Wprowadzenie do Sztucznej Inteligencji
Sprawozdanie 4

Michał Kallas

8 czerwca 2025

1 Zadanie 1

1.1 Wprowadzenie

Celem niniejszego eksperymentu była analiza skuteczności algorytmu k-średnich w klastrowaniu danych obrazowych cyfr ze zbioru MNIST. Zbadano wpływ liczby klastrów na jakość klastrowania oraz przeanalizowano możliwości zastosowania wyników w klasyfikatorze cyfr.

1.2 Algorytm k-średnich

Algorytm k-średnich (k-means) jest jedną z najpopularniejszych metod klastrowania nienadzorowanego. Jego działanie opiera się na następujących krokach:

1. **Inicjalizacja**: Wybór k początkowych centroidów. W podstawowej wersji algorytmu wybór ten jest losowy, ale można zastosować ulepszoną metodę inicjalizacji, taką jak k-means++, która wybiera centroidy tak, aby były możliwie daleko od siebie nawzajem.

2. **Przypisanie**: Każdy punkt danych zostaje przypisany do najbliższego centroidu (w sensie odległości euklidesowej).

3. **Aktualizacja**: Centroidy są przeliczane jako średnia arytmetyczna wszystkich punktów przypisanych do danego klastra.

4. **Iteracja**: Kroki 2-3 są powtarzane do momentu osiągnięcia zbieżności (gdy centroidy przestają się znacząco zmieniać).

Jakość klastrowania mierzona jest przez **inercję** - sumę kwadratów odległości wszystkich punktów od ich odpowiednich centroidów. Niższa inercja oznacza lepsze dopasowanie klastrów do danych.

1.3 Eksperyment

Wykorzystano pełny zbiór danych MNIST składający się z 70 000 obrazów cyfr (0-9) o rozdzielczości 28×28 pikseli. Dane zostały znormalizowane do zakresu [0,1] poprzez podzielenie przez 255. Eksperyment obejmował klastrowanie dla 10, 15, 20 i 30 klastrów

z wykorzystaniem poprawionej metody wyboru centroidów początkowych (k-means++). Dla każdej ilości klastrów zostało wykonanych po 5 prób w celu wyboru opcji z jak najmniejszą inercją.

