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 «Вступ до технології Data Science»

Тема:

СТАТИСТИЧНЕ НАВЧАННЯ З ПОЛІНОМІАЛЬНОЮ РЕГРЕСІЄЮ

Виконав:
Студент 3 курсу кафедри ІПІ ФІОТ,
Навчальної групи ІП-11
Сідак К.І.

Перевірив:
Професор кафедри ОТ ФІОТ
Писарчук О.О.

Київ 2023

I. 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.

II. 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.

III. Завдання IV рівня(максимум 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.
3.1. Здійснити розробку власного алгоритму виявлення аномальних вимірів та / або «навчання» параметрів відомих алгоритмів «бачити» властивості статистичної вибірки.

IV. Виконання лабораторної роботи.

4.1. Отримання вхідних даних із властивостями, заданими в Лр_1.

4.2. Модель вхідних даних із аномальними вимірами.

В якості вхідних даних я використав штучно згенеровані дані замість реальних даних з першої лабораторної роботи через ряд причин. По-перше що навчати поліноміальну регресію на них немає сенсу, адже в них відсутній як такий явно виражений тренд. По-друге, це відносно невеликий об’єм вибірки та невелика кількість аномальних вимірів.

Для генерації даних я використав адитивну модель з квадратичним трендом з ґауссівським шумом та аномальними вимірами, розподіленими за нормальним законом. Об’єм вибірки становить 10000, середньо квадратичне відхилення становить 240000 для шуму, а ймовірність аномальних вимірів становить 0,004. Квадратичний тренд є наступним:

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

```
In 2
def generate_data(size, std, outliers_probability, polynom_coefficients):
    trend = np.polyval(polynom_coefficients, np.arange(size))
    np.random.seed(1234)
    noise = np.random.normal(0, std, size)
    data = trend + noise
    mask = np.random.rand(size) <= outliers_probability
    anomalies = np.random.normal(0, 3 * std, np.sum(mask))
    data[mask] += anomalies

    return pd.Series(data)
Executed at 2023.10.16 22:51:47 in 10ms

In 3
size = 10000
std = 24000
outliers_probability = 0.004
polynom_coefficients = np.array([0.005, -20, 200000])
data = generate_data(size, std, outliers_probability, polynom_coefficients)
plt.plot(data)
plt.title('Synthetic Data with Outliers')
plt.show()
Executed at 2023.10.16 22:51:47 in 153ms
```

Рис. 4.1 – Генерація даних

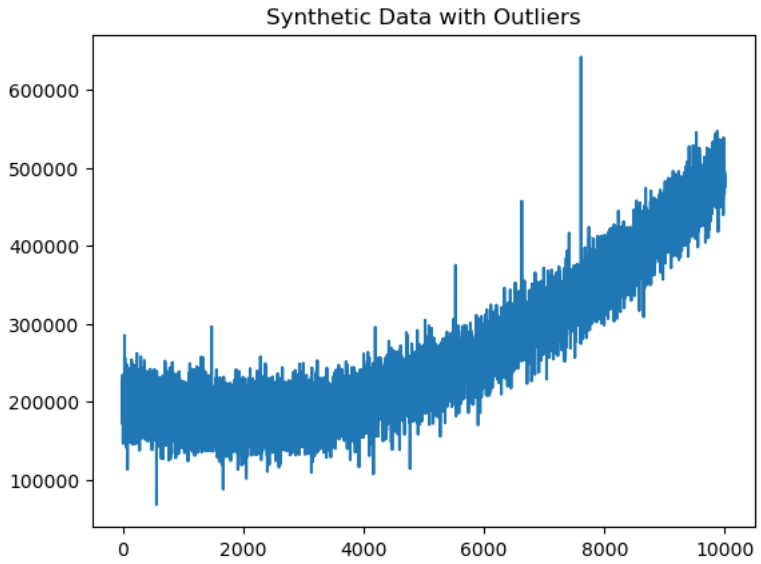

Рис. 4.2 – Графік згенерованих даних

4.3 Очищення вхідних даних від аномальних вимірів. Здійснити розробку власного алгоритму виявлення аномальних вимірів та / або «навчання» параметрів відомих алгоритмів «бачити» властивості статистичної вибірки.

Для виявлення аномальних вимірів та їх очищення я використав метод IQR та модифікував його з використанням ковзного вікна. Спочатку варто описати принцип роботи звичайного методу IQR:

1. Обраховуються перший та третій квартилі (Q_1 та Q_3 відповідно).
2. Обраховується міжквартильний розмах за наступною формулою:
$$IQR = Q_3 - Q_1$$
3. Вимір (позначимо як x) вважається аномальним, якщо виконується наступна умова:
$$x < Q_1 - 1.5 * IQR \text{ або } x > Q_3 + 1.5 * IQR$$

Тепер треба описати метод ковзного вікна з використанням IQR:

1. Для i від 0 до $n - window\_size$ включно, де n – розмір вибірки, а $window\_size$ – розмір вікна виконуються наступні кроки:
    1. Формується ковзне вікно заданого розміру window_size від елемента з індексом i до елемента з індексом $i + window\_size - 1$ включно.
    2. Обраховуються перший та третій квартилі для цього вікна.
    3. Обраховується IQR для поточного вікна.
    4. Якщо умова, описана вище, виконана, елемент з індексом i вважається аномальним та замінюється на значення медіани поточного вікна.
2. Для елементів з індексом i від $n - window\_size + 1$ до $n - 1$ включно виконується крок 1.4.

