Studia i Materiały Informatyki Stosowanej, Tom 1, Nr 1, 2009

NADESŁANE ARTYKUŁY

OBIEKTOWA IMPLEMENTACJA ALGORYTMU KLASTERYZACJI METDĄ K-ŚREDNICH

Mariusz Dramski

Uniwersytet Kazimierza Wielkiego
Instytut Techniki
ul.Chodkiewicza 30/ p.215, 85-064 Bydgoszcz
e-mail: mdramski@ukw.edu.pl

Streszczenie: *Algorytm klasteryzacji metodą k-średnich to jeden z najpopularniejszych sposobów służących do klasyfikacji danych przy użyciu metod sztucznej inteligencji. Otrzymane klastry mogą dalej posłużyć do budowy np. modeli neuronowych z wykorzystaniem dzwonów Gaussa (sieci RBF) czy rozmytych modeli Takagi-Sugeno. Niniejszy artykuł przedstawia implementację tego algorytmu w języku C++. Można tu znaleźć opis klasy, która może później posłużyć jako biblioteka do dowolnego programu napisanego w tym języku.*

Słowa kluczowe: *Klasteryzacja, metoda k-średnich, klasyfikacja danych, C++*

Object oriented implementation of k-means clustering algorithm

Abstarct: *K-means clustering algorithm is one of the most popular ways for data classification using artificial intelligence methods. Obtained clusters can be further used e.g. to build RBF networks or Takagi-Sugeno fuzzy models. This paper contains the implementation of this algorithm in C++ programming language. You can find there the description of the class, which can serve as a library in different programs written in C++.*

Keywords: *Clustering, k-means method, data classification, C++*

1. WSTĘP

Zadanie klasteryzacji polega na podzieleniu zbioru danych pomiarowych na pewne podzbiory, spełniające określone warunki. Takie podzbiory nazywamy klastrami. Najczęstszym stosowanym tutaj kryterium jest odległość w metryce euklidesowej. Otrzymane klastry mogą potem posłużyć do budowy lokalnych modeli systemów. Modele takie budujemy, kiedy nie interesuje nas przestrzeń całego systemu, bądź wówczas kiedy takie podejście jest wydajniejsze. Można również zbudować kilka modeli lokalnych, a za ich pomocą stworzyć jeden model globalny.

2. METODA K-ŚREDNICH

Jednym z najczęściej stosowanych metod klasteryzacji jest metoda k-średnich. Charakteryzuje się ona dość prostym aparatem matematycznym. Składa się ona z następujących kroków:

I. Przyjęcie klastrów startowych, ich środki wyznaczane są z reguły empirycznie, również na podstawie oceny wizualnej.

II. Przyporządkowanie próbek pomiarowych do poszczególnych klastrów na podstawie odległości euklidesowych d_{ij} poszczególnych próbek P_i od środków klastrów m_i.

$$d_{ij} = \|x_j - m_i\| = \sqrt{\sum_{l=1}^{2} (x_{l,j} - m_{xl,j})^2}$$

III. Określenie nowych środków klastrów, robi się to metodą skumulowaną na podstawie wzoru:

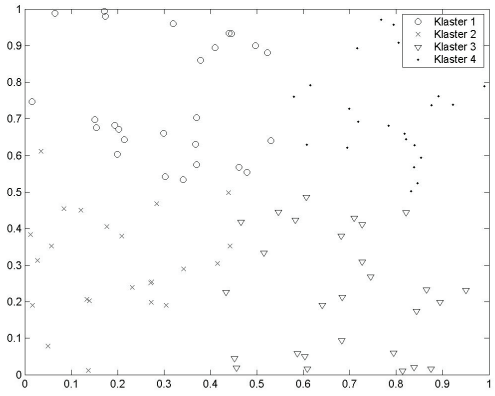
$$m_{xl,i}(1) = \frac{1}{N_i} \sum_{j=1}^{N_i} x_{lj}(0)$$

IV. Określenie wielkości przesunięcia klastrów Δm

$$\Delta m = \|m_i(0) - m_i(1)\|$$

V. Przyporządkowanie próbek pomiarowych do nowych klastrów.

VI. Określenie nowych środków klastrów; powyższe wszystkie operacje powtarza się aż do momentu, w którym przesunięcie środkow klastrów nie będzie przekraczało pewnej zadanej wartości progowej ε.



Rysunek. 1 Przykładowa ilustracja algorytmu klasteryzacji metodą k-średnich, źródło: opracowanie własne.

3. IMPLEMENTACJA ALGORYTMU W JĘZYKU C++

W celu zaimplementowania algorytmu klasteryzacji metodą k-średnich, zaprojektowano wzorzec klasy, który można przedstawić za pomocą poniższego diagramu UML:

Cluster
- NumberOfSamples: int - NumberOfClusters: int - Dimension: int - dm: float - NewCenters: **float - Epsilon: float + Data: **float + Centers: **float + ClusterNo: *int
- Distance(float *v1, float *v2, int Size): float - ClustersAssign(): void - CalculateNewCenters(): void - DeltaM(): void + Clusterize(): *int

Jak łatwo zauważyć, atrybutami publicznymi klasy `Cluster` są jedynie dane pomiarowe, współrzędne środków klastrów oraz wektor zawierający numery klastrów dla każdej kolejnej próbki.

