

Co telefony wiedzą o tobie, projekt z klasteryzacji - raport

Autorzy: Mateusz Deptuch, Paweł Florek

0. Spis treści

1. [Dane](#)
2. [EDA](#)
3. [Preprocessing danych i feature selection](#)
4. [Modelowanie](#)
5. [Wyniki](#)
6. [Wnioski](#)

1. Dane

Dane użyte w projekcie pochodziły ze strony kaggle, <https://www.kaggle.com/datasets/uciml/human-activity-recognition-with-smartphones>. Celem projektu była klasteryzacja danych zebranych z czujników znajdujących się w telefonach osób testowych (później *subject*), których było 30. Wykonywali oni codzienne aktywności mając przy sobie swoje telefony z aktywnymi czujnikami (akcelerometr, żyroskop). Założeniem projektu na kaggle była klasyfikacja aktywności do jednej z części kategorii, jednakże nie posiadaliśmy takowej informacji z powodu innego celu naszej pracy.

Sygnały jakie przechwytywały czujniki były następujące:

- tBodyAcc
- tGravityAccMag
- tBodyGyroJerkMag
- tBodyAccJerk
- tBodyGyro
- tBodyAccJerkMag
- tBodyGyroJerk
- tBodyAccMag
- tGravityAcc
- tBodyGyroMag
- fBodyAcc
- fBodyBodyGyroMag
- fBodyAccJerk
- fBodyBodyAccJerkMag
- fBodyAccMag
- fBodyGyro
- fBodyBodyGyroJerkMag,

w trzech osiach X, Y, Z. Przedrostek 't' oznacza, że dany sygnał został zapisany w domenie czasowej, natomiast przedrostek 'f' oznacza zapis w domenie częstotliwości po zastosowaniu Szybkiej Transformacji Fouriera. Zostały one zapisane przy użyciu funkcji:

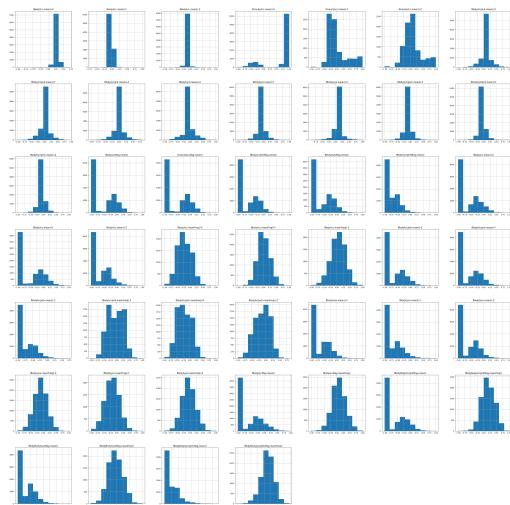
- max()
- mad()
- min()
- kurtosis()
- bandsEnergy()
- mean()
- meanFreq()
- arCoeff()
- entropy()
- iqr()
- sma()
- std()
- maxInds()
- skewness()
- energy()
- correlation().

2. EDA

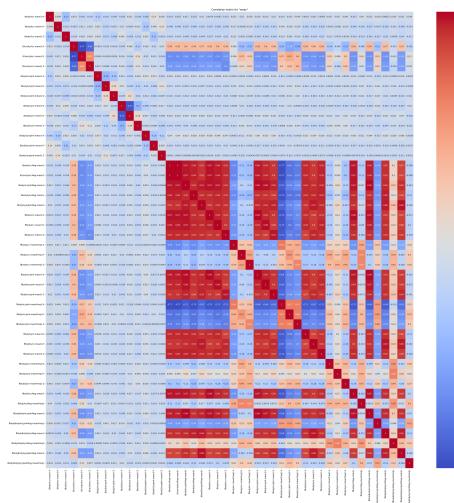
Przed rozpoczęciem pracy z przygotowywaniem danych oraz modelowaniem dokonaliśmy Eksploracyjnej Analizy Danych. Ramka danych po podziale zbioru na zbiór modelarzy oraz zbiór validatorów posiadała 10299 wierszy oraz 561 kolumn. Nie zawierała żadnych braków wartości oraz żadnych duplikujących się wierszy.

Dokonaliśmy analizy danych dla dwóch funkcji: mean oraz entropy.

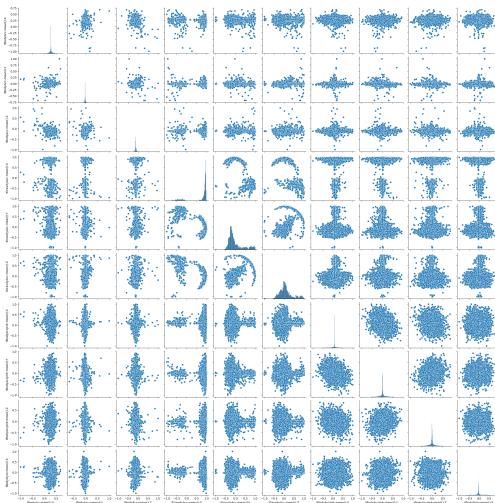
Analiza mean



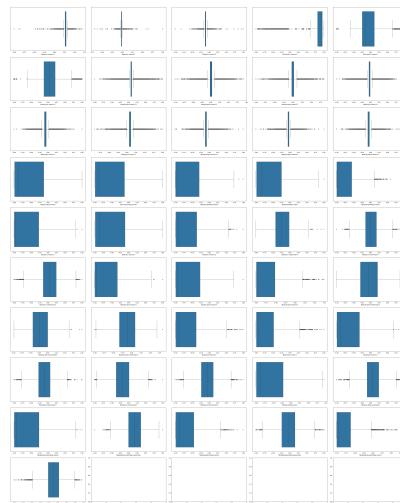
Dla mean obserwujemy dużo rozkładów.



