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

ЗВІТ з лабораторної роботи №4 з навчальної дисципліни «Технології Computer Vision»

Тема: РЕАЛІЗАЦІЯ ПРОЦЕСІВ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ: МІНІ ПРОЕКТИ В ГАЛУЗЯХ OLAP, Data Mining, Text Mining, Voice Recognition

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 інтелектуального аналізу даних та технологій OLAP, Data Mining, Text Mining, Voice Recognition.

ІІ. Завдання:

Лабораторія провідної IT-компанії реалізує проект із розробки ERP системи для автоматизації процесів управління у сфері завдань електронної комерції. Вам, як Data Science Engineer поставлене наступне завдання. Розробити програмний скрипт мовою Python що реалізує обчислювальний алгоритм ERP-системи із впровадженням технології інтелектуального аналізу даних відповідно до технічних умов:
ІІ рівень складності 9 балів, викладених у табл.2 – R&D міні проект.

3	Розробити програмний скрипт, що реалізує прогнозування зміни температури повітря в обраному регіоні на тиждень за даними Інтернет джерел. Передбачити OLAP – візуалізацію результатів.

ІІІ. Результати виконання лабораторної роботи.

3.1. Синтезована математична модель;

Для прогнозування температури на тиждень по даним за попередній місяць, будемо використовувати декілька моделей: експоненціальне згладжування, авторегресійне інтегроване ковзне середнє (ARIMA), та метод опорних векторів. Для фінального прогнозу братимемо усереднене значення передбачень чотирьох моделей.

Експоненційне згладжування — це метод математичного перетворення, який застосовується при прогнозуванні часових рядів. Сутність цього методу полягає в тому, що кожен елемент (рівень) часового ряду згладжується за допомогою зваженої плинної середньої, причому вага її зменшується по мірі віддалення від кінця ряду.

Авторегресійне інтегроване ковзне середнє (ARIMA) - це статистичний метод для аналізу та прогнозу часових рядів даних, який включає в себе три основні компоненти:

- Авторегресія (AR) - це модель, де поточне значення часового ряду залежить від попередніх значень цього ж ряду.
- Інтеграція (I) - ця компонента вказує на кількість диференціацій (різниць) даних, які необхідно зробити, щоб зробити ряд стаціонарним.
- Ковзне середнє (MA) - ця компонента враховує ковзне середнє (середнє значення) попередніх значень помилок моделі.

Для візуалізації результатів будемо використовувати OLAP (Online Analytical Processing): метод аналізу даних, який дозволяє інтерактивно взаємодіяти з багатовимірними даними, щоб отримати інсайти та побудувати звіти. OLAP візуалізація надає можливість відобразити дані у вигляді графіків, діаграм, таблиць та інших візуальних елементів.

3.2. Результати архітектурного проектування та їх опис;

Після парсингу даних сайту було отримано дані про максимальну та мінімальну температуру за день з 1 по 24 жовтня. Після обробки цих даних, було навчено моделі та пораховано відповідні середньоквадратичні помилки: для SVM моделі за допомогою випадкового розбиття на тестові та тренувальні дані, для всіх інших тренувальними даними були перші 20 днів жовтня, а тестовими (для обрахунку помилки) – останні 4. Таким чином кількість тестових та тренувальних даних для всіх моделей була однаковою, отже значення помилки могло давати приблизне уявлення про якість моделей.

Отримавши натреновані моделі, було здійснено візуалізацію результатів, спочатку на звичайному 2D графіку, а потім з використанням OLAP.

3.3. Опис структури проекту програми;

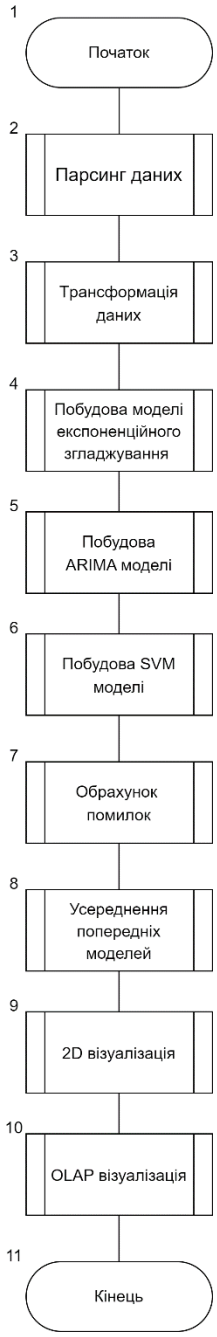

Рис.1. Блок схема алгоритму програми.

