Міністерство освіти і науки України
Національний технічний університет України «КПІ» імені Ігоря Сікорського
Кафедра обчислювальної техніки ФІОТ

ЗВІТ
з лабораторної роботи №1
з навчальної дисципліни «Data Science Technology»

Тема:

ПІДГОТОВКА ТА АНАЛІЗ ДАНИХ ДЛЯ СТАТИСТИЧНОГО НАВЧАННЯ

Виконав:
Студент 3 курсу кафедри ОТ ФІОТ,
Навчальної групи ІМ-13
Тавлуй Д. О.

Перевірив:
Професор кафедри ОТ ФІОТ
Писарчук О.О.

Київ 2023

I. Мета:

Виявити дослідити та узагальнити особливості застосування методів статистичного навчання для задач визначення статистичних характеристик вхідного потоку даних з використанням спеціалізованих пакетів мови програмування Python.

II. Завдання:

Лабораторія провідної IT-компанії реалізує масштабний проект розробки універсальної платформи з обробки Big Data масиву статистичних даних поточного спостереження для виявлення закономірностей і прогнозування розвитку контрольованого процесу. Платформа передбачає розташування back-end компоненти на власному хмарному сервері з наданням повноважень користувачам заздалегідь адаптованого frontend функціоналу універсальної платформи. Замовниками ресурсів платформи є: державні та комерційні компанії валютного трейдінгу для прогнозування динаміки зміни курсу валют та ціни інших товарів; метеорологічні служби для прогнозування параметрів метеоумов; департаменти охорони здоров'я для прогнозування зміни показників епідеміологічних ситуацій тощо. Вам, як Data Science Engineer поставлено завдання.

Завдання III рівня – максимально 9 балів.

1. Провести парсинг самостійно обраного сайту. Вміст даних, що підлягають парсингу – обрати самостійно.
2. Результати парсингу зберегти у файлі. Тип файлу обрати самостійно.
3. Оцінити динаміку тренду реальних даних.
4. Здійснити визначення статистичних характеристик результатів парсингу.
5. Синтезувати та веріфікувати модель даних, аналогічних за трендом і статистичними характеристиками реальним даним, які є результатом парсингу.
6. Провести аналіз отриманих результатів.

III. Результати виконання лабораторної роботи.

3.1. Синтезована математична модель

Пошук даних

Довго не думаючи, я вирішив обрати дані пов'язані з курсом валют, а саме ціна 31.10348 г золота у гривнях за останні 4 місяця:
https://index.minfin.com.ua/ua/exchange/nbu/bullion/xau/

Так як вартість 31.10348 г золота приблизно дорівнює 70.000 гривень, подальші результати будуть досить великими, але від того навіть цікавіше.

Визначення характеристик

Тренд

Я використав метод найменших квадратів для оцінки динаміки тренду:
Y(t) = 71364.48453635788 + -106.3126543202884 * t + 1.4803376418667489 * t^2

З цього випливає, що
a = 1.4803376418667489,
b = -106.3126543202884,
c = 71364.48453635788

Отже, враховуючи кількість k реальних вимірювань, можемо зробити модель тренду, яка буде працювати для будь-якої кількості n випадкових вимірювань:

Y (k, n, x) = c + (b / (n/k)) * x + (a / ((n/k)^2)) * x^2, де
n – кількість вимірювань
k - кількість реальних вимірювань
x – виміряне значення

Нормальний шум

Також за методом найменших квадратів визначаємо характеристики реальної вибірки даних:

Математичне очікуання = 166.40384544247354

Дисперсія = 1413136.9731218165

Середньоквадратичне відхилення = 1188.7543788023734

Використаємо визначені характеристики для моделювання нормального шуму з аномальними вимірами:

Математичне очікування = 0
СКВ = 1.7
Коефіцієнт переваги аномальних вимірів = 3
кількість АВ у відсотках = 5

Значення підібрані вручну, для отримання в результаті характеристик близьких до характеристик реальних даних.

3.2. Результати архітектурного проектування та їх опис

Я обрав модульну архітектуру проектування, так як програма стає більш зрозумілою, код багаторазовим для інших задач.

