# Wstęp

# Cel i zakres pracy

# Przegląd literatury i analiza rozwiązań

Rozdział ten stanowi techniczne wprowadzenie do zagadnienia, oparte na analizie literatury i istniejących rozwiązań inżynierskich. Analiza literatury i istniejących rozwiązań stanowi istotny etap procesu projektowego, umożliwiający lepsze zrozumienie kontekstu danego problemu oraz identyfikację potencjalnych obszarów doskonalenia. Rozdział skupia się na przeglądzie literatury związanej z tematyką pracy inżynierskiej oraz analizie istniejących rozwiązań, mającej na celu dostarczenie solidnej podstawy teoretycznej i technologicznej dla dalszych etapów badawczych. W tym rozdziale zostaną dogłębnie poruszone teoretyczne kwestie związane z tematem pracy.

## Strumienie danych

Strumieniem danych nazywamy uporządkowany zestaw danych, gdzie każda wartość jest przypisana do określonego momentu czasowego.

Strumień składa się z punktów danych, najczęściej zbieranych w regularnych odstępach czasowych, co pozwala na dokładniejszą analizę zmian w czasie. W ramach szeregów czasowych można identyfikować różne wzorce, trendy, sezonowe wahania oraz nieregularne zdarzenia.

W pracy A.Arsau, S.Babu, J.Widom(1) strumien danych jest nazywany nieograniczonym zbiorem elementów krotek należących do schematu strumienia i stempli czasowych tych elementów.

$$S = (s, t) \tag{1}$$

Z tego wynika, że charakterystyczną naturalną cechą strumieni jest szeregowość. Wartości nie są jedynym przedmiotem analizy, ale głównie ich kolejność i kontekst który zarysowują w czasie.

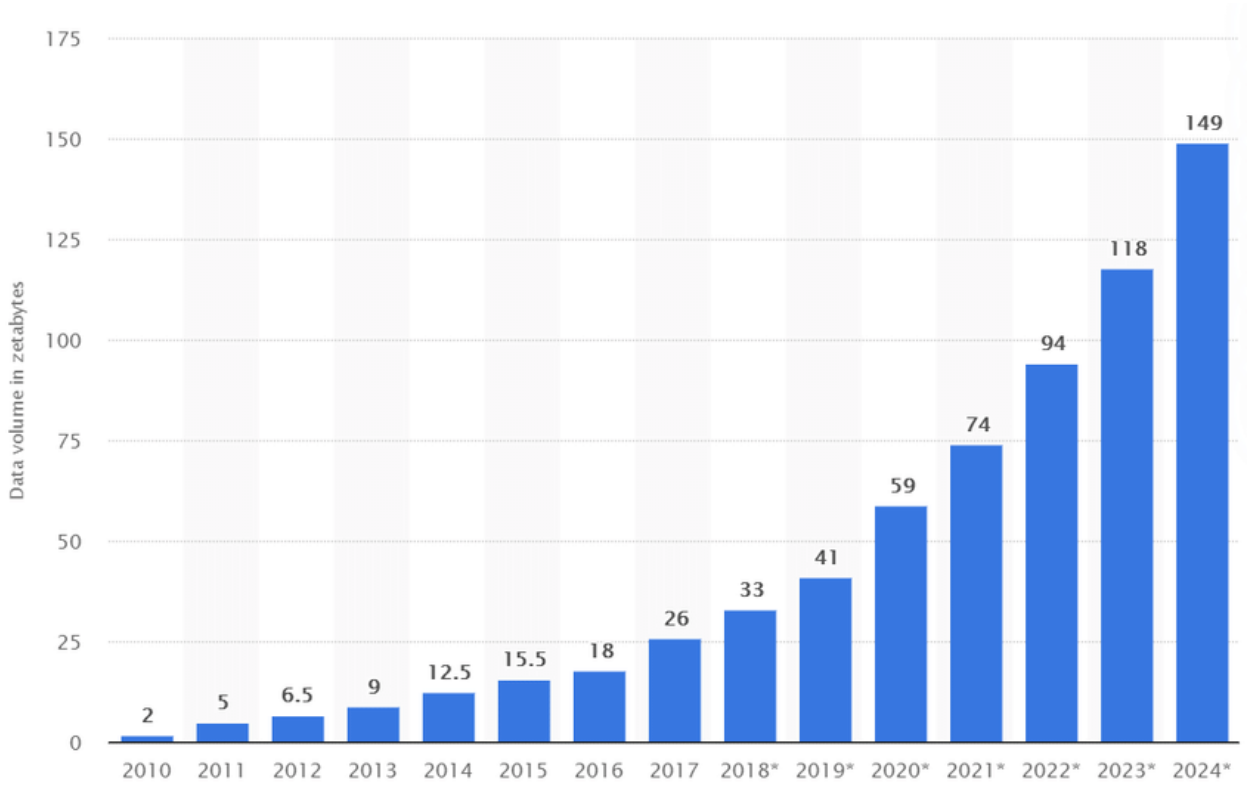
Szeregi czasowe są używane do monitorowania i prognozowania zmian, co pozwala wspierać procesy decyzyjne w każdej dziedzinie biznesu. Mogą obejmować dane z różnych dziedzin, takich jak gospodarka, nauki przyrodnicze, zdrowie, finanse czy technologia.

