Міністерство освіти і науки України Національний технічний університет України «КПІ» імені Ігоря Сікорського Кафедра інформатики та програмної інженерії ФІОТ

ЗВІТ з лабораторної роботи №2 з навчальної дисципліни «Технології Computer Vision»

Тема:

СТАТИСТИЧНЕ НАВЧАННЯ З ПОЛІНОМІАЛЬНОЮ РЕГРЕСІЄЮ

Виконав
Студент 3 курсу кафедри ІПІ ФІОТ,
Навчальної групи ІП-12
Васильєв Є.К.

Перевірив
Професор кафедри ОТ ФІОТ
Писарчук О.О.

Київ 2023

І. Мета:

Виявити дослідити та узагальнити особливості реалізації процесів статистичного навчання із застосуванням методів обробки Big Data масивів та калмановської рекурентної фільтрації з використанням можливостей мови програмування Python.

ІІ. Завдання:

Група вимог_1:

1. Отримання вхідних даних із властивостями, заданими в Лр_1;
2. Модель вхідних даних із аномальними вимірами;
3. Очищення вхідних даних від аномальних вимірів. Спосіб виявлення аномалій та очищення обрати самостійно;
4. Визначення показників якості та оптимізація моделі (вибір моделі залежно від значення показника якості). Показник якості та спосіб оптимізації обрати самостійно.
5. Статистичне навчання поліноміальної моделі за методом найменших квадратів (МНК – LSM) – поліноміальна регресія для вхідних даних, отриманих в п.1,2. Спосіб реалізації МНК обрати самостійно;
6. Прогнозування (екстраполяцію) параметрів досліджуваного процесу за «навченою» у п.5 моделлю на 0,5 інтервалу спостереження (об'єму вибірки);
7. Провести аналіз отриманих результатів та верифікацію розробленого скрипта.

ІІІ. Результати виконання лабораторної роботи.

3.1. Підготовка даних;

Аналізуючи помилки допущені у попередній роботі, було спочатку прибрано тренд з початкових даних, та підтверджено нормальний закон розподілу випадкової складової реальних даних критерієм Шапіро — Вілка. Також для того, щоб додати шум до синтезованої у попередній лабораторній роботі моделі, було прораховано статистичні характеристики вибірки без тренду. Після цього, до моделі було додано шум та аномальні виміри аналогічні реальним даним.

Використовуючи ковзне вікно та рухому статистику, було виявлено аномалії на основі Z – оцінки, які були замінені математичним сподіванням моделі. Аналізуючи статистичні характеристики моделі з видаленими аномальними вимірами, було підібрано оптимальні параметри (ширина вікна та поріг Z – оцінки) для відповідного методу. Також зберігаючи елементи вибірки, до яких були додані аномалії, було оцінено якість їх видалення: з 11 замінених аномальних вимірів - 5 (з 10 доданих) було видалено правильно та ще 6 неправильно. Отже в 45% метод виявляє аномалії коректно, та в 55% - некоректно.

3.2. Синтезована математична модель;

Використовуючи метод найменших квадратів було створено поліноміальну модель другого порядку (був обраний саме другий порядок оскільки кубічна модель не давала приросту в точності).

Загальний вигляд квадратичної моделі має наступний вигляд: $Y = a * X^2 + b * X + c$, Де Y – залежна змінна, яку ми намагаємось передбачити, X – незалежна змінна, яка впливає на Y, $a, b,$ і c – коефіцієнти моделі, які потрібно оцінити.

Квадратична модель має три параметри: $a, b,$ і c. Параметр a відповідає за ступінь квадратичного впливу X на Y. Якщо a додатнє, то це означає, що зміна X спричиняє квадратичний ріст в Y. Якщо а від'ємне, то зміна X спричиняє квадратичне зменшення в Y.

Параметр b представляє лінійний вплив X на Y. Він вказує на те, як зміна X впливає на Y в лінійному режимі.

Параметр c - це константа, або відома як вільний член, який вказує на значення Y, коли X рівний нулю.

3.3. Результати архітектурного проектування та їх опис;

Після навчання поліноміальної регресії було здійснено екстраполяцію параметрів досліджуваного процесу на 50% від об’єму вибірки. Також було відтворено коефіцієнти моделі та здійснено відповідну візуалізацію.

