# ROZWÓJ I ZASTOSOWANIA SZTUCZNEJ INTELIGENCJI I EKSPLORACJI DANYCH W BANKOWOŚCI

## Wprowadzenie i Obszary Zastosowań

**Adrian Horzyk**

horzyk@agh.edu.pl

AGH

AKADEMIA GÓRNICZO-HUTNICZA w KRAKOWIE

Na planecie Ziemia
wiele milionów lat temu
narodziła się inteligencja.

Obecne na świecie spory i konflikty
zagrażają dalszemu jej istnieniu…

Nadzieją dla Ziemi jest
rozwój inteligencji
i wola współpracy.

Studenci swoją pracą,
nauką i kooperacją
mogą przyczynić się
do ratowania i rozwoju
naszej cywilizacji…

# Na początku była światłość

z której zrodziła się inteligencja...

**OBECNIE**

**ludzie rządzą światem, a ich wiedza i inteligencja nieustannie rozwijają się...**

# Inteligencja dalej ewoluuje

**aż do momentu, w którym obdarzone inteligencją maszyny zwolnią ludzi z konieczności myślenia...**

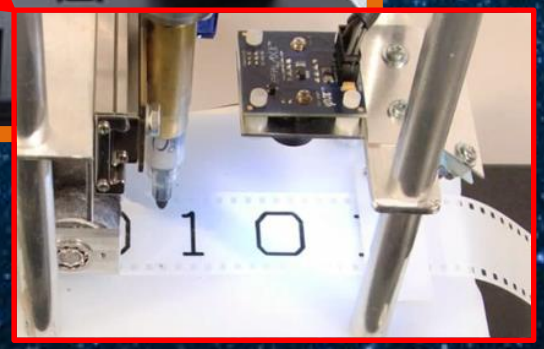
Nie!
Może usuńmy ten scenariusz?

# LUDZKA INTELIGENCJA

może nie tylko
dalej się rozwijać,
lecz również
kooperować
z inteligencją
sztuczną…

# INFORMATYKA, AUTOMATYKA i ROBOTYKA

**zaczęły się intensywnie rozwijać dopiero**
**po opracowaniu pierwszego teoretycznego modelu obliczeniowego, tzw. maszyny Turinga,**
**zdolnej operować na symbolach,**
**oraz pierwszej architektury komputerów**
**przez zespół Johna von Neumanna**
**po 2. wojnie światowej**



# Maszyna Turinga i jej ograniczenia

**Współczesna informatyka opiera się na deterministycznym modelu obliczeniowym, tzw. Maszynie Turinga (MT), który formalnie zdefiniowany jest jako krotka:**

**$MT = < Q, \Sigma, \delta, \Gamma, q_0, B, F >$**

**gdzie:**

**Q – skończony zbiór stanów, $q_0$ – stan początkowy, $q_0 \in Q$**

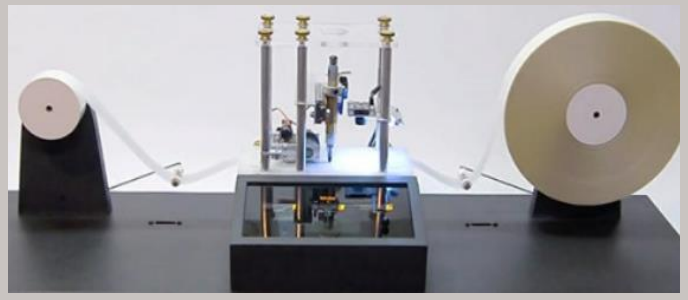
**F – zbiór stanów końcowych**

**Γ – skończony zbiór dopuszczalnych symboli**

**B – symbol pusty, $B \in \Gamma$**

**Σ – zbiór symboli wejściowych, taki że: $B \notin \Sigma \subseteq \Gamma$**

**$\delta: \Gamma \times Q \rightarrow Q \times \Gamma \times \{L,P,-\}$ – funkcja opisująca przejście od stanu $q_1 \in Q$ pod wpływem symbolu wejściowego ze zbioru Γ w kolejny stan ze zbioru $q_2 \in Q$ zwracający symbol ze zbioru Γ oraz przesunięcie głowicy w lewo (L), prawo (P) lub nie dokonując bez przesunięcia (-).**



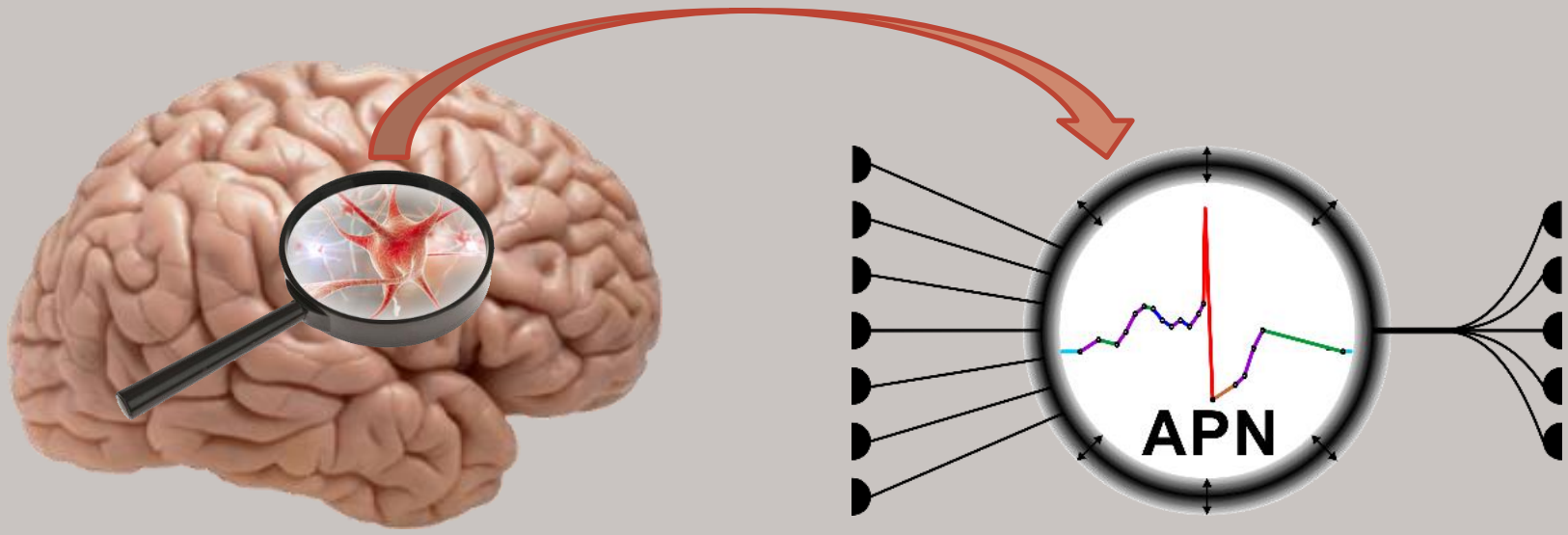
**Model MT skupia się na przetwarzaniu skończonego zbioru symboli, operuje na językach i gramatykach formalnych oraz jest podstawą działania współczesnych komputerów.**

**Maszyna ta ma jednak istotne ograniczenia, np. w stosunku do architektury i sposobu działania ludzkiego umysłu – mózgu oraz biologicznych sieci neuronowych, które z natury są w stanie przetwarzać różne dane, mimo iż nie są maszynami Turinga!**

# A CO NAM OFERUJE BIOLOGIA?

Mózg
?
Zupełnie inny model przetwarzania danych!

# Mózg i Sztuczna Inteligencja

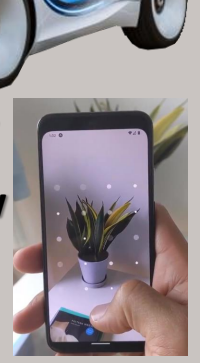


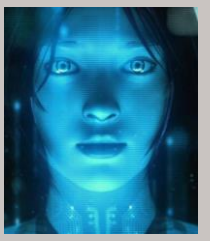

- **Badania nad sposobem działania ludzkiego mózgu zostały zapoczątkowane wiele wieków temu, ale dopiero w dwudziestym wieku rozwój techniki, mikroskopów i komputerów pozwolił na zrozumienie części procesów odgrywających się w ludzkim umyśle i rozwinięcie pierwszych modeli neuronów oraz ich sieci, które w jakimś stopniu modelują procesy kognitywne, pamięć, wiedzę i inteligencję... ale jeszcze długa droga przed nami!**
- **Po ponad pół wieku zmagań nad zdefiniowaniem i urealnieniem sztucznej inteligencji jeden z jej inicjatorów oraz pierwszych jej badaczy Marvin Minsky stwierdził niedawno, iż „prawdziwa sztuczna inteligencja czeka dopiero na swojego odkrywcę!”**
- **Sztuczną inteligencję dzisiaj rozwijamy jako naukę interdyscyplinarną wykorzystującą wiedzę ludzką z zakresu biologii (w tym neurologii, neurofizjologii, biofizjologii), psychologii, matematyki, fizyki, informatymi, logiki, lingwistyki i kognitywistyki.**

# Sztuczna Inteligencja Dzisiaj

## Sztuczna inteligencja dzisiaj obejmuje między innymi:

- Głębokie, konwolucyjne sieci neuronowe i uczenie głębokie
- Rozpoznawanie, identyfikację i klasyfikację obiektów, zdarzeń
- Grupowanie i klasteryzację obiektów o podobnych cechach
- Metody uczenia maszynowego, systemy uczące się i autonomiczne samochody
- Tworzenie AI silników gier komputerowych czy do wspomagania nauki języków
- Wspomaganie przetwarzania, grupowania i selekcji danych
- Predykcję i aproksymację (regresję) wartości liczbowych, np. poziomu ryzyka
- Ewolucyjne (genetyczne) podejścia do przeszukiwania przestrzeni rozwiązań
- Metody oparte na wiedzy, jej formowaniu w systemach komputerowych
- Logikę rozmytą, systemy rozmyte i operacje na danych symbolicznych
- Skojarzeniowe metody reprezentacji wiedzy i wnioskowania
- Tworzenie autonomicznych robotów i botów komputerowych
- Chatboty i automatyczne serwisy komputerowe lub telefoniczne
- Rozpoznawanie mowy wraz z analizą emocji lub charakteru
- Tłuczenia tekstów czy automatyczny opis sceny na obrazach.
- Inteligentne transformacje obrazów oraz ich generowanie





# Co to takiego ta sztuczna inteligencja?

- Czym dla Państwa jest inteligencja? Jakby Państwo zdefiniowali to pojęcie?
- Jaki ma związek z wiedzą, z otoczeniem, z naszymi potrzebami, czy ze środowiskiem?
- Jaką rolę w działaniu naszej (ludzkiej) inteligencji ma nasz mózg (umysł)?
- Czy inteligencją obdarzone są też inne byty, np. zwierzęta?
- Czy inteligencja zwierząt jest inna od naszej czy to tylko tzw. instynkt?
- Czym nasza inteligencja różni się od tej innych bytów i dlaczego?
- Czy to co robi inteligentny człowiek jest zawsze mądre i inteligentne?
- Czy komputery mogą być realnie obdarzone inteligencją, podobną do tej naszej ludzkiej?
- Czym jest lub może być sztuczna inteligencja i czy będzie chciała z nami kooperować?

# Jak definiuje się inteligencję?

**Inteligencja jest zwykle definiowana jako:**

- **umiejętność postrzegania informacji i zachowania ich jako wiedzy, którą należy zastosować w odniesieniu do zachowań adaptacyjnych w środowisku lub pewnym kontekście zdarzeń,**
- **zdolność rozumowania, planowania, rozwiązywania problemów, abstrakcyjnego myślenia, rozumienia złożonych pomysłów, szybkiego uczenia się i efektywnego wykorzystywania zasobów oraz dążenia do celów, które zdefiniowane są przez nasze potrzeby.**
- **Inteligencja obejmuje procesy tj. uczenie się, rozpoznawanie, klasyfikacja, rozumienie, logika, planowanie, kreatywność, rozwiązywanie problemów i samoświadomość.**
- **Inteligencja „działa”, o ile podejmowane działania odpowiadają potrzebom inteligentnego bytu, ponieważ potrzeby oraz zagrożenia (system motywacyjny) napędzają nas do działania, myślenia i poszukiwania rozwiązań problemów w kontekście naszych wartości!**

# Inteligencja umożliwia nam poznawanie

**Poznawanie umożliwiają nam procesy kognitywne ściśle związane z naszą inteligencją:**

- **Proces kognitywny (poznawczy) to mentalne działanie lub proces zdobywania wiedzy i zrozumienia za pomocą myślenia, doświadczenia i zmysłów, które można wykorzystać do eksploracji otoczenia i znalezienia interesujących obiektów zaspokajających potrzeby.**
- **Obejmuje procesy takie jak wiedza, uwaga, pamięć, osąd, ocena, rozumowanie, rozumienie, rozwiązywanie problemów, podejmowanie decyzji, produkcja języka, jego rozumienie itp.**
- **Proces poznawczy może być świadomy lub nieświadomy, konkretny lub abstrakcyjny, intuicyjny lub konceptualny.**
- **Procesy poznawcze wykorzystują wiedzę wszelkiego rodzaju i generują nową wiedzę, a także wynik przetwarzania informacji.**

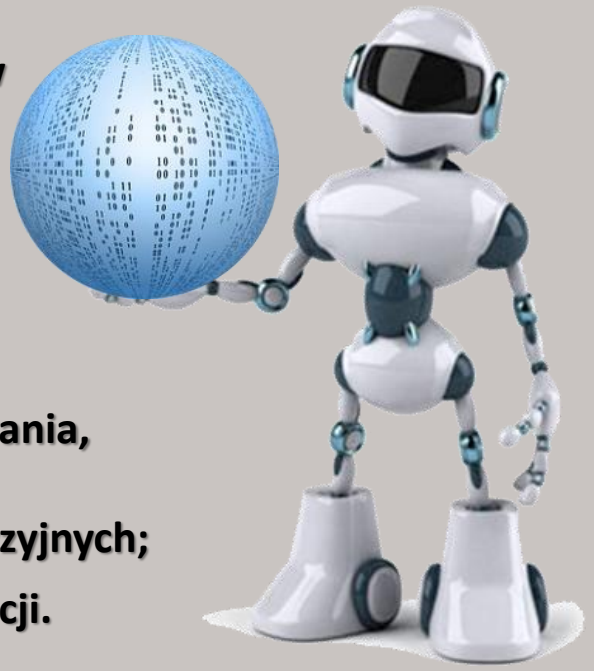
# Czym jest sztuczna inteligencja?

**Sztuczna inteligencja (do jakiej dążymy w nauce) powinna być zdolna do:**

- **Odtwarzania i naśladowania ludzkiej inteligencji oraz dostarczania nowych wartości.**
- **Rozpoznawania ludzkich potrzeb i wartości oraz reagowania na nie w pozytywny sposób.**
- **Rozpoznawania, grupowania i klasyfikacji podobnych przedmiotów (obiektów), faktów, działań (akcji i reakcji), reguł, procedur i ich uogólniania oraz dostosowania do nowych zdań.**
- **Kojarzenia obiektów i zdarzeń, dostosowania i uczenia się oraz zapamiętywania obiektów i zdarzeń w różnych kontekstach, przypominania sobie o istotnych w danym kontekście aktualnych obiektów i zdarzeń, wykorzystanie zapamiętanych procedur, metod i algorytmów oraz ich modernizacji, generalizacji i wykorzystania w korzystny sposób dla człowieka i świata.**
- **Zrozumienia ludzkiej psychiki, osobowości, potrzeb, dążeń i aspiracji celem umożliwienia jej współpracy z ludźmi oraz działania z wyczuciem i w kontekście ludzkich potrzeb i aspiracji.**
- **Zastępowania ludzi w częstych lub uciążliwych zadaniach.**

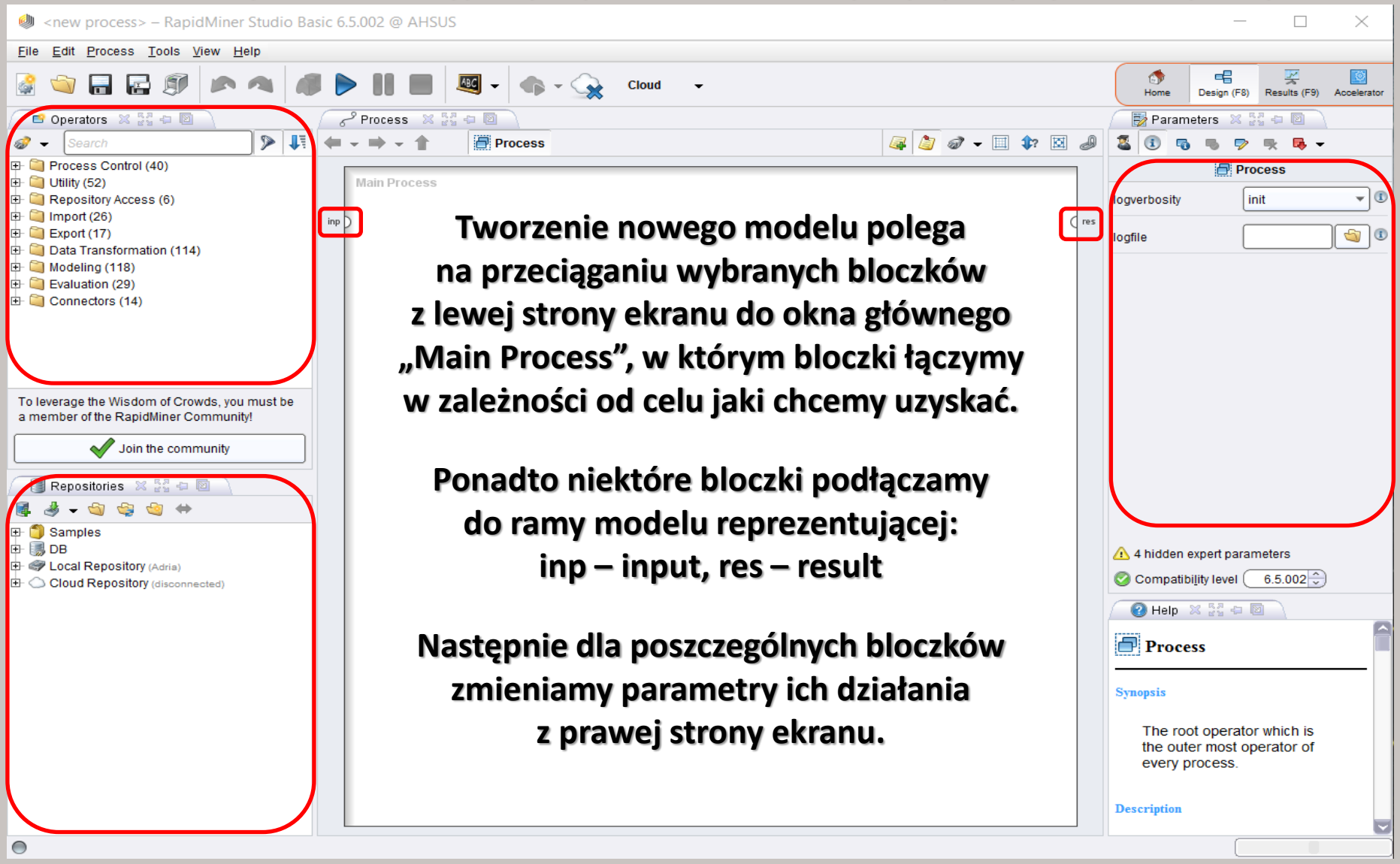
# Inteligencja obliczeniowa

- to inspirowane przyrodą podejścia i metodologie rozwiązywania złożonych problemów w świecie rzeczywistym, dla których modelowanie matematyczne jest bezużyteczne lub niewystarczająco wydajne (np. ze względu na zbyt dużą złożoność obliczeniową);
- zwykle odnosi się do zdolności komputera do uczenia się określonych zadań na podstawie danych lub obserwacji eksperymentalnych, a następnie wykorzystania zbudowanego modelu obliczeniowego na podstawie nauki do rozwiązywania podobnych zadań;
- koncentruje się na rozwiązywaniu zadań inżynierskich przy użyciu adaptacyjnych modeli matematycznych wzorowanych na ludzkim sposobie myślenia lub innych procesach biologicznych, np. genetycznych, logice rozmytej wykorzystujące symbole w rozumowaniu;
- nie odtwarza w wystarczającym stopniu ludzkiej inteligencji, ale modeluje inteligentne sposoby rozwiązywania problemów za pomocą mechanizmów, struktur danych i systemów inspirowanych naturą, w szczególności ludzkim mózgiem;
- obejmuje sztuczne sieci neuronowe, logikę rozmytą, obliczenia ewolucyjne, algorytmy genetyczne i różne metody probabilistyczne;
- służy do skutecznego rozpoznawania, klasyfikowania, grupowania, przewidywania lub przybliżania w celu podejmowania decyzji bez pomocy człowieka lub pomocy ludziom w procesach decyzyjnych;
- może być postrzegana jako ważny element sztucznej inteligencji.



# RapidMiner Studio

**Istnieją też różne gotowe narzędzia, które bez wiedzy informatycznej pozwalają stosować różne metody uczenia dla typowych problemów klasteryzacji, predykcji lub klasyfikacji.**




**Tworzenie nowego modelu polega na przeciąganiu wybranych bloczków z lewej strony ekranu do okna głównego „Main Process", w którym bloczki łączymy w zależności od celu jaki chcemy uzyskać.**

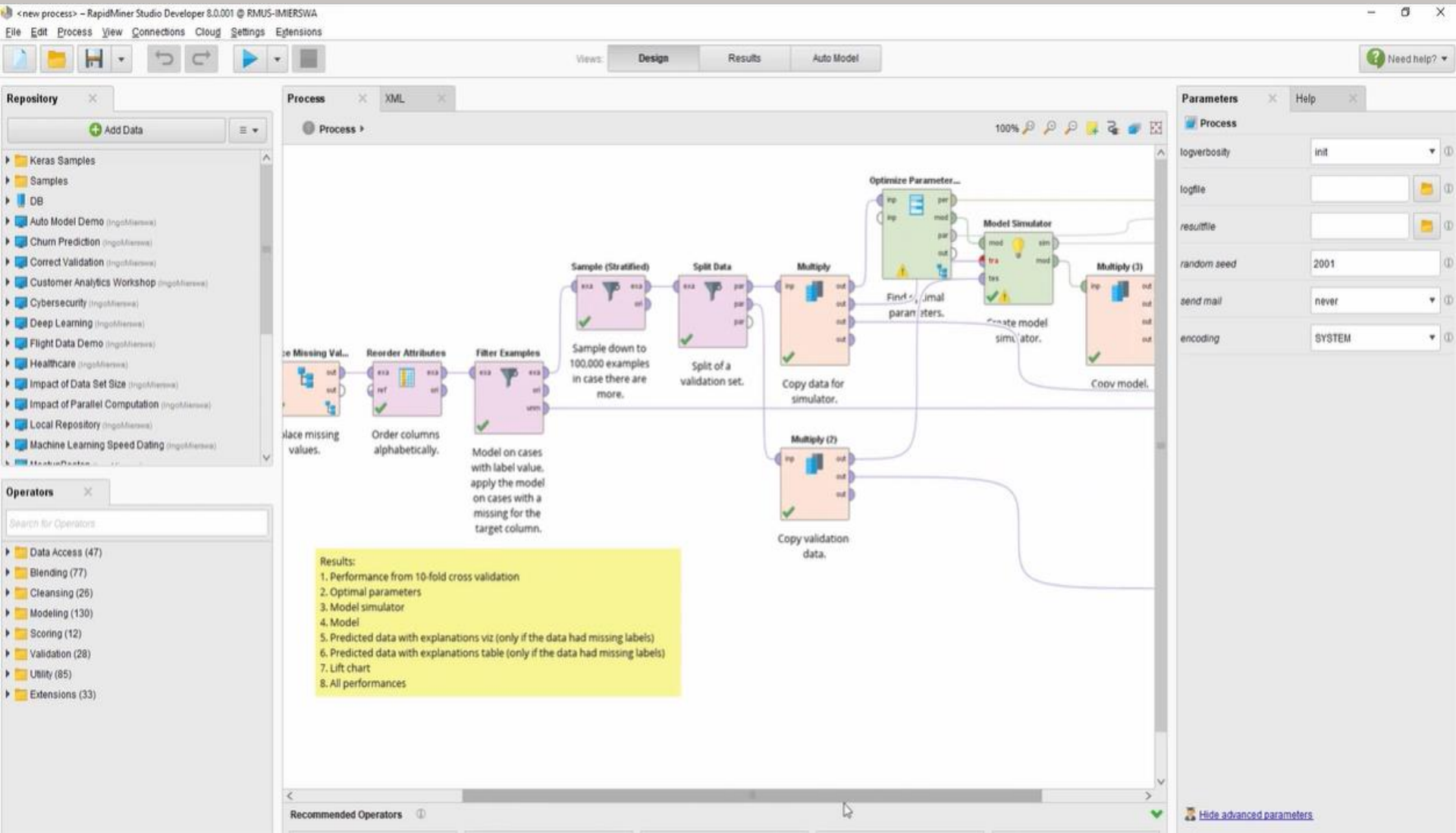
**Ponadto niektóre bloczki podłączamy do ramy modelu reprezentującej: inp – input, res – result**

**Następnie dla poszczególnych bloczków zmieniamy parametry ich działania z prawej strony ekranu.**

# RapidMiner Studio

**Istnieją też różne gotowe narzędzia, które bez wiedzy informatycznej pozwalają stosować różne metody uczenia dla typowych problemów klasteryzacji, predykcji lub klasyfikacji.**

**Rozwiązania budujemy z bloczków lub korzystamy z automatycznej budowy modelu:**



# RapidMiner Studio

**Możemy też skorzystać z automatycznej budowy modelu, wskazując dane i cele:**

<new process> – RapidMiner Studio Developer 8.0.001 @ RMUS-IMIERSWA

File Edit Process View Connections Cloud Settings Extensions

Views: Design | Results | Auto Model

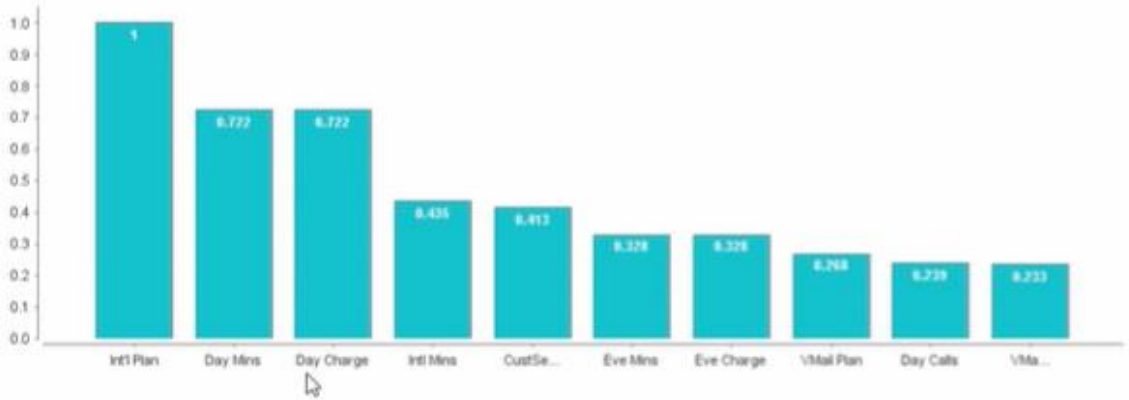
Auto Model

Select Data → Select Task → Prepare Target → Select Inputs → Model Types → Results

## Results

- General
  - Data
  - Weights
  - Correlations
- Comparison
  - Overview
  - ROC Comparison
- Naive Bayes
  - Model
  - Simulator
  - Performance
  - Lift Chart
  - Predictions
- Generalized Linear Model
- Logistic Regression
- Deep Learning
- Decision Tree
- Random Forest
- Gradient Boosted Trees

## Weights

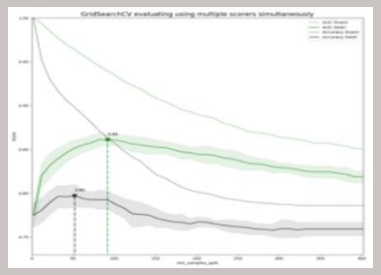


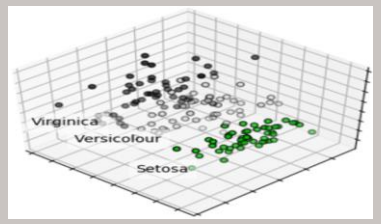

| Attribute | Weight |
|---|---|
| Int'l Plan | 1 |
| Day Mins | 0.722 |
| Day Charge | 0.722 |
| Intl Mins | 0.435 |
| CustServ Calls | 0.413 |
| Eve Mins | 0.328 |
| Eve Charge | 0.328 |
| VMail Plan | 0.268 |
| Day Calls | 0.239 |
| VMail Message | 0.233 |
| Intl Calls | 0.170 |
| Night Charge | 0.089 |

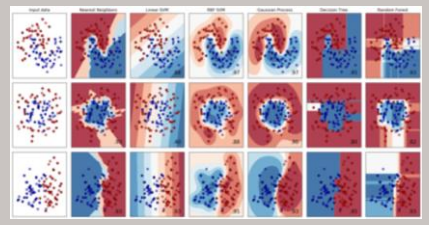
# Inteligencja obliczeniowa w praktyce

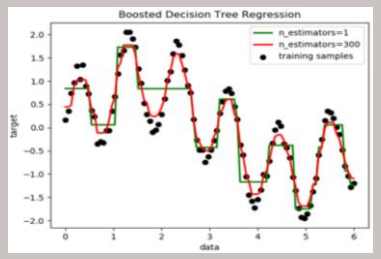
**Do realizacji postulatów i celów inteligencji obliczeniowej stworzono wiele narzędzi i bibliotek (frameworków), które mają zaimplementowane różne struktury, metody, algorytmy do predykcji, aproksymacji, rozpoznawania, detekcji, klasyfikacji oraz grupowania (klasteryzacji):**

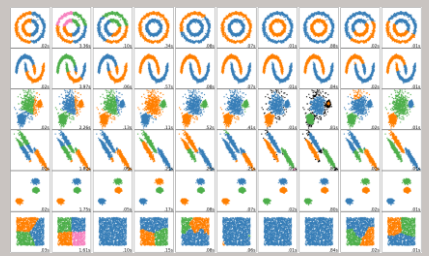
- **Jupyter Notebook**
- **Google Colab**
- **Tensorflow 2.0**
- **Keras**
- **PyTorch**
- **Caffe2**
- **Theano**
- **Lasagne**
- **DL4J** (deep learning for Java)
- **CNTK** (MS Cognitive Toolkit)
- **PaddlePaddle**
- **numpy**
- **scikit-learn**
- **pandas**
- **mxnet**
- **matplotlib**
- itd.

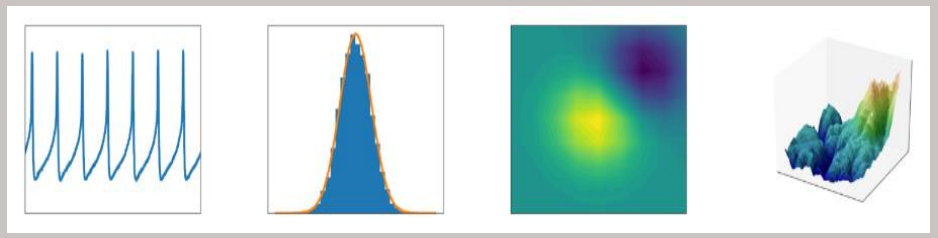


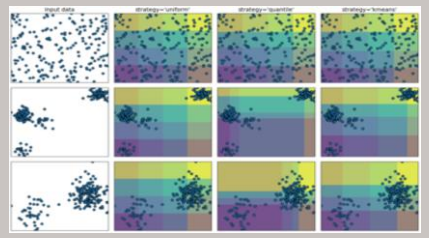




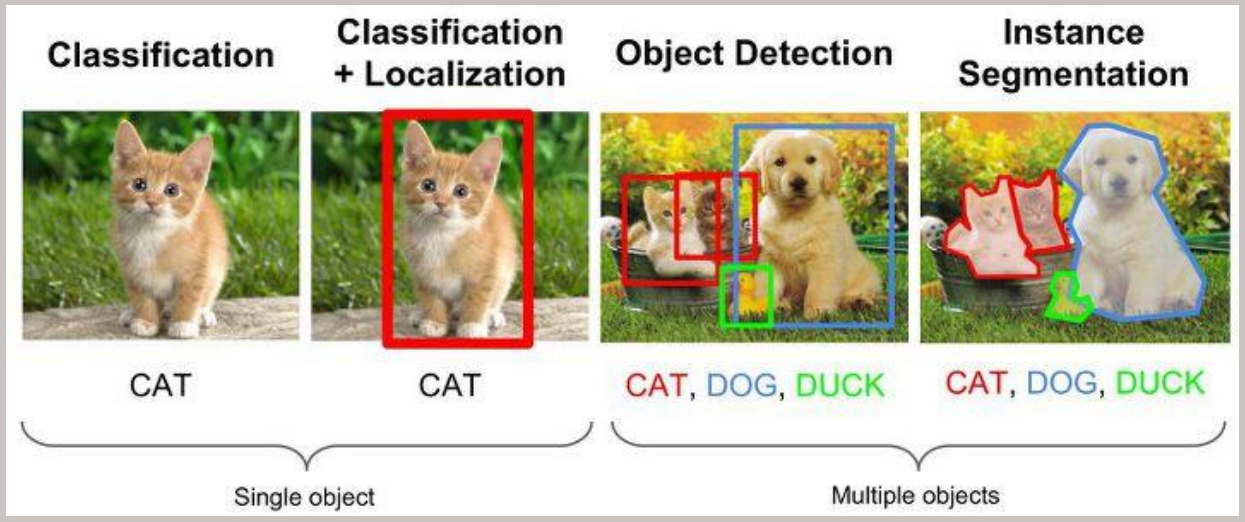



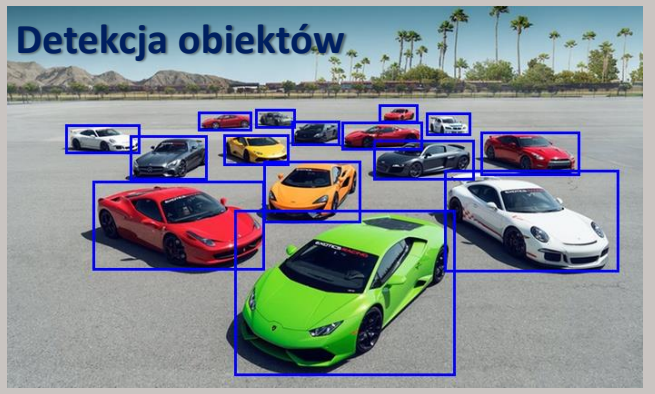


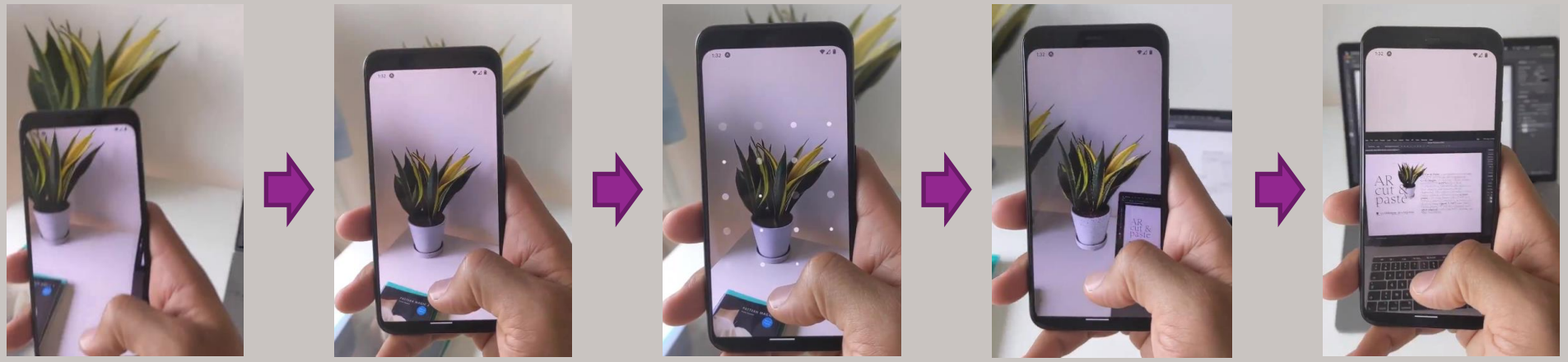


# Detekcja i Segmentacja Obiektów

**Dzisiaj popularne są różne metody wykrywania (detekcji), klasyfikacji, lokalizacji i segmentacji obiektów na scenie wykorzystywane w wielu sektorach gospodarki oraz różnych branżach przemysłu i bankowości:**

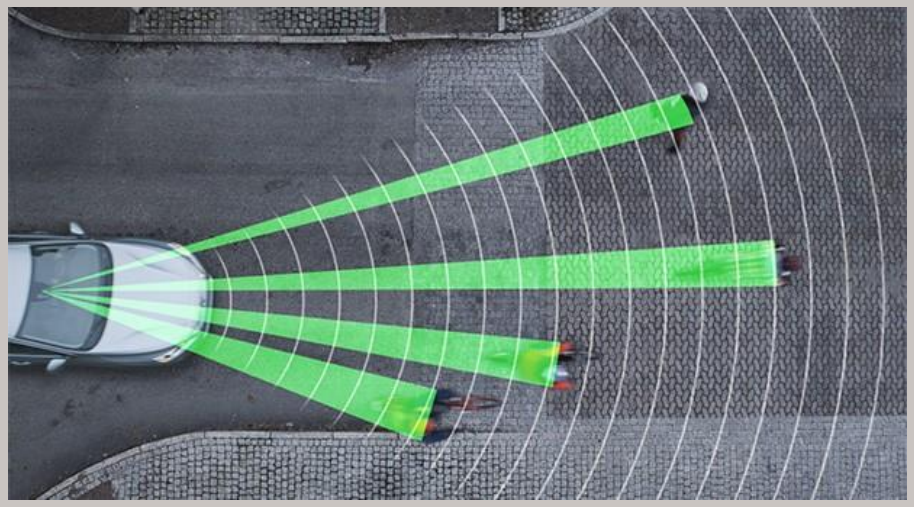


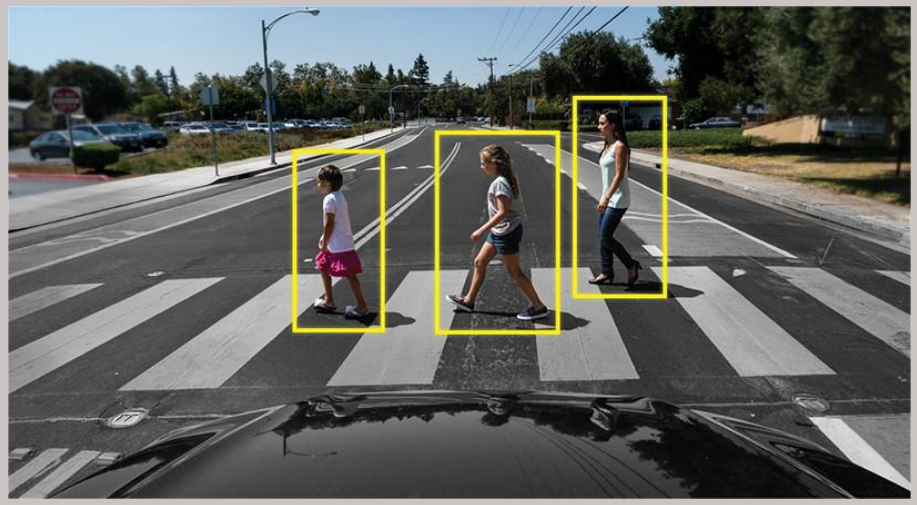



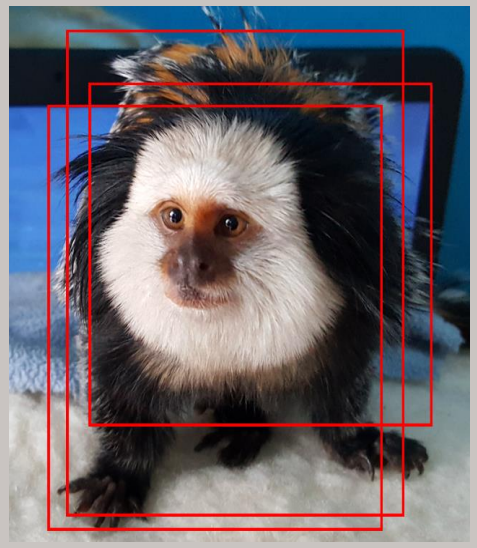



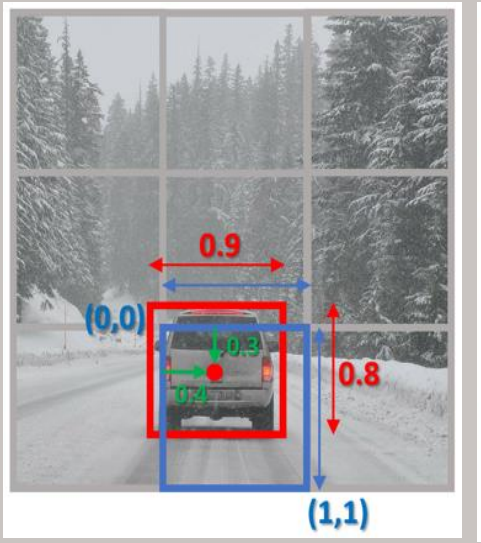
# Detekcja obiektów w obrazach

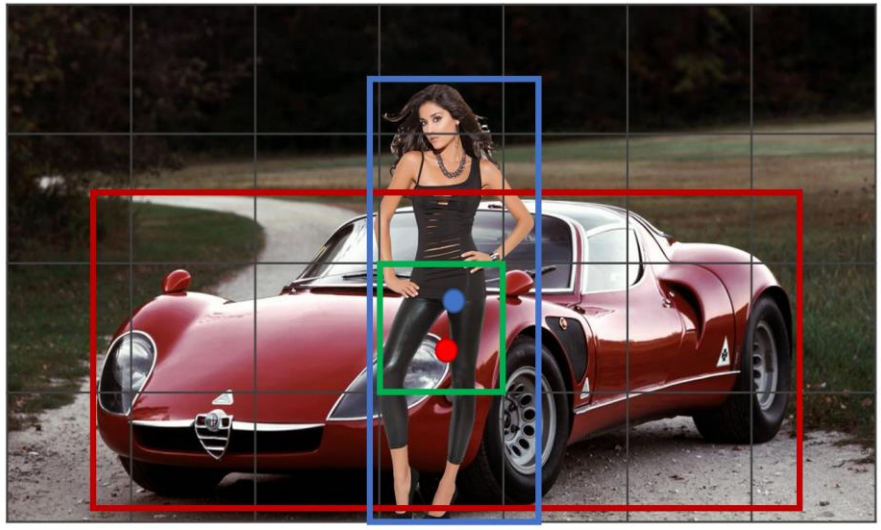
**Detekcja, lokalizacja i klasyfikacja obiektów w obrazach pozwala na wykrywanie interakcji obiektów, ich kojarzenie oraz reagowanie systemów informatycznych na potencjalne korzystne lub niekorzystne zdarzenia w sposób automatyczny.**