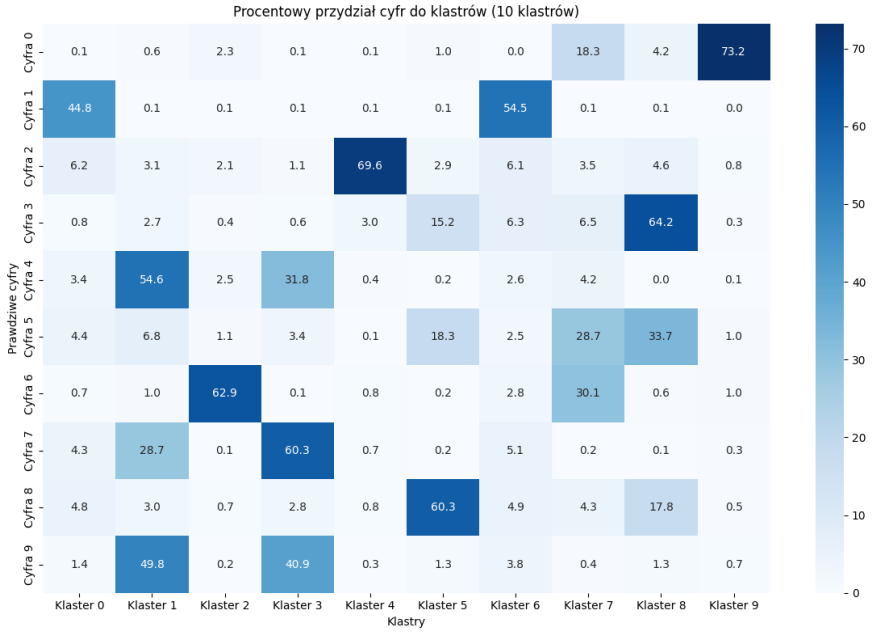
1.4 Wyniki eksperymentu

1.4.1 Podsumowanie wyników ilościowych

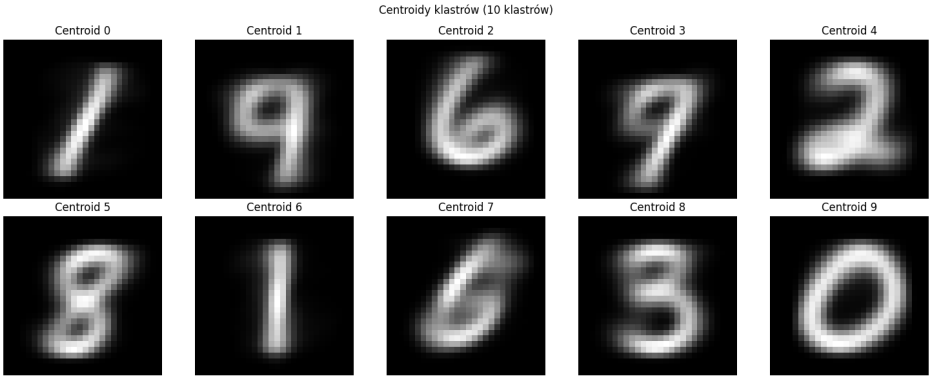
Tabela 1: Porównanie inercji względem ilości klastrów

Liczba klastrów	Inercja
10	2 744 056.01
15	2 573 264.04
20	2 458 773.40
30	2 311 527.15

1.4.2 Analiza dla 10 klastrów



Rysunek 1: Macierz przydziału cyfr do klastrów dla k=10



Rysunek 2: Centroidy klastrów dla k=10

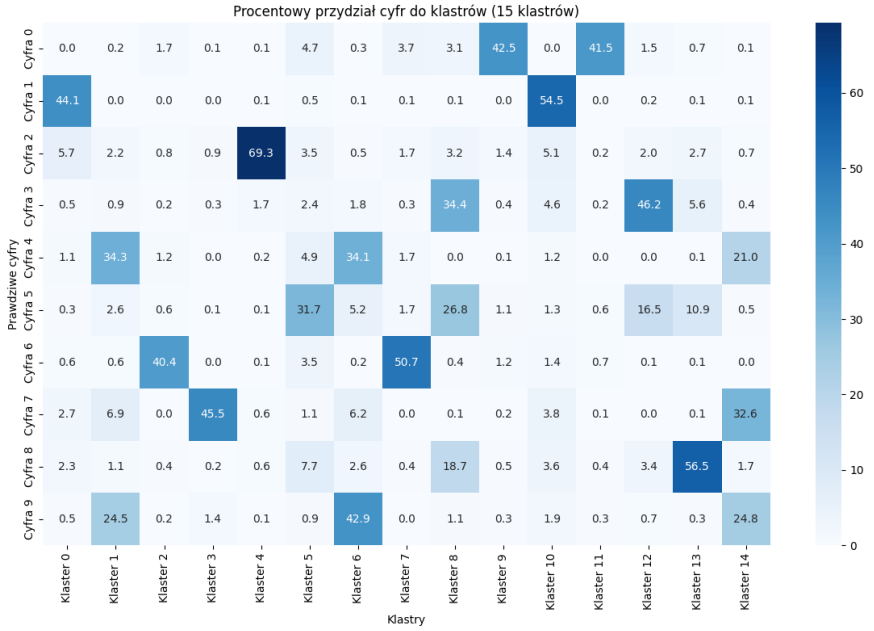
Klastrowanie dla 10 klastrów wykazało następujące charakterystyki:

- Centroidy przypominają odpowiednie cyfry
- Cyfra 0 najlepiej sklasyfikowana (73.2% czystości w klastrze 9)
- Cyfra 1 występująca w dwóch stylach - ukośnym oraz prostym
- Brak centroidów reprezentujących cyfy 4, 5 i 7
- Po 2 centroidy reprezentujące cyfry 1, 6 oraz 9