## Wykrywanie wyjątków w strumieniach danych

Procesy gromadzenia danych, mimo postępu technologicznego, zawsze niosą ze sobą pewne ryzyko i nie są idealne. Istnieje wiele czynników, zarówno technicznych, jak i ludzkich, które mogą wprowadzić błędy do zebranych danych. Może on wynikać z wadliwego sprzętu pomiarowego, błędu ludzkiego lub przypadkowego zbiegu okoliczności. Dane przesyłane do analizy

mogą zawierać szum, błędy pomiarowe, wartości niemożliwe lub w skrajnych przypadkach nie mieć wartości.

Proces czyszczenia danych stał się integralnym i fundamentalnym krokiem w procesie analizy danych, w szczególności w dzisiejszych czasach kiedy ilość przesyłanych danych z roku na rok jest coraz większa.




Rysunek 1: Ilość danych stworzonych / pobranych / skopiowanych w latach 2010 - 2021 z prognozami do roku 2024 (2)

Dzięki czyszczeniu analiza staję się dokładniejsza a modele lepiej spełniają swoją role w prognozowaniu kolejnych wartości. Podstawowym krokiem czyszczenia danych jest wykrywanie i usuwanie wyjątków z serii danych.

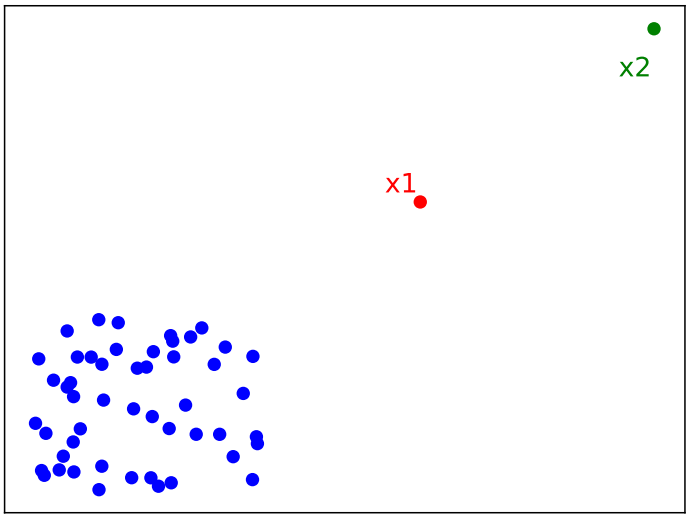
**Wyjątkiem** nazywamy obserwację, której wartość znacząco różni się od innych wartości w losowej próbie z populacji. Określenie "znacząco różni" nie jest precyzyjnym określeniem. Kontekst każdej analizy jest wyjątkowy. W rozumieniu tej definicji, każda analiza ma za zadanie zdefiniować czym i jaka będzie znacząca różnica.

## Efekt Maskowania

Efekt maskowania *(ang. masking effect)* jest obecnym problemem w wykrywaniu wyjątków, wpływa on negatywnie na dokonywane analizy. Dlatego metoda wykrywania wyjątków powinna być odporna na działanie efektu i wykryć zamaskowaną anomalię.