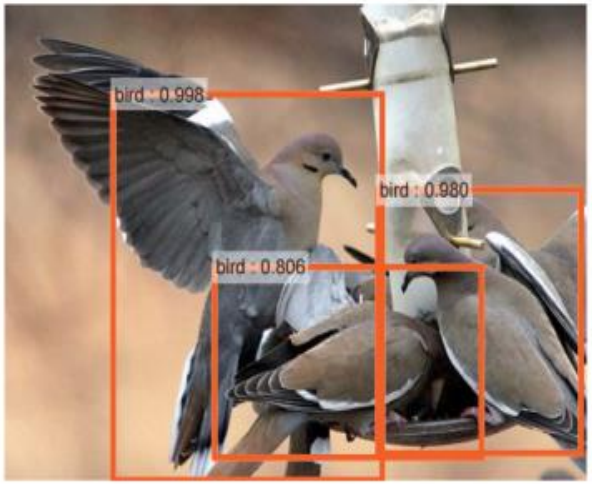


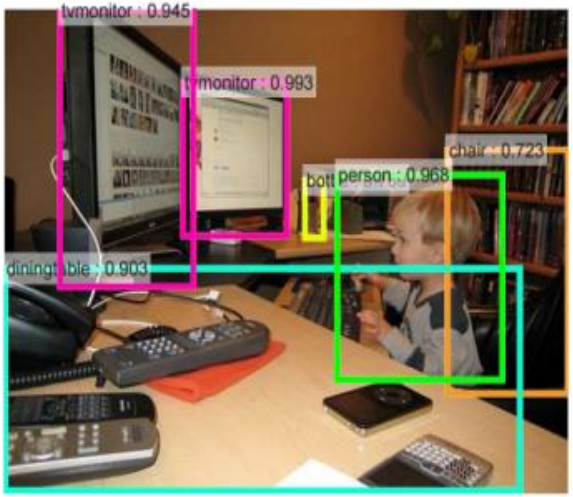


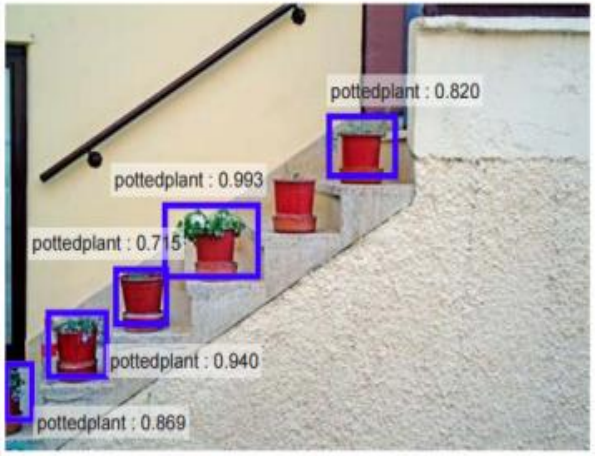



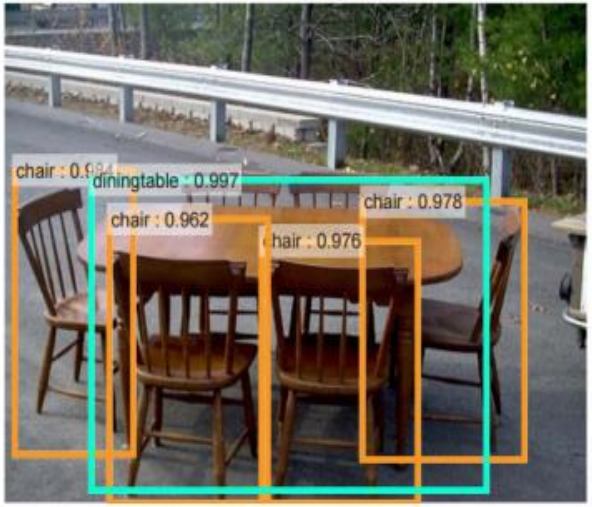
# Detekcja obiektów w obrazach

**Detekcja, lokalizacja i klasyfikacja obiektów w obrazach pozwala na wykrywanie interakcji obiektów, ich kojarzenie oraz reagowanie systemów informatycznych na potencjalne korzystne lub niekorzystne zdarzenia w sposób automatyczny.**



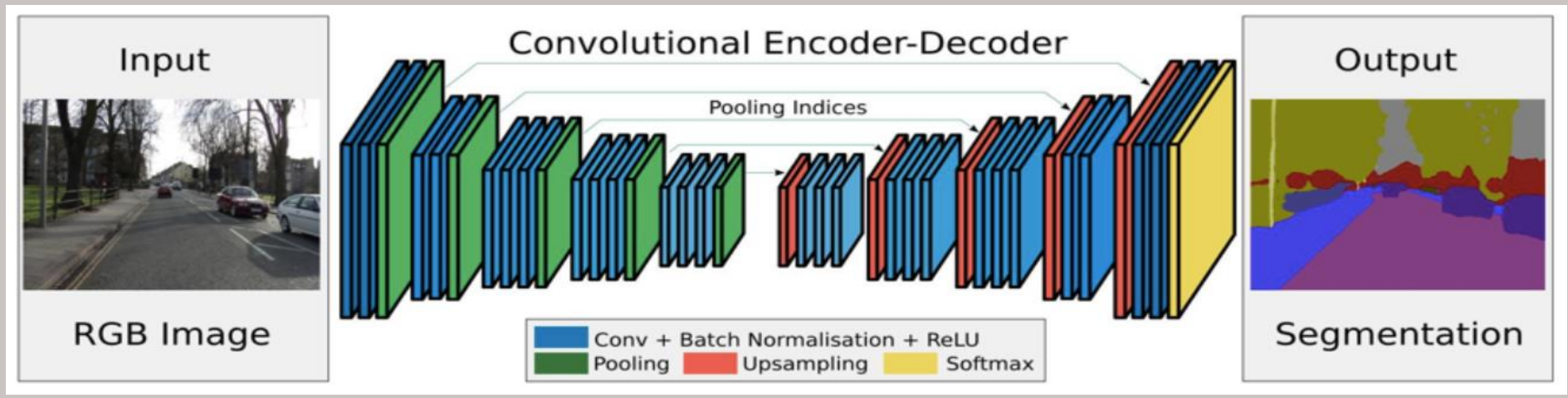



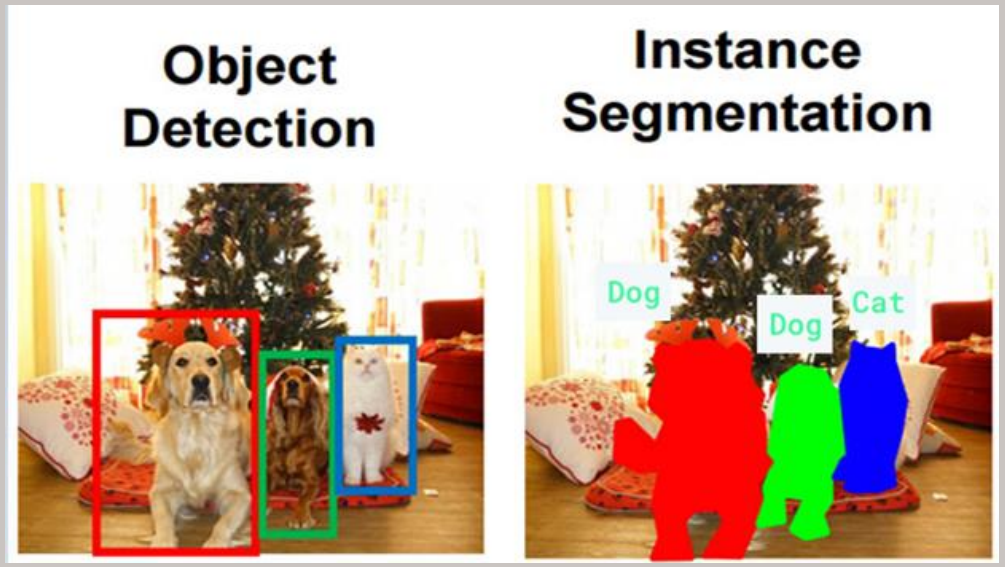



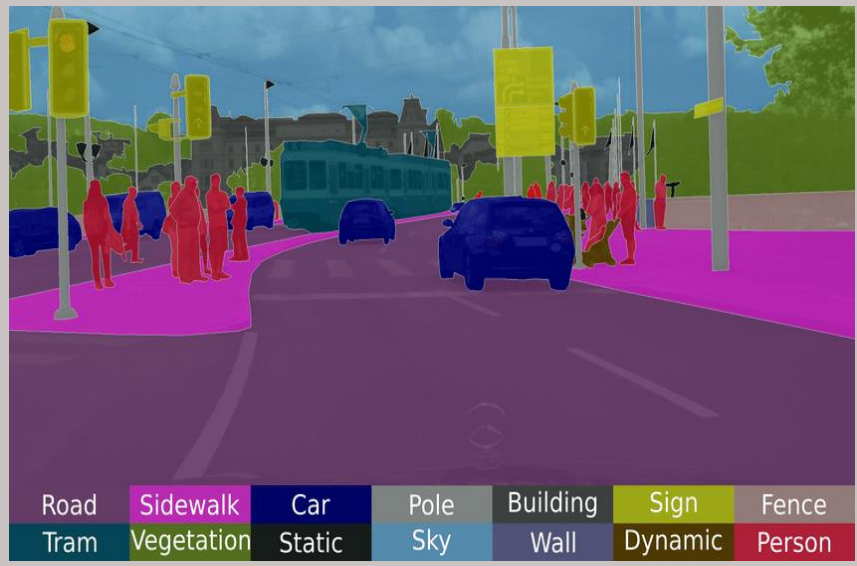



# Segmentacja obrazów na scenie

- **W systemach mobilnych i systemach bezpieczeństwa (pieszych) wykorzystywane są coraz bardziej zaawansowane modele i metody oparte o głębokie sieci neuronowe, które służą do detekcji istotnych obiektów na obrazach pozyskiwanych z kamer:**

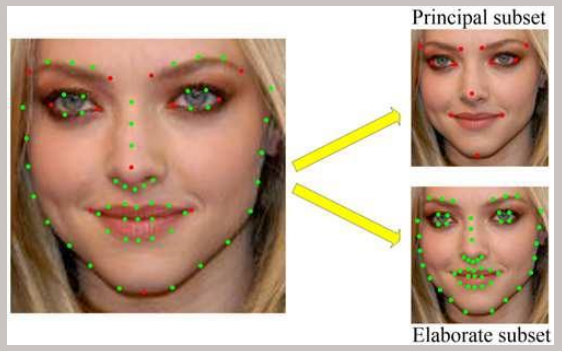


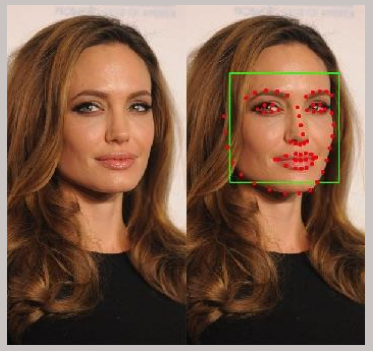



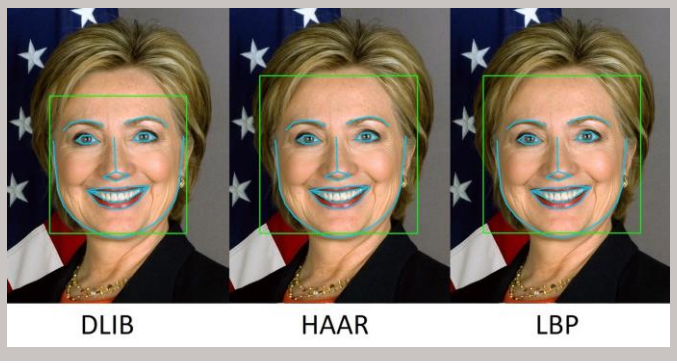



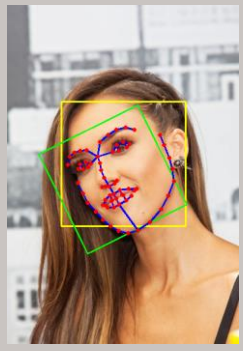
# Rozpoznawanie Twarzy i Emocji

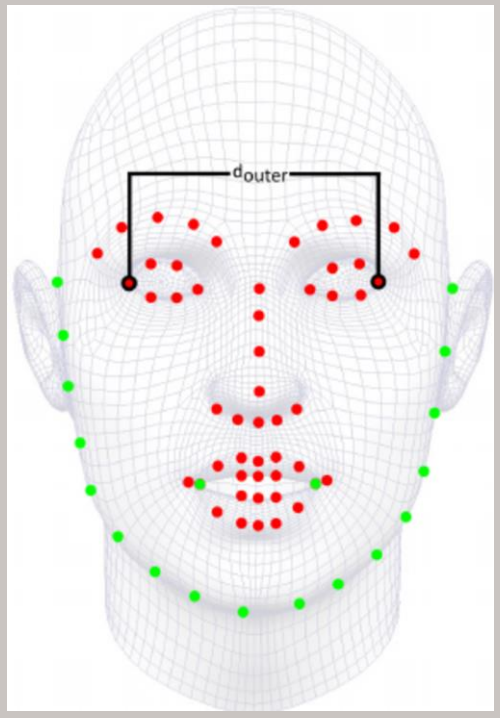
**Równie popularne w systemach bezpieczeństwa, marketingowych (badanie zadowolenia klientów) są systemy rozpoznające twarze oraz klasyfikujące emocje pojawiające się np. na skutek prezentacji nowego produktu, usługi oraz weryfikacji zadowolenia klienta z obsługi.**

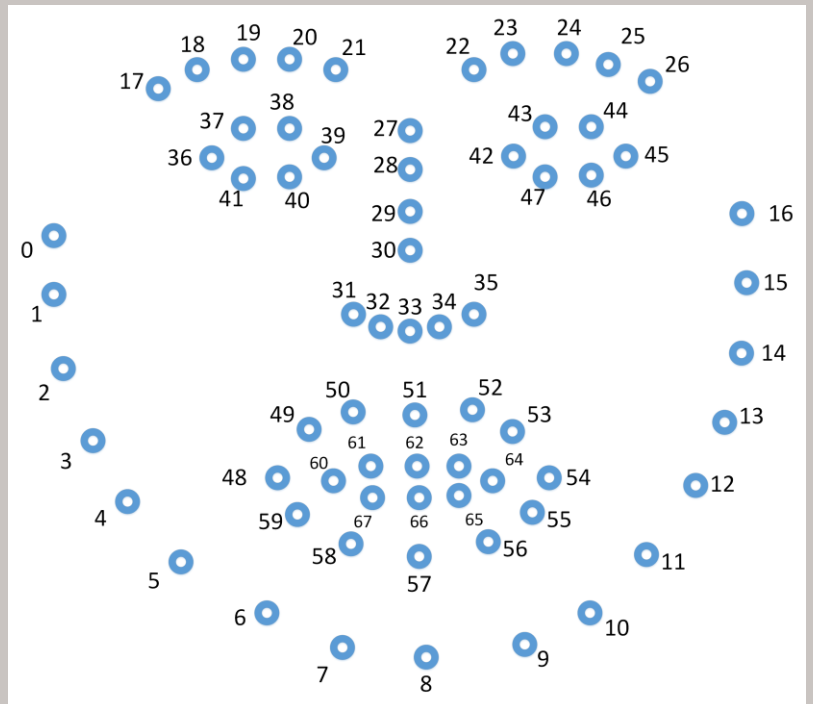






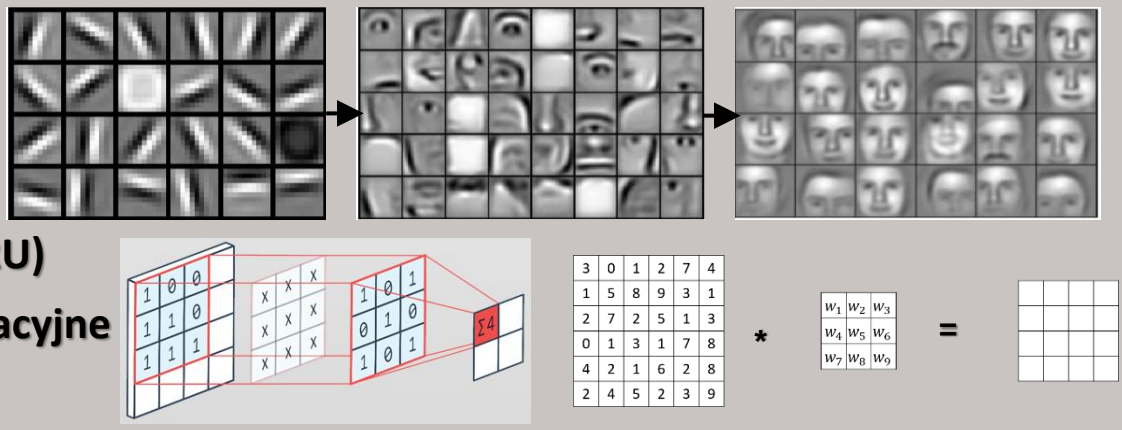



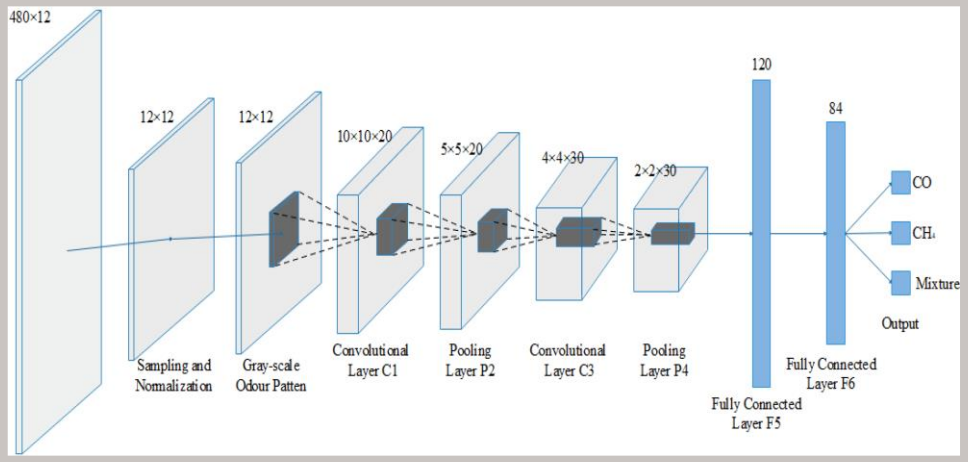
# Klasyfikacja i Multiklasyfikacja

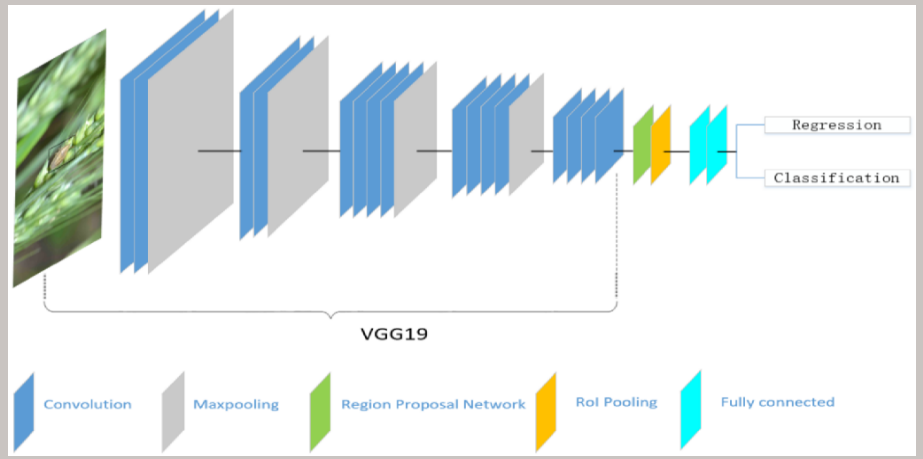
- **Klasyfikacja** to proces przyporządkowania obiektu do pewnej klasy podobnych obiektów zdefiniowanych przy pomocy reprezentantów (obiektów uczących), przy czym obiekt może zostać przyporządkowany tylko do jednej spośród zdefiniowanych klas.
- **Multiklasyfikacja** zakłada możliwość przyporządkowania obiektu do jednej lub więcej zdefiniowanych klas, np. lodówka jest meblem i wyposażeniem AGD równocześnie.

Klasyfikacji i multiklasyfikację możemy przeprowadzić przy pomocy szeregu metod:

- Proste metody, tj. KNN
- Sieci neuronowe typu MLP
- Głębokie sieci konwolucyjne CNN
- Sieci rekurencyjne (RNN, LSTM, GRU)
- Neuronowe sieci impulsowe i pulsacyjne
- Algorytmy logiki rozmytej



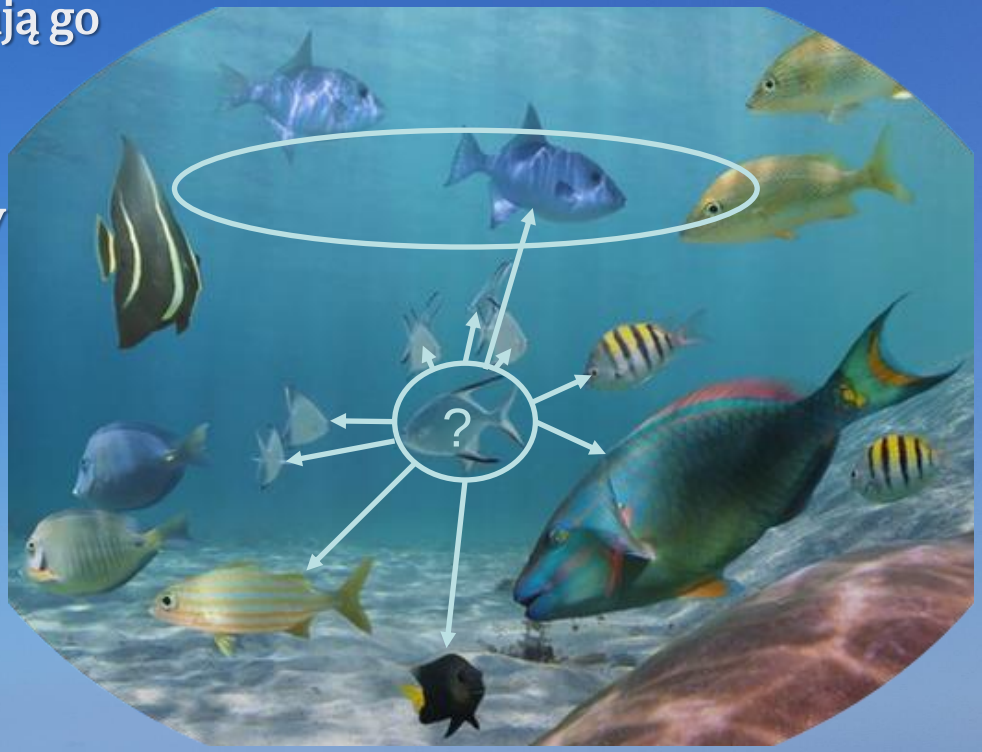



# Klasyfikator: Metoda K Najbliższych Sąsiadów (KNN)

Klasyfikatory K-Najbliższych Sąsiadów (KNN - K-Nearest Neighbors) należą do grupy algorytmów leniwych (*lazy*), tj. takich, które nie tworzą wewnętrznej reprezentacji wiedzy, lecz poszukają rozwiązania po prezentacji każdego wzorca testowego przeszukując dane uczące. Metoda KNN wymaga przechowywania wszystkich wzorców uczących, dla których określana jest odległość do wzorca testowego.

Istnieją również inne algorytmy uczenia się, zwane gorliwymi (*eager*), które najpierw rozwijają model, a następnie wykorzystują go do określonego zadania obliczeniowego, np. klasyfikacji.

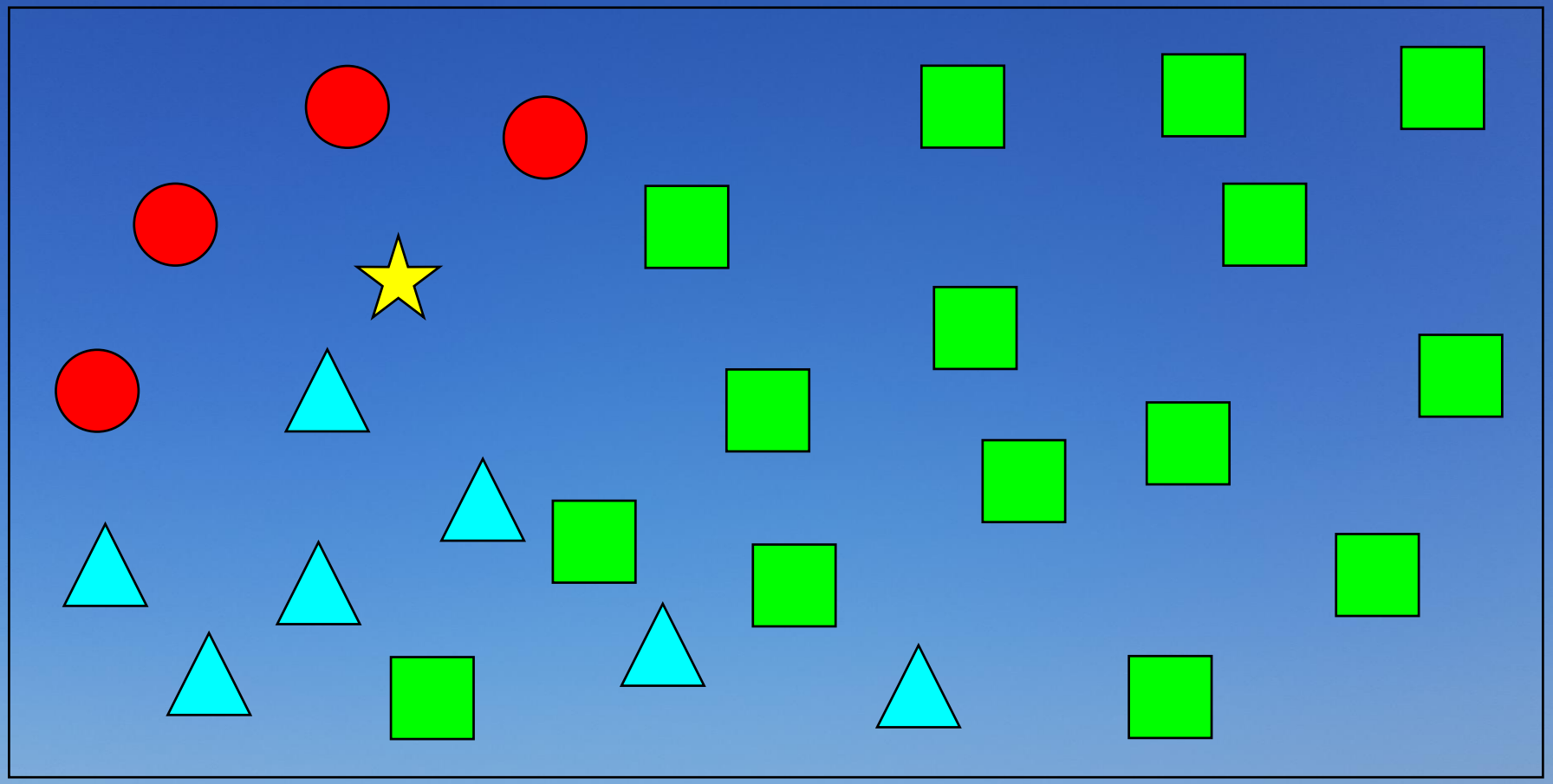



Ta grupa metod obejmuje wszystkie typy sieci neuronowych, systemy rozmyte, drzewa decyzyjne, SVM i wiele innych.

Po uczeniu (adaptacji) takich modeli dane uczące mogą zostać usunięte, ponieważ proces klasyfikacji wykorzystuje tylko utworzony model.

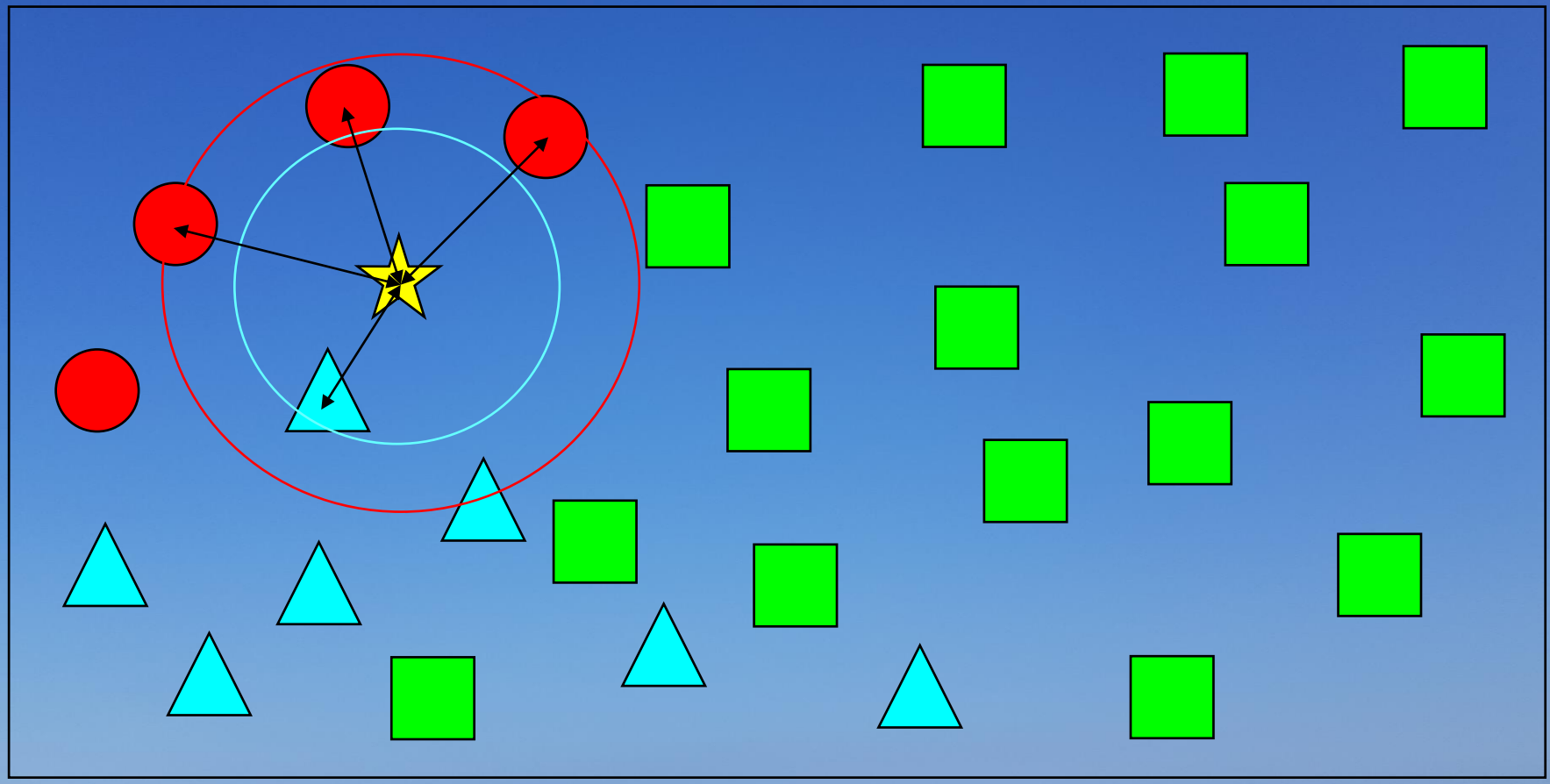
# Klasyfikator: Metoda K Najbliższych Sąsiadów (KNN)

Do której klasy należy gwiazdka: kółek, trójkątów lub kwadratów?



# Klasyfikator: Metoda K Najbliższych Sąsiadów (KNN)

**Metoda K Najbliższych Sąsiadów** ma za zadnie znaleźć k sąsiadów, do których klasyfikowane obiekty są najbliższe dla wybranej metryki (np. Euklidesa), a następnie określa wynik klasyfikacji na podstawie większości głosów tych najbliższych k sąsiadów, biorąc pod uwagę, która klasa jest reprezentowana przez największą liczbę sąsiadów w grupie k najbliższych.



# Klasyfikator: Metoda K Najbliższych Sąsiadów (KNN)

Zestaw wzorców uczących (treningowych) składa się z zestawu par $\langle x^i, y^i \rangle$, gdzie:

$x^i = [x_1^i, \ldots x_n^i]$ jest wektorem wejściowym definiującym obiekty (zwykle w postaci wektora lub macierzy), $y^i$ to przewidywana / pożądana wartość wyjściowa, np. indeks lub nazwa klasy, do której należy $x^i$.

Na rysunku mamy obiekty należące do trzech klas: kółek, trójkątów i kwadratów.

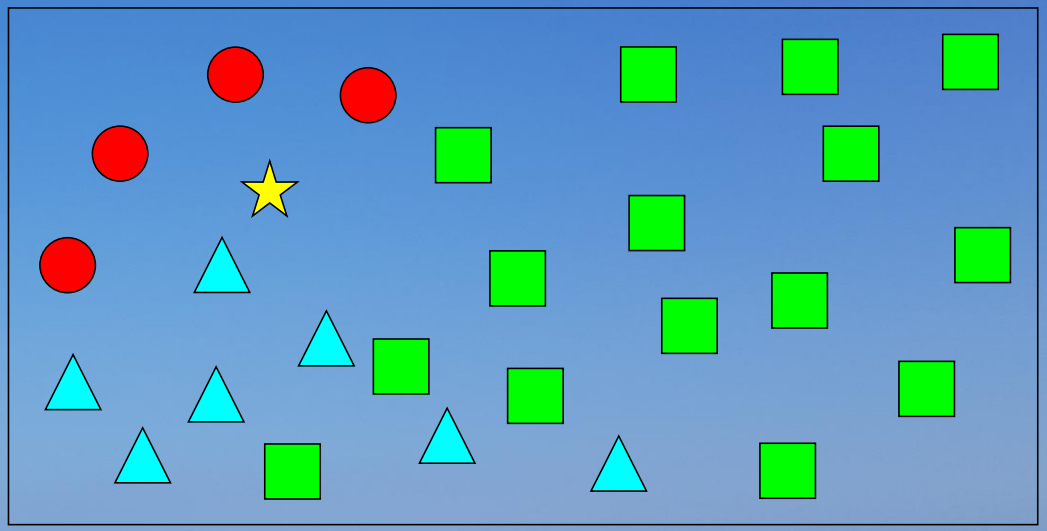
Gwiazdka to obiekt, który chcemy sklasyfikować, tj. przypisać go do jednej z istniejących klas.

Do której klasy należy gwiazdka: kółek, trójkątów czy kwadratów? Co nam podpowiada intuicja?

Na przykład, można zbadać odległość gwiazdki od innych obiektów, dla których klasa jest znana, używając jednej ze znanych metryk, np. odległości Euklidesa:

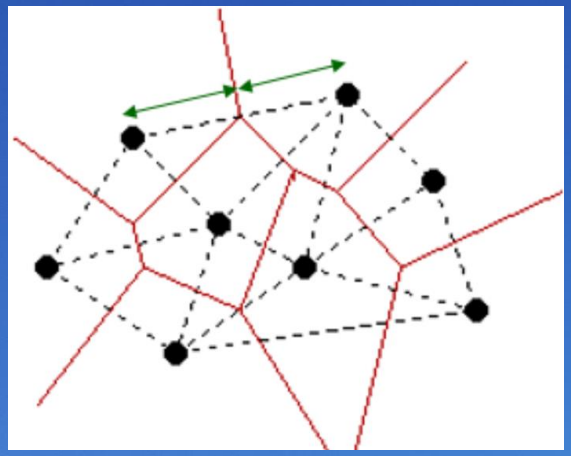
$$\|x - x^k\|_2 = \sqrt{\sum_{j=0}^{J} (x_j - x_j^k)^2}$$

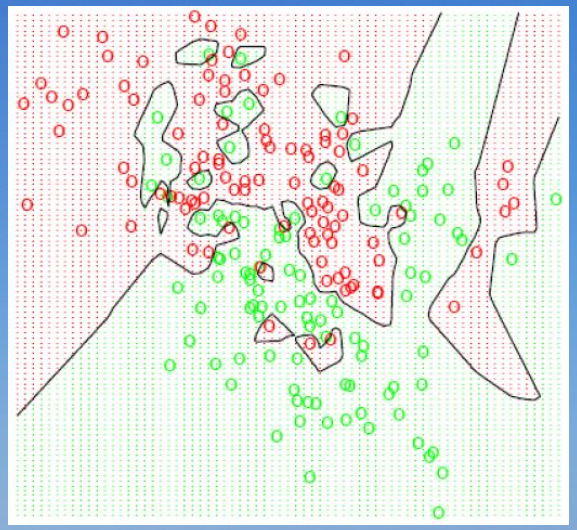
i na tej podstawie określ klasę gwiazdki.