Усі виклики функцій відбуваються у файлі **main.py**
Парсинг сайту з реальними даними відбувається у файлі **data_parsing.py**
Усі математичні дії та побудова графіку відбувається у файлі **math_functions.py**
Додавання шуму та аномалій відбувається у файлі **data_manipulating.py**

3.3. Опис структури проекту програми

Усі файли підключаються у головний файл, у якому відбуваються всі виклики функцій, графіки, генерації моделі, додавання шуму та аномалій.

3.4. Результати роботи програми відповідно до завдання

Реальні дані:

Характеристики:

y(t) = 71364.48453635788 + -106.3126543202884 * t + 1.4803376418667489 * t^2

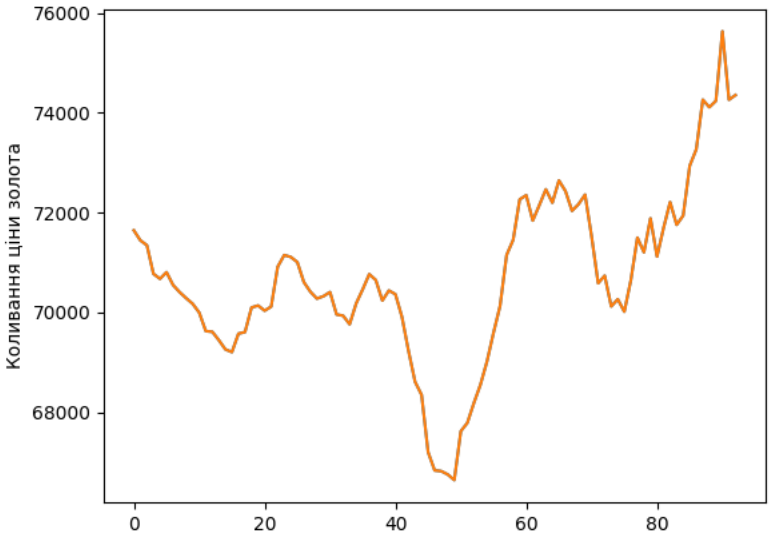
---------- Коливання ціни золота ----------

