Politechnika Krakowska
im. Tadeusza Kościuszki

WYDZIAŁ INFORMATYKI I TELEKOMUNIKACJI
ANALITYKA DANYCH

SPRAWOZDANIE Z LABORATORIUM 1

# OpenMP

*Autorzy:*

*Andrii Cherevko*

*Grzegorz Bogdał*

Prowadzący:
mgr inż. Wojciech Książek

13 października 2019

# 1 Zbiór danych

Wybrany przez nas zbiór danych to - Wine Quality Data Set (`https://archive.ics.uci.edu/ml/datasets/Wine+Quality`). Zbiór miesci w sobie dane o jakości białego i czerwonego wina. Oryginalnie jest podzielony na 2 pliki :

- winequality-red.csv
- winequality-white.csv

W pierwszym pliku z próbkami czewonego wina znajduje się: 1600 próbek. W drugim pliku z próbkami białego wina jest ich 4899. Podczas wykonania pracy nad zbiorem stworzyliśmy plik train_signals.csv w którym są połączone cechy białego i czerwonego wina oraz plik train_labels.csv który zawiera w sobie klasy wina. Następnie losow wymieszaliśmy wiersze.

Oryginalny zbiór danych to próbki win portugalskich „Vinho Verde". Pojedyncza próbka jest opisana za pomocą 11 cech oraz oceny jakości wina (quality). Ocena jakości wina jest zmienną zależną. Cechy zapisane są w postaci zmiennoprzecinkowej w pliku CSV. Cechy opisujące wino to:

1. kwasowość stała
2. kwasowość lotna
3. poziom zawartości kwasu cytrynowego
4. zawartość cukru resztkowego
5. zawartość chlorków
6. zawartość wolnego ditlenku siarki
7. zawartość ditlenku siarki ogółem
8. gęstość
9. pH
10. poziom zawartości siarczanów
11. poziom zawartości alkohol

# 2 Parametry komputera

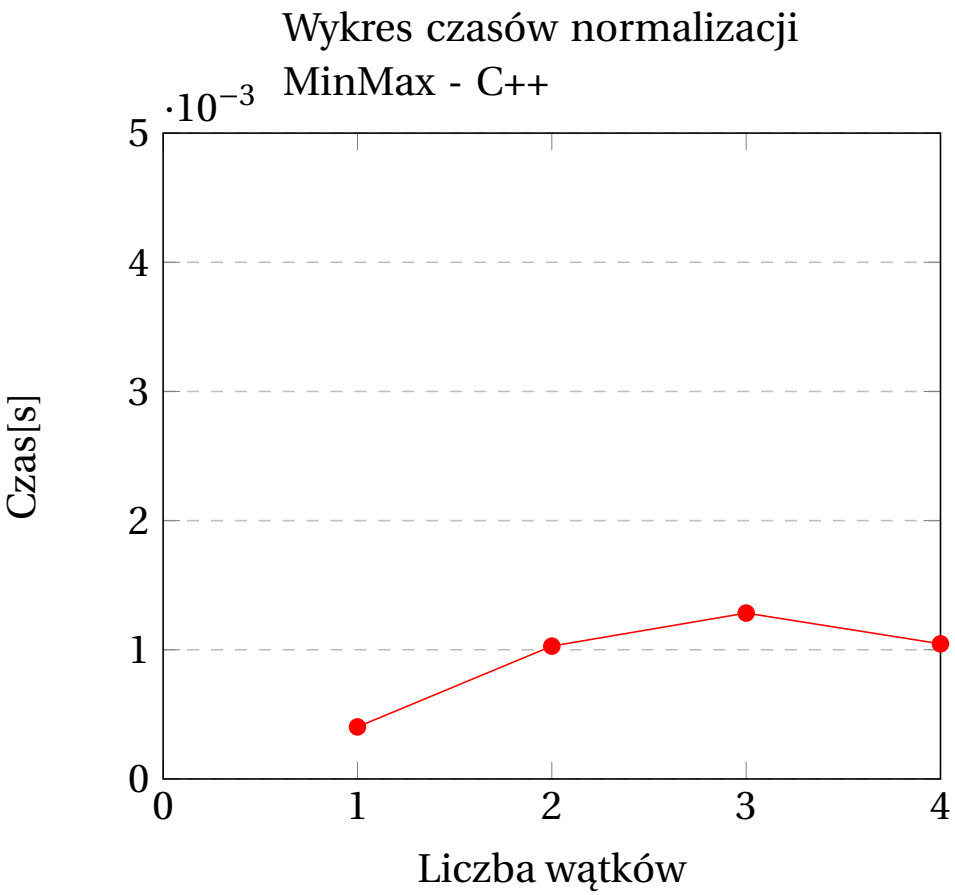
Laboratorium zostało wykonane na komputerze o następującej konfiguracji:

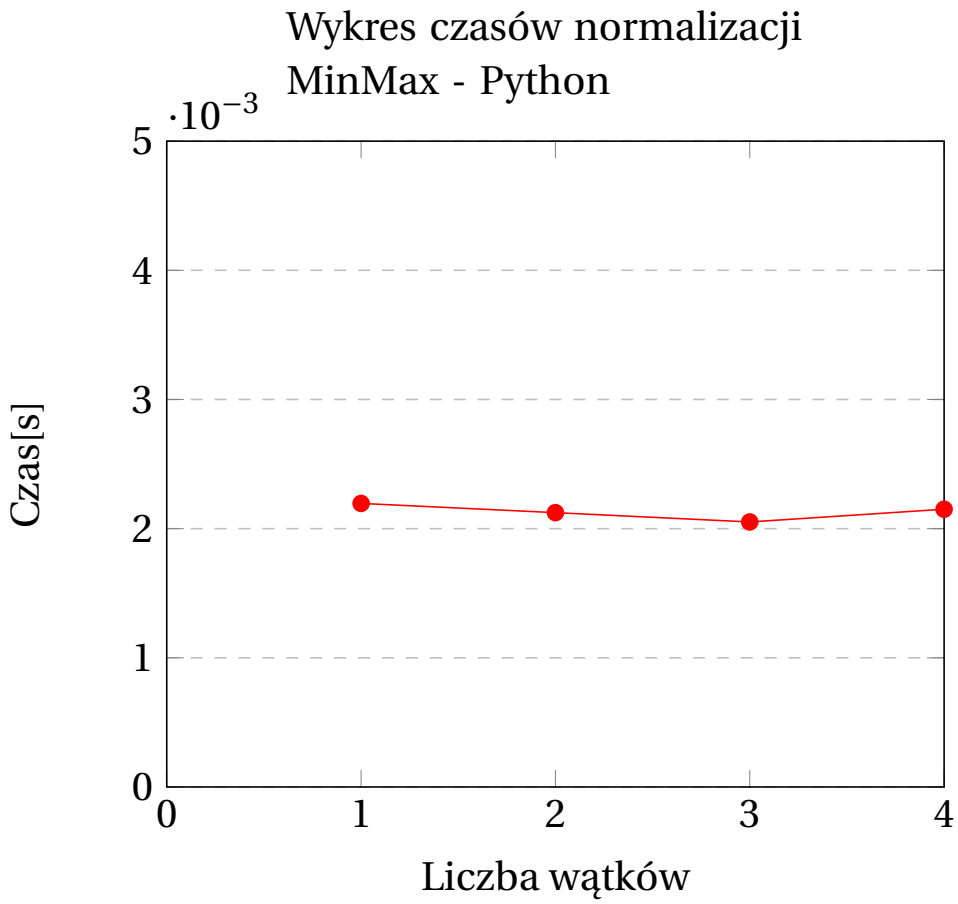
| Komponent | Opis |
|---|---|
| OS | Arch Linux |
| Kompilator | gcc |
| CPU | AMD A10-6800K, 4 rdzenie fizyczne, brak hyperthreadingu |
| RAM | 16 GB 1866MHz DDR3 |