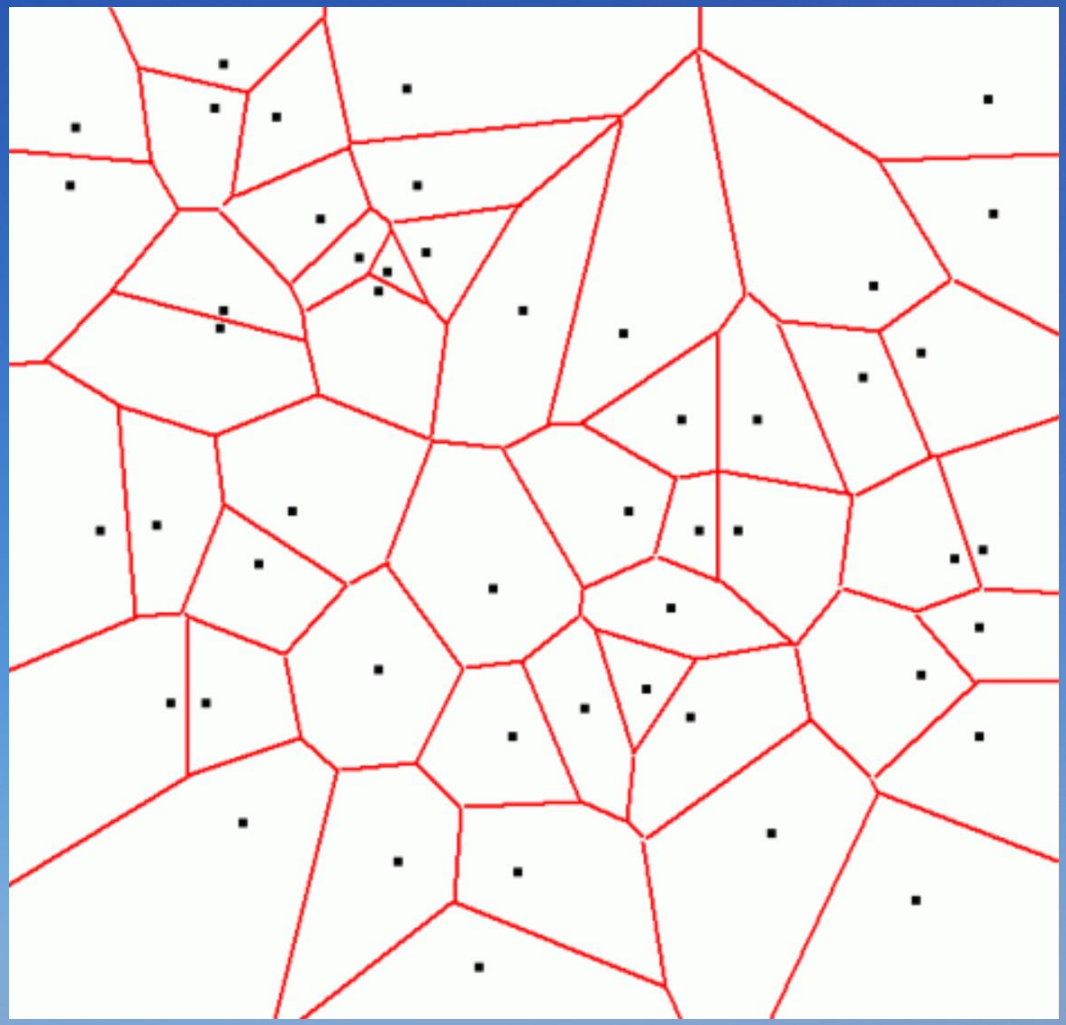


# Regiony Przyciągania Klasyfikatorów KNN

**Diagramy Voronoi** są stosowane do ilustracji obszarów przyciągania do najbliższych obiektów w przestrzeni danych wejściowych:







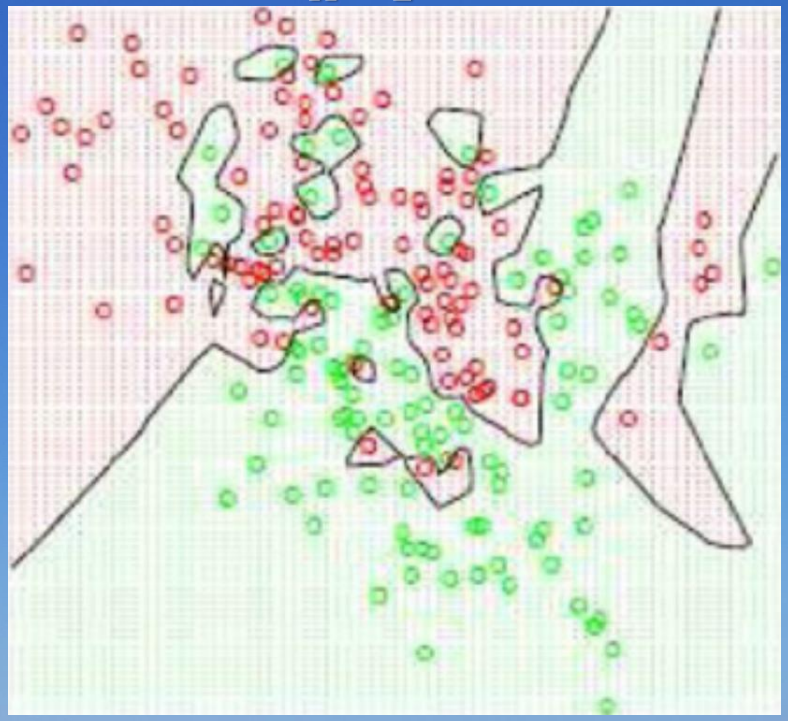
# Regiony Przyciągania Klasyfikatorów KNN

**Algorytm K Najbliższych Sąsiadów** daje różne wyniki w zależności od wyboru parametru **K**.

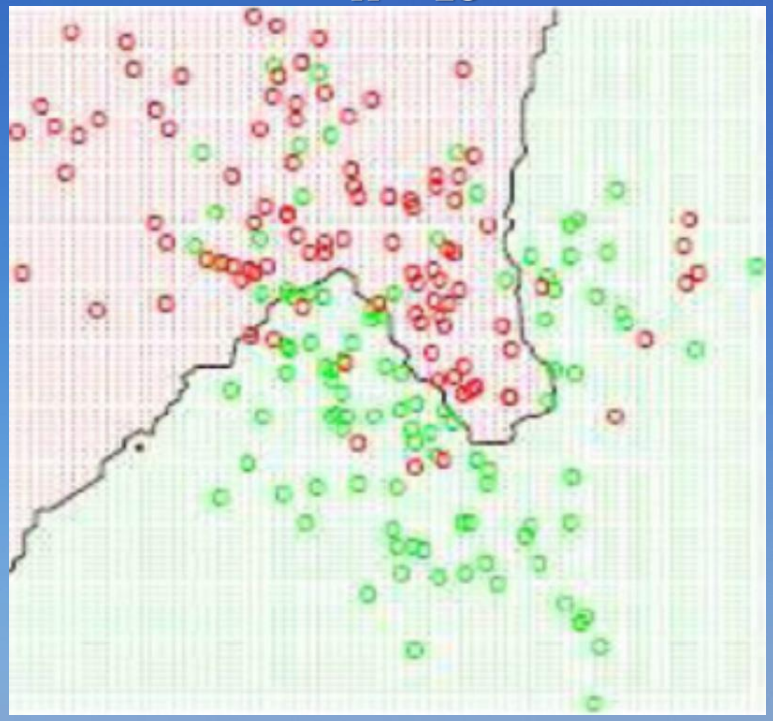
**Obszary przyciągania** determinują wyniki klasyfikacji.

Większe wartości K pozwalają na otrzymanie **bardziej wygładzonych obszarów przyciągania, usuwają szum i artefakty (mogą zapewnić lepsze uogólnienie (generalizację))**, ale również prowadzą do błędnej klasyfikacji odosobnionych poprawnych wzorców.

K = 1



K = 15

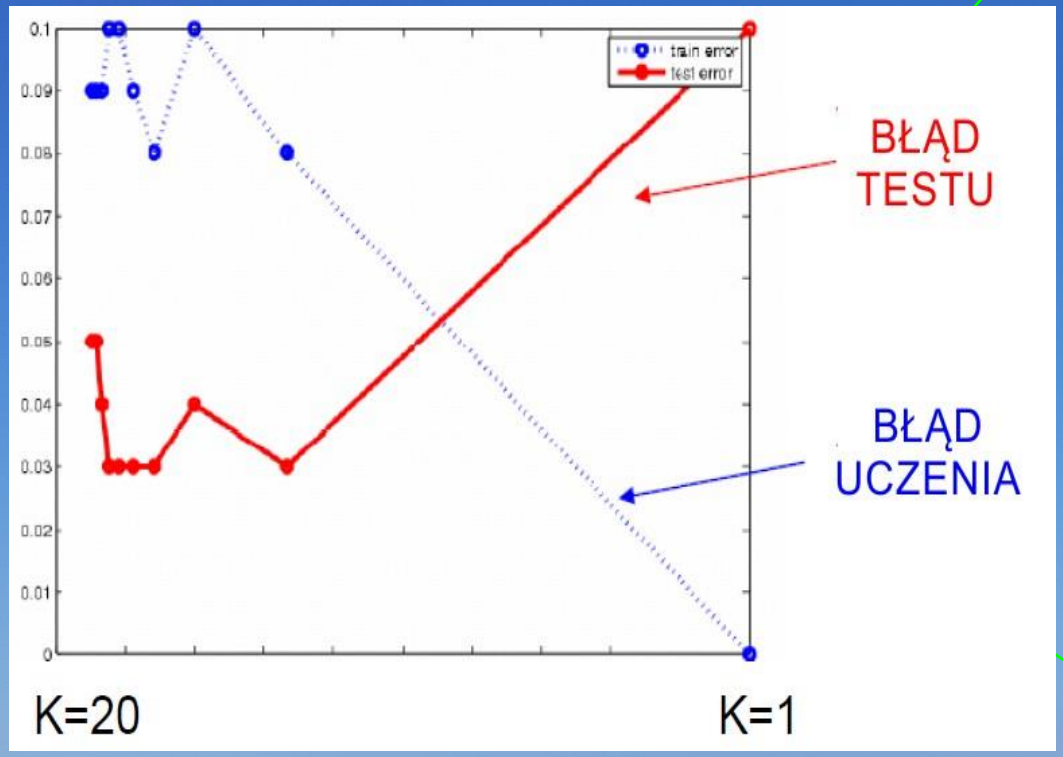


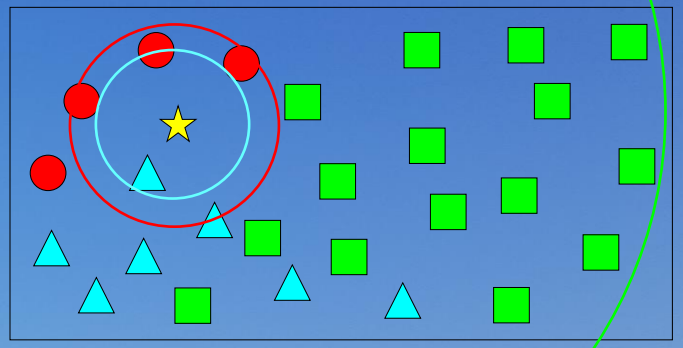
(źródło: Hastie, Tibshirani and Friedman (Elements of Statistical Learning)

# Jak wybrać optymalne K?

Jeśli wybierzemy $K = N$, gdzie N jest ilością wszystkich wzorców (obiektów) uczących, wtedy wynik klasyfikacji będzie tylko zależny od najbardziej licznie reprezentowanej klasy:

Błąd uczenia jest zawsze równy 0 dla $K = 1$, nie oznacza to jednak zawsze najlepszego uogólnienia (generalizacji).

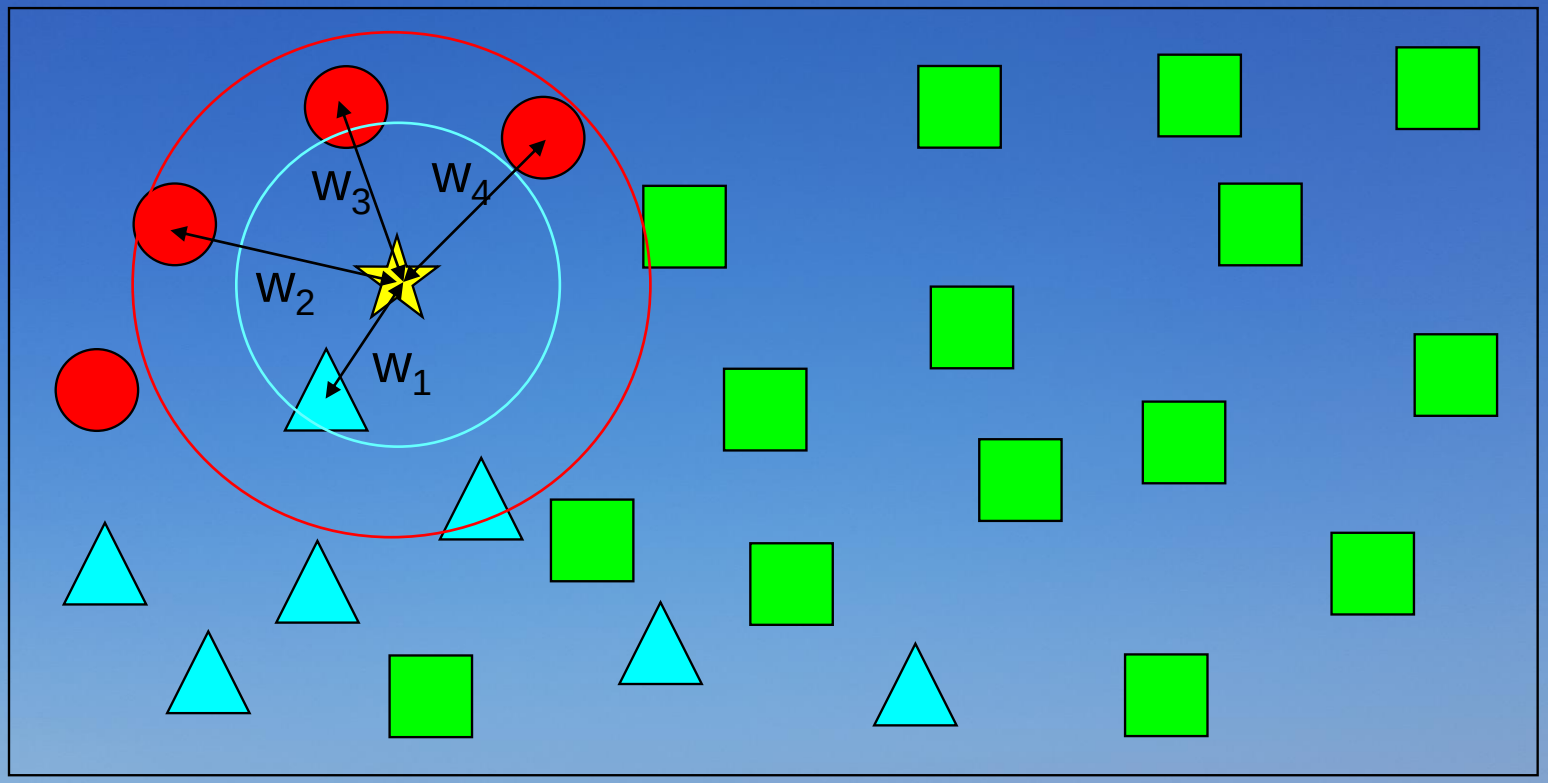


# Wariacje metody KNN

**Metoda K Najbliższych Sąsiadów z ważeniem głosów w zależności od odległości sąsiadów** bierze pod uwagę głosy dotyczące klasyfikowanego wzorca z wagą zależną od odległości każdego z najbliższych sąsiadów do tego wzorca.

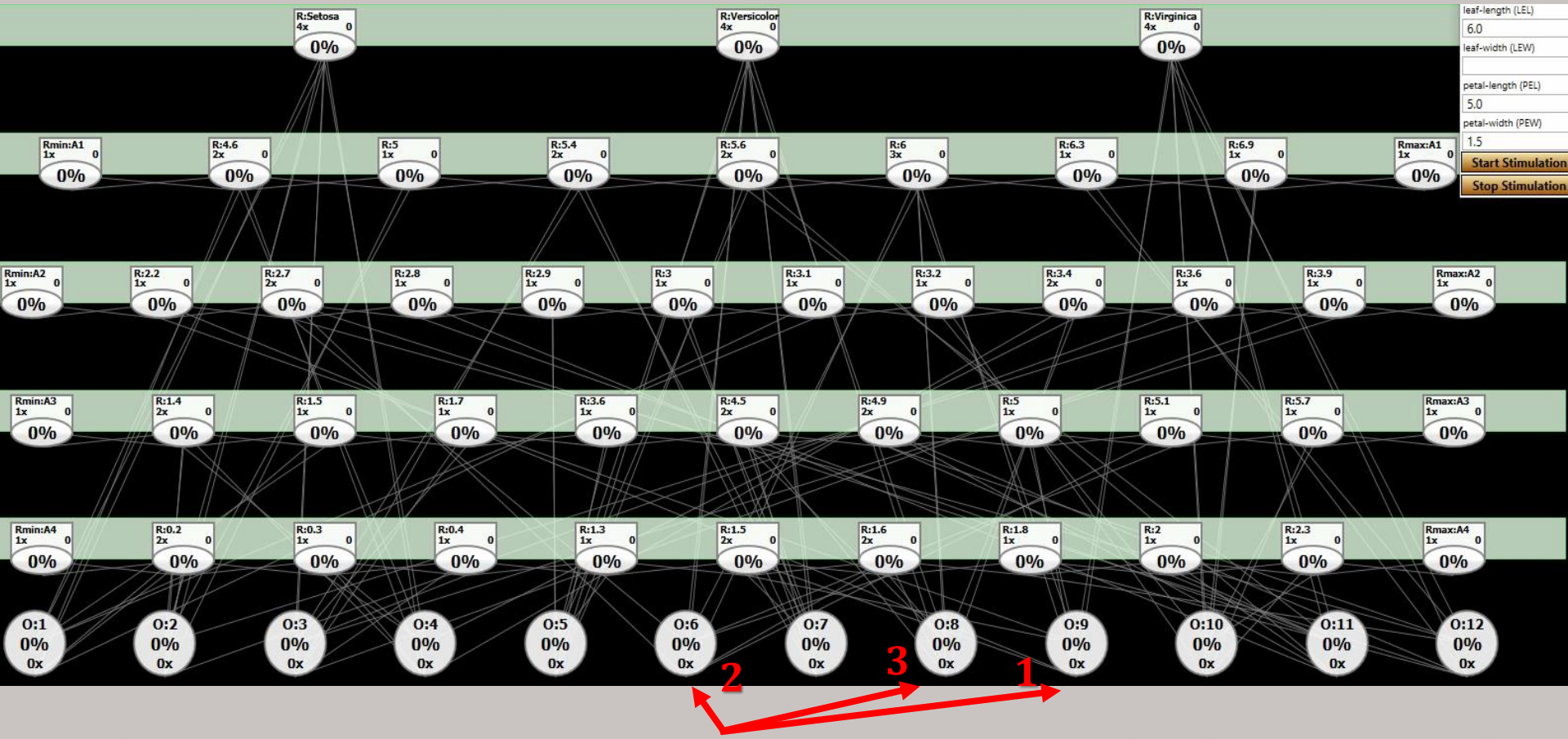
Odległość wyznaczamy zgodnie z przyjętą metryką. W związku z tym, bliżsi sąsiedzi będą mieli mocniejszy głos (większy wpływ) na wynik klasyfikacji, a bardziej odlegli mnieszy.



# Klasyfikacja z wykorzystaniem neuronowych sieci pulsacyjnych

**Dla neuronowej sieci pulsacyjnej zbudowanej dla poniższego zbioru danych uczących dokonajmy klasyfikacji danych wejściowych [?, 6.0, ?, 5.0, 1.5] oraz uzupełnijmy brakującą wartość:**

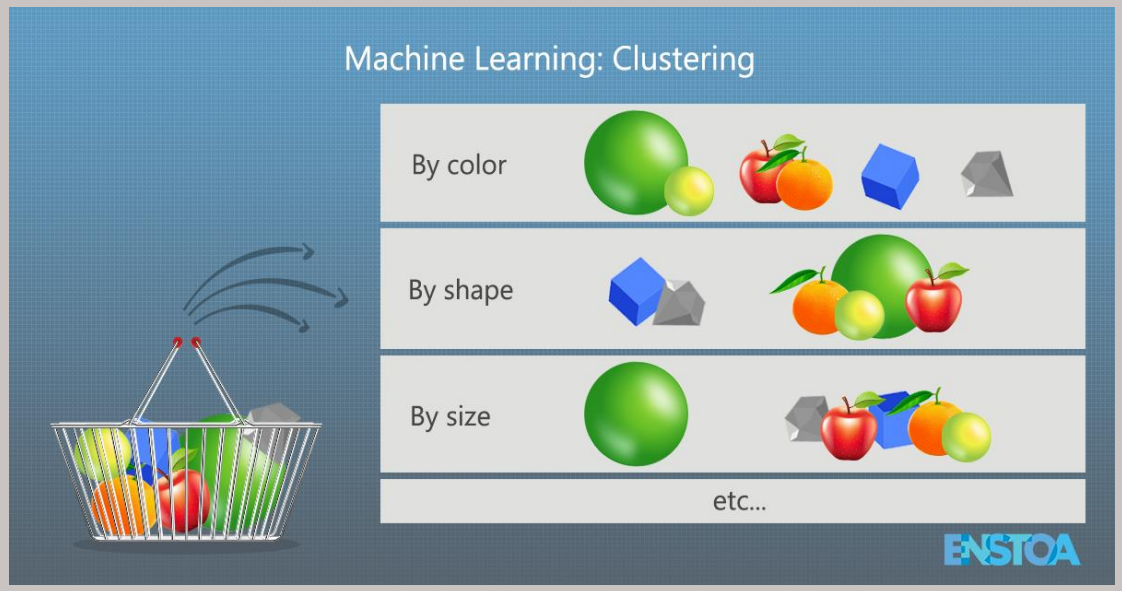
| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| O:1 | Setosa; | 4.6; | 3.1; | 1.5; | 0.2 | O:5 | Versicolor; | 5.6; | 2.9; | 3.6; | 1.3 | O:9 | Virginica; | 6.0; | 2.2; | 5.0; | 1.5 |
| O:2 | Setosa; | 5.0; | 3.6; | 1.4; | 0.2 | O:6 | Versicolor; | 6.0; | 2.7; | 5.1; | 1.6 | O:10 | Virginica; | 6.9; | 3.2; | 5.7; | 2.3 |
| O:3 | Setosa; | 5.4; | 3.9; | 1.7; | 0.4 | O:7 | Versicolor; | 5.4; | 3.0; | 4.5; | 1.5 | O:11 | Virginica; | 5.6; | 2.8; | 4.9; | 2.0 |
| O:4 | Setosa; | 4.6; | 3.4; | 1.4; | 0.3 | O:8 | Versicolor; | 6.0; | 3.4; | 4.5; | 1.6 | O:12 | Virginica; | 6.3; | 2.7; | 4.9; | 1.8 |

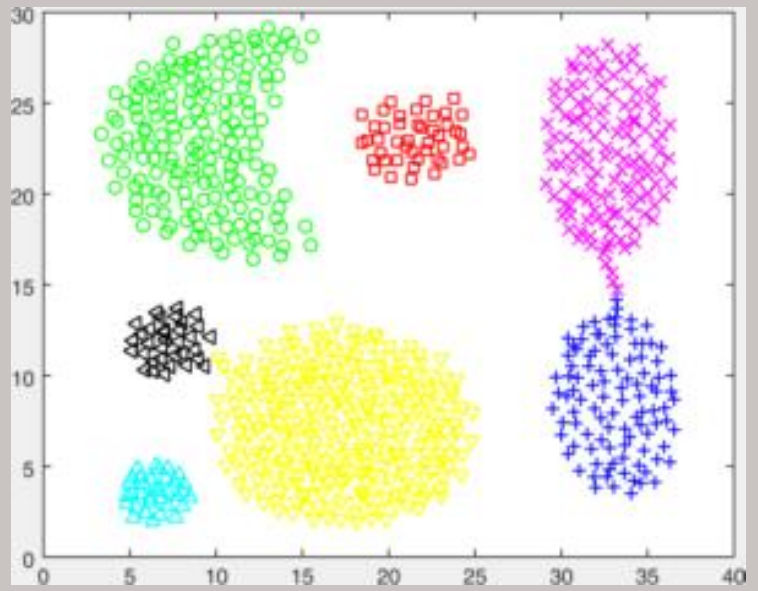



**Najbardziej skojarzone neurony reprezentujące najbardziej podobne wzorce uczące będą pulsować jako pierwsze i najczęściej!**

# Klasteryzacja i Grupowanie

- **Klasteryzacja** to proces grupowania obiektów względem ich podobieństwa.
- **Klasteryzacja** silna zakłada, iż każdy obiekt może znaleźć się tylko w jednym klastrze (grupie).
- **Klasteryzacja** słaba dopuszcza, iż obiekt występuje w więcej niż jednym klastrze (grupie).

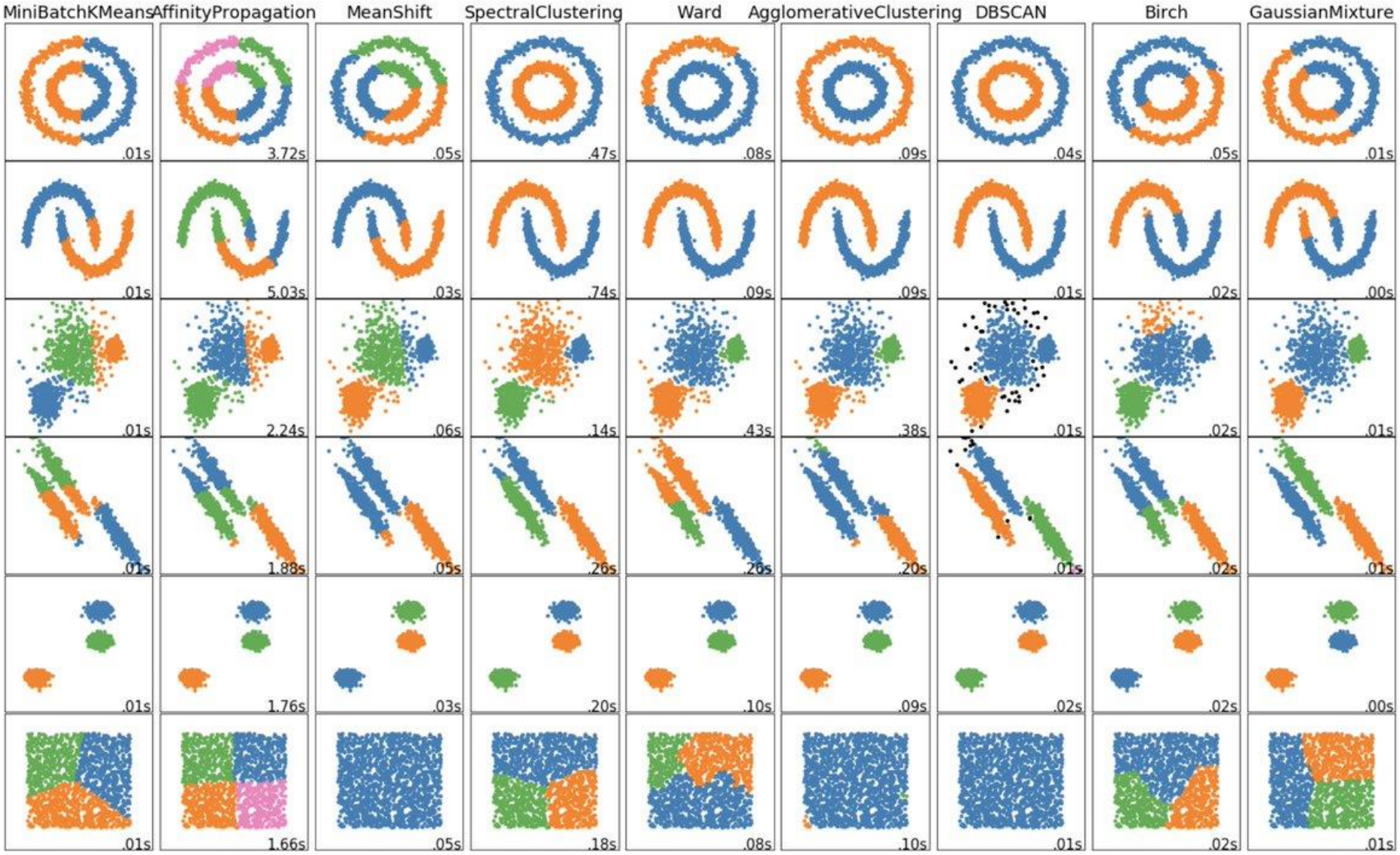



- **Klasteryzacja** to podstawowy proces odpowiedzialny za formowanie się wiedzy w naszym umyśle, jak również w sztucznych inteligentnych systemach.
- W trakcie klasteryzacji brane są pod uwagę wybrane cechy (czasami z różnym priorytetem), na podstawie których dochodzi do **grupowania obiektów**.
- Te wybrane cechy mają zwykle charakter liczbowy, a więc są porównywalne i mogą wyznaczać **podobieństwo** pomiędzy obiektami, ew. jego brak.

# Metody Klasteryzujące

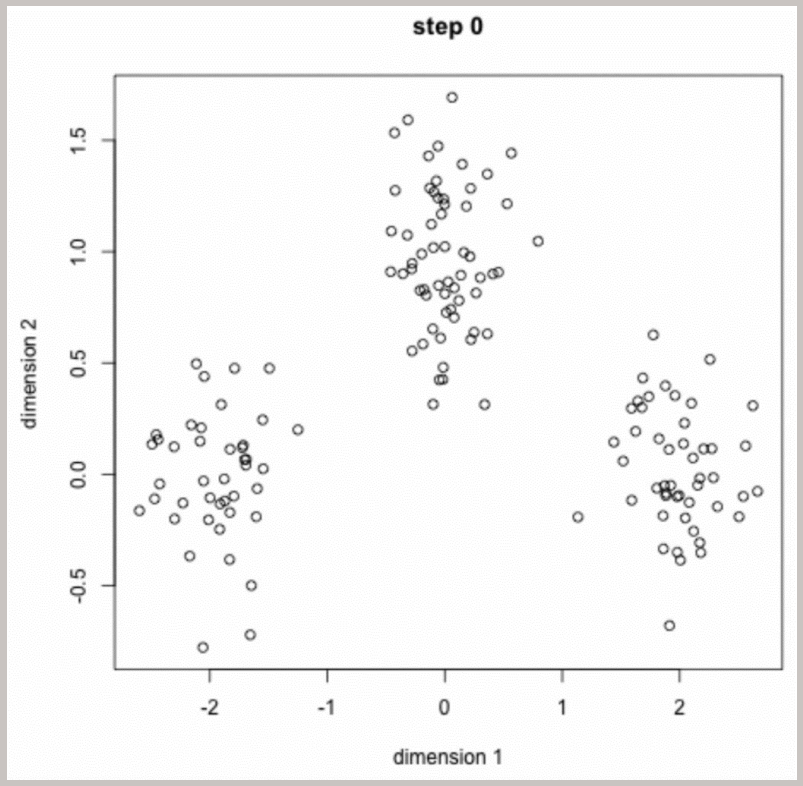
**Jak widać, nie każda metoda działa dobrze na różnego rodzaju zbiorach danych:**

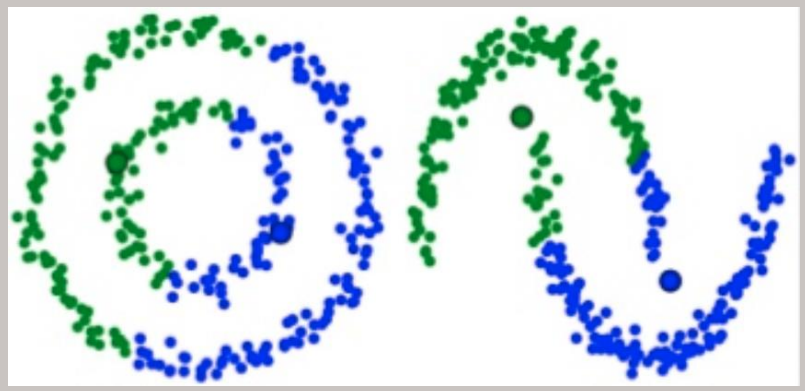


# Metoda K-Średnich i K-Median

## Metoda K-Średnich (algorytm):

1. Wybierz liczbę klastrów (k) i losowo zainicjuj ich punkty środkowe, które są wektorami o tej samej długości co wektory punktów danych.
2. Każdy punkt danych jest objęty punktem skupienia (środek grupy), przy czym odległość do tego punktu jest najmniejsza, tj. każdy punkt danych jest przypisany do najbliższego skupienia (jego środek).
3. Na podstawie tych przypisanych punktów obliczamy nowy środek grupy, biorąc pod uwagę średnią wszystkich punktów w grupie.
4. Powtarzaj te kroki, aż środki grupy nie zmienią się znacznie między kolejnymi iteracjami.
5. Możesz także kilka razy losowo zainicjować środki grup, a następnie ponownie uruchomić algorytm, aby znaleźć najlepsze możliwe klastry.

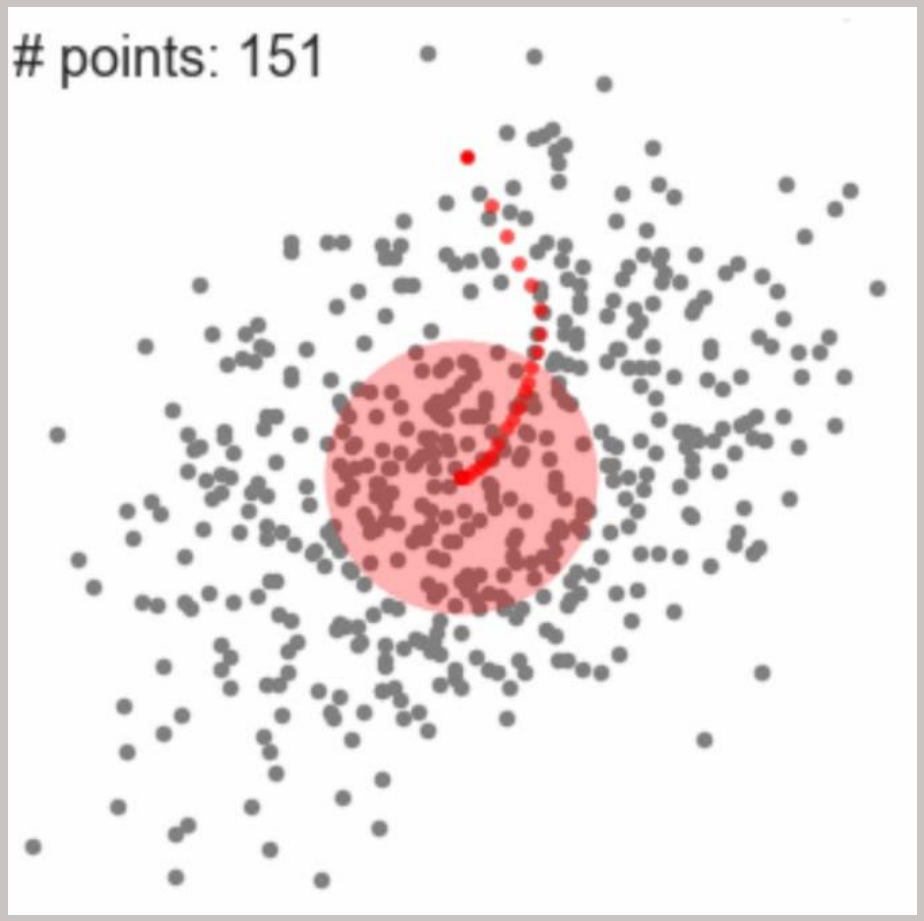


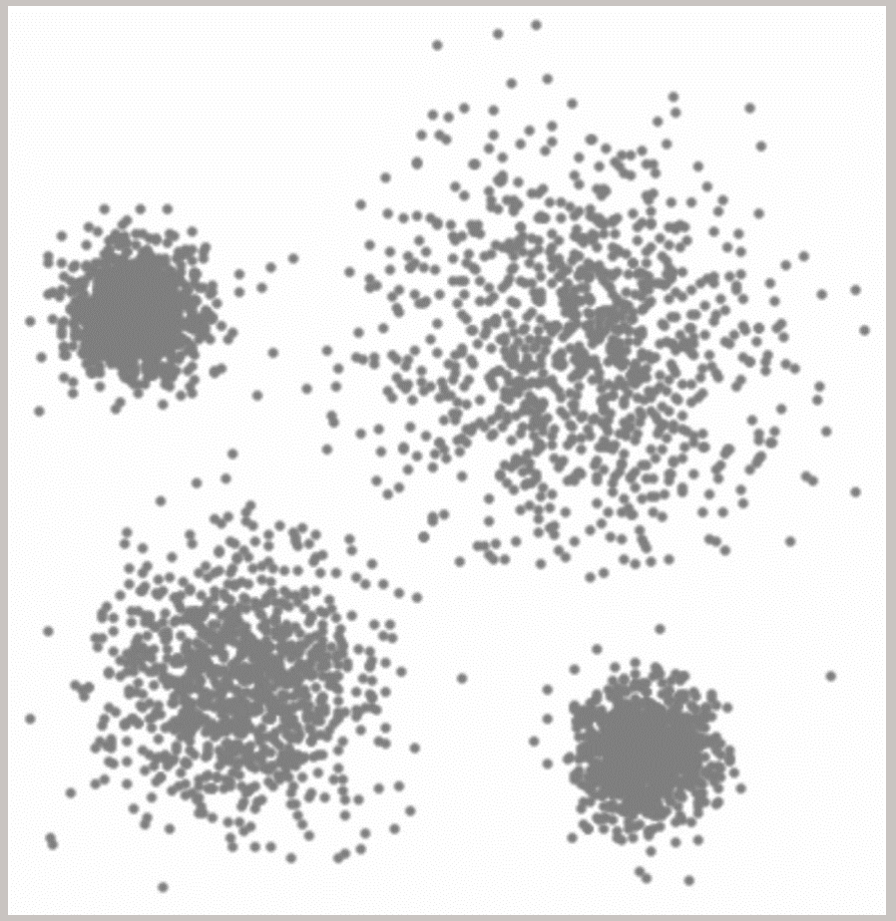



Metoda jest prosta, lecz sprawdza się jednak tylko dla danych o skupieniach centroidalnych!

# Klasteryzacja Przesunięć Średnich

- Rozpocznij od okrągłego przesuwanego okna wyśrodkowanego w losowo wybranym punkcie C o promieniu r jako jądro.
- W każdej iteracji okna są przesuwane w kierunku obszarów o większej gęstości poprzez przesunięcie punktu środkowego do średniej punktów w oknie. Gęstość w przesuwanym oknie jest proporcjonalna do liczby znajdujących się w nim punktów.
- Kontynuuj przesuwanie okien zgodnie z obliczonymi środkami (czerwone punkty reprezentujące średnie), dopóki nie będzie żadnych kierunków, w których można by je przesunąć i pomieścić więcej punktów wewnątrz okrągłej wielowymiarowej kuli, tj. klastra. Na rysunku poruszaj kółkiem, dopóki nie będziemy mogli zwiększyć gęstości, tj. liczby punktów w tym okrągłym oknie.
- Powtarzaj ten proces, aż wszystkie punkty znajdą się w oknach. Gdy wiele okien zachodzi na siebie, okno zawierające najwięcej punktów zostaje zachowane. Punkty są następnie grupowane zgodnie z przesuwanym okrągłym oknem, w którym się znajdują.



