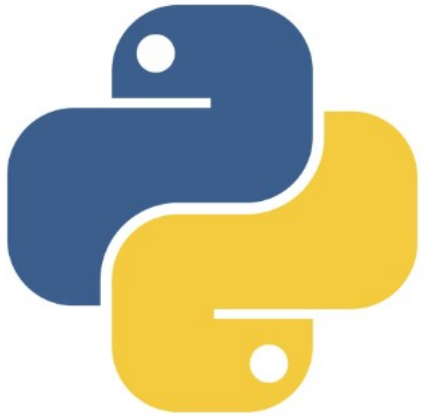
Wprowadzenie do uczenia maszynowego

dr inż. Marcin Nowak
marcin.nowak@poznan.merito.pl

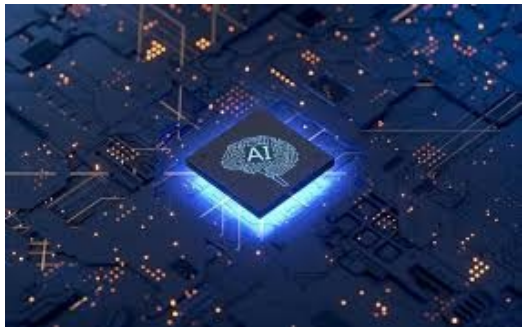


Algorytm *k*-nn

Algorytm ten, mimo że został opracowany w 1957 r., to został opublikowany dopiero w 1982 r. (Lloyd, s. 127-137).

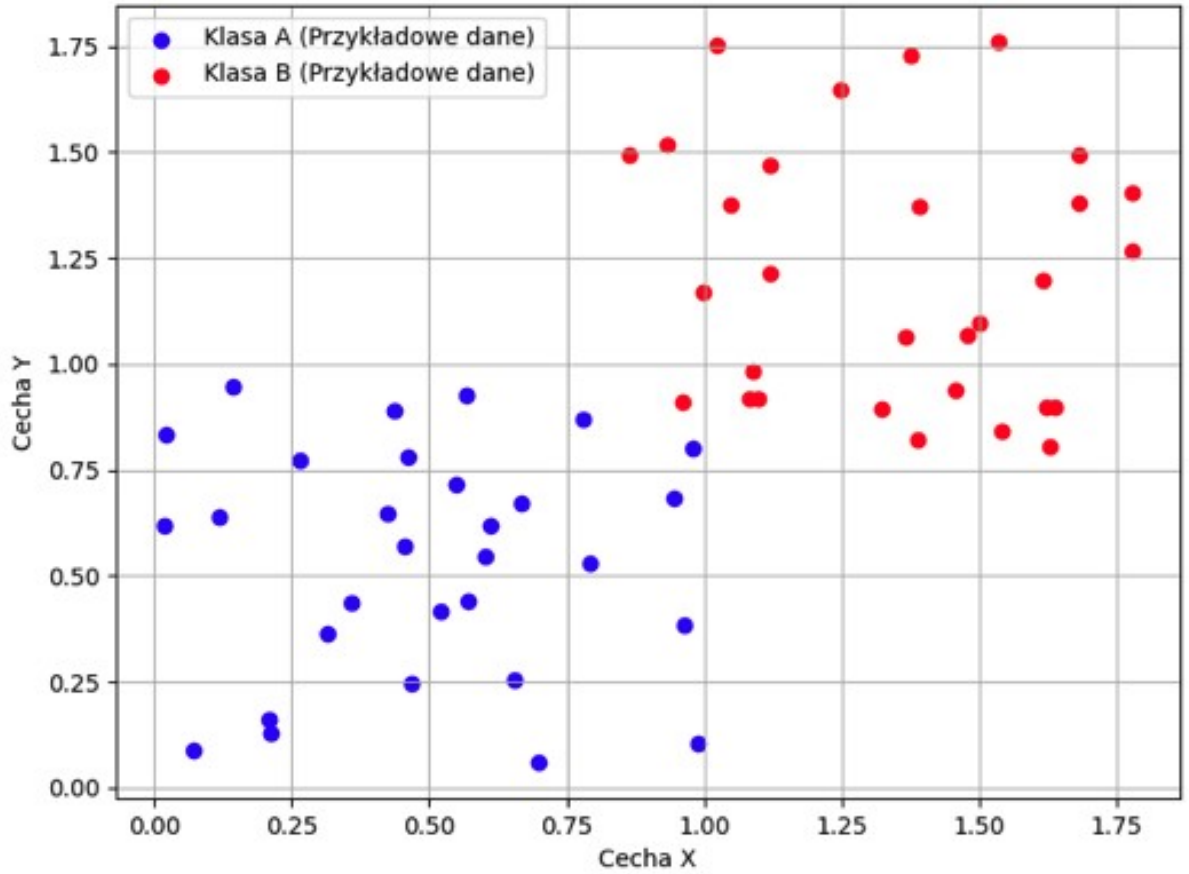
Algorytm *k*-najbliższych sąsiadów (*k*-nearest neighbours) – jeden z najprostszych i jednocześnie najbardziej intuicyjnych algorytmów uczenia maszynowego. Jego działanie opiera się na analizie odległości między punktami w przestrzeni cech.

