимальные весовые коэффициенты для преобразования входных данных в желаемый выход.

Преимущества:

* Способность к извлечению признаков: Нейронные сети способны автоматически извлекать признаки из данных, что делает их мощными для решения сложных задач.
* Гибкость: Sequential Model позволяет создавать разнообразные архитектуры с различными слоями, что позволяет адаптировать сеть к конкретным задачам.
* Применимость: Нейронные сети могут использоваться как для задач классификации, так и для задач регрессии, а также для обработки изображений, текста, временных рядов и других типов данных.

Недостатки:

* Обучение: Требуется большое количество данных и времени для обучения нейронных сетей, особенно для глубоких архитектур.
* Интерпретируемость: Нейронные сети часто сложно интерпретировать, что может быть проблемой в некоторых приложениях.

Области применения:

* Обработка изображений: Нейронные сети успешно применяются для задач компьютерного зрения, таких как распознавание объектов и сегментация изображений.
* Обработка текста: Их также можно использовать для обработки текста, включая задачи анализа тональности, машинного перевода и генерации текста.
* Рекомендации: Нейронные сети активно применяются в системах рекомендаций, например, в социальных сетях и интернет-магазинах.

Априорные предпосылки к работоспособности:

* Данные: Нейронные сети требуют большого объема данных для обучения, поэтому необходимо обеспечить их наличие.
* Архитектура: Выбор архитектуры нейронной сети (количество слоев, типы слоев) зависит от конкретной задачи и требует экспериментов.
* Гиперпараметры: Настройка гиперпараметров, таких как скорость обучения и размер мини-пакетов, также является важной частью успешного обучения нейронных сетей.

Нейронные сети, особенно в Sequential Model, представляют собой мощный инструмент для решения разнообразных задач машинного обучения, включая задачу рекомендации соотношения матрицы-наполнителя в вашем исследовании.

## 1.3 Методы для анализа данных

В этом разделе используются различные методы для анализа данных:

* Статистические метрики: Оценка основных статистических характеристик каждого признака, таких как среднее значение, медиана, стандартное отклонение, минимальное и максимальное значения. Эти метрики помогут понять общие характеристики признаков .
* Визуализация данных: Использование графиков, таких как гистограммы, box plots и scatter plots, для визуализации распределения данных и взаимосвязей между признаками. Визуализация поможет выявить аномалии и понять корреляции между признаками .
* Корреляционный анализ: Оценка степени взаимосвязи между признаками с помощью корреляционной матрицы. Это поможет определить, какие признаки могут быть взаимосвязаны и оказать влияние на модель .
* Обработка выбросов и пропусков: Идентификация выбросов и разработка стратегии их обработки. Также проверка данных на наличие пропущенных значений и принятие решений о их заполнении или удалении.
* Группировка и агрегация данных: Если необходимо, группировка данных по категориальным признакам и вычисление статистик для каждой группы .
* Оценка важности признаков: Оценка влияния каждого признака на целевую переменную с использованием методов, таких как Feature Importance. Это позволяет определить, какие признаки оказывают наибольшее влияние на исход .
* Визуализация целевой переменной: Построение графиков, отражающих распределение целевой переменной, для лучшего понимания ее характеристик и структуры.
* Дополнительные анализы: Проведение дополнительных анализов, связанных с особенностями данных и поставленной задачей .

Результаты разведочного анализа данных помогут лучше понять данные, выделить важные признаки и принять решения о предобработке данных, что является важным этапом в подготовке данных для построения моделей машинного обучения.

вот пример реализации

|  |
| --- |
| import pandas as pd  import matplotlib.pyplot as plt  import seaborn as sns  # Загрузим файл Excel  file\_path = “X\_bp.xlsx"  data = pd.read\_excel(file\_path)  # Предварительная обработка данных  # Переименуем столбцы для удобства  data.columns = ["id", "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"]  # 1. Статистические метрики  summary = data.describe()  # 2. Визуализация данных  data.hist(figsize=(12, 8))  plt.suptitle("Гистограммы признаков", y=1.02)  plt.show()  # 3. Анализ корреляции  correlation\_matrix = data.corr()  plt.figure(figsize=(10, 8))  sns.heatmap(correlation\_matrix, annot=True, cmap='coolwarm', fmt=".2f")  plt.title("Тепловая карта корреляции")  plt.show() |

