Informatyka, studia dzienne, I st. semestr VI

**Komputerowe systemy rozpoznawania** **2018/2019**
Prowadzący: dr inż. Marcin Kacprowicz poniedziałek, 14:10

Data oddania: _____ Ocena: _____

Justyna Hubert 210200
Karol Podlewski 210294

# Zadanie 1: Ekstrakcja cech, miary podobieństwa, klasyfikacja*

## 1. Cel

Celem zadania było stworzenie aplikacji do klasyfikacji tekstów metodą k-NN, korzystając z różnych sposób ekstrakcji wektorów cech oraz istniejących miar podobieństwa porównać kategorie do tych przypisanych przez aplikację.

## 2. Wprowadzenie

Zagadnieniem, jakim zajmowaliśmy się w ramach projektu jest klasyfikacja statystyczna, która jest rodzajem algorytmu statystycznego przydzielającego elementy do klas, bazując na cechach tych elementów. W ramach przeprowadzanego eksperymentu zaimplementowaliśmy klasyfikator k-najbliższych sąsiadów.

Algorytm k najbliższych sąsiadów, nazywany także algorytmem k-nn, należy do grupy algorytmów leniwych, czyli takich, które nie tworzą wewnętrznej reprezentacji danych uczących, lecz szukają rozwiązania dopiero w momencie pojawienia się wzorca testującego. Przechowuje wszystkie wzorce uczące, względem których wyznacza odległość wzorca testowego [1]. Metoda

* SVN: `https://github.com/hubjust/KSR`

k-nn wyznacza k sąsiadów, do których badany element ma najmniejszą odległość w danej metryce, a następnie wyznacza wynik w oparciu o najczęstszy element, wśród k najbliższych. W przypadku naszego projektu odległość definiujemy jako skalę podobieństwa tekstów.

W ramach zadania zostały użyte 2 metody ekstrakcji cech:

— Term frequency - metoda polegająca na zliczeniu częstości występowania danego słowa w dokumencie. Obliczana jest z poniższego wzoru:

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k nk, j}$$

— Inverse document frequency - metoda polegająca na wyznaczeniu, czy dane słowo występuje powszechnie we wszystkich dokumentach. Jest to logarytmicznie skalowana odwrotna część dokumentów zawierających wybrane słowo (uzyskana poprzez podzielenie całkowitej liczby dokumentów przez liczbę dokumentów zawierających ten termin). Obliczana jest z poniższego wzoru:

$$idf_i = \log \frac{|D|}{|\{d : t_i \in d\}|}$$

Do obliczenia odległości tekstów posłużyliśmy się 3 metrykami:

— metryka Euklidesowa - w celu obliczenia odległości $d_e(x, y)$ między dwoma punktami $x, y$ należy obliczyć pierwiastek kwadratowy z sumy drugich potęg różnic wartości współrzędnych o tych samych indeksach, zgodnie ze wzorem:

$$d_e(x, y) = \sqrt{(y_1 - x_1)^2 + \cdots + (y_n - x_n)^2}$$

— metryka uliczna (Manhattan, miejska) - w celu obliczenia odległości $d_e(x, y)$ między dwoma punktami $x, y$ należy obliczyć sumę wartości bezwzględnych różnic współrzędnych punktów $x$ oraz $y$, zgodnie ze wzorem:

$$d_m(x, y) = \sum_{k=1}^{n} |x_k - y_k|$$

— metryka Czebyszewa - w celu obliczenia odległości $d_e(x, y)$ między dwoma punktami $x, y$ należy obliczyć maksymalną wartość bezwzględnych różnic współrzędnych punktów $x$ oraz $y$, zgodnie ze wzorem:

$$d_{ch}(x, y) = \max_i |x_i - y_i|$$

# 3. Opis implementacji

Program został stworzony w języku C#. Graficzny interfejs użytkownika został stworzony przy wykorzystaniu Windows Presentation Foundation. Logika aplikacji została odseparowana od GUI, w zgodzie ze wzorcem projektowym Model-view-viewmodel (MVVM), poprzez implementacje trzech projektów (Logic, ViewModel i GUI).

