Eksploracja tekstowych zasobów internetowych

Artur Gramacki

Uniwersytet Zielonogórski
Instytut Sterowania i Systemów Informatycznych

Plan wykładów

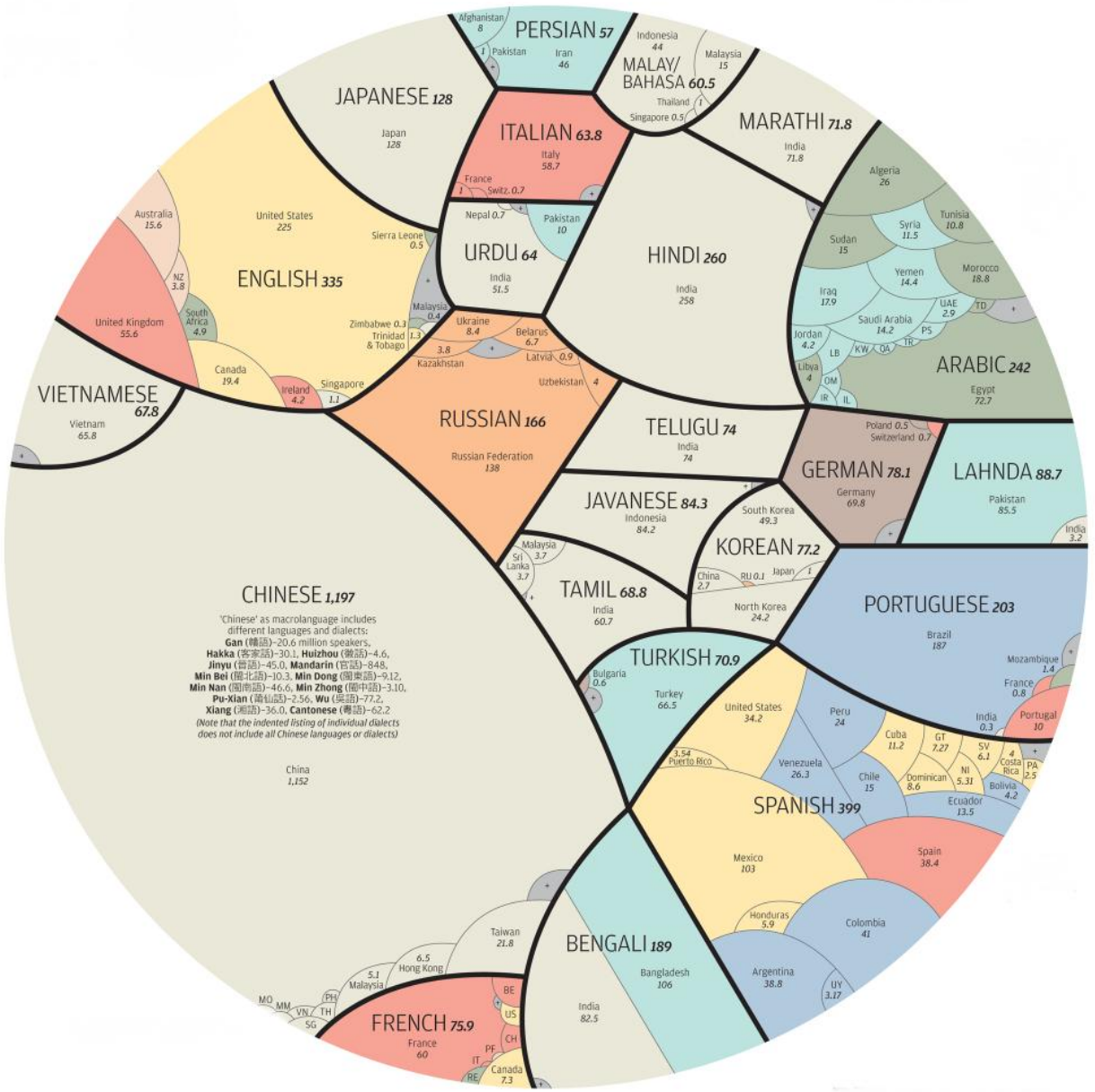
- Wektorowa reprezentacja tekstu, preprocessing, rangowanie dokumentów, informacja zwrotna (relevance feedback)
- Ukryte indeksowanie semantyczne (Latent Semantic Indexing (LSI)
  - Singular Value Decomposition (SVD)
  - rank-k approximation
  - automatyczne tworzenie podsumowań dokumentów
- Page Rank
- Grupowanie oraz klasyfikacja zasobów tekstowych
- Analiza wydźwięku (Sentimental Analysis)
- Systemy rekomendacyjne
- Modele wektorowe tekstów (word embeddings)

Uwagi wstępne

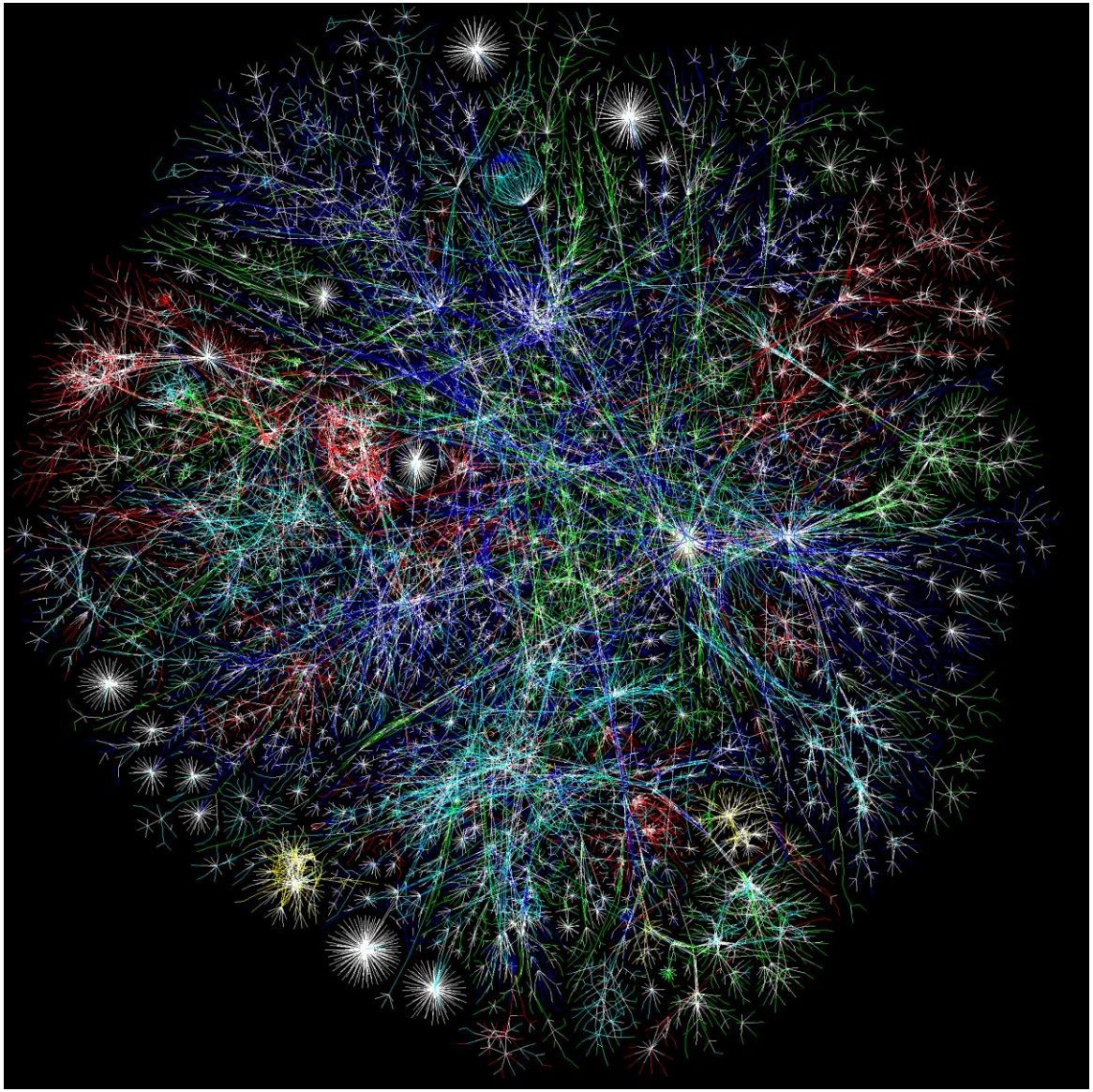
Uwagi wstępne

- Naturą internetu jest
  - przeogromny rozmiar zgromadzonych do tej port danych (nikt tak naprawdę nie zna jego rozmiaru, są tylko szacunki)
  - bardzo różnorodny (tekst, grafika statyczna, video, oprogramowanie, bazy danych itd.)
  - dynamicznie zmieniający się, w zasadzie ciągle się powiększa
    - ✓ ale też wiele danych „znika” (przynajmniej tak nam się wydaje, gdy jakieś dane były dostępne pod jakimś adresem a teraz ich tam nie znajdujemy)
  - wielojęzykowy
    - ✓ ludzie na świecie posługują się w sumie 7102 językami (a przynajmniej tyle do tej pory rozpoznano i opisano). Tylko 23 z nich uznaje się za języki ojczyste, którymi posługuje się przeszło 4 mld ludzi https://www.polityka.pl/tygodnikpolityka/ludzieistyle/1692254,1,ktory-jezyk-jest-najpotezniejszy-na-swiecie-nie-angielski.read
  - przewaga języka angielskiego. Ale nie w blogach (tam tylko ok. 1/3)

Uwagi wstępne



Uwagi wstępne



Popularne hasła

- IR – Information Retrieval
- KD – Knowledge Discovery
- DM – Data Mining
- WM – Web Mining
- TM – Text Mining
- DB – Bazy danych
- AI – Artificial Inteligence
- NN – Neural Networks
- CNN – Convolutiona Neural Networks
- ML – Machine Learning
- NLP – Natural Language Processing
- … i sporo innych

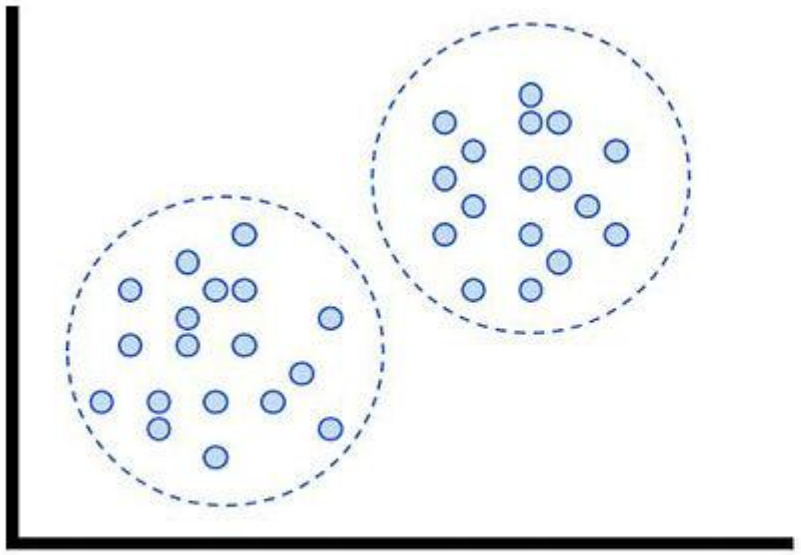
Główne zadania

- Wyszukiwanie informacji
- Grupowanie (Analiza skupień, *Unsupervised Learning, Cluster analysis*)
  - jest to metoda dokonująca grupowania elementów we względnie jednorodne klasy. Podstawą grupowania w większości algorytmów jest podobieństwo pomiędzy elementami – wyrażone przy pomocy funkcji (metryki) podobieństwa
- Klasyfikacja (*Classification*, *Supervised Learning*)
  - systematyczny podział elementów na klasy wykonywany według określonej zasady

Główne zadania

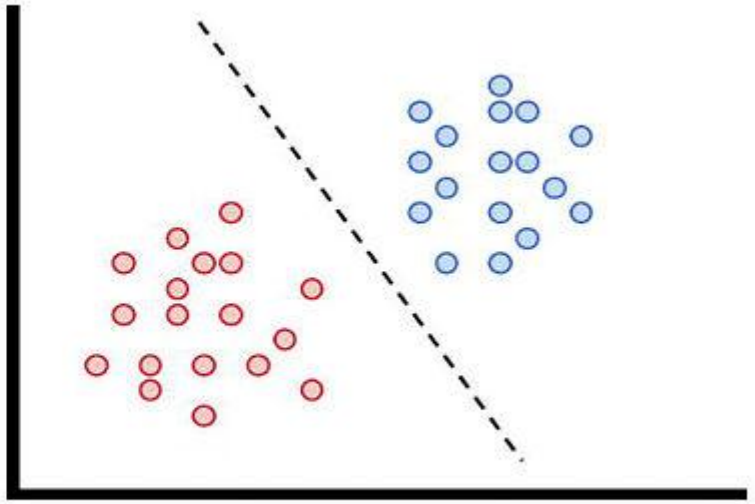
- Odkrywanie wiedzy z danych
- Wizualizacja danych
- Podsumowania danych
  - głównie tekstowych, ale też np. finansowych (eksploracja danych)
- Najprzeróżniejsze analizy
  - w dużej mierze oparte o statystykę
  - analiza sentymentu (straszne polskie tłumaczenie, lepiej analiza wydźwięku)
  - systemy rekomendacyjne
  - analiza współwystępowania
  - analiza mediów społecznościowych
  - itd.

Grupowanie a klasyfikacja



Clustering

- dane nie mają przypisanych etykiet
- obiekty leżące „blisko siebie” są łączone w grupy
- naturalne wykrywanie wzorców w danych
- uczenie nienadzorowane



Classification

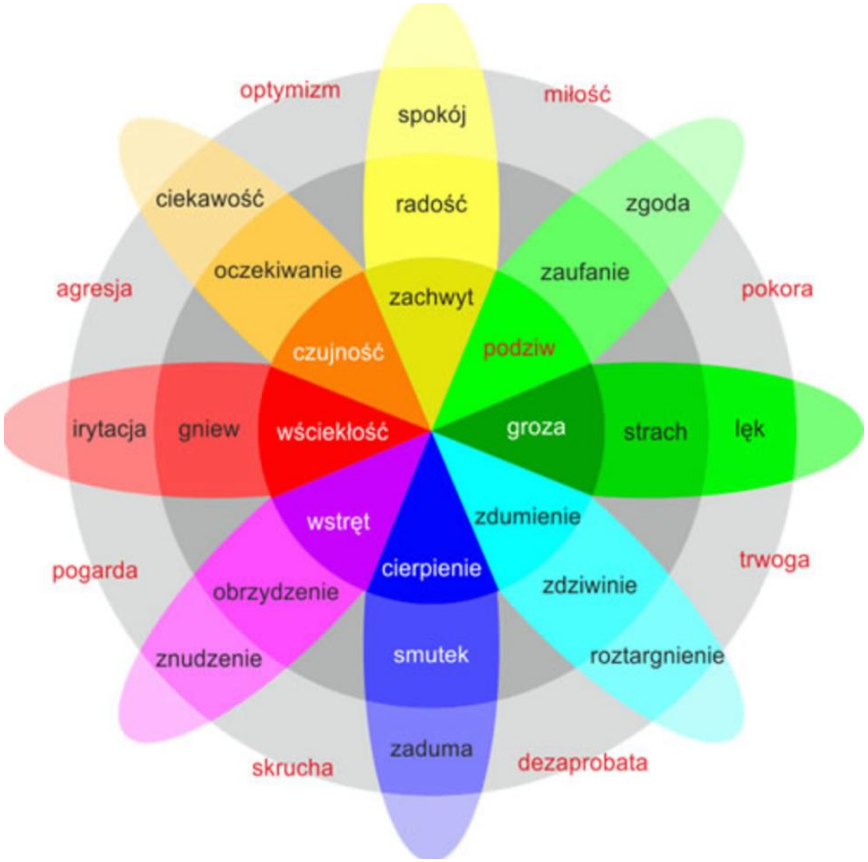
- dane mają przypisane etykiety
- poszukujemy pewnej reguły przypisywania nowych elementów do już istniejących klas
- uczenie nadzorowane

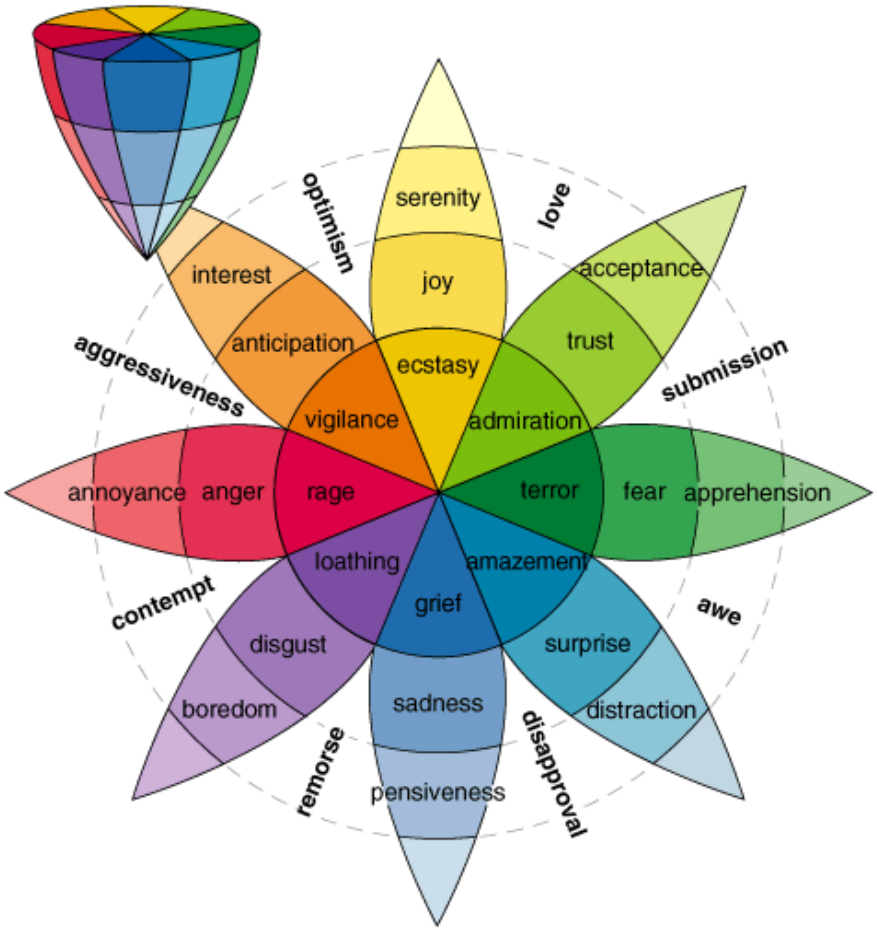
Analiza wydźwięku wypowiedzi

- Zastosowanie w analizie: komentarzy, blogów, portali społecznościowych, dyskusji, recenzji itp.
- Najprostrze: wydźwięk pozytywny, negatywny
- Bardziej złożone:
  - pozytywny, neutralny, negatywny
  - w skali np. od 1 do 5
- Koło emocji Plutchika:
  - W latach 1960-1980 amerykański psycholog Robert Plutchik opracował teorię emocji, w której zaproponował istnienie ośmiu podstawowych emocji, ewolucyjnie naturalnie rozwiniętych. Emocje te są wrodzone i bezpośrednio odnoszą się do zachowań adaptacyjnych, które mają na celu pomoc w przetrwaniu. Z nich wynikają wszystkie inne emocje.
  - https://pl.wikipedia.org/wiki/Teoria_emocji_Plutchika

Analiza wydźwięku wypowiedzi

- Koło emocji Plutchika





- Wymiar pionowy = intensywność emocji
  Wymiar poziomy = stopień podobieństwa
- Emocje przeciwległe na kole są emocjami przeciwnymi. Według Plutchika nie możemy doświadczać ich w tym samym czasie
- Emocje sąsiadujące na kole mogą być połączone w nowe emocje (bardziej złożone, subtelne w swej naturze), np. radość + zaufanie → miłość, wstręt + gniew → pogarda

Analiza wydźwięku wypowiedzi

- Koło emocji Plutchika
  - mieszanina dwóch dowolnych pierwotnych emocji jest nazywana przez Plutchika diadą
  - diady z przeciwstawnych emocji nie mogą zostać utworzone, gdyż zachodzi między nimi konflikt

Przykłady:

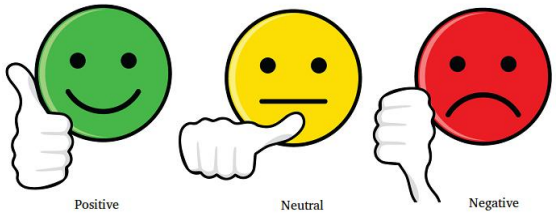
Podstawowe diady (często odczuwalne)	Drugorzędne diady (czasami odczuwalne)	Trzeciorzędne diady (rzadziej odczuwalne)	Przeciwieństwa
radość + zaufanie → miłość	radość + strach → poczucie winy	radość + zaskoczenie → zachwyt	radość + smutek → konflikt
zaufanie + strach → uległość	zaufanie + zaskoczenie → ciekawość	zaufanie + smutek → sentymentalizm	zaufanie + wstręt → konflikt,
strach + zaskoczenie → poruszenie	strach + smutek → rozpacz	strach + wstręt → wstyd	strach + gniew → konflikt
zaskoczenie + smutek → rozczarowanie	zaskoczenie + wstręt → szok	zaskoczenie + gniew → oburzenie	zaskoczenie + przeczuwanie → konflikt
smutek + wstręt → żal	smutek + gniew → cierpienie	smutek + przeczuwanie → pesymizm	-
wstręt + gniew → pogarda	wstręt + przeczuwanie → cynizm	wstręt + radość → patologia	-
gniew + przeczuwanie → agresja	gniew + radość → duma	gniew + zaufanie → dominacja	-
przeczuwanie + radość → optymizm	przeczuwanie + zaufanie → fatalizm	przeczuwanie + strach → lęk	-

Eksploracja tekstu, Text Mining

- Za Wikopedią PL
  - ogólna nazwa metod eksploracji danych służących do wydobywania danych z tekstu i ich późniejszej obróbki
- Za Wikipedią ENG
  - similar to text analytics, is the process of deriving high-quality information from text. It involves the discovery by computer of new, previously unknown information, by automatically extracting information from different written resources
- Cel maksymalistyczny: rozumienie tekstu, wraz ze wszelkimi niuansami językowymi
  - na razie dość daleko nam do tego
  - prostsze zadania są już całkiem dobrze rozwiązane

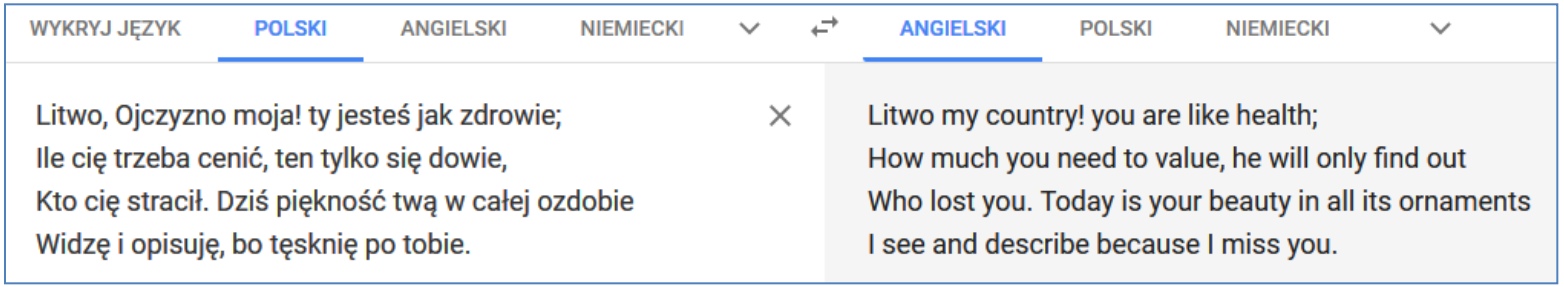
Rozumienie teksu

- Dlaczego chcemy, aby komputer umiał rozumieć tekst? Przykładowo:
  - automatyczne tłumaczenia
  - automatyczne prowadzenie dialogu
  - automatycznie udzielanie odpowiedzi na zadawane pytania
  - tworzenie sensownych streszczeń
  - ekstrakcja najważniejszych informacji
    - ✓ np. rozpoznanie daty i miejsca spotkania zapisanego w mailu i automatyczne wpisanie tej informacji do kalendarza Google
  - detekcja niechcianych wiadomości (spam)
  - analiza wydźwięku (analiza ~~sentymentu~~)
    - ✓ wypowiedź pozytywna
    - ✓ wypowiedź neutralna
    - ✓ wypowiedź negatywna

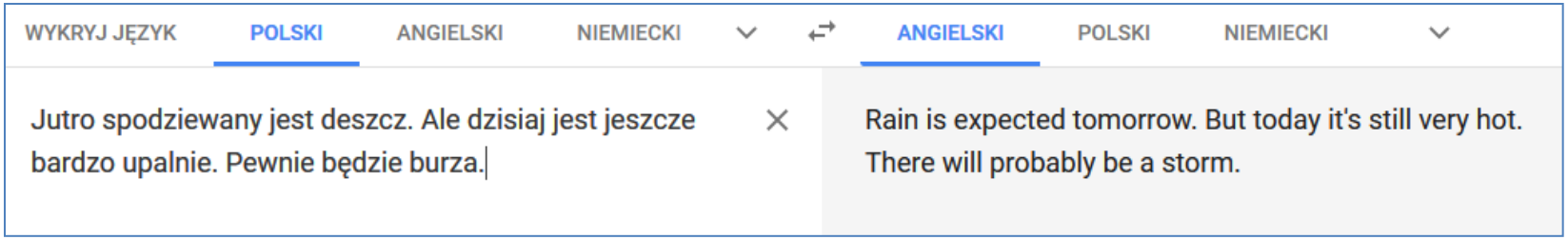


Rozumienie tekstu

- Szybki tekst tłumaczenia „bardzo trudnego tekstu”



- Szybki tekst tłumaczenia „dość łatwego tekstu”



Text Mining – problemy

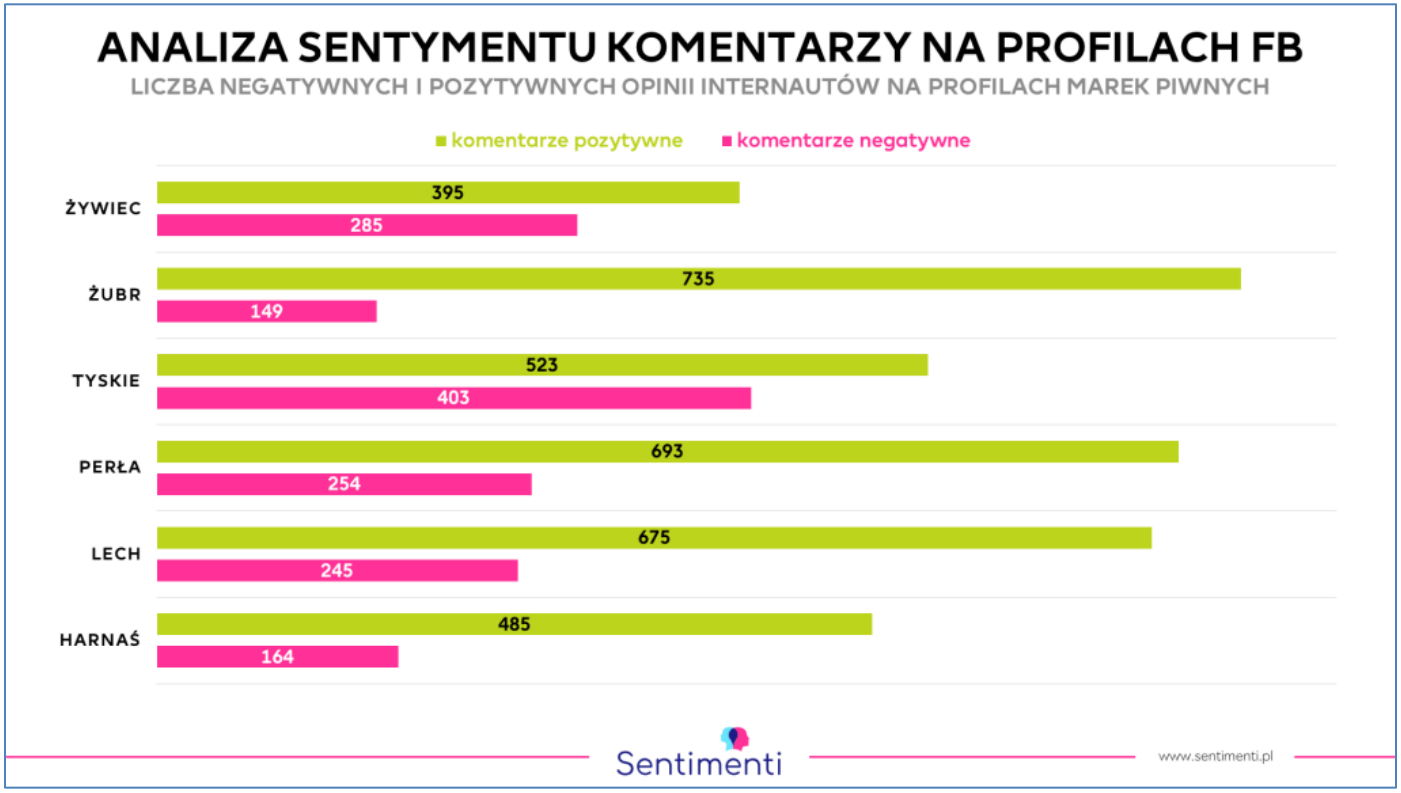
- Jest całe mnóstwo problemów, przykładowo:
  - idiomy: konstrukcja językowa, której znaczenie ma charakter swoisty, tj. nie daje się wyprowadzić ze znaczenia jej poszczególnych części składowych
    - ✓ *piąte koło u wozu* – osoba lub rzecz zawadzająca
    - ✓ *ręka rękę myje* – popieranie się przez ludzi w nieuczciwych sprawach (angielski idiom *one hand washes the other* oznacza wzajemną współpracę, nie ma negatywnej konotacji)
    - ✓ *włosy stają dęba* – być bardzo przestraszonym
    - ✓ *flaki z olejem* – coś wyjątkowo nudnego
  - kolokacje: związki wyrazowe wykazujące pewną ustaloną łączliwość, ale dające się rozumieć dosłownie
    - ✓ *odczuwać tęsknotę*, *mocna kawa, silny wiatr*, *na Węgrzech*, *we Francji*
  - żargon
  - synonimy i polisemia

Text Mining – problemy

- Jest całe mnóstwo problemów, przykładowo:
  - fleksja: odmiana wyrazów – wyróżnia się fleksję imienną (deklinację) oraz fleksję czasownika (koniugację)
    - ✓ deklinacja: odmiana wyrazów przez przypadki i liczby. W języku polskim mamy ich 7, ale np. w języku węgierskim jest ich ponad 20 :-)
    - ✓ koniugacja: odmiana czasownika przez osoby, czasy, tryby, strony, liczby, aspekty i inne kategorie gramatyczne
  - wielkość liter, znaki przystankowe, liczby
  - tzw. stop lista (że, który, jaki, przy, ale …)
  - lematyzacja, steeming

Text Mining

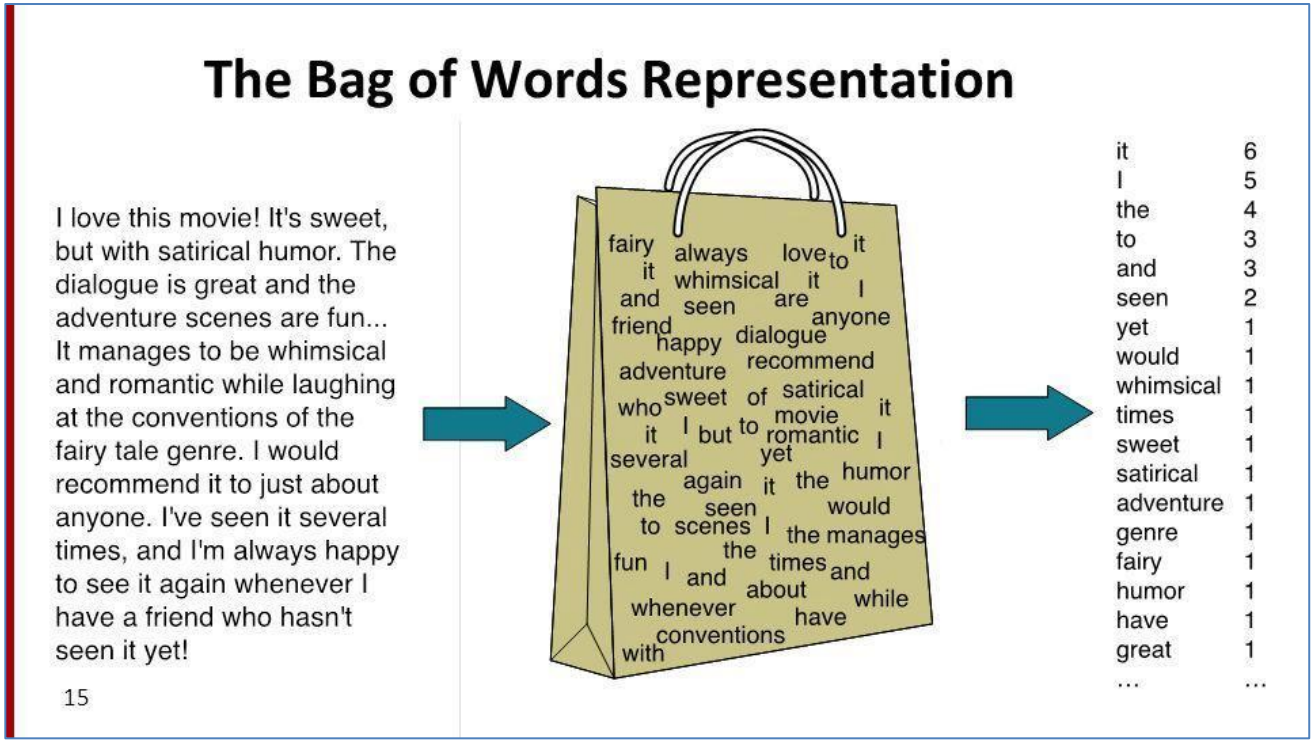
- Analiza wydźwięku wypowiedzi (komentarzy)



Wektorowa reprezentacja tekstu

Reprezentacja tekstu

- Bag-of-Words (worek słów), BOW



źródło: https://web.stanford.edu/~jurafsky/slp3/

Reprezentacja tekstu

- Bag-of-Words
  - bardzo uproszczona reprezentacja tekstów (trudno w sumie o prostszą)
  - całkowicie pomijana jest gramatyka
  - całkowicie pomijana jest kolejność słów
  - zachowane są częstości występowania poszczególnych wyrazów w tekście
  - nawet tak uproszczona reprezentacja tekstów daje bardzo duże możliwości automatycznej analizy tekstów

Operacje związane z analizą testu

- Identyfikacja języka
- Tokenizacja
- Stop lista
- Normalizacja
- Lematyzacja i/lub stemowanie (stemming, lemmatization)
- Rozpoznanie części mowy (Part-of-Speech Tagging, POS)
- Właściwe analizy

Tokenizacja

- Jeden z początkowych etapów przetwarzania tekstów. Tekst dzielony jest na tzw. tokeny (żetony). Token to w przybliżeniu „słowo”
- Separatory (czy usuwać ot tak, „z automatu”?)
  - `! @ # $ % ^ & * ( ) _ + = - , . ; ' : " < > / ?`
- Nie jest to zadanie, jakby mogło się wydawać, banale
  - biało-czerwony, matematyczno-fizyczny (myślnik)
  - Poland’s area, doesn’t, does not (apostrofy)
  - „Zielona Góra”, „Paragraf 22” (powieść amerykańskiego pisarza Josepha Hellera)
  - a.gramacki@issi.uz.zgora.pl (token niepodzielny)

Stoplista

- Nie ma jednej, powszechnie obowiązującej, są mniejsze i większe zestawy. Każdy język wypracował swoje indywidualne zestawy

a, aby, ach, acz, aczkolwiek, aj, albo, ale, ależ, ani, aż, bardziej, bardzo, bo, bowiem, by, byli, bynajmniej, być, był, była, było, były, będzie, będą, cali, cała, cały, ci, cię, ciebie, co, cokolwiek, coś, czasami, czasem, czemu, czy, czyli, daleko, dla, dlaczego, dlatego, do, dobrze, dokąd, dość, dużo, dwa, dwaj, dwie, dwoje, dziś, dzisiaj, gdy, gdyby, gdyż, gdzie, gdziekolwiek, gdzieś, i, ich, ile, im, inna, inne, inny, innych, iż, ja, ją, jak, jaka, jakaś, jakby, jaki, jakichś, jakie, jakiś, jakiż, jakkolwiek, jako, jakoś, je, jeden, jedna, jedno, jednak, jednakże, jego, jej, jemu, jest, jestem, jeszcze, jeśli, jeżeli, już, ją, każdy, kiedy, kilka, kimś, kto, ktokolwiek, ktoś, która, które, którego, której, który, których, którym, którzy, ku, lat, lecz, lub, ma, mają, mało, mam, mi, mimo, między, mną, mnie, mogą, moi, moim, moja, moje, może, możliwe, można, mój, mu, musi, my, na, nad, nam, nami, nas, nasi, nasz, nasza, nasze, naszego, naszych, natomiast, natychmiast, nawet, nią, nic, nich, nie, niech, niego, niej, niemu, nigdy, nim, nimi, niż, no, o, obok, od, około, on, ona, one, oni, ono, oraz, oto, owszem, pan, pana, pani, po, pod, podczas, pomimo, ponad, ponieważ, powinien, powinna, powinni, powinno, poza, prawie, przecież, przed, przede, przedtem, przez, przy, roku, również, sama, są, się, skąd, sobie, sobą, sposób, swoje, ta, tak, taka, taki, takie, także, tam, te, tego, tej, temu, ten, teraz, też, to, tobą, tobie, toteż, trzeba, tu, tutaj, twoi, twoim, twoja, twoje, twym, twój, ty, tych, tylko, tym, u, w, wam, wami, was, wasz, wasza, wasze, we, według, wiele, wielu, więc, więcej, wszyscy, wszystkich, wszystkie, wszystkim, wszystko, wtedy, wy, właśnie, z, za, zapewne, zawsze, ze, zł, znowu, znów, został, żaden, żadna, żadne, żadnych, że, żeby

Stoplista

- Język angielski
  - a, about, above, after, again, against, all, am, an, and, any, are, aren't, as, at, be, because, been, before, being, below, between, both, but, by, can't, cannot, could, couldn't, did, didn't, do, does, doesn't, doing, don't, down, during, each, few, for, from, further, had, hadn't, has, hasn't, have, haven't, having, he, he'd, he'll, he's, her, here, here's, hers, herself, him, himself, his, how, how's, i, i'd, i'll, i'm, i've, if, in, into, is, isn't, it, it's, its, itself, let's, me, more, most, mustn't, my, myself, no, nor, not, of, off, on, once, only, or, other, ought, our, ours, ourselves, out, over, own, same, shan't, she, she'd, she'll, she's, should, shouldn't, so, some, such, than, that, that's, the, their, theirs, them, themselves, then, there, there's, these, they, they'd, they'll, they're, they've, this, those, through, to, too, under, until, up, very, was, wasn't, we, we'd, we'll, we're, we've, were, weren't, what, what's, when, when's, where, where's, which, while, who, who's, whom, why, why's, with, won't, would, wouldn't, you, you'd, you'll, you're, you've, your, yours, yourself, yourselves

Stoplista

- Język arabski

Appendix A. General Stoplist

انها	الغير	اولئك	أمامنا	أى	إنكم	باللتين	بأى
اثناء	القادم	اي	أمامه	أي	إنكما	باللذان	بأي
اجل	اللاتي	اياه	أمامها	أياً	إنما	باللذين	بأياً
احدا	اللاحق	ايضا	أمامهم	أيان	إننا	باللواتي	بأية
احدى	اللتان	اين	أمامهما	أية	إنني	بالنسبة	بأيها
احيانا	اللتين	ايها	أمامهن	أيضا	إنه		بأيهم
اخرى	اللذان	آخر	أمامي	أيضاً	إنها	بامكان	بأيهما
اخيرا	اللذين	أبدا	أن	أين	إنهم	بان	بأيهن
اذ		أثناء	أنا	أينما	إنهما	بانه	بإحدى
اذا	اللواتي	أجل	أنت	أيها	إنهن	باولئك	بإذا
اذن	المقبل	أحد	أنتم	أيهم	إني	بآخر	بإلا
ازاء	الممكن	أحياناً	أنتما	أيهما	إياك	بأحد	بإياك
استمرا	المنصر	أخرى	أنتن	أيهن	إياكم	بأشياء	بإياكم
ر	م	أخيرا	أنك	إحدى	إياكما	بأقل	بإياكما
اصبح	النحو	أخيراً	أنكم	إذ	إياكن	بألا	بإياكن

Normalizacja

- Proces przetwarzania tekstów, nadający mu spójną formę, ułatwiającą dalszą interpretację
  - zmiana wielkości liter (na małe lub wielkie)
  - normalizacja skrótów
  - normalizacja wyrażeń numerycznych
  - normalizacja znaków specjalnych
  - zmiana znaków interpunkcyjnych
  - usuwanie (lub zmienianie) znaków diakrytycznych

źródło: https://pl.wikipedia.org/wiki/Normalizacja_tekstu

Normalizacja

- Przykłady
  - Zam. na os. Jana III Sobieskiego 45A/2
  - Zamieszkały na osiedlu Jana trzeciego Sobieskiego czterdzieści pięć A przez dwa
  - 7 IV odbędzie się 4. Olimpiada Matematyczna dla dzieci w wieku od 11-16 lat
  - Siódmego kwietnia odbędzie się czwarta Olimpiada Matematyczna dla dzieci w wieku od jedenastu do szesnastu lat
  - 1000$
  - tysiąc dolarów

źródło: https://pl.wikipedia.org/wiki/Normalizacja_tekstu

Lematyzacja

- Sprowadzanie danego słowa do jego formy podstawowej (hasłowej), która reprezentuje dany wyraz. Można wówczas traktować różne słowa, jako te samo słowo
  - jestem, jesteś, jesteśmy, są → być
  - kotów, koty, kotami → kot
- W przypadku czasownika będzie do bezokolicznik, w przypadku rzeczownika – mianownik liczby pojedynczej, itd.
- Do wykonania tego zadania potrzebny jest słownik lub rozbudowany zestaw reguł fleksyjnych dla danego języka

Stemming

- Brak sensownego polskiego tłumaczenia
- Prostsza wersja lematyzacji
- Jest to proces usunięcia ze słowa końcówki fleksyjnej pozostawiając tylko temat wyrazu (rdzeń znaczeniowy)
  - rdzeń niekoniecznie musi być „legalnym” słowem, np. computable, computation, computing, computed, computational → comput
  - inny przykład:
    connection, connections, connected, connecting → connect
- Dla języka angielskiego: względnie prosty algorytm Portera
  - algorytm regułowy, np. usuwanie koncówek *ed, *ing
- Dla języka polskiego zadanie to jest dużo bardziej złożone
  - np. https://bnosac.github.io/udpipe/en/

Stemming – Algorytm Portera

- https://tartarus.org/martin/PorterStemmer/def.txt

```
Step 1a

    SSES -> SS                      caresses  ->  caress
    IES  -> I                       ponies    ->  poni
                                    ties      ->  ti
    SS   -> SS                      caress    ->  caress
    S    ->                         cats      ->  cat
```

```
Step 1b

    (m>0) EED -> EE                 feed      ->  feed
                                    agreed    ->  agree
    (*v*) ED  ->                    plastered ->  plaster
                                    bled      ->  bled
    (*v*) ING ->                    motoring  ->  motor
                                    sing      ->  sing
```

```
Step 1c

    (*v*) Y -> I                    happy     ->  happi
                                    sky       ->  sky
```

```
Step 2

    (m>0) ATIONAL ->  ATE           relational     ->  relate
    (m>0) TIONAL  ->  TION          conditional    ->  condition
```

```
Step 3

    (m>0) ICATE ->  IC              triplicate     ->  triplic
    (m>0) ATIVE ->                  formative      ->  form
```

```
Step 4

    (m>1) AL    ->                  revival        ->  reviv
    (m>1) ANCE  ->                  allowance      ->  allow
```

```
Step 5a

    (m>1) E     ->                  probate        ->  probat
                                    rate           ->  rate
    (m=1 and not *o) E ->           cease          ->  ceas
Step 5b

    (m > 1 and *d and *L) -> single letter
                                    controll       ->  control
                                    roll           ->  roll
```

Stemming – Algorytm Portera

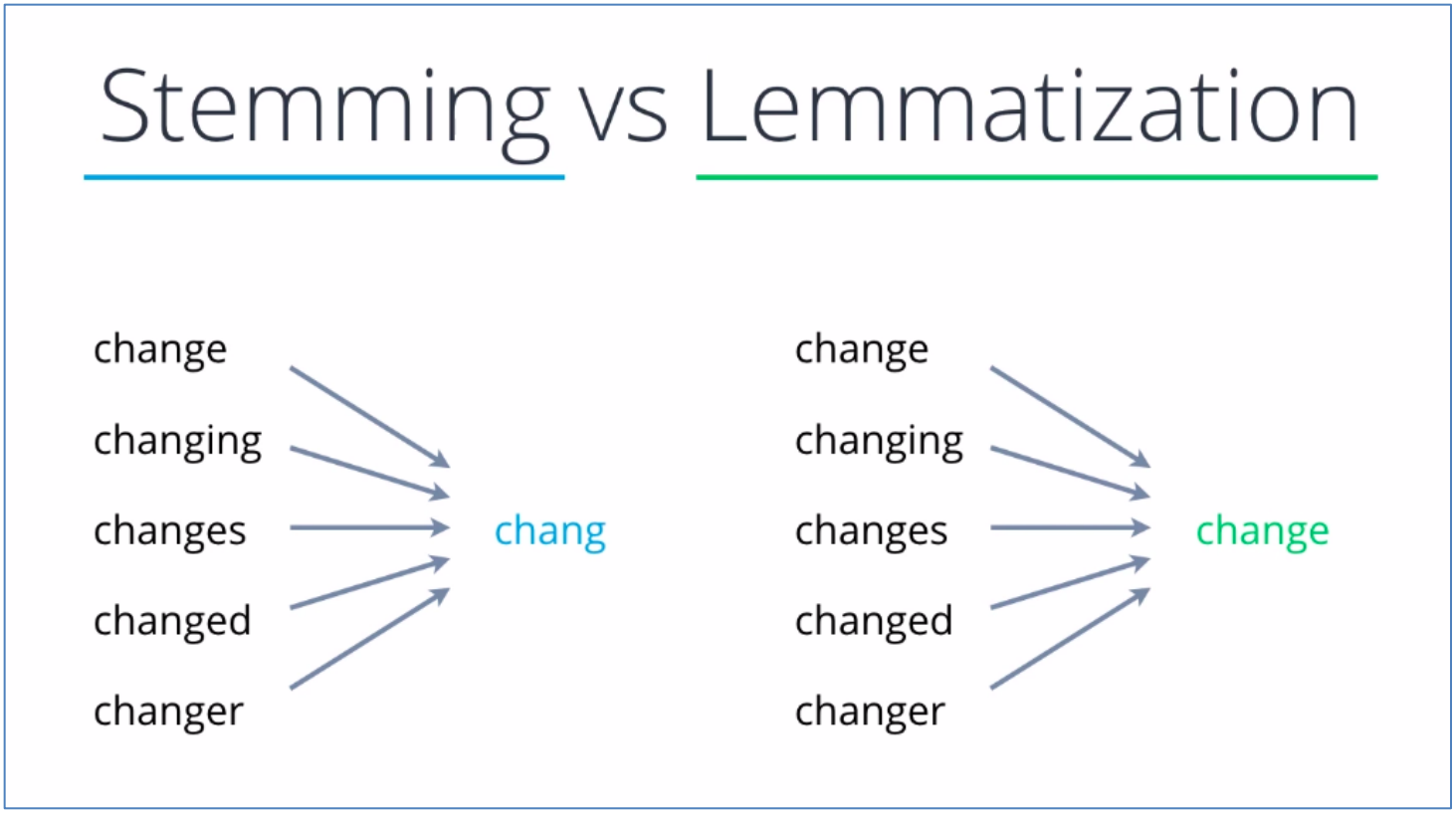
- Przykład działania
  - http://9ol.es/porter_js_demo.html

R is a programming language and free software environment for statistical computing and graphics supported by the R Foundation for Statistical Computing. The R language is widely used among statisticians and data miners for developing statistical software and data analysis.

```
R is a programming language and free software environment for statistical
computing and graphics supported by the R Foundation for Statistical Comput
ing The R language is widely used among statisticians and data miners for
developing statistical software and data analysis
```

```
R is a program languag and free softwar environ for statist comput and
graphic support by the R Foundat for Statist Comput The R languag is wide
us among statistician and data miner for develop statist softwar and data
analysi
```

Stemming a lematyzacja



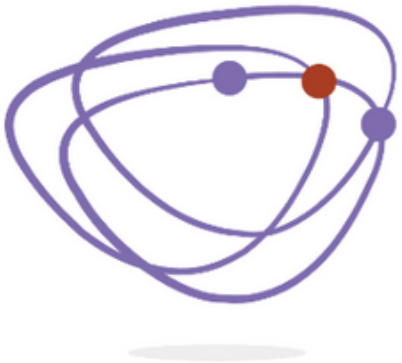
źródło: https://towardsdatascience.com/stemming-vs-lemmatization-8b30c1d3795a

Lematyzacja w R

- https://wilkowski.org/notka/1587

Prosta i szybka lematyzacja w R z wykorzystaniem usług Clarin

Opublikowano: 17.01.2018

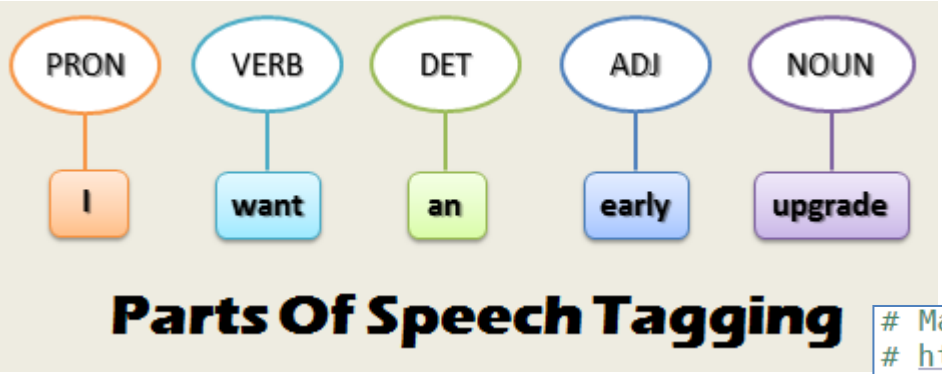


CENTRUM TECHNOLOGII JĘZYKOWYCH **CLARIN-PL**

Konsorcjum Clarin udostępnia interfejs programistyczny, pozwalający na zdalne przetwarzanie dokumentów tekstowych. Można swobodnie wykorzystać te usługi do pracy z tekstem w R. Oto prosty sposób na sprowadzanie do wspólnej, podstawowej postaci wyrazów z wektora (czyli – w dużym skrócie – wskazanie lematów, podstawowych form hasłowych). Dzięki temu jesteśmy w stanie np. przygotować dobrą statystykę wyrazów czy wygenerować poprawną chmurę słów kluczowych dla tekstu.

Rozpoznanie części mowy (Part-of-Speech Tagging, POS)

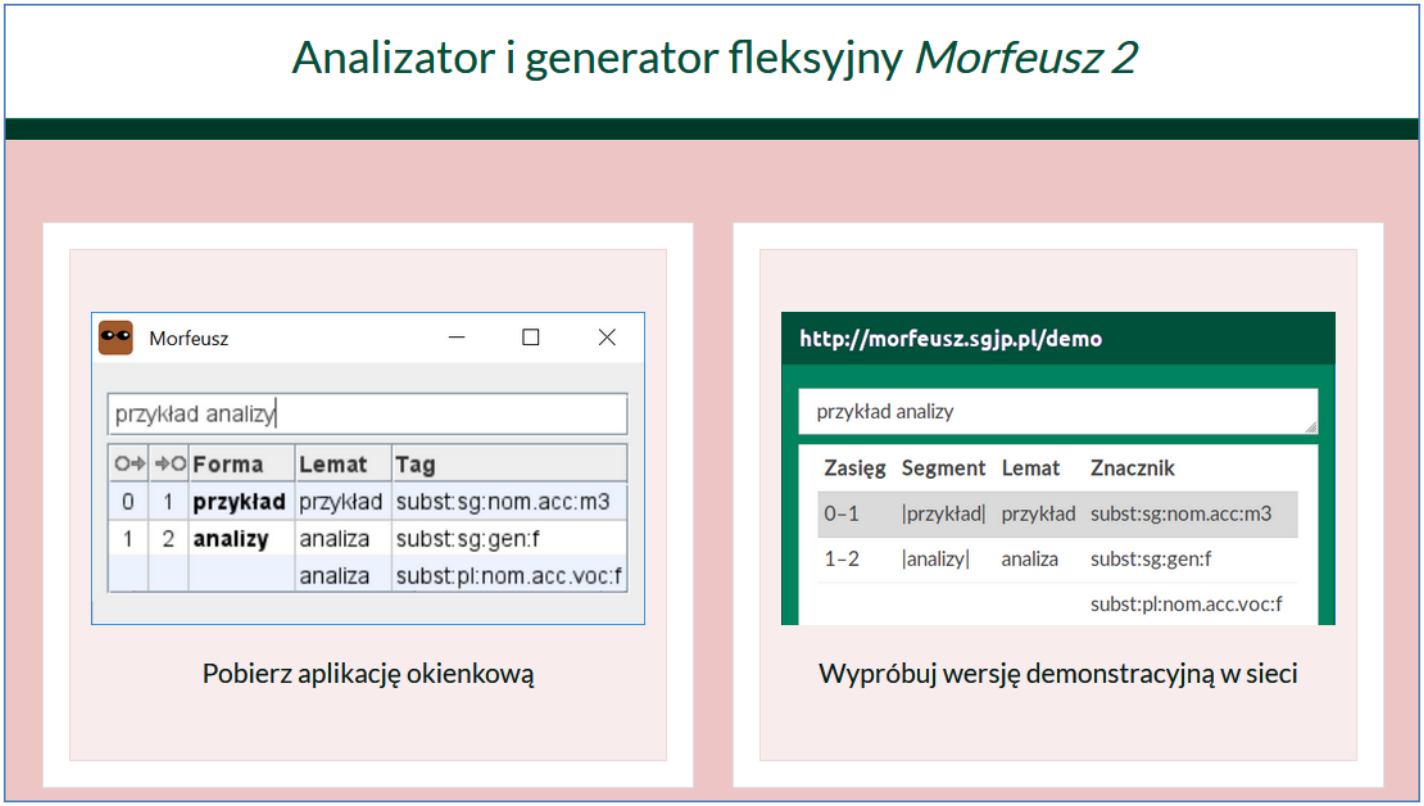
- Wiedza o części mowy bardzo wzbogaca rozumienie tekstu
- Zwykle POS wykonywany jest w ramach lematyzacji




```
# Mały słowniczek:
# https://www.ang.pl/gramatyka/czesci-mowy
# ---------------
# verb (czasownik)
# noun (rzeczowniki)
# adjective (przymiotnik)
# adverb (przysłówek)
# pronoun (zaimek)
# article (przedimek)
# preposition (przyimek)
# conjunction (spójnik)
# numeral (liczebnik)
# interjection (wykrzyknik, wtrącenie)
```

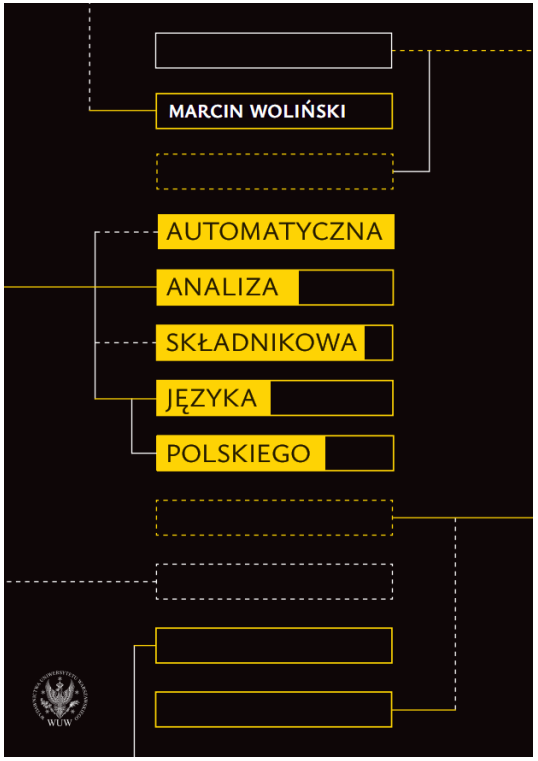
Rozpoznanie części mowy (Part-of-Speech Tagging, POS)

- Morfeusz R: http://sgjp.pl/morfeusz/morfeusz.html



Rozpoznanie części mowy (Part-of-Speech Tagging, POS)

- Warto zapoznać się z książką:
  Marcin Woliński. Automatyczna analiza składnikowa języka polskiego. Wydawnictwa Uniwersytetu Warszawskiego, Warsaw, 2019, aby mieć obraz, jak trudnym zadaniem jest lematyzacja dla języka polskiego
  - https://www.wuw.pl/data/include/cms/Automatyczna_analiza_skladnikowa_Wolinski_Marcin_2019.pdf



Rozpoznanie części mowy (Part-of-Speech Tagging, POS)

- Morfeusz 2 w działaniu
  - *Jutro spodziewany jest intensywny opad deszczu, mogą też pojawiać się lokalnie burze*
  - Na kolejnych slajdach
    - ✓ wynik analizy morfologicznej
    - ✓ wynik tagowania

Wynik analizy morfologicznej

Zasięg	Segment	Lemat	Znacznik	Pospolitość	Kwalifikatory
0–1	|Jutro|	jutro:d	adv		
		jutro:s	subst:sg:nom.acc.voc:n:ncol	nazwa_pospolita	
1–2	|spodziewany|	spodziewać	ppas:sg:acc:m3:imperf:aff		
			ppas:sg:nom.voc:m1.m2.m3:imperf:aff		
		spodziewany	adj:sg:acc:m3:pos		*przest.*
			adj:sg:nom.voc:m1.m2.m3:pos		*przest.*
2–3	|jest|	być	fin:sg:ter:imperf		
3–4	|intensywny|	intensywny	adj:sg:acc:m3:pos		
			adj:sg:nom.voc:m1.m2.m3:pos		
4–5	|opad|	opad	subst:sg:nom.acc:m3	nazwa_pospolita	
5–6	|deszczu|	deszcz	subst:sg:gen:m3	nazwa_pospolita	
			subst:sg:loc:m3	nazwa_pospolita	
			subst:sg:voc:m3	nazwa_pospolita	
6–7	|,|	,	interp		
7–8	|mogą|	móc	fin:pl:ter:imperf		
8–9	|też|	tenże	adj:pl:acc:m2.m3.f.n:pos		
			adj:pl:nom.voc:m2.m3.f.n:pos		
		też	part		
9–10	|pojawiać|	pojawiać	inf:imperf		
10–11	|się|	się	part		
11–12	|lokalnie|	lokalnie	adv:pos		
12–13	|burze|	burza	subst:pl:nom.acc.voc:f	nazwa_pospolita	
		bura	subst:sg:dat.loc:f	nazwa_pospolita	

Wynik tagowania

Zasięg	Segment	Lemat	Znacznik	Pospolitość	Kwalifikatory	Prawdopodobieństwo	Koniec zdania
0–1	Jutro	jutro:d	adv			1.0000	
1–2	spodziewany	spodziewany	adj:sg:nom:m3:pos		*przest.*	0.9999	
2–3	jest	być	fin:sg:ter:imperf			1.0000	
3–4	intensywny	intensywny	adj:sg:nom:m3:pos			1.0000	
4–5	opad	opad	subst:sg:nom:m3	nazwa_pospolita		1.0000	
5–6	deszczu	deszcz	subst:sg:gen:m3	nazwa_pospolita		1.0000	
6–7	,	,	interp			1.0000	
7–8	mogą	móc	fin:pl:ter:imperf			1.0000	
8–9	też	też	part			1.0000	
9–10	pojawiać	pojawiać	inf:imperf			1.0000	
10–11	się	się	part			1.0000	
11–12	lokalnie	lokalnie	adv:pos			1.0000	
12–13	burze	burza	subst:pl:nom:f	nazwa_pospolita		1.0000	●

morfologik-tools

- https://repo1.maven.org/maven2/org/carrot2/morfologik-tools/1.10.0/
  - **`java -jar morfologik-tools-1.10.0-standalone.jar plstem -i plik.input -o plik.output -ie utf8 -oe utf8`**
- *Jutro spodziewany jest intensywny opad deszczu, mogą też pojawiać się lokalnie burze*

```
jutro       jutro       adv+subst:sg:acc:n2+subst:sg:nom:n2+subst:sg:voc:n2
spodziewany spodziewany adj:sg:acc:m3:pos+adj:sg:nom.voc:m1.m2.m3:pos
spodziewany spodziewać  ppas:sg:acc:m3:imperf:aff+ppas:sg:nom.voc:m1.m2.m3:imperf:aff
jest        być         verb:fin:sg:ter:imperf:nonrefl
intensywny  intensywny  adj:sg:acc:m3:pos+adj:sg:nom.voc:m1.m2.m3:pos
opad        opad        subst:sg:acc:m3+subst:sg:nom:m3
deszczu     deszcz      subst:sg:gen:m3
deszczu     deszcz      subst:sg:loc:m3
deszczu     deszcz      subst:sg:voc:m3
mogą        móc         verb:fin:pl:ter:imperf:nonrefl
też         też         qub
też         tenże       adj:pl:acc:m2.m3.f.n1.n2.p2.p3:pos+adj:pl:nom.voc:m2.m3.f.n1.n2.p2.p3:pos
pojawiać    pojawiać    verb:inf:imperf:refl
się         się         siebie:acc:nakc+siebie:gen:nakc+subst:sg:nom:n2
lokalnie    lokalnie    adv:pos
burze       burza       subst:pl:acc:f+subst:pl:nom:f+subst:pl:voc:f
burze       bura        subst:sg:dat:f+subst:sg:loc:f
```

morfologik-tools

- Kolejny przykład, Inwokacja

Litwo! Ojczyzno moja! ty jesteś jak zdrowie.
Ile cię trzeba cenić, ten tylko się dowie,
Kto cię stracił. Dziś piękność twą w całej ozdobie
Widzę i opisuję, bo tęsknię po tobie.

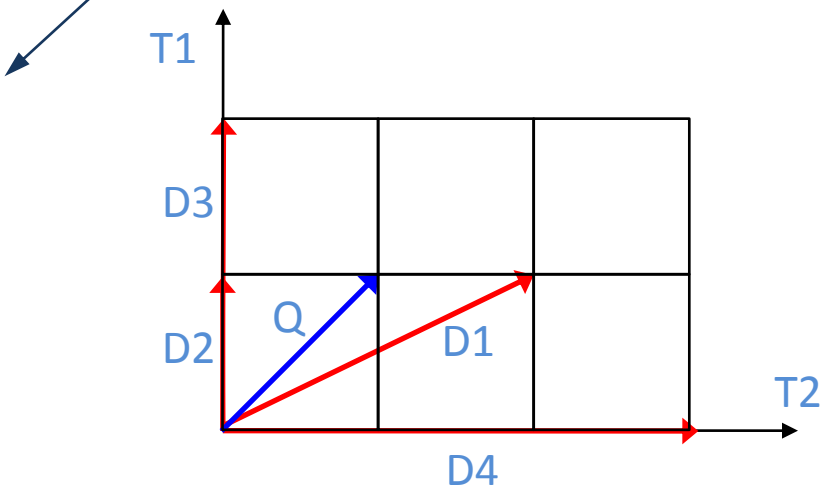
```
litwo       litwa       subst:sg:voc:f
ojczyzno    ojczyzna    subst:sg:voc:f
moja        moja        subst:sg:nom:f+subst:sg:voc:f
moja        mój         adj:sg:nom.voc:f:pos
ty          ty          ppron12:sg:nom:m1.m2.m3.f.n1.n2:sec+ppron12:sg:voc:m1.m2.m3.f.n1.n2:sec
jesteś      być         verb:fin:sg:sec:imperf:nonrefl
jak         jak         adv+adv:pos+conj+prep:nom+subst:sg:nom:m2
jak         jaka        subst:pl:gen:f
zdrowie     zdrowie     subst:sg:acc:n2+subst:sg:nom:n2+subst:sg:voc:n2
ile         ile         adv+num:pl:nom.acc.voc:m2.m3.f.n1.n2.p1.p2:rec+num:pl:nom.acc:m1.m2.m3.f.n1.n2.p1.p2:rec
ile         ił          subst:sg:loc:m3+subst:sg:voc:m3
cię         ty          ppron12:sg:acc:m1.m2.m3.f.n1.n2:sec:nakc+ppron12:sg:gen:m1.m2.m3.f.n1.n2:sec:nakc
trzeba      trzeba      pred
cenić       cenić       verb:inf:imperf:refl.nonrefl
ten         ten         adj:sg:acc:m3:pos+adj:sg:nom.voc:m1.m2.m3:pos
tylko       tylko       conj+qub
się         się         siebie:acc:nakc+siebie:gen:nakc+subst:sg:nom:n2
dowie       dowiedzieć  verb:fin:sg:ter:perf:refl
kto         kto         subst:sg:nom:m1
cię         ty          ppron12:sg:acc:m1.m2.m3.f.n1.n2:sec:nakc+ppron12:sg:gen:m1.m2.m3.f.n1.n2:sec:nakc
stracił     stracić     verb:praet:sg:m1.m2.m3:ter:perf:refl.nonrefl
dziś        dziś        adv+subst:pl:acc:n2+subst:pl:dat:n2+subst:pl:gen:n2
piękność    piękność    subst:sg:acc:f+subst:sg:nom:f
twą         twój        adj:sg:acc:f:pos+adj:sg:inst:f:pos
w           w           prep:acc:nwok+prep:loc:nwok
w           wiek        brev:pun
całej       cały        adj:sg:dat:f:pos+adj:sg:gen:f:pos+adj:sg:loc:f:pos
ozdobie     ozdoba      subst:sg:dat:f
ozdobie     ozdoba      subst:sg:loc:f
widzę       widzieć     verb:fin:sg:pri:imperf:refl.nonrefl
i           i           conj+interj+qub
opisuję     opisywać    verb:fin:sg:pri:imperf:refl.nonrefl
bo          bo          comp+qub
tęsknię     tęsknić     verb:fin:sg:pri:imperf:nonrefl
po          po          prep:acc+prep:loc
tobie       ty          ppron12:sg:dat:m1.m2.m3.f.n1.n2:sec:akc+ppron12:sg:loc:m1.m2.m3.f.n1.n2:sec
```

Wektorowa reprezentacja tekstów

- Term-Document Matrix
  - przykład minimalistyczny, cztery dokumenty i tylko dwa termy. Wówczas można wynik zaprezentować na wykresie 2D
    - ✓ D1: database database computer
    - ✓ D2: computer
    - ✓ D3: computer computer
    - ✓ D4: database database database
    - ✓ QUERY: database computer

	D1	D2	D3	D4	Q
T1	1	1	2	0	1
T2	2	0	0	3	1

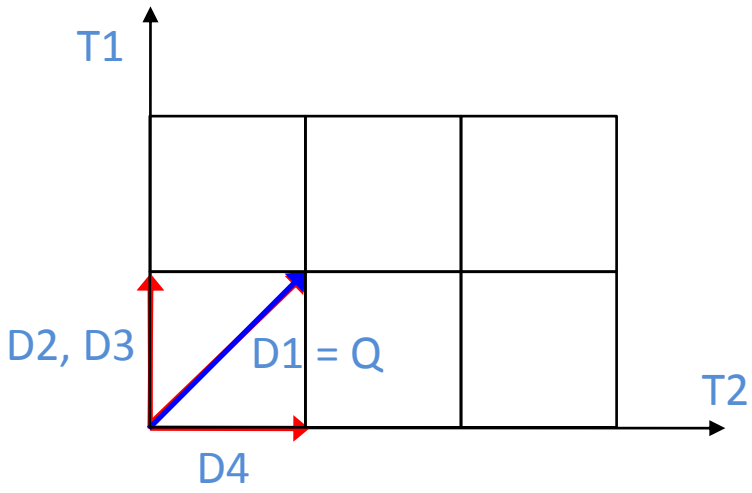
Reprezentacja Bag-Of-Words. Uwzględniamy liczbę wystąpień.



Wektorowa reprezentacja tekstów

- Term-Document Matrix
  - TDM w wersji binarnej

	D1	D2	D3	D4	Q
T1	1	1	1	0	1
T2	1	0	0	1	1



  - bardziej praktyczne wersje będą omówione niedługo

Model przestrzeni wektorowej (vector space model)

- Dokumenty reprezentowane są jako wektory w wielowymiarowej przestrzeni euklidesowej. Na każdej osi jest jeden term
  - w przykładach powyżej T1, T2
- Oznaczenia
  - $d_1, d_2, \ldots, d_n$ n dokumentów
  - $t_1, t_2, \ldots, t_m$ m termów

Miara Term-Frequency (TF)

- Miara TF (term frequency)

$$TF(t,d) = \begin{cases} 0 \ dla \ n_{td} = 0 \\ \dfrac{n_{td}}{\sum_{k=1}^{m} n_{kd}} \ dla \ n_{td} > 0 \end{cases}$$

Liczba wystąpień termu **t** w dokumencie **d**

Całkowita liczba termów w dokumencie **d**. Sumowanie po wszystkich **m** termach

Ten zapis oznacza, że TF jest funkcją dwóch parametrów: **t** oraz **d**

- Mierzy częstotliwość termów w dokumencie. Niweluje wpływ długości dokumentu, gdyż czym dłuższy dokument, tym oczywiście występuje w nim więcej słów
  - termy, które występują często w dokumentach wcale nie muszą być automatycznie najważniejsze

Miara Term-Frequency (TF)

- Prosty przykład ilustrujący „dobroczynny” efekt TF

	komputer	programowanie	laboratorium
D1	0	1	2
D2	0	0	1
D3	2	3	**4**
D4	0	0	0
D5	0	2	**4**

dokument	ilość termów
D1	10
D2	20
D3	15
D4	50
D5	60

	komputer	programowanie	laboratorium
D1	0	1/10=0,1	2/10=0,2
D2	0	0	1/20=0,05
D3	2/15=0,133	3/15=0,2	**4/15=0,266**
D4	0	0	0
D5	0	2/60=0,033	**4/60=0,067**

Dokumenty D3 oraz D5 mają tyle samo fraz „laboratorium”, jednak D5 jest dłuższy od D3. W efekcie fraza „laboratorium”, jest mniej „ważna” w D5, niż w D3

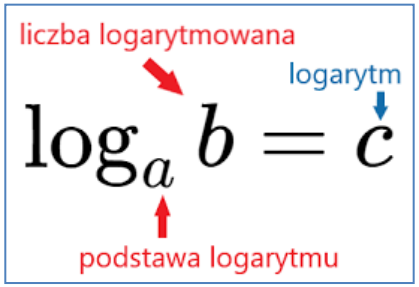
Dokumenty są w wierszach a termy w kolumnach. To jest w sumie sprawa umowna

Miara Inverse Document Frequency (IDF)

- Mierzy jak ważny jest dany term w analizowanym zbiorze dokumentów. Przy wyliczaniu miary TF wszystkie termy „ważyły” tyle samo. Ale oczywistym jest, że pewne termy występują dużo częściej niż inne i w efekcie ich „moc poznawcza” jest mniej znacząca. Takim termom trzeba obniżyć ranking a podnieść ranking termom występującym rzadziej
  - np. w dokumentach medycznych term „pacjent” ma niewielką wartość poznawczą. Podobnie w dokumentach biologicznych na temat białek term „białko”

Miara Inverse Document Frequency (IDF)

- Używamy tutaj funkcji logarytmicznej, więc krótkie przypomnienie

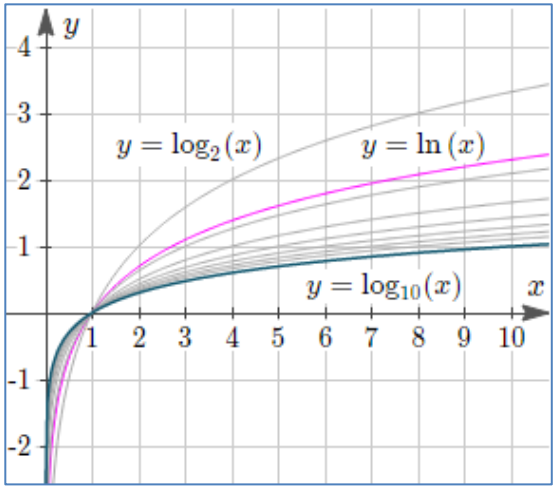


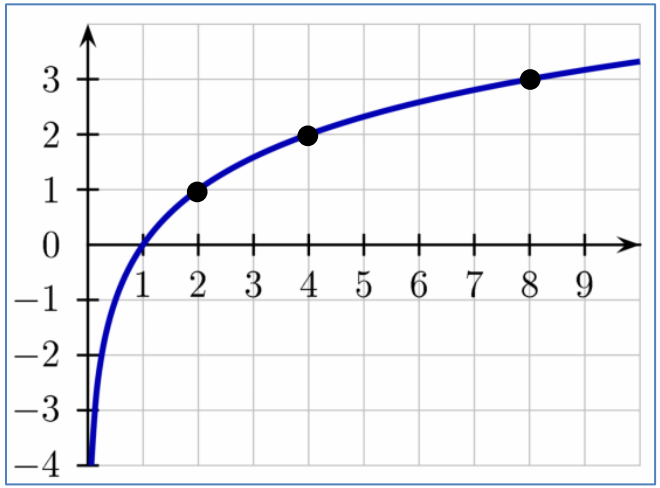
$\log_2 8 = 3$ bo: $2^3 = 8$

$\log_3 81 = 4$ bo: $3^4 = 81$

$\log_{\sqrt{2}} 2 = 2$ bo: $(\sqrt{2})^2 = 2$

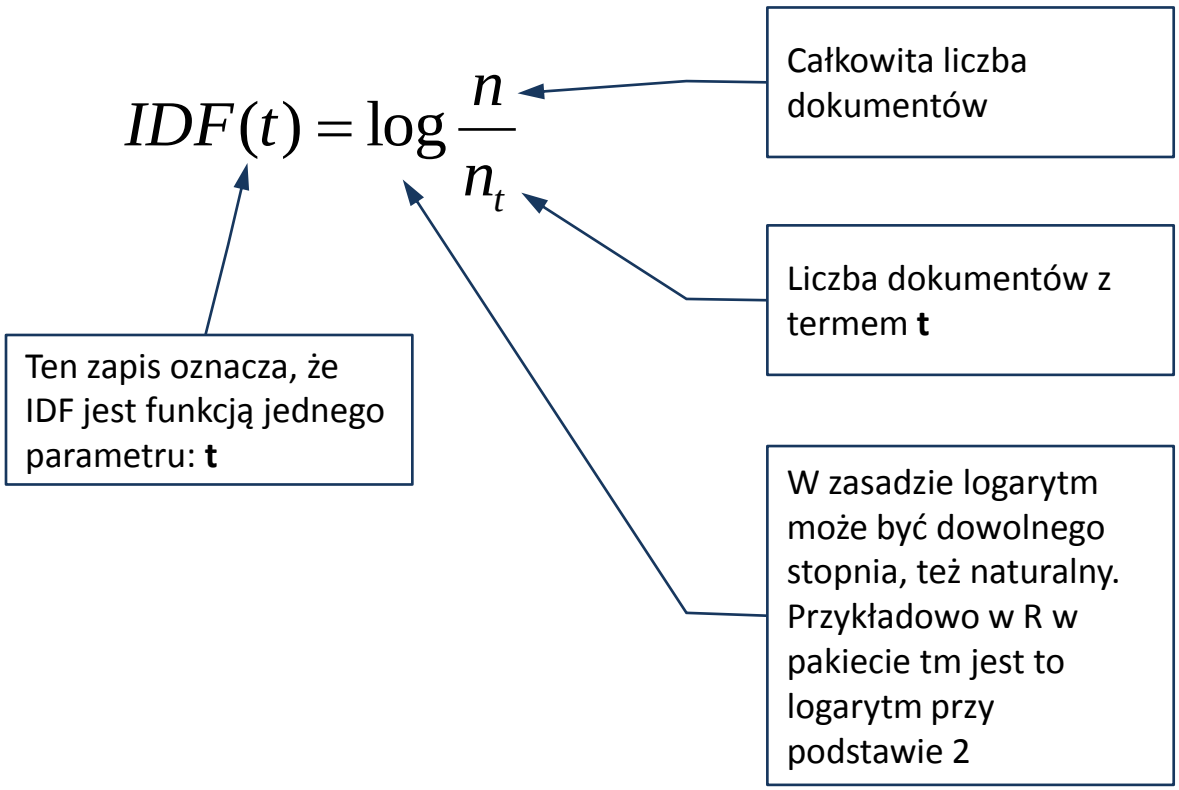
$\log_5 \sqrt[3]{5} = \frac{1}{3}$ bo: $5^{\frac{1}{3}} = \sqrt[3]{5}$





Miara Inverse Document Frequency (IDF)

- Wzór



Miara Inverse Document Frequency (IDF)

- Przykład

	komputer	programowanie	laboratorium
D1	0	1	**2**
D2	0	0	**1**
D3	**2**	3	**4**
D4	0	0	0
D5	0	2	**4**

	komputer	programowanie	laboratorium
n / n_t	5/1=5	5/3=1,67	5/4=1,25
$\log_2 (n / n_t)$	1,61	0.51	0.22

Term „laboratorium” występuje aż w 4 dokumentach, więc jego miara IDF jest niewielka w porównaniu np. do termu „komputer”, który występuje tylko w jednym dokumencie.

Połączenie miar TF oraz IDF

- Wymnażamy TF*IDF

TF

	komputer	programowanie	laboratorium
D1	0	0,1	0,2
D2	0	0	0,05
D3	0,133	0,2	0,266
D4	0	0	0
D5	0	0,033	0,067

IDF

	komputer	programowanie	laboratorium
n / n_t	5/1=5	5/3=1,67	5/4=1,25
$\log_2 (n / n_t)$	1,61	0,51	0,22

TF * IDF

	komputer	programowanie	laboratorium
D1	0·1,61=**0**	0,1·0,51=**0,051**	0,2·0,22=**0,044**
D2	0·1,61=**0**	0·0,51=**0**	0,05·0,22=**0,011**
D3	0,133·1,61=**0,21**	0,2·0,51=**0,102**	0,266·0,22=**0,058**
D4	0·1,61=**0**	0·0,51=**0**	0·0,22=**0**
D5	0·1,61=**0**	0,033·0,51=**0,017**	0,067·0,22=**0,147**

TF_{D3} = (0,133; 0,2; 0,266)

$TFIDF_{D3}$ = (0,21; 0,102; 0,058) ← Wnioski?

Miara Inverse Document Frequency (IDF)

- Różne inne wersje miary IDF

$$IDF(t) = \log_2 \frac{n}{n_t + 1}$$

1 zabezpiecza przed dzieleniem przez 0

$$IDF(t) = 1 + \log_e \frac{n}{n_t}$$

Tutaj mamy logarytm naturalny

$$IDF(t) = \log_2 \frac{1+n}{n_t}$$

Normalizacja wektorów

- Kierunek wektora nie zmieni się, gdy poddamy go normalizacji. Wówczas jego długość zawsze będzie wynosiła 1

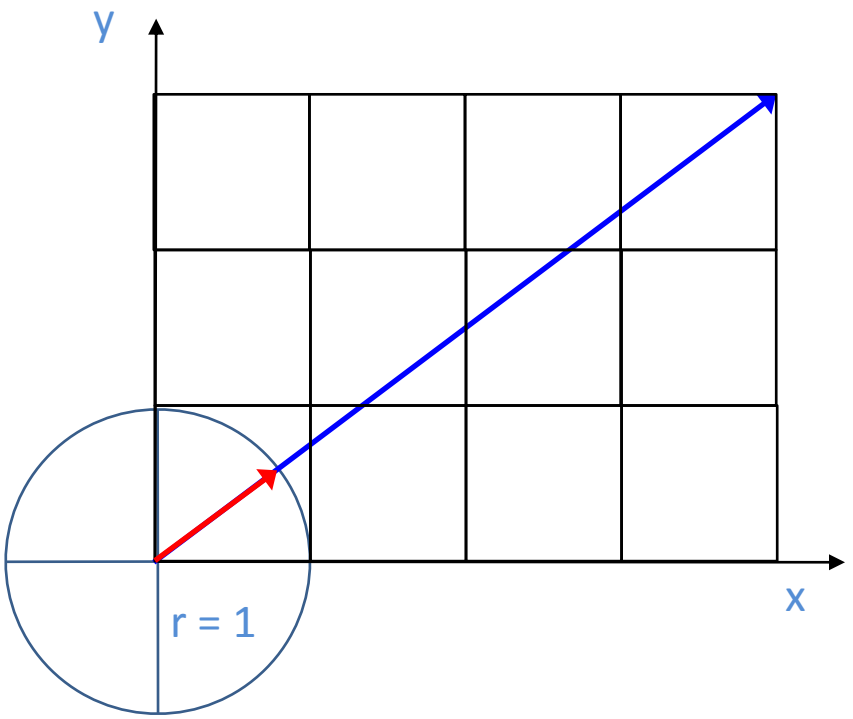
$$\hat{x} = \frac{x}{|x|}$$

Przyklad:

$$a = [4, 3]$$

$$|a| = \sqrt{4^2 + 3^2} = \sqrt{25} = 5$$

$$\hat{a} = \frac{[4, 3]}{\sqrt{25}} = \left[\frac{4}{5}, \frac{3}{5}\right]$$



Wzór ogólny

- Wartość wyznaczana dla każdej komórki (t,d) w macierzy TDM można zapisać w ogólny sposób jak poniżej:

$$a(t,d) = l(t,d)\, g(t)\, n(d)$$

  - $l(t,d)$: komponent lokalny (local weight), np. FT
  - $g(t)$: komponent globalny (global weight), np. IDF
  - n: normalizacja wektora dokumentu d, np. $x / |x|$

Postać tego wyrażenia, powszechnie tak właśnie przedstawiana w literaturze, jest myląca, gdyż sugeruje przemnożenie komponentów ***l*** oraz ***g*** przez ***n***. W rzeczywistości chodzi o dokonanie stosownej normalizacji kolumn macierzy po jej wyliczeniu z ew. uwzględnieniem komponentów ***l*** oraz ***g***.

