**Тема №1**: **Генерация выборки для заданных функций с моделированием случайной ошибки измерения.**

**Особенности реализации**: использовались библиотеки Python nympy, matplotlib.pyplot

**Эксперимент**:

1. Была сформирована выборка , , где – ошибка, генерируемая случайно из интервала , генерируется случайно в соответствии с равномерным распределением на интервале [-1, 1]
2. Ошибка распределена равномерно, коэффициенты сгенерированы случайно из интервала [-3, 3]
3. Ошибка распределена нормально,
4. График функции f и выборка были отображены на координатной плоскости

**Результаты**:

1. Изображение выглядит как текст, линия, График, диаграмма

   Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как линия, График, диаграмма, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, График, линия, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, линия, График, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, График, линия, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, График, линия, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

1. Изображение выглядит как текст, График, линия, диаграмма

   Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, График, линия, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, График, линия, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, График, линия, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, График, линия, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, График, линия, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

**Тема №2: Восстановление функциональной зависимости с помощью полиномиальной регрессии.**

**Особенности реализации**: использовались библиотеки Python nympy, matplotlib.pyplot, sklearn.preprocessing, sklearn.linear\_model, sklearn.pipeline, sklearn.metrics, sklearn.model\_selection

**Эксперимент**: было реализовано с помощью полиномиальной регрессии для полученной выборки восстановление функциональной зависимости, по которой получена выборка и приведены примеры недообучения, переобучения и пример, где полученная функциональная зависимость пригодна для прогнозирования значения восстанавливаемой функциональной зависимости в x, не содержащемся в выборке.  
**Результаты**:

1. Недообучение

N = 10

Epsilon = 0.5

Изображение выглядит как текст, График, линия, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

1. Переобучение

N = 10

Epsilon = 0.5

Изображение выглядит как текст, диаграмма, линия, График

Контент, сгенерированный ИИ, может содержать ошибки.

1. функциональная зависимость пригодна для прогнозирования

N = 10

Epsilon = 0.5

Изображение выглядит как текст, График, линия, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

1. Недообучение

N = 10

EPSILON = 0.5

MEAN = 1

STD\_DEV = EPSILON / 3

Изображение выглядит как текст, линия, График, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

1. Переобучение

N = 10

EPSILON = 0.5

MEAN = 1

STD\_DEV = EPSILON / 3

Изображение выглядит как текст, линия, График, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

1. функциональная зависимость пригодна для прогнозирования

тут особо нет примеров

N = 10

EPSILON = 0.5

MEAN = 1

STD\_DEV = EPSILON / 3

Изображение выглядит как текст, линия, График, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, График, линия, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.