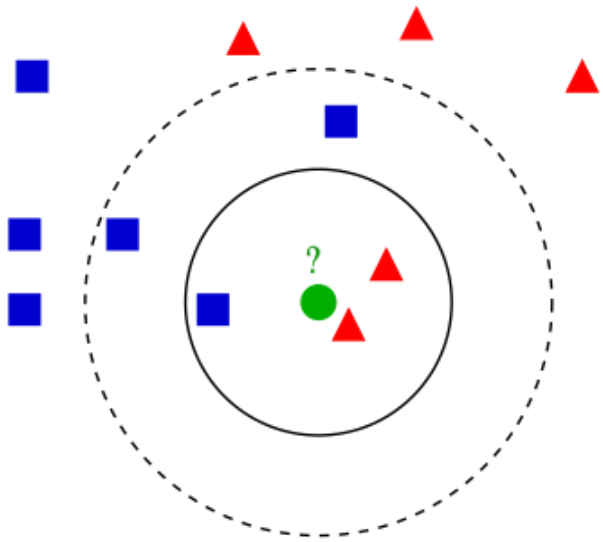
3. Zasada działania algorytmu k-NN dla klasyfikacji i regresji. Jego wady i zalety.

Zasada działania algorytmu k-NN (k-Nearest Neighbors)

Algorytm k-NN (k-Nearest Neighbors) jest jednym z najprostszych algorytmów uczenia maszynowego, wykorzystywany zarówno do klasyfikacji, jak i regresji. Jego główna zasada polega na tym, że dla nowego przykładu (punktu) decydujące są jego najbliżsi sąsiedzi w przestrzeni cech.



W przypadku k=3 (mniejszy okrąg), zielona kropka zostanie zakwalifikowana do czerwonych trójkątów. W przypadku k=5 (większy okrąg) – do niebieskich kwadratów

Klasyfikacja

W klasyfikacji, algorytm k-NN przypisuje nowemu przykładowi etykietę klasy na podstawie większości głosów jego kkk-najbliższych sąsiadów.

Kroki algorytmu k-NN dla klasyfikacji:

- Wybór parametru kkk: Liczba najbliższych sąsiadów, którzy będą brani pod uwagę.
- Obliczenie odległości: Użyj odpowiedniej metryki odległości, np. odległości euklidesowej, aby obliczyć odległość między nowym punktem a wszystkimi punktami w zbiorze treningowym.
- Wyznaczenie najbliższych sąsiadów: Znajdź kkk-najbliższych sąsiadów nowego punktu.
- Głosowanie: Przypisz nowemu punktowi etykietę klasy, która jest najczęstsza wśród kkk-najbliższych sąsiadów.

Regresja

W regresji, algorytm k-NN przewiduje wartość ciągłą dla nowego punktu na podstawie średniej wartości jego kkk-najbliższych sąsiadów.

Kroki algorytmu k-NN dla regresji:

- Wybór parametru kkk: Liczba najbliższych sąsiadów, którzy będą brani pod uwagę.
- Obliczenie odległości: Użyj odpowiedniej metryki odległości, aby obliczyć odległość między nowym punktem a wszystkimi punktami w zbiorze treningowym.
- Wyznaczenie najbliższych sąsiadów: Znajdź kkk-najbliższych sąsiadów nowego punktu.
- Predykcja: Przewidź wartość jako średnią wartość kkk-najbliższych sąsiadów.

Wady i zalety algorytmu k-NN

Zalety:

- Prostota: Łatwy do zrozumienia i zaimplementowania.
- Brak fazy treningowej: Nie wymaga trenowania modelu, co oszczędza czas na etapie treningu.
- Skuteczność w małych zbiorach danych: Działa dobrze w przypadkach, gdy zbiór danych jest mały i dobrze rozdzielony.

Wady:

- Skalowalność: Wysokie koszty obliczeniowe dla dużych zbiorów danych, ponieważ konieczne jest obliczanie odległości do wszystkich punktów w zbiorze treningowym.
- Wrażliwość na skalowanie cech: Wszystkie cechy powinny być odpowiednio skalowane, aby zapewnić, że odległości są porównywalne.
- Wpływ parametru kkk: Wybór odpowiedniego kkk jest kluczowy. Zbyt małe kkk może prowadzić do przetrenowania, a zbyt duże kkk może prowadzić do niedotrenowania.
- Wrażliwość na szum i dane odstające: Może być wrażliwy na szum w danych, ponieważ każdy punkt ma wpływ na wynik klasyfikacji lub regresji.

4. Zasada działania algorytmu k-means i k-medoids

Zasada działania algorytmu k-means

Algorytm k-means jest popularną metodą grupowania (klasteryzacji) danych, której celem jest podział zbioru danych na kk klastrów (grup), w taki sposób, aby obiekty w obrębie jednego klastra były do siebie jak najbardziej podobne, a obiekty z różnych klastrów jak najbardziej od siebie różne.