TF, IDF, TFIDF – kilka przykładów

	D1	D2	D3	D4	Q
T1	1	1	1	0	1
T2	1	0	0	1	1

binarna

	D1	D2	D3	D4	Q
T1	1	1	2	0	1
T2	2	0	0	3	1

z uwzględnienie ilości termów

	D1	D2	D3	D4	Q
T1	1/3	1/1	2/2	0	1/2
T2	2/3	0	0	3/3	1/2

TF

T1	$\log_2(4/3)=0{,}415$
T2	$\log_2(4/2)=1$

IDF

	D1	D2	D3	D4	Q
T1	0,138	0,415	0,415	0	0,2075
T2	0,666	0	0	1	0,5

TF*IDF

Uwaga: przy liczeniu IDF NIE uwzględniliśmy Q, czyli **n = 4**

$$TF(\mathrm{T1},\mathrm{D1}) * IDF(\mathrm{T1}) = \frac{1}{3} * \log_2\left(\frac{4}{3}\right)$$

TF, IDF, TFIDF – kilka przykładów

- Dane jak na poprzednim slajdzie ale tym razem uwzględniamy Q przy liczeniu IDF

T1	$\log_2(5/4)=0{,}322$
T2	$\log_2(5/3)=0{,}737$

IDF

	D1	D2	D3	D4	Q
T1	0,107	0,322	0,322	0	0,161
T2	0,491	0	0	0,737	0,368

TF*IDF

Uwaga: przy liczeniu IDF UWZGLĘDNIAMY Q, czyli **n = 5**

$$TF(\mathrm{T1},\mathrm{D1}) * IDF(\mathrm{T1}) = \frac{1}{3} * \log_2\left(\frac{5}{4}\right)$$

- Tak było poprzednio

	D1	D2	D3	D4	Q
T1	0,138	0,415	0,415	0	0,2075
T2	0,666	0	0	1	0,5

Różnice są dość znaczne, ale dla rzeczywistych dokumentów różnice będą w praktyce mniejsze

TF, IDF, TFIDF – kilka przykładów

- Bardziej rzeczywisty przypadek

Tutaj w Q nie ma T1.
Różnica nie jest wielka

$$TF(\mathrm{T500},\mathrm{D50}) = \frac{2}{7}$$

$$TF(\mathrm{T1000},\mathrm{D1}) = \frac{10}{17}$$

	D1	D2	…	D50	…	D100	Q
T1		5		1		4	
T2	2					1	1
…							
T500	5			2			1
…							
T1000	10			4		6	1

$$IDF(\mathrm{T1}) = \log_2\left(\frac{100}{3}\right) = 5{,}0589$$

$$IDF(\mathrm{T1}) = \log_2\left(\frac{101}{3}\right) = 5.0732$$

$$IDF(\mathrm{T1000}) = \log_2\left(\frac{100}{3}\right) = 5{,}0589$$

$$IDF(\mathrm{T1000}) = \log_2\left(\frac{101}{4}\right) = 4.6582$$

Tutaj w Q jest T1000.
Różnica jest znacząca

Ranking dokumentów, miara kosinusowa

- Iloczyn skalarny (ang. *dot product*)

$$a = [a_1, a_2, \ldots, a_n]$$

$$b = [b_1, b_2, \ldots, b_n]$$

$$a \cdot b = \sum_{i=1}^{n} a_i b_i$$

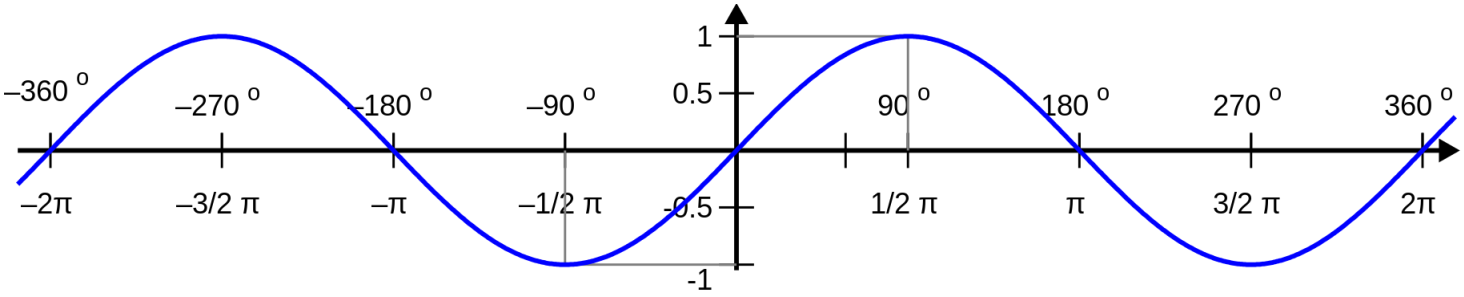
$$a \cdot b = ab^T = \begin{bmatrix} \cdot & \cdot & \cdot \end{bmatrix}_{1\,x\,n} \begin{bmatrix} \cdot \\ \cdot \\ \cdot \end{bmatrix}_{n\,x\,1}$$

- Właściwości
  - przemienność
    $a \cdot b = b \cdot a$
  - rozdzielność
    $a \cdot (b + c) = a \cdot b + a \cdot c$
  - zgodność z mnożeniem przez skalar
    $(ra) \cdot b = a \cdot (rb) = r(a \cdot b)$
  - ortogonalność
    $a \cdot b = 0 \Rightarrow a \perp b$
  - dodatnia określoność
    $a \cdot a > 0 \text{ dla } a > 0$

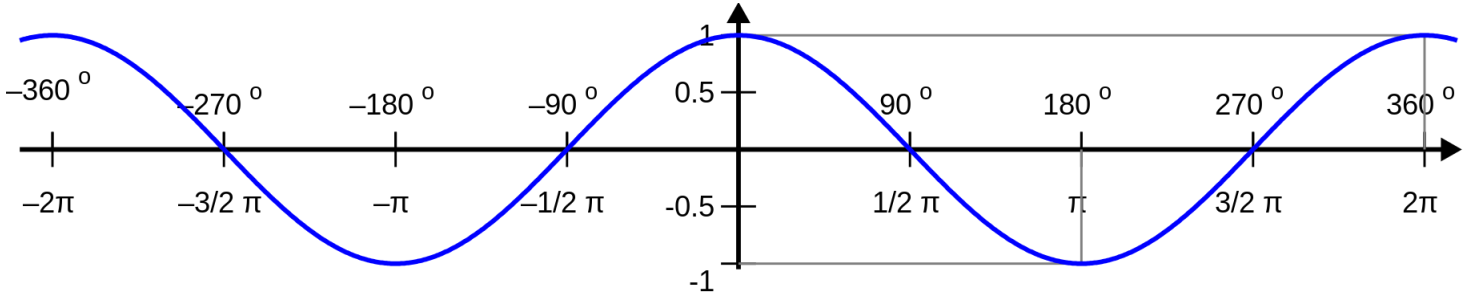
Funkcja cosinus i sinus

- Dla przypomnienia ze szkoły podstawowej ☺

y=sin(x)



y = cos(x)



Ranking dokumentów, miara kosinusowa

- Iloczyn skalarny, interpretacja graficzna

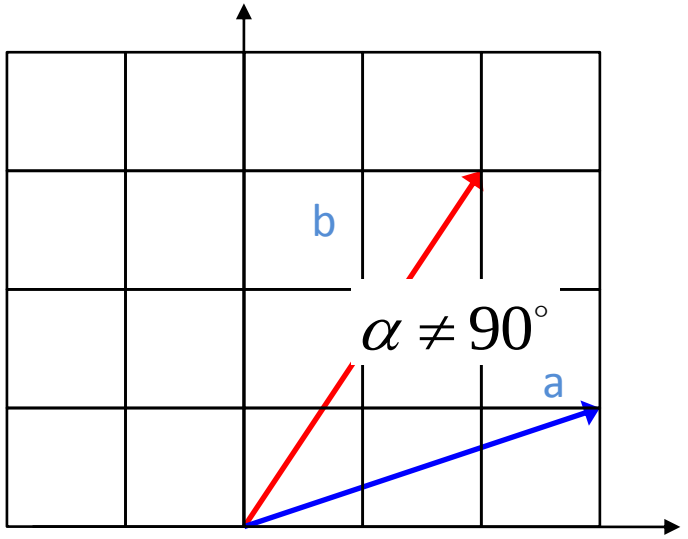
$$a \cdot b = |a|\,|b| \cos\alpha$$

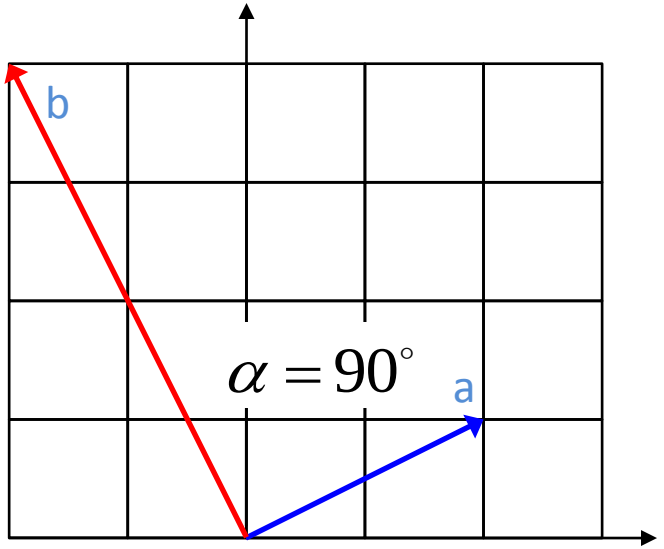
$$\cos\alpha = \frac{a \cdot b}{|a|\,|b|} \rightarrow \alpha = \arccos\frac{a \cdot b}{|a|\,|b|}$$

$$|a| = \sqrt{a \cdot a} = \sqrt{a_1^2 + \cdots a_n^2}$$

$$a \cdot b = [3,1] \cdot [2,3] = 3 \cdot 2 + 1 \cdot 3 = 9$$

$$a \cdot b = [2,1] \cdot [-2,4] = -4 + 4 = 0$$

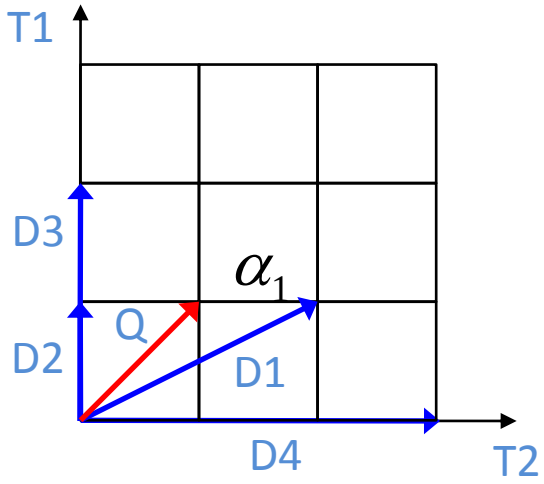




Ranking dokumentów, miara kosinusowa

- TDM bez normalizacji

	D1	D2	D3	D4	Q
T1	1	1	2	0	1
T2	2	0	0	3	1



- TDM znormalizowana

$$|D1| = \sqrt{1^2 + 2^2} = 2{,}24$$

$$|D2| = \sqrt{1^2 + 0^2} = 1$$

$$|D3| = \sqrt{2^2 + 0^2} = 2$$

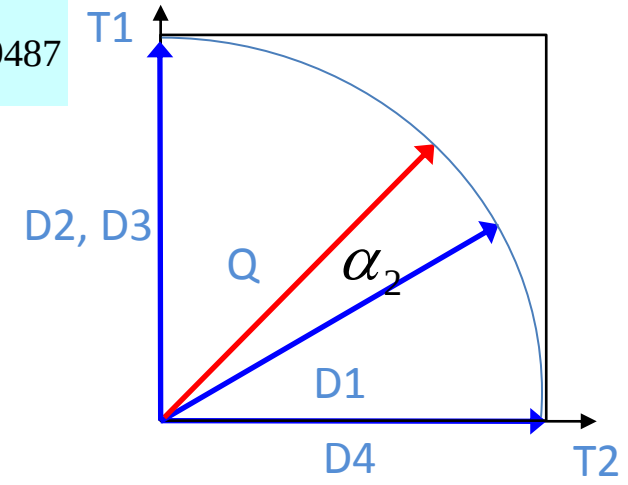
$$|D4| = \sqrt{0^2 + 3^2} = 3$$

$$|Q| = \sqrt{1^2 + 1^2} = 1{,}44$$

$$\cos\alpha_1 = \frac{[1,1]\cdot[1,2]}{\sqrt{2}\cdot\sqrt{5}} = 0{,}9487$$

$$\cos\alpha_2 = \frac{[0{,}7071,\ 0{,}7071]\cdot[0{,}4472,\ 0{,}8944]}{1\cdot 1} = 0{,}9487$$

	D1	D2	D3	D4	Q
T1	0,4472	1	1	0	0,7071
T2	0,8944	0	0	1	0,7071

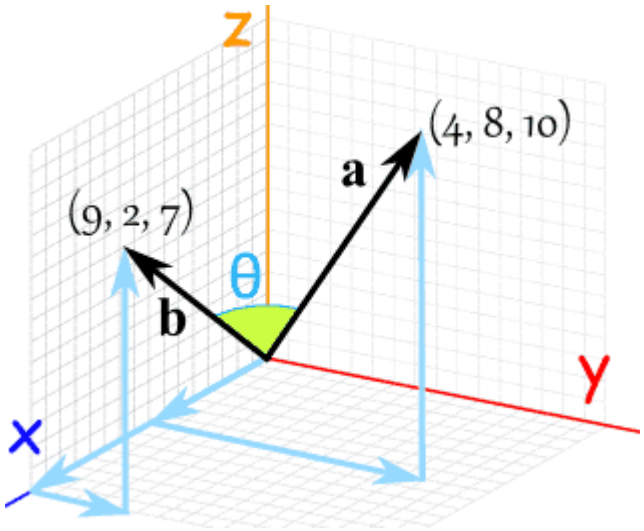


Ranking dokumentów, miara kosinusowa

- Dokumenty, których wektory biegną w tym samym (podobnym) kierunku uważamy za podobne do siebie
- Miara kosinusowa ograniczona jest do przedziału [0, 1].
  - 0: dokumenty są zupełnie niepodobne do siebie
  - 1: dokumenty są maksymalnie podobne do siebie

Wektory 3D

- Poniżej przykład w przestrzeni 3D. Uogólnienie na N wymiarów jest oczywiste



$$a \cdot b = [9, 2, 7] \cdot [4, 8, 10] = 122$$

$$|a| = \sqrt{4^2 + 8^2 + 10^2} = \sqrt{180}$$

$$|b| = \sqrt{9^2 + 2^2 + 7^2} = \sqrt{134}$$

$$\cos\alpha = \frac{122}{\sqrt{180}\sqrt{134}} = 0{,}7855$$

$$\alpha = \arccos(0{,}7855) = 38{,}2^\circ$$

źródło: https://www.mathsisfun.com/algebra/vectors-dot-product.html

TF, IDF, miara cosinus – większy przykład

- Wszystkie przykłady w oparciu o system R
  - … ale Python też nie pozostaje w tyle w dziedzinie eksploracji tekstu

D1: ~~The~~ sky ~~is~~ blue.
D2: ~~The~~ sun ~~is~~ brigh today.
D3: ~~The~~ sun ~~in~~ ~~the~~ sky ~~is~~ bright.
D4: ~~We~~ ~~can~~ ~~see~~ ~~the~~ shining sun, ~~the~~ bright sun.
Q: sky sun

podstawowa TDM

	Docs				
Terms	**1**	**2**	**3**	**4**	**5**
blue	1	0	0	0	0
bright	0	1	1	1	0
shine	0	0	0	1	0
sky	1	0	1	0	1
sun	0	1	1	2	1
today	0	1	0	0	0

*TF*IDF*

	Docs				
Terms	**1**	**2**	**3**	**4**	**5**
blue	1.1610	0.0000	0.0000	0.0000	0.0000
bright	0.0000	0.2457	0.2457	0.1842	0.0000
shine	0.0000	0.0000	0.0000	0.5805	0.0000
sky	0.3685	0.0000	0.2457	0.0000	0.3685
sun	0.0000	0.1073	0.1073	0.1610	0.1610
today	0.0000	0.7740	0.0000	0.0000	0.0000

TF, IDF, miara cosinus – większy przykład

- Ciąg dalszy przykładu. Pokazujemy w szczegółach, jak wykonuje się obliczenia

TF

	Docs				
Terms	**1**	**2**	**3**	**4**	**5**
blue	1/2	0	0	0	0
bright	0	1/3	1/3	1/4	0
shine	0	0	0	1/4	0
sky	1/2	0	1/3	0	1/2
sun	0	1/3	1/3	2/4	1/2
today	0	1/3	0	0	0

IDF

Terms	
blue	log_2(5/1) = 2.322
bright	log_2(5/3) = 0.737
shine	log_2(5/1) = 2.322
sky	log_2(5/3) = 0.737
sun	log_2(5/4) = 0.322
today	log_2(5/1) = 2.322

*TF*IDF*

	1	2	3	4	5
blue	1/2 * 2.322 = 1.1610	0	0	0	0
bright	0	1/3 * 0.737 = 0.2457	1/3 *0.737 = 0.2457	1/4 *0.737 = 0.1842	0
shine	0	0	0	1/4 * 2.322 = 0.5805	0
sky	1/2 * 0.737 = 0.3685	0	1/3 * 0.737 = 0.2457	1/2 * 0.737 = 0.3685	1/2 * 0.737 = 0.3685
sun	0	1/3 * 0.322 = 0.1073	1/3 * 0.322 = 0.1073	2/4 * 0.322 = 0.1610	1/2 * 0.322 = 0.1610
today	0	1/3 * 2.322 = 0.7740	0	0	0

TF, IDF, miara cosinus – większy przykład

- Ciąg dalszy przykładu. Miara kosinusowa

Docs	1	2	3	4
cos α	**0.27723**	**0.05244**	**0.73726**	**0.10229**

D1: ~~The~~ sky ~~is~~ blue.
D2: ~~The~~ sun ~~is~~ brigh today.
D3: ~~The~~ sun ~~in~~ ~~the~~ sky ~~is~~ bright.
D4: ~~We~~ ~~can~~ ~~see~~ ~~the~~ shining sun, ~~the~~ bright sun.
Q: sky sun

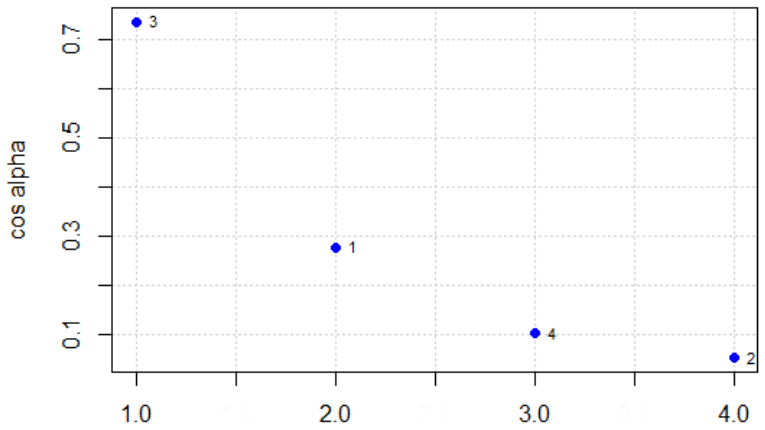
$$D1 = [1{,}161 \quad 0 \quad 0 \quad 0{,}685 \quad 0 \quad 0]$$

$$Q = [0 \quad 0 \quad 0 \quad 0{,}3685 \quad 0{,}161 \quad 0]$$

$$|D1| = 1{,}218$$

$$|Q| = 0{,}4021$$

$$\cos\alpha = \frac{D1 \cdot Q}{|D1||Q|} = \frac{0{,}1358}{1{,}218 \cdot 0{,}4021} = 0{,}2772$$



TF, IDF, miara cosinus – jeszcze większy przykład

- Z książki: Zdravko Markov and Daniel T. Larose, *Data Mining the Web: Uncovering Patterns in Web Content, Structure, and Usage*, Wiley, April 2007
  - polskie wydanie: Eksploracja zasobów internetowych, Wydawnictwo Naukowe PWN, 2009
  - https://cs.ccsu.edu/~markov/

```
Computer Science

            Computer Science
              Students majoring in computer science may choose between the
              honors program, accredited by the Computer Science Accreditation
              Board, and the alternative program, designed to facilitate
              graduation of transfer students with associate degrees. Computer
              Science faculty have expertise in artificial intelligence and
              knowledge engineering, computer graphics, networking and
              distributed processing, parallel programming, systems programming,
              and software engineering. Available hardware includes UNIX and Sun
              workstations along with student ftp and World-Wide Web servers
              running on a network of NeXTSTEP workstations.
              PROGRAM OF STUDY: BS
              DEPARTMENT CHAIR
              Joan Calvert
              Location: Maria Sanford Hall 203
              Phone: 832-2710
              Department Website
```

```
D1  Anthropology
D2  Art
D3  Biology
D4  Chemistry
D5  Communication
D6  Computer Science
D7  Criminal Justice
D8  Economics
D9  English
D10 Geography
D11 History
D12 Math
D13 Modern Languages
D14 Music
D15 Philosophy
D16 Physics
D17 Political Sciences
D18 Psychology
D19 Sociology
D20 Theatre
```

TF, IDF, miara cosinus – jeszcze większy przykład

- Obliczenia z wykorzystaniem pakietu tm w R

```
<<TermDocumentMatrix (terms: 518, documents: 20)>>
Non-/sparse entries: 881/9479
Sparsity           : 91%
Maximal term length: 18
Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
Sample             :
             Docs
Terms              1 12    14     16 2    20      6     7 8      9
  anthropolog 0.4549  0 0.000 0.0000 0 0.000 0.0000 0.000 0 0.0000
  chemistri   0.0000  0 0.000 0.0000 0 0.000 0.0000 0.000 0 0.0000
  crimin      0.0000  0 0.000 0.0000 0 0.000 0.0000 0.000 0 0.0000
  econom      0.0000  0 0.000 0.0000 0 0.000 0.0000 0.387 0 0.0000
  justic      0.0000  0 0.000 0.0000 0 0.000 0.0000 0.000 0 0.0000
  mathemat    0.0000  0 0.000 0.0000 0 0.000 0.0000 0.000 0 0.0000
  music       0.0000  0 0.602 0.0000 0 0.000 0.0000 0.000 0 0.0000
  scienc      0.0174  0 0.000 0.0319 0 0.000 0.0766 0.000 0 0.0232
  sociolog    0.0000  0 0.000 0.0000 0 0.000 0.0000 0.000 0 0.0000
  theatr      0.0000  0 0.000 0.0000 0 0.508 0.0000 0.000 0 0.0000
```

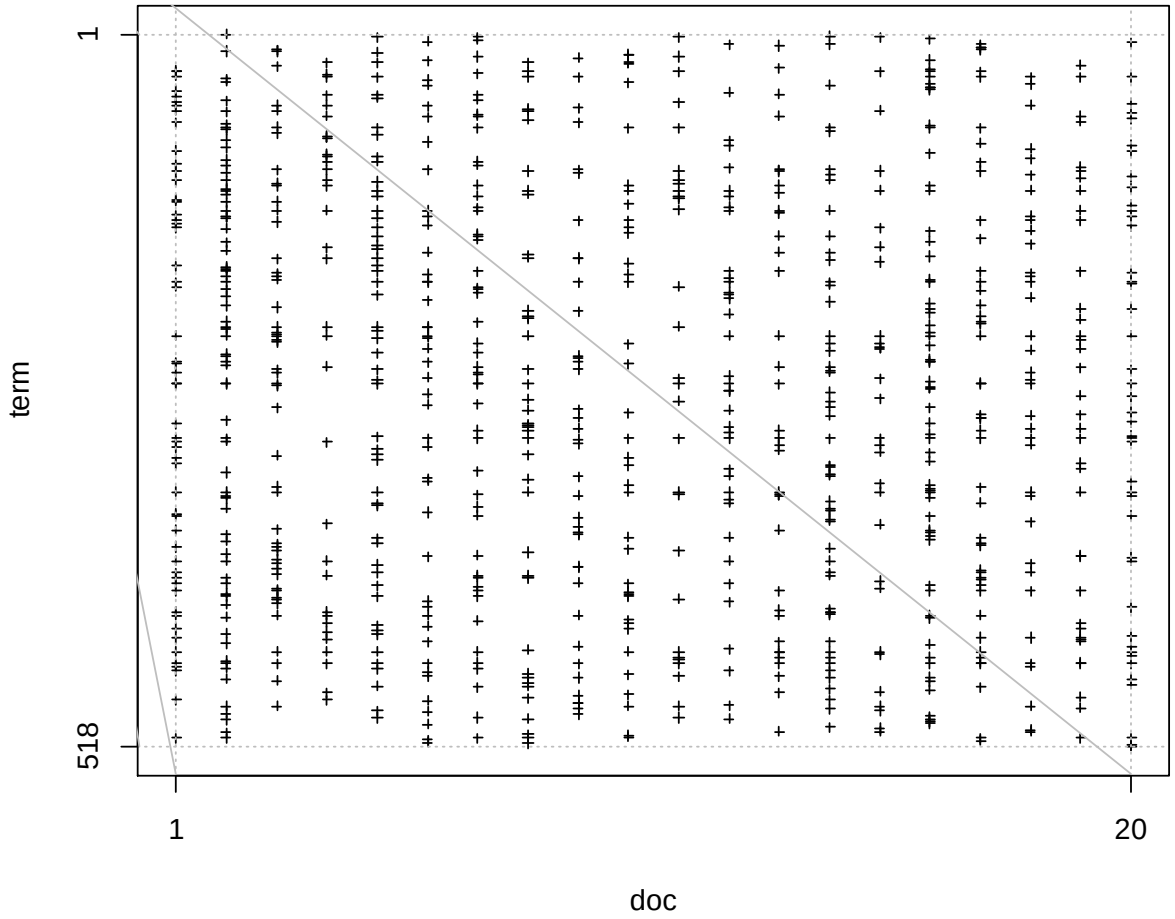
TF, IDF, miara cosinus – jeszcze większy przykład

- Inaczej zaprezentowany fragment TDM

	Docs																			
Terms	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
abound	0	**0.045**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0	0.000	0.000
abroad	0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.055**	0.000	0.000	0.000	0	0.000	0.000
academ	0	0.000	0.000	0.000	**0.061**	0.000	**0.035**	0.000	0.000	0.000	**0.046**	0.000	0.000	0.000	**0.049**	0.000	0.000	0	0.000	0.000
acceler	0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.052**	0.000	0	0.000	0.000
account	0	0.000	0.000	0.000	0.000	0.000	**0.065**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0	0.000	0.000
accredit	0	0.000	0.000	0.000	0.000	**0.125**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0	0.000	0.000
act	0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0	0.000	**0.064**
activ	0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.043**	0.000	**0.035**	0.000	0.000	**0.038**	0	0.000	0.000
actuari	0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.079**	0.000	0.000	0.000	0.000	0	0.000	0.000
addit	0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.060**	0	0.000	0.000
administr	0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.120**	0	0.000	0.000
advis	0	0.000	**0.042**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.046**	0	0.000	0.000
advisor	0	**0.035**	**0.042**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0	0.000	0.000
advisorbas	0	0.000	0.000	0.000	**0.057**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0	0.000	0.000
african	0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.065**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0	0.000	0.000
africanamerican	0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.065**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0	0.000	0.000
agenc	0	0.000	0.000	0.000	0.000	0.000	**0.050**	0.000	0.000	0.000	**0.065**	0.000	0.000	0.000	0.000	0.000	0.000	0	0.000	0.000
air	0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.076**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0	0.000	0.000
ali	0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.052**	0.000	0	0.000	0.000
altern	0	0.000	0.000	0.000	0.000	**0.063**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0	0.000	0.000
american	0	0.000	0.000	**0.046**	0.031	0.000	0.000	**0.034**	0.000	**0.104**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0	0.000	0.000
ancient	0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.065	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0	0.000	0.000
andor	0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0	**0.072**	0.000
anesthesia	0	0.000	**0.055**	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0	0.000	0.000
annual	0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	**0.079**	0.000	0.000	0.000	0.000	0	0.000	0.000

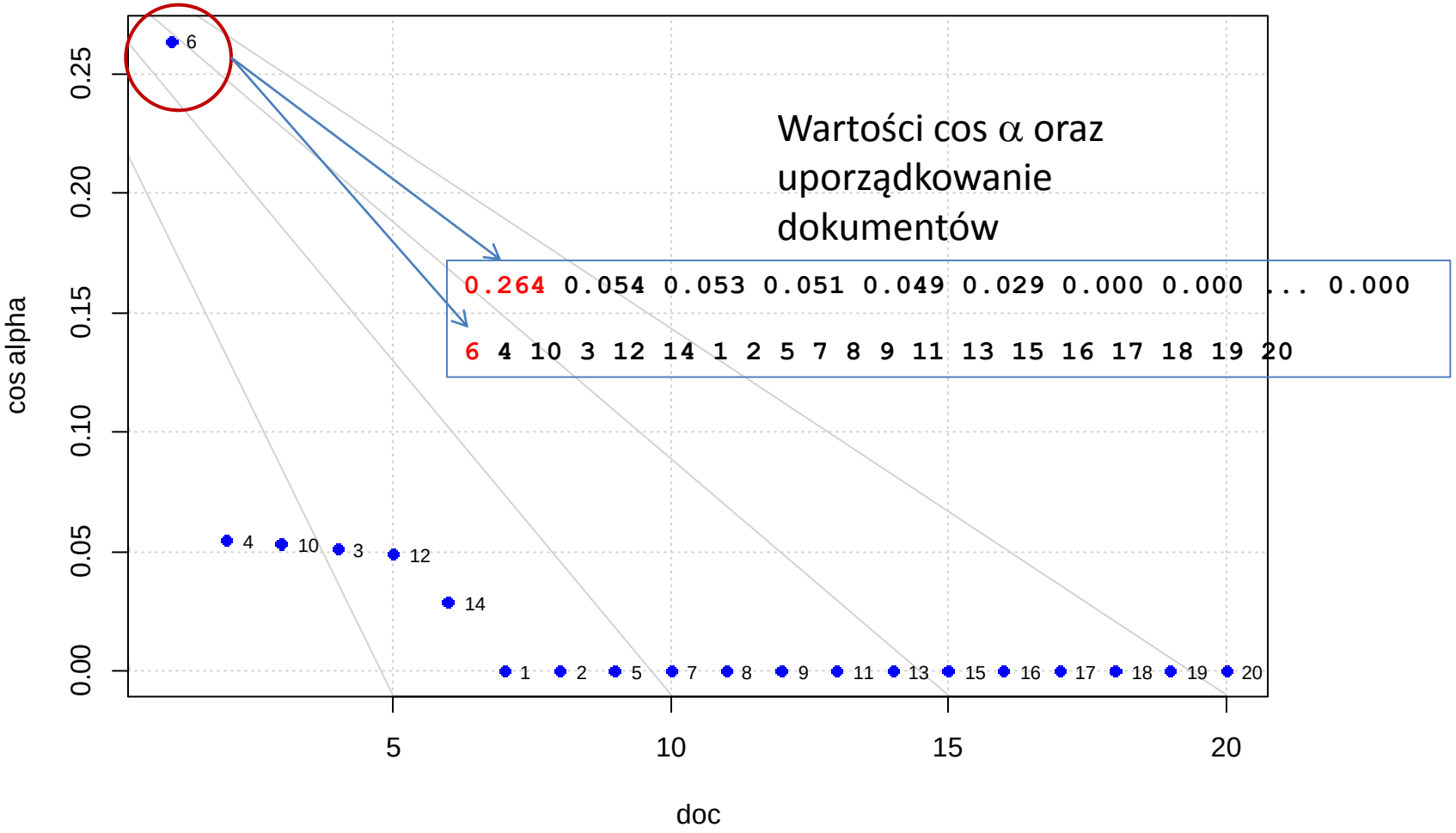
TF, IDF, miara cosinus – jeszcze większy przykład

- Wizualizacja zapełnienia TDM



TF, IDF, miara cosinus – jeszcze większy przykład

- Zadajemy zapytanie {computer program}



Ocena jakości wyszukiwania

- Dwie podstawe miary
  - precision - dokładność
  - recall – kompletność
- Oznaczenia
  - Q: zbiór zapytań
  - q: konkretne, jedno zapytanie
  - D: zbiór dokumentów
  - $R_q \in D$: zbiór dokumentów zwróconych przez system (retrieved)
  - $D_q \in D$: zbiór istotnych (relevant) dokumentów (wybranych ręcznie z całego zbiory dokumentów. Tego zadania raczej nie wykona maszyna. Musimy wykorzystać wiedzę ekspercką/dziedzinową)

Ocena jakości wyszukiwania

- Dokładność, precyzja (precision)
  - pokazuje, jak skuteczny jest algorytm w zwracaniu wyłącznie pasujących do zapytania dokumentów
  - zakres: [0, 1]
  - 0: najgorszy, nie otrzymano żadnych istotnych dokumentów
  - 1: najlepszy: wszystkie zwrócone dokument są istotne
  - Skrajny przypadek: *P* = 1, użyto bardzo ograniczonego zapytania, które zwraca tylko zbiór istotnych dokumentów

$$P = \frac{\left|D_q \cap R_q\right|}{\left|R_q\right|} = \frac{\left|relevant \cap retrieved\right|}{\left|retrieved\right|}$$

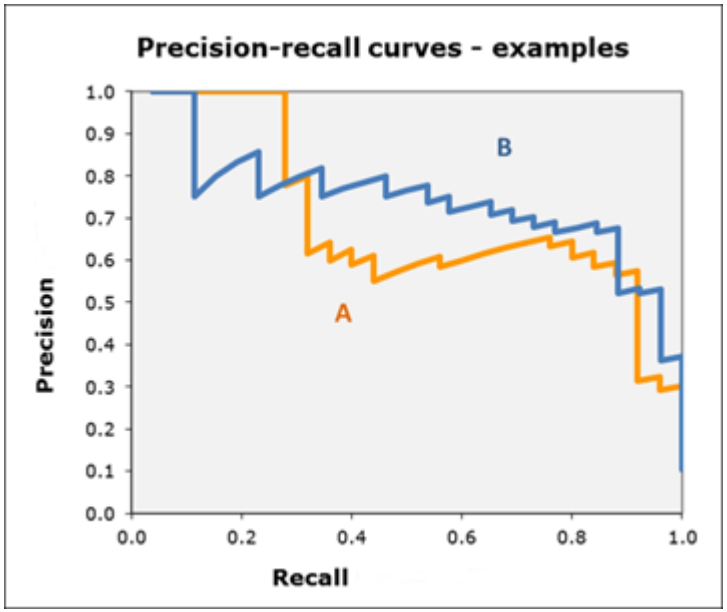
Ocena jakości wyszukiwania

- Kompletność (recall)
  - pokazuje, jak skuteczny jest algorytm w zwracaniu jak największej liczby pasujących do zapytania dokumentów
  - zakres: [0, 1]
  - 0: najgorszy, nie otrzymano żadnych istotnych dokumentów
  - 1: najlepszy: wszystkie zwrócone dokument są istotne
  - Skrajny przypadek: *R* = 1, użyto bardzo ogólnego zapytania, które zwraca zbiór iwszystkich dokumentów

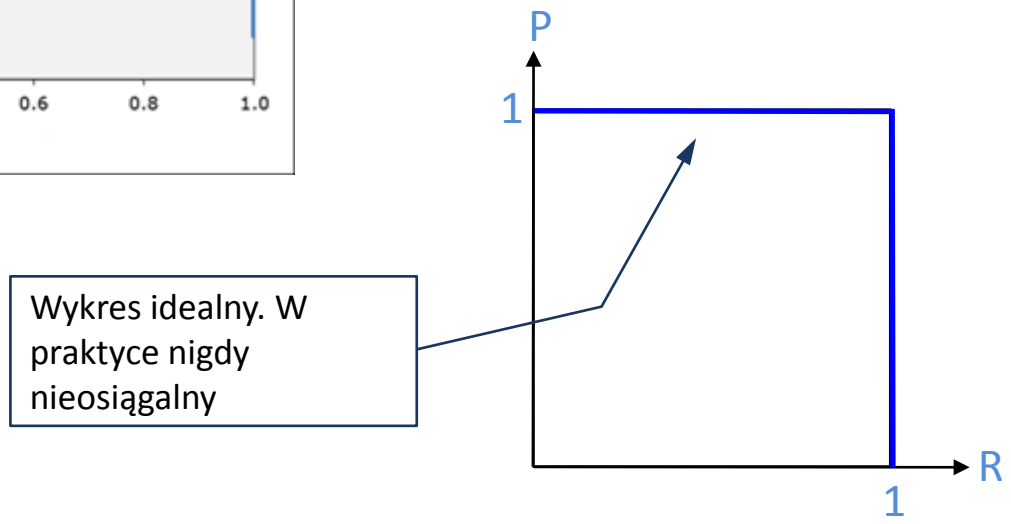
$$R = \frac{\left|D_q \cap R_q\right|}{\left|D_q\right|} = \frac{\left|relevant \cap retrieved\right|}{\left|relevant\right|}$$

Ocena jakości wyszukiwania

- Krzywa precision-recall (P-R)



Krzywa ma kształt piły. System A jest nieco gorszy niż system B



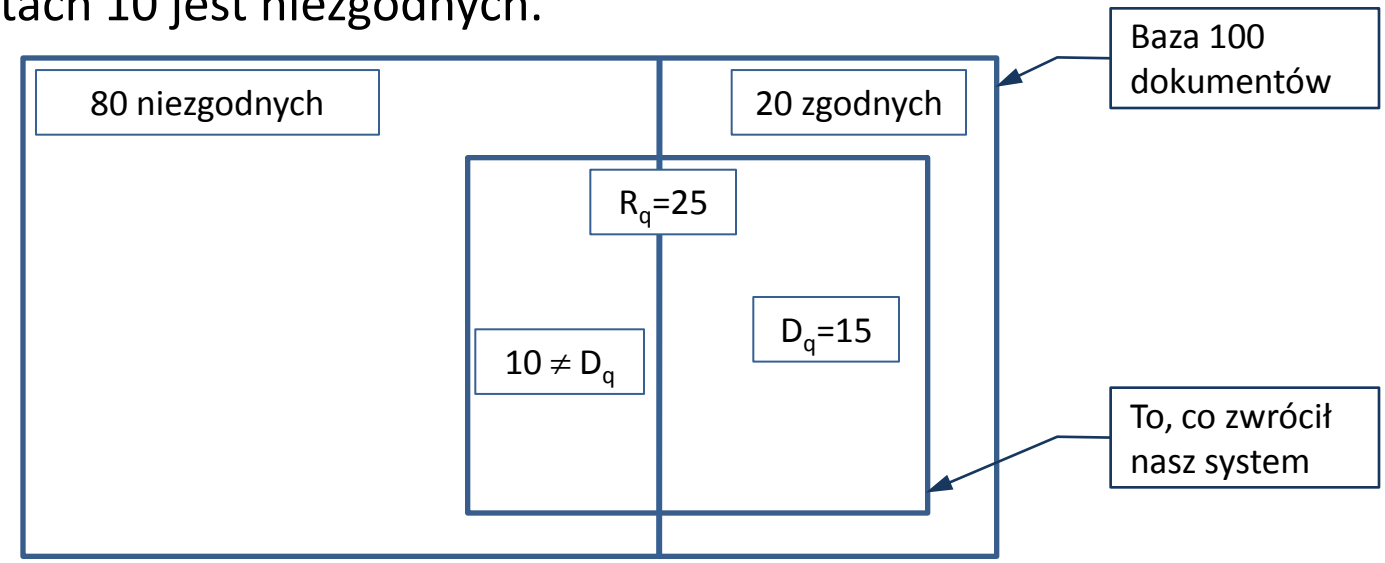
Wykres idealny. W praktyce nigdy nieosiągalny

Ocena jakości wyszukiwania

- Przykład 1
  - mamy 100 dokumentów. Wiemy, że w tych 100 dokumentach 20 jest zgodnych z pewnym zapytaniem (*q*) a pozostałe 80 nie jest z nim zgodnych.
    Teraz powiedzmy, że nasz system zwraca 25 dokumentów. Wśród tych dokumentów 15 jest zgodnych z q. Oznacza to, że system nie odszukał 5 dokumentów zgodnych z *q*. Oznacza to też, że w tych 25 dokumentach 10 jest niezgodnych.

$$P = \frac{|D_q \cap R_q|}{|R_q|} = \frac{15}{25} = 0{,}6$$

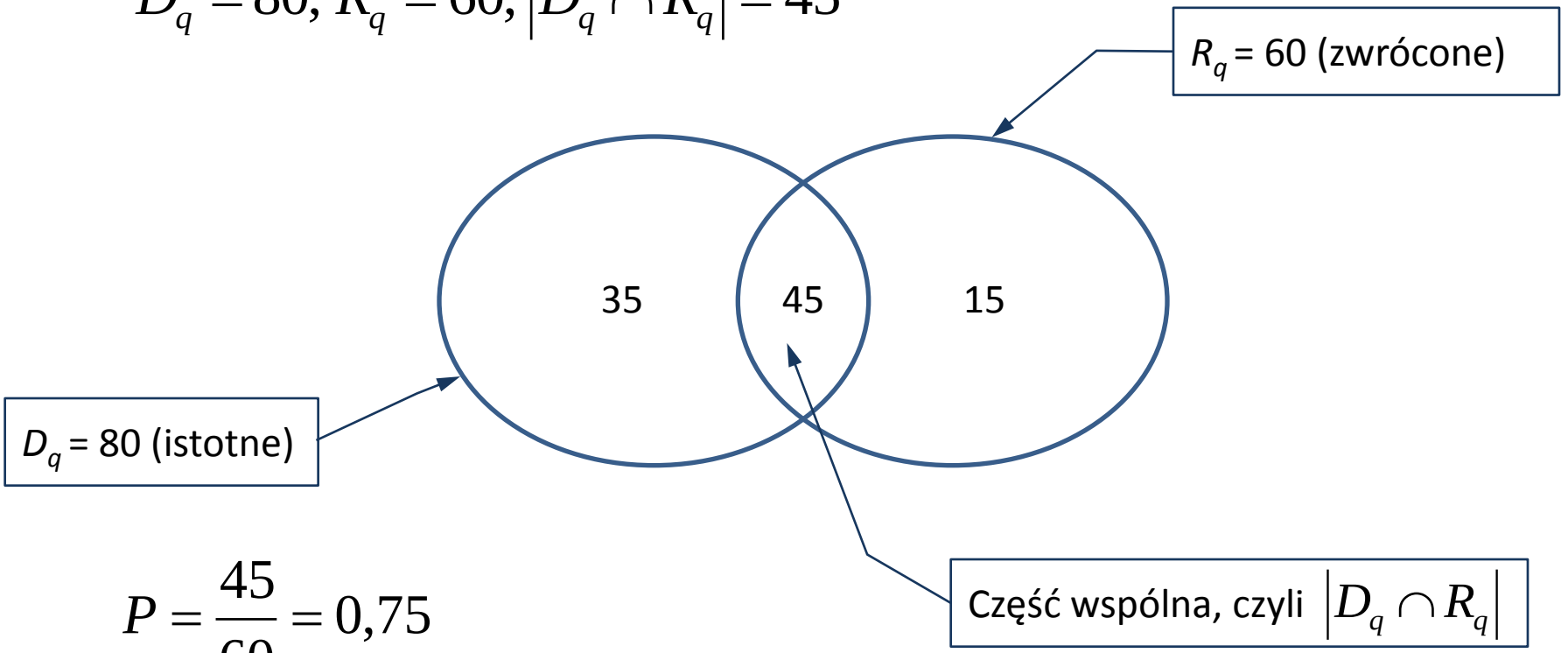
$$R = \frac{|D_q \cap R_q|}{|D_q|} = \frac{15}{20} = 0{,}75$$



Ocena jakości wyszukiwania

- Przykład 2

$$D_q = 80,\ R_q = 60,\ \left|D_q \cap R_q\right| = 45$$



$$P = \frac{45}{60} = 0{,}75$$

$$R = \frac{45}{80} = 0{,}56$$

Tworzenie krzywej P-R

- Algorytm
  - załóżmy, że odpowiedź na zapytanie *q* jest uporządkowaną listą dokumentów (np. wg miary kosinusowej):
    $R_q=(d_1, d_2, \dots, d_m)$
  - następnie używając zbioru istotnych dokumentów D_q dla każdego $d_i \in R_q$ możemy obliczyć jego wagę w_i jako wartość binarną

$$w_i = \begin{cases} 1 \; dla \; d_i \in D_q \\ 0 \; w \; przeciwnym \; wypadku \end{cases}$$

  - *niech* $k>0$ oznacza liczbę dokumentów z początku listy R_q które będziemy rozważać. Zatem definiujemy dokładność obciętą do *k* początkowych dokumentów

$$P(k) = \frac{1}{k} \sum_{i=1}^{k} w_i$$

oraz kompletność obciętą do k początkowych dokumentów.
Parametr *k* pozwala nam zobaczyć, jak zmieniają się wartości *P* oraz *R* wraz ze wzrostem *k* (czyli niższym miejscem rankingowym)

$$R(k) = \frac{1}{|D_q|} \sum_{i=1}^{k} w_i$$

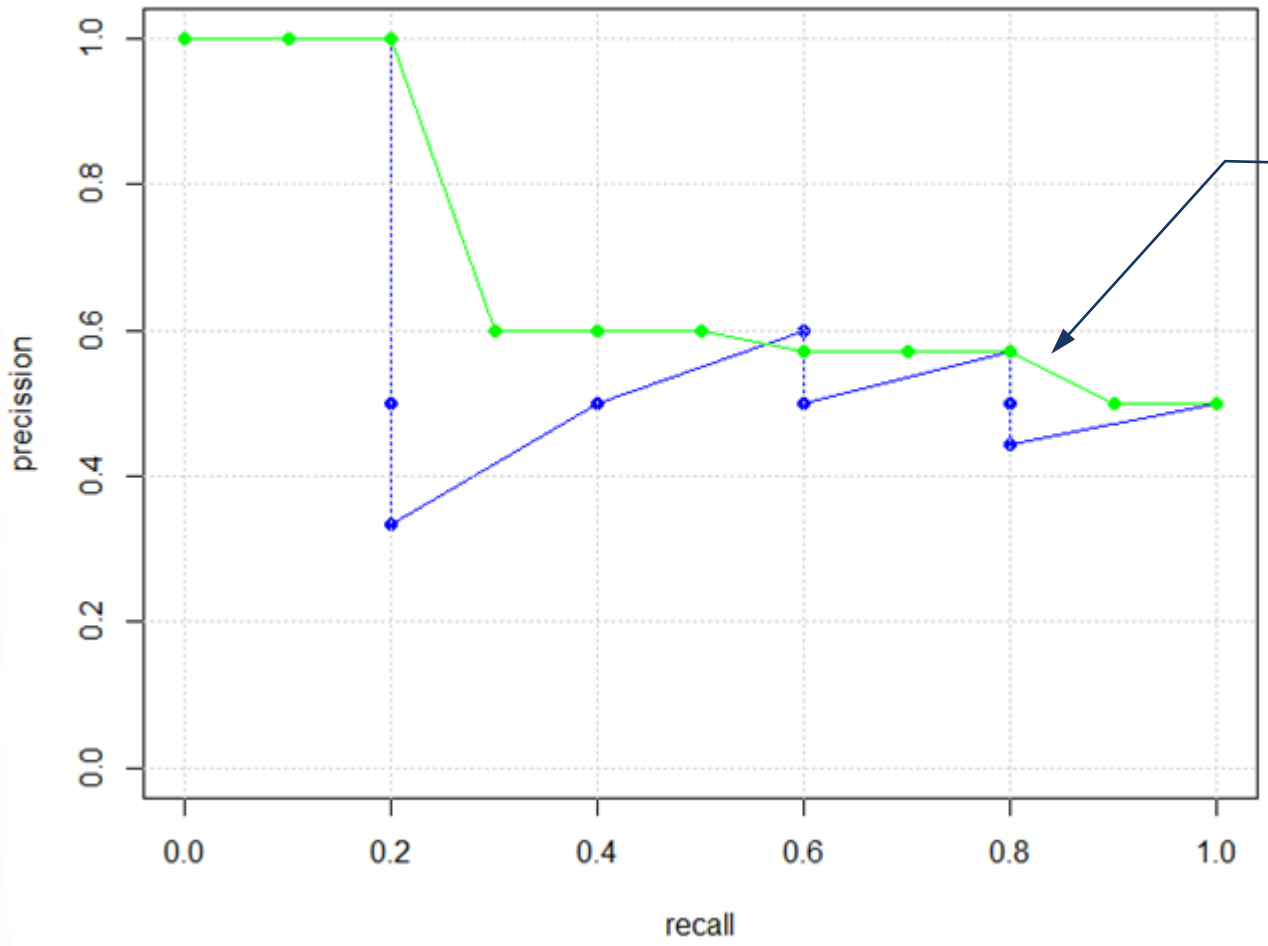
Tworzenie krzywej P-R

- Przykład. Niech pierwszych k=10 wyników wyszukiwania to dokument D1 do D10. $|D_q|$=5

dokument	w_k	$R(k)$	$P(k)$	k	cos α
D1	1	1 /5·1 = 0,2	1/1·1 = 1	1	1
D2	0	1 /5·1 = 0,2	1/2·1 = 0,5	2	0,92
D3	0	1 /5·1 = 0,2	1/3·1= 0,33	3	0.90
D4	1	1/5·2 = 0,4	1/4·2 = 0,5	4	0.7
D5	1	1/5·3 = 0,6	1/5·3 = 0,6	5	0,66
D6	0	1/5·3 = 0,6	1/6·3 = 0,5	6	0,5
D7	1	1/5·4 = 0,8	1/7·4 = 0,57	7	0,4
D8	0	1/5·4 = 0,8	1/8·4 = 0,5	8	0,33
D9	0	1/5·4 = 0,8	1/9·4 = 0,44	9	0,25
D10	1	1/5·5 = 1,0	1/10·5 = 0,5	10	0,11

Tworzenie krzywej P-R

- Wykres dla danych z przykładu



Krzywa interpolowana wg wzoru jak niżej.
Wyznaczamy zwykle dla punktów
r = {0,1 0,2 … 0,9 1}

$$P_{interp}(r) = \max_{r' \geq r} P(r')$$

Tworzenie krzywej P-R

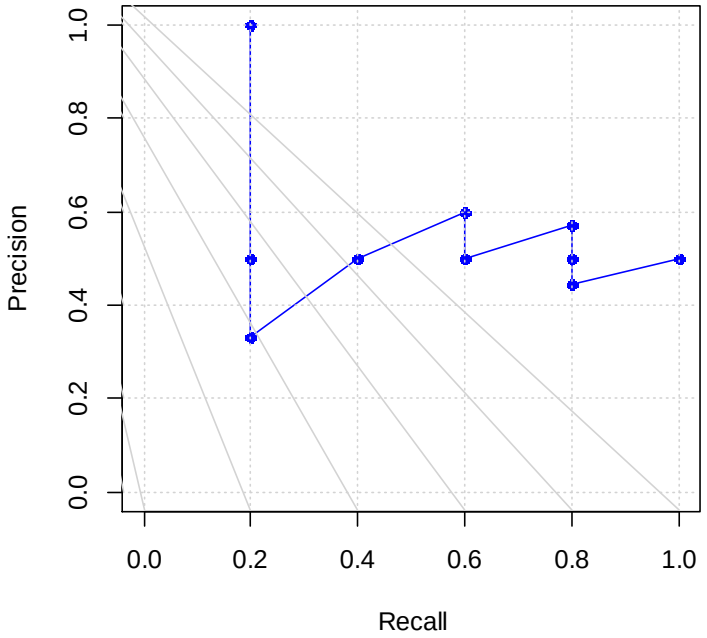
- Średnia dokładność
  - miara ta łączy dokładność i kompletność a także ocenia uporządkowanie dokumentów
  - $\acute{S}_d$ = 1 gdy:
    - ✓ wszystkie istotne dokumenty zostaną odnalezione
    - ✓ oraz znajdują się one na liście rankingowej przed dokumentami nieistotnymi
  - dla naszego przykładu mamy

$$\acute{S}_d = \frac{1}{5}\left(1\cdot 1 + 1\cdot 0{,}5 + 1\cdot 0.6 + 1\cdot 0.57 + 1\cdot 0{,}5\right) = 0{,}634$$

Tworzenie krzywej P-R

- Inne przykłady

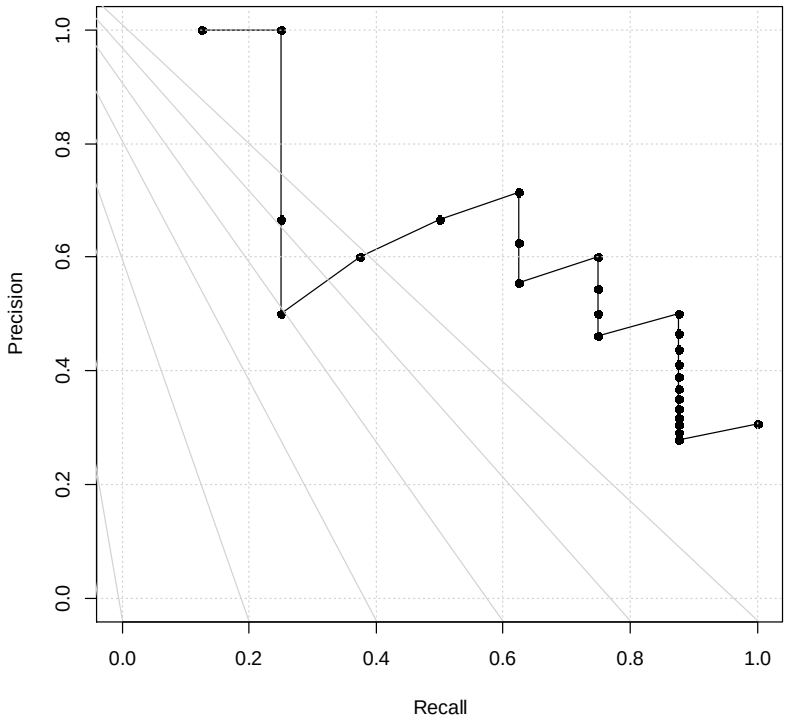
```
Wynik dla query1:
 1 0 0 1 1 0 1 0 0 1

1 – relevant
0 – not relevant (retrieved)

Precision: 1, 1/2, 1/3, 2/4, 3/5, 3/6, 4/7, 4/8, 4/9, 5/10
Recall: 1/5, 1/5, 1/5, 2/5, 3/5, 3/5, 4/5, 4/5, 4/5, 5/5
```

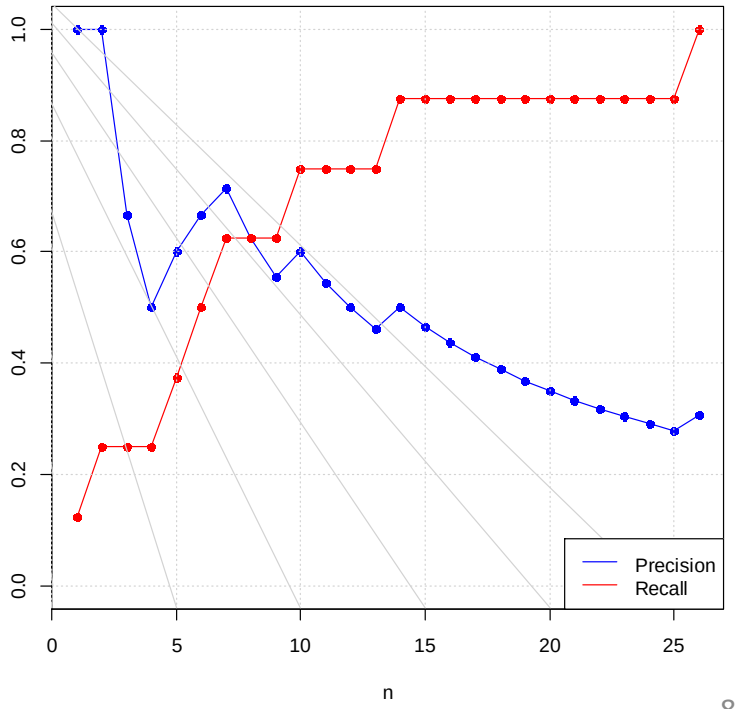


Tworzenie krzywej P-R

- Inne przykłady

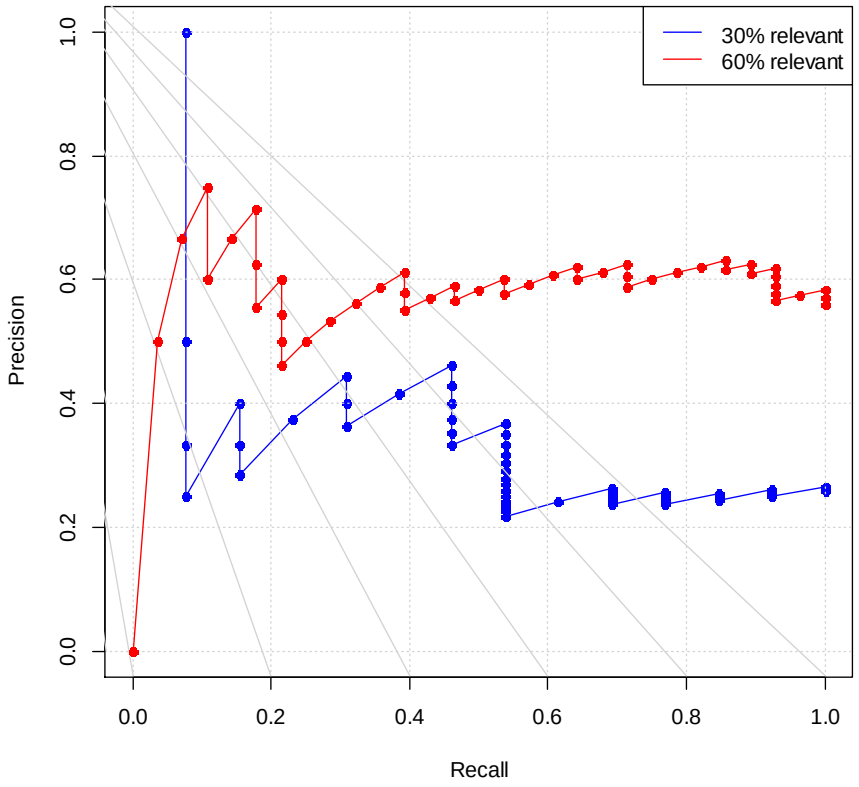
```
Wynik dla query1:
  1 1 0 0 1 1 1 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1

Precision: 1, 1, 2/3, 2/4, 3/5, ..., 8/26
Recall:    1/8, 2/8, 2/8, 2/8, 3/8, 4/8, 5/8, 5,8, ..., 8/8
```





Tworzenie krzywej P-R

```
(około 30% dokumentów relevant)
Wynik dla query1:
  1 0 0 0 1 0 0 1 1 0 0 1 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1
  0 0 0 0 1 0 0 0 1 0 0 1 0 0 1 0

(około 60% dokumentów relevant)
Wynik dla query2:
  0 1 1 1 0 1 1 0 0 1 0 0 0 1 1 1 1 1 0 0 1 1 0 1 1 0 1 1 1 0 1 1 0 0
  1 1 1 1 0 1 0 1 0 0 0 0 1 1 0 0
```



Tworzenie krzywej P-R

- Miara F-measure
  - https://en.wikipedia.org/wiki/Precision_and_recall

F-measure [edit]

Main article: F1 score

A measure that combines precision and recall is the harmonic mean of precision and recall, the traditional F-measure or balanced F-score:

$$F = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

This measure is approximately the average of the two when they are close, and is more generally the harmonic mean, which, for the case of two numbers, coincides with the square of the geometric mean divided by the arithmetic mean. There are several reasons that the F-score can be criticized in particular circumstances due to its bias as an evaluation metric.[1] This is also known as the F_1 measure, because recall and precision are evenly weighted.

It is a special case of the general F_β measure (for non-negative real values of β):

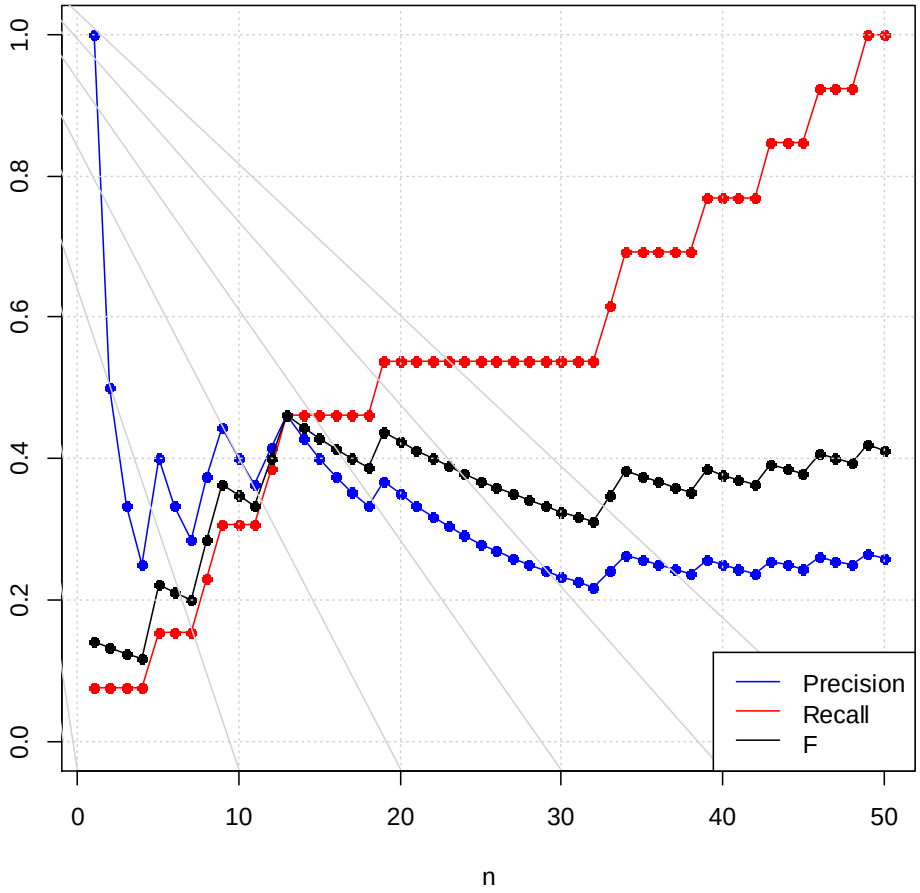
$$F_\beta = (1 + \beta^2) \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\beta^2 \cdot \mathrm{precision} + \mathrm{recall}}$$

Two other commonly used F measures are the F_2 measure, which weights recall higher than precision, and the $F_{0.5}$ measure, which puts more emphasis on precision than recall.

Tworzenie krzywej P-R

- Miara F-measure

```
F = (2 * Precision * Recall) / (Precision + Recall)

(średnia harmoniczna Precision oraz Recall)
```

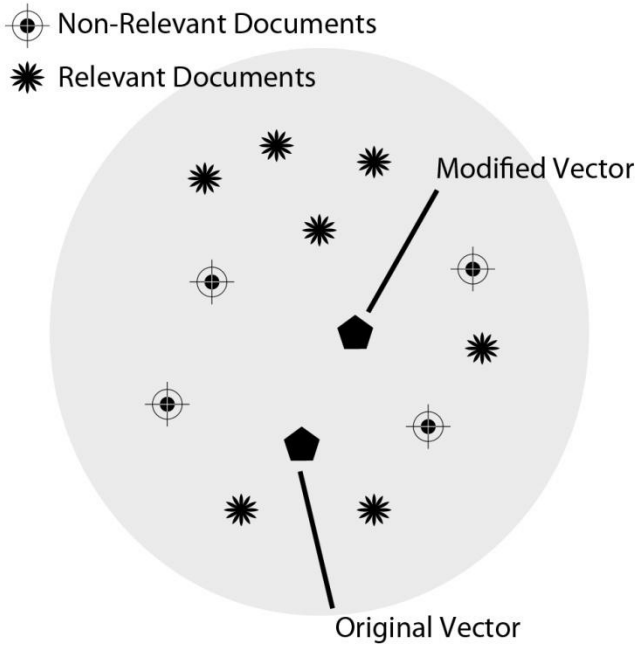


Relevance Feedback

- Motywacja: zapytania według słów kluczowych są często niekompletne i/lub niejednoznaczne. Zapytania należałoby doszczegółowić. Ale trudno użytkownika do tego zmusić. Jak rozwiązać problem:
  - wydane przez użytkownika zapytanie jest zostanie automatycznie doprecyzowane. Wymagana jest jednak interakcja z użytkownikiem, który oceni zwrócone dokumenty pod względem ich istotności. Stąd nazwa relevance feedback
  - po automatycznym przeformułowaniu zapytania system ponownie zwraca zmodyfikowaną listę dokumentów istotnych
- Problemy
  - metoda w zasadzie obecnie o znaczeniu teoretycznym, bo … nie możemy kazać użytkownikom coś robić za nas. Użytkownik oczekuje natychmiastowej odpowiedzi na swoje pytanie (nawet bardzo nieprecyzyjnie zadane)
  - pewne rozwiązanie: Pseudorelevance feedback, pierwsze *k* dokumentów w rankingu jest uważanych za istotne i na tej podstawie zapytanie jest automatycznie reformułowane
- Klasyczna metoda Rocchio

Relevance Feedback

- Klasyczna metoda Rocchio
  - istotą tej metody jest przesunięcie wektora zapytań w kierunki centroidu dokumentów istotnych i oddalenie od centroidu dokumentów nieistotnych



źródło: https://en.wikipedia.org/wiki/Rocchio_algorithm

Relevance Feedback

- Kroki do wykonania
  - Krok 1: użytkownik zadaje zapytanie
  - Krok 2: system zwraca wyniki najbardziej zgodne z wydanym zapytaniem
  - Krok 3: użytkownik oznacza wybrane dokumenty jako istotne, a inne jako nieistotne.
    - ✓ to jest najbardziej dyskusyjna część RF-a !
  - Krok 4: system zwraca zmodyfikowaną listę dokumentów na podstawie automatycznie przeformułowanego zapytania

Relevance Feedback

- Metoda Rocchio
  - uaktualniamy wektor zapytania za pomocą liniowej kombinacji poprzedniego wektora zapytań oraz wiedzy o istotności (D_r) lub nieistotności (D_{nr}) zwróconych wcześniej dokumentów
  - uwaga 1: ponieważ zbiór zwróconych dokumentów jest zwykle duży można arbitralnie ustalić, że np. D_r to będzie 10 pierwszych dokumentów a reszta to będą D_{nr}
  - uwaga 2: można też przyjąć, że nie będziemy używać dokumentów D_{nr} i wówczas ustawiamy $\gamma = 0$

$$\mathbf{q}' = \alpha\mathbf{q} + \beta\frac{1}{|D_r|}\sum_{d_j \in D_r}\mathbf{d}_j - \gamma\frac{1}{|D_{nr}|}\sum_{d_j \in D_{nr}}\mathbf{d}_j$$

D_r - zbiór istotnych dokumentów (relevant)

D_{nr} - zbiór nieistotnych dokumentów (non-relevant)

$\alpha\ \beta\ \gamma$ - parametry

Relevance Feedback

- Przykład
  - dane z książki „Eksploracja zasobów internetowych”
  - macierz TDM ograniczamy tylko do 5 termów:
    lab, laboratory, programming, computer, program
    Miary IDF dla tych termów to odpowiednio:
    4.3923, 4.3923, 4.3923, 1.8074, 0.6919
  - wektora zapytania *q* = {*computer, program*}
  - wektory (wierszami) normalizujemy. Wówczas łatwiej jest obliczyć odległość kosinusową. Jest nią iloczyn skalarny *q* oraz *d*

$$\cos\alpha = \frac{a \cdot b}{|a|\,|b|}$$

$$|a| = 1, |b| = 1 \quad \Rightarrow \quad \cos\alpha = a \cdot b$$

Relevance Feedback

- Przykład
  - TDM przed i po normalizacji
  - ostatni wiersz to nasze zapytanie *q*

Niektóre wektory wglądają jak wektory binarne. Tak działa normalizacja dla wektorów, które mają tylko jedną wartość niezerową

Docs	lab	laboratory	programming	computer	program
1	0.000	0.000	0.000	0.000	**1.000**
2	0.000	0.000	0.000	0.000	**1.000**
3	0.000	**0.979**	0.000	**0.202**	0.000
4	0.000	0.000	0.000	**0.794**	**0.608**
5	0.000	0.000	0.000	0.000	**1.000**
6	0.000	0.000	**0.688**	**0.708**	**0.163**
7	0.000	0.000	0.000	0.000	**1.000**
8	0.000	0.000	0.000	0.000	**1.000**
9	0.000	0.000	0.000	0.000	0.000
10	0.000	0.000	0.000	0.000	0.000
11	0.000	0.000	0.000	0.000	0.000
12	0.000	0.000	0.000	**1.000**	0.000
13	0.000	0.000	0.000	0.000	0.000
14	**0.915**	0.000	0.000	**0.377**	**0.144**
15	0.000	0.000	0.000	0.000	**1.000**
16	0.000	0.000	0.000	0.000	**1.000**
17	0.000	0.000	0.000	0.000	**1.000**
18	0.000	0.000	0.000	0.000	0.000
19	0.000	0.000	0.000	0.000	**1.000**
20	0.000	0.000	0.000	0.000	0.000
21	0.000	0.000	0.000	**0.934**	**0.358**

normalizacja

Docs	lab	laboratory	programming	computer	program
1	0.000	0.00	0.00	0.000	**0.0091**
2	0.000	0.00	0.00	0.000	**0.0073**
3	0.000	**0.11**	0.00	**0.023**	0.0000
4	0.000	0.00	0.00	**0.035**	**0.0271**
5	0.000	0.00	0.00	**0.000**	**0.0091**
6	0.000	0.00	**0.13**	**0.131**	**0.0301**
7	0.000	0.00	0.00	0.000	**0.0271**
8	0.000	0.00	0.00	0.000	**0.0103**
9	0.000	0.00	0.00	0.000	0.0000
10	0.000	0.00	0.00	0.000	0.0000
11	0.000	0.00	0.00	0.000	0.0000
12	0.000	0.00	0.00	**0.033**	0.0000
13	0.000	0.00	0.00	0.000	0.0000
14	**0.056**	0.00	0.00	**0.023**	**0.0088**
15	0.000	0.00	0.00	0.000	**0.0294**
16	0.000	0.00	0.00	0.000	**0.0083**
17	0.000	0.00	0.00	0.000	**0.0096**
18	0.000	0.00	0.00	0.000	0.0000
19	0.000	0.00	0.00	0.000	**0.0115**
20	0.000	0.00	0.00	0.000	0.0000
21	0.000	0.00	0.00	**0.904**	**0.3459**

bez normalizacji

Relevance Feedback

- Przykład
  - wynik miary kosinusowej (dla wariantu z oraz bez znormalizowanego)

Docs	[,1]	ranga
1	0.358	5
2	0.358	5
3	0.188	6
4	0.959	**1**
5	0.358	5
6	0.719	**3**
7	0.358	5
8	0.358	5
9	0.000	–
10	0.000	–
11	0.000	–
12	0.934	**2**
13	0.000	–
14	0.403	**4**
15	0.358	5
16	0.358	5
17	0.358	5
18	0.000	–
19	0.358	5
20	0.000	–

normalizacja

Docs	[,1]	ranga
1	0.00315	11
2	0.00252	13
3	0.02067	5
4	0.04141	**2**
5	0.00315	11
6	0.12876	**1**
7	0.00939	7
8	0.00357	9
9	0.00000	–
10	0.00000	–
11	0.00000	–
12	0.02970	**3**
13	0.00000	–
14	0.02370	**4**
15	0.01018	6
16	0.00288	12
17	0.00332	10
18	0.00000	–
19	0.00399	8
20	0.00000	–

bez normalizacji

D1	Anthropology
D2	Art
D3	Biology
D4	Chemistry
D5	Communication
D6	Computer Science
D7	Criminal Justice
D8	Economics
D9	English
D10	Geography
D11	History
D12	Math
D13	Modern Languages
D14	Music
D15	Philosophy
D16	Physics
D17	Political Sciences
D18	Psychology
D19	Sociology
D20	Theatre

Ranking nie jest identyczny ale dość podobny

Relevance Feedback

- Przypomnienie wzoru

$$\mathbf{q}' = \alpha\mathbf{q} + \beta\frac{1}{|D_r|}\sum_{d_j \in D_r}\mathbf{d}_j - \gamma\frac{1}{|D_{nr}|}\sum_{d_j \in D_{nr}}\mathbf{d}_j$$

- Parametry i obliczenia
- $\alpha = 1; \beta = 0{,}8; \gamma = 0$

$$\mathbf{q}' = \mathbf{q} + \frac{0{,}8}{4}(\mathbf{d_4} + \mathbf{d_6} + \mathbf{d_{12}} + \mathbf{d_{14}})$$

γ=0, eliminujemy dokumenty niezgodne z obliczeń

Nowy wektor q':

lab	laboratory	programming	computer	program
0.172	0.000	0.129	1.474	0.529

Nowy wektor q' po normalizacji:

lab	laboratory	programming	computer	program
0.1086	0.0000	0.0816	0.9325	0.3347

Relevance Feedback

- Wynik (w nawiasach wartości poprzednie)
- Wnioski
  - trudno ocenić jednoznacznie, „czy jest lepiej”, przykładowo:
    - ✓ ranga dokumenty D6 (Computer Science) jest nieco wyższa, to na plus dajemy
    - ✓ ranga dokumentu D14 (Music) podniosła się, to raczej na minus
  - pokazujemy tą metodę, jako ilustrację pewnej techniki poprawy działania wyszukiwarki. Pamiętajmy jednak, że ta prosta metoda nie jest specjalnie skuteczna a zagadnienie skutecznego modyfikowania zapytań wydawanych przez użytkowników jest niezwykle złożonym zagadnieniem

Docs	[,1]	poprz.	ranga
1	0.335		
2	0.335		
3	0.188		
4	0.944	(0.959)	**1**
5	0.335		
6	0.770	(0.719)	**3**
7	0.335		
8	0.335		
9	0.000		
10	0.000		
11	0.000		
12	0.932	(0.934)	**2**
13	0.000		
14	0.499	(0.403)	**4**
15	0.335		
16	0.335		
17	0.335		
18	0.000		
19	0.335		
20	0.000		

Wektory, macierze

Wektory i macierze

- Konwencje zapisu

$$\mathbf{A}=\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}_{m\times n} \in R^{m\times n} \quad \mathbf{x}=\begin{bmatrix} x_1 \\ \\ x_n \end{bmatrix}_{n\times 1} \quad \mathbf{x}^T=\begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix}_{1\times n}$$

- Mnożenie wektor-wektor
  - iloczyn zewnętrzny (ang. outer product)

$$\mathbf{x}\mathbf{y}^T=\begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}_{m\times 1} \begin{bmatrix} y_1 & \cdots & y_n \end{bmatrix}_{1\times n}=\begin{bmatrix} x_1y_1 & \cdots & x_1y_n \\ \vdots & & \vdots \\ x_my_1 & \cdots & x_my_n \end{bmatrix}_{m\times n}$$

  - iloczyn skalarny (ang. inner product) dozwolone tylko, gdy $m = n$)

$$\mathbf{x}^T\mathbf{y}=\begin{bmatrix} x_1 & \cdots & x_m \end{bmatrix}_{1\times m}\begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}_{n\times 1}=\begin{bmatrix} x_1y_1+\ldots+x_my_n \end{bmatrix}_{1\times 1}$$

Wektory i macierze

- Mnożenie macierz-wektor

$$\mathbf{y}_{m\times 1} = \mathbf{A}_{m\times n}\mathbf{x}_{n\times 1} \qquad y_i = \sum_{j=1}^{n} a_{ij}x_j \;\; i = 1,\ldots,m$$

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}_{2\times 2}\begin{bmatrix} 5 \\ 6 \end{bmatrix}_{2\times 1} = \begin{bmatrix} 1\cdot 5 + 2\cdot 6 \\ 3\cdot 5 + 4\cdot 6 \end{bmatrix} = \begin{bmatrix} 17 \\ 39 \end{bmatrix}_{2\times 1}$$

- Mnożenie macierz-macierz
  - w ogólności mnożenie macierzy nie jest przemienne

$$\mathbf{C}_{m\times k} = \mathbf{A}_{m\times n}\mathbf{B}_{n\times k}$$

$$\begin{bmatrix} 1 & 0 & 2 \\ -1 & 3 & 1 \end{bmatrix}_{2\times 3}\begin{bmatrix} 3 & 1 \\ 2 & 1 \\ 1 & 0 \end{bmatrix}_{3\times 2} = \begin{bmatrix} (1\cdot 5 + 0\cdot 2 + 2\cdot 1) & (1\cdot 2 + 0\cdot 1 + 2\cdot 0) \\ (-1\cdot 3 + 3\cdot 2 + 1\cdot 1) & (-1\cdot 1 + 3\cdot 1 + 1\cdot 0) \end{bmatrix}_{2\times 2} = \begin{bmatrix} 5 & 1 \\ 4 & 2 \end{bmatrix}_{2\times 2}$$

Wektory i macierz

- Norma wektora
  - liczba nieujemna, uogólnienie długości wektora („rozmiar wektora”)

$$\|\mathbf{x}\|_1 = \sum_{i=1}^{n} |x_i| \quad \text{norma - 1}$$

$$\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2} \quad \text{norma - 2, norma Euklidesowa}$$

$$\|\mathbf{x}\|_\infty = \max_{1 \le i \le n} |x_i| \quad \text{norma - nieskończoność}$$

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p} \quad \text{wzór ogólny}$$

  - własności normy wektora

$$\|\mathbf{x}\| \ge 0$$

$$\|\mathbf{x}\| \ge 0 \Leftrightarrow \mathbf{x} = 0$$

$$\|a\mathbf{x}\| = |a| \|\mathbf{x}\|$$

$$\|\mathbf{x} + \mathbf{y}\| \le \|\mathbf{x}\| + \|\mathbf{y}\|, \text{ nierówność trójkąta}$$

Wektory i macierz

- Norma macierzy
  - naturalny odpowiednik normy wektorowej dla macierzy

$$\|\mathbf{A}\|_1 = \max_{1\le j\le n}\sum_{i=1}^{m}|a_{ij}|$$ norma-1, największa norma „po kolumnach”

$$\|\mathbf{A}\|_\infty = \max_{1\le i\le m}\sum_{j=1}^{n}|a_{ij}|$$ norma-nieskończoność, największa norma „po wierszach”

$$\|\mathbf{A}\|_2 = \left(\max_{1\le i\le n}\sigma_{\max}(\mathbf{A}^T\mathbf{A})\right)^{1/2}$$ tzw. norma spektralna, największa wartość własna macierzy $\boldsymbol{A^TA}$

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}a_{ij}^2}$$ norma Frobeniusa, pierwiastek z sumy kwadratów elementów macierzy

$$\text{Zawsze zachodzi, że } \|\mathbf{A}\|_2 \le \|\mathbf{A}\|_F$$

Ortogonalność

- Ortogonalność wektorów i macierzy (uogólnienie pojęcia prostopadłości)
  - dwa wektory **x**, **y** są ortogonalne, gdy $\mathbf{x}^T\mathbf{y}=0$
  - gdy wektory są znormalizowane (czyli ich długość wynosi 1), mówimy, że wektory są ortonormalne
- Macierz ortogonalna
  - macierz kwadratowa, w której kolumny oraz wiersze są ortogonalne
  - zachodzi:

$$\mathbf{Q}^T\mathbf{Q} = \mathbf{Q}\mathbf{Q}^T = \mathbf{I}_n \quad (\mathbf{I}_n - \text{macierz jednostkowa})$$

$$\mathbf{I}_n = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

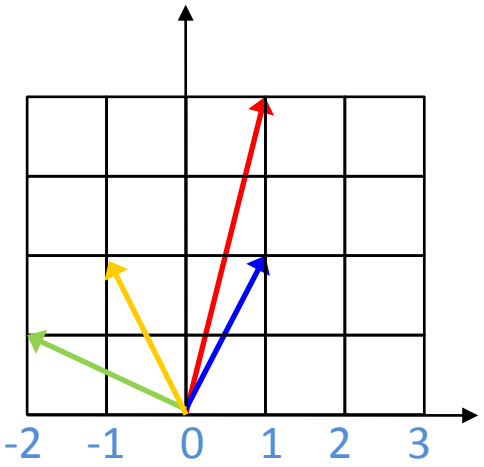
$$\mathbf{Q}^{-1} = \mathbf{Q}^T - \text{banalnie łatwo wyznacza się macierz odwrotną}$$

Przekształcanie liniowe wektorów

- Nazwy angielskie: eigenvectors, eigenvalues (niem. eigen – właściwy, charakterystyczny)
- Wektory przekształcane przez macierz

$$\mathbf{A}=\begin{bmatrix}1 & 0\\ 0 & 2\end{bmatrix},\mathbf{v}=\begin{bmatrix}1\\ 2\end{bmatrix}$$

$$\mathbf{w}=\mathbf{A}\mathbf{v}=\begin{bmatrix}1 & 0\\ 0 & 2\end{bmatrix}\begin{bmatrix}1\\ 2\end{bmatrix}=\begin{bmatrix}1\cdot 1+0\cdot 2\\ 0\cdot 1+2\cdot 2\end{bmatrix}=\begin{bmatrix}1\\ 4\end{bmatrix}$$



  - obrót -90 stopni

$$\mathbf{A}=\begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix},\mathbf{w}=\mathbf{A}\mathbf{v}=\begin{bmatrix}-2\\ 1\end{bmatrix}$$

  - odbicie lustrzane

$$\mathbf{A}=\begin{bmatrix}-1 & 0\\ 0 & 1\end{bmatrix},\mathbf{w}=\mathbf{A}\mathbf{v}=\begin{bmatrix}-1\\ 2\end{bmatrix}$$

Przekształcanie liniowe wektorów

- Uwaga: kolejność przekształceń ma znaczenie, przykładowo:
  - 1) obrót = -90 stopni, 2) odbicie lustrzane

$$\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}=\begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}$$

  - 1) odbicie lustrzane, 2) obrót = -90 stopni

$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}=\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

  - ogólna postać macierzy obrotu

$$\begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix}$$

  - np. obrót o 30 stopni:

$$\begin{bmatrix} 0{,}9 & -0{,}5 \\ 0.5 & 0{,}9 \end{bmatrix}$$

W 3 wymiarach – dowolne obroty

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

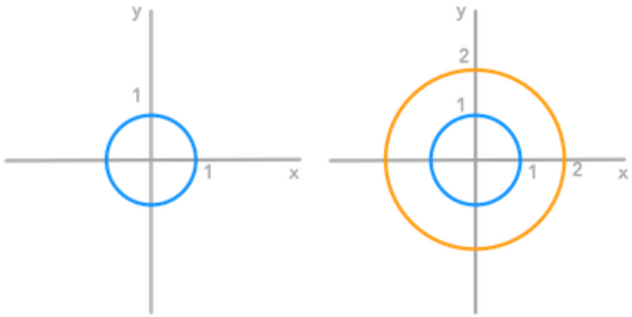
$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}$$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

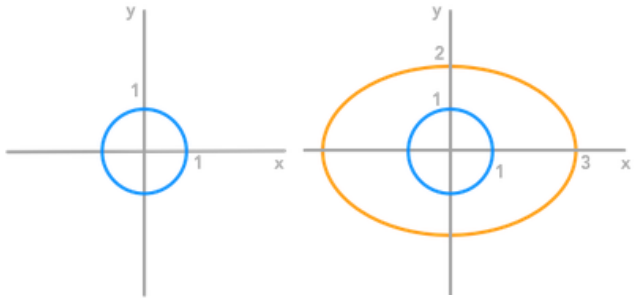
Przekształcanie liniowe wektorów

- Krok po kroku (na przykładzie przekształcania okręgu jednostkowego)

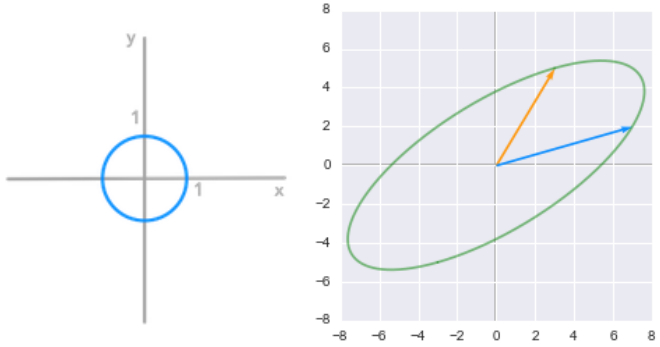
$$\mathbf{A} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$



$$\mathbf{A} = \begin{bmatrix} 3 & 0 \\ 0 & 2 \end{bmatrix}$$



$$\mathbf{A} = \begin{bmatrix} 3 & 7 \\ 5 & 2 \end{bmatrix}$$



Wektory i wartości własne

- Pytanie: czy dla danego odwzorowania danego macierzą kwadratową A może istnieć wektor, który po przekształceniu przez tą macierz będzie:
  - taki sam lub ewentualnie
  - przeskalowany o (+ lub -) jakąś wartość ?

$$\mathbf{w} = \mathbf{A}\mathbf{v} \Rightarrow \mathbf{w} = \mathbf{v}$$

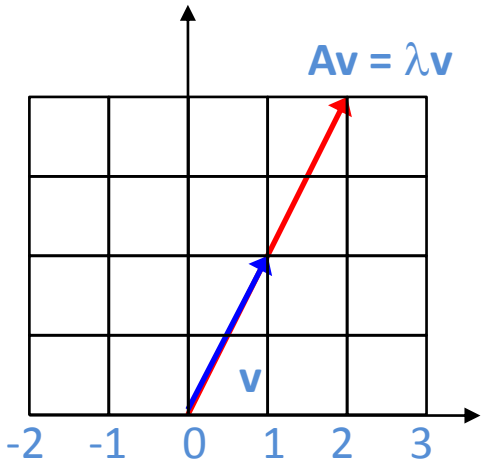
lub

$$\mathbf{w} = \mathbf{A}\mathbf{v} \Rightarrow \mathbf{w} = \lambda\mathbf{v}$$

czyli możemy to wyrazić o tak:

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$$

$$\mathbf{A} = \begin{bmatrix} 1 & 0{,}5 \\ 0 & 2 \end{bmatrix}, \mathbf{v} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \mathbf{A}\mathbf{v} = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$$



λ>1 → zwiększa długość wektora
0<λ<1 → zmniejsza długość wektora
λ<0 → zmiana zwrotu
λ<-1 → zwiększa długość + zmiana zwrotu
λ=1 → **v = w**

Wektory i wartości własne

- Stała λ, która odpowiednio skaluje wektor nazywa się wartością własną
- Każda wartość własna ma przypisany odpowiedni wektor własny **v**
- Mamy więc do rozwiązania takie równanie:

$$\mathbf{Av} = \lambda\mathbf{v}$$

$$\mathbf{Av} - \lambda\mathbf{v} = 0$$

$$(\mathbf{A} - \mathbf{I}\lambda)\mathbf{v} = 0$$

Nie wnikając w szczegóły, aby rozwiązać to zadanie musimy rozwiązać następujące równanie

$$\det(\mathbf{A} - \mathbf{I}\lambda) = 0$$

Wektory i wartości własne

- Przykład

$$\mathbf{A} = \begin{bmatrix} 1 & 0{,}5 \\ 0 & 2 \end{bmatrix}$$

$$\det(\mathbf{A} - \mathbf{I}\lambda) = \det\left(\begin{vmatrix} 1-\lambda & 0{,}5 \\ 0 & 2-\lambda \end{vmatrix} - \begin{vmatrix} 1-\lambda & 0{,}5 \\ 0 & 2-\lambda \end{vmatrix} \lambda \right) =$$

$$\begin{vmatrix} 1-\lambda & 0{,}5 \\ 0 & 2-\lambda \end{vmatrix} = (1-\lambda)(2-\lambda) = \lambda^2 - 3\lambda + 2 = 0$$

Mamy "szkolne" równanie kwadratowe. Po rozwiązaniu otrzymujemy

$\lambda_1 = 1$

$\lambda_2 = 2$

Mamy więc już wartości własne, teraz szukamy wektorów własnych odpowiadających danym wartościom własnym

Wektory i wartości własne

Dla $\lambda_1 = 1$

$$(\mathbf{A} - \lambda_1 \mathbf{I})\mathbf{v} = 0$$

$$\left(\begin{bmatrix} 1 & 0{,}5 \\ 0 & 2 \end{bmatrix} - 1 \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)\begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} = 0$$

Mamy "szkolny" układ dwóch równań, czyli

$$\begin{cases} 0 \cdot \mathbf{v}_1 + 0{,}5 \cdot \mathbf{v}_2 = 0 \\ 0 \cdot \mathbf{v}_1 + 1 \cdot \mathbf{v}_2 = 0 \end{cases} \rightarrow \begin{array}{l} \mathbf{v}_1 \in R \\ \mathbf{v}_2 = 0 \end{array}$$

Czyli otrzymaliśmy rozwiązanie, że wektorem własnym może być np. $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ ale też np. $\begin{bmatrix} 102 \\ 0 \end{bmatrix}$

Dla $\lambda_2 = 2$

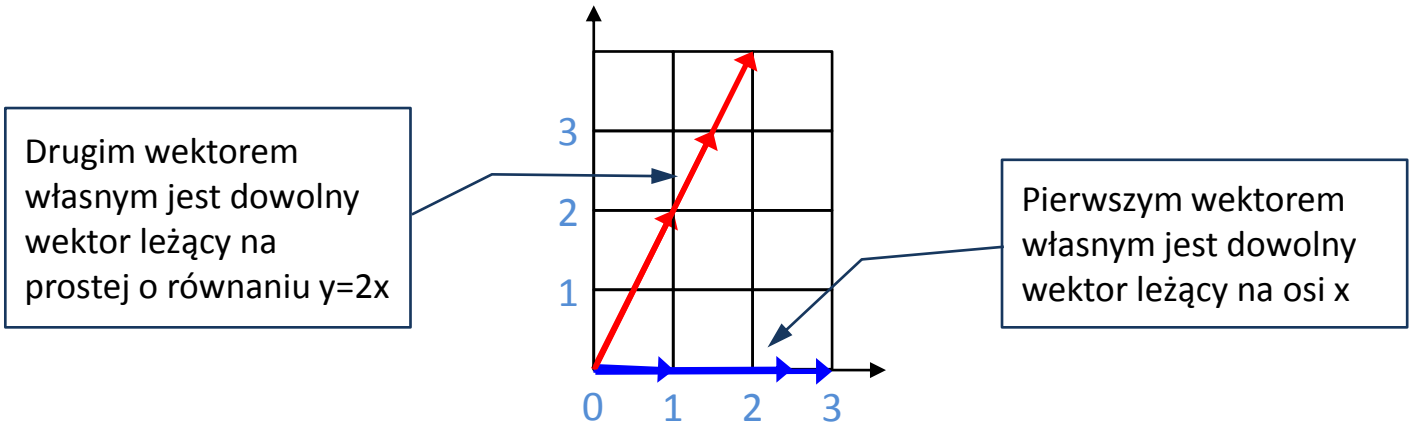
$$(\mathbf{A} - \lambda_2 \mathbf{I})\mathbf{v} = 0$$

$$\left(\begin{bmatrix} 1 & 0{,}5 \\ 0 & 2 \end{bmatrix} - 2 \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)\begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} = 0$$

Mamy "szkolny" układ dwóch równań, czyli

$$\begin{cases} -1 \cdot \mathbf{v}_1 + 0{,}5 \cdot \mathbf{v}_2 = 0 \\ 0 \cdot \mathbf{v}_1 + 1 \cdot \mathbf{v}_2 = 0 \end{cases} \rightarrow \mathbf{v}_2 = 2\mathbf{v}_1$$

Czyli otrzymaliśmy rozwiązanie, że wektorem własnym jest dowolny wektor leżący na prostej $y = 2x$



Wektory i wartości własne

- Gdy macierz jest diagonalna, to zadanie znalezienia wartości i wektorów własnych jest banalnie proste

$$\mathbf{A}=\begin{bmatrix}2 & 0 & 0\\ 0 & 9 & 0\\ 0 & 0 & 4\end{bmatrix}\Rightarrow \mathbf{v}_1=\begin{bmatrix}0\\1\\0\end{bmatrix},\quad \mathbf{v}_1=\begin{bmatrix}0\\0\\1\end{bmatrix},\quad \mathbf{v}_1=\begin{bmatrix}1\\0\\0\end{bmatrix}$$

$$\lambda_1=9,\quad \lambda_2=4,\quad \lambda_3=2$$

Wektory i wartości własne

- Przykład. Ten co na poprzednich stronach ale obliczenia wykonano w R

```
A <- matrix(c(1, 0, 0.5, 2), nrow = 2, ncol = 2, byrow = FALSE)
A
     [,1] [,2]
[1,]    1  0.5
[2,]    0  2.0
# wektory własne są normalizowane (długość = 1)
# wartości własne są sortowane malejąco

eigen(A)

$values
[1] 2 1

$vectors
          [,1] [,2]
[1,] 0.4472136    1
[2,] 0.8944272    0
```