Мат. очікуання = 166.40384544247354

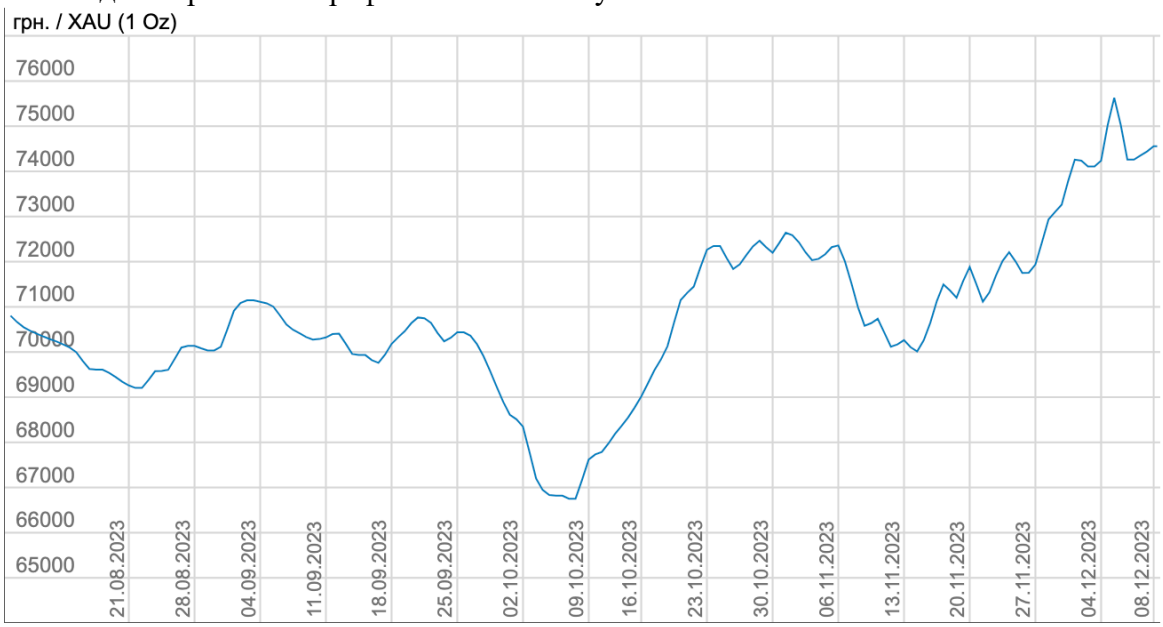
Дисперсія = 1413136.9731218165

Середньоквадратичне відхилення = 1188.7543788023734

Графік:



Також для порівняння графік з самого сайту



Можна побачити, що графіки досить схожі, відмінність лише в тому, що згенерований трохи приплюснутий.

Нормальний шум:

Характеристики:

y(t) = 4.627266888769706 + -0.008913824606809074 * t + 1.117640373631716e-06 * t^2

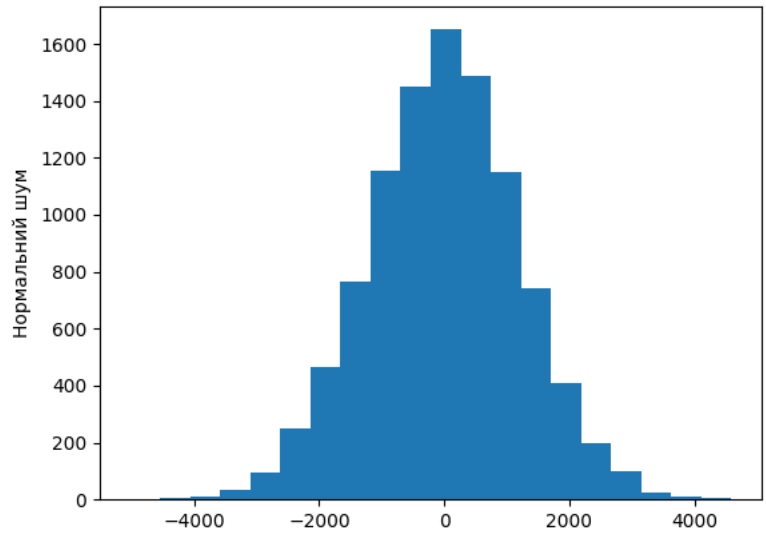
---------- Нормальний шум ----------

Мат. очікуання = 17.091574565089658

Дисперсія = 1428231.7000563443

Середньоквадратичне відхилення = 1195.0864822498597

Графік:



Модель із нормальним шумом:

Характеристики:

y(t) = 71369.11180324774 + -0.9976215097855773 * t + 0.00012915204301868573 * t^2

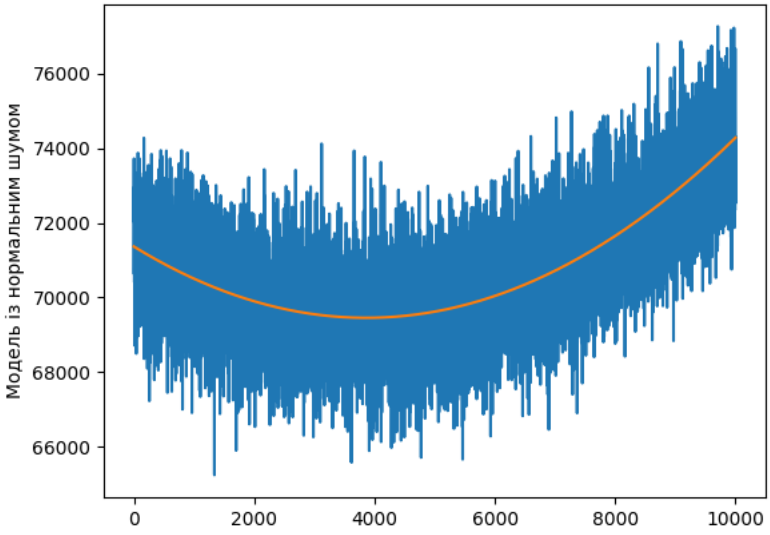
---------- Модель із нормальним шумом ----------