## 3.1. Logic

Klasy Chebyshev, Euclidean oraz Manhattan odpowiadają za prawidłowe obliczenia odległości tekstów. Dziedziczą one z klasy abstrakcyjnej Metric.

Klasa Article odwzorowuje artykuły wczytane do programu. Przechowuje informacje o dokumencie takie jak: tytuł, tekst, tagi, przypisane tagi, wektor cech, odległość.

Klasa FeatureExtraction implementuje dwie metody ekstrakcji cech - term frequency oraz inverse document frequency.

Klasa FileReader odpowiada za poprawne wczytywanie plików do programu - wyselekcjoowanie wybranych przez nas informacji (tytuł, ciało dokumentu, przypisaną etykietkę) i na ich podstawie stworzenie obiektu klasy Article. Z wczytanego tekstu usuwane są słowa, które występują w podanej przez nas stop liście. Ten zabieg ma za zadanie wykluczyć terminy, które nie wnoszą kluczowych, dla nas, informacji. Następnie, ciało dokumentu zostaje poddane stemizacji, czyli usunięciu ze słowa końcówki fleksyjnej pozostawiając tylko rdzeń wyrazu.

Klasa KnnAlgorithm odpowiada za implementację algorytmu k-najbliższych sąsiadów. W tym miejscu wyliczane są wystąpienia słów w podanych dokumentach.

Klasa Sets odpowiedzialna jest za odpowiedni dobór danych testowych oraz treningowych.

Klasa CategoryCompatibilityChecker ma za zadanie sprawdzić, czy dany artykuł zawiera jeden tag z danej kategorii, oraz czy ten tag zawiera się w tagach branych pod uwagę.

## 3.2. GUI

Projekt GUI (graphical user interface) implementuje przejrzysty oraz łatwy w obsłudze graficzny interfejs użytkownika.

### 3.3. ViewModel

Projekt ViewModel ma za zadanie odseparować logikę programu od interfejsu graficznego.

Klasa MainViewModel przyjmuje dane wejściowe od użytkownika i reaguje na jego poczynania wywołując wybrane akcje z logiki programu oraz odpowiada za odświeżanie widoków w interfejsie graficznym.

Należy tu zamieścić krótki i zwięzły opis zaprojektowanych klas oraz powiązań między nimi. Powinien się tu również znaleźć diagram UML (diagram klas) prezentujący najistotniejsze elementy stworzonej aplikacji. Należy także podać, w jakim języku programowania została stworzona aplikacja.

## 4. Materiały i metody

Klasyfikacja tekstów została wykonana wszystkimi dostępnymi metodami ekstrakcji cech dla wszystkich trzech metryk. Dla każdego przypadku testowego dokonano klasyfikacji tekstu dla k $\in \{2, 3, 5, 7, 10, 15, 20\}$ najbliższych sąsiadów. Wyniki porównano z faktyczną etykietą danego artykułu. Za każdym razem zbiór treningowy stanowił 60% artykułów, zaś zbiór testowy 40%.

Klasyfikacja dotycząca lokalizacji przeprowadzana była jedynie na danych, których pole places przyjmowało jedną z wartości: west-germany, usa, france, uk, canada, japan.

Klasyfikacja dotycząca tematów przeprowadzana była jedynie na danych, które pole topics przyjmowało jedną z wartości: gold, cocoa, sugar, coffe, grain.

Klasyfikacja własnych tesktów przeprowadzana była na danych, których pole author przyjmowało jedną z wartości: taylor swift, macklemore, tweny one pilots, eminem, ed sheeran, black eyed peas.

