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

ЗВІТ
з лабораторної роботи №2
з навчальної дисципліни «Технології Data Science»

Тема:

СТАТИСТИЧНЕ НАВЧАННЯ З ПОЛІНОМІАЛЬНОЮ РЕГРЕСІЄЮ

Виконав:
Студент 4 курсу кафедри ФІОТ,
Навчальної групи ІП-11
Олександр Головня

Перевірив:
Професор кафедри ОТ ФІОТ
Олексій Писарчук

Київ 2024

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:

Реалізація проекту триває та спрямовано на збільшення функціональності програмної компоненти

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 front-end функціоналу універсальної платформи.

Завдання ІІІ рівня, підвищеної складності – максимально 15 балів.

Реалізувати групу вимог 1 та (або) 2 з імплементацією однієї з групи вимог 3. Докладно опитати отримані R&D рішення.

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.

Група вимог_3:

Реалізувати R&D процеси для етапів статистичного навчання.

2.1. Здійснити розробку власного алгоритму виявлення аномальних вимірів та / або «навчання» параметрів відомих алгоритмів «бачити» властивості статистичної вибірки.

ІІІ. Результати виконання лабораторної роботи.

3.4.Блок схема алгоритму.

Рис.1 – Блок-схема алгоритму програми

3.1. Опис структури проекту програми.

Для реалізації розробленого алгоритму мовою програмування Python з використанням можливостей інтегрованого середовища сформовано проект.

Проект базується на лінійній бізнес-логіці функціонального програмування та має таку структуру.

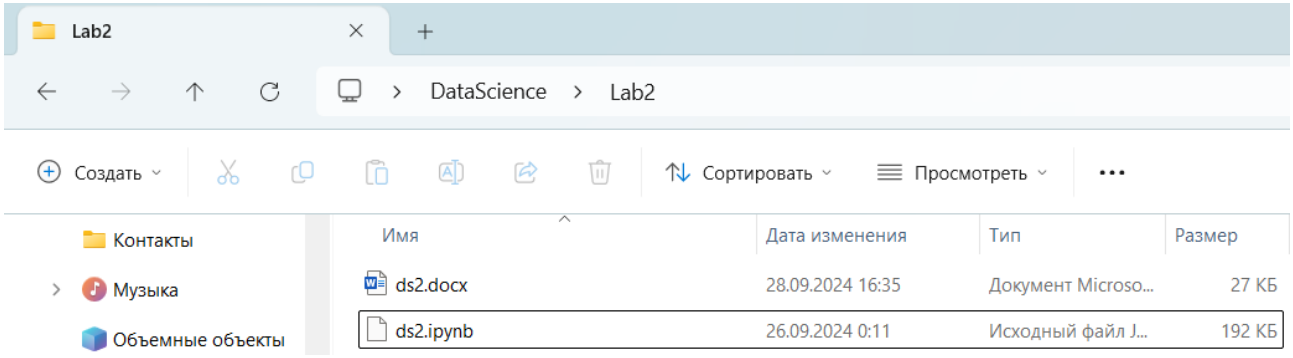

Рис.2 – Структура проєкту

ds2.ipynb – файл програмного коду лабораторної роботи;
ds2.docx – файл звіту лабораторної роботи

3.2. Результати роботи програми відповідно до завдання.

3.1. Отримання вхідних даних із властивостями, заданими в ЛР_1.

3.3. Модель вхідних даних з аномальними вимірами

В якості вхідних даних для лабораторної роботи я використав штучно згенеровані дані замість реальних, через кілька причин. По-перше, застосовувати поліноміальну регресію до них немає сенсу, оскільки в цих даних відсутній чітко виражений тренд. По-друге, вибірка мала невеликий обсяг і кількість аномальних вимірювань була незначною.

Для генерації даних я застосував адитивну модель з квадратичним трендом, додав ґауссівський шум і аномальні вимірювання, які розподілені за нормальним законом. Обсяг вибірки склав 10 000 спостережень, при цьому стандартне відхилення шуму дорівнювало 240 000, а ймовірність появи аномальних вимірювань становила 0,004.

Формула квадратичного тренду:

$$y = 0.05x^2 - 20x + 200000$$

3.4. Очищення вхідних даних від аномальних вимірів. Спосіб виявлення аномалій та очищення обрати самостійно

Метод IQR (Interquartile Range, міжквартильний розмах) використовується для виявлення аномальних вимірювань шляхом знаходження точок, що значно відхиляються від центральних значень розподілу даних. У класичному методі IQR спочатку визначаються перший (Q1) і третій (Q3) квартилі.