Kiedy chcemy zaklasyfikować nowy punkt danych, *k*-NN porównuje go z *k* najbliższymi sąsiadami z zestawu treningowego, a następnie przypisuje mu etykietę, która najczęściej występuje wśród tych sąsiadów (w przypadku klasyfikacji) lub oblicza średnią wartość (w przypadku regresji).



https://www.forbes.com/sites/bernardmarr/2020/06/22/10-wonderful-examples-of-using-artificial-intelligence-ai-for-good/

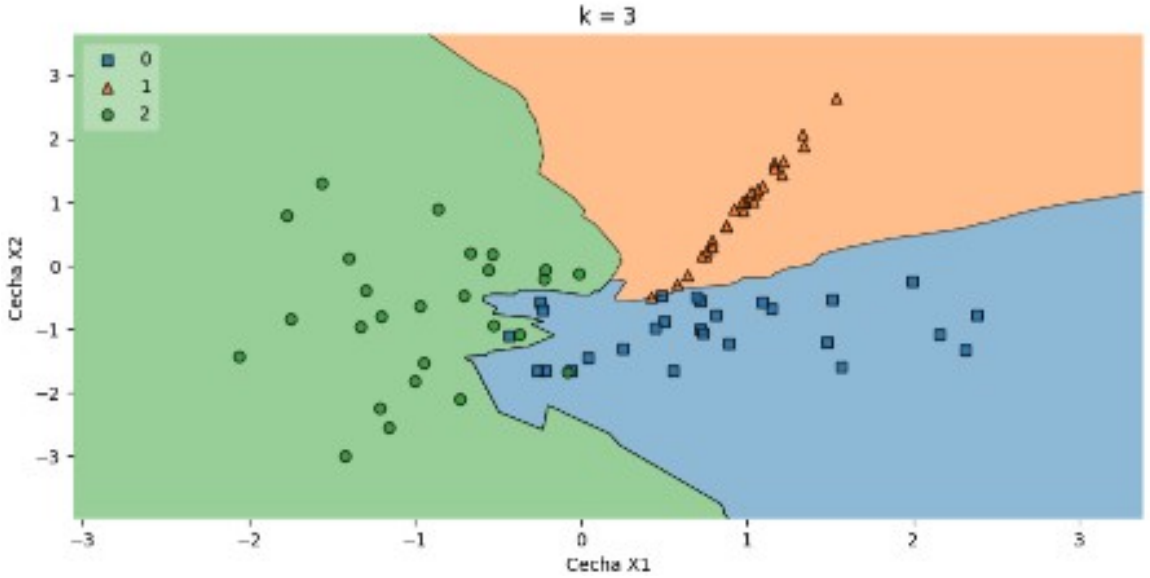
Algorytm *k*-nn

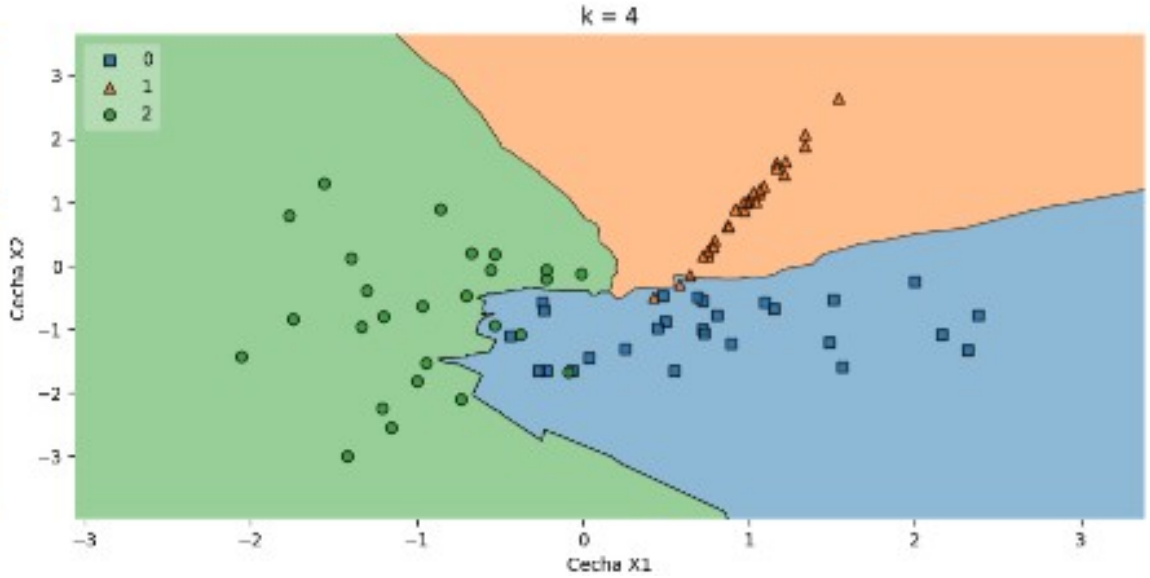


https://www.123rf.com/photo_89518160_smart-city-concept-powered-by-artificial-intelligence.html

Algorytm *k*-nn

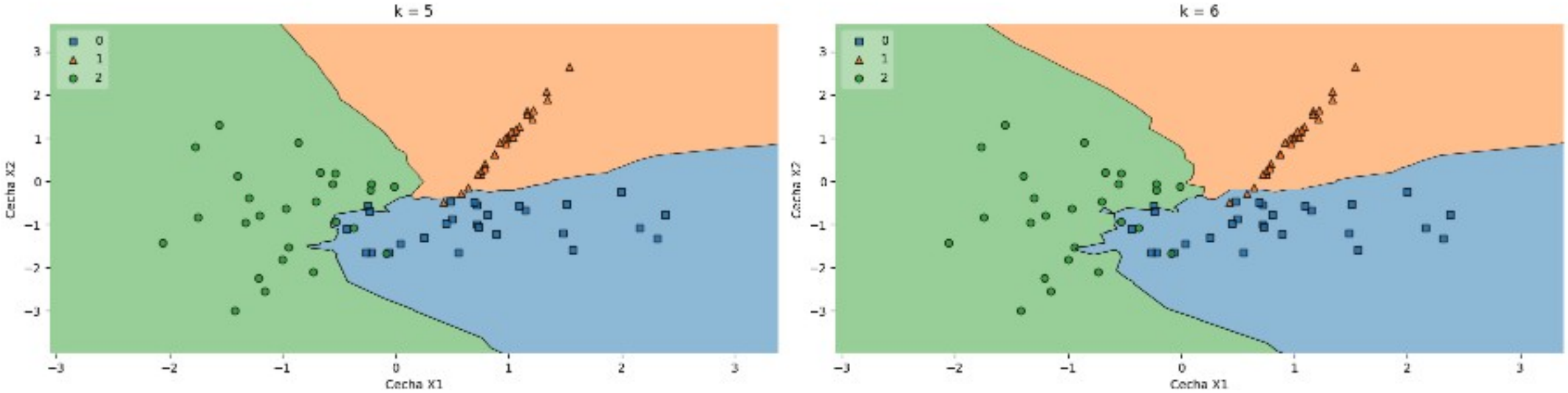
Realizacja algorytmu *k*-nn dla różnych wartości *k*