## 5. Wyniki

### 5.1. Klasyfikacja tekstów Reutersa

Wyniki dla kolejnych eksperymentów przedstawiają tabele. Tabele 1-3 przedstawiają odpowiednio eksperymenty przeprowadzone dla metryki Euklidesa, ulicznej oraz Czebyszewa przy użyciu Term frequency, natomiast Tabele 4-6 przedstawiają alogiczne dane, ale dla drugiego sposobu ekstrakcji - Inverse document frequency.

| k | skuteczność(places)[%] | skuteczność(topics)[%] |
|---|---|---|
| 2 | 74.4 | 53.7 |
| 3 | 78.5 | 52.2 |
| 5 | 80.2 | 52.2 |
| 7 | 81.0 | 53.7 |
| 10 | 81.5 | 60.4 |
| 15 | 81.6 | 62.7 |
| 20 | 81.4 | 61.2 |

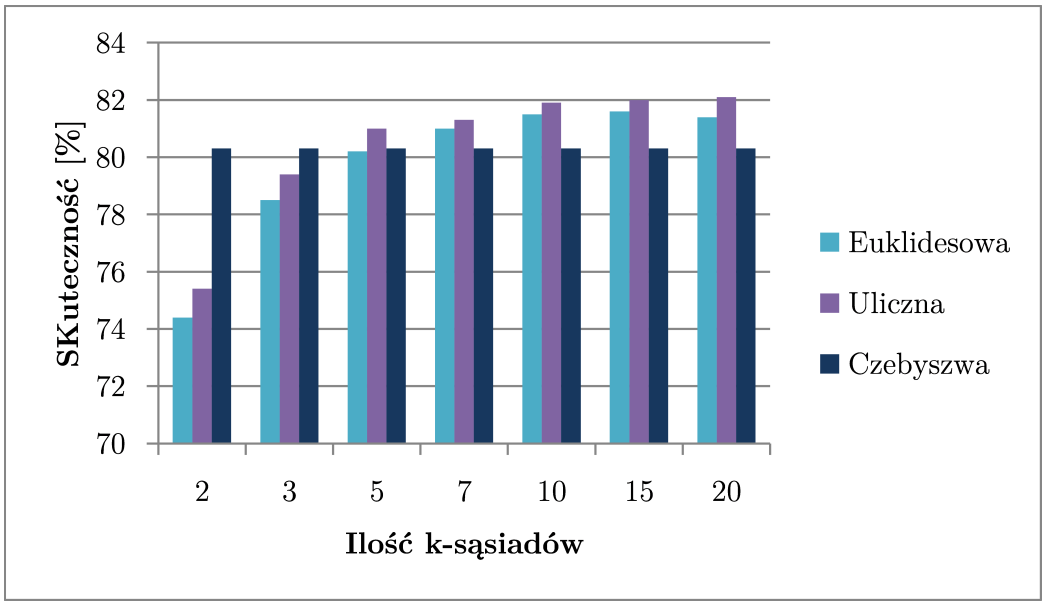
Tabela 1. Wyniki dla metryki Euklidesowej dla TF

| k | skuteczność(places)[%] | skuteczność(topics)[%] |
|---|---|---|
| 2 | 75.4 | 56.7 |
| 3 | 79.4 | 56.7 |
| 5 | 81.0 | 61.2 |
| 7 | 81.3 | 59.0 |
| 10 | 81.9 | 64.9 |
| 15 | 82.0 | 64.9 |
| 20 | 82.1 | 63.4 |