Zauważamy duże korelacje niektórych kolumn. Będziemy mogli rozważyć usunięcie niektórych z nich w celu redukcji wymiarów naszych danych.



Pomimo bardzo dużo skorelowanych kolumn, na powyższych pairplotach jesteśmy w stanie zaobserwować dużą ilość podziałów na klastry.

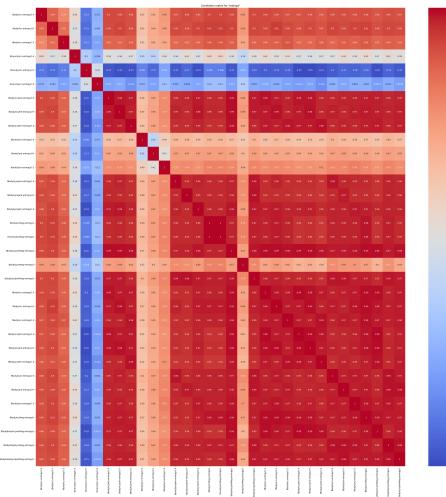


Nasze dane były poddane już obróbce, bowiem wszystkie wartości znajdują się w przedziale $[-1, 1]$, co potwierdza również poniższa tabela.

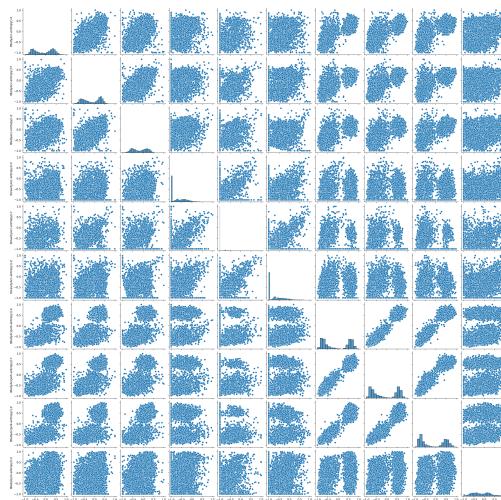
	tBodyAcc-mean()-X	tBodyAcc-mean()-Y	tBodyAcc-mean()-Z	tBodyAcc-std()-X	tBodyAcc-std()-Y	tBodyAcc-std()-Z	tBodyAcc-mad()-X	tBodyAcc-mad()-Y
count	10299.000000	10299.000000	10299.000000	10299.000000	10299.000000	10299.000000	10299.000000	10299.000000
mean	0.274347	-0.017743	-0.108925	-0.607784	-0.510191	-0.613064	-0.633593	-0.525697
std	0.067628	0.037128	0.053033	0.438694	0.500240	0.403657	0.413333	0.484201
min	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000	-1.000000
25%	0.262625	-0.024902	-0.121019	-0.992360	-0.976990	-0.979137	-0.993293	-0.977017
50%	0.277174	-0.017162	-0.108596	-0.943030	-0.835032	-0.850773	-0.948244	-0.843670
75%	0.288354	-0.010625	-0.097589	-0.250293	-0.057336	-0.278737	-0.302033	-0.087405
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

Ponadto nie musimy się przejmować wartościami odstającymi, ponieważ takowych nie posiadamy.

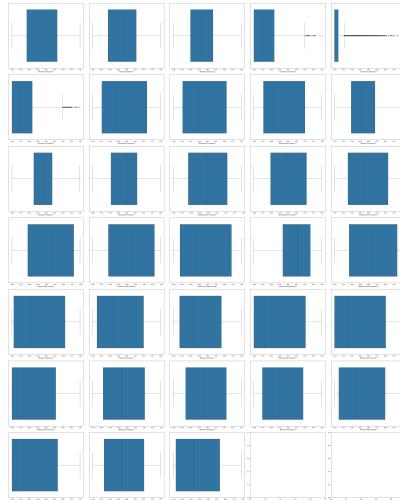
Analiza entropy



Ponownie obserwujemy bardzo dużo skorelowanych kolumn.

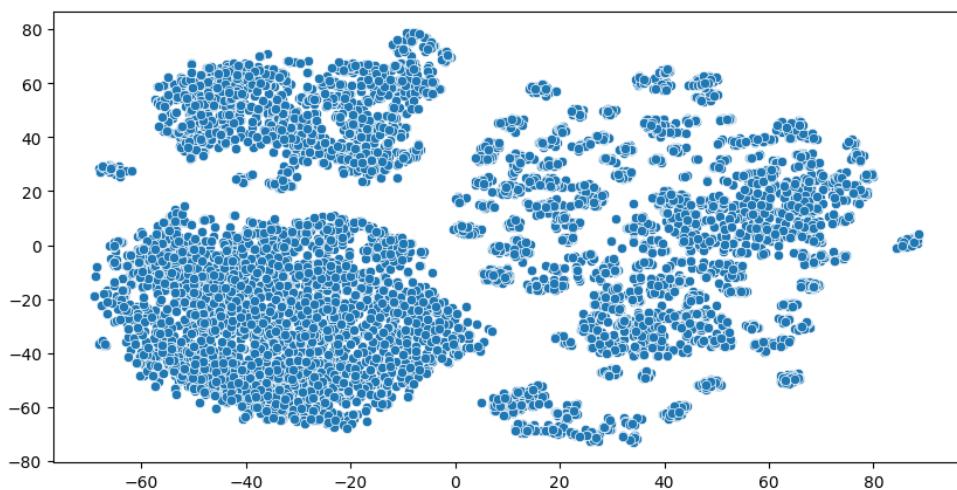


Możemy dostrzec powstanie paru ugrupowań.



Wizualizacja danych - tSNE

Na koniec EDY spójrzmy jak nasze dane wyglądają po zastosowaniu funkcji do redukcji wymiarów TSNE.



Łatwo dostrzec utworzenie się trzech klastrów, jednak nie możemy bazować na tej metodziej w celu ustalenia optymalnej liczby klastrów.

