

POLITECHNIKA WROCŁAWSKA
WYDZIAŁ INFORMATYKI I ZARZĄDZANIA

KIERUNEK: INFORMATYKA
SPECJALNOŚĆ: DANOLOGIA
KURS: INDUKCYJNE METODY ANALIZY DANYCH

Wybrane metody klasteryzacji w R
Dokumentacja ćwiczenia nr 3

AUTOR:
Adam Dłubak

PROWADZĄCY:
Dr inż. Paweł Myszkowski

Spis treści

1. Zbiory Badawcze	3
1.1. Iris Dataset	3
1.2. Wine Dataset	4
1.3. Glass Dataset	6
1.4. Pima Indians Diabetes Dataset	7
1.5. User Knowledge Modeling Dataset	8
2. Wstęp teoretyczny	10
2.1. Algorytm K-Means	10
2.2. Algorytm Partition Around Medoids (PAM)	11
2.3. Wybrane miary jakości klasteryzacji	13
2.4. Miary jakości klasyfikatora	15
3. Badania i analiza wyników	16
3.1. Analiza wyboru atrybutów do klasteryzacji	16
3.2. Analiza wyników klasteryzacji	17
3.2.1. User Knowledge Modeling Dataset	17
3.2.2. Wine Dataset	19
3.2.3. Glass Dataset	21
3.2.4. Pima Indians Diabetes Dataset	23
3.3. Prezentacja najlepszych uzyskanych wyników	25
3.3.1. Glass Dataset	25
3.3.2. Wine Dataset	26
3.3.3. Pima Indians Diabetes Dataset	27
3.3.4. User Knowledge Modeling Dataset	28
Indeks rzeczowy	30

Rozdział 1

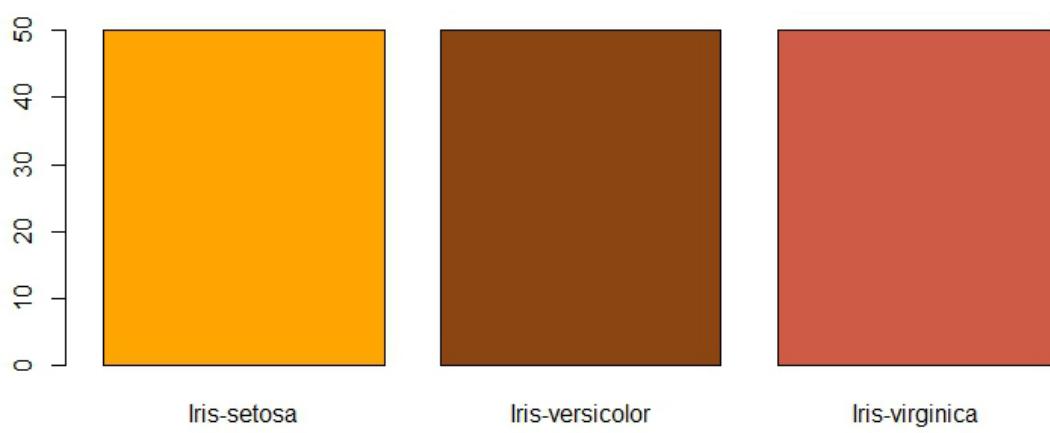
Zbiory Badawcze

W ramach realizacji ćwiczenia, badania zostały oparte o 1 zbiór wykorzystany do wstępnej weryfikacji, 3 zbiory danych testowych wykorzystywanych już wcześniej, przeznaczonych głównie do zadania klasyfikacji oraz 1 zbiór, który jeszcze nie był badany i przeznaczony jest szczególnie do zadań klasteryzacji. Wykorzystane zbiory danych to:

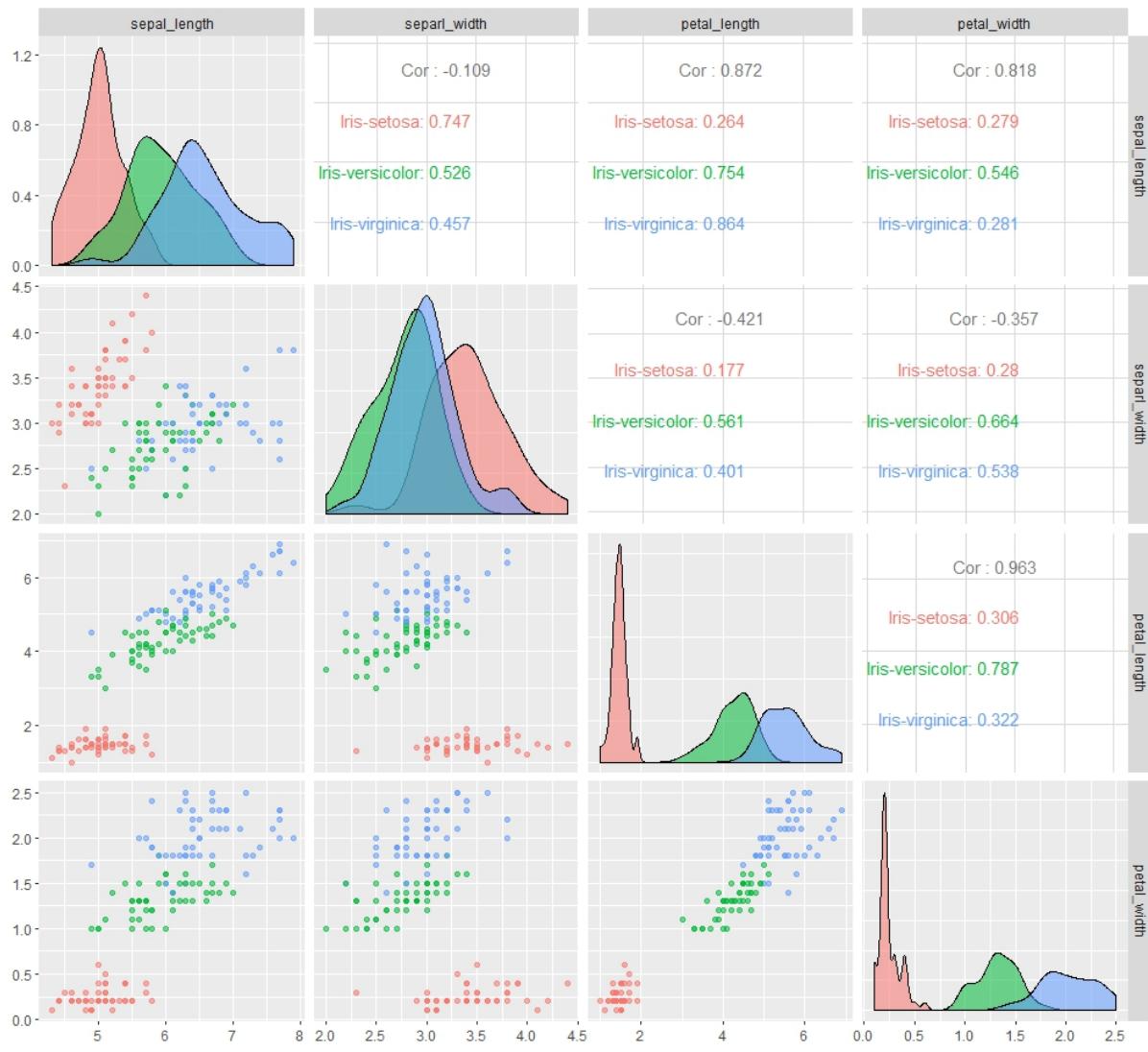
1.1. Iris Dataset

Zestaw pomiarów kwiatów irysa, udostępniony po raz pierwszy przez Ronaldiego Fishera w roku 1936. Jeden z najbardziej znanych zbiorów, a zarazem bardzo prosty i użyteczny. Zbiór irysów składa się z 4 wartości pomiarów jego płatków (szerokości i długość) oraz klasy do jakiej należy.

- Liczba atrybutów: 4
- Rodzaj atrybutów: wartości typu Float
- Liczba instancji: 150
- Liczba klas: 3



Rys. 1.1: Rozłożenie ilościowe poszczególnych klas zbioru Iris

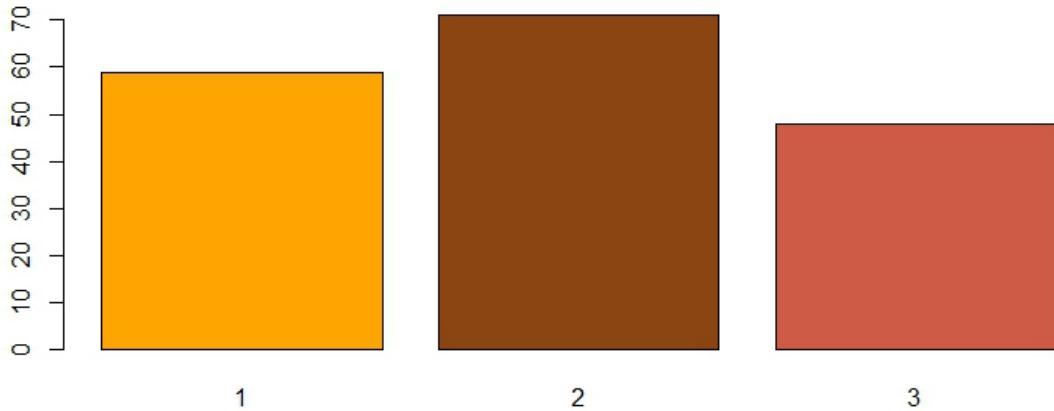


Rys. 1.2: Zestawienie zależności atrybutów i klas zbioru Iris

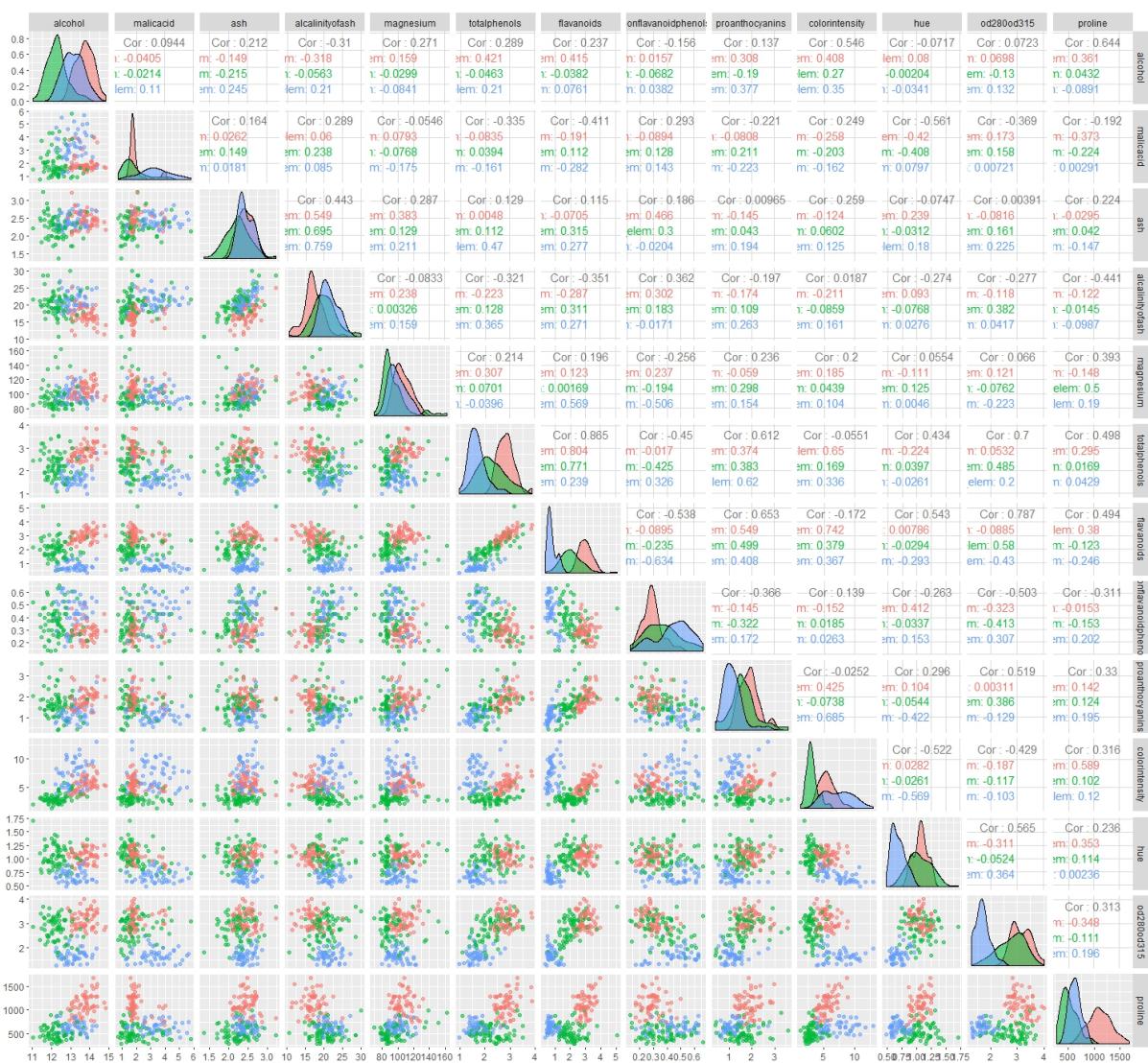
1.2. Wine Dataset

Zbiór danych jest wynikiem analizy chemicznej win uprawianych w tym samym regionie we Włoszech, ale uzyskanych z trzech różnych odmian. W analizie określono 13 składników znalezionych w każdym z trzech rodzajów win.