Мат. очікуання = 17.091574564117764

Дисперсія = 1428231.7000563438

Середньоквадратичне відхилення = 1195.0864822498595

Графік:



Модель із нормальним шумом та аномальними вимірами:

Характеристики:

y(t) = 71369.13828986994 + -0.9976332886058907 * t + 0.0001291531452106692 * t^2

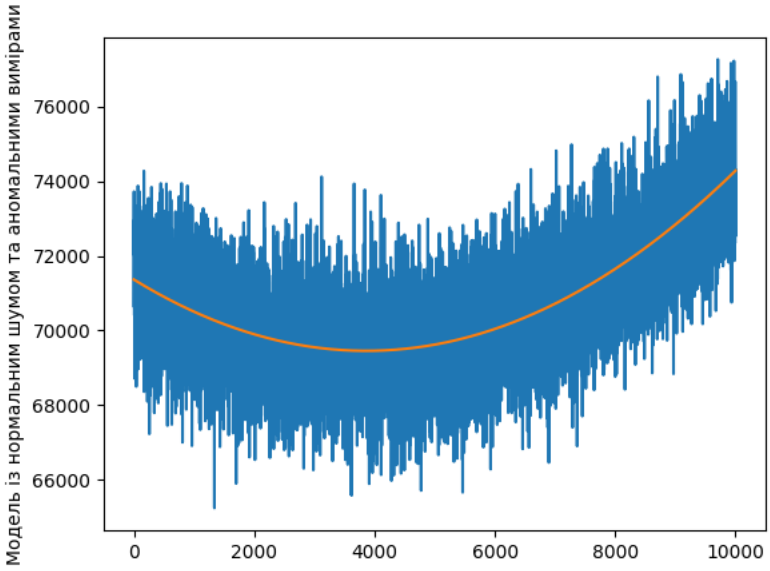
---------- Модель із нормальним шумом та аномальними вимірами ----------

Мат. очікуання = 17.042318845997215

Дисперсія = 1428260.6610713939

Середньоквадратичне відхилення = 1195.098598891068

Графік:



3.5. Програмний код, що забезпечує отримання результату

main.py

```
from data_parsing import parsing_site, file_parsing
from math_functions import Plot_AV, print_MNK_characteristics,
get_normal_model, get_square_model
from data_manipulating import add_datasets, add_anomalies
import matplotlib.pyplot as plt


if __name__ == '__main__':
    n = 10000

    print('Обрано: Парсинг табличних даних
https://index.minfin.com.ua/ua/exchange/nbu/bullion/xau/')
    url = "https://index.minfin.com.ua/ua/exchange/nbu/bullion/xau/"
    parsing_site(url)

    data_array = file_parsing(url, 'hrn-to-gold.xlsx', 'Курс (грн.)')
    meanSquare, median, var, a, b, c =
print_MNK_characteristics(data_array, 'Коливання ціни золота')
    Plot_AV(data_array, data_array, 'Коливання ціни золота')

    normal_noise = get_normal_model(0, meanSquare, n)
    print_MNK_characteristics(normal_noise, 'Нормальний шум')
    plt.hist(normal_noise, bins=20)
    plt.ylabel('Нормальний шум')
    plt.show()

    a = a/((n/len(data_array))**2)
    b = b/(n/len(data_array))
    square_model = get_square_model(n, a, b, c)
    Plot_AV(square_model, square_model, 'Квадратична модель')

    data_with_noise = add_datasets(square_model, normal_noise)
    print_MNK_characteristics(data_with_noise, 'Модель із нормальним
шумом')
    Plot_AV(square_model, data_with_noise, 'Модель із нормальним шумом')
```

```
    data_with_noise_and_anomalies = add_anomalies(data_with_noise, 0, 1.7,
5, 3)
    print_MNK_characteristics(data_with_noise_and_anomalies, 'Модель із
нормальним шумом та аномальними вимірами')
    Plot_AV(square_model, data_with_noise_and_anomalies, 'Модель із
нормальним шумом та аномальними вимірами')
```