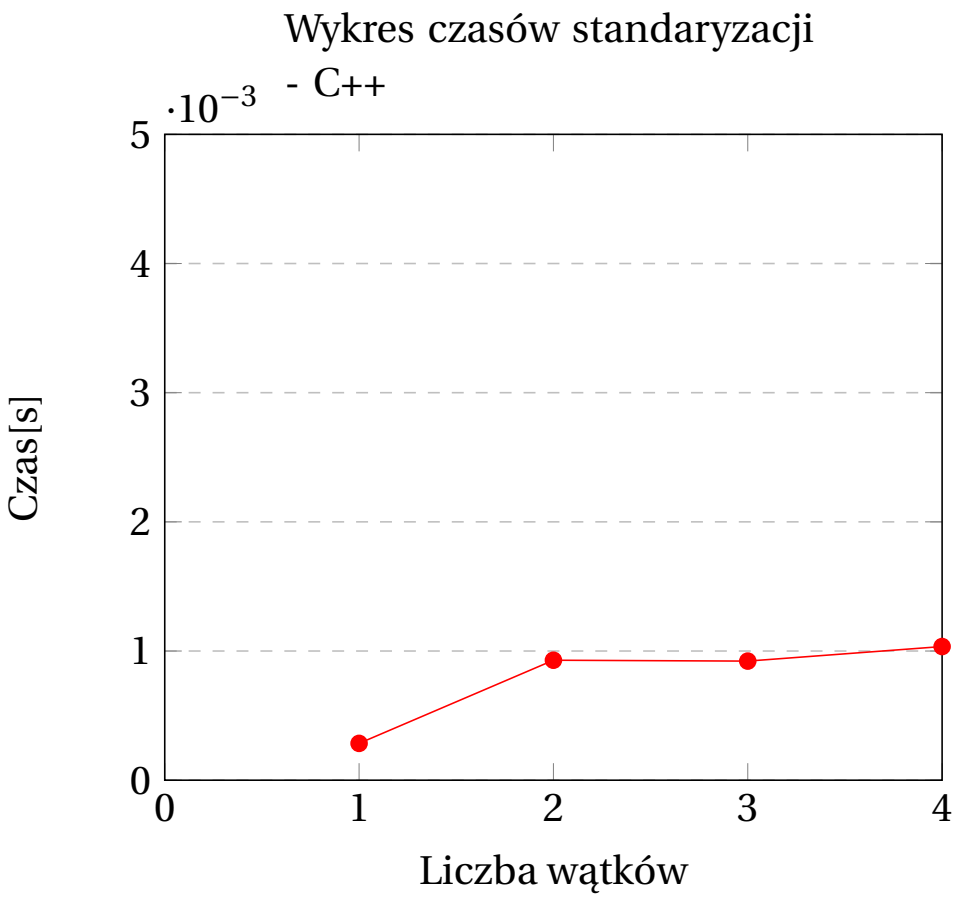
# 3 Metodologia

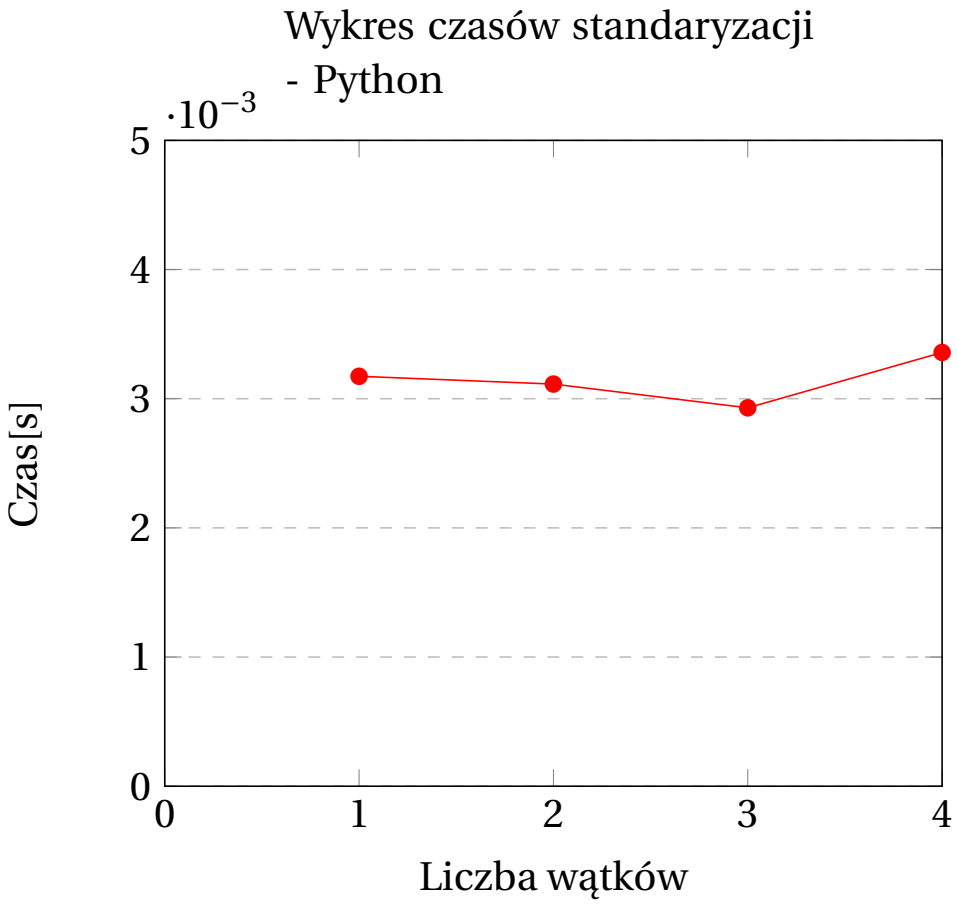
W języku C++ zaimplementowano normalizację MinMax, standaryzację i algorytm knn dla k=1. Program zrównoleglono używając standardu OpenMP w kompilatorze gcc. Analogiczny program napisano w języku Python przy pomocy bibliotek pandas i sklearn. Następnie zmierzono czasy w zależności od liczby rdzeni dziesięciokrotnie i uśredniono zaprezentowane poniżej wyniki.

