**Міністерство освіти і науки України**

**Національний технічний університет України «КПІ» імені Ігоря Сікорського**

**Кафедра обчислювальної техніки ФІОТ**

**ЗВІТ**

**з лабораторної роботи No1**

**з навчальної дисципліни «Технології Data Science»**

Тема:

ПІДГОТОВКА ТА АНАЛІЗ ДАНИХ ДЛЯ СТАТИСТИЧНОГО НАВЧАННЯ

**Виконла:**

Студентка 4 курсу кафедри ІПІ ФІОТ,

Навчальної групи ІП-12

Луговець Тетяна

**Перевірив:**

Професор кафедри ОТ ФІОТ

Баран Д. Р.

**Київ 2024**

**Мета роботи:**

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

**Завдання:**

Лабораторія провідної ІТ-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 front-end 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.

**Завдання ІІІ рівня.** Вам, як Data Science Engineer поставлено завдання:

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 верифікувати модель даних, аналогічних за трендом і статистичними характеристиками реальним даним, які є результатом парсингу.
6. Провести аналіз отриманих результатів.

**Результати виконання лабораторної роботи**

Синтезована математична модель

 П**оліноміальна регресія 12-го ступеня**:

Ключовою частиною синтезованої моделі є поліноміальна регресія 12-го ступеня. Для побудови моделі було використано функцію np.polyfit(), яка визначає коефіцієнти полінома, що найкраще описує тенденцію змін реальних даних інфляції. Високий ступінь полінома (12) дозволяє врахувати більш складні коливання та тенденції у часі. Модель поліноміальної регресії:yi=β0+β1xi+β2xi2+⋯+βmxim+εi (i=1,2,…,n)



 **Шум, доданий до тренду**:

До синтезованих значень інфляції додається випадковий шум з нормальним розподілом (np.random.normal()) із стандартним відхиленням 1 (noise\_std = 1). Це дозволяє моделі імітувати реальні дані, які, як правило, містять деякі випадкові відхилення від загального тренду.

 **Параметри статистичного аналізу**:

Після генерації синтезованих даних, код обчислює різні статистичні показники (середнє значення, стандартне відхилення, медіана, мінімальне та максимальне значення, дисперсія) для обох наборів даних — реальних і синтезованих. Це дозволяє порівняти основні характеристики синтезованих даних із реальними даними.

**Блок схема алгоритму та її опис**

**A close up of a label

Description automatically generated**

Рис.1. Блок-схема алгоритму програми

**Опис алгоритму:**

 **Початок**:

* Програма ініціюється та починається виконання основного алгоритму.

 **Надіслати HTTP-запит**:

* Використання бібліотеки requests для надсилання HTTP-запиту на URL https://services.dtkt.ua/catalogues/indexes/3.
* Запит отримує HTML-код веб-сторінки, який містить необхідні дані про інфляцію.

 **Парсинг HTML-коду**:

* Використання бібліотеки BeautifulSoup для аналізу HTML-коду сторінки.
* Пошук таблиці з класом 'in\_text\_tab table table-bordered table-services', яка містить потрібну інформацію.
* Знаходження першого рядка таблиці для отримання років (<tr> з заголовками).
* Знаходження останнього рядка таблиці для отримання значень інфляції.

 **Отримання даних**:

* **Роки (Years)**:
  + Витягуються значення років з першого рядка таблиці, ігноруючи перший та останній стовпці.
  + Значення конвертуються у цілі числа (int) та зберігаються у списку year\_values.
* **Значення інфляції (Inflation Values)**:
  + Витягуються значення інфляції з останнього рядка таблиці, ігноруючи перший стовпець.
  + Значення обробляються для заміни ком (,) на крапки (.) і конвертуються в числа з плаваючою комою (float).
  + Значення зберігаються у списку value\_values.

 **Збереження даних у JSON-файл**:

* Створення JSON-об'єкта data, який містить два ключі: "year" та "value", зі списками year\_values та value\_values.
* Використання модуля json для запису цього об'єкта у файл 'data.json'.