- Liczba atrybutów: 13.
- Rodzaj atrybutów: wartości typu Float i Intiger.
- Liczba instancji: 178.
- Liczba klas: 3



Rys. 1.3: Rozłożenie ilościowe poszczególnych klas zbioru Wine

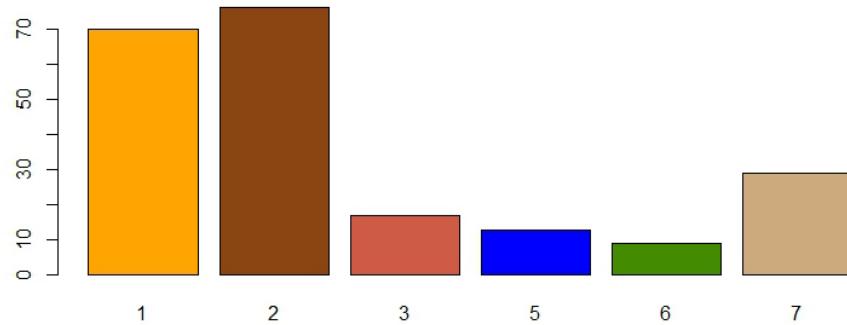


Rys. 1.4: Zestawienie zależności atrybutów i klas zbioru Wine

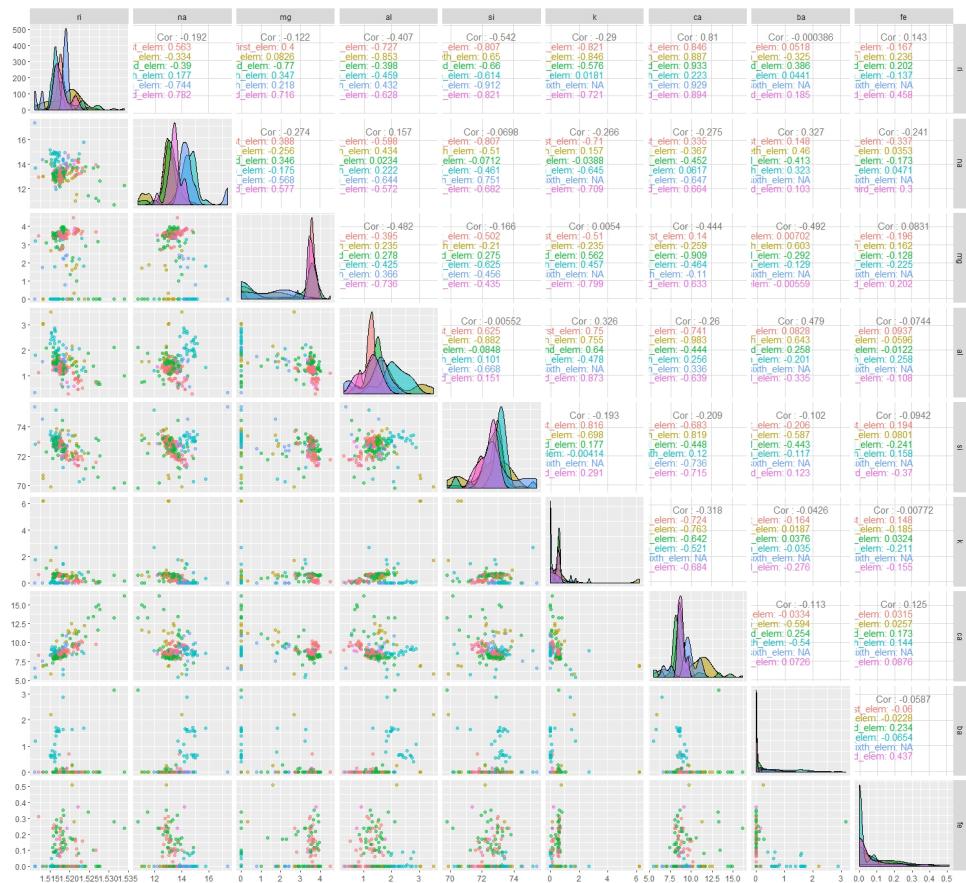
1.3. Glass Dataset

Zbiór danych powstał w wyniku motywacji badania dochodzeń kryminologicznych. Poprawne zidentyfikowanie rodzaju szkła znalezioneego na miejscu przestępstwa, na postawie jego składu pozwala na użycie go jako dowodu w sprawie.

- Liczba atrybutów: 9.
- Rodzaj atrybutów: realistyczne, ciągłe.
- Liczba instancji: 214.
- Liczba klas: 7.



Rys. 1.5: Rozłożenie ilościowe poszczególnych klas zbioru Glass

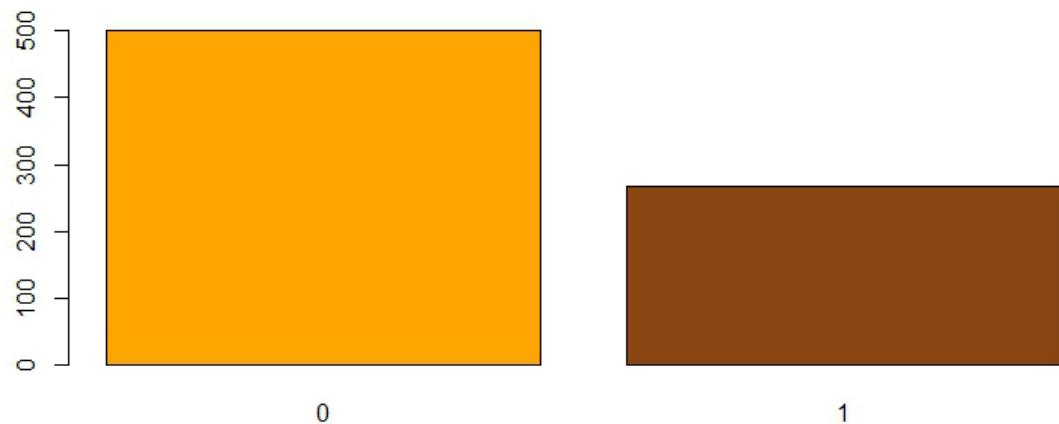


Rys. 1.6: Zestawienie zależności atrybutów i klas zbioru Glass

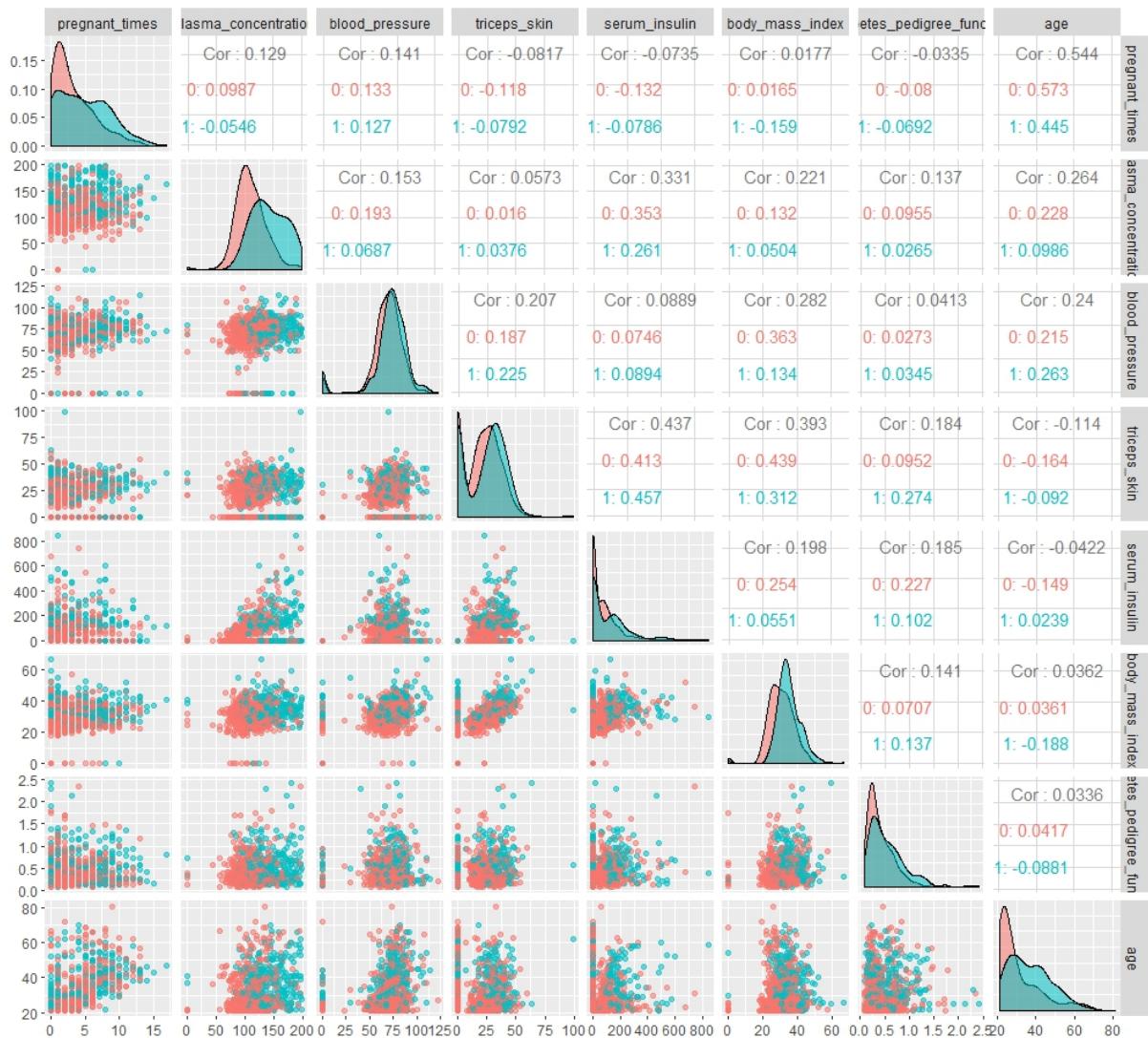
1.4. Pima Indians Diabetes Dataset

Zbiór danych Pima Indian Diabetes przewidywanie wystąpienie cukrzycy w oparciu o badania diagnostyczne. Pochodzi on z *National Institute of Diabetes and Digestive and Kidney Diseases*. Zawiera on dane dotyczące zachorowań na cukrzycę wśród kobiet z indiańskiego plemienia Pima. Każdy z 768 obiektów zbioru opisany jest przy pomocy 8 cech zawierających następujące informacje: ile razy pacjentka była w ciąży, test tolerancji glukozy, ciśnienie rozkurczowe, grubość zagięcia skóry, poziom insuliny, masę ciała, czy ktoś w rodzinie był chory na cukrzycę oraz wiek pacjentki. Każdy z obiektów przynależy do jednej z dwóch klas. Pierwsza klasa oznacza, że pacjentka nie choruje na cukrzycę, a druga klasa oznacza, że dana kobieta jest diabetykiem.

- Liczba atrybutów: 8.
- Rodzaj atrybutów: realistyczne, ciągłe i typu Intiger (wiek i ilość dotychczasowych ciąży).
- Liczba instancji: 768.
- Liczba klas: 2 - wartość 1 (pozytywna) lub 0 (negatywna).