Крім того, замість використання коефіцієнту 1.5 при IQR даний алгоритм перебирає значення від $min\_threshold$ до $max\_threshold$ (1 та 2 за замовчуванням відповідно) з кроком 0.1. Для кожного значення створюється копія вибірки, виконується алгоритм ковзного вікна та навчається поліноміальна (квадратичина) модель на очищених даних. Кращим вважається таке значення коефіцієнту при IQR, при якому коефіцієнти отриманої моделі найбільш близькі до «ідеальних» (у нашому випадку заданих трендом). Значення коефіцієнтів (отриманих та ідеальних) стандартизуються та обраховується Евклідова відстань між цими двома векторами. Коефіцієнт при IQR, при якому ця відстань мінімальна, буде обрано. У результаті очищена вибірка для даного коефіцієнту й буде результуючою.

```
In 5
def sliding_window_iqr(sample, window_size, min_threshold=1, max_threshold=2):
    min_distance = np.inf
    thresholds = np.round(np.arange(min_threshold, max_threshold + 0.1, 0.1), 2)
    for threshold in thresholds:
        sample_copy = sample.copy()
        y_copy = sample_copy.y.values
        for i in range(len(sample_copy) - window_size + 1):
            current_window = y_copy[i: i + window_size]
            current_window_q1 = np.percentile(current_window, 25)
            current_window_q3 = np.percentile(current_window, 75)
            current_window_iqr = current_window_q3 - current_window_q1
            if y_copy[i] > current_window_q3 + threshold * current_window_iqr or y_copy[
                i] < current_window_q1 - threshold * current_window_iqr:
                y_copy[i] = np.median(current_window)
        for i in range(len(sample_copy) - window_size + 1, len(sample_copy)):
            if y_copy[i] > current_window_q3 + threshold * current_window_iqr or y_copy[
                i] < current_window_q1 - threshold * current_window_iqr:
                y_copy[i] = np.median(current_window)
        polynomial_model = Polynomial.fit(sample_copy.X, sample_copy.y, 2)
        calculated_coefs = polynomial_model.convert().coef
        sc = StandardScaler()
        ideal_coefs_scaled = sc.fit_transform(np.array(polynom_coefficients).reshape(1, -1))
        calculated_coefs_scaled = sc.transform(np.array(calculated_coefs).reshape(1, -1))
        current_distance = np.linalg.norm(ideal_coefs_scaled - calculated_coefs_scaled)
        if current_distance < min_distance:
            min_distance = current_distance
            best_threshold = threshold
            new_sample = sample_copy.copy()

    return best_threshold, new_sample
```

Рис. 4.3 – Реалізація методу IQR з ковзним вікном

```
In 6  1  threshold, data = sliding_window_iqr(data, 8)
      2  print(f'Best threshold: {threshold}')
         Executed at 2023.10.17 20:29:29 in 5s 856ms

         Best threshold: 1.0
```

Рис. 4.4 – Оптимальне значення коефіцієнту при ковзному вікні розміру 8

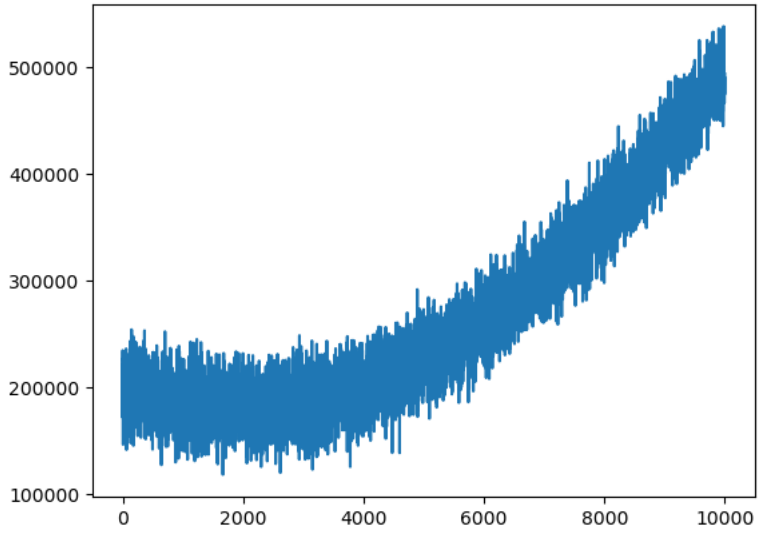

Рис. 4.5 – Очищені дані від аномальних вимірів

4.4. Визначення показників якості та оптимізація моделі (вибір моделі залежно від значення показника якості). 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.