 **Аналіз реальних даних інфляції**:

* Виклик функції analyze\_real\_data(), яка приймає parsed\_dates та parsed\_inflation\_values як аргументи.
* Внутрішні кроки:
  + Обчислення основних статистичних показників для реальних даних інфляції:
    - **Середнє значення (Mean)**: np.mean(real\_inflation\_values).
    - **Стандартне відхилення (Standard Deviation)**: np.std(real\_inflation\_values).
    - **Медіана (Median)**: np.median(real\_inflation\_values).
    - **Мінімальне значення (Min)**: np.min(real\_inflation\_values).
    - **Максимальне значення (Max)**: np.max(real\_inflation\_values).
    - **Дисперсія (Variance)**: np.var(real\_inflation\_values).
  + Виведення обчислених показників на екран.
  + Побудова графіку реальних даних інфляції (дата на осі X, інфляція на осі Y) за допомогою matplotlib.

 **Генерація синтетичних даних**:

* Виклик функції generate\_and\_analyze\_synthetic\_data(), яка приймає parsed\_dates та parsed\_inflation\_values як аргументи.
* Внутрішні кроки:
  + **Побудова моделі поліноміальної регресії**:
    - Використання np.polyfit() для визначення коефіцієнтів полінома 12-го ступеня, який найкраще підходить до реальних даних інфляції.
    - Використання отриманих коефіцієнтів для обчислення значень синтетичного тренду за допомогою np.polyval().
  + **Додавання шуму до синтетичного тренду**:
    - Генерація випадкового шуму з нормальним розподілом за допомогою np.random.normal().
    - Додавання цього шуму до синтетичного тренду для отримання остаточних синтетичних значень інфляції.
  + **Аналіз синтетичних даних**:
    - Обчислення статистичних показників для синтетичних даних (аналогічно до реальних даних).
    - Виведення обчислених показників на екран.

 **Побудова графіків**:

* **Графік реальних даних інфляції**:
  + Побудова графіку з використанням реальних даних інфляції.
* **Графік синтетичних даних та тренду**:
  + Побудова графіку, який включає як синтетичні дані, так і синтетичний тренд.

 **Кінець**:

* Програма завершує свою роботу після закриття вікна GUI.

**Опис структури проекту програми в середовищі VScode**

Для реалізації розробленого алгоритму мовою програмування Python з використанням можливостей інтегрованого середовища VScode сформовано проект. Проект базується на лінійній бізнес-логіці функціонального програмування та має таку структуру.

A screenshot of a computer program

Description automatically generated

Рис.2. Структура проекту

analytics.py – файл програмного коду з аналітикою для лабораторної роботи;

parse\_website.py – файл програмного коду з парсингом для лабораторної роботи;

data.json – файл для збереження даних.

**Результати роботи програми відповідно до завдання.**

Результатом роботи програми є

Діаграми: Програма генерує різні діаграми, які візуалізують дані.

Числові характеристики: Програма розраховує числові характеристики для вибірок, такі як математичне сподівання (середнє значення), дисперсія (середньоквадратичне відхилення) та інші. Ці характеристики надають кількісні оцінки розподілу даних.

Вивід результатів: Результати виводяться на екран у вигляді текстових повідомлень, діаграм і графіків. Вони можуть бути використані для аналізу та порівняння різних даних та моделей.

Графік та статистика даних з парсингу рис. 3-4

A screenshot of a graph

Description automatically generated

Рис. 3 Графік з парсигу

**A computer screen shot of a black screen

Description automatically generated**

Рис. 4 Статистика даних з парсингу

Синтезовані графік та статистика даних рис. 5-6

A screenshot of a graph

Description automatically generated

Рис. 5 Графік інфляції синтезований

**A computer screen shot of a black screen

Description automatically generated**

Рис. 6 Статистика синтезованих даних

**Програмний код.**

Програмний код послідовно реалізує алгоритм рис.1 та спрямовано на отримання результатів, поданих вище. При цьому використано можливості Python бібліотек: pip; numpy; matplotlib.

Контексні коментарі пояснюють сутність окремих скриптів наведеного коду програми.

Файл analytics.py

'''

Виконала Луговець Тетяна ІП-12

Завдання ІІІ 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 верифікувати модель даних, аналогічних за трендом і

статистичними характеристиками реальним даним, які є результатом парсингу.