Rys. 1.7: Rozłożenie ilościowe poszczególnych klas zbioru Diabetes

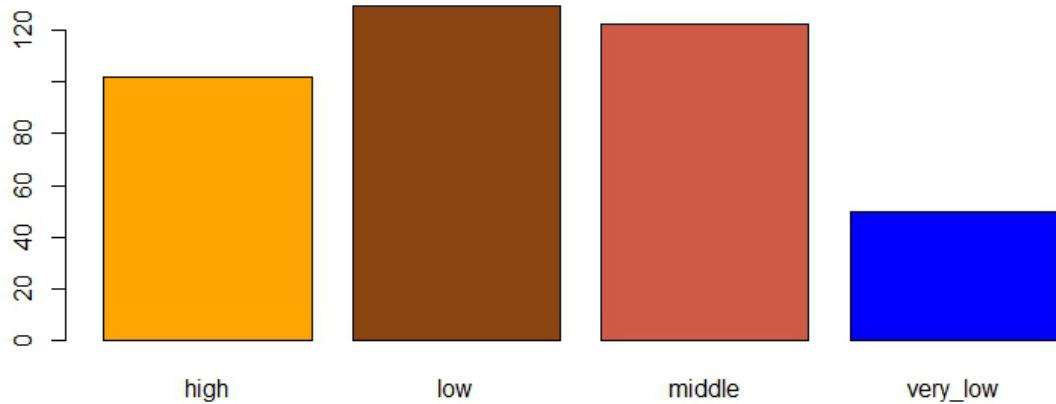


Rys. 1.8: Zestawienie zależności atrybutów i klas zbioru Diabetes

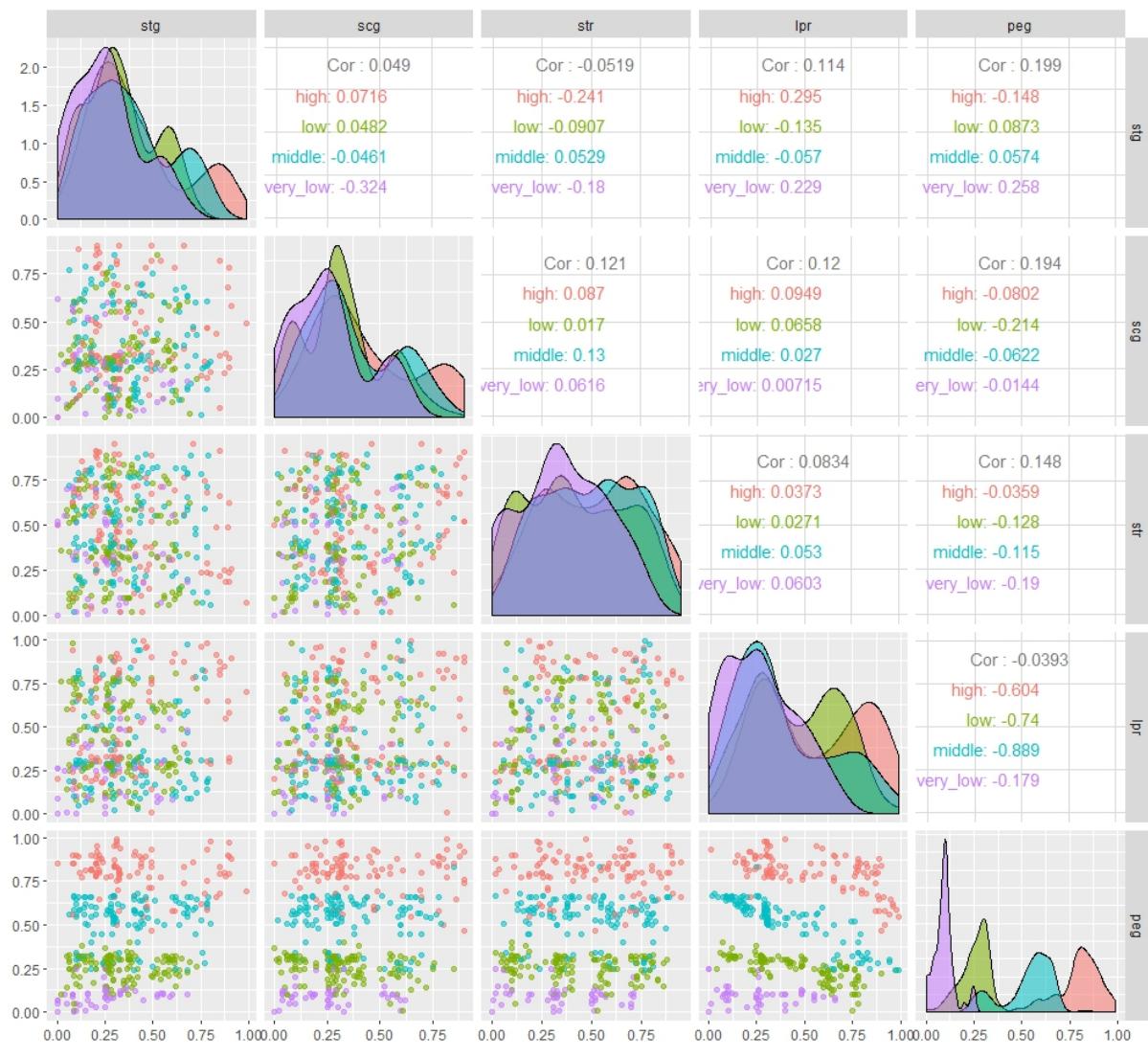
1.5. User Knowledge Modeling Dataset

Zbiór danych User Knowledge został sklasyfikowany przez autorów za pomocą intuicyjnego klasyfikatora wiedzy (technika hybrydowego uczenia maszynowego metod k-NN i meta-heurystycznych) oraz algorytmu k najbliższych sąsiadów. Zawiera 5 atrybutów opisujących m.in. stopień czasu nauki oraz liczbę powtórzeń koniecznych do osiągnięcia danego celu, czy chociażby osiągnięte rezultaty na egzaminie.

- Liczba atrybutów: 5.
- Rodzaj atrybutów: realistyczne, ciągłe.
- Liczba instancji: 403.
- Liczba klas: 4.



Rys. 1.9: Rozłożenie ilościowe poszczególnych klas zbioru User Knowledge



Rys. 1.10: Zestawienie zależności atrybutów i klas zbioru User Knowledge

Rozdział 2

Wstęp teoretyczny

2.1. Algorytm K-Means

Algorytm k-średnich (*ang. k-means*) inaczej zwany również algorytmem centroidów, służy do podziału danych wejściowych na **z góry założoną liczbę klas**. Jest to jeden z algorytmów stosowany w klasteryzacji (grupowaniu) i jest częścią uczenia nienadzorowanego w Machine Learning'u. W przypadku wykorzystania tego algorytmu, elementarnym elementem jest pojęcie **centroidu**, czyli reprezentanta skupienia (środka danej grupy).

Sposób działania algorytmu:

1. Wybór ilości centroidów i początkowe ułożenie ich w przestrzeni.
2. Ustalenie przynależności punktów do naniesionych centroidów.
3. Aktualizacja położenia centroidów, np. poprzez średnią arytmetyczną współrzędnych wszystkich punktów mających jego grupę.
4. Powtórzenie kroków drugiego i trzeciego aż do osiągnięcia kryterium zbieżności, którym najczęściej jest stan w którym nie zmieniła się przynależność punktów do klas.

Zalety:

- Niska złożoność, a co za tym idzie wysoka wydajność działania.
- Przy dużych zbiorach i niskich ilościach grupa algorytm ten będzie zdecydowanie szybszy niż pozostałe algorytmy tej klasy.
- Pogrupowane zbiory są z reguły ciasniejsze i bardziej zbite.

Wady:

- Nie pomaga w określeniu ilości grup (K).
- Różne wartości początkowe prowadzą do różnych wyników.
- Według literatury działa dobrze tylko dla „sferycznych” skupisk o jednorodnej gęstości.

Podsumowanie:

- Liczba grup, na które ma zostać podzielony zbiór jest z góry podawana na wejściu algorytmu.

- Algorytm działa do momentu stabilizacji, czyli gdy nie dochodzi już do żadnych zmian w używanych grupach.
- Wykorzystuje miarę odległości między punktami (zwykle euklidesową).

Parametry funkcji `kmeans()`:

- **x** - macierz z danymi (atrybutami) podlegającymi grupowaniu,
- **centers** - liczba skupień, które algorytm ma utworzyć, albo podane początkowe centra skupień. Jeśli jest to liczba skupień, wówczas procedura wyboru potencjalnych centrów skupień odbywa się w sposób losowy,
- **iter.max** - maksymalna liczba iteracji,
- **nstart** - jeśli w *center* podano liczbę grup, algorytm spróbuje wykonać X (podane jako parametr) różnych losowych ustawień początkowych, a następnie wybierze to z najniższą zmianą w obrębie klastrów.
- **algorithm** - określa, który algorytm będzie wykonany spośród dostępnych: Hartigan and Wong (1979) (domyślny), MacQueen (1967), Lloyd (1957) czy Forgy (1965).

2.2. Algorytm Partition Around Medoids (PAM)

PAM jest realizacją metody k-medoidowej, czyli takiej techniki grupowania, która dzieli zbiór danych zawierających n obiektów na k grup znanych a priori. Przydatnym narzędziem do oceny jakości grupowania jest **sylwetka** (ang. *silhouette*). Oblicza się ją dla każdego obiektu podlegającego grupowaniu. Na jej podstawie można stwierdzić, czy obiekty zostały prawidłowo zgrupowane, czy też trafiły do złych klastrów.

Działanie tego algorytmu zbliżone jest do działania algorytmu centroidów, z tą różnicą, że centroidy zostają tu zastąpione przez medoidy, czyli najbardziej centralne obiekty ze zbioru danych reprezentujące daną grupę, dla których odległość od wszystkich pozostałych elementów wewnętrz danej grupy jest minimalna. Algorytm ten dąży do **minimalizacji sumy odległości wszystkich elementów** niebędących medoidami od najbliższych im medoidów.

Sposób działania algorytmu:

Faza budowy:

1. Podział zbioru danych na k skupień z przypisanymi k medoidami.
2. Obliczenie macierzy odległości pomiędzy medoidami oraz pozostałymi obserwacjami.
3. Przypisanie każdej z obserwacji (nie będącą medoidem) do najbardziej zbliżonego skupienia.

Faza zamiany:

1. Przy użyciu iteracji zastąpienie jednego z medoidów jednym z niemedoidów i sprawdzenie, czy odległości wszystkich elementów niebędących medoidami od najbliższych im medoidów są mniejsze.
2. Jeśli nastąpiła przynajmniej jedna zamiana medoidów, powrót do punktu 3 fazy budowy. Jeśli nie, zakończenie algorytmu.

Zalety:

- Odporność na obserwacje odstające (*ang. outliers*).
- Odporność na szумy występujące w danych (*ang. robustness*).
- Pozwala wykorzystać inne miary odległości.

Wady:

- Brak możliwości zastosowania tej metody dla dużych zbiorów danych.
- Wyższa złożoność obliczeniowa, przez co słaba skalowalność.

Parametry funkcji *pam()*:

- **x** - macierz z danymi (atrybutami) podlegającymi grupowaniu lub macierz odmiенноści w zależności od wartości parametru *diss*. W przypadku macierzy danych, wszystkie zmienne muszą mieć wartość numeryczną. Brakujące wartości (NA) są dozwolone o ile każda para obserwacji ma co najmniej jeden przypadek, którego nie brakuje,
- **k** - liczba skupień, które algorytm ma utworzyć. Musi być mniejsza lub równa niż liczba obserwacji,
- **diss** - flaga logiczna; True - *x* traktowane jako macierz odmiенноści; False - *x* traktowane jako macierz danych,
- **metric** - wartość typu string określająca metrykę wykorzystywaną do obliczania wartości *odmiенноści* (*ang. dissimilarities*) pomiędzy obserwacjami. Obecnie dostępne opcje to „euclidean” and „manhattan”. Odległości euklidesowe są sumą kwadratów różnic kwadratowych, a odległości manhattan są sumą różnic bezwzględnych. Jeśli *x* jest już macierzą odmiенноści, to ten argument zostanie zignorowany,
- **medoids** - domyślnie wartość *NULL* lub wektor długości *k* liczb naturalnych określających numery obiektów przyjętych jako medoidy inicjalne w przypadku rezygnacji z „fazy budowy” algorytmu,
- **stand** - flaga logiczna; True - pomiary z *x* są standaryzowane przed obliczeniem *odmiенноści*. Jeśli *x* jest już macierzą odmiенноści, to ten argument zostanie zignorowany,
- **cluster.only** - wartość logiczna TRUE, gdy działanie *pam()* ogranicza się tylko do grupowania, wartość logiczna FALSE – gdy mają być obliczone również pozostałe wartości funkcji,
- **do.swap** - wartość logiczna TRUE, gdy *Swap Phase* ma być przeprowadzona, wartość logiczna FALSE – w przeciwnym przypadku.

2.3. Wybrane miary jakości klasteryzacji

- **Davies-Bouldin Index (DBI)**

Indeks ten jest obliczany według następującej formuły:

$$DB = \frac{1}{n} \sum_{i=1}^n \max_{j \neq i} \left(\frac{\theta_i + \theta_j}{d(c_i, c_j)} \right), \quad (2.1)$$

gdzie n jest liczbą klastrów, c_x jest centroidem (śrokiem ciężkości) klastra x , σ_x średnią odległośćą wszystkich elementów w klastrze x do centroidu c_x i $d(c_i, c_j)$ to odległość między centroidami c_i i c_j . Algorytmy generujące klastry o niskich odległościach wewnętrz klastra i dużych odległościach między klastrami, będą miały niski indeks Daviesa-Bouldina. Algorytm, który tworzy zbiór klastrów z najmniejszym indeksem Daviesa-Bouldina jest uważany za najlepszy algorytm oparty na tym kryterium.

- **Dunn Index**

Indeks Dunn'a ma na celu zidentyfikowanie gęstych i dobrze oddzielonych klastrów. Definiuje się go jako stosunek minimalnej odległości między klastrami do maksymalnej odległości wewnętrz klastra. Dla każdej partycji klastra indeks Dunn'a można obliczyć według następującej formuły:

$$D = \frac{\min_{1 \leq i < j \leq n} d(i, j)}{\max_{1 \leq k \leq n} d'(k)}, \quad (2.2)$$

gdzie $d(i, j)$ reprezentuje odległość między klastrami i i j , a $d'(k)$ mierzy odległości wewnętrz klastrowe klastra k . Odległość międzyklastrowa $d(i, j)$ między dwoma klastrami może być dowolną liczbą miar odległości, taką jak np. odległość między centroidami klastra. Podobnie, odległość wewnętrz klastra $d'(k)$ może być mierzona na różne sposoby, na przykład jako maksymalna odległość między dowolną parą elementów w klastrze k . Ponieważ wewnętrzne kryterium poszukuje klastrów o wysokim podobieństwie wewnętrz klastra i niskim podobieństwie między skupiskami, bardziej pożądane są algorytmy generujące klastry z wysokim indeksem Dunn'a.

- **Rand Measure**

Indeks Rand'a oblicza, jak podobne są klastry do klasyfikacji porównawczych. Można również patrzeć na indeks Rand jako na miarę procentu poprawnych decyzji podejmowanych przez algorytm. Jest on wyliczany za pomocą następującej formuły:

$$RI = \frac{TP + TN}{TP + FP + FN + TN}, \quad (2.3)$$

gdzie TP to liczba prawdziwych pozytywów, TN to liczba prawdziwych negatywów, FP to liczba fałszywych trafień, a FN to liczba fałszywych negatywów. Jednym z problemów z indeksem Rand jest to, że wartości fałszywie dodatnie i fałszywie negatywne są jednakowo ważone. Może to być niepożądana cecha niektórych aplikacji klastrowych. Miara *F-measure* rozwiązuje ten problem, podobnie jak skorygowany o szansę indeks Rand (*adjusted Rand index*).

