

**WOJSKOWA AKADEMIA TECHNICZNA**  
im. Jarosława Dąbrowskiego  
**WYDZIAŁ CYBERNETYKI**

---



**SPRAWOZDANIE**  
**Metody Eksploracji Danych**

Temat laboratorium: **FUNKCJA LOGISTYCZNA**

**INFORMATYKA**

(kierunek studiów)

**INŻYNIERIA SYSTEMÓW – ANALIZA DANYCH**

(specjalność)

Zespół:

**Michał ŚLĘZAK**  
**Szymon OLEŚKIEWICZ**

Prowadzący laboratorium:

**Dr inż. Romuald Hoffmann, prof.**  
**WAT**

---

**Warszawa 2025**



**Spis treści**

Rozdział I. Zadanie 2 – Błędy krytyczne .....	4
I.1. Wykorzystane narzędzia i zależności .....	5
I.2. Model .....	7
Wnioski.....	9
Bibliografia.....	10
Spis rysunków .....	11
Spis tabel .....	11
Załączniki .....	11

## Rozdział I. Zadanie 2 – Błędy krytyczne

### **Zadanie 2**

Przez cały okres eksploatacji pewnego systemu operacyjnego (OS) zbierano dane dotyczące liczby błędów krytycznych wykrytych w tym czasie w oprogramowaniu. Zebrane obserwacje w układzie miesięcznym przedstawiono w tabeli 3.

**Tabela 2. Liczba błędów krytycznych oprogramowania badanego OS w układzie miesięcznym**

Nr miesiąca	Liczba błędów								
1	1	21	9	41	3	61	2	81	8
2	0	22	0	42	2	62	19	82	0
3	0	23	0	43	1	63	7	83	6
4	0	24	3	44	6	64	2	84	5
5	0	25	1	45	3	65	5	85	10
6	0	26	1	46	0	66	1	86	0
7	0	27	0	47	1	67	12	87	5
8	0	28	1	48	1	68	4	88	1
9	0	29	0	49	0	69	6	89	2
10	0	30	2	50	0	70	4	90	1
11	0	31	10	51	14	71	7	91	2
12	0	32	0	52	1	72	2	92	1
13	0	33	16	53	4	73	2	93	0
14	0	34	0	54	1	74	3		
15	0	35	2	55	1	75	8		
16	0	36	2	56	7	76	4		
17	0	37	1	57	14	77	6		
18	0	38	1	58	6	78	3		
19	0	39	1	59	0	79	3		
20	1	40	0	60	1	80	5		

W zadaniu proszę:

1. Wyznaczyć zależność sumarycznej liczby błędów w okresie eksploatacji oprogramowania badanego systemu operacyjnego.
2. Na podstawie opracowanego modelu i przeprowadzonych obliczeń sformułować własne wnioski.
3. Wyniki analizy proszę zatrzymać w postaci sprawozdania, do którego proszę dodać jako załączniki wszystkie pliki z obliczeniami (obliczenia można przeprowadzić w dowolnie wybranym narzędziu)

## I.1. Wykorzystane narzędzia i zależności

### I.1.1. Opis teoretyczny

W celu wyznaczenia zależności sumarycznej liczby błędów w okresie eksploatacji oprogramowania badanego systemu operacyjnego na podstawie wykładów oraz wiedzy własnej przygotowano program w języku Python wykorzystujący biblioteki Pandas, NumPy, stastmodel oraz scipy, które również zostały wykorzystane do obliczeń oraz wizualizacji wyników.

Funkcja logistyczna to nieliniowa funkcja 1 zmiennej, zmienną tą jest zazwyczaj czas (t). Wykresem tej funkcji jest krzywa przypominająca kształtem literę „S” oraz należy do rodziny krzywych sigmoidalnych (kształtem przypominającym właśnie literę „S”). Funkcja ta ma następujące właściwości – dla  $t > 0$  wartość funkcji logistycznej początkowo bardzo szybko rośnie a później tempo jej wzrostu maleje i końcowo stabilizuje się w pobliżu pewnej asymptoty. Klasyczna postać funkcji logistycznej jest pokazana poniżej.

$$y(t) = \frac{\alpha}{1 + \beta \cdot e^{-\gamma \cdot t}}$$

gdzie  $\alpha > 0, \beta > 0, \gamma > 0$ .

