EKSPERYMENT: OPTYMALIZACJA LICZBY KLASTRÓW W K-MEANS

Bartosz Bryniarski

2024

## Wprowadzenie do eksperymentu

Ten eksperyment ma na celu praktyczne zastosowanie metod wyboru optymalnej liczby klastrów w algorytmie K-means. Wykorzystamy dataset Iris, który jest idealny do nauki, ponieważ:

* Zawiera 3 gatunki irysów (prawdziwa liczba klastrów jest znana)
* Ma 4 wymiary (cechy), co pozwala na ciekawe wizualizacje
* Jest dobrze zbadany i powszechnie używany w nauczaniu ML
* Jeden gatunek jest wyraźnie odseparowany, dwa pozostałe częściowo się pokrywają

**Cel eksperymentu:**

1. Przetestować K-means dla różnych wartości k (2, 3, 4, 5)
2. Zastosować metody wyboru optymalnego k (metoda łokcia, współczynnik sylwetkowy)
3. Zrozumieć, dlaczego metryki mogą wskazywać różne wartości k
4. Porównać wyniki klasteryzacji z prawdziwymi etykietami
5. Nauczyć się wizualizować dane wielowymiarowe (4D → 2D/3D)

## Dataset Iris

Dataset Iris to klasyczny zbiór danych w machine learning, zebrany przez botanika Ronalda Fishera w 1936 roku.

**Zawartość:**

* **150 próbek** kwiatów irysa (po 50 z każdego gatunku)
* **3 gatunki:** Iris Setosa, Iris Versicolor, Iris Virginica
* **4 cechy** (wszystkie w centymetrach):
  1. Sepal length (długość działki kielicha)
  2. Sepal width (szerokość działki kielicha)
  3. Petal length (długość płatka)
  4. Petal width (szerokość płatka)

**Charakterystyka danych:**

* **Iris Setosa** - wyraźnie odseparowana od innych gatunków (mniejsze wymiary)
* **Iris Versicolor i Virginica** - częściowo się nakładają (podobne cechy)
* Brak brakujących wartości
* Dane ciągłe (numeryczne)

## Metody wyboru optymalnego k

### 1. Metoda łokcia (Elbow Method)

**Jak działa?**

* Trenujemy K-means dla różnych wartości k (np. 2-10)
* Dla każdego k obliczamy **inercję** (WCSS - Within-Cluster Sum of Squares)
* Inercja = suma kwadratów odległości wszystkich punktów do ich centroidów
* Rysujemy wykres: k vs inercja

**Co szukamy?**

**"Łokieć"** - punkt, gdzie krzywa inercji zaczyna się spłaszczać (przestaje szybko spadać).

**Interpretacja:**

* Inercja zawsze maleje wraz ze wzrostem k
* Dla k=n (liczba wszystkich punktów), inercja = 0 (każdy punkt to osobny klaster)
* Szukamy kompromisu: k wystarczająco duże, aby dobrze zgrupować dane, ale nie za duże

**Wzór na inercję:**

Inercja = Σ Σ ||xi - μj||²

gdzie:

* μj to centroid klastra j
* xi to punkt należący do klastra j

### 2. Współczynnik sylwetkowy (Silhouette Score)

**Jak działa?**

Dla każdego punktu i oblicza:

* **a(i)** - średnia odległość do innych punktów w tym samym klastrze (spójność)
* **b(i)** - średnia odległość do punktów w najbliższym innym klastrze (separacja)

**Wzór dla punktu i:**

s(i) = (b(i) - a(i)) / max(a(i), b(i))

**Wartości:**

* **s(i) ≈ +1:** Punkt dobrze dopasowany do swojego klastra
* **s(i) ≈ 0:** Punkt na granicy między klastrami
* **s(i) ≈ -1:** Punkt prawdopodobnie w złym klastrze

**Średni współczynnik sylwetkowy:**

**Silhouette Score** = średnia s(i) dla wszystkich punktów

**Interpretacja:**

* **0.7 - 1.0:** Silna struktura klastrów
* **0.5 - 0.7:** Rozsądna struktura
* **0.25 - 0.5:** Słaba struktura, klastry się nakładają
* **< 0.25:** Brak wyraźnej struktury

### 3. Davies-Bouldin Index

**Jak działa?**

Mierzy stosunek:

* Średnie rozproszenie wewnątrz klastrów (jak "rozproszone" są punkty w klastrze)
* Odległość między centroidami klastrów (jak daleko są od siebie klastry)

**Wzór:**

DB = (1/k) Σ max[(σi + σj) / d(ci, cj)]

gdzie:

* σi - średnia odległość punktów do centroidu w klastrze i
* d(ci, cj) - odległość między centroidami i i j

**Interpretacja:**

* **0** = idealna klasteryzacja (klastry zwarte i daleko od siebie)
* Im niższy, tym lepiej
* Typowe wartości: 0.5 - 2.5

**Zalety:** Szybszy do obliczenia niż Silhouette

## Wizualizacja danych 4D

**Problem:** Mamy 4 cechy, ale możemy narysować tylko 2D lub 3D. Jak pokazać wszystkie wymiary?

### Metoda 1: PCA (Principal Component Analysis)

**Redukcja wymiarów:** 4D → 2D

**Co to jest PCA?**

* Matematyczna technika kompresji danych
* Znajduje nowe osie (główne komponenty), które najlepiej oddają zmienność danych
* **PC1** (pierwsza składowa) - kierunek największej zmienności
* **PC2** (druga składowa) - kierunek drugiej największej zmienności (prostopadły do PC1)

**Jak działa?**

1. Centrowanie danych (średnia = 0)
2. Obliczenie macierzy kowariancji
3. Znalezienie wektorów własnych (eigenvectors)
4. Projekcja danych na nowe osie

**Główne komponenty to kombinacje liniowe oryginalnych cech:**

PC1 = w1×cecha1 + w2×cecha2 + w3×cecha3 + w4×cecha4

PC2 = v1×cecha1 + v2×cecha2 + v3×cecha3 + v4×cecha4

**Explained variance ratio:**

* Pokazuje, ile procent informacji zachowuje każda składowa
* Dla Iris: **PC1 ≈ 92.5%**, **PC2 ≈ 5.3%**
* **Razem: 97.8% całkowitej wariancji!**

**Zalety PCA:**

* Zachowuje większość informacji przy mniejszej liczbie wymiarów
* Usuwa korelacje między cechami
* Zmniejsza szum w danych

**Wady PCA:**

* Nowe osie są trudniejsze do interpretacji (są mieszankami oryginalnych cech)
* Zakłada liniowe zależności
* Wrażliwe na skalę danych (wymaga standaryzacji)

### Metoda 2: Macierz scatter plotów

Pokazuje wszystkie możliwe pary cech: **4 cechy → 4×4 = 16 wykresów**

**Struktura:**

* **Przekątna:** histogramy rozkładu każdej cechy
* **Poza przekątną:** scatter ploty dla par cech (i, j)

**Zalety:**

* Pokazuje oryginalne cechy (nie przekształcone)
* Widać wszystkie pary zależności
* Łatwa interpretacja

**Wady:**

* Dużo wykresów (dla n cech: n² wykresów)
* Nie widać interakcji więcej niż 2 cech jednocześnie

### Metoda 3: Wykres 3D + kolor jako 4. wymiar

**Wykorzystuje:** 3 osie przestrzenne + kolor punktów

**Jak czytać:**

* **Oś X, Y, Z:** 3 pierwsze cechy
* **Kolor punktów:** 4. cecha (skala kolorów)

**Zalety:**

* Intuicyjne
* Pokazuje 4 wymiary jednocześnie

**Wady:**

* Trudne do odczytu na wydruku (3D wymaga interakcji)
* Kolory mogą być mylące

## Implementacja w Pythonie

from sklearn.cluster import KMeans

from sklearn.datasets import load\_iris

from sklearn.metrics import silhouette\_score, davies\_bouldin\_score

from sklearn.decomposition import PCA

import matplotlib.pyplot as plt

import numpy as np

import seaborn as sns

# ==============================================================================

# 1. WCZYTANIE DANYCH IRIS

# ==============================================================================

iris = load\_iris()

X = iris.data # 4 cechy: długość/szerokość działki i płatka

y\_true = iris.target # 3 gatunki (prawdziwe etykiety)

feature\_names = iris.feature\_names

target\_names = iris.target\_names

print("Dataset Iris:")

print(f"- Liczba próbek: {X.shape[0]}")

print(f"- Liczba cech: {X.shape[1]}")

print(f"- Cechy: {feature\_names}")

print(f"- Gatunki: {target\_names}")

print(f"- Prawdziwa liczba klastrów: 3\n")

# ==============================================================================

# 2. TESTOWANIE K-MEANS DLA RÓŻNYCH WARTOŚCI k

# ==============================================================================

print("Testuję K-means dla k = 2, 3, 4, 5...")