## 4 Wykres czasów normalizacji

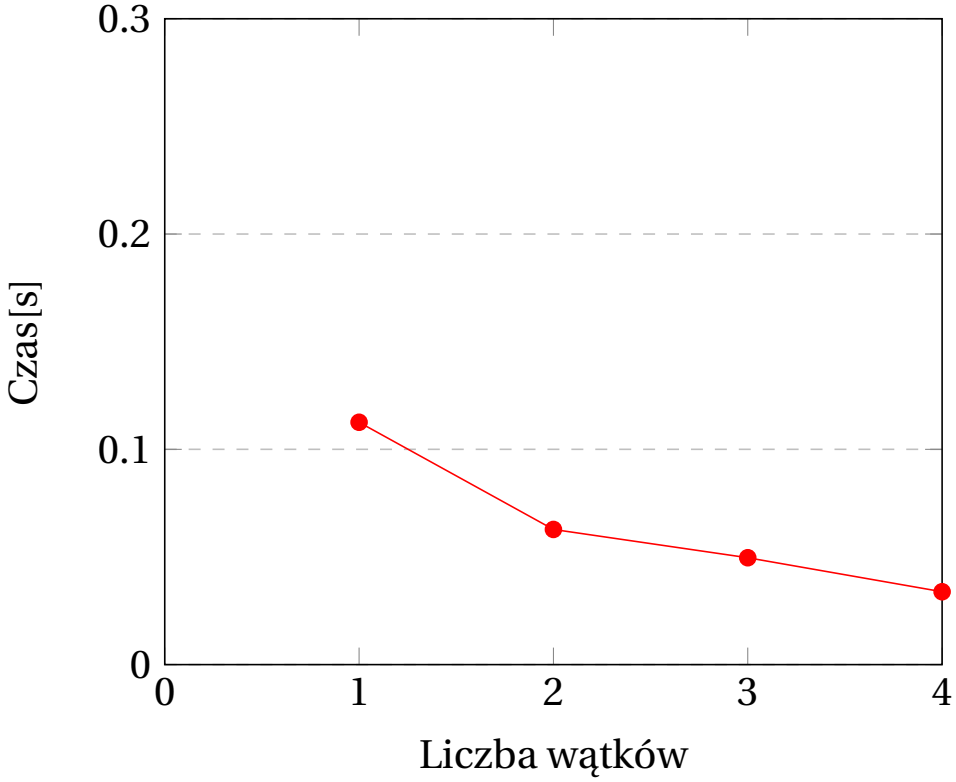


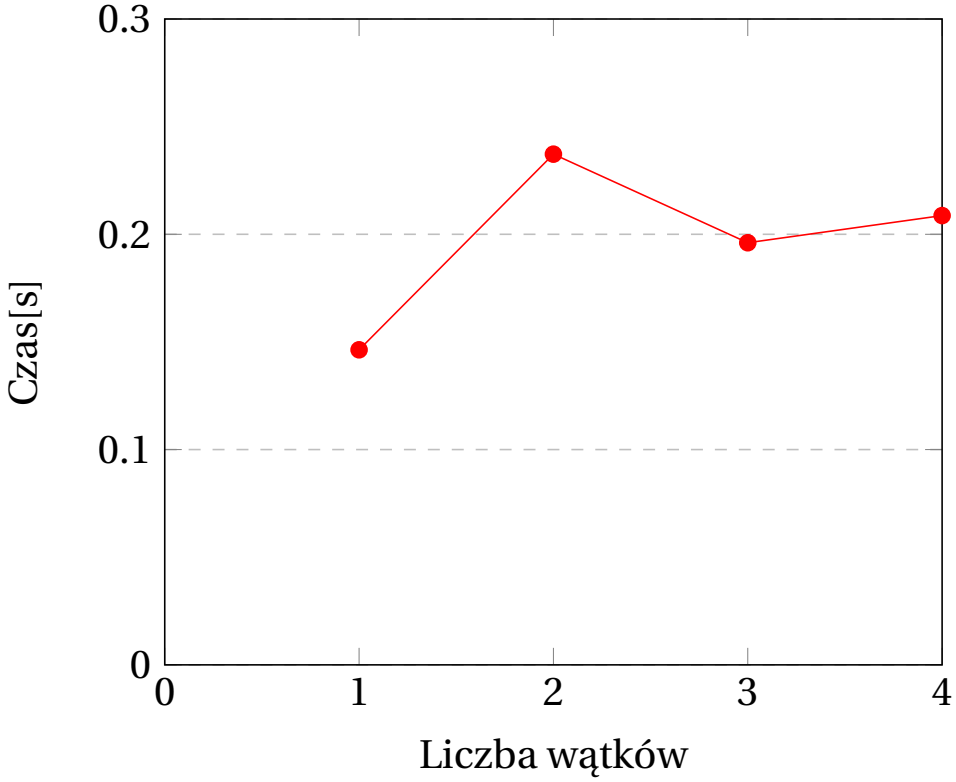

Wykres czasów standaryzacji
- C++
$\cdot 10^{-3}$
5
4
3
2
1
0
Czas[s]
0
1
2
3
4
Liczba wątków