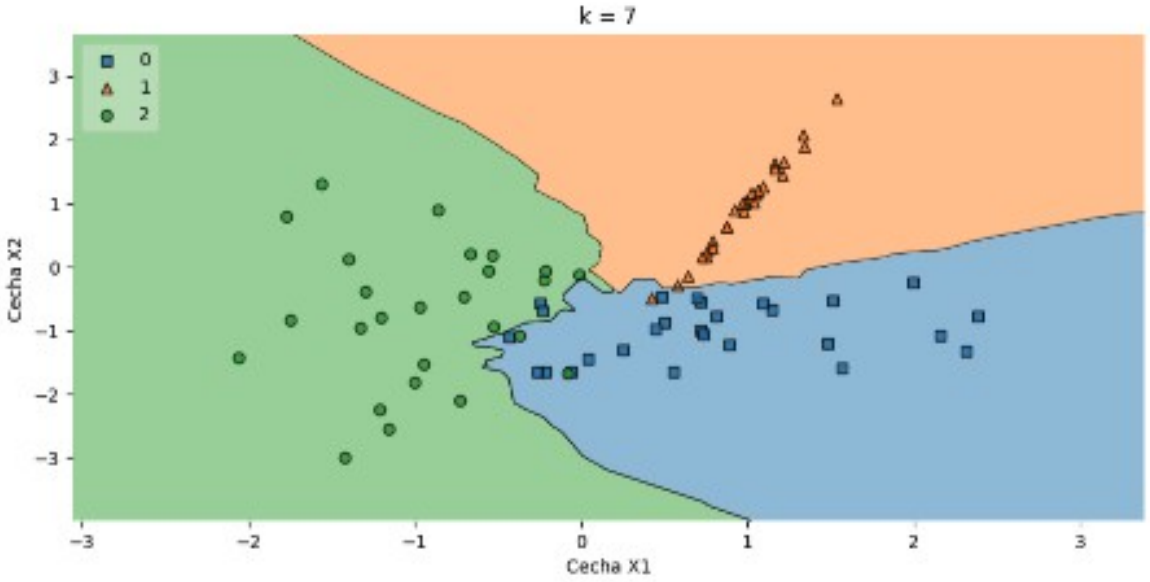
Algorytm *k*-nn

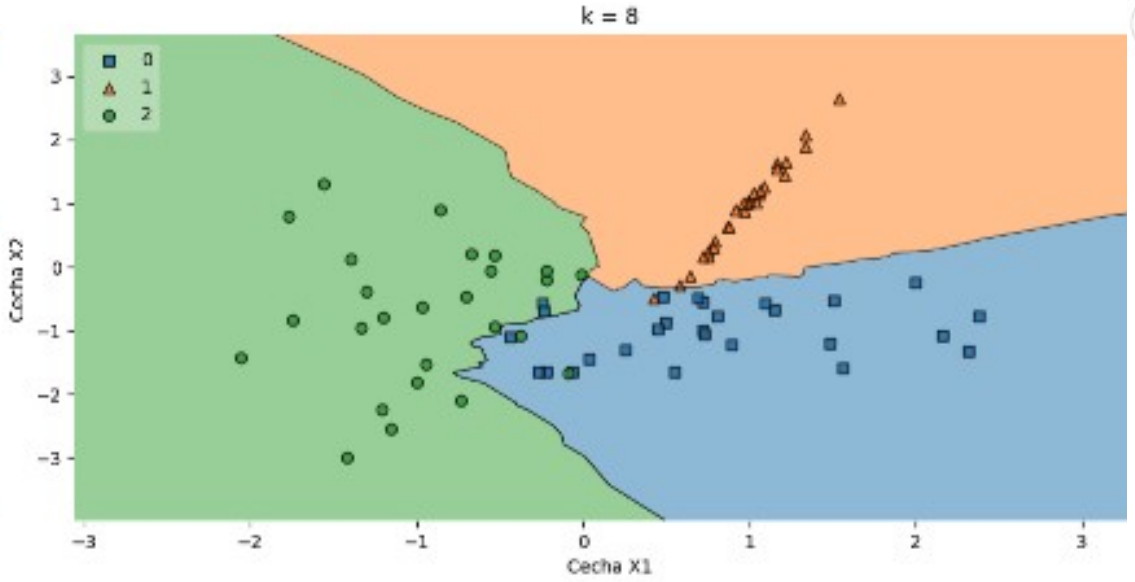
Realizacja algorytmu *k*-nn dla różnych wartości *k*