K\_range = range(2, 6) # k = 2, 3, 4, 5

inertias = []

silhouette\_scores = []

davies\_bouldin\_scores = []

for k in K\_range:

kmeans = KMeans(n\_clusters=k, random\_state=42, n\_init=10)

kmeans.fit(X)

inertias.append(kmeans.inertia\_)

silhouette\_scores.append(silhouette\_score(X, kmeans.labels\_))

davies\_bouldin\_scores.append(davies\_bouldin\_score(X, kmeans.labels\_))

# ==============================================================================

# 3. WIZUALIZACJA: METODA ŁOKCIA I WSPÓŁCZYNNIK SYLWETKOWY

# ==============================================================================

fig, axes = plt.subplots(1, 3, figsize=(18, 5))

# Wykres 1: Metoda łokcia (Inercja)

axes[0].plot(K\_range, inertias, 'bo-', linewidth=2, markersize=10)

axes[0].set\_xlabel('Liczba klastrów (k)', fontsize=12)

axes[0].set\_ylabel('Inercja', fontsize=12)

axes[0].set\_title('Metoda Łokcia\n(szukamy "łokcia" - punktu przegięcia)', fontsize=13)

axes[0].grid(True, alpha=0.3)

axes[0].set\_xticks(K\_range)

# Zaznaczenie k=3

axes[0].axvline(x=3, color='red', linestyle='--', alpha=0.5, label='k=3 (prawdziwe)')

axes[0].legend()

# Wykres 2: Współczynnik sylwetkowy

axes[1].plot(K\_range, silhouette\_scores, 'go-', linewidth=2, markersize=10)

axes[1].set\_xlabel('Liczba klastrów (k)', fontsize=12)

axes[1].set\_ylabel('Współczynnik sylwetkowy', fontsize=12)

axes[1].set\_title('Współczynnik Sylwetkowy\n(wyższy = lepiej)', fontsize=13)

axes[1].grid(True, alpha=0.3)

axes[1].set\_xticks(K\_range)

axes[1].axvline(x=3, color='red', linestyle='--', alpha=0.5, label='k=3 (prawdziwe)')

axes[1].legend()

# Zaznaczenie najlepszego wyniku

best\_k\_silhouette = K\_range[np.argmax(silhouette\_scores)]

axes[1].scatter([best\_k\_silhouette], [max(silhouette\_scores)],

color='gold', s=200, zorder=5, edgecolor='orange', linewidth=2)

# Wykres 3: Davies-Bouldin Index

axes[2].plot(K\_range, davies\_bouldin\_scores, 'ro-', linewidth=2, markersize=10)

axes[2].set\_xlabel('Liczba klastrów (k)', fontsize=12)

axes[2].set\_ylabel('Davies-Bouldin Index', fontsize=12)

axes[2].set\_title('Davies-Bouldin Index\n(niższy = lepiej)', fontsize=13)

axes[2].grid(True, alpha=0.3)

axes[2].set\_xticks(K\_range)

axes[2].axvline(x=3, color='red', linestyle='--', alpha=0.5, label='k=3 (prawdziwe)')

axes[2].legend()

# Zaznaczenie najlepszego wyniku

best\_k\_db = K\_range[np.argmin(davies\_bouldin\_scores)]

axes[2].scatter([best\_k\_db], [min(davies\_bouldin\_scores)],

color='gold', s=200, zorder=5, edgecolor='orange', linewidth=2)

plt.tight\_layout()

plt.show()

# ==============================================================================

# 4. ANALIZA WYNIKÓW

# ==============================================================================

print("\n" + "="\*70)

print("WYNIKI DLA RÓŻNYCH WARTOŚCI k:")

print("="\*70)

print(f"{'k':<5} {'Inercja':<15} {'Silhouette':<15} {'Davies-Bouldin':<15}")

print("-"\*70)

for i, k in enumerate(K\_range):

print(f"{k:<5} {inertias[i]:<15.2f} {silhouette\_scores[i]:<15.4f} {davies\_bouldin\_scores[i]:<15.4f}")

print("\n" + "="\*70)

print("KTÓRE k JEST OPTYMALNE?")