data_parsing.py

```
import numpy as np
import pandas as pd
import re
import requests


def parsing_site(url):
    headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.0.0 Safari/537.36'}
    html_source = requests.get(url, headers=headers).text
    html_source = re.sub(r'<.*?>', lambda g: g.group(0).upper(),
html_source)
    dataframe = pd.read_html(html_source)[0]

    dataframe.to_excel("hrn-to-gold.xlsx")
    return dataframe


def file_parsing(url, file_name, data_name):
    dataframe = pd.read_excel(file_name)
    dataframe = dataframe.iloc[:-1]
    values = dataframe[data_name].values / 10000
    S_real = np.zeros(len(values))

    for i, value in enumerate(values):
        S_real[i] = value

    print(f'Джерело даних: {url}')
    return S_real
```

math_functions.py

```
import numpy as np
import matplotlib.pyplot as plt
import math as mt


def Plot_AV (S0_L, SV_L, Text):
    plt.plot(SV_L)
    plt.plot(S0_L)
    plt.ylabel(Text)
    plt.show()
    return


def MNK (S0):
    iter = len(S0)
    Yout = np.zeros((iter, 1))
    F = np.ones((iter, 3))
    for i in range(iter):
        Yout[i, 0] = float(S0[i])
        F[i, 1] = float(i)
```

```
        F[i, 2] = float(i * i)
    FT = F.T
    FFT = FT.dot(F)
    FFTI = np.linalg.inv(FFT)
    FFTIFT = FFTI.dot(FT)
    C = FFTIFT.dot(Yout)
    y_output = F.dot(C)
    a, b, c = C[2,0], C[1,0], C[0,0]
    print('Регресійна модель:')
    print(f'y(t) =  {C[0,0]}  + {C[1,0]}  * t  +  {C[2,0]}  * t^2')
    return y_output, a, b, c


def print_MNK_characteristics(data, title):
    num_iterations = len(data)
    y_output, a, b, c = MNK(data)
    result_data = np.zeros((num_iterations))
    for i in range(num_iterations):
        result_data[i] = data[i] - y_output[i, 0]

    median = np.median(result_data)
    var = np.var(result_data)
    meanSquare = mt.sqrt(var)

    print(f'------------ {title} -------------')
    print(f'Мат. очікуання = {median}')
    print(f'Дисперсія = {var}')
    print(f'Середньоквадратичне відхилення = {meanSquare}')

    return meanSquare, median, var, a, b, c

def get_normal_model(loc, scale, n):
    return np.random.normal(loc, scale, n)


def get_square_model(n, a, b, c):
    S = np.zeros((n))

    koeffs = [c, b, a]
    for i in range(n):
        for index, koef in enumerate(koeffs):
            S[i] += koef*(i**index)
    return S
```

data_manipulating.py

```
import numpy as np


def add_datasets(data_one, data_two):
    if len(data_one) != len(data_two):
        raise ValueError('Datasets must have the same length')

    return np.array(data_one) + np.array(data_two)


def add_anomalies(data, loc, scale, percentage, q):
    result = np.copy(data)
    n = len(data)
    num_anomalies = int(n * (percentage / 100))

    indexes = np.random.choice(np.arange(n), size=num_anomalies,
```

```
replace=False)
    anomalies = np.random.normal(loc, q * scale, num_anomalies)

    result[indexes] += anomalies

    return result
```

Висновки

Характеристики зібраних даних та створеної моделі дуже схожі, що свідчить про успішність генерації. Це відображено на графіку, де тренд згенерованої моделі відповідає тренду реальних даних. Хоча більша вибірка дозволила б досягти більшої точності, похибка залишається прийнятною, підтверджуючи правильність обраної моделі. Крім того, в процесі роботи були отримані навички у парсингу сайту для збору даних та визначення характеристик вибірки, що в подальшому використовувалось для успішної генерації моделі.