**Maskowaniem** wyjątku nazywamy zjawisko nie wykrycia wyjątku, z powodu wpływu większej anomalii na statystykę testową, która determinuję wyjątek.

Efekt maskowania może wystąpić w sytuacji, gdy analiza, z góry narzuca wykrycie i usunięcie ustalonej liczby wyjątków. Maskowanie wystąpi w przypadku nieoszacowania liczby wyjątków. Ciekawym przypadkiem jest sytuacja odwrotna, gdy założenie liczby wyjątków przeszacowuję faktyczną liczbę wyjątków. Dochodzi do przeciwnego efektu zwanego **swamping**, kiedy element bliskiego skupiska zostaje rozpoznany jako wyjątek




Rysunek 2: Efekt Maskowania: Wyjątek *x2* jest bardziej odstający, x2 może zamaskować wykrycie wyjątku *x1*

## Algorytm Chen-Liu

Praca Chung Chen i Lon-Mu Liu *"Joint Estimation of Model Parameters and Outlier Effects in Time Series"* dokumentuję algorytm analizy strumienia danych. Podstawowym celem badań było przedstawienia procedury wykrywania wyjątków, która uwzględnia możliwość istnienia fałszywych i zamaskowanych wyjątków. Dodatkowo była w stanie obliczyć wpływ wyjątków na model, oraz oszacować nowe parametry modelu.

Dzięki precyzyjnemu zdefiniowaniu czterech różnych typów wyjątków, które pojawiały się w poprzednich badaniach, staje się możliwe pełniejsze zrozumienie ich wpływu na dane badawcze. Określenie obliczonego wpływu staje się kluczową podstawą do przeprowadzenia korekty parametrów modelu oraz umożliwia dalszą analizę.

Poniższy przykład bazuję na modelu ARIMA postaci:

$$Y_t = \frac{\theta(B)}{\alpha(B)\phi(B)} \tag{2}$$

Procedura "Chen-Liu" przedstawia szereg czasowy w następujący sposób:

$$Y_t^* = Y_t + \omega\xi(B)I_t(t_1) \tag{3}$$

Gdzie:

- Funkcja $I_t$ przyjmuję wartość 1 kiedy występuję wyjątek w każdym innym wypadku jest równa 0.
- $\omega$ jest początkową wartością odchylenia
- $\xi(B)$ określa jak będzie kształtował się wpływ wyjątku w czasie.

Algorytm przyjmuję rozróżnia następujące wyjątki na następujące typy:

- *Additive Outlier* (AO): Efekt charakteryzuję się pojedynczą, nagłą anomalią.

$$AO : \xi(B) = 1 \tag{4}$$

- *Level Shift* (LS): Trwały, ciągła zmiana wartości.

$$LS : \xi(B) = \frac{1}{1-B} \tag{5}$$

- *Temporary change* (TC): Efekt słabnie w czasie. Dodatkowym parametrem jest $\delta$ która określa krzywiznę.