Wektory i wartości własne

- Przykład dla macierzy 3 x 3
  - uwaga: wyznacznik dla macierzy większej niż 3 x 3 nie policzymy już tak łatwo metodami „szkolnymi”. Wykorzystywane są wówczas (matematycznie mocno zaawansowane) metody z algebry liniowej. Pomijamy te zagadnienia, jako całkowicie wykraczające poza zakres tematyczny przedmiotu

```
> A <- matrix(c(2, 0, 0, 0, 3, 4, 0, 4, 9), nrow = 3, ncol = 3)
     [,1] [,2] [,3]
[1,]    2    0    0
[2,]    0    3    4
[3,]    0    4    9
eigen(A)
$values
[1] 11  2  1

$vectors
          [,1] [,2]       [,3]
[1,] 0.0000000    1  0.0000000
[2,] 0.4472136    0  0.8944272
[3,] 0.8944272    0 -0.4472136
```

Wektory i wartości własne

- Przykład dla macierzy diagonalnej

```
A <- matrix(c(2, 0, 0, 0, 9, 0, 0, 0, 4), nrow = 3, ncol = 3, byrow = FALSE)

     [,1] [,2] [,3]
[1,]    2    0    0
[2,]    0    9    0
[3,]    0    0    4

eigen(A)

$values
[1] 9 4 2

$vectors
     [,1] [,2] [,3]
[1,]    0    0    1
[2,]    1    0    0
[3,]    0    1    0
```

Wektory i wartości własne

- Dla komputera nie stanowi żadnego problemu, aby policzyć wartości i wektory własne dla naprawdę dużych macierzy

```
A <- matrix(data = rexp(200, rate = 10), nrow = 100, ncol = 100)
out <- eigen(A)
out$values
```

```
  [1]  1.055157e+01+0.000000e+00i  4.448576e-01+0.000000e+00i -1.726123e-15+0.000000e+00i  6.006378e-16+6.342243e-16i  6.006378e-16-6.342243e-16i
  [6] -7.628836e-16+0.000000e+00i  4.424291e-17+2.387708e-16i  4.424291e-17-2.387708e-16i -1.002150e-16+6.650082e-17i -1.002150e-16-6.650082e-17i
 [11]  7.710036e-17+9.191950e-17i  7.710036e-17-9.191950e-17i -1.654776e-17+9.704105e-17i -1.654776e-17-9.704105e-17i  1.558869e-18+9.822345e-17i
 [16]  1.558869e-18-9.822345e-17i -9.431600e-17+1.608155e-17i -9.431600e-17-1.608155e-17i  4.789848e-17+8.104054e-17i  4.789848e-17-8.104054e-17i
 [21] -3.599198e-17+8.598995e-17i -3.599198e-17-8.598995e-17i  8.279900e-17+3.920744e-17i  8.279900e-17-3.920744e-17i  7.777436e-17+0.000000e+00i
 [26]  3.269621e-17+6.868032e-17i  3.269621e-17-6.868032e-17i -2.106696e-18+7.342590e-17i -2.106696e-18-7.342590e-17i  6.503906e-17+3.244144e-17i
 [31]  6.503906e-17-3.244144e-17i -4.190616e-17+5.462562e-17i -4.190616e-17-5.462562e-17i  6.738986e-17+1.269936e-17i  6.738986e-17-1.269936e-17i
 [36] -6.446244e-17+1.448173e-17i -6.446244e-17-1.448173e-17i  2.174096e-17+6.072852e-17i  2.174096e-17-6.072852e-17i -4.887337e-17+2.862751e-17i
 [41] -4.887337e-17-2.862751e-17i -2.441913e-17+5.022024e-17i -2.441913e-17-5.022024e-17i -5.492375e-17+8.387885e-18i -5.492375e-17-8.387885e-18i
 [46]  4.985730e-17+2.218701e-17i  4.985730e-17-2.218701e-17i -2.840004e-17+3.842202e-17i -2.840004e-17-3.842202e-17i -4.576448e-17+0.000000e+00i
 [51] -1.332436e-18+4.476782e-17i -1.332436e-18-4.476782e-17i  4.458962e-17+0.000000e+00i -4.266956e-17+0.000000e+00i -3.805882e-17+1.917927e-17i
 [56] -3.805882e-17-1.917927e-17i  3.143985e-17+2.222165e-17i  3.143985e-17-2.222165e-17i -1.733315e-17+3.369038e-1
 [61]  1.900022e-17+3.084925e-17i  1.900022e-17-3.084925e-17i  3.222533e-17+1.417138e-17i  3.222533e-17-1.417138e-1
 [66]  3.327796e-17-7.555016e-18i  1.328380e-18+3.264118e-17i  1.328380e-18-3.264118e-17i -2.636349e-17+5.124596e-1
 [71]  1.320955e-17+2.321459e-17i  1.320955e-17-2.321459e-17i  2.353102e-18+2.453386e-17i  2.353102e-18-2.453386e-1
 [76]  2.322501e-17-7.606593e-18i  2.188632e-18+1.838174e-17i  2.188632e-18-1.838174e-17i -4.975539e-18+1.760066e-1
 [81] -1.392858e-17+8.097802e-18i -1.392858e-17-8.097802e-18i -1.518269e-17+4.307925e-19i -1.518269e-17-4.307925e-1
 [86] -9.342145e-18+8.771387e-18i -9.342145e-18-8.771387e-18i  8.107821e-18+9.687013e-18i  8.107821e-18-9.687013e-1
 [91] -2.390223e-18-1.226707e-17i  1.022977e-17+0.000000e+00i -9.869571e-18+0.000000e+00i -2.686358e-19+7.801289e-1
 [96]  4.951419e-18+4.928444e-18i  4.951419e-18-4.928444e-18i -4.699113e-18+3.882347e-18i -4.699113e-18-3.882347e-1
```

Pojawiają się **liczby zespolone**. Całkowicie pomijamy zagadnienie „skąd i dlaczego się wzięły”

```
out$vectors[,1]
```

```
  [1] -0.019574302+0i -0.047903253+0i -0.040164240+0i -0.010476718+0i -0.139732155+0i -0.129179146+0i -0.035950175+0i -0.021343645+0i -0.092567432+0i
 [10] -0.026164566+0i -0.090512654+0i -0.156691179+0i -0.056310561+0i -0.083123302+0i -0.117882746+0i -0.041986850+0i -0.148507236+0i -0.051503155+0i
 [19] -0.020856127+0i -0.208593508+0i -0.107937432+0i -0.176560740+0i -0.012699879+0i -0.089527037+0i -0.082063759+0i -0.165696430+0i -0.067602867+0i
 [28] -0.038040017+0i -0.052688290+0i -0.107612594+0i -0.077314455+0i -0.180880362+0i -0.032756238+0i -0.032932089+0i -0.019018543+0i -0.023402765+0i
 [37] -0.059711701+0i -0.135235335+0i -0.280135901+0i -0.029612182+0i -0.039245262+0i -0.009478095+0i -0.022302451+0i -0.069610034+0i -0.051183219+0i
 [46] -0.073460835+0i -0.199109330+0i -0.093050312+0i -0.205500331+0i -0.149258573+0i -0.045187050+0i -0.039805492+0i -0.051481799+0i -0.151181975+0i
 [55] -0.090230825+0i -0.067013616+0i -0.137712455+0i -0.044250003+0i -0.160583645+0i -0.064150996+0i -0.090467593+0i -0.064883484+0i -0.110637241+0i
 [64] -0.030106523+0i -0.020774288+0i -0.022917919+0i -0.272127474+0i -0.091486666+0i -0.042398003+0i -0.021410729+0i -0.061439938+0i -0.057524754+0i
 [73] -0.168609157+0i -0.171573571+0i -0.045260318+0i -0.197604393+0i -0.085821099+0i -0.045117644+0i -0.110638873+0i -0.057280225+0i -0.047996622+0i
 [82] -0.059089807+0i -0.020910860+0i -0.075514949+0i -0.092300054+0i -0.067438133+0i -0.110607489+0i -0.100771750+0i -0.086689431+0i -0.114793362+0i
 [91] -0.027427821+0i -0.047609975+0i -0.012723727+0i -0.046730978+0i -0.040723840+0i -0.085113467+0i -0.086701134+0i -0.106567264+0i -0.038025361+0i
[100] -0.086640220+0i
```

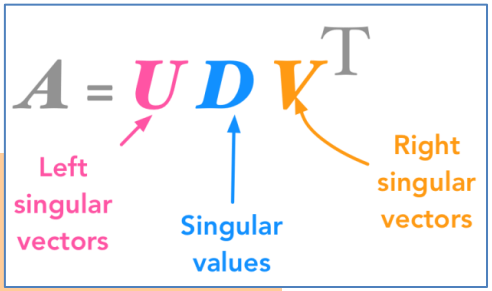
Ukryte indeksowanie semantyczne (Latent Semantic Indexing (LSI)

Latent Semantic Indexing (LSI)

- Główna idea: w danych tekstowych często jest pewna dodatkowa (i ukryta) struktura semantyczna, którą tracimy poprzez jedynie proste analizy typu „TDM, TFIDF, cos α”
  - semantyka bada relacje między znaczeniem podstawowym wyrazu a jego znaczeniem w konkretnej wypowiedzi
- Aby tą ukrytą wiedzę odkryć okazuje się, że warto najpierw „pozbyć się” różnych szczegółów w danych, które bardziej przeszkadzają niż pomagają
  - metoda: rzutowanie danych w większym wymiarze na dane w mniejszym wymiarze (redukcja wymiarowości)
  - jedną z bardziej klasycznych metod jest rzutowanie z wykorzystaniem przekształcenia SVD (*Singular Value Decomposition,* Rozkład według wartości osobliwych)
    - ✓ Wyznaczanie SVD to „wyższa szkoła jazdy” w dziedzinie algebry liniowej, zwłaszcza dla dużych wymiarów macierzy A

Singular Value Decomposition (SVD)

- Każdą macierz rzeczywistą (a więc niekoniecznie kwadratową, jak np. przy wartościach i wektorach własnych) **A** można zdekomponować w postaci iloczynu trzech specyficznych macierzy



$$\mathbf{A} = \mathbf{U} \cdot \mathbf{D} \cdot \mathbf{V}^T$$

gdzie

$\mathbf{U}, \mathbf{V}$ – macierze ortogonalne (czyli $\mathbf{U}^{-1} = \mathbf{U}^T, \mathbf{V}^{-1} = \mathbf{V}^T$)

$\mathbf{D}$ – macierz diagonalna, taka że $\mathbf{D} = diag(\sigma_1, \sigma_2, \ldots, \sigma_n)$

$\sigma_1 \geq \sigma_2 \geq, \ldots, \geq \sigma_n \geq 0$

gdzie σ_i – nieujemne wartości własne macierzy $\mathbf{A}$, zwyczajowo uporządkowane nierosnąco

Singular Value Decomposition (SVD)

- Przykłady, macierz A o wymiarze 5 x 3 oraz 3 x 5

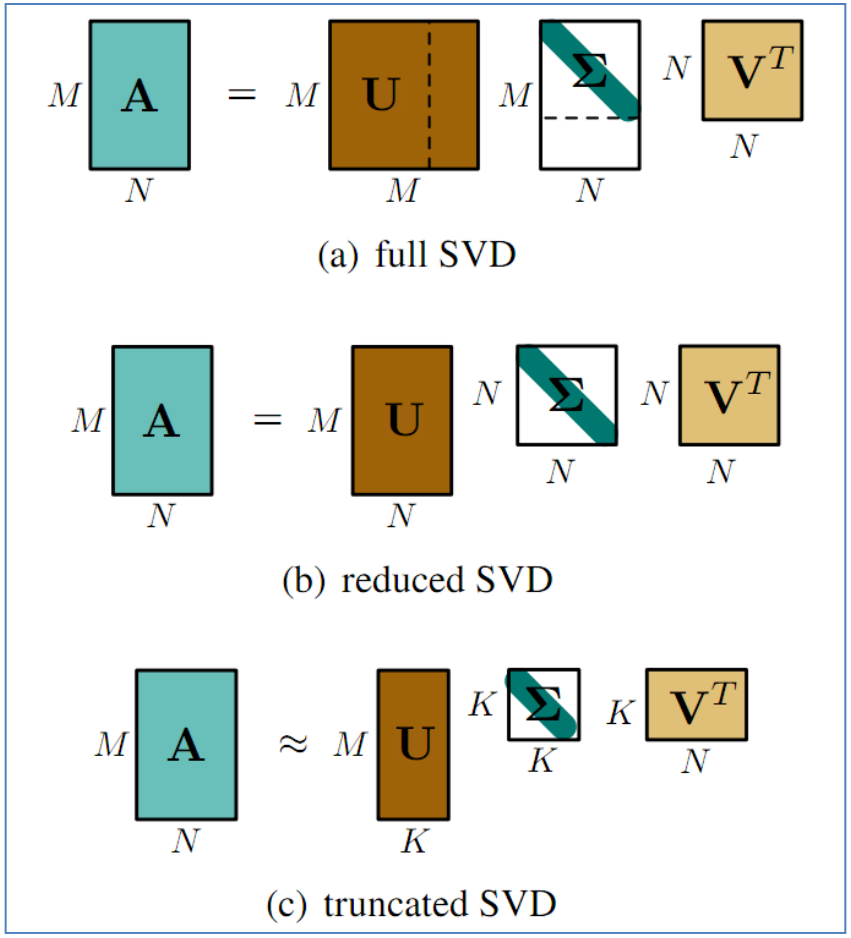
$$\mathbf{A}_{m\times n} = \mathbf{U}_{m\times m} \cdot \mathbf{D}_{m\times n} \cdot \mathbf{V}_{n\times n}^{T}$$

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix}_{5\times 3} = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \end{bmatrix}_{5\times 5} \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}_{5\times 3} \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix}_{3\times 3}$$

$$\begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \end{bmatrix}_{3\times 5} = \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix}_{3\times 3} \begin{bmatrix} \sigma_1 & 0 & 0 & 0 & 0 \\ 0 & \sigma_2 & 0 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 & 0 \end{bmatrix}_{3\times 5} \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \end{bmatrix}_{5\times 5}$$

Singular Value Decomposition (SVD)

- SVD, wersja pełna, zredukowana, obcięta - wizualizacja



Singular Value Decomposition (SVD)

- Przykład w R, macierz 2 x 3

```
A <- matrix(c(1, 2,3, 4, 5, 6), nrow = 2, ncol = 3
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6

svd <- svd(A)
```

```
U <- svd$u
        [,1]   [,2]
[1,] -0.386 -0.922
[2,] -0.922  0.386
```

```
V <- svd$v
        [,1]   [,2]
[1,] -0.429  0.806
[2,] -0.566  0.112
[3,] -0.704 -0.581
```

```
D <- diag(svd$d)
     [,1]  [,2]
[1,] 9.51 0.000
[2,] 0.00 0.773
```

```
U %*% D %*% t(V)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
```

Singular Value Decomposition (SVD)

- Przykład w R, macierz 3 x 2

```
A <- matrix(c(1, 2,3, 4, 5, 6), nrow = 3, ncol = 2)
     [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6

svd <- svd(A)
```

```
U <- svd$u
       [,1]   [,2]
[1,] -0.230  0.883
[2,] -0.525  0.241
[3,] -0.820 -0.402
```

```
V <- svd$v
       [,1]   [,2]
[1,] -0.620 -0.785
[2,] -0.785  0.620
```

```
D <- diag(svd$d)
     [,1]  [,2]
[1,] 9.53 0.000
[2,] 0.00 0.514
```

```
U %*% D %*% t(V)
     [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
```

Singular Value Decomposition (SVD)

- Przykład w R, macierz 3 x 5

```
A <- matrix(
  c(
    1, 0, 1, 0, 0,
    0, 1, 0, 1, 1,
    1, 1, 0, 0, 1),
  nrow=3, ncol=5, byrow=T)

> A
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    0    1    0    0
[2,]    0    1    0    1    1
[3,]    1    1    0    0    1
```

```
svd <- svd(A)
```

```
U <- svd$u
        [,1]   [,2]   [,3]
[1,] -0.226  0.846 -0.483
[2,] -0.661 -0.497 -0.562
[3,] -0.715  0.192  0.672

D <- diag(svd$d)
     [,1] [,2] [,3]
[1,] 2.27 0.00 0.00
[2,] 0.00 1.49 0.00
[3,] 0.00 0.00 0.78

V <- t(svd$v)
        [,1]   [,2]    [,3]   [,4]   [,5]
[1,] -0.414 -0.606 -0.0995 -0.291 -0.606
[2,]  0.696 -0.204  0.5669 -0.333 -0.204
[3,]  0.242  0.141 -0.6189 -0.720  0.141
```

```
U %*% D %*% V
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    0    1    0    0
[2,]    0    1    0    1    1
[3,]    1    1    0    0    1
```

Singular Value Decomposition (SVD)

- Przykład w R, macierz 3 x 5
  - często pojawiają się minimalne niedokładności, wynikające z różnych zaokrągleń podczas obliczeń. Nie mają one jednak żadnego znaczenia praktycznego. Taki wynik zwraca R:

```
U %*% D %*% V
[,1]     [,2]      [,3]      [,4]     [,5]
[1,] 1.00e+00 1.53e-16  1.00e+00  2.78e-16 9.02e-17
[2,] 2.78e-17 1.00e+00 -3.33e-16  1.00e+00 1.00e+00
[3,] 1.00e+00 1.00e+00 -1.11e-16 -2.78e-16 1.00e+00
```

  - ekstremalnie małe wartości możemy uznać za zerowe

```
U %*% D %*% V
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    0    1    0    0
[2,]    0    1    0    1    1
[3,]    1    1    0    0    1
```

Singular Value Decomposition (SVD)

- Przekształcenie SVD to tak naprawdę trzy przekształcenia
  - rotacja
  - skalowanie
  - druga rotacja

$$\mathbf{A} = \mathbf{U} \cdot \mathbf{D} \cdot \mathbf{V}^T$$

- Przykład
  - https://github.com/hadrienj/deepLearningBook-Notes/blob/master/2.8%20Singular%20Value%20Decomposition/2.8%20Singular%20Value%20Decomposition.ipynb
  - przekształcimy dwa wektory jednostkowe przez macierz

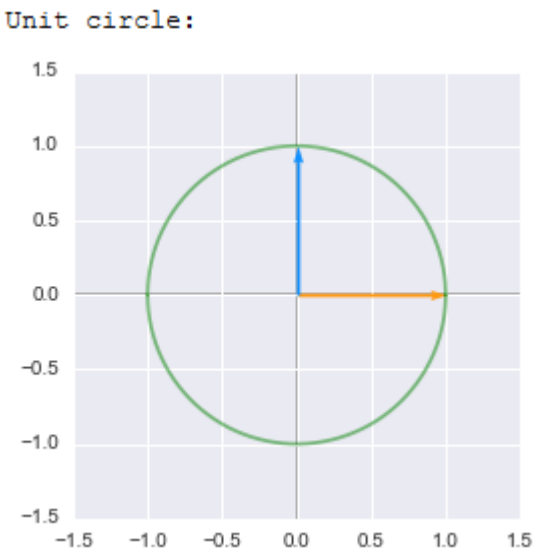
$$\mathbf{A} = \begin{bmatrix} 3 & 7 \\ 5 & 2 \end{bmatrix}, \quad \mathbf{b}_1 = [0,1], \quad \mathbf{b}_2 = [1,0]$$

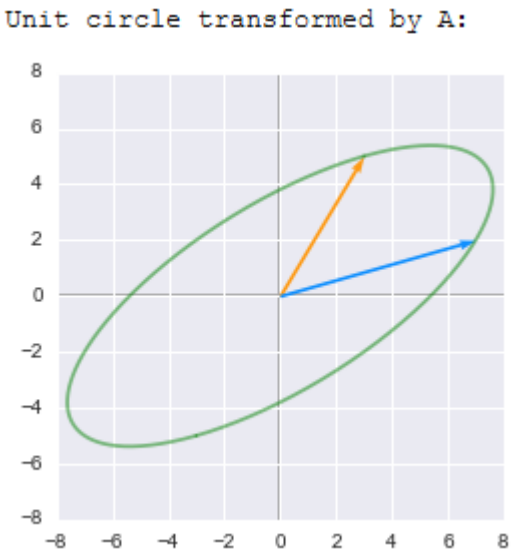
```
svd(A)
$d
[1] 8.713 3.328
$u
          [,1]    [,2]
[1,] -0.8507 -0.5257
[2,] -0.5257  0.8507
$v
          [,1]    [,2]
[1,] -0.5946  0.8041
[2,] -0.8041 -0.5946
```

Singular Value Decomposition (SVD)

- Wynik przekształcenia wektorów $\mathbf{b}_1$, $\mathbf{b}_2$ przez macierz $\mathbf{A}$

$$\mathbf{A} = \begin{bmatrix} 3 & 7 \\ 5 & 2 \end{bmatrix}, \quad \mathbf{b}_1 = [0,1], \quad \mathbf{b}_2 = [1,0] \qquad \mathbf{A} \cdot \mathbf{b}_1 = \begin{bmatrix} 7 \\ 2 \end{bmatrix}, \quad \mathbf{A} \cdot \mathbf{b}_2 = \begin{bmatrix} 3 \\ 5 \end{bmatrix}$$



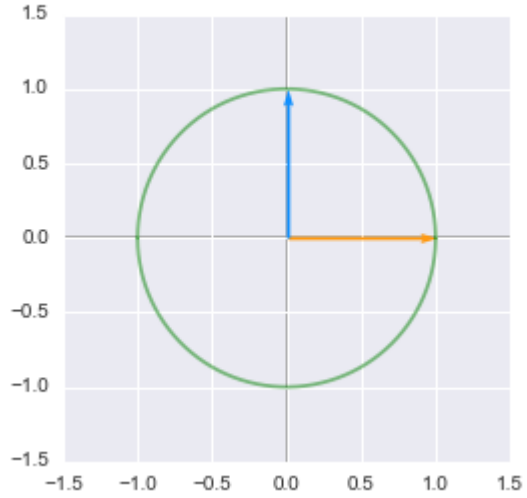


Singular Value Decomposition (SVD)

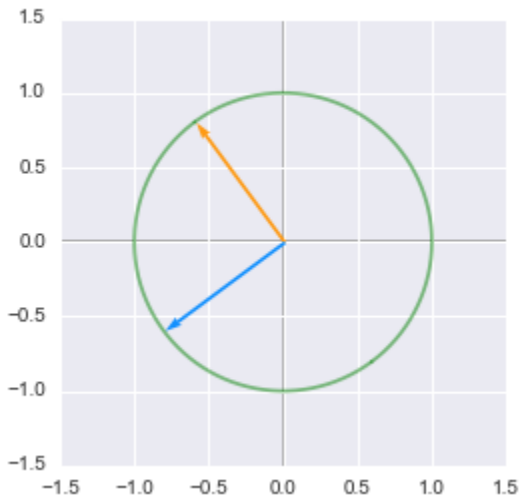
- Pierwsza rotacja

$$\mathbf{V}\cdot\mathbf{b}_1=\begin{bmatrix}-0{,}8041\\-0{,}5946\end{bmatrix},\quad \mathbf{V}\cdot\mathbf{b}_2=\begin{bmatrix}-0{,}5946\\0{,}8041\end{bmatrix}$$

Unit circle:



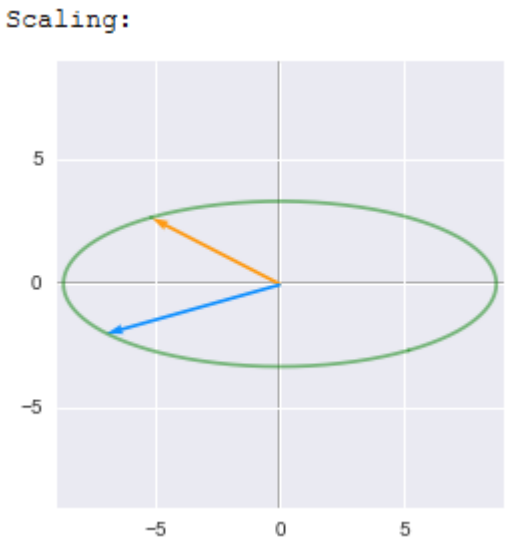
First rotation:



Singular Value Decomposition (SVD)

- Skalowanie

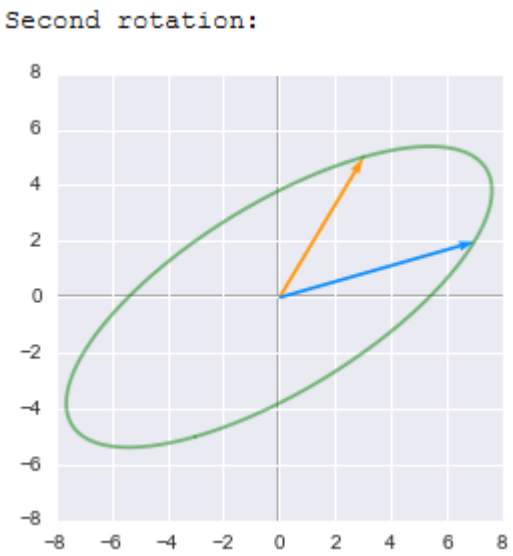
$$\mathbf{D}\cdot\mathbf{V}\cdot\mathbf{b}_1=\begin{bmatrix}-7{,}006\\-1{,}979\end{bmatrix},\quad \mathbf{D}\cdot\mathbf{V}\cdot\mathbf{b}_2=\begin{bmatrix}-5{,}181\\2{,}676\end{bmatrix}$$



Singular Value Decomposition (SVD)

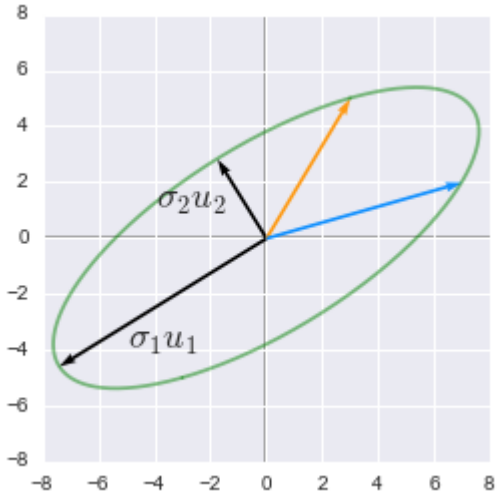
- Druga rotacja

$$\mathbf{U}\cdot\mathbf{D}\cdot\mathbf{V}\cdot\mathbf{b}_1=\begin{bmatrix}7\\2\end{bmatrix},\quad \mathbf{U}\cdot\mathbf{D}\cdot\mathbf{V}\cdot\mathbf{b}_2=\begin{bmatrix}3\\5\end{bmatrix}$$



Singular Value Decomposition (SVD)

- Interpretacja wartości singularnych
  - wartości singularne są uporządkowane malejąco. Odpowiadają one nowemu zestawowi cech (które są liniową kombinacją cech oryginalnych), przy czym pierwsza cecha wyjaśnia większość wariancji
  - dla naszego przykładu możemy w następujący sposób to zwizualizować



$$\sigma_1 \cdot u_1 = [-7{,}412 \;\; -4{,}581]$$
$$\sigma_2 \cdot u_2 = [-1{,}75 \;\; 2{,}831]$$

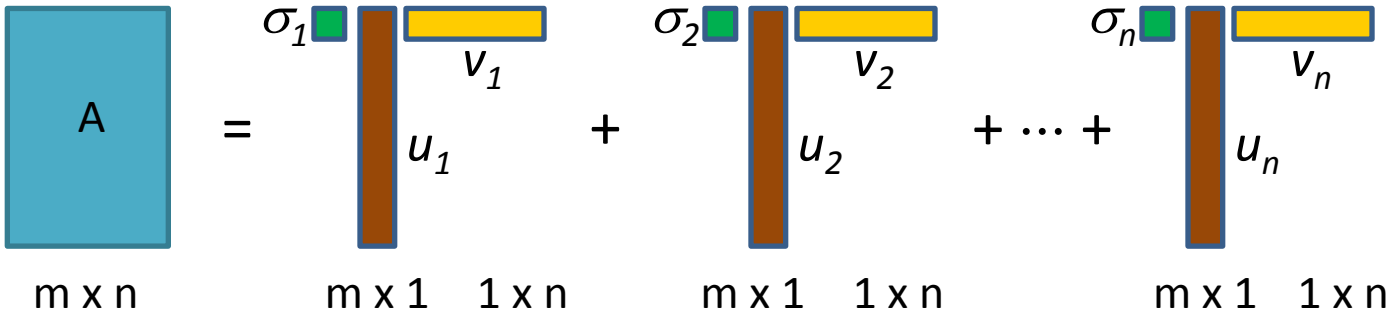
Singular Value Decomposition (SVD)

- Inny zapis (wynika on wprost z algebry macierzy)

$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T =$$

$$[\mathbf{u}_1\mathbf{u}_2\ldots\mathbf{u}_n]\begin{bmatrix}\sigma_1 & 0 & 0 & 0\\ 0 & \sigma_2 & 0 & 0\\ 0 & 0 & \ddots & 0\\ 0 & 0 & 0 & \sigma_n\end{bmatrix}\begin{bmatrix}\mathbf{v}_1^T\\ \mathbf{v}_2^T\\ \vdots\\ \mathbf{v}_n^T\end{bmatrix} = [\mathbf{u}_1\mathbf{u}_2\ldots\mathbf{u}_n]\begin{bmatrix}\sigma_1\mathbf{v}_1^T\\ \sigma_2\mathbf{v}_2^T\\ \vdots\\ \sigma_n\mathbf{v}_n^T\end{bmatrix} =$$

$$\sum_{i=1}^{n}\sigma_i\mathbf{u}_i\mathbf{v}_i^{\mathbf{T}}$$



Związek SVD z eigenvalue problem

- Macierze **U**, **D, V** mogą być wyznaczone wykorzystując następujące zależności
  - kolumny **U** to wektory własne macierzy $\mathbf{AA}^T$
  - kolumny **V** to wektory własne macierzy $\mathbf{A}^T\mathbf{A}$
  - niezerowe wartości singularne w **D** to pierwiastki kwadratowe niezerowych wartości własnych $\mathbf{AA}^T$ lub $\mathbf{A}^T\mathbf{A}$ (dają te same wartości)

$$\mathbf{A} = \mathbf{UDV}^T$$

$$\mathbf{Av} = \lambda\mathbf{v}$$

Tutaj mamy pomnożenie przez -1. Wynika to ze szczegółów technicznych obliczania SVD. Nie zmienia to jednak kierunku wektora oraz jego długości

```
A
     [,1] [,2]
[1,]    7    2
[2,]    3    4
[3,]    5    3
svd(A)
$d
[1] 10.251  2.628
$u
          [,1]    [,2]
[1,] -0.6937  0.5934
[2,] -0.4427 -0.7983
[3,] -0.5682 -0.1025
$v
          [,1]    [,2]
[1,] -0.8803  0.4743
[2,] -0.4743 -0.8803
```

```
# kolumny U to wektory własne macierzy AA^T
eigen(A %*% t(A))$vectors
       [,1]    [,2]    [,3]
[1,] 0.6937  0.5934 -0.4082
[2,] 0.4427 -0.7983 -0.4082
[3,] 0.5682 -0.1025  0.8165
# kolumny V to wektory własne macierzy A^TA
eigen(t(A) %*% A)$vectors
        [,1]    [,2]
[1,] -0.8803  0.4743
[2,] -0.4743 -0.8803
# wartości singularne w D to pierwiastki
# kwadratowe wartości własnych AA^T lub A^TA
sqrt(svd(A %*% t(A))$d)
[1] 1.025e+01 2.628e+00 6.892e-08

sqrt(svd(t(A) %*% A)$d)
[1] 10.251  2.628
```

Singular Value Decomposition (SVD)

- Rank-k approximation
  - zerujemy wszystkie najmniejsze σ_i za wyjątkiem k największych

$$\mathbf{A}_{m\times n} = \mathbf{U}_{m\times m} \cdot \mathbf{D}_{m\times n} \cdot \mathbf{V}_{n\times n}^{T}$$

$$\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \\ * & * & * \end{bmatrix}_{5\times 3} = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \end{bmatrix}_{5\times 5} \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}_{5\times 3} \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix}_{3\times 3}$$

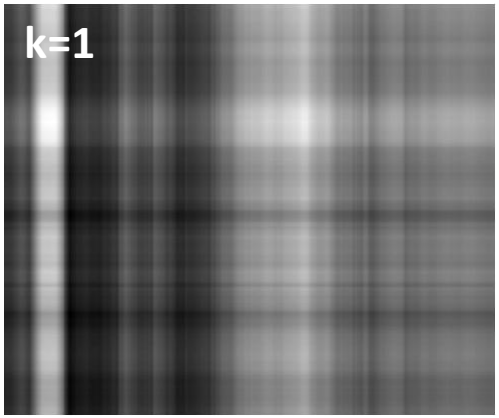
Ten element został wyzerowany. Mamy więc w tym przypadku rank-2 approximation

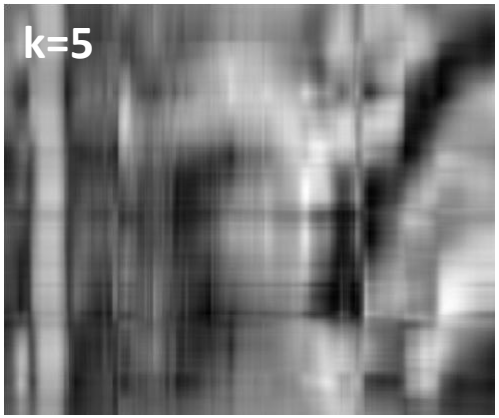
$$\mathbf{A} = \sum_{i=1}^{n} \sigma_i u_i v_i^{T} \approx \sum_{i=1}^{k} \sigma_i u_i v_i^{T} = \mathbf{A_k}$$

Zwrócić uwagę na indeks sumowana. Raz jest *n* a raz *k*. Przy czym n > k

Singular Value Decomposition (SVD)

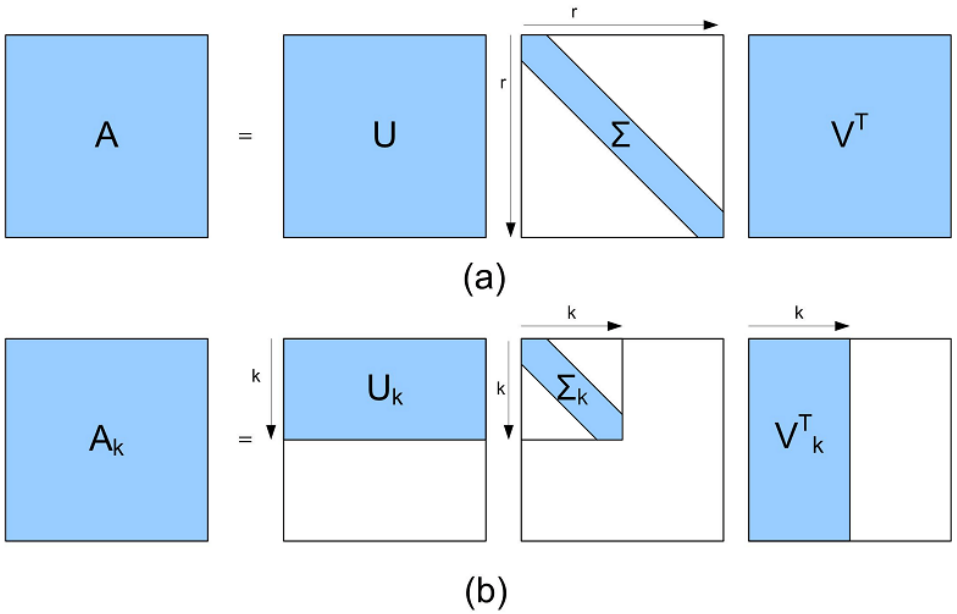
- Przykład, jak SVD zadziała w odniesieniu do grafiki





Singular Value Decomposition (SVD)

- Rank-k approximation
  - po wyzerowaniu najmniejszych wartości własnych powstała macierz $\mathbf{A}_k$ nie jest już tym samym, co macierz **A**. Jest to jej przybliżenie
  - to co utraciliśmy możemy uznać za rodzaj szumu informacyjnego. Pozostało nam tylko to, co najważniejsze
  - co przez to zyskujemy? Ano zobaczymy za chwilę na przykładach



Singular Value Decomposition (SVD)

- Przykład rank-k approximation, macierz 3 x 5

```
A
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    0    1    0    0
[2,]    0    1    0    1    1
[3,]    1    1    0    0    1
```

```
U <- svd$u
        [,1]    [,2]   [,3]
[1,] -0.226  0.846 -0.483
[2,] -0.661 -0.497 -0.562
[3,] -0.715  0.192  0.672

D <- diag(svd$d)
     [,1] [,2] [,3]
[1,] 2.27 0.00 0.00
[2,] 0.00 1.49 0.00
[3,] 0.00 0.00 0.00

V <- t(svd$v)
        [,1]    [,2]     [,3]    [,4]    [,5]
[1,] -0.414 -0.606 -0.0995 -0.291 -0.606
[2,]  0.696 -0.204  0.5669 -0.333 -0.204
[3,]  0.242  0.141 -0.6189 -0.720  0.141
```

Ta wartość własna została ręcznie przez nas wyzerowana

To już jest inna macierz niż A. Pewne podobieństwo jednak pozostało. Tam, gdzie były oryginalnie jedynki są wartość dość bliskie 1, tam gdzie były zera wartości są bliższe 0.

```
A.k <- U %*% D %*% V
       [,1]   [,2]   [,3]    [,4]   [,5]
[1,] 1.091 0.0531  0.767 -0.271 0.0531
[2,] 0.106 1.0618 -0.271  0.684 1.0618
[3,] 0.873 0.9261  0.324  0.377 0.9261
```

Singular Value Decomposition (SVD)

- rank-k approximation, miara utraty informacji

$$\text{utrata\_inf} = \frac{\|\mathbf{A} - \mathbf{A}_k\|_F}{\|\mathbf{A}\|_F}$$

można udowodnić, że

$$\|\mathbf{A} - \mathbf{A}_k\|_F = \left(\sum_{i=k+1}^{p} \sigma_i^2 \right)^{1/2} = \sqrt{\sigma_{k+1}^2 + \ldots + \sigma_{r_A}^2}$$

Czyli błąd aproksymacji $A \to A_k$ jest określony przez obcięte (wyzerowane) wartości własne

$$\sigma_{k+1}, \sigma_{k+2}, \ldots, \sigma_{r_A}$$

Singular Value Decomposition (SVD)

- rank-k approximation, miara utraty informacji, wynik dla podanego powyżej przykładu

```
A
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    0    1    0    0
[2,]    0    1    0    1    1
[3,]    1    1    0    0    1
```

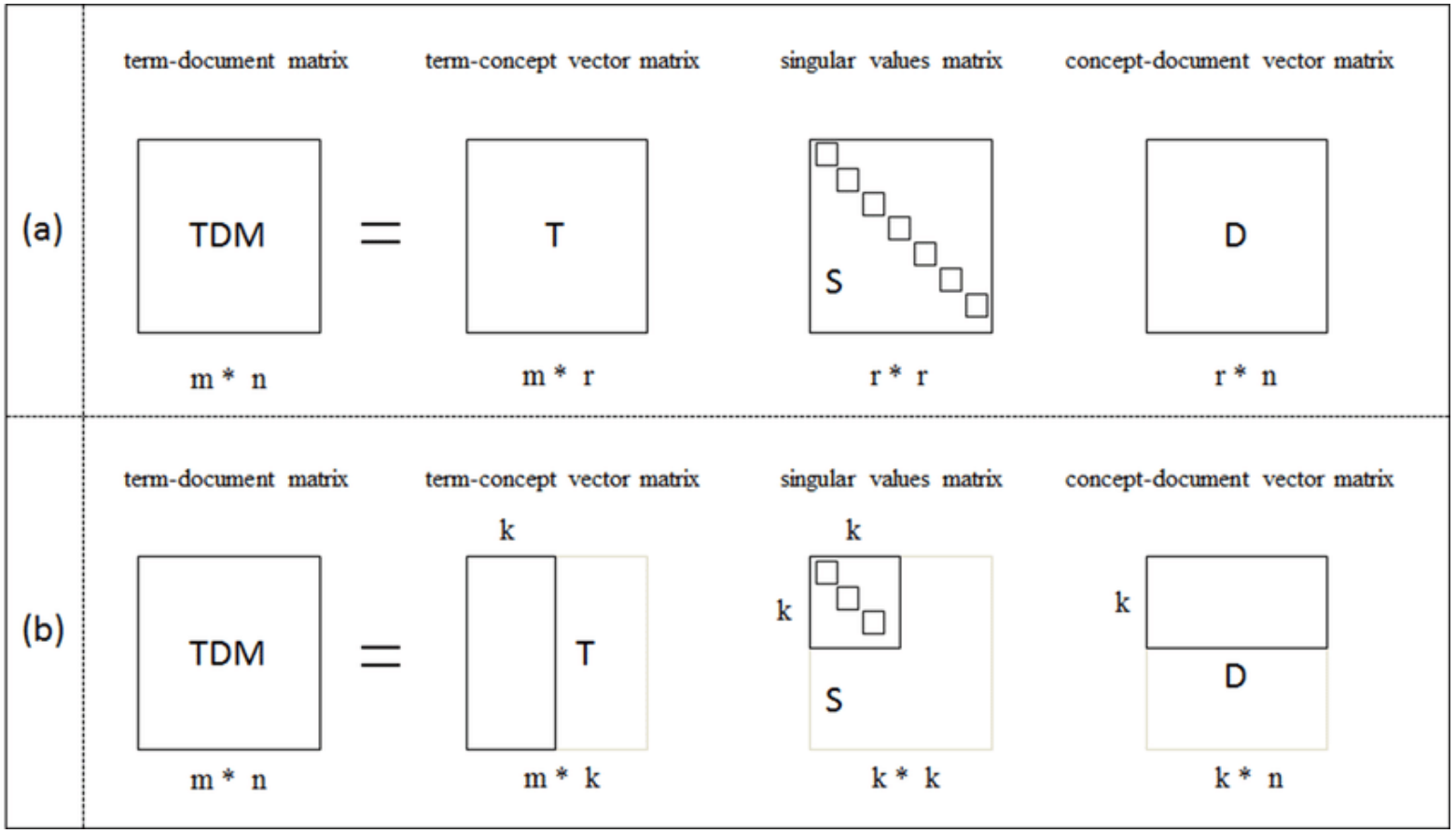
```
A.k <- U %*% D %*% V
      [,1]   [,2]   [,3]   [,4]   [,5]
[1,] 1.091 0.0531  0.767 -0.271 0.0531
[2,] 0.106 1.0618 -0.271  0.684 1.0618
[3,] 0.873 0.9261  0.324  0.377 0.9261
```

```
n1 <- norm(A - Ak, type = "F")
n2 <- norm(A, type = "F")
n1 / n2
0.276
```

Czyli utraciliśmy około 28% pierwotnej informacji

Singular Value Decomposition (SVD)

- SVD w kontekście macierzy Term-Document Matrix (TDM)



Rank-k approximation – przykład 1

```
A (macierz TDM)
        d1   d2   d3   d4   d5   d6
ship     1    0    1    0    0    0
boat     0    1    0    0    0    0
ocean    1    1    0    0    0    0
voyage   1    0    0    1    1    0
trip     0    0    0    1    0    1
```

```
U <- svd$u
       [,1]   [,2]   [,3]       [,4]    [,5]
[1,] 0.440 -0.296 -0.569  5.77e-01 -0.2464
[2,] 0.129 -0.331  0.587 -3.33e-16 -0.7272
[3,] 0.476 -0.511  0.368 -3.33e-16  0.6144
[4,] 0.703  0.351 -0.155 -5.77e-01 -0.1598
[5,] 0.263  0.647  0.415  5.77e-01  0.0866
```

```
> D <- diag(svd$d)
     [,1] [,2] [,3] [,4]  [,5]
[1,] 2.16 0.00 0.00    0 0.000
[2,] 0.00 1.59 0.00    0 0.000
[3,] 0.00 0.00 1.28    0 0.000
[4,] 0.00 0.00 0.00    1 0.000
[5,] 0.00 0.00 0.00    0 0.394
```

```
V <- svd$v
       [,1]   [,2]   [,3]       [,4]   [,5]
[1,] 0.749 -0.286 -0.280 -2.22e-16  0.528
[2,] 0.280 -0.528  0.749 -3.33e-16 -0.286
[3,] 0.204 -0.186 -0.447  5.77e-01 -0.626
[4,] 0.447  0.626  0.204  1.94e-16 -0.186
[5,] 0.325  0.220 -0.121 -5.77e-01 -0.406
[6,] 0.121  0.406  0.325  5.77e-01  0.220
```

Rank-k approximation – przykład 1

```
# rank-k, zerujemy najmniejsze wartości w D
# (zostawiamy 2. największe)
k <- 2
D.k <- D
D.k[(k+1):5, (k+1):5] = 0
     [,1] [,2] [,3] [,4] [,5]
[1,] 2.16 0.00    0    0    0
[2,] 0.00 1.59    0    0    0
[3,] 0.00 0.00    0    0    0
[4,] 0.00 0.00    0    0    0
[5,] 0.00 0.00    0    0    0
```

$\mathbf{D}_k$

```
A.k <- U %*% D.k %*% t(V)
      [,1]   [,2]    [,3]    [,4]    [,5]    [,6]
[1,] 0.848  0.516  0.2816  0.1299  0.2057 -0.0759
[2,] 0.361  0.358  0.1551 -0.2057 -0.0253 -0.1804
[3,] 1.003  0.718  0.3608 -0.0505  0.1551 -0.2057
[4,] 0.978  0.130  0.2057  1.0285  0.6171  0.4114
[5,] 0.130 -0.386 -0.0759  0.8987  0.4114  0.4873
```

$\mathbf{A}_k$

```
norm(A - A.k, type="F") / norm(A, type="F")
[1] 0.527
```

Rank-k approximation – przykład 1

```
# Przekształcamy oryginalne termy na przestrzeń konceptów
#(ang. concept space) w zredukowanej przestrzeni rank-k.
terms.k <- U %*% D.k
      [,1]   [,2] [,3] [,4] [,5]
[1,] 0.952 -0.472    0    0    0
[2,] 0.280 -0.528    0    0    0
[3,] 1.028 -0.815    0    0    0
[4,] 1.520  0.559    0    0    0
[5,] 0.568  1.031    0    0    0
```

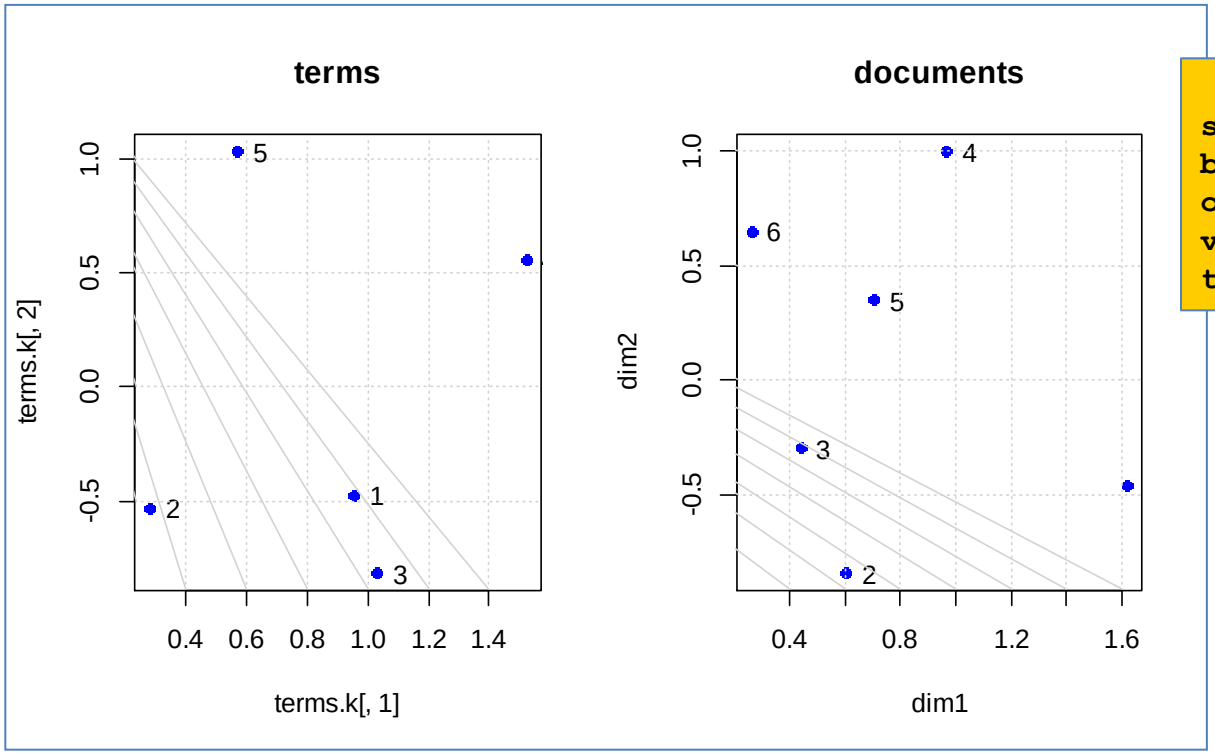
$$terms_k = \mathbf{U}\mathbf{D}_k$$

```
> # Przekształcamy oryginalne dokumenty na przestrzeń konceptów
> # (ang. concept space) w zredukowanej przestrzeni rank-k.
> docs.k <- D.k %*% t(V)
       [,1]   [,2]   [,3]  [,4]  [,5]  [,6]
[1,]  1.619  0.605  0.440 0.966 0.703 0.263
[2,] -0.457 -0.843 -0.296 0.997 0.351 0.647
[3,]  0.000  0.000  0.000 0.000 0.000 0.000
[4,]  0.000  0.000  0.000 0.000 0.000 0.000
[5,]  0.000  0.000  0.000 0.000 0.000 0.000
```

$$docs_k = \mathbf{D}_k \mathbf{V}^T$$

Rank-k approximation – przykład 1

- Wizualizacja „przestrzeni konceptów termów” oraz „przestrzeni konceptów dokumentów”. Po redukcji wymiaru do 2D łatwiej jest wzrokowo dostrzec jakieś zależności
  - podobieństwa / niepodobieństwa / ukryte zależności / powiązania



	d1	d2	d3	d4	d5	d6
ship	1	0	1	0	0	0
boat	0	1	0	0	0	0
ocean	1	1	0	0	0	0
voyage	1	0	0	1	1	0
trip	0	0	0	1	0	1

Rank-k Approximation
przykład 2

```
doc
[1] "The Google matrix P is a model of the Internet"
[2] "P ij is nonzero if there is a link from Web page j to i"
[3] "The Google matrix rank is used to rank all Web pages"
[4] "The ranking is done by solving a matrix eigenvalue problem"
[5] "England dropped out of the top 10 in the FIFA ranking"
```

```
doc
[1] "Google matrix Internet"
[2] "link Web page"
[3] "Google matrix rank Web pages"
[4] "ranking matrix eigenvalue"
[5] "England FIFA ranking"
```

```
query
[1] "ranking web pages"
```

```
# dla oryginalnego TDM
cos.alpha
[1] 0.0000 0.6667 0.7746 0.3333 0.3333

# dla przestrzeni zredukowanej do k=2
cos.alpha.k
[1] 0.7857 0.8332 0.9670 0.4873 0.1819
```

```
tdm
           Docs
Terms       1 2 3 4 5
  eigenvalu 0 0 0 1 0
  england   0 0 0 0 1
  fifa      0 0 0 0 1
  googl     1 0 1 0 0
  internet  1 0 0 0 0
  link      0 1 0 0 0
  matrix    1 0 1 1 0
  page      0 1 1 0 0
  rank      0 0 1 1 1
  web       0 1 1 0 0
```

```
tdm.query

           [,1]
eigenvalu     0
england       0
fifa          0
googl         0
internet      0
link          0
matrix        0
page          1
rank          1
web           1
```

```
n1 <- norm(out$tdm - out$A.k, type="F")
[1] 2.3
n2 <- norm(out$tdm, type="F")
[1] 4.12
n1 / n2
[1] 0.559
```

Przykład zaczerpnięto z książki: Lars Elden, Matrix Methods in Data Mining and Pattern Recognition, przykład 11.8, strona 138

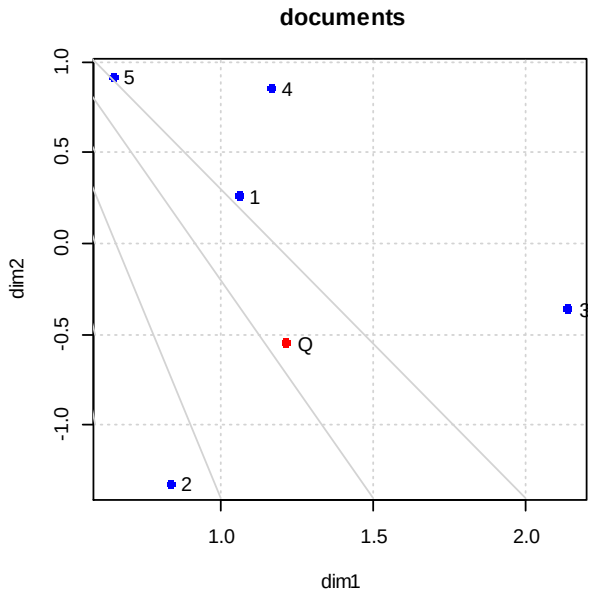
1. Patrząc na *doc* i *query* widać, że dokumenty 1-4 można uznać za w jakimś stopniu zgodne z query, natomiast dokument 5, tak „po ludzku”, nie za bardzo jest zgodny.

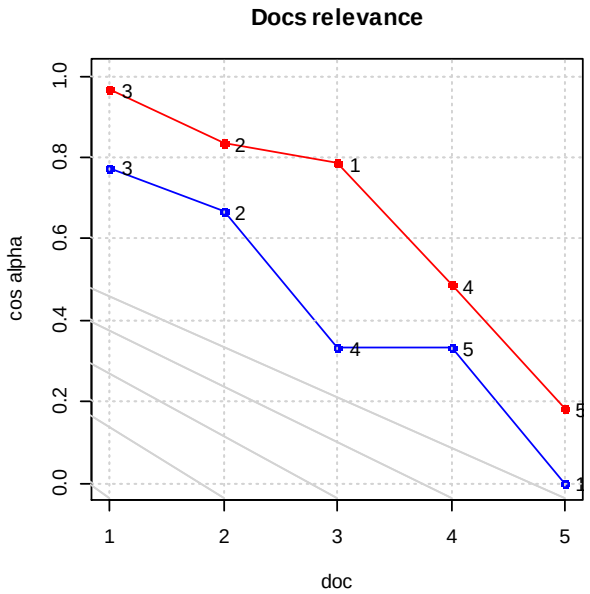
2. *cos.alpha* pokazuje, że dokumenty 4 i 5 mają identyczny stopień zgodności („po ludzku” nie wydaje się to poprawne).

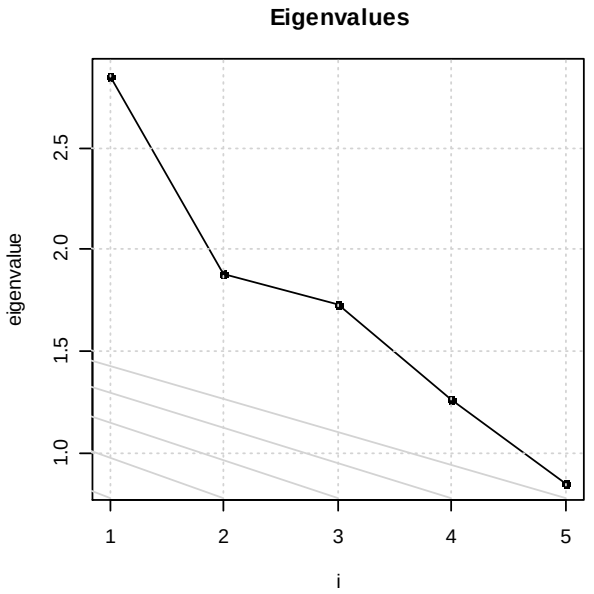
3. Dokument 1 ma zerową zgodność z *query* i „po ludzku” nie wydaje się to poprawne.

4. Po wykonaniu SVD i redukcji do rank-2 w wyniku otrzymujemy, że dokument 1 ma teraz całkiem wysoką zgodność z *query*. Dodatkowo współczynnik zgodności dla dokumentów 2-4 został wzmocniony a dla dokumentu 5 został lekko osłabiony.

5. Wniosek: w tym przykładzie LSI poprawiona została zdolność systemu do wyszukiwania dokumentów zgodnych z *query* (mimo, że literalnie np. dokument 1 nie ma żadnych







doc i query jak poprzednio, ale TDM według przekształcenia **TFIDF**

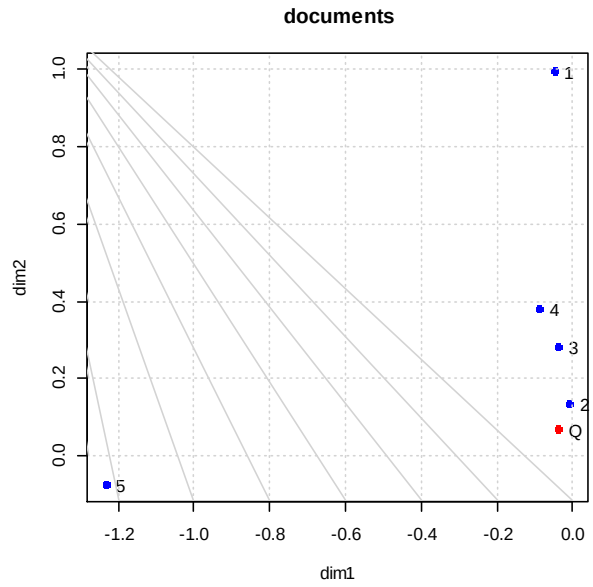
```
tdm
             Docs
Terms           1     2     3     4     5
  eigenvalu 0.000 0.000 0.000 0.862 0.000
  england   0.000 0.000 0.000 0.000 0.862
  fifa      0.000 0.000 0.000 0.000 0.862
  googl     0.528 0.000 0.317 0.000 0.000
  internet  0.862 0.000 0.000 0.000 0.000
  link      0.000 0.862 0.000 0.000 0.000
  matrix    0.333 0.000 0.200 0.333 0.000
  page      0.000 0.333 0.200 0.000 0.000
  rank      0.000 0.000 0.117 0.195 0.195
  web       0.000 0.333 0.200 0.000 0.000
```

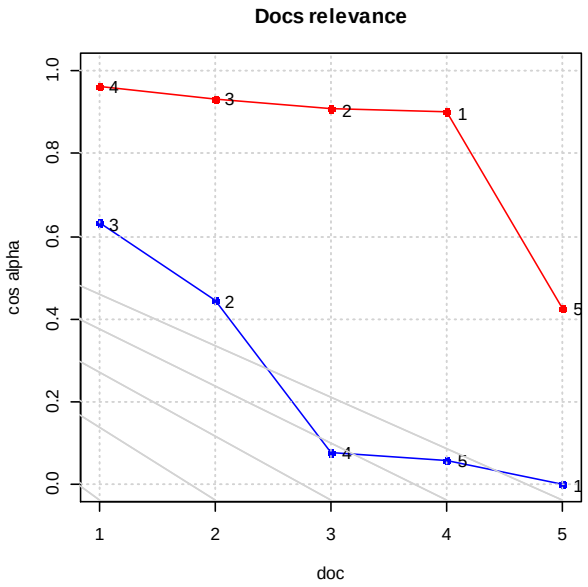
```
tdm.query

            [,1]
eigenvalu 0.000
england   0.000
fifa      0.000
googl     0.000
internet  0.000
link      0.000
matrix    0.000
page      0.333
rank      0.195
web       0.333
```

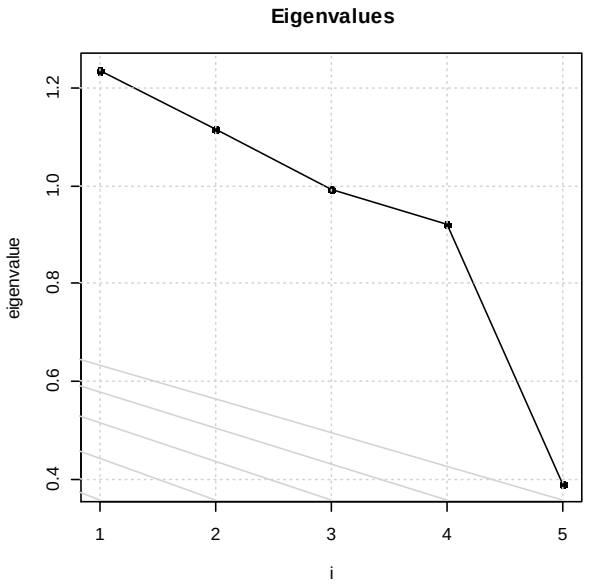
```
out$cos.alpha
[1] 0.0000 0.4435 0.6325 0.0789 0.0604
out$cos.alpha.k
[1] 0.898 0.906 0.932 0.962 0.427
```

```
n1 <- norm(out$tdm - out$A.k, type="F")
[1] 1.41
n2 <- norm(out$tdm, type="F")
[1] 2.18
n1 / n2
[1] 0.646
```

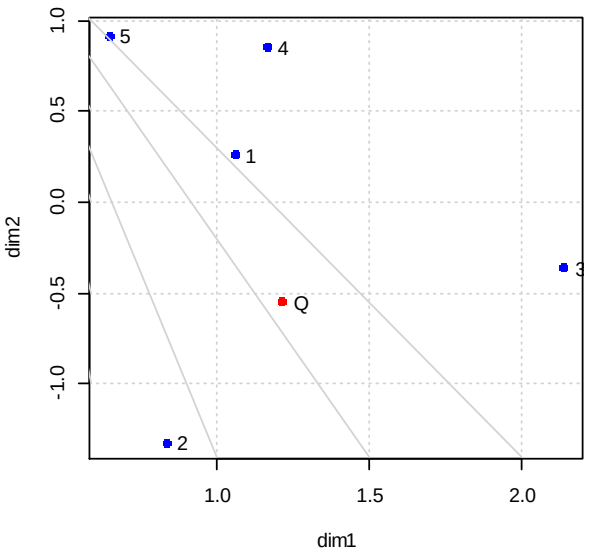
TFIDF

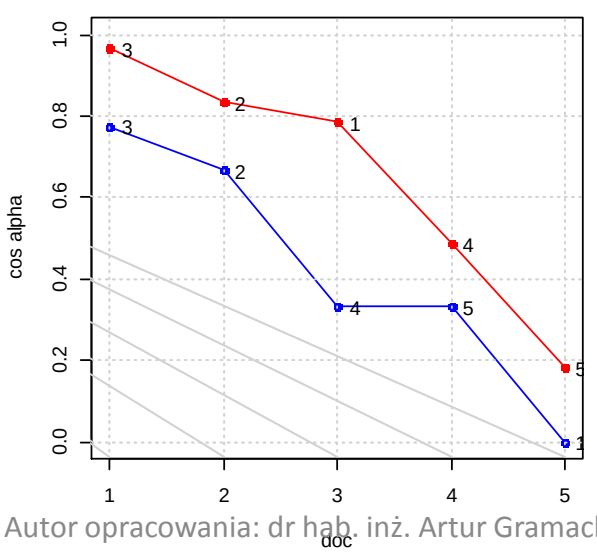


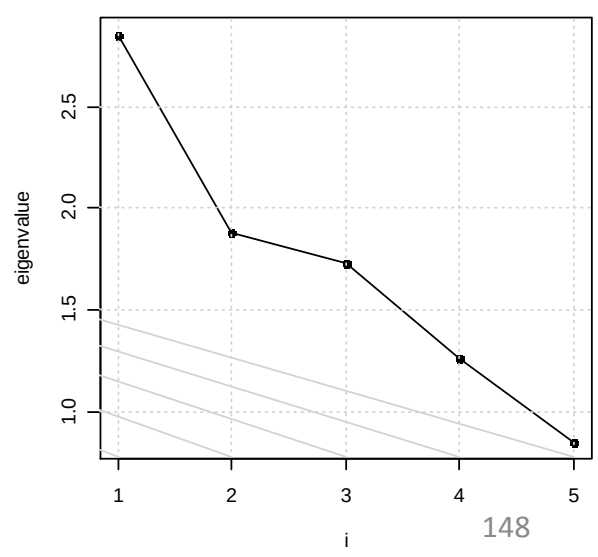




BINARY







Rank-k Approximation
przykład 3

75 tytułów artykułów z 2 różnych dziedzin (Statistics & Control)

```
1  [S] Anthology of Statistics in Sports
2  [S] A First Course in Order Statistics
3  [S] Mathematica Laboratories for Mathematical Statistics: Emphasizing Simulation and Computer Intensive Methods
4  [S] Some Limit Theorems in Statistics
5  [S] Engineering Reliability
6  [S] Mathematical Theory of Reliability
7  [S] Russian-English / English-Russian Dictionary on Probability, Statistics, and Combinatorics
8  [S] Time Series: Data Analysis and Theory
9  [S] Design and Analysis of Gauge R&R Studies: Making Decisions with Confidence Intervals in Random  ANOVA Models
10 [S] Sequential Analysis and Optimal Design
11 [S] Multivariate Splines
12 [S] Quantile Processes with Statistical Applications
13 [S] Statistical Case Studies for Industrial Process Improvement
14 [S] Selecting and Ordering Populations: A New Statistical Methodology
15 [S] Distribution Theory for Tests Based on the Sample Distribution Function
16 [S] The Jackknife, the Bootstrap, and Other Resampling Plans
17 [S] Multiple Decision Procedures: Theory and Methodology of Selecting and Ranking Populations
18 [S] Shanti S. Gupta and S. Panchapakesan
19 [S] Basic Concepts of Probability and Statistics, Second Edition
20 [S] Robust Statistical Procedures, Second Edition
21 [S] The Structural Representation of Proximity Matrices with MATLAB
22 [S] Statistical Design and Analysis of Experiments
23 [S] Pitman's Measure of Closeness
24 [S] Bayesian Nonparametrics via Neural Networks
25 [S] Bayesian Statistics, A Review
26 [S] Multivariate Statistical Process Control with Industrial Applications
27 [S] Eliciting and Analyzing Expert Judgment: A Practical Guide
28 [S] The Analysis of Means: A Graphical Method for Comparing Means, Rates and Proportions
29 [S] Recurrent Events Data Analysis for Product Repairs, Disease Recurrence, and Other Applications
30 [S] Applied Adaptive Statistical Methods: Tests of Significance and Confidence Intervals
31 [S] Statistical Case Studies: A Collaboration Between Academe and Industry
32 [S] Optimal Design of Experiments
33 [S] Theory and Applications of Sequential Nonparametrics
34 [S] Empirical Processes with Applications to Statistics
35 [S] A Primer for Sampling Solids, Liquids and Gases: Based on the Seven Sampling Errors of Pierre Gy
36 [S] Experimental Design for Formulation
37 [S] Nonparametric Function Estimation, Modeling, and Simulation
38 [S] Large Deviations and Applications
39 [S] Spline Models for Observational Data
```

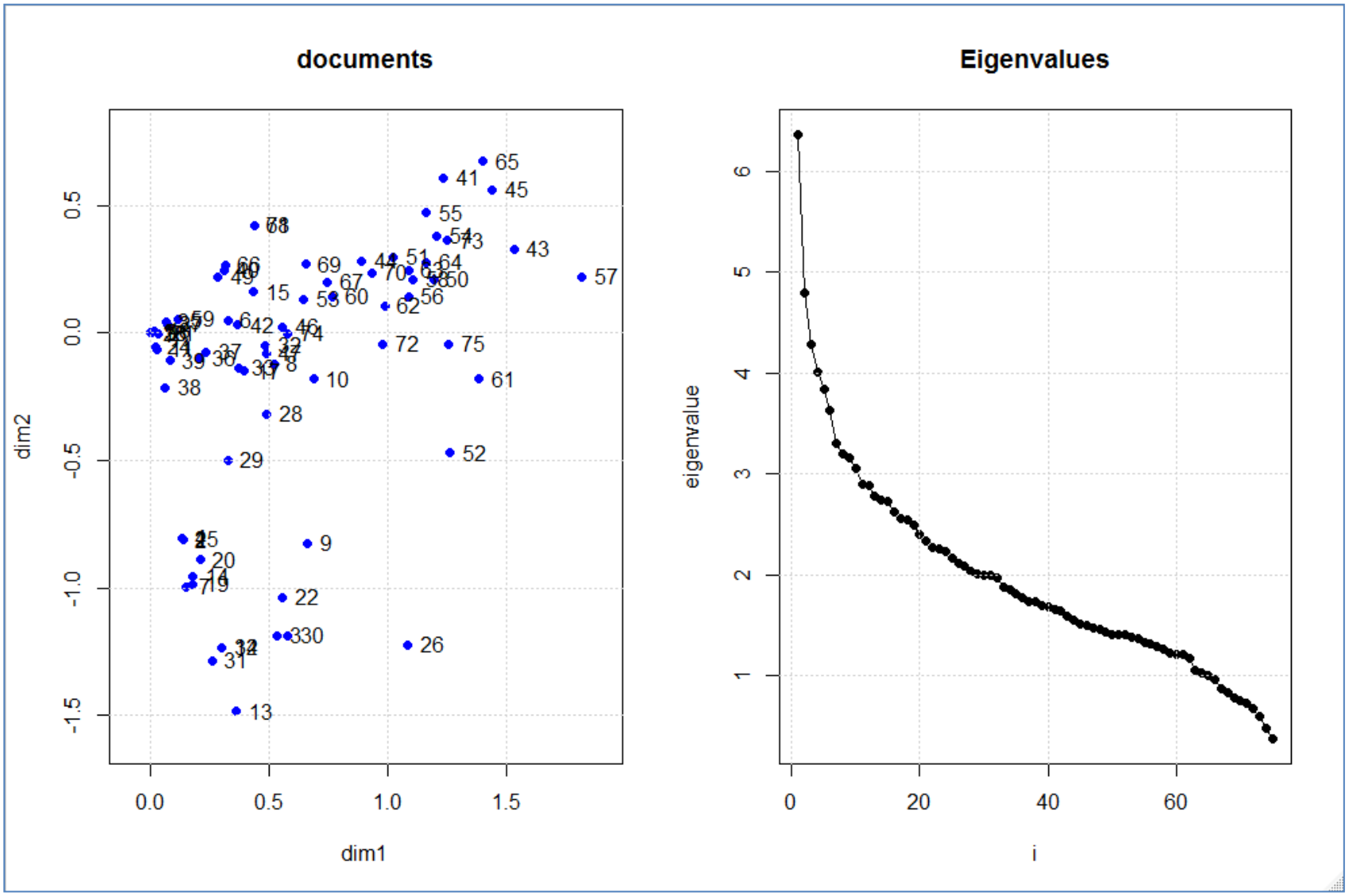
75 tytułów artykułów z 2 różnych dziedzin (Statistics & Control)

```
40 [C] Approximation of Large-Scale Dynamical Systems
41 [C] Control and Estimation in Distributed Parameter Systems
42 [C] Dynamic Noncooperative Game Theory, Second Edition
43 [C] Practical Methods for Optimal Control and Estimation Using Nonlinear Programming
44 [C] Control Perspectives on Numerical Algorithms and Matrix Problems
45 [C] Linear Matrix Inequalities in System and Control Theory
46 [C] Methods of Dynamic and Nonsmooth Optimization
47 [C] Optimization and Nonsmooth Analysis
48 [C] Shapes and Geometries
49 [C] Feedback Systems: Input-Output Properties
50 [C] Advances in Linear Matrix Inequality Methods in Control
51 [C] Perspectives in Flow Control and Optimization
52 [C] Applied Stochastic Processes and Control for Jump-Diffusions: Modeling, Analysis, and Computation
53 [C] Introduction to Shape Optimization: Theory, Approximation, and Computation
54 [C] Indefinite-Quadratic Estimation and Control: A Unified Approach to H2  and H• Theories
55 [C] Extending H-inf Control to Nonlinear Systems
56 [C] Classical Control Using H-inf Methods:
57 [C] Classical Control Using H-inf Methods: Theory, Optimization, and Design
58 [C] L1 Adaptive Control Theory: Guaranteed Robustness with Fast Adaptation
59 [C] Nonlinear Output Regulation:
60 [C] Adaptive Control Tutorial
61 [C] Singular Perturbation Methods in Control: Analysis and Design
62 [C] Boundary Control of PDEs: A Course on Backstepping Designs
63 [C] Mathematical Control Theory of Coupled PDEs
64 [C] Neuro-Fuzzy Control of Industrial Systems with Actuator Nonlinearities
65 [C] Some Aspects of the Optimal Control of Distributed Parameter Systems
66 [C] Stability and Stabilization of Time-Delay Systems: An Eigenvalue-Based Approach
67 [C] Control in an Information Rich World
68 [C] Strongly Stabilizable Distributed Parameter Systems
69 [C] Applied Dynamic Programming for Optimization of Dynamical Systems
70 [C] UAV Cooperative Decision and Control: Challenges and Practical Approaches
71 [C] Research Directions in Distributed Parameter Systems
72 [C] Stochastic Processes, Estimation, and Control
73 [C] Primer on Optimal Control Theory
74 [C] Nonlinear Systems Analysis, Second Edition
75 [C] Linear Feedback Control: Analysis and Design with MATLAB
```

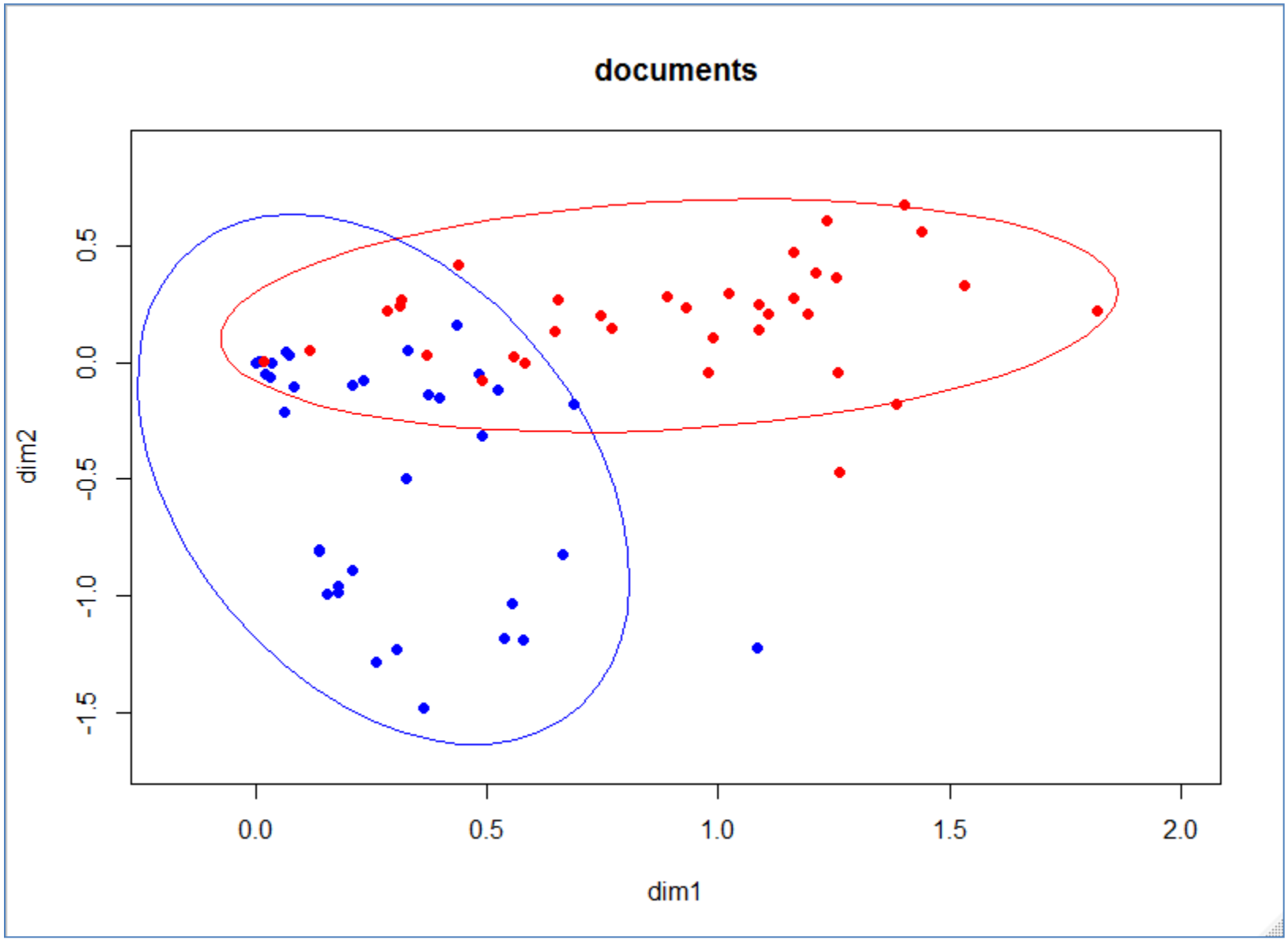
TDM według reguły Binary, po preprecessingu powstaje 177 termów

```
  [1] "academ"           "actuat"           "adapt"            "advanc"           "algorithm"
  [6] "analysi"          "analyz"           "anova"            "antholog"         "appli"
 [11] "applic"           "approach"         "approxim"         "aspect"           "backstep"
 [16] "base"             "basic"            "bayesian"         "bootstrap"        "boundari"
 [21] "case"             "challeng"         "classic"          "close"            "collabor"
 [26] "combinator"       "compar"           "comput"           "concept"          "confid"
 [31] "control"          "cooper"           "coupl"            "data"             "decis"
 [36] "design"           "deviat"           "dictionari"       "direct"           "diseas"
 [41] "distribut"        "dynam"            "edit"             "eigenvaluebas"    "elicit"
 [46] "emphas"           "empir"            "engin"            "englishrussian"   "error"
 [51] "estim"            "event"            "experi"           "experiment"       "expert"
 [56] "extend"           "fast"             "feedback"         "flow"             "formul"
 [61] "function"         "game"             "gase"             "gaug"             "geometri"
 [66] "graphic"          "guarante"         "guid"             "gupta"            "hinf"
 [71] "improv"           "indefinitequadrat" "industri"        "inequ"            "inform"
 [76] "inputoutput"      "intens"           "interv"           "introduct"        "jackknif"
 [81] "judgment"         "jumpdiffus"       "laboratori"       "larg"             "largescal"
 [86] "limit"            "linear"           "liquid"           "make"             "mathemat"
 [91] "mathematica"      "matlab"           "matric"           "matrix"           "mean"
 [96] "measur"           "method"           "methodolog"       "model"            "multipl"
[101] "multivari"        "network"          "neural"           "neurofuzzi"       "noncoop"
[106] "nonlinear"        "nonparametr"      "nonsmooth"        "numer"            "observ"
[111] "optim"            "order"            "output"           "panchapakesan"    "paramet"
[116] "pdes"             "perspect"         "perturb"          "pierr"            "pitman"
[121] "plan"             "popul"            "practic"          "primer"           "probabl"
[126] "problem"          "procedur"         "process"          "product"          "program"
[131] "properti"         "proport"          "proxim"           "quantil"          "random"
[136] "rank"             "rate"             "recurr"           "regul"            "reliabl"
[141] "repair"           "represent"        "resampl"          "research"         "review"
[146] "rich"             "robust"           "russianenglish"   "sampl"            "select"
[151] "sequenti"         "seri"             "shanti"           "shape"            "signific"
[156] "simul"            "singular"         "solid"            "spline"           "sport"
[161] "stabil"           "stabiliz"         "statist"          "stochast"         "strong"
[166] "structur"         "studi"            "system"           "test"             "theorem"
[171] "theori"           "time"             "timedelay"        "tutori"           "uav"
[176] "unifi"            "world"
>
```

k = 2 approximation



k = 2 approximation



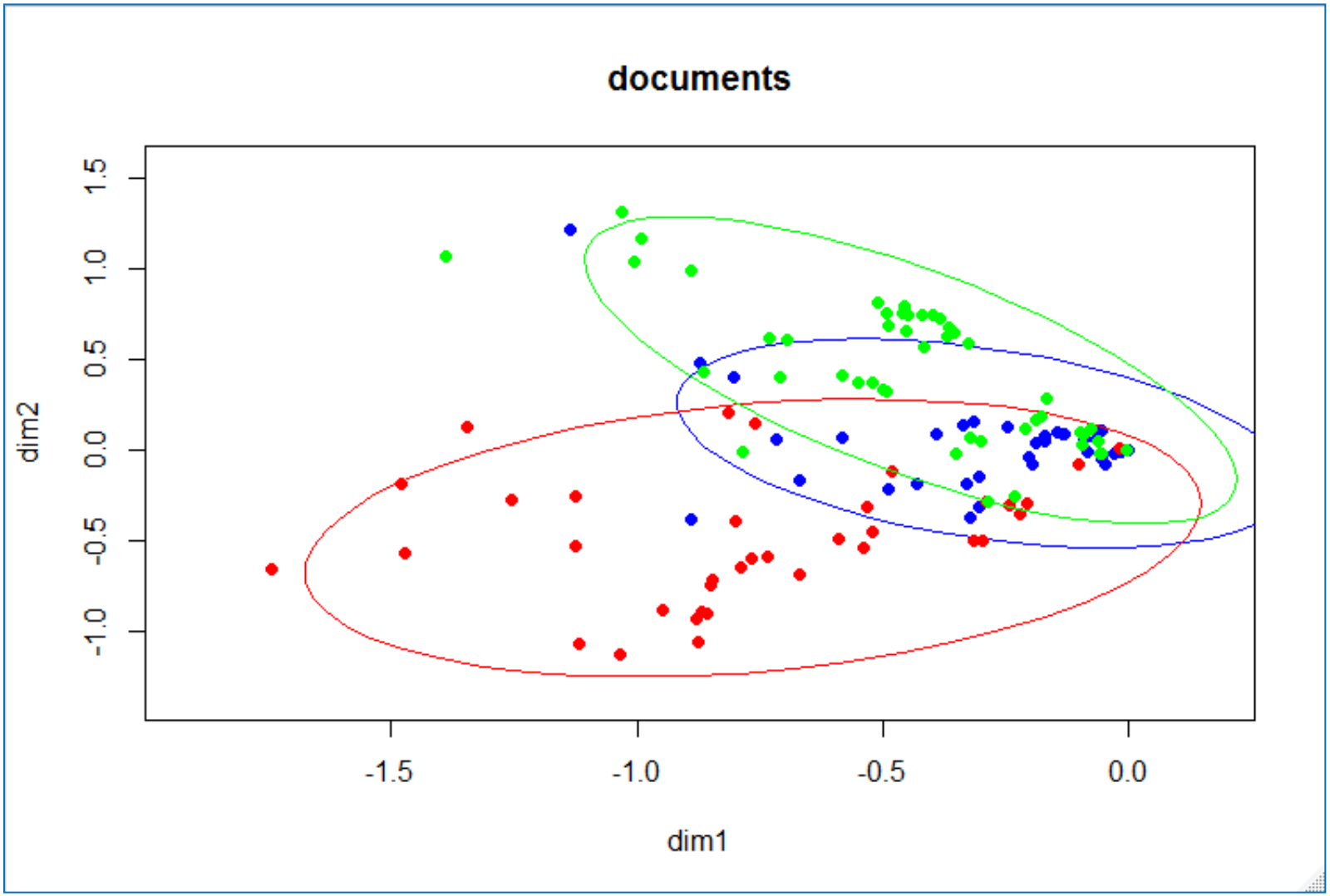
Dodajmy 3 klasę (Numerical Methods)

```
76 [M] Numerical Methods for Evolutionary Differential Equations
77 [M] Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations
78 [M] Applications on Advanced Architecture Computers
79 [M] Understanding Search Engines: Mathematical Modeling and Text Retrieval
80 [M] Numerical Methods for Least Squares
81 [M] Numerical Solution of Initial-Value Problems in Differential-Algebraic
82 [M] A Multigrid Tutorial, Second Edition
83 [M] Mathematical Aspects of Numerical Grid Generation
84 [M] Lectures on Finite Precision Computations
85 [M] Computational Methods for Multiphase Flows in Porous Media
86 [M] Multivariate Approximation Theory: Selected Topics
87 [M] Handbook for Matrix Computations
88 [M] Direct Methods for Sparse Linear Systems
89 [M] A Tutorial on Elliptic PDE Solvers and Their Parallelization
90 [M] Learning MATLAB
91 [M] Accuracy and Reliability in Scientific Computing
92 [M] Parallel Algorithms for Matrix Computations
93 [M] Theory of the Combination of Observations Least Subject to Errors
94 [M] Performance Optimization of Numerically Intensive Codes
95 [M] Symbolic Computation Applications to Scientific Computing
96 [M] Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion
97 [M] Parallel Processing for Scientific Computing
98 [M] MATLAB Guide, Second Edition
99 [M] The Science of Computer Benchmarking
100 [M] Finite Element Methods with B-Splines
101 [M] Solving Nonlinear Equations with Newton's Method
102 [M] Parallel MATLAB for Multicore and Multinode Computers
103 [M] Domain-Based Parallelism and Problem Decomposition Methods in Computational Science and Engineering
104 [M] Computational Integration
105 [M] Solving Least Squares Problems
106 [M] ARPACK Users' Guide
107 [M] The Immersed Interface Method
108 [M] Multigrid Methods
109 [M] Multilevel Projection Methods for Partial Differential Equations
110 [M] Numerical Computing with MATLAB, Revised Reprint
111 [M] Introduction to Interval Analysis
112 [M] Methods and Applications of Interval Analysis
113 [M] Solving Polynomials Using Continuation for Engineering and Scientific Problems
114 [M] Scientific Computing with Case Studies
115 [M] Numerical Computing with IEEE Floating Point Arithmetic
116 [M] Mathematical and Computational Techniques for Multilevel Adaptive Methods
117 [M] Solving PDEs in C++
118 [M] Design Sensitivity Analysis: Computational Issues of Sensitivity Equation Methods
119 [M] Numerical Polynomial Algebra
120 [M] Computational Science and Engineering
121 [M] Spectral Methods in MATLAB
122 [M] Computational Frameworks for the Fast Fourier Transform
```