3.4. Результати роботи програми відповідно до завдання (допускається у формі скриншотів);

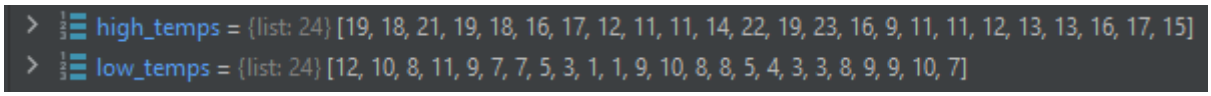

Рис.2. Отримані в результаті парсингу дані.

```
Exponential smoothing:
predicted: 15.50, expected: 7.00, difference: -54.84%
predicted: 15.75, expected: 7.50, difference: -52.37%
predicted: 12.32, expected: 15.50, difference: 25.82%
predicted: 13.55, expected: 7.50, difference: -44.66%


Autoregressive integrated moving average:
predicted: 11.25, expected: 7.00, difference: -37.78%
predicted: 15.50, expected: 7.50, difference: -51.61%
predicted: 13.78, expected: 15.50, difference: 12.48%
predicted: 14.71, expected: 7.50, difference: -49.02%


Support vector machine:
predicted: 11.29, expected: 7.00, difference: -38.01%
predicted: 11.21, expected: 7.50, difference: -33.09%
predicted: 14.72, expected: 15.50, difference: 5.31%
predicted: 10.71, expected: 7.50, difference: -29.94%


Mean squared error for exponential smoothing: 3.319
Mean squared error for autoregressive integrated moving average: 1.830
Mean squared error for support vector regression: 10.766
```

Рис.3. Оцінка натренованих моделей.

```
Final weather forecast for the week:
 [10.36783178 10.91071007 10.59313204 10.87260042 10.98894854 10.76310326
 11.85521176]
```

Рис.4. Прогнозування температури на основі передбачень трьох моделей.