3. Preprocessing danych i feature selection

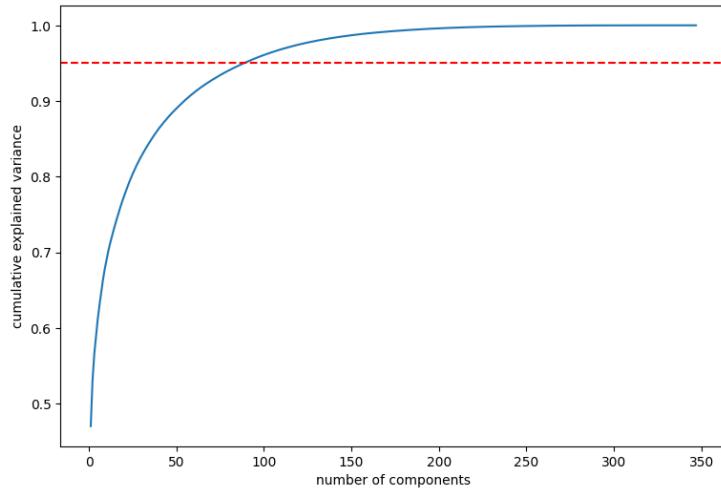
Nasze dane były już znormalizowane oraz nie zawierały żadnych braków. Nie było wartości odstających oraz danych kategorycznych. Zatem cały etap przetwarzania danych mogliśmy pominąć

Wybór cech

Od razu na początku pracy nad projektem usunęliśmy kolumny z informacją o rodzaju aktywności, ponieważ nie była nam potrzebna i mogła tylko przeszkodzić w pracy nad klasteryzacją. Podobny los spotkał kolumny o ID osoby testowej, gdyż nie niosło to żadnej użytecznej informacji.

Przejdzmy teraz do pozostałych kolumn. Jak ustaliliśmy w części EDA, nasze dane zawierają dużo skorelowanych kolumn. Po sprawdzeniu tego okazało się, że ustawiając nasz gorny limit korelacji na poziomie 0.98, mogliśmy pozbyć się 214 kolumn właśnie z powodu wysokiego skorelowania z inną kolumną. Tak też zostało uczynione, co doprowadziło do redukcji wymiaru danych do 347 kolumn.

Następnie zastosowaliśmy metodę PCA do redukcji wymiarów.

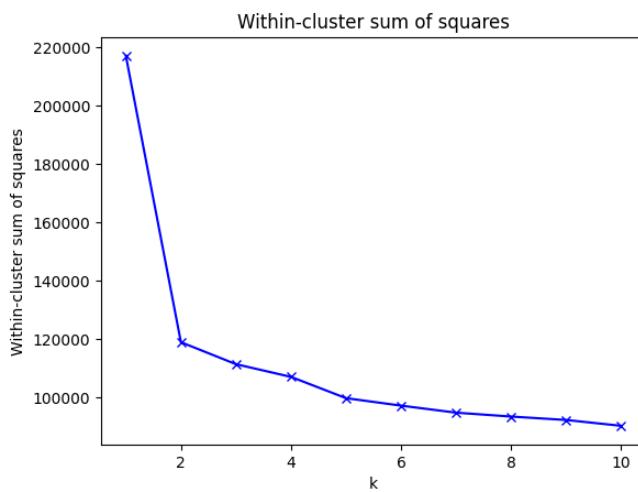


Zastosowaliśmy parametr $n_components = 0.95$, zachowując 95% wariancji. Dało to nam ostateczny wymiar 89 kolumn.

4. Modelowanie

KMeans

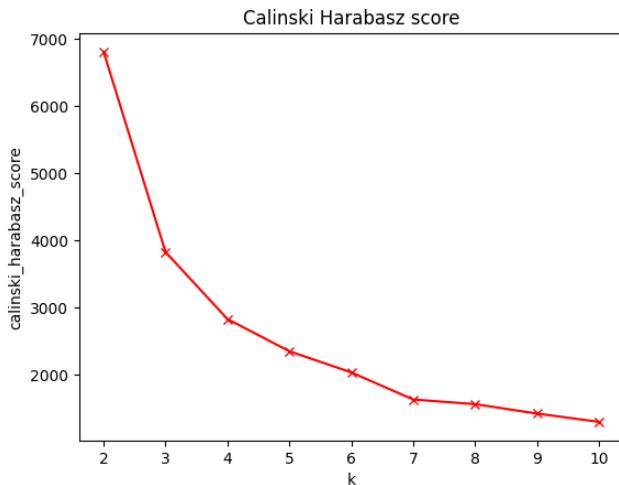
Pierwszym analizowanym modelem jest metoda KMeans. W celu określenia optymalnej liczby klastrów skorzystaliśmy z paru metod.



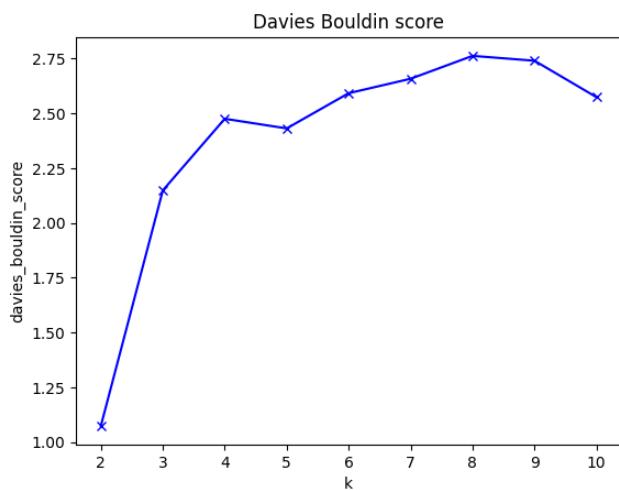
Na wykresie wyżej przedstawione są odległości wewnętrz danej liczby klastrów. Wykorzystując metodę łokcia ustaliliśmy, że najlepszą liczbą będzie dwa.