3.4. Опис структури проекту програми;

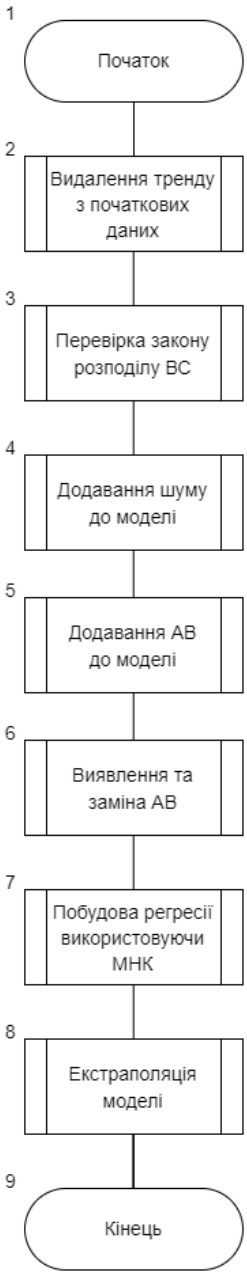

Рис.1. Блок схема алгоритму програми.

3.5. Результати роботи програми відповідно до завдання (допускається у формі скріншотів);

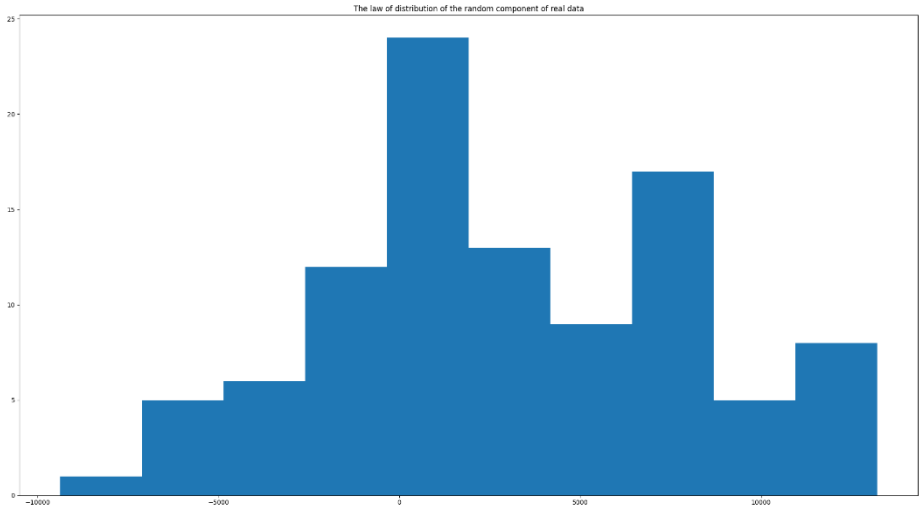

Рис.2. Розподіл випадкової складової реальних даних.

```
Statistical characteristics of the original data without trend:
Mean: 3014.92
Median: 2123.51
Variance: 24264275.47
Standard Deviation: 4925.88
```

Рис.3. Статистичні характеристики випадкової складової реальних даних.

```
0.2364574670791626
The data appears to be normally distributed (fail to reject the null hypothesis)
```

Рис.4. Перевірка закону розподілу ВС реальних даних (p-значення).