Міжквартильний розмах визначається як:

$$IQR = Q3 - Q1$$

Після цього для виявлення аномальних даних зазвичай використовується наступний діапазон:

$$\text{Нижня межа} = Q1 - k * IQR$$
$$\text{Верхня межа} = Q3 + k * IQR$$

Де k — множник, який зазвичай приймається рівним 1,5 або 3 для виявлення більш "екстремальних" аномалій. Будь-яке значення, яке виходить за ці межі, вважається аномальним.

Далі, для модифікації використав ковзне вікно, щоб застосувати цей метод локально до частин даних, замість аналізу всієї вибірки одночасно. Це допомагає краще виявляти аномалії в нерівномірно розподілених даних. Для кожного вікна обчислюються значення Q1, Q3, а також IQR, і застосовується той самий алгоритм:

$$\text{Нижня межа} = Q1_i - k * IQR_i$$
$$\text{Верхня межа} = Q3_i - k * IQR_i$$

Де і позначає кожен крок ковзного вікна. Значення, які виходять за ці межі в рамках кожного вікна, розглядаються як аномальні для відповідного сегменту даних. Таким чином, цей модифікований метод IQR з ковзним вікном дозволяє більш точно і локально очищати дані, виявляючи аномалії в різних частинах вибірки.

4.4. Визначення показників якості та оптимізація моделі. Показник якості та спосіб оптимізації обрати самостійно.
4.5. 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.

Для виконання поліноміальної регресії було обрано модель другого порядку, оскільки дані демонструють квадратичний тренд. Спочатку потрібно розділити вибірку на навчальний і тестовий набори. Далі, за допомогою створеного пайплайну, що включає трансформацію даних та застосування лінійної регресії до них, проводимо навчання поліноміальної моделі на навчальних даних.

Я побачив, що коефіцієнт детермінації є більшим за 0.95, що є досить високим показником, тому модель гарно підходить для прогнозування наших даних. Таким чином, немає потреби оптимізувати модель.
4.6 Прогнозування (екстраполяцію) параметрів досліджуваного процесу за «навченою» у п.5 моделлю на 0,5 інтервалу спостереження (об’єму вибірки).
4.7. Провести аналіз отриманих результатів та верифікацію розробленого скрипта.