Chcąc uzyskać jak największy Silhouette Score, ponownie ustaliliśmy liczbę klastrów na dwa.

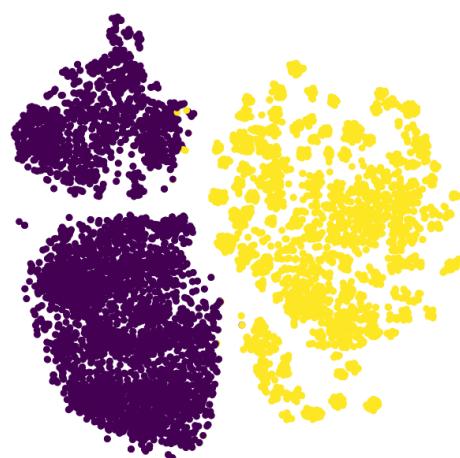


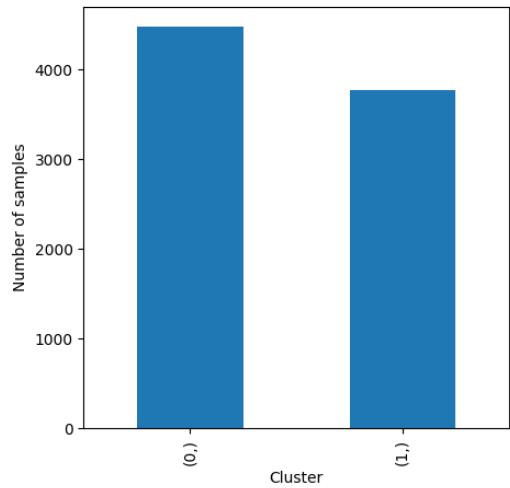
Ponownie chcemy zmaksymalizować Calinski-Harabasz Score, co jest osiągane przy wartości $n_clusters = 2$.



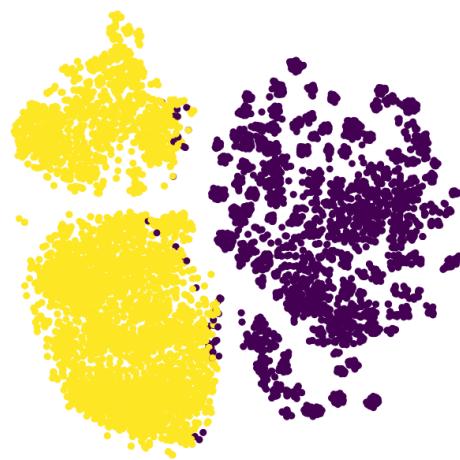
Tym razem minimalizujemy wyniki. Minimalna wartość *Davies-Bouldin Score* występuje dla dwóch klastrów.

Po uwzględnieniu powyższych wyników przeprowadziliśmy wizualizację danych przy użyciu t-SNE dla metody KMeans, ustawiając parametr $n_clusters = 2$.

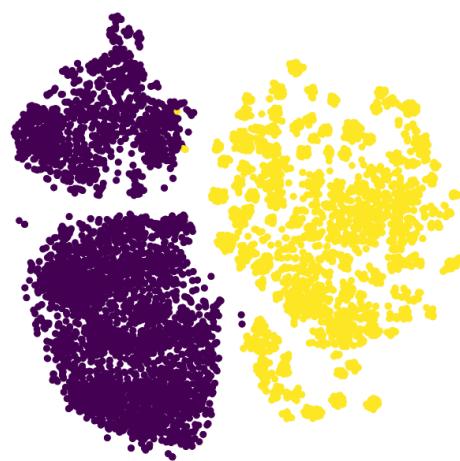




Dostaliśmy widoczny podział na dwa w miarę równe klastry. Jednakże na wizualizacji widać, że niektóre punkty z klastra oznaczone kolorem żółtym znajdują się pomiędzy punktami z klastra fioletowego. Mając to na uwadze sprawdziliśmy jeszcze dwie inne metody: *KMedoids* i *GMM*