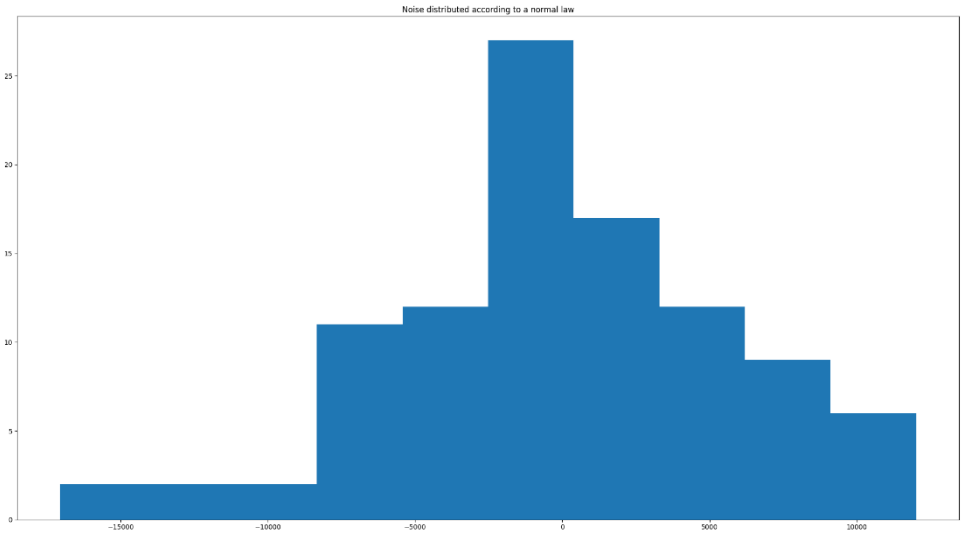

Рис.5. Розподіл створеного шуму, аналогічного до реальних даних.

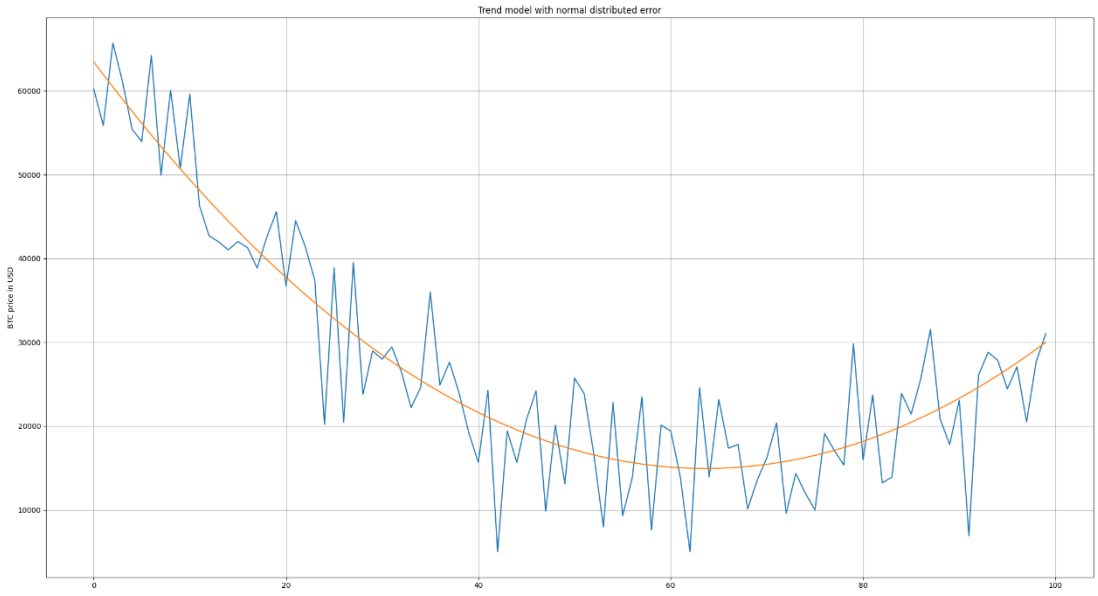

Рис.6. Візуалізація моделі з доданим шумом.