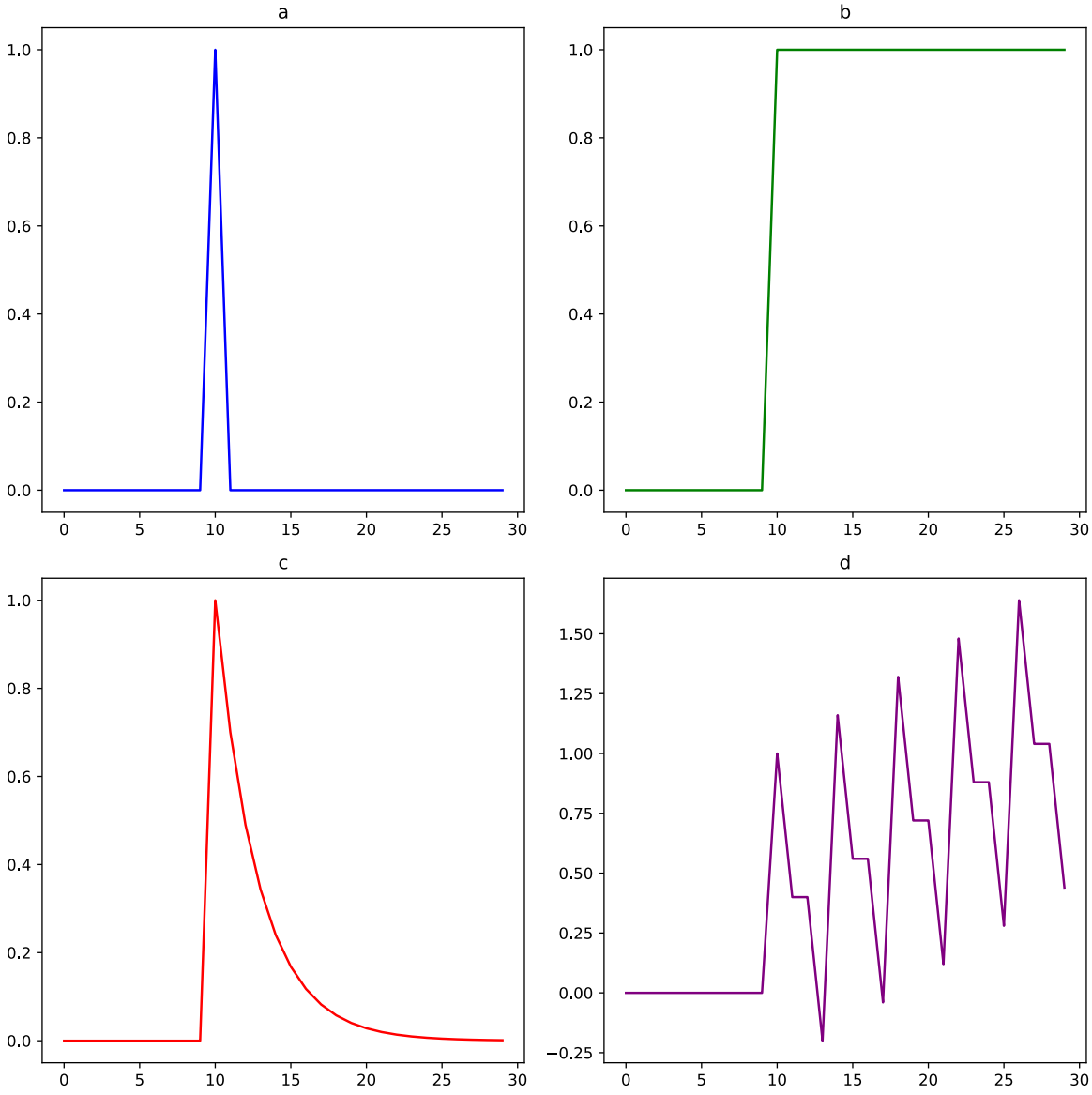
$$TC : \xi(B) = \frac{1}{1-\delta B} \qquad 0 < \delta < 1 \tag{6}$$

- *Innovational Outlier* (IO): Krzywa w czasie jest odzwierciedleniem modelu. W przypadku modelu ARMA wygląda następująco:

$$IO : \xi(B) = \frac{\theta(B)}{\alpha(B)\phi(B)} \tag{7}$$

W późniejszych pracach i implantacjach (**alsdkf?**) można napotkać na 5 typ wyjątków $SLS$. Ma za zadanie lepiej odwzorcowywać sezonowość szeregu czasowego niż typ IO, który nie musi dziedziczyć cech sezonowości z przyjętego modelu.

$$SLS : \xi(B) = 1/S \qquad S = 1 + B + ... + B^{s-1} \tag{8}$$




Rysunek 3: Porównanie efektów różnych wyjątków a) AO, b) LS, c) TC, d) IO ARIMA(0,1,1)(0,1,1)

Algorytm postępowania jest iteracyjny i jest podzielony na 3 oddzielne etapy. Przedstawione poniżej kroki algorytmu są uproszczone. Dokładny opis procedury można znaleźć w

oryginalnej pracy(**dupa?**):