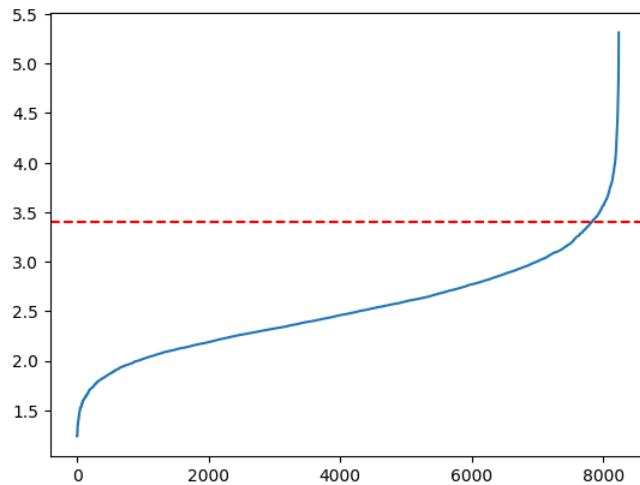
KMedoids



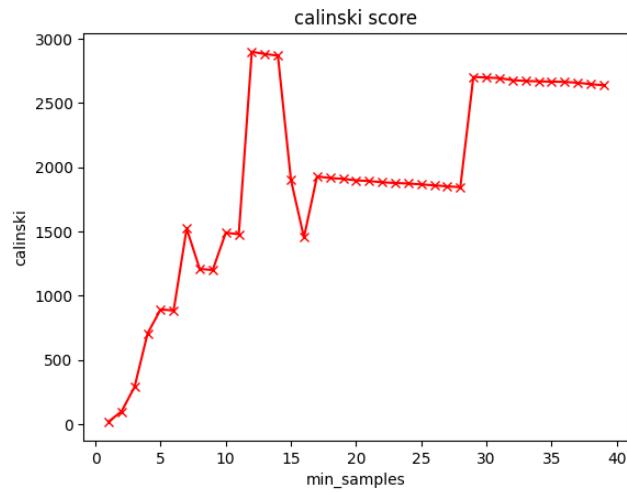
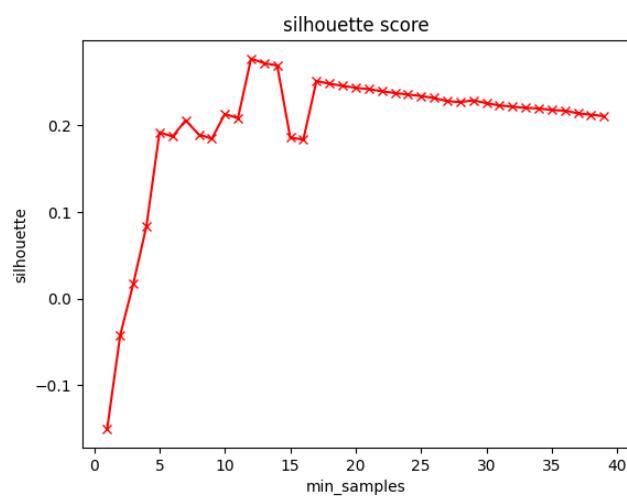
GMM

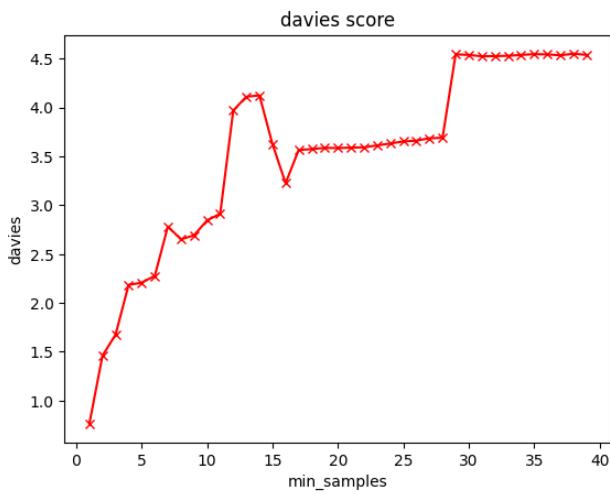
DBSCAN

W celu ustalenia optymalnego epsilona próbowaliśmy znaleźć moment, dla którego odległości zaczną gwałtownie rosnąć. Ustaliliśmy wartość 3.4

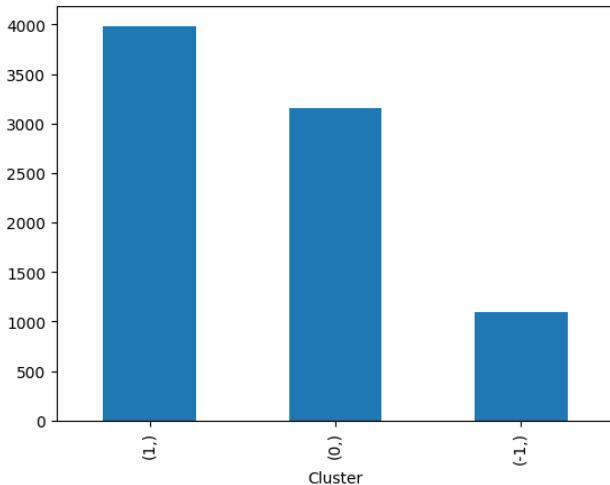


Następnie dla danego epsilona szukaliśmy dobrej wartości dla *min_samples*.





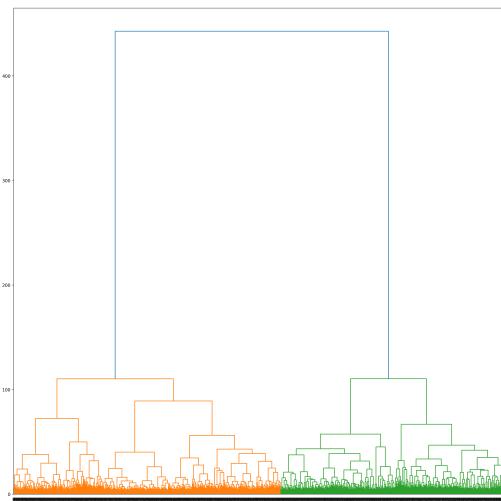
Analizując powyższe wykresy, ustawiliśmy wartość na 12. Po zastosowaniu powyższych parametrów otrzymaliśmy poniższe wyniki:



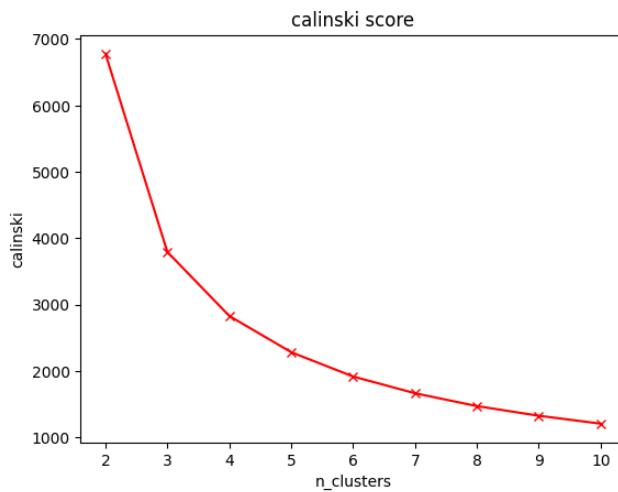
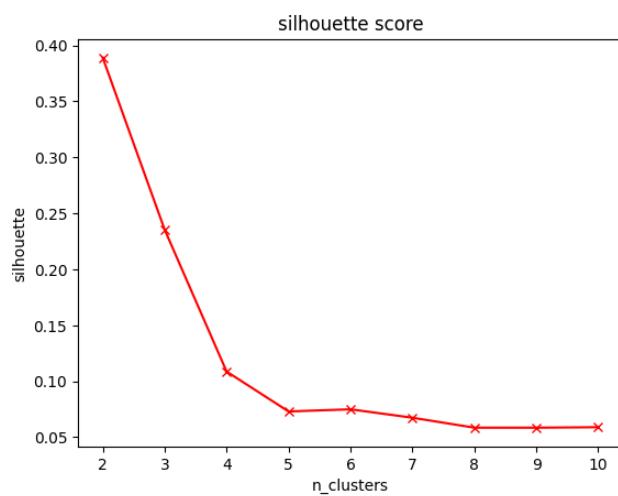
DBSCAN odrzuca aż 1000 wyników, co nie jest dla nas zadowalające. Są one zaznaczone kolorem fioletowym. Jest to na pewno duży problem, gdybyśmy zdecydowali się użyć modelu DBSCAN.