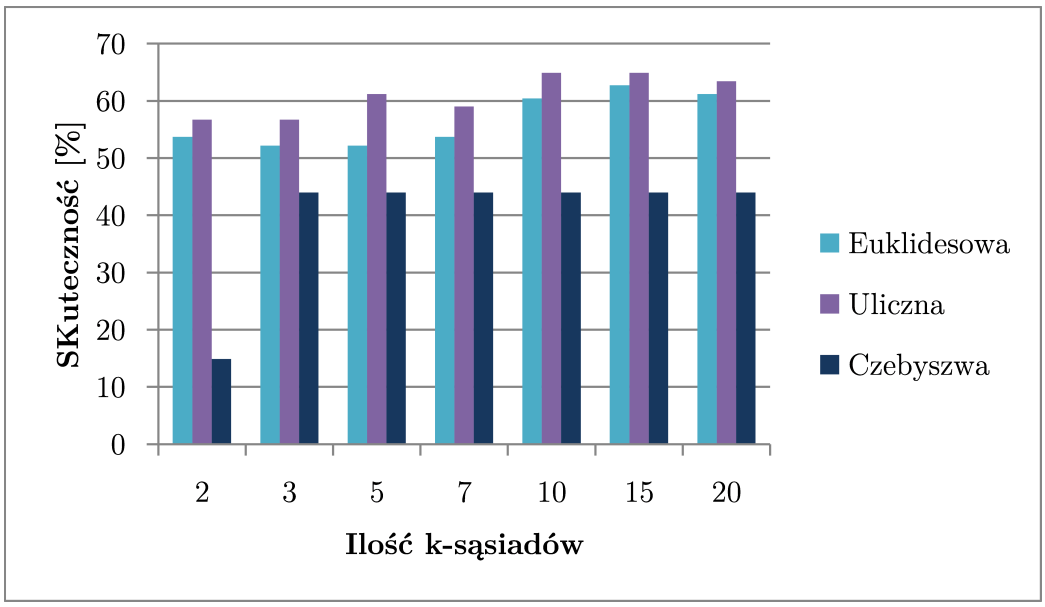
Tabela 2. Wyniki dla metryki ulicznej dla TF

| k | skuteczność(places)[%] | skuteczność(topics)[%] |
|---|---|---|
| 2 | 80.3 | 14.9 |
| 3 | 80.3 | 44.0 |
| 5 | 80.3 | 44.0 |
| 7 | 80.3 | 44.0 |
| 10 | 80.3 | 44.0 |
| 15 | 80.3 | 44.0 |
| 20 | 80.3 | 44.0 |

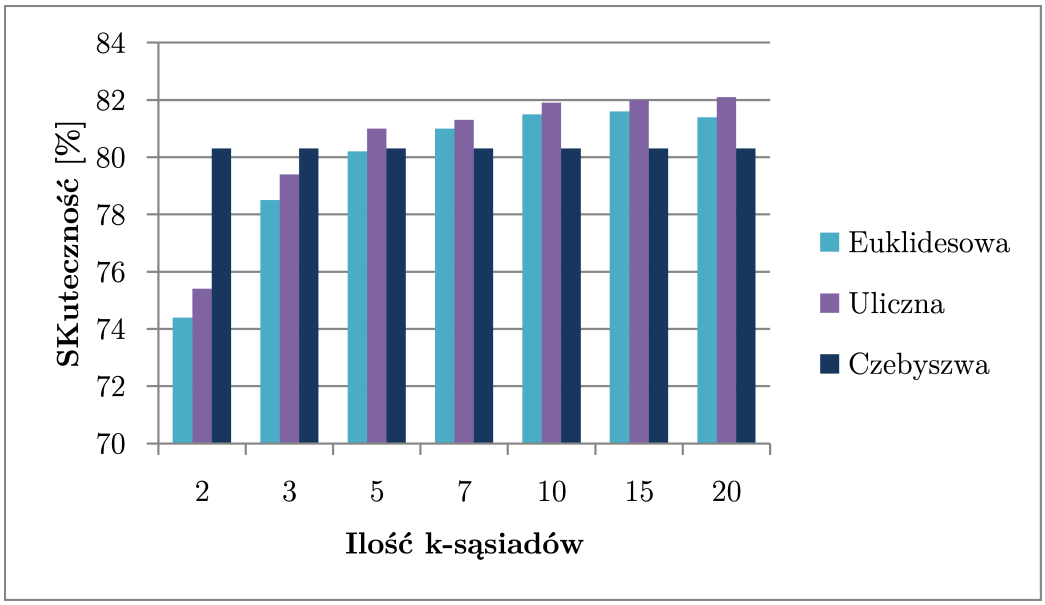
Tabela 3. Wyniki dla metryki Czebyszewa dla TF