- **Silhouette**

Miara sylwetki (*ang. silhouette*) porównuje średnią odległość do elementów w tym samym klastrze ze średnią odległością do elementów w innym klastrze. Im wyższa wartość tego parametru, tym lepsza klasteryzacja. Indeks Silhouette dobrze się sprawdza m.in. w metodzie k-means, gdzie znajduje zastosowanie do określania optymalnej liczby klastrów. Każdemu obiektowi przyporządkowana jest miara:

$$s(x_i) = \frac{b(x_i) - a(x_i)}{\max\{a(x_i), b(x_i)\}}, \quad (2.4)$$

Najpierw dla x_i znajduje się jego średnią odległość $a(x_i)$ od pozostałych obiektów grupy, do której został przydzielony, a następnie wybiera się minimalną wartość $b(x_i)$ spośród obliczonych odległości od x_i do każdej spośród pozostałych grup osobno; odległość x_i od danej grupy oblicza się jako średnią odległość od x_i do wszystkich elementów tej grupy.

Miara S_i przyjmuje wartości z przedziału $<-1,1>$. Wartość bliska 1 oznacza, że element jest prawidłowo przyporządkowany do klastra, natomiast bliska -1 mówi nam, że lepsze byłoby przypisanie i-tego elementu do klastra sąsiadniego. S_i bliskie 0 oznacza, że obiekt leży na granicy dwóch klastrów i pasuje do nich w równym stopniu. Dla każdego klastra oblicza się średnią wartość miary S_i obiektów wchodzących w jego skład i oznacza się SI_c , gdzie c jest numerem danego klastra. Określa ona jak gęsto są zgrupowane obiekty leżące wewnątrz klastra. SI_c dla całego zestawu danych jest miarą jakości klasteryzacji. Jeśli klastrów jest za dużo lub za mało, to część z nich będzie miała zbyt wąską sylwetkę w stosunku do pozostałych.

- **Purity**

Purity jest miarą określającą, w jakim stopniu klastry zawierają pojedynczą klasę. Jego obliczenia można sobie wyobrazić w następujący sposób: Dla każdego klastra policz liczbę punktów danych z najbardziej powszechniej klasy we wspomnianym klastrze. Teraz weź sumę wszystkich klastrów i podziel przez całkowitą liczbę punktów danych. Formalnie, biorąc pod uwagę pewien zbiór klastrów M i pewien zestaw klas D , oraz instancje danych N obu partycji, *purity* można zdefiniować jako:

$$P = \frac{1}{N} \sum_{m \in M} \max_{d \in D} |m \cap d|, \quad (2.5)$$

Należy zauważyć, że ten wskaźnik nie obniża swojej oceny za posiadanie wielu klastrów. Na przykład wynik *purity* wynoszący 1 jest możliwy dzięki umieszczeniu każdego punktu danych we własnym klastrze.

2.4. Miary jakości klasyfikatora

- **Trafność (ang. Accuracy)** - określa, jaka część prognozowanych etykiet jest zgodna z rzeczywistymi wynikami. Oznacza to procent poprawnie sklasyfikowanych etykiet.
- **Precyzja (ang. Precision)** - określa liczbę adekwatnych elementów w zbiorze wyników. W kontekście klasyfikacji jest to liczba poprawnych etykiet z wszystkich zbiorów klasyfikowanych etykiet. Wyniki są uśredniane dla wszystkich etykiet.
- **Czułość (ang. Recall)** - określa liczbę poprawnych wyników względem liczby wszystkich poprawnych etykiet. W kontekście klasyfikacji jest to liczba poprawnie sklasyfikowanych etykiet w zbiorze podzielona przez łączną liczbę etykiet ze zbioru. Wyniki są uśredniane.
- **Wskaźnik F1 (ang. F1 Score)** - jest to średnia harmoniczna precyzji i czułości. Najczęściej stosowana jest dla niezrównoważonych zbiorów danych w celu ustalenia, czy klasyfikator działa dobrze dla wszystkich klas.

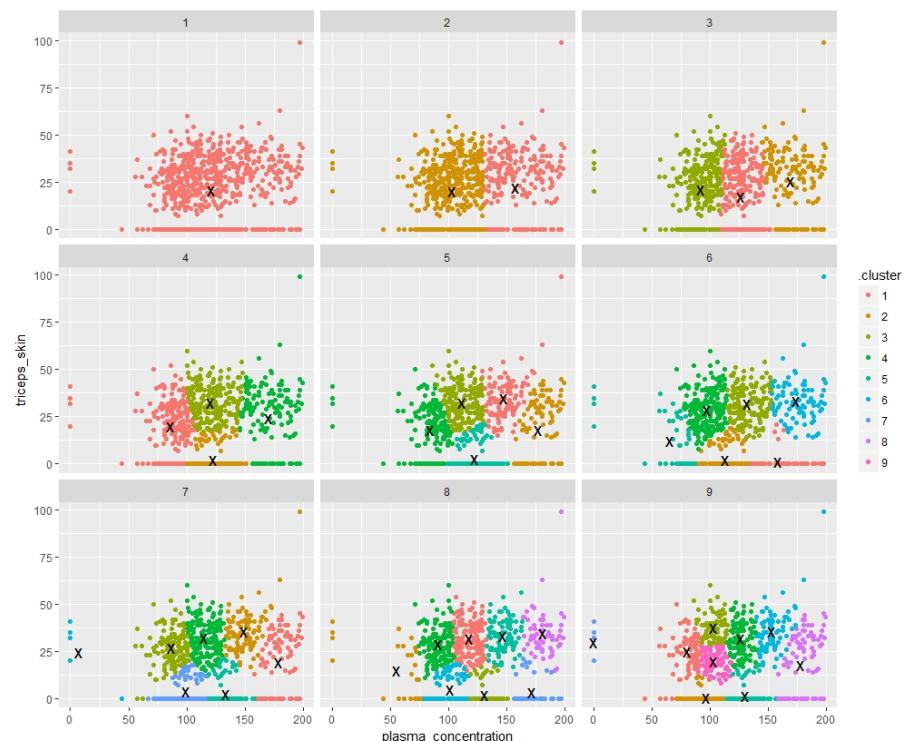
Rozdział 3

Badania i analiza wyników

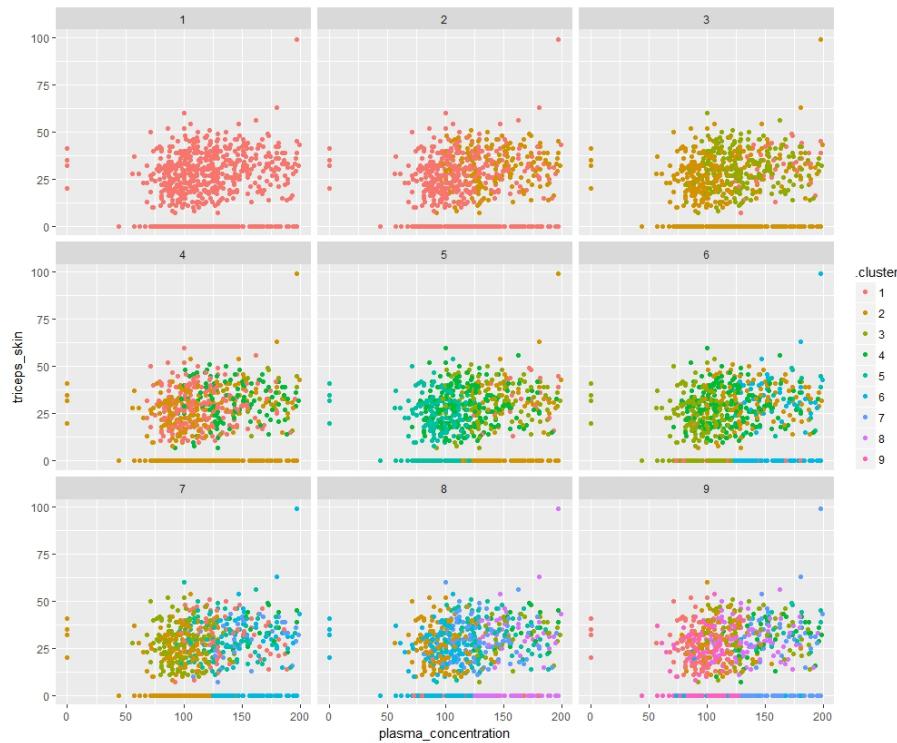
3.1. Analiza wyboru atrybutów do klasteryzacji

Wstępna analiza zbiorów oraz teorii operowania na klasteryzacji zbiorów danych wykazała, że najlepszym sposobem wyboru argumentów służących do dokonania klasteryzacji jest ich głęboka analiza. Na takową niestety ze względów czasowych i wydajnościowych nie możemy sobie pozwolić dlatego we wszystkich testach przyjęto strategię analizy wszystkich argumentów zbioru danych jaki on dysponuje.

Różnicę jaka występuje w momencie wyboru jedynie dwóch przykładowych atrybutów (rys. 3.1) oraz wyboru całego zbioru atrybutów (rys. 3.2) pokazują dwa poniższe obrazy.



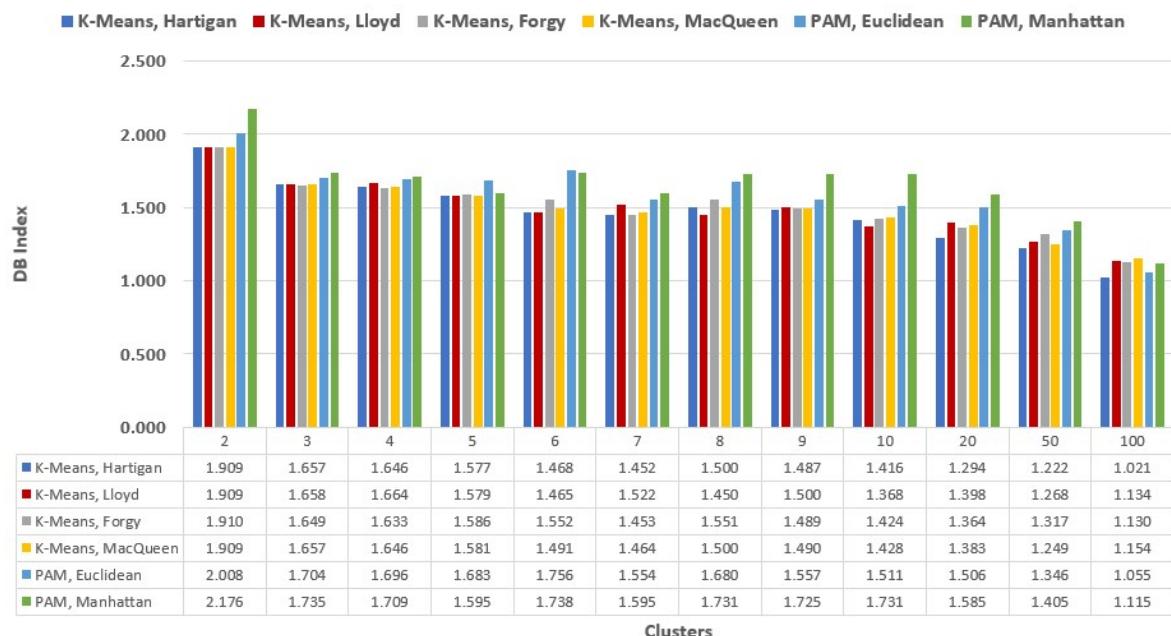
Rys. 3.1: Przykład (na zbiorze Diabetes) niepoprawnego doboru atrybutów do klasteryzacji

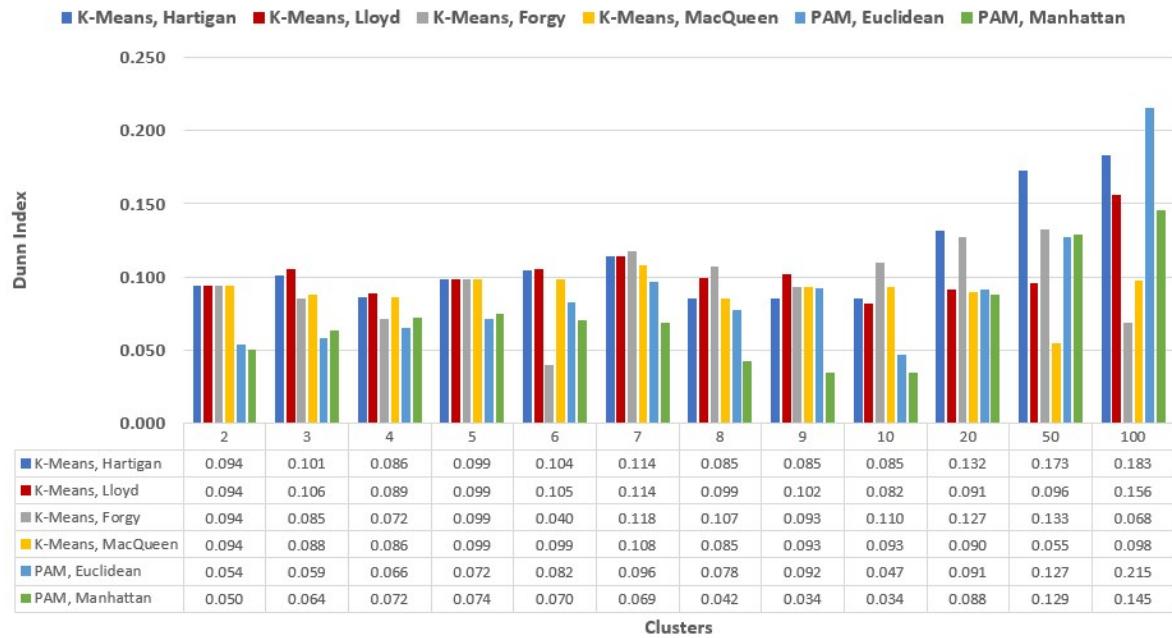


Rys. 3.2: Przykład (na zbiorze Diabetes) poprawnego doboru atrybutów do klasteryzacji