print("="\*70)

print(f"Według Silhouette Score: k = {best\_k\_silhouette} (najwyższy wynik)")

print(f"Według Davies-Bouldin: k = {best\_k\_db} (najniższy wynik)")

print(f"Prawdziwa liczba gatunków: k = 3")

print("\nWNIOSKI:")

print("- K=2 jest za mało (metrics pokazują, że można podzielić bardziej)")

print("- K=3 odpowiada prawdziwej liczbie gatunków - dobry kompromis")

print("- K=2 ma najwyższy Silhouette, ale to za mało klastrów dla Iris")

print("- Metoda łokcia sugeruje k=3 (punkt przegięcia)")

print("="\*70 + "\n")

# ==============================================================================

# 5. K-MEANS Z OPTYMALNYM k=3

# ==============================================================================

optimal\_k = 3

kmeans\_optimal = KMeans(n\_clusters=optimal\_k, random\_state=42, n\_init=10)

y\_kmeans = kmeans\_optimal.fit\_predict(X)

# ==============================================================================

# 6. WIZUALIZACJA 4D DANYCH - METODA 1: PCA (redukcja do 2D)

# ==============================================================================

print("Redukuję 4 wymiary do 2D za pomocą PCA...")

pca = PCA(n\_components=2)

X\_pca = pca.fit\_transform(X)

fig, axes = plt.subplots(1, 2, figsize=(16, 6))

# Prawdziwe etykiety

scatter1 = axes[0].scatter(X\_pca[:, 0], X\_pca[:, 1], c=y\_true,

cmap='viridis', alpha=0.7, s=100, edgecolor='black')

axes[0].set\_xlabel(f'PC1 ({pca.explained\_variance\_ratio\_[0]:.1%} wariancji)', fontsize=12)

axes[0].set\_ylabel(f'PC2 ({pca.explained\_variance\_ratio\_[1]:.1%} wariancji)', fontsize=12)

axes[0].set\_title('Prawdziwe gatunki (PCA 2D)', fontsize=13)

axes[0].grid(True, alpha=0.3)

cbar1 = plt.colorbar(scatter1, ax=axes[0], ticks=[0, 1, 2])

cbar1.set\_ticklabels(target\_names)

# K-means (k=3)

scatter2 = axes[1].scatter(X\_pca[:, 0], X\_pca[:, 1], c=y\_kmeans,

cmap='viridis', alpha=0.7, s=100, edgecolor='black')

# Dodanie centroidów

centroids\_pca = pca.transform(kmeans\_optimal.cluster\_centers\_)

axes[1].scatter(centroids\_pca[:, 0], centroids\_pca[:, 1],

s=500, c='red', marker='X', edgecolors='black',

linewidths=3, label='Centroidy', zorder=5)

axes[1].set\_xlabel(f'PC1 ({pca.explained\_variance\_ratio\_[0]:.1%} wariancji)', fontsize=12)

axes[1].set\_ylabel(f'PC2 ({pca.explained\_variance\_ratio\_[1]:.1%} wariancji)', fontsize=12)

axes[1].set\_title(f'K-means (k={optimal\_k}) - PCA 2D', fontsize=13)

axes[1].grid(True, alpha=0.3)

axes[1].legend()

cbar2 = plt.colorbar(scatter2, ax=axes[1], ticks=[0, 1, 2])

cbar2.set\_label('Klaster', fontsize=11)

plt.tight\_layout()

plt.show()

print(f"PCA: Zachowano {pca.explained\_variance\_ratio\_.sum():.1%} całkowitej wariancji")

# ==============================================================================

# 7. WIZUALIZACJA 4D - METODA 2: MACIERZ SCATTER PLOTÓW (wszystkie pary cech)

# ==============================================================================

fig, axes = plt.subplots(4, 4, figsize=(16, 16))

for i in range(4):

for j in range(4):

ax = axes[i, j]

if i == j:

# Histogram na przekątnej

ax.hist(X[:, i], bins=20, color='skyblue', edgecolor='black', alpha=0.7)

ax.set\_ylabel('Częstość', fontsize=9)

else:

# Scatter plot z kolorami według K-means

scatter = ax.scatter(X[:, j], X[:, i], c=y\_kmeans,

cmap='viridis', alpha=0.6, s=30, edgecolor='black', linewidth=0.5)