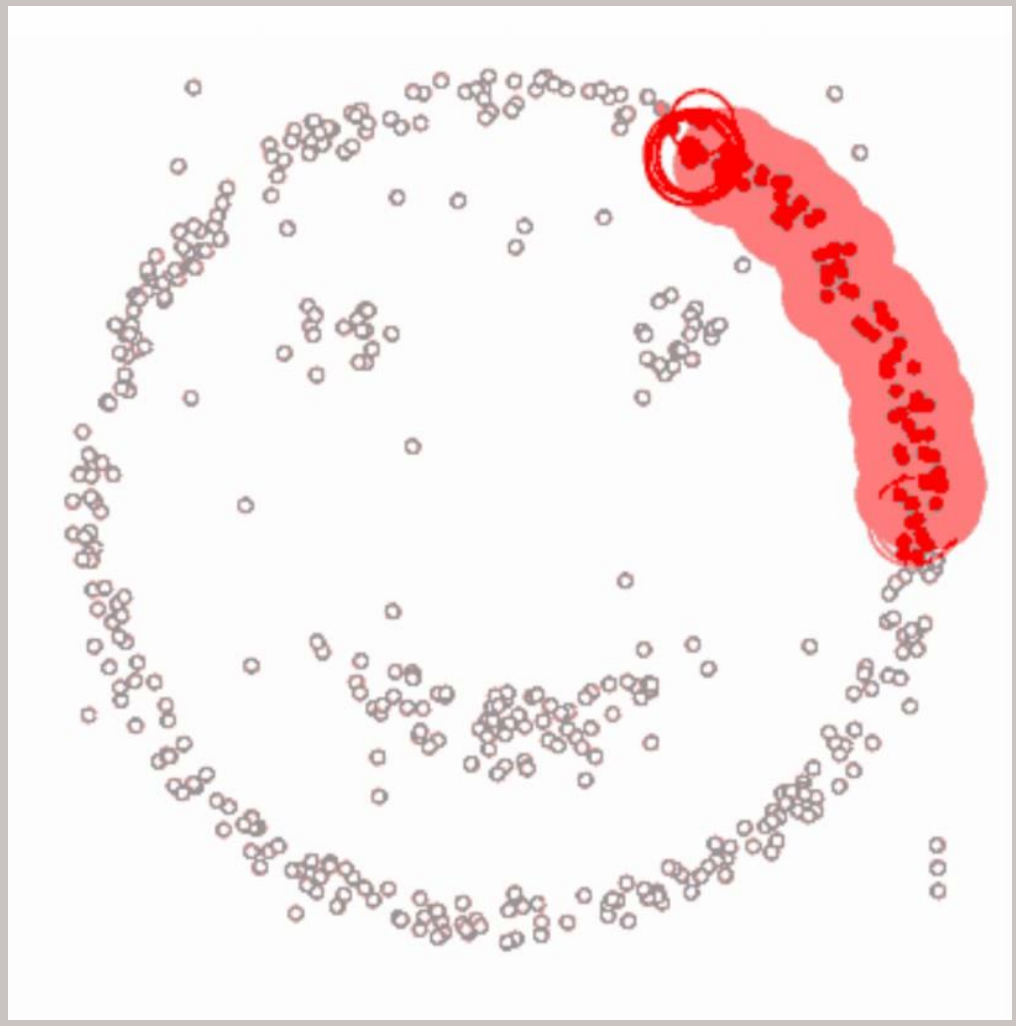



# Metoda klasteryzacji DBSCAN

**To jedna za najpopularniejszych, gdyż daje dobre wyniki klasteryzacji, działa szybko i nie wymusza wstępnego podania ilości klastrów, wskazując zarazem dane odstające.**

**Metoda DBSCAN (algorytm):**

1. Rozpocznij od dowolnego początkowego punktu danych, który nie był jeszcze odwiedzony. Weź pod uwagę jego sąsiedztwo określone przez okrąg o promieniu ε.
2. Jeśli w tej okolicy jest wystarczająca liczba punktów, wówczas rozpoczyna się proces klastrowania, a bieżący punkt danych staje się pierwszym punktem w nowym klastrze. W przeciwnym razie punkt zostanie oznaczony jako szum lub dana odstająca (później taki punkt może stać się częścią pewnej grupy). W obu przypadkach punkt ten jest oznaczony jako „odwiedzony”.
3. Dla tego pierwszego punktu w nowym klastrze punkty w jego sąsiedztwie ε również stają się częścią tego samego klastra. Procedura przypisania wszystkich punktów w sąsiedztwie ε do tego samego klastra jest następnie powtarzana dla wszystkich nowych punktów, które zostały właśnie dodane do aktualnie rozwijającego się klastra.
4. Proces ten jest powtarzany tak długo, aż wszystkie punkty w klastrze zostaną odwiedzone i oznaczone jako „odwiedzone”.
5. Gdy skończymy z bieżącym klastrem, pobierany jest nowy jeszcze nieodwiedzony punkt i począwszy od niego tworzony jest kolejny klaster lub punkt szumu. Proces ten jest powtarzany, dopóki wszystkie punkty nie zostaną oznaczone jako odwiedzone i nie zostaną przypisane do jednego z klastrów czy oznaczone jako szum (dane błędne/odstające).

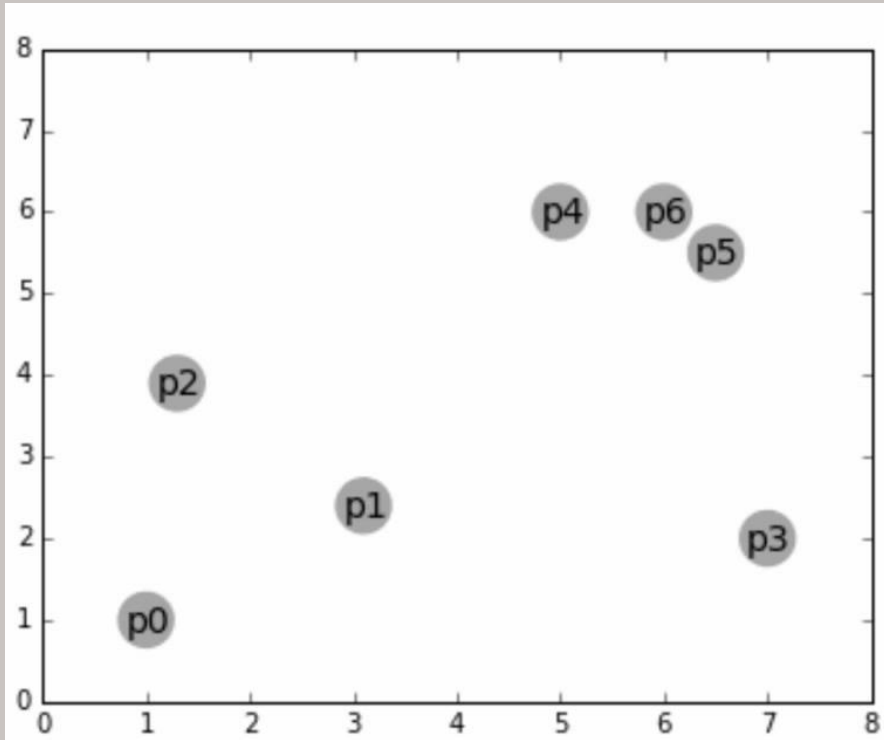


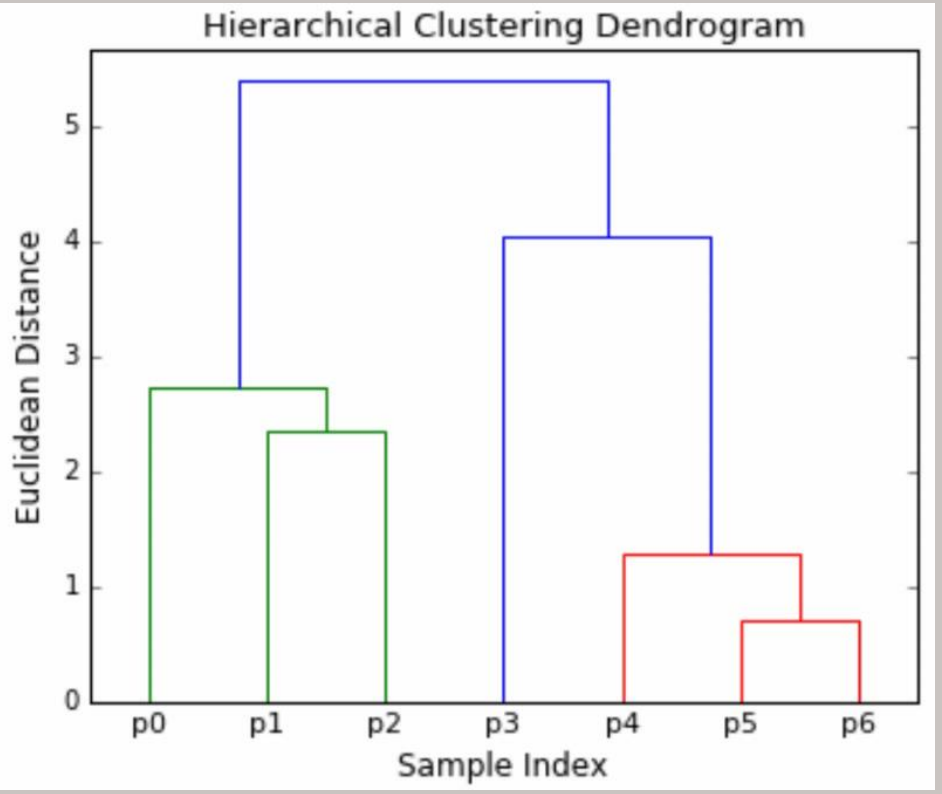
# Hierarchiczna Klasteryzacja Aglomeratywna

**Jest to algorytmem, który na początku traktuje każdy punkt danych jako pojedynczy klaster, a następnie sukcesywnie scala (skupia) pary klastrów na podstawie najbliższej ich odległości, aż wszystkie klastry zostaną połączone w jeden klaster, który zawiera wszystkie punkty danych.**

**Ta hierarchia klastrów jest reprezentowana jako drzewo (dendrogram). Korzeń drzewa to unikalny klaster, który zbiera wszystkie próbki, a liście stanowią klastry z tylko jedną próbką.**

**Metodę możemy przerwać w dowolnym momencie, gdy osiągniemy pożądaną liczbę klastrów.**



# Uczenie Maszynowe

**Uczenie maszynowe pozwala na:**

- **dostosowanie parametrów inteligentnego modelu komputerowego tak, by był w stanie rozpoznawać, klasyfikować lub dokonywać predykcji wartości na podstawie wzorców uczących podobnie, jak uczy się człowiek w trakcie swojego życia.**
- **uogólnienie (generalizację) zdobytych umiejętności rozpoznawania, klasyfikacji lub predykcji na inne obiekty lub działania, które nie były uczone, lecz są podobne do tych wzorców, które pojawiły się w zbiorze uczącym.**

**Uczenie maszynowe to różne metody:**

- **uczenia nadzorowanego („z nauczycielem”) – wymagającego zdefiniowania celu (tj. klas obiektów, np. kategorii operacji bankowej: wynagrodzenie, spłata karty kredytowej, opłaty i odsetki, podatki, itd., czy też pożądanych wartości, np. wartości wyceny nieruchomości dla zadanych parametrów lub predykcji kursów walut czy akcji)**
- **uczenia nienadzorowanego („bez nauczyciela”) – dokonującego zwykle grupowania danych uczących biorąc pod uwagę podobieństwo i częstość ich występowania w zbiorze uczącym.**
- **uczenia ze wzmacnianiem („z krytykiem”) – dopasowującego parametry na podstawie oceny, czy podjęte działanie przez system było pozytywne lub negatywne dla osiągnięcia zdefiniowanego celu, zaś w roli krytyka definiuje się zwykle jakąś funkcję matematyczną.**
- **uczenia motywowanego („wg potrzeb i zagrożeń”) – definiującego cele uczenia poprzez potrzeby i zagrożenia, które kontrolują proces adaptacji i rozwoju modelu.**
- **uczenia skojarzeniowego – na bazie istniejących relacji możliwych do wykrycia w danych uczących, czyli pomiędzy obiektami czy zdarzeniami, tj. podobieństwo, identyczność następstwo w czasie lub bliskość w przestrzeni, częstość występowania, powtarzalność itp.**

# Co przedstawia poniższy rysunek?

Nasze skojarzenia mogą podsuwać nam czasami zaskakujące odpowiedzi!

# Sztuczna Inteligencja w Bankowości

**Sztuczna inteligencja może znaleźć liczne zastosowania w bankowości:**

- Ocena ryzyka kredytowego potencjalnych lub obecnych klientów banku.
- Szacunkowa wycena nieruchomości pod kątem kredytów hipotecznych.
- Predykcja (przewidywanie) kursów walut lub akcji.
- Automatyczne wyszukiwanie docelowych grup klientów o podobnych cechach pod kątem promocji (klasteryzacja).

- Wykrywania i klasyfikacji oszustw bankowych.
- Automatyczna klasyfikacji kategorii przelewów bankowych na podstawie tytułów przelewów, kwot, powtarzalności, numeru konta wysyłającego lub docelowego, nazwy instytucji.

- Rozpoznawanie twarzy, głosu i innych cech biometrycznych pod kątem systemów bezpieczeństwa i nieinwazyjnej identyfikacji lub potwierdzania tożsamości klientów (PINy biometryczne, np. głosowe).
- Wspomaganie automatycznych bankowych serwisów telefonicznych, np. poprzez rozpoznawanie emocji (*sentimental analysis*).
- Tworzenie animowanych chatbotów lingwistycznych ułatwiających automatyczną obsługę klientów przez komórkę, tablet i komputer.
- Tworzenie skojarzeniowo-kognitywnych baz wiedzy o klientach, zdolnych automatycznie wyszukiwać ich ważne cechy i grupy oraz podpowiadać analitykom bankowym możliwe rozwiązania.

# Od danych i informacji, do wiedzy i inteligencji!

## Jak stworzyć i rozwijać komputerowe inteligentne systemy oparte o wiedzę?

# Dane

**Dane** to zbiory liczb, znaków, symboli, sygnałów, bodźców, miar fizycznych lub empirycznych oraz surowych wartości opisujących różne przedmioty lub działania, np. *36.6ºC, T, $, φ, 25cm, !*

Niepowiązane dane są bezużyteczne, ponieważ dane przyjmują znaczenie, gdy są powiązane z innymi danymi. Dane mogą być surowe, niespójne, niezorganizowane…

Dane zazwyczaj opisują fakty i są nośnikiem informacji.

# Tabele Danych

W informatyce stosujemy przede wszystkim tabele do przechowywania i zarządzania danymi,

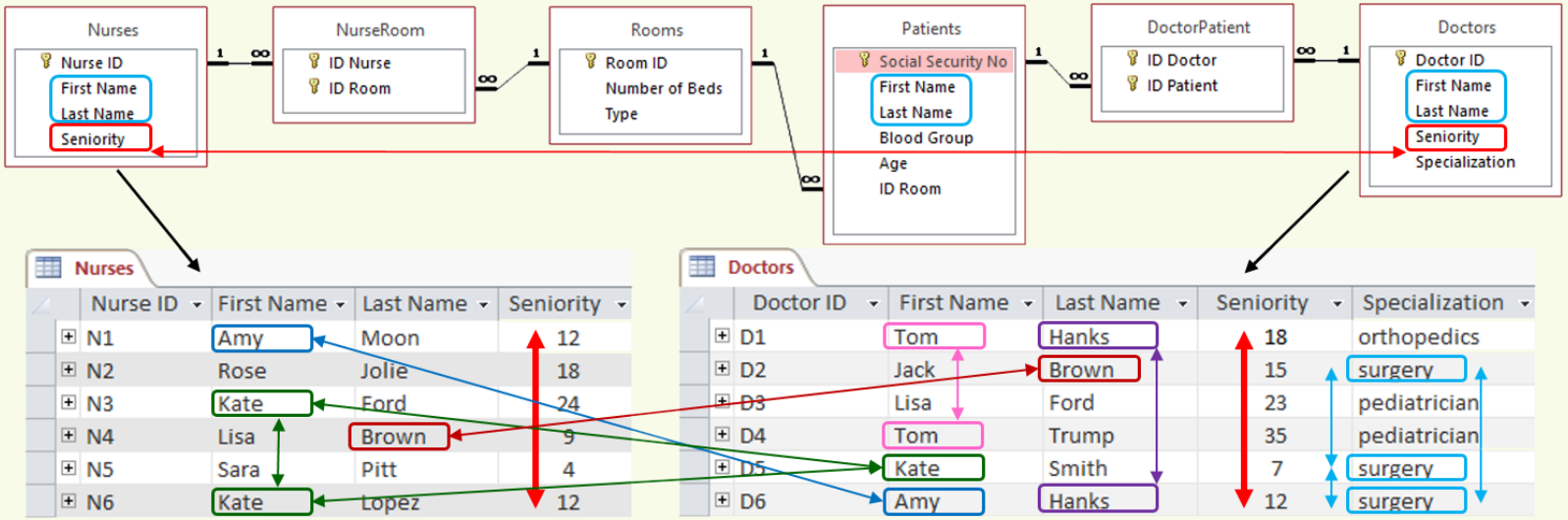
| | ATTRIBUTES | | | | |
|---|---|---|---|---|---|
| SAMPLE OBJECTS | SEPAL LENGTH | SEPAL WIDTH | PETAL LENGTH | PETAL WIDTH | CLASS LABEL |
| O1 | 5.4 | 3.0 | 4.5 | 1.5 | Versicolor |
| O2 | 6.3 | 3.3 | 4.7 | 1.6 | Versicolor |
| O3 | 6.0 | 2.7 | 5.1 | 1.6 | Versicolor |
| O4 | 6.7 | 3.0 | 5.0 | 1.7 | Versicolor |
| O5 | 6.0 | 2.2 | 5.0 | 1.5 | Virginica |
| O6 | 5.9 | 3.2 | 4.8 | 1.8 | Versicolor |
| O7 | 6.0 | 3.0 | 4.8 | 1.8 | Virginica |
| O8 | 5.7 | 2.5 | 5.0 | 2.0 | Virginica |
| O9 | 6.5 | 3.2 | 5.1 | 2.0 | Virginica |

lecz zwykle wykorzystywane relacje podczas wnioskowania, tj. podobieństwo, minima, maksima, ilość duplikatów, trzeba wyszukiwać, co wiąże się z wymiernym kosztem czasowym. Im więcej jest danych, tym więcej czasu i zasobów tracimy na samo wyszukiwanie relacji, które służą do dalszego przetwarzania danych i wnioskowania!

Such relations are not enough!

# Relacyjne Bazy Danych

Relacyjne bazy danych wiążą dane tylko w poziomie, a nie w pionie (w tabelach), więc nadal musimy szukać duplikatów, sąsiadów lub podobnych wartości i obiektów.




**Nurses**

| Nurse ID | First Name | Last Name | Seniority |
|---|---|---|---|
| N1 | Amy | Moon | 12 |
| N2 | Rose | Jolie | 18 |
| N3 | Kate | Ford | 24 |
| N4 | Lisa | Brown | 9 |
| N5 | Sara | Pitt | 4 |
| N6 | Kate | Lopez | 12 |

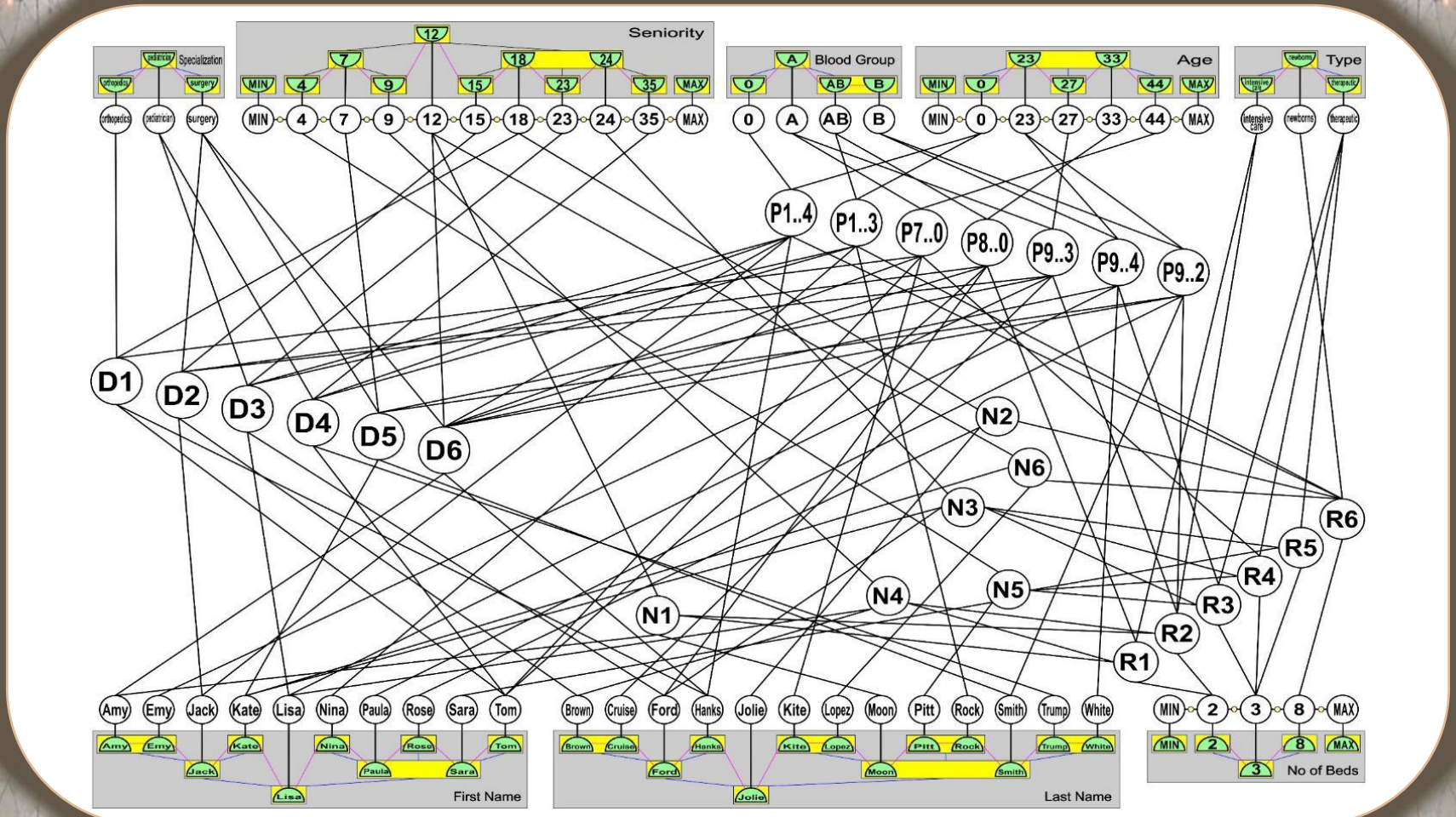
**Doctors**

| Doctor ID | First Name | Last Name | Seniority | Specialization |
|---|---|---|---|---|
| D1 | Tom | Hanks | 18 | orthopedics |
| D2 | Jack | Brown | 15 | surgery |
| D3 | Lisa | Ford | 23 | pediatrician |
| D4 | Tom | Trump | 35 | pediatrician |
| D5 | Kate | Smith | 7 | surgery |
| D6 | Amy | Hanks | 12 | surgery |

Dane nie są perfekcyjnie powiązane nawet w poziomie, więc zwykle występuje wiele niepowiązanych duplikatów tych samych kategorii w różnych tabelach
W rezultacie tracimy dużo czasu obliczeniowego (i pamięci), aby wyszukać niezbędne relacje pomiędzy danymi, aby obliczyć wyniki lub wyciągnąć wnioski.

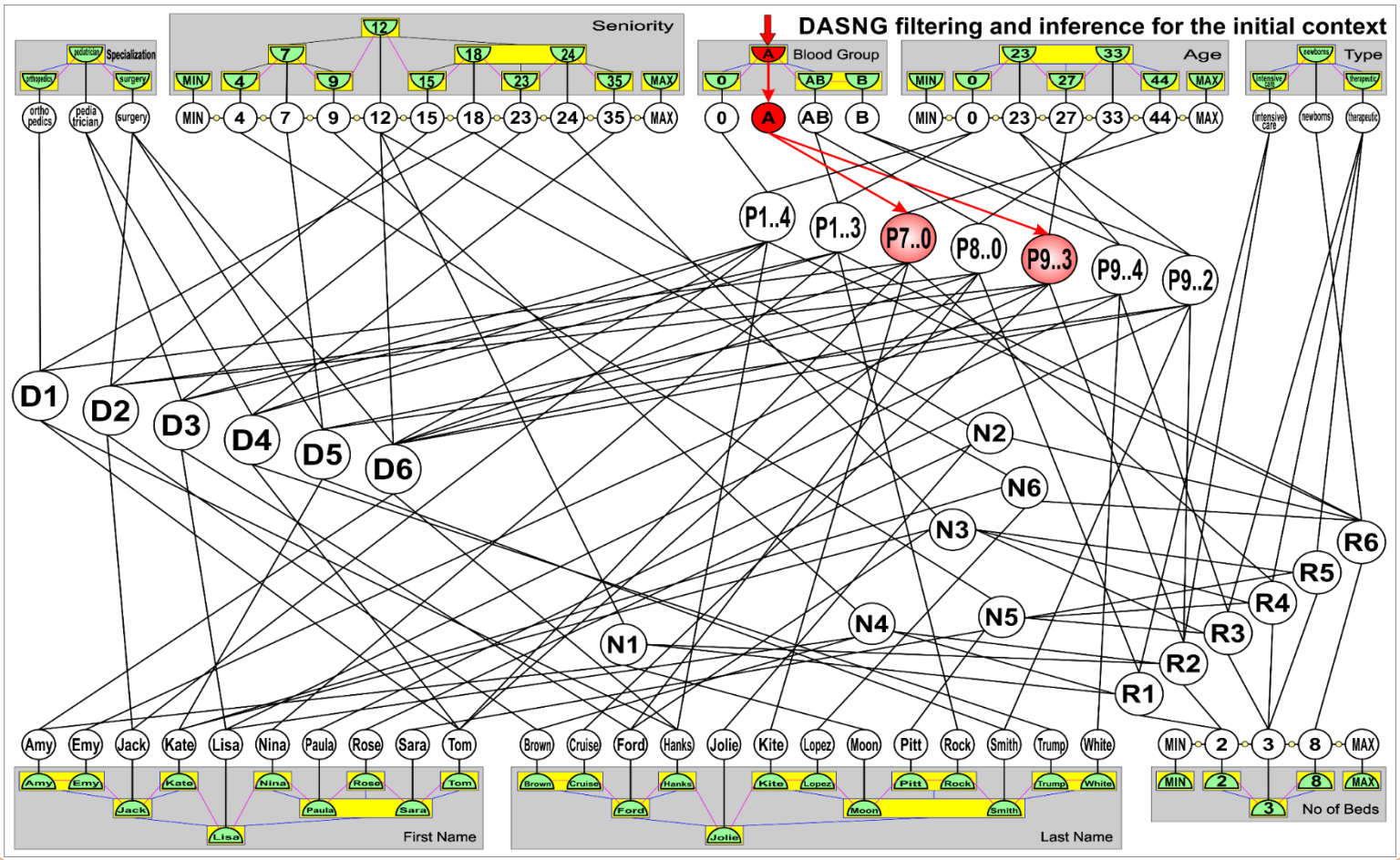
Czy to rozsądne tracić większość czasu obliczeń na poszukiwanie relacji pomiędzy danymi?!

# Struktury Asocjacyjne



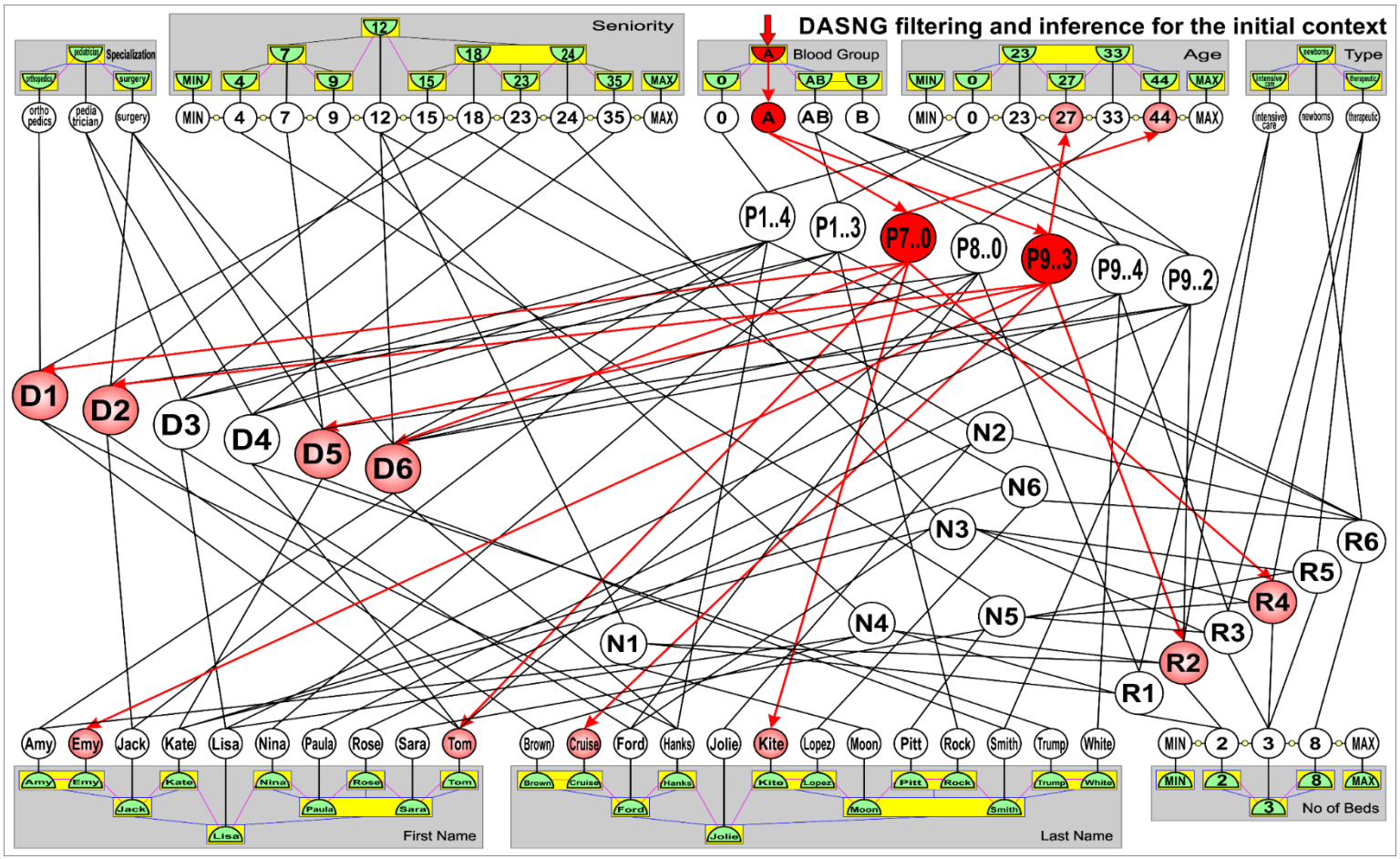

Dane można ze sobą skojarzyć na podstawie istniejących relacji i wykorzystać do wnioskowania.

# Wnioskowanie Asocjacyjne



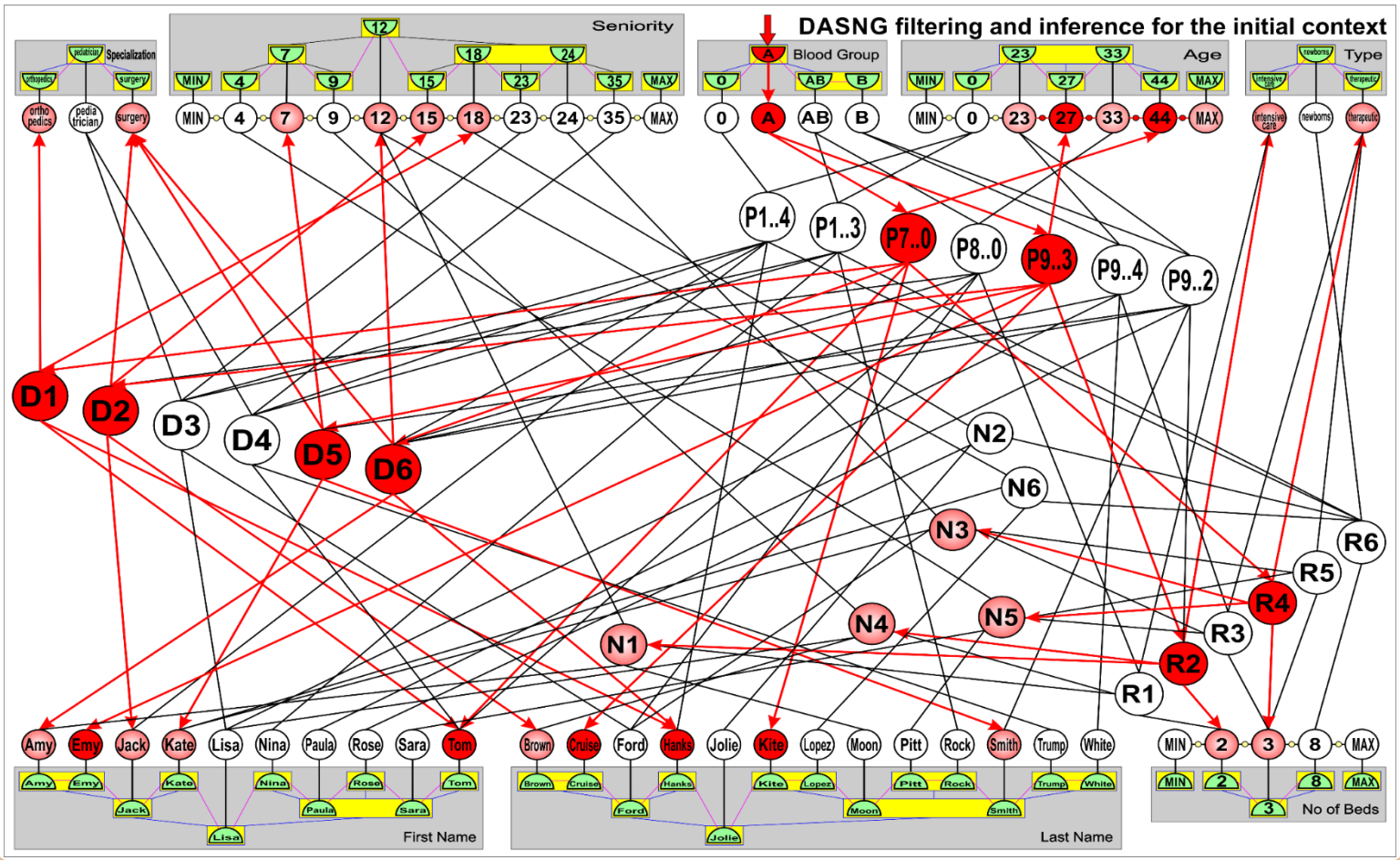

Dane można ze sobą skojarzyć na podstawie istniejących relacji i wykorzystać do wnioskowania.

# Wnioskowanie Asocjacyjne




Dane można ze sobą skojarzyć na podstawie istniejących relacji i wykorzystać do wnioskowania.

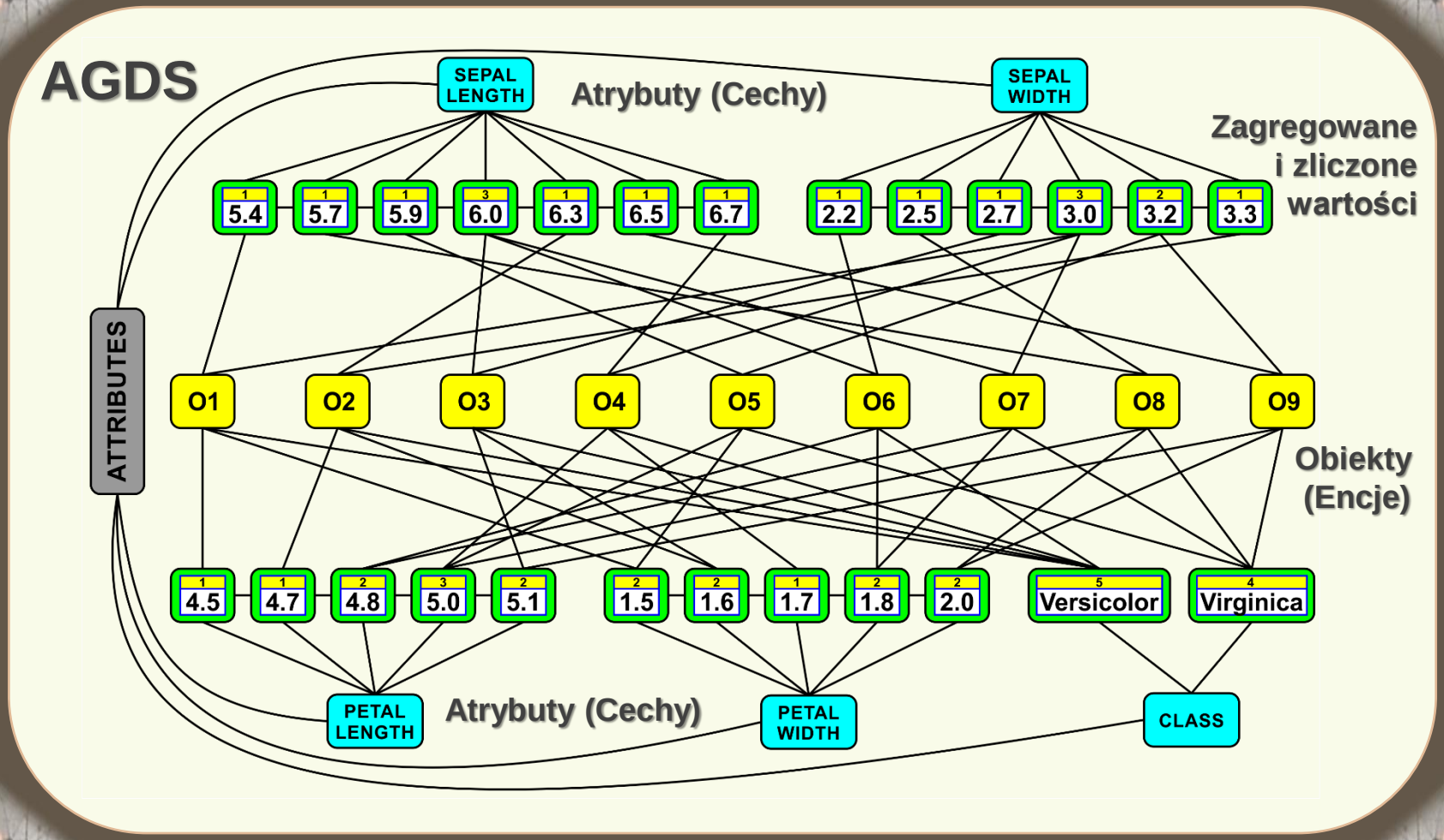
# Wnioskowanie Asocjacyjne




Dane można ze sobą skojarzyć na podstawie istniejących relacji i wykorzystać do wnioskowania.

# AGDS

## Asocjacyjne Grafowe Struktury Danych




Połączenia reprezentują różne relacje między elementami AGDS, tj. podobieństwo, bliskość, sąsiedztwo, definicja itp.

# AGDS
## Associative Graph Data Structure

Asocjacyjne grafowe struktury danych składają się z:

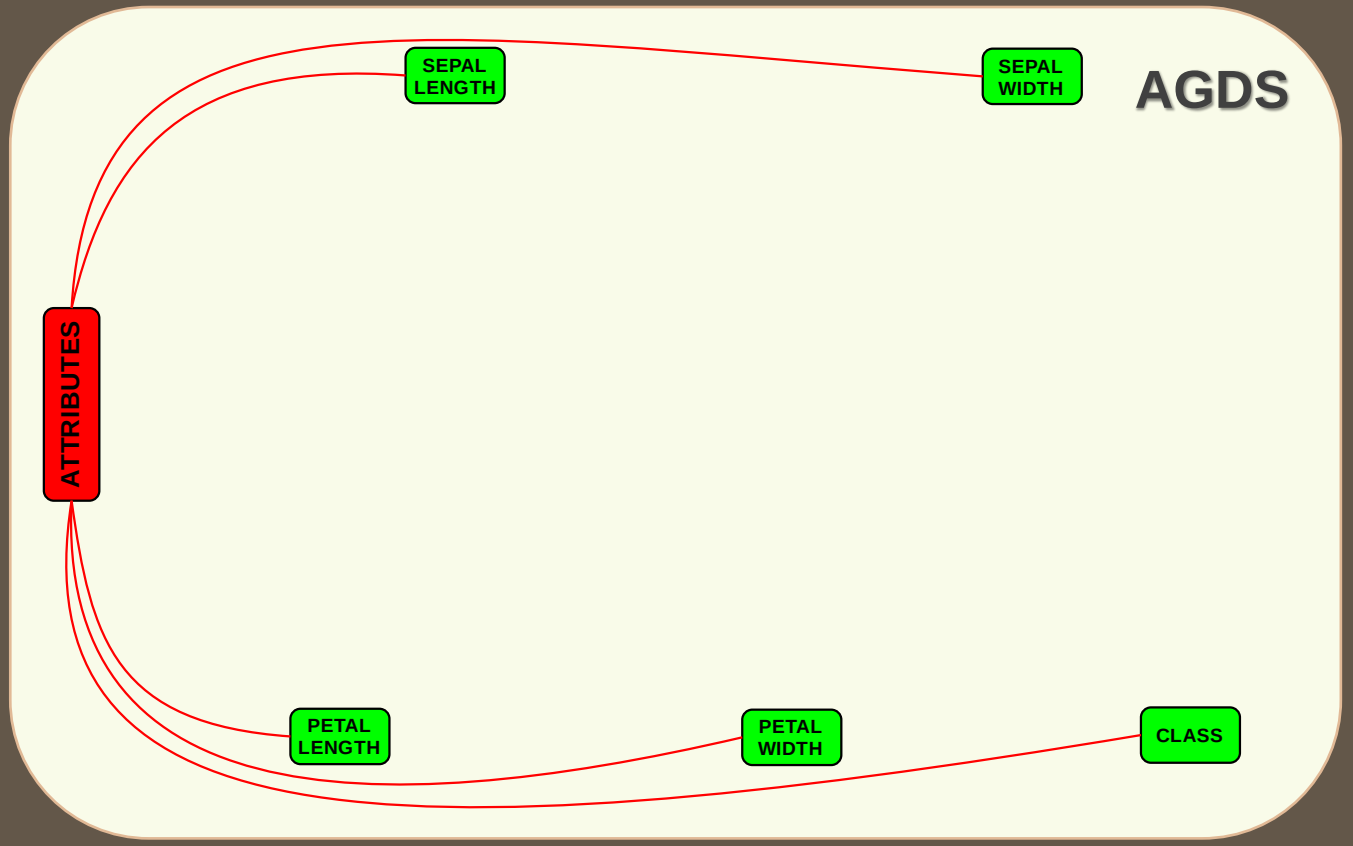
- **Wierzchołków** reprezentujących dane o pojedynczej wartości, zakresy, podzbiory, obiekty, klastry, klasy itp.
- **Krawędzi** reprezentujących różne relacje między węzłami, takie jak podobieństwo, definicja, sekwencja, sąsiedztwo itp.
- Możemy je użyć do przedstawienia dowolnych danych tabelarycznych bez utraty informacji, tj. przekształcenie tabeli w strukturę AGDS jest **odwracalne**, więc zawsze możemy przekształcić dane z powrotem w strukturę tabelaryczną.