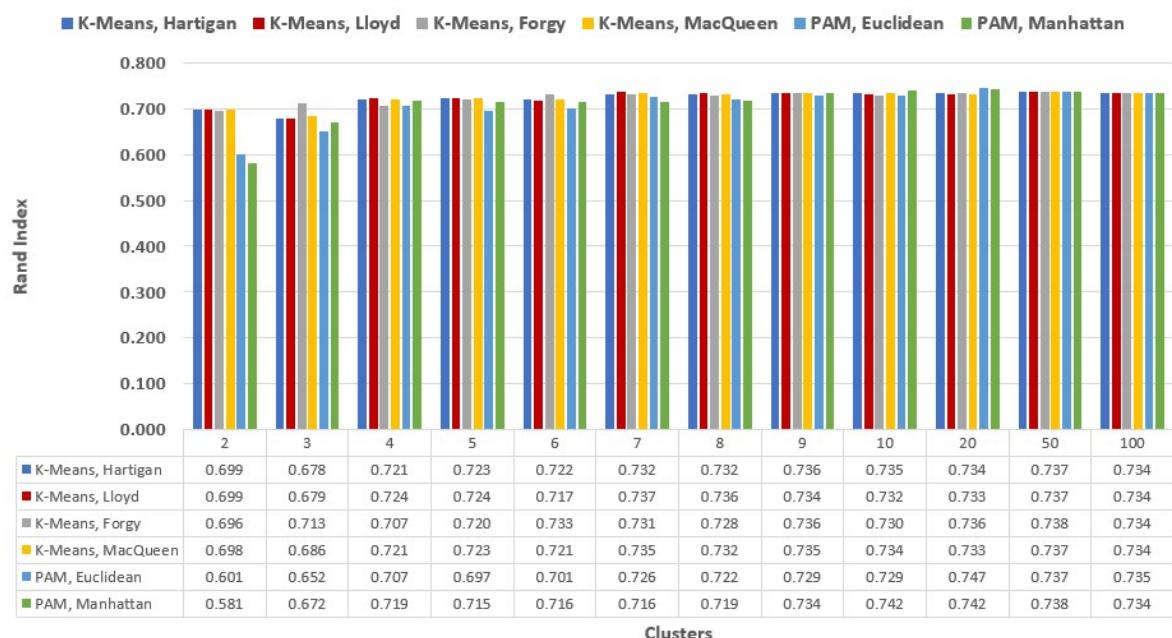
3.2. Analiza wyników klasteryzacji

3.2.1. User Knowledge Modeling Dataset

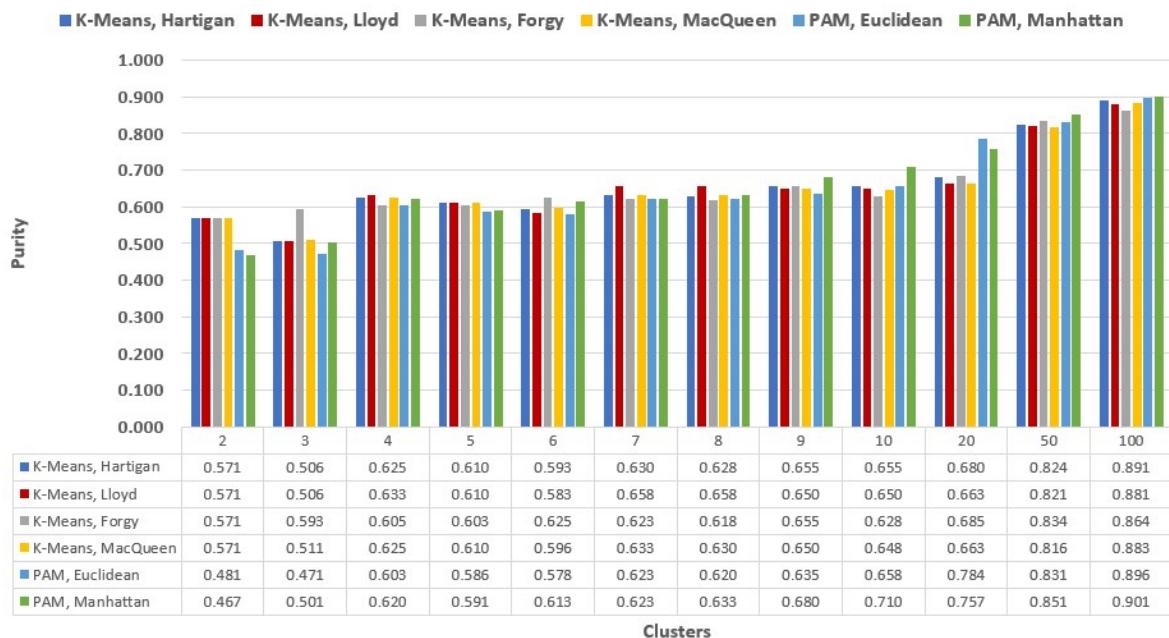
Rys. 3.3: Porównanie parametru *DB Index* dla zbioru User Knowledge



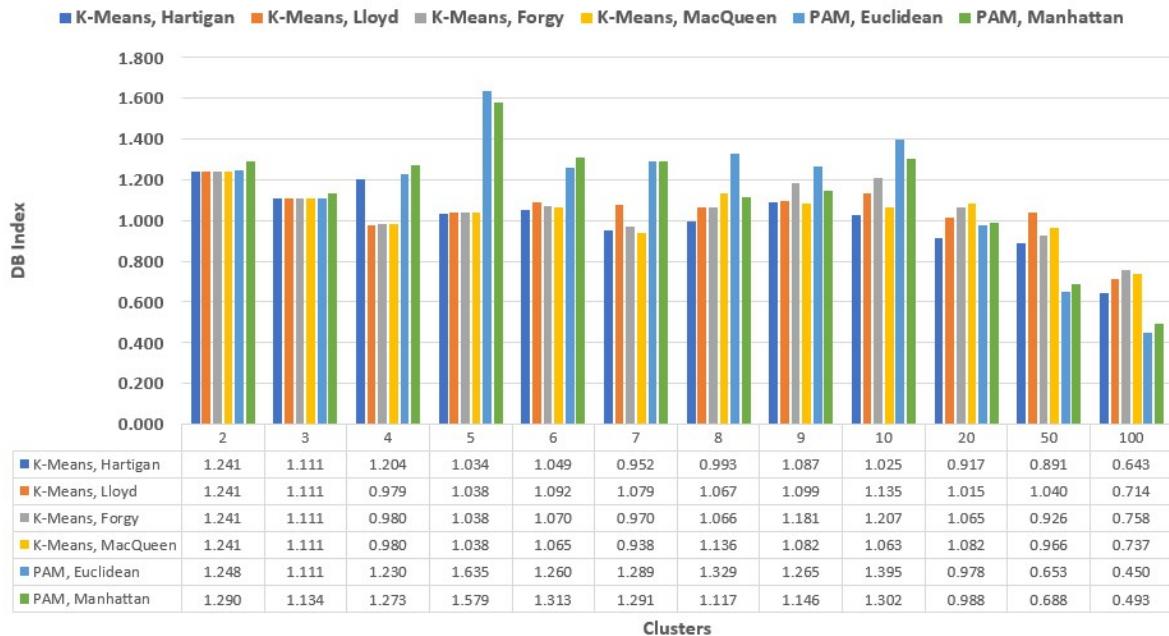
Rys. 3.4: Porównanie parametru Dunn Index dla zbioru User Knowledge

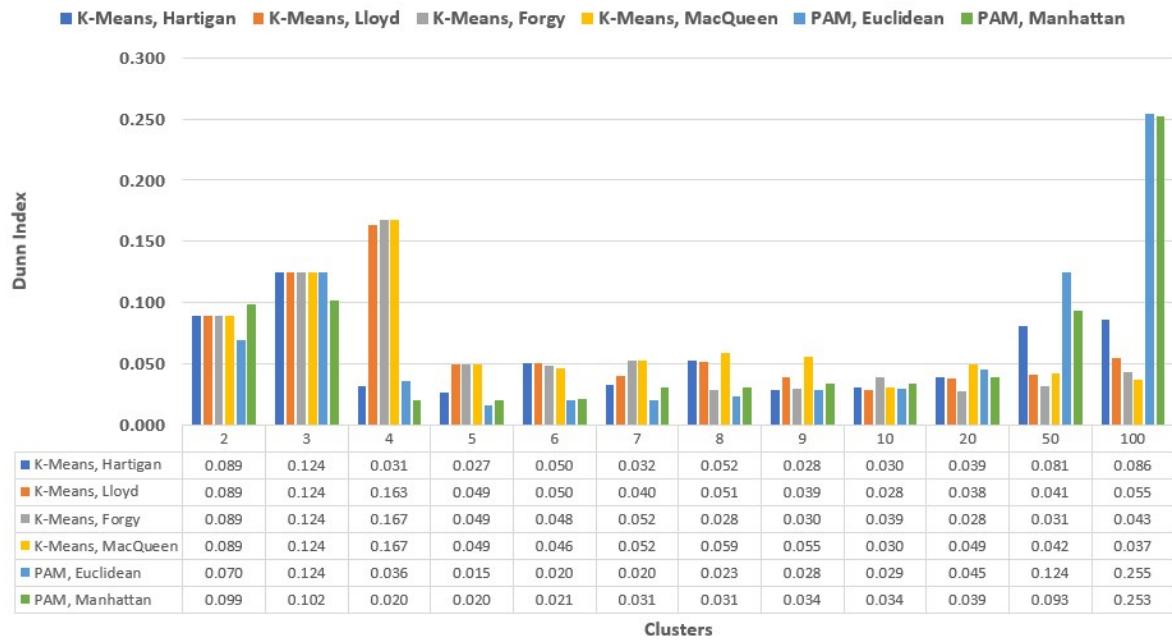
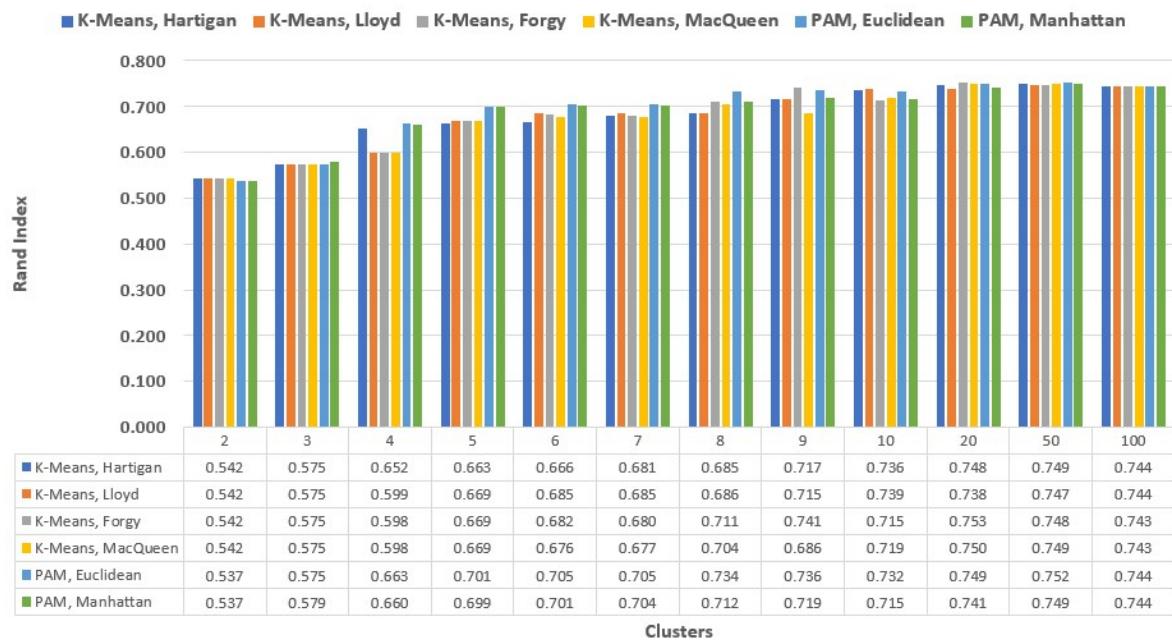