# Dodanie centroidów

ax.scatter(kmeans\_optimal.cluster\_centers\_[:, j],

kmeans\_optimal.cluster\_centers\_[:, i],

s=200, c='red', marker='X', edgecolors='black', linewidths=2)

# Etykiety

if i == 3:

ax.set\_xlabel(feature\_names[j], fontsize=10)

else:

ax.set\_xticklabels([])

if j == 0:

ax.set\_ylabel(feature\_names[i], fontsize=10)

else:

ax.set\_yticklabels([])

ax.grid(True, alpha=0.2)

plt.suptitle(f'Macierz Scatter Plotów - K-means (k={optimal\_k})\nCzerwone X = Centroidy',

fontsize=16, y=0.995)

plt.tight\_layout()

plt.show()

# ==============================================================================

# 8. WIZUALIZACJA 4D - METODA 3: SCATTER 3D Z KOLOREM jako 4. wymiar

# ==============================================================================

fig = plt.figure(figsize=(16, 6))

# Wykres 1: Prawdziwe gatunki

ax1 = fig.add\_subplot(121, projection='3d')

scatter1 = ax1.scatter(X[:, 0], X[:, 1], X[:, 2],

c=X[:, 3], cmap='coolwarm',

s=100, alpha=0.7, edgecolor='black')

ax1.set\_xlabel(feature\_names[0], fontsize=10)

ax1.set\_ylabel(feature\_names[1], fontsize=10)

ax1.set\_zlabel(feature\_names[2], fontsize=10)

ax1.set\_title('Iris 4D: 3 osie + kolor = 4. cecha\n(Prawdziwe dane)', fontsize=12)

cbar1 = plt.colorbar(scatter1, ax=ax1, pad=0.1, shrink=0.8)

cbar1.set\_label(feature\_names[3], fontsize=10)

# Wykres 2: K-means

ax2 = fig.add\_subplot(122, projection='3d')

scatter2 = ax2.scatter(X[:, 0], X[:, 1], X[:, 2],

c=y\_kmeans, cmap='viridis',

s=100, alpha=0.7, edgecolor='black')

# Centroidy

ax2.scatter(kmeans\_optimal.cluster\_centers\_[:, 0],

kmeans\_optimal.cluster\_centers\_[:, 1],

kmeans\_optimal.cluster\_centers\_[:, 2],

s=500, c='red', marker='X', edgecolors='black',

linewidths=3, label='Centroidy')

ax2.set\_xlabel(feature\_names[0], fontsize=10)

ax2.set\_ylabel(feature\_names[1], fontsize=10)

ax2.set\_zlabel(feature\_names[2], fontsize=10)

ax2.set\_title(f'K-means (k={optimal\_k})', fontsize=12)

ax2.legend()

cbar2 = plt.colorbar(scatter2, ax=ax2, pad=0.1, shrink=0.8, ticks=[0, 1, 2])

cbar2.set\_label('Klaster', fontsize=10)

plt.tight\_layout()

plt.show()

# ==============================================================================

# 9. PORÓWNANIE Z PRAWDZIWYMI ETYKIETAMI

# ==============================================================================

from sklearn.metrics import confusion\_matrix, adjusted\_rand\_score

print("\n" + "="\*70)

print("PORÓWNANIE K-MEANS (k=3) Z PRAWDZIWYMI GATUNKAMI:")

print("="\*70)

# Adjusted Rand Index - mierzy zgodność klastrów z prawdziwymi etykietami

ari = adjusted\_rand\_score(y\_true, y\_kmeans)

print(f"Adjusted Rand Index: {ari:.4f}")

print("(1.0 = idealne dopasowanie, 0.0 = losowe)")

# Macierz pomyłek

conf\_matrix = confusion\_matrix(y\_true, y\_kmeans)

print("\nMacierz pomyłek (prawdziwe vs K-means):")

print(conf\_matrix)

print("\nInterpretacja:")

print("- Wiersze = prawdziwe gatunki")

print("- Kolumny = klastry K-means")

print("- K-means może nadać inne numery klastrów niż prawdziwe etykiety!")