Parametry te oznaczają odpowiednio:

$\alpha$  – poziom nasycenia odpowiadający asymptotej górnej. Jest to graniczna wartość, do której dąży funkcja przy  $t$  dążącym do nieskończoności. W kontekście zadania laboratoryjnego oznacza przewidywaną, całkowitą liczbę błędów jakie zawiera system.

$\beta$  – parametr przesunięcia odpowiadający za położenie krzywej na osi OX, osi czasu. Wpływa na wartość początkową funkcji dla  $t = 0$ , czyli liczbę błędów wykrytych na początku eksploatacji systemu.

$\gamma$  – współczynnik tempa wzrostu określający jak bardzo krzywa jest stroma. Im większy ten parametr, tym szybciej funkcja osiąga nasycenie.

W realizowanym zadaniu laboratoryjnym zastosowano sparametryzowaną postać funkcji logistycznej jak poniżej, z jawnym punktem przegięcia, który jest parametrem funkcji, w przeciwieństwie do poprzedniej postaci, gdzie punkt przegięcia jest obliczany z wartością innych parametrów – dla  $t = \ln \frac{\beta}{\gamma}$ .

$$y(t) = \frac{K}{1 + e^{-a(t-t_0)}}$$

gdzie:

$K$  – odpowiada parametrowi alfa z klasycznego modelu. Oznacza poziom nasycenia, maksymalną wartość, którą funkcja może osiągnąć. W naszym przypadku jest to

estymowana całkowita liczba błędów krytycznych w eksploatowanych systemie operacyjnym.

$a$  – odpowiada parametrowi gamma z poprzedniego modelu. Określa współczynnik tempa wzrostu, który wpływa na to jak stroma jest krzywa. Wysoka wartość tego parametru oznacza, że błędy są wykrywane bardzo często w krótkim okresie czasu.  $t_0$  – punkt przegięcia funkcji, jest to punkt, w którym funkcja osiąga połowę swojej maksymalnej wartości. Jest to punkt w czasie, w którym tempo przyrostu błędów było największe. Dla  $t < t_0$  liczba wykrywanych błędów krytycznych rośnie (wartość funkcji szybko rośnie), dla  $t = t_0$  funkcja osiąga połowę maksymalnej wartości funkcji i dla  $t > t_0$  liczba wykrywanych błędów maleje.

W celu zbadania jak dobrze wykres funkcji logistycznej dopasowuje się do sumarycznej liczby błędów można wykorzystać współczynnik determinacji, który obliczany jest na podstawie poniższego wzoru. Współczynnik ten przyjmuje wartości od 0 do 1. Im bliżej 1, tym nasz model jest bardziej dopasowany.

$$R^2 = \frac{\sum_{i=1}^n (\hat{y}_i - \bar{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y}_i)^2}$$

### I.1.2. Metoda doboru parametrów

Ze względu na nieliniowy charakter rozpatrywanej funkcji logistycznej nie jest możliwe wyznaczenie jej parametrów za pomocą, np. Metody Najmniejszych Kwadratów, jak to miało miejsce w regresji liniowej. W celu estymacji parametrów modelu zastosowano Nieliniową Metodę Najmniejszych Kwadratów. Zadanie optymalizacyjne zdefiniowane jest jako minimalizacja sumy kwadratów reszt (błędów) pomiędzy wartościami rzeczywistymi ( $y$  - skumulowana liczba błędów) a wartościami teoretycznymi modelu (to co zwraca nam funkcja po podstawieniu odpowiedniego  $t$ ), jak poniżej.