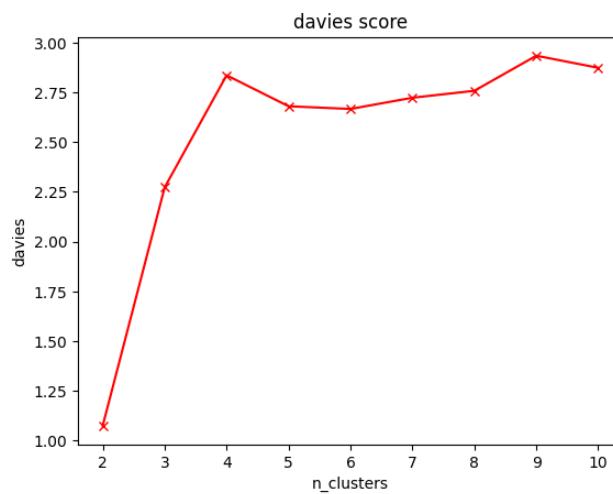
Agglomerative Clustering

Przejedźmy do metody grupowania aglomeracyjnego. Poniżej znajduję się wizualizacja, jak nasz model radzi sobie bez żadnych ustawiania hiperparametrów.

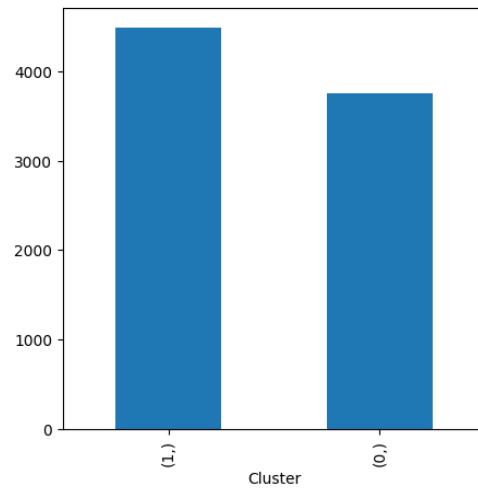


Ponownie sprawdziliśmy metryki w celu znalezienia optymalnej liczby klastrów.



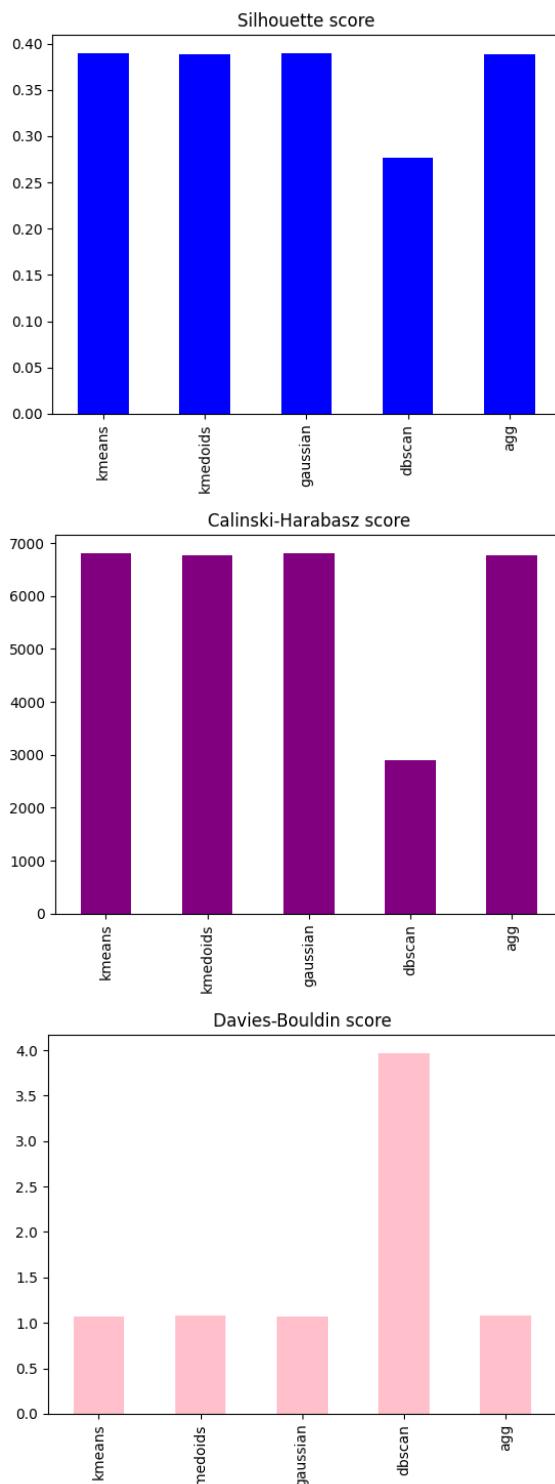


O to wyniki dla 2 klastrów, które zostały wybrane na podstawie powyższych wykresów.