# ==============================================================================

# 10. PODSUMOWANIE

# ==============================================================================

print("\n" + "="\*70)

print("PODSUMOWANIE:")

print("="\*70)

print(f"Optymalne k = {optimal\_k} (zgodne z prawdziwą liczbą gatunków)")

print(f"Silhouette Score: {silhouette\_scores[optimal\_k-2]:.4f}")

print(f"Davies-Bouldin Index: {davies\_bouldin\_scores[optimal\_k-2]:.4f}")

print(f"Zgodność z prawdziwymi etykietami (ARI): {ari:.4f}")

print("\nDLACZEGO k=3 jest optymalne?")

print("1. Metoda łokcia pokazuje punkt przegięcia przy k=3")

print("2. Współczynnik sylwetkowy jest wysoki")

print("3. Odpowiada prawdziwej liczbie gatunków")

print("4. K=2 byłoby zbyt uproszczone, k=4-5 to overfitting")

print("="\*70)

## Interpretacja wyników

**Analiza metryk dla różnych k**

**Typowe wyniki** (rzeczywiste mogą się nieznacznie różnić):

**k=2:**

* **Inercja:** ~152
* **Silhouette:** ~0.68 **(NAJWYŻSZY!)**
* **Davies-Bouldin:** ~0.65

**Dlaczego k=2 ma najwyższy Silhouette?**

Fizycznie rzeczywiście są 2 wyraźne grupy w danych:

1. Iris Setosa (wyraźnie oddzielona)
2. Versicolor + Virginica razem (nakładają się)

**k=3:**

* **Inercja:** ~78
* **Silhouette:** ~0.55
* **Davies-Bouldin:** ~0.66 **(NAJNIŻSZY!)**

To odpowiada prawdziwej liczbie gatunków, ale metryki są nieco niższe niż dla k=2, ponieważ Versicolor i Virginica częściowo się pokrywają.

**k=4:**

* **Inercja:** ~57
* **Silhouette:** ~0.50
* **Davies-Bouldin:** ~0.69

Zaczyna się overfitting - dzielimy naturalne grupy na mniejsze kawałki bez biologicznego uzasadnienia.

**k=5:**

* **Inercja:** ~46
* **Silhouette:** ~0.48
* **Davies-Bouldin:** ~0.73

Dalsze pogorszenie - zdecydowanie za dużo klastrów.

## Kluczowe obserwacje z wykresów

**1. Wykres PCA (2D):**

* **Iris Setosa** (fioletowe punkty) - wyraźnie odseparowana, tworzy zwarty klaster po lewej stronie
* **Iris Versicolor i Virginica** (turkusowe i żółte) - nakładają się w środkowej i prawej części wykresu
* **Centroidy K-means** (czerwone X) znajdują się w "centrach masy" każdego klastra
* PCA zachowuje ~98% całkowitej wariancji, więc wizualizacja 2D jest bardzo wiernym odzwierciedleniem rzeczywistych danych 4D

**Wnioski z PCA:**

* Jeden klaster jest doskonale separowalny (Setosa)
* Dwa klastry nakładają się, co tłumaczy niższy Silhouette dla k=3 niż k=2
* 4 cechy (długość i szerokość działki/płatka) nie wystarczają do idealnego rozdzielenia Versicolor od Virginica

**2. Macierz scatter plotów (4×4):**

* Histogramy na przekątnej pokazują rozkłady każdej cechy
* Wykresy poza przekątną pokazują wszystkie pary cech
* Widać, że niektóre pary cech lepiej rozdzielają klastry niż inne
* **Najlepsze rozdzielenie:** petal length vs petal width (dolny prawy róg)
* **Najgorsze rozdzielenie:** sepal width vs inne cechy (duże nakładanie)

Ta wizualizacja jest najbardziej szczegółowa, ale wymaga więcej miejsca i uwagi.

**3. Wykres 3D:**

* Pokazuje przestrzenne rozmieszczenie klastrów
* Kolor reprezentuje 4. wymiar
* Centroidy są wyraźnie widoczne jako czerwone gwiazdy
* Pozwala na obracanie i oglądanie z różnych perspektyw (w wersji interaktywnej)

## WNIOSKI KOŃCOWE

**Która wartość k jest optymalna?**

**Odpowiedź nie jest jednoznaczna i zależy od celu:**

**Z punktu widzenia metryk statystycznych:**

* **Silhouette** sugeruje k=2 (najwyższy wynik ~0.68)
* **Davies-Bouldin** sugeruje k=3 (najniższy wynik ~0.66)
* **Metoda łokcia** pokazuje wyraźne przegięcie przy k=3