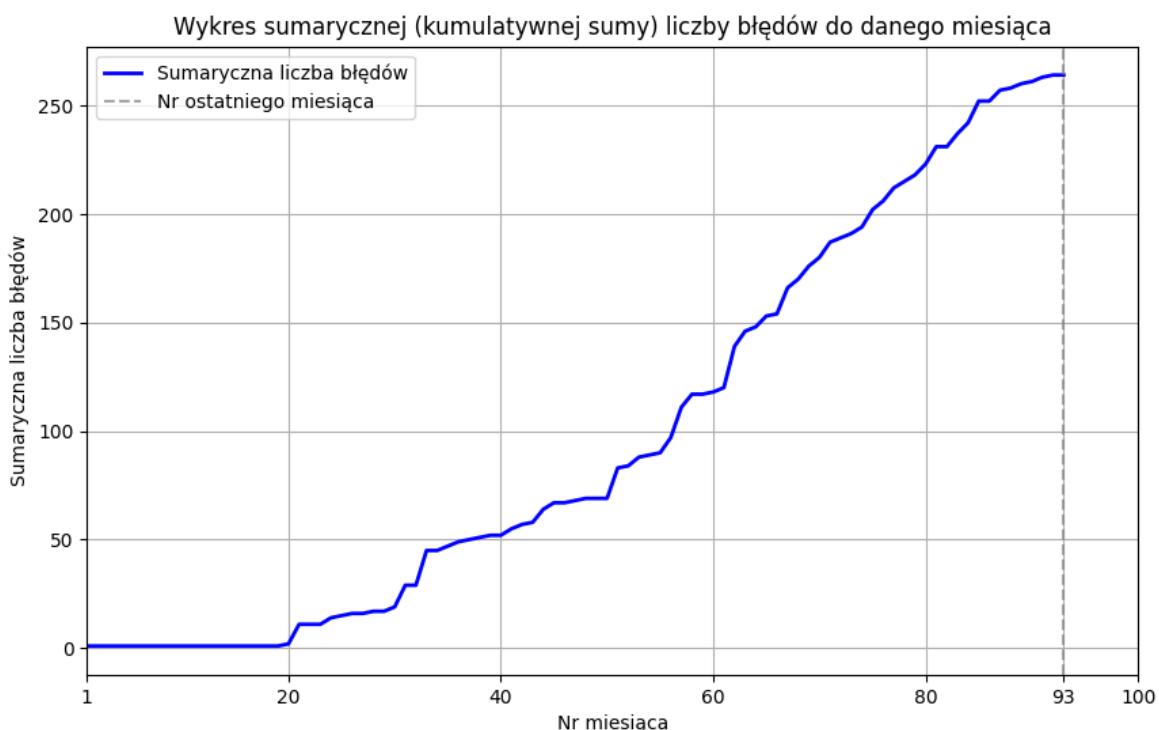
$$\min_{K,a,t_0} \sum_{i=1}^n \left( y_i - \frac{K}{1 + e^{-a(t_i - t_0)}} \right)^2$$

Do obliczeń wykorzystano funkcję `curve_fit` z biblioteki SciPy (`scipy.optimize`). Domyślnie korzysta ona algorytm Lavenberga-Marquardta. Jest to iteracyjna metoda łącząca cechy spadku gradient oraz metody Gaussa-Newtona. Aby rozpocząć minimalizację, należy podać swego rodzaju punkt startowy (wektor parametrów początkowych  $p0$ ). W naszej implementacji, zastosowano następujące wartości początkowe, jak poniżej. Dla parametru  $K$  przyjęto wartość maksymalną z danych rzeczywistych, która jest de facto dolnym ograniczeniem asymptoty, tzn. system operacyjny musi mieć co najmniej tyle błędów, ile już ich wykryto. Dla parametru tempa wzrostu a przyjęto wartość 1, co jest standardową wartością domyślną. Dla

parametru punktu przegięcia przyjęto medianę czasu trwania eksploatacji systemu, założono że punkt kulminacyjny sumarycznej liczby wykrytych błędów znajduje się w połowie badanego okresu, sugerując się wykresem skumulowanej liczby błędów.