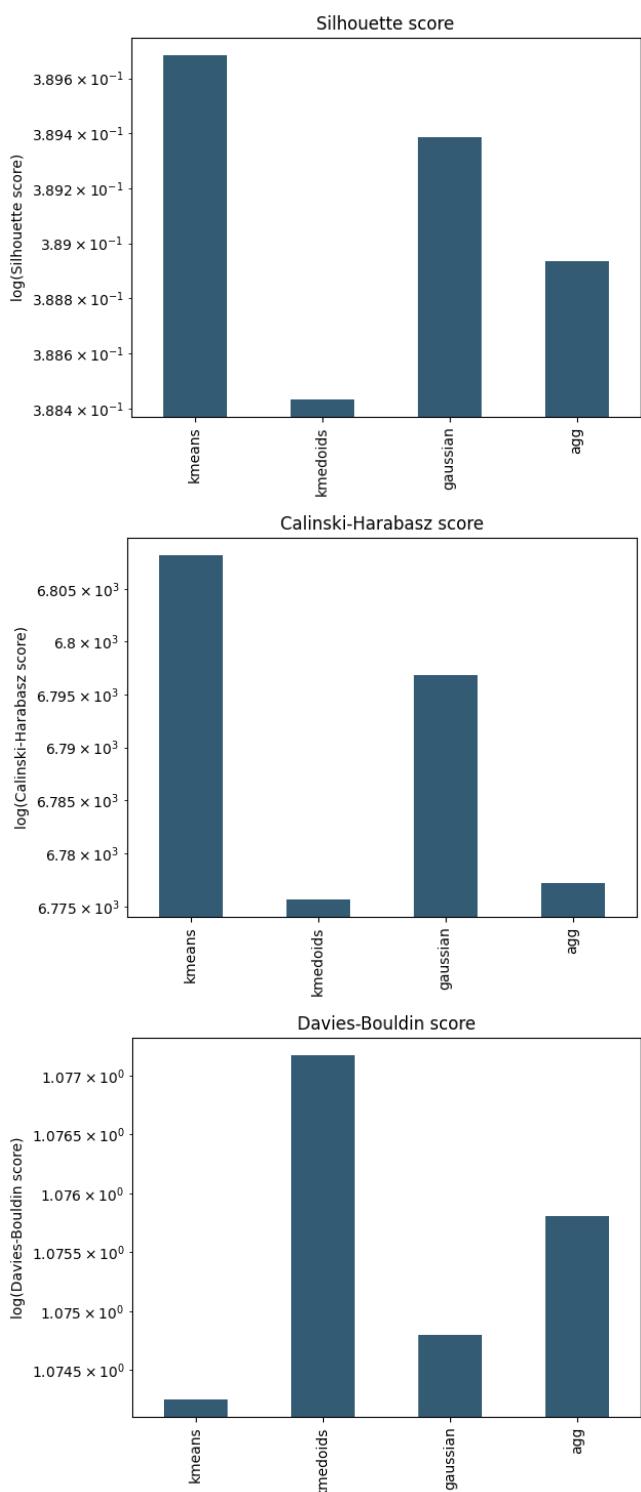


5. Wyniki

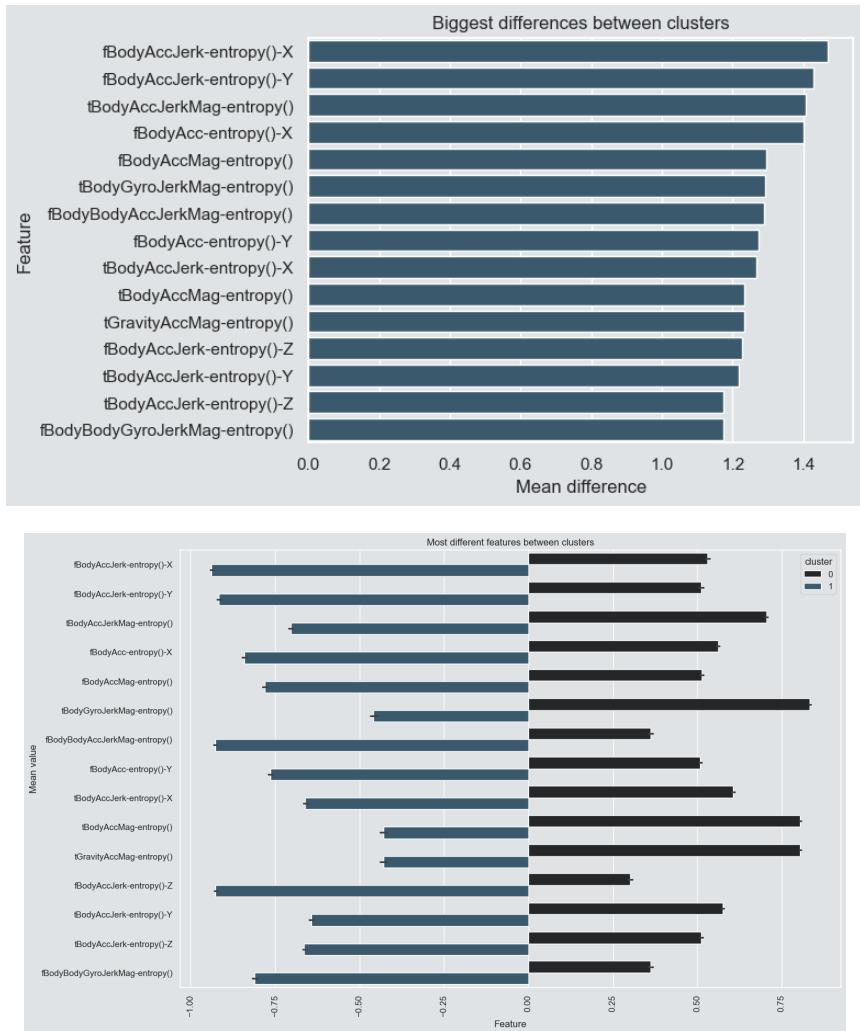
Mając wstępne wyniki modelowania przyszedł czas na wybór najlepiej sprawującego się modelu dla naszego problemu.