Kroki algorytmu k-means:

1. **Inicjalizacja**: Wybór kk początkowych centroidów (środków klastrów) losowo lub za pomocą bardziej zaawansowanych metod jak k-means++.
2. **Przypisanie punktów**: Każdy punkt danych jest przypisywany do najbliższego centroidu na podstawie wybranej metryki odległości, zazwyczaj Euklidesowej.
3. **Aktualizacja centroidów**: Nowe centroidy są obliczane jako średnia arytmetyczna punktów przypisanych do każdego klastra.
4. **Iteracja**: Kroki 2 i 3 są powtarzane, aż do momentu, gdy centroidy przestaną się zmieniać (osiągnięcie zbieżności) lub zostanie osiągnięta maksymalna liczba iteracji.

Zalety k-means:

- Prostota i szybkość działania.
- Efektywność dla dużych zbiorów danych.

Wady k-means:

- Wrażliwość na wybór początkowych centroidów.
- Problemy z wykrywaniem klastrów o nieregularnych kształtach.
- Wymaga podania liczby klastrów kk z góry.

Zasada działania algorytmu k-medoids

Algorytm k-medoids jest podobny do k-means, ale zamiast centroidów (średnich punktów), jako reprezentantów klastrów wybiera rzeczywiste punkty danych, zwane medoidami. Dzięki temu algorytm jest mniej wrażliwy na wartości odstające.

Kroki algorytmu k-medoids:

5. **Inicjalizacja**: Wybór kkk początkowych medoidów losowo.
6. **Przypisanie punktów**: Każdy punkt danych jest przypisywany do najbliższego medoidu na podstawie wybranej metryki odległości.
7. **Aktualizacja medoidów**: Dla każdego klastra, wybierany jest nowy medoid, który minimalizuje sumę odległości do wszystkich punktów w klastrze.
8. **Iteracja**: Kroki 2 i 3 są powtarzane, aż do osiągnięcia zbieżności lub maksymalnej liczby iteracji.

Zalety k-medoids:

- Odporniejszy na wartości odstające niż k-means.
- Lepsze w przypadku klastrów o nieregularnych kształtach.

Wady k-medoids:

- Wolniejszy niż k-means, zwłaszcza dla dużych zbiorów danych.
- Wymaga podania liczby klastrów kkk z góry.

18. Wpływ parametrów algorytmu genetycznego na szybkość i stabilność jego działania

Wpływ parametrów na algorytm genetyczny

Algorytmy genetyczne są inspirowane procesami biologicznej ewolucji i są stosowane do rozwiązywania problemów optymalizacyjnych. Wybór parametrów algorytmu genetycznego ma kluczowe znaczenie dla jego wydajności i stabilności.

Kluczowe parametry algorytmu genetycznego:

9. **Rozmiar populacji**:
    - Większa populacja może prowadzić do lepszego przeszukiwania przestrzeni rozwiązań, ale zwiększa również koszt obliczeniowy.
    - Zbyt mała populacja może prowadzić do przedwczesnej zbieżności do suboptymalnych rozwiązań.
10. **Liczba pokoleń (iteracji)**:
    - Większa liczba pokoleń zwiększa szansę na znalezienie optymalnego rozwiązania, ale również zwiększa czas obliczeń.
    - Zbyt mała liczba pokoleń może skutkować niedostatecznym przeszukaniem przestrzeni rozwiązań.
11. **Prawdopodobieństwo mutacji**:
    - Mutacja wprowadza różnorodność do populacji, zapobiegając przedwczesnej zbieżności.
    - Zbyt wysokie prawdopodobieństwo mutacji może prowadzić do losowego przeszukiwania i destabilizacji procesu.
    - Zbyt niskie prawdopodobieństwo mutacji może prowadzić do utraty różnorodności i zastoju w lokalnych minimach.
12. **Prawdopodobieństwo krzyżowania (crossover)**:

- Krzyżowanie pozwala na łączenie dobrych cech z dwóch rodziców, tworząc lepsze potomstwo.
- Zbyt wysokie prawdopodobieństwo krzyżowania może prowadzić do szybkiego zbieżności, ale ryzykuje utratą różnorodności.
- Zbyt niskie prawdopodobieństwo krzyżowania może prowadzić do wolniejszego przeszukiwania przestrzeni rozwiązań.

13. **Metoda selekcji**:
    - Różne metody selekcji, takie jak selekcja turniejowa, ruletka, czy selekcja rankingowa, wpływają na sposób wyboru osobników do reprodukcji.
    - Selekcja zbyt elitarna może prowadzić do przedwczesnej zbieżności, podczas gdy zbyt losowa selekcja może spowolnić proces optymalizacji.