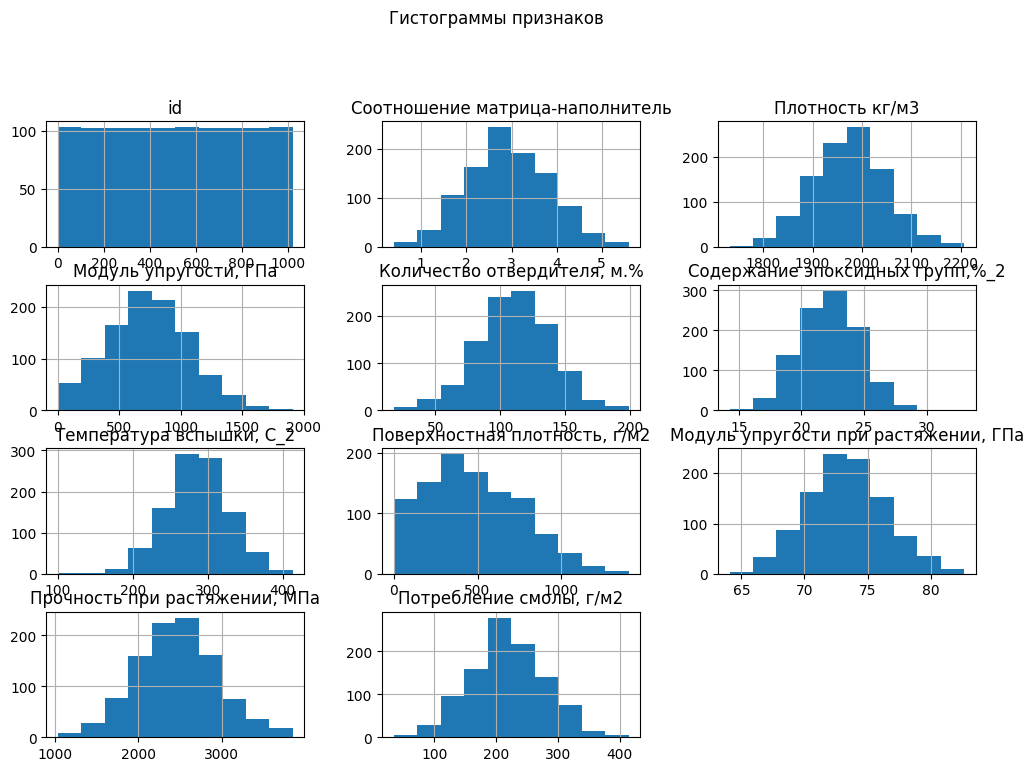


Рис : результаты анализа 1

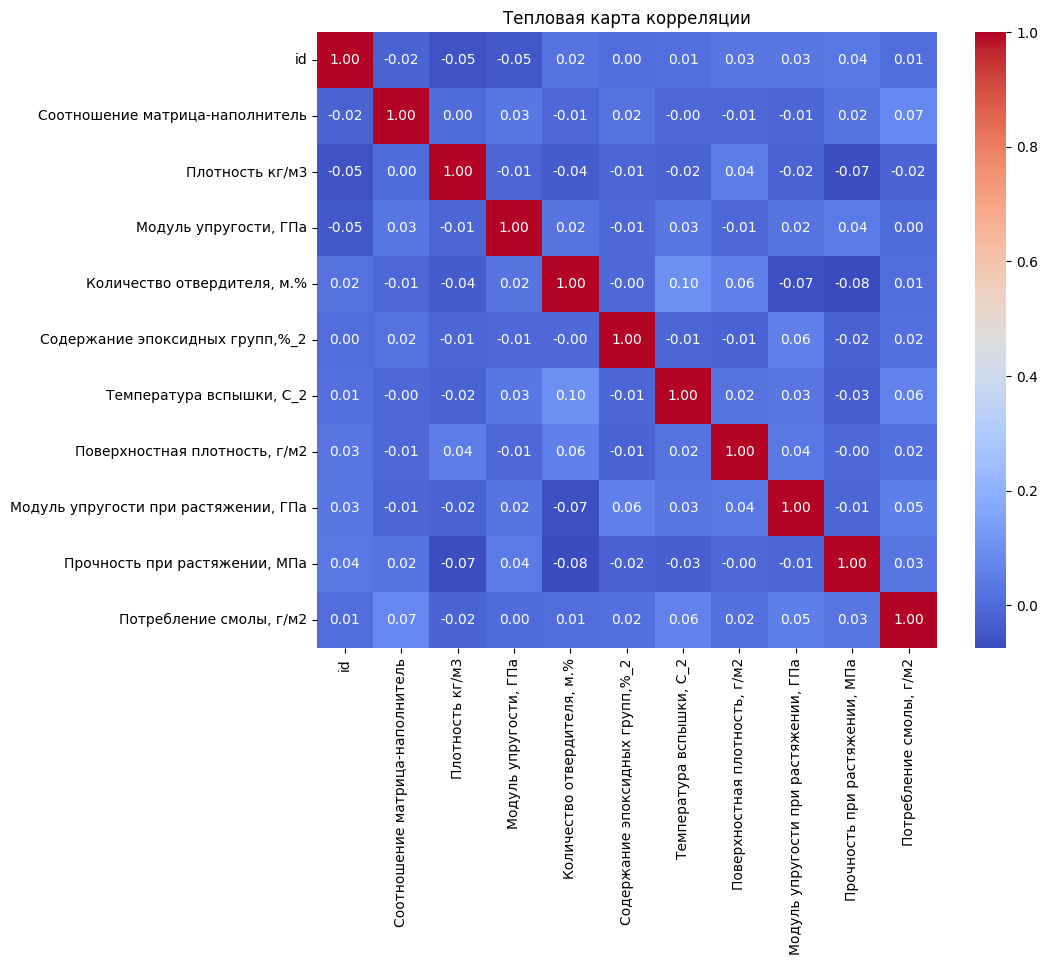


Рис 2: результаты анализа 2

простые методы также могут быть использованы для описания данных, как показано ниже :

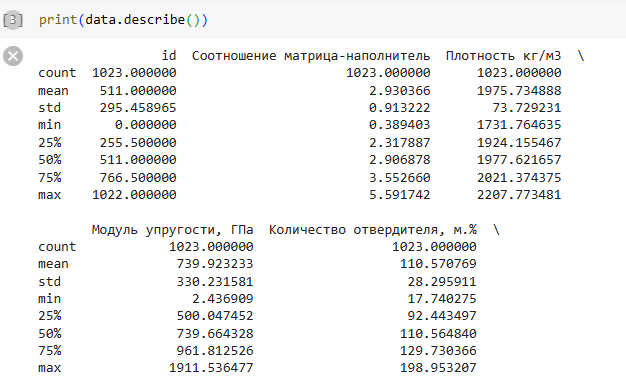


Рис : Результаты описаний 1

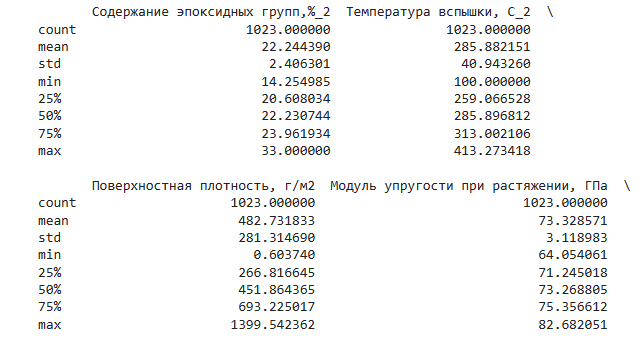


Рис : Результаты описаний 2

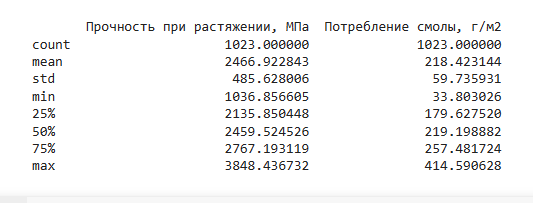


Рис : Результаты описаний 3

# Практическая часть

## 2.1. Предобработка данных.

вот код, который рисует и вычисляет некоторую статистику. результаты сохраняются в файл

|  |
| --- |
| import pandas as pd  import matplotlib.pyplot as plt  from sklearn.preprocessing import StandardScaler  # Создайте список всех признаков, которые не являются целевой переменной  features = ["Соотношение матрица-наполнитель",          "Количество отвердителя, м.%", "Содержание эпоксидных групп,%\_2","Температура вспышки, С\_2", "Поверхностная плотность, г/м2","Потребление смолы, г/м2"]  # Создайте экземпляр стандартизатора для нормализации данных  scaler = StandardScaler()  # Цикл по всем признакам  for feature in features:      # Начало исходных графиков      plt.figure(figsize=(12, 4))        # Гистограмма до нормализации      plt.subplot(1, 2, 1)      plt.title(f"Распределение {feature} (до нормализации)")      data[feature].hist()      plt.xlabel(feature)        # Нормализация признака  data[feature]= scaler.fit\_transform(data[feature].values.reshape(-1, 1))        # Гистограмма после нормализации      plt.subplot(1, 2, 2)      plt.title(f"Распределение {feature} (после нормализации)")      data[feature].hist()      plt.xlabel(feature)        # Вывод максимального и минимального значения      max\_value = data[feature].max()      min\_value = data[feature].min()      print(f"{feature}: Максимальное значение = {max\_value}, Минимальное значение = {min\_value}")        # Завершение и отображение графиков      plt.show()  # Сохраните обновленные данные с нормализованными признаками  data.to\_excel("normalized\_data.xlsx", index=False) |

Соотношение матрица-наполнитель:

Максимальное значение = 2.9156950458420567, Минимальное значение = -2.7837758391782335

Количество отвердителя, м.%:

Максимальное значение = 3.125033401790079,

Минимальное значение = -3.2823081091835653

Содержание эпоксидных групп,%\_2:

Максимальное значение = 4.4719550473120036,

Минимальное значение = -3.321825007080667

Температура вспышки, С\_2:

Максимальное значение = 3.1129318059869338,

Минимальное значение = -4.542214511443346

Поверхностная плотность, г/м2:

Максимальное значение = 3.2606152074884474,

Минимальное значение = -1.7146772884036003

**Данные после выполнения предыдущей нормализации были сохранены в виде нового файла Excel для следующих шагов**

### Машинное обучение для Модуль упругости при растяжении Прочность при растяжении

полные шаги по подготовке алгоритма машинного обучения, который будет определять значения "Модуль упругости при растяжении, Гпа" И "Прочность при растяжении, МПа." приведены ниже (приведенные ниже шаги указаны для алгоритма "Random Forest", также используется "Decision Tree"):

1. Импортируются необходимые библиотеки, такие как pandas, scikit-learn и numpy.
2. Загружается Excel-файл с данными, и столбцы переименовываются для удобства использования.
3. Данные разделяются на признаки (X) и целевую переменную (y).
4. Данные разделяются на обучающий и тестовый наборы для оценки моделей.
5. Создается модель RandomForestRegressor с настройками, такими как количество деревьев (n\_estimators).
6. Модель обучается на обучающих данных, и создаются предсказания для тестового набора.
7. Модель оценивается с использованием среднеквадратической ошибки (MSE) и коэффициента детерминации (R^2).
8. Создается модель DecisionTreeRegressor и аналогично обучается и оценивается.
9. Для новых данных создаются предсказания для обеих моделей (Random Forest и Decision Tree).

полный код для обоих алгоритмов приведен ниже

|  |
| --- |
| # Импорт необходимых библиотек  import pandas as pd  from sklearn.ensemble import RandomForestRegressor  from sklearn.tree import DecisionTreeRegressor  from sklearn.model\_selection import train\_test\_split  from sklearn.metrics import mean\_squared\_error, r2\_score  import numpy as np  # Загрузка Excel-файла  file\_path = “14377959\_1309152X\_bp.xlsx"  data = pd.read\_excel(file\_path)  # Предварительная обработка данных  # Переименование столбцов для удобства использования  data.columns = ["id","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"]  # Разделение данных на признаки (X) и целевую переменную (y)  X = data.drop("Модуль упругости при растяжении ГПа", axis=1)  # Признаки  y = data["Модуль упругости при растяжении ГПа"]  # Целевая переменная  # Разделение данных на обучающую и тестовую выборки  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.25, random\_state=42)  # Создание модели RandomForestRegressor  model = RandomForestRegressor(n\_estimators=500, random\_state=0)  # Обучение модели на обучающих данных  model.fit(X\_train, y\_train)  # Создание предсказаний на тестовом наборе данных  y\_pred = model.predict(X\_test)  # Оценка модели с использованием MSE и R-squared (R^2)  mse = mean\_squared\_error(y\_test, y\_pred)  # Mean Squared Error  r2 = r2\_score(y\_test, y\_pred)  # R-squared (R^2)  print(f"Mean Squared Error Random Forest: {mse}")  print(f"R-squared (R2) Score Random Forest: {r2}")  # Создание модели DecisionTreeRegressor  modelD = DecisionTreeRegressor(random\_state=0)  # Обучение модели на обучающих данных  modelD.fit(X\_train, y\_train)  # Создание предсказаний на тестовом наборе данных  y\_pred = modelD.predict(X\_test)  # Оценка модели с использованием MSE и R-squared (R^2)  mse = mean\_squared\_error(y\_test, y\_pred)  # Mean Squared Error  r2 = r2\_score(y\_test, y\_pred)  # R-squared (R^2)  print(f"Mean Squared Error Decision Tree: {mse}")  print(f"R-squared (R2) Score Decision Tree: {r2}")  # Теперь можно использовать обученные модели для предсказания "Модуль упругости при растяжении ГПа" для новых данных  # Например, можно предсказать значения для конкретного набора признаков, как показано ниже:  new\_data = pd.DataFrame({'Соотношение матрица-наполнитель':[0.6],'Плотность кг/м3':[750],'модуль упругости, ГПа':[620], 'Количество отвердителя, м.%':[10],'Содержание эпоксидных групп,%\_2':[5],'Температура вспышки, С\_2':[80], 'Поверхностная плотность, г/м2':[100],'Прочность при растяжении, МПа':[40], 'Потребление смолы, г/м2':[20]})  # Предсказание с использованием модели Random Forest  predicted\_elasticity = model.predict(new\_data)  print(f"Модуль упругости при растяжении ГПа (RF): {predicted\_elasticity[0]}")  # Предсказание с использованием модели Decision Tree  predicted\_elasticity = modelD.predict(new\_data)  print(f"Модуль упругости при растяжении ГПа (DT): {predicted\_elasticity[0]}") |