Algorytm *k*-nn

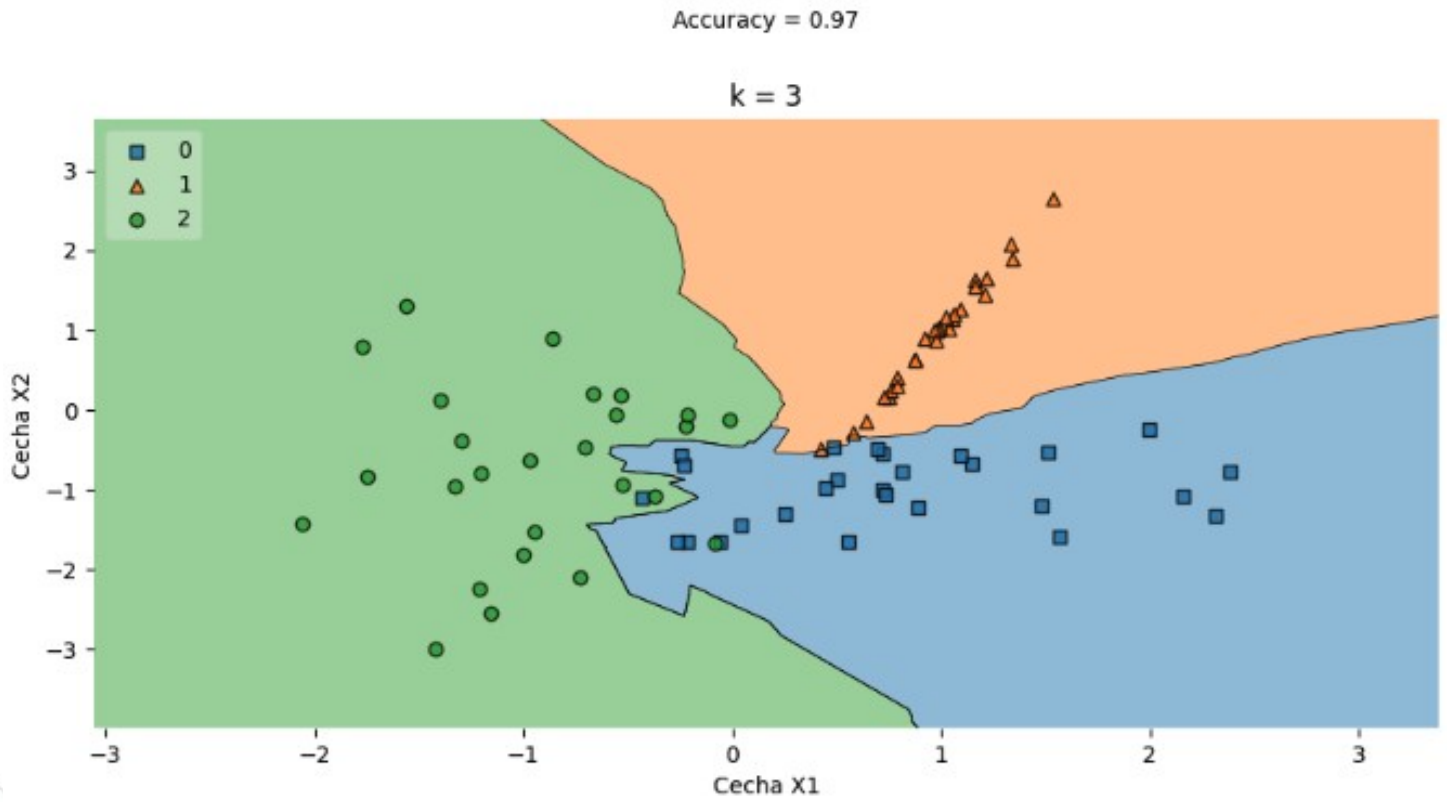
Realizacja algorytmu *k*-nn dla różnych wartości *k*





Algorytm *k*-nn

Optymalna wartość *k* w algorytmie *k*-nn ze względu na dokładność klasyfikacji w zbiorze testowym