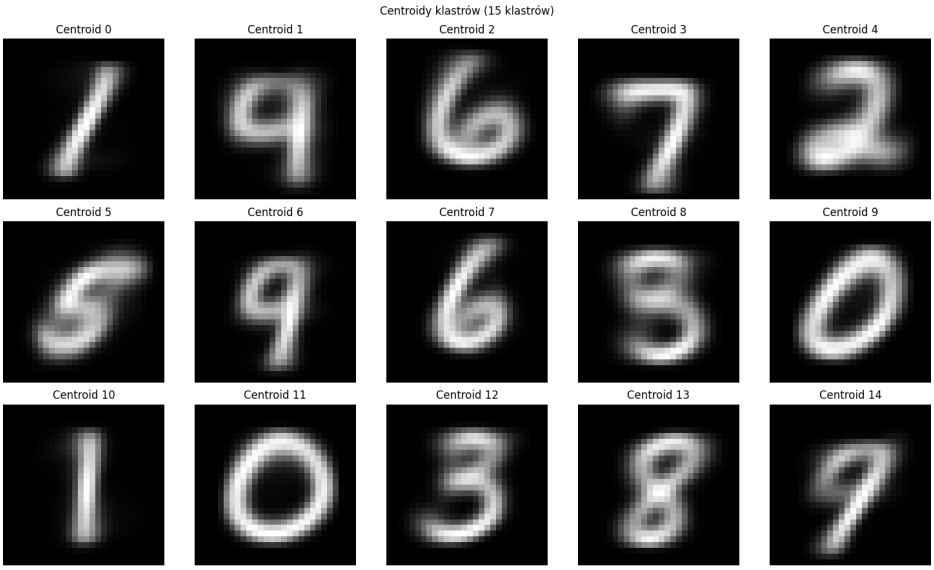
Sugerowane łączenia klastrów

- Cyfra 1: połączenie klastrów 0 i 6
- Cyfra 6: połączenie klastrów 2 i 7

1.4.3 Analiza dla 15 klastrów



Rysunek 3: Macierz przydziału cyfr do klastrów dla k=15



Rysunek 4: Centroidy klastrów dla k=15

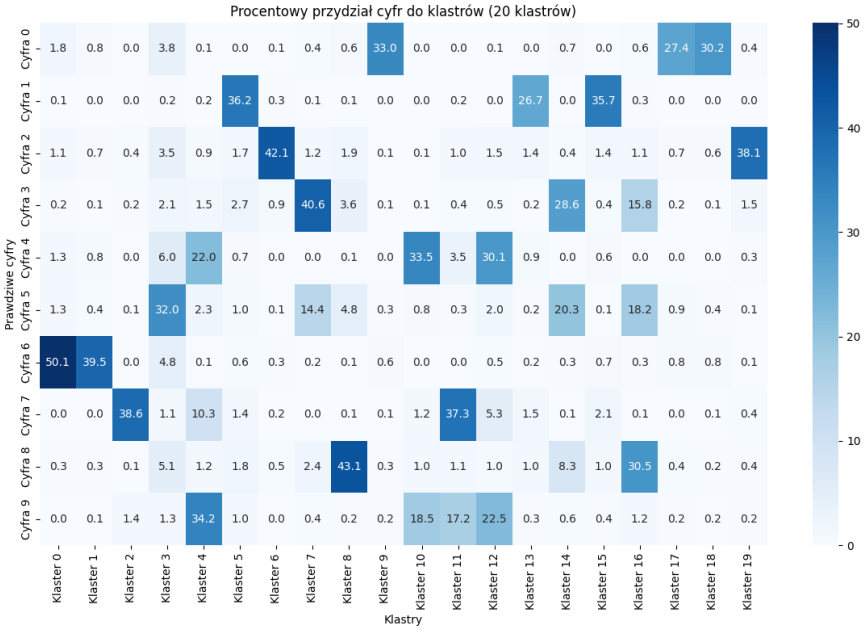
Zwiększenie liczby klastrów do 15 przyniosło:

- Pojawienie się centroidów reprezentujących cyfry 7 oraz 5(mocne rozmyte)
- Brak reprezentacji jedynie cyfry 4
- Aż 3 centroidy reprezentujące cyfrę 9, jednak należy zauważyć, że 4 jest bardzo podobne do 9 i przypisywane do tych centroidów