эти результаты предназначены для тестирования на части поезда, и таким образом метрика R^2 очень высока

результаты для "Модуль упругости при растяжении, Гпа"

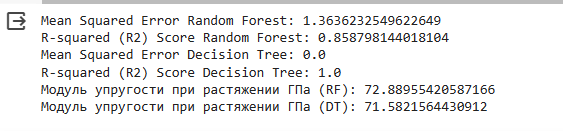


Рис 6: результаты для "Модуль упругости при растяжении, Гпа"

теперь при выборе реальных значений из файла для истинного значения "Модуль упругости при растяжении, Гпа" значение равно 70, результаты

Модуль упругости при растяжении ГПа (RF): 71.21908905889687

Модуль упругости при растяжении ГПа (DT): 70.0

результаты для "Прочность при растяжении, МПа."

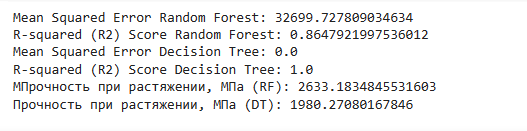


Рис 7: результаты для ""Прочность при растяжении, МПа "

теперь при выборе реальных значений из файла для истинного значения " Прочность при растяжении, МПа " значение равно 3000, результаты

МПрочность при растяжении, МПа (RF): 2877.614229716437

Прочность при растяжении, МПа (DT): 3000.0

### Нейронная сеть (Соотношение матрица-наполнитель).

Рассматриваем следующие шаги, поскольку нейронная сеть является частью искусственного интеллекта, но главным образом как форма глубокого обучения

1. **Загрузка данных**: Данные загружаются из файла Excel.
2. **Предварительная обработка данных**: Выделяются признаки и целевая переменная.
3. **Разделение данных**: Данные разделяются на обучающий и тестовый наборы.
4. **Создание модели регрессии**: Создается нейронная сеть с одним выходным нейроном для регрессии.
5. **Компиляция модели**: Устанавливается оптимизатор и функция потерь.
6. **Обучение модели**: Модель обучается на данных, и мониторятся значения MSE для обучения и валидации в каждой эпохе.
7. **Предсказания**: Делаются предсказания на тестовом наборе данных.
8. **Оценка модели**: Рассчитываются значения MSE и коэффициента детерминации R^2.
9. **Построение графика**: Строится график обучения и валидации для MSE для визуализации процесса обучения модели регрессии.

|  |
| --- |
| import pandas as pd  import numpy as np  from sklearn.model\_selection import train\_test\_split  from tensorflow import keras  from tensorflow.keras.layers import Dense  import matplotlib.pyplot as plt  from sklearn.metrics import mean\_squared\_error, r2\_score  # Загрузка данных из Excel-файла  file\_path = "14377959\_1309152X\_bp.xlsx"  data = pd.read\_excel(file\_path)  # Предварительная обработка данных  X = data.drop(columns=["Соотношение матрица-наполнитель"])  y = data["Соотношение матрица-наполнитель"]  # Разделение данных на обучающую и тестовую выборки  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)  # Создание модели для регрессии  model = keras.Sequential()  model.add(Dense(64, activation='relu', input\_shape=(X\_train.shape[1],)))  model.add(Dense(32, activation='relu'))  model.add(Dense(1, activation='linear'))  # Компиляция модели  model.compile(optimizer='adam', loss='mean\_squared\_error')  # Обучение модели и мониторинг точности  history = model.fit(X\_train, y\_train, epochs=10, batch\_size=64, validation\_data=(X\_test, y\_test))  # Создание предсказаний на тестовом наборе данных  y\_pred = model.predict(X\_test)  # Расчет среднеквадратической ошибки (MSE)  mse = mean\_squared\_error(y\_test, y\_pred)  print(f"Среднеквадратическая ошибка (MSE): {mse}")  # Расчет коэффициента детерминации (R^2) для оценки точности регрессии  r2 = r2\_score(y\_test, y\_pred)  print(f"Коэффициент детерминации (R2): {r2}")  # Построение графика обучения и валидации для MSE  plt.figure(figsize=(8, 6))  plt.plot(history.history['loss'], label='Обучение MSE')  plt.plot(history.history['val\_loss'], label='Валидация MSE')  plt.xlabel('Эпохи')  plt.ylabel('MSE')  plt.legend()  plt.show() |

на рисунке ниже показана кривая функции потерь (среднеквадратичная ошибка)