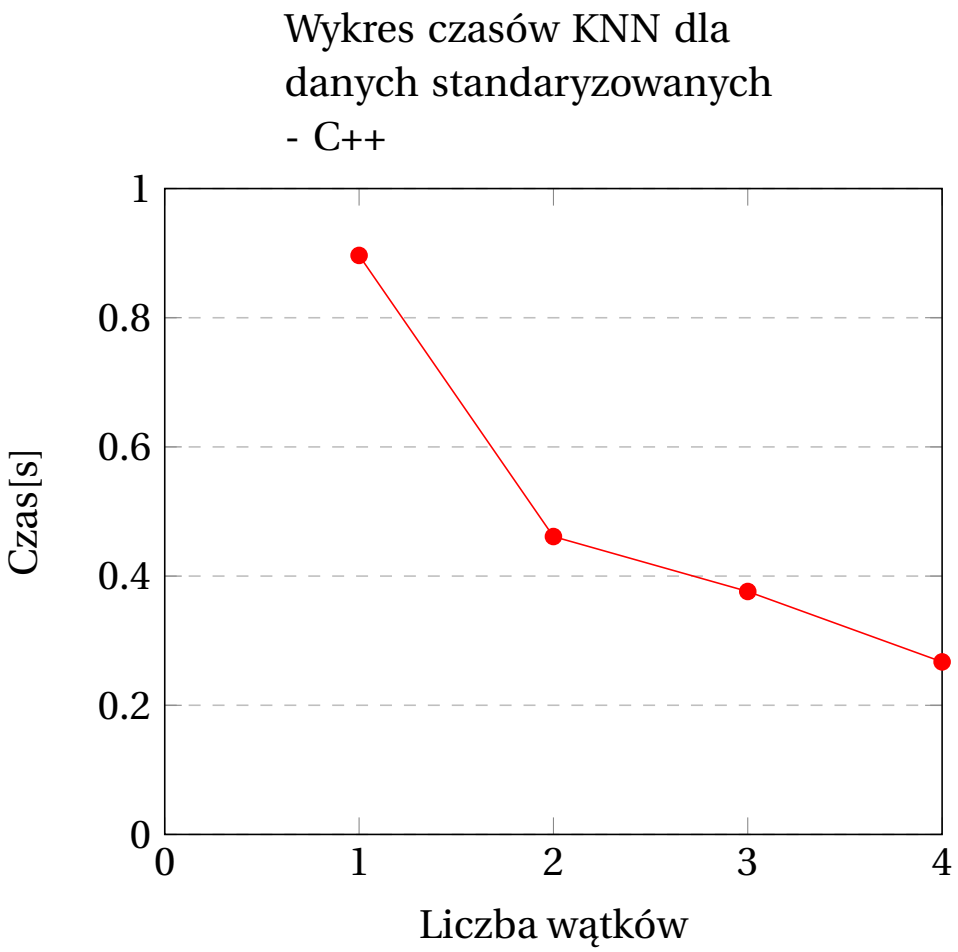

Wykres czasów standaryzacji
- Python
$\cdot 10^{-3}$
5
4
3
2
1
0
Czas[s]
0
1
2
3
4
Liczba wątków