Algorytm *k*-nn

Zalety algorytmu k-NN:

1. **Prostota i intuicyjność** - algorytm k-NN jest łatwy do zrozumienia i zaimplementowania, co czyni go jednym z pierwszych wyborów w prostych zadaniach klasyfikacji i regresji.
2. **Brak założeń co do rozkładu danych** - k-NN jest algorytmem nieparametrycznym, co oznacza, że nie zakłada żadnych specyficznych rozkładów dla danych, co sprawia, że jest bardzo elastyczny.
3. **Dobre wyniki przy małych zbiorach danych** - k-NN często osiąga dobre wyniki w zadaniach, gdzie mamy do czynienia z małymi zbiorami danych lub danymi o małej liczbie wymiarów.
4. **Możliwość łatwego dodania nowych danych** - k-NN nie wymaga żadnego procesu trenowania, więc nowe dane mogą być łatwo dodawane i natychmiast uwzględniane w klasyfikacji lub regresji.
5. **Wszechstronność** - k-NN może być stosowany zarówno do problemów klasyfikacji, jak i regresji, oraz dobrze działa w przypadku wieloklasowych problemów klasyfikacyjnych.

Algorytm *k*-nn

Wady algorytmu k-NN:

1. **Wysoka złożoność obliczeniowa -** k-NN wymaga przeszukiwania całego zbioru treningowego dla każdej klasyfikacji, co jest kosztowne obliczeniowo, zwłaszcza przy dużych zbiorach danych.
2. **Czułość na wielowymiarowość -** Wysoka liczba wymiarów danych (ang. curse of dimensionality) może osłabić efektywność k-NN, ponieważ wszystkie punkty mogą być równie odległe od siebie, co obniża jakość predykcji.
3. **Wymaga dobrego skalowania danych -** Wyniki k-NN są bardzo wrażliwe na różnice w skali cech. Jeśli cechy nie są odpowiednio skalowane, mogą dominować w miarę odległości, co może prowadzić do błędnych wyników.
4. **Wrażliwość na szumy i outliniery -** k-NN jest wrażliwy na obecność szumów i odchyleń w danych, ponieważ każdy sąsiad, niezależnie od swojej "jakości", wpływa na końcową decyzję.
5. **Potrzeba odpowiedniego wyboru k -** Wybór optymalnej liczby sąsiadów (k) może być trudny. Zbyt mała wartość k może prowadzić do nadmiernego dopasowania, podczas gdy zbyt duża wartość k może spowodować niedopasowanie.