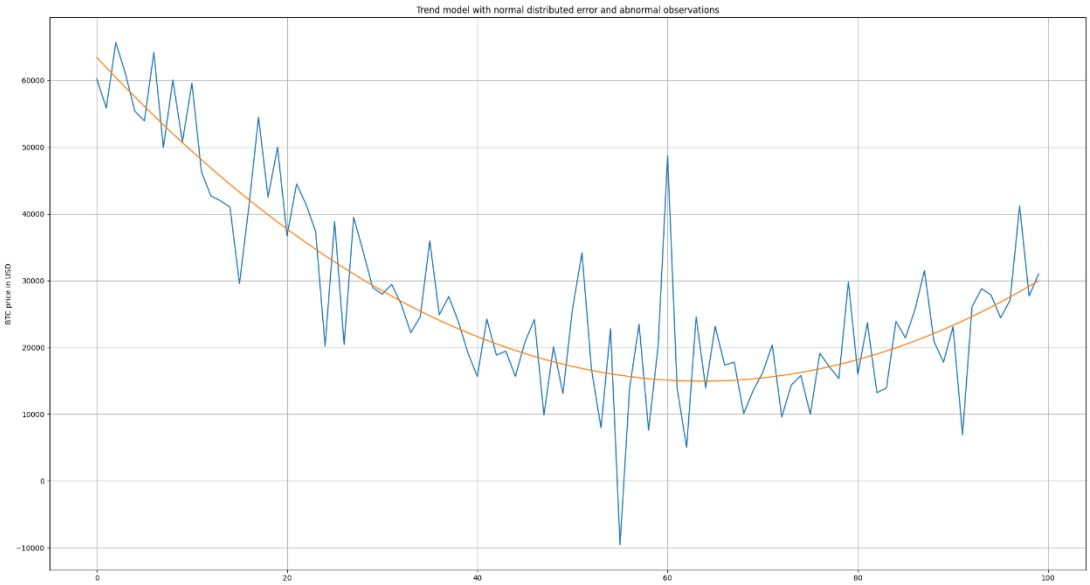

Рис.7. Візуалізація моделі з доданим шумом та аномальними вимірами.

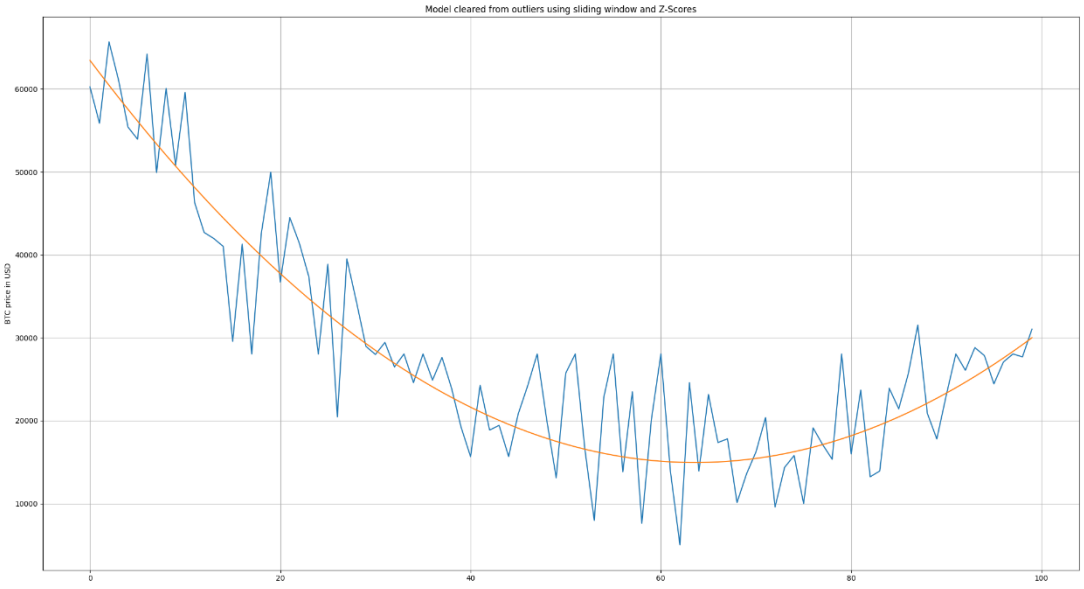

Рис.8. Візуалізація моделі з шумом очищеної від аномальних вимірів.

```
Indices where anomalies have been added:  [15, 17, 19, 28, 42, 51, 55, 60, 74, 97]
Indices where anomalies have been replaced:  [17 24 33 35 47 51 55 60 79 91 97]
```

Рис.9. Порівняння знайдених та фактичних аномалій.