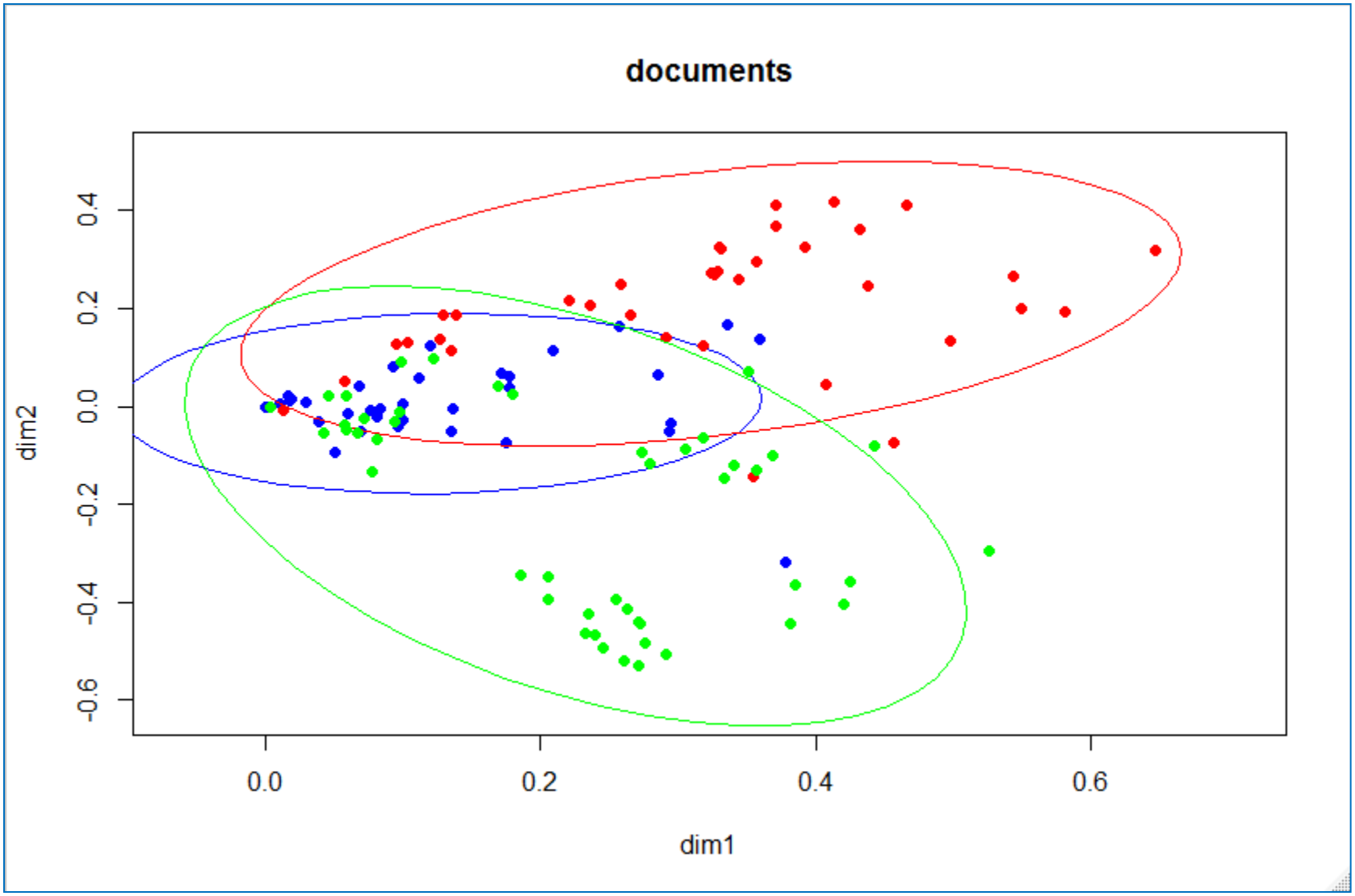
TDM według reguły Binary, po preprecessingu powstaje 252 termów

```
[1] "academ"              "accuraci"           "actuat"              "adapt"              "advanc"              "algebra"
  [7] "algorithm"           "analysi"            "analyz"              "anova"              "anthолog"            "appli"
 [13] "applic"              "approach"           "approxim"            "architectur"        "arithmet"            "arpack"
 [19] "aspect"              "backstep"           "base"                "basic"              "bayesian"            "benchmark"
 [25] "bootstrap"           "boundari"           "bspline"             "case"               "challeng"            "classic"
 [31] "close"               "code"               "collabor"            "combin"             "combinator"          "compar"
 [37] "comput"              "concept"            "confid"              "continu"            "control"             "cooper"
 [43] "coupl"               "data"               "decis"               "decomposit"         "design"              "deviat"
 [49] "dictionari"          "differenti"         "differentialalgebra" "direct"             "discret"             "diseas"
 [55] "distribut"           "domainbas"          "dynam"               "edit"               "eigenvaluebas"       "element"
 [61] "elicit"              "ellipt"             "emphas"              "empir"              "engin"               "englishrussian"
 [67] "equat"               "error"              "estim"               "event"              "evolutionari"        "experi"
 [73] "experiment"          "expert"             "extend"              "fast"               "feedback"            "finit"
 [79] "float"               "flow"               "formul"              "fourier"            "framework"           "function"
 [85] "game"                "gase"               "gaug"                "generat"            "geometri"            "graphic"
 [91] "grid"                "guarante"           "guid"                "gupta"              "handbook"            "hinf"
 [97] "ieee"                "illpos"             "immers"              "improv"             "indefinitequadrat"   "industri"
[103] "inequ"               "inform"             "initialvalu"         "inputoutput"        "integr"              "intens"
[109] "interfac"            "interv"             "introduct"           "invers"             "issu"                "jackknif"
[115] "judgment"            "jumpdiffus"         "laboratori"          "larg"               "largescal"           "learn"
[121] "lectur"              "limit"              "linear"              "liquid"             "make"                "mathemat"
[127] "mathematica"         "matlab"             "matric"              "matrix"             "mean"                "measur"
[133] "media"               "method"             "methodolog"          "model"              "multicor"            "multigrid"
[139] "multilevel"          "multinod"           "multiphas"           "multipl"            "multivari"           "network"
[145] "neural"              "neurofuzzi"         "newton"              "noncoop"            "nonlinear"           "nonparametr"
[151] "nonsmooth"           "numer"              "observ"              "optim"              "order"               "ordinari"
[157] "output"              "panchapakesan"      "parallel"            "paramet"            "partial"             "pde"
[163] "pdes"                "perform"            "perspect"            "perturb"            "pierr"               "pitman"
[169] "plan"                "point"              "polynomi"            "popul"              "porous"              "practic"
[175] "precis"              "primer"             "probabl"             "problem"            "procedur"            "process"
[181] "product"             "program"            "project"             "properti"           "proport"             "proxim"
[187] "quantil"             "random"             "rank"                "rankdefici"         "rate"                "recurr"
[193] "regul"               "reliabl"            "repair"              "represent"          "reprint"             "resampl"
[199] "research"            "retriev"            "review"              "revis"              "rich"                "robust"
[205] "russianenglish"      "sampl"              "scienc"              "scientif"           "search"              "select"
[211] "sensit"              "sequenti"           "seri"                "shanti"             "shape"               "signific"
[217] "simul"               "singular"           "solid"               "solut"              "solv"                "solver"
[223] "spars"               "spectral"           "spline"              "sport"              "squar"               "stabil"
[229] "stabiliz"            "statist"            "stochast"            "strong"             "structur"            "studi"
[235] "subject"             "symbol"             "system"              "techniqu"           "test"                "text"
[241] "theorem"             "theori"             "time"                "timedelay"          "topic"               "transform"
[247] "tutori"              "uav"                "understand"          "unifi"              "user"                "world"
```

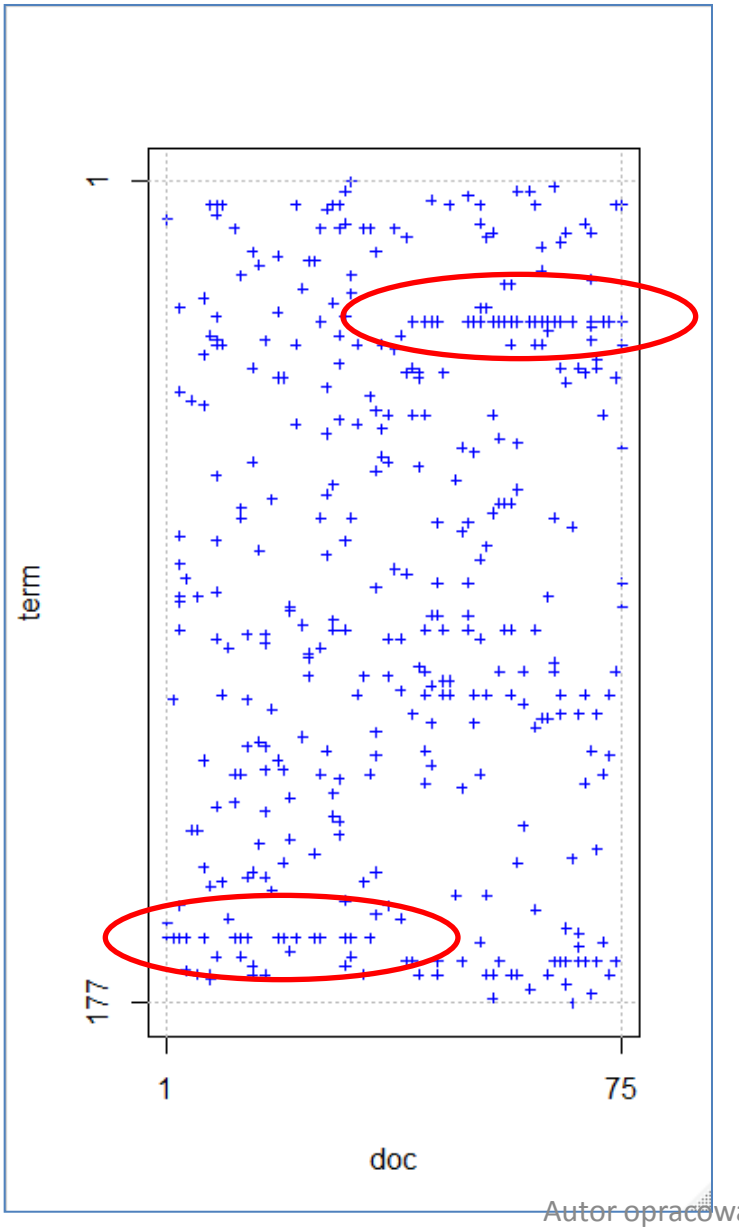
k = 2 approximation (tym razem jest nieco gorzej jednak)



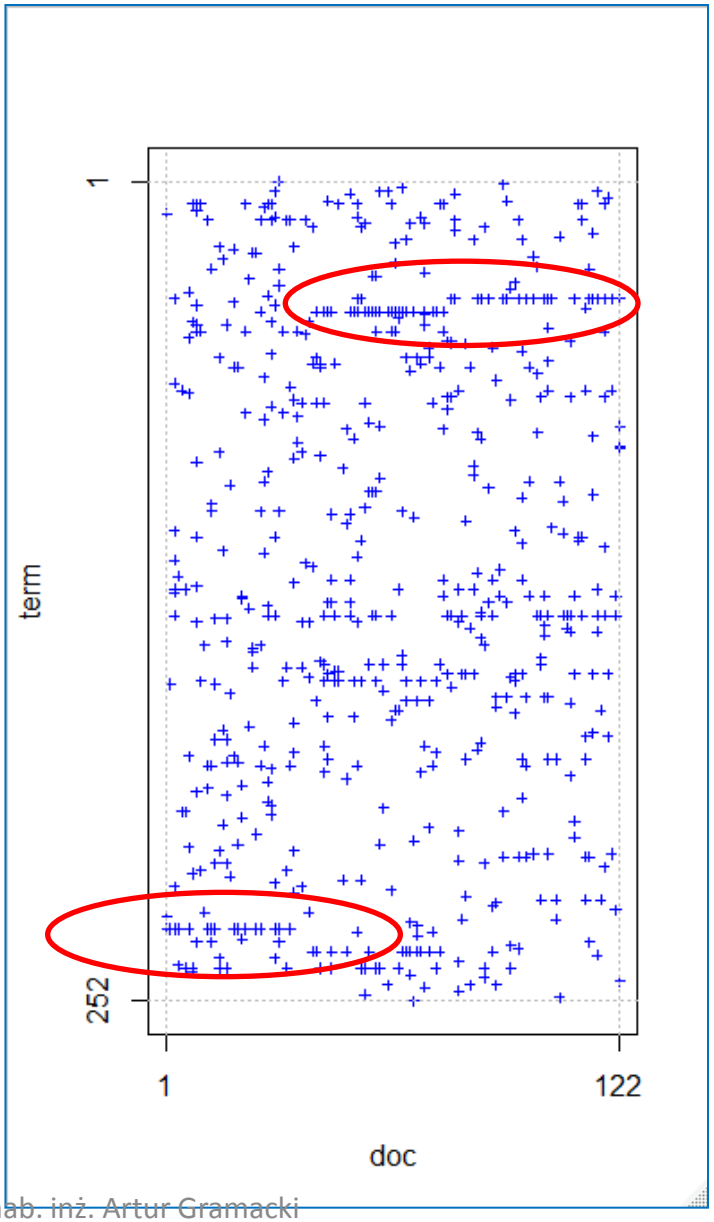
k = 2 approximation (TDM według reguły TFIDF)



Wizualizacja macierzy TDM (2 klasy)



Wizualizacja macierzy TDM (3 klasy)



Low-Rank Approximation przykład 4

- Przykład demonstracyjnego zbioru dokumentów MEDLINE

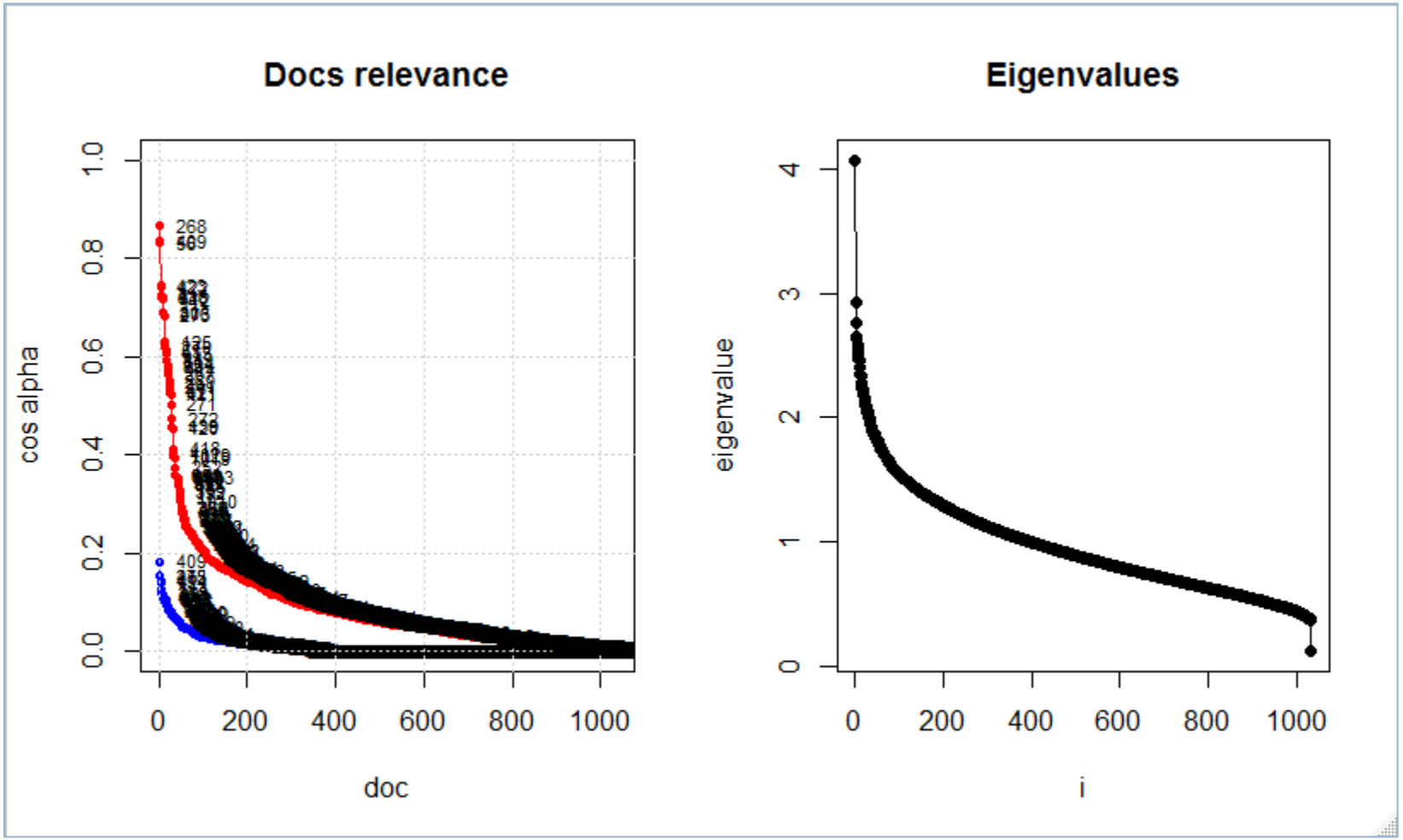
```
> doc <- read.docs(file = "MED.A.txt")
> query <- read.docs(file = "MED.Q.txt")
> (ldoc <- length(doc))
[1] 1033
> (lquery <- length(query))
[1] 30
> Q9.REL <- c(30,31,53,56,57,64,83,84,89,124,125,126,192,252,253,267,268,269,270,271,272,2
73,409,412,415,420,421,422)
> tdm <- create.TDM(doc = doc, query = query, weighting = "TfIdf.Norm.T")
> inspect(tdm)
<<TermDocumentMatrix (terms: 9312, documents: 1063)>>
Non-/sparse entries: 54436/9844220
Sparsity           : 99%
Maximal term length: 31
Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
Sample             :
          Docs
Terms      1060 335    34    520 606 727 729 732 765   823
  acid        0   0 0.000 0.0000   0   0   0   0   0 0.000
  cancer      0   0 0.000 0.0000   0   0   0   0   0 0.000
  case        0   0 0.109 0.0000   0   0   0   0   0 0.129
  cell        0   0 0.000 0.0000   0   0   0   0   0 0.000
  children    0   0 0.000 0.0000   0   0   0   0   0 0.000
  effect      0   0 0.000 0.0566   0   0   0   0   0 0.000
  growth      0   0 0.000 0.0000   0   0   0   0   0 0.000
  hormon      0   0 0.000 0.0000   0   0   0   0   0 0.000
  patient     0   0 0.000 0.0000   0   0   0   0   0 0.000
  rat         0   0 0.000 0.0000   0   0   0   0   0 0.000
>
```

Mamy 1033 dokumenty oraz 30 query (patrz pliki MED.A.txt oraz MED.Q.txt)

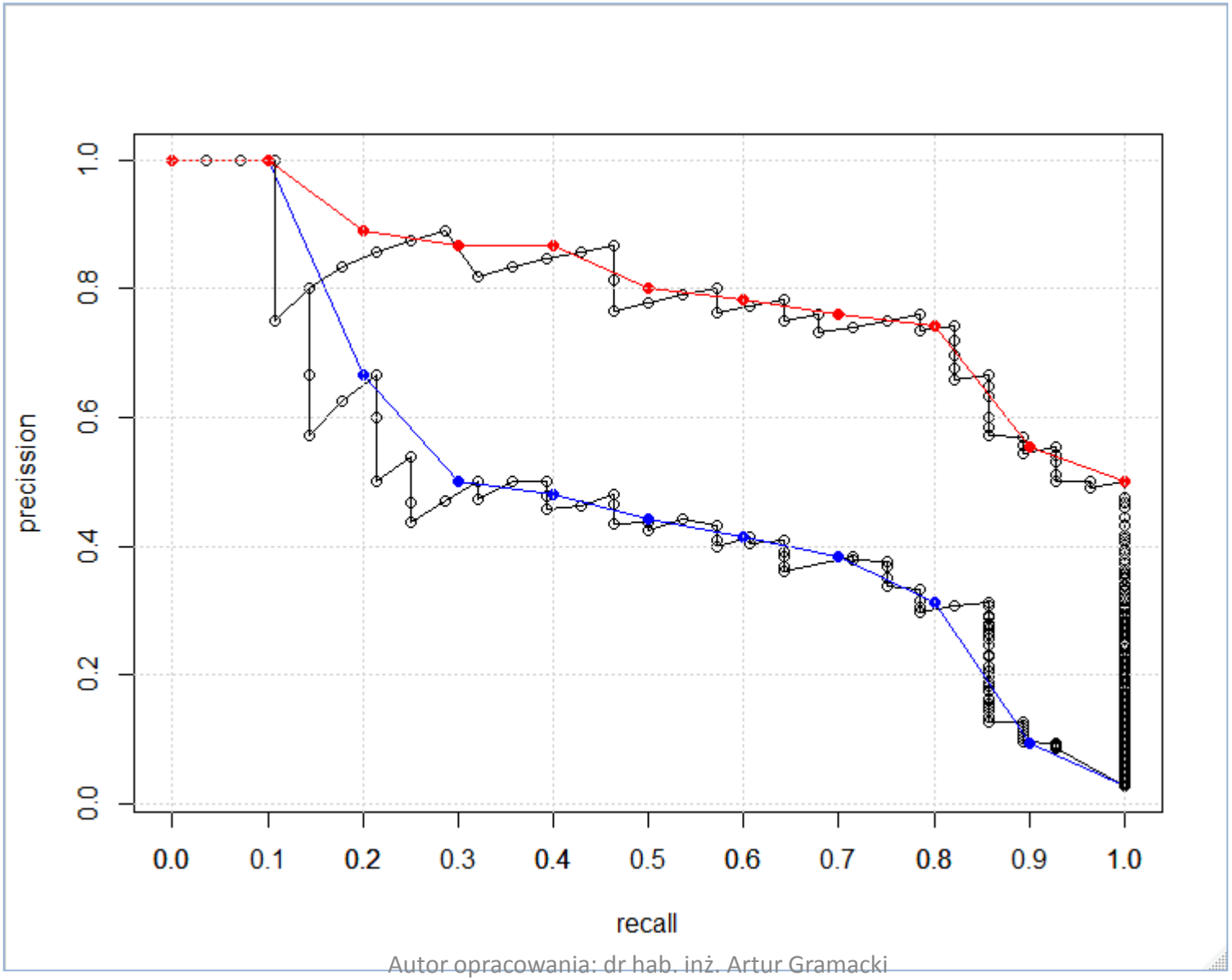
- Rank-k approximation

```
# Uwaga, dla bazy MEDLINE obliczenia mogą chwilę trwać.
out <-
  TDM.Rank.k(
    tdm = tdm,
    ldoc = ldoc,
    lquery = lquery,
    k = 100,
    q.number = 9,
    plot = TRUE,
    cex = 1
  )
```

- Wykresy cos(α) oraz osypiskowy



- Wykresy Precision-Recall

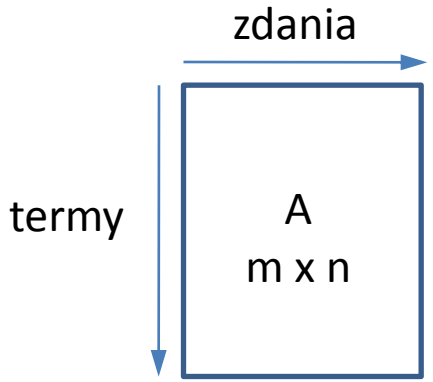


Automatyczne tworzenie podsumowań

- Zadaniem systemów automatycznie generujących podsumowania (ang. *automatic text summarization*) jest wybór z analizowanego tekstu tych zdań, które najlepiej oddają treść dokumentu
- Z reguły określa się z góry liczbę *p* wybranych zdań i / lub procentową objętość podsumowania w stosunku do całości tekstu
- Niezależnie od użytej metody, zadanie sprowadza się do określenia ważności (wagi) każdego zdania według jakiegoś kryterium a następnie wybraniu *p* zdań o największej wadze
- W literaturze angielskiej dotyczącej ogólnie dyscypliny naukowej IR używa się pojęcia *saliency score (SC)*
  - *saliency – najistotniejszy, najbardziej rzucający się w oczy*

Automatyczne tworzenie podsumowań

- Każde zdanie będziemy traktować jako niezależny dokument
  - mamy więc tym razem Term-Sentence Matrix (TSM)



- Istota procedury to tzw. zasada wzajemnego wzmocnienia (ang. *mutual reinforcement principle*)
  - term ma wysokie SC, gdy występuje w wielu zdaniach z wysokim SC
  - zdanie ma wysokie SC, gdy zawiera wiele słów z wysokim SC

Automatyczne tworzenie podsumowań

- Oznaczenia
  - t_i – saliency score termu *i*
  - s_j - saliency score zdania *j*
- Algorytm

$$t_i \propto \sum_{j=1}^{n} a_{ij} s_j \quad i = 1,2,\cdots,m$$

$$s_j \propto \sum_{i=1}^{m} a_{ij} t_i \quad j = 1,2,\cdots,n$$

Uwaga na indeksy (i, j, m, n). Łatwo się pomylić

Waga termu *i* jest proporcjonalna do sumy wag zdań, w którym się on pojawia

Waga zdania *j* jest proporcjonalna do sumy wag słów w tym zdaniu

Grupując dalej te elementy w wektory o wymiarach odpowiednio *m* i *n* otrzymujemy:

$$\beta_t \mathbf{t} = \mathbf{A}\mathbf{s},$$

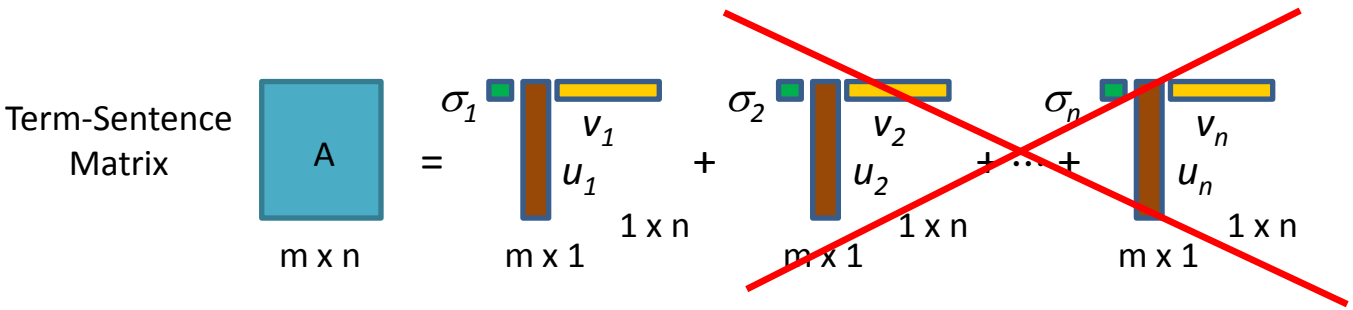
$$\beta_s \mathbf{s} = \mathbf{A}^T \mathbf{t},$$

Gdzie β_t, β_s są stałymi proporcjonalności. Podstawiając jedno równanie do drugiego otrzymujemy:

$$\beta_t \mathbf{t} = \frac{1}{\beta_s} \mathbf{A}\mathbf{A}^T \mathbf{t}, \quad \beta_s \mathbf{s} = \frac{1}{\beta_t} \mathbf{A}^T \mathbf{A}\mathbf{s}$$

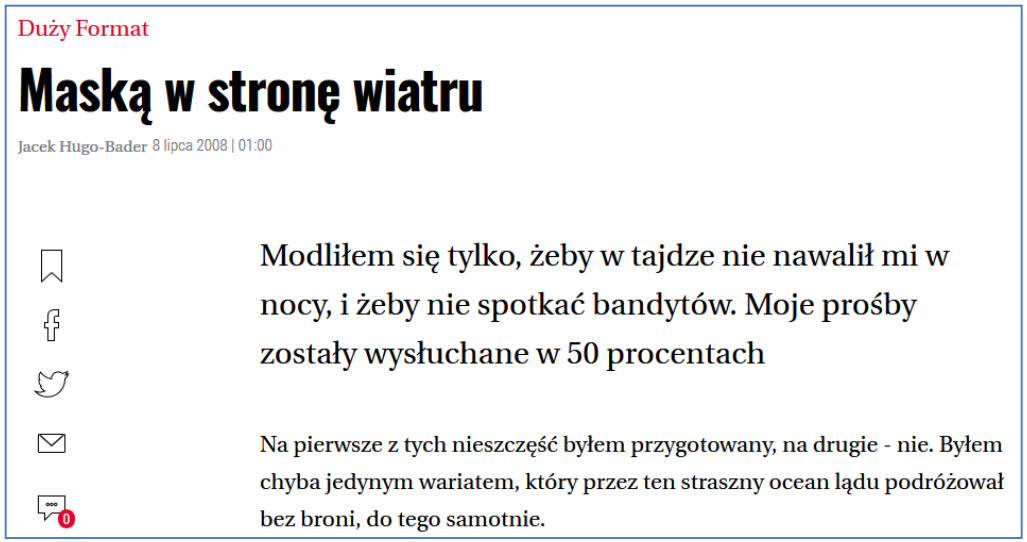
Automatyczne tworzenie podsumowań

- Algorytm, cd
  - przy założeniu, że β_t, β_s są sobie równe, łatwo zauważyć, że **t** oraz **s** są wektorami własnymi macierzy odpowiednio $\mathbf{AA}^T$ oraz $\mathbf{A}^T\mathbf{A}$ z tymi samymi wartościami własnymi
  - ponadto **t** oraz **s** są wektorami własnymi odpowiednich wartości własnych (patrz slajdy o SVD, była o tym mowa na slajdzie zatytułowanym „Związek SVD z eigenvalue problem”)
  - wykonując teraz rank-1 approximation możemy wynik interpretować tak, że
    - ✓ wektor *t* zawiera SC termów
    - ✓ wektor *s* zawiera SC zdań



Automatyczne tworzenie podsumowań

- Przykład
  - wybrano dosyć długi tekst z portalu „Duży Format” Gazety Wyborczej ttp://wyborcza.pl/1,95364,5427827,Maska_w_strone_wiatru.html Jest to reporterski opis dosyć szalonej samotnej podróży dziennikarza Jacka Hugo-Badera z Moskwy na daleką Syberię. Z tekstu usunięto słowa ze stop listy. Tekst nie został poddany stemmingowi (trudności z uwzględnieniem powtórzeń bardzo wielu słów). Otrzymana macierz TSM ma wymiary 2232 *x* 428



Duży Format

Maską w stronę wiatru

Jacek Hugo-Bader 8 lipca 2008 | 01:00

Modliłem się tylko, żeby w tajdze nie nawalił mi w nocy, i żeby nie spotkać bandytów. Moje prośby zostały wysłuchane w 50 procentach

Na pierwsze z tych nieszczęść byłem przygotowany, na drugie - nie. Byłem chyba jedynym wariatem, który przez ten straszny ocean lądu podróżował bez broni, do tego samotnie.

Automatyczne tworzenie podsumowań

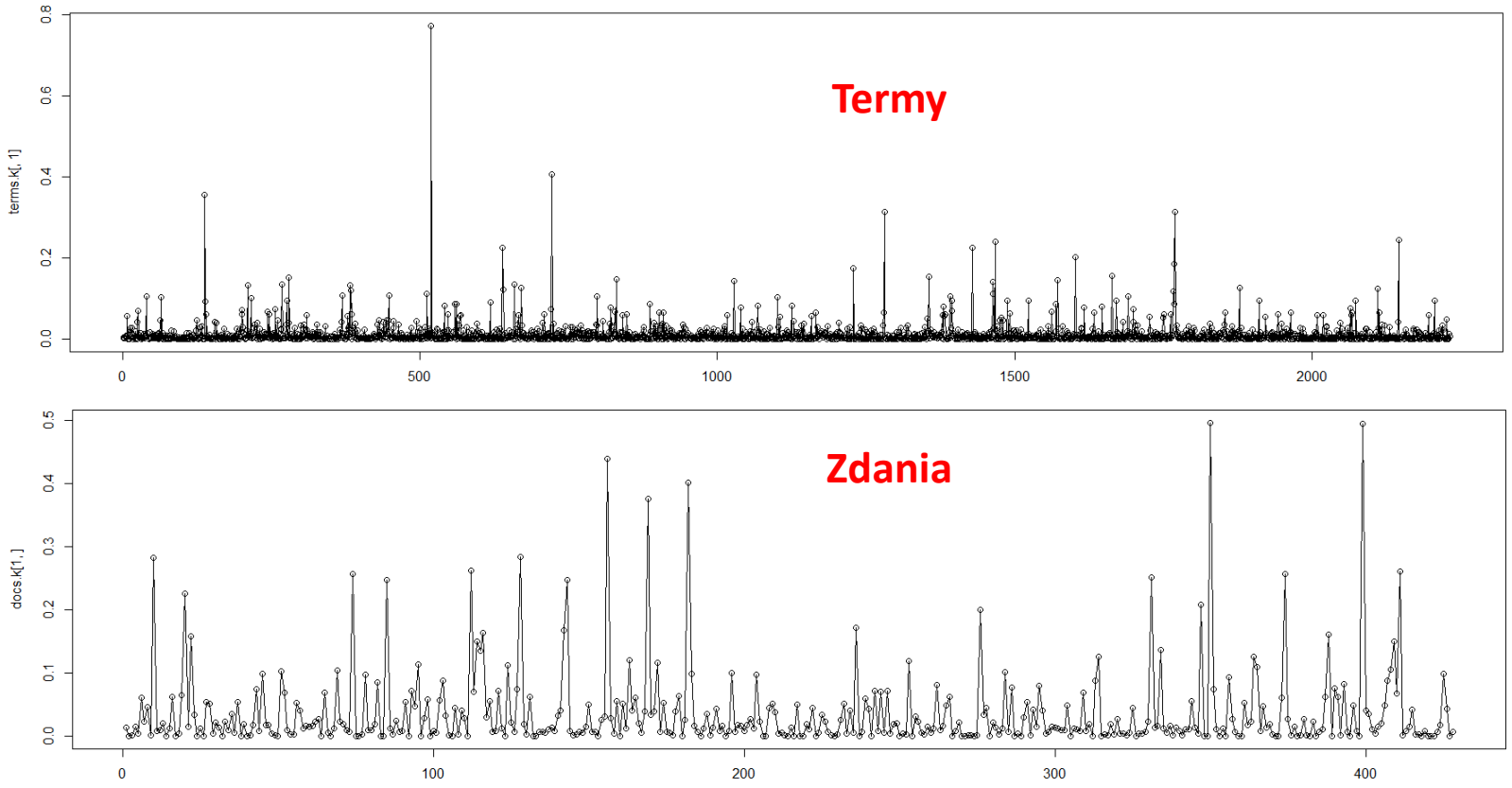
- Pierwsze 10 zdań przed i po preprocessingu

[1] Modliłem się tylko, żeby w tajdze nie nawalił mi w nocy, i żeby nie spotkać bandytów.
[2] Moje prośby zostały wysłuchane w 50 procentach.
[3] Na pierwsze z tych nieszczęść byłem przygotowany, na drugie - nie.
[4] Byłem chyba jedynym wariatem, który przez ten straszny ocean lądu podróżował bez broni, do tego samotnie.
[5] Ulubionym sportem miejscowych jest strzelectwo.
[6] Jadą normalnie, po prawej stronie drogi, ale kierownicę też mają z prawej, bo to samochody z Japonii.
[7] Kierownicę trzymają w lewej ręce, więc prawą bardzo łatwo wystawić przez okno i walić w biegu z tego, w co są uzbrojeni, do tablic informacyjnych, znaków drogowych i reklam.
[8] We wschodniej Syberii nie widziałem jednego drogowskazu, który by nie był podziurawiony jak durszlak.
[9] Małe i duże kalibry, ogień pojedynczy, serie, a czasem ogromne wyrwy po śrutowej armacie.
[10] A co kilkadziesiąt kilometrów wrak spalonego samochodu.

[1] modliłem tajdze nawalił nocy spotkać bandytów
[2] prośby zostały wysłuchane procentach
[3] pierwsze nieszczęść byłem przygotowany drugie
[4] byłem chyba jedynym wariatem straszny ocean lądu podróżował broni samotnie
[5] ulubionym sportem miejscowych strzelectwo
[6] jadą normalnie prawej stronie drogi kierownicę prawej samochody japonii
[7] kierownicę trzymają lewej ręce prawą łatwo wystawić okno walić biegu uzbrojeni tablic informacyjnych znaków drogowych reklam
[8] wschodniej syberii widziałem jednego drogowskazu podziurawiony durszlak
[9] małe duże kalibry ogień pojedynczy serie ogromne wyrwy śrutowej armacie
[10] kilkadziesiąt kilometrów wrak spalonego samochodu

Automatyczne tworzenie podsumowań

- Na rysunkach pokazano wykresy wektorów zdań oraz termów (te ostatnie są w macierzy posortowane alfabetycznie), na bazie których określono przedstawione w tabelach wyniki



Automatyczne tworzenie podsumowań

- Wynikowe 10 najważniejszych zdań oraz termów (posortowanych wg ważności)
  - przeczytaj samodzielnie tekst źródłowy i sam oceń trafność podsumowań

"kilometrów"	"miałem"	"chabarowska"	"tysięcy"	"przejechałem"	"złotych"	"samochód"	"rubli"	"litra"	"stu"

Miałem około stu kilometrów do Chabarowska.
Przez Rosję przejechałem 13 tysięcy kilometrów i zaprawiłem 2119 litrów benzyny, bo mój łazik żarł (tak mówią Rosjanie) średnio 16,3 litra paliwa na sto kilometrów.
Przejechałem już 7700 kilometrów.
Do Chabarowska dotarłem w końcu lutego, miałem stąd tylko 770 kilometrów niezłej drogi do Władywostoku, gdzie kończyłem podróż.
Ponad 2 tysiące kilometrów do Chabarowska tylko góry, bagna, śnieg i tajga.
Grisza wziął za robotę 9 tysięcy rubli (900 złotych), więc gotowe do drogi auto kosztowało mnie 201 tysięcy (20,1 tysiąca złotych).
A co kilkadziesiąt kilometrów wrak spalonego samochodu.
Jak dla mnie 170 tysięcy rubli (17 tysięcy złotych), ale Grisza wie, że jest w dobrym stanie.
50 kilometrów przed Szymanowskiem, między Czytą a Chabarowskiem, droga w robocie.
I miał dodge'a challengera rocznik 70 z silnikiem 4,4 litra, który wyciągał do 250 kilometrów na godzinę.

Algorytm PageRank

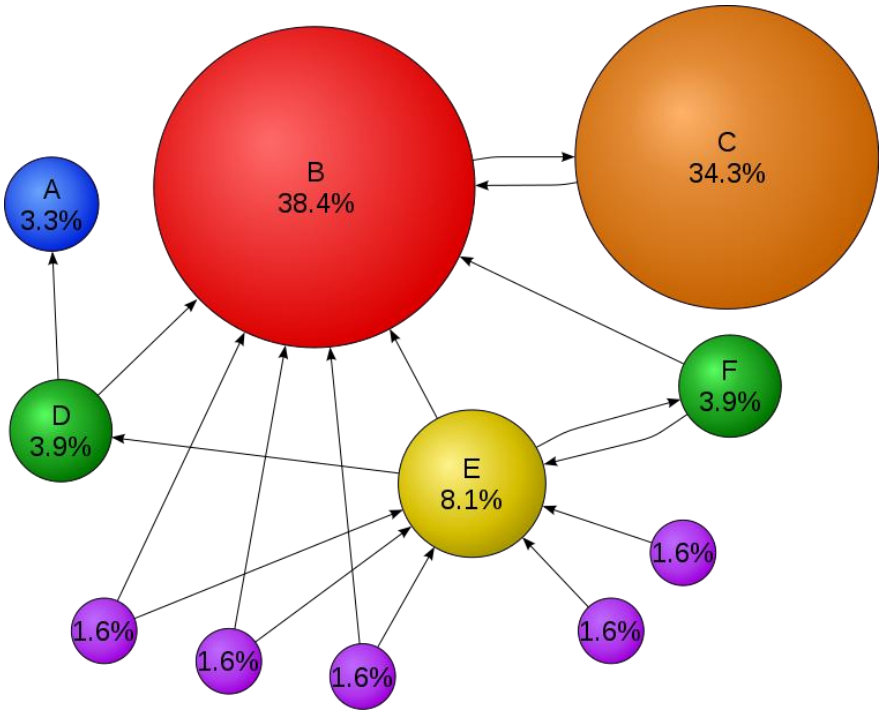
PageRank

- Metoda nadawania indeksowanym stronom internetowym określonej wartości liczbowej, oznaczającej ich jakość
- Algorytm PageRank jest wykorzystywany przez popularną wyszukiwarkę internetową Google. Został opracowany przez założycieli przedsiębiorstwa Google Larry'ego Page'a i Sergeya Brina podczas ich studiów na Uniwersytecie Stanforda w 1998 roku
- Nazwa algorytmu pochodzi nie od angielskiego wyrazu określającego stronę (ang. *page*), lecz od nazwiska twórcy, czyli Larry'ego Page'a

źródło: https://pl.wikipedia.org/wiki/PageRank

PageRank

- PageRank jest rozwinięciem znanej od dawna heurystyki, wedle której:
  - jakość tekstu jest proporcjonalna do liczby tekstów na niego się powołujących
  - cytowanie wysoko cytowanych tekstów jest ważniejsze niż cytowanie mniej popularnych tekstów



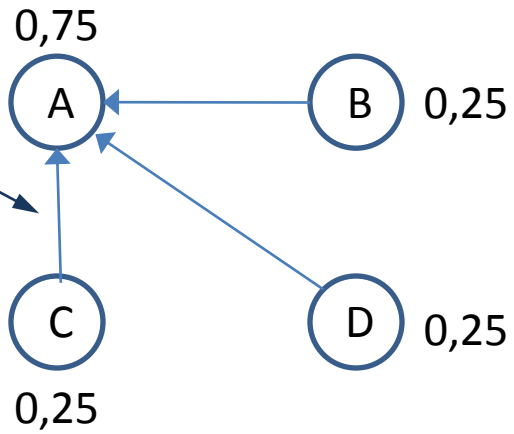
PageRank

- Linki między stronami stanowią ścieżki w grafie, po których użytkownicy podróżują od jednej strony do innej. Popularność danej strony można mierzyć według tego, jak często przeciętny użytkownik Internetu odwiedza daną stronę
- PageRank wykorzystuje metaforę „losowego surfera internetowego”. Surfer rozpoczyna swój spacer od dowolnej strony. W każdym kroku wybiera, z równym prawdopodobieństwem, jedną z dostępnych stron i przechodzi do niej

PageRank

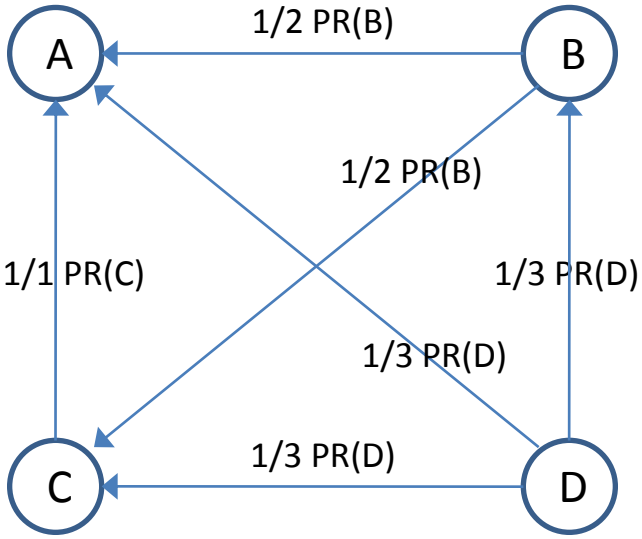
- Algorytm iteracyjny
  - krok 1: każda strona dostaje „na dzień dobry” taki sam ranking, w naszym przypadku po 0,25
  - krok 2: transfer swoich rankingów do stron, na które dana strona się powołuje
    PR(A) = PR(B) + PR(C) + PR(D) = 0,75

W angielskiej terminologii to są *outlinks* (linki wychodzące) lub *inlinks* (linki przychodzace)



PageRank

- Bardziej złożony przykład
  - krok 1: każdy link dostaje „na dzień dobry” taki sam ranking, w naszym przypadku po 0,25
  - krok 2:
    - ✓ B transferuje po 1/2 swojego aktualnego rankingu do A oraz do C
    - ✓ C transferuje cały swój aktualny ranking do A
    - ✓ D transferuje po 1/3 swojego aktualnego rankingu do A, B oraz do C
    - ✓ A niczego nie transferuje, bo żaden link z niego nie wychodzi



$$PR(A) = \frac{PR(B)}{2} + \frac{PR(C)}{1} + \frac{PR(D)}{3} = 0{,}125 + 0{,}25 + 0{,}083 = 0{,}458$$

  - krok 3 i kolejne: rozwiązujemy iteracyjnie, aż do osiągnięcia stanu równowagi

PageRank

- Powyższe można zapisać tak:

$$PR(A) = \frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \frac{PR(D)}{L(D)}$$

gdzie L(*) oznacza liczbę linków wychodzących (ang. outlinks) ze strony (*). Ogólnie można zapisać to tak:

$$PR(U) = \sum_{V \in N_U} \frac{PR(V)}{L(V)}$$

Zbiór wszystkich stron linkujących do strony U

PageRank

- Algorytm w wersji z tzw. współczynnikiem tłumienia *d* (ang. *damping factor*). Pozwala on na poprawne działanie algorytmu, gdy w sieci pojawią się strony bez linków wychodzących

$$PR(A) = \frac{1-d}{N} + d\left(\frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \frac{PR(D)}{L(D)} + \cdots\right)$$

Sugerowana wartość: d=0,85

$$PR(U) = \frac{1-d}{N} + d\sum_{V \in N_U} \frac{PR(V)}{L(V)}$$

N: całkowita liczba stron

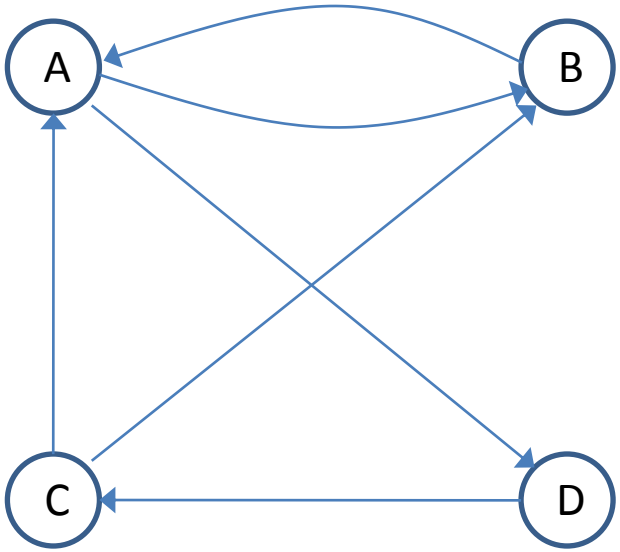
Zbiór wszystkich stron linkujących do strony U

PageRank

- Kolejny przykład
  - wynik końcowy (po odpowiedniej ilości iteracji, aż do osiągnięcia stanu równowagi, czyli, że PR(*) nie będą się już wiele zmieniały z kroku na krok)

PR(A) = 0,349
PR(B) = 0,269
PR(C) = 0,186
PR(D) = 0,196

SUMA = 1,00

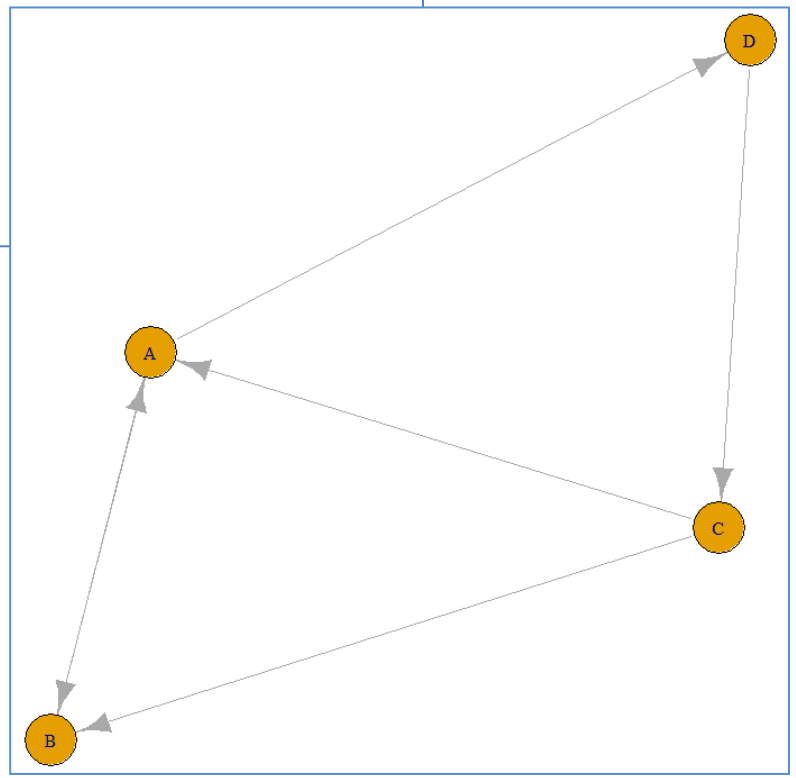


PageRank

- Przykład jak wyżej, implementacja w R

```
> library(igraph)

> graph <- graph.formula(A-+B, A-+D, B-+A, C-+B, C-+A, D-+C)
> pr <- page_rank(graph)

> pr$vector
        A         B         D         C
0.3493518 0.2690953 0.1859745 0.1955783

plot(graph)
```

plot(graph) generuje rysunek z losowym rozmieszczeniem węzłów (oczywiście linkowanie jest zgodne z podaną formułą)

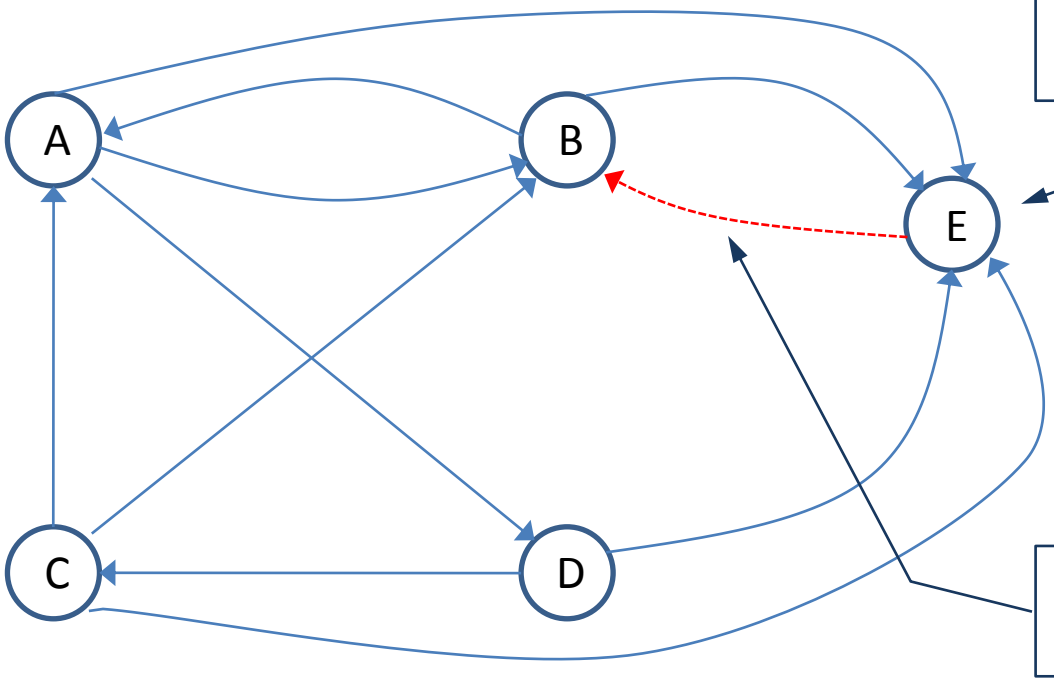


PageRank

- Kolejny przykład. Z jednej strony nie wychodzi żaden link.
  - skutek: losowy surfer, gdy stanie na tej stronie, utyka
  - recepta: skoczyć losowo do jakiejkolwiek innej strony. Symulujemy niejako, że strona E ma "połączenie" do wszystkich innych stron. Taki efekt zapewnia nap czynnik (1-d)/N

PR(A) = 0,20
PR(B) = 0,35
PR(C) = 0,067
PR(D) = 0,086
PR(E) = 0,29

Suma = 1



Ta strona nie linkuje do żadnej innej strony (tzw. strona "sink")

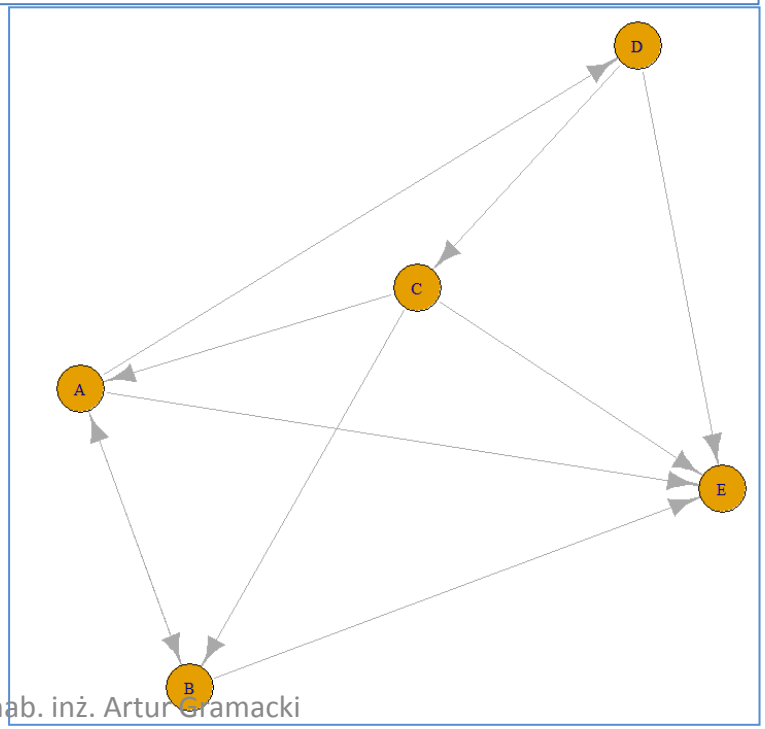
Gdyby pojawił się np. ten link, znika strona typu "sink"

PageRank

- Przykład jak wyżej, implementacja w R

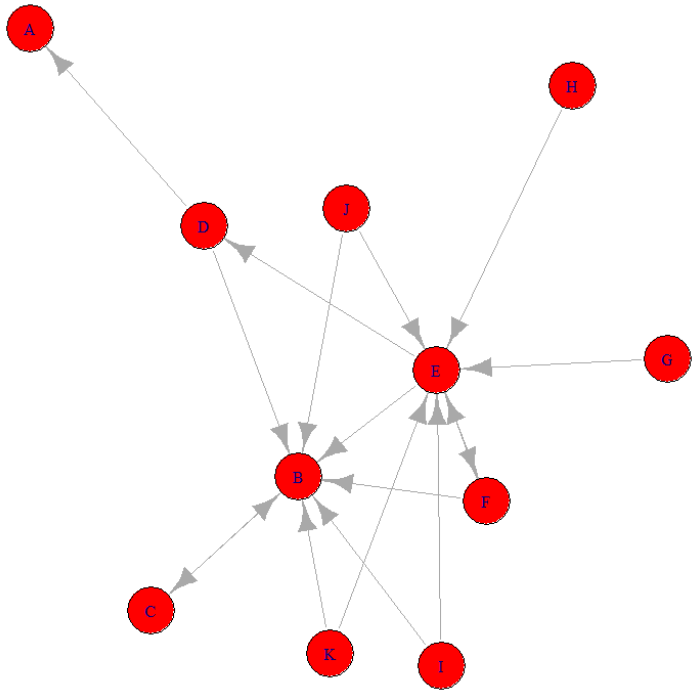
```
> library(igraph)
> graph <- graph.formula(A-+B, A-+D, B-+A, C-+B, C-+A, D-+C, A-+E,
                         B-+E, D-+E, C-+E)
> pr <- page_rank(graph)
> pr$vector
        A         B         D         C         E
0.2044151 0.1840931 0.1428339 0.1456207 0.3230371
```

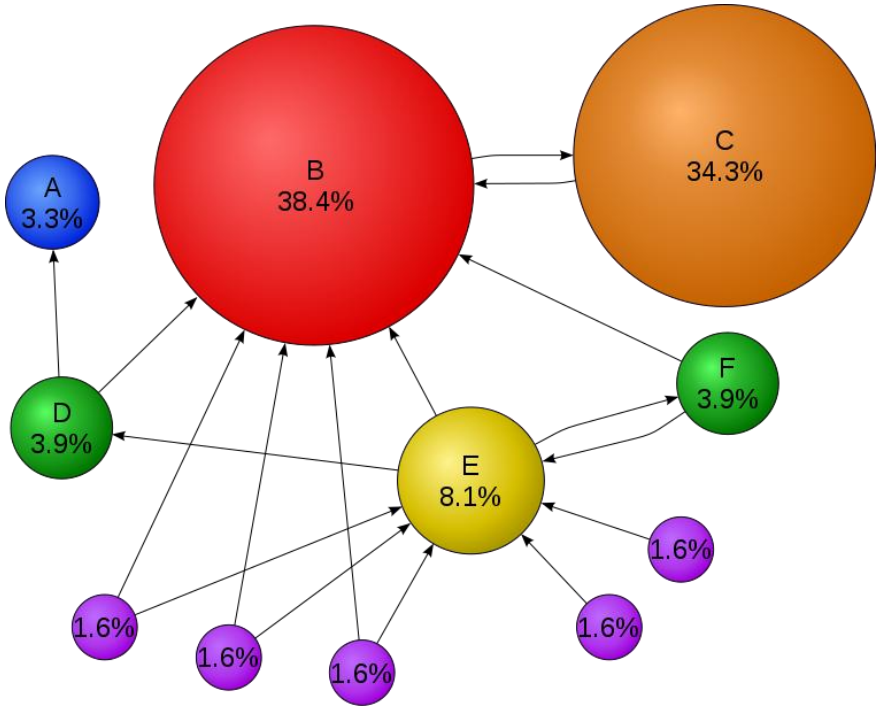
plot(graph) generuje rysunek z losowym rozmieszczeniem węzłów (oczywiście linkowanie jest zgodne z podaną formułą)



PageRank

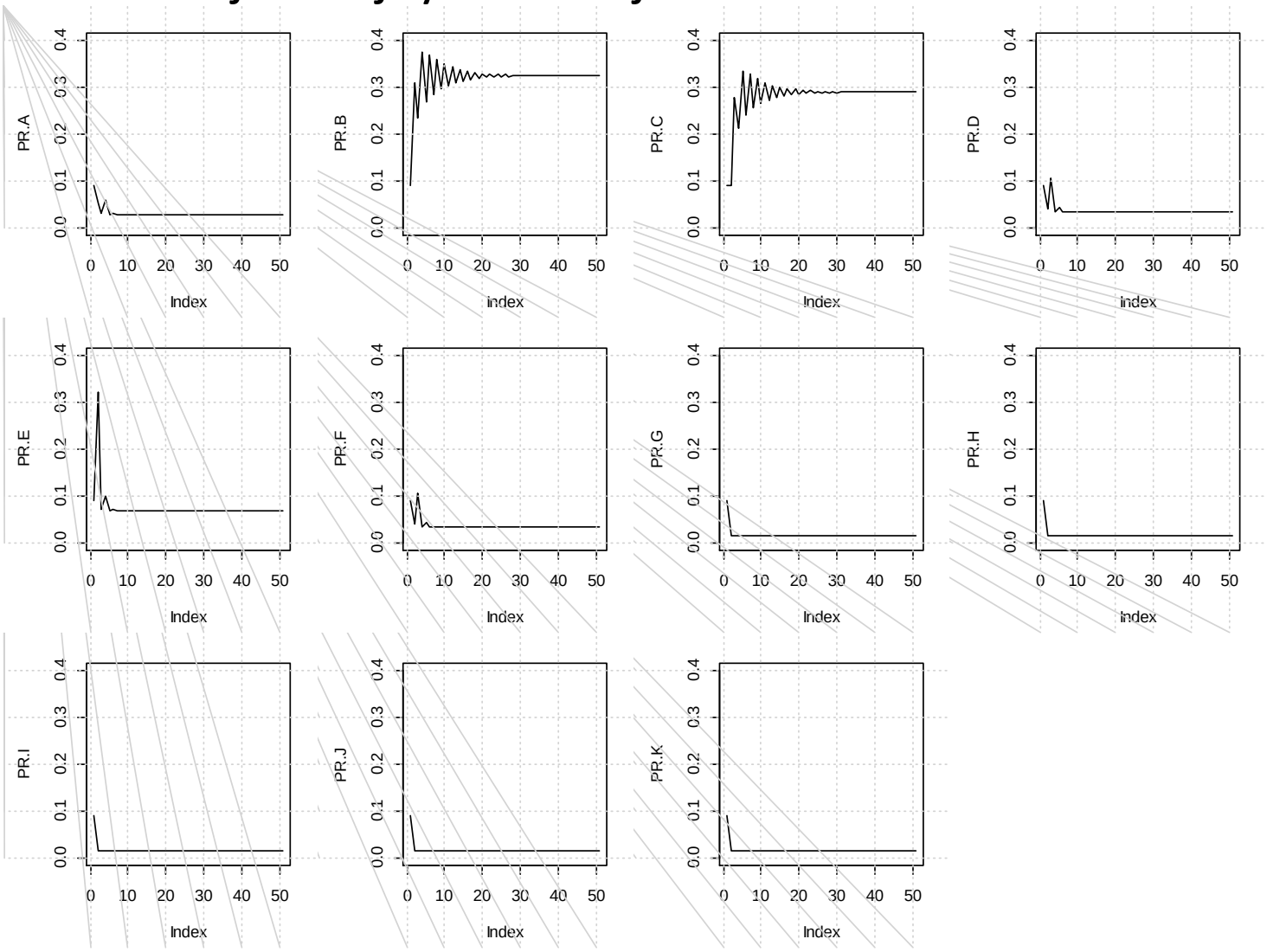
```
> graph <- graph.formula(
      D-+A, D-+B, C+B, E+F, F-+B, E-+B, G-+E, H-+E, I-+E, I-+B, J-+E, J-+B,
      K-+E, K-+B, E-+D)

> pr <- page_rank(graph)

> round(sort(pr$vector, decreasing = TRUE), 4) * 100
    B     C     E     D     F     A     G     H     I     J     K
38.44 34.29  8.09  3.91  3.91  3.28  1.62  1.62  1.62  1.62  1.62
```





PageRank

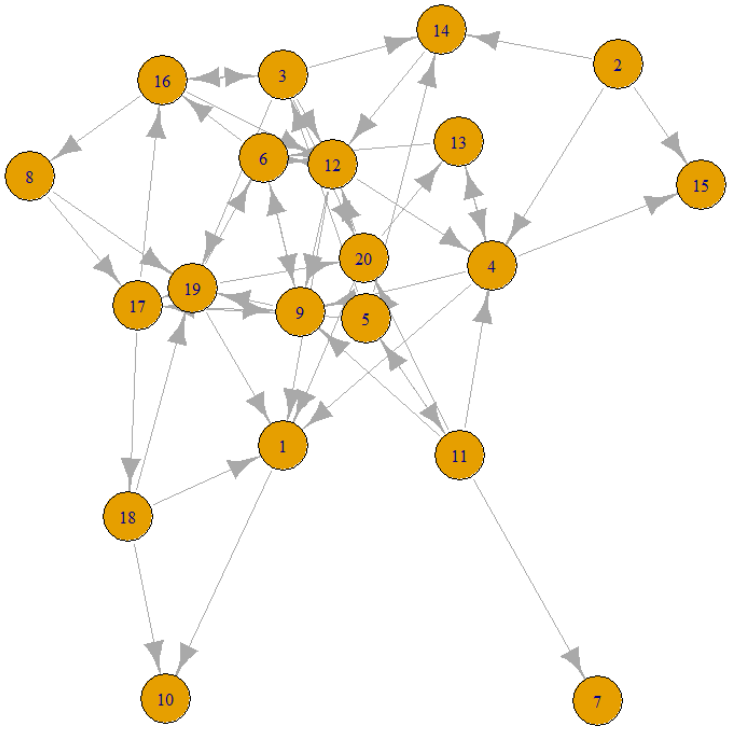
- Wizualizacja kolejnych iteracji



PageRank

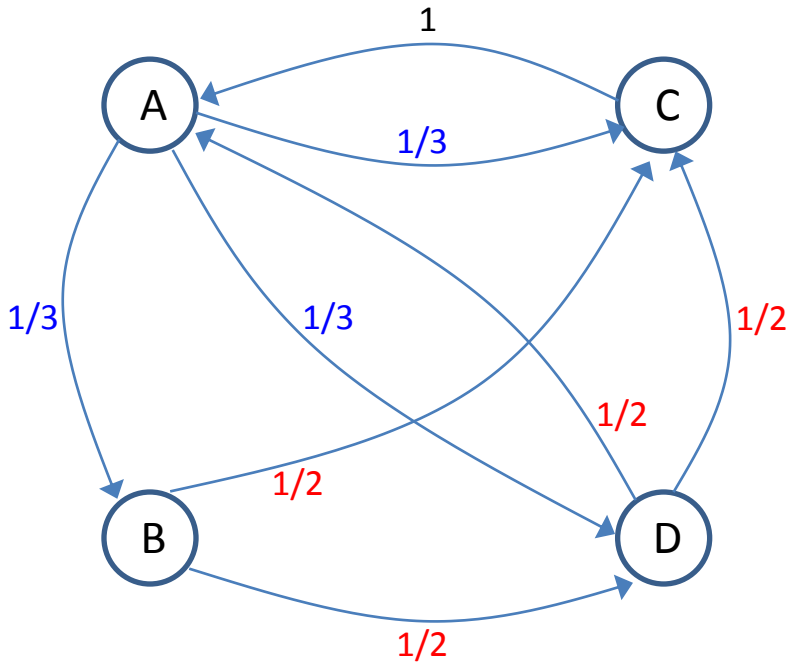
- Losowe generowanie grafu sieci

```
> graph <- sample_gnp(20, 3/20, directed = TRUE)
> plot(graph)
> round(page.rank(graph)$vector, 3)
 [1] 0.082 0.013 0.034 0.057 0.016 0.106 0.016 0.031 0.093 0.090 0.017 0.100 0.036
[14] 0.026 0.029 0.062 0.047 0.026 0.079 0.039
```



PageRank

- Algorytm algebraiczny
  - sieć zapisujemy w postaci macierzy zgodnie z zasadą: jeśli węzeł ma k krawędzi wychodzących, przekazuje swój aktualny PR każdemu węzłowi, do którego dochodzi



$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 1 & 1/2 \\ 1/3 & 0 & 0 & 0 \\ 1/3 & 1/2 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$

A B C D — linki wchodzące do węzłów

A B C D — linki wychodzące z węzłów

PageRank

- Algorytm algebraiczny
  - następnie musimy znaleźć PR, które osiągną stan ustalony (analogicznie jak w metodzie iteracyjnej, wcześniej omawianej)
  - zaczniemy od zapisania naszego zadania w postaci układu równań

$$\begin{cases} x_1 = 1 \cdot x_3 + \frac{1}{2} x_4 \\ x_2 = \frac{1}{3} \cdot x_1 \\ x_3 = \frac{1}{3} \cdot x_1 + \frac{1}{2} x_2 + \frac{1}{2} x_4 \\ x_4 = \frac{1}{3} \cdot x_1 + \frac{1}{2} x_2 \end{cases} \Rightarrow \quad \mathbf{A} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \quad \Rightarrow \quad \mathbf{Ax} = \lambda \mathbf{x}, \quad \lambda = 1$$

Musimy więc rozwiązać klasyczne zadanie własne (znaleźć wektor własny dla wartości własnej równej 1)
UWAGA: to jest uproszczone przedstawienie problemu. Nie zawsze największa wartość własna będzie równa 1. Tym problemem nie zajmujemy się tutaj

PageRank

- Algorytm algebraiczny, przykład w R

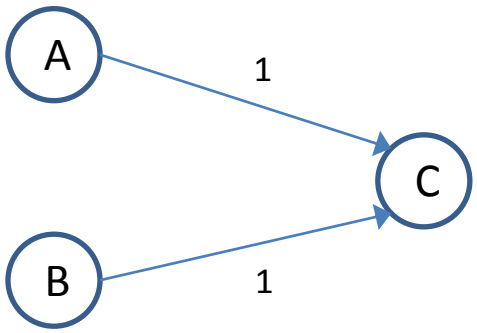
```
> P <- matrix(c(
+   0,   0,   1, 1/2,
+   1/3, 0,   0,   0,
+   1/3, 1/2, 0, 1/2,
+   1/3, 1/2, 0,   0), nrow = 4, ncol = 4, byrow = T)
> colnames(P) <- c("A", "B", "C", "D")
> rownames(P) <- c("A", "B", "C", "D")
> P
        A   B C   D
A 0.0000 0.0 1 0.5
B 0.3333 0.0 0 0.0
C 0.3333 0.5 0 0.5
D 0.3333 0.5 0 0.0

> ei <- eigen(P)
> PR <- as.numeric(ei$vectors[,1] / sum(ei$vectors[,1]))
> sum(PR)
[1] 1
> names(PR) <- c("A", "B", "C", "D")
> sort(PR, decreasing = TRUE)
     A      C      D      B
0.3871 0.2903 0.1935 0.1290
>
```

```
> graph <- graph.formula(
+   A-+B,
+   A-+C,
+   A-+D,
+   B-+C,
+   B-+D,
+   C-+A,
+   D-+A,
+   D-+C
+ )
> pr <- page_rank(graph)
> sort(pr$vector, dec = TRUE)
     A      C      D      B
0.3682 0.2880 0.2021 0.1418
> sum(pr$vector)
[1] 1
```

Liczbowo wynik jest co prawda inny niż zwracany przez igraph, niemniej jednak kolejność ważności węzłów jest dokładnie zachowana

PageRank

- Algorytm algebraiczny
  - przykład, gdy mamy węzeł typu „sink” i wówczas algorytm w tej podstawowej wersji nie zadziała

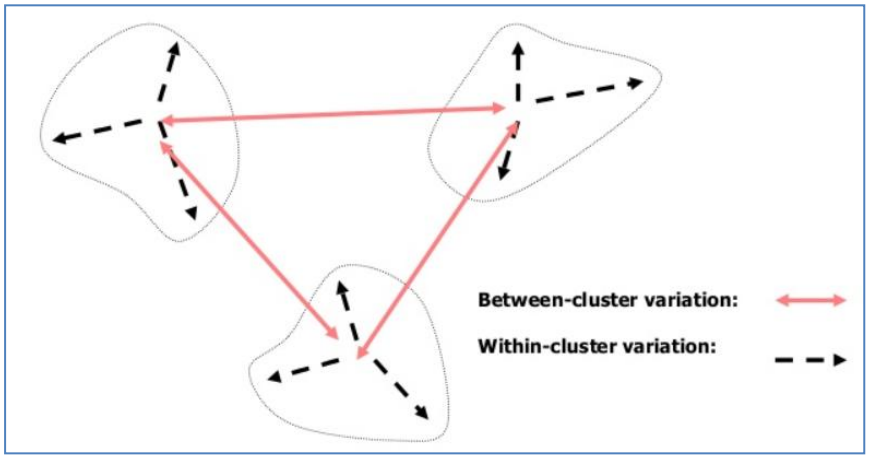



```
> P
     [,1] [,2] [,3]
[1,]    0    0    0
[2,]    0    0    0
[3,]    1    1    0
> ei <- eigen(P)
> PR <- as.numeric(ei$vectors[,1] / sum(ei$vectors[,1]))
> names(PR) <- c("A", "B", "C")
> sort(PR, decreasing = TRUE)
C A B
1 0 0

> # igraph
> graph <- graph.formula(
+   A-+C,
+   B-+C
+ )
> pr <- page_rank(graph)
> pr$vector
     A      C      B
0.2128 0.5745 0.2128
```

Grupowanie oraz klasyfikacja zasobów tekstowych

Grupowanie, klasyfikacja – przypomnienie

- Grupowanie (ang. clustering) oznacza łączenie obiektów w grupy, które są do siebie podobne a jednocześnie niepodobne do obiektów z innych grup

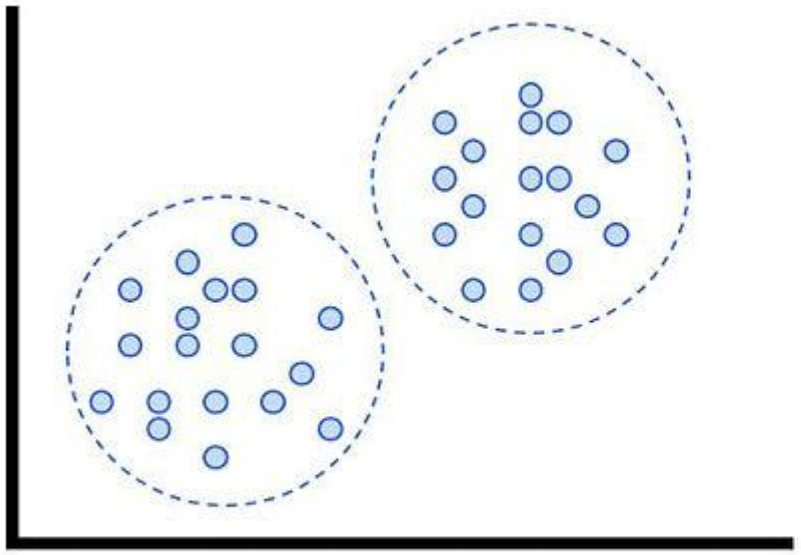


BCV (between-cluster variation, zmienność między grupami)

WCV (within-cluster variation, zmienność wewnątrz grupy)

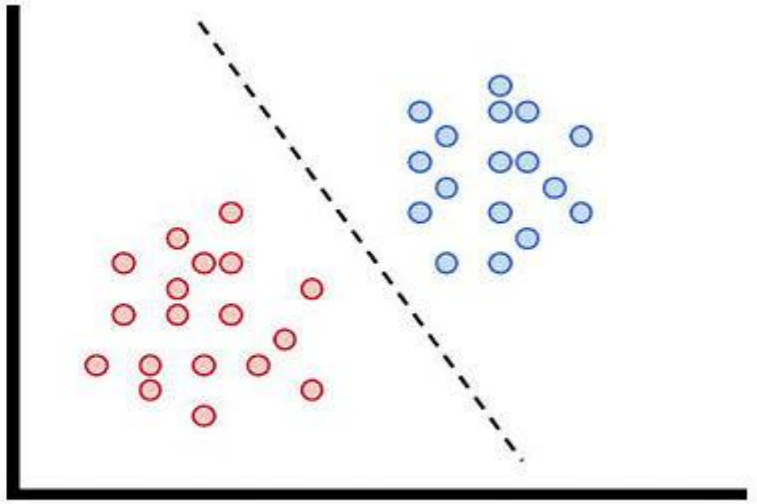
- Podobnym pojęciem jest klasyfikacja, tam jednak mamy tzw. zmienną celu. Najpierw, na bazie danych uczących tworzony jest klasyfikator. Następnie nowe obiekty są klasyfikowane (przypisywane) do odpowiedniej klasy

Grupowanie a klasyfikacja – przypomnienie



Clustering

- dane nie mają przypisanych etykiet
- obiekty leżące „blisko siebie” są łączone w grupy
- naturalne wykrywanie wzorców w danych
- uczenie nienadzorowane

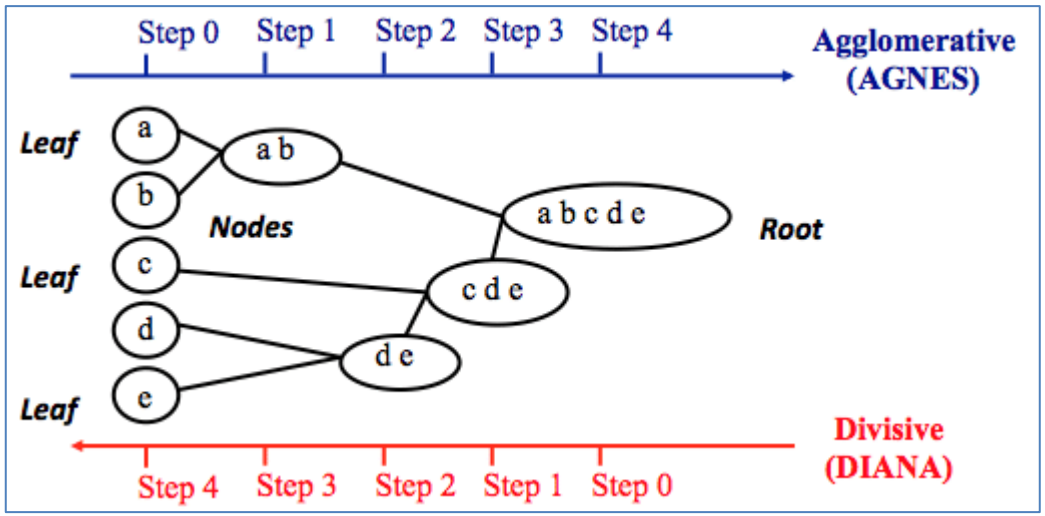


Classification

- dane mają przypisane etykiety
- poszukujemy pewnej reguły przypisywania nowych elementów do już istniejących klas
- uczenie nadzorowane

Grupowanie – przypomnienie

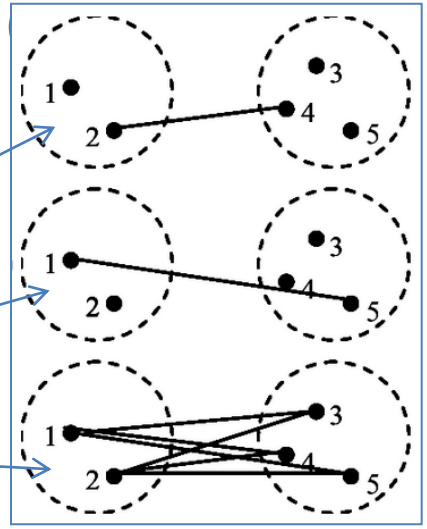
- Hierarchiczne (powstaje struktura drzewiasta, dendrogram)
  - aglomeracyjne. Początkowo zakłada się, że każdy obiekt tworzy oddzielną grupę, w kolejnych krokach grupy, które są najbardziej podobne do siebie łączy się w coraz to większe grupy. Na końcu wszystkie obiekty znajdują się w jednej grupie
    - ✓ ta metoda jest w praktyce zdecydowanie częściej stosowana
  - rozdzielające (początkowo wszystkie obiektu tworzą jedną grupę. W kolejnych krokach obiekty najbardziej niepodobne są rozdzielane na oddzielne grupy. Na końcu każdy obiekt tworzy jednoelementową grupę

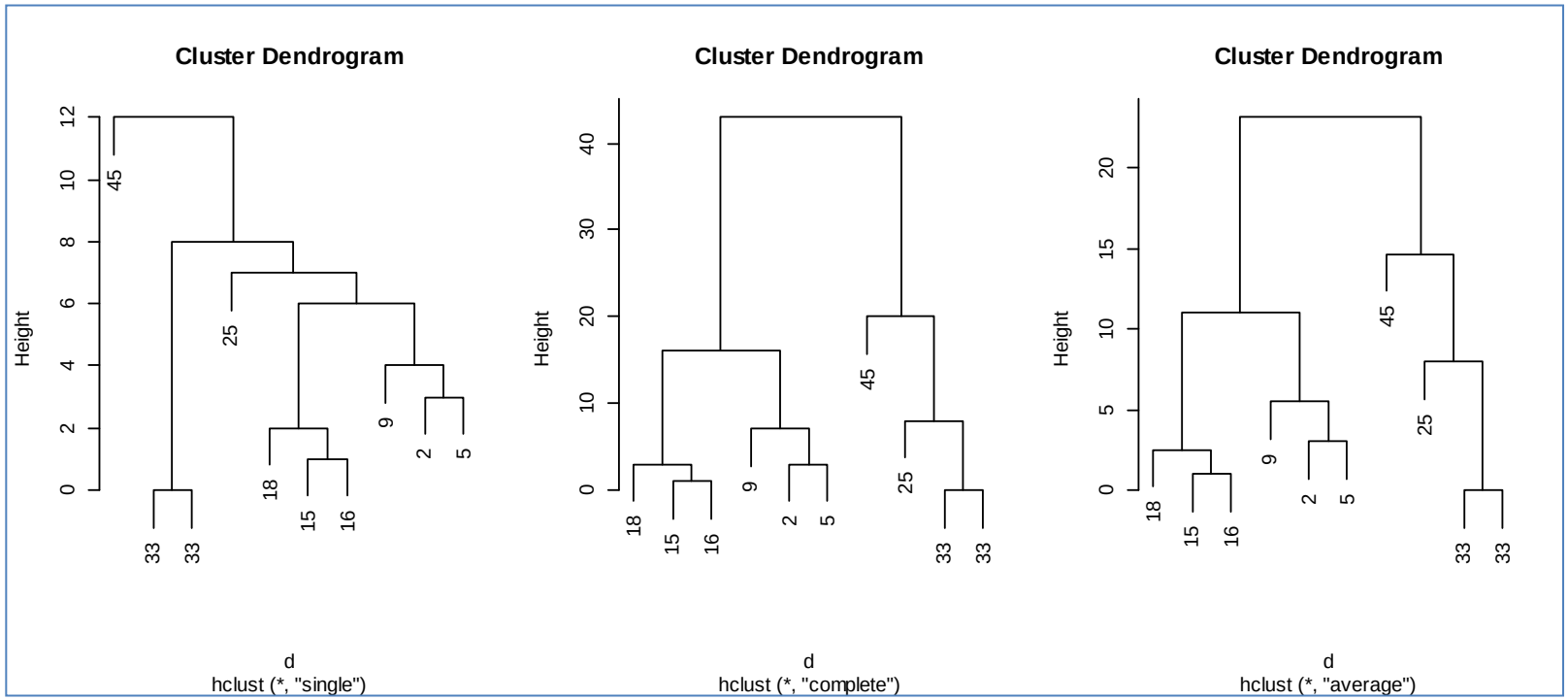


Grupowanie – przypomnienie

- Hierarchiczne, aglomeracyjne, przykłady w R

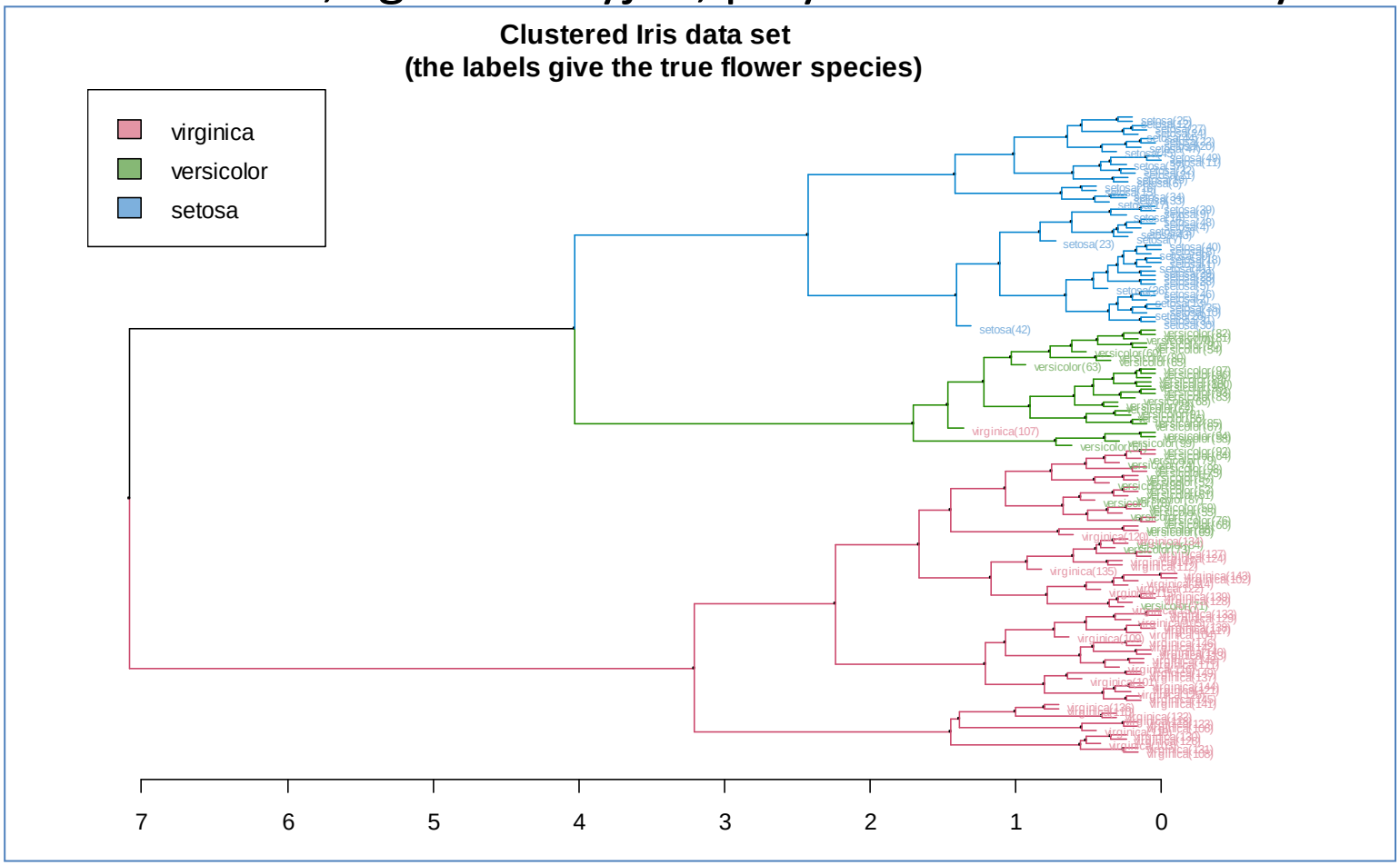
```
data <- c(2, 5, 9, 15, 16, 18, 25, 33, 33, 45)
d <- dist(as.matrix(data))
hc <- hclust(d, method = "single")
hc <- hclust(d, method = "complete")
hc <- hclust(d, method = "average")
plot(hc, labels = data)
```





Grupowanie – przypomnienie

- Hierarchiczne, aglomeracyjne, przykład dla zbioru danych iris



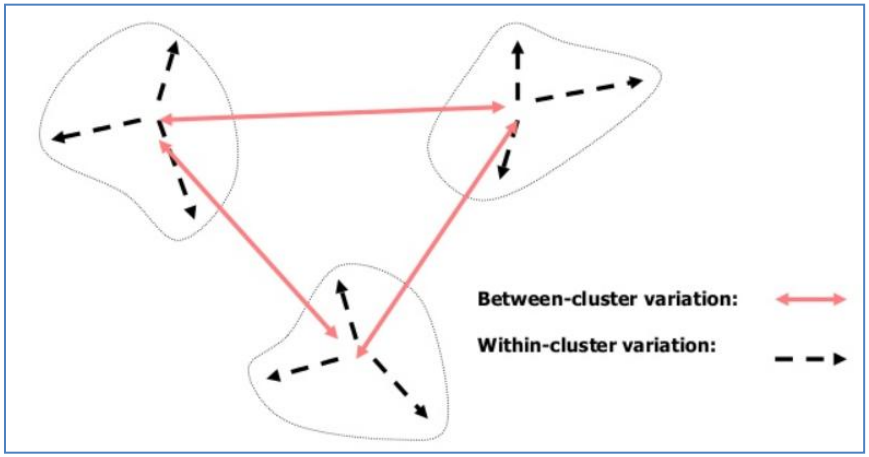
źródło: https://cran.r-project.org/web/packages/dendextend/vignettes/Cluster_Analysis.html

Grupowanie – przypomnienie

- Algorytm k-menas
  - jeden z najbardziej popularnych algorytmów grupowania
  - prosty i efektywny w działaniu
  - wymaga założenia na wstępie ilości grup. Analityk powinien mieć jakąś wiedzę *a priori* o liczbie ukrytych grup. Czasami trzeba poeksperymentować
  - algorytm nie gwarantuje znalezienia najbardziej optymalnego rozwiązania (często zatrzymuje się w minimum lokalnym)
  - zwykle należy kilkakrotnie wykonać obliczenia i wybrać najlepszy zdaniem analityka wynik
    - ✓ wybór początkowych środków grup odbywa się zwykle losowo

Grupowanie – przypomnienie

- Miary BCV, WCV



BCV (between-cluster variation, zmienność między grupami)