## I.2. Model

W pierwszej kolejności przekształcono dane o miesięcznej liczbie wykrytych błędów krytycznych. Dodano do danych kolumnę przedstawiającą skumulowaną liczbę błędów do danego miesiąca. Przedstawiono to na poniższym wykresie.



Rys. 1. Wykres kumulatywnej sumy liczby błędów do danego miesiąca

Jak widać po wykresie i analizie sumarycznej liczby błędów krytycznych, wykres ukazuje charakterystyczny dla krzywych sigmoidalnych kształt litery „S”. W początkowej fazie eksploatacji systemu (do ok. 20 miesiąca) przyrost liczby błędów był bardzo mały, znikomy. Następnie, obserwujemy gwałtowny wzrost dynamiki wykrywania błędów krytycznych, co sugeruje fazę aktywnego użytkowania systemu operacyjnego i jego testy. Pod koniec badanego okresu, krzywa wyraźnie stabilizuje się, wykazuje tendencję do nasycenia, tj. wypłaszczenia się, co oznacza, że tempo wykrywania nowych błędów spada. Uzasadnione jest zatem użycie modelu logistycznego. Warto także zauważyć, że punkt kulminacyjny wykrywania błędów następuje mniej więcej po środku badanego okresu czasu, więc uzasadnione jest użycie później w doborze parametrów mediany czasu jako wartości początkowej dla algorytmu estymacji parametrów.

W kolejnym etapie rozwiązania, wyznaczono parametry funkcji logistycznej metodą opisaną we wstępie teoretycznym: Metoda doboru parametrów. Wyznaczone parametry przedstawiono poniżej oraz obliczono współczynnik determinacji dopasowania modelu do danych.