**Z punktu widzenia biologicznego/domenowego:**

* **k=3** jest poprawne, bo są 3 różne gatunki irysów
* **ARI** (Adjusted Rand Index) ~0.73 pokazuje dobrą, ale nie idealną zgodność

**Nasze zalecenie: k=3**

**Dlaczego?**

1. Odpowiada rzeczywistej liczbie gatunków (wiedza domenowa)
2. Metoda łokcia wskazuje punkt przegięcia przy k=3
3. Davies-Bouldin Index jest najniższy dla k=3
4. Chociaż k=2 ma wyższy Silhouette, dzieli biologicznie różne gatunki (Versicolor i Virginica) do jednego klastra, co jest niepożądane

**Dlaczego metryki "mylą się"?**

**Metryki statystyczne** (Silhouette, Davies-Bouldin) mierzą tylko separację i spójność klastrów w przestrzeni cech. Nie wiedzą nic o biologicznej czy domenowej naturze danych.

**Problem z Iris:**

* Setosa jest wyraźnie odseparowana
* Versicolor i Virginica mają podobne cechy i częściowo się pokrywają

Dlatego z perspektywy czystej matematyki:

* **k=2** daje lepszą separację (Setosa vs Reszta)
* **k=3** próbuje rozdzielić nakładające się grupy (trudniejsze)

**Co to oznacza w praktyce?**

**Lekcja 1: Metryki to narzędzia pomocnicze, nie wyrocznie**

* Nie polegaj ślepo na jednej metryce
* Używaj kilku różnych metryk i porównuj wyniki
* Uwzględniaj wiedzę domenową

**Lekcja 2: Wizualizacja jest kluczowa**

* Wykresy PCA natychmiast pokazały problem: 2 klastry się nakładają
* Bez wizualizacji moglibyśmy nie zrozumieć, dlaczego metryki dają różne wyniki

**Lekcja 3: Cechy mają znaczenie**

* 4 cechy Iris nie wystarczają do idealnego rozdzielenia wszystkich gatunków
* Versicolor i Virginica wymagałyby dodatkowych cech (kolor, kształt, dane genetyczne)
* Feature engineering może znacząco poprawić klasteryzację

**Lekcja 4: Klasteryzacja nienadzorowana ma ograniczenia**

* K-means nie "wie" o biologicznych granicach między gatunkami
* Dla takich problemów klasyfikacja nadzorowana może być lepsza
* Klasteryzacja odkrywa strukturę matematyczną, nie semantyczną

**Kiedy użyć k=2 vs k=3 dla Iris?**

**Użyj k=2 gdy:**

* Nie znasz prawdziwych etykiet (eksploracja danych)
* Chcesz maksymalnej separacji klastrów
* Aplikacja wymaga prostoty (np. binarna decyzja: Setosa vs Nie-Setosa)

**Użyj k=3 gdy:**

* Znasz prawdziwą liczbę kategorii
* Ważna jest dokładność biologiczna/domenowa
* Musisz rozróżnić wszystkie 3 gatunki, nawet jeśli jest to trudniejsze

**Porównanie z prawdziwymi etykietami**

**Adjusted Rand Index (ARI)**

* **Typowa wartość:** ~0.73
* **Interpretacja:** Dobra zgodność, ale nie idealna
* **Dlaczego nie 1.0?** K-means myli niektóre Versicolor z Virginica (i odwrotnie)

**Macierz pomyłek (przykładowa):**

|  | **Klaster 0** | **Klaster 1** | **Klaster 2** |
| --- | --- | --- | --- |
| **Setosa** | 50 | 0 | 0 |
| **Versicolor** | 0 | 48 | 2 |
| **Virginica** | 0 | 14 | 36 |

**Interpretacja:**

* **Setosa:** 100% poprawnie sklasyfikowana (50/50)
* **Versicolor:** 96% poprawnie (48/50), 2 pomylone z Virginica
* **Virginica:** 72% poprawnie (36/50), 14 pomylone z Versicolor

**UWAGA:** K-means może nadawać inne numery klastrów!  
Klaster 0, 1, 2 w K-means NIE MUSZĄ odpowiadać gatunkom 0, 1, 2.  
Potrzebna jest analiza macierzy pomyłek, aby zrozumieć przypisanie.