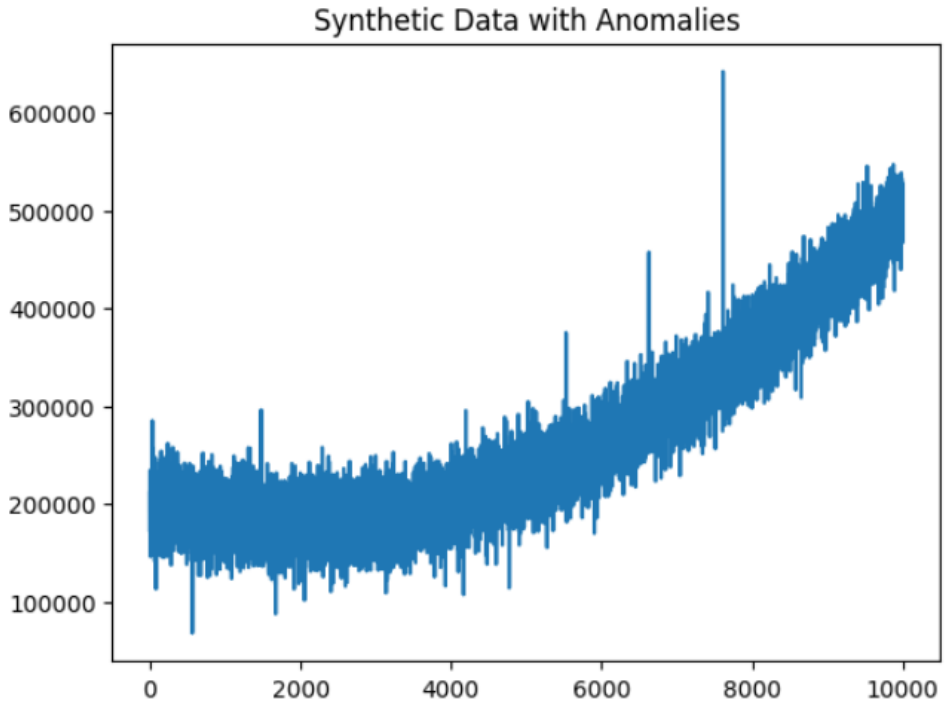

Рис.3 – Графік згенерованих даних

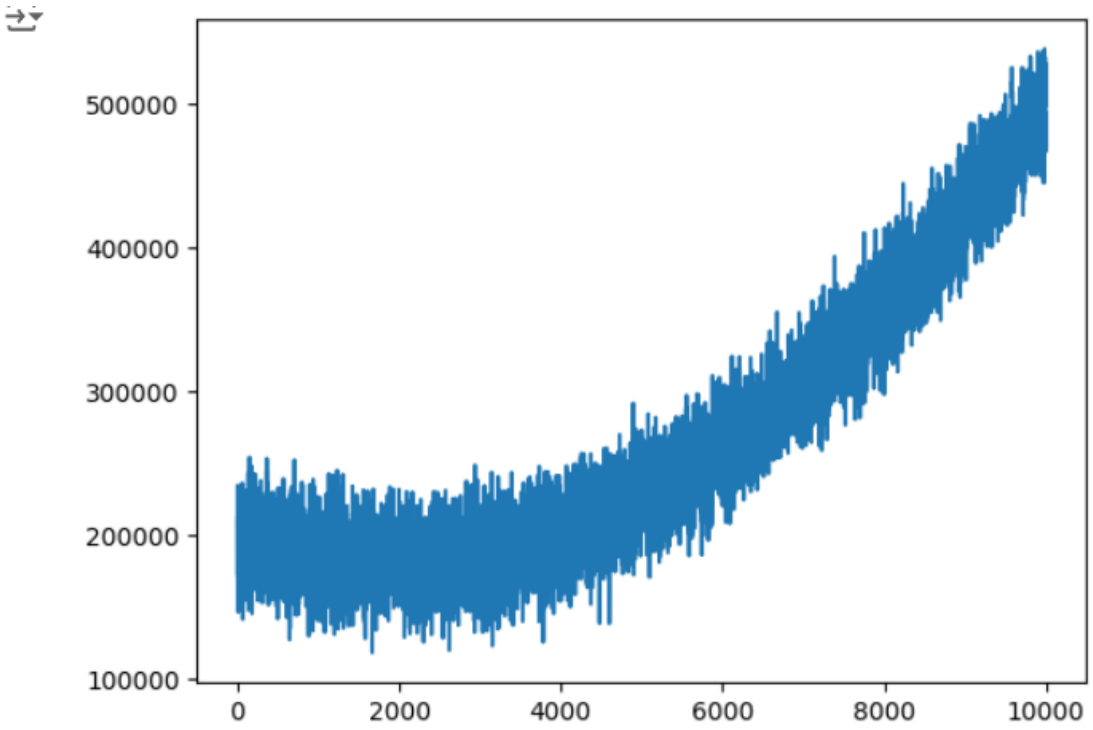

Рис.4 – Графік очищених даних

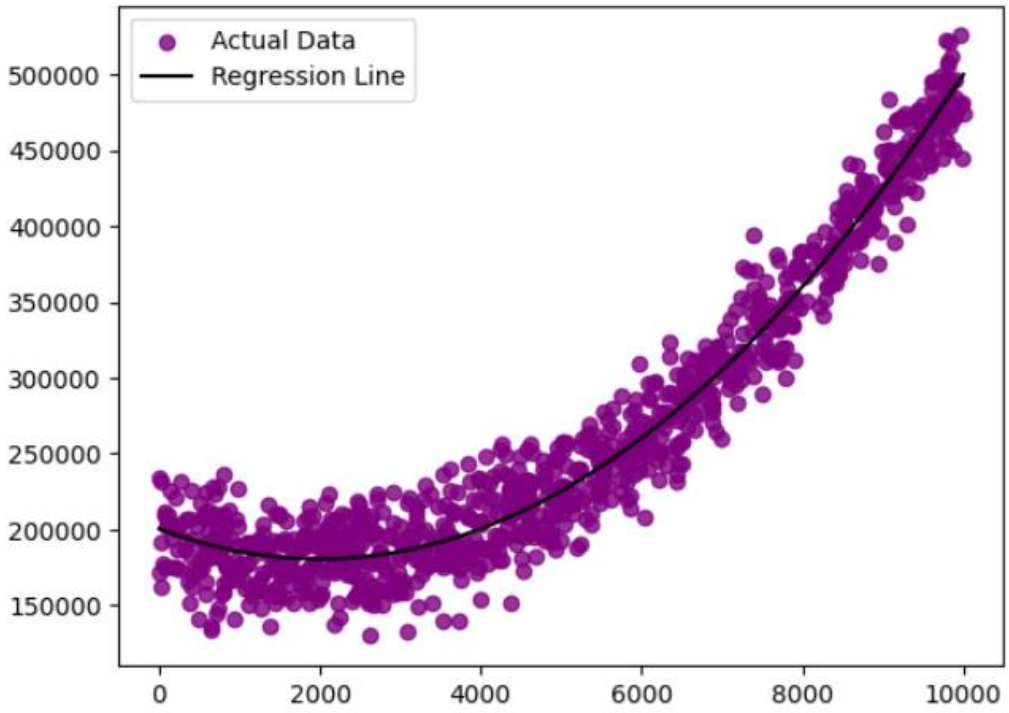

Рис.5 – Графік тестових даних та регресія, побудована на всій вибірці даних

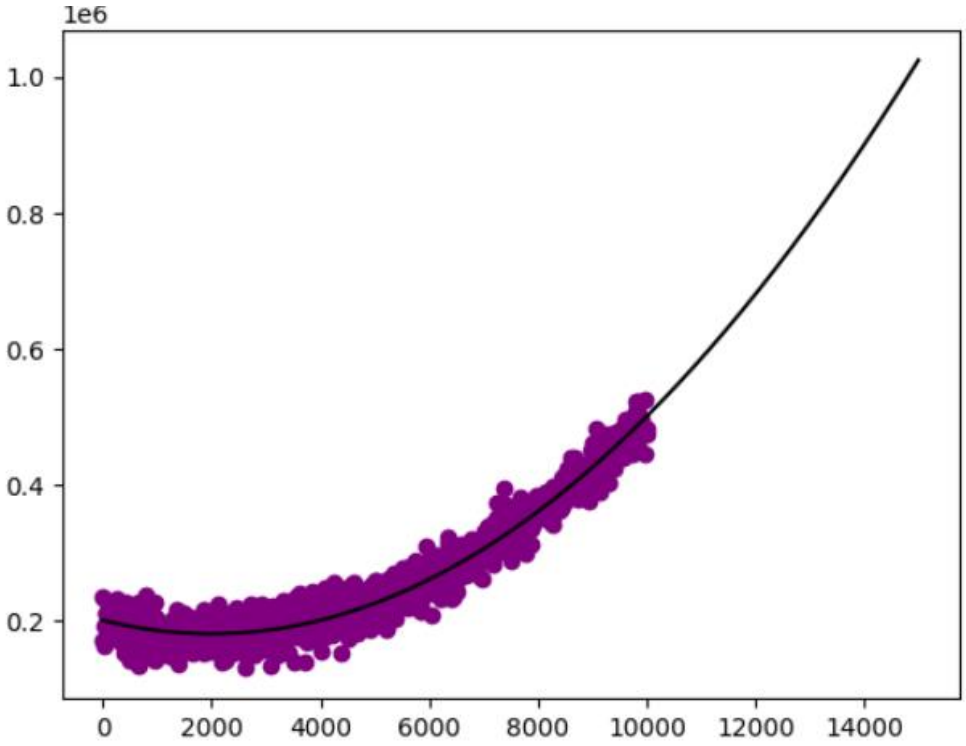

Рис.6 – Графік екстраполяції поліноміальної регресії на 0.5 від обсяду всієї вибірки

```
Mean: 425117.56
Standard deviation: 252049.89
```

Рис.7 – Результат вибіркового середнього та середньо-квадратичне відхилення

Результатом роботи програми є

Діаграми: Програма генерує різні діаграми, які візуалізують дані.

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.

3.4. Програмний код.

Програмний код послідовно реалізує алгоритм рис.1 та спрямовано на отримання результатів, поданих вище.

При цьому використано можливості Python бібліотек: pip; pandas; numpy; sklearn; matplotlib.

Контексні коментарі пояснюють сутність окремих скриптів наведеного коду програми.

3.5. 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.