Powyzsze wykresy jednoznacznie wskazują na słabość DBSCANA. Zatem nie będziemy się już jemu dalej przyglądalni, ponieważ psuje nam skalę i porównanie pomiędzy innymi modelami.



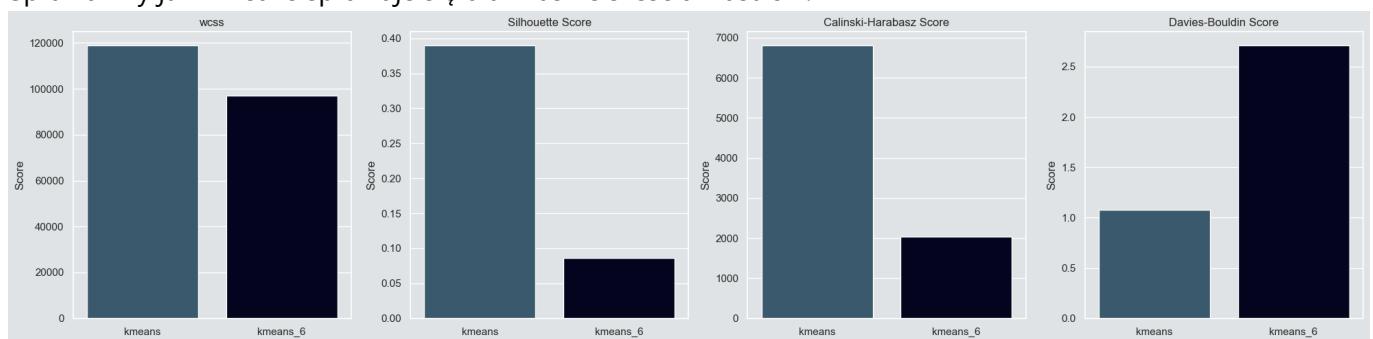
Widzimy, że najlepiej działa KMeans. Posiada on najwyższe wyniki w metrykach Silhouette'a i Calinskiego-Harabasza oraz najniższy w metryce Daviesa-Bouldina. Zostaje on zatem wybrany jako nasz główny model. Przyjrzyjmy się na czym nasz model rozróżnia, który element należy do którego klastera.

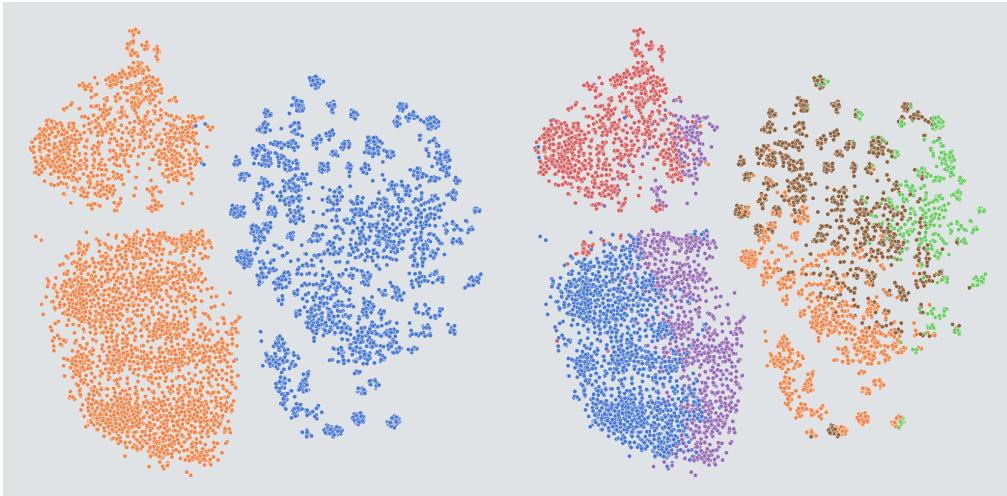


Jak można się było spodziewać, model stwierdza na podstawie sygnałów przyśpieszenia oraz zrywu, który klaszer pasuje dla danego elementu. Pokrywa się to z wizualizacjami, które pokazują dwa różne klastry. Mogą one reprezentować aktywny oraz bardziej spokojny tryb życia obiektów testowych

Sześć klastrów

Wracając do opisu naszej ramki, autorzy badania przydzieliły sześć różnych indeksów danym obserwacjom. Sprawdźmy jak KMeans sprawuje się dla właśnie sześciu klastrów.





Obserwujemy znaczące spadki w efektywności modelu. Obserwując wyniki danych metryk, możemy stwierdzić osłabienie nowego modelu oraz słabość konceptu sześciu klastrów. Na wizualizacji widać, że początkowe dwa klastry zostały wewnętrznie jeszcze bardziej podzielone. Może to cieszyć, ponieważ model nadal rozróżnia bardziej i mniej aktywny tryb życia. Możliwe że jest on w stanie rozróżnić inne aktywności, jednakże wyniki metryk nie są zachęcające.

6. Wnioski

Wybrany przez nas model KMeans dla dwóch klastrów efektywnie działa na danej ramce danych. Zalicza wysokie wyniki metryk oraz na wizualizacjach otrzymane klastry są łatwo rozróżnialne i dobrze wyszczególnione. Sposób rozróżniania obserwacji pokrywa się z intuicją, na podstawie której odzielilibyśmy aktywności na podstawie sygnałów o przyśpieszeniu i zrywie. Dla sześciu klastrów model nie spełnia naszych oczekiwów wypadając gorzej od naszej domyślnej wersji, pomimo tego że dane przewidują występowanie aż sześciu różnych aktywności obiektów testowych. Nie jest to jednak odzwierciedlone w naszym przypadku.