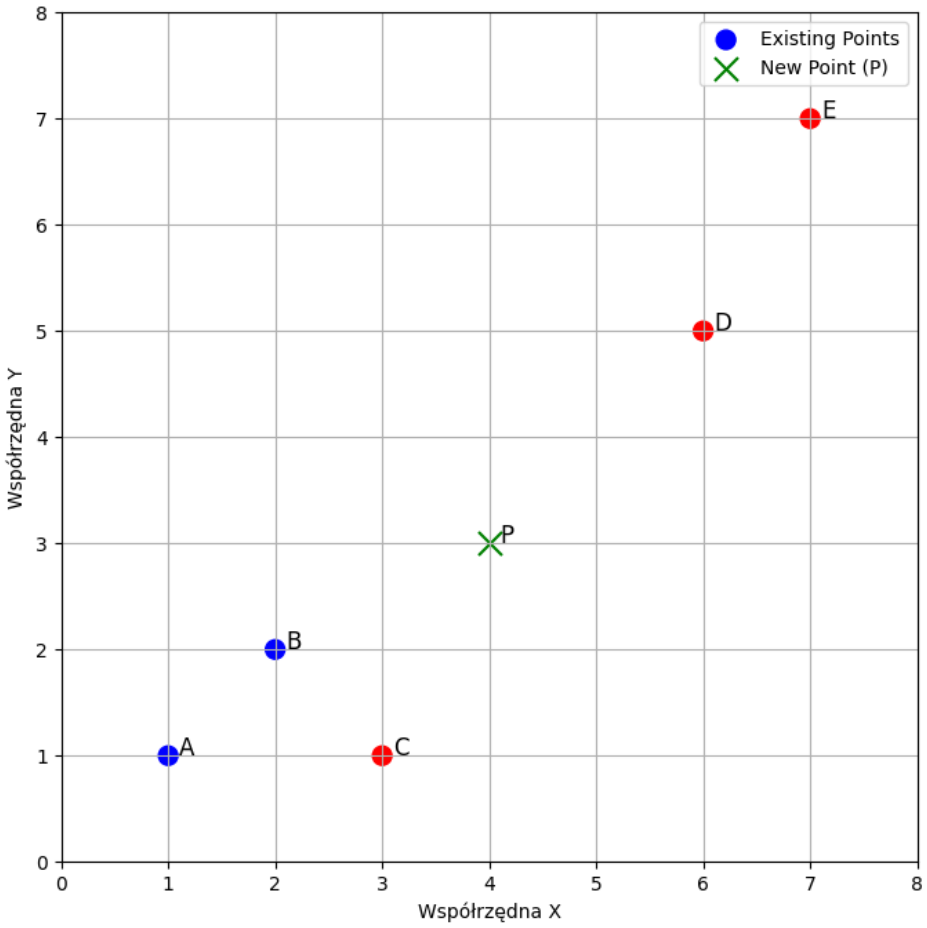
Algorytm *k*-nn

Załóżmy, że mamy 5 punktów w dwuwymiarowej przestrzeni, które są przypisane do dwóch klas: 0 i 1.

Punkt	Współrzędna (X, Y)	Klasa
A	(1, 1)	0
B	(2, 2)	0
C	(3, 1)	1
D	(6, 5)	1
E	(7, 7)	1

Nowy punkt do klasyfikacji:

Punkt	Współrzędna (X, Y)
P	(4, 3)



Algorytm *k*-nn

Kroki algorytmu k-NN:

1. Określenie liczby sąsiadów (k):

   Wybierzmy k = 3.

2. Obliczenie odległości od każdego punktu do nowego punktu P:

   Użyjemy metryki euklidesowej do obliczenia odległości.

Odległość między P i A:

$$d(P, A) = \sqrt{(4-1)^2 + (3-1)^2} = \sqrt{9+4} = \sqrt{13} \approx 3.61$$

Odległość między P i B:

$$d(P, B) = \sqrt{(4-2)^2 + (3-2)^2} = \sqrt{4+1} = \sqrt{5} \approx 2.24$$

Odległość między P i C:

$$d(P, C) = \sqrt{(4-3)^2 + (3-1)^2} = \sqrt{1+4} = \sqrt{5} \approx 2.24$$

Odległość między P i D:

$$d(P, D) = \sqrt{(4-6)^2 + (3-5)^2} = \sqrt{4+4} = \sqrt{8} \approx 2.83$$

Odległość między P i E:

$$d(P, E) = \sqrt{(4-7)^2 + (3-7)^2} = \sqrt{9+16} = \sqrt{25} = 5$$

Algorytm *k*-nn

3. Wybór k najbliższych sąsiadów:

   Najbliższe punkty do P to B (odległość 2.24), C (odległość 2.24) i D (odległość 2.83).

4. Klasyfikacja na podstawie sąsiadów:

   Punkty B i C są przypisane odpowiednio do klas 0 i 1.

   Punkt D jest przypisany do klasy 1.

   Większość z sąsiadów (2 z 3) to klasa 1, więc przypisujemy nowy punkt P do klasy 1.

Wynik:

Nowy punkt P o współrzędnych (4, 3) zostaje przypisany do klasy 1.

https://www.npr.org/2023/05/25/1177700852/ai-future-dangers-benefits

Algorytm *k*-nn

Niezbędnym elementem w budowaniu algorytmu *k*-najbliższych sąsiadów jest określenie metryki odległości.