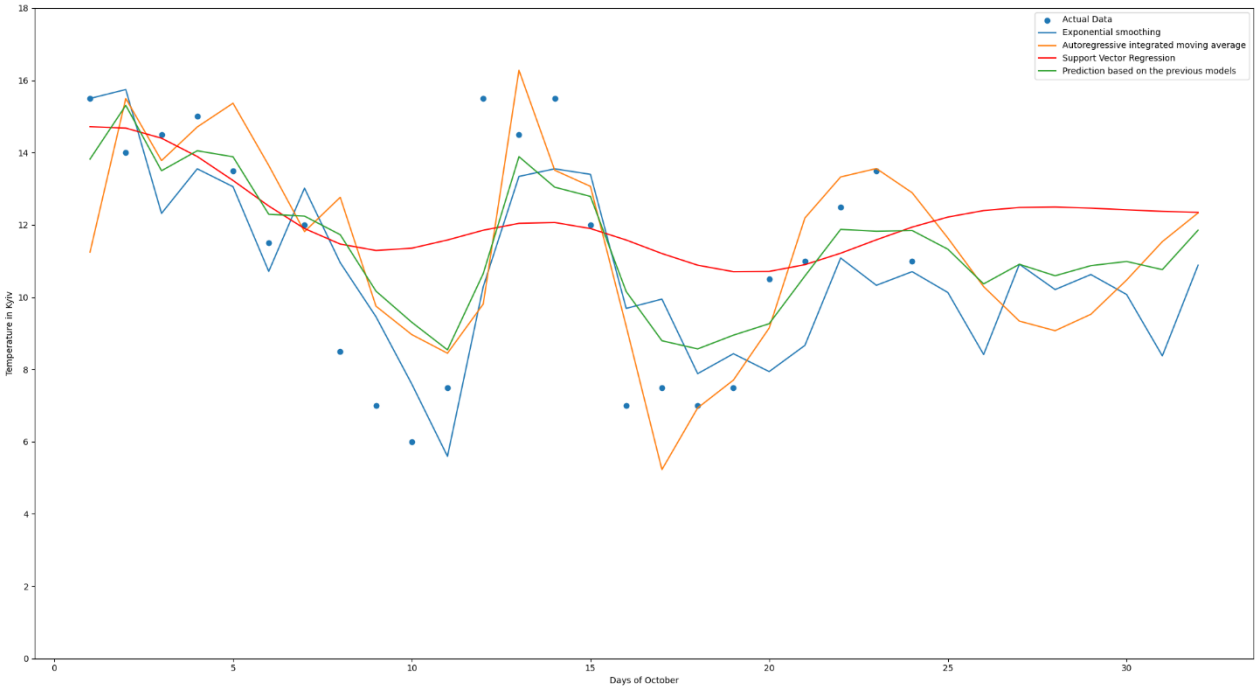

Рис.5. Візуалізація результатів.