WCV (within-cluster variation, zmienność wewnątrz grupy)

$$BCV(C_k) = \sum_{x_i \in C_k} (x_i - \mu_k)^2$$

$$WCV = \sum_{k=1}^{k} BCV(C_k) = \sum_{k=1}^{k} \sum_{x_i \in C_k} (x_i - \mu_k)^2$$

x_i – punkt należący do klastra C_k

μ_k – średnia punktów w klastrze C_k

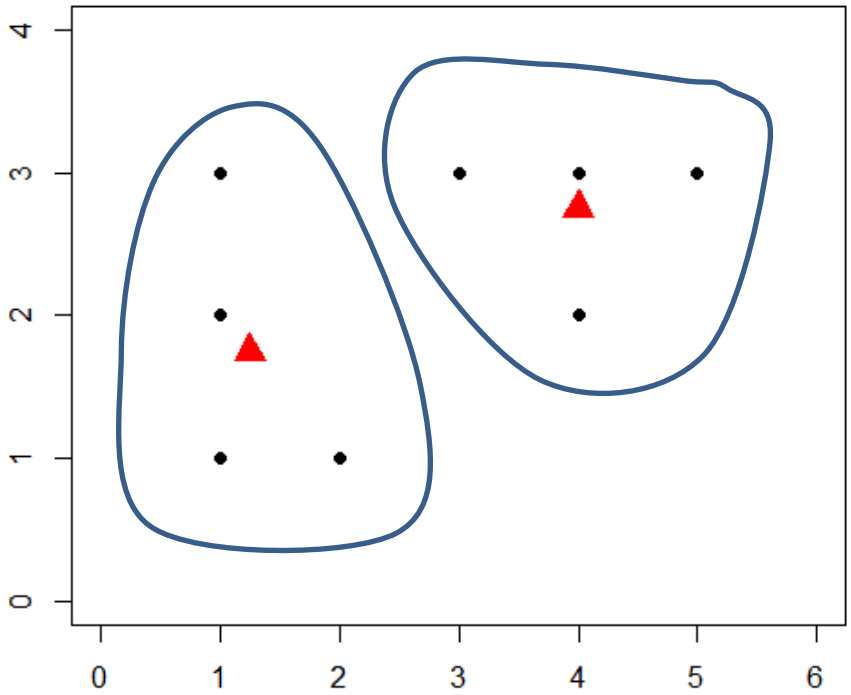
$\frac{BCV(C_k)}{WCV}$ – pożądana maksymalizacja

Grupowanie – przypomnienie

- K-means, przykład w R

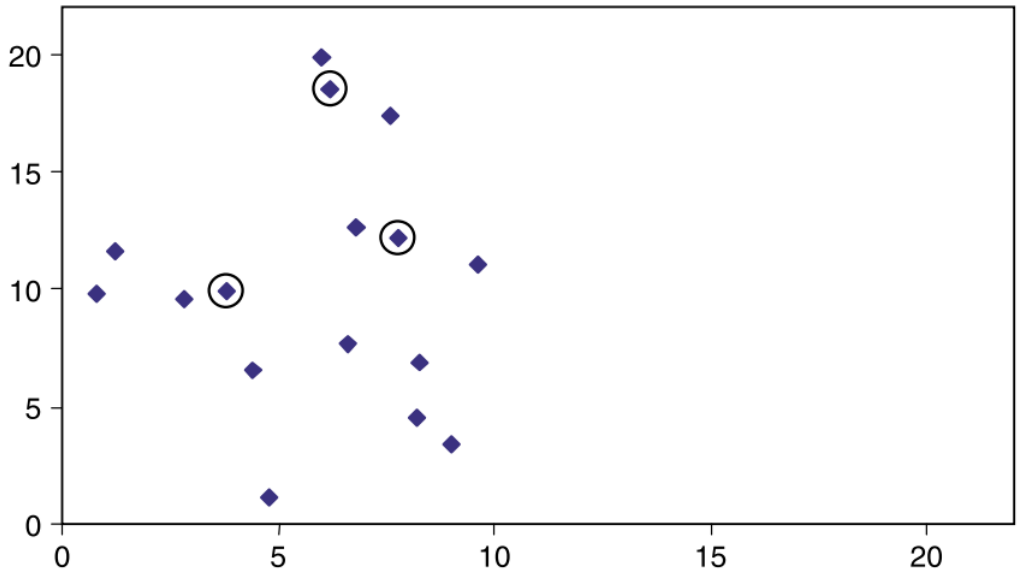
```
data <-
matrix(c(1,3,3,3,4,3,5,3,1,2,4,2,1,1,2,1),
       nrow = 8, ncol = 2, byrow = TRUE)

     [,1] [,2]
[1,]    1    3
[2,]    3    3
[3,]    4    3
[4,]    5    3
[5,]    1    2
[6,]    4    2
[7,]    1    1
[8,]    2    1
km <- kmeans(data, centers = 2)
plot(data,
     pch = 16,
     xlim = c(0,6),
     ylim = c(0,4))

points(km$centers,
       pch = 17,
       cex = 2,
       col = "red")
```

```
K-means clustering with 2 clusters of sizes 4, 4

Cluster means:
   [,1]  [,2]
1 1.25 1.75
2 4.00 2.75

Clustering vector:
[1] 1 2 2 2 1 2 1 1

Within cluster sum of squares by cluster:
[1] 3.50 2.75
 (between_SS / total_SS =  73.3 %)
```



Grupowanie – przypomnienie

- Przykład obliczeniowy, k-means
  - zaczerpnięto z książki:
    Max Bramer, Principles of Data Mining (4th Edition), Springer, 2020

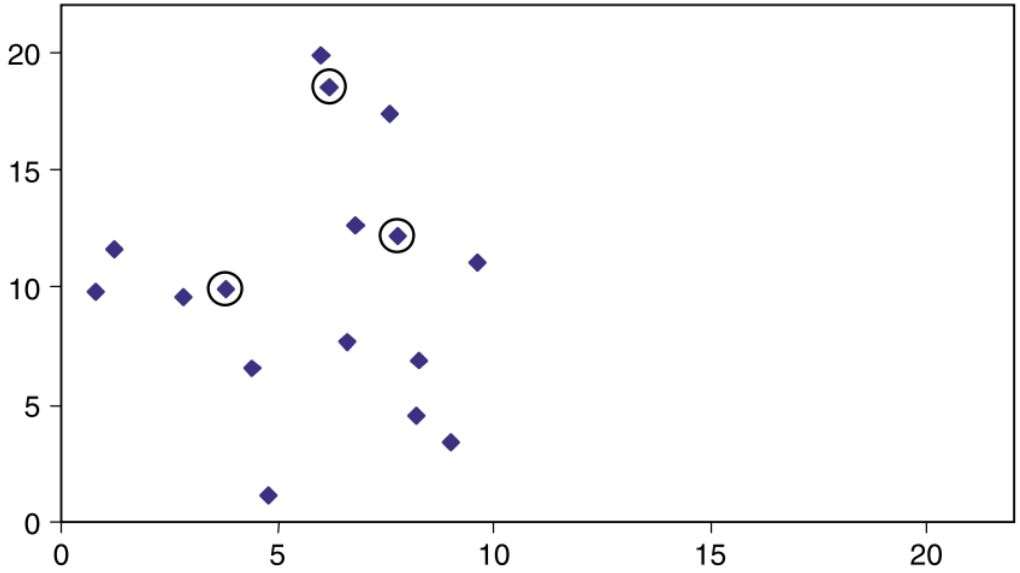


x	y
6.8	12.6
0.8	9.8
1.2	11.6
2.8	9.6
3.8	9.9
4.4	6.5
4.8	1.1
6.0	19.9
6.2	18.5
7.6	17.4
7.8	12.2
6.6	7.7
8.2	4.5
8.4	6.9
9.0	3.4
9.6	11.1

Grupowanie – przypomnienie

- K-menas, krok 1: wybór (arbitralny / losowy)
  1. ilości grup (wybraliśmy k = 3)
  2. startowych centroidów (wybraliśmy 3 wiersze z naszego zbioru)

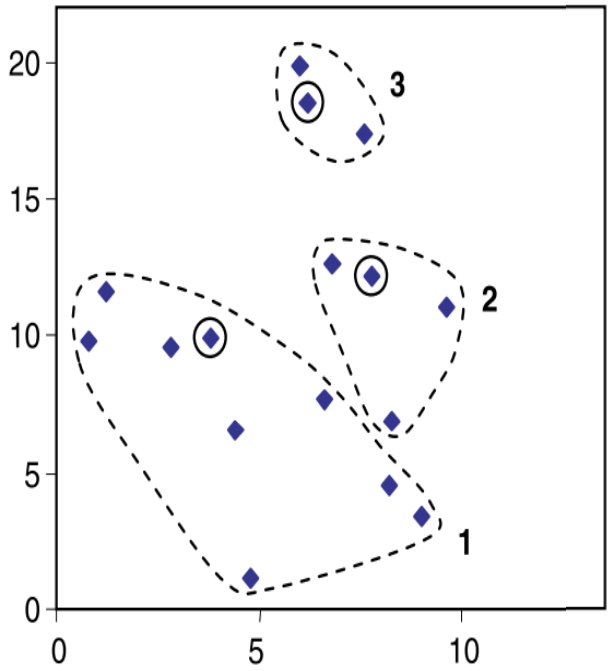
	Initial	
	x	y
Centroid 1	3.8	9.9
Centroid 2	7.8	12.2
Centroid 3	6.2	18.5



Grupowanie – przypomnienie

- K-menas, krok 2: wyliczenie odległości każdego punktu od centroidu startowego i przypisanie go do właściwego klastra

x	y	$d1$	$d2$	$d3$	cluster
6.8	12.6	4.0	1.1	5.9	2
0.8	9.8	3.0	7.4	10.2	1
1.2	11.6	3.1	6.6	8.5	1
2.8	9.6	1.0	5.6	9.5	1
3.8	9.9	0.0	4.6	8.9	1
4.4	6.5	3.5	6.6	12.1	1
4.8	1.1	8.9	11.5	17.5	1
6.0	19.9	10.2	7.9	1.4	3
6.2	18.5	8.9	6.5	0.0	3
7.6	17.4	8.4	5.2	1.8	3
7.8	12.2	4.6	0.0	6.5	2
6.6	7.7	3.6	4.7	10.8	1
8.2	4.5	7.0	7.7	14.1	1
8.4	6.9	5.5	5.3	11.8	2
9.0	3.4	8.3	8.9	15.4	1
9.6	11.1	5.9	2.1	8.1	2



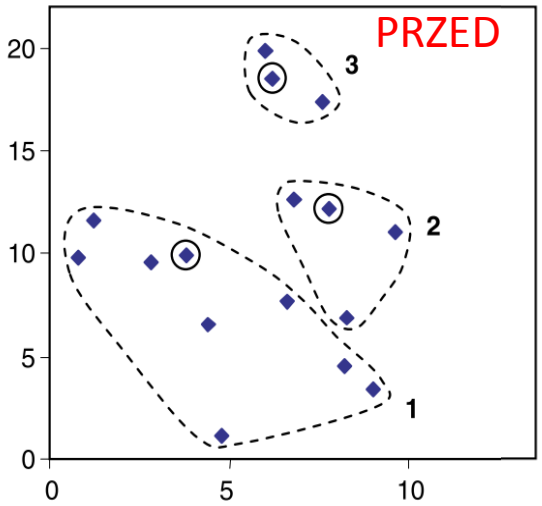
$$\sqrt{(6.8-3.8)^2+(12.6-9.9)^2}=4.0$$

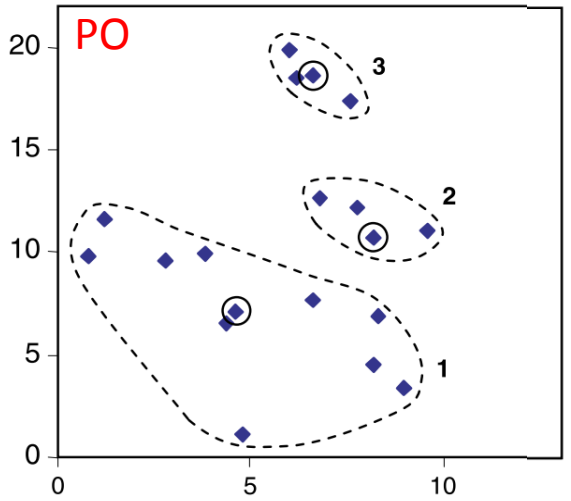
Grupowanie – przypomnienie

- K-menas, krok 3: wyznaczenie nowych centroidów i ponowne przyporządkowanie danych do klastrów
  - nowe centroidy wyznaczamy jako średnia wartość x oraz y wszystkich punktów przypisanych do danego klastra

	Initial		After first iteration	
	x	y	x	y
Centroid 1	3.8	9.9	4.6	7.1
Centroid 2	7.8	12.2	8.2	10.7
Centroid 3	6.2	18.5	6.6	18.6

nowe centroidy są teraz w innych nieco miejscach





Grupowanie – przypomnienie

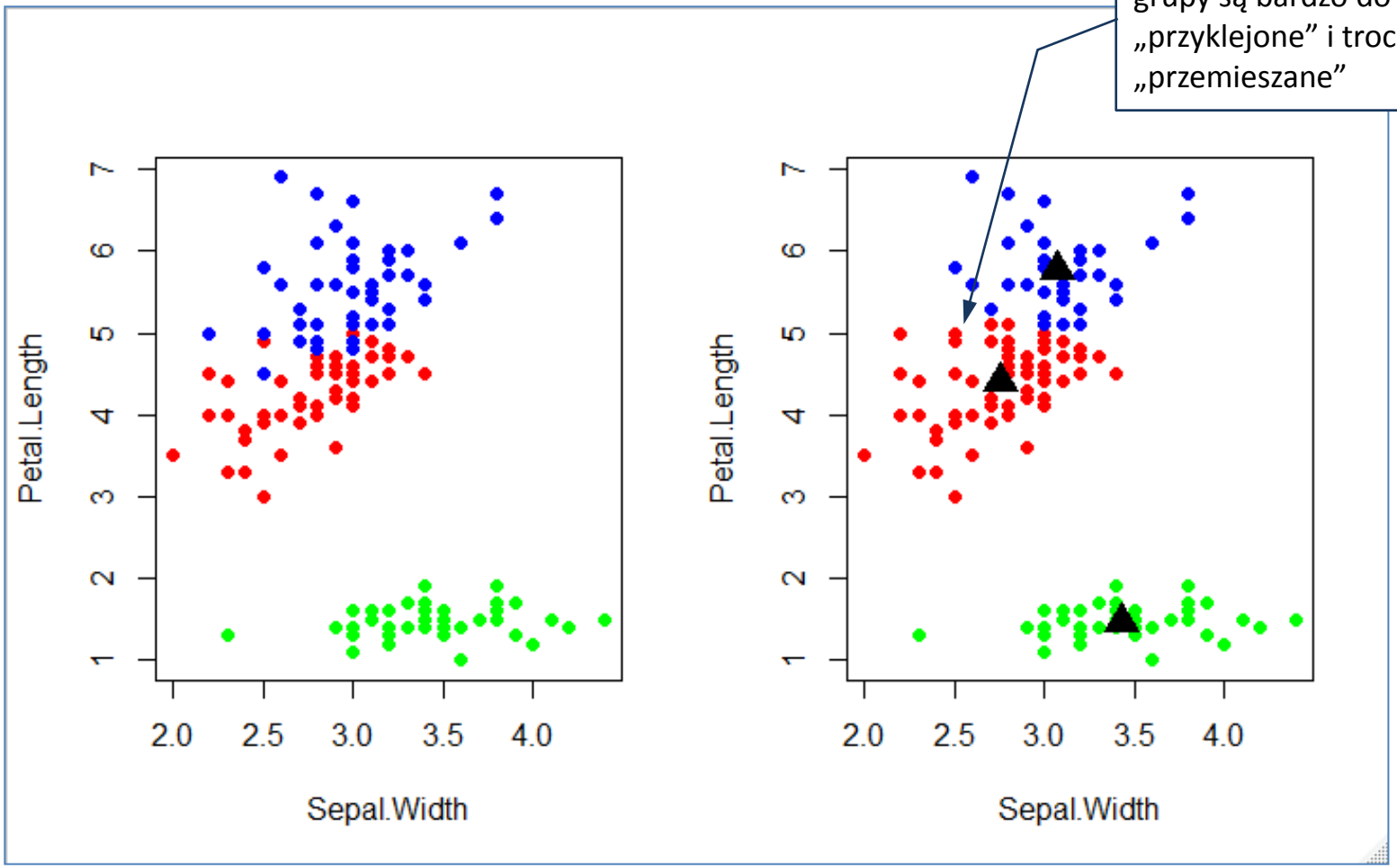
- K-menas, krok 4: ponowne wyznaczenie nowych pozycji centroidów i ponowne przyporządkowanie danych do klastrów

	Initial		After first iteration		After second iteration	
	x	y	x	y	x	y
Centroid 1	3.8	9.9	4.6	7.1	5.0	7.1
Centroid 2	7.8	12.2	8.2	10.7	8.1	12.0
Centroid 3	6.2	18.5	6.6	18.6	6.6	18.6

- K-menas, kolejne kroki: powtarzamy aż do osiągnięcia stabilizacji
  - czyli, gdy w kolejnych krokach nie zmienia się już przypisanie danych do klastrów

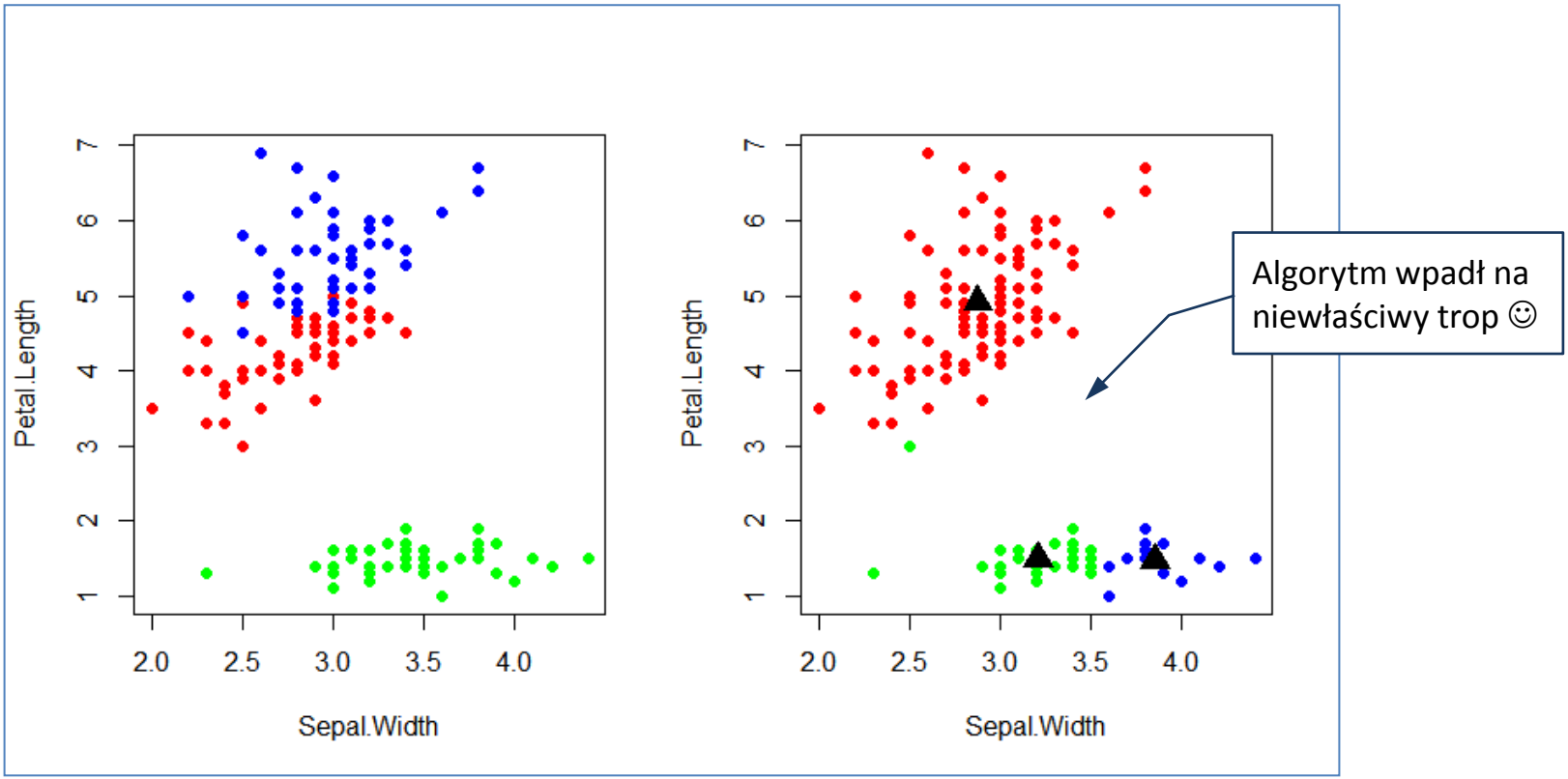
Grupowanie – przypomnienie

- k-means, przykład dla zbioru danych iris



Grupowanie – przypomnienie

- k-means, przykład dla zbioru danych iris
  - algorytm ma pewien element losowości i czasami może się pojawić poważne przekłamanie. Nic na to nie poradzimy. Trzeba mieć świadomość tego



Grupowanie – przypomnienie

- k-means, wady
  - ilość grup k musimy znać a-priori
  - wynik końcowy bardzo mocno zależy od tego, gdzie będą ustawione inicjujące środki grup. Nie ma żadnych "twardych" zleceń, jak te środki ustalać. Zwykle więc są one wybierane całkowicie losowo. Wynik na poprzednich slajdach (dla zbioru iris) pokazuje, jak różne możemy otrzymywać wyniki
    - ✓ remedium: uruchomić k-means wielokrotnie i wybrać rozwiązanie z największą wartością współczynnika BCV/WCV
  - algorytm jest bardzo czuły na obecność punktów odstających (outliers), które mogą bardzo zafałszować wynik końcowy
    - ✓ jeżeli możliwe, pozbyć się punktów odstających
  - przeorganizowanie danych (zamiana kolejności rekordów) może również bardzo zmienić wynik
    - ✓ brak sensownego remedium, ew. wielokrotne uruchomienie k-means

Grupowanie – przypomnienie

- Inne, bardziej „smart” algorytmy. Jest tego całkiem sporo. Te najbardziej popularne to:
  - k-medoids (inna nazwa: partitioning around medoids; PAM)
    - ✓ działanie tego algorytmu zbliżone jest do działania algorytmu centroidów (k-means), z tą różnicą, że centroidy zostają tu zastąpione przez medoidy, czyli najbardziej centralne obiekty ze zbioru danych reprezentujące daną grupę, dla których odległość od wszystkich pozostałych elementów wewnątrz danej grupy jest minimalna.
    - ✓ zaleta: odporność na obserwacje odstające (ang. *outliers*) oraz szumy występujące w danych (ang. *robustness*)
    - ✓ wada: brak możliwości zastosowania tej metody dla dużych zbiorów danych (powyżej 65536 obserwacji)
  - CLARA (Clustering Large Applications), jest to de facto rozszerzenie k-medoids
    - ✓ można używać dla dużych zbiorów danych
    - ✓ działa na zasadzie grupowania próbki z zestawu danych, a następnie przypisuje wszystkie obiekty w zestawie danych do tych klastrów

Grupowanie – przypomnienie

- Przykład obliczeniowy, metoda hierarchiczna aglomeracyjna, single-link clustering
  - zaczerpnięto z książki:
    Max Bramer, Principles of Data Mining (4th Edition), Springer, 2020
  - dane (obiekty a, b, c, d,e, f) przedstawione w postaci macierzy odległości (macierz jest oczywiście symetryczna). Liczby w przykładowej macierzy to mogą być np. odległości wg. miary Euklidesowej pomiędzy punktami a, b, c, d, e, f w przestrzeni 2D

	a	b	c	d	e	f
a	0	12	6	3	25	4
b	12	0	19	8	14	15
c	6	19	0	12	5	18
d	3	8	12	0	11	9
e	25	14	5	11	0	7
f	4	15	18	9	7	0

Grupowanie – przypomnienie

- Grupowanie hierarchiczne, krok 1
  - z macierzy odległości widać, że najbliżej są obiekty a oraz d, więc łączymy je w grupę (patrz czerwone kółeczko na poprzednim slajdzie)

	ad	b	c	e	f
ad	0	?	?	?	?
b	?	0	19	14	15
c	?	19	0	5	18
e	?	14	5	0	7
f	?	15	18	7	0

  - teraz należy policzyć odległości pomiędzy ad oraz b, c, e, f . Wybierzmy metodę single-link clustering (patrz wcześniejsze slajdy). Otrzymujemy więc

	ad	b	c	e	f
ad	0	8	6	11	4
b	8	0	19	14	15
c	6	19	0	5	18
e	11	14	5	0	7
f	4	15	18	7	0

Grupowanie – przypomnienie

- Grupowanie hierarchiczne, krok 2
  - teraz najmniejszą wartością w macierzy jest 4, czyli odległość pomiędzy klastrem ad a klastrem f . Dokonujemy więc kolejnego połączenia

	adf	*b*	*c*	*e*
adf	0	8	6	7
b	8	0	19	14
c	6	19	0	5
e	7	14	5	0

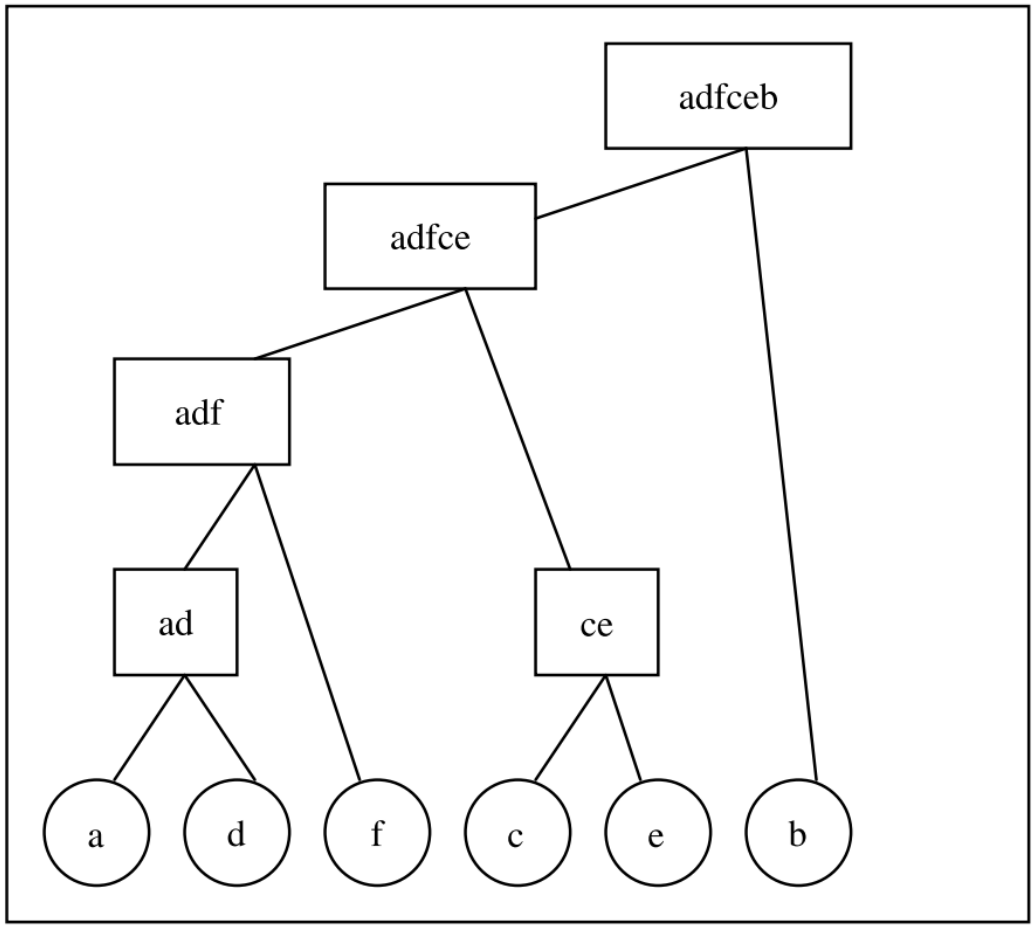
- Grupowanie hierarchiczne, kolejne kroki analogicznie
  - do momentu aż wszystkie obiekty skupią się w jeden klaster

	adf	*b*	*ce*
adf	0	8	6
b	8	0	14
ce	6	14	0

	adfce	*b*
adfce	0	8
b	8	0

Grupowanie – przypomnienie

- Wynik końcowy
  - powstaje dendrogram



Grupowanie dokumentów tekstowych

- Grupowanie opisów fabuły 17. filmów animowanych. Opisy pobrano w Wikipedii (stan na dzień 14.08.2020)

Despicable Me / Jak ukraść księżyc
Despicable Me 2 / Minionki rozrabiają
Finding Dory / Gdzie jest Dory?
Finding Nemo / Gdzie jest Nemo?
Frozen / Kraina lodu
Ice Age / Epoka lodowcowa
Ice Age: Dawn of the Dinosaurs / Epoka lodowcowa 3: Era dinozaurów
Kung Fu Panda / Kung Fu Panda
Kung Fu Panda 2 / Kung Fu Panda 2
Kung Fu Panda 3 / Kung Fu Panda 3
Madagascar / Madagascar
Madagascar 2: Escape to Africa / Madagascar 2
Minions / Minionki
Shrek / Shrek
Shrek 2 / Shrek 2
Shrek the Third / Shrek Trzeci
Shrek Forever After / Shrek Forever

Opis fabuły [edytuj | edytuj kod]

Ogr o imieniu Shrek chce za wszelką cenę odzyskać spokój na terenie swojej posiadłości na bagnach, gdzie w wyniku represji okrutnego Lorda Farquaada zesłane zostały różne bajkowe postacie, w tym Pinokio, Wilk czy siedmiu krasnoludków. Shrek decyduje się na wyprawę do siedziby Farquaada, miasta Duloc,

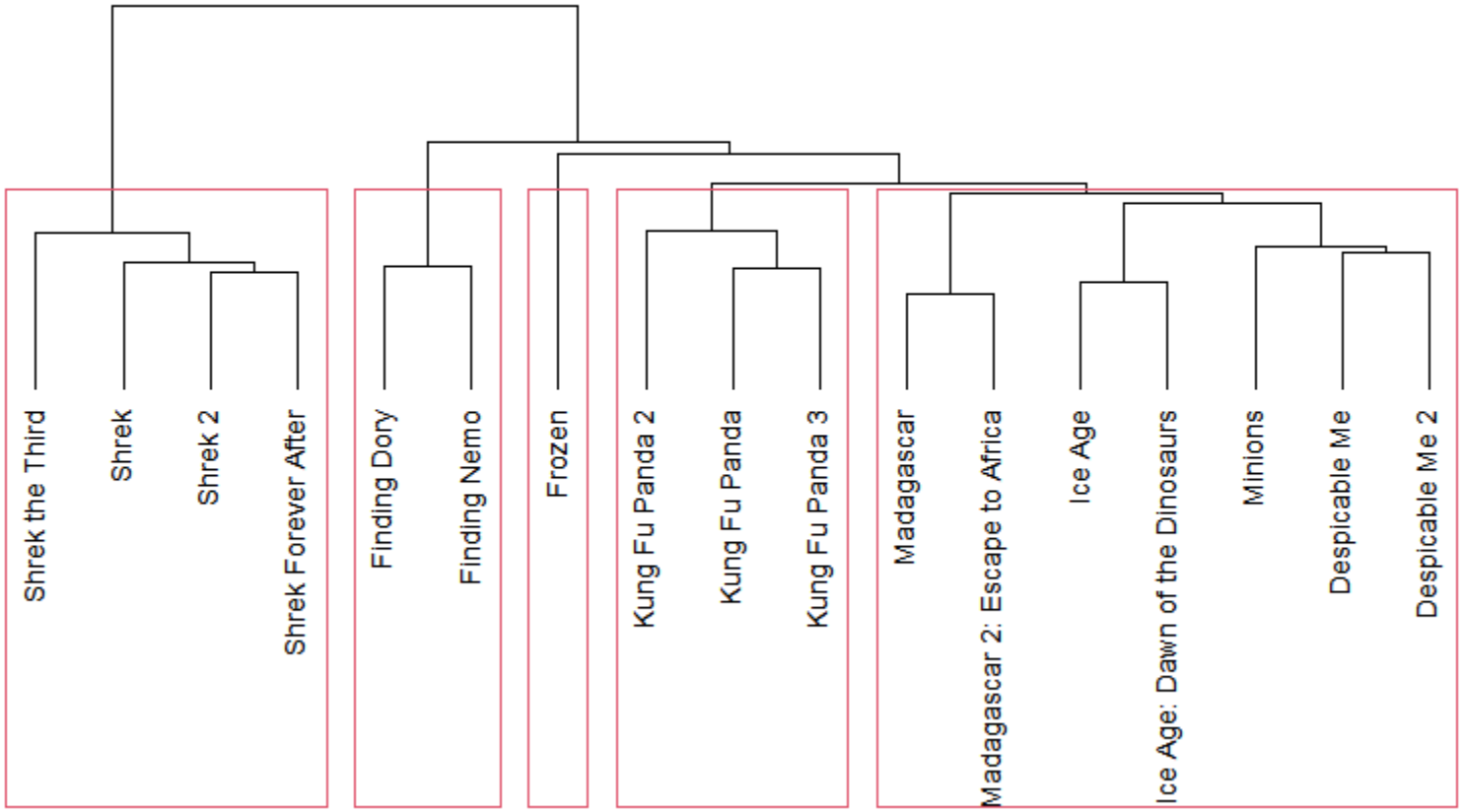
Plot

Shrek is an anti-social and highly-territorial green ogre who loves the solitude of his swamp. His life is interrupted after the vertically-challenged king of Duloc, Lord Farquaad, exiles a countless number of fairytale creatures to Shrek's swamp.

źródło: https://rpubs.com/saqib/DocumentClustering

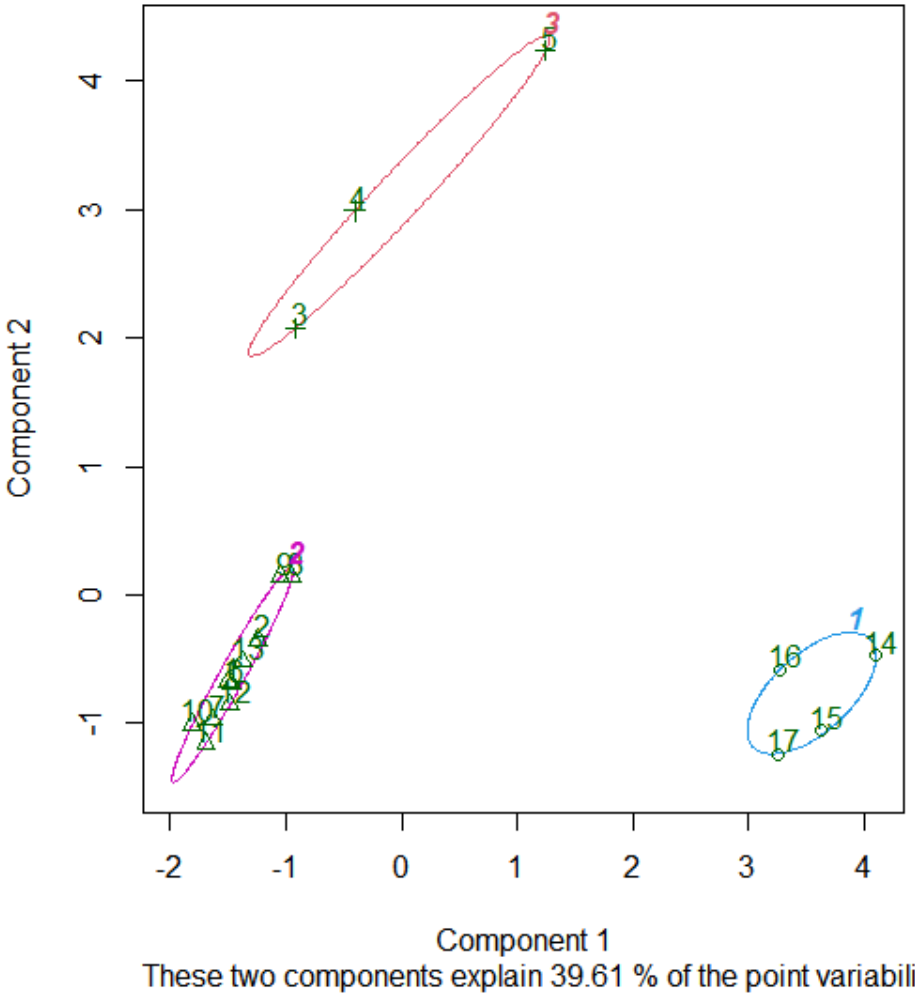
Grupowanie dokumentów tekstowych

- Wynik dla opisów w języku angielskim
  - grupowanie hierarchiczne (*hclust*, metoda *complete*)



Grupowanie dokumentów tekstowych

- Wynik dla opisów w języku angielskim
  - grupowanie k-means (*kmeans(dist.matrix, centers = 3, nstart = 100)*)



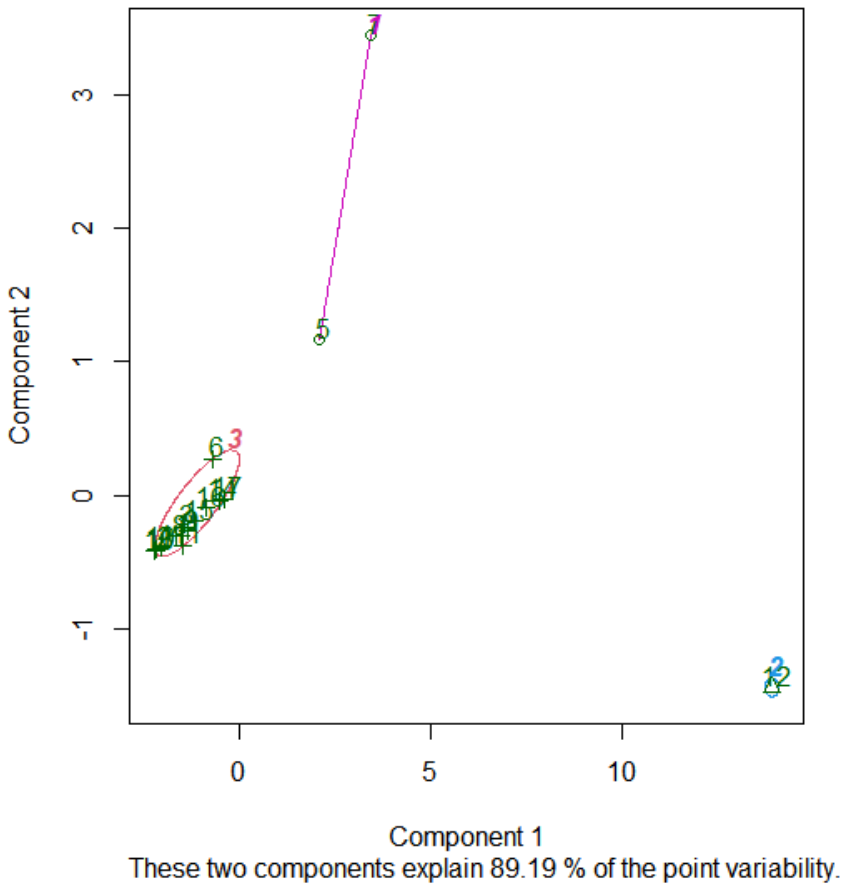
[1,] "Despicable Me"
[2,] "Despicable Me 2"
[3,] "Finding Dory"
[4,] "Finding Nemo"
[5,] "Frozen"
[6,] "Ice Age"
[7,] "Ice Age: Dawn of the Dinosaurs"
[8,] "Kung Fu Panda"
[9,] "Kung Fu Panda 2"
[10,] "Kung Fu Panda 3"
[11,] "Madagascar"
[12,] "Madagascar 2: Escape to Africa"
[13,] "Minions"
[14,] "Shrek"
[15,] "Shrek 2"
[16,] "Shrek the Third"
[17,] "Shrek Forever After"

Grupowanie dokumentów tekstowych

- Wynik dla opisów w języku polskim
  - grupowanie hierarchiczne (*hclust*, metoda *ward.D*)

Grupowanie dokumentów tekstowych

- Wynik dla opisów w języku Polskim
  - grupowanie k-means (*kmeans(dist.matrix, centers = 3, nstart = 100*)



[1,] "Jak ukraść księżyc"
[2,] "Minionki rozrabiają"
[3,] "Gdzie jest Dory?"
[4,] "Gdzie jest Nemo"
[5,] "Kraina lodu"
[6,] "Epoka lodowcowa"
[7,] "Epoka lodowcowa 3: Era dinozaurów"
[8,] "Kung Fu Panda"
[9,] "Kung Fu Panda 2"
[10,] "Kung Fu Panda 3"
[11,] "Madagascar"
[12,] "Madagascar 2"
[13,] "Minionki"
[14,] "Shrek"
[15,] "Shrek 2"
[16,] "Shrek Trzeci"
[17,] "Shrek Forever"
> kfit$cluster

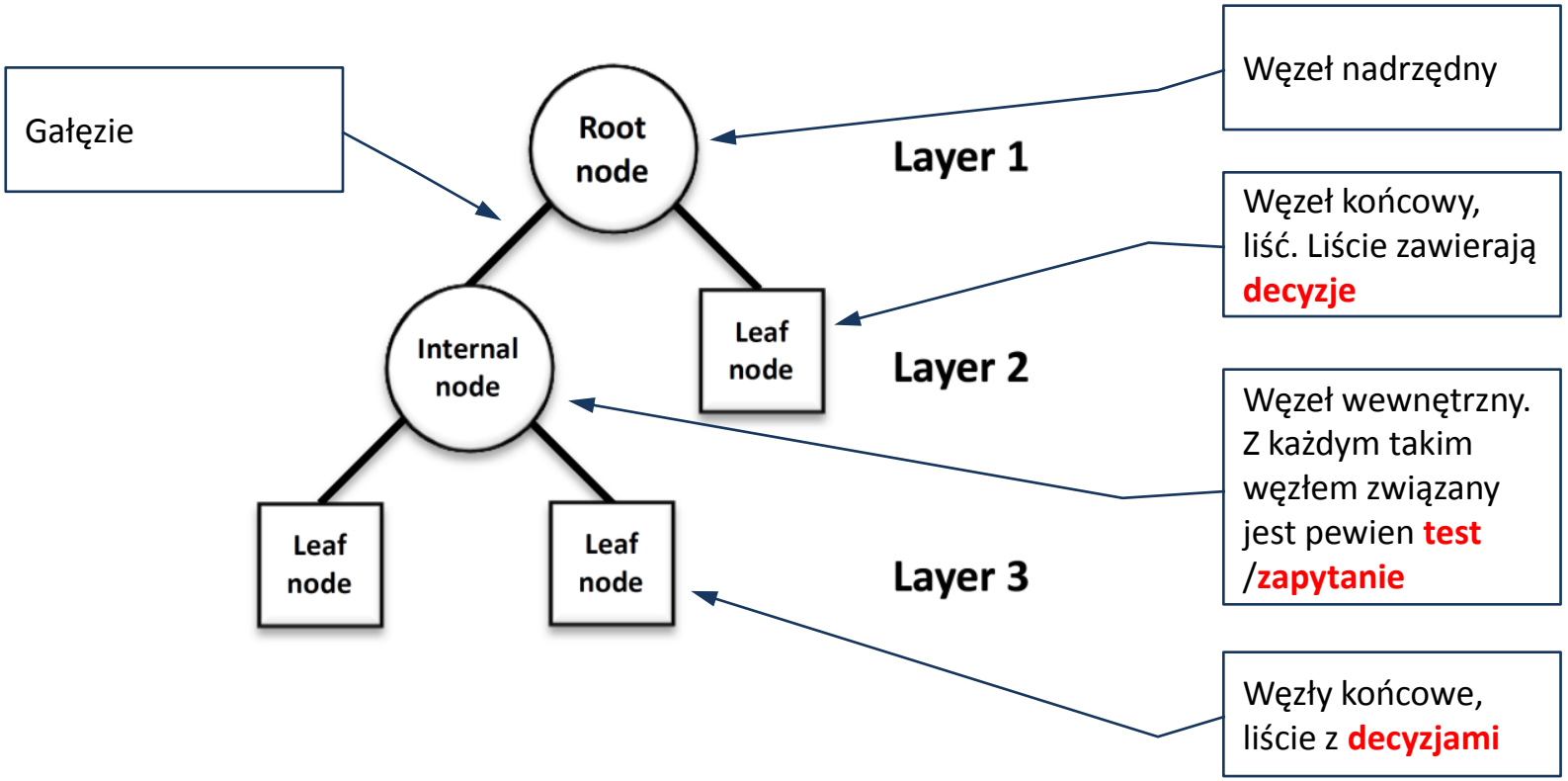
```
kfit$cluster
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17
 3  3  3  3  1  3  1  3  3  3  3  2  3  3  3  3  3
```

Klasyfikacja

- Pokażemy tylko jeden rodzaj klasyfikatora: drzewo klasyfikacyjne (decyzyjne). Wynik zaprezentowany jest w bardzo łatwej i intuicyjnej postaci graficznej
  - jest też naprawdę sporo innych technik, np. LDA (Linear Discriminat Analysis), SVM (Support Vector Machine), RF (Random Forests), NB (Naive Bayes), KNN (K-Nearest Neighbors), RPART (Recursive Partitioning and Regression Trees), KDA (Kernel Discriminant Analysis), ID3, C4.5, CART i wiele innych
- Etapy budowy klasyfikatora
  - uczenie, dane treningowe
  - testowanie, dane testowe
  - właściwa klasyfikacja nowych danych, które nie brały udziału w uczeniu i testowaniu klasyfikatora

Klasyfikacja

- Terminologia stosowana w drzewach decyzyjnych



Klasyfikacja

- *golf*, przykładowy zbiór danych

Zmienne opisujące poszczególne przypadki , predyktory (predictor variables)

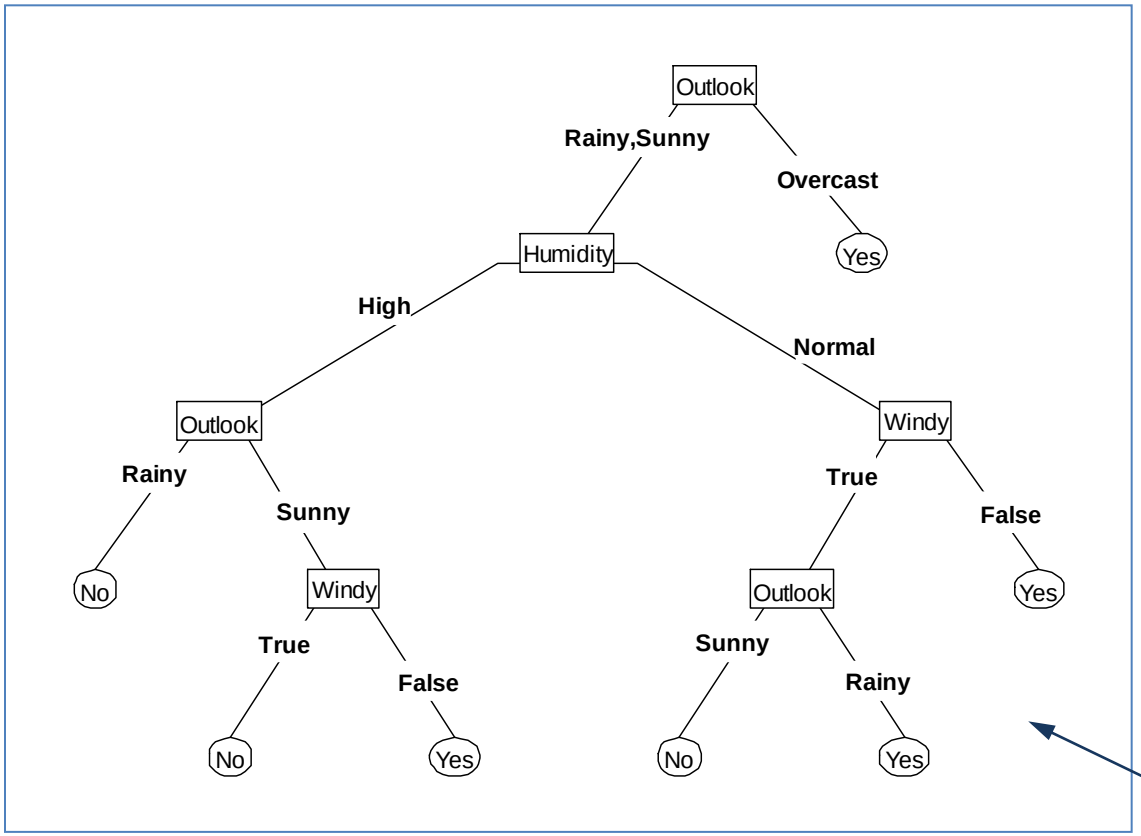
Zmienna odpowiedzi (response variable), zmienna celu (target variable)

	Outlook	Temp	Humidity	Windy	PlayGolf
1	Rainy	Hot	High	False	No
2	Rainy	Hot	High	True	No
3	Overcast	Hot	High	False	Yes
4	Sunny	Mild	High	False	Yes
5	Sunny	Cool	Normal	False	Yes
6	Sunny	Cool	Normal	True	No
7	Overcast	Cool	Normal	True	Yes
8	Rainy	Mild	High	False	No
9	Rainy	Cool	Normal	False	Yes
10	Sunny	Mild	Normal	False	Yes
11	Rainy	Mild	Normal	True	Yes
12	Overcast	Mild	High	True	Yes
13	Overcast	Hot	Normal	False	Yes
14	Sunny	Mild	High	True	No
15	Sunny	Hot	Normal	True	???

Nowy przypadek. Jaki będzie wynik klasyfikacji?

Klasyfikacja

- Zbiór testowy *golf*, wynikowe drzewo

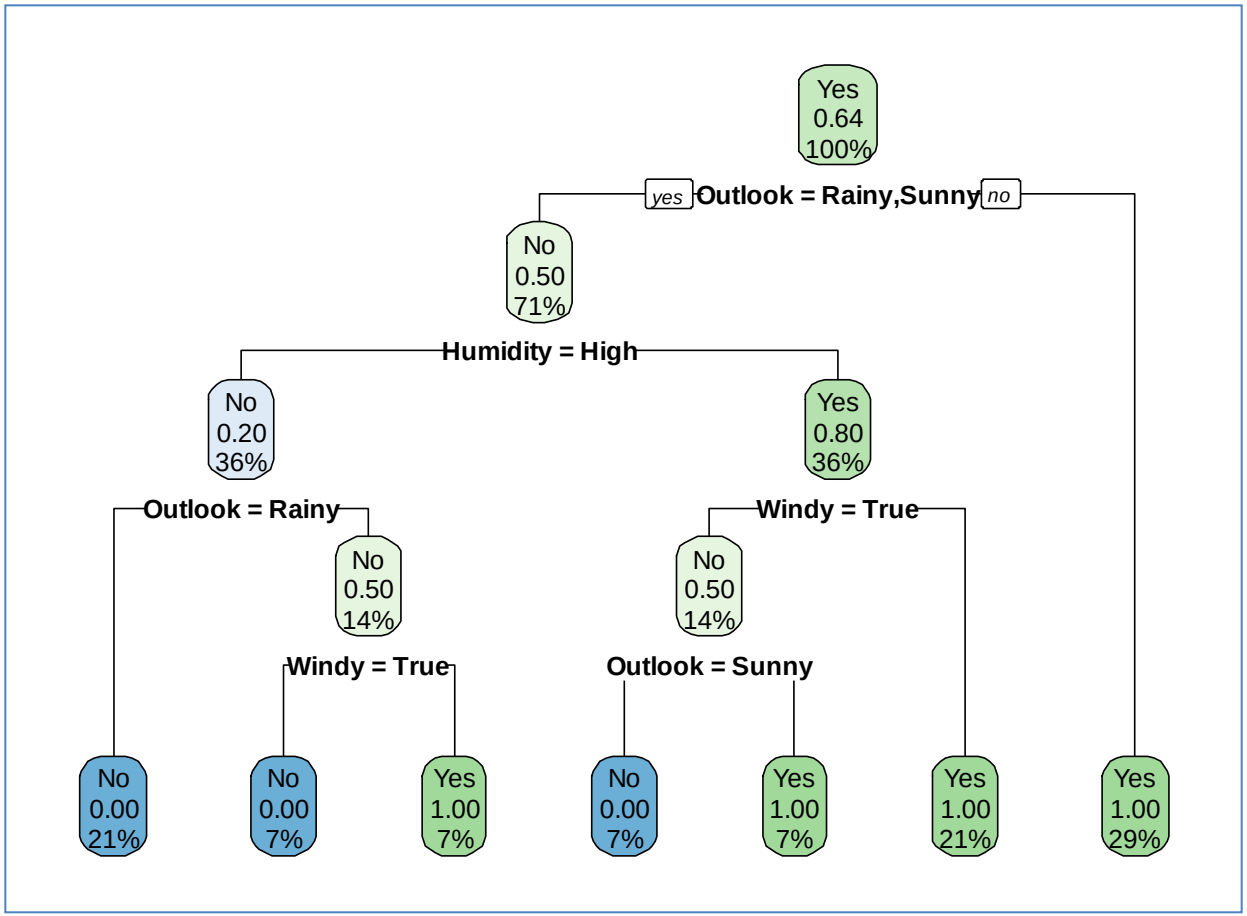


	Outlook	Temp	Humidity	Windy	PlayGolf
1	Rainy	Hot	High	False	No
2	Rainy	Hot	High	True	No
3	Overcast	Hot	High	False	Yes
4	Sunny	Mild	High	False	Yes
5	Sunny	Cool	Normal	False	Yes
6	Sunny	Cool	Normal	True	No
7	Overcast	Cool	Normal	True	Yes
8	Rainy	Mild	High	False	No
9	Rainy	Cool	Normal	False	Yes
10	Sunny	Mild	Normal	False	Yes
11	Rainy	Mild	Normal	True	Yes
12	Overcast	Mild	High	True	Yes
13	Overcast	Hot	Normal	False	Yes
14	Sunny	Mild	High	True	No

Zwróćmy uwagę, że zmienna Temp nie została tutaj użyta. Algorytm nie wybrał jej jako jedną ze zmiennych decyzyjnych

Klasyfikacja

- Zbiór testowy *golf*, wynikowe drzewo inaczej graficznie przedstawione



Klasyfikacja

- Zbiór testowy *golf*, wynikowe drzewo w postaci tekstowej

```
library(rpart)
library(rpart.plot)
golf <- read.table(file="golf.csv", header = TRUE, sep = ";", dec = ".", quote = "")
tree <- rpart(Play ~ ., data = golf, minsplit = 1, method = "class")
prp(tree, type = 5, extra = 0, varlen = 0, faclen = 0)
```

```
node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 14 5 Play (0.36 0.64)
   2) Outlook=Rainy,Sunny 10 5 Don't Play (0.50 0.50)
     4) Humidity=High 5 1 Don't Play (0.80 0.20)
       8) Outlook=Sunny 3 0 Don't Play (1.00 0.00) *
       9) Outlook=Rainy 2 1 Don't Play (0.50 0.50)
        18) Windy=True 1 0 Don't Play (1.00 0.00) *
        19) Windy=False 1 0 Play (0.00 1.00) *
     5) Humidity=Normal 5 1 Play (0.20 0.80)
      10) Windy=True 2 1 Don't Play (0.50 0.50)
        20) Outlook=Rainy 1 0 Don't Play (1.00 0.00) *
        21) Outlook=Sunny 1 0 Play (0.00 1.00) *
      11) Windy=False 3 0 Play (0.00 1.00) *
   3) Outlook=Overcast 4 0 Play (0.00 1.00) *
```

Klasyfikacja

- Bardzo ważne zagadnienie: który podział jest najlepszy?
  - dla każdego z czterech zmiennych (*Outlook*, *Temp*, *Humidity*, *Windy*) rozważamy wszystkie możliwe zmienne i podziały i wybieramy najlepszy na każdym etapie i dzielimy dane tak długo aż uznamy, że podział powinien się już zatrzymać
  - warunki stopu:
    - ✓ nie ma już danych lub
    - ✓ podział zbioru nie poprawia już wyniku
- Jak mierzyć jakość podziału zbioru danych?
  - kryterium zysku informacyjnego oraz pojęcie entropii
  - są też inne metody ale ta powyższa jest w jakimś sensie wzorcowa i dydaktycznie interesująca, warto więc ją poznać dokładniej

Klasyfikacja

- Drzewo może nadmiernie dopasować się dodanych uczących i w konsekwencji nie radzić sobie dobrze na nowych danych
- Takie zjawisko nazywamy przeuczeniem. Objawia się ono tym, że powstałe drzewo jest nadmiernie rozbudowane
- Ogólne, im dane są bardziej złożone, tym większe prawdopodobieństwo przeuczenia
- Co robić, gdy drzewo jest nadmiernie rozbudowane (za wysokie) i przez to nieczytelne?
  - przycinamy drzewo (tree pruning)

Klasyfikacja

- Przycinanie drzewa, przykład ze zbiorem *adult*
  - zbiór ten zawiera 48842 rekordów z danymi spisu ludności w USA. Zmienną celu jest dochód (zmienna *income:* >50000$ oraz <=50000$), liczba zmiennych wejściowych wynosi 14. Jesteśmy zainteresowani klasyfikacją, czy dochód osoby jest w pierwszej, czy też drugiej grupie
  - zbiór danych zawiera zarówno zmienne ilościowe, jak i jakościowe. W wielu miejscach brakuje danych. Rekordy, gdzie brakuje danych zostaną usunięte
  - do zbudowania drzewa użyjemy zmienne: *age* (wiek), *education-num* (liczba lat poświęconych nauce), *capital-gain* (przyrost kapitału), *houses-per-week* (liczba godzin pracy w tygodniu), *race* (rasa), *sex* (płeć), *workclass* (rodzaj zatrudnienia), *marital-status* (stan cywilny)

Klasyfikacja

- https://archive.ics.uci.edu/ml/datasets/adult

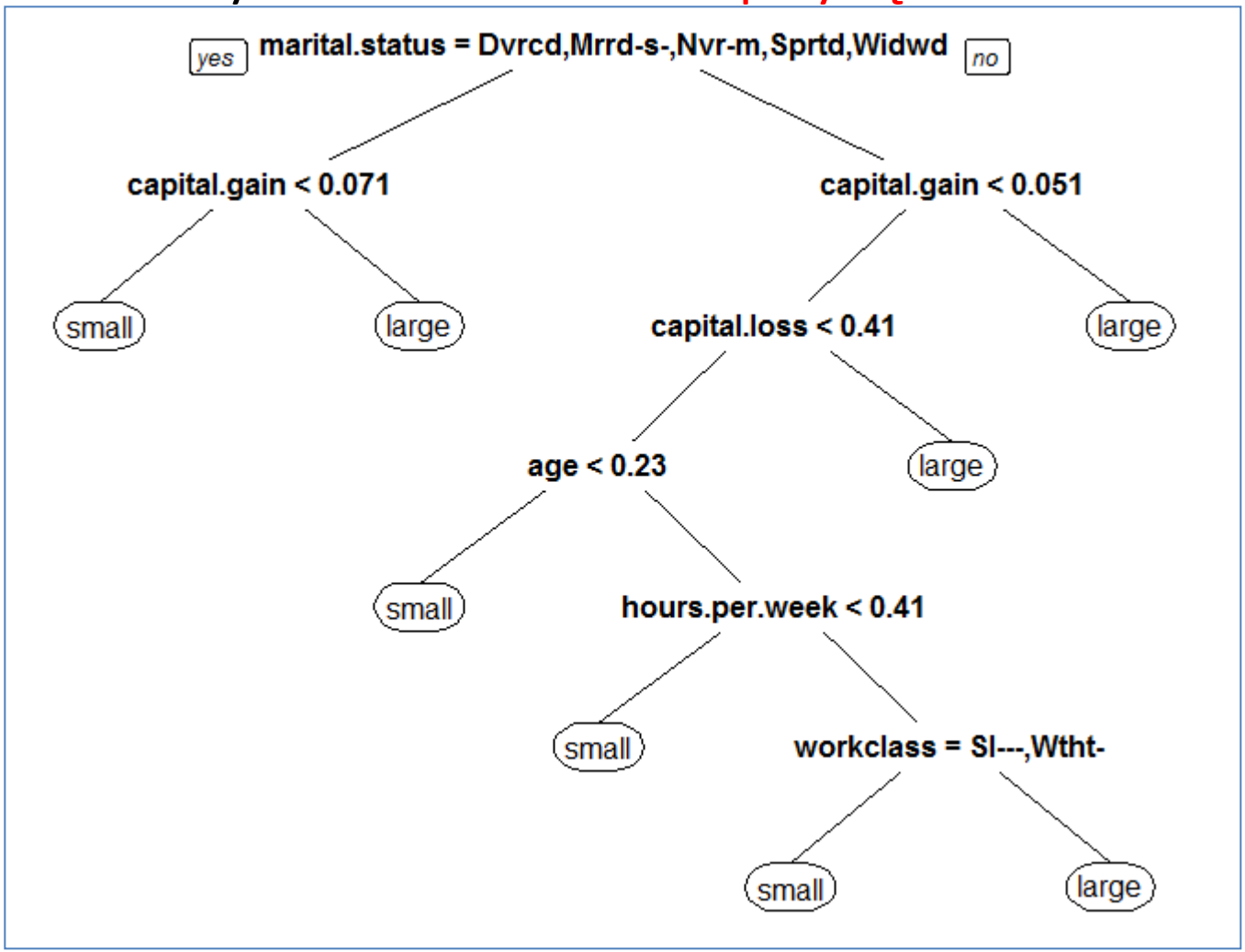
Data Set Characteristics:	Multivariate	**Number of Instances:**	48842	**Area:**	Social
Attribute Characteristics:	Categorical, Integer	**Number of Attributes:**	14	**Date Donated**	1996-05-01
Associated Tasks:	Classification	**Missing Values?**	Yes	**Number of Web Hits:**	1899488

Abstract: Predict whether income exceeds $50K/yr based on census data.

age: continuous.
workclass: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked.
fnlwgt: continuous.
education: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool.
education-num: continuous.
marital-status: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse.
occupation: Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces.
relationship: Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried.
race: White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black.
sex: Female, Male.
capital-gain: continuous.
capital-loss: continuous.
hours-per-week: continuous.
native-country: United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holand-Netherlands.

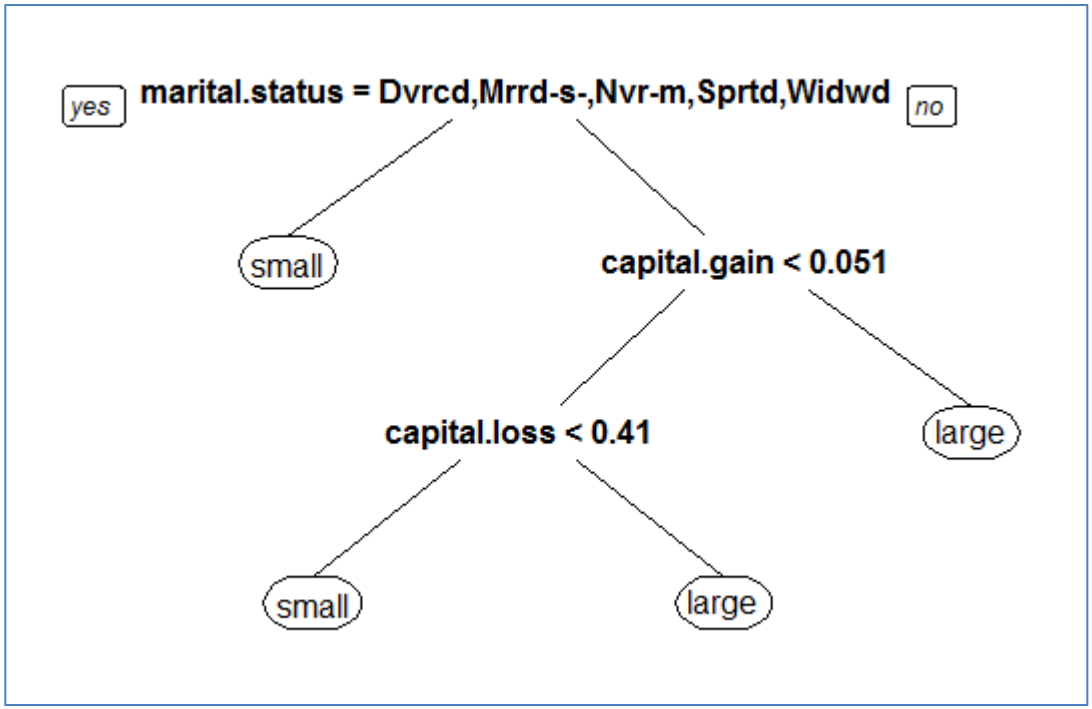
Klasyfikacja

- Zbiór danych *adult*. Drzewo nieprzycięte



Klasyfikacja

- Zbiór danych *adult*. Drzewo przycięte



Klasyfikacja

- Ocena jakości klasyfikatora
  - używamy najczęściej tzw. macierzy pomyłek (ang. confusion matrix)

postać liczbowa

predicted → / real ↓	*Klasa_pos*	*Klasa_neg*
Klasa_pos	114	86
Klasa_neg	7	93

postać procentowa

predicted → / real ↓	*Klasa_pos*	*Klasa_neg*
Klasa_pos	38%	29%
Klasa_neg	2%	31%

postać liczbowa

predicted → / real ↓	*Klasa_1*	*Klasa_2*	*Klasa_3*
Klasa_1	94	16	10
Klasa_2	21	113	16
Klasa_3	4	4	92

postać procentowa

predicted → / real ↓	*Klasa_1*	*Klasa_2*	*Klasa_3*
Klasa_1	25%	4%	3%
Klasa_2	6%	31%	4%
Klasa_3	1%	1%	25%

Źródło: https://algolytics.pl/jak-ocenic-jakosc-i-poprawnosc-modeli-klasyfikacyjnych-czesc-3-confusion-matrix/

Entropia

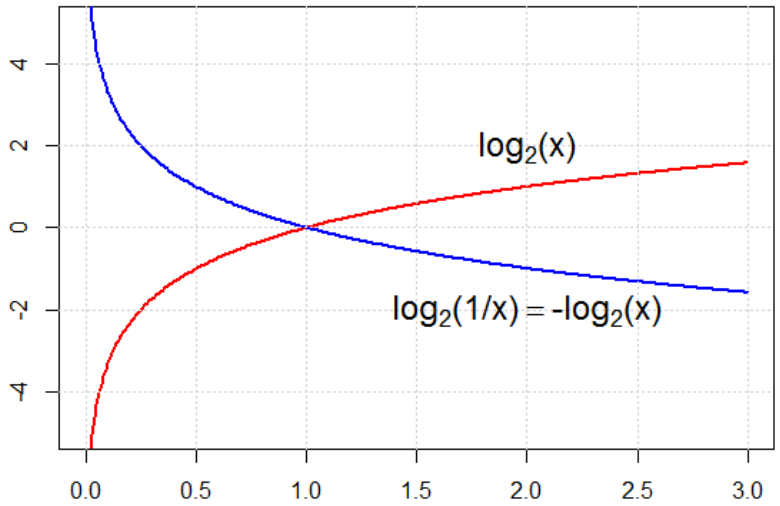
- Pojęcie entropii ma swoje źródło w termodynamice i jest związane z tym, jak bardzo nie znamy dokładnego mikrostanu układu
- Entropię można też interpretować jako niepewność wystąpienia danego zdarzenia elementarnego w następnej chwili. Jeżeli jakieś zdarzenie w zbiorze zdarzeń występuje z prawdopodobieństwem równym 1, to entropia układu wynosi wówczas 0, gdyż z góry wiadomo, co się stanie – nie ma niepewności. Entropia więc to miara niepewności (inaczej: miara niewiedzy)
- Własności entropii:
  - jest nieujemna
  - jest maksymalna, gdy prawdopodobieństwa zajść zdarzeń są takie same
  - jest równa 0, gdy stany systemu przyjmują wartości tylko 0 albo tylko 1
  - własność superpozycji – gdy dwa systemy są niezależne, to entropia sumy systemów równa się sumie entropii

Entropia

Uwaga: tutaj jest znak minus.

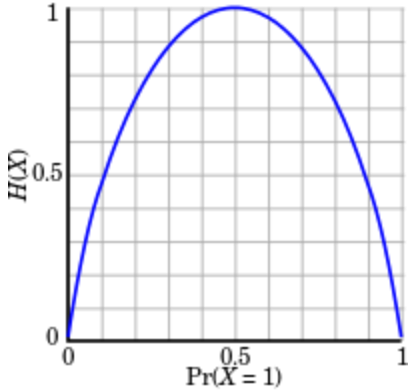
$$H(p_1, p_2, \ldots p_n) = \sum_{i=1}^{n} p_i \log_2\left(\frac{1}{p_i}\right) = -\sum_{i=1}^{n} p_i \log_2(p_i)$$

gdzie $p_1, \ldots, p_n$ są prawdopodobieństwami możliwych wyników przeprowadzanego eksperymentu. Będziemy używać logarytmu o podstawie 2, co odpowiada mierzeniu entropii w bitach



Entropia

- Przykłady
  - gdy mamy jedną możliwość entropia wynosi:
    $1 \cdot \log_2(1) = 0$
    (wszak nie ma tu miejsca na losowość)
  - gdy rzucimy uczciwą monetą to entropia wynosi:
    $- 0.5 \cdot \log_2(0.5) - 0.5 \cdot \log_2(0.5) = 0.5 + 0.5 = 1$
    (entropia przyjmuje maksymalną wartość)



  - jeżeli częściej wypadają reszki (załóżmy, że z prawdopodobieństwem 0.75) to entropia wynosi:
    $- 0.25 \cdot \log2(0.25) - 0.75 \cdot \log(0.75)$ 0.7158
  - entropia jest największa, gdy dla ustalonej liczby zdarzeń wszystkie są równoprawdopodobne – wtedy wynosi ona log(n). Dla kostki do gry z 6 ścianami to $\log_2(6) \approx 2,6$

Entropia

- Przykłady
  - warto pamiętać, że entropia jest zawsze miarą niewiedzy. Stąd np. kostka, na której widzimy wyrzucone dwa oczka, ma entropię zero. Dowiadujemy się dokładnie tyle, o ile entropia zmalała, tu:

$$H\left(\frac{1}{6},\frac{1}{6},\frac{1}{6},\frac{1}{6},\frac{1}{6},\frac{1}{6}\right)-H(0{,}1,0,0,0,0)=$$

$$\log_2(6)-\log_2(1)\approx 2{,}6$$

  - gdyby ktoś nam powiedział „wypadło jedno, dwa lub trzy oczka”, nasza wiedza końcowa byłaby niepełna i dowiedzielibyśmy się log(6) − log(3) = 1 bit informacji

Źródło: http://www.deltami.edu.pl/temat/matematyka/rachunek_prawdopodobienstwa/2015/10/26/1511delta-entropia.pdf

Algorytm C4.5

- Załóżmy, że mamy możliwy podział *S*, który dzieli zbiór *T* na podzbiory T_1, T_2, …, T_k . Wówczas entropia tego podziału może być obliczona jako suma ważona entropii dla pojedynczych podzbiorów, czyli:

$$H_S(T) = \sum_{i=1}^{k} p_i H_S(T_i)$$

gdzie p_i reprezentuje procent rekordów w *i*-tym podzbiorze. Możemy wtedy zdefiniować zysk informacji jako

$$zysk(S) = H(T) - H_S(T)$$

czyli zwiększenie informacji wytworzone przez podział zbiory *T* zgodnie z podziałem *S*. W każdym węźle decyzyjnym algorytm C4.5 wybiera podział optymalny, czyli mający największy zysk informacji *zysk(S)*

Entropia – budowa drzewa klasyfikacyjnego

- Obliczenia dla drzewa klasyfikacyjnego, zbiór danych *golf*

Outlook	Temp	Humidity	Windy	PlayGolf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Entropia – budowa drzewa klasyfikacyjnego

- Krok 1: entropia przed podziałem

Outlook	Temp	Humidity	Windy	PlayGolf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Play Golf	
Yes	No
9	5

$$H(PlayGolf) = H\left(\frac{5}{14}, \frac{9}{14}\right) = -(0{,}36 \cdot \log_2 0{,}36) - (0{,}64 \cdot \log_2 0{,}64) = 0{,}94$$

Źródło: http://monaen.github.io/PrivacyDecisionTree/

Entropia – budowa drzewa klasyfikacyjnego

- Krok 2: kolejne podziały mogą być dla 4 różnych zmiennych. Obliczamy więc entropie dla poszczególnych zmiennych
  - zmienna *Outlook*

$$H(PlayGolf, Outlook) =$$

$$P(Sunny)\cdot H\left(\frac{3}{5},\frac{2}{5}\right)+P(Overcast)\cdot H\left(\frac{4}{4},\frac{0}{4}\right)+P(Rainy)\cdot H\left(\frac{2}{5},\frac{3}{5}\right)=$$

$$\frac{5}{14}\left[-(0{,}6\cdot\log_2 0{,}6)-(0{,}4\cdot\log_2 0{,}4)\right]+$$

$$\frac{4}{14}\left[-(1\cdot\log_2 1)-(0\cdot\log_2 0)\right]+$$

$$\frac{5}{14}\left[-(0{,}4\cdot\log_2 0{,}4)-(0{,}6\cdot\log_2 0{,}6)\right]$$

$$=0{,}3468+0+0{,}3468=0{,}6935$$

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

Outlook	Temp	Humidity	Windy	PlayGolf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Entropia – budowa drzewa klasyfikacyjnego

- Krok 2
  - zmienna *Temp*

$$H(PlayGolf, Temp) =$$

$$P(Hot)\cdot H\left(\frac{2}{4},\frac{2}{4}\right)+P(Mild)\cdot H\left(\frac{4}{6},\frac{2}{6}\right)+P(Cool)\cdot H\left(\frac{3}{4},\frac{1}{4}\right)=$$

$$\frac{4}{14}\left[-(0{,}5\cdot\log_2 0{,}5)-(0{,}5\cdot\log_2 0{,}5)\right]+$$

$$\frac{6}{14}\left[-\left(\frac{4}{6}\cdot\log_2\frac{4}{6}\right)-\left(\frac{2}{6}\cdot\log_2\frac{2}{6}\right)\right]+$$

$$\frac{4}{14}\left[-(0{,}75\cdot\log_2 0{,}75)-(0{,}25\cdot\log_2 0{,}25)\right]$$

$$=0{,}2857+0{,}3936+0{,}2318=0{,}9111$$

		Play Golf	
		Yes	No
Temp.	Hot	2	2
	Mild	4	2
	Cool	3	1
Gain = 0.029			

Outlook	Temp	Humidity	Windy	PlayGolf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Entropia – budowa drzewa klasyfikacyjnego

- Krok 2
  - zmienna *Humidity*

$$H(PlayGolf, Humidity) =$$

$$P(High) \cdot H\left(\frac{3}{7}, \frac{4}{7}\right) + P(Normal) \cdot H\left(\frac{6}{7}, \frac{1}{7}\right) =$$

$$\frac{7}{14}\left[-\left(\frac{3}{7} \cdot \log_2 \frac{3}{7}\right) - \left(\frac{4}{7} \cdot \log_2 \frac{4}{7}\right)\right] +$$

$$\frac{7}{14}\left[-\left(\frac{6}{7} \cdot \log_2 \frac{6}{7}\right) - \left(\frac{1}{7} \cdot \log_2 \frac{1}{7}\right)\right]$$

$$= 0{,}4926 + 0{,}2958 = 0{,}7885$$

		Play Golf	
		Yes	No
Humidity	High	3	4
	Normal	6	1
Gain = 0.152			

Outlook	Temp	Humidity	Windy	PlayGolf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Entropia – budowa drzewa klasyfikacyjnego

- Krok 2
  - zmienna *Windy*

$$H(PlayGolf, Windy) =$$

$$P(False) \cdot H\left(\frac{6}{8}, \frac{2}{8}\right) + P(True) \cdot H\left(\frac{3}{6}, \frac{3}{6}\right) =$$

$$\frac{8}{14}\left[-\left(\frac{6}{8} \cdot \log_2 \frac{6}{8}\right) - \left(\frac{2}{8} \cdot \log_2 \frac{2}{8}\right)\right] +$$

$$\frac{6}{14}\left[-\left(\frac{3}{6} \cdot \log_2 \frac{3}{6}\right) - \left(\frac{3}{6} \cdot \log_2 \frac{3}{6}\right)\right]$$

$$= 0{,}4636 + 0{,}4286 = 0{,}8922$$

		Play Golf	
		Yes	No
Windy	False	6	2
	True	3	3
Gain = 0.048			

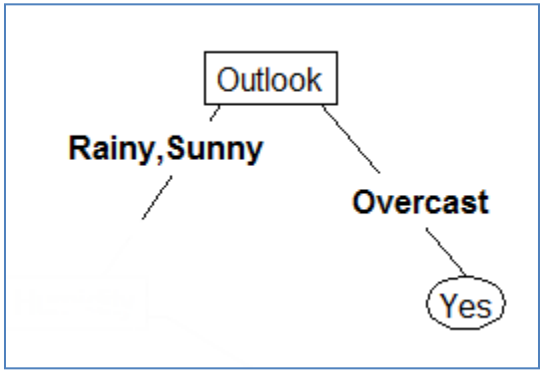
Outlook	Temp	Humidity	Windy	PlayGolf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Entropia – budowa drzewa klasyfikacyjnego

- Krok 3: musimy teraz policzyć zyski informacji (ang. information gain) reprezentowany przez każdy z atrybutów. Ten atrybut, gdzie zysk będzie największy, zostanie wybrany jako pierwszy podział

$$H(PlayGolf) - H(PlayGolf, Outlook) = 0{,}9403 - 0{,}6935 = 0{,}2467$$
$$H(PlayGolf) - H(PlayGolf, Temp) = 0{,}9403 - 0{,}9111 = 0{,}02922$$
$$H(PlayGolf) - H(PlayGolf, Humidity) = 0{,}9403 - 0{,}7885 = 0{,}1518$$
$$H(PlayGolf) - H(PlayGolf, Windy) = 0{,}9403 - 0{,}8922 = 0{,}04813$$

  - zysk informacji (inaczej: redukcja entropii) jest największy dla zmiennej *Outlook* i ona jest brana, jako pierwszy podział
  - ponadto gałąź z zerową entropią (*Overcast*) staje się liściem



Entropia – budowa drzewa klasyfikacyjnego

- Krok 4: Wyliczamy kolejne podziały, zbiór danych został już zredukowany o pozycje, gdzie *Outlook = Overcast*

Outlook	Temp	Humidity	Windy	PlayGolf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Sunny	Mild	High	True	No

$$H(PlayGolf) = H\left(\frac{5}{10}, \frac{5}{10}\right) = -(0{,}5 \cdot \log_2 0{,}5) - (0{,}5 \cdot \log_2 0{,}5) = 1{,}00$$

Entropia – budowa drzewa klasyfikacyjnego

- Krok 4

		PlayGolf	
		Yes	No
Temp	Hot	0	2
	Mild	3	2
	Cool	2	1

$$H(PlayGolf, Temp) =$$
$$P(Hot)\cdot H\left(\frac{0}{2},\frac{2}{2}\right) + P(Mild)\cdot H\left(\frac{3}{5},\frac{2}{5}\right) + P(Cool)\cdot H\left(\frac{2}{3},\frac{1}{3}\right) =$$
$$\frac{2}{10}\left[-\left(\frac{0}{2}\cdot\log_2\frac{0}{2}\right)-\left(\frac{2}{2}\cdot\log_2\frac{2}{2}\right)\right] + \frac{5}{10}\left[-\left(\frac{3}{5}\cdot\log_2\frac{3}{5}\right)-\left(\frac{2}{5}\cdot\log_2\frac{2}{5}\right)\right] + \frac{3}{10}\left[-\left(\frac{2}{3}\cdot\log_2\frac{2}{3}\right)-\left(\frac{1}{3}\cdot\log_2\frac{1}{3}\right)\right]$$
$$= 0 + 0{,}4855 + 0{,}2755 = 0{,}761$$

		PlayGolf	
		Yes	No
Humidity	High	1	4
	Normal	4	1

$$H(PlayGolf, Humidity) =$$
$$P(High)\cdot H\left(\frac{1}{5},\frac{4}{5}\right) + P(Normal)\cdot H\left(\frac{4}{5},\frac{1}{5}\right) =$$
$$\frac{5}{10}\left[-\left(\frac{1}{5}\cdot\log_2\frac{1}{5}\right)-\left(\frac{4}{5}\cdot\log_2\frac{4}{5}\right)\right] + \frac{5}{10}\left[-\left(\frac{4}{5}\cdot\log_2\frac{4}{5}\right)-\left(\frac{1}{5}\cdot\log_2\frac{1}{5}\right)\right]$$
$$= 0{,}361 + 0{,}361 = 0{,}7219$$

		PlayGolf	
		Yes	No
Windy	False	4	2
	True	1	3

$$H(PlayGolf, Windy) =$$
$$P(False)\cdot H\left(\frac{4}{6},\frac{2}{6}\right) + P(True)\cdot H\left(\frac{1}{4},\frac{3}{4}\right) =$$
$$\frac{6}{10}\left[-\left(\frac{4}{6}\cdot\log_2\frac{4}{6}\right)-\left(\frac{2}{6}\cdot\log_2\frac{2}{6}\right)\right] + \frac{4}{10}\left[-\left(\frac{1}{4}\cdot\log_2\frac{1}{4}\right)-\left(\frac{3}{4}\cdot\log_2\frac{3}{4}\right)\right]$$
$$= 0{,}551 + 0{,}3245 = 0{,}8755$$

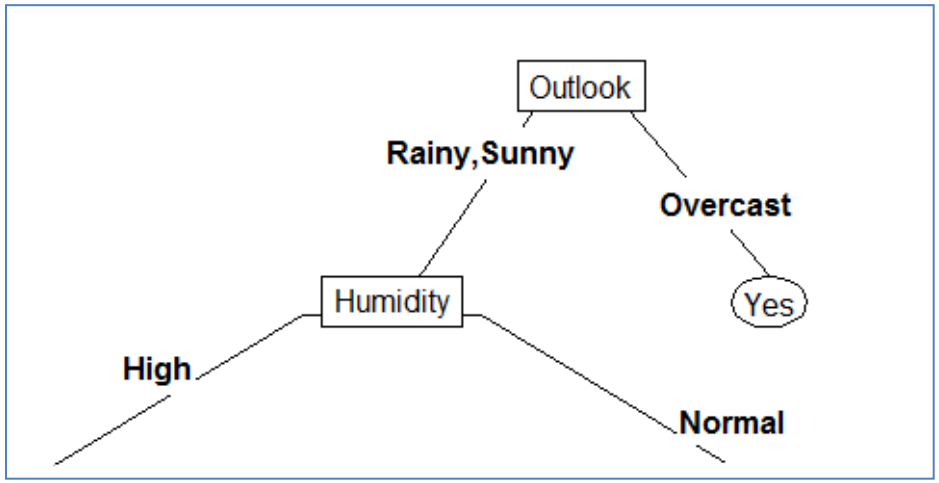
Entropia – budowa drzewa klasyfikacyjnego

- Krok 5: musimy teraz policzyć zyski informacji (ang. information gain) reprezentowany przez każdy z atrybutów. Ten atrybut, gdzie zysk będzie największy zostanie wybrany jako pierwszy podział
  - zysk informacji (inaczej: redukcja entropii) jest największy dla zmiennej *Humidity* i ona jest brana, jako kolejny podział

$$H(PlayGolf) - H(PlayGolf, Temp) = 1 - 0{,}761 = 0{,}239$$

$$H(PlayGolf) - H(PlayGolf, Humidity) = 1 - 0{,}7219 = 0{,}2781$$

$$H(PlayGolf) - H(PlayGolf, Windy) = 1 - 0{,}8755 = 0{,}1245$$



Entropia – budowa drzewa klasyfikacyjnego

- Krok 6: Wyliczamy kolejne podziały, zbiór danych został kolejny już raz zredukowany
  - *Humidity = High*

Outlook	Temp	Humidity	Windy	PlayGolf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Sunny	Mild	High	False	Yes
Rainy	Mild	High	False	No
Sunny	Mild	High	True	No

$$H(PlayGolf) = H\left(\frac{4}{5},\frac{1}{5}\right) = -(0{,}8\cdot\log_2 0{,}8) - (0{,}2\cdot\log_2 0{,}2) = 0{,}7219$$

		PlayGolf	
		Yes	No
Outlook	Sunny	1	1
	Rainy	0	3

$$H(PlayGolf, Outlook) = P(Sunny)\cdot H\left(\frac{1}{2},\frac{1}{2}\right) + P(Rainy)\cdot H\left(\frac{0}{3},\frac{3}{3}\right) =$$
$$\frac{2}{5}\left[-\left(\frac{1}{2}\cdot\log_2\frac{1}{2}\right)-\left(\frac{1}{2}\cdot\log_2\frac{1}{2}\right)\right] + \frac{3}{5}\left[-\left(\frac{0}{3}\cdot\log_2\frac{0}{3}\right)-\left(\frac{3}{3}\cdot\log_2\frac{3}{3}\right)\right] = 0{,}4 + 0 = 0{,}4$$

		PlayGolf	
		Yes	No
Temp	Hot	0	2
	Mild	1	2

$$H(PlayGolf, Temp) = P(Hot)\cdot H\left(\frac{0}{2},\frac{2}{2}\right) + P(Mild)\cdot H\left(\frac{1}{3},\frac{2}{3}\right) =$$
$$\frac{2}{5}\left[-\left(\frac{0}{2}\cdot\log_2\frac{0}{2}\right)-\left(\frac{2}{2}\cdot\log_2\frac{2}{2}\right)\right] + \frac{3}{5}\left[-\left(\frac{1}{3}\cdot\log_2\frac{1}{3}\right)-\left(\frac{2}{3}\cdot\log_2\frac{2}{3}\right)\right] = 0 + 0{,}551 = 0{,}551$$

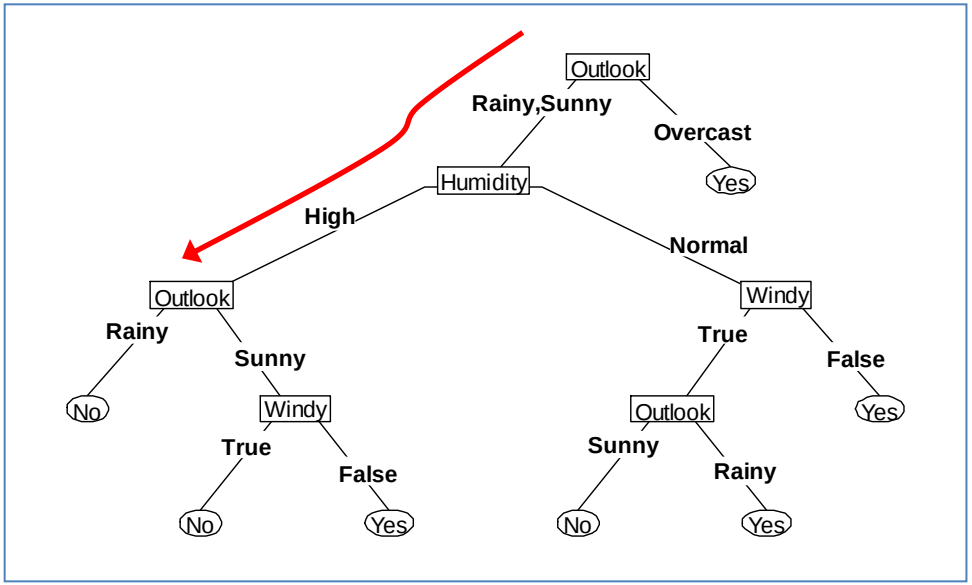
		PlayGolf	
		Yes	No
Windy	False	1	2
	True	0	2

$$H(PlayGolf, Windy) = P(False)\cdot H\left(\frac{1}{3},\frac{2}{3}\right) + P(True)\cdot H\left(\frac{0}{2},\frac{2}{2}\right) =$$
$$\frac{3}{5}\left[-\left(\frac{1}{3}\cdot\log_2\frac{1}{3}\right)-\left(\frac{2}{3}\cdot\log_2\frac{2}{3}\right)\right] + \frac{2}{5}\left[-\left(\frac{0}{2}\cdot\log_2\frac{0}{2}\right)-\left(\frac{2}{2}\cdot\log_2\frac{2}{2}\right)\right] = 0{,}551 + 0 = 0{,}551$$

Entropia – budowa drzewa klasyfikacyjnego

- Krok 7: zysk informacji (inaczej: redukcja entropii) jest największy dla zmiennej *Outlook* ona jest brana, jako kolejny podział

$$H(PlayGolf) - H(PlayGolf, Outlook) = 0{,}7219 - 0{,}4 = 0{,}3219$$
$$H(PlayGolf) - H(PlayGolf, Temp) = 0{,}7219 - 0{,}551 = 0{,}171$$
$$H(PlayGolf) - H(PlayGolf, Windy) = 0{,}7219 - 0{,}551 = 0{,}171$$