## Praktyczne zastosowania tej wiedzy

**1. Segmentacja klientów**

Podobnie jak w Iris, klienci mogą tworzyć "nakładające się" segmenty. Wysokie metryki nie zawsze oznaczają doskonałą segmentację.

**2. Analiza obrazów medycznych**

Niektóre schorzenia mogą mieć podobne cechy i się nakładać, podobnie jak Versicolor i Virginica.

**3. Grupowanie dokumentów**

Dokumenty o podobnej tematyce będą się nakładać, co obniży metryki, ale jest to naturalne.

**4. Wykrywanie anomalii**

Punkty odstające (jak potencjalna 4. grupa w Iris) mogą być trudne do wykrycia standardowymi metrykami.

## ZADANIA DO SAMODZIELNEGO ROZWIĄZANIA

**Zadanie 1: Eksperyment z innymi wartościami k**

Zmień K\_range na range(2, 11) i przetestuj k od 2 do 10:

* Jak zmieniają się metryki?
* Gdzie jest "łokieć" na wykresie inercji?
* Czy dla k>5 metryki dalej się pogarszają?

**Zadanie 2: Wpływ inicjalizacji**

Zmień init='random' i n\_init=1, uruchom kilka razy:

* Czy dostajesz za każdym razem takie same wyniki?
* Jak bardzo różnią się metryki?
* Porównaj z init='k-means++' i n\_init=10

**Zadanie 3: Standaryzacja danych**

Dodaj standaryzację przed K-means:

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

# Użyj X\_scaled zamiast X w K-means

* Jak zmieniają się wyniki?
* Czy optymalne k się zmienia?
* Dlaczego standaryzacja ma lub nie ma wpływu?

**Zadanie 4: Użycie tylko 2 najważniejszych cech**

Wybierz tylko petal length i petal width (kolumny 2 i 3):

X\_subset = X[:, [2, 3]]

* Czy metryki się poprawiają?
* Dlaczego te 2 cechy wystarczają?
* Porównaj wizualizacje

**Zadanie 5: Inny dataset - Wine**

Załaduj dataset Wine i powtórz cały eksperyment:

from sklearn.datasets import load\_wine

wine = load\_wine()

X = wine.data

y\_true = wine.target

* Ile gatunków wina jest w datasecie?
* Czy K-means lepiej radzi sobie z Wine niż z Iris?
* Które metryki sugerują optymalne k?
* Czy wszystkie klastry są dobrze odseparowane?

**Zadanie 6: Analiza konkretnych błędów klasyfikacji**

Wypisz indeksy punktów, które K-means źle przypisał:

wrong\_indices = np.where(y\_kmeans != y\_true)[0]

print("Źle przypisane punkty:", wrong\_indices)

# Wyświetl ich cechy

for idx in wrong\_indices[:5]: # pierwsze 5

print(f"Punkt {idx}: cechy={X[idx]}, prawdziwe={y\_true[idx]}, K-means={y\_kmeans[idx]}")

* Dlaczego te konkretne punkty zostały źle przypisane?
* Czy są blisko granicy między klastrami?
* Czy mają nietypowe wartości cech?

## PODSUMOWANIE

**Kluczowe wnioski z eksperymentu:**

1. Metoda łokcia, Silhouette i Davies-Bouldin to pomocne narzędzia, ale nie są nieomylne
2. Różne metryki mogą sugerować różne optymalne wartości k
3. Wizualizacja danych (PCA, scatter matrix, 3D) jest niezbędna do zrozumienia wyników
4. Wiedza domenowa (tu: 3 gatunki irysów) powinna być uwzględniana w interpretacji
5. Nakładające się klastry (Versicolor i Virginica) są naturalnym zjawiskiem i obniżają metryki
6. K-means to algorytm matematyczny - nie rozumie semantyki ani kontekstu danych
7. PCA pozwala zredukować wymiary z minimalną utratą informacji (~98% dla Iris)
8. Dla danych rzeczywistych często potrzebne są lepsze cechy lub inne algorytmy klasteryzacji

**Złota zasada:**

**"Optymalne k to kompromis między metrykami statystycznymi, wiedzą domenową i wizualną interpretacją wyników. Nigdy nie ufaj ślepo jednej metryce!"**

**Dla datasetu Iris:**

**k=3 jest najlepszym wyborem**, ponieważ odpowiada rzeczywistości biologicznej i ma dobre metryki, mimo że Silhouette preferuje k=2.