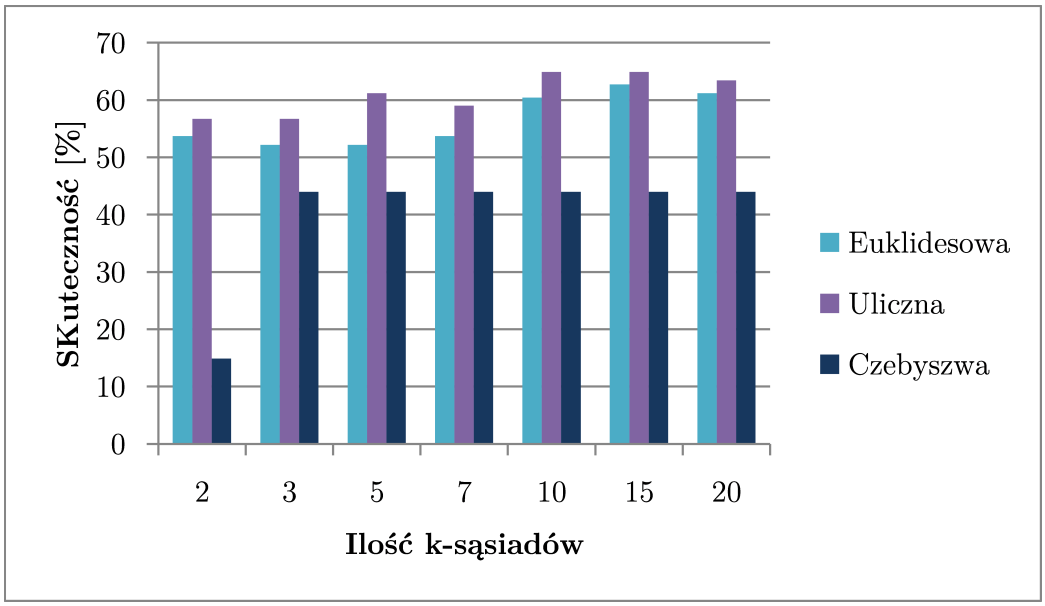

Rysunek 1. Dane z Tabel 1-3 dla kategorii places




Rysunek 2. Dane z Tabel 1-3 dla kategorii topics




Rysunek 3. Dane z Tabel 4-6 dla kategorii places




Rysunek 4. Dane z Tabel 4-6 dla kategorii topics

## 6. Dyskusja

Sekcja ta powinna zawierać dokładną interpretację uzyskanych wyników eksperymentów wraz ze szczegółowymi wnioskami z nich płynącymi. Najcenniejsze są, rzecz jasna, wnioski o charakterze uniwersalnym, które mogą być istotne przy innych, podobnych zadaniach. Należy również omówić i wyjaśnić wszystkie napotakane problemy (jeśli takie były). Każdy wniosek powinien mieć poparcie we wcześniej przeprowadzonych eksperymentach (odwołania

do konkretnych wyników). Jest to jedna z najważniejszych sekcji tego sprawozdania, gdyż prezentuje poziom zrozumienia badanego problemu.

## 7. Wnioski

W tej, przedostatniej, sekcji należy zamieścić podsumowanie najważniejszych wniosków z sekcji poprzedniej. Najlepiej jest je po prostu wypunktować. Znów, tak jak poprzednio, najistotniejsze są wnioski o charakterze uniwersalnym.

## Literatura

[1] Methods for the linguistic summarization of data - aplications of fuzzy sets and their extensions, Adam Niewiadomski, Akademicka Oficyna Wydawnicza EXIT, Warszawa 2008 1. http://home.agh.edu.pl/ horzyk/lectures/miw/KNN.pdf