Ta asocjacyjna transformacja wzbogaca jednak zestaw bezpośrednio reprezentowanych relacji pomiędzy danymi i obiektami przechowywanymi w przekształconych tabelach na strukturę grafu.

# Transformacja Asocjacyjna

Proces asocjacyjnego przekształcania tabeli w strukturę AGDS rozpoczyna się od utworzenia węzła atrybutów i węzłów reprezentujących etykiety atrybutów. Etykiety atrybutów zostaną połączone z unikatowymi wartościami atrybutów, które zostaną posortowane i zliczone podczas wstawiania kolejnych wartości.

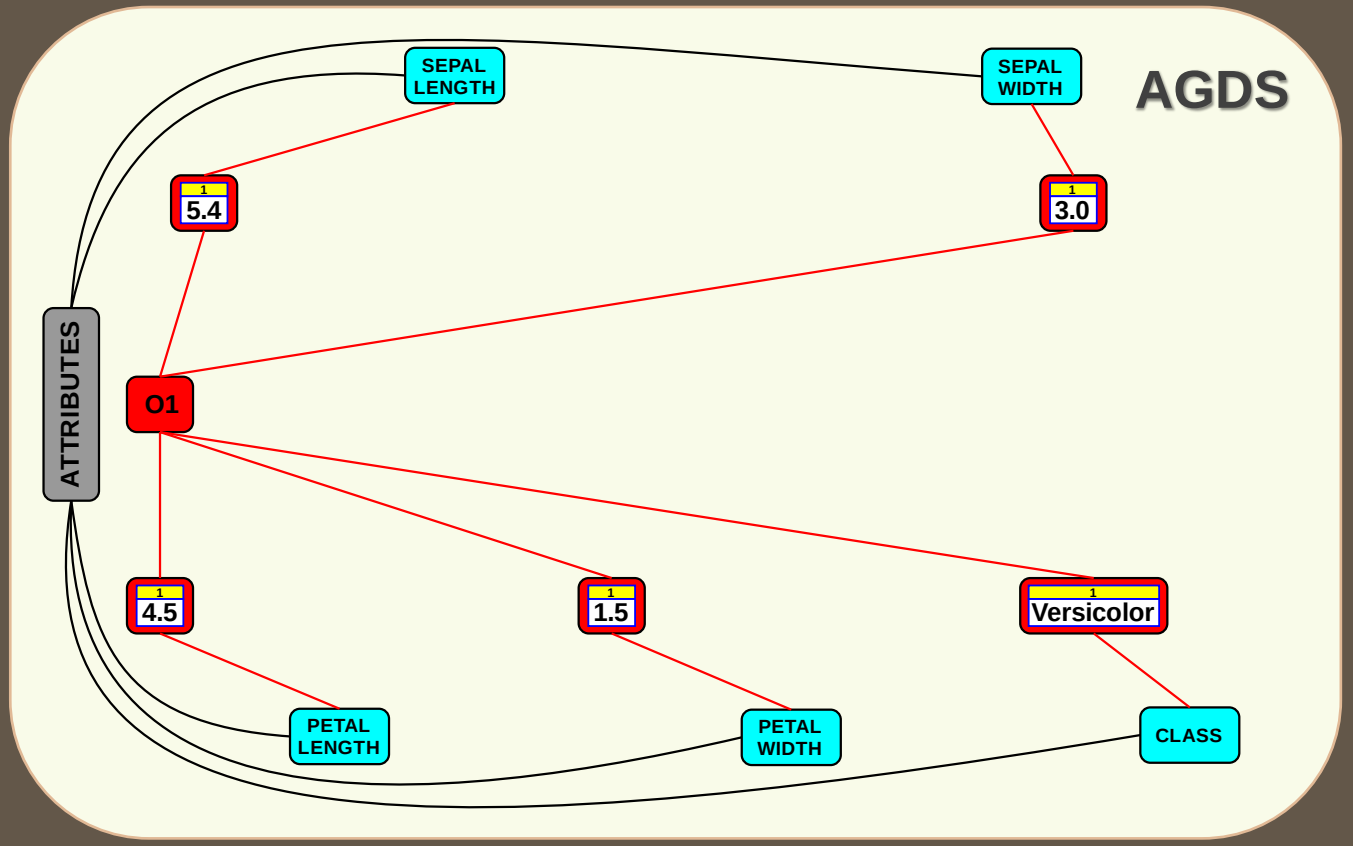
| DATASET | ATTRIBUTES | | | | |
|---|---|---|---|---|---|
| SAMPLE OBJECTS | SEPAL LENGTH | SEPAL WIDTH | PETAL LENGTH | PETAL WIDTH | CLASS LABEL |
| O1 | 5.4 | 3.0 | 4.5 | 1.5 | Versicolor |
| O2 | 6.3 | 3.3 | 4.7 | 1.6 | Versicolor |
| O3 | 6.0 | 2.7 | 5.1 | 1.6 | Versicolor |
| O4 | 6.7 | 3.0 | 5.0 | 1.7 | Versicolor |
| O5 | 6.0 | 2.2 | 5.0 | 1.5 | Virginica |
| O6 | 5.9 | 3.2 | 4.8 | 1.8 | Versicolor |
| O7 | 6.0 | 3.0 | 4.8 | 1.8 | Virginica |
| O8 | 5.7 | 2.5 | 5.0 | 2.0 | Virginica |
| O9 | 6.5 | 3.2 | 5.1 | 2.0 | Virginica |



# Transformacja Asocjacyjna

Do poprzednio utworzonej struktury szkieletowej dodany jest pierwszy obiekt (rekord, encja) O1 wraz ze wszystkimi cechami definiującymi go.
Cechy i obiekt są połączone wzajemnie z węzłami etykiet atrybutów.

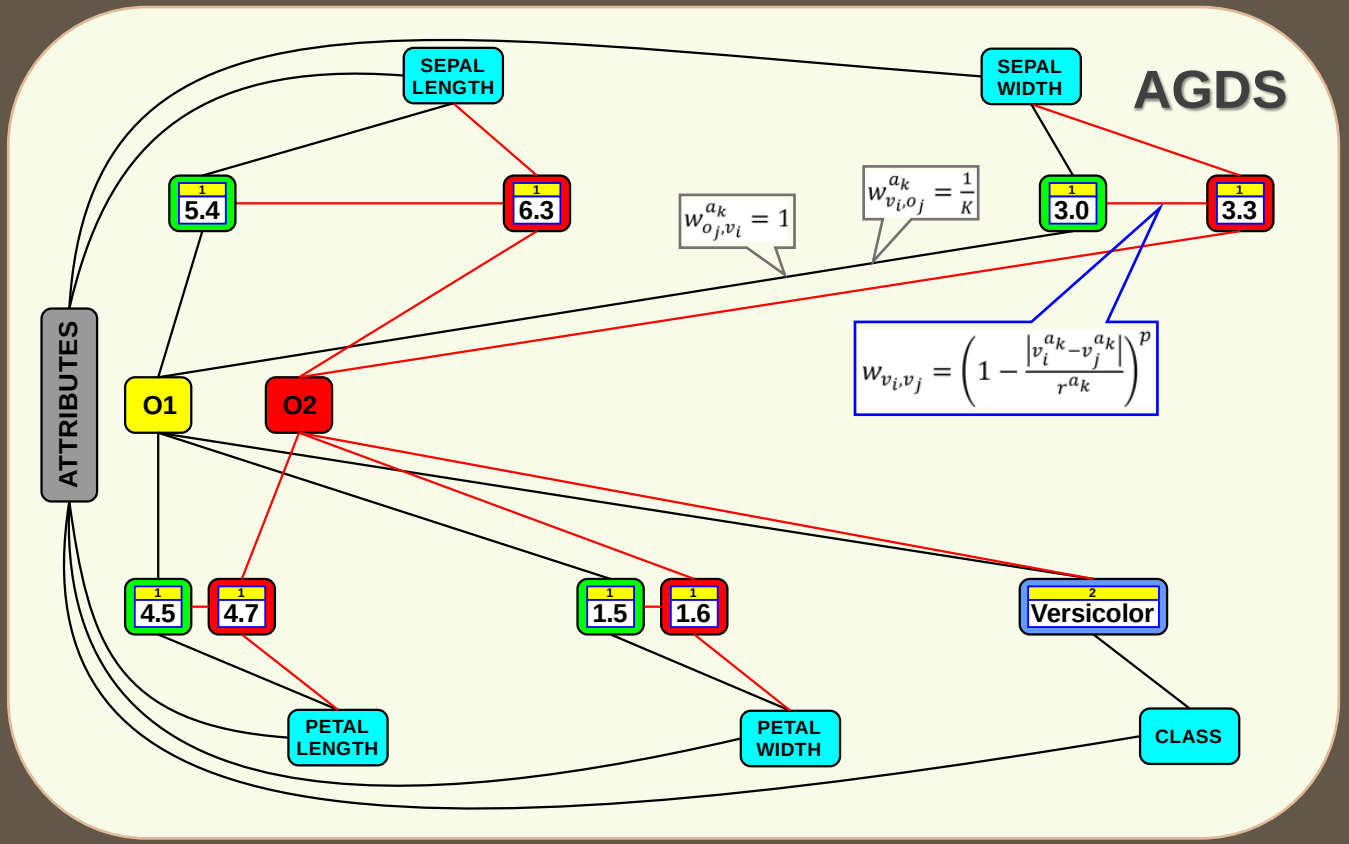
| DATASET | ATTRIBUTES | | | | |
|---|---|---|---|---|---|
| SAMPLE OBJECTS | SEPAL LENGTH | SEPAL WIDTH | PETAL LENGTH | PETAL WIDTH | CLASS LABEL |
| O1 | 5.4 | 3.0 | 4.5 | 1.5 | Versicolor |
| O2 | 6.3 | 3.3 | 4.7 | 1.6 | Versicolor |
| O3 | 6.0 | 2.7 | 5.1 | 1.6 | Versicolor |
| O4 | 6.7 | 3.0 | 5.0 | 1.7 | Versicolor |
| O5 | 6.0 | 2.2 | 5.0 | 1.5 | Virginica |
| O6 | 5.9 | 3.2 | 4.8 | 1.8 | Versicolor |
| O7 | 6.0 | 3.0 | 4.8 | 1.8 | Virginica |
| O8 | 5.7 | 2.5 | 5.0 | 2.0 | Virginica |
| O9 | 6.5 | 3.2 | 5.1 | 2.0 | Virginica |



# Transformacja Asocjacyjna

Drugi obiekt O2 jest dodawany do struktury AGDS, a wszystkie jego definiujące cechy są reprezentowane przez węzły wartości, które są połączone z etykietami atrybutów.

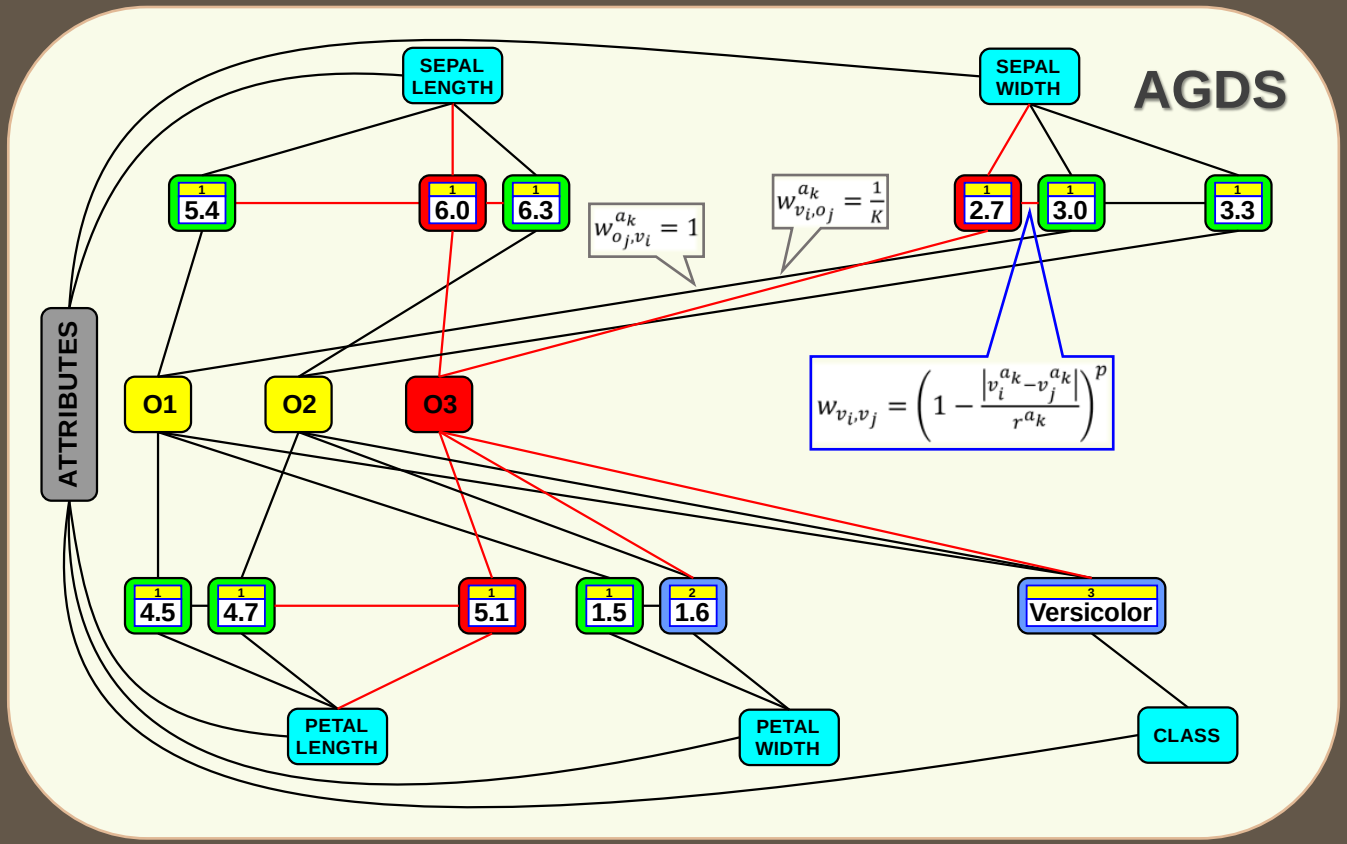
| DATASET | ATTRIBUTES | | | | |
|---|---|---|---|---|---|
| SAMPLE OBJECTS | SEPAL LENGTH | SEPAL WIDTH | PETAL LENGTH | PETAL WIDTH | CLASS LABEL |
| O1 | 5.4 | 3.0 | 4.5 | 1.5 | Versicolor |
| O2 | 6.3 | 3.3 | 4.7 | 1.6 | Versicolor |
| O3 | 6.0 | 2.7 | 5.1 | 1.6 | Versicolor |
| O4 | 6.7 | 3.0 | 5.0 | 1.7 | Versicolor |
| O5 | 6.0 | 2.2 | 5.0 | 1.5 | Virginica |
| O6 | 5.9 | 3.2 | 4.8 | 1.8 | Versicolor |
| O7 | 6.0 | 3.0 | 4.8 | 1.8 | Virginica |
| O8 | 5.7 | 2.5 | 5.0 | 2.0 | Virginica |
| O9 | 6.5 | 3.2 | 5.1 | 2.0 | Virginica |



# Transformacja Asocjacyjna

Podczas dodawania następnego obiektu możemy zauważyć, że nie wszystkie cechy definiujące utworzyły nowe węzły wartości (np. 1,6 szerokości płatka lub Versicolor etykiety klasy), ponieważ niektóre wartości były już reprezentowane w tej strukturze, więc duplikaty (na niebiesko) zostały zebrane i zliczone.

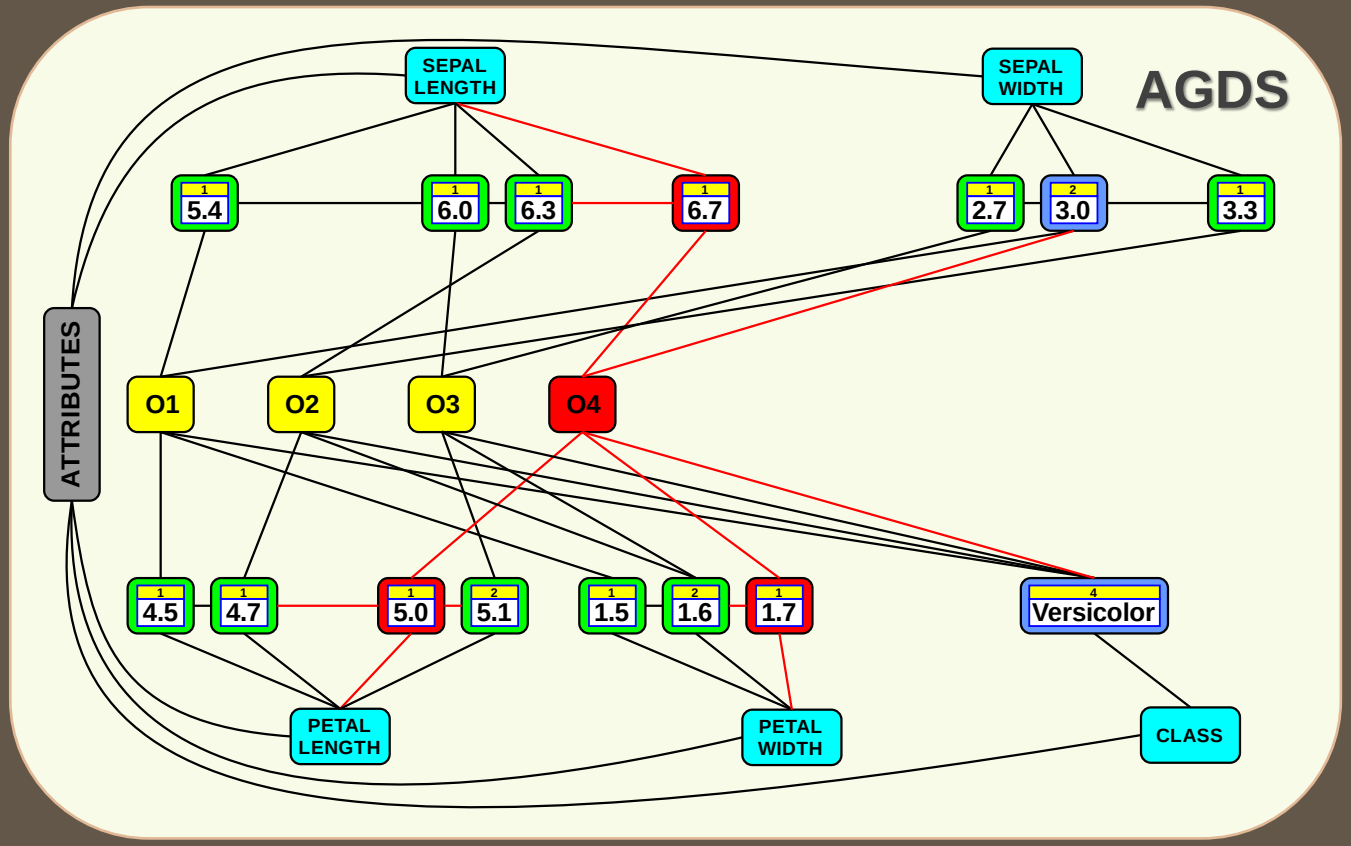
| DATASET | ATTRIBUTES | | | | |
|---|---|---|---|---|---|
| SAMPLE OBJECTS | SEPAL LENGTH | SEPAL WIDTH | PETAL LENGTH | PETAL WIDTH | CLASS LABEL |
| O1 | 5.4 | 3.0 | 4.5 | 1.5 | Versicolor |
| O2 | 6.3 | 3.3 | 4.7 | 1.6 | Versicolor |
| O3 | 6.0 | 2.7 | 5.1 | 1.6 | Versicolor |
| O4 | 6.7 | 3.0 | 5.0 | 1.7 | Versicolor |
| O5 | 6.0 | 2.2 | 5.0 | 1.5 | Virginica |
| O6 | 5.9 | 3.2 | 4.8 | 1.8 | Versicolor |
| O7 | 6.0 | 3.0 | 4.8 | 1.8 | Virginica |
| O8 | 5.7 | 2.5 | 5.0 | 2.0 | Virginica |
| O9 | 6.5 | 3.2 | 5.1 | 2.0 | Virginica |



# Transformacja Asocjacyjna

Poniższy obiekt O4 tworzy nowe węzły wartości, lecz również wykorzystuje dwa z istniejących węzłów wartości, zwiększając ich liczniki zagregowanych duplikatów. Proces agregacji duplikatów jest bardzo ważny z punktu widzenia reprezentacji wiedzy, ponieważ pozwala na wyciągnięcie głębszych wniosków.

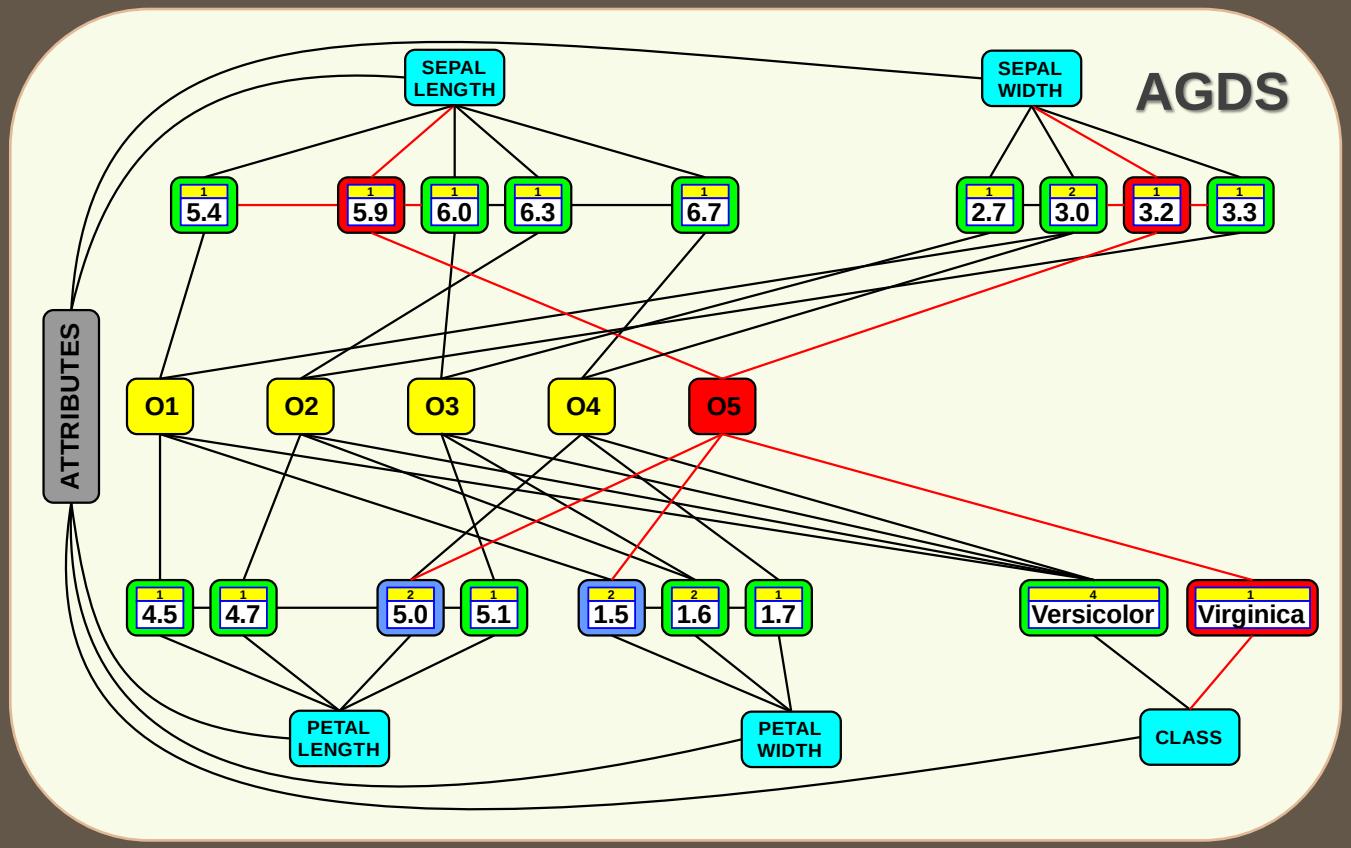
| DATASET | ATTRIBUTES | | | | |
|---|---|---|---|---|---|
| SAMPLE OBJECTS | SEPAL LENGTH | SEPAL WIDTH | PETAL LENGTH | PETAL WIDTH | CLASS LABEL |
| O1 | 5.4 | 3.0 | 4.5 | 1.5 | Versicolor |
| O2 | 6.3 | 3.3 | 4.7 | 1.6 | Versicolor |
| O3 | 6.0 | 2.7 | 5.1 | 1.6 | Versicolor |
| O4 | 6.7 | 3.0 | 5.0 | 1.7 | Versicolor |
| O5 | 6.0 | 2.2 | 5.0 | 1.5 | Virginica |
| O6 | 5.9 | 3.2 | 4.8 | 1.8 | Versicolor |
| O7 | 6.0 | 3.0 | 4.8 | 1.8 | Virginica |
| O8 | 5.7 | 2.5 | 5.0 | 2.0 | Virginica |
| O9 | 6.5 | 3.2 | 5.1 | 2.0 | Virginica |



# Transformacja Asocjacyjna

Obiekt O5 reprezentuje inną (nową) klasę Virginica, dlatego dodano nowy węzeł reprezentujący tę klasę.
Zauważmy, że wartości symboliczne (nienumeryczne) nie są łączone jako cechy liczbowe, które zawsze są połączone z sąsiadami, a połączenia są ważone.

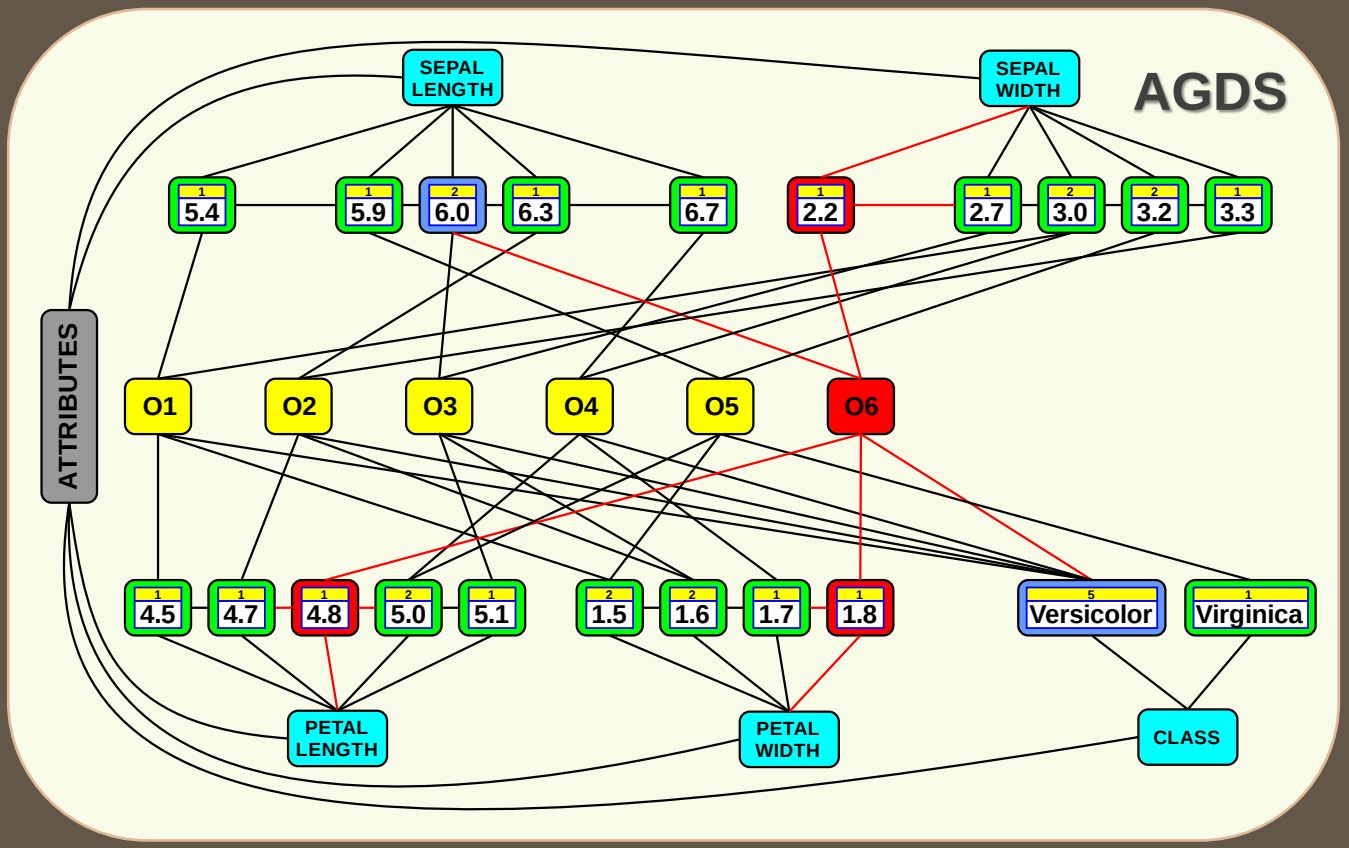
| DATASET | ATTRIBUTES | | | | |
|---|---|---|---|---|---|
| SAMPLE OBJECTS | SEPAL LENGTH | SEPAL WIDTH | PETAL LENGTH | PETAL WIDTH | CLASS LABEL |
| O1 | 5.4 | 3.0 | 4.5 | 1.5 | Versicolor |
| O2 | 6.3 | 3.3 | 4.7 | 1.6 | Versicolor |
| O3 | 6.0 | 2.7 | 5.1 | 1.6 | Versicolor |
| O4 | 6.7 | 3.0 | 5.0 | 1.7 | Versicolor |
| O5 | 6.0 | 2.2 | 5.0 | 1.5 | Virginica |
| O6 | 5.9 | 3.2 | 4.8 | 1.8 | Versicolor |
| O7 | 6.0 | 3.0 | 4.8 | 1.8 | Virginica |
| O8 | 5.7 | 2.5 | 5.0 | 2.0 | Virginica |
| O9 | 6.5 | 3.2 | 5.1 | 2.0 | Virginica |



# Transformacja Asocjacyjna

Im więcej obiektów dodamy do tej struktury, tym mniej nowych węzłów wartości jest dodawanych, gdy przekształcona tabela zawiera duplikaty. Wszystkie węzły obiektów połączone z wzajemnie połączonymi węzłami wartości z innymi węzłami obiektów automatycznie tworzą pośrednie powiązania między takimi obiektami.

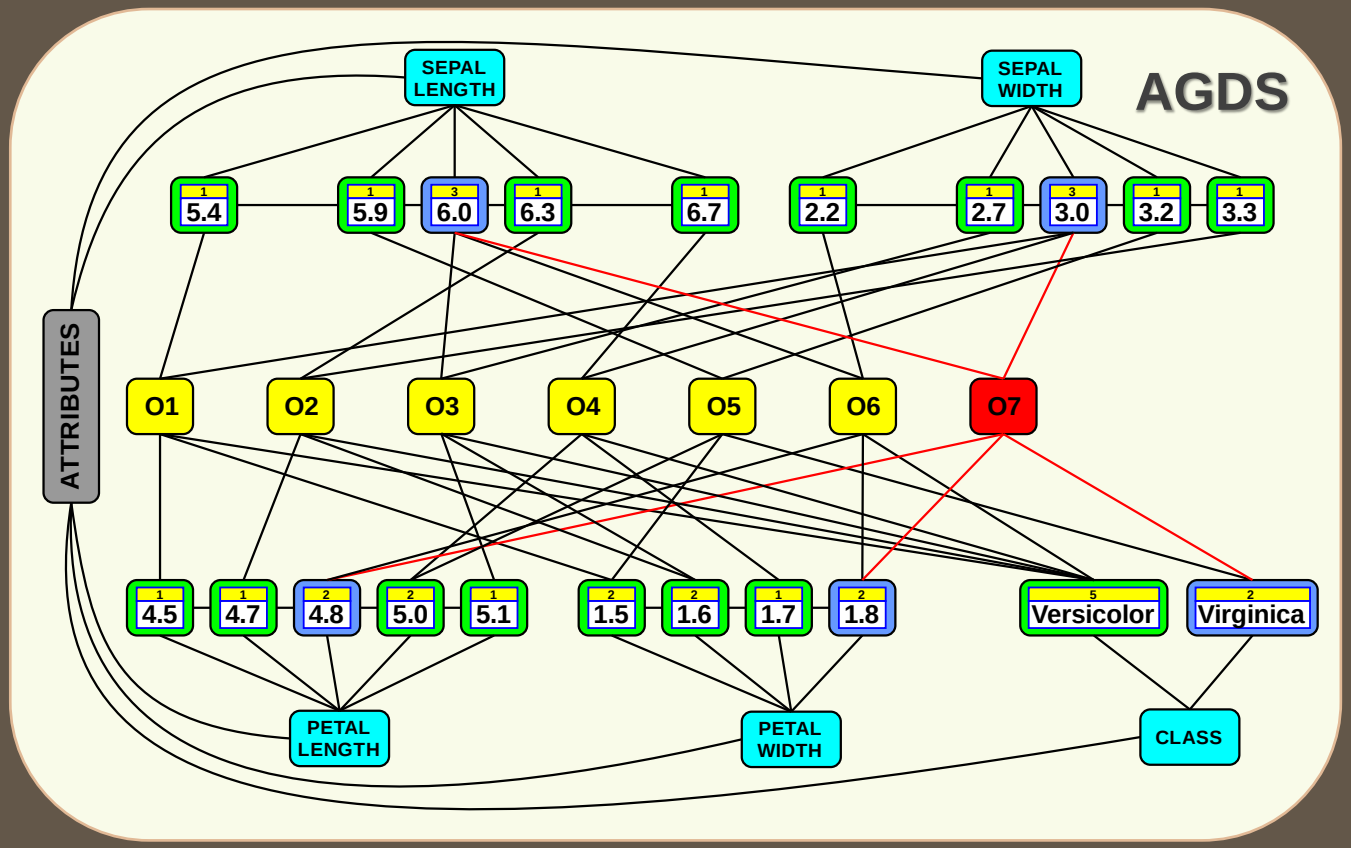
| DATASET | ATTRIBUTES | | | | |
|---|---|---|---|---|---|
| SAMPLE OBJECTS | SEPAL LENGTH | SEPAL WIDTH | PETAL LENGTH | PETAL WIDTH | CLASS LABEL |
| O1 | 5.4 | 3.0 | 4.5 | 1.5 | Versicolor |
| O2 | 6.3 | 3.3 | 4.7 | 1.6 | Versicolor |
| O3 | 6.0 | 2.7 | 5.1 | 1.6 | Versicolor |
| O4 | 6.7 | 3.0 | 5.0 | 1.7 | Versicolor |
| O5 | 6.0 | 2.2 | 5.0 | 1.5 | Virginica |
| O6 | 5.9 | 3.2 | 4.8 | 1.8 | Versicolor |
| O7 | 6.0 | 3.0 | 4.8 | 1.8 | Virginica |
| O8 | 5.7 | 2.5 | 5.0 | 2.0 | Virginica |
| O9 | 6.5 | 3.2 | 5.1 | 2.0 | Virginica |



# Transformacja Asocjacyjna

W takim przypadku obiekt O7 jest dodawany bez dodawania żadnych nowych węzłów wartości, ponieważ wszystkie zostały już dodane do tej struktury, więc dodawane są tylko nowe połączenia z istniejącymi węzłami, a ich liczniki reprezentowanych duplikatów są zwiększane. Uzyskujemy oszczędność pamięci.

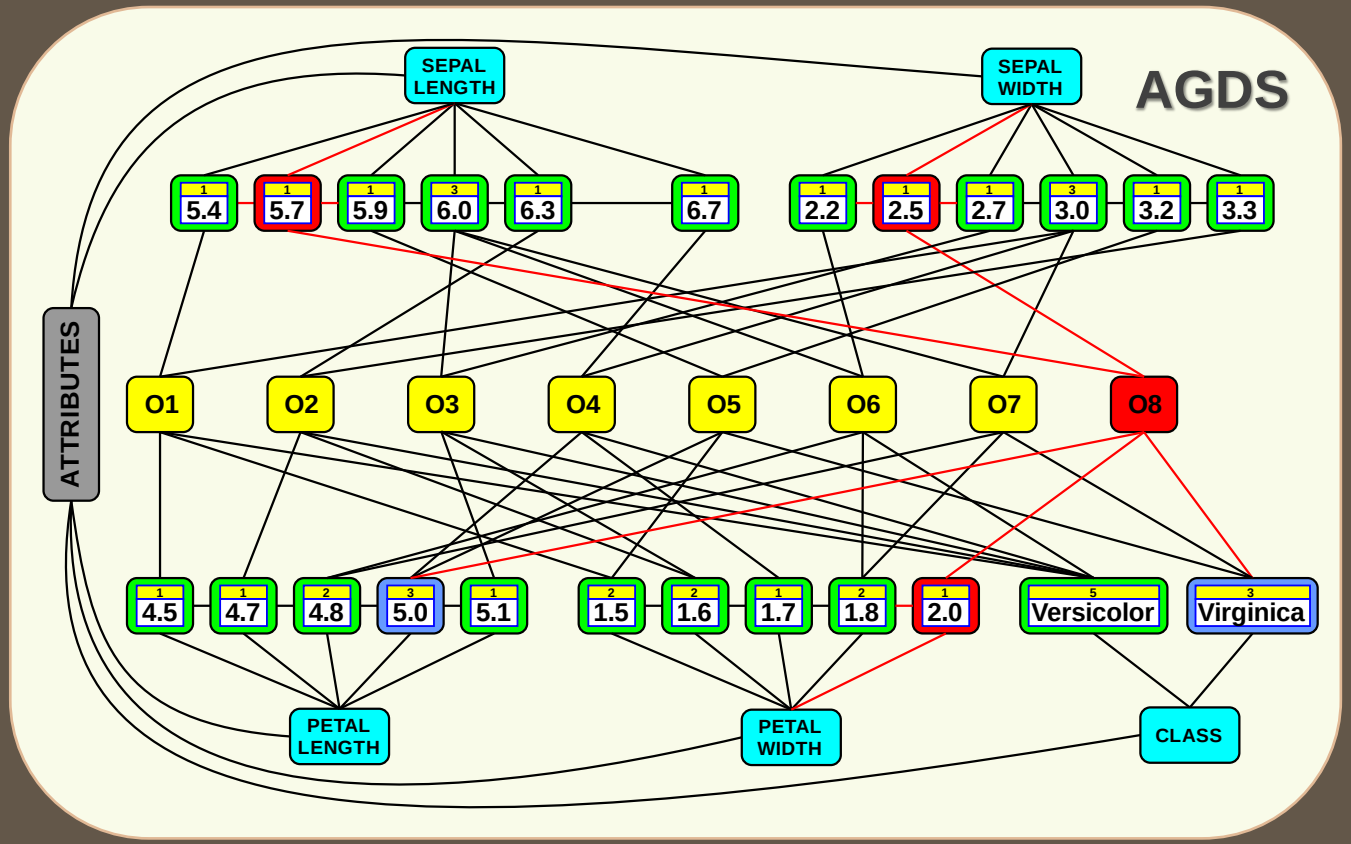
| DATASET | ATTRIBUTES | | | | |
|---|---|---|---|---|---|
| SAMPLE OBJECTS | SEPAL LENGTH | SEPAL WIDTH | PETAL LENGTH | PETAL WIDTH | CLASS LABEL |
| O1 | 5.4 | 3.0 | 4.5 | 1.5 | Versicolor |
| O2 | 6.3 | 3.3 | 4.7 | 1.6 | Versicolor |
| O3 | 6.0 | 2.7 | 5.1 | 1.6 | Versicolor |
| O4 | 6.7 | 3.0 | 5.0 | 1.7 | Versicolor |
| O5 | 6.0 | 2.2 | 5.0 | 1.5 | Virginica |
| O6 | 5.9 | 3.2 | 4.8 | 1.8 | Versicolor |
| O7 | 6.0 | 3.0 | 4.8 | 1.8 | Virginica |
| O8 | 5.7 | 2.5 | 5.0 | 2.0 | Virginica |
| O9 | 6.5 | 3.2 | 5.1 | 2.0 | Virginica |



# Transformacja Asocjacyjna

Obiekt O8 jest również podłączony do węzła wartości 5.0, który teraz definiuje pięć obiektów (O4, O5 i O8), więc istnieje widoczne podobieństwo między tymi obiektami. Podobieństwo między obiektami O5 i O8 jest większe niż między O4 i O8, ponieważ istnieje pierwsza wspólna cecja (Virginica) między pierwszą parą!