Entropia – budowa drzewa klasyfikacyjnego

- Krok 8: z naszej tabeli widać, że
  - gdy *Outlook = Rainy* → *PlayGolf* zawsze *No*

Outlook	Temp	Humidity	Windy	PlayGolf
Rainy	**Hot**	**High**	**False**	**No**
Rainy	**Hot**	**High**	**True**	**No**
Sunny	**Mild**	**High**	**False**	**Yes**
Rainy	**Mild**	**High**	**False**	**No**
Sunny	**Mild**	**High**	**True**	**No**

  - *gdy Outlook = Sunny* → mam tylko dwie opcje do wyboru więc nie musimy nic formalnie liczyć

Outlook	Temp	Humidity	Windy	PlayGolf
Sunny	**Mild**	**High**	**False**	**Yes**
Sunny	**Mild**	**High**	**True**	**No**

Entropia – budowa drzewa klasyfikacyjnego

- Krok 9: Wyliczamy kolejne podziały, zbiór danych został kolejny już raz zredukowany
  - *Humidity = Normal*

Outlook	Temp	Humidity	Windy	PlayGolf
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes

$$H(PlayGolf) = H\left(\frac{4}{5},\frac{1}{5}\right) = -(0{,}8\cdot\log_2 0{,}8) - (0{,}2\cdot\log_2 0{,}2) = 0{,}7219$$

		PlayGolf	
		Yes	No
Outlook	Sunny	2	1
	Rainy	2	0

$$H(PlayGolf, Outlook) = P(Sunny)\cdot H\left(\frac{2}{3},\frac{1}{3}\right) + P(Rainy)\cdot H\left(\frac{2}{2},\frac{0}{2}\right) =$$
$$\frac{3}{5}\left[-\left(\frac{2}{3}\cdot\log_2\frac{2}{3}\right)-\left(\frac{1}{3}\cdot\log_2\frac{1}{3}\right)\right] + \frac{2}{5}\left[-\left(\frac{2}{2}\cdot\log_2\frac{2}{2}\right)-\left(\frac{0}{2}\cdot\log_2\frac{0}{2}\right)\right] = 0{,}551 + 0 = 0{,}551$$

		PlayGolf	
		Yes	No
Temp	Cool	2	1
	Mild	2	0

$$H(PlayGolf, Temp) = P(Cool)\cdot H\left(\frac{2}{3},\frac{1}{3}\right) + P(Mild)\cdot H\left(\frac{2}{2},\frac{0}{2}\right) =$$
$$\frac{3}{5}\left[-\left(\frac{2}{3}\cdot\log_2\frac{2}{3}\right)-\left(\frac{1}{3}\cdot\log_2\frac{1}{3}\right)\right] + \frac{2}{5}\left[-\left(\frac{2}{2}\cdot\log_2\frac{2}{2}\right)-\left(\frac{0}{2}\cdot\log_2\frac{0}{2}\right)\right] = 0{,}551 + 0 = 0{,}551$$

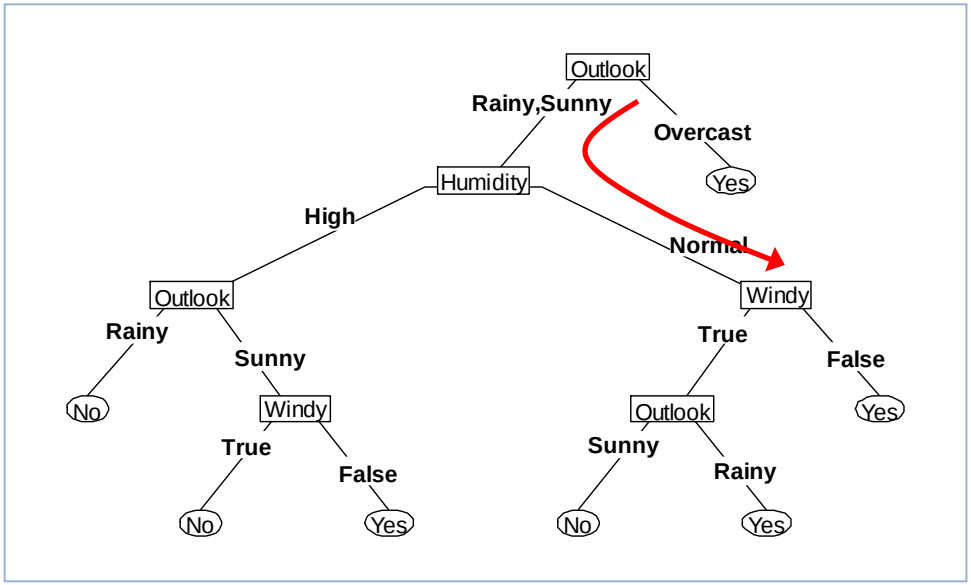
		PlayGolf	
		Yes	No
Windy	False	3	0
	True	1	1

$$H(PlayGolf, Windy) = P(False)\cdot H\left(\frac{3}{3},\frac{0}{3}\right) + P(True)\cdot H\left(\frac{1}{2},\frac{1}{2}\right) =$$
$$\frac{3}{5}\left[-\left(\frac{3}{3}\cdot\log_2\frac{3}{3}\right)-\left(\frac{0}{3}\cdot\log_2\frac{0}{3}\right)\right] + \frac{2}{5}\left[-\left(\frac{1}{2}\cdot\log_2\frac{1}{2}\right)-\left(\frac{1}{2}\cdot\log_2\frac{1}{2}\right)\right] = 0 + 0{,}4 = 0{,}4$$

Entropia – budowa drzewa klasyfikacyjnego

- Krok 10: zysk informacji (inaczej: redukcja entropii) jest największy dla zmiennej *Windy,* ona jest brana, jako kolejny podział

$$H(PlayGolf) - H(PlayGolf, Outlook) = 0{,}7219 - 0{,}551 = 0{,}171$$
$$H(PlayGolf) - H(PlayGolf, Temp) = 0{,}7219 - 0{,}551 = 0{,}171$$
$$H(PlayGolf) - H(PlayGolf, Windy) = 0{,}7219 - 0{,}4 = 0{,}3219$$



Entropia – budowa drzewa klasyfikacyjnego

- Krok 11: z naszej tabeli widać, że
  - gdy *Windy= False→ PlayGolf* zawsze *Yes*

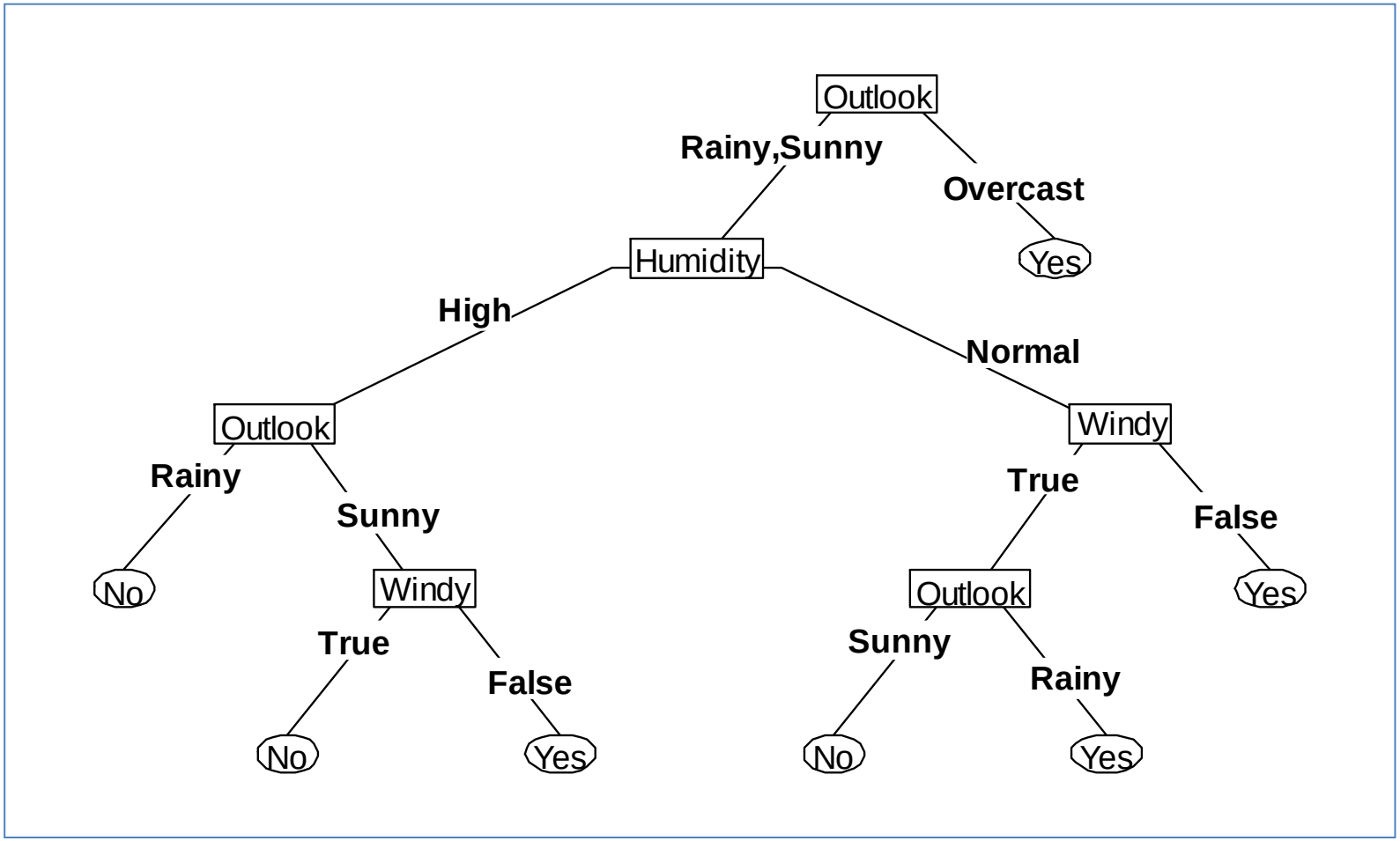
Outlook	Temp	Humidity	Windy	PlayGolf
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes

  - *gdy Windy = True* → mam tylko dwie opcje do wyboru więc nie musimy nic formalnie liczyć

Outlook	Temp	Humidity	Windy	PlayGolf
Sunny	Cool	Normal	True	No
Rainy	Mild	Normal	True	Yes

Entropia – budowa drzewa klasyfikacyjnego

- Przypomnijmy wynik końcowy



Entropia – budowa drzewa klasyfikacyjnego

- Po co pokazywaliśmy te bardzo szczegółowe kompletne obliczenia?
  - aby uzmysłowić, że nawet do bardzo prostych przypadków obliczeń jest „co nie miara”. W zasadzie bez użycia komputerów nie jesteśmy w stanie rozwiązywać bardziej złożonych przypadków
- Warto więc chwilę zastanowić się, jak wielką sprawą było pojawienie się na świecie ogólnie dostępnych komputerów

Klasyfikacja danych tekstowych – przykład

- Wykorzystamy 2000 recenzji filmów (po angielsku)
  - 1000 recenzji jest negatywnych a 1000 pozytywnych (plik *movie_reviews.csv*)
  - każda recenzja jest dość długa (kilkaset znaków)
- Skorzystamy również z tzw. leksykonów emocjonalnych zawierających starannie wybraną pewną liczbę słów mających wydźwięk pozytywny i negatywny
  - pliki *positive-words.txt* oraz *negative-words.txt*
  - w każdym pliku jest po kilka tysięcy słów
- Posiadany zbiór podzielmy na treningowy i testowy, zbudujemy klasyfikator i sprawdzimy jego jakość
  - na podstawie macierzy pomyłek

Klasyfikacja danych tekstowych – przykład

- Krok 1
  - wczytanie pliku z recenzjami, utworzenie korpusu (w rozumieniu pakietu tdm), wykonanie standardowego preprocessingu, utworzenie obiektu Document-Term Matrix oraz usunięcie bardzo rzadko pojawiających się fraz

```
reviews = read.csv("movie_reviews.csv", stringsAsFactors = F, row.names = 1)

reviews_corpus = VCorpus(VectorSource(reviews$content))

reviews_corpus = tm_map(reviews_corpus, content_transformer(tolower))
reviews_corpus = tm_map(reviews_corpus, removeNumbers)
reviews_corpus = tm_map(reviews_corpus, removePunctuation)
reviews_corpus = tm_map(reviews_corpus, removeWords, stopwords("SMART"))
reviews_corpus = tm_map(reviews_corpus, stripWhitespace)

dtm <- DocumentTermMatrix(reviews_corpus, control = list(weighting = weightTfIdf))
dtm = removeSparseTerms(dtm, 0.95)
```

Można poeksperymentować z innym schematami tworzenia DTM oraz z innym współczynnikiem rzadkości

Klasyfikacja danych tekstowych – przykład

- Krok 2
  - wczytanie listy słów o wydźwięku pozytywnym i negatywnym, wyszukanie w tekstach recenzji tych słów oraz ich policzenie
  - utworzenie finalnej ramki danych, na podstawie której zostanie zbudowane drzewo klasyfikacyjne

```
neg_words = read.table("negative-words.txt", header = F, stringsAsFactors = F)[, 1]
pos_words = read.table("positive-words.txt", header = F, stringsAsFactors = F)[, 1]

reviews$neg = tm_term_score(DocumentTermMatrix(reviews_corpus), neg_words)
reviews$pos = tm_term_score(DocumentTermMatrix(reviews_corpus), pos_words)

reviews$content = NULL
rev.dtm = cbind(reviews, as.matrix(dtm))
rev.dtm$polarity = as.factor(rev.dtm$polarity)
```

Klasyfikacja danych tekstowych – przykład

- Krok 2
  - finalna ramka danych (2000 wierszy, bo tyle mamy recenzji)

polarity	neg	pos	ability	absolutely	act	acting	action	actor	actors	...
0	24	22	0	0	0	0.000000	0.00000	0.00000	0.007498	...
0	6	6	0	0	0	0.037993	0.01884	0.00000	0.000000	...
0	17	16	0	0	0	0.009269	0.00000	0.00000	0.000000	...
0	25	17	0	0	0	0.000000	0.00000	0.00000	0.008451	...
0	29	15	0	0	0	0.000000	0.00000	0.01465	0.000000	...
				...						
1	12	29	0.00000	0	0	0.008527	0.00000	0.000000	0.00000	...
1	20	38	0.00000	0	0	0.006526	0.00000	0.000000	0.00000	...
1	13	14	0.00000	0	0	0.000000	0.00000	0.016403	0.00000	...
1	50	42	0.00779	0	0	0.000000	0.01115	0.000000	0.00000	...
1	13	33	0.00000	0	0	0.000000	0.00000	0.008627	0.01427	...
1	28	21	0.00000	0	0	0.000000	0.00000	0.000000	0.00000	...

Kolumna wskazuje rodzaj recenzji (pozytywna / negatywna)

Dwie kolumny z ilościami słów pozytywnych i negatywnych znalezionych w poszczególnych recenzjach

Macierz Document-Term-Matrix (pokazano tylko 5 pierwszych kolumn. W naszym przykładzie jest ich w sumie kilkaset)

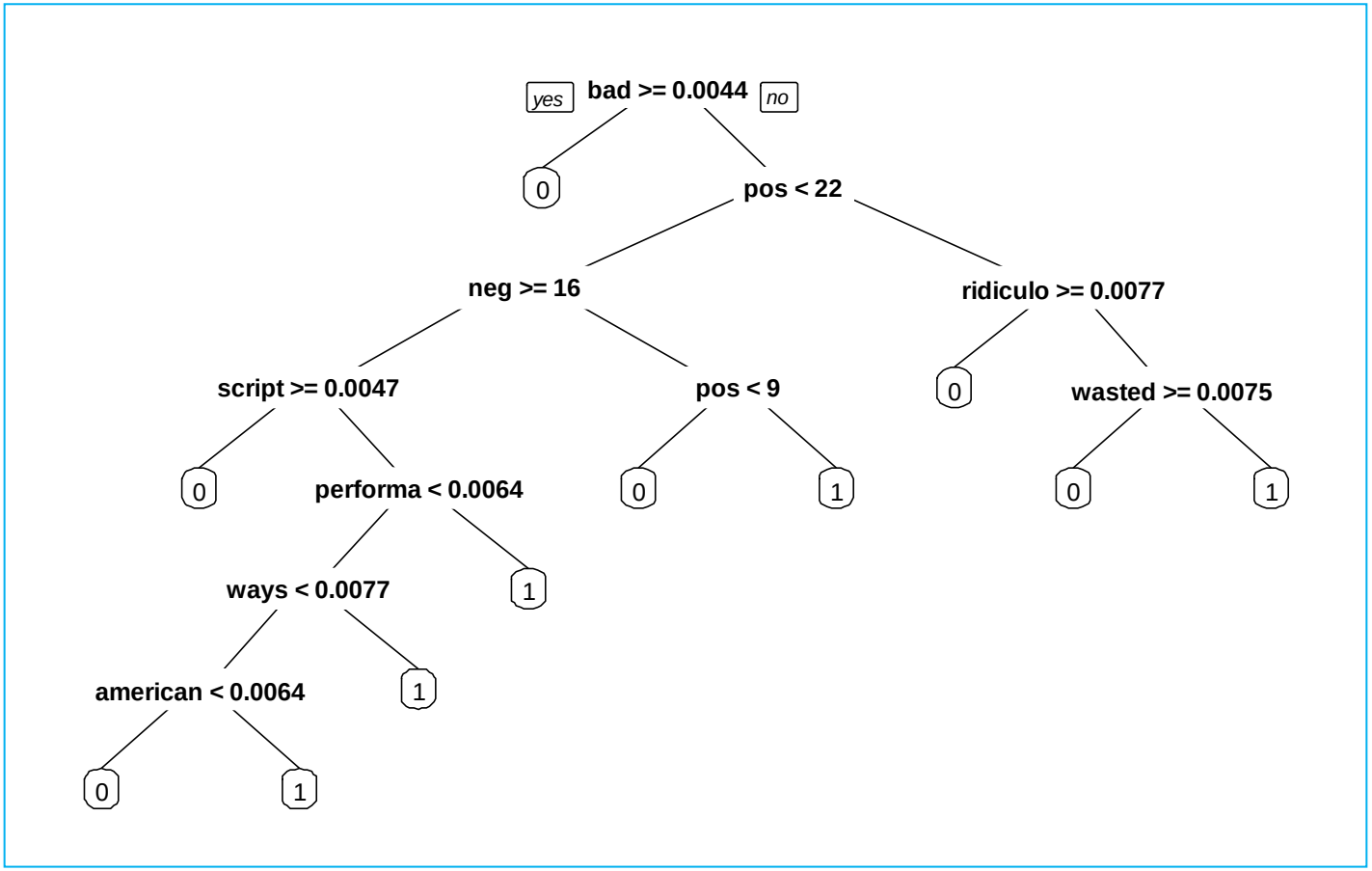
Klasyfikacja danych tekstowych – przykład

- Krok 3
  - podzielenie finalnej ramki danych na zbiór treningowy oraz testowy, utworzenie drzewa klasyfikacyjnego
  - uwaga: mamy tu pewną losowość (funkcja *sample*) i dlatego za pomocą *set.seed* wymuszamy „usunięcie losowości

```
set.seed(1234)
id_train <- sample(nrow(rev.dtm),nrow(rev.dtm) * 0.80)
rev.dtm.train = rev.dtm[id_train,]
rev.dtm.test = rev.dtm[-id_train,]

rev.dtm.tree = rpart(polarity ~ .,  method = "class", data = rev.dtm.train)
prp(rev.dtm.tree))
```

Klasyfikacja danych tekstowych – przykład

- Krok 3
  - finalne drzewo klasyfikacyjne (zwróćmy uwagę, które kolumny zostały użyte a które nie)



Klasyfikacja danych tekstowych – przykład

- Krok 4
  - „puszczenie” na zbudowany model danych testowych i wyświetlenie macierzy pomyłek

```
pred.tree = predict(rev.dtm.tree, rev.dtm.test,  type="class")
table(rev.dtm.test$polarity, pred.tree, dnn = c("Obs","Pred"))
Pred
Obs   0   1
  0 143  47
  1  77 133
```

  - na koniec pokazujemy wyniki dwóch zupełnie innych niż drzewo klasyfikacyjne klasyfikatorów

```
rev.dtm.svm = svm(polarity ~ ., data = rev.dtm.train)

Pred
Obs   0   1
  0 154  36
  1  46 164

rev.dtm.nnet = nnet(polarity ~ ., data = rev.dtm.train, size = 1, maxit = 500)

Pred
Obs   0   1
  0 143  47
  1  54 156
```

Analiza wydźwięku (Sentimental Analysis)

Analiza wydźwięku (~~sentymentu~~)

- Zajmuje się analizą zawartości emocjonalnej wypowiedzi (w praktyce zwykle chodzi o tekst pisany)
- Dwa główne podejścia
  - nienadzorowane. Wykorzystuje tzw. leksykony emocjonalne (ang. sentiment lexicons) i opierając się na nich ocenia wypowiedź.
  - nadzorowane. Musimy mieć jakiś zbiór uczący (treningowy) i na jego bazie budujemy odpowiedni klasyfikator i oceniamy inne wypowiedzi.
  - podejście nadzorowane jest dużo trudniejsze niż nadzorowane ale często daje lepsze efekty
  - podejście nienadzorowane jest szybkie, intuicyjne, łatwe w użyciu, ale bardziej „sztywne”, nie uwzględnia często kontekstu, zabarwienia, nastawienia, emocji itd.
    - ✓ on jest jakiś taki szalony ☹
    - ✓ on jest szalenie dobry ☺

Analiza wydźwięku

- Leksykony emocjonalne. Trzy (chyba) najpopularniejsze to:
  - AFINN from Finn Årup Nielsen,
  - bing from Bing Liu and collaborators
  - nrc from Saif Mohammad and Peter Turney

```
> afinn
# A tibble: 2,477 x 2
   word       value
   <chr>      <dbl>
 1 abandon       -2
 2 abandoned     -2
 3 abandons      -2
 4 abducted      -2
 5 abduction     -2
 6 abductions    -2
 7 abhor         -3
 8 abhorred      -3
 9 abhorrent     -3
10 abhors        -3
# ... with 2,467 more rows
```

```
> bing
# A tibble: 6,786 x 2
   word        sentiment
   <chr>       <chr>
 1 2-faces     negative
 2 abnormal    negative
 3 abolish     negative
 4 abominable  negative
 5 abominably  negative
 6 abominate   negative
 7 abomination negative
 8 abort       negative
 9 aborted     negative
10 aborts      negative
# ... with 6,776 more rows
```

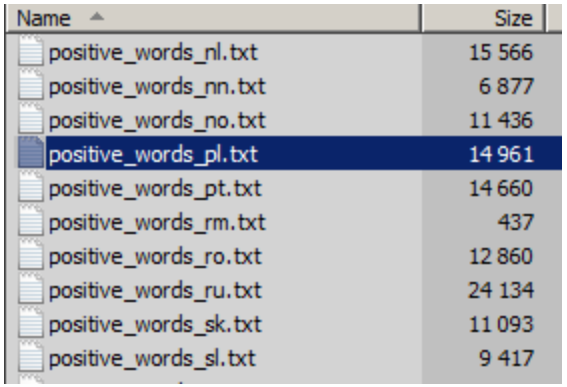
```
> nrc
# A tibble: 13,901 x 2
   word        sentiment
   <chr>       <chr>
 1 abacus      trust
 2 abandon     fear
 3 abandon     negative
 4 abandon     sadness
 5 abandoned   anger
 6 abandoned   fear
 7 abandoned   negative
 8 abandoned   sadness
 9 abandonment anger
10 abandonment fear
# ... with 13,891 more rows
```

Analiza wydźwięku

- Leksykony emocjonalne. Inne języki (niż angielski)
  - https://www.kaggle.com/rtatman/sentiment-lexicons-for-81-languages (gradacja tylko: pozytywny / negatywny)
  - 81 języków, w tym polski
  - sposób generowania tych leksykonów wydaje się dość kontrowersyjny. Opis ze strony projektu: „The sentiment lexicons in this dataset were generated via graph propagation based on a knowledge graph--a graphical representation of real-world entities and the links between them. The general intuition is that words which are closely linked on a knowledge graph probably have similar sentiment polarities. For this project, sentiments were generated based on English sentiment lexicons.”

Analiza wydźwięku

- Leksykony emocjonalne. Inne języki (niż angielski)
  - https://www.kaggle.com/rtatman/sentiment-lexicons-for-81-languages (gradacja tylko: pozytywny / negatywny)



Niektóre decyzje są kontrowersyjne. Dlaczego słowo „brat” znalazło się w grupie negatywnych?

Na liście pozytywnej jest ewidentnie neutralne słowo „jak” i inne.

Analiza wydźwięku

- Leksykony emocjonalne. Język polski
  - http://plwordnet.pwr.wroc.pl/wordnet/

Analiza wydźwięku

- Leksykony emocjonalne. Język polski
  - http://plwordnet.pwr.wroc.pl/wordnet/
    - ✓ słownik_anotacji_emocjonalnej.csv (178.562 pozycje)

słownik_anotacji_emocjonlanej.csv

```
lemat,wariant,czesc_mowy,jednostka_nacechowana,emocje,wartosciowane,stopien_nacechowania
afektowany,1,przymiotnik,Nie,złość;smutek,błąd;nieużyteczność,"- s","Denerwował mnie jeg
afektowany,1,przymiotnik,Nie,złość;smutek,błąd;nieużyteczność,"- s","Daj sobie spokój z
afera,2,rzeczownik,Nie,wstręt;złość;zaskoczenie,niewiedza;błąd,"- m","Zrobili wielką afe
afera,2,rzeczownik,Nie,wstręt;złość;zaskoczenie,niewiedza;błąd,"- m","Gdy wyciekło nagra
```

```
awantura,2,rzeczownik,Nie,wstręt;złość,błąd;krzywda,"- m","Straszna awantura tam była
awantura,2,rzeczownik,Nie,wstręt;złość,błąd;krzywda,"- m","W sejmie trwa awantura o u
awanturnictwo,1,rzeczownik,Nie,złość;zaskoczenie,nieszczęście;błąd;krzywda,"- m","Pio
awanturnictwo,1,rzeczownik,Nie,złość;zaskoczenie,nieszczęście;błąd;krzywda,"- m","Tad
awaria,1,rzeczownik,Nie,złość;smutek,nieszczęście;nieużyteczność,"- s","Przyczyną opó
awaria,1,rzeczownik,Nie,złość;smutek,nieszczęście;nieużyteczność,"- s","Przez awarię
awersja,1,rzeczownik,Nie,wstręt;strach,nieszczęście;brzydota,"- m","Od przedszkola ma
awersja,1,rzeczownik,Nie,wstręt;złość,nieszczęście;brzydota,"- m","Miał awersję do ki
```

```
nastawienie emocjonalne = {- m, - s, 0, +s, +m}
- m <-- mocno negatywne,
- s <-- słabo negatywne,
0 <-- neutralne,
+ s <-- słabo pozytywne,
+ m <-- mocno pozytywne
[ ] <-- nawias zawiera przykład użycia
```

Tylko formy podstawowe słów

Analiza wydźwięku

- Przykład w R (pakiet SentimentAnalysis)
  - na podstawie: https://stackoverflow.com/questions/56516317/sentiment-analysis-in-r-using-tdm-dtm

```
> documents
[1] "Wow I really like the new light sabers"
[2] "That book was excellent"
[3] "R is a fantastic language"
[4] "The service in this restaurant was miserable"
[5] "This is neither positive or negative"
[6] "The waiter forget about my dessert -- what poor service"
```

```
# A tibble: 6 x 4
    doc negative positive polarity
  <dbl>    <dbl>    <dbl>    <dbl>
1     1        0        2        2
2     2        0        1        1
3     3        0        1        1
4     4        1        0       -1
5     5        1        1        0
6     6        1        0       -1
```

Dokumenty 4 i 5 mają wydźwięk negatywny, 5 neutralny a pozostałe są pozytywne. Wynik jest niewątpliwie „po ludzku” właściwy

Systemy rekomendacyjne

Systemy rekomendacyjne

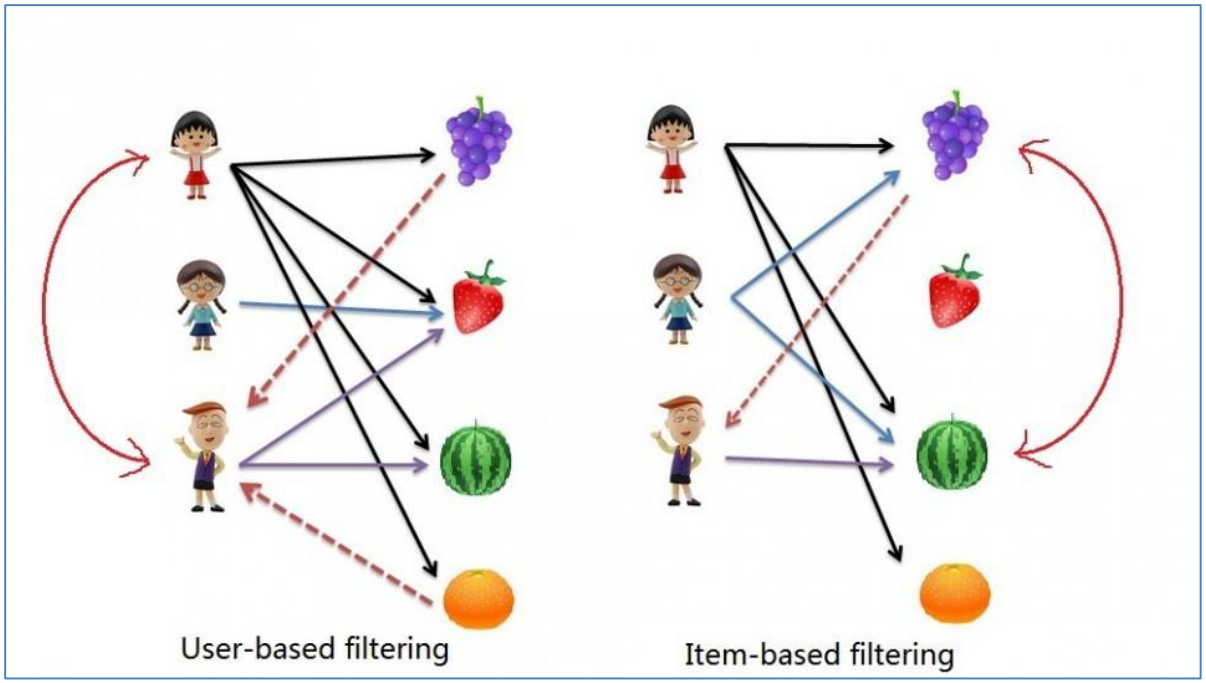
- Zamiast wstępu, skąd my to znamy?
  - „Osoby, które kupiły XXX kupiły też YYY oraz ZZZ”
  - „Rekomendujemy dla Ciebie”
  - „Ostatnio przeglądane”
  - „Inni klienci oglądali również”

Systemy rekomendacyjne

- Dwa podstawowe podejścia
  - Content Based Filtering: rekomenduje produkty podobne do tych, które użytkownik już ocenił pozytywnie lub kupił
  - Collaborative Filtering (CF, społeczne filtrowanie): rekomendacje na podstawie ocen i zakupów innych użytkowników. Gdy dwóch użytkowników kupuje podobne produkty, możemy założyć, że mają zbliżone upodobania i polecać im wzajemnie sprawdzone przez nich produkty („wykorzystanie mądrości tłumu"). CF dzieli się na:
    - ✓ item-based: bazuje tylko na wystawianych przez kupujących ocenach (np. w skali od 1 do 5), wyszukuje podobne przedmioty i je rekomenduje
    - ✓ user-based: wyobraźmy sobie, że mamy użytkownika X. Dla tego użytkownika znajdujemy grupę innych użytkowników, którzy są do siebie podobni, np. zgodnie oceniają te same filmy. Jest to tzw. sąsiedztwo użytkownika X. W tej grupie znajdujemy zbiór produktów, które również są przez nią ocenione, a użytkownik X ich nie oglądał i rekomendujemy mu je

Źródło: https://blog.consdata.tech/2018/08/07/algorytmy-rekomendacyjne-przyklad-implementacji-w-pythonie.html

Systemy rekomendacyjne, item-based vs. user-based



Odkryliśmy, że Ola i Kuba są podobnymi użytkownikami. Zakładamy, że mają oni podobne upodobania. Skoro Ola kupiła pomarańcze i winogron, to i Kubie można je też zarekomendować

Odkryliśmy, że winogrono i arbuz są podobnymi pozycjami. Kubie można więc zarekomendować kupno podobnego towaru, do tego który kupił wcześniej

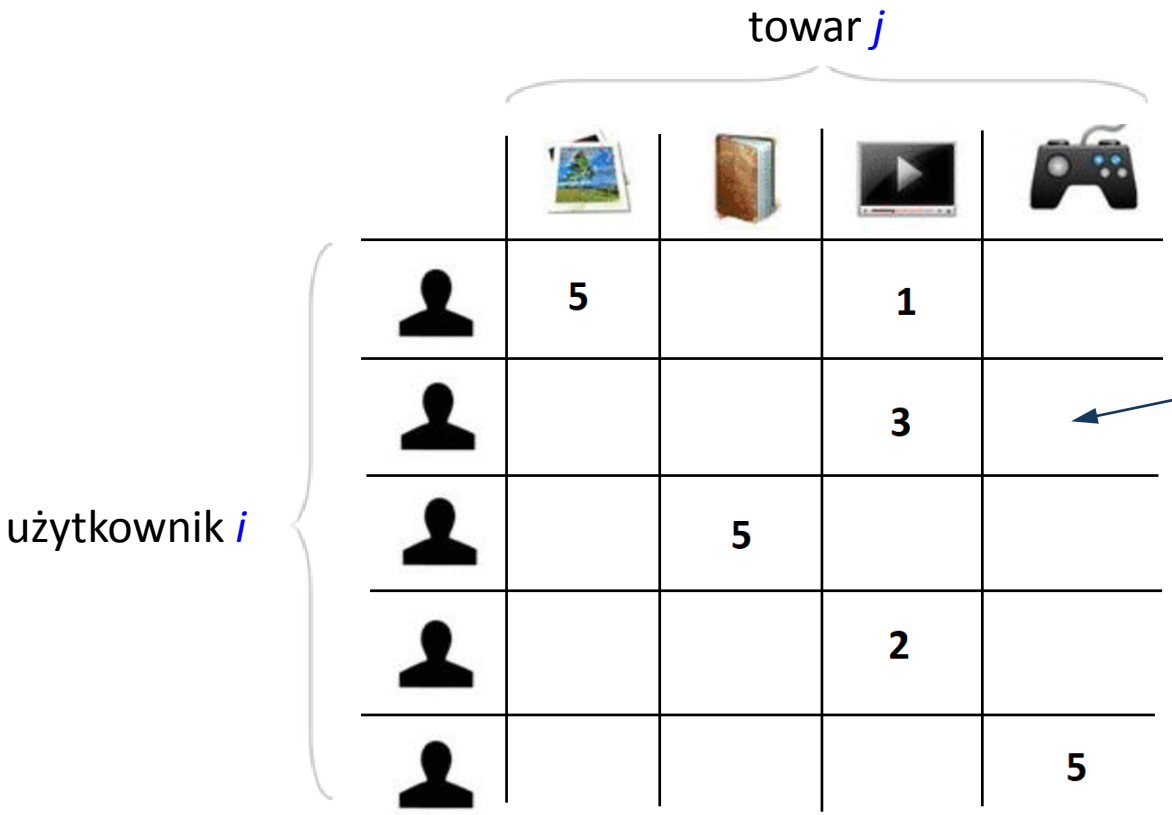
Źródło: https://medium.com/@cfpinela/recommender-systems-user-based-and-item-based-collaborative-filtering-5d5f375a127f

Systemy rekomendacyjne, item-based vs. user-based

- Oba podejścia są technicznie rzecz biorąc podobne
- Item-based jest w praktyce bardziej przydatny, gdyż:
  - towary (items) są „prostsze” niż użytkownicy, łatwiej je podzielić na logiczne grupy. Z ludźmi jest trudniej, mają oni bardzo różne upodobania i są po prostu bardzo różnorodni. Trudniej automatycznie połączyć ich sensowne grupy. A taki jest wymóg w metodzie user-based

Systemy rekomendacyjne

- Collaborative Filtering (CF), macierz ocen



Mogą być różne skale, np:
a) od 1 do 5
b) 0 lub 1
c) od -5 do 5

Macierz jest w praktyce bardzo rzadka

Systemy rekomendacyjne, CF

- Przykładowa, bardzo prosta macierz ocen (w skali 1-5)

	Film 1	Film 2	Film 3	Film 4	Film 5	Film 6	Film 7
Użytkownik A	4			5	1		
Użytkownik B	5	5	4				
Użytkownik C				2	4	5	
Użytkownik D		3					3

- Rozważmy użytkowników *x* oraz *y* oraz ich wektory rankingowe *r(x)* oraz *r(y)*
- Potrzebujemy pewnej miary podobieństwa sim(*x*, *y*)
- Intuicyjnie czujemy, że sim(A,B) > sim (A,C)
  - bo ocenili bardzo podobnie film 1 i całkiem różnie filmy 4 oraz 5

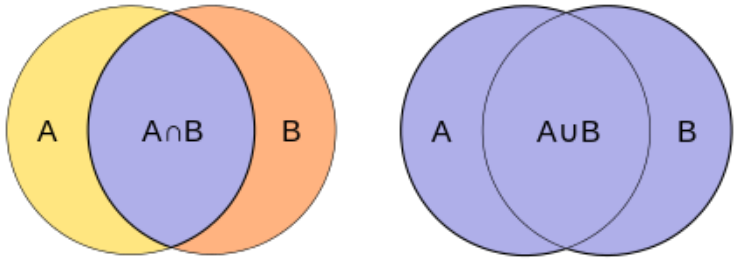
Systemy rekomendacyjne, CF

- Przykładowa, bardzo prosta macierz ocen (w skali 1-5)

	Film 1	Film 2	Film 3	Film 4	Film 5	Film 6	Film 7
Użytkownik A	4			5	1		
Użytkownik B	5	5	4				
Użytkownik C				2	4	5	
Użytkownik D		3					3

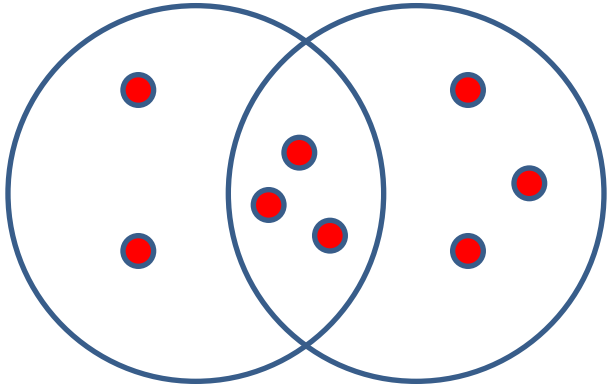
- Pierwszy pomysł: miara (współczynnik) Jacarda
  - mierzy podobieństwo między dwoma zbiorami i jest zdefiniowany jako iloraz części wspólnej zbiorów i sumy tych zbiorów
  - jeśli współczynnik Jaccarda przyjmuje wartości bliskie zeru, zbiory są od siebie różne, gdy jest bliski 1, zbiory są do siebie podobne

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

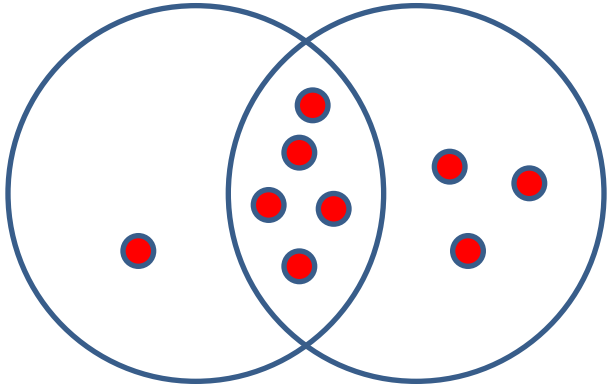


Systemy rekomendacyjne, CF

- współczynnik Jaccarda: przykład



$$J(A,B) = \frac{3}{8}$$



$$J(A,B) = \frac{5}{8}$$

Systemy rekomendacyjne, CF

- Przykładowa, bardzo prosta macierz ocen (w skali 1-5)

	Film 1	Film 2	Film 3	Film 4	Film 5	Film 6	Film 7
Użytkownik A	4			5	1		
Użytkownik B	5	5	4				
Użytkownik C				2	4	5	
Użytkownik D		3					3

  - miara (współczynnik) Jaccarda

$$sim(A,B) = 1/5$$
$$sim(A,C) = 2/4$$
$$sim(A,B) < sim(A,C)$$

A oraz B ocenili w sumie 5 filmów, czyli $|A \cup B| = 5$
A oraz B tylko 1 film ocenili obaj, więc $|A \cap B| = 1$

A oraz C ocenili w sumie 4 filmy, czyli $|A \cup B| = 4$
A oraz C tylko 2 film ocenili obaj, więc $|A \cap B| = 2$

  - podstawowa niedogodność: miara Jaccarda całkowicie ignoruje wartość oceny! Bierze tylko pod uwagę fakt ocenienia filmu

Systemy rekomendacyjne, CF

- Przykładowa, bardzo prosta macierz ocen (w skali 1-5)

	Film 1	Film 2	Film 3	Film 4	Film 5	Film 6	Film 7
Użytkownik A	4	0	0	5	1	0	0
Użytkownik B	5	5	4	0	0	0	0
Użytkownik C	0	0	0	2	4	5	0
Użytkownik D	0	3	0	0	0	0	3

  - miara kosinusowa

$$sim(A,B) = 0{,}38$$
$$sim(A,C) = 0{,}32$$
$$sim(A,B) > sim(A,C)$$

A ,B bardziej podobne do A, C ale różnica jest bardzo niewielka, nieoddająca odbioru subiektywnego

  - podstawowa niedogodność: trzeba przyjąć, że filmy bez wystawionej oceny mają ocenę 0, czyli faktycznie bardzo negatywną. To bardzo fałszuje rzeczywistość!

Systemy rekomendacyjne, CF

- Przykładowa, bardzo prosta macierz ocen (w skali 1-5)

	Film 1	Film 2	Film 3	Film 4	Film 5	Film 6	Film 7	Średnia po wierszach
Użytkownik A	**2/3**	0	0	**5/3**	**-7/3**	0	0	10/3
Użytkownik B	**1/3**	**1/3**	**-2/3**	0	0	0	0	14/3
Użytkownik C	0	0	0	**-5/3**	**1/3**	**4/3**	0	11/3
Użytkownik D	0	**0**	0	0	0	0	**0**	6/2

  - miara kosinusowa ale na danych wycentrowanych wokół zera
  - po wycentrowaniu uzyskaliśmy to, że oceny każdego użytkownika zostały wycentrowane wokół zera. Zero oznacza teraz średnią ocenę. Oceny > 0 oznaczają „powyżej średniej” a oceny < 0 oznaczają „poniżej średniej”

$$sim(A,B) = 0{,}09$$

$$sim(A,C) = -0{,}56$$

$$sim(A,B) > sim(A,C)$$

Od każdej wartości oryginalnych ocen odejmujemy średnią po wierszach, np.
4 - 10/3 = 2/3
1 - 10/3 = -7/3 (wartość ujemna)
Wartości zerowych NIE ZMIENIAMY

Systemy rekomendacyjne, CF

- Przykładowa, bardzo prosta macierz ocen (w skali 1-5)

	Film 1	Film 2	Film 3	Film 4	Film 5	Film 6	Film 7	Średnia po wierszach
Użytkownik A	**2/3**	0	0	**5/3**	**-7/3**	0	0	10/3
Użytkownik B	**1/3**	**1/3**	**-2/3**	0	0	0	0	14/3
Użytkownik C	0	0	0	**-5/3**	**1/3**	**4/3**	0	11/3
Użytkownik D	0	**0**	0	0	0	0	**0**	6/2

  - różnica między A,B oraz A,C jest teraz znaczna i bardziej odpowiada subiektywnym odczuciom
  - wartość ujemna (0,56) można interpretować tak, że użytkownicy A oraz C wydali przeciwstawne opinie
  - centrowanie dodatkowo powoduje, że zmniejszamy wpływ ocen użytkowników, którzy permanentnie oceniają filmy zawsze źle (1) lub zawsze bardzo dobrze (5)
  - wycentrowane podobieństwo kosinusowe to znana powszechnie korelacja Pearsona

Systemy rekomendacyjne, CF user-based

- Predykcja ocen poszczególnych użytkowników
  - niech *r(x)* będzie wektorem ocen użytkownika *x*
  - niech *k* będzie zbiorem użytkowników najbardziej podobnych (ang. neighborhood) do użytkownika *x*, którzy ocenili film *i*
  - Uwaga: użytkownik *x* nie ocenił filmu *i*. My chcemy estymować tą ocenę (wyliczyć jakoś) i tą estymację chcemy wykonać wykorzystując oceny innych użytkowników, którzy są najbardziej podobni do użytkownika *x*
  - najprostsze podejście: wyliczenie średniej oceny tych *k* użytkowników
    - ✓ *r(x,i)* ocena filmu *i* przez użytkownika *x*
    - ✓ *r(y,i)* ocena filmu *i* przez użytkownika *y*
  - podejście jest co prawda bardzo proste ale ignoruje zupełnie stopień podobieństwa pomiędzy użytkownikami

$$r(x,i) = \frac{1}{k}\sum_{y=1}^{k} r(y,i)$$

Systemy rekomendacyjne, CF user-based

- Opcja 1: (jak na poprzednim slajdzie)

$$r(x,i) = \frac{1}{k}\sum_{y=1}^{k} r(y,i)$$

Tutaj też można wykonać centrowanie wokół zera, jak opcja 3

- Opcja 2: jak wyżej ale uwzględniamy współczynniki podobieństwa pomiędzy użytkownikami *sim(x,y)*

$$r(x,i) = \frac{\sum_{y=1}^{k} r(y,i)\cdot sim(x,y)}{\sum_{y=1}^{k} sim(x,y)}$$

- Opcja 3: jak wyżej ale uwzględniamy współczynniki podobieństwa pomiędzy użytkownikami *sim(x,y)* oraz wykonujemy centrowanie wokół zera

$$r(x,i) = \bar{r}(x) + \frac{\sum_{y=1}^{k} \left(r(y,i) - \bar{r}(y)\right)\cdot sim(x,y)}{\sum_{y=1}^{k} sim(x,y)}$$

Tą wersję będziemy zdecydowanie preferować w praktyce

Systemy rekomendacyjne, CF user-based przykład w R (pakiet recommenderlab)

```
m
    item
user i1 i2 i3 i4 i5 i6 i7 i8
  u1 NA  4  4  2  1  2 NA NA
  u2  3 NA NA NA  5  1 NA NA
  u3  3 NA NA  3  2  2 NA  3
  u4  4 NA NA  2  1  1  2  4
  u5  1  1 NA NA NA NA NA  1
  u6 NA  1 NA NA  1  1 NA  1
  u7 NA NA  4  3 NA  1 NA  5

# Konwersja do klasy realRatingMatrix
r <- as(m, "realRatingMatrix")
getRatingMatrix(r)
7 x 8 sparse Matrix of class "dgCMatrix"
    item
user i1 i2 i3 i4 i5 i6 i7 i8
  u1  .  4  4  2  1  2  .  .
  u2  3  .  .  .  5  1  .  .
  u3  3  .  .  3  2  2  .  3
  u4  4  .  .  2  1  1  2  4
  u5  1  1  .  .  .  .  .  1
  u6  .  1  .  .  1  1  .  1
  u7  .  .  4  3  .  1  .  5
```

```
# Podział na dane treningowe (znane) i testowe
r.train <- r[1:6,]
r.test <- r[7,]

getRatingMatrix(r.train)
6 x 8 sparse Matrix of class "dgCMatrix"
    item
user i1 i2 i3 i4 i5 i6 i7 i8
  u1  .  4  4  2  1  2  .  .
  u2  3  .  .  .  5  1  .  .
  u3  3  .  .  3  2  2  .  3
  u4  4  .  .  2  1  1  2  4
  u5  1  1  .  .  .  .  .  1
  u6  .  1  .  .  1  1  .  1

getRatingMatrix(r.test)
1 x 8 sparse Matrix of class "dgCMatrix"
    item
user i1 i2 i3 i4 i5 i6 i7 i8
  u7  .  .  4  3  .  1  .  5
```

Systemy rekomendacyjne, CF user-based przykład w R (pakiet recommenderlab)

```
# Zobaczmy, jak będą wyglądały dane po normalizacji metodą "center"
# UWAGA: robiąc analizy nie musimy tego wykonywać jawnie.
# Tutaj pokazujemy dane po normalizacji tylko dla demonstracji
r.norm <- normalize(r,  method = "center")

getRatingMatrix(r.norm)
7 x 8 sparse Matrix of class "dgCMatrix"
    item
user    i1  i2   i3      i4     i5     i6      i7    i8
  u1 .     1.4 1.40 -0.6000 -1.600 -0.600  .       .
  u2 0.000 .    .      .       2.000 -2.000  .       .
  u3 0.400 .    .      0.4000 -0.600 -0.600  .       0.400
  u4 1.667 .    .     -0.3333 -1.333 -1.333 -0.3333 1.667
  u5 0.000 0.0  .      .       .      .      .       0.000
  u6 .     0.0  .      .       0.000  0.000  .       0.000
  u7 .     .    0.75 -0.2500  .     -2.250  .       1.750

# Możemy też wykonać denormalizację i wracamy oczywiście do macierzy r
getRatingMatrix(denormalize(r.norm))
7 x 8 sparse Matrix of class "dgCMatrix"
    item
user i1 i2 i3 i4 i5 i6 i7 i8
  u1  .  4  4  2  1  2  .  .
  u2  3  .  .  .  5  1  .  .
  u3  3  .  .  3  2  2  .  3
  u4  4  .  .  2  1  1  2  4
  u5  1  1  .  .  .  .  .  1
  u6  .  1  .  .  1  1  .  1
  u7  .  .  4  3  .  1  .  5
```

Systemy rekomendacyjne, CF user-based przykład w R (pakiet recommenderlab)

```
# Wyznaczamy wektor podobieństwa użytkownika u7 do pozostałych użytkowników
# Uwaga jak wyżej

sim <-
    similarity(
      normalize(r.test,  method = "center"),
      normalize(r.train,  method = "center"),
      method = "cosine",
      which = "users"
    )

sim
u1     u2     u3     u4     u5     u6
u7 0.6531 1.0000 0.8264 0.9707 0.0000 0.0000
```

Systemy rekomendacyjne, CF user-based przykład w R (pakiet recommenderlab)

```
# Budujemy nasz finalny model rekomendacyjny.
# nn = 25, czyli bierzemy do obliczeń wszystkich użytkowników (od u1 do u6)
# Tutaj decydujemy o tym, że będziemy pracować na danych znormalizowanych
# oraz, że nie uwzględniamy wartości podobieństw między u7 a pozostałymi użytkownikami
r.model <-
+   Recommender(
+     data = r.train,
+     method = "UBCF",
+     parameter = list(method = "cosine",
+                      nn = 25,
+                      sample = FALSE,
+                      weighted = FALSE,
+                      normalize = "center",
+                      min_matching_items = 0,
+                      min_predictive_items = 0,
+                      verbose = TRUE)
+   )

r.model
Recommender of type 'UBCF' for 'realRatingMatrix'
learned using 6 users.
```

Systemy rekomendacyjne, CF user-based przykład w R (pakiet recommenderlab)

```
# No i finalnie dokonujemy predykcji (wyliczamy rekomendacje) dla uzytkownika u7
recom <- predict(object = r.model, newdata = r.test, type = "ratings", n = 100)
as(recom, "matrix")

      i1    i2 i3 i4    i5 i6    i7 i8
u7 3.767 3.717 NA NA 2.943 NA 2.917 NA


# Możemy też wyliczyć rekomendacje dla pozycji, które de facto
# u7 sam ocenił i są one znane.
# Te wyliczone rekomendacje są zwykle nieco inne niż te wpisane
# jawnie przez użytkownika u7
recom <- predict(object = r.model, newdata = r.test, type = "ratingMatrix", n = 100)
as(recom, "matrix")
      i1    i2   i3    i4    i5    i6    i7    i8
u7 3.767 3.717 4.65 3.072 2.943 2.343 2.917 3.767
```

Systemy rekomendacyjne, CF user-based przykład w R (pakiet recommenderlab)

```
# Sprawdzenie "ręczne" kilku pozycji. Wyświetlamy sobie potrzebne dane

getRatingMatrix(normalize(r.train,  method = "center"))
6 x 8 sparse Matrix of class "dgCMatrix"
    item
user     i1  i2  i3      i4     i5     i6      i7    i8
  u1 .      1.4 1.4 -0.6000 -1.600 -0.600  .       .
  u2 0.000  .   .     .       2.000 -2.000  .       .
  u3 0.400  .   .     0.4000 -0.600 -0.600  .       0.400
  u4 1.667  .   .    -0.3333 -1.333 -1.333 -0.3333 1.667
  u5 0.000  0.0 .     .       .      .      .       0.000
  u6 .      0.0 .     .       0.000  0.000  .       0.000

as(recom, "matrix")
      i1    i2   i3    i4    i5    i6    i7    i8
u7 3.767 3.717 4.65 3.072 2.943 2.343 2.917 3.767

# Ręcznie sprawdzamy sobie wyniki
# Rekomendacja dla u7 dla pozycji i1
(0.000 + 0.400 + 1.667 + 0.000) / 4 + (13/4)
[1] 3.767

# Ręcznie sprawdzamy sobie wyniki
# Rekomendacja dla u7 dla pozycji i5
(-1.600 + 2.000 - 0.600 - 1.333 + 0.000) / 5 + (13/4)
[1] 2.943
```

$$r(x,i) = \bar{r}(x) + \frac{1}{k}\sum_{y=1}^{k}(r(y,i) - \bar{r}(y))$$

Systemy rekomendacyjne, CF user-based przykład w R (pakiet recommenderlab)

```
# Przykład jak wyżej ale dla opcji weighted = TRUE
r.model <-
  Recommender(
    data = r.train,
    method = "UBCF",
    parameter = list(method = "cosine",
                     nn = 25,
                     sample = FALSE,
                     weighted = TRUE,
                     normalize = "center",
                     min_matching_items = 0,
                     min_predictive_items = 0,
                     verbose = TRUE)
  )
sim
       u1     u2     u3     u4     u5     u6
u7 0.6531 1.0000 0.8264 0.9707 0.0000 0.0000

recom2 <- predict(object = r.model, newdata = r.test, type = "ratings", n = 100)
as(recom2, "matrix")
      i1   i2 i3 i4    i5 i6    i7 i8
u7 3.947 4.65 NA NA 3.008 NA 2.917 NA

# Ręcznie sprawdzamy sobie wyniki
# Rekomendacja dla u7 dla pozycji i5
(-1.600 * 0.6531 + 2.000 * 1.0000 - 0.600 * 0.8264 - 1.333 * 0.9707 + 0.000 * 0.0000) /
+ (0.6531 + 1.0000 + 0.8264 + 0.9707 + 0.0000 + 0.0000) + (13/4)
[1] 3.008
```

$$r(x,i) = \bar{r}(x) + \frac{\sum_{y=1}^{k} \left(r(y,i) - \bar{r}(y)\right) \cdot sim(x,y)}{\sum_{y=1}^{k} sim(x,y)}$$

Systemy rekomendacyjne, CF item-based

- Procedura
  - znajdź pozycje (items) podobne do pozycji *i*
  - wylicz (estymuj) dla pozycji i ranking opierając się na dostępnych innych rankingach (innych użytkowników) tychże podobnych pozycjach
- Przykładowo, chcąc wyznaczyć ranking dla filmu 4 dla użytkownika D musimy
  - najpierw znaleźć filmy podobne do filmu 4
  - następnie wykorzystując informacje o tym, jak te podobne filmy ocenili inni użytkownicy, wyliczymy ranking *rank(film 4, użytkownik D)*

	Film 1	Film 2	Film 3	Film 4	Film 5	Film 6	Film 7
Użytkownik A	4	0	0	5	1	0	0
Użytkownik B	5	5	4	0	0	0	0
Użytkownik C	0	0	0	2	4	5	0
Użytkownik D	0	3	0	0	0	0	3

Systemy rekomendacyjne, CF item-based

- Formuła matematyczna jest tak sama, jak dla user-based, inne są znaczenia poszczególnych składników
  - *k* – zbiór pozycji (items) podobnych do siebie

$$r(x,i) = \bar{r}(x) + \frac{\sum_{y=1}^{k} \left(r(y,i) - \bar{r}(y)\right) \cdot sim(x,y)}{\sum_{y=1}^{k} sim(x,y)}$$

Systemy rekomendacyjne, CF item-based przykład w R (pakiet recommenderlab)

```
m
item
user  i1 i2 i3 i4 i5 i6
  u1   1 NA  2 NA NA  1
  u2  NA NA  4  2 NA NA
  u3   3  5 NA  4  4  3
  u4  NA  4  1 NA  3 NA
  u5  NA NA  2  5  4  3
  u6   5 NA NA NA  2 NA
  u7  NA  4  3 NA NA NA
  u8  NA NA NA  4 NA  2
  u9   5 NA  4 NA NA NA
  u10 NA  2  3 NA NA NA
  u11  4  1  5  2  2  4
  u12 NA  3 NA NA  5 NA
```

```
# Konwersja do klasy realRatingMatrix

r <- as(m, "realRatingMatrix")
getRatingMatrix(r)

12 x 6 sparse Matrix of class "dgCMatrix"
     item
user  i1 i2 i3 i4 i5 i6
  u1   1  .  2  .  .  1
  u2   .  .  4  2  .  .
  u3   3  5  .  4  4  3
  u4   .  4  1  .  3  .
  u5   .  .  2  5  4  3
  u6   5  .  .  .  2  .
  u7   .  4  3  .  .  .
  u8   .  .  .  4  .  2
  u9   5  .  4  .  .  .
  u10  .  2  3  .  .  .
  u11  4  1  5  2  2  4
  u12  .  3  .  .  5  .
```

Systemy rekomendacyjne, CF item-based przykład w R (pakiet recommenderlab)

```
# Zobaczmy, jak będą wyglądały dane po normalizacji metodą "center"
# UWAGA: robiąc analizy nie musimy tego wykonywać jawnie.
# Tutaj pokazujemy dane po normalizacji tylko dla demonstracji

r.norm <- normalize(r,  method = "center")
getRatingMatrix(r.norm)

12 x 6 sparse Matrix of class "dgCMatrix"
     item
user       i1     i2      i3   i4      i5      i6
  u1  -0.3333  .      0.6667  .    .       -0.3333
  u2   .       .      1.0000 -1.0  .        .
  u3  -0.8000  1.200  .       0.2  0.2000 -0.8000
  u4   .       1.333 -1.6667  .    0.3333  .
  u5   .       .     -1.5000  1.5  0.5000 -0.5000
  u6   1.5000  .      .       .   -1.5000  .
  u7   .       0.500 -0.5000  .    .        .
  u8   .       .      .       1.0  .       -1.0000
  u9   0.5000  .     -0.5000  .    .        .
  u10  .      -0.500  0.5000  .    .        .
  u11  1.0000 -2.000  2.0000 -1.0 -1.0000  1.0000
  u12  .      -1.000  .       .    1.0000  .
```

Systemy rekomendacyjne, CF item-based przykład w R (pakiet recommenderlab)

```
# Wyznaczamy współczynniki podobieństwa poszczególnych POZYCJI (ITEMS)
# Uwaga jak wyżej

sim <-
+     similarity(
+       normalize(r,  method = "center", row = TRUE),
+       method = "cosine",
+       which = "items"
+     )

       i1     i2     i3     i4     i5
i2 0.9910
i3 0.6044 0.9945
i4 0.8882 0.9417 0.9458
i5 0.9532 0.4006 0.9430 0.8689
i6 1.0000 0.9910 0.8374 0.8264 0.9030
```

Systemy rekomendacyjne, CF item-based przykład w R (pakiet recommenderlab)

```
# Budujemy nasz finalny model rekomendacyjny.
# k= 30, czyli bierzemy do obliczeń wszystkie pozycje (od i1 do i6)
# Tutaj decydujemy o tym, że będziemy pracować na danych znormalizowanych
# metodą 'center'

r.model <-
  Recommender(
    data = r,
    method = "IBCF",
    parameter = list(method = "cosine",
                     k = 30,
                     normalize = "center",
                     normalize_sim_matrix = FALSE,
                     alpha = 0.5,
                     na_as_zero = FALSE,
                     verbose = TRUE)
  )

r.model
Recommender of type 'IBCF' for 'realRatingMatrix'
learned using 12 users.
```

Systemy rekomendacyjne, CF item-based przykład w R (pakiet recommenderlab)

```
# No i finalnie dokonujemy predykcji (wyliczamy rekomendacje) dla użytkownika u5
# Zwróćmy uwagę jakie podajemy parametry do funkcji predict()

recom <- predict(object = r.model, data = r, newdata = 5, type = "ratings", n = 100)
as(recom, "matrix")

      i1    i2 i3 i4 i5 i6
u5 3.617 3.388 NA NA NA NA


# Możemy też wyliczyć rekomendacje dla pozycji, które de facto
# u5 sam ocenił i są one znane.
# Te wyliczone rekomendacje są zwykle nieco inne niż te wpisane
# jawnie przez użytkownika u7

recom <- predict(object = r.model, data = r, newdata = 5, type = "ratingMatrix", n = 100)
as(recom, "matrix")

      i1    i2   i3    i4    i5    i6
u5 3.617 3.388 4.04 2.971 3.293 3.669
```

Systemy rekomendacyjne, CF item-based przykład w R (pakiet recommenderlab)

```
# Sprawdzenie "ręczne" kilku pozycji. Wyświetlamy sobie potrzebne dane
sim
       i1     i2     i3     i4     i5
i2 0.9910
i3 0.6044 0.9945
i4 0.8882 0.9417 0.9458
i5 0.9532 0.4006 0.9430 0.8689
i6 1.0000 0.9910 0.8374 0.8264 0.9030

getRatingMatrix(r.norm)
12 x 6 sparse Matrix of class "dgCMatrix"
     item
user       i1     i2       i3   i4      i5      i6
  u1  -0.3333   .      0.6667   .     .      -0.3333
  u2   .        .      1.0000 -1.0   .       .
  u3  -0.8000   1.200  .        0.2   0.2000 -0.8000
  u4   .        1.333 -1.6667   .     0.3333  .
  u5   .        .     -1.5000   1.5   0.5000 -0.5000
  u6   1.5000   .      .        .    -1.5000  .
  u7   .        0.500 -0.5000   .     .       .
  u8   .        .      .        1.0   .      -1.0000
  u9   0.5000   .     -0.5000   .     .       .
  u10  .       -0.500  0.5000   .     .       .
  u11  1.0000 -2.000   2.0000 -1.0  -1.0000   1.0000
  u12  .       -1.000  .        .     1.0000  .

as(recom, "matrix")
       i1    i2   i3    i4    i5    i6
u5 3.617 3.388 4.04 2.971 3.293 3.669
```

```
rating(u1, i5) =
(0.6044 * (-1.5) +
 0.8882 * 1.5 +
 0.9532 * 0.5 +
 1.000 * (-0.5)) /
(0.6044 + 0.8882 + 0.9532 + 1.000)
 + 14/4  = 3.617
```

```
rating(u2, i5) =
(0.9945 * (-1.5) +
 0.9417 * 1.5 +
 0.4006 * 0.5 +
 0.9910 * (-0.5)) /
(0.9945 + 0.9417 + 0.4006+ 0.9910)
 + 14/4 = 3.388
```

$$r(x,i) = \bar{r}(x) + \frac{\sum_{y=1}^{k} \left(r(y,i) - \bar{r}(y)\right) \cdot sim(x,y)}{\sum_{y=1}^{k} sim(x,y)}$$

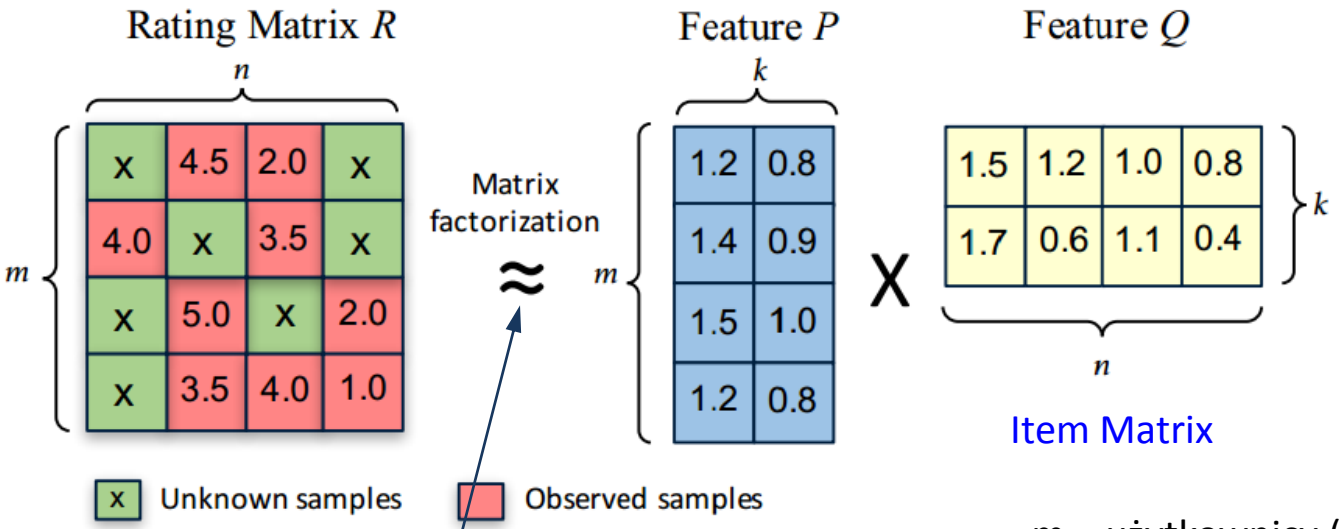
Systemy rekomendacyjne oparte o faktoryzację macierzy

- Geneza: konkurs Netflix Prize z 2006 roku
  - https://en.wikipedia.org/wiki/Netflix_Prize
  - zbiór treningowy: 100.480.507 ocen wystawionych przez 480.189 użytkowników dla 17.770 filmów
  - oceny w skali od 1 do 5 (jedna do pięciu gwiazdek)
  - zadanie: opracować nowy algorytm rekomendacyjny, który zredukuje błąd RMSE (Root Mean Square Error) o więcej niż 10%

$$RMSE = \sqrt{\frac{1}{n}\sum(rating\_actual - predicting\_rating)^2}$$

Systemy rekomendacyjne oparte o faktoryzację macierzy

- Matrix Factorization (MF)
  - poważną niedogodnością metod omówionych wcześniej jest to, że macierz ocen może być bardzo duża i bardzo rzadka
  - MF to sposób na pozbycie się w dużej mierze tych niedogodności



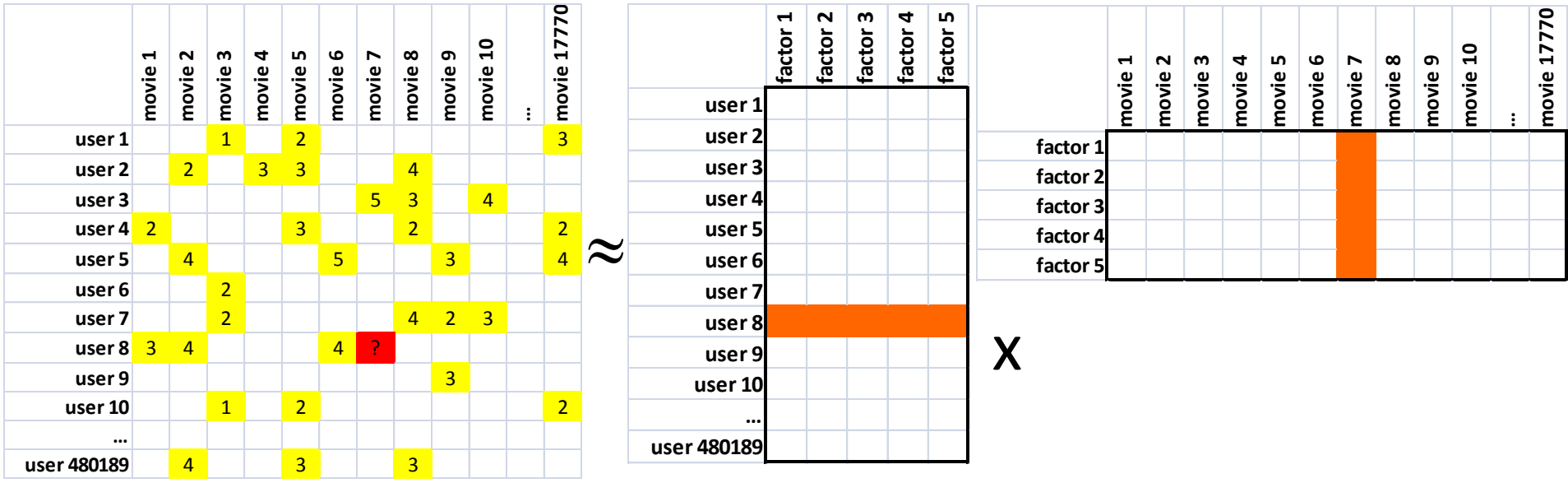
To jest przybliżenie a nie idealna równość !

m – użytkownicy (users)
n – przedmioty (items)
k – cechy ukryte (latent factors), w klasycznej wersji dobierane za pomocą algorytmu Gradient Descent

Źródło: https://arxiv.org/pdf/1610.05838.pdf

Systemy rekomendacyjne oparte o faktoryzację macierzy

- Kolejna wizualizacja



Źródło: https://slideplayer.com/slide/14862607/

Systemy rekomendacyjne oparte o faktoryzację macierzy

- Opis matematyczny (w pewnym uproszczeniu)

$$\mathbf{R} \approx \mathbf{P}^T \mathbf{Q}$$

p_u wektor użytkownika

q_i wektor przedmiotu

Aby dokonać predykcji dla użytkownika *u* oraz przedmiotu *i* obliczamy

$$\hat{r}_{ui} = p_u^T q_i$$

Aby obliczyć macierze P i Q staramy się zminimalizować błąd na obserwacjach, gdzie użytkownik podał swoją ocenę przedmiotu (np. tych na żółto na poprzednim slajdzie). Oznaczmy ten zbiór S

$$E = \sum_{(u,i) \in S} (\hat{r}_{ui} - r_{ui})^2$$

Pojawia się więc problem optymalizacyjny (L(P,Q) to tzw. funkcja straty, ją minimalizujemy)

$$L(\mathbf{P}, \mathbf{Q}) = \min_{\mathbf{P},\mathbf{Q}} \sum_{(u,i) \in S} (\hat{r}_{ui} - r_{ui})^2$$

Systemy rekomendacyjne oparte o faktoryzację macierzy

- Opis matematyczny (w pewnym uproszczeniu)

  Aby uniknąć zbytniego dopasowania (overfitting) do danych uczących (czyli tych ze zbioru S) wprowadza się tzw. regularyzację .

  Regularyzacja to sposób radzenia sobie z nadmiernym dopasowaniem do danych uczących.

  Regularyzację implementuje się go poprzez dodanie do funkcji straty L kosztu związanego z dużymi wartościami wektorów (u nas chodzi o duże wartości wektorów w user matrix oraz item matrix)

Źródło: https://cran.r-project.org/web/packages/recosystem/vignettes/introduction.html

Systemy rekomendacyjne oparte o faktoryzację macierzy

- Opis matematyczny (w pewnym uproszczeniu)

A popular technique to solve the recommender system problem is the matrix factorization method. The idea is to approximate the whole rating matrix $R_{m \times n}$ by the product of two matrices of lower dimensions, $P_{k \times m}$ and $Q_{k \times n}$, such that

$$R \approx P'Q$$

Let p_u be the u-th column of P, and q_v be the v-th column of Q, then the rating given by user u on item v would be predicted as $p'_u q_v$.

A typical solution for P and Q is given by the following optimization problem (Chin, Zhuang, et al. 2015a, 2015b):

$$\min_{P,Q} \sum_{(u,v) \in R} \left[f(p_u, q_v; r_{u,v}) + \mu_P ||p_u||_1 + \mu_Q ||q_v||_1 + \frac{\lambda_P}{2} ||p_u||_2^2 + \frac{\lambda_Q}{2} ||q_v||_2^2 \right]$$

where (u, v) are locations of observed entries in R, $r_{u,v}$ is the observed rating, f is the loss function, and $\mu_P, \mu_Q, \lambda_P, \lambda_Q$ are penalty parameters to avoid overfitting.

regularyzacja **L1** - koszt jest dodawany proporcjonalnie do bezwzględnej wartości współczynników wag (normy L1 wag)

regularyzacja **L2** - koszt jest dodawany proporcjonalnie do kwadratu wartości współczynników wag (normy L2 wag)

Źródło: https://cran.r-project.org/web/packages/recosystem/vignettes/introduction.html

Systemy rekomendacyjne oparte o faktoryzację macierzy

- Przykład w R

```
# Nasza minimalistyczna demonstracyjna macierz user-item
    i1 i2 i3 i4
u1   5  3  .  1
u2   4  .  .  1
u3   1  1  .  5
u4   1  .  .  4
u5   .  1  5  4

# W takim formacie (w pliku tekstowym) muszą być zapisane dane dla recosystem
1 1 5
1 2 3
1 4 1
2 1 4
2 4 1
3 1 1
3 2 1
3 4 5
4 1 1
4 4 4
5 2 1
5 3 5
5 4 4
```

Systemy rekomendacyjne oparte o faktoryzację macierzy

- Przykład w R

```
train_file = data_file("train_file_MF.txt", index1 = TRUE)
# W pliku test_file są pozycje, których nie ma w train_file
test_file = data_file("test_file_MF.txt", index1 = TRUE)
head(read.table(train_file@source, header = FALSE), 20)
head(read.table(test_file@source, header = FALSE), 20)

# Stworzenie obiektu klasy "RecoSys"
r = Reco()
# Algorytm jest losowy, więc żeby zapewnić powtarzalność wyników
set.seed(1)
# Strojenie (tuning). Wybór najlepszych parametrów z podanych zbiorów kandydujących
# UWAGA!!! when nthread > 1, the training result is NOT guaranteed to be reproducible,
# even if a random seed is set.
# https://statr.me/2016/07/recommender-system-using-parallel-matrix-factorization/
res = r$tune(train_file,
         opts = list(
           dim = c(10),                 # the number of latent factors
           costp_l1 = c(0, 0.1),        # the L1 regularization cost for user factors
           costp_l2 = c(0.01, 0.1),     # the L2 regularization cost for user factors
           costq_l1 = c(0, 0.1),        # the L1 regularization cost for item factors
           costq_l2 = c(0.01, 0.1),     # the L2 regularization cost for item factors
           lrate = c(0.01, 0.1),        # the learning rate, which can be thought of
                                        # as the step size in gradient descent.
           nthread = 1,                 # the number of threads for parallel computing
           niter = 20                   # the number of iterations
         ))
```

Systemy rekomendacyjne oparte o faktoryzację macierzy

- Przykład w R

```
# Trenowanie modelu. Parametry do trenowania zostały wyznaczone w poprzednim kroku
r$train(train_file, opts = c(res$min, nthread = 1, niter = 20))

# Trenowanie, wszystkie parametry domyślne
# r$train(train_file)

P_file = out_file("P.txt")
Q_file = out_file("Q.txt")
r$output(P_file, Q_file)
P <- read.table(P_file@dest, header = FALSE, sep = " ")

V1     V2     V3     V4      V5      V6     V7     V8     V9    V10
1 0.2324 0.2768 0.2805 0.4087 0.24860 0.33779 0.2679 0.3432 0.2235 0.2348
2 0.3859 0.3631 0.2353 0.3904 0.38410 0.04111 0.2884 0.4518 0.1530 0.1169
3 0.3625 0.4545 0.1096 0.3442 0.31322 0.48515 0.2576 0.3720 0.3572 0.4691
4 0.3298 0.1097 0.2903 0.2103 0.02940 0.51231 0.1476 0.3940 0.4342 0.2539
5 0.3773 0.4700 0.3553 0.2408 0.09615 0.38063 0.1820 0.2357 0.5337 0.4105

Q <- read.table(Q_file@dest, header = FALSE, sep = " ")
      V1      V2     V3     V4      V5     V6      V7     V8     V9    V10
1 0.4666 0.28140 0.1295 0.4832 0.34004 0.1120 0.21095 0.3626 0.2631 0.3532
2 0.2691 0.08659 0.4356 0.2035 0.08729 0.3504 0.28345 0.3790 0.3100 0.3264
3 0.2970 0.35459 0.1449 0.2570 0.25144 0.1555 0.06991 0.3277 0.5132 0.1493
4 0.4314 0.51088 0.1368 0.1561 0.07378 0.5618 0.30178 0.4522 0.4345 0.5899
```

Dane demonstracyjne są bardzo minimalistyczne, więc tutaj *de facto* dane są powiększone (do aż 10-ciu latent factors)

Systemy rekomendacyjne oparte o faktoryzację macierzy

- Przykład w R

```
# Predykcja i zapisanie wyniku predykcji do pliku
r$predict(test_file, out_file("predict_test.txt"))
prediction output generated at predict_test.txt
print(scan("predict_test.txt", n = 10))p
Read 7 items
[1] 3.3247 1.9546 2.6208 3.4403 0.5026 2.9667 3.0540

# Predykcję możemy też zrobić dla danych train i porównać z danymi rzeczywistymi
# (tymi z pliku train_file_MF.txt). Wyniki nie będą identyczne ale powinny być podobne,
# bo jednak R jest tylko PRZYBLIŻENIEM P * Q
r$predict(train_file, out_file("predict_train.txt"))
prediction output generated at predict_train.txt
print(scan("predict_train.txt", n = 20))
Read 13 items
[1] 4.9751 2.5595 1.0118 3.8067 0.9941 1.0888 0.6723 4.7931 0.9564 3.8866 1.4629 4.6287 4.1126
# po zaokrągleniu do liczby całkowitej
round(print(scan("predict_train.txt", n = 20)), 0)
[1] 5 3 1 4 1 1 1 5 1 4 1 5 4
print(read.table(train_file@source, header = FALSE, sep = " ")$V3)
[1] 5 3 1 4 1 1 1 5 1 4 1 5 4
```

Predykcja dla danych testowych

Tutaj akurat jest idealnie, bo wynik predykcji jest w 100% zgodny z wynikami z pliku treningowego

Systemy rekomendacyjne oparte o faktoryzację macierzy

- Przykład w R

```
# UWAGA!!!
# Wykonaj obliczenia dla set.seed(2). Wynik będzie bardzo odbiegający od oczekiwań.
# To jest jednak algorytm probabilistyczny i widać mamy tutaj do czynienia
# z czymś w rodzaju utykania w minimum lokalnym.
# Otrzymamy takie wyniki:
#
print(scan("predict_train.txt", n = 13))
Read 13 items
[1] 1.1601 0.7396 1.2131 0.8586 0.9329 1.0066 0.7629 1.4453 0.9886 1.2209 0.7665 0.9999
1.4237
round(print(scan("predict_train.txt", n = 13)), 0)
[1] 1 1 1 1 1 1 1 1 1 1 1 1 1
print(read.table(train_file@source, header = FALSE, sep = " ")$V3)
[1] 5 3 1 4 1 1 1 5 1 4 1 5 4
```

Tym razem jest fatalnie

```
# Dane dostępne w pakiecie recosystem
train_file = data_file(system.file("dat", "smalltrain.txt", package = "recosystem"))
test_file = data_file(system.file("dat", "smalltest.txt", package = "recosystem"))
```

Modele wektorowe tekstów, word embeddings

Modele wektorowe tekstów, word embeddings

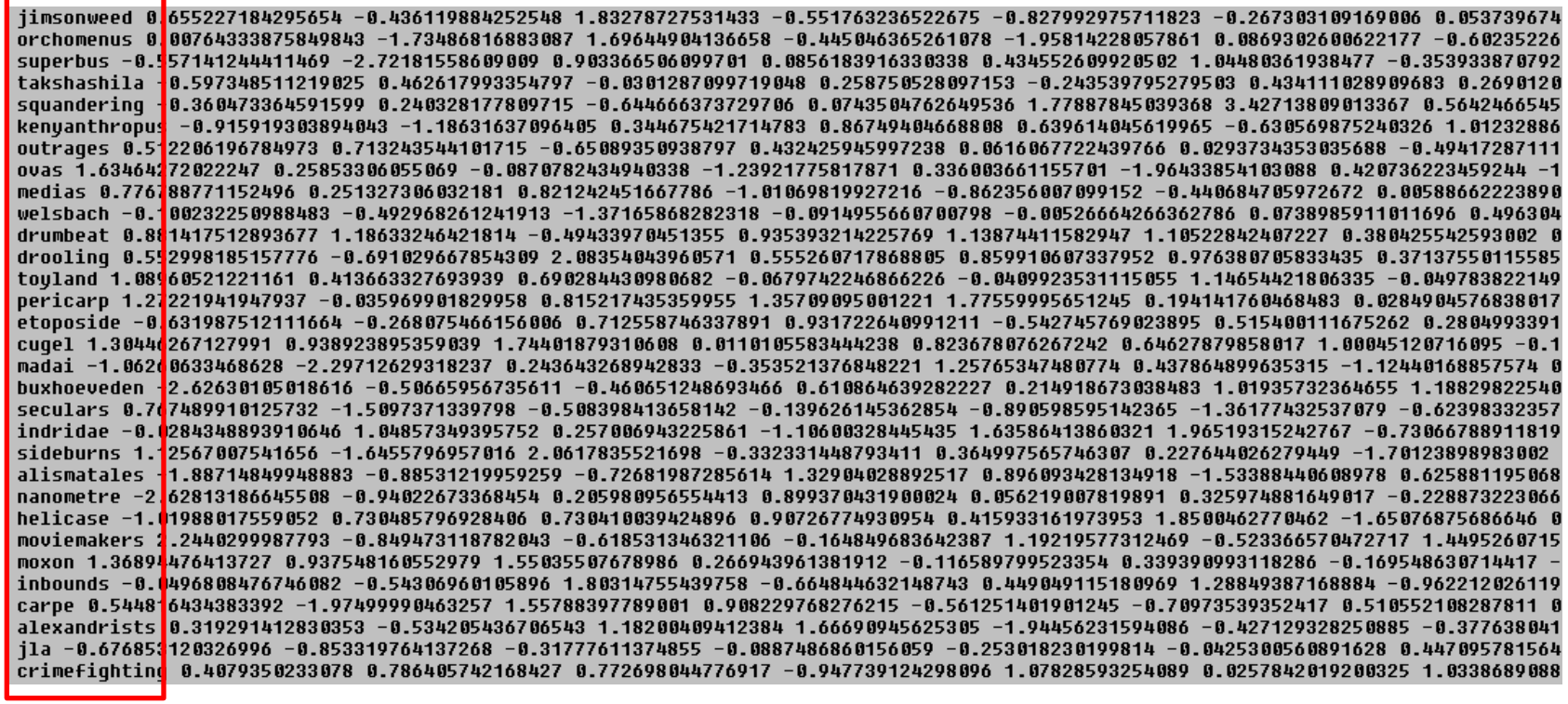
- word embeddings – dość trudno to zgrabnie przetłumaczyć na j. polski. Dosłowne tłumaczenie to mniej więcej „zanurzanie słów”, „osadzanie słów”
- Lepiej chyba nie tłumaczyć dosłownie i przyjąć bardziej wolne tłumaczenie: „reprezentacja wektorowa słów”
- Chodzi tutaj o to, że słowa o podobnym znaczeniu, użyte w podobnym kontekście zostaną zmapowanie do tej samej reprezentacji (tego samego wektora)
- Niektórzy autorzy używają (strasznej) kalki językowej: embeddingi ☹

Modele wektorowe tekstów, word embeddings

- Jedna z technik automatycznego przetwarzania języka naturalnego (*natural language processing*, NLP)
  - rozpoznawanie i przewidywanie kontekstu, relacji między frazami , wieloznaczności, synonimów, polisemii
  - kontekst: znając otoczenie wyrazu zdecydowanie szybciej „złapiemy” jego kontekst
    - ✓ dzisiaj rano była straszna pogoda
    - ✓ dostałem nowy komputer i strasznie się ucieszyłem
- Słowa lub frazy ze słownika są odwzorowane na wektory liczb rzeczywistych
  - dlaczego frazy jako wektory liczb? Bo komputery potrafią operować TYLKO na liczbach (dodatkowo tylko binarnych 0-1)
  - u jej podstaw leży hipoteza, że wyrazy o podobnym znaczeniu występują w podobnych kontekstach
    John Rupert Firth (1957): „You shall know a word by the company it keeps”

Modele wektorowe tekstów, word embeddings

- Przykładowe wektory (mały fragment całości)
  - wymiary: 71291 fraz, 50 wymiarów



Modele wektorowe tekstów, word embeddings

- Celem wizualizacji wektory w przestrzeni wielowymiarowej (u nas 50D) można zrzutować do przestrzeni dwuwymiarowej (2D)
  - PCA (Principal Component Analysis)
  - SNE (Stochastic Neighbour Embedding)
  - LDA (Linear Discriminant Analysis)
  - t-SNE(t-distributed Stochastic Neighbour Embedding)
  - inne

Modele wektorowe tekstów, word embeddings

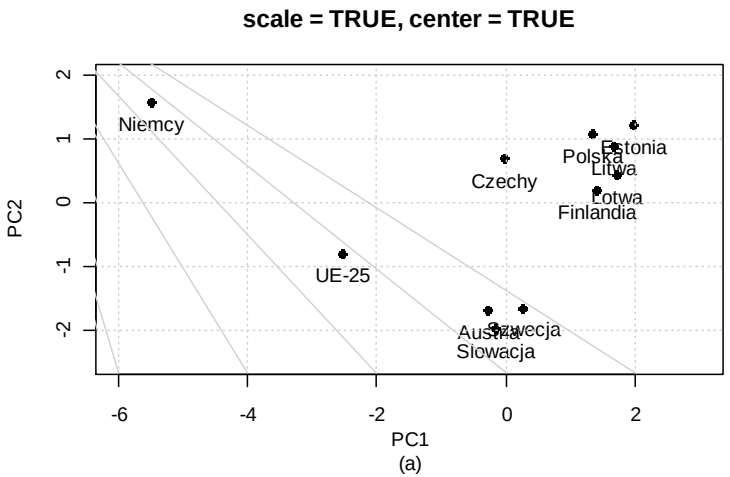
- Przykład redukcji wymiarowości nD→2D
  - zbiór danych *energia*
  - czy można coś powiedzieć o podobieństwie krajów?
  - mając dane w takiej postaci jak poniżej, raczej na powyższe pytanie trudno będzie odpowiedzieć twierdząco
  - Analiza PCA

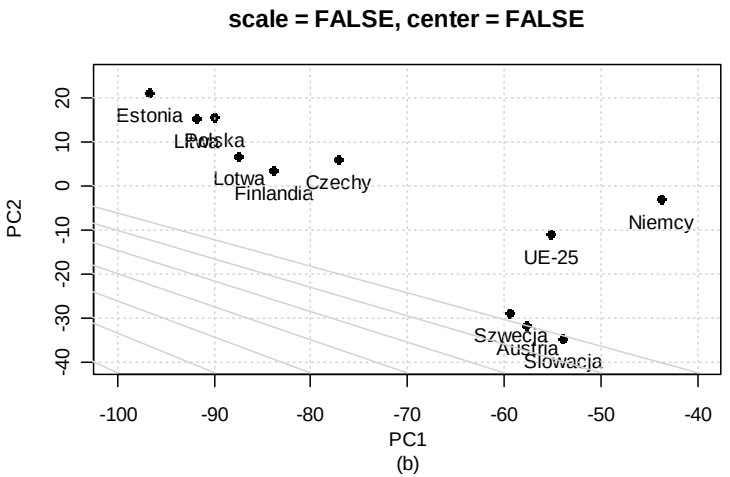
	X1	X2	X3	X4	X5	X6	X7	X8
UE-25	51,3	0,7	21,4	5,4	3,8	4,0	4,7	8,7
Austria	49,5	1,3	43,5	1,6	0,4	0,8	0,5	2,4
Czechy	76,4	0,1	10,2	0,1	2,8	5,6	0	4,8
Estonia	99,0	0	0,3	0,7	0	0	0	0
Finlandia	82,7	0	14,7	0,2	0,5	0	0	1,9
Litwa	92,9	0	5,0	0	0,3	1,4	0,4	0
Łotwa	86,9	0	12,5	0,2	0,3	0,1	0	0
Niemcy	41,3	2,2	10,1	14,0	8,6	13,1	0,8	9,9
Polska	91,2	0	4,1	0,3	1,2	2,6	0,2	0,4
Słowacja	45,1	0	45,2	0,1	0,6	4,1	0,9	4,0
Szwecja	51,7	0	40,7	0,5	0,2	2,1	0	4,8
Wariancja	485,0	0,5	274,0	18,0	6,7	14,5	1,9	12,2