1. Obejmuję wykrycie potencjalnych wyjątków. W tym celu dokonuję się dopasowania przyjętego modelu do serii danych i obliczenia odchyleń dla każdego punktu. W następnym kroku, dla każdego punktu i szukanego typu obliczane są statystyki $\tau$ i $\omega$. Jeśli statystyka $|\tau|$ w czasie $t$ jest większa niż przyjęta wartość krytyczna $C$ oznacza, że w tym punkcie wystąpił wyjątek. Jeżeli 2 lub więcej typów przekroczyła wartość krytyczną wybierany jest typ z największym współczynnikiem $\tau$. Następuję obliczenie efektów wykrytych wyjątków i usunięcie z serii danych. Poprawiona seria danych zostaję ponownie analizowana zgodnie z poprzednimi krokami, dopóki w iteracji nie zostanie wykryty żaden wyjątek, lub zostanie przekroczona ustalona liczba iteracji.

2. W tym etapie zostaję sprawdzony wpływ potencjalnych wyjątków. Do tego celu zostaje użyty model regresji obliczyć wielkość wyjątku $\hat{\omega}$. Obliczana jest ponownie $\tau_j$ korzystając ze wzoru: $\hat{\tau}_j = \hat{\omega}_j / std(\omega)$. Jeśli statystyka jest niższa niż wartość krytyczna $C$ wyjątek jest usuwany z listy potencjalnych. Pętla zostaję przerwana w przypadku braku wykrycia błędu lub przekroczenia liczby iteracji. Następuję kolejne dopasowanie modelu skorygowanej serii.

3. Ostatnim etapem jest powtórzenie I i II fazy algorymu wykorzystując nowe parametry modelu: W I fazie nie koryguję parametrów. W II fazie $\hat{\omega}$ jest końcową wartością.

## Wykrywanie wyjątków w systemach informatycznych

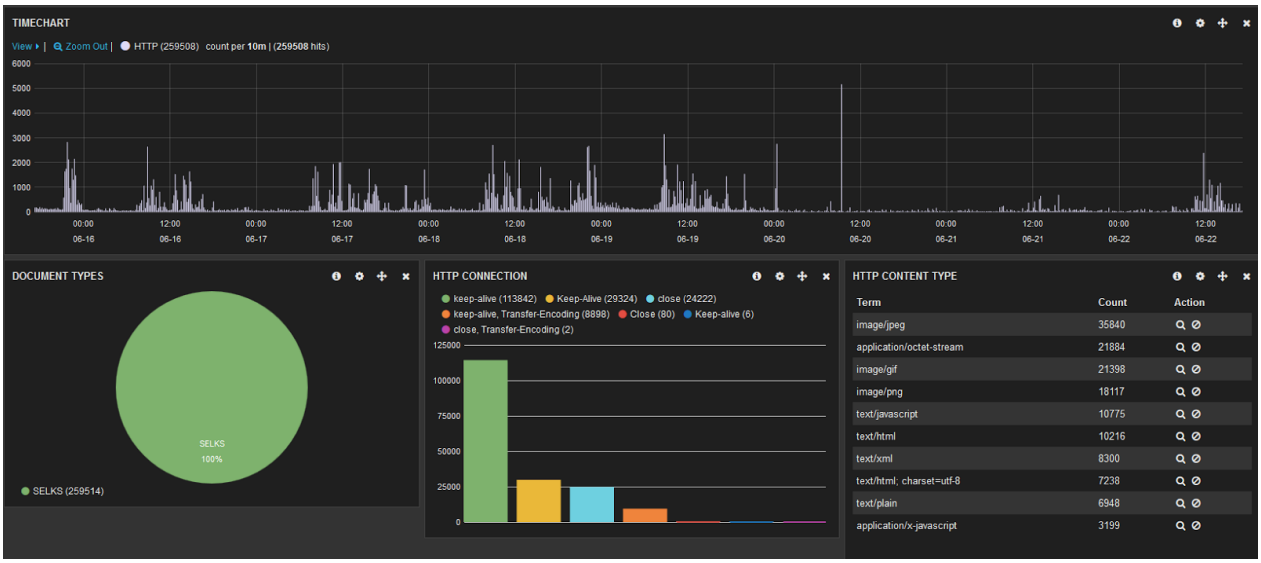
Algorytm "Chen-Liu" nie jest jedynym algorytmem wykrywania wyjątków. Na przestrzeni lat postało wiele metod stworzonych w tym celu. Przykładami takich algorytmów są:

- **Isolation Forest** Jedna z najnowszych metod wykrywania wyjątków. Metoda polega na wykorzystaniu drzew binarnych do losowego podziału serii danych. Implementacja jest dostępna w wielu językach programoawnia tj Python, R i dostępna dla platformy Apache Spark. (**iforest?**)

- **Auto enkodery** Grupa algorytmów oparta na sztucznej inteligencji. Główna idea stojąca za autoenkoderami polega na nauczeniu się skompresowanego przedstawienia lub kodowania danych wejściowych. Anomalie są wykrywane poprzez pomiar błędu rekonstrukcji między wejściem a odtworzonym wyjściem.

Istnieje dużo dziedzin gdzie wykrywanie wyjątków znalazło zastosowanie. Algorytmy są stosowanie w cyberbezpieczeństwie jako systemy **IDS** *(Intrusion Detection System).* Takie systemy mogą bazować na sztucznej inteligencji lub działać na zasadzie data-miningu.

Dzięki algorytmom, możliwe jest wykrywanie oszustw bankowych. Przykładem takich apli-

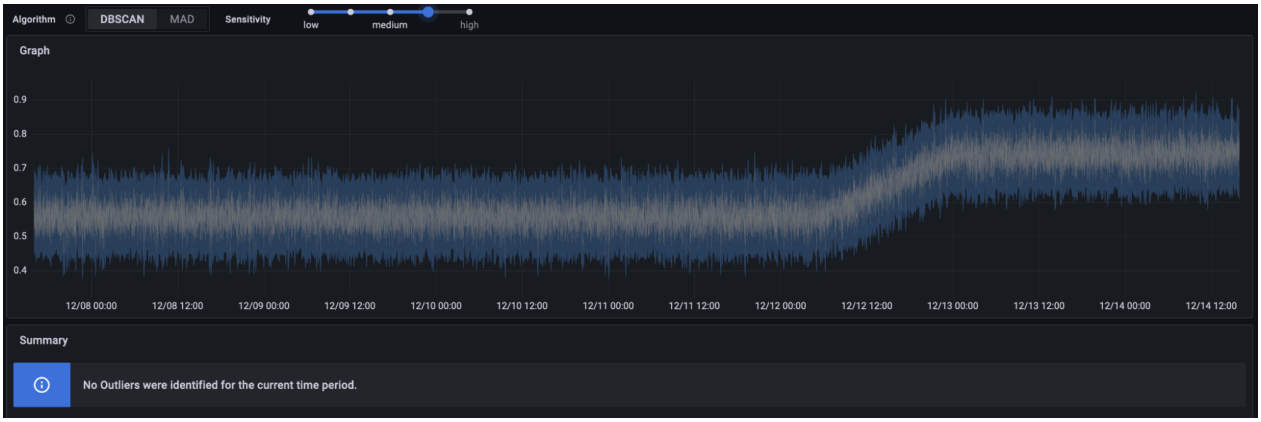



Rysunek 4: Przykładem systemu IDS jest Surikata

kacji jest “SEON”.

Wykrywanie wyjątków znalazło zastosowanie w systemach IOT, zarządzaniu infrastrukturą IT jak i mikro serwisów. Algorytmy stają się pomocne przy wykrywaniu usterek sprzętu. Wczesne wykrycie problemów z wydajnością systemów zwiększa jakość, efektywność i stabilność samego systemu jak i biznesu.

Przykładem oprogramowania służącej do monitorowania zasobów aplikacji / klastra jest Grafana, która daję możliwość podglądu na żywo statystyk CPU, pamięci, transferu internetowego. Grafana udostępnia płatnym użytkownikom wykrywanie wyjątków w wybranych strumieniach i zintegrowanie z systemem powiadomień.(3)




Rysunek 5: Konfiguracja detekcji wyjątków w programie grafana

Istnieją rozwiązania w postaci bibliotek. Sektorem, w którym wykrywanie wyjątków jest

szeroko stosowane są media społecznościowe. Firmy takie jak Meta *(dawniej Facebook)* czy X *(dawniej Twitter)* udostępniają kod swoich bibliotek przeznaczone do analizy danych i wykrywania wyjątków (**twitter-docs?**) (**prophet-docs?**).