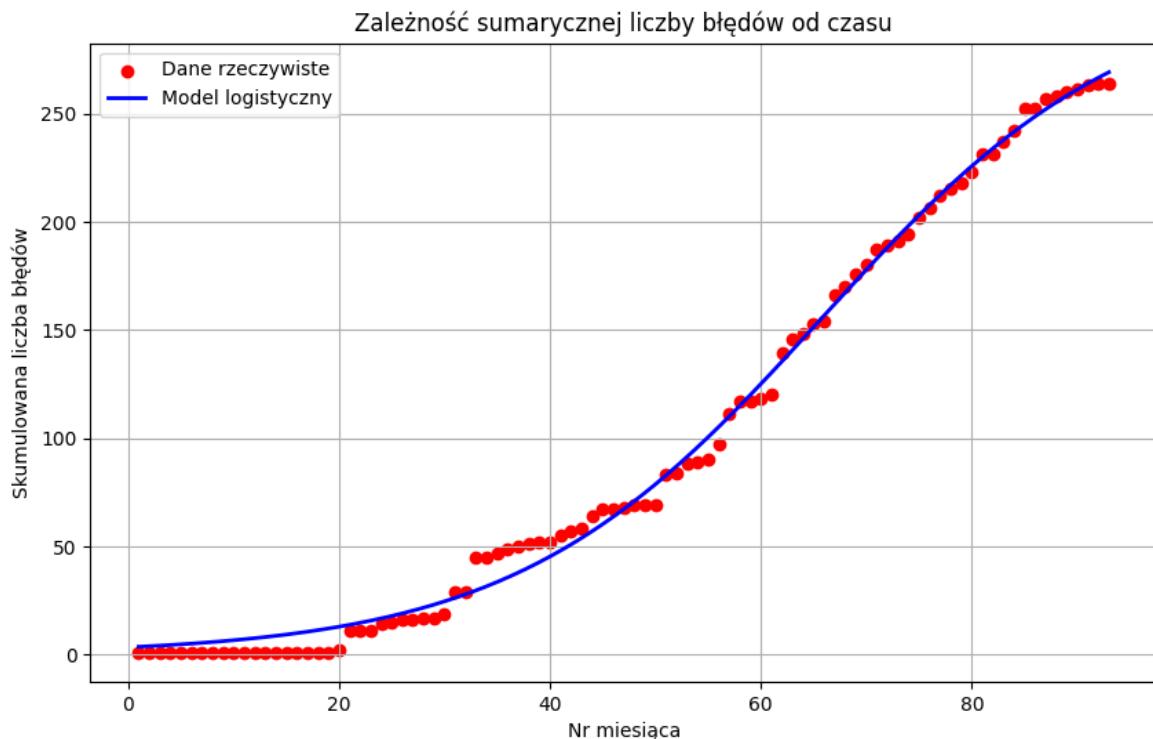
$$K \approx 310,38$$

$$a \approx 0,069$$

$$t_0 \approx 65,73$$

$$R^2 \approx 99,6\%$$

Dopasowanie modelu do danych przedstawiono także na poniższym wykresie.



Rys. 2. Wykres zależności sumarycznej liczby błędów od czasu (nr miesiąca)

Nasz model logistyczny bardzo dobrze opisuje proces wykrywania błędów krytycznych w badanym systemie operacyjnym, świadczy o tym wysoki współczynnik determinacji równy w przybliżeniu 99,6%. Krzywa niemalże idealnie pokrywa się z danymi.

Parametr nasycenia wynosi ok. 310,38. Oznacza to, że teoretyczna, całkowita sumaryczna liczba błędów krytycznych, które mogą wystąpić w okresie eksploatacji systemu operacyjnego wynosi ok. 310. Biorąc pod uwagę, że w ciągu 93 miesięcy eksploatacji wykryto dotychczas 264 błędy, można wnioskować, że wykryto ok. 85% wszystkich przewidywanych błędów krytycznych.

Parametr tempa wzrostu wynosi ok. 0,069, co odzwierciedla umiarkowane tempo wzrostu sumarycznej liczby błędów. Wartość ta wskazuje na stabilny i dość długofalowy proces pojawiania się błędów krytycznych w czasie jego eksploatacji.

Punkt przegięcia wynosi ok. 65,73, co odpowiada mniej więcej końcowi 65. miesiąca eksploatacji systemu operacyjnego w badanym okresie czasu. Po przekroczeniu tego miesiąca, tempo wykrywania kolejnych błędów krytycznych, liczby kolejnych błędów zaczęło stopniowo maleć, co na wykresie objawia się przejściem krzywej w fazę wypłaszczenia się.

## **Wnioski**

Proces pojawiania się błędów w badanym systemie operacyjnym przebiega zgodnie z modelem logistycznym, czego m.in. dowodzi charakterystyczny kształt krzywej – litera „S”. Początkowy powolny wzrost liczby błędów, następna faza intensywnego występowania błędów (przyrost liczby błędów) oraz końcowe wygasanie trendu wzrostowego są dość typowe dla oprogramowania. Choć system zbliża się do fazy nasycenia, parametr K sugeruje, że proces utrzymania tego systemu powinien być kontynuowany, gdyż system nie osiągnął jeszcze pełnego, teoretycznego poziomu bezawaryjności – nie osiągnął teoretycznej maksymalnej liczby błędów krytycznych, która jest reprezentowana przez obliczoną wartość K.

## Bibliografia

- [1] [https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve\\_fit.html](https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html)
- [2] <https://www.youtube.com/watch?v=pNE1ufDXNSc&t=3s>
- [3] [https://en.wikipedia.org/wiki/Levenberg%E2%80%93Marquardt\\_algorithm](https://en.wikipedia.org/wiki/Levenberg%E2%80%93Marquardt_algorithm)

## Spis rysunków

- Rys. 1. Wykres kumulatywnej sumy liczby błędów do danego miesiąca ..... 7  
Rys. 2. Wykres zależności sumarycznej liczby błędów od czasu (nr miesiąca) ..... 8

## Spis tabel

Nie znaleziono żadnych pozycji spisu treści.

## Załączniki



Lab-2-Zad-2-Michał-Ślęzak-Szymon-Oleśkiewicz.ipynb



Lab-2-Zad-2-Michał-Ślęzak-Szymon-Oleśkiewicz.py

<https://colab.research.google.com/drive/1mmNPkLnjXe-VCONVQfZEIW4PIgaljup6?usp=sharing>