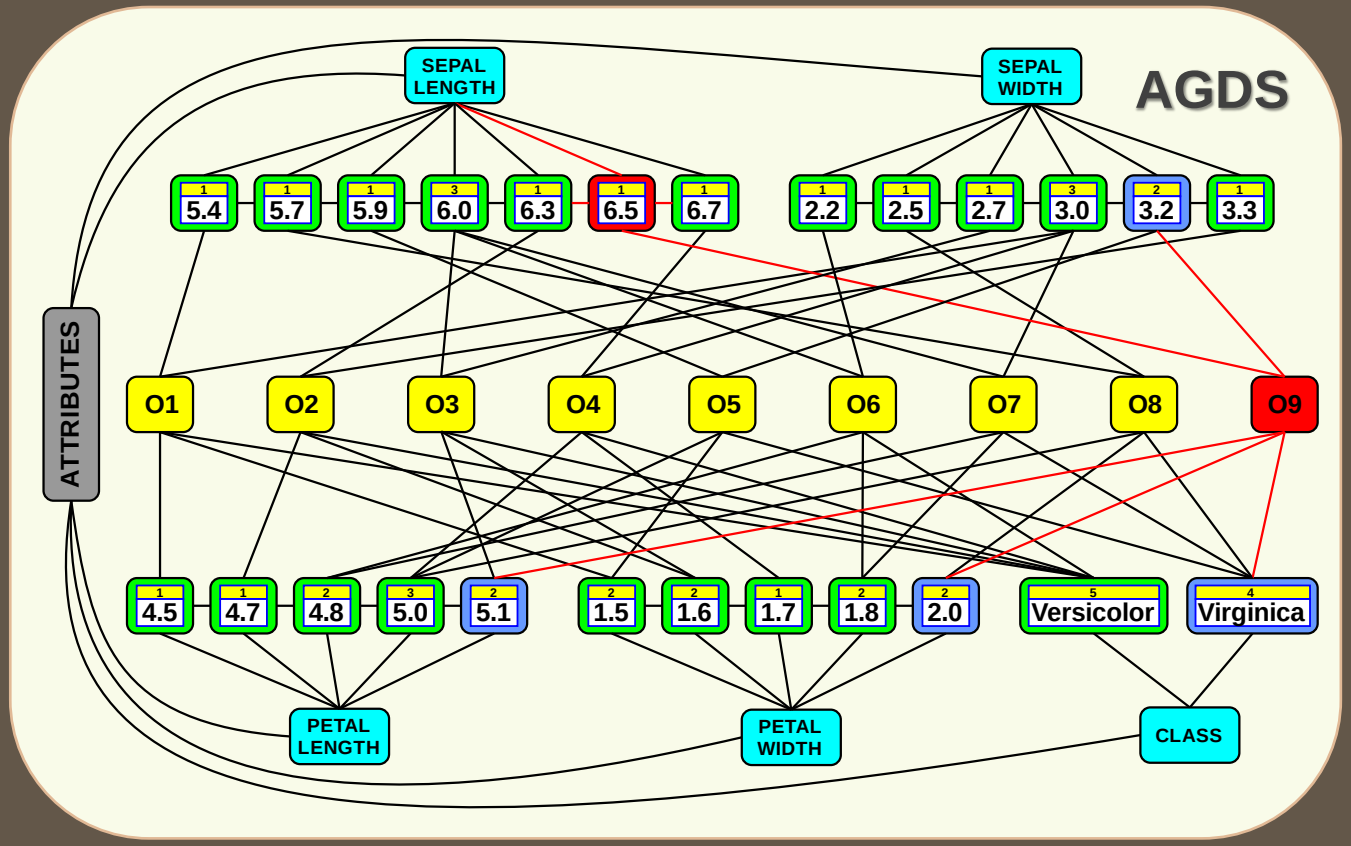
| DATASET | ATTRIBUTES | | | | |
|---|---|---|---|---|---|
| SAMPLE OBJECTS | SEPAL LENGTH | SEPAL WIDTH | PETAL LENGTH | PETAL WIDTH | CLASS LABEL |
| O1 | 5.4 | 3.0 | 4.5 | 1.5 | Versicolor |
| O2 | 6.3 | 3.3 | 4.7 | 1.6 | Versicolor |
| O3 | 6.0 | 2.7 | 5.1 | 1.6 | Versicolor |
| O4 | 6.7 | 3.0 | 5.0 | 1.7 | Versicolor |
| O5 | 6.0 | 2.2 | 5.0 | 1.5 | Virginica |
| O6 | 5.9 | 3.2 | 4.8 | 1.8 | Versicolor |
| O7 | 6.0 | 3.0 | 4.8 | 1.8 | Virginica |
| O8 | 5.7 | 2.5 | 5.0 | 2.0 | Virginica |
| O9 | 6.5 | 3.2 | 5.1 | 2.0 | Virginica |



# Transformacja Asocjacyjna

Obiekt O9 dodał tylko jedną nową cechę do tej struktury, ponieważ pozostałe wartości cech zostały już reprezentowane wcześniej.
Teraz proces transformacji tej małej tabeli jest już zakończony i możemy spróbować porównać te struktury i skorzystać z tego grafu asocjacyjnego!

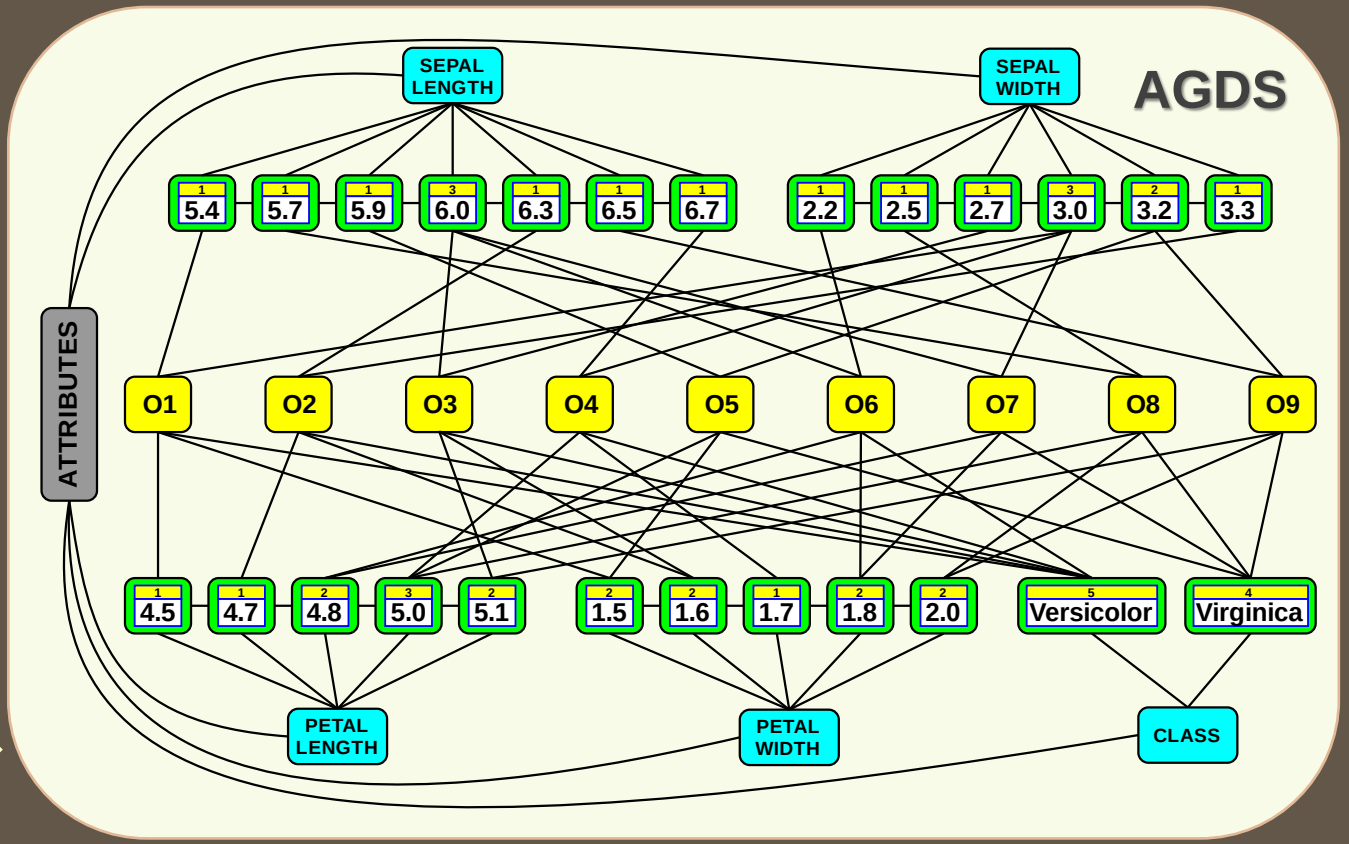
| DATASET | ATTRIBUTES | | | | |
|---|---|---|---|---|---|
| SAMPLE OBJECTS | SEPAL LENGTH | SEPAL WIDTH | PETAL LENGTH | PETAL WIDTH | CLASS LABEL |
| O1 | 5.4 | 3.0 | 4.5 | 1.5 | Versicolor |
| O2 | 6.3 | 3.3 | 4.7 | 1.6 | Versicolor |
| O3 | 6.0 | 2.7 | 5.1 | 1.6 | Versicolor |
| O4 | 6.7 | 3.0 | 5.0 | 1.7 | Versicolor |
| O5 | 6.0 | 2.2 | 5.0 | 1.5 | Virginica |
| O6 | 5.9 | 3.2 | 4.8 | 1.8 | Versicolor |
| O7 | 6.0 | 3.0 | 4.8 | 1.8 | Virginica |
| O8 | 5.7 | 2.5 | 5.0 | 2.0 | Virginica |
| O9 | 6.5 | 3.2 | 5.1 | 2.0 | Virginica |



# Porównanie struktur danych

Którą z dwóch przedstawionych struktur obecnie bardziej lubisz?
Struktura tabelaryczna reprezentuje dane i podstawowe relacje między nimi.
Struktura AGDS dodatkowo reprezentuje sąsiedztwo, porządek, podobieństwo, minima, maksima, liczbę duplikatów, liczbę unikalnych wartości i zakresy wszystkich funkcji.
Nie stracimy czasu na poszukiwanie takich relacji!

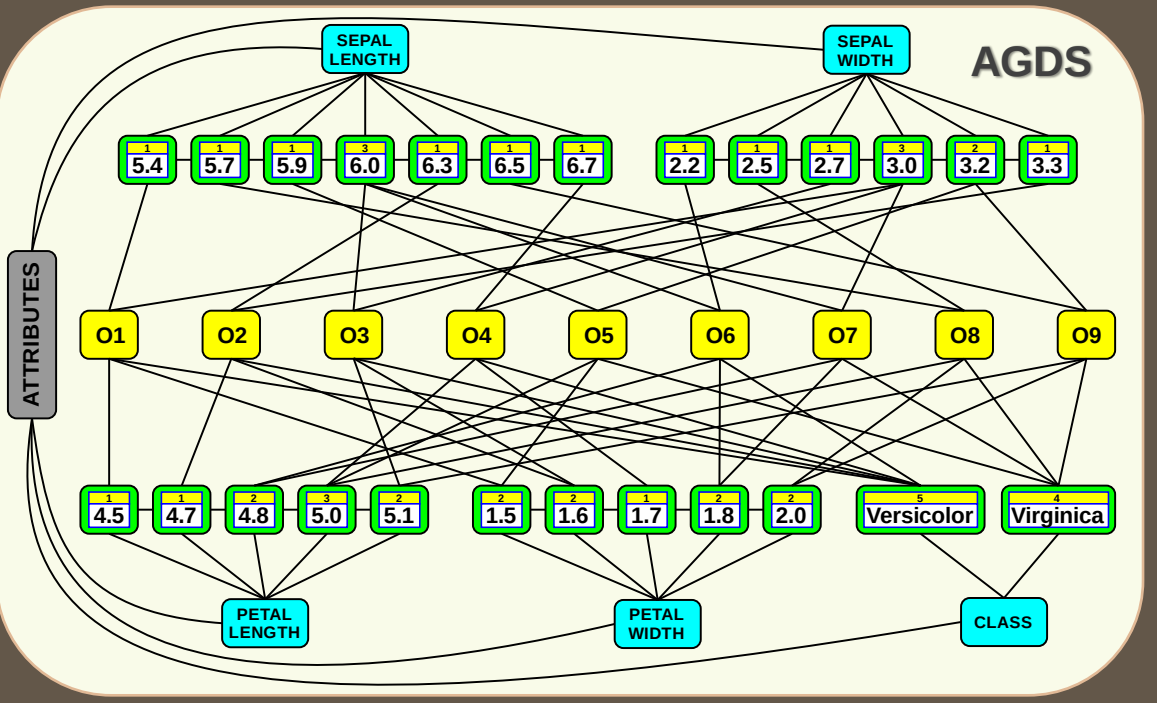
| DATASET | ATTRIBUTES | | | | |
|---|---|---|---|---|---|
| SAMPLE OBJECTS | SEPAL LENGTH | SEPAL WIDTH | PETAL LENGTH | PETAL WIDTH | CLASS LABEL |
| O1 | 5.4 | 3.0 | 4.5 | 1.5 | Versicolor |
| O2 | 6.3 | 3.3 | 4.7 | 1.6 | Versicolor |
| O3 | 6.0 | 2.7 | 5.1 | 1.6 | Versicolor |
| O4 | 6.7 | 3.0 | 5.0 | 1.7 | Versicolor |
| O5 | 6.0 | 2.2 | 5.0 | 1.5 | Virginica |
| O6 | 5.9 | 3.2 | 4.8 | 1.8 | Versicolor |
| O7 | 6.0 | 3.0 | 4.8 | 1.8 | Virginica |
| O8 | 5.7 | 2.5 | 5.0 | 2.0 | Virginica |
| O9 | 6.5 | 3.2 | 5.1 | 2.0 | Virginica |



# Alternative Construction of AGDS

Możemy stworzyć tę strukturę w alternatywny sposób, gdy zestaw danych (tabela) jest statyczny i nie zmienia się w czasie (tj. żadne rekordy nie są dodawane, usuwane ani aktualizowane).

| DATASET | ATTRIBUTES | | | | |
|---|---|---|---|---|---|
| SAMPLE OBJECTS | SEPAL LENGTH | SEPAL WIDTH | PETAL LENGTH | PETAL WIDTH | CLASS LABEL |
| O1 | 5.4 | 3.0 | 4.5 | 1.5 | Versicolor |
| O2 | 6.3 | 3.3 | 4.7 | 1.6 | Versicolor |
| O3 | 6.0 | 2.7 | 5.1 | 1.6 | Versicolor |
| O4 | 6.7 | 3.0 | 5.0 | 1.7 | Versicolor |
| O5 | 6.0 | 2.2 | 5.0 | 1.5 | Virginica |
| O6 | 5.9 | 3.2 | 4.8 | 1.8 | Versicolor |
| O7 | 6.0 | 3.0 | 4.8 | 1.8 | Virginica |
| O8 | 5.7 | 2.5 | 5.0 | 2.0 | Virginica |
| O9 | 6.5 | 3.2 | 5.1 | 2.0 | Virginica |




CAT OFF AND SEPARATE DATA FOR EACH ATTRIBUTE SEPARATELY

| DATASET | ATTRIBUTES | | | | |
|---|---|---|---|---|---|
| SAMPLE OBJECTS | SEPAL LENGTH | SEPAL WIDTH | PETAL LENGTH | PETAL WIDTH | CLASS LABEL |
| O1 | 5.4 | 3.0 | 4.5 | 1.5 | Versicolor |
| O2 | 6.3 | 3.3 | 4.7 | 1.6 | Versicolor |
| O3 | 6.0 | 2.7 | 5.1 | 1.6 | Versicolor |
| O4 | 6.7 | 3.0 | 5.0 | 1.7 | Versicolor |
| O5 | 6.0 | 2.2 | 5.0 | 1.5 | Virginica |
| O6 | 5.9 | 3.2 | 4.8 | 1.8 | Versicolor |
| O7 | 6.0 | 3.0 | 4.8 | 1.8 | Virginica |
| O8 | 5.7 | 2.5 | 5.0 | 2.0 | Virginica |
| O9 | 6.5 | 3.2 | 5.1 | 2.0 | Virginica |

SORT DATA FOR EACH ATTRIBUTE SEPARATELY

| DATASET | ATTRIBUTES | | | | |
|---|---|---|---|---|---|
| SAMPLE OBJECTS | SEPAL LENGTH | SEPAL WIDTH | PETAL LENGTH | PETAL WIDTH | CLASS LABEL |
| O1 | 5.4 | 2.2 | 4.5 | 1.5 | Versicolor |
| O2 | 5.7 | 2.5 | 4.7 | 1.5 | Versicolor |
| O3 | 5.9 | 2.7 | 4.8 | 1.6 | Versicolor |
| O4 | 6.0 | 3.0 | 4.8 | 1.6 | Versicolor |
| O5 | 6.0 | 3.0 | 5.0 | 1.7 | Versicolor |
| O6 | 6.0 | 3.0 | 5.0 | 1.8 | Virginica |
| O7 | 6.3 | 3.2 | 5.0 | 1.8 | Virginica |
| O8 | 6.5 | 3.2 | 5.1 | 2.0 | Virginica |
| O9 | 6.7 | 3.3 | 5.1 | 2.0 | Virginica |

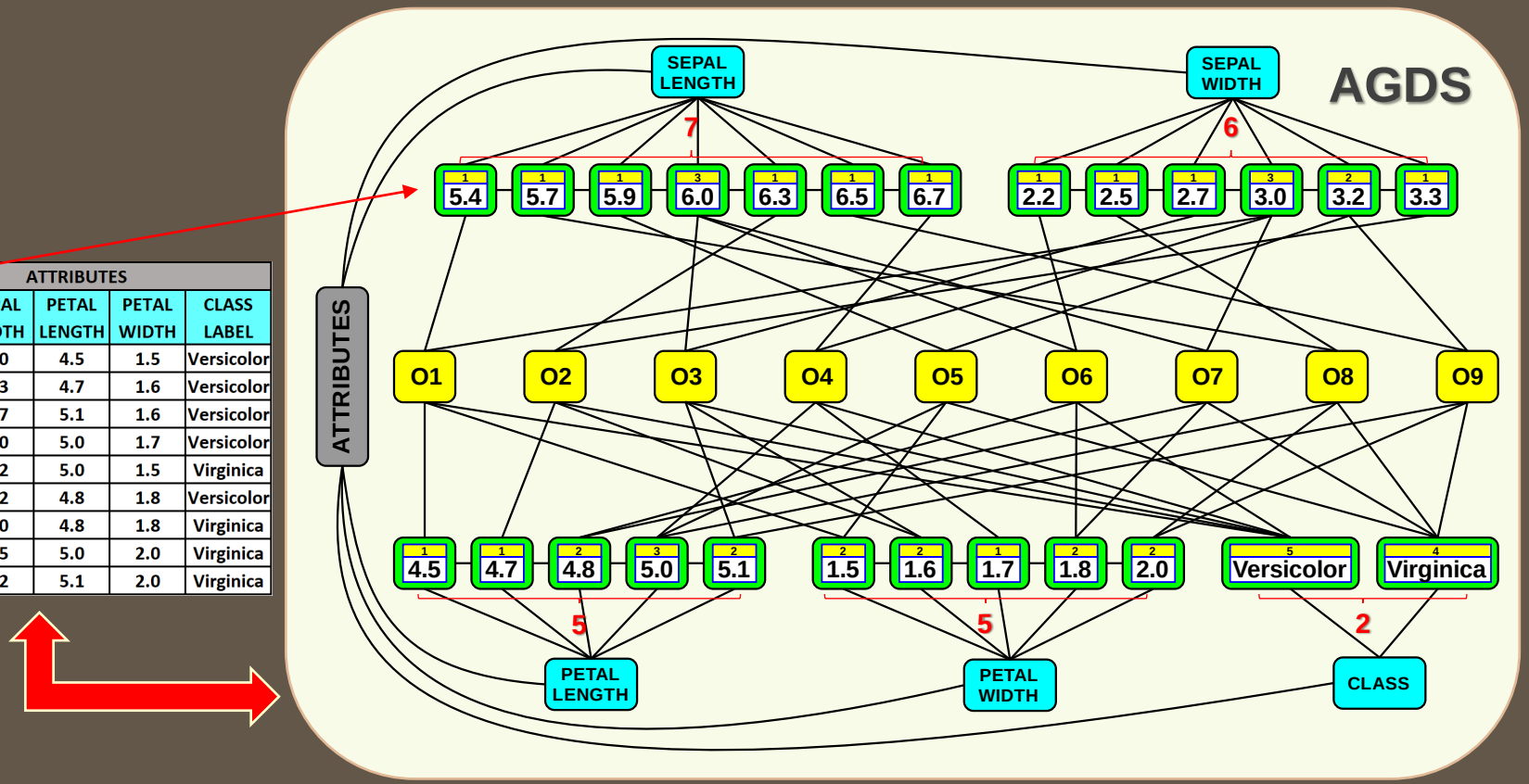
REMOVE DUPLICATES OF ALL ATTRIBUTES SEPARATELY

| DATASET | ATTRIBUTES | | | | |
|---|---|---|---|---|---|
| SAMPLE OBJECTS | SEPAL LENGTH | SEPAL WIDTH | PETAL LENGTH | PETAL WIDTH | CLASS LABEL |
| O1 | 5.4 | 2.2 | 4.5 | 1.5 | Versicolor |
| O2 | 5.7 | 2.5 | 4.7 | 1.6 | Virginica |
| O3 | 5.9 | 2.7 | 4.8 | 1.7 | |
| O4 | 6.0 | 3.0 | 5.0 | 1.8 | |
| O5 | 6.3 | 3.2 | 5.1 | 2.0 | |
| O6 | 6.5 | 3.3 | | | |
| O7 | 6.7 | | | | |
| O8 | | | | | |
| O9 | | | | | |

# Efektywność Dostępu do Danych

Funkcje każdego atrybutu można uporządkować za pomocą: posortowanych tabel, posortowanych list, tabel mieszania/haszujących lub AVB+drzew, aby zapewnić szybki dostęp do nich! Liczba unikalnych funkcji dla każdego atrybutu jest mniejsza lub równa liczbie wszystkich funkcji w tabeli.

| DATASET | ATTRIBUTES | | | | |
|---|---|---|---|---|---|
| SAMPLE OBJECTS | SEPAL LENGTH | SEPAL WIDTH | PETAL LENGTH | PETAL WIDTH | CLASS LABEL |
| O1 | 5.4 | 3.0 | 4.5 | 1.5 | Versicolor |
| O2 | 6.3 | 3.3 | 4.7 | 1.6 | Versicolor |
| O3 | 6.0 | 2.7 | 5.1 | 1.6 | Versicolor |
| O4 | 6.7 | 3.0 | 5.0 | 1.7 | Versicolor |
| O5 9 | 6.0 | 2.2 | 5.0 | 1.5 | Virginica |
| O6 | 5.9 | 3.2 | 4.8 | 1.8 | Versicolor |
| O7 | 6.0 | 3.0 | 4.8 | 1.8 | Virginica |
| O8 | 5.7 | 2.5 | 5.0 | 2.0 | Virginica |
| O9 | 6.5 | 3.2 | 5.1 | 2.0 | Virginica |



# Eksploracja Danych

**Eksploracja danych** zajmuje się wyszukiwaniem powtarzających się wzorców danych, tzw. **wzorców częstych (*frequent patterns*)**, w celu określenia ich przydatności lub wyprowadzenia wniosków.

**Wzorce danych** mogą mieć różną postać:

- zbiorów (wzorce proste),
- sekwencji (wzorce sekwencyjne),
- struktur złożonych, np. desenie, tekstury, mapy, obrazy 2D, 3D (wzorce strukturalne).

Czasami ciekawe mogą być również **wzorce rzadkie (*infrequent patterns*)** lub **unikalne (*unique patterns*)**, do wykrycia nowych i ciekawych zjawisk, np. przyrodniczych czy ekonomicznych.

**Eksplorację danych** możemy wykonywać:

- bezpośrednio (przeszukując dane),
- z wykorzystaniem modelu wiedzy.

# Informacja

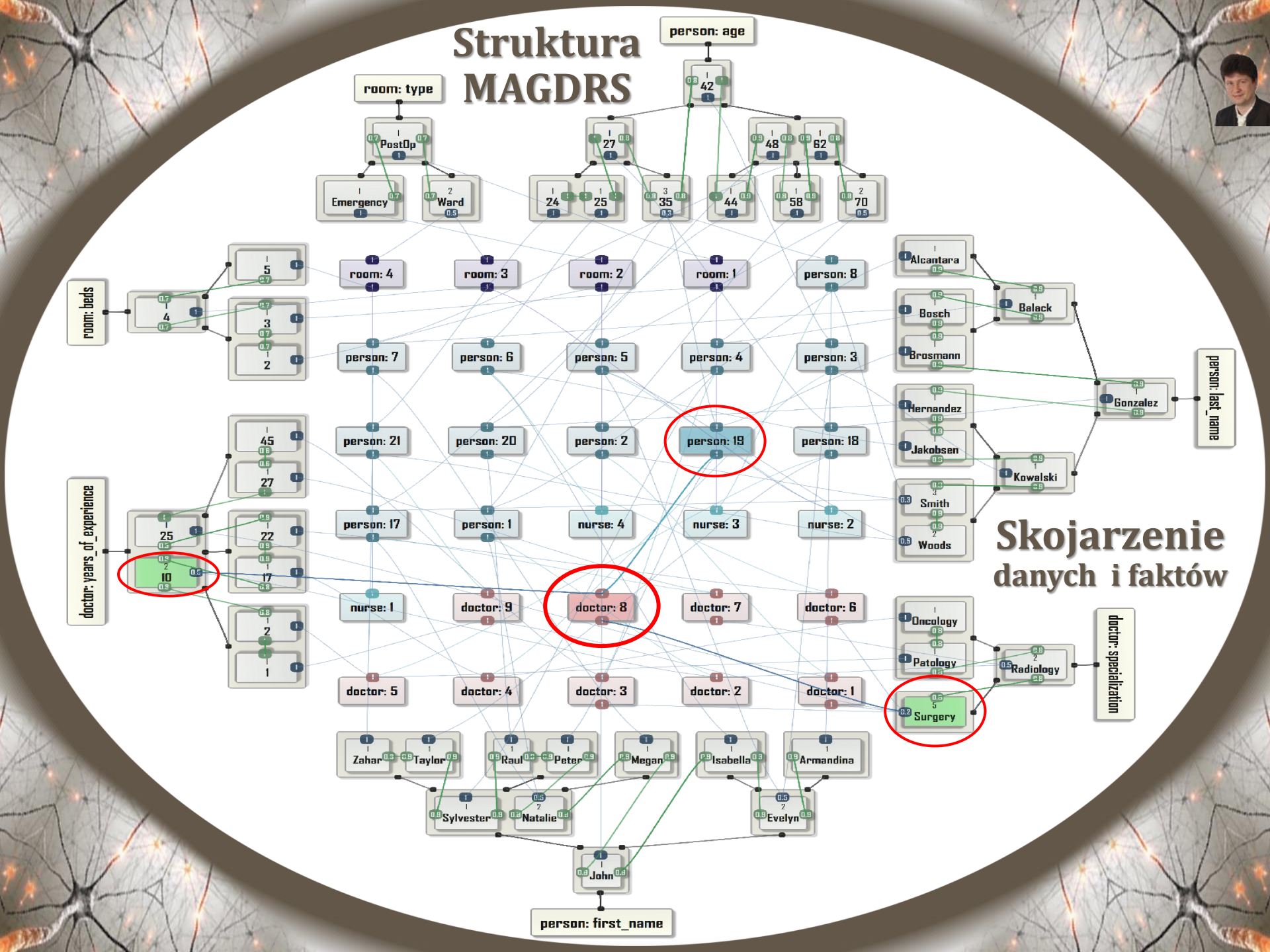
**Fakt** to zbiór uporządkowanych, spójnych i powiązanych danych.

**Informacja** to zbiór powiązanych danych (faktów) odbieranych przez odbiorcę, dla którego te dane mają określone znaczenie w kontekście posiadanej wiedzy, a odbiorca zmienia swój stan pod wpływem tych danych i/lub jego wiedza jest na ich podstawie aktualizowana lub poszerzana,

np. *normalna temperatura ciała człowieka wynosi 36,6 ºC*.

**Informacja** tworzy nowe lub modyfikuje istniejące powiązania pomiędzy znanymi obiektami oraz nowymi danymi.

**Odbiorca informacji** musi być w stanie powiązać te dane tak, aby zrozumieć przekazywaną informację.


Struktura
MAGDRS
person: age
room: type
PostOp
Emergency
Ward
42
27
48
62
24
25
35
44
58
70
room: beds
5
4
3
2
room: 4
room: 3
room: 2
room: 1
person: 8
person: 7
person: 6
person: 5
person: 4
person: 3
person: 21
person: 20
person: 2
person: 19
person: 18
person: 17
person: 1
nurse: 4
nurse: 3
nurse: 2
nurse: 1
doctor: 9
doctor: 8
doctor: 7
doctor: 6
doctor: 5
doctor: 4
doctor: 3
doctor: 2
doctor: 1
doctor: years_of_experience
45
27
25
22
10
17
7
1
Alcantara
Balack
Bosch
Brosmann
Gonzalez
Hernandez
Jakobsen
Kowalski
Smith
Woods
person: last_name
Skojarzenie
danych i faktów
Oncology
Patology
Radiology
Surgery
doctor: specialization
Zahar
Taylor
Raul
Peter
Megan
Isabella
Armandina
Sylvester
Natalie
Evelyn
John
person: first_name

# Poznanie

**Poznanie** jest działaniem mentalnym prowadzącym do pozyskiwania wiedzy z danych i ich rozumienia poprzez procesy myślowe, doświadczenie i zmysły.

**Poznanie** obejmuje wiele aspektów funkcji i procesów intelektualnych, tj. uwaga, kształtowanie wiedzy, pamięć, ocena i ewaluacja, rozumowanie, rozwiązywanie problemów i podejmowanie decyzji, rozumienie, przetwarzanie i posługiwanie się językiem.

**Procesy kognitywne** wykorzystują istniejącą wiedzę i generują nową wiedzę dla przetwarzanych danych.

# CZYM JEST WIEDZA?

**Dane bombardują nas z każdej strony!**

**A nasze mózgi jakoś sobie z nimi radzą!**

# Wiedza

**Wiedza** jest abstrakcyjnym rezultatem kontekstowej, skojarzeniowej konsolidacji i reprezentacji wzorców, faktów i reguł oraz ich uogólniania, tworzącym nowe metody, reguły i algorytmy przetwarzania danych oraz wnioskowania.

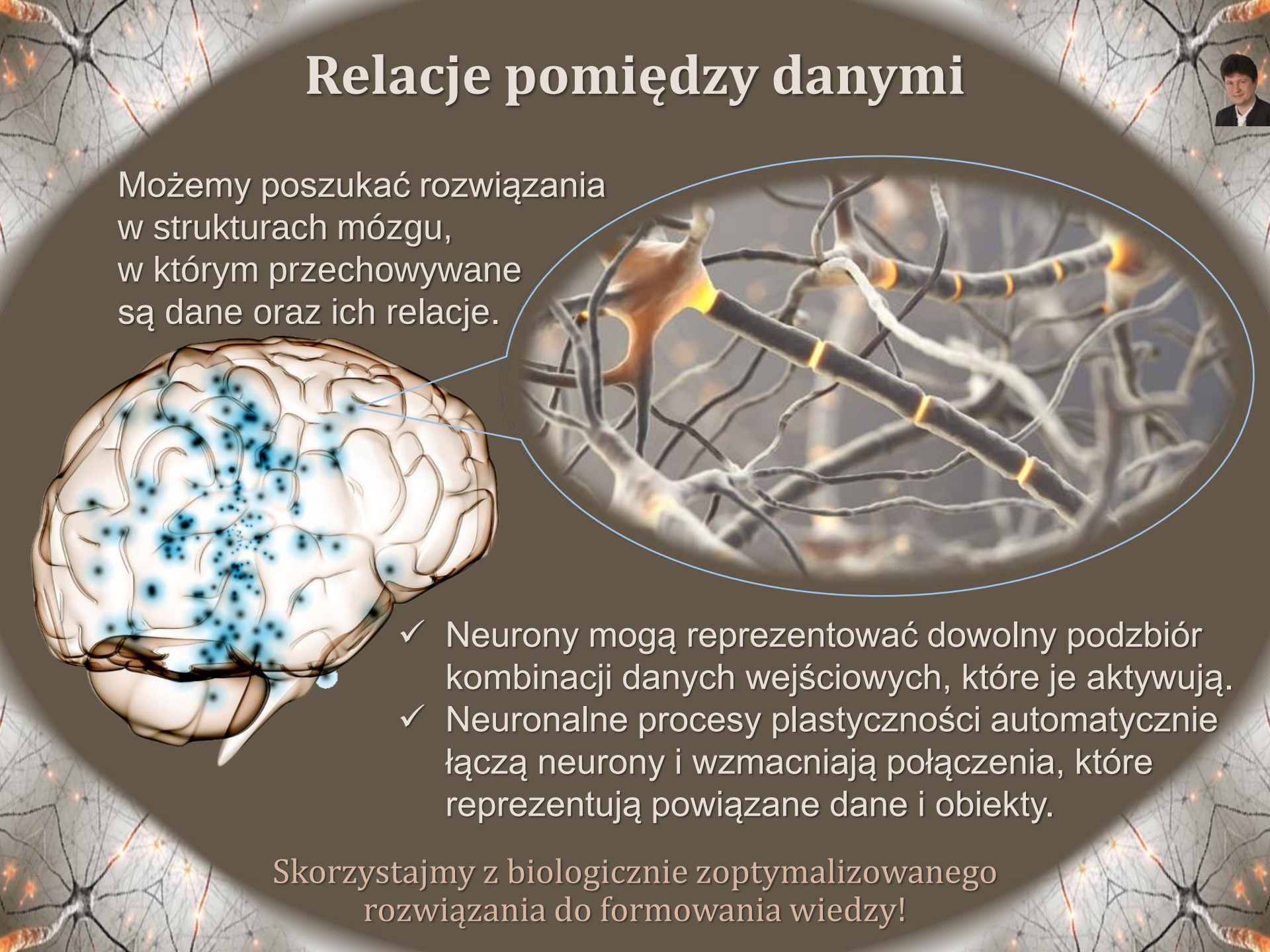
W informatyce może być postrzegana jako zbiór informacji z powiązanym kontekstem, który występuje w formie relacji między różnymi, zebranymi w czasie informacjami.

**Wiedza** jest ściśle związana z inteligencją, ponieważ umożliwia wnioskowanie i rozwój indywidualnej inteligencji, jak również wyodrębnienie własnego bytu i jestestwa.

# Wiedza

**Wiedza formuje się w naszym umyśle**
i powstaje jako skutek aktywności skojarzeniowej zachodzącej w mózgu, który jest również siedliskiem naszej inteligencji.

# Relacje pomiędzy danymi



Możemy poszukać rozwiązania w strukturach mózgu, w którym przechowywane są dane oraz ich relacje.

- ✓ Neurony mogą reprezentować dowolny podzbiór kombinacji danych wejściowych, które je aktywują.
- ✓ Neuronalne procesy plastyczności automatycznie łączą neurony i wzmacniają połączenia, które reprezentują powiązane dane i obiekty.

Skorzystajmy z biologicznie zoptymalizowanego rozwiązania do formowania wiedzy!

# Wiedza

**Wiedza** tworzy się w wyniku:

- aktywnej nauki (teoretycznej),
- zdobywania doświadczenia (praktycznego),
- introspekcji i rozważań (wewnętrznych),
- intuicji (pasywnej i nieświadomej formy).

**Wiedza** przekazywana jest poprzez:

- wzorce oraz zbiory faktów i reguł, które nazywamy danymi uczącymi (*training data*).

**Model wiedzy** powstaje poprzez:

- skojarzeniowe i kontekstowe powiązanie ze sobą wzorców, faktów i reguł (danych uczących), pozwalających na uogólnianie i wyprowadzanie wniosków oraz tworzenie nowych reguł i algorytmów;
- wykorzystanie aktywnego mechanizmu wnioskowania wykorzystującego utrwalone relacje pomiędzy danymi i obiektami.

# Czym nie jest wiedza!

**Wiedza** nie jest żadną:

~ kolekcją informacji, faktów, wzorców, danych ani reguł,

~ formą pamięci,

~ strukturą danych.

**Wiedza** nie zapewnia:

~ dokładnego zapamiętania wszystkich danych, wzorców, faktów ani reguł, lecz je konsoliduje, wiąże i uogólnia.

**Wiedza** nie skupia się:

~ na danych lecz na łączących je relacjach, na podstawie których umożliwia wnioskowanie.