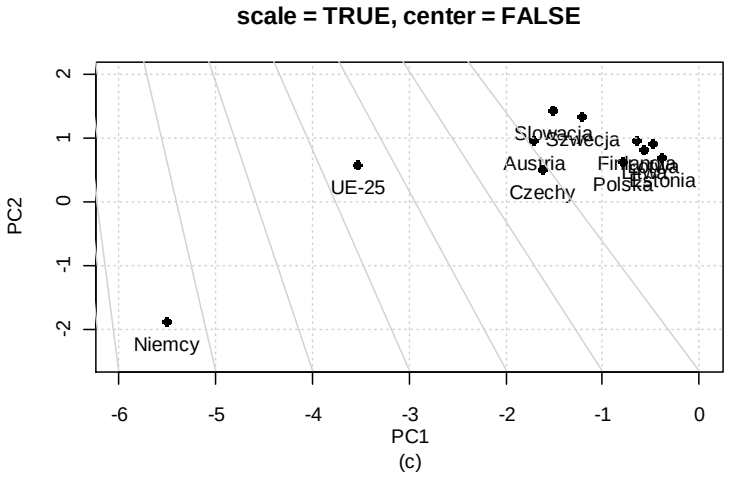
Struktura pozyskania energii w % wg. wybranych źródeł w wybranych krajach UE w 2005 roku

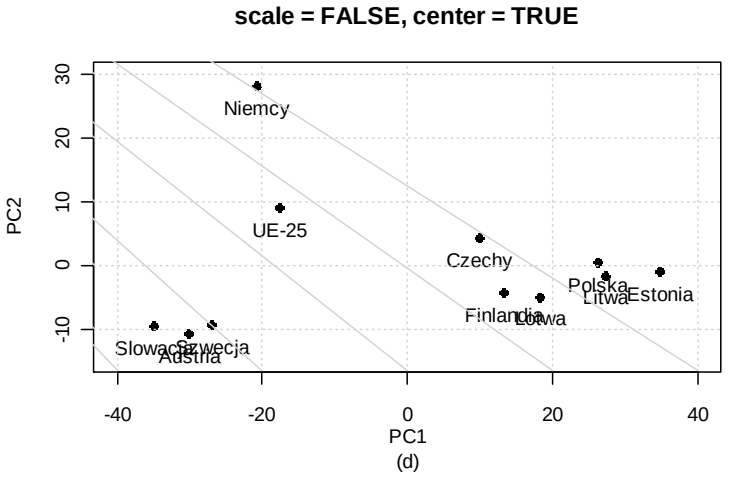
Modele wektorowe tekstów, word embeddings

```
wynik.TT <- prcomp(energia, scale = TRUE , center = TRUE,  retx = TRUE)
plot(wynik.TT$x[,1:2], main = "scale = TRUE, center = TRUE")
text(wynik.TT$x[,1:2], labels = row.names(energia))
```









Modele wektorowe tekstów, word embeddings

- Poprzednio omówiona metoda PCA jest liniowa, co nie zawsze jest właściwe
- MDS – *Multi-Dimensional Scaling*
  - MDS jest metodą nieliniową
  - idea: algorytm stara się tak rozmieścić punkty w przestrzeni o niższej wymiarowości, by odległości między parami punktów były jak najlepiej zachowane
  - działanie:
    - ✓ definiujemy pewną tzw. funkcję celu, której wartość zależy od zachowania odległości między danymi w nowej i w starej przestrzeni
    - ✓ iteracyjnie staramy się zminimalizować funkcję celu

$$E = \frac{1}{\sum_{i<j} D_{ij}} \sum_{i<j} \frac{\left(D_{ij} - d_{ij}\right)^2}{D_{ij}}$$

skalowania Sammona

$$E = \sqrt{\sum_{i<j} \left(D_{ij} - d_{ij}\right)^2 / \sum_{i<j} \left(D_{ij}\right)^2}$$

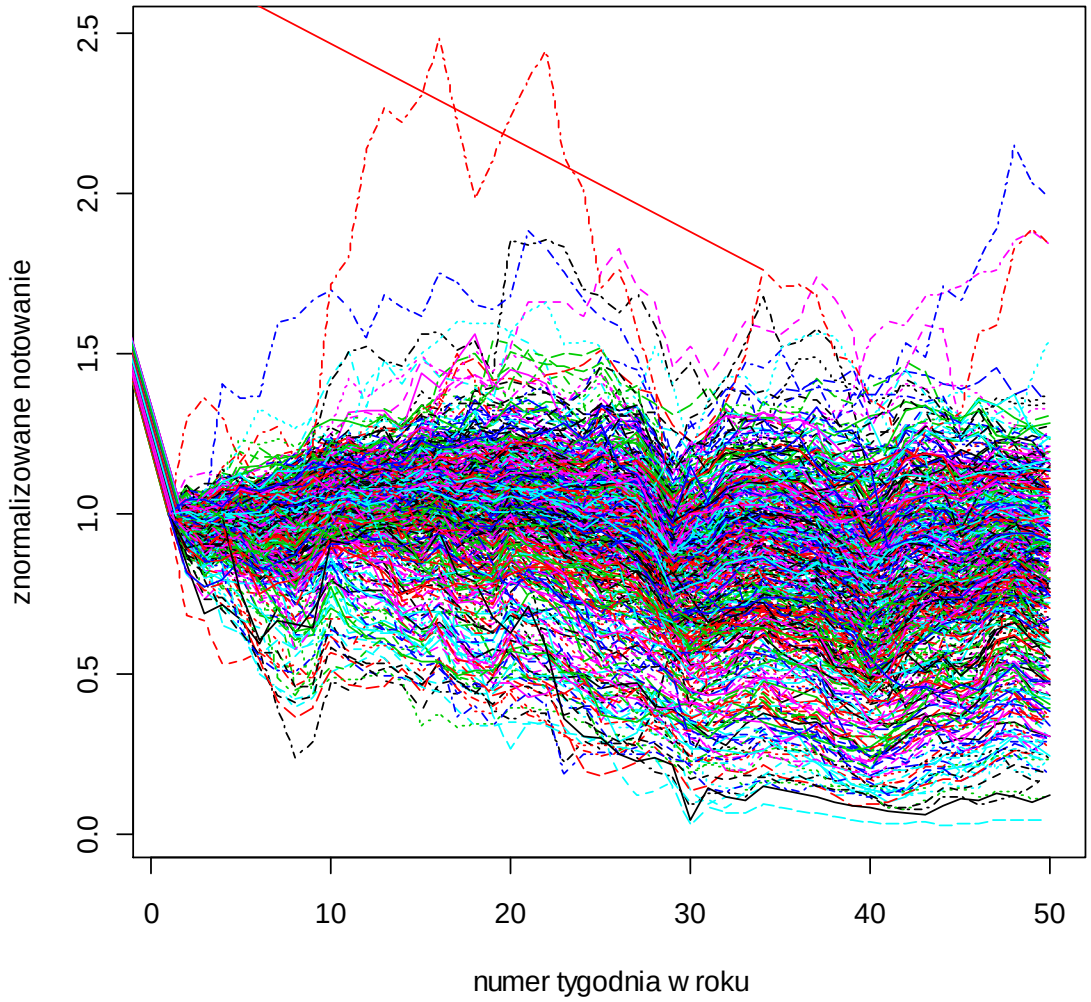
skalowania Kruskala

Dij - odległość pomiędzy i-tym i j-tym obiektem w oryginalnej przestrzeni N-wielowymiarowej.

dij - odległość pomiędzy i-tym i j-tym obiektem w przestrzeni n-wymiarowej

Modele wektorowe tekstów, word embeddings

```
sp500 <- read.table(file = "sp500.data)
matplot(t(sp500), type = "l")
```



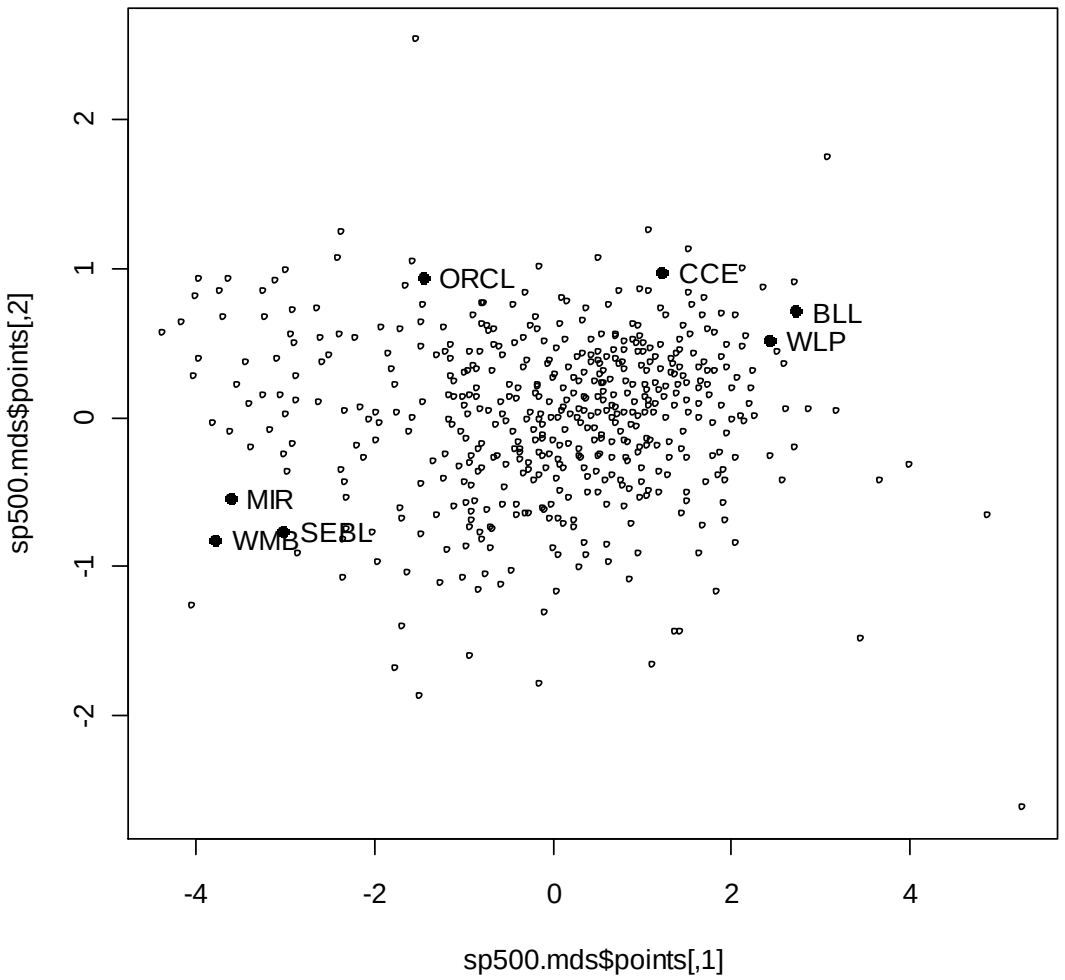
Zbiór danych: SP500

Zadanie:

Pogrupować firmy na te, których notowania w analizowanym okresie rosły, były w miarę równe oraz spadały

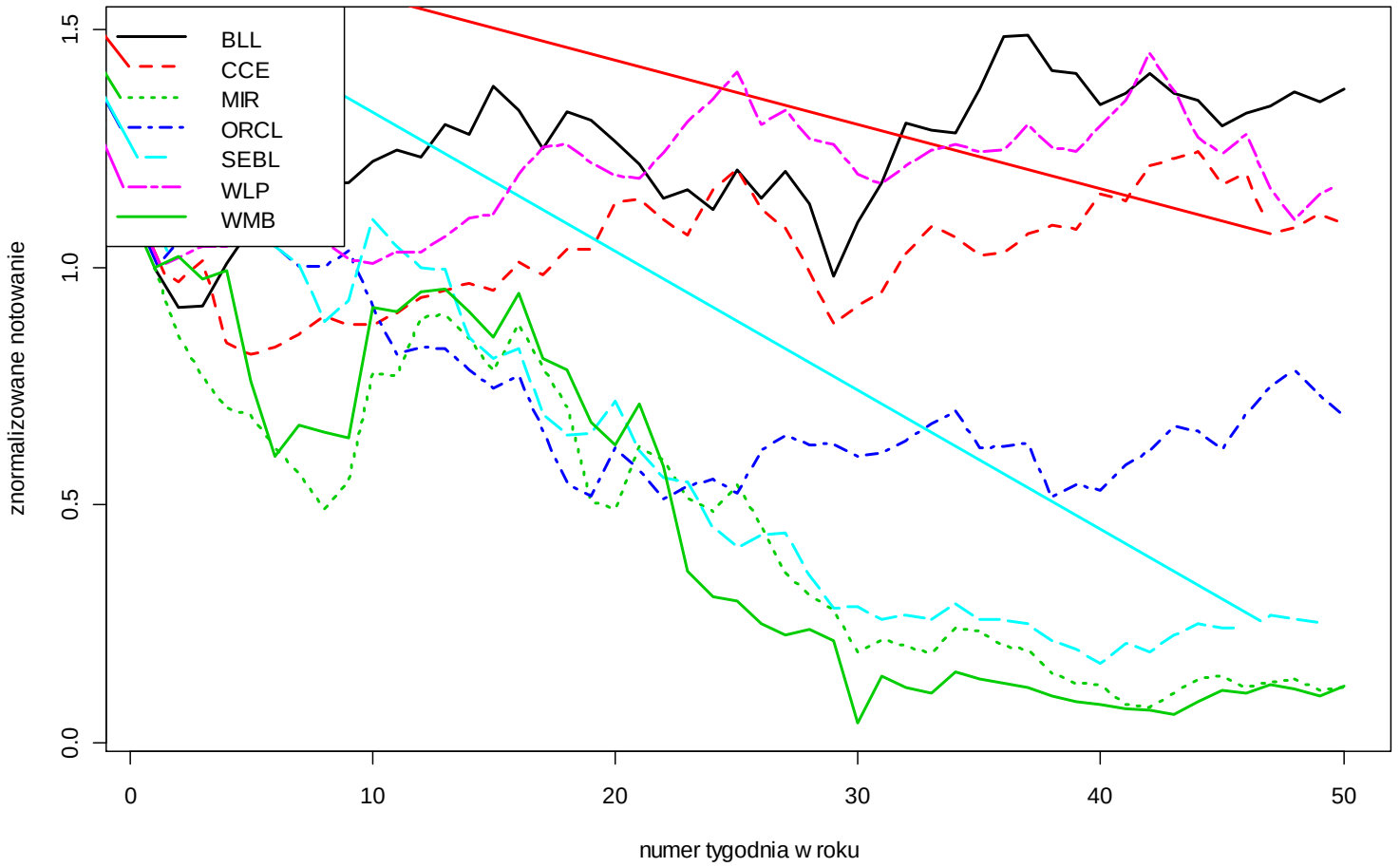
Modele wektorowe tekstów, word embeddings

```
sp500.odl <- dist(sp500, method = "euclidean")
sp500.mds <- sammon(sp500.odl, k = 2)
plot(sp500.mds$points)
```



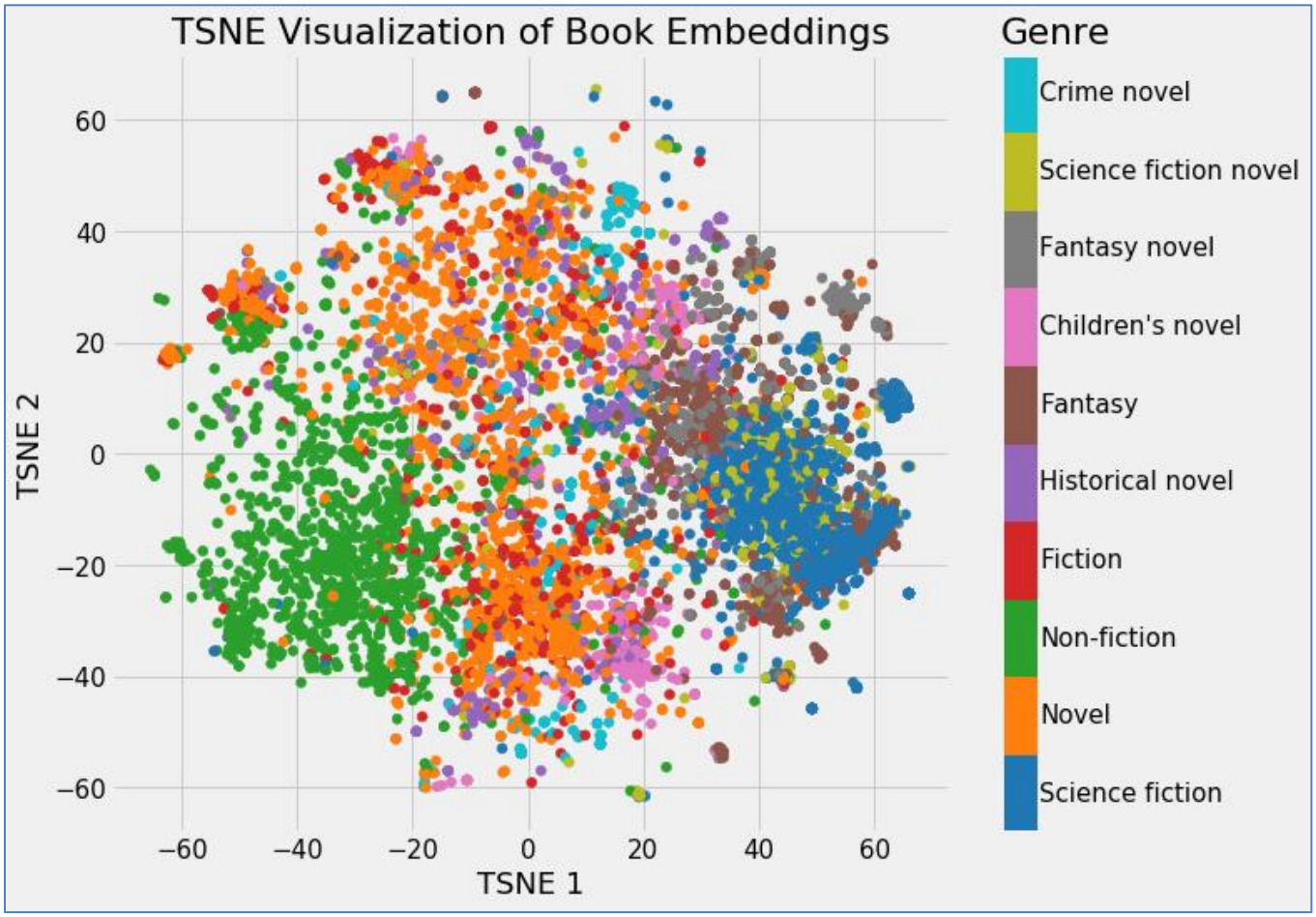
Modele wektorowe tekstów, word embeddings

```
matplot(t(sp500[firmy_num,]), type="l")
legend("topleft", legend = c("BLL", "CCE", "MIR", "ORCL", "SEBL", "WLP", "WMB"))
```



Modele wektorowe tekstów, word embeddings

- Przykładowa wizualizacja wektorów



Modele wektorowe tekstów, word embeddings

- w 2013 roku za sprawą publikacji prac zespołu Google pod przewodnictwem Tomasa Mikolova. Zespół opracował a następnie udoskonalił metodę zapisu znaczenia słów w postaci wektorów w wielowymiarowej przestrzeni (word2vec)
  - podstawowe założenie: znaczenie słowa można wywnioskować po otaczających go innych słowach. Dzięki temu można „podpowiedzieć" słowa o podobnym znaczeniu/kontekście. Skutek: wyszukiwarka może odnaleźć więcej pasujących (relevant) do zapytania dokumentów

Modele wektorowe tekstów, word embeddings

- Kontekst słowa bardzo często (niemal zawsze) odczytamy z otaczających to słowo innych słów

Modele wektorowe tekstów, word embeddings

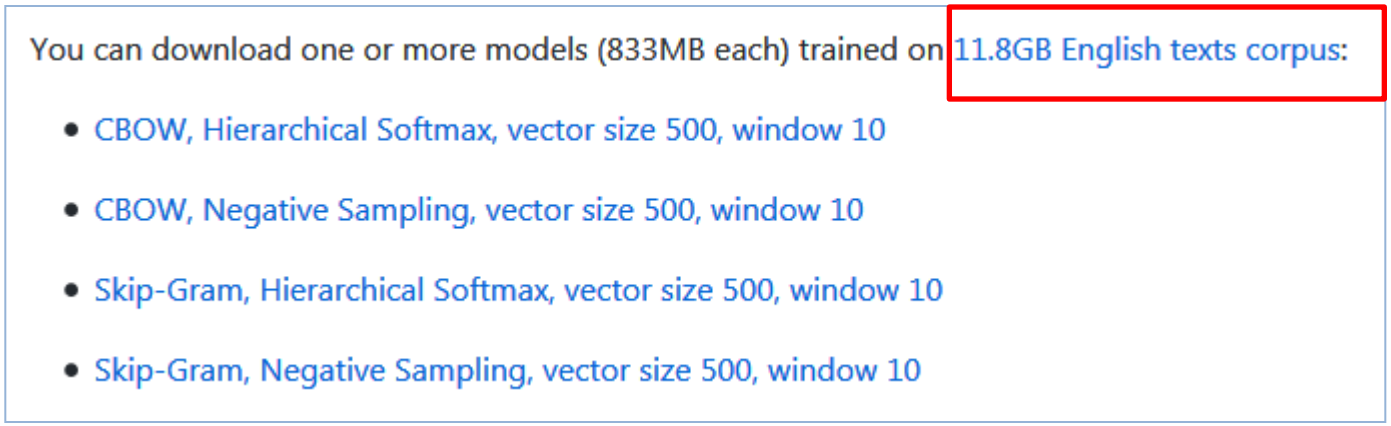
- Intuicje
  - „Człowiek został potrącony ____„
    - ✓ możemy się tylko domyślać końcówki zdania, może chodzić o atobus, pociąg, samochów, rower
  - „Człowiek został potrącony ____autobus”
    - ✓ tu znamy otoczenie poszukiwanego słowa, więc nasze domyślanie bardziej się konkretyzuje. „przez czerwony autobus”, „przez miejski autobus” itd

Modele wektorowe tekstów, word embeddings

- word2vec
  - typowo stosowane są przestrzenie od kilkudziesięciu do kilkuset wymiarów
  - do jego stworzenia stosuje się dwie metody Continuous Bag of Words (CBOW) oraz Continuous skip-gram
  - CBOW: model przewiduje aktualnie uczone słowo na podstawie słów otaczających bez uwzględnienia kolejności (bag of words).
  - skip-gram: model wykorzystuje aktualne słowo do prognozowania słów otaczających, przy czym słowom bliższym przyporządkowana jest wyższa waga
  - według autorów CBOW jest szybszy ale algorytm skip-gram osiąga lepsze rezultaty dla rzadszych słów
  - utworzone wektory podawane są na sieć neuronową, która uczy się wzajemnych powiązań słów oraz ich kontekstu

Modele wektorowe tekstów, word embeddings

- Aby metoda word2vec (oraz inne działające na podobnej zasadzie) działała zadawalająco modele muszą być tworzone na bazie ogromnych korpusów tekstowych



- Sporo różnych modeli (Word Embeddings) powstało już i są ogólnie dostępne w internecie

Modele wektorowe tekstów, word embeddings

- Modele dla języka polskiego też istnieją
  - ze względu na fleksyjność j. polskiego wymagają jednak dużo większych korpusów w porównaniu np. do języka angielskiego
  - https://clarin-pl.eu/dspace/bitstream/handle/11321/442/word_embeddings_pl.txt?sequence=1&isAllowed=y

```
Pre-trained models for Polish, external sources:

1. Word2Vec (IPI PAN)
source: http://dsmodels.nlp.ipipan.waw.pl/
data: NKJP / Wikipedia

- arch: skipgram / CBOW
- types: forms / lemmas
- alg: hierachical SoftMax, Negative Sampling
- dim: 100, 300
```

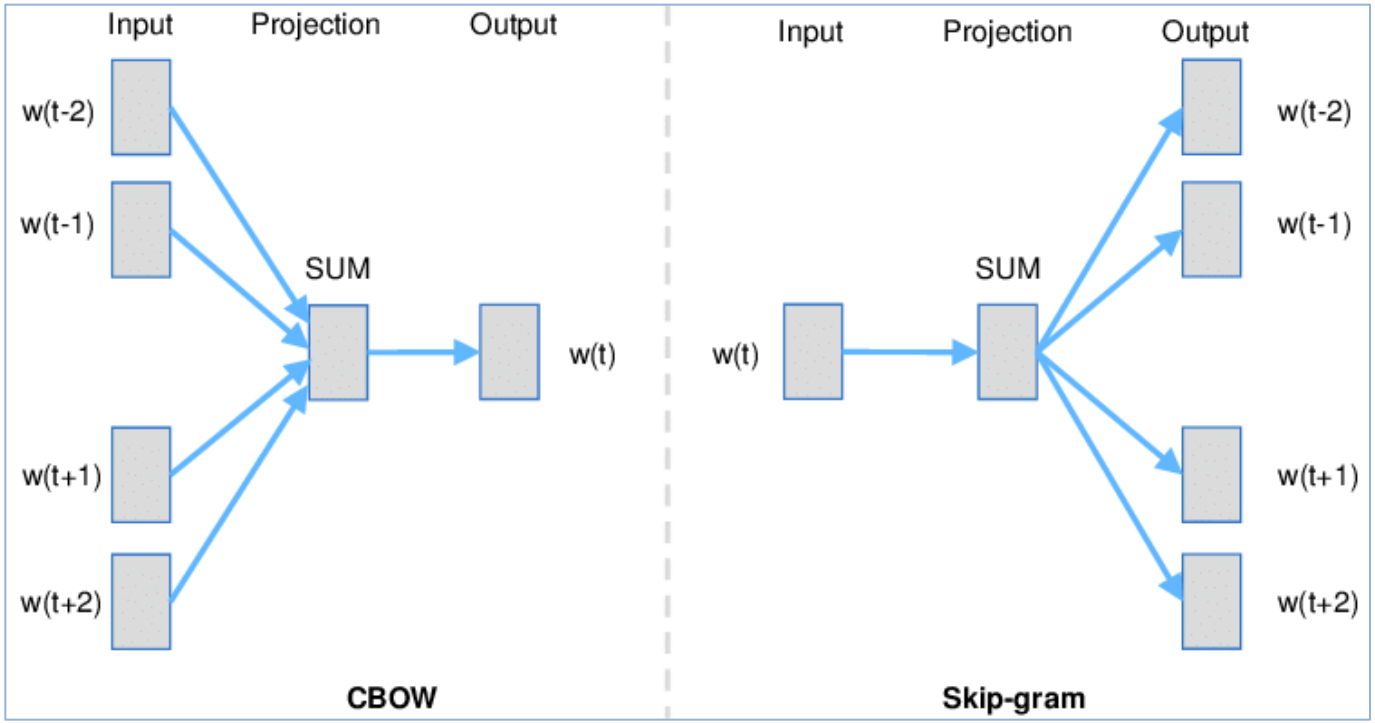
Źródło:	☑ NKJP	☑ Wiki	☑ NKJP+Wiki
Typ modelu:	☑ Formy	☑ Lematy	
Rozmiar wektora:	☑ 100	☑ 300	

#	Nazwa pliku do pobrania	MB
1	nkjp+wiki-forms-all-100-cbow-hs.txt.gz	802.3
2	nkjp+wiki-forms-all-100-cbow-ns.txt.gz	758.6
3	nkjp+wiki-forms-all-100-skipg-hs.txt.gz	780.0
4	nkjp+wiki-forms-all-100-skipg-ns.txt.gz	771.0
5	nkjp+wiki-forms-all-300-cbow-hs-50.txt.gz	634.3
6	nkjp+wiki-forms-all-300-cbow-hs.txt.gz	2312.1

  - modele na bazie
    - ✓ pełnej wersji Narodowego Korpusu Języka Polskiego
    - ✓ polskiej edycji Wikipedii z końca 2016 roku

Modele wektorowe tekstów, word embeddings

- CBOW, skip-gram



Podajemy słowa otaczające (neighboring) dane słowo a system próbuje odgadnąć pojedyncze słowo, które może potencjalnie wystąpić pomiędzy nimi

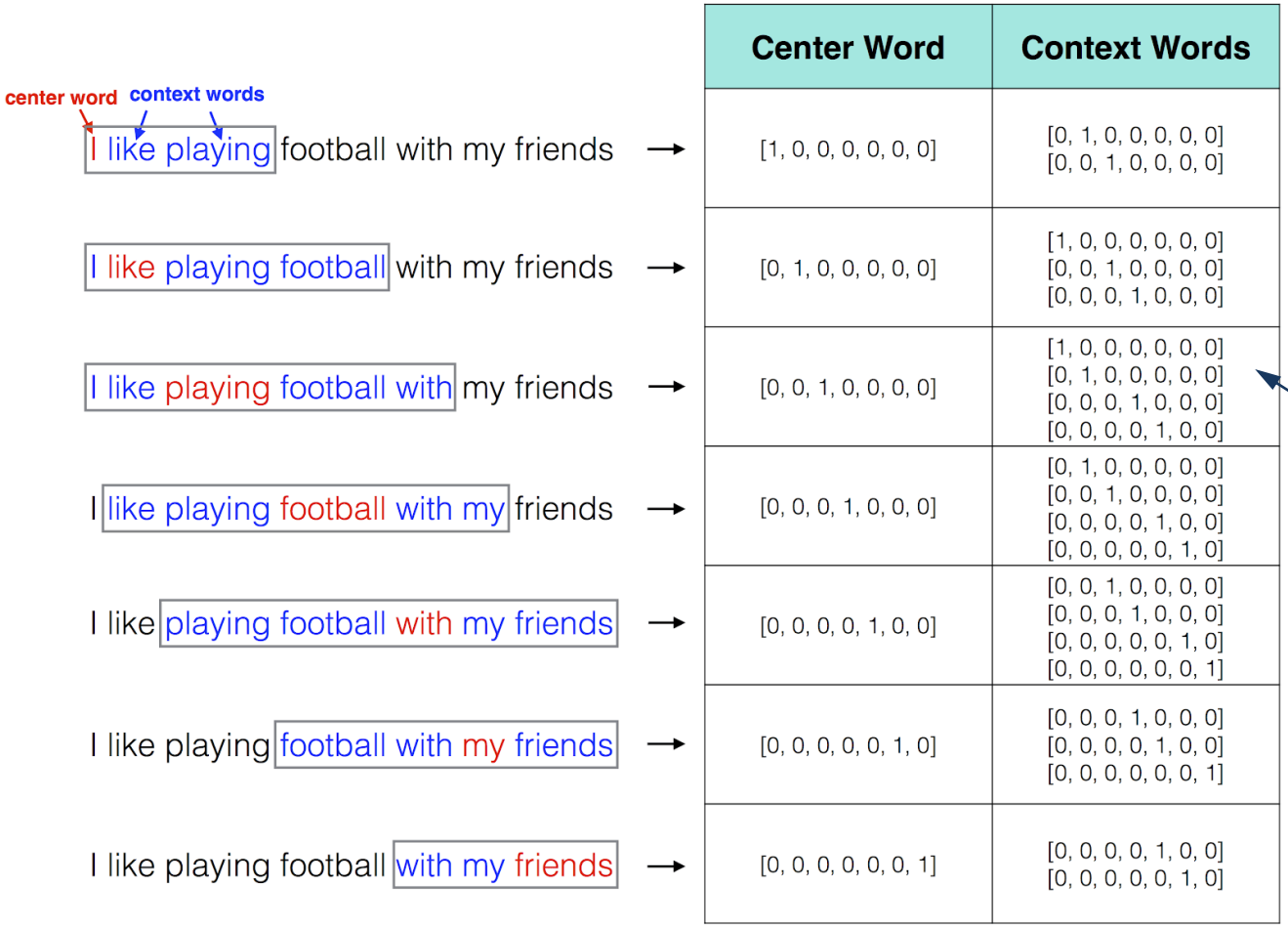
Próbuje odgadnąć sąsiednie słowa przy użyciu bieżącego słowa

Modele wektorowe tekstów, word embeddings

- CBOW (predykcja centralnego wyrazu z otaczającego go kontekstu)
  - kontekst: {dzisiaj, rano, była, pogoda}
  - predykcja: straszna
- skip-gram (predykcja kontekstu otaczającego centralny wyraz)
  - wyraz centralny: straszna
  - predykcja kontekstu (wyrazy otaczające) : {dzisiaj, rano, była, pogoda}

Modele wektorowe tekstów, word embeddings

- Tworzenie wektorów słów, okno przesuwne (sliding window)



Center Word	Context Words
[1, 0, 0, 0, 0, 0, 0]	[0, 1, 0, 0, 0, 0, 0] [0, 0, 1, 0, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 0]	[1, 0, 0, 0, 0, 0, 0] [0, 0, 1, 0, 0, 0, 0] [0, 0, 0, 1, 0, 0, 0]
[0, 0, 1, 0, 0, 0, 0]	[1, 0, 0, 0, 0, 0, 0] [0, 1, 0, 0, 0, 0, 0] [0, 0, 0, 1, 0, 0, 0] [0, 0, 0, 0, 1, 0, 0]
[0, 0, 0, 1, 0, 0, 0]	[0, 1, 0, 0, 0, 0, 0] [0, 0, 1, 0, 0, 0, 0] [0, 0, 0, 0, 1, 0, 0] [0, 0, 0, 0, 0, 1, 0]
[0, 0, 0, 0, 1, 0, 0]	[0, 0, 1, 0, 0, 0, 0] [0, 0, 0, 1, 0, 0, 0] [0, 0, 0, 0, 0, 1, 0] [0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 1, 0]	[0, 0, 0, 1, 0, 0, 0] [0, 0, 0, 0, 1, 0, 0] [0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 0, 1]	[0, 0, 0, 0, 1, 0, 0] [0, 0, 0, 0, 0, 1, 0]

Tzw. kodowanie z gorącą jedynką (one hot encoding)

Każdy wyraz reprezentowany jest jako wektor z jedną „1” i wieloma „0”

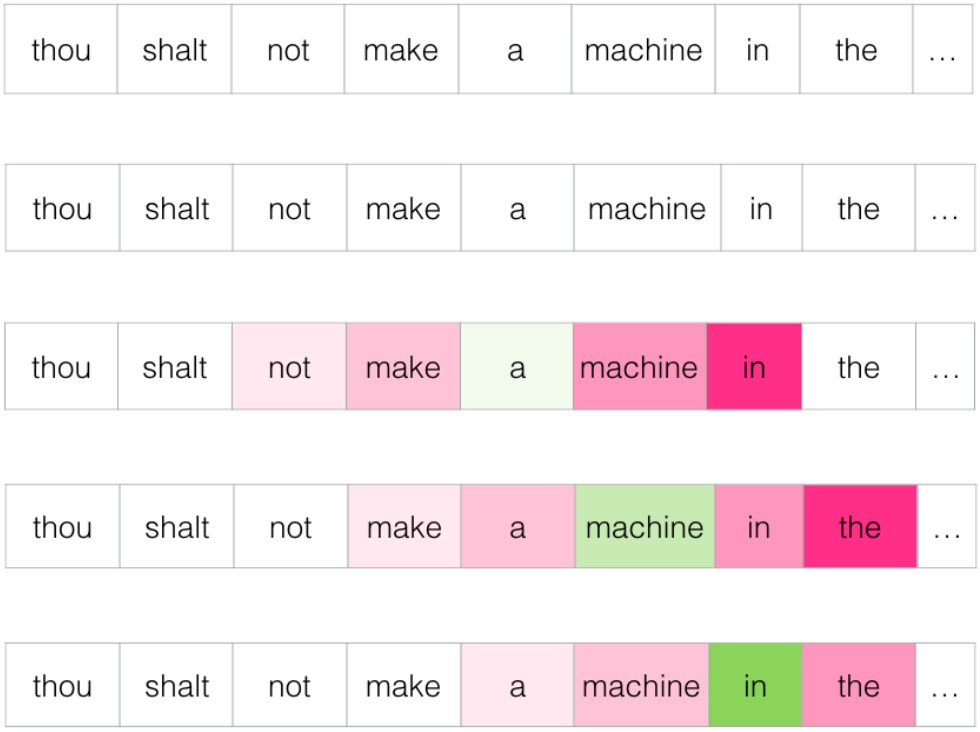
Wymiar wektora jest równy rozmiarowi słownika

W naszym przykładzie słownik ma wymiar 7 (bo amy tylko jedno zdanie z 7 wyrazami)

Modele wektorowe tekstów, word embeddings

- Tworzenie wektorów słów, okno przesuwne (sliding window)

Thou shalt not make a machine in the likeness of a human mind



input word	target word
not	thou
not	shalt
not	make
not	a
make	shalt
make	not
make	a
make	machine
a	not
a	make
a	machine
a	in
machine	make
machine	a
machine	in
machine	the
in	a
in	machine
in	the
in	likeness

Źródło: http://jalammar.github.io/illustrated-word2vec/

Modele wektorowe tekstów, word embeddings

- Kilka słów o rodzajach modeli wektorowych
  - rzadkie
    - ✓ większość elementów to 0
    - ✓ mogą być bardzo długie (długość to dziesiątki, setki tysięcy a nawet więcej)
    - ✓ lepsze w zapytaniach o konkretne, precyzyjnie podane frazy
  - gęste
    - ✓ większość elementów jest różna od 0
    - ✓ są dość krótkie (długość to kilkadziesiąt, kilkaset)
    - ✓ lepsze w uchwyceniu synonimów, polisemii, kontekstu, wieloznaczności znaczenia itp.
    - ✓ używane w zagadnieniach word embeddings

Modele wektorowe tekstów, word embeddings

- Reprezentacja wektorowa słów word2vec posiada bardzo ciekawą cechę, mianowicie pozwala ona przeprowadzać na słowach operacje arytmetyczne. Przykładowo:
  - brat - mężczyzna + kobieta = siostra
    - ✓ w powyższym wyrażeniu w słowie brat odjęty zostanie aspekt męski a dodany kobiecy co powoduje że w wyniku tej operacji otrzymujemy słowo siostra ☺
  - king - man + woman = queen
- Dla języka polskiego nie istnieje jeszcze zbyt wiele gotowych reprezentacji wektorowych word2vec
  - głównym problemem jest niedostatek korpusów językowych. Innym zagadnieniem jest obfitość wyrazów w naszym języku. Ze względu na bogatą fleksję ilość słów jest znacznie większa niż w językach posiadających ubogą odmianę, co ciągnie za sobą wymóg zbierania znacznie większych korpusów aby każde słowo miało odpowiednio liczną reprezentację

Źródło: http://www.deepdata.pl/uncategorized/przygotowanie-polskiego-modelu-word2vec-z-wykorzystaniem-korpusu-opensubtitles/

Modele wektorowe tekstów, word embeddings

- Nieco nowsza technika (2014) to GloVe
  - projekt stworzony na uniw. Stanforda
  - wykorzystuje globalne statystyki współwystąpień wyrazów w korpusie,
- fastText, stworzone przez Facebook
  - udostępnia wytrenowane modele dla 294 języków!
  - też używa sieci neuronowych do zadania pozyskania word embedding
- „**Świerzynka**” to algorytm BERT
  - od grudnia 2019 BERT oficjalnie stosowany jest w systemie wyszukiwania Google
  - ma nam pomóc zrozumieć, czego szukają ludzie
  - w odróżnieniu od starych, nieuwzględniających kontekstu rozwiązań takich jak word2vec lub GloVe, BERT uwzględnia kontekst słowa, by umiejscowić wprowadzane przez użytkownika dane w konkretnej sytuacji, ponieważ jedno słowo może mieć wiele konotacji i znaczeń w różnych kontekstach

Modele wektorowe tekstów, word embeddings

- Krytyka
  - wyniki bywają stronnicze
  - wyniki często odzwierciedlają kulturowe stereotypy (ang. *human-like biases*)

Modele wektorowe tekstów, word embeddings

- Przykłady dla word2vec w R

```
# Model będziemy trenowali na BARDZO KRÓTKICH tekstach (opisy fabuły 4. części
# filmu Shrek, pobrane w Wikipedii). Tekst poddano typowemu preprocessingowi
#(zamiana lister na małe, usunięcie słów ze stoplisty, usunięcie wszystkich znaków
# interpunkcyjnych i białych znaków). Plik Shrek-1-4-prep.txt zawiera więc już
# same słowa, które posłużą do wygenerowania modelu.
# UWAGA: w praktyce wykorzystywanie tak krótkich tekstów do techniki "word embeddings"
# jest całkowicie pozbawione sensu. Aby metoda ta dawała sensowne wyniki modele muszą
# być uczone na OGROMNYCH korpusach (zbiorach rzeczywistych tekstów,
# setki MB a wręcz GB).
```

```
shrek shrek antisocial highlyterritorial green ogre loves solitude swamp life interrupted verticallycha
farquaad exiles countless number fairytale creatures shrek swamp angered intrusion decides pay farquaad
elsewhere reluctantly allows talkative donkey exiled well tag along guide duloc meanwhile farquaad tort
location fairytale creatures still hiding guards interrupt present snow white magic mirror upon asking
farquaad told even king must marry princess become one decides pursue princess fiona locked castle towe
unwilling perform task organizes tournament winner receives privilege rescuing fiona shrek donkey arriv
defeat swarm farquaad knights farquaad proclaims champions threat death demands rescue fiona shrek nego
creatures relocated succeeds gladly accepts shrek donkey travel castle attacked dragon corners donkey d
beast revealed female dragon falls love carries donkey chambers meanwhile shrek locates fiona appalled
surprised slain dragon flee castle rescuing donkey thrilled fiona quickly becomes disappointed shrek re
insisting farquaad arrive person shrek forcibly carries fiona ventures back duloc donkey night setting
donkey frustration society judging unfairly based looks fiona overhears decides kind shrek next day enc
robin hood band merry men fiona dispatches easily martial arts attack shrek impressed fiona begin fall
fiona takes shelter windmill evening donkey hears strange noises within investigates discovering fiona
explains cursed since childhood forced transform every night sunset changing back sunrise tells donkey
spell change love true form meanwhile shrek confess feelings fiona overhears conversation calling ugly
fiona talking angrily leaves returns next morning lord farquaad confused hurt shrek abrupt hostility to
farquaad marriage proposal requests married nightfall shrek abandons donkey returns nowvacated swamp an
swamp confronts still upset shrek quarrel donkey explains ugly beast fiona referring someone else two r
shrek express feelings fiona marries two quickly travel duloc riding dragon donkey now relationship shr
```

Modele wektorowe tekstów, word embeddings

```
# Do porównania obie metod: CBOW oraz skip-gram.
# W dokumentacji zobacz, jakie są dostępne parametry.

model.cbow <- word2vec(x = "Shrek-1-4-prep.txt", type = "cbow", dim = 15)
model.skip.gram <- word2vec(x = "Shrek-1-4-prep.txt", type = "skip-gram", dim = 15)

# Trenowanie wielkich zbiorow plików trwa często bardzo długo. Wyniki warto
# więc zapisać do pliku. Plik binarny i tekstowy są merytorycznie identyczne, ale
binarny jest kilka razy mniejszy.
write.word2vec(model.cbow, "model.cbow.bin")
write.word2vec(model.cbow, "model.cbow.txt", type = "txt")
write.word2vec(model.skip.gram, "model.skip.gram.bin")
write.word2vec(model.skip.gram, "model.skip.gram.txt", type = "txt")

# Zapisanie  modelu jako zwykłej macierzy
model <- model.cbow
model <- model.skip.gram
embedding <- as.matrix(model)
```

Modele wektorowe tekstów, word embeddings

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]	[,10]	[,11]	[,12]	[,13]	[,14]	[,15]
duloc	-1.1147	0.13838	0.17087	-1.14526	-0.76818	1.2224	1.7932	-1.9731	-0.80666	-0.05467	0.23819	0.74721	1.12976	-0.62030	0.523946
now	-1.5356	0.61987	0.17016	-1.59379	0.77299	0.5470	1.2693	-1.9011	-0.02046	-0.49585	1.23742	-0.88835	0.30290	0.94854	0.090446
ogres	-1.1293	0.21789	-0.34948	-1.26392	-0.46952	1.4555	1.5803	-2.3742	-0.47028	-0.75768	0.36227	0.29867	0.24108	0.58589	0.259668
meanwhile	-0.6300	-0.27081	0.60654	-0.47554	1.18060	1.2953	0.4406	-2.7009	-0.85509	-0.63904	0.91185	-0.15138	1.11023	-0.16047	0.350434
man	-1.8852	1.02734	-0.26525	-0.42092	0.71114	1.2199	1.9278	-1.4655	-0.53087	0.08579	0.45330	-0.21477	0.87457	-0.98484	0.096985
become	-2.3226	1.13415	-0.33753	-0.89644	0.38949	0.6152	1.6077	-1.8876	-0.48643	-0.37208	-0.17232	-0.09427	0.25717	-0.20084	-0.451240
queen	-2.1463	0.20656	1.06726	-0.40920	0.27614	0.9361	1.4470	-1.9961	-0.18162	0.06473	0.97265	-0.13311	0.33935	-0.92270	-0.214499
kingdom	-1.5511	1.03698	0.94452	-0.58758	-0.22664	1.0889	1.1241	-2.5664	-0.79279	-0.54399	-0.22871	0.38859	-0.22286	-0.06619	-0.110813
day	-0.9359	0.70086	-0.48811	-0.79651	-0.07513	1.3959	1.8007	-2.1104	0.02251	-0.65321	0.68743	-0.78149	0.86992	-0.67534	0.621755
villains	-1.8018	0.41429	-0.05168	-1.49373	0.60955	1.1568	1.3452	-2.2692	-0.47901	-0.01671	-0.15038	0.07969	0.60742	0.06503	0.218164
form	-2.1675	1.05086	0.93047	-0.58074	0.53271	0.6864	0.4551	-2.1586	0.24056	-0.48391	1.14617	-0.05533	0.78923	0.24221	0.287537
tells	-1.9838	0.22484	-0.34496	-0.35003	-0.85252	0.9280	0.4497	-1.4307	0.60880	0.08139	1.47339	1.11772	-0.50498	-1.46034	-0.866181
swamp	-2.7157	-0.66626	1.35528	0.05440	0.91301	-0.2369	1.0511	-0.3277	0.97991	0.04313	-0.27125	-0.18539	1.07900	-0.74762	0.667994
back	-1.0243	-0.18039	1.26247	-2.00474	-0.66253	-0.5496	0.6570	-1.9554	0.58916	0.69520	-0.14591	-0.17863	-0.68721	1.37484	-0.251583
prince	-2.1407	1.17676	-0.53100	-1.07112	-0.34307	0.4814	0.5470	-2.3939	-0.44704	-0.14061	-0.25304	-0.36782	0.78166	-0.18711	0.398777
true	-2.2555	0.60263	-0.54628	-1.54629	0.85144	0.1721	1.0838	-1.7646	-0.57819	0.29254	0.38595	-0.48417	0.82824	-0.35609	-0.448794
puss	-2.3582	0.16131	0.73705	-0.50004	0.24418	0.6480	1.7459	-1.7640	-0.46717	-0.67968	0.64908	-0.23448	0.82754	0.34693	-0.136762
harold	-1.9930	0.49878	0.68059	-0.66314	0.44575	0.5039	1.5910	-2.1712	0.28076	-0.17372	0.93064	-0.49066	0.85597	-0.35614	-0.320226
rumpel	-1.7010	0.85544	0.16316	-1.35846	0.44399	0.9328	1.4614	-2.3583	-0.17686	-0.64677	0.33516	0.01075	0.17868	-0.37257	-0.072303
artie	-1.9394	0.33679	0.52638	-1.11064	0.47790	1.4249	1.6682	-2.0432	-0.22304	-0.03964	0.27474	0.12667	0.30894	-0.39550	0.062111
far	-1.4061	0.90492	-0.28874	-0.66654	-0.15191	1.6766	1.8324	-1.9912	-0.02364	-0.31567	-0.26197	-0.50578	0.81567	-0.55153	-0.354409
ogre	-1.9992	0.76386	0.81800	-1.19752	-0.38564	0.8804	1.5664	-2.1345	0.05746	-0.06247	0.24483	0.43665	-0.31912	0.06073	-0.140849
kiss	-2.3735	1.13610	-0.07778	-0.93592	0.16996	1.6588	0.6589	-1.7044	-0.12211	-0.45886	0.09668	-0.24835	0.34235	0.76920	0.262366
love	-1.4300	0.93543	0.25193	-0.93884	0.19182	1.3582	1.3111	-2.4692	-0.61762	-0.28351	0.24219	-0.26032	0.59941	-0.63445	-0.296209
fairy	-1.2520	1.21224	-0.83507	-0.01309	0.28036	0.3778	1.5082	-2.3702	0.27442	0.63965	-0.71234	0.77076	0.89255	-0.78942	0.381706
dragon	-1.3621	0.12796	0.38734	-1.32615	0.75649	1.1030	1.4235	-2.5079	-0.86486	-0.27171	0.17114	0.03490	0.10631	0.50102	-0.004117
castle	-1.8262	1.27826	0.16365	-0.24775	0.40224	0.8278	1.5044	-1.6103	0.55943	-1.18431	0.46092	-0.85378	0.98700	0.61692	-0.477115
godmother	-1.8302	0.89662	0.06466	-0.76633	0.47610	1.2707	0.7733	-2.3405	-0.85311	-0.14669	0.74793	-0.69434	0.44059	0.36231	0.470369
fiona	-1.9748	0.24385	-0.08801	-1.25107	-0.02946	1.3452	1.7625	-1.8218	-0.45437	-0.22671	0.67825	-0.33758	0.45267	-0.38433	-0.217735
lillian	-1.0352	0.63007	-0.40204	-1.29921	-0.26946	0.6179	1.4159	-2.5470	-0.85644	-0.80141	-0.34473	-0.96375	-0.02938	0.18445	-0.526711
king	-1.3819	-0.04079	0.63521	-1.39808	0.59482	1.6809	1.6333	-1.7643	-0.57910	-0.60436	0.33023	-0.60726	0.48249	-0.20845	0.562034
shrek	-1.7995	0.48746	0.34753	-1.31476	0.24755	1.0189	1.5185	-2.3684	-0.17809	-0.40892	0.24557	-0.06076	0.61124	-0.10742	0.110597
</s>	1.4757	1.19775	0.19239	-1.22297	-1.41889	0.9629	-0.4186	-0.4248	-0.50282	1.01660	0.95989	1.38770	-0.75605	-0.77341	-1.120663
charming	-1.9906	0.17954	0.19570	-1.42842	0.56848	1.5673	1.5203	-1.7062	-0.04727	-0.85802	0.07362	0.26701	0.28709	-0.14515	-0.077065
farquaad	-2.0938	-0.24026	0.52579	-0.70388	0.76427	1.1084	1.5394	-1.8499	0.10026	-0.91523	0.28254	0.65024	0.34879	-0.70628	-0.459528
potion	-1.4488	0.95617	-0.04733	-0.05045	0.99845	1.5229	1.5109	-1.8137	-0.49207	-0.49403	0.76852	-0.87542	-0.03610	-0.76386	-0.816030
away	-1.5894	0.82129	-0.07588	-1.65929	-0.06672	0.8738	1.2339	-2.4010	-0.79640	0.12676	0.34554	-0.24526	-0.02976	0.38231	0.088286
will	-0.5592	0.70979	-0.61745	-0.61168	-0.63426	0.4920	0.8416	-1.6924	1.31576	-1.37910	-1.35977	-1.26474	-0.48028	1.02186	-0.924372
donkey	-1.9249	-0.05461	0.73651	-0.90241	0.14334	1.2935	1.1530	-2.2395	-0.05639	-0.62116	0.70119	-0.33518	-0.10801	0.70124	-0.632303
reveals	-2.0325	0.38260	0.01433	-0.64953	-0.29018	0.9143	1.3874	-2.2581	-0.55496	-0.77850	0.29046	0.25241	1.10314	-0.21616	0.175378

Modele wektorowe tekstów, word embeddings

```
# Typ 'embedding'. Wyświetlenie wiersza z macierzy dla podanego słowa.
# Gdy słowa nie ma w modelu, wyświetlają się NA.
(embedding <- predict(model, c("shrek"), type = "embedding"))
      [,1]   [,2]   [,3]   [,4]   [,5]  [,6]  [,7]   [,8]    [,9]   [,10]  [,11]    [,12]  [,13]   [,14]  [,15]
shrek -1.8 0.4875 0.3475 -1.315 0.2476 1.019 1.518 -2.368 -0.1781 -0.4089 0.2456 -0.06076 0.6112 -0.1074 0.1106

# Typ 'nearest'. Pokazuje słowa najbardziej podobne do podanego wraz
# ze współczynnikiem podobieństwa
#(1 - najbardziej podobne, 0 - niepodobne, -1 - podobne "odwrotnie" )
(lookslike <- predict(model, c("fiona"), type = "nearest", top_n = 6))
$fiona
  term1    term2 similarity rank
1 fiona    artie     0.9782    1
2 fiona    shrek     0.9762    2
3 fiona   rumpel     0.9681    3
4 fiona     love     0.9665    4
5 fiona charming     0.9662    5
6 fiona villains     0.9616    6

(lookslike <- predict(model, c("shrek"), type = "nearest", top_n = 6))
$shrek
  term1    term2 similarity rank
1 shrek   rumpel     0.9901    1
2 shrek    artie     0.9864    2
3 shrek villains     0.9858    3
4 shrek     love     0.9772    4
5 shrek    fiona     0.9762    5
6 shrek charming     0.9746    6
```

Modele wektorowe tekstów, word embeddings

```
# Sensowne analizy wymagają dużych modeli, których trenowanie jest bardzo czasochłonne.
# Na szczęście w sieci można znaleźć sporo "gotowców", głównie dla języka angielskiego.
# Są też jednak modele dla j. polskiego (patrz wykład)
# https://github.com/maxoodf/word2vec#basic-usage
# "You can download one or more models (833MB each) trained on 11.8GB English texts corpus:"
#                                       ^^^^^       ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
model <- read.word2vec(file = "cb_ns_500_10.w2v", normalize = TRUE)

# Możliwe są operacje arytmetyczne.
# Klasyczny przykład: king - man + woman = queen
# https://cran.r-project.org/web/packages/word2vec/readme/README.html
wv <- predict(model, newdata = c("king", "man", "woman"), type = "embedding")
wv <- wv["king", ] - wv["man", ] + wv["woman", ]
predict(model, newdata = wv, type = "nearest", top_n = 3)
```

term similarity rank 1 king 0.9479 1 2 queen 0.7680 2 3 princess 0.7155 3

```
# W modelu (macierzy) są najzwyklejsze liczby, możemy więc różne rzeczy wymyślać
wv <- predict(model, newdata = c("black", "white", "racism", "person"), type = "embedding")
wv <- wv["white", ] - wv["person", ] + wv["racism", ]
predict(model, newdata = wv, type = "nearest", top_n = 10)

wv <- predict(model, newdata = c("black", "white"), type = "embedding")
wv <- wv["black", ] + wv["white", ]
predict(model, newdata = wv, type = "nearest", top_n = 3)
```

Omówienie skryptu TDM_basic.R

- Załadowanie wymaganych bibliotek

```
library(tm)
library(proxy) # funkcja dist
library(dplyr)
library(SnowballC) # żeby działała opcja stemming=TRUE
library(ggplot2)
library(wordcloud) # word clounds

options(digits = 4)
```

- Przykładowe dane
  - zwróćmy uwagę na nadmiarowe spacje i „dziwną” interpunkcję

```
doc = c(
  "The,,, Google        matrix P is a model of the Internet.",
  "P ij is nonzero if       there is a link from Web page j to i.",
  "The Google... matrix rank is used to rank all Web pages!",
  "The ranking;;; is done by          solving a matrix eigenvalue problem!!!",
  "England 123-222-999 dropped out of the               top 10 in the FIFA ranking?;;"
)
```

```
query <- c("ranking web pages")
```

- Tworzymy tzw. korpus i wyświetlamy jego zawartość

```
# Tworzymy TDM.
# Tworzymy korpus i zaczynamy obrabiać jego zawartość.
# Do korpusu wkladamy dokumenty i zapytania (c(doc, query)).
# Robimy tak, aby wektor query był zgodny z TDM (potafisz to wyjaśnić?).
# Ostatnie "lquery" kolumn to wektory query.
Corpus_obj <- Corpus(VectorSource(c(doc, query)))

# Oglądamy nasz korpus
lapply(Corpus_obj[1:length(Corpus_obj)], as.character)
inspect(Corpus_obj)
```

```
> inspect(Corpus_obj)
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 6

[1] The,,, Google        matrix P is a model of the Internet.
[2] P ij is nonzero if       there is a link from Web page j to i.
[3] The Google... matrix rank is used to rank all Web pages!
[4] The ranking;;; is done by          solving a matrix eigenvalue problem!!!
[5] England 123-222-999 dropped out of the                top 10 in the FIFA ranking?;;
[6] ranking web pages
>
```

- Zaczynamy tworzyć macierz TDM. Musimy najpierw wskazać systemowi, jak ma to zrobić

```
# Plik kontrolny dla funkcji TermDocumentMatrix
control_list <- list(
  # Usuwamy słowa krótsze niz 3 znaki.
  wordLengths = c(3, Inf),
  tolower = TRUE,
  removeNumbers = TRUE,
  # wykonanie niezależnie polecenia stopwords("SMART") pokaże listę słów
  stopwords = stopwords("SMART"),
  # Tutaj jest krótsza lista
  # stopwords = stopwords("en"),
  # Te znaki będę usunięte: ! " # $ % & ' ( ) * + , - . / : ; < = > ? @ [
  removePunctuation = TRUE,
  # Stemmer Portera
  stemming = TRUE,
  #stripWhitespace = TRUE,
  # Rożne schematy tworzenia TDM-a.
  # Doczytaj dokładnie w dokumentacji, w jaki sposób można podawać różne s
  # weighting = weightBin
  # weighting = function(x) weightTfIdf(x, normalize=TRUE)
  # weighting = function(x) weightSMART(x, spec = "bnc")
  weighting = function(x)  weightTfIdf(x, normalize = TRUE),
  #weighting = weightBin,
  language = 'en'
)
```

- Powstaje obiekt klasy TermDocumentMatrix

```
> tdm <- TermDocumentMatrix(Corpus_obj, control = control_list)
> # Uwaga, to polecenie pokazuje tylko kilka pierwszych termów
> inspect(tdm)
<<TermDocumentMatrix (terms: 16, documents: 6)>>
Non-/sparse entries: 26/70
Sparsity           : 73%
Maximal term length: 9
Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
Sample             :
          Docs
Terms            1      2      3     4     5      6
  drop      0.0000 0.0000 0.0000 0.000 0.517 0.0000
  googl     0.3962 0.0000 0.2642 0.000 0.000 0.0000
  internet  0.6462 0.0000 0.0000 0.000 0.000 0.0000
  link      0.0000 0.6462 0.0000 0.000 0.000 0.0000
  matrix    0.2500 0.0000 0.1667 0.200 0.000 0.0000
  model     0.6462 0.0000 0.0000 0.000 0.000 0.0000
  nonzero   0.0000 0.6462 0.0000 0.000 0.000 0.0000
  page      0.0000 0.2500 0.1667 0.000 0.000 0.3333
  rank      0.0000 0.0000 0.1950 0.117 0.117 0.1950
  web       0.0000 0.2500 0.1667 0.000 0.000 0.3333
> |
```

- Oglądamy wynikową listę termów

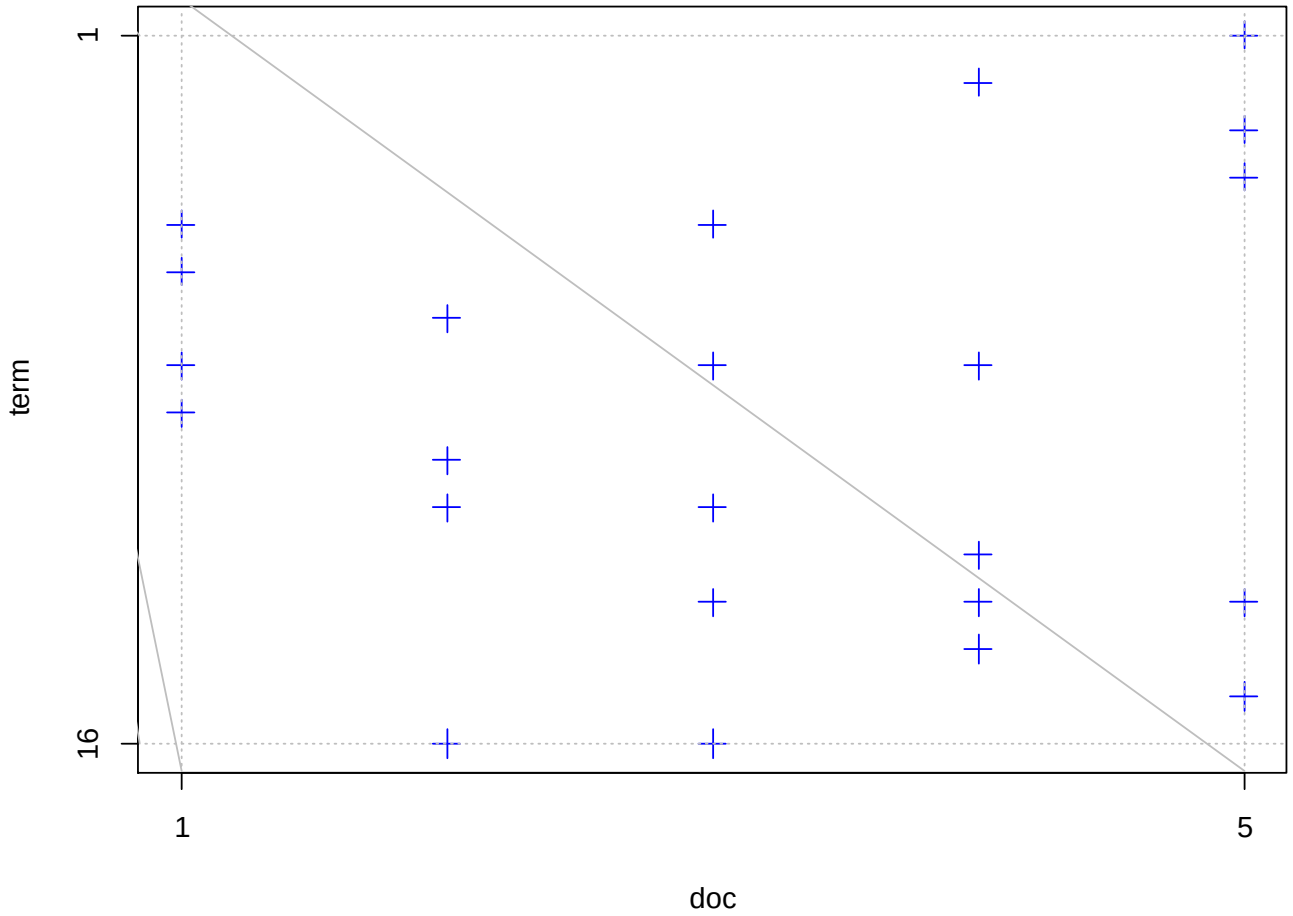
```
> # Tu wyświetlą się wszystkie termy.
> # Zwróćmy uwagę, jak stemmer pozmieniał słowa.
> tdm$dimnames$Terms
 [1] "drop"      "eigenvalu" "england"   "fifa"      "googl"     "internet"
 [7] "link"      "matrix"    "model"     "nonzero"   "page"      "problem"
[13] "rank"      "solv"      "top"       "web"
>
```

- Na potrzeby różnych demonstracji z obiektu tdm robimy zwykłe macierze

```
> tdm.mtx.all
           Docs
Terms           1      2      3     4     5      6
  drop      0.0000 0.0000 0.0000 0.000 0.517 0.0000
  eigenvalu 0.0000 0.0000 0.0000 0.517 0.000 0.0000
  england   0.0000 0.0000 0.0000 0.000 0.517 0.0000
  fifa      0.0000 0.0000 0.0000 0.000 0.517 0.0000
  googl     0.3962 0.0000 0.2642 0.000 0.000 0.0000
  internet  0.6462 0.0000 0.0000 0.000 0.000 0.0000
  link      0.0000 0.6462 0.0000 0.000 0.000 0.0000
  matrix    0.2500 0.0000 0.1667 0.200 0.000 0.0000
  model     0.6462 0.0000 0.0000 0.000 0.000 0.0000
  nonzero   0.0000 0.6462 0.0000 0.000 0.000 0.0000
  page      0.0000 0.2500 0.1667 0.000 0.000 0.3333
  problem   0.0000 0.0000 0.0000 0.517 0.000 0.0000
  rank      0.0000 0.0000 0.1950 0.117 0.117 0.1950
  solv      0.0000 0.0000 0.0000 0.517 0.000 0.0000
  top       0.0000 0.0000 0.0000 0.000 0.517 0.0000
  web       0.0000 0.2500 0.1667 0.000 0.000 0.3333
>
```

```
query <- c("ranking web pages")
```

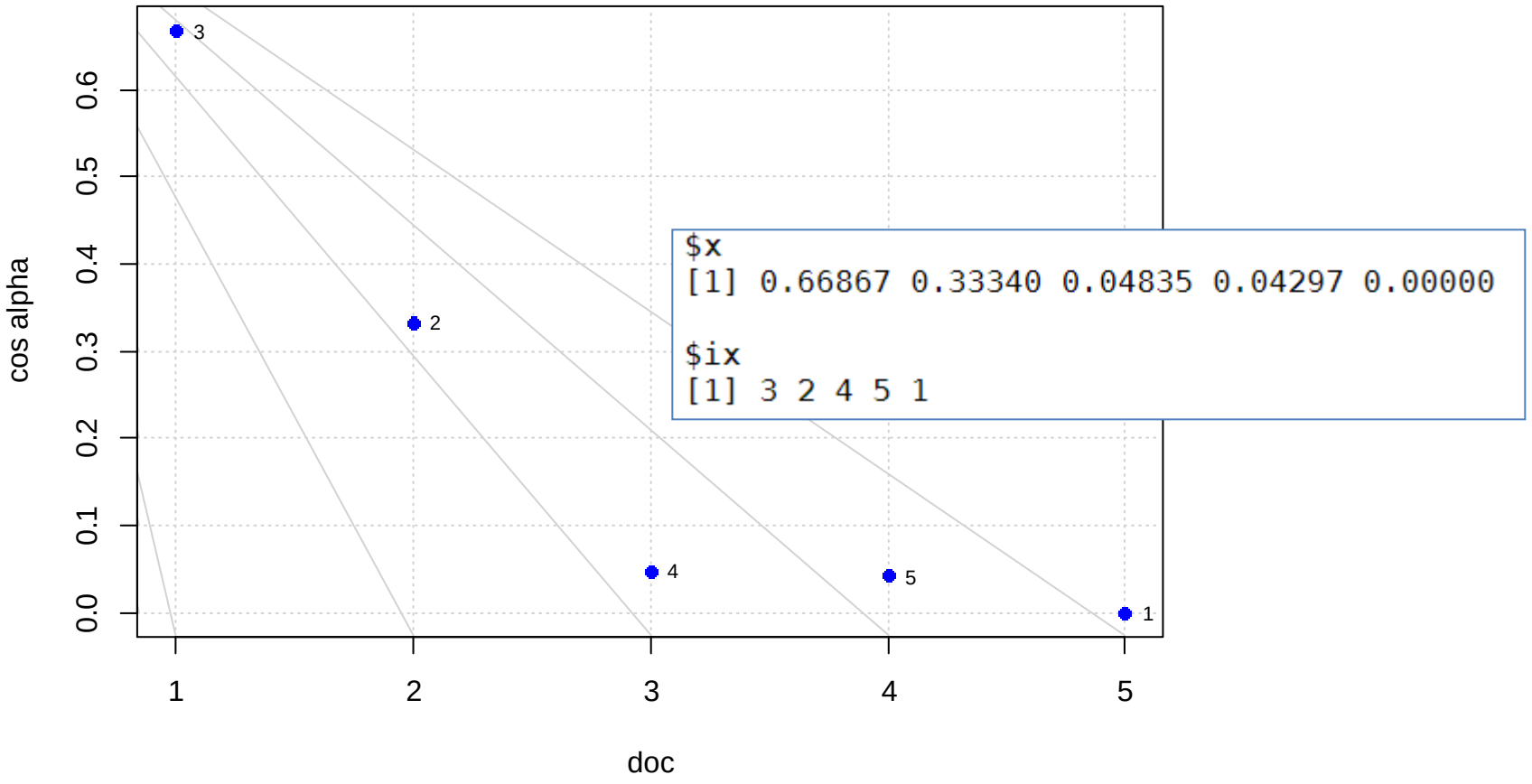
- Wizualizacja macierzy tdm
  - przeanalizuj dokładnie kod kreślący ten obrazek!



- Wyznaczamy cos α pomiędzy tdm.mtx a query i ustalamy ostateczny ranking dokumentów

```
> # ustawiamy ranking dokumentów według ważności
> (sorted <- sort.int(cos.alpha, decreasing = TRUE, index.return = TRUE))
$x
[1] 0.66867 0.33340 0.04835 0.04297 0.00000

$ix
[1] 3 2 4 5 1
```

- Wizualizujemy wynik jak wyżej
  - przeanalizuj dokładnie kod kreślący ten obrazek!



- Przykładowe proste analizy
  - częstość występowania termów

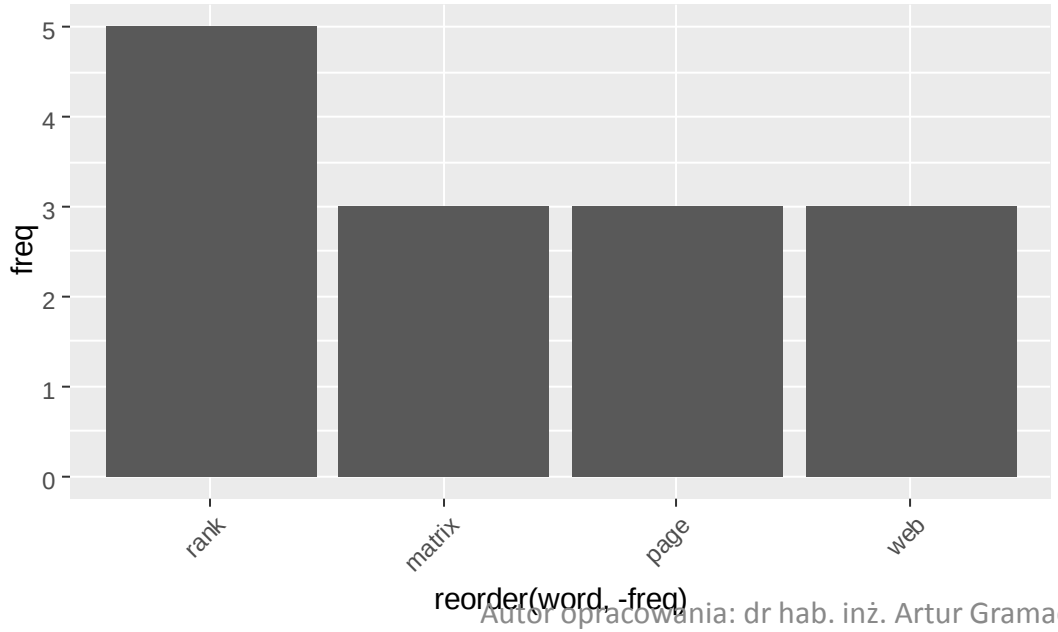
```
# Analiza częstości występowania słów.
# Najwygodniej robić to na bazie binarnej macierzy TDM (dlaczego?)
# Plik kontrolny dla funkcji TermDocumentMatrix
control_list <- list(
  wordLengths = c(3, Inf),
  tolower = TRUE,
  removeNumbers = TRUE,
  stopwords = stopwords("SMART"),
  removePunctuation = TRUE,
  stemming = TRUE,
  weighting = weightBin,
  language = 'en'
)
tdm.2 <- TermDocumentMatrix(Corpus_obj, control = control_list)
```

- Przykładowe proste analizy
  - częstość występowania termów

```
> inspect(tdm.2)
<<TermDocumentMatrix (terms: 16, documents: 6)>>
Non-/sparse entries: 26/70
Sparsity           : 73%
Maximal term length: 9
Weighting          : binary (bin)
Sample             :
           Docs
Terms       1 2 3 4 5 6
  drop      0 0 0 0 1 0
  eigenvalu 0 0 0 1 0 0
  england   0 0 0 0 1 0
  fifa      0 0 0 0 1 0
  googl     1 0 1 0 0 0
  internet  1 0 0 0 0 0
  matrix    1 0 1 1 0 0
  page      0 1 1 0 0 1
  rank      0 0 1 1 1 1
  web       0 1 1 0 0 1
>
```

- Przykładowe proste analizy
  - częstość występowania termów

```
> # Najczęściej pojawiające sie frazy
> (freq <- sort(rowSums(as.matrix(tdm.2)), decreasing = TRUE))
     rank    matrix      page       web     googl      drop eigenvalu   england
        4         3         3         3         2         1         1         1
     fifa  internet      link     model   nonzero   problem      solv       top
        1         1         1         1         1         1         1         1
```



- Analogiczne analizy ale dla nieco większego zbiory danych

```
> source("SIAM_S_C_M.R")
> doc
  [1] "1   Anthology of Statistics in Sports"
  [2] "2   A First Course in Order Statistics"
  [3] "3   Mathematica Laboratories for Mathematical Statistics: Emphasiz
  [4] "4   Some Limit Theorems in Statistics"
  [5] "5   Engineering Reliability"
```

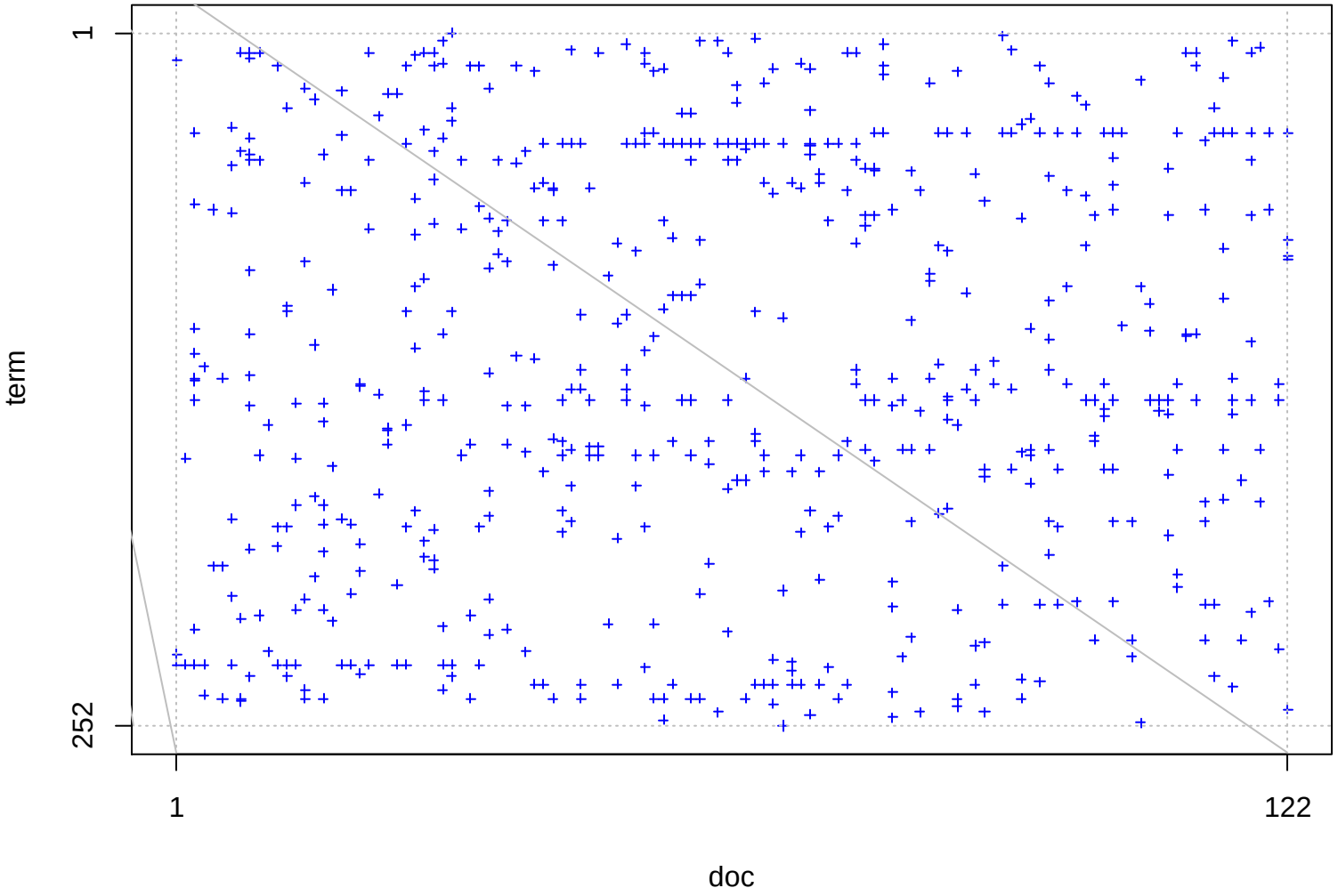
```
[117] "117  Solving PDEs in C++"
[118] "118  Design Sensitivity Analysis: Computational Issues of Sensitivity Eq
[119] "119  Numerical Polynomial Algebra"
[120] "120  Computational Science and Engineering"
[121] "121  Spectral Methods in MATLAB"
[122] "122  Computational Frameworks for the Fast Fourier Transform"
```

```
> query <- c("Gramacki matlab")
> (ldoc <- length(doc))
[1] 122
> (lquery <- length(query))
[1] 1
>
```

- Zwróćmy uwagę na dużo większy współczynnik rzadkości

```
> inspect(tdm)
<<TermDocumentMatrix (terms: 252, documents: 123)>>
Non-/sparse entries: 578/30418
Sparsity           : 98%
Maximal term length: 19
Weighting          : binary (bin)
Sample             :
          Docs
Terms      103 17 28 29 3 30 35 52 9 96
  analysi    0  0  1  1 0  0  0  1 1  0
  comput     1  0  0  0 1  0  0  1 0  0
  control    0  0  0  0 0  0  0  1 0  0
  design     0  0  0  0 0  0  0  0 1  0
  method     1  0  1  0 1  1  0  0 0  0
  numer      0  0  0  0 0  0  0  0 0  1
  optim      0  0  0  0 0  0  0  0 0  0
  statist    0  0  0  0 1  1  0  0 0  0
  system     0  0  0  0 0  0  0  0 0  0
  theori     0  1  0  0 0  0  0  0 0  0
```

- Wizualizacja macierzy tdm



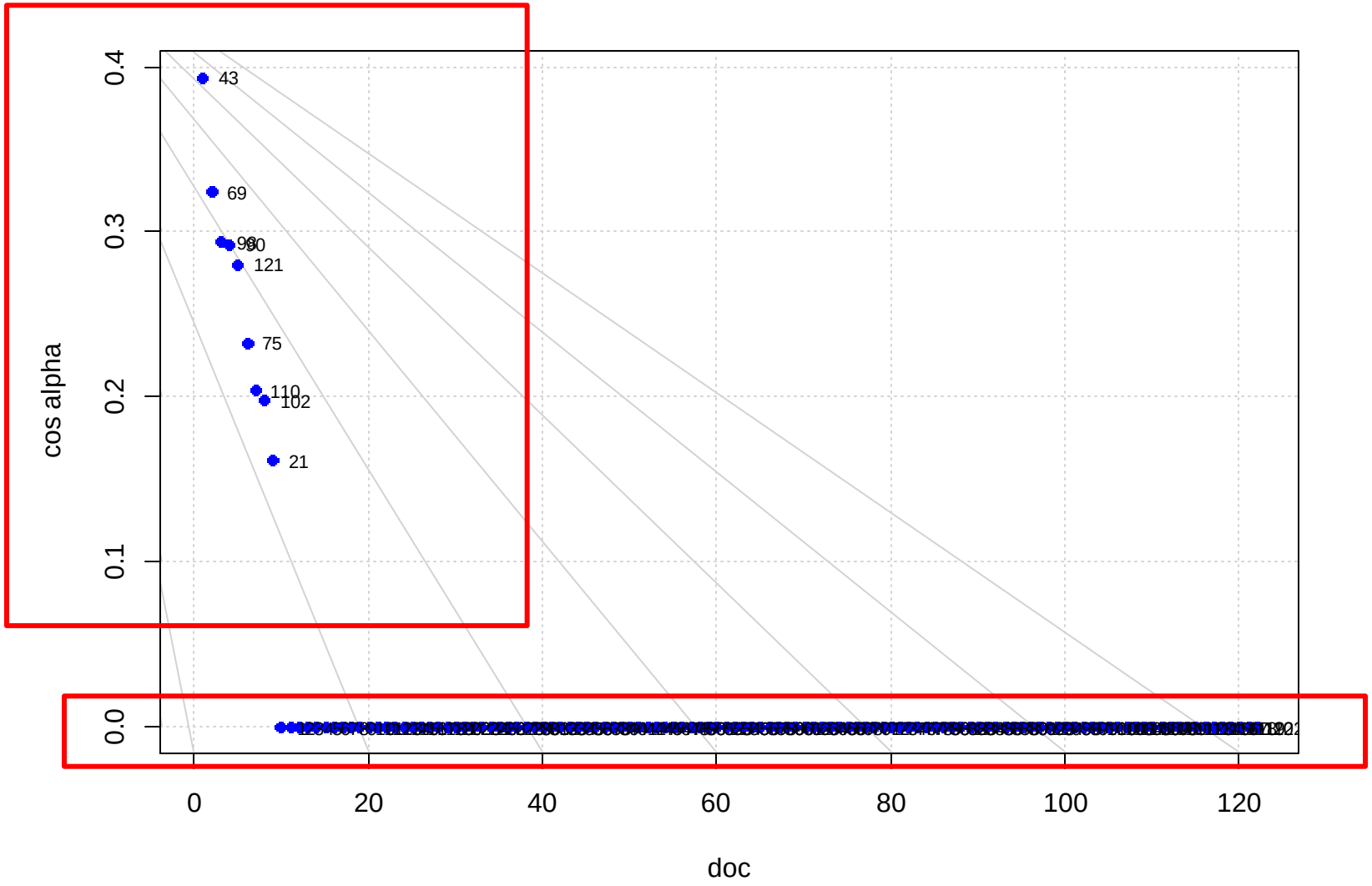
- Wyznaczamy cos α pomiędzy tdm.mtx a query i ustalamy ostateczny ranking dokumentów

```
> (cos.alpha <-
+     as.vector(1 - proxy::dist(t(tdm.mtx.query), t(tdm.mtx), method = 'cosine')))
  [1] 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
 [12] 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.1619 0.0000
 [23] 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
 [34] 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.3937 0.0000
 [45] 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
 [56] 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
 [67] 0.0000 0.0000 0.3244 0.0000 0.0000 0.0000 0.0000 0.0000 0.2329 0.0000 0.0000
 [78] 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000
 [89] 0.0000 0.2927 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.2939 0.0000
[100] 0.0000 0.0000 0.1981 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.2042
[111] 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.2807
[122] 0.0000
>
```

- Ranking dokumentów

```
> # Pokazujemy numery dokumentów, w kolejności od  najbardziej zgodnego z query.
> sorted$ix
  [1]  43  69  98  90 121  75 110 102  21   1   2   3   4   5   6   7   8   9  10
 [20]  11  12  13  14  15  16  17  18  19  20  22  23  24  25  26  27  28  29  30
 [39]  31  32  33  34  35  36  37  38  39  40  41  42  44  45  46  47  48  49  50
 [58]  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  70
 [77]  71  72  73  74  76  77  78  79  80  81  82  83  84  85  86  87  88  89  91
 [96]  92  93  94  95  96  97  99 100 101 103 104 105 106 107 108 109 111 112 113
[115] 114 115 116 117 118 119 120 122
>
```

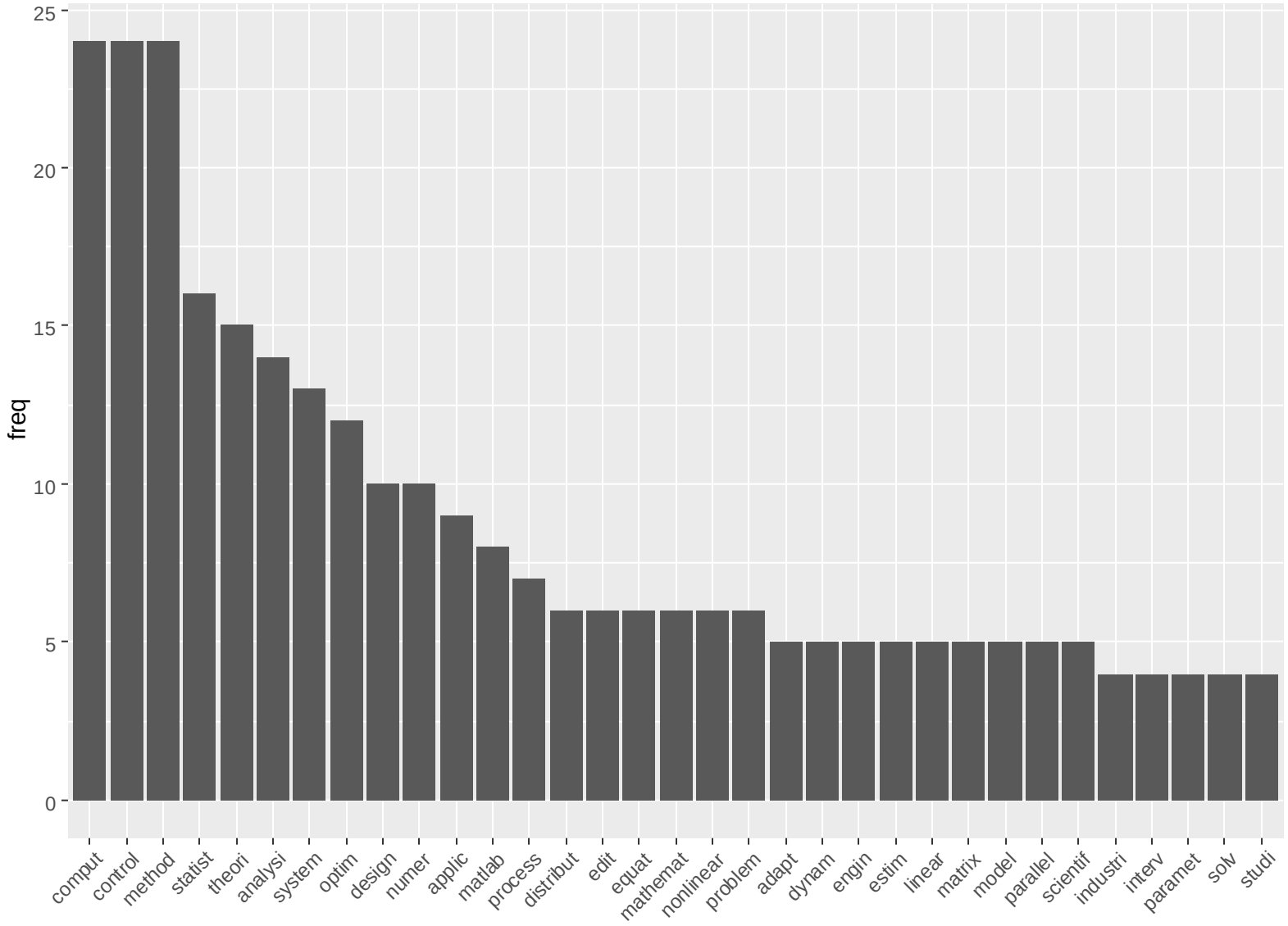
- Wynik końcowy



- Query i 5 dokumentów najbardziej zgodnych

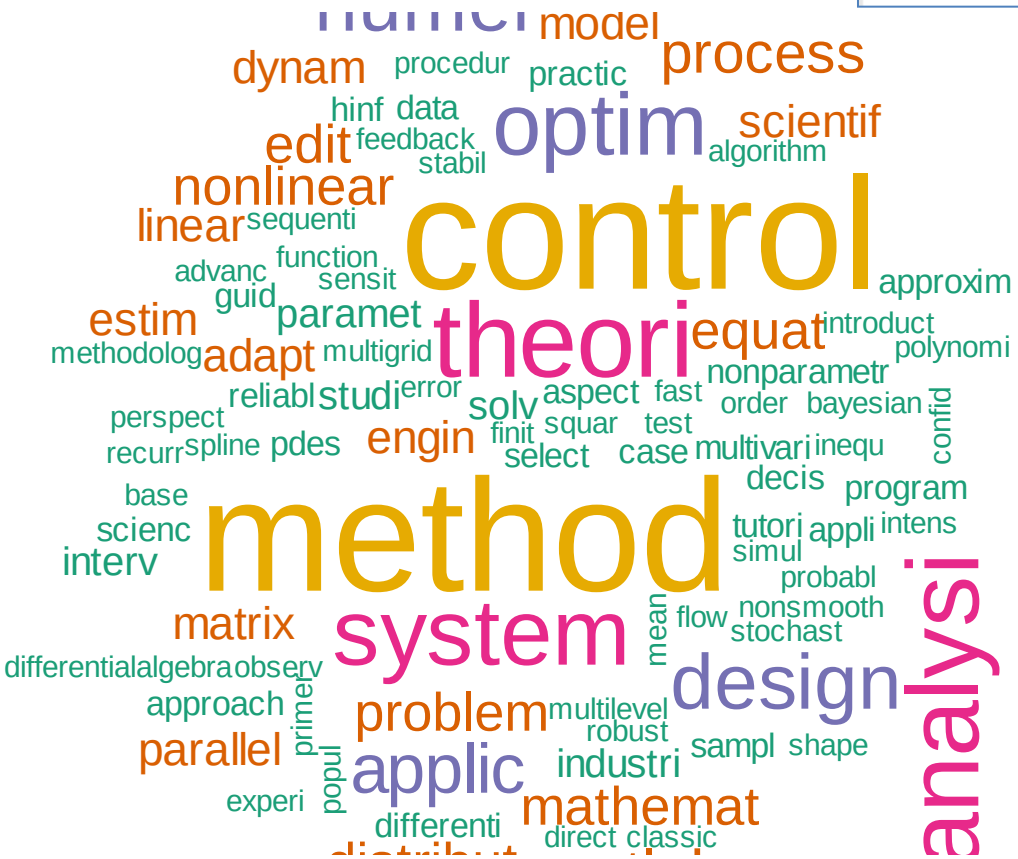
```
> query
[1] "matlab programming"
> doc[c(43, 69, 98, 90, 121)]
[1] "43  Practical Methods for Optimal Control and Estimation Using Nonlinear Programming"
[2] "69  Applied Dynamic Programming for Optimization of Dynamical Systems"
[3] "98  MATLAB Guide, Second Edition"
[4] "90  Learning MATLAB"
[5] "121  Spectral Methods in MATLAB"
```

- Częstość występowania termów



- Chmura termów

```
wordcloud(
  names(freq),
  freq,
  min.freq = 2,
  scale = c(5, .1),
  colors = brewer.pal(6, "Dark2")
)
```



Omówienie skryptu TDM_basic_2.R (przykład *MEDLINE*)

Dokumenty (1-1033)

1. correlation between maternal and fetal plasma levels of glucose and free fatty acids . correlation coefficients have been determined between the levels of glucose and ffa in maternal and fetal plasma collected at delivery . significant correlations were obtained between the maternal and fetal glucose levels and the maternal and fetal ffa levels . from the size of the correlation coefficients and the slopes of regression lines it appears that the fetal plasma glucose level at delivery is very strongly dependent upon the maternal level whereas the fetal ffa level at delivery is only slightly dependent upon the maternal level .