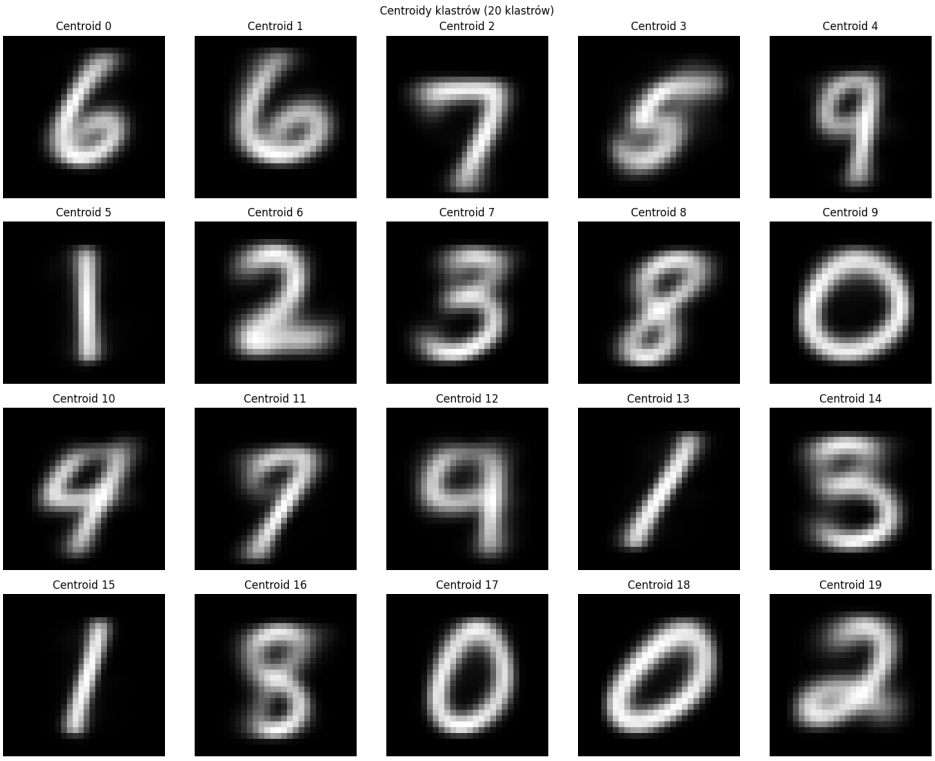
Sugerowane łączenia klastrów

- Cyfra 1: połączenie klastrów 0 i 10
- Cyfra 6: połączenie klastrów 2 i 7
- Cyfra 7: połączenie klastrów 3 i 14
- Cyfra 3: połączenie klastrów 8 i 12
- Cyfra 0: połączenie klastrów 9 i 11

1.4.4 Analiza dla 20 klastrów



Rysunek 5: Macierz przydziału cyfr do klastrów dla k=20



Rysunek 6: Centroidy klastrów dla k=20

Dalsze zwiększenie do 20 klastrów charakteryzowało się:

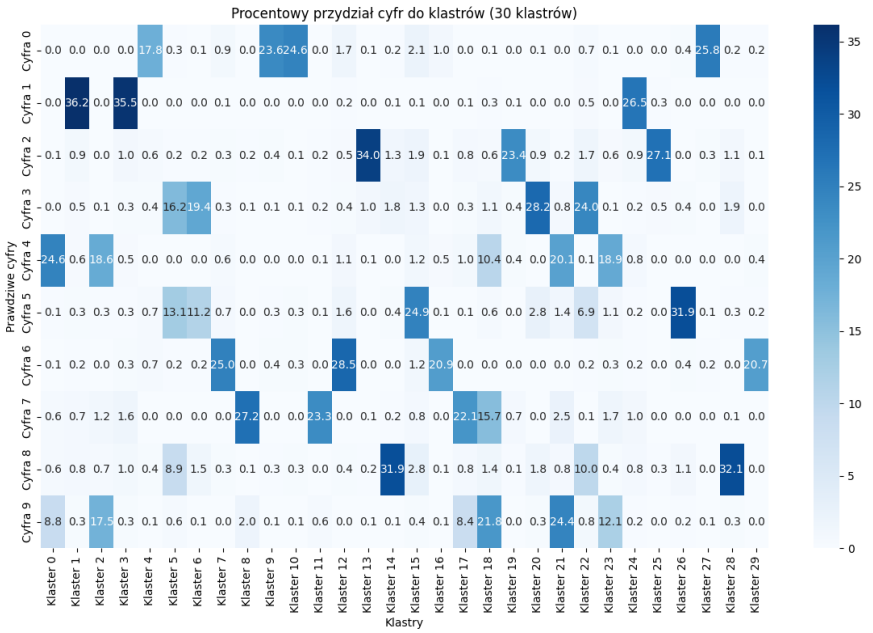
- 2 lub więcej centoridami dla większości cyfr