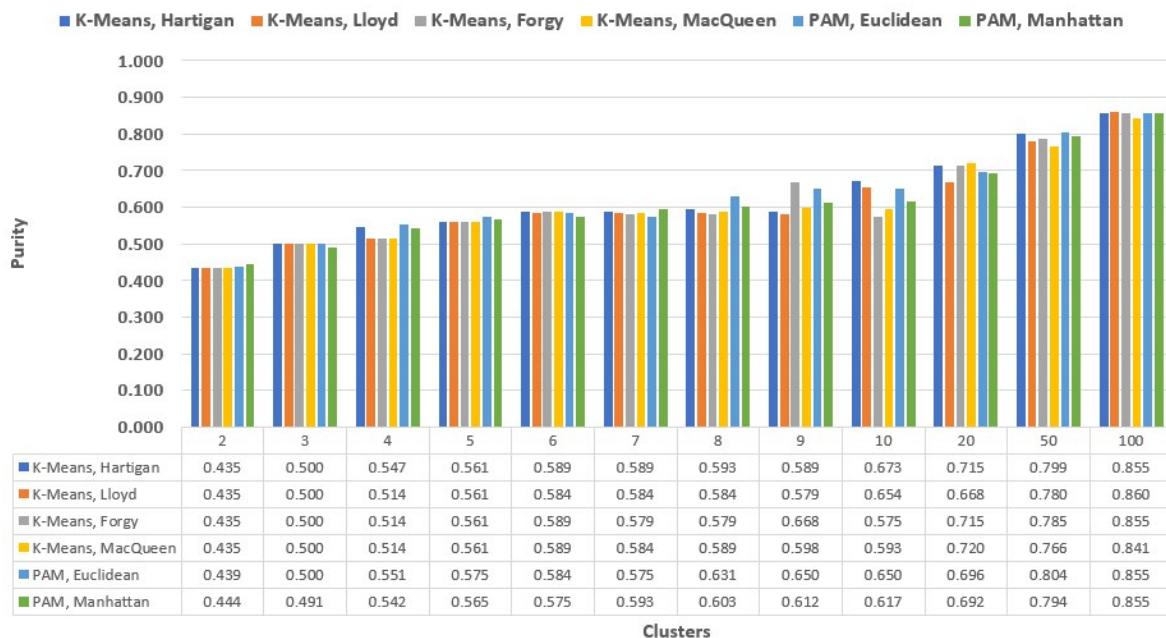
Rys. 3.5: Porównanie parametru Rand Index dla zbioru User Knowledge

Rys. 3.6: Porównanie parametru *Purity* dla zbioru User Knowledge

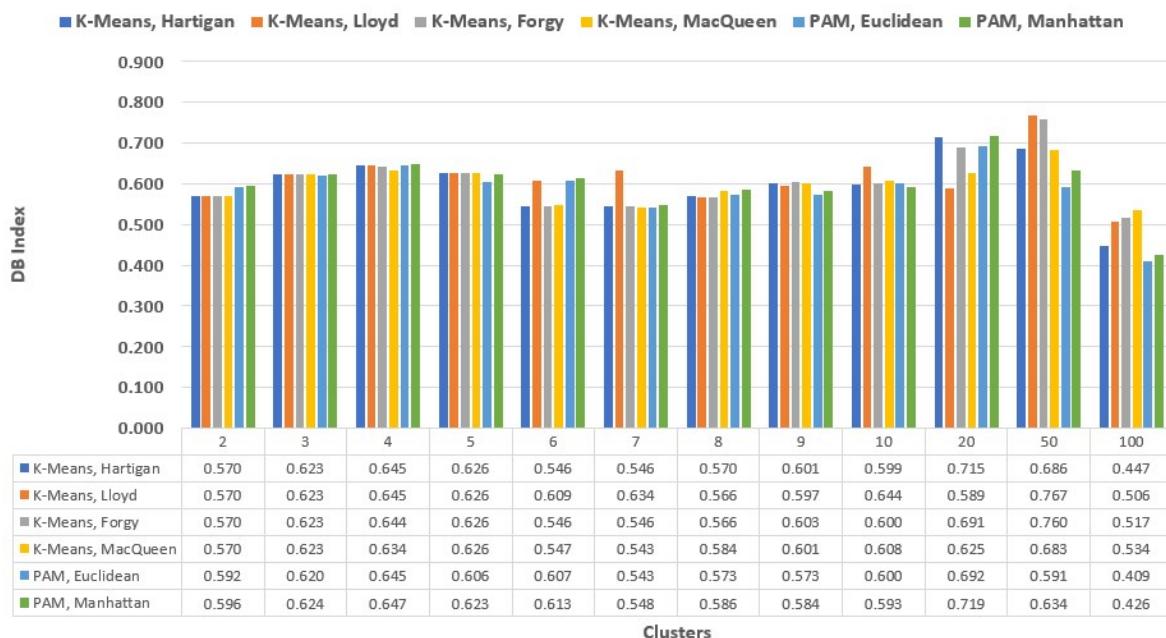
3.2.2. Wine Dataset

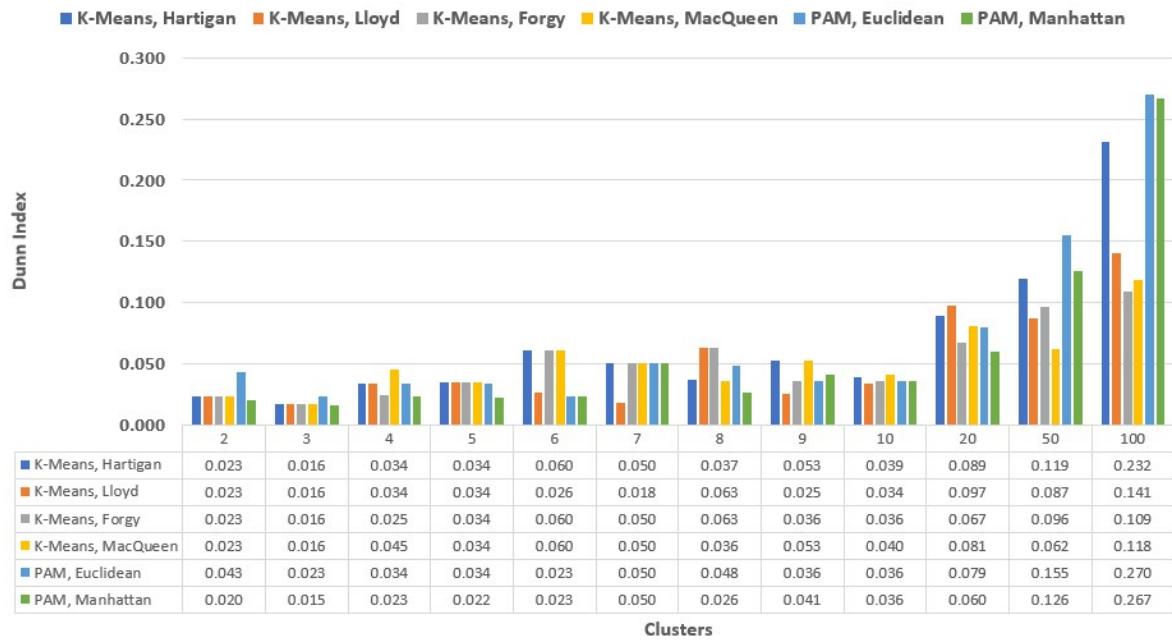
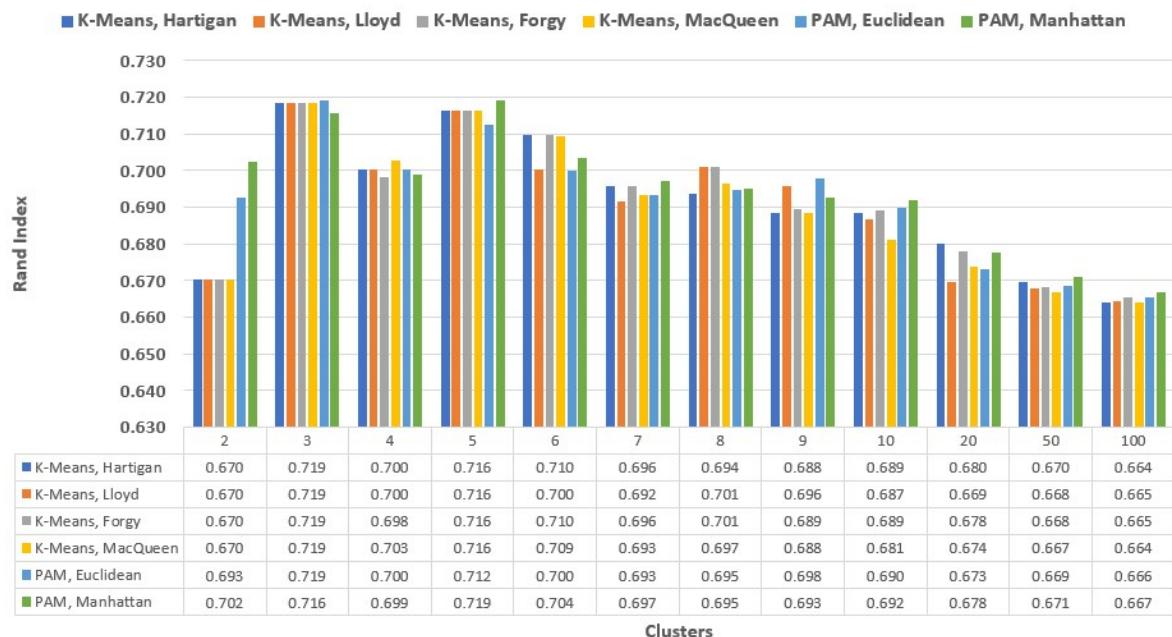
Rys. 3.7: Porównanie parametru *DB Index* dla zbioru Wine

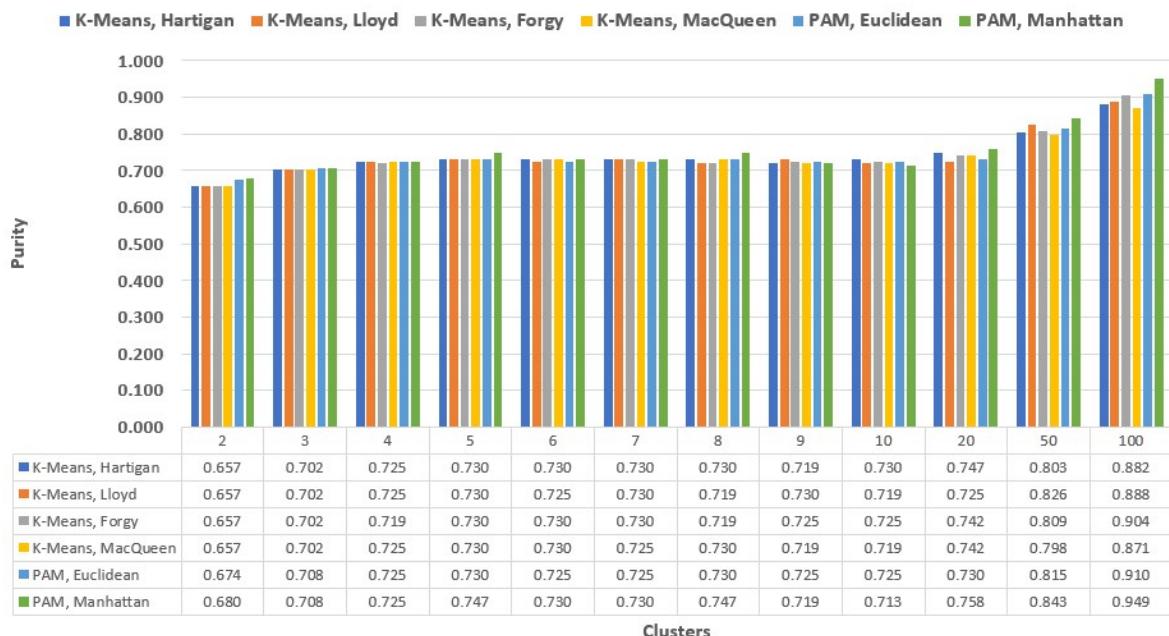
Rys. 3.8: Porównanie parametru *Dunn Index* dla zbioru WineRys. 3.9: Porównanie parametru *Rand Index* dla zbioru Wine