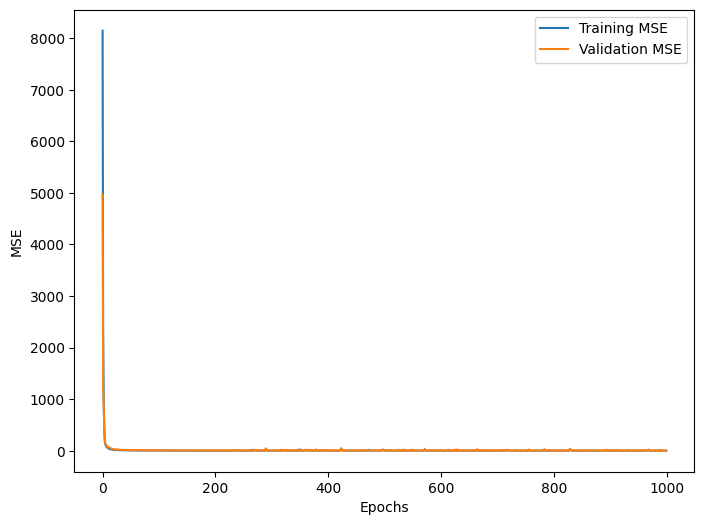


Рис 8: изменение ошибок в процессе обучения в обеих частях: обучающая часть, валидационная часть

### Приложение

полный код, позволяющий учащемуся выбрать переменную

выбор\_учащегося = 0: Модуль упругости при растяжении ГПа

выбор\_учащегося = 1: Прочность при растяжении, МПа

выбор\_учащегося > 1: Соотношение матрица-наполнитель

|  |
| --- |
| выбор\_учащегося=0 # 1 # 3  # Import necessary libraries  import pandas as pd  from sklearn.ensemble import RandomForestRegressor  from sklearn.tree import DecisionTreeRegressor  from sklearn.model\_selection import train\_test\_split  from sklearn.metrics import mean\_squared\_error, r2\_score,accuracy\_score  import numpy as np  from tensorflow import keras  from tensorflow.keras.layers import Dense  # Load the Excel file  file\_path = "X\_bp.xlsx"  data = pd.read\_excel(file\_path)  # Data Preprocessing  # Rename the columns for ease of use  data.columns = ["id","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"]  # Split data into features (X) and target (y)  if выбор\_учащегося==0:      X = data.drop(["id","Модуль упругости при растяжении ГПа"], axis=1)  # Features      y = data["Модуль упругости при растяжении ГПа"]  # Target      new\_data = pd.DataFrame({'Соотношение матрица-наполнитель':[1.857],'Плотность кг/м3':[2030],'модуль упругости, ГПа':[738],'Количество отвердителя, м.%':[30],'Содержание эпоксидных групп,%\_2':[22],'Температура вспышки, С\_2':[100],'Поверхностная плотность, г/м2':[210],'Прочность при растяжении, МПа':[3000],'Потребление смолы, г/м2':[220]})      st='Модуль упругости при растяжении ГПа: '  else    if выбор\_учащегося==1:        X = data.drop(["id","МПрочность при растяжении, МПа"], axis=1)  # Features        y = data["Прочность при растяжении, МПа"]  # Target        new\_data = pd.DataFrame({'Соотношение матрица-наполнитель':[1.857142857],'Плотность кг/м3':[2030],'модуль упругости, ГПа':[738],                    'Количество отвердителя, м.%':[30],'Содержание эпоксидных групп,%\_2':[22],'Температура вспышки, С\_2':[100],                    'Поверхностная плотность, г/м2':[210],'Модуль упругости при растяжении ГПа':[70],                    'Потребление смолы, г/м2':[220]})        st='Прочность при растяжении, МПа: '    else:      X = data.drop(columns=["id","Соотношение матрица-наполнитель"])      y = data["Соотношение матрица-наполнитель"]      new\_data = pd.DataFrame({'Плотность кг/м3':[2030],'модуль упругости, ГПа':[738],                  'Количество отвердителя, м.%':[30],'Содержание эпоксидных групп,%\_2':[22],'Температура вспышки, С\_2':[100],                  'Поверхностная плотность, г/м2':[210],'Модуль упругости при растяжении ГПа':[70],                  'МПрочность при растяжении, МПа':[3000],'Потребление смолы, г/м2':[220]})      st='Соотношение матрица-наполнитель: '  if выбор\_учащегося>1:    model = keras.Sequential()    model.add(Dense(128, activation='relu', input\_shape=(X\_train.shape[1],)))    model.add(Dense(64, activation='relu'))    model.add(Dense(32, activation='relu'))    model.add(Dense(1, activation='linear'))    model.compile(optimizer='adam', loss='mean\_squared\_error')    history = model.fit(X\_train, y\_train, epochs=50, batch\_size=16, validation\_data=(X\_test, y\_test))  else:    # Разделение данных на обучающую и тестовую выборки    X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.25, random\_state=42)    # Создание модели RandomForestRegressor    model = RandomForestRegressor(n\_estimators=500, random\_state=0)    # Обучение модели на обучающих данных    model.fit(X\_train, y\_train)  res = modelD.predict(new\_data)  print(st+str(res[0])) |