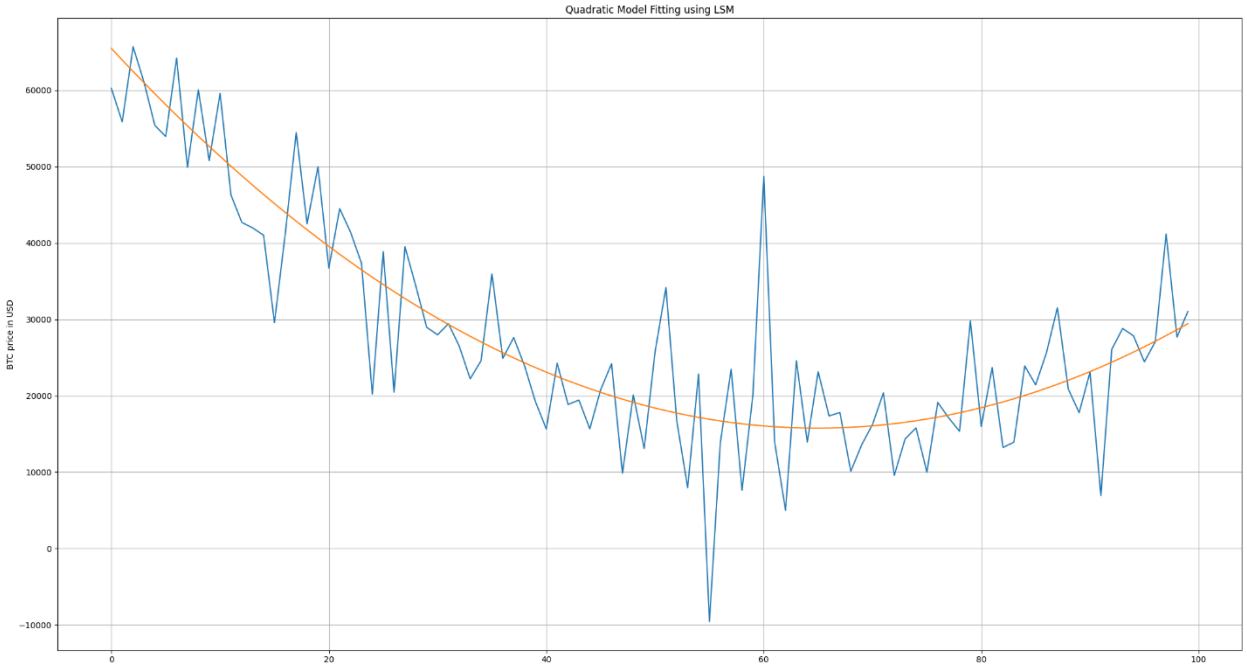

Рис.10. Візуалізація поліноміальної моделі навченої за МНК.

```
Regression Equation: y = 65485.10 + -1531.52x + 11.79x^2
```

Рис.11. Рівняння навченої квадратичної регресії.