Rys. 3.10: Porównanie parametru *Purity* dla zbioru Wine

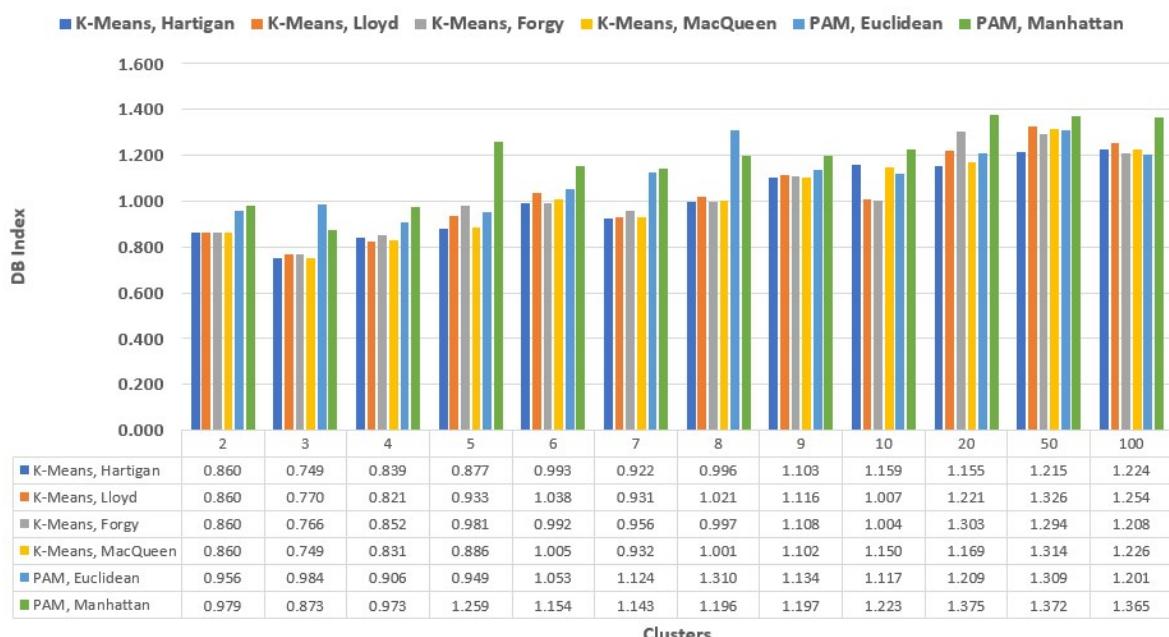
3.2.3. Glass Dataset

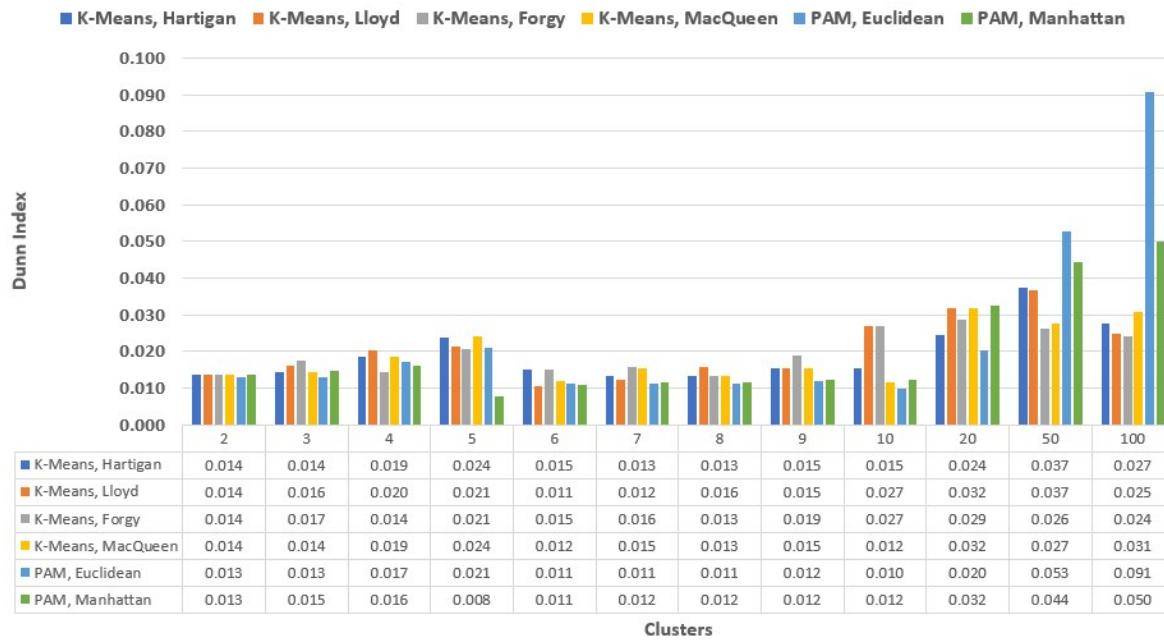
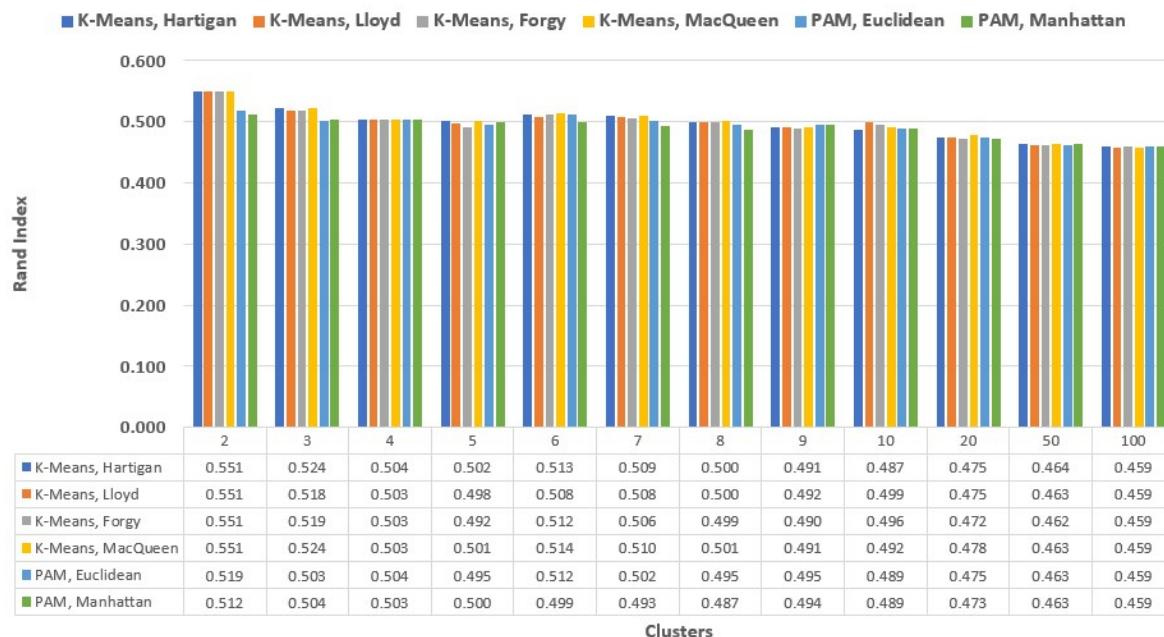
Rys. 3.11: Porównanie parametru *DB Index* dla zbioru Glass

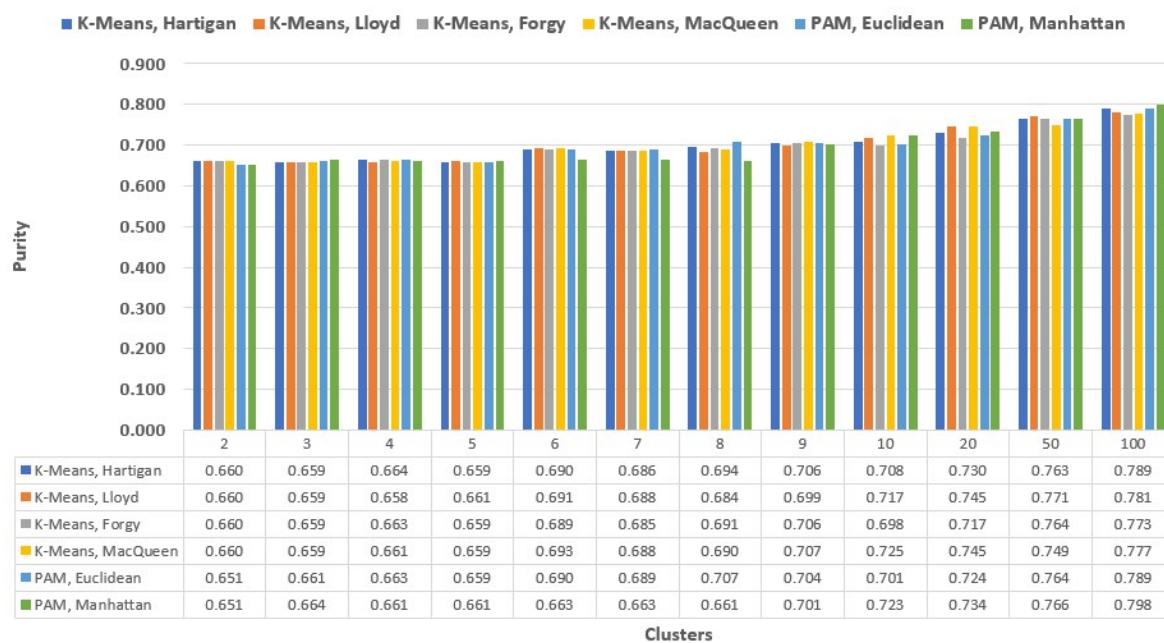
Rys. 3.12: Porównanie parametru *Dunn Index* dla zbioru GlassRys. 3.13: Porównanie parametru *Rand Index* dla zbioru Glass

Rys. 3.14: Porównanie parametru *Purity* dla zbioru Glass

3.2.4. Pima Indians Diabetes Dataset

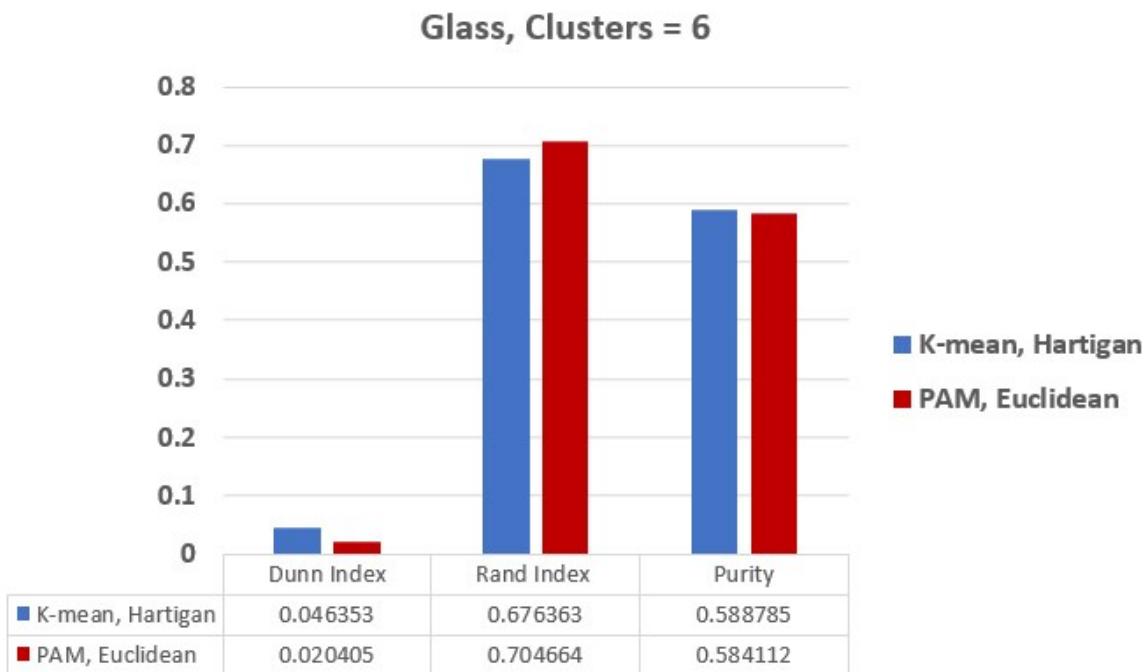
Rys. 3.15: Porównanie parametru *DB Index* dla zbioru Pima Diabetes

Rys. 3.16: Porównanie parametru *Dunn Index* dla zbioru Pima DiabetesRys. 3.17: Porównanie parametru *Rand Index* dla zbioru Pima Diabetes

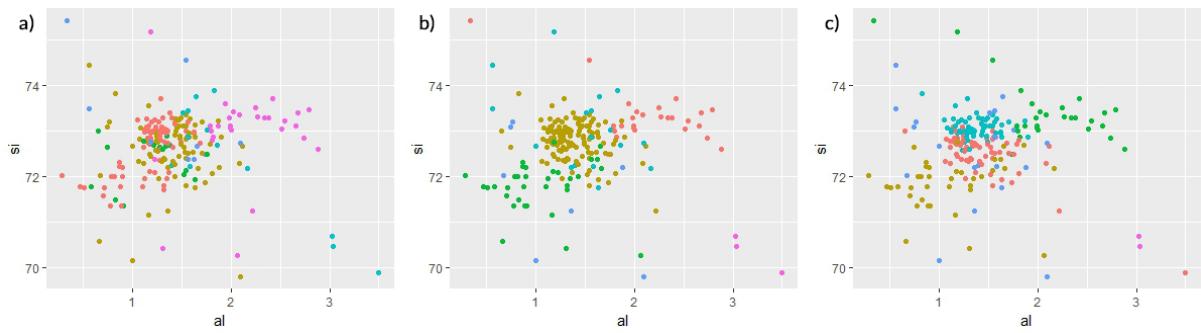
Rys. 3.18: Porównanie parametru *Purity* dla zbioru Pima Diabetes

3.3. Prezentacja najlepszych uzyskanych wyników

3.3.1. Glass Dataset



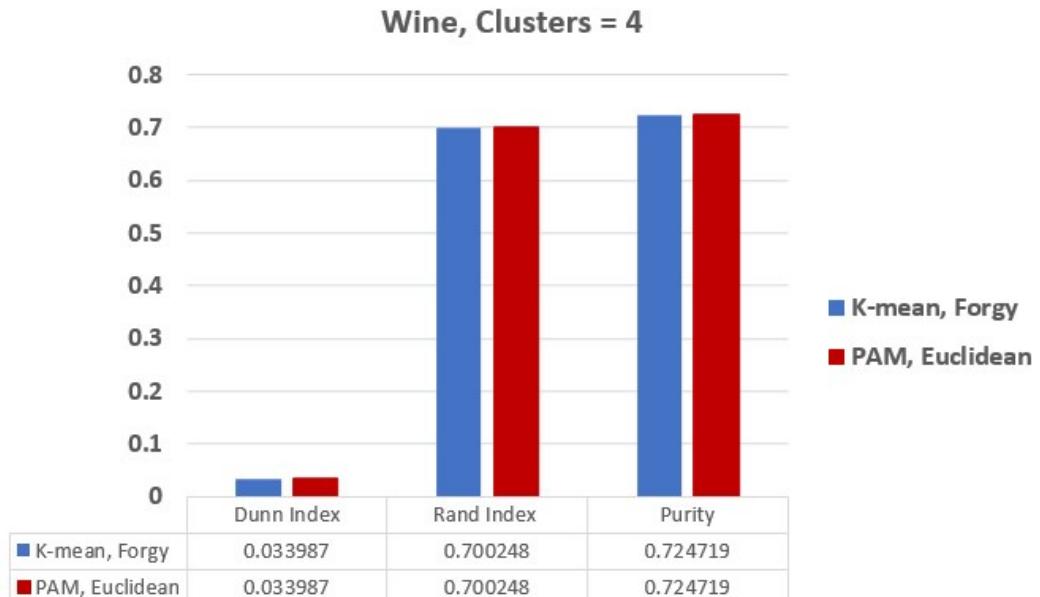
Rys. 3.19: Najlepsze uzyskane rezultaty dla zbioru Glass