6. Провести аналіз отриманих результатів.

'''

import numpy as np

from matplotlib.backends.backend\_tkagg import FigureCanvasTkAgg

import matplotlib.pyplot as plt

import tkinter as tk

import json

# Функція analyze\_real\_data:

# Приймає списки dates (дати) та real\_inflation\_values (реальні значення інфляції).

# Обчислює та виводить статистичні характеристики реальних даних: середнє, стандартне відхилення, медіану, мінімум, максимум і дисперсію.

# Будує графік реальних даних інфляції за допомогою matplotlib.

def analyze\_real\_data(dates, real\_inflation\_values):

    real\_mean\_inflation = np.mean(real\_inflation\_values)

    real\_std\_deviation\_inflation = np.std(real\_inflation\_values)

    real\_median\_inflation = np.median(real\_inflation\_values)

    real\_min\_inflation = np.min(real\_inflation\_values)

    real\_max\_inflation = np.max(real\_inflation\_values)

    real\_variance\_inflation = np.var(real\_inflation\_values)

    print("Статистика даних з парсингу:")

    print("Середнє значення інфляції:", real\_mean\_inflation)

    print("Стандартне відхилення інфляції:", real\_std\_deviation\_inflation)

    print("Медіанна інфляція:", real\_median\_inflation)

    print("Мінімальне значення інфляції:", real\_min\_inflation)

    print("Максимальне значення інфляції:", real\_max\_inflation)

    print("Відхилення інфляції:", real\_variance\_inflation)

    fig, ax = plt.subplots(figsize=(8, 4))

    ax.plot(dates, real\_inflation\_values, marker='o', linestyle='-', color='black', markersize=4)

    ax.set\_xlabel("Date")

    ax.set\_ylabel("Inflation")

    plt.xticks(rotation=45)

    ax.grid(True)

# Функція generate\_and\_analyze\_synthetic\_data:

# Приймає списки dates (дати) та real\_inflation\_values (реальні значення інфляції).

# Використовує метод numpy.polyfit для побудови поліноміальної моделі (тренду) для реальних даних інфляції ступеня degree (в нашому випадку 12).

# Генерує синтетичні дані інфляції, додаючи до тренду випадковий шум, створений функцією numpy.random.normal.

# Обчислює статистичні характеристики синтетичних даних: середнє, стандартне відхилення, медіану, мінімум, максимум і дисперсію.

# Повертає синтетичні трендові значення та синтетичні значення інфляції.

def generate\_and\_analyze\_synthetic\_data(dates, real\_inflation\_values):

    degree = 12

    noise\_std = 1

    np.random.seed(0)

    coefficients = np.polyfit(np.arange(len(dates)), real\_inflation\_values, degree)

    synthetic\_trend\_values = np.polyval(coefficients, np.arange(len(dates)))

    print("\nМодель:")

    print(np.poly1d(coefficients))

    synthetic\_inflation\_values = synthetic\_trend\_values + np.random.normal(0, noise\_std, len(dates))

    synthetic\_mean\_inflation = np.mean(synthetic\_inflation\_values)

    synthetic\_std\_deviation\_inflation = np.std(synthetic\_inflation\_values)

    synthetic\_median\_inflation = np.median(synthetic\_inflation\_values)

    synthetic\_min\_inflation = np.min(synthetic\_inflation\_values)

    synthetic\_max\_inflation = np.max(synthetic\_inflation\_values)

    synthetic\_variance\_inflation = np.var(synthetic\_inflation\_values)

    print("\nСтатистика синтезованих даних:")

    print("Середнє значення інфляції:", synthetic\_mean\_inflation)

    print("Стандартне відхилення інфляції:", synthetic\_std\_deviation\_inflation)

    print("Медіана інфляції:", synthetic\_median\_inflation)

    print("Мінімальне значення інфляції:", synthetic\_min\_inflation)

    print("Максимальне значення інфляції:", synthetic\_max\_inflation)

    print("Відхилення інфляції:", synthetic\_variance\_inflation)

    return synthetic\_trend\_values, synthetic\_inflation\_values

#Головний блок програми (if \_\_name\_\_ == '\_\_main\_\_'):

#Завантажує дані з файлу data.json, який був згенерований на попередньому етапі.