# MONKEY

**poniżej przykład zestawu faktów opisujących tę małpkę:**

*”I have a monkey. My monkey is very small.*
*It is very lovely. It likes to sit on my head.*
*It can jump very quickly. It is also very clever.*
*It learns quickly. My monkey is lovely.*
*I have also a small dog.”*

***Jaką wiedzę na podstawie opisu zgromadziliśmy na temat tej małpki?***
***Spróbujmy teraz odpowiedzieć na pytanie: What is this monkey like?***

# GRAF WIEDZY

## Konstrukcja asocjacyjnego grafu neuronowego dla następującego zestawu wzorców sekwencyjnych:

1x S1 I HAVE A MONKEY

1x S2 MY MONKEY IS VERY SMALL

1x S3 IT IS VERY LOVELY

1x S4 IT LIKES TO SIT ON MY HEAD

1x S5 IT CAN JUMP VERY QUICKLY

1x S6 IT IS ALSO VERY CLEVER

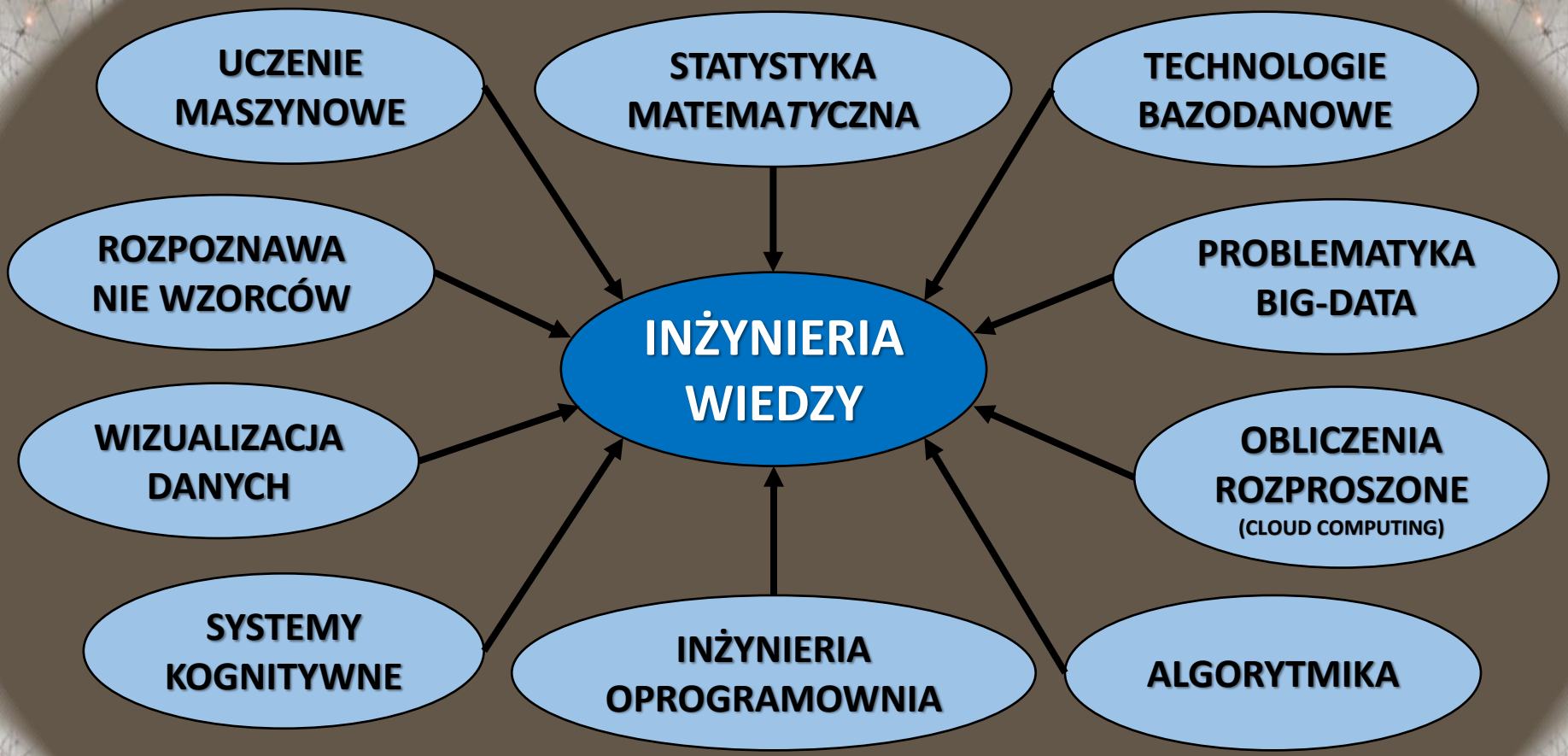
1x S7 IT LEARNS QUICKLY

1x S8 MY MONKEY IS LOVELY

1x S9 I HAVE ALSO A SMALL DOG

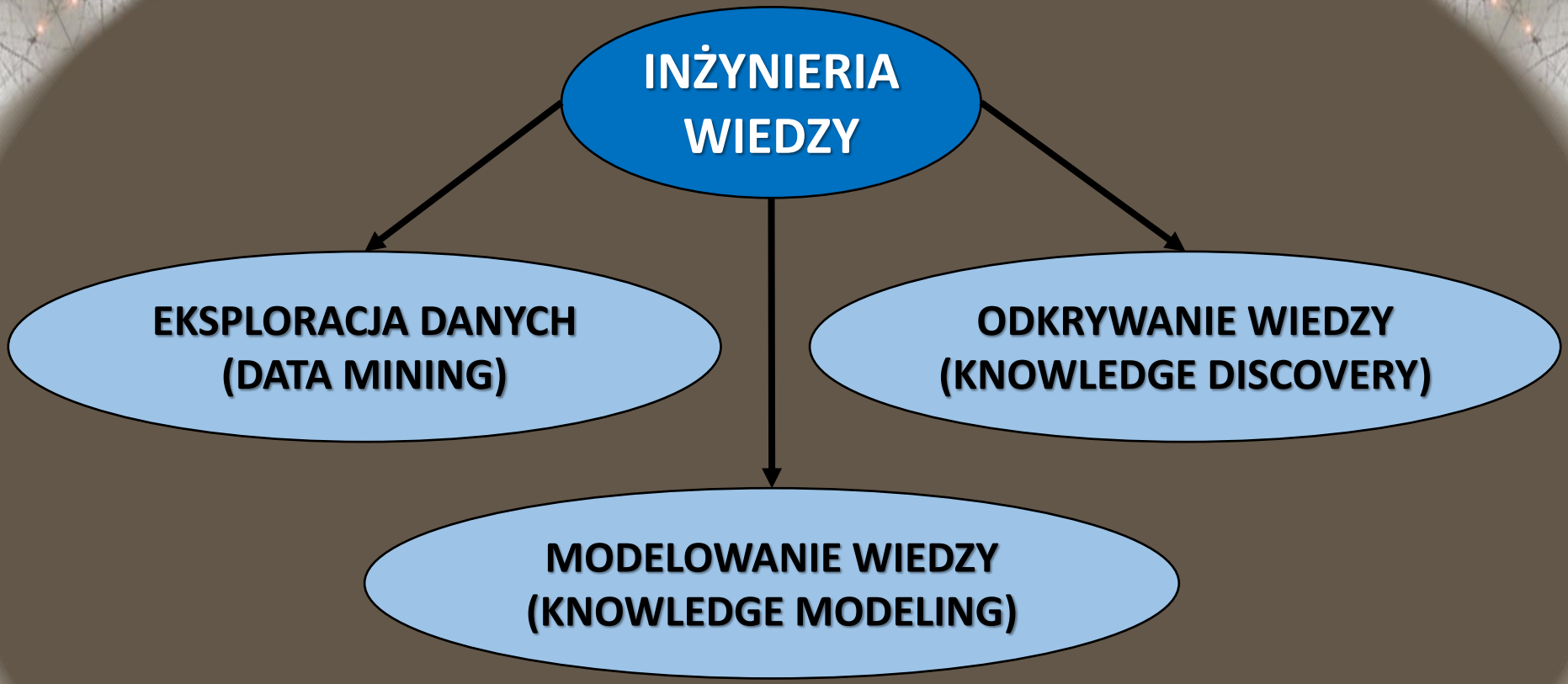
Open Data
Parameters
Learn
Learn Monkey
Learn Slower
2
Learn Faster
Balance Graph
View -
100
View +
Ask
Ask Monkey
Ask Faster
5
Ask Slower
Weights
Background
Exit
Ask times:
1
INPUT:
(Enter a new sentence)
SIMTIME:
0
OUTPUT:
(Results will be brought back by the ANAKG after asking)
Google: Adrian Horzyk (http://home.agh.edu.pl/~horzyk/index-eng.php) to cooperate in developing associative knowledge-based artificial intelligence solutions: horzyk@agh.edu.pl

# Inżynieria Wiedzy



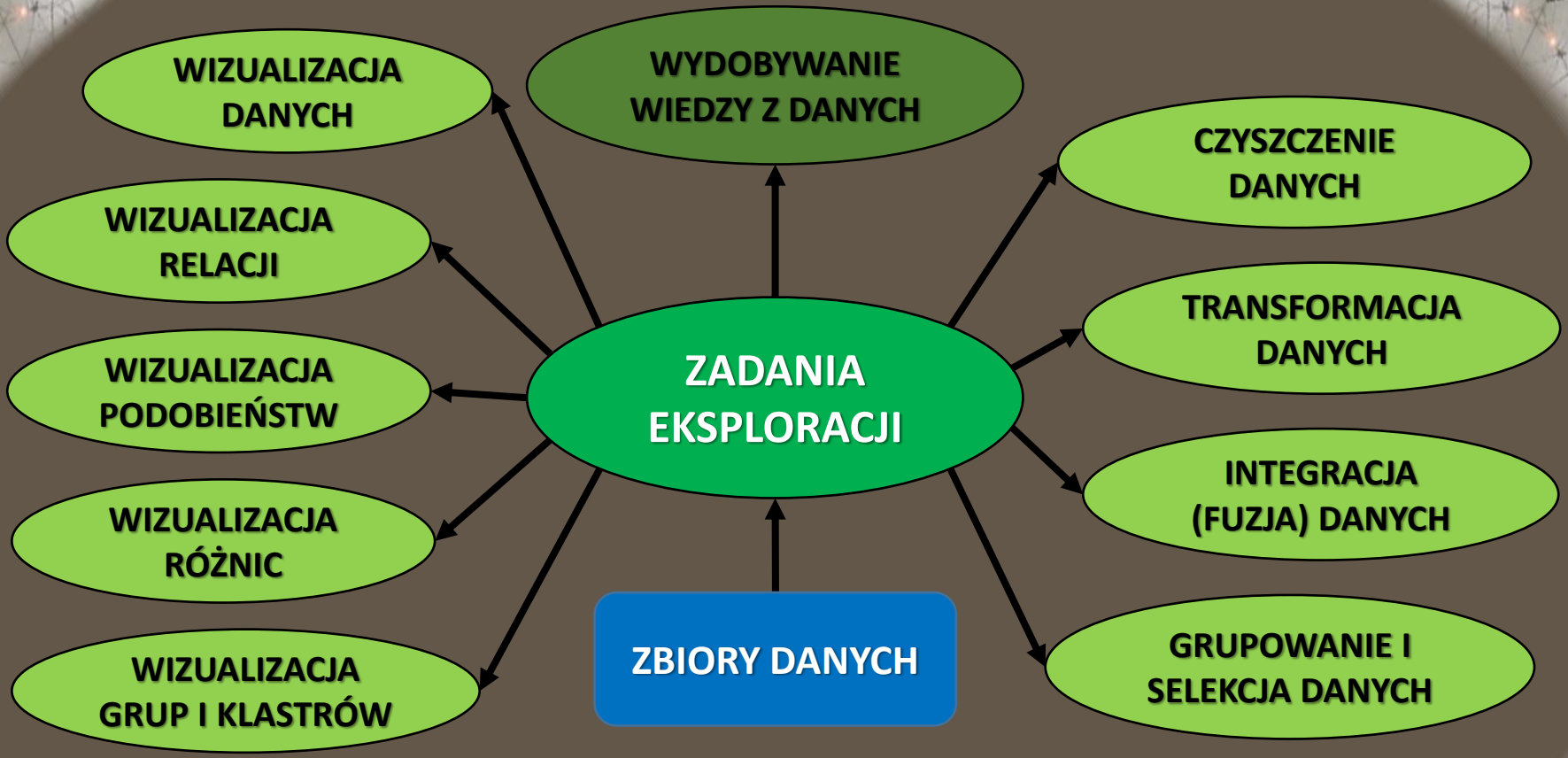

**Inżynieria wiedzy** to obszar informatyki zajmujący się metodami eksploracji, reprezentacji i modelowania wiedzy z danych (ich zbiorów, reguł, baz danych) oraz metodami wnioskowania na ich podstawie.

# Odkrywanie Wiedzy



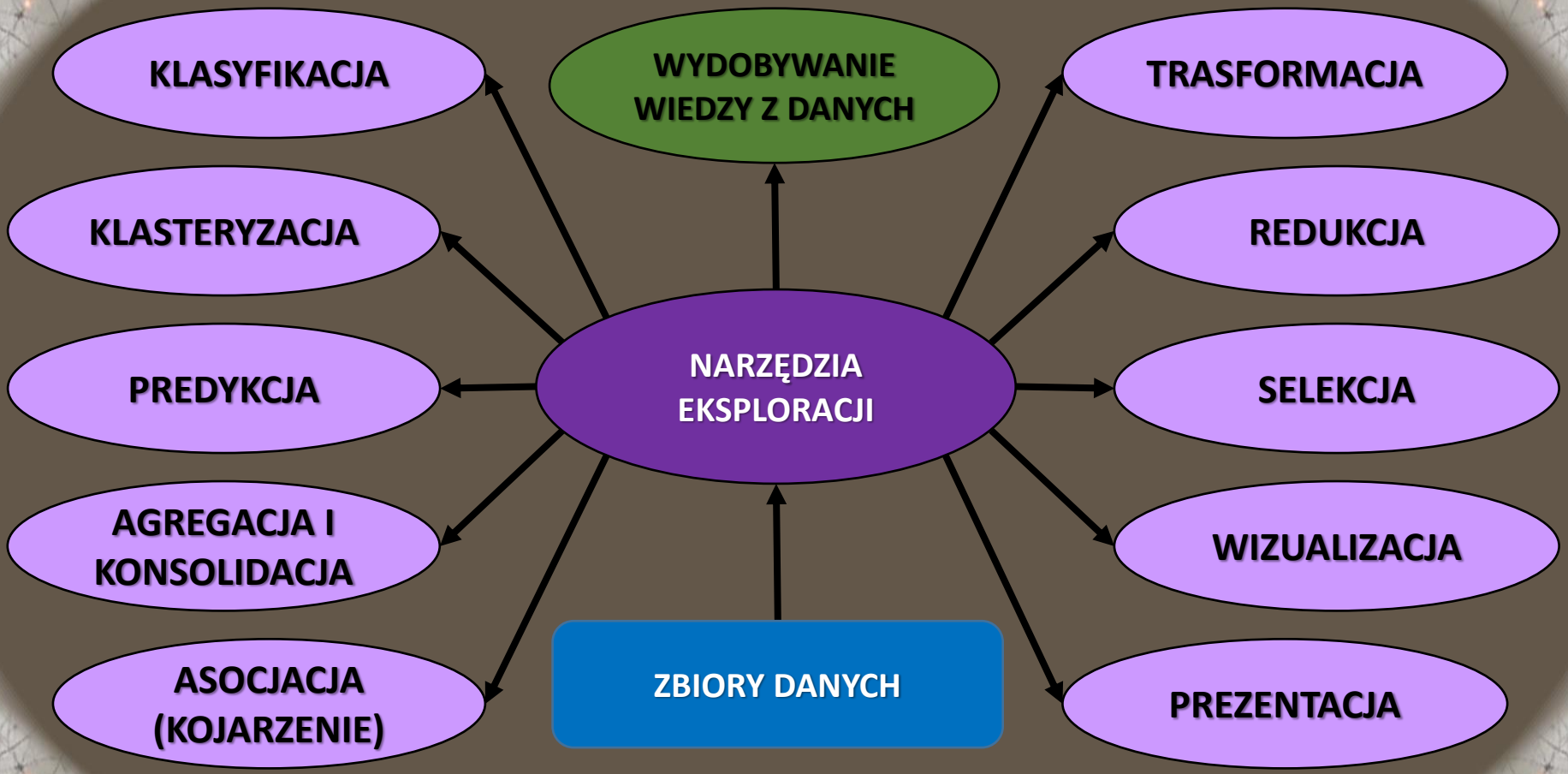

**Odkrywanie wiedzy z (baz) danych** to proces odkrywania wiedzy ukrytej w danych lub ich zbiorach (czyli bazach danych) polegający na wyszukiwaniu prawidłowości, powtarzalności, podobieństw i zależności (relacji) pomiędzy danymi.

# Zadania Eksploracji




**Eksploracja danych** to zwykle proces wieloetapowy związany z wstępną obróbką danych (czyszczenie, normalizacja, standaryzacja lub inny rodzaj transformacji), porównywaniem, integracją, grupowaniem i selekcją danych oraz wizualizacją danych, ich cech, grup, podobieństw, różnic i zależności (relacji).

# Narzędzia Eksploracji




**Narzędzia eksploracji danych** to metody i algorytmy informatyczne służące do realizacji celów eksploracji wiedzy z danych.

# ANALIZA I EKSPLORACJA DANYCH

- ✓ Większość repozytoriów zawiera ukryte informacje (mogące wzbogacić naszą wiedzę) w postaci regularności, korelacji, podobieństw, trendów, osobliwości, reguł..., lecz ich struktura i rozmiary uniemożliwiają ich „ręczną” analizę.
- ✓ W ostatnich latach powstało wiele ciekawych i efektywnych algorytmów, struktur i metod półautomatycznej lub automatycznej eksploracji danych (*data miningu*), zwanych też odkrywaniem wiedzy (*knowledge discovery*).
- ✓ Eksploracja danych (*data mining*) to skrót myślowy oznaczający eksplorację wiedzy z danych, a ściślej mówiąc wydobywanie informacji z danych, które tą wiedzę mogą ukształtować.
- ✓ Algorytmy te służą również do odkrywania związków (relacji) pomiędzy elementami lub grupami obiektów, które zapisywane są w postaci tzw. reguł asocjacyjnych.

# EKSPLORACJA WIEDZY Z DANYCH

- ✓ **Eksploracja danych** w zależności od:
  - rodzaju danych
  - sposobu przechowywania danych

  może być przeprowadzana za pośrednictwem:
  metod statystycznych, reguł, drzew lub diagramów decyzyjnych, badania zawierania, podzbiorów, bliskości lub podobieństw, wyszukiwania wzorców częstych lub rzadkich, systemów rozmytych,
  sieci neuronowych, metod skojarzeniowych itd.
- ✓ **Dane** mogą tworzyć charakterystyczne **wzorce** w postaci:
  - **Zbiorów** (np. encji, wektorów, macierzy),
  - **Sekwencji** (np. tekstów, ciągów instrukcji, sekwencji czasowych),
  - **Struktur złożonych** (np. podgrafów, obrazów, map, tekstur).

# ZBIORY ELEMENTÓW I WSPARCIE

- ✓ Zbiór elementów (*itemset*) $I = \{i_1, i_2, ..., i_N\}$ – to zbiór wszystkich dostępnych elementów (obiektów, towarów), gdzie $N \geq 1$.
- ✓ Transakcja (*transaction*) T jest parą T = (id, X) składającą się z identyfikatora transakcji id oraz pewnego podzbioru towarów $X \subseteq I$, zakładając pewną skończoną ilość identyfikatorów transakcji $id \in Tid = \{id_1, id_2, ..., id_M\}$.
- ✓ Z punktu widzenia eksploracji danych interesuje nas częstość (*frequency*) wzorców (*patterns*) W, czyli powtarzalność różnych k-elementowych podzbiorów (k-zbiorów) w zbiorze transakcji nazywanym bazą transakcyjną $D = \{T_1, T_2, ..., T_M\}$.
- ✓ k-zbiór (*k-itemset*) to k-elementowy zbiór $X = \{x_1, ..., x_K\} \subseteq I$, który zwykle definiuje pewną transakcję $T_m = (id_m, X)$ lub wzorzec $W \subseteq I$, np.:
  $W=\{kawa, cukier\} \subseteq X=\{kawa, cukier, jajka\} \subseteq I$,
  $W=\{kawa, cukier\} \subseteq I=\{kawa, mleko, cukier, orzeszki, jajka, chleb$, masło, $miód\}$.
- ✓ Mówimy, że transakcja T = (id, X) pokrywa wzorzec (np. zbiór towarów) W, jeśli $W \subseteq X$. Wzorzec W może być pokryty przez wiele transakcji. Zbiór transakcji pokrywających wzorzec W oznaczamy jako cover(W,D) = $\{T \in D: T$ pokrywa $W\}$.
- ✓ Mówimy, że wzorzec W jest częsty (*frequent*), jeśli jego pokrycie przez transakcje z rozważanej bazy transakcji D jest nie mniejsze niż ustalony próg $\sigma$.
- ✓ Wydobywanie częstych wzorców (mining frequent patterns) jest jednym podstawowych zadań eksploracji danych i niezbędnym krokiem w wydobywaniu reguł asocjacyjnych (Association Rule Mining) i analizie korelacji danych.

# ZBIORY ELEMENTÓW I WSPARCIE

- ✓ **Wsparcie (*support*) s** – to częstotliwość wystąpień **wzorca W** (zbioru elementów X) w analizowanym zbiorze encji lub transakcji wyrażone w procentach. Wsparcie liczone jest jako stosunek ilości wystąpień **wzorca W** w rozważanym zbiorze transakcji D, wyrażonej jako |cover(W,D)|, w stosunku do ilości wszystkich rozważanych transakcji M = |D|:

  $s = |cover(W,D)| \ / \ M$

- ✓ Korzystając z definicji wsparcia, mówimy, że **wzorzec W** jest **częsty (*frequent*)**, jeśli jego wsparcie (*support*) jest nie mniejsze niż ustalony próg $\sigma$ (*min support*): $\sigma = s_{min}$ (*minimum support*).

**PRZYKŁAD: Dla progu $\sigma$ = 50% i zbioru transakcji określ, które elementy są częste (*frequent*)?**

**CZĘSTE > 50%**

- ✓ Cukier (80%)
- ✓ Kawa (60%)
- ✓ Jajka (60%)
- ✓ Mleko (40%)
- ✓ Orzeszki (40%)
- ✓ Masło (40%)
- ✓ Chleb (20%)
- ✓ Miód (20%)

| ID TRANSAKCJI | ELEMENTY TRANSAKCJI |
|---|---|
| 1 | kawa, mleko, cukier, orzeszki |
| 2 | kawa, cukier, jajka |
| 3 | kawa, chleb, cukier, masło |
| 4 | orzeszki, cukier, miód, jajka |
| 5 | masło, mleko, jajka |

# REGUŁY ASOCJACYJNE

- ✓ **Reguły asocjacyjne (*association rules*) elementów transakcji/wzorców: X → Y (s, c).**
- ✓ **Wsparcie (*support*) s dla reguł asocjacyjnych zdefiniowane jest przez prawdopodobieństwo, że określona transakcja zawiera zarówno X jak również Y, czyli X ∪ Y. Prawdopodobieństwo to jest liczone względem wszystkich możliwych transakcji, wyrażając prawdopodobieństwo zaistnienia takiej asocjacji, czyli wystąpienia takiej reguły asocjacyjnej.**
- ✓ **Zaufanie/Pewność/Wiarygodność (*confidence*) c – to prawdopodobieństwo warunkowe p(Y|X), że transakcja zawierająca X zawiera również Y.**
- ✓ **Eksploracja reguł asocjacyjnych polega na odnalezieniu wszystkich reguł X → Y o określonym minimalnym wsparciu $s_{min}$ oraz o określonej minimalnej pewności $c_{min}$ : np. $s \geq s_{min} = 40\%$ oraz $c \geq c_{min} = 50\%$. Wtedy taką regułę asocjacyjną nazywamy silną.**
- ✓ **Wielowymiarowe reguły asocjacyjne zawierają rozbudowane reguły tj.:**
  **wiek (X, „18-24”) ∧ zawód (X, „student”) ⇒ kupuje (X, „cola”)**
  **wiek (X, „18-24”) ∧ kupuje (X, „pop-corn”) ⇒ kupuje (X, „cola”)**

# REGUŁY ASOCJACYJNE

**Eksploracja reguł asocjacyjnych** oznaczają wyszukiwanie takich reguł, które przewidują wystąpienie elementu na podstawie wystąpienia innych elementów w transakcji.

**Reguły asocjacyjne (*association rules*)** stosowane są np. do:

- ✓ organizowania promocji i sprzedaży wiązanej (up-selling i cross-selling),
- ✓ konstruowania katalogów wysyłkowych,
- ✓ ustalenia sposobu rozmieszczenia towarów na półkach w marketach,
- ✓ określania najlepiej rotujących się towarów.

**REGUŁY ASOCJACYJNE:**

- ✓ Kawa → Cukier (60%, 100%)
- ✓ Cukier → Kawa (60%, 75%)
- ✓ Cukier → Jajka (40%, 50%)
- ✓ Jajka → Cukier (40%, 67%)

**NIE SĄ NIMI dla s ≥ 50%, c ≥ 50%:**

- ❖ Kawa → Jajka (20%, 33%)
- ❖ Jajka → Kawa (20%, 33%)

| ID TRANSAKCJI | ELEMENTY TRANSAKCJI |
|---|---|
| 1 | kawa, mleko, cukier, orzeszki |
| 2 | kawa, cukier, jajka |
| 3 | kawa, chleb, cukier, masło |
| 4 | orzeszki, cukier, miód, jajka |
| 5 | masło, mleko, jajka |

# OKREŚLANIE REGUŁ ASOCJACYJNYCH

- ✓ **Wsparcie (*support*) s to ilość** transakcji zawierających zarówno X jak również Y, czyli $X \cup Y$.

$$s\ (A \Rightarrow B, D) = \frac{|\text{cover}(A \cup B\ ,\ D)|}{|D|}$$

- ✓ **Zaufanie/Pewność/Wiarygodność (*confidence*) c** – to prawdopodobieństwo warunkowe p(Y|X), że transakcja zawierająca X zawiera również Y.

$$c\ (A \Rightarrow B, D) = \frac{s\ (A \cup B\ ,\ D)}{s\ (A\ ,\ D)}$$

**Sposób obliczania wsparcia (*support*) s i pewności (*confidence*) c dla reguł asocjacyjnych określonych, dla których s ≥ 40%, c ≥ 50%:**

| ID TRANSAKCJI | ELEMENTY TRANSAKCJI |
|---|---|
| 1 | kawa, mleko, cukier, orzeszki |
| 2 | kawa, cukier, jajka |
| 3 | kawa, chleb, cukier, masło |
| 4 | orzeszki, cukier, miód, jajka |
| 5 | masło, mleko, jajka |

| ID TRANSAKCJI | ELEMENTY TRANSAKCJI |
|---|---|
| 1 | kawa, mleko, cukier, orzeszki |
| 2 | kawa, cukier, jajka |
| 3 | kawa, chleb, cukier, masło |
| 4 | orzeszki, cukier, miód, jajka |
| 5 | masło, mleko, jajka |

Kawa → Cukier (s = 60% = 3 / 5, c = 100% = 3 / 3)

Cukier → Kawa (s = 60% = 3 / 5, c = 75% = 3 / 4)

Cukier → Jajka (s = 40% = 2 / 5, c = 50% = 2 / 4)

Jajka → Cukier (s = 40% = 2 / 5, c = 67% = 2 / 3)

# OKREŚLANIE REGUŁ ASOCJACYJNYCH

- ✓ **Wsparcie (*support*) s to ilość** transakcji zawierających zarówno X jak również Y, czyli $X \cup Y$.

$$s\,(A \Rightarrow B, D) = \frac{|\text{cover}(A \cup B\ ,\ D)|}{|D|}$$

- ✓ **Zaufanie/Pewność/Wiarygodność (*confidence*) c** – to prawdopodobieństwo warunkowe p(Y|X), że transakcja zawierająca X zawiera również Y.

$$c\,(A \Rightarrow B, D) = \frac{s\,(A \cup B\ ,\ D)}{s\,(A\ ,\ D)}$$

**Sposób obliczania wsparcia (*support*) s i pewności (*confidence*) c dla reguł asocjacyjnych określonych, dla których s ≥ 40%, c ≥ 50%:**

| ID TRANSAKCJI | ELEMENTY TRANSAKCJI |
|---|---|
| 1 | kawa, mleko, cukier, orzeszki |
| 2 | kawa, cukier, jajka |
| 3 | kawa, chleb, cukier, masło |
| 4 | orzeszki, cukier, miód, jajka |
| 5 | masło, mleko, jajka |

| ID TRANSAKCJI | ELEMENTY TRANSAKCJI |
|---|---|
| 1 | kawa, mleko, cukier, orzeszki |
| 2 | kawa, cukier, jajka |
| 3 | kawa, chleb, cukier, masło |
| 4 | orzeszki, cukier, miód, jajka |
| 5 | masło, mleko, jajka |

Jajka → Cukier (s = 40% = 2 / 5, c = 67% = 2 / 3)

Kawa → Cukier (s = 60% = 3 / 5, c = 100% = 3 / 3)

Cukier → Kawa (s = 60% = 3 / 5, c = 75% = 3 / 4)

Cukier → Jajka (s = 40% = 2 / 5, c = 50% = 2 / 4)

# OKREŚLANIE REGUŁ ASOCJACYJNYCH

- ✓ **Wsparcie (*support*) s to ilość** transakcji zawierających zarówno X jak również Y, czyli $X \cup Y$.

$$s\ (A \Rightarrow B, D) = \frac{|\text{cover}(A \cup B\ ,\ D)|}{|D|}$$

- ✓ **Zaufanie/Pewność/Wiarygodność (*confidence*) c** – to prawdopodobieństwo warunkowe p(Y|X), że transakcja zawierająca X zawiera również Y.

$$c\ (A \Rightarrow B, D) = \frac{s\ (A \cup B\ ,\ D)}{s\ (A\ ,\ D)}$$

**Sposób obliczania wsparcia (*support*) s i pewności (*confidence*) c dla wielowymiarowej reguły asocjacyjnej:**

| ID TRANSAKCJI | ELEMENTY TRANSAKCJI |
|---|---|
| 1 | kawa, mleko, cukier, orzeszki |
| 2 | kawa, cukier, jajka |
| 3 | kawa, chleb, cukier, masło |
| 4 | orzeszki, cukier, miód, jajka |
| 5 | masło, mleko, jajka |

| ID TRANSAKCJI | ELEMENTY TRANSAKCJI |
|---|---|
| 1 | kawa, mleko, cukier, orzeszki |
| 2 | kawa, cukier, jajka |
| 3 | kawa, chleb, cukier, masło |
| 4 | orzeszki, cukier, miód, jajka |
| 5 | masło, mleko, jajka |

Jajka $\wedge$ Cukier → Kawa (s = 20% = 1 / 5, c = 50% = 1 / 2)

# REGUŁY ASOCJACYJNE

- ✓ **Reguły asocjacyjne** przypominają reguły decyzyjne, lecz decyzja (czyli prawa strona implikacji) nie jest z góry określona.
- ✓ **Reguły asocjacyjne** działają podobnie jak uczenie nienadzorowane (bez nauczyciela) dla problemów algorytmów grupowania (klasteryzacji). Taki algorytm nie ma z góry określonej prawidłowej odpowiedzi. Zamiast tego ma opisywać wewnętrzne zależności między atrybutami.
- ✓ **Reguły asocjacyjne** wzięły się z badań nad zagadnieniami **analizy koszykowej** (***Market Basket Analysis***) polegającej na odkrywaniu wzorców zachowania się klientów, czyli znajdywaniu grup produktów kupowanych razem oraz określania ich częstości:

$$\bigwedge_{i \in I} a_i = v_i \Rightarrow a_k = v_k$$

- ✓ Do mierzenia obiektywizmu reguł asocjacyjnych wykorzystujemy dwa wskaźniki:
  - ✓ **Wsparcie (*support*)** – określające, ile procent spośród zbadanych transakcji występuje razem, np. w ilu transakcjach występuje kawa i cukier równocześnie.
  - ✓ **Zaufanie/Pewność/Wiarygodność (*confidence*)** – określa, ile procent transakcji zawiera wniosek (decyzję, czyli lewą stronę implikacji) przy założeniu, że spełniona jest lewa strona implikacji (transakcji), czyli: $c\ (X \Rightarrow Y) = s\ (X \cup Y)\ /\ s\ (X)$.
- ✓ Odpowiednie poziomy wymaganego wsparcia i wiarygodności określa użytkownik na podstawie potrzeb wynikających z danej dziedziny, wiedzy eksperta, zadania itp.
- ✓ Regułę asocjacyjną nazywamy **silną**, jeśli $s \geq s_{min}$ oraz $c \geq c_{min}$.

# TWORZENIE REGUŁ ASOCJACYJNYCH

- ✓ Tworzenie reguł asocjacyjnych poprzedzone jest zwykle procesem wyznaczania częstych wzorców, dla których te reguły tworzymy, gdyż zwykle (np. w handlu) jest istotne to, co często się powtarza, np. istotne są często kupowane produkty, które występują razem w różnych transakcjach. Pozwala to na lepsze zlokalizowanie tych produktów na półkach sklepowych, w celu zwiększenia ich sprzedaży.
- ✓ Poszukujemy wobec tego zwykle silnych reguł asocjacyjnych o odpowiednio zdefiniowanym minimalnym wsparciu oraz minimalnej ufności (wiarygodności).
- ✓ Czasami ciekawe mogą być te wzorce, które wystąpiły po raz pierwszy lub występują rzadko, np.: w astronomii, fizyce (np. zderzaczu hadronów), genetyce, kognitywistyce itd., wtedy poszukiwane mogą być tylko wzorce o niewielkim wsparciu lub wzorce unikalne.
- ✓ Do wyznaczania reguł asocjacyjnych oraz poszukiwania wzorców częstych wykorzystywany jest bardzo popularny algorytm Apriori, który posiada również liczne rozszerzenia mające na celu przyspieszenie jego działania.

# WZORCE SKŁADOWE

- ✓ Duże i długie wzorce zawierają (kombinatorycznie rzecz biorąc) sporą ilość wzorców składowych – subwzorców (*sub-patterns*):
  - ✓ podzbiorów elementów
  - ✓ subsekwencji elementów
  - ✓ podgrafów elementów
  - ✓ wycinków/obszarów elementów (np. dla obrazów, map)

- ✓ Subwzorce (*sub-patterns*) umożliwiają znajdywanie podobieństw i różnic oraz tworzenie asocjacyjnych związków pomiędzy wzorcami.
- ✓ Ze względu na czasami dużą kombinatoryczną złożoność niezbędnych do wykonania porównań, opłaca się najpierw analizować i porównywać subwzorce o mniejszej ilości obiektów składowych, a dopiero potem na ich podstawie określać, np. częstość, rzadkość, wsparcie czy pewność dla wzorców o większej ilości obiektów.

# WZORCE ZAMKNIĘTE

- ✓ **Zamknięte wzorce (*closed patterns*) X to takie wzorce, które są częste (*frequent*) i nie istnieje żaden nadwzorzec (*super-pattern*) Y ⊃ X, który miałby takie same wsparcie (*support*) jak wzorzec X.**
- ✓ **Zamknięty wzorzec (*closed pattern*) jest więc kompresją stratną wszystkich zawartych w nich częstych (frequent) wzorców, gdyż tracona jest informacja o ich wsparciu (*support*).**

**Przykład: Czy wzorzec „kawa" jest wzorcem zamkniętym (*closed*)? Jest wzorcem częstym o wsparciu ≥ 50%, lecz nie jest wzorcem zamkniętym, gdyż istnieje nadwzorzec „kawa ∪ cukier", który ma takie samo wsparcie równe 60% i go zawiera. Natomiast wzorzec „kawa ∪ cukier" jest zamknięty, gdyż nie istnieje żaden wzorzec o takim samym wsparciu, który by go obejmował.**

| ID TRANSAKCJI | ELEMENTY TRANSAKCJI |
|---|---|
| 1 | kawa, mleko, cukier, orzeszki |
| 2 | kawa, cukier, jajka |
| 3 | kawa, chleb, cukier, masło |
| 4 | orzeszki, cukier, miód, jajka |
| 5 | masło, mleko, jajka |

# WZORCE MAKSYMALNE

- ✓ **Maksymalne wzorce (*max-patterns*) X to takie wzorce, które są częste (*frequent*) i nie istnieje żaden częsty nadwzorzec (*super-pattern*) Y ⊃ X.**
- ✓ **Maksymalne wzorce (*max-patterns*) reprezentują wszystkie częste wzorce (*frequent patterns*), których wszystkie elementy zawierają.**
- ✓ **Maksymalny wzorzec (*max-pattern*) jest więc kompresją stratną wszystkich częstych wzorców składających się z jego elementów.**

**Przykład: Wzorzec „kawa ∪ cukier” jest nie tylko zamknięty, lecz również maksymalny, gdyż nie istnieje żaden częsty wzorzec, który by go zawierał.**

**Wzorce zamknięte od maksymalnych różnią się tym, iż wzorce zamknięte mogą posiadać częste nadwzorce o mniejszym wsparciu, zaś wzorce maksymalne takich nadwzorców nie posiadają.**

| ID TRANSAKCJI | ELEMENTY TRANSAKCJI |
|---|---|
| 1 | kawa, mleko, cukier, orzeszki |
| 2 | kawa, cukier, jajka |
| 3 | kawa, chleb, cukier, masło |
| 4 | orzeszki, cukier, miód, jajka |
| 5 | masło, mleko, jajka |

# ALGORYTMY EKSPLORACJI DANYCH

**Najpopularniejsze algorytmy (*state-of-the-art algorithms for the search of frequent patterns*) wyszukiwania zbiorów wzorców częstych:**

- ✓ **Apriori (Rakesh Agrawal, John C. Shafer)**
- ✓ **Eclat (Mohammed J. Zaki, Mitsunori Ogihara, Srinivasan Parthasarathy, Wei Li)**
- ✓ **FP-growth (Jiawei Han, Jian Pei, Yiwen Yin)**
- ✓ **D-Club (Jianwei Li, Alok Choudhary, Nan Jiang, Wei-keng Liao), PD-Club etc.**
- ✓ **Mining Close Frequent Patterns (…)**
- ✓ **MaxPattern Eclat (…)**
- ✓ **Partition, MAFIA, LCM, kDCI i wiele innych…**

**Wszystkie te algorytmy posiadają wersje sekwencyjne i równoległe.**

**Równoległe wersje algorytmów można zaimplementować np. w zorientowanym obiektowo języku programowania Charm++, który wykorzystuje paradygmat programowania równoległego oparty o asynchroniczną wymianę komunikatów, aczkolwiek wymiana synchroniczna również jest możliwa:**

- ➢ **Aplikacje w Charm++ są kolekcjami kontenerów równoległych (*chares*), które mogą swobodnie migrować pomiędzy fizycznymi jednostkami (procesorami lub rdzeniami) przetwarzającymi je.**
- ➢ **Komunikacja pomiędzy kontenerami równoległymi odbywa się poprzez wysyłanie komunikatów,**
- ➢ **Po odebraniu komunikatu wywoływana jest metoda wejściowa (*entry method*) kontenera,**
- ➢ **Operacja przekazania wiadomości jest nie blokująca.**

# ALGORYTMY APRIORI

**Apriori iteracyjny** stopniowo generuje i testuje częstość kandydatów dzieląc ich na zbiory częste o długości od 1 do k aż do momentu gdy znalezione zostaną wszystkie wzorce częste.

**Apriori równoległe** dzielimy na następujące grupy algorytmów:

- **Count Distribution** – algorytmy bazujące na dekompozycji danych, tzn. baza dzielona jest na statyczne partycje, w których równolegle zliczane jest wsparcie dla kandydatów na zbiory częste (niezależnie od siebie).
- **Data Distribution** – algorytmy dążące do efektywniejszego wykorzystania pamięci RAM poprzez podzielenie bazy na partycje i rozdzielenie listy wyszukiwanych kandydatów pomiędzy procesory lub rdzenie. Każdy kandydat jest wyszukiwany przez tylko jeden proces, więc procesy muszą wymieniać się partycjami w czasie każdej iteracji.
- **Candidate Distribution** – algorytmy rozdzielające listę wyszukiwanych kandydatów i replikujące transakcje z bazy danych w celu równoległego ich przetwarzania.

**Apriori asocjacyjne** wykorzystują asocjacyjne grafowe struktury danych AGDS i mogą działać zarówno w wersji sekwencyjnej, jak i równoległej, a ze względu na fakt, iż nie muszą przeglądać całej kolekcji transakcji wielokrotnie, gdyż przechowują relacje pomiędzy transakcjami i elementami oraz częstość elementów jest określona w strukturze AGDS, działają najszybciej.