…

1033. hemorrhagic episodes in hemophilia: a 5-year prospective study. medicosocial studies of hemophilia are of particular clinical importance in allowing an assessment of the likely course of the disease at different ages and for differing grades of severity, and in providing knowledge of which complications cause the most disability, loss of education, and earning capacity. they also overcome the distorted clinical impression of the disease which arises from the recurrent admission of the same few severely affected hemophiliacs. owing to the considerable individual variation in the number and severity of complications in different hemophiliacs an accurate individual prognosis can never be given. in general, however, the number of spontaneous episodes per year decreases with age, while the severity of individual episodes tends to increase, at least until the age of 21 yr. there is general agreement that the bulk of hospital hemophilic admissions are due to hemarthroses and that hemophilic arthropathy involves the knee more than it does any other joint. the increased time spent in hospital per episode in later life is in part at least due to the development of relatively unstable weight-bearing joints due to hemophilic arthropathy and associated muscle atrophy. thus the correct management of individual hemarthroses in childhood is of considerable importance, and at the present time too little is known of the best possible treatment for these episodes. little is known of the pathological mechanisms of hemophilic arthropathy and whether it is the presence of blood or its presence under tension which leads to joint destruction. thus opinions differ concerning the routine admission of all hemarthroses to hospital regardless of severity and also about the advisability of joint aspiration in an attempt to avoid the development of destructive arthropathy. because of the individual variation between patients, the changes in the pattern of the disease with age, and the difficulty of obtaining suitable control patients these questions can be answered only by further longterm prospective medicosocial studies.

Zapytania (1-30)

1. the crystalline lens in vertebrates, including humans.

2. the relationship of blood and cerebrospinal fluid oxygen concentrations or partial pressures. a method of interest is polarography.

3. electron microscopy of lung or bronchi.

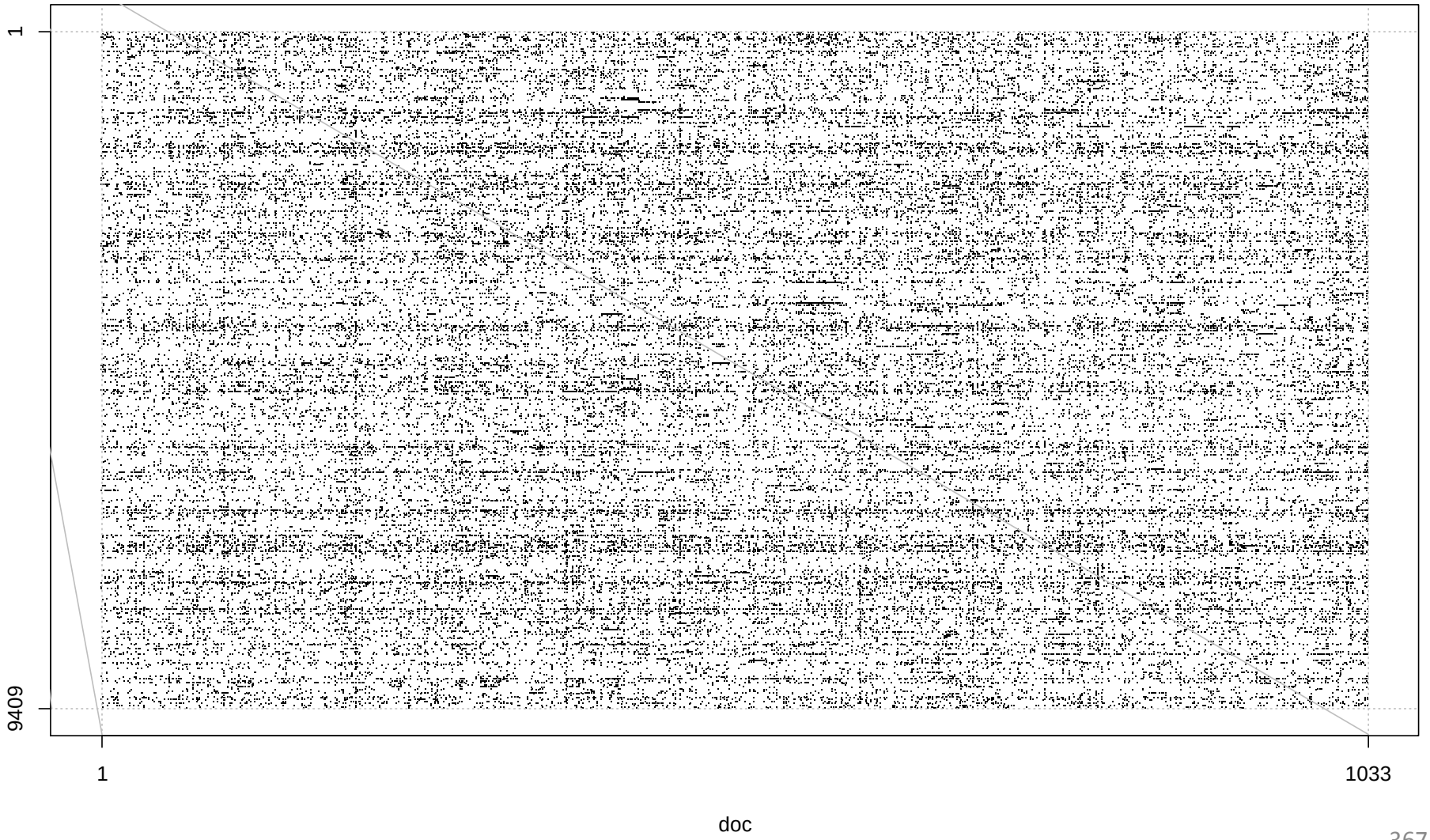
…

29. hereditary implications of prolonged neonatal obstructive jaundice associated with liver pathology. there are but two relatively common well-defined pathologic entities that present as such: 1) bile duct or biliary atresia and 2) giant cell transformation of the liver often referred to as neonatal hepatitis. information regarding liver and bile duct embryogenesis will be an important adjunct to papers on the disease processes.

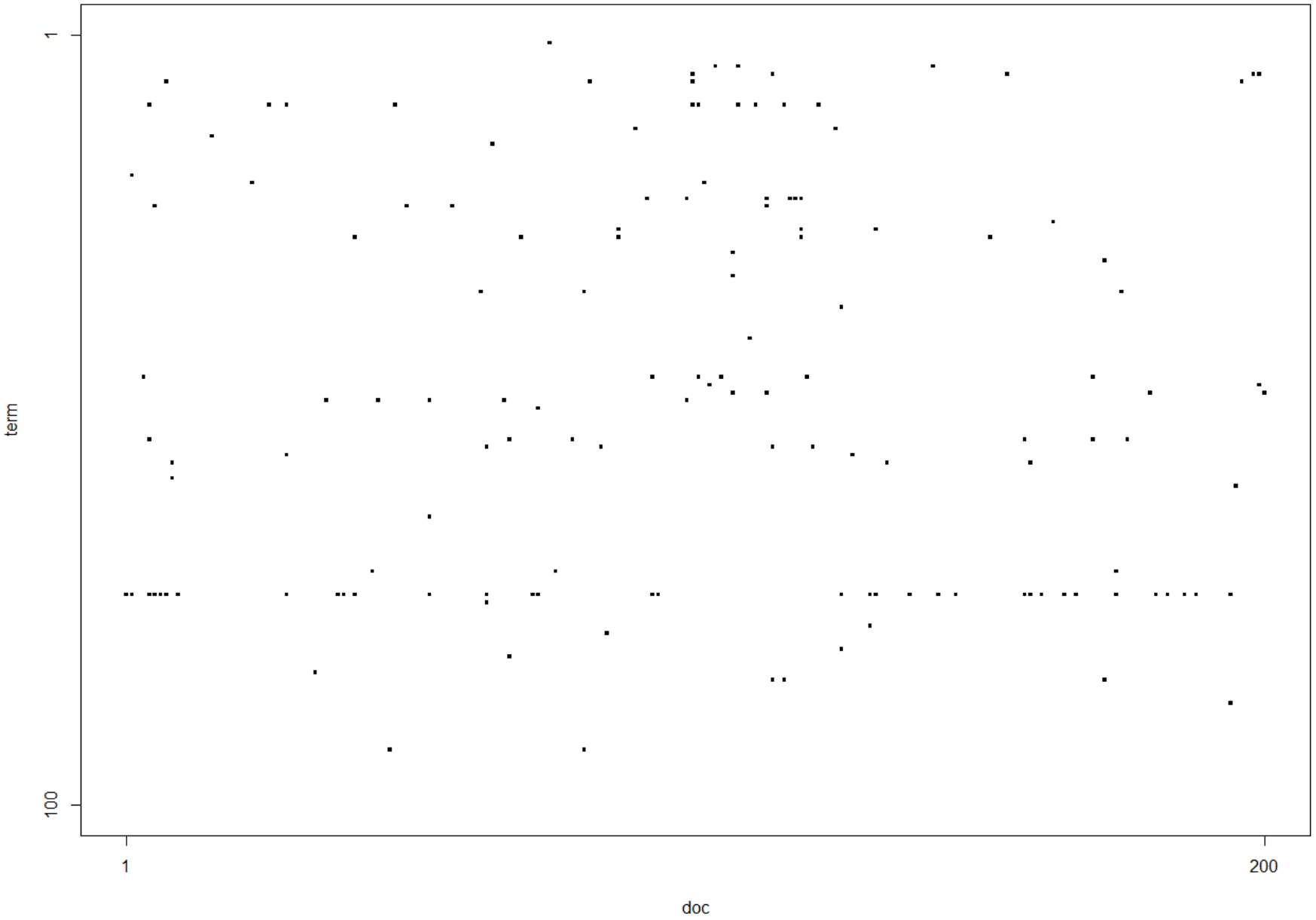
30. hemophilia and christmas disease, especially in regard to the specific complication of pseudotumor formation (occurrence, pathogenesis, treatment, prognosis).

Macierz TDM

```
inspect(tdm)
<<TermDocumentMatrix (terms: 9409, documents: 1063)>>
Non-/sparse entries: 60674/9941093 Sparsity : 99%
```



Macierz TDM - fragment



- Wczytanie dwóch plików: dokumenty i query
  - używamy tutaj dedykowanej funkcji read.docs(), gdzie zakładamy, że wszystkie dokumenty są w jednym pliku i są oddzielone przynajmniej jedną pustą linią

```
> doc <- read.docs(file="MED.A.txt")
> query <- read.docs(file="MED.Q.txt")
> (ldoc <- length(doc))
[1] 1033
> (lquery <- length(query))
[1] 30
```

```
# W bazie MEDLINE każde query (od 1 do 30) ma zdefiniowaną listę
# dokumentów zgodnych (ang. relevant) z zapytaniem. Domyślam sie, że
# to zadanie wykonali eksperci dziedzinowi.
# Nie ma tutaj jakiejś hierarchii, który dokument jest najbardziej zgodny, a który najmniej.
# Jest po prostu lista zgodnych i już (choć numery dokumentów zgodnych są posortowane).
Q1.REL <- c(13,14,15,72,79,138,142,164,165,166,167,168,169,170,171,172,180,181,182,183,184,18
Q2.REL <- c(80,90,162,187,236,237,258,289,290,292,293,294,296,300,301,303)
Q3.REL <- c(59,62,67,69,70,71,73,78,81,160,163,230,231,232,233,234,276,277,279,282,283,287)
Q4.REL <- c(93,94,96,141,173,174,175,176,177,178,207,208,209,210,259,396,397,399,400,404,405,
Q5.REL <- c(1,2,4,5,6,7,8,9,10,11,12,158,159,188,304,305,306,307,325,326,327,329,330,331,332,
```

- W bazie MEDLINE każde query (od 1 do 30) ma zdefiniowaną listę dokumentów zgodnych (ang. relevant) z zapytaniem.

```
> Q1.REL <- c(13,14,15,72,79,138,142,164,165,166,167,168,169,170,171,172,180,
181,182,183,184,185,186,211,212,499,500,501,502,503,504,506,507,508,510,511,5
13)
> Q9.REL <- c(30,31,53,56,57,64,83,84,89,124,125,126,192,252,253,267,268,269,
270,271,272,273,409,412,415,420,421,422)
```

- Dla wygody zrobiono dedykowaną funkcję do łatwego tworzenia macierzy TDM

```
> tdm <- create.TDM(doc = doc, query = query, weighting = "TfIdf.Norm.T", spec = "bnc",
 lang = 'en', min.words.length = 3)
> inspect(tdm)
<<TermDocumentMatrix (terms: 9312, documents: 1063)>>
Non-/sparse entries: 54436/9844220
Sparsity           : 99%
Maximal term length: 31
Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
Sample             :
          Docs
Terms      1060 335    34    520 606 727 729 732 765   823
  acid        0   0 0.000 0.0000   0   0   0   0   0 0.000
  cancer      0   0 0.000 0.0000   0   0   0   0   0 0.000
  case        0   0 0.109 0.0000   0   0   0   0   0 0.129
  cell        0   0 0.000 0.0000   0   0   0   0   0 0.000
  children    0   0 0.000 0.0000   0   0   0   0   0 0.000
  effect      0   0 0.000 0.0566   0   0   0   0   0 0.000
  growth      0   0 0.000 0.0000   0   0   0   0   0 0.000
  hormon      0   0 0.000 0.0000   0   0   0   0   0 0.000
  patient     0   0 0.000 0.0000   0   0   0   0   0 0.000
  rat         0   0 0.000 0.0000   0   0   0   0   0 0.000
```

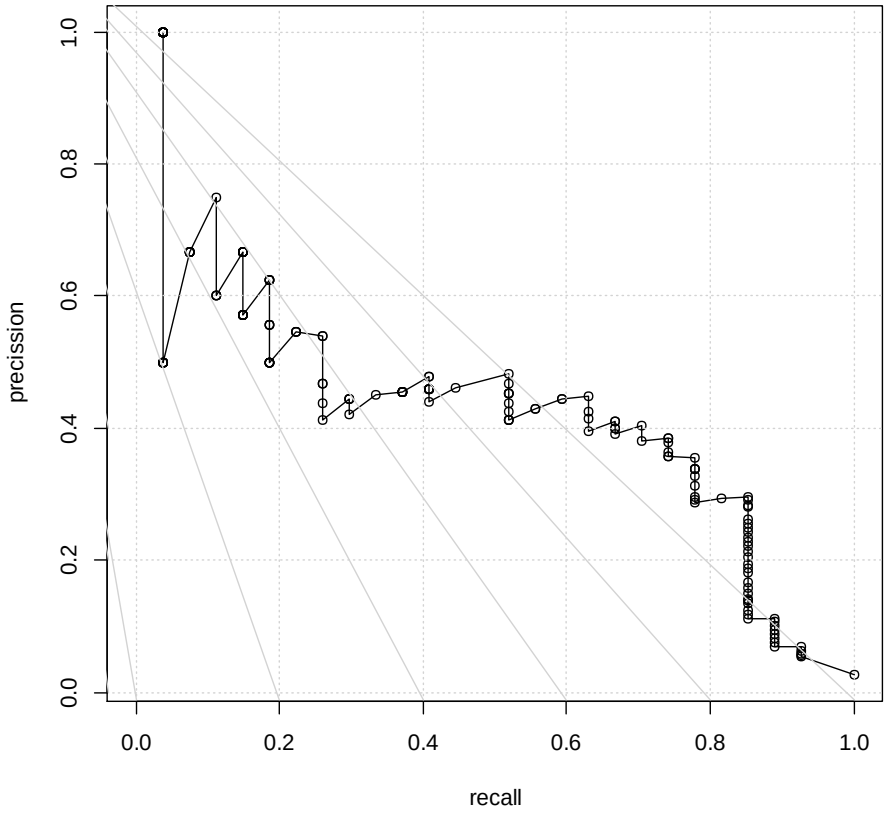
- Wyznaczamy cos(α) dla przykładowego query

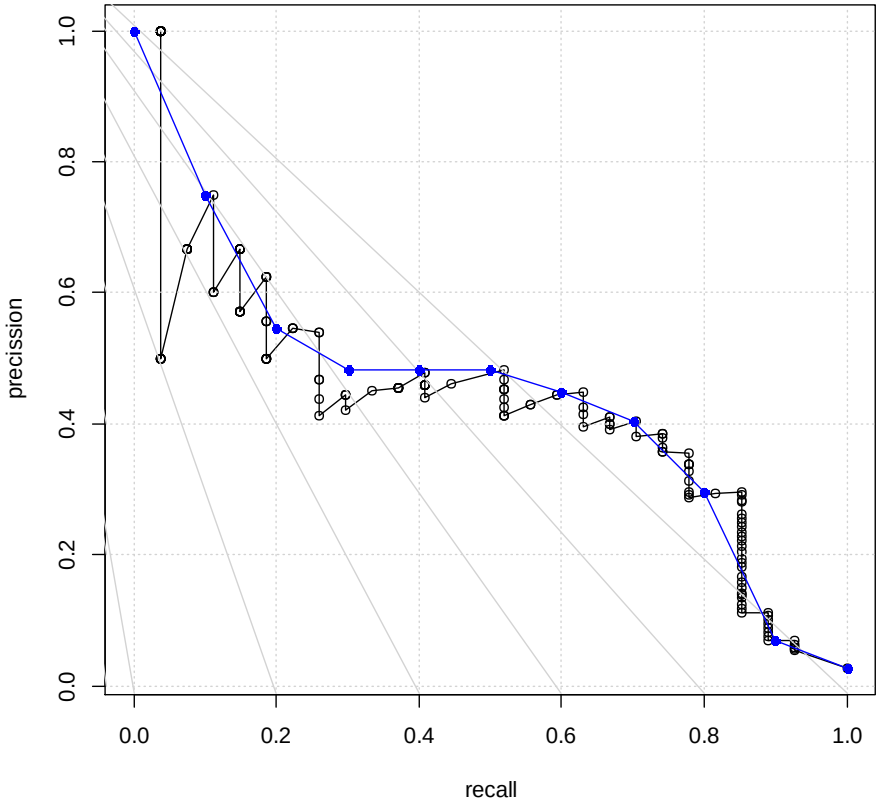
```
> mtx <- TDM2mtx(tdm, ldoc, lquery)
> Q.number <- 9
> cos.alpha <-  as.vector(1 - proxy::dist(t(mtx$tdm.mtx.query[, Q.number]), t(mtx$tdm.
mtx), method = 'cosine'))
> sort(cos.alpha, decreasing = TRUE)
   [1] 0.18258 0.15309 0.15207 0.14245 0.14054 0.12621 0.11894 0.11613 0.11388
  [10] 0.10750 0.10596 0.10515 0.10428 0.10367 0.10321 0.10159 0.09890 0.09536
  [19] 0.09213 0.08778 0.08379 0.08349 0.08223 0.08140 0.08060 0.08042 0.07936
  [28] 0.07810 0.07422 0.07414 0.07360 0.07321 0.07094 0.07000 0.06780 0.06765
  [37] 0.06758 0.06690 0.06629 0.06319 0.06216 0.06016 0.05934 0.05905 0.05849
  [46] 0.05829 0.05625 0.05499 0.05460 0.05369 0.05153 0.05107 0.05062 0.04974
  [55] 0.04968 0.04967 0.04899 0.04749 0.04749 0.04717 0.04648 0.04616 0.04597
  [64] 0.04578 0.04546 0.04501 0.04395 0.04394 0.04311 0.04311 0.04272 0.04220
  [73] 0.04111 0.04103 0.04014 0.03990 0.03969 0.03823 0.03681 0.03666 0.03644
  [82] 0.03640 0.03549 0.03461 0.03448 0.03417 0.03344 0.03260 0.03155 0.03130
  [91] 0.03125 0.03062 0.03053 0.03012 0.02953 0.02948 0.02913 0.02896 0.02893
```

Precision, Recall – baza MEDLINE

Query 9: „the use of induced hypothermia in heart surgery, neurosurgery, head injuries and infectious diseases.”
Zgodne (relevant) dokumenty:

31, 53, 56, 57, 64, 83, 84, 89, 124, 125, 126, 192, 252, 253, 267, 268, 269, 270, 271, 272, 273, 409, 412, 415, 420, 421, 422

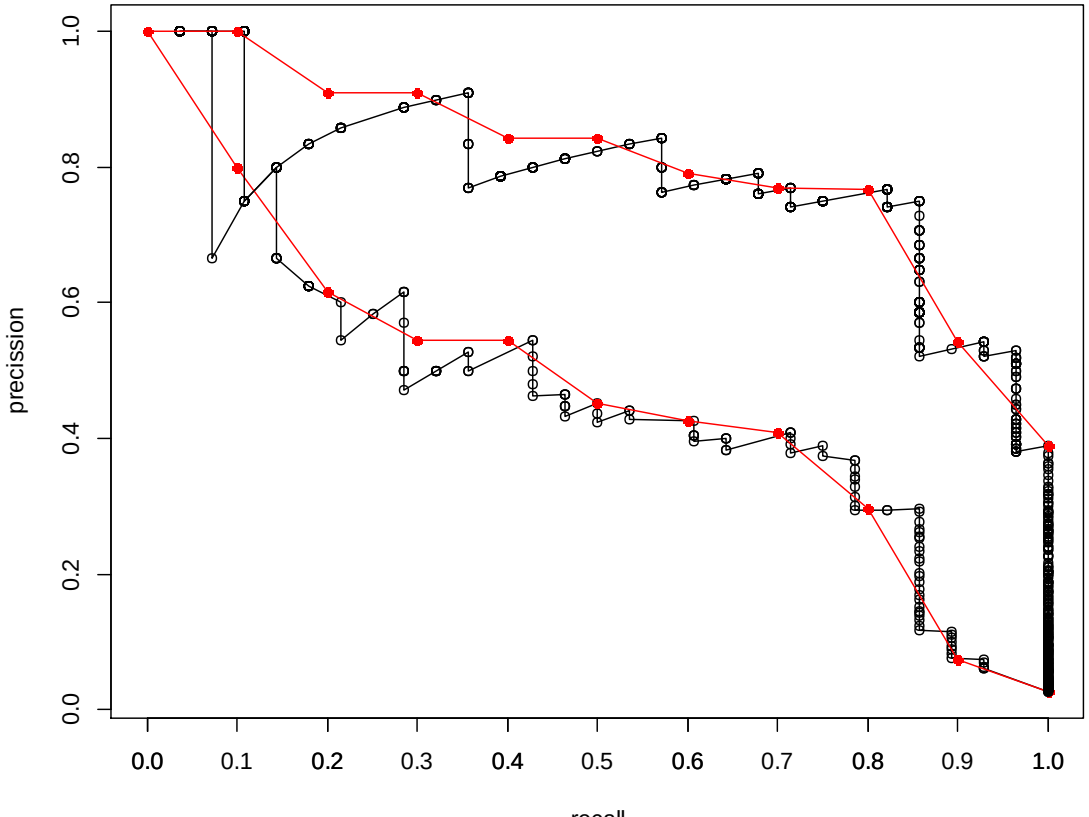




Query 9: „the use of induced hypothermia in heart surgery, neurosurgery, head injuries and infectious diseases.”
Zgodne (relevant) dokumenty:

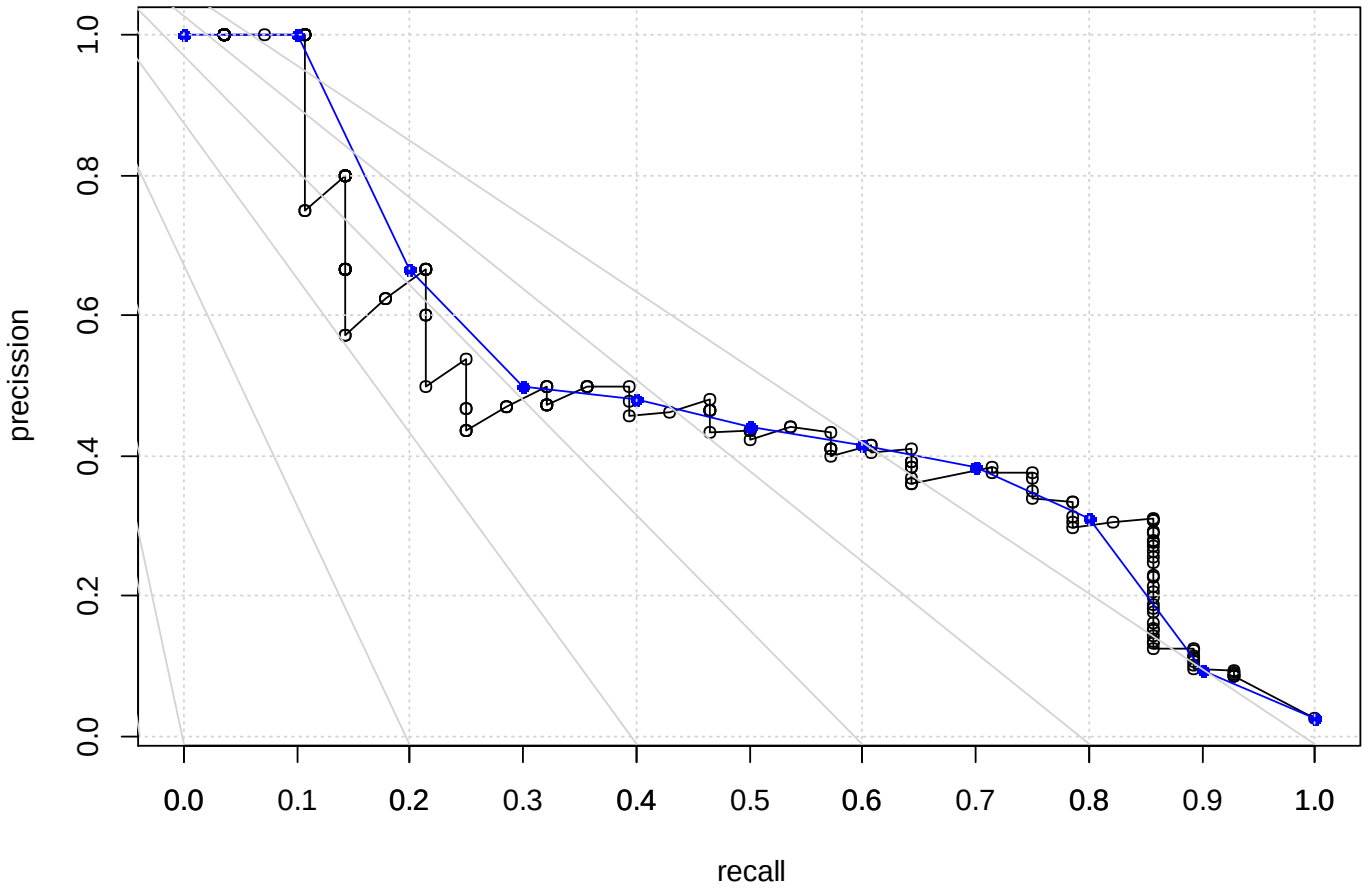
31, 53, 56, 57, 64, 83, 84, 89, 124, 125, 126, 192, 252, 253, 267, 268, 269, 270, 271, 272, 273, 409, 412, 415, 420, 421, 422

Porównanie z redukcją Low-Rank Approximation (k=100)



- Krzywa Precission-Recall

```
> Q <- Q9.REL
> wynik <- PrecRecall(Q, cos.alpha, 0.001)
> wynik.interp <- PrecRecall.Interp(wynik, 0.001)
> plot(wynik[4,], wynik[5,], type="o", xlab="recall", ylab="precission", col="black")
> axis(side=1, at=seq(0, 1, 0.1))
> points(wynik.interp[1,], wynik.interp[2,], pch=16, col="blue", type="o")
> grid()
```



Omówienie skryptu TDM_rank_k.R

- Załadowanie minimalistycznego przykładu

```
> doc = c(
+   "The Google matrix P is a model of the Internet",
+   "P ij is nonzero if there is a link from Web page j to i",
+   "The Google matrix rank is used to rank all Web pages",
+   "The ranking is done by solving a matrix eigenvalue problem",
+   "England dropped out of the top 10 in the FIFA ranking"
+ )
> # Jak wyżej ale po ręcznym usunięciu malo znaczących słów.
> doc = c(
+   "Google matrix Internet",
+   "link Web page",
+   "Google matrix rank Web pages",
+   "ranking matrix eigenvalue",
+   "England FIFA ranking"
+ )
> query <- c("ranking web pages")
> (ldoc <- length(doc))
[1] 5
> (lquery <- length(query))
[1] 1
>
```

- Zbudowanie obiektu TDM

```
> tdm <-
+   create.TDM(
+     doc = doc,
+     query = query,
+     weighting = "TfIdf.Norm.T",
+     spec = "bnc",
+     min.words.length = 3
+   )
> inspect(tdm)
<<TermDocumentMatrix (terms: 10, documents: 6)>>
Non-/sparse entries: 20/40
Sparsity           : 67%
Maximal term length: 9
Weighting          : term frequency - inverse document frequency (normalized) (tf-idf)
Sample             :
          Docs
Terms            1     2     3     4     5     6
  eigenvalu  0.000 0.000 0.000 0.862 0.000 0.000
  england    0.000 0.000 0.000 0.000 0.862 0.000
  fifa       0.000 0.000 0.000 0.000 0.862 0.000
  googl      0.528 0.000 0.317 0.000 0.000 0.000
  internet   0.862 0.000 0.000 0.000 0.000 0.000
  link       0.000 0.862 0.000 0.000 0.000 0.000
  matrix     0.333 0.000 0.200 0.333 0.000 0.000
  page       0.000 0.333 0.200 0.000 0.000 0.333
  rank       0.000 0.000 0.117 0.195 0.195 0.195
  web        0.000 0.333 0.200 0.000 0.000 0.333
>
```

ostatnia kolumna to query

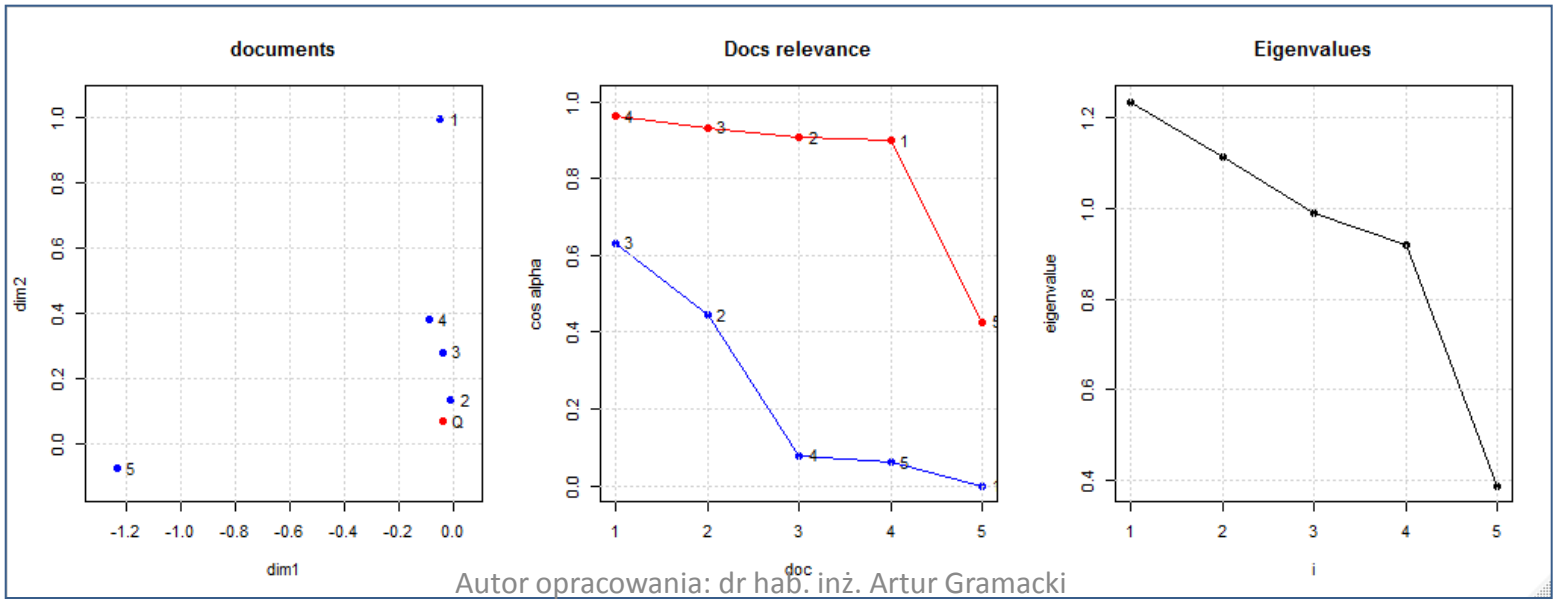
- Utworzenie k-rank approximation, k = 2

```
> out <-
+   TDM.Rank.k(
+     tdm = tdm,
+     ldoc = ldoc,
+     lquery = lquery,
+     k = 2,
+     q.number = 1,
+     plot = TRUE,
+     cex = 1
+   )
> out$cos.alpha
[1] 0.0000 0.4435 0.6325 0.0789 0.0604
> out$cos.alpha.k
[1] 0.898 0.906 0.932 0.962 0.427
> out$A.k
         [,1]     [,2]   [,3]   [,4]     [,5]
 [1,]  0.2656  0.03631 0.0763 0.1048  0.04094
 [2,] -0.0189 -0.00087 0.0113 0.0409  0.85914
 [3,] -0.0189 -0.00087 0.0113 0.0409  0.85914
 [4,]  0.4955  0.06758 0.1407 0.1909 -0.00743
 [5,]  0.6906  0.09418 0.1959 0.2656 -0.01890
 [6,]  0.0942  0.01285 0.0267 0.0363 -0.00087
 [7,]  0.4154  0.05669 0.1183 0.1610  0.01115
 [8,]  0.0819  0.01118 0.0233 0.0318  0.00228
 [9,]  0.0824  0.01165 0.0274 0.0433  0.20522
[10,]  0.0819  0.01118 0.0233 0.0318  0.00228
>
```

- Wyznaczenia współczynnika utraty informacji

```
> # Obliczenie współczynnika "zmniejszenia informacji".
> (n1 <- norm(out$tdm - out$A.k, type = "F"))
[1] 1.41
> (n2 <- norm(out$tdm, type = "F"))
[1] 2.18
> n1 / n2
[1] 0.646
>
```

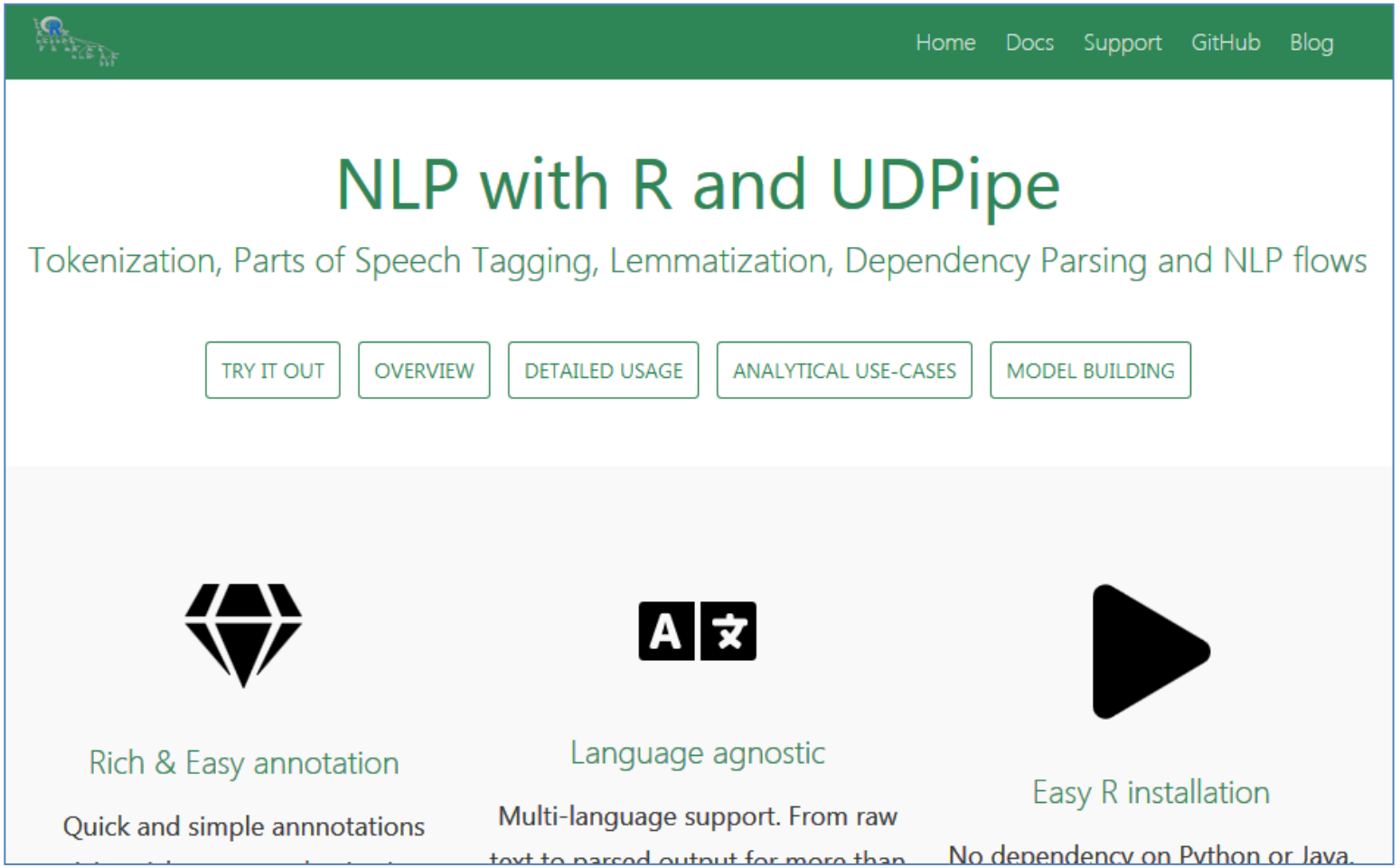
- Wykresy podsumowujące
  - parametr plot = TRUE w funkcji TDM.Rank.k



- W skrypcie TDM_rank_k.R jest też przykład dla zbioru MEDLINE
  - uwaga: obliczenia SVD dla takiego dużego zbioru mogą trwać kilka minut
- W skrypcie TDM_rank_k.R jest też kolejny przykład dla zbioru 122 tytułów artykułów z 3 dziedzin
  - patrz wyżej

Omówienie skryptu udpipe.R

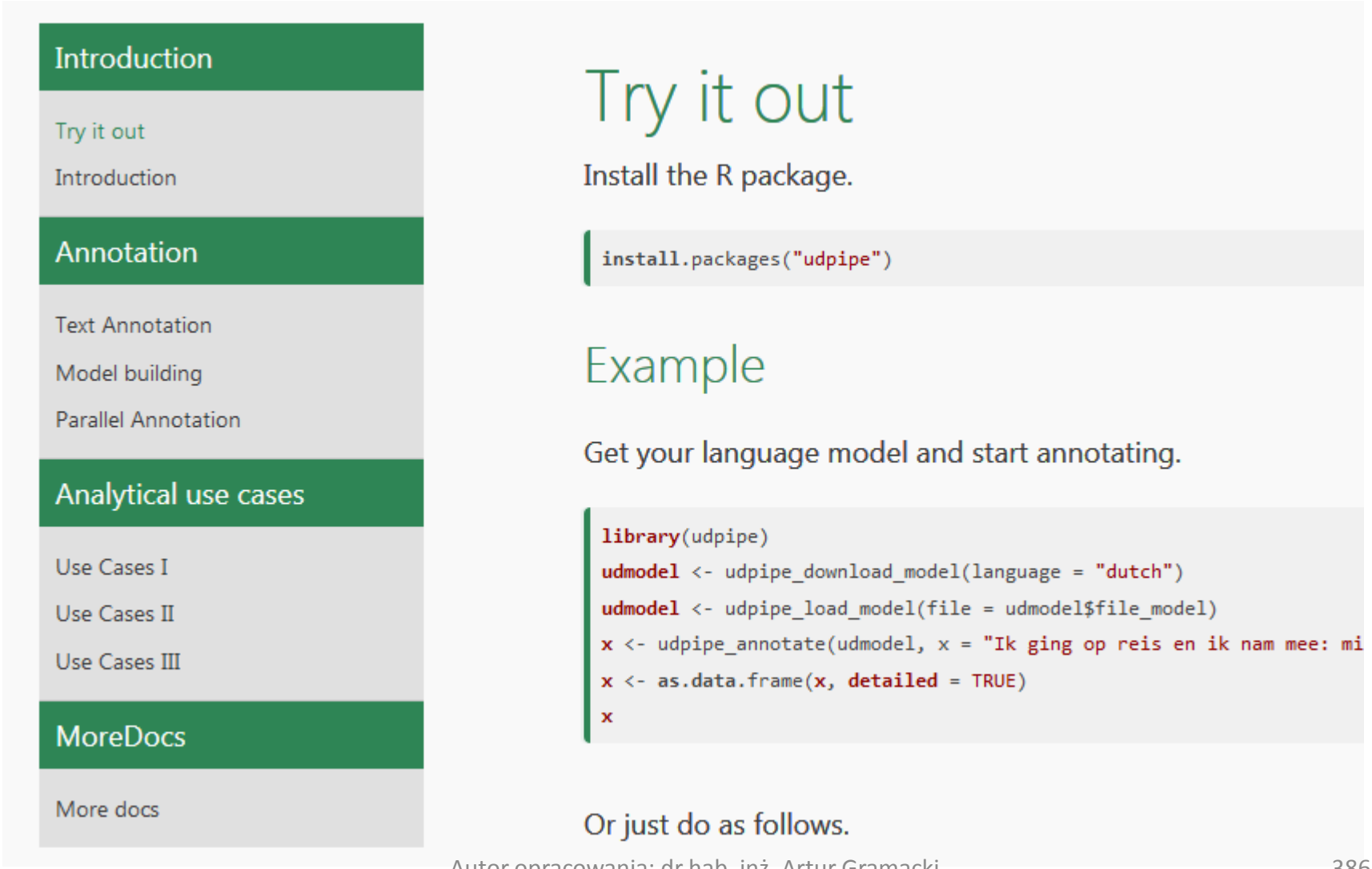
https://bnosac.github.io/udpipe/en/



UDPipe

- PL:
  Pozwala on w szybki i kompaktowy sposób podzielić tekst na tokeny, taguje części mowy, dokonuje lematyzacji, a także ustala relacje głębokie między elementami zdania, tworzy diagramy NLP
- ENG:
  Tokenization, Parts of Speech Tagging, Lemmatization, Dependency Parsing of Raw Text and NLP flows

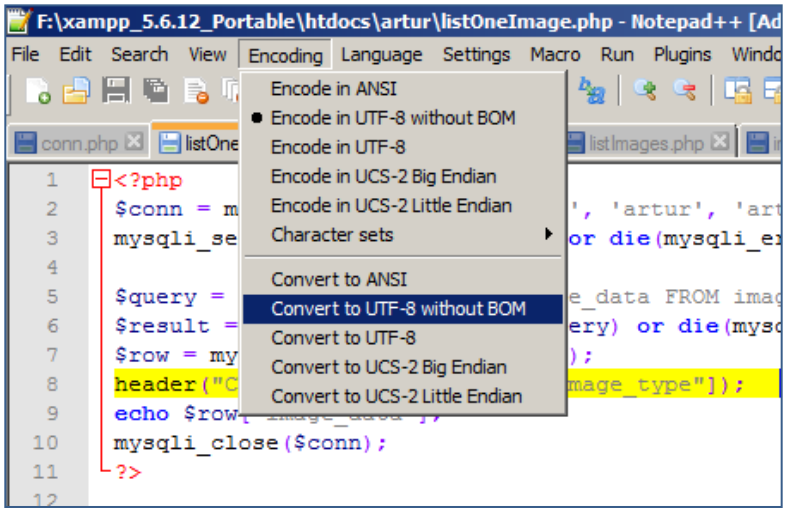
- Firmowy tutorial
  - niemal zawsze warto od tego zacząć poznawanie systemu



- Wspierane języki
  - udpipe models, re-trained models
  - 64 języki, oparte na 97 treebanks

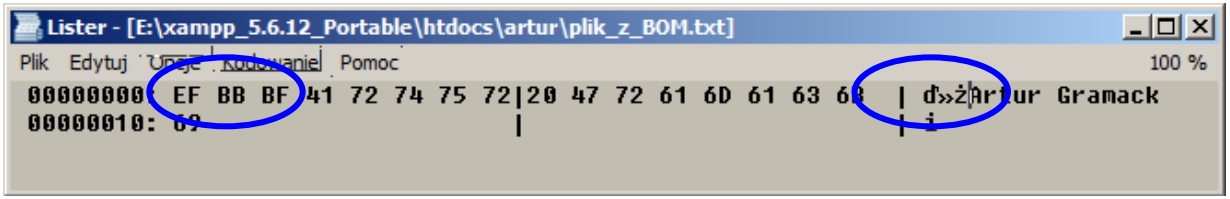
afrikaans-afribooms, ancient_greek-perseus, ancient_greek-proiel, arabic-padt, armenian-armtdp, basque-bdt, belarusian-hse, bulgarian-btb, buryat-bdt, catalan-ancora, chinese-gsd, classical_chinese-kyoto, coptic-scriptorium, croatian-set, czech-cac, czech-cltt, czech-fictree, czech-pdt, danish-ddt, dutch-alpino, dutch-lassysmall, english-ewt, english-gum, english-lines, english-partut, estonian-edt, estonian-ewt, finnish-ftb, finnish-tdt, french-gsd, french-partut, french-sequoia, french-spoken, galician-ctg, galician-treegal, german-gsd, gothic-proiel, greek-gdt, hebrew-htb, hindi-hdtb, hungarian-szeged, indonesian-gsd, irish-idt, italian-isdt, italian-partut, italian-postwita, italian-vit, japanese-gsd, kazakh-ktb, korean-gsd, korean-kaist, kurmanji-mg, latin-ittb, latin-perseus, latin-proiel, latvian-lvtb, lithuanian-alksnis, lithuanian-hse, maltese-mudt, marathi-ufal, north_sami-giella, norwegian-bokmaal, norwegian-nynorsk, norwegian-nynorsklia, old_church_slavonic-proiel, old_french-srcmf, old_russian-torot, persian-seraji, polish-lfg, polish-pdb, polish-sz, portuguese-bosque, portuguese-br, portuguese-gsd, romanian-nonstandard, romanian-rrt, russian-gsd, russian-syntagrus, russian-taiga, sanskrit-ufal, serbian-set, slovak-snk, slovenian-ssj, slovenian-sst, spanish-ancora, spanish-gsd, swedish-lines, swedish-talbanken, tamil-ttb, telugu-mtg, turkish-imst, ukrainian-iu, upper_sorbian-ufal, urdu-udtb, uyghur-udt, vietnamese-vtb, wolof-wtb.

- Teksty muszą być w UTF-8, bez tzw. „BOM-a”

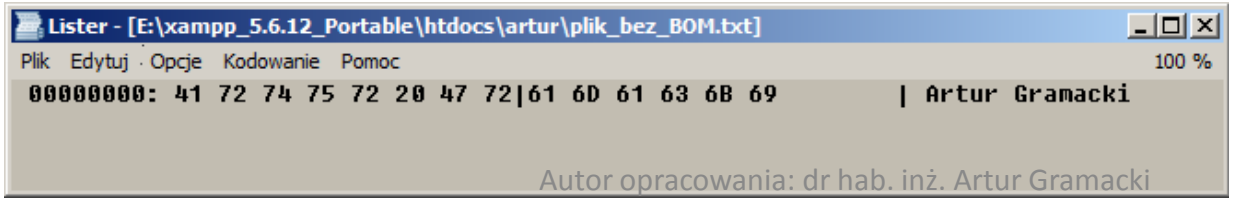


BOM (ang. **B**yte **O**rder **M**ark), znacznik kolejności bajtów
https://pl.wikipedia.org/wiki/BOM_%28informatyka%29

plik z BOM



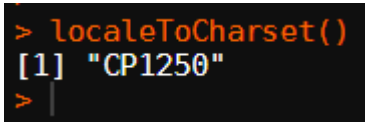
plik bez BOM



- Fragment dokumentacji na temat kodowania

A small note on encodings

Mark that it is important that the x argument to udpipe_annotate is in UTF-8 encoding.
You can check the encoding of your text with Encoding('your text').
You can convert your text to UTF-8, using standard R utilities:
as in iconv('your text', from = 'latin1', to = 'UTF-8') where you replace the 'from' part with whichever encoding you have your text in, possible your computers default as defined in localeToCharset().
So annotation would look something like this if your text is not already in UTF-8 encoding:



- udpipe_annotate(udmodel, x = iconv('your text', to = 'UTF-8'))
  if your text is in the encoding of the current locale of your computer.

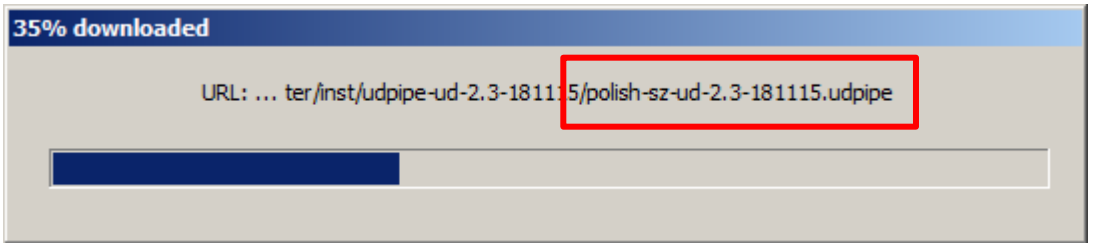
- udpipe_annotate(udmodel, x = iconv('your text', from = 'latin1', to = 'UTF-8'))
  if your text is in latin1 encoding.

- udpipe_annotate(udmodel, x = iconv('your text', from = 'CP949', to = 'UTF-8'))
  if your text is in CP949 encoding.

- Zaczynamy od załadowania biblioteki i zaciągnięcia modelu dla języka polskiego
  - Uwaga, na dzień dzisiejszy, 13-01-2020, gdy wpisujemy wersję najbardziej aktualną 2.4 to pojawia się błąd. Dlatego używamy 2.3

```
library(tm)
library(udpipe)
```

```
> pl_model <- udpipe_download_model(language = 'polish', udpipe_model_repo = "jwijffels/u
dpipe.models.ud.2.3")
Downloading udpipe model from https://raw.githubusercontent.com/jwijffels/udpipe.models.u
d.2.3/master/inst/udpipe-ud-2.3-181115/polish-sz-ud-2.3-181115.udpipe to X:/ag/__zajecia_
_/_Eksp_Zas_Internet/__dla_studentów/cwicz_4/polish-sz-ud-2.3-181115.udpipe
Visit https://github.com/jwijffels/udpipe.models.ud.2.3 for model license details
trying URL 'https://raw.githubusercontent.com/jwijffels/udpipe.models.ud.2.3/master/inst/
udpipe-ud-2.3-181115/polish-sz-ud-2.3-181115.udpipe'
Content type 'application/octet-stream' length 12717929 bytes (12.1 MB)
downloaded 12.1 MB
```



- Model UDPipe

```
> pl_model_path <- pl_model[1,2]
> loadedModel <- udpipe_load_model(file = basename(pl_model_path))
> loadedModel
$file
[1] "polish-sz-ud-2.3-181115.udpipe"

$model
<pointer: 0x000000001ed453e0>

attr(,"class")
[1] "udpipe_model"
> 
```

- Przykładowy tekst

```
> # wczytanie dokumentu z pliku.
> # Zwrócić uwagę, żeby plik był zapisany _BEZ_ tzw. "BOM".
> file <- paste(filesPath, "inwokacja_utf8.txt", sep="")
> con <- file(description=file, open="r")
> (doc <- readLines(con, encoding = "UTF-8"))
[1] "Litwo, Ojczyzno moja! ty jesteś jak zdrowie; Ile cię trzeba cenić, ten tylko się
 dowie, Kto cię stracił. Dziś piękność twą w całej ozdobie Widzę i opisuję, bo tęsknię
 po tobie."
> close(con)
> # Sprawdzamy, czy dane są w UTF-8. "unknown" oznacza, że w badanym
> # tekście nie ma znaków spoza podstawowego zbioru ASCII i wówczas
> # sprawdzanie kodowania nie ma za bardzo sensu.
> Encoding(doc)
[1] "UTF-8"
```

- Wczytanie polskiej stoplisty

```
> # wczytanie stopwords listy z pliku
> file <- paste(filesPath, "stopwords.txt", sep="")
> con <- file(description=file, open="r")
> stopwords <- readLines(con, encoding = "UTF-8")
> close(con)
> stopwords
  [1] "a"          "aby"        "ach"          "acz"          "aczkolwiek"
  [6] "aj"         "albo"       "ale"          "alez"         "ależ"
 [11] "ani"        "az"         "aż"           "bardziej"     "bardzo"
 [16] "beda"       "bedzie"     "bez"          "deda"         "będą"
 [21] "bede"       "będę"       "będzie"       "bo"           "bowiem"
 [26] "by"         "byc"        "być"          "byl"          "byla"
 [31] "byli"       "bylo"       "byly"         "był"          "była"
 [36] "było"       "były"       "bynajmniej"   "cala"         "cali"
 [41] "caly"       "cała"       "cały"         "ci"           "cie"
 [46] "ciebie"     "cię"        "co"           "cokolwiek"    "cos"
 [51] "coś"        "czasami"    "czasem"       "czemu"        "czy"
 [56] "czyli"      "daleko"     "dla"          "dlaczego"     "dlatego"
 [61] "do"         "dobrze"     "dokad"        "dokąd"        "dosc"
 [66] "dość"       "duzo"       "dużo"         "dwa"          "dwaj"
```

```
[306] "w"          "wam"        "wami"         "was"          "wasz"
[311] "wasza"      "wasze"      "we"           "według"       "wiele"
[316] "wielu"      "więc"       "więcej"       "wlasnie"      "właśnie"
[321] "wszyscy"    "wszystkich" "wszystkie"    "wszystkim"    "wszystko"
[326] "wtedy"      "wy"         "z"            "za"           "zaden"
[331] "zadna"      "zadne"      "zadnych"      "zapewne"      "zawsze"
[336] "ze"         "zeby"       "zeznowu"      "zł"           "znow"
[341] "znowu"      "znów"       "zostal"       "został"       "żaden"
[346] "żadna"      "żadne"      "żadnych"      "że"           "żeby"
```

- Preprocessing

```
> # Ew. preprocessing
> doc
[1] "Litwo, Ojczyzno moja! ty jesteś jak zdrowie; Ile cię trzeba cenić, ten tylko się
 dowie, Kto cię stracił. Dziś piękność twą w całej ozdobie Widzę i opisuję, bo tęsknię
 po tobie."
> (doc <- tolower(doc))
[1] "litwo, ojczyzno moja! ty jesteś jak zdrowie; ile cię trzeba cenić, ten tylko się
 dowie, kto cię stracił. dziś piękność twą w całej ozdobie widzę i opisuję, bo tęsknię
 po tobie."
> (doc <- removeNumbers(doc))
[1] "litwo, ojczyzno moja! ty jesteś jak zdrowie; ile cię trzeba cenić, ten tylko się
 dowie, kto cię stracił. dziś piękność twą w całej ozdobie widzę i opisuję, bo tęsknię
 po tobie."
> (doc <- removePunctuation(doc))
[1] "litwo ojczyzno moja ty jesteś jak zdrowie ile cię trzeba cenić ten tylko się dowi
e kto cię stracił dziś piękność twą w całej ozdobie widzę i opisuję bo tęsknię po tobi
e"
> (doc <- removeWords(doc, stopwords))
[1] "litwo ojczyzno   jesteś  zdrowie    cenić    dowie   stracił  piękność twą  całej
 ozdobie widzę  opisuję  tęsknię  "
> (doc <- stripWhitespace(doc))
[1] "litwo ojczyzno jesteś zdrowie cenić dowie stracił piękność twą całej ozdobie widz
ę opisuję tęsknię "
```

- Lematyzacja i określenie części mowy

```
> x <- udpipe_annotate(loadedModel, x=doc)
> # Zapisanie wyniku w formacie CONLL-U
> # http://universaldependencies.org/format.html
> cat(x$conllu, file = "myannotation.conllu")
> x <- as.data.frame(x, detailed = TRUE)
> out <- cbind(x$token, x$lemma, x$upos)
> colnames(out) <- c("token", "lemma", "upos")
> out
      token       lemma       upos
 [1,] "litwo"     "litwo"     "NOUN"
 [2,] "ojczyzno"  "ojczyzny"  "ADJ"
 [3,] "jesteś"    "być"       "AUX"
 [4,] "zdrowie"   "zdrowie"   "NOUN"
 [5,] "cenić"     "cenić"     "VERB"
 [6,] "dowie"     "d"         "NOUN"
 [7,] "stracił"   "stracić"   "VERB"
 [8,] "piękność"  "piękność"  "NOUN"
 [9,] "twą"       "twój"      "DET"
[10,] "całej"     "cały"      "ADJ"
[11,] "ozdobie"   "ozdób"     "NUM"
[12,] "widzę"     "widzieć"   "NOUN"
[13,] "opisuję"   "opisuja"   "VERB"
[14,] "tęsknię"   "tęsknia"   "NOUN"
>
```

```
# Mały słowniczek:
# https://www.ang.pl/gramatyka/czesci-mowy
# ---------------
# verb (czasownik)
# noun (rzeczowniki)
# adjective (przymiotnik)
# adverb (przysłówek)
# pronoun (zaimek)
# article (przedimek)
# preposition (przyimek)
# conjunction (spójnik)
# numeral (liczebnik)
# interjection (wykrzyknik, wtrącenie)
```

- Eksperymenty z dłuższym tekstem

```
ROZDZIAŁ PIERWSZY
Chłopiec, który przeżył

Państwo Dursleyowie spod numeru czwartego przy Privet Drive mogli z dumą twierdzić, że są całkowicie normalni, chwała
Bogu. Byli ostatnimi ludźmi, których można by posądzić o udział w czymś dziwnym lub tajemniczym, bo po prostu nie
wierzyli w takie bzdury.

Pan Dursley był dyrektorem firmy Grunnings produkującej świdry. Był to rosły, otyły mężczyzna pozbawiony szyi, za to
wyposażony w wielkie wąsy. Natomiast pani Dursley była drobną blondynką i miała szyję dwukrotnie dłuższą od normalnej,
co bardzo jej pomagało w życiu, ponieważ większość dnia spędzała na podglądaniu sąsiadów. Syn Dursleyów miał na imię
Dudley, a rodzice uważali go za najwspanialszego chłopca na świecie.

Dursleyowie mieli wszystko, czego dusza zapragnie, ale mieli też swoją tajemnicę i nic nie budziło w nich większego
przerażenia, jak myśl, że może zostać odkryta. Uważali, że znaleźliby się w sytuacji nie do zniesienia, gdyby ktoś
dowiedział się o istnieniu Potterów. Pani Potter była siostrą pani Dursley, ale nie widziały się od wielu lat. Prawdę
mówiąc, pani Dursley udawała, że w ogóle nie ma siostry, ponieważ pani Potter i jej żałosny mąż byli ludźmi całkowicie
innego rodzaju. Dursleyowie wzdrygali się na samą myśl, co by powiedzieli sąsiedzi, gdyby Potterowie pojawili się na
ich ulicy. Oczywiście wiedzieli, że Potterowie też mają synka, ale nigdy nie widzieli go na oczy i z całą pewnością
nie chcieli go nigdy oglądać. Ten chłopiec był jeszcze jednym powodem, by Dursleyowie trzymali się jak najdalej od
Potterów; nie życzyli sobie, by Dudley przebywał w towarzystwie takiego dziecka.

Kiedy Dursleyowie obudzili się rano w pewien nudny, szary wtorek, od którego zaczyna się nasza opowieść, w
zachmurzonym niebie nie było niczego, co by zapowiadało owe dziwne i tajemnicze rzeczy, które miały się wkrótce
wydarzyć w całym kraju. Pan Dursley nucił coś pod nosem, zawiązując swój najnudniejszy krawat, a pani Dursley wyrwała
się na chwilę z domu na plotki, gdy tylko udało się jej wepchnąć wrzeszczącego Dudleya do dziecinnego krzesła na
wysokich nogach.
```

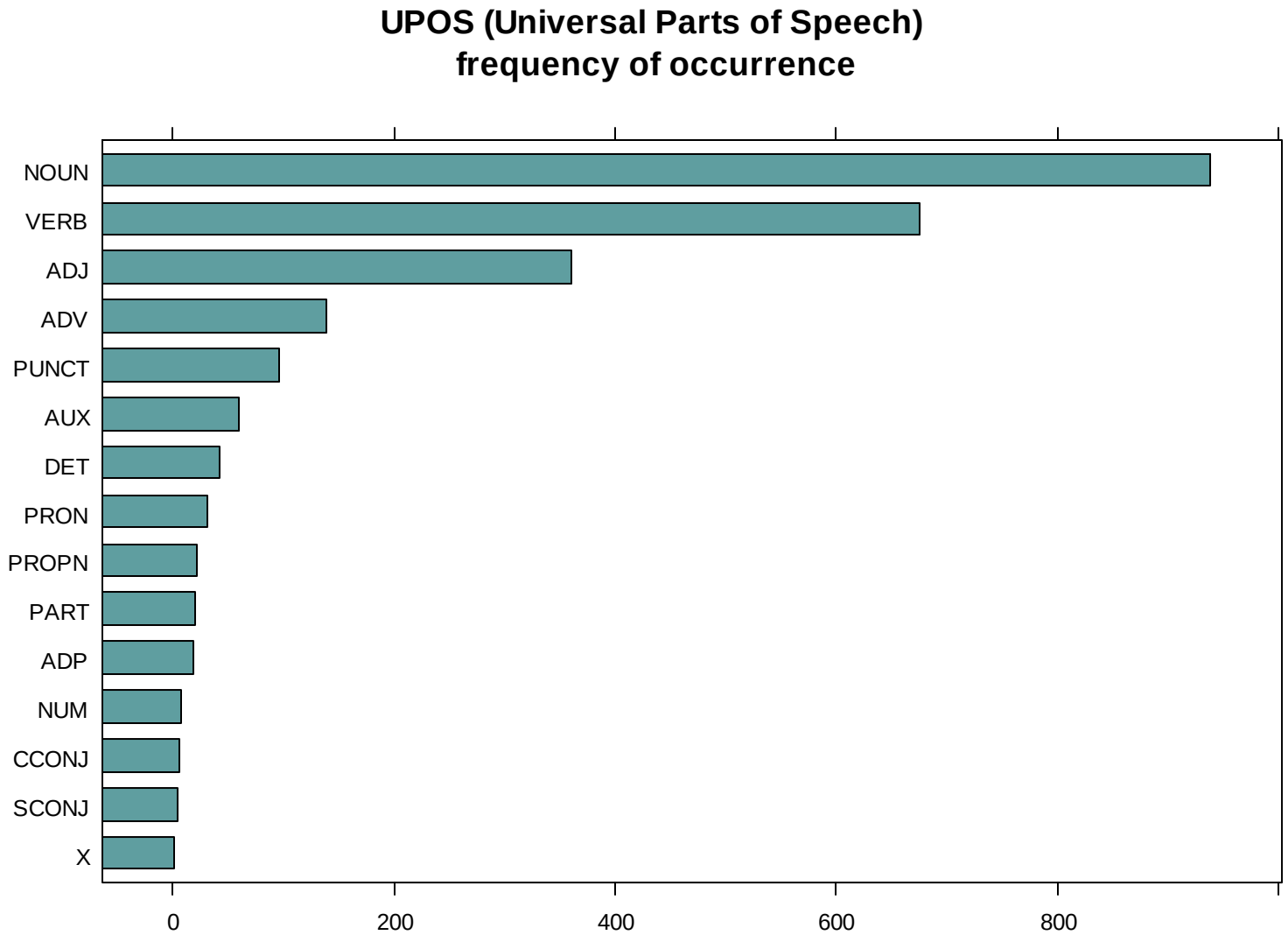
- Lematyzacja i określenie części mowy (pokazano tylko mały fragment)

[313,]	"wielkie"	"wielki"	"ADJ"
[314,]	"zamówienie"	"zamówienie"	"NOUN"
[315,]	"świdry"	"świdra"	"NOUN"
[316,]	"miał"	"mieć"	"VERB"
[317,]	"otrzymać"	"otrzymać"	"VERB"
[318,]	"skraju"	"skraj"	"NOUN"
[319,]	"miasta"	"miasto"	"NOUN"
[320,]	"zmuszony"	"zmuszony"	"ADJ"
[321,]	"zapomnienia"	"zapomnienie"	"NOUN"
[322,]	"świdrach"	"świdra"	"NOUN"
[323,]	"utkwił"	"utkwić"	"VERB"
[324,]	"normalnym"	"normalny"	"ADJ"
[325,]	"porannym"	"poranny"	"ADJ"
[326,]	"korku"	"korek"	"NOUN"
[327,]	"ulicznym"	"uliczny"	"ADJ"
[328,]	"mógł"	"móc"	"VERB"
[329,]	"zauważyć"	"zauważyć"	"VERB"
[330,]	"naokoło"	"naokoło"	"PART"
[331,]	"mnóstwo"	"mnóstwo"	"NOUN"
[332,]	"dziwacznie"	"dziwacznie"	"ADV"
[333,]	"ubranych"	"ubrany"	"ADJ"

- Przykładowe analizy, najczęściej występujące formy gramatyczne

```
> # Basic frequency statistics
> # In most languages, nouns (NOUN) are the most common types of words,
> # next to verbs (VERB) and these are the most relevant for analytical purposes,
> # next to the adjectives (ADJ) and proper nouns (PROPN).
> # For a detailed list of all POS tags: visit http://universaldependencies.org/u/pos/
index.html.
> library(lattice)
> stats <- txt_freq(x$upos)
> stats$key <- factor(stats$key, levels = rev(stats$key))
> barchart(key ~ freq, data = stats, col = "cadetblue",
+          main = "UPOS (Universal Parts of Speech)\n frequency of occurrence",
+          xlab = "Freq")
>
```

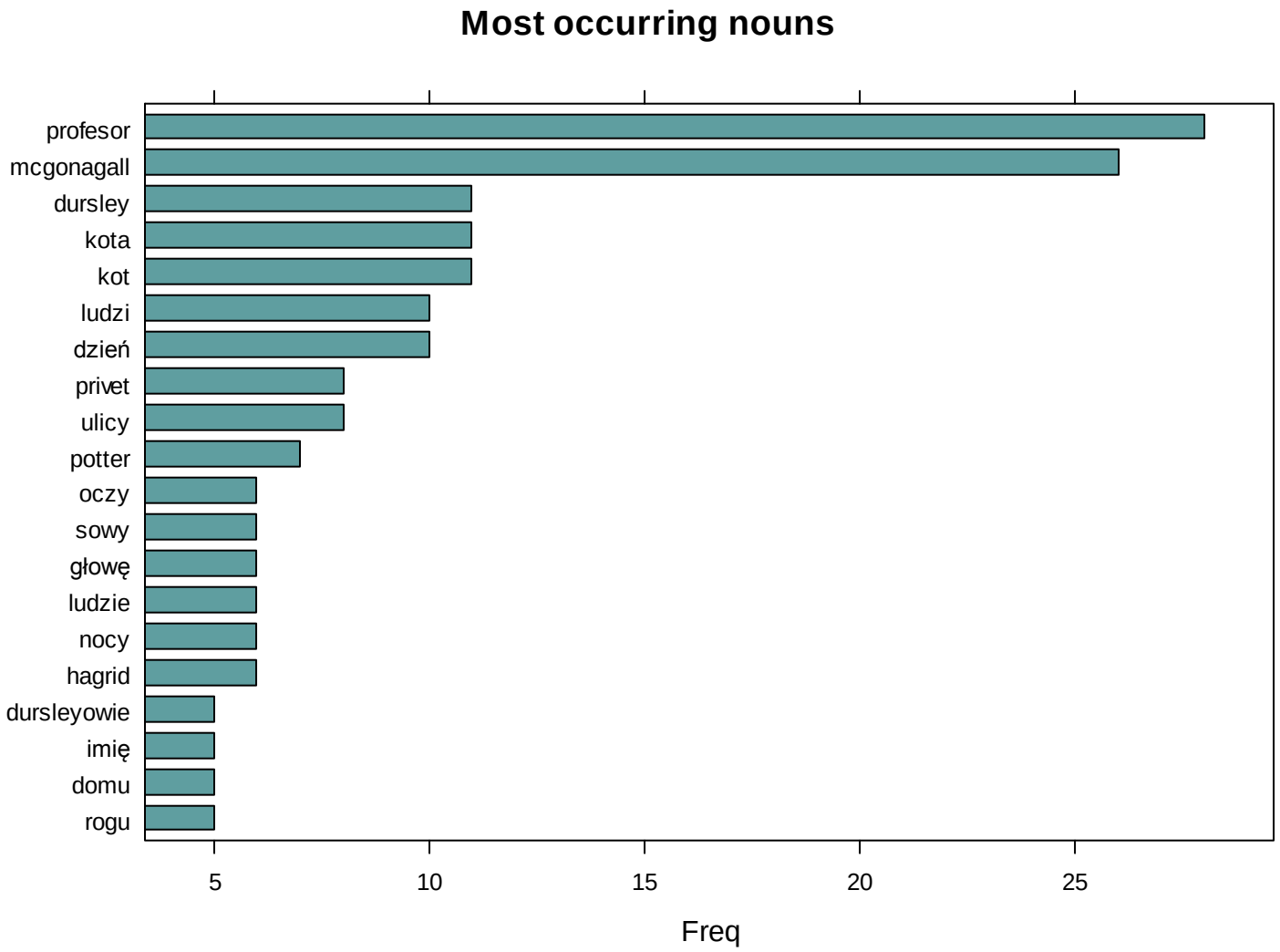
- Przykładowe analizy, najczęściej występujące formy gramatyczne



- Przykładowe analizy, najczęściej pojawiające się rzeczowniki

```
> stats <- subset(x, upos %in% c("NOUN"))
> stats <- txt_freq(stats$token)
> stats$key <- factor(stats$key, levels = rev(stats$key))
> barchart(key ~ freq, data = head(stats, 20), col = "cadetblue",
+          main = "Most occurring nouns", xlab = "Freq")
>
```

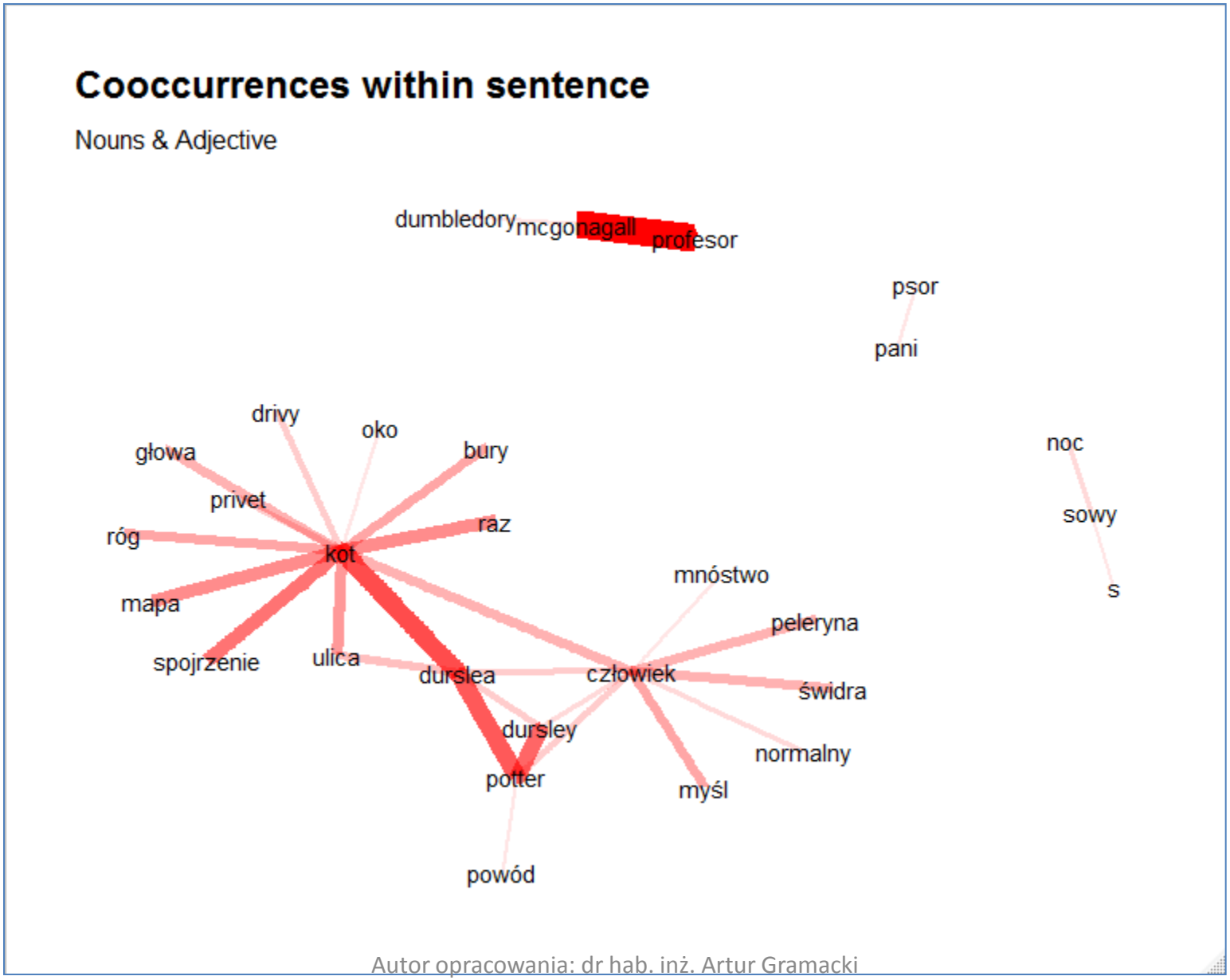
- Przykładowe analizy, najczęściej pojawiające się rzeczowniki



- Przykładowe analizy, współwystępowanie słów

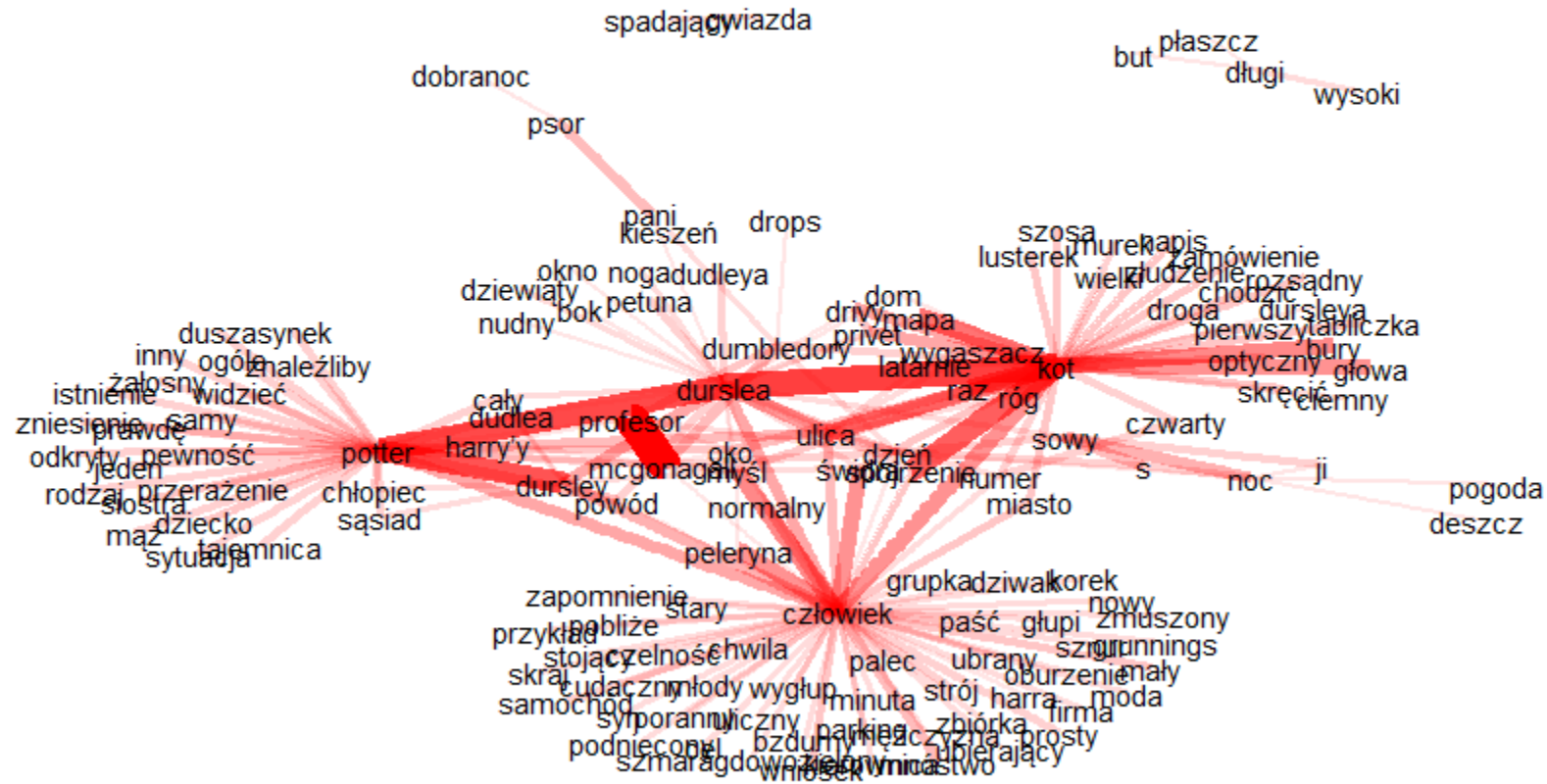
```
> # Co-occurrences
> cooc <- cooccurrence(x = subset(x, upos %in% c("NOUN", "ADJ")),
+                      term = "lemma",
+                      group = c("doc_id", "paragraph_id", "sentence_id"))
> head(cooc)
       term1      term2 cooc
1 mcgonagall   profesor   26
2    durslea        kot   20
3    durslea     potter   19
4    dursley     potter   18
5        kot spojrzenie   17
6        kot       mapa   15
> library(igraph)
> library(ggraph)
> library(ggplot2)
> wordnetwork <- head(cooc, 30)
> wordnetwork <- graph_from_data_frame(wordnetwork)
> ggraph(wordnetwork, layout = "fr") +
+   geom_edge_link(aes(width = cooc, edge_alpha = cooc), edge_colour = "pink") +
+   geom_node_text(aes(label = name), col = "darkgreen", size = 4) +
+   theme_graph(base_family = "Arial Narrow") +
+   theme(legend.position = "none") +
+   labs(title = "Cooccurrences within sentence", subtitle = "Nouns & Adjective")
```

- Przykładowe analizy, współwystępowanie słów

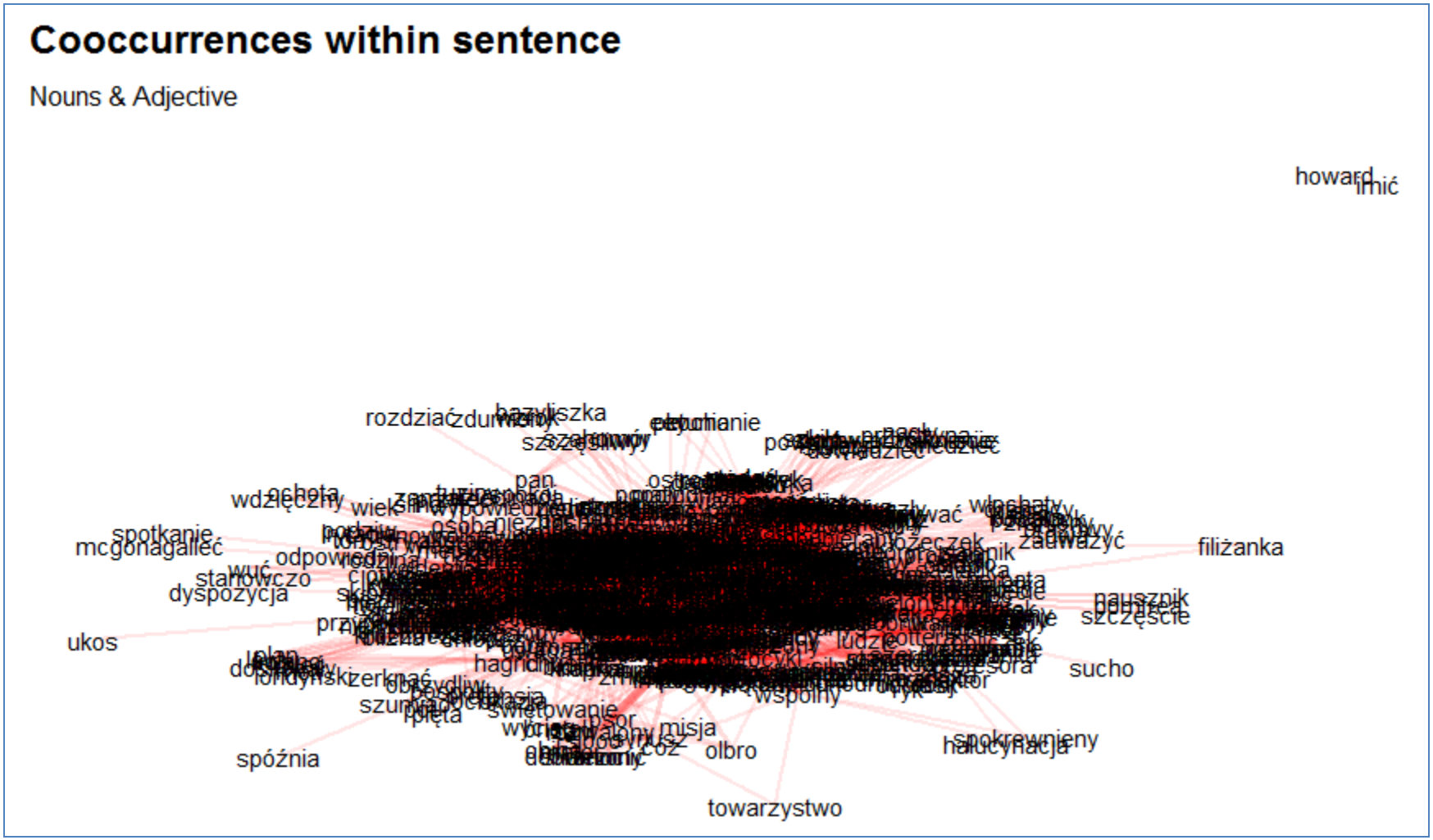


- Przykładowe analizy, współwystępowanie słów

```
wordnetwork <- head(cooc, 200)
```



- Przykładowe analizy, współwystępowanie słów. Lepiej nie „przeginać”



Literatura

- Książki
  - Z. Markov, D.T. Larose: Eksploracja zasobów internetowych: Analiza struktury, zawartości i użytkowania sieci WWW. Warszawa, PWN, 2009
  - D.T. Larose: Odkrywanie wiedzy z danych. Wprowadzenie do eksploracji danych. Warszawa, PWN, 2006
  - M. Bramer: Principles of Data Mining, Fourth Edition. Springer 2020
  - J. Silge, D. Robinson: Text Mining with R, O'Reilly 2017
  - Michael W. Berry, Murray Browne: Understanding Search Engines: Mathematical Modeling and Text Retrieval, Society for Industrial and Applied Mathematics, 2005
- Różne źródła internetowe
  - linki do stron, z których korzystałem umieszczone są bezpośrednio na odpowiednich stronach w prezentacji
- Oprogramowanie
  - dokumentacja systemu R (https://www.r-project.org/)

Dziękuję za uwagę!