[2]
```python
# Функція генерації даних із трендом та шумом
def create_synthetic_data(length, noise_level, anomaly_chance, poly_params):
    trend_line = np.polyval(poly_params, np.arange(length))
    np.random.seed(1234)
    random_noise = np.random.normal(0, noise_level, length)
    full_data = trend_line + random_noise

    anomaly_mask = np.random.rand(length) <= anomaly_chance
    anomaly_values = np.random.normal(0, 3 * noise_level, np.sum(anomaly_mask))
    full_data[anomaly_mask] += anomaly_values

    return pd.Series(full_data)
```

[3]
```python
# Параметри для генерації даних
length = 10000
noise_level = 24000
anomaly_chance = 0.004
poly_params = np.array([0.005, -20, 200000])
generated_data = create_synthetic_data(length, noise_level, anomaly_chance, poly_params)
# Візуалізація даних
plt.plot(generated_data)
plt.title('Synthetic Data with Anomalies')
plt.show()
```

```python
# Функція для виявлення та корекції аномалій за допомогою ковзного вікна та IQR
def iqr_outlier_detection(data_sample, window, min_thresh=1, max_thresh=2):
    min_diff = np.inf
    thresh_range = np.round(np.arange(min_thresh, max_thresh + 0.1, 0.1), 2)

    for current_thresh in thresh_range:
        sample_copy = data_sample.copy()
        data_values = sample_copy['values'].values

        for j in range(len(sample_copy) - window + 1):
            window_data = data_values[j: j + window]
            q1 = np.percentile(window_data, 25)
            q3 = np.percentile(window_data, 75)
            iqr_value = q3 - q1

            if data_values[j] > (q3 + current_thresh * iqr_value) or data_values[j] < (q1
                data_values[j] = np.median(window_data)

        # Застосування поліноміальної регресії та визначення відстані між коефіцієнтами
        poly_model = Polynomial.fit(sample_copy.X, sample_copy['values'], 2)
        current_coefs = poly_model.convert().coef

        sc = StandardScaler()
        scaled_true_coefs = sc.fit_transform(np.array(poly_params).reshape(1, -1))
        scaled_calculated_coefs = sc.transform(np.array(current_coefs).reshape(1, -1))
        current_diff = np.linalg.norm(scaled_true_coefs - scaled_calculated_coefs)

        if current_diff < min_diff:
            min_diff = current_diff
            best_thresh = current_thresh
            best_sample = sample_copy.copy()

    return best_thresh, best_sample
```

```python
# Визначення найкращого порогу для видалення аномалій
optimal_thresh, processed_data = iqr_outlier_detection(generated_data, 8)
print(f'Best threshold: {optimal_thresh}')
```

```
[8]  # Розбиття даних на навчальну та тестову вибірки
     X_train, X_test, y_train, y_test = train_test_split(processed_data['X'].values.reshape(-1,
     # Побудова поліноміальної регресії
     poly_regression_model = make_pipeline(PolynomialFeatures(degree=2), LinearRegression())
     poly_regression_model.fit(X_train, y_train)
     # Прогнозування та оцінка моделі
     preds = poly_regression_model.predict(X_test)
     print(f'R-squared value: {poly_regression_model.score(X_test, y_test)}')
```

```
R-squared value: 0.9560351380873512
```

```
[10] # Візуалізація результатів регресії
     plt.scatter(X_test, y_test, label='Actual Data', alpha=0.8, color='purple')
     preds_all = poly_regression_model.predict(processed_data['X'].values.reshape(-1, 1))
     plt.plot(processed_data['X'], preds_all, label='Regression Line', color='black')
     plt.legend()
     plt.show()
```

```
[11] # Прогнозування на більшій вибірці (розширені дані)
     X_extended = np.arange(0, processed_data['X'].max() + 0.5 * length)
     X_future = X_extended.reshape(-1, 1)
     extended_predictions = poly_regression_model.predict(X_future)
     # Візуалізація прогнозів на розширеній вибірці
     plt.plot(X_future, extended_predictions, color='black')
     plt.scatter(X_test, y_test, color='purple')
     plt.show()
```

```
# Статистичні характеристики прогнозованих даних
print(f'Mean: {extended_predictions.mean():.2f}')
print(f'Standard deviation: {extended_predictions.std():.2f}')
```

IV. Висновки.

В результаті виконаної лабораторної роботи, Застосувавши метод IQR з ковзним вікном і підбираючи коефіцієнт для IQR, я виявив та видалив аномальні вимірювання в синтезованих даних. На очищених даних я навчив і оцінив модель поліноміальної регресії. Отриманий коефіцієнт детермінації виявився дуже високим, що свідчить про адекватність моделі для наших даних. Крім того, я здійснив екстраполяцію результатів і обрахував статистичні характеристики спрогнозованих даних. Значення середньоквадратичного відхилення виявилося досить близьким до заданого під час генерації даних