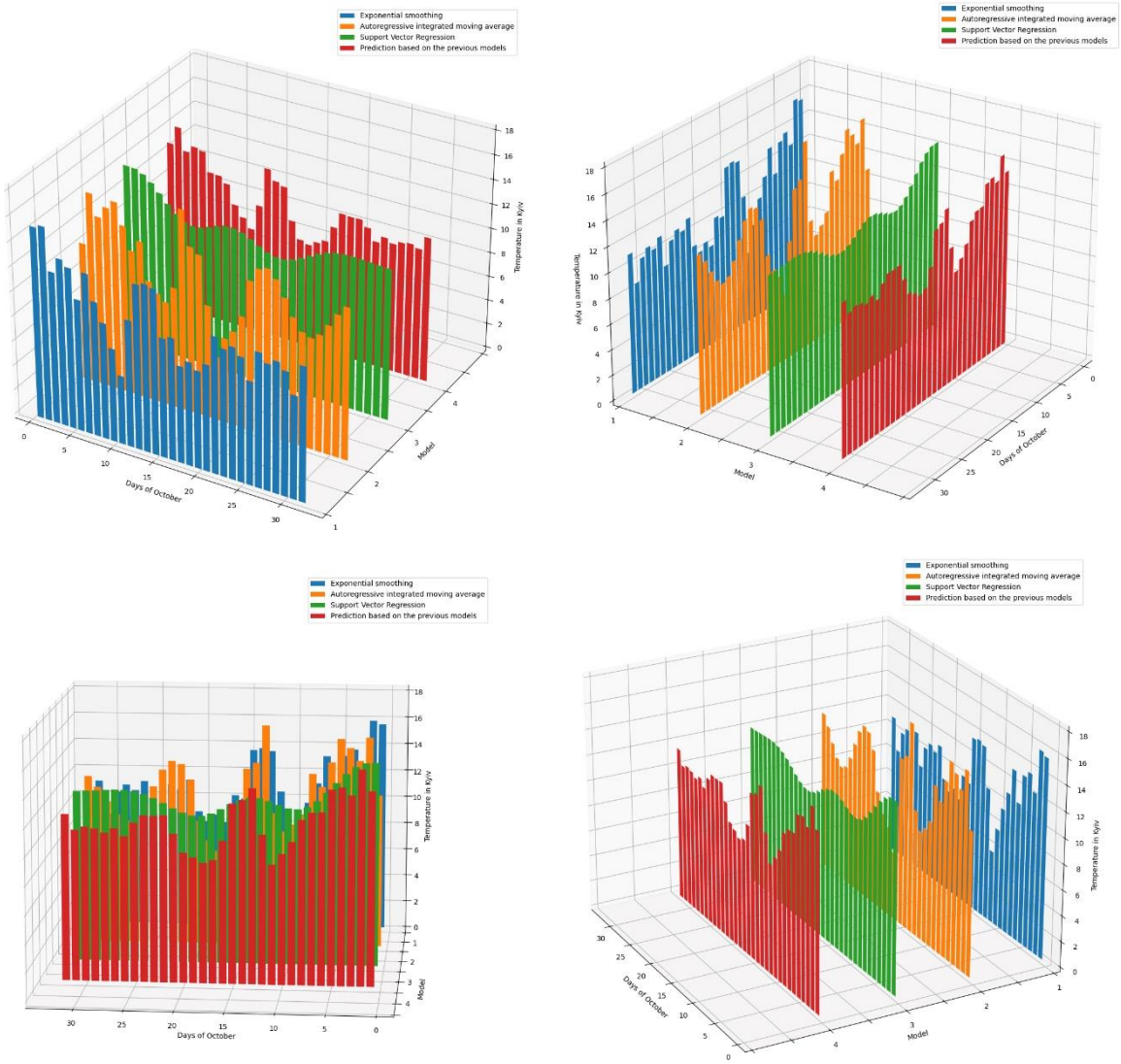

Рис.6. Візуалізація результатів (OLAP).

3.5. Програмний код, що забезпечує отримання результату (допускається у формі скриншотів).

```
import ...

today = 25  # datetime.today().day
warnings.filterwarnings("ignore")

# constants for accessing and parsing accuweather.com
url = "https://www.accuweather.com/ru/ua/kyiv/324505/october-weather/324505?year=2023"
user_agent = "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/104.0.5112.79 Safari/537.36"
header = {"User-Agent": user_agent}


# noinspection PyTypeChecker
def parsing(URL, headers):
    """

    Parameters
    ----------
    URL: URL of the website
    headers: headers for request

    Returns
    -------
    high_temperatures, low_temperatures: maximum and minimum temperatures for the day
    """
    response = requests.get(URL, headers=headers)
    soup = bs(response.content,  features: "html.parser")
    high_temperatures = soup.find_all( name: "div", class_="high")[:today - 1]
    low_temperatures = soup.find_all( name: "div", class_="low")[:today - 1]

    for i in range(today - 1):
        high_temperatures[i] = int(high_temperatures[i].get_text()[7:-7])
        low_temperatures[i] = int(low_temperatures[i].get_text()[7:-7])
    return high_temperatures, low_temperatures


# original data transformation
high_temps, low_temps = parsing(url, header)
dates = range(1, today + 8)
X = np.arange(1, today).reshape(-1, 1)
y = ((np.array(high_temps) + np.array(low_temps)) / 2).reshape(-1, 1)
X_train, X_test, y_train, y_test = train_test_split( *arrays: X, y, test_size=0.15, random_state=42)

# exponential smoothing model
exponential_smoothing = ExponentialSmoothing(y[:-4], trend='mul', seasonal='add', seasonal_periods=5)
exponential_smoothing_fit = exponential_smoothing.fit()
exp_predicted = exponential_smoothing_fit.predict(start=0, end=today + 6)
exp_mse = mean_squared_error(y[-4:], exp_predicted[today - 4:today])
print("Exponential smoothing:")
for p, e in zip(exp_predicted, y_test):
    e = float(e[0])
    print(f'predicted: {p:.2f}, expected: {e:.2f}, difference: {(e - p) / p * 100:.2f}%')
print('\n')
```

```
# ARIMA model
arima = sm.tsa.ARIMA(y[:-4], order=(2, 1, 2))
arima_fit = arima.fit()
arima_predicted = arima_fit.predict(start=0, end=today + 6)
arima_predicted[0] = np.mean(y)
arima_mse = mean_squared_error(y[-4:], arima_predicted[today - 4:today])
print("Autoregressive integrated moving average:")
for p, e in zip(arima_predicted, y_test):
    e = float(e[0])
    print(f'predicted: {p:.2f}, expected: {e:.2f}, difference: {(e - p) / (p) * 100:.2f}%')
print('\n')

# SVM model
svm = SVR(C=5)
svm.fit(X_train, y_train)
svm_predicted = svm.predict(X_test)
svm_forecast = svm.predict(np.arange(1, today + 8).reshape(-1, 1))
svm_mse = mean_squared_error(y_test, svm_predicted)
print("Support vector machine:")
for p, e in zip(svm_predicted, y_test):
    e = float(e[0])
    print(f'predicted: {p:.2f}, expected: {e:.2f}, difference: {(e - p) / p * 100:.2f}%')
print('\n')

print(f'Mean squared error for exponential smoothing: {exp_mse:.3f}')
print(f'Mean squared error for autoregressive integrated moving average: {arima_mse:.3f}')
print(f'Mean squared error for support vector regression: {svm_mse:.3f}')

average_prediction = ((exp_predicted + arima_predicted + svm_forecast) / 3)  # average prediction of three models
print("Final weather forecast for the week:\n", average_prediction[-7:])

# 2D visualisation
plt.scatter(X, y, label='Actual Data')
plt.plot( *args: dates, exp_predicted, label='Exponential smoothing')
plt.plot( *args: dates, arima_predicted, label='Autoregressive integrated moving average')
plt.plot( *args: dates, svm_forecast, color='red', label='Support Vector Regression')
plt.plot( *args: dates, average_prediction, label='Prediction based on the previous models')
plt.xlabel('Days of October')
plt.ylabel('Temperature in Kyiv')
plt.ylim( *args: 0, 18)
plt.legend()
plt.show()

fig = plt.figure(figsize=(10, 6))
ax = fig.add_subplot(111, projection='3d')

# Plot each 2D plot as a separate layer in 3D
ax.bar(dates, exp_predicted, zdir='y', zs=0, label='Exponential smoothing')
ax.bar(dates, arima_predicted, zdir='y', zs=1, label='Autoregressive integrated moving average')
ax.bar(dates, svm_forecast, zdir='y', zs=2, label='Support Vector Regression')
ax.bar(dates, average_prediction, zdir='y', zs=3, label='Prediction based on the previous models')

ax.set_xlabel('Days of October')
ax.set_zlabel('Temperature in Kyiv')
ax.set_ylabel('Model')
ax.set_ylim(0, 4)
ax.set_yticklabels(['1', '2', '3', '4'])
ax.set_zlim(0, 18)
# ax.view_init(100, -90)
plt.legend()
plt.show()
```