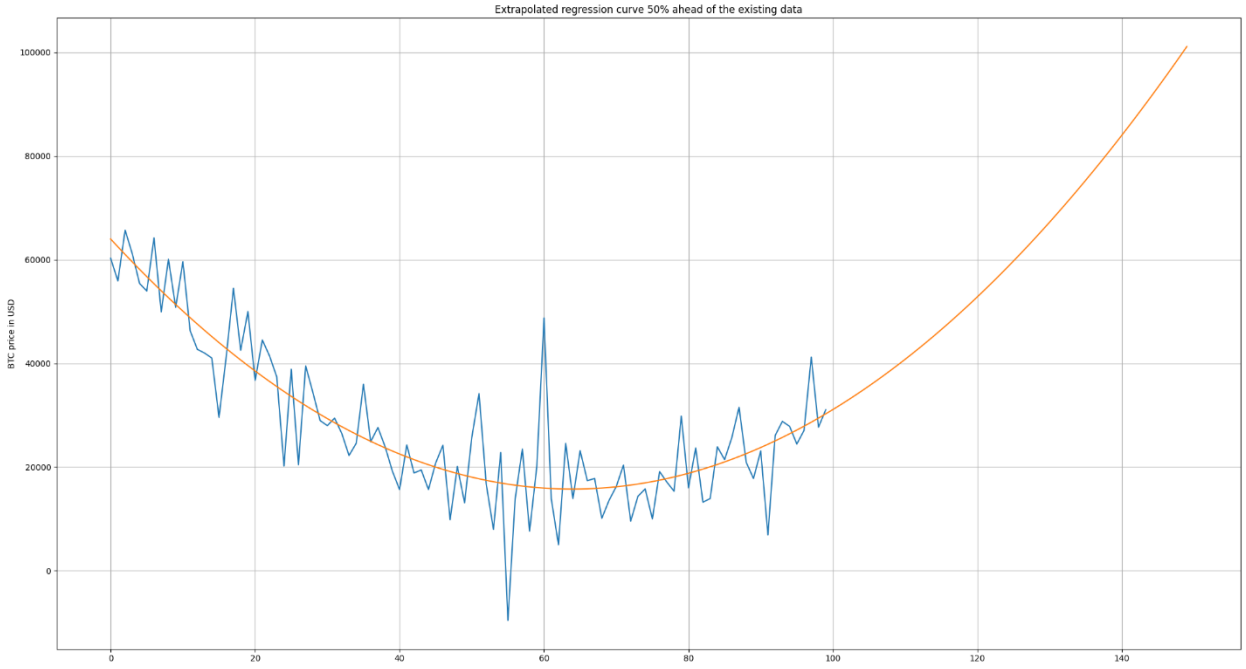

Рис.12. Екстраполяція моделі на 50% об’єму вибірки.

```
Statistical characteristics for the dataset
Mean: 30341.92
Median: 27355.22
Variance: 138221042.10
Standard Deviation: 11756.74

Statistical characteristics of the model with noise:
Mean: 27270.73
Median: 23895.47
Variance: 212210970.80
Standard Deviation: 14567.46

Statistical characteristics of the data with normal distributed error and abnormal observations:
Mean: 28042.91
Median: 24348.77
Variance: 228642564.71
Standard Deviation: 15120.93

Statistical characteristics of the data cleared from outliers:
Mean: 28187.84
Median: 25690.81
Variance: 191724401.07
Standard Deviation: 13846.46

Statistical characteristics of the quadratic model fitted with LSM:
Mean: 28395.20
Median: 23129.40
Variance: 187663951.71
Standard Deviation: 13699.05
```

Рис.13. Порівняння статистичних характеристик.