# REGUŁA OCZYSZCZANIA APRIORI

**Reguła oczyszczania Apriori mówi, iż każdy podzbiór zbioru częstego (*frequent itemset*) jest częsty (*frequent*).**

**Wnioski wynikające z tej reguły:**

- ✓ **Każdy podzbiór zbioru częstego nie może być rzadki, ale musi być częsty.**
- ✓ **Może być częstszy, czyli jego wsparcie (*support*) może być większe.**
- ✓ **Jeśli jakikolwiek podzbiór zbioru S jest rzadki (*infrequent*), wtedy zbiór S jest również rzadki (*infrequent*).**

**Powyższy wniosek umożliwia odfiltrowanie wszystkich nadwzorców (*super-patterns*), które zawierają rzadkie (*infrequent*) podzbiory (*itemsubsets*), w celu podniesienia efektywności przeszukiwania wzorców w trakcie ich eksploracji, gdyż wszystkie nadwzorce wzorców rzadkich są rzadkie, więc nie trzeba ich rozważać w trakcie dalszego przeszukiwania.**

**Reguła oczyszczania Apriori (*pruning principle*) mówi, iż jeśli istnieje jakikolwiek podzbiór (*itemsubset*), który jest rzadki (*infrequent*), wtedy jego dowolny zawierający go zbiór (*superset*) nie powinien być uwzględniany/generowany w procesie eksploracji.**

# ALGORYTM OCZYSZCZANIA APRIORI

Algorytm Apriori w C++ (szybszy)

Algorytm Apriori w Python (wolniejszy)

Prezentacja: https://www.youtube.com/watch?v=WGlMlS_Yydk

## Opis algorytmu APRIORI:

1. **Krok oczyszczania (*prune step*):**
   Przeszukujemy wszystkie wzorce w celu ustalenia ilości każdego kandydata w k-elementowym podzbiorze $C_k$. Ilości te są porównywane z ustalonym minimalnym wsparciem (suport) s, w celu ustalenia, czy dany kandydat może zostać umieszczony w zbiorze $L_k$ częstych wzorców.
   *It scans the entire database to find the count of each candidate in $C_k$ where $C_k$ represents candidate k-itemset. The count of each itemset in $C_k$ is compared with a predefined minimum support count to find whether that itemset can be placed in frequent k-itemset $L_k$.*

2. **Krok łączenia (*join step*):**
   Wzorce ze zbioru $L_k$ są naturalnie łączone ze sobą w celu wygenerowania następnych k+1 elementowych kandydatów $C_{k+1}$ (badane są ich kombinacje). Najważniejsze jest przeszukanie wszystkich wzorców w celu wyznaczenia ilości każdego podzbioru dla każdego k-elementowego kandydata $C_k$. W wyniku przeszukiwania należy określić wszystkie częste (powyżej pewnego progu minimalnego) podzbiory, jakie powstaną w wyniku takiego złączenia.
   *$L_k$ is natural joined with itself to generate the next candidate k+1-itemset $C_{k+1}$. The major step here is the prune step which requires scanning the entire database for finding the count of each itemset in every candidate k-itemset. If the database is huge then it requires more time to find all the frequent itemsets in the database.*

# ALGORYTM APRIORI

## Algorytm APRIORI:

```
L1 = {1-element frequent pattern sets};
for (k=2; Lk-1.IsNotEmpty(); k++) do
{
        Ck = apriori_gen(Lk-1);
        foreach t∈D do  // t - transaction, D -transaction set
        {
            Ct = subset(Ck, t);
            foreach c in Ct do      // c - candidate set
                c.count++;    // count the number of occurrences
        }
        Lk = {c∈Ck | c.count >= minsup}
}
all_frequent_pattern_set = ∪Lk;
```

# ALGORYTM APRIORI

## Algorytm APRIORI:

```
function apriori_gen(Ck)
{
    insert into Ck
    select p.item1, p.item2, ..., p.itemk-1, q.itemk-1
    from Lk-1 p, Lk-1 q
    where p.item1 = q.item1, ..., p.itemk-2 = q.itemk-2, p.itemk-1 <
    q.itemk-1;
    forall itemsets c in Ck do
        forall (k-1)-subsets s of c do
            if ( s not in Lk-1 ) then
                delete c from Ck;
}
```

**Algorytm Apriori do znajdywania reguł asocjacyjnych:**

http://www.borgelt.net/apriori.html, http://www.borgelt.net/docs/apriori.pdf

# EKWIWALENTNA TRANSFORMACJA KLAS

**Ekwiwalentna Transformacja Klas ECLAT (*Equivalence Class Transformation*) to algorytm przeszukiwania w głąb (*DFS depth-first search*) wykorzystujący przecięcie zbiorów. Służy do eksploracji częstych wzorców poprzez badanie ich wertykalnego (kolumnowego) formatu, przyspieszając działanie Apriori, gdyż nie musi przeszukiwać bazy danych w celu określenia wsparcia dla elementów k+1 elementowych:**

**$t(B) = \{T_2, T_3\}$; $t(C) = \{T_1, T_3\}$ ➔ $t(BC) = \{T_3\}$**

**$t(E) = \{T_1, T_2, T_3\}$ ➔ diffset (BE, E) = $\{T_1\}$ – zbiór różnic**

| HORYZONTALNY FORMAT DANYCH | |
|---|---|
| TRANSAKCJE | ELEMENTY ZBIORU |
| 1 | A, C, D, E |
| 2 | A, B, E |
| 3 | B, C, E |

TRANSFORMACJA

| WERTYKALNY FORMAT DANYCH | |
|---|---|
| ELEMENT | LISTA TRANSAKCJI |
| A | 1, 2 |
| B | 2, 3 |
| C | 1, 3 |
| D | 1 |
| E | 1, 2, 3 |

tablica asocjacji

**Dzięki takiej transformacji wiemy, w których transakcjach występują poszczególne elementy, więc łatwiej można je analizować!**

**Dodatkowy materiał poszerzający opis tej transformacji:**
**http://research.ijcaonline.org/volume90/number8/pxc3894337.pdf**

# ALGORYTM ECLAT
# Equivalent CLAss Transformation

**Klasyczne generowanie reguł asocjacyjnych obejmuje dwa kroki:**

1. **Kosztowne obliczeniowo generowanie zbiorów elementów częstych, czyli takich dla których wsparcie przekracza pewny przyjęty próg (czyli wsparcie minimalne).**
2. **Generowanie reguł asocjacyjnych.**

**Wykorzystanie ECLAT przebiega następująco:**

1. **Efektywne generowanie zbiorów elementów częstych dzięki zmianie formatu reprezentacji z horyzontalnego na wertykalny.**
2. **Generowanie reguł asocjacyjnych.**

| HORYZONTALNY FORMAT DANYCH | |
|---|---|
| TRANSAKCJE | ELEMENTY ZBIORU |
| 1 | A, C, D, E |
| 2 | A, B, E |
| 3 | B, C, E |

TRANSFORMACJA

| WERTYKALNY FORMAT DANYCH | |
|---|---|
| ELEMENT | LISTA TRANSAKCJI |
| A | 1, 2 |
| B | 2, 3 |
| C | 1, 3 |
| D | 1 |
| E | 1, 2, 3 |

tablica asocjacji

```
ECLAT (S_{k-1})
{
        forall itemsets I_a, I_b ∈ S_{k-1}, where a < b do
        {
                C = I_a ∩ I_b
                if (C.support ≥ minsup) add C to L_k
        }
        partition L_k into prefix-based (k-1)-lenght prefix classes S_k
        foreach class S_k in L_k do ECLAT (S_k)
}
```

Materiały uzupełniające: http://ijctjournal.org/Volume2/Issue3/IJCT-V2I3P17.pdf , https://www.youtube.com/watch?v=oBiq8cMkTCU

Eclat algorytm: http://www.borgelt.net/eclat.html

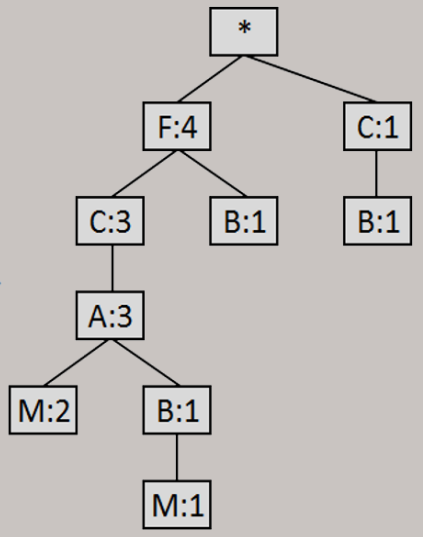
# Transformacja wzorców do FP-drzewa

**Algorytm FP-Growth dokonuje najpierw kompresję stratną reprezentacji wzorców w postaci drzewa, które następnie jest wykorzystywane do eksploracji:**

1. Znajdź wszystkie 1-elementowe zbiory częste w bazie transakcji D.
2. Transformuj każdą transakcję $T_i \in D$ do skompresowanej (uproszczonej) postaci $\breve{T}_i$ polegającej na usunięciu z $T_i$ wszystkich elementów, które nie są częste.
3. Posortuj elementy transakcji skompresowanych $\breve{T}_i$ według malejących wartości ich wsparcia tworząc listę elementów.
4. Transformuj posortowane transakcje $\breve{T}_1, \breve{T}_2, \ldots, \breve{T}_N$ do FP-drzewa.

| TRANSAKCJE | ELEMENTY TRANSAKCJI | UPORZĄDKOWANE CZĘSTE ELEMENTY |
|---|---|---|
| 1 | A,C,D,G,F,I,M,P | F,C,A,M |
| 2 | A,B,C,F,L,M,R | F,C,A,B,M |
| 3 | B,F,H,J,R,W | F,B |
| 4 | B,C,K,S,P | C,B |
| 5 | A,C,E,F,L,M,N | F,C,A,M |

| ELEMENTY minsup = 3 | CZĘSTOTLIWOŚĆ |
|---|---|
| F | 4 |
| C | 4 |
| A | 3 |
| B | 3 |
| M | 3 |



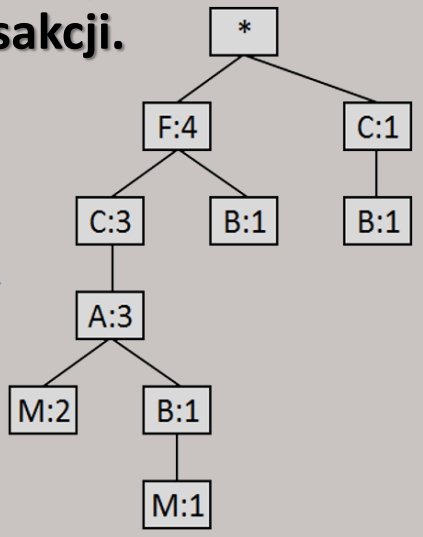
# Tworzenie FP-drzewa

**Budowa FP-drzewa:**

- Korzeń grafu posiada etykietę "null", a pozostałe wierzchołki grafu, zarówno wierzchołki wewnętrzne jak i liście, reprezentują 1-elementowe zbiory częste wraz z informacją o ich liczności (wsparciu) w zbiorze transakcji D.
- Kolejne skompresowane (uproszczone) transakcje $\breve{T}_i$ z posortowanymi elementami dodawaj po kolei do FP-drzewa poruszając się od korzenia przez istniejące węzły, jeśli dany element jest już reprezentowany, inkrementując ich liczniki, albo dodaj nowy element do drzewa, jeśli nie było jeszcze dla danego poprzednika reprezentowany oraz nadaj mu licznik 1.
- W taki sposób tworzone jest drzewo prefiksowe dla transakcji $\breve{T}_i$ na bazie agregacji wspólnych prefiksów dla elementów transakcji.

| TRANSAKCJE | ELEMENTY TRANSAKCJI | UPORZĄDKOWANE CZĘSTE ELEMENTY |
|---|---|---|
| 1 | A,C,D,G,F,I,M,P | F,C,A,M |
| 2 | A,B,C,F,L,M,R | F,C,A,B,M |
| 3 | B,F,H,J,R,W | F,B |
| 4 | B,C,K,S,P | C,B |
| 5 | A,C,E,F,L,M,N | F,C,A,M |

| ELEMENTY minsup = 3 | CZĘSTOTLIWOŚĆ |
|---|---|
| F | 4 |
| C | 4 |
| A | 3 |
| B | 3 |
| M | 3 |

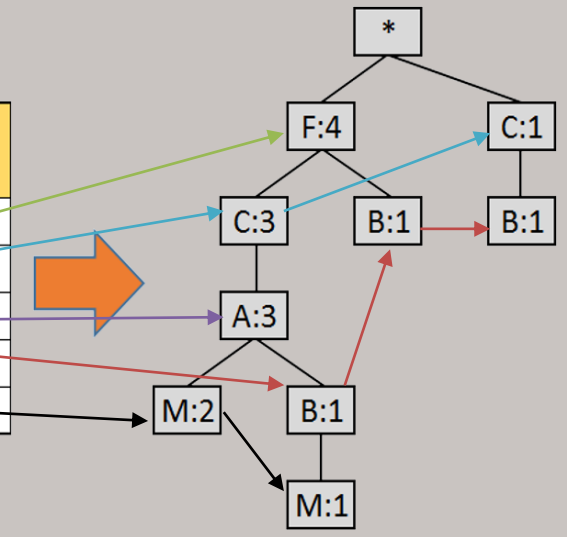


# Odczytywanie wsparcia z FP-drzewa

- Informację o wsparciu wzorca reprezentowanego przez prefiks określamy na podstawie liczności ostatniego elementu prefiksu (czyli jego wsparcia), np. FCA ma wsparcie 3, gdyż A ma wsparcie 3.
- W przypadku elementów występujących wielokrotnie w drzewie, np. B lub C, przydatna jest tablica nagłówkowa, przechowująca listy wskaźników do poszczególnych wystąpień danego elementu w drzewie, co przyspiesza przeszukiwanie FP-drzewa (jak zaprezentowano strzałkami poniżej).

| TRANSAKCJE | ELEMENTY TRANSAKCJI | UPORZĄDKOWANE CZĘSTE ELEMENTY |
|---|---|---|
| 1 | A,C,D,G,F,I,M,P | F,C,A,M |
| 2 | A,B,C,F,L,M,R | F,C,A,B,M |
| 3 | B,F,H,J,R,W | F,B |
| 4 | B,C,K,S,P | C,B |
| 5 | A,C,E,F,L,M,N | F,C,A,M |

| ELEMENTY minsup = 3 | CZĘSTOTLIWOŚĆ |
|---|---|
| F | 4 |
| C | 4 |
| A | 3 |
| B | 3 |
| M | 3 |



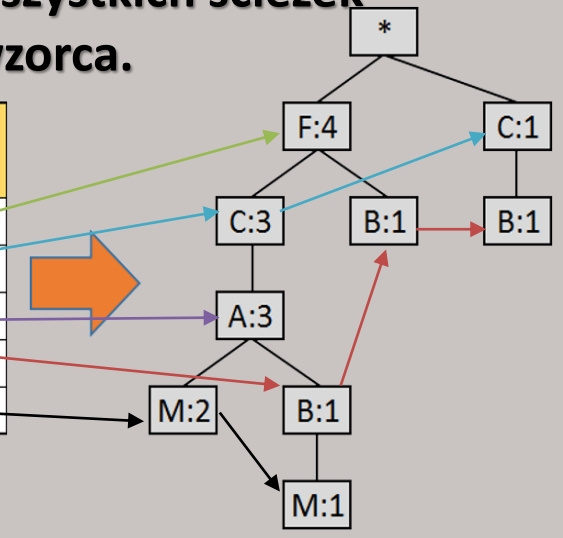
# Eksploracja FP-drzewa

**Proces eksploracji FP-drzewa bazuje na obserwacji, iż dla każdego 1-elementowego zbioru częstego α wszystkie częste nadzbiory zbioru α są reprezentowane w FP-drzewie poprzez ścieżki zawierające wierzchołek (wierzchołki) α:**

1. **Dla każdego 1-elementowego zbioru częstego α znajdujemy w FP-drzewie wszystkie ścieżki, których końcowym wierzchołkiem jest wierzchołek reprezentujący zbiór α. Taką ścieżki nazywamy ścieżkami prefiksowymi wzorca α.**
2. **Z każdą prefiksową ścieżką wzorca α związany jest licznik częstości ścieżki, którego wartość równa jest wartości licznika transakcji wierzchołka końcowego ścieżki reprezentującego zbiór α. Zbiór wszystkich ścieżek prefiksowych wzorca tworzy tzw. warunkową bazę wzorca.**

| TRANSAKCJE | ELEMENTY TRANSAKCJI | UPORZĄDKOWANE CZĘSTE ELEMENTY |
|---|---|---|
| 1 | A,C,D,G,F,I,M,P | F,C,A,M |
| 2 | A,B,C,F,L,M,R | F,C,A,B,M |
| 3 | B,F,H,J,R,W | F,B |
| 4 | B,C,K,S,P | C,B |
| 5 | A,C,E,F,L,M,N | F,C,A,M |

| ELEMENTY minsup = 3 | CZĘSTOTLIWOŚĆ |
|---|---|
| F | 4 |
| C | 4 |
| A | 3 |
| B | 3 |
| M | 3 |



# Eksploracja FP-drzewa

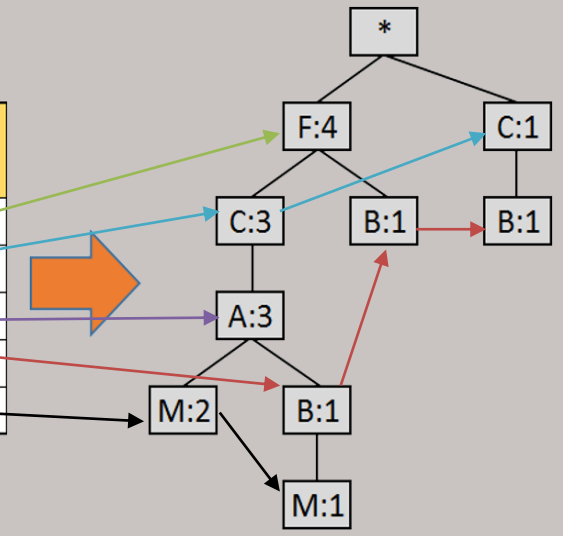
3. Warunkowa baza wzorca służy do konstrukcji tzw. warunkowego FP-drzewa wzorca α, oznaczanego Tree-α.
4. Następnie, warunkowe FP-drzewo jest rekurencyjnie eksplorowane w celu znalezienia wszystkich zbiorów częstych zawierających zbiór α.

Algorytm FP-Growth znajduje wszystkie zbiory częste.

Parametry początkowe procedury FP-Growth, w momencie inicjacji procedury, są następujące: Tree = FP-drzewo oraz α = null.

| TRANSAKCJE | ELEMENTY TRANSAKCJI | UPORZĄDKOWANE CZĘSTE ELEMENTY |
|---|---|---|
| 1 | A,C,D,G,F,I,M,P | F,C,A,M |
| 2 | A,B,C,F,L,M,R | F,C,A,B,M |
| 3 | B,F,H,J,R,W | F,B |
| 4 | B,C,K,S,P | C,B |
| 5 | A,C,E,F,L,M,N | F,C,A,M |

| ELEMENTY minsup = 3 | CZĘSTOTLIWOŚĆ |
|---|---|
| F | 4 |
| C | 4 |
| A | 3 |
| B | 3 |
| M | 3 |



# Eksploracja Danych

- **Inteligentna eksploracja danych** pozwala wyłuskać z posiadanych danych wiele istotnych informacji o zjawiskach częstych, powtarzalnych i przydatnych dla marketingu, reklamy i rozwoju firm, banków i instytucji.

- **Wyniki eksploracji** pomagają ukierunkować działania na rzeczy najistotniejsze, mogące przynosić największe korzyści, jak również wskazywać sprawy zaniedbane lub sytuacje, które odbiegają od normalnych, poprawnych i zgodnych z oczekiwaniami.

- **Eksploracja danych daje nam wiedzę** na temat częstych i istotnych zjawisk, na które powinniśmy zwrócić uwagę oraz uwzględnienie których może być dla nas korzystne, zaś ich zaniedbanie brzemienne w skutkach.

- **Eksploracja danych** pomaga formować wiedzę o analizowanych procesach i danych oraz wspomaga inteligentne procesy wnioskowania na podstawie danych, więc jest istotnym składnikiem systemów inteligentnych.

# Etapy Eksploracji

1. Zrozumienie zadania i zdefiniowanie celu praktycznego eksploracji, czyli przyporządkowanie zadania do grupy: klasyfikacji, grupowania, predykcji lub asocjacji.
2. Przygotowanie bazy danych do analizy poprzez wyselekcjonowanie rekordów z baz danych najlepiej charakteryzujących rozważany problem.
3. Czyszczenie i wstępna transformacja danych poprzez ich normalizację, standaryzację, usuwanie danych odstających, usuwanie lub uzupełnianie niekompletnych wzorców.
4. Transformacja danych z postaci symbolicznej na postać numeryczną poprzez przypisanie im wartości lub rozmywanie (*fuzzification*) w zależności od stosowanej metody ich dalszego przetwarzania.
5. Redukcja wymiaru danych i selekcja najbardziej znaczących i dyskryminujących cech pozwalających uzyskać najlepsze zdolności uogólniające projektowanego systemu.
6. Wybór techniki i metody eksploracji danych na podstawie możliwości danej metody oraz rodzaju i liczności danych: numeryczne, symboliczne, sekwencyjne...
7. Wybór algorytmu lub aplikacji implementującej wybraną technikę eksploracji danych oraz określenie optymalnych parametrów adaptacji/uczenia wybranej metody (przydatne mogą tutaj być metody ewolucyjne, genetyczne, walidacja krzyżowa).
8. Przeprowadzenie procesu konstrukcji, adaptacji lub uczenia wybraną metodą.
9. Eksploatacja systemu: wnioskowanie, określanie grup, podobieństw, różnic, zależności, następstwa lub implikacji.
10. Douczanie systemu na nowych danych lub utrwalanie zebranych wniosków z eksploracji.

# Klasyfikacja

**Klasyfikacja** to zadanie przyporządkowania **obiektu** do **pewnej klasy** na podstawie podobieństwa, czyli rozpoznawania **obiektu** jako elementu **pewnej klasy**. W wyniku klasyfikacji wzorcowi zostaje przyporządkowana pewna klasa, reprezentowana zwykle przez pewną **etykietę klasy**.

**Klasa** – to pewna **grupa wzorców** charakteryzujących się podobnymi cechami/właściwościami dla określających je atrybutów/parametrów.

Jeśli **wzorzec** należy równocześnie do kilku klas, wtedy mówimy o zagadnieniu **multiklasyfikacji** (*multiclass classification*), np.:

*ser Mozzarella należy do klas: serów, nabiału, produktów spożywczych.*

**Sklasyfikowanie wzorca** jako przynależnego do określonej klasy może być rozważane jako proces:

- rozmyty / predyktywny / ciągły: o określonym stopniu przynależności do klasy
- binarny / zero-jedynkowy / dyskretny: należy lub nie należy do klasy

Klasyfikacja to bardzo ważny proces,
bez którego trudne byłoby formowanie wiedzy,
jak również inteligentne działanie!

# Klasteryzacja

**Klasteryzacja** to proces grupowania obiektów (wzorców) na podstawie ich podobieństwa w taki sposób:

- iż wzorce należące do różnych klastrów są rozłączne (klasteryzacja silna),
- iż wzorce mogą równocześnie należeć do kilku klastrów (klasteryzacja słaba).

**Klaster** to grupa obiektów podobnych, czyli takich, które są bliskie siebie w pewnej przestrzeni w porównaniu do obiektów innych klastrów, do których są względnie dalekie (niepodobne).

Do najpopularniejszych **metod klasteryzacji** należą:

- Algorytm k-średnich (*k-means clustering*)
- Klasteryzacja hierarchiczna (*hierarchical clustering*)
- Klasteryzacja spektralna (spectra clustering)

# Model

**Model** – to zwykle pewien algorytm lub wzór matematyczny połączony z pewną strukturą lub sposobem reprezentacji przetworzonych danych źródłowych, określany w trakcie procesu uczenia, adaptacji lub konstrukcji.

**Obserwacja** – to zestaw pomiarów tworzących jeden rekord danych (krotkę).

**Predykcja** – to wynik procesu regresji lub kojarzenia, w którym otrzymujemy odpowiedź w postaci liczbowej lub innego obiektu.

**Redukcja** – to proces kompresji stratnej polegający na zmniejszeniu wymiaru wektorów lub macierzy obserwacji poprzez eliminację mało reprezentatywnych lub niekompletnych atrybutów albo w wyniku określania pochodnych reprezentatywnych cech (np. PCA, ICA).

# Uczenie

**Adaptacja** – to polegający na przedstawieniu danych uczących oraz dobraniu, dopasowaniu lub obliczeniu wartości modelu tak, aby dostosował swoje działanie do określonego zbioru, typu i ew. pożądanych wartości wyjściowych danych uczących.

**Uczenie** – to proces iteracyjny polegający na wielokrotnym przedstawianiu danych uczących oraz poprawianiu wartości modelu tak, aby dostosował swoje działanie do określonego zbioru, typu i ew. pożądanych wartości wyjściowych danych uczących.

**Uczenie** może być:

- nienadzorowane (bez nauczyciela, *unsupervised*),
- nadzorowane (z nauczycielem, *supervised*),
- konkurencyjne (*competitive*),
- przez wzmacnianie (*reinforcement*),
- motywowane (*motivated*),
- Hebbowskie,
- Bayesowskie (*Bayes*),
- skojarzeniowe (*associative*).

# Testowanie

**Testowanie** – to proces sprawdzania jakości modelu przeprowadzanym w trakcie procesu uczenia lub adaptacji modelu:

- na zbiorze danych chwilowo wydzielonych i wykluczonych z procesu uczenia (tzw. **walidacja** np. krzyżowa – *n-fold cross validation*) lub
- na zbiorze danych testowych całkowicie wykluczonych z procesu uczenia/adaptacji modelu (testowanie właściwe).

**Wzorzec** – to zestaw lub sekwencja albo inna struktura danych reprezentowanych w postaci zbioru, wektora, macierzy, sekwencji albo grafu danych stosowana do budowy, adaptacji, uczenia, walidacji i testowania modelu.

Wzorce stosowane w trakcie:

- uczenia nazywamy **wzorcami uczącymi**;
- walidacji nazywamy **wzorcami walidacyjnymi**;
- testowania nazywamy **wzorcami testującymi**.

# Atrybuty i Cechy

**Atrybut** – to jedna z cech (parametrów) opisujących obiekt za pośrednictwem wartości reprezentujących ten atrybut. Wartości te są określonego typu i mogą posiadać wartości z pewnego zakresu lub zbioru.

**Cecha diagnostyczna** – deskryptor numeryczny charakteryzujący i opisujący analizowany proces, zwany również atrybutem procesu.

**Ekstrakcja cech diagnostycznych** – to proces tworzenia atrybutów wejściowych dla modelu eksploracji na podstawie wyników pomiarowych. Proces ten nazywany jest również **generacją cech**.
Proces ten może być powiązany z **normalizacją**, **standaryzacją** lub inną transformacją danych, mających na celu uwydatnienie głównych cech modelowanego procesu, które mają istotny wpływ na budowę modelu oraz uzyskiwane wyniki i uogólnienie.

# Normalizacja

**Normalizacja** – to przeskalowanie danych względem wielkości skrajnych (min i max) danego wektora danych najczęściej do zakresu [0, 1] (czasami do [-1, 1]) zgodnie z następującą zależnością:

$$y_i = \frac{x_i - x_{min}}{x_{max} - x_{min}}$$

$x = [x_1, x_2, \ldots, x_N]$ – to N-elementowy wektor danych źródłowych,

$y = [y_1, y_2, \ldots, y_N]$ – to N-elementowy wektor danych po normalizacji.

Normalizacja jest wrażliwa na wartości odstające i o dużym rozrzucie, gdyż wtedy właściwe dane zostaną ściśnięte w wąskim przedziale, co może znacząco utrudnić ich dyskryminację!

Przeprowadzenie normalizacji jest czasami niezbędne do zastosowania metody, która wymaga, aby dane wejściowe lub wyjściowe mieściły się w pewnym zakresie, np. stosując funkcje sigmoidalną lub tangens hiperboliczny.

# Standaryzacja

**Standaryzacja** – to powszechnie stosowana w statystyce operacja polegająca na przeskalowaniu danych każdego elementu zbioru względem wartości średniej oraz odchylenia standardowego zgodnie z wzorem:

$$y_i = \frac{x_i - m}{\sigma}$$

$x = [x_1, x_2, \ldots, x_N]$ – to N-elementowy wektor danych źródłowych,

$y = [y_1, y_2, \ldots, y_N]$ – to N-elementowy wektor danych po standaryzacji.

$m$ – to wartość średnia wyznaczona z tych danych,

$\sigma$ – to odchylenie standardowe.

W wyniku standaryzacji otrzymujemy wektor cech, którego wartość średnia jest zerowa, natomiast odchylenie standardowe jest równe jedności.

Nie należy stosować dla danych o odchyleniu standardowym bliskim zeru!

# Asocjacje

**Asocjacja / Skojarzenie (*associations*)** – to proces stowarzyszenia ze sobą dwu lub więcej obserwacji (danych, obiektów, wzorców, encji).

W najprostszej postaci opisywana jest często przez **reguły asocjacyjne**.

**Asocjacje** są również postawą działania ludzkiego mózgu, pamięci i inteligencji, więc mogą być reprezentowane przez skomplikowane sieci neuronowe.

**Uogólnienie / Generalizacja (*generalization*)** – to zdolność lub właściwość modelu eksploracji danych polegająca na możliwości poprawnego działania (np. przewidywania, klasyfikacji, regresji) modelu na innych danych niż dane uczące.

# Transformacje

**Metody redukcji i transformacji danych** – mają za zadanie doprowadzić do optymalnej reprezentacji dużych ilości danych, tj. takiej ich reprezentacji, żeby dane w dalszym ciągu były reprezentatywne dla rozważanego problemu, np. klasyfikacji, czyli umożliwiały poprawną dyskryminację wzorców, tj. rozróżnienie ich według pozostałych po redukcji danych.

- **Optymalna reprezentacja** danych może być osiągnięta na skutek:
- **Redukcji wymiaru danych** – czyli **usuwania** mniej istotnych atrybutów danych, oraz **selekcji** atrybutów najistotniejszych pod kątem rozwiązywanego zadania.
- **Transformacji danych** – czyli przekształcenia danych do innej, bardziej oszczędnej lub mniej wymiarowej postaci, która dalej pozwala na ich poprawne rozróżnianie i przetwarzanie, np.:
  - metoda analizy głównych składowych (PCA – Principal Component Analysis),
  - metoda analizy składowych niezależnych (ICA – Independent Component Analysis).
- **Agregacji i Asocjacji danych (*Aggregate & Associate*)** – czyli takiej reprezentacji danych, która polega na zagregowaniu reprezentacji takich samych i/lub podobnych danych i ich grup oraz ich odpowiednim do rozwiązywanego zadania powiązaniu w celu przyspieszenia ich przeszukiwania i przetwarzania.

# Wizualizacja

**Wizualizacja i prezentacja** to zadania związane z graficzną reprezentacją danych w takiej postaci, żeby zaprezentować dane w taki sposób, aby możliwe było:

- porównanie liczności danych określonego typu/grupy/zbioru/klasy,
- wskazanie zależności (relacji) pomiędzy danymi i ich grupami,
- wskazanie minimów, maksimów, średnich, odchyleń i wariancji danych,
- wskazanie rozkładów, agregacji, środków ciężkości,
- wskazanie podobieństw i różnic pomiędzy danymi i ich grupami,
- wskazanie reprezentantów, typowych i nietypowych danych,
- wskazanie wzorców lub wartości odstających od przeciętnych (*outlier*), błędnych, brakujących lub szczególnych,
- podział, odfiltrowanie lub selekcja pewnej grupy wzorców,
- oceny pokrycia przestrzeni danych i ich reprezentatywności dla zadania,
- oceny jakości, zaszumienia, poprawności, dokładności i pełności danych.

# Inteligencja

**Inteligencja** to mentalna zdolność postrzegania informacji i wykorzystywania jej do formowania wiedzy, w celu jej zastosowania do adaptacji do środowiska, rozwiązywanego problemu lub efektywnego osiągania celów.

**Inteligencja** to mentalna zdolność rozumowania, planowania, rozwiązywania problemów, abstrakcyjnego myślenia, rozumienia złożonych idei, szybkiego uczenia się i efektywnego wykorzystywania zasobów.

**Inteligencja** obejmuje procesy uczenia się, rozpoznawania, klasyfikacji, rozumienia, logiki, planowania, kreatywności, rozwiązywania problemów i samoświadomości.

# Mądrość

**Mądrość** to umiejętność wyboru najlepszego, rozsądnego, wydajnego i najbardziej dochodowego sposobu osiągnięcia pożądanego rezultatu w oparciu o wiedzę, potrzeby, inteligencję i etyczne priorytety.

**Mądrość** pozwala na dobrą ocenę oraz wysoką jakość bycia.

**Mądrość** jest zwykle wynikiem wcześniejszych prób osiągnięcia pomyślnego wyniku na postawie posiadanego doświadczenia, wiedzy i inteligencji.

**Mądrość** jest więc traktowana jako przejaw wysokiej inteligencji oraz posiadanej szerokiej wiedzy.

# Czy mamy już sztuczną inteligencję?

Wbrew marketingowi i dobrze rozreklamowanym terminom, tj.:

- "Sztuczna inteligencja" [Google zwraca ok. 1 500 000 stron]
- "Artificial intelligence" [Google zwraca ok. 122 000 000 stron]
- "Uczenie maszynowe" [Google zwraca ok. 158 000 stron]
- "Machine learning" [Google zwraca ok. 107 000 000 stron]
- "Deep learning" [Google zwraca ok. 31 800 000 stron]

obecnie funkcjonujące algorytmy można nazwać co najwyżej inteligentnymi, a nie posiadającymi inteligencję, gdyż zostały opracowane przez inteligentnych ludzi i mają możliwość dostosowania swoich parametrów działania do wyznaczonego wąskiego obszaru działań, na którym zostały wytrenowane i dla których zostały opracowane. Daleko im jednak do ludzkiej inteligencji, sprytu i niezależności.

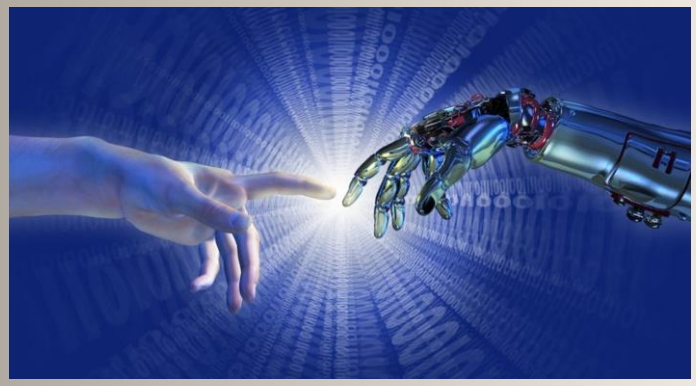
Z drugiej strony cały czas trwają intensywne prace naukowców, którzy pracują nad nowym:

asocjacyjnymi strukturami danych (działającymi podobnie jak ludzki mózg),

- algorytmami uczenia się,
- metodami adaptacji na podstawie różnego rodzaju danych,
- metodami formowania się wiedzy,
- metodami kognitywnymi,
- psychologicznymi aspektami inteligencji i modelowaniem potrzeb oraz czynników motywacyjnych

iż w pewnym momencie prawdziwa sztuczna inteligencja może się pojawić…

# Zbudujmy inteligentne systemy oparte na wiedzy!



- ✓ Pytania?
- ✓ Uwagi?
- ✓ Sugestie?
- ✓ Życzenia?

# Literatura i Bibliografia

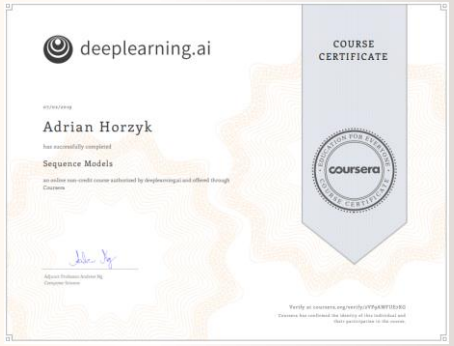

1. Ian Goodfellow, Yoshua Bengio, Aaron Courville, Deep Learning, MIT Press, 2016, ISBN 978-1-59327-741-3 or PWN 2018.
2. Holk Cruse, Neural Networks as Cybernetic Systems, 2nd and revised edition
3. R. Rojas, Neural Networks, Springer-Verlag, Berlin, 1996.
4. Convolutional Neural Network (Stanford)
5. Visualizing and Understanding Convolutional Networks, Zeiler, Fergus, ECCV 2014
6. Lectures of Alessandro Sperduti of Universita Degli Studi di Padova
7. Exploading Gradient Problem
8. LSTM cells from scratch and the code
9. Understanding LSTM

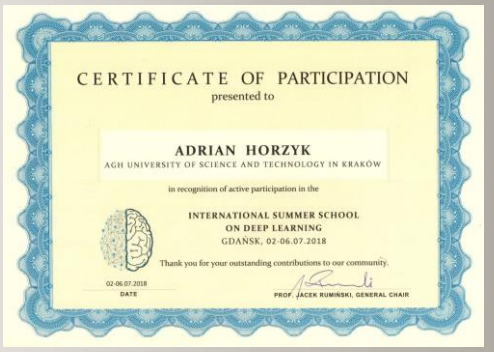
**Adrian Horzyk**

horzyk@agh.edu.pl

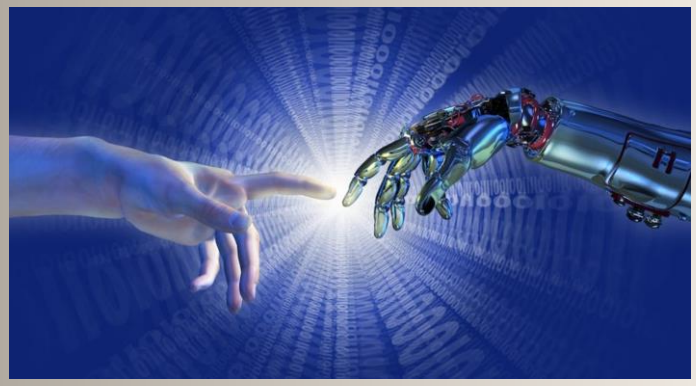
Google: Horzyk

Akademia Górniczo-Hutnicza w Krakowie



# Zbudujmy inteligentne systemy eksploracyjne!



- ✓ Pytania?
- ✓ Uwagi?
- ✓ Sugestie?
- ✓ Życzenia?

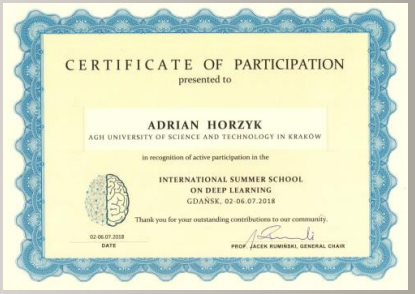
# Bibliografia i Literatura


1. Stanisław Osowski, *Metody i narzędzia eksploracji danych*, BTC, Legionowo, 2013.
2. Andrzej Łachwa, *Rozmyty świat zbiorów*, liczb, relacji, faktów, reguł i decyzji, Akademicka Oficyna Wydawnicza EXIT, Warszawa, 2001.
3. Leszek Rutkowski, *Metody i techniki sztucznej inteligencji*, PWN, Warszawa, 2012.
4. D. T. Larose, *Odkrywanie wiedzy z danych*. Wprowadzenie do eksploracji danych, PWN, Warszawa 2006.
5. Stanisław Osowski, *Sieci neuronowe do przetwarzania informacji*, Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa, 2013.
6. Nikola K. Kasabov, *Time-Space, Spiking Neural Networks and Brain-Inspired Artificial Intelligence*, In Springer Series on Bio- and Neurosystems, Vol 7., Springer, 2019.
7. Ian Goodfellow, Yoshua Bengio, Aaron Courville, *Deep Learning*, MIT Press, 2016, ISBN 978-1-59327-741-3 or PWN 2018.
8. Holk Cruse, *Neural Networks as Cybernetic Systems*, 2nd and revised edition
9. R. Rojas, *Neural Networks*, Springer-Verlag, Berlin, 1996.
10. *Convolutional Neural Network* (Stanford)
11. *Visualizing and Understanding Convolutional Networks*, Zeiler, Fergus, ECCV 2014
12. IBM: https://www.ibm.com/developerworks/library/ba-data-becomes-knowledge-1/index.html
13. NVIDIA: https://developer.nvidia.com/discover/convolutional-neural-network
14. Horzyk, A., *How Does Generalization and Creativity Come into Being in Neural Associative Systems and How Does It Form Human-Like Knowledge?*, Elsevier, Neurocomputing, Vol. 144, 2014, pp. 238 - 257, DOI: 10.1016/j.neucom.2014.04.046.
15. A. Horzyk, J. A. Starzyk, J. Graham, *Integration of Semantic and Episodic Memories*, IEEE Transactions on Neural Networks and Learning Systems, Vol. 28, Issue 12, Dec. 2017, pp. 3084 - 3095, 2017, DOI: 10.1109/TNNLS.2017.2728203.
16. A. Horzyk, J.A. Starzyk, *Multi-Class and Multi-Label Classification Using Associative Pulsing Neural Networks*, IEEE Xplore, In: 2018 IEEE World Congress on Computational Intelligence (WCCI IJCNN 2018), 2018, (in print).
17. A. Horzyk, J.A. Starzyk, *Fast Neural Network Adaptation with Associative Pulsing Neurons*, IEEE Xplore, In: 2017 IEEE Symposium Series on Computational Intelligence, pp. 339 -346, 2017, DOI: 10.1109/SSCI.2017.8285369.
18. A. Horzyk, K. Gołdon, *Associative Graph Data Structures Used for Acceleration of K Nearest Neighbor Classifiers*, LNCS, In: 27th International Conference on Artificial Neural Networks (ICANN 2018), 2018, (in print).
19. A. Horzyk, *Deep Associative Semantic Neural Graphs for Knowledge Representation and Fast Data Exploration*, Proc. of KEOD 2017, SCITEPRESS Digital Library, pp. 67 - 79, 2017, DOI: 10.13140/RG.2.2.30881.92005.
20. A. Horzyk, *Neurons Can Sort Data Efficiently*, Proc. of ICAISC 2017, Springer-Verlag, LNAI, 2017, pp. 64 - 74, ICAISC BEST PAPER AWARD 2017 sponsored by Springer.

**Adrian Horzyk**

horzyk@agh.edu.pl

Google: Horzyk



**Akademia Górniczo-Hutnicza im. St. Staszica w Krakowie**