Popularnymi bibliotekami w języku python są scikit-learn i TODS. Zaletą bezpośredniego użycia metody jest elastyczność rozwiązania, jednak wymagają pracy specjalistów w dziedzinie analizy danych (4) (5).

## Podsumowanie

Analiza strumieni danych jest popularnym i prężnie rozwijanym tematem w obecnej stanie technologi informatycznych. Niewątpliwie algorytmy bazujące na sztucznej inteligencji otwierają kolejne kierunki rozwoju. Z tego powodu istnieje ryzyko wyparcia metod tradycyjnych na rzecz technologi uczenia maszynowego. Jednakże, przedstawiona metoda "Chen-Liu" wyróżnia się swoimi właściwościami, a wyniki w postaci klasyfikacji wykrywanych wyjątków, mogą wzbogacić analizę danych.

Język Python jest jednym z najpopularniejszych języków w dziedzinie analizie danych, dlatego implementacja zaprojektowana dla tego języka, może być najbardziej dostrzeżona i jednocześnie najprzydatniejsza dla społeczności.

# Metodologia i implementacja projektu

Ten rozdziej opisuję szczegóły implementowanego rozwiązania. Zostaną poruszone dogłębnie aspekty techniczne i wykorzystanej technologii. Zostanie też przedstawiony proces implementacji jak i etap wdrożenia aplikacji.

## Wymagania

Wymagania odnośnie aplikacji możemy podzielić na funkcjonalne i niefunkcjonalne:

Funkcjonalne:

- Algorytm musi przedstawiać wynik analizy w formie listy wykrytych wyjątków i statystyk modelu.
- Użytkownik może przetestować zbiór danych.
- Rozwiązanie udostępnia możliwość przesłania serii danych do analizy w postaci pliku csv.
- Użytkownik może prosić o wygenerowanie danego typu efektu.
- Serwis daję możliwość połączenia efektów w jedno rozwiązanie.

Niefunkcjonalne:

- Wyjątki muszą być kategoryzowane.
- serwis być odporna na znaczny ruch.
- serwis przedstawia swój stan zdrowia.
- serwis loguję kolejne kroki postępowania wg przyjętego formatu.
- aplikacja musi być bezstanowa aby była lepiej skalowalna.

## Wybór technologii

Główną technologią użytą do implementacji algorytmu jak i usługi jest python. Python jest doskonałym narzędziem do szybkiego tworzenia aplikacji. Dodatkowo Python jest popularnym językiem w społeczności analityków danych.

## Zastosowane narzędzia i technologie

## Implementacja algorytmu

## Moduł REST

## Chmura obliczeniowa

## Docker

## Wdrażanie aplikacji sieciowej

# Opis przeprowadzonych testów

## Pokrycie kodu

## Lokalne testy aplikacji

# Bibliografia


1. FOUNDATION, Python Software. Python Documentation [online]. 2023. [udostępniono 11.11.2023]. Pobrano: https://docs.python.org/.

2. BERISHA, Blend & MËZIU, Endrit. *Big Data Analytics in Cloud Computing: An overview*[1]. luty 2021. S.l.: s.n.

3. LABS, Grafana. Grafana Documentation [online]. 2023. [udostępniono 11.11.2023]. Pobrano: https://grafana.com/docs/.

4. PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A., MICHEL, V., THIRION, B., GRISEL, O., BLONDEL, M., PRETTENHOFER, P., WEISS, R., DUBOURG, V., VANDERPLAS, J., PASSOS, A., COURNAPEAU, D., BRUCHER, M., PERROT, M. & DUCHESNAY, E. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*. 2011, T. 12, s. 2825–2830.

5. LAI, Kwei-Herng, ZHA, Daochen, WANG, Guanchu, XU, Junjie, ZHAO, Yue, KUMAR, Devesh, CHEN, Yile, ZUMKHAWAKA, Purav, WAN, Minyang, MARTINEZ, Diego & HU, Xia. TODS: An Automated Time Series Outlier Detection System. *Proceedings of the AAAI Conference on Artificial Intelligence*. maj 2021, T. 35, nr 18, s. 16060–16062.