3.6. Програмний код, що забезпечує отримання результату (допускається у формі скріншотів).

```python
"""
Виконав: Васильєв Єгор
Lab_work_2, І група вимог
"""

import sys
import random
import numpy as np
from scipy import stats
from scipy.optimize import curve_fit
import matplotlib.pyplot as plt

sys.path.append('C:/Users/egorv/PycharmProjects/pythonProject/Data science/Lab1')
from Lab1 import y_values_synthetic, rates, print_characteristics, stat_characteristics
```

```python
def check_normality(data):
    """

    Parameters
    ----------
    data: the data to be tested for normality.

    Returns
    -------
    void
    """
    # Perform the Shapiro-Wilk test
    statistic, p_value = stats.shapiro(data)
    alpha = 0.05
    print(p_value)
    if p_value > alpha:
        print("The data appears to be normally distributed (fail to reject the null hypothesis)\n")
    else:
        print("The data does not appear to be normally distributed (reject the null hypothesis)\n")


1 usage
def add_normal_noise(data, std_dev, mean=0):
    """

    Parameters
    ----------
    data: our data - synthetic model
    mean: mean of the normal distribution
    std_dev: standard deviation of the normal distributio

    Returns
    -------
    noisy_data: data containing the noice
    """
    errors = np.random.normal(mean, std_dev, len(data))
    print("Statistical characteristics of the normal measurement error:")
    print_characteristics(lst=errors)
    plt.hist(errors)
    plt.title('Noise distributed according to a normal law')
    plt.show()
    noisy_data = data + errors
    print("Statistical characteristics of the model with noise:")
    print_characteristics(lst=noisy_data)
    return noisy_data
```

```python
def add_abnormal_observation(data, noisy_data, std_dev, mean=0, abnormal_measurements_amount=10,
                             abnormal_coefficient=3):
    """

    Parameters
    ----------
    data: our data - synthetic model
    noisy_data: model + noice
    std_dev: standard deviation of the normal distributio
    mean: mean
    abnormal_measurements_amount: the number of anomalies that will be created
    abnormal_coefficient: advantage ratio of anomalous measurements

    Returns
    -------
    abnormal_model_with_noise: model + noice + anomalous measurements
    random_samples: indexes of the samples where anomalous measurements were added
    """
    abnormal_model_with_noise = noisy_data
    abnormal = np.random.normal(mean, (abnormal_coefficient * std_dev), abnormal_measurements_amount)
    random_samples = random.sample(range(len(data)), k: 10)
    for i in range(abnormal_measurements_amount):
        k = random_samples[i]
        abnormal_model_with_noise[k] += abnormal[i]
    return abnormal_model_with_noise, random_samples


5 usages
def show_plot(data1, data2, title=None):
    """

    Parameters
    ----------
    data1: first dataset for visualisation
    data2: second dataset for visualisation
    title: title of the plot

    Returns
    -------
    void
    """
    plt.plot(data1)
    plt.plot(data2)
    plt.grid(True)
    plt.ylabel('BTC price in USD')
    plt.title(title)
    plt.show()
```

```python
def replace_outliers(window_size, z_threshold, model):
    """

    Parameters
    ----------
    window_size: size of the sliding window
    z_threshold: threshold for identifying outliers
    model: our data - synthetic model with noice and anomalies

    Returns
    -------
    model without noice
    """
    moving_average = np.zeros_like(model)
    moving_std_dev = np.zeros_like(model)
    for i in range(len(model)):
        start_idx = max(0, i - window_size + 1)
        end_idx = i + 1
        window_data = model[start_idx:end_idx]
        moving_average[i] = np.mean(window_data)
        moving_std_dev[i] = np.std(window_data)
    z_scores = (model - moving_average) / (moving_std_dev + np.finfo(float).eps)
    outliers = np.abs(z_scores) > z_threshold  # Identify outliers based on the Z-Score threshold
    print("Indices where anomalies have been added: ", sorted(abnormal_indices))
    print("Indices where anomalies have been replaced: ", np.where(outliers)[0])
    clear_model = model.copy()
    clear_model[outliers] = np.mean(model)
    return clear_model


1 usage
def build_regression(model):
    """

    Parameters
    ----------
    model: our data - model + noice + anomalous measurements

    Returns
    -------
    coefficients a, b and c from quadratic polynomial
    """
    indices = np.arange(len(model)) + 1
    X = np.column_stack(
        (np.ones_like(indices), indices, indices ** 2))  # Construct the design matrix (matrix of features)
    coefficients = np.linalg.lstsq(X, model, rcond=None)[0]  # Solve for the coefficients using LSM formula
    coef1, coef2, coef3 = coefficients
    equation = f"Regression Equation: y = {coef1:.2f} + {coef2:.2f}x + {coef3:.2f}x^2"
    print(equation)
    return coef1, coef2, coef3
```

```python
# removing the trend from the original data
clear_data = rates["Close*"].values - y_values_synthetic
mean_clear, median_clear, variance_clear, std_dev_clear = stat_characteristics(clear_data)
print("\nStatistical characteristics of the original data without trend:")
print_characteristics(mean_clear, median_clear, variance_clear, std_dev_clear)

# normality test for data without trend
check_normality(clear_data)
plt.hist(clear_data)
plt.title('The law of distribution of the random component of real data')
plt.show()

# adding noice to the model
model_with_noise = add_normal_noise(y_values_synthetic, std_dev_clear)
show_plot(model_with_noise, y_values_synthetic, title: 'Trend model with normal distributed error')

# adding anomalous measurements to the model
complete_model, abnormal_indices = add_abnormal_observation(y_values_synthetic, model_with_noise, std_dev_clear)
show_plot(complete_model, y_values_synthetic, title: 'Trend model with normal distributed error and abnormal observations')
print("Statistical characteristics of the data with normal distributed error and abnormal observations:")
print_characteristics(lst=complete_model)

# removing outliers from the model using sliding window
cleared_model = replace_outliers( window_size: 6, z_threshold: 1.6, complete_model)
show_plot(cleared_model, y_values_synthetic, title: 'Model cleared from outliers using sliding window and Z-Scores')
print("\nStatistical characteristics of the data cleared from outliers:")
print_characteristics(lst=cleared_model)

# building polynomial regression
a, b, c = build_regression(complete_model)
lsm_regression = a + b * (np.arange(len(complete_model))) + c * (np.arange(len(complete_model))) ** 2
show_plot(complete_model, lsm_regression, title: "Quadratic Model Fitting using LSM")
print("\nStatistical characteristics of the quadratic model fitted with LSM:")
print_characteristics(lst=lsm_regression)

# extrapolation of the model
extrapolated_x = np.arange(1.5 * len(complete_model)) + 1
extrapolated_y = a + b * extrapolated_x + c * extrapolated_x ** 2
show_plot(complete_model, extrapolated_y, title: "Extrapolated regression curve 50% ahead of the existing data")
```

IV. Висновки.

В ході лабораторної роботи було досліджено випадкову величину реальних даних, створено аналогічний шум та аномальні виміри для накладення на синтезовану у попередній лабораторній роботі модель. Було розглянуто ковзне вікно та рухому статистику для виявлено аномалії на основі Z – оцінки та проаналізовано роботу цього метода. Було синтезовано поліноміальну модель методом найменших квадратів для даних, що містять шум та аномальні виміри. Було досліджено та візуалізовано дану модель з подальшою екстраполяцією вхідних даних.

Виконав: студент Васильєв Єгор