Top of Form

Для увеличения точности модели случайного леса можно рассмотреть следующие стратегии:

1. **Настройка гиперпараметров**:
   * Экспериментируйте с разными гиперпараметрами, такими как количество деревьев (n\_estimators), максимальная глубина деревьев (max\_depth) и минимальное количество выборок, необходимых для разделения узла (min\_samples\_split).
   * Используйте техники, такие как поиск по сетке или случайный поиск, чтобы найти наилучшую комбинацию гиперпараметров.
2. **Инженерия признаков**:
   * Тщательно выбирайте и создавайте признаки, наиболее релевантные задаче. Выбор признаков может улучшить производительность модели.
   * Рассмотрите использование методов кодирования категориальных переменных и масштабирования или нормализации числовых признаков при необходимости.
3. **Перекрестная проверка**:
   * Реализуйте техники перекрестной проверки, такие как перекрестная проверка k-блоков, для более надежной оценки производительности модели на невидимых данных. Это помогает избежать переобучения и обеспечивает более надежную оценку.
4. **Работа с несбалансированными данными**:
   * Если ваш набор данных несбалансирован, рассмотрите техники, такие как увеличение, уменьшение или использование синтетической генерации данных (например, SMOTE) для балансировки распределения классов.
5. **Методы ансамблирования**:
   * Экспериментируйте с другими методами ансамблирования, такими как градиентный бустинг (например, XGBoost, LightGBM) и сравните их производительность с моделью случайного леса.
6. **Очистка данных**:
   * Рассмотрите вопросы отсутствующих данных путем их замещения или удаления в зависимости от характера данных.
   * Выявление и обработка выбросов также может повысить точность модели.
7. **Выбор признаков**:
   * Используйте методы выбора признаков для идентификации и сохранения только наиболее значимых признаков, что поможет уменьшить шум в модели.
8. **Аугментация данных**:
   * Если у вас маленький набор данных, рассмотрите методы аугментации данных для искусственного увеличения обучающей выборки.
9. **Стекинг моделей**:
   * Комбинируйте предсказания нескольких моделей, включая случайный лес, с использованием стекинга, чтобы создать ансамбль моделей. Это часто способствует улучшению общей точности.
10. **Регуляризация**:
    * Применяйте техники регуляризации, такие как отсев признаков или L1/L2 регуляризация, для уменьшения переобучения.
11. **Оптимизация метрик производительности**:
    * Выбирайте подходящие метрики производительности, соответствующие вашей задаче и бизнес-целям. Это может включать точность, полноту, F1-меру или другие метрики в зависимости от задачи.
12. **Техники ресемплинга**:
    * Экспериментируйте с различными техниками ресемплинга, такими как бутстрэпинг, чтобы обучать модель на разных подмножествах данных.
13. **Снижение размерности**:
    * Рассмотрите методы снижения размерности, такие как анализ главных компонент (PCA), для уменьшения сложности данных.
14. **Интерпретируемость модели**:
    * Используйте техники анализа важности признаков, чтобы понять, какие признаки влияют на прогнозы модели.
15. **Сбор большего количества данных**:
    * При возможности собирайте больше данных для увеличения разнообразия и представительности набора данных.

Помните, что эффективность этих стратегий может варьироваться в зависимости от конкретных характеристик вашего набора данных и задачи. Часто разумно экспериментировать с несколькими подходами и тщательно оценивать производительность модели.

# Обсуждение

Результаты показали, что модели хороши в прогнозировании, но для этого требуется больше этапов обучения или оптимизации, кроме того, также тестируются другие методы, такие как регрессия опорных векторов (SVR) и K ближайших соседей (KNN) , но их результаты не так высоки, как у случайного леса и дерева решений

# Заключение

в этом исследовании объяснены многие важные этапы анализа и обработки данных, протестированы на данных и показаны результаты. Также были разработаны и реализованы некоторые модели машинного обучения для функции прогнозирования материала с использованием других доступных функций. Результаты показали, что реализованные модели обладают способностью к прогнозированию скважины.