IV. Висновки.

В ході лабораторної роботи було розроблено програмний скрипт, що реалізує прогнозування зміни температури повітря в Києві на тиждень за даними Інтернет джерела та передбачено OLAP – візуалізацію.

Отримані середньоквадратичні похибки для моделей можуть давати лише узагальнене представлення про якість моделі, оскільки неточне прогнозування у довільні дні (чи останні 4) не означає, що модель неточно передбачить температуру на тиждень вперед, але якщо орієнтуватись на них, то ARIMA модель вийшла найкращою на тестових даних.

Вибір методу візуалізації є важливим для ефективного розуміння та представлення даних. Доцільність вибору методу візуалізації залежить від конкретних завдань та цілей аналізу. Для даних про зміну температури цілком достатньо 2D графіку, та OLAP візуалізаціє не дає багато додаткової інформації, однак навички побудови OLAP інтерпретації є важливими для аналізу та розуміння багатовимірних даних. Ця компетенція дозволяє аналізувати великі обсяги інформації та виділяти в них важливі зв'язки і залежності, та неодмінно знадобиться в майбутньому.

Порівнюючи передбачення моделей з прогнозом погоди відповідного сайту, бачимо, що метод опорних векторів для регресії виявися найточнішим, а усереднене передбачення зайняло друге місце. В цілому моделі працюють не надто точно, але зважаючи не те, що ми не досліджували жодні чинники, які насправді впливають на температуру повітря, результат є прийнятним.

	25.10	26.10	27.10	28.10	29.10	30.10	31.10	**MSE**
Середнє передбачення	10,36	10,91	10,59	10,87	10,98	10,76	11,85	**7,289029**
Експоненціальне згладжування	8,41	10,91	10,21	10,62	10,07	8,37	10,88	**11,5887**
Метод опорних векторів	12,39	12,48	12,49	12,46	12,41	12,37	12,34	**6,562114**
Модель ARIMA	10,29	9,33	9,07	9,52	10,47	11,53	12,33	**8,992714**
Передбачення з сайту прогнозу погоди	**8**	**13**	**14**	**9,5**	**11,5**	**14,5**	**15,5**	**0**

Виконав: Васильєв Єгор