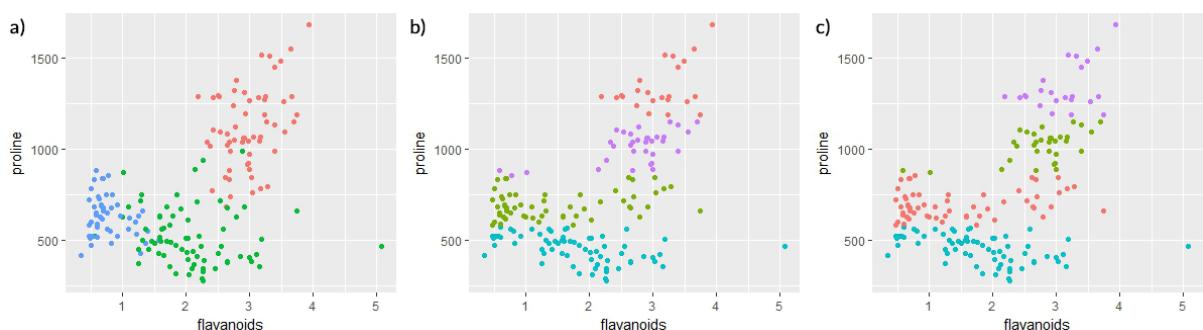
Rys. 3.20: Porównanie graficzne uzyskanych wyników dla zbioru Glass

diagramy zbiorów na rysunkach analogicznych do Rys.3.20 oznaczają: a) domyślny rozkład klas, b) rozkład klastrów dla algorytmu K-Means, c) rozkład klastrów dla algorytmu PAM.

3.3.2. Wine Dataset

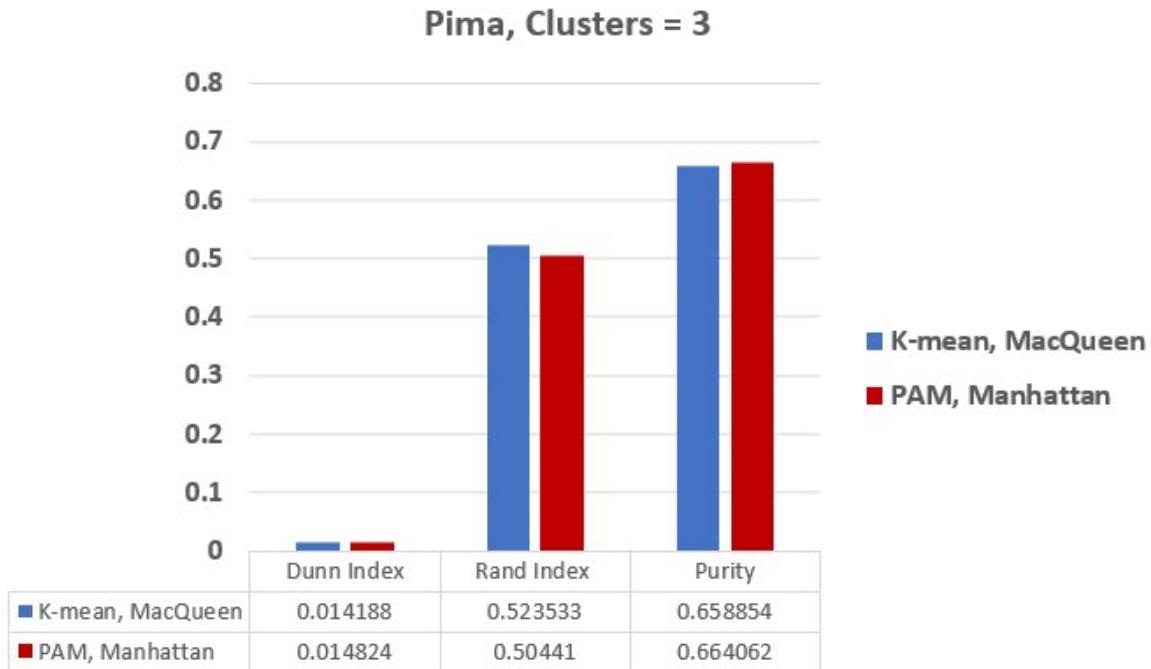


Rys. 3.21: Najlepsze uzyskane rezultaty dla zbioru Wine

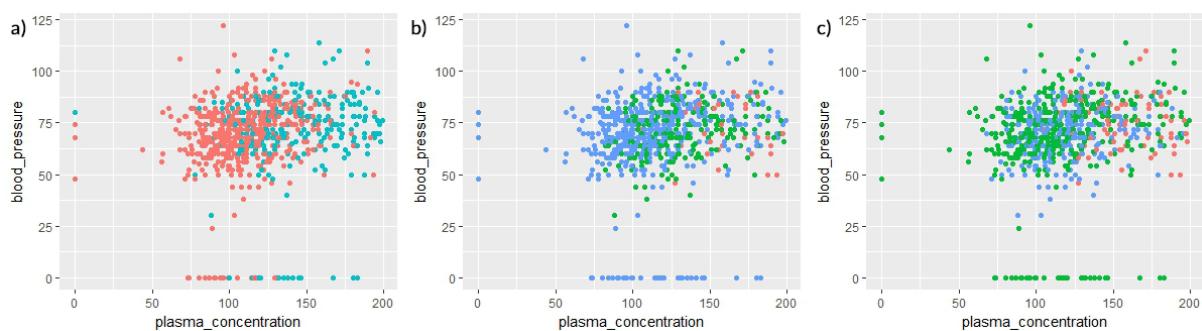


Rys. 3.22: Porównanie graficzne uzyskanych wyników dla zbioru Wine

3.3.3. Pima Indians Diabetes Dataset

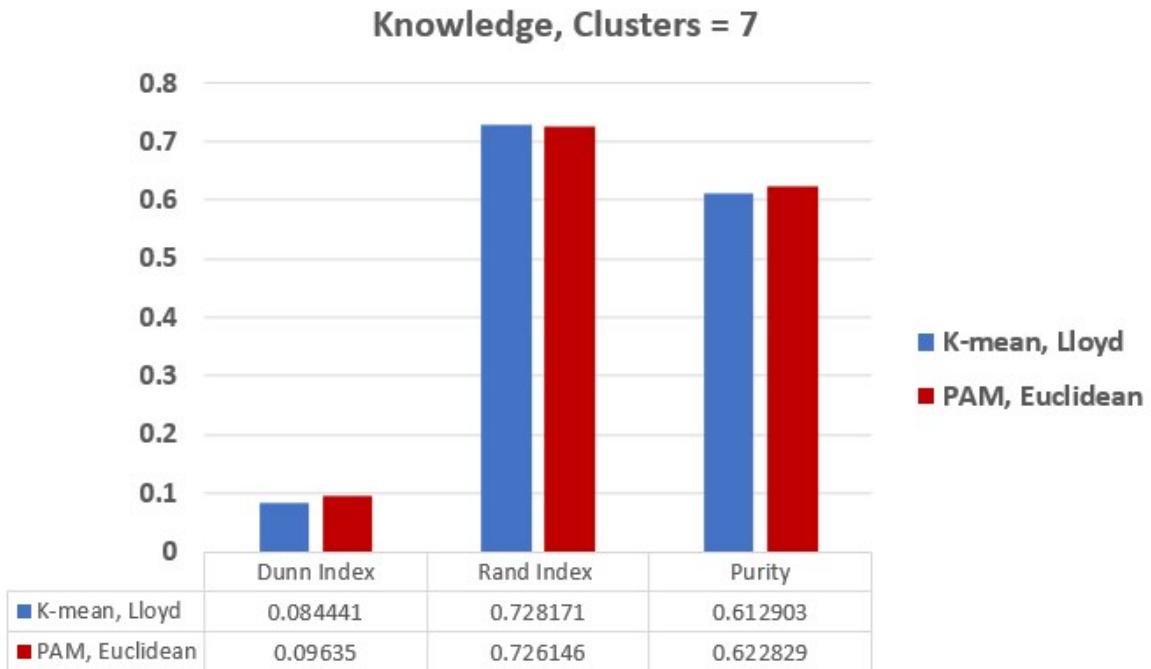


Rys. 3.23: Najlepsze uzyskane wyniki dla zbioru Pima Diabetes

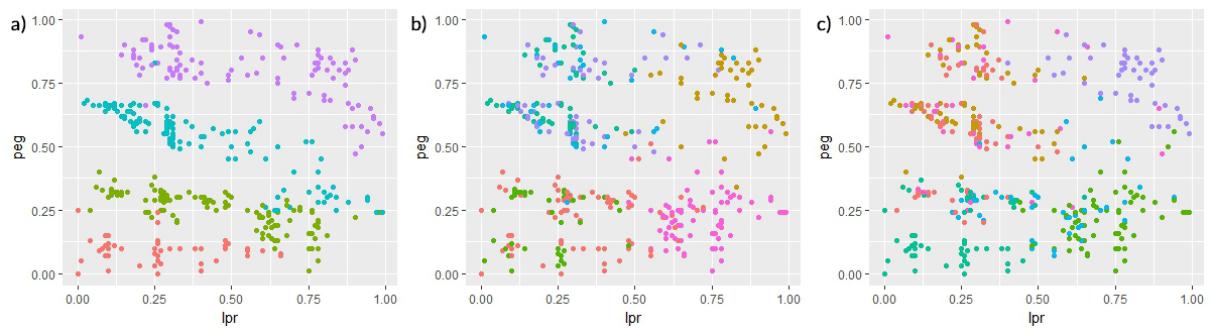


Rys. 3.24: Porównanie graficzne uzyskanych wyników dla zbioru Pima Diabetes

3.3.4. User Knowledge Modeling Dataset



Rys. 3.25: Najlepsze uzyskane rezultaty dla zbioru User Knowledge



Rys. 3.26: Porównanie graficzne uzyskanych wyników dla zbioru User Knowledge

Spis rysunków

1.1.	Rozłożenie ilościowe poszczególnych klas zbioru Iris	3
1.2.	Zestawienie zależności atrybutów i klas zbioru Iris	4
1.3.	Rozłożenie ilościowe poszczególnych klas zbioru Wine	5
1.4.	Zestawienie zależności atrybutów i klas zbioru Wine	5
1.5.	Rozłożenie ilościowe poszczególnych klas zbioru Glass	6
1.6.	Zestawienie zależności atrybutów i klas zbioru Glass	6
1.7.	Rozłożenie ilościowe poszczególnych klas zbioru Diabetes	7
1.8.	Zestawienie zależności atrybutów i klas zbioru Diabetes	8
1.9.	Rozłożenie ilościowe poszczególnych klas zbioru User Knowledge	9
1.10.	Zestawienie zależności atrybutów i klas zbioru User Knowledge	9
3.1.	Przykład (na zbiorze Diabetes) niepoprawnego doboru atrybutów do klasteryzacji .	16
3.2.	Przykład (na zbiorze Diabetes) poprawnego doboru atrybutów do klasteryzacji .	17
3.3.	Porównanie parametru <i>DB Index</i> dla zbioru User Knowledge	17
3.4.	Porównanie parametru <i>Dunn Index</i> dla zbioru User Knowledge	18
3.5.	Porównanie parametru <i>Rand Index</i> dla zbioru User Knowledge	18
3.6.	Porównanie parametru <i>Purity</i> dla zbioru User Knowledge	19
3.7.	Porównanie parametru <i>DB Index</i> dla zbioru Wine	19
3.8.	Porównanie parametru <i>Dunn Index</i> dla zbioru Wine	20
3.9.	Porównanie parametru <i>Rand Index</i> dla zbioru Wine	20
3.10.	Porównanie parametru <i>Purity</i> dla zbioru Wine	21
3.11.	Porównanie parametru <i>DB Index</i> dla zbioru Glass	21
3.12.	Porównanie parametru <i>Dunn Index</i> dla zbioru Glass	22
3.13.	Porównanie parametru <i>Rand Index</i> dla zbioru Glass	22
3.14.	Porównanie parametru <i>Purity</i> dla zbioru Glass	23
3.15.	Porównanie parametru <i>DB Index</i> dla zbioru Pima Diabetes	23
3.16.	Porównanie parametru <i>Dunn Index</i> dla zbioru Pima Diabetes	24
3.17.	Porównanie parametru <i>Rand Index</i> dla zbioru Pima Diabetes	24
3.18.	Porównanie parametru <i>Purity</i> dla zbioru Pima Diabetes	25
3.19.	Najlepsze uzyskane rezultaty dla zbioru Glass	25
3.20.	Porównanie graficzne uzyskanych wyników dla zbioru Glass	26
3.21.	Najlepsze uzyskane rezultaty dla zbioru Wine	26
3.22.	Porównanie graficzne uzyskanych wyników dla zbioru Wine	26

3.23. Najlepsze uzyskane rezultaty dla zbioru Pima Diabetes	27
3.24. Porównanie graficzne uzyskanych wyników dla zbioru Pima Diabetes	27
3.25. Najlepsze uzyskane rezultaty dla zbioru User Knowledge	28
3.26. Porównanie graficzne uzyskanych wyników dla zbioru User Knowledge	28