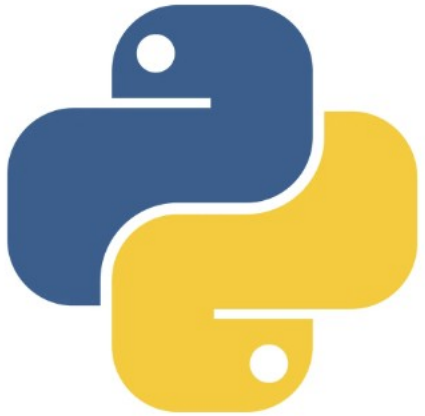
Możliwe jest zastosowanie szeregu metryk, np.:

- – odległość euklidesowa
- – kwadrat odległości euklidesowej
- – odległość miejska (Manhattan)
- – odległość Czebyszewa

gdzie:

– nowy przypadek

– jeden z przypadków przykładowych dla algorytmu



Algorytm k-najbliższych sąsiadów w Python

Algorytm knn

Zbiór danych Iris

Czym jest zbiór danych Iris? - jest to klasyczny zbiór danych używany w uczeniu maszynowym i statystyce.

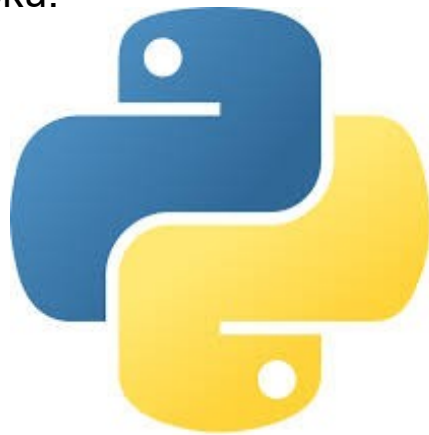
Zawiera informacje o 150 próbkach z trzech gatunków irysów:

- Setosa
- Versicolor
- Virginica

Dane zostały zebrane przez Edgara Andersona i spopularyzowane przez Ronalda Fishera w 1936 roku.

Zastosowanie:

- Klasyfikacja
- Wizualizacja danych
- Testowanie algorytmów uczenia maszynowego.



Algorytm knn

Struktura danych w zbiorze danych Iris

Kolumny w zbiorze:

- sepal_length: Długość działki kielicha (cm)
- sepal_width: Szerokość działki kielicha (cm)
- petal_length: Długość płatka (cm)
- petal_width: Szerokość płatka (cm)
- species: Gatunek kwiatu (Setosa, Versicolor, Virginica)

Algorytm knn

Struktura danych w zbiorze danych Iris

sepal_length	sepal_width	petal_length	petal_width	species
5.1	3.5	1.4	0.2	setosa
7.0	3.2	4.7	1.4	versicolor
6.3	3.3	6.0	2.5	virginica

Algorytm knn

Skalowanie zmiennych

Standaryzacja polega na sprowadzeniu dowolnego rozkładu normalnego $N(\mu,\sigma)$, o danych parametrach μ i σ do rozkładu standaryzowanego (modelowego) o wartości oczekiwanej $\mu=0$ i odchyleniu standardowym $\sigma=1$

$$P(X \leq x) = \Phi\left(\frac{x-\mu}{\sigma}\right)$$

czyli zmienną X zastępuje się zmienną standaryzowaną , która ma rozkład $N(0,1)$.

Algorytm knn

Accuracy

Dokładność klasyfikacji stanowi iloraz prawidłowo dokonanych predykcji przez model oraz wszystkich przeprowadzonych predykcji. Jest to zatem udział trafnych wskazań we wszystkich wskazaniach modelu.

Algorytm knn

Recall – iloraz liczby właściwie przewidzianych etykiet pozytywnych do sumy liczby właściwie przewidzianych etykiet pozytywnych i niewłaściwie przewidzianych etykiet negatywnych (dla klasyfikacji binarnej).

gdzie:

TP – *true positive*,

FN – *false negative*.

Algorytm knn

Precision - iloraz liczby właściwie przewidzianych etykiet pozytywnych do sumy liczby właściwie przewidzianych etykiet pozytywnych i niewłaściwie przewidzianych etykiet pozytywnych (dla klasyfikacji binarnej).

gdzie:

TP – *true positive*,

FP – *false positive*.

Algorytm knn

F1- score to miara używana w uczeniu maszynowym i statystyce do oceny skuteczności modelu klasyfikacji, szczególnie w przypadku niezrównoważonych danych. Jest to średnia harmoniczna precyzji i czułości (ang. precision i recall), która zapewnia równowagę między tymi dwoma metrykami.

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$