# 

# Библиографический список

1. Батаев А. А. Композиционные материалы. – Логос, 2006.
2. Адаменко Н. А., Агафонова Г. В., Фетисов А. В. Полимерные композиционные материалы. – 2016.
3. Колосова А. С. и др. Современные полимерные композиционные материалы и их применение //Международный журнал прикладных и фундаментальных исследований. – 2018. – №. 5-1. – С. 245-256.
4. Дунаевский А. С. Создание ionic чата для мобильных устройств при помощи react. js //Научные достижения и открытия современной молодёжи. – 2019. – С. 58.
5. Савицкий В. Я., Муйземнек А. Ю. РАЗРАБОТКА ПОДХОДА К ОЦЕНКЕ МИКРОСТРУКТУРНОЙ ПРОЧНОСТИ ПОДШИПНИКОВ СКОЛЬЖЕНИЯИЗ ПОЛИМЕРНЫХ КОМПОЗИЦИОННЫХ МАТЕРИАЛОВ //Национальные приоритеты России. Серия 1: Наука и военная безопасность. – 2015. – №. 1. – С. 20-25.
6. Гасанов А. Б., Вандоловский А. Г. Повышение прочности при растяжении конструкционного керамзитобетона //Науковий вісник будівництва. – 2015. – №. 2. – С. 162-166.
7. Раскутин А. Е., Соколов И. И. Углепластики и стеклопластики нового поколения //Труды ВИАМ. – 2013. – №. 4. – С. 9.
8. Чистяков С. П. Случайные леса: обзор //Труды Карельского научного центра Российской академии наук. – 2013. – №. 1. – С. 117-136.
9. Segal M. R. Machine learning benchmarks and random forest regression. – 2004.
10. Biau G., Scornet E. A random forest guided tour //Test. – 2016. – Т. 25. – С. 197-227.
11. Мифтахова А. А. Применение метода дерева решений для решения задач классификации и прогнозирования //Инфокоммуникационные технологии. – 2016. – Т. 14. – №. 1. – С. 64-70.
12. Гордеева С. М., Малинин В. Н. О предвычислении годового стока крупных рек Европейской части России на основе метода деревьев решений (decision trees) //Ученые записки Российского государственного гидрометеорологического университета. – 2018. – №. 50. – С. 53-65
13. Кафтанников И. Л., Парасич А. В. Особенности применения деревьев решений в задачах классификации //Вестник Южно-Уральского государственного университета. Серия: Компьютерные технологии, управление, радиоэлектроника. – 2015. – Т. 15. – №. 3. – С. 26-32.
14. Гафаров Ф. М. и др. Искусственные нейронные сети и приложения. – 2018.
15. Созыкин А. В. Обзор методов обучения глубоких нейронных сетей //Вестник Южно-Уральского государственного университета. Серия: Вычислительная математика и информатика. – 2017. – Т. 6. – №. 3. – С. 28-59.
16. Круг П. Г. Нейронные сети и нейрокомпьютеры. – 2002.
17. Вапнярская О. И. Развитие детского туризма: основные статистические метрики //Сервис в России и за рубежом. – 2018. – Т. 12. – №. 2 (80). – С. 46-58.
18. Нефедьева К. В. Инфографика визуализация данных в аналитической деятельности //Труды Санкт-Петербургского государственного института культуры. – 2013. – Т. 197. – С. 89-93.
19. Конюхова Г. П., Бритвина В. В., Конюхов В. Г. Методы корреляционного анализа. – 2012.
20. Попов Д. Е., Устовицкий М. В., Кудрявцева Т. С. Программный модуль агрегации для программы для визуализации данных. – 2017.
21. Колесникова С. И., Янковская А. Е. Оценка значимости признаков для тестов в интеллектуальных системах //Известия российской академии наук. Теория и системы управления. – 2008. – №. 6. – С. 99-112.
22. Епанчинцева Е. А. и др. Количественные и качественные нарушения в спермограмме и дополнительных анализах эякулята у мужчин из бесплодных пар //Problemy Reproduktsii. – 2017. – Т. 23. – №. 6.
23. Сузи Р. А. Язык программирования Python //М.: Бином. Лаборатория знаний. – 2006.
24. Paul A. et al. Improved random forest for classification //IEEE Transactions on Image Processing. – 2018. – Т. 27. – №. 8. – С. 4012-4024.
25. Yin L. et al. Research on stock trend prediction method based on optimized random forest //CAAI Transactions on Intelligence Technology. – 2023. – Т. 8. – №. 1. – С. 274-284.
26. Chaudhary A., Kolhe S., Kamal R. An improved random forest classifier for multi-class classification //Information Processing in Agriculture. – 2016. – Т. 3. – №. 4. – С. 215-222.
27. Smola A. J., Schölkopf B. A tutorial on support vector regression //Statistics and computing. – 2004. – Т. 14. – С. 199-222.
28. Awad M. et al. Support vector regression //Efficient learning machines: Theories, concepts, and applications for engineers and system designers. – 2015. – С. 67-80.
29. Kramer O., Kramer O. K-nearest neighbors //Dimensionality reduction with unsupervised nearest neighbors. – 2013. – С. 13-23.
30. Imandoust S. B. et al. Application of k-nearest neighbor (knn) approach for predicting economic events: Theoretical background //International journal of engineering research and applications. – 2013. – Т. 3. – №. 5. – С. 605-610.