- Wciąż dużymi trudnościami w rozróżnianiu cyfr 4 i 9
- Wyraźnym rozdzieleniem stylów pisania cyfr

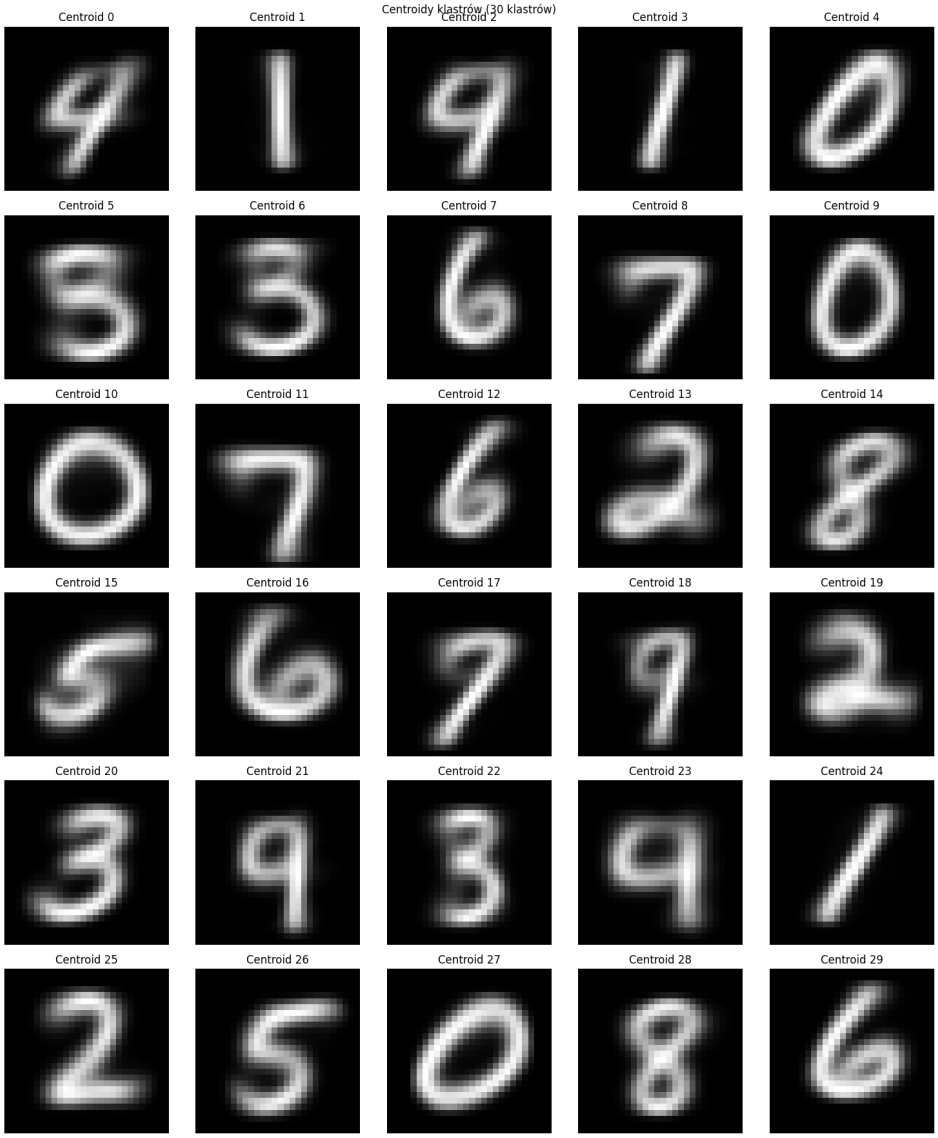
Sugerowane łączenia klastrów

- Wszystkie cyfry oprócz 5 mają możliwości łączenia 2-3 klastrów

1.4.5 Analiza dla 30 klastrów



Rysunek 7: Macierz przydziału cyfr do klastrów dla k=30



Rysunek 8: Centroidy klastrów dla k=30

Klastrowanie dla 30 klastrów wykazało:

- Bardzo szczegółowy podział na subtypy cyfr
- Dalsze problemy w odróżnieniu 4 od 9

Sugerowane łączenia klastrów

- Każda cyfra reprezentowana jest przez 2-4 klastry

1.5 Wnioski

Eksperyment wykazał, że algorytm k-średnich może skutecznie klastrować obrazy cyfr ze zbioru MNIST, mimo że jest to metoda nienadzorowana, a więc nie korzysta z etykiet klas. Centroidy uzyskane dla większości klastrów są wizualnie rozpoznawalne i mogą pełnić rolę prototypów cyfr.

Niektóre cyfry, takie jak 0, są znacznie łatwiejsze do zidentyfikowania w procesie klasteryzacji. Ich centroidy są wyraźne i rzadko mieszają się z innymi klasami. Z kolei cyfry o podobnych kształtach — szczególnie 4 i 9 — często występują w tych samych klastrach lub są wzajemnie mylone przez algorytm, co wynika z wizualnych podobieństw oraz różnorodności stylów pisma.

Zwiększanie liczby klastrów ma zarówno plusy, jak i minusy. Dla cyfr o prostej i jednoznacznej budowie wystarcza mniejsza liczba centroidów do ich prawidłowego odwzorowania. Tutaj zwiększanie liczby klastrów nie jest wymagane, a może wręcz prowadzić do nadmiernego dopasowania. Natomiast cyfry bardziej złożone i trudniejsze do rozróżnienia wymagają większej liczby centroidów, aby uchwycić pełne spektrum ich wariantów pisma oraz subtelnych różnic między nimi.

Warto również zauważyć, że mimo zastosowania metody *k-means++*, wyniki klasteryzacji mogą się różnić w zależności od losowego wyboru pierwszego centroidu. W kolejnych uruchomieniach algorytmu obserwowano różnice w strukturze klastrów spowodowane tą losowością.

2 Zadanie 2

2.1 Wprowadzenie

Celem zadania była implementacja algorytmu DBSCAN (Density-Based Spatial Clustering of Applications with Noise) do klasteryzacji zbioru danych MNIST. Zadanie polegało na implementacji algorytmu oraz na doborze odpowiednich parametrów algorytmu w celu minimalizacji szumu oraz uzyskania klastrów o wysokiej jednorodności.

2.2 Algorytm DBSCAN

DBSCAN to algorytm klasteryzacji oparty na gęstości, który grupuje punkty znajdujące się w obszarach o wysokiej gęstości, jednocześnie identyfikując punkty odstające jako szum. Algorytm wymaga dwóch kluczowych parametrów:

- ϵ (eps) - maksymalny promień sąsiedztwa punktu
- $minPts$ - minimalna liczba punktów w sąsiedztwie wymagana do utworzenia klastra

Algorytm klasyfikuje punkty na trzy kategorie:

- **Punkty centralne** - mają co najmniej $minPts$ sąsiadów w promieniu ϵ
- **Punkty brzegowe** - należą do sąsiedztwa punktu centralnego, ale same nie są centralne
- **Szum** - punkty, które nie należą do żadnego klastra

Główne zalety DBSCAN to zdolność do wykrywania klastrów o dowolnych kształtach, automatyczne określenie liczby klastrów oraz identyfikacja punktów odstających.

3 Redukcja wymiarowości - PCA i t-SNE

Dane MNIST w oryginalnej postaci mają 784 wymiary (28×28 pikseli), co sprawia, że podlegają **klątwie wielowymiarowości** (curse of dimensionality). Zjawisko to polega na tym, że w przestrzeniach o wysokiej wymiarowości:

- Odległości między punktami stają się mało znaczące
- Wszystkie punkty wydają się być w podobnej odległości od siebie
- Algorytmy oparte na odległości (jak DBSCAN) tracą na skuteczności
- Wzrasta złożoność obliczeniowa

Aby rozwiązać ten problem, zastosowano dwuetapową redukcję wymiarowości:

1. **PCA (Principal Component Analysis)** - redukcja z 784 do 50 wymiarów poprzez znalezienie kierunków największej wariancji w danych
2. **t-SNE (t-Distributed Stochastic Neighbor Embedding)** - redukcja z 50 do 2 wymiarów z zachowaniem lokalnej struktury danych

Takie podejście pozwala zachować istotne informacje o strukturze danych przy jednoczesnym znacznym zmniejszeniu wymiarowości, co czyni algorytm DBSCAN bardziej skutecznym.