#Розпаковує роки (parsed\_dates) та значення інфляції (parsed\_inflation\_values) для подальшого аналізу

#Виводить графіки

if \_\_name\_\_ == '\_\_main\_\_':

    with open('data.json', 'r') as f1:

        data = json.load(f1)

    parsed\_dates = data['year']

    parsed\_inflation\_values = data['value']

    root = tk.Tk()

    root.title("Діаграма інфляції гривні із трендом")

    frame1 = tk.Frame(root)

    frame1.pack(side=tk.LEFT)

    analyze\_real\_data(dates=parsed\_dates, real\_inflation\_values=parsed\_inflation\_values)

    fig1, ax1 = plt.subplots(figsize=(8, 4))

    ax1.plot(parsed\_dates, parsed\_inflation\_values, marker='o', linestyle='-', label='Дані парсингу')

    ax1.set\_xlabel("Дата")

    ax1.set\_ylabel("Інфляція")

    plt.xticks(rotation=45)

    ax1.grid(True)

    plt.legend()

    plt.title("Графік інфляції гривні (Парсинг)")

    canvas1 = FigureCanvasTkAgg(fig1, master=frame1)

    canvas\_widget1 = canvas1.get\_tk\_widget()

    canvas\_widget1.pack()

    frame2 = tk.Frame(root)

    frame2.pack(side=tk.RIGHT)

    synthetic\_trend\_values, synthetic\_inflation\_values = generate\_and\_analyze\_synthetic\_data(

        dates=parsed\_dates,

        real\_inflation\_values=parsed\_inflation\_values)

    fig2, ax2 = plt.subplots(figsize=(8, 4))

    ax2.plot(parsed\_dates, synthetic\_inflation\_values, marker='o', label='Синтезовані дані')

    ax2.plot(parsed\_dates, synthetic\_trend\_values, label='Синтезований тренд')

    ax2.set\_xlabel("Дата")

    ax2.set\_ylabel("Інфляція")

    plt.xticks(rotation=45)

    ax2.grid(True)

    plt.legend()

    plt.title("Графік інфляції гривні (Синтезований)")

    canvas2 = FigureCanvasTkAgg(fig2, master=frame2)

    canvas\_widget2 = canvas2.get\_tk\_widget()

    canvas\_widget2.pack()

    root.mainloop()

Файл parse\_website.py

from bs4 import BeautifulSoup

import requests

import json

url = "https://services.dtkt.ua/catalogues/indexes/3"

page = requests.get(url)

soup = BeautifulSoup(page.text, 'html.parser')

table = soup.find('table', class\_='in\_text\_tab table table-bordered table-services')

year\_row = table.find\_all('tr')[0]

year\_values = [int(cell.text.strip()) for cell in year\_row.find\_all('td')[1:-1] if cell.text.strip()]

value\_row = table.find\_all('tr')[-1]

value\_values = [float(cell.text.replace(',', '.').strip()) for cell in value\_row.find\_all('td')[1:] if

                cell.text.strip()]

data = {

    "year": year\_values,

    "value": value\_values

}

with open('data.json', 'w') as f1:

    json.dump(data, f1)

**Аналіз результатів відлагодження та верифікації результатів роботи програми.**

Результати відлагодження та тестування довели працездатність розробленого коду. Це підтверджується результатами розрахунків, які не суперечать теоретичним положенням. Верифікація функціоналу програмного коду, порівняння отриманих результатів з технічними умовами завдання на лабораторну роботу доводять, що усі завдання виконані у повному обсязі.

**IV. Висновки.**

В результаті виконаної лабораторної роботи був розроблений програмний скрипт мовою Python який забезпечує аналіз різних характеристик та властивостей вибірок даних, включаючи синтез випадкових величин, моделювання тренду та визначення статистичних параметрів. Синтезовані дані є корисними для тестування і верифікації алгоритмів їх обробки і будуть використані в подальших дослідженнях часових рядів.

Виконав: студент Луговець Тетяна