Jedyną publiczną funkcją jest funkcja `Clusterize()`, która uruchamia całą procedurę obliczeniową.

Korzystanie z klasy `Cluster` jest bardzo proste. Na początku należy wprowadzić macierz, zawierającą dane pomiarowe (atrybut `Data`). Następnie podaje się pierwotne środki klastrów (atrybut `Centers`). Ostatnim krokiem jest uruchomienie funkcji `Clusterize()`.

Przykładowy kod źródłowy, wykorzystujący klasę `Cluster`, może zatem wyglądać następująco:

```
Cluster c(b1,b2,data,centers);
c.Clusterize();
```

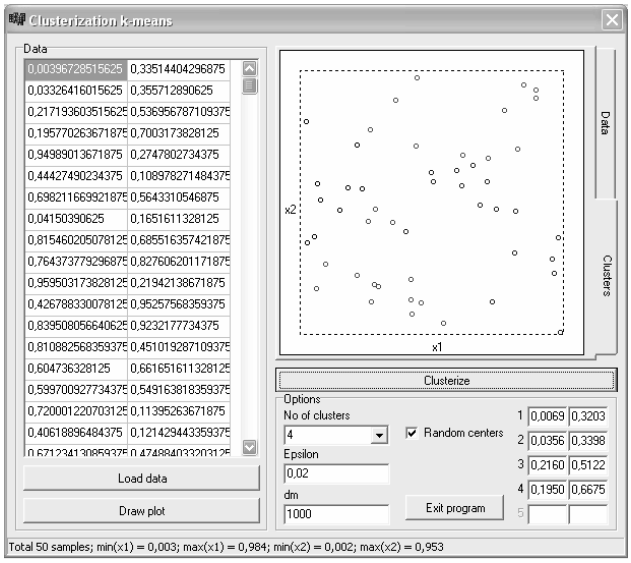
Wektor `b1` zawiera kolejno liczbę próbek, liczbę klastrów oraz liczbę wejść systemu. W wektorze `b2` zawarte są informacje na temat Δm oraz ε.

Wykonanie programu polegać będzie zatem na powołaniu do życia obiektu klasy `Cluster`, pamiętając o nadaniu mu stosownych wartości atrybutów i to już na poziomie konstruktora klasy. Nie istnieje potrzeba np. dalszej aktualizacji danych. Następną operacją jest uruchomienie funkcji `Clusterize()` na rzecz utworzonego obiektu. Funkcja ta korzysta z pozostałych funkcji klasy, które znajdują się w jej części prywatnej. Jej kod źródłowy wygląda następująco:

```
int* Cluster::Clusterize()
{
  ClustersAssign();
```

```
  while (dm > Epsilon)
  {
    CalculateNewCenters();
    for (int i=0; i<NumberOfClusters;
i++)
      for (int j=0; j<Dimension; j++)
        Centers[i][j]                =
NewCenters[i][j];
    DeltaM();
    ClustersAssign();
  }
  return ClusterNo;
}
```

Funkcja `Clusterize()` zwraca jako rezultat wektor `ClusterNo`, który zawiera informacje na temat klastra odpowiadającego każdej próbce ze zbioru danych pomiarowych (kolejny numer próbki to po prostu indeks pokazujący jej pozycję w wektorze).



Rysuek. 2 Interfejs przykładowej aplikacji wykorzystującej klasę Cluster.

Powyższy rysunek przedstawia interfejs przykładowej aplikacji wykorzystującej klasę `Cluster`. Jej celem jest zilustrowanie działania algorytmu klasteryzacji metodą k-średnich. Wynik klasteryzacji przedstawiony jest za pomocą kolorów. Każdemu klastrowi odpowiada inny kolor.

Taka aplikacja może służyć nie tylko do zademonstrowania działania algorytmu, ale może być także podstawą do tworzenia innych, bardziej skomplikowanych programów np. do modelowania systemów za pomocą sieci RBF.

4. PODSUMOWANIE I WNIOSKI

W niniejszym artykule przedstawiono algorytm klasteryzacji metodą k-średnich, a także jego implementację w języku C++. Nie zamieszczono kodu źródłowego niektórych funkcji, ale są one stosunkowo proste do odtworzenia, dlatego nie istnieje po prostu taka potrzeba. Dla zainteresowanych autor udostępni pełny kod źródłowy drogą mailową.

Klasteryzacja metodą k-średnich to stosunkowo prosty algorytm służący do klasyfikacji danych. Jak widać jest on również łatwy do implementacji.

Literatura

1. Broomhead S., Lowe D., „Multivariable fractional interpolation and adaptive network”, Complex Systems 2, 321-323, 1988
2. Dramski M., „Opracowanie metody modelowania systemów nieliniowych z wykorzystaniem sąsiedztwa okrężnego”, rozprawa doktorska, Politechnika Szczecińska, 2005
3. Gręboszz J., “Symfonia C++”, Oficyna Kalimach, Kraków 1999
4. Moody D., Darken J., “Fast learning in networks of locally-tuned processing units, Neural Computation 1, 281-294, 1988
5. Piegat A., “Modelowanie i sterowanie rozmyte”, Akademicka Oficyna Wydawnicza Exit, Warszawa 1999
6. Takagi T., Sugeno M., „Fuzzy identification of systems and its applications to modeling and control”, IEEE Transactions on Systems, Man, and Cybernetics 1985, vol. 15, No. 1, pp 116-132.