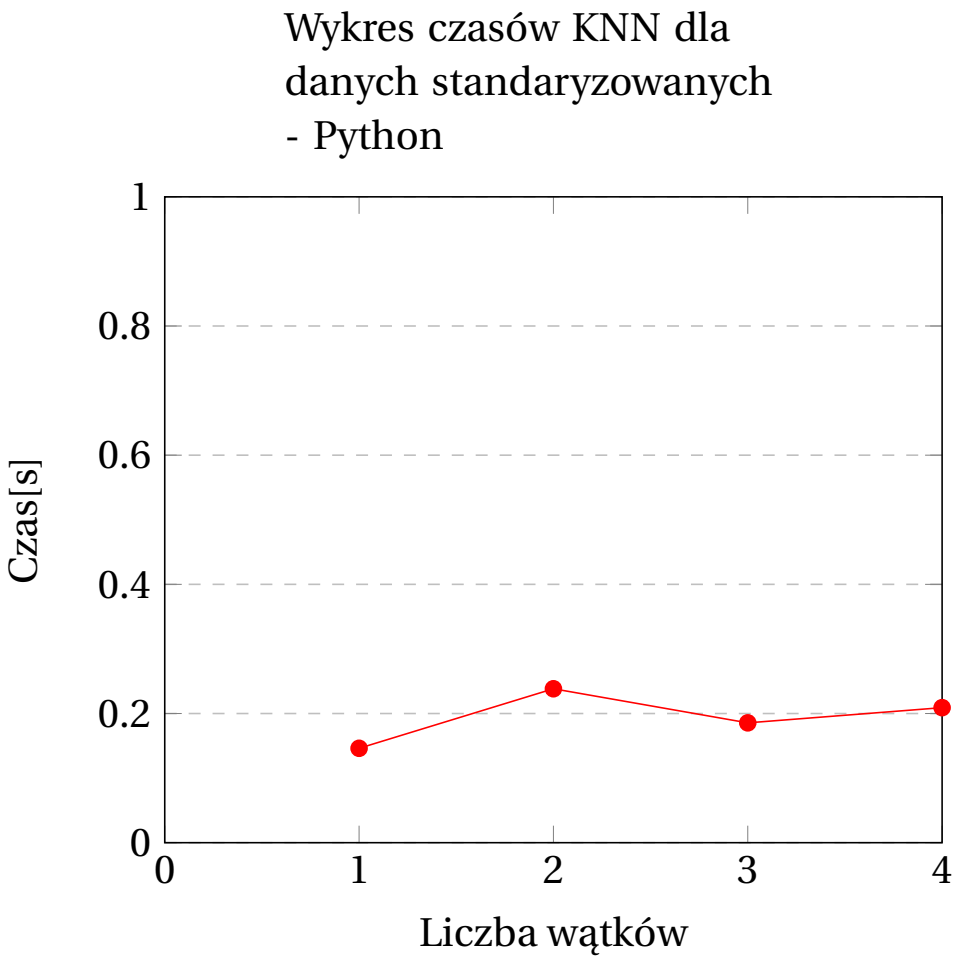
Wykres czasów KNN dla danych znormalizowanych metodą MinMax - C++




Wykres czasów KNN dla danych znormalizowanych metodą MinMax - Python






***Uwaga:*** Wykresy bazują się na średniej z czasów dla każdej z metod z 10 powtórzeń

# 5 Podsumowanie

Biblioteka sklearn nie oferuje opcji zrównoleglenia normalizacji i standaryzacji. Implementacja w języku C++ staje się wolniejsza wraz z liczbą wątków. Implementacja autorów w języku C++ jest szybsza. Przypuszczalnie z uwagi na szybką implementację i mały zbiór danych narzut związany z utworzeniem wielu wątków neguje korzyści związane z równoległości, które uwidoczniłyby się na większym zbiorze.

W przypadku zrównoleglenia algorytmu kNN w języku C++ można zobaczyć znaczne przyspieszenie czasów przy zwiększeniu ilości wątków. W implementacji autorów wszystkie wątki przeszukują jeden zbiór danych bez kopiowania go, przez co narzut związany z równoległością jest minimalny. Dla danych znormalizowanych metodą MinMax implementacja autorów jest szybsza od implementacji biblioteki sklearn. W przypadku implementacji kNN w Python zrównoleglenie spowalnia wykonanie programu. Czas wzrasta gdy pojawia się drugi wątek, kolejne wydają się nie mieć wpływu. Autorzy sprawozdania nie rozumieją tego zachowania.