В якості поліноміальної регресії обрано поліноміальну регресію другого порядку, адже маємо квадратичний тренд. Спершу треба розбити вибірку на навчальні та тестові набори у відношенні 9:1 відповідно. Далі за допомогою створення пайплайну, що складається з трансформації даних та використання лінійної регресії на них, навчимо поліноміальну регресію на навчальних наборах.

```
In 8  1  X_train, X_test, y_train, y_test = train_test_split(data.X.values.reshape(-1, 1), data.y, test_size=0.1,
      2                                                      random_state=1234)
      3  polynomial_regression = make_pipeline(PolynomialFeatures(degree=2), LinearRegression())
      4  polynomial_regression.fit(X_train, y_train)
      5  predictions = polynomial_regression.predict(X_test)
      6  print(f'R-squared value: {polynomial_regression.score(X_test, y_test)}')
         Executed at 2023.10.17 20:29:29 in 10ms

         R-squared value: 0.956035138087353
```

Рис. 4.6 – Коефіцієнт детермінації навченої моделі на тестовому наборі

Можна побачити, 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, тим паче, що отриманий показник саме на тестовому наборі.

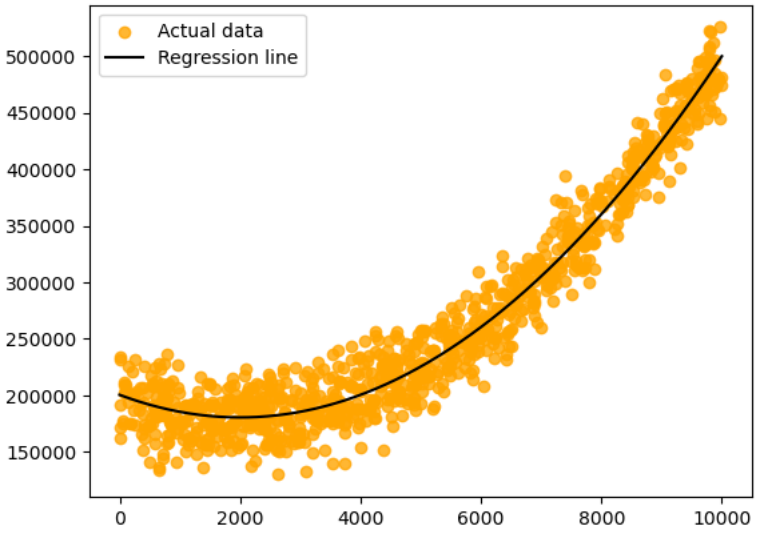

Рис. 4.6 – Дані тестового набору та регресія, побудована на всіх даних

4.6 Прогнозування (екстраполяцію) параметрів досліджуваного процесу за «навченою» у п.5 моделлю на 0,5 інтервалу спостереження (об'єму вибірки).

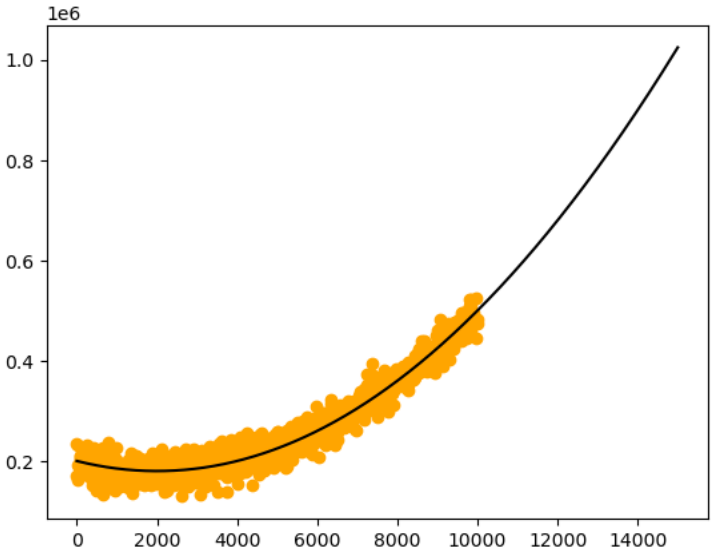

Рис. 4.7 – Екстраполяція поліноміальної регресії на 0.5 інтервалу спостереження від обсягу всієї вибірки

```
In 11  1  print(f'Mean: {extended_predictions.mean():.2f}')
       2  print(f'Standard deviation: {extended_predictions.std():.2f}')
Executed at 2023.10.17 21:22:50 in 21ms

   Mean: 425117.56
   Standard deviation: 252049.89
```

Рис. 4.8 – Вибіркове середнє та середньо квадратичне відхилення екстрапольованої моделі

4.7. Провести аналіз отриманих результатів та верифікацію розробленого скрипта.

Отже, методом IQR з ковзним вікном з підбором коефіцієнту при IQR я виявив та очистив синтезовані дані від аномальних вимірів. На очищених даних я навчив та оцінив модель поліноміальної регресії. Отриманий коефіцієнт детермінації є дуже високим, тому модель підходить для наших даних. Крім того, я провів екстраполяцію результатів та обрахував

статистичні характеристики спрогнозованих даних. Значення середньо квадратичного відхилення є досить близьким до заданого при генерації даних (24000).