4 Wyniki

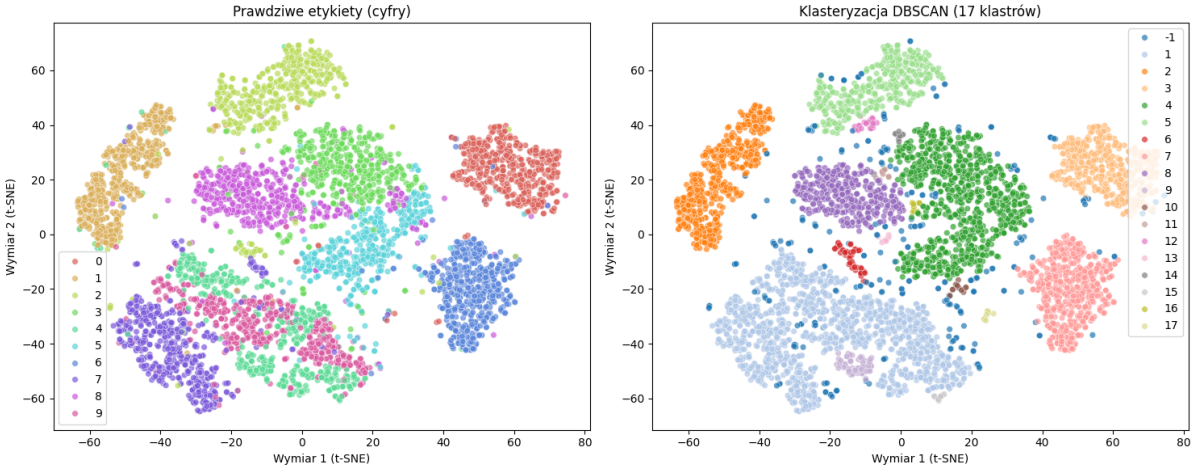
Bardzo ciężko było dobrać odpowiednie wartości ε i `min_samples`. Po przetestowaniu wielu różnych kombinacji parametrów, wybrano następujące wartości:

- $\varepsilon = 2.5$
- `min_samples` $= 9$

4.1 Uzyskane rezultaty

Metryka	**Wartość**
Liczba klastrów	17
Procent szumu	5.40%
Dokładność klasyfikacji	70.55%
Procent błędnych klasyfikacji	29.45%

Tabela 2: Wyniki klasteryzacji DBSCAN dla zbioru MNIST



Rysunek 9: Porównanie rzeczywistych etykiet z klasteryzacją DBSCAN

5 Wnioski

Na podstawie przeprowadzonego eksperymentu można sformułować następujące wnioski:

1. **Liczba klastrów**: Uzyskano 17 klastrów, co mieści się w zadanym przedziale 10-30 i wskazuje na rozsądne grupowanie danych.

2. **Poziom szumu**: 5.40% punktów zostało zaklasyfikowanych jako szum, co jest akceptowalnym wynikiem.

3. **Dokładność klasyfikacji**: 70.55% dokładności jest wynikiem umiarkowanym, ale należy pamiętać, że DBSCAN nie jest algorytmem nadzorowanym i nie wykorzystuje informacji o prawdziwych etykietach podczas klasteryzacji.

4. **Wpływ redukcji wymiarowości**: Zastosowanie PCA i t-SNE było kluczowe dla sukcesu algorytmu, pozwalając na efektywne działanie DBSCAN w przestrzeni o niskiej wymiarowości.

5. **Trudności w doborze parametrów**: Niezwykle trudno było dobrać odpowiednie wartości parametrów ε i `min_samples`. Algorytm DBSCAN jest bardzo wrażliwy na te parametry - zbyt małe wartości prowadzą do nadmiernej fragmentacji klastrów, a zbyt duże do łączenia różnych cyfr w jeden klaster lub klasyfikowania większości punktów jako szum.

Eksperyment potwierdza skuteczność algorytmu DBSCAN w kontekście klasteryzacji danych obrazowych po odpowiedniej redukcji wymiarowości.