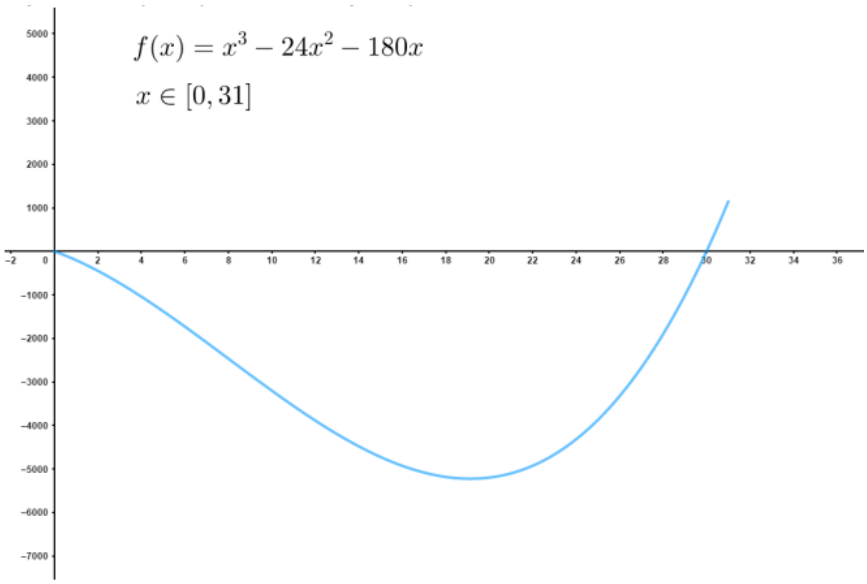
Podsumowanie wpływu parametrów:

- Dobór odpowiednich parametrów jest kluczowy dla zapewnienia efektywnego i stabilnego działania algorytmu genetycznego.
- Parametry muszą być dostosowane do specyfiki problemu i mogą wymagać eksperymentalnej optymalizacji.
- Równowaga między eksploracją (przeszukiwaniem nowych rozwiązań) a eksploatacją (wykorzystaniem najlepszych znalezionych rozwiązań) jest kluczowa dla sukcesu algorytmu.
-

Przykład algorytmu genetycznego

Do lepszego wyjaśnienia mechanizmów działania populacji, prześledzimy działanie jednej iteracji algorytmu genetycznego na przykładowym problemie. Będziemy dążyć do minimalizacji wartości funkcji na zbiorze liczb całkowitych z przedziału . Nasza funkcja jest dana następującym wzorem: $x^3 - 24x^2 - 180x$

Ponieważ poszukujemy najmniejszej wartości funkcji w przedziale , to osobnikiem w naszym algorytmie genetycznym będzie 5-bitowa liczba całkowita. Będzie to argument naszej funkcji zapisany w systemie dwójkowym.



Parametry i metody naszego algorytmu genetycznego:

metoda selekcji: metoda koła ruletki,
krzyżowanie: jednopunktowe,
prawdopodobieństwo krzyżowania: 1,
prawdopodobieństwo mutacji: 0,02,
metoda sukcesji: sukcesja z całkowitym zastępowaniem.
rozmiar populacji: 6

Pozostaje jeszcze kwestia wyboru populacji startowej algorytmu genetycznego. W tym przypadku została ona wygenerowana w sposób losowy. Przy użyciu generatora liczb pseudolosowych wylosowano sześć liczb całkowitych z przedziału [0,31].

Nr osobnika	Populacja początkowa	Wartość x	Wartość f(x)	Prawdopod. Wylosowania osobnika	Liczba wylosowanych kopii
1	00101	5	-1375	0,0856	0
2	10011	19	-5225	0,3253	2
3	00111	7	-2093	0,1304	1
4	01100	12	-3888	0,2421	2
5	11101	29	-1015	0,0632	0
6	01000	8	-2464	0,1534	1

Następnie po jednej iteracji algorytmu genetycznego otrzymujemy:

Pula rodzicielska	Wylosowany punkt krzyżowania	Populacja po krzyżowaniu	Populacja po mutacji	Wartość x	Wartość f(x)
10011	2	10100	10100	20	-5200
01100	2	01011	01011	11	-3553
10011	3	10000	10010	18	-5184
01000	3	01011	01011	11	-3553
01100	2	01111	0111	15	-4725
00111	2	00100	01100	12	-3888

Po jednej iteracji algorytmu najmniejszą wartością funkcji f jaką udało nam się uzyskać w nowym pokoleniu jest f(20)=-5200. Jest to wynik gorszy niż w poprzednim pokoleniu, który wynosił f(19)=-5225. Taka sytuacja może się wydarzyć ze względu na losowość naszego algorytmu genetycznego. Można temu zapobiec, stosując inną metodę selekcji lub sukcesji. Innym możliwym rozwiązaniem jest zwiększenie liczby iteracji i zapisywanie najlepszego wyniku spośród wszystkich dotychczasowych populacji.